An International Journal of Optimization
and Control: Theories & Applications
Vol.3, No.2, pp.99-110 (2013) © IJOCTA
ISSN:2146-0957 eISSN:2146-5703
DOI:10.11121/ijocta.01.2013.00144
http://www.ijocta.com

Simultaneous model spin-up and parameter identification with the one-shot method in a climate model example

Claudia Kratzenstein[a] and Thomas Slawig[b]

[a] Institute for Computer Science, Christian-Albrechts-Universität zu Kiel, 24098 Kiel, Germany
Email: ctu@informatik.uni-kiel.de

[b] Institute for Computer Science and Kiel Marine Science Centre for Interdisciplinary Marine Science, Cluster The Future Ocean, Christian-Albrechts Universität zu Kiel, 24098 Kiel, Germany
Email: ts@informatik.uni-kiel.de

(Received October 09, 2012; in final form May 29, 2013)

Abstract. We investigate the One-shot Optimization strategy introduced in this form by Hamdi and Griewank for the applicability and efficiency to identify parameters in models of the earth's climate system. Parameters of a box model of the North Atlantic Thermohaline Circulation are optimized with respect to the fit of model output to data given by another model of intermediate complexity. Since the model is run into a steady state by a pseudo time-stepping, efficient techniques are necessary to avoid extensive recomputations or storing when using gradient-based local optimization algorithms. The One-shot approach simultaneously updates state, adjoint and parameter values. For the required partial derivatives, the algorithmic/automatic differentiation tool TAF was used. Numerical results are compared to results obtained by the BFGS and L-BFGS quasi-Newton method.

Keywords: Algorithmic differentiation; bounded retardation; climate model; fixed point iteration; parameter identification.

AMS Classification: 49M29; 90C30; 90C53; 92-08

1. Introduction

Parameter optimization is an important task in all kinds of climate models or models that simulate parts of the climate system, as for example ocean or atmospheric models. Still, some processes are not well-known, some are too small-scaled in time or space, and others are just beyond the scope of the model. All these processes are *parameterized*, i.e. simplified model functions (parameterizations) are used. These necessarily include lots of – most of the time – only heuristically known parameters. A main task thus is to *calibrate* the models by optimizing the parameter w.r.t. data from measurements or other (more complex) models.

Similar to many applications in engineering applications of fluid mechanics, also in geophysical flows (e.g. ocean models) an optimization is at first performed for steady states of the equations before proceeding to transient problems. This means that only the stationary solution is used in the cost or objective function to be minimized. Moreover (and this is the second point where engineering and geophysical flow problems are similar), the computation of steady states is often performed by running a transient model into the steady state. This strategy is called pseudo time-stepping, since the time variable may be regarded as a kind of iteration counter.

Corresponding Author. Email: ctu@informatik.uni-kiel.de.

It is well known from optimal control of differential equations that the classical adjoint technique (that allows the representation of the gradient of the cost) leads to a huge amount of recomputations, storing or both. This problem looks even more frustrating in the pseudo-time stepping context, since here only the final, numerically converged state is important for the cost. Nevertheless a classical adjoint technique would need all intermediate iterates.

If the number of parameters to be optimized is small, a sensitivity equation approach is also reasonable. On the discrete level this is comparable to the application of the forward mode of Automatic or Algorithmic Differentiation (AD). Here, the sensitivity equation has the same temporal integration direction (namely forward) as the original pseudo time-stepping. But nevertheless it is worthwhile investigating how the two (for a non-linear model) coupled iterations for state and sensitivity are performed.

Griewank described in [1] the differences between two-phase (where the iteration for the state is run to the steady state or fixed point first, and then the sensitivity is computed) and piggy-back approaches (where both iterations are combined to one). Christianson in [2] proposed to perform the sensitivity iteration with the converged state instead of using its iterates. Giering, Kaminski and Vossbeck in [3] used the so-called *Full Jacobian approach*, where they directly used the steady state equation and differentiated it to obtain an equation for the gradient.

The approach used here is called One-shot approach, which was in this form developed by Hamdi and Griewank, and can be seen as an extension of the piggy-back strategy aiming for optimality and feasibility simultaneously with the so-called bounded retardation. That means that the number of One-shot iterations shall not too much exceed the number of fixed point iteration steps that are necessary for the computation of feasible states itself. Theoretical results were published in [4],[5], an engineering application was presented by Özkaya and Gauger in [6].

The idea of simultaneous solution of state equations and parameter correction is not new. In [7], S. Ta'asn uses a pseudo-time embedding for the state and adjoint state equations and the design equation is solved as an additional boundary condition. This still results in a differential algebraic equation which requires some strategy to solve the design equation alone.

In [8], the authors construct a system of only ODEs which is solved by a time-stepping method in the spirit of reduced SQP-methods. They develop a preconditioner working on the whole system of equations with state, costate and design equations.

In the One-shot approach used here, the idea is that for fixed parameters there is a given (not necessarily (pseudo-) time-stepping) strategy to solve the state equations. This strategy is assumed to demand no or disallow any changes. In each iteration step the update of the state is augmented by an update of the adjoint state and a kind of quasi-Newton step for the design correction with the distinctive feature that the required preconditioner controls convergence of the whole system. Here, the preconditioner is a squared matrix of only the size of the number of parameters.

Since the assumptions in the theoretical analysis of the One-shot method are very strict and the computation of the preconditioner seems at first glance laborious and expensive, the intention of this paper is to check the applicability of the One-shot strategy for real world problems and possibly propose simplifications. We compare numerical results to the gradient based BFGS and limited-memory BFGS (L-BFGS) methods. We set aside the comparison to genetic or so-called intelligent search algorithms, see e.g. [9], because the aim of the One-shot approach according to the authors of [4] and [5] is to offer an alternative to local gradient-based optimization techniques. Genetic algorithms usually demand a high number of function evaluations which we want to avoid because of the costly computation of steady states needed for the function evaluation.

In this paper, we apply the One-shot approach to a box model of the North Atlantic. This problem is different from the application in [6] in that the parameters enter in a nonlinear fashion resulting in so-called non-separable adjoints where the adjoint is no longer only the sum of a term on the state and a term on design.

The outline of this paper is the following. In section 2 we recall the One-shot optimization strategy according to [4] and [5]. We apply the One-shot method to an example in earth system modeling in section 3. There, we describe the Rahmstorf 4-box-model, the optimization problem and present numerical results. Section 4 draws conclusions.

2. One-shot Optimization Strategy

In this section, we recall the One-shot optimization strategy according to [4] and [5], its quintessence and difference to conventional optimization methods, and we derive and explain the One-shot iteration step. First of all, we describe the mathematical problem behind the parameter optimization problem.

2.1. Problem formulation

Parameters u of a model describing physical, biological, chemical or other real life phenomena are usually determined by fitting model output $y = y(u)$ to observed data denoted by y_{data}. This data can also be taken from other, more comprehensive models.

The fitting procedure then is a mathematical optimization problem with a least-squares cost functional with some regularization term

$$J(y,u) = \frac{1}{2}\|y - y_{data}\|_2^2 + \frac{\alpha}{2}\|u - u_{guess}\|_2^2, \alpha \in \mathbb{R}_0^+$$

under the constraint that model equations, namely $c(y,u) = 0$, are fulfilled.

In climate modeling, model equations are usually partial and/or ordinary differential equations solved by an iterative process.

The problem will become more difficult with respect to uniqueness of minima and computation of derivative information, if the quantity to be fit to data g_{data} is computed from a functional $g(y,u)$ such that J then is

$$J(y,u) = \frac{1}{2}\|g(y,u) - g_{data}\|_2^2 + \frac{\alpha}{2}\|u - u_{guess}\|_2^2.$$

In the finite dimensional case or the discretized version, where $y \in Y \subset \mathbb{R}^n$, $u \in U \subset \mathbb{R}^m$ and $g : Y \times U \to \mathbb{R}^l$, the cost function is the sum of the squared differences

$$\begin{aligned} J(y,u) &= \frac{1}{2}\sum_{i=1}^{l}(g_i(y,u) - g_{i,data})^2 \\ &+ \frac{\alpha}{2}\sum_{i=1}^{m}(u_i - u_{i,guess})^2. \end{aligned}$$

Here, the objective function J is $J : Y \times U \to \mathbb{R}$, $y \in Y$ is the state, $u \in U$ is the design or parameter vector to be optimized. With the help of the regularization term $\frac{\alpha}{2}\|u - u_{guess}\|_2^2$ parameters u are kept in an acceptable or presumed range around parameter values u_{guess}, where elements $u_{i,guess}$ can for example be taken as mean values of some maximum and minimum values. We assume J to be $C^{2,1}$, which means twice continuously differentiable in y and once in u. We further assume the Jacobian of c with respect to y, denoted c_y, to always be invertible, such that with the mean value theorem, there exists only one y^* with $c(y^*,u) = 0$ for a fixed u.

2.2. One-shot iteration and its properties

In practice, finding an analytical solution for a feasible state y^* with $c(y^*,u) = 0$ is often impossible. That is why usually an iterative method is called upon.

For the One-shot strategy, we assume that there is a given fixed point iteration, also called model spin-up , which has already been found reliable and successful in the search for the feasible state y^* for parameters u. Included step size or preconditioner strategies can be carried over and do not have influence on the One-shot iteration. Thus, there is a *given* contraction, (pseudo-) time-stepping strategy or fixed point iteration G, where y^* satisfies $y^* = G(y^*,u) = \lim_{k\to\infty} G(y_k,u)$.

The fundamental idea of the One-shot approach is to *reformulate* the condition $c(y,u) = 0$ into the *fixed point equation* $y = G(y,u)$. The iteration function $G : Y \times U \to Y$ is assumed to be $C^{2,1}$ with the contraction factor $\rho < 1$, i.e. for a suitable inner product norm $\|\cdot\|$ we have for G_y, denoting the Jacobian of G with respect to y, that

$$\|G_y(y,u)\| \leq \rho < 1, \qquad \forall y \in Y. \tag{1}$$

from which follows

$$\|G(y_1,u) - G(y_2,u)\| \leq \rho\|y_1 - y_2\|, \forall y_1, y_2 \in Y. \tag{2}$$

With the contraction property of G we can infer from the Banach fixed point theorem, for fixed u, the sequence $y_{k+1} = G(y_k,u)$ converges to a unique limit $y^* = y^*(u)$.

The assumptions on the model function c and the contraction G are very strict and rarely analytically or even numerically provable. However, we will see in our numerical example, that the One-shot strategy even converges under weaker assumptions on the contraction G. Here, in our example of the 4-box-ocean-model only the *Ciric* or *quasi-contraction* property, see [10], is fulfilled. With the help of the fixed point reformulation, the optimization problem can be written as

$$\min_{y,u} J(y,u) \quad s.t. \quad y = G(y,u). \tag{P}$$

A conventional optimization strategy performs the following steps:

In the outer loop do in the k-th iteration step:

- *Perform a complete model spin-up (inner loop) with parameter values u_k and obtain an admissible state $y_k = y^*(u_k) = \lim_{l\to\infty} G(y_l,u_k)$.*
- *Compute the value of the cost function $J(y_k,u_k)$.*
- *Adjust model parameters obtaining u_{k+1}.*

End the outer loop when a sufficient optimality condition is satisfied.

Of course, adjusting the parameters demands further full model spin-ups and/or expensive derivative information for whose computation again full model spin-ups are necessary.

The main idea of the One-shot strategy is to adjust model parameters already during the model spin-up.

Using the method of Lagrange Multipliers, in the finite dimensional case, the associated Lagrangian to problem (P) with the Lagrange multiplier or *adjoint state* $\bar{y} \in \bar{Y}$ is

$$\begin{aligned} L(y, \bar{y}, u) &= J(y, u) + \bar{y}^\top (G(y, u) - y) \\ &= N(y, \bar{y}, u) - \bar{y}^\top y, \end{aligned}$$

where we introduce the shifted Lagrangian N as

$$N(y, \bar{y}, u) \;:=\; J(y, u) + \bar{y}^\top G(y, u).$$

A Karush-Kuhn-Tucker (KKT) point $(y^*, \bar{y}^*, u^*)$ fulfilling the first order necessary optimality condition must satisfy

$$\left.\begin{aligned} 0 = \tfrac{\partial L}{\partial y} =& N_y(y^*, \bar{y}^*, u^*) - \bar{y}^{*\top} \\ =& J_y(y^*, u^*) + \bar{y}^{*\top} G_y(y^*, u^*) - \bar{y}^{*\top}, \\ 0 = \tfrac{\partial L}{\partial \bar{y}} =& G(y^*, u^*) - y^*, \\ 0 = \tfrac{\partial L}{\partial u} =& N_u(y^*, \bar{y}^*, u^*) \\ =& J_u(y^*, u^*) + \bar{y}^{*\top} G_u(y^*, u^*). \end{aligned}\right\} \quad (3)$$

Motivated by this system of equations, the following coupled full step iteration, called *One-shot strategy* according to the authors of [4], [5], to reach a KKT point is derived:

Do in the k-th iteration step:

$$\left.\begin{aligned} y_{k+1} &= G(y_k, u_k), \\ \bar{y}_{k+1} &= N_y(y_k, \bar{y}_k, u_k)^\top \\ u_{k+1} &= u_k - B_k^{-1} N_u(y_k, \bar{y}_k, u_k) \end{aligned}\right\} \quad (4)$$

until there is (numerically) no change in $(y_k, \bar{y}_k, u_k)$.

Here, B_k is a design space preconditioner which must be selected to be symmetric positive definite. As mentioned above, we do not want to introduce additional preconditioners for the updates of y and $\bar{y}$, because of the assumption that the model spin-up has already been found reliable and successful in the search for steady states.

The contractivity (2) ensures that the first equation in the coupled iteration step (4) converges ρ-linearly for fixed u. Although the second equation exhibits a certain time-lag, it converges with the same asymptotic R-factor (see [11]). As far as the convergence of the coupled iteration (4) is concerned, the goal is to find B_k that ensures that the spectral radius of the coupled iteration (4) stays below 1 and as close as possible to ρ. In subsection 2.3, we recall the formula of appropriate preconditioners B_k according to the authors of [4], [5].

Required derivatives and automatic differentiation

For the One-shot update (4) and also later in the computation of the preconditioners B_k, a lot of derivative information is needed. The costs for its calculation are small compared to those of a conventional approach, because they only depend on the previous iteration step value. The storing/recomputation of intermediate partial derivatives, as for example $\frac{\partial y}{\partial u}$ for the computation of derivatives of J or N with respect to u, is not necessary which is one of the main differences and advantages compared to traditional optimization techniques.

Applying a tool for automatic/algorithmic differentiation (AD) can even more reduce costs and most importantly, AD computes *exact* derivatives without any approximation errors.

AD is a software technology to compute the derivative of a function at costs of only a small multiple of the costs for the evaluation of the function itself. With the help of source code transformation or operator overloading an AD tool provides the user with a computer programme containing the derivatives.

Those tools are for example TAF or ADiMat, which use the source code transformation approach to generate FORTRAN or MATLAB subroutines to calculate function values and derivative information in one call, see [12] and [13], or for example ADOL-C using the operator overloading concept in C/C++ codes, see [14].

Regarding the One-shot optimization strategy, we need gradients (namely J_y, J_u) and vector-Jacobian-products which can cheaply be obtained with the reverse mode of AD. For the calculation of the preconditioner B also second derivatives and full Jacobians are needed which are calculated via the application of the reverse mode first and the forward mode afterwards. In our testings, we apply the (commercial) AD tool TAF for FORTRAN subroutines.

2.3. Preconditioner B and the doubly augmented Lagrangian

In this section, we explain the choice of the preconditioners B_k according to [4] and [5]. For the sake of simplicity, we omit the iteration index k using the notation B.

For the derivation of the preconditioner B, we introduce the doubly augmented Lagrangian L^a

$$
\begin{aligned}
L^a(y,\bar{y},u) &= \frac{\alpha_L}{2}\|G(y,u)-y\|^2 \\
&\quad +\frac{\beta_L}{2}\|N_y(y,\bar{y},u)^\top-\bar{y}\|^2 \\
&\quad +N(y,\bar{y},u)-\bar{y}^\top y,
\end{aligned}
$$

which is the Lagrangian of the original problem augmented by the errors in primal and dual feasibility. Here $\alpha_L > 0$ and $\beta_L > 0$ are weighting coefficients.

The authors of [4] prove that under certain conditions on α_L and β_L (see below), stationary points of problem (P) are also stationary points of L^a and that L^a is an exact penalty function. This leads to the idea to choose B as an approximation to the Hessian of L^a, i.e. $B \approx \nabla_{uu}L^a$.

In [4], it is proven that descent of the augmented Lagrangian is provided for any preconditioner B fulfilling

$$B \succeq B_0 := \frac{1}{\sigma}(\alpha_L G_u^\top G_u + \beta_L N_{yu}^\top N_{yu}) \tag{5}$$

i.e. $B - B_0$ is positive semidefinite, and where

$$\sigma := 1-\rho-\frac{(1+\frac{\|N_{yy}\|}{2}\beta_L)^2}{\alpha_L\beta_L(1-\rho)}. \tag{6}$$

The authors of [4] propose to choose α_L and β_L such that B_0^{-1} is as large as possible. Using (5) we get

$$
\begin{aligned}
\|B_0\|_2 &= \frac{1}{\sigma}\|\alpha_L G_u^\top G_u + \beta_L N_{yu}^\top N_{yu}\|_2 \\
&\le \frac{1}{\sigma}(\alpha_L\|G_u\|_2^2 + \beta_L\|N_{yu}\|_2^2).
\end{aligned}
$$

Minimizing the right most formula as a function of α_L and β_L and replacing σ with (6) yields: Under the assumption that $\sqrt{\alpha_L\beta_L}(1-\rho) > 1+\frac{\beta_L}{2}\|N_{yy}\|$ holds and $\|N_{yy}\| \neq 0$ we obtain

$$
\begin{aligned}
\beta_L &= \frac{3}{\sqrt{\|N_{yy}\|^2 + 3\frac{\|N_{yu}\|^2}{\|G_u\|^2}(1-\rho)^2} + \frac{\|N_{yy}\|}{2}} \quad \text{and} \\
\alpha_L &= \frac{\|N_{yu}\|^2\beta_L(1+\frac{\|N_{yy}\|}{2}\beta_L)}{\|G_u\|^2(1-\frac{\|N_{yy}\|}{2}\beta_L)}.
\end{aligned}
$$

As mentioned above, we pursue to $B \approx \nabla_{uu}L^a$. It turns out that at a stationary point of L^a, where primal and dual feasibility hold, the Hessian of L^a, namely $\nabla_{uu}L^a$, is

$$\nabla_{uu}L^a = \alpha_L G_u^\top G_u + \beta_L N_{yu}^\top N_{yu} + N_{uu}.$$

As L^a is an exact penalty function, we have $\nabla_{uu}L^a \succ 0$ in a neighbourhood of the constrained optimization solution. Assuming that $N_{uu} \succ 0$ implies that the preconditioner

$$B = \frac{1}{\sigma}(\alpha_L G_u^\top G_u + \beta_L N_{yu}^\top N_{yu} + N_{uu}) \tag{7}$$

fulfills (5) and thus the step $\Delta u_k = -B^{-1}N_u(y_k,\bar{y}_k,u_k)$ of the coupled iteration (4) yields descent on L^a.

2.3.1. BGFS update to avoid computation of full Jacobians and 2nd order derivatives

In the calculation of the preconditioner B full Jacobians and second derivatives are needed. On the one hand, those can also be calculated by algorithmic differentiation, but on the other hand, a possibility to avoid this is the application of a Low-Rank BFGS update to update the inverse approximation H_k of B_k. In view of the relation $B \approx \nabla_{uu}L^a$, we use the following secant equation in the update of H_k: $H_{k+1}R_k = \Delta u_k$, where

$$
\begin{aligned}
R_k := &\nabla_u L^a(y_k,\bar{y}_k,u_k+\Delta u_k) \\
&-\nabla_u L^a(y_k,\bar{y}_k,u_k).
\end{aligned}
$$

Imposing to the step multiplier η to satisfy the second Wolfe condition

$$
\begin{aligned}
&\Delta u_k{}^\top \nabla_u L^a(y_k,\bar{y}_k,u_k+\eta\Delta u_k) \\
&\quad \ge c_2\Delta u_k{}^\top\nabla_u L^a(y_k,\bar{y}_k,u_k)
\end{aligned}
$$

with $c_2 \in [0,1]$, implies the necessary curvature condition

$$R_k{}^\top \Delta u_k > 0. \tag{8}$$

A simpler procedure could skip the update whenever (8) does not hold by either setting H_{k+1} to identity or to the last iterate H_k. Provided (8) holds, we use

$$
\begin{aligned}
H_{k+1} &= (I - r_k\Delta u_k R_k{}^\top)H_k(I - r_k R_k\Delta u_k{}^\top) \\
&\quad + r_k\Delta u_k\Delta u_k{}^\top
\end{aligned}
$$

with $r_k = \frac{1}{R_k{}^\top\Delta u_k}$.

The weights α_L, β_L of L^a require norms of second order derivatives. In [5], the authors propose simpler approximations according to two different approaches. In the first version then

$$\alpha_L = \frac{2\|N_{yy}\|_2}{(1-\rho)^2} \quad \text{and} \quad \beta_L = \frac{2}{\|N_{yy}\|_2},$$

in the second approach

$$\alpha_L = \frac{6\|N_{yy}\|_2}{(1-\rho)^2} \quad \text{and} \quad \beta_L = \frac{6}{\|N_{yy}\|_2}.$$

$\|N_{yy}\|_2$ can be computed via the power iteration.

For the BFGS update, the calculation of R_k requires a pure design step (step with fixed primal and dual variables y and $\bar{y}$ respectively), which might be computed at high costs. We will pay attention to this fact in our numerical example.

3. Application in Earth System Modeling

To exemplify the benefit of the One-shot optimization strategy in the case of climate research, we present the application to a 4-box-model of the Atlantic Thermohaline Circulation. The 4-box-model described in [15] simulates the flow rate of the Atlantic Ocean known as the 'conveyor belt', carrying heat northward and having a significant impact on climate in northwestern Europe. Temperatures T_i and salinity differences S_i in four different boxes $i = 1, ..., 4$, namely the southern, northern, tropical and the deep Atlantic, are the characteristics inducing the flow rate. The surface boxes exchange heat and freshwater with the overlying atmosphere, which causes a pressure-driven circulation, compare figure 1.

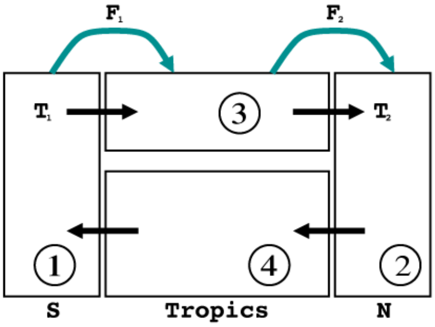

Figure 1. Rahmstorf box model, flow direction shown for $m > 0$.

In [16] a smooth coupling of the two possible flow directions is proposed. The resulting time dependent ODE system reads:

$$\begin{aligned}
\dot{T}_1 &= \lambda_1(T_1^* - T_1) + \tfrac{m^+}{V_1}(T_4 - T_1) + \tfrac{m^-}{V_1}(T_3 - T_1)\\
\dot{T}_2 &= \lambda_2(T_2^* - T_2) + \tfrac{m^+}{V_2}(T_3 - T_2) + \tfrac{m^-}{V_2}(T_4 - T_2)\\
\dot{T}_3 &= \lambda_3(T_3^* - T_3) + \tfrac{m^+}{V_3}(T_1 - T_3) + \tfrac{m^-}{V_3}(T_2 - T_4)\\
\dot{T}_4 &= \tfrac{m^+}{V_4}(T_2 - T_4) + \tfrac{m^-}{V_4}(T_1 - T_4)\\
\dot{S}_1 &= \tfrac{S_0 f_1}{V_1} + \tfrac{m^+}{V_1}(S_4 - S_1) + \tfrac{m^-}{V_1}(S_3 - S_1)\\
\dot{S}_2 &= -\tfrac{S_0 f_2}{V_2} + \tfrac{m^+}{V_2}(S_3 - S_2) + \tfrac{m^-}{V_2}(S_4 - S_2)\\
\dot{S}_3 &= \tfrac{S_0(f_2 - f_1)}{V_3} + \tfrac{m^+}{V_3}(S_1 - S_3) + \tfrac{m^-}{V_3}(S_2 - S_4)\\
\dot{S}_4 &= \tfrac{m^+}{V_4}(S_2 - S_4) + \tfrac{m^-}{V_4}(S_1 - S_4)
\end{aligned}$$

where for some positive a, $m^+ = \frac{m}{1-e^{-am}}$ almost coincides with the meridional volume transport or overturning

$$m = k(\beta_m(S_2 - S_1) - \alpha_m(T_2 - T_1))$$

for positive m and is almost zero for negative m. The term $m^- = \frac{-m}{1-e^{am}}$ becomes almost zero for positive m and $-m$ for negative m. That means the summands including m^+ and m^- are activated or deactivated depending on the flow direction. The deviation from the physically correct model becomes smaller the larger a is. Several model parameters are involved, the most important being the freshwater flux f_1 containing atmospheric water vapor transport and wind-driven oceanic transport; they are used to simulate global warming in the model and are chosen in the interval $[-0.2, 0.15]$. T_i^*, $i = 1, 2, 3$ are so-called restoring temperatures, which can be seen as counterparts of the three surface temperatures. Further model parameters are physical, relaxation and coupling constants among which there are well-known fixed parameters and those which are tunable parameters. See [15] for an explanation of the occurring constants, fixed parameters and tunable parameters.

3.1. The optimization problem

As mentioned in the introduction, in climate modeling an optimization is at first performed for steady states, which means in this example for temperatures and salinities which do not change in time anymore. Given fresh water fluxes $(f_{1,i})_{i=1}^l$, corresponding to different warming scenarios, the aim is to fit the overturning values $m_i = m(y(f_{1,i}), u)$ computed from stationary temperatures and salinities $(T_1, T_2, S_1, S_2)_i$ obtained by the model spin-up for $f_{1,i}$ to data $m_{d,i}$ from a more complex model *Climber2*, see [17]. $u = (T_1^*, T_2^*, T_3^*, \Gamma, k, a)$ are the control parameters. Here, Γ is a thermal coupling constant in the computation of the thermal relaxation constants λ_i, $i = 1, 2, 3$. All other parameters occur in the model description of the previous subsection. Using notations from section 2, the state is $y = (y_i)_{i=1}^l$ with $y_i = y(f_{1,i}) = (T_1, T_2, T_3, T_4, S_1, S_2, S_3, S_4)_i$.

If $F(y, u)$ denotes the right-hand side of the ODE system of the model, we get

$$\begin{aligned}
\min_{y,u} J(y,u) &:= \frac{1}{2}\|m(y(f_1), u) - m_d\|_2^2\\
&\quad + \frac{\alpha}{2}\|u - u_{guess}\|_2^2,\\
\text{s.t.}\quad 0 &= F(y(f_{1,i}), u), \quad i = 1, ..., l.
\end{aligned}$$

The regularization term incorporates a prior guess u_{guess} for the parameters. The larger α the more the parameters u are kept close to u_{guess}.

The difficulty here is that $m : \mathbb{R}^{8l} \times \mathbb{R}^6 \to \mathbb{R}^l$ is not injective. There are several combinations of steady/feasible T_1, T_2, S_1, S_2 and the parameter $u(5) = k$ to compute the same overturning m. The smaller α the more likely the different optimization strategies find completely different optimal parameters with almost the same function values $J(y^*, u^*)$.

In [15] the authors apply the Explicit Euler time stepping with a fixed step size of one year, i.e. $\Delta t = 1$, to run the model into a steady state. Otherwise, known model constants scaled on a time span of one year must be adjusted. Thus G defined in section 2 here represents one full Euler step $y_{k+1} = G(y_k, u) = y_k + F(y_k, u)$ operating on all freshwater fluxes $f_{1,i}$ together, i.e. for fixed u we have $G(\cdot, u) : \mathbb{R}^{8l} \to \mathbb{R}^{8l}$.

In this example, contractivity of G is not given in general, i.e. ρ in (1) exceeds 1 for several steps. However, in average it is less than 1. Here, for the explicit Euler sequence $y_{k+1} = G(y_k) = y_k + F(y_k, u)$, the quasi-contraction property [10]

$$\|y_{k+1} - y_k\| \leq q \max\{\|y_k - y_{k-1}\|, \|y_{k+1} - y_{k-1}\|\}$$

for $0 \leq q < 1$ holds. In our testings, G converges for fixed u but different starting values y_0 to the same stationary y^*.

3.2. Numerical results and discussion

In our numerical testing, we compare the two versions of the One-shot method, with full computation of the preconditioner B on the one hand and BFGS update of B on the other hand, to a traditional BFGS-quasi-Newton optimization approach. Furthermore, we compare results to values obtained by the Limited-memory BFGS (L-BFGS) algorithm implemented by Zhu, Byrd, Nocedal and Morales, see [18], version 3.0 from 2011, without and finally with box constraints on the control parameters (L-BFGS-B) because we find that computed optimal parameter values of the BFGS and L-BFGS method are far away from actual real world values. In the three different BFGS approaches, for each parameter value u_k during the optimization process the box model has to be run into a steady state. In our example, that takes between 4,000 and 15,000 Euler steps. Compared to more complex climate models, here the Euler time step evaluation is not expensive. However, during the optimization process a large number of Explicit Euler time steps will accumulate and for derivative calculation a huge amount of recomputations, storing or both is necessary. That becomes obvious in the calculation of derivatives using automatic differentiation. Whereas for the BFGS method in the reverse mode it is necessary to store all Euler steps until a steady state is reached, in the One-shot method the required derivatives depend on the current values only, i.e. on only one Euler step.

In our implementation we replaced $u_i = T_i^*$, $i = 1, ..., 3$, with $\tilde{u}_i = W_i$ such that $u_i = \tilde{u}_i + u_{fix}$ where $(u_{fix})_1^3 = (6.64, 2.68, 11.69)$, which are optimal values calculated in [15]. W_i, $i = 1, ..., 3$, can be interpreted as warming trends. We chose $u_{guess} = (0., 0., 0., 23., 25., 500)$ as starting parameters. Since only quasi-contraction is given, we expect the contraction factor ρ to exceed 1 for several iteration steps possibly resulting in arithmetic exceptions. Therefore, we fix ρ close to 1, namely $\rho = 0.9$.

For better initialization especially of the adjoint, we propose an update of only the state and adjoint state for the first 500 iteration steps.

The One-shot-BFGS strategy demands a line-search procedure, otherwise the method fails. Here, we applied a simple strategy constantly halfing the steplength until there is a reduction in the costfunction with the resulting step.

We perform our numerical testings on a SUN-W-Ultra-SPARC-IIIi CPU 1.3GHz machine.

3.2.1. Influence of rare update of weighting coefficients of the preconditioners B_k on the optimization

In the first version, we calculate preconditioners B_k defined in (7) in every iteration including all first and second order derivatives. Also the weighting coefficients α_L, β_L and σ are adjusted. We find, that the weights do not change significantly from iteration to iteration. As one can see in Table 1, an update performed only after several time-steps does not significantly influence the optimization but the computational time needed. Therefore, we prefer the version with a calculation of α_L, β_L and σ every 1,000 iterations.

3.2.2. Effect of the weighting factor α on the numerical results

In the following, our attention is drawn on the effect of the weighting factor α in front of the penalty term $\|u - u_{guess}\|$. For the last parameter a we chose the additional factor 0.01, because a is of higher dimension than the other parameters and can vary more. Here in the example of the 4-box-model, without any regularization, i.e. $\alpha = 0$, the One-shot method and the L-BFGS method without constraints do not converge or fail. The BFGS method and the L-BFGS with box constraints terminate with parameter values u^* where $\|J_u(y(u^*), u^*)\|$ still is very large, but the algorithms cannot find descent directions.

We recall from section 3.1 that the considered optimization problem has several local minima

Table 1. Effect on the optimization of rare update of the weights α_L, β_L and σ for $\alpha = 0.1$. We compare the values of the cost functional, the weighted data fit, the number of iterations and the computational time in minutes.

update of weights of B	$J(y^*, u^*)$	data fit	# iterations	comp.time
every 10,000 iterations	14.544	0.399	1,185,500	5.085
every 1,000 iterations	14.544	0.399	1,182,053	5.067
every iteration	14.544	0.399	1,181,701	10.045

which might be of the same quality regarding the data fit, even though the obtained model parameters are completely different. The larger the weighting factor α the less the obtained parameters vary.

In our testings with $\alpha > 0$, we compare the optimal value of the cost function, the data fit weighted to the number of observations, the number of iteration steps, the number of needed Euler steps, and the computational time in minutes. Furthermore, we take a look at the quality of optimality, which means for the One-shot strategies the norm of $L_{(y,\bar{y},u)}(y^*, \bar{y}^*, u^*)$ and for the BFGS methods the norm of $J_u(y(u^*), u^*)$. The numerical results are collected in tables 2 and 3 and illustrated in figures 2 and 3.

Not surprisingly, one generally detects that the smaller α the better the fit of data becomes.

We observe that for different α the qualities of the methods vary. Especially for large fresh water fluxes $f_{1,i}$ the outputs of the different optimization strategies strongly differ. These are $f_{1,i}$ for which the model switches the flow direction of m during the model spin-up.

Comparing the original One-shot and the One-shot-BFGS methods, the presumption that the One-shot-BFGS strategy might be rather time consuming due to the additional pure design steps is confirmed. Here, in an example with a very small number of parameters to be optimized, the One-shot-BFGS approach is not recommended. However, in problems with a large number of design variables, the One-shot-BFGS approach might be an alternative. The computed data fit can be regarded as equally good in this example.

For $\alpha = 10$ the strategies show almost no difference in their results, neither in the fit of the data nor in the computed optimal parameters. Concerning computational time and the number of Euler steps, the original One-shot strategy perform best.

For $\alpha = 1$ and $\alpha = 0.1$ the One-shot strategy shows difficulties in performance. We suspect that here the balance between keeping parameters close to u_{guess} and reducing the misfit has a disadvantageous influence on the One-shot method. However, also the BFGS method does not perform well for $\alpha = 0.1$.

For smaller α, we observe significant differences. The unconstrained BFGS strategies find the best fit, but parameter values (u_1^*, u_2^*, u_3^*) which are not acceptable in this real world problem. L-BFGS-B computes similar results as the One-shot method, but needs far more Euler steps and therewith a much longer computational time.

We detect that the parameters computed by the One-shot method stay in acceptable ranges without any box constraints. Computed parameters are to some extend similar to those of the L-BFGS-B method.

One main goal of the One-shot strategy was to achieve so-called bounded retardation for the speed of convergence compared to the number of time-steps needed to run the model into a steady state. Since the Explicit Euler time-stepping does not show quick converge and ratios $\theta_k = \frac{\|y_{k+1}-y_k\|}{\|y_k-y_{k-1}\|}$ even exceed 1 for several steps k, one cannot expect the One-shot method to converge very fast in this special example. The average value for θ in a pure model spin-up with parameters taken from [15] is 0.992 and for the One-shot strategy ($\alpha = 0.1$) $\theta = 0.9999884$.

Furthermore, the number of One-shot iteration steps was intended to exceed the number of Euler steps of a single model spin-up not too much. Especially for parameter sets near the computed solution, a model spin-up with fixed parameters needs 12,000 to 15,000 Euler steps. For $\alpha \in \{10, 0.01, 0.001\}$ where the One-shot strategy shows good performance, the observed number of iterations is about 10 to 40 times larger than the number of Euler steps for one single spin-up. Considering that the BFGS strategies need at least about 30 optimization steps requiring further function evaluations and model spin-ups the factor is not very large. Even in those cases, where the One-shot strategies does not show quick convergence, the number of iterations still is not too far away from the number of Euler steps required by the BFGS strategies. In applications, where the fixed point iteration G is more expensive than the Euler time-stepping applied in this example, the One-shot strategy

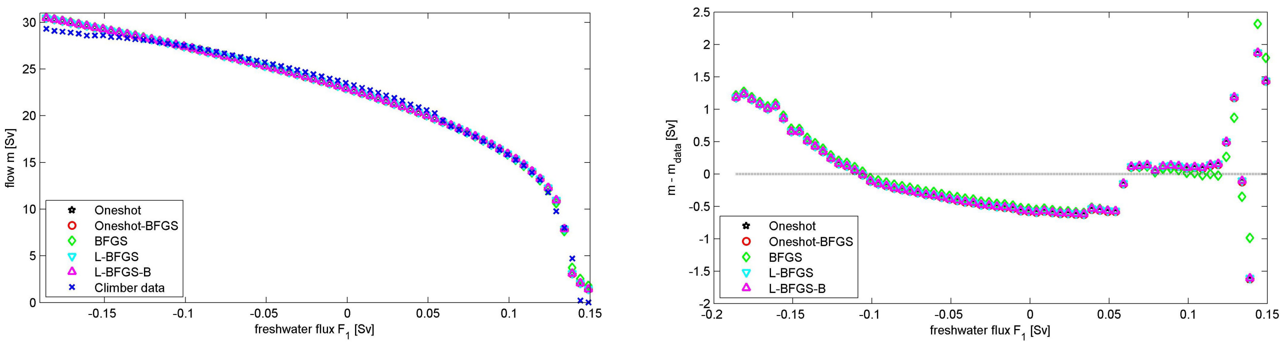

Figure 2. Results of the optimization comparing the One-shot strategies with the BFGS-quasi-Newton methods for $\alpha = 0.1$ (left) and the differences to the Climber data (right).

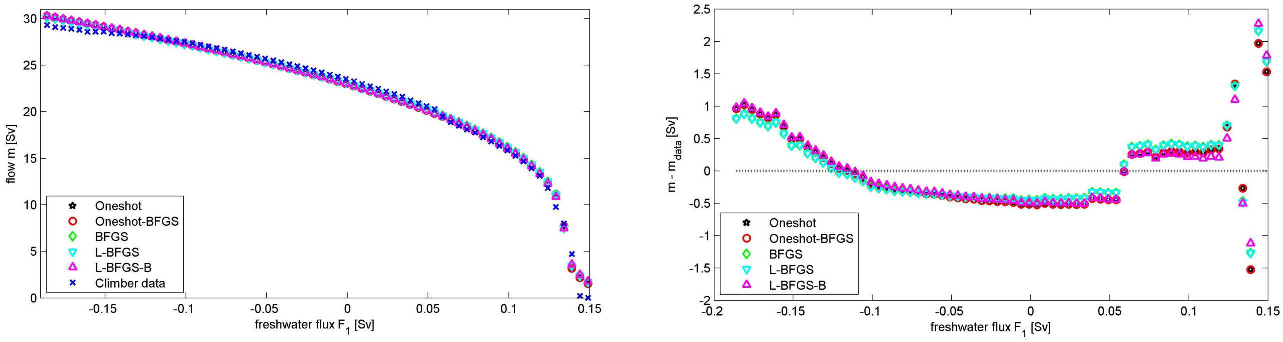

Figure 3. Results of the optimization comparing the One-shot strategies with the BFGS-quasi-Newton method for $\alpha = 0.001$ (left) and the difference to the Climber data (right).

then might catch up with the needed computational time.

4. Conclusions

We have successfully applied the One-shot method according to Hamdi and Griewank, [4], [5], to a parameter optimization problem in ocean modeling. We have analyzed its applicability and find that the One-shot strategy presents a promising approach to optimize models consuming much time and calculational costs for their spin-ups using (pseudo-)time stepping or a fixed point iteration. Our numerical example was the parameter optimization of the Rahmstorf 4-box-model of the North Atlantic with steady states achieved via an Explicit Euler spin-up. Optimization results of the original One-shot strategy and the One-shot-BFGS method with an BFGS update of the preconditioner of the parameter correction step are compared to a classical BFGS-quasi-Newton method and the L-BFGS-method with and without box constraints on the parameters.

We observed that the One-shot-BFGS strategy does not show good performance in this example with only 6 parameters. The original version with full computation of the preconditioner performs well for large and very small weighting factors α in front of the penalty term. Further analysis on why One-shot has difficulties in finding optimal values for weights $\alpha \in \{1, 0.1\}$ can be valuable.

We have found out that the One-shot method can be applied even though contractivity is not given in general and that fixing the contraction factor ρ to a number close to 1 is adequate. Furthermore, computation of the weights of B is not mandatory in each iteration step.

Considering examples with more expensive model spin-ups, the One-shot method might on the one hand even gain (or at least catch up in those examples with slow convergence) concerning computational time and on the other hand be the only applicable alternative for derivative based optimization methods, because derivatives depend on one spin-up step only instead of the whole spin-up, which is the main difference and advantage compared to standard methods. The application to earth system models involving nonlinear PDEs and/or a higher spatial resolution with computationally more expensive model solvers and periodic solutions will be of great interest for future investigations to demonstrate the efficiency of the One-shot approach.

Table 2. Results of the optimization comparing values of the cost functional, the weighted data fit, the number of iterations of the optimization procedure, the number of needed Euler steps, the quality of optimality ($\|L_{(y,\bar{y},u)}(y^*,\bar{y}^*,u^*)\|$ for methods 1-2 and $\|J_u(y(u^*),u^*)\|$ for methods 3-5 respectively) and the computational time in minutes

	Method	$J(y^*,u^*)$	data fit	#iterations	#Euler steps	opt.cond.	comp.time
$\alpha = 10$	1	25.810	0.539	254,859	254,859	2.0E-1	1,086
	2	25.810	0.539	220,687	220,687	1.5E-1	1.864
	3	25.809	0.539	41	761,661	2.5E-5	1.715
	4	25.809	0.539	31	411,732	5.5E-4	1.351
	5	25.809	0.539	39	540,946	2.2E-4	1.785
$\alpha = 1$	1	17.438	0.462	1,212,057	1,212,057	1.8E-1	5.155
	2	17.434	0.462	1,101,992	1,101,992	1.3E-1	9.315
	3	17.426	0.462	48	926,049	1.3E-4	2.084
	4	17.426	0.462	46	653250	3.3E-4	2.142
	5	17.426	0.462	53	1,194,483	3.2E00	3.938
$\alpha = 0.1$	1	14.544	0.399	1,182,053	1,182,053	2.3E-1	5.067
	2	14.469	0.398	3,122,016	3,122,016	1.3E-1	26.366
	3	15.571	0.401	54	1,403,864	3.1E-1	2.878
	4	14.417	0.393	71	1,250,796	2.6E-2	4.100
	5	14.417	0.393	76	1,412,728	6.0E-4	4.657
$\alpha = 0.01$	1	13.747	0.396	437,543	437,543	2.2E-1	1.917
	2	13.786	0.396	344,463	344,463	7.0E-1	2.906
	3	12.514	0.338	52	1,455,877	2.3E-1	3.137
	4	fails					
	5	13.747	0.397	29	672,388	4.1E00	2.217
$\alpha = 0.001$	1	12.232	0.352	150,410	150,410	1.4E-1	0.649
	2	12.364	0.353	170,597	170,597	6.6E-1	1.442
	3	11.411	0.331	64	2,134,827	5.9E-1	4.457
	4	11.412	0.331	63	2,018,101	3.6E-1	6.233
	5	12.257	0.352	77	2,928,986	4.2E00	9.078

Legend of Methods: 1 One-shot, 2 One-shot-BFGS, 3 BFGS, 4 L-BFGS, 5 L-BFGS-B

References

[1] Griewank, A., *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM, Philadelphia, PA (2000).

[2] Christianson, B., Reverse accumulation and implicit functions. *Optimization Methods and Software*, 9(4), 307–322 (1998).

[3] Kaminski, T., Giering, R., and Voßbeck, M., Efficient sensitivities for the spin-up phase. Automatic Differentiation: Applications, Theory, and Implementations, Lecture Notes in Computational Science and Engineering, Springer, New York, 50, 283–291 (2005).

[4] Hamdi, A. and Griewank, A., Reduced Quasi-Newton Method for Simultaneous Design and Optimization. Comput. Optim. Appl. online, Available at `www.springerlink.com` (2009).

[5] Hamdi, A. and Griewank, A., Properties of an Augmented Lagrangian for Design Optimization. *Optimization Methods and Software*, 25(4), 645–664 (2010).

[6] Özkaya, E. and Gauger, N., Single-Step One-Shot Aerodynamic Shape Optimization. *International Series of Numerical Mathematics*, 158, 191–204 (2009).

[7] Ta'asn, S., Pseudo-Time Methods for Constrained Optimization Problems Governed by PDE. ICASE Report No. 95-32 (1995).

[8] Hazra, S. B. and Schulz, V., Simultaneous Pseudo-Timestepping for PDE-Model Based Optimization Problems. *BIT Numerical Mathematics*, 44, 457–472 (2004).

[9] Pham, D. and Karaboga, D., Intelligent Optimisation Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks. Springer London, Limited (2012).

[10] Ciric, L. B., A Generalization of Banach's Contraction Principle. *Proceedings of the American Mathematical Society*, 45(2), 267–273 (1974).

Table 3. Optimal parameters computed by the different optimization strategies.

	Method	u_1	u_2	u_3	u_4	u_5	u_6
$\alpha = 10$	One-shot	0.482	0.457	-0.916	23.211	25.283	502.83
	One-shot-BFGS	0.480	0.457	-0.918	23.211	25.283	502.83
	BFGS	0.475	0.451	-0.927	23.209	25.280	502.83
	L-BFGS	0.475	0.451	-0.927	23.209	25.280	502.83
	L-BFGS-B	0.475	0.451	-0.927	23.209	25.280	502.83
$\alpha = 1$	One-shot	0.585	0.583	-0.957	23.492	24.902	512.44
	One-shot-BFGS	0.569	0.568	-0.977	23.488	24.893	512.42
	BFGS	0.521	0.522	-1.042	23.474	24.865	512.34
	L-BFGS	0.521	0.521	-1.042	23.474	24.865	512.34
	L-BFGS-B	0.520	0.520	-1.042	23.474	24.866	512.34
$\alpha = 0.1$	One-shot	1.342	1.376	-0.519	24.850	23.063	528.70
	One-shot-BFGS	1.189	1.238	-0.915	24.601	22.638	527.81
	BFGS	1.244	1.497	-2.741	23.524	19.831	503.50
	L-BFGS	0.752	0.816	-1.570	24.410	22.262	527.03
	L-BFGS-B	0.752	0.816	-1.570	24.410	22.262	527.03
$\alpha = 0.01$	One-shot	0.237	0.251	0.030	28.961	24.997	534.91
	One-shot-BFGS	0.184	0.286	0.079	32.577	23.732	531.10
	BFGS	2.277	2.899	-5.176	30.650	14.803	503.26
	L-BFGS	fails					
	L-BFGS-B	1.130	1.4351	-2.490	26.983	19.662	502.024
$\alpha = 0.001$	One-shot	1.280	1.652	-0.606	43.710	18.728	520.64
	One-shot-BFGS	0.423	0.764	-0.041	46.850	19.847	521.19
	BFGS	3.027	3.805	-6.832	32.289	12.999	503.93
	L-BFGS	3.023	3.800	-6.825	32.349	13.007	503.94
	L-BFGS-B	1.042	1.559	-3.000	45.568	16.741	507.95

[11] Griewank, A. and Kressner, D., ”Time-lag in Derivative Convergence for Fixed Point Iterations. ARIMA Numéro spécial CARI’04, 87–102 (2005).

[12] Giering, R., Kaminski, T., and Slawig, T., Generating Efficient Derivative Code with TAF: Adjoint and Tangent Linear Euler Flow Around an Airfoil. *Future Generation Computer Systems*, 21(8), 1345–1355 (2005).

[13] Bischof, C. H., Lang, B., and Vehreschild, A., Automatic Differentiation for MATLAB Programs. *Proceedings in Applied Mathematics and Mechanics*, 2(1), 50–53 (2003).

[14] Griewank, A., Juedes, D., and Utke, J., Algorithm 755: ADOL-C: A Package for the Automatic Differentiation of Algorithms Written in C/C++. *ACM Transactions on Mathematical Software*, 22(2), 131–167 (1996).

[15] Zickfeld, K., Slawig, T., and Rahmstorf, S., A low-order model for the response of the Atlantic thermohaline circulation to climate change. *Ocean Dynamics*, 54, 8–26 (2004).

[16] Titz, S., Kuhlbrodt, T., Rahmstorf, S., and Feudel, U., On freshwater-dependent bifurcations in box models of the interhemispheric thermohaline circulation. *Tellus A*, 54, 89 – 98 (2002).

[17] Rahmstorf, S., Brovkin, V., Claussen, M., and Kubatzki, C., CLIMBER-2: A climate system model of intermediate complexity. Part II: Model sensitivity. *Clim. Dyn.*, 17, 735–751 (2001).

[18] Zhu, C., Byrd, R. H., and Nocedal, J., L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization. *ACM Transactions on Mathematical Software*, 23(4), 550–560 (1997).

Acknowledgments

The authors are grateful to the DFG for financial support within the Priority Program SPP1253 *Optimization with partial differential equations.* They would like to thank the reviewers for their valuable comments that helped to improve the manuscript.

Claudia Kratzenstein *is a PhD student at Christian-Albrechts-University Kiel, Germany. She received her degree (Diplom) in mathematics at Humboldt University Berlin, Germany, in the field of non-linear optimisation. Among her research interests are non-linear optimisation methods, especially quasi-Newton-strategies, automatic differentiation, and their application in climate modeling.*

Thomas Slawig *is a Professor for Algorithmic Optimal Control – Oceanic CO_2 Uptake at Christian-Albrechts-University Kiel, Germany. He received his PhD in Applied Mathematics in 1998 from TU Berlin with a thesis on Shape Optimization for Navier-Stokes Equations. Since 1999 he is working in the field of parameter identification, sensitivity analysis and numerics for climate models. His research interests are numerical mathematics and optimisation, automatic differentiation, and optimal control.*