A Matrix-free Algorithm for PDE-governed Optimization with Inequality Constraints

Meng, Pengfei Dener, Alp Hicken, Jason

Abstract

We present a matrix-free method for Partial-Differential-Equation(PDE) constrained optimization problems formulated in the reduced space. When many state-based constraints are present in the reduced-space formulation, it may not be practical to compute the constraint Jacobian explicitly. This leads many practitioners to use constraint aggregation, which can produce overly conservative results. A matrix-free Newton-Krylov optimization algorithm avoids these problems, but presents additional challenges related to globalization, preconditioning, and inequality constraint handling abilities. To address the former challenge, the proposed method uses a globally convergent homotopy continuation approach that uses a predictor-corrector algorithm to trace the solution curve. The predictor and corrector phases are solved using a Krylov iterative method that uses second-order adjoints to evaluate the necessary matrix-vector products. To cope with the poorly conditioned primal-dual system, a matrix-free preconditioner is proposed that uses a low-rank SVD approximation of the condensed Hessian term, which is constructed using several iterations of the Lanczos method. The algorithm is verified using analytical problems and exercised on a complex stress-constrained mass-minimization problem. The method shows promising performance relative to a state-of-the-art matrix-based active-set algorithm, particularly for large numbers of design variables.

1 Introduction

We are interested in solving engineering design optimization problems that are governed by partial differential equations (PDEs). Such PDE-constrained optimization problems arise in many engineering applications including aerodynamic shape optimization [29, 30, 37], structural optimization [13, 29, 26], and thermodynamic optimization [12, 4, 3]. A generic PDE-constrained optimization problem can be stated as

$$\begin{aligned} \min_{x,u} \quad & f(x,u) \\ \text{subject to} \quad & h(x,u) = 0 \\ & g(x,u) \geq 0 \\ & \mathcal{F}(x,u) = 0, \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n, u \in \mathbb{R}^v$ are the design and state vectors respectively, and $f : \mathbb{R}^n \to \mathbb{R}, h : \mathbb{R}^n \to \mathbb{R}^l, g : \mathbb{R}^n \to \mathbb{R}^m$ are the objective, equality and inequality constraints respectively. We assume that f, h and g have continuous second derivatives.

Problem (1) can be solved in either the full space or the reduced space. Full-space methods [7, 8, 19] include the PDE constraint explicitly in the optimization problem, whereas reduced-space methods [1] assume that the states are defined by the design variables via the implicit-function theorem, i.e. $\mathcal{F}(x, u(x)) = 0$. One potential advantage of full-space approaches is that $\mathcal{F} = 0$ does not need to be satisfied during intermediate optimization iterations, avoiding the computational expense of tightly converging the PDE residual; however, this is also a potential disadvantage in practical engineering problems, because, if the optimization fails to converge, the intermediate solution may not be feasible with respect to the physics. Furthermore, for highly nonlinear PDEs, e.g. gas dynamics with shocks and boundary layers, practitioners have developed specialized globalization strategies that may be difficult to take advantage of in general-purpose full-space optimization algorithms.

For the reasons above, we believe that reduced-space algorithms are attractive for engineering optimization problems. Formally, the reduced-space formulation is given by

$$\begin{aligned} \min \quad & f(x, u(x)) \\ \text{s.t.} \quad & h(x, u(x)) = 0 \\ & g(x, u(x)) \geq 0 \end{aligned} \tag{2}$$

The challenge in solving (2) is that general-purpose gradient-based optimization algorithms [31, 11, 17] require the total derivative of the objective and constraints with respect to the design variables, and each of these total derivatives requires the solution of an $v \times v$ linear system, i.e. a discretized PDE. For example, the total derivative of the ith constraint g_i is given by

$$\frac{d}{dx}(g_i) = \frac{\partial}{\partial x}(g_i) + \left[\frac{\partial}{\partial x}\mathcal{F}\right]^T \psi \tag{3}$$

where $\psi \in \mathbb{R}^v$ is the adjoint which is governed by the discretized PDE,

$$\left[\frac{\partial}{\partial u}\mathcal{F}\right]^T \psi = -\frac{\partial}{\partial u}(g_i) \tag{4}$$

which is the adjoint equation [27]. Clearly, if the number of constraints m is sufficiently large, it will be prohibitively expensive to compute all the total derivatives for optimization as each one entails an adjoint.

The unconstrained version of (2) can be solved efficiently using a Newton-Krylov (NK) algorithm applied to the first-order optimality conditions; see, for example, [1, 20, 24, 9]. NK optimization algorithms have also shown promise for equality-constrained optimization in the reduced space, because they do not require the constraint Jacobian to be form explicitly and, thus, avoid the scaling issue described earlier. For instance, Dener and Hicken [14] applied a matrix-free NK algorithm to a class of equality-constrained optimization problems that arise in multidisciplinary design optimization and would otherwise be intractable with conventional matrix-based algorithms.

Motivated by its success in the unconstrained and equality-constrained cases, we would like to extend the NK methodology to more general, inequality constrained problems. However, while NK methods can avoid the cost associated with forming the explicit constraint Jacobian, this extension faces several significant challenges. First, active-set and interior-point algorithms that make use of an explicit basis for the null space of the constraint Jacobian cannot be used, because such a basis requires the Jacobian to be explicitly available. The null **Prof. Hicken, your comments stopped abruptly here**

Homotopy methods are robust, numerically stable, and globally convergent, see [2], [34]. They have been successfully applied to difficult nonlinear PDEs, see [23, 22], and [10]. In optimization, globally convergent probability-one homotopy methods have been successfully applied on engineering optimization problems. [36] developed general convergence theory for nonlinear inequality constrained optimization, from unconstrained, bound-constrained, linear and nonlinear inequality constrained convex cases to nonconvex cases. The complementarity conditions are replaced by a cubic nonlinear system involving the multipliers and constraints. However, equality constraints are not covered, and the element-wise cubic operations are expensive for large dimensional problems. Huang et al. [25] transfered a general nonlinear optimization with equality and inequality into an inequality only problem, and used the predictor-corrector method to track the homotopy interior point map using the Conjugate Gradient method. The method derived and achieved global linear convergence under the normal cone condition. However it does not handle nonconvex objective and constraint functions.

To address the challenges, we propose using homotopy continuation methods to solve the primal-dual system from reduced-space PDE-constrained optimization problems, which comes with a natural regularization term that functions like a globalization unit and improves the condition and convexity of the system. Alongside the iterative Krylov solver used in the homotopy method, a matrix-free preconditioner based on SVD approximation is used to accelerate its convergence rate.

The paper is organized as follows. In Section 2, the proposed homotopy-based globlization optimization method is explained in detail. Section 3 introduces matrix-free preconditioners accelerating the convergence of the Krylov method used in optimization. Section 4 first uses a simple problem to verify the method works, then presents numerical experiment results benchmarked SNOPT. Section 5 is on conclusions and future work.

2 A Homotopy-based Globalization for PDE-Constrained Optimization

In order to describe the homotopy method, we reformulate (2) using slack variables:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0 \\ & g(x) - s = 0 \\ & s \geq 0 \end{aligned} \tag{5}$$

where $s \in \mathbb{R}^m$ is a vector of slack variables and we have dropped the dependence on $u(x)$ to simplify the presentation.

The *Karush-Kuhn-Tucker* (KKT) conditions, that is, the first-order necessary optimality conditions at the solution of (5) are

$$\begin{aligned} \nabla f(x) + \lambda_h^T \nabla h(x) + \lambda_g^T \nabla g(x) = 0, \\ -\mathcal{S}\Lambda_g e = 0, \\ h(x) = 0, \\ g(x) - s = 0, \\ s \geq 0, \quad \lambda_g \leq 0. \end{aligned} \tag{6}$$

One of the difficulties in solving (6), in contrast with the PDEs in simulation systems, lies in the sign requirement for the slacks and Lagrangian multipliers. In the optimization process, the slacks have to be non-negative to guarantee feasibility of the inequality constraints, and the multipliers have to be non-positive at a local minimizer. Respecting the bounds during the optimization process can help it converge to a true solution, rather than a spurious solution of the KKT system. At a solution, for each inequality constraint, either the slack or Lagrangian multiplier is strictly zero under the strict complementarity assumption, which basically says the inequality constraints should be clearly either active or inactive at the solution. Active inequality constraints have zero-valued slack variables and negative multipliers in our way of formulation, while inactive inequality constraints have positive slack variables and zero-valued multipliers.

One type of the popular interior-point methods [31] deserves being briefly mentioned here due to its close proximity with this research; the Newton-Lagrangian line-search interior-point method. In the continuation approach, a series of perturbed KKT systems are solved for a series of continuation parameter μ which represents the perturbation amount and goes to zero in the end:

$$\begin{aligned} \nabla f(x) + \lambda_h^T \nabla h(x) + \lambda_g^T \nabla g(x) = 0 \\ -\mathcal{S}\Lambda_g e - \mu e = 0 \\ h(x) = 0 \\ g(x) - s = 0 \\ s \geq 0, \quad \lambda_g \leq 0 \end{aligned} \tag{7}$$

where $\mathcal{S}$ and Λ_g are diagonal matrices with the slacks and the inequality multipliers on the diagonal respectively, e is a vector of ones in proper dimension.

At each value of μ, the nonlinear system of (7) is usually solved by SQP method with direct linear solvers [11]. The solution trajectory converges to the KKT point of the original problem in the limit as $\mu \to 0$.

Several critical and sensitive issues need careful treatment for a successful solution [31]. The continuation parameter μ in Equation (7) is in place for addressing the nonlinear complimentarity condition, not convexity. Additional globalization methods are needed using either a merit function or a filter to select quality steps that guarantee smooth progress towards the solution. Also in needed are an efficient updating strategy for μ that decreases neither too slow nor too fast on the way to zero, and a proper amount of regularization to handle nonconvexity and singularity that guarantees the resulting step update is a descent direction.

In comparison, we have found that the proposed homotopy-based optimization method is advantageous, because it simultaneously addresses nonconvexity, complimentarity, and ill-conditioning of the KKT matrix at intermediate iterates in matrix-free ways. In contrast, most interior point algorithms use direct linear algebra solvers based on factorization of explicit matrices to solve the subproblems. Although inexact variations exist [18], both the method and the preconditioners used are quite sophisticated with many parameters.

2.1 Homotopy map for optimization

Conceptually, the idea of homotopy methods is simple. To find the solution of a function $f(x)$, a homotopy map is constructed connecting the target function with an easy function through a weighting parameter. For example:

$$\mathcal{H}_a(\mu, x) = (1-\mu)f(x) + \mu g_a(x), \quad 0 \le \mu \le 1$$

where $g_a(x)$ is the simple function with obvious solutions, and μ is the homotopy parameter.

We use a simple optimization example to illustrate the idea

$$\min \quad -\frac{x^3}{3} + \frac{x^2}{2}, \qquad x \in [0, 1.5]$$

The target in optimization is to find the solution of the first-order gradient function:

$$f(x) = -x^2 + x, \qquad x \in [0, 1.5]$$

The homotopy function is constructed as follows,

$$\mathcal{H}(\mu, x) = (1-\mu)f(x) + \mu(x-a), \quad a \in [0, 1.5]$$

whose solution curves corresponding to different values of a are shown in Fig. 1 below,

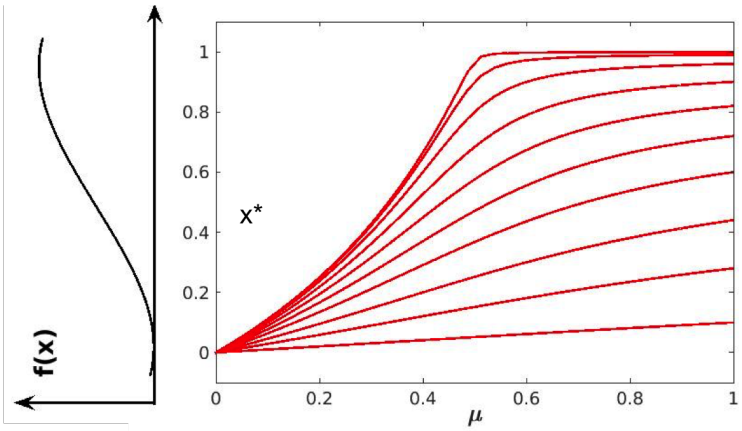


Figure 1: Solution curves for $\mathcal{H} = 0$ at different values of a

When constructing a homotopy map for complex large-scale engineering problems, the shape of the solution curves is much less intuitive and whether they are traceable needs proof. In such cases, the Parametrized Sard's Theorem, which is from Definition 2.1 and Theorem 2.2 in [35] and [32] comes to the rescue and is restated here:

Definition 2.1. Let $U \subset \mathbb{R}^m$ and $V \subset \mathbb{R}^p$ be open sets, and let $\mathcal{H} : U \times (0,1] \times V \to \mathbb{R}^p$ be a C^2 map. $\mathcal{H}$ is said to be transversal to zero if the $p \times (m+1+p)$ Jacobian matrix $D\mathcal{H}$ has full rank on $\mathcal{H}^{-1}(0)$.

Theorem 1 (Parametrized Sard's Theorem)**.** Let $\mathcal{H} : U \times (0,1] \times V \to \mathbb{R}^p$ be a C^2 map. If $\mathcal{H}$ is transversal to zero, then for almost all $a \in U$ the map

$$\mathcal{H}_a(\lambda, x) = \mathcal{H}(a, \lambda, x)$$

is also transversal to zero.

The theorem basically says that if there exists one homotopy map whose Jacobian matrix is nonsingular on its solution curve (the solution curve is smooth), then for almost all $a \in U$, the homotopy map $\mathcal{H}_a$ also has a nonsingular Jacobian on its solution curve.

In optimization, we want to solve the first order necessary optimality condition:

$$\mathcal{R}(\boldsymbol{q}) = \mathcal{R}(x, s, \lambda_h, \lambda_g) = \begin{bmatrix} \nabla f(x) + \lambda_h^T \nabla h(x) + \lambda_g^T \nabla g(x) \\ -\mathcal{S}\Lambda_g e \\ h(x) \\ g(x) - s \end{bmatrix} = 0. \tag{8}$$

For the homotopy map, we use a simple function $g_a(x) = x - a$ as the homotopy term, and a simple convex combination between the target function and the simple function:

$$\begin{aligned} \mathcal{H}(\boldsymbol{q}, \mu) &= (1-\mu)\mathcal{R}(\boldsymbol{q}) + \mu(\boldsymbol{q} - \boldsymbol{q}_0) \\ &= (1-\mu) \begin{bmatrix} \nabla f(x) + \lambda_h^T \nabla h(x) + \lambda_g^T \nabla g(x) \\ -\mathcal{S}\Lambda_g e \\ h(x) \\ g(x) - s \end{bmatrix} + \mu \begin{bmatrix} x - x_0 \\ s - s_0 \\ -(\lambda_h - \lambda_{h0}) \\ -(\lambda_g - \lambda_{g0}) \end{bmatrix} \end{aligned} \tag{9}$$

where $\boldsymbol{q} = [x, s, \lambda_h, \lambda_g]$ represents the KKT vector, and $\boldsymbol{q}_0$ is chosen as: x_0 any initial design point, s_0 the slacks corresponding to the inequality constraints, λ_{h0} and λ_{g0} are vectors of zeros.

In respect to the Theorem and Definition, the homotopy map in (9) is a C^2 map. Although we could not prove it theoretically for now, numerical evidence so far has confirm that the Jacobian of the homotopy map $\partial\mathcal{H}/\partial\boldsymbol{q}$ is nonsingular on the solution curve.

2.2 Predictor-Corrector methods

We use a predictor-corrector algorithm to trace the zero-curve of the homotopy map, see [2] and its references. Fig. 2 illustrates the predictor and corrector phases at a certain point on the path. At a predictor step, tangent direction is calculated, and the value of μ and $\boldsymbol{q}$ are updated simultaneously by moving along the tangent. At a corrector step, the value of μ is fixed, and the homotopy function is solved using inexact Newton method to bring the solution closer to the zero-curve path. The cycle is repeated until μ is equal to machine zero, and the solution of the primal-dual system is recovered.

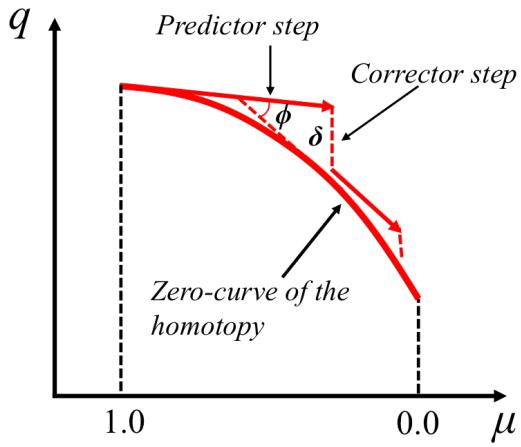


Figure 2: Illustration of Predictor-Corrector Algorithm

2.2.1 Predictor

During a predictor step we need the tangent direction $d\boldsymbol{q}/d\mu$, which can be found by taking the total derivative of $\mathcal{H}$ with respect to μ:

$$\begin{aligned} & \mathcal{H}(\boldsymbol{q}(\mu), \mu) = 0 \\ \Rightarrow \quad & \frac{d\mathcal{H}}{d\mu} = \frac{\partial \mathcal{H}}{\partial \boldsymbol{q}} \frac{d\boldsymbol{q}}{d\mu} + \frac{\partial \mathcal{H}}{\partial \mu} = 0 \\ \Rightarrow \quad & \frac{d\boldsymbol{q}}{d\mu} = -\left(\frac{\partial \mathcal{H}}{\partial \boldsymbol{q}}\right)^{-1} \frac{\partial \mathcal{H}}{\partial \mu} \end{aligned} \tag{10}$$

In practice, (10) is the linear system to be solved to get $\frac{d\boldsymbol{q}}{d\mu}$, and to put it in another way:

$$\frac{\partial \mathcal{H}}{\partial \boldsymbol{q}} \left(\frac{d\boldsymbol{q}}{d\mu}\right) = -\frac{\partial \mathcal{H}}{\partial \mu} \tag{11}$$

We use a preconditioned Krylov iterative method to solve (11) inexactly, which will be described in detail in Section 3.

Next, the predictor direction should reflect the changes in both $\boldsymbol{q}$ and μ, with μ decreasing along the solution curve. Therefore, the normalized tangent direction $\boldsymbol{t}$ given by

$$\begin{aligned} \boldsymbol{t} &= \frac{\boldsymbol{\tau}}{\|\boldsymbol{\tau}\|} \\ \boldsymbol{\tau} &= \begin{bmatrix} -\frac{d\boldsymbol{q}}{d\mu} \\ -1 \end{bmatrix} = \begin{bmatrix} \left(\frac{\partial \mathcal{H}}{\partial \boldsymbol{q}}\right)^{-1} \frac{\partial \mathcal{H}}{\partial \mu} \\ -1 \end{bmatrix} \end{aligned} \tag{12}$$

Then the predictor step is updated according to:

$$\begin{bmatrix} \boldsymbol{q}'_{k+1} \\ \mu_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_k \\ \mu_k \end{bmatrix} + \alpha \boldsymbol{\tau} \tag{13}$$

where α is the step length taken along the predictor direction and will be described shortly.

2.2.2 Corrector

At a corrector phase, with μ fixed based on the previous predictor step update, an inexact-Newton method is applied to solve the homotopy equation such that the relative residual of

the homotopy function is below some small tolerance

$$\frac{\|\mathcal{H}\left(\boldsymbol{q}_k^{(n)}, \mu_k\right)\|}{\|\mathcal{H}\left(\boldsymbol{q}_k^{(0)}, \mu_k\right)\|} \leq \epsilon_k \in [0.1, 0.5] \tag{14}$$

The loose tolerance $\epsilon_k \in [0.1, 0.5]$ is for intermediate iterations before μ_k reaches zero. At the last corrector phase when $\mu_k = 0.0$, the tolerance is tightened so that the targeted optimality and feasibility are satisfactory.

At each Newton step, the linear system to be solved is given by

$$\frac{\partial \mathcal{H}}{\partial \boldsymbol{q}} \Delta \boldsymbol{q} = -\mathcal{H} \tag{15}$$

The system matrix used in a corrector phase and a predictor phase have the same formulations:

$$\frac{\partial \mathcal{H}}{\partial \boldsymbol{q}} = \begin{bmatrix} (1-\mu)\nabla_{xx}\mathcal{L} + \mu\mathcal{I} & \mathbf{0} & (1-\mu)\mathcal{A}_h^T & (1-\mu)\mathcal{A}_g^T \\ \mathbf{0} & -(1-\mu)\Lambda_g\mathcal{S} + \mu\mathcal{S} & \mathbf{0} & -(1-\mu)\mathcal{S} \\ (1-\mu)\mathcal{A}_h & \mathbf{0} & -\mu\mathcal{I} & \mathbf{0} \\ (1-\mu)\mathcal{A}_g & -(1-\mu)\mathcal{S} & \mathbf{0} & -\mu\mathcal{I} \end{bmatrix} \tag{16}$$

2.2.3 Step-length α

Finally, the step-length α along the tangent direction is calculated using the asymptotic expansion method [10, 16]. Initially at $\mu = 1.0$, a conservative step length e.g. $\alpha_0 = 0.05$, is used for the first iteration of the predictor-corrector algorithm. At the end of the second predictor iteration, a factor ζ is calculated to update the step-length. The factor ζ is computed as follows. Referring to Fig.2, which shows two consecutive iterations of the predictor-corrector algorithm, the angle ϕ between two consecutive tangent vectors can be used to approximate the curvature of the solution path, and the distance δ approximates the distance from the predictor point to the solution path. If ϕ is high relative to an predefined ϕ_0, it means the solution path is highly curved as the tangential direction at current point make a large angle with previous tangential direction, then smaller step-length should be taken. If δ is large relative to a predefined δ_0, it means the predictor point is far away from the solution curve, and smaller step-length is advised. Therefore, the factor ζ is computed as follows.

$$\zeta = \max\{\sqrt{\frac{\delta}{\delta_0}}, \frac{\phi}{\phi_0}\} \tag{17}$$

The values of α_0, δ_0 and ϕ_0 can have a significant impact on the rate at which μ is decreased and, therefore, the performance of the algorithm. Large values of δ_0 and ϕ_0 mean that we anticipate and tolerate a coarse and twisted solution path, and the calculated factor ζ tends to be small delivering a larger step-length. In such cases μ will be driven to zero in fewer steps. Conversely, if δ_0 and ϕ_0 are small, the factor ζ will typically be large and many iterations will be needed to follow the path to $\mu = 0$. Unfortunately, the trade-off between speed and robustness is difficult to determine a priori. If the zero-curve is followed too closely, computational effort maybe wasted in over-solved middle steps as only the final corrector phase is significant. On the other hand, taking coarse steps may risk falling off the homotopy curve and lead to convergence failure. Selecting the best values for α_0 δ_0 and ϕ_0 is problem-dependent too. Currently it is done by trial-and-error, but an automated calculation scheme will be developed later.

2.3 Discussion

To safeguard the signs of the slacks and the inequality multipliers, several small schemes are applied. After a corrector iteration, the entries of the slacks and inequality multipliers with the wrong sign are set to zero. When calculating the step-length for the predictor iteration,

the fraction to the boundary rule is applied to determine the maximum allowable step based on the corrector point and the computed predictor direction. It is advisable to not interfere the Newton steps inside the corrector iteration with sign restrictions, as this will damage the convergence rate of the Newton method.

As mentioned, the added simple function in the homotopy map (9) functions as regularization terms to the system matrix (16), helping dealing with nonconvexity and ill-condition in the original KKT matrix. Starting from $\mu = 1.0$, the inertia of the system matrix (16) starts from $(n + m, l + m, 0)$ (Recall that n denotes the number of design and l and m the number of equality and inequality constraints, respectively). In such cases, the Lagrangian Hessian is positive definite in the null space of the active constraint Jacobian, see section 16.2 in [31], and the calculated design update is a descent feasible direction for the homotopy map. At $\mu = 1.0$, the system matrix is the identity saddle-point matrix. The inertia at $\mu = 1.0$ is the same as it should be at $\mu = 0.0$, provided in the latter case the necessary assumptions are met, particularly LICQ, the linear independence constraint qualification assumption (the set of active constraint gradients is linearly independent).

Finally, the Predictor-Corrector algorithm is summarized as follows:

Algorithm 1: Predictor-Corrector algorithm for optimization

Parameters: maxIter, optTol, feasTol, correctorTol, correctorMaxiter, α_0, δ_0, ϕ_0, $\mathrm{d}\mu_{max}$, $\mathrm{d}\mu_{min}$, k_{outer}, k_{inner}

Input : $\boldsymbol{q}_0$, $\mu_0 = 1.0$

1 Initial Predictor Step: compute $\frac{d\mathcal{H}}{d\mu}$, $\boldsymbol{t}$, $\boldsymbol{\tau}$ using Equation (9), (10), (13)
2 **while** $\mu > \epsilon$ *and* $k_{outer} \in [1, maxIter]$ **do**
3 $\quad \begin{bmatrix} \boldsymbol{q}'_{k+1} \\ \mu_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_k \\ \mu_k \end{bmatrix} + \alpha \boldsymbol{\tau}$
4 $\quad$ update state, adjoint at $\boldsymbol{q}'_{k+1}$
5 $\quad$ **while** (14) *or* $k_{inner} \in [1, correctorMaxiter]$ **do**
6 $\quad\quad$ solve $\frac{\partial \mathcal{H}(\boldsymbol{q}'_{k+1}, \mu_{k+1})}{\partial \boldsymbol{q}} \Delta \boldsymbol{q} = -\mathcal{H}(\boldsymbol{q}'_{k+1}, \mu_{k+1})$;
7 $\quad\quad$ $\boldsymbol{q}_{k+1} = \boldsymbol{q}'_{k+1} + \Delta \boldsymbol{q}$;
8 $\quad$ **end**
9 $\quad$ $k = k + 1$;
10 $\quad$ wrong sign of $\boldsymbol{q}_s$ and $\boldsymbol{q}_\lambda$ set to zero ;
11 $\quad$ predictor step, do line 1. ;
12 $\quad$ compute δ, ϕ, ζ using Equation (17) ;
13 $\quad$ compute $\alpha_\mu = \alpha_\mu / \zeta \in [d\mu_{min}, d\mu_{max}]$;
14 $\quad$ compute maximum allowable step-length α_s, α_λ using fraction to boundary rule ;
15 $\quad$ $\alpha = \min(\alpha_\mu, \alpha_s, \alpha_\lambda)$
16 **end**

3 Matrix-free Solution of the Linear Systems

A kernel step in the Predictor-Corrector algorithm is to solve the linear systems (11) and (15), where the system matrix takes the form (16). We use an iterative Krylov solver FGMRES as the kernel solver, together with a matrix-free preconditioner to accelerate its convergence rate.

3.1 Iterative Krylov Solver

Krylov methods construct an approximate solution to the linear system from Krylov subspaces. The subspaces are generated by the initial linear system residual and the products of the matrix with the residual, see Part VI in [33].

The method we use is FGMRES, as it allows changing preconditioners during each iteration and non positive definite systems. The linear system involved in this work (16) is a saddle

point system, which is indefinite and ill-conditioned, see [5, 6]. Using a general systems of equation to represent the two linear systems (11) and (15):

$$Ax = b \tag{18}$$

The Krylov subspace is built by the products of the system matrix A with the initial residual $r_0 = b - Ax_0$

$$K_n =< r_0, Ar_0, A^2r_0, ...A^nr_0 > \tag{19}$$

The subspace is not formed explicitly due to numerical stability issue. Instead, the Arnoldi iteration is used to construct a sequence of orthonormal matrices Q_n whose columns span the Krylov subspaces K_n. Then the problem becomes a least squares problem to find a vector y

$$\|AQ_ny - b\| = \text{minimum} \tag{20}$$

For more details, see Lecture 35 [33]. As can be seen, in solving the linear systems, the system matrix A, (16), need only to exist in the form of a matrix-vector product operation. The entries in the matrix does not have to be computed or stored explicitly, in contrast to using direct linear solvers who operates on the entries to solve the linear system. This quality is critically important in tackling large-scale problems, when computing and storing the matrix becomes a bottleneck for computational cost.

In our method, the product of (16) with an incoming vector v is computed using the Jacobian-free Newton-Krylov method as mentioned in [28], which further extended in [21] and [14] for application in reduced-space PDE-constrained optimization problems.

3.2 Preconditioner

Prevalent preconditioners are usually based on direct solvers, but we need truly matrix-free preconditioning that is generic to any PDE-constrained optimization problem in the reduced space.

During the optimization process, the preconditioning system that approximates (16) is:

$$\begin{bmatrix} (1-\mu)\tilde{\mathcal{W}} + \mu\mathcal{I} & \mathbf{0} & (1-\mu)\tilde{\mathcal{A}}_h^T & (1-\mu)\tilde{\mathcal{A}}_g^T \\ \mathbf{0} & -(1-\mu)\Lambda_g\mathcal{S} + \mu\mathcal{S} & \mathbf{0} & -(1-\mu)\mathcal{S} \\ (1-\mu)\tilde{\mathcal{A}}_h & \mathbf{0} & -\mu\mathcal{I} & \mathbf{0} \\ (1-\mu)\tilde{\mathcal{A}}_g & -(1-\mu)\mathcal{S} & \mathbf{0} & -\mu\mathcal{I} \end{bmatrix} \begin{bmatrix} v_x \\ v_s \\ v_h \\ v_g \end{bmatrix} = \begin{bmatrix} u_x \\ u_s \\ u_h \\ u_g \end{bmatrix} \tag{21}$$

where $\tilde{\mathcal{A}}_h$ and $\tilde{\mathcal{A}}_g$ are approximations to the equality and inequality constraint Jacobians, and $\tilde{\mathcal{W}}$ approximates the hessian of the Lagrangian; $\mathcal{I}$ is the Identity matrix, and Λ_g $\mathcal{S}$ are the diagonal matrices with the inequality multipliers and slack variables on the diagonal.

First we use some notations to represent each block in order to simplify the following derivation process:

$$\begin{bmatrix} \mathcal{W}' & 0 & \mathcal{A}_h'^T & \mathcal{A}_g'^T \\ 0 & \Lambda' & 0 & \mathcal{S}' \\ \mathcal{A}_h' & 0 & \mathcal{I}' & 0 \\ \mathcal{A}_g' & \mathcal{I}'' & 0 & \mathcal{I}' \end{bmatrix} \begin{bmatrix} v_x \\ v_s \\ v_h \\ v_g \end{bmatrix} = \begin{bmatrix} u_x \\ u_s \\ u_h \\ u_g \end{bmatrix} \tag{22}$$

Note the correspondence between (22) and (21)

$$\begin{aligned} \mathcal{W}' &= (1-\mu)\tilde{\mathcal{W}} + \mu\mathcal{I} \\ \mathcal{A}_h' &= (1-\mu)\tilde{\mathcal{A}}_h \\ \mathcal{A}_g' &= (1-\mu)\tilde{\mathcal{A}}_g \\ \Lambda' &= -(1-\mu)\Lambda_g\mathcal{S} + \mu\mathcal{S} \\ \mathcal{S}' &= -(1-\mu)\mathcal{S} \\ \mathcal{I}'' &= -(1-\mu)\mathcal{S} \\ \mathcal{I}' &= -\mu\mathcal{I} \end{aligned} \tag{23}$$

We use a scaled identity matrix to represent $\tilde{\mathcal{W}}$

$$\tilde{\mathcal{W}} = \beta I \tag{24}$$

where β is a scalar. Therefore, $\mathcal{W}'$ in (22) is a diagonal matrix, weighted by μ between $\tilde{\mathcal{W}}$ and $\mathcal{I}$. Note that $\mathcal{S}'$ and $\mathcal{I}''$ represents the same block in this case, which leaves room for notating the other asymmetrical system matrix of (16).

Then we reduce the system (22) by eliminating the slack and inequality multiplier row blocks to get:

$$\begin{bmatrix} \left(\mathcal{W}' + \mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}\mathcal{S}' - \mathcal{I}'\right)^{-1} \mathcal{A}_g'\right) & \mathcal{A}_h'^T \\ \mathcal{A}_h' & \mathcal{I}' \end{bmatrix} \begin{bmatrix} v_x \\ v_h \end{bmatrix} = \begin{bmatrix} u_x - \mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}\mathcal{S}' - \mathcal{I}'\right)^{-1} \left(-u_g + \mathcal{I}''\Lambda'^{-1}u_s\right) \\ u_h \end{bmatrix} \tag{25}$$

For inequality-only case, the focus is to solve:

$$\left(\mathcal{W}' + \mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}S' - \mathcal{I}'\right)^{-1} \mathcal{A}_g'\right) v_x = u_x - \mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}\mathcal{S}' - \mathcal{I}'\right)^{-1} \left(-u_g + \mathcal{I}''\Lambda'^{-1}u_s\right) \tag{26}$$

For the term $\mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}\mathcal{S}' - \mathcal{I}'\right)^{-1} \mathcal{A}_g'$, we have the matrix-vector products available, but not the explicit matrices. Therefore, we apply the Lanczos algorithm using matrix-vector products to make a low-rank truncated SVD approximation of the whole term.

$$\mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}\mathcal{S}' - \mathcal{I}'\right)^{-1} \mathcal{A}_g' = M_{m\times k}\Gamma_{k\times k}N^*_{k\times m} \tag{27}$$

where the columns of M are the left-singular vectors, the diagonal elements of Γ are the singular values, and the columns of N are the right-singular vectors. k is the number of singular values, or the number of Lanczos subspace sizes used for approximation, which is usually much smaller than the full space rank of the matrix, $k < m$.

Next, we use the generalized Sherman-Morrison equation to get the direct inverse of the matrix in (26). Recall that the generalized Sherman-Morrison is as follows:

$$\begin{aligned} B &= A + UV \\ B^{-1} &= A^{-1} - A^{-1}U(I_k + VA^{-1}U)^{-1}VA^{-1} \end{aligned} \tag{28}$$

where A is a square invertible $n \times n$ matrix, U is $n \times k$ matrix and V is $k \times n$ matrix, and it is assumed that $(I_k + VA^{-1}U)$ is invertible.

Comparing (28) and (26), we make the associations

$$\begin{aligned} B &= \left(\mathcal{W}' + \mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}S' - \mathcal{I}'\right)^{-1} \mathcal{A}_g'\right), \\ A &= \mathcal{W}', \\ U &= M_{m\times k}, \\ V &= \Gamma_{k\times k}N^*_{k\times m}. \end{aligned} \tag{29}$$

Therefore:

$$B^{-1} = \mathcal{W}'^{-1} - \mathcal{W}'^{-1}M \left(I_k + \Gamma N^*\mathcal{W}'^{-1}M\right)^{-1} \Gamma N^*\mathcal{W}'^{-1} \tag{30}$$

where the inverse block $\left(I_k + \Gamma N^*\mathcal{W}'^{-1}M\right)^{-1}$ is a square matrix with each dimension the size of the number of design, which is very small in our application field and whose inverse can be obtained by direct linear solvers e.g. LU factorization.

From here, we first calculate the right-hand-side of (26), then multiply it with (30) to get the preconditioned design update. The preconditioned inequality multiplier and slack update are computed using:

$$\begin{aligned} v_g &= \left(\mathcal{I}''\Lambda'^{-1}\mathcal{S}' - \mathcal{I}'\right)^{-1} \left(-u_g + \mathcal{I}''\Lambda'^{-1}u_s + \mathcal{A}_g'v_x\right) \\ v_s &= \Lambda'^{-1}(-\mathcal{S}'v_g + u_s) \end{aligned} \tag{31}$$

The Preconditioner routine is summarized as follows:

Algorithm 2: Preconditioner for Krylov method

Parameters: $\boldsymbol{q}_k = [x, s, \lambda_h, \lambda_g]$, μ_k
Input : $[u_x, u_s, u_h, u_g]$ in (21)
Output : $[v_x, v_s, v_h, v_g]$ in (21)

1 build SVD approximation of $\left(\mathcal{A}_g'^T \left(\mathcal{I}''\Lambda'^{-1}\mathcal{S}' - \mathcal{I}'\right)^{-1} \mathcal{A}_g'\right)$ using Lanczos as in (27)
2 compute the right-hand-side vector $v_x -$ rhs in (26)
3 compute the inverse approximation of the matrix in (26) from Sherman-Morrison (30)
4 compute v_x by multiplying (30) with $v_x -$ rhs in line 2
5 compute v_g and v_s following (31)

4 Numerical Tests

In this section, we show three numerical experiments. The first one is to test the new optimization method on a constructed non-convex problem. The second one is on a linear constrained quadratic problem with the Hessian and constraint Jacobian matrix constructed from known singular values a-prior, the scalability performance of the new optimization method is compared against SNOPT [17], which is a SQP active set method based on explicit constraint Jacobian. The third example is on a structural sizing design problem, which is PDE-constrained, and has a nonconvex Hessian; the performance of the new method and SNOPT is analyzed.

4.1 Constructed Non-convex Problem

As a start, we test the method on a simple nonconvex problem as follows:

$$\begin{aligned} \min_{x \in R^n} \quad & x^T Q x \\ \text{s.t.} \quad & x_l \leq x \leq x_u \end{aligned}$$

where:

$$\begin{aligned} Q &= \text{diag}[1, -1, ...1, -1, -1] \in \mathbb{R}^{100} \\ x_l &= -1 \\ x_u &= 1 \end{aligned}$$

The positive entries of Q represents positive definite quadratic systems, while the negative entries negative definite systems. For the positive entries of Q, the analytical solution is at $x_i = 0.0$, and for the negative entries of Q, the solution is at $x_i = -1.0 or 1.0$. This is just to test that the new method can handle the nonconvex problems, bypassing the local maximum stationery points and recover the true minimum point in case of nonconvexity.

The convergence plot is as follows:

4.2 Constructed Quadratic Problem

Next the method is tested on a constructed quadratic problem with linear inequality constraints to validate the SVD based preconditioner. The objective Hessian and constraint Jacobian matrices are constructed with known singular value distributions. The problem is formulated as follows:

$$\begin{aligned} \min_{x \in R^n} \quad & \frac{1}{2} x^T Q x + g^T x \\ \text{s.t.}_{b \in R^n} \quad & Ax \geq b \end{aligned}$$

where:

$$A = U\Sigma V^* \quad U \text{ and } V \text{are randomly generated orthonormal matrix }, A \in \mathbb{R}^{n\times n}$$
$$\Sigma = 10 * diag(\sigma_1, ...\sigma_i, ...\sigma_k, \sigma_k, \sigma_k, ...) \quad \text{with } \sigma_i = \frac{1}{i^2},$$
$$Q = E\Lambda E^T \quad E \text{ is a permutation of the Identity matrix, or orthonormal matrix}$$
$$\Lambda = 10 * diag(\lambda_1, ...\lambda_i, ...\lambda_t, \lambda_t, \lambda_t, ...) \quad \text{with } \lambda_i = \frac{1}{i},$$

The constant scalar 10 in Σ and Λ is simply to increase the magnitude of the matrix A and Q. Because b, U and V for the inequality constraints are all randomly generated, definitely some of the constraints are active. The random number generator seed is set to a certain value to make fair cross comparisons.

In this problem, the constraint Jacobian and Hessian are constructed a-prior thus explicitly available. However, the optimization algorithm only use their matrix vector products. The previously mentioned SVD preconditioner is used. At the moment, σ_i and λ_i are all positive, making Q positive definite. The value of k and t are randomly chosen to be a fixed positive value smaller than the number of design variables. This is to make the condition number of the matrix fixed, rather than hugely increasing with the dimension of the problem.

Notice that this test problem does not have state vectors, and the total constraint Jacobian exists in explicit matrix form to be used by SNOPT. For such problems, the new method cannot track the linear solver cost exactly, and SNOPT tracks cost differently, using the number of objective and sensitivity function evaluations. Therefore, the CPU computing time is used as the x-axis. When running the tests, all other routines on the computer are closed to make a fair comparison as much as possible.

A range of different number of design variables from 100 to 500 is tested, and the convergence plots for the 200 and 500 cases are selected and shown in Fig. 3 below. The convergence for other cases have similar trends and is omitted here for saving space. In the legend, Eye optimality Eye feasibility refers to unpreconditioned case; SVD optimality and feasibility refers to the new method with the matrix-free preconditioner mentioned before; SNOPT optimality and feasibility are the convergence results from SNOPT.

It can be seen that as the problem dimension increases, the matrix-based optimization method SNOPT takes longer time to complete each step, while the new matrix-free optimization method uses roughly the same time. The bottom right subfigure in Fig. 4 shows the CPU computing time cost versus number of design.

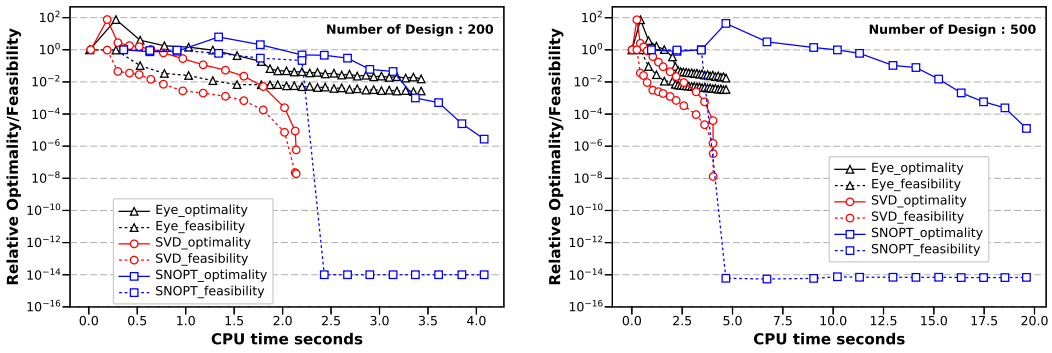


Figure 3: Convergence Plots for increasing dimensions

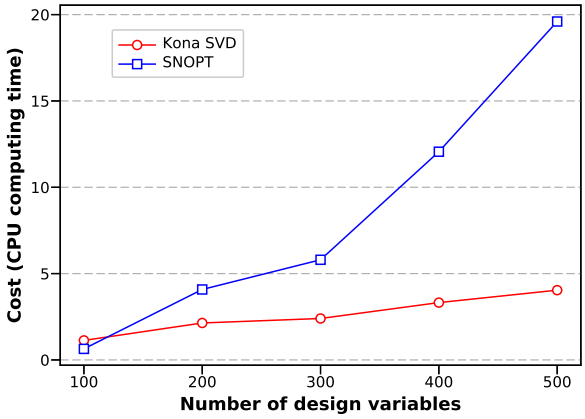


Figure 4: Scalability Performances

4.3 Stress-Constrained Mass Minimization

The second test problem is a structural sizing design problem [15]. The design variables are plate thickness on a 2-D plate with one side fixed and the other side subject to an external force. Fig.5 illustrates the problem.

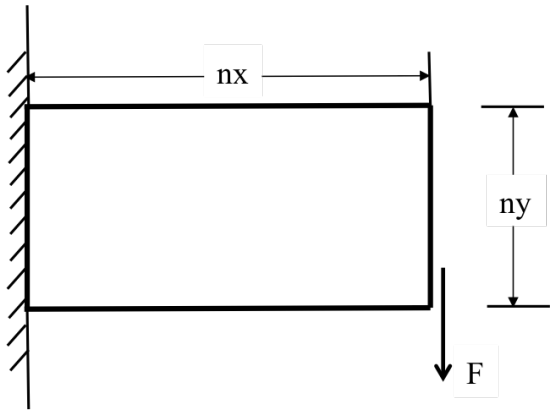


Figure 5: Plate thickness design problem

where nx and ny are the number of elements along the horizontal and vertical direction.

The objective function is the mass the of the plate, which directly depends on the thickness distribution, the design variables. The system is governed by the structural constitutive equations, with the displacements of each finite elements being the state variables. The stress distribution depends on the strain via Young's modulus, while the strain is related with the displacements. The constraints include lower and upper bound constraints on the design variables, and the von Mises stress constraint of the stress variables. The mathematical formulation is as follows:

$$\begin{aligned} \min \quad & \text{Mass}(x) \\ \text{subject to} \quad & x_l \leq x \leq x_u \\ & 0 \leq \sigma \leq \sigma_{\text{allowed}} \\ \text{governed by} \quad & \mathcal{F}(x, u) = 0 \end{aligned}$$

The problem can be easily scaled up by increasing nx and ny. We use small, medium, large to represent different scale of the problem:

Case	nx	ny	Number of design
Small	16	8	128
Medium	32	16	512
Large	64	32	2048

As the number of design increases, the number of singular values in the SVD approximation has to increase accordingly in order to capture sufficient features of the matrix (27). When the number of design variables is fixed, using more singular values can increase the convergence rate of the Krylov method more effectively, as shown in Fig. 6 below.

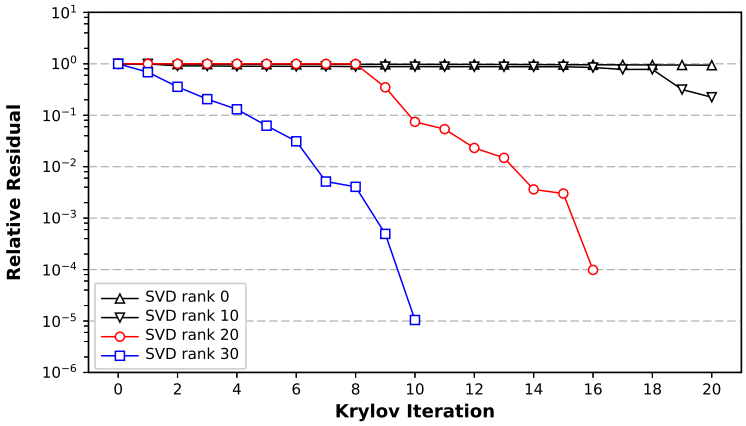


Figure 6: SVD rank on Krylov convergence rate

Fig. 7 below shows the convergence plots on the small, medium and large problem. Each plot contains the optimality and feasibility iteration using the Homotopy method with Identity preconditioner, SVD preconditioner, and SNOPT. When using the Identity preconditioner, it is very difficult for the iterative Krylov solver to make progress, as the KKT system is very ill-conditioned when μ gets close to zero. For the SVD preconditioner, the increasing cost with scaling dimension are mainly driven by the increasing number of singular values.

The y-axis in Fig. 7 shows the change of relative optimality and absolute feasibility with cost. As can be seen, using the Homotopy optimization method with the SVD preconditioner can effectively bring the optimality and feasibility down to satisfactory tolerances. SNOPT takes longer time to converge, and struggles with the large scale problem.

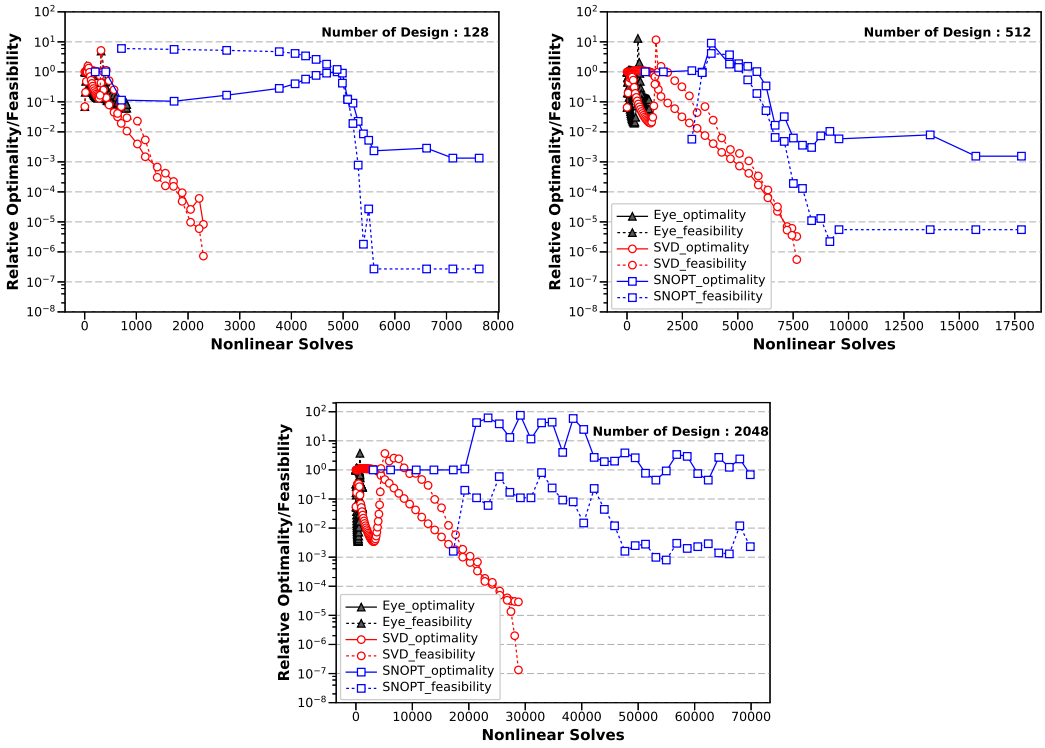
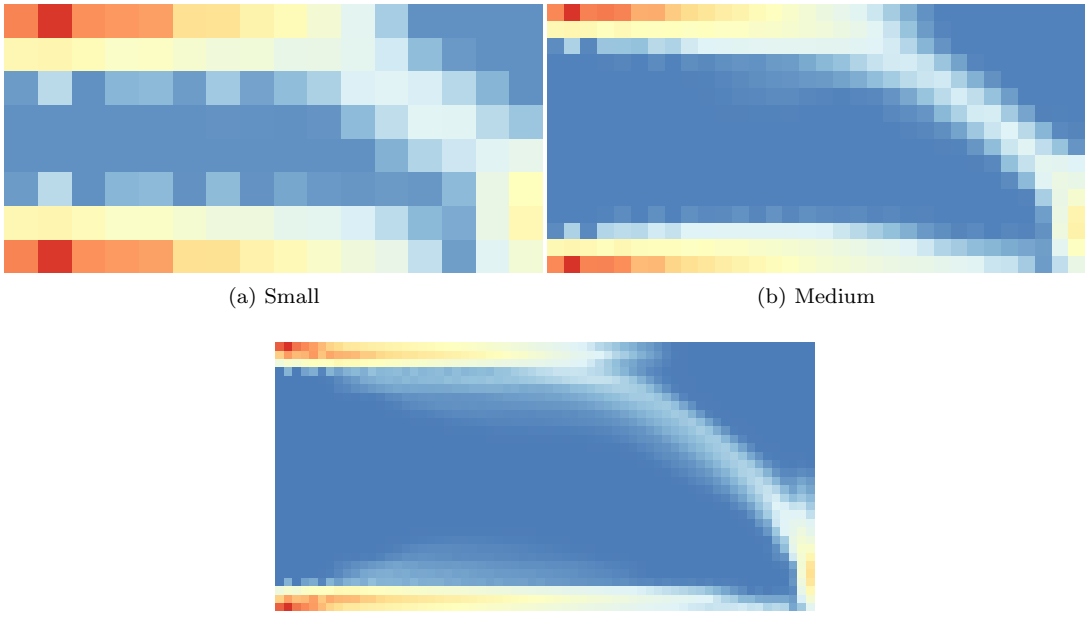


Figure 7: Convergence Plots for Structural Problem

The following Fig. 8 shows the final optimal thickness distribution obtained by the new optimization method. The presented results agree with physical intuition for the problem. The blue color at one end of the spectrum means thinner thickness, while the brighter colors like yellow and red means thicker thickness.



(a) Small

(b) Medium

(c) Large

Figure 8: Thickness distribution using the new method

5 Future Work

The matrix-free method itself can handle both equality and inequality constraints, but the SVD preconditioner can only handle inequality constraints. The next step would be to make the preconditioner equality tolerant. More test problems will be run to test the robustness of the method.

References

[1] Volkan Akçelik, George Biros, Omar Ghattas, Judith Hill, David Keyes, and Bart van Bloemen Waanders. Parallel Algorithms for PDE-Constrained Optimization. In Michael A. Heroux, Padma Raghavan, and Horst D. Simon, editors, *Parallel Processing for Scientific Computing*, chapter 16, pages 291–322. Society for Industrial and Applied Mathematics, January 2006.

[2] Eugene L. Allgower and Kurt Georg. Continuation and path following. *Acta Numerica*, 2:1–64, 1993.

[3] Adrian Bejan and Eden Mamut. *Thermodynamic optimization of complex energy systems*, volume 69. Springer Science & Business Media, 2012.

[4] Adrian Bejan, LAO Rocha, and S Lorente. Thermodynamic optimization of geometry: T-and y-shaped constructs of fluid streams. *International Journal of Thermal Sciences*, 39(9-11):949–960, 2000.

[5] Michele Benzi, Gene H. Golub, and Jörg Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14(-1):1–137, May 2005.

[6] Luca Bergamaschi, Jacek Gondzio, and Giovanni Zilli. Preconditioning indefinite systems in interior point methods for optimization. *Computational Optimization and Applications*, 28(2):149–171, 2004.

[7] George Biros and Omar Ghattas. Parallel lagrange-newton-krylov-schur methods for pde-constrained optimization. part I: the krylov-schur solver. *SIAM J. Scientific Computing*, 27(2):687–713, 2005.

[8] George Biros and Omar Ghattas. Parallel lagrange-newton-krylov-schur methods for pde-constrained optimization. part II: the lagrange-newton solver and its application to optimal control of steady viscous flows. *SIAM J. Scientific Computing*, 27(2):714–739, 2005.

[9] Alfio Borzì and Volker Schulz. *Computational Optimization of Systems Governed by Partial Differential Equations*. Society for Industrial and Applied Mathematics, January 2011.

[10] David A. Brown and David W. Zingg. A monolithic homotopy continuation algorithm with application to computational fluid dynamics. *Journal of Computational Physics*, 321:55 – 75, 2016.

[11] Richard H. Byrd, Mary E. Hribar, and Jorge Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM J. on Optimization*, 9(4):877–900, April 1999.

[12] Lingen Chen, Chih Wu, and Fengrui Sun. Finite time thermodynamic optimization or entropy generation minimization of energy systems. *Journal of Non-Equilibrium Thermodynamics*, 24(4):327–359, 1999.

[13] Klaus Deckelnick, Michael Hinze, and Tobias Jordan. An optimal shape design problem for plates. *SIAM J. Numerical Analysis*, 55(1):109–130, 2017.

[14] Alp Dener and Jason E. Hicken. Matrix-free Algorithm for the Optimization of Multidisciplinary Systems. *Structural and Multidisciplinary Optimization*, 2017. (accepted).

[15] Alp Dener, Jason E. Hicken, Pengfei Meng, Graeme J. Kennedy, John Hwang, and Justin Gray. Kona: a parallel optimization library for engineering-design problems. In *AIAA SciTech Conference*, January 2016. AIAA 2016-1422.

[16] K. Georg. *A Note on Stepsize Control for Numerical Curve Following*, pages 145–154. Springer US, Boston, MA, 1983.

[17] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: an SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.

[18] Jacek Gondzio. Matrix-free interior point method. *Computational Optimization and Applications*, 51(2):457–480, 2012.

[19] E. Haber and U. M. Ascher. Preconditioned all-at-once methods for large, sparse parameter estimation problems. *Inverse Problems*, 17(6):1847–1864, December 2001.

[20] Matthias Heinkenschloss and Luís N. Vicente. An interface optimization and application for the numerical solution of optimal control problems. *ACM Trans. Math. Softw.*, 25(2):157–190, June 1999.

[21] Jason E. Hicken. Inexact Hessian-vector products in reduced-space differential-equation constrained optimization. *Optimization and Engineering*, 15(3):575–608, September 2014.

[22] Jason E. Hicken, Howard Buckley, Michal Osusky, and David W. Zingg. Dissipation-based continuation: a globalization for inexact-Newton solvers. In *20th AIAA Computational Fluid Dynamics Conference*, number AIAA-2011-3237, Honolulu, Hawaii, United States, June 2011.

[23] Jason E. Hicken and David W. Zingg. Globalization strategies for inexact-Newton solvers. In *19th AIAA Computational Fluid Dynamics Conference*, number AIAA-2009-4139, San Antonio, Texas, United States, June 2009.

[24] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE Constraints*. Mathematical Modelling: Theory and Applications. Springer Netherlands, 2010.

[25] Qingqun Huang, Zhibin Zhu, and Xiangling Wang. A predictor-corrector algorithm combined conjugate gradient with homotopy interior point for general nonlinear programming. *Appl. Math. Comput.*, 219(9):4379–4386, January 2013.

[26] Kai A. James, Graeme J. Kennedy, and Joaquim R. R. A. Martins. Concurrent aerostructural topology optimization of a wing box. *Comput. Struct.*, 134:1–17, April 2014.

[27] Antony Jameson. Aerodynamic shape optimization using the adjoint method. In *VKI Lecture Series on Aerodynamic Drag Prediction and Reduction, von Karman Institute of Fluid Dynamics, Rhode St Genese*, pages 3–7, 2003.

[28] D. A. Knoll and D. E. Keyes. Jacobian-free newton-krylov methods: A survey of approaches and applications. *J. Comput. Phys.*, 193(2):357–397, January 2004.

[29] Andrew B. Lambe, Graeme J. Kennedy, and Joaquim R. R. A. Martins. Multidisciplinary Design Optimization of an Aircraft Wing via a Matrix-Free Approach. In *15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Atlanta, Georgia, 2014.

[30] Zhoujie Lyu and Joaquim RRA Martins. Aerodynamic design optimization studies of a blended-wing-body aircraft. *Journal of Aircraft*, 51(5):1604–1617, 2014.

[31] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[32] John Mallet-Paret Shui-Nee Chow and James A. Yorke. Finding zeroes of maps: Homotopy methods that are constructive with probability one. *Mathematics of Computation*, 32(143):887–899, 1978.

[33] Lloyd N Trefethen and David Bau III. Numerical linear algebra, 1997.

[34] layne t watson. *globally convergent homotopy methods: a tutorial*. 1986.

[35] Layne T. Watson. Probability-one homotopies in computational science. 2000.

[36] Layne T. Watson. Theory of globally convergent probability-one homotopies for nonlinear programming. *SIAM Journal on Optimization*, 11(3):761–780, 2001.

[37] Mengmeng Zhang, Arthur Rizzi, Pengfei Meng, Raj Nangia, Rafiz Amiree, and Olivier Amoignon. Aerodynamic design considerations and shape optimization of flying wings in transonic flight. In *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSM17 - 19 September 2012, Indianapolis, Indiana*, pages 1–17, 2012. QC 20121113.