Nonlinear Schwarz Preconditioning for Quasi-Newton Methods

Hardik Kothari

1 Introduction

In this work, we consider a nonlinear preconditioning strategy for Quasi-Newton (QN) methods. QN methods are a class of root-finding methods, where the full Jacobian is replaced with an approximation. In the context of this work, we consider secant methods, which take into account a variable number of secant equations at each nonlinear iteration. These types of methods are mostly used if the Jacobian of the nonlinear system is expensive to evaluate, requires more storage, or is simply unavailable. Such scenarios are often encountered while solving coupled multiphysics problems that require higher-order discretization, inverse problems, optimal control problems, training of deep neural networks, etc.

To this aim, we consider the following abstract nonlinear minimization problem:

$$\text{Find } x^* \in \mathcal{V} \text{ that minimizes} \quad \Psi(x), \tag{1}$$

where $\Psi\colon \mathcal{V} \to \mathbb{R}$ denotes a bounded, twice continuously differentiable objective function. The objective function Ψ is obtained by a finite element (FE) discretization of a nonlinear optimization problem, and $\mathcal{V}$ denotes some FE space. To solve (1), we can consider the first-order optimality condition for the function $\Psi(x)$, and then a nonlinear iterative method can be employed to find the root of the nonlinear equation $F(x^*) = 0$, where $F\colon \mathcal{V} \to \mathcal{V}'$ is defined as $F(\cdot) \equiv \nabla\Psi(\cdot)$. We also note that the Hessian of the objective function $\nabla^2\Psi$ is equivalent to the Jacobian F'. Depending on the properties of the objective function Ψ, multiple approaches can be considered to solve (1), for example, Newton's method and its variants; nonlinear Krylov methods; secant methods [11].

Among all these methods, Newton's method is one of the most popular methods to solve such problems due to its locally quadratic convergence property. However, its convergence might suffer if the objective function is highly nonlinear with locally stiff or unbalanced nonlinearities and/or if the initial guess is far from a local

Hardik Kothari
Università della Svizzera italiana, Switzerland, e-mail: hardik.kothari@usi.ch

minimizer. In recent years, some nonlinear preconditioning strategies have been developed to accelerate the convergence of Newton's method, e.g.: Additive Schwarz Preconditioned Inexact Newton (ASPIN) [1]; Nonlinear Elimination Preconditioned Inexact Newton (NEPIN) [2]; Restricted Additive Schwarz Preconditioned Exact Newton (RASPEN) [5]. Similarly, in the context of optimization methods, nonlinear preconditioning strategies have been considered to improve the convergence of a nonlinear Krylov method [3] and a quasi-Newton (QN) method [4]. To the best of our knowledge, unlike the ASPIN, NEPIN, and RASPEN methods, the nonlinear domain decomposition-based preconditioners have not yet been considered for nonlinear Krylov methods and QN methods.

In this work, we apply the nonlinear Schwarz preconditioning strategies to accelerate the convergence of the standard QN methods. We explore the "left" and "right" nonlinear preconditioning strategies and discuss the necessary modifications to the QN framework. Finally, we examine the efficiency of the preconditioned QN methods by means of some numerical experiments.

2 Preconditioned Quasi-Newton methods

In this section, we discuss QN methods, nonlinear restricted additive Schwarz (NRAS) methods, and how to nonlinearly precondition QN methods.

Quasi-Newton methods: Quasi-Newton (QN) methods are quite popular in the optimization community, especially when the Hessian of the underlying minimization problem is unavailable or very expensive to evaluate. In QN methods, the evaluation of the Hessian is replaced by its low-rank approximation. This low-rank approximation of the Hessian is carried out using a secant condition. At each iteration, the approximation of the Hessian B is constructed using the information between subsequent iterations. The approximate Hessian, $B^{(k+1)}$, satisfies the secant equation

$$B^{(k+1)} s^{(k)} = y^{(k)}, \tag{2}$$

where $s^{(k)} = x^{(k+1)} - x^{(k)}$ and $y^{(k)} = F(x^{(k+1)}) - F(x^{(k)})$. As the secant equation is not sufficient to uniquely determine the matrix B, additional constraints have to be imposed on B, which gives rise to different variants of the QN methods. In this work, we consider two types of multi-secant methods, namely the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, and the Andersen acceleration (AA) method. As one of the motivations of this work is reducing the memory footprint of the algorithm, the limited-memory variant of the BFGS method (L-BFGS), and of the AA method, becomes a natural choice. These methods utilize only the m pairs of the vectors $\{s^{(i)}, y^{(i)}\}_{i=k-m}^{k-1}$ from the m most recent iterations to construct the approximate Hessian. We note that the original AA method is not proposed in the context of the optimization but its interpretation as a QN method is established in [6, 12]. The approximate Hessians obtained by the L-BFGS method and the type-I AA method (AA-I) at an iterate $k+1$ can be written in a compact matrix format in the following manner:

$$\begin{aligned} &\text{(L-BFGS)} \quad B^{(k+1)} = B_0 - \begin{bmatrix} B_0 S_k & Y_k \end{bmatrix} \begin{bmatrix} S_k B_0 S_k & L_k \\ L_k^\top & -D_k \end{bmatrix}^{-1} \begin{bmatrix} S_k^\top B_0 \\ Y_k^\top \end{bmatrix}, \\ &\text{(AA-I)} \qquad B^{(k+1)} = I + (Y_k - S_k)(S_k^\top S_k)^{-1} S_k^\top. \end{aligned} \tag{3}$$

Here, $S_k := [s^{(k-m)}, \ldots, s^{(k-1)}]$, $Y_k := [y^{(k-m)}, \ldots, y^{(k-1)}]$, L_k and D_k denote the strictly lower triangular, and the diagonal part of matrix $S_k^\top Y_k$, B_0 denotes some initial Hessian approximation. In order to find the search direction $p^{(k)}$, we need the inverse of the approximate Hessians, which is generally obtained using the Sherman–Morrison–Woodbury formula. To accelerate the convergence speed of these methods, we propose to precondition the QN methods with an NRAS method.

Nonlinear Restricted Additive Schwarz Methods: We consider a decomposition of the domain Ω into n non-overlapping domains $\{\Omega_i\}_{i=1}^n$ and overlapping domains as $\{\Omega_i^\delta\}_{i=1}^n$, such that $\Omega_i \subset \Omega_i^\delta$, here δ denotes the size of the overlap. The FE spaces associated with the overlapping domains are defined as $\{\mathcal{V}_i^\delta\}_{i=1}^n$, $\mathcal{V}_i^\delta \subset \mathcal{V}$. On these overlapping subspaces, we define the restriction and prolongation operators as $R_i^\delta \colon \mathcal{V} \to \mathcal{V}_i^\delta$ and $P_i^\delta \colon \mathcal{V}_i^\delta \to \mathcal{V}$, respectively. We note that for $\delta = 0$, the overlapping decomposition degenerates to a non-overlapping decomposition, i.e., $\Omega_i = \Omega_i^0$. The prolongation operator on the non-overlapping subspaces is termed as *restricted* prolongation operator, $P_i^0 \colon \mathcal{V}_i^0 \to \mathcal{V}$. The overlapping and the non-overlapping decomposition of the subspaces ensures that the partition of unity is satisfied, e.g., $\sum_{i=1}^n P_i^0 R_i^\delta = I$.

Now, we can define a local nonlinear minimization problem restricted to each overlapping subspace as follows. For a given initial guess $x_i^{(0)} = R_i^\delta x^{(k)}$:

$$\text{Find } x_i^* \in \mathcal{V}_i^\delta \text{ that minimizes} \quad \Psi_i^\delta(x_i). \tag{4}$$

Here, $\Psi_i^\delta \colon \mathcal{V}_i^\delta \to \mathbb{R}$ is the restriction of the objective function Ψ to the subspace $\mathcal{V}_i^\delta$. Once the minimization problem is approximately solved on each subdomain, the global iterate is updated in the following manner

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \sum\nolimits_{i=1}^n P_i^0 (x_i^* - R_i^\delta x^{(k)}). \tag{5}$$

We note that the problem (4) is solved on the overlapping subdomains, but the correction is accepted only on the non-overlapping part. Furthermore, to construct a two-level variant of the NRAS method, we define a coarse space $\mathcal{V}_0 \subset \mathcal{V}$ and the restriction and the prolongation operators $R_0 \colon \mathcal{V}' \to \mathcal{V}_0'$ and $P_0 \colon \mathcal{V}_0 \to \mathcal{V}$, where $P_0^\top = R_0$. Also, we define a projection operator $\Pi_0 \colon \mathcal{V} \to \mathcal{V}_0$ to transfer the primal variables to the coarse level. The objective function on the coarse level is defined as $\Psi_0 \colon \mathcal{V}_0 \to \mathbb{R}$, which denotes a discretization of the function Ψ on the space $\mathcal{V}_0$. The coarse space plays an important role in the NRAS method, as it allows global communication between the subdomains and ensures the scalability of the algorithm. In this work, the coarse-level objective function is defined in the spirit of the full approximation scheme (FAS) or the MG-Opt method [10]. The coarse-level function is constructed by adding a first-order consistency term, which is also called

a "defect" in the context of FAS. Thus, the optimization problem on the coarse level is defined as follows. For an initial guess $x_0^{(0)} = \Pi_0 x^{(k)}$:

$$\text{Find } x_0^* \text{ that minimizes} \quad \hat{\Psi}_0(x_0) := \Psi_0(x_0) + \langle \delta g_0, x_0 \rangle \tag{6}$$

where $\delta g_0 = R_0 \nabla\Psi(x^{(k)}) - \nabla\Psi_0(\Pi_0 x^{(k)})$ denotes the first-order consistency term. Additionally, we employ a multiplicative variant of the coarse-level update, where we first approximately solve the problem on the coarse level and bootstrap the initial guess on the subdomains using the approximate solution from the coarse level. The update step for the two-level NRAS is given as follows:

$$x^{(k+1)} = x^{(k)} + \hat{\alpha} T_0(x^{(k)}) + \alpha \sum_{i=1}^{n} P_i^0 (x_i^* - R_i^\delta (x^{(k)} + \hat{\alpha} T_0(x^{(k)}))) \tag{7}$$

where $T_0(x^{(k)}) = P_0(x_0^* - \Pi_0 x^{(k)})$ denotes the coarse-level correction. We note that in (7), $\hat{\alpha}$ and α are computed using a line-search method, while x_i^* and x_0^* denote the approximate solutions of problems (4) and (6), respectively.

Nonlinear preconditioning: In this section, we discuss strategies to nonlinearly preconditioned quasi-Newton methods. Recall, we seek $x^* \in \mathcal{V}$ such that $F(x^*) = 0$. A nonlinear preconditioner G of the residual function F is defined such that the preconditioner G approximates the inverse of the residual i.e., $G \approx F^{-1}$. Practically, it is not possible to obtain such a preconditioning operator G explicitly but, generally, such an operator can be defined implicitly as a fixed-point nonlinear iterative scheme, given as $x = G(x)$. The operator, G, can be applied to the nonlinear residual as either a "left" or a "right" preconditioner, which gives rise to two different nonlinearly preconditioned residuals

$$\mathcal{F}_L(x) = G_L(F(x)) = x - G(x), \qquad \mathcal{F}_R(x) = F(G_R(x)) = F(G(x)). \tag{8}$$

We remark that the left preconditioning operator is not equivalent to a fixed-point nonlinear iterative method $G_L \neq G$, while the right preconditioning operator is a fixed-point iteration scheme $G_R = G$. The ASPIN and RASPEN methods are the "left" preconditioned methods, where the nonlinear residual is first computed using a fixed-point method, and Newton's method is used to solve the equation $\mathcal{F}_L(x) = 0$. The NEPIN method [2], nonlinear FETI-DP and BDDC methods [7] are considered to be the "right" preconditioned methods.

We define generic iterations for both types of preconditioning strategies. The iteration for the preconditioned QN method can be achieved by replacing the residual F with the preconditioned residual given as $\mathcal{F}_{L/R}$. For a given initial iterate $x^{(k)}$, we first compute $x^{(+)}$ using a NRAS method, i.e., $x^{(+)} = G(x^{(k)})$. Once the preconditioning step has been carried out, we can define the iteration for the "left-preconditioned" QN method as,

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \left(B_L^{(k)}\right)^{-1} \mathcal{F}_L(x^{(k)}), \quad \text{where } \mathcal{F}_L(x^{(k)}) = x^{(k)} - G(x^{(k)}). \tag{9}$$

Algorithm 1: Nonlinearly Preconditioned QN method

Data: $F : \mathcal{V} \to \mathcal{V}'$, $x^{(0)} \in \mathcal{V}$, $k \leftarrow 0$
Result: $x^{(k)}$
while $\|F(x^{(k)})\| \geqslant \epsilon_{rtol}\|F(x^{(0)})\|$ **do**
  For given $x^{(k)}$, compute the preconditioned residual $\mathcal{F}_{L/R}(x^{(k)})$
  Compute direction using L-BFGS/AA-I approximation of preconditioned Hessian $p_{L/R}^{(k)} \leftarrow -\left(B_{L/R}^{(k)}\right)^{-1}\mathcal{F}_{L/R}x^{(k)}$
  Find $\alpha^{(k)}$ using a line-search algorithm
  Update iterate: $x^{(k+1)} \leftarrow x^{(k)} + \alpha^{(k)}p_L^{(k)}$ or $x^{(k+1)} \leftarrow x^{(+)} + \alpha^{(k)}p_R^{(k)}$
  Compute $s_{L/R}^{(k)}$ as in (12) and $y_{L/R}^{(k)}$ using (13)
  Update the history of secant pairs $\{s_{L/R}^{(k)}, y_{L/R}^{(k)}\}$
  Update $k \leftarrow k+1$

The update process for the "right-preconditioned" QN method differs from the "left-preconditioning" approach. The iteration for the "right-preconditioned" QN method is given as

$$x^{(k+1)} = x^{(+)} - \alpha^{(k)}\left(B_R^{(k)}\right)^{-1}\mathcal{F}_R(x^{(k)}), \quad \text{where } \mathcal{F}_R(x^{(k)}) = F(G(x^{(k)})). \tag{10}$$

In (9) and (10), we compute $\alpha^{(k)}$ using a line-search method. We note that, the operator G can be explicitly given by (5) and (7) for one-level and two-level NRAS preconditioner, respectively. Here, $B_L^{(k)}$ and $B_R^{(k)}$ denote the approximation of the "left" and "right" preconditioned Hessians, respectively. The QN method aims to approximate the Hessian of the underlying optimization function utilizing a set of vectors $\{s^k, y^k\}$. As we have preconditioned the QN method, we also have to change the underlying secant equation and corresponding secant pairs. The corresponding secant equations for the "left" and the "right" preconditioned systems are now given as

$$B_L^{(k+1)}s_L^{(k)} = y_L^{(k)}, \qquad B_R^{(k+1)}s_R^{(k)} = y_R^{(k)}. \tag{11}$$

From (9) and (10), it is clear that $s_{L/R}^{(k)}$ at each iteration are defined as corrections, which are given as

$$s_L^{(k)} = x^{(k+1)} - x^{(k)}, \qquad s_R^{(k)} = x^{(k+1)} - x^{(+)}. \tag{12}$$

Now, we focus our attention on the computation of $y_{L/R}^{(k)}$, which are defined as the difference between the preconditioned residuals

$$y_L^{(k)} = \mathcal{F}_L(x^{(k+1)}) - \mathcal{F}_L(x^{(k)}), \qquad y_R^{(k)} = F(x^{(k+1)}) - F(x^{(+)}). \tag{13}$$

We note that for the "right" preconditioning approach, the nonlinear preconditioner can be simplified as $\mathcal{F}_R(x^{(k)}) = F(G(x^{(k)})) = F(x^{(+)})$, and the iteration in (10) can be further simplified as

$$x^{(k+1)} = x^{(+)} - \alpha^{(k)} \left(B_R^{(k)}\right)^{-1} F(x^{(+)}).$$

This update process can be interpreted as a half iteration, while the first half of the iteration is the preconditioning step $x^{(+)} = G(x^{(k)})$. Hence, the "right-preconditioned" QN method should only construct the approximation of the Hessian for the second half of the iteration.

A sketch of the nonlinearly preconditioned quasi-Newton method is provided in Algorithm 1.

3 Numerical experiments

We investigate the performance of the nonlinearly preconditioned QN method through some numerical experiments. To this aim, we consider a domain $\Omega = (0,1)^2$ with the boundary Γ. The boundary Γ is decomposed into four parts: top ($\Gamma_t = [0,1] \times \{1\}$), bottom ($\Gamma_b = [0,1] \times \{0\}$), left ($\Gamma_l = \{0\} \times [0,1]$) and right ($\Gamma_r = \{1\} \times [0,1]$). We use the discretize-then-optimize approach, where the discretization is done with the first-order FE method using a mesh with 200×200 quadrilateral elements. The coarse level is also constructed with the same approach, where a mesh with 10×10 elements is employed for discretization.

Minimal surface: This experiment aims to find the surface of the minimal area described by the function u by solving the following minimization problem:

$$\begin{aligned} &\min_{u \in H^1(\Omega)} \Psi_M(u) = \int_\Omega \sqrt{(1 + \|\nabla u\|^2)}\, dx, \\ &\text{subject to } \begin{cases} u = -0.5 \sin(2\pi x_2) \text{ on } \Gamma_l, \;\; u = 0.5 \sin(2\pi x_2) \text{ on } \Gamma_r, \\ u = -0.5 \sin(2\pi x_1) \text{ on } \Gamma_b, \;\; u = 0.5 \sin(2\pi x_1) \text{ on } \Gamma_t. \end{cases} \end{aligned} \tag{14}$$

Setup for the solution methods: As we aim to study the behavior of the preconditioned QN method, we use a fixed configuration of the TL-NRAS method. The overlap for all experiments is prescribed as $\delta = 2$, and the domain Ω is decomposed into 8 subdomains. The partitioning of the mesh is carried out using the METIS library. The preconditioned QN is terminated if one of these conditions is satisfied: $\|F(x^{(k)})\| < 10^{-7}$ or $\|F(x^{(k)})\| < 10^{-6}\|F(x^{(0)})\|$. The subdomain solvers in the TL-NRAS method employ Newton's method, which terminates if $\|F_i(x_i^{(k)})\| < 10^{-10}$ or $\|F_i(x_i^{(k)})\| < 10^{-1}\|F_i(x_i^{(0)})\|$ is satisfied. On the coarse level, we also employ Newton's method, which terminates if $\|F_0(x_0^{(k)})\| < 10^{-12}$ or $\|F_0(x_0^{(k)})\| < 10^{-10}\|F_0(x_0^{(0)})\|$ is satisfied, also the maximum number of iterations is set to 5. We note, Newton's method can be replaced by a multigrid preconditioned Jacobian-free Newton-Krylov method [9] to reduce the memory requirement of the overall methodology. We employ backtracking line-search algorithm with strong Wolfe condition [11, Alg. 3.1, Eq. (3.7)], with $c_1 = 10^{-4}, c_2 = 0.99$, and the value of ρ is chosen to be 0.5 for global solvers and 0.1 for subdomain solvers. The ex-

Table 1 Number of iterations and the time to solution for the L-BFGS method and the TL-NRAS preconditioned QN methods. (L)/(R) denote left/right preconditioning.

Memory	m = 1		m = 3		m = 5		m = 7		m = 10	
	Time (s)	# Iter	Time (s)	# Iter	Time (s)	# Iter	Time (s)	# Iter	Time (s)	# Iter
L-BFGS	698.16	643	720.01	642	699.53	646	702.62	679	536.40	513
L-BFGS (L)	*301.37*	*25*	288.69	23	296.82	24	288.61	23	300.00	25
L-BFGS (R)	426.21	36	296.99	22	*278.68*	*20*	273.99	*19*	*272.45*	*20*
AA-I (L)	350.60	30	285.56	22	284.20	22	287.44	23	284.43	22
AA-I (R)	374.47	36	*274.40*	*22*	281.74	23	*269.98*	21	281.21	22

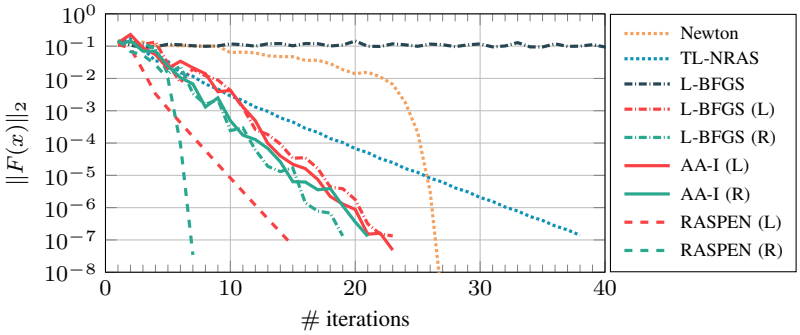

Fig. 1 Convergence history of the L-BFGS method, Newton's method, TL-NRAS method, and TL-NRAS preconditioned QN and RASPEN methods. The QN methods are configured to use the last 7 secant pairs.

periments are carried out using MATLAB on a system with an Intel Core i9-9880H processor, and 16 GB of memory.

Convergence study: In order to study the convergence behavior of the preconditioned QN method, the "left" and the "right" preconditioned variants of the L-BFGS methods and the AA-I method are considered. For this numerical experiment, we store m pairs of secant vectors, where $m \in \{1, 3, 5, 7, 10\}$. Table 1 depicts the time to solution and the required number of iterations to satisfy the termination criterion for different solution methods and different values of m. We have included only preconditioned AA-I method in our study.⋆ From Table 1, it is clear that the preconditioned QN methods outperform the standard L-BFGS method both in terms of the number of iterations and the computational time. Regardless of the number of stored secant pairs, the preconditioned L-BFGS methods and AA-I methods are two times faster than the L-BFGS method. The preconditioned AA-I methods and the preconditioned L-BFGS methods have comparable performance. While the "right" preconditioned L-BFGS methods outperform all other methods if more pairs of secant pairs are used. Figure 1 depicts the convergence history of the preconditioned QN methods, the two-level NRAS method, and Newton's method. We can observe that the preconditioned

⋆ The AA-I method requires factorization of $S_k^\top Y_k$, which is not possible if the successive pairs of $\{s^{(k)}, y^{(k)}\}$ are very similar. To avoid such issues, one can construct the pairs in such a way that successive $s^{(k)}$ are orthogonal [12].

QN method outperforms Newton's method and the L-BFGS method. Also, we can see that the TL-NRAS method has linear convergence, and by employing a QN method as an outer solver we can reduce the number of required iterations in half. Specifically for the left-preconditioning, in comparison with the RASPEN methods the preconditioned QN method shows only mild deterioration in the convergence.

From the performed experiments, we can conclude that the proposed domain decomposition-based preconditioning strategy is quite robust both in the case of the L-BFGS method and the type-I AA method. This works provides a promising future direction for problems when memory is a limiting factor, for example for solving the phase field fracture problems [8] or for the training of deep neural networks.

Acknowledgements This work is supported by the Swiss National Science Foundation (SNSF) and the Deutsche Forschungsgemeinschaft for their through the project SPP 1962 "Stress-Based Methods for Variational Inequalities in Solid Mechanics: Finite Element Discretization and Solution by Hierarchical Optimization" [186407]. We also acknowledge the support of Platform for Advanced Scientific Computing through the project FraNetG: Fracture Network Growth and SNSF through the project ML2 - Multilevel and Domain Decomposition Methods for Machine Learning [197041].

References

1. Cai, X.-C. and Keyes, D. E. Nonlinearly Preconditioned Inexact Newton Algorithms. *SIAM Journal on Scientific Computing* **24**(1), 183–200 (2002).
2. Cai, X.-C. and Li, X. Inexact Newton Methods with Restricted Additive Schwarz Based Nonlinear Elimination for Problems with High Local Nonlinearity. *SIAM Journal on Scientific Computing* **33**(2), 746–762 (2011).
3. De Sterck, H. and Howse, A. Nonlinearly Preconditioned Optimization on Grassmann Manifolds for Computing Approximate Tucker Tensor Decompositions. *SIAM Journal on Scientific Computing* **38**(2), A997–A1018 (2016).
4. De Sterck, H. and Howse, A. J. Nonlinearly preconditioned L-BFGS as an acceleration mechanism for alternating least squares with application to tensor decomposition. *Numerical Linear Algebra with Applications* **25**(6), e2202 (2018).
5. Dolean, V., Gander, M. J., Kheriji, W., Kwok, F., and Masson, R. Nonlinear Preconditioning: How to Use a Nonlinear Schwarz Method to Precondition Newton's Method. *SIAM Journal on Scientific Computing* **38**(6), A3357–A3380 (2016).
6. Fang, H. and Saad, Y. Two classes of multisecant methods for nonlinear acceleration. *Numerical Linear Algebra with Applications* **16**(3), 197–221 (2009).
7. Klawonn, A., Lanser, M., Rheinbach, O., and Uran, M. Nonlinear FETI-DP and BDDC Methods: A Unified Framework and Parallel Results. *SIAM Journal on Scientific Computing* **39**(6), C417–C451 (2017).
8. Kopaničáková, A., Kothari, H., and Krause, R. Nonlinear field-split preconditioners for solving monolithic phase-field models of brittle fracture. *Comput. Methods Appl. Mech. Engrg.* **403**, 115733 (2023).
9. Kothari, H., Kopaničáková, A., and Krause, R. A Multigrid Preconditioner for Jacobian-free Newton –Krylov Methods. In: *Domain Decomposition Methods in Science and Engineering XXVI*, 365–372 (2022).
10. Nash, S. A multigrid approach to discretized optimization problems. *Optimization Methods and Software* **14**(1), 99–116 (2000).
11. Nocedal, J. and Wright, S. *Numerical Optimization*. Springer (2000).
12. Zhang, J., O'Donoghue, B., and Boyd, S. Globally Convergent Type-I Anderson Acceleration for Nonsmooth Fixed-Point Iterations. *SIAM Journal on Optimization* **30**(4), 3170–3197 (2020).