Mathematical Programming Computation manuscript No.
(will be inserted by the editor)

An augmented Lagrangian method with constraint generations for shape-constrained convex regression problems

Meixia Lin · Defeng Sun · Kim-Chuan Toh

Received: date / Accepted: date

Abstract Shape-constrained convex regression problem deals with fitting a convex function to the observed data, where additional constraints are imposed, such as component-wise monotonicity and uniform Lipschitz continuity. This paper provides a unified framework for computing the least squares estimator of a multivariate shape-constrained convex regression function in $\mathbb{R}^d$. We prove that the least squares estimator is computable via solving an essentially constrained convex quadratic programming (QP) problem with $(n+1)d$ variables, $n(n-1)$ linear inequality constraints and n possibly non-polyhedral inequality constraints, where n is the number of data points. To efficiently solve the generally very large-scale convex QP, we design a proximal augmented Lagrangian method (`proxALM`) whose subproblems are solved by the semismooth Newton method (`SSN`). To further accelerate the computation when n is huge, we design a practical implementation of the constraint generation method such that each reduced problem is efficiently solved by our proposed `proxALM`. Comprehensive numerical experiments, including those in the pricing of basket options and estimation of production functions in economics, demonstrate that our proposed `proxALM` outperforms the state-of-the-art algorithms, and

Defeng Sun is supported in part by Hong Kong Research Grant Council under grant number 15304019 and Kim-Chuan Toh by the Ministry of Education, Singapore, under its Academic Research Fund Tier 3 grant call (MOE-2019-T3-1-010).

Meixia Lin
Institute of Operations Research and Analytics, National University of Singapore, Singapore
E-mail: lin_meixia@u.nus.edu

Defeng Sun
Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong
E-mail: defeng.sun@polyu.edu.hk

Kim-Chuan Toh
Department of Mathematics and Institute of Operations Research and Analytics, National University of Singapore, Singapore
E-mail: mattohkc@nus.edu.sg

the proposed acceleration technique further shortens the computation time by a large margin.

Keywords Shape-constrainted convex regression · Preconditioned proximal point algorithm · Semismooth Newton method · Constraint generation method

Mathematics Subject Classification (2010) 90C06 · 90C25 · 90C90

1 Introduction

Convex (or concave) regression is meant to estimate a convex (or concave) function based on a finite number of observations. It is a topic of interest in many fields such as economics, operations research and financial engineering. In economics, production functions [21,45,2], demand functions [44] and utility functions [31] are often required to be concave. In operations research, the performance measure expectations can be proved to be convex in the underlying model parameters, e.g. in the context of queueing network [10]. In financial engineering, the option pricing function has the convexity restriction under the no-arbitrage condition, as can be seen from [1]. In the literature, there are various methods for solving the convex regression problem. With the specification of a functional form, one can apply a parametric approach to estimate the convex function. For example, the Cobb-Douglas production function is a particular functional form of the production function that is widely used in applied production economics. To avoid strong prior assumptions on the functional form, one can also use a non-parametric approach to perform the function estimation. Generally, the nonparametric estimation is based on a given collection of primitive functions, such as local polynomial [29], trigonometric series, spline estimator [15,36] and kernel-type estimator [3]. However, such an approach may face some difficulties in imposing the convexity constraint and choosing appropriate smoothing parameters (e.g. the degree of the polynomial, or the kernel density bandwidth). To overcome these difficulties, one may choose to estimate the functions by empirical risk minimization [14] over the set of convex functions, wherein the squared error loss [21] and the absolute error loss [8] are studied. In this paper, we focus on the least squares estimator for convex regression, whose theoretical properties are carefully studied in [20,42,28].

Suppose that we observe n data points $\{(X_i, Y_i)\}_{i=1}^n$, which satisfy the regression model $Y = \psi(X) + \varepsilon$ for an unknown convex function $\psi : \Omega \to \mathbb{R}$, where $\Omega \subset \mathbb{R}^d$ is a δ-neighborhood of $\mathrm{conv}(X_1, \cdots, X_n)$ (the convex hull of $\{X_i\}_{i=1}^n$), ε is a random variable with expectation $\mathbb{E}[\varepsilon|X] = 0$. The least squares estimator $\hat{\psi}$ of ψ is defined as

$$\hat{\psi} \in \operatorname*{arg\,min}_{\psi \in \mathcal{C}} \sum_{i=1}^{n} (\psi(X_i) - Y_i)^2, \quad \mathcal{C} := \{\psi : \Omega \to \mathbb{R} \mid \psi \text{ is a convex function}\}.$$

This infinite dimensional model appears to be intractable. Fortunately, the authors in [23,42] have provided a computationally tractable optimal solution to it. They showed that the family of convex functions can be characterized by a subset of continuous, piecewise linear functions $\theta_i+\langle\xi_i, X-X_i\rangle$, $i=1,\cdots,n$, whose intercepts θ_i's and gradient vectors ξ_i's are restricted to satisfy the convexity conditions. That is, a convex quadratic programming (QP) problem

$$\min_{\substack{\theta_1,\ldots,\theta_n\in\mathbb{R},\\ \xi_1,\ldots,\xi_n\in\mathbb{R}^d}} \left\{ \frac{1}{2}\sum_{i=1}^{n}(\theta_i - Y_i)^2 \;\middle|\; \theta_i \geq \theta_j + \langle \xi_j, X_i - X_j\rangle,\ 1\leq i,j\leq n \right\} \tag{1}$$

needs to be solved. The problem (1) with $(n+1)d$ variables and $n(n-1)$ linear inequality constraints can be solved by interior point solvers such as those implemented in `MOSEK` when n is not too large, as stated in [42]. However, interior point solvers may quickly run out of memory when n is large due to the presence of a large number of $n(n-1)$ linear inequality constraints. Mazumder et al. [30] adapted a three-block alternating direction method of multipliers (`ADMM`) to solve (1) but the method has no convergence guarantee. It needs about 1000 seconds to solve an instance with $d=4$, $n\sim 3000$ to get a rough approximate solution. As the objective function in (1) is not strongly convex, some papers including [3,12] do not deal with (P) exactly but perturb the problem by adding an additional ℓ_2 regularization term on the ξ_i's. The regularization term allows one to apply the accelerated proximal gradient (`APG`) method to the dual of the perturbed QP. For example, Aybat et al. [3] proposed a parallel `APG` method. However, it is still not fast enough for solving large problems as it needs 17 minutes to solve a problem with $d=80$, $n=1600$ on a 16-core machine sharing 32 GB. It should be noted that the regularization parameter may need to be extremely small in order for a solution of the perturbed QP to be optimal to the original QP under some kind of exact penalty property, while the dual of the perturbed QP also becomes harder to solve as the parameter becomes smaller. The computational challenge in solving the problem (1) still remains in need of more progress, especially for the case when d and n are relatively large where existing methods are too expensive even for computing a solution with a moderate accuracy.

In many real applications, one may need to impose more shape constraints on the convex function ψ, such as component-wise monotonicity and uniform Lipschitz continuity. For example, the option pricing function under the no-arbitrage condition needs to be non-decreasing as well as convex as described in [1]. In addition, when dealing with the Lipschitz convex regression as in [27, 4,30], the uniform Lipschitz property of the convex function is added when performing the estimation. For the shape-constrained convex regression problem, the least squares estimator $\hat{\psi}$ is defined as

$$\begin{aligned} &\hat{\psi} \in \arg\min_{\psi\in\mathcal{C}_{\mathcal{S}}} \sum_{i=1}^{n}(\psi(X_i)-Y_i)^2, \\ &\mathcal{C}_{\mathcal{S}} = \{\psi : \Omega\to\mathbb{R} \mid \psi \text{ is a convex function with Property } \mathcal{S}\}, \end{aligned} \tag{2}$$

where Property $\mathcal{S}$ specifies the shape constraint of ψ. We restrict ourselves to the case when Property $\mathcal{S}$ takes one of the following forms:

(S1) (monotone constraint) ψ is non-decreasing in some of the coordinates (denoted as K_1) and non-increasing in some others (denoted as K_2), where K_1 and K_2 are disjoint subsets of $\{1, \cdots, d\}$;
(S2) (box constraint) the elements in $\partial\psi(x)$ for any $x \in \Omega$ are bounded by two given vectors $L, U \in \mathbb{R}^d$;
(S3) (Lipschitz constraint) ψ is Lipschitz, i.e., $|\psi(x) - \psi(y)| \leq L\|x - y\|_p$ for any $x, y \in \Omega$, where $p = 1, 2, \infty$, and L is a given positive constant.

In this paper, we provide a unified framework for computing the least squares estimator for the shape-constrained convex regression problem (2). We prove that the minimal sum of squared error can be achieved via a set of piecewise linear functions whose intercept and gradient vectors are constrained to satisfy the convexity conditions and required shape constraints (see Theorem 1). This conclusion leads us to an essentially constrained QP with $(n+1)d$ variables, $n(n-1)$ linear inequality constraints and n possibly non-polyhedral constraints[1]. The addition of the shape constraints obviously would make the QP even more complicated and difficult to solve. Note that the estimator obtained in this way is nonsmooth, one can apply the Moreau proximal smoothing technique to obtain a smooth approximation. In addition, we can use a generalized form of the proposed constrained QP model as well as a data-driven Lipschitz estimation method to handle the boundary effect of the least squares estimator.

The main task in this framework is to solve the constrained QP in a robust and efficient manner. Most existing methods for the QP in the standard convex regression problem are either not extendable or difficult to be modified to efficiently solve the constrained QP due to the additional shape constraints. Moreover, except for interior point solvers which are only suitable for moderate size problems, almost all the other existing methods are first-order methods which may suffer from slow convergence rate when solving large-scale problems. For the multivariate shape-constrained convex regression problem, even with only a moderate number of observations, say $n = 10^3$, the memory cost and computational cost are already massive since the underlying QP has about a million constraints. To tackle the potentially very large-scale constrained QPs, we design an asymptotically superlinearly convergent proximal augmented Lagrangian method (`proxALM`), whose subproblems are solved by the semismooth Newton method (`SSN`), a second order method that has quadratic convergence. In the algorithm, the second order sparsity structure of the problem is fully uncovered and exploited to highly reduce the computational cost of solving the Newton systems. Comprehensive numerical experiments demonstrate that our proposed `proxALM` outperforms the state-of-the-art algorithms such as `MOSEK` and `ADMM` by a large margin.

[1] Strictly speaking, it is no longer a conventional QP problem in the presence of the non-polyhedral constraints. Slightly abusing the notation, here we use QP for convenience.

Note that when the number of observations is very large, memory issues may appear. For the case when n is huge, say $n = 10^5$, the constrained QP contains 10^{10} linear inequality constraints. As an illustration, a vector with dimension 10^{10} requires 74.5GB of RAM to store in dense double precision, which implies that it is almost impossible to solve the constrained QP with $n = 10^5$ on an ordinary desktop PC. This motivates us to explore the problem structure to overcome the computational and memory challenges of solving high-sample problems. As constraint generation techniques (also known as cutting plane methods) have been popular in solving linear programs with a large number of constraints [7], some researchers have applied this idea to solve convex regression problems. Hannah and Dunson [18] considered a globally convex regression model from locally linear estimates fitted on adaptively selected observations, and Balázs et al. [4] proposed an aggregate cutting plane method for solving the convex regression problem, but their computation was limited to moderate problem sizes or low accuracy. Bertsimas and Mundru [6] used a cutting plane method with each reduced problem solved by the commercial solver `Gurobi`. They reported solving an instance with $(n, d) = (10^5, 10^2)$ to moderate accuracy in about 7 hours. Recently, Chen and Mazumder [12] adapted the constraint generation method to solve the perturbed QP for the case when $n = 10^4, 10^5$, $d \leq 10$, where they applied the `APG` method to solve the dual of each reduced problem. However, the solutions they obtained are not guaranteed to satisfy the optimality conditions.

The main challenges of applying the constraint generation method to solve convex regression problems are summarized in two aspects. First, each reduced problem of the original QP without the perturbation term needs to be solved to sufficiently high accuracy in order to determine the violated constraint unambiguously. Second, given an approximate optimal solution, it is computationally expensive to search all $O(n^2)$ constraints to find the violated ones and check the optimality conditions. Note that existing interior point solvers or first-order algorithms (such as `APG` and `ADMM`) could not solve large-scale problems to high accuracy efficiently. Thus a constraint generation method employing those solvers needs to be conservative in allowing a small number of violated constraints to be added in each round. As a result, it may take many rounds of the constraint generation to find a solution with the required accuracy for the original QP. This implies that the computational cost of searching for violated constraints and checking optimality conditions can be very high, which is unaffordable in practice. Fortunately, our proposed `proxALM` allows us to solve large-scales problems to high accuracy efficiently, which motivates us to design a practical implementation of the constraint generation method to solve the shape-constrained convex regression problem. In our implementation, we add a relatively large number of most violated constraints in each round to greatly reduce the number of rounds of the constraint generation. For each reduced problem, we apply our `proxALM`, which is demonstrated to be much more efficient in solving large-scale problems than other state-of-the-art algorithms.

We summarize our main contributions in this paper as follows.

1 We provide a unified framework for computing the least squares estimator for the shape-constrained convex regression problem (2), wherein a constrained QP with $(n+1)d$ variables, $n(n-1)$ linear inequality constraints and n possibly non-polyhedral inequality constraints needs to be solved.
2 We propose an asymptotically superlinearly convergent proximal augmented Lagrangian method to solve the constrained QP, where each subproblem of the `proxALM` is solved by the semismooth Newton method. We analyse the second order sparsity structure of the subproblems and develop novel numerical techniques to solve the semismooth Newton linear systems efficiently through exploiting the uncovered structure. Comprehensive numerical experiments, including those in the pricing of basket options and estimation of production functions, demonstrate that the proposed `proxALM` outperforms other state-of-the-art algorithms such as `MOSEK` and `ADMM` by a large margin, especially for large-scale problems.
3 To solve the shape-constrained convex regression problem with a huge sample size, we design a practical implementation of the constraint generation method where each of its reduced problem is solved by our proposed `proxALM`. Numerical experiments are also performed to demonstrate the high efficiency of the constraint generation method with `proxALM`.

In the remaining part of the paper, we provide a unified framework for estimating the multivariate shape-constrained convex function in Section 2. For solving the involved constrained QP, the proximal augmented Lagrangian method is described in Section 3. The implementation details of the proposed `proxALM` can be found in Section 4. In Section 5, we design a practical implementation of the constraint generation method to solve shape-constrained convex regression problems with huge samples sizes. Section 6 provides the numerical comparison of `proxALM` with other start-of-the-art algorithms. Experiments are also conducted to demonstrate the superior performance of the constraint generation method combined with the `proxALM` for solving instances with huge samples sizes. Then we apply our framework to perform the function estimation in several interesting real applications in Section 7. Finally, we conclude the paper.

Notation. Denote $X = (X_1, \cdots, X_n) \in \mathbb{R}^{d\times n}$, $e_n \in \mathbb{R}^n$ be the vector of all ones, I_n be the $n \times n$ identity matrix. For any matrix $Z \in \mathbb{R}^{m\times n}$, Z_i denotes the i-th column of Z. We use "Diag(z)" to denote the diagonal matrix whose diagonal is given by the vector z, and use Diag$(X_1, \cdots, X_n)$ to denote the block diagonal matrix whose i-th block is the matrix X_i. For any symmetric and positive semidefinite matrix $H \in \mathbb{R}^{n\times n}$, we define $\langle x, x'\rangle_H := \langle x, Hx'\rangle$, and $\|x\|_H := \sqrt{\langle x, x\rangle_H}$ for all $x, x' \in \mathbb{R}^n$. For a given closed subset C of $\mathbb{R}^n$ and $x \in \mathbb{R}^n$, we define $\mathrm{dist}_H(x, C) = \min\{\|x - y\|_H \mid y \in C\}$. The largest (smallest) eigenvalue of H is denoted as $\lambda_{\max}(H)$ ($\lambda_{\min}(H)$). Given $x \in \mathbb{R}^n$ and an index set $K \subset \{1, \cdots, n\}$, x_K denotes the sub-vector of x with those elements not in K being removed. Let $q : \mathbb{R}^n \to (-\infty, \infty]$ be a closed proper convex function. The conjugate of q is $q^*(z) := \sup_{x\in\mathbb{R}^n}\{\langle x, z\rangle - q(x)\}$ and the

Moreau envelope of q at x is defined by

$$\mathrm{E}_q(x) := \min_{y\in\mathbb{R}^n}\left\{q(y)+\frac{1}{2}\|y-x\|^2\right\},$$

and the associated proximal mapping $\mathrm{Prox}_q(x)$ is defined as the unique solution of the above minimization problem. As proved in [33], $\mathrm{E}_q(\cdot)$ is finite-valued, convex and differentiable with $\nabla\mathrm{E}_q(x) = x - \mathrm{Prox}_q(x)$. In addition, we can see from [35,41] that $\mathrm{Prox}_q(x)$ is Lipschitz continuous with modulus 1.

2 A unified framework to estimate the multivariate shape-constrained convex function

In this section, we provide a unified framework for computing the least squares estimator for the multivariate shape-constrained convex function defined in (2). Before describing the process, we first characterize Property $\mathcal{S}$ in the following proposition. For brevity, we omit the proof.

Proposition 1 *A convex function ψ has Property $\mathcal{S}$ if and only if for any $x\in\mathbb{R}^d$, the subdifferential of ψ satisfies $\partial\psi(x)\subset\mathcal{D}$, where $\mathcal{D}$ is defined corresponding to Property $\mathcal{S}$ as follows:*

(S1) (monotone constraint) $\mathcal{D}=\{x\in\mathbb{R}^d \mid x_{K_1}\geq 0, x_{K_2}\leq 0\}$,
(S2) (box constraint) $\mathcal{D}=\{x\in\mathbb{R}^d \mid L\leq x\leq U\}$,
(S3) (Lipschitz constraint) $\mathcal{D}=\{x\in\mathbb{R}^d \mid \|x\|_q\leq L\}$, *where q satisfies* $1/p+1/q=1$. *In particular, $q=\infty,2,1$ when $p=1,2,\infty$, respectively.*

The least squares estimation problem (2) attempts to find a best-fitting function $\hat{\psi}$ from the function family $\mathcal{C}_{\mathcal{S}}$, which is infinite dimensional. Therefore, this problem is intractable in practice. In order to design a tractable approach, we establish the following representation theorem to (2), which is motivated by [23].

Theorem 1 *Define the set of piecewise linear functions as*

$$\mathcal{K}_{\mathcal{S}} := \left\{\phi:\Omega\to\mathbb{R}\,\middle|\,\begin{array}{l}\phi(x)=\max\limits_{1\leq j\leq n}\{\theta_j+\langle\xi_j,x-X_j\rangle\},\\(\theta_1,\cdots,\theta_n,\xi_1,\cdots,\xi_n)\in\mathcal{F}_{\mathcal{S}}\end{array}\right\}, \tag{3}$$

where

$$\mathcal{F}_{\mathcal{S}} := \left\{(\theta_1,\cdots,\theta_n,\xi_1,\cdots,\xi_n)\,\middle|\,\begin{array}{l}\theta_i\in\mathbb{R},\xi_i\in\mathcal{D},i=1,\cdots,n,\\\theta_i\geq\theta_j+\langle\xi_j,X_i-X_j\rangle,1\leq i,j\leq n\end{array}\right\}, \tag{4}$$

and $\mathcal{D}$ is defined as in Proposition 1. Consider the problem

$$\min_{\phi\in\mathcal{K}_{\mathcal{S}}}\sum_{i=1}^{n}(\phi(X_i)-Y_i)^2. \tag{5}$$

Then the following equality holds:

$$\min_{\psi\in\mathcal{C}_{\mathcal{S}}}\sum_{i=1}^{n}(\psi(X_i)-Y_i)^2=\min_{\phi\in\mathcal{K}_{\mathcal{S}}}\sum_{i=1}^{n}(\phi(X_i)-Y_i)^2. \tag{6}$$

Moreover, any solution $\hat{\phi}$ *to* (5) *is a solution to the problem* (2).

Proof We first prove that $\mathcal{K}_{\mathcal{S}}\subset\mathcal{C}_{\mathcal{S}}$, that is, the functions in $\mathcal{K}_{\mathcal{S}}$ are convex functions with Property $\mathcal{S}$. Convexity comes from the fact that any pointwise maximum function is convex. Given any function $\phi\in\mathcal{K}_{\mathcal{S}}$ determined by $(\theta_1,\cdots,\theta_n,\xi_1,\cdots,\xi_n)\in\mathcal{F}_{\mathcal{S}}$, the subdifferential of this piecewise linear function is a polyhedron according to [40, Theorem 25.6], and it is given by

$$\partial\phi(x)=\mathrm{conv}\{\xi_i\mid i\in I(x)\},\quad I(x):=\{i\mid\theta_i+\langle\xi_i,x-X_i\rangle=\phi(x)\}.$$

By the definition of $\mathcal{D}$ and $\mathcal{F}_{\mathcal{S}}$, we can see that $\partial\phi(x)\subset\mathcal{D}$ for any $x\in\Omega$. According to Proposition 1, the convex function ϕ has Property $\mathcal{S}$, which means $\phi\in\mathcal{C}_{\mathcal{S}}$. Hence $\mathcal{K}_{\mathcal{S}}\subset\mathcal{C}_{\mathcal{S}}$. Therefore, we have that

$$\min_{\psi\in\mathcal{C}_{\mathcal{S}}}\sum_{i=1}^{n}(\psi(X_i)-Y_i)^2\le\min_{\phi\in\mathcal{K}_{\mathcal{S}}}\sum_{i=1}^{n}(\phi(X_i)-Y_i)^2.$$

Next we prove the reverse inequality. Let $\varepsilon>0$ be an arbitrary positive number. Then there exists $\hat{\psi}_\varepsilon\in\mathcal{C}_{\mathcal{S}}$ such that

$$\sum_{i=1}^{n}(\hat{\psi}_\varepsilon(X_i)-Y_i)^2\le\min_{\psi\in\mathcal{C}_{\mathcal{S}}}\sum_{i=1}^{n}(\psi(X_i)-Y_i)^2+\varepsilon.$$

For $i=1,\cdots,n$, choose $\hat{\xi}_{\varepsilon,i}\in\partial\hat{\psi}_\varepsilon(X_i)$. Then

$$(\hat{\psi}_\varepsilon(X_1),\cdots,\hat{\psi}_\varepsilon(X_n),\hat{\xi}_{\varepsilon,1},\cdots,\hat{\xi}_{\varepsilon,n})\in\mathcal{F}_{\mathcal{S}}$$

and

$$\hat{\phi}_\varepsilon(x):=\max_{1\le j\le n}\{\hat{\psi}_\varepsilon(X_j)+\langle\hat{\xi}_{\varepsilon,j},x-X_j\rangle\}\in\mathcal{K}_{\mathcal{S}}.$$

The fact that inequalities $\hat{\psi}_\varepsilon(X_i)\ge\hat{\psi}_\varepsilon(X_j)+\langle\hat{\xi}_{\varepsilon,j},X_i-X_j\rangle$ hold for all i,j implies that

$$\hat{\phi}_\varepsilon(X_i)=\max_{1\le j\le n}\{\hat{\psi}_\varepsilon(X_j)+\langle\hat{\xi}_{\varepsilon,j},X_i-X_j\rangle\}=\hat{\psi}_\varepsilon(X_i),\quad i=1,\cdots,n.$$

Then, it holds that

$$\begin{aligned}\min_{\phi\in\mathcal{K}_{\mathcal{S}}}\sum_{i=1}^{n}(\phi(X_i)-Y_i)^2&\le\sum_{i=1}^{n}(\hat{\phi}_\varepsilon(X_i)-Y_i)^2=\sum_{i=1}^{n}(\hat{\psi}_\varepsilon(X_i)-Y_i)^2\\&\le\min_{\psi\in\mathcal{C}_{\mathcal{S}}}\sum_{i=1}^{n}(\psi(X_i)-Y_i)^2+\varepsilon.\end{aligned}$$

Since the above inequality holds for any $\varepsilon>0$, the equality (6) follows. Now suppose that $\hat{\phi}$ is an optimal solution to (5). Since $\hat{\phi}\in\mathcal{C}_{\mathcal{S}}$, from (6) we know that $\hat{\phi}$ is a solution to the problem (2). □

The theorem above provides a tractable approach to compute (2) through solving (5). By definition, any function ϕ in $\mathcal{K}_{\mathcal{S}}$, which is determined by $(\theta_1, \cdots, \theta_n, \xi_1, \cdots, \xi_n) \in \mathcal{F}_{\mathcal{S}}$, satisfies

$$\phi(X_i) = \max_{1 \le j \le n} \{\theta_j + \langle \xi_j, X_i - X_j \rangle\} = \theta_i, \quad i = 1, \cdots, n.$$

Therefore, we can conclude the framework for computing an optimal solution to (2) as follows.

A unified framework for shape-constrained convex regression. Suppose that $\{(\hat{\theta}_i, \hat{\xi}_i)\}_{i=1}^n$ is an optimal solution to

$$\min_{\theta_1,\ldots,\theta_n \in \mathbb{R}; \xi_1,\ldots,\xi_n \in \mathbb{R}^d} \Big\{ \frac{1}{2}\|\theta - Y\|^2 \;\Big|\; (\theta_1, \cdots, \theta_n, \xi_1, \cdots, \xi_n) \in \mathcal{F}_{\mathcal{S}} \Big\}, \quad (7)$$

where the feasible set $\mathcal{F}_{\mathcal{S}}$ is defined as in (4). We can construct an optimal solution to (2) by taking

$$\hat{\psi}(x) = \max_{1 \le j \le n} \Big\{ \hat{\theta}_j + \langle \hat{\xi}_j, x - X_j \rangle \Big\}, \quad x \in \Omega. \quad (8)$$

As one can see, the main task in our framework for estimating the shape-constrained convex function is to solve the constrained convex quadratic programming problem (7).

Define the matrix $A = e_n \otimes I_n - I_n \otimes e_n \in \mathbb{R}^{n^2 \times n}$, where " $\otimes$ " denotes the Kronecker product. Then it could be seen that $A^T A = 2nI_n - 2e_n e_n^T$. Denote $\xi = (\xi_1; \cdots; \xi_n) \in \mathbb{R}^{dn}$ and $B = \mathrm{Diag}(B_1, \cdots, B_n) \in \mathbb{R}^{n^2 \times dn}$ with $B_i = e_n X_i^T - X^T \in \mathbb{R}^{n \times d}$ for $i = 1, \cdots, n$. Based on these notations, the problem (7) can equivalently be written as

$$\min_{\theta \in \mathbb{R}^n, \xi \in \mathbb{R}^{dn}} \Big\{ \frac{1}{2}\|\theta - Y\|^2 + p(\xi) + \delta_+(A\theta + B\xi) \Big\}, \quad (\mathrm{P})$$

where $p(\xi) = \sum_{i=1}^n \delta_{\mathcal{D}}(\xi_i)$ and $\delta_+(\cdot)$ is the indicator function of $\mathbb{R}_+^{n^2}$.

Smooth approximation. Note that the function $\hat{\psi}$ obtained by (8) is nonsmooth. When a smooth function is required, we can compute a smooth approximation to $\hat{\psi}$. The idea of Nesterov's smoothing [34] could be applied, and the details is described in [30, Section 3]. Alternatively, one can use the Moreau envelope as a smooth approximation of $\hat{\psi}$, namely

$$\hat{\psi}_\tau^{\mathrm{M}}(x) = \tau \mathrm{E}_{\hat{\psi}/\tau}(x) = \min_{y \in \mathbb{R}^d} \Big\{ \hat{\psi}(y) + \frac{\tau}{2}\|y - x\|^2 \Big\}, \quad (9)$$

where $\tau > 0$ is a regularization parameter. Note that

$$\hat{\psi}_\tau^{\mathrm{M}}(x) = \min_{y \in \mathbb{R}^d, t \in \mathbb{R}} \Big\{ t + \frac{\tau}{2}\|y - x\|^2 \;\Big|\; t \ge \langle \hat{\xi}_j, y \rangle - \langle \hat{\xi}_j, X_j \rangle + \hat{\theta}_j,\ j = 1, \cdots, n \Big\},$$

and the unique optimal solution $\mathrm{Prox}_{\hat{\psi}/\tau}(x)$ of (9) can be obtained by solving a quadratic programming of dimension $d+1$, which could be efficiently computed by `Gurobi` or `MOSEK`. One can see that for any $\tau > 0$, $\hat{\psi}_\tau^{\mathrm{M}}$ is convex, and differentiable with $\nabla\hat{\psi}_\tau^{\mathrm{M}}(x) = \tau(x - \mathrm{Prox}_{\hat{\psi}/\tau}(x))$. In addition, according to [5], the approximation $\hat{\psi}_\tau^{\mathrm{M}}$ of $\hat{\psi}$ satisfies the approximation bound

$$0 \le \hat{\psi}(x) - \hat{\psi}_\tau^{\mathrm{M}}(x) \le \frac{1}{2\tau}\mathrm{dist}^2(0, \partial\hat{\psi}(x)) \le \frac{L^2}{2\tau}, \quad \forall x \in \Omega,$$

where $L = \max\{\|\xi_j\|_2 \mid j = 1, \cdots, n\}$.

3 A proximal augmented Lagrangian method (`proxALM`) for (P)

The augmented Lagrangian method is a desirable method for solving convex composite programming problems due to its superlinear convergence. To take advantage of the fast local convergence, we design a proximal augmented Lagrangian method for solving (P). In order to solve the `proxALM` subproblems, we propose a semismooth Newton method, which is proved to have quadratic convergence. By making full use of the special structure of the problem, we can exploit the second-order sparsity structure of the underlying subproblems to greatly reduce the computational cost. It should be noted that in addition to the algorithmic design, the most important part of the `proxALM` is the numerical implementation, which will be discussed in detail in the next section.

The Lagrangian function associated with the unconstrained minimization problem (P) is given by

$$\begin{aligned} & l(\theta, \xi; u, v) \\ &= \inf_{\eta \in \mathbb{R}^{n^2}, y \in \mathbb{R}^{dn}} \Big\{ \frac{1}{2}\|\theta - Y\|^2 + p(\xi - y) + \delta_+(A\theta + B\xi - \eta) - \langle u, \eta\rangle - \langle v, y\rangle \Big\} \\ &= \frac{1}{2}\|\theta - Y\|^2 - p^*(-v) - \langle v, \xi\rangle - \delta_+(u) - \langle u, A\theta + B\xi\rangle. \end{aligned}$$

The dual problem of (P), $\max_{u \in \mathbb{R}^{n^2}, v \in \mathbb{R}^{dn}} \min_{\theta \in \mathbb{R}^n, \xi \in \mathbb{R}^{dn}} l(\theta, \xi; u, v)$, is explicitly given as follows:

$$\begin{aligned} \max_{u \in \mathbb{R}^{n^2}, v \in \mathbb{R}^{dn}} \ & \Big\{ -\frac{1}{2}\|A^T u\|^2 - \langle Y, A^T u\rangle - p^*(-v) - \delta_+(u) \Big\} \\ \text{s.t.} \ & B^T u + v = 0. \end{aligned} \tag{D}$$

The Karush-Kuhn-Tucker (KKT) conditions associated with (P) and (D) are:

$$\theta - Y - A^T u = 0, \ B^T u + v = 0, \ -v \in \partial p(\xi), \ -u \in \partial\delta_+(A\theta + B\xi). \tag{10}$$

The augmented Lagrangian function associated with (P) for any fixed $\sigma > 0$ can be derived as

$$\begin{aligned}
&\mathcal{L}_\sigma(\theta, \xi; u, v) \\
&= \sup_{s\in\mathbb{R}^{n^2}, t\in\mathbb{R}^{dn}} \Big\{ l(\theta, \xi; s, t) - \frac{1}{2\sigma}\|s-u\|^2 - \frac{1}{2\sigma}\|t-v\|^2 \Big\} \\
&= \frac{1}{2}\|\theta - Y\|^2 + \sigma \mathrm{E}_p(\xi - \frac{v}{\sigma}) + \sigma \mathrm{E}_{\delta_+}(A\theta + B\xi - \frac{u}{\sigma}) - \frac{1}{2\sigma}\|u\|^2 - \frac{1}{2\sigma}\|v\|^2.
\end{aligned}$$

Our proposed `proxALM` for solving (P) has the following template.

Algorithm 1 : Proximal augmented Lagrangian method for (P)

1: **Initialization:** Let $H_1 \in \mathbb{R}^{n\times n}$, $H_2 \in \mathbb{R}^{dn\times dn}$ be given symmetric and positive definite matrices, and $\{\varepsilon_k\}$ be a given summable sequence of nonnegative numbers. Choose an initial point $(\theta^0, \xi^0, u^0, v^0) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$, $\sigma_0 > 0$. For $k = 0, 1, 2, \ldots$

2: **repeat**

3: **Step 1**. Compute

$$\begin{aligned}
&(\theta^{k+1}, \xi^{k+1}) \\
&\approx \mathop{\arg\min}_{\theta\in\mathbb{R}^n, \xi\in\mathbb{R}^{dn}} \Big\{ \Phi_k(\theta, \xi) = \mathcal{L}_{\sigma_k}(\theta, \xi; u^k, v^k) + \frac{1}{2\sigma_k}\|\theta - \theta^k\|^2_{H_1} + \frac{1}{2\sigma_k}\|\xi - \xi^k\|^2_{H_2} \Big\} \quad (11)
\end{aligned}$$

such that the approximate solution $(\theta^{k+1}, \xi^{k+1})$ satisfies the following stopping criterion:

$$\|\nabla \Phi_k(\theta^{k+1}, \xi^{k+1})\| \le \frac{\sqrt{\lambda_{\min}}}{\sigma_k}\varepsilon_k, \quad (A)$$

where $\lambda_{\min} = \min\{\lambda_{\min}(H_1), \lambda_{\min}(H_2), 1\}$.

4: **Step 2**. Update u, v by

$$\begin{aligned}
u^{k+1} &= -\sigma_k \Big[A\theta^{k+1} + B\xi^{k+1} - u^k/\sigma^k - \Pi_+(A\theta^{k+1} + B\xi^{k+1} - u^k/\sigma^k) \Big], \\
v^{k+1} &= -\sigma_k \Big[\xi^{k+1} - v^k/\sigma_k - \mathrm{Prox}_p(\xi^{k+1} - v^k/\sigma^k) \Big].
\end{aligned}$$

5: **Step 3**. Update $\sigma_{k+1} \uparrow \sigma_\infty \le \infty$.

6: **until** Stopping criterion is satisfied.

3.1 Convergence results for the `proxALM`

Define the maximal monotone operator $\mathcal{T}_l$ as

$$\mathcal{T}_l(\theta, \xi, u, v) = \Big\{ (\theta', \xi', u', v') \mid (\theta', \xi', -u', -v') \in \partial l(\theta, \xi, u, v) \Big\},$$

and the block diagonal operator $\Sigma = \mathrm{Diag}(H_1, H_2, I_{n^2}, I_{dn})$. Note that the solution set of the KKT system (10) is exactly $\mathcal{T}_l^{-1}(0)$.

We follow the idea of [25, Theorem 2.3 and Theorem 2.5] to get the following convergence results of Algorithm 1, where the details of the proof are omitted here.

Theorem 2 *Suppose that the solution set to the KKT system* (10) *is nonempty, that is* $\Lambda := \mathcal{T}_l^{-1}(0) \neq \emptyset$.
(1) Let $\{(\theta^k, \xi^k, u^k, v^k)\}$ *be the infinite sequence generated by Algorithm 1. Then* $\{(\theta^k, \xi^k, u^k, v^k)\}$ *is bounded,* $\{(\theta^k, \xi^k)\}$ *converges to an optimal solution of* (P), *and* $\{(u^k, v^k)\}$ *converges to an optimal solution of* (D).
(2) Let $r := \sum_{k=0}^{\infty} \varepsilon_k + \mathrm{dist}_{\Sigma}((\theta^0, \xi^0, u^0, v^0), \Lambda)$. *Assume that for this* $r > 0$, *there exists a constant* $\kappa > 0$ *such that* $\mathcal{T}_l$ *satisfies the following error bound condition: for all* (θ, ξ, u, v) *satisfying* $\mathrm{dist}((\theta, \xi, u, v), \Lambda) \leq r$, *it holds that*

$$\mathrm{dist}((\theta, \xi, u, v), \Lambda) \leq \kappa \mathrm{dist}(0, \mathcal{T}_l(\theta, \xi, u, v)). \tag{12}$$

Suppose that $\{(\theta^k, \xi^k, u^k, v^k)\}$ *is the sequence generated by Algorithm 1, where in* ***Step 1****, the approximate solution* $(\theta^{k+1}, \xi^{k+1})$ *also satisfies the stopping criterion*

$$\|\nabla \Phi_k(\theta^{k+1}, \xi^{k+1})\| \leq \frac{\delta_k \sqrt{\lambda_{\min}}}{\sigma_k} \|(\theta^{k+1}, \xi^{k+1}, u^{k+1}, v^{k+1}) - (\theta^k, \xi^k, u^k, v^k)\|_{\Sigma}, \tag{B}$$

and $\{\delta_k \mid 0 \leq \delta_k < 1\}$ *is a given summable sequence. Then it holds for all* $k \geq 0$ *that*

$$\mathrm{dist}_{\Sigma}((\theta^{k+1}, \xi^{k+1}, u^{k+1}, v^{k+1}), \Lambda) \leq \mu_k \mathrm{dist}_{\Sigma}((\theta^k, \xi^k, u^k, v^k), \Lambda), \tag{13}$$

where

$$\mu_k = \frac{1}{1-\delta_k}\left(\delta_k + \frac{(1+\delta_k)\kappa\lambda_{\max}}{\sqrt{\sigma_k^2 + \kappa^2\lambda_{\max}^2}}\right) \to \mu_\infty = \frac{\kappa\lambda_{\max}}{\sqrt{\sigma_\infty^2 + \kappa^2\lambda_{\max}^2}} < 1,$$

and $\lambda_{\max} = \max\{\lambda_{\max}(H_1), \lambda_{\max}(H_2), 1\}$.

As one can see from Theorem 2, the fast linear convergence rate of Algorithm 1 depends on the error bound condition (12) for the maximal monotone operator $\mathcal{T}_l$. For specifying whether the error bound condition (12) holds for different choices of the closed convex set $\mathcal{D}$, we give the following remark.

Remark 1 It is well known that any polyhedral multifunction is upper Lipschitz continuous at every point of its domain according to [39], which means it satisfies the error bound condition (12) for any $r > 0$. For the cases when $\mathcal{D}$ is a polyhedral set, e.g. $\mathcal{D} = \mathbb{R}^d_+ (\mathbb{R}^d_-)$ or $\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_q \leq L\}$ with $q = 1$ or $q = \infty$, $\mathcal{T}_l$ is a polyhedral multifunction, and hence it satisfies the error bound condition (12). In general, one needs addtional assumptions such as partial complementarity for the error bound condition (12) to hold with the presence of nonpolyhedral constraints.

3.2 A semismooth Newton method for solving the proxALM subproblems

One can see that the most computationally intensive step in the proxALM is in solving the subproblem (11). Here we describe how it can be solved efficiently by the semismooth Newton method. For any given $\sigma > 0$, $(\tilde{\theta}, \tilde{\xi}, \tilde{u}, \tilde{v}) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$, we aim to solve the proxALM subproblem, which has the form:

$$\min_{\theta\in\mathbb{R}^n,\xi\in\mathbb{R}^{dn}} \left\{ \Phi(\theta,\xi) := \mathcal{L}_\sigma(\theta,\xi;\tilde{u},\tilde{v}) + \frac{1}{2\sigma}\|\theta-\tilde{\theta}\|_{H_1}^2 + \frac{1}{2\sigma}\|\xi-\tilde{\xi}\|_{H_2}^2 \right\}. \quad (14)$$

Since $\Phi(\cdot,\cdot)$ is strongly convex, the above minimization problem admits a unique solution $(\bar{\theta}, \bar{\xi})$, which can be computed by solving the nonsmooth equation

$$\nabla\Phi(\theta,\xi) = 0, \quad (15)$$

where

$$\nabla\Phi(\theta,\xi) = \begin{pmatrix} \sigma A^T\Big[A\theta + B\xi - \frac{\tilde{u}}{\sigma} - \Pi_+(A\theta + B\xi - \frac{\tilde{u}}{\sigma})\Big] \\ \sigma B^T\Big[A\theta + B\xi - \frac{\tilde{u}}{\sigma} - \Pi_+(A\theta + B\xi - \frac{\tilde{u}}{\sigma})\Big] \end{pmatrix} + \begin{pmatrix} \theta - Y + \frac{1}{\sigma}H_1(\theta-\tilde{\theta}) \\ \sigma\Big[\xi - \frac{\tilde{v}}{\sigma} - \mathrm{Prox}_p(\xi - \frac{\tilde{v}}{\sigma})\Big] + \frac{1}{\sigma}H_2(\xi-\tilde{\xi}) \end{pmatrix}.$$

To apply the SSN to solve the above nonsmooth equation, we need a suitable generalized Jacobian of $\nabla\Phi(\cdot,\cdot)$. Here we choose the following set as the candidate:

$$\hat{\partial}^2\Phi(\theta,\xi) = \sigma\begin{pmatrix} A^T \\ B^T \end{pmatrix}\Big[I_{n^2} - \partial\Pi_+(A\theta + B\xi - \frac{\tilde{u}}{\sigma})\Big]\begin{pmatrix} A & B \end{pmatrix} + \begin{pmatrix} I_n + \frac{1}{\sigma}H_1 & 0 \\ 0 & \sigma\Big[I_{dn} - \partial\mathrm{Prox}_p(\xi - \frac{\tilde{v}}{\sigma})\Big] + \frac{1}{\sigma}H_2 \end{pmatrix},$$

where $\partial\Pi_+$ is the Clarke generalized Jacobian of $\Pi_+(\cdot)$ defined as

$$\partial\Pi_+(\eta) = \left\{ \mathrm{Diag}(q) \,\middle|\, \begin{array}{ll} q_i = 0 & \text{if } \eta_i < 0 \\ q_i \in [0,1] & \text{if } \eta_i = 0 \\ q_i = 1 & \text{otherwise} \end{array} \right\}, \quad \forall \eta \in \mathbb{R}^{n^2}$$

and $\partial\mathrm{Prox}_p$ is the Clarke generalized Jacobian of Prox_p which will be described in Section 4.

We give the following proposition to identify the strong semismoothness of $\nabla\Phi(\cdot,\cdot)$ with respect to $\hat{\partial}^2\Phi(\cdot,\cdot)$, where the definition of strong semismoothness could be found in [32,37,22,43].

Proposition 2 *Suppose that* $\mathrm{Prox}_p(\cdot)$ *is strongly semismooth with respect to the Clarke generalized Jacobian* $\partial\mathrm{Prox}_p(\cdot)$. *Then* $\nabla\Phi(\cdot,\cdot)$ *is strongly semismooth with respect to* $\hat{\partial}^2\Phi(\cdot,\cdot)$.

Proof By the definition of $\partial\Pi_+(\cdot)$, we can see that $\partial\Pi_+(\cdot)$ is nonempty, compact valued, and upper-semicontinuous. Together with the property of $\partial\mathrm{Prox}_p(\cdot)$, it could be seen that the multifunction $\hat{\partial}^2\Phi(\cdot,\cdot)$ is nonempty, compact valued, and upper-semicontinuous.

Note that for any $(\theta,\xi)\in\mathbb{R}^n\times\mathbb{R}^{dn}$, $\nabla\Phi(\cdot,\cdot)$ is directionally differentiable at (θ,ξ). Let $(\Delta\theta,\Delta\xi)\in\mathbb{R}^n\times\mathbb{R}^{dn}$ be such that $\|(\Delta\theta,\Delta\xi)\|$ is sufficiently small. Let $\mathcal{H}\in\hat{\partial}^2\Phi(\theta+\Delta\theta,\xi+\Delta\xi)$, then by definition, there exists $P\in\partial\Pi_+(A(\theta+\Delta\theta)+B(\xi+\Delta\xi)-\tilde{u}/\sigma)$ and $Q\in\partial\mathrm{Prox}_p(\xi+\Delta\xi-\tilde{v}/\sigma)$ such that

$$\mathcal{H}=\sigma\begin{pmatrix}A^T\\B^T\end{pmatrix}\Big(I_{n^2}-P\Big)\begin{pmatrix}A & B\end{pmatrix}+\begin{pmatrix}I_n+\dfrac{1}{\sigma}H_1 & 0\\ 0 & \sigma\Big[I_{dn}-Q\Big]+\dfrac{1}{\sigma}H_2\end{pmatrix}.$$

Since $\Pi_+(\cdot)$ is piecewise affine, we know that

$$\Pi_+(A(\theta+\Delta\theta)+B(\xi+\Delta\xi)-\tilde{u}/\sigma)=\Pi_+(A\theta+B\xi-\tilde{u}/\sigma)+P(\Delta\theta;\Delta\xi).$$

By the strong semismoothness of $\mathrm{Prox}_p(\cdot)$ with respect to $\partial\mathrm{Prox}_p(\cdot)$, we have that

$$\mathrm{Prox}_p(\xi+\Delta\xi-\tilde{v}/\sigma)=\mathrm{Prox}_p(\xi-\tilde{v}/\sigma)+Q\Delta\xi+O(\|\Delta\xi\|^2).$$

Therefore, it holds that

$$\nabla\Phi(\theta+\Delta\theta,\xi+\Delta\xi)-\nabla\Phi(\theta,\xi)-\mathcal{H}(\Delta\theta,\Delta\xi)=O(\|(\Delta\theta,\Delta\xi)\|^2),$$

which means $\nabla\Phi(\cdot,\cdot)$ is strongly semismooth with respect to $\hat{\partial}^2\Phi(\cdot,\cdot)$. □

With the suitably chosen generalized Jacobian $\hat{\partial}^2\Phi(\cdot,\cdot)$, we can design the semismooth Newton method in Algorithm 2, which is a generalization of the standard Newton method, for solving (14).

The convergence analysis for Algorithm 2 can be established as follows.

Theorem 3 *Suppose that* $\mathrm{Prox}_p(\cdot)$ *is strongly semismooth with respect to* $\partial\mathrm{Prox}_p(\cdot)$. *Let* $\{(\theta^j,\xi^j)\}$ *be the infinite sequence generated by Algorithm 2. Then,* $\{(\theta^j,\xi^j)\}$ *converges to the unique optimal solution* $(\bar{\theta},\bar{\xi})$ *of problem* (14), *and*

$$\|(\theta^{j+1},\xi^{j+1})-(\bar{\theta},\bar{\xi})\|=O(\|(\theta^j,\xi^j)-(\bar{\theta},\bar{\xi})\|^{1+\tau}).$$

Proof According to Proposition 2, we have that $\nabla\Phi(\cdot,\cdot)$ is strongly semismooth with respect to $\hat{\partial}^2\Phi(\cdot,\cdot)$. From [48, Proposition 3.3 and Theorem 3.4], we can see that $\{(\theta^j,\xi^j)\}$ converges to the unique optimal solution $(\bar{\theta},\bar{\xi})$. By the formulation of $\hat{\partial}^2\Phi(\cdot,\cdot)$, we have that all the elements in $\hat{\partial}^2\Phi(\theta,\xi)$ are symmetric and positive definite for any $(\theta,\xi)\in\mathbb{R}^n\times\mathbb{R}^{dn}$ due to the positive

Algorithm 2 : Semismooth Newton method for (14)

1: **Initialization:** Given $(\theta^0, \xi^0) \in \mathbb{R}^n \times \mathbb{R}^{dn}$, $\bar{\gamma} \in (0,1)$, $\tau \in (0,1]$, $\delta \in (0,1)$, and $\mu \in (0, 1/2)$. For $j = 0, 1, 2, \ldots$

2: **repeat**

3: **Step 1**. Select an element $\mathcal{H}_j \in \hat{\partial}^2 \Phi(\theta^j, \xi^j)$. Apply a direct method or the preconditioned conjugate gradient (PCG) method to find an approximate solution $(\Delta\theta^j, \Delta\xi^j) \in \mathbb{R}^n \times \mathbb{R}^{dn}$ to

$$\mathcal{H}_j(\Delta\theta^j, \Delta\xi^j) \approx -\nabla\Phi(\theta^j, \xi^j), \tag{16}$$

such that $R_j := \mathcal{H}_j(\Delta\theta^j, \Delta\xi^j) + \nabla\Phi(\theta^j, \xi^j)$ satisfies $\|R_j\| \le \min(\bar{\gamma}, \|\nabla\Phi(\theta^j, \xi^j)\|^{1+\tau})$.

4: **Step 2**. Set $\alpha_j = \delta^{m_j}$, where m_j is the smallest nonnegative integer m such that

$$\Phi(\theta^j + \delta^m \Delta\theta^j, \xi^j + \delta^m \Delta\xi^j) \le \Phi(\theta^j, \xi^j) + \mu\delta^m \langle \nabla\Phi(\theta^j, \xi^j), (\Delta\theta^j, \Delta\xi^j)\rangle.$$

5: **Step 3**. Set $\theta^{j+1} = \theta^j + \alpha_j \Delta\theta^j$, $\xi^{j+1} = \xi^j + \alpha_j \Delta\xi^j$.

6: **until** Stopping criterion (A) or criterion (B) based on θ^{j+1} and ξ^{j+1} is satisfied.

definiteness of H_1 and H_2. Then for sufficiently large j, we have that $\{\|\mathcal{H}_j^{-1}\|\}$ is uniformly bounded from [16, Lemma 7.5.2], and thus

$$\begin{aligned}
&\|(\theta^j, \xi^j) + (\Delta\theta^j, \Delta\xi^j) - (\bar{\theta}, \bar{\xi})\| = \|(\theta^j, \xi^j) - (\bar{\theta}, \bar{\xi}) + \mathcal{H}_j^{-1}(R_j - \nabla\Phi(\theta^j, \xi^j))\| \\
&\le \|\mathcal{H}_j^{-1}\| \Big(\|\nabla\Phi(\theta^j, \xi^j)\|^{1+\tau} + \|\mathcal{H}_j((\theta^j, \xi^j) - (\bar{\theta}, \bar{\xi})) - \nabla\Phi(\theta^j, \xi^j)\| \Big) \\
&= O(\|\nabla\Phi(\theta^j, \xi^j) - \nabla\Phi(\bar{\theta}, \bar{\xi})\|^{1+\tau}) \\
&\quad + O(\|\nabla\Phi(\theta^j, \xi^j) - \nabla\Phi(\bar{\theta}, \bar{\xi}) - \mathcal{H}_j((\theta^j, \xi^j) - (\bar{\theta}, \bar{\xi}))\|) \\
&= O(\|(\theta^j, \xi^j) - (\bar{\theta}, \bar{\xi})\|^{1+\tau}),
\end{aligned} \tag{17}$$

where we have used the strong semismoothness property of $\nabla\Phi(\cdot,\cdot)$ at $(\bar{\theta}, \bar{\xi})$ to get the the last equality. In addition, we could prove that there exists $\hat{\delta} > 0$ such that

$$\langle \nabla\Phi(\theta^j, \xi^j), (\Delta\theta^j, \Delta\xi^j)\rangle \le -\hat{\delta}\|(\Delta\theta^j, \Delta\xi^j)\|^2.$$

Together with [24, Proposition 7] and [16, Proposition 8.3.18], we can derive that for $\mu \in (0, 1/2)$, there exists an integer j_0 such that for all $j \ge j_0$,

$$\Phi(\theta^j + \Delta\theta^j, \xi^j + \Delta\xi^j) \le \Phi(\theta^j, \xi^j) + \mu\langle \nabla\Phi(\theta^j, \xi^j), (\Delta\theta^j, \Delta\xi^j)\rangle,$$

which implies $\theta^{j+1} = \theta^j + \Delta\theta^j$, $\xi^{j+1} = \xi^j + \Delta\xi^j$, for $j \ge j_0$. Combing with (17), we complete the proof. □

Note that in the above theorem, we have proved the Q-superlinear convergence of the sequence $\{(\theta^j, \xi^j)\}$, which implies the R-superlinear convergence of $\{\|\nabla\Phi(\theta^j, \xi^j)\|\}$ due to the fact that

$$\|\nabla\Phi(\theta^j, \xi^j)\| = \|\nabla\Phi(\theta^j, \xi^j) - \nabla\Phi(\bar{\theta}, \bar{\xi})\| = O(\|(\theta^j, \xi^j) - (\bar{\theta}, \bar{\xi})\|).$$

This further implies that condition (A) or condition (B) in Algorithm 2 can be met in a small number of iterations, typically at most dozens of steps.

Remark 2 As a side note, for each closed convex set $\mathcal{D}$ in Proposition 1, we will prove in Proposition 3 that the assumption on $\partial\mathrm{Prox}_p(\cdot)$ in Theorem 3 always holds.

4 Numerical implementation of Algorithm proxALM

In this section, we discuss some numerical details concerning the efficient implementation of the proposed proxALM. For implementing the proxALM, we need the proximal mapping $\mathrm{Prox}_p(\xi)$ for any $\xi \in \mathbb{R}^{dn}$ and its generalized Jacobian. In addition, when evaluating the function value of the problem (D), we need the formula for $p^*(\cdot)$.

4.1 Computation associated with $\mathcal{D}$

For any $\xi = (\xi_1; \cdots; \xi_n) \in \mathbb{R}^{dn}$, since $p(\xi) = \sum_{i=1}^n \delta_{\mathcal{D}}(\xi_i)$, we have that

$$\begin{aligned} p^*(\xi) &= \sum_{i=1}^n \delta^*_{\mathcal{D}}(\xi_i), \quad \mathrm{Prox}_p(\xi) = \begin{pmatrix} \Pi_{\mathcal{D}}(\xi_1) \\ \vdots \\ \Pi_{\mathcal{D}}(\xi_n) \end{pmatrix}, \\ \partial \mathrm{Prox}_p(\xi) &= \begin{pmatrix} \partial \Pi_{\mathcal{D}}(\xi_1) & & \\ & \ddots & \\ & & \partial \Pi_{\mathcal{D}}(\xi_n) \end{pmatrix}, \end{aligned} \tag{18}$$

which means that we only need to focus on $\delta^*_{\mathcal{D}}(\cdot)$, $\Pi_{\mathcal{D}}(\cdot)$ and $\partial \Pi_{\mathcal{D}}(\cdot)$ for each of the set $\mathcal{D}$ defined in Proposition 1. We summarize the results in Table 1 – Table 3, where the detailed derivation associated with the case when $\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_1 \le L\}$ is given in Appendix A.

Table 1 Conjugate function $\delta^*_{\mathcal{D}}(\cdot)$

$\mathcal{D}$	$\delta^*_{\mathcal{D}}(x)$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid x_{K_1} \ge 0, x_{K_2} \le 0\}$	$\delta^*_{\mathcal{D}}(x) = \delta_-(x_{K_1}) + \delta_+(x_{K_2}) + \delta_{\{0\}}(x_{K_3})$ *
$\mathcal{D} = \{x \in \mathbb{R}^d \mid L \le x \le U\}$	$\delta^*_{\mathcal{D}}(x) = \langle U, \max\{x, 0\}\rangle + \langle L, \min\{x, 0\}\rangle$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_q \le L\}$	$\delta^*_{\mathcal{D}}(x) = L\|x\|_p, \quad 1/p + 1/q = 1$

* $K_3 := \{1, \cdots, d\} \backslash (K_1 \cup K_2)$

From the formula of $\partial \Pi_{\mathcal{D}}(\cdot)$ in Table 3, we could see that $\partial \mathrm{Prox}_p(\cdot)$ is a nonempty, compact valued and upper-semicontinuous multifunction. We prove the strong semismoothness of $\mathrm{Prox}_p(\cdot)$ with respect to $\partial \mathrm{Prox}_p(\cdot)$ in the following proposition.

Proposition 3 *For the closed convex set $\mathcal{D}$ defined in Proposition 1,* $\mathrm{Prox}_p(\cdot)$ *is strongly semismooth with respect to* $\partial \mathrm{Prox}_p(\cdot)$.

Proof By the formula of $p(\cdot)$ and the definition of strong semismoothness, it suffices to prove that for each choice of $\mathcal{D}$ defined in Proposition 1, $\Pi_{\mathcal{D}}(\cdot)$ is strongly semismooth with respect to the corresponding Clarke generalized Jacobian $\partial \Pi_{\mathcal{D}}(\cdot)$ defined in Table 3.

Table 2 Proximal mapping $\Pi_{\mathcal{D}}(\cdot)$

$\mathcal{D}$	$\Pi_{\mathcal{D}}(\cdot)$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid x_{K_1} \geq 0, x_{K_2} \leq 0\}$	$(\Pi_{\mathcal{D}}(x))_i = \begin{cases} 0 & \text{if } i \in K_1, x_i < 0, \text{ or } i \in K_2, x_i > 0 \\ x_i & \text{otherwise} \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid L \leq x \leq U\}$	$(\Pi_{\mathcal{D}}(x))_i = \begin{cases} x_i & \text{if } L_i \leq x_i \leq U_i \\ 0 & \text{otherwise} \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_\infty \leq L\}$	$(\Pi_{\mathcal{D}}(x))_i = \begin{cases} x_i & \text{if } \lvert x_i\rvert \leq L \\ \text{sign}(x_i)L & \text{if } \lvert x_i\rvert > L \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq L\}$	$\Pi_{\mathcal{D}}(x) = \begin{cases} x & \text{if } \|x\|_2 \leq L \\ L\dfrac{x}{\|x\|_2} & \text{otherwise} \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_1 \leq L\}$	$\Pi_{\mathcal{D}}(x) = \begin{cases} x & \text{if } \|x\|_1 \leq L \\ LP_x\Pi_{\Delta_d}(P_x x/L)^* & \text{otherwise} \end{cases}$

* $P_x = \text{Diag}(\text{sign}(x)) \in \mathbb{R}^{d\times d}$, $\Pi_{\Delta_d}(\cdot)$ denotes the projection onto the simplex $\Delta_d = \{x \in \mathbb{R}^d \mid e_d^T x = 1, x \geq 0\}$, which can be computed in $O(d\log(d))$ operations.

Table 3 Generalized Jacobian of $\Pi_{\mathcal{D}}(\cdot)$

$\mathcal{D}$	$\partial\Pi_{\mathcal{D}}(\cdot)$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid x_{K_1} \geq 0, x_{K_2} \leq 0\}$	$\partial\Pi_{\mathcal{D}}(x) = \text{Diag}(u),\ u_i \in \begin{cases} \{0\} & \text{if } i \in K_1, x_i < 0, \text{ or } i \in K_2, x_i > 0 \\ [0,1] & \text{if } i \in K_1 \cup K_2, x_i = 0 \\ \{1\} & \text{otherwise} \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid L \leq x \leq U\}$	$\partial\Pi_{\mathcal{D}}(x) = \text{Diag}(u),\ u_i \in \begin{cases} \{1\} & \text{if } L_i < x_i < U_i \\ [0,1] & \text{if } x_i = L_i \text{ or } x_i = U_i \\ \{0\} & \text{otherwise} \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_\infty \leq L\}$	$\partial\Pi_{\mathcal{D}}(x) = \text{Diag}(u),\ u_i \in \begin{cases} \{1\} & \text{if } \lvert x_i\rvert < L \\ [0,1] & \text{if } \lvert x_i\rvert = L \\ \{0\} & \text{otherwise} \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq L\}$	$\partial\Pi_{\mathcal{D}}(x) = \begin{cases} \{I_d\} & \text{if } \|x\|_2 < L \\ \left\{I_d - t\dfrac{xx^T}{L^2} \mid 0 \leq t \leq 1\right\} & \text{if } \|x\|_2 = L \\ \left\{\dfrac{L}{\|x\|_2}(I_d - \dfrac{xx^T}{\|x\|_2^2})\right\} & \text{otherwise} \end{cases}$
$\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_1 \leq L\}$	$H \in \partial\Pi_{\mathcal{D}}(x), \text{ where } H = \begin{cases} I_d & \text{if } \|x\|_1 \leq L \\ P_x\widetilde{H}P_x{}^* & \text{otherwise} \end{cases}$

* $\widetilde{H} = \text{Diag}(r) - \frac{1}{\text{nnz}(r)}rr^T \in \partial\Pi_{\Delta_d}(x)$, where $r \in \mathbb{R}^d$ is defined as $r_i = 1$ if $\left(\Pi_{\Delta_d}(P_x x/L)\right)_i \neq 0$, and $r_i = 0$ otherwise.

For the case when $\mathcal{D} = \{x \in \mathbb{R}^d \mid x_{K_1} \geq 0, x_{K_2} \leq 0\}$, $\mathcal{D} = \{x \in \mathbb{R}^d \mid L \leq x \leq U\}$ or $\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_\infty \leq L\}$, we can see that $\Pi_{\mathcal{D}}(\cdot)$ is a Lipschitz continuous piecewise affine function, and thus $\Pi_{\mathcal{D}}(\cdot)$ is strongly semismooth everywhere with respect to the corresponding Clarke generalized Jacobian $\partial\Pi_{\mathcal{D}}(\cdot)$ defined in Table 3 due to [16, Proposition 7.4.7]. For the case when $\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq L\}$, the strong semismoothness of $\Pi_{\mathcal{D}}(\cdot)$ with respect to $\partial\Pi_{\mathcal{D}}(\cdot)$ follows from the fact that the projection onto the second

order cone is strongly semismooth [13, Proposition 4.3]. When $\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_1 \leq L\}$, $\Pi_{\mathcal{D}}(\cdot)$ is strongly semismooth with respect to the corresponding $\partial\Pi_{\mathcal{D}}(\cdot)$ in Table 3, which is the so-called HS-Jacobian [17,26]. □

4.2 Finding a computable element in $\hat{\partial}^2\Phi(\theta,\xi)$

As already mentioned, the most difficult part of the proxALM is in solving the Newton system (16). For efficient practical implementation, we need to find an efficiently computable element in $\hat{\partial}^2\Phi(\theta,\xi)$ for any given $(\theta,\xi) \in \mathbb{R}^n \times \mathbb{R}^{dn}$. From the definition of $\hat{\partial}^2\Phi(\theta,\xi)$, we can rewrite it as

$$\hat{\partial}^2\Phi(\theta,\xi) = \mathcal{M}_1(\theta,\xi) + \mathcal{M}_2(\xi),$$

where

$$\mathcal{M}_1(\theta,\xi) = \sigma \begin{pmatrix} A^T \\ B^T \end{pmatrix} \left(I_{n^2} - \partial\Pi_+(A\theta + B\xi - \frac{\tilde{u}}{\sigma}) \right) \begin{pmatrix} A & B \end{pmatrix},$$

$$\mathcal{M}_2(\xi) = \begin{pmatrix} I_n + \frac{1}{\sigma}H_1 & 0 \\ 0 & \sigma\left[I_{dn} - \partial\mathrm{Prox}_p(\xi - \frac{\tilde{v}}{\sigma}) \right] + \frac{1}{\sigma}H_2 \end{pmatrix}.$$

Based on our discussion on $\partial\mathrm{Prox}_p(\cdot)$ in (18) and $\partial\Pi_{\mathcal{D}}(\cdot)$ in Table 3, we can see that the elements in $\partial\mathrm{Prox}_p(\cdot)$ are block diagonal matrices. In order to maintain the block diagonal structure, we choose H_1 and H_2 to be diagonal matrices, and hence the elements in $\mathcal{M}_2(\xi)$ for any $\xi \in \mathbb{R}^{dn}$ will also be block diagonal matrices. One can easily pick an element in $\mathcal{M}_2(\xi)$ by choosing an element in $\partial\mathrm{Prox}_p(\xi - \tilde{v}/\sigma)$.

For $\mathcal{M}_1(\theta,\xi)$, we choose an element $\mathrm{Diag}(w)$ in $\partial\Pi_+(A\theta + B\xi - \tilde{u}/\sigma)$, where

$$w_i = \begin{cases} 1 & \text{if } (A\theta + B\xi - \frac{\tilde{u}}{\sigma})_i \geq 0 \\ 0 & \text{otherwise} \end{cases}, \quad i = 1, \cdots, n^2.$$

By denoting $\bar{w} \in \mathbb{R}^{n^2}$ as $\bar{w}_i = 1 - w_i$ for $i = 1, \cdots, n^2$, then

$$M = \sigma \begin{pmatrix} A^T \\ B^T \end{pmatrix} \mathrm{Diag}(\bar{w}) \begin{pmatrix} A & B \end{pmatrix} = \sigma \begin{pmatrix} A^T\mathrm{Diag}(\bar{w})A & A^T\mathrm{Diag}(\bar{w})B \\ B^T\mathrm{Diag}(\bar{w})A & B^T\mathrm{Diag}(\bar{w})B \end{pmatrix}$$

is an element in $\mathcal{M}_1(\theta,\xi)$. After some algebraic manipulations by making use of the structure of A and B, we can prove the following results:

$$\begin{aligned}
A^T\mathrm{Diag}(\bar{w})A &= \mathrm{Diag}\left(\sum_{i=1}^{n}\bar{w}_{(i)}\right) + \mathrm{Diag}\left(\bar{w}_{(1)}^T\bar{w}_{(1)},\cdots,\bar{w}_{(n)}^T\bar{w}_{(n)}\right)\\
&\quad - (\bar{w}_{(1)},\cdots,\bar{w}_{(n)}) - (\bar{w}_{(1)}^T;\cdots;\bar{w}_{(n)}^T) \in \mathbb{R}^{n\times n},\\
A^T\mathrm{Diag}(\bar{w})B &= \left(\mathrm{Diag}(\bar{w}_{(1)})B_1,\cdots,\mathrm{Diag}(\bar{w}_{(n)})B_n\right)\\
&\quad - \mathrm{Diag}\left(\bar{w}_{(1)}^TB_1,\cdots,\bar{w}_{(n)}^TB_n\right) \in \mathbb{R}^{n\times dn},\\
B^T\mathrm{Diag}(\bar{w})B &= \mathrm{Diag}\left(B_1^T\mathrm{Diag}(\bar{w}_{(1)})B_1,\cdots,B_n^T\mathrm{Diag}(\bar{w}_{(n)})B_n\right) \in \mathbb{R}^{dn\times dn},
\end{aligned}$$

where $\bar{w}_{(i)} := \bar{w}_{(i-1)n+1:in} \in \mathbb{R}^n$. It can be seen that the 0-1 structure of $\bar{w}$ will reduce many operations in matrix-matrix multiplications, and hence highly reduce the computational cost for computing M or matrix-vector products with M. Note that for all i, $\mathrm{Diag}(\bar{w}_{(i)})B_i$ is a matrix in $\mathbb{R}^{n\times d}$, with its j-th row being the j-th row of B_i if $(\bar{w}_{(i)})_j = 1$, or the zero vector if $(\bar{w}_{(i)})_j = 0$. Then the computation of $\bar{w}_{(i)}^TB_i$ can be obtained by summing the non-zero rows of $\mathrm{Diag}(\bar{w}_{(i)})B_i$, and the computation of $B_i^T\mathrm{Diag}(\bar{w}_{(i)})B_i = (\mathrm{Diag}(\bar{w}_{(i)})B_i)^T(\mathrm{Diag}(\bar{w}_{(i)})B_i)$ can be highly reduced in the same way.

The special structure of the elements in $\hat{\partial}^2\Phi(\theta,\xi)$, which we call as the second-order sparsity, makes it possible for us to apply the `SSN` based `proxALM` algorithm to solve the huge QP problem (P) that contains $n(d+1)$ variables, $n(n-1)$ linear inequality constraints and n possibly non-polyhedral constraints.

5 A constraint generation method to accelerate the computation

Due to the existence of $n(n-1)$ linear inequality constraints, the problem (P) is quite difficult to solve for the case when the number of observations n is huge. This naturally motivated us to consider a constraint generation method to avoid handling the full set of constraints when solving the problem. In this section, we design a practical implementation of the constraint generation method for solving the problem (P) with large n, where each reduced problem is solved by the `proxALM`.

The basic idea of the constraint generation method is to start solving the constrained QP with a subset of constraints, then add the most violated constraints (or part of violated constraints) to form a new reduced problem until the optimality conditions are satisfied. In our implementation, there are three points that we should emphasize. First, we add a relatively large number of most violated constraints in each round to highly reduce the number of rounds needed for the constraint generation method to terminate. Second, we apply our proposed `proxALM` to solve each reduced problem to high accuracy, which is demonstrated to be quite efficient for solving large-scale problems. Third, we

divide the $O(n^2)$ constraints into blocks and check the optimality conditions block-wise to cope with the memory demand.

Suppose that $(\theta^*, \xi^*, u^*, v^*) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$ is a KKT solution of the problems (P) and (D). Note that in the problem (P), the condition

$$A\theta + B\xi \geq 0$$

imposes n^2 linear inequality constraints on $n(d+1)$ variables. For the case when $n \gg d$, no more than $n(d+1)$ independent constraints would be active at (θ^*, ξ^*). That is to say, there exists an index set $I^* \subset \{1, 2, \cdots, n^2\}$ with $|I^*| \leq n(d+1)$ such that

$$(A\theta^* + B\xi^*)_{I^*} = 0, \quad (A\theta^* + B\xi^*)_{\bar{I}^*} \geq 0,$$

where $\bar{I}^*$ denotes the complement of I^* in $\{1, 2, \cdots, n^2\}$. The small proportion of active constraints inspires us to apply the idea of the constraint generation as an acceleration technique to solve the problems with large n.

Given an index set $I \subset \{1, 2, \cdots, n^2\}$, we consider a variant of the problem (P) as

$$\min_{\theta \in \mathbb{R}^n, \xi \in \mathbb{R}^{dn}} \ \Big\{ \frac{1}{2}\|\theta - Y\|^2 + p(\xi) + \delta_+(A_I\theta + B_I\xi) \Big\}, \tag{19}$$

where A_I denotes the matrix consisting of the rows of A indexed by I. The corresponding dual problem is

$$\begin{aligned} \max_{u \in \mathbb{R}^{n^2}, v \in \mathbb{R}^{dn}} \ & \Big\{ -\frac{1}{2}\|A^T u\|^2 - \langle Y, A^T u \rangle - p^*(-v) - \delta_+(u) \Big\} \\ \text{s.t.} \quad & B^T u + v = 0, \quad u_{\bar{I}} = 0. \end{aligned} \tag{20}$$

The KKT system associated with the problems (19) and (20) is

$$\begin{aligned} & \theta - Y - A^T u = 0, \quad B^T u + v = 0, \quad u_{\bar{I}} = 0, \\ & -v \in \partial p(\xi), \quad -u_I \in \partial \delta_+(A_I\theta + B_I\xi). \end{aligned} \tag{21}$$

Suppose that $(\bar{\theta}, \bar{\xi}, \bar{u}, \bar{v}) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$ satisfies the KKT system (21). We could see that $(\bar{\theta}, \bar{\xi}, \bar{u}, \bar{v})$ naturally satisfies the KKT system (10) associated with the problems (P) and (D), except for the following inequality

$$A_{\bar{I}}\theta + B_{\bar{I}}\xi \geq 0.$$

Therefore, we add the indices in the index set $I' := \{i \in \bar{I} \mid A_{\{i\}}\bar{\theta} + B_{\{i\}}\bar{\xi} < 0\}$ into I to get a new variant of the problem (P) as stated in (19), then repeat the procedure until the stopping criteria of the problems (P) and (D) are satisfied.

Note that in this paper, we use the relative KKT residual

$$\begin{aligned} R_{\mathrm{KKT}} := \max \Big\{ & \frac{\|\theta - Y - A^T u\|}{1 + \|Y\| + \|\theta\| + \|u\|}, \frac{\|B^T u + v\|}{1 + \|u\| + \|v\|}, \frac{\|\xi - \mathrm{Prox}_p(\xi - v)\|}{1 + \|\xi\| + \|v\|}, \\ & \frac{\|A\theta + B\xi - \Pi_+(A\theta + B\xi - u)\|}{1 + \|A\theta\| + \|B\xi\| + \|u\|} \Big\} \leq \epsilon, \end{aligned} \tag{22}$$

where $\epsilon > 0$ is a given tolerance, to measure the accuracy of an approximate optimal solution (θ, ξ, u, v) to the KKT system (10). In addition, given an index set $I \subset \{1, 2, \cdots, n^2\}$, we define

$$R^I_{\mathrm{KKT}} = \max\Big\{ \frac{\|\theta - Y - A_I^T u_I\|}{1 + \|Y\| + \|\theta\| + \|u_I\|}, \frac{\|B_I^T u_I + v\|}{1 + \|u_I\| + \|v\|}, \frac{\|\xi - \mathrm{Prox}_p(\xi - v)\|}{1 + \|\xi\| + \|v\|}, \frac{\|A_I\theta + B_I\xi - \Pi_+(A_I\theta + B_I\xi - u_I)\|}{1 + \|A_I\theta\| + \|B_I\xi\| + \|u_I\|} \Big\}.$$

Next we present our practical implementation of the constraint generation method for solving the problem (P) in Algorithm 3, where we apply our proposed `proxALM` to solve each of the reduced problems.

Algorithm 3 : Constraint generation method for (P)

1: **Initialization:** Given a tolerance $\epsilon > 0$ and an initial index set $I^0 \subset \{1, 2, \cdots, n^2\}$, solve the problem

$$\min_{\theta \in \mathbb{R}^n, \xi \in \mathbb{R}^{dn}} \Big\{ \frac{1}{2}\|\theta - Y\|^2 + p(\xi) + \delta_+(A_{I_0}\theta + B_{I_0}\xi) \Big\} \tag{P_{I^0}}$$

to get an approximate KKT solution $(\theta^0, \xi^0, u^0, v^0) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$ such that $u^0_{\bar{I}^0} = 0$ and $R^{I_0}_{\mathrm{KKT}} \le \epsilon$. Compute R_{KKT} and set $k = 1$.

2: **repeat**

3: **Step 1**. Let

$$S^k := \{j \in \bar{I}^{k-1} \mid A_{\{j\}}\theta^{k-1} + B_{\{j\}}\xi^{k-1} < 0\}.$$

If $|S^k| > |I^{k-1}|$, set

$$I^k = I^{k-1} \cup \left\{ j \in \bar{I}^{k-1} \,\middle|\, \begin{array}{l} A_{\{j\}}\theta^{k-1} + B_{\{j\}}\xi^{k-1} \text{ is among the first } |I^{k-1}| \\ \text{smallest values in } A_{S^k}\theta^{k-1} + B_{S^k}\xi^{k-1} \end{array} \right\};$$

and otherwise, set $I^k = I^{k-1} \cup S^k$.

4: **Step 2**. Solve the problem

$$\min_{\theta \in \mathbb{R}^n, \xi \in \mathbb{R}^{dn}} \Big\{ \frac{1}{2}\|\theta - Y\|^2 + p(\xi) + \delta_+(A_{I_k}\theta + B_{I_k}\xi) \Big\} \tag{P_{I^k}}$$

to get an approximate KKT solution $(\theta^k, \xi^k, u^k, v^k) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$ such that $u^k_{\bar{I}^k} = 0$ and $R^{I_k}_{\mathrm{KKT}} \le \epsilon$.

5: **Step 3**. Compute R_{KKT} and set $k \leftarrow k + 1$.

6: **until** Stopping criteria $R_{\mathrm{KKT}} \le \epsilon$ is satisfied.

Remark 3 As a side note, in the kth iteration of Algorithm 3, we apply a warm start technique by setting the initialization as the solution obtained in the $(k-1)$th iteration.

The convergence property of Algorithm 3 is presented in the following theorem.

Theorem 4 *For any given tolerance ϵ and any initial index set $I^0 \subset \{1, 2, \cdots, n^2\}$, Algorithm 3 will terminate after a finite number of rounds.*

Proof We first prove that if $S^{k+1} = \emptyset$, the corresponding $(\theta^k, \xi^k, u^k, v^k) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$ satisfies $R_{\mathrm{KKT}} \leq \epsilon$. Suppose $S^{k+1} = \emptyset$, then we have

$$A_{\bar{I}^k}\theta^k + B_{\bar{I}^k}\xi^k \geq 0.$$

Together with $u^k_{\bar{I}^k} = 0$, we know that

$$\begin{aligned}
\frac{\|\theta^k - Y - A^T u^k\|}{1 + \|Y\| + \|\theta^k\| + \|u^k\|} &= \frac{\|\theta^k - Y - A^T_{I^k} u^k_{I^k}\|}{1 + \|Y\| + \|\theta^k\| + \|u^k\|}, \\
\frac{\|B^T u^k + v^k\|}{1 + \|u^k\| + \|v^k\|} &= \frac{\|B^T_{I^k} u^k_{I^k} + v^k\|}{1 + \|u^k_{I^k}\| + \|v^k\|}, \\
\frac{\|A\theta^k + B\xi^k - \Pi_+(A\theta^k + B\xi^k - u^k)\|}{1 + \|A\theta^k\| + \|B\xi^k\| + \|u^k\|} &\leq \frac{\|A_{I^k}\theta^k + B_{I^k}\xi^k - \Pi_+(A_{I^k}\theta^k + B_{I^k}\xi^k - u^k_{I^k})\|}{1 + \|A_{I^k}\theta^k\| + \|B_{I^k}\xi^k\| + \|u^k_{I^k}\|}.
\end{aligned}$$

Combining with the fact that $(\theta^k, \xi^k, u^k, v^k)$ satisfies $R^{I_k}_{\mathrm{KKT}} \leq \epsilon$, we have the corresponding relative KKT residual $R_{\mathrm{KKT}} \leq R^{I_k}_{\mathrm{KKT}} \leq \epsilon$. As a result, if $R_{\mathrm{KKT}} > \epsilon$, we have $S^{k+1} \neq \emptyset$, which means that new constraints will be added to construct a new reduced primal problem. Since the total number of the constraints in the primal problem (P) is finite, our algorithm will terminate after a finite number of rounds. □

Note that in the algorithm, we add a relatively large number of violated constraints instead of adding n violated constraints in each round as done in [6, 12]. The reason is that we have a highly efficient `proxALM` algorithm which can solve each reduced problem (P_{I^k}) with a relatively large number of constraints. The superior performance of this acceleration technique will be demonstrated in the numerical experiments.

6 Numerical experiments

In this section, we conduct some numerical experiments to demonstrate the performance of the `proxALM` for solving (P), under each case of $\mathcal{D}$ mentioned in Proposition 1, as well as the performance of the constraint generation method for the acceleration. In addition, we design a data-driven Lipschitz estimation method to deal with the boundary effect of the convex regression problem. All our computational results are obtained by running MATLAB R2018b on a windows workstation (12-core, Intel Xeon E5-2680 @ 2.50GHz, 128 G RAM).

6.1 Computational performance of the proxALM for solving (P)

In this subsection, we compare the performance of the proxALM, the sGS-ADMM, and MOSEK for different choices of d and n. In the experiments, we stop the algorithm when $R_{\rm KKT} \leq 10^{-4}$, where $R_{\rm KKT}$ is defined in (22). In Algorithm proxALM, we choose $H_1 = 10^{-3}I_n$, $H_2 = 10^{-3}I_{dn}$, and use the stopping criteria (B) in **Step 1** with $\delta_k = \max\{0.1, 10^{-6}/\|(\theta^{k+1}, \xi^{k+1}, u^{k+1}, v^{k+1}) - (\theta^k, \xi^k, u^k, v^k)\|_\Sigma\}/(\lceil k/20\rceil^2)$. Here, the sGS-ADMM is a symmetric Gauss-Seidel based multi-block ADMM, which is proved to be convergent and has been demonstrated to perform better than the possibly nonconvergent directly extended multi-block ADMM [11]. The detailed description of the sGS-ADMM could be found in Appendix B. As we can see in [3], as long as there is enough memory, MOSEK can perform quite a lot better than the parallel APG method. Since there is enough memory on our workstation, we just compare our proposed proxALM with the state-of-the-art algorithms MOSEK and sGS-ADMM.

For a given convex function $\psi : \mathbb{R}^d \to \mathbb{R}$, the synthetic dataset is generated via the procedure in [30]. We first generate n samples $X_i \in \mathbb{R}^d$, $i = 1, \cdots, n$ uniformly from $[-1, 1]^d$, then the corresponding responses are given as $Y_i = \psi(X_i) + \epsilon_i$. The error vector ϵ follows the normal distribution $\mathcal{N}(0, \sigma^2 I_n)$, where $\sigma^2 = {\rm Var}(\{\psi(X_i)\}_{i=1}^n)/{\rm SNR}$. In the experiments, we take ${\rm SNR} = 3$. Before we run the algorithms for the data $X = (X_1, \cdots, X_n) \in \mathbb{R}^{d\times n}$ and $Y \in \mathbb{R}^n$, we process the data so as to build a more predictive model. For the response Y and each row of the predictor X, we mean-center the vector and then standardize it to have unit ℓ_2-norm.

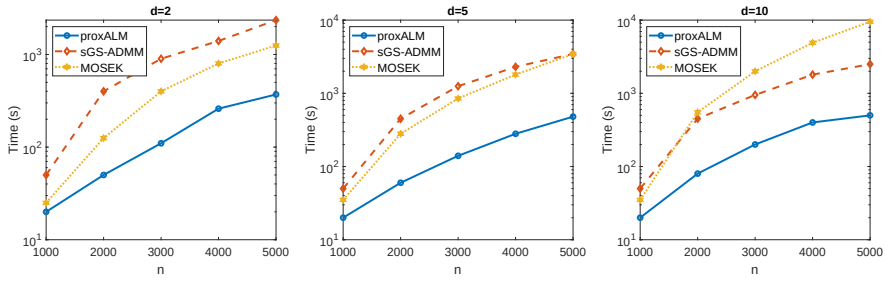


Fig. 1 Convex regression for test function $\psi(x) = \exp(p^T x)$, where p is a given random vector with each coordinate drawn from the standard normal distribution

The numerical results on the comparison among proxALM, sGS-ADMM and MOSEK can be found in Figure 1 – Figure 6. Note that we set the y-axes of all figures in log-scale to better show the functional dependence on n. We conduct experiments on the unconstrained convex regression problem and each case of shape-constrained convex regression we mentioned before, under different choices of (d, n). All the test functions are convex on $\mathbb{R}^d$ and satisfy some specified shape constraints. As one can see from the figures, proxALM outperforms the state-of-the-art solvers MOSEK and sGS-ADMM by a large mar-

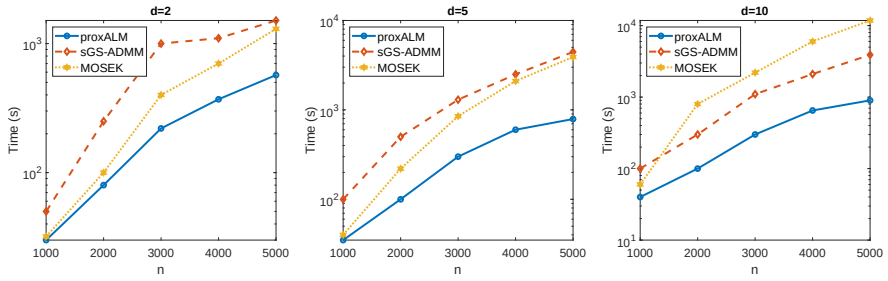


Fig. 2 Convex regression with monotone constraint (non-decreasing) for the test function $\psi(x) = (e_d^T x)_+$

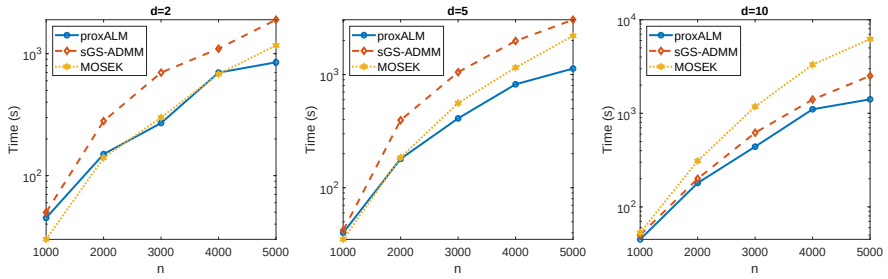


Fig. 3 Convex regression with box constraint ($L = 0_d$, $U = e_d$) for the test function $\psi(x) = \ln(1 + \exp(e_d^T x))$

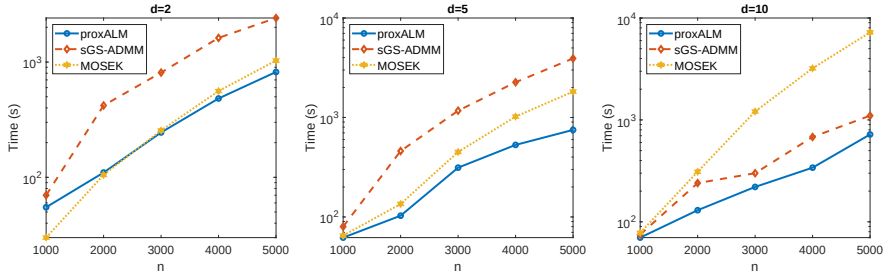


Fig. 4 Convex regression with Lipschitz constraint ($p = 1$, $q = \infty$, $L = 1$) for the test function $\psi(x) = \sqrt{1 + x^T x}$

gin, especially for large-scale cases. For example, for the convex regression with monotone constraint when $(d, n) = (5, 5000)$, the `proxALM` takes about 800 seconds, while `sGS-ADMM` and `MOSEK` take around 4000 seconds.

More numerical results of the comparison on instances with larger d could be found in Appendix C.

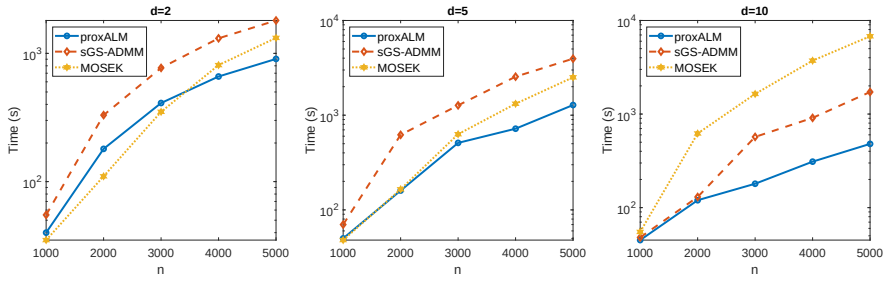


Fig. 5 Convex regression with Lipschitz constraint ($p = 2$, $q = 2$, $L = \lambda_{\max}(Q)$) for the test function $\psi(x) = \sqrt{x^T Q x}$, where $Q \in \mathbb{R}^{d\times d}$ is a randomly generated symmetric and positive definite matrix with known largest eigenvalue

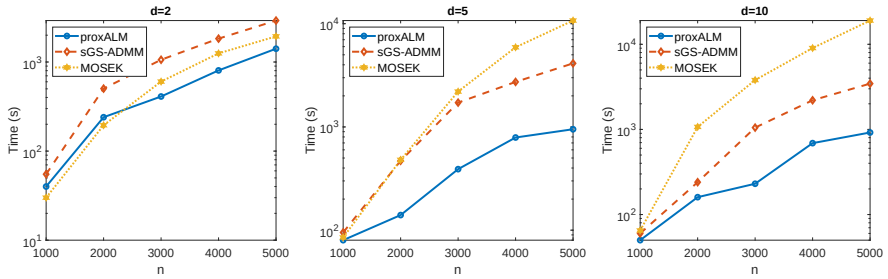


Fig. 6 Convex regression with Lipschitz constraint ($p = \infty$, $q = 1$, $L = 1$) for the test function $\psi(x) = \ln(1 + e^{x_1} + \cdots + e^{x_d})$

6.2 Computational performance of the acceleration with the constraint generation method

In this subsection, we mainly focus on the case when $n \gg d$. Consider the convex function $\psi(x) = 5\|x\|_\infty + \|x\|^2$, we sample n data points uniformly from $[-1, 1]^d$ and add the Gaussian noise as stated in Section 6.1 with SNR= 10. Figure 7 shows the time comparison among `CGM+proxALM`, `CGM+sGS-ADMM`, `CGM+MOSEK`, `proxALM`, `sGS-ADMM`, `MOSEK`, where `CGM+` means the constraint generation method is used for the acceleration, to solve the convex regression problems with $d = 2$. Note that in the `CGM`, we take $|I^0| = 10n$ and select the initial indices uniformly at random from the set $\{1, 2, \cdots, n^2\}$. We stop each algorithm when $R_{\mathrm{KKT}} \leq 10^{-4}$.

From the result, we can see that `CGM+proxALM` outperforms all other algorithms by quite a large margin. For example, for the case $(d, n) = (2, 5000)$, `CGM+proxALM` takes 28 seconds, `proxALM` takes 288 seconds, while the remaining four algorithms take around 1000 seconds.

To further demonstrate the performance of the `CGM` with the `proxALM`, we conduct experiments on examples with higher dimensions and larger sample sizes. The results are shown in Table 4. In Algorithm `CGM`, we set $|I^0| = 50n$ for $d = 2$, and $|I^0| = 10n$ for $d = 10, 20$. In consideration of memory cost, we divide

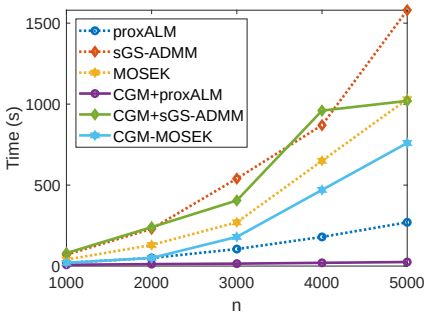


Fig. 7 Time comparison among `CGM+proxALM`, `CGM+sGS-ADMM`, `CGM+MOSEK`, `proxALM`, `sGS-ADMM`, `MOSEK`

the n^2 constraints into ten parts when checking the optimality conditions (10) and when selecting the new indices in **Step 1** of Algorithm `CGM`.

Table 4 Performance of `CGM+proxALM` on convex regression on instances with large sample sizes. Time is divides into three parts: `CGM` (constraint generation step), `proxALM` (running time of the `proxALM`), and OPT (checking optimality conditions)

(d,n)	`CGM` rounds	R_{KKT}	R_{gap}	$\hat{R}_{\text{viotol}}$	$\hat{R}_{\text{pinfeas}}$	Time(s)(`CGM`| `proxALM`| OPT)
$(2, 10000)$	3	1.39e-5	1.54e-4	1.88e-3	5.42e-6	36(2| 31| 3)
$(2, 50000)$	3	9.25e-5	2.01e-3	1.87e-3	1.61e-5	281(46| 172| 63)
$(2, 100000)$	4	6.87e-5	5.11e-3	6.16e-4	4.28e-6	1270(194| 740| 336)
$(10, 10000)$	4	1.58e-5	1.19e-3	1.30e-2	9.13e-6	27(3| 17| 7)
$(10, 50000)$	4	8.87e-5	9.23e-3	1.71e-2	2.24e-5	334(67| 181| 86)
$(10, 100000)$	5	2.42e-5	9.91e-3	7.93e-3	2.65e-6	1625(328| 855| 442)
$(20, 10000)$	3	6.57e-5	1.89e-3	1.17e-2	2.88e-5	45(3| 37| 5)
$(20, 50000)$	4	8.04e-5	5.59e-3	5.14e-3	1.57e-5	425(73| 265| 87)
$(20, 100000)$	5	3.88e-5	7.90e-3	1.31e-3	3.16e-7	1614(331| 836| 447)

Note that in the table, the number of `CGM` rounds includes the initialization step, and R_{rel} is defined as

$$R_{\text{rel}} = \frac{|\text{pobj} - \text{dobj}|}{1 + |\text{pobj}| + |\text{dobj}|},$$

where pobj and dobj denote the primal and dual objective function values. For better illustration, we also report the *primal infeasibility* [30,6] and the *violation tolerance* [6] as

$$\hat{R}_{\text{pinfeas}} = \frac{1}{n}\|(A\theta + B\xi)_-\|, \quad \hat{R}_{\text{viotol}} = \max|(A\theta + B\xi)_-|,$$

respectively, where $x_- := \min(x, 0)$.

We can see from the table that the `CGM` combined with the `proxALM` performs quite well for estimating the convex regression functions with huge sample sizes. Note that in the table, time is divided into three parts: constraint

generation step, running time of the proxALM and checking optimality conditions. As the sample size of the instance increases, the time taken by the constraint generation step and checking optimality conditions increases rapidly due to the huge number of n^2 linear inequality constraints. For example, for the instance with size $(d, n) = (10, 100000)$, we need to solve a constrained QP containing 1.1×10^6 variables and 10^{10} linear inequality constraints. From the table we can see that checking the optimality conditions five times cost 442 seconds while estimating the convex regression function with CGM+proxALM only costs 1625 seconds. The long computation time needed to check the optimality conditions for large n is the reason why we choose to add more violated constraints in each round so as to reduce the number of rounds in the constraint generation method. As a comparison, we note that the implementation in [6] of the constraint generation method with each reduced problem solved by Gurobi needs around 1 hour and 11 rounds of the constraint generation to solve the problem of the same size, but only achieves the accuracy $\hat{R}_{\text{viotol}} = 0.05$, $\hat{R}_{\text{pinfeas}} = 0.004$. The success of the proposed CGM combined with the proxALM lies in two aspects. First, the number of rounds of the constraint generation is highly reduced since we add a relatively large number of violated constraints in each round. Second, the proxALM is quite efficient to solve each reduced problem in the CGM compared to Gurobi or MOSEK.

6.3 Data-driven Lipschitz estimation method

An important issue in convex regression is over-fitting near the boundary of $\text{conv}(X_1, \cdots, X_n)$. That is, the norms of the fitted subgradients ξ_i's near the boundary can become arbitrarily large. The authors in [27,4,30] used the idea of Lipschitz convex regression to deal with this problem. They propose to compute the least squares estimator over the class of convex functions that are uniformly Lipschitz with a given bound, which means that they compute the estimator defined in (2) with Property $\mathcal{S}$ taking the form of (S3). In practice, the challenge is in choosing the unknown Lipschitz constant in the model based on the given data. Mazumder et al. [30] choose to estimate the Lipschitz constant by using the cross-validation. In this paper, we provide a data-driven Lipschitz estimation method for the Lipschitz convex regression.

For each X_i, we first find the k-nearest neighbors $\mathcal{N}(X_i)$ of X_i, and then define

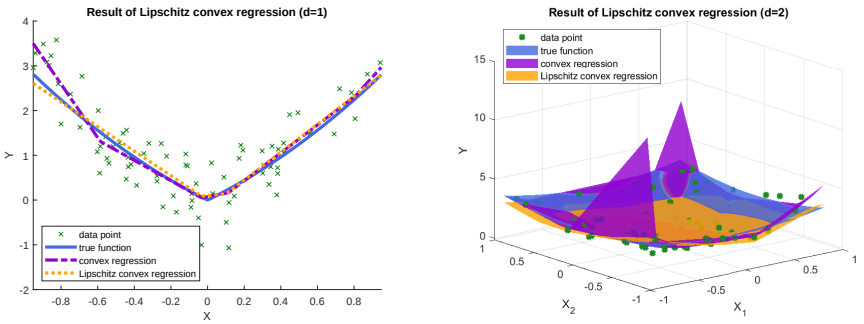
$$L_i = \text{median}\Big\{ \frac{|Y_i - Y_j|}{\|X_i - X_j\|_p}, j \in \mathcal{N}(X_i) \Big\},$$

where $p = 1, 2, \infty$ is given. Then we solve the generalization form of (7) as

$$\begin{aligned} \min_{\theta_1,\ldots,\theta_n \in \mathbb{R}; \xi_1,\ldots,\xi_n \in \mathbb{R}^d} \ & \frac{1}{2} \sum_{i=1}^{n} (\theta_i - Y_i)^2 \\ \text{s.t.} \quad & \theta_i \geq \theta_j + \langle \xi_j, X_i - X_j \rangle, \quad \forall\ 1 \leq i, j \leq n, \\ & \xi_i \in \mathcal{D}_i, \quad i = 1, \cdots, n, \end{aligned} \tag{23}$$

where $\mathcal{D}_i = \{x \in \mathbb{R}^d \mid \|x\|_q \leq L_i\}$ with $1/p + 1/q = 1$. The proposed `proxALM` can be easily extended to solve (23) by letting $p(\xi) = \sum_{i=1}^n \delta_{\mathcal{D}_i}(\xi_i)$.

We use an example here to demonstrate the performance of Lipschitz convex regression with the data-driven Lipschitz estimation method. Consider the convex function $\psi(x) = 2\|x\|_\infty + \|x\|^2$, we sample $n = 80$ data points uniformly from $[-1, 1]^d$ and add the Gaussian noise as stated in Section 6.1. The results for $d = 1, 2$ can be seen in Figure 8. When estimating the Lipschitz constant for each data point, we take $k = 5$ and $p = q = 2$. As shown in the figure, Lipschitz convex regression does reduce the estimation error near the boundary of the convex hull of X_i's.



(a) $d = 1$ (b) $d = 2$

Fig. 8 Result of Lipschitz convex regression with the data-driven Lipschitz estimation method

7 Real applications

In this section, we apply our framework for estimating the multivariate shape-constrained convex functions in some real applications, namely, pricing of European call options, pricing of basket options, prediction of average weekly wages and estimation of production functions.

7.1 Option pricing of European call options

Consider a European call option whose payoff at maturity T is $(S_T - K)_+$, where S_T is a random variable that stands for the stock price at T, and K is the predetermined strike price. We are interested in the option price at time t, which is defined as

$$V(S) := \mathbb{E}[e^{-r(T-t)}(S_T - K)_+ \mid S_t = S], \quad S > 0,$$

where r is the risk-free interest rate. Under the Black-Scholes model, the random variable S_T satisfies

$$\log S_T \sim \mathcal{N}\Big(\log S_t + (r - \frac{1}{2}\sigma^2)(T-t), \sigma^2(T-t)\Big),$$

where σ is the volatility. It is well-known that $V(\cdot)$ is a convex function with $0 \le V'(S) \le 1$ for $S > 0$. Therefore, we can use the shape-constrained convex regression model with Property (S2) to estimate the function $V(\cdot)$.

There are two reasons why we consider this application to demonstrate the numerical performance of our framework. One is that $V(\cdot)$ admits a closed-form solution as

$$V(S) = S\Phi(d_1) - Ke^{-r(T-t)}\Phi(d_2), \quad d_{1,2} = \frac{\log \frac{S}{K} + (r \pm \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}},$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. The second reason is that the estimation of function $V(\cdot)$ is used in pricing American-type options by approximate dynamic programming, e.g. [29].

In our experiment, we take $t = 0.1$, $T = 0.4$, $K = 10$, $r = 0$, $\sigma = 0.2$. We sample 200 data points, denoted as $\{(S_i, V_i)\}_{i=1}^{200}$. For each S_i, $\log S_i$ is sampled following the distribution $\mathcal{N}(\log K + (r - \sigma^2/2)t, \sigma^2 t)$, and the corresponding V_i is sampled such that $\log V_i$ follows the distribution $\mathcal{N}(\log S_i + (r - \sigma^2/2)(T - t), \sigma^2(T-t))$. For comparison, we apply several regression models to estimate the conditional expectation function V: linear regression, least squares linear regression on a set of basis functions (e.g. weighted Laguerre basis in [29]), unconstrained convex regression and convex regression with box constraint ($L = 0$, $U = 1$).

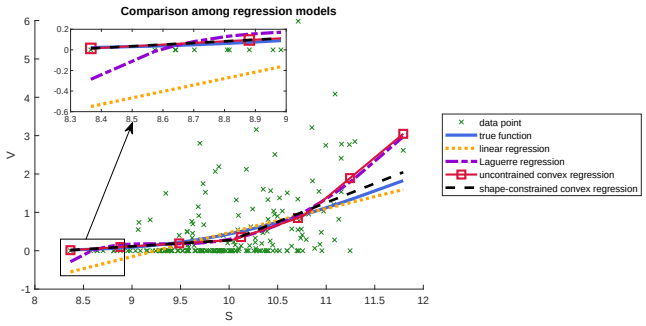


Fig. 9 Results of the estimation of the option pricing of European call option

The comparison among regression models is shown in Figure 9. We can see that the performance of shape-constrained convex regression is the best. The poor performance of linear regression, Laguerre regression and unconstrained convex regression appears near the boundary in three aspects. The first is that

the results from linear regression and Laguerre regression take negative values when S is small, which contradicts the fact that V is always non-negative. The second is that the Laguerre regression can not obtain the required convex property. The last is that when S is large, the gradients of the results obtained by Laguerre regression and unconstrained convex regression are too large. To deal with this over-fitting problem, we add the box constraint to the convex regression, which comes from prior knowledge. We can see that the result of shape-constrained convex regression performs better near the boundary, which demonstrates the advantage of the additional shape constraint.

7.2 Option pricing of basket options

To test multivariate convex regression problems, we consider pricing the basket option on weighted average of M underlying assets.

Basket option of two European call options ($M = 2$). We first consider a basket option of two European call options, where

$$V(x, y) = \mathbb{E}[e^{-r(T-t)}(w_1 S_T^1 + w_2 S_T^2 - K)_+ \mid S_t^1 = x, S_t^2 = y], \quad x, y > 0,$$

where $w = (w_1, w_2)^T$ is a given weight vector such that $w \geq 0$, $w_1 + w_2 = 1$. The random variables S_T^1 and S_T^2 satisfy

$$\begin{pmatrix} \log S_T^1 \\ \log S_T^2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \log S_t^1 + (r - \sigma_1^2/2)(T-t) \\ \log S_t^2 + (r - \sigma_2^2/2)(T-t) \end{pmatrix}, (T-t) \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} \right),$$

where σ_1, σ_2 are volatilities. One can show that $V(\cdot, \cdot)$ is a convex function with $0 \leq \nabla V(x, y) \leq w$, and the proof can be found in Appendix D. We can apply the multivariate shape-constrained convex regression model with Property (S2) ($L = 0$, $U = w$) to estimate the function V.

The convex function $V(\cdot, \cdot)$ does not admit a closed-form solution. However V is also the solution of the Black-Scholes PDE, which can be solved by the finite difference method. The details of the corresponding convection-diffusion equation and the finite difference method for solving it could be found in Appendix E. We use the solution obtained by the finite difference method as the benchmark.

In the experiment, we take $r = 0$, $\rho = 0.1$, $\sigma_1 = 0.2$, $\sigma_2 = 0.3$, $K = 10$, $t = 0$, $T = 0.5$, $w_1 = w_2 = 0.5$. We sample 200 data points, denoted as $\{(S_i, V_i)\}_{i=1}^{200}$, where S_i follows the uniform distribution on the open interval $(0, 5K) \times (0, 5K)$ and V_i follows the distribution

$$\mathcal{N}\left(\log S_i + (T-t) \begin{pmatrix} r - \sigma_1^2/2 \\ r - \sigma_2^2/2 \end{pmatrix}, (T-t) \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} \right).$$

The numerical result is shown in Figure 10. For better illustration, we also plot the absolute error and relative error of the results of the unconstrained

convex regression and shape-constrained convex regression. As we can see, the shape-constrained convex regression performs much better than unconstrained convex regression, especially near the boundary.

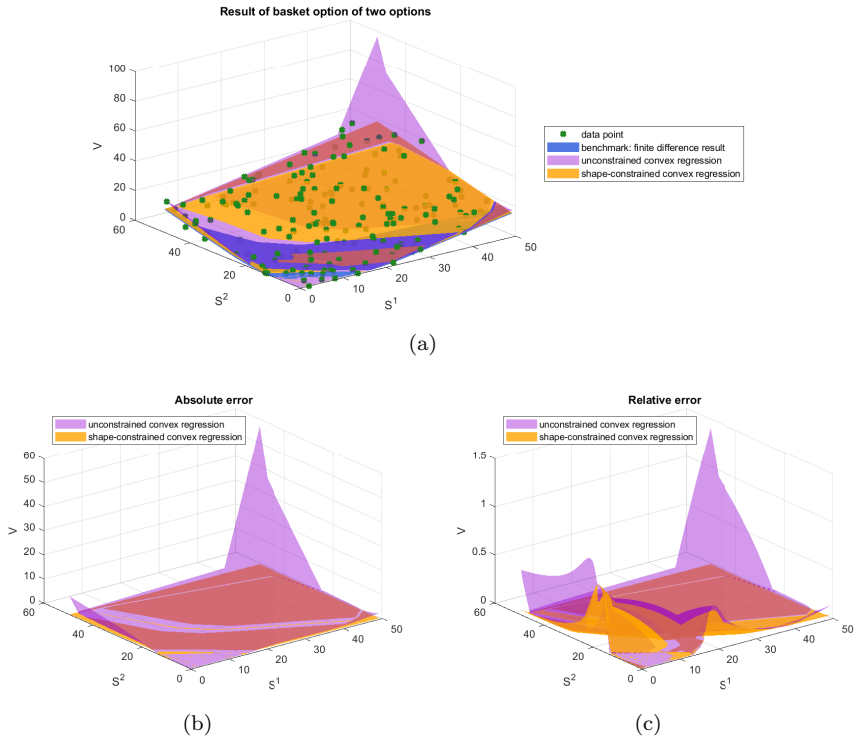


Fig. 10 Result of the estimation of the option pricing of basket option ($M = 2$)

Basket option of more underlying assets ($M > 2$). The basket option in practice always contains many underlying assets, possibly greater than two. The finite difference method is very time-consuming when solving the 3-dimensional convection-diffusion equation, and even impossible to be applied to the higher dimensional cases due to the curse of dimensionality. For $M > 2$, researchers tend to apply the Monte Carlo simulation to estimate the convex function associated with the basket option. Therefore, we treat the solution obtained by the Monte Carlo simulation as the benchmark.

To demonstrate the performance of the shape-constrained convex regression, we design the experiments for estimating the basket option for $M = 5$ and $M = 10$. That is, we consider a basket option of M European call options,

which is defined as: for any $x_1, \cdots, x_M > 0$,

$$V(x_1, \cdots, x_M) = \mathbb{E}[e^{-r(T-t)}(w_1 S_T^1 + \cdots + w_M S_T^M - K)_+ \mid S_t^1 = x_1, \cdots, S_t^M = x_M],$$

where $w = (w_1, \cdots, w_M)^T$ is a given weight vector such that $w \geq 0$, $w_1 + \cdots + w_M = 1$. The random variables $S_T^1, \cdots, S_T^M$ satisfy

$$\begin{pmatrix} \log S_T^1 \\ \vdots \\ \log S_T^M \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \log S_t^1 + (r - \sigma_1^2/2)(T-t) \\ \vdots \\ \log S_t^M + (r - \sigma_M^2/2)(T-t) \end{pmatrix}, (T-t) \begin{pmatrix} \sigma_1^2 & \cdots & \rho\sigma_1\sigma_M \\ \vdots & \ddots & \vdots \\ \rho\sigma_1\sigma_M & \cdots & \sigma_M^2 \end{pmatrix} \right),$$

where $\sigma_1, \cdots, \sigma_M$ are volatilities. Then V is a convex function with $0 \leq \nabla V \leq w$. We apply the multivariate shape-constrained convex regression model with Property (S2) ($L = 0$, $U = w$) to estimate the function V.

Table 5 Estimation of basket option with $M = 5$

Model	n	MSE	Time
UC	200	5.56e+1	00:00:07
	400	1.42e+1	00:00:27
	600	7.41e+1	00:00:22
SC	200	4.07e-1	00:00:12
	400	3.86e-1	00:00:51
	600	5.95e-1	00:00:27

Table 6 Estimation of basket option with $M = 10$

Model	n	MSE	Time
UC	200	2.05e+1	00:00:12
	400	4.06e+1	00:00:10
	600	5.98e+1	00:00:20
SC	200	2.21e+0	00:00:35
	400	1.32e+0	00:00:27
	600	1.00e+0	00:00:42

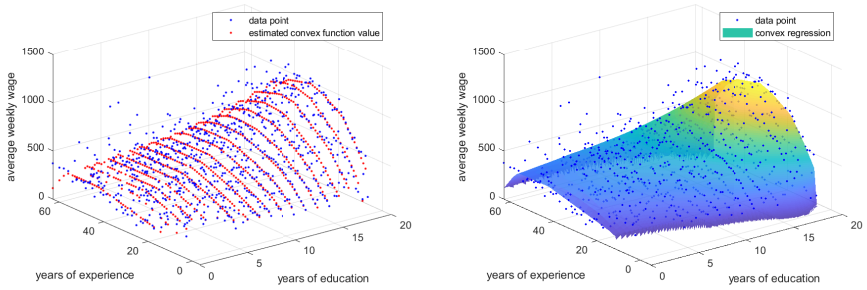
In the experiment, we set $r = 0$, $\rho = 0.1$, $K = 10$, $t = 0$, $T = 0.5$, $w_i = 1/M$, $\sigma_i = 0.2 + 0.025(i-1)$, $i = 1, \cdots, M$. We sample n data points as the case for $M = 2$. To illustrate the performance of our procedure, we uniformly generate 1000 test points in the range $(0, 5K)^M$. At each test point, we use the Monte Carlo simulation with 10^5 samples to compute the "true" function value. We summarize the results of $M = 5$ and $M = 10$ in Table 5 and Table 6, respectively. In the tables, "UC" represents the unconstrained convex regression, "SC" represents the shape-constrained convex regression, and "MSE" represents the mean squared error. As one can see, the shape-constrained convex regression takes a little bit longer time to be solved than the unconstrained convex regression, but get a much better estimated result.

7.3 Prediction of average weekly wages

We consider the problem of estimating the average weekly wages based on years of education and experience as given in [38, Chapter 10]. This dataset is

from 1988 March U.S. Current Population Survey, which can be downloaded as ex1029 in the R package Sleuth2. The set contains weekly wages in 1987 for a sample of 25632 males between the age of 18 and 70 who worked full-time, with their years of education and years of experience. After averaging over a grid with cell size of 1 year by 1 year and ignoring the outliers, we finally come to a dataset with 857 samples.

A reasonable assumption for this application is that the wages are concave in years of experience and a transformation of years of education, i.e., $1.2^{\text{years of education}}$, according to [18]. The estimated result is shown in Figure 11. The shape-constrained convex regression problem is solved within 1 minute.



(a) Estimated function values at each X_i (b) Visualization of the function

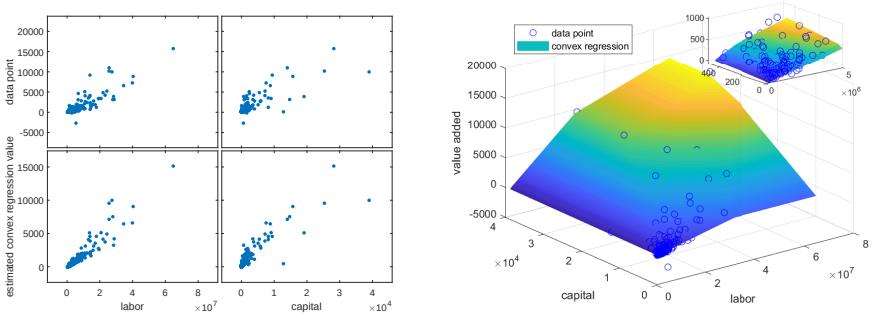
Fig. 11 Results of the estimation of average weekly wages

7.4 Estimation of production functions

In economics, a production function gives the technological relation between quantities of inputs and quantities of output of goods. Production functions are known to be concave and non-decreasing [19, 45, 47]. We apply our framework to estimate the production function for the plastic industry (CIIU3 industry code: 2520) in the year 2011. The dataset can be downloaded from the website of Chile's National Institute of Statistics. As in the setting in [47], we use labor and capital as the input variables, and value added as the output variable. In the dataset, labor is measured as the total man-hours per year, capital and value added are measured in millions of Chilean peso. After removing some outliers, the dataset contains 250 samples. The numerical results can be found in Figure 12. The shape-constrained convex regression problem is solved within 3 seconds.

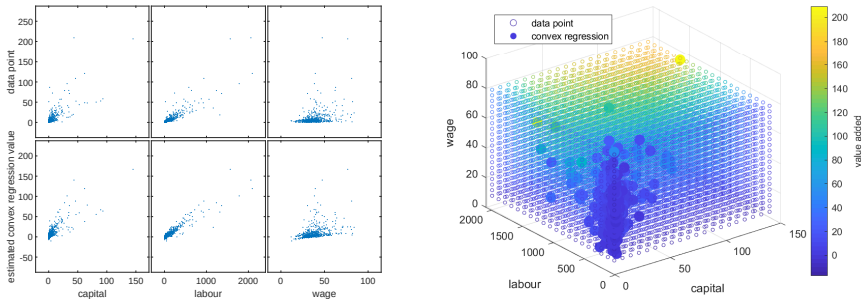
Another example is to explain the labour demand of 569 Belgian firms for the year 1996. The dataset can be obtained from [46][2]. The dataset includes the

[2] `https://www.wiley.com/legacy/wileychi/verbeek2ed/datasets.html`



(a) Correlogram of the function (b) Visualization of the function

Fig. 12 Result of estimation of production function of plastic in Chile



(a) Correlogram of the function (b) Visualization of the function

Fig. 13 Result of estimation of production function of Belgian firms

total number of employees (labour), their average wage (wage), the amount of capital (capital) and a measure of output (value added). The labour is measured as the number of workers, the wage is measured in units of 1000 euro, and the capital and value added is measured in units of a million euro. After removing the outliers, the dataset contains 562 samples. The result can be found in Figure 13. The problem is solved in 22 seconds.

8 Conclusion and future work

In this paper, we provide a unified framework for computing a least squares estimator for the multivariate shape-constrained convex regression function. In addition, we propose an efficient algorithm, which is a semismoothn Newton based proximal augmented Lagrangian method, to solve the large-scale constrained QP in the framework. Moreover, in order to accelerate the computation under the large-sample setting, we design a practical implementation of the constraint generation method, where the reduce problem in each round

is solved by the proxALM. We conduct extensive numerical experiments to demonstrate the efficiency and robustness of our proposed proxALM, as well as the superior performance of the acceleration with the constraint generation method.

Acknowledgements The authors would like to thank Professor Necdet S. Aybat for helpful clarifications on his work in [3].

References

1. Aıt-Sahalia, Y., Duarte, J.: Nonparametric option pricing under shape restrictions. Journal of Econometrics **116**(1-2), 9–47 (2003)
2. Allon, G., Beenstock, M., Hackman, S., Passy, U., Shapiro, A.: Nonparametric estimation of concave production technologies by entropic methods. Journal of Applied Econometrics **22**(4), 795–816 (2007)
3. Aybat, N.S., Wang, Z.: A parallel method for large scale convex regression problems. In: 53rd IEEE Conference on Decision and Control, pp. 5710–5717. IEEE (2014)
4. Balázs, G., György, A., Szepesvári, C.: Near-optimal max-affine estimators for convex regression. In: AISTATS (2015)
5. Beck, A., Teboulle, M.: Smoothing and first order methods: A unified framework. SIAM Journal on Optimization **22**(2), 557–580 (2012)
6. Bertsimas, D., Mundru, N.: Sparse convex regression. INFORMS Journal on Computing, to appear (2020)
7. Bertsimas, D., Tsitsiklis, J.N.: Introduction to Linear Optimization, vol. 6. Athena Scientific Belmont, MA (1997)
8. Blanchet, J., Glynn, P.W., Yan, J., Zhou, Z.: Multivariate distributionally robust convex regression under absolute error loss. In: Advances in Neural Information Processing Systems, pp. 11817–11826 (2019)
9. Chen, C., He, B., Ye, Y., Yuan, X.: The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. Mathematical Programming **155**(1-2), 57–79 (2016)
10. Chen, H., Yao, D.D.: Fundamentals of queueing networks: Performance, asymptotics, and optimization, vol. 46. Springer Science & Business Media (2013)
11. Chen, L., Sun, D.F., Toh, K.C.: An efficient inexact symmetric Gauss–Seidel based majorized ADMM for high-dimensional convex composite conic programming. Mathematical Programming **161**(1-2), 237–270 (2017)
12. Chen, W., Mazumder, R.: Multivariate convex regression at scale. arXiv preprint arXiv:2005.11588 (2020)
13. Chen, X., Sun, D., Sun, J.: Complementarity functions and numerical experiments on some smoothing Newton methods for second-order-cone complementarity problems. Computational Optimization and Applications **25**(1-3), 39–56 (2003)
14. Cui, Y., Pang, J.S., Sen, B.: Composite difference-max programs for modern statistical estimation problems. SIAM Journal on Optimization **28**(4), 3344–3374 (2018)
15. Dontchev, A.L., Qi, H., Qi, L.: Quadratic convergence of Newton's method for convex interpolation and smoothing. Constructive Approximation **19**(1) (2003)
16. Facchinei, F., Pang, J.S.: Finite-Dimensional Variational Inequalities and Complementarity Problems. Springer Science & Business Media (2007)
17. Han, J., Sun, D.F.: Newton and quasi-Newton methods for normal maps with polyhedral sets. Journal of Optimization Theory and Applications **94**(3), 659–676 (1997)
18. Hannah, L.A., Dunson, D.B.: Multivariate convex regression with adaptive partitioning. The Journal of Machine Learning Research **14**(1), 3261–3294 (2013)
19. Hanoch, G., Rothschild, M.: Testing the assumptions of production theory: a nonparametric approach. Journal of Political Economy **80**(2), 256–275 (1972)
20. Hanson, D., Pledger, G.: Consistency in concave regression. The Annals of Statistics pp. 1038–1050 (1976)

21. Hildreth, C.: Point estimates of ordinates of concave functions. Journal of the American Statistical Association **49**(267), 598–619 (1954)
22. Kummer, B.: Newton's method for non-differentiable functions. Advances in Mathematical Optimization **45**, 114–125 (1988)
23. Kuosmanen, T.: Representation theorem for convex nonparametric least squares. The Econometrics Journal **11**(2), 308–325 (2008)
24. Li, X., Sun, D., Toh, K.C.: On efficiently solving the subproblems of a level-set method for fused lasso problems. SIAM Journal on Optimization **28**(2), 1842–1866 (2018)
25. Li, X., Sun, D., Toh, K.C.: An asymptotically superlinearly convergent semismooth Newton augmented Lagrangian method for Linear Programming. SIAM Journal on Optimization **30**(3), 2410–2440 (2020)
26. Li, X., Sun, D., Toh, K.C.: On the efficient computation of a generalized Jacobian of the projector over the Birkhoff polytope. Mathematical Programming **179**(1-2), 419–446 (2020)
27. Lim, E.: On convergence rates of convex regression in multiple dimensions. INFORMS Journal on Computing **26**(3), 616–628 (2014)
28. Lim, E., Glynn, P.W.: Consistency of multidimensional convex regression. Operations Research **60**(1), 196–208 (2012)
29. Longstaff, F.A., Schwartz, E.S.: Valuing American options by simulation: a simple least-squares approach. The Review of Financial Studies **14**(1), 113–147 (2001)
30. Mazumder, R., Choudhury, A., Iyengar, G., Sen, B.: A computational framework for multivariate convex regression and its variants. Journal of the American Statistical Association **114**(525), 318–331 (2019)
31. Meyer, R.F., Pratt, J.W.: The consistent assessment and fairing of preference functions. IEEE Transactions on Systems Science and Cybernetics **4**(3), 270–278 (1968)
32. Mifflin, R.: Semismooth and semiconvex functions in constrained optimization. SIAM Journal on Control and Optimization **15**(6), 959–972 (1977)
33. Moreau, J.J.: Proximité et dualité dans un espace hilbertien. Bulletin de la Société mathématique de France **93**, 273–299 (1965)
34. Nesterov, Y.: Smooth minimization of non-smooth functions. Mathematical Programming **103**(1), 127–152 (2005)
35. Nocedal, J., Wright, S.: Numerical Optimization. Springer Science & Business Media (2006)
36. Qi, H., Yang, X.: Regularity and well-posedness of a dual program for convex best C^1-spline interpolation. Computational Optimization and Applications **37**(3), 409–425 (2007)
37. Qi, L., Sun, J.: A nonsmooth version of Newton's method. Mathematical Programming **58**(1-3), 353–367 (1993)
38. Ramsey, F., Schafer, D.: The Statistical Sleuth: A Course in Methods of Data Analysis. Boston: Cengage Learning (2012)
39. Robinson, S.M.: Some continuity properties of polyhedral multifunctions. In: Mathematical Programming at Oberwolfach, pp. 206–214. Springer (1981)
40. Rockafellar, R.T.: Convex Analysis, vol. 28. Princeton University Press (1970)
41. Rockafellar, R.T.: Monotone operators and the proximal point algorithm. SIAM Journal on Control and Optimization **14**(5), 877–898 (1976)
42. Seijo, E., Sen, B.: Nonparametric least squares estimation of a multivariate convex regression function. The Annals of Statistics **39**(3), 1633–1657 (2011)
43. Sun, D.F., Sun, J.: Semismooth matrix-valued functions. Mathematics of Operations Research **27**(1), 150–169 (2002)
44. Varian, H.R.: The nonparametric approach to demand analysis. Econometrica: Journal of the Econometric Society pp. 945–973 (1982)
45. Varian, H.R.: The nonparametric approach to production analysis. Econometrica: Journal of the Econometric Society pp. 579–597 (1984)
46. Verbeek, M.: A Guide to Modern Econometrics. John Wiley & Sons (2008)
47. Yagi, D., Chen, Y., Johnson, A.L., Kuosmanen, T.: Shape-constrained kernel-weighted least squares: Estimating production functions for Chilean manufacturing industries. Journal of Business & Economic Statistics pp. 1–12 (2018)
48. Zhao, X.Y., Sun, D.F., Toh, K.C.: A Newton-CG augmented Lagrangian method for semidefinite programming. SIAM Journal on Optimization **20**(4), 1737–1765 (2010)

Appendices

A Derivation of the proximal mapping and generalized Jacobian associated with $\mathcal{D} = \{x \in \mathbb{R}^d \mid \|x\|_1 \le L\}$

For $x \in \mathbb{R}^d$, let $P_x = \mathrm{Diag}(\mathrm{sign}(x)) \in \mathbb{R}^{d\times d}$, then

$$\begin{aligned}\Pi_{\mathcal{D}}(x) &= \operatorname*{arg\,min}_{y\in\mathbb{R}^d} \Big\{\frac{1}{2}\|y-x\|^2 \mid \|y\|_1 \le L\Big\}\\ &= LP_x\Big(\operatorname*{arg\,min}_{y\in\mathbb{R}^d} \Big\{\frac{1}{2}\|y-P_x x/L\|^2 \mid e_d^T y \le 1, y\ge 0\Big\}\Big)\\ &= \begin{cases} x & \text{if } \|x\|_1 \le L,\\ LP_x\Pi_{\Delta_d}(P_x x/L) & \text{otherwise,}\end{cases}\end{aligned}$$

where $\Delta_d = \{x \in \mathbb{R}^d \mid e_d^T x = 1, x \ge 0\}$. To derive the generalized Jacobian of $\Pi_{\mathcal{D}}(\cdot)$, we need the generalized Jacobian of $\Pi_{\Delta_d}(\cdot)$. Following the idea in [17,26], we can explicitly compute an element of the generalized Jacobian of $\Pi_{\Delta_d}(\cdot)$ at $P_x x/L$. Let K be the set of index i such that $(\Pi_\Delta(P_x x/L))_i = 0$. Then

$$\widetilde{H} = I_d - \begin{bmatrix} I_K^T & e_d\end{bmatrix}\Big(\begin{bmatrix} I_K\\ e_d^T\end{bmatrix}\begin{bmatrix} I_K^T & e_d\end{bmatrix}\Big)^\dagger \begin{bmatrix} I_K\\ e_d^T\end{bmatrix}$$

is an element in $\partial\Pi_{\Delta_d}(P_x x/L)$, where I_K means the matrix consisting of the rows of the identity matrix I_d, indexed by K. After some algebraic computation, we can see

$$\widetilde{H} = I_d - \begin{bmatrix} I_K^T & e_d\end{bmatrix}\begin{bmatrix} I_{|K|} + \frac{1}{n-|K|}e_{|K|}e_{|K|}^T & -\frac{1}{n-|K|}e_{|K|}\\ -\frac{1}{n-|K|}e_{|K|}^T & \frac{1}{n-|K|}\end{bmatrix}\begin{bmatrix} I_K\\ e_d^T\end{bmatrix} = \mathrm{Diag}(r) - \frac{1}{\mathrm{nnz}(r)}rr^T,$$

where $r \in \mathbb{R}^d$ is defined as $r_i = 1$ if $(\Pi_\Delta(P_x x/L))_i \ne 0$ and $r_i = 0$ otherwise. Therefore,

$$H \in \partial\Pi_{\mathcal{D}}(x), \quad \text{where } H = \begin{cases} I_d & \text{if } \|x\|_1 \le L,\\ P_x\widetilde{H}P_x & \text{otherwise.}\end{cases}$$

B A symmetric Gauss-Seidel based alternating direction method of multipliers (sGS-ADMM) for (P)

In the literature, popular first-order methods based on the framework of alternating direction method of multipliers have been applied to solve (P). In [30, Section A.2], the problem (P) is reformulated as

$$\min_{\theta\in\mathbb{R}^n,\xi\in\mathbb{R}^{dn},\eta\in\mathbb{R}^{n^2}} \Big\{\frac{1}{2}\|\theta - Y\|^2 + p(\xi) + \delta_+(\eta) \mid A\theta + B\xi - \eta = 0\Big\}.$$

The corresponding augmented Lagrangian function for a fixed $\sigma > 0$ is defined by

$$\widetilde{\mathcal{L}}_\sigma(\theta,\xi,\eta;u) = \frac{1}{2}\|\theta - Y\|^2 + p(\xi) + \delta_+(\eta) + \frac{\sigma}{2}\|A\theta + B\xi - \eta - \frac{u}{\sigma}\|^2 - \frac{1}{2\sigma}\|u\|^2.$$

Then the two-block ADMM is given as

$$\begin{cases}\xi^{k+1} = \arg\min\ \widetilde{\mathcal{L}}_\sigma(\theta^k,\xi,\eta^k;u^k) = \arg\min\ \Big\{p(\xi) + \frac{\sigma}{2}\|A\theta^k + B\xi - \eta^k - \frac{u^k}{\sigma}\|^2\Big\},\\ (\theta^{k+1},\eta^{k+1}) = \arg\min\ \widetilde{\mathcal{L}}_\sigma(\theta,\xi^{k+1},\eta;u^k),\\ u^{k+1} = u^k - \tau\sigma(A\theta^{k+1} + B\xi^{k+1} - \eta^{k+1}),\end{cases}$$

where $\tau \in (0, (1+\sqrt{5}/2))$ is a given step length. As described in [30], the subproblem of updating ξ is separable in the variables ξ_i's for $i = 1, \cdots, n$, and the update of each ξ_i can be solved by using an interior point method. The update of θ and η is performed by using a block coordinate descent method, which may converge slowly. One can also apply the directly extended three-block `ADMM` algorithm as in [30, Section 2.1] to solve (P), and the steps are given by

$$\begin{cases} \xi^{k+1} = \arg\min \ \widetilde{\mathcal{L}}_\sigma(\theta^k, \xi, \eta^k; u^k), \\ \theta^{k+1} = \arg\min \ \widetilde{\mathcal{L}}_\sigma(\theta, \xi^{k+1}, \eta^k; u^k), \\ \eta^{k+1} = \arg\min \ \widetilde{\mathcal{L}}_\sigma(\theta^{k+1}, \xi^{k+1}, \eta; u^k), \\ u^{k+1} = u^k - \tau\sigma(A\theta^{k+1} + B\xi^{k+1} - \eta^{k+1}). \end{cases}$$

In the directly extended three-block `ADMM`, the subproblem of updating θ can be computed by solving a linear system, and that of updating η can be solved by the projection onto $\mathbb{R}^{n^2}_+$. However, it is shown in [9] that the directly extended three-block `ADMM` may not be convergent. Thus it is desirable to employ an algorithm that is guaranteed to converge.

In this section, we aim to present an efficient and convergent multi-block `ADMM` for solving (P). The authors in [11] have proposed an inexact symmetric Gauss-Seidel based multi-block `ADMM` for solving high-dimensional convex composite conic optimization problems, and it was demonstrated to perform better than the possibly nonconvergent directly extended multi-block `ADMM`. To adapt the `sGS-ADMM` in [11] to solve (P), we first rewrite (P) as follows:

$$\min_{\theta\in\mathbb{R}^n, \xi, y\in\mathbb{R}^{dn}, \eta\in\mathbb{R}^{n^2}} \Big\{ \frac{1}{2}\|\theta - Y\|^2 + p(y) + \delta_+(\eta) \ \Big| \ A\theta + B\xi - \eta = 0, \ \xi - y = 0 \Big\}. \tag{24}$$

Given a parameter $\sigma > 0$, the augmented Lagrangian function associated with (24) is defined by

$$\begin{aligned} \widehat{\mathcal{L}}_\sigma(\theta, \xi, y, \eta; u, v) &= \frac{1}{2}\|\theta - Y\|^2 + p(y) + \delta_+(\eta) - \langle u, A\theta + B\xi - \eta\rangle - \langle v, \xi - y\rangle \\ &\quad + \frac{\sigma}{2}\|A\theta + B\xi - \eta\|^2 + \frac{\sigma}{2}\|\xi - y\|^2 \\ &= \frac{1}{2}\|\theta - Y\|^2 + p(y) + \delta_+(\eta) + \frac{\sigma}{2}\|A\theta + B\xi - \eta - \frac{u}{\sigma}\|^2 + \frac{\sigma}{2}\|\xi - y - \frac{v}{\sigma}\|^2 \\ &\quad - \frac{1}{2\sigma}\|u\|^2 - \frac{1}{2\sigma}\|v\|^2. \end{aligned} \tag{25}$$

Then the `sGS-ADMM` algorithm for solving (P) is given as in Algorithm 4.

In Algorithm 4, all the subproblems can be solved explicitly. In **Step 1**, η^{k+1} and y^{k+1} are separable and can be solved independently as

$$y^{k+1} = \mathrm{Prox}_{p/\sigma}(\xi^k - v^k/\sigma), \quad \eta^{k+1} = \Pi_+(A\theta^k + B\xi^k - u^k/\sigma),$$

where $\Pi_\pm(\cdot)$ denotes the projection onto $\mathbb{R}^{n^2}_\pm$. In **Step 2a** and **Step 2c**, θ can be computed by solving the following linear system

$$(I_n + \sigma A^T A)\theta = Y - \sigma A^T(B\xi - \eta - u/\sigma).$$

By noting that $A^T A = 2nI_n - 2e_n e_n^T$, one can apply the Sherman-Morrison-Woodbury formula to compute

$$(I_n + \sigma A^T A)^{-1} = \frac{1}{1 + 2\sigma n}(I_n + 2\sigma e_n e_n^T).$$

Thus θ can be computed in $O(n)$ operations. For **Step 2b**, ξ^{k+1} can be computed by solving the linear equation

$$(I_{dn} + B^T B)\xi = y^{k+1} + v^k/\sigma - B^T(A\widehat{\theta}^{k+1} - \eta^{k+1} - u^k/\sigma).$$

Algorithm 4 : Symmetric Gauss-Seidel based ADMM for (P)

1: **Initialization:** Choose an initial point $(\theta^0, \xi^0, y^0, \eta^0, u^0, v^0) \in \mathbb{R}^n \times \mathbb{R}^{dn} \times \mathbb{R}^{dn} \times \mathbb{R}^{n^2} \times \mathbb{R}^{n^2} \times \mathbb{R}^{dn}$, and a positive parameter $\sigma > 0$. For $k = 0, 1, 2, \ldots$

2: **repeat**

3: **Step 1**. Compute

$$(y^{k+1}, \eta^{k+1}) = \arg\min \ \widehat{\mathcal{L}}_\sigma(\theta^k, \xi^k, y, \eta; u^k, v^k).$$

4: **Step 2**. Compute

$$\begin{aligned}
&\textbf{Step 2a}.\ \widehat{\theta}^{k+1} = \arg\min \ \widehat{\mathcal{L}}_\sigma(\theta, \xi^k, y^{k+1}, \eta^{k+1}; u^k, v^k),\\
&\textbf{Step 2b}.\ \xi^{k+1} = \arg\min \ \widehat{\mathcal{L}}_\sigma(\widehat{\theta}^{k+1}, \xi, y^{k+1}, \eta^{k+1}; u^k, v^k),\\
&\textbf{Step 2c}.\ \theta^{k+1} = \arg\min \ \widehat{\mathcal{L}}_\sigma(\theta, \xi^{k+1}, y^{k+1}, \eta^{k+1}; u^k, v^k).
\end{aligned}$$

5: **Step 3**. Compute

$$u^{k+1} = u^k - \tau\sigma(A\theta^{k+1} + B\xi^{k+1} - \eta^{k+1}), \quad v^{k+1} = v^k - \tau\sigma(\xi^{k+1} - y^{k+1}),$$

where $\tau \in (0, (1+\sqrt{5})/2)$ is the step length that is typically chosen to be 1.618.

6: **until** Stopping criterion is satisfied.

As the coefficient matrix $I_{dn} + B^T B$ is a block diagonal matrix consisting of n blocks of $d \times d$ submatrices, each ξ_i can be computed separately, and the inverse of each block only needs to be computed once.

The convergence result of Algorithm 4 is presented in the following theorem, which is taken directly from [11, Theorem 5.1].

Theorem 5 *Suppose that the solution set to the KKT system* (10) *is nonempty. Let* $\{(\theta^k, \xi^k, y^k, \eta^k, u^k, v^k)\}$ *be the sequence generated by Algorithm 4. Then* $\{(\theta^k, \xi^k, y^k, \eta^k)\}$ *converges to an optimal solution of problem* (24)*, and* $\{(u^k, v^k)\}$ *converges to an optimal solution of its dual* (D)*.*

C More results on comparison of algorithms for solving (P)

Table 7 – Table 12 show the comparison among proxALM, sGS-ADMM and MOSEK on instances with relatively large d and n. Note that here we set the stopping criterion to $R_{\mathrm{KKT}} \leq 10^{-6}$ to show that our proposed proxALM is capable of solving the problem (P) to relatively high accuracy. As one can see that, when estimating the function $\psi(x) = \exp(p^T x)$ for moderate $(d, n) = (100, 1000)$, proxALM is about 3 times faster than sGS-ADMM, and about 29 times faster than MOSEK. For the case when $d = 100$, $n = 4000$, which is a large problem with $404,000$ variables and about $16,000,000$ inequality constraints, MOSEK runs out of memory, while proxALM could solve it within 7 minutes and sGS-ADMM takes 17 minutes. From the tables, we can see that sGS-ADMM performs much better than MOSEK in each instance, and proxALM performs even better than sGS-ADMM. In most of the cases, proxALM is at least 10 times faster than MOSEK.

D Property of basket option of two European call options

The function $V(x, y)$ is differentiable since it is the solution of the Black-Scholes PDE. By the definition of V, we can see that V is non-decreasing in x and y, which means that

Table 7 Convex regression for test function $\psi(x) = \exp(p^T x)$, where p is a given random vector with each coordinate drawn from the standard normal distribution

Algorithm \ (d,n)		$(50,500)$	$(50,1000)$	$(50,2000)$	$(100,1000)$	$(100,2000)$	$(100,4000)$
proxALM	Iteration	12(11)*	16(20)	21(38)	15(20)	20(38)	26(51)
	Time	00:00:02	00:00:06	00:00:57	00:00:07	00:01:14	00:06:44
	$R_{\rm KKT}$	4.18e-8	8.97e-8	9.14e-7	6.14e-7	3.41e-7	9.48e-7
sGS-ADMM	Iteration	389	562	1206	355	701	1263
	Time	00:00:05	00:00:25	00:03:57	00:00:19	00:02:39	00:16:59
	$R_{\rm KKT}$	9.95e-7	9.88e-7	9.92e-7	9.99e-7	9.91e-7	9.98e-7
MOSEK	Iteration	10	11	13	11	10	O.M.
	Time	00:00:20	00:01:50	00:10:50	00:03:22	00:19:46	O.M.
	$R_{\rm KKT}$	6.59e-9	3.92e-9	1.53e-7	7.98e-10	7.65e-8	O.M.

* "12(11)" means "proxALM iterations (total inner SSN iterations)". O.M. means the algorithm runs out of memory. Time is in the format of hours:minutes:seconds.

Table 8 Convex regression with monotone constraint (non-decreasing) for the test function $\psi(x) = (e_d^T x)_+$

Algorithm \ (d,n)		$(50,500)$	$(50,1000)$	$(50,2000)$	$(100,1000)$	$(100,2000)$	$(100,4000)$
proxALM	Iteration	15(18)	17(23)	23(77)	17(29)	21(59)	32(96)
	Time	00:00:02	00:00:07	00:02:48	00:00:12	00:02:09	00:12:34
	$R_{\rm KKT}$	1.87e-7	1.50e-7	1.38e-7	8.16e-7	8.23e-7	8.97e-7
sGS-ADMM	Iteration	529	917	1685	541	905	1582
	Time	00:00:08	00:00:49	00:06:18	00:00:34	00:03:50	00:25:18
	$R_{\rm KKT}$	9.79e-7	9.99e-7	9.98e-7	9.85e-7	9.88e-7	9.98e-7
MOSEK	Iteration	14	13	14	13	16	O.M.
	Time	00:00:24	00:02:00	00:11:32	00:03:47	00:25:23	O.M.
	$R_{\rm KKT}$	1.54e-9	1.45e-9	2.63e-8	2.37e-7	1.31e-9	O.M.

Table 9 Convex regression with box constraint ($L = 0_d$, $U = e_d$) for the test function $\psi(x) = \ln(1 + \exp(e_d^T x))$

Algorithm \ (d,n)		$(50,500)$	$(50,1000)$	$(50,2000)$	$(100,1000)$	$(100,2000)$	$(100,4000)$
proxALM	Iteration	23(40)	24(67)	30(135)	17(28)	21(60)	33(102)
	Time	00:00:03	00:00:18	00:04:35	00:00:12	00:02:32	00:12:56
	$R_{\rm KKT}$	9.55e-7	8.79e-7	7.02e-7	6.65e-8	3.54e-7	9.39e-7
sGS-ADMM	Iteration	663	1016	2689	513	871	1541
	Time	00:00:11	00:00:54	00:10:05	00:00:33	00:03:50	00:23:32
	$R_{\rm KKT}$	9.60e-7	9.73e-7	9.98e-7	9.92e-7	9.95e-7	1.00e-6
MOSEK	Iteration	19	24	31	18	15	O.M.
	Time	00:00:31	00:02:52	00:19:03	00:04:50	00:25:10	O.M.
	$R_{\rm KKT}$	2.40e-7	6.03e-8	1.11e-8	3.18e-9	2.23e-9	O.M.

Table 10 Convex regression with Lipschitz constraint ($p = 1$, $q = \infty$, $L = 1$) for the test function $\psi(x) = \sqrt{1 + x^T x}$

Algorithm \ (d, n)		$(50, 500)$	$(50, 1000)$	$(50, 2000)$	$(100, 1000)$	$(100, 2000)$	$(100, 4000)$
proxALM	Iteration	13(14)	17(30)	24(51)	16(26)	21(44)	33(72)
	Time	00:00:02	00:00:08	00:01:12	00:00:11	00:01:26	00:07:59
	R_{KKT}	5.05e-7	5.08e-7	9.45e-7	4.31e-7	2.41e-7	9.77e-7
sGS-ADMM	Iteration	531	928	1730	509	973	1691
	Time	00:00:09	00:00:50	00:06:47	00:00:33	00:04:21	00:27:33
	R_{KKT}	9.77e-7	9.97e-7	9.84e-7	9.89e-7	9.90e-7	9.98e-7
MOSEK	Iteration	10	11	12	10	11	O.M.
	Time	00:00:23	00:01:55	00:10:32	00:03:38	00:21:27	O.M.
	R_{KKT}	7.51e-9	3.46e-10	1.16e-9	5.87e-13	3.00e-10	O.M.

Table 11 Convex regression with Lipschitz constraint ($p = 2$, $q = 2$, $L = \lambda_{\max}(Q)$) for the test function $\psi(x) = \sqrt{x^T Q x}$

Algorithm \ (d, n)		$(50, 500)$	$(50, 1000)$	$(50, 2000)$	$(100, 1000)$	$(100, 2000)$	$(100, 4000)$
proxALM	Iteration	12(11)	17(30)	21(41)	15(20)	21(41)	23(48)
	Time	00:00:02	00:00:08	00:00:55	00:00:08	00:01:12	00:06:22
	R_{KKT}	1.27e-10	4.07e-7	1.94e-7	3.28e-7	7.10e-7	9.69e-7
sGS-ADMM	Iteration	541	953	1481	494	934	1591
	Time	00:00:11	00:00:53	00:05:39	00:00:35	00:04:22	00:23:59
	R_{KKT}	9.76e-7	9.99e-7	9.99e-7	9.94e-7	9.91e-7	9.91e-7
MOSEK	Iteration	10	13	13	11	12	O.M.
	Time	00:00:23	00:02:03	00:10:57	00:03:44	00:22:47	O.M.
	R_{KKT}	2.50e-7	1.06e-9	1.19e-8	7.53e-9	2.12e-12	O.M.

* $Q \in \mathbb{R}^{d \times d}$ is a randomly generated symmetric and positive definite matrix with known largest eigenvalue.

Table 12 Convex regression with Lipschitz constraint ($p = \infty$, $q = 1$, $L = 1$) for the test function $\psi(x) = \ln(1 + e^{x_1} + \cdots + e^{x_d})$

Algorithm \ (d, n)		$(50, 500)$	$(50, 1000)$	$(50, 2000)$	$(100, 1000)$	$(100, 2000)$	$(100, 4000)$
proxALM	Iteration	12(12)	16(22)	22(44)	15(21)	19(35)	27(62)
	Time	00:00:02	00:00:06	00:00:49	00:00:08	00:01:09	00:08:37
	R_{KKT}	3.04e-7	4.43e-7	8.49e-7	2.07e-7	6.63e-7	8.41e-7
sGS-ADMM	Iteration	413	767	1401	436	775	1379
	Time	00:00:08	00:00:45	00:05:25	00:00:31	00:03:24	00:21:59
	R_{KKT}	9.88e-7	9.96e-7	9.80e-7	9.79e-7	9.99e-7	1.00e-6
MOSEK	Iteration	12	12	14	13	8	O.M.
	Time	00:00:41	00:03:28	00:21:06	00:07:26	00:39:55	O.M.
	R_{KKT}	1.23e-8	1.26e-7	5.33e-9	3.09e-10	2.92e-9	O.M.

$\nabla V(x,y) \geq 0$. According to the distribution of S_T^1 and S_T^2, we have that

$$V(x,y) = e^{-r(T-t)}\mathbb{E}_z f(x,y,z),$$

where

$$f(x,y,z) = (xw_1 e^{(r-\sigma_1^2/2)(T-t)+\sqrt{T-t}z_1} + yw_2 e^{(r-\sigma_2^2/2)(T-t)+\sqrt{T-t}z_2} - K)_+,$$
$$\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \sim \mathcal{N}(0, \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}).$$

For any $x_1, x_2, y \in \mathbb{R}$, we can see that

$$\begin{aligned} |V(x_1,y) - V(x_2,y)| &= e^{-r(T-t)}\Big|\mathbb{E}_z[f(x_1,y,z) - f(x_1,y,z)]\Big| \\ &\leq e^{-r(T-t)}\mathbb{E}_z|f(x_1,y,z) - f(x_1,y,z)| \\ &\leq e^{-r(T-t)}\mathbb{E}_z[w_1 e^{(r-\sigma_1^2/2)(T-t)+\sqrt{T-t}z_1}|x_1 - x_2|] \\ &= w_1|x_1 - x_2|e^{-\sigma_1^2/2(T-t)}\mathbb{E}_z[e^{\sqrt{T-t}z_1}] \\ &= w_1|x_1 - x_2|. \end{aligned}$$

Similarly, we can prove that for any $x, y_1, y_2 \in \mathbb{R}$,

$$|V(x,y_1) - V(x,y_2)| \leq w_2|y_1 - y_2|.$$

Therefore, we have that fact that $0 \leq \nabla V(x,y) \leq w$ for any x, y.

E A finite difference method for estimating the basket option of two European call options

It is well-known that the function $V(x,y) = U(0,x,y)$, where U satisfies the Black-Scholes PDE

$$\begin{cases} \dfrac{\partial U}{\partial t} + rx\dfrac{\partial U}{\partial x} + ry\dfrac{\partial U}{\partial y} + \dfrac{1}{2}\sigma_1^2x^2\dfrac{\partial^2 U}{\partial^2 x^2} + \rho\sigma_1\sigma_2 xy\dfrac{\partial^2 U}{\partial xy} + \dfrac{1}{2}\sigma_2^2y^2\dfrac{\partial^2 U}{\partial^2 y^2} - rU = 0, \\ U(T,x,y) = (w_1x + w_2y - K)^+. \end{cases}$$

Let $\tau = T - t$, $u(\tau,x,y) = U(t,x,y)$, then u satisfies

$$\begin{cases} \dfrac{\partial u}{\partial \tau} - rx\dfrac{\partial u}{\partial x} - ry\dfrac{\partial u}{\partial y} - \dfrac{1}{2}\sigma_1^2x^2\dfrac{\partial^2 u}{\partial^2 x^2} - \rho\sigma_1\sigma_2 xy\dfrac{\partial^2 u}{\partial xy} - \dfrac{1}{2}\sigma_2^2y^2\dfrac{\partial^2 u}{\partial^2 y^2} + ru = 0, \\ u(0,x,y) = (w_1x + w_2y - K)^+. \end{cases}$$

The above convection-diffusion equation can be solved numerically on a bounded region $(0, x_{\max}) \times (0, y_{\max})$ by the standard finite difference method with the artificial boundary conditions

$$\begin{cases} u(\tau,x,0) = c(w_1x, K, r, \tau, \sigma_1), \\ u(\tau,0,y) = c(w_2y, K, r, \tau, \sigma_2), \\ \dfrac{\partial}{\partial x}u(\tau, x_{\max}, y) = w_1, \\ \dfrac{\partial}{\partial y}u(\tau, x, y_{\max}) = w_2, \end{cases}$$

where

$$c(x,K,r,\tau,\sigma) = x\Phi(d_1) - Ke^{-r\tau}\Phi(d_2), \quad d_{1,2} = \frac{\log\frac{x}{K} + (r \pm \frac{1}{2}\sigma^2)\tau}{\sigma\sqrt{\tau}},$$

and $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution.