INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2010; **62**:1047–1062
Published online 20 April 2009 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.2061

Sensitivity of shear rate in artificial grafts using automatic differentiation

M. Probst[1,*,†], M. Lülfesmann[2], H. M. Bücker[2], M. Behr[1] and C. H. Bischof[2]

[1]*Chair for Computational Analysis of Technical Systems (CATS), Center for Computational Engineering Science (CCES), RWTH Aachen University, 52056 Aachen, Germany*
[2]*Institute for Scientific Computing (SC), Center for Computational Engineering Science (CCES), RWTH Aachen University, 52056 Aachen, Germany*

SUMMARY

An accurate numerical simulation of blood requires the solution of incompressible Navier–Stokes equations coupled with specific constitutive models. We consider a generalized Newtonian fluid model in which viscosity depends on shear rate, accounting for the shear-thinning behavior of blood. Previous work on the design of an artificial graft indicated that there is an influence of the fluid model on the solution of the partial differential equation-constrained shape optimization problem. Therefore, we carry out a sensitivity analysis of the actual implementation of the flow solver using automatic differentiation (AD). We compare the sensitivities of shear rate with respect to viscosity for different configurations and validate the truncation-error-free sensitivities obtained from AD with those based on divided differencing and, if available, with analytic derivatives. Copyright © 2009 John Wiley & Sons, Ltd.

Received 21 September 2008; Revised 23 February 2009; Accepted 26 February 2009

KEY WORDS: sensitivity analysis; automatic differentiation; blood flow; shear-thinning model; shape optimization

1. INTRODUCTION

The World Health Organization (WHO) reports cardiovascular diseases as number one cause of deaths worldwide. In 2002, 12.7 million people died from heart disease or strokes, which accounts for an estimated 23% of all deaths. Occlusion and stenosis of arterial blood vessels inhibit regular blood flow and cause elevated shear stress and recirculation favoring further deposition, and, thus,

*Correspondence to: M. Probst, Chair for Computational Analysis of Technical Systems (CATS), Center for Computational Engineering Science (CCES), RWTH Aachen University, 52056 Aachen, Germany.
†E-mail: probst@cats.rwth-aachen.de

Contract/grant sponsor: German Research Foundation; contract/grant numbers: SPP 1253, GSC 111, EXC 128

have major effects on many of the diseases categorized as heart disease. Blood flow can be restored by either stenting or placing a bypass around the occlusion.

Later stages of heart disease often leave implantation of ventricular assist devices (VADs), which support the heart in maintaining sufficient blood flow, as the only temporary cure until a donor organ becomes available for transplantation. In both artificial grafts and VADs not only blood clotting, but also red blood cell (RBC) damage is a major concern. Exposure of RBCs to high stresses initiates the release of hemoglobin from the cells into the blood stream. This process—called hemolysis—can lead to toxic effects and, ultimately, organ failure when the free hemoglobin exceeds the amount that can be filtered by the kidneys.

Various attempts have been made to find a correlation between flow conditions and hemolysis that allows to assess overall blood damage in medical devices [1–3]. Since blood is a complex fluid showing viscoelastic behavior, considerable effort has been put into studies that investigate how the choice of different constitutive models for blood affects the computed flow field in numerical simulations [4–7]. More recently, Abraham [8] combined both aspects and studied the effects of different fluid models on the shape of artificial bypass grafts when these were optimized with the objective of minimizing hemolysis. The resulting optimal shapes were dependent on the fluid model, as briefly summarized in Section 3.1, which is the motivation behind the work presented here.

The aforementioned results suggest to compute sensitivities with respect to the fluid model and identify in advance scenarios in which complex models for blood are required. Eventually, our goal is to obtain sensitivity of optimal shapes. Computing and implementing these analytically can be tedious since it involves differentiating all software components. Automatic differentiation (AD) provides a reliable and effective alternative to the analytic approach. As a first step toward our goal, we transform the essential component of the objective function—the flow solver—using AD in order to obtain the sensitivity of the flow field and related quantities with respect to viscosity. We state the equations governing the flow and briefly comment on constitutive models for blood in Section 2. Section 3 gives a perspective on simulation and optimization of artery and bypass geometries and the sensitivity study of inherent parameters. This is followed in Section 4 by a short introduction on the solver and a brief description of the AD techniques used to transform the flow solver. In Section 5, the resulting sensitivities are validated against analytic values for a simple test case and sensitivities for the bypass graft are presented. Lastly, we summarize our results.

2. GOVERNING EQUATIONS OF BLOOD FLOW

For the remainder of this paper, we will be dealing with blood flow that is governed by incompressible Navier–Stokes equations

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t}+\mathbf{u}\cdot\nabla\mathbf{u}-\mathbf{f}\right)-\nabla\cdot\boldsymbol{\sigma}=\mathbf{0} \tag{1}$$

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

where $\boldsymbol{\sigma}$ is the stress tensor, ρ is density and $\mathbf{f}$ denotes external forces. The stress tensor is composed of isostatic components acting in normal direction and viscous components acting in the shear plane. For Newtonian fluids, i.e. fluids with constant viscosity, this can be written as $\boldsymbol{\sigma}=-p\mathbf{I}+2\mu\boldsymbol{\varepsilon}(\mathbf{u})$ where the rate of strain tensor $\boldsymbol{\varepsilon}(\mathbf{u})$ is the symmetric part of the velocity gradient

DOI: 10.1002/fld

and **I** is the identity tensor. In many situation it can be crucial how unique properties of blood are built into the model because of their nonnegligible influence on the flow field.

2.1. Constitutive models for blood

Owing to its three main constituents—RBCs, white blood cells and platelets—blood is a highly complex fluid requiring intricate constitutive models for accurate simulations. However, it can be treated as Newtonian fluid in some cases that involve high shear rates [4]. A compromise between the use of this simplification and the treatment of blood as a complex, viscoelastic fluid is the use of generalized Newtonian models where viscosity depends on shear rate, thus accounting for shear-thinning behavior of blood without the need to solve additional equations that model viscoelastic behavior. The shear rate is related to the second invariant of the rate of strain tensor, and, therefore, it can be directly obtained from the flow field as

$$\dot{\gamma} = \sqrt{2\boldsymbol{\varepsilon}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{u})} \tag{3}$$

where the colon operator denotes tensor inner product.

A frequently used generalized Newtonian model is the modified Cross model where the relation of dynamic viscosity μ and shear rate $\dot{\gamma}$ is given by

$$\mu(\dot{\gamma}) = \mu_\infty + \frac{\mu_0 - \mu_\infty}{(1 + (\lambda\dot{\gamma})^b)^a} \tag{4}$$

The symbols μ_0 and μ_∞ denote zero- and infinite-shear limit viscosity that, in combination with parameters λ, a and b, are used to adjust the model to a specific material or fluid. Following [6], a suitable choice to model human blood is $\mu_0 = 1.6 \cdot 10^{-1}\,\mathrm{Pa\,s}$, $\mu_\infty = 3.5 \cdot 10^{-3}\,\mathrm{Pa\,s}$, $a = 1.23$, $b = 0.64$ and $\lambda = 8.2\,\mathrm{s}$.

From here on, we will exclusively use kinematic viscosity that is obtained by simply dividing dynamic viscosity by density. Based on the infinite-shear limit viscosity and blood density $\rho = 1.057\,\mathrm{kg/l}$, the corresponding kinematic viscosity is found to be $v = 3.31 \cdot 10^{-2}\,\mathrm{cm^2 s^{-1}}$. For simplicity, we will keep writing p for kinematic pressure. The influence of the constitutive model in shape optimization will be discussed in Section 3 where a comparison of optimal bypass graft shapes for Newtonian and modified Cross constitutive model is presented.

2.2. Analytic solution for steady flow in 2D

Under certain assumptions, it is possible to obtain an analytic expression for the state variables in (1)–(2), i.e. velocity and pressure. For example, consider steady flow in a 2D rectangular domain where a parabolic profile is prescribed along the inflow boundary and a noslip condition is imposed on the adjacent edges. The continuity equation (2) will be satisfied by any flow profile that depends on one coordinate only and, thus, holds in particular for 2D parabolic flow. Furthermore, time-dependent and advection terms vanish in (1). Assuming that there are no external forces present, the momentum equation reduces to steady Stokes equation for the stress tensor

$$-\nabla \cdot \boldsymbol{\sigma} = \mathbf{0} \tag{5}$$

Note that despite using kinematic viscosity, we still write $\boldsymbol{\sigma}$ and p, respectively, for stress and pressure divided by density. The first component of the 2D parabolic velocity profile has the form $u(y) = u_c\, y\, (\bar{y} - y)$ where $\bar{y}$ denotes the domain extent in y-direction and the maximum velocity is given by $u_{\max} = \frac{1}{4}\bar{y}^2 u_c$.

DOI: 10.1002/fld

When taking the divergence of the rate of strain tensor in steady Stokes equation, almost all second-order derivatives vanish and the 2D system reduces to one equation,

$$\frac{\partial^2 u}{\partial y^2} = \frac{1}{\nu}\frac{\partial p}{\partial x} \tag{6}$$

In pressure-driven flow, the pressure slope will depend on the specified pressure drop $p_\mathrm{d} = p(0) - p(\bar{x}) = -(\partial p/\partial x)\bar{x}$ with $\bar{x}$ denoting the length of the domain in x-direction. The parameter u_c in the parabolic inflow profile depends on the pressure drop and the domain extents and the solution of (6) is

$$\begin{aligned} u &= u(y) = \frac{\bar{y}}{2\bar{x}}\frac{p_\mathrm{d}}{\nu} y(\bar{y} - y) \\ p &= p(x) = C_0 - \frac{p_\mathrm{d}}{\bar{x}} x \end{aligned} \tag{7}$$

with some constant C_0.

3. SHAPE OPTIMIZATION FOR ARTERIAL BYPASS GRAFTS

Computer-aided design has become common practice in many engineering applications. Classic applications include airfoil and automotive design. Numerical simulations of blood flow in arteries and medical devices have been used for many years to improve the understanding of cardiovascular disease and to develop efficient treatment. Because of their relatively simple geometry, bypass grafts have been the focus of many studies, and numerical results have been validated against experimental data collected in clinical practice [9, 10]. A recent review on flow simulation in arterial bypass and arteriovenous grafts is found in [11].

In shape optimization of grafts, several authors pointed out the influence of graft-to-artery angle and diameter as crucial design parameters. However, most of these studies aim to find an optimal configuration in a discrete, limited range of design alternatives [12, 13]. Only recently, approaches to mathematical optimization have been pursued. Optimization strategies include derivative-free methods using a surrogate model of the objective function [14, 15] and gradient-based methods that rely on the solution of adjoint equations [16, 17].

As previously mentioned, the choice of the constitutive model affects the results of numerical simulations. Abraham [8] showed that shape optimization results might depend on the constitutive model as well. This result motivates the sensitivity analysis carried out in this paper. We therefore briefly summarize the observations relevant to our analysis.

3.1. Sensitivity of optimal shapes

Abraham [8] reports on a 2D optimization framework that was used to optimize the shape of artificial bypass grafts that are placed in stenosed arteries to restore blood flow around the occlusion. Figure 1 shows the initial setup. Optimization of the centerline design curve was performed for two different diameters d of this idealized graft—narrow bypass ($d = 0.6$) and wide bypass ($d = 1.0$). Two different inflow conditions were imposed, adjusting the parabolic inflow profile so that associated Reynolds numbers were 50 and 300, respectively.

DOI: 10.1002/fld

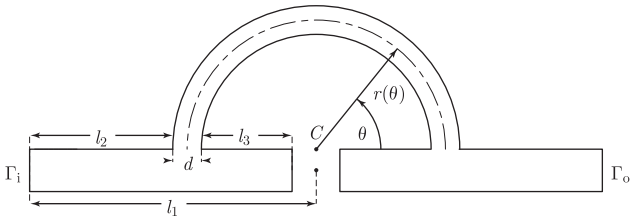

Figure 1. Computational domain for the arterial graft. The initial shape of the design curve of the graft (dashed line) is a semi-circle with center at C. Fixed geometry parameters are $l_1 = 6.0, l_2 = 3.0, l_3 + d = 2.5$. The downstream section of the host artery is symmetric with respect to the upstream one, both have a height of 0.8.

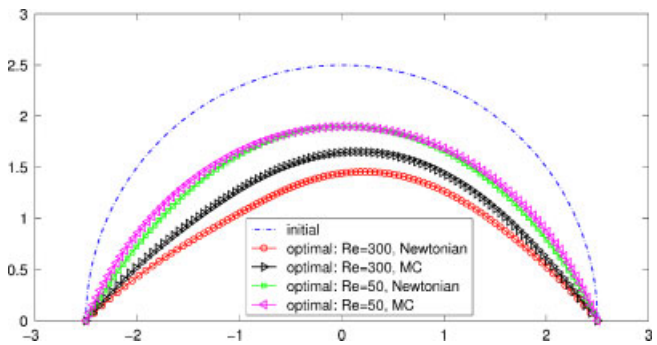

Figure 2. Graft diameter $d = 1.0$. Comparison of initial and optimal center line shapes for different Reynolds numbers, and Newtonian and modified Cross (MC) constitutive models. Optimal design parameters taken from [8].

Shape optimization of the graft aims to minimize hemolysis, which has been found to be correlated to exposure time and shear stress [1]. Abraham takes this into account by choosing the objective functional to be the integral over squared shear rate

$$J(\mathbf{u}, \boldsymbol{\alpha}) = \int_{\boldsymbol{\Omega}(\boldsymbol{\alpha})} \dot{\gamma}^2 \, \mathrm{d}\mathbf{x} \tag{8}$$

In this integral, $\boldsymbol{\Omega}$ represents the graft shape and design variables $\boldsymbol{\alpha}$ are coefficients of a fifth-order polynomial parameterizing the bypass' center line $r(\theta)$.

In the previously described setting, Abraham investigated the influence of the constitutive fluid model on the optimization outcome. Starting from the initial shape, the design variables that minimize the objective function (8) were computed at both Reynolds numbers. First, blood was treated as Newtonian fluid using constant viscosity in the forward solution of Navier–Stokes equations (1)–(2). Then, the optimization was repeated using the modified Cross constitutive model (4). For the wide bypass, resulting optimized center line shapes for these four different cases are compared in Figure 2.

While for the low Reynolds number no significant influence of the constitutive model on the optimal shape was observed, there is a clearly visible impact for the high Reynolds number.

DOI: 10.1002/fld

That is, the optimal shape of the wide graft is sensitive to viscosity variation at $Re=300$. In contrast, for $Re=50$ and for the narrow graft (results are not shown here) at both flow conditions, sensitivity was marginal. A sensitivity analysis could possibly confirm this result and, even more importantly, could have predicted the elevated sensitivity for the wide bypass at high Reynolds number beforehand.

4. COMPUTATIONAL FRAMEWORK

4.1. The flow solver XNS

As a first step toward derivatives of optimal shapes with respect to parameters that enter the simulation process, we focus on the flow solver that provides the forward solution in the optimization. The solver used here is an in-house research code called XNS.

XNS is a highly-parallel finite element solver that is portable and has been tested on a wide range of systems with different architectures. The scalability of the code has been shown to be satisfactory on up to 4096 processors on a Blue Gene/L system using message passing interface (MPI) parallelization [18]. MPI calls are wrapped in an in-house communication library that supports protocols other than MPI as well. XNS uses a stabilized space–time Galerkin/least-squares discretization that allows the use of equal-order linear interpolation. A Krylov subspace method is employed to solve the resulting system of linear equations. The implementation includes various types of equations prevalent in computational fluid dynamics practice: advection–diffusion processes, shallow-water equations, and compressible and incompressible Navier–Stokes equations. Areas of application include diffusion of chemicals, flow around submarine and ship propulsion systems, air flow around a helicopter, flows in spillways and channels, and tidal flows in the Tokyo bay. Furthermore, XNS is able to model viscoelastic fluids such as blood and features special mesh update techniques to accommodate deforming and rotating parts of the computational domain.

4.2. Code transformation by AD

For a given viscosity v, the forward simulation carried out by XNS solves the Navier–Stokes equations (1)–(2) and computes velocity and pressure values for each node in the computational domain. The shear rate $\dot{\gamma}$ is constant on each element and is directly related to the velocity field through Equation (3).

From an abstract point of view, the execution of XNS in this particular scenario represents the evaluation of some mathematical function $\dot{\gamma}=f(v)$. The term AD [19, 20] refers to a set of techniques to mechanically generate, from a computer program computing f, a new computer program for the computation of the derivative f'. Thus, AD can be used to transform the forward code XNS into a new code capable of computing derivatives $\partial\dot{\gamma}/\partial v$. The idea behind this technology is to systematically apply the chain rule of differential calculus to every elementary operation. More precisely, suppose that the code to be transformed contains a statement of the form

$$z=\varphi(x, y)$$

DOI: 10.1002/fld

implementing a binary elementary operation, φ, available in a programming language, for instance, multiplication of two scalar real values, x and y. Then, the so-called forward mode of AD, when used to compute derivatives with respect to viscosity v, generates a new statement implementing

$$\frac{\mathrm{d}z}{\mathrm{d}v}=\frac{\partial\varphi(x,y)}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}v}+\frac{\partial\varphi(x,y)}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}v}$$

The new statement is immediately preceding the original statement with which it is associated. In practice, each statement is decomposed into a sequence of elementary operations to which the chain rule is applied repeatedly. As a simple example, consider φ to stand for multiplication of two scalar real values and consider the statement

```
z = x*y*z
```

involving two multiplication operations φ. Then, the AD forward mode transforms the given code fragment into the new code fragment

```
g_z = y*z*g_x + x*z*g_y + x*y*g_z
z = x*y*z
```

where `g_z` is a new variable used to store $\mathrm{d}z/\mathrm{d}v$ and the new variables `g_x` and `g_y` are interpreted analogously. Transforming every statement of the original code in this way and starting with the known value $\mathrm{d}v/\mathrm{d}v=1$, the sensitivity with respect to v is carried forward along with the computation implemented in the original code. Thus, the new code to compute $\partial\dot{\gamma}/\partial v$ is generated mechanically by using the chain rule and the known partial derivatives of a finite set of elementary operations. Sophisticated AD transformations take care of data dependencies to keep track of which variable influences another variable. The process of finding out whether a variable is influenced by viscosity and has an influence on shear rate is called data dependence analysis and is useful to increase the performance of the AD-generated code.

There are different AD techniques going beyond the forward mode due to the associativity of the chain rule of differential calculus. In exact arithmetic, these techniques compute the same result, but may differ dramatically in terms of storage requirement and computing time. The successful use of these technologies in a wide range of application areas is reported, for example, in the proceedings of the international AD workshops [21–25]. Software tools implementing AD for various languages are listed at the community portal `www.autodiff.org`.

Rather than applying AD in a black-box fashion to XNS, we take advantage of its code structure in a so-called hierarchical way [26–29]. Hierarchical AD approaches are used not only to improve the performance of the AD-generated code but also to more precisely control its behavior. Here, the code for the solution of linear and nonlinear systems is transformed in a hierarchical way. In XNS, the solution of a system of linear equations

$$A\mathbf{u}=\mathbf{b} \tag{9}$$

with large sparse nonsingular nonsymmetric coefficient matrix A and right-hand side $\mathbf{b}$ is carried out by an iterative Krylov solver [30]. In a hierarchical AD approach, the code implementing the Krylov solver is not transformed explicitly as in a black-box AD approach. The hierarchical idea is based on differentiating (9) with respect to viscosity v leading to

$$A\frac{\partial\mathbf{u}}{\partial v}=\frac{\partial\mathbf{b}}{\partial v}-\frac{\partial A}{\partial v}\mathbf{u}$$

DOI: 10.1002/fld

Thus, compared with (9), the sensitivity $\partial \mathbf{u}/\partial v$ is obtained by solving a linear system with the same coefficient matrix A and a new right-hand side whose different terms are available in the AD forward mode. That is, differentiation is applied on a higher level of abstraction by gluing together existing pieces of code into a template for the differentiation of a linear solver; see [28, 29, 31] for more details on differentiation of linear solvers in a hierarchical way.

During the numerical solution of the Navier–Stokes equations, the resulting nonlinear system is solved iteratively. The code generated in a black-box fashion by AD then involves an iteration for the derivatives with respect to viscosity. Without manual intervention, the AD-generated code will stop this iteration as soon as the stopping criterion of the nonlinear iteration used in original code is fulfilled. However, the convergence of the iterative scheme in the original code does not generally imply the convergence of the corresponding scheme in the derivative code at the same iteration number. Rather, the rate of convergence of the derivative iteration tends to be lower than that of the original iteration. Therefore, there is need to manually introduce a separate stopping criterion for the derivative iteration. In our implementation, we adopt the stopping criterion for the original iteration that stops if the differences to the previous iteration become smaller than a threshold ε. That is, the derivative iteration is stopped when the differences in the derivatives are less than the same threshold ε.

From a practical point of view, the complexity of the AD transformation process depends on the programming language. A programming language that offers a high level of sophisticated language features tends to be more challenging for an AD tool than a language involving only a small set of simple language constructs. In addition, the AD transformation of programs involving constructs from different languages is not easily automated [32]. XNS is written primarily in Fortran. However, there are some programming constructs going beyond pure Fortran. In particular, some of the dynamical memory allocation is implemented by Cray pointers that connect a variable with storage that is allocated by a suitable function written in C. A typical use of Cray pointers is given by the following code fragment:

```
      real x(dim)
      pointer (xptr,x)
      xptr = ewdmalloc(dim*fsize)
C     The array x(1:dim) is accessible here.
      ...
      call ewdfree(xptr)
```

Here, the dimension `dim` of the array `x` is not known at compile time, but read in from a configuration file at run time. The second line declares the variable `xptr` to be a pointer to the array `x`. The C code `ewdmalloc` allocates storage for an array of a certain size and returns the address of its first element. Deallocation of that storage is carried out by the C code `ewdfree`.

The code fragment generated by the AD forward mode when differentiating with respect to a single scalar parameter like viscosity is then given by

```
      real x(dim), g_x(dim)
      pointer (xptr,x)
      pointer (g_xptr,g_x)
      xptr = ewdmalloc(dim*fsize)
      g_xptr = ewdmalloc(dim*fsize)
C     The arrays x(1:dim) and g_x(1:dim) are accessible here.
```


DOI: 10.1002/fld

```
...
call ewdfree(xptr)
call ewdfree(g_xptr)
```

That is, the derivative variable `g_xptr` is treated in the same way as the corresponding original variable `xptr`. In addition, all pointer assignments that are used, for instance when copying arrays that were generated this way, have to be transformed in such a way that the data dependence analysis is not violated.

Cray pointers are transformed by a script before passing the resulting Fortran code to the AD tool Adifor [33] that generates the code to compute $\partial\dot{\gamma}/\partial v$.

5. NUMERICAL RESULTS

5.1. Validation of AD-generated derivatives

In order to verify the correctness of derivatives generated with the AD transformation of XNS, we set up a simple test case for which analytic derivatives are known. The test case was also taken as a reference to determine a suitable mesh resolution for the sensitivity analysis of an artificial graft in Section 5.2.

We consider purely pressure-driven flow in a 3×2 rectangle where no-slip boundary conditions are imposed on the upper and lower wall. Furthermore, the velocity in the vertical direction is set to zero, at both inflow and outflow boundary.

Prescribing a fixed pressure head at the outflow results in a steady, parabolic flow profile. Without loss of generality, we can choose to prescribe the full pressure head at the outflow boundary as shown along with the other boundary conditions in Figure 3. Analytic expressions for velocity and pressure were previously derived in Section 2.2.

We would like the analytic test case to resemble flow conditions in the idealized graft geometry used in the work of Abraham [8], which was briefly presented in Section 3.1. Simulations are carried out using a Newtonian fluid model with kinematic viscosity set to $v=0.0331$ so as to emulate fluid properties of blood that were discussed in Section 2.1. The Reynolds number is determined

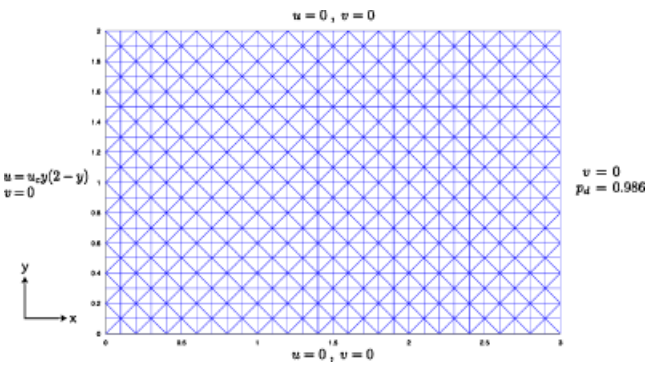

Figure 3. Boundary conditions shown for mesh with 1200 elements.

DOI: 10.1002/fld

by inflow height, maximum inflow velocity and viscosity. Inserting the analytic expression for velocity (for $\bar{y}=2$, we have $u_{\max}=u_c$) yields

$$Re=\frac{1}{3}\frac{p_{\mathrm{d}}}{v^2} \tag{10}$$

A flow with Reynolds number 300 will therefore necessitate a pressure head of

$$p_{\mathrm{d}}=900\,v^2=0.986 \tag{11}$$

We use this setting for validating the code obtained from XNS through AD as described in Section 4.2. The AD transformation of the original code will be referred to as XNS.AD from now on. The shear rate derivative with respect to viscosity is computed both with AD and divided differences (DD) and compared with analytic values.

From the analytic, parabolic velocity profile (7), we have $u_c=\frac{1}{3}(p_{\mathrm{d}}/v)$ and we obtain the analytic shear rate derivative with respect to viscosity

$$\begin{aligned}\frac{\partial\dot{\gamma}}{\partial v}&=\frac{\partial}{\partial v}\left|\frac{\partial u}{\partial y}\right|\\&=\frac{\partial}{\partial v}|2u_c(1-y)|\\&=\begin{cases}\dfrac{1}{v^2}\dfrac{1}{3}p_{\mathrm{d}}(1-y), & y\geqslant 1\\[2ex] -\dfrac{1}{v^2}\dfrac{1}{3}p_{\mathrm{d}}(1-y), & y<1\end{cases}\\&=-Re|(1-y)|\end{aligned} \tag{12}$$

Note that XNS.AD computes both shear rate and its derivative on a discrete level at the center of each element. For this reason, the continuous expression (12) is used to provide analytic values on an element level basis as well.

The mesh refinement study aims to determine a mesh resolution at which XNS.AD is capable of computing shear rate derivatives with acceptable precision. The initial resolution features 10 rectangular elements per length unit where each square is divided into two triangles resulting in a mesh size of 1200 elements. In each refinement step the number of elements per length unit is doubled. For each step, the relative difference between analytic and AD shear rate derivative is computed on each element as is exemplarily shown for a 19 200 element mesh in Figure 4. The values of the shear rate derivative are close to zero along the center line in x-direction and cause an exceptionally high relative error for elements adjacent to the center line compared with the rest of the domain.

DOI: 10.1002/fld

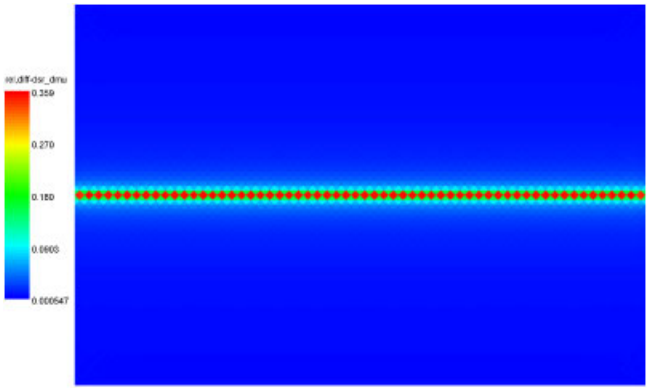

Figure 4. Mesh with 19 200 elements. Relative difference between analytic and AD shear rate derivative.

Table I. Mesh refinement study. Average absolute (columns 2–4) and relative (columns 5–7) errors between analytic, AD and DD shear rate derivatives. Relative errors in %.

Mesh size	$\bar{E}^{\mathrm{AD/ana}}$	$\bar{E}^{\mathrm{DD/ana}}$	$\bar{E}^{\mathrm{DD/AD}}$	$\bar{E}^{\mathrm{AD/ana}}_{\mathrm{rel}}$	$\bar{E}^{\mathrm{DD/ana}}_{\mathrm{rel}}$	$\bar{E}^{\mathrm{AD/DD}}_{\mathrm{rel}}$
$30\times20\times2=1200$	3.4202	3.4202	$1.5008\cdot10^{-5}$	5.1164	5.1164	$2.1869\cdot10^{-7}$
$60\times40\times2=4800$	1.6855	1.6855	$8.5012\cdot10^{-5}$	2.9362	2.9363	$1.2635\cdot10^{-6}$
$120\times80\times2=19200$	0.8377	0.8377	$1.6592\cdot10^{-4}$	1.6596	1.6597	$3.3960\cdot10^{-6}$

As a measure for accuracy of the derivative over the whole domain, we take the average of the absolute error and the relative error over all elements $i=1,\ldots,n_e$. The errors between AD and analytic values, for example, are defined as

$$\begin{aligned}\bar{E}^{\mathrm{AD/ana}} &:= \frac{1}{n_e}\sum_i |(\partial\dot{\gamma}/\partial v)_i^{\mathrm{AD}} - (\partial\dot{\gamma}/\partial v)_i^{\mathrm{ana}}| \\ \bar{E}^{\mathrm{AD/ana}}_{\mathrm{rel}} &:= \frac{1}{n_e}\sum_i \frac{|(\partial\dot{\gamma}/\partial v)_i^{\mathrm{AD}} - (\partial\dot{\gamma}/\partial v)_i^{\mathrm{ana}}|}{|(\partial\dot{\gamma}/\partial v)_i^{\mathrm{ana}}|}\end{aligned} \tag{13}$$

and similarly for derivatives computed with DD with a step size of 10^{-8}. Table I shows that the average error in the computation of the AD shear rate derivative is roughly reduced by a factor of two with each refinement step and AD derivatives approach analytic values as the mesh size is increased. DD provide additional evidence of the correctness of XNS.AD code. All computations were carried out on a single CPU of a Intel Harpertown E5450 Xeon node with a total of eight cores at 3000 MHz and 16 GB memory.

For additional validation we compare analytic, AD and DD values of the maximum absolute shear rate derivative for three different mesh resolutions. The results are presented in Table II. Owing to evaluation at element centers, even analytic values will not be as high as the maximum absolute nodal value of 300 that results from (12). Nevertheless, AD values approach the analytic derivative values with increasing mesh size in the same way as it could be observed for the average errors in Table I.

DOI: 10.1002/fld

Table II. Mesh refinement study. Comparison of maximum absolute shear rate derivative for analytic, AD and DD derivatives.

	$\max \partial\dot{\gamma}/\partial v$		
Mesh size	Analytic	AD	DD
$30\times 20\times 2=1200$	289.986	286.788	286.788
$60\times 40\times 2=4800$	294.985	293.372	293.372
$120\times 80\times 2=19\,200$	297.485	296.676	296.678

Concluding the analysis, we deduce that a mesh with 4800 elements and a relative error below 3% constitutes a good compromise between precision and computation time. Even though the latter is not an issue for the simple rectangular test case, a resolution of 40 elements for the parabolic inflow profile will result in computationally expensive meshes for the artificial graft sensitivity analysis, which makes it unfavorable to use an even higher resolution to gain precision.

5.2. Sensitivity analysis of shear rate in artificial grafts

The dependence of optimization results on the constitutive fluid model indicates the value of an investigation of the sensitivity of optimal shapes not only by carrying out simulations, but by computing derivatives of the optimal solution with respect to viscosity and graft diameter, for example. A sensitivity analysis choosing the first of the two ideally should predict the elevated sensitivity for the wide bypass at high Reynolds number.

However, computing derivatives of optimal shape, e.g. with respect to viscosity, requires differentiation through the entire optimization framework including Navier–Stokes equations and their adjoint, mesh update routines, and—depending on the choice of AD technology—the optimizer. While obtaining and combining this sequence of derivatives analytically is an extremely arduous and difficult task, AD provides a convenient and accurate way to solve this problem. The first step in that direction is to compute the AD-based derivative of the forward solution, i.e. applying AD to XNS as described in Section 4.2. Even though one cannot obtain sensitivities of optimal shapes this way, one can analyze the sensitivity of quantities derived from the forward solution that have significant influence on the optimization process.

The quantity of our interest for the remainder of this section will be shear rate and its derivative with respect to viscosity, because it directly affects the optimization outcome through the objective function (8). The simulations with XNS.AD were run using the Newtonian fluid model for blood, which means that the shear rate derivative will represent sensitivity at the prescribed viscosity $v=0.0331$. Here, the limitations of taking derivatives with respect to viscosity to predict the dependence of the solution (or the optimal shapes) on the constitutive model is clearly seen. Using modified Cross model in the sensitivity analysis would add no benefit because computing derivatives means considering a constant variation over the whole domain, while viscosity changes can be different for each node when switching from the Newtonian to the generalized Newtonian model.

Once the derivative of shear rate has been computed, one can easily obtain the derivative of squared shear rate as well by applying chain rule as part of the postprocessing procedure. This immediately provides the sensitivity of the objective function with respect to viscosity.

$$\frac{\partial(\dot{\gamma}^2)}{\partial v}=2\dot{\gamma}\frac{\partial\dot{\gamma}}{\partial v} \tag{14}$$

DOI: 10.1002/fld

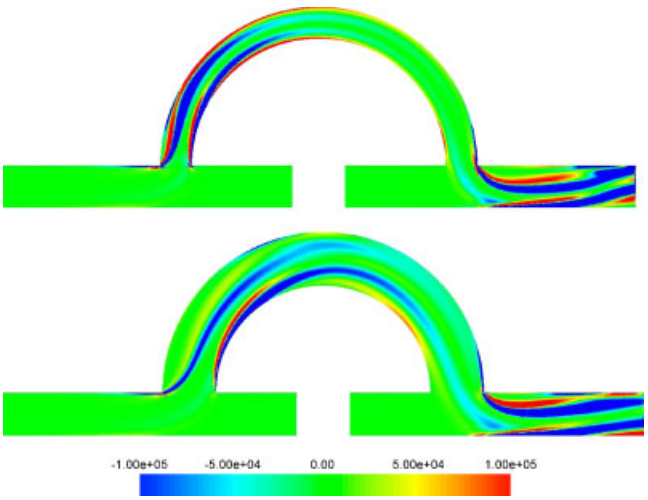

Figure 5. AD derivative of squared shear rate with respect to viscosity for $Re=300$. For a better perception of the spatial distribution, the color lookup table is scaled down. The actual data range of $\partial\dot{\gamma}/\partial v$ is $[-1.336\cdot10^7, 5.847\cdot10^5]$ for narrow and $[-1.298\cdot10^7, 3.349\cdot10^5]$ for wide graft, displayed at top and bottom, respectively.

Since a parabolic velocity profile is prescribed at the inflow boundary of both the rectangle and the graft, the conclusions drawn in the previous section can be used to build a mesh for the artificial graft with a suitable resolution. The rectangular mesh with 4800 elements had 40 elements at the inflow boundary to resolve the parabolic profile. For the graft, setting the number of elements at the inflow to 40 and keeping the same distance between nodes at the other boundaries results in a mesh with 119 402 elements for the narrow graft and 143 404 elements for the wide graft.

The shear rate derivative with respect to (kinematic) viscosity is computed for Reynolds numbers 50 and 300. Using (14), the derivative of squared shear rate is computed as a measure for the sensitivity of the objective function. The results are shown in Figures 5 and 6. By scaling down the color lookup table, the regions of high sensitivity are highlighted without losing too much information, because $\partial\dot{\gamma}/\partial v$ is in the lookup table's range anyway for most of the elements. For $Re=300$ this is true for about 92% of the elements of the wide and 87% of the narrow graft, respectively. These percentages are even higher for $Re=50$ with 92% and 98%, respectively.

The alternating patterns of increasing and decreasing derivative values through a section of the graft in flow direction are produced by the changes in the flow field. At higher viscosity, the flow inertia effects are reduced and the velocity profile in the graft is shifted toward the upper bypass boundary. This shift causes the alternating pattern that is more distinct for higher Reynolds number (Figure 5) than for lower (Figure 6) due to a higher potential for change at higher velocities. However, comparing narrow and wide bypass diameter reveals no significant characteristics that would distinguish the wide graft at high Reynolds number from the other cases as might have been expected from Abraham's observations.

As a scalar measure for sensitivity, the integral over both the derivative of squared shear rate and its absolute value are computed, see Table III. The former equals the derivative of the objective

DOI: 10.1002/fld

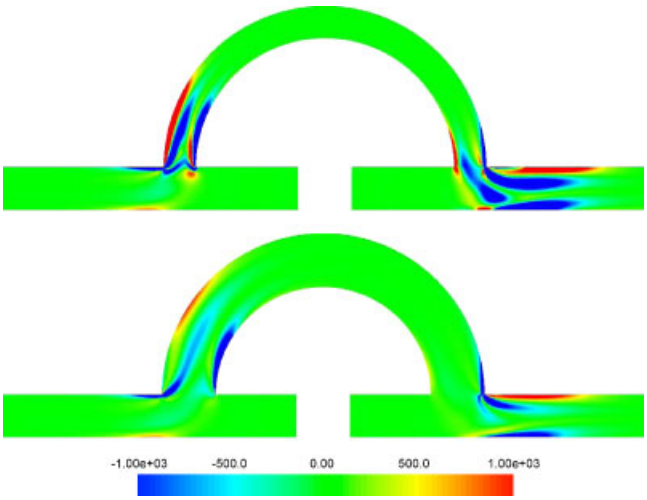

Figure 6. AD derivative of squared shear rate with respect to viscosity for $Re=50$. For a better perception of the spatial distribution, the color lookup table is scaled down. The actual data range of $\partial\dot{\gamma}/\partial v$ is $[-1.092\cdot10^5, 4.668\cdot10^3]$ for narrow and $[-0.853\cdot10^5, 2.636\cdot10^3]$ for wide graft, displayed at top and bottom, respectively.

Table III. Integrals as a measure for sensitivity.

	$Re=300$		$Re=50$	
	Narrow	Wide	Narrow	Wide
$\int(\partial\dot{\gamma}^2/\partial v)$	−510 842	−479 169	−2142	−1206
$\int\lvert(\partial\dot{\gamma}^2/\partial v)\rvert$	818 657	701 278	4314	2489

function (8) with respect to viscosity, whereas in the latter cancelation of values with opposing sign is avoided and, thus, an overall sensitivity is retained.

The scalar sensitivity measures in Table III do not identify the wide bypass geometry at high inflow velocity as a special case either. The sensitivity for high Reynolds number is significantly higher, and the sensitivity ratio of narrow and wide bypass goes down from about 1.7 for $Re=50$ to about 1.2 for $Re=300$.

In conclusion, computing derivatives of the flow solution (and related quantities like shear rate) with respect to viscosity using XNS.AD could not reveal the sensitivity of the optimal shape to the fluid model as reported in Abraham's work. Taking the derivatives at a fixed viscosity corresponds to varying viscosity in the same way for the whole domain while in generalized Newtonian fluid models, like the modified Cross model, viscosity might be different for each element. This could be one of the reasons why computing an overall sensitivity for the forward solution of Navier–Stokes equations is not able to capture the effects of the constitutive model on the optimized graft shape. The results of our analysis suggest that one actually needs to compute derivatives of the entire optimization chain.

DOI: 10.1002/fld

6. SUMMARY AND OUTLOOK

The contributions of this paper are as follows. AD was successfully applied to a complex flow solver resulting in a transformed code that can be used to compute derivatives with respect to viscosity. AD-generated shear rate derivatives were validated against analytic values for pressure-driven flow in a 2D rectangle. This test case was also used to find a suitable mesh resolution for parabolic inflow profiles. For artificial bypass grafts, Abraham [8] carried out a shape optimization study that showed an influence of the constitutive model for blood on the optimization outcome in case of a wide bypass at Reynolds number 300. Using the same setting, we investigated whether the shear rate derivative with respect to viscosity can expose this sensitivity of the optimized shape beforehand. This turned out to be not the case suggesting that a sensitivity analysis of the forward solution alone is not sufficient to predict sensitivity of the whole optimization process.

This observation clearly pinpoints the next steps that should be taken. AD should be applied to the entire optimization framework, which would enable to actually compute a sensitivity of the optimal shape. Another issue that will be addressed in future work is to overcome limitations given by the parametrization of the bypass center line that restricts the set of admissible shapes in the optimization. The sensitivity with respect to the constitutive blood model might be affected by these limitations.

ACKNOWLEDGEMENTS

The Aachen Institute for Advanced Study in Computational Engineering Science (AICES) provided a stimulating research environment for our work. This work was supported by the German Research Foundation (DFG) under GSC 111, EXC 128 and especially SPP 1253 programs. Computing resources were provided by the RWTH Aachen University Center for Computing and Communication and by the Forschungszentrum Jülich.

REFERENCES

1. Giersiepen M, Wurzinger LJ, Opitz R, Reul H. Estimation of shear stress-related blood damage in heart valve prostheses—in vitro comparison of 25 aortic valves. *The International Journal of Artificial Organs* 1990; **13**(5):300–306.
2. Bludszuweit C. Model for general mechanical blood damage prediction. *Artificial Organs* 1995; **19**(7):583–589.
3. Arvand A, Hormes M, Reul H. A validated computational fluid dynamics model to estimate hemolysis in a rotary blood pump. *Artificial Organs* 2005; **29**(7):531–540.
4. Johnston BM, Johnston PR, Corney S, Kilpatrick D. Non-Newtonian blood flow in human right coronary arteries: steady state simulations. *Journal of Biomechanics* 2004; **37**:709–720.
5. Johnston BM, Johnston PR, Corney S, Kilpatrick D. Non-Newtonian blood flow in human right coronary arteries: transient simulations. *Journal of Biomechanics* 2006; **39**:1116–1128.
6. Leuprecht A, Perktold K. Computer simulation of non-Newtonian effects on blood flow in large arteries. *Computer Methods in Biomechanics and Biomedical Engineering* 2001; **4**:149–163.
7. Siauw WL, Ng EYK, Mazumdar J. Unsteady stenosis flow prediction: a comparative study of non-Newtonian models with operator splitting scheme. *Medical Engineering and Physics* 2000; **22**:265–277.
8. Abraham F, Behr M, Heinkenschloss M. Shape optimization in steady blood flow: a numerical study of non-Newtonian effects. *Computer Methods in Biomechanics and Biomedical Engineering* 2005; **8**(2):127–137.
9. Ku JP, Draney MT, Arko FR, Lee WA, Chan FP, Pelc NJ, Zarins CK, Taylor CA. In vivo validation of numerical prediction of blood flow in arterial bypass grafts. *Annals of Biomedical Engineering* 2002; **30**:743–752.
10. Leuprecht A, Perktold K, Prosi M, Berk T, Trubel W, Schima H. Numerical study of hemodynamics and wall mechanics in distal end-to-side anastomoses of bypass grafts. *Journal of Biomechanics* 2002; **35**:225–236.

DOI: 10.1002/fld

11. Loth F, Fischer PF, Bassiouny HS. Blood flow in end-to-side anastomoses. *Annual Review of Fluid Mechanics* 2008; **40**:367–393.
12. Xiong FL, Chong CK. A parametric numerical investigation on hemodynamics in distal coronary anastomoses. *Medical Engineering and Physics* 2008; **30**(3):311–320.
13. Fei D, Thomas JD, Rittgers SE. The effect of angle and flow rate upon hemodynamics in distal vascular graft anastomoses: a numerical model study. *Journal of Biomedical Engineering* (ASME) 1994; **116**:331–336.
14. Marsden AL, Feinstein JA, Taylor CA. A computational framework for derivative-free optimization of cardiovascular geometries. *Computer Methods in Applied Mechanics and Engineering* 2008; **197**(21–24): 1890–1905.
15. Kolachalama VB, Bressloff NW, Nair PB, Shearman CP. Predictive haemodynamics in a one-dimensional human carotid artery bifurcation, part II: application to graft design. *IEEE Transactions on Biomedical Engineering* 2008; **3**:1176–1184.
16. Rozza G. On optimization, control and shape design of an arterial bypass. *International Journal for Numerical Methods in Fluids* 2005; **47**(10–11):1411–1419.
17. Agoshkov V, Quarteroni A, Rozza G. Shape design in aorto-coronaric bypass anastomoses using perturbation theory. *SIAM Journal on Numerical Analysis* 2006; **44**(1):367–384.
18. Wylie B, Geimer M, Nicolai M, Probst M. Performance analysis and tuning of the XNS CFD solver on BlueGene/L. In *Proceedings of the 14th EuroPVM/MPI Conference*, Capello F, Herault T, Dongarra J (eds). Lecture Notes in Computer Science, vol. 4757. Springer: Berlin, 2007; 107–116.
19. Griewank A, Walther A. *Evaluating Derivatives*: *Principles and Techniques of Algorithmic Differentiation* (2nd edn). Other Titles in Applied Mathematics, vol. 105. SIAM: Philadelphia, PA, 2008.
20. Rall LB. *Automatic Differentiation*: *Techniques and Applications*. Lecture Notes in Computer Science, vol. 120. Springer: Berlin, 1981.
21. Griewank A, Corliss G. *Automatic Differentiation of Algorithms*. SIAM: Philadelphia, PA, 1991.
22. Berz M, Bischof C, Corliss G, Griewank A (eds). *Computational Differentiation*: *Techniques*, *Applications*, *and Tools*. SIAM: Philadelphia, PA, 1996.
23. Corliss G, Faure C, Griewank A, Hascoët L, Naumann U (eds). *Automatic Differentiation of Algorithms*: *From Simulation to Optimization*. Springer: New York, 2002.
24. Bücker HM, Corliss GF, Hovland PD, Naumann U, Norris B (eds). *Automatic Differentiation*: *Applications*, *Theory*, *and Implementations*. Lecture Notes in Computational Science and Engineering, vol. 50. Springer: New York, 2005.
25. Bischof CH, Bücker HM, Hovland PD, Naumann U, Utke J (eds). *Advances in Automatic Differentiation*. Lecture Notes in Computational Science and Engineering, vol. 64. Springer: New York, 2008.
26. Bücker HM. Hierarchical algorithms for automatic differentiation. Habilitationsschrift, Faculty of Mathematics, Computer Science, and Natural Sciences, RWTH Aachen University, Aachen, 2002.
27. Bischof CH, Haghighat MR. Hierarchical approaches to automatic differentiation. In *Computational Differentiation*: *Techniques*, *Applications*, *and Tools*, Berz M, Bischof C, Corliss G, Griewank A (eds). SIAM: Philadelphia, PA, 1996; 83–94.
28. Hovland P, Norris B, Roh L, Smith B. Developing a derivative-enhanced object-oriented toolkit for scientific computations. In *Object Oriented Methods for Interoperable Scientific and Engineering Computing*: *Proceedings of the 1998 SIAM Workshop*, Henderson ME, Anderson CR, Lyons SL (eds). SIAM: Philadelphia, 1999; 129–137.
29. Bischof CH, Bücker HM, Hovland PD. On combining computational differentiation and toolkits for parallel scientific computing. In *Euro-Par 2000—Parallel Processing*, *Proceedings of the Sixth International Euro-Par Conference*, Munich, Germany, August/September 2000, Bode A, Ludwig T, Karl W, Wismüller R (eds). Lecture Notes in Computer Science, vol. 1900. Springer: Berlin, 2000; 86–94.
30. Saad Y. *Iterative Methods for Sparse Linear Systems* (2nd edn). SIAM: Philadelphia, 2003.
31. Tadjouddine M, Forth SA, Keane AJ. Adjoint differentiation of a structural dynamics solver. In *Automatic Differentiation*: *Applications*, *Theory*, *and Implementations*, Bücker HM, Corliss GF, Hovland PD, Naumann U, Norris B (eds). Lecture Notes in Computational Science and Engineering, vol. 50. Springer: New York, 2005; 309–319.
32. Bücker HM, Elsheikh A, Vehreschild A. A system for interfacing MATLAB with external software geared toward automatic differentiation. In *Mathematical Software—ICMS 2006*, *Proceedings of the Second International Congress on Mathematical Software*, Castro Urdiales, Spain, 1–3 September 2006, Iglesias A, Takayama N (eds). Lecture Notes in Computer Science, vol. 4151. Springer: Berlin, 2006; 373–384.
33. Bischof C, Carle A, Khademi P, Mauer A. ADIFOR 2.0: automatic differentiation of Fortran 77 programs. *IEEE Computational Science and Engineering* 1996; **3**(3):18–32.

DOI: 10.1002/fld