Wall Shear Stress Description

Yimo Wang

August 2023

Contents

1 Introduction

This system computes the wall shear stress at each vertex on a vessel mesh. Given the inputs, the system first computes a local orthonormal frame at each vertex of the vessel mesh.

> The orthonormal frame consists of 3 vectors: the axial, radial, and circumferential vectors. The axial vector is the vector tangent to the vessel in the direction of the vessel's centerline–it is parallel to the tangent of the local centerline point. The radial vector is the vector normal to the vessel mesh locally. The circumferential vector is the vector tangent to the vessel mesh, going around it–given an axial cross section of the vessel, the circumferential vector is parallel to a line tangent to the circumference of the crass section.

The system uses this orthonormal frame to compute the local wall shear stress at each vertex, as well as to compute the components of shear stress in each direction of the frame.

The **compute_and_save_wall_shear_stress()** function returns **points_map**, a bundle of data containing the coordinates, orthonormal frame, and components of shear stress along the frame for each vertex on the vessel mesh.

Inputs:

- **centerline**: The centerline of the vessel.
- **collars**: Given a centerline point, a collar is the subset of the mesh around the centerline point. The input **collars** is a list of such collars, one for

each centerline point. As such, the length of the list of centerline points is the same as the number of centerline points. The i-th member of the list is the triangulation of the mesh corresponding to the i-th centerline point. Each centerline point has exactly 1 corresponding collar. Each mesh vertex may be in multiple collars.

- **velocity_timeslices**: A list of regular LabelledGrids, one for each timeslice. The labels at each voxel of the grid is the velocity vector $\mathbf{u} = (u_1, u_2, u_3)$ sampled from the patient at that location. All velocities are given with units mm/sec.

- **velocity_derivative_timeslices**: A list of regular LabelledGrids, one for each timeslice. These grids are colocated from the **velocity_timeslices** grids, so these LabelledGrids are one cell shorter than the **velocity_timeslices**' LabelledGrids in each dimension. The labels at each voxel of the grid is a 3×3 matrix representing the velocity gradient at that location: $\begin{pmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} & \frac{\partial u_1}{\partial z} \\ \frac{\partial u_2}{\partial x} & \frac{\partial u_2}{\partial y} & \frac{\partial u_2}{\partial z} \\ \frac{\partial u_3}{\partial x} & \frac{\partial u_3}{\partial y} & \frac{\partial u_3}{\partial z} \end{pmatrix}$
The i-th element in this list correpsonds to the i-th element in **velocity_timeslices**. All velocity gradients have units 1/s.

- **surface**: A triangulation representing the surface of the mesh segmented.

- **output_directory**: A string determining the name of the output directory.

- **mu**: The physical viscosity constant. Set to 4×10^{-3} Pa-s.

Outputs:

- A .vtk database of n files, where n is the number of timeslices. Each .vtk file contains the following data for each vertex on the mesh:
  - A mesh of the structure and surface itself
  - Scalars: axial component of shear stress, radial component of shear stress, circumferential component of shear stress, the absolute magnitude of shear stress vector itself.
    The absolute magnitude of shear stress has units $\mathrm{N/m^2}$, or equivalently, Pa.
  - Vectors: shear stress, unit axial vector of orthonormal frame, unit radial vector of orthonormal frame, unit circumferential vector of orthonormal frame.
    The units of shear stress are $\mathrm{N/m^2}$, or equivalently, Pa.

Intermediate Outputs:

- **points_map**: A map where the keys are 3-dimensional Vectors representing the coordinates of a point in space and the values are **Point** objects corresponding to the key coordinates. This map helps to access information bundled into the **Point** objects, given the coordinates of the vertex in question.
  As a support mechanism for this map, a custom comparator **compVectorLex** is defined.

How to Use

All inputs are read or computed in other parts of Dan's bash script. To run computations for wall shear stress, run the bash script's analyze vessels section as per usual and wait until "DONE" is printed in the terminal.

This function is called in MRI/vessel_geometry_analysis_source_main.cpp. All required inputs are computed before entering the wall shear stress functions.

2 Structure

The Point class:

Each **Point** object corresponds to one vertex on a mesh. The **Point** class bundles together several pieces of information:

- **coords**: The (x, y, z) coordinates of the vertex in space, represented as a 3-dimensional Vector. In the default constructor, this is initialized to a Vector of 0's. The second constructor takes in a 3-dimensional Vector representing the coordinates as a parameter and initializes a **Point** object with the specified coordinates.
- **radial**: The local unit radial vector at this vertex, relative to the mesh surface, represented as as a 3-dimensional Vector. This is initialized to a Vector of 0's and is computed in the **compute_frame()** function.
- **tangent**: The local tangent vector at this vertex, relative to the mesh surface, represented as a 3-dimensional Vector. This is the vector parallel to the centerline of the vessel at this point. This is initialized to a vector of 0's and is computed in the **compute_frame()** function. Within the **Point** class, all uses of this field refer to it as "tangent," due this field being computed using the vectors tangent to the centerline. However, later uses of this field with respect to the **Point** object itself refer to it as "axial," due to the use of this vector as one of the vectors of the local orthonormal frame.
- **radial_aggregate**: This field is used as an intermediary field to calculate **radial**. This field is a 4-dimensional vector, where the first 3 elements represent a sum (x, y, z) of vectors added, and the last element is the number of vectors added. See **compute_frame()** for more.
- **tangent_aggregate**: This field is used as an intermediary field to calculate **tangent**. This field is a 4-dimensional vector, where the first 3 elements represent a sum (x, y, z) of vectors added, and the last element is the number of vectors added. See **compute_frame()** for more.
- **shear_stress_over_timeslices** A list of shear stress vectors at this vertex, over time. The number of elements in this list is equal to the number of timeslices. The i-th element of the list represents the shear stress at this vertex at the i-th timeslice.

compute_frame()

This function computes the orthonormal frames.

Each centerline point has its own tangent vector along the centerline. The final axial vector $\vec{a}$ at each vertex is the normalized average of the tangents of the centerline tangents that the vertex is in the collar for. The

tangent_aggregate field is an intermediate variable that helps compute the final radial vector.

Each vertex acts as a corner for some triangles. The final radial vector $\vec{r}$ for each vertex is a normalized area-weighted normal with respect to the triangles at the vertex. The radial_aggregate field is an intermediate variable that helps compute the final radial vector.

The final circumferential vector $\vec{c}$ for each vertex is the normalized cross product of the final axial vector and final radial vector: $\vec{c} = \vec{r} \times \vec{a}$.

This function returns a **points_map** with the coords, radial, tangent, circ, radial_aggregate, and tangent_aggregate computed and saved.

norm()

Given a 3-dimensional Vector, this function computes the norm of the vector.

compute_and_write_wall_shear_stress()

This function computes the wall shear stress $\vec{w}$ at each vertex in the mesh using the formula $\vec{w} = (\nabla \mathbf{u} + (\nabla \mathbf{u})^\intercal)/2 \cdot \vec{r}$. Calculations use vectors from the orthonormal frame as well as the tensor field from interpolating labels on **velocity_derivative_timeslices**.
This function writes the .vtk database described in the **Outputs** section.

compute_and_save_average_wall_shear_stress_along_centerline()

Dan wrote this function. This function computes the average components of shear stress at each centerline point using the corresponding collar for the centerline point. This function outputs three .vtk databases, one for each component. Within each plot, the x axis is distance along centerline, and the y axis is the respective average component of shear stress calculated at that centerline point.

compute_and_save_average_wall_shear_stress_along_centerline2()

This function also computes the average components of shear stress at each centerline point, but the values of shear stress are calculated using the outputs from **compute_and_write_wall_shear_stress()**. The output is the same format as the output of **compute_and_save_average_wall_shear_stress_along_centerline()**.

compute_and_save_residuals_along_centerline()

To find the difference in shear stress results between the two average shear stress along centerline functions, this function computes the residual of the two corresponding sets of outputs. The output of this function is the same format as the previous two functions, but the y-axis is the residual shear stress instead of the component of shear stress.

compute_and_save_wall_shear_stress()

This function is the main function of the *wall_shear_stress.cpp* file. This function calls **compute_frame()** and post-processes the output **points_map** to create four lists: **all_points**, **axial_vectors**, **radial_vectors**, and **circ_vectors**. These four lists contain, respectively, the coordinates, local unit axial vector, local unit radial vector, and unit circumferential vector for each vertex on the mesh, and are all in the same order with respect to the corresponding mesh vertex and **Point** object.

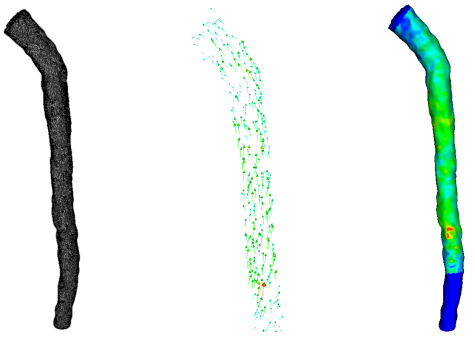
The function then calls **compute_and_write_wall_shear_stress()**. The returned **points_map** from this function call contains the lists of shear stress vectors as well.

3 Example Input and Output

Inputs

centerline:

The centerline of the Given inputs from Dan's code and data from the descending aorta, we can run the code on patient 1 and visualize the mesh of the aorta in VisIt:



Mesh Vectors Magnitudes

The mesh and shear stress outputs at timeslice 7.

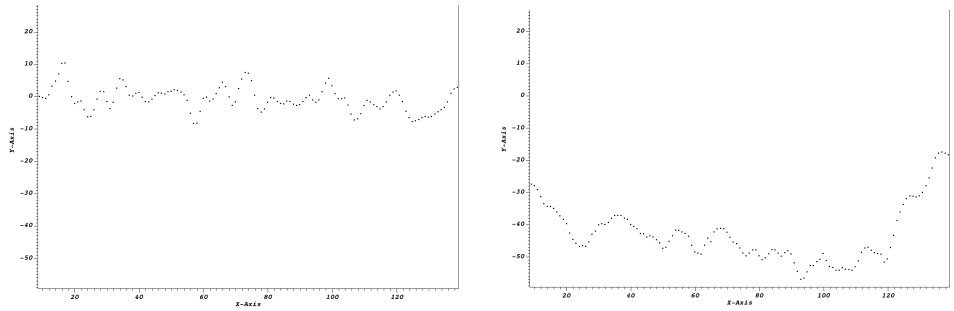
Axial Radial Circumferential

The orthonormal frame at each vertex on the mesh.

From various online sources, mean absolute time-averaged wall shear stress for healthy patients ranged between 0.25 ± 0.04 N/m^2 to 0.33 ± 0.07 N/m^2, or equivalently, 2.5 ± 0.4 dyn/cm^2 to 3.3 ± 0.7 dyn/cm^2. [1] Another source reported

[1] Frydrychowicz, A., Stalder, A.F., Russe, M.F., Bock, J., Bauer, S., Harloff, A., Berger, A., Langer, M., Hennig, J. and Markl, M. (2009), Three-dimensional analysis of segmental wall shear stress in the aorta by flow-sensitive four-dimensional-MRI. J. Magn. Reson. Imaging, 30: 77-84. https://doi.org/10.1002/jmri.21790

it as -1.7 to 1.4 dyn/cm^2. [2] A third source reported it as 0.4 ± 0.3 N/m^2, or equivalently, 4 ± 3 dyn/cm^2. [3]



Timeslice 0 Timeslice 6

The plots of axial component of shear stress along the centerline for patient 1 at timeslices 0 and 6. Systole occurs at timeslice 6, resulting in high shear stress. Not that for clarify of visualization, the y-axis is scaled up by 100.

From this system, time-averaged wall shear stress numbers generally ranged around 0.15 to 0.30 N/m^2, or equivalently, 1.5 to 3.0 dyn/cm^2. Further statistical analysis will need to be done on the results of time-averaged shear stress. All number are saved to a file called *time_averaged_shear_stresses.txt*. Some numbers in this file are 0 or very close to zero. These numbers are small due to lack of velocity data at certain parts of the aortic mesh (namely the ends of each centerline). These numbers should not be considered in the statistical analysis done.

[2] James E. Moore, Chengpei Xu, Seymour Glagov, Christopher K. Zarins, David N. Ku, Fluid wall shear stress measurements in a model of the human abdominal aorta: oscillatory behavior and relationship to atherosclerosis, Atherosclerosis, Volume 110, Issue 2, 1994, Pages 225-240, ISSN 0021-9150, https://doi.org/10.1016/0021-9150(94)90207-0.

[3] Alex J. Barker, Michael Markl, Jonas Bürk, Ramona Lorenz, Jelena Bock, Simon Bauer, Jeanette Schulz-Menger and Florian von Knobelsdorff-Brenkenhoff. (2012). Bicuspid Aortic Valve Is Associated With Altered Wall Shear Stress in the Ascending Aorta. Circulation: Cardiovascular Imaging, 5(4), 457–466. https://doi.org/10.1161/CIRCIMAGING.112.973370

4 Synthetic Data

At the top of the **compute_and_save_wall_shear_stress()** function, there are two sections of code that generate synthetic data to be used for debugging.

The first commented-out section of **compute_and_save_wall_shear_stress()** creates a set of synthetic data in the form of a unit cylinder. The cylinder has radius $= 0.5$, height $= 1$, and is centered at $(0.5, 0.5, 0.5)$. The synthetic velocity field is the unit cube centered at $(0.5, 0.5, 0.5)$. Inside the cylinder, velocity is $(0, 0, -8)$. Outside the cylinder, velocity is $(0, 0, 5)$. This section of the code requires that the path to a .obj file of a closed cylinder be hard-coded into the code. This section re-scales the cylinder to fit in the specifications described above.

The second commented-out section of **compute_and_save_wall_shear_stress()** creates a set of synthetic data along the descending aorta. This section of the code requires that the path to a .obj file the descending aorta of patient 1 be hard-coded into the code. No re-scaling is done. Inside the vessel, velocity is $(0, 0, -11)$. Outside the cylinder, velocity is $(0, 0, 1)$. To change the vessel used, change the input file, as well as the affinity of the grid created.