Smooth Strongly Convex Regression

Andrea Simonetto
IBM Research Europe
Dublin, Ireland
andrea.simonetto@ibm.com

***Abstract*—Convex regression (CR) is the problem of fitting a convex function to a finite number of noisy observations of an underlying convex function. CR is important in many domains and one of its workhorses is the non-parametric least square estimator (LSE). Currently, LSE delivers only non-smooth non-strongly convex function estimates. In this paper, leveraging recent results in convex interpolation, we generalize LSE to smooth strongly convex regression problems. The resulting algorithm relies on a convex quadratically constrained quadratic program. We also propose a parallel implementation, which leverages ADMM, that lessens the overall computational complexity to a tight $O(n^2)$ for n observations. Numerical results support our findings.**

I. Introduction

Convex regression (CR) is concerned with fitting a convex function to a finite number of observations. In particular, suppose that we are given n noisy observations of a convex function $\varphi : \mathbb{R}^d \to \mathbb{R}$ as

$$y_i = \varphi(\boldsymbol{x}_i) + \epsilon_i, \quad i \in I_n := \{1, \dots, n\}, \tag{1}$$

where ϵ_i's are random variables, while $\boldsymbol{x}_i \in \mathbb{R}^d$. The objective of CR is then to estimate the true function φ, given the observations y_i's, in a way in which the estimated function $\hat{\varphi}$ is convex.

CR is a particular class of shape-constrained regression problems, and since its first conception in the 50's, it has attracted much attention in various domains, such as statistics, economics, operations research, signal processing and control [1]–[3]. In economics, CR has been motivated by the need for approximating consumers' utility functions from empirical data [4], a task which has been recently re-considered in the context of personalized optimization with user's feedback [5].

In this paper, we study least squares estimators (LSEs) for CR. LSEs have some key advantages over many other estimators proposed in the literature for CR (e.g., constrained Gaussian processes [6], splines [7], or others [8]). First, LSEs are non-parametric and hence they do not require any tuning and avoid the issue of selecting an appropriate estimation structure. Second, LSEs can be computed by solving a convex quadratic program. Therefore, at least in theory, they can be solved very efficiently using interior point methods. Third, being based on the least squares paradigm, they can be naturally extended to time-varying cases (when the function to be estimated changes continuously in time) by, e.g., exponential forgetting coefficients; these cases are becoming more and more important in the current data streaming era [9].

One of the main theoretic drawback is however that the class of functions that can be enforced is limited to the general convex functions, while in many applications one would like to be able to impose at least smoothness and/or strong convexity.

In this paper, our contributions are as follows,

• First, we propose a novel smooth strongly convex CR algorithm. The resulting estimator has at its heart a convex quadratically constrained quadratic program, with $n + nd$ variables and $n(n-1)$ constraints. We also report on its computational complexity as a function of n. The building blocks for this novel algorithm are the recent results in smooth strongly convex interpolation [10], [11].

• Second, we propose a decomposition scheme based on the alternating direction method of multipliers (ADMM) to lessen the computational complexity and make the method parallel. The resulting computational complexity per iteration is then $O(n^2)$ which is tight (i.e., no LSE can obtain a lower computational complexity). The ADMM approach is based on a properly constructed constraint graph, as well as a dual formulation of the local sub-problems.

The results presented in this paper generalize LSE to smooth strongly convex functions, and the non-smooth results can be re-obtained as a special case[1].

II. Definitions and problem statement

We start by formally defining the functional class of interest. Given two parameters μ and L satisfying $0 \leqslant \mu < L \leqslant +\infty$, we consider convex functions satisfying both a smoothness and a strong convexity condition. Given a convex function $\varphi : \mathbb{R}^d \to \mathbb{R}$, we say that the function is L-smooth and μ-strongly convex, which we denote with $\mathcal{F}_{\mu,L}$, iff the following two conditions are satisfied:

- Inequality $1/L\|\boldsymbol{g}_1 - \boldsymbol{g}_2\|_2 \leqslant \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2$ holds $\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$ and corresponding subgradients $\boldsymbol{g}_1, \boldsymbol{g}_2 \in \mathbb{R}^d$;
- Function $\varphi(\boldsymbol{x}) - \mu/2\|\boldsymbol{x}\|_2^2$ is convex.

These definitions allow for L to be equal to $+\infty$ (i.e., the non-smooth case). In the case of a finite L, the first condition implies differentiability of the function. When $L = +\infty$, this condition becomes vacuous, and the function can be non-differentiable. The class of generic convex functions simply corresponds to $\mathcal{F}_{0,+\infty}$. The case $L = \mu$ can be discarded, as it only involves simple quadratic functions, that can be estimated parametrically much more efficiently.

[1]**Notation.** Vectors are indicated with $\boldsymbol{x} \in \mathbb{R}^n$, and matrices with $\boldsymbol{A} \in \mathbb{R}^{m\times n}$. The Euclidean norm is indicated with $\|\boldsymbol{x}\|_2$, the infinity norm as $\|\boldsymbol{x}\|_\infty$. $(\cdot)^\mathsf{T}$ is the transpose operator. Symmetric positive (semi)-definite matrices of dimension n are indicated as $\boldsymbol{A} \in \mathbb{S}^n_+ (\boldsymbol{A} \in \mathbb{S}^n_{++})$. For convex functions $\varphi : \mathbb{R}^n \to \mathbb{R}$, we indicate with $\partial\varphi(\boldsymbol{x})$ their subgradient at point $\boldsymbol{x}$ and with $\varphi^\star$ their convex conjugate. $O(\cdot)$ is the standard big-O notation.

The problem we are interested in solving can be then formalized by using the empirical ℓ_2 norm as:

$$\hat{\varphi}_n \in \underset{\psi \in \mathcal{F}_{\mu,L}}{\mathsf{argmin}} \Big\{ \sum_{i \in I_n} (y_i - \psi(\boldsymbol{x}_i))^2 \Big\}. \tag{2}$$

As we will see, the solution of (2) will consist of two parts: *(i)* the solution of a finite-dimensional optimization problem defined on the observation set, and *(ii)* an interpolating function which maintains the functional class also in all the other points of the domain. In this respect, we define the notion of $\mathcal{F}_{\mu,L}$-interpolation as follows.

Definition 1: The set $\{(\boldsymbol{x}_i, \boldsymbol{g}_i, f_i)\}_{i \in I_n}$ where $\boldsymbol{x}_i, \boldsymbol{g}_i \in \mathbb{R}^d$ and $f_i \in \mathbb{R}$ for all $i \in I_n$ is $\mathcal{F}_{\mu,L}$-interpolable iff there exists a function $\varphi \in \mathcal{F}_{\mu,L}$, $\varphi : \mathbb{R}^d \to \mathbb{R}$, such that $\boldsymbol{g}_i \in \partial\varphi(\boldsymbol{x}_i)$ and $f_i = \varphi(\boldsymbol{x}_i)$ for all $i \in I_n$. $\diamond$

III. Shape-constrained least-squares

A. State of the art

The infinite dimensional optimization problem in (2) can be reduced to a finite dimensional one for the case $\mathcal{F}_{0,+\infty}$ as follows. Let $f_i = \varphi_n(\boldsymbol{x}_i)$, for $i \in I_n$ and define the vector $\boldsymbol{f} = [f_1, \ldots, f_n]^\mathsf{T} \in \mathbb{R}^n$. Let $\boldsymbol{g}_i = \partial\varphi_n(\boldsymbol{x}_i)$, and define the vector $\boldsymbol{g} = [\boldsymbol{g}_1^\mathsf{T}, \ldots, \boldsymbol{g}_n^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{nd}$. We can now rewrite the optimization problem (2) for $\mathcal{F}_{0,+\infty}$ on the observation set as the following convex quadratic program (QP):

$$\underset{\boldsymbol{f} \in \mathbb{R}^n,\ \boldsymbol{g} \in \mathbb{R}^{nd}}{\mathsf{minimize}} \quad \sum_{i \in I_n} (y_i - f_i)^2 \tag{3a}$$

$$\mathsf{subject\ to:} \quad f_j + \boldsymbol{g}_j^\mathsf{T}(\boldsymbol{x}_i - \boldsymbol{x}_j) \leqslant f_i,\ \forall i,j \in I_n. \tag{3b}$$

Problem (3) is a convex QP with $n + nd$ variables and $n(n-1)$ constraints (after removing the trivial $i = j$ ones). Constraint (3b) imposes convexity on the observation points.

A solution of (3), that can be labelled as $\{(f_i^*, \boldsymbol{g}_i^*)\}_{i \in I_n}$, is $\mathcal{F}_{0,+\infty}$-interpolable by construction. In fact, once the solution is retrieved, an allowed estimator/interpolating function for function φ at point $\boldsymbol{x} \in \mathbb{R}^d$ is given by

$$\hat{\varphi}_n(\boldsymbol{x}) = \max_{i \in I_n} \left\{ f_i^* + \boldsymbol{g}_i^{*,\mathsf{T}}(\boldsymbol{x} - \boldsymbol{x}_i) \right\}. \tag{4}$$

The estimator (4) is a solution for the original problem (2), for the case $\mathcal{F}_{0,+\infty}$. Two comments are in order at this point:

• First, estimator (4) is non-smooth and non-strongly convex in general. While *ad-hoc* smoothing techniques do exist [12], one incurs a daunting trade-off between smoothing quality (in terms of low Lipschitz constant L) and estimation quality (in terms of small error ε w.r.t. the non-smooth solution). Typically, if one wants to retrieve functions of the class $\mathcal{F}_{\mu,L}$ with a low L, one has to expect poor estimation quality ε.

• Second, even though problem (3) is a convex QP and one can use off-the-shelf solvers to solve it efficiently (e.g., OSQP [13], or ECOS [14]), the number of shape constraints is still $O(n^2)$. In addition, its computational complexity grows at least as $O(n^3(d+1)^3)$, making practically hard to solve problems with $n > 200$ and even small d. However, thanks to the decomposable structure of problem (3) one can resort to first-order methods [3], [12], [15] whose computational complexity scales as $O(n^2)$ per iteration, to tackle problems up to $n \sim O(10^3)$ in dimensions that can go up to $d = 200$ for [15], or up to $n \sim O(10^5)$ in $d = 10$, for [16]. Note that a computational complexity of $O(n^2)$ is the least one can expect, given the $O(n^2)$ constraints. Finally, partitioning methods have also been advocated [17] but not explored here.

B. Smooth strongly convex regression

In this paper, we propose a new set of constraints (instead of (3b)) together with an interpolation procedure for $\hat{\varphi}_n(\boldsymbol{x})$ to enforce smoothness and strong convexity automatically. We use and adapt results from [10], [11] for this purpose. The basic idea is that smoothness and strong convexity interchange via the procedure of convex conjugation. While we leave the technical details to the above mentioned papers, we can cite the following interpolability results.

Theorem 1: ($\mathcal{F}_{\mu,L}$-interpolability) [10, Theorem 4] The set $\{(\boldsymbol{x}_i, \boldsymbol{g}_i, f_i)\}_{i \in I_n}$ is $\mathcal{F}_{\mu,L}$-interpolable iff the following set of conditions holds for every pair of indices $i, j \in I_n$:

$$\begin{aligned} f_i - f_j - \boldsymbol{g}_j^\mathsf{T}(\boldsymbol{x}_i - \boldsymbol{x}_j) \geqslant \\ \frac{1}{2(1-\mu/L)} \bigg(\frac{1}{L}\|\boldsymbol{g}_i - \boldsymbol{g}_j\|_2^2 + \mu\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 \\ -2\frac{\mu}{L}(\boldsymbol{g}_j - \boldsymbol{g}_i)^\mathsf{T}(\boldsymbol{x}_j - \boldsymbol{x}_i) \bigg). \end{aligned} \tag{5}$$

With this in place, we are ready to modify the constraint set (3b) to accommodate $\varphi \in \mathcal{F}_{\mu,L}$. In particular, to solve (2) for a $\varphi \in \mathcal{F}_{\mu,L}$, we can leverage the convex problem:

$$\underset{\boldsymbol{f} \in \mathbb{R}^n,\ \boldsymbol{g} \in \mathbb{R}^{nd}}{\mathsf{minimize}} \quad \sum_{i \in I_n} (y_i - f_i)^2 \tag{6a}$$

$$\mathsf{subject\ to:} \quad (5), \quad \forall i,j \in I_n. \tag{6b}$$

This is a convex quadratically constrained quadratic problem (QCQP), a special case of a second-order conic program [18]. Once a solution $\{(f_i^*, \boldsymbol{g}_i^*)\}_{i \in I_n}$ is found, the following theorem describe an allowed estimation/interpolation strategy.

Theorem 2: For any set $\{(\boldsymbol{x}_i, \boldsymbol{g}_i^*, f_i^*)\}_{i \in I_n}$ that is $\mathcal{F}_{\mu,L}$-interpolable, an allowed interpolating function is

$$\hat{\varphi}_n(\boldsymbol{x}) = \mathsf{conv}(p_i(\boldsymbol{x})) + \frac{\mu}{2}\|\boldsymbol{x}\|_2^2 \tag{7}$$

where

$$\begin{aligned} p_i(\boldsymbol{x}) := \frac{L-\mu}{2}\|\boldsymbol{x} - \boldsymbol{x}_i\|_2^2 + (\boldsymbol{g}_i^* - \mu\boldsymbol{x}_i)^\mathsf{T}\boldsymbol{x} + \\ - \boldsymbol{g}_i^{*,\mathsf{T}}\boldsymbol{x}_i + f_i^* + \mu/2\|\boldsymbol{x}_i\|_2^2, \end{aligned} \tag{8}$$

and where $\mathsf{conv}(\cdot)$ indicates the convex hull.

Proof: (Sketch) Follows from [10, Remark 2] by explicitly computing the convex conjugate of their function $h_i(\boldsymbol{x})$ and setting $p_i(\boldsymbol{x}) := h_i^\star(\boldsymbol{x})$. ■

Problem (6) together with the interpolation strategy (7) yield the promised smooth strongly convex estimator for function φ. If we set $\mu = 0$ and $L = +\infty$, we retrieve the non-smooth estimator. For $\mu = 0$, $L < +\infty$, we obtain a non-strongly convex smooth estimator, while for $\mu > 0$ and $L = +\infty$ a non-smooth strongly convex one.

Problem (6) is a QCQP in n+nd variables and $n(n$-$1)$ constraints, which can be solved with off-the-shelf convex solvers, e.g., ECOS [14], or MOSEK. Since the computational complexity grows at least as $O(n^3(d+1)^3)$, similar practical limitations than the non-smooth estimator apply here. We will show how to overcome them by resorting to ADMM next.

C. Parallel implementation

Since the computational complexity of (6) could be prohibitive for practical applications, we move now to understand how one can decompose the problem into smaller parts and reduce the overall complexity. Strategies to use first-order methods (proximal methods, primal/dual, randomization, and ADMM) on the quadratic problem (3) for non-smooth convex functions have been reported in [3], [12], [15], [16]. In the case of (3), the problem is separable in both cost and constraints (once dualized) and a decomposition strategy is rather direct. For the case of (6) however, each constraint couples the variables $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$ due to the quadratic term, and a decomposition is not immediate. We consider here a novel edge-based ADMM decomposition.

Consider the set of constraints of type (5) for all $i, j \in I_n$ and define the constraint graph, as a *directed* graph $G = (V, E)$, whose vertices are the nodes $i \in I_n$, and edges are *all the combinations* of i, j: $E = \{(i, j) | i \in I_n, j \in I_n, i \neq j\}$. The cardinality of E is $|E| = n(n-1)$. For each edge $e \in E$, consider its two nodes, say i and j, and define the edge variables $\boldsymbol{\eta}_{e,i} = [f_i^e, \boldsymbol{g}_i^{e,\mathsf{T}}]^\mathsf{T}$, $\boldsymbol{\eta}_{e,j} = [f_j^e, \boldsymbol{g}_j^{e,\mathsf{T}}]^\mathsf{T}$, as well as $\boldsymbol{\xi}_e = [\boldsymbol{\eta}_{e,i}^\mathsf{T}, \boldsymbol{\eta}_{e,j}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{2(1+d)}$. In this context, each directed edge has its own functional and derivative variables.

We use the notation $i \sim e$ to indicate that node i is one of the two vertices of edge e, while we explicitly write $e(i \to j)$ to indicate that edge e is the directed edge with i as source node and j as sink node. For ease of representation, we also define the local constraint,

$$\mathcal{C}_{e(i\to j)} := \Big\{ f_i^e - f_j^e - \boldsymbol{g}_j^{e,\mathsf{T}}(\boldsymbol{x}_i - \boldsymbol{x}_j) \geqslant \frac{1}{2(1-\mu/L)} \left(\frac{1}{L} \|\boldsymbol{g}_i^e - \boldsymbol{g}_j^e\|_2^2 + \mu \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 - 2\frac{\mu}{L}(\boldsymbol{g}_j^e - \boldsymbol{g}_i^e)^\mathsf{T}(\boldsymbol{x}_j - \boldsymbol{x}_i) \right) \Big\}. \quad (9)$$

We now split problem (6) at the edges, so that each variable $\boldsymbol{\eta}_{e,i}^\mathsf{T}$ containing node i, is different for each edge e having node i has one of its vertices. Then we enforce equality of all the node variables via the supporting vector $\boldsymbol{z}_i \in \mathbb{R}^{1+d}$ for each i, $\boldsymbol{z} = [\boldsymbol{z}_1^\mathsf{T}, \ldots, \boldsymbol{z}_n^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{n+nd}$. With this philosophy, problem (6) can be rewritten in the equivalent form:

$$\underset{\boldsymbol{\xi} \in \mathbb{R}^{2|E|(1+d)},\ \boldsymbol{z} \in \mathbb{R}^{n+nd}}{\mathsf{minimize}} \quad \frac{1}{2n} \sum_{e \in E} \sum_{i \sim e} (y_i - f_i^e)^2 \quad (10a)$$

$$\mathsf{subject\ to:} \quad \mathcal{C}_{e(i\to j)}, \quad \forall e \in E, \quad (10b)$$

$$\boldsymbol{\eta}_{e,i} = \boldsymbol{z}_i, \quad \forall e \in E,\ i \sim e. \quad (10c)$$

It is key that the constraint $\mathcal{C}_{e(i\to j)}$ is present only once for each directed edge.

The above problem can now be tackled with ADMM. We leave the derivations out, since standard, and report only the final result. Start with some initialization for the auxiliary variable $\boldsymbol{z}$ and initialize the *scaled* dual variables of the constraints (10c) as $\boldsymbol{\lambda} = [\ldots, \boldsymbol{\lambda}_{e,i}^\mathsf{T}, \boldsymbol{\lambda}_{e,j}^\mathsf{T}, \ldots]^\mathsf{T} \in \mathbb{R}^{2|E|(1+d)} = \boldsymbol{0}$. Set the penalization $\rho > 0$. Then at each step,

1) Solve the edge QCQP for each edge $e \in E$:

$$[\boldsymbol{\eta}_{e,i}^{+,\mathsf{T}}, \boldsymbol{\eta}_{e,j}^{+,\mathsf{T}}]^\mathsf{T} = \boldsymbol{\xi}_e^+ = \underset{\boldsymbol{\xi}_e \in \mathcal{C}_{e(i\to j)}}{\mathsf{argmin}} \Big\{ \frac{1}{2n} \sum_{i\sim e} (y_i - f_i^e)^2 + \sum_{i \sim e} \frac{\rho}{2} \|\boldsymbol{\eta}_{e,i} - \boldsymbol{z}_i + \boldsymbol{\lambda}_{e,i}\|_2^2, \Big\} \quad (11)$$

2) Update the $\boldsymbol{z}_i$ variables for each node $i \in V$:

$$\boldsymbol{z}_i^+ = \frac{1}{2n} \sum_{e | i \sim e} \boldsymbol{\eta}_{e,i}^+ \quad (12)$$

3) Update the $\boldsymbol{\lambda}_e$ variables $\forall e \in E$, and node $i \sim e$:

$$\boldsymbol{\lambda}_{e,i}^+ = \boldsymbol{\lambda}_{e,i} + (\boldsymbol{\eta}_{e,i}^+ - \boldsymbol{z}_i^+) \quad (13)$$

4) Set $\boldsymbol{\xi}_e = \boldsymbol{\xi}_e^+$ for all $e \in E$, $\boldsymbol{z}_i = \boldsymbol{z}_i^+$ for all $i \in V$, and $\boldsymbol{\lambda}_{e,i} = \boldsymbol{\lambda}_{e,i}^+$, for all $e \in E$, $i \sim e$, and go to 1).

Convergence of the above procedure to a minimizer of the original problem (6) is assured by the standard ADMM convergence results [19]. In practice, one stops the ADMM iterations after a specified error criterion has been met, or a number of iterations has been reached. In this case, we consider as approximate solution to our problem, the final $\boldsymbol{z}$ vector that ADMM yields (we let $\tilde{f}_i$ be the near-optimal $\boldsymbol{f}$ values and $\tilde{\boldsymbol{g}}_i$ be the near-optimal $\boldsymbol{g}$ values) and we use the approximate interpolating function [see Th. 2]:

$$\hat{\varphi}_n(\boldsymbol{x}) = \mathsf{conv}(\tilde{p}_i(\boldsymbol{x})) + \frac{\mu}{2}\|\boldsymbol{x}\|_2^2, \quad (14)$$

$$\tilde{p}_i(\boldsymbol{x}) := \frac{L-\mu}{2}\|\boldsymbol{x} - \boldsymbol{x}_i\|_2^2 + (\tilde{\boldsymbol{g}}_i - \mu \boldsymbol{x}_i)^\mathsf{T}\boldsymbol{x} + - \tilde{\boldsymbol{g}}_i^\mathsf{T}\boldsymbol{x}_i + \tilde{f}_i + \mu/2\|\boldsymbol{x}_i\|_2^2. \quad (15)$$

Note that function (14) is still a L-smooth μ-strongly convex function by construction, just the ADMM approximate solution $\boldsymbol{z}$ is only approximately $\mathcal{F}_{\mu,L}$-interpolable, which means that $\tilde{f}_i \approx \hat{\varphi}_n(\boldsymbol{x}_i)$ and $\tilde{\boldsymbol{g}}_i \approx \partial\hat{\varphi}_n(\boldsymbol{x}_i)$ instead of equality (meaning that function (14) is only approximately minimizing the LS metric). In practice, we notice that this approximation delivers good feasible results for small errors.

D. Solving the local dual

Solving (11) in the primal form is still too computational complex (especially if it has to be done for each edge for each iteration of the ADMM algorithm). However, *since there is only one scalar constraint*, the dual is mono-dimensional and easier to solve with a few iterations of a projected Newton's method. To do this, we write (11) in the standard form:

$$\underset{\boldsymbol{\xi}_e \in \mathbb{R}^{2(1+d)}}{\mathsf{minimize}} \quad \boldsymbol{\xi}_e^\mathsf{T} \boldsymbol{P}_0 \boldsymbol{\xi}_e + \boldsymbol{q}_0^\mathsf{T}\boldsymbol{\xi}_e + r_0 \quad (16a)$$

$$\mathsf{subject\ to:} \quad \boldsymbol{\xi}_e^\mathsf{T} \boldsymbol{P}_1 \boldsymbol{\xi}_e + \boldsymbol{q}_1^\mathsf{T}\boldsymbol{\xi}_e + r_1 \leqslant 0, \quad (16b)$$

where $\boldsymbol{P}_0 \in \mathbb{S}_{+}^{2(d+1)}$, $\boldsymbol{P}_1 \in \mathbb{S}_{++}^{2(d+1)}$, $\boldsymbol{q}_0 \in \mathbb{R}^{2(d+1)}$, $\boldsymbol{q}_1 \in \mathbb{R}^{2(d+1)}$, $r_0 \in \mathbb{R}$, and $r_1 \in \mathbb{R}$ are properly defined matrices, vectors, and scalars to match (16) to the original (11). Define the Lagrangian function

$$\mathcal{L}(\nu) := \boldsymbol{\xi}_e^\mathsf{T} \boldsymbol{P}_0 \boldsymbol{\xi}_e + \boldsymbol{q}_0^\mathsf{T} \boldsymbol{\xi}_e + r_0 + \nu(\boldsymbol{\xi}_e^\mathsf{T} \boldsymbol{P}_1 \boldsymbol{\xi}_e + \boldsymbol{q}_1^\mathsf{T} \boldsymbol{\xi}_e + r_1), \quad (17)$$

for the dual variable $\nu \geqslant 0$. Note that $\boldsymbol{P}_0$ is positive definite by construction (due to $\rho > 0$), and therefore setting $\boldsymbol{P}_\nu := \boldsymbol{P}_0 + \nu \boldsymbol{P}_1$, the inverse $\boldsymbol{P}_\nu^{-1}$ exists and it is well-defined. Set also $\boldsymbol{q}_\nu := \boldsymbol{q}_0 + \nu \boldsymbol{q}_1$. The dual problem of (16) is:

$$\underset{\nu \geqslant 0}{\text{maximize}}\, \phi(\nu) := -\frac{1}{4} \boldsymbol{q}_\nu^\mathsf{T} \boldsymbol{P}_\nu^{-1} \boldsymbol{q}_\nu + \nu r_1 + r_0 \quad (18)$$

whose solution can be found with standard projected Newton's method[2]. Once the dual problem is solved and the unique dual solution ν^* is found, the unique primal solution of (16) can be retrieved as

$$\boldsymbol{\xi}_e^* = -\frac{1}{2} \boldsymbol{P}_{\nu^*}^{-1} \boldsymbol{q}_{\nu^*}. \quad (19)$$

Putting things together, the ADMM procedure (1-4) with a Newton's method to solve the mono-dimensional local dual problems (18) has computational complexity of $n(n-1) \times O((d+1)^3)$ (where the $O((d+1)^3)$ term comes from the computation of the inverse $\boldsymbol{P}_\nu^{-1}$) and for $d \ll n$, the leading error is $O(n^2)$.

IV. Numerical Evaluation

We evaluate the presented algorithms in terms of computational time and mean square error on a toy problem. The aim is to evaluate scalability and quality in a simple one-dimensional setting. A more detailed numerical investigation is deferred to future research efforts. All the computations are performed using Python 3.6, on a 2.7 GHz Dual-Core Intel Core i5 laptop with 8GB of RAM. We use CVXPY [20] for solving the non-smooth and smooth problems. Internally, the QPs are solved with OSQP, while the QCQPs with ECOS.

The true generating function is $\varphi(x) = x^2$ over $x \in [-1, 1]$. Function φ is 2-smooth and 2-strongly convex. The observations are generated by adding noise drawn from $\mathcal{N}(0, \sigma^2)$.

We run *(i)* the non-smooth problem with QP (3) and estimator (4); *(ii)* the L-smooth μ-strongly convex problem with QCQP (6) and estimator (7); *(iii)* the edge-based ADMM to solve the smooth problem with local problems (11) and approximate estimator (14), with $\rho = 1/n$ and initial $\boldsymbol{z}$ specified running a Gaussian process estimator [21], which offers a computationally cheap smooth (but in general non-convex) first approximation. For the Gaussian process we use a square exponential kernel. For the ADMM we solve the local problems resorting to their duals and employing a projected Newton's method with back-tracking. The stopping criterion for the ADMM has been selected as

$$\varepsilon = \max\{\|[\boldsymbol{\eta}_{e,i}^+ - \boldsymbol{z}_i^+]_{e \in E, i \sim e}\|_\infty, \|\boldsymbol{z}^+ - \boldsymbol{z}\|_\infty\}. \quad (20)$$

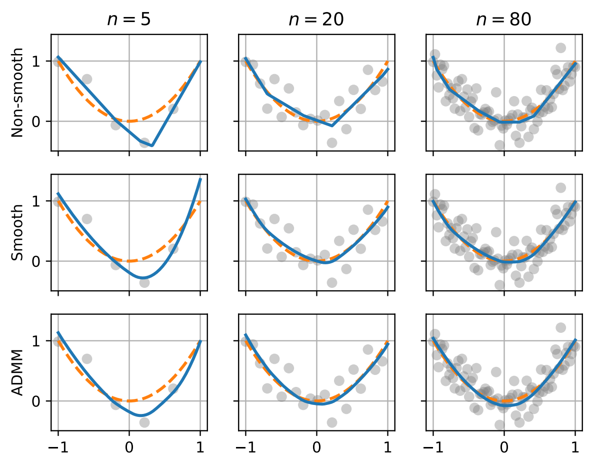

Fig. 1. Examples of estimation of the convex function $\varphi(x) = x^2$ with n noisy observations. In continuous blue line the estimated $\hat{\varphi}$, in dashed orange line the true function. The observations are indicated with grey dots.

In Figure 1, we report the three methods (non-smooth, smooth, and ADMM) for the task of estimating the given function $\varphi(x)$ with an increasing amount of observations. Here the observation noise standard deviation is set to $\sigma = 0.2$, while $\varepsilon = 0.01$. We also set $\mu = 1$ and $L = 5$. We display the estimated function $\hat{\varphi}(x)$ with a continuous blue line, while the true function is a dashed orange line. Using grey dots, we represents the observations. As we notice, non-smooth and smooth estimators achieve the desired estimation task with higher accuracy the more observations are present. ADMM also does well (even with a moderate error ε).

We run simulations for different n's, considering $\sigma = 0.1$, and two versions of ADMM, one with $\varepsilon_1 = 0.03$ and the other with $\varepsilon_2 = 0.01$. (All the other parameters have been left the same as the ones in Figure 1). All the results are averaged over 10 realizations of the observations for the centralized problems, and 25 for the parallel methods (which are more dependent on the realization for the number of iterations).

Figure 2 represents the computational times of the various methods. As we see, the smooth method is the slowest one (as n increases), as expected. The ADMM methods run slower than the other methods at the beginning due to the non-accurate initialization (and therefore due to the need for more iterations to reach the specified stopping criterion), while as n increases, they run the fastest (since the initialization becomes better and better).

Figure 3 represents an error metric defined as

$$E_{\hat{\varphi}_n} = \frac{1}{n_s} \sum_{s \in I_s} (\hat{\varphi}_n(y_s) - \varphi(y_s))^2 \quad (21)$$

This metric is defined over a finer equi-spaced sampling $y_s, s = \{1, \ldots, n_s\} =: I_s$, over $[-1, 1]$ with $n_s = 1000$. The

[2]For the sake of completeness, the gradient of the dual function is

$$\frac{\mathrm{d}\phi}{\mathrm{d}\nu} = -\frac{1}{2} \boldsymbol{q}_1^\mathsf{T} \boldsymbol{P}_\nu^{-1} \boldsymbol{q}_\nu + \frac{1}{4} \boldsymbol{q}_\nu^\mathsf{T} \boldsymbol{P}_\nu^{-1} \boldsymbol{P}_1 \boldsymbol{P}_\nu^{-1} \boldsymbol{q}_\nu + r_1,$$

while the Hessian is

$$\frac{\mathrm{d}^2\phi}{\mathrm{d}\nu^2} = -\frac{1}{2} \boldsymbol{q}_1^\mathsf{T} \boldsymbol{P}_\nu^{-1} \boldsymbol{q}_1 + \boldsymbol{q}_1^\mathsf{T} \boldsymbol{P}_\nu^{-1} \boldsymbol{P}_1 \boldsymbol{P}_\nu^{-1} \boldsymbol{q}_\nu - \frac{1}{2} \boldsymbol{q}_\nu^\mathsf{T} \boldsymbol{P}_\nu^{-1} \boldsymbol{P}_1 \boldsymbol{P}_\nu^{-1} \boldsymbol{P}_1 \boldsymbol{P}_\nu^{-1} \boldsymbol{q}_\nu.$$

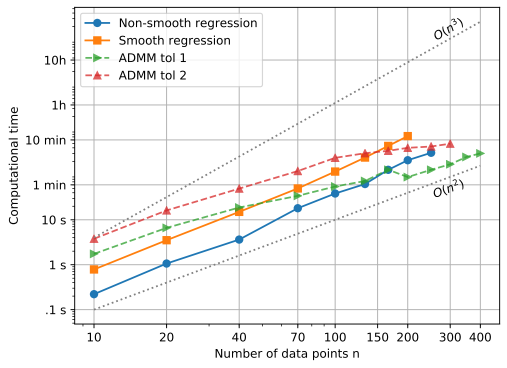

Fig. 2. Computational time for the different methods varying n. The dotted lines indicate the $O(n^2)$ and $O(n^3)$ growth.

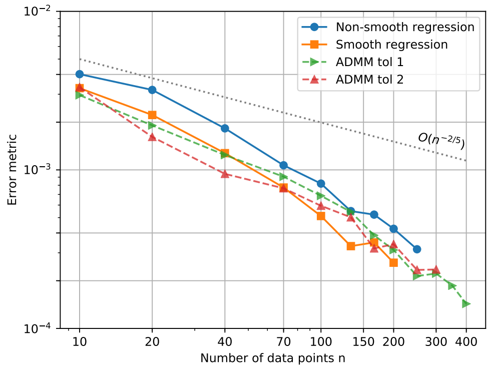

Fig. 3. Mean square error metric for the estimator $\hat{\varphi}_n$ for the different methods varying n (cf. (21)). The dotted line indicates the theoretical $O(n^{-2/5})$ convergence rate for the mean square error metric on the observation points.

idea is to capture the error in the estimator $\hat{\varphi}_n$, rather than just the mean square error on the observation points. As we see and expected, the smooth estimator is better than the non-smooth one. ADMM delivers estimates of comparable accuracy than the smooth one. In the figure, we also display an $O(n^{-2/5})$ line, which is the typical convergence rate of non-parametric LSE for $d = 1$ [12], [22]–[24], in the mean-square-error-on-the-observation-points sense.

What is interesting here is that the metric $E_{\hat{\varphi}_n}$ is built on the estimated $\hat{\varphi}_n$ which for ADMM is the approximate (14). Because of this, the ADMM's estimator can be better than the smooth one (especially in low observation settings) This is per se quite interesting and will be further explored in the future.

V. Conclusion

We have proposed a method to perform smooth strongly convex regression in a non-parametric least squares sense. The method relies on the solution of a convex quadratically constrained quadratic program. We have discussed computational complexity and offered a first-order alternative based on ADMM to lessen the computational load to a tight $O(n^2)$ for n noisy observations of the true function. Future research efforts will explore the use of other first-order methods, such as primal-dual [25].

References

[1] R. J. Samworth and B. Sen, "Editorial: Nonparametric Inference Under Shape Constraints," *Statistical Science*, vol. 33, no. 4, 2018.

[2] P. Groeneboom, G. Jongbloed, and J. A. Wellner, "Estimation of a Convex Function: Characterizations and Asymptotic Theory," *The Annals of Statistics*, vol. 29, no. 6, pp. 1653 – 1698, 2001.

[3] N. S. Aybat and Z. Wang, "A Parallel Method for Large Scale Convex Regression Problems," in *Proceedings of the IEEE Conference in Decision and Control*, 2014.

[4] R. F. Meyer and J. W. Pratt, "The Consistent Assessment and Fairing of Preference Functions," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 3, pp. 270 – 278, 1968.

[5] A. Simonetto, E. Dall'Anese, J. Monteil, and A. Bernstein, "Personalized Optimization with User's Feedback," *arXiv: 1905.00775*, 2019.

[6] X. Wang and J. O. Berger, "Estimating Shape Constrained Functions Using Gaussian Processes," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 4, no. 1, pp. 1 – 25, 2016.

[7] A. L. Dontchev, H. Qi, and L. Qi, "Quadratic Convergence of Newton's Method for Convex Interpolation and Smoothing," *Constructive Approximation*, vol. 19, pp. 123 – 143, 2003.

[8] M. Birke and H. Dette, "Estimating a Convex Function in Nonparametric Regression," *Scandinavian Journal of Statistics*, vol. 34, no. 2, pp. 384 – 404, 2007.

[9] E. Dall'Anese, A. Simonetto, S. Becker, and L. Madden, "Optimization and Learning with Information Streams: Time-varying Algorithms and Applications," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 71 – 83, 2020.

[10] A. Taylor, J. Hendrickx, and F. Glineur, "Smooth Strongly Convex Interpolation and Exact Worst-case Performance of First-order Methods," *Mathematical Programming*, vol. 161, no. 1, pp. 307 – 345, 2017.

[11] A. Taylor, "Convex Interpolation and Performance Estimation of First-order Methods for Convex Optimization," Ph.D. dissertation, Université catholique Louvain, Belgium, January 2017.

[12] R. Mazumder, A. Choudhury, G. Iyengar, and B. Sen, "A Computational Framework for Multivariate Convex Regression and Its Variants," *Journal of the American Statistical Association*, vol. 114, no. 525, pp. 318–331, 2019.

[13] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *arXiv: 1711.08013*, 2017.

[14] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP Solver for Embedded Systems," in *Proceedings of the ECC*, 2013.

[15] M. Lin, D. Sun, and K.-C. Toh, "Efficient algorithms for multivariate shape-constrained convex regression problems," *arXiv: 2002.11410*, 2020.

[16] W. Chen and R. Mazumder, "Multivariate Convex Regression at Scale," 2020, arXiv:2005.11588.

[17] L. A. Hannah and D. B. Dunson, "Multivariate Convex Regression with Adaptive Partitioning," *JMLR*, vol. 14, pp. 3261 – 3294, 2013.

[18] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1 – 122, 2011.

[20] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *JMLR*, vol. 17, no. 83, 2016.

[21] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, US: The MIT Press, 2006.

[22] E. Seijo and B. Sen, "Nonparametric Least Squares Estimation of a Multivariate Convex Regression Function," *The Annals of Statistics*, vol. 39, no. 3, pp. 1633 – 1657, 2011.

[23] E. Lim and P. W. Glynn, "Consistency of Multidimensional Convex Regression," *Operation Research*, vol. 60, no. 1, pp. 196 – 208, 2012.

[24] J. Blanchet, P. W. Glynn, J. Yan, and Z. Zhou, "Multivariate Distributionally Robust Convex Regression under Absolute Error Loss," in *Proceedings of NeurIPS*, 2019.

[25] N. Komodakis and J.-C. Pesquet, "Playing with Duality: An Overview of Recent Primal-Dual Approaches for Solving Large-Scale Optimization Problems," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 31 – 54, 2015.