PREDICTIVE SIMULATION OF GAS ADSORPTION IN FIXED-BEDS AND LIMITATIONS DUE TO THE ILL-POSED DANCKWERTS BOUNDARY CONDITION

by

JAMES CLINTON KNOX

A DISSERTATION

Submitted in partial fulfillment of the requirements
For the degree of Doctor of Philosophy
in
The Department of Mechanical and Aerospace Engineering
to
The School of Graduate Studies
of
The University of Alabama in Huntsville

HUNTSVILLE, ALABAMA

2016

In presenting this dissertation in partial fulfillment of the requirements for a doctoral degree from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department or the Dean of the School of Graduate studies. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this dissertation.

______________________ 11-10-2016
(student signature) (date)

DISSERTATION APPROVAL FORM

Submitted by James Knox in partial fulfillment of the requirements for the degree of Doctor of Philosophy awarded by the Department of Mechanical Engineering and accepted on behalf of the Faculty of the School of Graduate Studies by the dissertation committee.

We, the undersigned members of the Graduate Faculty of The University of Alabama in Huntsville, certify that we have advised and/or supervised the candidate on the work described in this dissertation. We further certify that we have reviewed the dissertation manuscript and approve it in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

______________________ Committee Chair 11/10/2016
(Date)

James E. Smith Jr. 11/10/2016 Advisor

________________ 11/10/2016

________________ 11/10/2016

________________ for Hugh Coleman 11/10/2016

D.K. Hollingsworth __________ Department Chair

________________ 11/11/16 College Dean

________________ 11/16/16 Graduate Dean

ABSTRACT

The School of Graduate Studies
The University of Alabama in Huntsville

Degree Doctor of Philosophy College/Dept. Engineering/Mechanical and Aerospace Engineering

Name of Candidate James Knox

Title Predictive Simulation of Gas Adsorption in Fixed-beds and Limitations due to the Ill-Posed Danckwerts Boundary Condition

The 1-D axially dispersed plug flow model is a mathematical model widely used for the simulation of adsorption processes. Lumped mass transfer coefficients such as the Glueckauf linear driving force (LDF) term and the axial dispersion coefficient are generally obtained by fitting simulation results to the experimental breakthrough test data. An approach is introduced where these parameters, along with the only free parameter in the energy balance equations, are individually fit to specific test data that isolates the appropriate physics. It is shown that with this approach this model provides excellent simulation results for the CO_2 on zeolite 5A sorbent/sorbate system; however, for the H_2O on zeolite 5A system, non-physical deviations from constant pattern behavior occur when fitting dispersive experimental results with a large axial dispersion coefficient.

A method has also been developed that determines a priori what values of the LDF and axial dispersion terms will result in non-physical simulation results for a specific sorbent/sorbate system when using the one-dimensional axially dispersed plug flow model. A relationship between the steepness of the adsorption equilibrium isotherm as indicated by the distribution factor, the magnitude of the axial dispersion and mass transfer coefficient, and the resulting non-physical behavior is derived. This relationship is intended to provide a guide for avoiding non-physical behavior by limiting the magnitude of the axial dispersion term on the basis of the mass transfer coefficient and distribution factor.

Abstract Approval: Committee Chair ____________________

Department Chair ____________________

Graduate Dean ____________________

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure Page

LIST OF TABLES

LIST OF ACRONYMS

CCS	carbon capture and sequestration
CDRA	carbon dioxide removal assembly
CO_2	carbon dioxide
CPB	constant pattern behavior
CPI	chemical processing industry
DOE	Department of Energy
GC	gas chromatograph
H_2O	water
He	helium
ID	inside diameter
ISS	International Space Station
LDF	linear driving force
LiOH	lithium hydroxide
LPG	liquid petroleum gas
MSMBT	molecular sieve material bench test
NASA	National Aeronautics and Space Administration
OD	outside diameter
PSA	pressure swing adsorption
QDF	quadratic driving force
$SAPO_4$	silicoaluminophosphate
SSE	sum of squared errors
TCD	temperature conductivity detector

TVSA	temperature/vacuum swing adsorption
VATS	virtual adsorption test suite
VPSA	combined pressure and vacuum swing adsorption
VSA	vacuum swing adsorption

NOMENCLATURE

a	saturation capacity in Toth equation, mol kg^{-1} kPa^{-1}
a_0	Toth equation parameter, mol kg^{-1} kPa^{-1}
a_f	superficial free flow area, m^2
a_s	pellet external surface area per unit volume, m^{-1}
a_w	column cross-sectional area, m^2
b	equilibrium constant in Toth equation, kPa^{-1}
b_0	Toth equation parameter, kPa^{-1}
c	concentration, mol m^{-3}; also parameter in Toth equation, K
c_0	inlet concentration, mol m^{-3}
c_{pf}	gas heat capacity, J kg^{-1} K^{-1}
c_{ps}	sorbent heat capacity, J kg^{-1} K^{-1}
c_{pw}	column wall heat capacity, J kg^{-1} K^{-1}
D	fluid-phase diffusion coefficient, m^2 s^{-1}
D_L	axial dispersion coefficient, m^2 s^{-1}
d	fitting parameter
E	Toth equation parameter, K^{-1}
e	second fitting parameter
h_i	column wall to gas heat transfer coefficient, W m^{-2} K^{-1}
h_o	column wall to ambient heat transfer coefficient, W m^{-2} K^{-1}
h_s	sorbent to gas heat transfer coefficient, W m^{-2} K^{-1}
k_e	quiescent bed gas conductivity, W m^{-1} K^{-1}

k_{eff}	effective axial thermal conductivity, W m^{-1} K^{-1}
k_f	gas conduction, W m^{-1} K^{-1}
k_n	mass transfer coefficient, s^{-1}
k_s	sorbent conduction, W m^{-1} K^{-1}
k_w	column wall conduction, W m^{-1} K^{-1}
L	bed height, m
N	number of individual readings of X_i
Nu	Nusselt number
n	sorbent loading, mol kg^{-1}
P_1	low pressure setting at calibration gas bottle regulator
P_2	high pressure setting at calibration gas bottle regulator
Pe	particle Peclet number
Pr	Prandtl number
p	partial pressure in Toth equation, kPa
p_i	column inner perimeter, m^2
p_i^s	sorbate saturated pressure
p_o	column outer perimeter, m^2
$\overline{q}$	average adsorbed concentration, mol m^{-3}
q^*	equilibrium adsorption concentration, mol m^{-3}
q_0^*	equilibrium sorbate concentration
q_m	monolayer capacity
R	ideal gas constant
R^2	coefficient of determination

Re	Reynolds number
R_p	pellet radius, m
r	distance between sorbate and sorbent materials
Sc	Schmidt number
Sh	Sherwood number
S_x	standard deviation
t	time, s; also heterogeneity parameter in Toth equation
t_0	Toth equation parameter
T	temperature, K
T_0	inlet temperature, K
T_a	ambient temperature, K
T_f	gas temperature, K
T_s	sorbent temperature, K
T_w	column wall temperature, K
X_i	mole fraction of *i* in the absorbed phase at equilibrium
x	axial coordinate, m
Y_i	mole fraction of *i* in the gas phase at equilibrium
ε	void fraction
θ	threshold parameter
λ	isosteric heat of adsorption, kJ mol^{-1}
μ	gas viscosity, micro-poise
υ_i	interstitial velocity, m s^{-1}
ρ_f	gas density, kg m^{-3}

ρ_s	sorbent density, kg m^{-3}
ρ_w	column wall density, kg m^{-3}
σ	characteristic of molecule

CHAPTER I

1. INTRODUCTION

Fixed adsorbent beds are used for gaseous separations across a wide range of applications in the chemical processing industry, for thermochemical energy storage, and for atmospheric control in habitable volumes. These are generally multiple bed cyclic processes such as pressure swing adsorption (PSA) or temperature swing adsorption (TSA). To reduce the high cost and lengthy schedule associated with hardware testing, design of the process cycles typically utilizes computer simulations. However, direct simulation of the highly random sorbent particle packing and small-scale features of the flow between particles in a fixed-bed are impractical from a computer execution time standpoint. This is compounded by the complexity of sorbate transport into composite structures such as clay-bound zeolite particles, which have four distinct mass transfer modes with disparate length scales ranging from Angstroms to microns. Another difficulty in a full simulation is the simultaneous solution of the coupled thermal, mass, and momentum balance partial differential equations, which are of mixed type. Since the design and optimization of a cyclic process often requires simulating many repeated cycles in order to attain a cyclic steady state response, the simulation execution times must be short.

Due to these constraints, simplifying assumptions must be applied to the mathematical models and simulations used for the design and optimization of fixed-bed gas separation processes. Based on a literature survey conducted on fixed-bed gas adsorption models (discussed in detail in the next chapter), all models actually used for process design were 1-D with respect to the gaseous mass and thermal balance equations. Although the 2-D models reviewed (Pentchev and Seikova, 2002; Mohamadinejad et al., 2003; Mette et al., 2014) provided important information about the actual 2-D flow in small columns, none were actually applied in a process design. The majority of the 1-D models reviewed are based on the 1-D axially dispersed plug flow model, which includes a dispersion term that arises from Fickian diffusion in the derivation. However, in practice, the axial dispersion term lumps not only molecular diffusion but also the effects of turbulence, flow splitting, and rejoining around particles, Taylor dispersion, channeling, and wall effects. A few of the models reviewed omit the dispersion term and thus are true plug flow models.

The other simplification commonly employed is the lumping of the four mass transfer mechanisms from the free stream to the interior of the zeolite crystal into one or two linear expressions. Most of the models studied employed the linear driving force (LDF) equation, which was first proposed by Glueckauf (1955) and lumps all mechanisms into a single constant term. A few of the models employed an adjustment of the LDF mass transfer term based on local concentration, temperature, or adsorbent loading.

The most frequently used approach to obtain the lumped LDF mass transfer coefficient is empirical via experimental data from breakthrough experiments. To conserve costs and materials, the breakthrough experiments are generally performed on small diameter columns. These coefficients are then typically used for simulation-based design of separation processes such as PSA, TSA, etc. The axial dispersion term is generally estimated, however the range of values resulting from the commonly applied correlations is quite wide.

This work shows that the common approach briefly described above is fundamentally flawed for a specific sorbate/sorbent system (H_2O/5A) and by extension of other systems with a high distribution factor (which indicates the steepness or curvature of the equilibrium adsorption isotherm). Even using widely accepted correlations for the axial dispersion coefficient, this approach results in a non-physical simulation behavior, i.e., sharpening of the concentration front just prior to breakthrough. This non-physical behavior is only evident upon inspection of the internal concentration history, yet most articles on simulations of breakthrough curves in the literature omit an inspection of the internal concentration history. Therefore, many simulations in the literature have likely overlooked these non-physical simulation circumstances.

It is further shown in this work that increases in the axial dispersion coefficient outside the bounds of accepted correlations to capture dispersive breakthrough curves (which occur due to channeling in small diameter packed beds typically used in breakthrough experiments) results in a more severe non-physical trend or the complete departure of the simulated internal concentration profile from the expected constant pattern behavior (CPB).

In this work, a new metric for breakthrough sharpening is developed based on the maximum slope ratio. Through extensive parametric studies, a threshold equation and threshold parameter was found for four sorbate/sorbent systems. This threshold equation maps the onset of unacceptable breakthrough sharpening as a function of two parameters in the axially dispersed plug flow model, i.e., the axial dispersion coefficient (D_L) and the LDF mass transfer coefficient. Once determined, the threshold equation may be used to govern the magnitude of D_L and LDF for a specific sorbate/sorbent system in order to avoid a non-physical simulation.

This work also determines a relationship between the distribution factor for each of the sorbate/sorbent pairs studied and the threshold parameter. This relationship may then be used to determine a priori the limiting magnitudes of D_L and LDF for any sorbate/sorbent system given the distribution factor for that system.

In addition to showing the limitations of a common approach for finding mass transfer coefficients in the axially dispersed plug flow model, an alternate, three-step deterministic approach is discussed and demonstrated where:

(1) A method is described where, through the use of empirical correlations for heat transfer coefficients and a simple thermal characterization test, the sole remaining undetermined heat transfer coefficient in the energy equations may be found independently via an empirical determination.

(2) A method is described and demonstrated where experimental data that include centerline concentration measurements is used for independent empirical determination of the LDF coefficient. This method isolates the physics to that appropriate for the LDF coefficient (the mass transfer occurring between the free stream and the sorbed state) and is not confounded by channeling effects generally observed in small diameter packed beds.

(3) Finally, the three-step deterministic method is completed by the empirical determination of D_L from concentration data taken far downstream. Magnitude of the D_L term is limited via use of the relationships described above to avoid a non-physical simulation.

Chapter 2 discusses the wide scale of gas adsorption applications, describes the principle adsorbents used in those applications, and the principle mechanisms important in modeling gas adsorption processes. Also discussed in Chapter 2 are the emerging gas adsorption applications, to which this work is particularly beneficial. Finally, Chapter 2 provides a review of recent literature on adsorption modeling and simulation, and establishes the applicability of this work to current state-of-the-art simulations in this field.

Chapter 3 describes the experimental apparatus and procedures used to develop the experimental results simulated in this work. Also, Chapter 3 provides an analysis of the experimental uncertainty for these experimental results.

Chapter 4 establishes a method for determining mass transfer coefficients and addresses the limitations of breakthrough curve analysis in fixed-bed adsorption. After discussing the relevant issues in detail, the mathematical model used in this work is developed and the simulation code validated. The experimental and simulation results for two adsorbent/sorbate systems, carbon dioxide (CO_2) and H_2O vapor on zeolite 5A, are used to (1) investigate the uncertainty in simulation results due to uncertainties in correlations used for heat transfer coefficients, (2) illustrate the breakthrough curve analysis methodology, and (3) address the limitations of breakthrough curve analysis in fixed-bed adsorption.

Based on simulation results from four adsorbent/sorbate systems, Chapter 5 develops a limiting relationship between the mass transfer coefficients, LDF and D_L, which maps the onset of non-physical behavior. This relationship is then generalized to apply to other sorbent/sorbate systems based on their distribution factor to determine the limiting values of LDF and D_L in order to prevent non-physical simulation results.

Finally, Chapter 6 summarizes this work, its applicability to chemical, thermal, and aerospace processes, and its importance in enhancing the accuracy of breakthrough curve analysis and thus simulations used to develop and refine a wide range of gas adsorption processes that are in use or being developed for use today.

CHAPTER II

2. ADSORBENTS AND GAS SEPARATION PROCESSES

The subject of this work is to refine methods used to derive mass transfer coefficients for adsorbent processes to increase the accuracy of computer simulations used to design gas separation systems. The importance of this work is illustrated below by discussing gas separation applications in current use in the chemical processing industry (CPI) and in atmospheric control of habitable volumes. The nature of adsorbents and the physics involved with adsorbent processes are discussed. Other gas separation processes under development for future applications such as post-combustion carbon capture will be discussed in Section 2.2. Finally, a review of recent mathematical models and computer simulations in the open literature used for derivation of mass transfer coefficients in gas separation processes is provided in Section 2.4.

2.1 Gas Separations in Current Use

Separation processes are defined as those processes that transform a mixture of substances into two or more product streams of differing compositions (King, 1980). The importance of separation processes may be inferred from their centrality to many key chemical-processing industries. A few examples are petroleum refining, air and water purification, and the food and pharmaceutical industries. The wide range and criticality of these example industries help illustrate the significance of separation processes in society.

The study of separation processes, especially as it pertains to improvements in the efficiency of separations, is of critical importance as separations are the costliest step in many chemical processes (King, 1980). The inherent difficulty in separations stems from the fact that separations reverse the mixing of substances and thus requires a decrease in entropy (Yang, 2003).

Separation processes are generally classified into two types, purification and bulk removal processes. In purification processes the mixture is relatively dilute, and at least one product stream consists of an essentially pure substance. An example of a purification process is the thorough dehumidification of gases for use in highly moisture sensitive instruments, or in a highly moisture sensitive downstream process. In bulk separation processes, there may be one or more valuable products, and these may be only partially purified substances. Examples of bulk separations are the partial scrubbing of CO_2 from a submarine or spacecraft atmosphere and the separation of linear paraffins from branched and cyclic isomers (Ruthven, 1984).

2.1.1 Criteria for Commercial Use of Gas Adsorption

This work addresses gas separations via adsorption. However, due to its simplicity, scalability, and ability to produce high purity products, cryogenic separation (e.g., liquefaction followed by distillation) is more common for commercial gas separations. Yet due to the high energy cost of cryogenic separations, there are certain conditions where gas adsorption is advantageous over the cryogenic process. The principle criteria used to determine which process is preferred are the relative volatility in the distillation step and the separation factor in gas adsorption. For an ideal binary mixture, the relative volatility is defined as the ratio of the volatility of the component with greater volatility to the volatility of the second component. Equation (2.1) defines the separation factor in gas adsorption;

$$\alpha_{ij} = \frac{X_i / Y_i}{X_j / Y_j} \tag{2.1}$$

where X_i represents the mole fraction of i in the adsorbed phase at equilibrium, and Y_i represents the mole fraction of i in the gas phase at equilibrium. For a separation process where the relative volatility is low and the separation factor is high, adsorption is likely to be favored, although additional factors must also be taken into account. Other criteria favoring adsorption, as discussed in more detail by Keller, 1983, are the feed stream having pure substances with overlapping boiling points and the feed gas already at a state of elevated pressure facilitating pressure swing adsorption. The CPI gas separation applications for which adsorption is favored are shown in Table 2.1.

Table 2.1 Common commercial and industrial uses for sorbents (Keller, 1983; Yang, 2003).

Component to be adsorbed	Other components	Adsorbent(s)
Gas purification		
H_2O	Olefin-containing cracked gas, natural gas, air, synthesis gas, etc.	Silica, alumina, zeolite (3A)
CO_2	C_2H_4, natural gas, etc.	Zeolite, carbon molecular sieve
Hydrocarbons, halogenated organics, solvents	Vent streams	Activated carbon, silicalite, others
Sulfur compounds	natural gas, hydrogen, liquefied petroleum gas (LPG)	Zeolite, activated alumina
SO_2	Vent streams	Zeolite, activated carbon
Gas bulk separations		
Normal paraffins	iso-paraffins, aromatics	Zeolite
N_2	O_2	Zeolite
O_2	N_2	Carbon molecular sieve
CO	CH_4, CO_2, N_2, Ar, NH_3/H_2	Zeolite, activated carbon
Acetone	Vent streams	Activated carbon
C_2H_4	Vent streams	Activated carbon

2.1.2 Applications of Gas Adsorption for Atmospheric Control in Habitable Volumes

Humans exhale CO_2, a product of normal metabolic oxidation, at a rate of approximately 1 kg per day. Plants consume CO_2 and maintain the Earth's atmosphere at CO_2 concentrations near 300 ppm (Parker, et al., 1973). However, physiochemical processes are required in habitable volumes separated from the Earth's atmosphere, such as in submarines, on spacecraft, in mine safety shelters, and in military shelters in development to protect warfighters from chemical warfare agents (James and Macatangay, 2009; Raatschen et al., 2009). Currently, physiologically safe concentrations of CO_2 in spacecraft are estimated at 7000 ppm for mission durations up to 180 days and 5000 ppm for longer duration missions (James, 2008). Various physiological disorders result from higher CO_2 levels, including headaches, increased heart rate and respiration rate, and at levels above 6% CO_2 concentration, dizziness, stupor, unconsciousness, and even death can result (Waligora et al., 1991). The artificial means used in spacecraft to replace the function of plants on Earth and maintain CO_2 levels at safe concentrations are described below.

Prior to Skylab, and on short-duration Space Shuttle flights, lithium hydroxide (LiOH) canisters scrubbed the air of CO_2. However, the LiOH scrubbing process is a non-regenerable chemical reaction, requiring approximately 1.18 kg of LiOH per crew-day. This resupply penalty is prohibitive on longer flights. The National Aeronautics and Space Administration (NASA) and space administrations in other countries exploit the reversible adsorption of CO_2 onto zeolite sorbents for longer duration missions. On Skylab, a two-bed, zeolite-based system was used (Hopson et al., 1971); on the International Space Station (ISS), a four-bed system with CO_2 and water-save capability is in operation, as shown in Figure 2.1 (AiResearch Los Angeles Division, 1992). Two beds operate in the adsorption mode (a desiccant and CO_2 sorbent bed), while the other set of identical beds are being desorbed. The desiccant beds are desorbed via heated gas stripping, while the sorbent beds are heated and subjected to a vacuum thus undergoing the vacuum and thermal swing process.

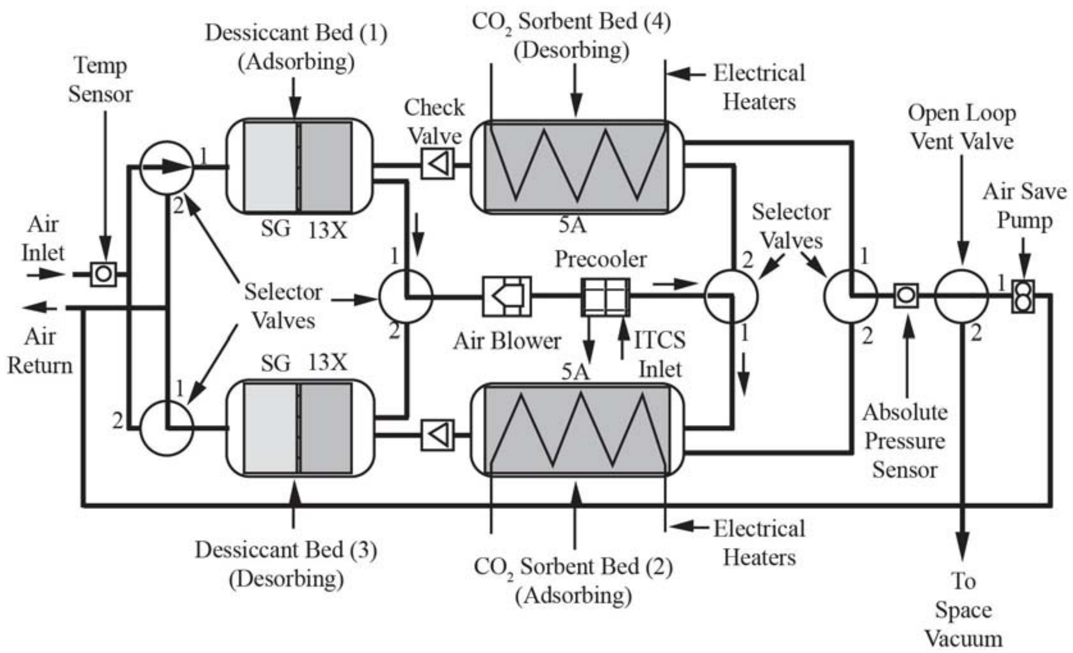


Figure 2.1 Four-bed molecular sieve CO_2 removal system schematic (Knox, 2000).

2.1.3 Adsorbents in Current Use

Although many new sorbent types are being developed, particularly for future application in the removal of carbon dioxide from coal-fired energy plants, there are surprising few types in current commercial use, or even commercially available today. Three types in highest common usage, in order of annual sales, are activated carbons, zeolites, and silica gels. Selection of the appropriate sorbent requires matching sorbent characteristics to the specific gas separation process. The characteristics and applications of these commercial sorbents are reviewed below.

2.1.3.1 Activated Carbon History

Activated carbon has a long history, having been in continual use since the 1800's. It was preceded by charcoal, which was in use as early as 1794 to decolorize sugar syrup. Development of modern processes were initiated during World War I, where activated carbon produced from coconut shells was used in gas masks to filter out chemical warfare agents. The manufacturing processes were matured in the 1930's and remain largely unchanged today. Among sorbents, activated carbon has the highest production rate; in 1977, it had yearly sales of about $1 billion (Yang, 2003).

2.1.3.2 Activated Carbon Synthesis

Organic materials with high carbon content are used as precursors in the manufacture of activated carbons. These include coal, lignite, wood, nut shells, peat, pitches, and cokes. Two manufacturing processes are used in activated carbon production, thermal activation and chemical activation. In the former process, high temperatures (>1000°C) are required to produce a carbon skeleton; then the pore volume and surface area is increased via oxidation. For chemical activation, phosphoric acid is typically used at more moderate temperatures (450-700°C), followed by rinsing and drying (Baker et al., 1997).

2.1.3.3 Activated Carbon Surface Chemistry

As a result of the surface oxidation and open structure, activated carbons have unique advantages for certain sorbates. Surface area is as high as 2,500 m^2/g, the highest for any sorbent (Yang, 1997). The oxidized surface is nonpolar or only slightly polar, and adsorbs organic compounds more strongly than water vapor. Dispersion-repulsion, or van der Waals forces, are dominant, which are weak in comparison with the electrostatic forces present for sorbents with an ionic structure such as zeolites. The van der Waals forces are due to instantaneous induced dipole and quadrupole interactions, which may be described by the Lennard-Jones potential function,

$$\phi = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right], \tag{2.2}$$

where r is the distance between the sorbate and sorbent molecules, and ϵ and σ are characteristics of the molecules. As a result of the weak bond for water vapor, activated carbon is the only commercial sorbent that does not require nearly complete desiccation of the upstream gas. The open structure, with high surface area and pore volume, enables the adsorption of greater quantities of more nonpolar and weakly polar gases than other sorbents. Finally, due to the lack of strong polar bonds, the bond strength is lower than for other sorbents and thus desorption may be accomplished easier (Yang, 1997). However, the disadvantage of activated carbon is also due to the nonpolar or weak polar surface, which results in lower capacities for polar molecules than other sorbents, particularly at low partial pressures.

2.1.3.4 Applications of Activated Carbon

Activated carbons are used for atmospheric trace contaminant control and water purification in spacecraft life support applications. The most common commercial and industrial uses for activated carbon were shown in Table 2.1.

2.1.3.5 Silica Gel History

Silica gel was originally developed as an alternative sorbent to activated carbon for use in gas masks during WWI. Although in practice it was unable to compete with activated carbon for warfare chemical agent adsorption, silica gel has since become the most widely used desiccant for commercial separation processes. Sales in 1997 were estimated at $27 million (Yang, 2003).

2.1.3.6 Silica Gel Synthesis

Commercial silica gel is synthesized via the polymerization of silicic acid. This acid is prepared by mixing a solution of sodium silicate with (typically) sulfuric or hydrochloric acid. The silicic acid is liberated as fine particles, which then precipitates upon standing into primary particles that are linked silicate tetrahedral chains. Silica gel derives its name from this white jelly-like precipitate. After washing, drying of the precipitate results in bond formation between adjacent primary particles. The final structure was presented by Radenburg, 2013 as shown in Figure 2.2. The micropore size in the final product is a consequence of the primary particle size of the liberated silicic acid, which can be controlled by varying the solution pH, temperature, and silica concentration (Yang, 2003).

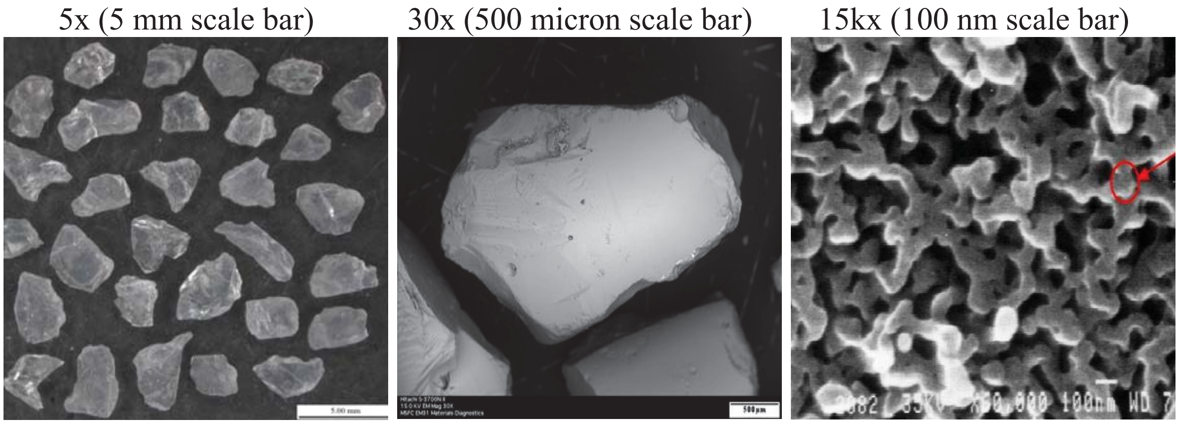

5x (5 mm scale bar)
30x (500 micron scale bar)
15kx (100 nm scale bar)

Figure 2.2 SEM images of silica gel used in the ISS CDRA. Note the circle identifying one of the primary particles in the 15kx view (15kx image from http://www.grace.com/EngineeringMaterialScience/SilicaGel/SilicaGelStructure.aspx), other images from Radenburg, 2013).

2.1.3.7 Silica Gel Surface Chemistry

Hydroxyl (or silanol) groups on the silica gel surface account for an intermediate degree of surface polarity. The surface polarity of silica gel is greater than for activated carbon, but not as high as for zeolites. As a result, polar molecules are adsorbed preferentially over nonpolar molecules, yet the bond strength (and thus energy required for desorption) is not as great as for zeolites. Aside from water, silica gels also have good selectivity for alcohols, phenols, and amines.

2.1.3.8 Applications of Silica Gel

Two variants of silica gel types are commercially available. The narrow pore (also called high area as well as regular density) silica gel has a greater surface area, 800 square meters per gram vs. 400 square meters per gram for the wide pore type (Grace Davison, 2010). As may be expected from its larger surface area, water capacity is much greater for the narrow pore type. However, as the wide pore type can tolerate liquid water without fracturing, unlike the narrow pore type, and is therefore used as a guard bed in dryers.

Silica gel is well suited to molecules that have a high dipole moment and high polarizability, such as water vapor. Silica gel is used to desiccate the air stream prior to CO_2 removal in spacecraft life support applications as described later in more detail. The most common commercial uses are shown in Table 1.1.

2.1.3.9 Zeolite History

The naturally occurring form of zeolite takes its name from the Greek words meaning “to boil” and “a stone,” based on the observation that steam is produced upon rapid heating of the mineral. It was so named and classified as a new form of mineral in

1956 by Axel Fredrik Cronstedt, a Swedish mineralogist (Kuhl and Kresge, 1997). As early as 1840, it was recognized that mineral zeolites could reversibly adsorb water with no change in morphology. F. Grandjean observed gas adsorption in 1909 on chabazite, a natural zeolite. Later observations by McBain on the selectivity of chabazite for gas molecules with sizes below 5 angstroms lead him to call these minerals molecular sieves. However, due to impurity of the mineral zeolites they are less well suited for industrial applications. For example, iron (a common impurity) can strongly affect the catalysis processes (Breck, 1974). A purer form of zeolites became available upon the invention of the synthesis process of zeolites A and X by Milton (1959).

2.1.3.10 Zeolite Synthesis

The most common process for zeolite manufacture is the hydrogel process. The primary steps in the hydrogel zeolite synthesis are crystallization, ion exchange, and pelletization.

Synthesis of the zeolite crystals starts with a solution of sodium silicate and sodium aluminate in sodium hydroxide. The solution is held in a hydrothermal synthesis autoclave at conditions of pressure, pH, concentration, and temperature specific to the desired zeolite. Over a period of time, varying from a few hours to a few days, aluminosilicate gels crystallize out of the solution to form a gel. The zeolite crystals are filtered out of the synthesis liquor prior to undergoing ion exchange in an aqueous solution.

Zeolite crystals consist of SiO_4 and AlO_4 tetrahedrals sharing an oxygen atom to form polyhedral building blocks. These in turn form framework structures such as shown in Figure 2.3a, which repeat to form a crystal lattice resulting in cubic geometries. The primary geometry types are A, X, and Y. Exchanging the cation may be used to change the adsorptive properties of zeolites. For example, the A type with calcium cation, or CaA, adsorbs water and carbon dioxide, while the potassium form, or KA, adsorbs water but not carbon dioxide due to a smaller pore size. Thus the ion exchange step is used to tailor the zeolite crystal for a specific application. The crystals are dried at 150°C prior to the pelletization step.

Pelletization of the approximately 2-μm crystals with clay binder is required to reduce flow resistance through a fixed-bed. The most common processes are extrusion to form cylindrical pellets, extrusion followed by rolling to form spherical pellets, and granulation (also to form spherical pellets). The composite nature of the pellets is shown in Figure 2.3b and in the 1kx view in Figure 2.4. However, the open composite structure required for easy gas penetration tends to have a low resistance to attrition and crushing, and weakens due to humidity and/or large temperature excursions (Watson et al., 2015, Knox et al., 2015b). Scanning electron microscope (SEM) images of sorbent pellets used in the ISS CDRA are shown in Figure 2.4. The final steps in the synthesis product are drying at 200°C and calcination at 650°C (Ruthven, 1984).

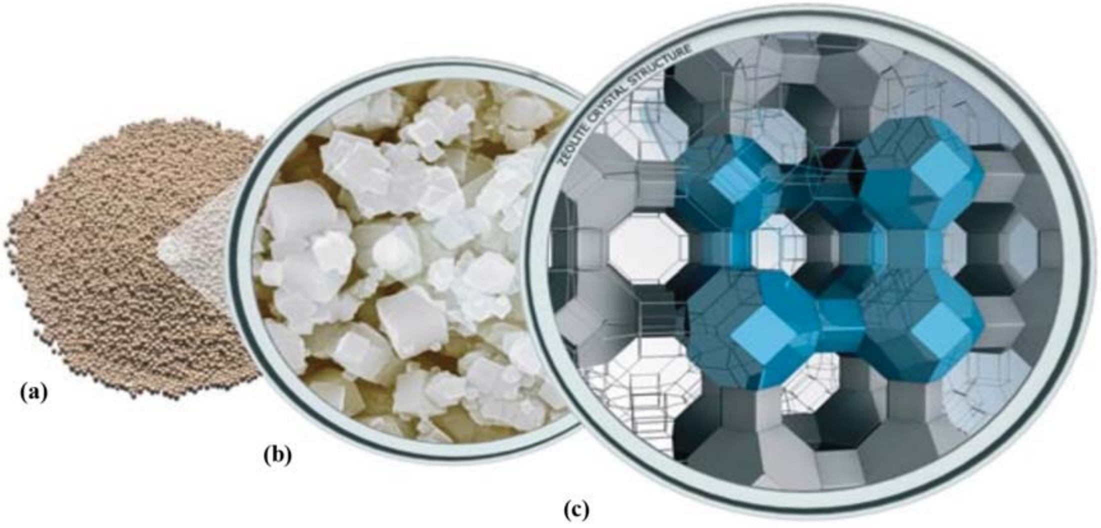


Figure 2.3 (a) Pelletized zeolite pellets, (b) crystals, and (c) framework structure http://www.grace.com/engineeredmaterials/productsandapplications/InsulatingGlass/SieveBeads/Grades.aspx).

2.1.3.11 Zeolite Surface Chemistry

Adsorption in zeolites for non-polar molecules is due in part to van der Waals forces as with silica gels and carbons. However, the forces between zeolites and polar gases (particularly H_2O and CO_2) are much higher, resulting in superior separation and capacity for polar molecules compared to activated carbon or silica gel. Conversely, high temperatures are also required for desorption. Also, since adsorbed H_2O will exclude CO_2 adsorption due to the higher polarity of the water molecule, the stream must be desiccated prior to the CO_2 adsorption step.

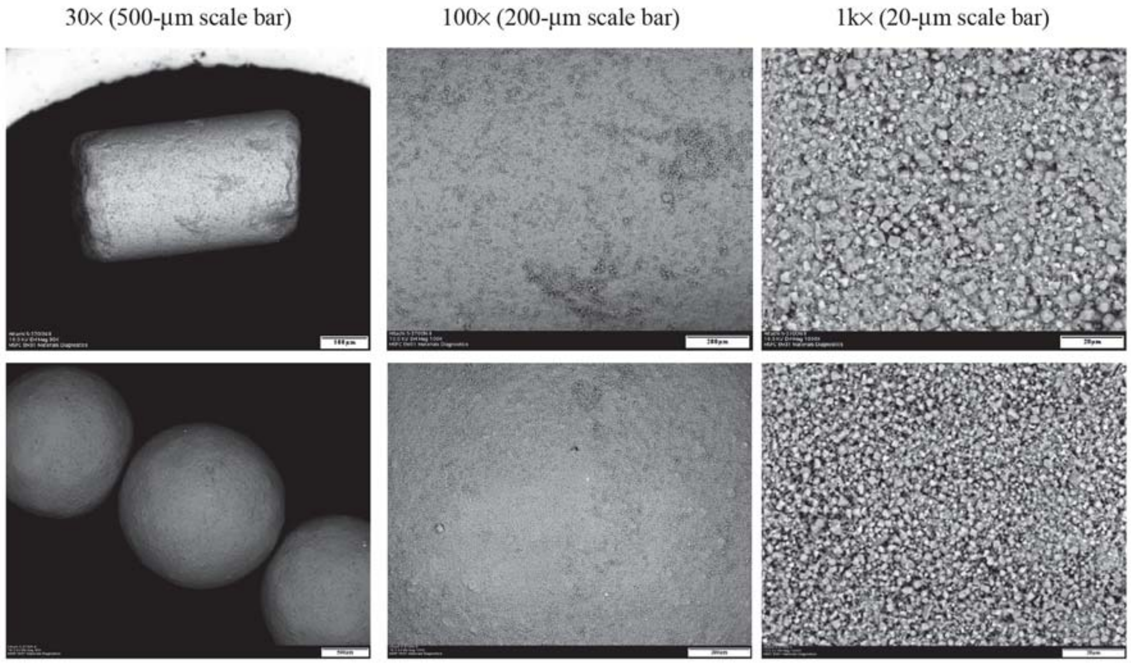


Figure 2.4 SEM images of pelletized zeolite 5A used in the ISS CDRA. Individual zeolite crystals are evident in the 1kx views (Radenburg, 2013).

2.1.3.12 Applications of Zeolites

Zeolite crystals are unique among sorbents due to their constant pore size. This property allows the zeolite to act as a molecular sieve. The other unique feature of zeolites is their high electric field gradient due to the cations being situated above the negatively charged surface oxides. This favors a molecule with a high quadrupole moment, such as water vapor, over carbon dioxide (Yang, 1987).

As shown in Table 2.1, there are many commercial separations made possible by both the sieving and polar properties of zeolites. Zeolites are also used in spacecraft life support systems for both the desiccation of air and the subsequent removal of CO_2.

2.1.4 Equilibrium Capacity Isotherms

The most important metric used in evaluation of a sorbent for a gas separation process is sorbent capacity. To quantify the capacity of a sorbent for a sorbate, experimental capacity data are collected on a small amount of sorbent in a closed chamber with sorbate partial pressure and temperature held at constant conditions until equilibrium capacity is attained. In general, a series of these experiments are conducted by varying the sorbate pressure over the desired range while holding temperature constant, resulting in a single equilibrium capacity isotherm. The experiment is then repeated for each temperature of interest. Figure 2.5 shows an example set of isotherms for CO_2 on zeolite 5A (Wang and LeVan, 2009).

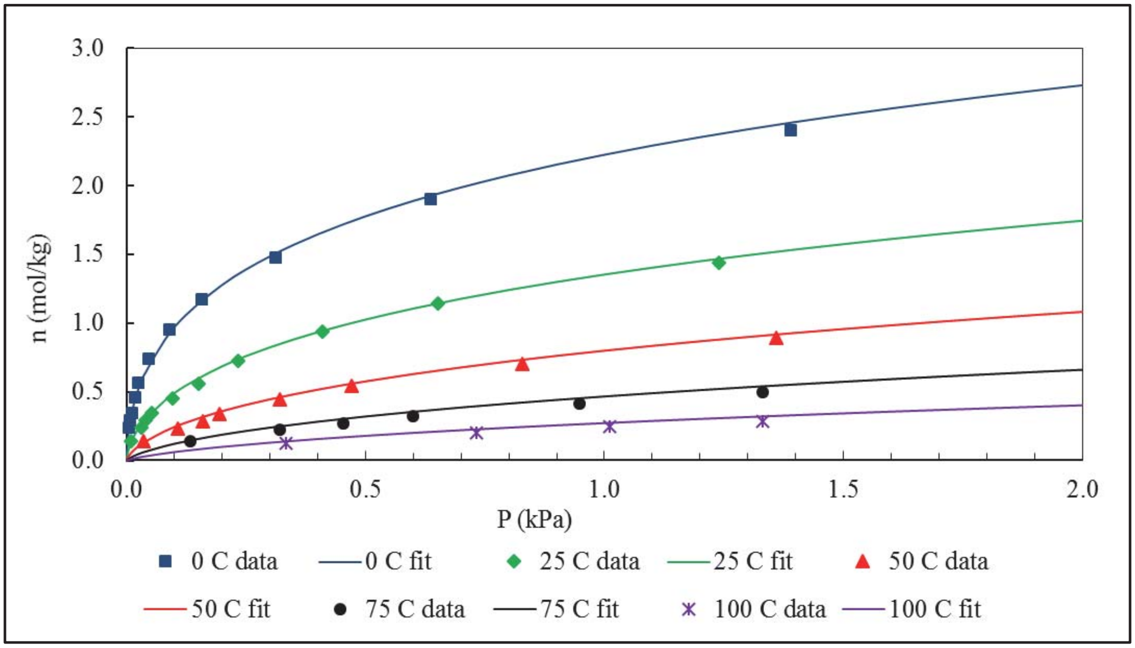


Figure 2.5 Equilibrium capacity isotherms for CO_2 on zeolite 5A, where data points represent test data and the lines represent the Toth fit to the data (Wang and LeVan, 2009).

The shapes of isotherms are known to have a dramatic effect on the breakthrough curve shape (Park and Knaebel, 1992). As will be shown in this work, the steepness of an isotherm is a contributing factor in the onset of non-physical behavior in the simulation of breakthrough curves. Isotherms are categorized by their shape into types I through V as shown in Figure 2.6 (Brunauer et al., 1940). The sorbents used in this study concave downward, or type I. This shape is considered favorable, if it leads to a compact wave shape and constant pattern behavior in a breakthrough experiment, whereas the concave downward or type III shape will have a spreading pattern that is undesirable for effective fixed-bed utilization (LeVan and Carta, 2008).

Equilibrium capacity isotherms also provide critical input data for a computer simulation. Mathematically, the driving force for adsorption is the difference between the loading of a sorbent particle and the loading that the particle would have if it were in equilibrium with the gas stream. Equilibrium equations are used to fit the equilibrium capacity isotherm data; for example, the Toth equation is shown in Equation (2.3);

$$n = \frac{ap}{\left[1+(bp)^t\right]^{1/t}}; \qquad b = b_0 \exp(E/T); \qquad a = a_0 \exp(E/T); \qquad t = t_0 + c/T\,, \quad (2.3)$$

where n is the sorbent loading, a is the saturation capacity, p is the partial pressure, b is an equilibrium constant, and t is the heterogeneity parameter. Parameters a, b, and t are temperature dependent as shown, whereas a_0, b_0, and t_0 are system dependent adsorption isotherm parameters. A comparison of the Toth equation and the experimental data are shown in Figure 2.5. The adsorption isotherm parameters are given in Wang and LeVan (2009).

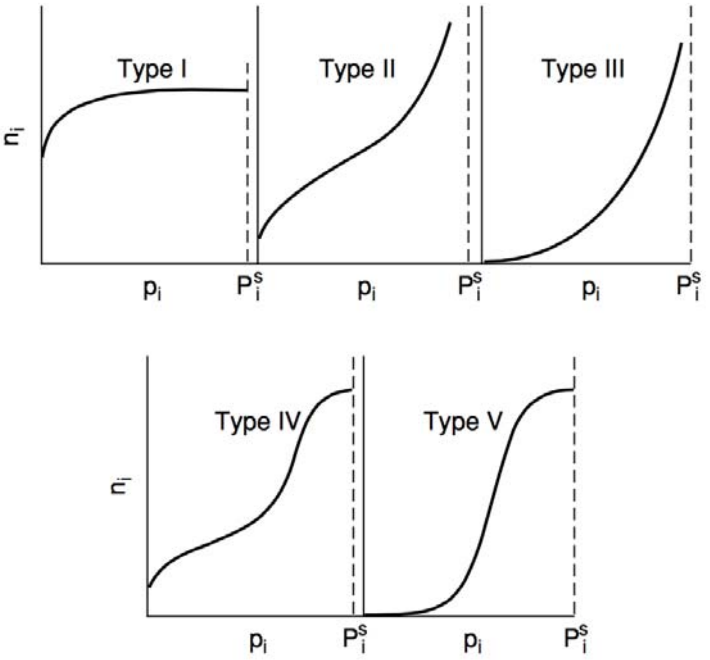


Figure 2.6 Isotherm types I through V, where p_i is sorbate partial pressure, n_i is sorbent loading, and P_i^s is the sorbate saturated pressure (Brunauer et al., 1940).

2.1.5 Mass Transfer Mechanisms in Fixed-Beds

Pelletized sorbents may be used in fixed or fluidized beds. In fluidized beds, the pellets are transported from an adsorption zone after adsorption is complete to a desorption zone with conditions suitable for removal of the sorbate. In a fixed-bed, as illustrated in Figure 2.7, pellets are retained in a single bed; after adsorption is complete, conditions are altered to encourage desorption to occur. The change in conditions may be a swing in pressure or temperature (termed pressure swing or temperature swing adsorption, respectively), purging in the same direction as adsorption with inert gas (co-current inert purge stripping), purging with a reversal of flow direction (counter-current inert purge stripping), or a combination of these such as temperature/vacuum swing adsorption (TVSA) as used for the carbon dioxide sorbent bed in the ISS CDRA. Another method used for example, in the separation of paraffins from gasoline, is displacement purge; in this case, a species that is preferentially adsorbed is used to replace and purge the desired species out of the bed. Distillation is then required to separate the paraffin from the regeneration effluent. This was one of the earliest commercial uses of zeolites (Kuhl and Kresge, 1997).

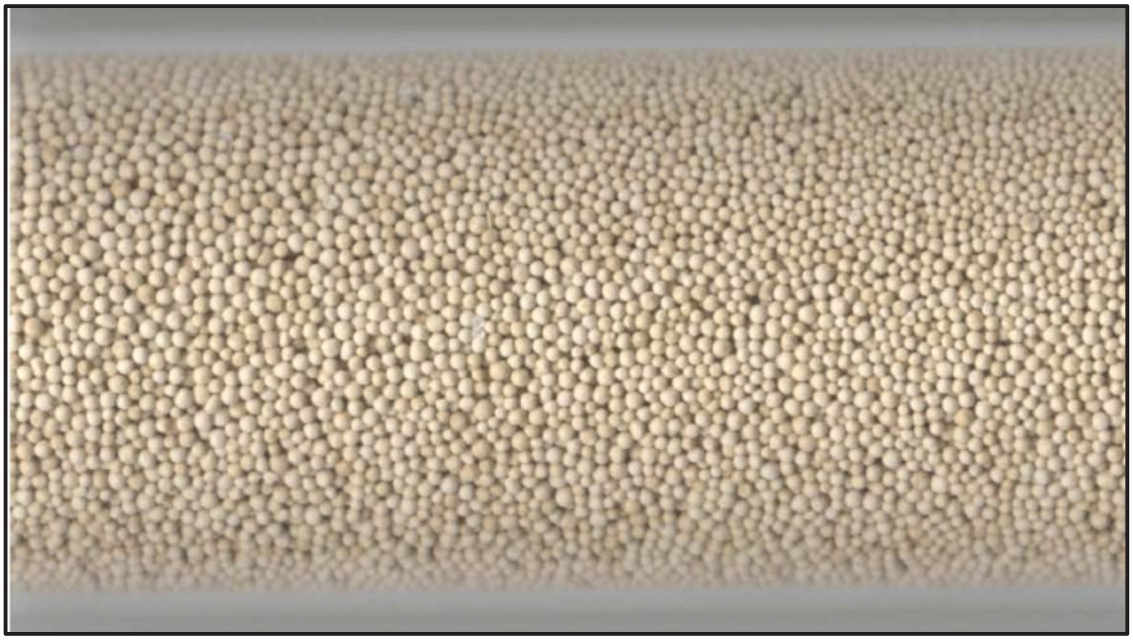


Figure 2.7 Packed (or fixed) bed of zeolite 13X beads. Photo taken by author.

The physics of the adsorption process in a fixed-bed can be broken down into five general mass transfer modes, described in detail below and illustrated in Figure 2.8. The mass transfer modes are as follows: (1) Convection and dispersion of the sorbate via the carrier gas longitudinally down the packed bed, (2) mass transfer of the sorbate from the free stream through the external fluid film around the zeolite pellet, (3) mass transfer through the pellet macropores, (4) mass transfer into the zeolite micropores through the pore mouth, or barrier resistance, (5) surface migration of molecules from external adsorption sites to sites through the micropores further in the interior of the crystal. Each of the transfer mechanisms is described in greater detail below.

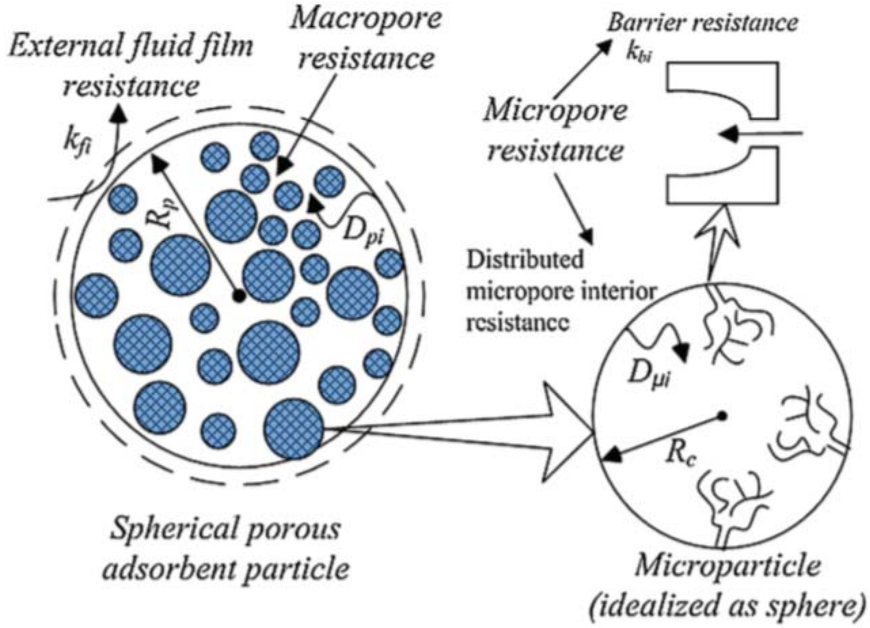

External fluid film resistance
k_{fi}
Macropore resistance
R_p
D_{pi}
Spherical porous adsorbent particle
Barrier resistance k_{bi}
Micropore resistance
Distributed micropore interior resistance
$D_{\mu i}$
R_c
Microparticle (idealized as sphere)

Figure 2.8 Depiction of fixed-bed and zeolite mass transfer mechanisms (Shareeyan et al., 2014).

2.1.5.1 Convection and Dispersion

Fixed-beds consist of a non-ordered packing of randomly sized sorbent particles as shown in Figure 2.7. It follows that the resulting flow field is also random with dispersive attributes. Dispersion can include turbulence, flow splitting and rejoining around particles, Taylor dispersion, channeling, and wall effects. For small channels, dispersion can greatly reduce efficiency by broadening the mass transfer zone, yet as discussed later in detail, care must be taken to avoid the non-physical simulation phenomenon resulting from the interaction of a high axial dispersion coefficient and the Danckwerts boundary condition.

2.1.5.2 External Pellet Film Diffusion

The diffusion of gas from a free stream into a stagnant film layer, and across a stagnant film layer to the zeolite surface, is based on Fick's law (see, for example, Bird, Stewart, and Lightfoot, 1960). Although the film resistance is significant for heat transfer from the gas to the pellet, film resistance is insignificant for mass transfer from the gas to the pellet for zeolites due to the dominant resistance in the macropores and micropores (Ruthven, 1984). The film resistance term and those resulting from following mass transfer modes will be lumped together as a single term. Heat diffusion in the film, however, will be determined via correlation and used in the energy balance equations.

2.1.5.3 Mass Transfer Through the Zeolite Macropores

The zeolite pellets are composed of the zeolite crystals (the cubic structures in Figure 2.3b and clay binder (the randomly shaped material around the cubes). The spaces between the cubic crystals are the macropores. The macropores are visible as the spaces between crystals in the 1kx images in Figure 2.4.

2.1.5.4 Mass Transfer into the Zeolite Crystal through the Pore Mouth

A schematic representation of the framework structure of zeolite A is shown in Figure 2.3c. Sorbate molecules enter into the crystal via the pore mouth, or outermost crystal framework. The effective opening size for zeolite A ranges from approximately 3 to 4.5 Angstroms depending on which molecule is acting as the cation in the framework structure. Examples are: potassium for 3A, sodium for 4A, and calcium for 5A (Ruthven, 1984).

2.1.5.5 Mass Transfer into the Zeolite Crystal Interior through the Micropores

Full utilization of the zeolite crystal requires movement of the sorbate from the external to interior crystal lattices. However, since the molecules are initially captured by the external adsorption sites, the transfer occurs by a “hopping” or surface diffusion mechanism from one adsorption site to another (Yang, 1987; Ruthven, 1984; Karger and Ruthven, 1992).

2.1.6 Linear Driving Force Model

In the LDF model shown in Equation (2.4), the following mass transfer resistances are lumped into a single mass transfer resistance: external pellet film diffusion, mass transfer through the pellet macropores, mass transfer into the zeolite crystal through the pore mouth, and mass transfer into the zeolite crystal interior. If the mass transfer resistance is assumed to be a single mass transfer mechanism that is dominant and constant throughout the adsorption process, then this approach is valid. Also, it has been established that the LDF model incurs little error for most commercial gas phase cycle adsorption processes when the LDF coefficient is empirically derived (Yang, 1997; Sircar, 2000). The most commonly used model for adsorption mass transfer, and the one used in this work, is the LDF model.

$$\frac{\partial \overline{q}}{\partial t} = k_n(q^* - \overline{q}) \tag{2.4}$$

2.2 Emerging Applications of Gas Separations

The subject of this work is to refine the methods used to derive mass transfer coefficients for gas adsorption systems. Although these refinements may be applied to existing gas adsorption systems to increase performance, an even more beneficial application is during the design of new systems, with either newly developed sorbents or where existing sorbents are being newly applied to a separation process. Some emerging applications are described below, both in the commercial and spacecraft life support arenas.

2.2.1 Emerging Applications in the Chemical Processing Industry

The separation of air into high purity nitrogen and oxygen is extremely important in the chemical, processing industry, as these are second and third most produced chemicals respectively. Two innovations have contributed to the significant reduction in cost of oxygen production: (1) the development of the vacuum swing adsorption (VSA) process, and (2) the invention in 1989 of a new zeolite, LiLSX (Yang, 2003; Chao, 1989). The LiLSX zeolite is also under consideration for CO_2 removal on spacecraft life support systems due to its high capacity for CO_2 at low partial pressures (Knox et al., 2016b). A related sorbent developed for the production of oxygen is AgLiLSX, where a small percentage of silver ions (1%-3%) are exchanged with the lithium ions in the LiLSX zeolite, with a resultant increase in the nitrogen capacity (Chiang, 2002). A combined pressure and vacuum swing adsorption (VPSA) process using 40% Ag-exchanged LiLSX was invented by Whitley (2010). The commercial status of this sorbent and process is unknown.

There are a number of other gas separations that may be enabled by emerging sorbents, which will in turn enable future applications, such as purification of hydrogen for fuel cells, and potential enhancements to current applications, such as the use of PSA for recovery of ethylene and propylene from refineries. These are discussed in detail by Yang (2003) and summarized in Table 2.2.

Table 2.2 Gas separation and purification applications enabled by new sorbents (Yang, 2003).

Application	Sorbent and Notes
N_2/CH_4 separation for natural gas upgrading	Clinoptilolite, tinanosilicates by kinetic separation, single-wall carbon nanotubes
CO removal from H2 to < 1 ppm for fuel cell applications	π-complexation sorbents such as CuCl/γ-Al_2O_3, CuY, and, AgY
NO_x removal	Fe-Mn-Ti oxides, Fe-Mn-Zr oxides, Cu-Mn oxides, multi-wall carbon nanotubes
C_3H_6/C_3H_8 (+hydrocarbons) separation	π-complexation sorbents such as CuCl/γ-Al_2O_3, AgNO3/SiO2, AgNO3/clays, aluminophosphate
C_2H_6/C_2H_6 (+hydrocarbons) separation	π-complexation sorbents such as CuCl/γ-Al_2O_3, AgNO3/SiO2, AgNO3/clays

2.2.2 Efforts to Develop Affordable Flue Gas CO_2 Capture Systems

The separation of CO_2 from the flue gas produced by the combustion of coal in power generating plants is an area of gas separations that has seen an extraordinary amount of research activity recently, and which has a close tie to the sorbents and sorbates studied in work. In 2009, coal-based power plants provided 45% of the electrical power in the United States. The use of coal is not expected to decrease significantly in the near future; in 2030, coal-based power plants are predicted to provide 44% of the electrical power in the U.S. However, CO_2 emissions from coal-based power plants in the U.S. produced 1.7 billion metric tons of CO_2 in 2009, constituting 32% of the total U.S. CO_2 emissions resulting from human activities (Vora et al., 2013). To reduce CO_2 emissions, which is considered responsible for the current global warming trend, the U.S. government began funding the development of technologies for the capture of CO_2 from coal-based power plants in 2008. Between 2008 and 2016, the U.S. Congress alone has appropriated more than $7 billion for carbon capture and sequestration (CCS) activities at the Department of Energy (DOE), with CO_2 capture activities accounting for approximately one quarter of that budget. For 2017, the DOE carbon capture budget request is the highest yet at $170 million (Folger, 2016). A large portion of this funding has been directed to the development of technologies that would enable cost-effective CO_2 removal from coal-based power plant smoke stacks, as currently available commercial CO_2 capture technologies are not yet practical for this application. The options under consideration for coal-based power plant smoke stacks (or post-combustion) are shown in Figure 2.9. The current technologies would incur parasitic loads that would reduce power-generating capacity by approximately one-third (Vora et al., 2013). Carbon dioxide capture technologies funded under DOE that could benefit from the methods described in this dissertation are summarized in Table 2.3.

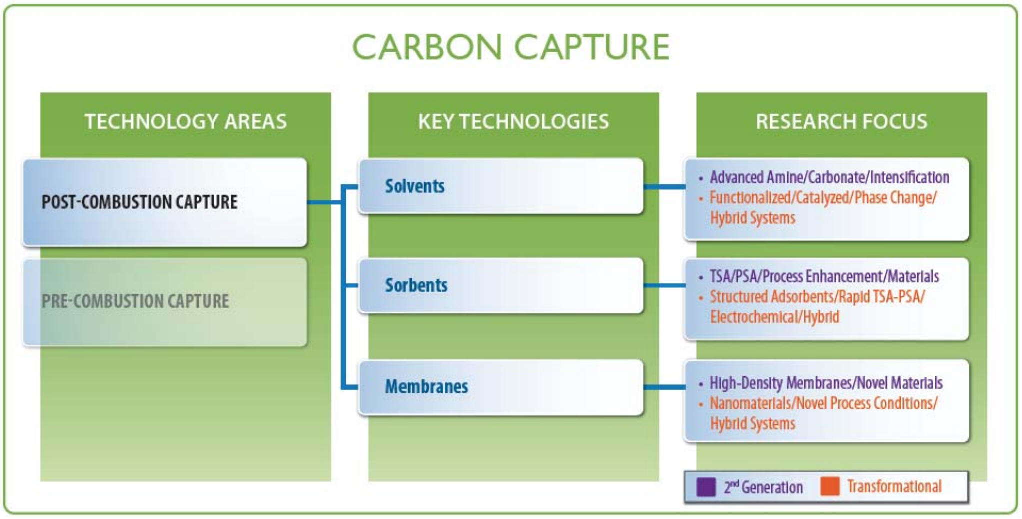

CARBON CAPTURE
TECHNOLOGY AREAS
KEY TECHNOLOGIES
RESEARCH FOCUS
POST-COMBUSTION CAPTURE
PRE-COMBUSTION CAPTURE
Solvents
Sorbents
Membranes
• Advanced Amine/Carbonate/Intensification
• Functionalized/Catalyzed/Phase Change/Hybrid Systems
• TSA/PSA/Process Enhancement/Materials
• Structured Adsorbents/Rapid TSA-PSA/Electrochemical/Hybrid
• High-Density Membranes/Novel Materials
• Nanomaterials/Novel Process Conditions/Hybrid Systems
2nd Generation
Transformational

Figure 2.9 Key technologies and associated research focus for post-combustion capture (National Energy Technology Laboratory, 2013).

Table 2.3 Carbon dioxide capture technologies funded under DOE (Vora et al., 2013).

Project Name	Project Focus	Sorbent Materials	Process Approach	Technology Maturity	Ref.
Bench-Scale Development & Testing of a Novel Adsorption Process for Post-Combustion CO_2 Capture	Novel adsorption process	Micro-porous carbon	TSA, fixed-beds	Bench-Scale, Actual Flue Gas	Jain, 2012
Low-Cost Sorbent for Capturing CO_2 Emission Generated by Existing Coal-Fired Power Plants	Low-cost solid sorbent	Alkalized alumina sorbent	Simulated moving beds	Bench-Scale Using Actual Flue Gas	Elliot, 2012
CO_2 Removal from Flue Gas Using Microporous Metal Organic Frameworks	Micro-porous MOFs	Alumina and Mg/ DOBDC	VPSA	Laboratory-Scale, Simulated Flue Gas	Benin, 2012

Recent reviews of carbon capture progress note that further work on sorbents is required in the areas of capacity, kinetics, selectivity, and operational stability. These reviews also include the more recent approach of removal of CO_2 from the atmosphere despite the difficulty of doing so at greatly reduced concentrations. In this application, the physisorption candidates are activated carbons, zeolites, and MOFs, or the same candidates as for the post-combustion application (Yuan et al., 2015; Miller et al., 2016).

Other recent developments in crystalline molecular sieve development include the silicoaluminophosphates, or $SAPO_4$ sorbents. These molecular sieves have an unusual degree of diversity in the framework structures. $SAPO_4$-34 in particular has shown a high selectivity of for CO_2 over other atmospheric gases (Arevalo-Hilalgo et al., 2010).

2.2.3 Spacecraft Life Support Needs for Reduced Mass/Power/Volume Systems

The long-term goal for NASA is to enable crewed missions to Mars, first to the vicinity of Mars, and then to the Mars surface (NASA Headquarters, 2015). These missions present new challenges for all aspects of spacecraft design in comparison with ISS, as resupply is unavailable in the transit phase, and early return is not possible. Additionally, mass, power, and volume must be minimized for all phases to reduce propulsion needs. Mass reduction is particularly crucial for Mars surface landing and liftoff due to the challenges inherent in these operations for even much smaller payloads.

In manned space vehicles, CO_2 partial pressures have historically been maintained below 7.6 mmHg for short-duration missions (e.g., Apollo or Shuttle flights) and averaging approximately 5 mmHg for long-duration missions (e.g., 30-day Skylab and 180-day ISS missions). However, after 15 years of experience on the ISS, the medical community has gained knowledge on the combined influence of microgravity and CO_2 concentrations that are ten times Earth's normal on human physiology. This greater understanding seems to point towards the need for lower CO_2 concentration requirements for future spacecraft, which will drive the CO_2 removal system to higher flow rates and greater efficiency (James et al., 2011; James 2013a, 2013b).

Recently, it was recognized by the life support community that the current ISS state-of-the-art CO_2 removal technology, the Carbon Dioxide Removal Assembly (CDRA), has reliability and capability gaps that must be solved both for ISS and future Exploration missions (Knox et al., 2015a). Technical interchanges with the adsorption industry and academia were conducted to determine the appropriate development path to achieve the system reliability and capability needed for a crewed mission to Mars. The following goals were established:

1. Select superior desiccants and CO_2 sorbents for ISS CDRA. Criteria for selection include performance, structural stability, and sensitivity to contamination. Only sorbents compatible with the current ISS CDRA hardware are under consideration.
2. Using the ISS CDRA design as a basis, complete design of a next-generation CO_2 removal system with appropriate attributes for a 2-year mission with no resupply. Implement this design in the fabrication of a technology demonstration to fly on ISS by 2019.
3. Down-select between promising alternate technologies after further development as required to bring the candidates to similar technology levels. Complete design of a next-generation CO_2 removal system based on the selected technology with appropriate attributes for a 2-year mission with no resupply. Implement this design in the fabrication of a technology demonstration to fly on ISS by 2019.

All three goals require accurate selection of the superior sorbent (zeolite, silica gel, alumina, solid amine, etc.). However, goals 2 and 3 also require selection of the superior process (four-bed molecular sieve, pressure swing adsorption, temperature swing adsorption, etc.) coupled with the superior sorbent format (clay-bound pellets, polymer-bound monolith, honeycomb monolith, sorbent coated metal, etc.).

2.3 Virtual Design of Gas Separation Systems

The fundamental performance aspects of many sorbents (such as surface area, equilibrium working capacity, and selectivity) have been used as metrics to rank their potential superiority in a particular application. However, these metrics can provide conflicting data regarding which sorbent is superior, and have been shown to be unreliable in predicting superiority in process performance for post-combustion CO_2 capture (Haghpanah et al., 2014).

For spacecraft CO_2 removal, an equally complex application, standard figures of merit alone will not be used to optimize the processes involved. Rather, computer simulations that capture the key physics of the process, including coupled heat and mass transfer in porous media, must be applied. A large number of parametric simulations (also referred to as virtual tests) are required to converge on the optimal solution. Parametric hardware testing could also be employed, of course, but would be prohibitively expensive and time-consuming, severely limiting the number of options that can be explored.

In Figure 2.10, the integrated optimization approach is shown. Essentially, there are three steps in performance optimization: the first to screen out the worst performing sorbents, the second to obtain the "Startup Performance Optimization," and the third to obtain the final "Ranking of Sorbent/Process Systems."

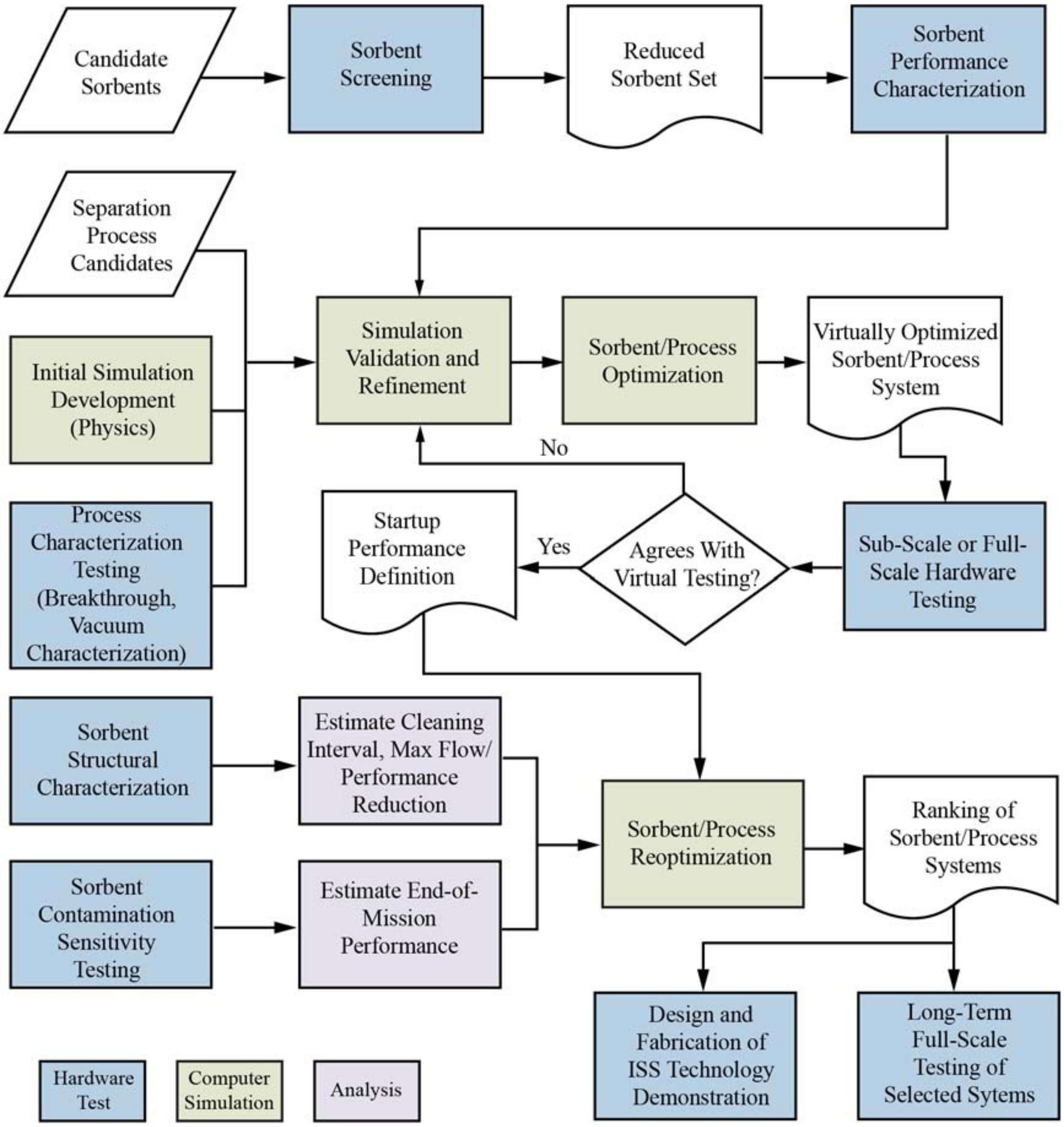

Candidate Sorbents
Sorbent Screening
Reduced Sorbent Set
Sorbent Performance Characterization
Separation Process Candidates
Initial Simulation Development (Physics)
Process Characterization Testing (Breakthrough, Vacuum Characterization)
Simulation Validation and Refinement
Sorbent/Process Optimization
Virtually Optimized Sorbent/Process System
No
Startup Performance Definition
Yes
Agrees With Virtual Testing?
Sub-Scale or Full-Scale Hardware Testing
Sorbent Structural Characterization
Estimate Cleaning Interval, Max Flow/ Performance Reduction
Sorbent Contamination Sensitivity Testing
Estimate End-of-Mission Performance
Sorbent/Process Reoptimization
Ranking of Sorbent/Process Systems
Design and Fabrication of ISS Technology Demonstration
Long-Term Full-Scale Testing of Selected Sytems
Hardware Test
Computer Simulation
Analysis

Figure 2.10 Integrated optimization approach flowchart (Knox et al., 2015a).

In the second step, sorbent and process characteristics are determined and simulated. Process parameters such as bed sizes, flow rates, cycle times, etc. are varied in a series of virtual tests in order to optimize the performance of each candidate process and sorbent combination. The top performing combinations are subjected to hardware testing to confirm results and validate the simulation. If required, simulation refinement and hardware tests are repeated until validation is successful. Upon completion of this step, the optimal process, sorbents, and process parameters are determined based on the characteristics of unused sorbents. This is the "Startup Performance Definition" in Figure 2.10. The number of sorbent/process systems carried forward to step 3 will depend on their relative ranking at this point.

As the ISS CDRA and commercial experience have revealed, long-term operation can lead to sorbent degradation, including capacity losses and attrition. These factors will be accounted for in the third step, where a second round of parametric virtual testing is conducted. As different sorbents have varying sensitivities to contamination and varying long-term stability, the initial ranking may change when end-of-life performance, flow losses, and maintenance requirements are factored in.

Development of an accurate mathematical model of these candidate systems is required in order to optimize design parameters such as bed size, cycle time, heater power and temperature set point, and vacuum pump parameters for the particular mission scenario without resorting to exhaustive (and expensive) hardware testing. The results from these studies will form the basis for the down-selection process shown in Figure 2.10. However, schedule constraints dictate that the computational time requirements are minimized, both due to the large number of system optimizations required for down-selection. Moreover, since for these cycle systems, a periodic state must be achieved for any one test case to be completed, which can require running a simulation for many cycles.

2.4 Literature Review of Adsorption Models

Appendix A provides the results of a literature survey of recent fixed-bed gas adsorption models. Those selected for inclusion in Appendix A were published in the year 2000 or later, are generally based on either the 1-D axially dispersed plug flow equation or the 1-D plug flow equation, and include a description of the experimental data used to validate the model. The subset of models meeting these criteria represents about half of the original set, but provide a representative cross-section of all those published. The data provided in Appendix A is summarized below.

2.4.1 Applications

The model applications range from post-combustion carbon capture (models 1, 3, 10, 13, 15, and 16) to industrial purity (2, 9, 11, 12) and dehumidification applications (4, 5, 7, 8, 18, and 19) to spacecraft CO_2 control (3, 6) to thermochemical energy storage (17). Thus, it is evident that the 1-D modeling approach is used across many disciplines.

2.4.2 Experimental System

The experimental system in this context refers to the sorbate/sorbent system. The use of standard zeolite types (3A, 4A, 5A, 13X, ZSM-5, and silicalite) predominates, representing one or more of the sorbents in 15 of the 19 publications. Carbons (including carbon molecular sieve) show up in six publications, silica gel in three, and hydrotalsite in one publication. It is clear that, although interest does exist in emerging sorbents, the standard zeolites, silica gels, and carbons capture the greatest interest in the model-based design of new and enhanced processes.

2.4.3 Spatial Dimensions

1-D models are used in 16 of the 19 publications represented. As stated earlier, although the 2-D models reviewed (models 4, 6, and 17) provided important information about the actual 2-D flow in small columns, none were actually applied to a model-based process design. The design and optimization of a cyclic process often requires simulating many repeated cycles (in order to attain a cyclic steady state response) such that the shorter simulation execution times required for a 1-D model are needed.

2.4.4 Tube Inner Diameter/Particle Diameter

The tube inside diameter (ID) to particle diameter ratio provides an indication of the impact of the tube wall on the overall flow through the tube. Since the particles, regardless of geometry (beads, pellets, or granular), cannot nestle at the flat wall surface, the packing density is lower near the wall. The void fraction approaches unity at the wall, and decreases until it approaches the core void faction at about five particle diameters from the wall (Nield and Bejan, 1992). Due to the higher void fraction at the wall, wall channeling (or higher flows) occurs in that location. For fixed-bed adsorption, the combination of lower sorbent density and higher flow result in earlier breakthrough at the walls. The effect of wall channeling is generally considered negligible at a tube to particle diameter ratio of 20 (Richardson et al., 2002). However, it will be shown in this work that, even for a ratio of 20, wall channeling is observed.

Nearly all the testing reported on in the publications in Appendix A was performed with test apparatus at or below a value of 20 for the tube to particle diameter ratio, with only publications 3, 14, and 17 being significantly higher. Most publications have a much lower ratio. It is evident that subscale testing such as breakthrough analysis is generally performed in small columns, which reduces both the sorbent required and experiment time. As a result, appropriately accounting for the wall channeling via an approach such as that shown in this work is required for accurate empirical derivation of the mass transfer coefficients discussed next.

2.4.5 Gas to Particle Rate Expression

The modes of mass transfer from the gas to the adsorbed state were discussed in section 2.1.5. As discussed, the LDF model shown in Equation (2.1) may be used to lump the four mass transfer mechanisms from the free stream to the interior of the zeolite crystal into one or two linear expressions. Most (12 of 19) of the models studied employed the LDF approach. Another approach used in publications 1, 7, and 9 is to condition the LDF term based on local concentration, temperature, or loading. The models in publications 4, 11, and 15 use variations of the bi-disperse approach for composite zeolite particles, which conceptually models both the diffusion in the macropores between the zeolite crystals and the diffusion within the zeolite crystal. Finally, the model described by publication 13 uses a variant of the LDF equation, the quadratic driving force (QDF) equation. Here, the terms inside Equation (2.4) are squared as shown in Equation (2.5):

$$\frac{\partial \overline{q}}{\partial t} = k_n(q^{*2} - \overline{q}^2) \ . \tag{2.5}$$

It should be noted that the approach presented in this work to obtain an empirical fit of the gas to particle rate coefficient is independent of the gas to particle rate model used.

2.4.6 Method to Determine Gas to Particle Rate

In the publications examined, 11 of 19 used breakthrough analysis to determine the gas to particle mass transfer rate. Breakthrough analysis consists of comparing the results of an adsorption breakthrough test with the model results and fitting the mass transfer coefficient to best match the test data. Publication 1 also examined the desorption profile, and found that modifications to the simple LDF approach was required. Various approaches were used to estimate the LDF value in five of the publications. These estimates are based on equations that sum the film resistance, micropore diffusion, and micropore diffusion or a subset thereof. However, each of these terms includes a diffusion term that must be empirically obtained, either through breakthrough analysis or other testing. For the publications reviewed, it was evident that a breakthrough analysis tended to provide a better match between model results and test data than the estimation approach, perhaps because the breakthrough analysis accounts for the all of fixed-bed physics.

2.4.7 Axial Dispersion

Axial dispersion was included in 14 of the 19 publications reviewed. The most common correlations used to estimate axial dispersion were that of Edwards and Richardson 1998 and Wakao and Funazkri 1978. These two correlations are also used in this work to bound the full range of values calculated by the five correlations considered in Chapter 4. Note that the value obtained from the Wakao and Funazkri correlation is 2 to 3 times greater than that obtained from the Edwards and Richardson correlation, thus the choice of an axial dispersion correlation provides a limited capability to fit the dispersion value.

2.4.8 Internal Profile Shown?

In Chapter 5, this work will show that for a specific sorbate/sorbent system (H_2O/5A) and by extension, other systems with a high distribution factor (which indicates the steepness or curvature of the equilibrium adsorption isotherm), non-physical simulation behavior, i.e., sharpening of the concentration front just prior to breakthrough, can occur even using commonly used correlations for the axial dispersion coefficient. This non-physical behavior is only evident upon inspection of the internal concentration history, but only the breakthrough curve is generally examined for the fitting process. For the publications reviewed, 13 provided only the breakthrough curve, and thus provided no evidence that the internal profile showed a physical behavior. Of these 13, 7 simulated H_2O/zeolite systems, which have a high distribution factor and thus a high potential for non-physical behavior.

CHAPTER III

3. EXPERIMENTAL

The development of computer simulations of adsorption processes requires two key experimental inputs. The first is capacity of the sorbent under consideration for the sorbate of interest when at equilibrium with a gas held at constant temperature and sorbate partial pressure. A series of these capacity measurements, or equilibrium adsorption capacity isotherms, may be fit to a standard correlation as described later. In the computer simulation, the driving force for adsorption is the difference between the current sorbent loading and the loading that would be attained if the sorbent were equilibrium with sorbate in the surrounding gas stream.

However, the sorbent does not come to equilibrium with the surrounding gas stream immediately. The second experimental input provides a limiting rate of adsorption, or mass transfer coefficient. A standard method for derivation of this mass transfer coefficient is via breakthrough testing and comparison of the test results with the computer simulation. In order to support the development of computer models and simulations of the ISS CDRA, a breakthrough test stand, the Molecular Sieve Material Bench Test (MSMBT), was fabricated at NASA's MSFC in the early 1990's (Knox, 1992a; Knox, 1992b; Mohamadinejad, 1999; Mohamadinejad et al., 2000).

3.1 Objective

The MSMBT test objectives are to characterize the adsorption material by mapping the dynamic adsorption and desorption characteristics for a range of flow rates and adsorbent concentrations. Characterization of both single-gas adsorption (CO_2, O_2, N_2, and H_2O with He carrier gas) and multi-gas adsorption (combinations of CO_2, H_2O, N_2, and O_2) was required. Results from single gas adsorption of CO_2 and H_2O on zeolite 5A, is reported on in this document. Inlet gas concentrations (held constant for a particular test) were varied in the range expected in manned spacecraft (0.7% to 1.5% by volume). Helium was chosen as the carrier gas since is not appreciably adsorbed by molecular sieves.

3.2 Test Apparatus

3.2.1 Packed Column

A small packed column with approximate dimensions of 2 inches in diameter and 20 inches long (Figures 3.1 and 3.2) was used during adsorption testing. Actual ISS CDRA CO_2 removal system beds have cross-sectional areas of about 80 square inches; this size would have required a prohibitive quantity of the helium carrier gas. The test flow rate was scaled to give gas velocities similar to that in the flight-like beds. Instrumentation of the packed column includes temperature probes and sampling tubes for measurements at sorbent material endpoints and one intermediate point. Two duplicate columns were fabricated of two-inch OD, 1/16-inch thick stainless steel 24 inches in length. Compression fittings (reducing unions with copper crush washers) provided a gas-tight enclosure. A spring seated on the union at one end maintained compaction of the bed material. The columns were packed using stainless steel, 40 mesh screens for containment of sorbent material and separation from the glass beads. Following placement of the glass beds to the proper level, filling of the test columns with adsorbent proceeded by securing the columns to a shaker table, pouring the sorbent material into the column in 200 ml batches, and vibrating for 10 seconds. Thermocouple probes and sampling tubes were inserted as required and aligned visibly during the filling process. When the filling process was complete, the second union nut was tightened until the seal quality was confirmed with a helium leak-check apparatus.

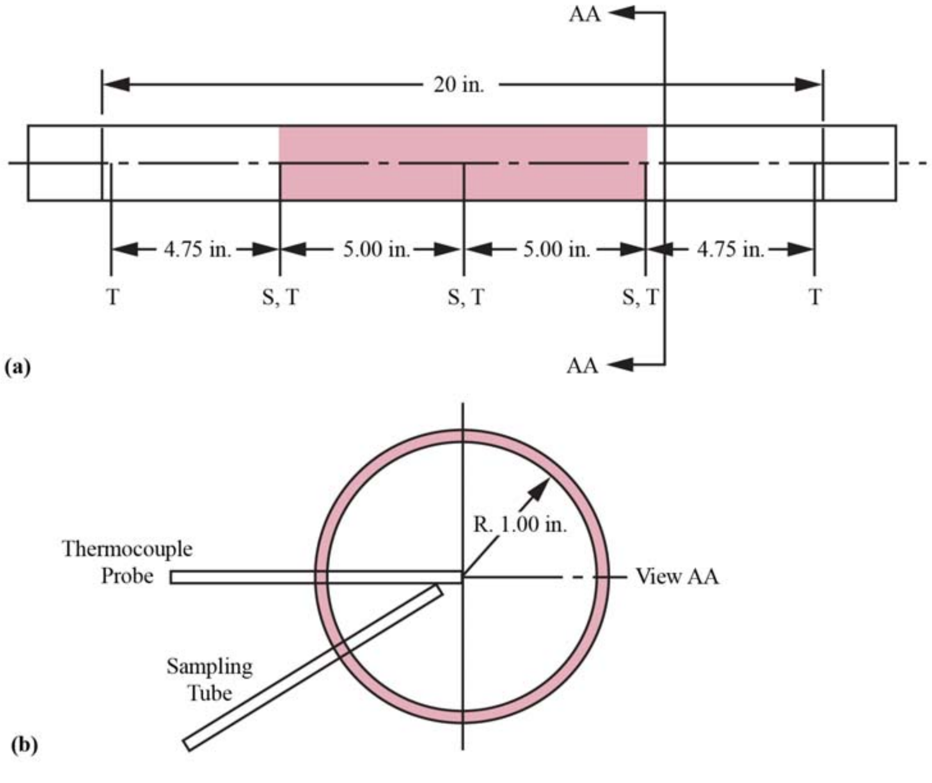


Figure 3.1 (a) Breakthrough test apparatus of Knox (1992) and (b) cross-sectional view of typical sampling location.

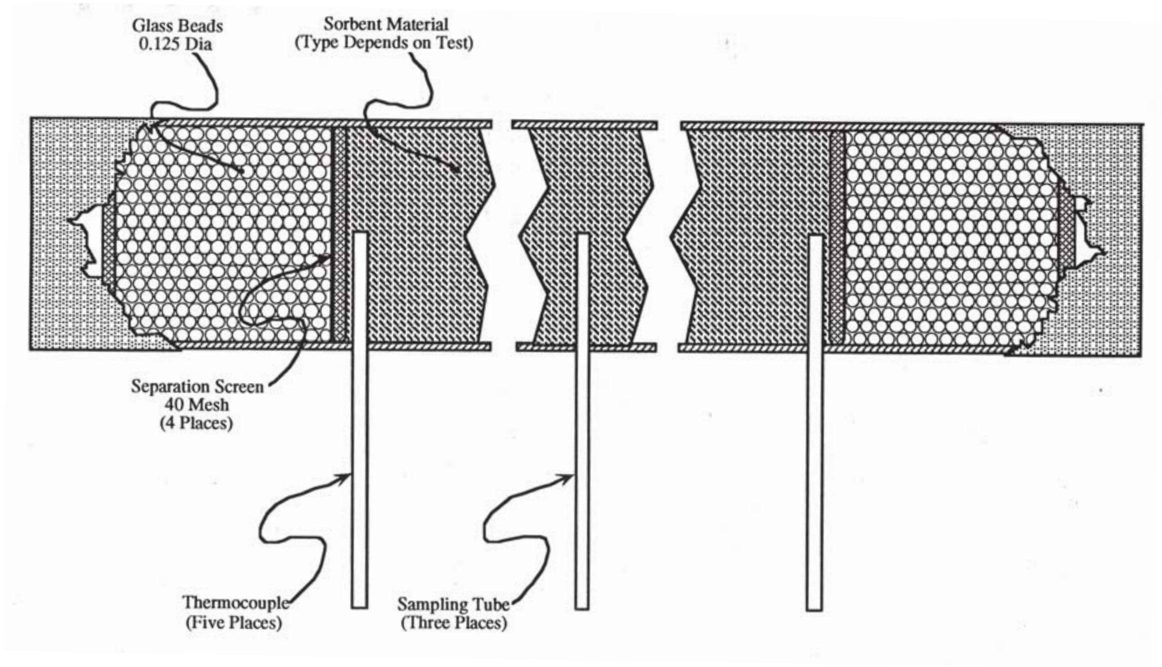


Figure 3.2 Fixed adsorbent bed cutaway (Knox, 1992b).

The adsorbent was a commercial zeolite 5A, Grace Davison Grade 522, in bead form. The adsorbent (Finn and Ho, 1995; Radenburg, 2013) and experimental apparatus properties are listed in Table 3.1.

Table 3.1 Properties of the adsorbent and fixed-bed.

Adsorbent		Fixed-bed	
Pellet radius (spherical)	$R_p = 1.16$ mm	Bed height	$L = 0.254$ m
Particle density	$\rho_s = 1180$ kg m^{-3}	Bed internal diameter	$R_i = 47.6$ mm
Skeletal density	$\rho_{sk} = 2040$ kg m^{-3}	Column wall thickness	$l = 1.59$ mm
Heat capacity	$c_{ps} = 920$ J kg^{-1} K^{-1}	Wall heat capacity	$c_{pw} = 475$ J kg^{-1} K^{-1}
Langmuir surface area	$A_L = 463$ m^2 g^{-1}	Wall density	$\rho_w = 7833$ kg m^{-3}

Brisk-Heat heat traces (12-foot length with 8.6-watt density per inch) were wrapped around the stainless steel tubing for the thermal desorption procedure. Adiabatic bed conditions were desired for the initial phase of testing, and to aid in the thermal desorption process. The column was first covered with Mansfield Q-fiber felt insulation, and then wrapped with a thermal blanket of Mansfield Min-K material. The insulation, approximately 1 inch thick in all, was used with satisfactory results.

3.2.2 Sensors

Instrumentation provided for continuous measurement of packed column inlet and outlet CO_2 concentration via Horiba model APBA-250E infrared CO_2 sensors. Flow rate was measured at various points with Porter and Edwards brand thermal mass flow meters, dew points via General Eastern optical dew point monitors (model Hygro-M1, with sensor 0111D), and pressure via a Heise digital pressure indicator and Viatran pressure transducers. Manufacturer-provided data for sensor accuracy and repeatability is provided in Table 3.2. The location of sensors and other equipment comprising the adsorbing apparatus are shown in Figure 3.3.

Table 3.2 Instrumentation with manufacturer-provided accuracy and repeatability.

Sensor Type	Sensor	Accuracy	Repeatability
Flow meter	Porter 250	± 1.0% full scale	± 0.2% full scale
Dew point sensor	General Eastern 1111H	± 0.2C of reading	± 0.05C of reading
Pressure sensor	Viatran	± 0.25 full scale	± 0.1 full scale
Thermocouple	Omega T-type	± 1.0C of reading	
Gas standard	N/A	± 1.0% of concentration	

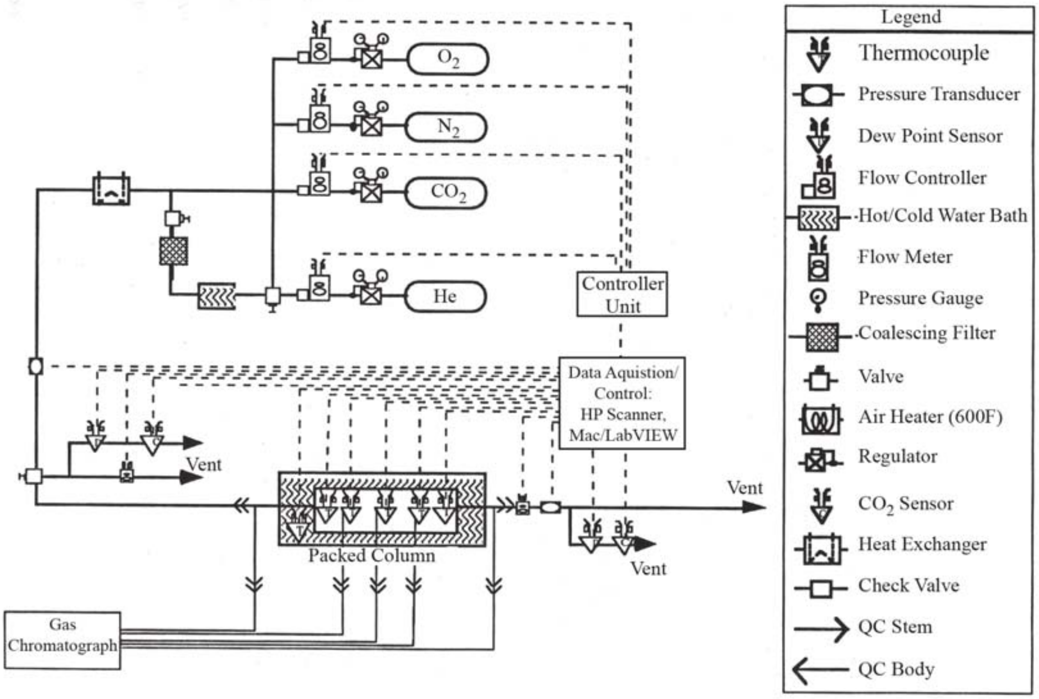


Figure 3.3 MSMBT adsorption schematic (Knox, 1992b).

The gas chromatograph (GC), a Shimadzu GC-14A with CR601 integrator, was used to sample at five sample port locations to determine gas constituent volumetric fractions during the adsorption runs. Figure 3.2 showed the location of sample probes 2, 3, and 4. Sample probes 1 and 5 are at the inlet and outlet of the column, respectively, as shown in Figure 3.3. To enable automated sampling at the five sampling locations, a Valco multi-position electric actuator (model E6) and six-port gas sampling valve were installed. The sixth port was used for the desorption apparatus. In addition, a Valco 2-position electric actuator (model E60) and sampling valve automatically controlled the sequence of purging of sampling lines and introduction of the gas sample from the constant volume loop to the GC column. To maintain constant pressure at the GC constant volume loop, a vacuum pump pulled the gases through the sample lines.

Different pressures at the sample port and different pressure drops in the sample lines were accommodated by adjusting a metering valve for consistent flow rate. A CR601 BASIC program was written for automated control of the GC and sample port selection.

The GC proved essential during the testing. The GC could receive gas from five sample port locations and determined gas constituent volumetric fractions during the adsorption runs. The infrared CO_2 sensors, intended to provide a continuous measurement at the outlet of the column, did not provide reliable data during the early portions of a test run due to the very low partial pressures at the outlet. The infrared sensor was difficult to calibrate for the He carrier gas due to this low partial pressure inconsistency. Therefore, the GC was used as the primary instrument for measurement of gas constituent volumetric fractions. The dew point sensors were limited by a minimum frost point of approximately -10°C (14°F). Measurements on the GC were found to be consistent down to the tens of parts per million range.

3.2.3 Data Acquisition

Data from the continuously measuring sensors were acquired with a Macintosh IIx computer running the LabVIEW II data acquisition and control software. The flow controller flow rates were also set through LabVIEW. Strip charts allowed monitoring of flow rates, inlet and outlet CO_2 partial pressures, and dew point conditions during the test. All data were continuously logged to an ASCII data file and could be accessed real time through a separate plotting package.

3.2.4 Support Equipment

Supporting equipment for the test included Porter models 250 and 201 thermal mass flow controllers (PID controlled solenoid valves with thermal mass flow meter input signal) to obtain the proper inlet constituent volumetric fractions. Pressurized "K" bottles supplied all gases. Ultra-high purity He (99.9999% pure) and high purity CO_2 (99.8% pure in the liquid state) was used for the first phase of testing. Helium is preferred since it does not appreciably adsorb onto the zeolite sorbents tested.

To control temperatures and dew points at the inlet, a saturation column was used for introduction of water vapor into the gas inlet stream. This column was, like the packed columns, 2 inches in diameter by 24 inches long, but customized with the required fittings and filled with beryl saddles to increase gas retention lime. A gas/liquid heat exchanger reduced the dew point to a specified level and provided control of inlet temperature. A NESLAB Instruments Coolflow refrigerated recirculator supplied cooling water to the heat exchanger.

3.3 Procedures

3.3.1 Adsorption Procedure

Preparation for adsorption test runs included placement of the test column into the adsorption apparatus and connection of the quick-disconnect fittings. Flow rates were set and allowed to stabilize with flow bypassing the column. With all conditions stable, the bypass valve was turned to initiate flow through the bed coincident with setting the data acquisition clock to zero. Gas chromatograph sampling, with the integrator clock reset to reflect test elapsed time, was then initiated using the CR601 BASIC routine. Quick-disconnects on the sample ports 1 through 4 were connected only as required to prevent excessive gas loss from the remainder of the column. Sample port 5 was left attached throughout the test. Testing continued until the outlet CO_2 or H_2O concentration closely approached the inlet concentration.

3.3.2 Desorption Procedure

Following bed packing, and before each adsorption test run, the packed columns were subjected to a desorption process. Shown schematically in Figure 3.4, the primary desorption method was heating while purging with nitrogen or helium gas. Heat was applied to the column with Brisk-Heat heat traces wrapped around and in contact with the stainless steel tubing. Due to the large unions and end effects, the ends of the column were observed to be cooler than the center. The heat trace was wrapped more at the ends to help counter this effect. Mansfield Q-Fiber felt and Min-K insulation wrapped around the column insured that most of the heat from the trace heated the bed pellets.

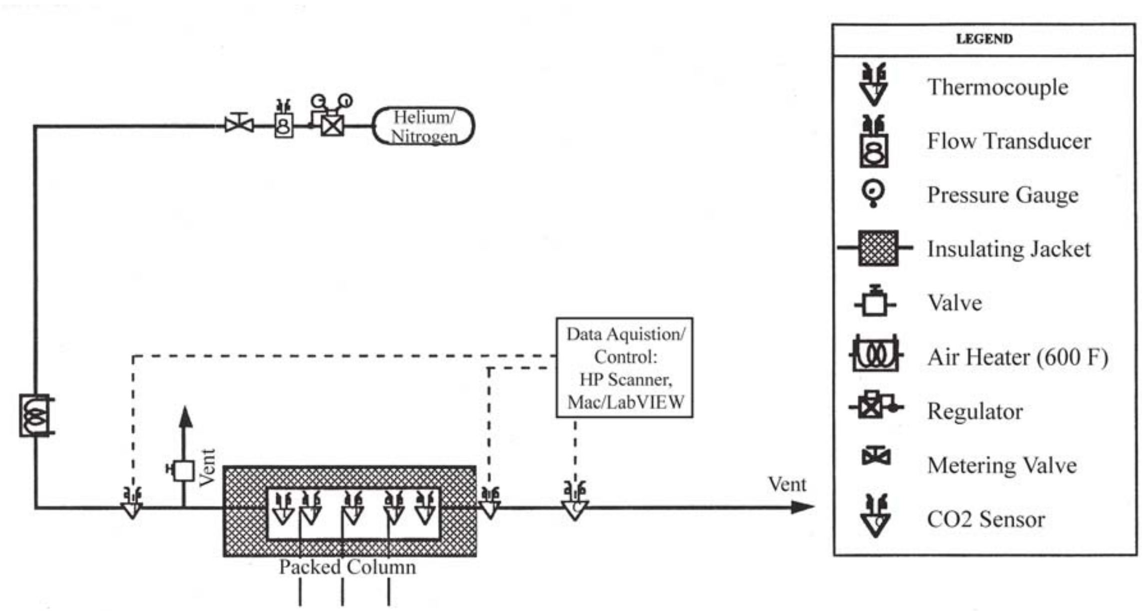


Figure 3.4 MSMBT desorption test apparatus (Knox, 1992b).

Due to thermal end effects of the columns, it was not possible to maintain the desired minimum desorption temperature (316°C) throughout the bed. A Eurotherm 808 temperature controller controlled the center point of the column at the set point temperature. The cooler inlet gas also reduced temperatures at the inlet end even at low (5 standard liters per minute or less) flow rates. Gas pre-heaters were used, though with little effect since the inlet gas could not be heated above 204°C due to the temperature limitation of the Swagelok Quick-Connect couplings.

The problem was resolved by configuring the bed for two-way flow during desorption. Temperatures in the downstream half of the bed exceeded 316°C (temperatures up to 400°C were observed) due to the heat convection of the flow. Flow in one direction was continued until GC sampling confirmed bed desorption (concentrations of less than 20 ppm CO_2 and N_2 were noted). Flow in the alternate direction then continued until temperatures in the new downstream half exceeded 316°C and the GC samples again confirmed desorption. Nitrogen gas was used for economy in the initial stages of desorption, followed by ultra-high purity helium gas to remove the nitrogen and allow complete bed desorption.

Vacuum and thermal conditioning was an alternate technique examined for bed desorption. A KNF model N035.3 SVP diaphragm pump reduced pressures in the bed to 2.2 kPa with accompanying heating via the heat trace. Although somewhat successful for rough desorption, the relatively high vacuum pressure was not sufficient to complete the process alone, and subsequently desorption was performed with the thermal/purge method described above.

3.4 Analysis to Determine Experimental Uncertainty

To appropriately apply the test data generated by the apparatus described above, whether for the interpretation of observed physical phenomenon or for computer simulation verification, the degree of uncertainty in the test results must be estimated. In this section, the data collection techniques are discussed in greater detail as required to describe the propagation of sensor errors via the data reduction equations. This is followed by the analysis used to estimate the uncertainty of the reported values of concentration and temperature. The complete uncertainty analysis performed in Mathcad 3.1 is provided in Appendix B.

3.4.1 Carbon Dioxide Breakthrough Data Reduction Procedure

Concentration data were obtained for the CO_2 breakthrough tests using a GC with a temperature conductivity detector (TCD). The GC was calibrated for CO_2 using a gas standard with 0.39994% CO_2 in nitrogen.

During the CO_2 breakthrough tests, it was observed that there was a variation in the inlet concentration as measured by the GC and that this variation could be correlated with changes in inlet pressure. Pressure variations at different ports is due to the differences in the differential pressure, since the downstream bed length is different at each sample port.

To correct for the variation in GC concentrations, pressure at each port was recorded prior to each test with nitrogen flowing at the same flow rate as during the test. Gas chromatograph measurements were then taken using a certified CO_2 standard of 0.39994% in nitrogen. While performing the GC analyses, the calibration gas pressure was varied with the K-bottle regulator. Based on these data, a relationship between total pressure and GC area was obtained. This relationship was used to determine a unique calibration value for each sample port.

Generally, a one- or two-point calibration of the GC is performed, where a gas standard with a known percentage of the gas of interest is injected into the GC and the resultant area measured. Since the area corresponding to a gas with zero concentration will be zero, the appropriate calibration curve is a line through the origin with the slope defined by the ratio of the calibration gas percentage to GC area. The GC area obtained during breakthrough testing is then multiplied by the slope or ratio to obtain the gas percentage.

As mentioned previously, variations in the GC area with total pressure were observed for the same calibration gas. To adjust the GC results, a correlation is developed to correct for total gas pressure at the port location of interest. As a result, the conversion from the GC measurement (area) to partial pressure of CO_2 requires the following steps:

1. For each port pressure, find the value of GC area that corresponds to the calibration gas percent CO_2.
2. Convert GC area for each test point to CO_2 percentage.
3. Convert from CO_2 percentage to CO_2 concentration for each test point.

For step 1, a linear correlation was developed based on GC area and pressure measurements based on the data shown in Table 3.3. Step 2 uses a single point calibration based on step 1 and the local port pressure. The data reduction equation for step 3 requires the total pressure and temperature.

Table 3.3 Test data taken to determine calibration constants for differing pressures.

Gas Bottle Pressure, psia	Gas Chromatograph Area
14.9	12036
	12048
15.3	12309
	12335

To calculate the slope, equation (3.1) is used:

$$m = \frac{GC_2 - GC_1}{P_2 - Y_1} \tag{3.1}$$

where m is the slope, P_1 and P_2 are the low and high pressure settings at the calibration gas bottle regulator, and GC_1 and GC_2 are the gas chromatograph areas measured at the low and high pressure, respectively. To determine the GC area that corresponds to each sample port pressure, equation (3.2) is used:

$$GC_{cal} = m\left(P_{cal} - P_1\right) + GC_1, \tag{3.2}$$

where GC_{cal} is the area corresponding to sample port pressure P_{cal}. To convert from GC area to CO_2 partial pressure, equation (3.3) is used:

$$f_{cal} = \frac{GS}{GC_{cal}} P_{cal}, \tag{3.3}$$

where f_{cal} is the conversion factor from GC area to CO_2 partial pressure and GS is the volume fraction of CO_2 in the gas standard (calibration gas). The final equation used to convert from GC area to CO_2 concentration is a form of the ideal gas law as shown in equation (3.4):

$$c = \frac{f_{cal} GC}{RT}, \tag{3.4}$$

where c is the CO_2 concentration, R is the ideal gas constant, and T is the local gas temperature.

3.4.2 Carbon Dioxide Breakthrough Test Uncertainty Analysis

The objective of this analysis is to determine the uncertainty in the reported results of CO_2 concentration. The uncertainty analysis must take into account the propagation of errors due to all equations used to convert the GC results, expressed as the area under the curve of the chromatogram for the gas of interest, to CO_2 concentration. This analysis must also account for errors in the temperature and pressure measurements as well as the error in the gas standard reported percentage. For this analysis, the relative standard uncertainty of the ideal gas constant ($5.7 \text{ x } 10^{-7}$) may be neglected as it is orders of magnitude below the next smallest uncertainty (Mohr, 2015). The uncertainty of each of the remaining variables is discussed below. In all cases, manufacturer's data on the appropriate sensor model is used in the analysis, as the test stand is not available, thus eliminating the option of performing direct calibration checks on the actual sensors.

An uncertainty value for the CG area is not provided by the manufacturer. An estimate may be made by calculating the standard deviation of the sample population (Coleman, 2009) based on the data from Table 3.3 via Equation (3.5):

$$S_X = \left[\frac{1}{N-1} \sum_{i=1}^{N} \left(X_i - \bar{X} \right) \right]^{\frac{1}{2}} = 18.38 , \qquad (3.5)$$

where S_X is the standard deviation, N is the number of individual readings of X_i and $\bar{X}$ is the sample mean. Since this is a (very) small population of two samples, the value of t corresponding to one degree of freedom is large, and results in a large value for the 95% confidence limit for the high pressure GC area and for the relative uncertainty:

$$P_x = tS_x = (12.7)(18.38) = 233.6 \; ; \quad P_{\%} = \frac{P_x}{\bar{X}} = \frac{233.6}{12322} = 1.9\% \, . \qquad (3.6)$$

To consider whether this magnitude of random uncertainty is appropriate or inflated due to the inherent uncertainty in a small sample, a literature search on the uncertainty of other GCs using TCD was conducted. In the natural gas industry, CO_2 in the raw gas must be removed. Zuas (2015) reported that the percent relative standard deviation for CO_2 as detected by a GC with TCD over a wide range of GC flow rates varied from 0.10% to 0.40%. Sutan (2013) found that combined relative uncertainty for CO_2 analyzed with a GC and TCD to be 0.3594%. Based on these references, a relative precision uncertainty of 0.4% is applied in this analysis.

Pressure transducer uncertainty is provided as a percent of full scale (Viatran 2010), which is taken to be 30 psia. Based on an email exchange with the manufacturer (Jerry Webb, e-mail communication, November 23, 2015) the specifications provided in Table 3.2 provide a pass/fail criterion as opposed to a statistical confidence interval. Thus, the manufacturer specifications are considered to be maximum values (3 standard deviations) following the approach of Nakos (2004). Errors are normalized to 2 standard deviations as per Lipak (1995).

The uncertainty associated with calibration gas, or gas standard, is considered next. For the bias error associated with Matheson standards, a 95% confidence interval is specified by Geib (2005). The gas standard typically used for testing of this type is a high grade standard. The higher grades for Matheson and Airgas have an overall uncertainty of 1% in this CO_2 partial pressure range. (Matheson, 2007; Airgas 2015).

For the uncertainty associated with the measured temperature values, the Omega value of ±1°C could be used (OMEGA, 2010). However, experience in other testing indicates that higher accuracy is generally achieved for standard T-type thermocouples. Testing was recently conducted on 5 T-type thermocouples using a drywell calibrator and a thermistor probe (Croan, 2015). The thermocouples were calibrated at five temperatures, from 25°C to 100°C. The uncalibrated thermocouple readings from the data acquisition system were compared with the thermistor probe. The thermistor probe uncertainty is negligible; it is shown by the manufacturer as ±0.01°C calibration uncertainty with ±0.01°C drift and ±0.005 repeatability per year (Fluke, 2015). Based on the thermocouple error analysis shown in Appendix B for the temperature range of the CO_2 breakthrough test, reasonable uncertainties are $B_T = 0.23C$ and $P_T = 0.36C$. To account for installation errors, the values shown are twice the calculated 95% confidence interval for both bias and precision uncertainty.

The Monte Carlo Method, or MCM, was used for the uncertainty analysis of the CO_2 concentration test results following the approach of Coleman (2009). Convergence of the method was reasonably assured by using 10,000 iterations. Error evaluation for each of the 41 data points must be performed individually due to individual variations in GC area and temperature. All error distributions are assumed to come from Gaussian distributions, and are created via the Mathcad "rnorm" function, which returns a vector of m random numbers having the normal distribution. Variables GC_1 and P_1 have a common correlated systematic error, and thus the systematic uncertainty distribution is calculated prior to the assembly of the uncertainty distribution for these variables as required. Data reduction equations (3.1), (3.2), and (3.3) were used to establish the uncertainty in f_{cal} for each of the four port locations at each iteration. Data reduction equation (3.4) was then applied for each of the 41 GC area data points along with the current temperature at that port to determine the 95% confidence interval uncertainty for each CO_2 concentration data point.

The relative uncertainty for the concentration measurement was calculated in Appendix B to be approximately 1.2% of the concentration. A typical distribution is shown in Figure 3.5; here the blue line shows a normal distribution based on the mean and standard deviation. It is noted that the concentration uncertainty, shown by the red bars, follows a normal distribution. The callouts show the values for one standard deviation about the mean (0.331 mol per cubic meter).

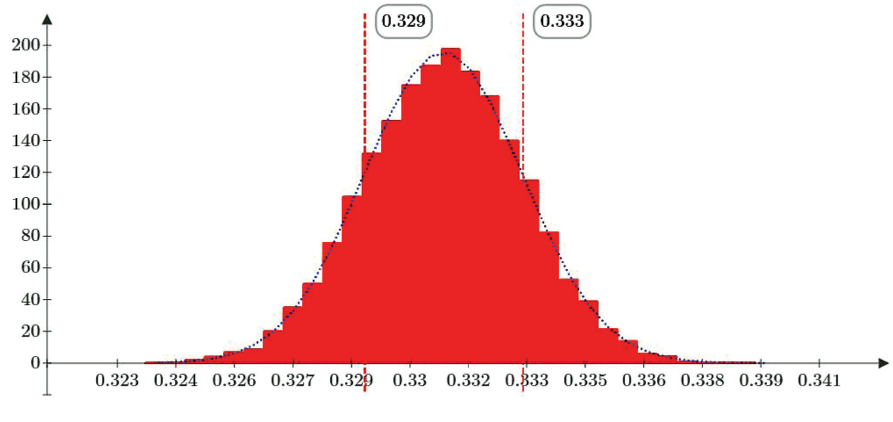


Figure 3.5 Typical uncertainty distribution for a CO_2 concentration measurement.

The uncertainty of the flow controller used to regulate the nitrogen gas flow was examined next (Porter Instrument Division, 2012). Based on phone conversation with the manufacturer (Patrick Flaherty, phone conversation, December 11, 2015), the specifications provided in Table 3.2 provide a pass/fail criteria as opposed to a statistical confidence interval. Thus, the manufacturer specifications are considered to be maximum values (3 standard deviations) following the approach of Nakos (2004). Errors are normalized to 2 standard deviations as per Lipak (1995). The resulting uncertainties for a 95% confidence interval are $B_{FM} = 0.33$ liter/minute and $P_{FM} = 0.07$ liter/minute. The overall uncertainty is then $U_{FM} = 0.34$ liter/minute. The relative uncertainty is 1.2%.

Experimental uncertainty in the temporal dimension for the breakthrough concentration and temperature data can be attributed primarily to uncertainty in the flow controllers. Due to the complexity of the adsorption process, the effect of flow rate changes on the timing of the experimental results may be most accurately estimated by using a computer simulation of the breakthrough test. The simulated inlet flow rate was varied two sigma above and below the flow meter set point for this analysis.

Simulation results with flow rates varied for CO_2 breakthrough test are shown in Figure 3.6. Each of the figures has three concentration or temperature curves. The center curve in each figure is the result with the set point flow rate. The curve that is earlier in time is the result from setting the flow rate lower by 2 sigma. The curve that is later in time is the result from setting the flow rate higher by 2 sigma. The average relative uncertainty in time resulting from the flow meter with a relative uncertainty of 1.2% is

1.30%. Each of the uncertainties associated with the CO_2 breakthrough test is summarized in Table 3.4.

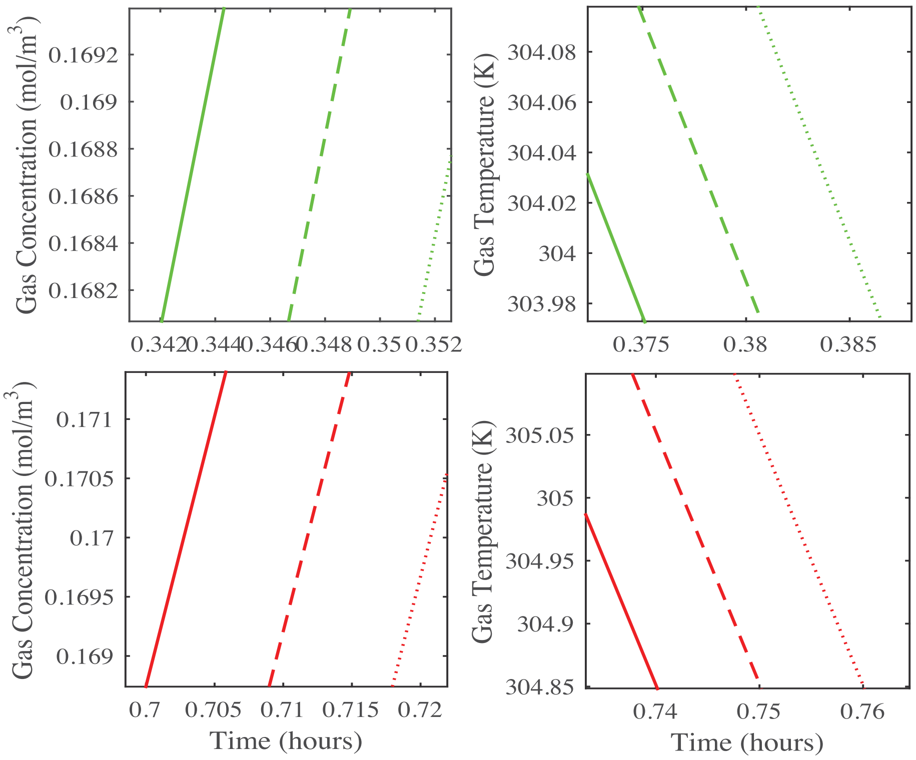


Figure 3.6 Temporal variation in CO_2 breakthrough test due to uncertainty in flow meter: Variation in concentration (left) and temperature (right) at midpoint (top) and exit (bottom).

Table 3.4 Estimated uncertainty for CO_2 breakthrough test data based on a 95% confidence interval.

Data Element	Uncertainty
Temporal Uncertainty	±1.3%
Concentration Uncertainty	± 1.2%
Temperature Uncertainty (K)	± 0.42 K

3.4.3 Humidity Breakthrough Test Uncertainty Analysis

The objective of this analysis is to determine the uncertainty in the reported results of water vapor concentration. The uncertainty analysis must take into account the propagation of errors due to all equations used to convert the GC results, expressed as the area under the curve of the chromatogram for the gas of interest, to water vapor concentration. This analysis must also account for errors in the temperature and pressure measurements as well as the error in the chilled mirror due point sensors used for GC calibration. For this analysis, as for the CO_2 breakthrough uncertainty analysis, the relative standard uncertainty of the ideal gas constant (5.7×10^{-7}) may be neglected as it is orders of magnitude below the next smallest uncertainty (Mohr, 2015). The uncertainty of each of the remaining variables is discussed below. Once again manufacturers' data on the appropriate sensor model are used in the analysis, as the test stand is not available, thus eliminating the option of performing direct calibration checks on the actual sensors.

Concentration data for the humidity tests were also obtained with the GC. The GC was calibrated for humidity using a General Eastern 1111H chilled mirror dew point sensor with Hygro-M1 electronics. Helium flowed through a temperature-controlled gas sparger to provide a gas stream at constant humidity. Sparger outlet humidity was measured with the dew point sensor. The GC sampled this gas stream to obtain an area measurement. This process was repeated twice at each of the three humidity settings to obtain a total of six area measurements as shown in Table 3.5.

Table 3.5 Gas chromatograph calibration data for water vapor.

Water Vapor per Dew Point Sensor (Pa)	Gas Chromatograph Area
530.9	9654
	9667
1278.4	22579
	12335
2090.8	36444
	36343

The conversion from the GC measurement (area) to vapor pressure of H_2O requires the following steps:

1. Convert dew point calibration measurements (as measured by the hygrometer) to vapor pressure.
2. Determine the conversion factor from GC area to H_2O vapor pressure.
3. Convert from H_2O vapor pressure to H_2O concentration.

For step 1, the Hardy formulation (Hardy, 1998) is used. Step 2 uses a three-point calibration to find a conversion factor. The data reduction equation for step 3 requires the vapor pressure and temperature. The Monte Carlo simulation approach is used to determine the uncertainty for each step of this analysis.

Conversion to vapor pressure is via the Hardy (1998) formulation. This conversion incurs less than 0.02% error in the worst incident compared to Table 3 in section 6.5 of the ASHRAE Handbook Fundamentals (Parsons, 1997). Based on the 1/4 rule of thumb, where 1/4 is the ratio of the uncertainty that may be neglected to the largest uncertainty (Coleman, 2009), this error is considered negligible.

The dew point sensor uncertainty (Optica, 2008) is provided without specification of the associated confidence interval. The manufacturer specifications are considered to be maximum values (3 standard deviations) per Nakos (2004). Errors are normalized to 2 standard deviations based on the approach suggested by Lipak (1995).

The bias errors used for the three calibration measurements will be correlated. For the Monte Carlo analysis, this means a common distribution for the error should be used in the calculation of the measurement distribution, so it is determined prior to the measurement distributions. The random (precision) errors will not be correlated and so are calculated individually for each measurement.

Gas chromatograph area data for each of the dew point calibration points are provided in the test data sheet (Mohamadinejad, 1995). As this is a calibration, the bias error is accounted for in the dew point sensor error. However, the GC repeatability should be considered. For this limited set of data, repeatability is calculated using the t value for $N = 2$ (Coleman, 2009).

Once again equations (3.5) and (3.6) are used to calculate relative uncertainties of 1.2%, 0.67%, and 2.5% for the three vapor pressure readings given Table 3.5. However just as in the CO_2 calibration case, these values will be biased high due to the large value of t. As described earlier, a relative uncertainty of 0.4% is more typical for CO_2; however, no similar literature was found providing typical values for H_2O analyses. Another approach is the commonly used single point calibration. Here, we use only the middle point, the first calibration point above the magnitude of the GC area (about 14,000) in the actual test, to allow use of interpolation rather than extrapolation. The relative uncertainty for this calibration point was determined to be 0.67%.

Step 2 is to determine the conversion factor from GC area to vapor pressure. In order to convert GC area measurements taken during the test to concentration values, the ratio of the calibration vapor pressure vs. GC area is calculated. The intercept is at the origin as appropriate for the relationship between the GC area and partial pressure. For the error analysis, the slope is calculated for each iteration of the Monte Carlo analysis. The relative uncertainty in the calibration factor was found in Appendix B to be 1.14%.

Step 3 is to convert from H_2O vapor pressure to H_2O concentration. The ideal gas law as shown in Equation (3.7) was used to find the H_2O concentration. All uncertainties have already been determined. (Uncertainties in the temperature measurement were discussed in the previous section.) The same Monte Carlo approach described above for the CO_2 breakthrough analysis is used to determine the 95% confidence interval for each H_2O concentration data point. The approach is also shown in detail in Appendix B. The resulting uncertainty for the H_2O concentration averaged 1.32%.

$$c = \frac{mGC}{RT}, \tag{3.7}$$

where c is the H_2O concentration, m is the factor used to convert GC area to H_2O vapor pressure, R is the ideal gas constant, and T is the local gas temperature.

As with the CO_2 breakthrough tests, experimental uncertainty in the temporal dimension for the breakthrough concentration and temperature data can be attributed primarily to uncertainty in the flow controllers. Once again, multiple simulation runs are used to determine the temporal uncertainty, with the simulated flow rates varied 2 sigma below and 2 sigma above the indicated flow controller set point. The results from the computer simulation are shown in Figure 3.7. For the H_2O breakthrough tests, the relative uncertainty in time resulting from the flow meter with relative uncertainty of 1.2% is 1.26%. This is a similar value to the 1.30% relative uncertainty observed for the CO_2 breakthrough case. The uncertainties associated with the H_2O breakthrough test are summarized in Table 3.6.

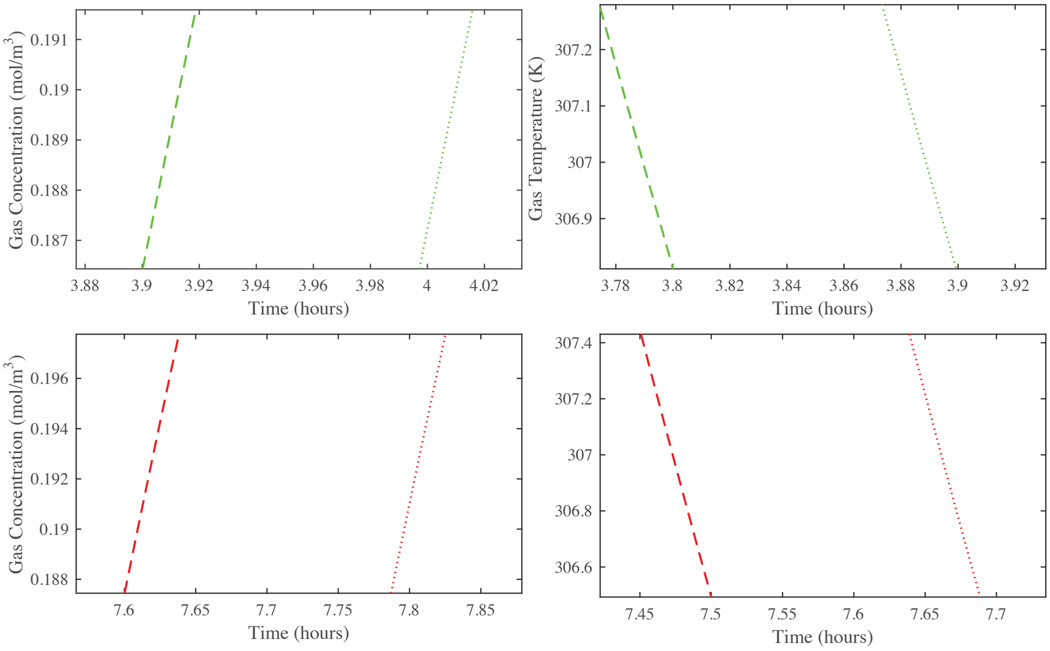


Figure 3.7 Temporal variation in water dioxide breakthrough test due to uncertainty in flow meter: Variation in concentration (left) and temperature (right) at midpoint (top) and exit (bottom).

Table 3.6 Estimated uncertainty for H_2O breakthrough test data based on a 95% confidence interval.

Data Element	Uncertainty
Temporal Uncertainty	±1.26%
Concentration Uncertainty	± 1.32%
Temperature Uncertainty (K)	± 0.42 K

CHAPTER IV

4. LIMITATIONS OF BREAKTHROUGH CURVE ANALYSIS IN FIXED-BED ADSORPTION

This work examined in detail the *a priori* prediction of the axial dispersion coefficient from available correlations versus obtaining both it and mass transfer information from experimental breakthrough data. Consequences may arise when doing so based on using a 1-D axially dispersed plug flow model and its associated Danckwerts outlet boundary condition. These consequences mainly included determining the potential for erroneous extraction of the axial dispersion coefficient and/or the LDF mass transfer coefficient from experimental data, especially when non-plug flow conditions prevail in the bed. Two adsorbent/adsorbate cases were considered, i.e., CO_2 and H_2O vapor in zeolite 5A, because they both experimentally exhibited significant non-plug flow behavior, and the H_2O-zeolite 5A destroyed the expected CPB when modeled with the 1-D axially dispersed plug flow model. Overall, this work showed that it was possible to extract accurate mass transfer and dispersion information from experimental breakthrough curves using a 1-D axial dispersed plug flow model when they were measured both inside and outside the bed. To ensure the extracted information was accurate, the inside the bed breakthrough curves and their derivatives from the model were plotted to confirm whether or not the adsorbate/adsorbent system was exhibiting

CPB or any concentration front sharpening near the bed exit. Even when concentration front sharpening was occurring with the H_2O-zeolite 5A system, it was still possible to use experimental inside and outside breakthrough curves to extract fundamental mass transfer and dispersion information from the 1-D axial dispersed plug flow model based on the systematic methodology developed in this work.

4.1 Introduction

A major issue associated with achieving a fully predictive simulation of gas adsorption in fixed-beds is finding values for the free (i.e., unknown) parameters in either the mass balance or energy balance partial differential equations. It is advantageous to reduce the number of free parameters by using verified correlations to determine the mass and heat transfer coefficients *a priori*. However, when parameters cannot be determined *a priori*, simplifications are utilized that lump multiple heat or mass transfer mechanisms together, with the corresponding coefficients (i.e., lumped free parameters) potentially losing their meaning. These coefficients are necessarily determined empirically by fitting to experimental data (Knox et al., 2016a).

One such simplification is the 1-D axially dispersed plug flow model that is frequently used to simulate fixed-bed adsorption processes (Beeyani et al., 2010; Ahn and Lee, 2003; Chou and Chen, 2004; Ruthven, 1984; Yang, 1987). The axial dispersion term in this model leads naturally to the ubiquitous use of the Danckwerts boundary condition at the outlet of the bed. This is a Neumann boundary condition that can be derived rigorously when pure axial molecular diffusion is accounted for with continuity of concentration and mass flux across the outlet boundary (Coppola and LeVan, 1981). Two issues arise from the use of this simplified 1-D model that limit its utility.

The first issue is associated with the *a priori* prediction of the axial dispersion coefficient from available correlations. Numerous correlations are available based on the particle Peclet number, gas velocity, and pellet diameter (Ruthven, 1984; Wakao and Funazkri, 1978; Edwards and Richardson, 1968; Wicke, 1973; Wen and Fan, 1975). The actual mechanisms that contribute to axial and radial mixing in fixed-beds are necessarily lumped into the axial dispersion term. These mechanisms include turbulence, flow splitting and rejoining around particles, Taylor dispersion, channeling, and wall effects (Aviles and LeVan, 1991; Mahle and Friday, 1991; Russell and LeVan, 1997). Not only do none of these correlations account for all the different dispersion mechanisms mentioned above, but also there is considerable variance in the values obtained from them. In addition, the 1-D axially dispersed plug flow model only accounts for dispersion mechanisms that fall within the framework of the plug flow condition.

Nevertheless, axial dispersion in a fixed-bed adsorber cannot be ignored because it reduces the adsorption process efficiency. To capture its influence, the axial dispersion coefficient is typically estimated from one of the correlations available in the literature. However, if the experiments are not designed properly, the information obtained from them may be erroneous.

The second issue concerns the development of CPB inside the bed, wherein the concentration front (i.e., concentration bed profile) propagates through the bed without changing its shape. Constant pattern behavior has been widely established theoretically (Ruthven, 1984; LeVan and Carta, 1997) and confirmed experimentally for systems with favorable Type I isotherms (Mahle and Friday, 1991). However, an unusual situation may arise when modeling a fixed-bed adsorber with the 1-D axially dispersed plug flow model because of its inherent assumptions. For example, when used to analyze experimental data, solutions obtained for fixed-bed adsorption with axial diffusion described by the Fickian model may produce breakthrough curve sharpening for both shallow and deep beds (Coppola and LeVan, 1981; Coppola and LeVan, 1983).

This concentration front sharpening effect has been largely ignored in the literature, except for a few studies (Coppola and LeVan, 1981; Mahle and Friday, 1991). In many simulation studies, neither the internal concentration histories nor the bed concentration profiles are shown to verify CPB. Quite possibly, the breakthrough curve from the model is just numerically fitted to the experimental breakthrough curve to obtain mass transfer information, like the LDF mass transfer coefficient, while being limited to dispersion coefficients predicted from known correlations (Ding and Alpay, 2000; Ahn and Lee, 2003; Jee et al., 2005; Delgado et al., 2006; Nastaj and Ambrozek, 2015; Park et al., 2000; Rouf, 1998; Soares et al, 2005; Bastos-Neto et al., 2011; Mette et al., 2014). The results obtained in such cases may be erroneous because they may have been obtained from experimental results dominated by non-plug conditions or from simulated breakthrough curves that deviated from the expected and real CPB physics due to concentration front sharpening occurring near the exit of the bed.

The objective of this section is to examine in detail the issues described above. These issues are the *a priori* prediction of the axial dispersion coefficient from available correlations versus obtaining it and also the LDF mass transfer coefficient from experimental breakthrough data and the consequences that may arise when doing so based on using the 1-D axially dispersed plug flow model and its associated Danckwerts outlet boundary condition. Two adsorbent/adsorbate cases are considered, i.e., CO_2 and H_2O vapor in zeolite 5A, which illuminate these issues.

4.2 Mathematical Model

4.2.1 Gas-Phase Mass Balance

The commonly employed 1-D axially dispersed plug flow model is;

$$\frac{\partial c}{\partial t}+\left(\frac{1-\varepsilon}{\varepsilon}\right)\frac{\partial \overline{q}}{\partial t}-D_L\frac{\partial^2 c}{\partial x^2}=-\frac{\partial \upsilon_i c}{\partial x} \quad , \tag{4.1}$$

where D_L is the axial dispersion coefficient, x is the axial coordinate, υ_i is the interstitial velocity, t is the time, ε is the bulk void fraction, $\overline{q}$ is the average adsorbed-phase concentration of the adsorbate and c is the gas-phase concentration of the adsorbate defined according to ideal gas law, i.e.,

$$c=\frac{p}{RT_f} \quad , \tag{4.2}$$

where p is the partial pressure of the adsorbate, T_f is the fluid (gas phase) temperature, and R is the universal gas constant.

Equation (4.1) is derived from a differential mass balance based on the following assumptions: All mechanical dispersion effects are lumped together with molecular diffusion in the axial dispersion term. Plug flow is assumed, i.e., there is no gradient of velocity, concentration, temperature, or porosity in the radial direction. Velocity in the axial direction is not compensated for loss of adsorbate since the adsorbate gas-phase mole fraction is << 1. Velocity is temperature compensated per the ideal gas law.

The boundary conditions are shown in Equation (4.3). A constant flux boundary condition is used for the inlet concentration and the Danckwerts boundary condition is used for the outlet (Danckwerts, 1995) respectively;

$$-D_L \left.\frac{\partial c}{\partial x}\right|_{x=0} = \frac{\upsilon_s}{\varepsilon}\left(c_0 - c\right) \quad \text{and} \quad \left.\frac{\partial c}{\partial x}\right|_{x=L} = 0 \tag{4.3}$$

where c_0 is the concentration and υ_s is the superficial velocity (both far upstream), and L is the bed height.

4.2.2 Adsorbed-Phase Mass Balance

The transport of the adsorbate from the gas phase to the adsorbed phase is described by a LDF approximation (Glueckauf, 1955), as

$$\frac{\partial \overline{q}}{\partial t} = k_n(q^* - \overline{q}), \tag{4.4}$$

where k_n is the LDF mass transfer coefficient and q^* is the equilibrium adsorbed-phase concentration that corresponds to the adsorbate gas-phase partial pressure (p) at the sorbent temperature (T_s) based on the equilibrium adsorption isotherm shown later. The LDF approximation is frequently used with the 1-D axially dispersed plug flow model in the analysis of adsorption processes. All the transfer resistances, including micropore and macropore resistances and surface diffusion, are lumped into the LDF mass transfer coefficient. If the mass transfer resistance is assumed to be a single mass transfer mechanism that is dominant and constant throughout the adsorption process, then this approach is valid. Moreover, it is well known that the LDF approximation incurs little error for most commercial gas phase cycle adsorption processes when empirically derived (Yang, 1997; Sircar and Hufton, 2000).

4.2.3 Energy Balance

For the adsorbent/adsorbate systems and concentrations studied, significant deviations from isothermal conditions were observed (Knox, 1992b; Mohamadinejad, et al., 1996). Therefore, energy balance equations for the gas (fluid), adsorbent, and column wall are included in the model. The gas-phase energy balance includes transient heat storage, gas conduction, gas convection, and heat transfer from the adsorbent to the column wall via Newton's law of cooling (Bird et al., 2002), and is;

$$\varepsilon a_f \rho_f c_{pf} \frac{\partial T_f}{\partial t} - \varepsilon a_f k_{eff} \frac{\partial^2 T_f}{\partial x^2} = -\varepsilon a_f \rho_f \upsilon_i c_{pf} \frac{\partial T_f}{\partial x} + a_f a_s h_s \left(T_s - T_f\right) + P_i h_i \left(T_w - T_f\right) , \qquad (4.5)$$

where a_f is the superficial free flow area, ρ_f is the gas-phase density, c_{pf} is the gas-phase heat capacity, k_{eff} is the effective gas-phase conductivity, a_s is the pellet external surface area per unit volume, h_s is the adsorbent to gas heat transfer coefficient, T_s is the adsorbent temperature, P_i is the inner perimeter of the column, h_i is the heat transfer coefficient between the column wall and the gas-phase, and T_w is the column wall temperature.

The boundary conditions for the gas-phase energy balance are shown in Equation (4.6). A constant flux boundary condition is used for the gas inlet and a Danckwerts-type boundary condition is used for the outlet that specifies no thermal dispersion, respectively.

$$-k_{eff}\left.\frac{\partial T_f}{\partial x}\right|_{x=0} = \rho_{f0}\upsilon_{i0}c_{pf0}\left(T_0 - T_f\right) \text{ and } \left.\frac{\partial T_f}{\partial x}\right|_{x=L} = 0, \tag{4.6}$$

where T_0, ρ_{f0}, and c_{pf0} are the temperature, density, and heat capacity far upstream.

The adsorbent energy balance is provided in Equation (4.7). This equation includes transient energy storage and heat transfer from the gas phase via Newton's law of cooling and the heat of adsorption:

$$(1-\varepsilon)\rho_s c_{ps}\frac{\partial T_s}{\partial t} = a_f a_s h_s\left(T_f - T_s\right) - (1-\varepsilon)a_f\lambda\frac{\partial q}{\partial t} \tag{4.7}$$

where ρ_s is the adsorbent density, c_{ps} is the adsorbent heat capacity, and λ is the isosteric heat of adsorption.

The column wall energy balance is similar and includes transient energy storage, heat conduction, and heat transfer from the internal gas phase to the ambient environment via Newton's law of cooling (Bird et al., 2002) and is written as

$$a_w\rho_w c_{pw}\frac{\partial T_w}{\partial t} - a_w k_w\frac{\partial^2 T_w}{\partial x^2} = p_i h_i\left(T_f - T_w\right) + P_o h_o\left(T_a - T_w\right), \tag{4.8}$$

where a_w is the cross-sectional area of the column, ρ_w is the column wall density, c_{pw} is the column wall heat capacity, k_w is the column wall conductivity, P_o is the column wall outer perimeter, T_a is the ambient temperature and h_o is the column wall to ambient heat transfer coefficient.

4.2.4 Equilibrium Adsorption Isotherms

The Toth equilibrium adoption isotherm (Toth, 1971) was used to calculate the equilibrium adsorbed-phase loading corresponding to the adsorbate gas-phase partial pressure. The single gas Toth isotherm is shown in Equation (4.9):

$$n = \frac{ap}{\left[1+(bp)^t\right]^{1/t}}; \qquad b = b_0 \exp(E/T); \qquad a = a_0 \exp(E/T); \qquad t = t_0 + c/T \quad , \qquad (4.9)$$

where n is the loading of the adsorbate in the adsorbed phase, a is the saturation capacity, b is an equilibrium constant, and t is the heterogeneity parameter. Parameters a, b and t are temperature dependent as shown, whereas a_0, b_0, and t_0 are system dependent adsorption isotherm parameters. A comparison of the Toth equation and the experimental data are shown in Figure 4.1; the corresponding adsorption isotherm parameters were obtained from Wang and LeVan (2009) and given in Table 4.1.

In the next four sections, the correlations used to obtain the mass and heat transfer coefficients in the mass and heat balance equations are discussed. It should be noted that these coefficients are calculated based on the fixed-bed inlet conditions as opposed to local conditions. This approach is taken in the interest of obtaining faster execution times, and because 1-D correlations in general are derived from experimental data using fixed-bed inlet conditions, not local bed conditions, for the independent variables. The simulation is shown to be insensitive to the use of constant coefficients in Section 4.4.2.

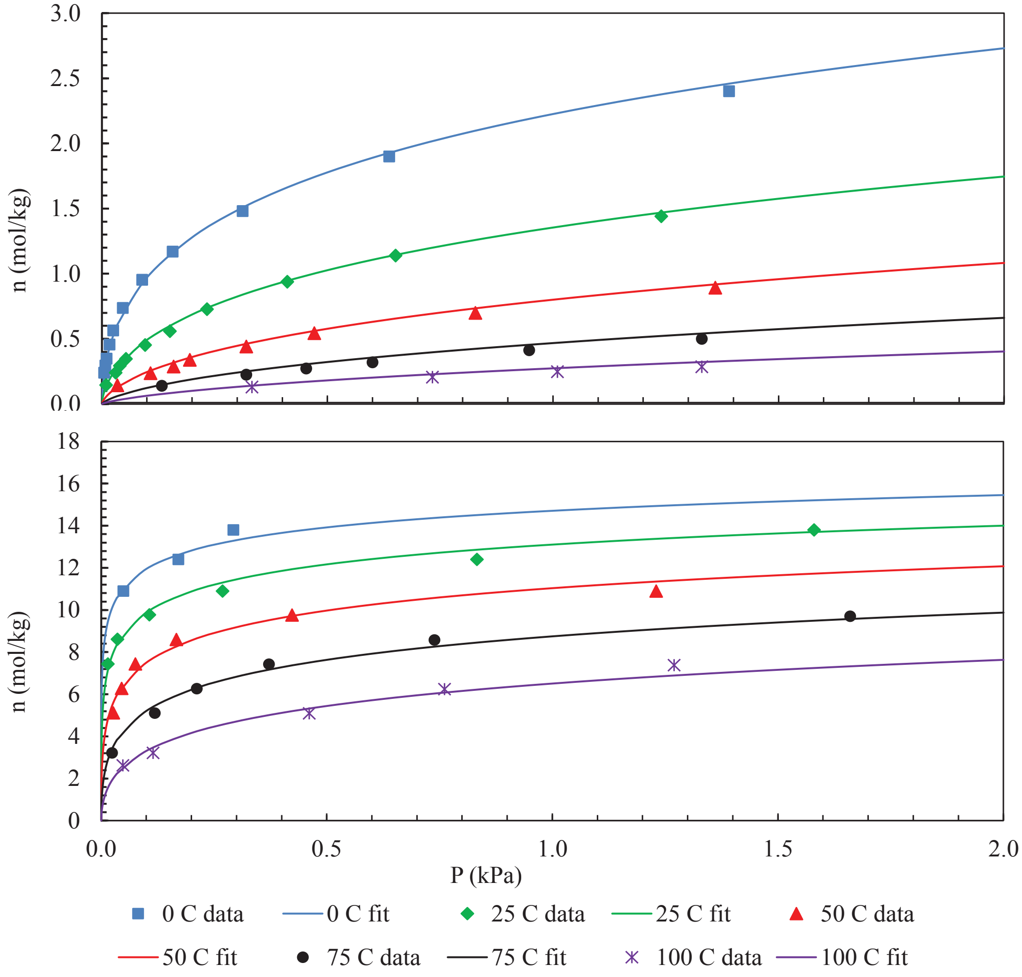


Figure 4.1 Equilibrium adsorption isotherms for CO_2 (top) and H_2O vapor (bottom) on zeolite 5A at temperatures from 0°C to 100°C as indicated. Symbols represent experimental data; Toth isotherm fits are shown as lines (Wang and LeVan, 2009).

Table 4.1 Toth equation equilibrium adsorption isotherm parameters for CO_2 and H_2O vapor on zeolite 5A (Wang and LeVan, 2009).

	a_0	b_0	E	t_0	c
system	mol · kg^{-1} · kPa^{-1}	kPa^{-1}	K		K
CO_2/5A	9.875x10^{-7}	6.761x10^{-8}	5.625x10^{3}	2.700x10^{-1}	–2.002x10^{1}
H_2O/5A	1.106x10^{-8}	4.714x10^{-10}	9.955x10^{3}	3.548x10^{-1}	–5.114x10^{1}

4.2.5 Axial Dispersion Coefficient

Five different correlations that describe axial dispersion in packed beds based on the pellet Peclet (*Pe*) number as a function of the product of the Reynolds (*Re*) and Schmidt (*Sc*) numbers are shown in Equation (4.10a), Wakao and Funazkri (1978), Equation (4.10b), Edwards and Richardson (1968), Equation (4.10c), Wicke (1973), Equation (4.10d), Ruthven (1984) and Equation (4.10e), Wen and Fan (1975):

$$\frac{1}{Pe}=\frac{20}{\varepsilon}\left(\frac{D}{2\upsilon R_p}\right)+\frac{1}{2}=\frac{20}{ReSc}+\frac{1}{2} \tag{4.10a}$$

$$\frac{1}{Pe}=\frac{0.73\varepsilon}{ReSc}+\frac{1}{2\left(1+\dfrac{13\cdot 0.73\varepsilon}{ReSc}\right)} \quad 0.0377<2R_p<0.607\text{ cm} \tag{4.10b}$$

$$\frac{1}{Pe}=\frac{0.45+0.55\varepsilon}{ReSc}+0.5 \tag{4.10c}$$

$$\frac{1}{Pe}=\frac{0.7\varepsilon}{ReSc}+0.5 \tag{4.10d}$$

$$\frac{1}{Pe}=\frac{0.3\varepsilon}{ReSc}+\frac{0.5}{\left(1+\dfrac{3.8}{ReSc}\right)} \quad 0.008<Re<400 \quad 0.28<Sc<2.2\ . \tag{4.10e}$$

The definitions of the *Re*, *Sc*, and *Pe* numbers are provided in Equation (4.11):

$$Re=\frac{2\rho_f\varepsilon\upsilon_i R_p}{\mu} \qquad Sc=\frac{\mu}{\rho_f D} \qquad D_L=\frac{2\upsilon_i R_p}{Pe}\ , \tag{4.11}$$

where R_p is the pellet radius, *D* is the fluid diffusion coefficient, and μ is the fluid viscosity. As mentioned above, the large variation in the ranges of values and the trends provided by these five different correlations for axial dispersion is well known.

Determination of molecular diffusivity for binary gas mixtures is based on kinetic theory and the corresponding state approach (Slattery, 1958). For the diffusion of one gas in a mixture, the effective molecular diffusivity is defined by analogy with binary diffusivity (Hougen, 1947) and can be applied to the special case where the other gases in the mixture travel at the same velocity (Wilke, 1950). This approach is outlined by Bird (2002).

The Lucas method for pure gas viscosity uses critical gas properties to estimate viscosity, then applies corrections factors to account for quantum effects and polarity of the gas molecules (Lucas, 1980; Lucas, 1984). In the case of mixture viscosity, pseudo-critical properties are estimated from pure component critical properties and the molar fractions. This method was selected as the required properties are available for all gases of interest and the error associated with this approach is less than 2% a mixture of nitrogen and carbon dioxide (Poling, 2001).

4.2.6 Gas-Phase Properties: Heat Transfer

The gas-phase heat capacity (c_p) is calculated based on parameters obtained from Reid et al. (1987). The polynomial equation used is shown in Equation (4.12):

$$c_p = a_o + a_1 T_f + a_2 {T_f}^2 + a_3 {T_f}^3 \ , \tag{4.12}$$

where a_0 through a_3 are the four parameters fitted to experimental heat capacity values. The mixture gas heat capacity was obtained via a weighted average on a molar basis.

4.2.7 Correlations for Heat Transfer Coefficients

The heat transfer coefficient (h_s) from the gas phase to the pellet is calculated using a film diffusion relationship developed for mass transfer by Wakao et al. (1979) and similarity by Ruthven (1984). These relationships are given in Equations (4.13) and

(4.14), respectively. Equations (4.13) and (4.14) have been verified experimentally for fluid-to-particle heat transfer (Ruthven, 1984; Wakao and Kaguei, 1982):

$$Sh = 2 + 1.1Sc^{\frac{1}{3}}Re^{0.6} \tag{4.13}$$

$$h_s = \frac{ShD}{2R_p} \tag{4.14}$$

where *Sh* is the Sherwood number defined in Equation (4.13).

The heat transfer coefficient h_i from the gas phase to the interior wall of the column is calculated based on the correlation of Li and Finlayson (1977) for 1-D models, as shown in Equation (4.15):

$$h_i = \frac{k_f}{2R_i}Nu \quad \text{with} \quad Nu = 2.03Re^{0.8}\exp\left(-6\frac{R_p}{R_i}\right), \tag{4.15}$$

where *Nu* is the Nusselt number.

4.2.8 Effective Thermal Conductivity

The Krupiczka equation (Li and Finlayson, 1977; Wakao and Kaguei, 1982), given by Equation (4.16), is used to calculate the effective thermal conductivity (k_e) of a quiescent bed of spherical particles:

$$k_e = k_f\left(\frac{k_s}{k_f}\right)^n \quad \text{with} \quad n = 0.280 - 0.757\log_{10}\varepsilon - 0.057\log_{10}\left(\frac{k_s}{k_f}\right). \tag{4.16}$$

The effective axial thermal conductivity for a fixed-bed of spherical particles with flow is calculated from the correlation of Yagi et al. (1960), as shown in Equation (4.17); it was verified against test data by Kaviany (1995):

$$k_{eff} = k_f \left(\frac{k_e}{k_f} + 0.75 Pr Re \right) \quad \text{where} \quad Pr = \frac{c_p \mu}{\rho_f k_f} \ , \tag{4.17}$$

where *Pr* is the Prandtl number. The gas-phase conductivity (k_f) is calculated based on the analogy between mass and heat transfer and the fluid diffusivity as;

$$k_f = D c_p \ . \tag{4.18}$$

4.3 Numerical Approach and Validation

The overall numerical approach includes the use of a custom, interactive program (the virtual adsorption test suite, or VATS) developed in the Matlab® software package (version R2015b) for simulation definition, front-end calculations, creation of a COMSOL® model based on the simulation definition, and initiation of a COMSOL® simulation. The COMSOL® Multiphysics® software package (Version 5.1) is used to solve the mass and energy balance partial differential equations as shown in Section 4.4. The VATS program then performs post-simulation summary calculations based on the COMSOL® transient results, and provides user-specified plots of breakthrough simulations. The VATS Matlab® code is provided as Appendix C.

4.3.1 Code Validation

The VATS program is validated in two respects. First, correlations used to estimate two important physical properties that are widely used in other correlations (diffusion and viscosity) are validated against published experimental data. Secondly, all calculations in the Matlab® VATS routines are validated by comparing the results with the same calculations written in PTC® Mathcad Prime 3.1.

4.3.1.1 Validation of Diffusion and Viscosity

The estimation for diffusivity is important as it is used in the calculation of both mass and heat transfer coefficients in Section 4.2. Correlations for binary gas diffusion and tertiary gas diffusion are validated against calculated results and experimental data found in Bird et al., 2002 and Poling et al., 2001. The estimation of viscosity is also of importance as it factors into many of the gas property correlations. Correlations for pure and mixed gas viscosity are validated against experimental data from the CRC Handbook of Chemistry and Physics (1993). The full tabular and graphical results are shown in detail in Appendix D.

To summarize Appendix D, results for binary diffusion using the Fuller method (Poling et al., 2001) are shown to be consistent with calculated results shown in Poling et al., 2001, validating the proper use of the correlation. For the gas mixtures studied in this work, the Fuller method results in differences from experimental data of about 3% for N_2 and CO_2, about 5% for N_2 and H_2O vapor, and less than 1% for air and CO_2. Air is treated as a pure gas for this binary calculation. For fluxes in multicomponent systems where a dilute component diffuses into a homogenous mixture, the simple relationship known as Blanc’s law (Poling et al., 2001) may be applied. As shown in Appendix D, the accuracy of the tertiary diffusion calculation for CO_2 (the trace component) diffusing into air (but now with N_2 and O_2 represented as discrete gases) has greater error magnitudes in some cases. Similar results for traces of water vapor and helium diffusing into N_2 and O_2 are observed. Due to these observations, the VATS routine allows air to be treated as a pure gas for diffusion calculations.

Compared with experimental data from CRC Handbook of Chemistry and Physics (1993), accuracies for pure gas viscosities calculated with the Lucas method (Poling et al., 2001) for gases of interest in this work (N_2, CO_2, and H_2O) over temperatures of 200 to 500 K are less than 1% for N_2, 2% for H_2O, and about 1% for CO_2. For the binary mixture of N_2 and CO_2, accuracy was about 1.5% when compared with experimental data from Reid et al. (1987). For a mixture of N_2, $CO_{2,}$ O_2 and Ar, the calculated viscosities over a temperature range of 220 to 300 K were compared with experimental data from the CRC Handbook of Chemistry and Physics, and were accurate within 0.5%, with accuracy increasing with temperature. These results indicate the Lucas method is satisfactory for the range of conditions studied in this work.

4.3.1.2 Validation of VATS Code

One of the primary purposes of the VATS routine is to prepare the input parameters for the COMSOL® simulation run based on the user input data. The inputs to the VATS routine are in the form of engineering data that can typically be obtained directly from test instrumentation. From these, the constants used in the mass and energy balances must be calculated. Additionally, the heat and mass transfer gas properties are calculated based on the correlations in Section 4.2. To guard against errors in code development, and after code modification, a parallel Mathcad® routine was developed.

This routine, shown in Appendix E for the CO_2 breakthrough simulation described in Section 4.5.2, compares the calculation results in VATS with the same equations in the Mathcad® file. The VATS results are imported into the Mathcad® files and are identified with the letters "ML" (for Matlab) appended onto the variable names.

The Mathcad routine successfully validates the VATS code. For calculations without non-integer exponentials, the difference between the Matlab® and Mathcad® calculations is less than 10^{-15} which Mathcad® reports as errors. For equations with non-integer exponentials, the largest difference between the calculations is still less than 0.1% and can most likely be attributed to differences in round-off errors incurred by Matlab®.

4.4 Sensitivity of Simulation to Heat Transfer Correlations

The correlations selected for heat transfer coefficients were presented in Section 4.2.7. As discussed in that section, these were selected based on verification against experimental data and/or recommendations in the literature. However, if all the correlations reviewed are taken into consideration, a wide range of values for h_s, h_i, and k_{eff} will be obtained even for identical conditions, raising questions about accuracy. Also, as noted earlier, initial fixed-bed inlet conditions (temperature, pressure, and concentration) are used to calculate the heat transfer coefficients, which are then held constant during the simulation in the interest of minimal execution times. To determine if simulation results would be significantly altered due to potential inaccuracies in the heat transfer correlations or due to the use of constant values, an analysis was conducted and is summarized in this section. The complete numerical analysis in Mathcad® is provided as Appendix F.

4.4.1 Sensitivity of Simulation Results to Heat Transfer Correlations

To examine the variations in heat transfer correlation results based on appropriate conditions, the temperature, pressure, and sorbate partial pressure ranges for the three experiments studied in this work were reviewed. Table 4.2 provides these ranges and percent differences (Mohamadinejad, 1995; Mohamadinejad, 1999).

Table 4.2 Variation in temperature, pressure, and sorbate partial pressure during the thermal characterization experiment, CO_2 on zeolite 5A breakthrough experiment, and H_2O vapor on zeolite 5A breakthrough experiment (Mohamadinejad, 1995; Mohamadinejad, 1999).

	Temperature			Pressure			Sorbate Pressure	
Experiment	Low K	High K	Diff %	Low psia	High psia	Diff %	Input torr	Diff. %
Thermal Characterization	297	381	28.1%	15.4	15.5	0.8%	n/a	n/a
CO_2 Breakthrough	298	310	4.1%	15.3	15.4	0.5%	6.14	0.78%
H_2O Breakthrough	296	315	6.4%	15.5	15.6	0.6%	6.04	0.76%

As shown in Table 4.2, the experiments studied in the work have low sorbate concentrations (less than 1% of the total pressure), and the pressure differential across the fixed-bed is less than 1% of the total pressure. However, the change in temperature is much higher; approximately 28% during the thermal characterization experiment, 4% during the CO_2 on 5A breakthrough experiment, and 6% during the H_2O on 5A breakthrough experiment. Since the temperature changes are 5 to 35 times greater than the largest change of total pressure, this sensitivity analysis focusses on the temperature changes during the experiment. In Appendix F, heat transfer coefficients are calculated for the range of temperatures appropriate to each experiment. For this analysis, total pressure and sorbate partial pressure are held constant at the inlet conditions for each experiment. In the following sections, the correlations examined for h_s, h_i, and k_{eff} are reviewed and the values obtained from those correlations are provided.

4.4.1.1 Variations in Correlation Values for Sorbent to Gas Heat Transfer (h_s)

Correlations examined for the heat transfer coefficient from the sorbent to the gas stream (h_s) include four mass transfer equations which are based on the assumption of similarity between heat and mass transfer transport in gases as shown in Equation (4.14) (Ruthven 1982). Poling (2001) contends that although elementary kinetic arguments concerning gas transport lead to reasonable values for viscosity and mass diffusion, this approach is quite inaccurate for thermal conductivity. Yet based on a favorable comparison over a wide range of test data (Wakao and Kaguei, 1982) the heat transfer analogy with the film mass diffusion equation of Wakao et al. (1979) is shown to be appropriate for higher Reynolds numbers and was thus applied here. The three other correlations under consideration based on the heat transfer analogy (Dwivedi and Upadhyay, 1977; Petrovic and Thodos, 1968; Carberry, 1960) are shown in Appendix F, as well as a fifth equation derived from thermal considerations for heat transfer from a single particle inside a particle array (Molerus and Wirth, 2012). The uncertainty in h_s is based on the span of values for all these correlations. The correlation values, and thus the span, changes for different temperatures, so the temperature associated with the highest span is used to evaluate the uncertainty. The lowest and highest correlation values and percent relative uncertainty for h_s are shown for the thermal characterization experiment and breakthrough experiments in Table 4.3. The relative uncertainties range from ±16% for the thermal characterization experiment to ±6% for the H_2O on 5A breakthrough experiment when the Molerus and Wirth (2012) correlation is included. However, it is of questionable accuracy and not in family with the other correlations. If the Molerus and Wirth (2012) correlation is not included, the relative uncertainty for h_s decreases by a factor of about three; however, for conservatism, the larger uncertainty estimate is used to evaluate the sensitivity of the simulation to changes in h_s.

Table 4.3 Variation in thermal coefficients calculated with multiple heat transfer correlations and estimated uncertainties for the thermal characterization experiment, CO_2 on zeolite 5A breakthrough experiment, and H_2O vapor on zeolite 5A breakthrough experiment.

	h_s, $W \cdot m^{-2} \cdot K^{-1}$			k_{eff}, $W \cdot m \cdot K^{-1}$			h_i, $W \cdot m^{-2} \cdot K^{-1}$		
Experiment	Low	High	$P_{\%}$	Low	High	$P_{\%}$	Low	High	$P_{\%}$
Thermal Characterization	68.2	93.6	15.6%	0.553	2.74	66.4%	14.3	15.7	4.7%
CO_2 Breakthrough	85.8	106	10.3%	0.453	2.27	66.8%	12.6	13.8	4.6%
H_2O Breakthrough	137	154	5.8%	3.15	0.472	74.0%	18.9	20.0	3.0%

4.4.1.2 Effective Axial Thermal Conductivity (k_{eff}) Correlation Variations

Ruthven (1984) suggests use of the similarity relationship shown in Equation (4.19) to calculate effective axial thermal conductivity (k_{eff}) based on the axial mass dispersion. Using this argument, the five axial dispersion correlations listed in Equations (4.10a) to (4.10e) were evaluated as shown in Appendix F.

$$k_{eff} = c_{pg} D_L \tag{4.19}$$

However, the accuracy of this approach is questionable particularly at low flow rates, as unlike fluid dispersion, thermal conductivity in a packed bed includes at least four other transport mechanisms in addition to the axial fluid mixing:

(1) Through the solid part of the pellets, including contact resistance between pellets;

(2) A mixed path through the pellet with both zeolite and fluid. The pellets are comprised of crystals and binder with micropores and macropores, both containing gas specific to the process near the process pressure and temperature;

(3) A pure fluid path through the gas outside the pellets (conduction, diffusion and fluid mixing contributions), and

(4) Radiative heat transfer between pellets.

An alternative approach per Wakao and Kaguei (1982) is shown in Equation 4.20. This equation is based on thermal test data, including fluid mixing as a function of flow rate but neglecting radiation (which is only important for high temperatures).

$$k_{eff} = k_e + 0.5 PrRe \tag{4.20}$$

However, the equation was not validated against experimental data. The Yagi and Kunii (1960) expression in Equation (4.17) for k_{eff} was selected for this work with the quiescent thermal conductivity of Krupiczka (Li and Finlayson, 1977; Wakao and Kaguei, 1982) as shown in Equation (4.16). This expression was chosen based on experimental data in Kaviany (1995). The percent variation in the axial thermal conductivity correlations for the thermal characterization experiment and breakthrough experiments is given in Table 4.3. As with the sorbent to gas heat transfer coefficient, in the interest of conservatism all correlations are used to calculate uncertainty even though the mass transfer analogy approach used in five correlations is questionable. The estimated uncertainty without the questionable correlations would again be reduced by a factor of about three.

4.4.1.3 Variations in Correlation Values for Gas Phase to Column Interior Wall Heat Transfer (h_i)

Only two correlations were found in the literature for the 1-D correlations for heat transfer between the inside of the column wall and fixed-bed (h_i), that of Rase (1990) and Li and Finlayson (1977). The corresponding 2-D correlation of Li and Finlayson was verified against experimental data (Wakao and Kaguei, 1982). Although experimental data were not available to correlate the 1-D correlation of Li and Finlayson (1977), it was selected based on the success of these researchers with the 2-D correlation. The estimated relative uncertainty in the correlation values is shown in Table 4.3.

Thermal characterization simulations were run with the ranges of values shown in Table 4.3. The results in Figure 4.2 show that the simulation results are relatively insensitive to the uncertainty in the heat transfer correlations.

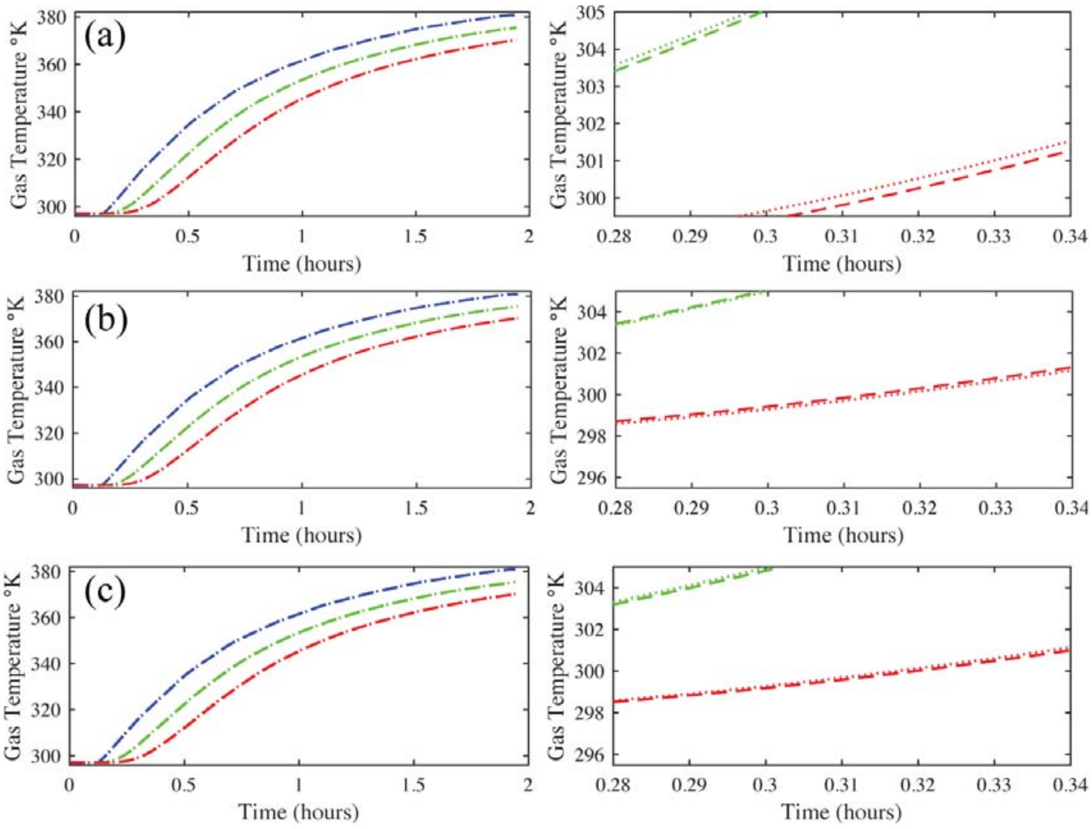


Figure 4.2a Thermal characterization simulation results with varying sorbent to gas heat transfer (h_s) coefficient of 68.2 (dotted line) and 93.6 (dashed line) W•m^{-2}•K^{-1}. Left side plots show the full simulation, while right plots are zoomed in to observe differences in the simulation results.

Figure 4.2b Thermal characterization simulation results with varying effective axial transfer conductance (k_{eff}) of 0.453 (dotted line) and 2.27 (dashed line) W•m^{-2}•K^{-1}.

Figure 4.2c Thermal characterization simulation results with varying gas to internal column wall heat transfer coefficient (h_i) of 14.3 (dotted line) and 15.7 (dashed line) W• m^{-2}•K^{-1}.

In Section 4.5, the value for the heat transfer coefficient from the outside the column to the environment (h_o) was obtained by varying the value of h_o in the simulation until the difference between the experimental temperature history and the simulated temperature history, as calculated by the sum of the square errors (SSE), was found. It is important to verify that the value obtained for h_o via fitting is not significantly altered by the slightly altered simulation behavior shown in Figure 4.2. This was accomplished by repeating the fitting process with the worst-case values for h_s, k_{eff}, and h_i (e.g., the combination of these three values that alter the simulation results the most based on Figure 4.2). The SSE values obtained from a series of simulations with varying values of h_o are shown in Figure 4.3. The minimum error with the selected thermal correlations occurs for $h_o = 1.69$ W•m^{-2}•K^{-1}. For the worst-case values, the minimum error occurs for $h_o = 1.66$ W•m^{-2}•K^{-1}. Thus the fitted value for h_o shifted by only 2% as the result of uncertainty in the thermal correlations.

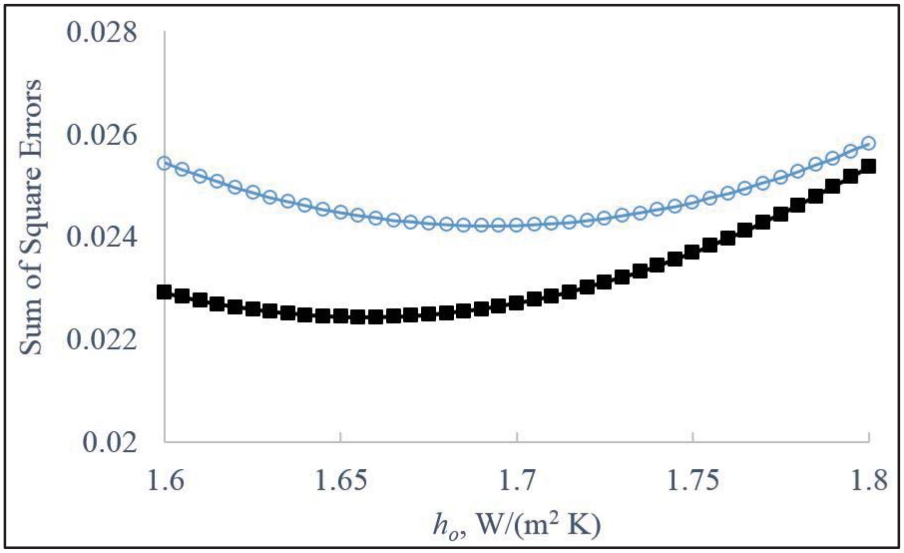


Figure 4.3 Sum of square error vs. h_o for the thermal characterization simulation. Blue circles: SSE for values from thermal correlations selected for use in the work: $h_s = 92.2$ W•m^{-2}•K^{-1}; $k_{eff} = 0.793$ W•m•K^{-1}; and $h_i = 13.7$ W•m^{-2}•K^{-1}. Black squares: SSE for values from worst-case thermal correlations: $h_s = 68.2$ W•m^{-2}•K^{-1}; $k_{eff} = 2.74$ W•m•K^{-1}; and $h_i = 14.3$ W•m^{-2}•K^{-1}.

It could be argued that such a small change in the fitted value for h_o indicates that the uncertainty in the thermal correlations is insignificant. However, the primary goal of this work is an accurate determination of the LDF mass transfer coefficient (k_n). Thus the impact of using worst-case values for thermal coefficients in determination of k_n is considered next.

Similarly, to the determination of h_o, the value for k_n is obtained by varying the value of k_n in the simulation until the minimum difference between the experimental concentration and the simulated concentration history, as calculated by the SSE, is found. The detailed approach is provided in Section 4.5.3. To assess the sensitivity of this procedure to the uncertainty in the thermal correlations, the fitting process is executed with the correlations selected for this work and with the correlations that provide the worst-case values previously determined.

The comparative results from the two fitting processes are shown in Figure 4.4. Although it is clear from the figure that some change in value of the SSE occurs, the figure also indicates the minimum SSE value is unchanged at $k_n = 0.24$ s^{-1}. These results indicate that the fitting of k_n is insensitive to variations in the thermal coefficients, even when the variations are based on a conservative estimate of uncertainty in the thermal correlations used in this study.

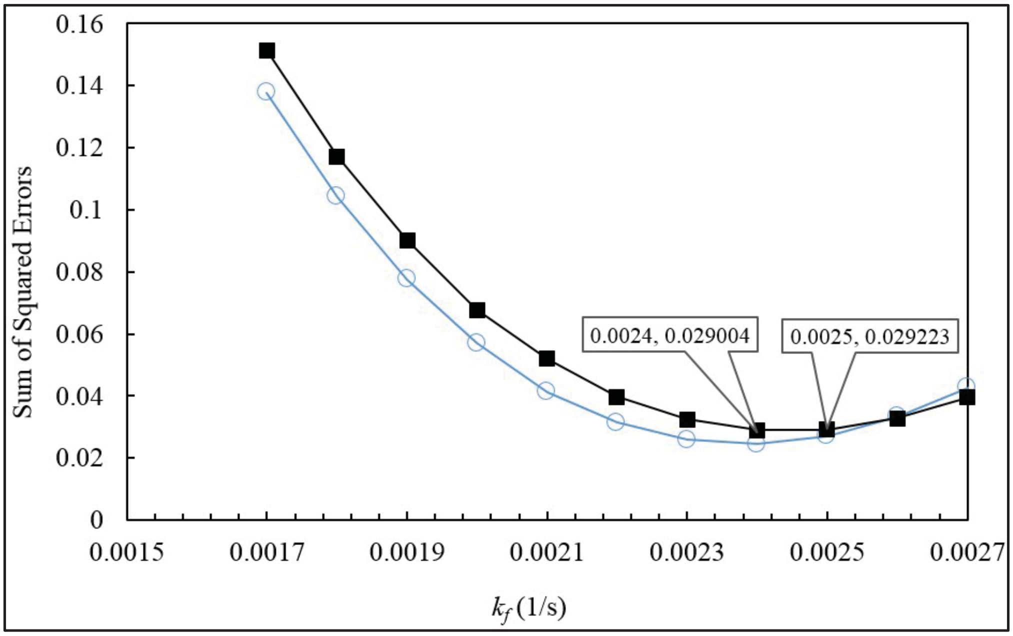


Figure 4.4 Sum of square errors vs. k_n for the CO_2 breakthrough simulation. Blue circles: SSE for values from thermal correlations selected for use in the work: h_s = 104 W $\cdot$m$^{-2}\cdot$K^{1}; k_{eff} = 0.653 W$\cdot$m$\cdot$K^{-1}; h_i = 12.5 W$\cdot$m$^{-2}\cdot$K^{-1}; and h_o = 1.69 W$\cdot$m$^{-2}\cdot$K^{-1}. Black squares: SSE for values from worst-case thermal correlations: h_s = 85.8 W$\cdot$m$^{-2}\cdot$ K^{-1}; k_{eff}= 2.27 W$\cdot$m$\cdot$K^{-1}; h_i = 12.6 W$\cdot$m$^{-2}\cdot$K^{-1}; and h_o = 1.66 W$\cdot$m$^{-2}\cdot$K^{-1}. Callouts show that the minimum SSE occurs for a value of k_n = 0.24 s^{-1}.

4.4.2 Sensitivity of Simulation Results to Use of Constant Heat Transfer Coefficients

The simulations described in this work use constant thermal and gas properties in order to minimize run time. Since, (as shown in Table 4.2) the temperature changes are 5 to 35 times greater than the largest change of total pressure, this sensitivity analysis also focusses on the temperature changes during the experiment. In Table 4.4, the percent differences in the values obtained via the correlations selected for use in this work due to changes in temperature during the experiment are shown. By comparing Table 4.4 with Table 4.2, we see that the percent differences resulting from temperature changes is always lower than the percent difference due to uncertainty in the thermal correlations.

Since the previous section established that the determination of the mass transfer coefficient is insensitive to changes resulting from uncertainty in the thermal correlations, it follows that determination of k_n will be not be compromised by the use of constant heat transfer coefficients.

Table 4.4 Variance in selected correlation results for thermal coefficients due to temperature changes in thermal characterization experiment, CO_2 on zeolite 5A breakthrough experiment, and H_2O vapor on zeolite 5A breakthrough experiment.

		Percent difference due to temperature change		
Experiment	Change in temperature Δ °K	h_s per Wakao and Funazkri (1978), $W{\cdot}m^2{\cdot}K$	k_{eff} per Yagi et al. (1960) $W{\cdot}m{\cdot}K$	h_i per Li and Finlayson (1977) $W{\cdot}m^2{\cdot}K$
Thermal Characterization	83.6	9.30	0.776	3.85
CO_2 Breakthrough	11.2	2.24	0.175	0.806
H_2O Breakthrough	18.7	2.28	0.183	0.772

4.5 Results and Discussion

Two of the three free (unknown) parameters, which included one of the heat transfer coefficients (h_o) and the axial dispersion coefficient (D_L), were determined either *a priori* using the correlations described in Section 4.2 or by fitting the model to the experimental data. The LDF mass transfer coefficient (k_n) was determined by fitting the model to the experimental data. In all cases, when determining a free parameter by fitting the model to experimental data, the SSE was minimized as shown in Figures 4.3 and 4.4. When comparing simulated breakthrough curves to experimental data, the SSE was minimized between 25% and 75% of the inlet concentration to focus the fit on the mid-height slope of the breakthrough curve.

The heat transfer properties of the experimental apparatus were determined first. Then, the mass transfer properties of each adsorbent/adsorbate system were determined in terms of finding k_n for each adsorbent/adsorbate system by fitting the model to experimental data with D_L determined *a priori* from the correlations in Section 4.4. Finally, it was necessary to reevaluate the D_L for each system by fitting the model to experimental data while using the value of k_n just found for each adsorbent/adsorbate system. The features of each adsorbent/adsorbate system are discussed in detail throughout this systematic analysis that was developed to determine their heat and mass transfer properties.

4.5.1 Thermal Characterization Tests and Fitting of Heat Transfer Parameter

To determine and verify the heat transfer parameters, the model was compared to thermal characterization experiments that were performed by introducing heated nitrogen to the inlet of the column. The adsorbent was regenerated prior to the test, as described in Chapter 3. The test conditions are shown in Table 4.5.

Table 4.5 Test conditions for thermal characterization, breakthrough tests with CO_2 on zeolite 5A, and breakthrough tests with H_2O vapor on zeolite 5A.

Parameter	Thermal Characterization	CO_2/5A	H_2O/5A
Flow rate, liters min^{-1} at STP	28.0	28.3	28.3
Initial temperature, K	297	299	297
Initial inlet temperature, K	297	298	297
Inlet pressure, kPa	107	106	107
Inlet partial pressure, kPa	n/a	0.819	0.805

Figure 4.5 shows the thermal characterization test data with experimental uncertainty along with a comparison of the data and the corresponding thermal characterization simulation results. Measurement uncertainty for a 95% confidence interval was determined in Chapter 3 to be ±0.4 K for temperature readings. The inlet temperature provided the boundary condition for the simulation. The only adjustable parameter was the heat transfer coefficient (h_o) from the column wall to the surroundings; all the other parameters were obtained from the heat transfer correlations given in Section 4.2. A value of h_o = 1.69 Wm^{-1}K^{-1} provided the best fit to the thermal characterization test data, with the resulting simulated temperatures closely matching the experimental temperatures measured at the inlet (2.5%), middle (50%), and exit (97.5%) of the bed.

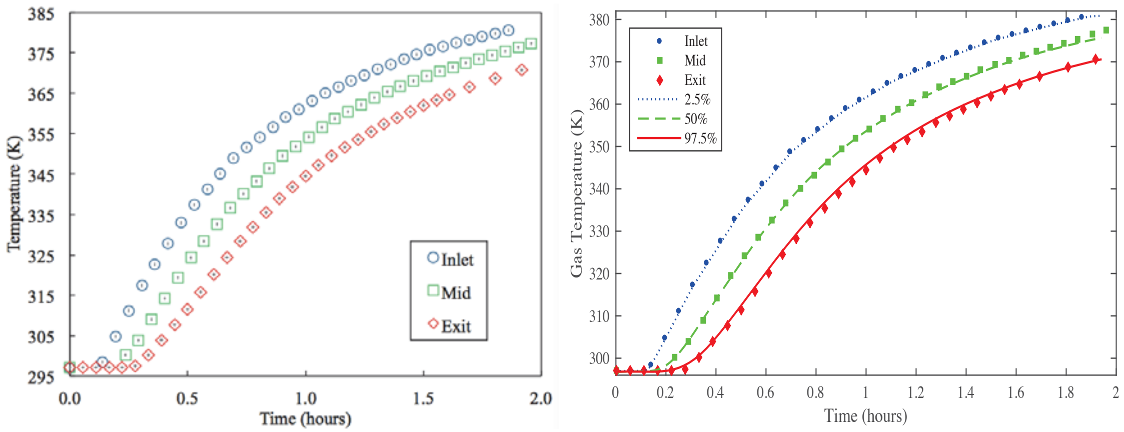


Figure 4.5 Temperature history data for the thermal characterization test with N_2 on zeolite 5A at three centerline locations in the bed (circles: 2.5%, squares: 50%, and diamonds: 97.5%). Left Panel: Experimental data with error bars showing experimental uncertainty. Right Panel: Experimental data with corresponding predictions from the model with the heat transfer coefficient from the column wall to the surroundings h_o = 1.69 $Wm^{-1}K^{-1}$.

4.5.2 Experimental Breakthrough Tests for CO_2 and H_2O Vapor on Zeolite 5A

The experimental breakthrough test conditions for CO_2 and H_2O vapor on zeolite 5A are provided in Table 4.2. In preparation for these tests, the adsorbent was purged with helium gas heated to 590 K to ensure starting with a fully regenerated bed. Nitrogen was used as the carrier gas for these breakthrough tests. The breakthrough test results for both CO_2 and H_2O vapor are shown in Figure 4.6 in terms of the resulting experimental gas-phase concentration and temperature profile histories. The centerline gas-phase concentration profile histories were measured just inside the bed (2.5% into the bed), in the middle of the bed (50% into the bed), and just inside the exit of the bed (97.5% into the bed). The typical gas-phase concentration breakthrough curve was also measured just outside the bed. Measurement uncertainty for a 95% confidence interval was determined to be ±0.4 K for temperature readings, ±1.3% of reading for water vapor concentrations, and ±1.2% of reading for CO_2 concentrations. Uncertainty in time was determined to be ±1.3% of reported time in Chapter 3.

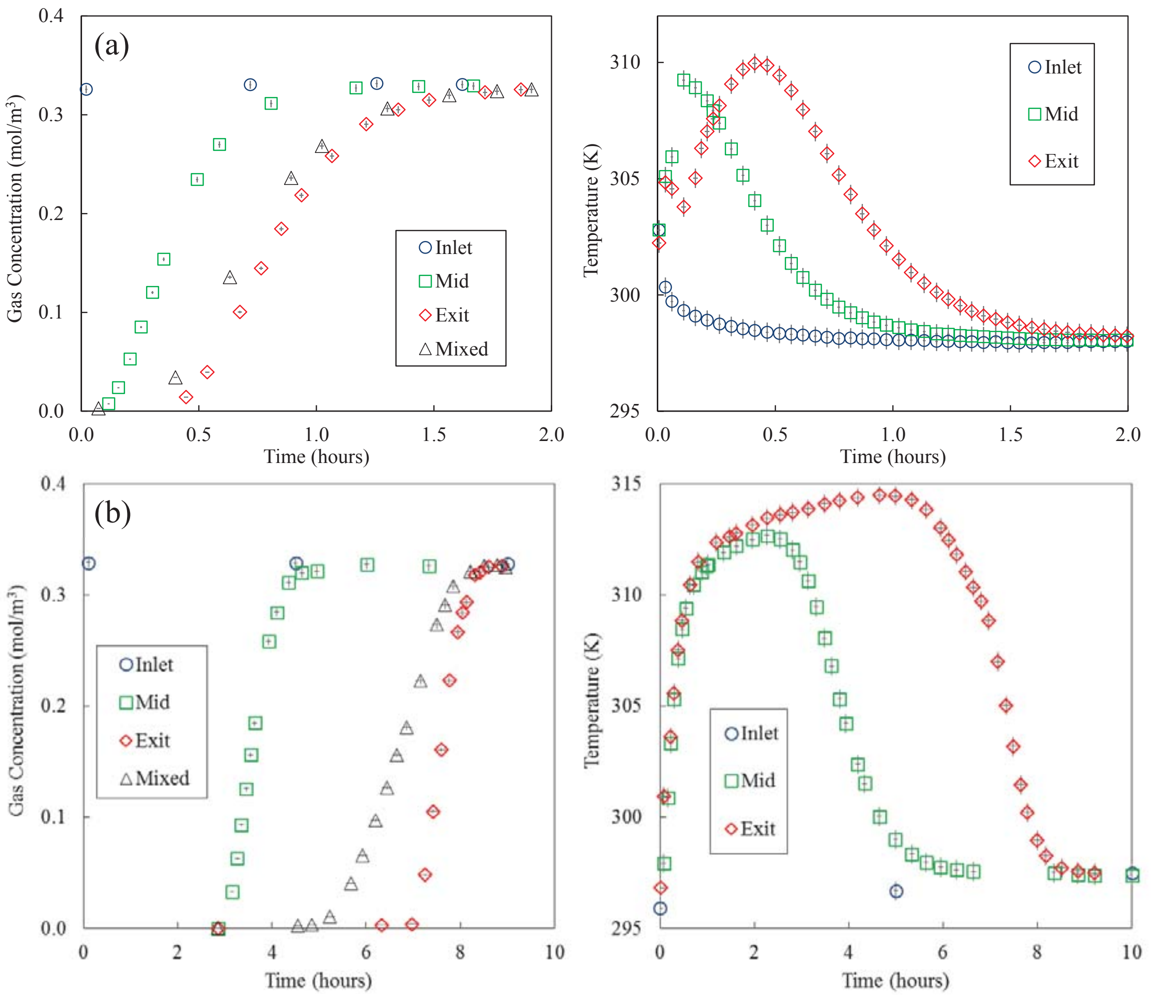


Figure 4.6a Experimental gas-phase concentration profile history breakthrough curves for CO_2 on zeolite 5A at three centerline locations in the bed (circles: 2.5%, squares: 50%, and diamonds: 97.5%), and just outside the bed (triangles).

Figure 4.6b Corresponding experimental temperature profile histories for H_2O vapor on zeolite 5A at three centerline locations in the bed. Error bars show experimental uncertainty.

The early peaks in the experimental temperature profile histories observed for CO_2 (Figure 4.6a) at the 97.5% location were due to the initial adsorption of N_2. Recall the bed was filled with He at the start of each run. This feature was not observed with the H_2O vapor-zeolite 5A system (Figure 4.6b) simply due to the much longer time scale of that run.

The discrepancies between the gas-phase concentration profile histories for both adsorbates at the 97.5% location, which are not generally available in breakthrough studies in the literature, and those just outside the bed provided insight to the nature of the actual, non-plug flow conditions existing in the bed. The earlier breakthroughs observed with the outside the bed profiles indicated that channeling was probably occurring along the inner wall of the column. This non-plug flow behavior was most readily observed for H_2O vapor.

Channeling is generally known to occur due to a higher near-wall gas flow rate that is associated with a lower packing density (i.e., higher void fraction) close to the wall. This was initially unexpected, especially when considering that the ratio of the bed to pellet diameter for this packed bed was around 20. A value of 20 is generally considered large enough to obviate wall effects due to the near-wall lower packing density (Richardson et al., 2002).

The analyses in Figure 4.7 and Table 4.6 show that the origin of this non-plug flow condition was independent of the adsorbates involved. This was expected, but only if the dispersion for each system was the same and derived from a mechanical phenomenon like that associated with near-wall channeling. To prove this supposition, the same experimental gas-phase concentration profile histories are shown in Figure 4.7

for the 50%, 97.5%, and just outside the bed locations for both CO_2 and H_2O vapor, but now plotted against a dimensionless time (t/t_{BT}) defined relative to the respective breakthrough time for each adsorbate for the breakthrough curve measured just outside the bed, i.e., t_{BT}. Table 4.6 shows the dimensionless breakthrough times for both species at the 50% and 97.5% locations relative to t_{BT}, where $\bar{t}$ was evaluated based on the formulation for dilute systems, i.e.,

$$\bar{t} = \int_0^\infty \left(1 - \frac{C}{C_o}\right) dt \tag{4.21}$$

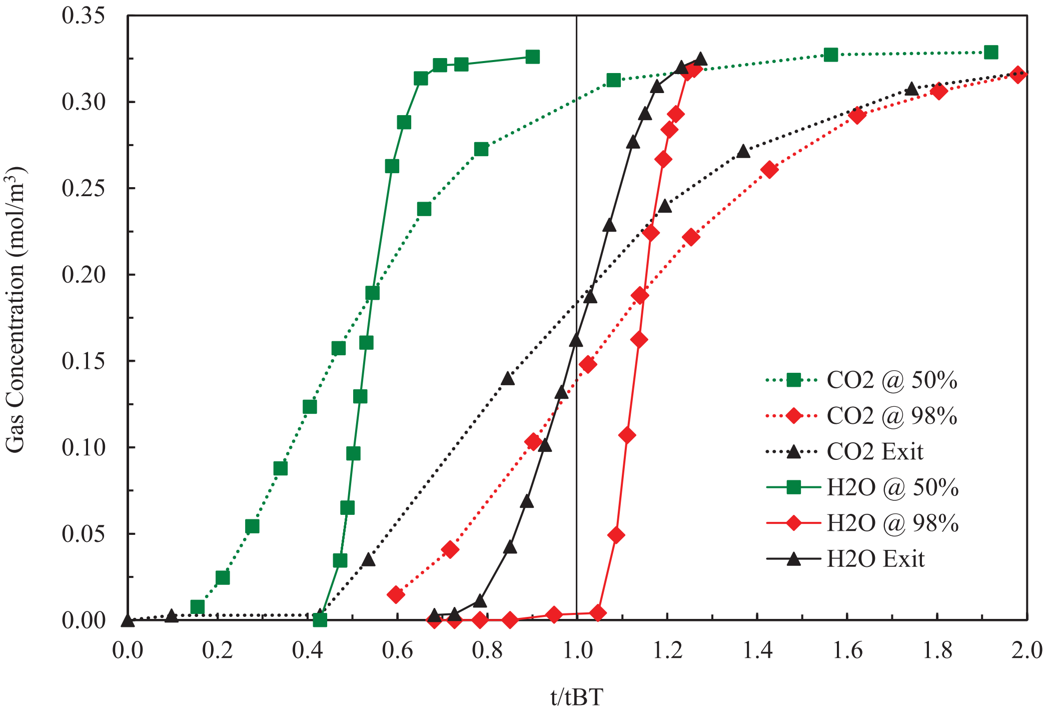


Figure 4.7 Experimental gas-phase concentration profile history breakthrough curves for CO_2 (dotted lines) and H_2O vapor (solid lines) on zeolite 5A at two centerline locations in the bed (squares: 50%, and diamonds: 97.5%), and just outside the bed (triangles) plotted against dimensionless time defined relative to the respective breakthrough time for each adsorbate for the breakthrough curve measured just outside the bed, i.e., t_{BT}.

Table 4.6 Center of mass gas-phase concentration profile history breakthrough curve time ratios for CO_2 and H_2O vapor on zeolite 5A at 2 centerline locations in the bed (from Figure 4.7).

CO_2 on Zeolite 5A		H_2O Vapor on Zeolite 5A		% Difference
97.5% to Outside	1.144	97.5% to Outside	1.140	0.34%
50% to Outside	0.5422	50% to Outside	0.5396	0.48%

The relative temporal locations of the breakthrough curve times at the 50% and 97.5% locations were nearly identical for both CO_2 and H_2O vapor, as expected for a mechanical dispersion phenomenon that should be independent of the adsorbate. It is also worth pointing out the self-consistency of the experimental outside the bed breakthrough curves for the CO_2 and H_2O vapor systems. When plotted as shown in Figure 4.7, the two curves should cross at $t/t_{BT} = 1$. A vertical line was drawn at $t/t_{BT} = 1$ in Figure 4.7 to emphasize this point. Clearly, only a slight vertical difference existed between the two curves at the crossing point.

4.5.3 Empirical Determination of the LDF Mass Transfer Coefficient k_n

The determination of a mass transfer parameter, like k_n, is commonly accomplished by fitting the 1-D axial dispersed plug flow model to an experimental breakthrough curve measured at a location outside the bed, just like those shown in Figure 4.6. From the analysis provided so far it should be clear that even for a proper bed to pellet diameter ratio of 20, a breakthrough curve obtained just outside the bed may not be providing fundamental mass transfer information, because it may be strongly subjected to non-plug flow effects that are most likely due to near-wall channeling. It is shown below that this dilemma can be resolved by using the experimental centerline gas-phase concentration profile histories to determine k_n, as the non-plug flow, near-wall channeling effects should not exist along the column centerline.

First, the dispersion coefficients were predicted for each adsorbate/adsorbent system from the five correlations given in Equation (4.10). The results are summarized in Table 4.7. The dispersion coefficients predicted from the Edwards and Richardson correlation (Equation (4.10b)) were within 2% of the smallest values obtained from the Wen and Fan correlation (Equation (4.10e)), and those from the Wakao and Funazkri correlation were (Equation (4.10a)) the largest values. The dispersion coefficients predicted from the other two correlations fell in between. Between the largest and smallest values, there was a factor of two for CO_2 on zeolite 5A and a factor of nearly three for H_2O vapor on zeolite 5A. Based on these findings, both the Edwards and Richardson (Equation (4.10b)) and Wakao and Funazkri (Equation (4.10a)) correlations (which encompass the extremes) were used in the determination of k_n to see if there was any effect of the magnitude of the predicted dispersion coefficient.

Table 4.7 Axial dispersion coefficients predicted from the five correlations given in Equation (4.10), and the resulting LDF k_n values obtained from fitting the 1-D axial dispersed plug flow model to the 97.5% location experimental centerline gas-phase concentration breakthrough curves for CO_2 and H_2O vapor on zeolite 5A using only the top two dispersion coefficient correlations listed.

CO_2 on Zeolite 5A			H_2O on Zeolite 5A		
k_n, s^{-1}	D_L correlation	D_L, m^2 s^{-1}	k_n, s^{-1}	D_L correlation	D_L, m^2 s^{-1}
$2.2x10^{-3}$	Edwards and Richardson	$8.99x10^{-4}$	$8.8x10^{-4}$ s^{-1}	Edwards and Richardson	$8.62x10^{-4}$
$2.3x10^{-3}$	Wakao and Funazkri	$1.89x10^{-3}$	$9.8x10^{-4}$ s^{-1}	Wakao and Funazkri	$2.40x10^{-3}$
-	Wicke	$9.91x10^{-4}$	-	Wicke	$9.91x10^{-4}$
-	Ruthven	$9.72x10^{-4}$	-	Ruthven	$9.63x10^{-4}$
-	Wen and Fan	$8.93x10^{-4}$	-	Wen and Fan	$8.47x10^{-4}$

Figure 4.8 shows fits of the model to the 97.5% location experimental gas-phase concentration breakthrough curves for both adsorbate/adsorbent systems using axial dispersion coefficients predicted from the Edwards and Richardson (Equation (4.10a)) and Wakao and Funazkri (Equation (4.10b)) correlations. The corresponding LDF k_n values, the only adjustable parameter, are listed in Table 4.7. In all cases, the saturation terms of the isotherms for both CO_2 and H_2O were adjusted to make the model agree with the location of the experimental results along the x-axis. These capacity adjustments were inconsequential to the resulting k_n values and were done to show how well the model fitted the data. Figure 4.8 also shows predictions from the model at the 2.5% and 50% experimental locations for both systems using the resulting k_n values.

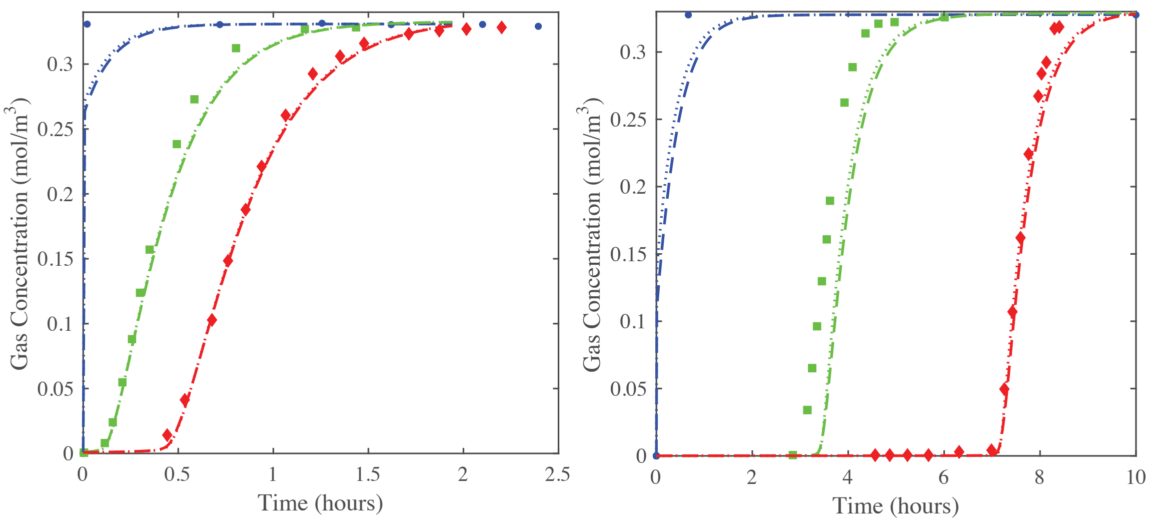


Figure 4.8 Fits of the 1-D axial dispersed plug flow model to the 97.5% location (diamonds) experimental centerline gas-phase concentration breakthrough curves for CO_2 (left) and H_2O vapor (right) on zeolite 5A, and corresponding predictions from the model of the 2.5% (circles) and 50% (squares) locations. Diamonds: experimental data; dashed lines: simulations with the Edwards and Richardson correlation for axial dispersion (Equation (4.10a)) and corresponding k_n values (Table 4.7); dotted lines: simulations with the Wakao and Funazkri correlation for axial dispersion (Equation (4.10b)) and corresponding k_n values (Table 4.7d). The saturation term in the CO_2-zeolite 5A isotherm was increased by 15%. The saturation term in the H_2O vapor-zeolite 5A isotherm was decreased by 3%. The void fraction was reduced to 0.33 based on the Cheng distribution (Cheng et al., 1991) with $C = 1.4$ and $N = 5$, as recommended by Nield and Bejan (1992).

The fitted and predicted modeling results in Figure 4.8 show good agreement with the experimental data. The modeling results in Figure 4.8 also show essentially no effect of using extreme values of the predicted dispersion coefficients on the resulting values of k_n. As the results in Table 4.7 show, similar values of k_n were obtained for CO_2 (0.0022 vs. 0.0023 s^{-1}) and H_2O vapor (0.00088 vs. 0.00098 s^{-1}) regardless of the dispersion coefficient correlation. These results further show that particle-scale dispersion stemming from bed packing (i.e., turbulence and flow splitting), which are the only types of dispersions accounted for with these correlations, had a negligible influence on the breakthrough results, corroborating what has been known for some time (Yang, 1987; Richardson et al., 2002). Consequently, these results show that it was indeed possible to extract fundamental adsorbate/adsorbent mass transfer information from these well-designed breakthrough experiments using the 1-D axial dispersed plug flow model with D_L predicted from a common correlation. This was the case because the experimental center line gas-phase concentration breakthrough curves, as alluded to earlier, experienced conditions very far removed from any near-wall channel effects, thereby allowing them to be described well by such a 1-D model. It was surmised that the consistent displacement between model and experiment at the 50% location for both CO_2 and H_2O vapor perhaps indicated a misplacement of the gas sampling lines.

Figure 4.9 compares the experimental gas-phase concentration breakthrough curves at the three inside centerline bed locations with those from the model for both CO_2 and H_2O vapor but now without any adjustments to the saturation terms of the isotherms and using the largest dispersion coefficients predicted from the Wakao and Funazkri correlation (Equation (4.10a)). The agreement was still quite good, especially in terms of shape, but not so much in terms of capacity, as expected without any adjustments. Notice that the shape and location of the experimental breakthrough curve obtained just outside the bed for CO_2 was only slightly more dispersed than the one at the 97.5% location, so the model also coincidently predicted it well; this was not the case for H_2O vapor. These interesting observations are addressed in more detail below after the temperature profile histories are discussed.

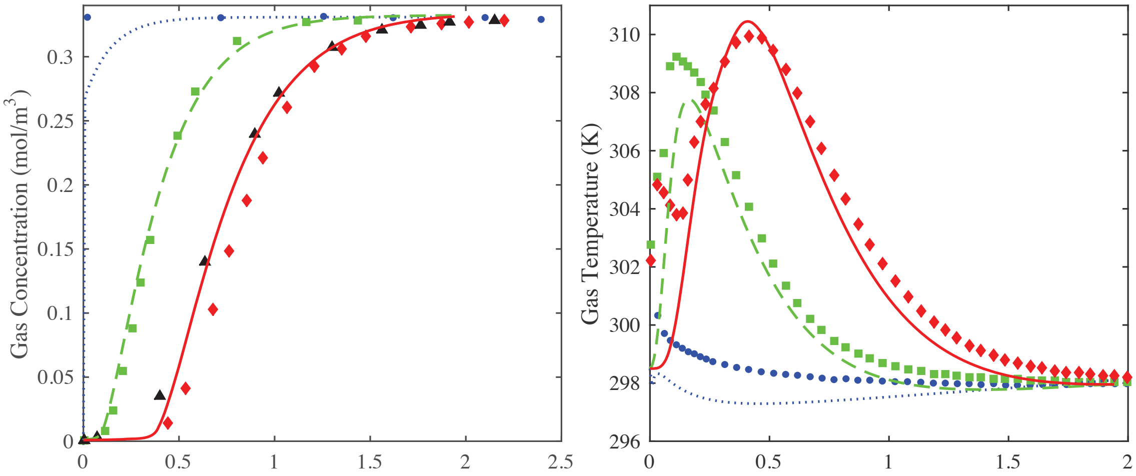


Figure 4.9a CO_2 on zeolite 5A: Predictions from the model (lines) shown in Figure 4.8 of the 2.5% (circles), 50% (squares), and 97.5% location (diamonds) experimental center line gas-phase concentration breakthrough curves (left), but now using the reported saturation term for the CO_2-zeolite 5A isotherm (no adjustment), a void fraction of 0.33, the Wakao and Funazkri correlation (Equation (4.10a)) for axial dispersion and LDF k_n = 0.0023 s^{-1}. The experimental outside the bed (triangles) breakthrough curve is shown for comparison. Predictions from the model (lines) of the 2.5% location (circles), 50% location (squares), and 97.5% location (diamonds) experimental center line temperature profile histories (right).

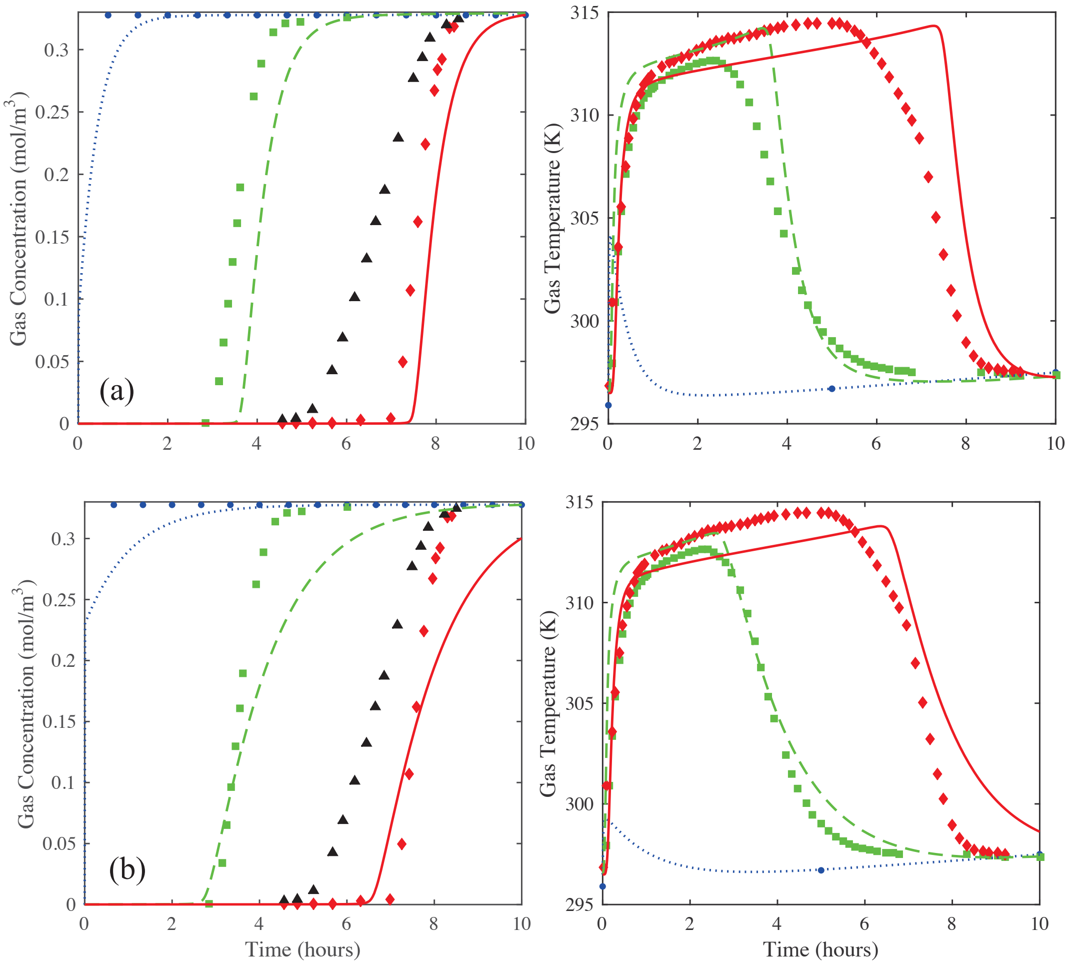


Figure 4.9b H_2O on zeolite 5A: (a) Predictions from the model (lines) shown in Figure 4.5 of the 2.5% location (circles), 50% location (squares), and 97.5% location (diamonds) experimental center line gas-phase concentration breakthrough curves, but now using the reported saturation term for the H_2O-zeolite 5A isotherm (no adjustment), a void fraction of 0.33, the Wakao and Funazkri correlation (Equation 4.10b) for axial dispersion and LDF k_n = 0.0008 s^{-1}. The experimental outside the bed (triangles) breakthrough curve is shown for comparison. Predictions from the model (lines) of the 2.5% location (circles), 50% location (squares), and 97.5% location (diamonds) experimental center line temperature profile histories. (b) H_2O on zeolite 5A (bottom panels): same as (a), but now with LDF k_n adjusted to k_n = 0.0002 s^{-1} to match the slope of the experimental outside the bed (triangles) breakthrough curve.

Figures 4.9a and 4.9b also compare the experimental centerline temperature profile histories at the three locations in the bed with those predicted from the model. In terms of shape, the model and experiment agreed quite well, especially for CO_2, and despite the fact that for H_2O vapor the model did not match the location of the experimental breakthrough curve just outside the bed as it did for CO_2. Notice how below 307 K in the cooling branch of the temperature profile histories for H_2O vapor, both the model and experiment tracked parallel to each other, as they should in this mass transfer dominated region of the temperature profile histories. This result indicated that the correct mass transfer information was extracted from the model by fitting it to the experimental center line gas-phase concentration breakthrough curves. To exemplify this point, the LDF k_n was purposely adjusted to match the slope of the concentration breakthrough curve just outside the bed (this is what is typically accomplished in the literature to obtain k_n), which required decreasing it by a factor of four compared to the supposedly correct value. This result is shown in Figure 4.9b. Notice how the model and experiment now deviated significantly from each other in the mass transfer limited region of the temperature profile histories. The point made here is that the experimental centerline temperatures in the bed and the experimental concentration breakthrough curve measured just outside the bed did not reflect the same phenomena, the former being dominated by adsorbate/adsorbent mass transfer and the latter being dominated by mechanical dispersion. As for the differences observed between the model and experiment above 307 K in the temperature profile histories for H_2O vapor, it was surmised that this was most likely due to the same non-plug flow, near-wall channeling phenomena that most certainly could not be predicted by the 1-D axial dispersed plug-flow model.

As an aside, it is noted that the differences between model and experiment measured just outside the bed (black triangles), which represent the capacity of the bed, differ for the two systems under consideration. Two factors that in combination may be responsible for the observed differences are (1) lower CO_2 and H_2O capacity for the lot of zeolite 5A used in the experiment that was used by Wang and LeVan (2009) and (2) incomplete desorption of water off the 5A prior to measurement of isotherms by Wang and LeVan (regeneration was performed overnight at 175°C at vacuum). For the CO_2/5A system, the two factors appear to cancel, resulting in good agreement between model and experiment. In this case even a minute amount of water remaining on the 5A after regeneration can significantly reduce CO_2 adsorption. For the H_2O/5A system, the lower H_2O capacity for the lot of zeolite 5A used in the experiment is not cancelled by the incomplete regeneration, since if only a minute amount of water remained, the impact on the measured capacity would be small.

4.5.4 Non-Plug Flow Axial Dispersion Coefficient Determination on Zeolite 5A

As shown above, the D_L values predicted from two correlations representing the extreme high and low values did not have a significant influence on the simulation results and thus the resulting k_n values. It was also shown above that the breakthrough curves obtained just outside the bed were subjected to a non-plug flow, mechanical dispersion mechanism. This mechanical dispersion mechanism was most likely due to near-wall channeling associated with higher velocities that naturally occur due to higher porosities near the wall.

Figure 4.10 compares the model to the experiment for CO_2 on zeolite 5A using the fitted parameter k_n = 0.0023 s^{-1} and a value of the dispersion coefficient that was 7 times larger than that predicted from the Wakao and Funazkri correlation (Equation (4.10a)). It shows the modeling and experimental gas-phase concentration breakthrough curves at several locations in the bed and just outside the bed, the corresponding derivatives or slopes of the concentration breakthrough curves from the model, and the modeling and experimental centerline temperature profile histories within the bed. Notice how the model now captured the contour of the experimental breakthrough curve just outside the bed. To do this, a value of D_L that was 7 times larger than the largest value predicted from any of the correlations was needed. This substantiated the fact that the dominant dispersion mechanism in the experimental data was not the same as any of those accounted for in any of the correlations. It was also interesting that the slopes of the concentration breakthrough curves show only slight concentration front sharpening at the end of the bed and they indicated that CPB was just approached near the end of the bed. This was not the case for the H_2O vapor system, as shown later.

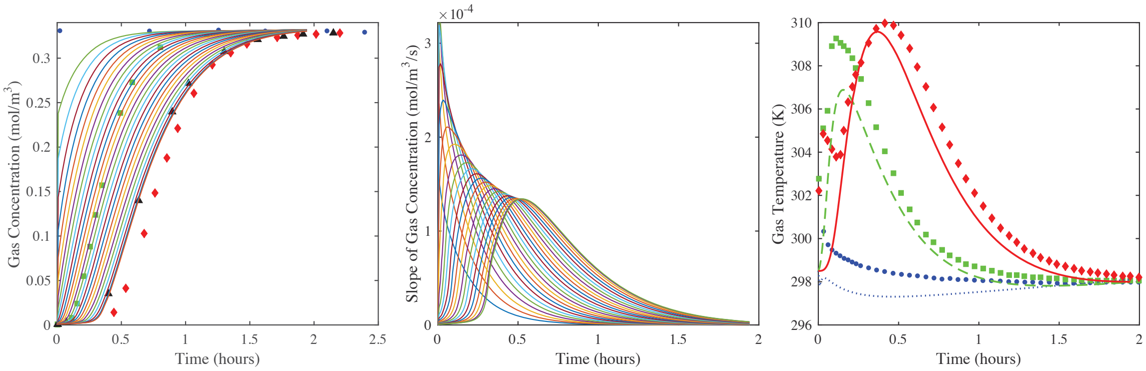


Figure 4.10 CO_2 on zeolite 5A: Fit of the 1-D axial dispersed plug flow model to the outside bed (triangles) experimental breakthrough curve using a value of D_L 7 times greater than that from the Wakao and Funazkri correlation and the fitted LDF k_n = 0.0023 s^{-1} (left panel). The reported saturation term for the CO_2-zeolite 5A isotherm was used, along with the reported void fraction of 0.35. Predictions from the model (lines) of the gas-phase concentration breakthrough curves at 0, 4, 8, 12, …, 92, 96, and 100% locations in the bed are also shown in the left panel, along with the 2.5% (circles), 50% (squares), and 97.5% location (diamonds) experimental center line gas-phase concentration breakthrough curves (left panel). The corresponding derivative (or slope) of the predicted gas-phase concentration breakthrough curves in the bed are shown in the middle panel. Predictions from the model (lines) of the 2.5% (circles), 50% (squares), and 97.5% location (diamonds) experimental center line temperature profile histories are shown in the right panel.

Figure 4.11 compares the model to the experiment for H_2O vapor on zeolite 5A using the fitted k_n = 0.00098 s^{-1} and values of the dispersion coefficients that were 7, 30, 50, and 70 times larger than that predicted from the Wakao and Funazkri correlation (Equation (4.10a)). These results show that it took a dispersion coefficient value ~50 times larger than the value predicted by the Wakao and Funazkri correlation to reasonably fit the slope and shape of the experimental concentration breakthrough curve just outside the bed. However, as the axial dispersion coefficient increased, the shape of the temperature profile histories increasingly deviated from the experimental results. These results again clearly show that the experimental temperature profile histories and the experimental concentration breakthrough curve obtained just outside the bed did not reflect the same dominating mechanism. As mentioned above, the experimental temperatures reflected the mass transfer process taking place, while the experimental concentration breakthrough curve measured outside the bed reflected mechanical dispersion caused by non-plug flow conditions due to near-wall channeling effects.

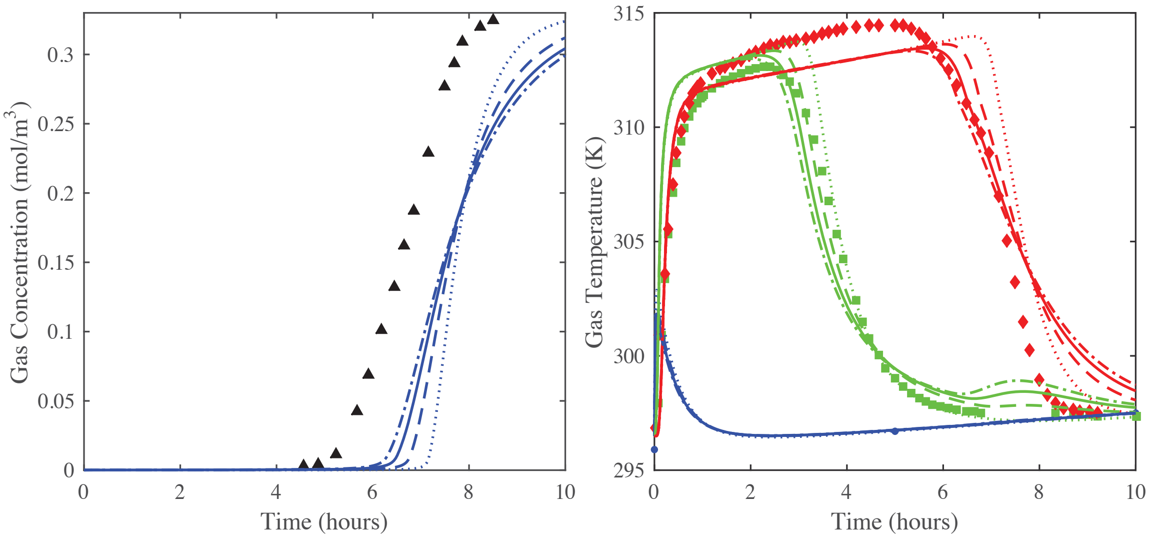
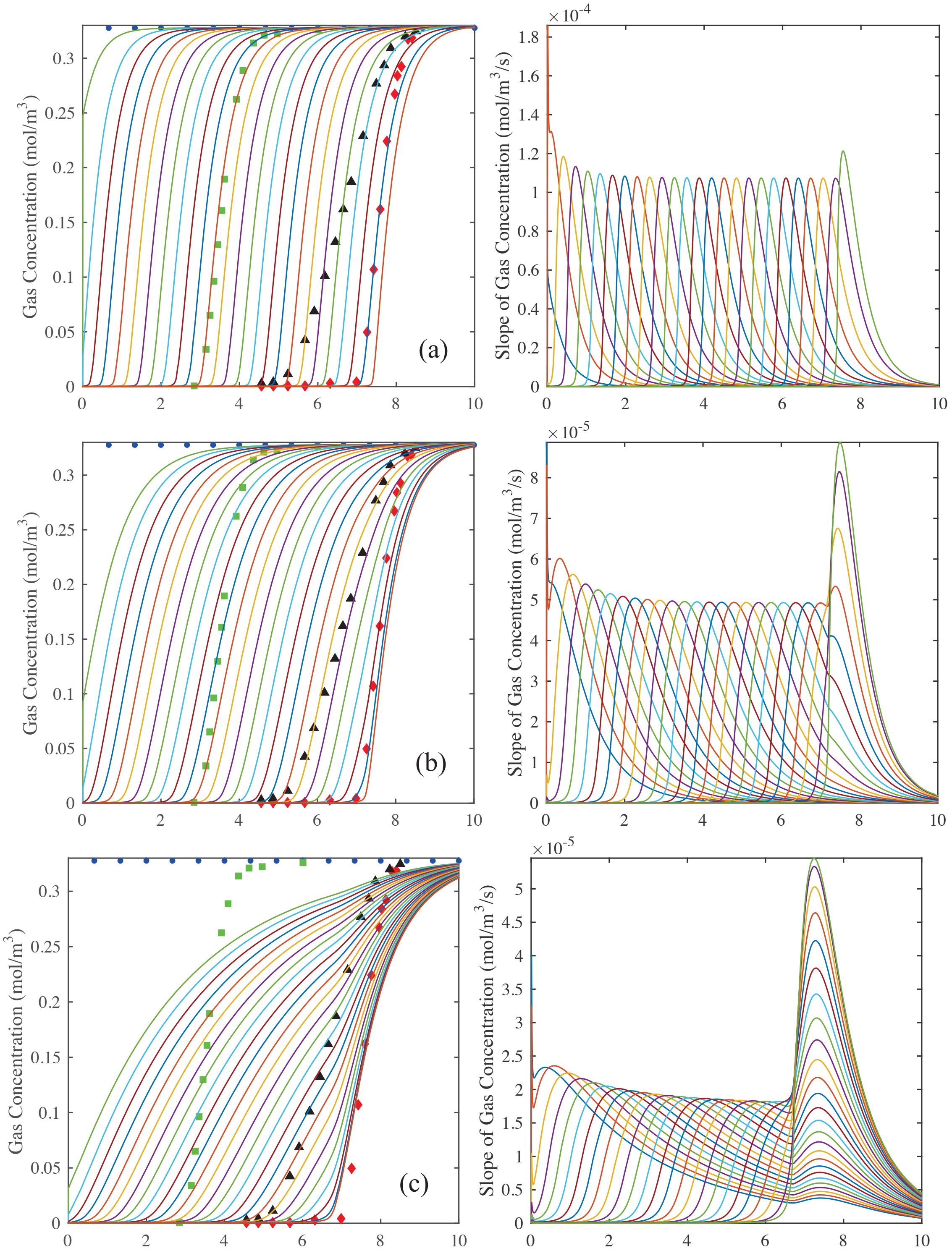


Figure 4.11 H_2O vapor on zeolite 5A: Predictions from the 1-D axial dispersed plug flow model of the outside the bed (triangles) experimental breakthrough curve when varying the value of D_L. D_L = 10 (dotted lines), 30 (dashed lines), 50 (solid lines), and 70 (dash-dot lines) times greater than Wakao and Funazkri correlation with the LDF k_n = 0.00083 s^{-1} (left panel). The reported saturation term for the H_2O-zeolite 5A isotherm was used, along with the reported void fraction of 0.35. The corresponding predictions from the model (lines) of the 2.5% (circles), 50% (squares), and 97.5% location (diamonds) experimental center line temperature profile histories are shown in the right panel.

Furthermore, the fact that H_2O vapor required such a large value of the dispersion coefficient (~50 times the value from the Wakao and Funazkri correlation) to capture the shape of the experimental concentration breakthrough curve measured just outside the bed was inconsistent with the value required by the CO_2 system, which was only 7 times the value from the Wakao and Funazkri correlation. If the dispersion mechanism explaining these deviations was indeed the same for both adsorbate/adsorbent systems, then the respective deviations from the Wakao and Funazkri correlation should have also been about the same. The explanation to this apparent conflict was associated with the breakdown of the 1-D axially dispersed plug flow model, wherein its inherent limitations prevented it from accounting for dispersion phenomena beyond that associated with molecular diffusion, especially for systems with highly non-linear Type I isotherms (Coppola and LeVan, 1981; Coppola and LeVan, 1983).

Figure 4.12 again compares the model to the experiment for H_2O vapor on zeolite 5A using the fitted $k_n = 0.00098\ s^{-1}$ and for values of the dispersion coefficient that were 1, 7, 30, and 50 times larger than that predicted from the Wakao and Funazkri correlation (Equation (4.10a)). Both the gas-phase concentration profile histories at numerous locations in the bed and the corresponding slopes are displayed, along with the experimental gas-phase concentration breakthrough curves within and just outside the bed. There are a number of characteristic features in this set of graphs that revealed the issues associated with the use of the 1-D axial dispersed plug flow model with this adsorbate/adsorbent system. First, note that the results in Figure 4.12a correspond to those used to obtain the k_n. As such, the shapes of the experimental gas-phase concentration profiles in the bed matched quite well with those predicted from the model. Also, note that the model clearly predicted CPB, as observed by the maximum in the slopes gradually approaching a constant value, except just at the end of the bed where the onset of concentration front sharpening was predicted by the model. This phenomenon was revealed by the maximum in the slope increasing slightly beyond that clearly associated with CPB. However, in this case, the concentration front sharpening was not enough to distort the internal gas-phase concentration profiles predicted from the model, thereby resulting in a reasonable value for k_n when the model was fitted to the experimental centerline gas-phase concentration profile at the 97.5% location. Despite these insignificant effects on the gas-phase concentration profile histories and also on the temperature profile histories (Figure 4.9), the results in Figure 4.12a began to expose the fact that the 1-D axial dispersed plug flow model might predict erroneous results for some systems. The results in Figures 4.12b, 4.12c, and 4.12d were even more revealing.


Gas Concentration (mol/m^3)
Slope of Gas Concentration (mol/m^3/s)
(a)
(b)
(c)
×10^{-4}
×10^{-5}
×10^{-5}
0
0.05
0.1
0.15
0.2
0.25
0.3
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
1
2
3
4
5
6
7
8
0.5
1.5
2.5
3.5
4.5
2
4
6
8
10

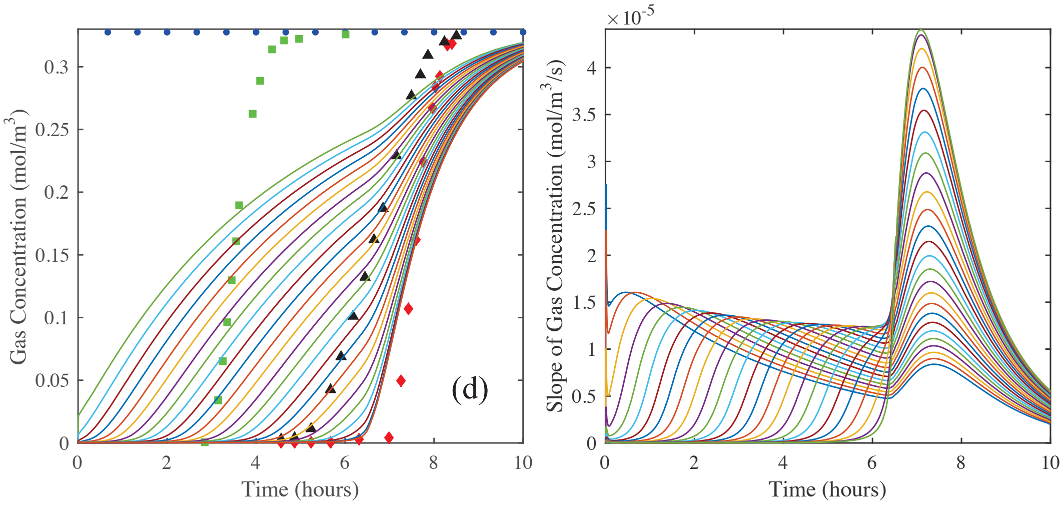


Figure 4.12 H_2O vapor on zeolite 5A: Predictions from the model (lines) shown in Figure 4.11 of the gas-phase concentration breakthrough curves at 0, 4, 8, 12, …, 92, 96, and 100% locations in the bed (left panels). The 2.5% (circles), 50% (squares), and 97.5% location (diamonds) experimental centerline gas-phase concentration breakthrough curves are also shown for comparison in the left panels. The corresponding derivatives (or slopes) of the gas-phase concentration breakthrough curves in the bed are shown in the right panels. (a) D_L = Wakao-Funazkri correlation, (b) D_L = 7, (c) 30, and (d) 50 times greater than Wakao and Funazkri correlation.

Figures 4.12b, 4.12c, and 4.12d show increasingly worse distortions of the gas-phase concentration profile histories and corresponding slopes predicted from the model near the exit of the bed when using values of D_L = 7, 30 and 50 times that predicted by the Wakao and Funazkri correlation. It was interesting that when the value of D_L was just 7 times greater (Figure 4.12b), the shapes of the internal gas-phase concentration profiles predicted from the model agreed quite well with the experimental concentration profile obtained just outside the bed. The fact that a value 7 times greater was required by CO_2 to fit the experimental breakthrough curve outside the bed (Figure 4.7) was not a coincidence and further substantiated that the same non-plug flow dispersive mechanism prevailed for both the CO_2 and H_2O vapor systems, independent of the adsorbate. The reason the model required a value ~50 times greater to fit the H_2O vapor experimental breakthrough curve outside the bed (Figure 4.11) was due to extensive concentration front sharpening occurring for the H_2O vapor system, as shown especially in Figures 4.12c and 4.12d. To overcome it and make the concentration front more dispersed, an artificially large value of the dispersion coefficient was required. The difference between these systems is due to the much stronger attractive force between H_2O and the highly polar zeolite 5A than between CO_2 and 5A. The H_2O molecule has a strong dipole moment of 1.8 Debye, compared with a moment of 0 Debye for CO_2 (Poling et al., 2000). It should be noted that the CO_2 molecule does have a quadrupole moment resulting in a greater attractive force between it and zeolite 5A than other atmospheric gases. In Chapter 6 a formal relationship between the degree of breakthrough sharpening and the attractive force between sorbate and sorbent as indicated by the steepness of the equilibrium capacity isotherm is obtained.

The results in Figure 4.12 further show that, at such large values of D_L, the concentration front sharpening actually propagated all the way to the entrance of the bed, as observed in Figure 4.12d. In this case, not only was CPB clearly not preserved anywhere in the bed, but also, and more importantly, the predicted breakthrough curves outside the bed no longer provided any useful fundamental information. Evidence for this supposition was provided by the experimental non-plug flow dispersive behavior of the bed being predicted very well by a value of D_L that was only 7 times greater, not 50 times greater, than that from the Wakao and Funazkri correlation.

The problem with the correctly derived Danckwerts boundary condition at the exit of the bed for the Fickian axial diffusion model (Equation (4.3)) stems from its inability to correctly describe the dispersive or non-convective aspect of the flux, even under a plug flow regime, as was just observed. The resulting mathematically derived zero slope, as required by the satisfaction of the continuity of both concentration and flux in the Fickian diffusion model, is not preserved experimentally when breakthrough takes place at the boundary because of the complicated dispersive dynamics. However, forcing the slope of the concentration front to be zero at the exit of the bed for a system with a steep Type I isotherm, like the H_2O vapor-zeolite 5A system, has such a large influence on the mass balance that it causes unusually large changes in the gas-phase concentration near the exit of the bed. This results in concentration front sharpening and loss of CPB near the exit of bed that may propagate throughout the entire bed, as clearly revealed in the modeling results in Figure 4.12 for this system.

4.6. Modeling Conclusions

This work with CO_2 and H_2O vapor on zeolite 5A revealed that special caution must be taken when using typical experimental breakthrough curves measured just outside the bed to extract mass transfer and dispersion information from a fixed-bed adsorber based on the widely utilized 1-D axial dispersion plug flow model; otherwise, the resulting information may be erroneous. An experimental breakthrough curve measured just outside the bed, as commonly practiced, should, in principle, provide fundamental adsorbate/adsorbent mass transfer information when such a model is fitted to the data. This is because the dispersion coefficient predicted from known correlations does not have a significant impact on the model results.

However, this work showed that, to use these correlations, the design of the fixed-bed adsorber must satisfy the plug-flow condition. This work also showed that the only way to verify when the plug flow condition was satisfied was to compare experimental breakthrough curves obtained outside the bed with those obtained inside the bed along its axial center. From this comparison, it was determined that even a well-accepted bed diameter to pellet ratio of about 20 was not large enough to ensure plug-flow conditions prevailed in the bed. The experimental outside of bed and inside of bed centerline breakthrough curve results consistently revealed that the bed was experiencing considerable near-wall channeling, i.e., mechanical dispersion phenomena.

Because of the presence of mechanical dispersion, the 1-D axial dispersed plug flow model could not simultaneously predict the experimental concentration profile histories obtained just outside the bed and the experimental centerline temperature profile histories measured inside the bed for either adsorbate/adsorbent system. It was deduced that the temperature profile histories reflected the adsorbate/adsorbent mass transfer process involved, while the outside of bed concentration profile histories reflected a mixing process akin to a non-plug flow pattern existing in the bed that was independent of the adsorbate, i.e., near-wall channeling. It was nevertheless shown that the sought after fundamental mass transfer information could still be obtained, in this case an LDF k_n for each adsorbent/adsorbate system, when experimental centerline gas-phase concentration and temperature profile histories were measured somewhere in the bed. It is therefore strongly recommended that this be the preferred method for obtaining experimental mass transfer information from a 1-D axial dispersed plug flow model.

Moreover, despite the alluded to limitations of the 1-D axial dispersed plug flow model, an effort was put forth to extract a dispersion coefficient from the model using the experimental outside the bed breakthrough curves that inherently included the non-plug flow dispersion taking place. In this case, the k_n values obtained by fitting the experimental inside the bed breakthrough curves were used. The resulting D_L values for CO_2 and H_2O vapor were both 7 times greater than the largest value predicted from five established D_L correlations. This analysis confirmed the unique nature of the non-plug flow mechanical dispersion phenomena taking place in the bed, and it correctly showed that such phenomena should be independent of the adsorbate. However, while obtaining D_L, significant differences were observed between the CO_2 and H_2O vapor systems. Extracting D_L from the experimental data for CO_2 was straightforward, but not for H_2O vapor.

The process of extracting D_L from the experimental data for H_2O vapor revealed the mathematical inability of the 1-D axial dispersed plug flow model to obtain such information at the exit of the bed. Depending on the value of D_L, significant concentration front sharpening occurred for this system. This concentration front sharpening is an unusual but real phenomenon that is scarcely known and a consequence of the limited ability of the 1-D axial dispersed plug flow model and its Danckwerts boundary condition to represent non-diffusive dispersive mechanisms for very rectangular Type I isotherms, such as H_2O vapor in zeolite 5A. To obtain a D_L value for H_2O vapor that was consistent with that obtained for CO_2, the predicted inside the bed centerline breakthrough curves, necessarily chosen from a location unaffected by any concentration front sharpening, were matched to the experiential outside the bed breakthrough curve. Otherwise, the D_L value obtained for H_2O vapor when using the experimental outside the bed breakthrough curves was 50 times greater, as opposed to 7 times greater, due to compensating for the significant concentration front sharpening that the model predicted for this system.

Overall, this work clearly showed that it was possible to extract accurate mass transfer and dispersion information from experimental breakthrough curves using a 1-D axial dispersed plug flow model when they were measured both inside and outside the bed. To ensure the extracted information was accurate, inside the bed breakthrough curves and their derivatives from the model were plotted to confirm whether or not the adsorbate/adsorbent system was exhibiting CPB or any concentration front sharpening near the bed exit. Even when concentration front sharpening was occurring, as with the H_2O vapor-zeolite 5A system, it was still possible to use the experimental inside and outside the bed breakthrough curves to extract fundamental mass transfer and dispersion information from the 1-D axial dispersed plug flow model based on the systematic methodology developed in this work.

4.7 Ramifications for Large and Small Diameter Fixed-beds

This section has shown that accurate values for the LDF mass transfer coefficient k_n may be obtained via experimental breakthrough analysis from small diameter columns (that is, with a bed to pellet diameter ratio of approximately 20) given that concentration measurements are taken inside the bed at the centerline. This value for k_n is independent of wall effects and may therefore be applied for scale-up studies, where the final column diameter is much larger. For large beds, the correlations for the axial dispersion coefficient D_L provided in the literature and discussed in Section 4.2.5 appropriately capture dispersion effects due solely to pellet packing (again independent of wall effects).

This initial work has also shown that it is possible to obtain accurate values for D_L for small diameter beds for the two sorbate/sorbent systems studied by increasing the value of D_L well beyond the values given by the correlations. The larger values of D_L are required to account for wall effects (channeling), which are important to model for a small diameter column, as they have a strong dispersive effect on the separation process. However, for the H_2O on 5A system, the internal concentration history must be used to determine D_L since, as discussed in Section 4.5.4, the axial dispersed plug flow model and Danckwerts boundary condition in combination with rectangular isotherms and high values of the axial dispersion coefficient result in non-physical breakthrough sharpening for systems with rectangular isotherms (Coppola and LeVan, 1981; Coppola and LeVan, 1983; Mahle and Friday, 1991).

The literature research in Section 2.4 found that small diameter columns are generally used in breakthrough analysis presumably to conserve sorbent material and reduce experimental costs associated with larger experimental systems. If much larger diameter columns are used in the final production system, channeling does not have a significant effect. However, some production gas separation systems do employ small diameter columns and thus obtaining an accurate value for D_L including channeling effects is important to capture resulting inefficiencies. For example, as shown in Figure 4.13, the CO_2 removal system in use on the ISS uses CO_2 sorbent beds with imbedded heat sheets and fins as heat spreaders, which together divide the bed space into a large number of small individual channels (Coker et al, 2015). In this case, the bed to pellet diameter ratio is only about six.

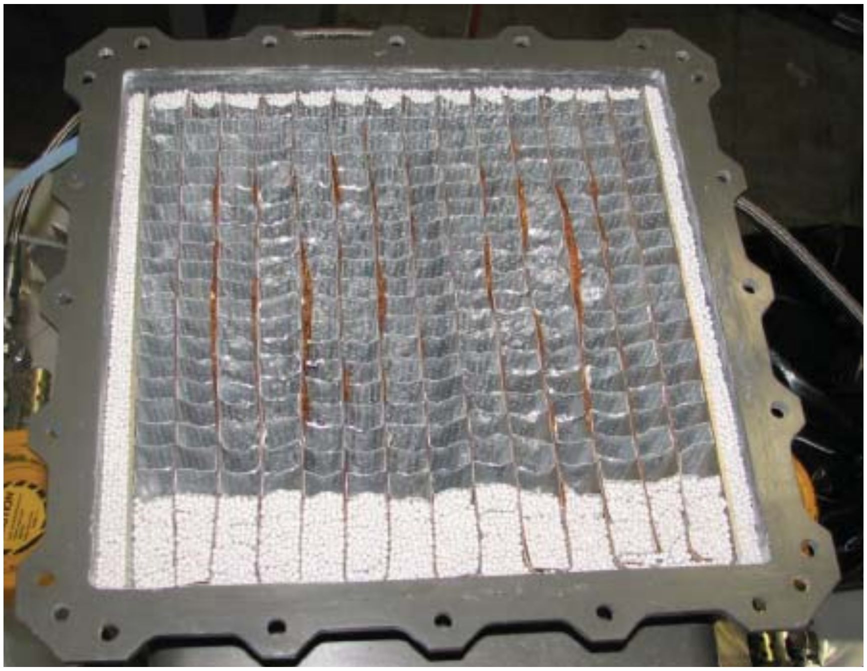


Figure 4.13 Carbon dioxide removal system sorbent bed. Heater sheets are brown in color and run vertically. Folded aluminum channel fins (gray) distribute heat from the heater sheets. The sorbent material (light tan) has been partially loaded to the outside and lower portion of the bed (Coker et al., 2015).

Two other applications that make use of small diameter beds are preparative and production chromatography. Similarly, in analytical GC, a pulse of mixed gas is carried through a fixed-bed using either helium or hydrogen as a carrier gas. As the pulse moves through the column, the gas mixture is separated due to differing adsorptive properties. Unlike analytical GC where a single pulse is separated and analyzed, in preparative and production chromatography pulses are sent in an optimized periodic pulse sequence that maximizes production rate. The separated gases are collected sequentially as they exit the column. However, bed diameters are limited, as maintaining consistent packing (voidage) through the bed is crucial to efficient separations, and becomes more difficult with increasing bed size (Ruthven, 1984; Yang, 2003). Other small diameter bed applications include those that essentially employ packed bed heat exchangers, similar to that shown in Figure 4.13, but with internal fluid passages for cooling and heating of buildings and vehicles (Alefeld et al., 1981; Lu et al, 2003; Saha et al., 2003).

As previously mentioned, capturing the dispersive effects of channeling is important for accurate simulation of gas separations utilizing small diameter fixed-beds. In Chapter 4, a systematic methodology was presented to extract the value for the dispersion coefficient D_L based on experimental breakthrough curves from inside the column and far downstream or outside the bed. This approach is used successfully for the CO_2 on 5A system as shown in Figure 4.10, where the simulated outside breakthrough curve is matched to experimental data by increasing the axial dispersion coefficient seven times over the Wakao and Funazkri correlation.

However, the same approach did not yield the same results for the H_2O on the 5A system. As shown in Figure 4.12b for the H_2O on the 5A system, increasing the axial dispersion coefficient seven times over the Wakao and Funazkri correlation did provide a reasonable match of the slope of the experimental outside breakthrough curve with that of the simulated internal concentration history. However, in comparison, the slope of the simulated outside breakthrough curve has sharpened considerably. This is clearly illustrated by the accompanying plot in Figure 4.12b showing the slope, or time derivative, of the concentration history. Further increases in D_L heightened the breakthrough sharpening effect, and in addition, resulted in a complete breakdown of the expected constant pattern behavior.

CHAPTER V

5. MAPPING THE SENSITIVITY OF SORBATE/SORBENT SYSTEMS TO THE AXIAL DISPERSION COEFFICIENT AND LDF COEFFICIENT

5.1 Introduction

In this section, relationships to define the threshold values, or largest permissible values to obtain physically consistent simulation results, for the axial dispersion coefficient and the LDF coefficient k_n are determined for various sorbent/sorbate systems. By keeping the values of D_L and k_n below these threshold values, non-physical and therefore non-predictive simulation results may be avoided in the design of gas separation processes. In order to generalize the threshold value relationships to any sorbate/sorbent system, the equilibrium capacity isotherm is approximated by the separation factor (see, for example, Yang, 1997; LeVan and Carta, 2008). Finally, a relationship between the threshold value and the separation factor is developed. The practical use of this relationship is the ability to determine the physical/non-physical threshold for any sorbate/sorbent pair through calculation of its separation factor.

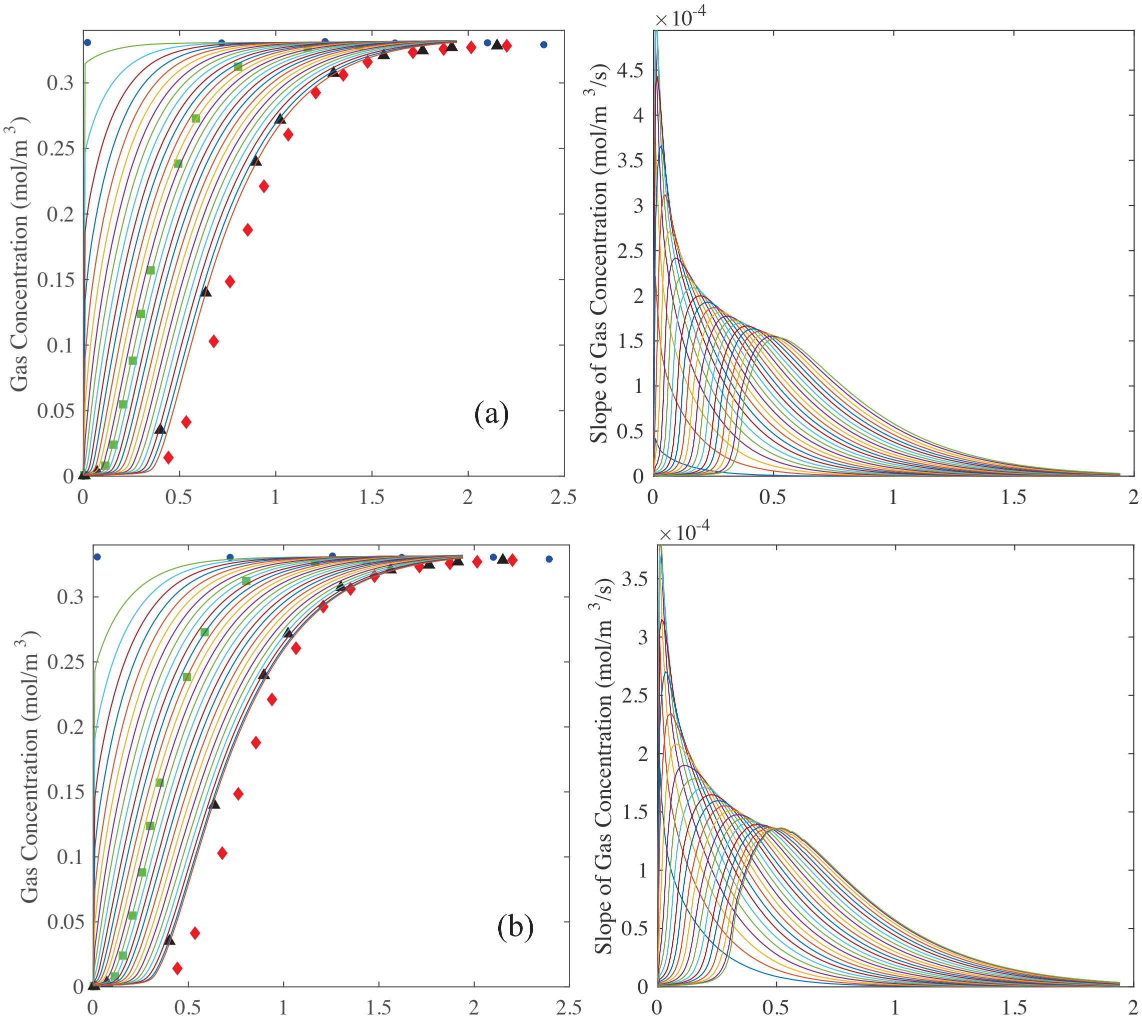
5.2 Identifying the Non-physical Threshold for CO_2 on Zeolite 5A

In Section 4.5.3, the LDF mass transfer coefficient k_n is determined for the CO_2 on the zeolite 5A system by comparison with the experimental centerline gas-phase concentration history profile, as shown in Figure 4.8. This figure also illustrated that the simulation results were essentially unchanged when using values for the axial dispersion coefficient at the extremes of the correlations considered. Thus, the particle-scale dispersion was shown to have a small influence on the breakthrough results.

However, the axial dispersion coefficient found to provide the best fit of the outside bed gas-phase concentration history profile, as shown in the left panel of Figure 4.10, was seven times that of the Wakao and Funazkri correlation. This large of a value was required to compensate for the channeling present where the column to pellet diameter ratio was about 20. The time derivative, or slope, of the simulated gas-phase concentration history is shown in the center panel of Figure 4.10.

In Figure 5.1, the simulation conditions shown in Figure 4.10 are repeated, but now starting with axial dispersion per Wakao and Funazkri (Figure 5.1a) and then 7, 30, and 50 times the Wakao and Funazkri value, as was shown for the water on 5A system in Figure 4.12. In Figure 4.12a the constant pattern behavior is developing but has not yet been attained before initial breakthrough occurs. For simulations that do not attain constant pattern behavior, a logical criterion to avoid non-physical simulation behavior is that the maximum slope of the breakthrough curve is not significantly greater than the maximum slope of the internal concentration history. In other words, simulations with slope ratio greater than 1.0 will have non-physical behavior. Figure 5.1b illustrates a reasonable limiting case where slight breakthrough sharpening occurs without noticeable distortion to the internal gas-phase concentration history.

Figures 5.2c and 5.2d show cases where the criterion stated above is not met, that is, the maximum slope of the breakthrough curve is significantly greater than the maximum slope of the internal concentration history. Here, it is also evident that the exit boundary distortion has permeated through the simulation of the entire bed: unlike in Figures 5.2a and 5.b, the simulations shown in Figures 5.2c and 5.2d are not approaching the constant pattern profile as expected.



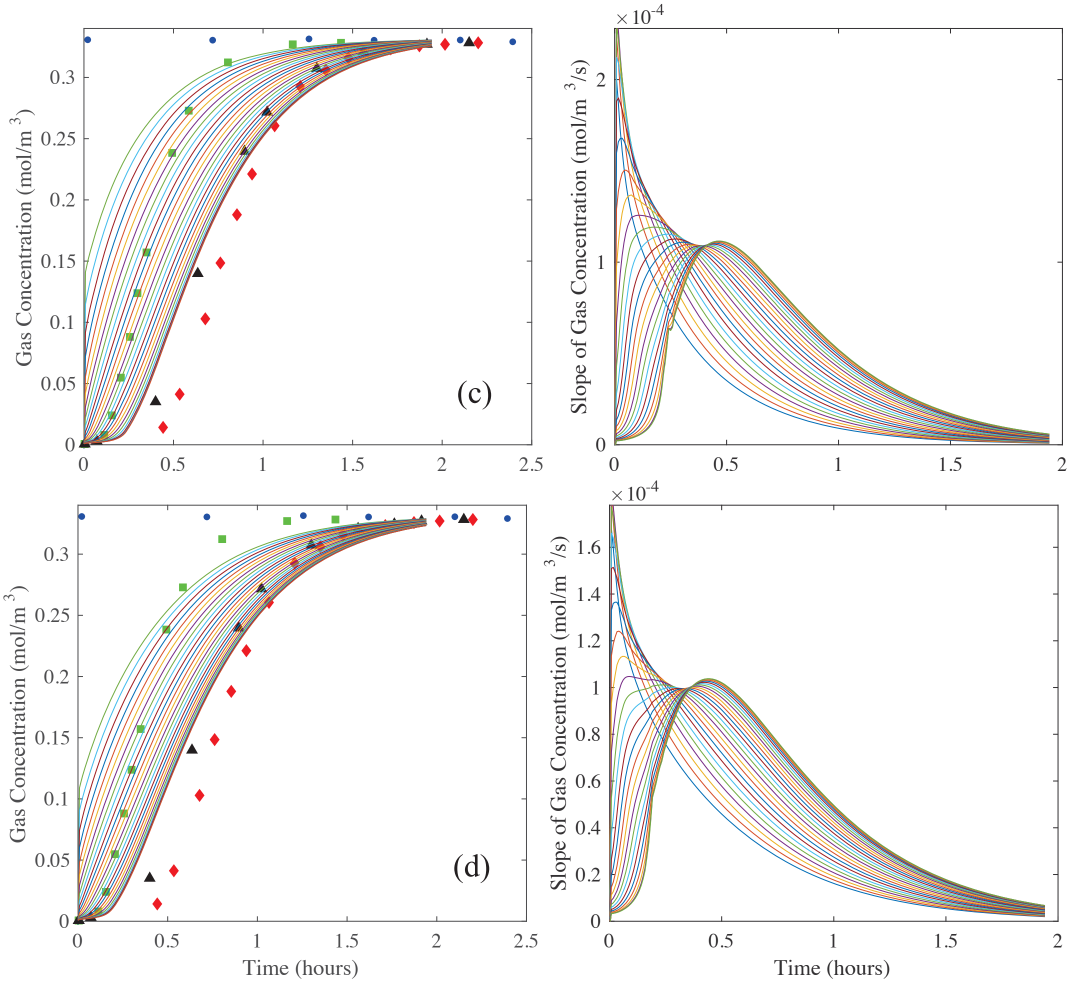


Figure 5.1 Carbon dioxide on zeolite 5A: Predictions from the model (lines) of the gas-phase concentration breakthrough curves at 0, 4, 8, 12… 92, 96, and 100% locations in the bed (left panels). The 2.5% (circles), 50% (squares), and 97.5% location (diamonds) experimental centerline gas-phase concentration breakthrough curves are also shown for comparison in the left panels. The corresponding derivatives (or slopes) of the gas-phase concentration breakthrough curves in the bed are shown in the right panels. (a) D_L = Wakao-Funazkri correlation, (b) D_L = 7, (c) 30, and (d) 50 times greater than Wakao and Funazkri correlation.

For columns with smaller column to pellet diameter ratios, such as those shown in Figure 4.13, the influence of channeling on the adsorption process is anticipated to be of even greater significance. In this case, a larger dispersion coefficient would be required for a good fit of the simulation to the outside of the bed breakthrough curve shape. For other sorbents with higher mass transfer rates, for example, zeolite LiLSX, a higher LDF coefficient will also be required for a good fit (Knox et al., 2016b). The influence of the LDF coefficient on simulations with respect to non-physical results will be illustrated below.

To determine the combined influence of the magnitudes of the axial dispersion coefficient and the LDF coefficient on the simulation of the CO_2 on zeolite 5A system, a large number of simulations were conducted using the VATS Matlab® program used in conjunction with COMSOL®. The matrix of simulation runs is shown graphically in Figure 5.2. For each simulation, the ratio of the maximum slope of the breakthrough curve, or maximum derivative of concentration with respect to time, to the maximum slope of the internal concentration profile was calculated. This may be observed graphically from Figure 5.1 by considering an imaginary line connecting the maximum values of each derivative curve. The slope ratio is then the ratio of the maximum value of this imaginary line, which occurs at the breakthrough curve, to the minimum value, which occurs in at an interior location. For the simulation shown in Figure 5.1(c) and (d), the slope ratio will be significantly greater than one, indicating that it is non-physical. For the Figure 5.1(b), the slope ratio will be approximately one, which was chosen to be the limiting value in this study, as the observed concentration sharpening was insignificant.

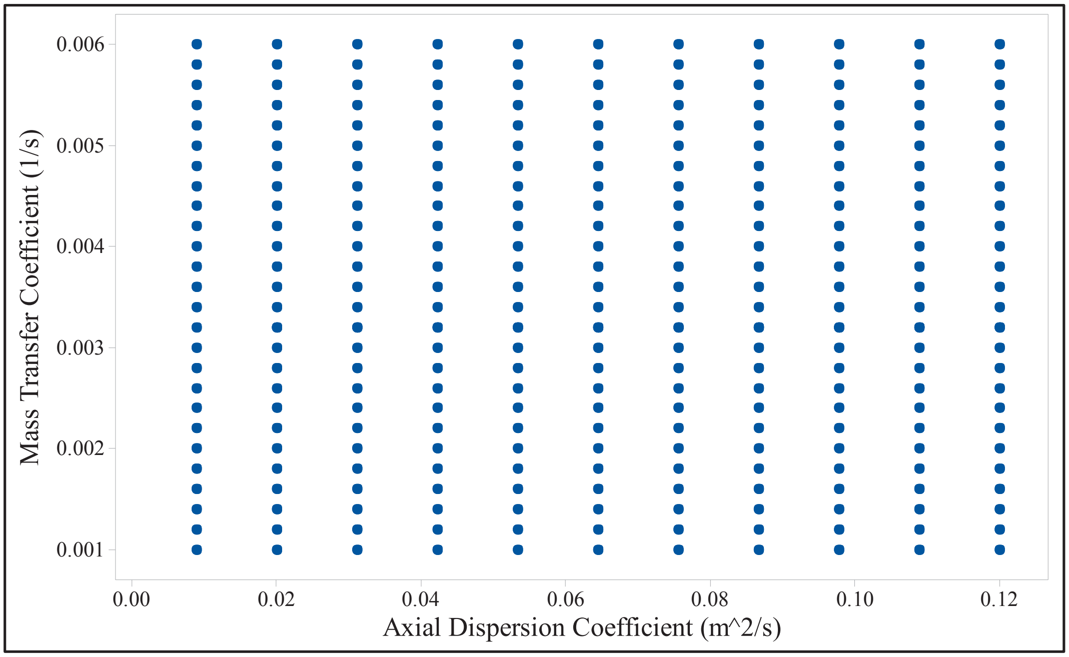


Figure 5.2 Matrix of simulation runs for the CO_2 on zeolite 5A system. Each point represents a single breakthrough run with the values for D_L shown on the x-axis and the values for LDF shown on the y-axis. In total, 275 simulations were performed.

The matrix of slope ratio values was imported in Minitab® 17, which was used to create the contour plot shown in Figure 5.3. Minimum slope ratio values occur for lower values of both D_L and k_n. Maximum slope ratios occur for higher values of both D_L and k_n. Considered independently, changes in D_L and k_n have similar influence on the magnitude of the slope ratio, though the vertical contour lines are closer than the horizontal lines indicating the D_L has the stronger influence of the two.

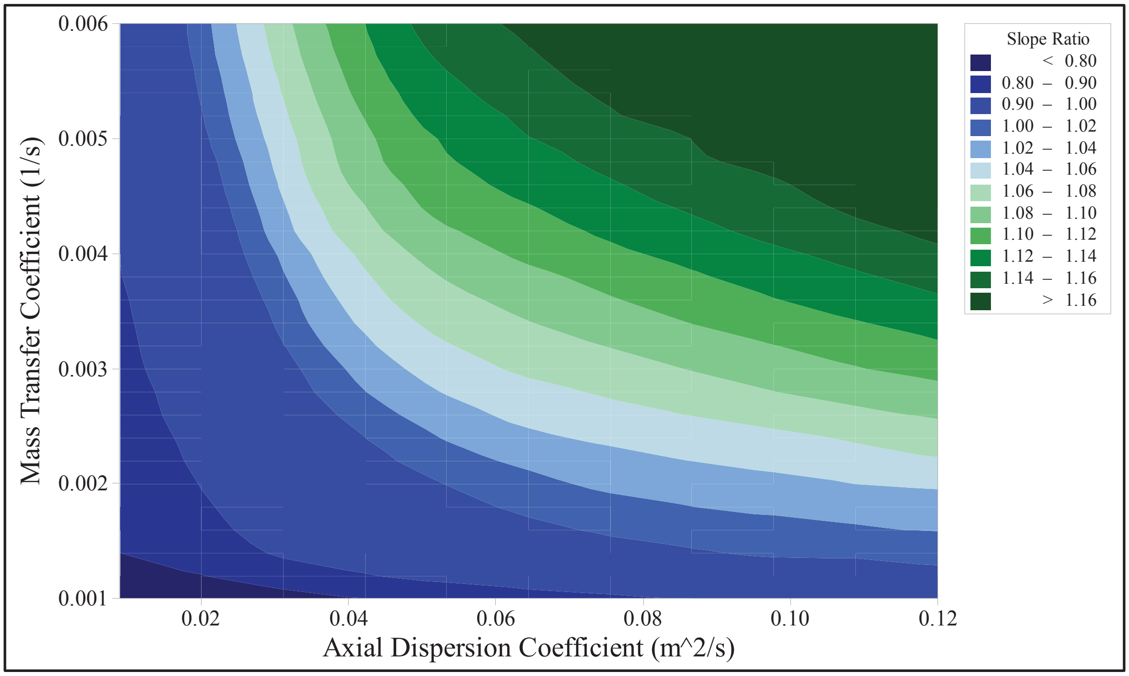


Figure 5.3 Contour plot showing the slope ratio values for a series of breakthrough curves for CO_2 on 5A with the values for D_L shown on the x-axis and the values for k_n shown on the y-axis. The colors corresponding to the slope ratio values are shown in the legend.

To define the values of k_n and D_L at the onset of non-physical behavior in order to determine *a priori* what combination of values will result in a predictive simulation, a relationship between k_n and D_L in the threshold region is desired. The upper bound of the threshold region was defined at 1.0 above, and the lower threshold is set such that sufficient points are available to generate a reasonable curve fit. Figure 5.4 illustrates more clearly the threshold region in a contour plot similar to Figure 5.3, but here focusing on simulation results with slope ratios between 0.98 and 1.0. Using TableCurve® 2D, a curve fit for these simulations resulted in a threshold region that is shown in Figure 5.5. The relationship is given in Equation (5.1), with θ defined as the threshold parameter.

For this system, the simulation will result in a slope ratio limit below the threshold if the conditions in Equation (5.2) are satisfied with a threshold parameter of 10241. In the following sections, the threshold parameters for other sorbent/sorbate systems are obtained.

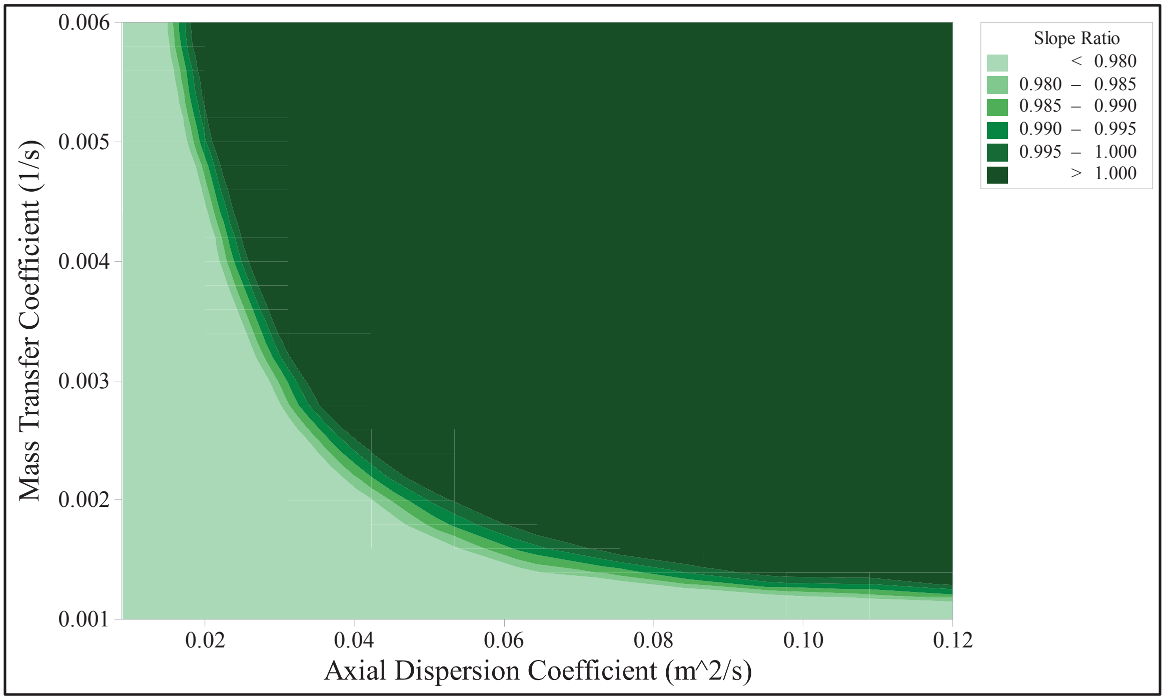


Figure 5.4 Contour plot with colors differentiated only for slope ratios greater than 0.98 and less than 1.0.

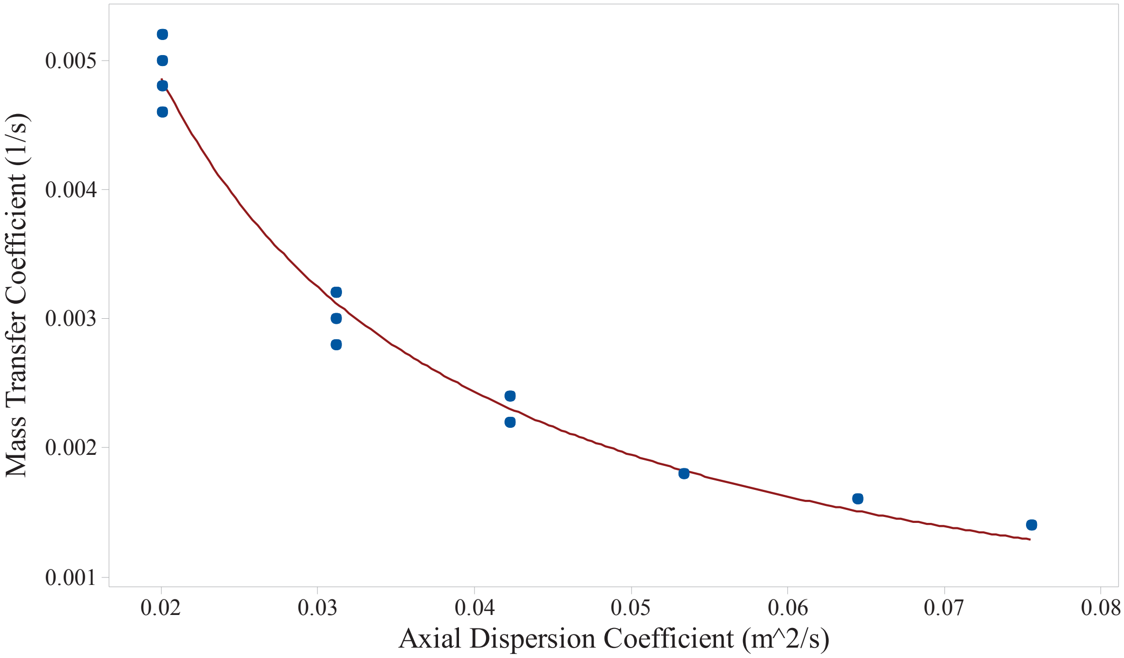


Figure 5.5 Comparison of curve fit equation (line) to breakthrough simulations with slope ratio between 0.98 and 1.0 (points). Coefficient of determination (R^2) is 0.982.

$$k_n = \frac{1}{\left(1+\theta D_L\right)} \tag{5.1}$$

$$k_n\left(1+\theta D_L\right)-1<0 \tag{5.2}$$

5.3 Identifying the Non-physical Threshold for H_2O on Zeolite 5A

The VATS Matlab® program is again used in conjunction with COMSOL® and Minitab® to obtain the threshold parameter for the H_2O on the zeolite 5A system. A series of simulations were performed for the matrix of D_L and k_n values shown in Figure 5.6. For each simulation, the ratio of the maximum slope of the breakthrough curve, or maximum derivative of concentration with respect to time, to the maximum slope of the internal concentration profile were again calculated. A contour plot of these slope ratios is shown in Figure 5.7.

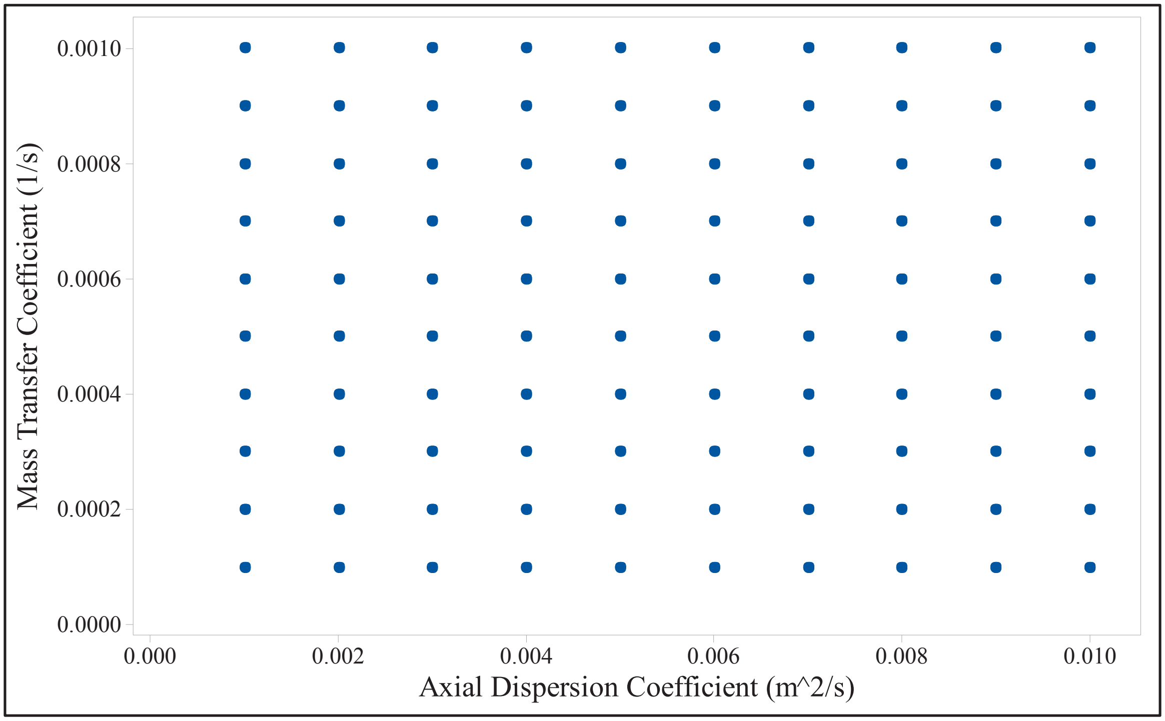


Figure 5.6 Matrix of simulation runs for the H_2O on zeolite 5A system. Each point represents a single breakthrough run with the values for D_L shown on the x-axis and the values for LDF shown on the y-axis.

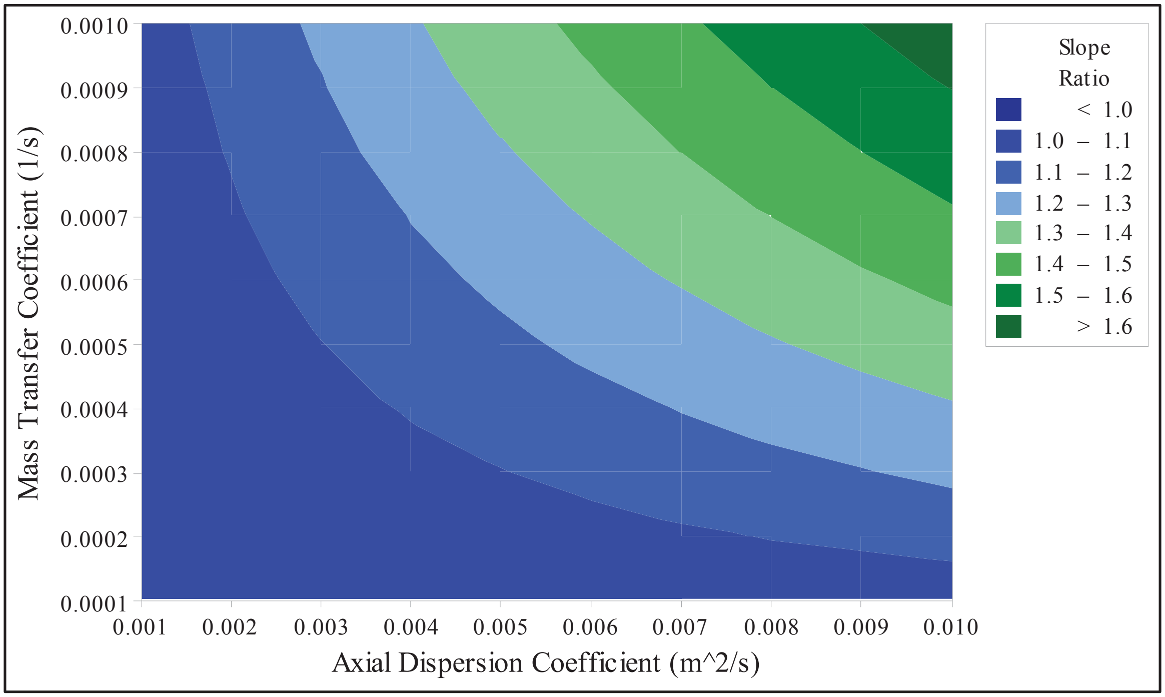


Figure 5.7 Contour plot showing the slope ratio values for a series of breakthrough curves for H_2O on 5A with the values for D_L shown on the x-axis and the values for k_n shown on the y-axis. The colors corresponding to the slope ratio values are shown in the legend.

In Figure 4.12, a series of simulations illustrates the increasing distortion of the H_2O/5A system with increasing D_L. The Wakao and Funazkri correlation for axial dispersion was used for the simulation in Figure 4.12a. These results show that constant pattern behavior has been reached for this system, and that although a small amount of breakthrough sharpening is present, the effect of the internal gas-phase concentration histories, and on breakthrough sharpening, are minimal. An acceptable slope ratio limit of 1.15 was determined based on these simulation results.

Figure 5.8 shows a contour plot for a much narrower band of the slope ratio to show the approximate location of the slope ratio threshold. To obtain a relationship between D_L and k_n values and the slope limit threshold, the simulation points falling within the band in Figure 5.8 were again fit to Equation (5.1) as shown in Figure 5.9. In this case, the threshold parameter θ was found to be 522353, or more than 66 times the threshold parameter for the CO_2 on 5A system. It will be shown in the following section that a similar ratio occurs between the H_2O and CO_2 on zeolite 13X systems, and that the threshold ratio is closely related to the equilibrium capacity.

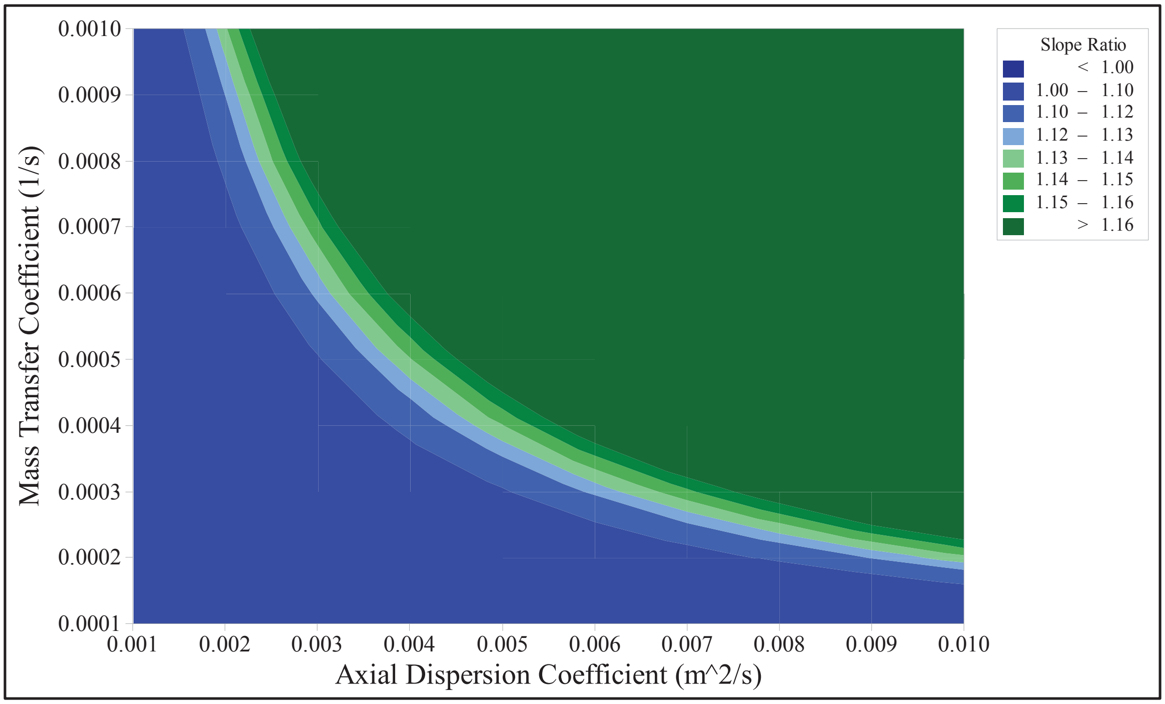


Figure 5.8 Contour plot with colors differentiated only for slope ratios greater than 1.0 and less than 1.15.

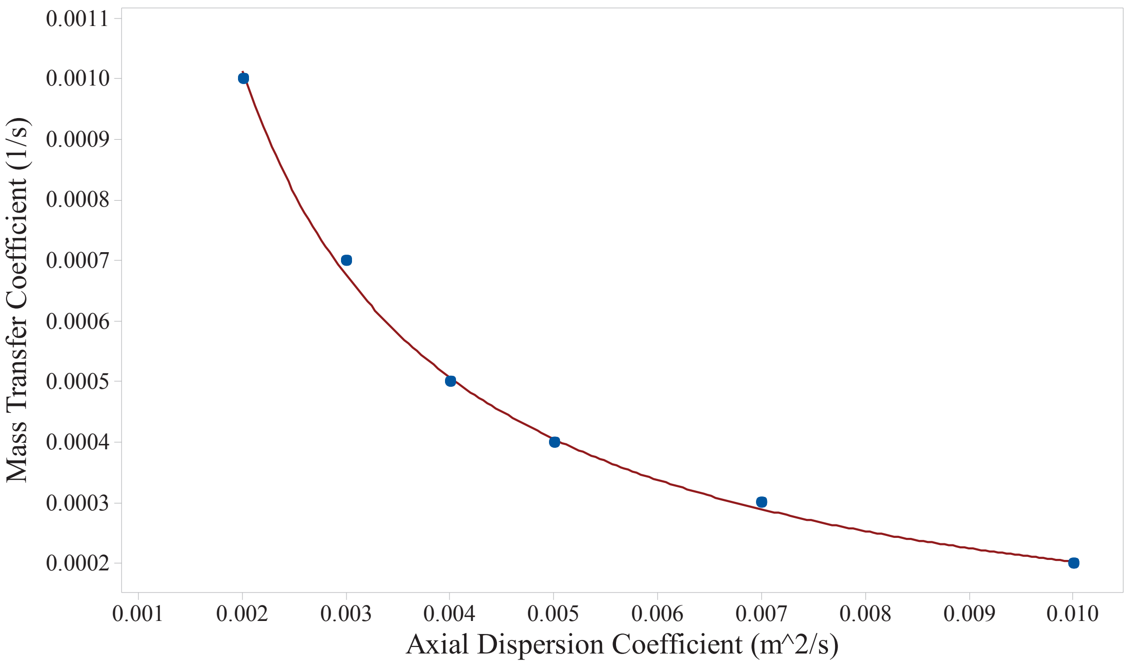


Figure 5.9 Comparison of curve fit equation (line) to breakthrough simulations with slope ratio between 1.0 and 1.15 (points). Coefficient of determination (R^2) is 0.998.

5.4 Identifying the Non-physical Threshold for CO_2 and H_2O on Zeolite 13X

The approach taken in the previous sections was followed to develop limiting relationships for the sorbate/sorbent systems CO_2 on zeolite 13X and H_2O on zeolite 13X. Using the same rationale as used for the CO_2 on 5A system, the limiting value of the slope ratio is 1.0. For the H_2O on zeolite 13X system, the limiting value is again set to 1.15. Simulations with slope ratios from 0.98 to 1.0 are used to find the threshold ratio as defined by Equation (5.1) to be 11713 for the CO_2 on zeolite 13X system as shown in Figures 5.10 and 5.11. For the H_2O on zeolite 13X system, simulations with a slope ratio of 1.0 to 1.15 are used to find a threshold ratio of 679013. The threshold value for H_2O on 13X is approximately 58 times greater than for CO_2 on 13X, showing a similar trend as the same sorbates on 5A. The threshold values are summarized in Table 5.1. In addition, Table 5.1 provides the equilibrium loading values for the inlet conditions used in these simulations based on the Toth equation shown in Figure 4.9 and the equilibrium adsorption isotherm parameters in Table 4.1. It will be shown in the following section that the magnitude of the threshold values was closely related to the curvature of the equilibrium capacity isotherms, and that the threshold values may be predicted from the isotherms.

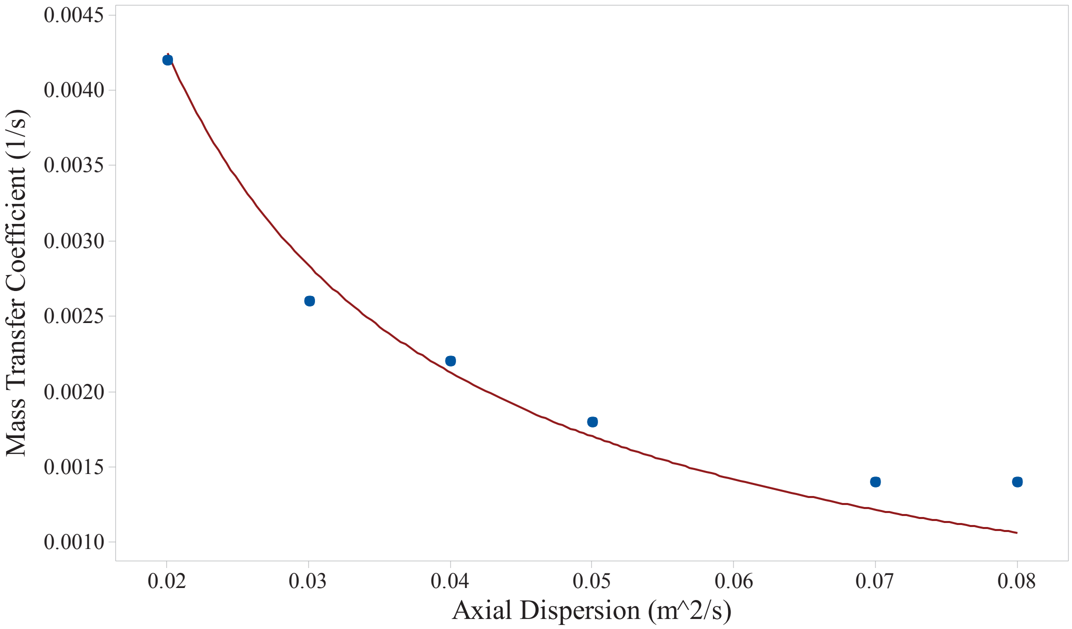


Figure 5.10 Comparison of curve fit equation (line) to CO_2 on 13X breakthrough simulations with slope ratio between 0.98 and 1.0 (points). Comparison of curve fit equation (line) to H_2O on 13X breakthrough simulations with slope ratio between 1.0 and 1.15 (points). Coefficient of determination (R^2) is 0.961.

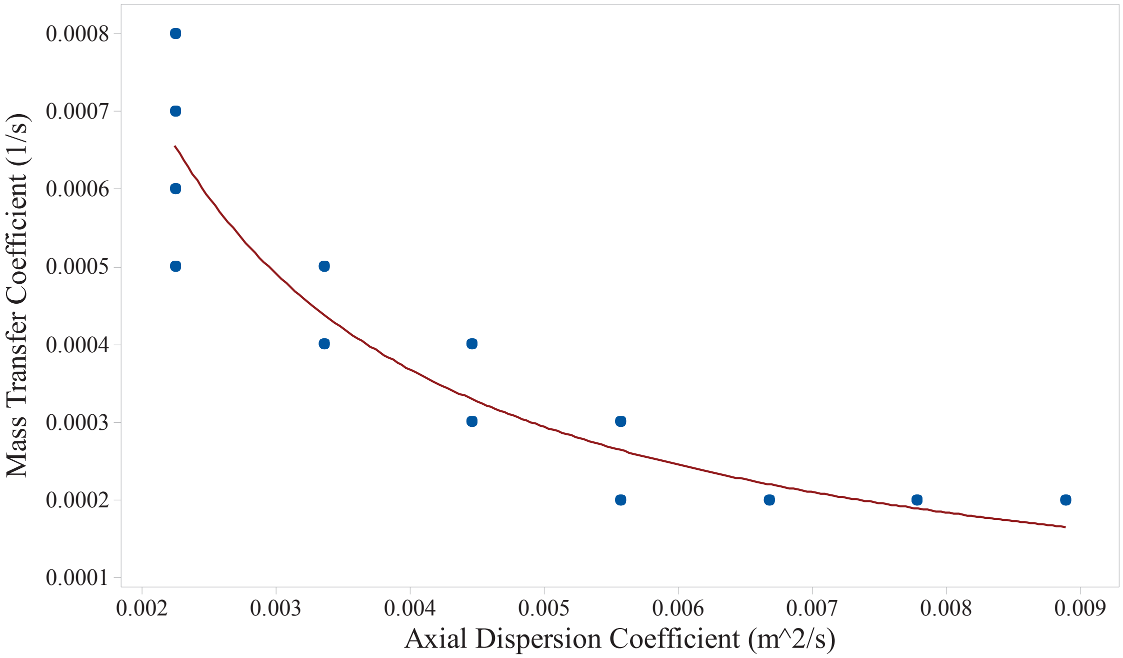


Figure 5.11 Comparison of curve fit equation (line) to H_2O on 13X breakthrough simulations with slope ratio between 1.0 and 1.15 (points). Coefficient of determination (R^2) is 0.860.

Table 5.1 Threshold parameter values for four sorbate/sorbent systems and equilibrium loading conditions at T_s = 22.4°C and p = 0.81 kPa.

Sorbate/Sorbent System	θ, s/m^2	n, mol/kg	$\theta_{H_2O}/\theta_{CO_2}$	n_{H_2O}/n_{CO_2}
CO_2 on zeolite 5A	10241	1.32	51	9.9
H_2O on zeolite 5A	522353	13.01		
CO_2 on zeolite 13X	11713	1.21	58	10.9
H_2O on zeolite 13X	679013	13.23		

5.5 Generalization to Other Sorbent/Sorbate Systems

The analysis in the preceding section provides practical limits on k_n and D_L for four specific sorbate/sorbent systems. Inspection of the values in Table 5.1 indicates that there is a relationship between the magnitude of the derived threshold values and magnitude of the equilibrium loading at these conditions. The loading is calculated via the Toth isotherm using the same Toth constants as used for the simulations (Wang and LeVan 2009). Since the strength of the sorbate/sorbent bond appears to have an influence on the threshold parameter, it seems reasonable to investigate the relationship between the threshold parameter and the sorbate/sorbent isotherms themselves. Given that the phenomenon of breakthrough sharpening is due to the numerical interaction of the Danckwerts boundary condition and the isotherms (Coppola and LeVan, 1981), it would follow that a steeper isotherm would result in numerical instability at lower k_n and D_L values. Due to the definition of the threshold parameter in Equation (5.1), a steeper isotherm also leads to higher values of the threshold parameter.

One useful indicator of the curvature of an isotherm is the distribution factor K_d (Yang, 1997), as shown in Equation (5.3). This form shown applies where sorbent is initially free of sorbate, such as for the breakthrough experiments presented in this work. The reciprocal of the distribution factor is the equilibrium factor $\beta = 1/K_d$, also called the separation factor (LeVan and Carta, 2008). For the Langmuir adsorption isotherm (shown in Equation (5.4)), the distribution factor may be simply expressed as shown in Equation (5.5). It should be noted that the local concentration, c, has been factored out of Equation (5.5) such that the loading is dependent only on the inlet concentration and the distribution factor:

$$K_d = \frac{q^*}{q_0^* - q^*} \frac{c_0 - c}{c}, \tag{5.3}$$

$$\frac{q^*}{q_m} = \frac{bc}{1 + bc}, \tag{5.4}$$

$$K_d = 1 + bc_0, \tag{5.5}$$

where q_m is the monolayer capacity, q_0^* is the equilibrium sorbate concentration corresponding to the inlet gas concentration, and b is an equilibrium constant. Since this work uses the Toth isotherm equation as shown in Equation (4.9), the isotherms must be recast into Langmuir isotherms to estimate the distribution factor K_d. However, as the Langmuir isotherm is a linear relationship, and the Toth isotherm is non-linear due to the exponential terms in the Toth isotherm, only an approximate fit could be achieved. Figure 5.12 compares the Toth and Langmuir isotherms for six sorbate/sorbent systems. In addition to the isotherms for systems discussed in this work, the isotherms for silica gel are shown to provide examples of isotherms that are essentially linear (CO_2 on silica gel) and nearly linear (H_2O on silica gel) thus unlikely to exhibit non-physical behavior (Yang and LeVan, 2009).

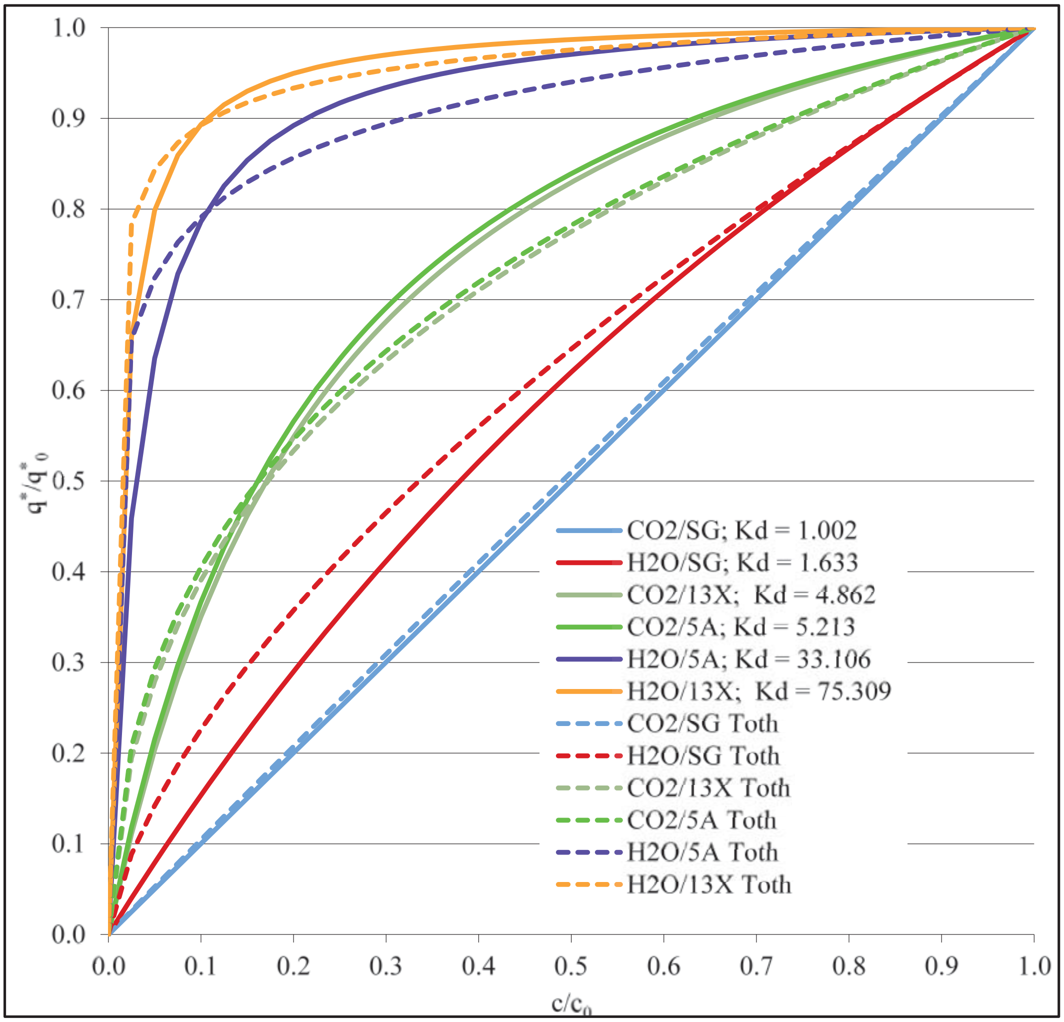


Figure 5.12 Normalized concentration vs. normalized bed loading for six sorbate/sorbent systems for conditions of 10°C and 1.0 kPa. Solid lines: Langmuir isotherms; Dashed lines: Toth isotherms. Value of distribution factor K_d provided in legend.

The distribution values for the Langmuir isotherms shown in Figure 5.1 are also shown in Table 5.2 along with the threshold values determined previously. As anticipated, there appears to be a strong correlation between θ and $K_{d,}$ thus relating the steepness of the isotherm and the onset of non-physical behavior.

Given a general relationship relating θ and K_d, the limiting values of the axial dispersion coefficient D_L and the LDF mass transfer coefficient k_n to avoid non-physical behavior in a simulation could be determined *a priori*. The software package TableCurve™ 2D was used to determine the mathematical form of a simple equation that would fit the data in Table 5.2. The resulting relationship shown Equation (5.6) was found to fit the data reasonably well, as shown in Figure 5.13.

Table 5.2 Threshold parameter values and distribution factor values.

Sorbate/Sorbent System	θ	K_d (fitted)
CO_2 on zeolite 5A	10241	5.213
CO_2 on zeolite 13X	11713	4.862
H_2O on zeolite 5A	522353	33.106
H_2O on zeolite 13X	679013	75.309

$$\ln(\theta) = d + \frac{e}{K_d} , \qquad 5.6$$

where d is a fitting parameter = 13.682 and e is a second fitting parameter = –18.020.

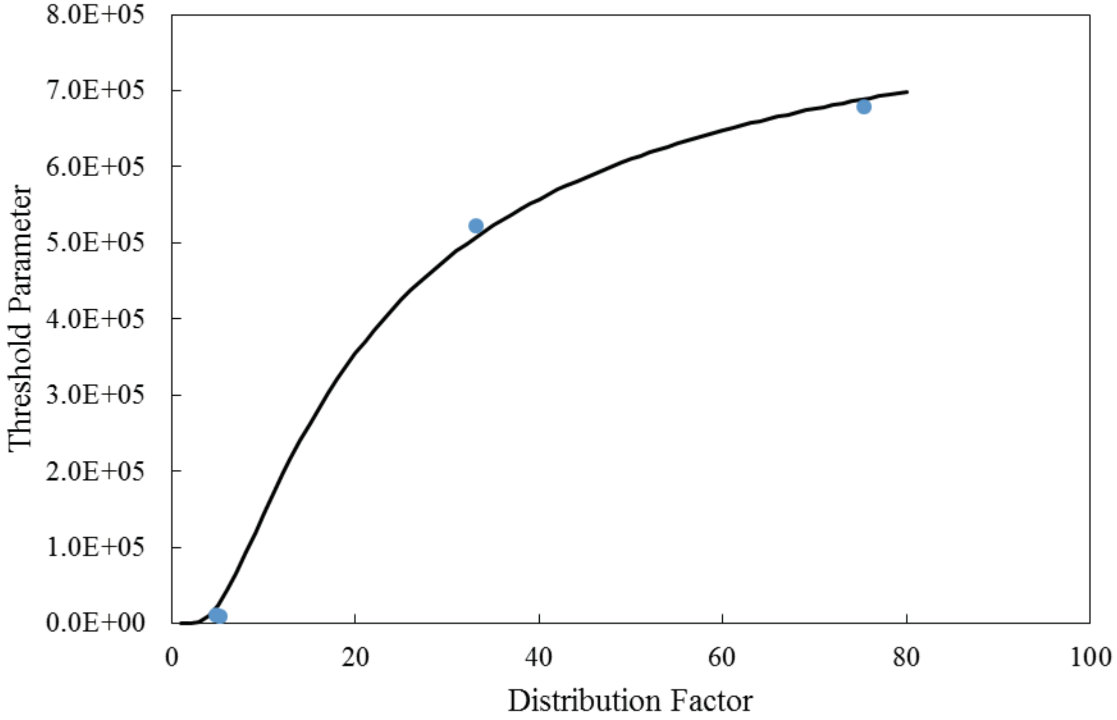


Figure 5.13 Estimated threshold value θ vs. distribution factor K_d for four sorbate/sorbent systems (filled circles) and fitted relationship shown in Equation 5.6 (line). Coefficient of determination (R^2) is 0.997.

5.6 Conclusions for Parameter Mapping

For applications with small diameter fixed sorbent beds, it is important to accurately capture the negative effects of wall channeling on adsorption efficiency. However, as also shown in Chapter 4, increasing the axial dispersion coefficient in the axially dispersed plug flow equation can result in a non-physical simulation.

In this chapter, multiple applications using small diameter fixed sorbent beds were presented. In order to capture the threshold for transition to non-physical behavior for these and similar applications, numerous breakthrough simulations were conducted while varying the values of the LDF mass transfer coefficient k_n and the axial dispersion coefficient D_L widely. The ratio between the slope of the breakthrough curve and the slope of gas-phase concentration history (or slope ratio) was used as a limiting metric, as values greater than one indicate breakthrough sharpening. Fitting the values for k_n and D_L for simulation cases where the slope ratio is close to one resulted a threshold equation, Equation (5.2), with a single fitted parameter, the threshold parameter, θ. This equation may be used to limit k_n and D_L such that simulations using the axially dispersed plug flow equation do not generate non-physical results.

However, the threshold parameter is different for each sorbate/sorbent system, and so must be determined via extensive parametric simulations for each sorbate/sorbent system. The observation that the magnitude of the threshold parameter was closely related to the equilibrium capacity suggested the approach of obtaining the threshold parameter based on the shape of the equilibrium isotherm for each system. The distribution factor, or the inverse of the constant separation factor approximation characterizes the sorbent loading vs. sorbate pressure relationship using a single parameter for a specific inlet concentration and temperature. A relationship was found between the distribution factor K_d and the threshold parameter θ, providing a means to predict the distribution factor using already available experimental data on a sorbate/sorbent pair. The estimated distribution factor may then be used to govern the magnitude of the LDF mass transfer coefficient k_n and the axial dispersion coefficient D_L in the axially dispersed plug flow equation such that non-physical simulation results are avoided.

CHAPTER VI

6. CONCLUSIONS

The separation of gases through adsorption plays an important role in the chemical processing industry, where the separation step is often the costliest part of a chemical process and thus worthy of careful study and optimization. It was established in Chapter 1 that CPI applications, where separation via adsorption is advantageous over other methods, are both numerous and of central importance. Other current applications include atmospheric control in habitable volumes, including submarines, spacecraft, and military and mine shelters. The primary adsorbents in current use are carbons, silica gels, and zeolites.

Emerging applications of gas separations via adsorption including new CPI applications have been made possible by recently developed adsorbents such as LiLSX zeolite and carbon nanotubes. In addition, plans for long-term missions to Mars, combined with requirements for lower CO_2 levels, have spurred a renewed effort in the development of spacecraft atmospheric control systems. Finally, a relatively new field that is receiving a great deal of attention is the capture of CO_2 from coal-fired power generation plants. The development of a wide range of processes incorporating liquid and solid amines and membranes in addition to adsorbents are under investigation by many researchers. Yet considerable work remains to achieve an economically viable solution due to the scale of CO_2 emissions.

The work documented herein developed a number of insights on the computer simulations used for the refinement and design of these gas adsorption processes. The axially dispersed plug flow equation is the primary focus as it is the predominate equation in use based on recent journal articles on fixed-bed adsorption models and breakthrough analysis. This work (1) presented a new approach to fit the undetermined mass transfer coefficients in the axially dispersed plug flow equation, (2) examined and described the conditions where non-physical simulation results can arise, and (3) presented an approach to determine the limits of the axial dispersion and LDF mass transfer term above which non-physical simulation results occur. The new archival information developed in this work in these three areas is summarized below.

Breakthrough analysis is favored for the determination of undetermined coefficients in the axially dispersed plug flow equation. Small diameter beds are almost universally used to reduce both adsorbent quantity and test stand costs. However, the influence of wall channeling is shown in this work to be significant even for a bed to a sorbent diameter ratio of 20, which is greater than for most of the breakthrough analyses reviewed. In order to evaluate the mass transfer coefficients independent of wall channeling effects, a centerline concentration measurement is taken just inside the exit of the column. This measurement allows for the empirical determination of the pellet-based axial dispersion and the lumped mass transfer coefficient. For cases where use of much larger beds for the actual process is the intent, these coefficients alone will provide the required kinetic data, as the channeling effects for much larger beds is negligible.

Two sorbent/sorbate systems as simulated as a proof of concept for the centerline measurement approach. Comparisons of experimental data and simulation results for

CO_2 and H_2O on zeolite 5A were shown in Figure 4.9, where good agreement is evident for both systems for concentrations and temperatures after fitting only the LDF mass transfer coefficient (a single heat transfer coefficient had been fit against a purely thermal experiment as shown in Figure 4.5). The LDF mass transfer coefficient obtained by this method may be used directly for simulation studies in large beds. This approach eliminates the confounding effects of channeling present for the standard breakthrough analysis approach, that is, the use of a small diameter bed and of a breakthrough curve based on a concentration taken far downstream.

In systems where the bed to sorbent diameter ratio is on the order of 20, channeling effects have to be taken into account. For the axially dispersed plug flow equation, the axial dispersion coefficient alone provides a means to simulate channeling. This work has shown that this approach is viable for some systems, such as CO_2 on zeolite 5A. A concentration measurement far downstream (such that the gas flowing along the column wall is fully mixed with the flow from the bed core) is used to empirically derive an effective column-dependent axial dispersion coefficient. The mass transfer coefficient fitted to the centerline measurement is used without change. The resulting comparison between simulation and experiment, shown in Figure 4.10, is excellent for both concentration and temperature. However, fitting the downstream breakthrough curve required an axial dispersion coefficient seven times higher than that specified by the Wakao and Funazkri axial dispersion correlation one of the six correlations examined gave the highest values.

This approach was also shown to be successful for the CO_2 on the 5A system. However, for the H_2O on the 5A system, the fitting process revealed a non-physical result, breakthrough sharpening, with the axially dispersed plug flow equation and a high axial dispersion coefficient (though within the bounds of standard correlations). When the D_L term is increased sufficiently for the simulation to approach the dispersive test results, the internal concentration history of the simulation degrades completely from the expected constant pattern profile.

To develop a mapping of the non-physical results as a function of the D_L and k_n magnitudes, multiple breakthrough simulations were performed to develop a 10 x 10 matrix for each sorbate/sorbent system. The onset of non-physical behavior was based on the ratio of the maximum slope of the breakthrough curve, or maximum derivative of concentration with respect to time, to the maximum slope of the internal concentration history. The maximum slope ratio provides a means to map the onset of non-physical behavior based on the magnitudes of D_L and k_n. The parameter in the threshold equation was termed the threshold parameter, and is unique for each sorbate/sorbent pair.

To obtain a limit on the magnitude of D_L and k_n for any sorbate/sorbent system where the equilibrium isotherm is available required developing a relationship between the curvature of the isotherm and the threshold parameter. Recasting the Toth isotherms for the sorbate/sorbent systems used in this study into Langmuir isotherms enabled characterization of the curvature via the distribution factor. Finally, a simple equation was found to fit the relationship between the threshold parameter and the distribution factor. The methodology established herein may be used to guide parametric studies employing the 1-D axially dispersed plug flow equation such that non-physical breakthrough simulations are avoided that could result in incorrect and non-predictive determination of the linear driving force mass transfer coefficient.

APPENDICES

APPENDIX A

LITERATURE REVIEW OF FIXED BED GAS ADSORPTION MODELS UNCERTAINTY ANAYLYSIS

Table A.1 Literature review of fixed bed gas adsorption models.

					Mass Transfer				
#	*Application*	*Experimental System*	*Spatial Dimensions*	*Tube ID/Particle Diameter*	*Gas to Particle Rate Expression*	*Method to Determine Gas to Particle Rate*	*Axial Dispersion*	*Internal Profile Shown?*	*Reference*
1	Post-combustion CO_2 capture, steam methane reforming	CO_2 on hydrotalsite (dry and wet)	1	14	Modified LDF: function of loading and isotherm (pore diffusion model)	Breakthrough analysis (adsorb and desorb)	Estimated per Edwards and Richardson 1968	no	Ding and Alpay 2000
2	Hydrogen purification	H2, CO_2, CH4, CO on 5A, Activated Carbon	1	8 and 15	LDF	Breakthrough analysis (Park et al. 1998)	none	no	Park et al., 2000
3	CO_2 scrubbing	H_2O and CO_2 on 5A	1	20	LDF	Breakthrough analysis	Estimated per Edwards and Richardson 1968	yes	Mohamadinejad et al., 2000
4	Dehumidification of organic solvents	H_2O and Acetone on 4A	2	40	Bi-disperse pellet modeled	Not specified	Breakthrough Analysis	no	Pentchev and Seikova, 2002
5	Dehumidification of organic solvents	H2O on Silica gel, 13X	1	10	LDF	Uptake curve analysis (Yang and Lee, 1998)	Estimated per Wakao and Funazkri, 1978	no	Ahn and Lee, 2003
6	CO_2 scrubbing	CO_2 and H_2O on 5A	2	20	LDF	Breakthrough analysis	Estimated per Edwards and Richardson 1968	Yes (CO_2 only)	Mohamadinejad et al., 2003
7	Drying of instrument air	H_2O on 4A	1	10	Modified LDF as a function of pressure, temperature and molar fraction	Breakthrough analysis	none	no	Gorbach et al., 2004
8	Dehumidification of organic solvents	H2O on Silica gel, Alumina, 13X	1	10	LDF	Breakthrough analysis (Ahn and Lee, 2003)	Estimated per Wakao and Funazkri, 1978	no	Ahn and Lee, 2004
9	Oxygen purity	O2, Ar, and N2 on CMS	1	n/a	Modified LDF (concentration-dependent diffusivity combined with Langmuir isotherm)	Estimated per Bae and Lee (2005)	yes	no	Jee et al., 2005

Table A.1 Literature review of fixed bed gas adsorption models (continued).

					Mass Transfer				
#	*Industrial Application*	*Experimental System*	*Spatial Dimensions*	*Tube ID/Particle Diameter*	*Gas to Particle Rate Expression*	*Method to Determine Gas to Particle Rate*	*Axial Dispersion*	*Internal Profile Shown?*	*Reference*
10	Post-combustion CO_2 capture	CO_2 and N2 on silicalite	1	11	LDF	Estimated per Farooq and Ruthven 1990	Estimated per Wakao et al., 1978	no	Delgado et al., 2006
11	Gas Separation	CH4 and CO_2 on 5A, butane and pentane on silicalite	1	8	Double LDF	Estimated per method reported in this paper, adjusted via breakthrough analysis	Included but not defined	no	Leinekugel-le-Cocq et al., 2007
12	Hydrogen purification	CO on 5A (4 samples) and AC in H2	1	7 and 11	LDF	Breakthrough analysis	Estimated per Lopes et al., 2009	no	Bastos-Neto et al., 2011
13	Post-combustion CO_2 capture	CO_2 and N2 on 13X	1	6	Combined LDF and QDF	Breakthrough analysis	None	No	Won et al., 2012
14	Capture of hydrocarbon emissions from gasoline engines during the cold start period of the engine	Propane on Na-ZSM-5	1	256	LDF	Estimated from film and micropore calculations per Brosillon et al. 2001	Yes (not specified)	Yes	Puertolas et al., 2012
15	Post-combustion CO_2 capture	CO_2 on 13X	1	21	Macropore and micropore solved simultaneously	Estimated + breakthrough analysis	Estimated per Wakao and Funazkri, 1978	Yes	Mulgundmath et al., 2012
16	Post-combustion CO_2 capture	CO_2 and N2 on carbon	1	9, 3	LDF	Breakthrough analysis and estimated via Ruthven 1984	None	No	Gonzlez et a., 2013
17	Development process of a reactor design for open thermochemical energy storage	H_2O on 13X	2	25	LDF	Breakthrough analysis	Yes	Temperature only	Mette et al., 2014
18	Drying of ethanol for fuel production	Ethanol and H_2O on 3A	1	6	LDF	Breakthrough analysis	None	Yes	Kupiec et al., 2014
19	Dehumidification of gases containing organic component	H_2O and benzene on Silica gel, 13X, activated carbon	1	n/a	LDF	Breakthrough analysis (Ko et al., 2002, Park and Knaebel 1992, and Ahn and Lee 2003)	Estimated per Edwards and Richardson 1968	No	Nastaj and Ambrozek, 2015

APPENDIX B

UNCERTAINTY ANALYSIS

Evaluation of Gas Chromatograph Calibration for Carbon Dioxide

The objective of this analysis is to determine the uncertainty in the calibration used to convert the gas chromatograph (GC) results, expressed as the area under the curve of the chromatogram for the gas of interest, to carbon dioxide concentration. Generally a one or two point calibration of the GC is performed, where a gas standard with a known percentage of the gas of interest is injected into the GC and the resultant area measured. Since the area corresponding to a gas with zero concentration will be zero, the ratio of the concentration to area is the slope of line through the origin. This slope or ratio may then be multipled times the GC area obtained during breakthrough testing to obtain a gas percentage.

However, during the GC calibrations for the CO2 adsorption testing, variations in the GC area with total pressure were observed for the same calibration gas. To adjust the GC results, a correlation is developed to correct for total gas pressure at the port location of interest. As a result, the conversion from the GC measurement (area) to partial pressure of CO2 requires the following steps:

1. For each port pressure, find the value of GC area that corresponds to calibration gas percent CO2
2. Convert GC area to CO2 percentage
3. Convert from CO2 percentage to CO2 concentration

For step 1, a linear correlation was developed based on GC area and pressure measurements. Step 2 uses a single point calibration based on step 1 and the local port pressure. The data reduction equation for step 3 requires the total pressure and temperature. The Monte Carlo simulation approach is used to determine the uncertainty for each step of this analysis.

(1) For each port pressure, find the value of GC area that corresponds to calibration gas percent CO2: Determine the uncertainty in correlation used to adjust GC area per total pressure via Monte Carlo simulation with $N := 10000$

The data reduction equations for this step are shown below. Uncertainties for each of the variables are developed in the following pages.

$$m = \frac{GC_2 - GC_1}{P_2 - P_1} \qquad\qquad GC_{cal_j} = m \cdot \left(P_{cal_j} - P_1\right) + GC_1$$

The GC area data is shown below for each of the CO2 calibration points. The GC repeatibility for this limited set of data is calculated using the t value for a sample number of 2.

$$GC_{1true_1} := 12036 \qquad GC_{2true_1} := 12309 \qquad\qquad n := 2$$

$$GC_{1true_2} := 12048 \qquad GC_{2true_2} := 12335 \qquad\qquad t := 12.706$$

$$GC_{mean_1} := \text{mean}\left(GC_{1true_1}, GC_{1true_2}\right) = 1.2042 \cdot 10^4$$

$$S_1 := \sqrt{\left(\frac{1}{n-1} \cdot \sum_{i=1}^{n} \left(GC_{1true_i} - GC_{mean_1}\right)^2\right)} = 8.4853 \qquad P_{GC1} := t \cdot S_1 = 107.814$$

$$GC_{mean_2} := \text{mean}\left(GC_{2true_1}, GC_{2true_2}\right) = 1.2322 \cdot 10^4$$

$$S_2 := \sqrt{\left(\frac{1}{n-1} \cdot \sum_{i=1}^{n} \left(GC_{2true_i} - GC_{mean_2}\right)^2\right)} = 18.3848 \quad P_{GC2} := t \cdot S_2 = 233.597$$

$$\frac{P_{GC1}}{GC_{mean_1}} = 0.8953 \ 1\% \qquad\qquad \frac{P_{GC2}}{GC_{mean_2}} = 1.8958 \ 1\%$$

$$P_{GCP} := \frac{1}{2} \cdot \left(\frac{P_{GC1}}{GC_{mean_1}} + \frac{P_{GC2}}{GC_{mean_2}}\right) = 1.3955 \ 1\%$$

To consider whether this magnitude of random uncertainty is appropriate or inflated due to the inherent uncertainty in a small sample, a literature search on the uncertainty of other gas chromagraphs using temperature conductivity detectors (TCD) was conducted. In the natural gas industry, carbon dixoide in the raw gas must be removed. This is one of many instances of CO2 removal using zeolites (Breck, 1974).

Zuas (2015) reported that the percent relative standard deviation for CO2 as detected by a GC with TCD over a wide range of GC flow rates varied from 0.10 to 0.40%. Sutan (2013) found that combined relative uncertainty for CO2 analyzed with a GC and TCD to be 0.3594%. Based on these references, a relative precision uncertainty of 0.4% is applied in this analysis. The Mathcad function Stdev is verified to correctly calculate the square root of the sample variance in the following equations.

<u>Assembly of Variable Distributions for each GC area measurement:</u>

$$j := 1..2 \qquad i := 1..N \qquad P_{GCP} := 0.4\%$$

$$GC_{i,j} := GC_{mean_j} + GC_{mean_j} \cdot \text{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$X := GC^{\langle 1\rangle} \qquad S_I := \sqrt{\left(\frac{1}{N-1} \cdot \sum_{i=1}^{N} \left(X_i - \text{mean}(X)\right)^2\right)} = 23.9604$$

$$S_{GC_j} := \text{Stdev}\left(GC^{\langle j\rangle}\right) = \begin{bmatrix} 23.9604 \\ 24.5857 \end{bmatrix} \qquad U_{GCper_j} := \frac{2 \cdot S_{GC_j}}{\text{mean}\left(GC^{\langle j\rangle}\right)} = \begin{bmatrix} 0.3979 \\ 0.3991 \end{bmatrix} 1\%$$

Pressure transducer uncertainty is provided as a percent of full scale (Viatran 2010), which is assumed to be 30 psia. The bias error distribution is calculated prior to the calculating the error distribution for pressure as this is a correlated systematic uncertainty, and the same distribution is used in the calculation of error distribution for the calibration pressures. The manufacturer specifications are considered to be maximum values (3 standard deviations) per Nakos (2004). Errors are normalized to 2 standard deviations based on the approach suggested by Lipak (1995).

$$P_{true} := \begin{bmatrix} 14.9 \\ 15.3 \end{bmatrix} \cdot \mathbf{\mathit{psi}} \qquad B_P := \frac{2}{3} \cdot 0.25\% \qquad P_P := \frac{2}{3} \cdot 0.1\% \qquad P_{FS} := 30 \cdot \mathbf{\mathit{psi}}$$

Distribution for B_P(percent of full scale):

$\mu_P := 0$; $\sigma_P := 1$; let $\beta_P := \text{rnorm}\left(N, \mu_P, \sigma_P\right)$

$$P_{i,j} := P_{true_j} + P_{FS} \cdot \frac{B_P}{2} \cdot \beta_{P_i} + P_{true_j} \cdot \text{rnorm}\left(N, 0, \frac{P_P}{2}\right)_i \qquad \text{mean}\left(P^{\langle j\rangle}\right) = \begin{bmatrix} 14.9 \\ 15.3 \end{bmatrix} \mathbf{\mathit{psi}}$$

$$S_{P_j} := \text{Stdev}\left(P^{\langle j\rangle}\right) = \begin{bmatrix}0.0258\\0.0259\end{bmatrix} \, \boldsymbol{psi} \qquad U_{Pper_j} := \frac{2 \cdot S_{P_j}}{\text{mean}\left(P^{\langle j\rangle}\right)} = \begin{bmatrix}0.3463\\0.339\end{bmatrix} \, 1\%$$

In order to adjust GC area measurements that correspond to the calibration gas CO2 percentage for the total pressure, the slope will be calculated for each iteration of the Monte Carlo analysis. The uncertainty of this slope is then calculated.

$$m_i := \frac{GC_{i,2} - GC_{i,1}}{P_{i,2} - P_{i,1}} \qquad \text{mean}(m) = 699.8214 \, \frac{1}{\boldsymbol{psi}}$$

$$X := m \qquad S_X := \text{Stdev}(X) \qquad U_{Xpercent} := \frac{2 \cdot S_X}{\text{mean}(X)} = 24.7815 \, 1\%$$

The effect of subtraction is to the increase the percent uncertainty by a factor of 60. However, as this slope is applied to a correction factor that is small compared to the overall magnitude of the GC area, the uncertainty for the GC area will be much smaller.

Systematic uncertainty for the pressure transducers was specified above. The values for total pressure at each sample port and for the percent CO2 above are used to find the GC area corresponding to the gas standard percent CO2.

$$j := 1 \,..\, 4 \qquad P_{true} := \begin{bmatrix}15.348\\15.313\\15.265\\15.265\end{bmatrix} \cdot \boldsymbol{psi}$$

$$P_{cal_{i,j}} := P_{true_j} + P_{FS} \cdot \frac{B_P}{2} \cdot \beta_{P_i} + P_{true_j} \cdot \text{rnorm}\left(N, 0, \frac{P_P}{2}\right)_i$$

The data reduction equation is used to find the GC area that corresponds to the gas standard percent CO2 is used below.

$$GC_{cal_{i,j}} := m_i \cdot \left(P_{cal_{i,j}} - P_{i,1}\right) + GC_{i,1} \qquad \text{mean}\left(GC_{cal}^{\langle j\rangle}\right) = \begin{bmatrix}12356\\12331\\12298\\12298\end{bmatrix}$$

$$S_{GCcal_j} := \text{Stdev}\left(GC_{cal}^{\langle j\rangle}\right) = \begin{bmatrix}28.2492\\25.9384\\23.063\\23.0877\end{bmatrix} \qquad U_{GCcalper_j} := \frac{2 \cdot S_{GCcal_j}}{\text{mean}\left(GC_{cal}^{\langle j\rangle}\right)} = \begin{bmatrix}0.4573\\0.4207\\0.3751\\0.3755\end{bmatrix} \, 1\%$$

(2) Convert GC area to CO2 percentage: Determine the uncertainty in one-point GC calibration via Monte Carlo simulation. The data reduction equation for this step is:

$$P_{CO2} = f_{cal} \cdot GC \qquad \text{where} \qquad f_{cal} = \frac{GS}{GC_{cal}} \cdot P_{cal}$$

The uncertainty associated with each of the variables, except the gas standard (*GS*), has already been determined. For the bias error associated with Matheson gas standards, a 95% confidence interval is specified by Geib (2005). The gas standard used for testing of this type is in the authors experience a high grade standard. The higher grades for both Matheson and Airgas have an overall uncertainty of 1% in this carbon dioxide partial pressure range. (Matheson, 2015, Airgas 2007)

Calibration Gas Standard (carbon dioxide in nitrogen): Overall Uncertainty:

$$GS_{true} := 0.39992\% \qquad\qquad B_{GS} := 1\%$$

Assembly of variable distribution for gas standard for each carbon dioxide calibration measurement:

$$GS := GS_{true} + GS_{true} \cdot \text{rnorm}\left(N, 0, \frac{B_{GS}}{2}\right)$$

$$X := GS \qquad S_X := \text{Stdev}(X) \qquad U_{Xpercent} := \frac{2 \cdot S_X}{\text{mean}(X)} = 1.0071\ \ 1\%$$

Test data was taken at four locations, each having a different total pressure, thus a different value to convert GC area to CO2 partial pressure is required for each location. This factor, and its uncertainty, is determined below.

$$f_{cal_{i,j}} := \frac{GS_i}{GC_{cal_{i,j}}} \cdot P_{cal_{i,j}} \qquad \text{mean}\left(f_{cal}^{\langle j \rangle}\right) = \begin{bmatrix} 4.968 \cdot 10^{-6} \\ 4.966 \cdot 10^{-6} \\ 4.964 \cdot 10^{-6} \\ 4.964 \cdot 10^{-6} \end{bmatrix} \textbf{\textit{psi}}$$

$$S_{fcal_j} := \text{Stdev}\left(f_{cal}^{\langle j \rangle}\right) = \begin{bmatrix} 2.8498 \cdot 10^{-8} \\ 2.8148 \cdot 10^{-8} \\ 2.7737 \cdot 10^{-8} \\ 2.7743 \cdot 10^{-8} \end{bmatrix} \textbf{\textit{psi}} \qquad U_{fcalper_j} := \frac{2 \cdot S_{fcal_j}}{\text{mean}\left(f_{cal}^{\langle j \rangle}\right)} = \begin{bmatrix} 1.1474 \\ 1.1336 \\ 1.1175 \\ 1.1178 \end{bmatrix} 1\%$$

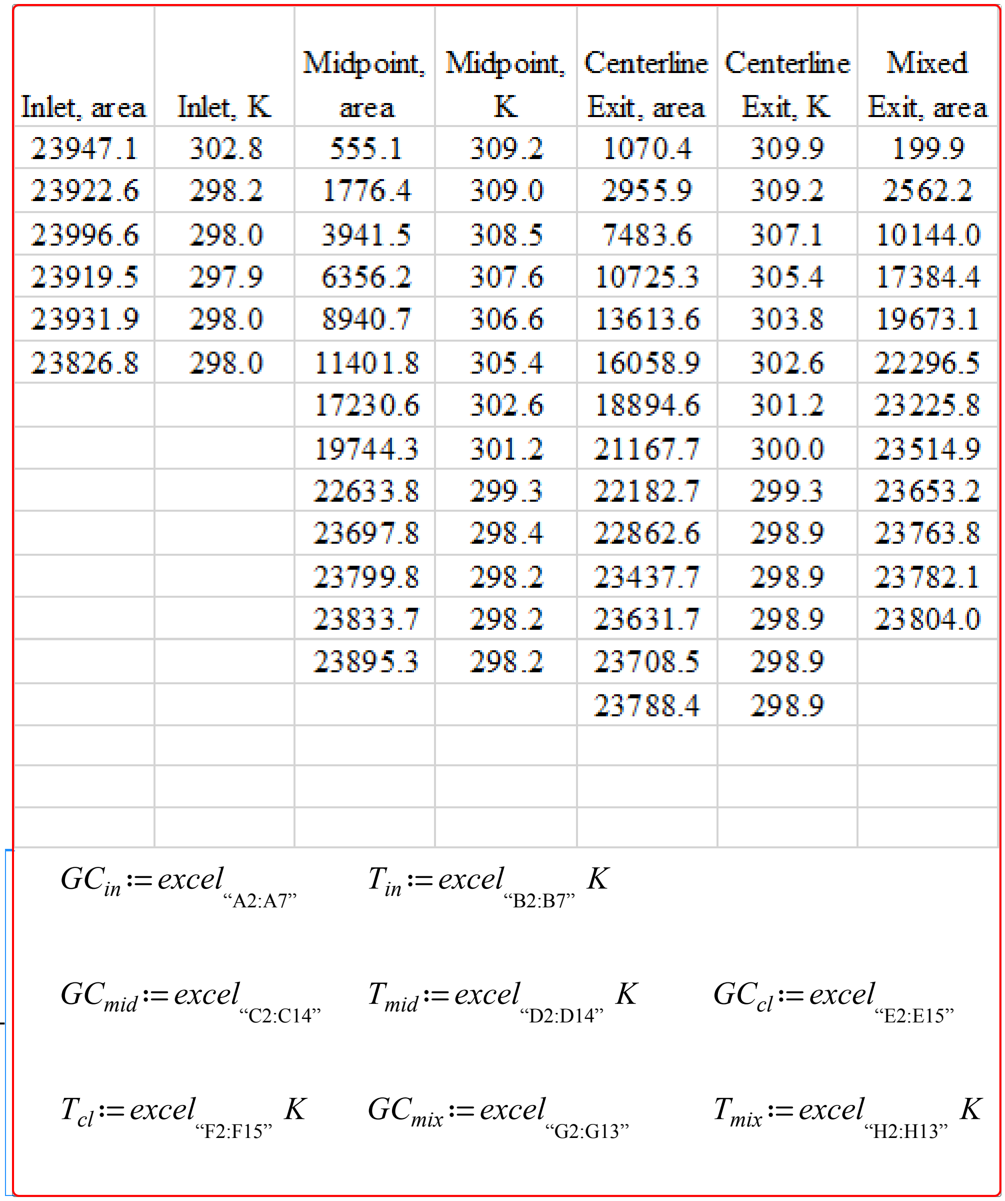
The remaining "true" variables, GC area and temperature, in the data reduction equation are specific to each data point in the CO2 breakthrough test. Since the GC area error is random (and not known to be correlated), the errors are generated from the Gaussian random number generator for each data point and scaled with the same standard deviations specified for the GC calibration.

The ideal gas law will be used for finding the CO2 concentration. All uncertainties except for temperature have already been determined. For the uncertainty associated with the measured temperature values, the Omega value of +/- 1 degree C is used (Omega, 2015). Based on a thermocouple error analysis shown later, a more reasonable overall uncertainty is 0.22 C, including data aquisition system errors. To account for installation errors, the values from that analysis are doubled as shown below.

$$c = \frac{f_{cal} \cdot GC}{\boldsymbol{R} \cdot T} \qquad B_T := 0.23 \cdot \boldsymbol{K} \qquad P_T := 0.36 \cdot \boldsymbol{K}$$

Error evaluation for each of the 41 data points must be performed individually due to the variations in GC area and temperature. This data is entered in an Excel component below.

Table B.1 Gas chromatograph data for CO_2 breakthrough test.



Inlet, area	Inlet, K	Midpoint, area	Midpoint, K	Centerline Exit, area	Centerline Exit, K	Mixed Exit, area
23947.1	302.8	555.1	309.2	1070.4	309.9	199.9
23922.6	298.2	1776.4	309.0	2955.9	309.2	2562.2
23996.6	298.0	3941.5	308.5	7483.6	307.1	10144.0
23919.5	297.9	6356.2	307.6	10725.3	305.4	17384.4
23931.9	298.0	8940.7	306.6	13613.6	303.8	19673.1
23826.8	298.0	11401.8	305.4	16058.9	302.6	22296.5
		17230.6	302.6	18894.6	301.2	23225.8
		19744.3	301.2	21167.7	300.0	23514.9
		22633.8	299.3	22182.7	299.3	23653.2
		23697.8	298.4	22862.6	298.9	23763.8
		23799.8	298.2	23437.7	298.9	23782.1
		23833.7	298.2	23631.7	298.9	23804.0
		23895.3	298.2	23708.5	298.9	
				23788.4	298.9	

Outputs

$GC_{in} := excel_{\text{“A2:A7”}}$ $\quad T_{in} := excel_{\text{“B2:B7”}}\ K$

$GC_{mid} := excel_{\text{“C2:C14”}}$ $\quad T_{mid} := excel_{\text{“D2:D14”}}\ K$ $\quad GC_{cl} := excel_{\text{“E2:E15”}}$

$T_{cl} := excel_{\text{“F2:F15”}}\ K$ $\quad GC_{mix} := excel_{\text{“G2:G13”}}$ $\quad T_{mix} := excel_{\text{“H2:H13”}}\ K$

Uncertainty analysis for inlet concentrations:

$i := 1 .. 10000$

$j := 1 .. \text{length}\left(T_{in}\right) \qquad \text{length}\left(T_{in}\right) = 6$

$$T_{i,j} := T_{in_j} + \text{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \text{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{in_j} + GC_{in_j} \cdot \mathrm{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{in_{i,j}} := \frac{\left(f_{cal}{}^{\langle 1 \rangle}\right)_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$$S_{cin_j} := \mathrm{Stdev}\left(c_{in}{}^{\langle j \rangle}\right) \qquad U_{cin_j} := 2 \cdot S_{cin_j} \qquad U_{cinper_j} := \frac{2 \cdot S_{cin_j}}{\mathrm{mean}\left(c_{in}{}^{\langle j \rangle}\right)} = \begin{bmatrix} 1.2209 \\ 1.2214 \\ 1.2207 \\ 1.2201 \\ 1.2279 \\ 1.2094 \end{bmatrix} 1\%$$

Uncertainty analysis for midpoint concentrations:

$$j := 1 \,..\, \mathrm{length}\left(T_{mid}\right) \qquad \mathrm{length}\left(T_{mid}\right) = 13$$

$$T_{i,j} := T_{mid_j} + \mathrm{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \mathrm{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{mid_j} + GC_{mid_j} \cdot \mathrm{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{mid_{i,j}} := \frac{\left(f_{cal}{}^{\langle 2 \rangle}\right)_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$$S_{cmid_j} := \mathrm{Stdev}\left(c_{mid}{}^{\langle j \rangle}\right) \quad U_{cmid_j} := 2 \cdot S_{cmid_j} \quad U_{cmidper_j} := \frac{2 \cdot S_{cmid_j}}{\mathrm{mean}\left(c_{mid}{}^{\langle j \rangle}\right)} = \begin{array}{c|c|} 0 & \\ \vdots & \vdots \\ 1 & 1.2083 \\ 2 & 1.2094 \\ 3 & 1.2105 \\ 4 & 1.207 \\ 5 & 1.2042 \\ 6 & 1.2091 \\ 7 & 1.2116 \\ 8 & 1.2097 \\ 9 & 1.2044 \\ 10 & 1.2096 \\ 11 & 1.2063 \\ 12 & 1.2049 \end{array} \; 1\%$$

Uncertainty analysis for centerline concentrations:

$$j := 1 \,..\, \mathrm{length}\left(T_{cl}\right) \qquad \mathrm{length}\left(T_{cl}\right) = 14$$

$$T_{i,j} := T_{cl_j} + \mathrm{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \mathrm{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{cl_j} + GC_{cl_j} \cdot \mathrm{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{cl_{i,j}} := \frac{\left(f_{cal}^{\langle 3\rangle}\right)_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$$S_{ccl_j} := \mathrm{Stdev}\left(c_{cl}^{\langle j\rangle}\right) \qquad U_{ccl_j} := 2 \cdot S_{ccl_j} \qquad U_{cclper_j} := \frac{2 \cdot S_{ccl_j}}{\mathrm{mean}\left(c_{cl}^{\langle j\rangle}\right)} = \begin{array}{r|l|} 0 & \\ \vdots & \vdots \\ 2 & 1.1926 \\ 3 & 1.1903 \\ 4 & 1.1902 \\ 5 & 1.1965 \\ 6 & 1.1916 \\ 7 & 1.1984 \\ 8 & 1.1871 \\ 9 & 1.1915 \\ 10 & 1.1959 \\ 11 & 1.1884 \\ 12 & 1.1905 \\ 13 & 1.1921 \end{array} \ 1\%$$

Uncertainty analysis for mixed exit concentrations:

$$j := 1 \,..\, \mathrm{length}\left(T_{mix}\right) \qquad \mathrm{length}\left(T_{mix}\right) = 12$$

$$T_{i,j} := T_{mix_j} + \mathrm{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \mathrm{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{mix_j} + GC_{mix_j} \cdot \mathrm{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{mix_{i,j}} := \frac{\left(f_{cal}^{\langle 4\rangle}\right)_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$$S_{cmix_j} := \mathrm{Stdev}\left(c_{mix}^{\langle j\rangle}\right) \quad U_{cmix_j} := 2 \cdot S_{cmix_j} \quad U_{cmixper_j} := \frac{2 \cdot S_{cmix_j}}{\mathrm{mean}\left(c_{mix}^{\langle j\rangle}\right)} = \begin{bmatrix} 1.1969 \\ 1.1892 \\ 1.1958 \\ 1.1919 \\ 1.1967 \\ 1.1951 \\ 1.1888 \\ 1.1915 \\ 1.193 \\ 1.1889 \\ 1.1949 \\ 1.1878 \end{bmatrix} 1\%$$

Plot histogram showing the uncertainty distribution for the concentration with the highest uncertainty. The blue line shows a normal distribution based on the mean and standard deviation. It is noted that the concentration uncertainty, shown by the red bars, follows a normal distribution.

$$X := \frac{c_{in}^{\langle 5\rangle}}{\frac{\boldsymbol{mol}}{\boldsymbol{m}^3}} \qquad N := \mathrm{rows}(X) \qquad X_{mean} := \mathrm{mean}(X) \qquad X_{mean} = 0.3309$$

$$S_X := \text{stdev}(X)\cdot\sqrt{\frac{N}{N-1}} \qquad S_X = 0.002$$

$$int := interval(X, S_X, N) \qquad h := int_2 - int_1 \qquad f := \text{hist}(int, X)$$

$$int := int + 0.5\cdot h \qquad F(x) := \text{dnorm}(x, X_{mean}, S_X)$$

$$U_X := 2\cdot S_X \qquad U_X = 0.0041 \qquad U_{Xpercent} := \frac{U_X}{X_{mean}} = 1.2279\ 1\%$$

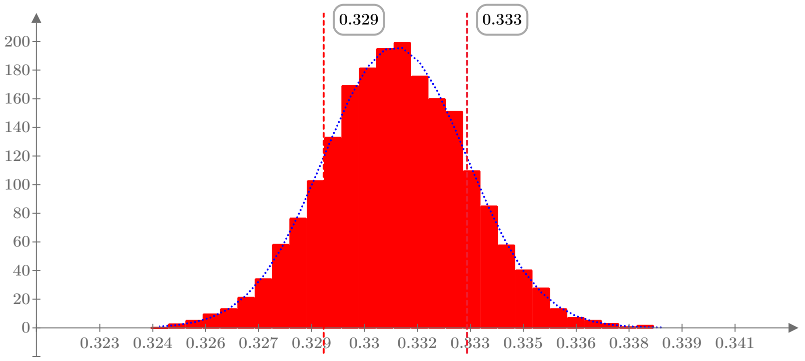
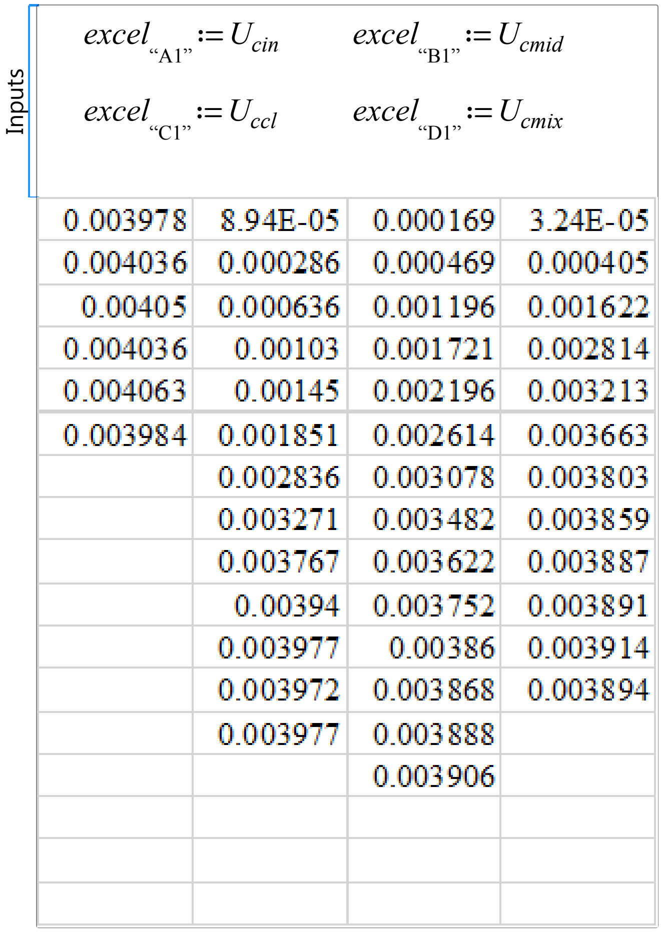


Figure B.1 Typical uncertainty distribution for a CO_2 concentration measurement.

Table B.2 Uncertainties for CO_2 concentration based on a 95% confidence interval.



Inputs

$excel_{\text{“A1”}} := U_{cin}$ $excel_{\text{“B1”}} := U_{cmid}$

$excel_{\text{“C1”}} := U_{ccl}$ $excel_{\text{“D1”}} := U_{cmix}$

0.003978	8.94E-05	0.000169	3.24E-05
0.004036	0.000286	0.000469	0.000405
0.00405	0.000636	0.001196	0.001622
0.004036	0.00103	0.001721	0.002814
0.004063	0.00145	0.002196	0.003213
0.003984	0.001851	0.002614	0.003663
	0.002836	0.003078	0.003803
	0.003271	0.003482	0.003859
	0.003767	0.003622	0.003887
	0.00394	0.003752	0.003891
	0.003977	0.00386	0.003914
	0.003972	0.003868	0.003894
	0.003977	0.003888	
		0.003906	

Evaluation of Gas Chromatograph Calibration for Water Vapor

The objective of this analysis is to determine the uncertainty in the calibration used to convert the gas chromatograph (GC) results, expressed as the area under the curve of the chromatogram for the gas of interest, to water vapor concentration. For the water vapor breakthrough testing, a three point calibration of the GC is performed. Here a gas with a known dew point is injected into the GC and the resultant area measured. Since the area corresponding to a gas with zero concentration will be zero, the slope of line through the origin provides the ratio of the concentration to GC area. This slope or ratio may then be multipled times the GC area obtained during breakthrough testing to obtain a water vapor pressure. A correlation to convert dew point to water vapor pressure required for this procedure.

The conversion from the GC measurement (area) to vapor pressure of H2O requires the following steps:

1. Convert dew point calibration measurements (as measured by the hygrometer) to vapor pressure
2. Determine the conversion factor from GC area to vapor pressure
3. Convert from H2O vapor pressure to CO2 concentration

For step 1, the Hardy formulation (Hardy 1998) is used. Step 2 uses a three point calibration to find a conversion factor. The data reduction equation for step 3 requires the vapor pressure and temperature. The Monte Carlo simulation approach is used to determine the uncertainty for each step of this analysis. Propagation analysis is compared with the Monte Carlo results as application to many data points is more direct.

(1) Conversion to vapor pressure is via the Hardy (1998) formulation, as the conversion used in the original data reduction was not provided. This conversion incurs less then 0.02% error in the worst incident compared to Table 3 in section 6.5 of the ASHRAE Handbook Fundamentals, 1997. Based on the 1/4 rule of thumb, where 1/4 is the ratio of the uncertainty that may be neglected to the largest uncertainty (Coleman 2009) this error is considered negligible. The correlation used to convert from dewpoint (DP) to vapor pressure (VP) is shown below.

$$VP = exp\left(k0 \cdot DP^{-1} + k1 + k2 \cdot DP + k3 \cdot DP^{2} + k4 \cdot DP^{3} + k5 \cdot ln\left(DP\right)\right)$$

The equation shown is for dew points greater than 0 C. For dew points lower than 0 C, the equation is similar, but has two additonal terms. The following dew points were used in the three-point calibration:

$N := 10000 \qquad i := 1..N \qquad j := 1..3$

$DP_{true_1} := -1.7\ \mathbf{°C} \qquad DP_{true_2} := 10.6\ \mathbf{°C} \qquad DP_{true_3} := 18.2\ \mathbf{°C}$

Dew point sensor uncertainty (Optica, 2008) is provided without specification of the associated confidence interval. The manufacturer specifications are considered to be maximum values (3 standard deviations) per Nakos (2004). Errors are normalized to 2 standard deviations based on the approach suggested by Lipak (1995).

$B_{DP} := \frac{2}{3} \cdot 0.2\ \mathbf{K} \qquad P_{DP} := \frac{2}{3} \cdot 0.05\ \mathbf{K}$

The bias errors used for the three calibration measurements will be correlated. For the Monte Carlo analysis, this means a common distribution for the error should be used in the calculation of the measurement distribution, so it is determined prior to the measurement distributions. The random (precision) errors will not be correlated and so are calculated individually for each measurement.

Distribution for B_{DP}(absolute uncertainty):

$\mu_{DP} := 0$; $\sigma_{DP} := \frac{B_{DP}}{2}$; let $\beta_{DP} := \text{rnorm}\left(N, \mu_{DP}, \sigma_{DP}\right)\ \mathbf{K}$

$$DP_{i,j} := DP_{true_j} + \beta_{DP_i} + \text{rnorm}\left(N, 0, \frac{P_{DP}}{2}\right)_i\ \mathbf{K} \qquad \text{mean}\left(DP^{\langle j\rangle}\right) = \begin{bmatrix} 271.5 \\ 283.8 \\ 291.4 \end{bmatrix} \mathbf{K}$$

$$S_{DP_j} := \text{Stdev}\left(DP^{\langle j\rangle}\right) = \begin{bmatrix} 0.0687 \\ 0.0688 \\ 0.0687 \end{bmatrix} \mathbf{K} \qquad U_{DPper_j} := \frac{2 \cdot S_{DP_j}}{\text{mean}\left(DP^{\langle j\rangle}\right)} = \begin{bmatrix} 0.0506 \\ 0.0485 \\ 0.0472 \end{bmatrix} 1\%$$

The dewpoints for each instance are converted to dewpoint using the Hardy (1998) formulation; dew point (DP) is in Kelvin and vapor pressure (VP) is in Pa. Since the function uses a log function, the units must be stripped off and then reapplied.

$$VP_{i,j} := DPtoVP\left(\frac{DP_{i,j}}{\mathbf{K}}\right) \mathbf{Pa} \qquad \text{mean}\left(VP^{\langle j\rangle}\right) = \begin{bmatrix} 530.9 \\ 1278.4 \\ 2090.8 \end{bmatrix} \mathbf{Pa}$$

GC area data for each of the dew point calibration points is provided in the test data sheet (Mohamadinejad 1995). As this is a calibration, the bias error is accounted for in the dew point sensor error. However the GC repeatibility should be considered. For this limited set of data, repeatibility is calculated using the t value for N = 2 (Coleman 2009).

$GC_{1true_1} := 9654 \qquad GC_{2true_1} := 22579 \qquad GC_{3true_1} := 36444 \qquad n := 2$

$GC_{1true_2} := 9667 \qquad GC_{2true_2} := 22596 \qquad GC_{3true_2} := 36343 \qquad t := 12.706$

$$GC_{mean_1} := \text{mean}\left(GC_{1true_1}, GC_{1true_2}\right) = 9660.5$$

$$S_1 := \sqrt{\left(\frac{1}{n-1} \cdot \sum_{i=1}^{n} \left(GC_{1true_i} - GC_{mean_1}\right)^2\right)} = 9.1924 \qquad P_{GC1} := t \cdot S_1 = 116.7985$$

$$GC_{mean_2} := \text{mean}\left(GC_{2true_1}, GC_{2true_2}\right) = 2.2588 \cdot 10^4 \qquad \frac{P_{GC1}}{GC_{mean_1}} = 1.209 \;\; 1\%$$

$$S_2 := \sqrt{\left(\frac{1}{n-1} \cdot \sum_{i=1}^{n} \left(GC_{2true_i} - GC_{mean_2}\right)^2\right)} = 12.0208 \qquad P_{GC2} := t \cdot S_2 = 152.7365$$

$$GC_{mean_3} := \text{mean}\left(GC_{3true_1}, GC_{3true_2}\right) = 3.6394 \cdot 10^4 \qquad \frac{P_{GC2}}{GC_{mean_2}} = 0.6762 \;\; 1\%$$

$$S_3 := \sqrt{\left(\frac{1}{n-1} \cdot \sum_{i=1}^{n} \left(GC_{3true_i} - GC_{mean_3}\right)^2\right)} = 71.4178 \qquad P_{GC3} := t \cdot S_3 = 907.4344$$

$$\frac{P_{GC3}}{GC_{mean_3}} = 2.4934 \;\; 1\%$$

To consider whether this magnitude of random uncertainty is appropriate or inflated due to the inherent uncertainty in a small sample, a literature search on the uncertainty of other gas chromagraphs using temperature conductivity detectors (TCD) was conducted. Zuas (2015) reported that the percent relative standard deviation for CO2 as detected by a GC with TCD over a wide range of GC flow rates varied from 0.10 to 0.40%. Sutan (2013) found that combined relative uncertainty for CO2 analyzed with a GC and TCD to be 0.3594%. However no similar references were located providing typical values for H2O analysis uncertainty.
A second approach follows the consistency of the calibration data and the magnitude of the GC area in the actual test. The maximum GC area is about 14000 during the test, putting the highest calibration point far out of range. Since single point calibrations are commonly used, the middle point will be selected for this calibration as it has the highest precision and thus confidence and will allow for interpolation rather than extrapolation.

Assembly of Variable Distributions for each GC area measurement:

$$j := 2..2 \qquad i := 1..N \qquad P_{GCP} := \frac{P_{GC2}}{GC_{mean_2}} = 0.6762\ 1\%$$

$$GC_{i,j} := GC_{mean_j} + GC_{mean_j} \cdot \mathrm{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i \qquad \mathrm{mean}\left(GC^{\langle j\rangle}\right) = [22588]$$

$$S_{GC_j} := \mathrm{Stdev}\left(GC^{\langle j\rangle}\right) = [76.0793] \qquad U_{GCper_j} := \frac{2 \cdot S_{GC_j}}{\mathrm{mean}\left(GC^{\langle j\rangle}\right)} = [0.6736]\ 1\%$$

(2) Determine the conversion factor from GC area to vapor pressure: In order to convert GC area measurements taken during the test to concentration values, the ratio of the calibration vapor pressure vs. GC area is calculated. The intercept is at the origin as appropriate for the relationship between GC area and partial pressure. For the error analysis, the slope will be calculated for each iteration of the Monte Carlo analysis.

$$m_i := \frac{VP_{i,2}}{GC_{i,2}} \qquad m_{mean} := \mathrm{mean}(m) = 0.0566\ \boldsymbol{Pa} \qquad m_{min} := min(m) = 0.0554\ \boldsymbol{Pa}$$

$$m_{max} := \max(m) = 0.0579\ \boldsymbol{Pa}$$

Verify equations by plotting vapor pressure points, GC area, and max/min fit lines. Vapor pressure is expressed in Pascals.

$$k := 1, 10..37000 \qquad XI_k := k \qquad YI_{max_k} := m_{max} \cdot XI_k \qquad YI_{min_k} := m_{min} \cdot XI_k$$

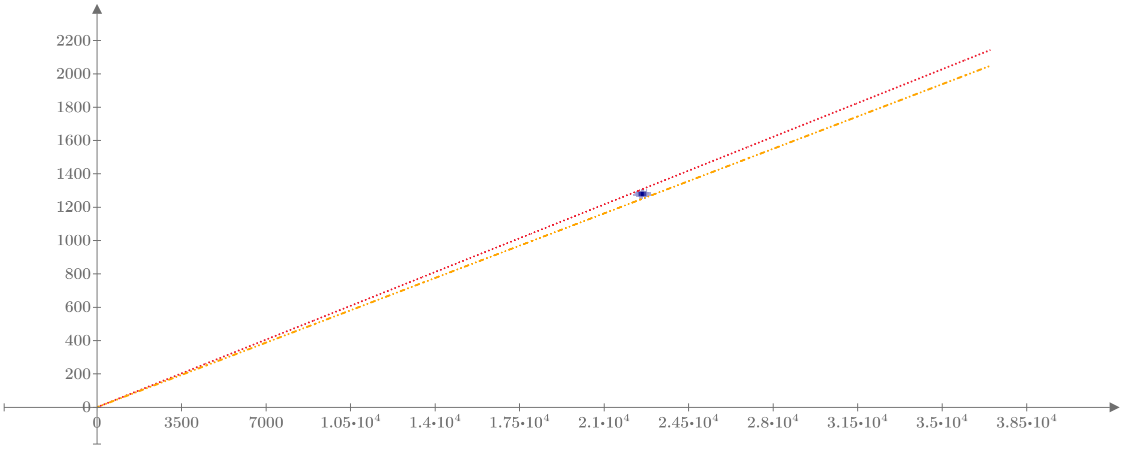


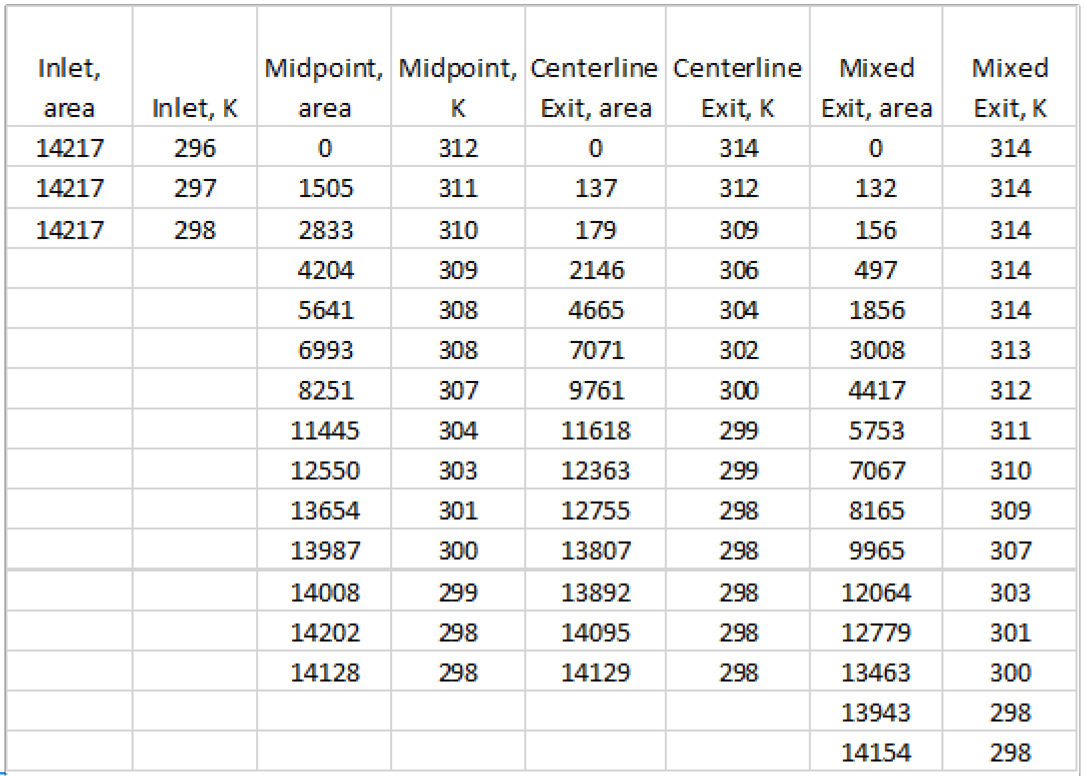
Figure B.2 Uncertainty in gas chromatograph area versus vapor pressure conversion.

$S_m := \mathrm{Stdev}(m) = 0.0003\ \boldsymbol{Pa}$ $\qquad$ $U_{mper} := \dfrac{2 \cdot S_m}{\mathrm{mean}(m)} = 1.1425\ 1\%$

(3) Convert from H2O vapor pressure to CO2 concentration: The ideal gas law will be used for finding the H2O concentration. All uncertainties except for temperature have already been determined. For the uncertainty associated with the measured temperature values, the Omega value of +/- 1 degree C is used (Omega, 2015). As with the calibration analysis the manufacturer specifications are considered to be maximum values (3 standard deviations) per Nakos (2004). Errors are normalized to 2 standard deviations based on the general approach suggested by Lipak (1995).

$c = \dfrac{m \cdot GC}{\boldsymbol{R} \cdot T}$ $\qquad$ $B_T := 0.23 \cdot \boldsymbol{K}$ $\qquad$ $P_T := 0.36 \cdot \boldsymbol{K}$

Table B.3 Gas chromatograph data for H_2O breakthrough test.



Inlet, area	Inlet, K	Midpoint, area	Midpoint, K	Centerline Exit, area	Centerline Exit, K	Mixed Exit, area	Mixed Exit, K
14217	296	0	312	0	314	0	314
14217	297	1505	311	137	312	132	314
14217	298	2833	310	179	309	156	314
		4204	309	2146	306	497	314
		5641	308	4665	304	1856	314
		6993	308	7071	302	3008	313
		8251	307	9761	300	4417	312
		11445	304	11618	299	5753	311
		12550	303	12363	299	7067	310
		13654	301	12755	298	8165	309
		13987	300	13807	298	9965	307
		14008	299	13892	298	12064	303
		14202	298	14095	298	12779	301
		14128	298	14129	298	13463	300
						13943	298
						14154	298

Outputs

$GC_{in} := excel_{\text{“A2:A4”}}$ $\qquad$ $T_{in} := excel_{\text{“B2:B4”}}\ K$

$GC_{mid} := excel_{\text{“C3:C15”}}$ $\qquad$ $T_{mid} := excel_{\text{“D3:D15”}}\ K$ $\qquad$ $GC_{cl} := excel_{\text{“E3:E15”}}$

$T_{cl} := excel_{\text{“F3:F15”}}\ K$ $\qquad$ $GC_{mix} := excel_{\text{“G3:G19”}}$ $\qquad$ $T_{mix} := excel_{\text{“H3:H19”}}\ K$

Data analysis for inlet concentrations:

$i := 1 .. 10000$ $\qquad B_T = 0.23\ \boldsymbol{K}$ $\qquad P_{GCP} = 0.0068$

$j := 1 .. \operatorname{length}\left(T_{in}\right)$ $\qquad \operatorname{length}\left(T_{in}\right) = 3$

$$T_{i,j} := T_{in_j} + \operatorname{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \operatorname{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{in_j} + GC_{in_j} \cdot \operatorname{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{in_{i,j}} := \frac{m_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$S_{cin_j} := \operatorname{Stdev}\left(c_{in}^{\langle j \rangle}\right)$ $\qquad U_{cin_j} := 2 \cdot S_{cin_j}$ $\qquad U_{cinper_j} := \dfrac{2 \cdot S_{cin_j}}{\operatorname{mean}\left(c_{in}^{\langle j \rangle}\right)} = \begin{bmatrix} 1.3413 \\ 1.3352 \\ 1.3329 \end{bmatrix} 1\%$

Data analysis for midpoint concentrations:

$j := 1 .. \operatorname{length}\left(T_{mid}\right)$ $\qquad \operatorname{length}\left(T_{mid}\right) = 13$

$$T_{i,j} := T_{mid_j} + \operatorname{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \operatorname{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{mid_j} + GC_{mid_j} \cdot \operatorname{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{mid_{i,j}} := \frac{m_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$$S_{cmid_j} := \operatorname{Stdev}\left(c_{mid}^{\langle j \rangle}\right) \quad U_{cmid_j} := 2 \cdot S_{cmid_j} \quad U_{cmidper_j} := \frac{2 \cdot S_{cmid_j}}{\operatorname{mean}\left(c_{mid}^{\langle j \rangle}\right)} = \begin{array}{r|l|} 0 & \\ \vdots & \vdots \\ 1 & 1.3362 \\ 2 & 1.3257 \\ 3 & 1.3223 \\ 4 & 1.3529 \\ 5 & 1.3318 \\ 6 & 1.3393 \\ 7 & 1.3283 \\ 8 & 1.3349 \\ 9 & 1.327 \\ 10 & 1.3384 \\ 11 & 1.3418 \\ 12 & 1.3334 \end{array}\ 1\%$$

Data analysis for centerline concentrations:

$j := 1 \,..\, \text{length}\left(T_{cl}\right) \qquad \text{length}\left(T_{cl}\right) = 13$

$$T_{i,j} := T_{cl_j} + \text{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \text{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{cl_j} + GC_{cl_j} \cdot \text{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{cl_{i,j}} := \frac{m_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$$S_{ccl_j} := \text{Stdev}\left(c_{cl}^{\langle j\rangle}\right) \qquad U_{ccl_j} := 2 \cdot S_{ccl_j} \qquad U_{cclper_j} := \frac{2 \cdot S_{ccl_j}}{\text{mean}\left(c_{cl}^{\langle j\rangle}\right)} = \begin{array}{r|l|} 0 & \\ \vdots & \vdots \\ 1 & 1.3325 \\ 2 & 1.3343 \\ 3 & 1.3379 \\ 4 & 1.334 \\ 5 & 1.3252 \\ 6 & 1.3418 \\ 7 & 1.3485 \\ 8 & 1.334 \\ 9 & 1.3359 \\ 10 & 1.3493 \\ 11 & 1.3399 \\ 12 & 1.3389 \end{array} \ 1\%$$

Data analysis for mixed exit concentrations:

$j := 1 \,..\, \text{length}\left(T_{mix}\right) \qquad \text{length}\left(T_{mix}\right) = 17$

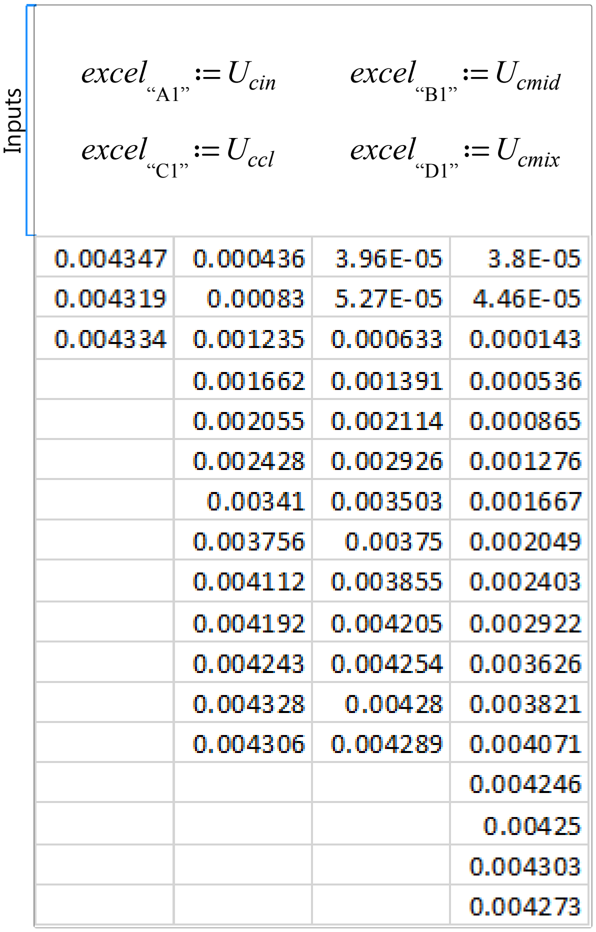
$$T_{i,j} := T_{mix_j} + \text{rnorm}\left(N, 0, \frac{B_T}{2}\right)_i \cdot \boldsymbol{K} + \text{rnorm}\left(N, 0, \frac{P_T}{2}\right)_i \cdot \boldsymbol{K}$$

$$GC_{i,j} := GC_{mix_j} + GC_{mix_j} \cdot \text{rnorm}\left(N, 0, \frac{P_{GCP}}{2}\right)_i$$

$$c_{mix_{i,j}} := \frac{m_i \cdot GC_{i,j}}{\boldsymbol{R} \cdot T_{i,j}}$$

$$S_{cmix_j} := \text{Stdev}\left(c_{mix}^{\langle j\rangle}\right) \quad U_{cmix_j} := 2 \cdot S_{cmix_j} \quad U_{cmixper_j} := \frac{2 \cdot S_{cmix_j}}{\text{mean}\left(c_{mix}^{\langle j\rangle}\right)} = \begin{array}{r|l|} 0 & \\ \vdots & \vdots \\ 2 & 1.3469 \\ 3 & 1.3272 \\ 4 & 1.3382 \\ 5 & 1.3502 \\ 6 & 1.3379 \\ 7 & 1.3414 \\ 8 & 1.335 \\ 9 & 1.3384 \\ 10 & 1.3334 \\ 11 & 1.3426 \\ 12 & 1.346 \\ 13 & 1.3311 \\ \vdots & \vdots \\ 16 & \end{array} \ 1\%$$

Table B.4 Uncertainties for H_2O concentration based on a 95% confidence interval.



$excel_{\text{“A1”}} := U_{cin}$ $\quad$ $excel_{\text{“B1”}} := U_{cmid}$

$excel_{\text{“C1”}} := U_{ccl}$ $\quad$ $excel_{\text{“D1”}} := U_{cmix}$

0.004347	0.000436	3.96E-05	3.8E-05
0.004319	0.00083	5.27E-05	4.46E-05
0.004334	0.001235	0.000633	0.000143
	0.001662	0.001391	0.000536
	0.002055	0.002114	0.000865
	0.002428	0.002926	0.001276
	0.00341	0.003503	0.001667
	0.003756	0.00375	0.002049
	0.004112	0.003855	0.002403
	0.004192	0.004205	0.002922
	0.004243	0.004254	0.003626
	0.004328	0.00428	0.003821
	0.004306	0.004289	0.004071
			0.004246
			0.00425
			0.004303
			0.004273

APPENDIX C

VIRTUAL ADSORPTION TEST SUITE

```
function [] = Adsorption()
% Execute command to open user selected Process or Simulation Excel file
% and convert to structure array
[GenIn] = xl2GenIn;
% Execute command to create the Process Description GUI
% Generic(GenIn,GenIn.Title)
Generic(GenIn);
% end of "Adsorption" function
end
function [] = AdsorptionCL(xlFileName,ParaFileName,RecNum,PlotFlag)
% AdsorptionCL is a version of Adsorption customized for use with X-TOOLSS.
% As such, it must be executed from the Matlab command line without
% interaction. A text file (AdsorptionCLOut) is generated that contains a
% comparison between concentration test data and simulation test data (sum
% of squared errors, or sse). The sse is provided to X-TOOLSS for guidance
% in picking the next values of the parameterized simulation inputs.
% Execute command to open user selected Process or Simulation Excel file
% and convert to structure array
[GenIn] = xl2GenIn(xlFileName);
% Close plot window and GUI so these do not pile up for numerous runs. Put
% a breakpoint or comment out here to debug.
close all
% Execute routine to create the Process Description GUI. Although the GUI
% will be unused for the command line routine, it is assumed to be present
% in many subroutines and provides storage locations for variables.
[hObject] = Generic(GenIn);
% Load record specified by RecNum
for i = 1:numel(GenIn.heading(1,:))
for j = 1: GenIn.NumInputs(i)
DynName = char(GenIn.varname(i,j));
eh = findobj(hObject,'Tag',DynName);
set(eh,'String',GenIn.values(i,j,RecNum))
end
end
rh = findobj(hObject,'Tag','Rec_Number');
set(rh,'String',RecNum);
% Execute routine to modify parametric values updated by X-TOOLSS. First
% call before GenCalc to insure changes are incorporated into calculations.
XTOOLSS_AdjPara(hObject,RecNum,ParaFileName)
% Execute routine to perform calculations required prior to COMSOL
% simulation execution. In interactive mode, eventdata is unused in all
% routines excep Calc, where it signifies adsorption or desorption modes.
% Here, in command line mode, eventdata is used for the record number..
GenCalc(hObject,RecNum)
% Execute routine to modify parametric values updated by X-TOOLSS. Second
% call after GenCalc to replace any calculated values with those in
% ParaFileName.
XTOOLSS_AdjPara(hObject,RecNum,ParaFileName)
% Draw database window for review during execution
drawnow expose
% Execute simulation based on GenIn array values; write SSE value to
% file 'SSE.txt', and plot based on GenIn array selections. First turn off
% automatic saving of .mat file and automatic plotting if user indicated
if PlotFlag == 0;
rh = findobj(hObject,'Label','Auto Save .mat file');
set(rh, 'Checked', 'off');
UserData.AutoPlot = 'off';
set(rh,'UserData',UserData);
rh = findobj(hObject,'Label','Auto Plot');
set(rh, 'Checked', 'off');
UserData.AutoPlot = 'off';
set(rh,'UserData',UserData);
end
RunSimulation(hObject,RecNum)
% end of "AdsorptionCL" function
```

```
function Calc(hObject, eventdata)
% Interactive mode:
% eventdata is 1 or -1
% hObject is GUI handle created by Matlab
% Command line mode:
% eventdata contains RecNum or -RecNum
% hObject is passed figure handle
% Obtain handles for figure and record number. Fetch data storage
% structure array and RecNum variable.
if abs(eventdata) == 0.99
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
RecNum = str2double(get(rh,'String'));
else
RecNum = abs(eventdata);
fh = hObject;
end
GenIn = getappdata(fh,'AppGenIn');
if eventdata > 0; mode = 'Ads';else mode = 'Des';end
Rbar = 8.314472; % [joule/mol/K]
% Calculations
% Total Pressure [kPa] TotPress
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'TotPress']));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag',[mode,'InPres']);
InPres = cell(get(ch,'String'));
InPresParsed = ParseInput(InPres);
InCO2 = InPresParsed(1);
InH2O = InPresParsed(2);
InO2 = InPresParsed(3);
InN2 = InPresParsed(4);
ch = findobj(fh,'Tag',[mode,'InAir']);
InAir = str2double(get(ch,'String'));
ch = findobj(fh,'Tag',[mode,'InHe']);
InHe = str2double(get(ch,'String'));
warningstring1 = ['Air and O2 and/or N2 pressure are defined ',...
'simultaneously for ',mode,'orption. Partial pressures will be combined.',...
' Diffusion calculation accuracy will be poor.'];
warningstring2 = ['Air and He pressures are defined ',...
'simultaneously for ',mode,'orption. N2 and O2 partial pressures will',...
' be used. Diffusion calculation accuracy will be poor.'];
AirDiff = -1; % Air is not present (used for diffusion calculations)
if InAir ~= 0 % "Air" here is H2O, CO2, Ar free
AirDiff = 1; % Can use air for binary diffusion calculations
if InO2 + InN2 ~= 0 && (isnan(InO2) == 0 || isnan(InN2) == 0)
AirDiff = 0; % Can't use air for binary diffusion calculations
dlgname = 'Possible Extraneous Entries!';
warndlg(warningstring1,dlgname);
end
if InHe ~= 0 && isnan(InHe) == 0
AirDiff = 0; % Can't use air for binary diffusion calculations
dlgname = 'Possible Extraneous Entries!';
warndlg(warningstring2,dlgname);
end
InO2 = InO2 + 0.212*InAir;
InN2 = InN2 + 0.788*InAir;
end
TotPress = sum([InCO2,InH2O,InO2,InN2,InHe]);
GenIn.values{HeNu,ItNu,RecNum} = TotPress;
ch = findobj(fh,'Tag',[mode,'TotPress']);
set(ch,'String',TotPress);
end
% Mix Mol Wt [gm/mole] MolWt
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'MolWt']));
if ~isempty([HeNu,ItNu])
```

```
MolWt = (InCO2*44.01 + InH2O*18.015 + InO2*28.013 + ...
InN2*28.013 + InHe*4.003) / TotPress;
GenIn.values{HeNu,ItNu,RecNum} = MolWt;
ch = findobj(fh,'Tag',[mode,'MolWt']);
set(ch,'String',MolWt);
end
% Inlet Gas Density [kg/m3] GasDens
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'GasDens']));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag',[mode,'InTemp']);
InTemp = str2double(get(ch,'String'));
GasDens = TotPress*MolWt/(Rbar*(InTemp+273.15));
GenIn.values{HeNu,ItNu,RecNum} = GasDens;
ch = findobj(fh,'Tag',[mode,'GasDens']);
set(ch,'String',GasDens);
end
% Concentration [mol/m3] Conc
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'Conc']));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag',[mode,'Sorbate']);
Sorbate = char(get(ch,'String'));
switch Sorbate % Find Inlet pp based on Sorbate specified
case 'CO2'; InPP = InCO2; GasNum = 31;
case 'H2O'; InPP = InH2O; GasNum = 440;
case 'O2'; InPP = InO2; GasNum = 460;
case 'N2'; InPP = InN2; GasNum = 455;
case 'He'; InPP = InHe; GasNum = 450;
end
Conc = 1000*InPP/(Rbar*(InTemp + 273.15));
GenIn.values{HeNu,ItNu,RecNum} = Conc;
ch = findobj(fh,'Tag',[mode,'Conc']);
set(ch,'String',Conc);
end
% Superficial Vel [m/s] SupVel
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'SupVel']));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag',[mode,'FlowRate']);
FlowRate = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','StandTemp');
StandTemp = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','StandPress');
StandPress = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','FreeFlowArea');
FreeFlowArea = str2double(get(ch,'String'));
SupVel = FlowRate*(InTemp+273.15)...
*StandPress/(6*FreeFlowArea*(StandTemp+273.15)...
*TotPress); %Inlet Superficial Velocity
GenIn.values{HeNu,ItNu,RecNum} = SupVel;
ch = findobj(fh,'Tag',[mode,'SupVel']);
set(ch,'String',SupVel);
end
% Void Fraction and Sorbent Mass [kg] VoidFraction and SorbentMass
% VoidFractionChk is calculated if VoidFraction is zero; otherwise
% SorbentMassChk is calculated
[HeNuV,ItNuV]=find(strcmp(GenIn.varname,'VoidFractionChk'));
[HeNuS,ItNuS]=find(strcmp(GenIn.varname,'SorbentMassChk'));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'PackedDensity'));
if ~isempty([HeNu,ItNu]) && ~isempty([HeNuV,ItNuV]) && ~isempty([HeNuS,ItNuS])
ch = findobj(fh,'Tag','VoidFraction');
VoidFraction = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','BedLength');
BedLength = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','PartDensity');
PartDensity = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','SorbentMass');
```

```
SorbentMass = str2double(get(ch,'String'));
if VoidFraction == 0
PackedDensity = SorbentMass*10/(FreeFlowArea*BedLength);
VoidFraction = 1 - PackedDensity/PartDensity;
GenIn.values{HeNuV,ItNuV,RecNum} = VoidFraction;
ch = findobj(fh,'Tag','VoidFractionChk');
set(ch,'String',VoidFraction);
GenIn.values{HeNuS,ItNuS,RecNum} = 0;
ch = findobj(fh,'Tag','SorbentMassChk');
set(ch,'String',0);
else
PackedDensity = (1-VoidFraction)*PartDensity;
SorbentMass = PackedDensity*FreeFlowArea*BedLength/10;
GenIn.values{HeNuS,ItNuS,RecNum} = SorbentMass;
ch = findobj(fh,'Tag','SorbentMassChk');
set(ch,'String',SorbentMass);
GenIn.values{HeNuV,ItNuV,RecNum} = 0;
ch = findobj(fh,'Tag','VoidFractionChk');
set(ch,'String',0);
end
GenIn.values{HeNu,ItNu,RecNum} = PackedDensity;
ch = findobj(fh,'Tag','PackedDensity');
set(ch,'String',PackedDensity);
end
% Interstitial Vel [m/s] InterVel
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'InterVel']));
if ~isempty([HeNu,ItNu])
InterVel = SupVel/VoidFraction;
GenIn.values{HeNu,ItNu,RecNum} = InterVel;
ch = findobj(fh,'Tag',[mode,'InterVel']);
set(ch,'String',InterVel);
end
% Viscosity [microPa*s] Visc NOTE: does not accomodate Helium!!!
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'Visc']));
if ~isempty([HeNu,ItNu])
Visc = 0.1*viscmix(31,InCO2,440,InH2O,460,InO2,455,InN2,...
(InTemp + 273.15),0,0,0);
GenIn.values{HeNu,ItNu,RecNum} = Visc;
ch = findobj(fh,'Tag',[mode,'Visc']);
set(ch,'String',Visc);
end
% Molecular Diff [cm2/s] Diff
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'Diff']));
if ~isempty([HeNu,ItNu])
if AirDiff == 1
% Cases with Inlet Air and no N2 or O2; for more than two gases,
% Sorbate should be the trace component
if strcmp(Sorbate,'CO2') % CO2 trace gas in mixture
Diff = diffmix(31,InCO2/100,440,InH2O/100,...
501,InAir/100,450,InHe/100,...
(InTemp + 273.15),0,0,0);
elseif strcmp(Sorbate,'H2O') % H2O trace gas in mixture
Diff = diffmix(440,InH2O/100,31,InCO2/100,...
501,InAir/100,450,InHe/100,...
(InTemp + 273.15),0,0,0);
elseif strcmp(Sorbate,'He') % He trace gas in mixture
Diff = diffmix(450,InHe/100,31,InCO2/100,...
501,InAir/100,440,InH2O/100,...
(InTemp + 273.15),0,0,0);
end
else
% Cases with N2 or O2; for more than two gases, Sorbate should be the
% trace component
if strcmp(Sorbate,'CO2') % CO2 trace gas in mixture
Diff = diffmix(31,InCO2/100,440,InH2O/100,...
460,InO2/100,455,InN2/100,450,InHe/100,...
(InTemp + 273.15),0,0,0);
elseif strcmp(Sorbate,'H2O') % H2O trace gas in mixture
Diff = diffmix(440,InH2O/100,31,InCO2/100,...
460,InO2/100,455,InN2/100,450,InHe/100,...
(InTemp + 273.15),0,0,0);
elseif strcmp(Sorbate,'He') % He trace gas in mixture
Diff = diffmix(450,InHe/100,31,InCO2/100,...
460,InO2/100,455,InN2/100,440,InH2O/100,...
(InTemp + 273.15),0,0,0);
elseif strcmp(Sorbate,'O2') % O2 trace gas in mixture
Diff = diffmix(460,InO2/100,31,InCO2/100,...
450,InHe/100,455,InN2/100,440,InH2O/100,...
(InTemp + 273.15),0,0,0);
elseif strcmp(Sorbate,'N2') % N2 trace gas in mixture
Diff = diffmix(455,InN2/100,31,InCO2/100,...
460,InO2/100,450,InHe/100,440,InH2O/100,...
(InTemp + 273.15),0,0,0);
end
end
Diff=Diff/10000;
GenIn.values{HeNu,ItNu,RecNum} = Diff;
ch = findobj(fh,'Tag',[mode,'Diff']);
set(ch,'String',Diff);
end
% Equivalent Pellet Diameter EqPelDia
% Using the Sauter or surface-volume diameter as recommended by Rase (1990)
% for cylinders. Other geometries have not been included as yet.
[HeNu,ItNu]=find(strcmp(GenIn.varname,'EqPelDia'));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag','PelletLen');
PelletLen = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','PelletDia');
PelletDia = str2double(get(ch,'String'));
if PelletLen == 0
EqPelDia = PelletDia;
else
EqPelDia = (3*PelletDia*PelletLen)/(2*PelletLen+PelletDia);
end
GenIn.values{HeNu,ItNu,RecNum} = EqPelDia;
ch = findobj(fh,'Tag','EqPelDia');
set(ch,'String',EqPelDia);
end
% Area to Volume Ratio [1/m] AreaVolRat
AreaVolRat = 2*1000*(1 - VoidFraction)/(3*EqPelDia);
[HeNu,ItNu]=find(strcmp(GenIn.varname,'AreaVolRat'));
if ~isempty([HeNu,ItNu])
GenIn.values{HeNu,ItNu,RecNum} = AreaVolRat;
ch = findobj(fh,'Tag','AreaVolRat');
set(ch,'String',AreaVolRat);
end
% Reynolds Number ReynNum
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'ReynNum']));
if ~isempty([HeNu,ItNu])
ReynNum = 10^4*GasDens*SupVel*EqPelDia/(Visc*10);
GenIn.values{HeNu,ItNu,RecNum} = ReynNum;
ch = findobj(fh,'Tag',[mode,'ReynNum']);
set(ch,'String',ReynNum);
end
% Residence Time [s] ResTime
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'ResTime']));
if ~isempty([HeNu,ItNu])
ResTime = BedLength/InterVel;
GenIn.values{HeNu,ItNu,RecNum} = ResTime;
ch = findobj(fh,'Tag',[mode,'ResTime']);
```

```
set(ch,'String',ResTime);
end
% Schmidt Number SchmidtNum
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'SchmidtNum']));
if ~isempty([HeNu,ItNu])
SchmidtNum = (Visc*10) / (GasDens * Diff * 1e7);
GenIn.values{HeNu,ItNu,RecNum} = SchmidtNum;
ch = findobj(fh,'Tag',[mode,'SchmidtNum']);
set(ch,'String',SchmidtNum);
end
% Axial Dispersion [m2/s] AxialDisp
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'AxialDisp']));
if ~isempty([HeNu,ItNu])
% Peclet number for 5 correlations;
% (1) Wakao and Funazkri
% (2) Edwards and Richardson
% (3) Wicke
% (4) Ruthven
% (5) Wen and Fan
PecletNum(1) = 1/(20/(ReynNum*SchmidtNum) + 1/2);
PecletNum(2) = 1/(0.73*VoidFraction/(ReynNum*SchmidtNum) + ...
1/(2*(1 + 13*0.73*VoidFraction/(ReynNum*SchmidtNum))));
PecletNum(3) = 1/((0.45 + 0.55*VoidFraction)/(ReynNum*SchmidtNum) + 1/2);
PecletNum(4) = 1/(0.7*VoidFraction/(ReynNum*SchmidtNum) + 1/2);
PecletNum(5) = 1/(0.3/(ReynNum*SchmidtNum) + ...
1/(2*(1 + 3.8/(ReynNum*SchmidtNum))));
AxialDispAll = InterVel*EqPelDia/1000./PecletNum;
AxialDisp = AxialDispAll(2);
GenIn.values{HeNu,ItNu,RecNum} = AxialDisp;
ch = findobj(fh,'Tag',[mode,'AxialDisp']);
set(ch,'String',AxialDisp);
end
% Max Axial Dispersion [m2/s] AxialDisMax
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'AxialDisMax']));
if ~isempty([HeNu,ItNu])
AxialDisMax = max(AxialDispAll);
GenIn.values{HeNu,ItNu,RecNum} = AxialDisMax;
ch = findobj(fh,'Tag',[mode,'AxialDisMax']);
set(ch,'String',AxialDisMax);
end
% Min Axial Dispersion [m2/s] AxialDispMin
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'AxialDisMin']));
if ~isempty([HeNu,ItNu])
AxialDisMin = min(AxialDispAll);
GenIn.values{HeNu,ItNu,RecNum} = AxialDisMin;
ch = findobj(fh,'Tag',[mode,'AxialDisMin']);
set(ch,'String',AxialDisMin);
end
% Film Diffusion [1/s] FilmDiff
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'FilmDiff']));
if ~isempty([HeNu,ItNu])
% Sherwood number for 5 correlations;
% (1) Wakao and Funazkri
% (2) Dwivedi and Upadhyay; Re > 10
% (3) Petrovic and Thodos; 3 < Re < 2000
% (4) Carberry (1960) from Suzuki
% (5) Molerus and Wirth (1997)
SherwoodNum(1) = 2.0 + 1.1*SchmidtNum^(1/3)*ReynNum^(0.6);
SherwoodNum(2) = 0.458/VoidFraction*SchmidtNum^(1/3)*ReynNum^(0.593);
SherwoodNum(3) = 0.357/VoidFraction*SchmidtNum^(0.33)*ReynNum^(0.64);
SherwoodNum(4) = 1.15*VoidFraction^(-1/2)*SchmidtNum^(1/3)*ReynNum^(0.5);
SherwoodNum(5) = 5.49*(1 - VoidFraction)/VoidFraction;
FilmDiffAll = Diff/(EqPelDia*1e-3)*SherwoodNum;
FilmDiff = FilmDiffAll(1);
GenIn.values{HeNu,ItNu,RecNum} = FilmDiff;
```

```
ch = findobj(fh,'Tag',[mode,'FilmDiff']);
set(ch,'String',FilmDiff);
end
% Min Film Diffusion [1/s] FilmDiffMin
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'FilmDiffMin']));
if ~isempty([HeNu,ItNu])
FilmDiffMin = min(FilmDiffAll);
GenIn.values{HeNu,ItNu,RecNum} = FilmDiffMin;
ch = findobj(fh,'Tag',[mode,'FilmDiffMin']);
set(ch,'String',FilmDiffMin);
end
% Max Film Diffusion [1/s] FilmDiffMax
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'FilmDiffMax']));
if ~isempty([HeNu,ItNu])
FilmDiffMax = max(FilmDiffAll);
GenIn.values{HeNu,ItNu,RecNum} = FilmDiffMax;
ch = findobj(fh,'Tag',[mode,'FilmDiffMax']);
set(ch,'String',FilmDiffMax);
end
% Bed Loading [mole/kg] BedLoad
% Return n/a if adsorption isotherm is not present
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'BedLoad']));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag','Sorbent');
Sorbent = char(get(ch,'String'));
switch Sorbent % Assign number based on Sorbent specified
case '5A'; SorbNum = 1;
case '13X'; SorbNum = 2;
case 'Silica Gel'; SorbNum = 3;
otherwise; SorbNum = 0;
end
if SorbNum == 0 || GasNum > 440
BedLoad = NaN;
TothA0 = NaN;
TothB0 = NaN;
TothE = NaN;
TothT0 = NaN;
TothC = NaN;
else
[BedLoad, Toth] = eqiso(1,GasNum,SorbNum,InPP,(InTemp + 273.15));
TothA0 = Toth.a0;
TothB0 = Toth.b0;
TothE = Toth.E;
TothT0 = Toth.t0;
TothC = Toth.c;
end
GenIn.values{HeNu,ItNu,RecNum} = BedLoad;
ch = findobj(fh,'Tag',[mode,'BedLoad']);
set(ch,'String',BedLoad);
[HeNu,ItNu]=find(strcmp(GenIn.varname,'TothA0'));
GenIn.values{HeNu,ItNu,RecNum} = TothA0;
ch = findobj(fh,'Tag','TothA0');
set(ch,'String',TothA0);
[HeNu,ItNu]=find(strcmp(GenIn.varname,'TothB0'));
GenIn.values{HeNu,ItNu,RecNum} = TothB0;
ch = findobj(fh,'Tag','TothB0');
set(ch,'String',TothB0);
[HeNu,ItNu]=find(strcmp(GenIn.varname,'TothE'));
GenIn.values{HeNu,ItNu,RecNum} = TothE;
ch = findobj(fh,'Tag','TothE');
set(ch,'String',TothE);
[HeNu,ItNu]=find(strcmp(GenIn.varname,'TothT0'));
GenIn.values{HeNu,ItNu,RecNum} = TothT0;
ch = findobj(fh,'Tag','TothT0');
set(ch,'String',TothT0);
```

```
[HeNu,ItNu]=find(strcmp(GenIn.varname,'TothC'));
GenIn.values{HeNu,ItNu,RecNum} = TothC;
ch = findobj(fh,'Tag','TothC');
set(ch,'String',TothC);
end
% Solid Concen [mol/m3] SolidConc
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'SolidConc']));
if ~isempty([HeNu,ItNu])
SolidConc = BedLoad*PartDensity;
GenIn.values{HeNu,ItNu,RecNum} = SolidConc;
ch = findobj(fh,'Tag',[mode,'SolidConc']);
set(ch,'String',SolidConc);
end
% Stoichiometric BT Time [s] StoichioTime
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'StoichioTime']));
if ~isempty([HeNu,ItNu])
FlowRateAct = FlowRate*(InTemp+273.15)*StandPress/(StandTemp+273.15)/TotPress;
StoichioTime = BedLoad*Rbar*(InTemp+273.15)*SorbentMass/...
(InPP*FlowRateAct*1000);
GenIn.values{HeNu,ItNu,RecNum} = StoichioTime;
ch = findobj(fh,'Tag',[mode,'StoichioTime']);
set(ch,'String',StoichioTime);
end
% Gas Heat Capacity [kJ/(kg*K)] GasHeatCap *No data for Helium!*
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode 'GasHeatCap']));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag',[mode 'Sorbate']);
Sorbate = char(get(ch,'String'));
GasNumber(1) = 31; GasPP(1) = InCO2; %CO2
GasNumber(2) = 440; GasPP(2) = InH2O; %H2O
GasNumber(3) = 460; GasPP(3) = InO2; %O2
GasNumber(4) = 455; GasPP(4) = InN2; %N2
switch Sorbate % Assign number based on Sorbate specified
case 'CO2'; SorbateNum = 1;
case 'H2O'; SorbateNum = 2;
case 'O2'; SorbateNum = 3;
case 'N2'; SorbateNum = 4;
otherwise; SorbateNum = 0;
end
CpNum = 0; CpMol = zeros(4); Cp = zeros(4);
for i = 1:4
prop = gas_prop(GasNumber(i));
CpMol(i) = prop.CPVAPA + prop.CPVAPB*(InTemp+273.15) +...
prop.CPVAPC*(InTemp+273.15)^2 + prop.CPVAPD*(InTemp+273.15)^3;
Cp(i) = CpMol(i)/prop.Molwt;
if SorbateNum == i
SorbGasHeatCap = Cp(i);
end
CpNum = CpNum + Cp(i)*GasPP(i);
end
GasHeatCap = CpNum/TotPress;
GenIn.values{HeNu,ItNu,RecNum} = GasHeatCap;
ch = findobj(fh,'Tag',[mode,'GasHeatCap']);
set(ch,'String',GasHeatCap);
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode,'SorbGasHeatCap']));
ch = findobj(fh,'Tag',[mode,'SorbGasHeatCap']);
set(ch,'String',SorbGasHeatCap);
GenIn.values{HeNu,ItNu,RecNum} = SorbGasHeatCap;
end
% Heat Transfer Coefficient from Particle to Free Stream via similarity
% with film diffusion. Sherwood number previously found from 5
% correlations but uses Wakao and Funazkri based on recommendations in the
% literature
% Sorb-Gas H [W/(m^2*K)] SorbGasH
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode,'SorbGasH']));
```

```
if ~isempty([HeNu,ItNu])
GasHeatCapVol = GasHeatCap*GasDens;
SorbGasHAll = 1000*FilmDiffAll*GasHeatCapVol;
SorbGasH = SorbGasHAll(1);
GenIn.values{HeNu,ItNu,RecNum} = SorbGasH;
ch = findobj(fh,'Tag',[mode,'SorbGasH']);
set(ch,'String',SorbGasH);
end
% Effective Axial Thermal Conductivity: The Yagi et al. correlation is
% recommended on the basis of verification against test data in Kavinany
% 1995, Figure 4.15. Here it is noted that the Yagi et al. prediction is
% not accurate at packed bed Reynolds numbers above about 50.
% Axial Conduct [W/(m*K)] AxiCond
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode,'AxiCond']));
if ~isempty([HeNu,ItNu])
% Gas conductivity is based on the mass similarity approach;
% alternatively it can be calculated via Chapman-Enskog (Bird et al.
% section 8.3)
GasCond = Diff*GasHeatCapVol*1000;
ch = findobj(fh,'Tag','SorbCond');
SorbCond = str2double(get(ch,'String'));
nExp = 0.280 - 0.757*log10(VoidFraction) - 0.057*log10(SorbCond/GasCond);
CondQuies = GasCond*(SorbCond/GasCond)^nExp;
Prandtl = GasHeatCapVol*Visc/(1000*GasDens*GasCond);
AxiCond = GasCond*(CondQuies/GasCond + 0.75*Prandtl*ReynNum);
GenIn.values{HeNu,ItNu,RecNum} = AxiCond;
ch = findobj(fh,'Tag',[mode,'AxiCond']);
set(ch,'String',AxiCond);
end
% Heat Transfer to Column Wall
%
% Recommended Correlation for Heat Transfer Coefficient (2-D) to Column
% Wall and Overall Heat Transfer Coefficient (1-D) to Column Wall:
%
% Li and Finlayson (Chem. Eng. Sci. 1977) evaluated published data and
% isolated the data free from entrance or bed length effects. Their
% resulting correlations for the wall heat transfer coefficient,
% applicable to 2-D models, is shown to fit data with packed bed Reynolds
% number of 10 to 2000 for cylindrical and spherical particles (Wakao and
% Kaguei, Figure 5.17 and 5.18). Their correlation is also compared with
% data with packed bed Reynolds numbers from 100 to 2000 in Wen and Ding,
% and recommended over 4 other correlations. Thus the Li and Finlayson
% correlation is recommended for use.
%
% By extension, the Li and Finlayson correlation for 1-D packed bed models
% is also recommended. No data was found to compare with the Overall
% Heater Transfer Coefficient correlations.
% Gas-Can H [W/(m^2*K)] GasCanH
[HeNu,ItNu]=find(strcmp(GenIn.varname,[mode,'GasCanH']));
if ~isempty([HeNu,ItNu])
EqCanID = 2/100*sqrt(FreeFlowArea/pi);
if PelletLen == 0
Nusselt = 2.03*ReynNum^0.8*exp(-6*EqPelDia/1000/EqCanID);
else
Nusselt = 1.26*ReynNum^0.95*exp(-6*EqPelDia/1000/EqCanID);
end
GasCanH = GasCond/EqCanID*Nusselt;
GenIn.values{HeNu,ItNu,RecNum} = GasCanH;
ch = findobj(fh,'Tag',[mode,'GasCanH']);
set(ch,'String',GasCanH);
end
% Save current values to GenIn array
setappdata(fh,'AppGenIn',GenIn);
% end of function
% --------------------------------------------------------------------
```

```
function diffb = diffbin(gasnum1,pres1,gasnum2,pres2,temp,x,y,z)
% diffbin gives binary gas diffusion given temp. and partial pressures
% Syntax: diff = diffusion(gasnum1,pres1,gasnum2,pres2,temp,x,y,z)
% Input: Inputs are vectors of length nnode from COMSOL GUI
% gasnum1, gasnum2 = identifying number for each gas
% pres1, pres2 = partial pressure of each gas in bars
% temp = temperature of gas mixture at each node, Kelvin
% x = COMSOL x dimension of each node
% y = COMSOL y dimension of each node; set to zero if 1-D
% z = COMSOL z dimension of each node; set to zero if 2-D
% Local: nnode = number of COMSOL nodes for 1-D model
% gasnum(ngas) - local array version
% pres(ngas, nnode) - local array version
% Called
% Routines:
% function prop = gas_prop(gasnum)
% function provides gas properties based on identifying gas number
% Output: diffusion of gas in cm^2 / second
% Notes:
% Function is based on Fuller method as presented in "Properties of Gases
% and Liquids" by Polling et. al., 5th Edition.
%
% MATLAB version by J. Knox 01-30-10
%%
if nargin~=8
error(['Gas number, pressures , temperature,', ...
' and x, y, z node dimensions are required inputs for diffbin function']);
end
% Get properties and perform calcs for each gas
%% Set up local arrays from inputs and preallocate arrays
nnode = length(x);
if length(y) ~= nnode || length(z) ~= nnode
error('x, y, and z must be of the same length');
end
%% Calculate node-dependent properties
diffb =zeros(1,nnode);
for i = 1:nnode;
%
% Binary molecular weight, total pressure
%
gas_p1 = gas_prop(gasnum1(i));
gas_p2 = gas_prop(gasnum2(i));
Mab = 2*((1/gas_p1.Molwt)+(1/gas_p2.Molwt))^(-1);
pres = pres1(i) + pres2(i);
% Equation 11-4.4
%
diffb(i) = 0.00143*temp(i)^1.75/(pres*Mab^0.5*(gas_p1.Sigma^(1/3)...
+ gas_p2.Sigma^(1/3))^2);
end
function diffm = diffmix(gasnum1,pres1,gasnum2,pres2,gasnum3,pres3, ...
gasnum4,pres4,gasnum5,pres5,temp,x,y,z)
% diffmix gives gas mixture diffusion given temp. and partial pressures
% Syntax: diffm = diffmix(gasnum1,pres1,gasnum2,pres2,gasnum3,pres3, ...
% gasnum4,pres4,gasnum5,pres5,temp,x,y,z)
% Input: Inputs are vectors of length nnode from COMSOL GUI
% gasnum1 - gasnum5 = identifying number for each gas. gasnum1 is
% assumed to be the dilute gas for which diffusion is returned.
% pres1 - pres5 = partial pressure or mole fraction of gas 1 - 5
% in units of bars
% temp = temperature of gas mixture at each node, Kelvin
% x = COMSOL x dimension of each node
% y = COMSOL y dimension of each node; set to zero if 1-D
% z = COMSOL z dimension of each node; set to zero if 2-D
% Local: nnode = number of COMSOL nodes for 1-D model
% gasnum(ngas) - local array version
% pres(ngas, nnode) - local array version
% Called
% Routines:
% function prop = gas_prop(gasnum)
% function provides gas properties based on identifying gas number
% Output: diffusion of gas in cm^2 / second
% Notes:
% Function is based on Blanc's law as presented in "Properties of Gases
% and Liquids" by Polling et. al., 5th Edition. Function "diffbin" is used
% to find binary diffusion. Appliciability is limited to a dilute gas
% diffusing in a homogeneous mixture.
%
% Uses binary diffusion (diffbin) if possible for better accuracy. Use air
% (No. 501) to utilize diffbin instead of oxygen and nitrogen (which will
% use diffmix) for much better accuracy.
%
% MATLAB version by J. Knox 02-4-10
% added 5th gas: J. Knox 07-17-10
%
if nargin>14 || nargin<8
error(['Gas number, pressures or mole fractions, temperature,', ...
' are required inputs for viscosity function', ...
' with maximum of 4 gases and minimum of 2 gases allowed']);
end
% Get properties and perform calcs for each gas
%% Set up local arrays from inputs and preallocate arrays
gasnum(1,:) = gasnum1;
gasnum(2,:) = gasnum2;
pres(1,:) = pres1;
pres(2,:) = pres2;
switch nargin
case 14
gasnum(3,:) = gasnum3;
pres(3,:) = pres3;
gasnum(4,:) = gasnum4;
pres(4,:) = pres4;
gasnum(5,:) = gasnum5;
pres(5,:) = pres5;
case 12
gasnum(3,:) = gasnum3;
pres(3,:) = pres3;
gasnum(4,:) = gasnum4;
pres(4,:) = pres4;
temp = gasnum5;
x = pres5;
y = temp;
z = x;
case 10
gasnum(3,:) = gasnum3;
pres(3,:) = pres3;
temp = gasnum4;
x = pres4;
y = gasnum5;
z = pres5;
case 8
temp = gasnum3;
x = pres3;
y = gasnum4;
z = pres4;
end
nnode = length(x);
if length(y) ~= nnode || length(z) ~= nnode
error('x, y, and z must be of the same length');
end
% Sort input gases by pressure value. First pressure assumed lowest for
% non-binary mixtures.
diffm = zeros(1,nnode);
for i = 1:nnode
ngas = (nargin - 4)/2;
PreSort = [gasnum(2:end,i) pres(2:end,i)];
PostSort = sortrows(PreSort,-2);
gasnum(2:ngas,i) = PostSort(:,1);
pres(2:ngas,i) = PostSort(:,2);
% For pressure inputs of zero, reduce ngas to non-zero inputs
switch nargin
case 14
ngas = 5;
if (pres(5,i) == 0) && (pres(4,i) == 0) && (pres(3,i) == 0)
ngas = 2;
elseif (pres(5,i) == 0) && (pres(4,i) == 0)
ngas = 3;
elseif pres(5,i) == 0
ngas = 4;
end
case 12
ngas = 4;
if (pres(4,i) == 0) && (pres(3,i) == 0)
ngas = 2;
elseif pres(4,i) == 0
ngas = 3;
end
case 10
ngas = 3;
if pres(3,i) == 0
ngas = 2;
end
case 8
ngas = 2;
end
% For two gases, use diffbin
if ngas == 2
diffm(i) = diffbin(gasnum(1,i),pres(1,i),gasnum(2,i),pres(2,i),...
temp(i),x(i),y(i),z(i));
else
sumBlanc = 0;
for j = 2:ngas
% Obtain binary gas diffusion for current gas; first find
% (total pressure - first gas pressure) and assign to second
% gas such that binary diffusion is calculated with total
% pressure
presj = sum(pres(:,i)) - pres(1,i);
diffb=diffbin(gasnum(1,i),pres(1,i),gasnum(j,i),presj,...
temp(i),x(i),y(i),z(i));
% mole fraction current gas, sum in Blanc equation
xmole = pres(j,i)./sum(pres(:,i));
sumBlanc = sumBlanc + xmole./diffb;
end
diffm(i)=1./sumBlanc;
end
end
function DoubleXYPlot(XData1, YData1, YData2, YData3, x1Label, y1Label,
x2Label, ...
y2Label, Data1Label, Data2Label,Data3Label)
%DoubleXYPlot(XData1, YData1, YData2, YData3, x1Label, y1Label, x2Label, ...
% y2Label, Data1Label, Data2Label,Data3Label)XDATA1,YDATA1,YDATA2,YDATA3)
% XDATA1: line xdata
% YDATA1: line ydata
% YDATA2: line ydata
% YDATA3: line ydata
% Auto-generated by MATLAB on 01-Dec-2008 18:56:25
% Modifed by J. Knox on 01-Dec-2008
% Create figure
figure1 = figure('XVisual',...
'0x24 (TrueColor, depth 24, RGB mask 0xff0000 0xff00 0x00ff)');
% Create axes
axes1 = axes('Parent',figure1,...
'YGrid','on',...
'YColor',[0 0 0],...
'XGrid','on',...
'XColor',[0 0 0]);
% Create line
line(XData1,YData1,'Parent',axes1,'MarkerFaceColor',[0 0 0],...
'MarkerEdgeColor',[0 0 0],...
'Marker','.',...
'MarkerSize',6,...
'LineWidth',2,...
'LineStyle','none',...
'Color',[0 1 0],...
'DisplayName',Data1Label);
% Create line
line(XData1,YData2,'Parent',axes1,'Marker','o',...
'MarkerSize',8,...
'DisplayName',Data2Label,...
'MarkerEdgeColor',[0 0 0],...
'Color',[0 0 0]);
% Create xlabel
xlabel(x1Label);
% Create ylabel
ylabel(y1Label);
% Create axes
axes2 = axes('Parent',figure1,'ZColor',[1 0 0],'YAxisLocation','right',...
'XAxisLocation','top',...
'Color','none',...
'YColor',[1 0 0],...
'XColor',[1 0 0]);
% Create line
line(XData1,YData3,'Parent',axes2,'MarkerEdgeColor',[1 0 0],'Marker','o',...
'Color',[1 0 0],...
'DisplayName',Data3Label);
% Create xlabel
xlabel(x2Label,...
'VerticalAlignment','bottom',...
'FontWeight','bold',...
'FontSize',12,...
'Color',[0 0 0]);
% Create ylabel
ylabel(y2Label,'VerticalAlignment','cap','Color',[1 0 0]);
% Create legend
legend1 = legend(axes1,'show');
set(legend1,'Position',[0.74 0.6275 0.1241 0.05868]);
% Create legend
legend2 = legend(axes2,'show');
set(legend2,'Position',[0.75 0.4975 0.09384 0.03261],'Color',[1 1 1]);
function EditCallback(hObject, ~)
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
GenIn = getappdata(fh,'AppGenIn');
RecNum = str2double(get(rh,'String'));
UserData = get(hObject,'UserData');
if isnan(str2double(get(hObject,'String')))
GenIn.values{UserData.i,UserData.j,RecNum} = char(get(hObject,'String'));
else
GenIn.values{UserData.i,UserData.j,RecNum} = str2double(get(hObject,'String'));
end
setappdata(fh,'AppGenIn',GenIn);
```

```
cmhck = findobj('Label','Auto Calc');
UserDataCheck = get(cmhck(1),'UserData');
if strcmp('on',UserDataCheck.AutoCalc)
GenCalc(hObject,0)
end
% end of function
function [BedLoad, Toth] = eqiso(eq,gas,sorbent,pres,temp)
% EQISO function determines sorbent bed loading from gas pressure, temp
%
% Syntax: bedld = eqiso(eq,gas,sorbent,pres,temp)
%
% Input: eq = number of equilibrium isotherm equation: Toth = 1
% gas = sorbate defined using number in gas_prop routine
% sorbent = sorbent type; 5A = 1, 13X = 2, Silica Gel = 3
% pres = sorbate partial pressure, kPa
% temp = temperature to be evaluated, K
%
% Output: bedld = equilibrium bed loading, mol/kg
%
% Notes: Data for the Toth isotherm is taken from Wang and LeVan,
% "Adsorption Equilibrium of Carbon Dioxide and Water Vapor on Zeolites 5A
% and 13X and Silica Gel: Pure Properties, J. Chem. Eng. Data, 2009
%
% Temperature a = saturation capacity
% dependent b = an equilibrium constant related to the a
% variables: t = heterogeneity constant
%
% Constant a0 = constant, [mol/kg/kPa]
% Values: b0 = constant, [1/kPa]
% E = constant, [K]
% t0 = constant, dimless
% c = constant, [K]
%
if nargin<5
error('equation name, gas #, sorbent, pressure and temp required for eqiso');
end
if eq == 1 % Toth equation numer
% -- do nothing
else
error('equation ''%s'' is not supported in eqiso',eq);
end
%
% --- Build adsorption isotherm database
%
TothEq = struct(...
'No', { 1, 2, 3, 4, 5 6 },...
'System',{ 'CO2_5A', 'H2O_5A', 'CO2_13X', 'H2O_13X', 'CO2_SG', 'H2O_SG' },...
'a0' ,{ 9.875E-07, 1.106E-08,6.509E-03, 3.634E-06, 7.678E-06, 1.767E+02},...
'b0' ,{ 6.761E-08, 4.714E-10,4.884E-04, 2.408E-07, 5.164E-07, 2.787E-05},...
'E' ,{ 5.625E+03, 9.955E+03,2.991E+03, 6.852E+03, 2.330E+03, 1.093E+03},...
't0' ,{ 2.700E-01, 3.548E-01,7.487E-02, 3.974E-01,-3.053E-01,-1.190E-03},...
'c' ,{-2.002E+01,-5.114E+01,3.805E+01,-4.199E+00, 2.386E+02, 2.213E+01});
% Determine system and apply appropriate equation
switch sorbent
case 1
switch gas
case 31
No = 1;
case 440
No = 2;
end
case 2
switch gas
case 31
No = 3;
```

```
case 440
No = 4;
end
case 3
switch gas
case 31
No = 5;
case 440
No = 6;
end
end
% Load values array
Toth.a0 = TothEq(No).a0;
Toth.b0 = TothEq(No).b0;
Toth.E = TothEq(No).E;
Toth.t0 = TothEq(No).t0;
Toth.c = TothEq(No).c;
% Apply Toth equation
a = TothEq(No).a0*exp(TothEq(No).E/temp);
b = TothEq(No).b0*exp(TothEq(No).E/temp);
t = TothEq(No).t0 + TothEq(No).c/temp;
BedLoad = a*pres / (1 + (b*pres)^t)^(1/t);
function [SSE, OffSet] = ErrFun(time,out,SSEFileName,ScopeErrIn)
% Function to calculate Sum of Squared Errors (SSE) between time histories
% Files containing test histories are assumed to have identical time steps as
% simulation data output.
%
% Approach is to ignore any time differential and match shape of curve
% within a user specified percent of full scale (max-min) from the
% mid-height to simplify optimization
%
% Test data file name format is "Conc 1000 SSE.txt" for outlet concentration;
% "Conc 500 975 SSE.txt" for samples at 50% and 97.5% of bed length.
%
% Added option to bypass complexities of picking midheight since it does
% not always work and just evaluate entire dataset if scope of error check
% is set to 101% and set offset to 1.01
TestData = importdata(cell2mat(SSEFileName));
DataLength = numel(TestData(:,1));
if numel(time)== DataLength
ParsedSSEFileName = ParseInputText(SSEFileName);
ParsedScopeErr = ParseInput(ScopeErrIn);
TimeStep = time(end)/(DataLength-1);
SSEVar = cell2mat(ParsedSSEFileName(1));
NumTestLocs = length(ParsedSSEFileName) - 2;
switch SSEVar
case 'Conc'
SimData = out.c;
ScopeErr = ParsedScopeErr(1);
PeakIndTest = 1;
PeakIndSim = 1;
case 'GasTemp'
SimData = out.Tg;
ScopeErr = ParsedScopeErr(2);
PeakIndTest = 0;
case 'ColTemp'
SimData = out.Tw;
ScopeErr = ParsedScopeErr(3);
PeakIndTest = 0;
end
NumSimLocs = length(SimData(1,:))-1;
for i = 1:NumTestLocs
TestLoc = str2double(cell2mat(ParsedSSEFileName(1+i)))/1000;
LocIndex = round((NumSimLocs)*TestLoc)+1;
MinTest = min(TestData(:,i+1));
```

```
RangeTest = max(TestData(:,i+1)) - MinTest;
MidHtTest = RangeTest/2 + min(TestData(:,i+1));
%Skip complexities of picking midheight since it does
% not always work and just evaluate entire dataset if
% scope of error check is set to 101% and set offset to 1.01
if ScopeErr ~= 101
% For temperature profiles, capture portion following peak for
% evaluation of SSE and OffSet. An exception is thermal
% characterization data; here treat like a breakthrough curve
if PeakIndTest ~= 1
PeakIndTest =
TestData(find(TestData(:,i+1)>=max(TestData(:,i+1)),1),1)/TimeStep + 1;
PeakIndSim =
time(find(SimData(:,LocIndex)>=max(SimData(:,LocIndex)),1),1)/TimeStep + 1;
if PeakIndSim == length(SimData(:,1))
PeakIndTest = 1;
PeakIndSim = 1;
end
end
% del MinSim = min(SimData(:,i+1));
% del RangeSim = max(SimData(:,LocIndex)) - MinSim;
%MidHtSim = RangeSim/2 + min(SimData(:,LocIndex));
% For concentration, looking at range of entire curve
if PeakIndSim == 1
MidTimeTest = TestData(find(TestData(:,i+1)>=MidHtTest,1),1);
MidTimeSim = time(find(SimData(:,LocIndex)>=MidHtTest,1));
% For temperature, looking at range of portion following peak
else
MidTimeTest = TestData(find(TestData(PeakIndTest:end,i+1)<=MidHtTest,1) +
PeakIndTest -
1,1);
MidTimeSim = time(find(SimData(PeakIndSim:end,LocIndex)<=MidHtTest,1) +
PeakIndSim -
1);
end
% Calculate OffSet between sim data and test data and adjust sim
% data for SSE calculation
% debug figure
% debug plot (time,SimData(:,LocIndex),TestData(:,1),TestData(:,i+1))
OffSet(i) = MidTimeTest - MidTimeSim; %#ok<AGROW>
% del OffSet(i) = MidTimeSim - MidTimeTest; %#ok<AGROW>
OffSetStep = OffSet(i)/TimeStep;
if OffSetStep > 0
SimData(OffSetStep+1:end,LocIndex) = SimData(1:end-OffSetStep,LocIndex);
SimData(1:OffSetStep,LocIndex) = SimData(OffSetStep+1,LocIndex);
elseif OffSetStep < 0
SimData(1:end+OffSetStep,LocIndex) = SimData(1-OffSetStep:end,LocIndex);
SimData(end+OffSetStep+1:end,LocIndex) = SimData(end+OffSetStep,LocIndex);
end
% debug hold
% debug plot (time,SimData(:,LocIndex),TestData(:,1),TestData(:,i+1))
% SSE Calculation within ScopeErr normalized by RangeSim or RangeTest
% Calculate range around mid height.
ScopeHtLo = MidHtTest - RangeTest*ScopeErr/2/100;
ScopeHtHi = MidHtTest + RangeTest*ScopeErr/2/100;
% For concentration and thermal characterization, looking at range of entire
curve
if PeakIndSim == 1
% For 100% comparison, skip calculation and compare all data points
if ScopeErr == 100
ScopeIndLo = 1;
ScopeIndHi = DataLength;
else
ScopeIndLo = time(find(TestData(:,i+1)>=ScopeHtLo,1))/TimeStep;
if ScopeIndLo == 0; ScopeIndLo = 1; end
```

```
ScopeIndHi = time(find(TestData(:,i+1)>=ScopeHtHi,1))/TimeStep;
end
% For temperature during BT test, looking at range of portion following peak
else
ScopeIndHi = time(find(TestData(PeakIndTest:end,i+1)<=ScopeHtHi,1))/TimeStep +
PeakIndTest;
ScopeIndLo = time(find(TestData(PeakIndTest:end,i+1)<=ScopeHtLo,1))/TimeStep +
PeakIndTest;
end
SSE(i) = sum(((TestData(ScopeIndLo:ScopeIndHi,i+1)...
-SimData(ScopeIndLo:ScopeIndHi,LocIndex))./RangeTest).^2); %#ok<AGROW>
else
OffSet(i) = 1.01; %#ok<AGROW>
SSE(i) = sum(((TestData(:,i+1)...
-SimData(:,LocIndex))./RangeTest).^2); %#ok<AGROW>
end
end
end
% Prior approach is stored below....
% % Compare simulation data with test data for goodness of fit. Test data has
% % same time steps as simulation, and is in text file with column format:
% % %Time %Conc1 %Conc2. This file may be generated in COMSOL interactively.
% % Create a plot and Export; then remove headers to fit the specified
% % format.
%
% % Test data file name format is "Conc 1000 SSE.txt" for outlet concentration;
% % "Conc 500 975 SSE.txt" for samples at 50% and 97.5% of bed length.
% % Handles max of 2 SSE comparison locations
%
%
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'SSEFile'));
% SSEFileName = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
% ParsedSSEFileName = ParseInputText(GenIn.values(HeNu,ItNu,RecNum));
% if ~strcmp(SSEFileName,'none')
% TestDataSSE = importdata(SSEFileName);
% DataLength = numel(TestDataSSE(:,1));
% if numel(time)== DataLength
% % User input OffSet slides test data to right if positive. Adjustment
% % requires (1) shifting data to closest time step and (3) buffering start
% % or end of data by extrapolating from closest value
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'OffSet'));
% OffSet = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'TimeStep'));
% TimeStep = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
% ShiftSteps = round(OffSet./TimeStep);
% SSEVar = cell2mat(ParsedSSEFileName(1));
% Location = length(out.c(1,:))-1;
% SSELocations = length(ParsedSSEFileName) - 2;
% if SSELocations == 1
% SSELoc(1) = str2double(cell2mat(ParsedSSEFileName(2)))/1000;
% LocIndex(1) = round((Location)*SSELoc(1))+1;
% switch SSEVar
% case 'Conc'
% if ShiftSteps(1) > 0
% TestDataSSE(ShiftSteps(1)+1:end,2) = TestDataSSE(1:end-ShiftSteps(1),2);
% TestDataSSE(1:ShiftSteps(1),2) = TestDataSSE(ShiftSteps(1)+1,2);
% elseif ShiftSteps(1) < 0
% TestDataSSE(1:end+ShiftSteps(1),2) = TestDataSSE(1-ShiftSteps(1):end,2);
% TestDataSSE(end+ShiftSteps(1)+1:end,2) = TestDataSSE(end+ShiftSteps(1),2);
% end
% SimTestSSE=sum((out.c(:,LocIndex(1)) - TestDataSSE(:,2)).^2);
% case 'Temp'
% if ShiftSteps(2) > 0
% TestDataSSE(ShiftSteps(2)+1:end,2) = TestDataSSE(1:end-ShiftSteps(2),2);
% TestDataSSE(1:ShiftSteps(2),2) = TestDataSSE(ShiftSteps(2)+1,2);
% elseif ShiftSteps(2) < 0
% TestDataSSE(1:end+ShiftSteps(2),2) = TestDataSSE(1-ShiftSteps(2):end,2);
% TestDataSSE(end+ShiftSteps(2)+1:end,2) = TestDataSSE(end+ShiftSteps(2),2);
% end
% SimTestSSE=sum((out.Tg(:,LocIndex(1)) - TestDataSSE(:,2)).^2);
% end
% elseif SSELocations == 2
% SSELoc(1) = str2double(cell2mat(ParsedSSEFileName(2)))/1000;
% SSELoc(2) = str2double(cell2mat(ParsedSSEFileName(3)))/1000;
% LocIndex(1) = round((Location)*SSELoc(1))+1;
% LocIndex(2) = round((Location)*SSELoc(2))+1;
% switch SSEVar
% case 'Conc'
% for i = 2:3
% if ShiftSteps(1) > 0
% TestDataSSE(ShiftSteps(1)+1:end,i) = TestDataSSE(1:end-ShiftSteps(1),i);
% TestDataSSE(1:ShiftSteps(1),i) = TestDataSSE(ShiftSteps(1)+1,i);
% elseif ShiftSteps(1) < 0
% TestDataSSE(1:end+ShiftSteps(1),i) = TestDataSSE(1-ShiftSteps(1):end,i);
% TestDataSSE(end+ShiftSteps(1)+1:end,i) = TestDataSSE(end+ShiftSteps(1),i);
% end
% end
% SimTestSSE=sum((out.c(:,LocIndex(1)) - TestDataSSE(:,2)).^2) + ...
% sum((out.c(:,LocIndex(2)) - TestDataSSE(:,3)).^2);
% case 'Temp'
% for i = 2:3
% if ShiftSteps(2) > 0
% TestDataSSE(ShiftSteps(2)+1:end,i) = TestDataSSE(1:end-ShiftSteps(2),i);
% TestDataSSE(1:ShiftSteps(2),i) = TestDataSSE(ShiftSteps(2)+1,i);
% elseif ShiftSteps(2) < 0
% TestDataSSE(1:end+ShiftSteps(2),i) = TestDataSSE(1-ShiftSteps(2):end,i);
% TestDataSSE(end+ShiftSteps(2)+1:end,i) = TestDataSSE(end+ShiftSteps(2),i);
% end
% end
% SimTestSSE=sum((out.Tg(:,LocIndex(1)) - TestDataSSE(:,2)).^2) + ...
% sum((out.Tg(:,LocIndex(2)) - TestDataSSE(:,3)).^2);
% end
% end
% end
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'SimTestSSE'));
% GenIn.values{HeNu,ItNu,RecNum} = SimTestSSE;
% ch = findobj(fh,'Tag','SimTestSSE');
% set(ch,'String',SimTestSSE);
% % Save SSE value to file for X-TOOLSS
% dlmwrite('SSE.txt',SimTestSSE);
% end
function ErrorToFile(hObject,time,out)
% ErrorToFile Function to calculate errors between simulation and test and
% assess breakthrough sharpening
%
% Call routines to calculate errors between simulation and test and
% calculate maximum slope of internal concentration history and
% breakthrough curve. Function is called from RunSimulation function.
% Get GenIn array and Rec_Number. mode is hard coded since X-TOOLSS is not
% being used at this point
fh = ancestor(hObject,'figure');
GenIn = getappdata(fh,'AppGenIn');
rh = findobj(fh,'Tag','Rec_Number');
RecNum = str2double(get(rh,'String'));
mode = 'Interactive';
% Get user settings (ScopeErr, ScaleErr)
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ScopeErr'));
ScopeErr = GenIn.values(HeNu,ItNu,RecNum);
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ScaleErr'));
ScaleErr = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
```

```
% Get required variables for calculations
[HeNu,ItNu]=find(strcmp(GenIn.varname,'HalfCycleLength'));
HalfCycleLength = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'AdsSupVel'));
AdsSupVel = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'FreeFlowArea'));
FreeFlowArea = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'SorbentMass'));
SorbentMass = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'SorbentMassChk'));
SorbentMassChk = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'PartDensity'));
PartDensity = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
% Initialize variables
SumSimTest = 0;
NumScaleErr = 0;
CalcOffSet = [];
% Calculate errors in concentration profiles
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ConcSSEFile'));
ConcSSEFile = GenIn.values(HeNu,ItNu,RecNum);
if ~strcmp(cell2mat(ConcSSEFile),'none')
[ConcSimTestSSE, ConcOffSet] = ErrFun(time,out,ConcSSEFile,ScopeErr);
NumScaleErr = length(ConcSimTestSSE);
SumSimTest = sum(ScaleErr(1:NumScaleErr).*ConcSimTestSSE);
% NumScaleErr = NumScaleErr + 1; dont get this statement today
SumSimTest = SumSimTest +
ScaleErr(NumScaleErr)*sum(abs(ConcOffSet))./HalfCycleLength;
CalcOffSet = ConcOffSet;
else
ConcSimTestSSE = 0;
ConcOffSet = 0;
end
% Calculate errors in gas temperature profiles
[HeNu,ItNu]=find(strcmp(GenIn.varname,'GasTempSSEFile'));
GasTempSSEFile = GenIn.values(HeNu,ItNu,RecNum);
if ~strcmp(cell2mat(GasTempSSEFile),'none')
[GasSimTestSSE, GasTempOffSet] = ErrFun(time,out,GasTempSSEFile,ScopeErr);
NumScaleErrEnd = NumScaleErr + length(GasSimTestSSE);
SumSimTest = SumSimTest +
sum(ScaleErr(NumScaleErr+1:NumScaleErrEnd).*GasSimTestSSE);
NumScaleErr = NumScaleErrEnd + 1;
SumSimTest = SumSimTest +
ScaleErr(NumScaleErr)*sum(abs(GasTempOffSet))./HalfCycleLength;
CalcOffSet = [CalcOffSet GasTempOffSet];
else
GasSimTestSSE = 0;
GasTempOffSet = 0;
end
% Calculate errors in column temperature profiles
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ColTempSSEFile'));
ColTempSSEFile = GenIn.values(HeNu,ItNu,RecNum);
if ~strcmp(cell2mat(ColTempSSEFile),'none')
[ColSimTestSSE, ColTempOffSet] = ErrFun(time,out,ColTempSSEFile,ScopeErr);
NumScaleErrEnd = NumScaleErr + length(ColSimTestSSE);
SumSimTest = SumSimTest +
sum(ScaleErr(NumScaleErr+1:NumScaleErrEnd).*ColSimTestSSE);
NumScaleErr = NumScaleErrEnd + 1;
SumSimTest = SumSimTest +
ScaleErr(NumScaleErr)*sum(abs(ColTempOffSet))./HalfCycleLength;
CalcOffSet = [CalcOffSet ColTempOffSet];
else
ColSimTestSSE = 0;
ColTempOffSet = 0;
end
% Write sum of errors and OffSet to GUI, array
```

```
[HeNu,ItNu]=find(strcmp(GenIn.varname,'SimTestErr'));
GenIn.values{HeNu,ItNu,RecNum} = SumSimTest;
ch = findobj(fh,'Tag','SimTestErr');
set(ch,'String',SumSimTest);
[HeNu,ItNu]=find(strcmp(GenIn.varname,'CalcOffSet'));
GenIn.values{HeNu,ItNu,RecNum} = num2str(CalcOffSet);
ch = findobj(fh,'Tag','CalcOffSet');
set(ch,'String',num2str(CalcOffSet));
% Save current values to GenIn array
setappdata(fh,'AppGenIn',GenIn);
% Save error value to file for X-TOOLSS, also file to gather BT sharpening
% data
if strcmp('CommandLine',mode)
dlmwrite('Err.txt',SumSimTest);
end
[HeNu,ItNu]=find(strcmp(GenIn.varname,'AdsAxialDisp'));
AdsAxialDisp = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'LDF'));
LDF = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
% Determine maximum slope of concentration history for interior and
% breakthrough curve
[HeNu,ItNu]=find(strcmp(GenIn.varname,'SlopeMax'));
SlopeMax = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'Locations'));
Locations = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
IndStart = int16(SlopeMax(1)/100*Locations);
IndEnd = int16(SlopeMax(2)/100*Locations);
% if IndStart < 1; IndStart = 1; end
% if IndEnd > Locations; IndEnd = Locations; end
ctMaxInt = min(max(out(1).ct(3:end,IndStart:IndEnd)));
ctMaxBT = max(out(1).ct(3:end,end));
% Integration and difference of Inlet and Outlet moles
MolIn = out(1).c(:,1)*AdsSupVel*FreeFlowArea/10000;
TotMolIn = trapz(time,MolIn);
MolOut = out(1).c(:,end)*AdsSupVel*FreeFlowArea/10000;
TotMolOut = trapz(time,MolOut);
MolFlowTot = TotMolIn-TotMolOut;
% Integration of bed loading. Note MolSorbedTot is converted to
% mol sorbate / kg sorbent
if SorbentMass == 0; SorbentMass = SorbentMassChk; end
TotMolSorbInit = trapz(out(1).q(1,:))/Locations*SorbentMass/1000;
TotMolSorbEnd = trapz(out(1).q(end,:))/Locations*SorbentMass/1000;
MolSorbedTot = (TotMolSorbEnd-TotMolSorbInit)/PartDensity;
% Write data to text file for post-run analysis
RunData = [MolFlowTot MolSorbedTot SumSimTest ConcSimTestSSE ConcOffSet...
GasSimTestSSE GasTempOffSet ColSimTestSSE ColTempOffSet LDF...
AdsAxialDisp ctMaxInt ctMaxBT];
dlmwrite('RunData.csv',RunData,'-append');
end
function ExportParameters(hObject, eventdata)
% Export numeric data to text file, which can then be loaded into a COMSOL
% adsorption model Variables node
% Initiation code
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
RecNum = str2double(get(rh,'String'));
% Execute command for to update calculations
GenCalc(hObject, eventdata)
GenIn = getappdata(fh,'AppGenIn');
% Write data provided by current database record. Notes:
% (1) If the value is a string, it is put into the Description field and
% the COMSOL Expression field is set to 'text'
% (2) Otherwise, units are appended to the value for the COMSOL Expression
field
% (3) The Export Flag value determines if a field is exported
```

```
[file] = uiputfile('*.txt','Save in Tab-Delimited File As');
if file;
fid = fopen(file, 'wt');
for i = 1:size(GenIn.NumInputs,2)
for j = 1:GenIn.NumInputs(i)
if strcmp('yes',GenIn.export(i,j))
if iscellstr(GenIn.values(i,j))
values = char(GenIn.values(i,j,RecNum));
else
values = cell2mat(GenIn.values(i,j,RecNum));
end
if strcmp('text',GenIn.unit(i,j))
stringText = values;
values = 'text';
else
stringText = GenIn.string(i,j);
values = strcat(num2str(values),GenIn.unit(i,j));
end
fprintf(fid, '%s\t',char(GenIn.varname(i,j)),...
char(values),char(stringText));
fprintf(fid, '\n');
end
end
end
fclose(fid);
end
% end of function
function prop = gas_prop(gasnum)
% GAS_PROP function provides gas properties based on gas name
%
% Syntax: gas_prop(gasnum)
%
% Input: gasnum = number of gas of interest or formula
%
% Output: gas_prop is a structure array with gas properties required
% for viscosity and other calculations; field values as shown
% below
%
% Notes: Data is from appendix A, the Property Data Bank, of "Properties
% of Gases and Liquids" by Poling et. al., 5th Edition. Units are:
%
% Properties:
%
% number in Reid et. al. - No
% chemical formula - Formula
% gas name - Name
% molecular weight - Molwt, g/mol
% normal freezing point - Tfp, K
% normal boiling point (at 1 atm) - Tb, K
% critical temperature - Tc, K
% critical pressure - Pc, bar
% critical volume - Vc, cm^3/mole
% critical compressibility factor - Zc, PcVc/RTc
% Pitzer's acentric factor - Omega
% dipole moment - Dipm, debeyes
% CPVAPA, CPVAPB, CPVABC, CPVAPD - constants to calculate the isobaric
% heat capacity of the ideal gas, with Cp in J/(mol*K) and T in kelvins:
% Cp = CPVAPA + (CPVAPB)T + (CPVAPC)T^2 + (CPVAPD)T^3
% Molecular Diffusion Volumes for Use in Estimating Dab via the Method of
% Fuller, Schettler, and Giddings - cm^3/mole
%
% MATLAB version by J. Knox 10-25-05
%
if nargin<1
error('gas number required for gas_prop');
```

```
end
%
% --- Build gas properties database
%
gas_prop = struct(...
'No', { 455, 460, 440, 31, 447, 438, 1, 501, 450},...
'Formula',{'N2', 'O2', 'H2O', 'CO2','H3N','H2', 'Ar','Air','He'},...
'Name',{'Nitrogen','Oxygen','Water','Carbon Dioxide','Ammonia',...
'Hydrogen','Argon','Air','Helium'},...
'Molwt' ,{ 28.013, 31.999, 18.015, 44.010, 17.031, 2.016, 39.948, 28.850,
4.003},...
'Tfp' ,{ 63.83, 54.4, 273.15, 216.6, 0, 0, 83.8, 0, 2.15},...
'Tb' ,{ 77.4, 90.2, 373.3, 0, 0, 0, 87.3, 0, 4.30},...
'Tc' ,{126.2, 154.6, 647.3, 304.1, 405.5, 33.2, 150.8, 132.6, 5.19},...
'Pc' ,{ 33.9, 50.4, 221.2, 73.8, 113.5, 13.0, 48.7, 37.37, 2.27},...
'Vc' ,{ 89.8, 73.4, 57.1, 93.9, 72.5, 65.1, 74.9, 0, 57.3},...
'Zc' ,{ 0.290, 0.288, 0.235, 0.274, 0.244, 0.306, 0.291, 0, 0.301},...
'Omega' ,{ 0.039, 0.025, 0.344, 0.239, 0, 0, 0.001, 0, -3.90},...
'Dipm' ,{ 0.0, 0.0, 1.8, 0.0, 1.47, 0, 0, 0, 0},...
'CPVAPA',{ 3.115e+1, 2.811e+1, 3.194e+1, 1.980e+1, 0, 0, 2.080e+1, 0, 0},...
'CPVAPB',{-1.357e-2, -3.680e-6, 1.436e-3, 7.344e-2, 0, 0, 0, 0, 0},...
'CPVAPC',{ 2.680e-5, 1.746e-5, 2.432e-5, -5.602e-5, 0, 0, 0, 0, 0},...
'CPVAPD',{-1.168e-8, -1.065e-8, -1.176e-8, 1.715e-8, 0, 0, 0, 0, 0},...
'Sigma' ,{18.5, 16.3, 13.1, 26.9, 20.7, 6.12, 16.2, 19.7 2.67});
% Check for formula
if ischar(gasnum)
switch gasnum
case 'N2'
prop = gas_prop(1);
case 'O2'
prop = gas_prop(2);
case 'H2O'
prop = gas_prop(3);
case 'CO2'
prop = gas_prop(4);
case 'H3N'
prop = gas_prop(5);
case 'H2'
prop = gas_prop(6);
case 'Ar'
prop = gas_prop(7);
case 'Air'
prop = gas_prop(8);
case 'He'
prop = gas_prop(9);
end
else
switch gasnum
case 455
prop = gas_prop(1);
case 460
prop = gas_prop(2);
case 440
prop = gas_prop(3);
case 31
prop = gas_prop(4);
case 447
prop = gas_prop(5);
case 438
prop = gas_prop(6);
case 1
prop = gas_prop(7);
case 501
prop = gas_prop(8);
case 450
```

```
prop = gas_prop(9);
end
end
function GenCalc(hObject, eventdata)
% Perform calculations and conversions on input data
%!!! change in graphics broke this. Fixing for interactive mode only at
% present time.
% % Interactive mode:
% % eventdata is 0 or empty
% % hObject is GUI handle created by Matlab
%
% % Command line mode:
% % eventdata contains RecNum
% % hObject is passed figure handle
%
% if isempty(eventdata)|| eventdata == 0
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
RecNum = str2double(get(rh,'String'));
% elseif eventdata > 0
% RecNum = eventdata;
% fh = hObject;
% end
% GenIn = getappdata(fh,'AppGenIn'); (moved below to get calc updates prior
% to modifying working capacity)
% Execute command for Adsorption and Desorption Calculations.
% % For interactive mode, 0.99 is passed as positive for adsorption and
% % passed as a negative number for desorption
% % For command line mode, RecNum is passed as positive for adsorption and
% % passed as a negative number for desorption
% if isempty(eventdata)|| eventdata == 0
Calc(hObject,0.99)
Calc(hObject,-0.99)
% elseif eventdata > 0
% Calc(hObject,RecNum)
% Calc(hObject,-RecNum )
% end
% Update GenIn array in local scope prior to updating
GenIn = getappdata(fh,'AppGenIn');
% Calculation of Equilibrium Working Capacity
% Equil Capacity [mol/kg] EquilCap
[HeNu,ItNu]=find(strcmp(GenIn.varname,'EquilCap'));
if ~isempty([HeNu,ItNu])
ch = findobj(fh,'Tag','AdsBedLoad');
AdsBedLoad = str2double(get(ch,'String'));
ch = findobj(fh,'Tag','DesBedLoad');
DesBedLoad = str2double(get(ch,'String'));
EquilCap = AdsBedLoad - DesBedLoad;
GenIn.values{HeNu,ItNu,RecNum} = EquilCap;
ch = findobj(fh,'Tag','EquilCap');
set(ch,'String',EquilCap);
end
% Save current values to AppGenIn array
setappdata(fh,'AppGenIn',GenIn);
% end of function
function GenDel(hObject, eventdata)
% Delete Current Record and prompt for record to load
delete = questdlg('Deleting a record cannot be undone. Continue?',...
'Delete Record?','Delete Record','Cancel','Cancel');
switch delete
case 'Delete Record'
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
GenIn = getappdata(fh,'AppGenIn');
RecNum = str2double(get(rh,'String'));
```

```
RecEnd = size(GenIn.values,3);
if RecNum == 1
GenIn.values = GenIn.values(:,:,2:end);
elseif RecNum == RecEnd
GenIn.values = GenIn.values(:,:,1:RecNum-1);
else
GenIn.values = [GenIn.values(1:RecNum-1) GenIn.values(RecNum+1:end)];
end
setappdata(fh,'AppGenIn',GenIn);
GenLoad(hObject, eventdata)
end
% end of function
function GenDup(hObject, ~)
% Load GenIn and increment max record number for duplicate
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
GenIn = getappdata(fh,'AppGenIn');
RecNum = size(GenIn.values,3);
GenIn.values(:,:,RecNum+1)=GenIn.values(:,:,RecNum);
% GenIn length just increased by 1
NumRec = RecNum+1;
set(rh,'String',NumRec);
setappdata(fh,'AppGenIn',GenIn);
% end of function
function [hObject] = Generic(GenIn)
%GENERIC function to create a GUI based on input specifications
% Usage:
% The Generic function provides a means to create a GUI based on a array
% input from a user that specifies number of text or numeric
% record locations. The user specifies a description and a variable name
% for each record. The description is placed on the GUI and the variable
% name is mapped to the edit uicontrol.
%
% The GUI also provides a File menu to save a database with the
% specified information, as well as load an existing database. A Record
% menu item allows for new and duplicate records, and deleting of
% records.
%
% Inputs:
% GenIn is a structured array providing the data to construct a
% GUI and store the records generated by entering data in the
% GUI. The array is created via a Excel or text import function, xlGenIn
% or (TBD text import program).
%
% Author: Jim Knox
% Date: 4/8/2010
% Revision: 10/14/2010
% Last record is used to populate GUI
RecNum = size(GenIn.values,3);
MaxHeightW = 950; %Final value is calculated from inputs
% Calculate Window Size, Locations for each heading
ColNum = 1; % Current Column Location,
for i = 1:numel(GenIn.heading(1,:))
if i == 1
StartHeight(i) = MaxHeightW - 25;
ColHt(ColNum) = 25*(GenIn.NumInputs(i)+1);
else
StartHeight(i) = StartHeight(i-1) - 25*(GenIn.NumInputs(i-1)+1);
ColHt(ColNum) = ColHt(ColNum) + 25*(GenIn.NumInputs(i)+1);
end
% Check for next heading dropping off GUI window; increment ColNum
% and reset StartHeight if so. Recalculate window height
NxtSrtHt = StartHeight(i) - 25*(GenIn.NumInputs(i)+1);
if NxtSrtHt < 25
ColHt(ColNum) = MaxHeightW - StartHeight(i) - 25;
```

```
ColNum = ColNum + 1;
StartHeight(i) = MaxHeightW - 25; %#ok<*AGROW>
ColHt(ColNum) = 25*(GenIn.NumInputs(i)+1);
end
StartWidth(i) = 5 + (ColNum-1)*350;
end
WidthW = ColNum*355 - 25;
HeightW = max(ColHt) + 5;
StartHeight = StartHeight - (MaxHeightW - HeightW);
% Draw figure
Gen.fh = figure('units','pixels',...
'position',[100 100 WidthW HeightW],...
'menubar','none',...
'numbertitle','off',...
'name',GenIn.Title,...
'Color',[0.7,0.7,0.7],...
'resize','on');
% Draw "Record Number" text
Gen.Rec_Number_text = uicontrol('style','text',...
'units','pixels','position',[190 StartHeight(1) 110 25],...
'fontname','Arial','fontsize',11,'FontWeight','normal',...
'ForegroundColor',[0,0,0],'BackgroundColor',[0.7,0.7,0.7],...
'HorizontalAlignment','left','string','Record Number');
% Draw Record Number value
Gen.Rec_Number = uicontrol('style','text',...
'units','pixels','position',[315 StartHeight(1) 20 25],...
'fontname','Arial','fontsize',11,'FontWeight','normal',...
'ForegroundColor',[0,0,0],'BackgroundColor',[0.7,0.7,0.7],...
'HorizontalAlignment','left','string',RecNum,'Tag','Rec_Number');
for i = 1:numel(GenIn.heading(1,:))
% Draw heading text
HeadWidth = 250;
if i == 1 ; HeadWidth = 185; end;
Gen.head = uicontrol('style','text',...
'units','pixels','position',[StartWidth(i) StartHeight(i) HeadWidth 25],...
'fontname','Arial','fontsize',11,'FontWeight','bold',...
'ForegroundColor',[0,0,0],'BackgroundColor',[0.7,0.7,0.7],...
'HorizontalAlignment','left','string',GenIn.heading(i));
% Draw string text
for j = 1: GenIn.NumInputs(i)
if strcmp('text',GenIn.unit(i,j))
stringText = GenIn.string(i,j);
else
stringText = strcat(GenIn.string(i,j),GenIn.unit(i,j));
end
Gen.text = uicontrol('style','text',...
'units','pixels','position',[StartWidth(i) StartHeight(i)-(j*25) 150 25],...
'fontname','Arial','fontsize',10,'FontWeight','normal',...
'ForegroundColor',[0,0,0],'BackgroundColor',[0.7,0.7,0.7],...
'HorizontalAlignment','left','string',stringText);
% Draw values for edit fields and enter varname (DynName) and TooltipString
DynName = char(GenIn.varname(i,j));
TooltipString = char(GenIn.tooltip(i,j));
BckCol = 0.7;
if strcmp(GenIn.export(i,j),'yes'); BckCol = 0.9; end
if strcmp(GenIn.export(i,j),'sum'); BckCol = 0.6; end
UserData.i = i;
UserData.j = j;
Gen.(DynName) = uicontrol('style','edit',...
'units','pixels','position',[StartWidth(i)+150 StartHeight(i)-(j*25) 175
25],...
'fontname','Arial','fontsize',11,'FontWeight','normal',...
'ForegroundColor',[0,0,0],'BackgroundColor',[0.7,0.7,BckCol],...
'HorizontalAlignment','left','string',GenIn.values(i,j,RecNum),...
'Callback',makeHandle('EditCallback'),'Tag',DynName,...
'UserData',UserData,'TooltipString',TooltipString);
end
end
% Store GenIn array in appdata for access in functions
setappdata(Gen.fh,'AppGenIn',GenIn);
% Create menus for loading and saving database.mat files, and manipulating
% records (new, duplicate, load, delete)
fmh = uimenu(Gen.fh,'Label','File');
uimenu(fmh,'Label','Open File','Callback',@GenOpen);
uimenu(fmh,'Label','Export Comsol Parameters
File','Callback',@ExportParameters);
uimenu(fmh,'Label','Save Excel File','Callback',@GenSaveXL);
uimenu(fmh,'Label','Save Tab-Delimited File','Callback',@GenSaveTab);
uimenu(fmh,'Label','Save GenIn Array','Callback',@GenSaveMat);
uimenu(fmh,'Label','Quit','Callback',@GenQuit);
rmh = uimenu(Gen.fh,'Label','Record');
uimenu(rmh,'Label','New Record','Callback',@GenNew);
uimenu(rmh,'Label','Load Record','Callback',@GenLoad);
uimenu(rmh,'Label','Duplicate Record','Callback',@GenDup);
uimenu(rmh,'Label','Delete Record','Callback',@GenDel);
cmh = uimenu(Gen.fh,'Label','Calculations');
uimenu(cmh,'Label','Perform Calculations','Callback',@GenCalc);
UserDataCheck.AutoCalc = 'on';
uimenu(cmh,'Label','Auto Calc','Callback',@ToggleAutoCalc,'Checked','on',...
'UserData',UserDataCheck);
commh = uimenu(Gen.fh,'Label','COMSOL');
uimenu(commh,'Label','Run Simulation','Callback',@RunSimulation);
uimenu(commh,'Label','Plot Data in RunID Workspace','Callback',@PlotOnly);
UserDataCheck.AutoPlot = 'on';
uimenu(commh,'Label','Auto Plot','Callback',@ToggleAutoPlot,'Checked','on',...
'UserData',UserDataCheck);
UserDataCheck.AutoSaveMat = 'on';
uimenu(commh,'Label','Auto Save .mat
file','Callback',@ToggleAutoSaveMat,'Checked','on',...
'UserData',UserDataCheck);
hObject = Gen.fh;
function GenLoad(hObject, ~)
% Load GenIn and check for max record number
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
GenIn = getappdata(fh,'AppGenIn');
NumRec = size(GenIn.values,3);
% Prompt for desired record number from 1 to max
prompt = {['Enter Record Number up to ' num2str(NumRec)]};
dlg_title = 'Load Record';
num_lines = 1;
def = {'1'};
Rec = inputdlg(prompt,dlg_title,num_lines,def);
RecNum = str2double(char(Rec));
% If user clicked "Cancel", RecNum is NaN
if isnan(RecNum) == 0
% Import data from RecNum to GUI
for i = 1:numel(GenIn.heading(1,:))
for j = 1: GenIn.NumInputs(i)
DynName = char(GenIn.varname(i,j));
% % DynName = ['edit' num2str(i) num2str(j)];
% % UserData.i = i;
% % UserData.j = j;
eh = findobj(fh,'Tag',DynName);
set(eh,'String',GenIn.values(i,j,RecNum))
end
end
set(rh,'String',RecNum);
end
% Update Calculations
```

```
GenCalc(hObject, 0)
setappdata(fh,'AppGenIn',GenIn);
% end of function
function GenNew(hObject, ~)
% Create New record after existing records
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
GenIn = getappdata(fh,'AppGenIn');
RecNum = size(GenIn.values,3);
GenIn.values(:,:,RecNum+1)=GenIn.values(:,:,RecNum);
% GenIn.values length just increased by 1
NumRec = RecNum+1;
set(rh,'String',NumRec);
% Put blank entries in all edit fields
for i = 1:numel(GenIn.heading(1,:))
for j = 1: GenIn.NumInputs(i)
DynName = char(GenIn.varname(i,j));
% % DynName = ['edit' num2str(i) num2str(j)];
% % UserData.i = i;
% % UserData.j = j;
eh = findobj(fh,'Tag',DynName);
set(eh,'String','')
end
end
setappdata(fh,'AppGenIn',GenIn);
% end of function
function GenOpen(hObject, eventdata)
% Execute menu command to open user selected Excel file and convert to "GenIn"
% structure array. Opens a second input window.
[GenIn] = xl2GenIn;
% Execute command create the GUI
% Generic(GenIn,GenIn.Title)
Generic(GenIn)
% Update calculations
GenCalc(hObject, eventdata)
% end of GenOpen function
function GenQuit(hObject, ~)
% Remind user to save current files or lose changes
quit = questdlg('Unsaved changes will be lost. Continue?',...
'Quit?','Quit','Cancel','Cancel');
switch quit
case 'Quit'
fh = ancestor(hObject,'figure');
delete(fh)
case 'Cancel'
%do nothing
end
% end of function
function GenSaveMat(hObject, ~)
% Save Current Array in .mat file
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number'); %#ok<NASGU>
GenIn = getappdata(fh,'AppGenIn'); %#ok<NASGU>
[file] = uiputfile('*.mat','Save Workspace As');
if file; save(file, 'GenIn'); end
% end of function
function GenSaveTab(hObject, ~)
% Save Current File as Tab delimited text file
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number'); %#ok<NASGU>
GenIn = getappdata(fh,'AppGenIn');
[file] = uiputfile('*.txt','Save in Tab-Delimited File As');
if file;
NumRec = size(GenIn.values,3);
fid = fopen(file, 'wt');
```

```
fprintf(fid,'Description\tVariable Name\tTootip Text\tUnits\tExport Flag\t');
for i = 1:NumRec;
fprintf(fid,'%s\t',int2str(i));
end
fprintf(fid,'\n');
for i = 1:size(GenIn.NumInputs,2)
fprintf(fid, '%s\n', char(GenIn.heading(i)));
for j = 1:GenIn.NumInputs(i)
fprintf(fid, '%s\t',char(GenIn.string(i,j)),...
char(GenIn.varname(i,j)),char(GenIn.tooltip(i,j)),...
char(GenIn.unit(i,j)),char(GenIn.export(i,j)));
for k = 1:NumRec
if iscellstr(GenIn.values(i,j,k))
fprintf(fid,'%s\t',char(GenIn.values(i,j,k)));
else
fprintf(fid,'%g\t',cell2mat(GenIn.values(i,j,k)));
end
end
fprintf(fid, '\n');
end
fprintf(fid, '\n');
end
fclose(fid);
end
% end of function
function GenSaveXL(hObject, ~)
% Save Current File as Excel file
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number'); %#ok<NASGU>
GenIn = getappdata(fh,'AppGenIn');
[file] = uiputfile('*.xlsx','Save in Excel File As');
if file;
NumRec = size(GenIn.values,3);
%r% fid = fopen(file, 'wt');
%r% fprintf(fid,'Description\tVariable Name\tTootip Text\tUnits\tExport
Flag\t');
% Header Line with Fields only
CellXL = {'Description','Variable Name','Tootip Text','Units','Export Flag'};
for i = 1:NumRec;
%r% fprintf(fid,'%s\t',int2str(i));
CellXL(1,i+5) = num2cell(i);
end
%r% fprintf(fid,'\n');
%d% CellXL(1,2) = GenIn.Heading(1);
HeadingRow(1) = 2;
for i = 1:size(GenIn.NumInputs,2)
%r5 fprintf(fid, '%s\n', char(GenIn.heading(i)));
if i > 1; HeadingRow(i) = HeadingRow(i-1)+GenIn.NumInputs(i-1)+2; end
CellXL(HeadingRow(i),1) = GenIn.heading(i);
for j = 1:GenIn.NumInputs(i)
%r% fprintf(fid, '%s\t',char(GenIn.string(i,j)),...
%r% char(GenIn.varname(i,j)),char(GenIn.tooltip(i,j)),...
%r% char(GenIn.unit(i,j)),char(GenIn.export(i,j)));
CellXL(HeadingRow(i)+j,1) = GenIn.string(i,j);
CellXL(HeadingRow(i)+j,2) = GenIn.varname(i,j);
CellXL(HeadingRow(i)+j,3) = GenIn.tooltip(i,j);
CellXL(HeadingRow(i)+j,4) = GenIn.unit(i,j);
CellXL(HeadingRow(i)+j,5) = GenIn.export(i,j);
for k = 1:NumRec
if iscellstr(GenIn.values(i,j,k))
%r% fprintf(fid,'%s\t',char(GenIn.values(i,j,k)));
CellXL(HeadingRow(i)+j,k+5)=GenIn.values(i,j,k);
else
%r% fprintf(fid,'%g\t',cell2mat(GenIn.values(i,j,k)));
CellXL(HeadingRow(i)+j,k+5)=GenIn.values(i,j,k);
```

```
end
end
%r% fprintf(fid, '\n');
end
%r% fprintf(fid, '\n');
end
xlswrite(file,CellXL,'Sheet1');
%r% fclose(fid);
end
% end of function
function DataSet = importXLSsheets(FileToRead)
% importXLSsheets(FileToRead) imports data from the specified EXCEL file
% into structured array for use in correlation development. Each sheet
% contains a dataset including description and test data. Organization is
% as follows:
% Column A: field names for dataset description
% Column B: dataset description text
% Column C: abbreviated description for entry title for local ref, not
% input
% Column D to last column used: Test data, with field name for data column
% in the first row.
% Retrieve the sheet name strings. 'dummy' is dummy variable.
[dummy, names]=xlsfinfo(FileToRead);
% Setup loop based on number of sheets
[dummy NumSheets] = size(names);
% Preallocate structured array (minimal data to satisfy M-lint)
DataSet(NumSheets) = struct('Name','text');
% Get sheet content and size; populate structured array
for i = 1:NumSheets;
NameSheet = char(names(i)); % Sheet Names
[data,text] = xlsread(FileToRead,NameSheet); % Get data and text
[NumDesc NumCol] = size(text); % Get # of text items
[NumRow NumDataCol] = size(data); % Get # of data items
if NumCol ~= NumDataCol
error('unexpected file format')
end
DataSet(i).Name = ''; % Initialize DataSet name
% Put text in DataSet
for j = 2:NumDesc;
DataSet(i).(text{j,1}) = text{j,2};
% Create abbreviated entry name
nospace = '';
if (j < 5);
for k = 1:size(text{j,3},2);
if isspace(text{j,3}(k));
% Do not include spaces in name
else
nospace = [nospace text{j,3}(k)];
end
end
DataSet(i).Name = [DataSet(i).Name nospace '_'];
end
end
% Date is last description to be entered. Add date and time. Expected
% format is DD MM YY HH:MM; convert to DD_MM_YY
NameEnd = size(DataSet(i).Name,2);
DataSet(i).Name(NameEnd+1:NameEnd+8) = text{NumDesc,2}(1:8);
DataSet(i).Name(NameEnd+3) = '_';
DataSet(i).Name(NameEnd+6) = '_';
% Put data in Dataset
for m = 1:NumRow;
for k = 4:NumCol;
DataSet(i).(text{1,k})(m,1) = data(m,k);
end
end
```

```
% % Unwrap datasets from each Excel tab to named dataset
% setname = [DataSet(1,i).Name '=DataSet(1,i)'];
% eval(setname);
end
function fh = makeHandle(funcname)
fh = str2func(funcname);
% Function to create function handle
function [Output] = ParseInput(InputText)
% Function to parse space delimited text to individual floating point values.
% Original commented out since limited in number of values.
% InputString = cell2mat(InputText);
% %First Value
% [first remainder1] = strtok(InputString);
% if isfloat(first)
% Output(1) = first;
% else
% Output(1) = str2double(first);
% %Second Value
% [second remainder2] = strtok(remainder1);
% if ~strcmp(second,'')
% Output(2) = str2double(second);
% %Third Value
% [third remainder3] = strtok(remainder2);
% if ~strcmp(third,'')
% Output(3) = str2double(third);
% % Fourth Value
% fourth = strtok(remainder3);
% if ~strcmp(fourth,'')
% Output(4) = str2double(fourth);
% end
% end
% end
% end
%Get Values
i = 1;
InputString = cell2mat(InputText);
% Spooky Matlab behavior - if the input is 10, strtok returns an empty
% matrix. So here I change the double to string to avoid that.
if isfloat(InputString)
InputString = num2str(InputString);
end
[first remainder] = strtok(InputString);
if isfloat(first)
Output(i) = first;
else
Output(i) = str2double(first);
end
while ~strcmp(remainder,'')
i = i + 1;
InputString = remainder;
[first remainder] = strtok(InputString);
if isfloat(first)
Output(i) = first;
else
Output(i) = str2double(first);
end
end
% Last Value is NaN; trim off
if isnan(Output(end))
Output = Output(1:end-1);
end
function [Output] = ParseInputText(InputText)
% Function to parse space delimited text into up to seven text cells
% %First Value
% [first remainder1] = strtok(InputText);
% Output(1) = first;
% %Second Value
% [second remainder2] = strtok(remainder1);
% if ~strcmp(second,'')
% Output(2) = (second);
% %Third Value
% [third remainder3] = strtok(remainder2);
% if ~strcmp(third,'')
% Output(3) = (third);
% % Fourth Value
% [fourth remainder4]= strtok(remainder3);
% if ~strcmp(fourth,'')
% Output(4) = (fourth);
% end
% end
% end
% end
%Get Values
i = 1;
[Output(1), remainder] = strtok(InputText);
while ~strcmp(remainder,'')
i = i + 1;
InputText = remainder;
[Output(i), remainder] = strtok(InputText); %#ok<AGROW>
end
function [time, out] = PDE_Adsorb(GenIn,RecNum)
% COMSOL function for Parametric Runs of Axial Dispersion and LDF as well
% as general model runs
%
% Model exported on Aug 8 2015, 15:36 by COMSOL 5.1.0.180.
% *** Start COMSOL statements ************************************
import com.comsol.model.*
import com.comsol.model.util.*
model = ModelUtil.create('Model');
% model.modelPath('D:\James Knox\My Documents\COMSOL\2015\COMSOL and
Matlab\Update 2012
code');
% model.label('PDE_940126_08082015.mph');
% model.comments(['PDE 940126\n\n']);
% The following Parametric values setting commands are replaced by the
% command following the comment "Call to set COMSOL parameters"
% model.param.set('Description', 'Simulation', ['and Comparison with 01-26-94
Test: CO2 Adsorption on
5A in a 2-in Column' native2unicode(hex2dec({'00' '09'}), 'unicode')
'Description']);
% model.param.set('AdsInitTemp', '25.18[degC]', 'Ads Initial Temp');
% model.param.set('AdsInTemp', '25.18[degC]', 'Ads Inlet Temp');
% model.param.set('AdsInitConc', '0.001[mol/m^3]', 'Ads Initial Conc');
% model.param.set('AdsInitLoad', '1[mol/m^3]', 'Ads Initial Load');
% model.param.set('SorbGasMW', '0.044[kg/mol]', 'Sorbate MolWt');
% model.param.set('FreeFlowArea', '17.8139[cm^2]', 'Free Flow Area');
% model.param.set('CanCSArea', '2.4544[cm^2]', 'Canister CS Area');
% model.param.set('CanIPerim', '14.96[cm]', 'Can Inner Perimeter');
% model.param.set('CanOPerim', '15.96[cm]', 'Can Outer Perimeter');
% model.param.set('BedLength', '0.254[m]', 'Bed Length');
% model.param.set('VoidFraction', '0.35', 'Void Fraction');
% model.param.set('WallVoid', '1', 'Wall Void Fraction');
% model.param.set('CanCond', '16.8[W/(m*K)]', 'Can Cond');
% model.param.set('CanHeatCap', '475[J/(kg*K)]', 'Can Q Capac');
% model.param.set('CanDensity', '7833[kg/m^3]', 'Can Density');
% model.param.set('AmbTemp', '25.18[degC]', 'Ambient Temp');
% model.param.set('CanAmbH', '15[W/(m^2*K)]', 'Can-Amb H');
% model.param.set('PartDensity', '1180[kg/m^3]', 'Part Density');
% model.param.set('LDF', '0.0023[1/s]', 'Mass Trans Coeff');
% model.param.set('SorbCond', '0[W/(m*K)]', 'Sorb Q Cond');
% model.param.set('SorbHeatCap', '1046.7[J/(kg*K)]', 'Sorb Q Capac');
% model.param.set('dHSorb', '-44.4[kJ/mol]', 'Heat of Ads');
% model.param.set('HalfCycleLength', '6990', 'Half-Cycle Length[s]');
% model.param.set('TimeStep', '30', 'Time Step[s]');
% model.param.set('NodeSepMax', 'Physics[m]', 'Node Sep Max');
% model.param.set('NodeSepInit', '0.0001[m]', 'Node Sep Init');
% model.param.set('AdsConc', '0.3295[mol/m^3]', 'Ads Concentrat');
% model.param.set('AdsGasHeatCap', '1.04[kJ/(kg*K)]', 'Gas Q Cap');
% model.param.set('AdsAxiCond', '0.51376[W/(m*K)]', 'Axial Cond');
% model.param.set('AdsSorbGasHeatCap', '1[kJ/(kg*K)]', 'Sorbate Q Cap');
% model.param.set('AdsSorbGasH', '111.1016[W/(m^2*K)]', 'Sorb-Gas H');
% model.param.set('AdsGasCanH', '11.5514[W/(m^2*K)]', 'Gas-Can H');
% model.param.set('AdsAxialDisp', '0.00073176[m^2/s]', 'Ads Axial Disp');
% model.param.set('AdsTotPress', '105.2487[kPa]', 'Ads Total Press');
% model.param.set('AdsGasDens', '1.1939[kg/m^3]', 'Ads Gas Dens');
% model.param.set('AdsSupVel', '0.27584[m/s]', 'Ads Superfic Vel');
% model.param.set('EqPelDia', '2[mm]', 'Equiv Pellet Dia');
% model.param.set('AreaVolRat', '216.6667[1/m]', 'Area to Vol ratio');
% model.param.set('TothA0', '9.875e-07[mol/kg/kPa]', 'Toth a0');
% model.param.set('TothB0', '6.761e-08[1/kPa]', 'Toth b0');
% model.param.set('TothE', '5625[K]', 'Toth E');
% model.param.set('TothT0', '0.27', 'Toth to');
% model.param.set('TothC', '-20.02[K]', 'Toth c');
%
% *** End COMSOL Statements ************************************
% Load output variables into character arrays to be input into param
% statements.
% Data provided by current database record is searched for needed local
% variables and then for COMSOL parameters. Notes:
% (1) The Export Flag value determines if a field is exported
% (2) If the value is a string, it is not sent to COMSOL
% (3) Units are appended to the value for the COMSOL Expression field
for i = 1:size(GenIn.NumInputs,2)
for j = 1:GenIn.NumInputs(i)
% % % Pick out needed values for plotting, etc. from GenIn array
switch char(GenIn.varname(i,j))
case 'ModelName'
ModelNSolverName = ParseInputText(GenIn.values(i,j,RecNum));
ModelSolve = cell2mat(ModelNSolverName(2));
case 'NodeSepMax'
NodeSepMax=cell2mat(GenIn.values(i,j,RecNum));
case 'BedLength'
BedLength=cell2mat(GenIn.values(i,j,RecNum));
case 'TimeStep'
TimeStep=cell2mat(GenIn.values(i,j,RecNum)); %#ok<NASGU>
case 'HalfCycleLength'
HalfCycleLength=cell2mat(GenIn.values(i,j,RecNum)); %#ok<NASGU>
case 'Locations'
Locations=cell2mat(GenIn.values(i,j,RecNum));
case 'InTemp'
InTemp=cell2mat(GenIn.values(i,j,RecNum));
case 'TimeLineFile'
TimeLineFile=cell2mat(GenIn.values(i,j,RecNum));
end
if strcmp('yes',GenIn.export(i,j))
if ~iscellstr(GenIn.values(i,j,RecNum))
num = cell2mat(GenIn.values(i,j,RecNum));
values = strcat(num2str(num),GenIn.unit(i,j));
% Call to set COMSOL parameters
model.param.set(GenIn.varname(i,j),values,GenIn.string(i,j));
end
end
end
end
% *** Start COMSOL statements ************************************
```

```
model.modelNode.create('mod1');
model.modelNode('mod1').label('Model 1');
model.modelNode('mod1').defineLocalCoord(false);
model.file.clear;
model.func.create('step1', 'Step');
model.func.create('step2', 'Step');
% *** Start Custom Statements ***************************************
%model.func.create('int1', 'Interpolation');
if strcmp(InTemp,'Constant') || strcmp(InTemp,'Therm_CST')
% do nothing
elseif strcmp(InTemp,'Timelined') || strcmp(InTemp,'Therm_TL')
model.func.create('int1', 'Interpolation');
else
errordlg('Specify Inlet Temperature Control','Fatal Error');
end
% Test data comparisons functions are commented out below
% *** End Custom Statements ***************************************
% model.func.create('int1', 'Interpolation');
% model.func.create('int2', 'Interpolation');
% model.func.create('int3', 'Interpolation');
% model.func.create('int4', 'Interpolation');
% model.func.create('int5', 'Interpolation');
model.func('step1').set('funcname', 'isostep');
model.func('step1').set('smooth', '0.01');
model.func('step1').set('location', '0.005');
model.func('step2').set('funcname', 'initstep');
model.func('step2').set('smooth', '10');
model.func('step2').set('location', '5');
model.geom.create('geom1', 1);
% model.mesh.create('mesh1', 'geom1');
% model.mesh.create('mesh2', 'geom1');
model.geom('geom1').create('i1', 'Interval');
model.geom('geom1').feature('i1').set('p2', 'BedLength');
model.geom('geom1').run;
model.variable.create('var1');
model.variable('var1').model('mod1');
model.variable('var1').set('Pgas', 'max(1e-
99,isostep(c*1[m^3/mol])*c*R_const*Tg)');
model.variable('var1').set('a_Toth', 'TothA0*exp(TothE/Tg)');
model.variable('var1').set('b_Toth', 'TothB0*exp(TothE/Tg)');
model.variable('var1').set('t_Toth', 'TothT0+TothC/Tg');
model.variable('var1').set('q_star',
'PartDensity*a_Toth*Pgas/(1+(b_Toth*Pgas)^t_Toth)^(1/t_Toth)',
'Adsorption Equilibrium Loading of Gas Concentration');
model.variable('var1').set('InterVel', 'AdsSupVel/VoidFraction*Tg/AdsInTemp');
model.variable('var1').set('GasDensity', 'AdsGasDens*AdsInTemp/Tg');
model.physics.create('g', 'GeneralFormPDE', 'geom1');
model.physics('g').identifier('Mass_Balance');
model.physics('g').field('dimensionless').component({'c' 'q'});
model.physics('g').create('flux1', 'FluxBoundary', 0);
model.physics('g').feature('flux1').selection.set([1]); %#ok<*NBRAK>
model.physics('g').create('cons1', 'Constraint', 0);
model.physics('g').feature('cons1').selection.set([2]);
model.physics.create('phys1', 'GeneralFormPDE', 'geom1');
model.physics('phys1').identifier('Thermal_Balance');
model.physics('phys1').field('dimensionless').field('T');
model.physics('phys1').field('dimensionless').component({'Ts' 'Tg' 'Tw'});
model.physics('phys1').create('flux1', 'FluxBoundary', 0);
model.physics('phys1').feature('flux1').selection.set([1]);
model.physics('phys1').create('cons1', 'Constraint', 0);
model.physics('phys1').feature('cons1').selection.set([2]);
model.physics('phys1').create('dir1', 'DirichletBoundary', 0);
model.physics('phys1').feature('dir1').selection.set([1]);
% *** Start Custom Statements ***************************************
if strcmp('Physics',NodeSepMax)
```

```
model.mesh.create('mesh1', 'geom1');
model.mesh('mesh1').feature.create('edg1', 'Edge');
else
model.mesh.create('mesh1', 'geom1');
model.mesh('mesh1').feature.create('size1', 'Size');
model.mesh('mesh1').feature('size1').selection.geom('geom1', 0);
model.mesh('mesh1').feature('size1').selection.set([1]);
model.mesh('mesh1').feature.create('edg1', 'Edge');
end
if strcmp(InTemp,'Constant') || strcmp(InTemp,'Therm_CST')
% do nothing
elseif strcmp(InTemp,'Timelined') || strcmp(InTemp,'Therm_TL')
model.func('int1').set('argunit', 's');
model.func('int1').set('filename', TimeLineFile);
model.func('int1').set('source', 'file');
model.func('int1').set('funcs', {'TCintemp' '1'});
model.func('int1').set('fununit', 'K');
else
errordlg('Specify Inlet Temperature Control','Fatal Error');
end
% *** End Custom Statements ****************************************
model.physics('g').label('PDE 1');
model.physics('g').prop('ShapeProperty').set('boundaryFlux', '0');
model.physics('g').prop('Units').set('DependentVariableQuantity',
'concentration');
model.physics('g').prop('Units').set('CustomSourceTermUnit', 'mol/m^3/s');
model.physics('g').feature('gfeq1').set('f', {'-InterVel*cx'; 'LDF*(q_star-
q)'});
model.physics('g').feature('gfeq1').set('Ga', {'-AdsAxialDisp*cx'; '0'});
model.physics('g').feature('gfeq1').set('da', {'1'; '0'; '(1-
VoidFraction)/VoidFraction'; '1'});
model.physics('g').feature('init1').set('c', 'AdsInitConc');
model.physics('g').feature('init1').set('q', 'AdsInitLoad');
model.physics('g').feature('flux1').set('g',
{'initstep(t*1[1/s])*InterVel*(AdsConc-c)'; '0'});
% model.physics('g').feature('cons1').set('R', {'cx'; '0'});
model.physics('g').feature('cons1').set('R', {'d(c,x)'; 'd(q,x)'});
model.physics('g').feature('cons1').set('constraintType',
'unidirectionalConstraint');
model.physics('g').feature('cons1').label('Constraint 2');
model.physics('phys1').label('PDE 2');
model.physics('phys1').prop('ShapeProperty').set('boundaryFlux', '0');
model.physics('phys1').prop('Units').set('DependentVariableQuantity',
'temperature');
model.physics('phys1').prop('Units').set('CustomSourceTermUnit', 'W/m');
model.physics('phys1').feature('gfeq1').set('f',
{'FreeFlowArea*AreaVolRat*AdsSorbGasH*(Tg-Ts)-(1-
VoidFraction)*FreeFlowArea*dHSorb*d(q,t)'; '-
VoidFraction*FreeFlowArea*GasDensity*InterVel*AdsGasHeatCap*d(Tg,x)
+FreeFlowArea*AreaVolRat*AdsSorbGasH*(Ts-Tg)+WallVoid*CanIPerim*AdsGasCanH*(Tw-
Tg)';
'WallVoid*CanIPerim*AdsGasCanH*(Tg-Tw)+CanOPerim*CanAmbH*(AdsInTemp-Tw)'});
model.physics('phys1').feature('gfeq1').set('Ga', {'-(1-
VoidFraction)*FreeFlowArea*SorbCond*d(Ts,x)'; '-
VoidFraction*FreeFlowArea*AdsAxiCond*d(Tg,x)'; '-CanCSArea*CanCond*d(Tw,x)'});
model.physics('phys1').feature('gfeq1').set('da', {'(1-VoidFraction)
*FreeFlowArea*PartDensity*SorbHeatCap'; '0'; '0'; '0';
'VoidFraction*FreeFlowArea*GasDensity*AdsGasHeatCap'; '0'; '0'; '0';
'CanCSArea*CanDensity*CanHeatCap'});
model.physics('phys1').feature('init1').set('Ts', 'AdsInitTemp');
model.physics('phys1').feature('init1').set('Tg', 'AdsInitTemp');
model.physics('phys1').feature('init1').set('Tw', 'AdsInitTemp');
% *** Start Custom Statements ****************************************
% model.physics('phys1').feature('flux1').set('g', {'0'; 'initstep(t*1[1/s])
*GasDensity*InterVel*AdsGasHeatCap*(AdsInTemp-Tg)*FreeFlowArea'; '0'});
```

```
if strcmp(InTemp,'Constant') || strcmp(InTemp,'Therm_CST')
model.physics('phys1').feature('flux1').set('g', {'0'; 'initstep(t*1[1/s])
*GasDensity*InterVel*AdsGasHeatCap*(AdsInTemp-Tg)*FreeFlowArea'; '0'});
elseif strcmp(InTemp,'Timelined') || strcmp(InTemp,'Therm_TL')
model.physics('phys1').feature('flux1').set('g', {'0'; 'initstep(t*1[1/s])
*GasDensity*InterVel*AdsGasHeatCap*(TCintemp(t)-Tg)*FreeFlowArea'; '0'});
else
errordlg('Specify Inlet Temperature Control','Fatal Error');
end
% model.physics('phys1').feature('cons1').set('R', {'0'; 'Tgx'; '0'});
model.physics('phys1').feature('cons1').set('R', {'d(Ts,x)'; 'd(Tg,x)';
'd(Tw,x)'});
model.physics('phys1').feature('cons1').label('Constraint 2');
model.physics('phys1').feature('dir1').set('r', {'0'; 'intemp(t)'; '0'});
model.physics('phys1').feature('dir1').set('useDirichletCondition', {'0'; '1';
'0'});
model.physics('phys1').feature('dir1').active(false);
% *** Start Custom Statements ***************************************
if strcmp('Physics',NodeSepMax)
model.mesh('mesh1').feature('size').set('hauto', 1);
model.mesh('mesh1').run;
else
model.mesh('mesh1').feature('size').set('hmax', 'NodeSepMax');
model.mesh('mesh1').feature('size1').set('hmax', 'NodeSepInit');
model.mesh('mesh1').feature('size1').set('hminactive', false);
model.mesh('mesh1').feature('size1').set('hcurveactive', false);
model.mesh('mesh1').feature('size1').set('hnarrowactive', false);
model.mesh('mesh1').feature('size1').set('hgradactive', false);
model.mesh('mesh1').run;
end
% *** End Custom Statements ***************************************
% model.mesh('mesh1').run;
% model.mesh('mesh2').feature('size').set('custom', 'on');
% model.mesh('mesh2').feature('size').set('hmax', 'NodeSepMax');
% model.mesh('mesh2').feature('size1').set('custom', 'on');
% model.mesh('mesh2').feature('size1').set('hmaxactive', true);
% model.mesh('mesh2').feature('size1').set('hmax', 'NodeSepInit');
% model.mesh('mesh2').run;
% model.result.table('tbl1').comments('Line Integration 1 (q)');
% model.result.table('tbl2').comments('Point Evaluation 1 (c)');
model.study.create('std1');
model.study('std1').create('time', 'Transient');
model.sol.create('sol1');
model.sol('sol1').study('std1');
model.sol('sol1').attach('std1');
model.sol('sol1').create('st1', 'StudyStep');
model.sol('sol1').create('v1', 'Variables');
model.sol('sol1').create('t1', 'Time');
model.sol('sol1').feature('t1').create('fc1', 'FullyCoupled');
model.sol('sol1').feature('t1').create('d1', 'Direct');
% *** Start Custom Statements ***************************************
if strcmp(ModelSolve,'SEG')
model.sol('sol1').feature('t1').feature.create('se1', 'Segregated');
model.sol('sol1').feature('t1').feature('se1').feature.create('ss1',
'SegregatedStep');
end
% model.sol('sol1').feature('t1').create('se1', 'Segregated');
% model.sol('sol1').feature('t1').create('tpDef', 'TimeParametric');
% model.sol('sol1').feature('t1').feature('se1').create('ss1',
'SegregatedStep');
% *** End Custom Statements ***************************************
model.sol('sol1').feature('t1').feature.remove('fcDef');
model.study('std1').feature('time').set('initstudyhide', 'on');
model.study('std1').feature('time').set('initsolhide', 'on');
model.study('std1').feature('time').set('solnumhide', 'on');
model.study('std1').feature('time').set('notstudyhide', 'on');
model.study('std1').feature('time').set('notsolhide', 'on');
model.study('std1').feature('time').set('notsolnumhide', 'on');
model.study('std1').feature('time').set('rtolactive', true);
model.study('std1').feature('time').set('rtol', '0.0001');
% model.study('std1').feature('time').set('mesh', {'geom1' 'mesh1'});
model.study('std1').feature('time').set('tlist',
'range(0,TimeStep,HalfCycleLength)');
% *** Start Custom Statements ****************************************
% model.study('std1').feature('time').set('physselection', 'phys1');
% model.study('std1').feature('time').set('activate', {'g' 'on' 'phys1'
'off'});
% Disable adsorption to allow use of timelined or constant temperature input
for
% thermal characterization cases.
if strcmp(InTemp,'Constant') || strcmp(InTemp,'Timelined')
model.study('std1').feature('time').set('physselection', 'g');
elseif strcmp(InTemp,'Therm_CST') || strcmp(InTemp,'Therm_TL')
% turn off adsorption; first two lines added 08-08-2015 for 5.1 code
model.sol('sol1').feature('v1').feature('mod1_c').set('solvefor', false);
model.sol('sol1').feature('v1').feature('mod1_q').set('solvefor', false);
model.study('std1').feature('time').set('physselection', 'phys1');
model.study('std1').feature('time').set('activate', {'g' 'off' 'phys1' 'on'});
else
errordlg('Specify Inlet Temperature Control','Fatal Error');
end
model.study('std1').feature('time').set('physselection', 'phys1');
% *** End Custom Statements ****************************************
model.sol('sol1').attach('std1');
model.sol('sol1').label('Solver 1');
model.sol('sol1').feature('t1').set('atoludotactive', {'mod1_c' 'off' 'mod1_q'
'off' 'mod1_Ts' 'off' 'mod1_Tg' 'off'
'mod1_Tw' 'off'});
model.sol('sol1').feature('t1').set('fieldselection', 'mod1_c');
model.sol('sol1').feature('t1').set('atolmethod', {'mod1_c' 'global' 'mod1_q'
'global' 'mod1_Ts' 'global'
'mod1_Tg' 'global' 'mod1_Tw' 'global'});
model.sol('sol1').feature('t1').set('atol', {'mod1_c' '1e-3' 'mod1_q' '1e-3'
'mod1_Ts' '1e-3' 'mod1_Tg' '1e-3'
'mod1_Tw' '1e-3'});
model.sol('sol1').feature('t1').set('atoludot', {'mod1_c' '1e-3' 'mod1_q' '1e-
3' 'mod1_Ts' '1e-3' 'mod1_Tg' '1e-3'
'mod1_Tw' '1e-3'});
model.sol('sol1').feature('t1').set('atolglobal', '0.00010');
model.sol('sol1').feature('t1').set('bwinitstepfrac', '1.0');
model.sol('sol1').feature('t1').set('initialstepbdf', '0.0010');
model.sol('sol1').feature('t1').set('ewtrescale', false);
model.sol('sol1').feature('t1').set('tlist',
'range(0,TimeStep,HalfCycleLength)');
model.sol('sol1').feature('t1').set('rtol', '0.0001');
model.sol('sol1').feature('t1').set('stabcntrl', true);
model.sol('sol1').feature('t1').set('eventtol', '0.0001');
model.sol('sol1').feature('t1').feature('fc1').active(true);
model.sol('sol1').feature('t1').feature('fc1').set('ntermauto', 'itertol');
model.sol('sol1').feature('t1').feature('fc1').set('niter', '100');
model.sol('sol1').feature('t1').feature('fc1').set('dtech', 'hnlin');
% model.sol('sol1').feature('t1').feature('se1').set('segiter', '100');
% model.sol('sol1').feature('t1').feature('se1').set('segterm', 'itertol');
% model.sol('sol1').feature('t1').feature('se1').set('probesel', 'manual');
% model.sol('sol1').feature('t1').feature('se1').feature('ssDef').set('segvar',
{'mod1_c' 'mod1_q'});
%
model.sol('sol1').feature('t1').feature('se1').feature('ssDef').set('subdtech',
'hnlin');
% model.sol('sol1').feature('t1').feature('se1').feature('ss1').set('segvar',
{'mod1_Ts' 'mod1_Tg' 'mod1_Tw'});
% model.sol('sol1').feature('t1').feature('se1').feature('ss1').set('subdtech',
'auto');
% model.sol('sol1').feature('t1').feature('tpDef').active(false);
% *** Start Custom Statements ****************************************
if strcmp(ModelSolve,'SEG')
model.sol('sol1').feature('t1').feature('se1').set('segterm', 'itertol');
model.sol('sol1').feature('t1').feature('se1').set('segiter', '100');
model.sol('sol1').feature('t1').feature('se1').feature('ssDef').set('segvar',
{'mod1_c' 'mod1_q'});
model.sol('sol1').feature('t1').feature('se1').feature('ssDef').set('subdtech',
'hnlin');
model.sol('sol1').feature('t1').feature('se1').feature('ss1').set('segvar',
{'mod1_Ts' 'mod1_Tg' 'mod1_Tw'});
model.sol('sol1').feature('t1').feature('se1').feature('ss1').set('subdtech',
'auto');
end
% *** End Custom Statements ****************************************
model.sol('sol1').runAll;
% *** End COMSOL Statements ****************************************
% Extract COMSOL data
time = mphinterp(model,{'t'},'coord',0);
% Get specified coordinates including inlet or zero
Coords = [0 BedLength/Locations:(BedLength-BedLength/Locations)/(Locations-
1):BedLength];
i=1;
[out(i).c,out(i).ct,out(i).q,out(i).qt,out(i).Tg,out(i).Tw] = ...
mphinterp(model,{'c','ct','q','qt','Tg','Tw'},'coord',Coords);
% mphinterp(model,{'c','ct','q','qt','Tg','Tw'},'coord',...
% BedLength/Locations:(BedLength-BedLength/Locations)/(Locations-
1):BedLength]);
% ,Coords);
%out = model;
function PlotCOMSOL(GenIn,RecNum)
% Plotting Commands
% Function plots results from data stored as a Matlab workspace that was
% saved in RunSimulation.m. It is called by RunSimulation.m immediately
% following a simulation or by PlotOnly.m; in either case the RunID in the
% current record is used for the name of the .mat file.
% Initialize strings for plot footer, uitable
LongString = ' Input Values: ';
Row = 0;
% Get plotting parameters, values used for postprocessing calculations
for i = 1:size(GenIn.NumInputs,2)
for j = 1:GenIn.NumInputs(i)
% % Pick out needed values for plotting from GenIn array
% if RecNum < 0
% RecNumTemp = -RecNum;
% else
RecNumTemp = RecNum;
% end
switch char(GenIn.varname(i,j))
% case 'DataSource'
% DataSource=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'RunID'
RunID=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'Description'
Description=cell2mat(GenIn.values(i,j,RecNumTemp));
% case 'Notes'
% Notes=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'ParaIter'
ParaIterAll=ParseInput(GenIn.values(i,j,RecNumTemp));
ParaIter = ParaIterAll(1);
if ParaIter(1) == 0; ParaIter(1) = 1; end % Used to determine # of plots
case 'ParaName'
```

```
ParaName=ParseInputText(GenIn.values(i,j,RecNumTemp));
% Get header and item number for specified parameter
[HeNu,ItNu]=find(strcmp(GenIn.varname,ParaName(1)));
ParaDescr = GenIn.string{HeNu,ItNu};
ParaUnits = GenIn.unit{HeNu,ItNu};
case 'ParaMin'
ParaMin = ParseInput(GenIn.values(i,j,RecNumTemp));
case 'ParaMax'
ParaMax=ParseInput(GenIn.values(i,j,RecNumTemp));
case 'Locations'
Locations = cell2mat(GenIn.values(i,j,RecNumTemp));
case 'PlotPoints'
PlotPoints = ParseInput(GenIn.values(i,j,RecNumTemp));
case 'PlotType'
PlotType=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'PlotsPerPage'
PlotsPerPage=cell2mat(GenIn.values(i,j,RecNumTemp));
% Maintain cell structure for SimData definitions
case 'SimData'
SimData = ParseInputText(GenIn.values(i,j,RecNumTemp));
case 'ConcDataFile'
ConcDataFile=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'TempIntDataFile'
TempIntDataFile=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'TempColDataFile'
TempColDataFile=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'OffSet'
OffSet = ParseInput(GenIn.values(i,j,RecNumTemp));
% case 'SSEFileName'
% SSEFileName = ParseInputText(GenIn.values(i,j,RecNumTemp));
case 'LimParaHi'
LimParaHi = ParseInput(GenIn.values(i,j,RecNumTemp));
case 'LimParaLo'
LimParaLo = ParseInput(GenIn.values(i,j,RecNumTemp));
case 'SlopeMax'
SlopeMax = ParseInput(GenIn.values(i,j,RecNumTemp));
case 'WriteSim'
WriteSim=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'AdsConc'
AdsConc=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'AdsAxialDisp'
AdsAxialDisp=cell2mat(GenIn.values(i,j,RecNumTemp));
AxialDispStr=num2str(cell2mat(GenIn.values(i,j,RecNumTemp))); %#ok<NASGU>
case 'LDF'
LDF=cell2mat(GenIn.values(i,j,RecNumTemp));
LDFStr=num2str(cell2mat(GenIn.values(i,j,RecNumTemp))); %#ok<NASGU>
case 'AdsSupVel'
AdsSupVel=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'FreeFlowArea'
FreeFlowArea=cell2mat(GenIn.values(i,j,RecNumTemp));
% case 'Sorb'
% Sorb=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'AdsInTemp'
AdsInTemp=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'PartDensity'
PartDensity=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'SorbentMass'
SorbentMass=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'SorbentMassChk'
SorbentMassChk=cell2mat(GenIn.values(i,j,RecNumTemp));
case 'LegendLoc'
LegendLoc = ParseInputText(GenIn.values(i,j,RecNumTemp));
case 'ParaLoc'
ParaLoc = ParseInputText(GenIn.values(i,j,RecNumTemp));
case 'ConcLegends'
```

```
ConcLegends = ParseInputText(GenIn.values(i,j,RecNumTemp));
case 'GasTempLeg'
GasTempLeg = ParseInputText(GenIn.values(i,j,RecNumTemp));
case 'ColTempLeg'
ColTempLeg = ParseInputText(GenIn.values(i,j,RecNumTemp));
case 'ModelName'
ModelNSolverName = ParseInputText(GenIn.values(i,j,RecNum));
ModelName = cell2mat(ModelNSolverName(1));
ModelSolver = cell2mat(ModelNSolverName(2));
end
if strcmp('yes',GenIn.export(i,j)) || strcmp('sum',GenIn.export(i,j))
% Reordering steps and go with staight printout with summary
% test in first box.
% % Build Arrays for Summary printout. Description is in title,
% % so is skipped. Next 4 are single column text. Remaining
% % fields go into the two column section. These are later reordered by
% % topic with spaces between topics.
Row = Row + 1;
% if Row >1 && Row <= 4
if Row <= 7
% Data1(Row-1,1) = strcat(GenIn.string(i,j),':');
% Data1(Row-1,2) = GenIn.values(i,j,RecNum);
Data1(Row,1) = strcat(GenIn.string(i,j),':');
Data1(Row,2) = GenIn.values(i,j,RecNum);
% elseif Row > 4
elseif Row > 7
% Data2(Row-4,1) = strcat(GenIn.string(i,j),GenIn.unit(i,j),':');
Data2(Row-7,1) = strcat(GenIn.string(i,j),GenIn.unit(i,j),':');
if ~iscellstr(GenIn.values(i,j,RecNum))
% Data2(Row-4,2) = {sprintf('%0.3g',(cell2mat(GenIn.values(i,j,RecNum))))};
Data2(Row-7,2) = {sprintf('%0.3g',(cell2mat(GenIn.values(i,j,RecNum))))};
else
% Data2(Row-4,2) = GenIn.values(i,j,RecNum);
Data2(Row-7,2) = GenIn.values(i,j,RecNum);
end
end
% Straight output of reported values; skip reordering steps
% below.
% Add topic headers to Data2 array
% Data2(60,1) = {'__________Key Inputs__________'};
% Data2(61,1) = {'___________Sorbent___________'};
% Data2(62,1) = {'____________Canister____________'};
% Data2(63,1) = {'_____Inlet and Initial Conditions______'};
% Data2(64,1) = {'________Model Parameters_________'};
% Data2(65,1) = {'__________Calculations___________'};
% Data2(66,1) = {'Sim Mass In - Out[mole]:'};
% Data2(67,1) = {'Sim Mass Adsorbed[mole]:'};
% Data2(68,1) = {'Sim Stoich BT[min]:'};
% Data2(69,1) = {'Database Record'};
% Data2(69,2) = {RecNumTemp};
% Data2(70,1) = {''};
% Data2(71,1) = {'Notes and Observations:'};
% Data2(72,1) = {'_________Error Checking__________'};
% Assembly of input values for plot footer
if ~iscellstr(GenIn.values(i,j)) && strcmp('yes',GenIn.export(i,j))
num = cell2mat(GenIn.values(i,j,RecNum));
values = strcat(num2str(num),GenIn.unit(i,j));
% Assembly of input values for plot footer
LongString = [LongString,cell2mat(GenIn.string(i,j)),...
' = ',cell2mat(values),'; '];
end
end
end
end
Row=Row-7;
```

```
Data2(Row+1,1) = {'Sim Moles In - Out[mole]:'};
Data2(Row+2,1) = {'Sim Moles Adsorbed[mole]:'};
Data2(Row+3,1) = {'Sim Stoich BT[min]:'};
Data2(Row+4,1) = {'Database Record'};
Data2(Row+4,2) = {RecNumTemp};
% Load .mat file. Renaming the RunID variable to RunIDPlot allows
% specification of a .mat file that has a different filename than what is
% recorded as RunID inside the .mat file
RunIDPlot = RunID;
load([RunID,'.mat']);
if NumHC > 1
NumCycle = NumHC/2;
ParaIter = 1;
else
NumCycle = 1;
if ParaIter == 0; ParaIter = 1; end % Used to determine # of plots
end
% Set Y-scale limits; LimPara sets tightness
% Initialize axis limit variables
ylimHiAct_c = 0;
ylimHiAct_ct = 0;
ylimHiAct_q = 0;
ylimHiAct_qt = 0;
ylimHiAct_Tg = 0;
ylimHiAct_Tw = 0;
ylimLoAct_c = 1e10;
ylimLoAct_ct = 1e10;
ylimLoAct_q = 1e10;
ylimLoAct_qt = 1e10;
ylimLoAct_Tg = 1e10;
ylimLoAct_Tw = 1e10;
% Get y axis limits. Ignore 1st 2 for slope plot max only.
% Number of repeated plots are based on number of cycles for cyclic run,
% then on number of parameters if parametric run
if NumCycle > 1;
NumRep = NumCycle;
elseif ParaIter > 1
NumRep = ParaIter;
else
NumRep = 1;
end
for i=1:NumRep
%Max
if max(max(out(i).c)) > ylimHiAct_c; %#ok<NODEF>
ylimHiAct_c = max(max(out(i).c));
end
if max(max(out(i).ct(3:end,:))) > ylimHiAct_ct;
ylimHiAct_ct = max(max(out(i).ct(3:end,:)));
end
if max(max(out(i).q)) > ylimHiAct_q;
ylimHiAct_q = max(max(out(i).q));
end
if max(max(out(i).qt)) > ylimHiAct_qt;
ylimHiAct_qt = max(max(out(i).qt));
end
if max(max(out(i).Tg)) > ylimHiAct_Tg;
ylimHiAct_Tg = max(max(out(i).Tg));
end
if max(max(out(i).Tw)) > ylimHiAct_Tw;
ylimHiAct_Tw = max(max(out(i).Tw));
end
%Min
if min(min(out(i).c)) < ylimLoAct_c;
ylimLoAct_c = min(min(out(i).c));
end
```

```
if min(min(out(i).ct)) < ylimLoAct_ct;
ylimLoAct_ct = min(min(out(i).ct));
end
if min(min(out(i).q)) < ylimLoAct_q;
ylimLoAct_q = min(min(out(i).q));
end
if min(min(out(i).qt)) < ylimLoAct_qt;
ylimLoAct_qt = min(min(out(i).qt));
end
if min(min(out(i).Tg)) < ylimLoAct_Tg;
ylimLoAct_Tg = min(min(out(i).Tg));
end
if min(min(out(i).Tw)) < ylimLoAct_Tw;
ylimLoAct_Tw = min(min(out(i).Tw));
end
end
% Setup plot page based on user inputs. DataBox is in first column and
% contains key breakthrough data. ParaBox is in specified column(s) and
% contains values for varied parameter. Summary is a fixed format with 7
% plots and a table with COMSOL input data (instead of in paragraph form at
% the bottom of the page).
% Figure position with aspect ratio of letter size page in landscape
% orientation
FigFromLeft = 0; FigFromBot = 0; FigWidth = 1100; FigHeight = 850;
% Constant figure element sizes (these are in fractional page size)
Margin = 0.02; HeightHead = 0.06; HeightTitle = 0.03; HeightFoot = 0.12;
PlotMarg = 0.05; WidthDB = 0.13; WidthPara = 0.175; HeightPara = 0.03;
% Calculated figure element sizes. For Summary, remove HeightFoot
if strcmp(PlotType,'Summary'); HeightFoot = 0.0; end;
WidthHeadFoot = 1 - 2*Margin;
HeightPlots = 1 - HeightHead - HeightTitle - HeightFoot - 2*Margin;
WidthPlots = 1 - 2*Margin;
% Verticals for plots, DataBox, Title, and ParaText. For Summary,
% PlotsPerPage is always = 3
if strcmp(PlotType,'Summary'); PlotsPerPage = 3; end;
HeightPlot = HeightPlots/PlotsPerPage - PlotMarg;
FromBotTitle = 1 - HeightHead - HeightTitle;
for i = 1:PlotsPerPage
FromBot(i) = 1 - HeightHead - HeightTitle...
- i*(HeightPlots/PlotsPerPage)+Margin;
end
% Horizontals for plots and DataBox, Define Plot Data. For Summary, plots
% are defined later; to force calculation of spacing for 3 plots irregardless
% of record input, SimData is redefined.
if strcmp(PlotType,'Summary')
SimData(1) = {'cWT'}; SimData(2) = {'TgWT'}; SimData(3) = {'TgWT'};
end
% DataBox on left side
if strcmp(SimData(1),'DataBox')
FromLeftP(1) = WidthDB + Margin + PlotMarg;
if strcmp(SimData(3),'none') %Left Side DataBox and one plot
PlotColumns = 1;
WidthPlot = WidthPlots - WidthDB - PlotMarg;
PlotDef(1) = SimData(2);
else %Left Side DataBox and two plots
PlotColumns = 2;
WidthPlot = (WidthPlots - WidthDB)/2 - PlotMarg;
FromLeftP(2) = FromLeftP(1) + WidthPlot + PlotMarg;
PlotDef(1) = SimData(2);
PlotDef(2) = SimData(3);
% FromLeftPara2 = FromLeftP(2) + WidthPlot/4 - WidthPara/2;
end
else
% No DataBox
FromLeftP(1) = Margin + PlotMarg;
```

```
if strcmp(SimData(2),'none') % One plot
PlotColumns = 1;
WidthPlot = WidthPlots - PlotMarg;
PlotDef(1) = SimData(1);
elseif strcmp(SimData(3),'none') % Two plots
PlotColumns = 2;
WidthPlot = WidthPlots/2 - PlotMarg;
FromLeftP(2) = FromLeftP(1) + WidthPlot + PlotMarg;
PlotDef(1) = SimData(1);
PlotDef(2) = SimData(2);
% FromLeftPara3 = FromLeftP(2) + WidthPlot/4 - WidthPara/2;
else
PlotColumns = 3; % Three plots
WidthPlot = WidthPlots/3 - PlotMarg;
FromLeftP(2) = FromLeftP(1) + WidthPlot + PlotMarg;
FromLeftP(3) = FromLeftP(2) + WidthPlot + PlotMarg;
PlotDef(1) = SimData(1);
PlotDef(2) = SimData(2);
PlotDef(3) = SimData(3);
% FromLeftPara3 = FromLeftP(3) + WidthPlot/4 - WidthPara/2;
end
end
% ParaBox locations
for i = 1:PlotsPerPage
for j = 1:PlotColumns
switch cell2mat(ParaLoc(j))
case 'NorthWest'
FromLeftPara(j) = FromLeftP(j) + 0.008;
FromBotPara(i) = FromBot(i) + HeightPlot - HeightPara - 0.005;
case 'NorthEast'
FromLeftPara(j) = FromLeftP(j) + WidthPlot - WidthPara - 0.008;
FromBotPara(i) = FromBot(i) + HeightPlot - HeightPara - 0.005;
case 'SouthWest'
FromLeftPara(j) = FromLeftP(j) + 0.008;
FromBotPara(i) = FromBot(i) + 0.008;
case 'SouthEast'
FromLeftPara(j) = FromLeftP(j) + WidthPlot - WidthPara - 0.008;
FromBotPara(i) = FromBot(i) + 0.008;
case 'ctSpecial'
HeightPara = 0.1; WidthPara = 0.15;
FromLeftPara(j) = FromLeftP(j) + WidthPlot/2.5 - WidthPara/2 - 0.008;
FromBotPara(i) = FromBot(i) + HeightPlot - HeightPara - 0.008;
end
end
end
%Create faint outline, also forces fullsize page after eps conversion
%(using loose command in eps write instead)
% annotation(figure1,'textbox',...
% [0 0 1 1],'FitBoxToText','off','LineWidth',0.01,...
% 'Color',[.01 .01 .01],'HorizontalAlignment','center','Interpreter','none');
%% Specialized Summary Plot code - eventually to become a general use function
if strcmp(PlotType,'Summary')
% Convert bed loading and slope of bed loading to mol/kg
out(1).q(:,:) = out(1).q(:,:)/PartDensity;
out(1).qt(:,:) = out(1).qt(:,:)/PartDensity;
% Create figure with aspect ratio of letter size page in landscape
% orientation
figure1=figure('Name','figure1','Position',...
[FigFromLeft FigFromBot FigWidth FigHeight],'Color',[1 1 1]);
% Plot Title
% str2(1) = {[Description,'. Data Source = ',DataSource,', Model Name =
',ModelName]};
annotation(figure1,'textbox',...
[Margin FromBotTitle WidthHeadFoot
HeightTitle],'String',Description,'FitBoxToText',...
```

```
'off','HorizontalAlignment','center','Interpreter','none');
% Specify number of simulation points to plot as follows in GUI, or
% uncomment the following line:
PlotPoints = [3,3,3,50,50,50,50];
% Collection of test data and determination of limits for summary plot;
% Sim data is in structure array out(i).VAR, with VAR =
% 'c','ct','q','qt','Tg','Tw'; i is the parametric run number. For first cut,
ignore parametric
% runs (plot only out(1).VAR(i,j)
% plot(1) - concentration compared with test data
% plot(4) - concentration profile
PlotYLabel(1)={'Gas Concentration (mol/m^3)'};
PlotYLabel(4)={'Gas Concentration (mol/m^3)'};
PlotData(:,:,1) = out(1).c;
PlotData(:,:,4) = out(1).c;
TestFile(1) = {ConcDataFile};
% plot(2) - gas temperature compared with test data
PlotYLabel(2)={'Gas Temperature (K)'};
PlotData(:,:,2) = out(1).Tg;
TestFile(2) = {TempIntDataFile};
% plot(3) - column temperature compared with test data
PlotYLabel(3)={'Column Temperature (K)'};
PlotData(:,:,3) = out(1).Tw;
TestFile(3) = {TempColDataFile};
% plot(5) - slope of concentration profile
PlotYLabel(5)={'Slope of Concentration (mol/m^3/s)'};
PlotData(:,:,5) = out(1).ct;
% plot(6) - bed loading
PlotYLabel(6)={'Solid Concentration (mol/kg)'};
PlotData(:,:,6) = out(1).q;
% plot(7) - slope of concentration profile
PlotYLabel(7)={'Slope of Solid Conc (mol/kg/s)'};
PlotData(:,:,7) = out(1).qt;
% Obtain limits based on sim data and on test data for plots 1-3.
% Perform file operations for plots 1-3.
for i = 1:7
% Obtain axis limits based on sim data
[yLimLo(i), yLimHi(i)] =
HiLoSimData(PlotData(:,:,i),LimParaLo(i),LimParaHi(i));
% Special case for ct, where first locations are much higher
[yLimLo(5), yLimHi(5)] =
HiLoSimData(PlotData(3:end,:,5),LimParaLo(5),LimParaHi(5));
% Check for test data, compare/update current axis limits for first
% three plots
if i < 4 && ~strcmp(char(TestFile(i)),'none')
TestData(i).a(:,:) = importdata(char(TestFile(i)));
[yLimLoTest, yLimHiTest] =
HiLoTestData(TestData(i).a(:,:),LimParaLo(i),LimParaHi(i));
if yLimLoTest < yLimLo(i); yLimLo(i) = yLimLoTest; end
if yLimHiTest > yLimHi(i); yLimHi(i) = yLimHiTest; end
end
end
% Setup plot styles for test data
Marker(1) = '.'; Marker(2) = 's'; Marker(3) = 'd';Marker(4) = '^';
MarkerSizes(1) = 10; MarkerSizes(2) = 5; MarkerSizes(3) = 6; MarkerSizes(4) =
6;
Color(1) = 'b'; Color(2) = 'g'; Color(3) = 'r'; Color(4) = 'k';
Styles(1) = {':'}; Styles(2) = {'--'}; Styles(3) = {'-'}; Styles(4) = {'-.'};
% Loop for summary plot. Use row,column for loop to use existing
% plot layout calculations; calculate ps for 1-7 plot sequence.
for i=1:3
for j = 1:3
ps = j + 3*(i-1);
ii = i; jj = j;
if ps < 8
if ps == 6; ii=3; jj=1; end;
if ps == 7; ii=3; jj=2; end;
Axes(ps) = axes('Parent',figure1,...
'Position',[FromLeftP(jj) FromBot(ii) WidthPlot HeightPlot]);
box(Axes(ps),'on');
hold(Axes(ps),'all');
ylim(Axes(ps),[min(yLimLo(ps)) max(yLimHi(ps))]);
% Test Data plots - first three only.
LabelText={};
if ps < 4 && ~strcmp(char(TestFile(ps)),'none')
for k = 1:length(TestData(ps).a(1,:))/2
plot((TestData(ps).a(:,k*2-1)+OffSet(ps))/3600,...
TestData(ps).a(:,k*2),Marker(k),'Parent',Axes(ps),...
'MarkerSize',MarkerSizes(k),'MarkerFaceColor',Color(k),...
'MarkerEdgeColor',Color(k));
end
% Setup Legend Text for Test Data
if ~strcmp(LegendLoc(ps),'none')
switch ps
case 1
LabelText = ConcLegends;
case 2
LabelText = GasTempLeg;
case 3
LabelText = ColTempLeg;
end
end
end
% Create Plot data. Plot last location if PlotPoints(j) = 1, or sampled
% locations otherwise
if PlotPoints(ps) == 1
LocIndex = 1;
PercentBed = 100;
PlotDat = PlotData(:,end,ps);
else
% Setup Indices and Legend Text for Sim Data. The array
% indices start from 1 so 1 must be subtracted to get
% actual bed location.
DeltaLoc = round((Locations)/(PlotPoints(ps)-1));
LocIndex = [0 DeltaLoc:DeltaLoc:Locations]+1;
% Special case for 1/4 inch into a 10 inch bed at each end with 200
% locations to get the right location (MSMBT)
if Locations == 200 && PlotPoints(ps) == 3; LocIndex = [5 100 195]+1; end
% Special case for 25, 50, 75% into a 2 inch bed at
% each end with 100 locations to get the right location
% (KSCARL)
if Locations == 100 && PlotPoints(ps) == 3; LocIndex = [25 50 75]+1; end
PercentBed = 100*(LocIndex-1)/Locations;
PlotDat = PlotData(:,LocIndex,ps);
end
if length(LocIndex) < 5
% Plot data with custom lines and colors to match
% test data. Compile array to write to file for
% comparisons using Series or PlotOver plots
outdata = [];
for k = 1:length(LocIndex)
plot(time/3600,PlotDat(:,k),'Parent',Axes(ps),'Color',Color(k),...
'LineStyle',char(Styles(k)),'LineWidth',1.25);
outdata = cat(2,outdata,time,PlotDat(:,k));
end
% Write sim data to file if so directed by GUI
if strcmp(WriteSim,'yes')
SimVar = fieldnames(out);
outname = ['Sim_' char(SimVar(ps)) '_R' num2str(RecNum) '_Data' '.txt'];
dlmwrite(outname,outdata,'delimiter','\t');
```

```
end
LabelNum = length(LabelText);
for m = 1:PlotPoints(ps)
LabelNum = LabelNum+1;
LabelText(LabelNum) = {[num2str(PercentBed(m),...
'%3.0f'),'%']};
end
else
% Plot data with defaults (lines, colors cycle)
plot(time/3600,PlotDat,'Parent',Axes(ps));
end
% Plot x label
xlabel(Axes(ps),'Time (hours)');
% Plot y label
ylabel(Axes(ps),PlotYLabel(ps));
% For first three plots, plot legend if specified
if ps < 4 && ~strcmp(LegendLoc(ps),'none')
if PlotPoints(ps) < 5
legend(Axes(ps),LabelText,'Location',cell2mat(LegendLoc(ps)));
end
end
if ps == 5 && strcmp(ParaLoc(2),'ctSpecial')
% For ct plot, show metrics for breakthrough curve
% sharpening
IndStart = int16(SlopeMax(1)/100*Locations);
IndEnd = int16(SlopeMax(2)/100*Locations);
ctMaxInt = min(max(out(1).ct(3:end,IndStart:IndEnd)));
ctMaxBT = max(out(1).ct(3:end,end));
str(1) = {['Disp/LDF =',num2str(AdsAxialDisp/LDF,2)]};
str(2) = {['Max Int Slope = ',num2str(ctMaxInt,2)]};
str(3) = {['Max BT Slope = ',num2str(ctMaxBT,2)]};
str(4) = {['Slope Ratio = ',num2str(ctMaxBT/ctMaxInt,3)]};
annotation(figure1,'textbox',...
[FromLeftPara(jj) FromBotPara(ii) WidthPara HeightPara],...
'String',str,'FitBoxToText','on','FontSize',9,...
'HorizontalAlignment','center','VerticalAlignment',...
'middle','Margin',0.5);
end
if ps == 7 && strcmp(ParaLoc(2),'ctSpecial')
% For qt plot, show metrics for breakthrough curve
% sharpening
qtMaxInt = min(max(out(1).qt(3:end,IndStart:IndEnd)));
qtMaxBT = max(out(1).qt(3:end,end));
qtstr(1) = {['Max Int Slope = ',num2str(qtMaxInt,2)]};
qtstr(2) = {['Max BT Slope = ',num2str(qtMaxBT,2)]};
qtstr(3) = {['Slope Ratio = ',num2str(qtMaxBT/qtMaxInt,3)]};
annotation(figure1,'textbox',...
[FromLeftPara(jj) FromBotPara(ii) WidthPara HeightPara],...
'String',qtstr,'FitBoxToText','on','FontSize',9,...
'HorizontalAlignment','center','VerticalAlignment',...
'middle','Margin',0.5);
end
end
end
end
% Post processing calculations (mass balance and stoich. BT) for
% summary textbox
% Integration and difference of Inlet and Outlet moles
% MolIn =
out(1).c(:,1).*out(1).Tg(:,1)/(AdsInTemp+273.15)*AdsSupVel*FreeFlowArea/10000;
% Use below for no adjustment of velocity, density with temperature
MolIn = out(1).c(:,1)*AdsSupVel*FreeFlowArea/10000;
TotMolIn = trapz(time,MolIn);
% MolOut =
out(1).c(:,end).*out(1).Tg(:,end)/(AdsInTemp+273.15)*AdsSupVel*FreeFlowArea/100
```

```
00;
% Use below of for adjustment of velocity, density with temperature
MolOut = out(1).c(:,end)*AdsSupVel*FreeFlowArea/10000;
TotMolOut = trapz(time,MolOut);
MolFlowTot = TotMolIn-TotMolOut %#ok<NOPRT>
Data2(Row+1,2) = {sprintf('%0.3g',MolFlowTot)};
% Integration of bed loading. Note q here has been converted to
% mol sorbate / kg sorbent
if SorbentMass == 0; SorbentMass = SorbentMassChk; end
TotMolSorbInit = trapz(out(1).q(1,:))/Locations*SorbentMass/1000;
TotMolSorbEnd = trapz(out(1).q(end,:))/Locations*SorbentMass/1000;
MolSorbedTot = TotMolSorbEnd-TotMolSorbInit %#ok<NOPRT>
Data2(Row+2,2) = {sprintf('%0.3g',MolSorbedTot)};
% Find time where BT curve crosses midheight
NearZero = 10000;
for i = 1:length(out(1).c(:,end))
if abs(1-(AdsConc/2)/out(1).c(i,end)) < NearZero
NearZero = abs(1-(AdsConc/2)/out(1).c(i,end));
StoichTime = time(i)/60;
end
end
Data2(Row+3,2) = {sprintf('%0.3g',StoichTime)};
%? Skip following table reordering code
%? % Reorder Summary data table in logical grouping
%? SumInd1 = [69 60 22 45 13 14 39 48 26 15:17 61 20 21 23:25 40 49:50 53:57 58
72 28:32 71];
%? SumInd2 = [1 62 8:12 63 4:7 64 33:37 65 41:44 66:68 51:52];
%? Data2(1:length(SumInd1),3) = Data2(SumInd1,1);
%? Data2(1:length(SumInd1),4) = Data2(SumInd1,2);
%? Data2(1:length(SumInd2),5) = Data2(SumInd2,1);
%? Data2(1:length(SumInd2),6) = Data2(SumInd2,2);
%? Data3(1,1) = Data2(2,2);
%? Data4(1,1) = Data2(3,2);
%? Data2 = Data2(1:max(length(SumInd1),length(SumInd2)),3:6);
% Swap RunID with label and move Notes header to center
%? LastRow2 = 34;
% Split data into two columns for table
if int8(length(Data2(:,1))/2)*2 == length(Data2(:,1)); % even number
LastRow2 = int8(length(Data2(:,1)))/2;
else
LastRow2 = int8((length(Data2(:,1)) + 1)/2);
Data2(LastRow2*2,1) = {''};
Data2(LastRow2*2,2) = {''};
end
for Row = 1:LastRow2
Data2(Row,3) = Data2(Row+LastRow2,1);
Data2(Row,4) = Data2(Row+LastRow2,2);
end
%? Skip following table reordering code
%? Data2(1,3) = Data2(1,4);
%? Data2(1,4) = {'(Run ID)'};
%? Data2(LastRow2,2) = Data2(LastRow2,1);
%? Data2(LastRow2,1) = {''};
% Create textboxes for COMSOL input data and calculations.
% Locations, height, and width of upper section (two columns), middle
% section (four columns), and bottom section (one column)
Tweak = 0.1; % fix for non-sorted text
FromLeftT(1) = 0.67;
FromLeftT(2) = 0.7375;
FromLeftT(3) = 0.75;
FromLeftT(4) = 0.826;
FromLeftT(5) = 0.909;
FromBotT(1) = 0.532 - Tweak;
WidthT(1) = 0.325;
WidthT(2) = 0.2;
WidthT(2) = 0.32;
WidthT(3) = 0.16;
HeightT(1) = 0.057 + Tweak;
% HeightT(3) = 0.05;
% FromBotT(2) = FromBot(3)+2*HeightT(3)+0.002;
FromBotT(2) = FromBot(3)+0.002;
% FromBotT(2) = Margin+HeightT(3);
% HeightT(2) = 0.468 - 2*HeightT(3);
HeightT(2) = 0.468 - Tweak;
LitFont = 5;
BigFont = 6;
% MyFont = 'Arial Narrow';
MyFont = 'Times New Roman';
% Labels for upper section, column 1
annotation(figure1,'textbox',...
[FromLeftT(1) FromBotT(1) WidthT(1) HeightT(1)],...
'String',Data1(:,1),'FontSize',BigFont,'FitBoxToText','off','FontName',MyFont);
% Data for upper section, column 2
annotation(figure1,'textbox',...
[FromLeftT(2) FromBotT(1) WidthT(2) HeightT(1)],...
'String',Data1(:,2),'FontSize',BigFont,'EdgeColor','none','FitBoxToText','off',
'FontName',MyFont);
% Labels for lower section, column 1
% [FromLeftT(1) FromBotT(2) WidthT(1) HeightT(2)],... %note box size temp. fix
below;
annotation(figure1,'textbox',...
[FromLeftT(1) 0.001 WidthT(1) HeightT(2)+FromBotT(2)],...
'String',Data2(1:LastRow2,1),'FontSize',LitFont,'FitBoxToText','off','Interpret
er','none','FontName',
MyFont);
% Data for lower section, column 2
annotation(figure1,'textbox',...
[FromLeftT(3) FromBotT(2) WidthT(3) HeightT(2)],...
'String',Data2(1:LastRow2,2),'FontSize',LitFont,'FitBoxToText','off','EdgeColor
','none','FontName',
MyFont);
% Labels for lower section, column 3
annotation(figure1,'textbox',...
[FromLeftT(4) FromBotT(2) WidthT(1) HeightT(2)],...
'String',Data2(:,3),'FontSize',
LitFont,'FitBoxToText','off','Interpreter','none','EdgeColor','none','FontName'
,MyFont);
% Data for lower section, column 4
annotation(figure1,'textbox',...
[FromLeftT(5) FromBotT(2) WidthT(1) HeightT(2)],...
'String',Data2(:,4),'FontSize',LitFont,'FitBoxToText','off','EdgeColor','none',
'FontName',MyFont);
% % Text for Notes section
% annotation(figure1,'textbox',...
% [FromLeftT(1) FromBot(3)+HeightT(3) WidthT(1) HeightT(3)],...
% 'String',Data3,'FontSize',LitFont,'FitBoxToText','off','FontName',MyFont)
%
% % Text for Observations section
% annotation(figure1,'textbox',...
% [FromLeftT(1) FromBot(3) WidthT(1) HeightT(3)],...
% 'String',Data4,'FontSize',LitFont,'FitBoxToText','off','FontName',MyFont);
% % ,'EdgeColor','none');
% % Create uitables for COMSOL input data
% !!! produces bitmap even in EPS !!!!
% t1 = uitable('Parent',figure1,'Units','normalized');
% set(t1,'Position',[FromLeftP(3)-0.8*PlotMarg FromBot(2)+HeightPlot/2
WidthPlot+PlotMarg
HeightPlot/2]);
% set(t1,'RowName',[],'Data',Data1,'ColumnName',[],'ColumnWidth',{80
268},'FontSize',9);
```

```
% set(t1,'ColumnFormat',{'char' 'char'});
% set(t1,'FontName','Helvetica')
%
% t2 = uitable('Parent',figure1,'Units','normalized');
% set(t2,'Position',[FromLeftP(3)-0.8*PlotMarg FromBot(3) WidthPlot+PlotMarg
1.78*HeightPlot]);
% set(t2,'RowName',[],'Data',Data2,'ColumnName',[],'ColumnWidth',{110 60 117
61},'FontSize',7);
% set(t2,'ColumnFormat',{'char' 'char' 'char' 'char'});
% set(t2,'FontName','Helvetica')
end
%% Flexible Plotting Code
if ~strcmp(PlotType,'Summary')
% Create figure with aspect ratio of letter size page in landscape
% orientation
figure1=figure('Name','figure1','Position',...
[FigFromLeft FigFromBot FigWidth FigHeight],'Color',[1 1 1]);
for i = 1:NumRep
for j = 1:PlotColumns % Plot Columns
switch char(PlotDef(j)) % Plot Selection
case 'cWT'
PlotData(:,:,i,j) = out(i).c; %#ok<*AGROW>
PlotYLabel(j)={'Gas Concentration (mol/m^3)'};
TestData(j).a = importdata(ConcDataFile);
if max(max(TestData(j).a(:,2:2:end))) > ylimHiAct_c;
ylimHiAct_c = max(max(TestData(j).a(:,2:2:end)));
end
ylimHi(i,j) = (ceil((ylimHiAct_c*10^(-floor(log10(ylimHiAct_c))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_c));
if min(min(TestData(j).a(:,2:2:end))) < ylimLoAct_c;
ylimLoAct_c = min(min(TestData(j).a(:,2:2:end)));
end
if ylimLoAct_c <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_c*10^(-floor(log10(ylimLoAct_c))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_c));
end
case 'cWoT'
PlotData(:,:,i,j) = out(i).c; %#ok<*AGROW>
PlotYLabel(j)={'Gas Concentration (mol/m^3)'};
ylimHi(i,j) = (ceil((ylimHiAct_c*10^(-floor(log10(ylimHiAct_c))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_c));
if ylimLoAct_c <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_c*10^(-floor(log10(ylimLoAct_c))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_c));
end
case 'ct'
PlotData(:,:,i,j) = out(i).ct; %#ok<*AGROW>
PlotYLabel(j)={'Slope of Gas Concentration (mol/m^3/s)'};
ylimHi(i,j) = (ceil((ylimHiAct_ct*10^(-floor(log10(ylimHiAct_ct))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_ct));
if ylimLoAct_ct <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_ct*10^(-floor(log10(ylimLoAct_ct))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_ct));
end
case 'q'
PlotData(:,:,i,j) = out(i).q; %#ok<*AGROW>
PlotYLabel(j)={'Solid Concentration (mol/m^3)'};
ylimHi(i,j) = (ceil((ylimHiAct_q*10^(-floor(log10(ylimHiAct_q))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_q));
if ylimLoAct_q <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_q*10^(-floor(log10(ylimLoAct_q))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_q));
end
case 'qt'
PlotData(:,:,i,j) = out(i).qt; %#ok<*AGROW>
```

```
PlotYLabel(j)={'Slope of Solid Concentration (mol/m^3/s)'};
ylimHi(i,j) = (ceil((ylimHiAct_qt*10^(-floor(log10(ylimHiAct_qt))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_qt));
if ylimLoAct_qt <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_qt*10^(-floor(log10(ylimLoAct_qt))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_qt));
end
case 'TgWT'
PlotData(:,:,i,j) = out(i).Tg; %#ok<*AGROW>
PlotYLabel(j)={'Gas Temperature (K)'};
% Set up lower and upper bounds on y axis
TestData(j).a = importdata(TempIntDataFile);
if max(max(TestData(j).a(:,2:2:end))) > ylimHiAct_Tg
ylimHiAct_Tg = max(max(TestData(j).a(:,2:2:end)));
end
ylimHi(i,j) = (ceil((ylimHiAct_Tg*10^(-floor(log10(ylimHiAct_Tg))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_Tg));
if min(min(TestData(j).a(:,2:2:end))) < ylimLoAct_Tg;
ylimLoAct_Tg = min(min(TestData(j).a(:,2:2:end)));
end
if ylimLoAct_Tg <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_Tg*10^(-floor(log10(ylimLoAct_Tg))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_Tg));
end
case 'TgWoT'
PlotData(:,:,i,j) = out(i).Tg; %#ok<*AGROW>
PlotYLabel(j)={'Gas Temperature °K'};
ylimHi(i,j) = (ceil((ylimHiAct_Tg*10^(-floor(log10(ylimHiAct_Tg))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_Tg));
if ylimLoAct_Tg <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_Tg*10^(-floor(log10(ylimLoAct_Tg))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_Tg));
end
case 'TwWT'
PlotData(:,:,i,j) = out(i).Tw; %#ok<*AGROW>
PlotYLabel(j)={'Column Temperature °K'};
% Set up lower and upper bounds on y axis
TestData(j).a = importdata(TempColDataFile);
if max(max(TestData(j).a(:,2:2:end))) > ylimHiAct_Tw;
ylimHiAct_Tw = max(max(TestData(j).a(:,2:2:end)));
end
ylimHi(i,j) = (ceil((ylimHiAct_Tw*10^(-floor(log10(ylimHiAct_Tw))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_Tw));
if min(min(TestData(j).a(:,2:2:end))) < ylimLoAct_Tw;
ylimLoAct_Tw = min(min(TestData(j).a(:,2:2:end)));
end
if ylimLoAct_Tw <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_Tw*10^(-floor(log10(ylimLoAct_Tw))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_Tw));
end
case 'TwWoT'
PlotData(:,:,i,j) = out(i).Tw; %#ok<*AGROW>
PlotYLabel(j)={'Column Temperature °K'};
ylimHi(i,j) = (ceil((ylimHiAct_Tw*10^(-floor(log10(ylimHiAct_Tw))))...
*10^LimParaHi(j))*10^-LimParaHi(j))*10^floor(log10(ylimHiAct_Tw));
if ylimLoAct_Tw <= 0 ; ylimLo(i,j) = 0 ; else
ylimLo(i,j) = (floor((ylimLoAct_Tw*10^(-floor(log10(ylimLoAct_Tw))))...
*10^LimParaLo(j))*10^-LimParaLo(j))*10^floor(log10(ylimLoAct_Tw));
end
end
end
end
if ParaIter > 1
% Determine range of values, parametric variable name, and units
if length(ParaMin) == 1
```

```
ParaValues=ParaMin:(ParaMax-ParaMin)/(ParaIter-1):ParaMax;
else
% Manual Assignments
ParaValues=ParaMin;
end
% Create Main Title
str2(1) = {['Parametric Study on ',ParaDescr,' for ',...
Description,'. Run ID = ',RunIDPlot,', Model Name = ',ModelName,...
', Solver = ',ModelSolver]};
else
str2(1) = {[Description,'. Run ID = ',RunIDPlot,', Model Name = ',ModelName,...
', Solver = ',ModelSolver]};
end
annotation(figure1,'textbox',...
[Margin FromBotTitle WidthHeadFoot
HeightTitle],'String',str2,'FitBoxToText',...
'off','HorizontalAlignment','center','Interpreter','none');
%Create faint outline, also forces fullsize page after eps conversion
annotation(figure1,'textbox',...
[0 0 1 1],'FitBoxToText','off','LineWidth',0.01,...
'Color',[.01 .01 .01],'HorizontalAlignment','center','Interpreter','none');
%Create Bottom Label
% LongString is compiled at start of function
str3 = {LongString};
annotation(figure1,'textbox',...
[Margin Margin WidthHeadFoot HeightFoot],...
'String',str3,'FitBoxToText','off','HorizontalAlignment','left',...
'FontSize',8,'Interpreter','None','LineStyle','none','VerticalAlignment','middl
e');
% Determine number of rows of plots.
switch PlotType
case 'PlotOver'
PlotRows = 1;
case 'Series'
PlotRows = NumRep;
% PlotRows = ParaIter;
end
LabelText(1:PlotColumns)={''};
% Main loop for plotting data if not 'Summary'
% Definition for MyFont added here as not inside loop of original
% definition
MyFont = 'Times New Roman';
for i=1:NumRep
% for i=1:ParaIter
for j=1:PlotColumns
% Create Plot axes; if not "Series", skip following first pass
if i == 1 || (i > 1 && strcmp(PlotType,'Series'))
Axes(j) = axes('Parent',figure1,'FontName',MyFont,...
'Position',[FromLeftP(j) FromBot(i) WidthPlot HeightPlot]);
box(Axes(j),'on');
hold(Axes(j),'all');
ylim(Axes(j),[min(ylimLo(:,j)) max(ylimHi(:,j))]);
TestPlots(j) = 0;
% Get user-specified legends for current test data plot
if ~strcmp(PlotDef(j),'none')
switch char(PlotDef(j))
case 'cWT'
TestLabel(1:length(ConcLegends),j) = ConcLegends;
case 'TgWT'
TestLabel(1:length(GasTempLeg),j) = GasTempLeg;
case 'TwWT'
TestLabel(1:length(ColTempLeg),j) = ColTempLeg;
end
end
% For test data plotover plot test data.
```

```
if strcmp(PlotDef(j),'cWT') || strcmp(PlotDef(j),'TgWT')...
|| strcmp(PlotDef(j),'TwWT')
switch length(TestData(j).a(1,:))
case 2
TestPlots(j) = 1;
% TestLabel(1,j) = {'Exit'};
plot((TestData(j).a(:,1)+OffSet(j))/3600,...
TestData(j).a(:,2),'.','Parent',Axes(j),...
'MarkerSize',10,'MarkerFaceColor','b',...
'MarkerEdgeColor','b');
case 4
TestPlots(j) = 2;
% TestLabel(1,j) = {'Mid'};
% TestLabel(2,j) = {'Exit'};
plot((TestData(j).a(:,1)+OffSet(j))/3600,...
TestData(j).a(:,2),'.','Parent',Axes(j),...
'MarkerSize',10,'MarkerFaceColor','b',...
'MarkerEdgeColor','b');
plot((TestData(j).a(:,3)+OffSet(j))/3600,...
TestData(j).a(:,4),'s','Parent',Axes(j),...
'MarkerSize',4,'MarkerFaceColor','g',...
'MarkerEdgeColor','g');
case 6
TestPlots(j) = 3;
% TestLabel(1,j) = {'Mid'};
% TestLabel(2,j) = {'Exit'};
% TestLabel(3,j) = {'Mixed'};
plot((TestData(j).a(:,1)+OffSet(j))/3600,...
TestData(j).a(:,2),'.','Parent',Axes(j),...
'MarkerSize',10,'MarkerFaceColor','b',...
'MarkerEdgeColor','b');
plot((TestData(j).a(:,3)+OffSet(j))/3600,...
TestData(j).a(:,4),'s','Parent',Axes(j),...
'MarkerSize',4,'MarkerFaceColor','g',...
'MarkerEdgeColor','g');
plot((TestData(j).a(:,5)+OffSet(j))/3600,...
TestData(j).a(:,6),'d','Parent',Axes(j),...
'MarkerSize',4,'MarkerFaceColor','r',...
'MarkerEdgeColor','r');
case 8
TestPlots(j) = 4;
% TestLabel(1,j) = {'Inlet'};
% TestLabel(2,j) = {'Mid'};
% TestLabel(3,j) = {'Exit'};
% TestLabel(4,j) = {'Mixed'};
plot((TestData(j).a(:,1)+OffSet(j))/3600,...
TestData(j).a(:,2),'.','Parent',Axes(j),...
'MarkerSize',10,'MarkerFaceColor','b',...
'MarkerEdgeColor','b');
plot((TestData(j).a(:,3)+OffSet(j))/3600,...
TestData(j).a(:,4),'s','Parent',Axes(j),...
'MarkerSize',4,'MarkerFaceColor','g',...
'MarkerEdgeColor','g');
plot((TestData(j).a(:,5)+OffSet(j))/3600,...
TestData(j).a(:,6),'d','Parent',Axes(j),...
'MarkerSize',4,'MarkerFaceColor','r',...
'MarkerEdgeColor','r');
plot((TestData(j).a(:,7)+OffSet(j))/3600,...
TestData(j).a(:,8),'^','Parent',Axes(j),...
'MarkerSize',4,'MarkerFaceColor','k',...
'MarkerEdgeColor','k');
end
end
end
% Create Plot data. Plot last location if PlotPoints(j) = 1, or sampled
% locations otherwise
if PlotPoints(j) == 1
LocIndex = 1;
PercentBed = 100;
PlotDat = PlotData(:,end,i,j);
else
% If inlet included in data...
% DeltaLoc = round((Locations)/(PlotPoints(j)-1));
% LocIndex = [1 DeltaLoc:DeltaLoc:Locations];
% If inlet not included in data...
% DeltaLoc = round((Locations)/(PlotPoints(j)));
% LocIndex = DeltaLoc:DeltaLoc:Locations;
% PercentBed = 100*LocIndex/Locations;
% PlotDat = PlotData(:,LocIndex,i,j);
% Setup Indices and Legend Text for Sim Data. The array
% indices start from 1 so 1 must be subtracted to get
% actual bed location.
DeltaLoc = round((Locations)/(PlotPoints(j)-1));
LocIndex = [0 DeltaLoc:DeltaLoc:Locations]+1;
% Special case for 1/4 inch into a 10 inch bed at each end with 200
% locations to get the right location (MSMBT)
if Locations == 200 && PlotPoints(j) == 3; LocIndex = [5 100 195]+1; end
PercentBed = 100*(LocIndex-1)/Locations;
PlotDat = PlotData(:,LocIndex,i,j);
end
% Plot Simulation Data. Set color to match location and linestyle
% unique for each parametric iteration if using PlotOver
% if NumCycle > 1 && strcmp(PlotType,'PlotOver') && PlotPoints(j) < 5
if ParaIter > 1 && strcmp(PlotType,'PlotOver') && PlotPoints(j) < 5
switch i
case 1
LineStyles = ':';
case 2
LineStyles = '--';
case 3
LineStyles = '-';
case 4
LineStyles = '-.';
end
if length(LocIndex) < 5
for k = 1:length(LocIndex)
switch k
case 1
LineColors = 'b';
case 2
LineColors = 'g';
case 3
LineColors = 'r';
case 4
LineColors = 'k';
end
plot(time/3600,PlotDat(:,k),'Parent',Axes(j),'Color',LineColors,...
'LineStyle',LineStyles,'LineWidth',1.25);
end
end
elseif PlotPoints(j) < 5
% For no parametric run or series plots, change line and
% color be unique for each location
for k = 1:length(LocIndex)
switch k
case 1
LineStyles = ':';
LineColors = 'b';
case 2
LineStyles = '--';
```

```
LineColors = 'g';
case 3
LineStyles = '-';
LineColors = 'r';
case 4
LineStyles = '-.';
LineColors = 'k';
end
plot(time/3600,PlotDat(:,k),'Parent',Axes(j),...
'Color',LineColors,'LineStyle',LineStyles,'LineWidth',1.25);
end
else
plot(time/3600,PlotDat,'Parent',Axes(j));
end
% Create xlabel on bottom plot
if i == NumCycle; xlabel(Axes(j),'Time (hours)'); end;
% if i == ParaIter; xlabel(Axes(j),'Time (hours)'); end;
% Create ylabel on midheight plot
if strcmp(PlotType,'Series')
% if i == ceil(ParaIter/2); ylabel(Axes(j),PlotYLabel(j)); end
else
ylabel(Axes(j),PlotYLabel(j));
end
% Create Legend if specified. Place parametric values after
% test plot definitions if parametric run.
%
% Legend not debugged for DataBox, so is disabled in this case
if ~strcmp(LegendLoc(j),'none') && ~strcmp(SimData(1),'DataBox') && ...
(strcmp(cell2mat(LegendLoc(j)),'NorthEast') || ...
strcmp(cell2mat(LegendLoc(j)),'NorthWest') || ...
strcmp(cell2mat(LegendLoc(j)),'SouthEast') || ...
strcmp(cell2mat(LegendLoc(j)),'SouthWest') || ...
strcmp(cell2mat(LegendLoc(j)),'North') || ...
strcmp(cell2mat(LegendLoc(j)),'South') || ...
strcmp(cell2mat(LegendLoc(j)),'East') || ...
strcmp(cell2mat(LegendLoc(j)),'West') || ...
strcmp(cell2mat(LegendLoc(j)),'Best'))
% Gather info for legend plotting during final plotting iteration
LabelNum = 0;
if (ParaIter == 1 || strcmp(PlotType,'Series')) && PlotPoints(j) < 5
for m = 1:PlotPoints(j)
LabelNum = LabelNum+1;
LabelText(LabelNum) = {[num2str(PercentBed(m),...
'%3.0f'),'%']};
end
elseif ParaIter > 1 && i == ParaIter && PlotPoints(j) < 5
for k = 1:ParaIter
for m = 1:PlotPoints(j)
LabelNum = LabelNum+1;
LabelText(LabelNum) = {[num2str(PercentBed(m),...
'%3.0f'),'% - ',num2str(ParaValues(k),2)]};
end
end
end
% Plot legend during final plotting iteration
if i == ParaIter
if TestPlots(j) ~= 0 && PlotPoints(j) < 5
LabelTextComp = {[TestLabel(1:TestPlots(j),j)' LabelText(:)']};
legend(Axes(j),LabelTextComp{1,1},'Location',cell2mat(LegendLoc(j)));
elseif TestPlots(j) ~= 0 && PlotPoints(j) > 4
LabelTextComp = {TestLabel(1:TestPlots(j),j)'};
legend(Axes(j),LabelTextComp{1,1},'Location',cell2mat(LegendLoc(j)));
elseif TestPlots(j) == 0 && PlotPoints(j) < 5
LabelTextComp = {LabelText(:)'};
legend(Axes(j),LabelTextComp{1,1},'Location',cell2mat(LegendLoc(j)));
```

```matlab
end
end
end
% Create label inside plot to show current parametric iteration
% value if plotting Series. Location in plot is based on
% ParaLoc(j) variable, unless it is "ctSpecial", then DataBox
% parameters are included in the label at the top center of the
% second plot.
if strcmp(PlotType,'Series') && ~strcmp(ParaLoc(j),'none') && ...
~strcmp(ParaLoc(1),'ctSpecial') && PlotRows > 1 && ...
(strcmp(cell2mat(ParaLoc(j)),'NorthEast') || ...
strcmp(cell2mat(ParaLoc(j)),'NorthWest') || ...
strcmp(cell2mat(ParaLoc(j)),'SouthEast') || ...
strcmp(cell2mat(ParaLoc(j)),'SouthWest'))
ParaString = {[ParaName,' = ',num2str(ParaValues(i),2),ParaUnits]};
annotation(figure1,'textbox',...
[FromLeftPara(j) FromBotPara(i) WidthPara HeightPara],...
'BackgroundColor',[1 1 1],...
'String',ParaString,'FitBoxToText','off','FontSize',10,...
'HorizontalAlignment','center','VerticalAlignment','middle',...
'Margin',0.2);
end
% Create DataBox during iteration for first plot. Location at left
% end is a feature that will removed after placement inside ct plot
% at any location is fully developed.
if (strcmp(SimData(1),'DataBox') && j == 1) || ...
strcmp(ParaLoc(j),'ctSpecial')
% For PlotOver, only one DataBox plotted on first iteration
if strcmp(PlotType,'Series') || i == 1
MaxInt = max(max(out(i).ct(3:end,SlopeMax(1):SlopeMax(2))));
MaxBT = max(out(i).ct(3:end,end));
if strcmp(ParaName,'AdsAxialDisp');
AxialDispStr = num2str(ParaValues(i),2);
LDFStr = num2str(LDF,2);
str(3) = {['Disp/LDF =',num2str(ParaValues(i)/LDF,2)]};
elseif strcmp(ParaName,'LDF')
AxialDispStr = num2str(AdsAxialDisp,2);
LDFStr = num2str(ParaValues(i),2);
str(3) = {['Disp/LDF =',num2str(AdsAxialDisp/ParaValues(i),2)]};
else
AxialDispStr = num2str(AdsAxialDisp,2);
LDFStr = num2str(LDF,2);
str(3) = {['Disp/LDF =',num2str(AdsAxialDisp/LDF,2)]};
end
str(1) = {['Axial Disp = ',AxialDispStr,'[m^2/s]']};
str(2) = {['LDF = ',LDFStr,'[1/s]']};
str(4) = {['Max Int Slope = ',num2str(MaxInt,2)]};
str(5) = {['Max BT Slope = ',num2str(MaxBT,2)]};
str(6) = {['Slope Ratio = ',num2str(MaxBT/MaxInt,3)]};
if strcmp(SimData(1),'DataBox') && j == 1
annotation(figure1,'textbox',...
[Margin FromBot(i) WidthDB HeightPlot],...
'String',str,'FitBoxToText','off','FontSize',9);
elseif strcmp(ParaLoc(j),'ctSpecial')
annotation(figure1,'textbox',...
[FromLeftPara(j) FromBotPara(i) WidthPara HeightPara],...
'String',str,'FitBoxToText','off','FontSize',9,...
'HorizontalAlignment','center','VerticalAlignment',...
'middle','Margin',0.5);
end
end
end
end
end
end
```

```
%% Print postscript to file named after RunIDPlot
filename = [RunIDPlot,'.eps'];
set(gcf, 'PaperPositionMode', 'auto') % Use screen size
print(figure1,'-depsc','-r300','-loose','-tiff', filename)
beep
end
function [LimLo, LimHi] = HiLoTestData(DataFile,LimLoSet,LimHiSet)
% Set plot limits; LimLoSet and LimHiSet sets tightness
if ~isempty(DataFile)
% Initialize axis limit variables
limHiAct = 0;
limLoAct = 1e10;
% Get actual data min/max for test data format (data in even rows with time in
odd rows)
if min(min(DataFile(:,2:2:end))) < limLoAct;
limLoAct = min(min(DataFile(:,2:2:end)));
end
if limLoAct <= 0 ; LimLo = 0 ; else
LimLo = (floor((limLoAct*10^(-floor(log10(limLoAct))))...
*10^LimLoSet)*10^-LimLoSet)*10^floor(log10(limLoAct));
end
if max(max(DataFile(:,2:2:end))) > limHiAct;
limHiAct = max(max(DataFile(:,2:2:end)));
end
LimHi = (ceil((limHiAct*10^(-floor(log10(limHiAct))))...
*10^LimHiSet)*10^-LimHiSet)*10^floor(log10(limHiAct));
end
end
function [LimLo, LimHi] = HiLoSimData(DataFile,LimLoSet,LimHiSet)
% Set plot limits; LimLoSet and LimHiSet sets tightness
if ~isempty(DataFile)
% Initialize axis limit variables
limHiAct = 0;
limLoAct = 1e10;
% Get actual data min/max (data in columns)
if min(min(DataFile)) < limLoAct;
limLoAct = min(min(DataFile));
end
if limLoAct <= 0 ; LimLo = 0 ; else
LimLo = (floor((limLoAct*10^(-floor(log10(limLoAct))))...
*10^LimLoSet)*10^-LimLoSet)*10^floor(log10(limLoAct));
end
if max(max(DataFile)) > limHiAct;
limHiAct = max(max(DataFile));
end
% For data files of all zeros, set LimHi to 1 to prevent plotting
% error. If limLoAct = limHiAct, set to +/- 10% of value
if limHiAct == 0 && limLoAct == 0
LimHi = 1;
elseif limLoAct == limHiAct
LimLo = 0.9*limLoAct;
LimHi = 1.1*limHiAct;
else
LimHi = (ceil((limHiAct*10^(-floor(log10(limHiAct))))...
*10^LimHiSet)*10^-LimHiSet)*10^floor(log10(limHiAct));
end
end
end
function PlotOnly(hObject, ~)
% Run plotting code on COMSOL data currently stored as a Matlab
% workspace (.mat file)
% Initiation code
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
RecNum = str2double(get(rh,'String'));
```

```
% Execute command to update calculations if set to Auto Calc
cmhck = findobj('Label','Auto Calc');
UserDataCheck = get(cmhck(1),'UserData');
if strcmp('on',UserDataCheck.AutoCalc)
GenCalc(hObject, 0)
end
% Change RecNum to negative number to indicate plotting only
%RecNum = -RecNum;
% Pass GenIn array to PlotCOMSOL Matlab Function
GenIn = getappdata(fh,'AppGenIn');
PlotCOMSOL(GenIn,RecNum)
% end of function
function [time, out] = PMDS_Adsorb(GenIn,RecNum)
%
% PMDS_Adsorb.m
%
% Model exported on Feb 4 2011, 15:05 by COMSOL 4.1.0.154.
import com.comsol.model.*
import com.comsol.model.util.*
model = ModelUtil.create('Model');
% model.modelPath('/Users/jcknox/Documents/My Work Files/Computer
Modeling/COMSOL/2011/COMSOL and MATLAB');
% model.name('PMDS_Adsorb.mph');
% The following Parametric values setting commands are replaced by the
% command following the comment "Call to set COMSOL parameters"
% model.param.set('Description', 'text', 'Simulation of CO2 Adsorption on 5A in
a 2-in Column');
% model.param.set('AdsInitTemp', '25.183[degC]', 'Ads Initial Temp');
% model.param.set('AdsInTemp', '25.183[degC]', 'Ads Inlet Temp');
% model.param.set('AdsInitConc', '0.001[mol/m^3]', 'Ads Initial Conc');
% model.param.set('AdsInitLoad', '1[mol/kg]', 'Ads Initial Load');
% model.param.set('FreeFlowArea', '17.814[cm^2]', 'Free Flow Area');
% model.param.set('CanCSArea', '2.45[cm^2]', 'Canister CS Area');
% model.param.set('CanIPerim', '14.96[cm]', 'Can Inner Perimeter');
% model.param.set('CanOPerim', '15.96[cm]', 'Can Outer Perimeter');
% model.param.set('BedLength', '0.254[m]', 'Bed Length');
% model.param.set('VoidFraction', '0.35', 'Void Fraction');
% model.param.set('WallVoid', '1', 'Wall Void Fraction');
% model.param.set('GasCanH', '14.1957[W/(m^2*K)]', 'Gas-Can H');
% model.param.set('CanAmbH', '1.4196[W/(m^2*K)]', 'Can-Amb H');
% model.param.set('CanCond', '0[W/(m*K)]', 'Can Cond');
% model.param.set('CanHeatCap', '475[J/(kg*K)]', 'Can Q Capac');
% model.param.set('CanDensity', '7833[kg/m^3]', 'Can Density');
% model.param.set('PartDensity', '1201[kg/m^3]', 'Part Density');
% model.param.set('LDF', '0.0023[1/s]', 'Mass Trans Coeff');
% model.param.set('SorbGasH', '20[W/(m^2*K)]', 'Sorb-Gas H');
% model.param.set('SorbCond', '0.1731[W/(m*K)]', 'Sorb Q Cond');
% model.param.set('SorbHeatCap', '1046.7[J/(kg*K)]', 'Sorb Q Capac');
% model.param.set('dHSorb', '-41.8673[kJ/mol]', 'Heat of Ads');
% model.param.set('SorbArea', '7.4255[m^2/m]', 'Ext Sorb Area');
% model.param.set('HalfCycleLength', '7000', 'Half-Cycle Length[s]');
% model.param.set('TimeStep', '30', 'Time Step[s]');
% model.param.set('NodeSepMax', '0.001[m]', 'Node Sep Max');
% model.param.set('NodeSepInit', '0.0001[m]', 'Node Sep Init');
% model.param.set('EqPelDia', '2.879[mm]', 'Equiv Pellet Dia');
% model.param.set('SorbentMassChk', '353.2251[g]', 'Sorb Mass via Void');
% model.param.set('AdsConc', '0.33018[mol/m^3]', 'Ads Concentrat');
% model.param.set('AdsAxialDisp', '0.00106[m^2/s]', 'Ads Axial Disp');
% model.param.set('AdsTotPress', '106.869[kPa]', 'Ads Total Press');
% model.param.set('AdsGasDens', '1.2122[kg/m^3]', 'Ads Gas Dens');
% model.param.set('AdsSupVel', '0.27166[m/s]', 'Ads Superfic Vel');
% model.param.set('AdsSolidConc', '1499.6326[mol/m^3]', 'Ads Solid Conc');
% model.param.set('TothA0', '9.875e-07[mol/kg/kPa]', 'Toth a0');
% model.param.set('TothB0', '6.761e-08[1/kPa]', 'Toth b0');
% model.param.set('TothE', '5625[K]', 'Toth E');
% model.param.set('TothT0', '0.27', 'Toth to');
% model.param.set('TothC', '-20.02[K]', 'Toth c');
%
% *** End COMSOL Statements ************************************
% Load output variables into character arrays to be input into param
% statements.
% Data provided by current database record is searched for needed local
% variables and then for COMSOL parameters. Notes:
% (1) The Export Flag value determines if a field is exported
% (2) If the value is a string, it is not sent to COMSOL
% (3) Units are appended to the value for the COMSOL Expression field
for i = 1:size(GenIn.NumInputs,2)
for j = 1:GenIn.NumInputs(i)
% % % Pick out needed values for plotting, etc. from GenIn array
switch char(GenIn.varname(i,j))
case 'ModelName'
ModelNSolverName = ParseInputText(GenIn.values(i,j,RecNum));
ModelSolve = cell2mat(ModelNSolverName(2));
case 'NodeSepMax'
NodeSepMax=cell2mat(GenIn.values(i,j,RecNum));
case 'BedLength'
BedLength=cell2mat(GenIn.values(i,j,RecNum));
case 'TimeStep'
TimeStep=cell2mat(GenIn.values(i,j,RecNum));
case 'HalfCycleLength'
HalfCycleLength=cell2mat(GenIn.values(i,j,RecNum));
case 'NumHC'
NumHC=cell2mat(GenIn.values(i,j,RecNum));
case 'FreeFlowArea'
FreeFlowArea=cell2mat(GenIn.values(i,j,RecNum)); %#ok<NASGU>
case 'VoidFraction'
VoidFraction=cell2mat(GenIn.values(i,j,RecNum)); %#ok<NASGU>
case 'Locations'
Locations=cell2mat(GenIn.values(i,j,RecNum));
case 'AdsInTempFile'
AdsInTempFile=cell2mat(GenIn.values(i,j,RecNum));
case 'AdsInConcFile'
AdsInConcFile=cell2mat(GenIn.values(i,j,RecNum));
case 'DesInTempFile'
DesInTempFile=cell2mat(GenIn.values(i,j,RecNum));
case 'DesInConcFile'
DesInConcFile=cell2mat(GenIn.values(i,j,RecNum));
% case 'InTemp'
% InTemp=cell2mat(GenIn.values(i,j,RecNum));
% case 'TimeLineFile'
% TimeLineFile=cell2mat(GenIn.values(i,j,RecNum));
end
if strcmp('yes',GenIn.export(i,j))
if ~iscellstr(GenIn.values(i,j,RecNum))
num = cell2mat(GenIn.values(i,j,RecNum));
values = strcat(num2str(num),GenIn.unit(i,j));
% Call to set COMSOL parameters
model.param.set(GenIn.varname(i,j),values,GenIn.string(i,j));
end
end
end
end
% *** Start COMSOL statements ************************************
model.modelNode.create('mod1');
model.func.create('step1', 'Step');
model.func.create('step2', 'Step');
model.geom.create('geom1', 1);
model.geom('geom1').feature.create('i1', 'Interval');
model.geom('geom1').feature('i1').set('p2', 'BedLength');
model.geom('geom1').run;
model.variable.create('var1');
model.variable('var1').model('mod1');
model.variable('var1').set('Tfix', 'initstep(T/1[K])*T');
model.variable('var1').set('Pgas', 'isostep(c/1[mol/m^3])*c*R_const*Tfix');
model.variable('var1').set('a_Toth', 'TothA0*exp(TothE/Tfix)');
model.variable('var1').set('b_Toth', 'TothB0*exp(TothE/Tfix)');
model.variable('var1').set('t_Toth', '(TothT0+TothC/Tfix)');
model.variable('var1').set('q_star', 'PartDensity*a_Toth*Pgas/(1+(max(1e-
99,b_Toth*Pgas))^t_Toth)^
(1/t_Toth)');
model.variable('var1').set('GasCond', '(3.6969697E-4+9.74353924E-5*(T/1[K])^1-
4.07587413E-8*(T/1[K])
^2+7.68453768E-12*(T/1[K])^3)*1[W/(m*K)]', 'COMSOL Gas Conductivity');
model.variable('var1').set('GasHeatCap', '(1088.22121-
0.365941919*(T/1[K])^1+7.88715035E-4*(T/1[K])
^2-3.749223E-7*(T/1[K])^3+3.17599068E-11*(T/1[K])^4)*1[J/(kg*K)]', 'COMSOL Gas
Heat Capacity');
%Older code being replaced - delete when satisfied with swap.
% model.variable('var1').set('Tfix', 'initstep(T)*T');
% model.variable('var1').set('Pgas', 'isostep(c)*c*R_const*Tfix');
% model.variable('var1').set('a_Toth', 'TothA0*exp(TothE/Tfix)');
% model.variable('var1').set('b_Toth', 'TothB0*exp(TothE/Tfix)');
% model.variable('var1').set('t_Toth', '(TothT0+TothC/Tfix)');
% % Following code used to check Tfix influence on mass balance - none found
% % model.variable('var1').set('Pgas', 'isostep(c)*c*R_const*T');
% % model.variable('var1').set('a_Toth', 'TothA0*exp(TothE/T)');
% % model.variable('var1').set('b_Toth', 'TothB0*exp(TothE/T)');
% % model.variable('var1').set('t_Toth', '(TothT0+TothC/T)');
% model.variable('var1').set('q_star',
'PartDensity*a_Toth*Pgas/(1+(b_Toth*Pgas)^t_Toth)^(1/t_Toth)');
% model.variable('var1').set('GasCond', '(3.6969697E-4+9.74353924E-5*T^1-
4.07587413E-8*T^2+7.
68453768E-12*T^3)*1[W/(m*K)]', 'COMSOL Gas Conductivity');
% model.variable('var1').set('GasHeatCap', '(1088.22121-
0.365941919*T^1+7.88715035E-4*T^2-3.749223
E-7*T^3+3.17599068E-11*T^4)*1[J/(kg*K)]', 'COMSOL Gas Heat Capacity');
model.physics.create('chds', 'DilutedSpecies', 'geom1');
model.physics('chds').feature.create('flux1', 'Flux', 0);
model.physics('chds').feature('flux1').selection.set([1]);
model.physics('chds').feature.create('reac1', 'Reactions', 1);
model.physics('chds').feature('reac1').selection.all;
model.physics('chds').feature.create('out1', 'Outflow', 0);
model.physics('chds').feature('out1').selection.set([2]); %#ok<*NBRAK>
model.physics.create('g', 'GeneralFormPDE', 'geom1');
model.physics('g').field('dimensionless').component({'q'});
model.physics.create('ht', 'PorousMediaHeat', 'geom1');
model.physics('ht').feature.create('hs1', 'HeatSource', 1);
model.physics('ht').feature('hs1').selection.all;
model.physics('ht').feature.create('ofl1', 'ConvectiveOutflow', 0);
model.physics('ht').feature('ofl1').selection.set([2]);
model.physics('ht').feature.create('opcc1', 'OutOfPlaneConvectiveCooling', 1);
model.physics('ht').feature('opcc1').selection.all;
model.physics('ht').feature.create('inhf1', 'InflowHeatFlux', 0);
model.physics('ht').feature('inhf1').selection.set([1]);
model.physics.create('ht2', 'HeatTransfer', 'geom1');
model.physics('ht2').feature.create('opcc1', 'OutOfPlaneConvectiveCooling', 1);
model.physics('ht2').feature('opcc1').selection.all;
model.physics('ht2').feature.create('ophf1', 'OutOfPlaneHeatFlux', 1);
model.physics('ht2').feature('ophf1').selection.all;
% *** Start Custom Statements ****************************************
if strcmp('Physics',NodeSepMax)
model.mesh.create('mesh1', 'geom1');
model.mesh('mesh1').feature.create('edg1', 'Edge');
else
model.mesh.create('mesh1', 'geom1');
model.mesh('mesh1').feature.create('size1', 'Size');
model.mesh('mesh1').feature('size1').selection.geom('geom1', 0);
model.mesh('mesh1').feature('size1').selection.set([1]);
model.mesh('mesh1').feature.create('edg1', 'Edge');
end
% *** End Custom Statements ****************************************
model.func('step1').set('funcname', 'isostep');
model.func('step1').set('location', '0.005');
model.func('step1').set('smooth', '0.01');
model.func('step2').set('funcname', 'initstep');
model.func('step2').set('location', '5');
model.func('step2').set('smooth', '10');
% *** Start Custom Statements ****************************************
% model.func('int1').set('funcs', {'TCintemp' '1'});
% model.func('int1').set('source', 'file');
% model.func('int1').set('filename', '/Users/jcknox/Documents/My Work
Files/Computer
Modeling/COMSOL/2011/COMSOL and MATLAB/Thermal
Characterization/ThermCharTestIn.txt');
% if strcmp(InTemp,'Constant') % do nothing
% elseif strcmp(InTemp,'Timelined') ||strcmp(InTemp,'Timelined_Des')
% model.func('int1').set('funcs', {'TCintemp' '1'});
% model.func('int1').set('source', 'file');
% model.func('int1').set('filename', TimeLineFile);
% else
% errordlg('Specify Inlet Temperature Control','Fatal Error');
% end
% *** End Custom Statements ****************************************
model.physics('chds').prop('ConvectiveTerm').set('ConvectiveTerm', 'cons');
model.physics('chds').prop('MassConsistentStabilization').set('massStreamlineDi
ffusion', '0');
model.physics('chds').prop('MassConsistentStabilization').set('massCrosswindDif
fusion', '0');
% Vel/Temp Adj Check
model.physics('chds').feature('cdm1').set('u',
{'AdsSupVel*T/(AdsInTemp*VoidFraction)'; '0'; '0'});
% model.physics('chds').feature('cdm1').set('u', {'AdsSupVel/VoidFraction';
'0'; '0'});
model.physics('chds').feature('cdm1').set('D', {'AdsAxialDisp' '0' '0' '0'
'AdsAxialDisp' '0' '0' '0'
'AdsAxialDisp'});
model.physics('chds').feature('init1').set('c', 'AdsInitConc');
model.physics('chds').feature('flux1').set('species', '1');
% model.physics('chds').feature('flux1').set('N0',
'AdsConc*initstep(t)*AdsSupVel/VoidFraction');
model.physics('chds').feature('reac1').set('R', '-qt*(1-
VoidFraction)/VoidFraction');
model.physics('g').prop('Units').set('DependentVariableQuantity',
'concentration');
model.physics('g').prop('Units').set('CustomSourceTermUnit', 'mol/m^3/s');
model.physics('g').feature('gfeq1').set('f', 'LDF*(q_star-q)');
model.physics('g').feature('gfeq1').set('Ga', '0');
model.physics('g').feature('init1').set('q', 'AdsInitLoad');
model.physics('ht').prop('EquationForm').set('showAllModelInputs', '1');
model.physics('ht').prop('outOfPlaneProperty').set('outOfPlaneProperty', '1');
model.physics('ht').prop('Ac').set('Ac', 'FreeFlowArea');
model.physics('ht').prop('Pc').set('Pc', 'CanIPerim');
model.physics('ht').prop('HeatConsistentStabilization').set('heatStreamlineDiff
usion', '0');
model.physics('ht').feature('pm1').set('k_p_mat', 'userdef');
model.physics('ht').feature('pm1').set('k_p', {'SorbCond'; '0'; '0'; '0';
'SorbCond'; '0'; '0'; '0'; 'SorbCond'});
model.physics('ht').feature('pm1').set('rho_p_mat', 'userdef');
model.physics('ht').feature('pm1').set('rho_p', 'PartDensity+q*SorbGasMW');
model.physics('ht').feature('pm1').set('C_pp_mat', 'userdef');
model.physics('ht').feature('pm1').set('C_pp',
'(SorbHeatCap*PartDensity+q*SorbGasMW*SorbGasHeatCap)/(PartDensity+q*SorbGasMW)
');
model.physics('ht').feature('pm1').set('theta_p', '(1-VoidFraction)');
model.physics('ht').feature('pm1').set('minput_pressure', 'AdsTotPress');
model.physics('ht').feature('fluid1').set('k_mat', 'userdef');
model.physics('ht').feature('fluid1').set('k', {'GasCond'; '0'; '0'; '0';
'GasCond'; '0'; '0'; '0'; 'GasCond'});
model.physics('ht').feature('fluid1').set('rho_mat', 'userdef');
% Vel/Temp Adj Check
model.physics('ht').feature('fluid1').set('rho', 'AdsGasDens*AdsInTemp/T');
% model.physics('ht').feature('fluid1').set('rho', 'AdsGasDens');
model.physics('ht').feature('fluid1').set('Cp_mat', 'userdef');
model.physics('ht').feature('fluid1').set('Cp', 'GasHeatCap');
model.physics('ht').feature('fluid1').set('gamma_mat', 'userdef');
model.physics('ht').feature('fluid1').set('gamma', '1.4');
model.physics('ht').feature('fluid1').set('Rs_mat', 'userdef');
model.physics('ht').feature('fluid1').set('Rs', '1.4');
model.physics('ht').feature('fluid1').set('Mn_mat', 'userdef');
model.physics('ht').feature('fluid1').set('Mn', '.028');
model.physics('ht').feature('fluid1').set('gasConstantType', 'numberAve');
model.physics('ht').feature('fluid1').set('fluidType', 'idealGas');
model.physics('ht').feature('fluid1').set('minput_pressure', 'AdsTotPress');
% Vel/Temp Adj Check
model.physics('ht').feature('fluid1').set('minput_velocity',
{'AdsSupVel*T/AdsInTemp'; '0'; '0'});
% model.physics('ht').feature('fluid1').set('minput_velocity', {'AdsSupVel';
'0'; '0'});
model.physics('ht').feature('init1').set('T', 'AdsInitTemp');
model.physics('ht').feature('hs1').set('Q', '-(1-VoidFraction)*dHSorb*qt');
model.physics('ht').feature('opcc1').set('h_z', 'WallVoid*GasCanH');
model.physics('ht').feature('opcc1').set('Text_z', 'T2');
model.physics('ht').feature('opcc1').set('Lpl', 'BedLength');
model.physics('ht').feature('opcc1').set('Uext', 'AdsSupVel/VoidFraction');
model.physics('ht').feature('opcc1').set('pA', 'TotPress');
% *** Start Custom Statements ****************************************
%model.physics('ht').feature('inhf1').set('Text', 'AdsInTemp');
% if strcmp(InTemp,'Constant') || strcmp(InTemp,'Timelined_Des')
% model.physics('ht').feature('inhf1').set('Text', 'AdsInTemp');
% elseif strcmp(InTemp,'Timelined')
% model.physics('ht').feature('inhf1').set('Text', 'TCintemp(t)');
% else
% errordlg('Specify Inlet Temperature Control','Fatal Error');
% end
% *** Start Custom Statements ****************************************
%model.func.create('int1', 'Interpolation');
if strcmp(AdsInTempFile,'Constant')
model.physics('ht').feature('inhf1').set('Text', 'AdsInTemp');
else
model.func.create('int1', 'Interpolation');
model.func('int1').set('funcs', {'AdsInTempFunc' '1'});
model.func('int1').set('source', 'file');
model.func('int1').set('filename', AdsInTempFile);
model.physics('ht').feature('inhf1').set('Text', 'AdsInTempFunc(t)');
end
if strcmp(AdsInConcFile,'Constant')
model.physics('chds').feature('flux1').set('N0',
'AdsConc*initstep(t)*AdsSupVel/VoidFraction');
else
model.func.create('int2', 'Interpolation');
model.func('int2').set('funcs', {'AdsInConcFunc' '1'});
model.func('int2').set('source', 'file');
model.func('int2').set('filename', AdsInConcFile);
model.physics('chds').feature('flux1').set('N0',
'AdsInConcFunc(t)*initstep(t)*AdsSupVel/VoidFraction');
```

```
end
% *** End Custom Statements ****************************************
model.physics('ht').feature('inhf1').set('pext', 'AdsTotPress');
model.physics('ht2').prop('outOfPlaneProperty').set('outOfPlaneProperty', '1');
model.physics('ht2').prop('Ac').set('Ac', 'CanCSArea');
model.physics('ht2').prop('Pc').set('Pc', 'CanIPerim');
model.physics('ht2').prop('HeatConsistentStabilization').set('heatStreamlineDif
fusion', '0');
model.physics('ht2').feature('solid1').set('k_mat', 'userdef');
model.physics('ht2').feature('solid1').set('k', {'CanCond'; '0'; '0'; '0';
'CanCond'; '0'; '0'; '0'; 'CanCond'});
model.physics('ht2').feature('solid1').set('rho_mat', 'userdef');
model.physics('ht2').feature('solid1').set('rho', 'CanDensity');
model.physics('ht2').feature('solid1').set('Cp_mat', 'userdef');
model.physics('ht2').feature('solid1').set('Cp', 'CanHeatCap');
model.physics('ht2').feature('solid1').set('minput_pressure', 'TotPress');
model.physics('ht2').feature('init1').set('T2', 'AdsInitTemp');
model.physics('ht2').feature('opcc1').set('h_z', 'WallVoid*GasCanH');
model.physics('ht2').feature('opcc1').set('Text_z', 'T');
model.physics('ht2').feature('opcc1').set('Lpl', 'BedLength');
model.physics('ht2').feature('opcc1').set('Uext', 'AdsSupVel/VoidFraction');
model.physics('ht2').feature('opcc1').set('pA', 'TotPress');
model.physics('ht2').feature('ophf1').set('HeatFluxType', 'InwardHeatFlux');
model.physics('ht2').feature('ophf1').set('h_z',
'CanOPerim/CanIPerim*CanAmbH');
model.physics('ht2').feature('ophf1').set('Text_z', 'AmbTemp');
% *** Start Custom Statements **************************************
if strcmp('Physics',NodeSepMax)
model.mesh('mesh1').feature('size').set('hauto', 1);
model.mesh('mesh1').run;
else
model.mesh('mesh1').feature('size').set('hmax', 'NodeSepMax');
model.mesh('mesh1').feature('size').set('hmin', '3.05E-6');
model.mesh('mesh1').feature('size').set('hcurve', '0.2');
model.mesh('mesh1').feature('size').set('hgrad', '1.1');
model.mesh('mesh1').feature('size').set('hauto', '1');
model.mesh('mesh1').feature('size').set('hmax', 'NodeSepMax');
model.mesh('mesh1').feature('size').set('hmin', '3.05E-6');
model.mesh('mesh1').feature('size1').set('hmax', 'NodeSepInit');
model.mesh('mesh1').feature('size1').set('hmin', '4.59E-5');
model.mesh('mesh1').feature('size1').set('hminactive', false);
model.mesh('mesh1').feature('size1').set('hcurveactive', false);
model.mesh('mesh1').feature('size1').set('hnarrowactive', false);
model.mesh('mesh1').feature('size1').set('hgrad', '1.01');
model.mesh('mesh1').feature('edg1').name('Edge 1a');
model.mesh('mesh1').run;
end
% *** End Custom Statements ****************************************
% Prior version used this sequence above, differs frome the PMDS.m. But
% there does not appear to be a difference in results. Leave the full
% version for conservatism.
% else
% model.mesh('mesh1').feature('size').set('hmax', 'NodeSepMax');
% model.mesh('mesh1').feature('size1').set('hmax', 'NodeSepInit');
% model.mesh('mesh1').feature('size1').set('hminactive', false);
% model.mesh('mesh1').feature('size1').set('hcurveactive', false);
% model.mesh('mesh1').feature('size1').set('hnarrowactive', false);
% model.mesh('mesh1').feature('size1').set('hgradactive', false);
% model.mesh('mesh1').run;
% end
model.study.create('std1');
model.study('std1').feature.create('time', 'Transient');
model.sol.create('sol1');
model.sol('sol1').study('std1');
model.sol('sol1').attach('std1');
model.sol('sol1').feature.create('st1', 'StudyStep');
model.sol('sol1').feature.create('v1', 'Variables');
model.sol('sol1').feature.create('t1', 'Time');
model.sol('sol1').feature('t1').feature.create('fc1', 'FullyCoupled');
model.sol('sol1').feature('t1').feature.create('d1', 'Direct');
% *** Start Custom Statements ****************************************
if strcmp(ModelSolve,'SEG')
model.sol('sol1').feature('t1').feature.create('se1', 'Segregated');
model.sol('sol1').feature('t1').feature('se1').feature.create('ss1',
'SegregatedStep');
end
% *** End Custom Statements ****************************************
model.sol('sol1').feature('t1').feature.remove('fcDef');
% model.study('std1').feature('time').set('mesh', {'geom1' 'mesh1'});
% *** Start Custom Statements ****************************************
% model.study('std1').feature('time').set('physselection', 'g');
% model.study('std1').feature('time').set('activate', {'chds' 'off' 'g' 'off'
'ht' 'on' 'ht2' 'on'});
% if strcmp(InTemp,'Constant') || strcmp(InTemp,'Timelined_Des')
model.study('std1').feature('time').set('physselection', 'g');
% elseif strcmp(InTemp,'Timelined')
% turn off adsorption
% model.study('std1').feature('time').set('physselection', 'g');
% model.study('std1').feature('time').set('activate', {'chds' 'off' 'g' 'off'
'ht' 'on' 'ht2' 'on'});
% else
% errordlg('Specify Inlet Temperature Control','Fatal Error');
% end
% *** End Custom Statements ****************************************
model.study('std1').feature('time').set('tlist',
'range(0,TimeStep,HalfCycleLength)');
model.sol('sol1').feature('st1').name('Compile Equations: Time Dependent');
model.sol('sol1').feature('st1').set('studystep', 'time');
model.sol('sol1').feature('v1').set('control', 'time');
model.sol('sol1').feature('t1').set('tlist',
'range(0,TimeStep,HalfCycleLength)');
model.sol('sol1').feature('t1').set('rtol', '0.001');
model.sol('sol1').feature('t1').set('fieldselection', 'mod1_T');
model.sol('sol1').feature('t1').set('atolmethod', {'mod1_T' 'global' 'mod1_T2'
'global' 'mod1_c' 'global'
'mod1_q' 'global'});
model.sol('sol1').feature('t1').set('atol', {'mod1_T' '1e-3' 'mod1_T2' '1e-3'
'mod1_c' '1e-3' 'mod1_q' '1e-3'});
model.sol('sol1').feature('t1').set('atoludot', {'mod1_T' '1e-3' 'mod1_T2' '1e-
3' 'mod1_c' '1e-3' 'mod1_q' '1e-
3'});
model.sol('sol1').feature('t1').set('atoludotactive', {'mod1_T' 'off' 'mod1_T2'
'off' 'mod1_c' 'off' 'mod1_q'
'off'});
model.sol('sol1').feature('t1').set('maxorder', '2');
model.sol('sol1').feature('t1').feature('fc1').set('dtech', 'hnlin');
model.sol('sol1').feature('t1').feature('fc1').set('maxiter', '100');
% *** Start Custom Statements ****************************************
if strcmp(ModelSolve,'SEG')
model.sol('sol1').feature('t1').feature('se1').set('segterm', 'itertol');
model.sol('sol1').feature('t1').feature('se1').set('segiter', '100');
model.sol('sol1').feature('t1').feature('se1').feature('ssDef').set('segvar',
{'mod1_c' 'mod1_q'});
model.sol('sol1').feature('t1').feature('se1').feature('ssDef').set('subdtech',
'hnlin');
model.sol('sol1').feature('t1').feature('se1').feature('ss1').set('segvar',
{'mod1_T' 'mod1_T2'});
model.sol('sol1').feature('t1').feature('se1').feature('ss1').set('subdtech',
'auto');
end
```

```
% First or Single run of model. Show Progress bar during execution.
ModelUtil.showProgress(true);
model.sol('sol1').runAll;
% out = model;
% Extract COMSOL data
time = mphinterp(model,{'t'},'coord',0);
Coords = [0 BedLength/Locations:(BedLength-BedLength/Locations)/(Locations-
1):BedLength];
i=1;
[out(i).c,out(i).ct,out(i).q,out(i).qt,out(i).Tg,out(i).Tw] = ...
mphinterp(model,{'c','ct','q','qt','T','T2'},'coord',Coords);
% Cyclic run of model following first run
if NumHC > 1
% Append HC two time to existing time array. Approach is to plot all
% data vs. cycle time to easily observe approach to cyclic steady state
% and perform cycle-based calculations.
HC2Time=HalfCycleLength+TimeStep:TimeStep:HalfCycleLength*2;
time = vertcat(time,HC2Time');
% Set inital conditions to results of previous solution
model.sol('sol1').feature('v1').set('initsol', 'sol1');
model.sol('sol1').feature('v1').set('initmethod', 'sol');
%Set up desorption functions
if strcmp(DesInTempFile,'Constant') % do not create function
else
model.func.create('int3', 'Interpolation');
model.func('int3').set('funcs', {'DesInTempFunc' '1'});
model.func('int3').set('source', 'file');
model.func('int3').set('filename', DesInTempFile);
end
if strcmp(DesInConcFile,'Constant') % do not create function
else
model.func.create('int4', 'Interpolation');
model.func('int4').set('funcs', {'DesInConcFunc' '1'});
model.func('int4').set('source', 'file');
model.func('int4').set('filename', DesInConcFile);
end
% Start HC iterations
for HC = 2:NumHC
HC %#ok<NOPRT>
if int8(HC/2)*2 == HC % HCmode = 'Desorb'; Countercurrent flow. Cocurrent flow
to be
implemented.....
Cycle = HC/2; % Set Cycle to build arrays
model.sol('sol1').feature('t1').set('tlist',
'range(HalfCycleLength,TimeStep,HalfCycleLength*2)');
model.physics('chds').feature('flux1').selection.set([2]);
model.physics('chds').feature('out1').selection.set([1]);
model.physics('ht').feature('ofl1').selection.set([1]);
model.physics('ht').feature('inhf1').selection.set([2]);
model.physics('chds').feature('cdm1').set('u', {'-
DesSupVel*T/(DesInTemp*VoidFraction)'; '0'; '0'});
model.physics('chds').feature('init1').set('c', 'DesInitConc');
% model.physics('chds').feature('flux1').set('N0', 'DesConc*initstep(t/1[s])
*DesSupVel/VoidFraction');
model.physics('g').feature('init1').set('q', 'DesInitLoad');
model.physics('ht').feature('fluid1').set('minput_pressure', 'DesTotPress');
model.physics('ht').feature('fluid1').set('minput_velocity', {'-
DesSupVel*T/DesInTemp'; '0'; '0'});
model.physics('ht').feature('init1').set('T', 'DesInTemp');
model.physics('ht').feature('inhf1').set('pext', 'DesTotPress');
model.physics('ht2').feature('init1').set('T2', 'DesInTemp');
% Set inlet conditions
if strcmp(DesInTempFile,'Constant')
model.physics('ht').feature('inhf1').set('Text', 'DesInTemp');
else
```

```
model.physics('ht').feature('inhf1').set('Text', 'DesInTempFunc(t)');
end
if strcmp(DesInConcFile,'Constant')
model.physics('chds').feature('flux1').set('N0', 'DesConc*initstep(t/1[s])
*DesSupVel/VoidFraction');
else
model.physics('chds').feature('flux1').set('N0',
'DesInConcFunc(t)*initstep(t/1[s])
*DesSupVel/VoidFraction');
end
% Cyclic run of model
model.sol('sol1').runAll;
% Extract HC 2 COMSOL data and append to Adsorption data. Note
% that the first (zeroth) data point is skipped in the output
% array as it should simply be intial conditions from the HC 1
% run.
[Des.c,Des.ct,Des.q,Des.qt,Des.Tg,Des.Tw] = ...
mphinterp(model,{'c','ct','q','qt','T','T2'},'coord',Coords);
out(Cycle).c = vertcat(out(Cycle).c,Des.c(2:end,:));
out(Cycle).ct = vertcat(out(Cycle).ct,Des.ct(2:end,:));
out(Cycle).q = vertcat(out(Cycle).q,Des.q(2:end,:));
out(Cycle).qt = vertcat(out(Cycle).qt,Des.qt(2:end,:));
out(Cycle).Tg = vertcat(out(Cycle).Tg,Des.Tg(2:end,:));
out(Cycle).Tw = vertcat(out(Cycle).Tw,Des.Tw(2:end,:));
else % HCmode = 'Adsorb';
Cycle = (HC + 1)/2; % Set Cycle to build arrays
model.sol('sol1').feature('t1').set('tlist',
'range(0,TimeStep,HalfCycleLength)');
model.physics('chds').feature('flux1').selection.set([1]);
model.physics('chds').feature('out1').selection.set([2]);
model.physics('ht').feature('ofl1').selection.set([2]);
model.physics('ht').feature('inhf1').selection.set([1]);
model.physics('chds').feature('cdm1').set('u',
{'AdsSupVel*T/(AdsInTemp*VoidFraction)'; '0'; '0'});
model.physics('chds').feature('init1').set('c', 'AdsInitConc');
% model.physics('chds').feature('flux1').set('N0', 'AdsConc*initstep(t/1[s])
*AdsSupVel/VoidFraction');
model.physics('g').feature('init1').set('q', 'AdsInitLoad');
model.physics('ht').feature('fluid1').set('minput_pressure', 'AdsTotPress');
model.physics('ht').feature('fluid1').set('minput_velocity',
{'AdsSupVel*T/AdsInTemp'; '0'; '0'});
model.physics('ht').feature('init1').set('T', 'AdsInTemp');
% model.physics('ht').feature('inhf1').set('Text', 'AdsInTemp');
model.physics('ht').feature('inhf1').set('pext', 'AdsTotPress');
model.physics('ht2').feature('init1').set('T2', 'AdsInTemp');
% Set inlet conditions
if strcmp(AdsInTempFile,'Constant')
model.physics('ht').feature('inhf1').set('Text', 'AdsInTemp');
else
model.physics('ht').feature('inhf1').set('Text', 'AdsInTempFunc(t)');
end
if strcmp(AdsInConcFile,'Constant')
model.physics('chds').feature('flux1').set('N0', 'AdsConc*initstep(t/1[s])
*AdsSupVel/VoidFraction');
else
model.physics('chds').feature('flux1').set('N0',
'AdsInConcFunc(t)*initstep(t/1[s])
*AdsSupVel/VoidFraction');
end
% Cyclic run of model
model.sol('sol1').runAll;
% Extract COMSOL data
[out(Cycle).c,out(Cycle).ct,out(Cycle).q,out(Cycle).qt,out(Cycle).Tg,out(Cycle)
.Tw] = ...
mphinterp(model,{'c','ct','q','qt','T','T2'},'coord',Coords);
```

```
end
% Cyclic run of model
% model.sol('sol1').runAll;
% % Extract COMSOL data
% % time = mphinterp(model,{'t'},'coord',0);
% % Coords = [0 BedLength/Locations:(BedLength-BedLength/Locations)/(Locations-
1):BedLength];
% [out(HC).c,out(HC).ct,out(HC).q,out(HC).qt,out(HC).Tg,out(HC).Tw] = ...
% mphinterp(model,{'c','ct','q','qt','T','T2'},'coord',Coords);
end
end
end
% Integration for bed loading; not used following verification of
% calculation in PlotCOMSOL routine
% qInt = mphint(model,'q','Edim',1,'Selection','all','T',HalfCycleLength);
% TotMolSorb = qInt*FreeFlowArea*(1-VoidFraction)/10000
% 'coord',...
% BedLength/Locations:(BedLength-BedLength/Locations)/(Locations-1):BedLength);
function RunSimulation(hObject, ~)
% Extract data parameters and send to specialized Matlab function with
% imbedded COMSOL model specified by the ModelName variable.
%!!! change in graphics broke this. Fixing for interactive mode only at
% present time.
% % Interactive mode:
% % eventdata is empty
% % hObject is GUI handle created by Matlab
%
% % Command line mode:
% % eventdata contains RecNum
% % hObject is passed figure handle
%
% if length(eventdata) %#ok<ISMT>
% RecNum = eventdata;
% fh = hObject;
% mode = 'CommandLine';
% else
fh = ancestor(hObject,'figure');
rh = findobj(fh,'Tag','Rec_Number');
RecNum = str2double(get(rh,'String'));
% Execute command to update calculations if set to Auto Calc
cmhck = findobj('Label','Auto Calc');
UserDataCheck = get(cmhck(1),'UserData');
if strcmp('on',UserDataCheck.AutoCalc)
GenCalc(hObject, 0)
end
mode = 'Interactive';
% end
GenIn = getappdata(fh,'AppGenIn');
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ModelName'));
ModelNSolverName = ParseInputText(GenIn.values(HeNu,ItNu,RecNum));
ModelName = cell2mat(ModelNSolverName(1));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'RunID'));
RunID = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
[HeNu,ItNu]=find(strcmp(GenIn.varname,'NumHC'));
NumHC = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
% Parametric Iterations input can contain 1 or 2 values. Parse this input
% to set ParaIter to the first value, and ParaIter2 to the second. Same for
% ParaName and ParaMax. ParaMin is already expected to possibly possess
% more than one variable
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ParaIter'));
ParaIterAll = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
ParaIter = ParaIterAll(1);
if length(ParaIterAll) == 2; ParaIter2 = ParaIterAll(2); end
% Find header and item numbers for ParaMin
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ParaMin'));
```

```
% Get value of ParaMin
ParaMin = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
% Find header and item numbers for specified parameter
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ParaName'));
% Get name of parameters specified to be varied
ParaNameAll = ParseInputText(GenIn.values(HeNu,ItNu,RecNum));
ParaName = ParaNameAll(1);
if length(ParaNameAll) == 2; ParaName2 = ParaNameAll(2); end
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ParaMax'));
% Get name of parameters specified to be varied
ParaMaxAll = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
ParaMax = ParaMaxAll(1);
if length(ParaMaxAll) == 2; ParaMax2 = ParaMaxAll(2); end
% For Cyclic Runs, cycle between user specified Adsorption and Desorption
% inlet conditions. Set model boundaries based on flow rate, which is
% positive for cocurrent and negative for countercurrent flow. Use previous
% results to build cyclic data array for plotting.
if NumHC > 1
% If ParaIter is 1, use value of ParaMin to set a normally calculated
% parameter without disabling calc
if ParaIter == 1
% Get header and item number for specified parameter
[HeNu,ItNu]=find(strcmp(GenIn.varname,ParaName));
% put in GUI
ch = findobj(fh,'Tag',ParaName);
set(ch,'String',ParaMin);
% Set specified parameter to current value
GenIn.values(HeNu,ItNu,RecNum)={ParaMin};
% Save current values to GenIn array
setappdata(fh,'AppGenIn',GenIn);
end
% Execute code for cyclic COMSOL runs
% Send input parameters to specified model
switch ModelName
case 'PDE'
[time, out] = PDE_Adsorb(GenIn,RecNum);
case 'PMDS'
[time, out] = PMDS_Adsorb(GenIn,RecNum);
case 'PMDS_Ads_NC'
[time, out] = PMDS_Ads_NC(GenIn,RecNum);
end
ErrorToFile(hObject,time,out)
end
% For single run (ParaIter=0), execute code with current database values
if ParaIter == 0 && NumHC == 1
switch ModelName
case 'PDE'
[time, out] = PDE_Adsorb(GenIn,RecNum);
case 'PMDS'
[time, out] = PMDS_Adsorb(GenIn,RecNum);
case 'PMDS_Ads_NC'
[time, out] = PMDS_Ads_NC(GenIn,RecNum);
end
ErrorToFile(hObject,time,out)
end
% For Parametric Iterations, Modify Parametric Variable and repeat
% simulation specified number of times. This location can also be used to
% set a normally calculated parameter without disabling calc or if in use
% in command line mode, such as with X-TOOLSS
if ParaIter >= 1 && NumHC == 1
[HeNu,ItNu]=find(strcmp(GenIn.varname,'ParaMin'));
ParaMin = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
if ParaIter == 1
% For value of 1: use to replace calculated values with manual
% entry or entries of the parameteric
% values.
ParaValues=ParaMin;
elseif ParaIter > 1
if strcmp('Interactive',mode) % Do not want parametrics within X-TOOLSS runs
% Setup range of values (now done above)
%[HeNu,ItNu]=find(strcmp(GenIn.varname,'ParaMax'));
%ParaMax = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
if length(ParaMin) == 1
ParaValues=ParaMin:(ParaMax-ParaMin)/(ParaIter-1):ParaMax;
else
% Manual Assignments
ParaValues=ParaMin;
end
end
end
%[HeNu,ItNu]=find(strcmp(GenIn.varname,'ParaName'));(done above now)
% Get name of parameter specified to be varied
%ParaName = cell2mat(GenIn.values(HeNu,ItNu,RecNum));(done above now)
% Find header and item numbers for specified parameter
% ParseInputText(ParaName);
% For two Parameter Names, setup array around original array for second
% dimension
if length(ParaNameAll) == 2
ParaMin2 = ParaMin(2);
ParaValues2=ParaMin2:(ParaMax2-ParaMin2)/(ParaIter2-1):ParaMax2;
ParaValues=ParaMin:(ParaMax-ParaMin)/(ParaIter-1):ParaMax;
end
% Original (now outer) loop
% Loop for parametric COMSOL runs
for i=1:ParaIter
[HeNu,ItNu]=find(strcmp(GenIn.varname,ParaName));
% Set specified parameter to current value
GenIn.values(HeNu,ItNu,RecNum)={ParaValues(i)};
%Put parameter value in GUI
ch = findobj(fh,'Tag',ParaName);
set(ch,'String',{ParaValues(i)});
% Save current values to GenIn array
setappdata(fh,'AppGenIn',GenIn);
% New inner loop for second variable
if length(ParaNameAll) ~= 2; ParaIter2 = 1; end
for j=1:ParaIter2
if length(ParaNameAll) == 2;
[HeNu,ItNu]=find(strcmp(GenIn.varname,ParaName2));
% Set specified parameter to current value
GenIn.values(HeNu,ItNu,RecNum)={ParaValues2(j)};
%Put parameter value in GUI
ch = findobj(fh,'Tag',ParaName2);
set(ch,'String',{ParaValues2(j)});
% Save current values to GenIn array
setappdata(fh,'AppGenIn',GenIn);
end
% Send input parameters to specified model
switch ModelName
case 'PDE'
[time, out(i)] = PDE_Adsorb(GenIn,RecNum);
case 'PMDS'
[time, out(i)] = PMDS_Adsorb(GenIn,RecNum);
case 'PMDS_Ads_NC'
[time, out] = PMDS_Ads_NC(GenIn,RecNum);
end
ErrorToFile(hObject,time,out(i))
end
end
end
% Code below has been moved to ErrorToFile function
% % Call routines to calculate errors between simulation and test
%
% % Get user settings (ScopeErr, ScaleErr)
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'ScopeErr'));
% ScopeErr = GenIn.values(HeNu,ItNu,RecNum);
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'ScaleErr'));
% ScaleErr = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'HalfCycleLength'));
% HalfCycleLength = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
% SumSimTest = 0;
% NumScaleErr = 0;
% CalcOffSet = [];
%
% % Calculate errors in concentration profiles
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'ConcSSEFile'));
% ConcSSEFile = GenIn.values(HeNu,ItNu,RecNum);
% if ~strcmp(cell2mat(ConcSSEFile),'none')
% [ConcSimTestSSE, ConcOffSet] = ErrFun(time,out,ConcSSEFile,ScopeErr);
% NumScaleErr = length(ConcSimTestSSE);
% SumSimTest = sum(ScaleErr(1:NumScaleErr).*ConcSimTestSSE);
% NumScaleErr = NumScaleErr + 1;
% SumSimTest = SumSimTest +
ScaleErr(NumScaleErr)*sum(abs(ConcOffSet))./HalfCycleLength;
% CalcOffSet = ConcOffSet;
% else
% ConcSimTestSSE = 0;
% ConcOffSet = 0;
% end
%
% % Calculate errors in gas temperature profiles
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'GasTempSSEFile'));
% GasTempSSEFile = GenIn.values(HeNu,ItNu,RecNum);
% if ~strcmp(cell2mat(GasTempSSEFile),'none')
% [GasSimTestSSE, GasTempOffSet] = ErrFun(time,out,GasTempSSEFile,ScopeErr);
% NumScaleErrEnd = NumScaleErr + length(GasSimTestSSE);
% SumSimTest = SumSimTest +
sum(ScaleErr(NumScaleErr+1:NumScaleErrEnd).*GasSimTestSSE);
% NumScaleErr = NumScaleErrEnd + 1;
% SumSimTest = SumSimTest +
ScaleErr(NumScaleErr)*sum(abs(GasTempOffSet))./HalfCycleLength;
% CalcOffSet = [CalcOffSet GasTempOffSet];
% else
% GasSimTestSSE = 0;
% GasTempOffSet = 0;
% end
% % Calculate errors in column temperature profiles
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'ColTempSSEFile'));
% ColTempSSEFile = GenIn.values(HeNu,ItNu,RecNum);
% if ~strcmp(cell2mat(ColTempSSEFile),'none')
% [ColSimTestSSE, ColTempOffSet] = ErrFun(time,out,ColTempSSEFile,ScopeErr);
% NumScaleErrEnd = NumScaleErr + length(ColSimTestSSE);
% SumSimTest = SumSimTest +
sum(ScaleErr(NumScaleErr+1:NumScaleErrEnd).*ColSimTestSSE);
% NumScaleErr = NumScaleErrEnd + 1;
% SumSimTest = SumSimTest +
ScaleErr(NumScaleErr)*sum(abs(ColTempOffSet))./HalfCycleLength;
% CalcOffSet = [CalcOffSet ColTempOffSet];
% else
% ColSimTestSSE = 0;
% ColTempOffSet = 0;
% end
%
% % Write sum of errors and OffSet to GUI, array
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'SimTestErr'));
% GenIn.values{HeNu,ItNu,RecNum} = SumSimTest;
% ch = findobj(fh,'Tag','SimTestErr');
% set(ch,'String',SumSimTest);
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'CalcOffSet'));
% GenIn.values{HeNu,ItNu,RecNum} = num2str(CalcOffSet);
% ch = findobj(fh,'Tag','CalcOffSet');
% set(ch,'String',num2str(CalcOffSet));
%
% % Save current values to GenIn array
% setappdata(fh,'AppGenIn',GenIn);
%
% % Save error value to file for X-TOOLSS, also file to gather BT sharpening
% % data
% if strcmp('CommandLine',mode)
% dlmwrite('Err.txt',SumSimTest);
% end
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'SlopeMax'));
% SlopeMax = ParseInput(GenIn.values(HeNu,ItNu,RecNum));
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'AdsAxialDisp'));
% AdsAxialDisp = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
% [HeNu,ItNu]=find(strcmp(GenIn.varname,'LDF'));
% LDF = cell2mat(GenIn.values(HeNu,ItNu,RecNum));
% ctMaxInt = max(max(out(1).ct(3:end,SlopeMax(1):SlopeMax(2))));
% ctMaxBT = max(out(1).ct(3:end,end));
% BTSharpening = [SumSimTest ConcSimTestSSE ConcOffSet GasSimTestSSE
GasTempOffSet ...
% ColSimTestSSE ColTempOffSet LDF AdsAxialDisp ctMaxInt ctMaxBT];
% dlmwrite('BTSharpening.csv',BTSharpening,'-append');
% Save data to mat file named after RunID if use specified
mh = findobj('Label','Auto Save .mat file');
AutoSaveMatCheck = get(mh,'Checked');
if strcmp('on',AutoSaveMatCheck)
save([RunID,'.mat',],'GenIn','NumHC','out','time');
% Plot Solution if user specified
mh = findobj('Label','Auto Plot');
AutoPlotCheck = get(mh,'Checked');
if strcmp('on',AutoPlotCheck)
PlotCOMSOL(GenIn,RecNum)
end
end
end
% end of function
function ToggleAutoCalc(~, ~)
% Toggle autocalculate
% Initiation code
%fh = ancestor(hObject,'figure');
% Execute command to update menu checkmark and set UserData.AutoCalc
if strcmp(get(gcbo, 'Checked'),'on')
set(gcbo, 'Checked', 'off');
UserData.AutoCalc = 'off';
else
set(gcbo, 'Checked', 'on');
UserData.AutoCalc = 'on';
end
set(gcbo,'UserData',UserData);
% end of function
function ToggleAutoSaveMat(hObject, ~)
% Toggle Auto SaveMat
% Execute command to update menu checkmark and set UserData.AutoSaveMat. If
% turning off also turn off 'Auto Plot' since plot PlotCOMSOL opens the
% .mat file to obtain plotting configuration and solution results
if strcmp(get(gcbo, 'Checked'),'on')
set(gcbo, 'Checked', 'off');
UserData.AutoSaveMat = 'off';
% Change AutoPlot to off
fh = ancestor(hObject,'figure');
```

```
rh = findobj(fh,'Label','Auto Plot');
set(rh, 'Checked', 'off');
UserData.AutoPlot = 'off';
set(rh,'UserData',UserData);
else
set(gcbo, 'Checked', 'on');
UserData.AutoSaveMat = 'on';
end
set(gcbo,'UserData',UserData);
% end of function
function visc = viscmix(gasnum1,pres1,gasnum2,pres2,gasnum3,pres3, ...
gasnum4,pres4,temp,x,y,z)
% viscmix gives gas mixture viscosity given temp. and partial pressures
% Syntax: visc = viscmix(gasnum1,pres1,gasnum2,pres2, ...
% gasnum3,pres3,gasnum4,pres4,temp)
% Input: Inputs are vectors of length nnode from COMSOL GUI
% gasnum1 - gasnum4 = identifying number for each gas
% pres1 - pres4 = partial pressure or mole fraction of gas 1 - 4
% temp = temperature of gas mixture at each node, Kelvin
% x = COMSOL x dimension of each node
% y = COMSOL y dimension of each node; set to zero if 1-D
% z = COMSOL z dimension of each node; set to zero if 2-D
% Local: ngas = number of gases
% nnode = number of COMSOL nodes for 1-D model
% gasnum(ngas) - local array version
% pres(ngas, nnode) - local array version
% Called
% Routines:
% function prop = gas_prop(gasnum)
% function provides gas properties based on identifying gas number
% Output: viscosity of gas mixture in micropoise (pascal second x 10-7)
% Notes:
% Function is based on Lucas method as compiled in "Properties of Gases
% and Liquids" by Reid et. al., 4th Edition. The method of corresponding
% states (9-5.18 to 9-5.23) with the Lucas rules is used. The Lucas method
% is 9-4.15, with supporting equations 9-4.8, 9-4.14, 9-4.16, and 9-4.17.
%
% Partial pressures or mole fractions may be input with equivalent results.
%
% MATLAB version by J. Knox 10-27-05
%
% modification by J. Knox 01-02-08 - change from structure array input to
% variable number of inputs, also input arrays for each node for use in
% COMSOL
%
if nargin>12 || nargin<8
error(['Gas number, pressures or mole fractions, temperature,', ...
' are required inputs for viscosity function', ...
' with maximum of 4 gases and minimum of 2 gases allowed']);
end
% Get properties and perform calcs for each gas
%% Set up local arrays from inputs and preallocate arrays
gasnum(2) = gasnum2(1);
gasnum(1) = gasnum1(1);
pres(2,:) = pres2;
pres(1,:) = pres1;
switch nargin
case 12
gasnum(4) = gasnum4(1);
gasnum(3) = gasnum3(1);
pres(4,:) = pres4;
pres(3,:) = pres3;
ngas = 4;
case 10
gasnum(3) = gasnum3(1);
```

```
pres(3,:) = pres3;
temp = gasnum4;
x = pres4;
y = temp;
z = x;
ngas = 3;
case 8
temp = gasnum3;
x = pres3;
y = gasnum4;
z = pres4;
ngas = 2;
end
nnode = length(x);
if length(y) ~= nnode || length(z) ~= nnode
error('x, y, and z must be of the same length');
end
% Preallocate arrays
gas_p(ngas) = struct('No',0,'Formula','a','Name','a','Molwt',0,'Tfp',0, ...
'Tb',0,'Tc',0,'Pc',0,'Vc',0,'Zc',0,'Omega',0, ...
'Dipm',0,'CPVAPA',0,'CPVAPB',0,'CPVAPC',0, ...
'CPVAPD',0,'Sigma',0);
Tr = zeros(ngas,nnode);
Dipmr = zeros(1,ngas);
Fp = zeros(1,ngas);
Q = zeros(1,ngas);
Fq = zeros(1,ngas);
Fqm = zeros(1,nnode);
Trm = zeros(1,nnode);
y_loc = zeros(ngas,nnode);
visc = zeros(1,nnode);
%% Evaluate node-independent properties
for i = 1:ngas
%
% --- Obtain gas properties
%
gas_p(i) = gas_prop(gasnum(i));
%
% Reduced dipole moment (9-4.16)
%
Dipmr(i) = 52.46*gas_p(i).Dipm^2*gas_p(i).Pc/gas_p(i).Tc^2;
%
% Equation 9-4.18 for quantum gases (He, H2, D2)
%
switch gas_p(i).Name
case 'Helium'
Q(i) = 1.38;
case 'Hydrogen'
Q(i) = 0.76;
case 'Deuteruim'
Q(i) = 0.52;
otherwise
Q(i) = 0.0;
end
end
%% Calculate node-dependent properties
for j = 1:nnode;
for i = 1:ngas;
%
% Reduced Temperature
%
Tr(i,j) = temp(j)/gas_p(i).Tc;
% Equation 9-4.17
%
if (Dipmr(i) >= 0) && (Dipmr(i) < 0.022)
```

```
Fp(i,j) = 1;
elseif (Dipmr(i) >= 0.022) && (Dipmr(i) < 0.075)
Fp(i,j) = 1 + 30.55*(0.292-gas_p(i).Zc)^1.72;
elseif Dipmr(i) >= 0.075
Fp(i,j) = 1 + 30.55*(0.292-gas_p(i).Zc)^1.72 * ...
abs(0.96 + 0.1*(Tr(i,j)-0.7));
end
%
Fq(i,j) = 1.22*Q(i)^0.15*(1 + 0.00385*((Tr(i,j) - 12)^2)^(1/2.016)...
*sign(Tr(i,j) - 12));
if Fq(i,j) == 0
Fq(i,j) = 1;
end
%
% mole fraction
%
y_loc(i,j) = pres(i,j)/sum(pres(:,j));
end
%
% Calculate mixture properties (9-5.18 to 9-5.22)
%
Tcm = sum(y_loc(:,j).*[gas_p.Tc]');
Rbar = 83.14472; % ideal gas constant for bar*cm3/mole/K (units in gas_prop)
Pcm = Rbar*Tcm*sum(y_loc(:,j).*[gas_p.Zc]')./sum(y_loc(:,j).*[gas_p.Vc]');
Mm = sum(y_loc(:,j).*[gas_p.Molwt]');
Fpm = sum(y_loc(:,j).*Fp(:,j));
[MH,iH]=max([gas_p.Molwt]);
ML=min([gas_p.Molwt]);
if MH/ML > 9 && y_loc(iH,j) > 0.05 && y_loc(iH,j) < 0.7
A = 1 - 0.01*(MH/ML)^0.87;
else
A = 1;
end
Fqm(j) = A*sum(y_loc(:,j).*Fq(:,j));
zetam = 0.176*(Tcm/(Mm^3*Pcm^4))^(1/6);
Trm(j) = temp(j)/Tcm;
%
% Calculate mixture viscosity (Equation 9-4.15)
%
visc(j) = 1/zetam*(0.807*Trm(j)^0.618 - 0.357*exp(-0.449*Trm(j)) + ...
0.340*exp(-4.058*Trm(j)) + 0.018)*Fqm(j)*Fpm;
end
function visc = viscosity(gasnum,temp,x,y,z)
% VISCOSITY function provides pure gas viscosity given gas number and temp.
% Syntax: visc = viscosity(gasnum,temp,x,y,z)
%
% Input: gasnum = number of gas of interest at each node (should be
% unchanging in practice)
% temp = temperature at each node, degrees Kelvin
% x = COMSOL x dimension of each node
% y = COMSOL y dimension of each node; set to zero if 1-D
% z = COMSOL z dimension of each node; set to zero if 2-D
%
% Called Routines:
% function prop = gas_prop(gasnum)
% GAS_PROP_FEM provides gas properties based on identifying gas number
% Output: viscosity of gas mixture in micropoise (pascal second x 10-7)
%
% Notes: Function is based on Lucas method as compiled in "Properties
% of Gases and Liquids" by Reid et. al., 4th Edition. Primary
% equation is 9-4.15, with supporting equations 9-4.8, 9-4.14,
% 9-4.16, and 9-4.17.
%
% MATLAB version by J. Knox 10-25-05
% Modification by J. Knox 01-09-08 - add input arrays for each node for use
% in COMSOL
%% Input Check
%
if nargin<5
error('gas number, temperature, x, y, z input required for viscosity');
end
nnode = length(x);
if length(y) ~= nnode || length(z) ~= nnode
error('x, y, and z must be of the same length');
end
%% Obtain gas properties
visc = zeros(1,nnode);
for i = 1:nnode;
gas_p = gas_prop(gasnum(i));
%% Reduced Temperature
Tr = temp(i)/gas_p.Tc;
%% Reduced dipole moment (9-4.16)
Dipmr = 52.46*gas_p.Dipm^2*gas_p.Pc/gas_p.Tc^2;
%% Equation 9-4.17
if (Dipmr >= 0) && (Dipmr < 0.022)
Fp = 1;
elseif (Dipmr >= 0.022) && (Dipmr < 0.075)
Fp = 1 + 30.55*(0.292-gas_p.Zc)^1.72;
elseif Dipmr >= 0.075
Fp = 1 + 30.55*(0.292-gas_p.Zc)^1.72 * abs(0.96 + 0.1*(Tr-0.7));
else
Fp = 1;
end
%% Equation 9-4.18 for quantum gases (He, H2, D2)
switch gas_p.Name
case 'Helium'
Q = 1.38;
case 'Hydrogen'
Q = 0.76;
case 'Deuteruim'
Q = 0.52;
otherwise
Q = 0.0;
end
Fq = 1.22*Q^0.15*(1 + 0.00385*((Tr - 12)^2)*(Tr-12)/abs(Tr-12));
if Fq == 0
Fq = 1;
end
%% Equation 9-4.14
zeta = 0.176*(gas_p.Tc/(gas_p.Molwt^3*gas_p.Pc^4))^(1/6);
%% Viscosity (Equation 9-4.15)
visc(i) = 1/zeta*(0.807*Tr^0.618 - 0.357*exp(-0.449*Tr) + ...
0.340*exp(-4.058*Tr) + 0.018)*Fq*Fp;
end
function [GenIn] = xl2GenIn(xlFileName)
% If interactive, xlFileName is blank; check for mode.
if exist('xlFileName','var')
[FileName] = xlFileName;
else
% Execute command to open user selected Excel file and convert to "GenIn"
% structure array
[FileName] = uigetfile('*.xlsx','Select the Excel file');
end
% Skip conversion if user clicked cancel or did not include FileName on
% command line
if exist('FileName','var')
GenIn.Title = FileName(1:size(FileName,2)-4);
% Create GenIn.heading and GenIn.string and GenIn.tooltip and GenIn.values
xlData = importdata(FileName);
% Merge resulting textdata and data arrays into textdata cell array
```

```
RecNum = size(xlData.data,2);
for i = 1:RecNum;
MergeInd = find(~isnan(xlData.data(:,i)));
xlData.textdata(MergeInd,i+5)=num2cell(xlData.data(MergeInd,i));
end
% Transfer Excel data (single tab) to GenIn structure array
%MT_RowsInd = strmatch('',xlData.textdata(:,1),'exact');
MT_RowsInd = strcmp('',xlData.textdata(:,1));
MT_RowsInd = [1 find(MT_RowsInd)' (size(xlData.textdata(:,1),1)+1)];
for i = 1:size(MT_RowsInd,2)-1
GenIn.NumInputs(i) = MT_RowsInd(i+1)-(MT_RowsInd(i)+2);
GenIn.heading(i) = xlData.textdata(MT_RowsInd(i)+1,1);
GenIn.string(i,1:GenIn.NumInputs(i))= ...
xlData.textdata(MT_RowsInd(i)+2:MT_RowsInd(i+1)-1,1);
GenIn.varname(i,1:GenIn.NumInputs(i))= ...
xlData.textdata(MT_RowsInd(i)+2:MT_RowsInd(i+1)-1,2);
GenIn.tooltip(i,1:GenIn.NumInputs(i))= ...
xlData.textdata(MT_RowsInd(i)+2:MT_RowsInd(i+1)-1,3);
GenIn.unit(i,1:GenIn.NumInputs(i))= ...
xlData.textdata(MT_RowsInd(i)+2:MT_RowsInd(i+1)-1,4);
GenIn.export(i,1:GenIn.NumInputs(i))= ...
xlData.textdata(MT_RowsInd(i)+2:MT_RowsInd(i+1)-1,5);
% Vectorized version
GenIn.values(i,1:GenIn.NumInputs(i),1:RecNum)= ...
xlData.textdata(MT_RowsInd(i)+2:MT_RowsInd(i+1)-1,6:RecNum+5);
end
end
function XTOOLSS_AdjPara(hObject,RecNum,ParaFileName)
% Execute routine to modify parametric values updated by X-TOOLSS. This
% routine is only used in command line mode (called by function
% AdsorptionCL. It is not called by function Adsorption (in interactive
% mode).
% hObject is passed figure handle
fh = hObject;
GenIn = getappdata(fh,'AppGenIn');
% import data file
[Para] = importdata(ParaFileName);
% Replace specified parameters with those from ParaFileName
for i = 1:length(Para.data(:,1))
[HeNu,ItNu]=find(strcmp(GenIn.varname,char(Para.textdata(i))));
GenIn.values{HeNu,ItNu,RecNum} = Para.data(i);
ch = findobj(fh,'Tag',char(Para.textdata(i)));
set(ch,'String',Para.data(i));
end
% Save current values to GenIn array
setappdata(fh,'AppGenIn',GenIn);
```

APPENDIX D

VALIDATION OF DIFFUSION AND VISCOSITY CORRELATIONS

```
%% INTRODUCTION
% M-file verifies the accuracy of diffusion property correlations
% developed for use in COMSOL, MATLAB, and SIMULINK computer models of
% atmosphere revitalization systems.
%
% Initial version created by J. Knox on 02/27/2010

% Include first three gas_prop values for plot text. Old gas values below.
    %         {  56,   59,   77,   99,    80,   76,  4,   501,  450}
    %         { 455,  460,  440,   31,   447,  438,  1,   501,  450}

gas_prop = struct(...
    'No',      { 455,  460,  440,   31,   447,  438,  1,   501,  450},...
    'Formula',{'N2', 'O2', 'H2O', 'CO2','H3N','H2', 'Ar','Air','He'},...
    'Name',{'Nitrogen','Oxygen','Water','Carbon dioxide','Ammonia',...
    'Hydrogen','Argon','Air','Helium'});

%% BINARY GAS DIFFUSION

% This section will test the pure gas diffusion function in "diffbin.m",
% which applies to low pressure gases.

% Function is based on Fuller method as presented in "Properties of Gases
% and Liquids" by Polling et. al., 5th Edition.

% Experimental diffusion data is found in "Transport Phenomena" by Bird et.
% al. and in table 1-2 of "Properties of Gases and Liquids" by Poling et.
% al., 5th Edition

% In order to closest emulate usage in COMSOL, the diffusion function
% is called with vector arrays just as if values at each node were being
% sent.

% Start and end indices for each gas in vector arrays:

starti = [1 5 7 9  11 12 13];
endi   = [4 6 8 10 11 12 13];

% Temperature Data Vector

N2_CO2_temp  = [273.2,288.2,298,298.2];
Air_CO2_temp = [276,317];
N2_H2O_temp  = [308,352];
He_CO2_temp  = [298,498];
CO2_H2O_temp = (307);
He_H2O_temp = (352);
Air_H2O_temp = (313);

temp = [N2_CO2_temp Air_CO2_temp N2_H2O_temp He_CO2_temp CO2_H2O_temp ...
        He_H2O_temp Air_H2O_temp];

% Diffusion Data Vector

N2_CO2 = [0.144,0.158,0.169,0.165];
Air_CO2 = [0.144,0.179];
N2_H2O = [0.259,0.364];
He_CO2 = [0.62,1.433];
CO2_H2O = (0.201);
He_H2O = (1.136);
Air_H2O = (0.292);
diff_exp = [N2_CO2 Air_CO2 N2_H2O He_CO2 CO2_H2O He_H2O Air_H2O];

% Pressure Data Vector: for diffbin, total pressure only important
```

```
pres1 = [1.0132,1.0132,1,1.0132, ones(1,9)];
pres2 = zeros(1,13);

% Gas Number Data Vector for diffbin call and for table headers

    %         {  56,   59,   77,   99,    80,   76,  4,   501,  450}
    %         { 455,  460,  440,   31,   447,  438,  1,   501,  450}

gasnum1 = [455*ones(1,4), 501, 501, 455, 455, 450, 450, 31, 450, 501];
gasnum2 = [31*ones(1,4), 31, 31, 440, 440, 31, 31, 440, 440, 440];

% Gas Number Data Vector for plotting names

gasname1 = [1, 8, 1, 9, 4, 9, 8];
gasname2 = [4, 4, 3, 4, 3, 3, 3];

% Setup unchanging plot labels

x1Label = 'Temperature, °K';
y1Label = 'Diffusion, cm^2/sec';
y2Label = 'Percent Error';
Data1Label = 'Experimental';
Data2Label = 'Correlation';
Data3Label = '% Error';

%  Setup dummy x,y,z variables

x = (0.1:0.1:1.3);
y = (1.1:0.1:2.3);
z = (2.1:0.1:3.3);

% Obtain diffusion based on correlation, cm^2/sec (input temperature in °K
% and pressure in bar; calculate percent error)

% Verification of gases; all are computed in one call as in COMSOL

diff_cor = diffbin(gasnum1,pres1,gasnum2,pres2,temp,x,y,z);
per_err = 100.*(diff_cor-diff_exp)./diff_exp;

% Plot data comparison and percent error for two or more points

for i = 1:4
    xData1 = temp(starti(i):endi(i));
    yData1 = diff_exp(starti(i):endi(i));
    yData2 = diff_cor(starti(i):endi(i));
    yData3 = per_err(starti(i):endi(i));
    x2Label = ['Binary Diffusion of ',...
                gas_prop(gasname1(i)).Name,' and ',gas_prop(gasname2(i)).Name];

DoubleXYPlot(xData1, yData1, yData2, yData3, x1Label, y1Label, x2Label, ...
              y2Label, Data1Label, Data2Label,Data3Label)

end

% Print comparison table

display(sprintf('%s\n','   Gas 1    Gas 2    Temp     Exper    Correl  % Error
'));
for i= 1:13
    display(sprintf('%8.0f %8.0f %8.2f %8.3f %8.3f %8.2f\n', ...

gasnum1(i),gasnum2(i),temp(i),diff_exp(i),diff_cor(i),per_err(i)));
end
```

Table D.1 Experimental and correlated diffusion values for binary gas diffusion.

Gas 1	Gas 2	Temp	Exper	Correl	% Error
455	31	273.20	0.144	0.139	-3.36
455	31	288.20	0.158	0.153	-3.29
455	31	298.00	0.169	0.164	-2.87
455	31	298.20	0.165	0.162	-1.69
501	31	276.00	0.144	0.139	-3.14
501	31	317.00	0.179	0.178	-0.71
455	440	308.00	0.259	0.276	6.71
455	440	352.00	0.364	0.349	-4.09
450	31	298.00	0.620	0.587	-5.30
450	31	498.00	1.433	1.442	0.64
31	440	307.00	0.201	0.222	10.54
450	440	352.00	1.136	1.140	0.33
501	440	313.00	0.292	0.276	-5.33

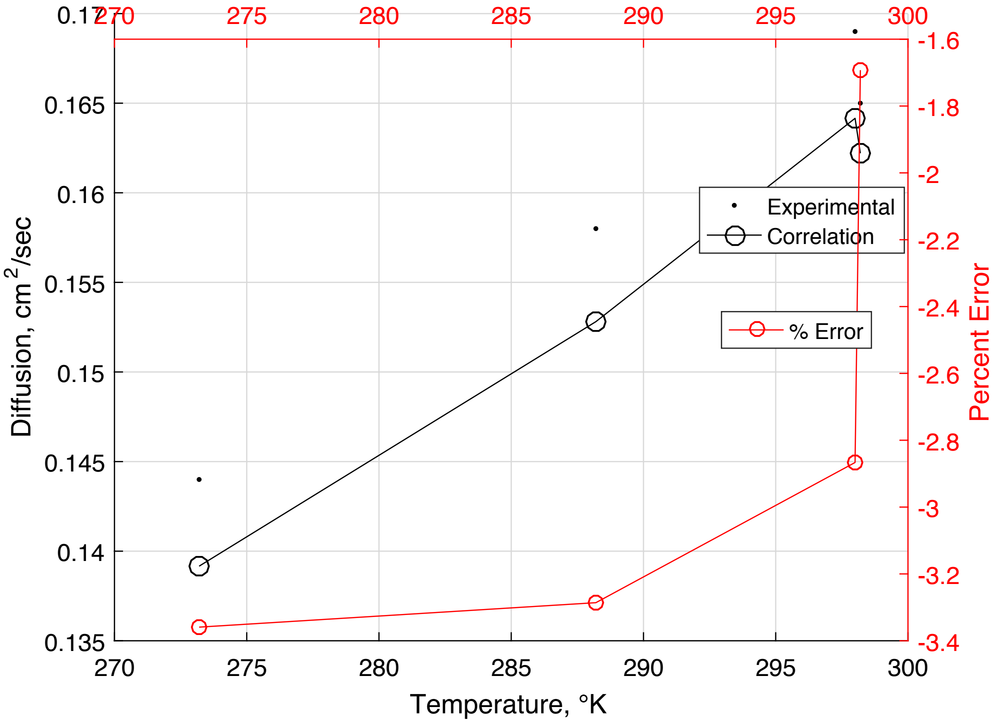


Figure D.1 Binary diffusion of nitrogen and carbon dioxide.

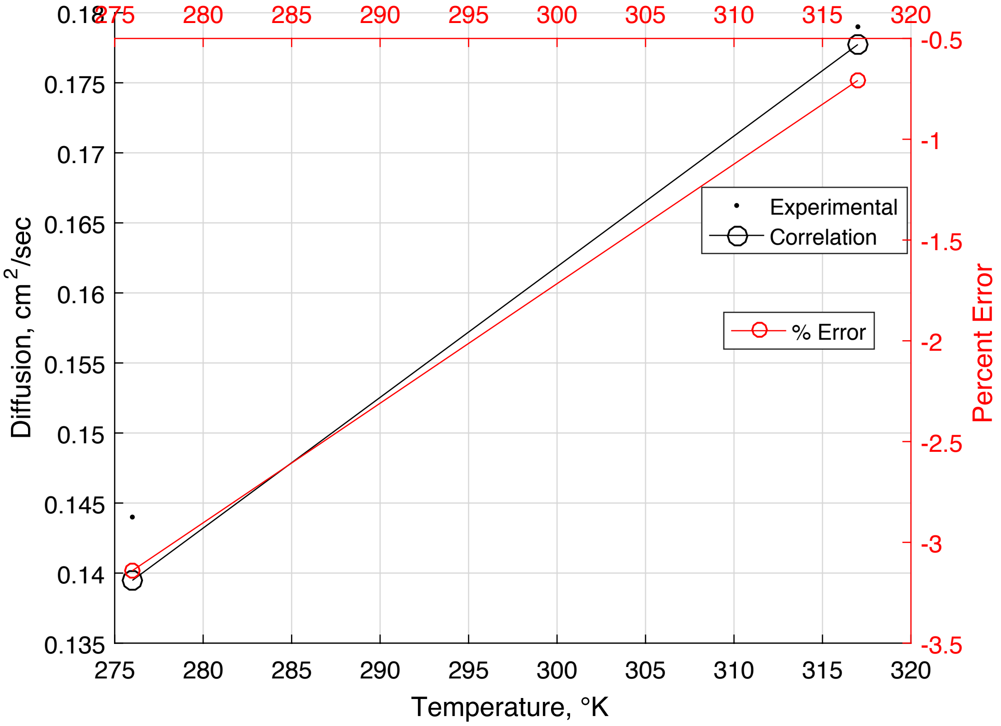


Figure D.2 Binary diffusion of air and carbon dioxide.

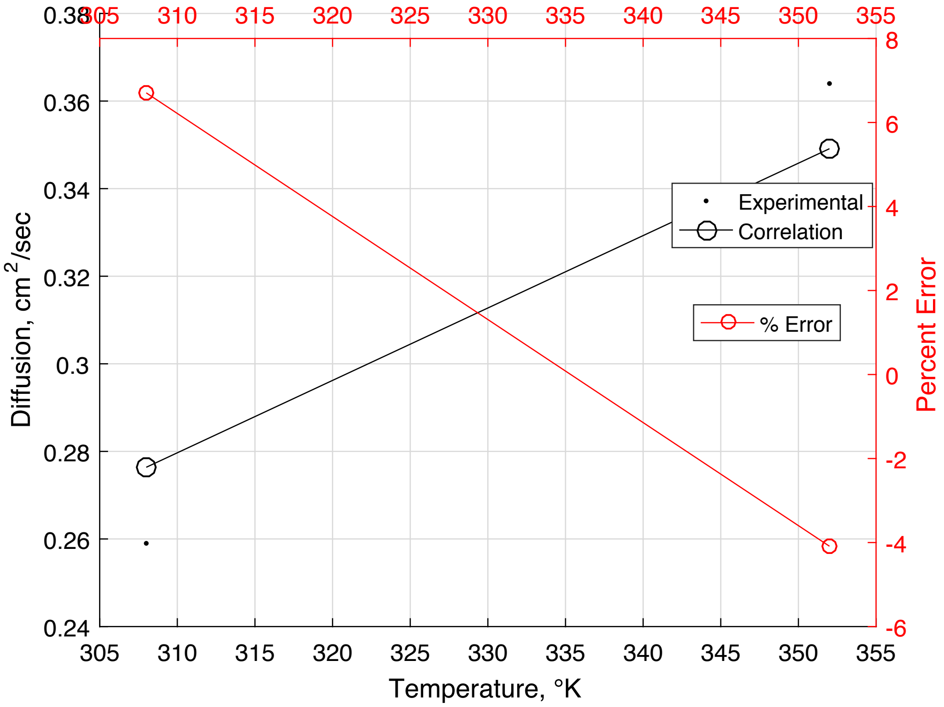


Figure D.3 Binary diffusion of nitrogen and water vapor.

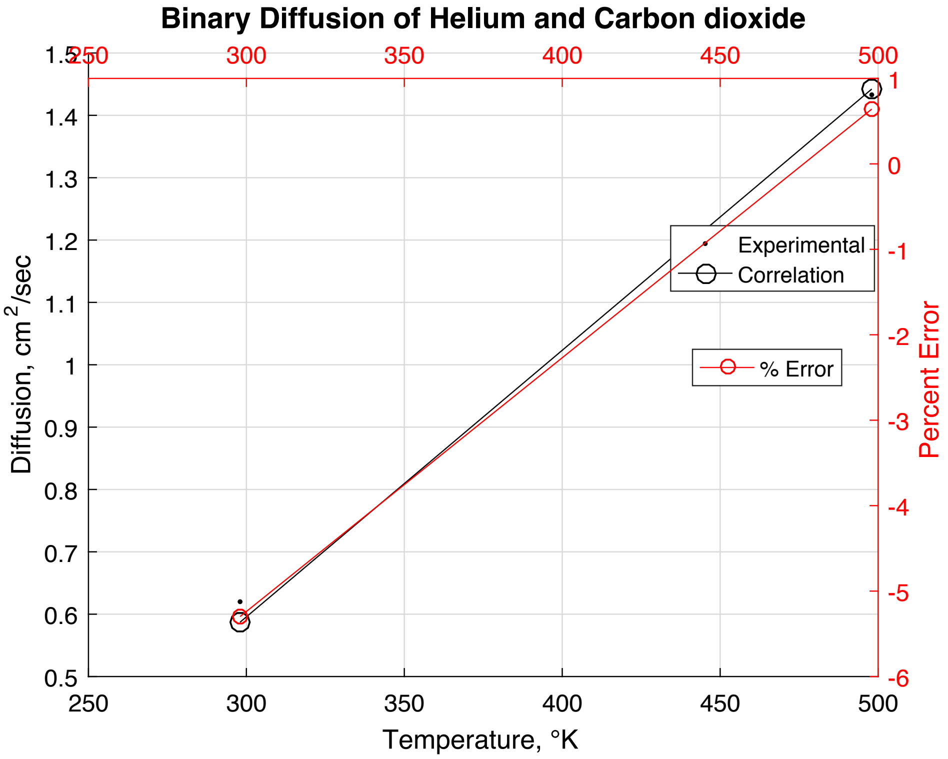


Figure D.4 Binary diffusion of helium and carbon dioxide.

```
%% VERIFICATION FOR THREE GAS MIXTURES

% This section will test the mixed gas viscosity function in
% "diffmix.m", which applies to low pressure gases.

% Function is based on Blanc's law as presented in "Properties of Gases
% and Liquids" by Polling et. al., 5th Edition. Function "diffbin" is used
% to find binary diffusion. Applicability is limited to a dilute gas
% diffusing in a homogeneous mixture.
%
% Use binary diffusion (diffbin) if possible for better accuracy. Use air
% (No. 501) with diffbin instead of oxygen and nitrogen with diffmix for
% much better accuracy.

% Air-gas mixtures of interest may be found in the "Properties of
% Gases and Liquids" 5th edition in table 11-2.

% In order to closest emulate usage in COMSOL, the viscosity function
% is called with vector arrays just as if values at each node were being
% sent.

% Start and end indices for each gas in vector arrays:

starti = [1 3 4];
endi   = [2 3 4];

% Temperature Data Vector

Air_CO2_temp  = [276,317];
Air_H2O_temp = 313;
Air_He_temp  = 276;

temp = [Air_CO2_temp Air_H2O_temp Air_He_temp];

% Diffusion Data Vector

Air_CO2 = [0.144 0.179];
Air_H2O = 0.292;
Air_He = 0.632;
diff_exp = [Air_CO2 Air_H2O Air_He];

% Pressure Data Vector: Note this must include trace gas, here
% assumed to be at 1%, N2, and O2

pres1 = ones(1,4)*0.01;
pres2 = ones(1,4)*0.78;
pres3 = ones(1,4)*0.21;

% Gas Number Data Vector for diffmix call and for table headers

gasnum1 = [31 31 440 450];
gasnum2 = 455*ones(1,4);
gasnum3 = 460*ones(1,4);

% Gas Number Data Vector for plot labels

gasname = [4 4 3 9];

% Setup unchanging plot labels

x1Label = 'Temperature, °K';
y1Label = 'Diffusion, cm^2/sec';
y2Label = 'Percent Error';
```

```
Data1Label = 'Experimental';
Data2Label = 'Correlation';
Data3Label = '% Error';

%  Setup dummy x,y,z variables

x = (0.1:0.1:0.4);
y = (1.1:0.1:1.4);
z = (2.1:0.1:2.4);

% Obtain diffusion based on correlation, cm^2/sec (input temperature in °K
% and pressure in bar; calculate percent error)

% Verification of gases; all are computed in one call as in COMSOL

diff_cor = diffmix(gasnum1,pres1,gasnum2,pres2,gasnum3,pres3,temp,x,y,z);
%diff_cor = diff_cor(:,1)';
per_err = 100.*(diff_cor-diff_exp)./diff_exp;

% Plot data comparison and percent error

for i = 1:1
    xData1 = temp(starti(i):endi(i));
    yData1 = diff_exp(starti(i):endi(i));
    yData2 = diff_cor(starti(i):endi(i));
    yData3 = per_err(starti(i):endi(i));
    x2Label = ['Tertiary Diffusion of Nitrogen, Oxygen, and ',...
                gas_prop(gasname(i)).Name];

DoubleXYPlot(xData1, yData1, yData2, yData3, x1Label, y1Label, x2Label, ...
              y2Label, Data1Label, Data2Label,Data3Label)

end

% Print comparison table

display(sprintf('%s\n','   Trace    Temp     Exper    Correl  % Error '));
for i= 1:4
    display(sprintf('%8.0f %8.2f %8.3f %8.3f %8.2f\n', ...
                    gasnum1(i),temp(i),diff_exp(i),diff_cor(i),per_err(i)));
end
```

Table D.2 Experimental and correlated diffusion values for tertiary gas diffusion.

Trace	Temp	Exper	Correl	% Error
31	276.00	0.144	0.145	0.68
31	317.00	0.179	0.185	3.20
440	313.00	0.292	0.288	-1.27
450	276.00	0.632	0.634	0.25

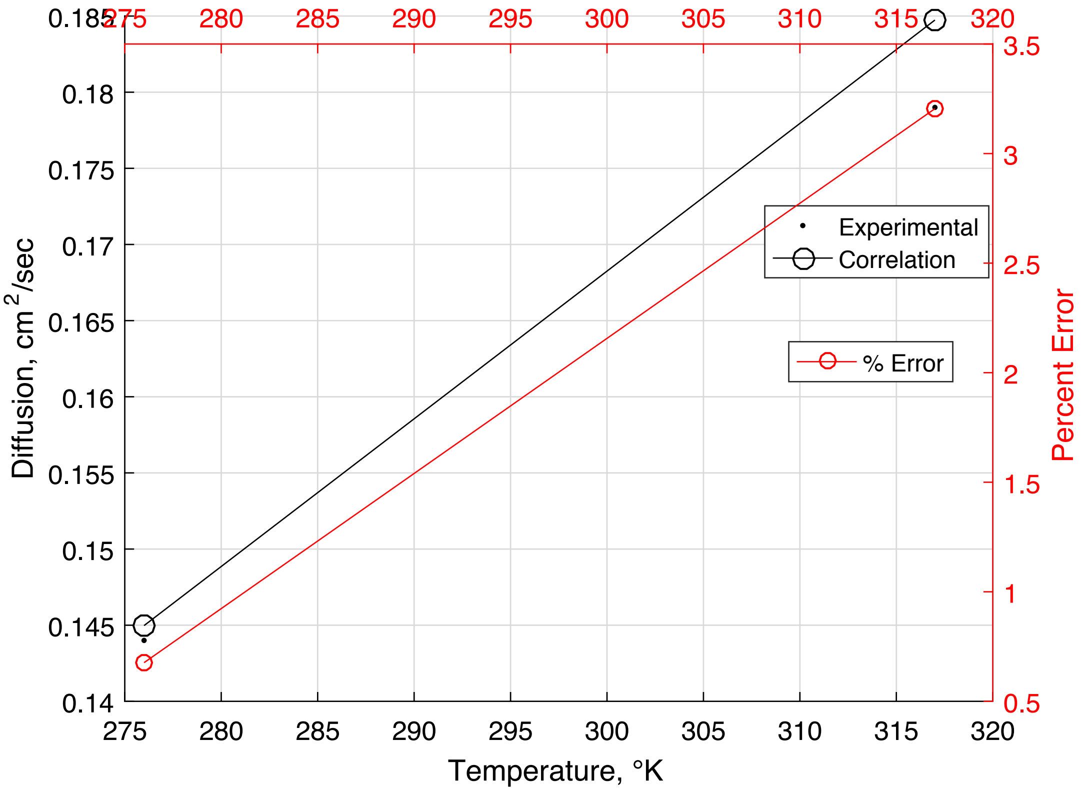


Figure D.5 Tertiary diffusion of nitrogen, oxygen, and carbon dioxide.

```
%% INTRODUCTION
%M-file verifies the accuracy of viscosity property correlations developed for
use in COMSOL, MATLAB, and SIMULINK computer models of atmosphere
%revitalization systems.
%
%Initial version created by J. Knox on 12/01/2008 Version to test COMSOL
compatible functions completed on 01/15/2009

% Include first three gas_prop values for plot text
gas_prop = struct(...
    'No',      { 455,  460,  440,   31,   447,  438,  1},...
    'Formula',{'N2', 'O2', 'H2O', 'CO2','H3N','H2', 'Ar'},...
    'Name',{'Nitrogen','Oxygen','Water','Carbon dioxide','Ammonia',...
    'Hydrogen','Argon'});

%% PURE GAS VISCOSITY
% This section will test the pure gas viscosity function in "viscosity.m",
% which applies to low pressure gases.

% Function is based on Lucas method as compiled in "Properties of Gases and
% Liquids" by Reid et. al., 4th Edition. Primary equation is 9-4.15, with
% supporting equations 9-4.8, 9-4.14, 9-4.16, and 9-4.17.

% Experimental viscosity data for temperatures ranging from 200 to 600 by
% increments of 100, °K and viscosity, ?P are found on page 6-191 of "CRC
% Handbook of Chemistry and Physics, 74th edition. Ammonia data is found in
% table 9-2 of "Properties of Gases and Liquids" by Reid et. al., 4th
% Edition

% In order to closest emulate usage in COMSOL, the viscosity function
% is called with vector arrays just as if values at each node were being
% sent.

% Start and end indices for each gas in vector arrays:
starti = [1 6 11 15 20 23 28];
endi = [5 10 14 19 22 27 32];
% Gas Number Vector
gasnum = [gas_prop(1).No*ones(1,5) gas_prop(2).No*ones(1,5) ...
gas_prop(3).No*ones(1,4) gas_prop(4).No*ones(1,5) ...
gas_prop(5).No*ones(1,3) gas_prop(6).No*ones(1,5) ...
gas_prop(7).No*ones(1,5)];
% Temperature Data Vector
crc_temp = [200,300,400,500,600];
crc_temp3 = [300,400,500,600];
NH3_temp = [37,147,267] + 273.15;
temp = [crc_temp crc_temp crc_temp3 crc_temp NH3_temp crc_temp crc_temp];
% Viscosity Data Vector
crc_exp1 = [129,179,222,261,296];
crc_exp2 = [146,208,261,308,351];
crc_exp3 = [100,133,173,214];
crc_exp4 = [100,150,197,240,280];
NH3_exp5 = [106,146,189];
crc_exp6 = [68,90,109,127,144];
crc_exp7 = [159,229,288,342,390];
visc_exp = [crc_exp1 crc_exp2 crc_exp3 crc_exp4 NH3_exp5 crc_exp6 crc_exp7];
% Setup unchanging plot labels
x1Label = 'Temperature, °K';
y1Label = 'Viscosity, micropoise';
y2Label = 'Percent Error';
Data1Label = 'Experimental';
Data2Label = 'Correlation';
Data3Label = '% Error';
% Setup dummy x,y,z variables
```

```
x = (0.1:0.1:3.2);
y = (1.1:0.1:4.2);
z = (2.1:0.1:5.2);
% Obtain viscosity based on correlation, ?P (input temperature in °K)
% and calculate percent error
% Verification of gases; all are computed in one call as in COMSOL
visc_cor = viscosity(gasnum,temp,x,y,z);
per_err = 100.*(visc_cor-visc_exp)./visc_exp;
% Plot data comparison and percent error
for i = 1:7
xData1 = temp(starti(i):endi(i));
yData1 = visc_exp(starti(i):endi(i));
yData2 = visc_cor(starti(i):endi(i));
yData3 = per_err(starti(i):endi(i));
x2Label = ['Correlated vs. Experimental Pure ',...
gas_prop(i).Name,' Viscosity'];
DoubleXYPlot(xData1, yData1, yData2, yData3, x1Label, y1Label, x2Label, ...
y2Label, Data1Label, Data2Label,Data3Label)
end
% Print comparison table
display(sprintf('%s\n',' GasNum Temp Exper Correl % Error '));
for i= 1:32
display(sprintf('%8.2f %8.2f %8.2f %8.2f %8.2f\n', ...
gasnum(i),temp(i),visc_exp(i),visc_cor(i),per_err(i)));
end
```

Table D.3 Experimental and correlated values for pure gas viscosity.

GasNum	Temp	Exper	Correl	% Error
455.00	200.00	129.00	128.82	-0.14
455.00	300.00	179.00	179.09	0.05
455.00	400.00	222.00	221.97	-0.01
455.00	500.00	261.00	259.83	-0.45
455.00	600.00	296.00	294.06	-0.66
460.00	200.00	146.00	145.05	-0.65
460.00	300.00	208.00	205.27	-1.31
460.00	400.00	261.00	257.15	-1.48
460.00	500.00	308.00	303.12	-1.58
460.00	600.00	351.00	344.74	-1.78
440.00	300.00	100.00	101.97	1.97
440.00	400.00	133.00	135.99	2.25
440.00	500.00	173.00	170.55	-1.42
440.00	600.00	214.00	204.65	-4.37
31.00	200.00	100.00	101.97	1.97
31.00	300.00	150.00	152.22	1.48
31.00	400.00	197.00	198.92	0.97
31.00	500.00	240.00	241.70	0.71
31.00	600.00	280.00	281.08	0.39
447.00	310.15	106.00	108.63	2.48
447.00	420.15	146.00	146.66	0.45
447.00	540.15	189.00	185.73	-1.73
438.00	200.00	68.00	61.36	-9.76
438.00	300.00	90.00	88.85	-1.27
438.00	400.00	109.00	109.93	0.85
438.00	500.00	127.00	130.69	2.91
438.00	600.00	144.00	161.17	11.92
1.00	200.00	159.00	162.66	2.30
1.00	300.00	229.00	229.68	0.30
1.00	400.00	288.00	287.33	-0.23
1.00	500.00	342.00	338.39	-1.06
1.00	600.00	390.00	384.60	-1.38

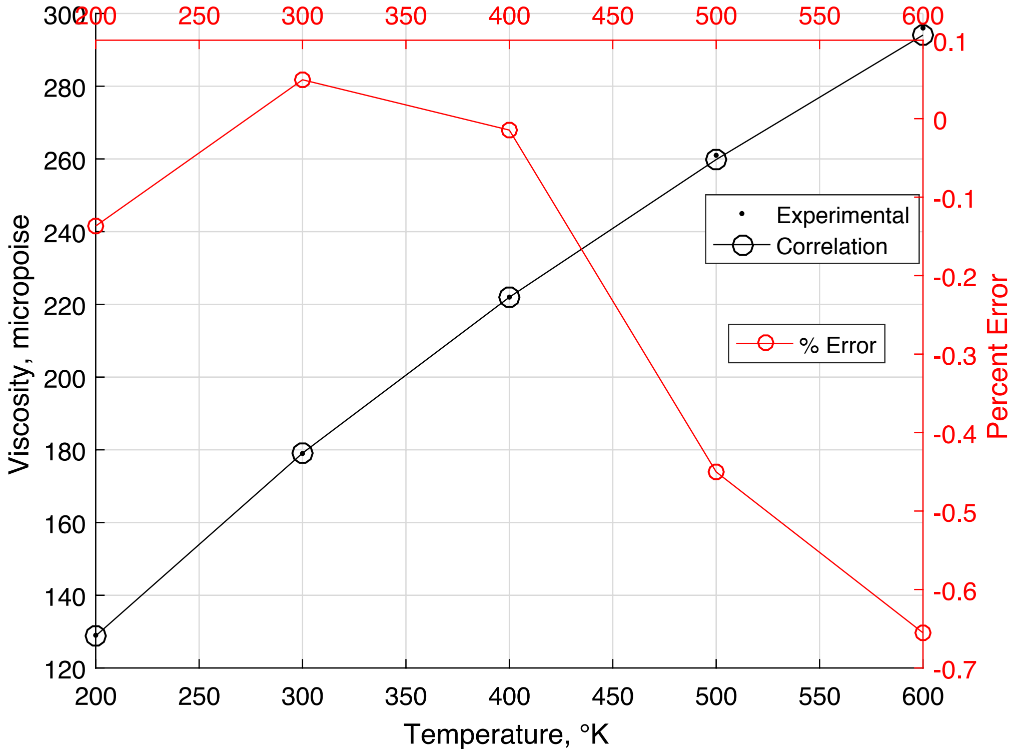


Figure D.6 Correlated versus experimental pure nitrogen viscosity.

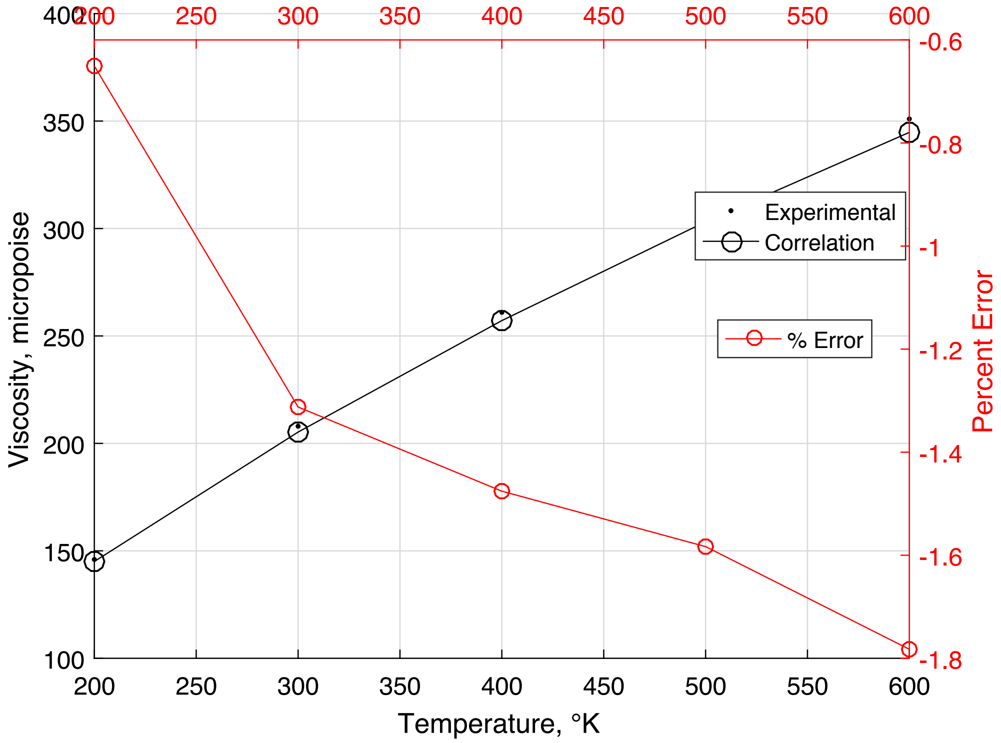


Figure D.7 Correlated versus experimental pure oxygen viscosity.

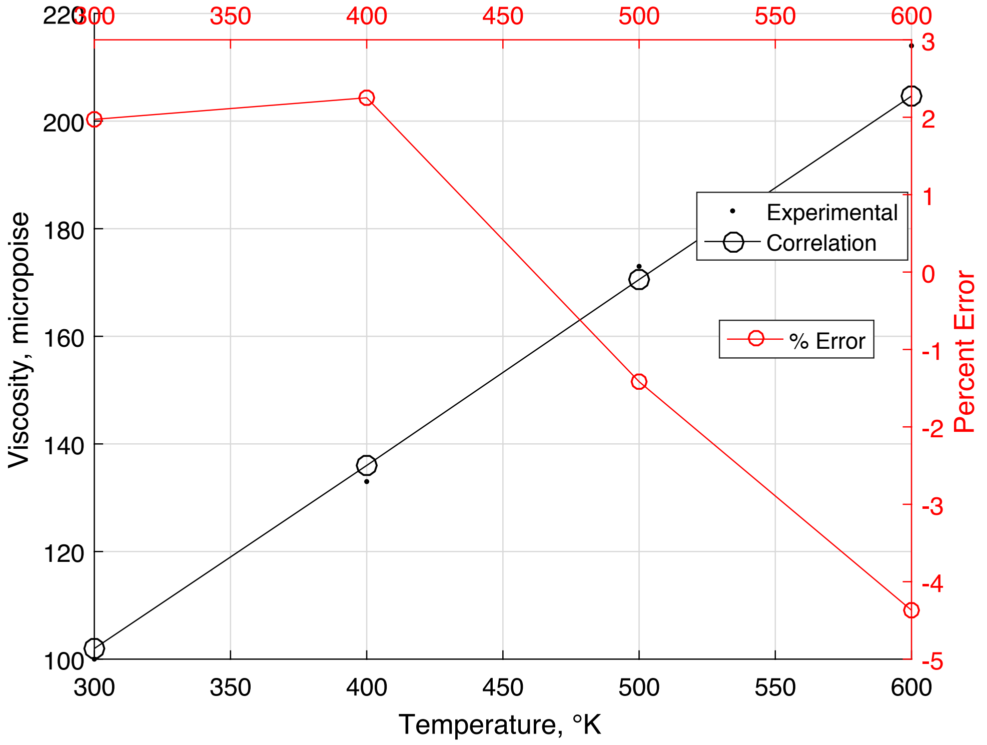


Figure D.8 Correlated versus experimental pure water vapor viscosity.

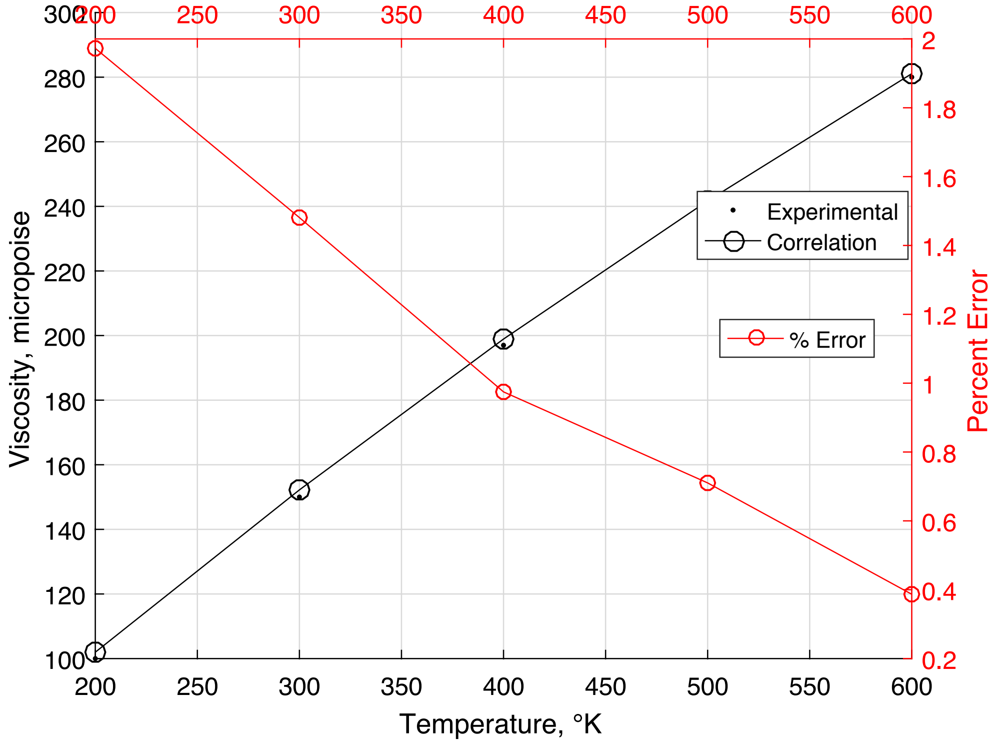


Figure D.9 Correlated versus experimental pure carbon dioxide viscosity.

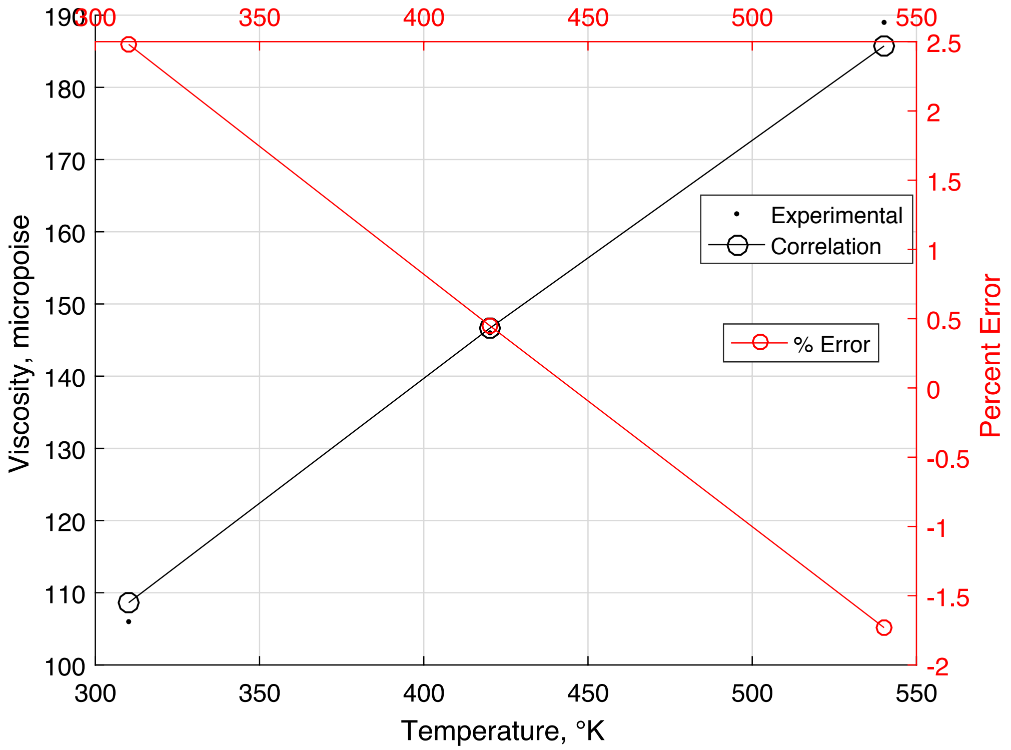


Figure D.10 Correlated versus experimental pure ammonia viscosity.

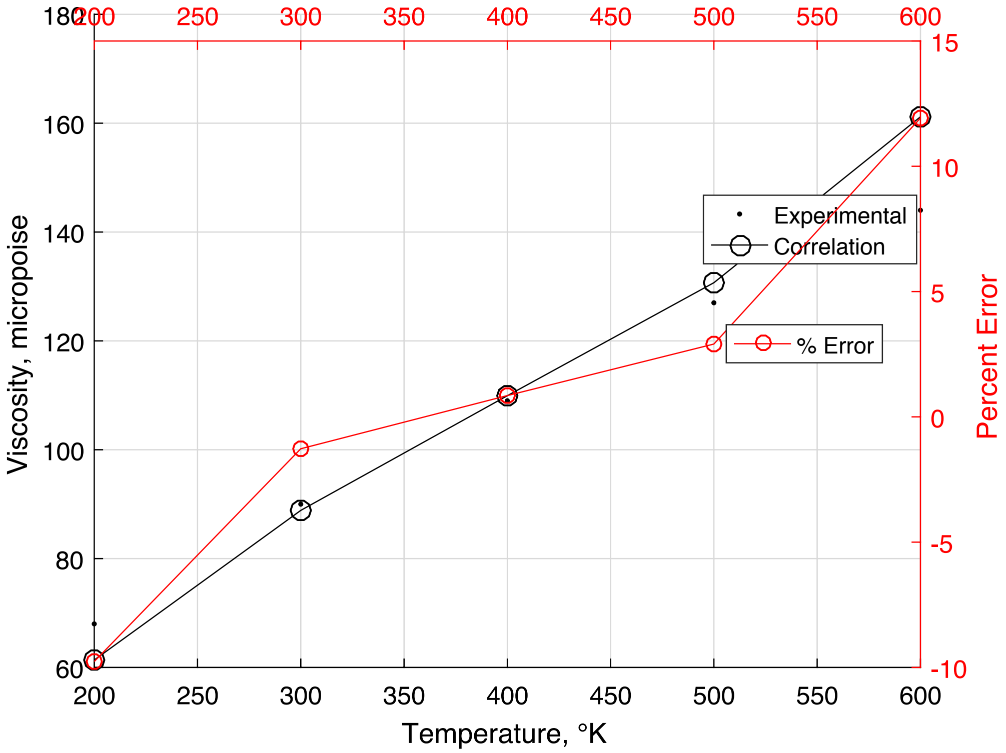


Figure D.11 Correlated versus experimental pure hydrogen viscosity.

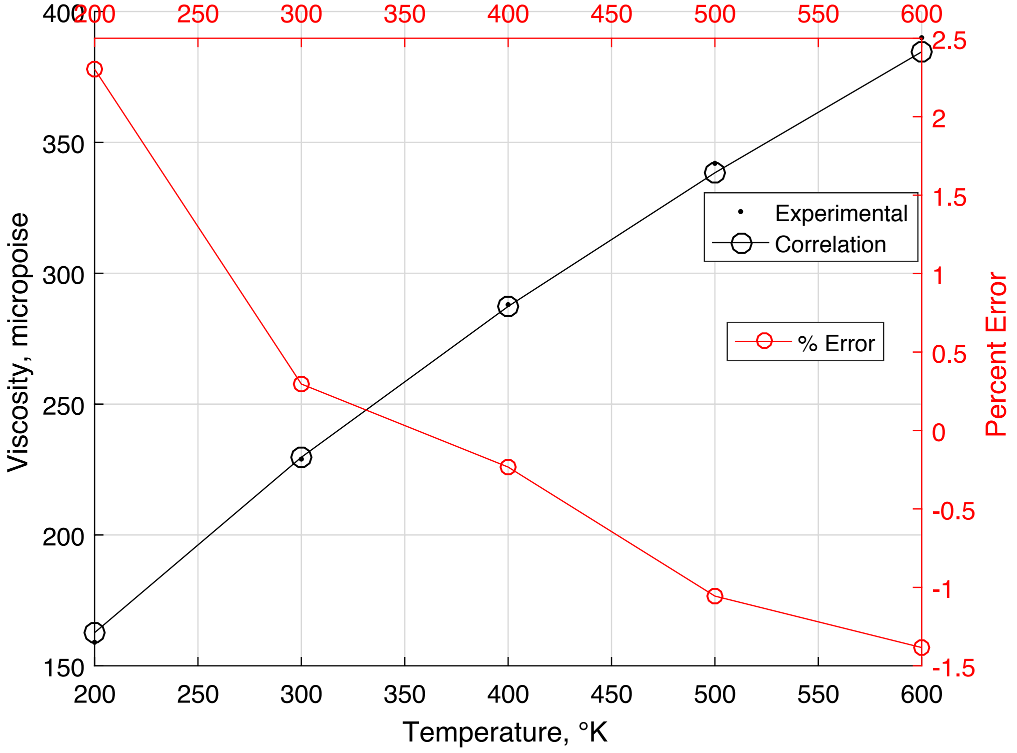


Figure D.12 Correlated versus experimental pure argon viscosity.

```
%%MIXED GAS VISCOSITY
% This section will test the mixed gas viscosity function in
% "viscmix.m", which applies to low pressure gases.
% The mixed gas function is based on Lucas method as compiled in
% "Properties of Gases and Liquids" by Reid et. al., 4th Edition. The
% method of corresponding states (9-5.18 to 9-5.23) with the Lucas rules is
% used. The Lucas method is 9-4.15, with supporting equations 9-4.8,
% 9-4.14, 9-4.16, and 9-4.17.
% Three binary gas mixtures of interest may be found in the "Properties of
% Gases and Liquids" 4th edition in table 9-4. Dry air properties at
% varying altitude is found on page 14-14 of "CRC Handbook of Chemistry and
% Physics, 74th edition. Here N2, O2, CO2, and Ar are included in the
% viscosity calculations.
% In order to closest emulate usage in COMSOL, the viscosity function
% is called with vector arrays just as if values at each node were being
% sent.

%%VERIFICATION FOR BINARY GAS MIXTURES
% Start and end indices for each binary gas data set in vector arrays:
starti = [1 6 11];
endi = [5 10 16];
% Gas Number Vectors and Pointer for plot names
gasnum1 = [gas_prop(1).No*ones(1,10) gas_prop(5).No*ones(1,6)];
gasnum2 = [gas_prop(6).No*ones(1,5) gas_prop(4).No*ones(1,5) ...
gas_prop(6).No*ones(1,6)];
gn1 = [1 1 5];
gn2 = [6 4 6];
% Temperature Data Vector
N2_H2_temp = 373*ones(1,5);
N2_CO2_temp = 293*ones(1,5);
```

```
NH3_H2_temp = 306*ones(1,6);
temp = [N2_H2_temp N2_CO2_temp NH3_H2_temp];
% Mole Fraction of first gas Vector
N2_H2_mol = [0.0,0.2,0.51,0.80,1.0];
N2_CO2_mol = [0.0,0.213,0.495,0.767,1.0];
NH3_H2_mol = [0.0,0.195,0.399,0.536,0.677,1.0];
first_mol = [N2_H2_mol N2_CO2_mol NH3_H2_mol];
% Viscosity Data Vector
N2_H2_exp = [104.2,152.3,190.3,205.8,210.1];
N2_CO2_exp = [146.6,153.5,161.8,172.1,175.8];
NH3_H2_exp = [90.6,118.4,123.8,122.4,120.0,105.9];
visc_exp = [N2_H2_exp N2_CO2_exp NH3_H2_exp];
% Setup unchanging plot labels (others defined in pure viscosity section)
x1Label = 'Mole Fraction First Component';
% Setup dummy x,y,z variables
x = (0.1:0.1:1.6);
y = (1.1:0.1:2.6);
z = (2.1:0.1:3.6);
% Obtain viscosity based on correlation, micropoise (input temperature in °K)
% and calculate percent error
% Verification of gases; each set is computed in one call as in COMSOL
% Plot data comparison and percent error
for i = 1:3
visc_cor(starti(i):endi(i)) = viscmix(gasnum1(starti(i):endi(i)),...
first_mol(starti(i):endi(i)),gasnum2(starti(i):endi(i)),...
1-first_mol(starti(i):endi(i)),temp(starti(i):endi(i)),...
x(starti(i):endi(i)),y(starti(i):endi(i)),z(starti(i):endi(i)));
per_err(starti(i):endi(i)) = 100.*(visc_cor(starti(i):endi(i))-...
visc_exp(starti(i):endi(i)))./visc_exp(starti(i):endi(i));
xData1 = first_mol(starti(i):endi(i));
yData1 = visc_exp(starti(i):endi(i));
yData2 = visc_cor(starti(i):endi(i));
yData3 = per_err(starti(i):endi(i));
x2Label = ['Correlated vs. Experimental ',gas_prop(gn1(i)).Name,...
' and ',gas_prop(gn2(i)).Name,' Viscosity'];
DoubleXYPlot(xData1, yData1, yData2, yData3, x1Label, y1Label, x2Label, ...
y2Label, Data1Label, Data2Label,Data3Label)
end

% Print comparison table
display(sprintf('%s\n',' Gas1 Gas2 Temp Exper Correl % Error '));
for i= 1:16
display(sprintf('%5.0f %4.0f %8.2f %8.2f %8.2f %8.2f\n', ...
gasnum1(i),gasnum2(i),temp(i),visc_exp(i), ...
visc_cor(i),per_err(i)));
end
```

Table D.4 Experimental and correlated values for binary gas viscosities.

Gas1	Gas2	Temp	Exper	Correl	% Error
455	438	373.00	104.20	104.83	0.61
455	438	373.00	152.30	155.36	2.01
455	438	373.00	190.30	186.26	-2.12
455	438	373.00	205.80	213.13	3.56
455	438	373.00	210.10	210.90	0.38
455	31	293.00	146.60	148.78	1.48
455	31	293.00	153.50	153.96	0.30
455	31	293.00	161.80	161.35	-0.28
455	31	293.00	172.10	168.95	-1.83
455	31	293.00	175.80	175.80	-0.00
447	438	306.00	90.60	92.00	1.55
447	438	306.00	118.40	114.36	-3.42
447	438	306.00	123.80	119.42	-3.54
447	438	306.00	122.40	118.55	-3.15
447	438	306.00	120.00	115.91	-3.41
447	438	306.00	105.90	107.15	1.18

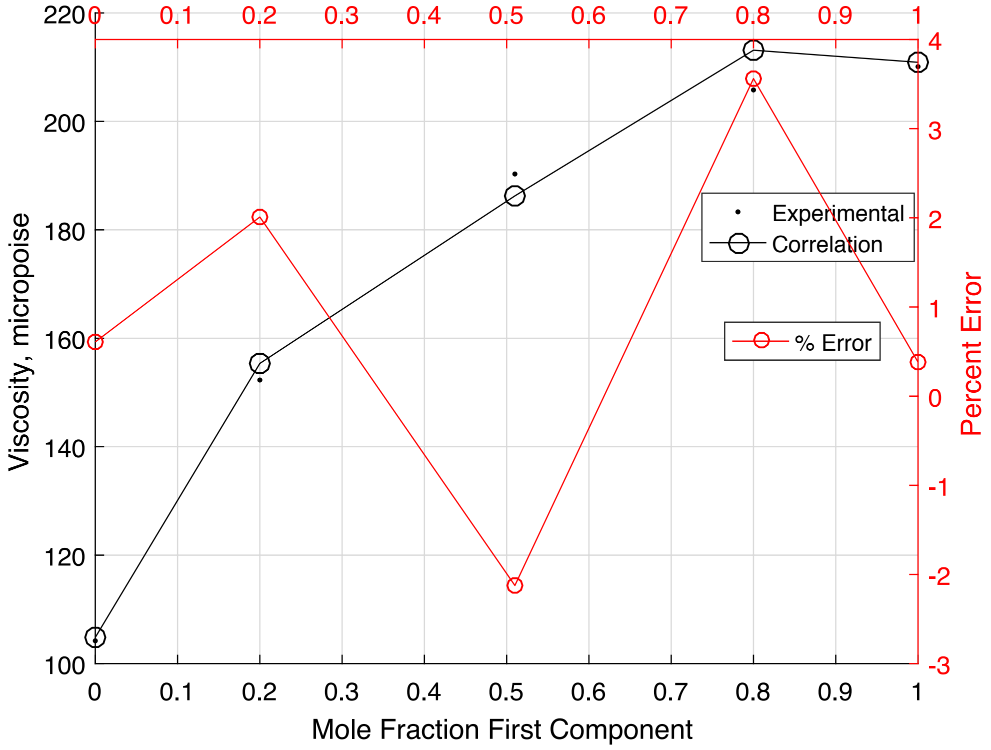


Figure D.13 Correlated versus experimental nitrogen and hydrogen viscosity.

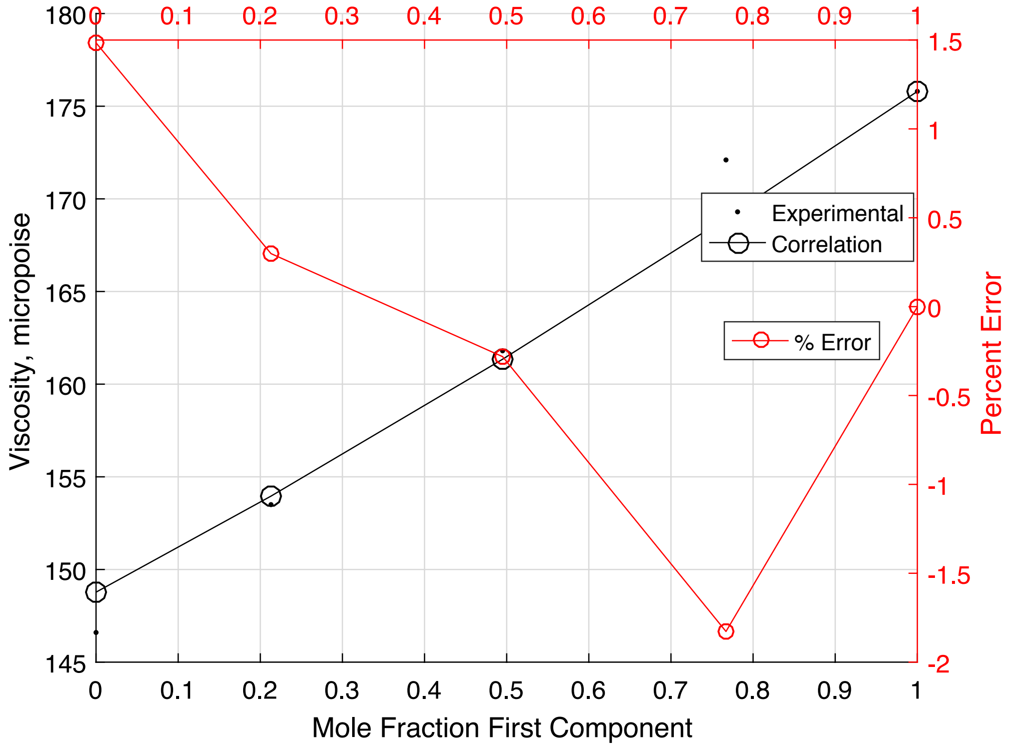


Figure D.14 Correlated versus experimental nitrogen and carbon dioxide viscosity.

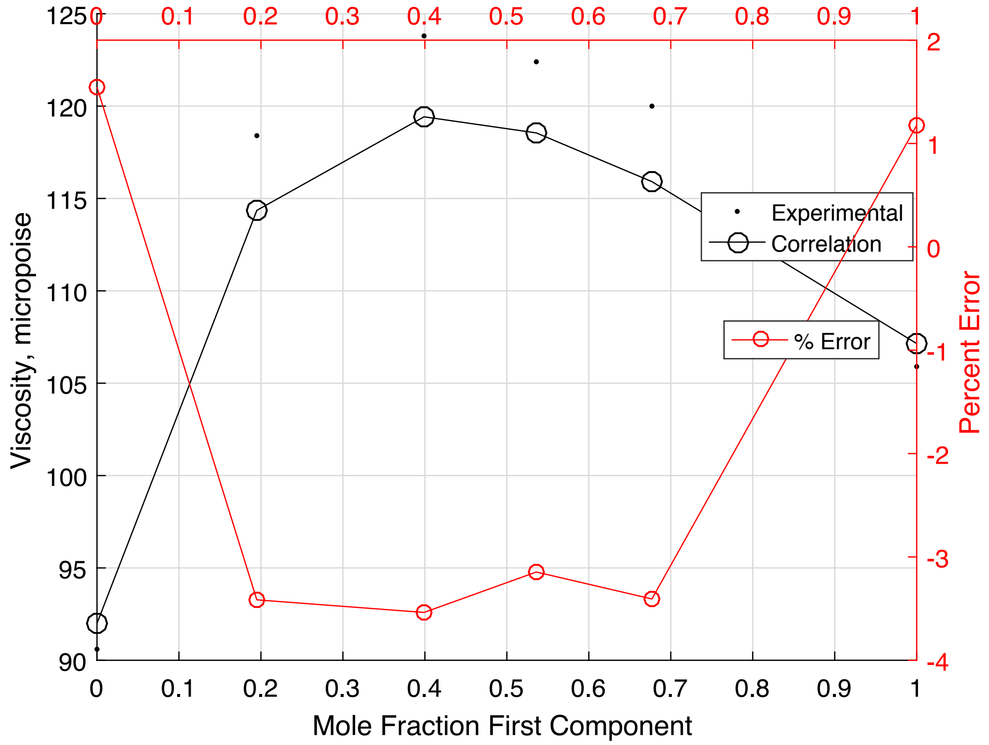


Figure D.15 Correlated versus experimental ammonia and hydrogen viscosity.

```
%%VERIFICATION FOR DRY AIR
% Gas Number Vectors
gasnum1 = gas_prop(1).No*ones(1,10);
gasnum2 = gas_prop(2).No*ones(1,10);
gasnum3 = gas_prop(4).No*ones(1,10);
gasnum4 = gas_prop(7).No*ones(1,10);
% Mole Fractions
pres1 = 0.78084*ones(1,10);
pres2 = 0.209476*ones(1,10);
pres3 = 0.000314*ones(1,10);
pres4 = 0.00934*ones(1,10);
% Temperature, Pressure, and Viscosity Data Vector
temp = [301.15 294.65 288.15 278.40 268.66 258.92 249.19 239.46 229.73 220.01];
pres = [1.2778 1.1393 1.0133 0.8456 0.7012 0.5775 0.4722 0.3830 0.3080 0.2454];
visc_exp = [185.15 182.06 178.94 174.20 169.38 164.48 159.49 154.42 149.26
144.00];
% Setup unchanging plot labels (others defined in pure viscosity section)
x1Label = 'Temperature, °K';
% Setup dummy x,y,z variables
x = (0.1:0.1:1.0);
y = (1.1:0.1:2.0);
z = (2.1:0.1:3.0);
% Obtain viscosity based on correlation, ?P (input temperature in °K)
% and calculate percent error
% Verification of gases; each set is computed in one call as in COMSOL
% Plot data comparison and percent error
visc_cor = viscmix(gasnum1,pres1,gasnum2,pres2,gasnum3,pres3, ...
gasnum4,pres4,temp,x,y,z);
per_err = 100.*(visc_cor-visc_exp)./visc_exp;
xData1 = temp;
```

```
yData1 = visc_exp;
yData2 = visc_cor;
yData3 = per_err;
x2Label = 'Correlated vs. Experimental Air Viscosity';
DoubleXYPlot(xData1, yData1, yData2, yData3, x1Label, y1Label, x2Label, ...
y2Label, Data1Label, Data2Label,Data3Label)
% Print comparison table
display(sprintf('%s\n',' Temp Exper Correl % Error '));
for i= 1:10
display(sprintf('%8.2f %8.2f %8.2f %8.2f\n', ...
temp(i),visc_exp(i), ...
visc_cor(i),per_err(i)));
end
```

Table D.5 Experimental and correlated values for viscosities of dry air.

```
  Temp     Exper    Correl  % Error

  301.15   185.15   184.95    -0.11

  294.65   182.06   181.82    -0.13

  288.15   178.94   178.66    -0.16

  278.40   174.20   173.85    -0.20

  268.66   169.38   168.98    -0.24

  258.92   164.48   164.02    -0.28

  249.19   159.49   158.98    -0.32

  239.46   154.42   153.86    -0.37

  229.73   149.26   148.64    -0.42

  220.01   144.00   143.34    -0.46
```

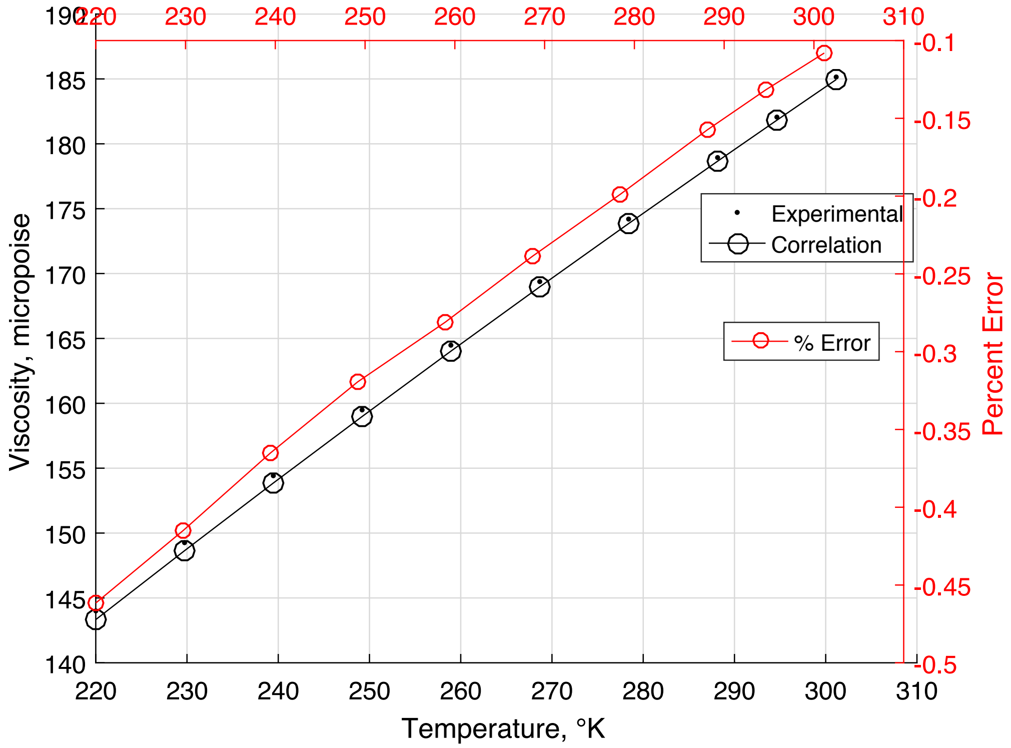


Figure D.16 Correlated versus experimental air viscosity.

APPENDIX E

VALIDATION OF VIRTUAL ADSORPTION TEST SUITE

Verification of Matlab Adsorption Routine

This file performs a verification of the Adsorption calculations for the particular simulation run described in the excel input file.

Input from Excel file is performed in "Input for Adsorption Verification", referenced below

Include << ./Inputs for Adsorption Verification Mixed fits database.mcdx

Matlab Verification Calculations

Gas properties copied from Matlab "gas_prop" routine shown below

```
gas_prop = struct(...
   'No',    { 455,  460,  440,   31,   447,  438,  1,   501, 450},...
   'Formula',{'N2', 'O2', 'H2O', 'CO2','H3N','H2', 'Ar','Air','He'},...
   'Name',{'Nitrogen','Oxygen','Water','Carbon Dioxide','Ammonia',...
   'Hydrogen','Argon','Air','Helium'},...
  'Molwt' ,{  28.013,   31.999,    18.015,    44.010,  17.031,  2.016, 39.948,   28.850,  4.003},...
   'Tfp'   ,{ 63.83,     54.4,      273.15,     216.6,      0,       0,      83.8,       0,       2.15},...
   'Tb'     ,{ 77.4,      90.2,       373.3,        0,         0,       0,      87.3,       0,       4.30},...
   'Tc'     ,{126.2,     154.6,      647.3,      304.1,    405.5,   33.2,   150.8,     132.6,    5.19},...
   'Pc'     ,{ 33.9,      50.4,       221.2,       73.8,     113.5,   13.0, 48.7,      37.37,   2.27},...
   'Vc'     ,{ 89.8,      73.4,        57.1,        93.9,     72.5,   65.1,   74.9,        0,      57.3},...
   'Zc'     ,{  0.290,    0.288,      0.235,      0.274,   0.244,  0.306,  0.291,     0,       0.301},...
   'Omega' ,{  0.039,    0.025,      0.344,      0.239,   0,        0,       0.001,    0,     -3.90},...
   'Dipm'  ,{  0.0,       0.0,         1.8,         0.0,      1.47,   0,       0,          0,      0},...
   'CPVAPA',{ 3.115e+1,  2.811e+1,  3.194e+1,  1.980e+1, 0,      0,      2.080e+1, 0,      0},...
   'CPVAPB',{-1.357e-2, -3.680e-6,  1.436e-3,  7.344e-2, 0,      0,       0,         0,      0},...
   'CPVAPC',{ 2.680e-5,  1.746e-5,  2.432e-5, -5.602e-5, 0,      0,       0,         0,      0},...
   'CPVAPD',{-1.168e-8, -1.065e-8, -1.176e-8,  1.715e-8, 0,      0,       0,         0,      0},...
   'Sigma' ,{18.5,       16.3,       13.1,       26.9,     20.7,   6.12,  16.2,     19.7     2.67});
```

$$gas\_prop := \begin{bmatrix} 28.013 & 31.999 & 18.015 & 44.01 & 17.031 & 2.016 & 39.948 & 28.85 & 4.003 \\ 63.83 & 54.4 & 273.15 & 216.6 & 0 & 0 & 83.8 & 0 & 2.15 \\ 77.4 & 90.2 & 373.3 & 0 & 0 & 0 & 87.3 & 0 & 4.3 \\ 126.2 & 154.6 & 647.3 & 304.1 & 405.5 & 33.2 & 150.8 & 132.6 & 5.19 \\ 33.9 & 50.4 & 221.2 & 73.8 & 113.5 & 13 & 48.7 & 37.37 & 2.27 \\ 89.8 & 73.4 & 57.1 & 93.9 & 72.5 & 65.1 & 74.9 & 0 & 57.3 \\ 0.29 & 0.288 & 0.235 & 0.274 & 0.244 & 0.306 & 0.291 & 0 & 0.301 \\ 0.039 & 0.025 & 0.344 & 0.239 & 0 & 0 & 0.001 & 0 & -3.9 \\ 0 & 0 & 1.8 & 0 & 1.47 & 0 & 0 & 0 & 0 \\ 3.115 \cdot 10 & 2.811 \cdot 10 & 3.194 \cdot 10 & 1.98 \cdot 10 & 0 & 0 & 2.08 \cdot 10 & 0 & 0 \\ -1.357 \cdot 10^{-2} & -3.68 \cdot 10^{-6} & 1.436 \cdot 10^{-3} & 7.344 \cdot 10^{-2} & 0 & 0 & 0 & 0 & 0 \\ 2.68 \cdot 10^{-5} & 1.746 \cdot 10^{-5} & 2.432 \cdot 10^{-5} & -5.602 \cdot 10^{-5} & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & \vdots \end{bmatrix}$$

$debyes := 3.162 \cdot 10^{-25} \cdot \sqrt{\mathbf{newton} \cdot \mathbf{m}^4}$

$Rbar := 8.314472 \cdot \mathbf{joule} \cdot \mathbf{mole}^{-1} \cdot \mathbf{K}^{-1}$

$MCO2 := gas\_prop_{1,4} \cdot \mathbf{gm} \cdot \mathbf{mole}^{-1} = 44.01\ \mathbf{gm} \cdot \mathbf{mole}^{-1}$

$MH2O := gas\_prop_{1,3} \cdot \mathbf{gm} \cdot \mathbf{mole}^{-1} = 18.015\ \mathbf{gm} \cdot \mathbf{mole}^{-1}$

$MO2 := gas\_prop_{1,2} \cdot \mathbf{gm} \cdot \mathbf{mole}^{-1} = 31.999\ \mathbf{gm} \cdot \mathbf{mole}^{-1}$

$MN2 := gas\_prop_{1,1} \cdot \mathbf{gm} \cdot \mathbf{mole}^{-1} = 28.013\ \mathbf{gm} \cdot \mathbf{mole}^{-1}$

$MHe := gas\_prop_{1,9} \cdot \mathbf{gm} \cdot \mathbf{mole}^{-1} = 4.003\ \mathbf{gm} \cdot \mathbf{mole}^{-1}$

Inlet Pressures (parse InPress for constituent gases)

$space1 := \mathrm{search}\left(InPress, \text{“ ”}, 1\right)$ $space1 = 5$

$space2 := \mathrm{search}\left(InPress, \text{“ ”}, space1 + 1\right)$ $space2 = 7$

$space3 := \mathrm{search}\left(InPress, \text{“ ”}, space2 + 1\right)$ $space3 = 9$

$InCO2 := \mathrm{str2num}\left(\mathrm{substr}\left(InPress, 0, space1\right)\right) \cdot \mathbf{kPa}$ $InCO2 = 0.819\ \mathbf{kPa}$

$InH2O := \mathrm{str2num}\left(\mathrm{substr}\left(InPress, space1 + 1, space2 - space1 - 1\right)\right) \cdot \mathbf{kPa}$ $InH2O = 0\ \mathbf{kPa}$

$InO2 := \mathrm{str2num}\left(\mathrm{substr}\left(InPress, space2 + 1, space3 - space2 - 1\right)\right) \cdot \mathbf{kPa}$ $InO2 = 0\ \mathbf{kPa}$

$InN2 := \mathrm{str2num}\left(\mathrm{substr}\left(InPress, space3 + 1, \mathrm{strlen}\left(InPress\right) - space3\right)\right) \cdot \mathbf{kPa}$ $InN2 = 105.82\ \mathbf{kPa}$

Add air pressure to oxygen and nitrogen

$InN2 := InN2 + 0.788 \cdot InAir = 105.82\ \mathbf{kPa}$

$InO2 := InO2 + 0.212 \cdot InAir = 0\ \mathbf{kPa}$

Total Pressure

$TotPress := InCO2 + InH2O + InO2 + InN2 + InAir + InHe = 106.639\ \mathbf{kPa}$

$TotPressML = 106.639\ \mathbf{kPa}$

$$TotPressErr := \frac{TotPress - TotPressML}{TotPress} = 0\ 1\%$$

Mixture Molecular Weight

$$MolWt := \frac{(InCO2 \cdot MCO2 + InH2O \cdot MH2O + InO2 \cdot MO2 + InN2 \cdot MN2 + InHe \cdot MHe)}{TotPress} = 28.136\ \frac{\mathbf{gm}}{\mathbf{mole}}$$

$$MolWtML = 28.136\ \frac{\mathbf{gm}}{\mathbf{mole}}$$

$$MolWtErr := \frac{MolWt - MolWtML}{MolWt} = 0\ \ 1\%$$

Inlet Gas Density

$$GasDens := \frac{TotPress \cdot MolWt}{Rbar \cdot InTemp} = 1.212\ \frac{\mathbf{kg}}{\mathbf{m}^3}$$

$$GasDensML = 1.212\ \frac{\mathbf{kg}}{\mathbf{m}^3}$$

$$GasDensErr := \frac{GasDens - GasDensML}{GasDens} = 0\ \ 1\%$$

Inlet Concentration

$$InPP := \left\| \begin{array}{l} \text{if } Sorbate = \text{“CO2”} \\ \quad \| \; InCO2 \\ \text{if } Sorbate = \text{“H2O”} \\ \quad \| \; InH2O \\ \text{if } Sorbate = \text{“O2”} \\ \quad \| \; InO2 \\ \text{if } Sorbate = \text{“N2”} \\ \quad \| \; InN2 \end{array} \right. = 0.819\ \mathbf{kPa}$$

$$Conc := \frac{InPP}{Rbar \cdot InTemp} = 0.331\ \frac{\mathbf{mol}}{\mathbf{m}^3}$$

$$ConcML = 0.331\ \frac{\mathbf{mol}}{\mathbf{m}^3}$$

$$ConcErr := \frac{Conc - ConcML}{Conc} = 0\ \ 1\%$$

Note for thermal characterization (zero concentation) a divide by zero error occurs for this error check.

Superficial Velocity

$$SupVel := \frac{FlowRate \cdot InTemp \cdot StandPress}{FreeFlowArea \cdot StandTemp \cdot TotPress} = 0.274\ \frac{\mathbf{m}}{\mathbf{s}}$$

$$SupVelML = 0.274\ \frac{\mathbf{m}}{\mathbf{s}}$$

$$SupVelErr := \frac{SupVel - SupVelML}{SupVel} = -2.027 \cdot 10^{-14}\ \ 1\%$$

Void Fraction, Packed Density, and Sorbent Mass

SorbentMass is used to calculate PackedDensity and VoidFractionChk if VoidFraction is zero; otherwise VoidFraction is used to calculate PackedDensity and SorbentMassChk. Note: SorbentMass is overwritten if VoidFraction is nonzero

$$PackedDensity := \left\| \begin{array}{l} \text{if } VoidFraction = 0 \\ \quad \left\| \dfrac{SorbentMass}{FreeFlowArea \cdot BedLength} \right. \\ \text{else} \\ \quad \left\| \left(1 - VoidFraction\right) \cdot PartDensity \right. \end{array} \right| = 767 \, \frac{\boldsymbol{kg}}{\boldsymbol{m}^3}$$

$$SorbentMass := \left\| \begin{array}{l} \text{if } VoidFraction \neq 0 \\ \quad \left\| PackedDensity \cdot FreeFlowArea \cdot BedLength \right. \\ \text{else} \\ \quad \left\| SorbentMass \right. \end{array} \right| = 0.347 \, \boldsymbol{kg}$$

$$VoidFractionChk := \left\| \begin{array}{l} \text{if } VoidFraction = 0 \\ \quad \left\| 1 - \dfrac{PackedDensity}{PartDensity} \right. \\ \text{else} \\ \quad \left\| 0 \right. \end{array} \right| = 0$$

$$VoidFraction := \left\| \begin{array}{l} \text{if } VoidFraction = 0 \\ \quad \left\| 1 - \dfrac{PackedDensity}{PartDensity} \right. \\ \text{else} \\ \quad \left\| VoidFraction \right. \end{array} \right| = 0.35$$

$$VoidFractionErr := \left\| \begin{array}{l} \text{if } VoidFractionChk \neq 0 \\ \quad \left\| \dfrac{VoidFractionChk - VoidFractionChkML}{VoidFractionChk} \right. \\ \text{else} \\ \quad \left\| 0 \right. \end{array} \right| = 0$$

$$PackedDensityErr := \frac{PackedDensity - PackedDensityML}{PackedDensity} = 0 \ 1\%$$

Interstitial Velocity

$$InterVel := \frac{SupVel}{VoidFraction} = 0.783\ \frac{\boldsymbol{m}}{\boldsymbol{s}}$$

$$InterVelErr := \frac{InterVel - InterVelML}{InterVel} = -1.419 \cdot 10^{-14}\ 1\%$$

Viscosity and Diffusivity Gas Property Data

Set up arrays for inputs to viscosity and diffusivity routines. Order arrays with sorbate gas (should be lowest partial pressure) first as diffusivity calcuations for more than 3 gases assume dilute gas is first. Helium is neglected as is rarely used.

$$InPPall := \begin{bmatrix} InN2 \\ InO2 \\ InH2O \\ InCO2 \end{bmatrix} = \begin{bmatrix} 1.058 \cdot 10^{5} \\ 0 \\ 0 \\ 819 \end{bmatrix} \boldsymbol{Pa}$$

```
$Sorb :=$  if $Sorbate = \text{"CO2"}$
               $4$
           if $Sorbate = \text{"H2O"}$
               $3$
           if $Sorbate = \text{"O2"}$
               $2$
           if $Sorbate = \text{"N2"}$
               $1$
```

$$ii_1 := Sorb$$

```
$InPP3 :=$  if $Sorb = 4$
                $[InH2O\ \ InO2\ \ InN2]^{T}$
            if $Sorb = 3$
                $[InCO2\ \ InO2\ \ InN2]^{T}$
            if $Sorb = 2$
                $[InCO2\ \ InH2O\ \ InN2]^{T}$
            if $Sorb = 1$
                $[InCO2\ \ InH2O\ \ InO2]^{T}$
```

```
$ii_2 :=$  if $\max(InPP3) = InN2$
               $1$
           if $\max(InPP3) = InO2$
               $2$
           if $\max(InPP3) = InH2O$
               $3$
           if $\max(InPP3) = InCO2$
               $4$
$= 1$
```

$$ii = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$$

```
$IndexLeft :=$  if $(ii_1 + ii_2) = 3$
                    $[3\ \ 4]^{T}$
                if $(ii_1 + ii_2) = 4$
                    $[2\ \ 4]^{T}$
                if $(ii_1 + ii_2) = 6$
                    $[1\ \ 3]^{T}$
                if $(ii_1 + ii_2) = 7$
                    $[1\ \ 2]^{T}$
                if $(ii_1 + ii_2) = 5$
                    if  $ii_1 = 1$
```

$$IndexLeft = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

```
$ii_3 :=$  if $\left(InPPall_{\left(IndexLeft_2\right)}\right) < \left(InPPall_{\left(IndexLeft_1\right)}\right)$
               $IndexLeft_1$
           else
               $IndexLeft_2$
```

```
        $ii_1 = 4$
      $\begin{bmatrix} 2 & 3 \end{bmatrix}^T$
    if  $ii_1 = 2$
        $ii_1 = 3$
      $\begin{bmatrix} 1 & 4 \end{bmatrix}^T$
```

```
$ii_4 :=$ if $ii_3 = IndexLeft_2$
              $IndexLeft_1$
          else
              $IndexLeft_2$
```

$ii = \begin{bmatrix} 4 \\ 1 \\ 3 \\ 2 \end{bmatrix}$ $\quad i := 1..4 \quad$ $InPPall2_i := InPPall_{(ii_i)}$ $\quad$ $InPPall2 = \begin{bmatrix} 819 \\ 1.058 \cdot 10^5 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{Pa}$

$MolWtI_i := gas\_prop_{1, ii_i} \cdot \dfrac{\boldsymbol{gm}}{\boldsymbol{mole}}$ $\quad$ $MolWtI = \begin{bmatrix} 44.01 \\ 28.013 \\ 18.015 \\ 31.999 \end{bmatrix} \dfrac{\boldsymbol{gm}}{\boldsymbol{mole}}$ $\quad$ $Tc_i := gas\_prop_{4, ii_i} \cdot \boldsymbol{K}$ $\quad$ $Tc = \begin{bmatrix} 304.1 \\ 126.2 \\ 647.3 \\ 154.6 \end{bmatrix} \boldsymbol{K}$

$Pc_i := gas\_prop_{5, ii_i} \cdot \boldsymbol{bar}$ $\quad$ $Pc = \begin{bmatrix} 73.8 \\ 33.9 \\ 221.2 \\ 50.4 \end{bmatrix} \boldsymbol{bar}$ $\quad$ $Vc_i := gas\_prop_{6, ii_i} \cdot \dfrac{\boldsymbol{cm}^3}{\boldsymbol{mole}}$ $\quad$ $Vc = \begin{bmatrix} 93.9 \\ 89.8 \\ 57.1 \\ 73.4 \end{bmatrix} \dfrac{\boldsymbol{cm}^3}{\boldsymbol{mole}}$

$Zc_i := gas\_prop_{7, ii_i}$ $\quad$ $Zc = \begin{bmatrix} 0.274 \\ 0.29 \\ 0.235 \\ 0.288 \end{bmatrix}$ $\quad$ $QGF := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ Quantum Gas Factoris given as 1.38, 0.76, and 0.52 for He, H2, and D2 respectively, and 0 for all other gases

$Dipm_i := gas\_prop_{9, ii_i} \cdot debyes$ $\quad$ $Dipm = \begin{bmatrix} 0 \\ 0 \\ 1.8 \\ 0 \end{bmatrix} debyes$ $\quad$ $ADV_i := gas\_prop_{14, ii_i}$ $\quad$ $ADV = \begin{bmatrix} 26.9 \\ 18.5 \\ 13.1 \\ 16.3 \end{bmatrix}$

Viscosity

(calculated via Lucas Method in referenced worksheet as reported in Reid et al; He will have to be added as required)

Include << ./Viscosity Functions.mcdx

Single gas Viscosity

The results vector below gives single gas viscosities for the gas component data given.

$i := 1 \,..\, 4$

$Visc_i := visc\left(InTemp, Tc_i, Pc_i, Zc_i, Dipm_i, MolWtI_i, QGF_i\right)$ $\qquad Visc = \begin{bmatrix} 151.175 \\ 178.105 \\ 101.26 \\ 204.082 \end{bmatrix} \mu P$

Viscosity of the Gas Mixture

$ViscMix := viscmix\left(InTemp, Tc, Pc, Vc, Zc, Dipm, MolWtI, QGF, InPPall2, 4\right)$ $\qquad ViscMix = 177.896\ \mu P$

$ViscMixML = 177.796\ \mu P$

$$ViscMixErr := \frac{ViscMix - ViscMixML}{ViscMix} = 0.056\ 1\%$$

Molecular Diffusion

Molecular diffusion is calculated in referenced sheet via Fuller's method and Blanc's law as presented in Poling et al. Binary diffusion is more accurate and should be calculated for a binary mixture.

Include << ./Diffusion Functions Fuller Blanc.mcdx

$DiffMix := diffmix\left(InPPall2, MolWtI, ADV, 4, InTemp\right) \cdot \frac{\mathbf{cm}^2}{sec}$ $\qquad DiffMix = 0.154\ \frac{\mathbf{cm}^2}{sec}$ $\quad DiffML = 0.154\ \frac{\mathbf{cm}^2}{\mathbf{s}}$

$$DiffMixErr := \frac{DiffMix - DiffML}{DiffMix} = -0.026\ 1\%$$

$Diff := diff\left(InPPall2, MolWtI, ADV, InTemp\right) \cdot \frac{\mathbf{cm}^2}{sec}$ $\qquad Diff = 0.154\ \frac{\mathbf{cm}^2}{\mathbf{s}}$

$$DiffErr := \frac{Diff - DiffML}{Diff} = -0.026\ 1\%$$

Equivalent Pellet Diameter

Calculated based on Sauter diameter, which applies for cylinders only

$$EqPelDia := \left\| \begin{array}{l} \text{if } PelletLen = 0 \\ \quad \left\| PelletDia \right. \\ \text{else} \\ \quad \left\| \dfrac{3 \cdot PelletDia \cdot PelletLen}{2 \cdot PelletLen + PelletDia} \right. \end{array} \right. = 0.002\ \boldsymbol{m}$$

$$EqPelDiaErr := \frac{EqPelDia - EqPelDiaML}{EqPelDia} = 0\ 1\%$$

Reynolds Number

$$ReynNum := \frac{GasDens \cdot SupVel \cdot EqPelDia}{ViscMix} = 42.911 \qquad ReynNumErr := \frac{ReynNum - ReynNumML}{ReynNum} = -0.056\ 1\%$$

Residence Time

$$ResTime := \frac{BedLength}{InterVel} = 0.325\ \boldsymbol{s}$$

$$ResTimeErr := \frac{ResTime - ResTimeML}{ResTime} = \left(1.71 \cdot 10^{-14}\right)\ 1\%$$

Schmidt Number

$$SchmidtNum := \frac{ViscMix}{GasDens \cdot DiffMix} = 0.955 \qquad SchmidtNumML = 0.954$$

$$SchmidtNumErr := \frac{SchmidtNum - SchmidtNumML}{SchmidtNum} = 0.082\ 1\%$$

Pellet Area to Volume Ratio

AreaVolRat also used in the sorbent energy balance equation at the pellet to gas interface as the solid particle surface area per unit bed volume (Bird equation 13.4-1).

$$AreaVolRat := 2 \cdot \frac{(1 - VoidFraction)}{3 \cdot EqPelDia} = 188.406\ \frac{1}{\boldsymbol{m}}$$

$$AreaVolRatML = 188.406\ \frac{1}{\mathbf{m}}$$

$$AreaVolRatErr := \frac{AreaVolRat - AreaVolRatML}{AreaVolRat} = 0\ 1\%$$

Axial Dispersion

Peclet Number

Axial Dispersion is calculated per Edwards and Richardson. Other correlations will be explored in an independent Mathcad file.

$$PecletNum := \left(\frac{0.73 \cdot VoidFraction}{ReynNum \cdot SchmidtNum} + \frac{1}{2 \cdot \left(1 + \frac{13 \cdot 0.73 \cdot VoidFraction}{ReynNum \cdot SchmidtNum}\right)} \right)^{-1} = 2.133$$

Axial Dispersion Coefficient

$$AxialDisp := \frac{InterVel \cdot EqPelDia}{PecletNum} = \left(8.437 \cdot 10^{-4}\right) \frac{\mathbf{m}^2}{\mathbf{s}} \qquad AxialDispML = \left(8.437 \cdot 10^{-4}\right) \frac{\mathbf{m}^2}{\mathbf{s}}$$

$$AxialDispErr := \frac{AxialDisp - AxialDispML}{AxialDisp} = 0.002\ 1\%$$

Film Diffusion

Film Diffusion is calculated via the Petrovic and Thodos correlation and that of Wakao and Funazkri. Other correlations will be explored in an independent Mathcad file.

$$SherwoodNumPT := 0.357 \cdot \frac{SchmidtNum^{0.33} \cdot ReynNum^{0.64}}{VoidFraction} = 11.139$$

$$SherwoodNum := 2.0 + 1.1 \cdot SchmidtNum^{0.333} \cdot ReynNum^{0.6} = 12.334 \qquad EqPelDia = 0.002\ \mathbf{m}$$

$$FilmDiff := \frac{DiffMix \cdot SherwoodNum}{EqPelDia} = 0.082\ \frac{\mathbf{m}}{\mathbf{s}} \qquad FilmDiffML = 0.082\ \frac{\mathbf{m}}{\mathbf{s}}$$

$$FilmDiffErr := \frac{FilmDiff - FilmDiffML}{FilmDiff} = -0.03\ 1\%$$

Equilibrium Isotherms

Wang and LeVan provided pure component adsorption equilibrium capacities for CO2 and H2O on silica gel, Grace Davison 5A, and Grace Davison 13X. The Matlab array for the Toth parameters and Toth equation is copied below.

```
 'No',    {    1,         2,        3,         4,        5        6    },...
   'System',{ 'CO2_5A',  'H2O_5A', 'CO2_13X',  'H2O_13X', 'CO2_SG', 'H2O_SG' },...
   'a0'    ,{ 9.875E-07, 1.106E-08,6.509E-03, 3.634E-06, 7.678E-06, 1.767E+02},...
   'b0'    ,{ 6.761E-08, 4.714E-10,4.884E-04, 2.408E-07, 5.164E-07, 2.787E-05},...
   'E'     ,{ 5.625E+03, 9.955E+03,2.991E+03, 6.852E+03, 2.330E+03, 1.093E+03},...
   't0'    ,{ 2.700E-01, 3.548E-01,7.487E-02, 3.974E-01,-3.053E-01,-1.190E-03},...
   'c'     ,{-2.002E+01,-5.114E+01,3.805E+01,-4.199E+00, 2.386E+02, 2.213E+01});
```

```
a = TothEq(No).a0*exp(TothEq(No).E/temp);
b = TothEq(No).b0*exp(TothEq(No).E/temp);
t = TothEq(No).t0 + TothEq(No).c/temp;

BedLoad = a*pres / (1 + (b*pres)^t)^(1/t);
```

Create "iso" array with Toth pre-parameters. Select appropriate column based on Sorbent and Sorbate.

$$iso := \begin{bmatrix} 9.875 \cdot 10^{-7} & 1.106 \cdot 10^{-8} & 6.509 \cdot 10^{-3} & 3.634 \cdot 10^{-6} & 7.678 \cdot 10^{-6} & 1.767 \cdot 10^{2} \\ 6.761 \cdot 10^{-8} & 4.714 \cdot 10^{-10} & 4.884 \cdot 10^{-4} & 2.408 \cdot 10^{-7} & 5.164 \cdot 10^{-7} & 2.787 \cdot 10^{-5} \\ 5.625 \cdot 10^{3} & 9.955 \cdot 10^{3} & 2.991 \cdot 10^{3} & 6.852 \cdot 10^{3} & 2.330 \cdot 10^{3} & 1.093 \cdot 10^{3} \\ 2.700 \cdot 10^{-1} & 3.548 \cdot 10^{-1} & 7.487 \cdot 10^{-2} & 3.974 \cdot 10^{-1} & -3.053 \cdot 10^{-1} & -1.19 \cdot 10^{-3} \\ -2.002 \cdot 10 & -5.114 \cdot 10 & 3.805 \cdot 10 & -4.199 & 2.386 \cdot 10^{2} & 2.213 \cdot 10 \end{bmatrix}$$

```
tn :=  if Sorbent = “5A”
           if Sorbate = “CO2”
               1
           if Sorbate = “H2O”
               2
       if Sorbent = “13X”
           if Sorbate = “CO2”
               3
           if Sorbate = “H2O”
               4
       if Sorbent = “Silica Gel”
           if Sorbate = “CO2”
               5
           if Sorbate = “H2O”
               6
```

$tn = 1$

Assign Toth pre-parameters and units; calculate temperature dependent Toth parameters, and calculate bed loading.

$a0 := iso_{1,tn} \cdot \frac{mole}{kg \cdot kPa} = \left(9.875 \cdot 10^{-7}\right) \frac{mole}{kg \cdot kPa}$

$E := iso_{3,tn} \cdot K = \left(5.625 \cdot 10^{3}\right) K$

$c := iso_{5,tn} \cdot K = -20.02\ K$

$b0 := iso_{2,tn} \cdot \frac{1}{kPa} = \left(6.761 \cdot 10^{-8}\right) \frac{1}{kPa}$

$t0 := iso_{4,tn} = 0.27$

$Toth_a := a0 \cdot e^{\frac{E}{InTemp}} = 157.658 \frac{mol}{kg \cdot kPa}$

$Toth_b := b0 \cdot e^{\frac{E}{InTemp}} = 10.794 \frac{1}{kPa}$

$Toth_t := t0 + \frac{c}{InTemp} = 0.203$

$$BedLoad := \frac{Toth_a \cdot InPP}{\left(1 + \left(Toth_b \cdot InPP\right)^{Toth_t}\right)^{\frac{1.}{Toth_t}}} = 1.263 \frac{mol}{kg}$$

$BedLoadML = 1.263 \frac{mol}{kg}$

$$BedLoadErr := \frac{BedLoad - BedLoadML}{BedLoad} = -8.792 \cdot 10^{-14}\ 1\%$$

Note for thermal characterization (zero concentation) a divide by zero error occurs for this error check.

Solid Concentration

$$SolidConc := BedLoad \cdot PartDensity = \left(1.49 \cdot 10^{3}\right) \frac{mol}{m^3}$$

$$SolidConcErr := \frac{SolidConc - SolidConcML}{SolidConc} = -9.155 \cdot 10^{-14}\ 1\%$$

Note for thermal characterization (zero concentation) a divide by zero error occurs for this error check.

Stoichiometric Breakthrough Time

$$FlowRateAct := \frac{FlowRate \cdot InTemp \cdot StandPress}{StandTemp \cdot TotPress} = 29.275 \frac{liter}{min}$$

$$StoichioTime := \frac{BedLoad \cdot Rbar \cdot InTemp \cdot SorbentMass}{InPP \cdot FlowRateAct} = 45.259\ min$$

$StoichioTimeML = 45.259\ min$

$$StoichioTimeErr := \frac{StoichioTime - StoichioTimeML}{StoichioTime} = -8.373 \cdot 10^{-14}\ 1\%$$

Note for thermal characterization (zero concentation) errors occur for this error check.

Gas Heat Capacity Estimation based on Reid values

% CPVAPA, CPVAPB, CPVABC, CPVAPD - constants to calculate the isobaric
% heat capacity of the ideal gas, with Cp in J/(mol*K) and T in kelvins:
% Cp = CPVAPA + (CPVAPB)T + (CPVAPC)T^2 + (CPVAPD)T^3

$j := 1 .. 4$

$$CpVapA_j := gas\_prop_{10,ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}} \qquad CpVapA = \begin{bmatrix} 19.8 \\ 31.15 \\ 31.94 \\ 28.11 \end{bmatrix} \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}} \qquad MolWtI = \begin{bmatrix} 0.044 \\ 0.028 \\ 0.018 \\ 0.032 \end{bmatrix} \frac{\boldsymbol{kg}}{\boldsymbol{mol}}$$

$$CpVapB_j := gas\_prop_{11,ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}^2} \qquad CpVapB = \begin{bmatrix} 0.073 \\ -0.014 \\ 0.001 \\ -3.68 \cdot 10^{-6} \end{bmatrix} \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}^2} \qquad ii = \begin{bmatrix} 4 \\ 1 \\ 3 \\ 2 \end{bmatrix}$$

$$CpVapC_j := gas\_prop_{12,ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}^3} \qquad CpVapC = \begin{bmatrix} -5.602 \cdot 10^{-5} \\ 2.68 \cdot 10^{-5} \\ 2.432 \cdot 10^{-5} \\ 1.746 \cdot 10^{-5} \end{bmatrix} \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}^3}$$

$$CpVapD_j := gas\_prop_{13,ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}^4} \qquad CpVapD = \begin{bmatrix} 1.715 \cdot 10^{-8} \\ -1.168 \cdot 10^{-8} \\ -1.176 \cdot 10^{-8} \\ -1.065 \cdot 10^{-8} \end{bmatrix} \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}^4}$$

Gas Heat Capacity

$$CpMol := CpVapA + CpVapB \cdot InTemp + CpVapC \cdot InTemp^2 + CpVapD \cdot InTemp^3$$

$$CpMol^{T} = \begin{bmatrix} 37.155 & 29.177 & 34.214 & 29.376 \end{bmatrix} \frac{\boldsymbol{J}}{\boldsymbol{mole} \cdot \boldsymbol{K}}$$

Verify Nitrogen gas against published values of 1.040 kJ/kg/K at 25C (Lide, 1993)

$$Cp := \frac{CpMol}{MolWtI} \qquad Cp^{T} = \begin{bmatrix} 0.844 & 1.042 & 1.899 & 0.918 \end{bmatrix} \frac{kJ}{(\boldsymbol{kg} \cdot \boldsymbol{K})} \qquad MolWt = 0.028 \frac{\boldsymbol{kg}}{\boldsymbol{mol}}$$

Sorbate Heat Capacity $\qquad Cp_1 = 0.844 \dfrac{kJ}{(\boldsymbol{kg} \cdot \boldsymbol{K})} \qquad SorbGasHeatCapML = 0.844 \dfrac{kJ}{(\boldsymbol{kg} \cdot \boldsymbol{K})}$

$$GasHeatCap := \frac{\sum_j \left(Cp_j \cdot InPPall2_j\right)}{\sum_j InPPall2_j} = 1.04 \, \frac{kJ}{(\mathbf{kg} \cdot \mathbf{K})} \qquad GasHeatCapML = 1.04 \, \frac{kJ}{(\mathbf{kg} \cdot \mathbf{K})}$$

$$GasHeatCapErr := \frac{GasHeatCap - GasHeatCapML}{GasHeatCap} = -2.186 \cdot 10^{-14} \; 1\%$$

$$SorbGasHeatCapErr := \frac{Cp_1 - SorbGasHeatCapML}{Cp_1} = -1.347 \cdot 10^{-14} \; 1\%$$

Volumetric Heat Capacity as defined by Ruthven:

$$GasHeatCapVol := GasHeatCap \cdot GasDens \qquad GasDens = 1.212 \, \frac{\mathbf{kg}}{\mathbf{m}^3}$$

$$GasHeatCapVol = 1.2603 \, \frac{kJ}{\mathbf{m}^3 \cdot \mathbf{K}}$$

Heat Transfer Coefficient from Particle to Free Stream

via similarity per Ruthven 7.19, with Wakao and Funazkri correlation based on Sherwood number calculated as $Sh_1 \leftarrow 2.0 + 1.1 \cdot Sc^{\frac{1}{3}} \cdot Re^{0.6}$ for calculation of the Fluid-Particle Heat Transfer Coefficient on the basis of recommendations in the literature.

$$SorbGasH := FilmDiff \cdot GasHeatCapVol \qquad SorbGasH = 103.889 \, \frac{\mathbf{W}}{\mathbf{m}^2 \cdot \mathbf{K}} \qquad SorbGasHML = 103.921 \, \frac{\mathbf{W}}{\mathbf{m}^2 \cdot \mathbf{K}}$$

$$SorbGasHErr := \frac{SorbGasH - SorbGasHML}{SorbGasH} = -0.03 \; 1\%$$

Gas Conductivity

Here gas conductivity is based on the mass similarity approach.

$$GasCond := DiffMix \cdot GasHeatCapVol$$

$$GasCond = 0.0194 \, \frac{\mathbf{W}}{\mathbf{m} \cdot \mathbf{K}}$$

Effective Axial Thermal Conductivity

Effective axial thermal conductivity is calculated per Wakao and Kaguei equation 5.69 which is based on thermal test data, including fluid mixing as a function of flow rate but neglecting radiation (which is only important for high temperatures). For this approach the quiescent conductivity is calculated based on the Krupickzka equation.

Effective thermal conductivities of quiescent beds of spherical particles

The Krupickzka equation is used to find effective thermal conductivities of quiescent beds of spherical particles (Wakao eq. 5.33, Kaviany eq. 3.68).

$$SorbCond := 0.12 \cdot \mathbf{W} \cdot \mathbf{m}^{-1} \cdot \mathbf{K}^{-1}$$

$$n_{exp} := 0.280 - 0.757 \cdot \log(VoidFraction) - 0.057 \cdot \log\left(\frac{SorbCond}{GasCond}\right) \qquad n_{exp} = 0.58$$

$$CondQuies := GasCond \cdot \left(\frac{SorbCond}{GasCond}\right)^{n_{exp}} \qquad CondQuies = 0.056 \frac{\mathbf{W}}{\mathbf{m} \cdot \mathbf{K}}$$

Prandtl Number

$$Pr := \frac{GasHeatCapVol \cdot ViscMix}{GasDens \cdot GasCond} \qquad Pr = 0.9551$$

Correlation for Effective Axial Thermal Conductivity

The Yagi et al. correlation is used on the basis of verification against test data in Kavinany 1995, Figure 4.15. Here it is noted that the Yagi et al. prediction is not accurate at packed bed Reynolds numbers above about 50.

$$AxiCond := GasCond \cdot \left(\frac{CondQuies}{GasCond} + 0.75 \cdot Pr \cdot ReynNum\right) \qquad AxiCond = 0.651 \frac{\mathbf{W}}{\mathbf{m} \cdot \mathbf{K}}$$

$$AxiCondErr := \frac{AxiCond - AxiCondML}{AxiCond} = -0.001\ 1\% \qquad AxiCondML = 0.651 \frac{\mathbf{W}}{\mathbf{m} \cdot \mathbf{K}}$$

Heat Transfer to Column Wall

Correlation for Overall Heat Transfer Coefficient (1-D) to Column Wall:

Li and Finlayson (Chem. Eng. Sci. 1977) evaluated published data and isolated the data free from entrance or bed length effects. Their resulting correlations for the wall heat transfer coefficient, applicable to 2-D models, is shown to fit data with packed bed Reynolds number of 10 to 2000 for cylindrical and spherical particles (Wakao and Kaguei, Figure 5.17 and 5.18). Their correlation is also compared with data with packed bed Reynolds numbers from 100 to 2000 in Wen and Ding, and recommended over 4 other correlations. Thus the Li and Finlayson correlation is recommended for use. By extension, the Li and Finlayson correlation for 1-D packed bed models is also recommended. No data was found to compare with the Overall Heater Transfer Coefficient correlations.

Equivalent Column Diameter

$$EqCanID := 2 \cdot \sqrt{\frac{FreeFlowArea}{\pi}} = 4.762\ \mathbf{cm}$$

Correlations of Li and Finlayson as presented in Wakao and Kaguei for Overall Heat Transfer Coefficient (1-D models)

Eq. 5.64 for a cylindrical vessel with spherical packing. Applies for $20 \le Re_p \le 7600$ and $0.05 \le \frac{d_p}{D} \le 0.3$

$$NusseltBead := 2.03 \cdot (ReynNum)^{0.8} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)} \qquad NusseltBead = 30.74 \qquad \frac{EqPelDia}{EqCanID} = 0.048$$

$$GasCanHBead := \frac{GasCond}{EqCanID} \cdot NusseltBead \qquad GasCanHBead = 12.504\ \frac{\mathbf{W}}{\mathbf{m}^2 \cdot \mathbf{K}}$$

$$GasCanHBead := \frac{GasCanHBead - GasCanHML}{GasCanHBead} = -0.071\ 1\%$$

APPENDIX F

VARIANCE IN CORRELATIONS AND DUE TO TEMPERATURE CHANGES

Thermal Film Diffusion and Axial Diffusion Correlation Sensitivity Analysis: CO2 Breakthrough Experiment

Examine variations in correlations for film diffusivity and axial diffusivity in the range of the expected process conditions for the water vapor breakthrough test case for 5A in the MSMBT stand

Include << ./Adsorption Verification Mixed fits database.mcdx

Expected ranges of temperature, sorbate pressure, and total pressure for process.

$TempMin := 297.9 \cdot \mathbf{K}$ $\quad PPMin := 0.01 \cdot \mathbf{kPa}$ $\quad TotPressMin := 15.456\ \mathbf{psi} = 106.5654\ \mathbf{kPa}$

$TempMax := 310.0 \cdot \mathbf{K}$ $\quad PPMax := 1 \cdot \mathbf{kPa}$ $\quad TotPressMax := 15.554\ \mathbf{psi} = 107.2411\ \mathbf{kPa}$

Temperature Sensitivity

$i := 1 \,..\, 11$ $\quad Temp_i := \left(TempMin + \left(\frac{TempMax - TempMin}{10}\right) \cdot (i-1)\right)$ $\quad Temp_4 = 301.53\ \mathbf{K}$

$Temp^{\mathrm{T}} = \begin{bmatrix} 297.9 & 299.11 & 300.32 & 301.53 & 302.74 & 303.95 & 305.16 & 306.37 & 307.58 & 308.79 & 310 \end{bmatrix}\ \mathbf{K}$

Viscosity

$ViscMixT_i := viscmix\left(Temp_i, Tc, Pc, Vc, Zc, Dipm, MolWtI, QGF, InPPall2, 4\right)$ $\quad ViscMixT_4 = \left(1.7962 \cdot 10^{-5}\right)\ Pa \cdot s$

Density

$$GasDensT_i := \frac{TotPress \cdot MolWt}{Rbar \cdot Temp_i}$$

Velocity

$$SupVelT_i := \frac{FlowRate \cdot Temp_i \cdot StandPress}{FreeFlowArea \cdot StandTemp \cdot TotPress}$$

Reynolds Number

$ReT_i := \dfrac{GasDensT_i \cdot SupVelT_i \cdot EqPelDia}{ViscMixT_i}$ $\quad ReT_1 = 42.8995$ $\quad ReT_{11} = 41.6023$

Molecular Diffusivity

$DiffMixT_i := diffmix\left(InPPall2, MolWtI, ADV, 4, Temp_i\right) \cdot \dfrac{\mathbf{cm}^2}{sec}$ $\quad DiffMixT_4 = \left(1.571 \cdot 10^{-5}\right)\ \dfrac{\mathbf{m}^2}{\mathbf{s}}$

Schmidt Number

$$ScT_i := \frac{ViscMixT_i}{GasDensT_i \cdot DiffMixT_i}$$

$k := 1 .. 5$ $$ScT^T = \begin{bmatrix} 0.955 & 0.955 & 0.955 & 0.955 & 0.955 & 0.956 & 0.956 & 0.956 & 0.956 & 0.956 & 0.956 \end{bmatrix}$$

Sherwood Number (Based on five correlations, function at end of Appendix)

$$Sh_{T_i} := Sher\left(ScT_i, ReT_i, VoidFraction\right) \qquad {Sh_{T_4}}^T = \begin{bmatrix} 12.2756 & 11.9073 & 11.0718 & 12.4808 & 10.1957 \end{bmatrix}$$

Film Diffusion Mass Transfer Coefficient (used via similarity to determine thermal film diffusion)

$$k_{fT_i} := \frac{DiffMixT_i}{EqPelDia} \cdot Sh_{T_i} \qquad {k_{fT_4}}^T = \begin{bmatrix} 0.0838 & 0.0813 & 0.0756 & 0.0852 & 0.0696 \end{bmatrix} \frac{\boldsymbol{m}}{\boldsymbol{s}}$$

Particle Peclet Number (Based on five correlations, function at end of Appendix)

$$Pe_{T_i} := Pec\left(ScT_i, ReT_i, VoidFraction\right) \qquad {Pe_{T_4}}^T = \begin{bmatrix} 1.0075 & 2.1346 & 1.9386 & 1.9759 & 2.1524 \end{bmatrix}$$

Axial Dispersion Coefficient

$$D_{LT_i} := \frac{SupVelT_i \cdot EqPelDia}{VoidFraction \cdot Pe_{T_i}} \qquad {D_{LT_4}}^T = \begin{bmatrix} 0.0018 & 0.0009 & 0.0009 & 0.0009 & 0.0008 \end{bmatrix} \frac{\boldsymbol{m}^2}{\boldsymbol{s}}$$

Gas Heat Capacity Estimation based on Reid et al., 1987

CPVAPA, CPVAPB, CPVABC, CPVAPD - constants to calculate the isobaric heat capacity of the ideal gas, with Cp in J/(mol*K) and T in kelvins:

Cp = CPVAPA + (CPVAPB)T + (CPVAPC)T^2 + (CPVAPD)T^3

$j := 1 .. 4$

$$CpVapA_j := gas\_prop_{10, ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole \cdot K}} \qquad CpVapA = \begin{bmatrix} 19.8 \\ 31.15 \\ 31.94 \\ 28.11 \end{bmatrix} \frac{J}{mole \cdot K} \qquad MolWtI = \begin{bmatrix} 0.044 \\ 0.028 \\ 0.018 \\ 0.032 \end{bmatrix} \frac{\boldsymbol{kg}}{\boldsymbol{mol}}$$

$$CpVapB_j := gas\_prop_{11, ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole \cdot K^2}} \qquad CpVapB = \begin{bmatrix} 0.0734 \\ -0.0136 \\ 0.0014 \\ -3.68 \cdot 10^{-6} \end{bmatrix} \frac{J}{mole \cdot K^2}$$

$$CpVapC_j := gas\_prop_{12,ii_j} \cdot \frac{J}{mole \cdot K^3} \qquad CpVapC = \begin{bmatrix} -5.602 \cdot 10^{-5} \\ 2.68 \cdot 10^{-5} \\ 2.432 \cdot 10^{-5} \\ 1.746 \cdot 10^{-5} \end{bmatrix} \frac{J}{mole \cdot K^3}$$

$$CpVapD_j := gas\_prop_{13,ii_j} \cdot \frac{J}{mole \cdot K^4} \qquad CpVapD = \begin{bmatrix} 1.715 \cdot 10^{-8} \\ -1.168 \cdot 10^{-8} \\ -1.176 \cdot 10^{-8} \\ -1.065 \cdot 10^{-8} \end{bmatrix} \frac{J}{mole \cdot K^4}$$

Gas Heat Capacity

$$Cp_j := CpVapA_j + CpVapB_j \cdot Temp + CpVapC_j \cdot Temp^2 + CpVapD_j \cdot Temp^3 \qquad \left(Cp_2\right)_4 = 29.1747 \frac{J}{mole \cdot K}$$

Verify Nitrogen gas against published values of 1.040 kJ/kg/K at 25C (Lide, 1993)

$$CpM := \frac{Cp}{MolWtI} \qquad kJ := 1000 \cdot J \qquad Temp_1 = 297.9\ K \qquad \left(CpM_2\right)_1 = 1.0416 \frac{kJ}{kg \cdot K}$$

$$GasHeatCapMol := \frac{\sum_j \left(Cp_j \cdot InPPall2_j\right)}{\sum_j InPPall2_j} \qquad GasHeatCapMol_4 = 29.2373 \frac{J}{mole \cdot K}$$

Volumetric Heat Capacity as defined by Ruthven:

$$GasHeatCapVol_i := GasHeatCapMol_i \cdot \frac{GasDensT_i}{MolWt}$$

$$GasHeatCapVol^{\mathrm{T}} = \begin{bmatrix} 1.259 & 1.254 & 1.249 & 1.244 & 1.239 & 1.234 & 1.229 & 1.224 & 1.219 & 1.214 & 1.21 \end{bmatrix} \frac{kJ}{m^3 \cdot K}$$

Heat Transfer Coefficient from Particle to Free Stream

via similarity per Ruthven 7.19, with 4 equations for film diffusion considered. Wakao equation is referenced in W-C Yang, Ruthven, and Wakao. Equation 5 is specific to heat transfer (W-C Yang eq. 112)

$$SorbGasH_{T_i} := k_{fT_i} \cdot GasHeatCapVol_i \qquad {SorbGasH_{T_4}}^{\mathrm{T}} = \begin{bmatrix} 104.275 & 101.1465 & 94.0489 & 106.0176 & 86.6071 \end{bmatrix} \frac{W}{m^2 \cdot K}$$

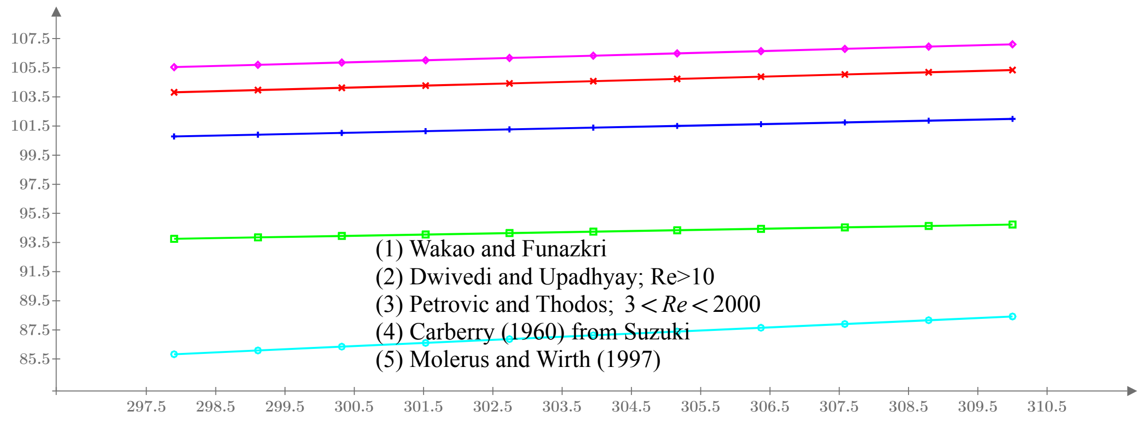


Figure F.1 Heat transfer coefficient from particle to free stream vs. temperature for five correlations. Temperatures are in the range of the carbon dioxide on 5A breakthrough test.

The correlation used in this work is the Wakao and Funazkri correlation. The variation in the coefficient encoutered due to temperature range of the test is:

$$WF_i := \left(\left(SorbGasH_T\right)_i\right)_1$$

$$SorbGasHTempDiff := \max(WF) - min(WF) = 1.5281 \ \frac{W}{m^2 \cdot K}$$

$$SorbGasHTempDiffPer := \frac{SorbGasHTempDiff}{\mathrm{mean}(WF)} = 1.4612 \ 1\%$$

The variation between correlations is found below:

$$SGHT := Unnest\left(SorbGasH_T, 11, 5\right) \qquad HiCorr_i := \max\left(SGHT^{\langle i \rangle}\right) \qquad LoCorr_i := min\left(SGHT^{\langle i \rangle}\right)$$

$$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$$

$$SorbGasHCorrDiff := \max(DiffAll) = 19.719 \ \frac{W}{m^2 \cdot K}$$

$$rownum := \mathrm{match}(\max(DiffAll), DiffAll) = [1] \qquad rownum = [1] \qquad k := rownum_1 = 1$$

$$DiffMean := \mathrm{mean}\left(HiCorr_k, LoCorr_k\right) = 95.6867 \ \frac{W}{m^2 \cdot K}$$

$$SorbGasHCorrDiffPer := \frac{SorbGasHCorrDiff}{2 \cdot DiffMean} = 10.3039 \ 1\%$$

The variation between correlations at all temperatures (neglecting Molerus and Wirth (1997) as it is out of family) is found below:

$$SGHT := Unnest\left(SorbGasH_T, 11, 4\right) \qquad HiCorr_i := \max\left(SGHT^{\langle i \rangle}\right) \qquad LoCorr_i := min\left(SGHT^{\langle i \rangle}\right)$$

$$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$$

$$SorbGasHCorrDiff := \max(DiffAll) = 12.3757 \ \frac{W}{m^2 \cdot K}$$

$rownum := \text{match}\left(\max\left(DiffAll\right), DiffAll\right) = \left[11\right] \qquad rownum = \left[11\right] \qquad k := rownum_1 = 11$

$$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 100.9233\ \frac{W}{m^2 \cdot K}$$

$$SorbGasHCorrDiffPer := \frac{SorbGasHCorrDiff}{2 \cdot DiffMean} = 6.1312\ 1\%$$

Gas Conductivity

Here gas conductivity is based on the mass similarity approach; alternatively it can be calculated via Chapman-Enskog (Bird et al. section 8.3).

$$GasCond_{T_i} := DiffMixT_i \cdot GasHeatCapVol_i$$

$$GasCond_T^{\mathrm{T}} = \left[0.0194\ \ 0.0194\ \ 0.0195\ \ 0.0195\ \ 0.0196\ \ 0.0197\ \ 0.0197\ \ 0.0198\ \ 0.0198\ \ 0.0199\ \ 0.0199\right]\ \frac{W}{m \cdot K}$$

Effective Axial Thermal Conductivity

Effective axial thermal conductivity calculated via similarity per Ruthven 1984 eq. 7.19 and 7.20

$$AxialGasCond_{T_i} := GasHeatCapVol_i \cdot D_{LT_i} \qquad AxialGasCond_{T_4}^{\mathrm{T}} = \left[2.2496\ \ 1.0618\ \ 1.1691\ \ 1.147\ \ 1.053\right]\ \frac{W}{m \cdot K}$$

Effective thermal conductivities of quiescent beds of spherical particles

The Krupickzka equation is used to find effective thermal conductivities of quiescent beds of spherical particles (Wakao and Kaguei, 1982) eq. 5.33, Kaviany 1995 eq. 3.68). For sorb to gas conductivity ratio > 3000, use Hadley's function (Kaviany 1995 eq. 3.49 to 3.51). One difficulty of this equation is that the solid sorbent conductance is not readily available, and is not constant in the case of zeolites due to gas inclusion. Use of GRACE Davison's constant value, however, yielded a very similar resulting quiescant conductivity to test data provided by Griesinger in Figure 2 (0.635 vs. 0.065 for air at 1 bar and 300C)

$$SorbCond := 0.12 \cdot \boldsymbol{W} \cdot \boldsymbol{m}^{-1} \cdot \boldsymbol{K}^{-1}$$

$$n_{exp_i} := 0.280 - 0.757 \cdot \log\left(VoidFraction\right) - 0.057 \cdot \log\left(\frac{SorbCond}{GasCond_{T_i}}\right)$$

$$CondQuies_i := GasCond_{T_i} \cdot \left(\frac{SorbCond}{GasCond_{T_i}}\right)^{n_{exp_i}}$$

$$CondQuies^{\mathrm{T}} = \left[0.0558\ \ 0.0559\ \ 0.0559\ \ 0.056\ \ 0.0561\ \ 0.0562\ \ 0.0562\ \ 0.0563\ \ 0.0564\ \ 0.0565\ \ 0.0566\right]\ \frac{W}{m \cdot K}$$

Packed bed quiescent conductivities are from Honeywell evaluation (SDRL NA PCN 458314-12-77742) for zeolite RK-38.

$$ik := 1\,..\,4 \qquad CondQuies4_T := \begin{bmatrix} 0.1441 \\ 0.1718 \\ 0.2185 \\ 0.2342 \end{bmatrix} \cdot \frac{\boldsymbol{W}}{\boldsymbol{m} \cdot \boldsymbol{K}} \qquad Temp4 := \left(\begin{bmatrix} 20 \\ 100 \\ 204 \\ 300 \end{bmatrix} + 273.15 \right) \cdot \boldsymbol{K}$$

$$LinFitCond_i := \text{slope}\left(\frac{Temp4}{\boldsymbol{K}}, \frac{CondQuies4_T}{\boldsymbol{W} \cdot \boldsymbol{m}^{-1} \cdot \boldsymbol{K}^{-1}} \right) \cdot \frac{Temp_i}{\boldsymbol{K}} + \text{intercept}\left(\frac{Temp4}{\boldsymbol{K}}, \frac{CondQuies4_T}{\boldsymbol{W} \cdot \boldsymbol{m}^{-1} \cdot \boldsymbol{K}^{-1}} \right)$$

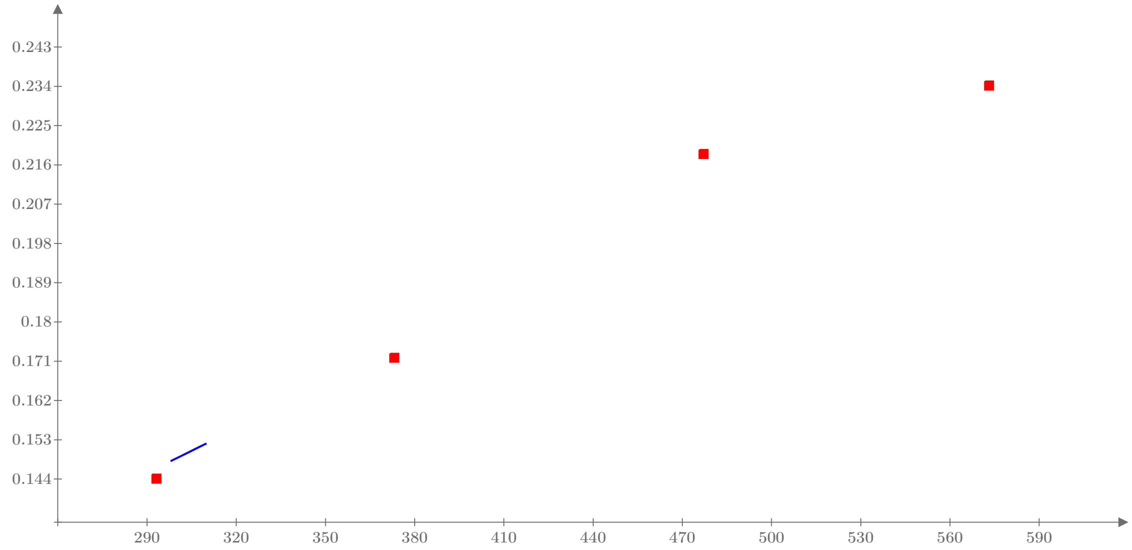


Figure F.2 Packed bed quiescent conductivities vs. temperature: experimental (points) and fit (line). Temperatures for the fitted data are in the range of the carbon dioxide on 5A breakthrough test.

$$SlopeCond := \text{slope}\left(\frac{Temp4}{\boldsymbol{K}}, \frac{CondQuies4_T}{\boldsymbol{W} \cdot \boldsymbol{m}^{-1} \cdot \boldsymbol{K}^{-1}} \right) = 0.0003$$

$$InterCond := \text{intercept}\left(\frac{Temp4}{\boldsymbol{K}}, \frac{CondQuies4_T}{\boldsymbol{W} \cdot \boldsymbol{m}^{-1} \cdot \boldsymbol{K}^{-1}} \right) = 0.0481$$

$$CondQuiesHWI_{T_i} := SlopeCond \cdot \frac{\boldsymbol{W}}{\boldsymbol{m} \cdot \boldsymbol{K}^2} \cdot Temp_i + InterCond \cdot \frac{\boldsymbol{W}}{\boldsymbol{m} \cdot \boldsymbol{K}}$$

$$CondQuiesHWI_T{}^{\mathrm{T}} = \begin{bmatrix} 0.1481 & 0.1485 & 0.1489 & 0.1493 & 0.1497 & 0.1501 & 0.1505 & 0.1509 & 0.1513 & \ldots \end{bmatrix} \frac{W}{m \cdot K}$$

Prandtl Number

$$Pr_{T_i} := \frac{GasHeatCapVol_i \cdot ViscMixT_i}{GasDensT_i \cdot GasCond_{T_i}}$$

$$Pr_T^{\mathrm{T}} = \begin{bmatrix} 0.9551 & \dots \end{bmatrix}$$

```
$AGC_T :=$ for $i \in 1..11$
    for $j \in 1..5$
        $AGC_{T_{i,j}} \leftarrow \left(AxialGasCond_{T_i}\right)_j$
    $AGC_T$
```

Effective Axial Thermal Conductivity equation recommended by Wakao and Kaguei (1982) for a wide range of Reynolds numbers (equation 5.69), and variation of Yagi et al. 1960 per Kaviany 1995 (with quiescant conductivity via HWI data and per Krupickzka equation).

$$AGC_{T_{i,6}} := GasCond_{T_i} \cdot \left(\frac{CondQuies_i}{GasCond_{T_i}} + 0.5 \cdot Pr_{T_i} \cdot ReT_i\right) \qquad AGC_{T_{i,7}} := GasCond_{T_i} \cdot \left(\frac{CondQuies_i}{GasCond_{T_i}} + 0.75 \cdot Pr_{T_i} \cdot ReT_i\right)$$

$$AGC_{T_{i,8}} := GasCond_{T_i} \cdot \left(\frac{CondQuiesHWI_{T_i}}{GasCond_{T_i}} + 0.75 \cdot Pr_{T_i} \cdot ReT_i\right)$$

$$\left(\left(AGC_T^{\mathrm{T}}\right)^{\langle 4\rangle}\right)^{\mathrm{T}} = \begin{bmatrix} 2.2496 & 1.0618 & 1.1691 & 1.147 & 1.053 & 0.4526 & 0.6509 & 0.7442 \end{bmatrix} W \cdot m^{-1} \cdot K^{-1}$$

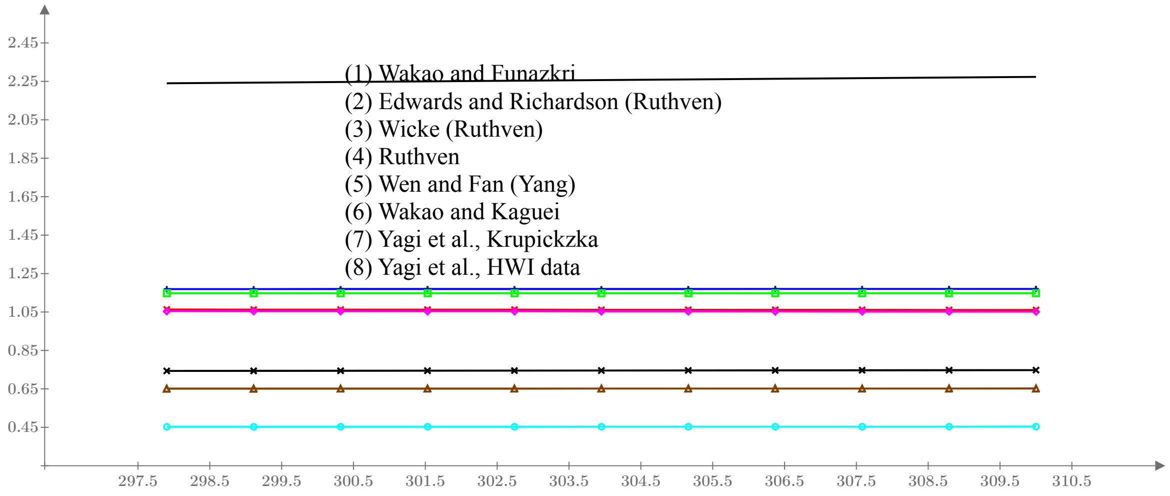


Figure F.3 Effective axial thermal conductivity versus temperature for eight correlations. Temperatures are in the range of the carbon dioxide on 5A breakthrough test.

Note that the first 5 correlations are based on similarity with mass dispersion and correlations for Peclet number. The sixth is recommended by Wakao and Kaguei 1982. Yagi et al. correlation is compared with test data in Kavinany 1995.

The correlation used in this work is Yagi et al. with Krupickzka. The variation in the coefficient encoutered due to temperature range of the test is:

$YK_i := \left(AGC_T^{\langle 7\rangle}\right)_i$

$AxiCondTempDiff := \max\left(YK\right) - min\left(YK\right) = 0.0007\ W \cdot m^{-1} \cdot K^{-1}$

$$AxiCondTempDiffPer := \frac{AxiCondTempDiff}{\text{mean}\left(YK\right)} = 0.1128\ 1\%$$

The variation between all correlations at all temperatures is:

$HiCorr_i := \max\left(AGC_T^{\underset{\smile}{\overset{\frown}{i}}}\right) \qquad LoCorr_i := min\left(AGC_T^{\underset{\smile}{\overset{\frown}{i}}}\right)$

$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$

$$AxiCondCorrDiff := \max\left(DiffAll\right) = 1.8198\ \frac{\boldsymbol{W}}{\boldsymbol{m} \cdot \boldsymbol{K}}$$

$rownum := \text{match}\left(\max\left(DiffAll\right), DiffAll\right) = \left[11\right] \qquad rownum = \left[11\right] \qquad k := rownum_1 = 11$

$$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 1.3631\ \frac{\boldsymbol{W}}{\boldsymbol{m} \cdot \boldsymbol{K}} \qquad AxiCondCorrDiffPer := \frac{AxiCondCorrDiff}{2 \cdot DiffMean} = 66.7546\ 1\%$$

The variation between the last three correlations (not based on similarity) at all temperatures is:

$AC3^{\langle 1\rangle} := AGC_T^{\langle 6\rangle} \qquad AC3^{\langle 2\rangle} := AGC_T^{\langle 7\rangle} \qquad AC3^{\langle 3\rangle} := AGC_T^{\langle 8\rangle} \qquad k := 1\,..\,3$

$HiCorr_i := \max\left(AC3^{\underset{\smile}{\overset{\frown}{i}}}\right) \qquad LoCorr_i := min\left(AC3^{\underset{\smile}{\overset{\frown}{i}}}\right)$

$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$

$$AxiCondCorrDiff := \max\left(DiffAll\right) = 0.2939\ \frac{\boldsymbol{W}}{\boldsymbol{m} \cdot \boldsymbol{K}}$$

$rownum := \text{match}\left(\max\left(DiffAll\right), DiffAll\right) = \left[11\right] \qquad rownum = \left[11\right] \qquad k := rownum_1 = 11$

$$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 0.6001\ \frac{\boldsymbol{W}}{\boldsymbol{m} \cdot \boldsymbol{K}} \qquad AxiCondCorrDiffPer := \frac{AxiCondCorrDiff}{2 \cdot DiffMean} = 24.4879\ 1\%$$

Equivalent Column Diameter

$$EqCanID := 2 \cdot \sqrt{\frac{FreeFlowArea}{\pi}} = 4.7625\ cm$$

Heat Transfer to Column Wall

Correlation of Leva as presented in Ruthven 1984 eq. 7.26 [M. Leva, Chem. Eng. 56, 115 (1949)]

Heat Transfer coefficient at internal surface of the vessel:

$$Nu2DCol_{T_{i,1}} := 0.813 \cdot \left(ReT_i\right)^{0.19} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)}$$

$$\left(Nu2DCol_T^{\langle 1 \rangle}\right)^T = \left[1.2428 \quad 1.2421 \quad 1.2414 \quad 1.2406 \quad 1.2399 \quad 1.2392 \quad 1.2384 \quad 1.2377 \quad 1.237 \quad 1.2363 \quad 1.2356\right]$$

Correlations of Rase 1900 as presented in W.C. Yang 2003

9.1.2 Heat Transfer Through Wall - One Dimensional Model - Axial; q = hA(T-Tw) where A is inside surface of cylindrical vessel and Tw = wall temperature of fluid. T of the fluid and bed are assumed identical.

For a cylindrical vessel with spherical packing, Applies for $20 \le Re_p \le 7600$ and $0.05 \le \frac{d_p}{D} \le 0.3$

$$Nu1DCol_{T_{i,1}} := 2.26 \cdot \left(ReT_i\right)^{0.8} \cdot \left(Pr_{T_i}\right)^{0.33} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)} \qquad \frac{EqPelDia}{EqCanID} = 0.0483$$

$$\left(Nu1DCol_T^{\langle 1 \rangle}\right)^T = \left[33.7007 \quad 33.6173 \quad 33.5344 \quad 33.4522 \quad 33.3705 \quad 33.2894 \quad 33.2088 \quad 33.1288 \quad 33.0494 \quad \ldots\right]$$

For a cylindrical vessel with cylindrical packing, Applies for $20 \le Re_p \le 8000$ and $0.03 \le \frac{d_p}{D} \le 0.2$

$$Nu1DCol_{T_{i,2}} := 1.40 \cdot \left(ReT_i\right)^{0.95} \cdot \left(Pr_{T_i}\right)^{0.33} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)} \qquad \frac{EqPelDia}{EqCanID} = 0.0483$$

$$\left(Nu1DCol_T^{\langle 2 \rangle}\right)^T = \left[36.6882 \quad 36.5801 \quad 36.4729 \quad 36.3665 \quad 36.2608 \quad 36.156 \quad 36.052 \quad 35.9487 \quad 35.8462 \quad \ldots\right]$$

Correlations of Li and Finlayson as presented in Wakao and Kaguei 1982 for Overall Heat Transfer Coefficient (1-D models) to Column Wall for Spherical and Cylindrical Pellets

Eq. 5.64 for a cylindrical vessel with spherical packing. Applies for $20 \leq Re_p \leq 7600$ and $0.05 \leq \frac{d_p}{D} \leq 0.3$

$$Nu1DCol_{T_{i,3}} := 2.03 \cdot \left(ReT_i\right)^{0.8} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)}$$

$$\frac{EqPelDia}{EqCanID} = 0.0483$$

$$\left(Nu1DCol_T^{\langle 3 \rangle}\right)^T = [\,30.7337 \;\; 30.6566 \;\; 30.5801 \;\; 30.5041 \;\; 30.4287 \;\; 30.3539 \;\; 30.2796 \;\; 30.2059 \;\; 30.1327 \;\; \ldots\,]$$

Eq. 5.65 or a cylindrical vessel with cylindrical packing. Applies for $20 \leq Re_p \leq 800$ and $0.03 \leq \frac{d_p}{D} \leq 0.2$

$$Nu1DCol_{T_{i,4}} := 1.26 \cdot \left(ReT_i\right)^{0.95} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)}$$

$$EqPelDia = 0.0023\ \boldsymbol{m}$$

$$PelletDia = 0.0023\ \boldsymbol{m}$$

$$\left(Nu1DCol_T^{\langle 4 \rangle}\right)^T = [\,33.5241 \;\; 33.4242 \;\; 33.3252 \;\; 33.2269 \;\; 33.1295 \;\; 33.0327 \;\; 32.9368 \;\; 32.8416 \;\; 32.7471 \;\; \ldots\,]$$

$$jj := 1 \,..\, 4$$

$$GasCanH_{1DT_{i,jj}} := \frac{GasCond_{T_i}}{EqCanID} \cdot Nu1DCol_{T_{i,jj}}$$

$$\left(\left(GasCanH_{1DT}{}^T\right)^{\langle 4 \rangle}\right)^T = [\,13.7231 \;\; 14.9187 \;\; 12.5138 \;\; 13.6307\,]\ \frac{W}{m^2 \cdot K}$$

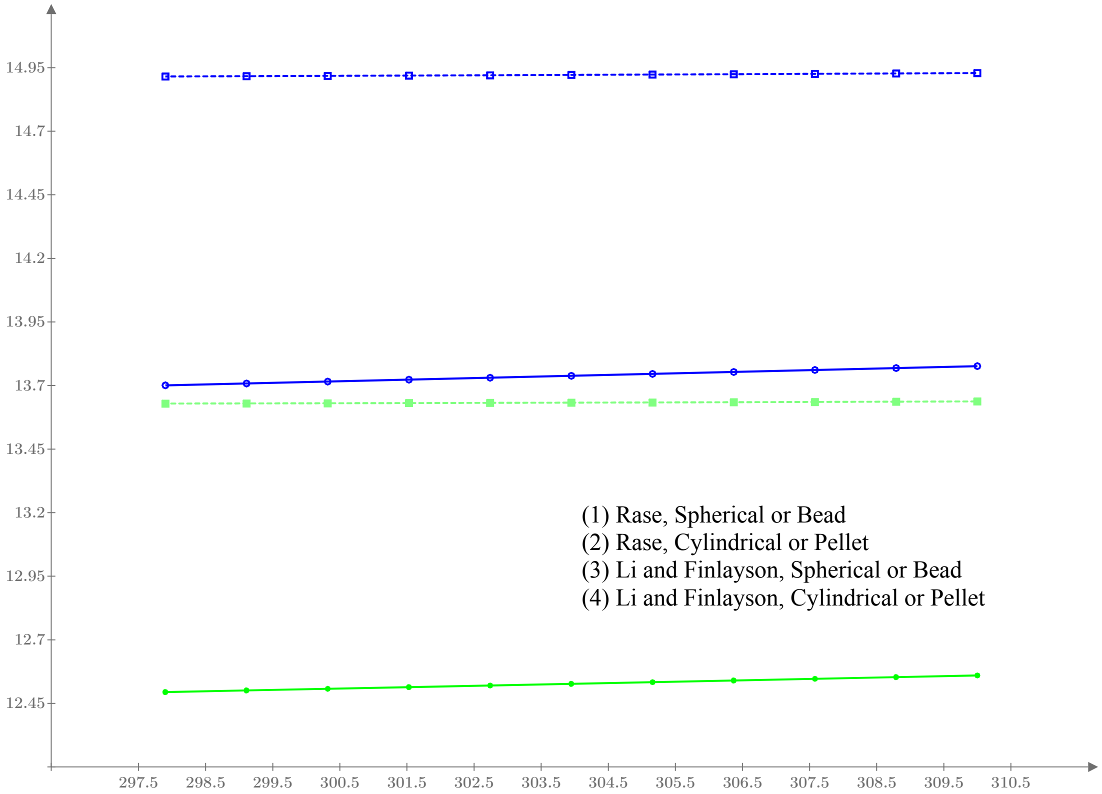


Figure F.4 Overall 1-D heat transfer coefficient to column wall vs. temperature for spherical and cylindrical pellets. Temperatures are in the range of the carbon dioxide on 5A breakthrough test.

The correlation used in this work is Li and Finlayson for a bead. The variation in the coefficient encoutered due to temperature range of the test is:

$LF_i := \left(GasCanH_{1DT}^{\langle 3\rangle}\right)_i$

$GasCanHTempDiff := \max\left(LF\right) - min\left(LF\right) = 0.0655\ \mathbf{W \cdot m^{-2} \cdot K^{-1}}$

$GasCanHTempDiffPer := \dfrac{GasCanHTempDiff}{\text{mean}\left(LF\right)} = 0.5225\ 1\%$

The variation between the correlations for beads at all temperatures is:

$GasCanH_{1DTB}^{\langle 1\rangle} := GasCanH_{1DT}^{\langle 1\rangle}$ $GasCanH_{1DTB}^{\langle 2\rangle} := GasCanH_{1DT}^{\langle 3\rangle}$

$HiCorr_i := \max\left(GasCanH_{1DTB}^{\overset{\frown}{\underset{\smile}{i}}}\right)$ $LoCorr_i := min\left(GasCanH_{1DTB}^{\overset{\frown}{\underset{\smile}{i}}}\right)$

$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$

$GasCanHTempDiff := \max\left(DiffAll\right) = 1.2164\ \mathbf{\dfrac{W}{m^2 \cdot K}}$

$rownum := \text{match}\left(\max\left(DiffAll\right), DiffAll\right) = \left[11\right]$ $rownum = \left[11\right]$ $k := rownum_1 = 11$

$$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 13.168\ \frac{\boldsymbol{W}}{\boldsymbol{m}^2 \cdot \boldsymbol{K}}$$

$$GasCanHTempDiffPer := \frac{GasCanHTempDiff}{2 \cdot DiffMean} = 4.6186\ 1\%$$

Thermal Film Diffusion and Axial Diffusion Correlation Sensitivity Analysis: H2O Breakthrough Experiment

Examine variations in correlations for film diffusivity and axial diffusivity in the range of the expected process conditions for the water vapor breakthrough test case for 5A in the MSMBT stand

Include << ./Adsorption Verification Mixed LDF and DL fits database.mcdx

Expected ranges of temperature, sorbate pressure, and total pressure for process.

$TempMin := 295.9 \cdot K$ $\quad PPMin := 0.01 \cdot kPa$ $\quad TotPressMin := 15.456\ psi = 106.5654\ kPa$

$TempMax := 314.5 \cdot K$ $\quad PPMax := 1 \cdot kPa$ $\quad TotPressMax := 15.554\ psi = 107.2411\ kPa$

Temperature Sensitivity

$i := 1 \,..\, 11$ $\quad Temp_i := \left(TempMin + \left(\dfrac{TempMax - TempMin}{10}\right) \cdot (i-1)\right)$ $\quad Temp_4 = 301.48\ K$

$Temp^{\mathrm{T}} = [295.9 \quad 297.76 \quad 299.62 \quad 301.48 \quad 303.34 \quad 305.2 \quad 307.06 \quad 308.92 \quad 310.78 \quad 312.64 \quad 314.5]\ K$

Viscosity

$ViscMixT_i := viscmix\left(Temp_i, Tc, Pc, Vc, Zc, Dipm, MolWtI, QGF, InPPall2, 4\right)$ $\quad ViscMixT_4 = \left(1.7853 \cdot 10^{-5}\right)\ Pa \cdot s$

Density

$$GasDensT_i := \frac{TotPress \cdot MolWt}{Rbar \cdot Temp_i}$$

Velocity

$$SupVelT_i := \frac{FlowRate \cdot Temp_i \cdot StandPress}{FreeFlowArea \cdot StandTemp \cdot TotPress}$$

Reynolds Number

$ReT_i := \dfrac{GasDensT_i \cdot SupVelT_i \cdot EqPelDia}{ViscMixT_i}$ $\quad ReT_1 = 43.4548$ $\quad ReT_{11} = 41.4514$

Molecular Diffusivity

$DiffMixT_i := diffmix\left(InPPall2, MolWtI, ADV, 4, Temp_i\right) \cdot \dfrac{cm^2}{sec}$ $\quad DiffMixT_4 = \left(2.4633 \cdot 10^{-5}\right) \dfrac{m^2}{s}$

Schmidt Number

$$ScT_i := \frac{ViscMixT_i}{GasDensT_i \cdot DiffMixT_i}$$

$k := 1 .. 5$ $\qquad ScT^{\mathrm{T}} = [0.602\ \ 0.602\ \ 0.602\ \ 0.602\ \ 0.602\ \ 0.602\ \ 0.602\ \ 0.602\ \ 0.602\ \ 0.602\ \ 0.602]$

Sherwood Number (Based on five correlations, function at end of Appendix)

$Sh_{T_i} := Sher\left(ScT_i, ReT_i, VoidFraction\right)$ $\qquad {Sh_{T_4}}^{\mathrm{T}} = [10.8496\ \ 10.8757\ \ 10.1318\ \ 11.0611\ \ 11.1464]$

Film Diffusion Mass Transfer Coefficient (used via similarity to determine thermal film diffusion)

$k_{fT_i} := \frac{DiffMixT_i}{EqPelDia} \cdot Sh_{T_i}$ $\qquad {k_{fT_4}}^{\mathrm{T}} = [0.1152\ \ 0.1155\ \ 0.1076\ \ 0.1174\ \ 0.1183]\ \frac{\boldsymbol{m}}{\boldsymbol{s}}$

Particle Peclet Number (Based on five correlations, function at end of Appendix)

$Pe_{T_i} := Pec\left(ScT_i, ReT_i, VoidFraction\right)$ $\qquad {Pe_{T_4}}^{\mathrm{T}} = [0.7837\ \ 2.1969\ \ 1.9066\ \ 1.9644\ \ 2.2352]$

Axial Dispersion Coefficient

$D_{LT_i} := \frac{SupVelT_i \cdot EqPelDia}{VoidFraction \cdot Pe_{T_i}}$ $\qquad {D_{LT_4}}^{\mathrm{T}} = [0.0025\ \ 0.0009\ \ 0.001\ \ 0.001\ \ 0.0009]\ \frac{\boldsymbol{m}^2}{\boldsymbol{s}}$

Gas Heat Capacity Estimation based on Reid et al., 1987

CPVAPA, CPVAPB, CPVABC, CPVAPD - constants to calculate the isobaric heat capacity of the ideal gas, with Cp in J/(mol*K) and T in kelvins:

Cp = CPVAPA + (CPVAPB)T + (CPVAPC)T^2 + (CPVAPD)T^3

$j := 1 .. 4$

$CpVapA_j := gas\_prop_{10, ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole \cdot K}}$ $\qquad CpVapA = \begin{bmatrix} 31.94 \\ 31.15 \\ 19.8 \\ 28.11 \end{bmatrix} \frac{J}{mole \cdot K}$ $\qquad MolWtI = \begin{bmatrix} 0.018 \\ 0.028 \\ 0.044 \\ 0.032 \end{bmatrix} \frac{\boldsymbol{kg}}{\boldsymbol{mol}}$

$CpVapB_j := gas\_prop_{11, ii_j} \cdot \frac{\boldsymbol{J}}{\boldsymbol{mole \cdot K^2}}$ $\qquad CpVapB = \begin{bmatrix} 0.0014 \\ -0.0136 \\ 0.0734 \\ -3.68 \cdot 10^{-6} \end{bmatrix} \frac{J}{mole \cdot K^2}$

$$CpVapC_j := gas\_prop_{12,ii_j} \cdot \frac{J}{mole \cdot K^3} \qquad CpVapC = \begin{bmatrix} 2.432 \cdot 10^{-5} \\ 2.68 \cdot 10^{-5} \\ -5.602 \cdot 10^{-5} \\ 1.746 \cdot 10^{-5} \end{bmatrix} \frac{J}{mole \cdot K^3}$$

$$CpVapD_j := gas\_prop_{13,ii_j} \cdot \frac{J}{mole \cdot K^4} \qquad CpVapD = \begin{bmatrix} -1.176 \cdot 10^{-8} \\ -1.168 \cdot 10^{-8} \\ 1.715 \cdot 10^{-8} \\ -1.065 \cdot 10^{-8} \end{bmatrix} \frac{J}{mole \cdot K^4}$$

Gas Heat Capacity

$$Cp_j := CpVapA_j + CpVapB_j \cdot Temp + CpVapC_j \cdot Temp^2 + CpVapD_j \cdot Temp^3 \qquad \left(Cp_2\right)_4 = 29.1747 \frac{J}{mole \cdot K}$$

Verify Nitrogen gas against published values of 1.040 kJ/kg/K at 25C (Lide, 1993)

$$CpM := \frac{Cp}{MolWtI} \qquad kJ := 1000 \cdot J \qquad Temp_1 = 295.9\ K \qquad \left(CpM_2\right)_1 = 1.0416 \frac{kJ}{kg \cdot K}$$

$$GasHeatCapMol := \frac{\sum_j \left(Cp_j \cdot InPPall2_j\right)}{\sum_j InPPall2_j} \qquad GasHeatCapMol_4 = 29.2126 \frac{J}{mole \cdot K}$$

Volumetric Heat Capacity as defined by Ruthven:

$$GasHeatCapVol_i := GasHeatCapMol_i \cdot \frac{GasDensT_i}{MolWt}$$

$$GasHeatCapVol^{\mathrm{T}} = \begin{bmatrix} 1.283 & 1.275 & 1.267 & 1.259 & 1.251 & 1.244 & 1.236 & 1.229 & 1.221 & 1.214 & 1.207 \end{bmatrix} \frac{kJ}{m^3 \cdot K}$$

Heat Transfer Coefficient from Particle to Free Stream

via similarity per Ruthven 7.19, with 4 equations for film diffusion considered. Wakao equation is referenced in W-C Yang, Ruthven, and Wakao. Equation 5 is specific to heat transfer (W-C Yang eq. 112)

$$SorbGasH_{T_i} := k_{fT_i} \cdot GasHeatCapVol_i \qquad {SorbGasH_{T_4}}^{\mathrm{T}} = \begin{bmatrix} 145.0491 & 145.399 & 135.4535 & 147.877 & 149.0172 \end{bmatrix} \frac{W}{m^2 \cdot K}$$

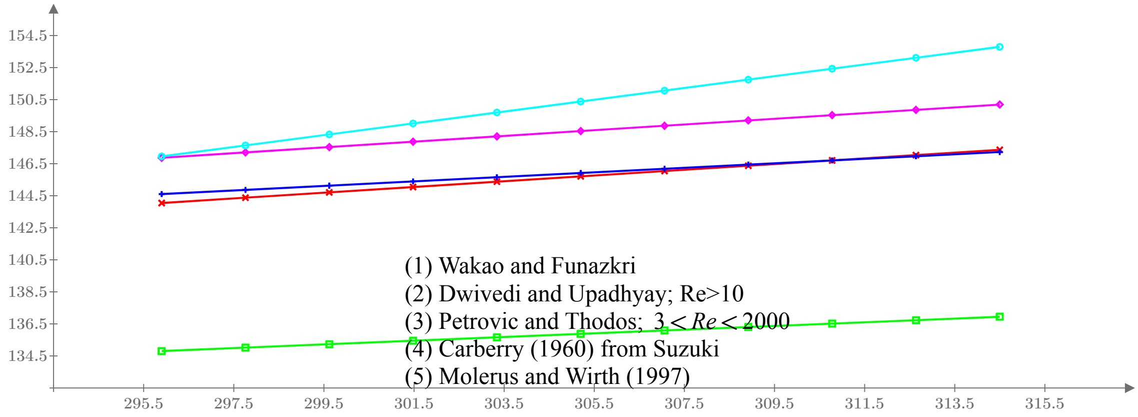


Figure F.5 Heat transfer coefficient from particle to free stream vs. temperature for five correlations. Temperatures are in the range of the water vapor on 5A breakthrough test.

The correlation used in this work is the Wakao and Funazkri correlation. The variation in the coefficient encoutered due to temperature range of the test is:

$$WF_i := \left(\left(SorbGasH_{T_i}\right)_1\right)$$

$$SorbGasHTempDiff := \max(WF) - min(WF) = 3.3232\ \frac{W}{m^2 \cdot K}$$

$$SorbGasHTempDiffPer := \frac{SorbGasHTempDiff}{\text{mean}(WF)} = 2.2806\ 1\%$$

The variation between correlations is found below:

$$SGHT := Unnest\left(SorbGasH_T, 11, 5\right) \qquad HiCorr_i := \max\left(SGHT^{\langle i \rangle}\right) \qquad LoCorr_i := min\left(SGHT^{\langle i \rangle}\right)$$

$$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$$

$$SorbGasHCorrDiff := \max(DiffAll) = 16.8478\ \frac{W}{m^2 \cdot K}$$

$$rownum := \text{match}(\max(DiffAll), DiffAll) = [11] \qquad rownum = [11] \qquad k := rownum_1 = 11$$

$$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 145.3748\ \frac{W}{m^2 \cdot K}$$

$$SorbGasHCorrDiffPer := \frac{SorbGasHCorrDiff}{2 \cdot DiffMean} = 5.7946\ 1\%$$

The variation between correlations at all temperatures (neglecting Molerus and Wirth (1997) as it is out of family) is found below:

$$SGHT := Unnest\left(SorbGasH_T, 11, 4\right) \qquad HiCorr_i := \max\left(SGHT^{\langle i \rangle}\right) \qquad LoCorr_i := min\left(SGHT^{\langle i \rangle}\right)$$

$$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$$

$$SorbGasHCorrDiff := \max(DiffAll) = 13.2464\ \frac{W}{m^2 \cdot K}$$

$rownum := \text{match}\left(\max\left(DiffAll\right), DiffAll\right) = [11]$ $\quad rownum = [11]$ $\quad k := rownum_1 = 11$

$$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 143.5741\ \frac{W}{m^2 \cdot K}$$

$$SorbGasHCorrDiffPer := \frac{SorbGasHCorrDiff}{2 \cdot DiffMean} = 4.6131\ 1\%$$

Gas Conductivity

Here gas conductivity is based on the mass similarity approach; alternatively it can be calculated via Chapman-Enskog (Bird et al. section 8.3).

$$GasCond_{T_i} := DiffMixT_i \cdot GasHeatCapVol_i$$

$$GasCond_T^{\mathrm{T}} = [0.0306\ \ 0.0307\ \ 0.0309\ \ 0.031\ \ 0.0312\ \ 0.0313\ \ 0.0314\ \ 0.0316\ \ 0.0317\ \ 0.0319\ \ 0.032]\ \frac{W}{m \cdot K}$$

Effective Axial Thermal Conductivity

Effective axial thermal conductivity calculated via similarity per Ruthven 1984 eq. 7.19 and 7.20

$AxialGasCond_{T_i} := GasHeatCapVol_i \cdot D_{LT_i}$ $\quad AxialGasCond_{T_4}^{\mathrm{T}} = [3.0911\ \ 1.1027\ \ 1.2707\ \ 1.2332\ \ 1.0839]\ \frac{W}{m \cdot K}$

Effective thermal conductivities of quiescent beds of spherical particles

The Krupickzka equation is used to find effective thermal conductivities of quiescent beds of spherical particles (Wakao and Kaguei, 1982) eq. 5.33, Kaviany 1995 eq. 3.68). For sorb to gas conductivity ratio > 3000, use Hadley's function (Kaviany 1995 eq. 3.49 to 3.51). One difficulty of this equation is that the solid sorbent conductance is not readily available, and is not constant in the case of zeolites due to gas inclusion. Use of GRACE Davison's constant value, however, yielded a very similar resulting quiescant conductivity to test data provided by Griesinger in Figure 2 (0.635 vs. 0.065 for air at 1 bar and 300C)

$$SorbCond := 0.12 \cdot W \cdot m^{-1} \cdot K^{-1}$$

$$n_{exp_i} := 0.280 - 0.757 \cdot \log\left(VoidFraction\right) - 0.057 \cdot \log\left(\frac{SorbCond}{GasCond_{T_i}}\right)$$

$$CondQuies_i := GasCond_{T_i} \cdot \left(\frac{SorbCond}{GasCond_{T_i}}\right)^{n_{exp_i}}$$

$$CondQuies^{\mathrm{T}} = [0.0705\ \ 0.0706\ \ 0.0708\ \ 0.0709\ \ 0.071\ \ 0.0712\ \ 0.0713\ \ 0.0714\ \ 0.0716\ \ 0.0717\ \ 0.0718]\ \frac{W}{m \cdot K}$$

Packed bed quiescent conductivities are from Honeywell evaluation (SDRL NA PCN 458314-12-77742) for zeolite RK-38.

$$ik := 1..4 \qquad CondQuies4_T := \begin{bmatrix} 0.1441 \\ 0.1718 \\ 0.2185 \\ 0.2342 \end{bmatrix} \cdot \frac{W}{m \cdot K} \qquad Temp4 := \left(\begin{bmatrix} 20 \\ 100 \\ 204 \\ 300 \end{bmatrix} + 273.15 \right) \cdot K$$

$$LinFitCond_i := \text{slope}\left(\frac{Temp4}{K}, \frac{CondQuies4_T}{W \cdot m^{-1} \cdot K^{-1}} \right) \cdot \frac{Temp_i}{K} + \text{intercept}\left(\frac{Temp4}{K}, \frac{CondQuies4_T}{W \cdot m^{-1} \cdot K^{-1}} \right)$$

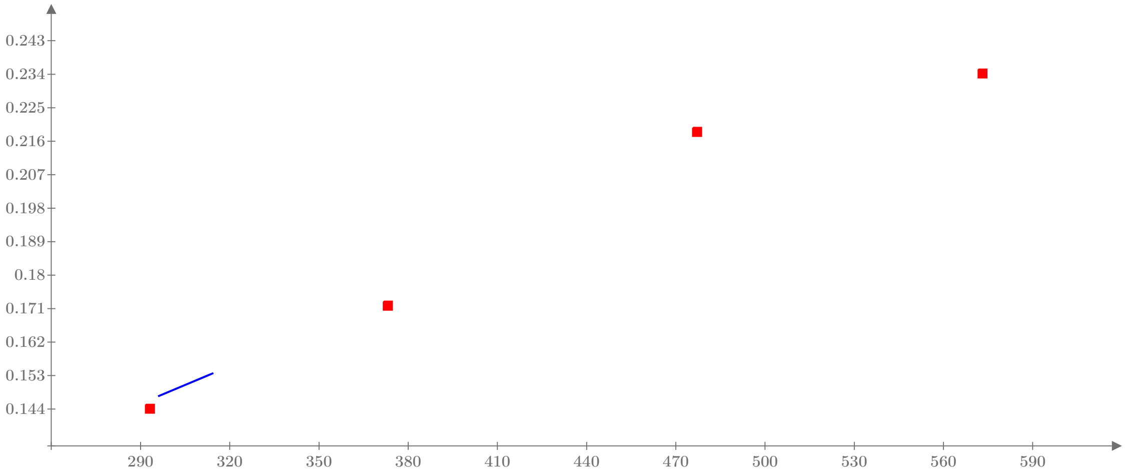


Figure F.6 Packed bed quiescent conductivities vs. temperature: experimental (points) and fit (line). Temperatures for the fitted data are in the range of the water vapor on 5A breakthrough test.

$$SlopeCond := \text{slope}\left(\frac{Temp4}{K}, \frac{CondQuies4_T}{W \cdot m^{-1} \cdot K^{-1}} \right) = 0.0003$$

$$InterCond := \text{intercept}\left(\frac{Temp4}{K}, \frac{CondQuies4_T}{W \cdot m^{-1} \cdot K^{-1}} \right) = 0.0481$$

$$CondQuiesHWI_{T_i} := SlopeCond \cdot \frac{W}{m \cdot K^2} \cdot Temp_i + InterCond \cdot \frac{W}{m \cdot K}$$

$$CondQuiesHWI_T^{\mathrm{T}} = \begin{bmatrix} 0.1474 & 0.148 & 0.1487 & 0.1493 & 0.1499 & 0.1505 & 0.1512 & 0.1518 & 0.1524 & \ldots \end{bmatrix} \frac{W}{m \cdot K}$$

Prandtl Number

$$Pr_{T_i} := \frac{GasHeatCapVol_i \cdot ViscMixT_i}{GasDensT_i \cdot GasCond_{T_i}}$$

$$Pr_T^{\mathrm{T}} = \left[0.6015 \quad \dots \right]$$

```
$AGC_T :=$ for $i \in 1..11$
    for $j \in 1..5$
        $AGC_{T_{i,j}} \leftarrow \left(AxialGasCond_{T_i}\right)_j$
    $AGC_T$
```

Effective Axial Thermal Conductivity equation recommended by Wakao and Kaguei (1982) for a wide range of Reynolds numbers (equation 5.69), and variation of Yagi et al. 1960 per Kaviany 1995 (with quiescant conductivity via HWI data and per Krupickzka equation).

$$AGC_{T_{i,6}} := GasCond_{T_i} \cdot \left(\frac{CondQuies_i}{GasCond_{T_i}} + 0.5 \cdot Pr_{T_i} \cdot ReT_i \right) \qquad AGC_{T_{i,7}} := GasCond_{T_i} \cdot \left(\frac{CondQuies_i}{GasCond_{T_i}} + 0.75 \cdot Pr_{T_i} \cdot ReT_i \right)$$

$$AGC_{T_{i,8}} := GasCond_{T_i} \cdot \left(\frac{CondQuiesHWI_{T_i}}{GasCond_{T_i}} + 0.75 \cdot Pr_{T_i} \cdot ReT_i \right)$$

$$\left(\left(AGC_T^{\mathrm{T}} \right)^{\langle 4 \rangle} \right)^{\mathrm{T}} = \left[3.0911 \quad 1.1027 \quad 1.2707 \quad 1.2332 \quad 1.0839 \quad 0.4706 \quad 0.6705 \quad 0.7489 \right] \; W \cdot m^{-1} \cdot K^{-1}$$

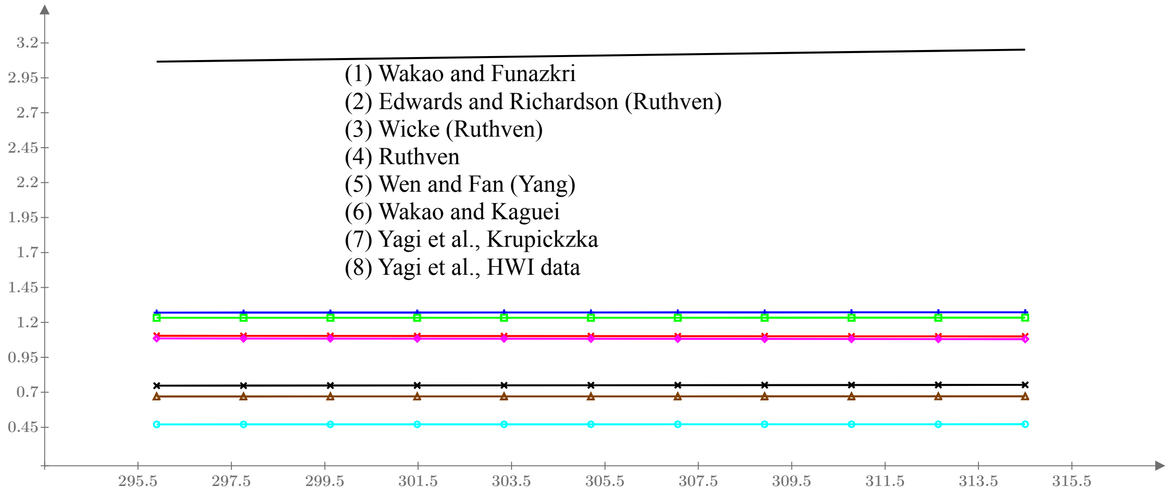


Figure F.7 Effective axial thermal conductivity vs. temperature for eight correlations. Temperatures are in the range of the water vapor on 5A breakthrough test.

Note that the first 5 correlations are based on similarity with mass dispersion and correlations for Peclet number. The sixth is recommended by Wakao and Kaguei 1982. Yagi et al. correlation is compared with test data in Kavinany 1995.

The correlation used in this work is Yagi et al. with Krupickzka. The variation in the coefficient encoutered due to temperature range of the test is:

$YK_i := \left(AGC_T^{\langle 7\rangle}\right)_i$

$AxiCondTempDiff := \max(YK) - min(YK) = 0.0012\ W \cdot m^{-1} \cdot K^{-1}$

$AxiCondTempDiffPer := \dfrac{AxiCondTempDiff}{\text{mean}(YK)} = 0.1823\ 1\%$

The variation between all correlations at all temperatures is:

$HiCorr_i := \max\left(AGC_T^{\langle i\rangle}\right) \qquad LoCorr_i := min\left(AGC_T^{\langle i\rangle}\right)$

$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$

$AxiCondCorrDiff := \max(DiffAll) = 2.6797\ \dfrac{\boldsymbol{W}}{\boldsymbol{m \cdot K}}$

$rownum := \text{match}(\max(DiffAll), DiffAll) = [11] \qquad rownum = [11] \qquad k := rownum_1 = 11$

$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 1.8114\ \dfrac{\boldsymbol{W}}{\boldsymbol{m \cdot K}}$

$AxiCondCorrDiffPer := \dfrac{AxiCondCorrDiff}{2 \cdot DiffMean} = 73.9689\ 1\%$

The variation between the last three correlations (not based on similarity) at all temperatures is:

$AC3^{\langle 1\rangle} := AGC_T^{\langle 6\rangle} \qquad AC3^{\langle 2\rangle} := AGC_T^{\langle 7\rangle} \qquad AC3^{\langle 3\rangle} := AGC_T^{\langle 8\rangle} \qquad k := 1..3$

$HiCorr_i := \max\left(AC3^{\langle i\rangle}\right) \qquad LoCorr_i := min\left(AC3^{\langle i\rangle}\right)$

$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$

$AxiCondCorrDiff := \max(DiffAll) = 0.2817\ \dfrac{\boldsymbol{W}}{\boldsymbol{m \cdot K}}$

$rownum := \text{match}(\max(DiffAll), DiffAll) = [11] \qquad rownum = [11] \qquad k := rownum_1 = 11$

$DiffMean := \text{mean}\left(HiCorr_k, LoCorr_k\right) = 0.6124\ \dfrac{\boldsymbol{W}}{\boldsymbol{m \cdot K}}$

$AxiCondCorrDiffPer := \dfrac{AxiCondCorrDiff}{2 \cdot DiffMean} = 22.9984\ 1\%$

Equivalent Column Diameter

$EqCanID := 2 \cdot \sqrt{\dfrac{FreeFlowArea}{\pi}} = 4.7625\ cm$

Heat Transfer to Column Wall

Correlation of Leva as presented in Ruthven 1984 eq. 7.26 [M. Leva, Chem. Eng. 56, 115 (1949)]

Heat Transfer coefficient at internal surface of the vessel:

$$Nu2DCol_{T_{i,1}} := 0.813 \cdot \left(ReT_i\right)^{0.19} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)}$$

$$\left(Nu2DCol_T^{\langle 1 \rangle}\right)^{\mathrm{T}} = \begin{bmatrix} 1.2427 & 1.2416 & 1.2404 & 1.2393 & 1.2382 & 1.2371 & 1.236 & 1.2349 & 1.2338 & 1.2327 & 1.2316 \end{bmatrix}$$

Correlations of Rase 1900 as presented in W.C. Yang 2003

9.1.2 Heat Transfer Through Wall - One Dimensional Model - Axial; q = hA(T-Tw) where A is inside surface of cylindrical vessel and Tw = wall temperature of fluid. T of the fluid and bed are assumed identical.

For a cylindrical vessel with spherical packing, Applies for $20 \le Re_p \le 7600$ and $0.05 \le \frac{d_p}{D} \le 0.3$

$$Nu1DCol_{T_{i,1}} := 2.26 \cdot \left(ReT_i\right)^{0.8} \cdot \left(Pr_{T_i}\right)^{0.33} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)} \qquad \frac{EqPelDia}{EqCanID} = 0.0487$$

$$\left(Nu1DCol_T^{\langle 1 \rangle}\right)^{\mathrm{T}} = \begin{bmatrix} 29.1579 & 29.046 & 28.9352 & 28.8257 & 28.7174 & 28.6102 & 28.5042 & 28.3993 & 28.2955 & \ldots \end{bmatrix}$$

For a cylindrical vessel with cylindrical packing, Applies for $20 \le Re_p \le 8000$ and $0.03 \le \frac{d_p}{D} \le 0.2$

$$Nu1DCol_{T_{i,2}} := 1.40 \cdot \left(ReT_i\right)^{0.95} \cdot \left(Pr_{T_i}\right)^{0.33} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)} \qquad \frac{EqPelDia}{EqCanID} = 0.0487$$

$$\left(Nu1DCol_T^{\langle 2 \rangle}\right)^{\mathrm{T}} = \begin{bmatrix} 31.804 & 31.6587 & 31.5151 & 31.3732 & 31.2329 & 31.0943 & 30.9572 & 30.8217 & 30.6877 & \ldots \end{bmatrix}$$

Correlations of Li and Finlayson as presented in Wakao and Kaguei 1982 for Overall Heat Transfer Coefficient (1-D models) to Column Wall for Spherical and Cylindrical Pellets

Eq. 5.64 for a cylindrical vessel with spherical packing. Applies for $20 \le Re_p \le 7600$ and $0.05 \le \frac{d_p}{D} \le 0.3$

$$Nu1DCol_{T_{i,3}} := 2.03 \cdot \left(ReT_i\right)^{0.8} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)}$$

$$\frac{EqPelDia}{EqCanID} = 0.0487$$

$$\left(Nu1DCol_T^{\langle 3\rangle}\right)^{T} = [\,30.9735 \quad 30.8527 \quad 30.7333 \quad 30.6153 \quad 30.4986 \quad 30.3833 \quad 30.2693 \quad 30.1565 \quad 30.045 \quad \ldots\,]$$

Eq. 5.65 or a cylindrical vessel with cylindrical packing. Applies for $20 \le Re_p \le 800$ and $0.03 \le \frac{d_p}{D} \le 0.2$

$$Nu1DCol_{T_{i,4}} := 1.26 \cdot \left(ReT_i\right)^{0.95} \cdot e^{\left(-6 \cdot \frac{EqPelDia}{EqCanID}\right)}$$

$$EqPelDia = 0.0023\ \boldsymbol{m}$$

$$PelletDia = 0.0023\ \boldsymbol{m}$$

$$\left(Nu1DCol_T^{\langle 4\rangle}\right)^{T} = [\,33.8509 \quad 33.6942 \quad 33.5394 \quad 33.3865 \quad 33.2355 \quad 33.0863 \quad 32.9389 \quad 32.7933 \quad 32.6493 \quad \ldots\,]$$

$jj := 1 .. 4$

$$GasCanH_{1DT_{i,jj}} := \frac{GasCond_{T_i}}{EqCanID} \cdot Nu1DCol_{T_{i,jj}}$$

$$\left(\left(GasCanH_{1DT}{}^{T}\right)^{\langle 4\rangle}\right)^{T} = [\,18.7731 \quad 20.4321 \quad 19.9385 \quad 21.7434\,]\ \frac{W}{m^2 \cdot K}$$

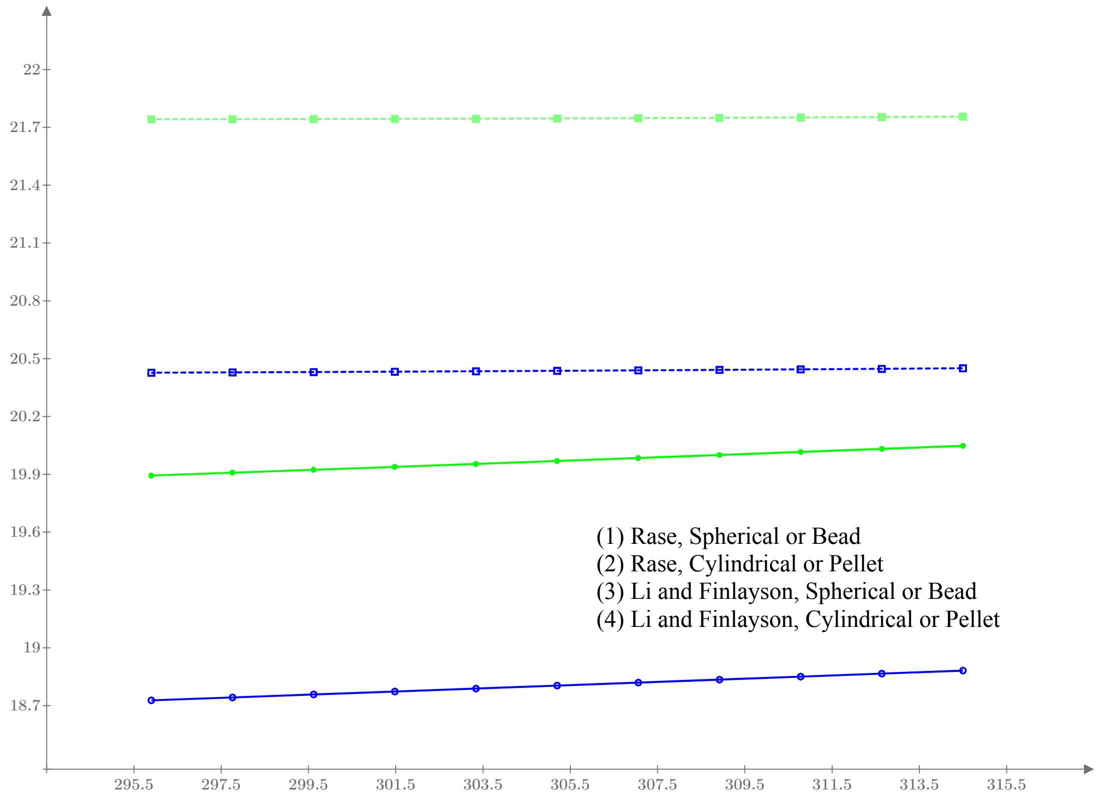


Figure F.8 Overall 1-D heat transfer coefficient to column wall vs. temperature for spherical and cylindrical pellets. Temperatures are in the range of the water vapor on 5A breakthrough test.

The correlation used in this work is Li and Finlayson for a bead. The variation in the coefficient encoutered due to temperature range of the test is:

$$LF_i := \left(GasCanH_{1DT}^{\langle 3\rangle}\right)_i$$

$$GasCanHTempDiff := \max\left(LF\right) - min\left(LF\right) = 0.1542\ \mathbf{W} \cdot \mathbf{m}^{-2} \cdot \mathbf{K}^{-1}$$

$$GasCanHTempDiffPer := \frac{GasCanHTempDiff}{\text{mean}\left(LF\right)} = 0.7723\ 1\%$$

The variation between the correlations for beads at all temperatures is:

$$GasCanH_{1DTB}^{\langle 1\rangle} := GasCanH_{1DT}^{\langle 1\rangle} \qquad GasCanH_{1DTB}^{\langle 2\rangle} := GasCanH_{1DT}^{\langle 3\rangle}$$

$$HiCorr_i := \max\left(GasCanH_{1DTB}^{\overset{\frown}{\underset{\smile}{i}}}\right)\ LoCorr_i := min\left(GasCanH_{1DTB}^{\overset{\frown}{\underset{\smile}{i}}}\right)$$

$$DiffAll_i := \left(HiCorr_i - LoCorr_i\right)$$

$$GasCanHTempDiff := \max\left(DiffAll\right) = 1.1661\ \frac{\mathbf{W}}{\mathbf{m}^2 \cdot \mathbf{K}}$$

$rownum := \text{match}\left(\max\left(DiffAll\right), DiffAll\right) = \left[\,1\,\right]$ $\quad rownum = \left[\,1\,\right]$ $\quad k := rownum_{1} = 1$

$DiffMean := \text{mean}\left(HiCorr_{k}, LoCorr_{k}\right) = 19.3103\ \frac{W}{m^{2} \cdot K}$

$GasCanHTempDiffPer := \frac{GasCanHTempDiff}{2 \cdot DiffMean} = 3.0193\ 1\%$

REFERENCES

Ahn, H.; Lee, C. H. Adsorption dynamics of water in layered bed for air-drying TSA process. *AlChE J.* **2003,** *49* (6), 1601-1609.

Ahn, H.; Lee, C.-H. Effects of capillary condensation on adsorption and thermal desorption dynamics of water in zeolite 13X and layered beds. *Chem. Eng. Sci.* **2004,** *59* (13), 2727-2743.

AiResearch Los Angeles Division. *System and Subsystem Analyses and Analytical Models*; Allied-Signal Aerospace Company: Los Angeles, November 9, 1992.

Airgas. *Specialty Gases and Equipment Product Reference Guide*. Air Liquide: 2015.

Alefeld, G.; Bauer, H.; Maier-Laxhuber, P.; Rothmeyer, M. A zeolite heat pump, heat transformer and heat accumulator, In *Proceedings of the International Conference on Energy Storage*, 1981.

Arévalo-Hidalgo, A. G.; Santana, J. A.; Fu, R.; Ishikawa, Y.; Hernández-Maldonado, A. J. Separation of CO2 from light gas mixtures using nanoporous silicoaluminophosphate sorbents: Effect of multiple-step ion exchange and adsorption mechanism via computational studies. *Microporous Mesoporous Mater.* **2010,** *130* (1-3), 142-153.

Aviles, B. E.; Levan, M. D. Network Models for Nonuniform Flow and Adsorption in Fixed-Beds. *Chem. Eng. Sci.* **1991,** *46* (8), 1935-1944.

Baker, F. S.; Miller, C. E.; Repik, A. J.; Tolles, E. D. Activated Carbon. In *Encyclopedia of Separation Technology*, Ruthven, D. M., Ed. John Wiley & Sons: New York, 1997; Vol. *1*, pp 72-93.

Barwick, V. J. Sources of uncertainty in gas chromatography and high-performance liquid chromatography. *J. Chromatogr. A* **1999,** *849*, 20.

Beeyani, A. K.; Singh, K.; Vyas, R. K.; Kumar, S.; Kumar, S. Parametric studies and simulation of PSA process for oxygen production from air. *Pol. J. Chem. Technol.* **2010,** *12* (2), 18-28.

Benin, A. I.; Jakubczak, P.; Low, J. J.; Faheem, S. A.; Venimadhavan, G.; Yazaydin, O.; Snurr, R. Q.; Wong-Foy, A.; Koh, K.; Matzger, A. J.; Liu, J.; LeVan, M. D.; Hu, X. E.; Brandani, S.; Willis, R. R. Bench-Scale Development & Testing of a Novel

Adsorption Process for Post-Combustion CO2 Capture, In *2012 NETL CO2 Capture Technology Meeting*, Pittsburgh, 2012.

Bird, R. B.; Stewart, W. E.; Lightfoot, E. N. *Transport Phenomena*; John Wiley & Sons, New York: 2002.

Brosillon, S.; Manero, M.-H.; Foussard, J.-N. Mass transfer in VOC adsorption on zeolite: experimental and theoretical breakthrough curves. *Environ. Sci. Technol.* **2001,** *35* (17), 3571-3575.

Brunauer, S.; Deming, L. S.; Deming, W. E.; Teller, E. On a theory of the van der Waals adsorption of gases. *J. Am. Chem. Soc.* **1940,** *62* (7), 1723-1732.

Chao, C. C. Process for separating nitrogen from mixtures thereof with less polar substances. U.S. Patent 4859217, 1989.

Chiang, R. L.; Whitley, R. D.; Ostroski, J. E.; Dee, D. P. Argon/oxygen selective X-zeolite. U.S. Patent 6432170, 2002.

Chou, C. T.; Chen, C. Y. Carbon dioxide recovery by vacuum swing adsorption. *Sep. Purif. Technol.* **2004,** *39* (1-2), 51-65.

Coker, R. F.; Knox, J. C.; Schunk, G.; Gomez, C. Computer Simulation and Modeling of CO2 Removal Systems for Exploration, In *45th International Conference on Environmental Systems*, SAE: Bellevue, Washington, 2015.

Coleman, H. W.; Steele, W. G.; Coleman, H. W. *Experimentation, validation, and uncertainty analysis for engineers*; 3rd ed.; John Wiley & Sons: Hoboken, N.J., 2009; p xvi, 317 p.

Coppola, A. P.; Levan, M. D. Adsorption with Axial Diffusion in Deep Beds. *Chem. Eng. Sci.* **1981,** *36* (6), 967-971.

Coppola, A. P.; Levan, M. D. Adsorption with Axial Diffusion in Shallow Beds. *Chem. Eng. Sci.* **1983,** *38* (7), 991-997.

Croan, A. *Calibration Certificates 3219437, 321940, 321943, 321949, 321953 for Omega Thermocouple, Type T*; MSFC Metrology & Calibration Laboratory: Huntsville, Alabama, 2015.

Danckwerts, P. V. Continuous flow systems. Distribution of residence times. *Chem. Eng. Sci.* **1953,** *50* (24), 3857-3866.

Delgado, J. A.; Uguina, M. A.; Sotelo, J. L.; Ruíz, B. Fixed-bed adsorption of carbon dioxide–helium, nitrogen–helium and carbon dioxide–nitrogen mixtures onto silicalite pellets. *Sep. Purif. Technol.* **2006,** *49* (1), 91-100.

Drolc, A.; Djinović, P.; Pintar, A. Gas chromatography analysis: method validation and measurement uncertainty evaluation for volume fraction measurements of gases in simulated reformate gas stream. *Accreditation and Quality Assurance* **2013,** *18* (3), 225-233.

Dwivedi, P. N.; Upadhyay, S. N. Particle-Fluid Mass Transfer in Fixed and Fluidized Beds. *Ind. Eng. Chem. Process Des. Dev.* **1977,** *16* (2), 157-165.

Edwards, M. F.; Richardson, J. F. Gas dispersion in packed beds. *Chem. Eng. Sci.* **1968,** *23* (2), 109-123.

Elliott, J.; Srinivas, G.; Copeland, R. Low Cost Sorbent for CO2 Capture on Existing Plants, In *2012 NETL CO2 Capture Technology Meeting*, Pittsburgh, 2012.

Farooq, S.; Ruthven, D. M. Heat effects in adsorption column dynamics. 1. Comparison of one- and two-dimensional models. *Ind. Eng. Chem. Res.* **1990,** *29* (6), 1076-1084.

Fluke Corporation. 5610, 5611, 5611T, 5665 Temperature Probes, Reference Thermistor Probes. http://us.flukecal.com/products/process-calibration-tools/temperature-calibrators/thermistors/5610-5611-5611t-5665-temperat (accessed October 25, 2015).

Folger, P. *Funding for Carbon Capture and Sequestration (CSS) at DOE: In Brief*; 2016.

Geib, R. C. *Calibration Standard Gases Are Key to Reliable Measurements*; AR-28; Matheson Tri-Gas: 2005.

Glueckauf, E. Theory of chromatography. Part 10.—Formulæ for diffusion into spheres and their application to chromatography. *Trans. Faraday Soc.* **1955,** *51*, 1540-1551.

González, A. S.; Plaza, M. G.; Rubiera, F.; Pevida, C. Sustainable biomass-based carbon adsorbents for post-combustion CO2 capture. *Chem. Eng. J.* **2013,** *230*, 456-465.

Gorbach, A.; Stegmaier, M.; Eigenberger, G. Measurement and Modeling of Water Vapor Adsorption on Zeolite 4A—Equilibria and Kinetics. *Adsorption* **2004,** *10* (1), 29-46.

GRACE Davison. *Adsorbents for Process Applications*. W. R. Grace & Co.: Germany, 2010.

Haghpanah, R.; Rajendran, A.; Farooq, S. Do Performance Indicators Predict Adsorption Process Performance? the Case of Post-Combustion CO2 Capture, In *American Inst. of Chemical Engineers (AIChE) Annual Meeting*, Atlanta, 2014.

Hardy, B. In *ITS-90 formulations for vapor pressure, frostpoint temperature, dewpoint temperature, and enhancement factors in the range–100 to+ 100 C*, Proceedings of the Third International Symposium on Humidity and Moisture, Teddington, London, England, 1998.

Hopson, G. D.; Littles, J. W.; Patterson, W. C. Skylab Environmental Control and Life Support System, In *Life Support and Environmental Control Conference*, ASME: San Francisco, 1971.

Hougen, O. A.; Watson, K. M. *Chemical Process Principles* **1947,** *Vol. III.*

Jain, R. Bench-scale Development and Testing of a Novel Adsorption Process for Post-Combustion CO2 Capture, In *2012 NETL CO2 Capture Technology Meeting*, Pittsburgh, 2012.

James, J. T. *Spacecraft Maximum Allowable Concentrations for Airborne Contaminants*; JSC 20584; Houston, Texas, 2008.

James, J. T.; Macatangay, A. Carbon Dioxide-Our Common "Enemy", In *SAMAP Submarine Air Monitoring Air Purification Conference*, NASA Johnson Space Center: San Diego, CA, 2009.

James, J. T.; Meyers, V. E.; Sipes, W.; Scully, R. R.; Matty, C. M. Crew health and performance improvements with reduced carbon dioxide levels and the resource impact to accomplish those reductions, In *41st International Conference on Environmental Systems*, Portland, Oregon, 2011.

James, J. T. Surprising Effects of CO2 Exposure on Decision Making, In *International Conference on Environmental Systems*, AIAA: Vail, CO, 2013a.

James, J. T. *Provisional SMACs for CO2*. Davis, J., Ed. Houston, TX, 2013b.

Kärger, J.; Ruthven, D. *Diffusion in zeolites and other microporous materials*; John Wiley and Sons: New York, 1992.

Kaviany, M. *Principles of heat transfer in porous media*; 2nd ed.; Springer-Verlag: New York, 1995; p xxii, 708 p.

Keller, G. E. Gas-Adsorption Processes: State of the Art. In *Industrial Gas Separations*, Whyte, T. E.; Yon, C. M.; Wagener, E. H., Eds. American Chemical Society: Washington, D.C., 1983; pp 145-170.

King, C. J. *Separation Processes*; Second ed.; 1980.

Knox, J. C. *Test Requirements for the Characterization of Adsorption Materials Used in the Carbon Dioxide Removal Assembly*; Memorandum ED62(18-92); NASA: Marshall Space Flight Center, 1992a.

Knox, J. C. Finite Difference Modeling and Experimental Investigation of Carbon Dioxide Adsorption on a Molecular Sieve Sorbent Material Used in Spacecraft Carbon Dioxide Removal Systems. M.S. Thesis, University of Alabama, Huntsville, AL, 1992b.

Knox, J. C. International Space Station CO2 Removal Assembly Testing, In *International Conference on Environmental Systems*, SAE: Toulouse, France, 2000.

Knox, J. C.; Coker, R. F.; Huff, T.; Gatens, R.; Miller, L. A. Development of Carbon Dioxide Removal Systems for Advanced Exploration Systems 2014-2015, In *45th International Conference on Environmental Systems*, Bellevue, Washington, 2015a.

Knox, J. C.; Gauto, H.; Miller, L. A. Development of a Test for Evaluation of the Hydrothermal Stability of Sorbents used in Closed-Loop CO2 Removal Systems, In *International Conference on Environmental Systems*, Bellevue, Washington, 2015b.

Knox, J. C.; Ebner, A. D.; LeVan, M. D.; Coker, R. F.; Ritter, J. A. Limitations of Breakthrough Curve Analysis in Fixed-Bed Adsorption. *Ind. Eng. Chem. Res.* **2016a**.

Knox, J. C.; Cmarik, G.; Watson, D.; Wingard, C. D.; West, P.; Miller, L. A. Investigation of Desiccants and CO2 Sorbents for Advanced Exploration Systems 2015-2016, In *46th International Conference on Environmental Systems*, Vienna, 2016b.

Ko, D.; Kim, M.; Moon, I.; Choi, D.-k. Analysis of purge gas temperature in cyclic TSA process. *Chem. Eng. Sci.* **2002,** *57* (1), 179-195.

Kuhl, G. H.; Kresge, C. T. Molecular Sieves. In *Encyclopedia of Separation Technology*, Ruthven, D. M., Ed. John Wiley & Sons: New York, 1997; Vol. *2*, pp 1339-1369.

Kupiec, K.; Rakoczy, J.; Komorowicz, T.; Larwa, B. Heat and mass transfer in adsorption–desorption cyclic process for ethanol dehydration. *Chem. Eng. J.* **2014,** *241*, 485-494.

Leinekugel-le-Cocq, D.; Tayakout-Fayolle, M.; Le Gorrec, Y.; Jallut, C. A double linear driving force approximation for non-isothermal mass transfer modeling through bi-disperse adsorbents. *Chem. Eng. Sci.* **2007,** *62* (15), 4040-4053.

LeVan, D. M.; Carta, G. Adsorption and Ion Exchange. In *Perry's Chemical Engineer's Handbook*, Eighth Edition ed.; McGraw-Hill: New York, 2008.

Li, C.; Finlayson, B. Heat transfer in packed beds—a re-evaluation. *Chem. Eng. Sci.* **1977,** *32*, pp. 1055–1066.

Lide, D. R.; Frederikse, H. P. R. *CRC Handbook of Chemistry and Physics*. 74 ed.; Boca Raton, 1993.

Liptak, B. G. *Instrument Engineers' Handbook*. Fourth ed.; CRC Press: Boca Raton, 2003; Vol. One.

Lopes, F. V. S.; Grande, C. A.; Ribeiro, A. M.; Loureiro, J. M.; Evaggelos, O.; Nikolakis, V.; Rodrigues, A. E. Adsorption of H2, CO2, CH4, CO, N2 and H2O in Activated Carbon and Zeolite for Hydrogen Production. *Sep. Sci. Technol.* **2009,** *44* (5), 1045-1073.

Lu, Y. Z.; Wang, R. Z.; Zhang, M.; Jiangzhou, S. Adsorption cold storage system with zeolite–water working pair used for locomotive air conditioning. *Energy Convers. Manage.* **2003,** *44* (10), 1733-1743.

Lucas, K. Phase equilibria and fluid properties in the chemical industry. *Dechema, Frankfurt* **1980,** *100*.

Mahle, J. J.; Friday, D. K. In *Axial Dispersion Effects on the Breakthrough Behavior of Favorably Adsorbed Vapors*, Recents Progres en Genie Des Procedes, France, Meunier, F.; LeVan, M. D., Eds. Groupe Francais de Genie des Procedes: France, 1991.

Matheson. *Gases and Equipment*. Matheson Tri-Gas: USA, 2007.

Mette, B.; Kerskes, H.; Drück, H.; Müller-Steinhagen, H. Experimental and numerical investigations on the water vapor adsorption isotherms and kinetics of binderless zeolite 13X. *Int. J. Heat Mass Transfer* **2014,** *71*, 555-561.

Miller, D. C.; Litynski, J. T.; Brickett, L. A.; Morreale, B. D. Toward transformational carbon capture systems. *AlChE J.* **2016,** *62* (1), 2-10.

Milton, R. M. Molecular sieve adsorbents. U.S. Patent 2882243, 1959.

Moffat, R. J. Describing the uncertainties in experimental results. *Exp. Therm. Fluid Sci.* **1988,** *1* (1), 3-17.

Mohamadinejad, H. The Adsorption of CO2/H2O/N2 on 5A Zeolite in a Packed Column in One and Two-Dimensional Flows. Ph.D Dissertation, The University of Alabama, Huntsville, AL, 1999.

Mohamadinejad, H.; DaLee, R. C.; Blackmon, J. B. *Molecular Sieve Bench Testing and Computer Modeling*; MDC-95W5104; NAS 1.26:199834; NASA-CR-199834; Huntsville, AL, 1995; p 322.

Mohamadinejad, H.; Knox, J.; Smith, J. Experimental and numerical investigation of adsorption/desorption in packed sorption beds under ideal and nonideal flows. *Sep. Sci. Technol.* **2000,** *35* (1), 1-22.

Mohamadinejad, H.; Knox, J. C.; Smith, J. E. Experimental and numerical investigation of two-dimensional CO2 adsorption/desorption in packed sorption beds under non-ideal flows. *Sep. Sci. Technol.* **2003,** *38* (16), 3875-3904.

Molerus, O.; Wirth, K.-E. *Heat transfer in fluidized beds*; Springer Science & Business Media: 2012; Vol. *11*.

Mulgundmath, V. P.; Jones, R. A.; Tezel, F. H.; Thibault, J. Fixed bed adsorption for the removal of carbon dioxide from nitrogen: Breakthrough behaviour and modelling for heat and mass transfer. *Sep. Purif. Technol.* **2012,** *85*, 17-27.

Nakos, J. T. *Uncertainty Analysis of Thermocouple Measurements Used in Normal and Abnormal Thermal Environment Experiments at Sandia's Radiant Heat Facility and Lurance Canyon Burn Site*; SAND2004-1023; Sandia National Laboratories: Albuquerque, New Mexico, 2004.

NASA Headquarters. *NASA's Journey to Mars*; NP-2015-08-2018-HQ; National Aeronautics and Space Administration: Washington D.C., 2015.

Nastaj, J.; Ambrozek, B. Analysis of gas dehydration in TSA system with multi-layered bed of solid adsorbents. *Chem. Eng. Process.* **2015,** *96*, 44-53.

National Energy Technology Laboratory. *Carbon Capture Technology Program Plan*; U.S. Department of Energy: Washington, DC, 2013.

Nield, D. A.; Bejan, A. *Convection in Porous Media*; Springer-Verlag: New York, 1992.

OMEGA. *Complete Temperature Measurement Handbook & Encyclopedia*. 7th ed.; OMEGA Engineering: Stamford, CT, 2010; Vol. MMXIV.

Optica. *Optica™ The Next Generation Humidity Reference Standard*. In *Sensing & Inspection Technologies*, General Electric: 2008.

Park, I.; Knaebel, K. S. Adsorption breakthrough behavior: Unusual effects and possible causes. *AlChE J.* **1992,** *38* (5), 660-670.

Park, J.-H.; Kim, J.-N.; Cho, S.-H. Performance Analysis of Four-Bed H2 PSA Process Using Layered Beds. *AlChE J.* **2000,** *46* (4).

Park, J.-H.; Kim, J.-N.; Cho, S.-H.; Kim, J.-D.; Yang, R. T. Adsorber dynamics and optimal design of layered beds for multicomponent gas adsorption. *Chem. Eng. Sci.* **1998,** *53* (23), 3951-3963.

Parker, J. F.; West, V. R.; Webb, P. *Bioastronautics data book*; Scientific and Technical Information Office, National Aeronautics and Space Administration: Washington, 1973.

Parsons, R. *ASHRAE handbook-fundamentals*; 1997.

Pentchev, I.; Paev, K.; Seikova, I. Dynamics of non-isothermal adsorption in packed bed of biporous zeolites. *Chem. Eng. J.* **2002,** *85* (2–3), 245-257.

Petrovic, L. J.; Thodos, G. Mass transfer in flow of gases through packed beds. Low Reynolds number region. *Ind. Eng. Chem. Fundam.* **1968,** *7* (2), 274-280.

Poling, B. E.; Prausnitz, J. M.; O'Connell, J. P. *The Properties of Gases and Liquids*; 5th ed.; McGraw-Hill: New York, 2001.

Porter, I. D. *Model 251 Mass Flow Instruments*. Parker Hannifin Corporation: Hatfield, PA, 2011.

Puértolas, B.; Navarro, M. V.; Lopez, J. M.; Murillo, R.; Mastral, A. M.; Garcia, T. Modelling the heat and mass transfers of propane onto a ZSM-5 zeolite. *Sep. Purif. Technol.* **2012,** *86*, 127-136.

Raatschen, W.; Schauer, L.; Mesnage, F. In *Differences and commonness between submarine and spacecraft life support systems*, Submarine Air Monitoring Air Purification Conference, San Diego CA: 2009.

Radenburg, E. *Particle Size Analysis*; National Aeronautics and Space Administration: Marshall Space Flight Center, 2013.

Reid, R. C.; Prausnitz, J. M.; Poling, B. E. *The Properties of Gases and Liquids*; 4th ed.; McGraw-Hill: 1987.

Richardson, J. F.; Harker, J. H.; Backhurst, J. R. *Particle Technology and Separation Processes*. In *Coulson and Richardson's Chemical Engineering*, 5th ed.; Coulson, J. M.; Richardson, J. F., Eds. Butterworth-Heinemann: Oxford, 2002; Vol. 2, p 1229.

Russell, B. P.; LeVan, M. D. Nonlinear adsorption and hydrodynamic dispersion in self-similar networks. *Chem. Eng. Sci.* **1997,** *52* (9), 1501-1510.

Ruthven, D. M. *Principles of Adsorption and Adsorption Processes*; 1984; p 433.

Ruthven, D. M.; Farooq, S.; Knaebel, K. S. *Pressure Swing Adsorption*; John Wiley & Sons: 1994.

Saha, B.; Koyama, S.; Lee, J. e. a.; Kuwahara, K.; Alam, K.; Hamamoto, Y.; Akisawa, A.; Kashiwagi, T. Performance evaluation of a low-temperature waste heat driven multi-bed adsorption chiller. *Int. J. Multiphase Flow* **2003,** *29* (8), 1249-1263.

Shafeeyan, M. S.; Daud, W. M. A. W.; Shamiri, A. A review of mathematical modeling of fixed-bed columns for carbon dioxide adsorption. *Chem. Eng. Res. Des.* **2014,** *92* (5), 961-988.

Sircar, S.; Hufton, J. Why does the linear driving force model for adsorption kinetics work? *Adsorption* **2000,** *6* (2), 137-147.

Slattery, J. C.; Bird, R. B. *AlChE J.* **1958,** *4*.

Sutan, A.; Daniel, P. Gas chromatography maintenance using uncertainty based CBM, In *The Americas Flow Measurement Conference*, 2014.

Suzuki, M. *Adsorption engineering*; Kodansha: Tokyo, 1990; Vol. *551*, p 128.

Toth, J. State equations of the solid-gas interface layers. *Acta Chim. Acad. Sci. Hung.* **1971,** *69* (3), 311-328.

Viatran. *Pressure Transmitter Models 118/218/318*. Dynisco Company: Wheatfield, NY, 2010.

Vora, S.; Bricket, L.; Indrikanti, P.; Munson, R.; Murphy, J.; Rife, T.; Strock, J.; Zaremsky, C. *DOE/NETL Advanced Carbon Dioxide Capture R&D Program: Technology Update*; United States: 2013.

Wakao, N.; Funazkri, T. Effect of fluid dispersion coefficients on particle-to-fluid mass transfer coefficients in packed beds : Correlation of sherwood numbers. *Chem. Eng. Sci.* **1978,** *33* (10), 1375-1384.

Wakao, N.; Kaguei, S. *Heat Transfer in Packed Beds*; 1982.

Wakao, N.; Kaguei, S.; Nagai, H. Effective diffusion coefficients for fluid species reacting with first order kinetics in packed bed reactors and discussion on evaluation of catalyst effectiveness factors. *Chem. Eng. Sci.* **1978,** *33* (2), 183-187.

Waligora, J.; Horrigan, D.; Nicogossian, A. The physiology of spacecraft and space suit atmosphere selection. *Acta Astronaut.* **1991,** *23*, 171-177.

Wang, Y.; LeVan, M. D. Adsorption Equilibrium of Carbon Dioxide and Water Vapor on Zeolites 5A and 13X and Silica Gel: Pure Components. *J. Chem. Eng. Data* **2009,** *54* (10), 2839-2844.

Watson, D.; Knox, J. C.; West, P.; Stanley, C.; Bush, R. Sorbent Structural Impacts due to Humidity on Carbon Dioxide Removal Sorbents for Advanced Exploration Systems, In *45th International Conference on Environmental Systems*, Bellevue, Washington, 2015.

Wen, C. Y.; Fan, L. T. *Models for flow systems and chemical reactors*; Marcel Dekker: New York, 1975; Vol. *20*, p 570 pages.

Whitley, R. D. Pressure swing adsorption process with improved recovery of high-purity product. U.S. Patent 7651549, January 26, 2010.

Wicke, E. *Physical Chemistry* **1973,** *77* (160).

Winterberg, M.; Tsotsas, E. Correlations for effective heat transport coefficients in beds packed with cylindrical particles. *Chem. Eng. Sci.* **2000,** *55* (23), 5937-5943.

Won, W.; Lee, S.; Lee, K. S. Modeling and parameter estimation for a fixed-bed adsorption process for CO2 capture using zeolite 13X. *Sep. Purif. Technol.* **2012,** *85*, 120-129.

Yagi, S.; Kunii, D.; Wakao, N. Studies on axial effective thermal conductivities in packed beds. *AlChE J.* **1960,** *6* (4), 543-546.

Yang, J.; Chang-Ha, L. Adsorption dynamics of a layered bed PSA for H2 recovery from coke oven gas. *AlChE J.* **1998,** *44* (6), 1325.

Yang, R. T. *Gas Separation by Adsorption Processes*; Imperial College Press: London, 1997; Vol. *1*.

Yang, R. T. *Adsorbents: Fundamentals and Applications*; John Wiley and Sons: 2003.

Yang, W-C. *Handbook of fluidization and fluid-particle systems*; CRC press: 2003.

Yuan, Z.; Eden, M. R.; Gani, R. Toward the Development and Deployment of Large-Scale Carbon Dioxide Capture and Conversion Processes. *Ind. Eng. Chem. Res.* **2015**.