Planetary Physics Challenge Problems

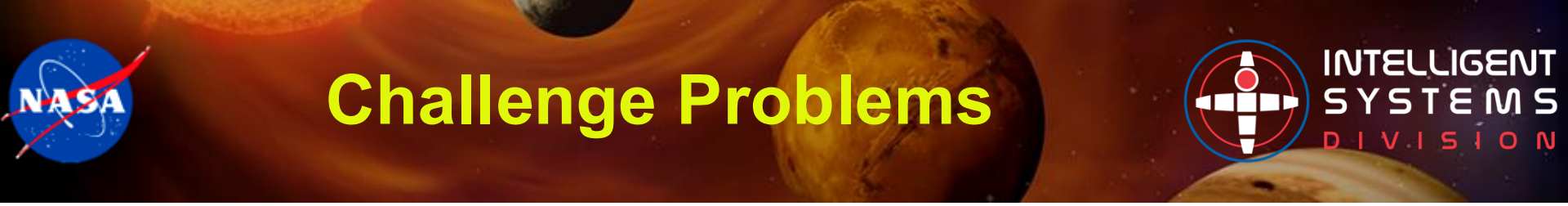
Shon Grabbe (NASA ARC)
Zoe Gonzalez-Izquierdo (KBR, Inc.)
Erik Gustafson (KBR, Inc.)

Super Intelligence (SI) Usage Disclosure

This document was created with assistance from NASA's subscription to Office 365 Copilot and Gemini for background research, problem specification and identification. The content has been reviewed and edited by all authors. More information on the extent and nature of SI usage is included in the Acknowledgements sections of the document

NASA

INTELLIGENT SYSTEMS DIVISION



Challenge Problems

1. Mars Weather Forecasting

2. Retrieval Modelling: Earth Like Exoplanets

3. Retrieval Modelling: Hot Jupiter Exoplanets



Mars Weather Forecasting

Mars Weather Global Forecasting

Application Details

- Accurate and faster modelling of GCM supporting Observing system simulation experiments (OSSE)
- Human and Robotic Exploration of Mars
- Moon 2 Mars Missions

Problem Details

- Forecasting Mars Weather
- Solving complex Navier stokes equations and correlated Kalman filters.
- Input: Model Descriptors
- Output: Observables

Current Approaches

- State-of-the-art methods use Local Ensemble Transform Kalman Filters (LETKF) and OSSE methods
- LETKF breaks when underlying assumptions are violated (common during dust storms)
- OSSE breaks when the underlying methods lack realism

Impact of New/Better Solution

- A better solution would have accurate forecasting minimizing.
- Better forecasting would improve the current precision of mars landings from <7km to <100m necessary for crewed missions.
- Impacts for NOAA and Airforce using LIDAR measurements and OSSE techniques

Problem Summary

Objective

- Solving high-dimensional atmospheric data-assimilation (DA) and Observing System Simulation Experiment (OSSE) problems for Mars:
  - Obtaining global profiles of line-of-sight winds, dust aerosol backscatter, cross-polarized backscatter for water-ice aerosols, and surface range from orbit.
- Improving weather forecasting on Mars, leading to safer future Mars missions.

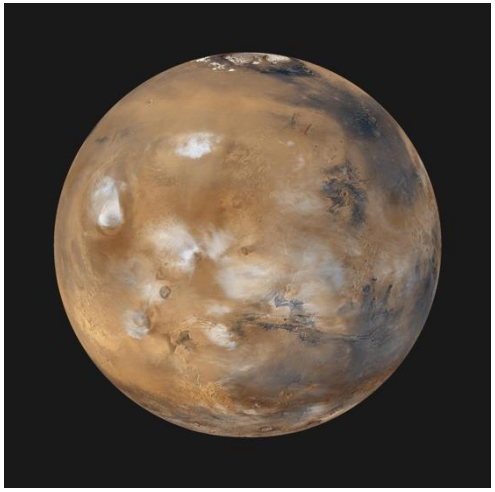


Image Credit: NASA/JPL/MSSS

Computational problem

- Combining a Mars Global Climate (MGC) model forecast ensemble with observations (real or simulated) to estimate the evolving 3D atmospheric state and quantify the quality of a proposed observing system.
- Tackled through an iterative approach that cycles through two phases:
  - Evolve an initial ensemble of states through a global physics engine (which solves the PDEs governing the state of the atmosphere).
  - Perform a Bayesian update of the ensemble by finding the linear combination of its members that minimizes the error between forecast and observation.

Problem Summary

Background

- Observing System Simulation Experiment (OSSE) are important concepts for mission. It's a simulation done with existing sensors plus fictive ones to assess new impact and progress and identify the minimal number of new sensors to adequately reach analysis accuracy requirements.

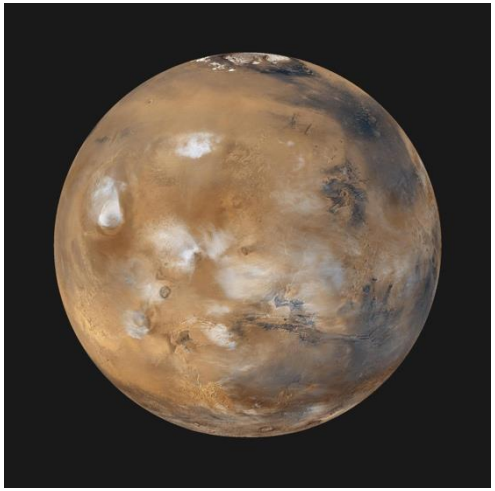


Image Credit: NASA/JPL/MSSS

Benefits beyond Mars Climate Modelling

- Similar studies and developments could be used for other planetary missions such as robotic ones to Titan or Venus.
- Additionally these developments would have impacts for earth observations including launch windows for Earth, as well as weather forecasting on Earth.

Problem Summary

Background

- Observing System Simulation Experiment (OSSE) are important concepts for mission. It's a simulation done with existing sensors plus fictive ones to assess new impact and progress and identify the minimal number of new sensors to adequately reach analysis accuracy requirements.

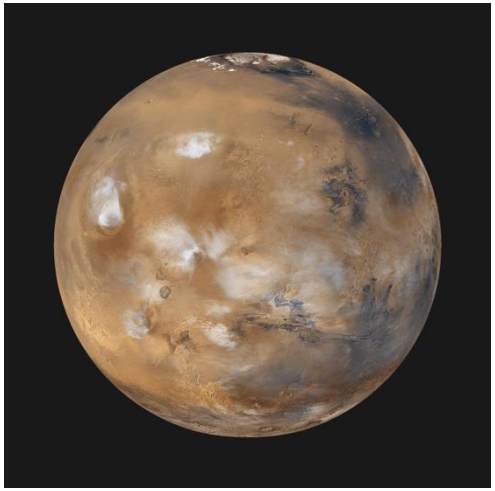


Image Credit: NASA/JPL/MSSS

Benefits beyond Mars Climate Modelling

- Similar studies and developments could be used for other planetary missions such as robotic ones to Titan or Venus.
- Additionally these developments would have impacts for earth observations including launch windows for Earth, as well as weather forecasting on Earth.

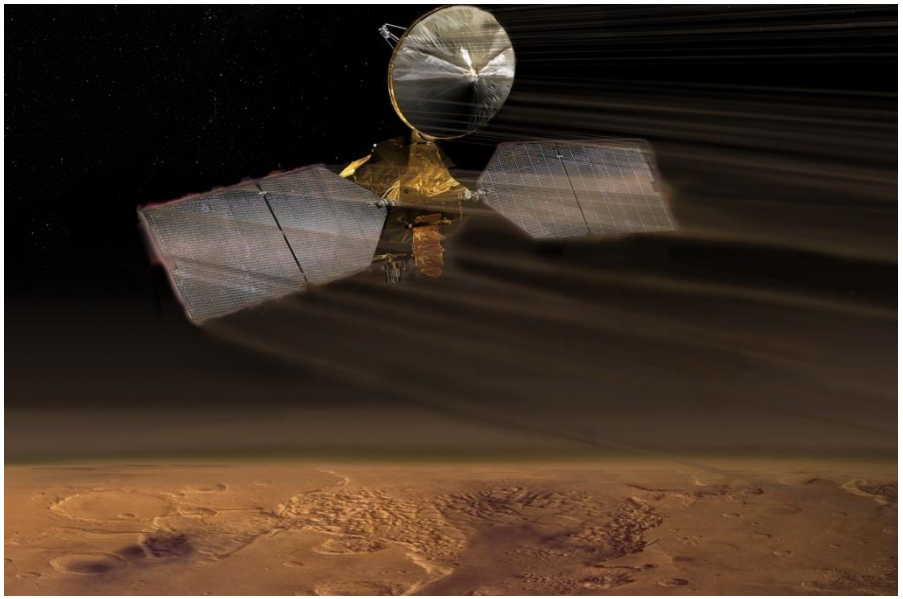
Application : Moon 2 Mars

Accurate and fast weather forecasting for Mars is crucially important for future robotic and human Mars missions.

Weather considerations for Moon 2 Mars mission include:

- Dust storms (local and global):[1]
  - Dust can block solar panels, reducing efficiency, and can also damage equipment.
  - Electric discharge can cause chemical reactions introducing percolates which damage organic molecules.

- Temperature fluctuations:
  - Temperatures can range up to 100°C daily, depending on time of day and location (poles vs equator), potentially interfering with robotics and affecting human mission planning.



Mars Reconnaissance Orbiter. Image Credit: NASA/JPL-Caltech

[1] https://www.nasa.gov/missions/mars-2020-perseverance/perseverance-rover/nasa-rover-detects-electric-sparks-in-mars-dust-devils-storms/

Impacts of New / Better Solution

Benefits to NASA and beyond

Benefits to NASA

- Improved forecasting and near optimal estimates of satellite resources via Observing Simulated System Experiments
- Lessons learned can be ported over to other planetary systems such as Venus or Titan
- Lessons learned can be reapplied to Earth observing systems

External Impacts

- Effective global climate models are useful for both NOAA in weather prediction
- Improved simulations for earth models will have down stream impacts DoW AFRL, NRO, ARL in terms of mission deployments

Financial Impacts

- Near optimal estimates of satellite resources for observing systems
- Cost of replacement for astronaut training or robotic systems

State-of-the-Art Approach: LETKF

LETKF: Local Ensemble Transform Kalman Filter

- It is a Bayesian analysis approach to simulating weather systems
- It uses an ensemble of states to represent uncertainty about future behaviors
- Uses a transform-based update (matrix multiplications) that update the ensemble

- LETKF methods are fast because they assume **local neighborhoods, reasonable ensemble size, strictly gaussian noise**. However, these assumptions are not always reasonable.

- Heaviest component of the LETKF is the underlying physics engine that is used to evolve the ensemble states. This physics engine typically is a highly specialized solver for the underlying partial differential equation (e.g. Navier-Stokes)

- LETKF implementation is available at https://github.com/takemasa-miyoshi/letkf

- Seminal paper: Hunt 2007
- State-of-the-art implementation: Greybush 2019

LETKF Steps

1. Generate initial ensemble
   I. Start with generic “best guess” system state vector (e.g., typical atmospheric conditions).
   II. Evolve state through physics engine (can be a standard CFD engine) for a long simulation time. Final state is our base state.
   III. Generate ensemble members, i.e., obtain K (~ 50-100) valid physical states from our base state.
2. Global physics engine
   IV. Feed ensemble to physics engine (Navier-Stokes solving software) and evolve each state independently to the next time-step.
   V. Pause engine at next time where observational data is available. Take the K evolved states at that time.
1. **Local Bayesian math**

   This is the only part of the process that happens at the “local neighborhood” level. **For every cell in our grid**, do the following steps:

- Draw radius around cell: this is our local neighborhood.
- From the K ensemble members, get the values that correspond to cells inside the neighborhood.
- Get observation data from the same area.
- For each member of the ensemble, calculate optimal weights using Bayesian matrix inversion as discussed above.
- For each member of the ensemble, update ONLY the current grid cell using the optimal weights. E.g. update member j using w j,1*member1 +

  w j,2*member2 + … + w j,K*memberK

  Take this updated ensemble, plug back into physics engine, repeat steps 2 and 3.

Math background for LETKF

Consider a chaotic system governed by the differential equation:

$$\frac{d\vec{x}}{dt} = F(x,t),$$

where $\vec{x}(t)$ represents the state of the system at time t. We want to use Bayesian modelling to update the system and estimate the trajectory $\vec{x}(t)$ at later times that best fits the observations from earlier times.

Observed values are denoted by $\vec{y}_j^0$; with the function H_j describing the relationship between the system state $\vec{x}(t_j)$ and those observations. The observations are assumed to have Gaussian error ϵ_j with covariance matrix R_j.

Our goal is to find a trajectory that maximizes the likelihood function:

$$\mathcal{L} = \prod_{j=1}^{n} \exp\left(-\frac{1}{2}\left[\vec{y}_j^0 - H_j\left(\vec{x}(t_j)\right)\right]^T R_j^{-1}\left[\vec{y}_j^0 - H_j\left(\vec{x}(t_j)\right)\right]\right),$$

or equivalently minimizes the associated cost function:

$$J^0(\{\vec{x}(t)\}) = \sum_{j=1}^{n}\left[\vec{y}_j^0 - H_j\left(\vec{x}(t_j)\right)\right]^T R_j^{-1}\left[\vec{y}_j^0 - H_j\left(\vec{x}(t_j)\right)\right].$$

Math background for LETKF

It is more convenient to rewrite the cost function in terms of a map that takes the system state from t to t_j, M_{t,t_j} :

$$J_t^0(\vec{x}) = \sum_{j=1}^{n} \left[\vec{y}_j^0 - H_j\left(M_{t,t_j}(\vec{x})\right)\right]^T R_j^{-1} \left[\vec{y}_j^0 - H_j\left(M_{t,t_j}(\vec{x})\right)\right]$$

If H_j is linear, we use standard Kalman filter. Otherwise, we define an ensemble $\left\{\vec{x}_{n-1}^{a(i)} : i = 1, 2, \ldots, k\right\}$ of model states at time t_{n-1}, whose average represents the best estimate of the system state. The ensemble members are evolved through M_{t,t_j} to the background ensemble $\left\{\vec{x}_n^{b(i)} : i = 1,2, \ldots, k\right\}$ at time t_n.

Math background for LETKF

We parametrize this vector as $\vec{x} = \bar{x}^b + X^b\overrightarrow{w}$, with w being a Gaussian-distributed random variable with mean 0.

Kalman update equations for the parameters are

$$\bar{w}^a = \tilde{P}^a\left(Y^b\right)^T R^{-1}\left(\vec{y}^0 - \bar{y}^b\right)$$

$$\tilde{P}^a = \left[(k-1)\boldsymbol{I} + \left(Y^b\right)^T R^{-1} Y^b\right]^{-1}$$

The update parameters are related to the model parameters via

$$\bar{x}^a = \bar{x}^b + X^b\bar{w}^a$$

$$P^a = X^b\tilde{P}^a\left(X^b\right)^T$$

A simulation over a set of ensemble equations are done to generate background state denoted by the superscript b, and evolved forward using Bayesian analysis and the update filter equations to drive the system forward to an analysis state

Computational Complexity of LETKF

- LETKF was developed to improve upon the exponential scaling of other DA methods.
- The time complexity of each iterative update includes:
  - Inversion of symmetric matrix: $\mathcal{O}(K^3)$; K = size of ensemble.
  - Covariance matrix construction: $\mathcal{O}(\ell K^2)$; ℓ = # observations for local neighborhood.
  - Calculating weight perturbations: $\mathcal{O}(K^3)$.
  - Performing update: $\mathcal{O}(K)$.
- These steps are performed for each of the N grid cells.

Total time complexity:
$$\mathcal{O}(N(\ell K^2 + K^3))$$

- This reflects both worst-case scenario and typical complexity, given that it directly comes from deterministic linear algebra with known dimensions.
- Since we address each grid point separately, memory is no issue. Parallelization can easily be implemented by running several grid points at a time as memory allows.

Improvements to LETKF

LETKF makes trade-offs for speed that affect the underlying accuracy of the simulation. The following are areas of improvements that could be applied to LETKF if they can be implemented in such a way that they do not decrease the speed of the LETKF.

1. **Adaptive localization techniques**
   - **Issue:** Local neighborhoods assume no correlations exist beyond the arbitrary local radius. But the physics or weather happen at a multitude of scales, and by setting a fixed radius we're missing effects of different scales.
   - **Improvement:** Scale-dependent localization: mathematically decompose the state into different frequency bands (using wavelets or spectral filters) and apply different local radii to different physical scales simultaneously.
   - **References:** A Multiscale Local Gain Form Ensemble Transform Kalman Filter

2. **Increased Ensemble size for better resolution**
   - **Issue:** Representing incredibly high-dimensional space with far fewer degrees of freedom misses some features. But increasing the size of the ensemble leads to RAM bottlenecks.
   - **Improvement:** Treat the global covariant matrix as a tensor network and do MPO compression. This maintains full dimensionality while having reasonable memory scaling.
   - **References:** Schrödinger-Navier-Stokes Equation for the Quantum Simulation of Navier-Stokes Flows (tensor network techniques to solve Navier-Stokes).
   - Quantum-inspired fluid simulation of two-dimensional turbulence with GPU acceleration (uses MPS formulation and demonstrates how the mathematical inter-scale correlations of chaotic fluid dynamics can be efficiently resolved by treating them as analogous to quantum entanglement).

Improvements to LETKF

LETKF makes trade-offs for speed that affect the underlying accuracy of the simulation. The following are areas of improvements that could be applied to LETKF if they can be implemented in such a way that they do not decrease the speed of the LETKF.

3. Non gaussianity assumptions for the underlying noise.

- **Issue:** We assume Gaussian distributions, but in chaotic weather systems this is not the case. The true physical state might be better represented by a non-linear combination of the ensemble.
- **Improvement:** consider neural network-type methods instead of the standard LETKF. We can use ML to either dynamically alter the Bayesian cost functions or replace the linear matrix operations entirely.
- **References:** Non-Gaussian Detection Using Machine Learning With Data Assimilation Applications, A Neural Network–Based Observation Operator for Coupled Variational Data Assimilation

Previous Works: EMARS

Reference: Greybush, Steven J., et al. "The ensemble Mars atmosphere reanalysis system (EMARS) version 1.0." *Geoscience data journal* 6.2 (2019): 137-150.

It contains: An hourly gridded atmospheric dataset $60\times36\times28$ (Long x Lat X Alt)[1] for mars weather from observations from the Thermal Emission Spectrometer (TES) and Mars Climate Sounder (MCS) temperature retrievals. Include data w/r/t **Temperature, Wind, Surface pressure, dust, water ice, CO_2 ice,** and other quantities.

A reanalysis software performed using the LETKF method that

LETKF is augmented by data assimilation techniques which combine background estimates with physical observations from the state of the planet.

[1] the vertical height follows the terrain and is anisotropic (it varies with distance).

Novel Computing Methods

https://arxiv.org/abs/2512.14444 ML based methods for solving climate forecasts with LETKF methods.

Quantum or quantum-inspired methods:

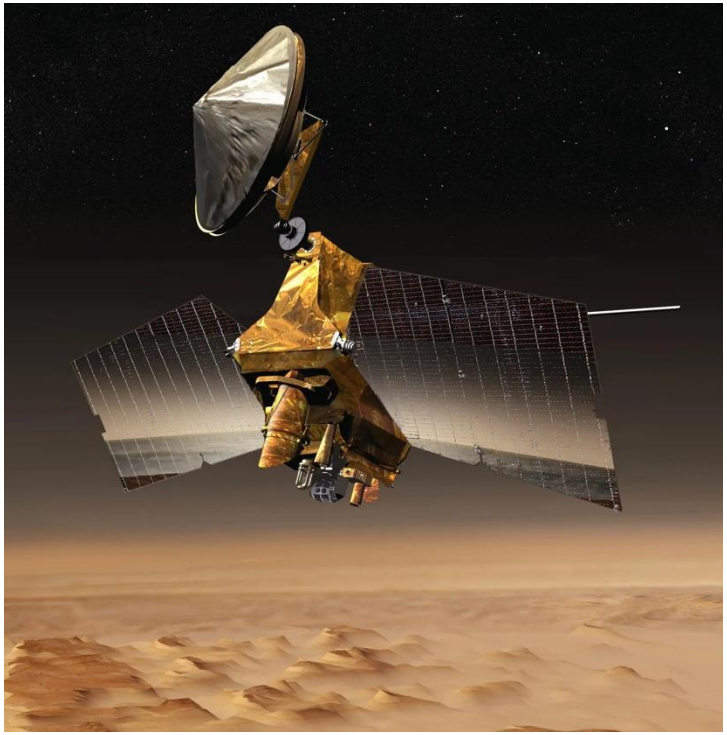
Quantum-inspired fluid simulation of two-dimensional turbulence with GPU acceleration

Toy Problem Part 1

Incorporating non-independent sensor errors

- **R** is the Observation Error Covariance Matrix. When we solve the LETKF equations, $\mathbf{R}^{-1}$ is used to determine the level of confidence in the data measured by the sensors.
- In the LETKF, sensor errors are assumed to be independent, which leads to **R** being a diagonal matrix containing the variance σ of each sensor, making the math easily tractable.
- In reality, these errors are highly correlated[1], and **R** is a dense matrix that makes LETKF computationally intractable.

This is a key computational bottleneck of the LETKF. As a first step towards a full weather forecasting framework that can handle realistic error matrices, we start by building a DA pipeline that can handle them in a one-dimensional, but still chaotic, physics engine.



Mars Reconniassance Orbiter. Image credit: NASA

[1] See Greybush 2019, discussions of "superobbing" and representativeness error

Toy Problem Part II

Incorporating non-independent sensor errors

Goal: Build a 1D DA pipeline that can handle a chaotic PDE with a dense, correlated **R** matrix.

Inputs:
- "True" trajectory.
- Initial background ensemble of perturbed states.
- Sparse observations with heavily spatially correlated noise (resulting in dense, block-diagonal **R** matrix).

Outputs:
- Updated trajectory.
- Benchmark data: the wall-clock time and peak memory consumed during the local neighborhood matrix inversions.

The physics engine

- We can use the 1D Kuramoto-Sivashinsky (KS)[1] equation for our physics engine.
- It is the simplest equation that shows chaotic behavior, making it a good candidate for testing.
- We get the "True" trajectory by initializing the KS eq with a random state and running the solver for our entire time window.
- This trajectory is a stand-in for the true physical state of the atmosphere at every time and location.
- We sample the true trajectory and add correlated Gaussian noise to obtain the sparse observations.
- The DA pipeline does not have the true trajectory; only the noisy observations.
- After running the noisy data through the LETKF, we calculate the error by comparing with the truth.

[1] https://encyclopediaofmath.org/wiki/Kuramoto-Sivashinsky_equation

Toy Problem Part III

Incorporating non-independent sensor errors

Full algorithm

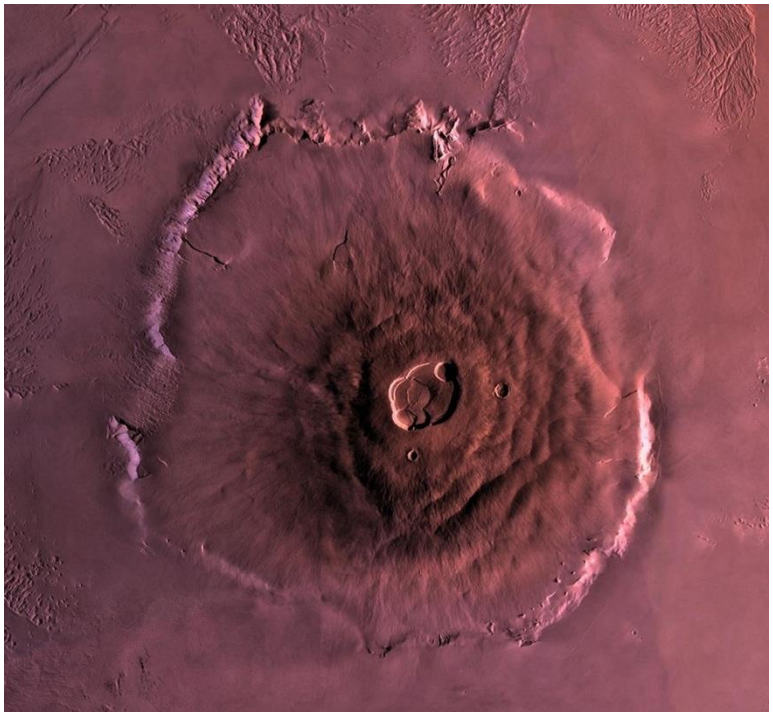
1. **Forecast step (physics engine)**
   - Run ensemble members forward in time using 1D KS equation solver.
   - The 1D wave evolves, bifurcates, and exhibits smooth spatiotemporal chaos.
   - The state tensor is a 2D array that tracks the scalar wave amplitude for each ensemble member.
2. **Correlated data generation**
   - The physics engine pauses. Using the ”true" trajectory, it applies the Cholesky decomposition to inject correlated Gaussian noise.
   - It hands the filter an observation vector $\mathbf{y}^o$ where the errors are entangled, accompanied by the true, dense **R** matrix.
3. **Analysis step (at every grid cell)**
   - Bayesian math of the LETKF. Because **R** is dense, we’ll use e.g. a Cholesky decomposition at every grid point. Will handle it with GPU tensor batching (using e.g. JAX).
   - Standard LETKF uses R-localization to ignore everything outside of the local neighborhood. We cannot do this with a dense matrix, so instead we apply localization to the observation perturbations matrix directly.
   - The local scalar wave amplitude gets updated.
   - All the local updates get put back together into the updated global 1D array, and the updated ensemble members get passed to the physics engine to start the next loop.

Stepping Stone Problem Part I

INTELLIGENT SYSTEMS DIVISION

Solving the shallow water equations on Mars

- The LETKF math inherently assumes that all probability distributions are Gaussian, and that the physics engine represents linear or weakly non-linear dynamics. This is not a good assumption for Mars.
- Mars has extreme topography (like Mons Olympus or the Hellas Basin). When wind encounters these boundaries, it generates severe non-linear gravity waves and gradients.
- During winter, extreme cold causes ~25% of the atmospheric CO_2 to freeze near the poles. This creates a localized vacuum, driving wind currents that break standard fluid mass-conservation.
- Resolution is critical as course dynamical cores will not see as sharp features



Olympus Mons, the largest known volcano in the Solar System, as captured by the Viking 1 Orbiter. It is 27 km tall, and over 600 km at the base.
Image credit: NASA/JPL-Caltech/USGS

We will capture this dynamics through the non-linear shallow water equations (SWE) with static bottom topography and a localized mass sink

Stepping Stone Problem Part II

INTELLIGENT SYSTEMS DIVISION

Solving the shallow water equations on Mars

Goal: Prevent variance collapse when the physics engine encounters extreme topography and localized mass sinks.

Inputs:

- Initial 2D SWE state matrix storing height and momentum.
- Boundary Conditions: topographical map with large mountain and localized, time-dependent mass sink term (polar freeze).
- Initial background ensemble of perturbed states.
- Synthetic satellite data passed through a diagonal **R** matrix.

Outputs:

- Array of 2D momentum and fluid height at each timestep and grid cell.
- Spatial heatmap of the background ensemble variance, which should be non-zero even around the mountain boundaries and mass sinks.

- We go back to the diagonal **R** matrix used in the LETKF algorithm, and focus on $\mathbf{P}^b$ instead.
- $\mathbf{P}^b$ is the Background Error Covariance Matrix, representing the uncertainty of the ensemble before it looks at new satellite data. A large $\mathbf{P}^b$ means the ensemble members disagree, and the filter will put more trust on the satellite data than on the physics engine. The opposite is true for a small $\mathbf{P}^b$.
- When the filter encounters large gradients (due to e.g., mountains), it misinterprets them as model errors and pushes the state towards satellite data, forcing $\mathbf{P}^b$ to 0 (variance collapse). Once $\mathbf{P}^b$ is 0, so is the Kalman gain, which leads to only trusting the physics engine and ignoring all future satellite data.
- Note that $\mathbf{P}^b$ is never computed fully, but instead the matrix of ensemble perturbations $\mathbf{X}^b$ is used to do the math locally.

Stepping Stone Problem Part III

Solving the shallow water equations on Mars

How existing models handle Mars topographic and atmospheric conditions

- In Hunt 2007, where the LETKF was first proposed, the authors only considered a relatively smooth physical model similar to Earth. The LETKF worked well there, but was not tested on conditions representative of the Mars topography and atmosphere.
- In Greybush 2019, their Mars model does include Olympus Mons and CO_2 freezing around the poles. However, they did not address those within the LETKF. They used the legacy GFDL Mars GCM climate engine to pre-process the wind before the data enters the LETKF. And then post-processed any remaining variance issues globally over the whole grid.
- It is difficult to update only parts of this legacy climate engine, or to integrate with newer, more modular approaches, since it works more like a global black box. Our overall goal is moving towards a more modern and flexible approach.

Stepping Stone Problem Part IV

INTELLIGENT SYSTEMS DIVISION

Solving the shallow water equations on Mars

Full algorithm

1. **Forecast step (physics engine)**
   - Run ensemble members forward in time using 2D SWE solver.
   - The fluid is slamming into Olympus Mons and swirling into the polar mass sink, causing non-linear shocks that warp the ensemble.
2. **Data generation**
   - Take the next synthetic satellite observations $\mathbf{y}^o$ and run them through the diagonal **R** matrix.
3. **Analysis step (at every grid cell)**
   - **Localization (Gaspari-Cohn).** The localization function masks observations that are too far from the current cell. This is to make sure that a shockwave at the mountain won't corrupt the math at the equator.
   - **Bayesian math of the LETKF.** Calculate Kalman Gain and update fluid height and momentum based on the trusted observations. Because of the shockwaves and the diagonal **R**, the filter over-trusts the satellite data, and the ensemble variance collapses to zero.
   - **Covariance Inflation (Relaxation to Prior Spread).** The RTPS algorithm multiplies the collapsed ensemble perturbations by a scaling factor to restore the original variance that the LETKF update just tried to destroy.
   - All the local updates get put back together into the updated global 2D array, and the updated ensemble members get passed to the physics engine to start the next loop.

Integration: Toy + Stepping Stone Part I

INTELLIGENT SYSTEMS DIVISION

Integrate non-independent sensor errors with 2D SWE on Mars

Goal: Build a pipeline that can assimilate correlated satellite data while simulating the dynamics of a non-linear topographic shockwave.

Inputs:

- Initial 2D SWE state matrix storing height and momentum.
- Boundary Conditions: topographical map with large mountain and localized, time-dependent mass sink term (polar freeze).
- Initial background ensemble of perturbed states.
- Observation data with correlated noise (resulting in dense, block-diagonal **R** matrix).

Outputs:

- Array of 2D momentum and fluid height at each timestep and grid cell..
- Spatial heatmap of the background ensemble variance, which should be non-zero even around the mountain boundaries and mass sinks.

- We've separately handled inverting dense **R** matrices and the dynamics of physical shocks.
- However, these two added complexities interfere with each other:
  - If we input correlated observations into LETKF (dense **R**) it mathematically trusts them less than if it falsely assumed they were independent. This changes the analysis weights.
  - **The Re-Collapse Risk:** even though the lower trust and subsequent weight update is mathematically correct, because the LETKF is built from the ground up to expect diagonal matrices, this weight shift can lead to an incorrect collapse of the variance, which we need to prevent.

Integration: Toy + Stepping Stone Part II

INTELLIGENT SYSTEMS DIVISION

Integrate non-independent sensor errors with 2D SWE on Mars

Preventing variance re-collapse:

1. **Re-Calibrating RTPS**
   - **Problem:** With a false diagonal matrix, the filter over-trusted the satellite and collapse the variance. We increase RTPS to counter this.
   - **Integration:** But the dense matrix recognizes redundant data, lowering the Kalman Gain, so the variance does not fully collapse. If we leave RTPS at the higher setting, the variance will explode. We need to dial down the RTPS coefficient to a level that's just enough to counteract the loss of variance, but no more.
2. **Matrix Regularization**
   - **Problem:** When observations are highly correlated, the dense matrix has nearly identical rows, which causes the Cholesky solver to output highly oscillating analysis weights right next to each other, creating an artificial, jagged wave that can lead the physics engine astray.
   - **Integration:** Tikhonov Regularization. Before inverting the dense matrix, we artificially add a small amount of independent noise to the diagonal, which caps the extreme negative weights and ensures the fluid remains smooth.
3. **Localization**
   - **Problem:** If we handle localization like for the 2D SWE on a dense matrix, it is no longer positive-definite and the Cholesky solver won't be able to invert it.
   - **Integration:** Apply the Gaspari-Cohn taper directly to the observation perturbations **Y** or the background perturbations $\mathbf{X}^{\mathbf{b}}$ before they interact with **R**.

Challenge Problem Part I

3D Mars primitive equations and global dust storms

- The toy, stepping stone, and integration problems use modern computational tensor methods to handle both correlated satellite noise (dense **R**) and non-linear physical shocks (2D topography) while keeping the Kalman filter from collapsing.
- The final step in our pipeline of Martian Data Assimilation is addressing the OSSE Realism Gap, by moving to a complex 3D thermodynamic engine.
- The standard LETKF assumes strictly Gaussian, linear errors, which is a particularly bad assumption during Martian dust storms, where the radiative heat from the dust creates spatiotemporal chaos and the dust physically blinds satellite sensors, creating non-Gaussian observation errors.
- Legacy systems like EMARS (Greybush 2019) manage this by artificially smoothing the data (superobbing), while standard LETKF math (Hunt 2007) does not provide any means for integrating thermodynamic shocks at the planetary scale.

Challenge Problem Part II

3D Mars primitive equations and global dust storms

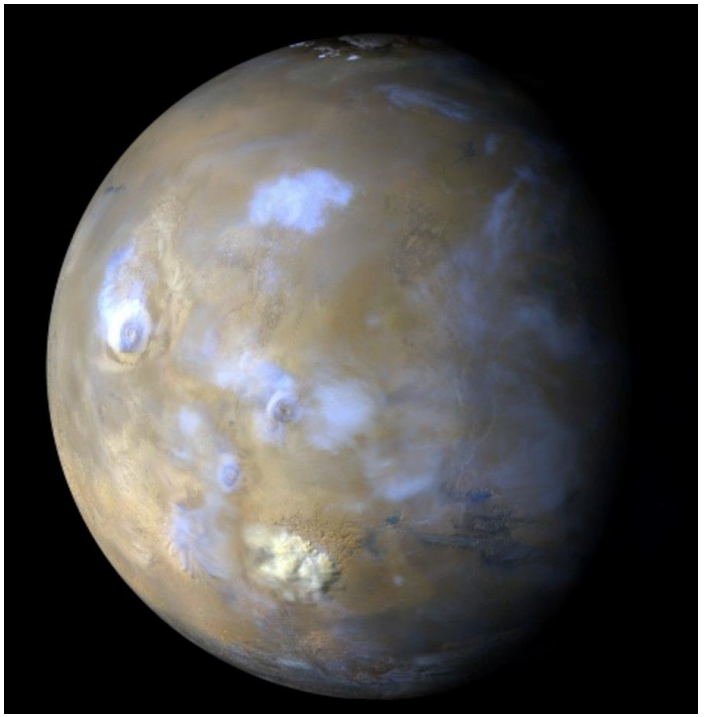
Goal: Scale the integrated, GPU-native LETKF architecture into a 3D thermodynamic environment, capable of assimilating high-resolution, correlated satellite data during a global dust storm.

Inputs:

- Initial 3D state tensor containing 3D momentum, temperature, surface pressure, and dust mixing ratio.
- Initial background ensemble of perturbed states.
- 3D observation data with correlated noise (resulting in dense **R** matrix).

Outputs:

- Array of 3D momentum, temperature, pressure and dust mixing ratio at each timestep and grid cell.
- Spatial heatmap of the background ensemble variance, which should be non-zero even around the mountain boundaries and mass sinks.
- Benchmark showing that assimilating correlated data with a dense matrix produces a better 3D forecast than the legacy method.



Martian dust storm. Image credit: NASA/JPL-Caltech/Malin Space Science Systems

Challenge Problem Part III

3D Mars primitive equations and global dust storms

Full algorithm:

1. **Forecast step (realistic OSSE engine)**
   - Swap the 2D SWE for the 3D primitive equations (incorporating deep-atmosphere thermodynamics and dust advection) and run the ensemble members forward.
2. **Data generation**
   - Because dust can physically block satellite sensors, their errors become non-Gaussian. The filter needs to pass observations through a background quality control gate: any reading that drastically deviates from the ensemble's physical consensus is flagged as a dust-blinded anomaly and assigned a weight of zero, protecting the dense matrix from corrupt data.
3. **Analysis step (at every grid cell)**
   - To protect the dense matrix geometry, apply Y-Space Localization (Gaspari-Cohn on the perturbations, not **R**).
   - To perform the Cholesky inversion correctly on the dense matrix, use Tikhonov regularization.
4. **The 3D Stabilization (RTPS)**
   - Before the local grid updates are stitched back into the global 3D tensor, covariance inflation (RTPS) is applied point-by-point.
   - Because the dense matrix prevents the filter from over-trusting the satellite data, the RTPS is tuned down to a maintenance level. This ensures the ensemble spread survives the transition to the next time step.



Exoplanet Retrieval Modelling

Exoplanet Retrieval: Forward Modelling

INTELLIGENT SYSTEMS DIVISION

Application Details

- Provide accurate retrieval models for JWST and Habitable worlds observatory
- Solutions would help understand exoplanets as well as ones in our solar system

Problem Details

- Solving Radiative Transfer Equations (PDE)
- Inputs: molecular data, opacities, temperatures, pressures, star radiative data
- Output: spectral profile
- $\mu \frac{dI_\nu(\tau_\nu,\mu)}{d\tau_\nu} = I_\nu(\tau_\nu,\mu) - S_\nu(\tau_\nu,\mu)$

Current Approaches

- Use Radiative transfer equations and MCMC to find best description of exoplanet atmospheres using 1d models
- These approaches can describe broad features but often fail if the experimental data is extremely accurate.
- Can fail to capture features of tidely locked exoplanets

Impact of New/Better Solution

- A better solution would reduce approximations with minimal compromise on speed to solving forward model
- Fully utilize JWST and HWO data

Background

- There has been significant investment from NASA into ever improving telescopes to look at distance stars, exoplanets, and galaxies.
- These include James Webb Space Telescope and the future Habitable Worlds Observatory

James Webb Space Telescope

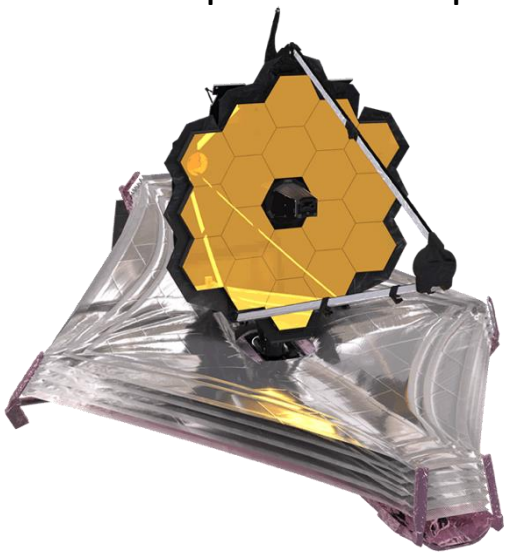
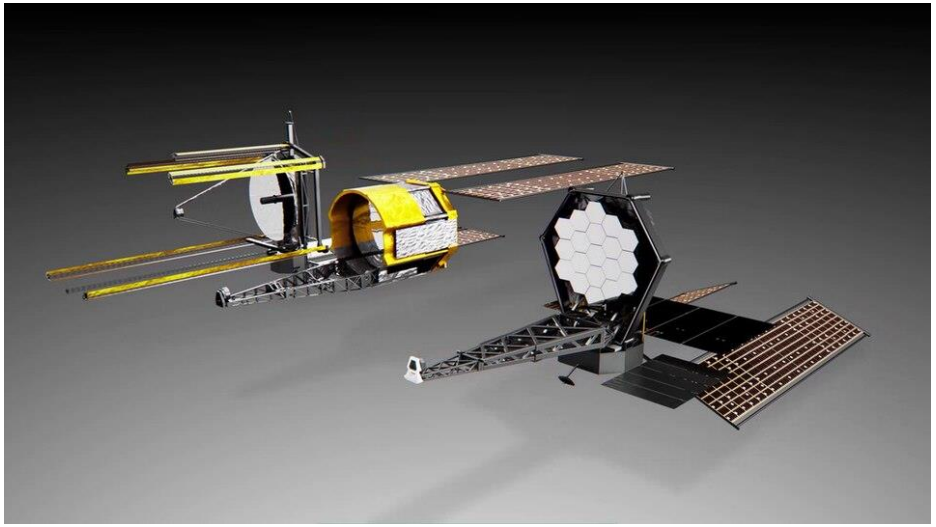


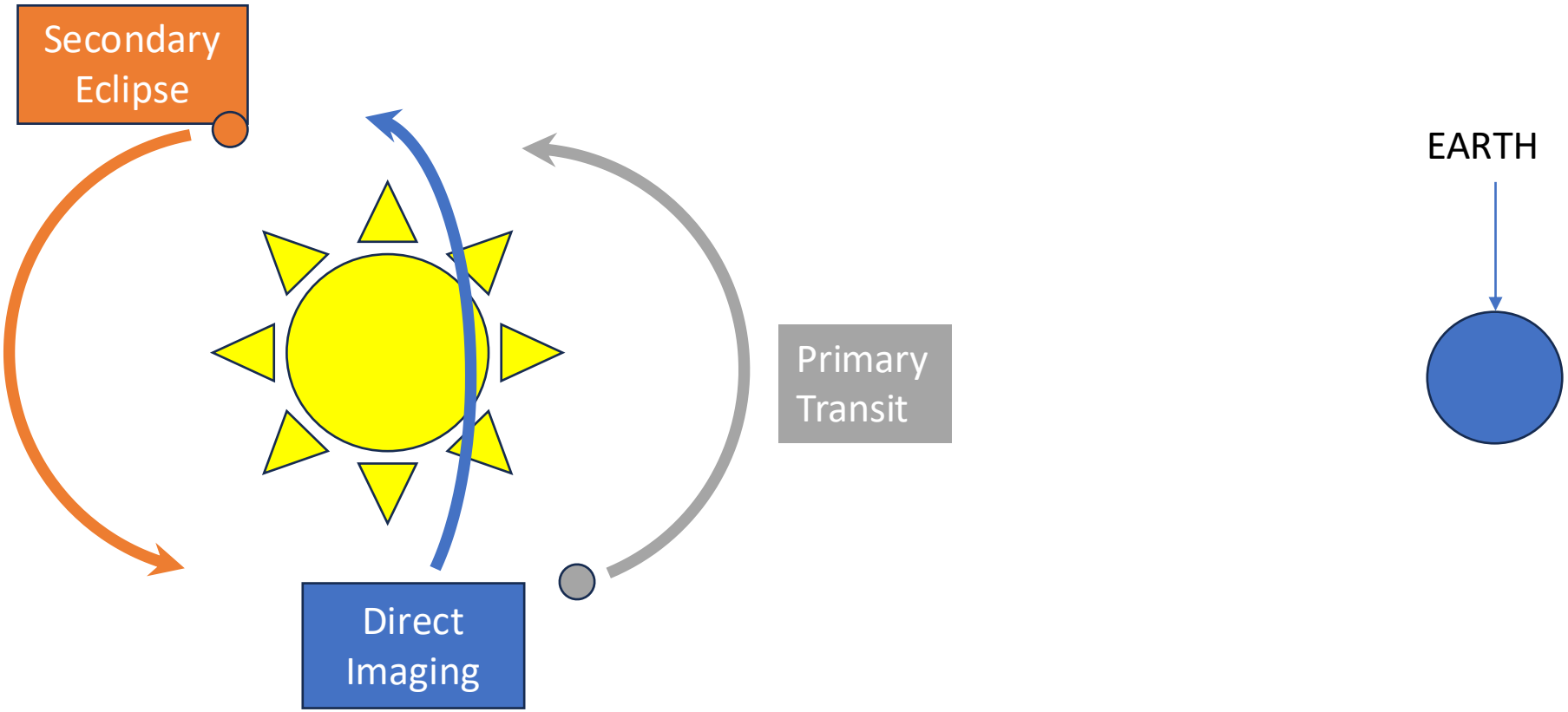
Image Credit: NASA

Habitable Worlds Observatory



NASA's Goddard Space Flight Center Conceptual Image Lab

Types of Data Observed

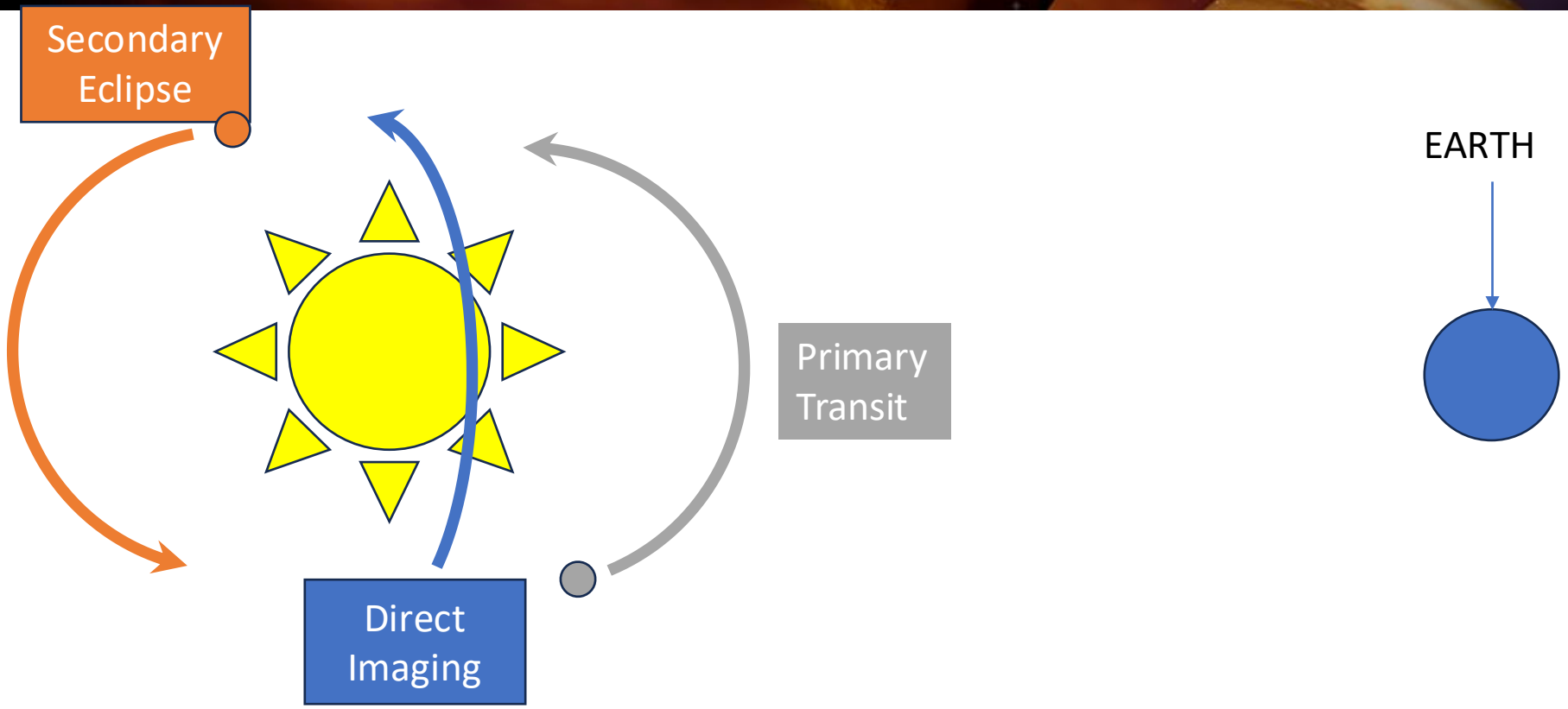


Primary Transit: Exoplanet travels in front of star, dimming caused by light travelling through planet atmosphere

Secondary Eclipse: Emitted Light change by planet travelling behind the star is used to characterize planet

Direct imaging: Light reflects of planet but is not interfering with stellar spectrum

Observation Complexity



Primary Transit: Requires high precision as light typically transmits through the exoplanet atmosphere

Secondary Eclipse: Requires lower precision as light is reflected off the atmosphere with broader signals

Direct imaging: Similar to secondary eclipses these require lower precision as most light received is reflected (as opposed to transmitted) through the atmosphere.

Impact of improved solution

Financial

- Greater output for existing and future observing missions

Additional NASA impacts

- Improved analysis of existing planetary bodies in solar system without existing satellite observing

Retrieval Modelling for Exoplanets





INTELLIGENT SYSTEMS DIVISION

This consists of two parts: **Forward model** and **comparison with spectrum**

The forward model is used to derive an example exoplanet and the spectrum expected to be seen using the 1d or 3d versions of the radiative transfer equation

$$\mu \frac{dI_\nu(\tau_\nu, \mu)}{d\tau_\nu} = I_\nu(\tau_\nu, \mu) - S_\nu(\tau_\nu, \mu)$$

This needs to be *fast* so that millions of different models can be generated and solved for and updated efficiently to find new parameters that best describe the observed planet.

The comparison to an exoplanet spectra is relatively fast and uses Bayesian inferencing to update the forward model with improved parameters and different assumptions, this might use Markov chain monte Carlo or some other related method.

The key computational bottleneck is solving the Radiative Transfer Equation in the forward model.

Retrieval Modelling for Exoplanets



Image created by Reto Stöckli, Nazmi El Saleous, and Marit Jentoft-Nilsen, NASA GSFC.

Planet defined by a set of parameters $\vec{x}$ such as temperature, chemical composition, etc.

Output of model

Update model from fit

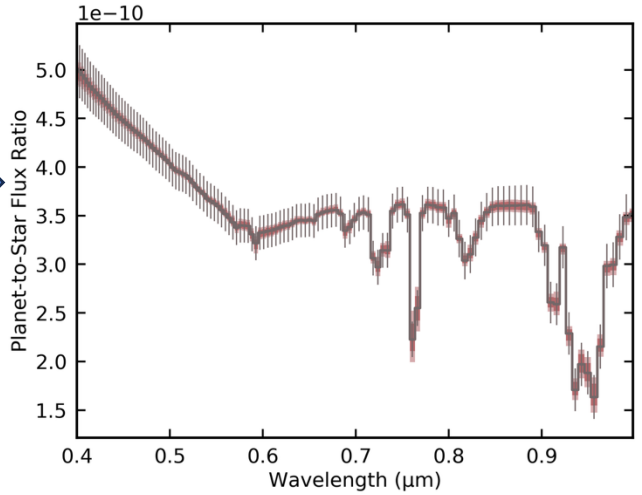


Image taken from Robinson et al. "**Exploring and Validating Exoplanet Atmospheric Retrievals with Solar System Analog Observations**" under cc-by-4.0

Output of the forward model $F(\vec{x})$ that's compared to observational data.

Updates and generation of $F(\vec{x})$ needs to be done quickly to ensure that millions of configurations can be identified and a reasonably good solution for the right hand figure can be found.

Mathematical background for Plane Parallel RTE

We want to solve the following radiative transfer equation

$$\mu \frac{dI_\nu(\tau_\nu, \mu)}{d\tau_\nu} = I_\nu(\tau_\nu, \mu) - S_\nu(\tau_\nu, \mu)$$

$S_\nu(\tau_\nu, \mu)$ is the source function defined as

$$S_\nu(\tau_\nu, \mu, \phi, \vec{x}) = \omega_\nu F_\nu^m e^{-\tau_\nu/\mu_m} \frac{P_\nu(\tau_\nu, \mu, \phi, -\mu_m, \phi_m, \vec{x})}{4\pi} + (1 - \omega_\nu) B_\nu\big(T(\tau_\nu)\big)$$

$$+\omega_\nu \int_0^{2\pi} d\phi' \int_{-1}^{1} d\mu'\, I_\nu(\tau_\nu, \mu', \phi', \vec{x}) P_\nu(\tau_\nu, \mu, \phi, \mu', \phi', \vec{x})$$

With "$\omega_\nu = \omega_\nu(\tau_\nu)$ is the frequency-dependent single scattering albedo, F_ν is the top-of-atmosphere solar irradiance, μ is the solar zenith angle, ϕ is the solar azimuth angle, P_ν is the scattering phase function, B_ν is the Planck function, and $T(\tau_\nu)$ is the temperature profile in the atmosphere" [1]

Boundary conditions need to be chosen to find the "steady-state" solution to problem.

[1] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018) page 81.

Mathematical background for Plane Parallel RTE

Data bases such as HITRAN and are what inform the optical extinction depth term τ_ν
These databases are typically loaded into the software that runs the forward model.

The two steam or disort methods typically solve the problems at scale using vectorized linear algebra techniques to reduce the computation time.

Mathematical background for 3D RTE

The 3d case for direct imaging typically replaces the 1d plane parallel solution used above with NxN pixation of the globe.

The flux is then defined as

$$F_\lambda(\alpha) = \frac{\cos(\alpha)+1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{G,i} w_{C,j} I_\lambda(\tau = 0; \alpha, \mu_{s,ij}, \mu_{o,ij})$$

With $\mu_{s,ij} = \sin(\cos^{-1}(x_{C,j}))\cos(\zeta_i - \alpha)$ and $\mu_{o,ij} = \sin(\cos^{-1}(x_{Cj}))\cos(\zeta_i)$ where α is the planetary phase angle, $\zeta_i = \sin^{-1}\left(\frac{\cos(\alpha)+1}{2}\left(x_{Gi} - \frac{\cos\alpha - 1}{\cos(\alpha)+1}\right)\right)$, x_{Ci} and x_{Gi} are the Chebyshev and Gaussian points used in the integration procedure. [1]

[1] Horak, H. G., & Little, S. J. 1965, ApJS, 11, 373, doi: 10.1086/190119

Specification of a Problem Instance

Planet Composition

- Surface Pressure P_0
- Gas composition f_g
- Gas mixing ratios R_g
- Planet Radius $R_\oplus$
- Surface gravity g
- Surface albedo A_s
- Cloud top pressure p_t
- Cloud thickness d_p
- Cloud optical depth τ
- Cloudiness fraction f_c
- Surface Temperature T
- Gas absobtion tables (gigabytes)

Orbit Composition

- Orbiting star temperature T_*

Orbit radius R_p



Challenge Problems: Earthlike planets for HWO

Toy Problem: Earth Model

INTELLIGENT SYSTEMS DIVISION

Inputs

- $P_0 = 1$ (bar)
- $f_{H20} = 3\times10^{-3}$
- $f_{O_3} = 7\times10^{-7}$
- $f_{O_2} = 0.21$
- $R_\oplus = 1$ (Earth Radii)
- $g = 9.8$ (m/s^2)
- $A_s = 0.05$
- $p_t = 0.6$
- $d_p = 0.1$
- $\tau = 10$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 1d modeling techniques
Study using both direct imaging of light

Direct Image Orbit

Observation Point



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of [1] other SOA

[1] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018) page 81.

Stepping Stone Problem: Earth Model

INTELLIGENT SYSTEMS DIVISION

Inputs

- $P_0 = 1$ (bar)
- $f_{H20} = 3\times10^{-3}$
- $f_{O_3} = 7\times10^{-7}$
- $f_{O_2} = 0.21$
- $R_\oplus = 1$ (Earth Radii)
- $g = 9.8$ (m/s^2)
- $A_s = 0.05$
- $p_t = 0.6$
- $d_p = 0.1$
- $\tau = 10$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 3d modeling techniques
Study using both direct imaging of light

Direct Image Orbit

Observation Point



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of [1] other SOA

[1] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018) page 81.

Stepping Stone Problem: Earth Model

INTELLIGENT SYSTEMS DIVISION

Inputs

- $P_0 = 1$ (bar)
- $f_{H20} = 3\times10^{-3}$
- $f_{O_3} = 7\times10^{-7}$
- $f_{O_2} = 0.21$
- $R_\oplus = 1$ (Earth Radii)
- $g = 9.8$ (m/s^2)
- $A_s = 0.05$
- $p_t = 0.6$
- $d_p = 0.1$
- $\tau = 10$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 3d modeling techniques
Study using direct imaging of light and removing constraint on two stream methods for RTE

Direct Image Orbit

Observation Point



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of [1] other SOA

[1] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018) page 81.

Stepping Stone Problem: Earth Model

INTELLIGENT SYSTEMS DIVISION

Inputs

- $P_0 = 1$ (bar)
- $f_{H20} = 3\times10^{-3}$
- $f_{O_3} = 7\times10^{-7}$
- $f_{O_2} = 0.21$
- $R_\oplus = 1$ (Earth Radii)
- $g = 9.8$ (m/s^2)
- $A_s = 0.05$
- $p_t = 0.6$
- $d_p = 0.1$
- $\tau = 10$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 3d modeling techniques
Study using direct imaging of light and removing using high resolution absorbtion data

Direct Image Orbit

Observation Point



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of [1] or other SOA

[1] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018) page 81.

Stepping Stone Problem: Earth Model

INTELLIGENT SYSTEMS DIVISION

Inputs

- $P_0 = 1$ (bar)
- $f_{H20} = 3\times10^{-3}$
- $f_{O_3} = 7\times10^{-7}$
- $f_{O_2} = 0.21$
- $R_\oplus = 1$ (Earth Radii)
- $g = 9.8$ (m/s^2)
- $A_s = 0.05$
- $p_t = 0.6$
- $d_p = 0.1$
- $\tau = 10$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 3d modeling techniques
Study using direct imaging of light and relax chemical equilibrium

Direct Image Orbit

Observation Point



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of [1] or other SOA

[1] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018) page 81.

Challenge Problem: Earth Model

INTELLIGENT SYSTEMS DIVISION

Inputs

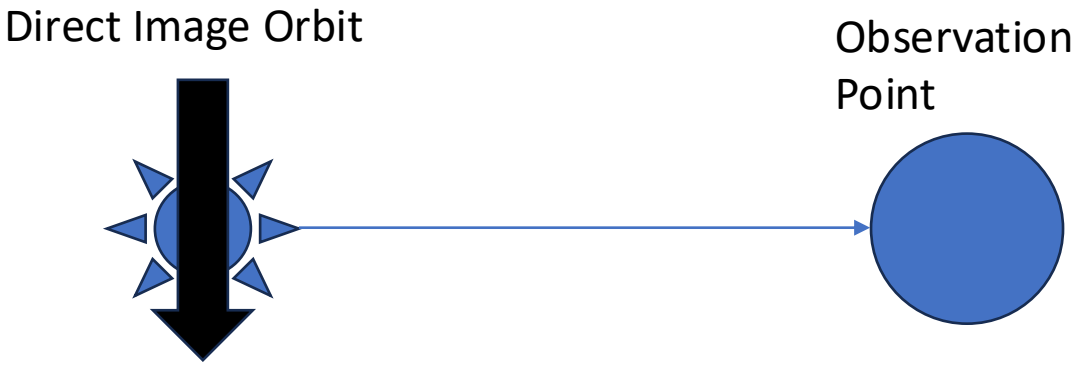
- $P_0 = 1$ (bar)
- $f_{H20} = 3\times10^{-3}$
- $f_{O_3} = 7\times10^{-7}$
- $f_{O_2} = 0.21$
- $R_\oplus = 1$ (Earth Radii)
- $g = 9.8$ (m/s^2)
- $A_s = 0.05$
- $p_t = 0.6$
- $d_p = 0.1$
- $\tau = 10$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 3d modeling techniques
Study using direct imaging of light and relaxing joint constraints from stepping stone problems



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of [1] or other SOA

[1] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018) page 81.



Challenge Problems: Gas Giant like planets for James Webb Space Telescope

Toy Problem: WASP 96B

INTELLIGENT SYSTEMS DIVISION

Inputs

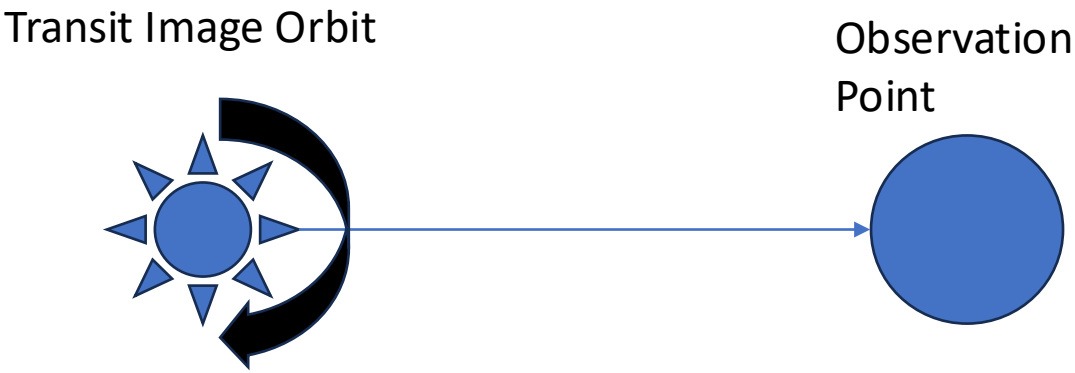
- $P_0 = 1$ (bar)
- $f_{H_2} = 0.85$
- $f_{He} = 0.15$
- $f_{H20} = 2.6\times10^{-4}$
- $f_{CO} = 5.6\times10^{-4}$
- $f_{CO_2} = 4.1\times10^{-5}$
- $f_K = 1.4\times10^{-7}$
- $f_{\mathrm{Na}} = 10^{-8}$
- M = 0.49 (Jupiter Mass)
- $R_J = 1.2$ (Jupiter Radii)
- $g = 8.1$(m/s^2)
- $p_t = 0.1$ bar
- $d_p = 0.1$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 1d modeling techniques
Study using both transit imaging of light



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of SOA

Stepping Stone Problem: WASP 96B

INTELLIGENT SYSTEMS DIVISION

Inputs

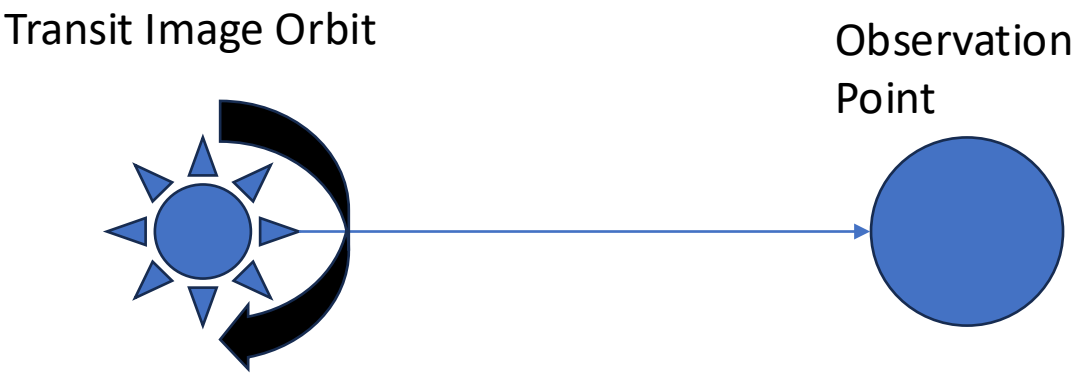
- $P_0 = 1$ (bar)
- $f_{H_2} = 0.85$
- $f_{He} = 0.15$
- $f_{H20} = 2.6\times10^{-4}$
- $f_{CO} = 5.6\times10^{-4}$
- $f_{CO_2} = 4.1\times10^{-5}$
- $f_K = 1.4\times10^{-7}$
- $f_{\mathrm{Na}} = 10^{-8}$
- M = 0.49 (Jupiter Mass)
- $R_J = 1.2$ (Jupiter Radii)
- $g = 8.1$(m/s^2)
- $p_t = 0.1$ bar
- $d_p = 0.1$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 1d modeling techniques
Study using both transit imaging of light relax two stream approximation



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of SOA

Stepping Stone Problem: WASP 96B

INTELLIGENT SYSTEMS DIVISION

Inputs

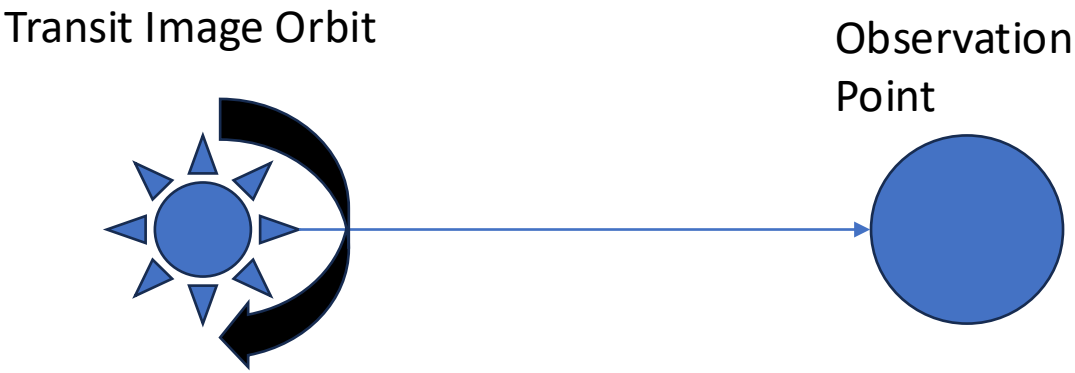
- $P_0 = 1$ (bar)
- $f_{H_2} = 0.85$
- $f_{He} = 0.15$
- $f_{H20} = 2.6\times10^{-4}$
- $f_{CO} = 5.6\times10^{-4}$
- $f_{CO_2} = 4.1\times10^{-5}$
- $f_K = 1.4\times10^{-7}$
- $f_{\mathrm{Na}} = 10^{-8}$
- M = 0.49 (Jupiter Mass)
- $R_J = 1.2$ (Jupiter Radii)
- $g = 8.1$(m/s^2)
- $p_t = 0.1$ bar
- $d_p = 0.1$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 3d modeling techniques
Study using both transit imaging of light



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of SOA

Stepping Stone Problem: WASP 96B

INTELLIGENT SYSTEMS DIVISION

Inputs

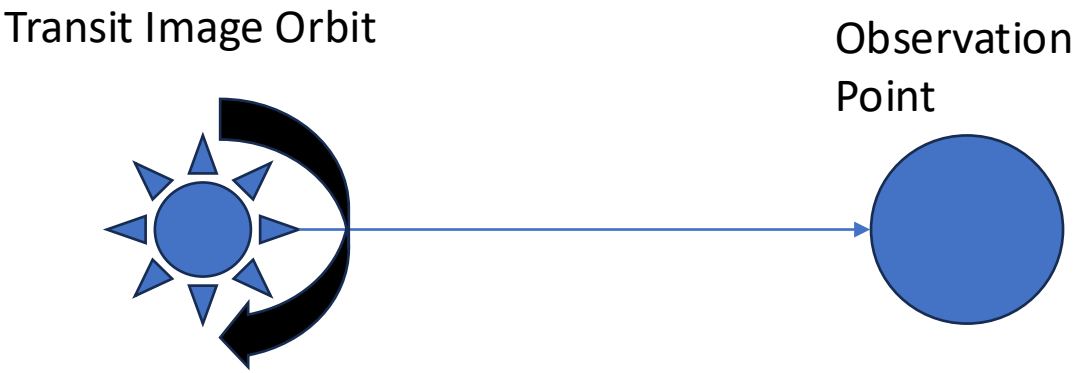
- $P_0 = 1$ (bar)
- $f_{H_2} = 0.85$
- $f_{He} = 0.15$
- $f_{H20} = 2.6\times10^{-4}$
- $f_{CO} = 5.6\times10^{-4}$
- $f_{CO_2} = 4.1\times10^{-5}$
- $f_K = 1.4\times10^{-7}$
- $f_{\mathrm{Na}} = 10^{-8}$
- M = 0.49 (Jupiter Mass)
- $R_J = 1.2$ (Jupiter Radii)
- $g = 8.1$(m/s^2)
- $p_t = 0.1$ bar
- $d_p = 0.1$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 1d modeling techniques
Study using both transit imaging of light using high resolution absorbtion data



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of SOA

Stepping Stone Problem: WASP 96B

INTELLIGENT SYSTEMS DIVISION

Inputs

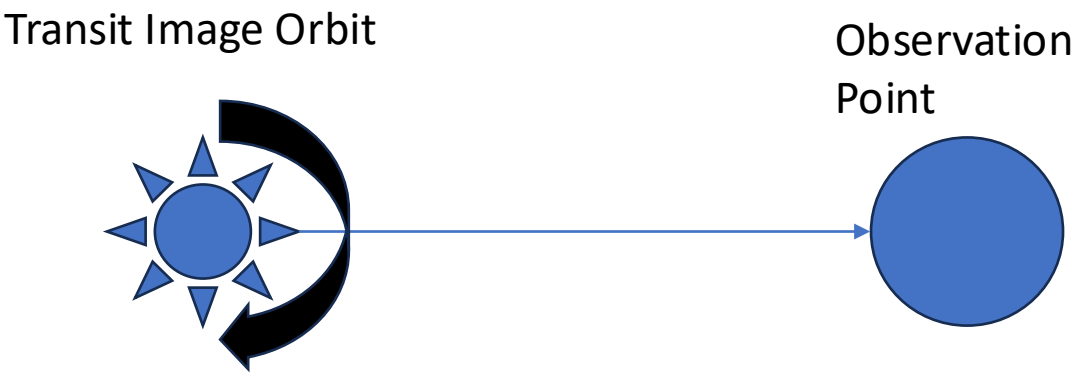
- $P_0 = 1$ (bar)
- $f_{H_2} = 0.85$
- $f_{He} = 0.15$
- $f_{H20} = 2.6\times10^{-4}$
- $f_{CO} = 5.6\times10^{-4}$
- $f_{CO_2} = 4.1\times10^{-5}$
- $f_K = 1.4\times10^{-7}$
- $f_{\mathrm{Na}} = 10^{-8}$
- M = 0.49 (Jupiter Mass)
- $R_J = 1.2$ (Jupiter Radii)
- $g = 8.1$(m/s^2)
- $p_t = 0.1$ bar
- $d_p = 0.1$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 1d modeling techniques
Study using both transit imaging of light using relaxing chemical equilibrium constraints



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of SOA

Challenge Problem: WASP 96B

INTELLIGENT SYSTEMS DIVISION

Inputs

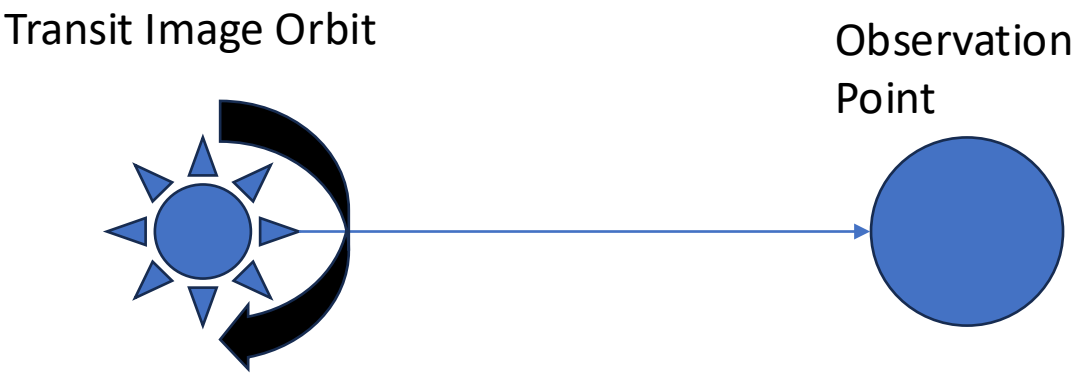
- $P_0 = 1$ (bar)
- $f_{H_2} = 0.85$
- $f_{He} = 0.15$
- $f_{H20} = 2.6\times10^{-4}$
- $f_{CO} = 5.6\times10^{-4}$
- $f_{CO_2} = 4.1\times10^{-5}$
- $f_K = 1.4\times10^{-7}$
- $f_{\mathrm{Na}} = 10^{-8}$
- M = 0.49 (Jupiter Mass)
- $R_J = 1.2$ (Jupiter Radii)
- $g = 8.1$(m/s^2)
- $p_t = 0.1$ bar
- $d_p = 0.1$
- $f_c = 0.5$

Outputs

- Spectrum as a function of wavelength 5-30um

Requirements

Solution using 3d modelling technique relaxing all prior constraints for transit imaging.



Goals

Perform calculations that are equivalent to in speed and accuracy the Retrieval models of SOA

Existing RTE software

Software	Citation
CHIMERA	https://github.com/mrline/chimera
TauREx	https://taurex3.readthedocs.io/en/latest/
NEMESIS	https://nemesiscode.github.io/nemesis.html
HyDRA	https://arxiv.org/abs/1710.06433
POSEIDON	https://poseidon-retrievals.readthedocs.io/en/latest/
Trident	https://arxiv.org/abs/2111.05862
SMART	Meadows, V. S., & Crisp, D. 1996, J. Geophys. Res., 101, 4595, doi: 10.1029/95JE03567
Rfast	https://arxiv.org/pdf/2204.04231

References

[Rob18] T.D. Robinson, D. Crisp / Journal of Quantitative Spectroscopy & Radiative Transfer 211 (2018)

[May26] L. C. Mayorga *et al* 2026 *AJ* **172** 87

[Nik26] Parenteau, Niki, et al. "NASA Decadal Astrobiology Research and Exploration Strategy (NASA-DARES 2025) White Paper--Habitable Worlds Observatory Living Worlds Science Cases: Research Gaps and Needs." *arXiv preprint arXiv:2601.06386*(2026).

Points of Contact

Jacob Lustig-Yaeger (John's Hopkins Applied Physics Laboratory)

Tyler Robinson (Arizona State University)

Thomas Fauchez (American University / GSFC)

Oreste Reale (Science Systems and Applications Inc / GSFC)

Acknowledgements

This work was supported by NASA's Office of the Chief Science Data Officer.

Zoe Gonzalez-Izquierdo and Erik Gustafson are KBR employees working under the Prime Contract No. 80ARC020D0010 with the NASA Ames Research Center.

The authors acknowledge the use of NASA LLM tools for help with background research, problem identification and specification.