Subspace Methods for Nonlinear Optimization

Xin Liu*
Zaiwen Wen†
Ya-xiang Yuan‡

Abstract. Subspace techniques such as Krylov subspace methods have been well known and extensively used in numerical linear algebra. They are also ubiquitous and becoming indispensable tools in nonlinear optimization due to their ability to handle large scale problems. There are generally two types of principals: i) the decision variable is updated in a lower dimensional subspace; ii) the objective function or constraints are approximated in a certain smaller functional subspace. The key ingredients are the constructions of suitable subspaces and subproblems according to the specific structures of the variables and functions such that either the exact or inexact solutions of subproblems are readily available and the corresponding computational cost is significantly reduced. A few relevant techniques include but not limited to direct combinations, block coordinate descent, active sets, limited-memory, Anderson acceleration, subspace correction, sampling and sketching. This paper gives a comprehensive survey on the subspace methods and their recipes in unconstrained and constrained optimization, nonlinear least squares problem, sparse and low rank optimization, linear and nonlinear eigenvalue computation, semidefinite programming, stochastic optimization and etc. In order to provide helpful guidelines, we emphasize on high level concepts for the development and implementation of practical algorithms from the subspace framework.

Key words. nonlinear optimization, subspace techniques, block coordinate descent, active sets, limited memory, Anderson acceleration, subspace correction, subsampling, sketching

AMS subject classification. 65K05, 90C30

*State Key Laboratory of Scientific and Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, and University of Chinese Academy of Sciences, China (liuxin@lsec.cc.ac.cn). Research supported in part by NSFC grants 11622112, 11471325, 91530204 and 11688101, the National Center for Mathematics and Interdisciplinary Sciences, CAS, and Key Research Program of Frontier Sciences QYZDJ-SSW-SYS010, CAS.

†Beijing International Center for Mathematical Research, Peking University, China (wenzw@pku.edu.cn). Research supported in part by NSFC grant 11831002, and by Beijing Academy of Artificial Intelligence (BAAI).

‡State Key Laboratory of Scientific and Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, China (yyx@lsec.cc.ac.cn). Research supported in part by NSFC grants 11331012 and 11688101.

1. Introduction.

Large scale optimization problems appear in a wide variety of scientific and engineering domains. In this paper, we consider a general optimization problem

$$\min_x \ f(x), \ \text{s.t.} \ \ x \in \mathcal{X}, \tag{1.1}$$

where x is the decision variable, $f(x)$ is the objective function and $\mathcal{X}$ is the feasible set. Efficient numerical optimization algorithms have been extensively developed for (1.1) with various types of objective functions and constraints [111, 88]. With the rapidly increasing problem scales, subspace techniques are ubiquitous and becoming indispensable tools in nonlinear optimization due to their ability to handle large scale problems. For example, the Krylov subspace methods developed in the numerical linear algebraic community have been widely used for the linear least squares problem and linear eigenvalue problem. The characteristics of the subspaces are clear in many popular optimization algorithms such as the linear and nonlinear conjugate gradient methods, Nesterov's accelerated gradient method, the Quasi-Newton methods and the block coordinate decent (BCD) method. The subspace correction method for convex optimization can be viewed as generalizations of multigrid and domain decomposition methods. The Anderson acceleration or the direct inversion of iterative subspace (DIIS) methods have been successful in computational quantum physics and chemistry. The stochastic gradient type methods usually take a mini-batch from a large collection samples so that the computational cost of each inner iteration is small. The sketching techniques formulate a reduced problem by a multiplication with random matrices with certain properties.

The purpose of this paper is to provide a review of the subspace methods for nonlinear optimization, for their further improvement and for their future usage in even more diverse and emerging fields. The subspaces techniques for (1.1) are generally divided into two categories. The first type is to update the decision variable in a lower dimensional subspace, while the second type is to construct approximations of the objective function or constraints in a certain smaller subspace of functions. Usually, there are three key steps.

- Identify a suitable subspace either for the decision variables or the functions.
- Construct a proper subproblem by various restrictions or approximations.
- Find either an exact or inexact solution of subproblems.

These steps are often mixed together using the specific structures of the problems case by case. The essence is how to reduce the corresponding computational cost significantly. During the practice in unconstrained and constrained optimization, nonlinear least squares problem, sparse and low rank optimization, linear and nonlinear eigenvalue computation, semidefinite programming, stochastic optimization, manifold optimization, phase retrieval,

variational minimization and etc, the collection of subspaces techniques is growing ever rich. It includes but not limited to direct combinations, BCD, active sets, limited-memory, Anderson acceleration, subspace correction, sampling and sketching. We aim to provide helpful guidelines for the development and implementation of practical algorithms using the subspace framework. Hence, only high level algorithmic ideas rather than theoretical properties of the subspace techniques are covered in various contexts.

1.1. Overview of Subspace Techniques. We next summarize the concepts and contexts of a few main subspace techniques.

Direct Combinations. It is a common practice to update the decision variables using a combination of a few known directions which forms a subspace. The linear and nonlinear conjugate gradient methods [111, 88], the Nesterov's accelerated gradient method [84, 85], the Heavy-ball method [90], the search direction correction method [126] and the momentum method [47] take a linear combination of the gradient and the previous search direction. The main difference is reflected in the choices of the coefficients according to different explicit formulas.

BCD. The variables in many problems can be split naturally into a few blocks whose subspaces are spanned by the coordinate directions. The Gauss-Seidel type of the BCD method updates only one block by minimizing the objective function or its surrogate while all other blocks are fixed at each iteration. It has been one of the core algorithmic idea in solving problems with block structures, such as convex programming [77], nonlinear programming [9], semidefinite programming [129, 145], compressive sensing [72, 32], etc. A proximal alternating linearized minimization method is developed in [10] for solving a summation of nonconvex but differentiable and nonsmooth functions. The alternating direction methods of multipliers (ADMM) [11, 27, 41, 45, 55, 125] minimize the augmented Lagrangian function with respect to the primal variables by BCD, then update the Lagrangian multiplier.

Active Sets. When a clear partition of variables is not available, a subset of the variables can be fixed in the so-called active sets under certain mechanisms and the remaining variables are determined from certain subproblems for optimization problems with bound constraints or linear constraints in [17, 18, 51, 81, 82], ℓ_1-regularized problem for sparse optimization in [133, 105, 64] and general nonlinear programs in [19, 20]. In quadratic programming, the inequality constraints that have zero values at the optimal solution are called active, and they are replaced by equality constraints in the subproblem [111].

Limited-memory. A typical subspace is constructed from a number of history information, for example, the previous iterates $\{x_k\}$, the previous gradients $\{\nabla f(x_k)\}$, the differences between two consecutive iterates $\{x_k - x_{k-1}\}$, and the differences between two consecutive gradients $\{\nabla f(x_k) - \nabla f(x_{k-1})\}$. After the new iterate is formed, the oldest vectors in the storage are replaced by the most recent vectors if certain justification rules are satisfied. Two examples are the limited memory BFGS method [111, 88], and the limited memory block Krylov subspace optimization method (LMSVD) [74].

Anderson Acceleration. For a sequence $\{x_k\}$ generated by a general fixed-point iteration, the Anderson acceleration produces a new point using a linear combination of a few points in $\{x_k\}$, where the coefficients are determined from an extra linear least squares problem with a normalized constraint [13, 4, 123]. A few related schemes include the minimal polynomial extrapolation, modified minimal polynomial extrapolation, reduced rank extrapolation, the vector Epsilon algorithm and the topological Epsilon algorithm. The Anderson acceleration is also known as Anderson mixing, Pulay mixing, DIIS or the commutator DIIS [92, 93, 115] in electronic structure calculation. These techniques have also been applied to optimization problems in [99, 147].

Subspace correction. For variational problems, the domain decomposition methods

split the spatial domain into several subdomains and solve the corresponding problems on these subdomains iteratively using certain strategies. The successive subspace correction (SSC) and parallel subspace correction (PSC) methods [22, 36, 39, 38, 68, 112] are similar to the Gauss-Seidel-type and Jacobian-type BCD methods, respectively. However, the subspace correction is significantly different from BCD due to the strong connections between variables in the subdomains. The PSC methods have been studied for LASSO in [36, 39, 29] and total variation minimization in [37, 38, 39, 68].

Sampling. Assume that there are a large number of data. The general concept of sampling is to randomly select a small set of samples with an appropriate probability distribution with or without replacement. In the stochastic gradient descent type methods, the gradient in expectation is approximated by a sum of sample gradients over a mini-batch [47]. Random sampling is also helpful in many other contexts, for example, a greedy algorithm for a mixed integer programming in volumetric modulated arc therapy [139].

Sketching. For huge data represented in matrices, the sketching technique builds low-dimensional approximations using random linear maps [78, 136, 118]. It has been adopted for nonlinear least squares problems in [141, 103] and large scale SDP problems in [144]. The Nyström approximation can be viewed as a special sketching scheme. An initial quasi-Newton matrix can be constructed if a single Hessian-matrix multiplication is affordable in [58].

1.2. Notation. Let $\mathcal{S}^n$ be the collection of all n-by-n symmetric matrices. For any matrix $X \in \mathbb{R}^{n\times n}$, $\mathrm{diag}(X)$ denotes a column vector consisting of all diagonal entries of X. For any vector $x \in \mathbb{R}^n$, $\mathrm{Diag}(x)$ is an n-by-n diagonal matrix whose i-th diagonal entry is x_i. Given two matrices $A, B \in \mathbb{C}^{n\times p}$, the Frobenius inner product is defined as $\langle A, B\rangle = \mathrm{tr}(A^*B)$, and the corresponding Frobenius norm is defined as $\|A\|_F = \sqrt{\mathrm{tr}(A^*A)}$. The operation $A \odot B$ denotes the Hadamard product between two matrices A and B of the same sizes. Let e_n be a vector of all ones in $\mathbb{R}^n$. For any matrix $X \in \mathbb{R}^{n\times p}$, $\mathrm{Range}(X)$ denotes the subspace spanned by the columns of X. The subscript usually denotes the iteration number, while the supscript is reserved as the index of a vector or matrix.

1.3. Organization. The rest of this paper is organized as follows. The subspace methods applied in general unconstrained optimization, nonlinear equations and nonlinear least squares problem, stochastic optimization, sparse optimization, the domain decomposition, general constrained optimization, eigenvalue computation, optimization problems with orthogonality constraints, semidefinite programming and low rank matrix optimization are discussed in Sections 2 to 11, respectively. Finally, a few typical scenarios are summarized in Section 12.

2. General Unconstrained Optimization.

In this section, we consider the unconstrained optimization

$$\min_{x\in\mathbb{R}^n} f(x)\,, \tag{2.1}$$

where $f(x) : \mathbb{R}^n \to \mathbb{R}$ is a differentiable function. The line search and trust region methods are the two main types of approaches for solving (2.1). The main difference between them is the order of determining the so-called step size and search direction. Subspace techniques have been substantially studied in [26, 48, 140, 142, 143, 87, 128, 127, 49].

2.1. The Line Search Methods. At the k-th iteration x_k, the line search methods first generate a descent search direction d_k and then search along this direction for a step size α_k such that the objective function at the next point

$$x_{k+1} = x_k + \alpha_k d_k \tag{2.2}$$

is suitably reduced. The step size α_k is often selected by the monotone line search procedures with the Armijo, Goldstein or the Wolfe-Powell rules. The nonmonotone line procedures are also widely used. Interested readers are referred to [111, 88] for further information. Here, we mainly focus on generating the direction d_k in a subspace $\mathfrak{S}_k$, i.e.,

$$d \in \mathfrak{S}_k.$$

For simplicity, we often denote $g_k = \nabla f(x_k)$.

2.1.1. The Nonlinear Conjugate Gradient (CG) Method. The nonlinear CG method is popular for solving large scale optimization problems. The search direction d_k lies in a particular subspace

$$\mathfrak{S}_k = \text{span}\{g_k, d_{k-1}\}, \tag{2.3}$$

which is spanned by the gradient g_k and the last search direction d_{k-1}. More specifically, d_k is a linear combination of $-g_k$ and d_{k-1} with a weight β_{k-1}, i.e.,

$$d_k = -g_k + \beta_{k-1} d_{k-1}, \tag{2.4}$$

where $d_0 = -g_0$ and $\beta_0 = 0$. A few widely used choices for the weight β_{k-1} are

$$\beta_{k-1} = \frac{g_k^\top g_k}{g_{k-1}^\top g_{k-1}}, \quad \text{(F-R Formula)},$$

$$\beta_{k-1} = \frac{g_k^\top (g_k - g_{k-1})}{d_{k-1}^\top (g_k - g_{k-1})}, \quad \text{(H-S or C-W Formula)},$$

$$\beta_{k-1} = \frac{g_k^\top (g_k - g_{k-1})}{g_{k-1}^\top g_{k-1}}, \quad \text{(PRP Formula)},$$

$$\beta_{k-1} = -\frac{g_k^\top g_k}{d_{k-1}^\top g_{k-1}}, \quad \text{(Dixon Formula)},$$

$$\beta_{k-1} = -\frac{g_k^\top g_k}{d_{k-1}^\top (g_k - g_{k-1})}, \quad \text{(D-Y Formula)}.$$

It is easy to observe that these formulas are equivalent in the sense that they yield the same search directions when the function $f(x)$ is quadratic with a positive definite Hessian matrix. In this case, the directions $d_1, \ldots, d_k$ are conjugate to each other with respect to the Hessian matrix. It can also be proved that the CG method has global convergence and n-step local quadratic convergence. However, for a general nonlinear function with inexact line search, the behavior of the methods with different β_k can be significantly different.

2.1.2. Nesterov's Accelerated Gradient Method. The steepest descent gradient method simply uses $d_k = -g_k$ in (2.2) for unconstrained optimization. Assume that the function $f(x)$ is convex, the optimal value f^* of (2.1) is finite and it attains at a point x^*, and the gradient $f(x)$ is Lipschitz continuous with a constant L, i.e.,

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|.$$

Let $\{x_k\}_{k=0}^{\infty}$ be a sequence generated by the gradient method with a fixed step size $\alpha_k = \frac{1}{L}$. Then it can be proved that the convergence of the objective function values is

$$f(x_k) - f(x^*) \leq \frac{L}{2k}\|x_0 - x^*\|^2,$$

which is often described as a convergence rate at $O(1/k)$.

A natural question is whether a faster convergence rate can be achieved if only the gradient information is used. We now present the so-called FISTA method proposed by Beck and Teboulle [5] which is equivalent to Nesterov accelerated gradient method [84, 85]. The FISTA method first calculates a new point by an extrapolation of the previous two points, then performs a gradient step at this new point:

$$y_k = x_{k-1} + \frac{k-2}{k+1}(x_{k-1} - x_{k-2}),$$
$$x_k = y_k - \alpha_k \nabla f(y_k).$$

An illustration of the FISTA method is shown in Figure 2.1. Under the same assumptions as

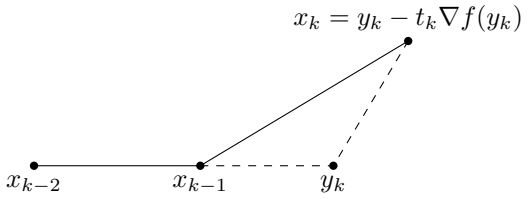

Fig. 2.1 *The FISTA method*

the gradient method, the FISTA method with a fixed step size $\alpha_k = \frac{1}{L}$ has a convergence rate of $O(1/k^2)$, i.e.,

$$f(x_k) - f^* \leq \frac{2L}{(k+1)^2}\|x_0 - x^*\|^2.$$

Obviously, the FISTA method can also be interpreted as a subspace method whose subspace is

$$\mathfrak{S}_k = \text{span}\{x_{k-1}, x_{k-2}, \nabla f(y_k)\}. \tag{2.5}$$

2.1.3. The Heavy-ball Method. The heavy-ball method [90] is also a two-step scheme:

$$d_k = -g_k + \beta d_{k-1},$$
$$x_{k+1} = x_k + \alpha d_k,$$

with $p_0 = 0$ and $\alpha, \beta > 0$. If $\beta \in [0, 1)$ and $\alpha \in \left(0, \frac{1-\beta}{L}\right]$ and under the same assumptions as in Sec. 2.1.2, it is established in [42] that

$$f(\bar{x}_k) - f^* \leq \frac{1}{k+1}\left(\frac{\beta}{1-\beta}(f(x_0) - f^*) + \frac{1-\beta}{2\alpha}\|x_0 - x^*\|^2\right),$$

where $\bar{x}_k = \frac{1}{1+k}\sum_{i=1}^{k} x_i$. We can see that the Heavy-ball method is the same as the nonlinear CG method (2.4) except that the parameter β is different.

2.1.4. A Search Direction Correction (SDC) Method. The search direction (2.4) can also be modified by adding a non-trivial weight to g_k. Let $d_0 = 0$. At the beginning of the $(k+1)$-th iteration, if a descent condition

$$\langle g_k, d_k \rangle \leq 0 \tag{2.6}$$

holds, we update

$$d_{k+1} = (1-\beta_k)d_k - \gamma_k \frac{\|d_k\|}{\|g_k\|} g_k - g_k. \tag{2.7}$$

Then we update β_{k+1} and γ_{k+1} as follows:

$$\beta_k = \frac{r}{l_k - 1 + r}, \quad \gamma_k = \frac{r-3}{l_k - 1 + r}, \tag{2.8}$$

where $r \geq 3$, $\{l_k\}$ is a sequence of parameters with of $l_1 = 1$ and $l_{k+1} = l_k + 1$. If the criterion (2.6) is not met, we reset d_{k+1}, β_{k+1} and γ_{k+1} as

$$d_{k+1} = -g_k, \beta_{k+1} = \beta_1, \gamma_{k+1} = \gamma_1, l_{k+1} = l_1.$$

For more details, we refer the reader to [126].

2.1.5. Quasi-Newton Methods. The search directions of the limited-memory quasi-Newton methods [111, 88] also lie in subsapces. Let B_k be the limited-memory BFGS (L-BFGS) matrix and H_k be its inverse matrix generated from a few most recent pairs $\{s_i, y_i\}$, where

$$s_i = x_{i+1} - x_i, \quad y_i = g_{i+1} - g_i.$$

Then the search direction is

$$d_k = -B_k^{-1} g_k = -H_k g_k, \tag{2.9}$$

which is usually computed by the two-loop recursion. In fact, both B_k and H_k can be written in a compact representation [21]. Assume that there are p pairs of vectors:

$$U_k = [s_{k-p}, \ldots, s_{k-1}] \in \mathbb{R}^{n\times p}, \quad Y_k = [y_{k-p}, \ldots, y_{k-1}] \in \mathbb{R}^{n\times p}. \tag{2.10}$$

For a given initial matrix H_k^0, the H_k matrix is:

$$H_k = H_k^0 + C_k P_k C_k^\top, \tag{2.11}$$

where

$$C_k := \left[U_k, H_k^0 Y_k\right] \in \mathbb{R}^{n\times 2p}, \quad D_k = \operatorname{diag}\left[s_{k-p}^\top y_{k-p}, \ldots, s_{k-1}^\top y_{k-1}\right]$$

$$P_k := \begin{bmatrix} R_k^{-\top}(D_k + Y_k^\top H_k^0 Y_k) R_k^{-1} & -R_k^{-\top} \\ -R_k^{-1} & 0 \end{bmatrix}, (R_k)_{i,j} = \begin{cases} s_{k-p+i-1}^\top y_{k-p+j-1}, & \text{if } i \leq j, \\ 0, & \text{o.w.} \end{cases}$$

The initial matrix H_k^0 is usually set to be a positive scalar γ_k times the identity matrix, i.e., $\gamma_k I$. Therefore, we have

$$d_k \in \operatorname{span}\{g_k, s_{k-1}, \ldots, s_{k-p}, y_{k-1}, \ldots, y_{k-p}\}.$$

2.1.6. Acceleration Techniques. Gradient descent algorithms may converge slowly after certain iterations. This issue can be resolved by using acceleration techniques such as Anderson Acceleration (AA) [4, 123]. An extrapolation-based acceleration techniques proposed in [99] can be applied to overcome the instability of the Anderson Acceleration. To be precise, we perform linear combinations of the points x_k every $l+2$ iterations to obtain a better estimation $\tilde{x} = \sum_{i=0}^{l} \tilde{c}_i x_{k-l+i}$. Define the difference of $l+2$ iteration points as

$$U = [x_{k-l+1} - x_{k-l}, \ldots, x_{k+1} - x_k].$$

Then the coefficients $\tilde{c} = (\tilde{c}_0, \ldots, \tilde{c}_l)^T$ is the solution of the following problem

$$\tilde{c} = \underset{c^\top e_{l+1}=1}{\arg\min}\ c^T (U^T U + \lambda I) c, \tag{2.12}$$

where $\lambda > 0$ is a regularization parameter.

2.1.7. Search Direction From Minimization Subproblems. We next construct the search direction by solving a subproblem defined in a subspace $\mathfrak{S}_k$ as

$$\min_{d\in\mathfrak{S}_k} Q_k(d)\,, \tag{2.13}$$

where $Q_k(d)$ is an approximation to $f(x_k+d)$ for d in the subspace $\mathfrak{S}_k$. It would be desirable that the approximation model $Q_k(d)$ has the following properties: (i) it is easy to be minimized in the subspace $\mathfrak{S}_k$; (ii) it is a good approximation to f and the solution of the subspace subproblem will give a sufficient reduction with respect to the original objective function f.

It is natural to use quadratic approximations to the objective function. This leads to quadratic models in subspaces. A successive two-dimensional search algorithm is developed by Stoer and Yuan in [143] based on

$$\min_{d\in\mathrm{span}\{-g_k,d_{k-1}\}} \quad Q_k(d).$$

Let the dimension $dim(\mathfrak{S}_k)=\tau_k$ and $\mathfrak{S}_k$ be a set generated by all linear combinations of vectors $p_1,p_2,\ldots,p_{\tau_k}\in\mathbb{R}^n$, i.e.,

$$\mathfrak{S}_k=\mathrm{span}\{p_1,p_2,\ldots,p_{\tau_k}\}\,.$$

Define $P_k=[p_1,p_2,...,p_{\tau_k}]$. Then $d\in\mathfrak{S}_k$ can be represented as $d=P_k\bar{d}$ with $\bar{d}\in\mathbb{R}^{\tau_k}$. Hence, a quadratic function $Q_k(d)$ defined in the subspace can be expressed as a function $\bar{Q}_k$ in a lower dimension space $\mathbb{R}^{\tau_k}$ in terms of $Q_k(d)=\bar{Q}_k(\bar{d})$. Since τ_k usually is quite small, the Newton method can be used to solve (2.13) efficiently.

We now discuss a few possible choices for the subspace $\mathfrak{S}_k$. A special subspace is a combination of the current gradient and the previous search directions, i.e.,

$$\mathfrak{S}_k=\mathrm{span}\{-g_k,s_{k-1},...,s_{k-m}\}\,. \tag{2.14}$$

In this case, any vector d in the subspace $\mathfrak{S}_k$ can be expressed as

$$d=\alpha g_k+\sum_{i=1}^{m}\beta_i s_{k-i}=(-g_k,s_{k-1},\cdots,s_{k-m})\bar{d} \tag{2.15}$$

where $\bar{d}=(\alpha,\beta_1,\cdots,\beta_m)^\top\in\mathbb{R}^{m+1}$. All second order terms of the Taylor expansion of $f(x_k+d)$ in the subspace $\mathfrak{S}_k$ can be approximated by secant conditions

$$s_{k-i}^\top\nabla^2 f(x_k)s_{k-j}\approx s_{k-i}^\top y_{k-j}\,,\quad s_{k-i}^\top\nabla^2 f(x_k)g_k\approx y_{k-i}^\top g_k\,, \tag{2.16}$$

except $g_k^\top\nabla^2 f(x_k)g_k$. Therefore, it is reasonable to use the following quadratic model in the subspace $\mathfrak{S}_k$:

$$\bar{Q}_k(\bar{d})=(-\|g_k\|^2,g_k^\top s_{k-1},\cdots,g_k^\top s_{k-m})\bar{d}+\frac{1}{2}\bar{d}^\top\bar{B}_k\bar{d}\,, \tag{2.17}$$

where

$$\bar{B}_k=\begin{pmatrix}\rho_k & -g_k^\top y_{k-1} & \cdots & -g_k^\top y_{k-m}\\ -g_k^\top y_{k-1} & y_{k-1}^\top s_{k-1} & \cdots & y_{k-m}^\top s_{k-1}\\ \vdots & \vdots & \ddots & \vdots\\ -g_k^\top y_{y-m} & y_{k-m}^\top s_{k-1} & \cdots & y_{k-m}^\top s_{k-m}\end{pmatrix} \tag{2.18}$$

with $\rho_k \approx g_k^\top \nabla^2 f(x_k) g_k$. Hence, once a good estimation to the term $g_k^\top \nabla^2 f(x_k) g_k$ is available, we can obtain a good quadratic model in the subspace $\mathfrak{S}_k$.

There are different ways to choose ρ_k. Similar to the approach in [143], we can let

$$\rho_k = 2\frac{(s_{k-1}^\top g_k)^2}{s_{k-1}^\top y_{k-1}}, \tag{2.19}$$

due to the fact that the mean value of $\cos^2(\theta)$ is $\frac{1}{2}$, which gives

$$g_k^\top \nabla^2 f(x_k) g_k = \frac{1}{\cos^2\theta_k}\frac{(s_{k-1}^\top \nabla^2 f(x_k) g_k)^2}{s_{k-1}^\top \nabla^2 f(x_k) s_{k-1}} \approx 2\frac{(s_{k-1}^\top g_k)^2}{s_{k-1}^\top y_{k-1}}, \tag{2.20}$$

where θ_k is the angle between $(\nabla^2 f(x_k))^{\frac{1}{2}} g_k$ and $(\nabla^2 f(x_k))^{\frac{1}{2}} s_{k-1}$. Another way to estimate $g_k^\top (\nabla^2 f(x_k)) g_k$ is by replacing $\nabla^2 f(x_k)$ by a quasi-Newton matrix. We can also obtain ρ_k by computing an extra function value $f(x_k + t g_k)$ and setting

$$\rho_k = \frac{2(f(x_k + t g_k) - f(x_k) - t\|g_k\|_2^2)}{t^2}. \tag{2.21}$$

By letting the second order curvature along g_k to be the average of those along $s_{k-i} (i = 1, ..., m)$, we obtain

$$\rho_k = \frac{\|g_k\|_2^2}{m}\sum_{i=1}^{m}\frac{s_{k-i}^\top y_{k-i}}{s_{k-i}^\top s_{k-i}}. \tag{2.22}$$

Similar to (2.14), a slightly different subspace is

$$\mathfrak{S}_k = \text{span}\{-g_k, y_{k-1}, ..., y_{k-m}\}. \tag{2.23}$$

In this case, any vector in $\mathfrak{S}_k$ can be represented as

$$d = \alpha g_k + \sum_{i=1}^{m}\beta_i y_{k-i} = W_k \bar{d} \tag{2.24}$$

where $W_k = [-g_k, y_{k-1}, ..., y_{k-m}] \in \mathbb{R}^{n\times(m+1)}$. The Newton step in the subspace $\mathfrak{S}_k$ is $W_k \bar{d}_k$ with

$$\bar{d}_k = -\left[W_k^\top \nabla^2 f(x_k) W_k\right]^{-1} W_k^\top \nabla f(x_k). \tag{2.25}$$

Thus, the remaining issue is to obtain a good estimate of $\bar{d}_k$, using the fact that all the elements of $\left[W_k^\top (\nabla^2 f(x_k))^{-1} W_k\right]$ is known except one entry $g_k \nabla^2 f(x_k)^{-1} g_k$.

2.1.8. Subspace By Coordinate Directions. We next consider subspaces spanned by coordinate directions with sparsity structures. Let g_k^i be the i-th component of the gradient g_k. The absolute values $|g_k^i|$ are sorted in the descending order such that

$$|g_k^{i_1}| \geq |g_k^{i_2}| \geq |g_k^{i_3}| \geq \cdots \geq |g_k^{i_n}|. \tag{2.26}$$

The subspace

$$\mathfrak{S}_k = \text{span}\{e^{i_1}, e^{i_2}, ..., e^{i_\tau}\} \tag{2.27}$$

is called as the τ-*steepest coordinates subspace*, where e^i is a vector of all zeros except that the i-th component is one. Then, the steepest descent direction in the subspace is sufficiently descent, namely

$$\min_{d\in\mathfrak{S}_k} \frac{d^\top g_k}{\|d\|_2\|g_k\|_2} \le -\frac{\tau}{n}\,. \tag{2.28}$$

When $(g_k^{i_{\tau+1}})^2 \le \epsilon\sum_{j=1}^{\tau}(g_k^{i_j})^2$, the following estimation can be established:

$$\min_{d\in\mathfrak{S}_k} \frac{d^\top g_k}{\|d\|_2\|g_k\|_2} \le -\frac{1}{\sqrt{1+\epsilon(n-\tau)}}\,. \tag{2.29}$$

Consequently, a sequential steepest coordinates search (SSCS) technique can be designed by augmenting the steepest coordinate directions into the subspace sequentially. For example, consider minimizing a convex quadratic function

$$Q(x) = g^\top x + \frac{1}{2}x^\top Bx.$$

At the k-th iteration of SSCS, we first compute $g_k = \nabla Q(x_k)$, then choose

$$i_k = \arg\min_i\{|g_k^i|\}.$$

Let $\mathfrak{S}_k = \mathrm{span}\{e^{i_1}, ..., e^{i_k}\}$. The next iteration is to find

$$x_{k+1} = \arg\min_{x\in x_k+\mathfrak{S}_k} Q(x).$$

2.2. Trust Region Methods. The trust region methods for (2.1) compute a search direction in a ball determined by a given trust region radius whose role is similar to the step size. The trust region subproblem (TRS) is normally

$$\begin{aligned}\min_{s\in\mathbb{R}^n}\quad & Q_k(s) = g_k^\top s + \frac{1}{2}s^\top B_k d\\ \text{s. t.}\quad & \|s\|_2 \le \Delta_k\,,\end{aligned} \tag{2.30}$$

where B_k is an approximation to the Hessian $\nabla^2 f(x_k)$ and $\Delta_k > 0$ is the trust region radius.

A subspace version of the trust region subproblem is suggested in [101]:

$$\begin{aligned}\min_{s\in\mathbb{R}^n}\quad & Q_k(s)\\ \text{s. t.}\quad & \|s\|_2 \le \Delta_k, \quad s\in\mathfrak{S}_k.\end{aligned}$$

The Steihaug truncated CG method [107] for solving (2.30) is essentially a subspace method. When the approximate Hessian B_k is generated by the quasi-Newton updates such as the SR1, PSB or the Broyden family [111, 88], the TRS has subspace properties. Suppose $B_1 = \sigma I$ with $\sigma > 0$, let s_k be an optimal solution of TRS (2.30) and set $x_{k+1} = x_k + s_k$. Let $\mathcal{G}_k = \mathrm{span}\{g_1, g_2, \cdots, g_k\}$. Then it can be proved that $s_k \in \mathcal{G}_k$ and for any $z \in \mathcal{G}_k$, $w \in \mathcal{G}_k^\perp$, it holds

$$B_k z \in \mathcal{G}_k, \quad B_k u = \sigma u\,. \tag{2.31}$$

Therefore, the subspace trust region algorithm generates the same sequences as the full space trust region quasi-Newton algorithm. Based on the above results, Wang and Yuan [128]

presented a subspace trust region quasi-Newton method for large scale unconstrained optimization. Similar results for the line search quasi-Newton methods were obtained by Gill and Leonard [44, 43].

We next discuss a special trust region subproblem which can make good use of subspace properties. If the norm $\|.\|_2$ is replaced by a general norm $\|.\|_W$ in (2.30), we can obtain a general TRS subproblem

$$\begin{aligned} \min_{s\in\mathbb{R}^n} \quad & g^\top s + \frac{1}{2} s^\top B s \\ \text{s. t.} \quad & \|s\|_W \le \Delta. \end{aligned}$$

Here, the subscript k in g_k and B_k is omitted for simplicity. Intuitively, we should choose the norm $\|.\|_W$ properly so that the TRS can easily be solved by using the corresponding subspace properties of the objective function $g^\top s + \frac{1}{2} s^\top B s$. Assume that B is a limited memory quasi-Newton matrix which can be expressed as

$$B = \sigma I + P D P^\top, \qquad P \in \mathbb{R}^{n\times l},$$

where $P^\top P = I$. Let $P_\perp^\top$ be the projection onto the space orthogonal to $\mathrm{Range}(P)$. Then the following function

$$\|s\|_P = \max\{\|P^\top s\|_\infty,\ \|P_\perp^\top s\|_2\} \tag{2.32}$$

is a well-defined norm, which leads to a P-norm TRS:

$$\begin{aligned} \min_{s\in\mathbb{R}^n} \quad & g^\top s + \frac{1}{2} s^\top B s \\ \text{s. t.} \quad & \|s\|_P \le \Delta. \end{aligned} \tag{2.33}$$

Due to the definition of the $\|.\|_P$, the solution s of the TRS (2.33) can be expressed by

$$s = P s_1 + P_\perp s_2,$$

where s_1 and s_2 can be computed easily. In fact, s_1 is the solution of the box constrained quadratic program (QP):

$$\begin{aligned} \min_{s\in\mathbb{R}^l} \quad & s^\top (P^\top g) + \frac{1}{2} s^\top (\sigma I + D) s \\ \text{s. t.} \quad & \|s\|_\infty \le \Delta, \end{aligned}$$

It can be verified that s_1 has a closed form solution:

$$(s_1)_i = \begin{cases} \frac{-(P^\top g)_i}{\sigma + D_{ii}} & \text{if} \quad |(P^\top g)_i| < (\sigma + D_{ii})\Delta\,, \\ \Delta \mathrm{sign}(-(P^\top g)_i) & \text{otherwise}\,, \end{cases}$$

for $i = 1, ..., l$. On the other hand, s_2 is solution of the 2-norm constrained special QP

$$\begin{aligned} \min_{s\in\mathbb{R}^{n-l}} \quad & s^\top (P_\perp^\top g) + \frac{1}{2}\sigma s^\top s \\ \text{s. t.} \quad & \|s\|_2 \le \Delta. \end{aligned}$$

whose closed-form solution is

$$s_2 = -\min\left(\frac{1}{\sigma}\,,\ \frac{\Delta}{\|P_\perp^\top g\|}\right) P_\perp^\top g\,.$$

Numerical results based on a trust region algorithm that uses the the W-norm TRS were shown in [15].

3. Nonlinear Equations and Nonlinear Least Squares Problem. In this section, we consider the system of nonlinear equations

$$F(x) = 0, \quad x \in \mathbb{R}^n\,, \tag{3.1}$$

and nonlinear least squares problem:

$$\min_{x\in\mathbb{R}^n} \quad \|F(x)\|_2^2, \tag{3.2}$$

where $F(x) = (F^1(x), F^2(x), \ldots, F^m(x))^\top \in \mathbb{R}^m$.

3.1. General Subspace Methods. Due to the special structures of nonlinear equations, several implementations of Newton-like iteration schemes based on Krylov subspace projection methods are considered in [14]. Newton–Krylov methods with a global strategy restricted to a suitable Krylov subspace are developed in [7]. Because the nonlinear least squares problem (3.2) is also an unconstrained optimization problem, all the subspace techniques discussed in Section 2 can be applied. For example, assume that there are i_k linearly independent vectors $\{q_k^1, q_k^2, ..., q_k^{i_k}\}$ which spans $\mathfrak{S}_k$. Let $Q_k = [q_k^1, q_k^2, ..., q_k^{i_k}]$. Then $d \in \mathfrak{S}_k$ can be expressed as $Q_k z$ with respect to a variable $z \in \mathbb{R}^{i_k}$. For (3.1), one can find a subspace step from

$$F(x_k + Q_k z) = 0. \tag{3.3}$$

The linearized system for subproblem (3.3) is

$$F(x_k) + J_k Q_k z = 0, \tag{3.4}$$

where J_k is the Jacobian of F at x_k, Similarly, one can construct a subspace type of the Levenberg-Marquardt method for (3.2) as

$$\min_{z} \quad \|F(x_k) + J_k Q_k z\|_2^2 + \frac{\lambda_k}{2}\|z\|_2^2,$$

where λ_k is a regularization parameter adjusted to ensure global convergence.

3.2. Subspace by Subsampling/Sketching. We start from solving a linear least squares problem on massive data sets:

$$\min_{x} \quad \|Ax - b\|_2^2, \tag{3.5}$$

where $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Although applying the direct or iterative methods to (3.5) is straightforward, it may be prohibitive for large values of m. The sketching technique chooses a matrix $W \in \mathbb{R}^{r\times m}$ with $r \ll m$ and formulates a reduced problem

$$\min_{x} \quad \|W(Ax - b)\|_2^2. \tag{3.6}$$

It can be proved that the solution of (3.6) can be a good approximation to that of (3.5) in certain sense if the matrix W is chosen suitably. For example, one may randomly select r rows from the identity matrix to form W so that WA is a submatrix of A. Another choice is that each element of W is sampled from an i.i.d. normal random variable with mean zero and variance $1/r$. These concepts have been extensively investigated for randomized algorithms in numerical linear algebra [78, 136].

For nonlinear equations, the simple sketching approach is to ignore some equations. Instead of requiring the original system (3.1), we consider

$$F^i(x) = 0, \quad i \in I_k\,, \tag{3.7}$$

which is an incomplete set of equations. To solve the nonlinear equations (3.1) is to find a x at which F maps to the origin [141]. Let $P_k^\top$ be a mapping from $\mathbb{R}^m$ to a lower dimensional subspace. Solving the reduced system

$$P_k^\top F(x) = 0 \tag{3.8}$$

is exactly replacing $F = 0$ by requiring its mapping to the subspace spanned by P_k to be zero. Together with (3.3) yields:

$$P_k^\top F(x_k + Q_k z) = 0, \tag{3.9}$$

The linearized system for subproblem (3.9) is

$$P_k^\top [\, F(x_k) + J_k Q_k z \,] = 0\,. \tag{3.10}$$

Of course, the efficiency of such an approach depends on how to select P_k and Q_k. We can borrow ideas from subspace techniques for large scale linear systems [98]. Instead of using (3.10), we can use a subproblem of the following form:

$$P_k^\top F(x_k) + \hat{J}_k z = 0\,, \tag{3.11}$$

where $\hat{J}_k \in \mathbb{R}^{i_k \times i_k}$ is an approximation to $P_k^\top J_k Q_k$. The reason for preferring (3.11) over (3.10) is that in (3.11) we do not need the Jacobian matrix J_k, whose size is normally significantly larger than that of $\hat{J}_k$.

Similar idea has also been studied for nonlinear least squares problems. At the k-th iteration, we consider minimizing the sum of squares of some randomly selected terms in an index set $I_k \subset \{1, ..., m\}$ instead of all terms:

$$\min_{x \in \mathbb{R}^n} \sum_{i \in I_k} (F^i(x))^2\,. \tag{3.12}$$

This approach works quite well on the distance geometry problem which has lots of applications including protein structure prediction, where the nonlinear least squares of all the terms have lots of local minimizers [103]. Combining subspace with sketching yields a counterpart to (3.9) for nonlinear least squares:

$$\min_{d \in \mathcal{S}_k} \|P_k^\top F(x_k + d)\|_2^2\,. \tag{3.13}$$

A projected nonlinear least squares method is studied in [57] to fit a model ψ to (noisy) measurements y for the exponential fitting problem:

$$\min_{x \in \mathbb{R}^n} \quad \|\psi(x) - y\|_2^2, \tag{3.14}$$

where $\psi(x) \in \mathbb{R}^m$ and $n \ll m$. Since computing the Jacobian of (3.14) can be expensive, a sequence of low-dimensional surrogate problems are constructed from a sequence of subspaces $\{\mathcal{W}_\ell\} \subset \mathbb{R}^m$. Let $P_{\mathcal{W}_\ell}$ be an orthogonal projection onto $\mathcal{W}_\ell$ and W_ℓ is an orthonormal basis for $\mathcal{W}_\ell$, i.e., $P_{\mathcal{W}_\ell} = W_\ell W_\ell^\top$. Then it solves the following minimization problem:

$$\min_x \|P_{\mathcal{W}_\ell}[\psi(x) - y]\|_2^2 = \min_x \|W_\ell^\top \psi(x) - W_\ell^\top y\|_2^2.$$

3.3. Partition of Variables. We now consider the partition of variables, which is also a subspace technique for nonlinear least squares problem. Let I_k be a subset of $\{1, ..., n\}$. The variables are partitioned into two group $x = (\bar{x}\,, \hat{x})$, where $\bar{x} = (x^i, i \in I_k)$ and $\hat{x} = (x^i\,,\ i \notin I_k)$. At the k-th iteration, the variables $x^i (i \notin I_k)$ are fixed and $x^i (i \in I_k)$ are free to be changed in order to obtain a better iterate point. This procedure yields a subproblem with fewer variables:

$$\min_{\bar{x}\in\mathbb{R}^{|I_k|}} \sum_{i=1}^{n} (F^i(\bar{x}, \hat{x}_k))^2\,. \tag{3.15}$$

It is easy to see that partition of variables use special subspaces that spanned by coordinate directions. An obvious generalization of partition of variables is decomposition of the space which uses subspaces spanned by any given directions.

3.4. $\tau-$steepest Descent Coordinate Subspace. The $\tau-$steepest descent coordinate subspace discussed in Section 2 can also be extended to nonlinear equations and nonlinear least squares. Assume that

$$|F^{i_1}(x_k)| > \cdots > |F^{i_\tau}(x_k)| > \cdots \tag{3.16}$$

at the $k-$th iteration. If $F(x)$ is a monotone operator, applying the method in a subspace spanned by the coordinate directions $\{e^{i_j},\ j = 1, ..., \tau\}$ generates a system

$$F^{i_j}(x_k) + d^\top \nabla F^{i_j}(x_k) = 0, \quad j = 1, ..., \tau\,. \tag{3.17}$$

For general nonlinear functions $F(x)$, it is better to replace e^{i_j} by the steepest descent coordinate direction of the function $F^{i_j}(x)$ at x_k, i.e., substituting i_j by an index l_j such that

$$l_j = \arg\max_{t=1,\ldots,n} \left| \frac{\partial F^{i_j}(x_k)}{\partial x^t} \right|\,.$$

However, it may be possible to have two different j at one l_j so that subproblem (3.17) has no solution in the subspace spanned by $\{e^{l_1}, ..., e^{l_\tau}\}$. Further discussion on subspace methods for nonlinear equations and nonlinear least squares can be found in [141].

4. Stochastic Optimization. The supervised learning model in machine learning assumes that (a, b) follows a probability distribution P, where a is an input data and b is the corresponding label. Let $\phi(a, x)$ be a prediction function in a certain functional space and $\ell(\cdot, \cdot)$ represent a loss function to measure the accuracy between the prediction and the true label. The task is to predict a label b from a given input a, that is, finding a function ϕ such that the expected risk $\mathbb{E}[\ell(\phi(a, x), b)]$ is minimized. In practice, the real probability distribution P is unknown, but a data set $\mathcal{D} = \{(a_1, b_1), (a_2, b_2), \cdots, (a_N, b_N)\}$ is obtained by random sampling, where $(a_i, b_i) \sim P$ i.i.d. Then an empirical risk minimization is formulated as

$$\min_x f(x) := \frac{1}{N} \sum_{i=1}^{N} f_i(x), \tag{4.1}$$

where $f_i(x) = \ell(\phi(a_i; x), b_i))$. In machine learning, a large number of problems can be expressed in the form of (4.1). For example, the function ϕ in deep learning is expressed by a deep neural network. Since the size N usually is huge, it is time consuming to use the information of all components $f_i(x)$. However, it is affordable to compute the information at a few samples so that the amount of calculation in each step is greatly reduced.

4.1. Stochastic First-order Methods. In this subsection, we briefly review a few widely used stochastic first-order methods [47]. Instead of using the full gradient $\nabla f(x_k)$, the stochastic gradient method (SGD) for (4.1) selects a uniformly random sample s_k from $\{1, \ldots, N\}$ and updates

$$x_{k+1} = x_k - \alpha_k \nabla f_{s_k}(x_k). \tag{4.2}$$

A common assumption for convergence is that the expectation of the stochastic gradient is equal to the full gradient, i.e.,

$$\mathbb{E}[\nabla f_{s_k}(x_k) \mid x_k] = \nabla f(x_k).$$

When $f_i(x_k)$ is not differentiable, then its subgradient is used in (4.2). Note that only one sample is used in (4.2). The mini-batch SGD method tries to balance between the robustness of the SGD and the computational cost. It randomly selects a mini-batch $I_k \subset \{1, \ldots, N\}$ such that $|I_k|$ is quite small, then computes

$$x_{k+1} = x_k - \frac{\alpha_k}{|I_k|} \sum_{s_k \in I_k} \nabla f_{s_k}(x_k). \tag{4.3}$$

Obviously, subsampling defines a kind of subspace in terms of the component functions $\{f_1(x), \ldots, f_N(x)\}$. For simplicity, we next only consider extensions based on (4.2).

After calculating a random gradient $\nabla f_{s_k}(x_k)$ at the current point, the momentum method does not directly update it to the variable x_k. It introduces a speed variable v, which represents the direction and magnitude of the parameter movements. Formally, the iterative scheme is

$$\begin{aligned} v_{k+1} &= \mu_k v_k - \alpha_k \nabla f_{s_k}(x_k), \\ x_{k+1} &= x_k + v_{k+1}. \end{aligned} \tag{4.4}$$

This new update direction v is a linear combination of the previous update direction v_k and the gradient $\nabla f_{s_k}(x_k)$ to obtain a new v_{k+1}. When $\mu_k = 0$, the algorithm degenerates to SGD. In the momentum method, the parameter μ_k is often in the range of $[0, 1)$. A typical value is $\mu_k \geq 0.5$, which means that the iteration point has a large inertia and each iteration will make a small correction to the previous direction.

Since the dimension of the variable x can be more than 10 million and the convergence speed of each variable may be different, updating all components of x using a single step size α_k may not be suitable. The adaptive subgradient method (AdaGrad) controls the step sizes of each component separately by monitoring the accumulation of the gradients elementwisely:

$$G_k = \sum_{i=1}^{k} \nabla f_{s_i}(x_i) \odot \nabla f_{s_i}(x_i),$$

where $\odot$ the Hadamard product between two vectors. The AdaGrad method is

$$\begin{aligned} x_{k+1} &= x_k - \frac{\alpha_k}{\sqrt{G_k + \epsilon e_n}} \odot \nabla f_{s_{k+1}}(x_{k+1}), \\ G_{k+1} &= G_k + \nabla f_{s_{k+1}}(x_{k+1}) \odot \nabla f_{s_{k+1}}(x_{k+1}), \end{aligned} \tag{4.5}$$

where the division in $\frac{\alpha_k}{\sqrt{G_k + \epsilon e_n}}$ is also performed elementwisely. Adding the term $\epsilon 1_n$ is to prevent the division by zeros.

The adaptive moment estimation (Adam) method combines the momentum and AdaGrad together by adding a few small corrections. At each iteration, it performs:

$$v_k = \rho_1 v_{k-1} + (1 - \rho_1) \nabla f_{s_k}(x_k),$$

$$G_k = \rho_2 G_{k-1} + (1-\rho_2)\nabla f_{s_k}(x_k) \odot \nabla f_{s_k}(x_k),$$
$$\hat{v}_k = \frac{v_k}{1-\rho_1^k},$$
$$\hat{G}_k = \frac{G_k}{1-\rho_2^k},$$
$$x_{k+1} = x_k - \frac{\alpha_k}{\sqrt{\hat{G}_k + \epsilon e_n}} \odot \hat{v}_k,$$

where the typical values for ρ_1 and ρ_2 are $\rho_1 = 0.9$ and $\rho_2 = 0.999$. We can see that the direction v_k is a convex combination of v_{k-1} and $\nabla f_{s_k}(x_k)$, then it is corrected to $\hat{v}_k$. The value $\hat{G}_k$ is also obtained in a similar fashion. The main advantage of Adam is that after the deviation correction, the step size of each iteration has a certain range, making the parameters relatively stable.

The above algorithms have been implemented in mainstream deep learning frameworks, which can be very convenient for training neural networks. The algorithms implemented in Pytorch are AdaDelta, AdaGrad, Adam, Nesterov, RMSProp, etc. The algorithms implemented in Tensorflow are AdaDelta, AdaGradDA, AdaGrad, ProximalAdagrad, Ftrl, Momentum, Adam and CenteredRMSProp, etc.

4.2. Stochastic Second-Order method. The subsampled Newton method takes an additional random set $I_k^H \subset \{1, \ldots, N\}$ independent to I_k and compute a search direction as

$$\left[\frac{1}{|I_k^H|}\sum_{i\in I_k^H}\nabla^2 f_i(x)\right] d_k = -\frac{1}{|I_k|}\sum_{s_k\in I_k}\nabla f_{s_k}(x_k).$$

Therefore, the subspace techniques in section 2 can also be adopted here.

Assume that the loss function is the negative logarithm probability associated with a distribution with a density function $p(y|a,x)$ defined by the neural network and parameterized by x. The so-called KFAC method [79] is based on the Kronecker-factored approximate Fisher matrix. Take an L-layer feed-forward neural network for example. Namely, each layer $j \in \{1, 2, \ldots, L\}$ is given by

$$s_j = T_j w_{j-1}, \quad w_j = \psi_j(s_j), \tag{4.6}$$

where $w_0 = a$ is the input of the neural network, $w_L(x) \in \mathbb{R}^m$ is the output of the neural network under the input a, the constant term 1 is not considered in w_{j-1} for simplicity, T_j is the weight matrix and ψ_j is the block-wise activation function. The jth diagonal block of F corresponding to the parameters in the jth layer using a sample set B can be written in the following way:

$$F^j := Q_{j-1,j-1} \otimes G_{j,j}, \tag{4.7}$$

where

$$Q_{j-1,j-1} = \frac{1}{|B|}\sum_{i\in B} w_{j-1}^i (w_{j-1}^i)^\top,$$
$$G_{j,j} = \frac{1}{|B|}\sum_{i\in B} \mathbb{E}_{z\sim p(z|a_i,x)}[\tilde{g}_j^i(z)\tilde{g}_j^i(z)^\top],$$

and $\tilde{g}_j^i(z) := \frac{\partial \ell(\phi(a_i,x),z)}{\partial s_j^i}$. Therefore, the KFAC method computes a search direction in the jth layer from

$$F^j d_k^j = -g_k^j,$$

where g_k^j is the corresponding subsampled gradient in the jth layer.

5. Sparse Optimization.

5.1. Basis Pursuit. Given a matrix $A \in \mathbb{R}^{m\times n}$ and a vector $b \in \mathbb{R}^m$ such that $m \ll n$, basis pursuit is to find the sparsest signal among all solutions of the equation $Ax = b$. It leads to a NP-hard problem:

$$\min_x \ \|x\|_0, \quad \text{s. t. } Ax = b, \tag{5.1}$$

where $\|x\|_0 = |\{j \mid x_j \neq 0\}|$, i.e., the number of the nonzero elements of x. An exact recovery of the sparest signal often requires the so-called restricted isometry property (RIP), i.e., there exists a constant δ_r such that

$$(1-\delta_r)\|x\|_2^2 \le \|Ax\|_2^2 \le (1+\delta_r)\|x\|_2^2, \text{ whenever } \|x\|_0 \le r.$$

The greedy pursuit methods build up an approximation in a subspace at the k-th iteration. Let T_k be a subset of $\{1,\dots,n\}$, x^{T_k} be a subvector of x corresponding to the set T_k and A_{T_k} be a column submatrix of A whose column indices are collected in the set T_k. Then the subspace problem is

$$x_k^{T_k} = \arg\min_x \quad \frac{1}{2}\|A_{T_k}x - b\|_2^2.$$

Clearly, the solution is $x_k^{T_k} = A_{T_k}^\dagger b$ where $A_{T_k}^\dagger$ is the pseudoinverse of A_{T_k}. Since the size of T_k is controlled to be small, A_{T_k} roughly has full rank column due to the RIP property. All other elements of x_k whose indices are in the complementary set of T_k are set to 0.

We next explain the choices of T_k. Assume that the initial index set T_0 is empty. The orthogonal matching pursuit (OMP) [116] computes the gradient

$$g_k = A^\top(A_{T_k}x_k^{T_k} - b),$$

then selects an index such that $t_k = \arg\max_{j=1,\dots,n} |g_j|$. If multiple indices attain the maximum, one can break the tie deterministically or randomly. Then the index set at the next iteration is augmented as

$$T_{k+1} = T_k \cup \{t_k\}.$$

The CoSaMP [83] method generates an s-sparse solution, i.e., the number of nonzero components is at most s. Let $(x_k)_s$ be a truncation of x_k such that only the s largest entries in the absolute values are kept and all other elements are set to 0. The support of $(x_k)_s$ is denoted as $\mathrm{supp}((x_k)_s)$. Then a gradient g_k is computed at the point $(x_k)_s$ and the set T_{k+1} is updated by

$$T_{k+1} = \mathrm{supp}((g_k)_{2s}) \cup \mathrm{supp}((x_k)_s).$$

5.2. Active Set Methods. Consider the ℓ_1-regularized minimization problem

$$\min_{x\in\mathbb{R}^n} \psi_\mu(x) := \mu\|x\|_1 + f(x), \tag{5.2}$$

where $\mu > 0$ and $f(x) : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. The optimality condition of (5.2) is that there exists a vector

$$(\nabla f(x))^i \begin{cases} = -\mu, & x_i > 0, \\ = +\mu, & x_i < 0, \\ \in [-\mu,\mu], & otherwise. \end{cases} \tag{5.3}$$

A two-stage active-set algorithm called "FPC_AS" is proposed in [133]. First, an active set is identified by a first-order type method using the so-called "shrinkage" operation. Then, a smooth and smaller subproblem is constructed based on this active set and solved by a second-order type method. These two operations are iterated until convergence criteria are satisfied. While shrinkage is very effective in obtaining a support superset, it can take a lot of steps to find the corresponding values. On the other hand, if one imposes the signs of the components of the variable x that are the same as those of the exact solution, problem (5.2) reduces to a small smooth optimization problem, which can be relatively easily solved to obtain x. Consequently, the key components are the identification of a "good" support set by using shrinkage and the construction of a suitable approximate smooth optimization problem.

The iterative shrinkage procedure for solving (5.2) is indeed a proximal gradient method. Given an initial point x_0, the algorithm iteratively computes

$$x_{k+1} := \arg\min_x \quad \mu\|x\|_1 + (x - x_k)^\top g_k + \frac{1}{2\alpha_k}\|x - x_k\|_2^2,$$

where $g_k := \nabla f(x_k)$ and $\alpha_k > 0$. A simple calculation shows that

$$x_{k+1} = \mathcal{S}\left(x_k - \alpha_k g_k, \mu\alpha_k\right), \tag{5.4}$$

where for $y \in \mathbb{R}^n$ and $\nu \in \mathbb{R}$, the shrinkage operator is defined as

$$\begin{aligned}\mathcal{S}(y, \nu) &= \arg\min_x \ \nu\|x\|_1 + \frac{1}{2}\|x - y\|_2^2 \\ &= \operatorname{sign}(y) \odot \max\left\{|y| - \nu, \mathbf{0}\right\}.\end{aligned}$$

Note that the scheme (5.4) first executes a gradient step with a step size α_k, then performs a shrinkage. In practice, α_k can be computed by a non-monotone line search in which the initial value is set to the BB step size. The convergence of (5.4) has been studied in [53] under suitable conditions on α_k and the Hessian $\nabla^2 f$. An appealing feature proved in [53] is that (5.4) yields the support and the signs of the minimizer x^* of (5.2) after a finite number of steps under favorable conditions. For more references related to shrinkage, the reader is referred to [133].

We now describe the subspace optimization in the second stage. For a given vector $x \in \mathbb{R}^n$, the active set is denoted by $\mathcal{A}(x)$ and the inactive set (or support) is denoted by $\mathcal{I}(x)$ as follows

$$\mathcal{A}(x) := \{i \in \{1, \cdots, n\} \,|\, |x^i| = 0\} \text{ and } \mathcal{I}(x) := \{i \in \{1, \cdots, n\} \,|\, |x^i| > 0\}. \tag{5.5}$$

We require that each component x^i either has the same sign as x_k^i or is zero, i.e., x is required to be in the set

$$\Omega(x_k) := \left\{x \in \mathbb{R}^n : \operatorname{sign}(x_k^i)x^i \geq 0, i \in \mathcal{I}(x_k) \text{ and } x^i = 0, i \in \mathcal{A}(x_k)\right\}. \tag{5.6}$$

Then, a smooth subproblem is formulated as either an essentially unconstrained problem

$$\min_x \quad \mu \operatorname{sign}(x_k^{\mathcal{I}_k})^\top x^{\mathcal{I}_k} + f(x), \text{ s. t. } x^i = 0, i \in \mathcal{A}(x_k) \tag{5.7}$$

or a simple bound-constrained problem

$$\min_x \quad \mu \operatorname{sign}(x_k^{\mathcal{I}_k})^\top x^{\mathcal{I}_k} + f(x), \text{ s. t. } x \in \Omega(x_k). \tag{5.8}$$

Since the size of the variables in (5.7) and (5.8) is relatively small, these two problems can be solved efficiently by methods such as L-BFGS-B. If $f(x)$ is quadratic, problem (5.7) can be solved by the CG method for a system of linear equations.

The active set strategies have also been studied in [105, 64]. Specifically, the method in [64] solves a smooth quadratic subproblem determined by the active sets and invokes a corrective cycle that greatly improves the efficiency and robustness of the algorithm. The method is globalized by using a proximal gradient step to check the desired progress.

6. The Domain Decomposition Methods.

6.1. A Two-level Subspace Method. Consider an infinite dimensional minimization problem

$$\min_{\mathbf{x}\in\mathcal{V}} \quad \mathcal{F}(\mathbf{x}), \tag{6.1}$$

where $\mathcal{F}$ is a mapping from an infinite-dimensional space $\mathcal{V}$ to $\mathbb{R}$. Many large-scale finite dimensional optimization problems arise from infinite dimensional optimization problems [28]. Since explicit solutions for these problems are usually not available, we solve the discretized version of them from the "discretize-then-optimize" strategy by using the concept of multilevel optimization.

Let $\mathcal{V}_h$ be a finite dimensional subset of $\mathcal{V}$ at the grid level h, for example, a standard finite element space associated with the grid level h. For consecutive coarser levels, we choose nested spaces, so that $\mathcal{V}_1 \subset \cdots \subset \mathcal{V}_{\mathrm{N}-1} \subset \mathcal{V}_{\mathrm{N}} \subset \mathcal{V}$, where N is reserved for the index of the finest level and 1 for the coarsest level. The functional $\mathcal{F}(\mathbf{x})$ restricted on $\mathcal{V}_h$ is constructed as $f_h(\mathbf{x}_h)$ for $\mathbf{x}_h \in \mathcal{V}_h$. Therefore, the discretization of problem (6.1) is

$$\min_{\mathbf{x}_h\in\mathcal{V}_h} \quad f_h(\mathbf{x}_h). \tag{6.2}$$

Let $\mathbf{x}_{h,k}$ be a vector where the first subscript h denotes the discretization level of the multigrid and the second subscript k denotes the iteration count. We next briefly describe a two-level subspace method in [24] instead of simply finding a point $\mathbf{x}_{h,k+1}$ in the coarser grid space $\mathcal{V}_H$. We seek a point $\mathbf{x}_{h,k+1}$ in $\mathfrak{S}_{h,k} + \mathcal{V}_H$, satisfying some conditions, where $S_{h,k}$ is a subspace including the descent information, such as the coordinate direction of the current iteration and the previous iterations or the gradient $\mathcal{D}_h f(\mathbf{x}_{h,k})$ in the finite space $\mathcal{V}_h$. Then, we solve

$$\mathfrak{S}_{h,k} = \mathrm{span}\{\mathbf{x}_{h,k-1}, \mathbf{x}_{h,k}, \mathcal{D}_h f(\mathbf{x}_{h,k})\} \subseteq \mathcal{V}_h. \tag{6.3}$$

When $\mathbf{x}_{h,k}$ is not optimal on a coarse level $H \in \{1, 2, \ldots, N\}$, we go to this level and compute a new solution $\mathbf{x}_{h,k+1}$ by

$$\mathbf{x}_{h,k+1} \approx \arg\min\ f(\mathbf{x}), \quad \mathrm{s.t.} \quad \mathbf{x} \in \mathfrak{S}_{h,k} + \mathcal{V}_H. \tag{6.4}$$

Otherwise, we find a point $\mathbf{x}_{h,k+1} \in \mathcal{V}_h$ on level h.

The so-called full multigrid skill or mesh refinement technique can often help to generate a good initial point so that the total number of iterations is reduced. Firstly, we solve the target problem at the coarsest level which is computationally cheap. After an approximate solution $\mathbf{x}_h^*$ at the current level is obtained, we prolongate it to the next finer level $h+1$ with interpolation as an initial point, and then apply the two-level subspace method at this new level to find a solution $\mathbf{x}_{h+1}^*$. This process is repeated until the finest level is reached.

6.2. The Subspace Correction Method. We next briefly review the subspace correction methods [112]. Given the current point $\mathbf{x}_{h,k}$, the relaxation (or smoothing) procedure is the operation on the current level h, namely, to find a direction $d_{h,k}$ to approximate the solution of

$$\min_{\mathbf{d}\in\mathcal{V}_h} \quad f_h(\mathbf{x}_{h,k}+\mathbf{d}), \tag{6.5}$$

and the coarse grid correction procedure is the operation on the coarse level H, namely, to find a direction $\mathbf{d}_{h,k}$ to approximate the solution

$$\min_{\mathbf{d}\in\mathcal{V}_H} \quad f_h(\mathbf{x}_{h,k}+\mathbf{d}). \tag{6.6}$$

The concept of the subspace correction methods can be used to solve subproblems (6.5) and (6.6). Let $\{\phi_h^{(j)}\}_{j=1}^{n_h}$ be a basis for $\mathcal{V}_h$, where n_h is the dimension of $\mathcal{V}_h$. Denote $\mathcal{V}_h$ as a direct sum of the one-dimensional subspaces $\mathcal{V}_h = \mathcal{V}_h^{(1)} \oplus \cdots \oplus \mathcal{V}_h^{(n_h)}$. Then for each $j = 1, \cdots, n_h$ in turn, we perform the following correction step for subproblem (6.5) at the k-th iteration:

$$\begin{cases} \mathbf{d}_{h,k}^{(j)^*} = \min\limits_{\mathbf{d}_{h,k}^{(j)}\in\mathcal{V}_h^{(j)}} \quad f_h(\mathbf{x}_{h,k}+\mathbf{d}_{h,k}^{(j)}) \\ \mathbf{x}_{h,k} = \mathbf{x}_{h,k} + \mathbf{d}_{h,k}^{(j)^*}. \end{cases} \tag{6.7}$$

For subproblem (6.6), a similar strategy can be adopted by decompose space $\mathcal{V}_H$ into a direct sum. Global convergence of this algorithm has been proved in [113] for strictly convex functions under some assumptions. The subspace correction method can be viewed as a generalization of the coordinate search method or the pattern search method.

6.3. Parallel Line Search Subspace Correction Method. In this subsection, we consider the following optimization problem

$$\min_{x\in\mathbb{R}^n} \ \varphi(x) := f(x) + h(x), \tag{6.8}$$

where $f(x)$ is differentiable convex function and $h(x)$ is a convex function that is possibly nonsmooth. The ℓ_1-regularized minimization (LASSO) [114] and the sparse logistic regression [100] are examples of (6.8). The PSC methods have been studied for LASSO in [36, 39] and total variation minimization in [37, 38, 39, 68].

Suppose that $\mathbb{R}^n$ is split into p subspaces, namely,

$$\mathbb{R}^n = X^1 + X^2 + \cdots + X^p, \tag{6.9}$$

where

$$X^i = \{x \in \mathbb{R}^n | \text{supp}(x) \subset \mathcal{J}_i\}, \quad 1 \le i \le p,$$

such that $\mathcal{J} := \{1, ..., n\}$ and $\mathcal{J} = \bigcup\limits_{i=1}^{p} \mathcal{J}_i$. For any $i \neq j, 1 \le i, j \le p$, $\mathcal{J}_i \bigcap \mathcal{J}_j = \emptyset$ holds in a non-overlapping domain decomposition of $\mathbb{R}^n$. Otherwise, there exist $i, j \in \{1, ..., p\}$ and $i \neq j$ such that $\mathcal{J}_i \bigcap \mathcal{J}_j \neq \emptyset$ in an overlapping domain decomposition of $\mathbb{R}^n$.

Let φ_k^i be a surrogate function of φ restricted to the i-th subspace at k-th iteration. The PSC framework for solving (6.8) is:

$$d_k^i = \arg\min_{d^i\in X^i} \varphi_k^i(d^i), \ i = 1, ..., p, \tag{6.10}$$

$$x_{k+1} = x_k + \sum_{i=1}^{p} \alpha_k^i d_k^i.$$

The convergence can be proved if the step sizes α_k^i $(1 \leq i \leq p)$ satisfy the conditions: $\sum_{i=1}^{p} \alpha_k^i \leq 1$ and $\alpha_k^i > 0$ $(1 \leq i \leq p)$. Usually, the step size α_k^i is quite small under these conditions and convergence becomes slow. For example, the diminishing step size $\alpha_k^i = \frac{1}{p}$ tends to be smaller and smaller as the number of subspaces increases.

A parallel subspace correction method (PSCL) with the Armijo backtracking line search for a large step size is proposed in [29]. At the k-th iteration, it chooses a surrogate functions φ_k^i and solves the subproblem (6.10) for each block, then computes a summation of the direction $d_k = \sum_{i=1}^{p} d_k^i$. The next iteration is

$$x_{k+1} = x_k + \alpha_k d_k,$$

where α_k satisfies the Armijo backtracking conditions. When $h(x) = 0$ and $f(x)$ is strongly convex, the surrogate function can be set to the original objective function φ. Otherwise, it can be a first-order Taylor expansion of the smooth part $f(x)$ with a proximal term and the nonsmooth part $h(x)$:

$$\varphi_k^i(d^i) = \nabla f(x_k)^\top d^i + \frac{1}{2\lambda^i}\|d^i\|_2^2 + h(x_k + d^i), \text{ for } d^i \in X^i. \tag{6.11}$$

Both non-overlapping and overlapping schemes can be designed for PSCL.

The directions from different subproblems can be equipped with different step sizes. Let $Z_k = \left(d_k^1, d_k^2, \ldots, d_k^p\right)$. The next iteration is set to

$$x_{k+1} = x_k + Z_k \alpha_k.$$

One can find α_k as an optimal solution of

$$\alpha_k = \underset{\alpha \in \mathbb{R}^p}{\arg\min}\ \varphi(x_k + Z_k \alpha).$$

Alternatively, we can solve the following approximation:

$$a_k \approx \underset{\alpha \in \mathbb{R}^p}{\arg\min} \quad \nabla f(x_k)^\top Z_k \alpha + \frac{1}{2t_k}\|Z_k \alpha\|_2^2 + h(x_k + Z_k a).$$

The global convergence of PSCL is established by following the convergence analysis of the subspace correction methods for strongly convex problem [112], the active-set method for l_1 minimization [134] and the BCD method for nonsmooth separable minimization [119]. Specifically, linear convergence rate is proved for the strongly convex case and convergence to the solution set of problem (6.8) globally is obtained for the general nonsmooth case.

7. General Constrained Optimization. In this section, we first present subspace methods for solving general equality constrained optimization problems:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & c(x) = 0, \end{aligned} \tag{7.1}$$

where $c(x) = (c_1(x), \cdots, c_m(x))^\top$, $f(x)$ and $c_i(x)$ are real functions defined in $\mathbb{R}^n$ and at least one of the functions $f(x)$ and $c_i(x)$ is nonlinear. Note that inequality constraints can also be added to (7.1) but they are omitted to simplify our discussion in the first few subsections.

In the last subsection, we discuss methods for bound-constrained minimization problems. Problem (7.1) is often minimized by computing solutions of a sequence of subproblems which are simpler than (7.1) itself. However, they are still large-scale linear or quadratic problems because normally subproblems are also defined in the same dimensional space as the original nonlinear problem.

7.1. Direct Subspace Techniques. The sequential quadratic programming (SQP) is an important method for solving (7.1). It successively minimizes quadratic approximations to the Lagrangian function subject to the linearized constraints. Let $Q_k(d)$ be a quadratic approximation to the Lagrangian function of (7.1) at the k-th iteration:

$$Q_k(d) = g_k^\top d + \frac{1}{2} d^\top B_k d, \tag{7.2}$$

where $g_k = \nabla f(x_k)$ and B_k is an approximation to the Hessian of the Lagrangian function. The search direction d_k of a line search type SQP method is obtained by solving the following QP subproblem

$$\min_{d \in \mathbb{R}^n} \quad Q_k(d) \tag{7.3}$$

$$\text{s. t.} \quad c(x_k) + A_k^\top d = 0, \tag{7.4}$$

where $A_k = \nabla c(x_k)$. Although the SQP subproblem is simpler than (7.1), it is still difficult when the dimension n is large.

In general, the subspace SQP method constructs the search direction d_k by solving a QP in a subspace:

$$\begin{aligned} \min \quad & Q_k(d) \\ \text{s. t.} \quad & c_k + A_k^\top d = 0, \qquad d \in \mathfrak{S}_k \,, \end{aligned} \tag{7.5}$$

where $\mathfrak{S}_k$ is a subspace. Lee et al. [70] considered the following choice:

$$\mathfrak{S}_k = \text{span}\{g_k, s_{k-\bar{m}}, ..., s_{k-1}, \bar{y}_{k-\bar{m}}, ..., \bar{y}_{k-1}, \nabla c_1(x_k), ..., \nabla c_m(x_k)\},$$

where $\bar{m}$ is the memory size of the limited memory BFGS method for constructing B_k in (7.2), and $\bar{y}_i$ is a linear combination of y_i and $B_i s_i$. Let U_k be a matrix of linearly independent vectors in $\mathfrak{S}_k$. A reduced constrained version of (7.5) is

$$\begin{aligned} \min_z \quad & (U_k^\top g_k)^\top z + \frac{1}{2} z^\top U_k^\top B_k U_k z \\ \text{s. t.} \quad & T_k^\top (c_k + A_k^\top U_k z) = 0, \end{aligned} \tag{7.6}$$

where T_k is a projection matrix so that the constraints are not over-determined.

7.2. Second Order Correction Steps. The SQP step d_k can be decomposed into two parts $d_k = h_k + v_k$ where $v_k \in \text{Range}(A_k)$ and $h_k \in (A_k^\top)$. Thus, v_k is a solution of the linearized constrained (7.4) in the range space of A_k, while h_k is the minimizer of the quadratic function $Q_k(v_k + d)$ in the null space of $A_k^\top$.

One good property of the SQP method is its superlinear convergence rate, namely when x_k is close to a Karush–Kuhn–Tucker (KKT) point x^* it holds

$$x_k + d_k - x^* = o(\|x_k - x^*\|) \,. \tag{7.7}$$

However, a superlinearly convergent step d_k may generate a point that seems "bad" since it may increase both the objective function and the constraint violations. Even though (7.7) holds, the Maratos effect shows that it is possible for the SQP step d_k to have both

$$f(x_k + d_k) > f(x_k), \qquad \|c(x_k + d_k)\| > \|c(x_k)\|.$$

The second order correction step method [35, 80] solves a QP subproblem whose constraints (7.4) are replaced by

$$c(x_k + d_k) + A_k^\top (d - d_k) = 0, \tag{7.8}$$

because the left hand side of (7.8) is a better approximation to $c(x_k + d)$ close to the point $d = d_k$. Since the modification of the constraints is a second order term, the new step can be viewed as the SQP step d_k adding a second order correction step $\hat{d}_k$. Consequently, the Maratos effect is overcomed. For detailed discussions on the SQP method and the second order correction step, we refer the reader to [111].

We now examine the second order correction step from subspace point of views. It can be verified that the second order correction step $\hat{d}_k$ is a solution of

$$\begin{aligned} \min_{d \in \mathbb{R}^n} \quad & Q_k(d_k + d) \\ \text{s.t.} \quad & c(x_k + d_k) + A_k^\top d = 0\,. \end{aligned}$$

Compute the QR factorization

$$A_k = [Y_k, Z_k] \begin{bmatrix} R_k \\ 0 \end{bmatrix}$$

and assume that R_k is nonsingular. Therefore, the second order correction step can be represented as $\hat{d}_k = \hat{v}_k + \hat{h}_k$, where $\hat{v}_k = -Y_k R_k^{-T} c(x_k + d_k)$ and $\hat{h}_k$ is the minimizer of

$$\min_{h \in \mathrm{Null}(A_k^\top)} Q(d_k + \hat{v}_k + h)\,. \tag{7.9}$$

Since d_k is the SQP step, it follows that $g_k + B_k d_k \in \mathrm{Range}(A_k)$, which implies that the minimization problem (7.9) is equivalent to

$$\min_{h \in \mathrm{Null}(A_k^\top)} \frac{1}{2}(\hat{v}_k + h)^\top B_k (\hat{v}_k + h)\,. \tag{7.10}$$

Examining the SQP method from the perspective of subspace enables us to get more insights. If $Y_k^\top B_k Z_k = 0$, it holds $\hat{h}_k = 0$, which means that the second order correction step $\hat{d}_k \in \mathrm{Range}(A_k)$ is also a range space step. Hence, the second order correction uses two range space steps and one null space step. Note that a range space step is fast since it is a Newton step, while a null space step is normally slower because B_k is often approximated by a quasi-Newton approximation to the Hessian of the Lagrangian function. Intuitively, it might be better to have two slower steps with one fast step. Therefore, it might be reasonable to study a correction step $\hat{d}_k \in \mathrm{Null}(A_k^\top)$ in a modified SQP method.

7.3. The Celis-Dennis-Tapia (CDT) Subproblem. The CDT subproblem [23] is often needed in some trust region algorithms for constrained optimization. It has two trust region ball constraints:

$$\begin{aligned} \min_{d \in \mathbb{R}^n} \quad & Q_k(d) = g_k^\top d + \frac{1}{2} d^\top B_k d \\ \text{s.t.} \quad & \|c_k + A_k^\top d\|_2 \le \xi_k, \quad \|d\|_2 \le \Delta_k, \end{aligned} \tag{7.11}$$

where ξ_k and Δ_k are both trust region radii. Let $S_k = \text{span}\{Z_k\}$, $Z_k^\top Z_k = I$, $\text{span}\{A_k, g_k\} \subset S_k$ and $B_k u = \sigma u$, $\forall u \in S_k^\perp$. It is shown in [50] that the CDT subproblem is equivalent to

$$\begin{aligned} \min_{\bar{d}\in\mathbb{R}^r} \quad & \bar{Q}_k(\bar{d}) = \bar{g}_k^\top \bar{d} + \frac{1}{2}\bar{d}^\top \bar{B}_k \bar{d} \\ \text{s.t.} \quad & \|c_k + \bar{A}_k^\top \bar{d}\|_2 \le \xi_k, \quad \|\bar{d}\|_2 \le \Delta_k\,, \end{aligned}$$

where $\bar{g}_k = Z_k^\top g_k$, $\bar{B}_k = Z_k^\top B_k Z_k$ and $\bar{A}_k = Z_k^\top A_k$. Consequently, a subspace version of the Powell-Yuan trust algorithm [91] was developed in [50].

7.4. Simple Bound-constrained Problems. We now consider the optimization problems with simple bound-constraints:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & l \le x \le u, \end{aligned} \tag{7.12}$$

where l and u are two given vectors in $\mathbb{R}^n$. In this subsection, the superscript of a vector denotes its indices, for example, x^i is the ith component of x.

A subspace adaptation of the Coleman-Li trust region and interior method is proposed in [12]. The affine scaling matrices D_k and C_k are defined from examining the KKT conditions of (7.12) as:

$$D_k = D(x_k) = \text{diag}(|v(x_k)|^{-1/2}), \quad C_k = D_k \text{diag}(g_k) J_k^v D_k$$

where $J^v(x)$ is a diagonal matrix whose diagonal elements equal to zero or ± 1, and

$$v^i = \begin{cases} x^i - u^i, & \text{if } g^i < 0 \text{ and } u^i < +\infty, \\ x^i - l^i, & \text{if } g^i \ge 0 \text{ and } l^i > -\infty, \\ -1, & \text{if } g^i < 0 \text{ and } u^i = +\infty, \\ +1, & \text{if } g^i \ge 0 \text{ and } l^i = -\infty. \end{cases}$$

Let H_k be an approximation to the Hessian matrix $\nabla^f(x_k)$ and define

$$\hat{g}_k = D_k^{-1} g_k, \quad \hat{M}_k = D_k^{-1} H_k D_k^{-1} + \text{diag}(g_k) J_k^v.$$

Then the subspace trust region subproblem is

$$\begin{aligned} \min_{s} \quad & g_k^\top s + \frac{1}{2} s^\top (H_k + C_k) s \\ \text{s.t.} \quad & \|D_k s\|_2 \le \Delta_k, s \in \mathfrak{S}_k. \end{aligned} \tag{7.13}$$

If the matrix $\hat{M}_k$ is positive definite, the subspace is taken as

$$\mathfrak{S}_k = \text{span}\{D_k^{-2} g_k, w_k\},$$

where w_k is either $\hat{s}_k^N = -\hat{M}_k^{-1}\hat{g}_k$ or its inexact version. Otherwise, $\mathfrak{S}_k$ is set to

$$\text{span}\{D_k^{-2}\text{sign}(g_k)\} \text{ or } \text{span}\{D_k^{-2}\text{sign}(g_k), w_k\},$$

where $\hat{w}_k$ is a vector of nonpositive curvature of $\hat{M}_k$.

A subspace limited memory quasi-Newton method is developed by Ni and Yuan in [87]. There are three types of search directions: a subspace quasi-Newton direction, subspace gradient and modified gradient directions. The limited memory quasi-Newton method is used to update the variables with indices outside of the active set, while the projected gradient method is used to update the active variables. An active set algorithm is designed in [52]. The algorithm consists of a nonmonotone gradient projection step, an unconstrained optimization step, and a set of rules for branching between the two steps. After a suitable active set is detected, some components of variables are fixed and the method is switched to the unconstrained optimization algorithm in a lower-dimensional space.

8. Eigenvalue Computation. The eigenvalue decompostion (EVD) and singular value decomposition (SVD) are fundamental computational tools with extraordinarily wide-ranging applications in science and engineering. For example, most algorithms in high dimensionality reduction, such as the principal component analyses (PCA), the multidimensional scaling, Isomap and etc, use them to transform the data into a meaningful representation of reduced dimensionality. More recently, identifying dominant eigenvalue or singular value decompositions of a sequence of closely related matrices has become an indispensable algorithmic component for many first-order optimization methods for various convex and nonconvex optimization problems, such as semidefinite programming, low-rank matrix computation, robust principal component analysis, sparse principal component analysis, sparse inverse covariance matrix estimation, nearest correlation matrix estimation and the self-consistent iteration in electronic struture calculation. The computational cost of these decompositions is a major bottleneck which significantly affects the overall efficiency of these algorithms.

For a given real symmetric matrix $A \in \mathbb{R}^{n\times n}$, let $\lambda_1, \lambda_2, \cdots, \lambda_n$ be the eigenvalues of A sorted in a descending order: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, and $q_1, \ldots, q_n \in \mathbb{R}^n$ be the corresponding eigenvectors such that $Aq_i = \lambda_i q_i$, $\|q_i\|_2 = 1$, $i = 1, \ldots, n$ and $q_i^\top q_j = 0$ for $i \neq j$. The eigenvalue decomposition of A is defined as $A = Q_n \Lambda_n Q_n^\top$, where, for any integer $i \in [1, n]$,

$$Q_i = [q_1,\ q_2,\ \ldots, q_i] \in \mathbb{R}^{n\times i}, \quad \Lambda_i = \mathrm{Diag}(\lambda_1, \lambda_2, \ldots, \lambda_i) \in \mathbb{R}^{i\times i}, \tag{8.1}$$

and $\mathrm{Diag}(\cdot)$ denotes a diagonal matrix with its arguments on the diagonal. For simplicity, we also write $A = Q\Lambda Q^\top$ where $Q = Q_n$ and $\Lambda = \Lambda_n$. Without loss of generality, we assume for convenience that A is positive definite (after a shift if necessary). Our task is to compute p largest eigenpairs (Q_p, Λ_p) for some $p \ll n$ where by definition $AQ_p = Q_p\Lambda_p$ and $Q_p^\top Q_p = I \in \mathbb{R}^{p\times p}$. Replacing A by a suitable function of A, say $\lambda_1 I - A$, one can also in principle apply the same algorithms to finding p smallest eigenpairs as well.

We next describe the Rayleigh-Ritz (RR) step which is to extract approximate eigenpairs, called Ritz-pairs, from the range space $\mathcal{R}(Z)$ spanned by a given matrix $Z \in \mathbb{R}^{n\times m}$. This procedure is widely used as an important component for an approximation to a desired m-dimensional eigenspace of A. It consists of the following four steps.

(i) Given $Z \in \mathbb{R}^{n\times m}$, orthonormalize Z to obtain $U \in \mathrm{orth}(Z)$, where $\mathrm{orth}(Z)$ is the set of orthonormal bases for the range space of Z

(ii) Compute $H = U^\top AU \in \mathbb{R}^{m\times m}$, the projection of A onto the range space of U.

(iii) Compute the eigenvalue decomposition $H = V^\top \Sigma V$, where $V^\top V = I$ and Σ is diagonal.

(iv) Assemble the Ritz pairs (Y, Σ) where $Y = UV \in \mathbb{R}^{n\times m}$ satisfies $Y^\top Y = I$.

The RR procedure is denoted as a map $(Y, \Sigma) = \mathrm{RR}(A, Z)$ where the output (Y, Σ) is a Ritz pair block.

8.1. Classic Subspace Iteration. The simple (simultaneous) subspace iteration (SSI) method [95, 96, 108, 110] is an extension of the power method which computes a single eigen-

pair corresponding to the largest eigenvalue in magnitude. Starting from an initial matrix U, SSI repeatedly performs the matrix-matrix multiplications AU, followed by an orthogonalization and RR projection, i.e.,

$$Z = \mathrm{orth}(AU), \quad U = \mathrm{RR}(A, Z). \tag{8.2}$$

The major purpose of orthogonalization is to guarantee the full-rankness of the matrix Z since AU may lose rank numerically. The so-called deflation can be executed after each RR projection to fix the numerically converged eigenvectors since the convergence rates for different eigenpairs are not the same. Moreover, q extra vectors, often called guard vectors, are added to U to accelerate convergence. Although the iteration cost is increased at the initial stage, the overall performance may be better.

Due to fast memory access and highly parallelizable computation on modern computer architectures, simultaneous matrix-block multiplications have advantages over individual matrix-vector multiplications. Whenever there is a gap between the p-th and the $(p+1)$-th eigenvalues of A, the SSI method is ensured to converge to the largest p eigenpairs from any generic starting point. However, the convergence speed of the SSI method depends critically on eigenvalue distributions. It can be intolerably slow if the eigenvalue distributions are not favorable.

8.2. Polynomial Filtering. The idea of polynomial filtering is originated from a well-known fact that polynomials are able to manipulate the eigenvalues of any symmetric matrix A while keeping its eigenvectors unchanged. Due to the eigenvalue decomposition (8.1), it holds that

$$\rho(A) = Q\rho(\Lambda)Q^T = \sum_{i=1}^{n} \rho(\lambda_i) q_i q_i^T, \tag{8.3}$$

where $\rho(\Lambda) = \mathrm{diag}(\rho(\lambda_1), \rho(\lambda_2), \ldots, \rho(\lambda_n))$. Ideally, the eigenvalue distribution $\rho(A)$ is more favorable than the original one.

The convergence of the desired eigen-space of SSI is determined by the gap of the eigenvalues, which can be very slow if the gap is nearly zero. Polynomial filtering has been used to manipulate the gap in eigenvalue computation through various ways [97, 109, 150, 34] in order to obtain a faster convergence. One popular choice of $\rho(t)$ is the Chebyshev polynomial of the first kind, which can be written as

$$T_d(t) = \begin{cases} \cos(d \arccos t) & |t| \le 1, \\ \frac{1}{2}((t - \sqrt{t^2-1})^d + (t + \sqrt{t^2-1})^d) & |t| > 1, \end{cases} \tag{8.4}$$

where d is the degree of the polynomial. Since Chebyshev polynomials grow pretty fast outside the interval $[-1, 1]$, they can help to suppress all unwanted eigenvalues in this interval efficiently. For these eigenvalues in a general interval $[a, b]$, the polynomial can be chosen as

$$\rho(t) = T_d\left(\frac{t - (b+a)/2}{(b-a)/2}\right). \tag{8.5}$$

From an initial matrix U, the polynomial-filtered subspace iteration is simply

$$Z = \mathrm{orth}(\rho(A)U), \quad U = \mathrm{RR}(A, Z). \tag{8.6}$$

8.3. Limited Memory Methods. Finding a p-dimensional eigenspace associated with p largest eigenvalues of A is equivalent to solving a trace maxmization problem with orthogonality constraints:

$$\max_{X \in \mathbb{R}^{n\times p}} \operatorname{tr}(X^\top AX), \text{ s. t. } X^\top X = I. \tag{8.7}$$

The first-order optimality conditions of (8.7) are

$$AX = X\Lambda, \quad X^\top X = I,$$

where $\Lambda = X^\top AX \in \mathbb{R}^{p\times p}$ is the matrix of Lagrangian multipliers. Once the matrix Λ is diagonalized, the matrix pair (Λ, X) provides p eigenpairs of A. When maximization is replaced by minimization, (8.7) computes an eigenspace associated with p smallest eigenvalues. A few block algorithms have been designed based on solving (8.7), including the locally optimal block preconditioned conjugate gradient method (LOBPCG) [65] and the limited memory block Krylov subspace optimization method (LMSVD) [74]. At each iteration, these methods in fact solve a subspace trace maximization problem of the form

$$Y = \arg\max_{X \in \mathbb{R}^{n\times p}} \left\{ \operatorname{tr}(X^\top AX) : X^\top X = I,\ X \in \mathfrak{S} \right\}. \tag{8.8}$$

Obviously, the closed-form solution of (8.8) can be obtained by using the RR procedure.

The subspace $\mathfrak{S}$ is varied from method to method. In LOBPCG, $\mathfrak{S}$ is the span of the two most recent iterations X_{i-1} and X_i, and the residual $AX_i - X_i\Lambda_i$ at X_i, which is essentially equivalent to

$$\mathfrak{S} = \operatorname{span}\{X_{i-1}, X_i, AX_i\}. \tag{8.9}$$

The term AX_i can be pre-multiplied by a pre-conditioning matrix if it is available. The LMSVD method constructs the subspace $\mathfrak{S}$ as a limited memory of the current i-th iterate and the previous t iterates; i.e.,

$$\mathfrak{S} = \operatorname{span}\{X_i, X_{i-1}, ..., X_{i-t}\}. \tag{8.10}$$

In general, the subspace $\mathfrak{S}$ should be constructed such that the computational cost of solving the subproblem (8.8) is relatively small.

8.4. Augmented Rayleigh-Ritz Method. We next introduce the augmented Rayleigh-Ritz (ARR) procedure. It is easy to see that the RR map $(Y, \Sigma) = \mathrm{RR}(A, Z)$ is equivalent to solving the trace-maximization subproblem (8.8) with the subspace $\mathfrak{S} = \mathcal{R}(Z)$, while requiring $Y^\top AY$ to be a diagonal matrix Σ. For a fixed number p, the larger the subspace $\mathcal{R}(Z)$ is, the greater chance there is to extract better Ritz pairs. The augmentation of the subspaces in LOGPCG and LMSVD is the main reason why they generally achieve faster convergence than the classic SSI.

The augmentation in ARR is based on a block Krylov subspace structure, i.e., for some integer $t \geq 0$,

$$\mathfrak{S} = \operatorname{span}\{X, AX, A^2X, \ldots, A^tX\}. \tag{8.11}$$

Then the optimal solution of the trace maximization problem (8.8), restricted in the subspace $\mathfrak{S}$ in (8.11), is computed via the RR procedure using $(\hat{Y}, \hat{\Sigma}) = \mathrm{RR}(A, K_t)$, where $K_t = [X, AX, A^2X, \ldots, A^tX]$. Finally, the p leading Ritz pairs (Y, Σ) is extracted from $(\hat{Y}, \hat{\Sigma})$. This augmented RR procedure is simply referred as ARR. It looks identical to a

block Lanczos algorithm. However, a fundamental dissimilarity is that the ARR is primarily developed to compute a relatively large number of eigenpairs by using only a few augmentation blocks.

We next describe an "Arrabit" algorithmic framework with two main steps at each outer iteration: a subspace update (SU) step and an ARR projection step, for computing a subset of eigenpairs of large matrices. The goal of the subspace update step is finding a matrix $X \in \mathbb{R}^{n\times p}$ so that its column space is a good approximation to the p-dimensional eigenspace spanned by p desired eigenvectors. Once X is obtained, the projection step aims to extract from X a set of approximate eigenpairs that are optimal in certain sense. The SU step is often performed on a transformed matrix $\rho(A)$, where $\rho(t) : \mathbb{R} \to \mathbb{R}$ is a suitable polynomial function. For a reasonable choice $X \in \mathbb{R}^{n\times p}$, it follows from (8.3) that $\rho(A)X \approx Q_p Q_p^T X$ would be an approximate basis for the desired eigenspace. The analysis of ARR in [135, Corollary 4.6] shows that the convergence rate of SSI is improved from $|\rho(\lambda_{p+1})/\rho(\lambda_p)|$ for RR ($t = 0$) to $|\rho(\lambda_{(t+1)p+1})/\rho(\lambda_p)|$ for ARR ($t > 0$). Therefore, a significant improvement is possible with a suitably chosen polynomial $\rho(\cdot)$ such that $|\rho(\lambda_{(t+1)p+1})| \ll |\rho(\lambda_{p+1})|$.

In principle, the SU step can be fulfilled by many kinds of updating schemes without explicit orthogonalizations. The Gauss-Newton (GN) algorithm in [75] solves the nonlinear least squares problem:

$$\min \|XX^\top - A\|_F^2.$$

For any full-rank matrix $X \in \mathbb{R}^{n\times p}$, it takes the simple closed form

$$X_+ = X + \alpha \left(I - \frac{1}{2} X(X^\top X)^{-1} X^\top \right) \left(AX(X^\top X)^{-1} - X \right),$$

where the parameter $\alpha > 0$ is a step size. The classic power iteration can be modified without orthogonalization at each step. For $X = [x^1\ x^2\ \cdots, x^m] \in \mathbb{R}^{n\times m}$, the power iteration is applied individually to all columns of the iterate matrix X, i.e.,

$$x^i = \rho(A)x^i \quad \text{and} \quad x^i = \frac{x^i}{\|x^i\|_2}, \ \ i = 1, 2, \cdots, m.$$

This scheme is called a multi-power method.

8.5. Singular Value Decomposition. Computing the singular value decomposition of a real symmetric matrix $A \in \mathbb{R}^{m\times n}$ is equivalent to finding the eigenvalue decomposition of $AA^\top$. Although the methods in the previous subsections can be applied to $AA^\top$ directly, the efficiency can be improved when some operations are performed carefully. We first state the abstract form of the LMSVD method [74], then describe a few implementation details.

There are two main steps. For a chosen subspace $\mathfrak{S}_i$ with a block Krylov subspace structure, an intermediate iterate is computed from

$$\hat{X}_i := \underset{X\in\mathbb{R}^{m\times p}}{\arg\max} \ \|A^\top X\|_{\mathrm{F}}^2, \ \text{s.t.}\ X^\top X = I, \ X \in \mathfrak{S}_i. \tag{8.12}$$

The next iterate X_{i+1} is generated from a SSI step on $\hat{X}_i$, i.e.,

$$X_{i+1} \in \mathbf{orth}\left(AA^\top \hat{X}_i \right). \tag{8.13}$$

We collect a limited memory of the last a few iterates in (8.10) into a matrix

$$\mathbf{X} = \mathbf{X}_i^t := [X_i, X_{i-1}, ..., X_{i-t}] \in \mathbb{R}^{m\times q} \tag{8.14}$$

where $q = (t+1)p$ is the total number of columns in $\mathbf{X}_i^t$. For simplicity of notation, the superscript and subscript of $\mathbf{X}_i^t$ are dropped whenever no confusion would arise. The collection matrix $\mathbf{X}$ is written in boldfaces to differentiate it from its blocks. Similarly, a collection of matrix-vector multiplications from the SSI steps are saved in

$$\mathbf{Y} = \mathbf{Y}_i^t := A^\top \mathbf{X}_i^t := \left[A^\top X_i, A^\top X_{i-1}, ..., A^\top X_{i-t}\right] \in \mathbb{R}^{m\times q}. \tag{8.15}$$

Assume that $\mathbf{X}$ has a full rank and this assumption will be relaxed later. A stable approach for solving (8.12) is to find an orthonormal basis for $\mathfrak{S}_i$, say,

$$\mathbf{Q} = \mathbf{Q}_i^t \in \mathbf{orth}\left(\mathbf{X}_i^t\right).$$

Note that $X \in \mathfrak{S}_i$ if and only if $X = \mathbf{Q}V$ for some $V \in \mathbb{R}^{q\times p}$. The generalized eigenvalue problem (8.12) is converted into an equivalent eigenvalue problem

$$\max_{V\in\mathbb{R}^{q\times p}} \|\mathbf{R}V\|_{\mathrm{F}}^2, \text{ s.t. } V^\top V = I, \tag{8.16}$$

where

$$\mathbf{R} = \mathbf{R}_i^t := A^\top \mathbf{Q}_i^t. \tag{8.17}$$

The matrix product $\mathbf{R}$ in (8.17) can be computed from historical information without any additional computation involving the matrix A. Since $\mathbf{Q} \in \mathbf{orth}(\mathbf{X})$ and $\mathbf{X}$ has a full rank, there exists a nonsingular matrix $C \in \mathbb{R}^{q\times q}$ such that $\mathbf{X} = \mathbf{Q}C$. Therefore, $\mathbf{Q} = \mathbf{X}C^{-1}$, and $\mathbf{R}$ in (8.17) can be assembled as

$$\mathbf{R} = A^\top \mathbf{Q} = (A^\top \mathbf{X})C^{-1} = \mathbf{Y}C^{-1}, \tag{8.18}$$

where $\mathbf{Y} = A^\top \mathbf{X}$ is accessible from our limited memory. Once $\mathbf{R}$ is available, a solution $\hat{V}$ to (8.16) can be computed from the p leading eigenvectors of the $q \times q$ matrix $\mathbf{R}^\top\mathbf{R}$. The matrix product can then be calculated as

$$AA^\top \hat{X}_i = A\mathbf{R}\hat{V} = A\mathbf{Y}C^{-1}\hat{V}. \tag{8.19}$$

We now explain how to efficiently and stably compute $\mathbf{Q}$ and $\mathbf{R}$ when the matrix $\mathbf{X}$ is numerically rank deficient. Since each block itself in $\mathbf{X}$ is orthonormal, keeping the latest block X_i intact and projecting the rest of the blocks onto the null space of $X_i^\top$ yields

$$\mathbf{P}_X = \mathbf{P}_i^X := \left(I - X_i X_i^\top\right)\left[X_{i-1} \ \cdots \ X_{i-p}\right]. \tag{8.20}$$

An orthonormalization of $\mathbf{P}_X$ is performed via the eigenvalue decomposition of its Gram matrix

$$\mathbf{P}_X^\top \mathbf{P}_X = U_X \Lambda_X U_X^\top, \tag{8.21}$$

where U_X is orthogonal and Λ_X is diagonal. If Λ_X is invertible, it holds

$$\mathbf{Q} = \mathbf{Q}_i^t := \left[X_i, \ \mathbf{P}_X U_X \Lambda_X^{-\frac{1}{2}}\right] \in \mathbf{orth}\left(\mathbf{X}_i^t\right). \tag{8.22}$$

The above procedure can be stabilized by deleting the columns of $\mathbf{P}_X$ whose Euclidean norms are below a threshold or deleting the small eigenvalues in Λ_X and the corresponding columns in U_X. The same notations are still used for $\mathbf{P}_X$, U_X and Λ_X after these possible deletions. Therefore, a stable construction of $\mathbf{Q}$ is still provided by formula (8.22) and the corresponding $\mathbf{R}$ matrix can be formulated as

$$\mathbf{R} = \mathbf{R}_i^t := \left[Y_i, \ \mathbf{P}_Y U_X \Lambda_X^{-\frac{1}{2}}\right], \tag{8.23}$$

where $\mathbf{P}_Y = \mathbf{P}_i^Y := A^\top \mathbf{P}_X$ before the stabilization procedure but some of the columns of $\mathbf{P}_Y$ may have been deleted due to the stabilization steps. Therefore, the $\mathbf{R}$ matrix in (8.23) is well defined as is the $\mathbf{Q}$ matrix in (8.22) after the numerical rank deficiency is removed.

8.6. Randomized SVD. Given an $m \times n$ matrix A and an integer $p < \min(m, n)$, we want to find an orthonormal $m \times p$ matrix Q such that

$$A \approx QQ^T A.$$

A prototype randomized SVD in [54] is essentially one step of the Power method using an initial random input. We select an oversampling parameter $l \geq 2$ and an exponent t (for example, $t = 1$ or $t = 2$), then perform the following steps.

- Generate an $n \times (p + l)$ Gaussian matrix Ω.
- Compute $Y = (AA^\top)^t A\Omega$ by the multiplications of A and $A^\top$ alternatively.
- Construct a matrix $Q = \mathrm{orth}(Y)$ by the QR factorization.
- Form the matrix $B = Q^\top A$.
- Calculate an SVD of B to obtain $B = \tilde{U}\Sigma V^\top$, and set $U = Q\tilde{U}$.

Consequently, we have the approximation $A \approx U\Sigma V^\top$. For the eigenvalue computation, we can simply run the SSI (8.2) for only one step with an Gaussian matrix U. Assume that the computation is performed in exact arithmetic. It is shown in [54] that

$$\mathbb{E}\|A - QQ^\top A\|_2 \leq \left[1 + \frac{4\sqrt{p+l}}{l-1}\right] \sigma_{p+1},$$

where the expectation is taken with respect to the random matrix Ω and σ_{p+1} is the $(p+1)$-th largest singular value of A.

Suppose that a low rank approximation of A with a target rank r is needed. A sketching method is further developed in [118] for selected p and ℓ. Again, we draw independent Gaussian matrix $\Omega \in \mathbb{R}^{n\times p}$ and $\Psi \in \mathbb{R}^{\ell\times m}$, and compute the matrix-matrix multiplications:

$$Y = A\Omega, \qquad W = \Psi A,$$

Then an approximation $\hat{A}$ is computed:

- Calculate an orthogonal-triangular factorization $Y = QR$ where $Q \in \mathbb{R}^{m\times p}$.
- Compute a least-squares problem to derive $X = (\Psi Q)^\dagger W \in \mathbb{R}^{p\times n}$
- Assemble the rank-p approximation $\hat{A} = QX$

Assume that $p = 2r + 1$ and $\ell = 4r + 2$. It is established that

$$\mathbb{E}\|A - \hat{A}\|_F \leq 2 \min_{\mathrm{rank}(Z)\leq r} \|A - Z\|_F.$$

8.7. Truncated Subspace Method for Tensor Train. In this subsection, we consider the trace maximization problem (8.7) whose dimension reaches the magnitude of $O(10^{42})$. Due to the scale of data storage, a tensor train (TT) format is used to express data matrices and eigenvectors in [148]. The corresponding eigenvalue problem can be solved based on the subspace algorithm and the alternating direction method with suitable truncations.

The goal is to express a vector $x \in \mathbb{R}^n$ as a tensor $\mathbf{x} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_d}$ for some positive integers $n_1, \ldots, n_d$ such that $n = n_1 n_2 \ldots n_d$ using a collection of three-dimensional tensor cores $\mathbf{X}_\mu \in \mathbb{R}^{r_{\mu-1}\times r_\mu\times n_u}$ with fixed dimensions r_μ, $\mu = 1, \ldots, d$ and $r_0 = r_d = 1$. A tensor $\mathbf{x}$ is stored in the TT format if its elements can be written as

$$x_{i_1 i_2 \ldots i_d} = X_1(i_1)X_2(i_2)\cdots X_d(i_d),$$

where $X_\mu(i_\mu) \in \mathbb{R}^{r_{\mu-1}\times r_\mu}$ is the i_μ-th slice of $\mathbf{X}_\mu$ for $i_\mu = 1, 2, \ldots, n_\mu$. The values r_μ are often equal to a constant r, which is then called the TT-rank. Consequently, storing a vector $x \in \mathbb{R}^{n_1^d}$ only needs $\mathcal{O}(dn_1r^2)$ entries if the corresponding tensor $\mathbf{x}$ has a TT format. The representation of $\mathbf{x}$ is shown as graphs in Figure 8.1.

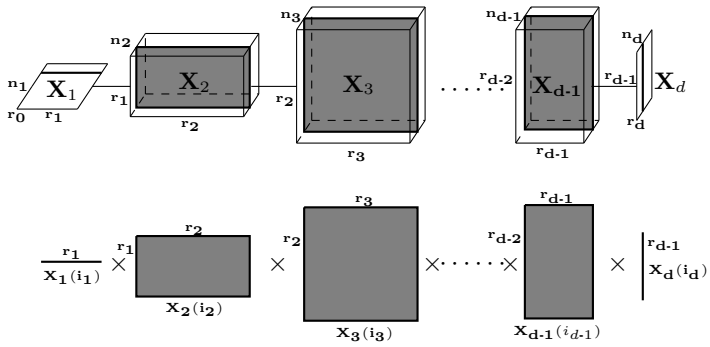

Fig. 8.1 *The first row is a TT format of* $\mathbf{u}$ *with cores* $\mathbf{X}_\mu,\ \mu = 1, 2, \ldots, d$. *The second row is a representation of its elements* $x_{i_1 i_2 \ldots i_d}$.

There are several ways to express a matrix $X \in \mathbb{R}^{n \times p}$ with $p \ll n$ in the TT format. A direct way is to store each column of X as tensors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p$ in the TT format separately. Another economic choice is that these p tensors share all except one core. Let the shared cores be $\mathbf{X}_i$, $i \neq \mu$ and the μ-th core of $\mathbf{x}_i$ be $\mathbf{X}_{\mu,i}$, for $i = 1, 2, \ldots, p$. Then the $i_1 i_2 \cdots i_d$ component of $\mathbf{x}_j$ is

$$X(i_1, \ldots, i_\mu, \ldots, i_d; j) = X_1(i_1) \cdots X_{\mu,j}(i_\mu) \cdots X_d(i_d). \tag{8.24}$$

The above scheme generates a block-μ TT (μ-BTT) format, which is depicted in Figure 8.2. Similarly, a matrix $A \in \mathbb{R}^{n \times n}$ is in an operator TT format $\mathbf{A}$ if the components of A can be assembled as

$$A_{i_1 i_2 \cdots i_d, j_1 j_2 \cdots j_d} = A_1(i_1, j_1) A_2(i_2, j_2) \cdots A_d(i_d, j_d), \tag{8.25}$$

where $A_\mu(i_\mu, j_\mu) \in \mathbb{R}^{r_{\mu-1} \times r_\mu}$ for $i_\mu, j_\mu \in \{1, \ldots, n_\mu\}$.

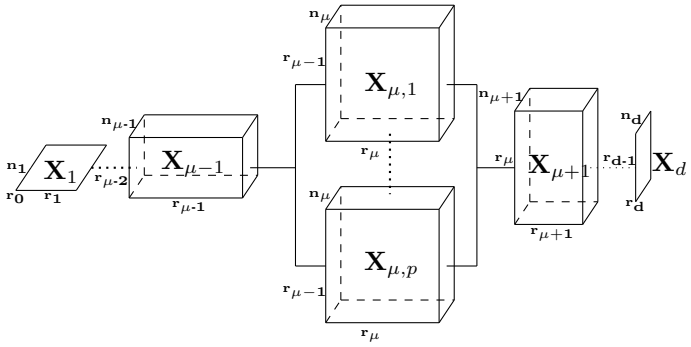

Fig. 8.2 *Demonstration of a* μ*-BTT format.*

Assume that the matrix A itself can be written in the operator TT format. Let $X \in \mathbb{R}^{n \times p}$ with $n = n_1 n_2 \ldots n_d$ whose BTT format is $\mathbf{X}$, and $\mathbf{T}_{\mathbf{n},r,p}$ be the set of the BTT formats whose TT-ranks are no more than r. Then the eigenvalue problem in the block BTT format is

$$\min_{X \in \mathbb{R}^{n \times p}} \operatorname{tr}(\mathbf{X}^\top \mathbf{A} \mathbf{X}), \quad \text{s.t.} \quad \mathbf{X}^\top \mathbf{X} = I_p \text{ and } \mathbf{X} \in \mathbf{T}_{\mathbf{n},r,p}, \tag{8.26}$$

where $\mathbf{X} \in \mathbf{T}_{\mathbf{n},r,p}$ means that all calculations are performed in the BTT format. Since the TT-ranks increase dramatically after operations such as the addition and matrix-vector multiplication in the TT formats, the computational cost and the storage becomes more and more expensive as the TT-ranks increase. Therefore, the subspace methods in subsection 8.3 can only be applied with projections to $\mathbf{T}_{\mathbf{n},r,p}$ at some suitable places so that the overall computational cost is still tractable.

In our truncated subspace optimization methods, solving the subproblem (8.8) is split into a few steps. First, the subspace $\mathfrak{S}_k$ is modified with truncations so that the computation of the coefficient matrix $U^\top AU$ in the RR procedure is affordable. Let $\mathcal{P}_{\mathbf{T}}(\mathbf{X})$ be the truncation of $\mathbf{X}$ to the BTT format $\mathbf{T}_{\mathbf{n},r,p}$. One can choose either the following subspace

$$\mathfrak{S}_k^{\mathbf{T}} = \mathrm{span}\{\mathcal{P}_{\mathbf{T}}(\mathbf{A}\mathbf{X}_k), \mathbf{X}_k, \mathbf{X}_{k-1}\}, \tag{8.27}$$

or a subspace similar to that of LOBPCG with two truncations as

$$\mathfrak{S}_k^{\mathbf{T}} = \mathrm{span}\{\mathbf{X}_k, \mathcal{P}_{\mathbf{T}}(\mathbf{R}_k), \mathcal{P}_{\mathbf{T}}(\mathbf{P}_k)\}, \tag{8.28}$$

where the conjugate gradient direction is $\mathbf{P}_k = X_k - X_{k-1}$ and the residual vector is $\mathbf{R}_k = AX_k - X_k\Lambda_k$.

Consequently, the subspace problem in the BTT format is

$$\mathbf{Y}_{k+1} := \underset{X\in\mathbb{R}^{n\times p}}{\arg\min}\ \mathrm{tr}(\mathbf{X}^\top \mathbf{A}\mathbf{X}),\ \mathrm{s.t.}\ \ \mathbf{X}^\top\mathbf{X} = I_p,\ \mathbf{X} \in \mathfrak{S}_k^{\mathbf{T}}, \tag{8.29}$$

which is equivalent to a generalized eigenvalue decomposition problem:

$$\min_{V\in\mathbb{R}^{q\times p}}\ \mathrm{tr}(V^\top (S^\top AS)V),\ \mathrm{s.t.}\ \ V^\top S^\top SV = I_p. \tag{8.30}$$

Note that $\mathbf{Y}_{k+1} \notin \mathbf{T}_{\mathbf{n},r,p}$ because the rank of $\mathbf{Y}_{k+1}$ is larger than r due to several additions between the BTT formats. Since $\mathbf{Y}_{k+1}$ is a linear combination of the BTT formats in $\mathfrak{S}_k^{\mathbf{T}}$, problem (8.29) still can be solved easily but only the coefficients of the linear combinations are stored.

We next project $\mathbf{Y}_{k+1}$ to the required space $\mathbf{T}_{\mathbf{n},r,p}$ as

$$\mathbf{X}_{k+1} = \arg\min_{\mathbf{X}\in\mathbb{R}^{n\times p}} \|\mathbf{X} - \mathbf{Y}_{k+1}\|_F^2, \quad \mathrm{s.t.} \quad \mathbf{X}^\top\mathbf{X} = I_p, \quad \mathbf{X} \in \mathbf{T}_{\mathbf{n},r,p}. \tag{8.31}$$

This problem can be solved by using the alternating minimization scheme. By fixing all except the μth core, we obtain

$$\min_V \quad \|\mathcal{X}_{\neq\mu}V - \mathrm{vec}(\mathbf{Y}_{k+1})\|_F^2,\ \mathrm{s.t.}\ \ V^\top\mathcal{X}_{\neq\mu}^\top\mathcal{X}_{\neq\mu}V = I_p, \tag{8.32}$$

where

$$\mathcal{X}_{\neq\mu} := (\mathbf{X}_{\geq\mu+1} \otimes I_{n_\mu} \otimes \mathbf{X}_{\leq\mu-1}),$$

and

$$X_{\leq\mu} = [X_1(i_1)X_2(i_2)\cdots X_\mu(i_\mu)] \in \mathbb{R}^{n_1n_2\cdots n_\mu\times r_\mu},$$
$$X_{\geq\mu} = [X_\mu(i_\mu)X_{\mu+1}(i_{\mu+1})\cdots X_d(i_d)]^\top \in \mathbb{R}^{n_\mu n_{\mu+1}\cdots n_d\times r_{\mu-1}}.$$

Therefore, after imposing orthogonality on $\mathcal{X}_{\neq\mu}$, (8.32) is reformulated as

$$\min_V \quad \|V - \mathcal{X}_{\neq\mu}^\top\mathrm{vec}(\mathbf{Y}_{k+1})\|_F^2,\ \mathrm{s.t.}\ \ V^\top V = I_p, \tag{8.33}$$

whose optimal solution can be computed by the p-dominant SVD of $\mathcal{X}_{\neq\mu}^\top\mathrm{vec}(\mathbf{Y}_{k+1})$.

9. Optimization with Orthogonality Constraints. In this section, we consider the optimization problem with orthogonality constraints [132, 59, 2]:

$$\min_{X \in \mathbb{C}^{n\times p}} \quad f(X) \quad \text{s.t.} \quad X^*X = I_p, \tag{9.1}$$

where $f(X) : \mathbb{C}^{n\times p} \to \mathbb{R}$ is a $\mathbb{R}$-differentiable function [67]. The set $\mathrm{St}(n,p) := \{X \in \mathbb{C}^{n\times p} : X^*X = I_p\}$ is called the Stiefel manifold. Obviously, the eigenvalue problem in section 8 is a special case of (9.1). Other important applications include the density functional theory [131], Bose-Einstein condensates [137], low rank nearest correlation matrix completion [121], and etc. Although (9.1) can be treated from the perspective of general nonlinear programming, the intrinsic structure of the Stiefel manifold enables us to develop more efficient algorithms. In fact, it can be solved by the Riemannian gradient descent, Riemannian conjugate gradient, proximal Riemannian gradient methods [40, 104, 2, 59]. The Riemannian Newton, trust-region, adaptive regularized Newton methods [120, 1, 2, 59] can used when the Hessian information is available. Otherwise, the quasi-Newton types methods are good alternatives [62, 61, 58].

The tangent space is $T_X := \{\xi \in \mathbb{C}^{n\times p} : X^*\xi + \xi^*X = 0\}$. The operator $\mathrm{Proj}_X(Z) := Z - X\mathrm{sym}(X^*Z)$ is the projection of Z onto the tangent space T_X and $\mathrm{sym}(A) := (A + A^*)/2$. The symbols $\nabla f(X)$ ($\nabla^2 f(X)$) and $\mathrm{grad}\, f(X)$ ($\mathrm{Hess}\, f(X)$) denote the Euclidean and Riemannian gradient (Hessian) of f at X. Using the real part of the Frobenius inner product $\Re\langle A, B\rangle$ as the Euclidean metric, the Riemannian Hessian $\mathrm{Hess}\, f(X)$ [31, 3] can be written as

$$\mathrm{Hess}\, f(X)[\xi] = \mathrm{Proj}_X(\nabla^2 f(X)[\xi] - \xi\mathrm{sym}(X^*\nabla f(X))), \tag{9.2}$$

where ξ is any tangent vector in T_X. A retraction R is a smooth mapping from the tangent bundle to the manifold. Moreover, the restriction R_X of R to T_X has to satisfy $R_X(0_X) = X$ and $\mathrm{D}R_X(0_X) = \mathrm{id}_{T_X}$, where id_{T_X} is the identity mapping on T_X.

9.1. Regularized Newton Type Approaches. We now describe an adaptively regularized Riemannian Newton type method with a subspace refinement procedure [59, 58]. Note that the Riemannian Hessian-vector multiplication (9.2) involves the Euclidean Hessian and gradient with simple structures. We construct a second-order Taylor approximation in the Euclidean space rather than the Riemannian space at the k-th iteration:

$$m_k(X) := \Re\langle \nabla f(X_k), X - X_k\rangle + \frac{1}{2}\Re\langle \mathcal{B}_k[X - X_k], X - X_k\rangle + \frac{\tau_k}{2}\|X - X_k\|_F^2, \tag{9.3}$$

where $\mathcal{B}_k$ is either $\nabla^2 f(X_k)$ or its approximation based on whether $\nabla^2 f(X_k)$ is affordable or not, and τ_k is a regularization parameter to control the distance between X and X_k. Then the subproblem is

$$\min_{X \in \mathbb{C}^{n\times p}} \quad m_k(X) \quad \text{s.t.} \quad X^*X = I. \tag{9.4}$$

After obtaining an approximate solution Z_k of (9.4), we calculate a ratio between the predicted reduction and the actual reduction, then use the ratio to decide whether X_{k+1} is set to Z_k or X_k and to adjust the parameter τ_k similar to the trust region methods.

In particular, the model (9.4) can be minimized by using a modified CG method to solve a single Riemannian Newton system:

$$\mathrm{grad}\, m_k(X_k) + \mathrm{Hess}\, m_k(X_k)[\xi] = 0. \tag{9.5}$$

A simple calculation yields:

$$\text{Hess}\, m_k(X_k)[\xi] = \text{Proj}_{X_k}\left(\mathcal{B}_k[\xi] - \xi \text{sym}((X_k)^* \nabla f(X_k)) + \tau_k \xi,\ \xi \in T_{X_k}.\right. \tag{9.6}$$

Hence, the regularization term shifts the spectrum of the Riemannian Hessian by τ_k. The modified CG method is a direct adaption of the truncated CG method for solving the classic trust region subproblem, see [88, Chapter 5] and [2, Chapter 7] for a comparison. It is terminated when either the residual becomes small or a negative curvature is detected since the Hessian may be indefinite. During the process, two different vectors s_k and d_k are generated, where the vector d_k represents the negative curvature direction and s_k corresponds to the conjugate direction from the CG iteration. The direction d_k is zero unless a negative curvature is detected. Therefore, a possible choice of the search direction ξ_k is

$$\xi_k = \begin{cases} s_k + \tau_k d_k & \text{if } d_k \neq 0, \\ s_k & \text{if } d_k = 0, \end{cases} \quad \text{with} \quad \tau_k := \frac{\langle d_k, \text{grad}\, m_k(X_k)\rangle}{\langle d_k, \text{Hess}\, m_k(X_k)[d_k]\rangle}. \tag{9.7}$$

Once the direction ξ_k is computed, a trial point Z_k is searched along ξ_k followed by a retraction, i.e.,

$$Z_k = R_{X_k}(\alpha_k \xi_k). \tag{9.8}$$

The step size $\alpha_k = \alpha_0 \delta^h$ is chosen by the Armijo rule such that h is the smallest integer satisfying

$$m_k(R_{X_k}(\alpha_0 \delta^h \xi_k)) \leq \rho \alpha_0 \delta^h \left\langle \text{grad}\, m_k(X_k), \xi_k \right\rangle, \tag{9.9}$$

where $\rho, \delta \in (0,1)$ and $\alpha_0 \in (0,1]$ are given constants.

The performance of the Newton-type method may be seriously deteriorated when the Hessian is close to be singular. One reason is that the Riemannian Newton direction is nearly parallel to the negative gradient direction. Consequently, the next iteration X_{k+1} very likely belongs to the subspace $\text{span}\{X_k, \text{grad}\, f(X_k)\}$, which is similar to the Riemannian gradient approach. To overcome the numerical difficulty, we can further solve (9.1) in a restricted subspace. Specifically, a q-dimensional subspace $\mathfrak{S}_k$ is constructed with an orthogonal basis $Q_k \in \mathbb{C}^{n\times q} (p \leq q \leq n)$. Then the representation of any point X in the subspace $\mathfrak{S}_k$ is

$$X = Q_k M$$

for some $M \in \mathbb{C}^{q\times p}$. In a similar fashion to these constructions for the linear eigenvalue problems in section 8, the subspace $\mathfrak{S}_k$ can be built by using the history information $\{X_k, X_{k-1}, \ldots\}$, $\{\text{grad}\ f(X_k), \text{grad}\ f(X_{k-1}), \ldots\}$ and other useful information. Once a subspace $\mathfrak{S}_k$ is given, (9.1) with an additional constraint $X \in \mathfrak{S}_k$ becomes

$$\min_{M\in\mathbb{C}^{q\times p}} \quad f(Q_k M) \quad \text{s. t.} \quad M^* M = I_p. \tag{9.10}$$

Suppose that M_k is an inexact solution of the problem (9.10) from existing optimization methods on manifold. Then $X_{k+1} = Q_k M_k$ is a better point than X_k. For extremely difficult problems, one may alternate between the Newton type method and the subspace refinement procedure for a few cycles.

9.2. A Structured Quasi-Newton Update with Nyström Approximation. The secant condition in the classical quasi-Newton methods for constructing the quasi-Newton matrix $\mathcal{B}_k$

$$\mathcal{B}_k[S_k] = \nabla f(X_k) - \nabla f(X_{k-1}), \tag{9.11}$$

where

$$S_k := X_k - X_{k-1}.$$

Assume that the Euclidean Hessian $\nabla^2 f(X)$ is a summation of a relatively cheap part $\mathcal{H}^{\mathrm{c}}(X)$ and a relatively expensive or even inaccessible part $\mathcal{H}^{\mathrm{e}}(X)$, i.e.,

$$\nabla^2 f(X) = \mathcal{H}^{\mathrm{c}}(X) + \mathcal{H}^{\mathrm{e}}(X). \tag{9.12}$$

Then it is reasonable to keep the cheaper part $\mathcal{H}^{\mathrm{c}}(X)$ but approximate $\mathcal{H}^{\mathrm{e}}(X)$ using the quasi-Newton update $\mathcal{E}_k$. It yields an approximation $\mathcal{B}_k$ to the Hessian $\nabla^2 f(X_k)$ as

$$\mathcal{B}_k = \mathcal{H}^{\mathrm{c}}(X_k) + \mathcal{E}_k, \tag{9.13}$$

Plugging (9.13) into (9.11) gives the following revised secant condition

$$\mathcal{E}_k[S_k] = Y_k, \tag{9.14}$$

where

$$Y_k := \nabla f(X_k) - \nabla f(X_{k-1}) - \mathcal{H}^{\mathrm{c}}(X_k)[S_k]. \tag{9.15}$$

A good initial matrix $\mathcal{E}_k^0$ to $\mathcal{E}_k$ is important to ensure the convergence speed of the limited-memory quasi-Newton method. We assume that a known matrix $\hat{\mathcal{E}}_k^0$ can approximate the expensive part of the Hessian $\mathcal{H}^{\mathrm{e}}(X_k)$ well, a very limited number of matrix-matrix products involving $\hat{\mathcal{E}}_k^0$ is affordable but many of them are still prohibitive. We next use the Nyström approximation [117] to construct a low rank matrix. Let Ω be a matrix whose columns constitute an orthogonal basis of a well-chosen subspace $\mathfrak{S}$ and denote $W = \hat{\mathcal{E}}_k^0[\Omega]$. The Nyström approximation is

$$\mathcal{E}_k^0[U] \coloneqq W(W^*\Omega)^\dagger W^* U, \tag{9.16}$$

where $U \in \mathbb{C}^{n\times p}$ is any direction. When the dimension of the subspace $\mathfrak{S}$ is small enough, the rank of $W(W^*\Omega)^\dagger W^*$ is also small so that the computational cost of $\mathcal{E}_k^0[U]$ is significantly cheaper than the original $\hat{\mathcal{E}}_k^0[U]$. Suppose the subspace $\mathfrak{S}$ is chosen as

$$\mathrm{span}\{X_{k-1}, X_k\},$$

which contains the element S_k. If $\hat{\mathcal{E}}_k^0[UV] = \hat{\mathcal{E}}_k^0[U]V$ for any matrices U, V with proper dimension (this condition is satisfied when $\hat{\mathcal{E}}_k^0$ is a matrix), then the secant condition still holds at $\mathcal{E}_k^0$, i.e.,

$$\mathcal{E}_k^0[S_k] = Y_k.$$

The subspace $\mathfrak{S}$ can also be defined as

$$\mathrm{span}\{X_{k-1}, X_k, \hat{\mathcal{E}}_k^0[X_k]\} \quad \text{or} \quad \mathrm{span}\{X_{k-h}, \ldots, X_{k-1}, X_k\} \tag{9.17}$$

with small memory length h. Consequently, we obtain a limited-memory Nyström approximation.

9.3. Electronic Structure Calculations. The density functional theory (DFT) in electronic structure calculation is an important source of optimization problems with orthogonality constraints. By abuse of notation, we refer to Kohn-Sham (KS) equations with local or semi-local exchange-correlation functionals as KSDFT, and KS equations with hybrid functionals as HF (short for Hartree-Fock). The KS/HF equations try to identify orthogonal eigenvectors to satisfy the nonlinear eigenvalue problems, while the KS/HF minimization problem minimizes the KS/HF total energy functionals under the orthogonality constraints. These two problems are connected by the optimality conditions.

9.3.1. The Mathematical Models. The wave functions of p occupied states can be expressed as $X = [x_1, \ldots, x_p] \in \mathbb{C}^{n\times p}$ with $X^*X = I_p$ after some suitable discretization. The KS total energy functional is defined as

(9.18)
$$E_{\rm ks}(X) := \frac{1}{4}{\rm tr}(X^*LX) + \frac{1}{2}{\rm tr}(X^*V_{\rm ion}X) + \frac{1}{2}\sum_l\sum_i \zeta_l|x_i^*w_l|^2 + \frac{1}{4}\rho^\top L^\dagger\rho + \frac{1}{2}e_n^\top\epsilon_{\rm xc}(\rho),$$

where L is a discretized Laplacian operator, the charge density is $\rho(X) = {\rm diag}(XX^*)$, $V_{\rm ion}$ is the constant ionic pseudopotentials, w_l represents a discretized pseudopotential reference projection function, ζ_l is a constant whose value is ± 1, and $\epsilon_{\rm xc}$ is related to the exchange correlation energy. The Fock exchange operator $\mathcal{V}(\cdot) : \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ is usually a fourth-order tensor [69] which satisfies the following properties: (i) $\langle\mathcal{V}(D_1), D_2\rangle = \langle\mathcal{V}(D_2), D_1\rangle$ for any $D_1, D_2 \in \mathbb{C}^{n\times n}$; (ii) $\mathcal{V}(D)$ is Hermitian if D is Hermitian. Then the Fock exchange energy is

$$E_{\rm f}(X) := \frac{1}{4}\left\langle\mathcal{V}(XX^*)X, X\right\rangle = \frac{1}{4}\left\langle\mathcal{V}(XX^*), XX^*\right\rangle. \tag{9.19}$$

Therefore, the total energy minimization problem can be formulated as

$$\min_{X\in\mathbb{C}^{n\times p}} E(X), \quad \text{s.t.} \quad X^*X = I_p, \tag{9.20}$$

where $E(X)$ is $E_{\rm ks}(X)$ in KSDFT and

$$E_{\rm hf}(X) := E_{\rm ks}(X) + E_{\rm f}(X)$$

in HF. Computing $E_{\rm f}(X)$ is very expensive since a multiplication between an $n\times n\times n\times n$ fourth-order tensor and an n-by-n matrix is needed in $\mathcal{V}(\cdot)$.

Denote the KS Hamiltonian $H_{\rm ks}(X)$ as

$$H_{\rm ks}(X) := \frac{1}{2}L + V_{\rm ion} + \sum_l \zeta_l w_l w_l^* + {\rm Diag}((\Re L^\dagger)\rho) + {\rm Diag}(\mu_{\rm xc}(\rho)^* e_n), \tag{9.21}$$

where $\mu_{\rm xc}(\rho) = \frac{\partial\epsilon_{\rm xc}(\rho)}{\partial\rho}$. Since $H_{\rm ks}(X)$ is essentially determined by the charge density $\rho(X)$, it is often written as $H_{\rm ks}(\rho)$. The HF Hamiltonian is

$$H_{\rm hf}(X) := H_{\rm ks}(X) + \mathcal{V}(XX^*). \tag{9.22}$$

A detailed calculation shows that the Euclidean gradient of $E_{\rm ks}(X)$ is

$$\nabla E_{\rm ks}(X) = H_{\rm ks}(X)X. \tag{9.23}$$

The gradient of $E_{\rm f}(X)$ is $\nabla E_{\rm f}(X) = \mathcal{V}(XX^*)X$. Assume that $\epsilon_{\rm xc}(\rho(X))$ is twice differentiable with respect to $\rho(X)$, the Hessian of $E_{\rm ks}(X)$ is

$$\nabla^2 E_{\rm ks}(X)[U] = H_{\rm ks}(X)U + \mathcal{R}(X)[U], \tag{9.24}$$

where $U \in \mathbb{C}^{n\times p}$ and $\mathcal{R}(X)[U] := {\rm Diag}\left(\left(\Re L^\dagger + \frac{\partial^2\epsilon_{\rm xc}}{\partial\rho^2}e_n\right)(\bar{X}\odot U + X\odot\bar{U})e_n\right)X$. The Hessian of $E_{\rm f}(X)$ is

$$\nabla^2 E_{\rm f}(X)[U] = \mathcal{V}(XX^*)U + \mathcal{V}(XU^* + UX^*)X. \tag{9.25}$$

9.3.2. The Self-Consistent Field (SCF) Iteration. The first-order optimality conditions for the total energy minimization problem are

$$H(X)X = X\Lambda, \quad X^*X = I_p, \tag{9.26}$$

where $X \in \mathbb{C}^{n\times p}$, Λ is a diagonal matrix and H represents $H_{\rm ks}$ in (9.21) or $H_{\rm hf}$ in (9.22). For KSDFT, one of the most popular methods is the SCF iteration. At the k-th iteration, we first fix the Hamiltonian to be $H_{\rm ks}(\tilde{\rho}_k)$ for a given $\tilde{\rho}_k$ and solve the following linear eigenvalue problem

$$H_{\rm ks}(\tilde{\rho}_k)X = X\Lambda, \quad X^*X = I_p. \tag{9.27}$$

The eigenvectors corresponding to the p smallest eigenvalues of $H_{\rm ks}(\rho_k)$ is denoted as X_{k+1}, which leads to a new charge density $\rho_{k+1} = \rho(X_{k+1})$. It is then mixed with charge densities from previous steps to produce the new charge density $\tilde{\rho}_{k+1}$ in order to accelerate the convergence instead of using ρ_{k+1} directly. This procedure is repeated until self-consistency is reached.

A particular charge mixing scheme is the direct inversion of iterative subspace (DIIS) or the Pulay mixing [92, 93, 115]. Choose an integer m with $m \le k$. Let

$$W = (\Delta\rho_k, \Delta\rho_{k-1}, \ldots, \Delta\rho_{k-m+1}), \quad \Delta\rho_j = \rho_j - \rho_{j-1}.$$

The Pulay mixing generates the charge density $\tilde{\rho}_k$ by a linear combination of the previously charge densities

$$\tilde{\rho}_k = \sum_{j=0}^{m-1} c_j \rho_{k-j},$$

where $c = (c_0, c_1, \ldots, c_{m-1})$ is the solution to the minimization problem:

$$\min_c \quad \|Wc\|^2, \ \text{s.t.} \ \ c^\top e_m = 1.$$

Other types of mixing includes Broyden mixing, Kerker mixing and Anderson mixing, etc. Charge mixing is widely used for improving the convergence of SCF even though its convergence property is still not clear in few cases.

In HF, the SCF method at the k-th iteration solves:

$$\tilde{H}_k X = X\Lambda, \quad X^*X = I_p,$$

where $\tilde{H}_k$ is formed from certain mixing schemes. Note that the Hamiltonian (9.22) can be written as $H_{\rm hf}(D)$ with respect to the density matrix $D = XX^*$. In the commutator DIIS (C-DIIS) method [92, 93], the residual W_j is defined as the commutator between $H_{\rm hf}(D_j)$ and D_j, i.e.,

$$W_j = H_{\rm hf}(D_j)D_j - D_j H_{\rm hf}(D_j). \tag{9.28}$$

We next solve the following minimization to obtain a coefficient c:

$$\min_c \quad \left\| \sum_{j=0}^{m-1} c_j W_j \right\|_F^2, \ \text{s.t.} \ \ c^\top e_m = 1.$$

Then, a new Hamiltonian matrix is obtained $\tilde{H}_k = \sum_{j=0}^{m-1} c_j H_{k-j}$. Since an explicit storage of the density matrix can be prohibitive, the projected C-DIIS in [60] uses projections of the density and commutator matrices so that the sizes are much smaller.

9.3.3. Subspace Methods For HF using Nyström Approximation. Note that the most expensive part in HF is the evaluation of $E_{\rm f}(X)$ and the related derivatives. We apply the limited-memory Nyström technique to approximate $\mathcal{V}(X_kX_k^*)$ by $\hat{\mathcal{V}}(X_kX_k^*)$. Let $Z = \mathcal{V}(X_kX_k^*)\,\Omega$ where Ω is an orthogonal basis of the subspace such as

$$\mathrm{span}\{X_k\},\ \mathrm{span}\{X_{k-1}, X_k\} \text{ or } \mathrm{span}\{X_{k-1}, X_k, \mathcal{V}(X_kX_k^*)X_k\}.$$

Then the low rank approximation

$$\hat{\mathcal{V}}(X_kX_k^*) := Z(Z^*\Omega)^\dagger Z^* \tag{9.29}$$

is able to reduce the computational cost significantly. Note that the adaptive compression method in [73] compresses the operator $\mathcal{V}(X_kX_k^*)$ on the subspace $\mathrm{span}\{X_k\}$. Consequently, we can keep the easier parts $E_{\rm ks}$ but approximate $E_{\rm f}(X)$ by using (9.29). Hence, a new subproblem is formulated as

$$\min_{X\in\mathbb{C}^{n\times p}} \quad E_{\rm ks}(X) + \frac{1}{4}\left\langle \hat{\mathcal{V}}(X_kX_k^*)\,X, X\right\rangle \quad \text{s.t.} \quad X^*X = I_p. \tag{9.30}$$

The subproblem (9.30) can be solved by the SCF iteration, the Riemannian gradient method or the modified CG method based on the following linear equation

$$\mathrm{Proj}_{X_k}\left(\nabla^2 E_{\rm ks}(X_k)[\xi] + \frac{1}{2}\hat{\mathcal{V}}(X_kX_k^*)\xi - \xi\,\mathrm{sym}(X_k^*\nabla f(X_k))\right) = -\mathrm{grad}\, E_{\rm hf}(X_k).$$

9.3.4. A Regularized Newton Type Method. Computing the p-smallest eigenpairs of $H_{\rm ks}(\tilde{\rho})$ is equivalent to a trace minimization problem

$$\min_{X\in\mathbb{C}^{n\times p}} \quad q(X) := \frac{1}{2}\mathrm{tr}(X^*H_{\rm ks}(\tilde{\rho})X) \quad \text{s.t.} \quad X^*X = I_p. \tag{9.31}$$

Note that $q(X)$ is a second-order approximation to the total energy $E_{\rm ks}(X)$ without considering the second term in the Hessian (9.24). Hence, the SCF method may not converge if this second term dominates. The regularized Newton in (9.1) can be applied to solve both KSDFT and HF with convergence guarantees. We next explain a particular version in [138] whose subproblem is

$$\min_{X\in\mathbb{C}^{n\times p}} \quad \frac{1}{2}\mathrm{tr}(X^*H_{\rm ks}(\tilde{\rho})X) + \frac{\tau_k}{4}\|XX^\top - X_kX_k^\top\|_F^2 \quad \text{s.t.} \quad X^*X = I_p. \tag{9.32}$$

Since X_k and X are orthonormal matrices, we have

$$\begin{aligned}\|XX^\top - X_kX_k^\top\|_F^2 &= \mathrm{tr}((XX^\top - X_kX_k^\top)(XX^\top - X_kX_k^\top))\\ &= 2p - 2\mathrm{tr}(X^\top X_kX_k^\top X).\end{aligned}$$

Therefore, (9.32) is a linear eigenvalue problem:

$$\begin{aligned}(H_{\rm ks}(\tilde{\rho}) - \tau_k X_kX_k^\top)X &= X\Lambda,\\ X^\top X &= I_p.\end{aligned}$$

9.3.5. Subspace Refinement for KSDFT. The direct minimization method in [138] is a kind of subspace refinement procedure using

$$Y = [X_k, P_k, R_k],$$

where $P_k = X_k - X_{k-1}$ and $R_k = H_{\rm ks}(X_k)X_k - X_k\Lambda_k$. Then the variable X can be expressed as $X = YG$ where $G \in \mathbb{C}^{3p\times p}$. The total energy minimization problem (9.20) becomes:

$$\min_{G}\ E_{\rm ks}(YG),\ \text{s.t.}\ \ G^*Y^*YG = I_p,$$

whose first-order optimality condition is a generalized linear eigenvalue problem:

$$(Y^*H_{\rm ks}(YG)Y)G = Y^*YG\Omega, \quad G^*Y^*YG = I_p.$$

The subspace refinement method may help when the regularized Newton method does not perform well. Note that the total energy minimization problem (9.20) is not necessary equivalent to a nonlinear eigenvalue problem (9.26) for finding the p smallest eigenvalues of $H(X)$. Although an intermediate iterate X is orthogonal and contains eigenvectors of $H(X)$, these eigenvectors are not necessary the eigenvectors corresponding to the p smallest eigenvalues. Hence, we can form a subspace which contains these possible target eigenvectors. In particular, we first compute the first γp smallest eigenvalues for some small integer γ. Their corresponding eigenvectors of $H(X_k)$, denoted by Γ_k, are put in a subspace as

$$\text{span}\{X_{k-1}, X_k, \text{grad}\, E(X_k), \Gamma_k\}. \tag{9.33}$$

Numerical experience shows that the refinement scheme in subsection 9.1 with this subspace is likely escape a stagnated point.

10. Semidefinite Programming (SDP). In this section, we present two specialized subspace methods for solving the maxcut SDP and the maxcut SDP with nonnegative constraints from community detection.

10.1. The Maxcut SDP. The maxcut problem partition the vertices of a graph into two sets so that the sum of the weights of the edges connecting vertices in one set with these in the other set is maximized. The corresponding SDP relaxation [46, 16, 56, 8] is

$$\begin{aligned} \min_{X\in\mathcal{S}^n}\quad & \langle C, X\rangle \\ \text{s.t.}\quad & X^{ii} = 1, \quad i = 1, \cdots, n, \\ & X \succeq 0. \end{aligned} \tag{10.1}$$

We first describe a second-order cone program (SOCP) restriction for the SDP problem (10.1) by fixing all except one row and column of the matrix X. For any integer $i \in \{1, \ldots, n\}$, the complement of the set $\{i\}$ is $i^c = \{1, \ldots, n\}\backslash\{i\}$. Let $B = X^{i^c,i^c}$ be the submatrix of X after deleting its i-th row and column, and $y = X^{i^c,i}$ be the ith column of the matrix X without the element $X^{i,i}$. Since $X_{ii} = 1$, the variable X of (10.1) can be written as

$$X := \begin{pmatrix} 1 & y^\top \\ y & B \end{pmatrix} := \begin{pmatrix} 1 & y^\top \\ y & X^{i^c,i^c} \end{pmatrix}$$

without loss of generality. Suppose that the submatrix B is fixed. It then follows from the Schur complement theorem that $X \succeq 0$ is equivalent to

$$\xi - y^\top B^{-1} y \geq 0.$$

In order to maintain the strict positive definiteness of X, we require $1 - y^\top B^{-1} y \geq \nu$ for a small constant $\nu > 0$. Therefore, the SDP problem (10.1) is reduced to a SOCP:

$$\begin{aligned} \min_{y \in \mathbb{R}^{n-1}} \quad & \widehat{c}^\top y \\ \text{s.t.} \quad & 1 - y^\top B^\dagger y \geq \nu, \quad y \in \mathrm{Range}(B), \end{aligned} \tag{10.2}$$

where $\widehat{c} := 2C^{i^c,i}$. If $\gamma := \widehat{c}^\top B \widehat{c} > 0$, an explicit solution of (10.2) is given by

$$y = -\sqrt{\frac{1-\nu}{\gamma}} B\widehat{c}. \tag{10.3}$$

Otherwise, the solution is $y = 0$.

We next describe the RBR method [130]. Starting from a positive definite feasible solution X_1, it updates one row/column of X at each of the inner steps. The operations from the first row to the last row is called a cycle. At the first step of the k-th cycle, the matrix B is set to $X_k^{1^c,1^c}$ and y is computed by (10.3). Then the first row/column of X_k is substituted by $X_k^{1^c,1} := y$. Other rows/columns are updated in a similar fashion until all row/column are updated. Then we set $X_{k+1} := X_k$ and this procedure is repeated until certain stopping criteria are met.

The RBR method can also be derived from the logarithmic barrier problem

$$\begin{aligned} \min_{X \in \mathcal{S}^n} \quad & \phi_\sigma(X) := \langle C, X \rangle - \sigma \log\det X \\ \text{s.t.} \quad & X^{ii} = 1, \forall i = 1, \cdots, n, \quad X \succ 0. \end{aligned} \tag{10.4}$$

Fixing the block $B = X^{i^c,i^c}$ gives

$$\det(X) = \det(B)(1 - (X^{i^c,i})^\top B^{-1} X^{i^c,i}).$$

Therefore, the RBR subproblem for (10.4) is

$$\min_{y \in \mathbb{R}^{n-1}} \quad \widehat{c}^\top y - \sigma \log(1 - y^\top B^{-1} y). \tag{10.5}$$

If $\gamma := \widehat{c}^\top B \widehat{c} > 0$, the solution of problem (10.5) is

$$y = -\frac{\sqrt{\sigma^2 + \gamma} - \sigma}{\gamma} B\widehat{c}. \tag{10.6}$$

Consequently, the subproblem (10.2) has the same solution as (10.5) if $\nu = 2\sigma \frac{\sqrt{\sigma^2+\gamma}-\sigma}{\gamma}$.

10.1.1. Examples: Phase Retrieval. Given a matrix $A \in \mathbb{C}^{m \times n}$ and a vector $b \in \mathbb{R}^m$, the phase retrieval problem can be formulated as a feasibility problem:

$$\text{find } x, \ \text{s.t. } |Ax| = b.$$

An equivalent model in [122] is

$$\begin{aligned} \min_{x \in \mathbb{C}^n, y \in \mathbb{R}^m} \quad & \frac{1}{2}\|Ax - y\|_2^2 \\ \text{s.t.} \quad & |y| = b, \end{aligned}$$

which can be further reformulated as

$$\begin{aligned} \min_{x\in\mathbb{C}^n, u\in\mathbb{C}^m} \ & \frac{1}{2}\|Ax - \mathrm{diag}(b)u\|_2^2 \\ \text{s.t.} \ & |u^i| = 1, , i = 1, \ldots, m. \end{aligned} \tag{10.7}$$

By fixing the variable u, it becomes a least squares problem with respect to x, whose explicit solution is $x = A^\dagger \mathrm{diag}(b)u$. Substituting x back to (10.7) yields a general maxcut problem:

$$\begin{aligned} \min_{u\in\mathbb{C}^m} \ & u^* M u \\ \text{s.t.} \ & |u^i| = 1, i = 1, \ldots, m, \end{aligned}$$

where $M = \mathrm{diag}(b)(I - AA^\dagger)\mathrm{diag}(b)$ is positive semidefinite. Hence, the corresponding SDP relaxation is

$$\begin{aligned} \min_{U\in\mathcal{S}^m} \ & \mathrm{tr}(UM) \\ \text{s.t.} \ & U^{ii} = 1, \ i = 1, \cdots, m, \ U \succeq 0. \end{aligned}$$

The above problem can be further solved by the RBR method.

10.2. Community Detection. Suppose that the nodes $[n] = \{1, \ldots, n\}$ of a network can be partitioned into $r \geq 2$ disjoint sets $\{K_1, \ldots, K_r\}$. A binary matrix X is called a partition matrix if $X^{ij} = 1$ for $i, j \in K_t, t \in \{1, ..., r\}$ and otherwise $X^{ij} = 0$. Let A be the adjacency matrix and d be the degree vector, where $d_i = \sum_j A^{ij}, i \in [n]$. Define the matrix

$$C = -(A - \lambda dd^\top), \tag{10.8}$$

where $\lambda = 1/\|d\|_1$. A popular method for the community detection problem is to maximize the modularity [86] as:

$$\min_X \langle C, X\rangle \ \text{s.t.} \ X \in \mathcal{P}_n^r, \tag{10.9}$$

where $\mathcal{P}_n^r$ is the set of all partition matrices of n nodes with no more than r subsets. Since the modularity optimization (10.9) is NP-hard, a SDP relaxation proposed in [25] is:

$$\begin{aligned} \min_{X\in\mathbb{R}^{n\times n}} \ & \langle C, X\rangle \\ \text{s.t.} \ & X^{ii} = 1, i = 1, \ldots, n, \\ & 0 \leq X^{ij} \leq 1, \forall i, j, \\ & X \succeq 0. \end{aligned} \tag{10.10}$$

The RBR method in subsection 10.1 can not be applied to (10.10) directly due to the componentwise constraints $0 \leq X^{ij} \leq 1$.

Note that the true partition matrix X^* can be decomposed as $X^* = \Phi^*(\Phi^*)^\top$, where $\Phi^* \in \{0, 1\}^{n\times r}$ is the true assignment matrix. This decomposition is unique up to a permutation of the columns of Φ^*. The structures of Φ^* leads to a new relaxation of the original partition matrix X [146]. Define a matrix

$$U = \left[u^1, ..., u^n\right]^\top \in \mathbb{R}^{n\times r}.$$

We can consider a decomposition $X = UU^\top$. The constraints $X^{ii} = 1$ and $\Phi^* \in \{0, 1\}^{n\times r}$ imply that

$$\|u^i\|_2 = 1, \quad U \geq 0, \quad \|u^i\|_0 \leq p,$$

where the cardinality constraints are added to impose sparsity of the solution. Therefore, an alternative relaxation to (10.9) is

$$\begin{aligned} \min_{U\in\mathbb{R}^{n\times r}} \quad & \left\langle C, UU^\top \right\rangle \\ \text{s.t.} \quad & \|u^i\|_2 = 1, i = 1,\dots,n, \\ & \|u^i\|_0 \le p, i = 1,\dots,n, \\ & U \ge 0. \end{aligned} \tag{10.11}$$

Although (10.11) is still NP-hard, it enables us to develop a computationally efficient RBR method. The feasible set for each block u^i is

$$\mathcal{U} = \{u \in \mathbb{R}^r \mid \|u\|_2 = 1, \quad u \ge 0, \quad \|u\|_0 \le p\}.$$

Then, problem (10.11) can be rewritten as

$$\min_{U\in\mathbb{R}^{n\times r}} \quad f(U) \equiv \left\langle C, UU^\top \right\rangle, \quad \text{s.t.} \quad u^i \in \mathcal{U}. \tag{10.12}$$

For the i-th subproblem, we fix all except the i-th row of U and formulate the subproblem as

$$u^i = \arg\min_{x\in\mathcal{U}} f(u^1,...,u^{i-1},x,u^{i+1},...,u^n) + \frac{\sigma}{2}\|x-\bar{u}^i\|^2,$$

where the last part in the objective function is the proximal term and $\sigma > 0$ is a parameter. Note that the quadratic term $\|x\|^2$ is eliminated due to the constraint $\|u\|_2 = 1$. Therefore, the subproblem becomes

$$u^i = \arg\min_{x\in\mathcal{U}} b^\top x, \tag{10.13}$$

where $b = 2C^{i,i^c}U^{-i} - \sigma\bar{u}^i$, and C^{i,i^c} is the i-th row of C without the i-th component, U^{-i} is the matrix U without the i-th row. Define $b_+ = \max\{b,0\}, b_- = \max\{-b,0\}$, where the max is taken component-wisely. Then the closed-form solution of (10.13) is given by

$$u = \begin{cases} \frac{b_-^p}{\|b_-^p\|}, & \text{if } b_- \ne 0, \\ e^{j_0}, \text{ with } j_0 = \arg\min_j b^j, & \text{otherwise}, \end{cases} \tag{10.14}$$

where b_-^p is obtained by keeping the largest p components in b_- and letting the others be zero, and when $\|b_-\|_0 \le p$, $b_-^p = b_-$. Then the RBR method goes over all rows of U by using (10.14).

We next briefly describe the parallelization of the RBR method on a shared memory computer with many threads. The variable U is stored in the shared memory so that it can be accessed by all threads. Even when some row u^i is updating in a thread, the other threads can still access U whenever necessary. In the sequential RBR method, the main cost of updating one row u^i is the computation of $b = 2C^{i,i^c}U^{-i} - \sigma\bar{u}^i$, where $\bar{u}^i$ and U are the current iterates. The definition of C in (10.8) gives

$$b^\top = -2A^{i,i^c}U^{-i} + 2\lambda d^i (d^{i^c})^\top U^{-i} - \sigma\bar{u}^i, \tag{10.15}$$

where A^{i,i^c} is the i-th row of A without the i-th component. The parallel RBR method is outlined in Figure 10.1 where many threads are working simultaneously. The vector $d^\top U$ and matrix U are stored in the shared memory and all threads can access and update them.

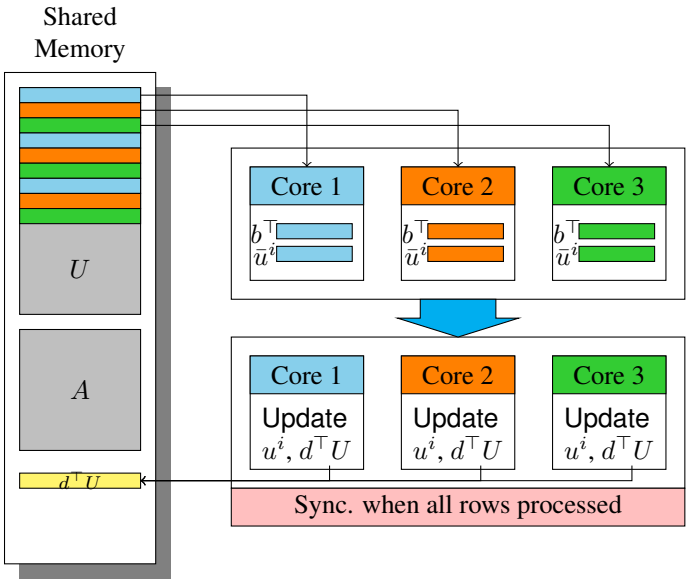

Fig. 10.1 *An illustration of the asynchronous parallel proximal RBR method*

Every thread picks up their own row u^i at a time and then reads U and the vector $d^\top U$. Then, a private copy of $b^\top$ is computed. Thereafter, the variable u^i is updated and $d^\top U$ is set to $d^\top U \leftarrow d^\top U + d^i(u^i - \bar{u}^i)$ in the shared memory. It immediately proceeds to another row without waiting for other threads to finish their tasks. Therefore, when a thread is updating its variables, other blocks of variables $u^j, j \neq i$ are not necessarily the most new version. Moreover, if this thread is reading some row u^j or the vector $d^\top U$ from memory and another thread is just modifying them, the data of u^i will be partially updated. Since the memory locking is removed, the parallel RBR method may be able to provide near-linear speedups. See also the HOGWILD! [94] and CYCLADES [89] for the asynchronous methods.

11. Low Rank Matrix Optimization. Optimization problems whose variable is related to low-rank matrices arise in many applications, for example, semidefinite programming (SDP), matrix completion, robust principle component analysis, control and systems theory, model reduction [76], phase retrieval, blind deconvolution, data mining, pattern recognitions [33], latent semantic indexing, collaborative prediction and low-dimensional embedding.

11.1. Low Rank Structure of First-order Methods. A common feature of many first-order methods for the low rank matrix optimization problems is that the next iterate x_{k+1} is defined by the current iterate x_k and a partial eigenvalue decomposition of certain matrix. They can be unified as the following fixed-point iteration scheme [71]:

$$x_{k+1} = \mathcal{T}(x_k, \Psi(\mathcal{B}(x_k))), \quad x_k \in \mathcal{D}, \tag{11.1}$$

where $\mathcal{B} : \mathcal{D} \to \mathcal{S}^n$ is a bounded mapping from a given Euclidean space $\mathcal{D}$ to the n-dimensional symmetric matrix space $\mathcal{S}^n$, and $\mathcal{T}$ is a general mapping from $\mathcal{D} \times \mathcal{S}^n$ to $\mathcal{D}$. The spectral operator $\Psi : \mathcal{S}^n \to \mathcal{S}^n$ is given by

$$\Psi(X) = V\mathrm{Diag}(\psi(\lambda(X)))V^\top, \tag{11.2}$$

where $X = V\mathrm{Diag}(\lambda_1, \ldots, \lambda_n)V^\top$ is the eigenvalue decomposition of X with eigenvalues in descending order $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_n$, $\lambda(X) = (\lambda_1, \ldots, \lambda_n)^T$, the operator $\psi : \mathbb{R}^n \to \mathbb{R}^n$ is a vector-valued *symmetric mapping*, i.e., $\psi(P\lambda) = P\psi(\lambda)$ for any permutation matrix P. The orthogonal projection of a symmetric matrix X on to a given $\mathrm{Range}(Q)$ with $Q^\top Q = I$ is defined as:

$$\mathcal{P}_Q(X) := \underset{Y \in \mathcal{S}^n,\ \mathrm{Range}(Y)=\mathrm{Range}(Q)}{\arg\min} \|Y - X\|_F^2 = QQ^\top XQQ^\top. \tag{11.3}$$

The operator Ψ has the low-rank property at X if there exists an orthogonal matrix $V_{\mathcal{I}} \in \mathbb{R}^{n\times p}$ $(p \ll n)$ that span a p-dimensional eigen-space corresponding to $\lambda_i(X), i \in \mathcal{I}$, such that $\Psi(X) = \Phi(\mathcal{P}_{V_{\mathcal{I}}}(X))$, where Φ is either the same as Ψ or a different spectral operator induced by ϕ, and $\mathcal{I}$ is an index set depending on X. The low-rank property ensures that the full eigenvalue decomposition is not needed.

The scheme (11.1) is time-consuming for large scale problems since first-order methods often take thousands of iterations to converge and each iteration requires at least one full or partial eigenvalue decomposition for evaluating Ψ. However, $\Psi(\mathcal{B}(x_k))$ often lives in a low-dimensional eigenspace in practice. A common practice is to use inexact method such as the Lanczos method, LOBPCG, and randomized methods with early stopping rules [149, 6, 106]. The so-called subspace method performs refinement on a low-dimensional subspace for univariate maximal eigenvalue optimization problem [66, 102, 63] and in the SCF iteration for KSDFT [151]. In the rest of this section, we present approaches [71] which integrate eigenvalue computation coherently with the underlying optimization methods.

11.2. A Polynomial-filtered Subspace Method. We now describe a general subspace framework for the scheme (11.1) using Chebyshev polynomials $\rho_k(\cdot)$ defined in (8.5). Assume that x^* is a limit point of the fixed-point iteration (11.1) and the low-rank property holds for every $\mathcal{B}(x_k)$ in (11.1). Consequently, the scheme (11.1) is equivalent to

$$x_{k+1} = \mathcal{T}(x_k, \Phi(\mathcal{P}_{V_{\mathcal{I}_k}}(\mathcal{B}(x_k)))), \tag{11.4}$$

where $V_{\mathcal{I}_k}$ is determined by $\mathcal{B}(x_k)$. Although the exact subspace $V_{\mathcal{I}_k}$ usually is unknown, it can be approximated by an estimated subspace U_k so that the computational cost of Ψ is significantly reduced. After the next point x_{k+1} is formed, a polynomial filter step is performed in order to extract a new subspace U_{k+1} based on U_k. Therefore, combining the two steps (8.6) and (11.4) together gives

$$x_{k+1} = \mathcal{T}(x_k, \Phi(\mathcal{P}_{U_k}(\mathcal{B}(x_k)))), \tag{11.5}$$

$$U_{k+1} = \mathrm{orth}(\rho_{k+1}^{q_{k+1}}(\mathcal{B}(x_{k+1}))U_k), \tag{11.6}$$

where q_k is a small number (e.g. 1 to 3) of the polynomial filter $\rho_k(\cdot)$ applied to U_k. The Chebyshev polynomials are suitable when the targeted eigenvalues are located within an interval, for example, finding a few largest/smallest eigenvalues in magnitude or all positive/negative eigenvalues.

The main feature is that the exact subspace $V_{\mathcal{I}_k}$ is substituted by its approximation U_k in (11.5). The principle angle between the true and extracted subspace is controlled by the polynomial degree. Then the error between one exact and inexact iteration is bounded. When the initial space is not orthogonal to the target space, the convergence of (11.5)-(11.6) is established under mild assumptions. In fact, the subspace often becomes more and more accurate so that the warm start property is helpful, i.e., the subspace of the current iteration can be refined from the previous one.

11.3. The Polynomial-filtered Proximal Gradient Method. We next show how to apply the subspace update (11.5) and (11.6) to the proximal gradient method on a set of composite optimization problems

$$\min \quad h(x) := F(x) + R(x), \tag{11.7}$$

where $F(x) = f \circ \lambda(\mathcal{B}(x))$ with $\mathcal{B}(x) = G + \mathcal{A}^*(x)$ and $R(x)$ is a regularization term with simple structures but need not be smooth. Here G is a known matrix in $\mathcal{S}^n$, the linear operator $\mathcal{A}$ and its adjoint operator $\mathcal{A}^*$ are defined as

$$\mathcal{A}(X) = [\langle A_1, X\rangle, \ldots, \langle A_m, X\rangle]^T, \quad \mathcal{A}^*(x) = \sum_{i=1}^m x_i A_i, \tag{11.8}$$

for given symmetric matrices $A_i \in \mathcal{S}^n$. The function $f : \mathbb{R}^n \to \mathbb{R}$ is smooth and *absolutely symmetric*, i.e., $f(x) = f(Px)$ for all $x \in \mathbb{R}^n$ and any permutation matrix $P \in \mathbb{R}^{n\times n}$.

Let Ψ be a spectral operator induced by $\psi = \nabla f$. It can be verified that the gradient of F in (11.7) is

$$\nabla F(x) = \mathcal{A}(\Psi(\mathcal{B}(x))). \tag{11.9}$$

The proximal operator is defined by

$$\mathrm{prox}_{tR}(x) = \arg\min_u R(u) + \frac{1}{2t}\|u - x\|_2^2. \tag{11.10}$$

Consequently, the proximal gradient method is

$$x_{k+1} = \mathrm{prox}_{\tau_k R}(x_k - \tau_k \mathcal{A}(\Psi(\mathcal{B}(x_k)))), \tag{11.11}$$

where τ_k is the step size. Therefore, the iteration (11.11) is a special case of (11.1) with

$$\begin{aligned} \mathcal{T}(x, X) &= \mathrm{prox}_{\tau_k R}(x - \mathcal{A}(X)), \\ \Psi(X) &= V\mathrm{Diag}(\nabla f(\lambda(X)))V^\top. \end{aligned}$$

Assume that the low-rank property holds at every iteration. The corresponding polynomial-filtered method can be written as

$$x_{k+1} = \mathrm{prox}_{\tau_k R}(x_k - \tau_k \mathcal{A}(\Phi(\mathcal{P}_{U_k}(\mathcal{B}(x_k))))), \tag{11.12}$$

$$U_{k+1} = \mathrm{orth}(\rho_{k+1}^{q_{k+1}}(\mathcal{B}(x_{k+1}))U_k). \tag{11.13}$$

11.3.1. Examples: Maximal Eigenvalue and Matrix Completion. Consider the maximal eigenvalue optimization problem:

$$\min_x F(x) + R(x) := \lambda_1(\mathcal{B}(x)) + R(x), \tag{11.14}$$

where $\mathcal{B}(x) = G + \mathcal{A}^*(x)$. Certain specific formulations of phase recovery and blind deconvolution are special case of (11.14). The subgradient of $F(x)$ is

$$\partial F(x) = \{\mathcal{A}(U_1 S U_1^T) \mid S \succeq 0, \mathrm{tr}(S) = 1\},$$

where $U_1 \in \mathbb{R}^{n\times r_1}$ is the subspace spanned by eigenvectors of $\lambda_1(\mathcal{B}(x))$ with multiplicity r_1. For simplicity, we assume $r_1 = 1$ and $\lambda_1(\mathcal{B}(x)) > 0$, which means that $\partial F(x)$ has only one element and the function $F(x)$ is differentiable. Then the polynomial-filtered method is

$$x_{k+1} = \mathrm{prox}_{\tau R}(x_k - \tau \mathcal{A}(u_1 u_1^T)),$$

where u_1 is the eigenvector of $\lambda_1(\mathcal{B}(x_k))$. Hence, we have

$$\mathcal{T}(x, W) = \mathrm{prox}_{\tau R}(x - \tau\mathcal{A}(W)), \quad \Psi(X) = u_1 u_1^T.$$

In addition, $\Psi(\cdot)$ satisfies the low-rank property around x^* with $\mathcal{I} = \{1\}$ and

$$(\psi(\lambda))_i = (\phi(\lambda))_i = \begin{cases} 1, & i = 1, \\ 0, & \text{otherwise.} \end{cases}$$

Another example is the penalized formulation of the matrix completion problem:

$$\min \|X\|_* + \frac{1}{2\mu}\|\mathcal{P}_\Omega(X - M)\|_F^2, \tag{11.15}$$

where Ω is a given index set of the true matrix M, and $\mathcal{P}_\Omega : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ denotes the projection operator onto the sparse matrix space with non-zero entries on Ω. Problem (11.15) can be solved by the proximal gradient method. At the k-th iteration, the main cost is to compute the truncated SVD of a matrix. Although (11.15) is not a direct special case of (11.7), we can still insert the polynomial filter into the proximal gradient method to reduce the cost of SVD.

11.4. The Polynomial-filtered ADMM Method. Consider the standard SDP:

$$\begin{aligned} \min \quad & \langle C, X\rangle, \\ \text{s.t.} \quad & \mathcal{A}X = b, \\ & X \succeq 0, \end{aligned} \tag{11.16}$$

where C, $\mathcal{A}$ and b are given, the linear operator $\mathcal{A}$ and its adjoint are defined in (11.8). Note that the ADMM on the dual problem of (11.16) is equivalent to the Douglas-Rachford Splitting (DRS) method [30] on the primal SDP (11.16). Define $F(X) = 1_{\{X\succeq 0\}}(X)$ and $G(X) = 1_{\{\mathcal{A}X=b\}}(X) + \langle C, X\rangle$, where $1_\Omega(X)$ is the indicator function on a set Ω. The proximal operators $\mathrm{prox}_{tF}(Z)$ and $\mathrm{prox}_{tG}(Y)$ can be computed explicitly as

$$\begin{aligned} \mathrm{prox}_{tF}(Z) &= \mathcal{P}_+(Z), \\ \mathrm{prox}_{tG}(Y) &= (Y + tC) - \mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}(\mathcal{A}Y + t\mathcal{A}C - b), \end{aligned}$$

where $\mathcal{P}_+(Z)$ is the projection operator onto the positive semi-definite cone. Hence, DRS can be formulated as

$$Z_{k+1} = T_{\text{DRS}}(Z_k) \triangleq \mathrm{prox}_{tG}(2\mathrm{prox}_{tF}(Z_k) - Z_k) - \mathrm{prox}_{tF}(Z_k) + Z_k,$$

which is also a special case of (11.1) with

$$\begin{aligned} \mathcal{T}(x, X) &= \mathrm{prox}_{tG}(2X - x) - X + x, \\ \Psi(X) &= \mathcal{P}_+(X). \end{aligned}$$

Note that $\mathcal{P}_+(X)$ is a spectral operator induced by ψ with the form

$$(\psi(\lambda))_i = \max\{\lambda_i, 0\}, \quad \forall i.$$

It can be verified that $\Psi(X) = \Psi(\mathcal{P}_{V_\mathcal{I}}(X))$, where $\mathcal{I}$ contains all indices of the positive eigenvalues $\lambda_i(X)$. The operator $\Psi(X)$ satisfies the low-rank property if X only has a few positive eigenvalues. Hence, the polynomial-filtered method method can be written as

$$Z_{k+1} = \mathrm{prox}_{tG}(2\mathcal{P}_+(\mathcal{P}_{U_k}(Z_k)) - Z_k) - \mathcal{P}_+(\mathcal{P}_{U_k}(Z_k)) + Z_k, \tag{11.17}$$

$$U_{k+1} = \mathrm{orth}(\rho_{k+1}^{q_{k+1}}(Z_{k+1})U_k). \tag{11.18}$$

11.4.1. Examples: 2-RDM and Cryo-EM. The two-body reduced density matrix (2-RDM) problem can be formulated as a standard SDP. It has a block diagonal structure with respect to the variable X, where each block is a low rank matrix. Hence, the polynomial filters can be applied to each block to reduce the cost. As an extension, we can plug polynomial filters into multi-block ADMM for the nonlinear SDPs from the weighted LS model with spectral norm constraints and least unsquared deviations (LUD) model in orientation determination of cryo-EM images [124]. For these examples we only introduce the formulation of the corresponding model. The details of the multi-block ADMM can be found in [124].

Suppose K is a given integer and S and W are two known matrices, the weighted LS model with spectral norm constraints is

$$\begin{array}{ll} \max & \langle W \odot S, G \rangle, \\ \text{s.t.} & G_{ii} = I_2, \\ & G \succeq 0, \\ & \|G\|_2 \leq \alpha K, \end{array} \tag{11.19}$$

where $G = (G_{ij})_{i,j=1,\ldots,K} \in \mathcal{S}^{2K}$ is the variable, with each block G_{ij} being a 2-by-2 small matrix, and $\|\cdot\|_2$ is the spectral norm. A three-block ADMM is introduced to solve (11.19). The cost of the projection onto the semidefinite cone can be reduced by the polynomial filters.

The semidefinite relaxation of the LUD problem is

$$\begin{array}{ll} \min & \sum_{1\leq i<j\leq K} \|c_{ij} - G_{ij}c_{ji}\|_2, \\ \text{s.t.} & G_{ii} = I_2, \\ & G \succeq 0, \\ & \|G\|_2 \leq \alpha K, \end{array} \tag{11.20}$$

where G, G_{ij}, K are defined the same in (11.19), and $c_{ij} \in \mathbb{R}^2$ are known vectors. The spectral norm constraint in (11.20) is optional. A four-block ADMM is proposed to solve (11.20). Similarly, the polynomial filters can be inserted into the ADMM update to reduce the computational cost.

12. Conclusion. In this paper, we provide a comprehensive survey on various subspace techniques for nonlinear optimization. The main idea of subspace algorithms aims to conquer large scale nonlinear problems by performing iterations in a lower dimensional subspace. We next summarize a few typical scenarios as follows.

- Find a linear combination of several known directions. Examples are the linear and nonlinear conjugate gradient methods, the Nesterov's accelerated gradient method, the Heavy-ball method and the momentum method.
- Keep the objective function and constraints, but add an extra restriction in a certain subspace. Examples are OMP, CoSaMP, LOBPCG, LMSVD, Arrabit, subspace refinement and multilevel methods.
- Approximate the objective objective function but keep the constraints. Examples are BCD, RBR, trust region with subspaces and parallel subspace correction.
- Approximate the objective objective function and design new constraints. Examples are trust region with subspaces and FPC_AS.
- Add a postprocess procedure after the subspace problem is solved. An example is the truncated subspace method for tensor train.
- Use subspace techniques to approximate the objective functions. Examples are sampling, sketching and Nyström approximation.
- Integrate the optimization method and subspace update in one framework. An example is the polynomial-filtered subspace method for low-rank matrix optimization.

The competitive performance of the methods adopting the above mentioned subspace techniques in the related examples implies that the subspace methods are very promising tools for large scale optimization problems. In fact, how to choose subspaces, how to construct subproblems, and how to solve them efficiently are the key questions of designing a successful subspace method. A good tradeoff between the simplicity of subproblems and the computational cost has to be made carefully. We are confident that many future directions are worth to be pursued from the point view of subspaces.

REFERENCES

[1] P.-A. ABSIL, C. G. BAKER, AND K. A. GALLIVAN, *Trust-region methods on Riemannian manifolds*, Found. Comput. Math., 7 (2007), pp. 303–330.

[2] P.-A. ABSIL, R. MAHONY, AND R. SEPULCHRE, *Optimization algorithms on matrix manifolds*, Princeton University Press, Princeton, NJ, 2008.

[3] P.-A. ABSIL, R. MAHONY, AND J. TRUMPF, *An extrinsic look at the Riemannian Hessian*, in Geometric science of information, Springer, 2013, pp. 361–368.

[4] D. G. ANDERSON, *Iterative procedures for nonlinear integral equations*, Journal of the ACM (JACM), 12 (1965), pp. 547–560.

[5] A. BECK AND M. TEBOULLE, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 183–202.

[6] S. BECKER, V. CEVHER, AND A. KYRILLIDIS, *Randomized low-memory singular value projection*, arXiv preprint arXiv:1303.0167, (2013).

[7] S. BELLAVIA AND B. MORINI, *A globally convergent Newton-GMRES subspace method for systems of nonlinear equations*, SIAM J. Sci. Comput., 23 (2001), pp. 940–960.

[8] S. J. BENSON, Y. YE, AND X. ZHANG, *Solving large-scale sparse semidefinite programs for combinatorial optimization*, SIAM J. Optim., 10 (2000), pp. 443–461.

[9] D. P. BERTSEKAS, *Nonlinear Programming*, Athena Scientific, September 1999.

[10] J. BOLTE, S. SABACH, AND M. TEBOULLE, *Proximal alternating linearized minimization for nonconvex and nonsmooth problems*, Math. Program., 146 (2014), pp. 459–494.

[11] S. BOYD, N. PARIKH, E. CHU, B. PELEATO, AND J. ECKSTEIN, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends® in Machine Learning, 3 (2011), pp. 1–122.

[12] M. A. BRANCH, T. F. COLEMAN, AND Y. LI, *A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems*, SIAM J. Sci. Comput., 21 (1999), pp. 1–23.

[13] C. BREZINSKI, M. REDIVO-ZAGLIA, AND Y. SAAD, *Shanks sequence transformations and Anderson acceleration*, SIAM Rev., 60 (2018), pp. 646–669.

[14] P. N. BROWN AND Y. SAAD, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 450–481.

[15] O. BURDAKOV, L. GONG, S. ZIKRIN, AND Y.-X. YUAN, *On efficiently combining limited-memory and trust-region techniques*, Math. Program. Comput., 9 (2017), pp. 101–134.

[16] S. BURER AND R. D. C. MONTEIRO, *A projected gradient algorithm for solving the maxcut SDP relaxation*, Optim. Methods Softw., 15 (2001), pp. 175–200.

[17] J. V. BURKE AND J. J. MORÉ, *On the identification of active constraints*, SIAM J. Numer. Anal., 25 (1988), pp. 1197–1211.

[18] ———, *Exposing constraints*, SIAM J. Optim., 4 (1994), pp. 573–595.

[19] R. H. BYRD, N. I. M. GOULD, J. NOCEDAL, AND R. A. WALTZ, *An algorithm for nonlinear optimization using linear programming and equality constrained subproblems*, Math. Program., 100 (2004), pp. 27–48.

[20] ———, *On the convergence of successive linear-quadratic programming algorithms*, SIAM J. Optim., 16 (2005), pp. 471–489.

[21] R. H. BYRD, J. NOCEDAL, AND R. B. SCHNABEL, *Representations of quasi-Newton matrices and their use in limited memory methods*, Math. Programming, 63 (1994), pp. 129–156.

[22] C. CARSTENSEN, *Domain decomposition for a non-smooth convex minimization problem and its application to plasticity*, Numerical linear algebra with applications, 4 (1997), pp. 177–190.

[23] M. R. CELIS, J. E. DENNIS, AND R. A. TAPIA, *A trust region strategy for nonlinear equality constrained optimization*, in Numerical optimization, 1984 (Boulder, Colo., 1984), SIAM, Philadelphia, PA, 1985, pp. 71–82.

[24] C. CHEN, Z. WEN, AND Y.-X. YUAN, *A general two-level subspace method for nonlinear optimization*, J. Comput. Math., 36 (2018), pp. 881–902.

[25] Y. CHEN, X. LI, AND J. XU, *Convexified modularity maximization for degree-corrected stochastic block*

models, Ann. Statist., 46 (2018), pp. 1573–1602.
[26] A. CONN, N. GOULD, A. SARTENAER, AND P. TOINT, *On iterated-subspace methods for nonlinear optimization*, in Linear and Nonlinear Conjugate Gradient-Related Methods, J. Adams and J. Nazareth, eds., 1996, pp. 50–79.
[27] W. DENG, M.-J. LAI, Z. PENG, AND W. YIN, *Parallel multi-block ADMM with $o(1/k)$ convergence*, J. Sci. Comput., 71 (2017), pp. 712–736.
[28] E. D. DOLAN, J. J. MORÉ, AND T. S. MUNSON, *Benchmarking optimization software with cops 3.0*, tech. rep., Mathematics and Computer Science Division, Argonne National Laboratory, February 2004.
[29] Q. DONG, X. LIU, Z.-W. WEN, AND Y.-X. YUAN, *A parallel line search subspace correction method for composite convex optimization*, J. Oper. Res. Soc. China, 3 (2015), pp. 163–187.
[30] J. DOUGLAS AND H. H. RACHFORD, *On the numerical solution of heat conduction problems in two and three space variables*, Transactions of the American mathematical Society, 82 (1956), pp. 421–439.
[31] A. EDELMAN, T. A. ARIAS, AND S. T. SMITH, *The geometry of algorithms with orthogonality constraints*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 303–353.
[32] M. ELAD, B. MATALON, AND M. ZIBULEVSKY, *Coordinate and subspace optimization methods for linear least squares with non-quadratic regularization*, Appl. Comput. Harmon. Anal., 23 (2007), pp. 346–367.
[33] L. ELDÉN, *Matrix Methods in Data Mining and Pattern Recognition (Fundamentals of Algorithms)*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2007.
[34] H.-R. FANG AND Y. SAAD, *A filtered Lanczos procedure for extreme and interior eigenvalue problems*, SIAM J. Sci. Comput., 34 (2012), pp. A2220–A2246.
[35] R. FLETCHER, *Second order corrections for nondifferentiable optimization*, in Numerical analysis (Dundee, 1981), vol. 912 of Lecture Notes in Math., Springer, Berlin-New York, 1982, pp. 85–114.
[36] M. FORNASIER, *Domain decomposition methods for linear inverse problems with sparsity constraints*, Inverse Problems, 23 (2007), p. 2505.
[37] M. FORNASIER, Y. KIM, A. LANGER, AND C.-B. SCHÖNLIEB, *Wavelet decomposition method for l_2/tv-image deblurring*, SIAM Journal on Imaging Sciences, 5 (2012), pp. 857–885.
[38] M. FORNASIER, A. LANGER, AND C.-B. SCHÖNLIEB, *A convergent overlapping domain decomposition method for total variation minimization*, Numerische Mathematik, 116 (2010), pp. 645–685.
[39] M. FORNASIER AND C.-B. SCHÖNLIEB, *Subspace correction methods for total variation and l_1-minimization*, SIAM Journal on Numerical Analysis, 47 (2009), pp. 3397–3428.
[40] D. GABAY, *Minimizing a differentiable function over a differential manifold*, J. Optim. Theory Appl., 37 (1982), pp. 177–219.
[41] D. GABAY AND B. MERCIER, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Computers & Mathematics with Applications, 2 (1976), pp. 17–40.
[42] E. GHADIMI, H. R. FEYZMAHDAVIAN, AND M. JOHANSSON, *Global convergence of the heavy-ball method for convex optimization*, in 2015 European Control Conference (ECC), 2015, pp. 310–315.
[43] P. E. GILL AND M. W. LEONARD, *Reduced-Hessian quasi-Newton methods for unconstrained optimization*, SIAM J. Optim., 12 (2001), pp. 209–237.
[44] ———, *Limited-memory reduced-Hessian methods for large-scale unconstrained optimization*, SIAM J. Optim., 14 (2003), pp. 380–401.
[45] R. GLOWINSKI AND A. MARROCCO, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité, d'une classe de problèmes de Dirichlet non linéaires*, Rev. Française Automat. Informat. Recherche Opérationnelle Sér. Rouge Anal. Numér., 9 (1975), pp. 41–76.
[46] M. X. GOEMANS AND D. P. WILLIAMSON, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, J. Assoc. Comput. Mach., 42 (1995), pp. 1115–1145.
[47] I. GOODFELLOW, Y. BENGIO, AND A. COURVILLE, *Deep Learning*, MIT Press, 2016. http://www.deeplearningbook.org.
[48] N. GOULD, D. ORBAN, AND P. TOINT, *Numerical methods for large-scale nonlinear optimization*, Acta Numer., 14 (2005), pp. 299–361.
[49] G. N. GRAPIGLIA, J. YUAN, AND Y.-X. YUAN, *A subspace version of the powell–yuan trust-region algorithm for equality constrained optimization*, Journal of the Operations Research Society of China, 1 (2013), pp. 425–451.
[50] G. N. GRAPIGLIA, Y. YUAN, AND Y.-X. YUAN, *A subspace version of the powell–yuan trust-region algorithm for equality constrained optimization*, J. Operations Research Society of China, 1 (2013), pp. 425–451.
[51] W. W. HAGER AND H. ZHANG, *A new active set algorithm for box constrained optimization*, SIAM J. Optim., 17 (2006), pp. 526–557.
[52] ———, *A new active set algorithm for box constrained optimization*, SIAM J. Optim., 17 (2006), pp. 526–557.
[53] E. T. HALE, W. YIN, AND Y. ZHANG, *Fixed-point continuation for l_1-minimization: methodology and*

convergence, SIAM J. Optim., 19 (2008), pp. 1107–1130.
[54] N. HALKO, P. G. MARTINSSON, AND J. A. TROPP, *Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288.
[55] B. HE, H.-K. XU, AND X. YUAN, *On the proximal Jacobian decomposition of ALM for multiple-block separable convex minimization problems and its relationship to ADMM*, J. Sci. Comput., 66 (2016), pp. 1204–1217.
[56] C. HELMBERG AND F. RENDL, *A spectral bundle method for semidefinite programming*, SIAM J. Optim., 10 (2000), pp. 673–696.
[57] J. M. HOKANSON, *Projected nonlinear least squares for exponential fitting*, SIAM J. Sci. Comput., 39 (2017), pp. A3107–A3128.
[58] J. HU, B. JIANG, L. LIN, Z. WEN, AND Y.-X. YUAN, *Structured quasi-Newton methods for optimization with orthogonality constraints*, SIAM J. Sci. Comput., 41 (2019), pp. A2239–A2269.
[59] J. HU, A. MILZAREK, Z. WEN, AND Y. YUAN, *Adaptive quadratically regularized Newton method for Riemannian optimization*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 1181–1207.
[60] W. HU, L. LIN, AND C. YANG, *Projected commutator DIIS method for accelerating hybrid functional electronic structure calculations*, J. Chem. Theory Comput., (2017).
[61] W. HUANG, P.-A. ABSIL, AND K. GALLIVAN, *A Riemannian BFGS method without differentiated retraction for nonconvex optimization problems*, SIAM J. Optim., 28 (2018), pp. 470–495.
[62] W. HUANG, K. A. GALLIVAN, AND P.-A. ABSIL, *A Broyden class of quasi-Newton methods for Riemannian optimization*, SIAM J. Optim., 25 (2015), pp. 1660–1685.
[63] F. KANGAL, K. MEERBERGEN, E. MENGI, AND W. MICHIELS, *A subspace method for large-scale eigenvalue optimization*, SIAM Journal on Matrix Analysis and Applications, 39 (2018), pp. 48–82.
[64] N. KESKAR, J. NOCEDAL, F. ÖZTOPRAK, AND A. WÄCHTER, *A second-order method for convex ℓ_1-regularized optimization with active-set prediction*, Optim. Methods Softw., 31 (2016), pp. 605–621.
[65] A. V. KNYAZEV, *Toward the optimal preconditioned eigensolver: locally optimal block preconditioned conjugate gradient method*, SIAM J. Sci. Comput., 23 (2001), pp. 517–541. Copper Mountain Conference (2000).
[66] D. KRESSNER, D. LU, AND B. VANDEREYCKEN, *Subspace acceleration for the crawford number and related eigenvalue optimization problems*, SIAM Journal on Matrix Analysis and Applications, 39 (2018), pp. 961–982.
[67] K. KREUTZ-DELGADO, *The complex gradient operator and the CR-calculus*, 2009. http://arxiv.org/abs/0906.4835.
[68] A. LANGER, S. OSHER, AND C.-B. SCHÖNLIEB, *Bregmanized domain decomposition for image restoration*, Journal of Scientific Computing, 54 (2013), pp. 549–576.
[69] C. LE BRIS, *Computational chemistry from the perspective of numerical analysis*, Acta Numer., 14 (2005), pp. 363–444.
[70] J. H. LEE, Y. M. JUNG, Y.-X. YUAN, AND S. YUN, *A subspace SQP method for equality constrained optimization*, Comput. Optim. Appl., 74 (2019), pp. 177–194.
[71] Y. LI, H. LIU, Z. WEN, AND Y. YUAN, *Low-rank matrix optimization using polynomial-filtered subspace extraction*.
[72] Y. LI AND S. OSHER, *Coordinate descent optimization for ℓ^1 minimization with application to compressed sensing; a greedy algorithm*, Inverse Probl. Imaging, 3 (2009), pp. 487–503.
[73] L. LIN, *Adaptively compressed exchange operator*, J. Chem. Theory Comput., 12 (2016), pp. 2242–2249.
[74] X. LIU, Z. WEN, AND Y. ZHANG, *Limited memory block Krylov subspace optimization for computing dominant singular value decompositions*, SIAM J. Sci. Comput., 35 (2013), pp. A1641–A1668.
[75] ———, *An efficient Gauss-Newton algorithm for symmetric low-rank product matrix approximations*, SIAM J. Optim., 25 (2015), pp. 1571–1608.
[76] Z. LIU AND L. VANDENBERGHE, *Interior-point method for nuclear norm approximation with application to system identification*, SIAM Journal on Matrix Analysis and Applications, 31 (2009), pp. 1235–1256.
[77] Z. Q. LUO AND P. TSENG, *On the convergence of the coordinate descent method for convex differentiable minimization*, J. Optim. Theory Appl., 72 (1992), pp. 7–35.
[78] M. W. MAHONEY, *Randomized algorithms for matrices and data*, Foundations and Trends® in Machine Learning, 3 (2011), pp. 123–224.
[79] J. MARTENS AND R. GROSSE, *Optimizing neural networks with kronecker-factored approximate curvature*, in International Conference on Machine Learning, 2015, pp. 2408–2417.
[80] D. Q. MAYNE AND E. POLAK, *A superlinearly convergent algorithm for constrained optimization problems*, Math. Programming Stud., (1982), pp. 45–61.
[81] J. J. MORÉ AND G. TORALDO, *Algorithms for bound constrained quadratic programming problems*, Numer. Math., 55 (1989), pp. 377–400.
[82] ———, *On the solution of large quadratic programming problems with bound constraints*, SIAM J. Optim., 1 (1991), pp. 93–113.
[83] D. NEEDELL AND J. A. TROPP, *CoSaMP: iterative signal recovery from incomplete and inaccurate sam-*

ples, Appl. Comput. Harmon. Anal., 26 (2009), pp. 301–321.
[84] Y. NESTEROV, *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer Science & Business Media, 2013.
[85] Y. E. NESTEROV, *A method for solving the convex programming problem with convergence rate o (1/k$\hat{2}$)*, in Dokl. Akad. Nauk SSSR, vol. 269, 1983, pp. 543–547.
[86] M. E. NEWMAN, *Modularity and community structure in networks*, Proceedings of the national academy of sciences, 103 (2006), pp. 8577–8582.
[87] Q. NI AND Y. YUAN, *A subspace limited memory quasi-Newton algorithm for large-scale nonlinear bound constrained optimization*, Math. Comp., 66 (1997), pp. 1509–1520.
[88] J. NOCEDAL AND S. J. WRIGHT, *Numerical Optimization*, Springer Series in Operations Research and Financial Engineering, Springer, New York, second ed., 2006.
[89] X. PAN, M. LAM, S. TU, D. PAPAILIOPOULOS, C. ZHANG, M. I. JORDAN, K. RAMCHANDRAN, C. RE, AND B. RECHT, *Cyclades: Conflict-free asynchronous machine learning*, in Advances in Neural Information Processing Systems, 2016, p. 2576–2584.
[90] B. POLYAK, *Some methods of speeding up the convergence of iteration methods*, USSR Computational Mathematics and Mathematical Physics, 4 (1964), pp. 1 – 17.
[91] M. J. D. POWELL AND Y. YUAN, *A trust region algorithm for equality constrained optimization*, Math. Programming, 49 (1990/91), pp. 189–211.
[92] P. PULAY, *Convergence acceleration of iterative sequences. the case of SCF iteration*, Chemical Physics Letters, 73 (1980), pp. 393–398.
[93] P. PULAY, *Improved SCF convergence acceleration*, Journal of Computational Chemistry, 3 (1982), pp. 556–560.
[94] B. RECHT, C. RE, S. WRIGHT, AND F. NIU, *Hogwild: A lock-free approach to parallelizing stochastic gradient descent*, in Advances in Neural Information Processing Systems, 2011, pp. 693–701.
[95] H. RUTISHAUSER, *Computational aspects of F. L. Bauer's simultaneous iteration method*, Numer. Math., 13 (1969), pp. 4–13.
[96] H. RUTISHAUSER, *Simultaneous iteration method for symmetric matrices*, Numer. Math., 16 (1970), pp. 205–223.
[97] Y. SAAD, *Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems*, Mathematics of Computation, 42 (1984), pp. 567–588.
[98] Y. SAAD, *Iterative methods for sparse linear systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, second ed., 2003.
[99] D. SCIEUR, A. D'ASPREMONT, AND F. BACH, *Regularized nonlinear acceleration*, in Advances In Neural Information Processing Systems, 2016, pp. 712–720.
[100] S. K. SHEVADE AND S. S. KEERTHI, *A simple and efficient algorithm for gene selection using sparse logistic regression*, Bioinformatics, 19 (2003), pp. 2246–2253.
[101] G. A. SHULTZ, R. B. SCHNABEL, AND R. H. BYRD, *A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties*, SIAM J. Numer. Anal., 22 (1985), pp. 47–67.
[102] P. SIRKOVIC AND D. KRESSNER, *Subspace acceleration for large-scale parameter-dependent Hermitian eigenproblems*, SIAM Journal on Matrix Analysis and Applications, 37 (2016), pp. 695–718.
[103] A. SIT, Z. WU, AND Y. YUAN, *A geometric buildup algorithm for the solution of the distance geometry problem using least-squares approximation*, Bull. Math. Biol., 71 (2009), pp. 1914–1933.
[104] S. T. SMITH, *Optimization techniques on Riemannian manifolds*, Fields Institute Communications, 3 (1994).
[105] S. SOLNTSEV, J. NOCEDAL, AND R. H. BYRD, *An algorithm for quadratic ℓ_1-regularized optimization with a flexible active-set strategy*, Optim. Methods Softw., 30 (2015), pp. 1213–1237.
[106] M. SOLTANI AND C. HEGDE, *Fast low-rank matrix estimation without the condition number*, arXiv preprint arXiv:1712.03281, (2017).
[107] T. STEIHAUG, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.
[108] G. W. STEWART, *Simultaneous iteration for computing invariant subspaces of non-Hermitian matrices*, Numer. Math., 25 (1975/76), pp. 123–136.
[109] ———, *Matrix algorithms Vol. II: Eigensystems*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001.
[110] W. J. STEWART AND A. JENNINGS, *A simultaneous iteration algorithm for real matrices*, ACM Trans. Math. Software, 7 (1981), pp. 184–198.
[111] W. SUN AND Y. YUAN, *Optimization theory and methods: nonlinear programming*, vol. 1, Springer Science & Business Media, 2006.
[112] X.-C. TAI AND J. XU, *Global and uniform convergence of subspace correction methods for some convex optimization problems*, Mathematics of Computation, 71 (2002), pp. 105–124.
[113] ———, *Global and uniform convergence of subspace correction methods for some convex optimization problems*, Math. Comp., 71 (2002), pp. 105–124.

[114] R. TIBSHIRANI, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society. Series B (Methodological), (1996), pp. 267–288.
[115] A. TOTH, J. A. ELLIS, T. EVANS, S. HAMILTON, C. KELLEY, R. PAWLOWSKI, AND S. SLATTERY, *Local improvement results for Anderson acceleration with inaccurate function evaluations*, SIAM Journal on Scientific Computing, 39 (2017), pp. S47–S65.
[116] J. A. TROPP AND A. C. GILBERT, *Signal recovery from random measurements via orthogonal matching pursuit*, IEEE Trans. Inform. Theory, 53 (2007), pp. 4655–4666.
[117] J. A. TROPP, A. YURTSEVER, M. UDELL, AND V. CEVHER, *Fixed-rank approximation of a positive-semidefinite matrix from streaming data*, in Advances in Neural Information Processing Systems, 2017, pp. 1225–1234.
[118] J. A. TROPP, A. YURTSEVER, M. UDELL, AND V. CEVHER, *Practical sketching algorithms for low-rank matrix approximation*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 1454–1485.
[119] P. TSENG AND S. YUN, *A coordinate gradient descent method for nonsmooth separable minimization*, Mathematical Programming, 117 (2009), pp. 387–423.
[120] C. UDRISTE, *Convex functions and optimization methods on Riemannian manifolds*, vol. 297, Springer Science & Business Media, 1994.
[121] B. VANDEREYCKEN, *Low-rank matrix completion by Riemannian optimization*, SIAM Journal on Optimization, 23 (2013), pp. 1214–1236.
[122] I. WALDSPURGER, A. D'ASPREMONT, AND S. MALLAT, *Phase recovery, MaxCut and complex semidefinite programming*, Math. Program., 149 (2015), pp. 47–81.
[123] H. F. WALKER AND P. NI, *Anderson acceleration for fixed-point iterations*, SIAM Journal on Numerical Analysis, 49 (2011), pp. 1715–1735.
[124] L. WANG, A. SINGER, AND Z. WEN, *Orientation determination of cryo-em images using least unsquared deviations*, SIAM journal on imaging sciences, 6 (2013), pp. 2450–2483.
[125] X. WANG, M. HONG, S. MA, AND Z.-Q. LUO, *Solving multiple-block separable convex minimization problems using two-block alternating direction method of multipliers*, arXiv preprint arXiv:1308.5294, (2013).
[126] Y. WANG, Z. JIA, AND Z. WEN, *The search direction correction makes first-order methods faster*, (2019). Arxiv 1905.06507.
[127] Z. WANG, Z. WEN, AND Y. YUAN, *A subspace trust region method for large scale unconstrained optimization*, in Numerical Linear Algebra and Optimization, Y.Yuan, ed., 2004, pp. 265–274.
[128] Z.-H. WANG AND Y.-X. YUAN, *A subspace implementation of quasi-Newton trust region methods for unconstrained optimization*, Numer. Math., 104 (2006), pp. 241–269.
[129] Z. WEN, D. GOLDFARB, AND K. SCHEINBERG, *Block coordinate descent methods for semidefinite programming*, Handbook on Semidefinite, Cone and Polynomial Optimization, (2011).
[130] ———, *Block coordinate descent methods for semidefinite programming*, in Handbook on semidefinite, conic and polynomial optimization, vol. 166 of Internat. Ser. Oper. Res. Management Sci., Springer, New York, 2012, pp. 533–564.
[131] Z. WEN, A. MILZAREK, M. ULBRICH, AND H. ZHANG, *Adaptive regularized self-consistent field iteration with exact Hessian for electronic structure calculation*, SIAM J. Sci. Comput., 35 (2013), pp. A1299–A1324.
[132] Z. WEN AND W. YIN, *A feasible method for optimization with orthogonality constraints*, Math. Program., 142 (2013), pp. 397–434.
[133] Z. WEN, W. YIN, D. GOLDFARB, AND Y. ZHANG, *A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization and continuation*, SIAM Journal on Scientific Computing, 32 (2010), pp. 1832–1857.
[134] Z. WEN, W. YIN, H. ZHANG, AND D. GOLDFARB, *On the convergence of an active-set method for l_1 minimization*, Optimization Methods and Software, 27 (2012), pp. 1127–1146.
[135] Z. WEN AND Y. ZHANG, *Accelerating convergence by augmented rayleigh–ritz projections for large-scale eigenpair computation*, SIAM Journal on Matrix Analysis and Applications, 38 (2017), pp. 273–296.
[136] D. P. WOODRUFF, *Sketching as a tool for numerical linear algebra*, Found. Trends Theor. Comput. Sci., 10 (2014), pp. iv+157.
[137] X. WU, Z. WEN, AND W. BAO, *A regularized newton method for computing ground states of Bose-Einstein condensates*, arXiv preprint arXiv:1504.02891, (2015).
[138] C. YANG, J. C. MEZA, AND L.-W. WANG, *A trust region direct constrained minimization algorithm for the Kohn-Sham equation*, SIAM J. Sci. Comput., 29 (2007), pp. 1854–1875.
[139] Y. YANG, B. DONG, AND Z. WEN, *Randomized algorithms for high quality treatment planning in volumetric modulated arc therapy*, Inverse Problems, 33 (2017), pp. 025007, 22.
[140] Y.-X. YUAN, *Subspace techniques for nonlinear optimization*, in Some topics in industrial and applied mathematics, vol. 8 of Ser. Contemp. Appl. Math. CAM, Higher Ed. Press, Beijing, 2007, pp. 206–218.
[141] Y.-X. YUAN, *Subspace methods for large scale nonlinear equations and nonlinear least squares*, Optim.

Eng., 10 (2009), pp. 207–218.
[142] Y.-X. YUAN, *A review on subspace methods for nonlinear optimization*, in Proceedings of the International Congress of Mathematicians—Seoul 2014. Vol. IV, Kyung Moon Sa, Seoul, 2014, pp. 807–827.
[143] Y.-X. YUAN AND J. STOER, *A subspace study on conjugate gradient algorithms*, Z. Angew. Math. Mech., 75 (1995), pp. 69–77.
[144] A. YURTSEVER, J. A. TROPP, O. FERCOQ, M. UDELL, AND V. CEVHER, *Scalable semidefinite programming*, 2019. arXiv:1912.02949.
[145] J. ZHANG, H. LIU, Z. WEN, AND S. ZHANG, *A sparse completely positive relaxation of the modularity maximization for community detection*, SIAM J. Sci. Comput., 40 (2018), pp. A3091–A3120.
[146] ———, *A sparse completely positive relaxation of the modularity maximization for community detection*, SIAM J. Sci. Comput., 40 (2018), pp. A3091–A3120.
[147] J. ZHANG, B. O'DONOGHUE, AND S. BOYD, *Globally convergent Type-I Anderson acceleration for nonsmooth fixed-point iterations*, (2018). arXiv:1808.03971.
[148] J. ZHANG, Z. WEN, AND Y. ZHANG, *Subspace methods with local refinements for eigenvalue computation using low-rank tensor-train format*, Journal of Scientific Computing, 70 (2017), pp. 478–499.
[149] T. ZHOU AND D. TAO, *Godec: Randomized low-rank & sparse matrix decomposition in noisy case*, in International conference on machine learning, Omnipress, 2011.
[150] Y. ZHOU AND Y. SAAD, *A Chebyshev–Davidson algorithm for large symmetric eigenproblems*, SIAM J. Matrix Anal. and Appl., 29 (2007), pp. 954–971.
[151] Y. ZHOU, Y. SAAD, M. L. TIAGO, AND J. R. CHELIKOWSKY, *Self-consistent-field calculations using Chebyshev-filtered subspace iteration*, Journal of Computational Physics, 219 (2006), pp. 172–184.