International Series of Numerical Mathematics, Vol. 160, 85–108
© 2011 Springer Basel AG

Automated Extension of Fixed Point PDE Solvers for Optimal Design with Bounded Retardation

Nicolas Gauger, Andreas Griewank, Thomas Slawig, Adel Hamdi, Claudia Kratzenstein and Emre Özkaya

Abstract. We study PDE-constrained optimization problems where the state equation is solved by a pseudo-time stepping or fixed point iteration. We present a technique that improves primal, dual feasibility and optimality simultaneously in each iteration step, thus coupling state and adjoint iteration and control/design update. Our goal is to obtain bounded retardation of this coupled iteration compared to the original one for the state, since the latter in many cases has only a Q-factor close to one. For this purpose and based on a doubly augmented Lagrangian, which can be shown to be an exact penalty function, we discuss in detail the choice of an appropriate control or design space preconditioner, discuss implementation issues and present a convergence analysis. We show numerical examples, among them applications from shape design in fluid mechanics and parameter optimization in a climate model.

Mathematics Subject Classification (2000). Primary 90C30; Secondary 99Z99.

Keywords. Nonlinear optimization, fixed point solvers, exact penalty function, augmented Lagrangian, automatic differentiation, aerodynamic shape optimization, parameter optimization, climate model.

1. Introduction

Design optimization or control problems with PDEs may be distinguished from general nonlinear programming problems by the fact that the vector of variables is partitioned into a state vector $y \in Y$ and control or design variables $u \in \mathcal{A} \subset U$. For applications of this scenario in Computational Fluid Dynamics (CFD) see for example [22, 27, 28]. In this paper, we aim to solve an optimization problem

$$\min_{y,u} f(y,u) \quad \text{s.t.} \quad c(y,u) = 0, \tag{1.1}$$

where the constraint or state equation is solved by an iterative process. Usually, the admissible set $\mathcal{A}$ is a closed convex subset of the design space U. For simplicity, we assume that Y and U are Hilbert spaces. When Y and U have finite dimensions $n = \dim(Y)$ and $m = \dim(U)$, their elements may be identified by coordinate vectors in $\mathbb{R}^n$ and $\mathbb{R}^m$ with respect to suitable Hilbert bases. This convention allows us to write duals as transposed vectors and inner products as the usual scalar products in Euclidean space.

The problem of augmenting fixed point solvers for PDEs with sensitivity and optimization has been considered by various authors during the last few years [13, 17, 19, 18, 11, 24]. In [17], the author studied a *One-Shot* approach involving preconditioned design corrections to solve design optimization problems. It is an approach that aims at attaining feasibility and optimality simultaneously (see [17, 11]). In fact, within one step the primal, dual and design variables are updated simultaneously. Using automatic differentiation [14, 16] this requires only one simultaneous evaluation of the function value with one directional and one adjoint derivative. The focus in [17] was about the derivation of an “ideal” preconditioner that ensures local contractivity of the three coupled iterative processes. From analyzing the eigenvalues of the associated Jacobian, the author derived necessary but not sufficient conditions to bound those eigenvalues below 1 in modulus.

Deriving a preconditioner that ensures even local convergence of the three coupled iteration seems to be quite difficult. Instead, we study in this paper the introduction of an exact penalty function of doubly augmented Lagrangian type (see [6, 7, 8, 9]) to coordinate the three iterative processes. This penalty function is defined from the Lagrangian of the optimization problem augmented by weighted primal and dual residuals. The approach should be useful for any combination and sequencing of steps for improving primal, dual feasibility and optimality. In this work we firstly analyze the dual retardation behavior that means the slow down in the overall convergence when the adjoint is coupled with the primal iteration. Section 3 is devoted to derive conditions on the involved weighting coefficients in view of a consistent reduction of the considered exact penalty function. In Section 4, we establish reasonable choices for the weighting coefficients. Then we elaborate a line search procedure that does not require the computation of any second derivatives of the original problem and propose a suitable preconditioner in Section 5. We show a global convergence result in Section 6 and present three examples in Sections 7–9.

1.1. Problem statement

We suppose that the constraint $c(y,u) = 0$ is solved by a fixed point iteration and can be thus be equivalently written as $y = G(y,u)$. We assume that f and G are $C^{2,1}$ functions on the closed convex set $Y \times \mathcal{A}$, and that the Jacobian $G_y := \partial G/\partial y$ has a spectral radius $\rho < 1$. Then, for a suitable inner product norm $\|.\|$, we have

$$\|G_y(y,u)\| \ = \ \|G_y^\top(y,u)\| \quad \leq \quad \rho \ < \ 1. \tag{1.2}$$

Hence, the mean value theorem implies on any convex subdomain of Y that G is a contraction. By 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 Lagrangian associated to the constrained optimization problem is

$$L(y, \bar{y}, u) \;=\; f(y,u) + (G(y,u) - y)^\top \bar{y} \;=\; N(y,\bar{y},u) - y^\top \bar{y},$$

where we introduced is the shifted Lagrangian N as

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

Furthermore, according to the first-order necessary condition [1], a KKT point $(y^*, \bar{y}^*, u^*)$ of the optimization problem (1.1) must satisfy

$$\left.\begin{array}{rcl} y^* &=& G(y^*,u^*) \\ \bar{y}^* &=& N_y(y^*,\bar{y}^*,u^*)^\top \;=\; f_y(y^*,u^*)^\top + G_y(y^*,u^*)^\top \bar{y}^* \\ 0 &=& N_u(y^*,\bar{y}^*,u^*)^\top \;=\; f_u(y^*,u^*)^\top + G_u(y^*,u^*)^\top \bar{y}^*. \end{array}\right\} \tag{1.3}$$

Denoting by $\mathcal{F} := \{z = (y,u) \in Y \times \mathcal{A} : y = G(y,u)\}$ the feasible set, any z in the tangent plane $\mathcal{T}$ can be represented by the Implicit Function Theorem as $z = Z\tilde{z}$ where $\tilde{z} \in \mathbb{R}^m$ and

$$Z = \begin{bmatrix} (I - G_y)^{-1} G_u \\ I \end{bmatrix}.$$

In view of (1.2), we have that $I - G_y$ is invertible. Therefore, $\mathcal{F}$ is a smooth manifold of dimension $\dim(u) = m$ with tangent space spanned by the columns of Z. According to the second-order necessary condition, the reduced Hessian

$$H = Z^\top N_{xx} Z \qquad \text{where} \quad N_{xx} = \begin{bmatrix} N_{yy} & N_{yu} \\ N_{uy} & N_{uu} \end{bmatrix}, \tag{1.4}$$

must be positive semi-definite at a local minimizer. We will make the slightly stronger assumption of second-order sufficiency, i.e., H is positive definite.

1.2. One-shot strategy

Motivated by (1.3), one can use the following coupled full step iteration, called *One-shot strategy*, to reach a KKT point (see [17, 11]):

$$\left.\begin{array}{rcl} y_{k+1} &=& G(y_k,u_k), \\ \bar{y}_{k+1} &=& N_y(y_k,\bar{y}_k,u_k)^\top, \\ u_{k+1} &=& u_k - B_k^{-1} N_u(y_k,\bar{y}_k,u_k)^\top. \end{array}\right\} \tag{1.5}$$

Here, B_k is a design space preconditioner which must be selected to be symmetric positive definite. The contractivity (1.2) ensures that the first equation in the coupled full step (1.5) converges ρ-linearly for fixed u. Although the second equation exhibits a certain time-lag, it converges with the same asymptotic R-factor (see [19]). As far as the convergence of the coupled iteration (1.5) is concerned, the goal is to find B_k that ensures that the spectral radius of the coupled iteration (1.5) stays below 1 and as close as possible to ρ. We use the following notations:

$$\begin{aligned}\Delta y_k &:= G(y_k, u_k) - y_k,\\ \Delta \bar{y}_k &:= N_y(y_k, \bar{y}_k, u_k)^\top - \bar{y}_k,\\ \Delta u_k &:= -B_k^{-1} N_u(y_k, \bar{y}_k, u_k)^\top.\end{aligned}$$

2. Dual retardation

Since the solution $\bar{y}^*$ of the second equation introduced in (1.3) depends on the primal solution y^*, it cannot be computed accurately as long as y^* is not known, and the dual iterates $\bar{y}_k$ computed from the second equation will be affected by the remaining inaccuracy in the primal iterates y_k. Actually, the dual step corrections typically lag a little bit behind as the perturbations caused by the errors in the primal iterates tend to accumulate initially. We refer to this delay of convergence relative to the primal iterates as dual retardation. Nevertheless, asymptotically the dual correction steps tend to be no larger than the primal correction steps.

Theorem 2.1. *For u fixed let f, G be once Lipschitz continuously differentiable with respect to y near the fixed point $y^* = y^*(u)$ of G, and let $y_k \to y^*$ such that*

$$\lim_{k\to\infty} \frac{\|\Delta y_k\|}{\|\Delta y_{k-1}\|} = \rho^* := \|G_y(y^*, u)\|.$$

Now define for any $\varepsilon, \alpha, \beta > 0$ the smallest pair of integers $(\ell_p^\varepsilon, \ell_d^\varepsilon)$ such that

$$\sqrt{\alpha}\|\Delta y_{\ell_p^\varepsilon}\| \le \varepsilon \quad \text{and} \quad \sqrt{\beta}\|\Delta \bar{y}_{\ell_d^\varepsilon}\| \le \varepsilon.$$

Then, we have

$$\limsup_{\varepsilon\to 0} \frac{\ell_d^\varepsilon}{\ell_p^\varepsilon} \le 1. \tag{2.1}$$

Proof. See [21, Theorem 2.1]. □

One can show that this result is sharp considering the scalar case with the cost function

$$f(y) = \frac{\theta}{2} y^2 + \nu y, \quad y \in \mathbb{R},\ \nu \in \mathbb{R},\ \theta \in \mathbb{R}_+^*,$$

and $G(y, u) = \rho y + u$ with $0 < \rho < 1$. The coupled primal and dual iteration is

$$\begin{aligned} y_{k+1} &= \rho y_k + u,\\ \bar{y}_{k+1} &= \rho \bar{y}_k + \theta y_k + \nu.\end{aligned}$$

Then, we get the limit in (2.1) is 1. For details see again [21].

As we will see later, a more or less rather natural choice for the weights α, β is $\sqrt{\beta/\alpha} = (1-\rho)/\theta$. If we use $|\Delta y_0| = 1$, $\sqrt{\beta/\alpha} = (1-\rho)/\theta$, we can show that in this example $|\Delta y_k| = \rho^k$ and $\sqrt{\beta/\alpha}|\Delta \bar{y}_k| = k(1-\rho)\rho^{k-1}$.

3. Doubly augmented Lagrangian

The asymptotic rate of convergence of the coupled iteration (1.5) to a limit point $(y^*, \bar{y}^*, u^*)$ is determined by the spectral radius $\hat{\rho}(J^*)$ of the block Jacobian:

$$J^* = \left.\frac{\partial(y_{k+1}, \bar{y}_{k+1}, u_{k+1})}{\partial(y_k, \bar{y}_k, u_k)}\right|_{(y^*,\bar{y}^*,u^*)} = \begin{bmatrix} G_y & 0 & G_u \\ N_{yy} & G_y^\top & N_{yu} \\ -B^{-1}N_{uy} & -B^{-1}G_u^\top & I - B^{-1}N_{uu} \end{bmatrix}.$$

In [17], the author proved that unless they happen to coincide with those of G_y, the eigenvalues of J^* solve the following nonlinear eigenvalues problem:

$$\det[(\lambda - 1)B + H(\lambda)] = 0,$$

where

$$H(\lambda) = Z(\lambda)^\top N_{xx} Z(\lambda) \quad \text{and} \quad Z(\lambda) = \begin{bmatrix} (\lambda I - G_y)^{-1} G_u \\ I \end{bmatrix}.$$

As discussed in [17], although the conditions $B = B^\top \succ 0$ and $B \succ \frac{1}{2}H(-1)$ ensure that real eigenvalues of J^* stay less than 1, they are just necessary but not sufficient to exclude real eigenvalues less than -1. In addition, no constructive condition to also bound complex eigenvalues below 1 in modulus has been found. However, these do arise even when the underlying primal solver is Jacobi's method on the elliptic boundary value problem $y'' = 0$ in one space dimension. Therefore, deriving a design space preconditioner that ensures even local convergence of the coupled full step iteration (1.5) seems to be quite difficult.

Instead, we base our analysis on the following penalty or merit function of doubly augmented Lagrangian type (see [6, 7]), defined for $\alpha, \beta > 0$:

$$L^a(y, \bar{y}, u) := \frac{\alpha}{2}\|G(y,u) - y\|^2 + \frac{\beta}{2}\|N_y(y, \bar{y}, u)^\top - \bar{y}\|^2 + N(y, \bar{y}, u) - \bar{y}^\top y. \quad (3.1)$$

Now, we aim to solve the optimization problem (1.1) by looking for descent on L^a.

3.1. Gradient of L^a

In the remainder, we use the notation $\Delta G_y = I - G_y$. Note that ΔG_y is invertible. By an elementary calculation, we obtain:

Proposition 3.1. *The gradient of L^a is given by*

$$\nabla L^a(y, \bar{y}, u) = -Ms(y, \bar{y}, u)\ , \quad \textit{where} \quad M = \begin{bmatrix} \alpha \Delta G_y^\top & -I - \beta N_{yy} & 0 \\ -I & \beta \Delta G_y & 0 \\ -\alpha G_u^\top & -\beta N_{yu}^\top & B \end{bmatrix}, \quad (3.2)$$

and s is the step increment vector associated with the iteration (1.5)

$$s(y, \bar{y}, u) = \begin{bmatrix} G(y,u) - y \\ N_y(y, \bar{y}, u)^\top - \bar{y} \\ -B^{-1}N_u(y, \bar{y}, u)^\top \end{bmatrix}. \quad (3.3)$$

The gradient ∇L^a involves vector derivatives as well as matrix derivatives where the complexity of their computations may grow with respect to the dimension of u. To avoid that dependence, we propose an economical computation of ∇L^a using Automatic Differentiation (AD), see [16]. Actually, to compute vector derivatives we can use the *reverse mode* of the package ADOL-C developed at Dresden University of Technology [15]. Furthermore, we present two options to compute terms in ∇L^a involving matrix derivatives namely $\Delta\bar{y}^\top N_{yy}$ and $\Delta\bar{y}^\top N_{yu}$. The first option consists on using one reverse sweep of *Second Order Adjoint* (SOA) by employing some (AD) tools, like ADOL-C [16] that ensures a cost proportional to the cost of (f, G) evaluation and independent of dimensions. Whereas the second option consists on simply using the definition

$$\frac{\partial}{\partial t}(N_y(y + t\Delta\bar{y}, \bar{y}, u))\bigg|_{t=0} = N_{yy}(y, \bar{y}, u)\Delta\bar{y},$$

to approximate $\Delta\bar{y}^\top N_{yy}$. In fact, for $t \neq 0$, we have

$$\Delta\bar{y}^\top N_{yy}(y, \bar{y}, u) = \frac{N_y(y + t\Delta\bar{y}, \bar{y}, u)^\top - N_y(y, \bar{y}, u)^\top}{t} + o(t), \tag{3.4}$$

and thus the $\Delta\bar{y}^\top N_{yy}$ (and similarly $\Delta\bar{y}^\top N_{yu}$) can be approximated using (3.4).

4. Conditions on the weights α, β

Here we derive conditions on the weights α, β which in turn influence the merit function L^a to be an exact penalty function and the step increment vector associated to iteration (1.5) to yield descent on it.

4.1. Correspondence conditions

The first condition characterizes the correspondence between stationary points of L^a and zero increments s of the one-shot iteration, i.e., stationary points of 85.

Corollary 4.1 (Correspondence condition). *There is a one-to-one correspondence between the stationary points of L^a and the roots of s, defined in* (3.3)*, wherever*

$$\det[\alpha\beta\Delta G_y^\top \Delta G_y - I - \beta N_{yy}] \quad \neq \quad 0,$$

which is implied by the correspondence condition

$$\alpha\beta(1-\rho)^2 \quad > \quad 1 + \beta\theta, \tag{4.1}$$

where $\theta = \|N_{yy}\|$.

Proof. See [21, Corollary 3.2]. □

The correspondence condition (4.1) now implies that the merit function L^a introduced in (3.1) is an exact penalty function:

Corollary 4.2. *If the condition*

$$\alpha\beta\Delta G_y^\top \Delta G_y > I + \beta N_{yy},$$

holds, then the penalty function L^a introduced in (3.1) *has a positive definite Hessian at a stationary point of the optimization problem* (1.1) *if and only if the reduced Hessian H introduced in* (1.4) *is positive definite at that point.*

Proof. See [21, Corollary 3.3]. □

4.2. Descent properties of the step increment

Here we derive conditions under which the step increment vector s introduced in (3.3) yields descent on the exact penalty function L^a.

Proposition 4.3 (Descent condition). *The step increment vector s yields descent on L^a for all large positive B if*

$$\alpha\beta\Delta\bar{G}_y \quad > \quad \left(I + \frac{\beta}{2}N_{yy}\right)(\Delta\bar{G}_y)^{-1}\left(I + \frac{\beta}{2}N_{yy}\right), \tag{4.2}$$

where $\Delta\bar{G}_y = \frac{1}{2}(\Delta G_y + \Delta G_y^\top)$. Moreover, (4.2) *is implied by the condition*

$$\sqrt{\alpha\beta}(1-\rho) \quad > \quad 1 + \frac{\beta}{2}\theta. \tag{4.3}$$

Proof. See [21, Proposition 3.4]. □

The design corrections given by the third component in s involve the inverse of B. Thus, provided (4.3) holds, a pure feasibility step (with fixed design) yields also descent on L^a. Considering a base point

$$(y_{k-1}, \bar{y}_{k-1}, u)$$

where $k \geq 1$ and analyzing L^a at the current point

$$(y_k = G(y_{k-1}, u), \bar{y}_k = N_y(y_{k-1}, \bar{y}_{k-1}, u), u),$$

we can establish a condition that leads to reduction on the exact penalty function L^a using a full pure feasibility step.

Theorem 4.4 (Full step descent condition). *Let $\alpha, \beta > 0$ satisfy*

$$\alpha \quad > \quad \frac{\theta(\beta\theta + 2)}{1-\rho^2} + \frac{(5 + \rho(1+\beta\theta))^2}{\beta(1-\rho^2)^2}. \tag{4.4}$$

Then a full pure feasibility step yields descent in L^a, i.e.,

$$L^a(y_k, \bar{y}_k, u) - L^a(y_{k-1}, \bar{y}_{k-1}, u) \quad < \quad 0.$$

Proof. See [21, Theorem 3.6] □

Note that the descent condition (4.3) is a bit stronger than the correspondence condition (4.1). However, the condition (4.4) is stronger than (4.3).

4.3. Bounded level sets of L^a

In order to establish a global convergence result, we show that under reasonable assumptions all level sets of the doubly augmented Lagrangian L^a are bounded.

Theorem 4.5. *Let $f \in C^{1,1}(Y \times U)$ be radially unbounded and satisfy*

$$\liminf_{\|y\|+\|u\|\to\infty} \frac{f}{\|\nabla_y f\|^2} \quad > \quad 0. \tag{4.5}$$

Then there exists (α, β) fulfilling (4.3) such that

$$\lim_{\|y\|+\|\bar{y}\|+\|u\|\to\infty} L^a(y, \bar{y}, u) \quad = \quad +\infty. \tag{4.6}$$

If the limit in (4.5) is equal to infinity, the assertion (4.6) holds without any additional restriction on (α, β).

Proof. See [20, Theorem 2.1]. □

Assumption (4.5) requires that f grows quadratically or slower as a function of $\|y\| + \|u\|$. If for example f is quadratic, i.e.,

$$f(x) = \frac{1}{2}x^\top Ax + b^\top x \qquad \text{where} \quad A \in \mathbb{R}^{n,n}, A^\top = A \succ 0, \quad b, x \in \mathbb{R}^n,$$

we have (with $\lambda_{\min}, \lambda_{\max}$ the smallest and biggest eigenvalue, respectively, of A)

$$\lim_{\|x\|\to\infty} \frac{f(x)}{\|\nabla f(x)\|^2} \quad = \quad \frac{1}{2} \lim_{\|x\|\to\infty} \frac{x^\top Ax}{\|Ax\|^2} \geq \frac{\lambda_{\min}}{2\|A\|_2^2} = \frac{\lambda_{\min}}{2\lambda_{\max}^2} > 0.$$

4.4. Particular choice of α and β

To ensure consistent reduction on L^a, a rather large primal weight α may be necessary, which severely restricts the step size and slows down the convergence. We present two options to compute the α, β based on selecting α as small as possible which is still fulfilling at least the condition (4.3) or (4.4) as an equality.

Option 1: Deriving α, β from (4.3) and minimizing α as a function of β leads to

$$\beta_1 = \frac{2}{\theta} \quad \text{and} \quad \alpha_1 = \frac{2\theta}{(1-\rho)^2}. \tag{4.7}$$

Option 2: Deriving α, β from the condition (4.4):
Fulfilling (4.4) as an equality implies

$$\alpha = \frac{\theta^2\beta^2 + 2\theta(1+5\rho)\beta + 5(5+2\rho) + \rho^2}{\beta(1-\rho^2)^2}. \tag{4.8}$$

Minimizing the right-hand side with respect to β gives

$$\beta_2 = \frac{\sqrt{5(5+2\rho)+\rho^2}}{\theta} \quad \text{and} \quad \alpha_2 = \frac{2\theta(1+5\rho+\theta\beta_2)}{(1-\rho^2)^2}. \tag{4.9}$$

If as usually $\rho \approx 1$, we obtain

$$\beta_2 \approx \frac{6}{\theta} \quad \text{and} \quad \alpha_2 \approx \frac{24\theta}{(1+\rho)^2(1-\rho)^2} \approx \frac{6\theta}{(1-\rho)^2}. \tag{4.10}$$

4.5. Estimation of problem parameters

To compute the weights we need estimates for ρ and θ. Neglecting the change in u, they can be obtained from the primal and dual iterates. From (1.2) we have

$$\|G(y_k,u)-G(y_{k-1},u)\| \le \rho\|y_k-y_{k-1}\| \quad \text{for all } k\in\mathbb{N}.$$

Therefore, starting with an initial value ρ_0, we may update ρ using

$$\rho_{k+1} = \max\{\frac{\|\Delta y_k\|}{\|\Delta y_{k-1}\|}, \tau\rho_k\},$$

where $\tau\in(0,1)$. To estimate the value of θ, we use the approximation

$$\begin{bmatrix}\Delta y_{k+1}\\ \Delta\bar y_{k+1}\end{bmatrix} \approx \begin{bmatrix} G_y(y_k,u_k) & 0\\ N_{yy}(y_k,\bar y_k,u_k) & G_y(y_k,u_k)^\top\end{bmatrix}\begin{bmatrix}\Delta y_k\\ \Delta\bar y_k\end{bmatrix}.$$

Then, we obtain

$$\begin{bmatrix}\Delta\bar y_k\\ -\Delta y_k\end{bmatrix}^\top \begin{bmatrix}\Delta y_{k+1}\\ \Delta\bar y_{k+1}\end{bmatrix} \approx -(\Delta y_k)^\top N_{yy}(y_k,\bar y_k,u_k)\Delta y_k,$$

and

$$(\Delta y_k)^\top N_{yy}(y_k,\bar y_k,u_k)\Delta y_k \approx (\Delta y_k)^\top\Delta\bar y_{k+1} - (\Delta\bar y_k)^\top\Delta y_{k+1}.$$

Thus, one can approximate the value of θ as follows:

$$\theta_{k+1}\approx \max\{\frac{|(\Delta y_k)^\top\Delta\bar y_{k+1} - (\Delta\bar y_k)^\top\Delta y_{k+1}|}{\|\Delta y_k\|^2}, \tau\theta_k\}.$$

As far as numerical experiments are concerned, we obtained the best results when $N_{yy}=I$ which theoretically can be attained by using a coordinate transformation provided that $N_{yy}\succ 0$. In fact, let $N_{yy}=N_{yy}^{\frac{\top}{2}}\,N_{yy}^{\frac12}$ be the Cholesky factorization of N_{yy}. Then, by considering $\tilde y = N_{yy}^{\frac12}\,y$ and $\tilde{\bar y} = N_{yy}^{-\frac{\top}{2}}\,\bar y$ we obtain

$$\tilde L^a(\tilde y,\tilde{\bar y},u) = \frac{\alpha}{2}\|\tilde G(\tilde y,u)-\tilde y\|^2 + \frac{\beta}{2}\|\tilde N_{\tilde y}(\tilde y,\tilde{\bar y},u)-\tilde{\bar y}\|^2 + \tilde N(\tilde y,\tilde{\bar y},u) - \tilde{\bar y}^\top\tilde y.$$

where $\tilde G(\tilde y,u) = N_{yy}^{\frac12}G(N_{yy}^{-\frac12}\tilde y,u)$, $\tilde f(\tilde y,u)=f(N_{yy}^{-\frac12}\tilde y,u)$ and $\tilde N_{\tilde y}(\tilde y,\tilde{\bar y},u)=\tilde f(\tilde y,u)+\tilde G(\tilde y,u)^\top\tilde{\bar y}$. Thus, we get $\nabla\tilde L^a(\tilde y,\tilde{\bar y},u) = -\tilde M\tilde s(\tilde y,\tilde{\bar y},u)$ where

$$\tilde M = \begin{bmatrix}\alpha\Delta\tilde G_{\tilde y}^\top & -(1+\beta)I & 0\\ -I & \beta\Delta\tilde G_{\tilde y} & 0\\ -\alpha\tilde G_u^\top & -\beta\tilde N_{\tilde yu}^\top & B\end{bmatrix}, \quad \tilde s(\tilde y,\tilde{\bar y},u) = \begin{bmatrix}\tilde G(\tilde y,u)-\tilde y\\ \tilde N_{\tilde y}(\tilde y,\tilde{\bar y},u)^\top - \tilde{\bar y}\\ -B^{-1}\tilde N_u(\tilde y,\tilde{\bar y},u)^\top\end{bmatrix}.$$

With $\Delta\tilde G_{\tilde y} = I - N_{yy}^{\frac12}\,G_y\,N_{yy}^{-\frac12}$, $\tilde G_u = N_{yy}^{\frac12}\,G_u$ and $\tilde N_{\tilde yu} = N_{yy}^{-\frac{\top}{2}}\,N_{yu}$. Furthermore, $\rho(\tilde G_{\tilde y}) = \rho(G_y) = \rho$. Therefore, if we have a state space preconditioner $P\approx N_{yy}^{\frac12}$ for which P and P^{-1} can be evaluated at reasonable cost, then we may use

$$\tilde L^a(y,\bar y,u) = \frac{\alpha}{2}\|P(G(y,u)-y)\|^2 + \frac{\beta}{2}\|P^{-\top}(N_y(y,\bar y,u)-\bar y)\|^2 + N(y,\bar y,u) - \bar y^\top y.$$

Hence, by the same techniques used to prove Proposition 4.3, we obtain the descent condition associated to the transformed coordinates given by

$$\sqrt{\alpha\beta}(1-\rho) > 1 + \frac{\beta\tilde{\theta}}{2} \quad \text{where} \quad \tilde{\theta} = \|P^{-1}N_{yy}^{\frac{1}{2}}\|$$

and the following approximations:

$$\beta_1 = \frac{2}{\tilde{\theta}},\ \alpha_1 = \frac{2\tilde{\theta}}{(1-\rho)^2} \quad \text{and} \quad \beta_2 = \frac{6}{\tilde{\theta}},\ \alpha_2 = \frac{6\tilde{\theta}}{(1-\rho)^2}.$$

5. Choice of the preconditioner B

Here, we derive a design space preconditioner B that results in a step s that yields descent on L^a. We assume that α, β are chosen such that (4.1) holds.

5.1. Explicit condition on B

In this subsection, we derive an explicit condition that leads to a first choice for the design space preconditioner. Since $s^\top Ms = \frac{1}{2}s^\top(M+M^\top)s$ and using (3.2), (3.3), we compute the symmetric matrix M_S defined as follows:

$$M_S := \frac{1}{2}(M^\top + M) = \begin{bmatrix} \alpha\Delta\bar{G}_y & -I-\frac{\beta}{2}N_{yy} & -\frac{\alpha}{2}G_u \\ -I-\frac{\beta}{2}N_{yy} & \beta\Delta\bar{G}_y & -\frac{\beta}{2}N_{yu} \\ -\frac{\alpha}{2}G_u^\top & -\frac{\beta}{2}N_{yu}^\top & B \end{bmatrix}. \tag{5.1}$$

Here $\Delta\bar{G}_y$ is the symmetric matrix given in Proposition 4.3. Therefore, we obtain

$$s^\top \nabla L^a = -s^\top M_S s. \tag{5.2}$$

Let $B^{\frac{1}{2}}$ be a Cholesky factor of B. Rescaling $u = B^{-\frac{1}{2}}\tilde{u}$, we find a result similar to (5.1) involving $\tilde{s}$ and $\tilde{M}_S$ where $\tilde{s}$ is obtained from the increment vector s by replacing its third component $\Delta u = -B^{-1}{N_u}^\top$ by

$$\Delta\tilde{u} = B^{\frac{1}{2}}\Delta u = -B^{-\frac{\top}{2}}{N_u}^\top = -{N_{\tilde{u}}}^\top$$

and the matrix $\tilde{M}_S$ is derived from M_S by substituting B with I and all derivatives with respect to the design u with $G_{\tilde{u}} = G_u B^{-\frac{1}{2}}$, $N_{\tilde{u}} = N_u B^{-\frac{1}{2}}$ and $N_{y\tilde{u}} = N_{yu}B^{-\frac{1}{2}}$. Thus, we get

$$\tilde{M}_S = \begin{bmatrix} \alpha\Delta\bar{G}_y & -I-\frac{\beta}{2}N_{yy} & -\frac{\alpha}{2}G_{\tilde{u}} \\ -I-\frac{\beta}{2}N_{yy} & \beta\Delta\bar{G}_y & -\frac{\beta}{2}N_{y\tilde{u}} \\ -\frac{\alpha}{2}G_{\tilde{u}}^\top & -\frac{\beta}{2}N_{y\tilde{u}}^\top & I \end{bmatrix}.$$

and $\tilde{M}_S$ is obtained from the matrix M_S as follows:

$$\tilde{M}_S = \operatorname{diag}(I, I, B^{-\frac{\top}{2}})\ M_S\ \operatorname{diag}(I, I, B^{-\frac{1}{2}}). \tag{5.3}$$

The aim now is to derive explicit conditions on B that ensure positive definiteness of $\tilde{M}_S$ which in view of (5.2), (5.3) implies that the increment vector s introduced in (3.3) yields descent on L^a. In fact it suffices to show positive-definiteness of a much simpler, real matrix in order to get the desired result for $\tilde{M}_S$:

Proposition 5.1. *Let $\theta = \|N_{yy}\|$ and D_C be the matrix defined by*

$$D_C = \begin{bmatrix} \alpha(1-\rho) & -1-\frac{\beta}{2}\theta & -\frac{\alpha}{2}\|G_{\tilde{u}}\| \\ -1-\frac{\beta}{2}\theta & \beta(1-\rho) & -\frac{\beta}{2}\|N_{y\tilde{u}}\| \\ -\frac{\alpha}{2}\|G_{\tilde{u}}\| & -\frac{\beta}{2}\|N_{y\tilde{u}}\| & 1 \end{bmatrix}. \tag{5.4}$$

Then, we have for all $v_1, v_2 \in \mathbb{R}^n$ and $v_3 \in \mathbb{R}^m$,

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}^\top \tilde{M}_S \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \geq \begin{bmatrix} \|v_1\| \\ \|v_2\| \\ \|v_3\| \end{bmatrix}^\top D_C \begin{bmatrix} \|v_1\| \\ \|v_2\| \\ \|v_3\| \end{bmatrix}.$$

Proof. See [20, Proposition 3.1]. □

Now we get a condition on B that ensures positive definiteness of D_C:

Proposition 5.2. *Let $\theta = \|N_{yy}\|$ and α, β satisfy* (4.3). *If*

$$\left(\frac{\sqrt{\alpha}}{2}\|G_{\tilde{u}}\| + \frac{\sqrt{\beta}}{2}\|N_{y\tilde{u}}\|\right)^2 \leq (1-\rho) - \frac{(1+\frac{\theta}{2}\beta)^2}{\alpha\beta(1-\rho)}, \tag{5.5}$$

then D_C introduced in (5.4) *is a positive definite matrix.*

Proof. See [20, Proposition 3.2] □

To get explicit conditions on B that ensure (5.5), we note that

$$\frac{1}{2}(\sqrt{\alpha}\|G_{\tilde{u}}\| + \sqrt{\beta}\|N_{y\tilde{u}}\|) \leq \left\| \begin{matrix} \sqrt{\alpha}G_{\tilde{u}} \\ \sqrt{\beta}N_{y\tilde{u}} \end{matrix} \right\|_2 = \left\| \begin{pmatrix} \sqrt{\alpha}G_u \\ \sqrt{\beta}N_{yu} \end{pmatrix} B^{-\frac{1}{2}} \right\|_2, \tag{5.6}$$

and, using a QR decomposition on the right-hand side obtain

$$\left\| \begin{pmatrix} \sqrt{\alpha}G_u \\ \sqrt{\beta}N_{yu} \end{pmatrix} B^{-\frac{1}{2}} \right\|_2^2 = \|RB^{-\frac{1}{2}}\|_2^2 = \|RB^{-1}R^\top\|_2.$$

As design corrections involve B^{-1}, the aim is to chose it as large as possible. The largest B_0^{-1} for which $\|RB^{-1}R^\top\|_2$ is equal to some $\sigma > 0$ is $RB_0^{-1}R^\top = \sigma I$, i.e., according to (5.6), all preconditioners B satisfying

$$B = B^\top \succeq B_0 = \frac{1}{\sigma}R^\top R = \frac{1}{\sigma}(\alpha G_u^\top G_u + \beta N_{yu}{}^\top N_{yu}). \tag{5.7}$$

lead to $D_C \succ 0$, and thus to a descent on L^a by the increment vector s. Here, σ must be chosen such that Proposition 5.2 applies, i.e.,

$$\sigma = 1 - \rho - \frac{(1+\frac{\theta}{2}\beta)^2}{\alpha\beta(1-\rho)} > 0. \tag{5.8}$$

5.2. Particular choice of weighting coefficients

Now we define weighting coefficients α, β fulfilling (4.3) and independent of all linear transformation in the design space. If we assume that the rectangular matrix $G_u \in \mathbb{R}^{n,m}$ has full column rank and denote by C a Cholesky factor such that $G_u^\top G_u = C^\top C \succ 0$. Then we can show (see [20, Section 3.2]) that

$$\|C^{-\top} B_0 C^{-1}\| \;\leq\; \varphi(\alpha,\beta) \;:=\; \frac{\alpha + q\beta}{\psi(\alpha,\beta)}, \tag{5.9}$$

where

$$\begin{aligned} \psi(\alpha,\beta) \;&:=\; 1-\rho-\frac{(1+\frac{\theta}{2}\beta)^2}{\alpha\beta(1-\rho)} \;>\; 0, \\ q \;&:=\; \|C^{-\top}N_{yu}^\top N_{yu}C^{-1}\|_2 = \|N_{yu}C^{-\top}\|_2^2 \;=\; \max_{0\neq z\in U}\frac{\|N_{yu}z\|_2^2}{\|G_u^\top z\|_2^2}. \end{aligned} \tag{5.10}$$

The ratio q quantifies the perturbation of the adjoint equation $N_y = 0$ caused by a design variation z relative to that in the primal equation. Since the aim is to maximize B^{-1} in order to make significant design corrections, we define optimal penalty weights α, β which satisfy (4.3) and realize a minimum of the function φ.

Proposition 5.3. *For $q > 0$ the function φ in* (5.9) *reaches its minimum for*

$$\beta = \frac{3}{\sqrt{\theta^2 + 3q(1-\rho)^2} + \frac{\theta}{2}} \quad \textit{and} \quad \alpha = q\frac{\beta(1+\frac{\theta}{2}\beta)}{1-\frac{\theta}{2}\beta}. \tag{5.11}$$

Proof. See [20, Proposition 3.3 and Appendix]. □

For $q = 0$ this directly gives $\beta = 2/\theta$. Combining both equations in (5.11) (with q kept in there) and setting $q = 0$ afterwards gives

$$\alpha = \frac{4\theta}{(1-\rho)^2}. \tag{5.12}$$

5.3. Suitable B and relation to $\nabla_{uu}L^a$

Here, using B_0 derived in (5.7) we define a suitable B and establish its relation to the Hessian of L^a with respect to the design. We consider Δu such that

$$\min_{\Delta u} \; L^a(y+\Delta y, \bar{y}+\Delta\bar{y}, u+\Delta u).$$

Using a quadratic approximation of L^a, assuming $\nabla_{uu}L^a \succ 0$ and $N_{uu} \succeq 0$, we define a suitable design space preconditioner B from (5.7) and (5.8) such that

$$B \;=\; B_0 + \frac{1}{\sigma}N_{uu} \;=\; \frac{1}{\sigma}\left(\alpha G_u^\top G_u + \beta N_{yu}^\top N_{yu} + N_{uu}\right). \tag{5.13}$$

In view of (5.7) the increment vector s obtained using the preconditioner B introduced in (5.13) yields descent on L^a. In addition, we have $B \approx \nabla_{uu}L^a$. This approximation turns to an equality at primal and dual feasibility. Besides, as L^a is an exact penalty function, we have $\nabla^2 L^a \succ 0$ in a neighborhood of a local minimizer and then in particular $\nabla_{uu}L^a = B \succ 0$.

5.4. BFGS update to an approximation of B

As the suitable preconditioner B derived in (5.13) involves matrix derivatives which may be costly evaluated, numerically we use the BFGS method to update its approximation H_k. In view of the relation $B \approx \nabla_{uu} L^a$ established in the previous subsection, we use the following secant equation in the update of H_k:

$$H_{k+1} R_k = \Delta u_k, \quad R_k := \nabla_u L^a(y_k, \bar{y}_k, u_k + \Delta u_k) - \nabla_u L^a(y_k, \bar{y}_k, u_k).$$

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

$$\Delta u_k{}^\top \nabla_u L^a(y_k, \bar{y}_k, u_k + \eta \Delta u_k) \geq c_2 \Delta u_k{}^\top \nabla_u L^a(y_k, \bar{y}_k, u_k) \quad \text{with } c_2 \in [0,1],$$

implies the necessary curvature condition

$$R_k{}^\top \Delta u_k > 0. \tag{5.14}$$

A simpler procedure could skip the update whenever (5.14) does not hold by either setting H_{k+1} to identity or to the last iterate H_k. Provided (5.14) holds, we use

$$H_{k+1} = (I - r_k \Delta u_k R_k{}^\top) H_k (I - r_k R_k \Delta u_k{}^\top) + r_k \Delta u_k \Delta u_k{}^\top \text{ with } r_k = \frac{1}{R_k{}^\top \Delta u_k}.$$

5.5. Alternating approach

Each BFGS update of H_k needs to make a pure design step (step with fixed primal and dual variables) in order to compute the coefficient R_k. The approach presented here aims to achieve minimization of L^a using alternating between pure design and pure feasibility steps. For several applications, design corrections may be costly evaluated especially where each design update implies a modification of the geometry which requires to remesh and update the data structure (see [22, 27]. Thus it could be more convenient to perform only *significant* design corrections. If the suggested change in the design variable u is small, we directly improve feasibility, leaving u unchanged. Actually, we perform a design correction only if

$$\Delta u^\top \nabla_u L^a < 0 \quad \text{and} \quad \tau\, \Delta u^\top \nabla_u L^a < \Delta y^\top \nabla_y L^a + \Delta \bar{y}^\top \nabla_{\bar{y}} L^a, \tag{5.15}$$

where $\tau \in]0,1]$ is a percent which may be fixed by the user. We suppose there exists $\bar{B}$ such that for all iteration k we have

$$B(y, \bar{y}, u) \leq B_k \leq \bar{B} \qquad \text{for all } (y, \bar{y}, u) \in \mathcal{N}_0, \tag{5.16}$$

where $\mathcal{N}_0$ is a level set of L^a. And thus $\|B_k\|$ is finite for all iterations. An algorithm realizing this approach is presented in details in [20, Section 4].

5.6. Line search procedures

With the preconditioner B derived in (5.13), we expect full step convergence near a local minimizer of L^a. To enforce convergence in the earlier stage of iterations, we briefly sketch two backtracking line search procedures based on two slightly different quadratic forms: The first one consists in applying a standard backtracking line search on a quadratic interpolation Q of L^a (see [1, 5]):

$$Q(\eta) = \xi_2 \eta^2 + \xi_1 \eta + \xi_0 \qquad \text{for} \quad \eta \in [0, \eta_c],$$

where

$$\begin{aligned}\xi_0 &= L^a(y_k, \bar{y}_k, u_k), \qquad \xi_1 = \nabla L^a(y_k, \bar{y}_k, u_k)^\top s_k \ < \ 0,\\ \xi_2 &= \frac{1}{{\eta_c}^2}\left(L^a(y_k + \eta_c \Delta y_k, \bar{y}_k + \eta_c \Delta \bar{y}_k, u_k + \eta_c \Delta u_k) - \xi_1 \eta_c - \xi_0\right).\end{aligned}$$

Here, $\xi_1 < 0$ is implied by the fact that the increment vector yields descent on L^a.

The second procedure does not require the computation of ∇L^a which may save calculation cost. Linear interpolations P, D of the primal and dual residuals, respectively, and a standard parabolic interpolation q based on the initial descent and two function values for the unpenalized Lagrangian lead to the approximation

$$\tilde{Q}(\eta) = \frac{\alpha}{2}\|P(\eta)\|_2^2 + \frac{\beta}{2}\|D(\eta)\|_2^2 + q(\eta), \quad \eta \in [0, \eta_c],$$

of L^a. Here η_c is a tentative step size. If η^* denotes the (explicitly computable) stationary point of $\tilde{Q}$ multiplied by the sign of its second-order term, we accept η_c only if $\eta^* \geq \frac{2}{3}\eta_c$ which ensures $\tilde{Q}(\eta_c) < \tilde{Q}(0)$ and thus

$$L^a(y_k + \eta_c \Delta y_k, \bar{y}_k + \eta_c \Delta \bar{y}_k, u_k + \eta_c \Delta u_k) < L^a(y_k, \bar{y}_k, u_k).$$

As long as $\eta^* \geq \frac{2}{3}\eta_c$ is violated, we set

$$\eta_c = \operatorname{sign}(\eta^*) \max\{0.2|\eta_c|, \min\{0.8|\eta_c|, |\eta^*|\}\}$$

and recompute η^*. For the acceptance of the initial step multiplier $\eta_c = 1$, we also require $\eta^* \leq \frac{4}{3}\eta_c$. Failing this, η_c is once increased to $\eta_c = \eta^*$ and then always reduced until the main condition $\eta^* \geq \frac{2}{3}\eta_c$ is fulfilled. In both cases, $\frac{\eta^*}{\eta_c} \geq \frac{2}{3}$ is fulfilled after a finite number of steps and the line search procedure terminates.

6. Global convergence

If the assumptions of Theorem 4.5 apply and the line search ensures a monotonic decrease of L^a, all iterates during the optimization lie in the bounded level set

$$\mathcal{N}_0 := \{(y, \bar{y}, u) : L^a(y, \bar{y}, u) \leq L^a(y_0, \bar{y}_0, u_0)\}.$$

We can show that the search directions s are gradient related:

Proposition 6.1. *If Theorem* 4.5 *applies and* $N_{uu} \succeq 0$, *then there exists* $C > 0$ *with*

$$\cos\gamma = -\frac{s^\top \nabla L^a}{\|\nabla L^a\|\|s\|} \geq C > 0 \qquad \textit{for all} \quad (y, \bar{y}, u) \in \mathcal{N}_0,$$

where the step s is computed with the preconditioner B introduced in (5.13).

Proof. See [20, Proposition 5.1]. □

The alternating approach does not affect the above result. Actually, we employ a pure design step only if (5.15) holds and thus

$$\frac{-\Delta u^\top \nabla_u L^a}{\|\nabla L^a\|\|\Delta u\|} \geq \frac{1}{(1+\tau)} \frac{-s^\top \nabla L^a}{\|\nabla L^a\|\|s\|}.$$

We use a pure feasibility step if $\tau\ \Delta u^\top \nabla_u L^a \geq \Delta y^\top \nabla_y L^a + \Delta \bar{y}^\top \nabla_{\bar{y}} L^a$ which gives

$$-(\Delta y^\top \nabla_y L^a + \Delta \bar{y}^\top \nabla_{\bar{y}} L^a) \geq -\frac{\tau}{(1+\tau)}\ s^\top \nabla L^a.$$

In addition, since Theorem 4.5 applies all level sets of the continuous function L^a are bounded which implies that L^a is bounded below. Therefore, using the well-known effectiveness of the line search procedure based on a standard backtracking [1] and the gradient relatedness result established in Proposition 6.1, we obtain

$$\lim_{k\to\infty} \|\nabla L^a(y_k, \bar{y}_k, u_k)\| = 0.$$

7. Numerical experiment: The Bratu problem

The Bratu problem is frequently used in combustion modeling:

$$\begin{array}{rcll} \Delta y(x) + e^{y(x)} & = & 0 & x = (x_1, x_2) \in [0,1]^2 \\ y(0, x_2) & = & y(1, x_2) & x_2 \in [0,1] \\ y(x_1, 0) & = & \sin(2\pi x_1) & x_1 \in [0,1] \\ y(x_1, 1) & = & u(x_1) & x_1 \in [0,1]. \end{array}$$

The function u is a boundary control that can be varied to minimize the objective

$$f(y,u) = \int_0^1 \left(\partial_{x_2} y(x_1, 1) - 4 - \cos(2\pi x_1)\right)^2 dx_1 + \sigma \int_0^1 \left(u^2 + u'^2\right) dx.$$

We use $\sigma = 0.001$, an initial control is $u(x_1) = 2.2$ (see [17]), a five point central difference scheme with $h = 1/10$, and Jacobi's method.

To solve the minimization problem, we use power iterations to compute the spectral radius $\rho_{N_{yy}}$ of the matrix N_{yy} and $\rho_{G_y^*}$ of $G_y^\top G_y$. Then, we update $\theta = \rho_{N_{yy}}$ and $\rho = \sqrt{\rho_{G_y^*}}$. We update the ratio q introduced in (5.10) from

$$q_k = \max\{q_{k-1}, \frac{\|N_y(y_k, \bar{y}_k, u_k + \Delta u_k) - N_y(y_k, \bar{y}_k, u_k)\|_2^2}{\|G(y_k, u_k + \Delta u_k) - G(y_k, u_k)\|_2^2}\},$$

and used α, β, σ as in (5.11), (5.8). We compared the number of iterations N_{opt} needed for the optimization with the alternating approach, i.e., to reach

$$\alpha \|G(y_k, u_k) - y_k\|_2^2 + \beta \|N_y(y_k, \bar{y}_k, u_k) - \bar{y}_k\|_2^2 + \|\Delta u_k\|_2^2 \leq \varepsilon := 10^{-4},$$

and the number of iterations N_f required to reach feasibility with fixed u:

$$\|G(y_k, u) - y_k\|_2^2 + \|N_y(y_k, \bar{y}_k, u) - \bar{y}_k\|_2^2 \leq \varepsilon,$$

We used a mesh size $h \in [0.055, 0.125]$. The behaviors with respect to h of N_{opt}, N_f and of the ratio $R = N_{\text{opt}}/N_f$ are depicted in Figure 1. It shows that the number of iterations N_{opt} needed to solve the optimization problem is always bounded by a reasonable factor (here 4.6 at maximum) times the number of iterations N_f required to reach feasibility: bounded retardation. Although both numbers grow while decreasing h, the ratio $R = N_{\text{opt}}/N_f$ in Figure 2 seems reaching some limit slightly bigger than 2 for small values of h. In the case of L^a, the two line search procedures give results that are numerically indistinguishable.

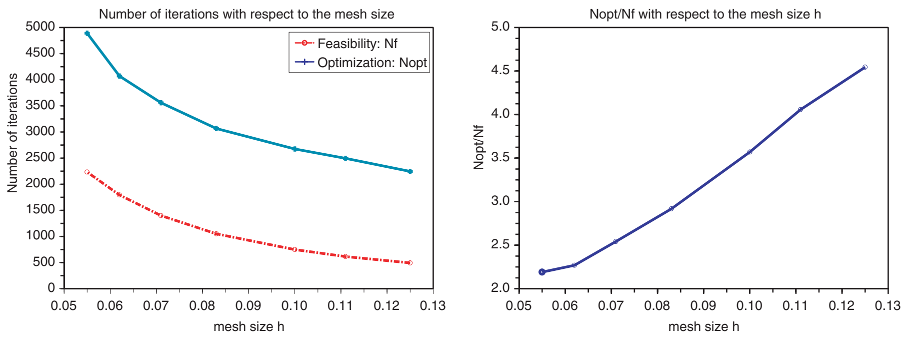

FIGURE 1. Iterations (left) and retardation factor w.r.t. mesh size h

8. Application: Aerodynamic shape optimization

As a first realistic application, we consider shape optimization of a RAE2822 transonic airfoil whose aerodynamic properties are calculated by a structured Euler solver with quasi-unsteady formulation based on pseudo time steps. The objective is to reduce the inviscid shock that is present on the initial airfoil and therefore to minimize the drag. The adjoint solver which calculates the necessary sensitivities for the optimization is based on discrete adjoints and derived by using reverse mode of automatic differentiation. A detailed information about the presented work can be found in [29].

8.1. Governing equations and boundary conditions

Since we are interested in drag reduction in transonic flow regime, the compressible Euler equations are an appropriate choice. They are capable of describing the (inviscid) shocks, which are the main sources of the pressure drag.

Even though the flow steady, the solution is obtained by integrating the (quasi-)unsteady Euler equations in time until a steady state is reached. These time steps do not have any physical meaning and are called pseudo-time steps.

For 2D flow, the compressible Euler equations in Cartesian coordinates read:

$$\frac{\partial w}{\partial t} + \frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} = 0 \ \text{ with } \ f = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u H \end{bmatrix} \ \text{and } g = \begin{bmatrix} \rho v \\ \rho v u \\ \rho v^2 + p \\ \rho v H \end{bmatrix}, \tag{8.1}$$

where w is the vector of conserved variables $\{\rho,\ \rho u,\ \rho v,\ \rho E\}$ (ρ is the density, u and v are the velocity components and E denotes the energy).

As boundary conditions, we assume the Euler slip condition on the wall ($\vec{n}^T \vec{v} = 0$) and free stream conditions at the farfield. For a perfect gas holds

$$\begin{aligned} p &= (\gamma - 1)\rho(E - \frac{1}{2}(u^2 + v^2)) && (8.2) \\ \rho H &= \rho E + p && (8.3) \end{aligned}$$

for pressure p and enthalpy H. The pressure and drag coefficients are defined as

$$C_p := \frac{2(p - p_\infty)}{\gamma M_\infty^2 p_\infty}, \qquad C_d := \frac{1}{C_{ref}} \int_C C_p(n_x \cos\alpha + n_y \cos\alpha) dl \; . \tag{8.4}$$

8.2. Shape parameterization

In aerodynamic shape optimization, there are mainly two ways of doing the shape updates: Either parameterizing the shape itself or parameterizing shape deformations. In [27] these possibilities are investigated in detail. In the following, we take the second approach, such that an initial airfoil shape is deformed by some set of basis functions that are scaled by certain design parameters. Here, the basic idea of shape deformation is to evaluate these basis functions scaled with certain design parameters and to deform the camberline of the airfoil accordingly. Then, the new shape is simply obtained by using the deformed camberline and the initial thickness distribution. The result is a surface deformation that maintains the airfoil thickness.

We have chosen Hicks-Henne functions, which are widely used in airfoil optimization. These function have the positive property that they are defined in the interval $[0, 1]$ with a peak position at a and they are analytically smooth at zero and one. The normalized airfoil shape is deformed by using Hicks-Henne functions multiplied by the design parameters u_i:

$$\Delta\,\mathrm{camber}(x) = \sum u_i h\left\{a, b\right\}(x) \text{ and } \mathrm{camber}(x) + = \Delta\,\mathrm{camber}(x) \; . \tag{8.5}$$

After deforming the airfoil geometry, a difference vector is calculated and finally performs a mesh deformation by using this difference vector. This approach is also very advantageous in terms of gradient computations, since we have to differentiate only the simple structured mesh and shape deformation tools, instead of complex mesh generators.

The numerical solution of (8.1) is computed by the TAUij code, which is a structured quasi 2D version of the TAU code, developed at the German Aerospace Center (DLR). For the spatial discretization the MAPS+ [31] scheme is used. For the pseudo time stepping, a fourth-order Runge-Kutta scheme is applied. To accelerate the convergence, local time stepping, explicit residual smoothing and a multigrid methods are used. The code TAUij is written in C and comprises approximately 6000 lines of code distributed over several files.

8.3. Gradient computation and implementation issues

One of the key points in aerodynamic shape optimization with gradient-based methods is the computation of the derivatives. For this study, we generate the adjoint codes in a semi-automatic fashion by using reverse mode of automatic differentiation. The reverse mode of AD allows to generate discrete adjoint codes in which computational cost is independent from number of optimization parameters. The freeware AD tool ADOL-C [15] gives the possibility of applying reverse AD. Since ADOL-C is based on operator overloading strategy, for the reverse mode it is usually necessary to tape all operations that are done on the active variables (the variables which are to differentiate with respect to the selected independent parameters) on memory or disk. Because in our case the primal iteration is a fixed-point iteration, a complete taping of the primal iteration is not necessary. Since we use a one-shot approach rather than a hierarchical approach, we need to tape only one pseudo-time step in each iteration instead of the whole time-stepping. This is of course very advantageous, since the tape sizes would be extremely large, even for the case of rather coarse meshes, because the tape size of a primal iterate would be multiplied by the number of pseudo-time steps. Nevertheless, in [32] it is demonstrated how to overcome this kind of drawbacks in cases of hierarchical approaches by the so-called reverse accumulation of adjoints [3].

For the coupled iteration, we need to evaluate several derivative vectors

$$\nabla_u L^a = \alpha \Delta y^T G_u + \beta \Delta \overline{y}^T N_{yu} + N_u \ , \tag{8.6}$$

in order to update the design vectors u. Furthermore, we need to evaluate the terms N_y for the update of the adjoint states $\overline{y}$.

Note, that all expressions in (8.6) are either vectors or matrix vector products. Several subroutines of ADOL-C allow us to calculate these matrix vector products easily by using the reverse mode of AD for the first-order terms and reverse on tangent for the second-order term.

For the differentiation, we simply set the independent vector as $[u; y]$, the dependent vector as $[N; y]$ and correspondingly calculate N inside the routine that returns the goal functional C_d. It should also be mentioned that, apart from N_y, the derivatives with respect to the design parameters u are propagated within the design chain in the reverse order as vector matrix products. In addition to the flow solver, the other programs of the design chain, namely **meshdefo, difgeo, defgeo**, have to be differentiated, too. In [10], this reverse propagation of the adjoint vectors is covered in detail, and comparisons of the resulting adjoint sensitivities versus finite differences are also illustrated.

8.4. Numerical results

The numerical tests are done on the transonic shape optimization problem of a RAE2822 airfoil that is introduced previously. The number of Hicks-Henne functions for the shape parameterization are chosen to be 20. The single-step one-shot method is applied in the sense that full steps are taken in the design update. As a stopping criteria, we choose $|\Delta u| < \epsilon$, where ϵ is a user defined tolerance. For our particular application we have chosen $\epsilon = 0.0001$.

As flow conditions, we have an inflow Mach number of $M_\infty = 0.73$ and an angle of attack of $\alpha = 2°$. Within the first 30 iterations, in order to smooth out

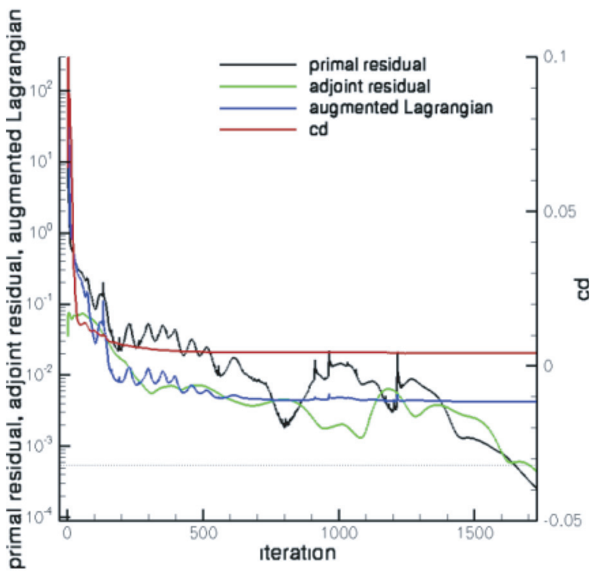

FIGURE 2. Optimization history

possible oscillatory effects, caused by the initialization of the flow field, we do only updates of the state and the adjoint state, without changes of the airfoil geometry. After these smoothing iterations, we do one-shot iterations. Figure 2 shows the optimization histories of the augmented Lagrangian, the cost functional C_d, the primal as well as the adjoint state residual. We observe, that after approximately 1600 iterations, the coupled iteration converges and the drag coefficient is reduced drastically. Consequently, we just measure a deterioration factor of 4 from the simulation to the one-shot optimization.

9. Application: Parameter optimization in a climate model

Here we present a second real-world example, this time from climate modeling, which is in detail described in [25, 26]. Parameter optimization is an important task in all kind of climate models. Many processes are not well known, some are too small-scaled in time or space, and others are just beyond the scope of the model. Here, parameters of a simplified model of the north Atlantic thermohaline circulation (THC) are optimized to fit the results to data given by a more detailed climate model of intermediate complexity.

The 4-box model of the Atlantic THC described in [34] 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, namely the southern, northern,

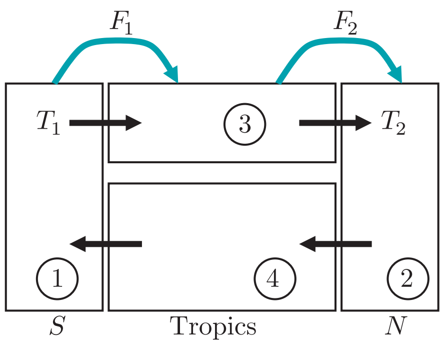

FIGURE 3. Rahmstorf box model, flow direction shown for $m > 0$.

tropical and the deep Atlantic, are the characteristics inducing the flow rate. The surface boxes exchange heat and freshwater with the atmosphere, which causes a pressure-driven circulation, compare Figure 3.

In [33] a smooth coupling of the two possible flow directions is proposed. The resulting ODE system, e.g., for boxes $i = 1, 2$, reads:

$$\begin{aligned}
\dot{T}_1 &= \lambda_1(T_1^* - T_1) + \frac{m^+}{V_1}(T_4 - T_1) + \frac{m^-}{V_1}(T_3 - T_1)\\
\dot{T}_2 &= \lambda_2(T_2^* - T_2) + \frac{m^+}{V_2}(T_3 - T_2) + \frac{m^-}{V_2}(T_4 - T_2)\\
\dot{S}_1 &= \frac{S_0 f_1}{V1} + \frac{m^+}{V_1}(S_4 - S_1) + \frac{m^-}{V_1}(S_3 - S_1)\\
\dot{S}_2 &= -\frac{S_0 f_2}{V_2} + \frac{m^+}{V_2}(S_3 - S_2) + \frac{m^-}{V_2}(S_4 - S_2).
\end{aligned}$$

Here $m = k(\beta(S_2 - S_1) - \alpha(T_2 - T_1))$ is the meridional volume transport or overturning. For boxes $i = 3, 4$ there are similar, also coupled equations. 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. The parameter a in $m^+ := \frac{m}{1-e^{-am}}, m^- := \frac{-m}{1-e^{am}}$ allows to use the model for both flow directions.

9.1. The optimization problem

Given fresh water fluxes $(f_{1,i})_{i=1}^n$ (with $n = 68$ here), the aim is to fit the values $m_i = m(f_{1,i})$ obtained by the model to data $m_{d,i}$ from a more complex model *Climber* 2, see [30], while $u = (T_1^*, T_2^*, T_3^*, \Gamma, k, \alpha)$ are the control parameters. If $F(y, u)$ denotes the right-hand side of the ODE system of the model, we get

$$\begin{aligned}
&\min_{y,u} J(y,u) := \frac{1}{2}\|m - m_d\|_2^2 + \frac{\alpha_w}{2}\|u - u_0\|_2^2, \quad m = (m_i)_{i=1}^n, m_d = (m_{d,i})_{i=1}^n,\\
&\text{s.t.} \quad \dot{y}(f_{1,i}) = F(y(f_{1,i}), u), \quad i = 1, \ldots, n.
\end{aligned}$$

The regularization term incorporates a prior guess u_0 for the parameters. The ODE system is solved by an explicit Euler method, thus G defined in Section 1.1 here represents one Euler step, but operating on all parameters f_{1i} together, i.e., for fixed u we have $G(\cdot, u) : \mathbb{R}^{8n} \to \mathbb{R}^{8n}$. Contractivity of G is not given in general, i.e., ρ in (1.2) exceeds 1 for several steps, but in average is less than 1. Also the assumption $\partial c/\partial y$ being always invertible is violated. Nevertheless, in practice G converges for fixed u but different starting values y_0 to the same stationary y^*. About 400 to $11,000$ steps are needed to reach $\|y_{k+1} - y_k\| < \varepsilon = 10^{-6}$ for iteration index k.

9.2. One-shot method for the box model

We calculate the preconditioner B defined in (5.13) in every iteration including all first- and second-order derivatives. To compute α, β and σ, we set the (iteration-dependent) contraction factor of the Euler time stepping to $\rho = 0.9$. We determine $\|G_u\|_2$ and $\|N_{yu}\|_2$ computing the Eigenvalues of $G_u^\top G_u, N_{yu}^\top N_{yu} \in \mathbb{R}^{6\times 6}$ directly, whereas for those of $N_{yy}^\top N_{yy}$ we apply a power iteration.

The forward mode of TAF [12] is used for G_u, the reverse mode for $\bar{y}^\top G_y$, and for $\bar{y}^\top G_{yu}$ and $\bar{y}^\top G_{yy}$ first the reverse and then the forward mode is applied. With only 6 parameters in our optimization problem, the reverse mode is only slightly cheaper than the forward mode for this example, and therefore is not mandatory.

For the calculation of necessary matrix-vector products (i.e., directional derivatives) we determine $\bar{y}^\top G_{yy} b, b \in \mathbb{R}^{8n}$, with a TAF generated subroutine and $J_{yy} b$ by hand. A second call of the TAF subroutine computes $N_{yy}^\top N_{yy} b = (J_{yy} + \bar{y}^\top G_{yy})^\top N_{yy} b$. In our testings, the dominant part of $N_{yy}^\top N_{yy}$ is the constant matrix $J_{yy}^\top J_{yy}$ and thus $\|N_{yy}\|_2$ does 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. In our calculations β becomes very

update of N_{yy}	#iterations	time (min)	$J(y^*, u^*)$	data fit
every 10000 iterations	1,037,804	6.174	14.879	14.053
every 1000 iterations	1,010,011	6.148	14.879	14.053
every iteration	1,015,563	10.394	14.879	14.053

Table 1. Effect on the optimization of rare update of N_{yy}, $\alpha_w = 0.1$.

small ($\approx 10^{-5}$) whereas α is large ($\approx 10^5$). Since $N_{yu}^\top N_{yu}$ contains quite large values and $G_u^\top G_u$ only small ones, we assume that α, β are well chosen.

9.3. Comparison between BFGS and One-shot strategy, bounded retardation

We compared the One-shot approach with a standard BFGS method. For the latter, in each iteration the model has to be run into a steady state for all $f_{1,i}$.

Without any regularization, the One-shot method does not converge, and BFGS finds optimal values u^* with $\frac{\partial J}{\partial u}(u^*) = 0$ being far away from reality. Generally, the smaller α_w the better the fit of the data becomes. The optimization

	$\alpha_w = 10$		$\alpha_w = 0.1$		$\alpha_w = 0.001$	
	One-shot	BFGS	One-shot	BFGS	One-shot	BFGS
$J(y^*, u^*)$	25.854	26.221	14.879	15.926	12.748	11.411
data fit	0.269	0.277	0.206	0.213	0.183	0.166
# iterations	1,269,019	20	1,010,011	28	10,678,000	65
# Euler steps	1,269,019	1,285,203	1,010,011	1,808,823	10,678,000	4,236,481

TABLE 2. Results of the optimization

results by both methods differ only slightly, see Table 2. The total number of needed time steps for the One-shot method was smaller compared to BFGS for $\alpha_w \geq 0.1$. Concerning computational time, the BFGS method is a little bit faster (since the Euler steps are cheaper due to the smaller system size), and only in the case where $\alpha_w = 0.001$ even significantly with a relation of $1 : 5.5$.

The most promising point of this study is that the One-shot strategy is much faster *close* to the optimal pair (y^*, u^*) than the BFGS method, which is actually the motivation for the One-shot approach, and one could save iterations mitigating the stopping criterion. We refer to [26] for details.

Finally we remark that it is typical in climate models that less theoretical analysis can be provided because of the complexity of real world models. For the optimization strategy, its quality and usefulness is even more convincing if good results are achieved even though convergence assumptions are not fulfilled.

References

[1] J.F. Bonnans, J. Charles Gilbert, C. Lemaréchal, C.A. Sagastizábal,*Numerical Optimization Theoretical and Practical Aspects*, Springer Berlin Heidelberg (2003).

[2] A. Brandt, *Multi-Grid techniques:* 1984 *guide with applications to fluid dynamics*, GMD-Studien. no. 85, St. Augustin, Germany (1984).

[3] Christianson B., *Reverse accumulation and attractive fixed points*, Optimization Methods and Software, **3** (1994), 311–326.

[4] Christianson, B., Reverse Accumulation and Implicit Functions. *Optimization Methods and Software* **9**:4 (1998), 307–322.

[5] J.E. Dennis, Jr. and R.B. Schnabel, *Numerical Methods for unconstrained optimization and Nonlinear Equations*, Prentice-Hall (1983).

[6] G. Di Pillo and L. Grippo. *A Continuously Differentiable Exact Penalty Function for Nonlinear Programming Problems with Inequality Constraints.* SIAM J. Control Optim. **23** (1986), 72–84.

[7] G. Di Pillo, *Exact penalty methods, in E. Spedicato* (*ed.*), *Algorithms for continuous optimization: the state of the art*, Kluwer Academic Publishers (1994), 209–253.

[8] L.C.W. Dixon, *Exact Penalty Function Methods in Nonlinear Programming.* Report NOC, The Hatfield Polytechnic, **103** (1979).

[9] R. Fontecilla, T. Steihaug, R.A. Tapia, *A Convergence Theory for a Class of Quasi-Newton Methods for Constrained Optimization.* SIAM Num. An. **24** (1987), 1133–1151.

[10] N.R. Gauger, A. Walther, C. Moldenhauer, M. Widhalm, *Automatic Differentiation of an Entire Design Chain for Aerodynamic Shape Optimization*, Notes on Numerical Fluid Mechanics and Multidisciplinary Design **96** (2007), 454–461.

[11] N.R. Gauger, A. Griewank, J. Riehme, *Extension of fixed point PDE solvers for optimal design by one-shot method – with first applications to aerodynamic shape optimization*, European J. Computational Mechanics (REMN), **17** (2008), 87–102.

[12] R. Giering, T. Kaminski, T. Slawig, Generating efficient derivative code with TAF: Adjoint and tangent linear Euler flow around an airfoil. *Future Generation Computer Systems* **21**:8 (2005), 1345–1355.

[13] I. Gherman, V. Schulz, *Preconditioning of one-shot pseudo-timestepping methods for shape optimization*, PAMM **5**:1(2005), 741–759.

[14] J.C. Gilbert, *Automatic Differentiation and iterative Processes*, Optimization Methods and Software **1** (1992), 13–21.

[15] A. Griewank, D. Juedes, J. Utke, *ADOL-C: A package for the automatic differentiation of algorithms written in C/C++.* ACM Trans. Math. Softw. **22** (1996), 131–167.

[16] A. Griewank, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation.* No. 19 in Frontiers in Appl. Math. SIAM, Philadelphia, PA, (2000).

[17] A. Griewank, *Projected Hessians for Preconditioning in One-Step Design Optimization*, Large Scale Nonlinear Optimization, Kluwer Academic Publ. (2006), 151–171.

[18] A. Griewank, C. Faure, *Reduced Functions, Gradients and Hessians from Fixed Point Iteration for state Equations*, Numerical Algorithms **30,2** (2002), 113–139.

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

[20] A. Hamdi, A. Griewank, *Reduced quasi-Newton method for simultaneous design and optimization*, Comput. Optim. Appl. online `www.springerlink.com`

[21] A. Hamdi, A. Griewank, *Properties of an augmented Lagrangian for design optimization*, Optimization Methods and Software **25**:4 (2010), 645–664.

[22] A. Jameson, *Optimum aerodynamic design using CFD and control theory*, In 12th AIAA Computational Fluid Dynamics Conference, AIAA paper 95-1729, San Diego, CA, 1995. American Institute of Aeronautics and Astronautics (1995).

[23] T. Kaminski, R. Giering, M. Vossbeck, Efficient sensitivities for the spin-up phase. In *Automatic Differentiation: Applications, Theory, and Implementations*, H.M. Bücker, G. Corliss, P. Hovland, U. Naumann, and B. Norris, eds., Springer, New York. Lecture Notes in Computational Science and Engineering **50** (2005), 283–291.

[24] C.T. Kelley, *Iterative Methods for optimizations*, Society for Industrial and Applied Mathematics (1999).

[25] C. Kratzenstein, T. Slawig, One-shot parameter optimization in a box-model of the North Atlantic thermohaline circulation. *DFG Preprint SPP*1253-11-03 (2008).

[26] C. Kratzenstein, T. Slawig, One-shot Parameter Identification – Simultaneous Model Spin-up and Parameter Optimization in a Box Model of the North Atlantic Thermohaline Circulation. *DFG Preprint SPP*1253-082 (2009).

[27] B. Mohammadi, O. Pironneau. *Applied Shape Optimization for Fluids. Numerical Mathematics and scientific Computation*, Cladenen Press, Oxford (2001).

[28] P.A. Newman, G.J.-W. Hou, H.E. Jones, A.C. Taylor. V.M. Korivi. *Observations on computational methodologies for use in large-scale, gradient-based, multidisciplinary design incorporating advanced CFD codes.* Technical Memorandum 104206, NASA Langley Research Center. AVSCOM Technical Report 92-B-007 (1992).

[29] E. Özkaya, N. Gauger, Single-step one-shot aerodynamic shape optimization. *DFG Preprint SPP*1253-10-04 (2008).

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

[31] C.C. Rossow, *A flux splitting scheme for compressible and incompressible flows*, Journal of Computational Physics, **164** (2000), 104–122.

[32] S. Schlenkrich, A. Walther, N.R. Gauger, R. Heinrich, *Differentiating fixed point iterations with ADOL-C: Gradient calculation for fluid dynamics*, in H.G. Bock, E. Kostina, H.X. Phu, R. Rannacher, editors, Modeling, Simulation and Optimization of Complex Processes – Proceedings of the Third International Conference on High Performance Scientific Computing 2006 (2008), 499–508.

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

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

Nicolas Gauger
DLR Braunschweig und Humboldt-Universität zu Berlin
Institut für Mathematik
Unter den Linden 6
D-10099 Berlin, Germany
e-mail: `nicolas.gauger@dlr.de`

Andreas Griewank, Adel Hamdi and Emre Özkaya
Institut für Mathematik
Fakultät Math.-Nat. II
Humboldt-Universität
Unter den Linden 6
D-10099 Berlin, Germany
e-mail: `Andreas.Griewank@math.hu-berlin.de`
`adel.hamdi@insa-rouen.fr`
`ozkaya@math.hu-berlin.de`

Thomas Slawig and Claudia Kratzenstein
Christian-Albrechts-Universität zu Kiel
Institut für Informatik and DFG Cluster of Excellence *The Future Ocean*
D-24098 Kiel, Germany
e-mail: `ts@informatik.uni-kiel.de`
`ctu@informatik.uni-kiel.de`