Derivation of effective properties based on porous scale simulations using filtering techniques

Predictive Materials Modeling Group

Aleksander Zibitsker

University of Kentucky

Bruno Dias, Jeremie Meurisse and Nagi Mansour

Analytical Mechanics Associates, INC.

August 27, 2022

Abstract

This study presents a method for derivation of effective properties at the interface and in-depth of porous materials. The method defines a Representative Elementary Volume (REV) and applies filtering techniques to compute effective properties such as porosity and flow quantities, such as velocity and pressure. The script, developed to process the data was tested on the VTK type files that contain the mesh information and the flow solution. The method allows to choose between two types of filters, such as cellular and top-hat and define the size of the REV and number of samples along the domain. Extraction of the REV from the domain is performed to exact boundaries requested for the user. This is done using a triangulation technique and cutting through the cells to comply to the requested boundaries of the volume. The method can be applied to both structured and unstructured meshes.

Filtering the material porosity and flow quantities involves integration of the numerical data. The algorithm provides three integration methods, such as Riemann sum, Monte Carlo and Quadrature rule to perform the integration. The Monte-Carlo technique permits the use of either uniform or linearly spaced distribution of points. The Quadrature rule is currently applicable to tetrahedral element types. The Monte Carlo and Quadrature rule methods require interpolation of the flow quantities at the sample points. For interpolation, two methods were tested and are readily available, Gaussian interpolation and re-sampling. It has been shown that re-sampling method has better consistency and acceptable accuracy in interpolation of the data.

The algorithm was written in Python language and uses a number of modules. The major module besides numpy is PyVista. It is used to process the computational domain, clip the REV and interpolate the data. Quadrature rule integration was performed using a quadpy module. ParaView software was used externally to convert the flow solution to the VTK (or more specifically VTU) format. Integration of ParaView in the same environment with PyVista encountered problems and could not be implemented in this work.

The developed algorithm is expected to be applicable to unstructured meshes and more complex porous structures as soon as the data can be passed in VTK type format.

With the report is provided Python script for filtering the solution and a Matlab script for simple generation and processing of 2-D and 3-D porous channel geometries. The two scripts don't communicate.

Contents

List of Figures

Chapter 1

Introduction

Simulation of flow dynamics in porous medium is in general a very challenging task. The challenge arises from the fact that the structure of a porous medium is very complex and contains very small, microscopic features. Simulation of flow through such complex structure would require to resolve vastly different physical and geometrical scales. One example, is the meshing problem of such complex structures that would require very sophisticated technique to resolve the boundary layer region near the solid structures, while adapting to the curvy and random nature of the pores. This increases drastically the number of computational cells required to properly resolve the porous region and leads to the increase in computational complexity of such solution.

Simplification of the flow problem inside a porous material can be made by derivation of the Volume Averaged Navier Stokes (VANS) equations as described in details by Whitaker [1999]. The averaging of the Navier Stokes equations is performed over a Representative Elementary Volume (REV) which contains the smallest volume over which the properties of the material and local physics do not change. It means, that for larger volume, averaging over the REV would lead to the same effective result. This approach allows to derive the effective properties of a non-homogeneous materials and assume continuity in the derived properties and the modeled physics.

Example of a porous structure consisting of randomly distributed thin carbon fibers bonded together is shown in Fig. 1.1. The three squares red, orange and green represent three potential averaging volumes to derive the effective properties of the material. It can be seen that averaging porosity of the material over the red volume would mostly represent a solid material, with very low value of porosity. Averaging, instead, over the orange volume, would lead to a serious overestimation of the effective porosity, as the orange region mostly consists of void (gas). Only averaging over the green volume would encompass enough material to capture the effect of both media (void and solid) to represent the effective porosity of the material. Increasing the green volume size would not effect the averaged value of the derived property and hence, the green volume would be the representative one.

Modeling interaction of the flow and porous material, which is done in simulation of charring Thermal Protection Materials (TPM), can be performed in various ways. One of the approaches is solution of the unified system of equations for both mediums, the external flow and the porous material. Such approach, of using VANS equations has been recently demonstrated in the work of Schrooyen [2015] and Duzel [2020]. In this approach, both,

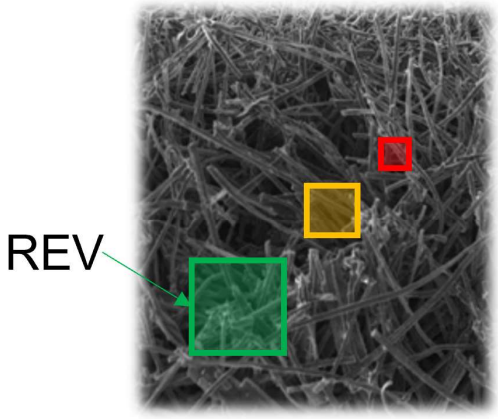


Figure 1.1: *Example of porous structure and three averaging volumes*

the flow and the material domains are solved using one set of VANS equations in a unified fashion. In this approach, VANS governing equations depend on the local averaged properties of the modeled medium, whether the solution is computed within the external flow region or within the porous material. The additional aspect in application of the VANS equations to the unified problem comes in the treatment of the interface region between the bulk flow and the porous region. At the interface the local properties change drastically and the correct application of the VANS equations requires a proper derivation of the effective properties across the interface. Figure 1.2 shows a porous material embedded into a flow domain and a REV placed at the interface between the two media. Following the application of the averaged approach in derivation of the effective properties, the procedure needs to be applied also across the interface to proper model the transition of the properties between the two medias. Application of the averaging procedure across the interface will result in a transition zone where the effective properties of the porous material will gradually transition to the effective properties of the pure flow, preserving the continuity assumption in the governing equations.

The two main properties, characteristic to the porous material is the porosity and permeability. Porosity solely depends on the geometrical structure of the material, while permeability is a function of the local porosity and the flow quantities such velocity and pressure. To derive the effective porosity, only the porous material structure and a proper meshing of the domain is required. Derivation of the permeability, in turn, requires simulation of the flow through the porous medium. It appears as a chicken and egg problem. To simulate the flow inside the porous material we need the effective properties and to derive the effective properties we need the flow solution. To solve this problem, flow behavior needs to be simulated on the explicit structure of the porous material using a Direct Numerical Simulation (DNS). After the flow solution is obtained, the effective permeability of the medium can be computed.

Derivation of the effective properties is based on the averaging inside the REV and depends on the type of the filtering applied. The type of the filter appears to depend on the type of porosity of the material, whether the porous structure is ordered or disordered. The choice of the filter in derivation of the effective properties adds additional aspect in the use of VANS equations in the flow modeling problem.

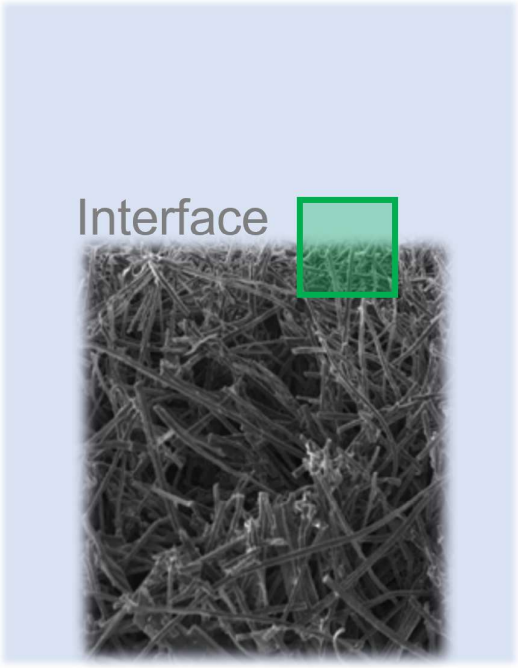


Figure 1.2: *Unified flow problem and interface region*

In this report, the complete sequence of the procedure in derivation of the effective properties of the material and across the interface is presented and discussed. Starting from a short background on the filtering techniques, the report presents a short study on verification of the filtering methods, followed by description of the DNS setup and post-processing steps in derivation of the effective properties. In the results chapter, first verification of the porosity filtering approach is presented, showing excellent agreement with the reference data. Later, the filtered results of the modeled problem, including the filtering of the DNS solution are presented, followed by discussion of the various aspects and challenges encountered in derivation of the filtering algorithm. Finally, conclusions and future work suggestions are presented.

Chapter 2

Methodology

2.1 Filtering background

Volume average of a certain quantity in porous medium is defined as a convolution integral shown in Eq. (2.1)

$$\langle f \rangle = \int_V \gamma(\boldsymbol{r}) m(\boldsymbol{r} - \boldsymbol{x}) f(\boldsymbol{r}) d^3 r, \tag{2.1}$$

where $< f >$ is the spatial average of the quantity f, γ is the phase indicator that equals to 1 when the vector $\boldsymbol{r}$ points into the fluid phase and equals to 0 when $\boldsymbol{r}$ points into the solid phase and finally, m is the filter kernel that depends on the relative distance from the center of the REV. Figure 2.1 shows the schematics of a sample porous medium and the filtering REV, shown here for simplicity in the two-dimensional form. The black squares represent the solid phase and the domain within the red square is the averaging REV. The size of the REV is designated as $2l$ in each direction. Vector $\boldsymbol{r}$ points at the sampling location and vector $\boldsymbol{x}$ points at the center of the REV. Integration is performed over the volume by sampling the value of the phase indicator, filter kernel and the measured quantity at every point $\boldsymbol{r}$ in the REV domain.

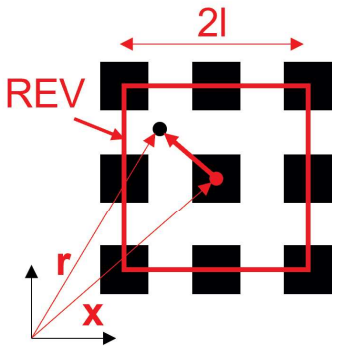


Figure 2.1: *Schematics of the 2-D filtering domain*

The filter kernel m should be selected with a special care, as the type of porous medium

would require different types of filters to be applied to represent correctly the effective properties. The general form of the filter used in this study is shown in Eq. (2.2)

$$m(\boldsymbol{y}) = \frac{1}{C}\left[G(y_1, l_1)G(y_2, l_2)G(y_3, l_3)\right], \tag{2.2}$$

where G is the filter kernel function, $\boldsymbol{y} = \boldsymbol{r} - \boldsymbol{x}$ and C is the filter normalization coefficient. This filter form is called a separable form, where the filter can be written as a product of more simple filters. Davit and Quintard [2017] studied the effect of three different types of filter kernels on the averaged quantities in different types of porous medium. Two major types of filters can be defined based on this work.

The first type is a rectangular filter, or the so called "top-hat" filter, which has been shown to perform well in disordered porous medium Quintard and Whitaker [1994]. This type of the filter is a simple arithmetic average of the filtered quantity over the REV and its shape in 2-D can be seen in Fig. 2.2(a). The kernel function of the top-hat filter is defined in Eq. (2.3) below

$$G(\boldsymbol{y}, \boldsymbol{l}) = \begin{cases} 1, & |\boldsymbol{y}| \leq 1 \\ 0, & |\boldsymbol{y}| > 1 \end{cases} \tag{2.3}$$

for which, the normalization coefficient C is the volume of the rectangular REV, defined as $C = \frac{1}{2l_1 2l_2 2l_3}$.

The second filter type is the triangular filter, or the so called "cellular" filter. This filter type has been shown to be applicable to an ordered porous medium Quintard and Whitaker [1994]. The shape of the cellular filter in one dimension is a triangle and in two dimensions is a pyramid as shown in Fig. 2.2(b). The kernel function of the cellular filter is defined in Eq. 2.4 below

$$G(\boldsymbol{y}, \boldsymbol{l}) = \begin{cases} \boldsymbol{l} - |\boldsymbol{y}|, & |\boldsymbol{y}| \leq 1 \\ 0, & |\boldsymbol{y}| > 1 \end{cases} \tag{2.4}$$

for which, the normalization coefficient C is defined as $C = \frac{1}{l_1^2 l_2^2 l_3^2}$.

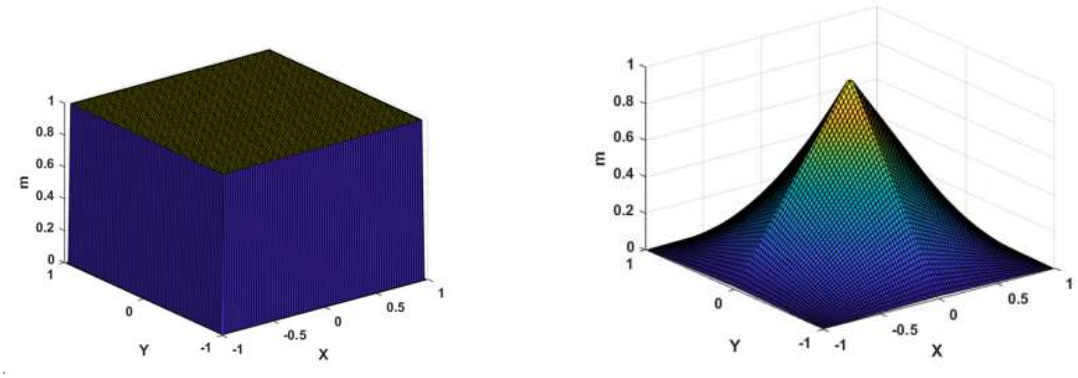
The important property of filters is the normalization condition, it requires that the integral of the filter over the domain must be equal to 1, i.e.

$$\int_V m(\boldsymbol{r} - \boldsymbol{x}) d^3 r = 1 \tag{2.5}$$

2.2 Problem description

2.2.1 Porous channel flow problem

Development of the filtering methodology for porous materials was based in this study on a well known porous channel flow problem from Beavers and Joseph [1967]. The original problem represents a flow through a three-dimensional channel with permeable wall, modeled as a Cartesian grid of cubes, shown in Fig. 2.3 in two-dimensions. The half height of the channel is 80 μm and the channel consists of $60 \times 7 \times 3$ number of solid cubes with side $d_s = 5\ \mu m$. The distance between the cubes is also equal to d_s, which makes this an ordered



(a) Top-hat filter. (b) Cellular filter.

Figure 2.2: *Examples of two types of filters in 2-D.*

porous medium. The porous region is located $2d_s$ away from the centerline of the channel and one d_s from the bottom wall.

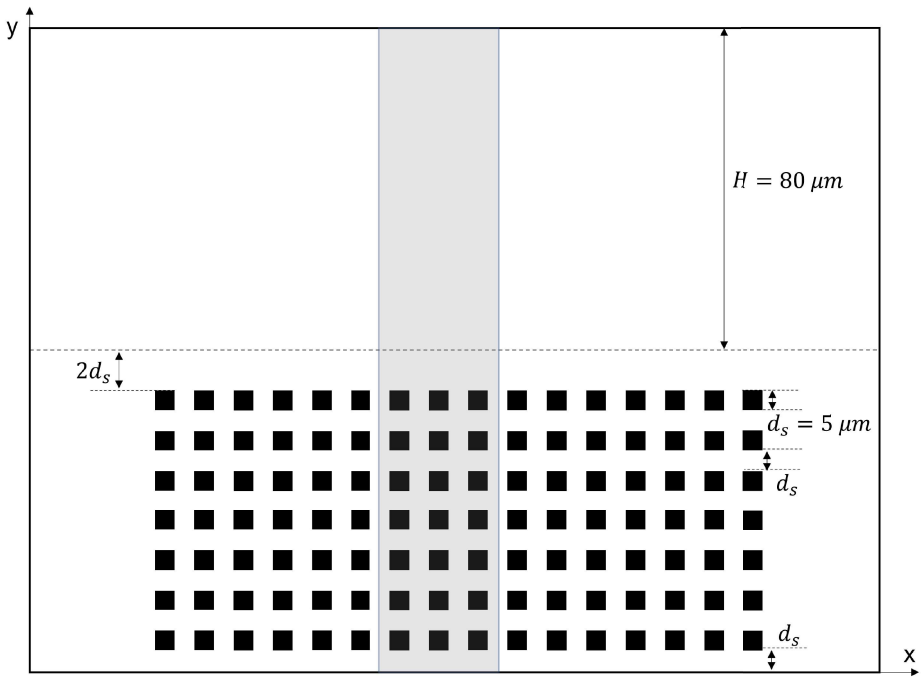


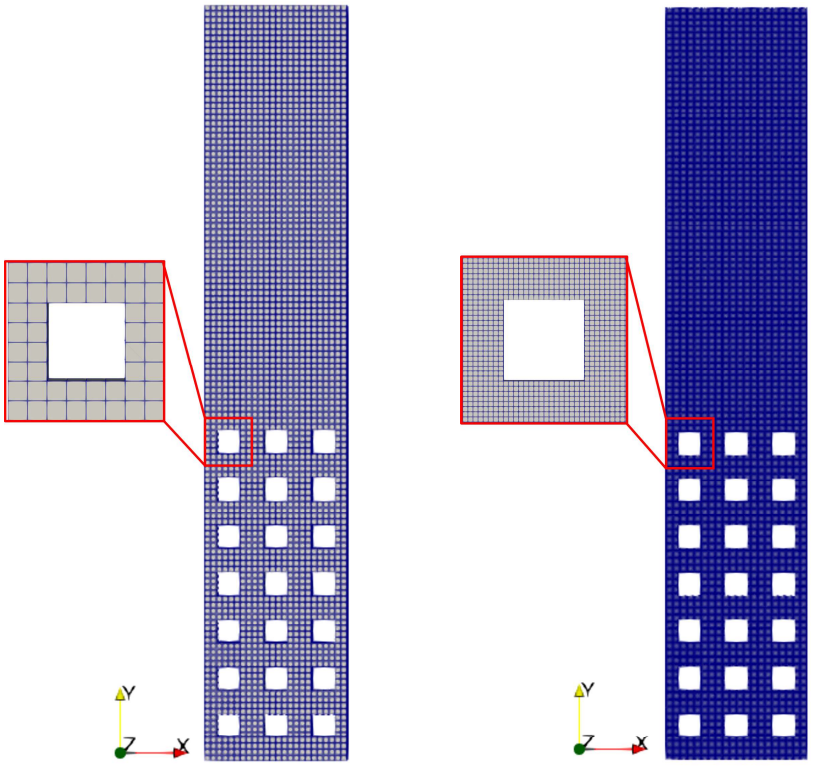
Figure 2.3: *Schematics of Beavers & Joseph problem*

2.2.2 Meshing

Meshing of the domain was performed using a Cartesian grid in Pointwise software. Initially, the original porous channel problem was meshed in Pointwise in three-dimensions. After a

tedious meshing process the size of the mesh consisted of $\sim 85 \times 10^6$ cells and was too heavy to perform a DNS simulation on it and post-process. The fact that the porous medium is an ordered one, with periodic structures, the problem could be greatly simplified. The marked region in Fig. 2.3 represents the reduced domain that has been studied in the work. The reduced domain has the same height as the original channel and the width equal to 30 μm. The porous domain consists of $3 \times 7 \times 3$ number of cubes. This reduced problem is expected to represent the original structure and is wide enough to accommodate the REV size.

For the reduced domain, two mesh sizes were created, a coarse and a refined one. The coarse mesh consisted of 4 cells per cube, totalling to $\sim 70,000$ cells in the domain. The refined mesh consisted of 16 cells per cube, counting $\sim 4.5 \times 10^6$ cells. Cells in both meshes were uniformly distributed in the domain. Example of the coarse and refined mesh are shown in Fig. 2.4(a) and Fig. 2.4(b) respectively.



(a) Coarse mesh - 4 cells per cube.

(b) Refined mesh - 16 cells per cube.

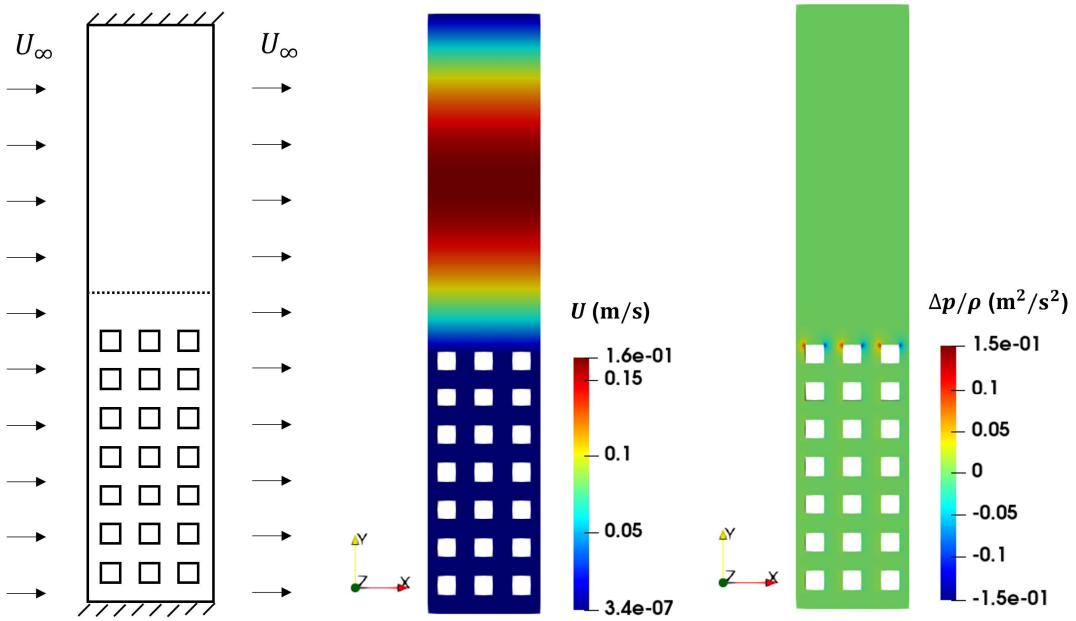
Figure 2.4: *Coarse and refined mesh used in the study.*

2.2.3 DNS simulation

In order to obtain spatially filtered permeability in the porous medium, a DNS simulation was performed to obtain the velocity and pressure fields in the domain. In order to reproduce the solution in the original channel flow problem, the boundary conditions in the reduced problem

were set as cyclic. For periodic structure as in the problem, cyclic boundary conditions allow to reproduce a fully developed flow in the reduced domain. The setup of the simulation was set as shown in Fig. 2.5(a). The top and the bottom wall, as well as the surface of the cubes maintain no-slip boundary condition. At the inlet and outlet a constant velocity $U_\infty = 0.125\ m/s$ is prescribed and the pressure gradient across the domain is computed to satisfy a constant flow rate. Simulation of the problem was performed using a simpleFOAM module in the OpenFOAM open-source software. The simpleFOAM module is a steady-state solver for incompressible flow, using the SIMPLE (Semi-Implicit Method for Pressure Linked Equations) algorithm.

Figure 2.5(b) and 2.5(c) show a 2-D slice of the obtained DNS solution. Figure to the left shows the velocity magnitude in the domain and figure to the right shows the relative pressure field normalized by density. It can be seen from Fig. 2.5(b) that the velocity in the free flow region resembles a Poiseuille flow profile, with maximum magnitude of around 0.16 m/s, slightly higher than the inlet velocity due to flow acceleration in the confined region. Velocity in the porous region is non-zero, but very small compare to the core region. Relative pressure shown in Fig. 2.5(c) shows distinct regions of over-pressure and under-pressure in the first row of cubes. Overall, the DNS solution looks physical and can be further processed using the filtering techniques.



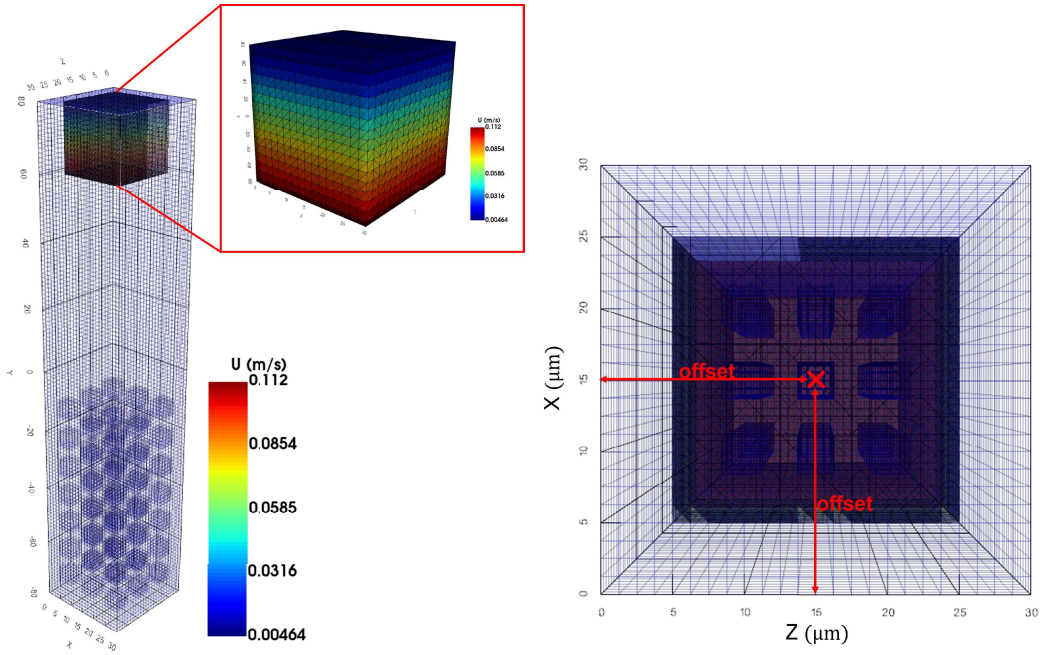
(a) DNS simulation setup. (b) Velocity magnitude field. (c) Relative pressure field.

Figure 2.5: *DNS simulation setup and solution using refined mesh.*

2.3 Post-processing DNS solution

2.3.1 Clipping REV from solution domain

In order to compute the spatially varying filtered properties along the simulation domain the geometry and the DNS solution need to be post-processed. The solution from the OpenFOAM is first opened in ParaView software and converted to a VTU format, which stands for Visualization Toolkit for Unstructured grids. The VTU file is later processed in Python language, using PyVista module. It should be mentioned that processing the solution could be done within a single Python script with ParaView and PyVista modules. However, installation of both PyVista and ParaView packages in the same environment in Python encountered errors and prevented the use of ParaView in the same environment as PyVista. This approach was left aside and the post-processing was done in two major steps, load solution into ParaView and convert to VTU format $->$ load VTU file in Python using PyVista reader and start processing.



(a) Clipping REV from domain.

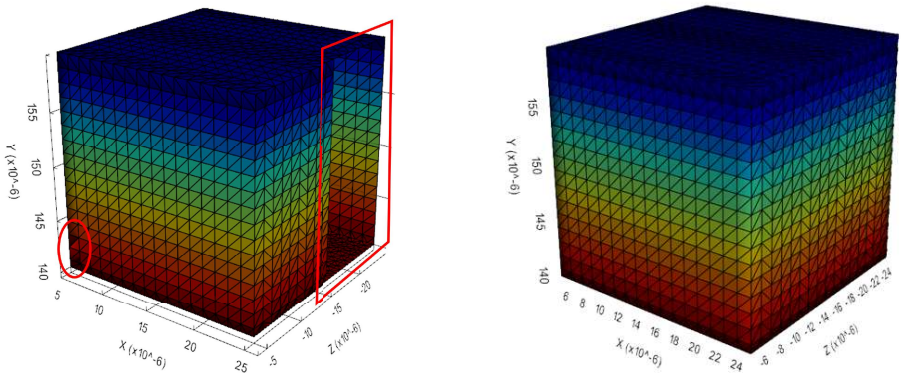
(b) Spatial position of REV in XZ plane.

Figure 2.6: *Schematics of REV clipping from the domain.*

The complete DNS solution loaded into Python and further processed is named "domain". Computation of the filtered quantities is performed using a defined REV which is set to slide along the domain to compute the spatially varying filtered quantities. The size of the REV is set here to $2l$, where $l = 2d_s$ as was described in Fig. 2.1. The actual size of the REV is 20 μm in each direction. The reason for this size of the REV is that $2l$ in each direction captures the representative part of the given porous structure. Once the size of the REV is defined, its spatial location is set within the domain. Figure 2.6(a) shows the structured

domain and the REV clipped at a certain location. As an example, the quantity in the REV is the velocity magnitude. Once the REV sliding direction is set, in this case it's the y-axis, two parameters define the spatial location of the REV centroid. The first parameter is the vertical position of the REV along the y-axis. This position is set by dividing the height of the domain minus one REV length by the requested number of samples designated as N_samp. The second parameter is the centroid location within the horizontal x-z plane, defined by the offset from the boundaries. Schematics of setting the REV centroid within the x-z plane is shown in Fig. 2.6(b).

Once the spatial location of the REV centroid is set, the boundaries of the REV are defined as one l in each direction from the centroid. The REV is clipped from the domain using PyVista $domain.clip\_box(args)$ function. By clipping the subdomain to exact boundaries of the REV, PyVista performs triangulation of the subdomain and interpolates the solution into the newly created tetrahedral cells. It should be mentioned that some peculiar behavior of the clipping function was observed. The clipped volume to exact dimensions was still containing the external boundary, that appeared to be the boundary of the volume that would be clipped if preservation of the edge cells was requested. This behavior of the clipping function complicated extraction of the actual boundaries of the clipped REV. Another unusual behavior of the clipper was that certain clipped REVs appeared to be missing external faces as shown in shown in Fig. 2.7(a). This unusual representation of the clipped volume was resolved after finding that the clipped REV contained cells with identically zero volume and extremely small volume, $\sim 1 \times 10^{-31}\ m^3$, appearing to be some artifacts from triangulation. Removing these cells from the subdomain helped to restore the actual boundary of the clipped volume and the missing faces on the external boundary as shown in Fig. 2.7(b). The exact reason for such unusual behavior of the clipper function and the affect of removal of zero volume cells is not fully clear. As a verification of the correct representation of the clipped REV after the manipulation with cell removal, the volume and boundaries of the restored REV were as expected.



(a) Appearance of external boundary and missing faces after clipping.

(b) Restored REV after removing zero value cells.

Figure 2.7: *Unusual behavior of the clipping function and restoration of the clipped volume.*

2.3.2 Integration methods

Once the REV is clipped from the domain, evaluation of the filtered quantity requires to compute numerically the convolution integral defined in Eq. 2.1. Following a performed study, integration of the numeric data in the REV was identified as the crucial one to obtain the accurate and smooth filtered results. In this study three integration methods were explored, leading to different levels of accuracy and challenges.

Riemann sum

The Riemann sum numerical integration is the simplest approach and requires minimum work on the sampled data in the REV. Riemann integration for each of the evaluated quantities in the problem, such as porosity, velocity and pressure is evaluated using Eqns. (2.6) below

$$\langle \phi(\boldsymbol{x_j}) \rangle \approx \sum_{i=0}^{i=N} \gamma(\boldsymbol{r_i}) m(\boldsymbol{r_i} - \boldsymbol{x_j}) \Delta V_i, \tag{2.6a}$$

$$\langle \boldsymbol{u}(\boldsymbol{x_j}) \rangle \approx \sum_{i=0}^{i=N} \gamma(\boldsymbol{r_i}) m(\boldsymbol{r_i} - \boldsymbol{x_j}) \boldsymbol{u}(\boldsymbol{r_i}) \Delta V_i, \tag{2.6b}$$

$$\langle p(\boldsymbol{x_j}) \rangle \approx \sum_{i=0}^{i=N} \gamma(\boldsymbol{r_i}) m(\boldsymbol{r_i} - \boldsymbol{x_j}) p(\boldsymbol{r_i}) \Delta V_i, \tag{2.6c}$$

where N is equal to the number of cells in the REV, $\boldsymbol{r_i}$ points to the center of each cell, $\boldsymbol{x_j}$ is the centroid of the REV and ΔV_i is the volume of each cell. Implementation of the Riemann sum approach is simple as the clipped REV already contains all the necessary information, such as quantities in each cell center and cell centroids and volumes, to compute the filtered values. The summation is performed over each cell in the clipped REV. The solid regions (cubes) do not contain any cells inside, as the flow solution was obtained in the gas phase region and hence, the value of the phase indicator $\gamma(\boldsymbol{r_i})$ is set to 1 for each existing cell and the summation is trivial. The accuracy of this approach, depends on the number of cells in the domain. The more refined is the mesh, the more accurate is the integration.

Monte Carlo

Monte Carlo integration is based on random distribution of a large number of sampling points inside the domain and evaluation of the integral as the expected value of the distribution. For a uniform distribution of Monte Carlo (MC) points in a three-dimensional domain, the estimation of the convolution integral is shown below

$$\langle f(\boldsymbol{x_j}) \rangle \approx \frac{V}{N_x N y N_z} \sum_{i=1}^{i=N_{MC}} \gamma(\boldsymbol{r_i}) m(\boldsymbol{r_i} - \boldsymbol{x_j}) f(\boldsymbol{r_i}), \tag{2.7}$$

where V is the volume enclosed within the REV envelope, i.e. for a cubical REV the volume is simply a^3, where a is the side of the box, $N_x,\ N_y,\ N_z$ is the total number of MC points in each direction and $N_{MC} = N_x + N_y + N_z$.

To evaluate this integral, several pre-processing steps need to be performed. First, the number of MC points in each direction is defined in the beginning of the computation. Then, location of each point in a 3-D space is sampled from a distribution based on the provided boundaries of the clipped REV. After the points are placed, it is required to identify what points fall in the void region and what points fall in the solid region to be able to define the phase indicator function. This step is easily performed by using the PyVista function $domain.find\_containing\_cell(points)$, that returns non-negative indices of cells containing the sampled MC points and -1 for points that didn't fall into any physical cell (solid regions with empty data space). With this approach, the MC points that fall inside the gas phase are easily identified. Figure 2.8 shows the REV clipped in the porous region and the MC points placed in the volume using a uniform distribution. The gray points are located in the void region and the red points are located in the solid region.

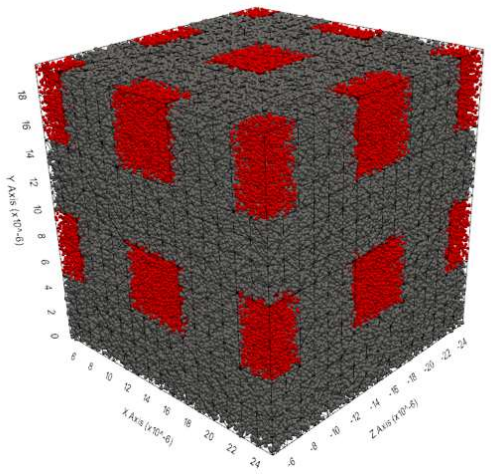


Figure 2.8: *Schematics of REV with distributed MC points. Gray - void, red - solid regions.*

The third step in evaluation of the integral is the interpolation of the flow quantities, such as velocity and pressure onto the MC points in the void region. The interpolation procedure will be separately discussed in the end of this section. After the identification of the flow phase MC points and interpolation of the flow quantities, the filtering is performed using Eq. 2.7. Accuracy of the Monte Carlo method depends on the number of distributed points and interpolation accuracy. For the infinite number of MC points, the integration is exact. However, the interpolation of the data introduces additional level of uncertainty and accuracy of the Monte Carlo integration needs to be inspected.

Quadrature rule

The quadrature rule integration is an approximation of the integral of a function as a weighted sum of function values at specified points (quadrature points) within the domain of the integration. Equation 2.8 presents the general form of approximation of a one-dimensional definite integral using the quadrature rule

$$\int_a^b f(x)dx \approx \sum_{i=1}^{i=N} f(q_i)w_i, \tag{2.8}$$

where w_i are the weights, q_i are the quadrature points and N is the number of quadrature points in the integrated domain. The quadrature points and weights are generally chosen using various techniques and depend on the shape of the domains of integration. One of the common methods is known as the Gauss-Legendre quadrature rule. For this method, the integral is exact for polynomials of degree $2n-1$ or less and it is an accurate approximation of the integral, if $f(x)$ is well-approximated by a polynomial of degree $2n-1$ or less.

The implementation of the quadrature rule integration in the study was done using a package quadpy in Python. The package handles complex-, vector-, matrix-valued integrands, and "intervals" in spaces of arbitrary dimension. In the problem, the clipped REV is discretized into tetrahedral cells and the integration can be performed on a cell to cell basis. The computed values of the integral over each cell are then summed to obtain the integral of the complete REV domain. Quadpy package allows to chose an optimal scheme for the integration for specific domain shape and degree of the approximating polynomial. For example, for polynomial of degree 5 for a tetrahedral cell, the scheme choice would be: $scheme = quadpy.t3.get\_good\_scheme(5)$. Once the scheme is obtained, the integration over a specific tetrahedral cell would be as in the equation below

$$f = \text{lambda } x : \text{func}(x[0], x[1], x[2], \text{var}), \tag{2.9}$$

$$\text{value} = \text{scheme.integrate}(f, [n_1, n_2, n_3, n_4]), \tag{2.10}$$

where lambda is a Python anonymous function, $x[0],\ x[1],\ x[2]$ are the vectors of coordinates of the quadrature points in each dimension, func is the interpolation function that returns the value of the requested variable var at the quadrature point and n_i are the node coordinates of the tetrahedral cell. In computation, each tetrahedral cell contains a certain number of quadrature points depending on the degree of approximating polynomial at which the value of the flow quantities needs to be interpolated. The quadrature rule integration is not dependent on the general shape of the integrated REV, but on the shape of the discrete cells. The quadpy package contains a number of algorithms for different types of cell shapes.

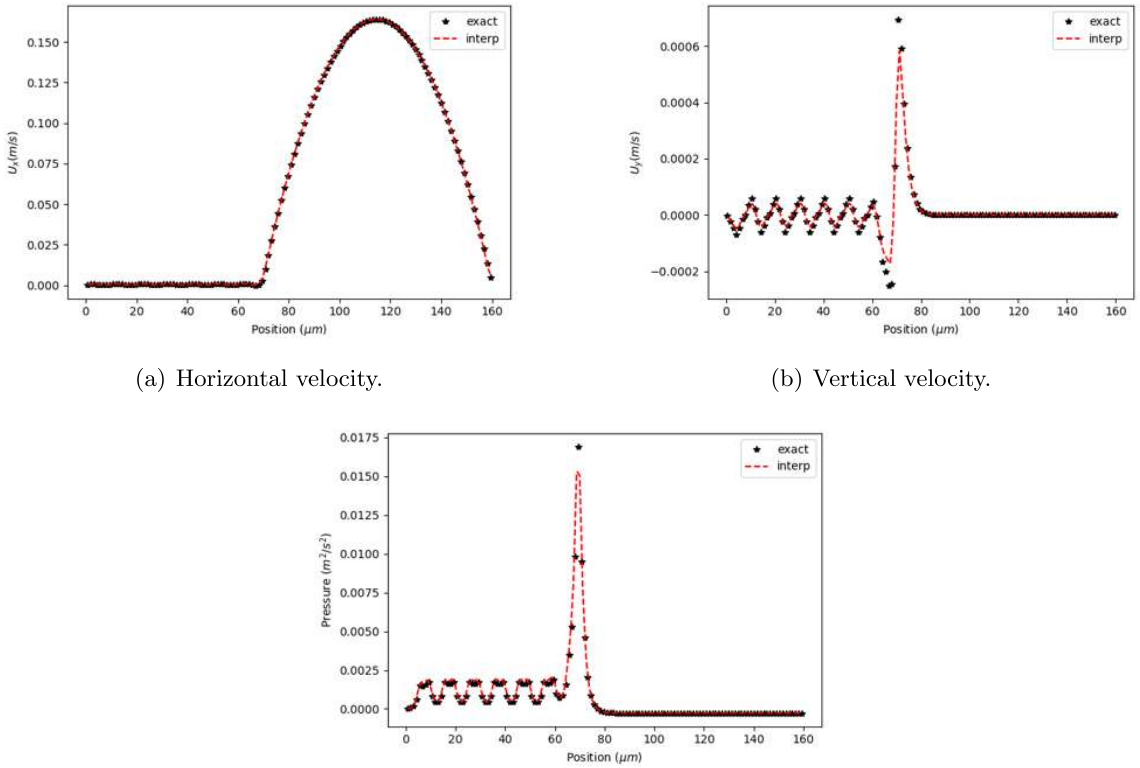
2.3.3 Interpolation methods

Interpolation accuracy is another important aspect in filtering the flow quantities using the Monte Carlo and Quadrature rule integration methods. In a three-dimensional space, the interpolation is complicated and the number of available interpolation techniques is reduced. The simplest interpolation method is the nearest neighbor method that assigns the value to the interpolated point based on the value at the nearest cell. This approach would be very inaccurate if the number of interpolated (requested) points is much larger than the number of cells. Additional interpolation methods are trilinear interpolation, Gaussian interpolation and Radial Basis Functions.

PyVista package already provides convenient way for interpolation of data. The two common methods are $requested.interpolate(domain, args)$ and $requested.sample(domain)$. The interpolation method performs Gaussian interpolation, provided the radius of the point

cloud, sharpness of the Gaussian kernel and strategy for cases how to treat an empty local neighborhood. The sample method is a more straightforward one, that re-samples array data from a source mesh onto the requested points. To evaluate the performance of the two interpolation methods, comparison was made between the exact values extracted along a cell line through the domain and the interpolated values given the number of points along the same line.

Figure 2.9 shows comparison between exact and interpolated values of horizontal and vertical velocity and relative pressure, where the number of known (exact) data points is 128 and the number of requested (interpolated) points is 500. The comparison was performed on the coarse type of mesh, shown previously in Fig. 2.4(a). The interpolation method used here is *requested.sample*(*domain*). It can be seen that the interpolation curve follows quite accurately the exact data points in all three graphs, except the sharp change regions in vicinity of the permeable wall in the vertical velocity and relative pressure values.



(a) Horizontal velocity.

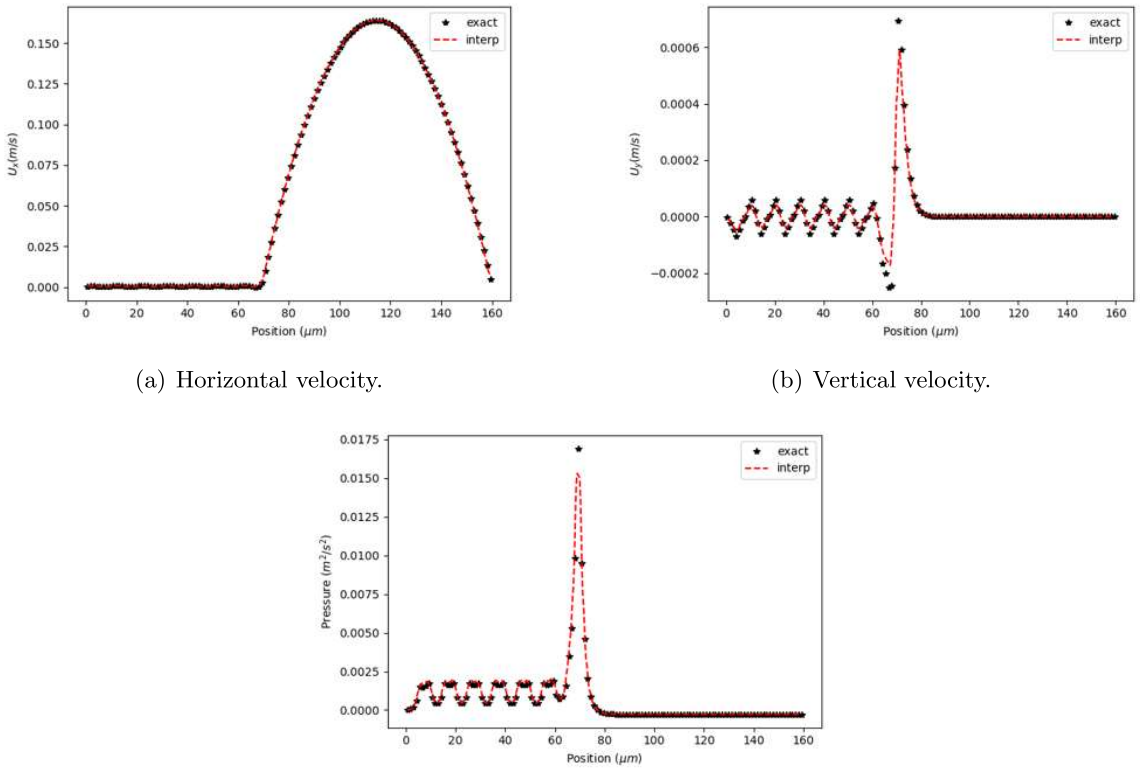
(b) Vertical velocity.

(c) Relative pressure.

Figure 2.9: *Interpolation method: sample. Number of requested points: 500. Number of known points: 128*

Figure 2.10 shows comparison with the same setup as before, except that the interpolator is the Gaussian with sharpness = 2 and radius = 1.5 μm. It can be seen that the exact data points are also followed quite accurately by the interpolation curve for all three quantities. A similar underestimation of the data points is observed in the vicinity of the permeable wall.

The agreement between the sample and Gaussian interpolation methods is quite similar.



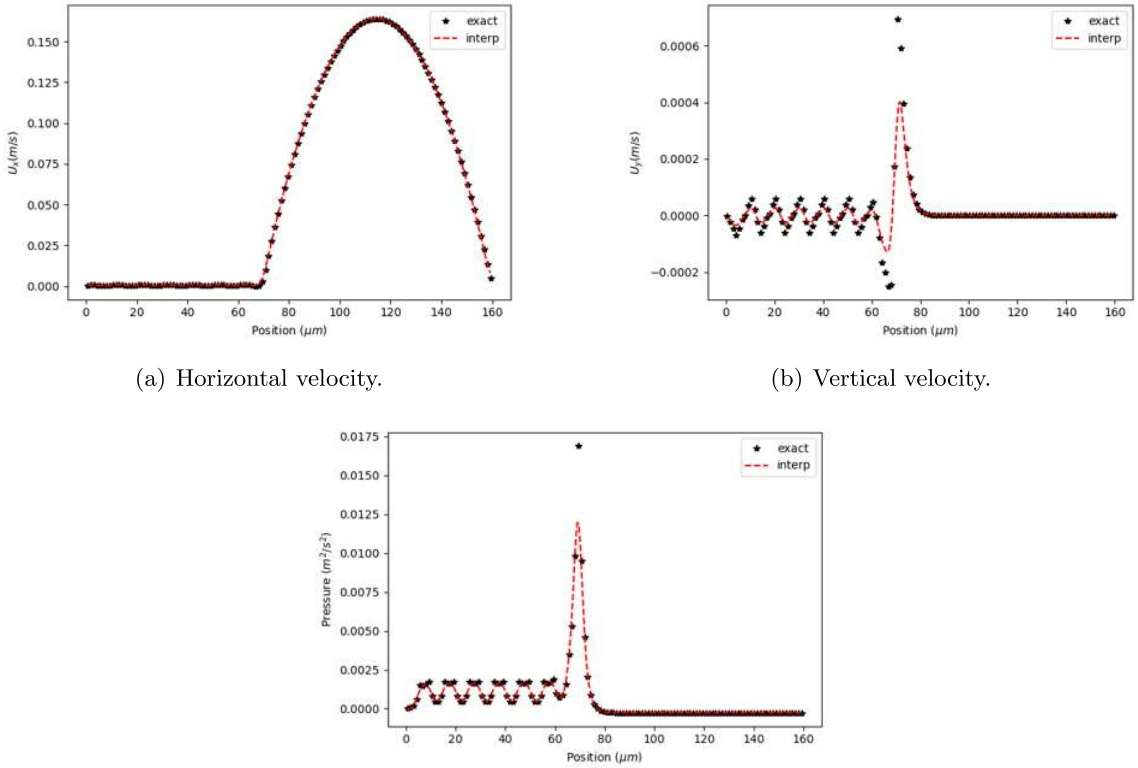
(a) Horizontal velocity.

(b) Vertical velocity.

(c) Relative pressure.

Figure 2.10: *Interpolation method: Gaussian(sharpness=2, radius = 1.5 μm). Number of requested points: 500. Number of known points: 128*

Since the Gaussian interpolation method depends on the preset parameters such as sharpness and radius of the kernel, additional comparison is presented for the same sharpness = 2, but different radius, equal to 3 μm. The results are shown in Fig. 2.11. It can be seen that increase of the kernel radius caused smearing of the interpolated data in the vertical velocity and relative pressure values. Increasing further the kernel radius would smear the interpolated data even more until the wavy data in the porous region (left) would be completely flattened. This behavior of the Gaussian interpolator would require a preliminary testing for each new mesh and solution to determine the optimal radius of the kernel and its sharpness. The sample method, shown previously, appears to be more robust in estimating the data at the interpolated points and this method was chosen for all later cases in this study.



(a) Horizontal velocity.

(b) Vertical velocity.

(c) Relative pressure.

Figure 2.11: *Interpolation method: Gaussian(sharpness=2, radius = 3 μm). Number of requested points: 500. Number of known points: 128*

Chapter 3

Results

Presentation of results is organized in two sections. In the first section, preliminary verification results are shown. These results were obtained using two filter types (top-hat and cellular) running on numerically generated 2-D and 3-D channel geometries. Porosity results are compared to an analytical solution from Breugem et al. [2004]. In the second section, filtering of the DNS results, including the mesh and the flow solution, is presented and discussed. The results are shown for three integration methods introduced earlier 2.3.2.

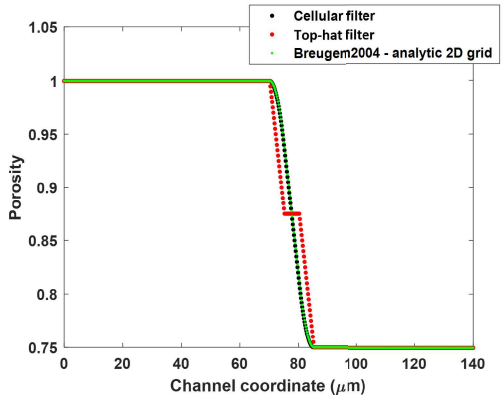
3.1 Verification study using numerically generated channel geometry

Channel geometry, shown in Fig. 2.3, was numerically generated for 2-D and 3-D configurations to perform a verification study of the filtering approach. In the 2-D configuration, the solid regions are represented by squares and in 3-D configuration by cubes, replicating exactly the problem studied by Breugem et al. [2004]. The domain in each configuration was discretized using a Cartesian, uniformly distributed grid with 21 cells per cube. Effective porosity of each configuration was filtered using cellular and top-hat filters, defined on the domain shown in Fig. 2.1. Sliding of the filter was performed in the vertical, y-axis direction from the free flow region into the porous. Sampling of the REV box was done on cell to cell basis along a line.

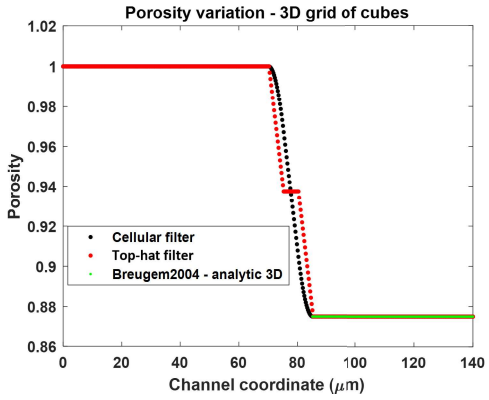
Figure 3.1(a) shows the profile of effective porosity in a 2-D channel. The results are shown for cellular and top-hat filters and compared against analytical expression from the study of Breugem et al. [2004]. The value of effective porosity in the free flow region is 1 and as the REV edge reaches the permeable wall the effective porosity starts to decrease as more and more solid region is being averaged by the filter. The transition region extends to $3d_s$ or 3 cube sizes until the effective porosity reaches a constant value of 0.75 in the fully porous region. It can be seen that cellular and top-hat filters align with each other in the free flow and porous regions, but show some misalignment in the transition zone. The discrepancy corresponds to the difference in the filtering weights used by each filter. The top-hat filter applies equal weight to all samples in the REV, while the cellular filter applies weights decreasing to zero away from the center of the REV. This is the reason why we see a steeper decrease in porosity predicted by the top-hat filter. The top-hat filter also shows a sudden flattening of the curve for a distance of one cube/void size. The reason for the curve

flattening is explained later in the DNS section results. The analytical expression from the reference study is based on a definition of a cellular filter approach and shows here excellent agreement with the current study.

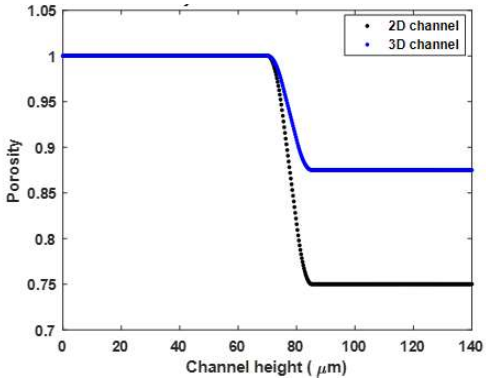
Figure 3.1(b) shows the effective porosity for a 3-D channel geometry, computed using cellular and top-hat filters. Similar behavior of the porosity curve is seen in this case. Following the transition region that also extends to $3d_s$, porosity reaches a constant value of 0.875, higher than in the 2-D case. The analytical expression for a 3-D case was provided only for the porous region and it can be seen that the results align perfectly with the data. Finally, Fig. 3.1(c) shows comparison between 2-D and 3-D porosity computed with a cellular filter. A clear difference between the volumetric and area porosity can be seen in the slope and in the final value.



(a) 2-D channel. Comparison of cellular and top-hat filters with reference study.



(b) 3-D channel. Comparison of cellular and top-hat filters with reference study.



(c) Comparison of cellular filter for 2-D and 3-D channels.

Figure 3.1: *Filtering verification. Comparison of porosity for two types of filters and two types of channels.*

3.2 Filtering the DNS solution

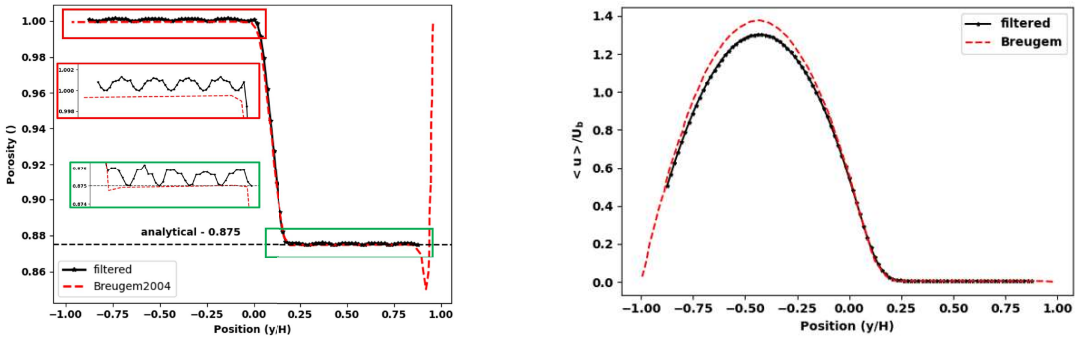
The DNS solution for a refined grid was shown previously in Fig. 2.5. Here the filtering results are shown for three integration methods. In all integration methods a "sample" interpolation method was used as described in Sec. 2.3.3.

Riemann sum

Figure 3.2 shows verification of the effective porosity and horizontal velocity using a cellular filter computed on the coarse mesh solution (Fig. 2.4(a). The number of REV clips along the channel height is 100. The x-axis shows position of the REV centroid normalized by the half channel height. The comparison is done against the numerical results from the work of Breugem et al. [2004] after digitizing the data from the plots. It can be seen that the general trend of the porosity in Fig. 3.2(a) aligns very well with the reference data, except a certain level of noise present in the filtered data shown in the zoomed in regions and the edge of the porous region (right). The edge disagreement is related, presumably, to the way the REV box was moved close to the wall. To get a sudden decrease and then increase in the effective porosity, the size of the REV should have been shrunk as the box approached the wall, so that initially the proportion of solid in the volume increases as the box shrinks and looses void volume and then, the proportion of the solid decreases back as more and more volume captures the remained void at the bottom of the channel. The noise in the filtered data is related to accuracy of the Riemann sum approach. For arbitrary shapes of the discrete cells, the accuracy of the method will increase for higher number of cells. It was observed for porosity computation (in the preliminary study and generated in Pointwise meshes) that if the discrete cell shapes were cubes of equal size, the Riemann integration would be exact. In the current case, even though the original channel mesh is Cartesian and uniform, clipping of the box by PyVista module essentially converts it to an unstructured mesh consisting of tetrahedral elements of different sizes. And thus, the accuracy of the Riemann sum method depends on the refinement level of each specific REV box.

Comparison of the normalized horizontal velocity profile with the reference data is shown in Fig. 3.2(b). It can be seen that velocity profile follows a Poiseuille flow distribution in the free flow region and gradually decays to a very low, but non-zero value in the porous region, indicating a slip condition at the permeable wall. The data agrees well with the reference except some underestimation at the peak, which is attributed to the different types of solvers between the current and reference study.

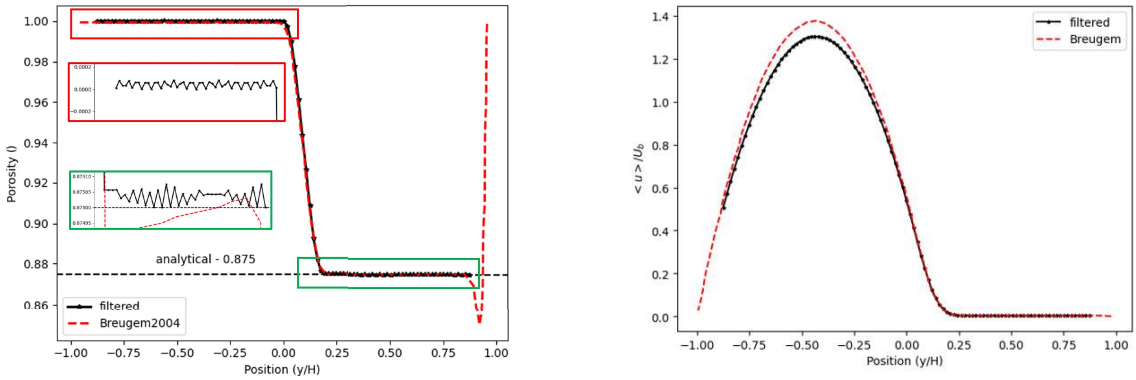
Verification of the Riemann sum approach was also done for the refined mesh case, for the same cellular filter and number of REV clips as in the coarse case. Figure 3.3(a) shows the porosity profile, compared against the same reference data and Fig. 3.3(b) shows the comparison of the normalized horizontal velocity profile. The curves align similarly well the the reference data. It can be seen that the level of noise shown in the porosity profile is much smaller and reduced by about an order of magnitude compare to the coarse mesh case. The increase in the refinement level between the two meshes is a factor of 4 in each dimension. It can be noticed, that refinement of the mesh did not improve the agreement of the normalized velocity with the reference data, implying that the difference could be attributed to the different solver types and not different mesh refinement levels. Overall, the agreement of the filtered porosity and the DNS solution with the reference data is good,



(a) Porosity. (b) Normalized horizontal velocity.

Figure 3.2: *Riemann sum method. Coarse grid. Verification against a reference study from Breugem et al. [2004].*

but requires additional improvement in the accuracy of the integration to obtain smoother profiles required for computation of derivative quantities.

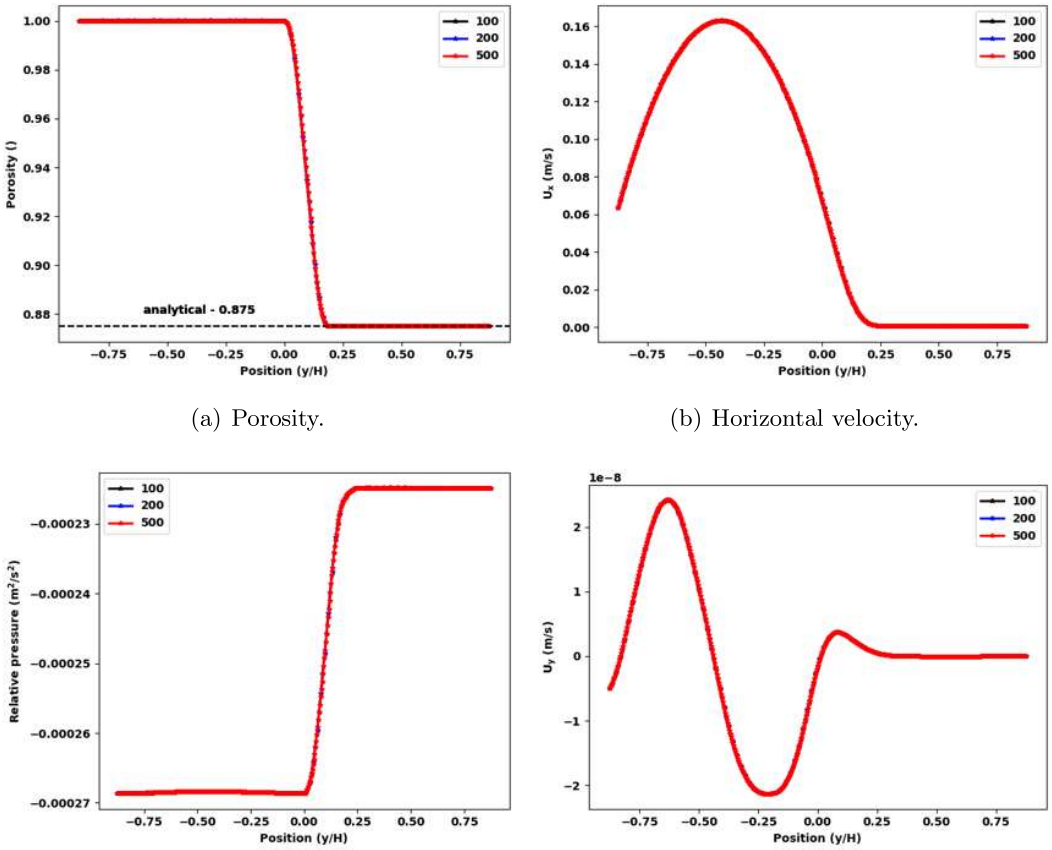


(a) Porosity. (b) Normalized horizontal velocity.

Figure 3.3: *Riemann sum method. Refined grid. Verification against a reference study from Breugem et al. [2004].*

Following the verification study, consistency of the REV clipping approach is explored based on the refined case. Figure 3.4 shows the filtered profiles for porosity, horizontal and vertical velocity and relative pressure based on the different number of clipped REV boxes, i.e. 100, 200 and 500. It can be seen that all three profiles in each figure align perfectly with each other. This confirms the consistency of the clipping procedure. Besides the two already explored profiles of porosity and horizontal velocity, relative pressure and vertical velocity in y-axis direction are shown in the figure. Relative pressure increases between the free flow and porous regions. A clear transition zone can be seen also in this case. The vertical velocity shown in Fig. 3.4(d) is very small, around 7 orders of magnitude smaller than the horizontal

velocity in the free flow region.



(a) Porosity.

(b) Horizontal velocity.

(c) Relative pressure.

(d) Vertical velocity.

Figure 3.4: *Riemann sum method. Refined grid. Solution comparison for three different number of REV samples (N_{samp}).*

Finally, the Riemann sum integration section is concluded by comparison of the cellular and top-hat filters based on the refined grid case. Figure 3.5 shows the porosity profile zoomed to the transition zone. As in the preliminary study, we see that the top-hat filter leads to a flat region at the center of the transition zone. The reason for this flattening is explained in the schematics in the figure. The filter window as it starts sliding through the ordered porous region, "sees" the same structure over the first rows of cubes. Since the top-hat filter assigns equal weights to each point in the REV, the effective porosity value appears to be the same until the window starts capturing additional solid region in the second rows of solids. This is the reason that top-hat filter is not recommended for use in the ordered porous media. For disordered porous media this problem should not appear as the solid region captured by the REV will be always different between the consecutive steps by the REV window.

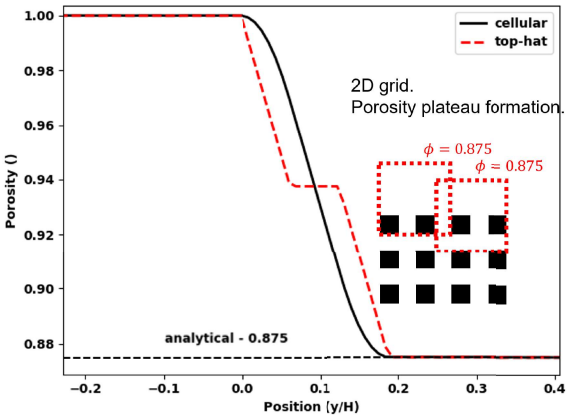
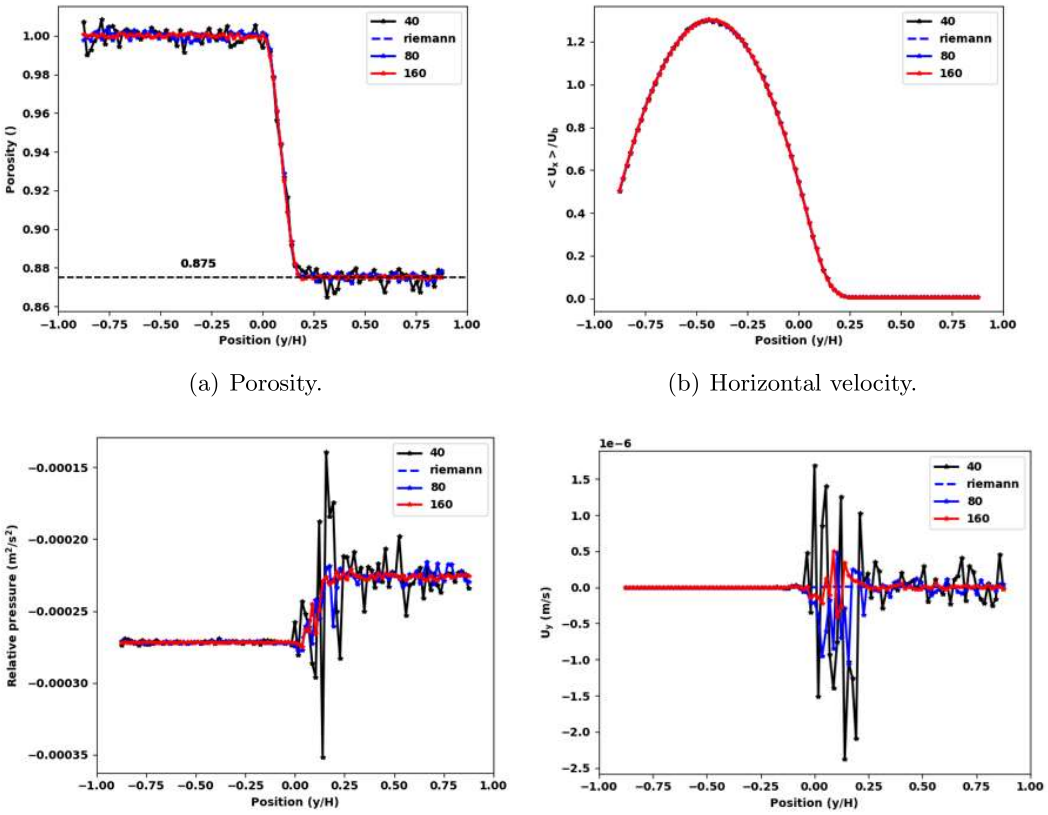


Figure 3.5: *Refined grid. Comparison of porosity for cellular and top-hat filters.*

Monte Carlo

The Monte Carlo integration technique is based on distribution of a large number of sample points in the domain to estimate the value of the integral. In this study, two technique for distribution of the MC points were explored. In the first technique, the MC points in the REV domain are placed based on a uniform distribution. After every clipping of the REV, the same number of MC points is randomly distributed between the boundaries of the box. The example of uniform distribution of MC points in the REV was shown previously in Fig. 2.8. To explore the performance of the method, filtered quantities were computed for three different number of MC points in certain direction using a coarse grid solution. Figure 3.6 shows the obtained filtered results for porosity, velocity and relative pressure for 40, 80 and 160 MC points distributed in each direction. The corresponding total number of points in the REV is 64×10^3, 512×10^3 and 4.096×10^6. The filtered profiles are compared against Riemann solution for the same grid and clipping direction. The number of clipped REVs is 100. It can be seen from the figure, that quite high level of noise is present in the filtered data. For 40 MC points the level of noise is the highest and as the number of MC points increases, the profiles get smoother and better resemble the reference curve from the Riemann sum solution. The accuracy of the filtering appears to depend on the magnitude of the sampled data. Vertical velocity, that has the lowest magnitude, many orders of magnitude lower than the horizontal velocity and shows the least accurate prediction of the reference data even using 4.096×10^6 MC points. The pressure curve is approximated considerably better using 160 points, but still shows unacceptable level of noise relative to the Riemann integration data. For much higher number of MC points, the accuracy of the integral approximation will increase and potentially converge to the true value, but the computational cost of the method is much higher than of the Riemann one, as the total number points grows exponentially.

The second approach of distributing the MC points was based on linear spacing of the given number of points in the domain. Distribution of the points using this approach resembles a structured, uniform grid that produces accurate results in the Riemann sum integration. This approach is a not a strictly Monte Carlo one, as the points are not sampled from any random distribution, but are placed at a deterministic location in the REV domain. The procedure of evaluation of the integral though, remains the same. Figure 3.7 shows the filtered



(a) Porosity.

(b) Horizontal velocity.

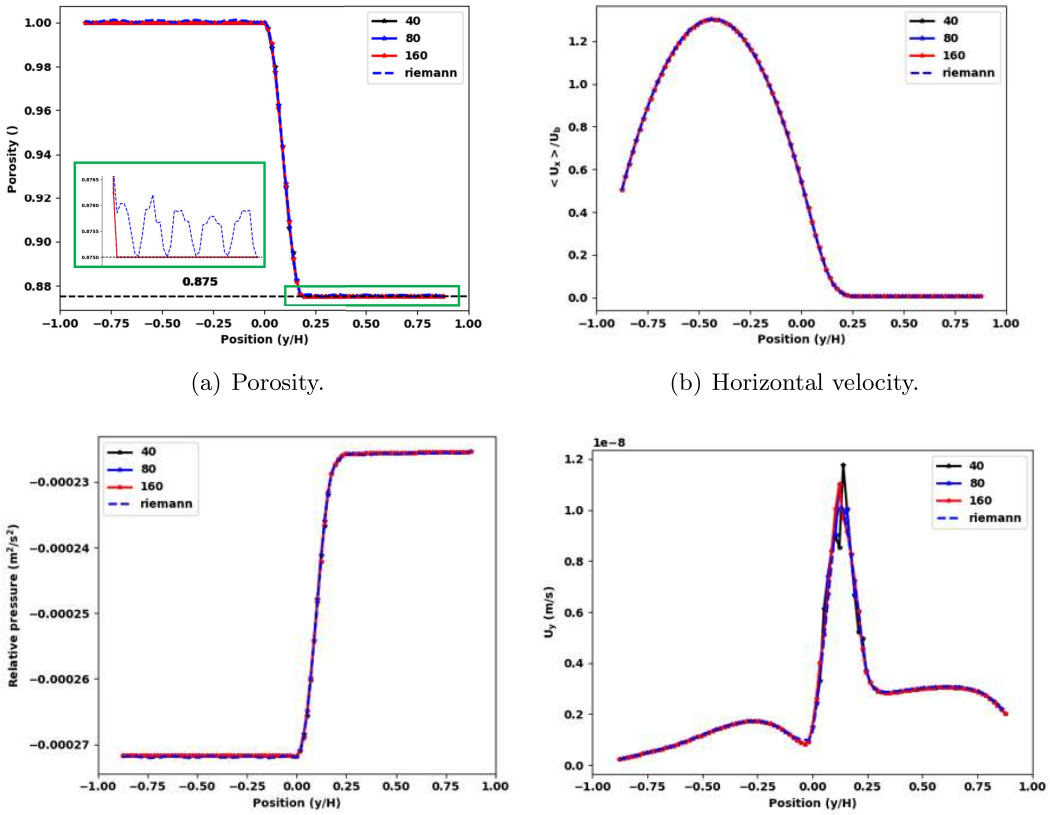
(c) Relative pressure.

(d) Vertical velocity.

Figure 3.6: *Monte Carlo. Coarse grid. Uniform distribution of MC points. Comparison against Riemann sum method for different number of MC points.*

profiles for 3 different number of distributed points, 40, 80 and 160 in each direction. It can be seen, that the integration accuracy is drastically increased, showing a perfect match of the porosity, horizontal velocity and relative pressure with the Riemann sum solution. The oscillation level in porosity data, for example, is essentially zero. The accuracy of recovering the vertical velocity profile is less accurate at the peak, which falls somewhere in the beginning of the porous region. The sharp peak value makes it harder to accurately recover the data during interpolation and is prone to noise.

The accurate prediction of the filtered quantities by the linear distribution method of MC points is not perfect though. The accuracy of the method depends on the geometry of the porous medium or specifically, the proportion of the distributed points that fall in the void and solid regions. For clarity, porosity of the material is determined by the volume of void, divided by the total volume. The void and solid regions in the approach are sampled by the distributed points, meaning that the relative proportion of the void to the total volume is determined by how many points fall into the solid and void regions. For example, if the ratio of void to total volume is 3/4, it means that 3m number of points must have fallen into the



(a) Porosity.

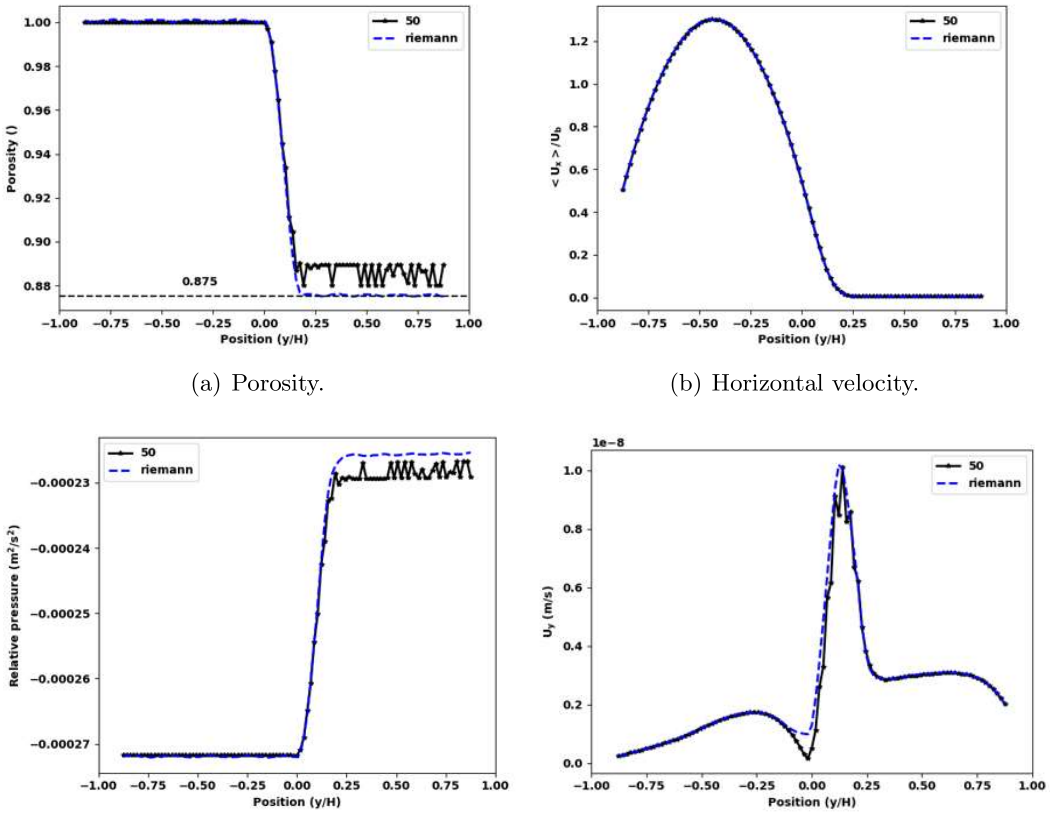
(b) Horizontal velocity.

(c) Relative pressure.

(d) Vertical velocity.

Figure 3.7: *Monte Carlo. Coarse grid. Linear distribution of MC points. Comparison against Riemann sum method for different number of MC points.*

void and 1m must have fallen into the solid. However, if more points fell into the void than into the solid or vise versa, but the total number of points is the same 4m, the physical ratio of void to total volume will still be the same, but the ratio based on the relative count of points in the void and slid will lead to a wrong value of porosity. The linear spacing of the points might lead exactly to this problem. This behavior is demonstrated in Fig. 3.8. In this case 50 MC points were used and we see that prediction of porosity in the cubes region is totally off from the expected value. Similar effect we see in the pressure profile and very small, but present offset appears in the velocity profiles. Moreover, for this specific distribution of voids and solids in the porous region, the number of MC points that will predict accurate porosity must be dividable by 4. This dependence should be related to the fact that there is 4 blocks, 2 void and 2 solid in each REV. Unfortunately, the method of linear distribution of the MC points appears to strongly depend on the geometry of the porosity and appears to be good for only particular problems of ordered porous media.



(a) Porosity.

(b) Horizontal velocity.

(c) Relative pressure.

(d) Vertical velocity.

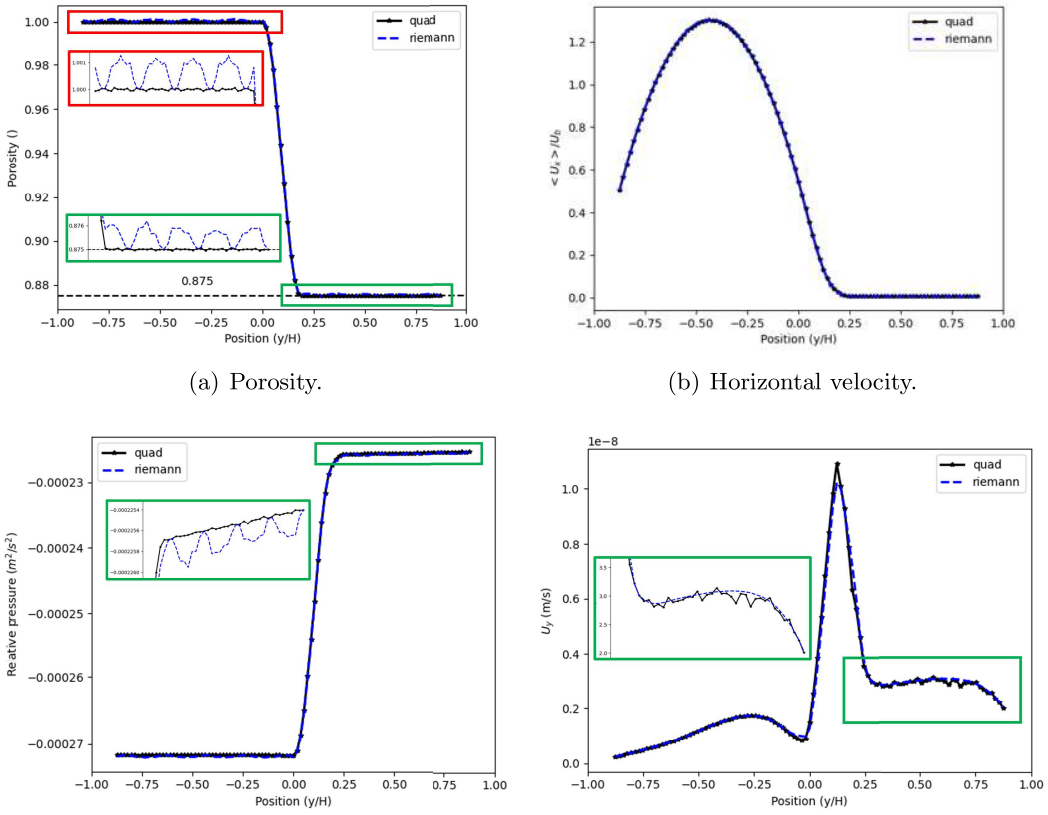
Figure 3.8: *Monte Carlo. Coarse grid. Linear distribution of 50 MC points. Comparison against Riemann sum method.*

Quadrature rule

The last approach explored in this study is the quadrature rule integration. This approach is more sophisticated than the two previous and relies here on a use of Python quadpy package that provides methods for different shapes of the integrated domains. In this study the clipped REV domain consisted of tetrahedral elements and only this type of element was tested here. The package contains methods for additional shapes such as hexahedras. Using this method, integration of the REV domain is reduced to integration on each cell individually and summation of the integrated values. The integral over each cell is evaluated using a number of quadrature points, depending on the requested degree of the approximating polynomial. In this study, the degree was set to 5 and the integration method was chosen with $method = quadpy.t3.get\_good\_scheme(5)$.

The studied case was again the coarse grid case, with 100 clipped REVs along the domain and the interpolation method is the same as in the Monte Carlo method. The results are compared against Riemann sum integration. Figure 3.9 shows the four profiles for porosity,

two velocity components and relative pressure. It can be seen that the method shows a decent level of accuracy for all 4 filtered quantities. The noise level is still present, but much lower than in the Riemann sum approach for coarse mesh. The vertical component of velocity is also approximated quite accurately, with some visible level of noise in the porous region. Accuracy of the quadrature rule method depends whether the measured field can be accurately approximated by a polynomial of degree 2n - 1 or less, where n is the number of quadrature points. Additional level of inaccuracy might come from the interpolation. However, it can be seen that the peak value of the vertical velocity component is approximated quite good, attributing probably to the fact that integration of each cell is performed over a number of quadrature points at which the solution is interpolated. This should increase the accuracy. Finally, it should be mentioned that this integration method is quite computationally expensive, as for each cell the method requires interpolation on a number of quadrature points, thus increasing the total number of operations. Also, the method was not vectorized in the current implementation and it was impossible to test it on a refined mesh.



(a) Porosity.

(b) Horizontal velocity.

(c) Relative pressure.

(d) Vertical velocity.

Figure 3.9: *Quadrature rule. Coarse grid. Comparison against Riemann sum method.*

Chapter 4

Summary

4.1 Conclusions

In this study, a work was performed to develop a filtering methodology to derive effective material properties, such as porosity and flow quantities based on a DNS solution. The development of the methodology was based on the well known porous channel flow problem from Beavers and Joseph [1967] and a reference study from Breugem et al. [2004]. The development started from a simple verification study based on numerically generated channel geometry in two and three dimensions. The verification was performed using cellular and top-hat filters showing excellent agreement with the reference analytical data. In this study, the difference between the two types of filters was emphasized, showing that the top-hat filter should not be used for ordered porous media and further work is required for its applicability for disordered media. Finally, it was shown that porosity varies across the interface between the free flow and porous regions. For this type of studied geometry and REV size, the interface region extended to three cube sizes beyond the permeable wall.

Next, the methodology was extended to process a full case scenario, including the meshed channel geometry generated with commercial software such as Pointwise and DNS solution obtained with OpenFOAM solver. Processing of the results was done in Python using PyVista module, that allowed effective clipping of the REV volume and manipulation of data. It was identified, that numerical integration of the convolution integral used in the filtering technique was a crucial part in obtaining an accurate solution. In this study, three integration methods such as Riemann sum, Monte Carlo and Quadrature rule integration were explored. In addition, the two last techniques required data interpolation to the sample points. For that, two interpolation methods of the PyVista module such as Gaussian interpolation and re-sampling method were explored. It was found that the re-sampling method provided the most consistent recovery of the data compare to the Gaussian method that depends on the kernel radius and sharpness, that smears the interpolated data. In this study the sample interpolation method provided by PyVista was used.

In terms of the integration methods, the Riemann sum showed good agreement with the reference numerical data, but accuracy of the method highly depends on the level of refinement of the mesh. The Monte Carlo method was studied with two types of distribution of the points, uniform and linearly spaced. It appeared that the uniform distribution required increasingly high number of points to improve the accuracy beyond the Riemann sum method.

An alternative method of distributing the points with linear spacing showed excellent results when the number of points was dividable by 4. For other configurations, the method generated corrupted data and offset relative to the reference. Finally, the quadrature rule integration method showed the most promising results as even for the coarse mesh case it outperformed the Riemann sum integration in terms of accuracy, with some minor noise in the vertical velocity profile.

Overall, the generality of the developed methodology is expected to be quite high in terms of the REV clipping procedure and integration and the method should be applicable to more complex porous structures.

4.2 Future work

- The future work should include more testing of the integration methods. The Monte Carlo method with uniform distribution of the points could be tested on significantly higher number of points, probably > 1000 in each direction. The quadrature rule integration needs to be better programmed to increase the speed of the computations and needs to be tested on refined geometry. In addition, the method should be extended to other types of cell shapes, such as hexahedra.

- Integration of Paraview and PyVista library should be performed in the same Python environment. Paraview software has additional clipping method that preserves the shape of the cells, but still cuts the box to the requested boundaries. Using Paraview to clip the REV would allow to increase the computational speed, as the number of cells would remained almost the same, compare to the triangulation method in PyVista.

- The developed method should be tested on different directions of filtering the data, even though this shouldn't be too complicated. Currently the algorithm allows clipping in one of the Cartesian directions, but hasn't been extensively tested.

- Once the level of accuracy of the integration and interpolation methods is satisfactory, permeability of the medium can be readily calculated.

- Finally, the method should be tested on more complex cases, such as unstructured meshes and more complex porous structures.

- For more complex cases, requiring increased number of cells, the method needs to be parallelized. The relatively simple approach is to distribute the number of requested clipped REVs among different processors. Partitioning the REV will be more complicated.

Bibliography

Beavers, G. S. and Joseph, D. D. [1967], 'Boundary conditions at a naturally permeable wall', *Journal of fluid mechanics* **30**(1), 197–207.

Breugem, W., Boersma, B. and Uittenbogaard, R. [2004], 'Direct numerical simulations of plane channel flow over a 3d cartesian grid of cubes', *Applications of porous media* pp. 27–35.

Davit, Y. and Quintard, M. [2017], 'Technical notes on volume averaging in porous media part i: how to choose a spatial averaging operator for periodic and quasiperiodic structures', *Transport in Porous Media* **119**(3), 555–584.

Duzel, U. [2020], 'Development of universal solver for high enthalpy flows through ablative materials'.

Quintard, M. and Whitaker, S. [1994], 'Transport in ordered and disordered porous media ii: Generalized volume averaging', *Transport in porous media* **14**(2), 179–206.

Schrooyen, P. [2015], Numerical simulation of aerothermal flows through ablative thermal protection systems, PhD thesis, PhD thesis, UCL.

Whitaker, S. [1999], 'Theory and applications of transport in porous media: The method of volume averaging', *The Netherlands: Kluwer Academic Publishers* .