# Mesh Neural Networks for $\mathrm{SE}(3)$-Equivariant Hemodynamics Estimation on the Artery Wall

Julian Suk, Pim de Haan, Phillip Lippe, Christoph Brune, Jelmer M. Wolterink

**Abstract**—Computational fluid dynamics (CFD) is a valuable asset for patient-specific cardiovascular-disease diagnosis and prognosis, but its high computational demands hamper its adoption in practice. Machine-learning methods that estimate blood flow in individual patients could accelerate or replace CFD simulation to overcome these limitations. In this work, we consider the estimation of vector-valued quantities on the wall of three-dimensional geometric artery models. We employ group-equivariant graph convolution in an end-to-end SE(3)-equivariant neural network that operates directly on triangular surface meshes and makes efficient use of training data. We run experiments on a large dataset of synthetic coronary arteries and find that our method estimates directional wall shear stress (WSS) with an approximation error of 7.6% and normalised mean absolute error (NMAE) of 0.4% while up to two orders of magnitude faster than CFD. Furthermore, we show that our method is powerful enough to accurately predict transient, vector-valued WSS over the cardiac cycle while conditioned on a range of different inflow boundary conditions. These results demonstrate the potential of our proposed method as a plugin replacement for CFD in the personalised prediction of hemodynamic vector and scalar fields.

**Index Terms**—Graph convolutional networks, group-equivariance, computational fluid dynamics, wall shear stress, coronary arteries.



## 1 Introduction

Computational fluid dynamics (CFD) is ubiquitous in science and engineering. In medicine, it allows for patient-specific, *non-invasive* estimation of functional quantities related to blood flow from static cardiac computed tomography (CT) [1] or magnetic resonance imaging (MRI) [2]. Hemodynamic scalar or vector fields (e.g. pressure or velocity) computed by CFD are valuable biomarkers for diagnosis [3], prognosis [4], or treatment planning in patients with cardiovascular disease [5]. For instance, the pressure drop after a stenosis, quantified as fractional flow reserve (FFR), can be calculated from CFD simulations and used as an indication for the need of percutaneous intervention [6]. Similarly, CFD simulation can be used to compute localised quantities on the artery wall. Wall shear stress (WSS), i.e. the force exerted by the blood flow on the artery wall in tangential direction, is a highly localised physical quantity that has been shown to correlate with local atherosclerotic plaque development and arterial remodelling in patients suffering from atherosclerosis [7]. Patient-specific local WSS values could be used to assess atherosclerosis risk in healthy, diseased, and stented arteries [8].

While CFD has a strong potential as an *in-silico* replacement for *in-vivo* measurement of hemodynamic fields, it also has some practical drawbacks. High-quality CFD simulations require fine discretisation of the spatial and temporal domains, leading to long computation times [1]. The time-intensive nature of high-fidelity CFD simulations limits their applicability in practice, e.g. for virtual surgery planning or shape optimisation of medical devices [9]. There is a practical need for fast but accurate estimation of hemodynamics.

Efforts to speed up CFD have focused on several directions. For example, model order reduction for fluid dynamics [10] can significantly speed up CFD simulation by reformulating the governing equations to an extent that preserves a satisfactory accuracy while enabling fast numerical solution. However, existing workflows have to be significantly altered to apply these methods, which requires considerable effort and expertise and hinders wide-spread applicability and acceptance. Alternatively, lumped-parameter models for the quantification of vascular hemodynamics have been proposed, e.g. for coronary circulation [11]. These replace parts of the spatial domain with 1D physical networks of resistance and capacitance that provide satisfactory local accuracy for the global flow. Such a method has the drawback that no localised information on the artery wall (e.g. WSS) is obtained.

Recent works have shown that there is great potential in machine learning to accelerate and improve cardiovascular biomechanics modelling [12]. In particular, deep neural networks can be used for hemodynamic scalar or vector field estimation. In contrast to CFD simulation, in which multiple systems of equations have to be iteratively solved *online* for each new artery, machine learning approaches move these time-consuming computations *offline*. To generate training data for a neural network, high-accuracy CFD simulations are run on geometric artery models. Once a neural network has been trained, hemodynamics estimation in a new artery


- *J. Suk, C. Brune and J. M. Wolterink are with the Department of Applied Mathematics & the Technical Medical Center, University of Twente, 7522 NB Enschede, The Netherlands.*
  *E-mail: {j.m.suk, c.brune, j.m.wolterink}@utwente.nl*
- *P. de Haan is with Qualcomm AI Research, Qualcomm Technologies Netherlands B.V., 6546 AS Nijmegen, The Netherlands (Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.) and QUVA Lab, University of Amsterdam, 1012 WX Amsterdam, The Netherlands.*
  *E-mail: pim@qti.qualcomm.com*
- *P. Lippe is with QUVA Lab, University of Amsterdam, 1012 WX Amsterdam, The Netherlands.*
  *E-mail: p.lippe@uva.nl*


*Preprint. Under review.*

only requires a single forward pass through the network, leading to significant speed-up.

Machine learning methods for hemodynamic parameter estimation can be subdivided into three categories. First, *parameterisation and projection* methods that re-parameterise or project the 2D artery-wall manifold from 3D to a Cartesian 1D or 2D domain and use multilayer perceptrons (MLP) or convolutional neural networks (CNN) on this domain. Second, 3D *point-cloud* methods that use MLPs on points representing the artery wall. Third, *mesh-based* methods that use graph convolutional network (GCN) architectures and incorporate information on artery-wall structure.

In the category of parameterisation and projection methods, Itu et al. [13] have used MLPs to estimate (scalar) FFR along the artery centerline based on shape descriptors, Su et al. [14] have used CNNs to estimate (scalar) WSS magnitude based on uniform shape sampling, Gharleghi et al. [15], [16] have used CNNs to estimate (scalar) time-averaged and transient WSS magnitude based on a cylindrical parametrisation of the vessel wall, and Ferdian et al. [17] have used CNNs to estimate vector-valued WSS based on a cylindrical parameterisation plus uniformly sampled projections of the velocity field at several distances from the artery wall of the aorta reconstructed from 4D flow MRI. Parameterisation and projection methods have the disadvantage that they cannot necessarily be adapted to more complex artery shapes and might fail in cases with severe pathology (e.g. aneurysms).

In contrast, point-cloud methods operate on native geometric representations of the artery. Point-cloud methods have been widely used for classification, detection, and segmentation tasks [18]. In hemodynamic field estimation, Liang et al. [19] used MLPs to estimate pressure and vector-valued velocity fields on 3D point clouds. Li et al. [20] estimated vector-valued hemodynamic fields using a method similar to PointNet [21] based on uniformly sampled points inside the cardiovascular lumen from CFD results. Even though point-cloud methods excel at learning spatial relations from geometric data, they disregard an important part of information that is available in surface representations of arteries: the surface connectivity and curvature.

Mesh-based approaches incorporate additional local geometry information from the mesh in addition to the point coordinates. Morales Ferez et al. [22] used the surface normal vector and connectivity to construct input features to a GCN predicting (scalar) endothelial cell activation potential on the left atrial appendage surface. A shortcoming of this approach is that the network predictions depend on the embedding of the mesh vertex normals in 3D Euclidean space but the quantity of interest only depends on the intrinsic shape of the mesh. Thus, predictions are sensitive to orientation of the input and shape alignment is required.

In this work, we propose a mesh-based approach that processes signals intrinsically on the artery wall (Fig. 1). The proposed method is informed by mesh properties and does not depend on the embedding of local geometry descriptors in 3D. Instead, it is invariant to translations and equivariant to rotations of the mesh. This means that vector-valued quantities like WSS rotate with the artery wall. This is data-efficient, as a single training sample covers all possible rotations and shifts of that artery and no data augmentation is required during training. Furthermore, our method is informed by anisotropic spatial interactions on the mesh, giving our filters high expressive capacity.

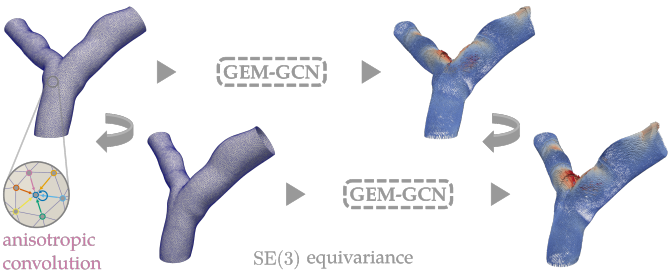


Fig. 1. **Overview.** We propose a gauge-equivariant mesh-graph convolutional network (GEM-GCN) to estimate discrete hemodynamic fields mapped to the vertices of a surface mesh of the artery wall. The GCN is powered by anisotropic (spatially-oriented) gauge-equivariant mesh (GEM) convolution with high filter expressivity. The combination of GEM convolution with appropriate input features leads to an end-to-end $\mathrm{SE}(3)$-equivariant neural network.

A preliminary version of this method was presented in [23], where we estimated steady-flow WSS with fixed boundary conditions. However, temporally multi-directional WSS acts as clinical biomarker for coronary plaque development [24] and different patients have distinct coronary blood flow which influences the WSS. Here, we substantially extend our method to also estimate pulsatile-flow WSS and to adapt its estimation based on a given boundary condition. We present results indicating that our GCN can perform some mild *extrapolation* beyond boundary conditions contained in the training data. Furthermore, we formally prove the empirical result that our method is end-to-end equivariant under rotation and translation, provide thorough experimental analysis on the influence of receptive field and sensitivity to remeshing, and include additional baseline experiments.

## 2 Data

We propose a general method for hemodynamic field estimation on artery walls and demonstrate its value in coronary arteries, which are a key application domain for CFD. We synthesise two distinct classes of representative 3D models with different topology (Fig. 2) for training and validation of our GCN. The first class consists of idealised, single-outlet arteries with stenoses at random locations. The second class consists of bifurcating arteries and is used to demonstrate the versatility of our method for more complex geometries as may be encountered in real-life.

### 2.1 Single arteries

Emulating the shapes used in [14], we generate synthetic coronary arteries with a single inlet and a single outlet (Fig. 2). The artery centerline is defined by control points spaced at fixed increments along the horizontal axis and random uniform increments along the vertical axis in a fixed 2D plane embedded in 3D. The resulting 3D models are symmetric to that plane. We assume that the lumen contour is circular and sample its base radius $r$ from a uniform distribution $r \sim \mathcal{U}(1.25, 2.0)$ mm, roughly corresponding

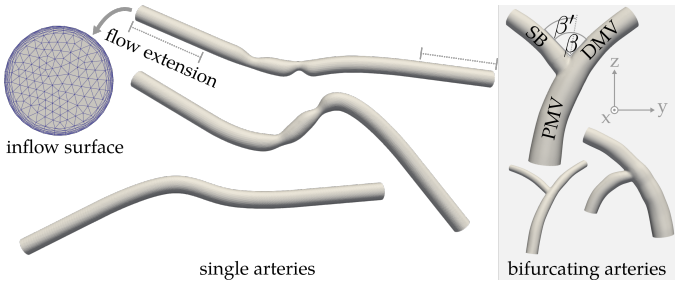


Fig. 2. **Artery datasets.** We develop and evaluate our method using two distinct classes of geometric models: synthetic single arteries (left) and bifurcating arteries modelled after the left main bifurcation of the coronary artery tree (right). The single arteries contain flow extensions to let the flow fully develop from a uniform inflow boundary condition. The bifurcating arteries are simulated with parabolic inflow and thus without flow extensions. They consist of the proximal main vessel (PMV) that branches into distal main vessel (DMV) and side branch (SB). Each bifurcation can be described by the angles $\beta$ and $\beta'$.

to [14]. We randomly introduce up to two stenoses which consist of a randomly determined narrowing of up to 50 % of the diameter, asymmetrically distributed between the top and the bottom vessel wall. The generated lumen contours are then lofted to create a watertight polygon mesh. The mesh is refined proportionally to the vessel radius along the artery centerline to give flow-critical regions finer spatial resolution for fluid simulation. Analogously to [14], we add flow extensions to the inlet and outlet, whose length is five times the vessel diameter. The shape synthesis is implemented using SimVascular [25].

### 2.2 Bifurcating arteries

We construct the bifurcating artery models using an atlas of coronary shape statistics [26], [27]. In the left main coronary bifurcation, the proximal main vessel (PMV) splits up into distal main vessel (DMV) and side branch (SB). The bifurcation can be fully described by the angles $\beta$ between centerlines of the branches DMV and SB and $\beta'$ between the bisecting line of the bifurcation and the centerline of SB (Fig. 2). We sample angles and lumen diameters from the atlas and use them to construct lumen contours. Appendix B provides a detailed overview of this process. Subsequently, the generated lumen contours are lofted to create a solid polygon model, merged, and meshed. After blending of the bifurcation region to produce a more natural transition, the final surface mesh is created in a refining meshing step. The entire shape synthesis is implemented with the SimVascular Python shell.

### 2.3 Blood-flow simulation

For each triangluar surface mesh (Sec. 2.1, Sec. 2.2), a tetrahedral volume mesh is created with five tetrahedral boundary layers (Fig. 2). We simulate steady and pulsatile blood flow in these meshes using the SimVascular solver for the three-dimensional, incompressible Navier-Stokes equations

$$\varrho\left(\frac{\partial u}{\partial t} + (u\cdot\nabla)u\right) = -\nabla p + \mu\Delta u$$
$$\nabla\cdot u = 0$$

where $u\colon \Omega \to \mathbb{R}^3$ is the fluid velocity and $p\colon \Omega \to \mathbb{R}$ is the pressure in the spatial domain $\Omega$ of the artery. Dynamic viscosity and blood density are assumed to be $\mu = 0.04\ \frac{\mathrm{g}}{\mathrm{cm}\cdot\mathrm{s}}$ and $\varrho = 1.06\ \frac{\mathrm{g}}{\mathrm{cm}^3}$, respectively. We model the blood vessel as rigid and apply a no-slip boundary condition, i.e. the velocity is zero at the lumen wall at all times. The inlet velocity profile is uniform for the idealised arteries and parabolic for the bifurcating arteries and follows a pulsatile waveform, scaled so that the coronary blood flow agrees with measurements in female and male patients (myocardial perfusion [28] times myocardial mass [29]). A constant heart rate of 80 $\frac{1}{s}$ is used across all simulations. We model the artery outlets of the bifurcating arteries as an RCR ("Windkessel") system consisting of proximal and distal resistances and intermediate capacitance. The total applied resistance and capacitance is tuned to agree with realistic values for pressure. The simplified boundary conditions for the steady simulations are $u_{\text{in}} = 20\ \frac{\mathrm{cm}}{\mathrm{s}}$ for the idealised and $u_{\text{in}} = 11.8\ \frac{\mathrm{cm}}{\mathrm{s}}$ for the bifurcating arteries as well as a pressure of $p_{\text{out}} = 100$ mmHg $\approx 13.332$ kPa weakly applied at the outlet, i.e. controlled by an outlet resistance.

The Reynolds number for the fluid flow is Re $\approx 700$ suggesting laminar flow. The WSS, which we denote as $\tau$, is defined as the force exerted on the lumen wall $\partial\Omega$ by the blood flow in tangential direction and can be computed from the resulting velocity field near the lumen wall. It linearly depends on fluid velocity $u$, assuming blood to be a Newtonian fluid:

$$\tau\colon \begin{cases} \partial\Omega \to T\partial\Omega \\ x \mapsto \mu\ \mathrm{J}_u(x)\vec{n}(x)|_{\perp\vec{n}} \end{cases}$$

where $T\partial\Omega$ denotes the tangent bundle of $\partial\Omega$, $\mathrm{J}_u$ the Jacobian of $u$, $\vec{n}\colon \partial\Omega \to \mathbb{R}^3$ the unit surface normal on the lumen wall and $\cdot|_{\perp\vec{n}}$ the perpendicular projection to $\vec{n}$.

The single-artery surface meshes have around 8,000 vertices and 17,000 triangular faces and the bifurcating artery surfaces meshes have around 17,000 vertices and 32,000 triangular faces. For an individual artery, steady-flow simulations take 10 to 24 min on an Intel Xeon Gold 5218 (16 cores, 22 MB cache, 2.3 GHz) and pulsatile-flow simulations take up to 1.6 h parallelised over 128 threads on a high-performance computing cluster. The resulting steady-flow datasets contain simulations for 2000 single arteries as well as 2000 bifurcating arteries. In addition, we generate a dataset of pulsatile-flow simulations in 731 single arteries. Note that the boundary conditions are fixed across samples and thus inherently encoded in these datasets. Therefore, we also generate pulsatile-flow datasets with varying boundary conditions, containing 187 and 117 geometric models for single and bifurcating arteries, respectively. In this set, simulations for each artery are run with five random-uniform coronary blood flow values from the interval $[1.87, 4.36]$ $\frac{\mathrm{ml}}{\mathrm{s}}$. We run additional simulations with two values from $[0.63, 1.87]$ $\frac{\mathrm{ml}}{\mathrm{s}}$ and $[4.36, 5.61]$ $\frac{\mathrm{ml}}{\mathrm{s}}$, respectively, for 19 single arteries. In total, our simulation data encompasses 5,035 CFD simulations with a total elapsed runtime of ca. 2800 h.

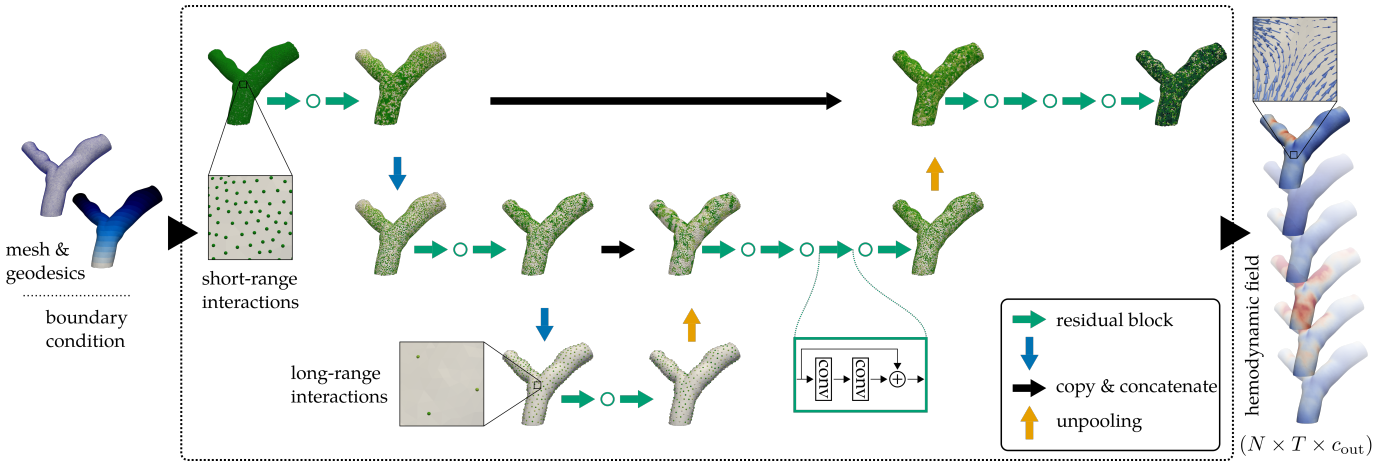


Fig. 3. **Network architecture.** Our mesh-based GCN outputs time-discretised, pulsatile hemodynamic fields $f^{\text{out}}\colon \mathcal{V} \to \mathbb{R}^{T \times c_{\text{out}}}$, where $|\mathcal{V}| = N$, subject to a (scalar) coronary blood flow parameter, given an input consisting of artery-wall mesh and vertex-wise geodesic distance to the artery inlet. A large receptive field is efficiently obtained using a three-level pooling scheme. To enable deep networks, we employ residual blocks consisting of two convolution modules and skip connection. The per-vertex colour of the signal before and after residual blocks corresponds to the scalar activation mapped to the vertices.

## 3 Learning on 3D Surface Meshes

We propose a neural network that can estimate hemodynamic fields in the data described in Sec. 2. At the core of our approach is the hypothesis that hemodynamics, in the laminar regime, depend in good approximation on local artery-wall curvature, flow direction, and flow boundary conditions. As is common in CFD (see also Sec. 2), we represent the artery wall as a triangular surface mesh. Let $\Omega \subset \mathbb{R}^3$ be the arterial lumen and $\partial\Omega$ its 2-dimensional boundary, the artery wall. The surface mesh $\mathcal{M}$ is a discretisation of $\partial\Omega$ that can be fully described by a tuple of vertices and faces $\mathcal{M} = (\mathcal{V}, \mathcal{F})$. We use the same mesh $\mathcal{M}$ from the CFD simulation to construct input features to a GCN which in turn outputs a scalar or vector for each vertex in the mesh, making use of local spatial interactions on the mesh $\mathcal{M}$ (Fig. 1).

### 3.1 Network architecture

We propose a mesh-based GCN that takes as input a scalar or vector field of features mapped to the vertices $f^{\text{in}}\colon \mathcal{V} \to \mathbb{R}^{c_{\text{in}}}$ and outputs scalar or vector-valued predictions $f^{\text{out}}\colon \mathcal{V} \to \mathbb{R}^{c_{\text{out}}}$ mapped to the same vertices. Fig. 3 visualises the network architecture used in our experiments. The GCN is composed of convolution and pooling layers. To enable the flow of long-range information across the manifold $\partial\Omega$, we opt for an encoder-decoder architecture with three pooling levels and "copy & concatenate" connections between corresponding layers in the contracting and expanding pathway. To prevent vanishing gradients, we use residual blocks consisting of two convolution layers and a skip connection. We use ReLU activation functions and employ batch normalisation before each activation.

### 3.2 Convolution layer

Convolution operators on meshes transform scalar or vector fields mapped to the mesh vertices $\mathcal{V}$. We define signals $f\colon \mathcal{V} \to \mathbb{R}^c$ with channel size $c$. For ease of notation, we compactly denote the set of all fields mapping from $\mathcal{V}$ to $\mathbb{R}^c$ as $\mathcal{X}(\mathcal{V}, \mathbb{R}^c)$ so that we can write $f \in \mathcal{X}(\mathcal{V}, \mathbb{R}^c)$. As a central building block of our neural network, we define convolution layers on $\mathcal{M}$ via message passing [30]. Let $c_i$ and $c_{i+1}$ denote the channel size before and after the layer.

$$(\phi * f)\colon \begin{cases} \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_i}) \to \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_{i+1}}) \\ f \mapsto \gamma(\phi(p, f)) \qquad \forall p \in \mathcal{V}. \end{cases}$$

The messages $\phi$ aggregate information from the neighbourhood $\mathrm{B}_r(p) \cap \mathcal{V}$, where $\mathrm{B}_r(p)$ consists of all vertices that are contained in the ball in Euclidean space with radius $r$ around $p \in \mathcal{V}$. The update function $\gamma$ creates the signal update from these messages. Alternatively, the neighbourhood could be defined by a 1-ring neighbourhood on the mesh $\mathcal{M}$ or by a geodesic ball on the manifold $\partial\Omega$. Our definition is an approximation to these options that is robust to varying mesh resolutions and scalable to large meshes. We construct convolution layers with kernel $K\colon \mathcal{V} \times \mathcal{V} \to \mathbb{R}^{c_i \times c_{i+1}}$ by choosing the messages

$$\phi(p, f) \coloneqq \sum_{q \in \mathrm{B}_r(p) \cap \mathcal{V}} K(p, q)\rho(p, q)f(q) \tag{1}$$

We refer to a neural network containing the aforementioned convolution layer as mesh-based GCN with the following rationale. The neighbourhood of a mesh vertex induces a set of graph edges $\mathcal{E}$ by connecting $p$ to all $q \in \mathrm{B}_r(p) \cap \mathcal{V}$. With this "latent" graph structure $(\mathcal{V}, \mathcal{E})$ we can make use of efficiently implemented graph deep-learning libraries (like PyG) to realise our layers. Additionally, this GCN can be **mesh-based** by explicitly incorporating face information in the message passing $\phi = \phi^{(\mathcal{V}, \mathcal{F})}$.

We distinguish between **isotropic** and **anisotropic** convolution layers based on kernel $K(p, q)$ and aggregation matrix $\rho(p, q)\colon \mathcal{V} \times \mathcal{V} \to \mathbb{R}^{c_i \times c_i}$. Intuitively, isotropic convolution filters process all signals mapped to the surrounding vertices in a neighbourhood in the same manner, while anisotropic filters process them distinctly.

**Definition 1** (Anisotropy)**.** We call bivariable functions $G\colon p, q \mapsto G(p, q)$ with $p \in \mathcal{V}$ and $q \in \mathrm{B}_r(p) \cap \mathcal{V}$ isotropic,

if $G(p,q)$ is constant in $q$. We call $G$ anisotropic, if $G(p,q)$ is not constant in $q$ for all $q \in \mathrm{B}_r(p) \cap \mathcal{V}$. Consequently, we call a layer anisotropic, if it contains any anisotropic function.

#### 3.2.1 Gauge-equivariant mesh convolution

Defining general anisotropic kernels $K(p,q)$ on meshes is difficult due to the lack of a local canonical orientation on the mesh: there is no obvious choice of reference vertex $q \in \mathrm{B}_r \cap \mathcal{V}$ in the filter support that canonically orients the local filter at $p$ for all $p \in \mathcal{V}$. To address this, we implement anisotropic kernels using gauge-equivariant mesh (GEM) convolution [31]. The idea behind GEM convolution is to recognise that possible kernel orientations are related by group actions of the symmetry group of planar rotations $\mathrm{SO}(2)$ and use this insight to spatially orient kernels "along" its group elements.

To achieve this, the signal $f \in \mathcal{X}(\mathcal{V}, \mathbb{R}^c)$ is composed of a linear combination of irreducible representations ("irreps") of the symmetry group $\mathrm{SO}(2)$, resulting in so-called $\mathrm{SO}(2)$ features. We can then choose an invertible parallel transport matrix

$$\rho(p,q) = \rho(p,q)_{(\mathcal{V},\mathcal{F})}$$

composed of group action representations that can rotate signals $f$ using mesh information. Specifically, the tangential plane at each vertex can be determined from the surrounding triangles and geodesic shortest paths between vertices can be found from adjacent faces [32]. Parallel transport refers to transporting signals along the manifold $\partial\Omega$ while maintaining a fixed angle to the shortest geodesic curve. It provides a unique and thus canonical transformation that allows linearly combining vector fields $f \in \mathcal{X}(\mathcal{V}, \mathbb{R}^c)$ at a vertex $p \in \mathcal{V}$ on the mesh. This is required for our notion of convolution Eq. (1).

On 2D manifolds $\partial\Omega$ embedded in 3D Euclidean space, picking a kernel orientation amounts to picking a locally tangential coordinate system ("gauge"). This choice can, on general manifolds, only be made arbitrarily. To prevent this to arbitrarily affect the outcome of the convolution, GEM convolution imposes an *equivariance* relation between layer input and output. Let $P$ and $P'$ be representations of the same (linear) gauge transformation that rotates the feature vector. GEM convolution requires message passing Eq. (1) to be equivariant under such transformations. Since all other variables in Eq. (1) are fixed, this imposes a linear constraint on the kernel $K(p,q)$ with solutions

$$\{K(p,q) \;\mid\; P'\phi(p,f) \overset{!}{\equiv} \phi(p,Pf)\}$$

A detailed derivation can be found in [31].

### 3.3 Pooling

Hemodynamics are characterised by long-range interactions across the artery wall $\partial\Omega$ and the lumen $\Omega$. Capturing these by stacking convolution layers, i.e. linearly increasing the receptive field, becomes infeasible for large and finely discretised surfaces. In contrast, pooling layers can exponentially increase the network's receptive field. Here, we use the mesh's "latent" computation graph $(\mathcal{V}, \mathcal{E})$ to implement pooling. Similar to the procedure used by Wiersma et al. [33], we sample a hierarchy of vertex subsets $(\mathcal{V} = \mathcal{V}_0) \supset \mathcal{V}_1 \supset \cdots \supset \mathcal{V}_n$ and construct according $r$-radius graph edges $\mathcal{E}_i$ encoding the filter support $\mathrm{B}_{r_i}(p) \cap \mathcal{V}_i$ for all $p \in \mathcal{V}_i$. Additionally, we find disjoint partitions of clusters

$$\bigcup_{p \in \mathcal{V}_{i+1}} C(p) = \mathcal{V}_i, \qquad \bigcap_{p \in \mathcal{V}_{i+1}} C(p) = \emptyset$$

that relate fine-scale vertices to exactly one coarse-scale vertex. This can be done with $k$-nearest neighbours ($k = 1$) by finding for each $p \in \mathcal{V}_i$ the nearest vertex in $\mathcal{V}_{i+1}$. Using these, a pooling operator can be defined as

$$\psi_{\text{pool}} \colon \begin{cases} \mathcal{X}(\mathcal{V}_i, \mathbb{R}^c) \to \mathcal{X}(\mathcal{V}_{i+1}, \mathbb{R}^c) \\ f \mapsto \frac{1}{|C(p)|} \sum\limits_{q \in C(p)} \rho(p,q) f(q) \quad \forall p \in \mathcal{V}_{i+1} \end{cases}$$

We implement unpooling by simply transporting signals $f$ back to their respective cluster locations:

$$\psi_{\text{unpool}} \colon \begin{cases} \mathcal{X}(\mathcal{V}_{i+1}, \mathbb{R}^c) \to \mathcal{X}(\mathcal{V}_i, \mathbb{R}^c) \\ f \mapsto \rho^{-1}(C^{-1}(p), p) f(C^{-1}(p)) \quad \forall p \in \mathcal{V}_i \end{cases}$$

### 3.4 Input features

We construct input features $f^{\text{in}} \colon \mathcal{V} \to \mathbb{R}^{c_{\text{in}}}$ with $c_{\text{in}}$ channels that describe the local shape of $\partial\Omega$ as well as global properties and are computed from the mesh $\mathcal{M}$. In particular, we compute a surface normal for each vertex $p \in V$ from adjacent mesh faces. We then construct three matrices that describe the local neighbourhood $q \in \mathrm{B}_r(p) \cap \mathcal{V}$ by, for each neighbour $q$, taking the outer products of

- the vector from $p$ to $q$ with itself,
- the surface normal at $q$ with itself, and
- the vector from $p$ to $q$ with the surface normal at $q$

For each of the three resulting sets of $(3 \times 3)$-matrices, we take the average over the neighbourhood. Two of these matrices are symmetric by construction, so we can drop entries without losing information. The radius $r$ of the local neighbourhood balls is a hyperparameter and must be chosen based on the structure of the input meshes, so that no neighbourhood is disconnected, i.e. consists of a single vertex. We chose the same radius that is used to construct the mesh's "latent" computation graph $(\mathcal{V}_0, \mathcal{E}_0)$.

The motivation behind these input features is that they define meaningful local surface descriptors that are not $\mathrm{SO}(2)$-invariant, a precursor to employing GEM convolution [31]. In contrast, the vanilla surface normal would simply be constant in any coordinate system induced by the surface normal. Since the surface normal describes the local surface (orientation) in an infinitesimally small neighbourhood $\mathrm{B}_{r \to 0}(p)$, i.e. the precise local curvature of the artery wall $\partial\Omega$, it is the preferred input feature for conventional message passing formulations.

We can extend the per-vertex features with any scalar or vector field. Since we assume that hemodynamics depend on flow direction, we append the shortest geodesic distance from each vertex $p$ to the inflow surface, which we compute with the vector heat method [34]. Moreover, we add global parameters such as blood-flow boundary conditions as a constant scalar field over the vertices.

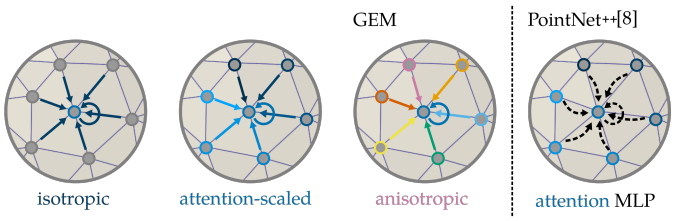


Fig. 4. **Filter comparison.** Isotropic, attention-scaled, and GEM convolution use kernels, in comparison to PointNet++ message passing. While attention-scaled convolution and PointNet++ both learn to distinguish neighbouring vertices through an attention mechanism, GEM convolution is equipped with a notion of direction.

### 3.5 Network output

We predict vector-valued hemodynamic quantities arising from transient, pulsatile flow by discretising a full cardiac cycle at $T$ points in time and let our neural network output a vector field $f^{\text{out}} \in \mathcal{X}(\mathcal{V}, \mathbb{R}^{3T})$. Alternatively, we can predict hemodynamic fields under steady flow by setting $T = 1$.

### 3.6 $\mathrm{SE}(3)$ equivariance

We model hemodynamics without the influence of gravity. Therefore, rigid rotation (or translation) of the domain should have no influence on the magnitude of the flow quantities and only change their direction. More precisely, our problem exhibits equivariance under $\mathrm{SE}(3)$ transformation. Inducing this symmetry in our neural network makes it oblivious to particular transformations which reduces the problem's complexity. We do so in the form of GEM convolution.

**Proposition 1.** (Informal) Composition of rotation-equivariant and translation-invariant input features with a gauge-equivariant mesh (graph) convolutional neural network (GEM-GCN) is end-to-end $\mathrm{SE}(3)$-equivariant.

GEM convolution layers define message passing intrinsically on the mesh $\mathcal{M}$ without dependence on the embedding in the ambient space, such as Euclidean vertex coordinates. $\mathrm{SO}(2)$ features can be expressed in ambient coordinates, which is done at the network output. Since tangential planes by definition rotate with the geometric model of the artery, the GEM convolution operator $(K * f)$ preserves $\mathrm{SE}(3)$ equivariance if the tangential input features move along with the surface.

Our input features $f^{\text{in}}$ are equivariant under rotation and invariant under translation of the mesh $\mathcal{M}$ by construction. Furthermore, our pooling and unpooling operators $\psi_{\text{pool}}$ and $\psi_{\text{unpool}}$ preserve $\mathrm{SE}(3)$ equivariance because they do not depend on the embedding of $\mathcal{M}$ in ambient space. Consequently, neural networks composed entirely of GEM convolution and pooling layers yield an end-to-end $\mathrm{SE}(3)$-equivariant operator together with our input features $f^{\text{in}}$ (proof in Appendix A).

### 3.7 Baseline models

We perform ablation studies to investigate the influence of the anisotropic aggregation matrix $\rho(p, q)$ and the anisotropic kernel $K(p, q)$ on prediction accuracy. To this end, we define two additional types of convolution (Figure 4): one fully isotropic and one with a learned anisotropic aggregation matrix. Additionally, we compare our method to another baseline model, PointNet++ [21], a point cloud method without explicit convolution kernels.

#### 3.7.1 *Isotropic convolution*

We construct purely isotropic convolution by choosing

$$\rho(p, q) = \rho \coloneqq \mathrm{I}$$
$$K(p, q) = K(p) \coloneqq \frac{1}{|\mathrm{B}_r(p) \cap \mathcal{V}|} W$$

in Eq. (1) where I is the identity matrix and $W \in \mathbb{R}^{c_i \times c_{i+1}}$ are trainable weights.

#### 3.7.2 *Attention-scaled convolution*

We construct anisotropic convolution with an isotropic kernel via a learned neighbourhood-attention mechanism by choosing:

$$\rho(p, q) \coloneqq \sigma((f(q) - f(p)) \cdot w)\mathrm{I}$$
$$K(p, q) = K(p) \coloneqq \frac{1}{|\mathrm{B}_r(p) \cap \mathcal{V}|} W$$

in Eq. (1) where $\sigma(\cdot)$ is the element-wise softmax activation and $W \in \mathbb{R}^{c_i \times c_{i+1}}$ as well as $w \in \mathbb{R}^{c_i}$ are trainable weights. This is equivalent to a graph attention layer [35] with separate weights and no LeakyReLU activation in the attention mechanism. Note that here, the message passing is not mesh-based and only depends on the vertices: $\phi = \phi^{\mathcal{V}}$.

In our definition of pooling in Sec. 3.3 we require the inverse of $\rho$ for the unpooling step. Since for attention-scaled convolution, $\rho$ may be ill-conditioned with diagonal elements close to zero, we fall back to using I for pooling.

#### 3.7.3 *PointNet++*

We compare kernel-based graph convolution to PointNet++ [21], a popular point cloud method consisting of message passing layers that redefine Eq. 1 by

$$\phi_k = \max_{q \in \mathrm{B}_r(p) \cap \mathcal{V}} \Theta_k(f(q), v_{p \to q})$$

where $k \leq c_{i+1}$ denotes the $k$-th component, $v_{p \to q}$ the Euclidean vector pointing from $p$ to $q$, and $\Theta \colon \mathbb{R}^{c_i} \times \mathbb{R}^3 \to \mathbb{R}^{c_{i+1}}$ an MLP of arbitrary depth. PointNet++ uses sampling and grouping operations that hierarchically sub-sample the graph vertices in the contracting pathway and interpolate in the expanding pathway. Note that, for PointNet++, choosing the same pooling architecture as for the kernel-based GCNs does not lead to the same level of accuracy, since the convolution paradigms are fundamentally different. Thus, we lay out PointNet++ separately, to achieve the best possible performance.

### 3.8 Quantitative evaluation

Quantitative results for WSS estimation are reported in terms of mean absolute error of the elements of $\triangle$, normalised by the maximum ground truth magnitude across the test split ("NMAE") and approximation error $\varepsilon \coloneqq \|\triangle\|_2 / \|L\|_2$. $\triangle$ is a vector whose elements are vertex-wise $\mathrm{L}^2$-normed differences between the network output

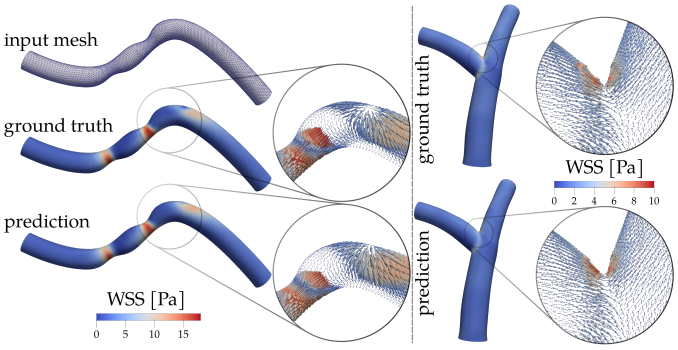


Fig. 5. **Steady-flow WSS estimation** of GEM-GCN on arteries of the held-out test splits of the single (left) and bifurcation artery (right) datasets.

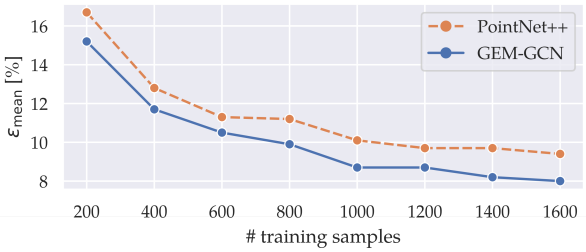


Fig. 6. **Mean approximation error** $\varepsilon_{\text{mean}}$ over the test split for different training set sizes on the steady-flow single-artery dataset. GEM-GCN weights are updated for ca. 10,000 iterations, PointNet++ weights for ca. 80,000 iterations.

$f^{\text{out}} \in \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_{\text{out}}})$ and ground truth label $l \in \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_{\text{out}}})$ so that the $i$-th element of vector $\triangle_i = \|f^{\text{out}}(p^i) - l(p^i)\|_2$ and $L_i = \|l(p^i)\|_2$ for $p^i \in \mathcal{V}$. Additionally, we report the maximum and mean vertex-wise difference, i.e. $\triangle^{\text{max}} = \max\{\triangle_i\}_i$ and $\triangle^{\text{mean}} = (\sum_i \triangle_i)/|\mathcal{V}|$ as well as the mean of the label statistics $\max\{L_i\}_i$ and $\text{median}\{L_i\}_i$ over the test set for scale.

## 4 Experiments and Results

We evaluate to what extent GEM-GCN can predict directional wall shear stress on the artery models described in Sec. 2. All datasets are split 80:10:10 into training, validation, and test splits, respectively. Network width and depth are set so that each neural network has around $1.02 \times 10^6$ trainable weights. All neural networks are trained by stochastic $L^1$-loss regression using an Adam optimiser with batches of 12 samples and a learning rate of $1 \times 10^{-3}$ on ground truth values obtained by CFD. All experiments are run on NVIDIA A40 (48 GB) GPUs. Parallelisation over two GPUs was necessary to fit batches of 12 bifurcating artery models into memory. Inference for a previously unseen artery wall takes less than 5 s including geometric pre-processing. Our open-source implementation in PyTorch and PyG using the vector heat method [34] can be found online.[1]

### 4.1 Steady-flow WSS estimation

We train GEM-GCN as well as the isotropic GCN (IsoGCN), the attention-scaled GCN (AttGCN), and PointNet++ (Sec. 3.7) to perform WSS estimation in the steady-flow single and bifurcating artery datasets. Fig. 5 shows examples of directional WSS prediction by GEM-GCN in a single and a bifurcating artery. The examples suggest that there is good agreement between ground truth and prediction. In particular, flow vorticity is captured well in the single artery model. The quantitative results in Table 1 show that GEM-GCN strictly outperforms IsoGCN and AttGCN on both the single and the bifurcating artery dataset. Moreover, the learned anisotropic convolution filters used in AttGCN achieve better performance than the isotropic filters used in IsoGCN. GEM-GCN and PointNet++ perform similarly in accuracy on the bifurcating artery dataset while GEM-GCN performs marginally better on the single arteries. Details to convergence and training time can be found in Appendix D.

[1] github.com/sukjulian/coronary-mesh-convolution

To investigate how much data is required to train our neural network, we evaluate how the amount of training data affects performance of GEM-GCN, as well as PointNet++ for comparison. Fig. 6 shows mean approximation error $\varepsilon_{\text{mean}}$ as a function of the number of training samples. For each training set size, GEM-GCN is trained from scratch on the single artery dataset, for a number of epochs chosen so that it receives ca. 10,000 gradient-descent updates. Since PointNet++ requires more epochs to converge we train it for 80,000 gradient-descent updates for comparison. The results in Fig. 6 indicate that both architectures can reach good accuracy with ca. 1000 training samples.

### 4.2 $\text{SO}(3)$ equivariance

GEM-GCN only depends on relative vertex features and is trivially invariant to translation. To empirically verify $\text{SO}(3)$ equivariance of GEM-GCN, we perform predictions on randomly rotated test samples. For this we use the neural network trained on the original, canonically oriented samples. The results in Table 1 show that rotation does indeed not affect performance of GEM-GCN. All quantitative metrics are nearly identical to those on the non-rotated samples up until numerical errors originating from discretisation of the kernels and activation function [31]. In contrast, results show that for PointNet++ (the best-performing baseline model) rotation of test samples drastically reduces prediction accuracy: performance drops from a mean NMAE of 0.5 % to 10.1 % for the single and 0.6 % to 7.8 % for the bifurcating artery dataset, respectively. This is expected as PointNet++ – like previously published models [19], [20], [22] – depends on the embedding of the mesh vertices in Euclidean space.

In order to make PointNet++ account for differently rotated samples, we re-train it with data augmentation by batch-wise, randomly sampling rotation matrices and applying them to the training samples. This is a common strategy for methods that lack rotation equivariance. Results show that training with this augmentation approximately recovers PointNet++'s accuracy to 0.7 % and 0.6 % mean NMAE for single and bifurcating arteries, respectively. This is slightly lower than before for the single arteries. However, training time until convergence is roughly 1.5 times longer, going from 20:48 [h] to 31:29 [h] and 35:01 [h] to 57:16 [h] for single and bifurcating arteries, respectively.

TABLE 1
**Quantitative evaluation** of prediction error for steady-flow WSS on synthetic single and bifurcating coronary arteries. The columns list the mean, median, and 75th percentile of NMAE, approximation error $\varepsilon$, maximum absolute error $\Delta^{\text{max}}$, and mean absolute error $\Delta^{\text{mean}}$ over the held-out test splits. Maximum and median WSS magnitude per dataset are indicated as $L_{\text{max}}$ and $L_{\text{median}}$, respectively. We additionally evaluate PointNet++ and GEM-GCN on randomly 3D-rotated test samples with previous training on canonically oriented samples (†). In the rotated case we additionally present accuracy metrics for PointNet++ for training on rotationally augmented data (‡).

| | | | NMAE [%] | | | $\varepsilon$ [%] | | | $\Delta^{\text{max}}$ [Pa] | | | $\Delta^{\text{mean}}$ [Pa] | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | mean | median | 75th | mean | median | 75th | mean | median | 75th | mean | median | 75th | |
| **Single arteries** | **oriented** | IsoGCN | 0.9 | 0.9 | 1.2 | 15.7 | 15.2 | 19.3 | 5.93 | 5.88 | 7.96 | 0.45 | 0.45 | 0.60 | $L_{\text{max}}$ = 22.53 [Pa]<br>$L_{\text{median}}$ = 2.07 [Pa] |
| | | AttGCN | 0.6 | 0.6 | 0.8 | 10.1 | 9.7 | 11.9 | 4.33 | 3.78 | 6.38 | 0.31 | 0.30 | 0.41 | |
| | | PointNet++ | **0.5** | **0.4** | 0.7 | 8.6 | 8.2 | 11.0 | 4.67 | 3.87 | 7.14 | 0.25 | **0.21** | 0.34 | |
| | | GEM-GCN | **0.5** | **0.4** | **0.6** | **7.8** | **7.6** | **9.1** | **4.10** | **3.55** | **6.13** | **0.23** | 0.23 | **0.31** | |
| | **rotated** | PointNet++† | 10.1 | 10.0 | 11.9 | 154.4 | 141.1 | 180.6 | 31.18 | 28.7 | 41.73 | 5.14 | 5.09 | 6.04 | |
| | | PointNet++‡ | 0.7 | 0.6 | 1.0 | 12.3 | 11.4 | 15.9 | 6.17 | 5.41 | 8.97 | 0.36 | 0.32 | 0.49 | |
| | | GEM-GCN† | **0.5** | **0.4** | **0.6** | **7.7** | **7.5** | **9.2** | **4.10** | **3.50** | **5.79** | **0.23** | **0.22** | **0.31** | |
| **Bifurcating arteries** | **oriented** | IsoGCN | 1.0 | 0.9 | 1.0 | 16.9 | 15.3 | 17.4 | 3.64 | 3.34 | 4.24 | 0.19 | 0.17 | 0.20 | $L_{\text{max}}$ = 7.16 [Pa]<br>$L_{\text{median}}$ = 1.37 [Pa] |
| | | AttGCN | 0.7 | 0.6 | 0.7 | 12.6 | 11.3 | 13.0 | 3.50 | 3.34 | 4.07 | 0.14 | 0.12 | 0.14 | |
| | | PointNet++ | **0.6** | **0.5** | **0.6** | **11.2** | **10.5** | **12.1** | **3.29** | **2.96** | 4.01 | **0.12** | **0.10** | **0.13** | |
| | | GEM-GCN | **0.6** | 0.6 | 0.7 | 11.9 | 11.3 | 13.0 | 3.38 | 3.25 | **3.92** | 0.13 | 0.11 | **0.13** | |
| | **rotated** | PointNet++† | 7.8 | 7.6 | 11.0 | 114.6 | 124.7 | 153.9 | 7.81 | 7.98 | 9.52 | 1.56 | 1.52 | 2.18 | |
| | | PointNet++‡ | **0.6** | **0.6** | **0.7** | 12.3 | 11.5 | 13.5 | 3.48 | 3.28 | 4.01 | **0.13** | **0.11** | **0.14** | |
| | | GEM-GCN† | **0.6** | **0.6** | **0.7** | **12.1** | **11.3** | **13.2** | **3.42** | **3.25** | **3.91** | **0.13** | 0.12 | **0.14** | |

† trained on canonically oriented samples
‡ trained under data augmentation (random rotation in 3D)

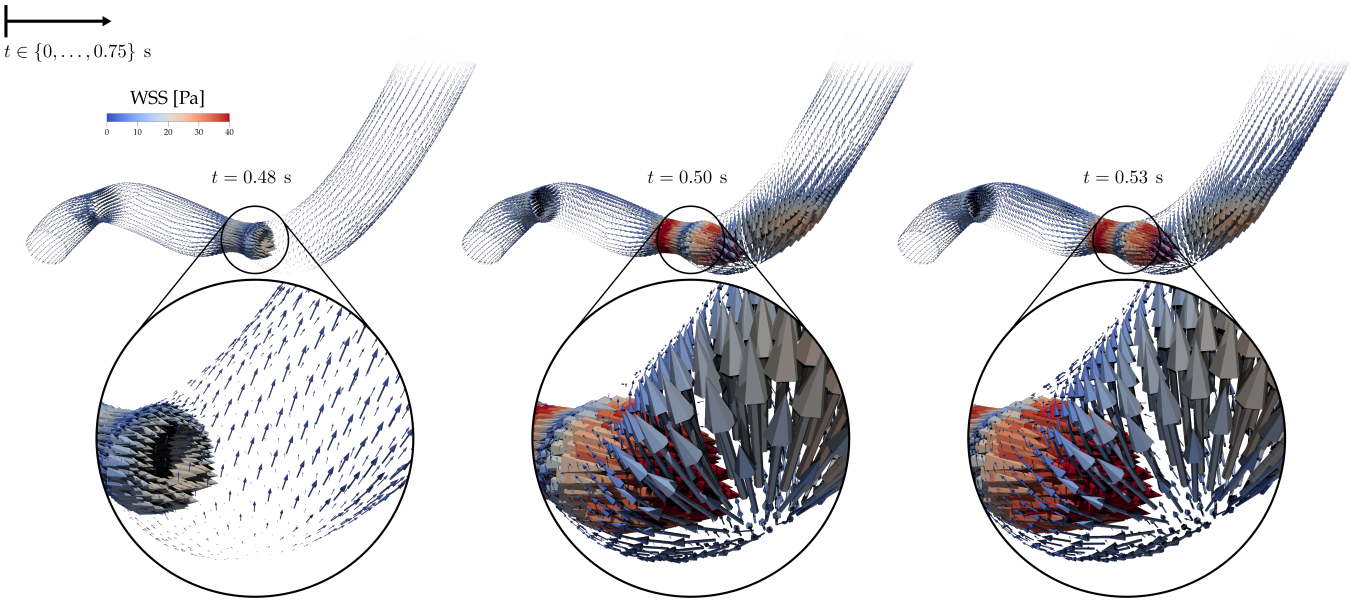


Fig. 7. **Pulsatile** WSS predictions for a single artery. GEM-GCN returns a vector field for each discrete point in time over one cardiac cycle.

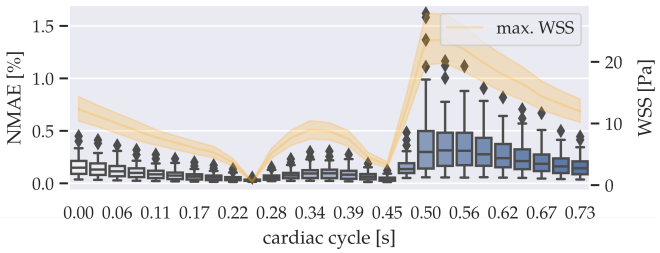


Fig. 8. **Pulsatile** single-artery WSS prediction error across the test split over time. NMAE is normalised by the maximum WSS magnitude over all samples in the test set over time (indicated in yellow) which follows a pulsatile waveform.

### 4.3 Pulsatile-flow WSS estimation

We train GEM-GCN for pulsatile-flow WSS estimation in single arteries with the modifications described in Sec. 3.5. Fig. 7 illustrates predictions of GEM-GCN on test samples for each discrete point in time in a cardiac cycle. In these experiments, WSS is dependent on both space and time. Therefore, we present estimation accuracy as time-dependent distributions in Fig. 8. The pulsatile-flow NMAE over time is comparable to the steady-flow NMAE, suggesting generally accurate predictions. However, the pulsatile-flow NMAE depends on the maximum WSS, which fluctuates over the cardiac cycle. As a consequence, the NMAE fluctuates as well and follows the pattern of the maximum WSS (indicated in yellow).

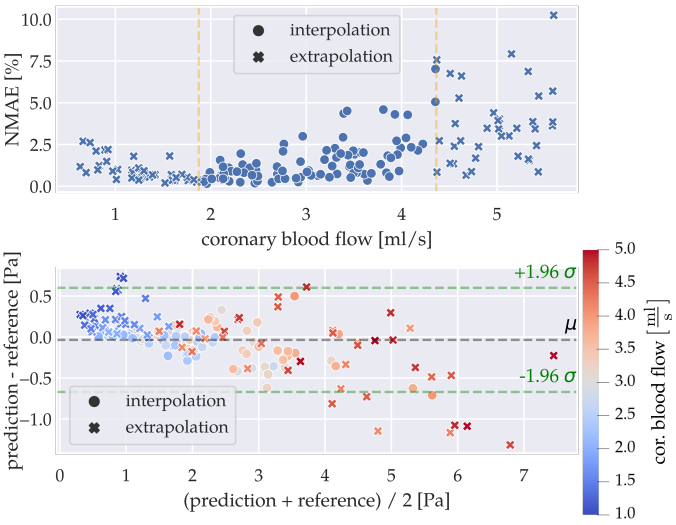


Fig. 9. **Conditional, pulsatile** single-artery WSS prediction accuracy, subject to changing coronary blood flow boundary condition. Scatter plot (top) shows NMAE over the boundary condition value. Bland-Altman plot (bottom) shows the difference between neural-network prediction and ground-truth reference over their average, collapsed into a scalar value per artery by taking the mean over xyz-components, time, and mesh vertices. The mean of the difference is denoted by $\mu$ and the standard deviation by $\sigma$. GEM-GCN is trained on boundary conditions in $[1.87, 4.36]$ $\frac{\text{ml}}{\text{s}}$. Beyond, neural-network predictions are extrapolated.

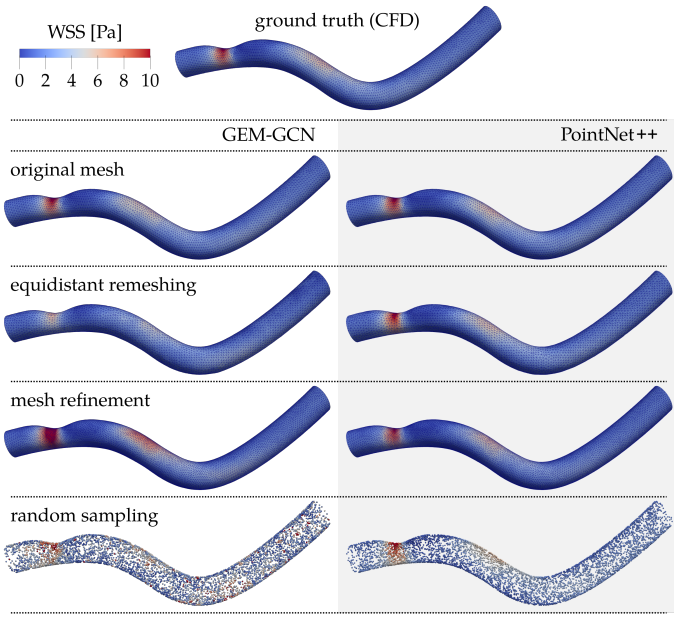


Fig. 10. **Sensitivity to remeshing.** GEM-GCN (left column) and PointNet++ (right column) trained on the original CFD mesh and evaluated on a differently remeshed artery wall $\partial\Omega$.

### 4.4 Incorporating boundary conditions

We re-train GEM-GCN on the dataset of pulsatile-flow WSS in single and bifurcating arteries, subject to varying coronary blood flow boundary conditions. We investigate interpolation between and extrapolation to different boundary conditions outside the limits of the training distribution: As described in Sec. 2, values in $[1.87, 4.36]$ $\frac{\text{ml}}{\text{s}}$ are contained in the training data and neural-network predictions subject to boundary conditions within this domain require interpolation. Values in $[0.63, 1.87]$ $\frac{\text{ml}}{\text{s}}$ and $[4.36, 5.61]$ $\frac{\text{ml}}{\text{s}}$ require extrapolation, as GEM-GCN is not trained on simulations subject to these inflow values. However, GEM-GCN can perform WSS prediction based on an arbitrary boundary condition. Here, we restrict our analysis to a discrete set of boundary conditions from a continuous range for which we have performed CFD simulation (Sec. 2).

Fig. 9 quantifies the prediction error for varying boundary conditions in two ways: First, we plot (mean) NMAE over coronary blood flow from which we observe the following: within the training range, the infimum of the NMAE displays a linear dependence on the boundary condition. The NMAE values corresponding to boundary conditions higher than this training range stay below this slope, while the NMAE values corresponding to lower values go above it. Second, we show a Bland-Altman plot comparing neural-network prediction and ground-truth reference. This plot shows that GEM-GCN overestimates WSS for low average magnitude and underestimates WSS for high average magnitude. A large amount of data points corresponding to extrapolation fall within the upper and lower bounds of the distribution of interpolated data points. From these two plots we conclude that GEM-GCN extrapolates to some extent to boundary condition values higher than those in the ground-truth distribution.

### 4.5 Sensitivity to remeshing

Recent works suggest that mesh neural networks might overfit to mesh connectivity [36]. For the problem of estimating hemodynamics on polygonal surface meshes this means that predictions are not independent of the sampling of vertex positions on the underlying manifold. To investigate the susceptibility of our models to overfitting, we let the trained GEM-GCN and PointNet++ networks described in Sec. 4.1 estimate WSS fields on three kinds of remeshed versions of the same surface $\partial\Omega$ of a sample from the test set of the single arteries:

1) We randomly sample vertices from $\partial\Omega$ and apply Poisson surface reconstruction, followed by an isotropic meshing procedure. This relaxes the mesh refinement around the stenoses and leads to approximately **equidistant** vertex spacing.
2) We globally **refine** the original mesh $\mathcal{M}$ so faces $\mathcal{F}$ have smaller edge lengths, while maintaining proportionally higher resolution around the stenoses.
3) We randomly sample mesh vertices from $\partial\Omega$, completely **randomising** vertex placement beyond refinement or coarsening. GEM-GCN extracts mesh information from the vertices and corresponding surface normals, which are well-defined here. Thus, we can do without an explicit mesh in this particular case.

The results in Fig. 10 suggest that GEM-GCN is still able to identify regions of interest on the surface $\partial\Omega$: in the equidistant mesh, it predicts high WSS magnitude in the stenosed area even with different mesh connectivity. However, GEM-GCN does overfit, to some extent, to mesh connectivity: regions of high vertex density, especially in the

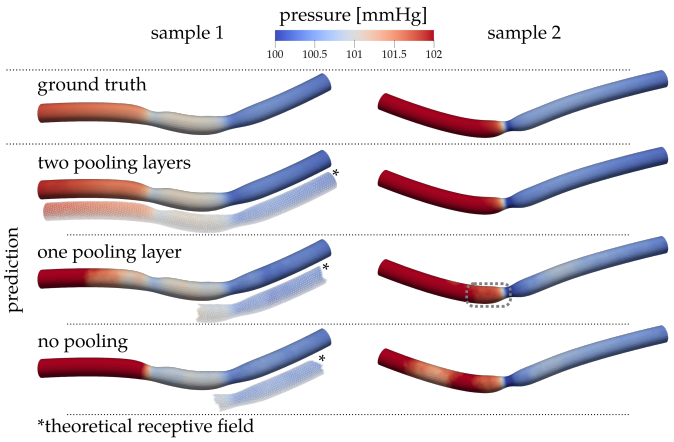


Fig. 11. **Influence of receptive field.** We train three GEM-GCN architectures with two, one, a no pooling layers on ground-truth pressure for two samples of the steady-flow single-artery dataset. For the sample in the left column, the theoretical receptive field is visualised.

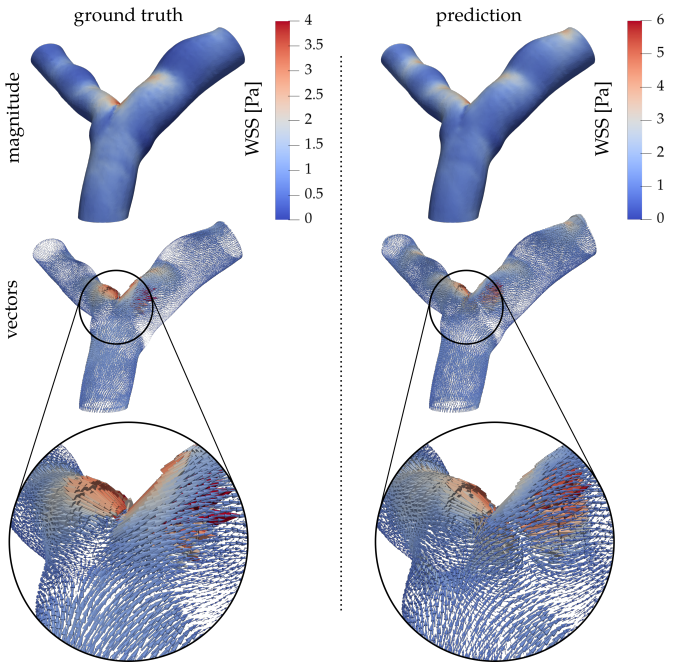


Fig. 12. **WSS prediction for patient-specific** left main coronary bifurcation. Ground truth (left) versus GEM-GCN prediction (right). To produce these results, GEM-GCN is trained purely on the synthetic (steady-flow) bifurcating-artery dataset. Note that the colourbars are in different scales to facilitate qualitative comparison. The colour and size of the WSS vectors scale with magnitude.

refined mesh, are predicted to have high WSS magnitude and vice versa. This might be because the training data has higher resolution around stenoses and WSS values are typically highest in stenotic regions. Thus, the network learns that high resolution corresponds to high WSS. The predictions on randomly sampled vertices show artifacts of this behaviour in the form of arbitrary peaks, caused by high local vertex density. This conditioning on resolution may be due to the aggregation scheme (see Equation (1)) used by GEM convolution: the filters sum over the vertex neighbourhoods, as opposed to e.g. taking the maximum. PointNet++ seems more robust to remeshing and random surface sampling, perhaps due to its maximum-aggregation (see Equation (3.7.3)) scheme.

### 4.6 Influence of receptive field

We use pooling and unpooling layers to access long-range information across the artery wall. Here, we investigate to what extent the resulting receptive field affects hemodynamics estimation in single arteries. WSS is a relatively local phenomenom: it mostly depends on the local geometry, and to a lesser extent on upstream morphology. Therefore, we expect that the effect of the receptive field will be less pronounced for WSS. In contrast, hemodynamic pressure is a global phenomenom: a stenosis in an artery will result in reduced pressure in all downstream locations. Thus, we consider pressure here, which is available from the same fluid simulations (Sec. 2). Note that since we apply a pressure boundary condition at the artery outlet, the outlet pressure is the same across simulations. Stenoses affect the upstream pressure rather than the downstream pressure in this simulated case.

Fig. 11 shows pressure predictions by three different GEM-GCN networks: the three-level architecture with two pooling layers and two unpooling layers described in Sec. 3.1, a two-level architecture with only one pooling and one unpooling layer, and a one-level architecture consisting solely of stacked convolution layers. All of these have the same number of convolution layers and ca. $1.02 \times 10^6$, $0.99 \times 10^6$, and $0.95 \times 10^6$ trainable parameters, respectively, due to the decreased channel size in absence of "copy & concatenate" connections. We train all models until convergence. The shown samples are chosen because they nicely display the effect that the receptive field can have on the neural-network prediction. For one of the samples, we visualise the theoretical receptive field that arises by recursively adding the filter support of each layer starting from a seed vertex. Quantitatively, we find that omitting the pooling levels results in mean approximation error $\varepsilon$ of 0.5 % and mean NMAE of 0.3 %. Using two pooling levels reduces these values to 0.4 % and 0.2 %, respectively. Our original architecture with three pooling levels achieves the lowest errors, at 0.1 % and 0.1 %, respectively. Considering Fig. 11 we conclude, since stenoses strongly influence the hemodynamic pressure locally, a large receptive field is necessary to convey this information upstream. When two stenoses occur in sequence, this effect overlays and we can observe a near-instant drop in accuracy where the receptive field fades.

### 4.7 Generalisation to real-life patient data

While we develop and evaluate our method on synthetic data, clinical application would be on anatomies extracted from individual patients. To assess generalisation to such data, we use the same GEM-GCN trained on the bifurcating arteries from Sec. 4.1 and let it predict WSS in a left main coronary bifurcation geometry extracted from a cardiac CT angiography scan [37]. We simulate blood flow with the same boundary conditions as in Sec. 2 to obtain ground-truth WSS which takes ca. 30 min. Fig. 12 shows the ground truth and estimated WSS vectors. As previously,

prediction and geometric pre-processing take less than 5 s. Even though GEM-GCN is trained exclusively on synthetic arteries, it produces a qualitatively plausible prediction. However, there is a considerable quantitative error which can be explained by the highly nonlinear dependence of blood flow on lumen wall shape: even small differences in morphology between the synthetic and real-life arteries can influence hemodynamics to an extent that cannot be easily extrapolated by GEM-GCN. Nevertheless, Fig. 12 suggests that GEM-GCN is able to qualitatively transfer the relation between local surface curvature and WSS.

## 5 Discussion and conclusion

We have presented an $\mathrm{SE}(3)$-equivariant GCN for the prediction of hemodynamic fields, operating on high-resolution surface mesh representations of the artery wall. Our results show that our method can learn to accurately predict vertex-wise, vector-valued, steady as well as pulsatile WSS in single and bifurcating, synthetic coronary arteries. Furthermore, results suggest that models can learn to inter- and extrapolate between and beyond coronary blood flow boundary conditions in the training set. Once trained, evaluation on unseen shapes takes less than 5 s, including geometric pre-processing. Our neural network is robust and flexible enough to be applied to a wide variety of different surface meshes. Fast estimation of physical quantities on meshes could benefit biomedical engineering applications where simulation data is abundant due to iterative fast prototyping, e.g. stent placement, but remains largely unused beyond a particular iteration. In this context, deep neural networks could estimate quantities of interest (e.g. oscillatory shear index (OSI)) for prototyping sub-iterations or supply initialisation for numerical solvers to speed up convergence of simulations.

While previous works on hemodynamics estimation using deep learning employed parameterisation and projection methods [13], [14], [15], [16], [17], our method operates natively on the geometric representation of the artery. In contrast to previous works featuring point-cloud methods [19], [20], we incorporate surface connectivity and curvature of the artery wall in our message passing. Furthermore, our method does not depend on the embedding of the mesh in Euclidean space, in contrast to previous work on mesh-based methods [22]. We have demonstrated in Sec. 3 how to exploit rotational and translational symmetry in our problem by an end-to-end $\mathrm{SE}(3)$-equivariant neural network. We leverage the control that kernel-based convolution, as compared to PointNet++ message passing, adds to the layers in gauge-equivariant convolution. In contrast, PointNet++ (Sec. 3.7) operates in 3D Euclidean coordinate space in which the geometric artery models are expressed. Thus, PointNet++ is implicitly conditioned on the embedding of the input mesh. The only way to correct for this in non-equivariant neural networks is to perform data augmentation during training, effectively adding redundancy. We have demonstrated in Sec. 4.2 that recovering the same accuracy as on registered input meshes requires longer training times and leads to lower accuracy. In fact, initial accuracy may never be fully recovered. Thus, when dealing with symmetric problems, GEM-GCN removes the need for roto-translational data augmentation and can lead to improved accuracy and data efficiency.

Data-driven estimation of hemodynamic fields on the artery wall requires large amounts of training data [12]. To learn how geometry and hemodynamic fields relate, the neural network needs access to a sufficiently large and representative dataset. In Sec. 4.1, we have quantified this requirement for GEM-GCN. Neural networks have previously been found to do well at interpolating, but poorly at extrapolating training data [12]. However, we have demonstrated in Sec. 4.4 that our method can to some extent extrapolate to different coronary blood flow boundary conditions. Our quantitative results have all been obtained on synthetic artery shapes and we have only provided preliminary results on a patient-specific artery in this work. Nevertheless, we have found that our method mildly generalises to real-life patient data. In future work, we aim to perform further validation on patient data with neural networks trained on synthetic data, which we can easily synthesise.

Additionally, we have investigated an important limitation of our method: accurate predictions require similar mesh connectivity, i.e. our method is sensitive to remeshing of the input surface. We hypothesise that this limitation can be alleviated by data augmentation. We find that PointNet++ is more robust to remeshing, so it can be an option if heterogeneous mesh size is more important than $\mathrm{SE}(3)$ symmetry. Furthermore, we see this as an opportunity for discretisation-independent neural networks, e.g. [36].

Our method is based on the observation that WSS and pressure, in the laminar regime, depend in good approximation on artery wall shape and boundary conditions *only*. This imposes a limitation on our work: in the turbulent regime, this hypothesis may be violated and thus our method would not be applicable. Furthermore, as in recent work by Gharleghi et al. [16], we let our neural network output hemodynamic fields over a complete cardiac cycle discretised into fixed time steps simultaneously rather than iterating from one time step to the next, since the cardiac cycle is periodic and clinically relevant in its entirety. This is limiting if we want temporally finer resolved WSS estimation. Extending our approach to volumetric meshes and time-step simulation in future works could enable us to incorporate physical relations based on fluid velocity as additional inductive bias.

Even though we have collected a large dataset of hemodynamic simulations in arteries, we had to be selective with the types of simulations to run. We did not include pulsatile-flow fixed-inflow simulations for the bifurcating arteries, due to their extensive computational demand. In future work we could add them, but for now we already have pulsatile-flow varying-inflow simulations for the bifurcating arteries and fixed-inflow simulations would have limited additional value.

In conclusion, we have shown that our proposed method can be a feasible plugin replacement for CFD for the task of fast, personalised estimation of hemodynamic quantities in high resolution on the artery wall.

## REFERENCES


[1] C. A. Taylor, T. A. Fonte, and J. K. Min, "Computational fluid dynamics applied to cardiac computed tomography for noninvasive quantification of fractional flow reserve," *J. Am. Coll. Cardiol.*, vol. 61, no. 22, pp. 2233–2241, Jun 2013.

[2] D. Steinman, "Image-based computational fluid dynamics modeling in realistic arterial geometries," *Ann. Biomed. Eng.*, vol. 30, pp. 483–487, Apr 2002.

[3] R. S. Driessen, I. Danad, W. J. Stuijfzand, P. G. Raijmakers, S. P. Schumacher, P. A. van Diemen, J. A. Leipsic, J. Knuuti, S. R. Underwood, P. M. van de Ven, A. C. van Rossum, C. A. Taylor, and P. Knaapen, "Comparison of coronary computed tomography angiography, fractional flow reserve, and perfusion imaging for ischemia diagnosis," *J. Am. Coll. Cardiol.*, vol. 73, no. 2, pp. 161–173, Jan 2019.

[4] M. Barral, I. El-Sanharawi, A. Dohan, M. Sebuhyan, A. Guédon, A. Delarue, A. Boutigny, N. Mohamedi, B. Magnan, S. Kemel, C. Ketfi, N. Kubis, A. Bisdorff, M. Pocard, and P. Bonnin, "Blood flow and shear stress allow monitoring of progression and prognosis of tumor diseases," *Front. Physiol.*, vol. 12, Aug 2021.

[5] B. Chung and J. R. Cebral, "CFD for evaluation and treatment planning of aneurysms: review of proposed clinical uses and their challenges," *Ann. Biomed. Eng.*, vol. 43, no. 1, pp. 122–138, Jan 2015.

[6] T. A. Fairbairn, K. Nieman, T. Akasaka, B. L. Nørgaard, D. S. Berman, G. Raff, L. M. Hurwitz-Koweek, G. Pontone, T. Kawasaki, N. P. Sand, J. M. Jensen, T. Amano, M. Poon, K. Øvrehus, J. Sonck, M. Rabbat, S. Mullen, B. D. Bruyne, C. Rogers, H. Matsuo, J. J. Bax, J. Leipsic, and M. R. Patel, "Real-world clinical utility and impact on clinical decision-making of coronary computed tomography angiography-derived fractional flow reserve: lessons from the ADVANCE registry," *Eur. Heart J.*, vol. 39, no. 41, pp. 3701–3711, Nov 2018.

[7] H. Samady, P. Eshtehardi, M. McDaniel, J. Suo, S. Dhawan, C. Maynard, L. Timmins, A. Quyyumi, and D. Giddens, "Coronary artery wall shear stress is associated with progression and transformation of atherosclerotic plaque and arterial remodeling in patients with coronary artery disease," *Circulation*, vol. 124, no. 7, pp. 779–788, Aug 2011.

[8] F. Gijsen, Y. Katagiri, P. Barlis, C. Bourantas, C. Collet, U. Coskun, J. Daemen, J. Dijkstra, E. Edelman, P. Evans, K. van der Heiden, R. Hose, B.-K. Koo, B. Krams, A. Marsden, F. Migliavacca, Y. Onuma, A. Ooi, E. Poon, H. Samady, P. Stone, K. Takahashi, D. Tang, V. Thondapu, E. Tenekecioglu, L. Timmins, R. Torii, J. Wentzel, and P. Serruys, "Expert recommendations on the assessment of wall shear stress in human coronary arteries: existing methodologies, technical considerations, and clinical applications," *Eur. Heart J.*, vol. 40, no. 41, pp. 3421–3433, Nov 2019.

[9] A. Marsden, "Optimization in cardiovascular modeling," *Annu. Rev. of Fluid Mech.*, vol. 46, no. 1, pp. 519–546, Jan 2013.

[10] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza, *Model Order Reduction in Fluid Dynamics: Challenges and Perspectives*. Springer, 2014, vol. 9, pp. 235–273.

[11] L. Itu, P. Sharma, V. Mihalef, A. Kamen, C. Suciu, and D. Lomaniciu, "A patient-specific reduced-order model for coronary circulation," in *IEEE 9th International Symposium on Biomedical Imaging (ISBI)*, 2012.

[12] A. Arzani, J.-X. Wang, M. Sacks, and S. Shadden, "Machine learning for cardiovascular biomechanics modeling: challenges and beyond," *Ann. Biomed. Eng.*, vol. 50, no. 6, pp. 615–627, Jun 2022.

[13] L. M. Itu, S. Rapaka, T. Passerini, B. Georgescu, C. Schwemmer, M. Schoebinger, T. Flohr, P. Sharma, and D. Comaniciu, "A machine learning approach for computation of fractional flow reserve from coronary computed tomography," *J. Appl. Physiol.*, vol. 121, no. 1, pp. 42–52, Jul 2016.

[14] B. Su, J.-M. Zhang, H. Zou, D. Ghista, T. T. Le, and C. Chin, "Generating wall shear stress for coronary artery in real-time using neural networks: feasibility and initial results based on idealized models," *Comput. Biol. Med.*, vol. 126, Nov 2020.

[15] R. Gharleghi, G. Samarasinghe, A. Sowmya, and S. Beier, "Deep learning for time averaged wall shear stress prediction in left main coronary bifurcations," in *IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, 2020.

[16] R. Gharleghi, A. Sowmya, and S. Beier, "Transient wall shear stress estimation in coronary bifurcations using convolutional neural networks," *Comput. Methods Programs in Biomed.*, vol. 225, no. 21, Oct 2022.

[17] E. Ferdian, D. J. Dubowitz, C. A. Mauger, A. Wang, and A. A. Young, "WSSNet: aortic wall shear stress estimation using deep learning on 4d flow MRI," *Front. Cardiovasc. Med.*, vol. 8, Jan 2022.

[18] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: a survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec 2021.

[19] L. Liang, W. Mao, and W. Sun, "A feasibility study of deep learning for predicting hemodynamics of human thoracic aorta," *J. Biomech.*, vol. 99, no. 138, Nov 2020.

[20] G. Li, H. Wang, M. Zhang, S. Tupin, A. Qiao, Y. Liu, M. Ohta, and H. Anzai, "Prediction of 3D cardiovascular hemodynamics before and after coronary artery bypass surgery via deep learning," *Commun. Biol.*, vol. 4, no. 1, Jan 2021.

[21] C. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, vol. 31, 2017.

[22] X. M. Ferez, J. Mill, K. A. Juhl, C. Acebes, X. Iriart, B. Legghe, H. Cochet, O. de Backer, R. R. Paulsen, and O. Camara, "Deep learning framework for real-time estimation of in-silico thrombotic risk indices in the left atrial appendage," *Front. Physiol.*, vol. 12, Jun 2021.

[23] J. Suk, P. de Haan, P. Lippe, C. Brune, and J. M. Wolterink, "Mesh convolutional neural networks for wall shear stress estimation in 3d artery models," in *Statistical Atlases and Computational Models of the Heart*, 2022.

[24] A. Hoogendoorn, A. Kok, E. M. J. Hartman, G. de Nisco, L. Casadonte, C. Chiastra, A. Coenen, S.-A. Korteland, K. van der Heiden, F. Gijsen, D. Duncker, A. V. D. van der Steen, and J. Wentzel, "Multidirectional wall shear stress promotes advanced coronary plaque development: comparing five shear stress metrics," *Cardiovasc. Res.*, vol. 116, no. 6, pp. 1136–1146, May 2020.

[25] H. Lan, A. Updegrove, N. M. Wilson, G. D. Maher, S. C. Shadden, and A. L. Marsden, "A re-engineered software interface and workflow for the open-source SimVascular cardiovascular modeling package," *Journal of Biomechanical Engineering*, vol. 140, 2018.

[26] P. Medrano-Gracia, J. Ormiston, M. Webster, S. Beier, A. Young, C. Ellis, C. Wang, Ö. Smedby, and B. Cowan, "A computational atlas of normal coronary artery anatomy," *EuroIntervention*, vol. 12, no. 7, pp. 845–854, Sep 2016.

[27] P. Medrano-Gracia, J. Ormiston, M. Webster, S. Beier, C. Ellis, C. Wang, Ö. Smedby, A. Young, and B. Cowan, "A study of coronary bifurcation shape in a normal population," *J. Cardiovasc. Transl. Res.*, vol. 10, no. 1, pp. 82–90, Dec 2017.

[28] M. B. Patel, L. P. Bui, R. L. Kirkeeide, and K. L. Gould, "Imaging microvascular dysfunction and mechanisms for female-male differences in CAD," *JACC: Cardiovasc. Imaging*, vol. 9, no. 4, pp. 465–482, Apr 2016.

[29] D. Corradi, R. Maestri, S. Callegari, P. Pastori, M. Goldoni, T. V. Luong, and C. Bordi, "The ventricular epicardial fat is related to the myocardial mass in normal, ischemic and hypertrophic hearts," *Cardiovasc. Pathol.*, vol. 13, no. 6, pp. 313–316, Nov 2004.

[30] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[31] P. de Haan, M. Weiler, T. Cohen, and M. Welling, "Gauge equivariant mesh CNNs: anisotropic convolutions on geometric graphs," in *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[32] N. Sharp and K. Crane, "You can find geodesic paths in triangle meshes by just flipping edges," *ACM Trans. Graph.*, vol. 39, no. 6, pp. 249:1–249:15, Dec 2020.

[33] R. Wiersma, E. Eisemann, and K. Hildebrandt, "CNNs on surfaces using rotation-equivariant features," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 92:1–92:12, Aug 2020.

[34] N. Sharp, Y. Soliman, and K. Crane, "The vector heat method," *ACM Trans. Graph.*, vol. 38, no. 3, pp. 24:1–24:19, Jun 2019.

[35] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[36] N. Sharp, S. Attaiki, K. Crane, and M. Ovsjanikov, "DiffusionNet: discretization agnostic learning on surfaces," *ACM Trans. Graph.*, vol. 41, no. 3, pp. 27:1–27:16, Mar 2022.

[37] J. M. Wolterink, T. Leiner, and I. Išgum, "Graph convolutional networks for coronary artery segmentation in cardiac CT angiography," in *Graph Learning in Medical Imaging*, 2019.

## APPENDIX A
## PROOF OF $\mathrm{SE}(3)$ EQUIVARIANCE (PROP. 1)

An $\mathrm{SO}(3)$ representation $(\mathbb{R}^c, \rho)$ is a vector space $\mathbb{R}^c$ with an $\mathrm{SO}(3)$ action $\rho : \mathrm{SO}(3) \to \mathbb{R}^c \times \mathbb{R}^c$. Let $\mathrm{SO}(2) \subset \mathrm{SO}(3)$ be the subgroup that leaves the z-axis invariant. The function $\rho : \mathrm{SO}(3) \to \mathbb{R}^c \times \mathbb{R}^c$ can be restricted $\rho|_{\mathrm{SO}(2)} : \mathrm{SO}(2) \to \mathbb{R}^c \times \mathbb{R}^c$ to give a representation of $\mathrm{SO}(2)$.

**Proposition.** Choose input and output $\mathrm{SO}(3)$ features $(\mathbb{R}^{c_{\text{in}}}, \rho^{\text{in}})$ and $(\mathbb{R}^{c_{\text{out}}}, \rho^{\text{out}})$, which are also $\mathrm{SE}(3)$ representations that are invariant to translations. Choose a neural network consisting of GEM convolution, the pooling defined in Sec. 3.3, gauge-equivariant activation functions [31], and parameters such that the input and output $\mathrm{SO}(2)$ features are $(\mathbb{R}^{c_{\text{in}}}, \rho^{\text{in}}|_{\mathrm{SO}(2)})$ and $(\mathbb{R}^{c_{\text{out}}}, \rho^{\text{out}}|_{\mathrm{SO}(2)})$. For a mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$ including a choice of gauge, let $F_{\mathcal{M}} : \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_{\text{in}}}) \to \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_{\text{out}}})$ denote the neural network.

For a transformation $g \in \mathrm{SE}(3)$, denote by $g\mathcal{M}$ the mesh where all the vertex positions are moved by the translation and rotation of $g$, and the normals and gauges are rotated by the rotation of $g$.

For vertex $p \in \mathcal{V}$, let $w_{\mathcal{M},p} \in \mathrm{SO}(3)$ be the rotation that maps the z-axis of (the ambient space) $\mathbb{R}^3$ to the normal vector of vertex $p$ and maps the x and y axes of $\mathbb{R}^3$ to the x and y axes on the tangent plane of vertex $p$, expressed in the choice of gauge. This is a basis transformation that maps from the global basis to a local basis at point $p$, consistent with the choice of gauge on the tangent plane. Applying this transformation for all vertices in $\mathrm{SO}(3)$ representation $(\rho, \mathbb{R}^c)$ gives an orthogonal linear transformation $\rho(w_{\mathcal{M}}) : \mathcal{X}(\mathcal{V}, \mathbb{R}^c) \to \mathcal{X}(\mathcal{V}, \mathbb{R}^c)$. Now define the composition $\tilde{F}_{\mathcal{M}} = \rho^{\text{out}}(w_{\mathcal{M}})^{-1} \circ F_{\mathcal{M}} \circ \rho^{\text{in}}(w_{\mathcal{M}}) : \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_{\text{in}}}) \to \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_{\text{out}}})$. This composition is equivariant:

$$\rho^{\text{out}}(g) \circ \tilde{F}_{\mathcal{M}} = \tilde{F}_{g\mathcal{M}} \circ \rho^{\text{in}}(g) \quad \forall g \in SE(3) \tag{2}$$

*Proof.* The network only depends on the mesh through the intrinsic quantities of the parallel transport and the logarithmic map, which are equal in $g\mathcal{M}$ and $\mathcal{M}$ expressed in the respective gauges. In particular, $g$ preserves distances and angles, so the neighbourhoods $N(p)$ remain fixed under $g$. Thus, the network is invariant $F_{\mathcal{M}} = F_{g\mathcal{M}}$. Furthermore, as the gauge rotates with the transformation, if $r$ is the rotational part of $g$, then $w_{g\mathcal{M},p} = w_{\mathcal{M},p} r^{-1}$ and thus $\rho(w_{g\mathcal{M}}) = \rho(w_{\mathcal{M}}) \circ \rho(g^{-1})$. Filling this in leads to

$$\tilde{F}_{g\mathcal{M}} = \rho^{\text{out}}(g) \circ \rho^{\text{out}}(w_{\mathcal{M}})^{-1} \circ F_{\mathcal{M}} \circ \rho^{\text{in}}(w_{\mathcal{M}}) \circ \rho^{\text{in}}(g^{-1}) \quad \square$$

**Remark.** In the above, we chose the gauge of the transformed mesh $g\mathcal{M}$ to equal the rotated gauge of the original mesh $\mathcal{M}$. By construction, GEM-GCN is equivariant to the choice of gauge, so any argument that holds for this case extends to the general case as well.

**Corollary.** GEM-GCN together with input features defined in Sec. 3 is $\mathrm{SE}(3)$-equivariant.

*Proof.* The input features defined in Sec. 3.4 can be expressed vertex-wise as a $3 \cdot 3 \cdot 3$ dimensional $\mathrm{SO}(3)$ representation, given by the elements of three $(3 \times 3)$ matrices:

$$m^1_{\mathcal{M}}(p) = \sum_{\mathrm{B}_r(p) \cap \mathcal{V}} \vec{v}_{p \to q} \vec{v}^{\mathsf{T}}_{p \to q}$$
$$m^2_{\mathcal{M}}(p) = \sum_{\mathrm{B}_r(p) \cap \mathcal{V}} \vec{n}_q \vec{n}^{\mathsf{T}}_q$$
$$m^3_{\mathcal{M}}(p) = \sum_{\mathrm{B}_r(p) \cap \mathcal{V}} \vec{v}_{p \to q} \vec{n}^{\mathsf{T}}_q$$

where $\vec{v}_{p \to q} \in \mathbb{R}^3$ is the vector pointing from $p$ to $q$ and $\vec{n}_q$ is the vertex normal at $q$. Combined, these form a feature $m_{\mathcal{M}} \in \mathcal{X}(\mathcal{V}, \mathbb{R}^{27})$ with a $\mathrm{SO}(3)$ representation that acts on each matrix by conjugation: $\rho(g)(m) = gmg^T$. This feature is equivariant: $m_{g\mathcal{M}} = \rho(g) m_{\mathcal{M}}$. When this feature is used as an input to the network, the output is equivariant by Prop. 1:

$$\tilde{F}_{g\mathcal{M}}(m_{g\mathcal{M}}) = \rho^{\text{out}}(g)(\tilde{F}_{\mathcal{M}}(m_{\mathcal{M}}))$$

$\square$

## APPENDIX B
## BIFURCATING ARTERY SYNTHESIS

The artery centerline of the parent vessel, PMV followed by DMV, is developed along seven control points and branches off into the child vessel SB at the fourth control point. The control points are evenly distanced $4$ [mm] apart. We construct the bifurcation in the y-z plane of a generic 3D coordinate system and sample two angles from the atlas [26] which together fully describe the bifurcation:

- $\beta \sim \mathcal{N}(\mu_\beta, \sigma_\beta^2)$ with mean $\mu_\beta = 78.9°$ and standard deviation $\sigma_\beta = 23.1°$ which is the angle between centerlines of the branches DMV and SB and
- $\beta' \sim \mathcal{N}(\mu_{\beta'}, \sigma_{\beta'}^2)$ with mean $\mu_{\beta'} = 61.5°$ and standard deviation $\sigma_{\beta'} = 21.5°$ which is the angle between the bisecting line of the bifurcation and the centerline of SB.

The angle $\beta'$ describes how much the bifurcation is skewed towards the child branch (Fig. 2). We place the control points so that the angle between the line connecting the fourth and fifth point and the z-axis is $\beta'$ for SB and $\beta - \beta'$ for DMV. For a more realistic curvature, the angles between the lines connecting the other control points and the z-axis are linearly inter- and extrapolated starting from zero at the origin. To add curvature in x-direction, we sample a third angle $\gamma$ from the atlas:

- $\gamma \sim \mathcal{N}(\mu_\gamma, \sigma_\gamma^2)$ with mean $\mu_\gamma = 9.5°$ and standard deviation $\sigma_\gamma = 21.5°$ which is the angle at which the PMV centerline enters the bifurcation plane.

We place the control points so that the angle between the line connecting the third and fourth point and the z-axis is $\gamma$ while linearly inter- and extrapolating the angles between the lines connecting the other control points and the z-axis, starting from zero. To avoid unrealistic curvature, none of these angles must exceed $90°$. The same (constant) curvature extends to both DMV and SB. It is anatomically unlikely for the LCX to curve upwards, so we restrict the SB to curve downwards. To arrive at the final centerline, the branching centerline path is smoothed using non-uniform rational basis splines (NURBS).

We model the vessel lumen with ellipse contours that are arbitrarily oriented in the plane normal to the centerline-curve tangent. The lumen radii are drawn from the coronary atlas [26]:

- $r_{\text{PMV}} \sim \mathcal{N}(\mu_{r_{\text{PMV}}}, \sigma_{r_{\text{PMV}}}^2)$ with mean $\mu_{r_{\text{PMV}}} = 1.75$ [mm] and standard deviation $\sigma_{r_{\text{PMV}}} = 0.4$ [mm]
- $r_{\text{DMV}} \sim \mathcal{N}(\mu_{r_{\text{DMV}}}, \sigma_{r_{\text{DMV}}}^2)$ with mean $\mu_{r_{\text{DMV}}} = 1.6$ [mm] and standard deviation $\sigma_{r_{\text{DMV}}} = 0.35$ [mm]
- $r_{\text{SB}} \sim \mathcal{N}(\mu_{r_{\text{SB}}}, \sigma_{r_{\text{SB}}}^2)$ with mean $\mu_{r_{\text{SB}}} = 1.5$ [mm] and standard deviation $\sigma_{r_{\text{SB}}} = 0.35$ [mm]

Medrano-Gracia et al. empirically show that the measured lumen diameters coincide best, i.e. at the lowest root mean square error $\varepsilon$ across samples, with a bifurcation law of the form

$$(d_{\text{PMV}})^a = (d_{\text{DMV}})^a + (d_{\text{SB}})^a + \varepsilon$$

where $a = 2.4$ [27]. As threshold we use the empirical root mean square error $\varepsilon = 0.165$ for the Huo-Kassab bifurcation law $a = \frac{7}{3}$, since it is the bifurcation law with the closest value $a$ reported in [27]. Accordingly, we choose values so that $\varepsilon \leq 0.165$ with the constraints that

- $r_{\text{PMV}} < r_{\text{DMV}}$ or $r_{\text{DMV}} < r_{\text{SB}}$, based on the intuitions that the parent vessel should be larger than the child vessel and should not grow after a bifurcation and
- $\frac{r_{\text{SB}}}{r_{\text{DMV}}} < 0.4$ according to empirical evidence from the atlas.

We observe that vessel diameter decreases approximately linearly with vessel length in the relevant interval and linearly decrease it towards the end to 87.5 % its initial size. To give the lumen a more realistic, non-smooth texture, we draw the contour ellipses' semi-minor and semi-major axes from a uniform noise distribution $\mathcal{U}(r-\delta, r+\delta)$ where $\delta = r\eta$ and $\eta = 5$ %.

## APPENDIX C

TABLE 2
**Dataset overview.** We run CFD simulations for synthetic single and bifurcating arteries for steady flow with fixed boundary condition, pulsatile flow with fixed boundary condition, and pulsatile flow with variable boundary conditions.

| | Single arteries | Bifurcating arteries |
|---|---|---|
| Steady flow | ✓ | ✓ |
| Pulsatile | ✓ | |
| Conditional | ✓ | ✓ |

## APPENDIX D

TABLE 3
**Training time until convergence** measured in number of epochs and wall-clock time of parallelised training on two NVIDIA A40 GPUs.

| | # epochs | | Wall-clock time [h] | |
|---|---|---|---|---|
| | single | bifurcating | single | bifurcating |
| IsoGCN | 1145 | 270 | 10:03 | 6:13 |
| AttGCN | 1470 | 660 | 31:18 | 30:48 |
| PointNet++ | 3920 | 1600 | 20:48 | 35:01 |
| GEM-GCN | 700 | 165 | 22:24 | 15:57 |
| PointNet++‡ | 6735 | 3330 | 31:29 | 57:16 |

‡ trained under data augmentation (random rotation in 3D)

Table 3 lists the number of epochs and the wall-clock time required for convergence, indicated by a plateau in training and validation loss, of GEM-GCN and PointNet++ on the full training set. We find that GEM-GCN converges in considerably fewer epochs than PointNet++. In terms of wall-clock time, GEM-GCN takes slightly longer than PointNet++ and considerably more time than IsoGCN for the single arteries, while AttGCN takes the most time. GEM-GCN has the highest ratio of wall-clock time to epochs because its message passing contains high-order tensor products (details in [31]) which require a lot of computational work. However, the included group symmetry acts as loss regularisation that accelerates the convergence in terms of epochs. We find that PointNet++ and AttGCN are disproportionally harder to train on the bifurcating artery dataset compared to GEM-GCN and IsoGCN. We have no proper explanation for this but hint at the fact that both PointNet++ and AttGCN work with neighbourhood attention scaling which seems to harmonise better with the single artery dataset.

## APPENDIX E

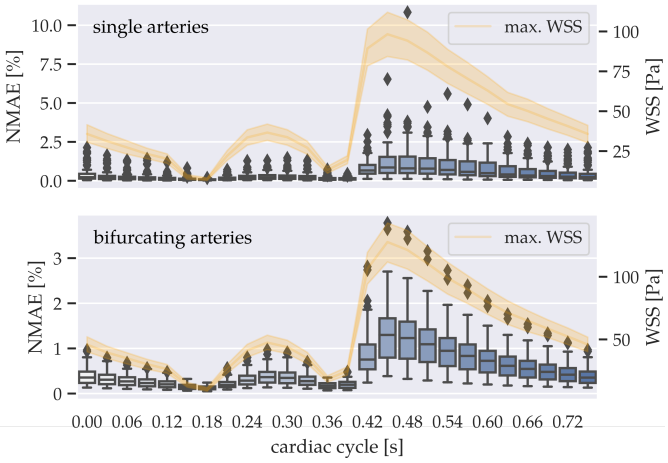


Fig. 13. **Conditional, pulsatile** single (top) and bifurcating (bottom) artery WSS prediction error across the test split over time. Five boundary conditions per artery are contained as separate data points. Maximum WSS magnitude is indicated in yellow.

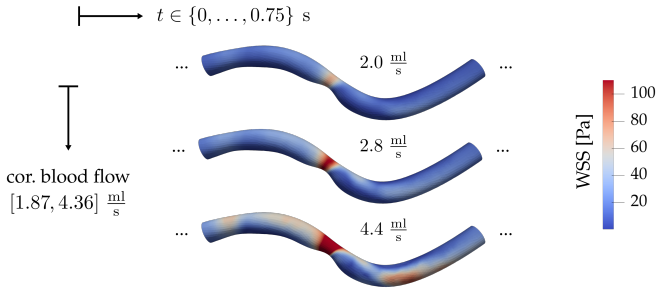


Fig. 14. **Conditional, pulsatile** single-artery WSS prediction. GEM-GCN is conditioned on coronary blood flow and maps boundary conditions to according wall shear stress.