SoftwareX 32 (2025) 102392

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

VaSP: Vascular Fluid–Structure Interaction Pipeline

Kei Yamamoto [a,e,*], David A. Bruneau [b], Johannes Ring [c], Jørgen S. Dokken [d], Kristian Valen-Sendstad [a]

[a] Department of Computational Physiology, Simula Research Laboratory, Oslo, Norway
[b] Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada
[c] Administration and Management, Simula Research Laboratory, Oslo, Norway
[d] Department of Numerical Analysis and Scientific Computing, Simula Research Laboratory, Oslo, Norway
[e] Department of Mathematics, University of Oslo, Oslo, Norway

ARTICLE INFO

Keywords:
Fluid-structure interaction
FEniCS
VMTK
Vascular research

ABSTRACT

A variety of commercial and open-source software packages exist for modeling blood flow dynamics and vascular wall mechanics of cardiovascular systems via fluid–structure interaction simulations (FSI). While these existing tools are feature-rich, this breadth often comes at the cost of increased complexity and large codebases, which, combined with implementation in low-level programming languages, effectively limit transparency and accessibility. Moreover, many of these software packages primarily rely on graphical user interfaces for a more intuitive experience; however, this can hinder reproducibility and automation of research workflows. Here, we present Vascular Fluid–Structure Interaction Pipeline (VaSP), a transparent, flexible, and compact Python package with a command-line interface. In contrast to the vast majority of existing software, VaSP is tailored for transitional flow and high-frequency vascular wall vibrations. VaSP takes a medical image-derived surface model as input, generates a volumetric mesh with fluid and solid domains for FSI simulations, and post-processes the results for hemodynamic and wall mechanical analyses. By leveraging high-level Python packages such as FEniCS and VMTK, VaSP ensures accessibility for users of diverse expertise levels and promotes reproducible, scriptable workflows.

Code metadata

Current code version	v1.1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-25-00427
Permanent link to Reproducible Capsule	https://github.com/KVSlab/VaSP/archive/refs/tags/v1.1.zip
Legal Code License	GPL-3.0 license
Code versioning system used	git
Software code languages, tools, and services used	Python
Compilation requirements, operating environments & dependencies	Linux, MacOS, Windows, MPI, FEniCS, VMTK, turtleFSI, VaMPy
If available Link to developer documentation/manual	https://kvslab.github.io/VaSP/
Support email for questions	keiya@simula.no, kvs@simula.no

1. Motivation and significance

Computational modeling of blood flow and/or the vascular wall in the cardiovascular system is actively used to gain a fundamental understanding of healthy and diseased states [1–3]. In particular, FSI simulations have been utilized to provide more realistic estimates of wall shear stress than rigid wall simulations [4], to understand the long-term remodeling of the vascular wall [5], or to estimate wall stress that results in mechanical failure of aneurysm wall [6]. In practice, performing physiologically plausible simulations consists of multiple steps, including generating a high-quality mesh and post-processing the results. While dedicated open-source software packages have been used for each individual step, a single, streamlined pipeline that integrates these processes is essential for improving research efficiency.

* Corresponding author at: Department of Mathematics, University of Oslo, Oslo, Norway.
E-mail address: keiya@math.uio.no (K. Yamamoto).

https://doi.org/10.1016/j.softx.2025.102392
Received 23 June 2025; Received in revised form 9 September 2025; Accepted 30 September 2025

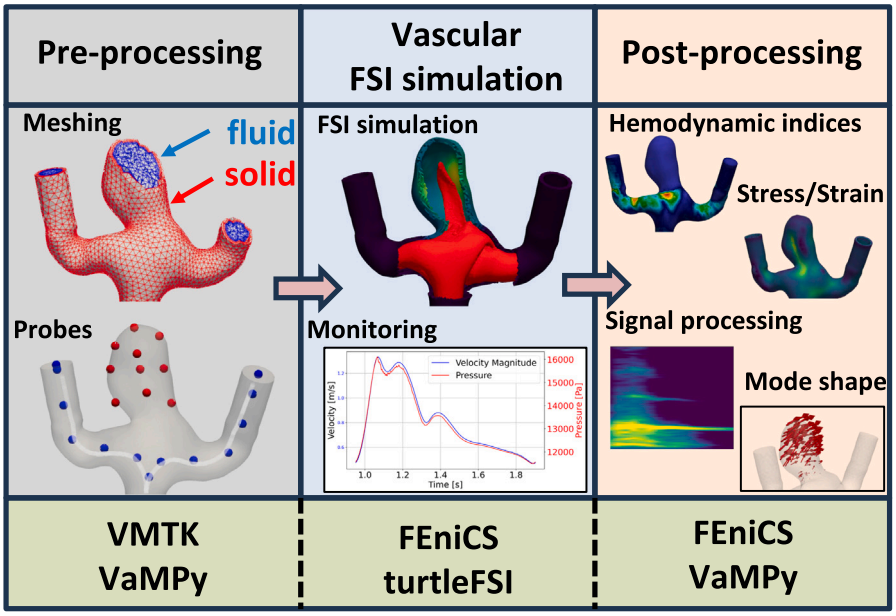

Fig. 1. An illustration of modular architecture of VaSP, consisting of three functionalities: pre-processing, vascular FSI simulation, and post-processing, as indicated in the top box. Within the middle box, selected key features of each functionality are further shown, using a diseased blood vessel (aneurysm) in the brain as an example. The bottom box indicates the primary dependencies for each functionality.

The majority of FSI studies in cardiovascular research use commercial software such as ANSYS [7], Abaqus [8], and COMSOL [9]; however, there are now several excellent open-source software packages for performing FSI simulations in anatomically realistic geometries, such as SimVascular [10], CRIMSON [11] and FEBio [12]. While these software packages are feature-rich, offer a user-friendly workflow from pre-processing to post-processing, and are partially scriptable in Python, the source code is written in low-level languages such as Fortran and C++, consisting of hundreds of thousands of lines. The complexity and volume of its codebase limit the practical transparency and flexibility of the software. For example, implementing new material models or modifying boundary conditions would require extensive knowledge of the code structure and considerable development effort, making it difficult to adapt the software to new research goals beyond the existing capabilities. Moreover, most of the packages rely on a graphical user interface (GUI), which, while user-friendly, is also inherently operator dependent, leading to variability in results and hindering reproducibility [13].

Therefore, the objective of developing the Vascular Fluid–Structure Interaction Pipeline (VaSP) is to provide flexible, transparent, and compact software written in a high-level language (i.e., Python). VaSP offers a command-line interface, which avoids operator-dependency and enhances the reproducibility of the research through scriptable and traceable processes. We leveraged FEniCS [14] for a numerical implementation that closely resembles mathematical notation, making VaSP easily accessible to students and educators with various levels of knowledge in numerical methods and programming. Owing to the high-performance computing capabilities inherent from FEniCS, VaSP can efficiently perform physiologically plausible FSI simulations, with anatomically realistic geometries, over a time window of several cardiac cycles with sufficient spatial and temporal resolution. Lastly, conventional software packages for modeling blood flow and vascular walls have been primarily designed to study laminar flow, time averaged wall shear stress, and strain/stress associated with pulsatile wall deformation. In addition to these capabilities, VaSP has the unique capability to analyze transitional flow and high-frequency wall vibrations that remain largely unexplored, opening avenues for new mechanobiological investigations.

2. Software description

2.1. Software architecture

VaSP is a Python package with three main functionalities: pre-processing, FSI simulations, and post-processing, as illustrated in Fig. 1. At its most fundamental level, VaSP relies on two widely used packages: Vascular Modeling Toolkit (VMTK) [15] and FEniCS.

VMTK provides a collection of tools, covering segmentation of medical images, geometric analysis of surface data, mesh generation, and post-processing of data obtained from CFD simulations. However, for those new to cardiovascular simulations, adjusting the meshing parameters for generating a high-quality mesh can be challenging. To overcome this barrier, we created a streamlined meshing pipeline in VaSP, where meshing parameters, such as the length of flow extensions, are pre-determined based on our rigorous research experience.

FEniCS is an open-source software for solving partial differential equations with the finite element method and is mainly used in the FSI solver and post-processing component of VaSP. FEniCS uses unified form language (UFL) [16] to define variational problems in a way that closely resembles the mathematical representation.

VaSP integrates two other open-source Python packages: turtleFSI [17] and Vascular Modeling Pypeline (VaMPy) [18]. The FEniCS-based solver turtleFSI is verified, validated, and space/time centered, offering second-order accuracy in time and $\mathbb{P}+1$ in space. Users only need to provide the mesh, define parameters (e.g., time step, constitutive model for structures), and specify boundary conditions to perform numerically complex FSI simulations. VaMPy is an automated computational fluid dynamics (CFD) pipeline for modeling cardiovascular flow and serves as the precursor to VaSP.

2.2. Software functionalities

2.2.1. Pre-processing

The pre-processing module of VaSP accepts surface meshes in VMTK compatible formats such as .vtp or .stl. These surface meshes are typically segmented from medical images, a process that can be performed using various open-source software packages such as VMTK, 3D Slicer [19], and Medical Imaging Interaction Toolkit [20]. The simplest approach for running the pre-processing is to call

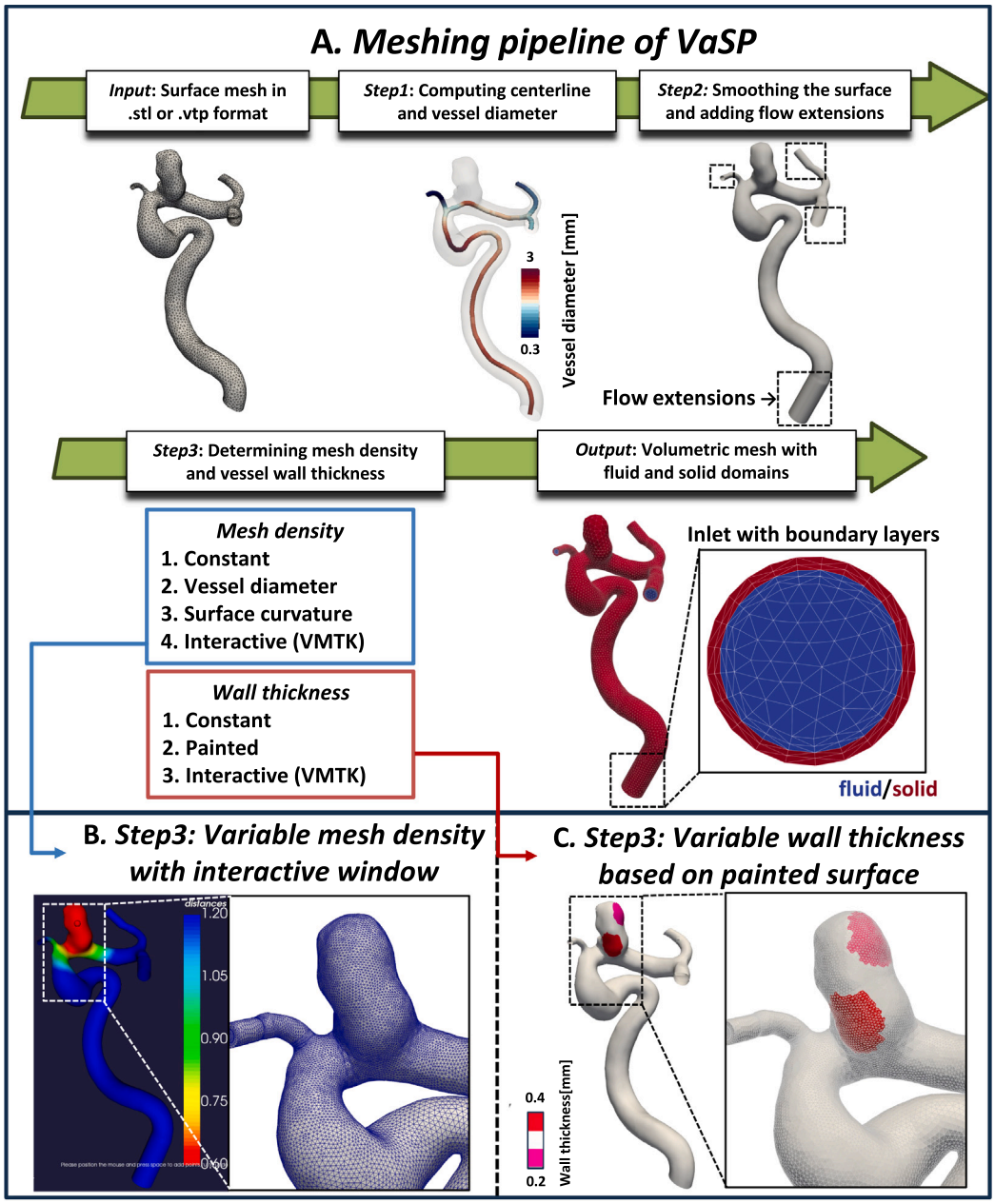

Fig. 2. An overview of the automated meshing pipeline in VaSP, using a diseased blood vessel (aneurysm) in the brain as an example. The pipeline generates a volumetric mesh from a triangulated surface with three steps. **A** illustrates the general process. **B** shows how users can define regions of high mesh density (refinement) using an interactive window. **C** demonstrates a variable wall thickness generation with a pre-defined mesh.

```
$ vasp-generate-mesh --input-model mesh.stl
```

from a terminal. Here `--input-model` specifies the path to the input surface mesh (in this case `mesh.stl`). While the exact procedure of pre-processing can vary depending on the specified meshing choices, such as variable mesh density and wall thickness, Fig. 2**A** summarizes the automated meshing pipeline.

The first step of meshing, which is automated and requires no user interaction, is to compute the centerline that contains a vessel diameter (Fig. 2**A** step1). The latter can be used to generate a variable density mesh in the subsequent process. The surface mesh is then smoothed, as high-frequency features, such as artifacts from medical image segmentation, are unphysiological and can cause problems when generating volumetric meshes. The level of smoothing can be adjusted by changing the number of iterations (`--smoothing-iterations`). After the smoothing, the next step of the pipeline is to add cylindrical flow extensions with flat surfaces at the automatically annotated inlet(s) and outlet(s)(Fig. 2**A** step2). Medical image-derived models are typically cut at arbitrary sections of the vasculature and have non-circular cross sections, where the spatial or temporal distribution of the flow is unknown. Cylindrical inlets, positioned a few diameters upstream of the region of interest, enable us to impose analytically derived velocity profiles, such as parabolic and Womersley flows. Note that users can adjust the smoothing method (Laplace, Taubin, and Voronoi) and the length of flow extensions. For example, the following command will add flow extensions with a length of eight (if not specified, the default value is five) times the local radius at both the inlet and the outlet.

```
$ vasp-generate-mesh --input-model mesh.stl
    --smoothing-method laplace --add-
    flowextensions --inlet-flowextension 8 --
    outlet-flowextension 8
```

See the VaMPy documentation [21] for more examples of different smoothing methods and flow extensions. The last step before generating the volumetric mesh is to determine the local mesh size and solid thickness, controlled by `--meshing-method` and `--solid-thickness` flags, respectively (Fig. 2**A** step3). VaSP provides three automatic methods and one manual method for controlling the local mesh size. The automatic methods determine the local mesh density based on constant edge length, surface curvature, or vessel diameter, ensuring objectivity and reproducibility of the final mesh, and are explained in the VaMPy documentation [22]. It is important here to note that users can also choose a specific region for mesh refinement interactively by placing seed points, as shown in Fig. 2**B**. Similarly, VaSP has three methods for specifying the local wall thickness: constant, painted, or interactive. The first option only requires a single value as an input, whereas the second option, painted, necessitates a surface mesh with locally predefined wall thickness, as shown in Fig. 2**C**. The last option is similar to the interactive refinement of mesh density (Fig. 2**B**) and will be explained in detail with illustrative examples (see Section 3.3 and Fig. 8).

Finally, an unstructured volumetric mesh, consisting of tetrahedral cells, is generated. The output mesh from pre-processing is saved in hierarchical data format 5 (HDF5) [23], containing three objects: mesh, domains, and boundaries, following the syntax of turtleFSI. The mesh object contains geometrical and topological information, such as cell coordinates and connectivity. Meanwhile, domains and boundaries hold unique markers to distinguish the fluid and solid domains, as well as their boundaries (e.g., inlet, outlet, and interface). These markers are used later in the FSI simulations to define governing equations for fluid and solid domains and to specify boundary conditions. Listing 1 shows the list of mesh files generated, where `mesh.stl` is an example input surface mesh and `mesh.h5` is the resulting volumetric mesh that will be used as an input to FSI simulations. The purpose of the other files is explained in the VaSP documentation [24].

```

mesh.stl
mesh.h5
mesh.vtu
mesh.xml.gz
mesh_domains.pvd/.vtu
mesh_boundaries.pvd/.vtu
mesh_edge_length.xdmf/.h5
mesh_info.json
mesh_probe_point.json
```

Listing 1: An example of files generated as a result of pre-processing, where mesh.stl is an input surface mesh and mesh.h5 is an output volumetric mesh.

While the process of meshing is as described above, there is one remaining procedure before running a FSI simulation, which is to pre-deform the mesh. This process is essential because the medical images capture the vascular wall in an in vivo stress equilibrium state, where both the blood pressure and wall stress are unknown. A detailed description is given in the online documentation [25].

2.2.2. FSI simulations

VaSP uses turtleFSI, a FEniCS-based open-source package, for performing FSI simulations. While turtleFSI is a generic FSI solver, it has been primarily used in the context of transitional blood flow and soft tissue mechanics [26]. This monolithic solver couples equations for fluid, solid, and mesh motion to form a single non-linear system. Its modular structure enables a wide range of simulations, including rigid-wall or moving-domain CFD, structural mechanics only, and FSI, all possible within the same framework. This flexibility is important, for example, to directly compare CFD and FSI simulations under equivalent conditions or to perform a numerical uniaxial tension test on the structural domain alone to fit material parameters against experimental data, as previously presented [26,27]. To further demonstrate the usage of turtleFSI as a CFD solver, we added new problem files for rigid-wall and moving-domain simulations: the 2D Taylor–Green vortex problem (Appendix A) and the 2D oscillating cylinder problem (Appendix B), respectively.

In practice, users configure turtleFSI through a problem file, written in Python, which specifies a set of parameters, loads the mesh, and defines boundary and initial conditions. While a detailed explanation of how to create a problem file is covered in the turtleFSI documentation [28], we provide five problem files in VaSP under the `src/vasp/simulations` folder that can be used as a template for new problems. A detailed explanation of how to apply physiologically plausible boundary conditions is also provided in the VaSP documentation [29]. Some of the problems will be further explained in detail as illustrative examples in Section 3.

turtleFSI leverages the entry-level high-performance computing capabilities of FEniCS via the Message Passing Interface [30]. To further support cluster-based simulations, we developed a diagnostic tool, accessible via the `vasp-log-plotter` command, allowing users to visually inspect critical information, such as the Courant number and the convergence of Newton iterations, directly from simulation log files, as shown in Listing 2 and Fig. 3. For example, the following command will plot the Courant number from the second to last cardiac cycle:

```
$ vasp-log-plotter log_file.txt --plot-cfl --
    start-cycle 2
```

This diagnostic information can be used to monitor the progress of simulations and also to modify the problem setup in case of divergence.

```
Newton iteration 0: r (atol) = 6.336e-02, r (
    rel) = 1.018e-03
Newton iteration 1: r (atol) = 1.623e-05, r (
    rel) = 4.127e-04
Newton iteration 2: r (atol) = 2.529e-07, r (
    rel) = 2.581e-06

Flow Properties:
  Flow Rate at Inlet: 1.48e-06
  Velocity (mean, min, max): 0.073, 1.69-06,
    0.479
  CFL (mean, min, max): 0.0323, 7.47e-07,
    0.212
  Reynolds Numbers (mean, min, max): 100,
    0.00232, 659

Solved for timestep 1848, t = 0.3515 in 4.9 s
```

Listing 2: An example of output during the FSI simulations written to a log file.

2.2.3. Post-processing

The post-processing part of VaSP is divided into three subcategories with the tree structure shown in Listing 3.

```
src/
 vasp/
     postprocessing/
         postprocessing_mesh/...
         postprocessing_fenics/...
         postprocessing_h5py/...
```

Listing 3: Directory tree strucutre of post-processing part of VaSP.

The first category addresses the mesh. Since turtleFSI is a monolithic solver, all variables (displacement, velocity, and pressure) are defined and saved across the entire domain during simulations. However, it is preferable to separate variables by domain so that fluid- and solid-related post-processing can be performed independently and

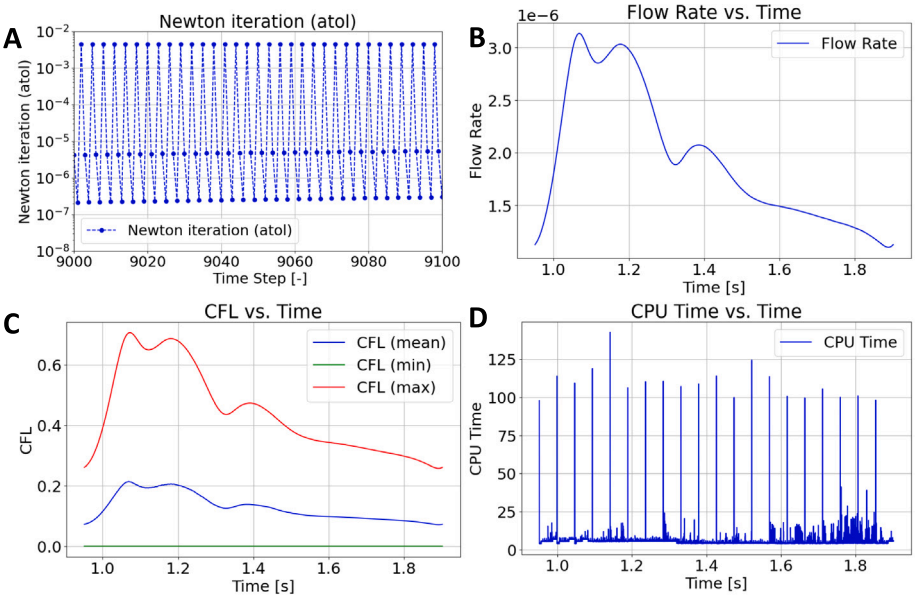

Fig. 3. Diagnostic plots generated based on a log file using `vasp-log-plotter`. **A** the convergence of Newton iterations where the tolerance was set as 10^{-6}. **B** flow rate at the inlet over the cardiac cycle. **C** mean, minimum, and maximum Courant number over the entire domain. **D** CPU time spent per time-step.

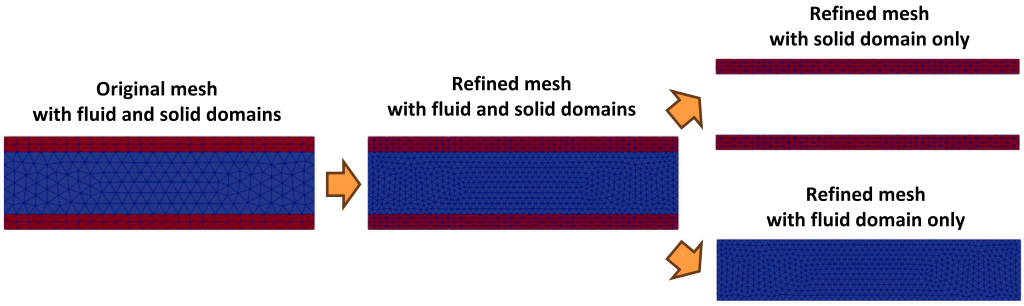

Fig. 4. An example of post-processing the mesh by refining the original mesh first and then separating fluid (blue) and solid (red) domains.

simultaneously, thereby enhancing efficiency. As a prior step for separating variables, users need to obtain fluid- and solid-only meshes using the `vasp-separate-mesh` command. Moreover, turtleFSI includes a feature that maps solutions from higher- to lower-order elements on a refined mesh. This ensures that all the degrees of freedom are saved for post-processing. While not strictly required, to take full advantage of higher-order solutions, it is recommended to generate a refined mesh for the rest of post-processing using the `vasp-refine-mesh` command. Fig. 4 illustrates the procedure of refining and separating the mesh and Listing 4 shows the list of mesh files generated.

```
Mesh/
 mesh.h5
 mesh_fluid.h5
 mesh_solid.h5
 mesh_refined.h5
 mesh_refined_fluid.h5
 mesh_refined_solid.h5
```

Listing 4: Mesh files generated as a result of refining and separating original mesh.

The second component of post-processing is to compute hemodynamic indices (e.g., wall shear stress and oscillatory shear index) and wall strain/stress (e.g., Green–Lagrange strain and Cauchy stress) using FEniCS. By default, fluid velocity and solid displacement are automatically separated during post-processing. However, if needed, this separation can be performed using the `vasp-create-hdf5` command. This process involves reading the original simulation results obtained with the entire mesh and then extracting fluid and solid components using segregated meshes. Then, VaSP uses fluid velocity and solid displacement combined with a fluid- and solid-only mesh to compute hemodynamic indices (`vasp-compute-hemo`) and wall strain/stress (`vasp-compute-stress`), respectively, as illustrated in Fig. 5. Mathematical definitions of computed hemodynamic and solid-mechanical indices are available in the VaMPy documentation [31] and Appendix C, respectively.

The third component of post-processing is to perform signal processing tasks, including computing power spectral density (`vasp-create-spectrum`), generating spectrograms (`vasp-create -spectrograms-chromagrams`), and applying low/high-pass filtering to results (`vasp-create-hi-pass-viz`). This post-processing component originated from recent CFD and FSI studies reporting flow instabilities and high-frequency wall vibrations in various vascular systems [26,32]. VaSP uses a periodogram for estimating power spectral density, short-time Fourier Transform for generating spectrograms [33], and Butterworth filters for low-pass, high-pass, bandpass, and bandstop, all implemented using `scipy.signal` [34]. Additionally, VaSP can compute the spectral bandness index and generate chromagrams [35].

3. Illustrative examples

This section presents three examples to demonstrate the key capabilities of VaSP using vascular problems, provided as Python scripts

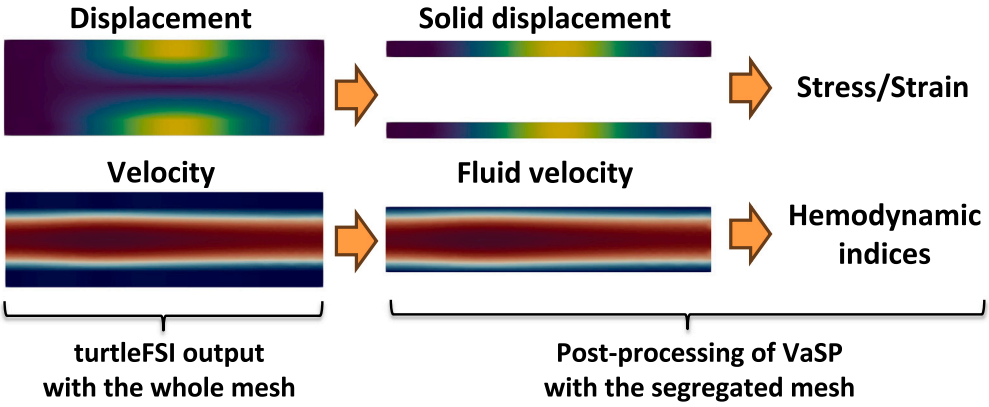

Fig. 5. An example of separating solid displacement and fluid velocity and post-processing independently for domain specific metrics.

under the `simulations` folder and as a tutorial in the online VaSP documentation along with corresponding commands [36]. The first example, offset stenosis, is a computationally efficient problem, allowing novice users to experiment with VaSP on a standard laptop. Here, we focus on one of the unique features of VaSP, i.e. the signal processing component of the post-processing as the fluid and solid exhibit flow instabilities and high-frequency wall vibrations, respectively. The second example, a patient-specific cerebral aneurysm model, previously used in several publications [26,37,38], is presented to demonstrate hemodynamic and solid mechanics analyses that are typically of physiological interest. The third example, arteriovenous fistula (AVF), highlights advanced meshing features like multi-solid domains and variable wall thickness, showcasing VaSP's capability to represent heterogeneous physiological systems.

3.1. Offset stenosis

This offset stenosis model is a synthetic geometry that can be analytically described and has long been studied in the fluid mechanics community [39,40]. The area reduction with an offset in the stenotic region causes three dimensional flows that transition to turbulence at a low Reynolds number.

As shown in Fig. 6**A**, the flow inside the stenosis is characterized by the unstable post-stenotic jet. The spectrogram (Fig. 6**B**) of fluid velocity reflects unstable flow as a continuous and broad frequency band. In contrast, pressure and displacement spectrograms exhibit narrow-frequency bands, indicating the wall vibration. As shown in Fig. 6**C**, the vibration motion can be band-pass filtered to extract directional motions associated with modal frequencies. In this case, mode 1 (70–90 Hz) and 2 (150–170 Hz) correspond to contraction/expansion and side-to-side motions, consistent with previous studies [37,38].

3.2. Cerebral aneurysm

A cerebral aneurysm is a pathological dilation of an artery in the brain. It has been extensively studied using CFD [32,41,42] and FSI [4,6,43] simulations. Given the research interest in cerebral aneurysms from both computational and physiological perspectives, it serves as an excellent example to showcase the functionalities of VaSP. Fig. 7 displays fluid velocity, solid displacement, hemodynamic and solid-mechanical indices, and high-pass filtered displacement/strain.

3.3. Arteriovenous fistula

An AVF, shown in Fig. 8**A**, is a surgically-created vascular connection between an artery and a vein. To independently assign physiologically plausible wall stiffness and thickness to each vessel, it is necessary to construct a mesh with unique domain markers. VaSP leverages the "branch splitting" functionality in VMTK, which objectively splits branches with individual markers, as shown in Fig. 8**B**. VaSP uses the split centerlines to assign different solid domain markers. Additionally, VaSP can generate a mesh with different wall thicknesses for each branch. As shown in Fig. 8**C**, variable wall thickness can be interactively defined with spheres. Specifically, users need to provide minimum and maximum thickness, and VaSP assigns local wall thickness to all surface points based on the Euclidean distance from the spheres. Fig. 8**D** is a volumetric mesh, where the blue and red domains represent the artery of 0.3 mm and 0.15 mm wall thickness, respectively, as highlighted in Fig. 8**E**.

4. Impact

Setting aside the strength of VaSP as a flexible, transparent, and compact software, it is, to the author's knowledge, the first cardiovascular software designed to analyze flow instabilities and associated high-frequency wall vibrations. The first work that formed the basis of VaSP was by Souche et al. [26], who reported cerebral aneurysm wall vibration using patient-specific geometry. In contrast to previous FSI studies of cerebral aneurysms that simulated blood flow as laminar and focused on low-frequency pulsatile deformation, and time-averaged quantities [4,6], Souche et al. employed high-fidelity FSI simulations, revealing transitional flows and non-linear, two-way coupling between fluid pressure and solid displacement. Later, Bruneau et al. [37] developed a methodology for computing vibration amplitude, extracting mode shapes, and provided a mechanistic explanation for aneurysm wall vibrations. Furthermore, Bruneau et al. [38] incorporated pulsatile inflow, improving the physiological realism of the FSI simulations, and offered a plausible explanation for clinically reported aneurysm sounds [44,45]. Wall vibrations could also be relevant in other vascular diseases, such as AVF. Bozzetto et al. [46] were the first to demonstrate flow-induced wall vibrations in AVF, whereas a recent study by Soliveri et al. [27] introduced a more advanced model by assigning different wall stiffnesses to the artery and vein and incorporating perivascular tissues. Similarly, in contrast to previous work on AVF [47,48], Bozzetto et al. and Soliveri et al. were the first to consider transitional flow and flow-induced vibrations. These studies suggested that the latter could indeed cause degenerative changes in the vascular wall. To

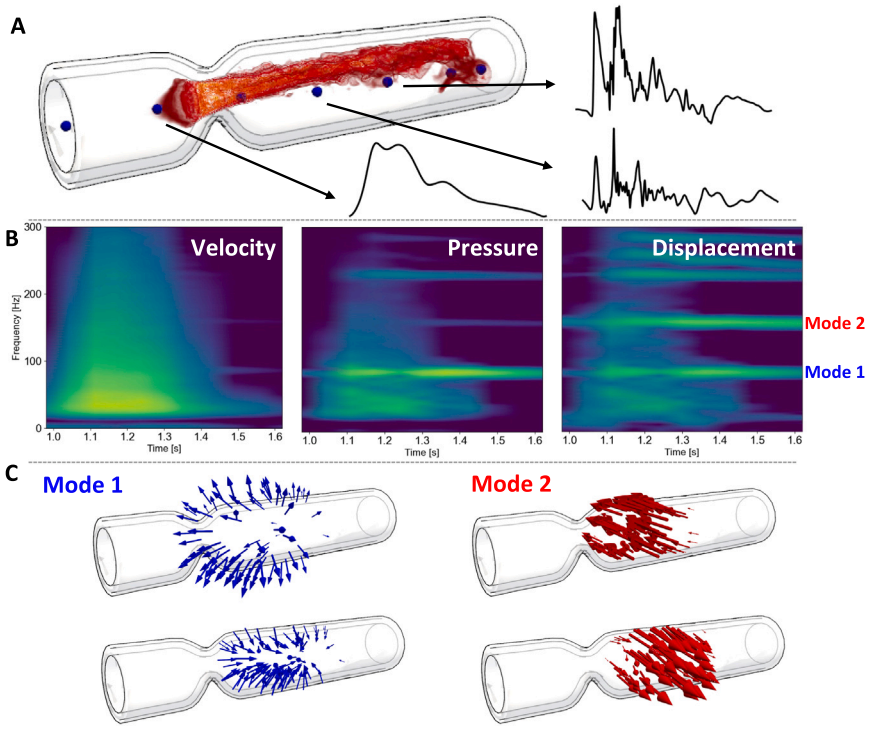

Fig. 6. Depictions of flow instability and high-frequency wall vibrations in an offset stenosis. **A** displays a volumetric rendering of instantaneous velocity magnitude, with probe points and corresponding velocity plots. **B** shows spectrograms of fluid velocity, pressure, and wall displacement, illustrating the evolution of frequency over time. **C** visualizes the first two mode shapes (70–90 Hz and 150–170 Hz), with displacement vectors amplified to show the specific patterns of vibration.

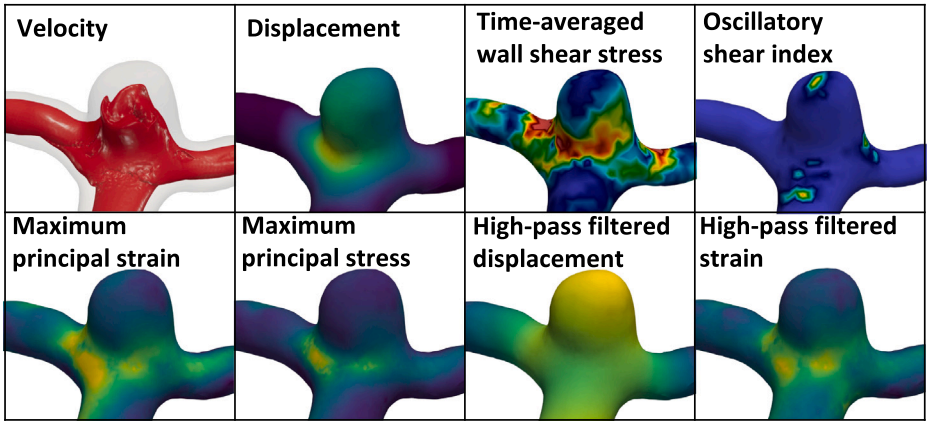

Fig. 7. Examples of post-processed hemodynamic and solid-mechanical indices, as well as high-pass filtered displacement and strain.

provide a unified platform for investigating potential flow-induced wall vibrations in the cardiovascular system, all the components developed and presented in those studies have been incorporated into VaSP.

5. Conclusions

VaSP is an open-source, flexible, and intuitive Python package that streamlines all the complex processes of performing vascular FSI simulations. The meshing pipeline is automated, reducing the variability across users and ensuring the reproducibility of simulation results. A verified and validated monolithic FSI solver with high-performance computing capabilities is integrated to perform vascular FSI simulations, where several problem files are provided as a template. Moreover, a diagnostic tool is added to support simulations. A wide range of post-processing functionalities is also provided for hemodynamic analysis and characterizing flow instabilities and high-frequency vascular wall vibrations. VaSP originated from research activities with several journal publications, and new features are continuously added to the pipeline as a result of active use. In conclusion, VaSP is a valuable resource that can produce publication-standard results by students and

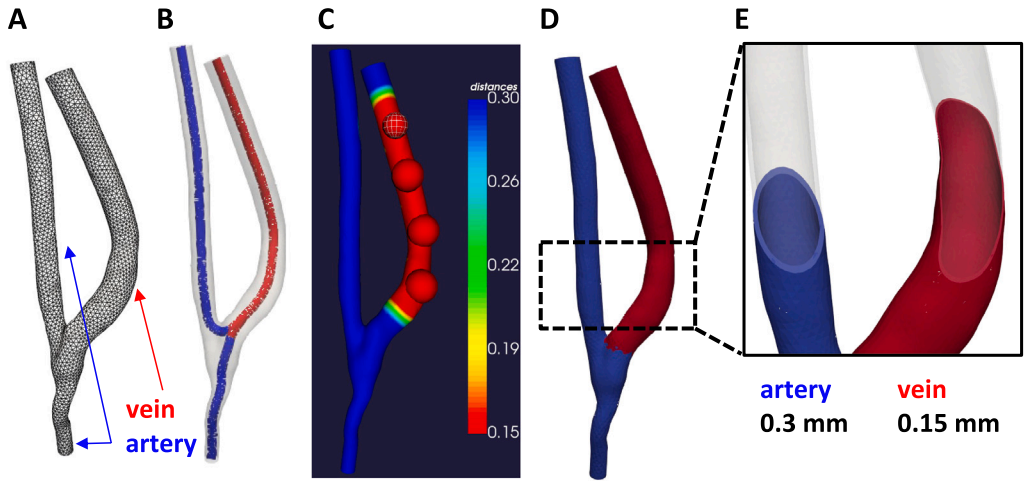

Fig. 8. Advanced meshing functionalities exemplified with an arteriovenous fistula model. **A** is an input surface mesh, **B** is a centerline with different marker for artery and vein, **C** demonstrates an interactive way of specifying variable wall thickness using spheres in red, **D** shows a volumetric mesh with unique domain marker for artery (blue) and vein (red) based on branch splitting, and **E** shows clipped a solid mesh, showing arterial and venous wall thickness of 0.3 mm and 0.15 mm, respectively.

researchers with diverse backgrounds, and has a novel capability to investigate high-frequency flow-induced vascular wall vibrations.

CRediT authorship contribution statement

Kei Yamamoto: Writing – original draft, Visualization, Software, Methodology. **David A. Bruneau:** Writing – review & editing, Software, Methodology, Conceptualization. **Johannes Ring:** Writing – review & editing, Software, Methodology. **Jørgen S. Dokken:** Writing – review & editing, Software, Methodology. **Kristian Valen-Sendstad:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the author(s) used ChatGPT to check grammar, clarity, and coherence. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This project is supported by the EU Horizon 2020 SimCardioTest project (101016496). D.A.B. acknowledges the support of an NSERC Canada Graduate Scholarship. The simulations were performed on the Saga cluster, with resources provided by UNINETT Sigma2 – the National Infrastructure for High Performance Computing and Data Storage in Norway, grant number nn9249k. The authors would like to thank Dr. Daniel E. MacDonald for the contribution to the software implementation, Dr. Luca Soliveri, Sofia Poloni, and Dr. Michela Bozzetto for fruitful discussions, and Dr. Henrik Nicolay Finsberg and Dr. Mohammad Javad Sadeghinia for providing insightful comments on earlier versions of the manuscript.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.softx.2025.102392.

References

[1] Steinman DA, Taylor CA. Flow imaging and computing: Large artery hemodynamics. Ann Biomed Eng 2005;33:1704–9. http://dx.doi.org/10.1007/s10439-005-8772-2.

[2] Holzapfel GA, Ogden RW. Constitutive modelling of arteries. Proc R Soc A: Math Phys Eng Sci 2010;466(2118):1551–97. http://dx.doi.org/10.1098/rspa.2010.0058.

[3] Taylor CA, Petersen K, Xiao N, Sinclair M, Bai Y, Lynch SR, UpdePac A, Schaap M. Patient-specific modeling of blood flow in the coronary arteries. Comput Methods Appl Mech Engrg 2023;417:116414. http://dx.doi.org/10.1016/j.cma.2023.116414.

[4] Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE. Computer modeling of cardiovascular fluid-structure interactions with the deforming-spatial-domain/stabilized space-time formulation. Comput Methods Appl Mech Engrg 2006;195(13–16):1885–95. http://dx.doi.org/10.1016/j.cma.2005.05.050.

[5] Baumler K, Rolf-Pissarczyk M, Schussnig R, Fries TP, Mistelbauer G, Pfaller MR, Marsden AL, Fleischmann D, Holzapfel GA. Assessment of aortic dissection remodeling with patient-specific fluid-structure interaction models. IEEE Trans Biomed Eng 2024;72(3):953–64. http://dx.doi.org/10.1109/TBME.2024.3480362.

[6] Bazilevs Y, Hsu MC, Zhang Y, Wang W, Kvamsdal T, Hentschel S, Isaksen JG. Computational vascular fluid-structure interaction: Methodology and application to cerebral aneurysms. Biomech Model Mechanobiol 2010;9(4):481–98. http://dx.doi.org/10.1007/s10237-010-0189-7.

[7] Madenci E, Guven I. The finite element method and applications in engineering using ANSYS®. Springer; 2015.

[8] Smith M. ABAQUS/standard user's manual, version 6.9. United States: Dassault Systèmes Simulia Corp; 2009.

[9] Multiphysics C. Introduction to COMSOL multiphysics®. 1998,

[10] Updegrove A, Wilson NM, Merkow J, Lan H, Marsden AL, Shadden SC. SimVascular: An Open Source Pipeline for Cardiovascular Simulation. Ann Biomed Eng 2017;45(3):525–41. http://dx.doi.org/10.1007/s10439-016-1762-8.

[11] Arthurs CJ, Khlebnikov R, Melville A, Marčan M, Gomez A, Dillon-Murphy D, Cuomo F, Vieira MS, Schollenberger J, Lynch SR, Tossas-Betancourt C, Iyer K, Hopper S, Livingston E, Youssefi P, Noorani A, Ahmed SB, Nauta FJ, van Bakel TM, Ahmed Y, van Bakel PA, Mynard J, Di Achille P, Gharahi H, Lau KD, Filonova V, Aguirre M, Nama N, Xiao N, Baek S, Garikipati K, Sahni O, Nordsletten D, Alberto Figueroa C. CRIMSON: An open-source software framework for cardiovascular integrated modelling and simulation. PLoS Comput Biol 2021;17(5):1–21. http://dx.doi.org/10.1371/journal.pcbi.1008881.

[12] Maas SA, Ellis BJ, Ateshian GA, Weiss JA. FEBio: Finite elements for biomechanics. J Biomech Eng 2012;134(1):1–10. http://dx.doi.org/10.1115/1.4005694.

[13] Valen-Sendstad K, Bergersen AW, Shimogonya Y, Goubergrits L, Bruening J, Pallares J, Cito S, Piskin S, Pekkan K, Geers AJ, Larrabide I, Rapaka S, Mihalef V, Fu W, Qiao A, Jain K, Roller S, Mardal K-A, Kamakoti R, Spirka T, Ashton N, Revell A, Aristokleous N, Houston JG, Tsuji M, Ishida F, Menon PG, Browne LD, Broderick S, Shojima M, Koizumi S, Barbour M, Aliseda A, Morales HG, Lefèvre T, Hodis S, Al-Smadi YM, Tran JS, Marsden AL, Vaippummadhom S, Einstein GA, Brown AG, Debus K, Niizuma K, Rashad S, Sugiyama S-i, Owais Khan M, Updegrove AR, Shadden SC, Cornelissen BMW, Majoie CBLM, Berg P, Saalfield S,

Kono K, Steinman DA. Real-World Variability in the Prediction of Intracranial Aneurysm Wall Shear Stress: The 2015 International Aneurysm CFD Challenge. Cardiovasc Eng Technol 2018;9(4):544–64. http://dx.doi.org/10.1007/s13239-018-00374-2.

[14] Logg A, Mardal K-A, Wells G. Automated solution of differential equations by the finite element method, vol. 84, Berlin, Heidelberg: Springer Berlin Heidelberg; 2012, http://dx.doi.org/10.1007/978-3-642-23099-8.

[15] Piccinelli M, Veneziani A, Steinman DA, Remuzzi A, Antiga L. A framework for geometric analysis of vascular structures: Application to cerebral aneurysms. IEEE Trans Med Imaging 2009;28(8):1141–55. http://dx.doi.org/10.1109/TMI.2009.2021652.

[16] Alnæs MS, Logg A, Ølgaard KB, Rognes ME, Wells GN. Unified form language: A domain-specific language for weak formulations of partial differential equations. ACM Trans Math Software 2014;40(2). http://dx.doi.org/10.1145/2566630.

[17] Bergersen A, Slyngstad A, Gjertsen S, Souche A, Valen-Sendstad K. turtleFSI: A Robust and Monolithic FEniCS-based Fluid-Structure Interaction Solver. J Open Source Softw 2020;5(50):2089. http://dx.doi.org/10.21105/joss.02089.

[18] Kjeldsberg HA, Bergersen AW, Valen-Sendstad K. VaMPy: An Automated and Objective Pipeline for Modeling Vascular Geometries. J Open Source Softw 2023;8(85):5278. http://dx.doi.org/10.21105/joss.05278.

[19] Kapur T, Egger J, Jayender J, Toews M, Wells WM. Registration and Segmentation for Image-Guided Therapy. In: Intraoperative imaging and image-guided therapy. Springer; 2014, p. 79–91. http://dx.doi.org/10.1007/978-1-4614-7657-3_5.

[20] Nolden M, Zelzer S, Seitel A, Wald D, Müller M, Franz AM, Maleike D, Fangerau M, Baumhauer M, Maier-Hein L, Maier-Hein KH, Meinzer HP, Wolf I. The medical imaging interaction toolkit: Challenges and advances: 10 years of open-source development. Int J Comput Assist Radiol Surg 2013;8(4):607–20. http://dx.doi.org/10.1007/s11548-013-0840-8.

[21] Kjeldsberg H. Pre-processing. 2024, https://kvslab.github.io/VaMPy/preprocess.html#smoothing.

[22] Kjeldsberg H. High-resolution CFD simulation in the internal carotid artery. 2024, https://kvslab.github.io/VaMPy/artery.html.

[23] Koranne S. Hierarchical Data Format 5 : HDF5. In: Handbook of open source tools. Boston, MA: Springer US; 2011, p. 191–200. http://dx.doi.org/10.1007/978-1-4419-7719-9_10.

[24] Yamamoto K. VaSP preprocess. 2025, https://kvslab.github.io/VaSP/preprocess.html.

[25] Yamamoto K. VaSP prestress. 2025, https://kvslab.github.io/VaSP/prestress.html.

[26] Souche A, Valen-Sendstad K. High-fidelity fluid structure interaction simulations of turbulent-like aneurysm flows reveals high-frequency narrowband wall vibrations: A stimulus of mechanobiological relevance? J Biomech 2022;145:111369. http://dx.doi.org/10.1016/j.jbiomech.2022.111369.

[27] Soliveri L, Bruneau D, Ring J, Bozzetto M, Remuzzi A, Valen-Sendstad K. Toward a physiological model of vascular wall vibrations in the arteriovenous fistula. Biomech Model Mechanobiol 2024;23(0123456789). http://dx.doi.org/10.1007/s10237-024-01865-z.

[28] Bergersen AW, Gjertsen S, Souche A, Slyngstad A. Create your own problem file. 2024, https://turtlefsi2.readthedocs.io/en/latest/using_turtleFSI.html#create-your-own-problem-file.

[29] Yamamoto K. VaSP boundary conditions. 2025, https://kvslab.github.io/VaSP/aneurysm.html#fluid-boundary-conditions.

[30] Forum MPI. MPI: A message-passing interface standard version 4.0. 2021.

[31] Kjeldsberg H. Computed quantities. 2024, https://kvslab.github.io/VaMPy/quantities.html.

[32] Valen-Sendstad K, Mardal KA, Mortensen M, Reif BAP, Langtangen HP. Direct numerical simulation of transitional flow in a patient-specific intracranial aneurysm. J Biomech 2011;44(16):2826–32. http://dx.doi.org/10.1016/j.jbiomech.2011.08.015.

[33] Natarajan T, MacDonald DE, Najafi M, Khan MO, Steinman DA. On the spectrographic representation of cardiovascular flow instabilities. J Biomech 2020;110:109977. http://dx.doi.org/10.1016/j.jbiomech.2020.109977.

[34] Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E, Carey CJ, Polat İ, Feng Y, Moore EW, VanderPlas J, Laxalde D, Perktold J, Cimrman R, Henriksen I, Quintero EA, Harris CR, Archibald AM, Ribeiro AH, Pedregosa F, van Mulbregt P, SciPy 10 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods 2020;17:261–72. http://dx.doi.org/10.1038/s41592-019-0686-2.

[35] MacDonald DE, Najafi M, Temor L, Steinman DA. Spectral Bandedness in High-Fidelity Computational Fluid Dynamics Predicts Rupture Status in Intracranial Aneurysms. J Biomech Eng 2022;144(6). http://dx.doi.org/10.1115/1.4053403.

[36] Yamamoto K. VaSP tutorial. 2025, https://kvslab.github.io/VaSP/tutorials.html.

[37] Bruneau DA, Valen-Sendstad K, Steinman DA. Onset and nature of flow-induced vibrations in cerebral aneurysms via fluid–structure interaction simulations. Biomech Model Mechanobiol 2023;22(0123456789). http://dx.doi.org/10.1007/s10237-022-01679-x.

[38] Bruneau DA, Steinman DA, Valen-Sendstad K. Understanding intracranial aneurysm sounds via high-fidelity fluid-structure-interaction modelling. Commun Med 2023;3(1):1–11. http://dx.doi.org/10.1038/s43856-023-00396-5.

[39] Varghese SS, Frankel SH, Fischer PF. Direct numerical simulation of stenotic flows. Part 1. Steady flow. J Fluid Mech 2007;582:253–80. http://dx.doi.org/10.1017/S0022112007005848.

[40] Khan MO, Valen-Sendstad K, Steinman DA. Direct Numerical Simulation of Laminar-Turbulent Transition in a Non-Axisymmetric Stenosis Model for Newtonian vs. Shear-Thinning Non-Newtonian Rheologies. Flow, Turbul Combust 2019;102(1):43–72. http://dx.doi.org/10.1007/s10494-018-9905-7.

[41] Steinman DA, Milner JS, Norley CJ, Lownie SP, Holdsworth DW. Image-based computational simulation of flow dynamics in a giant intracranial aneurysm. Am J Neuroradiol 2003;24(4):559–66.

[42] Cebral JR, Castro MA, Burgess JE, Pergolizzi RS, Sheridan MJ, Putman CM. Characterization of cerebral aneurysms for assessing risk of rupture by using patient-specific computational hemodynamics models. Am J Neuroradiol 2005;26(10):2550–9. http://dx.doi.org/10.1016/s0098-1672(08)70473-9.

[43] Sanchez M, Ecker O, Ambard D, Jourdan F, Nicoud F, Mendez S, Lejeune JP, Thines L, Dufour H, Brunel H, Machi P, Lobotesis K, Bonafe A, Costalat V. Intracranial aneurysmal pulsatility as a new individual criterion for rupture risk evaluation: Biomechanical and numeric approach (IRRAs Project). Am J Neuroradiol 2014;35(9):1765–71. http://dx.doi.org/10.3174/ajnr.A3949.

[44] Ferguson GG. Turbulence in human intracranial saccular aneurysms. J Neurosurg 1970;33(5):485–97. http://dx.doi.org/10.3171/jns.1970.33.5.0485.

[45] Kurokawa Y, Abiko S, Watanabe K. Noninvasive detection of intracranial vascular lesions by recording blood flow sounds. Stroke 1994;25(2):397–402. http://dx.doi.org/10.1161/01.STR.25.2.397.

[46] Bozzetto M, Remuzzi A, Valen-Sendstad K. Flow-induced high frequency vascular wall vibrations in an arteriovenous fistula: a specific stimulus for stenosis development? Phys Eng Sci Med 2024;47(1):187–97. http://dx.doi.org/10.1007/s13246-023-01355-z.

[47] McGah PM, Leotta DF, Beach KW, Aliseda A. Effects of wall distensibility in hemodynamic simulations of an arteriovenous fistula. Biomech Model Mechanobiol 2014;13(3):679–95. http://dx.doi.org/10.1007/s10237-013-0527-7.

[48] de Villiers AM, McBride AT, Reddy BD, Franz T, Spottiswoode BS. A validated patient-specific FSI model for vascular access in haemodialysis. Biomech Model Mechanobiol 2018;17(2):479–97. http://dx.doi.org/10.1007/s10237-017-0973-8.