General deflation for finding multiple local optima in non-convex optimization

Mohamed Tarek[1,2*] and Yijiang Huang[3,4]

[1*]Pumas-AI Inc., USA.
[2]Business School, University of Sydney, Australia.
[3]Department of Architecture, Massachusetts Institute of Technology, United States.
[4]Department of Computer Science, ETH Zürich, Switzerland.

*Corresponding author(s). E-mail(s): mohamed82008@gmail.com;
Contributing authors: yijiang.huang@inf.ethz.ch;

Abstract

Many useful classes of non-convex optimization problems have multiple local optimal solutions. These problems are commonly found in numerous applications including topology optimization and machine learning. One of the methods recently proposed to efficiently explore multiple local optimal solutions without random re-initialization relies on the concept of deflation. In this paper, different ways to use deflation in non-convex optimization and nonlinear system solving are discussed. A simple, general and novel deflation constraint is proposed to enable the use of deflation together with existing nonlinear programming solvers or nonlinear system solvers. The connection between the proposed deflation constraint and a minimum distance constraint is presented. Additionally, a number of variations of deflation constraints and their limitations are discussed. Finally, a number of applications of the proposed methodology in the fields of approximate Bayesian inference and topology optimization are presented.

Keywords: deflation, non-convex optimization, multi-start optimization, global optimization, local minima

1 Introduction

Non-convex optimization problems are used in numerous applications including: machine learning, mechanical design, economics, and the design of clinical trials, among many other applications. One of the fundamental challenges of non-convex optimization is the existence of multiple local minimizers. Local first- and second-order optimization algorithms often get stuck in a particular local minimizer without any guarantees that this is the best solution that can be found. The ability to explore multiple local minimizers is often important in practice to find better solutions or to provide more diverse choices to decision makers if all the choices are nearly equally good or if the optimization objectives cannot capture users' preferences completely. For example, multiple car body designs can be reported and decision-makers can choose one based on the subjective aesthetic appeal.

Despite the popularity of using standard non-convex optimization techniques with various random or heuristic restart strategies to find multiple optima (Rinnooy Kan and Timmer, 1987; Kaelo and Ali, 2006; Arnoud et al, 2019), these methods can be computationally inefficient because they do not protect against converging to the same solution from different starting points. In contrast, deflation-based methods (Papadopoulos et al, 2021) have the benefit of provably converging to distinct solutions under certain assumptions, even when starting from *the same initial guess*. However, deflation was originally developed for solving nonlinear systems of equations (Brown and Gearhart, 1971; Farrell et al, 2016), and later extended to non-smooth equation solving (Patrick E. Farrell and Surowiec, 2020) which generalize finding feasible solutions to equality and inequality constraints. Using deflation for optimization required a significant adaptation of a specific nonlinear programming algorithm (Papadopoulos et al, 2021). Different optimization problems and applications are generally best solved by different optimization algorithms. Having to re-design the optimization algorithm to use deflation prevents deflation from being used with other optimization algorithms.

In this paper, we propose a simple, yet effective and provably correct way of making use of deflation by reformulating the non-convex optimization problem, instead of adapting the backend optimization algorithm. We then prove the equivalence of this approach to a minimum-distance constraint under certain assumptions. The proposed approach has the benefit of (1) being simpler to implement since it is a formulation change rather than an algorithmic change; and (2) being more flexible allowing the use of arbitrary optimization algorithms that are suitable for the problem class at hand. We show a number of examples from different applications, each using deflation together with the most suitable and/or popular optimization algorithm for the respective application, which is not achievable by previous deflation-based approaches.

One limitation of the proposed approach is that it requires the handling of a non-convex, inequality constraint even if the original problem was unconstrained. However, one way to workaround this limitation is demonstrated in the examples section.

2 Related work

Systematic multi-start. Systematically restarting the optimization multiple times from random or deterministically diverse (McKay M. D., 1979; Kucherenko and Sytsko, 2005) initial solutions is a common strategy to find multiple local minimizers in non-convex optimization. Some popular algorithms following this approach are the multilevel single linkage algorithm (MLSL) (Rinnooy Kan and Timmer, 1987), the controlled random search algorithm (CRS) (Kaelo and Ali, 2006), and the TikTak algorithm (Arnoud et al, 2019). However, all the above approaches can be computationally expensive and wasteful, since restarting the optimization algorithm from a different initial solution may not give a different local minimizer.

Divide and conquer Other similar algorithms rely on sub-dividing the search space into smaller hyper-rectangles to narrow down the search space for each sub-problem. Algorithms in this category include the StoGo algorithm (Gudmundsson, 1998; K. Madsen and Zilinskas, 1998) and the DIviding RECTangles (DIRECT) algorithm (Jones et al, 1993; Gablonsky and Kelley, 2001). However, these approaches do not scale well with the number of decision variables m since the number of fixed size hyper-rectangles one can divide an m dimensional solution space into grows exponentially with m.

Hyperparameter and bilevel optimization. Alternatively, hyperparameter optimization techniques (Li et al, 2018; Falkner et al, 2018) or bilevel optimization (Sinha et al, 2018) can be used to optimize the starting point of the lower-level optimization algorithm. One can compose a global search algorithm such as an evolutionary algorithm (Gendreau and Potvin, 2010) and a local search algorithm together to create an algorithm that can explore different initial solutions and find the best local minimizers in different neighborhoods. This family of global-local algorithms is also sometimes termed memetic algorithms (Moscato and Cotta, 2010).

Global metaheuristics. Besides their use in memetic algorithms, global metaheuristic optimization algorithms (Gendreau and Potvin, 2010) can also be used as standalone algorithms, but these algorithms do not tend to scale well to large problems since they usually do not exploit the often available gradients and sometimes Hessians of objective and constraint functions.

Tunneling-based multi-start. Tunneling (Levy and Gomez, 1981; Gomez and Levy, 1982; Barron and Gomez, 1991; Gomez et al, 2003; Zhang and Norato, 2018) is another heuristic technique often used to find multiple local minimizers by finding a sufficiently different starting point with a similar or better objective value as the best solution found so far. The optimization problem is then re-solved starting from this new point in order to converge to a different solution. However, this is a two-step approach consisting of first solving the optimization problem and then finding a new, sufficiently different initial point. And the success in finding distinct minimizers depends on the success in finding a sufficiently different starting point.

Exact global optimization. In addition to restart use, there are also some well-known exact global optimization algorithms for some classes of non-convex optimization (Tawarmalani and Sahinidis, 2005; Sahinidis, 2017; Belotti et al, 2009; Vigerske and Gleixner, 2018; Wilhelm and Stuber, 2020; Nagarajan et al, 2019, 2016; Ratschek and Rokne, 2007; Gecode Team, 2006). These approaches however tend to require the explicit analytical mathematical expressions of the objective and constraint functions so they are not suitable for black-box non-convex optimization, and they typically don't scale well to large problems in practice, where the exactness guarantee has an exponential computational time complexity in the number of variables.

Deflation-based multi-start. Deflation is a recently proposed technique that employs Newton-like methods to find multiple solutions of non-linear systems of equations and optimization problems with equality and inequality constraints (Brown and Gearhart, 1971; Farrell et al, 2015, 2016, 2020). Assuming that the underlying Newton-like algorithm converges, under certain assumptions, a deflation-based solver is guaranteed to converge to a unique solution each time the solver is started from the same initial solution. This has the promise of being much less wasteful than multi-start optimization approaches. Verifying whether an operator is a deflation operator can be a difficult task in practice. This is because certain conditions must be met for it to be considered a deflation operator (see Section 3.2.1). However, the method has been shown to perform well in practical applications even when the candidate deflation operator is not proven to be a true deflation operator. This is because even in cases where the optimization algorithm converges to the same solution, deflating the same solution multiple times eventually leads to convergence to a different solution. The number of times a solution needs to be deflated is also termed its multiplicity when solving for roots of a polynomial.

Deflation was also used in a primal-dual interior point optimization algorithm (Wright, 1997) to find multiple locally optimal designs in mechanical design optimization problems (Papadopoulos et al, 2021). However, this deflated optimizer had to re-invent a primal-dual interior point optimization algorithm (Wright, 1997) since off-the-shelf solvers such as the Interior Point OPTimizer (IPOPT) (Wächter and Biegler, 2006) could not be used directly for reasons to be presented in this work. The inability to use existing optimization solvers is a huge limitation of this approach, since different optimization algorithms tend to be more suitable for different problems and applications.

3 Background

3.1 Sufficient optimality conditions for regular points

We aim to find multiple solutions for the following nonlinear program (NLP):

$$\begin{aligned} &\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{x}) \\ &\text{subject to} \\ &\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0} \ , \\ &\boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u} \end{aligned} \tag{1}$$

where $\boldsymbol{l} \in (-\infty,\infty)^n, \boldsymbol{u} \in (-\infty,\infty)^n$ are the finite (for simplicity) lower and upper bounds of the variable $\boldsymbol{x}$. The objective function $f : \mathbb{R}^n \to \mathbb{R}$ and the equality constraints $c : \mathbb{R}^n \to \mathbb{R}^m$, with $m \le n$ are assumed to be twice continuously differentiable. Problems with general nonlinear inequality constraints $\boldsymbol{d}(\boldsymbol{x}) \le \boldsymbol{0}$ can be converted to equality constraints by adding slack variables.

Let:

$$\begin{aligned} \mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{z}_+,\boldsymbol{z}_-) = f(\boldsymbol{x}) + \boldsymbol{c}(\boldsymbol{x})^T\boldsymbol{\lambda} + \\ (\boldsymbol{x}-\boldsymbol{u})^T\boldsymbol{z}_+ - (\boldsymbol{x}-\boldsymbol{l})^T\boldsymbol{z}_- \end{aligned} \tag{2}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^m$ is the Lagrangian multiplier vector of the equality constraints, $\boldsymbol{z}_- \in \mathbb{R}^n_+, \boldsymbol{z}_+ \in \mathbb{R}^n_+$ are the Lagrangian multiplier vectors of the bound constraints.

If $\boldsymbol{x}$ is regular and is a local minimizer of the NLP, then $\exists(\boldsymbol{\lambda},\boldsymbol{z}_+,\boldsymbol{z}_-)$ such that:

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{z}_+,\boldsymbol{z}_-) = \boldsymbol{0} \tag{3a}$$

$$\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0} \tag{3b}$$
$$\boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u} \tag{3c}$$
$$\boldsymbol{z}_+ \ge \boldsymbol{0} \tag{3d}$$
$$\boldsymbol{z}_- \ge \boldsymbol{0} \tag{3e}$$
$$(\boldsymbol{x} - \boldsymbol{u})^T \boldsymbol{z}_+ = 0 \tag{3f}$$
$$(\boldsymbol{x} - \boldsymbol{l})^T \boldsymbol{z}_- = 0 \tag{3g}$$

Conditions 3 are known as the first–order Karush-Kuhn-Tucker (KKT) sufficient conditions for optimality respectively. A point $\boldsymbol{x}$ that satisfies Condition 3 is typically called a KKT point (Nocedal and Wright, 2006). Finding multiple solutions of the NLP can be reduced to finding solutions that satisfy Condition 3, while following descent directions and avoiding saddle points.

3.2 Deflation

Deflation is a technique that systematically modifies a nonlinear problem to ensure that Newton's method does not converge to a known root, allowing unknown roots to be discovered from the same initial guess (Farrell et al, 2016).

3.2.1 Deflation for solving nonlinear equations

Farrell et al (2016) proved that under certain conditions, solving a system of n nonlinear equations:

$$\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$$

starting from the same initial solution $\boldsymbol{x}_0$ can converge to multiple locally optimal solutions by applying a deflation operator whenever a solution is found. Let the first solution found be $\boldsymbol{x}_1$. The deflated nonlinear system of equations is given by:

$$\boldsymbol{M}(\boldsymbol{x};\ \boldsymbol{x}_1)\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$$

where $\boldsymbol{M}(\boldsymbol{x};\ \boldsymbol{x}_1)$ is the deflation operator defined as:

$$\boldsymbol{M}(\boldsymbol{x};\ \boldsymbol{x}_1) = m(\boldsymbol{x};\ \boldsymbol{x}_1)\mathcal{I}$$
$$m(\boldsymbol{x};\ \boldsymbol{x}_1) = ||\boldsymbol{x} - \boldsymbol{x}_1||^{-p} + \sigma$$

The power p controls the rate of blow-up as $\boldsymbol{x}$ approches $\boldsymbol{x_1}$ and $\mathcal{I}$ is the $n \times n$ identity matrix. The shift parameter σ is used to ensure that the deflated system converges to the original $\boldsymbol{F}(\boldsymbol{x})$ as $||\boldsymbol{x} - \boldsymbol{x}_1|| \to \infty$ (Farrell et al, 2015).

After solving the system once, one can solve the deflated system and obtain a new solution $\boldsymbol{x}_2$. One of the conditions required to prove the convergence of the proposed algorithm to a different solution $\boldsymbol{x}_2$ is:

$$\lim_{\boldsymbol{x} \to \boldsymbol{x}_1} ||\boldsymbol{M}(\boldsymbol{x};\ \boldsymbol{x}_1)\boldsymbol{F}(\boldsymbol{x})|| > 0 \tag{5}$$

To deflate away from multiple found solutions, the original method proposes multiplying the operators and solve

$$\prod_{k=1}^{\tilde{K}} \boldsymbol{M}(\boldsymbol{x};\ \boldsymbol{x}_k)\, \boldsymbol{F}(\boldsymbol{x}) = 0$$

where $\tilde{K}$ is the number of found solutions. To improve numerical stability, summation instead of multiplication can also be used.

3.2.2 Deflation for NLP optimization

To apply deflation to finding multiple solutions of an NLP, one can write most of Condition 3 as $\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$ (see the Appendix for more detailed derivations) and solve for different solutions of the deflated system:

$$\boldsymbol{M}(\boldsymbol{x};\ \boldsymbol{x}_1)\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$$

while ensuring a descent direction is taken at every step. However, solving this nonlinear system of equations using Newton-like algorithms requires evaluating the Jacobian of the residual of the deflated nonlinear system of equations. One of the most popular existing implementations of the primal-dual interior point algorithm in IPOPT (Wächter and Biegler, 2006) assumes the Jacobian of the top block of $\boldsymbol{F}$ to be symmetric (since it is the Hessian of the Lagrangian). This makes it difficult to reuse IPOPT with deflation directly. Arguably, this is also the main motivation for Papadopoulos et al (2021) to develop a specialized primal-dual interior point algorithm from scratch for use with deflation. For more details on the derivations relevant to this discussion, see Appendix Section A.

4 Method

In this section, the use of deflation and the choice of the optimization algorithm to use will be decoupled by making use of a *deflation constraint* that only requires a change in the formulation of the optimization problem without specifying the algorithm.

4.1 Formulating deflation as a constraint

Instead of deflating the optimality conditions, one can create a new constraint and variable that enforce the same deflation effect. Let $y \geq 0$ be a new scalar variable. One possible constraint to add is:

$$m(\boldsymbol{x};\ \boldsymbol{x}_1) = ||\boldsymbol{x} - \boldsymbol{x}_1||^{-p} + \sigma \leq y \tag{6}$$

If after adding this constraint, the optimal solution $\boldsymbol{x}_2$ obtained has a finite y in exact arithmetic, then $\boldsymbol{x}_1 \neq \boldsymbol{x}_2$ since $\lim_{\boldsymbol{x} \to \boldsymbol{x}_1} m(\boldsymbol{x};\ \boldsymbol{x}_1) = \infty$. Furthermore, it can be shown that every KKT point of the new NLP with a finite y is a KKT point of the original formulation.

Lemma 4.1. *If $(\boldsymbol{x}^*, y^*)$ is a regular KKT point to the following NLP:*

$$\begin{aligned} &\underset{\boldsymbol{x}, y}{\text{minimize}} \quad f(\boldsymbol{x}) \\ &\text{subject to} \\ &\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}, \\ &\boldsymbol{m}(\boldsymbol{x};\ \boldsymbol{x}_1) \leq y, \\ &\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u} \end{aligned} \tag{7}$$

for a finite y^*, and $\boldsymbol{m}$ is bounded from below, then $\boldsymbol{x}^*$ is a regular KKT point to problem 1 and $\boldsymbol{x}^* \neq \boldsymbol{x}_1$.

Conversely, if $\boldsymbol{x}^* \neq \boldsymbol{x}_1$ is a regular KKT point to problem 1, there exists y^* such that $(\boldsymbol{x}^*, y^*)$ is a regular KKT point to the above NLP where m is bounded from below.

Proof Let the Lagrangian multipliers of the equality constraints be $\boldsymbol{\lambda}$, $\boldsymbol{z}_-$ be those of the $\geq$ bound constraints, and $\boldsymbol{z}_+$ be those of the $\leq$ bound constraints. Let the additional Lagrangian multiplier of the deflation constraint be η. The stationary conditions of problem 7 are:

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} + \boldsymbol{z}_+ + \boldsymbol{z}_- + \eta \nabla_{\boldsymbol{x}} m(\boldsymbol{x};\ \boldsymbol{x}_1) = \boldsymbol{0}$$
$$\eta = 0$$

Since η will be 0 at any KKT point, the stationarity conditions of problem 1 will be satisfied:

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} + \boldsymbol{z}_+ - \boldsymbol{z}_- = \boldsymbol{0}$$

Additionally, the complementarity condition of the deflation constraint and the dual feasibility constraint $\eta \geq 0$ are trivially satisfied at $\eta = 0$. Since the constraints of problem 1 are a subset of the constraints of problem 7, $\boldsymbol{x}^*$ must be feasible to the original problem and the complementarity conditions of those constraints must be satisfied.

Given that y^* is finite, $(\boldsymbol{x}^*, y^*)$ is feasible to the deflation constraint in problem 7 and m is bounded from below, then $m(\boldsymbol{x}^*;\ \boldsymbol{x}_1)$ must be finite which implies that $\boldsymbol{x}^* \neq \boldsymbol{x}_1$. This completes the first part of the proof.

To prove the converse, consider the stationarity conditions of the original problem:

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} + \boldsymbol{z}_+ - \boldsymbol{z}_- = \boldsymbol{0}$$

Setting $y^* = m(\boldsymbol{x}^*, \boldsymbol{x}_1)$ and the Lagrangian multiplier $\eta = 0$ would satisfy the deflation constraint, its complimentarity slackness condition and the stationarity conditions of the deflation problem. This completes the proof. □

Note that the proof above can be trivially generalized to inequality-constrained and even conic-constrained nonlinear programs. Therefore, this deflation constraint approach is a completely generic and noninvasive way to use deflation in optimization.

Since the deflation constraint approach only requires a change in the optimization formulation rather than the optimization routine, any KKT seeking nonlinear programming algorithm, for example, the method of moving asymptotes (Svanberg, 1987, 2002) or the augmented Lagrangian algorithm (Bertsekas, 1996) can be used to solve the deflated formulation. This is a much more generic and simpler way to use deflation than to deflate all of the optimality conditions.

4.2 Alternative deflation constraint

One can also avoid the introduction of an additional variable y replacing it with a large finite constant M.

Lemma 4.2. *If $(\boldsymbol{x}^*)$ is a regular KKT point to the following NLP:*

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) \\ &\text{subject to} \\ &\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}, \\ &\boldsymbol{m}(\boldsymbol{x};\ \boldsymbol{x}_1) \leq M, \\ &\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u} \end{aligned} \tag{9}$$

for some finite constant M, the constraint $\boldsymbol{m}(\boldsymbol{x};\ \boldsymbol{x}_1) \leq M$ is satisfied at a strict inequality, and $\boldsymbol{m}$ is bounded from below, then $\boldsymbol{x}^*$ is a regular KKT point to problem 1 and $\boldsymbol{x}^* \neq \boldsymbol{x}_1$.

Proof Let the Lagrangian multipliers of the equality constraints be $\boldsymbol{\lambda}$, $\boldsymbol{z}_-$ be those of the $\geq$ bound constraints, and $\boldsymbol{z}_+$ be those of the $\leq$ bound constraints. Let the additional Lagrangian multiplier of the deflation constraint be η. The stationary conditions of problem 9 are:

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} + \boldsymbol{z}_+ - \boldsymbol{z}_- + \eta \nabla_{\boldsymbol{x}} m(\boldsymbol{x};\ \boldsymbol{x}_1) = \boldsymbol{0}$$

By the complimentarity slackness conditions, since the deflation constraint is satisfied at a strict inequality, η must be equal to 0 at any KKT point. The rest of the proof is identical to the proof of Lemma 4.1 □

4.3 Simple deflation for nonlinear systems

Much like in optimization, the following deflation equality constraint can be added to a nonlinear system of equations:

$$m(\boldsymbol{x};\ \boldsymbol{x}_1) = y \tag{11}$$

solving for both $\boldsymbol{x}$ and y. It is trivial to see that if the algorithm converges to a solution $(\boldsymbol{x}^*, y^*)$ with a finite y^*, then $\boldsymbol{x}^*$ will satisfy the original set of equations and $\boldsymbol{x}^* \neq \boldsymbol{x}_1$.

4.4 Deflating multiple intermediate solutions

In Papadopoulos et al (2021), the deflation operator was used to deflate away from the intermediate solutions of the barrier subproblems. This is a nice way to automatically adapt the deflation power by ensuring that we deflate away from the entire critical path of the interior point optimizer the next time the system is solved. It can also be used to provide accelerated convergence by encouraging solutions to intermediate barrier subproblems to be more different from the previous barrier subproblems. The same approach can be used in any nonlinear programming algorithm by using a manual callback and adding new "known solutions" to the deflation operator for every fixed number of iterations.

4.5 Deflation constraint is a minimum distance constraint

Assuming finite non-zero $\boldsymbol{m}(\boldsymbol{x};\ \boldsymbol{x}_1) > 0$ and $y > \sigma$, the deflation constraint in Equation (7) is equivalent to the following distance constraint :

$$||\boldsymbol{x} - \boldsymbol{x}_1||^p \geq z \tag{12}$$

where $z = \frac{1}{y-\sigma}$. It can be seen more easily that the deflation technique simply imposes a constraint on the proximity to the already found solutions. If the NLP optimizer approaches $z = 0$ or $y = \infty$, then the deflation operator may not be swaying the solution sufficiently away from known solutions. When using the fixed, large-M formulation in Equation (9), instead of y tending to infinity, the deflation constraint will be satisfied in equality when deflation fails to make the algorithm converge to a different solution. The main hyperparameters that can be tuned in the deflation constraint are the distance measure used, the power p and the shift parameter σ to ensure convergence to a different solution.

4.6 Different distance measures

One can change the distance measure used in the deflation function to more interesting choices than a simple power of a 2-norm. The following are different ways to change the distance measure to achieve different desired effects:

1. Use a ℓ_q norm distance measure instead of a power p of a ℓ_2 norm, or use a power p of a ℓ_q norm distance measure.
2. Use a positive semi-definite weight matrix $\boldsymbol{Q}$ and define the distance as the generalized ℓ_2 norm distance: $(\boldsymbol{x} - \boldsymbol{x}_1)^T \boldsymbol{Q} (\boldsymbol{x} - \boldsymbol{x}_1)$. If $\boldsymbol{Q}$ is diagonal, it can be used to give different weights to different deviation terms.
3. The distance measure can detect symmetries in the solution space if 2 solutions are numerically

different but practically identical. In topology optimization, for example, it may make more sense to define the distance measure in the pseudo-densities $\boldsymbol{\rho}$ space after applying density filtering, interpolation and Heaviside projection to the solution $\boldsymbol{x}$, rather than calculating distance in the $\boldsymbol{x}$ space. This more accurately reflects the desire of the optimizer to obtain different designs rather than different, symmetric representations of the same design.

4. In topology optimization also, the number of design variables can be extremely large with many similar looking designs that have slightly different numerical values. A particular modification to the distance measure can therefore be useful to workaround this curse of dimensionality problem. Let $\sigma = 0$ and consider the following deflation function:

$$m(\boldsymbol{x};\ \boldsymbol{x}_1) = \max(||\boldsymbol{x} - \boldsymbol{x}_1|| - r, 0)^{-p} \qquad (13)$$

   This deflation function treats all solutions in the hyper-ball of radius r around $\boldsymbol{x}_1$ as identical to $\boldsymbol{x}_1$. This function is smooth and twice differentiable for all $\boldsymbol{x} \in \{\boldsymbol{x} : ||\boldsymbol{x} - \boldsymbol{x}_1|| > r\}$. A KKT solution with a finite y (in exact arithmetic) can be similarly shown to be equivalent to finding a new optimal solution outside the hyper-ball. This new parameter r can be useful for high-dimensional problems in topology optimization where two designs can be numerically different but visually and practically identical. Another side advantage of using $r > 0$ is that even if deflation fails to push the optimization algorithm away from $\boldsymbol{x}_1$, the new solution obtained is guaranteed to be outside the hyper-ball.
5. In variational inference, the Kullback-Leibler (K-L) divergence can be used to deflate away from known distributions that locally optimize the K-L divergence to the posterior distribution.
6. In deep learning, a distance measure between the neuron values can be used instead of weights and biases to account for symmetries in the weights.

4.7 Potential problems with deflation

Although deflation has proven rather successful in a number of root finding and optimization applications, it is not free of limitations which need to be addressed or acknowledged when implementing or using the algorithm. The following are some common problems associated with the proposed deflation constraint, most of which are also concerns when using the traditional deflation method.

1. The deflation effect may not be strong enough where the algorithm can still approach $\boldsymbol{x}_1$, asymptotically increasing y to ∞. This is similar to what can happen in a classic deflation formulation when the condition in Equation (5) is violated.
2. The optimization is not done in exact arithmetic but rather up to machine precision. This can make the algorithm converge to a solution with a finite but large value value of y^* instead of overflowing to ∞. Therefore, the finiteness of y^* may not be a good enough indication that the algorithm has converged to a new solution other than $\boldsymbol{x}_1$.
3. Deflation guarantees that if convergence happens, the solution will be different. However, it does not guarantee that convergence will happen to begin with. Even more so, deflating away from one solution is likely to push the optimizer away from other nearby locally optimal solutions if they exist. This can make it particularly challenging to fine-tune the hyperparameters of the algorithm. However, arguably this can also be a desirable effect of deflation since it means that more diverse solutions are naturally more likely to be output by the algorithm.
4. Careful selection of the power p and other hyperparameters used in the chosen distance measure is required.

5 Results and Discussion

In this section, a number of applications of deflation and non-convex optimization are showcased from machine learning and topology optimization. The main highlight of this section is that known popular algorithms were used or minimally modified to solve the deflation sub-problems. In all of the examples, the same initial solution was used in the deflation sub-problems to showcase the efficacy of the approach proposed. The implementation and examples, including detailed hyperparameter settings, can be found in the supplementary code[1].

5.1 Classic variational inference on mixture of Gaussian distributions

In this section, an example of the use of deflation in variational inference will be demonstrated. Variational inference is a computational method that approximates complex probabilistic distributions by simpler, tractable distributions, facilitating scalable and efficient inference for large datasets (chap. 10, Bishop and Nasrabadi (2006)).

In this example, we consider a standard variational inference test problem that uses a Gaussian distribution $q(x;\ \mu, \sigma)$ with mean μ and standard deviation σ to approximate a mixture model $p(x)$ of 10 univariate Gaussian distributions $g_i(x)$ for $i \in 1 \dots 10$, where $p(x) = \frac{1}{10} \sum_{i=1}^{10} g_i(x)$. In this problem, the g_is are given and the goal is to find $\boldsymbol{\theta} = (\mu, \sigma)$ which minimize the Kullback-Leibler divergence between the two distributions p and q:

$$\begin{aligned} L(\boldsymbol{\theta}) &= \text{ELBO}(\mu, \sigma) \\ &= \int \log \frac{p(x)}{q(x;\ \mu, \sigma)} q(x;\ \mu, \sigma)\, dx \end{aligned} \tag{14}$$

In our implementation, the automatic differentiation variational inference algorithm (ADVI) (Kucukelbir et al, 2015) was used together with the decayed ADAGrad stochastic optimization algorithm (Duchi et al, 2011) from the AdvancedVI.jl [2] package to minimize a stochastic estimator of the K-L divergence. Distributions.jl (Lin et al, 2022)[3] was used to define the mixture of Gaussians, the variational family, and the estimator of the K-L divergence objective function. The deflated variational approximation problem can be formulated using the loss function $L(\boldsymbol{\theta})$ as follows:

$$\begin{aligned} \underset{\boldsymbol{\theta}, y}{\text{minimize}} \quad & L(\boldsymbol{\theta}) \\ \text{subject to} \quad & \sum_{k=1}^{\tilde{K}} \boldsymbol{m}(\boldsymbol{\theta};\ \boldsymbol{\theta}_k) \leq y \end{aligned} \tag{15}$$

where $\boldsymbol{\theta}$ is the vector of parameters (μ, σ) of the variational family q and $\boldsymbol{\theta}_k$ is the previously found local minimizer of the loss function from subproblem k, and $\tilde{K}$ is the total number of found solutions so far. $\boldsymbol{m}$ and y are the deflation function and variable respectively as described in the previous sections.

Since the objective is a stochastic estimator of the K-L divergence and the constraint is non-convex, a log-barrier approach (Fiacco and Mccormick, 1968) is used to transform the stochastic constrained problem to an unconstrained one. The log-barrier formulation becomes:

$$\underset{\boldsymbol{\theta}, y}{\text{minimize}} \quad L(\boldsymbol{\theta}) - r \log \left(y - \sum_{k=1}^{\tilde{K}} \boldsymbol{m}(\boldsymbol{\theta};\ \boldsymbol{\theta}_k) \right) \tag{16}$$

where r is defined as a decaying coefficient approaching 0. The above problem was solved 10 times from the same initial solution $\boldsymbol{\theta}_0$ each time converging to different Gaussian approximations and appending it to the list of found solutions. The distance function used between solutions in the deflation function was the analytic K-L divergence between the 2 Gaussian distributions offset by a radius r. Let $d(\boldsymbol{\theta}) = \mathcal{N}(\mu = \boldsymbol{\theta}[1], \sigma = \exp(\boldsymbol{\theta}[2]))$ be the Gaussian distribution obtained from solution $\boldsymbol{\theta}$. The K-L divergence based deflation function m used was therefore:

$$m(\boldsymbol{\theta};\ \boldsymbol{\theta}_k) = \max\left(\text{div}\Big(d(\boldsymbol{\theta}), d(\boldsymbol{\theta}_k)\Big) - 1, 0 \right)^{-3} \tag{17}$$

where div is the K-L divergence between two Gaussian distributions. Note that the minimum radius of 1 was used to enforce convergence to a different distribution even if the convergence condition (Equation (5)) of deflation were violated. The optimal Gaussian distributions' obtained and the target distribution's probability density functions are shown in figure 1. The results indicate that deflation was successful at generating reasonable mode-seeking approximations of the target mixture of Gaussians.

[1] https://github.com/JuliaTopOpt/deflation_examples
[2] https://github.com/TuringLang/AdvancedVI.jl
[3] https://github.com/JuliaStats/Distributions.jl

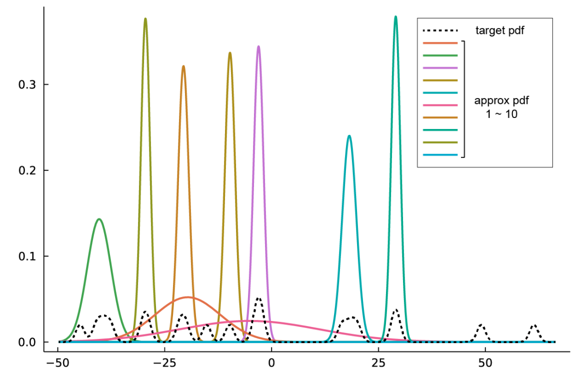

Fig. 1: The figure shows the target probability density function (pdf) of the mixture of Gaussians and the pdf curves of the multiple Gaussian approximations obtained by solving the deflated variational inference problem 10 times from the same initial Gaussian solution $\mathcal{N}(\mu = 0, \sigma = \exp(5.0))$.

5.2 Pathfinder algorithm for variational inference

In this section, we demonstrate how to combine our deflation formulation with a recently proposed variational inference algorithm called the pathfinder algorithm (Zhang et al, 2021) to approximate a target distribution $p(x)$ with a Gaussian mixture approximation $q(x)$. The same mixture of k Gaussian $p(x)$ was used in this experiment, for $k = 2, 4, 6$ and 8.

The pathfinder algorithm relies on multi-start optimization of the log probability of the target distribution $p(x)$ to find maximizers x^*. The trajectory of intermediate solutions and gradients from the optimization then get used to construct a Gaussian approximation per trajectory. Re-starting the optimization from a different initial point can result in a different optimal solution and trajectory, leading to a potentially different local Gaussian approximation. These Gaussian approximations are then combined and weighted in a mixture of Gaussians $q(x)$ that is used to approximate the target distribution $p(x)$. The original algorithm relies on restarting the optimization from random initial solutions. In this work, we use deflation instead, starting the optimization from the same initial point every time. The readers are referred to Zhang et al (2021) for more details of the algorithm.

The Pathfinder.jl (Axen, 2021)[4] package was adapted to use IPOPT (Wächter and Biegler, 2006) as an optimizer as wrapped in the Nonconvex.jl [5] package. Instead of random restarts, the same initial solution of 0.0 was used in all the deflation-based pathfinder sub-problems. The target (blue) and approximate (orange) mixtures of Gaussians are shown in figure 2. The results look good given that all the optimization sub-problems were started from the same initial solution ($x = 0$), this is a fairly positive result.

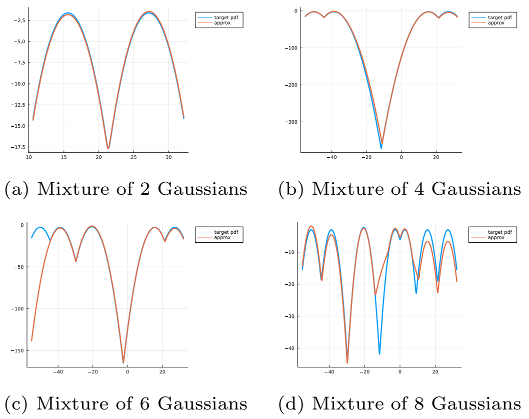

Fig. 2: Target (blue) and approximate (red) mixtures of Gaussian probability density functions using the same initial solution of 0.0 in all of the deflation-based sub-problems in the pathfinder algorithm.

5.3 Topology optimization - volume constrained compliance minimization

The volume-constrained compliance minimization problem from topology optimization (TO) seeks to find designs for physical structures that are as stiff as possible (i.e., least compliant) with respect to known boundary conditions and loading forces while adhering to a given material demand. It has been well studied and widely applied in mechanical, aerospace, and architectural engineering (Bendsoe and Sigmund, 2003). However, because TO problems are high-dimensional, partial differential equation (PDE) constrained,

[4] https://github.com/mlcolab/Pathfinder.jl
[5] https://github.com/JuliaNonconvex/Nonconvex.jl

and non-convex, finding multiple local minima that are visually different has been challenging and rarely studied, despite its practical values. A compliance-minimizing, volume-constrained TO problem can be formulated as:

$$\begin{aligned}
&\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad \boldsymbol{f}\cdot\boldsymbol{u}\\
&\text{subject to}\\
&\boldsymbol{K}(\boldsymbol{x})\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{f},\\
&\sum_{i=1}^{n} x_i V_i \le \tilde{V},\\
&\boldsymbol{0} \le \boldsymbol{x} \le \boldsymbol{1}
\end{aligned} \tag{18}$$

where $\boldsymbol{x}$ is the pseudo-density of cells in the domain, $\boldsymbol{f}$ the given load vector, $\boldsymbol{K}(\boldsymbol{x})$ the stiffness matrix from the finite-element discretization, V_i the volume of grid element i, and $\tilde{V}$ the user-specified total volume bound. When $\boldsymbol{x}_i = 0$, the corresponding cell is removed and when $\boldsymbol{x}_i = 1$ the cell is kept.

We apply our deflation technique by adding an extra constraint to problem 18, using the distance measure that encourages visually different designs mentioned in Section 4.6:

$$\sum_{k=1}^{\tilde{K}} m(\boldsymbol{x};\ \boldsymbol{x}_k) = \sum_{k=1}^{\tilde{K}} \max(\|\boldsymbol{x}-\boldsymbol{x}_k\| - r, 0)^{-p} \le y \tag{19}$$

where $\tilde{K}$ is the number of found local minima from previous deflation iterations. In our experiments, $p = 4$, $r = 20$, $0 \le y \le 100$.

Here, two classic topology optimization benchmark problems are tested: a continuum MBB beam domain of dimension 120×40 and a cantilever truss domain of dimension 40×10 (Figure 3). TopOpt.jl[6] is used for the implementation of problem modeling, finite element analysis, and density filters. We use the widely used Method of Moving Asymptotes (MMA) algorithm (Svanberg, 1987) to solve both the undeflated and deflated problems. Averaged runtime results are presented in Table 1 and per-deflation-iteration runtime statistics are presented in Figure 6 and Figure 7. Figure 4 and Figure 5 visualize the results of 20 iterations of deflation in both domains. These results show that the proposed deflation formulation generates visually different, near-optimal designs deterministically, all starting from the same initial guess. The runtime results suggest that there are no significant changes in running time due to the addition of a deflation constraint to an already constrained optimization problem.

[6] https://github.com/JuliaTopOpt/TopOpt.jl

Table 1: Deflated topology optimization runtime. *: deflated problem runtime is averaged over all 20 deflation iterations.

TopOpt Problems	$\boldsymbol{x}$ dim	undeflated	deflated*
Continuum (Fig. 4)	4800	121s	122s
Truss (Fig. 5)	3608	1.9s	1.6s

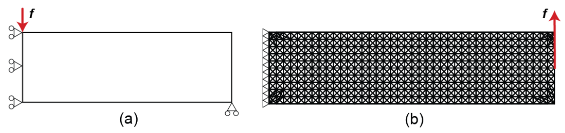

Fig. 3: Topology optimization domains: (a) half MBB beam continuum domain; (b) cantilevering truss domain.

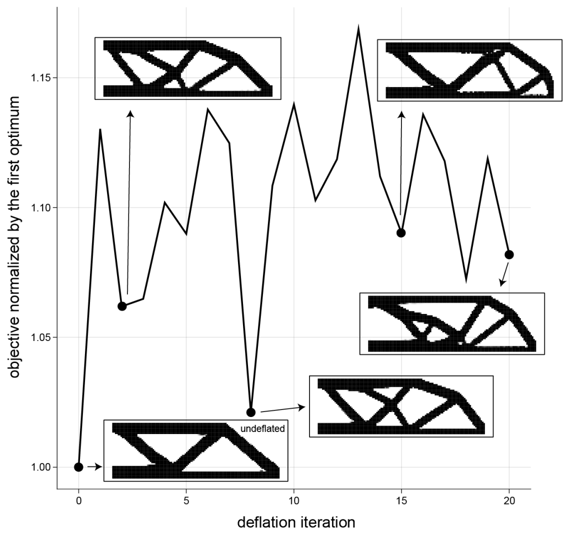

Fig. 4: Deflated results of the continuum half MBB beam problem. Iteration 0 is the solution found by solving the undeflated problem.

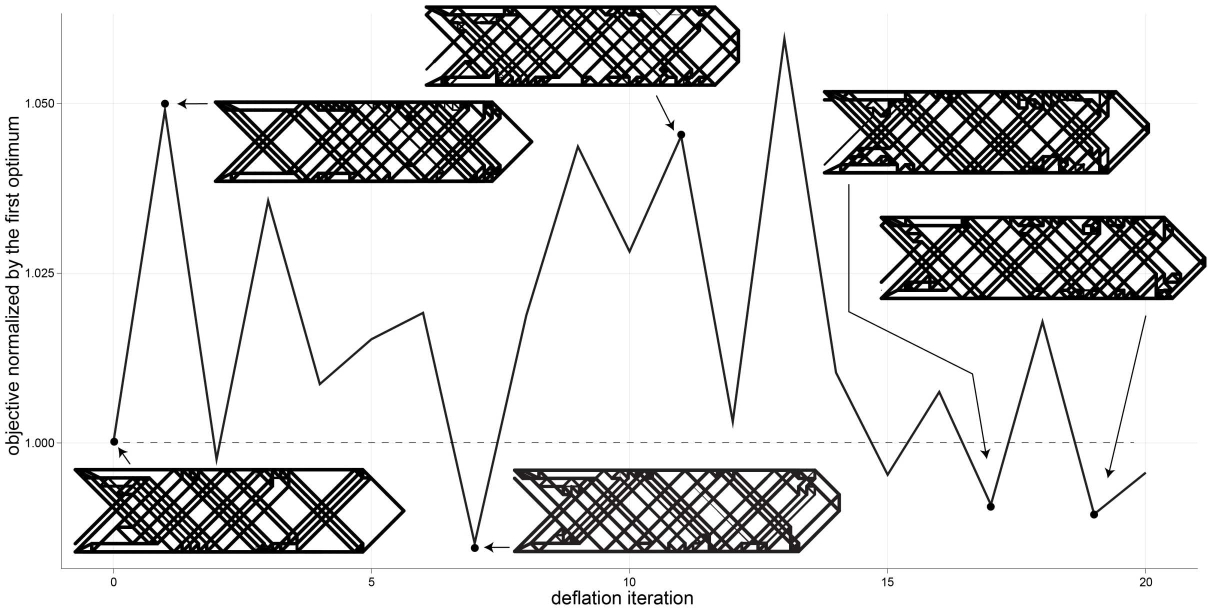

Fig. 5: Deflated results of the discrete, cantilevering truss problem. Iteration 0 is the solution found by solving the undeflated problem.

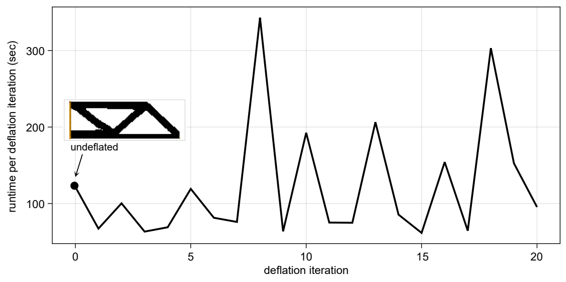

Fig. 6: Runtime per deflation iteration for the continuum half MBB beam problem in Figure 4.

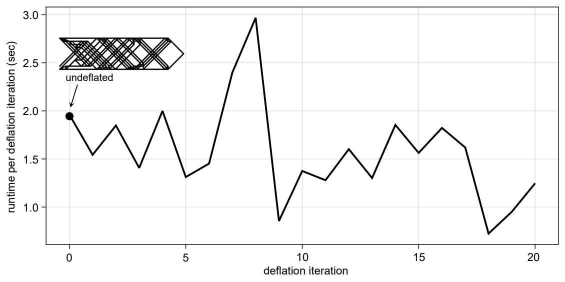

Fig. 7: Runtime per deflation iteration for the discrete, cantilevering truss problem in Figure 5.

6 Conclusion and future work

In this work, a new way of using deflation was proposed to find diverse solutions to non-convex optimization problems. With the proposed problem reformulation, the deflation technique can be easily applied to any non-convex problem using existing, off-the-shelf optimizers. Promising results of applying the proposed technique in solving problems from topology optimization and variational inference are presented. We hope that our work can enable applications of this simple, yet powerful idea of deflation to a broader set of problems where multiple local optimal solutions are required.

In the future, we hope to make use of deflation-based optimization to enhance optimization-based machine learning algorithms such as maximum likelihood estimation and to use it for model-based design of experiments in clinical trials to explore multiple possible designs.

Declarations

The authors have no relevant financial or non-financial interests to disclose.

Acknowledgements

This work was not part of any particular grant. The authors broadly acknowledge the support of their respective organizations, Pumas-AI Inc., MIT, and ETH Zürich. Yijiang Huang acknowledges the generous support of the ETH Postdoctoral Fellowship.

7 Replication of Results

The implementation and examples, including detailed hyperparameter settings, can be found in the supplementary code in https://github.com/JuliaTopOpt/deflation_examples.

References

Arnoud A, Guvenen F, Kleineberg T (2019) Benchmarking Global Optimizers. Tech. Rep. w26340, National Bureau of Economic Research, Cambridge, MA, https://doi.org/10.3386/w26340

Axen S (2021) Pathfinder.jl. https://doi.org/10.5281/zenodo.5914976, URL https://doi.org/10.5281/zenodo.5914976

Barron C, Gomez S (1991) The exponential tunneling method

Belotti P, Lee J, Liberti L, et al (2009) Branching and bounds tightening techniques for non-convex MINLP. Optimization Methods and Software 24(4-5):597–634

Bendsoe MP, Sigmund O (2003) Topology optimization: theory, methods, and applications. Springer Science & Business Media

Bertsekas DP (1996) Constrained Optimization and Lagrange Multiplier Methods. Athena Scientific

Bishop CM, Nasrabadi NM (2006) Pattern recognition and machine learning, vol 4. Springer

Brown KM, Gearhart WB (1971) Deflation techniques for the calculation of further solutions of a nonlinear system. Numerische Mathematik 16(4):334–342

Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research 12(61):2121–2159

Falkner S, Klein A, Hutter F (2018) Bohb: Robust and efficient hyperparameter optimization at scale. 1807.01774

Farrell PE, Birkisson A, Funke SW (2015) Deflation techniques for finding distinct solutions of nonlinear partial differential equations. SIAM Journal on Scientific Computing 37(4):A2026–A2045

Farrell PE, Beentjes CHL, Birkisson A (2016) The computation of disconnected bifurcation diagrams. arXiv:160300809 [math] ArXiv: 1603.00809

Farrell PE, Croci M, Surowiec TM (2020) Deflation for semismooth equations. Optimization Methods and Software 35(6):1248–1271

Fiacco A, Mccormick G (1968) Nonlinear Programming: Sequential Unconstrained Minimization Techniques. John Wiley, New York

Gablonsky JM, Kelley CT (2001) A Locally-Biased form of the DIRECT Algorithm. Journal of Global Optimization 21:27–37

Gecode Team (2006) Gecode: Generic constraint development environment. Available from `http://www.gecode.org`

Gendreau M, Potvin JY (eds) (2010) Handbook of metaheuristics, 2nd edn. Springer, New York, NY, USA

Gomez S, Levy AV (1982) The tunnelling method for solving the constrained global optimization problem with several non-connected feasible regions. Numerical analysis p 34–47

Gomez S, del Castillo N, Castellanos L, et al (2003) The parallel tunneling method. Parallel Computing p 523–533

Gudmundsson S (1998) Parallel Global Optimization, M.Sc. Thesis, IMM, Technical University of Denmark

Jones DR, Perttunen CD, Stuckman BE (1993) Lipschitzian optimization without the Lipschitz constant. Journal of Optimization Theory and Applications 79(1):157–181. https://doi.org/10.1007/BF00941892

K. Madsen SZ, Zilinskas A (1998) Global Optimization using Branch-and-Bound

Kaelo P, Ali MM (2006) Some Variants of the Controlled Random Search Algorithm for Global Optimization. Journal of Optimization Theory and Applications 130(2):253–264. https://doi.org/10.1007/s10957-006-9101-0

Kucherenko S, Sytsko Y (2005) Application of Deterministic Low-Discrepancy Sequences in Global Optimization. Computational Optimization and Applications 30(3):297–318. https://doi.org/10.1007/s10589-005-4615-1

Kucukelbir A, Ranganath R, Gelman A, et al (2015) Automatic variational inference in stan. In: Cortes C, Lawrence N, Lee D, et al (eds) Advances in Neural Information Processing Systems, vol 28. Curran Associates, Inc.

Levy AV, Gomez S (1981) The tunneling method applied to global optimization. Numerical optimization p 213–244

Li L, Jamieson K, DeSalvo G, et al (2018) Hyperband: A novel bandit-based approach to hyperparameter optimization. 1603.06560

Lin D, Byrne S, White JM, et al (2022) Juliastats/distributions.jl: v0.25.79. https://doi.org/10.5281/zenodo.7343446, URL https://doi.org/10.5281/zenodo.7343446

McKay M. D. CWJBeckman R. J. (1979) A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21(2):239–245. https://doi.org/10.2307/1268522

Moscato P, Cotta C (2010) A Modern Introduction to Memetic Algorithms, Springer US, Boston, MA, pp 141–183. https://doi.org/10.1007/978-1-4419-1665-5_6

Nagarajan H, Lu M, Yamangil E, et al (2016) Tightening McCormick relaxations for nonlinear programs via dynamic multivariate partitioning. In: International Conference on Principles and Practice of Constraint Programming, Springer, pp 369–387, https://doi.org/10.1007/978-3-319-44953-1_24

Nagarajan H, Lu M, Wang S, et al (2019) An adaptive, multivariate partitioning algorithm for global optimization of nonconvex programs. Journal of Global Optimization https://doi.org/10.1007/s10898-018-00734-1

Nocedal J, Wright S (2006) Numerical optimization. Springer Science & Business Media

Papadopoulos IPA, Farrell PE, Surowiec TM (2021) Computing multiple solutions of topology optimization problems. arXiv:200411797 [cs, math] ArXiv: 2004.11797

Patrick E. Farrell MC, Surowiec TM (2020) Deflation for semismooth equations. Optimization Methods and Software 35(6):1248–1271. https://doi.org/10.1080/10556788.2019.1613655, URL https://doi.org/10.1080/10556788.2019.1613655, https://arxiv.org/abs/https://doi.org/10.1080/10556788.2019.1613655

Ratschek H, Rokne J (2007) New computer methods for global optimization

Rinnooy Kan AHG, Timmer GT (1987) Stochastic global optimization methods part I: Clustering methods. Mathematical Programming 39(1):27–56. https://doi.org/10.1007/BF02592070

Sahinidis NV (2017) BARON 21.1.13: Global Optimization of Mixed-Integer Nonlinear Programs, *User's Manual*

Sinha A, Malo P, Deb K (2018) A review on bilevel optimization: From classical to evolutionary approaches and applications. IEEE Transactions on Evolutionary Computation 22(2):276–295. https://doi.org/10.1109/TEVC.2017.2712906

Svanberg K (1987) The method of moving asymptotes - a new method for structural optimization. International Journal for Numerical Methods in Engineering 24(2):359–373

Svanberg K (2002) A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations. SIAM Journal on Optimization 12(2):555–573

Tawarmalani M, Sahinidis NV (2005) A polyhedral branch-and-cut approach to global optimization. Mathematical Programming 103(2):225–249. https://doi.org/10.1007/s10107-005-0581-8

Vigerske S, Gleixner A (2018) Scip: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. Optimization Methods and Software 33(3):563–593. https://doi.org/10.1080/10556788.2017.1335312, https://arxiv.org/abs/https://doi.org/10.1080/10556788.2017.1335312

Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical programming 106(1):25–57

Wilhelm ME, Stuber MD (2020) Eago.jl: easy advanced global optimization in julia. Optimization Methods and Software pp 1–26. https://doi.org/10.1080/10556788.2020.1786566, https://arxiv.org/abs/https://doi.org/10.1080/10556788.2020.1786566

Wright S (1997) Primal-dual interior-point methods. https://doi.org/10.1137/1.9781611971453, https://epubs.siam.org/doi/pdf/10.1137/1.9781611971453

Zhang L, Carpenter B, Gelman A, et al (2021) Pathfinder: Parallel quasi-newton variational inference. arXiv:2108.03782

Zhang S, Norato JA (2018) Finding better local optima in topology optimization via tunneling. In: Volume 2B: 44th Design Automation Conference. American Society of Mechanical Engineers, Quebec City, Quebec, Canada, p V02BT03A014, https://doi.org/10.1115/DETC2018-86116

Appendix A

In this section, derivations for the deflated KKT system of the barrier sub-problem used by the interior point method (Wright, 1997) as implemented in the IPOPT software (Wächter and Biegler, 2006) are presented.

A.1 KKT system of the barrier problem

In IPOPT, a log-barrier method is used to ensure that $\boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u}$ remains satisfied at every intermediate solution if the initial solution is within the bounds. The barrier function is defined as:

$$\mathcal{B}_\mu(\boldsymbol{x}, \boldsymbol{l}, \boldsymbol{u}) = \\ -\mu\Big(\sum_i \log(x_i - l_i) + \sum_i \log(u_i - x_i)\Big) \tag{A1}$$

for some $\mu > 0$ which would go to ∞ if any of the decision variables approaches one of its finite bounds. This creates a barrier that prevents the optimizer from ever reaching the finite bound. The barrier sub-problem is defined as:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \phi_\mu(\boldsymbol{x}) = f(\boldsymbol{x}) + \mathcal{B}_\mu(\boldsymbol{x}, \boldsymbol{l}, \boldsymbol{u}) \\ \text{subject to} \quad & \boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0} \end{aligned}$$

The KKT stationarity condition is therefore:

$$\nabla f(\boldsymbol{x}) + \nabla \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} - \boldsymbol{z_l} - \boldsymbol{z_u} = \boldsymbol{0}$$

where $\boldsymbol{\lambda}$ is the vector Lagrangian multipliers associated with the equality constraint $\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}$, $\boldsymbol{z_l}$ is a vector whose i^{th} element is $z_{l_i} = \frac{\mu}{x_i - l_i}$ and $\boldsymbol{z_u}$ is a vector whose i^{th} element is $z_{u_i} = \frac{\mu}{u_i - x_i}$.

Additionally, let $\boldsymbol{Z_l}$ be the diagonal matrix whose diagonal is $\boldsymbol{z_l}$, $\boldsymbol{Z_u}$ be the diagonal matrix whose diagonal is $\boldsymbol{z_u}$, $\boldsymbol{X_l}$ be the diagonal matrix whose diagonal is: $\tilde{x}_{l_i} = x_i - l_i$ and $\boldsymbol{X_u}$ be the diagonal matrix whose diagonal is: $\tilde{x}_{u_i} = u_i - x_i$.

The first-order KKT sufficient conditions for optimality of the barrier problem, assuming the constraint qualifications are satisfied, can be written as:

$$\begin{aligned} \nabla f(\boldsymbol{x}) + \nabla \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} - \boldsymbol{z_l} - \boldsymbol{z_u} &= \boldsymbol{0} && \text{(A2a)} \\ \boldsymbol{c}(\boldsymbol{x}) &= \boldsymbol{0} && \text{(A2b)} \\ \boldsymbol{X_l Z_l 1} - \mu \boldsymbol{1} &= \boldsymbol{0} && \text{(A2c)} \\ \boldsymbol{X_u Z_u 1} - \mu \boldsymbol{1} &= \boldsymbol{0} && \text{(A2d)} \\ \boldsymbol{l} \le \boldsymbol{x} &\le \boldsymbol{u} && \text{(A2e)} \\ \boldsymbol{z_l} &\ge \boldsymbol{0} && \text{(A2f)} \end{aligned}$$

$$\boldsymbol{z_u} \geq \mathbf{0} \tag{A2g}$$

where $\mathbf{1}$ is a vector of ones. Conditions A2c and A2d ensure that the relationship between $\boldsymbol{X_l}$, $\boldsymbol{Z_l}$, $\boldsymbol{X_u}$ and $\boldsymbol{Z_u}$ is maintained according to the definitions of $\boldsymbol{z_l}$ and $\boldsymbol{z_u}$. In an interior point algorithm (e.g. IPOPT Wächter and Biegler (2006)), primal-dual solutions to the equality KKT conditions are found using a Newton-like method while ensuing that the inequality conditions are satisfied by projection.

In order to solve the barrier problem for a given value $\mu = \mu_j$, a damped Newton's method is usually applied to the primal-dual optimality conditions. Here we use k to denote the iteration counter for the inner iterations when solving the barrier problem. Given an iterate $(\boldsymbol{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{z}_{\boldsymbol{l},k}, \boldsymbol{z}_{\boldsymbol{u},k})$ with $\boldsymbol{l} < \boldsymbol{x}_k < \boldsymbol{u}$ and $\boldsymbol{z}_{\boldsymbol{l},k}, \boldsymbol{z}_{\boldsymbol{u},k} > \mathbf{0}$, some search directions $(\boldsymbol{d}_k^{\boldsymbol{x}}, \boldsymbol{d}_k^{\boldsymbol{\lambda}}, \boldsymbol{d}_k^{\boldsymbol{z_l}}, \boldsymbol{d}_k^{\boldsymbol{z_u}})$ are obtained using the regularized linearization of the optimality conditions (excluding the inequality conditions):

$$\begin{bmatrix} \boldsymbol{W}_k & \boldsymbol{A}_k & -\boldsymbol{I} & -\boldsymbol{I} \\ \boldsymbol{A}_k^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{Z}_{\boldsymbol{l},k} & \mathbf{0} & \boldsymbol{X}_{\boldsymbol{l},k} & \mathbf{0} \\ \boldsymbol{Z}_{\boldsymbol{u},k} & \mathbf{0} & \mathbf{0} & \boldsymbol{X}_{\boldsymbol{u},k} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^{\boldsymbol{x}} \\ \boldsymbol{d}_k^{\boldsymbol{\lambda}} \\ \boldsymbol{d}_k^{\boldsymbol{z_l}} \\ \boldsymbol{d}_k^{\boldsymbol{z_u}} \end{pmatrix} = -\begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \boldsymbol{z}_{\boldsymbol{l},k} - \boldsymbol{z}_{\boldsymbol{u},k} \\ \boldsymbol{c}(\boldsymbol{x}_k) \\ \boldsymbol{X}_{\boldsymbol{l},k} \boldsymbol{Z}_{\boldsymbol{l},k} \mathbf{1} - \mu_j \mathbf{1} \\ \boldsymbol{X}_{\boldsymbol{u},k} \boldsymbol{Z}_{\boldsymbol{u},k} \mathbf{1} - \mu_j \mathbf{1} \end{pmatrix} \tag{A3}$$

where $\boldsymbol{A}_k := \nabla \boldsymbol{c}(\boldsymbol{x}_k)^T$ and $\boldsymbol{W}_k := \nabla^2_{\boldsymbol{xx}}(f(\boldsymbol{x}_k) + \boldsymbol{c}(\boldsymbol{x}_k)^T \boldsymbol{\lambda}_k)$ is the Hessian of the Lagrangian function of the original problem. The Lagrangian terms from the bounds constraints are ignored because they don't contribute to the Hessian. When the Hessian of the Lagrangian is not available, a l-BFGS approximation (Nocedal and Wright, 2006) of the Hessian can be used instead. This changes the IPOPT algorithm from a second-order algorithm to a first-order one. And the Newton update becomes a quasi-Newton update.

Instead of solving the non-symmetric system of equations above, one can instead change the system as such:

$$\begin{bmatrix} \boldsymbol{W}_k & \boldsymbol{A}_k & -\boldsymbol{I} & -\boldsymbol{I} \\ \boldsymbol{A}_k^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{X}_{\boldsymbol{l}}^{-1} \boldsymbol{Z}_{\boldsymbol{l},k} & \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \boldsymbol{X}_{\boldsymbol{u}}^{-1} \boldsymbol{Z}_{\boldsymbol{u},k} & \mathbf{0} & \mathbf{0} & \boldsymbol{I} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^{\boldsymbol{x}} \\ \boldsymbol{d}_k^{\boldsymbol{\lambda}} \\ \boldsymbol{d}_k^{\boldsymbol{z_l}} \\ \boldsymbol{d}_k^{\boldsymbol{z_u}} \end{pmatrix} = -\begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \boldsymbol{z}_{\boldsymbol{l},k} - \boldsymbol{z}_{\boldsymbol{u},k} \\ \boldsymbol{c}(\boldsymbol{x}_k) \\ \boldsymbol{z}_{\boldsymbol{l},k} - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^{-1} \mathbf{1} \\ \boldsymbol{z}_{\boldsymbol{u},k} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^{-1} \mathbf{1} \end{pmatrix} \tag{A4}$$

Adding the third and fourth equations to the first one, the third and fourth blocks of coefficients of the first equation will be eliminated.

$$\begin{bmatrix} \boldsymbol{W}_k + \boldsymbol{\Sigma}_k & \boldsymbol{A}_k & \mathbf{0} & \mathbf{0} \\ \boldsymbol{A}_k^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{X}_{\boldsymbol{l}}^{-1} \boldsymbol{Z}_{\boldsymbol{l},k} & \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \boldsymbol{X}_{\boldsymbol{u}}^{-1} \boldsymbol{Z}_{\boldsymbol{u},k} & \mathbf{0} & \mathbf{0} & \boldsymbol{I} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^{\boldsymbol{x}} \\ \boldsymbol{d}_k^{\boldsymbol{\lambda}} \\ \boldsymbol{d}_k^{\boldsymbol{z_l}} \\ \boldsymbol{d}_k^{\boldsymbol{z_u}} \end{pmatrix} = -\begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^{-1} \mathbf{1} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^{-1} \mathbf{1} \\ \boldsymbol{c}(\boldsymbol{x}_k) \\ \boldsymbol{z}_{\boldsymbol{l},k} - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^{-1} \mathbf{1} \\ \boldsymbol{z}_{\boldsymbol{u},k} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^{-1} \mathbf{1} \end{pmatrix} \tag{A5}$$

where $\boldsymbol{\Sigma}_k = \boldsymbol{X}_{\boldsymbol{l}}^{-1} \boldsymbol{Z}_{\boldsymbol{l},k} + \boldsymbol{X}_{\boldsymbol{u}}^{-1} \boldsymbol{Z}_{\boldsymbol{u},k}$. Therefore, one can now solve for $\boldsymbol{d}_k^{\boldsymbol{x}}$ and $\boldsymbol{d}_k^{\boldsymbol{\lambda}}$ by solving the following symmetric linear system:

$$\begin{bmatrix} \boldsymbol{W}_k + \boldsymbol{\Sigma}_k & \boldsymbol{A}_k \\ \boldsymbol{A}_k^T & \mathbf{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^{\boldsymbol{x}} \\ \boldsymbol{d}_k^{\boldsymbol{\lambda}} \end{pmatrix} = -\begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^{-1} \mathbf{1} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^{-1} \mathbf{1} \\ \boldsymbol{c}(\boldsymbol{x}_k) \end{pmatrix} \tag{A6}$$

then use the value of $\boldsymbol{d}_k^{\boldsymbol{x}}$ to find $\boldsymbol{d}_k^{\boldsymbol{z_l}}$ and $\boldsymbol{d}_k^{\boldsymbol{z_u}}$ using:

$$\boldsymbol{d}_k^{\boldsymbol{z_l}} = -\boldsymbol{z}_{\boldsymbol{l},k} + \mu_j \boldsymbol{X}_{\boldsymbol{l}}^{-1} \mathbf{1} - \boldsymbol{X}_{\boldsymbol{l}}^{-1} \boldsymbol{Z}_{\boldsymbol{l},k} \boldsymbol{d}_k^{\boldsymbol{x}} \tag{A7a}$$

$$\boldsymbol{d}_k^{\boldsymbol{z_u}} = -\boldsymbol{z}_{\boldsymbol{u},k} + \mu_j \boldsymbol{X}_{\boldsymbol{u}}^{-1} \mathbf{1} - \boldsymbol{X}_{\boldsymbol{u}}^{-1} \boldsymbol{Z}_{\boldsymbol{u},k} \boldsymbol{d}_k^{\boldsymbol{x}} \tag{A7b}$$

A.2 Deflating the KKT system

If we define the RHS of Equation (A6) as:

$$\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{pmatrix} \nabla f(\boldsymbol{x}) + \boldsymbol{A}^T \boldsymbol{\lambda} - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^{-1} \mathbf{1} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^{-1} \mathbf{1} \\ \boldsymbol{c}(\boldsymbol{x}) \end{pmatrix} \tag{A8}$$

Then the goal of the primal-dual optimizer then becomes solving for $\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) = \mathbf{0}$.

If we apply the deflation operator to the entire $\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda})$, we will obtain $G(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{M}(\boldsymbol{x};\ \boldsymbol{x}_1)\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$, whose Jacobian is:

$$\nabla_{\boldsymbol{x}} \boldsymbol{G}(\boldsymbol{x}) = m(\boldsymbol{x};\ \boldsymbol{x}_1) \nabla_{\boldsymbol{x}} \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) +$$

$$\mathrm{diag}(F(\boldsymbol{x},\boldsymbol{\lambda}))\mathbf{1}(\nabla_{\boldsymbol{x}} m(\boldsymbol{x};\ \boldsymbol{x}_1))^T \tag{A9}$$

where $\nabla_{\boldsymbol{x}} m(\boldsymbol{x};\ \boldsymbol{x}_1)$ is the gradient vector of the deflation function, $\mathbf{1}$ is a column vector of 1s and $\mathrm{diag}(\boldsymbol{F}(\boldsymbol{x}))$ is a diagonal matrix with diagonal $\boldsymbol{F}(\boldsymbol{x})$. The Jacobian of this function wrt $\boldsymbol{x}$ is:

$$\begin{bmatrix} \boldsymbol{W} + \boldsymbol{\Sigma} \\ \boldsymbol{A}^T \end{bmatrix} \tag{A10}$$

where $\boldsymbol{W}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{A}$ are defined as $\boldsymbol{W}_k$, $\boldsymbol{\Sigma}_k$ and $\boldsymbol{A}_k$ in the last section. The top matrix is symmetric because it is the Hessian matrix of the barrier objective function. However, $\boldsymbol{A}_k := \nabla \boldsymbol{c}(\boldsymbol{x}_k)^T$ in general is not symmetric, and thus prevents the usage of efficient linear algebra algorithms for symmetric linear system.