Flow, Turbulence and Combustion
https://doi.org/10.1007/s10494-019-00100-5

The Discrete-Continuous, Global Optimisation of an Axial Flow Blood Pump

Krzysztof Tesch[1] · **Katarzyna Kaczorowska-Ditrich**[1]

Received: 25 June 2019 / Accepted: 27 October 2019 /

Abstract
This paper presents the results of the discrete-continuous optimisation of an axial flow blood pump. Differential evolution (DE) is used as a global optimisation method in order to localise the optimal solution in a relatively short time. The whole optimisation process is fully automated. This also applies to geometry modelling. Numerical simulations of the flow inside the pump are performed by means of the Reynolds-Average Navier-Stokes approach. All equations are discretised by means of the finite volume method, and the corresponding algebraic equation systems are solved by the open source software for CFD, namely OpenFOAM. Finally, the optimisation results are presented and discussed. The objective function to be maximised is simply pressure increase. The higher pressure increase the lower angular velocities required. This makes it possible to minimise the effect of haemolysis because it is mainly caused by high shear stresses which are related, among others, to angular velocities.

Keywords CFD · Global optimisation · Blood flow · Axial blood pump

1 Introduction

Since advanced medical treatment is usually not enough to prevent the further decline of patients with heart failure, two treatments of patients with such a disease can be distinguished, namely heart transplantation and artificial heart blood pumps. The former is somewhat difficult because of the relatively high costs and, what is even more important, lack of donor organs [1], not to mention organ rejection and mortality rates. The latter approach to the heart failure problem, i.e. artificial heart pumps, have gained popularity due to constantly improved design and features.

Several designs of artificial hearts are known. The most complicated aim to replace the ailing heart whereas simpler pumps are designed in order to support it. The supporting devices fall into three groups: LVAD (the left ventricular assist device), RVAD (the right

✉ Krzysztof Tesch
krzyte@pg.edu.pl

[1] Gdansk University of Technology, ul. G. Narutowicza 11/12, 80-233 Gdansk, Poland

ventricular assist device) and BIVAD (the bi-ventricular assist device) [2]. Despite the fact that the human heart's nature is pulsatile, continuous flow pumps are the most popular solutions. This is because of the simplicity and size of the device. Furthermore, we can distinguish centrifugal and axial blood pumps. The latter being smaller in comparison to the former. However, axial blood pumps require significantly higher angular velocities in order to increase outlet pressure. This may lead to blood damage, i.e. thrombosis and haemolysis [3–5] in particular. Haemolysis is mainly caused by high shear stress [6]. In order to minimise the effect of haemolysis an optimisation process of the axial pump is undertaken. The optimisation results lead to improvement in the reduction of the wall shear stresses.

According to Behbahani et al. [6] the design optimisation process is a fully automated iterative improvement process, in which CFD simulation is coupled with an optimisation algorithm. What is more, this process is in contrast to a design approach. This is because the latter includes a human expert in the iteration loop.

Most published instances of CFD optimisation involve only regional optimisation, without parameterising the geometry of the whole pump [7]. What is more, in various published works, part of the optimisation remained a manual process [7]. For instance, Zhu et al. [8] optimised only the diffuser of an axial flow blood pump, with a fixed-shape straightener and a rotor by means of commercial CAD, CFD and optimisation solvers. Derakhshan et al. [9] investigated an optimisation of a centrifugal pump with only six geometry variables with the help of ANN and an Artificial Bee Colony algorithm. Zhang et al. [10] optimised, or in fact rationalised, several geometry parameters such as diameters, heights and blade shapes. However, no optimisation algorithm and method are reported, suggesting a trial and error approach rather than an optimisation process. Gouskov et al. [11] optimised a circulatory support pump with only four geometry variables by means of the little know LP-Tau sequence generator regarded as a global optimisation algorithm. The whole process was performed in three stages involving initial pump geometry, meaning that the part of the optimisation remains manual. Another method was developed by Hai et al. [7] for Archimedes screw rotary blood pumps with guide vanes by means of a commercial optimisation tool and a commercial CFD package. However, several simplifications were introduced such as a constant shaft diameter. Also, the head and tail of the shaft were missing. The optimisation problem involved only seven design variables. Frazier et al. [12] modified an existing axial-flow pump by increasing its inducer-impeller inlet angle hence increasing its pressure responsivity. Korakianitis et al. [13] tested the performance of sixty two axial pump impellers with varying outlet angles and number of blades, i.e. only two design variables. Furthermore, the gathered data allowed the estimation of the optimal axial impeller geometry for any desired operating condition. The optimisation (rationalisation) method appears to be generate-and-test or the so called exhaustive search.

In this paper a fully automated iterative improvement process is discussed, coupling CFD simulations with an efficient global optimisation algorithm. The optimisation process is carried out by means of a slightly modified Differential Evolution (DE) algorithm, which is regarded as one of the current state-of-the-art algorithms [14] and typically forms the basis of the best performing algorithms. A simple and effective method of geometry modelling is proposed parameterising the geometry of the whole pump by means of fourteen variables. There is no need for any intermediate CAD software since the geometry is created directly by means of freely available GNU Octave and Octave Geometry scripts [15]. The same concerns the optimisation algorithms [16] available for everyone. A steady state solution is also compared with transient solutions in terms of pressure increases for various time steps and two different methods.

2 Geometry Description

The whole geometry consists of a rotor and stator and is described by fourteen parameters (design variables) $\mathbf{x} = \{x_1, \ldots, x_{14}\}$ listed in Table 1. First of all, the rotor blade shape is a non-linear helix given by

$$x(t) = R\cos t, \tag{1a}$$
$$y(t) = R\sin t, \tag{1b}$$
$$z(t) = f(t) \tag{1c}$$

where R stands for the radius of rotor blades. Further, the parameter $t \in [0; 2\pi x_1]$ is related to the rotor blade pitch $2\pi x_1$. The non-linearity $f(t)$ of the helix Eq. 1a is described by one parameter x_2 shown in Fig. 1.

Two of fourteen design variables are discrete, i.e. the number of rotor and stator blades. The former is named x_3 and the latter x_8. Next, the shape of the shaft is described by four points (x_4, x_5), (x_6, x_7) of the spline shown in Fig. 1. Stator blades are given by the so called camber line and the blade thickness. The former is represented by two variables (one point) (x_{11}, x_{12}) of the spline whereas the latter by two variables (two points) $(0, x_{13})$, $(1, x_{14})$, see Fig. 2. Finally, the two remaining variables x_9 and x_{10} are the stator blade upper and lower twist angles, respectively. The considered rotor radius R is 8 mm and the length of the shaft is 4.5 R.

The whole optimisation process is fully automated. This also applies to geometry modelling. Several random example geometries are shown in Fig. 3. What is important is that the proposed method is simple and effective. There is no need for any intermediate CAD software since the geometry is created directly by means of GNU Octave and Octave 'Geometry' script in a semi-discrete form (STL format).

Table 1 Constraints of variables and optimal values

Name	Constraints	Optimum	Description
x_1	[0.2; 1]	0.566	rotor blade pitch
x_2	[0.1; 0.5]	0.100	non-linearity of the helix
x_3	$\{2, \ldots, 4\}$	4	number of rotor blades
x_4	[0; 0.5]	0.251	two points of shaft's spline
x_5	[0.7; 1]	0.700	
x_6	[0.7; 1]	1.000	
x_7	[0.7; 1.2]	0.813	
x_8	$\{4, \ldots, 8\}$	8	number of stator blades
x_9	[0; 45°]	45.00°	stator blade upper twist angle
x_{10}	[0; 30°]	30.00°	stator blade lower twist angle
x_{11}	[0.2; 0.6]	0.600	point of stator shape blade spline
x_{12}	[0; 0.3]	0.300	
x_{13}	[0.05; 0.3]	0.300	point of stator thickness blade spline
x_{14}	[0.05; 0.2]	0.175	

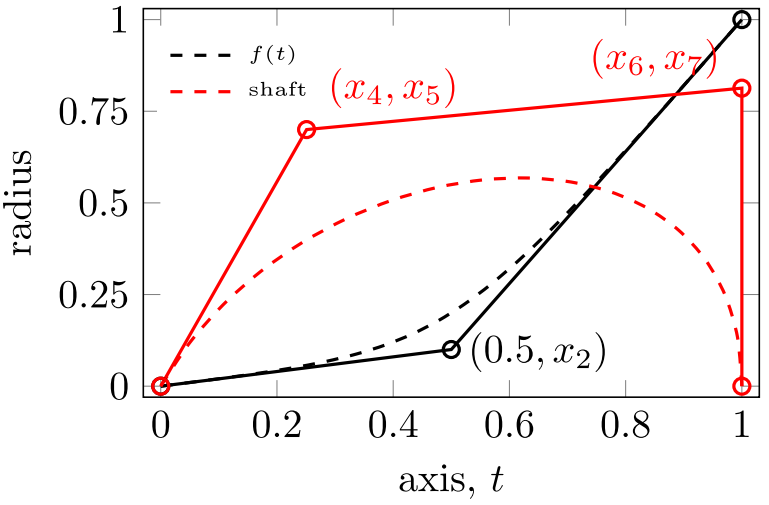

Fig. 1 Shaft and helix splines

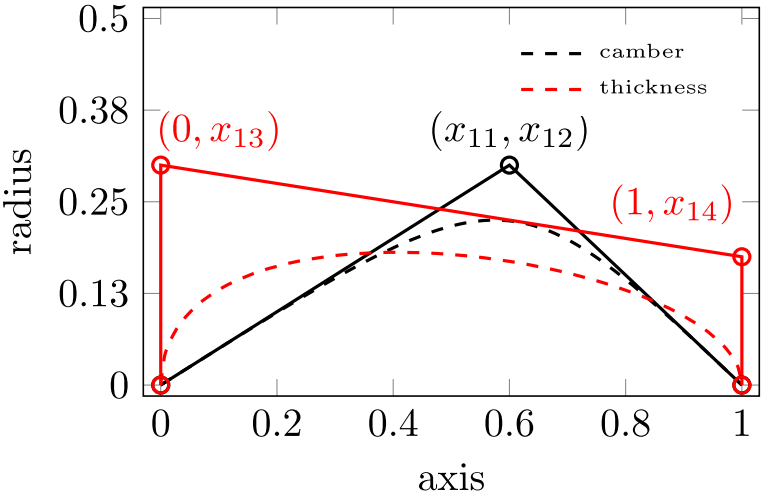

Fig. 2 Stator splines

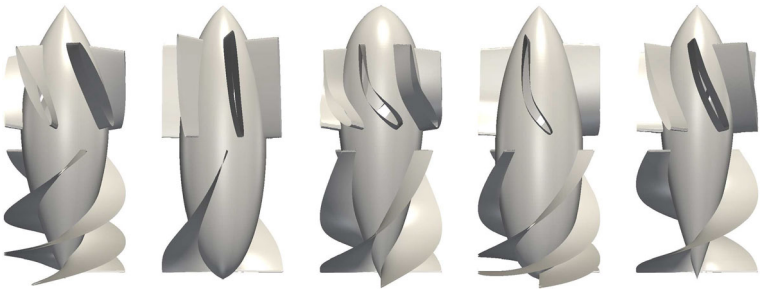

Fig. 3 Examples of randomly generated geometries

3 Blood Flow Modelling

3.1 Governing equations

Although blood is a suspension of blood cells in the plasma [17], the blood flow in large vessels as well as blood pumps can be regarded as a single-component and single-phase fluid, i.e. blood may be treated as a Newtonian fluid. This method is suitable for vessels larger than 0.1mm. If, however, the diameters are in the range of 0.1 to 1 mm, which is not the case here, then non-Newtonian constitutive equations are required in order to account for the non-Newtonian phenomena, such as shear thinning, yield stress and constant viscosity values at high shear rates.

Numerical simulation of the flow inside the pump is performed by means of the Reynolds-Average Navier-Stokes approach. A closed system of equations [18] for incompressible fluid, involving a mass conservation equation, the Reynolds equation and two additional transport equations for the two-equation SST model, is solved. An additional equation for the eddy viscosity ν_t is also necessary together with two blending functions F_1 and F_2 [19]

$$\nabla \cdot \bar{\mathbf{u}} = 0, \tag{2a}$$

$$\frac{\partial \bar{\mathbf{u}}}{\partial t} + \nabla \cdot (\bar{\mathbf{u}}\bar{\mathbf{u}}) = -\nabla \left(p\rho^{-1} + \tfrac{2}{3}k \right) + \nabla \cdot \left(2\left(\nu_t + \nu\right) \bar{\mathbf{D}} \right), \tag{2b}$$

$$\frac{\partial k}{\partial t} + \nabla \cdot (k\bar{\mathbf{u}}) = 2\nu_t \bar{\mathbf{D}}^2 + \nabla \cdot \left(\left(\nu_t \sigma_{k3}^{-1} + \nu \right) \nabla k \right) - C_\mu k\omega, \tag{2c}$$

$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\omega\bar{\mathbf{u}}) = \alpha_3 \omega k^{-1} 2\nu_t \bar{\mathbf{D}}^2 + \nabla \cdot \left(\left(\nu_t \sigma_{\omega 3}^{-1} + \nu \right) \nabla \omega \right) - \beta_3 \omega^2 + (1 - F_1) 2\omega^{-1} \sigma_{\omega 3} \nabla k \cdot \nabla \omega, \tag{2d}$$

$$\nu_t = a_1 k \max^{-1} \left(a_1 \omega, \sqrt{2\bar{\mathbf{D}}^2} F_2 \right). \tag{2e}$$

In the above equations $\mathbf{u}$ is the velocity vector, p – pressure, ρ – density, ν – kinematic viscosity coefficient and $\mathbf{D}$ – strain rate tensor. The two additional transport quantities are the kinetic energy of velocity fluctuations k and the turbulence frequency ω.

The shear stress transport (SST) model combines the k-ω model near the wall with the k-ε far from it. Constants marked with the subscript '3', namely σ_{k3}, $\sigma_{\omega 3}$, α_3, β_3 are linear combinations of constants from the component models, i.e. $C_3 = F_1 C_1 + (1F_1)C_2$. The additional constants are $a_1 = 0.31$, $C_\mu = 0.09$.

3.2 Equation discretisation

All the equations are discretised by means of the finite volume method, and the corresponding algebraic equation systems are solved by the open source software for CFD, namely OpenFOAM [20]. Since all the transport Eq. 2a–2e have common terms, the general transport equation for a quantity ϕ has the form of

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \mathbf{u}) = \nabla \cdot (\Gamma \nabla \phi) + S_\phi \tag{3}$$

where the overall source term S_ϕ should be linearised first, if necessary $S_\phi = S_u + S_p \phi$. The diffusivity for ϕ is denoted as Γ.

The integral version of the above transport equation over a control volume V_P can be expressed as

$$\frac{\mathrm{d}\phi_P}{\mathrm{d}t}|V_P| + \sum_f \phi_f \mathbf{u}_f \cdot \mathbf{S}_f = \sum_f \Gamma_f \left(\nabla\phi\right)_f \cdot \mathbf{S}_f + S_u|V_P| + S_p|V_P|\phi_P \tag{4}$$

where $|V_P|$ is a measure of V_P and $\mathbf{S}_f$ is a surface normal vector pointing outward. The considered convex control volume V_P around a centroid P consists of f planar surfaces S_f.

Divergence schemes include both convection terms $\nabla \cdot (\phi\mathbf{u})$ and other diffusive $\nabla \cdot (\Gamma\nabla\phi)$ terms and involve Gauss integration. The discretised convection term needs to be interpolated by means of cell centred values because the values ϕ_f are located at the face centroids. Limited linear interpolation is used (the steady state case) or linear upwind (the transient case), both being second order accurate. Further, the discretised diffusive terms involve surface normal gradients $(\nabla\phi)_f \cdot \mathbf{S}_f$, and are evaluated at a cell face that connects two cells. In order to maintain second order accuracy for non-orthogonal meshes, apart from orthogonal schemes, a non-orthogonal correction is considered.

The SIMPLE algorithm is used in order to solve pressure-velocity coupling, and the pressure equation is solved by means of the GAMG solver with the DIC smoother for the steady state version of the system Eq. 2a–2e. For the velocity fields and turbulent quantities standard solvers using a GS smoother are utilised. Under-relaxation factors are used in order to improve the stability of a solution. This is particularly important when solving steady-state flows. The assumed factors are 0.3 for pressure, 0.7 for velocity and 0.5 for the turbulent quantities k and ω. The transient version of the system is solved by means of the PISO [21] or PIMPLE (PISO + SIMPLE) algorithm, discussed further in Section 3.3.

The integrand $\frac{\mathrm{d}\phi_P}{\mathrm{d}t}$ of the left hand side of Eq. 4 is discretised by means of an implicit multi-level scheme

$$\frac{\mathrm{d}\phi_P}{\mathrm{d}t} = \frac{3\phi_P^{n+1} - 4\phi_P^n + \phi_P^{n-1}}{2\Delta t}. \tag{5}$$

What is more, this method is known to be second order accurate in time. Additionally, this approach is the so called three-level method because it requires the values of the unknown function ϕ_P at three different time steps, namely ϕ_P^{n+1}, ϕ_P^n and ϕ_P^{n-1}.

3.3 Space and temporal discretisation

The flow domain is divided into three parts, see Fig. 4 (bottom). Apart from the rotating rotor, two additional steady pipes (cannulae) are considered. The cannula shape optimisation

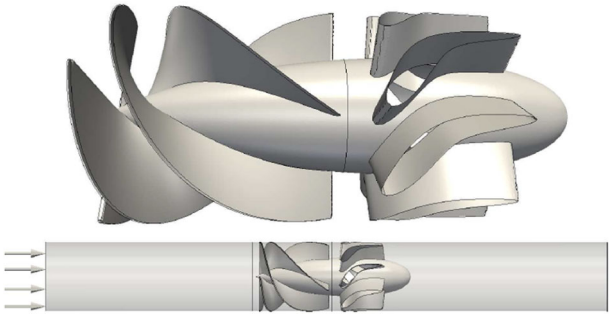

Fig. 4 Flow domain (bottom) and optimal pump geometry (top)

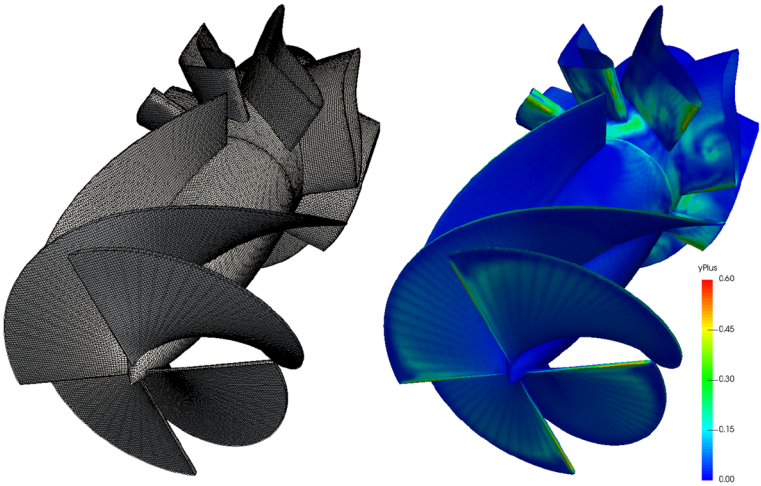

Fig. 5 Mesh (left) and y^+ distribution (right)

problem is a separate problem discussed elsewhere [14]. Furthermore, all three domains are discretised separately and merged by the so called arbitrary mesh interface (AMI). The total number of nodes is 1 689 534 and the total number of volumes is 1 495 877 where 1 381 669 of them are hexahedra. Formally, the mesh used may be classified as Cartesian, see Fig. 5 (left).

In order to make certain that the flow near the walls is properly resolved thin layers around the physical walls are generated. The quality of the mesh near the walls is inspected in terms of the maximal y^+ values. Maximal values of y^+ are below 1 for all the considered walls including the blades, see Fig. 5 (right). It has to be clarified, however, that the OpenFoam implementation of the k-ω family models (the SST among them) in the near wall region allows for a scalable wall function if $1 < y^+ < 300$ or no wall function if $y^+ < 6$. Two options are then possible and two were inspected giving negligible differences in terms of pressure rises. However, the former appears to be more stable which is crucial when it comes to a fully automated optimisation process.

Figure 6 demonstrates mesh convergence by showing the influence of the number of nodes on the pressure increase $\Delta p\, \rho^{-1}$. It is obvious that increasing the number of the mesh node above 2×10^6 has negligible effects on the pressure increase. This is because results are nearly constant above 2×10^6. Given that the mesh size is crucial from the CFD calculations perspective and thus optimisation time, the corresponding number of nodes was chosen at the level of 1.7×10^6.

A steady state solution (MRF) in terms of pressure increase is next compared with transient solutions for various time steps. Two methods are considered, namely PISO (Pressure-Implicit with Splitting of Operators) and PIMPLE. Figure 7 presents transient results in the form of pressure increase $\Delta p\, \rho^{-1}$ as a function of an angle of revolution (thin dashed lines). In order to compare the steady state (MRF) with transient solutions a time averaged has to be introduced first [22]

$$\overline{\frac{\Delta p}{\rho}} = \frac{1}{\Delta t} \int_{t_i - \Delta t}^{t_i} \frac{\Delta p(t)}{\rho} \mathrm{d}t \tag{6}$$

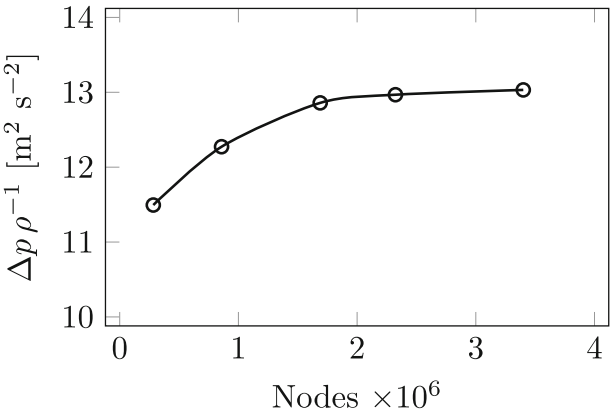

Fig. 6 Mesh convergence

where Δt represents the time of averaging (typically one revolution). If the time step of the transient CFD calculations is constant the integral in Eq. 6 can be approximated by the arithmetic mean resulting in

$$\overline{\frac{\Delta p}{\rho}} \approx \frac{1}{N} \sum_{j=i}^{i+N} \frac{\Delta p_j}{\rho}. \tag{7}$$

The number of time steps corresponding to the CFD time step is N. For instance, if the time step corresponds to the 4° angle of revolution then $N = 90$ and so on.

A time step convergence for the considered pump for five revolutions and two different approaches (PISO and PIMPLE) is shown in Fig. 7. As for the PISO algorithm five different

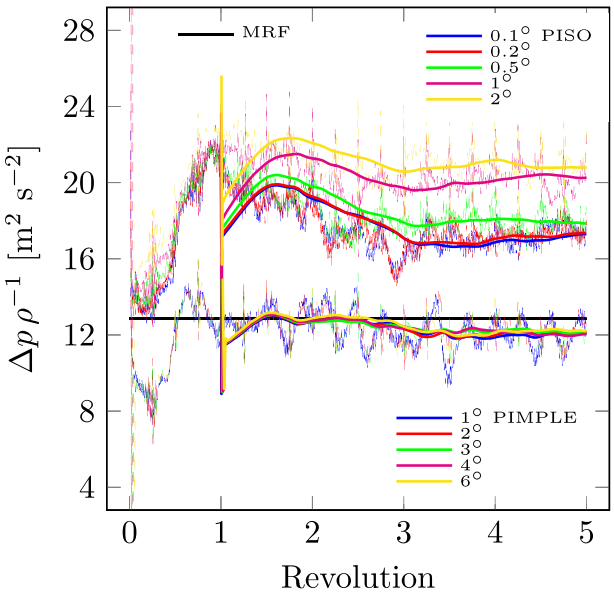

Fig. 7 Temporal convergence

cases are presented namely 2°, 1°, 0.5°, 0.2° and 0.1° time-step. Furthermore, the moving averages (the thick solid lines) of the pressure increase Eq. 7 are super-imposed. It is well visible that the PISO algorithm over-predicts the pressure rise in comparison to the steady solution (MRF – the solid black line). Decreasing the angle of the revolution below 0.2° has no effects on the pressure increase. In the second case, i.e. PIMPLE, the situation is completely different. It is evident that decreasing the temporal resolution from 1° to 6° per time-step has a negligible effect on the results in terms of pressure increase. What is more, the PIMPLE algorithm gives results similar to a steady solution (MRF). Thus, it can be concluded that stationary solutions are representative. This is exceptionally important from the point of view of the optimisation algorithm. This is because such CFD calculations must be performed repeatedly. Finally, the transient simulations require typically 3 full revolutions to reach the pseudo-periodic state.

3.4 Boundary conditions

The main boundary conditions are:

- Inlet. The specified constant volumetric flow rate $\dot{V} = 3$ dm/min is directed perpendicularly to the inlet surface accompanied by the zero normal gradient pressure. Low turbulence intensity is also considered. This means that the turbulence intensity is at the level of 1% and viscosity ratio $\nu_t/\nu = 1$ where $\nu = 3.3019 \times 10^{-6}\,\mathrm{m^2 s^{-1}}$ and the reference density $\rho = 1060\,\mathrm{kg\,m^{-3}}$.
- Outlet. The constant pressure distribution is assumed here together with zero gradient velocity for the flow out of the domain. This is because the outlet surface is located relatively far from the rotor.
- Walls. The so called no-slip conditions is applied meaning that impermeability and adhesion requirements are forced. Rotating wall velocity $n = 6000$ rev/min is considered in the rotating frame of reference. The flow in the near wall region is modelled by means of the scalable wall function.
- Interfaces. In order to allow for coupling between stationary pipes and rotating part of the pump the so called cyclic arbitrary mesh interfaces (AMI) are considered. A steady state approach is used rather than a full transient rotor-stator interaction as explained in Section 3.3. This is crucial for time consuming optimisation processes since CFD calculations need to be repeated hundreds of times. This approach is commonly known as a multiple reference frame (MRF) simulation.

4 Optimisation

4.1 General remarks

A discrete-continuous optimisation approach is needed to find the optimal shape. This is because two of fourteen design variables are discrete, namely the number of rotor and stator blades. The remaining variables such as the shaft, rotor and stator shapes as well as their thickness are continuous.

Metaheuristic procedures or more precisely Differential Evolution (DE) are used as a global optimisation method to localise the optimal solution in a relatively short time [16, 23]. Another commonly used term is nature-inspired metaheuristic. DE can be further classified

as multi-point (population based), derivative free optimisation algorithms. Most importantly, no additional information about the objective function is required. This is because DE is not problem-specific, stochastic algorithms with randomisation and local search. Additionally, randomisation is introduced through the probability of crossover. This makes it possible to efficiently explore the design space and escape local minima.

4.2 Objective function

The optimisation problem is to find a maximal pressure increase Δp. Since most optimisation algorithms are designed for the minimisation of the objective function, the pressure increase is considered with a minus sign

$$\Delta p_0 = \min_{\mathbf{x}\in\Omega\subseteq\mathbb{R}^D} \left(-\Delta p(\mathbf{x})\right). \tag{8}$$

Furthermore, the argument of the global minimum value of the objective function is expressed as

$$\mathbf{g} = \arg\min_{\mathbf{x}\in\Omega\subseteq\mathbb{R}^D} \left(-\Delta p(\mathbf{x})\right) \tag{9}$$

where $D = 14$ stands for the dimension of constraint space Ω or simply the so called optimisation domain. Thus, the objective function is subjected to box constraints listed in Table 1

$$\Omega = \left\{\mathbf{x} \in \mathbb{R}^D : L_i \leq x_i \leq U_i\right\} \tag{10}$$

where L_i and U_i are lower and upper bounds, respectively. Box constraints are regarded as a special case of inequality constraints. This type of constraint is commonly met in optimisation problems and does not need any special treatment.

4.3 Algorithm

Differential Evolution [24] is a simple, fast and effective metaheuristic algorithm. Furthermore, DE is regarded as the next step in evolution of the Genetic Algorithms (GA). Crossover and mutation are utilised on floating-point vectors. Additionally, selection is also present in DE. Most importantly, an explicit update equation is provided in contrast with GA.

Four main features of the algorithm can be distinguished, namely three different individuals selection, mutation, crossover and selection. Once three randomly individuals $\mathbf{x}_{a_1}$, $\mathbf{x}_{a_2}$, $\mathbf{x}_{a_3}$ are selected, a mutant vector $\mathbf{v}_i$ is generated according to the so called DE/Rand/1/Bin variant

$$\mathbf{v}_i := \mathbf{x}^n_{a_1} + F(\mathbf{x}^n_{a_2} - \mathbf{x}^n_{a_3}). \tag{11}$$

The scale factor F, or the so called differential weight, is used in order to control the rate of population development and is assumed here to be $F = 0.7$. Furthermore, the trial vector $\mathbf{y}_i$ is created via binomial crossover with probability $C = 0.9$ (line 10 in Fig. 8). The crossover probability regulates how much of the mutant vector is copied to the trial vector. It is possible to combine mutation and binomial crossover in a single vector equation by means of the Heaviside step (theta) function H. An auxiliary D-dimensional vector $\mathbf{K}$ consists of 0 and 1 (line 7 in Fig. 8)

$$\mathbf{K} := \mathrm{H}\left(C - \boldsymbol{\mathcal{U}}(0, 1)\right) \tag{12}$$

where $\boldsymbol{\mathcal{U}}(0, 1)$ stands for the continuous uniform distribution realisations characterised by its minimum 0 and maximum value 1. This approach is somewhat different in comparison to the original algorithm [23, 24]. What is more, it is now possible to combine mutation

```
   Input: C, F, N, n_max, L, U
   Output: g
1  for i := 0 to N − 1 do
2      x_i := L + (U − L) ∘ 𝒰(0, 1);
3      y_i := 0;
4  g := arg min_{x_i} f(x_i);
5  for n := 1 to n_max − 1 do
6      for i := 0 to N − 1 do
7          K := H(C − 𝒰(0, 1));
8          K_{𝒰{0,D−1}} := 1;
9          a := RandomPermutation({0, ..., N − 1} \ {i});
10         y_i := K ∘ (x_{a_3} + F(x_{a_1} − x_{a_2})) + (1 − K) ∘ x_i;
11     CheckRange(y);
12     for i := 0 to N − 1 do
13         x_i := arg min{f(x_i), f(y_i)};
14         g := arg min{f(g), f(y_i)};
```

Fig. 8 The vectorised differential evolution pseudocode

and crossover, into a single vector formula – line 10 in Fig. 8. Three different and randomly chosen individuals are indexed on the basis of a random permutation vector **a** (line 9 in Fig. 8). Additionally, line 8 corresponds to setting a random index of the vector **K** to 1 in order to guarantee that the $\mathbf{y}_i \neq \mathbf{x}_\mathbf{i}^\mathbf{n}$. Here $\mathcal{U}\{0, D-1\}$ represents the discrete uniform distribution realisations where 0 and $D-1$ are the parameters of the distribution. The selection step is shown in line 12 in Fig. 8. Simply, the best solution of the trial vector $\mathbf{y}_i$ and original individual $\mathbf{x}_\mathbf{i}^\mathbf{n}$, in terms of the objective function value, is passed onto a next generation $\mathbf{x}_i^{n+1}$. What is interesting is that this step is fully deterministic in contrast with mutation and crossover. The algorithm terminates if a given stop criterion is satisfied, i.e. the maximum number of objective function evaluations. Finally, a uniform random distribution is generated within the search domain with a random seed based on the clock (line 2 in Fig. 8). Lower $\mathbf{L} = \{L_1, \ldots, L_D\}$ and upper $\mathbf{U} = \{U_1, \ldots, U_D\}$ domain constraints are listed in Table 1 (the second column).

The presented algorithm [16] is almost identical in comparison to the original algorithm [24]. The differences include a vector representation, which is executed faster by mathematical packages such as GNU Octave. Furthermore, the presented version of the algorithm includes an option to check the range of variables (line 11 in Fig. 8), which allows to avoid non-physical configurations such as a negative number of blades.

4.4 Results

The DE population size N was set as 15, 20 and 30. The number of generations n_{max} was set as 15, 20, 30 for $N = 15$ and 20, 30, 40 for $N = 20$. Finally, the number of generations was set as 20, 30 for $N = 30$. This results in 225, 300, 450 objective function evaluations for $N = 15$ and 400, 600, 800 objective function evaluations for $N = 20$ and finally 600, 900 objective function evaluations for $N = 30$. Convergence can be monitored in Fig. 9 displaying box-and-whisker diagrams. The height of the error bars (boxes) is proportional to the interquartile range (IQR). The line inside the box is the second quartile (median). Further,

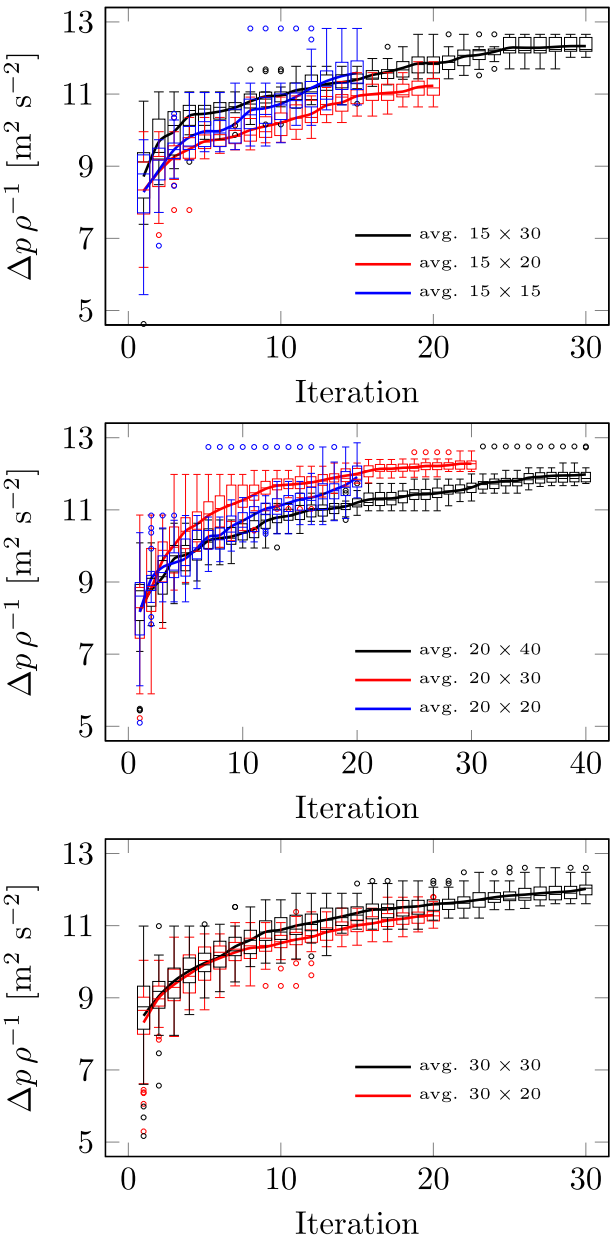

Fig. 9 Convergence for population sizes 15 (top), 20 (middle) and 30 (bottom)

the ends of the whiskers represent the data within 1.5 IQR of the lower quartile and upper quartile. Finally, the circles (outliers) represent data not included between the whiskers, i.e. the best and worst solutions (objective function values). Additionally, the solid lines denote the average pressure increase for the entire populations. The results of the calculations for

Table 2 Best solutions for different population sizes N and iteration numbers n_{max}

$N \times n_{max}$	$\Delta p\, \rho^{-1}\, [m^2 s^{-2}]$
15×15	12.819
15×20	11.896
15×30	12.656
20×20	12.820
20×30	12.633
20×40	12.757
30×20	11.797
30×30	12.605

different population sizes N and iteration numbers n_{max} are shown in Table 2. The best solutions were obtained for the 20×20 configuration of DE.

The individual computing time for calculating one objective function, i.e. geometry modelling, discretisation and CFD calculation, is about 15 minutes on a i7-6850K 3.60 GHz processor (3 out of 6 cores involved). For example, the calculation for the entire optimization process with a population size of $N = 20$ and a number of iterations of $n_{max} = 20$, i.e. 400 evaluations, takes about 4 days.

From Fig. 9 and Table 2 it arises that the best results are obtained for medium populations with the size N of 20 individuals. Furthermore, increasing the number of iterations n_{max} above 20 does not bring further improvement. This applies to both population sizes, i.e. $n_{max} = 20$ and $n_{max} = 15$. Populations with larger sizes, i.e. $N = 30$, require more iterations to obtain results comparable to smaller iteration numbers n_{max}. This is important due to the time-consuming CFD calculations. Out of all the optimisation processes, the best results are obtained for $N = 20$ and $n_{max} = 20$. Due to the stochastic nature of the DE algorithm, it is obvious that restarting the algorithm may give slightly different results, which, however, are typically close to the optimal solution.

The optimal values of $\mathbf{g}$ are listed in Table 1. The upper part of Fig. 4 presents the optimal pump shape according to Table 1. It may be observed that the optimal geometry consists of 4 rotor and 8 stator blades being the upper accessible ranges. As for the remaining variables, most of them are located on the boundary of the considered box constraints which is a typical situation during constrained optimisation. It is worth mentioning that some constraints in Table 1 allow us to avoid non-physical configurations. Other constraints require preliminary optimisations or at least geometry creation and check.

The average shape evolution for the selected generations (iterations) for the 20×20 DE, reflecting convergence process, is presented in Fig. 10. It is interesting that the first shape (iteration 1) is simply an arithmetical average of purely random shapes. This is because the initial population is randomly generated. At the same time, the last shape (iteration 20) is similar to the optimal shape since the population here is nearly uniform. Finally, since these shapes are arithmetical averages of individual generations, it should be noted that none of these has been subject to any CFD calculations.

The upper part of Fig. 11 displays the best shape evolution resulting from the shape optimisation process ($N = 20$, $n_{max} = 20$). The last shape (iteration 20) is the optimal solution shown also in Fig. 4. Interestingly, all shapes apart from the second (iteration 2) are similar in terms of rotor configurations. What distinguishes the second geometry from the remaining three is the shape of the shaft and stator blades in particular. The lower part of Fig. 11 presents the corresponding wall shear stresses (WSS) distributions. The higher the

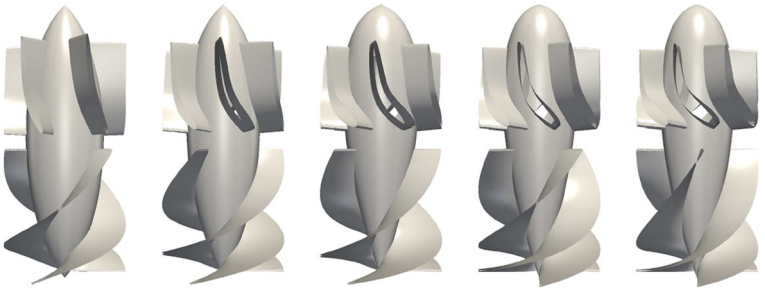

Fig. 10 The average shape evolution for $N = 20$ and $n_{max} = 20$ (iteration number 1, 6, 12, 17 and 20)

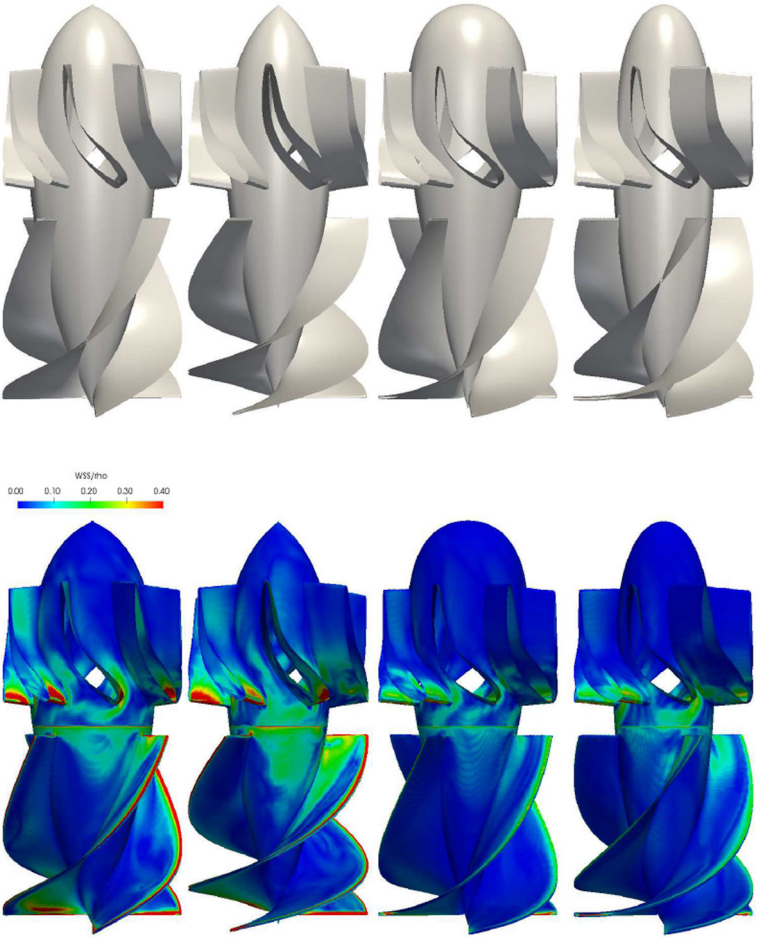

Fig. 11 Best shape evolution and corresponding wall shear stresses distributions for $N = 20$ and $n_{max} = 20$ (iteration number 1, 2, 7 and 20)

iteration number the smoother the distribution on the shaft and blades. Also, the maximum values of WSS are lower. One has to keep in mind, however, that high shear stresses are related primarily to angular velocities.

5 Conclusions and Limits of the Current Study

5.1 Conclusions

- The results of a discrete-continuous global optimisation process of an axial flow blood pump are presented. The optimal results are achieved by means of DE in a relatively short time. Since 2 of 14 design variables are discrete (integer) special care of DE is necessary.
- A simple and effective method of geometry modelling is proposed. This makes it possible to make the whole optimisation process fully automated. There is no need for any intermediate CAD software since the geometry is created directly by means of the GNU Octave and Octave Geometry scripts.
- A steady state solution is compared with transient solutions for various time steps. It is shown that the PISO algorithm always over-predicts the pressure rise in comparison to the steady solution. Unlike PISO the second method, i.e. the PIMPLE algorithm gives results similar to a steady solution which is important from the point of view of the optimisation algorithm. Typically, three full revolutions are necessary to reach the pseudo-periodic state.
- Figure 11 presents the wall shear stress distribution which is responsible for haemolysis. It may be observed that the highest values are localised, among other, on the rotor and stator blade tips. The higher pressure increase the lower angular velocity required. Thus the optimisation process leads to improvement in the reduction of the wall shear stresses and the effect of haemolysis. This is because haemolysis is mainly caused by high shear stresses which are related, among others, to angular velocities.

5.2 Limits of the current study

It should be mentioned that many ventricular assist devices designs are also equipped with guide vanes (flow straighteners) not only at the outlet, but also at the inlet of the device. This kind of flow straightener is missing in the current study. However, taking into account the presence of a straightener would not change the algorithm itself, but only extend the calculation time. The same concerns the gap sizes between rotor and housing.

One has to also keep in mind that the current optimisations will, most likely, not be sufficient to develop an efficient and biocompatible blood pump. It is well known that the wall shear stress is responsible for the phenomenon of haemolysis. In addition to haemolysis, another phenomenon is exposure time. Considering the above criteria, if at all possible, could lead to other, less effective solutions than those presented in this paper. One possible solution is multi-objective optimisation that would take into account both pressure increases, haemolysis and exposure time. The first problem is the much longer multi-objective optimisation time in comparison with the single-objective problem. The main problem is the availability of a CFD model allowing an accurate prediction of haemolysis in such complex flows (stagnations, recirculations). This problem is constantly being analysed and works devoted to it are still being published [7, 25, 26]. According to Hai et al. [26] there is no clear guideline concerning the applicability or the limitations of the studied models. What

is more, a greater problem is the lack of well documented experimental data, adequate for a validation of the discussed models [26]. It seems that one of the possible solutions to this problem is an experimental stand that is currently being prepared. The experiments carried out together with biologists will most likely answer the above questions. This issue will be addressed at length in a separate work.

Compliance with Ethical Standards

Conflict of interests The authors declare that they have no conflict of interest.

References

1. Kafagy, D.H., Dwyer, T.W., McKenna, K.L., Mulles, J.P., Chopski, S.G., Moskowitz, W.B., Throckmorton, A.L.: Design of axial blood pumps for patients with dysfunctional fontan physiology: computational studies and performance testing. Artif. Organs **39**(1), 34–42 (2015)
2. Carr, C.M., Jacob, J., Park, S.J., Karon, B.L., Williamson, E.E., Araoz, P.A.: CT of left ventricular assist devices. RadioGraphics **30**(2), 429–444 (2010)
3. Aaronson, K.D. et al.: Use of an intrapericardial, continuous-flow, centrifugal pump in patients awaiting heart transplantation. Circulation **125**(25), 3191–3200 (2012)
4. Rogers, J. et al.: Intrapericardial left ventricular assist device for advanced heart failure. N. Engl. J. Med. **376**, 451–460 (2017)
5. Slaughter, M. et al.: HeartWare ventricular assist system for bridge to transplant: combined results of the bridge to transplant and continued access protocol trial. J. Heart Lung Transplant. **32**(7), 675–683 (2013)
6. Behbahani, M., Behr, M., Hormes, M., Steinseifer, U., Arora, D., Coronado, O., Pasquali, M.: A review of computational fluid dynamics analysis of blood pumps. Eur. J. Appl. Math. **20**, 363–397 (2009)
7. Yu, H., Janiga, G., Thévenin, D.: Computational fluid dynamics-based design optimization method for Archimedes screw blood pumps. Artif. Organs **40**(4), 341–352 (2016)
8. Zhu, L., Zhang, X., Yao, Z.: Shape optimization of the diffuser blade of an axial blood pump by computational fluid dynamics. Artif. Organs **34**, 185–192 (2010)
9. Derakhshan, S., Pourmahdavi, M., Abdolahnejad, E., Reihani, A., Ojaghi, A.: Numerical shape optimization of a centrifugal pump impeller using artificial bee colony algorithm. Comput. Fluids **81**, 145–151 (2013)
10. Zhang, Y., Zhan, Z., Gui, X.M., Sun, H.S., Zhang, H., Zheng, Z., Zhou, J.Y., Zhu, X.D., Li, G.R., Hu, S.S., Jin, D.H.: Design optimization of an axial blood pump with computational fluid dynamics. ASAIO J. **54**, 150–155 (2008)
11. Gouskov, A.M., Lomakin, V.O., Banin, E.P., Kuleshova, M.S.: Minimization of hemolysis and improvement of the hydrodynamic efficiency of a circulatory support pump by optimizing the pump flowpath. Biomed. Eng. **51**(4), 229–233 (2017)
12. Frazier, O.H., Khalil, H.A., Benkowski, R.J., Cohn, W.E.: Optimization of axial-pump pressure sensitivity for a continuous-flow total artificial heart. J. Heart Lung Transplant. **29**(6), 687–691 (2010)
13. Korakianitis, T., Rezaienia, M.A., Paul, G.M., Avital, E.J., Rothman, M.T., Mozafari, S.: Optimization of axial pump characteristic dimensions and induced hemolysis for mechanical circulatory support devices. ASAIO J. **64**(6), 727–734 (2018)
14. Tesch, K., Kaczorowska, K.: Arterial cannula shape optimization by means of the rotational firefly algorithm. Eng. Optim. **48**(3), 497–518 (2016)
15. Eaton, J.W., et al.: GNU Octave version 4.2.1 manual: a high-level interactive language for numerical computations, https://www.gnu.org/software/octave/doc/v4.2.1/ (2017)
16. Tesch, K.: Continuous optimisation algorithms. GUT Publishers, Gdansk (2016)
17. Kaczorowska, K., Tesch, K.: A short review of blood modelling methods: from macro- to microscales. Task Quarterly **22**(1), 5–16 (2017)
18. Wilcox, D.C.: Turbulence modeling for CFD, DCW Industries, California (1994)

19. Menter, F.R.: Two-equations eddy-viscosity turbulence models for engineering applications. AIAA-J. **32**(8), 1598–1605 (1994)
20. OpenFOAM user guide 2015, OpenFOAM Foundation Ltd
21. Issa, R.I.: Solution of the implicitly discretised fluid flow equations by operator-splitting. J. Comput. Phys. **62**(1), 40–65 (1986)
22. Tesch, K., Kludzinska, K., Doerffer, P.: Investigation of the aerodynamics of an innovative vertical-axis wind turbine. Flow Turbulence and Combustion **95**, 739–754 (2015)
23. Price, K.V., Storn, R., Lampinen, J.: Differential evolution: A practical approach to global optimization. Springer-Verlag, Berlin (2005)
24. Storn, R., Price, K.: Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. **11**, 341–359 (1997)
25. Lacasse, D., Garon, A., Pelletier, D.: Mechanical hemolysis in blood flow: User-independent predictions with the solution of a partial differential equation. Comput. Methods Biomech. Biomed. Engin. **10**(1), 1–12 (2007)
26. Yu, H., Engel, S., Janiga, G., Thévenin, D.: A Review of hemolysis prediction models for computational fluid dynamics. Artif. Organs **41**(7), 603–621 (2017)