A SCALED GRADIENT PROJECTION METHOD FOR BAYESIAN LEARNING IN DYNAMICAL SYSTEMS *

S. BONETTINI†, A. CHIUSO‡, AND M. PRATO§

Abstract. A crucial task in system identification problems is the selection of the most appropriate model class, and is classically addressed resorting to cross-validation or using order selection criteria based on asymptotic arguments. As recently suggested in the literature, this can be addressed in a Bayesian framework, where model complexity is regulated by few hyperparameters, which can be estimated via marginal likelihood maximization. It is thus of primary importance to design effective optimization methods to solve the corresponding optimization problem. If the unknown impulse response is modeled as a Gaussian process with a suitable kernel, the maximization of the marginal likelihood leads to a challenging nonconvex optimization problem, which requires a stable and effective solution strategy.
In this paper we address this problem by means of a scaled gradient projection algorithm, in which the scaling matrix and the steplength parameter play a crucial role to provide a meaningful solution in a computational time comparable with second order methods. In particular, we propose both a generalization of the split gradient approach to design the scaling matrix in the presence of box constraints, and an effective implementation of the gradient and objective function.
The extensive numerical experiments carried out on several test problems show that our method is very effective in providing in few tenths of a second solutions of the problems with accuracy comparable with state-of-the-art approaches. Moreover, the flexibility of the proposed strategy makes it easily adaptable to a wider range of problems arising in different areas of machine learning, signal processing and system identification.

Key words. System identification, Optimization methods, Regularization, Empirical Bayes method, Marginal likelihood maximization

AMS subject classifications. 65K05, 90C30, 90C90, 93B30

1. Introduction.

System identification is concerned with automatic dynamic model building from measured data. Under this unifying umbrella, this field spans a rather broad spectrum of topics, considering different model classes (linear, hybrid, nonlinear, continuous and discrete time) as well as a variety of methodologies and algorithms, bringing together in a nontrivial way concepts from classical statistics, machine learning and dynamical systems.

The demand for reliable automatic tools for data based modeling of dynamical systems has attracted a considerable interest in the automatic control as well as in the statistics and econometrics communities since the 60′s and has been mainly developed following the parametric maximum likelihood (ML)/prediction error (PE) framework, whose widespread use is to be attributed mainly to its attractive asymptotic statistical properties [34, 46, 13]. Even if we restrict to linear, time-invariant, finite "order" dynamical systems (i.e. systems described by linear differential or difference equations with constant coefficients), where parametric methods are by now well developed and understood (see [34, 46]), it is fair to say that modeling cannot still be considered a

*This work has been partially supported by MIUR (Italian Ministry for University and Research), under the projects FIRB - Futuro in Ricerca 2012, contract RBFR12M3AC, and PRIN 2012, contract 2012MTE38N. The Italian GNCS - INdAM (Gruppo Nazionale per il Calcolo Scientifico - Istituto Nazionale di Alta Matematica) is also acknowledged.

†Dipartimento di Matematica e Informatica, Università di Ferrara, Via Saragat 1, 44121 Ferrara, Italy (silvia.bonettini@unife.it)

‡Dipartimento di Ingegneria dell'Informazione, Università di Padova, Via Gradenigo 6/b, 35131 Padova, Italy

§Dipartimento di Scienze Fisiche, Informatiche e Matematiche, Università di Modena e Reggio Emilia, Via Campi 213/b, 41125 Modena, Italy

"completely automated" task. For instance, in advanced process control applications [53], modeling still is, by far, the most time consuming and costly step [35]. As such, the demand for fast and reliable automated procedures for system identification makes this exciting field still a very active and lively one.

The system identification community, inspired by work in statistics [48, 36], machine learning [44, 49, 4] and signal processing [24, 50], has recently developed and adapted methods based on regularization to jointly perform model selection and estimation in a computationally efficient and statistically robust manner [41, 40, 17, 19, 2, 42, 3, 25]. The main task of regularization is to control model complexity to face the so-called *bias/variance dilemma* [34]. Different regularization strategies have been employed which can be classified in two main classes: regularization induced by smoothness priors (aka Tikhonov regularization, see [30, 22] for early references in the field of dynamical systems) and regularization for selection. This latter is usually achieved by convex relaxation of the ℓ_0 quasi-norm (such as ℓ_1 norm and variations thereof such as sum-of-norms, nuclear norm etc.) or other nonconvex sparsity inducing penalties which can be conveniently derived in a Bayesian framework, aka sparse Bayesian learning (SBL) [36, 49, 50].

In this paper we shall be concerned with regularization induced by smoothness priors; the structure of the chosen prior will bring in features usually encountered in SBL/automatic relevance determination [36, 49] and multiple kernel learning [4, 2]. This makes the algorithms and results in this paper of a rather general interest. In particular we shall address the impulse response estimation problem for single input, single output (SISO) systems described by a convolution equation of the form

$$y(t) = \sum_{k=1}^{\infty} h(k)u(t-k) + e(t), \quad t \in \mathbb{Z}, \tag{1.1}$$

where $y(t) \in \mathbb{R}$ is the *output* signal, $u(t) \in \mathbb{R}$, is the measurable input signal, $h(k)$ is the (unknown) *impulse response* and $e(t)$ is a zero mean white noise signal with unknown variance σ^2. As discussed in [15], the very same framework studied in this paper can be easily adapted to identification of multi input single output (MISO) systems (see also [19]), maintaining the key features which allow the application of the class of algorithms discussed herein.

We shall work in a Bayesian framework, thus modeling the unknown impulse response h (possibly an infinite dimensional object) as a Gaussian process [44] with a suitable (prior) covariance $P(\nu)$ [41, 40, 17] (also known as *kernel*). The chosen covariance is usually described by some unknown hyperparameters ν which give the prior enough flexibility to encode a sufficiently wide class of impulse responses. The number of hyperparameters is typically small as compared to the number of data as well as to the "dimension" of h, which as mentioned above can be infinite dimensional. These hyperparameters can be estimated from data in a variety of ways; empirical evidence as well as some theoretical results [39] support the use of the so-called *marginal likelihood* (i.e. the data likelihood as a function of the unknown hyperparameters, having marginalized the unknown impulse responses from the joint density of data and unknowns) for hyperparameter estimation. This boils down to a challenging optimization problem with the following features:

- it is nonconvex;
- it requires to handle a large number of data (y, u) (also several thousands) even when the number of unknowns (hyperparameters) is not too large (some tenths in most cases);

- the Hessian matrix is, in some cases, quite costly to compute;
- the computation of the objective function and its gradient requires the factorization of matrices which can be extremely ill-conditioned.

Thus, stable and effective algorithms should be designed carefully taking into account the features of the problem. In particular, the simple structure of the constraints, which usually reduce to non-negativity or box, can be exploited by suitable projection methods.

In this paper we propose a scaled gradient projection method for marginal likelihood optimization, whose basic ingredients are the variable stepsize and scaling matrix, which are computed with a negligible computational cost at each gradient projection iteration. The stepsize parameter is chosen according to the Barzilai–Borwein rules, while the scaling matrix is based on a gradient decomposition technique. In spite of the theoretical convergence rate estimate, which in general classifies the classical gradient projection method as linearly convergent, it has been shown in the recent literature that the combination of these choices makes it a very practical, effective and robust numerical tool for several signal and image restoration problems [12, 43, 51, 52]. In this paper we show that, with a suitable choice of the scaling matrix and a careful implementation, the scaled gradient projection method applied to the impulse response estimation problem outperforms some second order state-of-the-art methods, leading to a significant reduction of the computational time.

The plan of the paper is the following. In Section 2 we describe the system identification problem in the framework of the Bayesian approach, deriving the corresponding optimization problem, whose main features are described in Section 3. The proposed optimization method is presented in Section 4, focusing on steplength and scaling matrix selection. In particular, in Section 4.2, we consider the split gradient strategy, which is a state-of-the-art approach for defining the scaling matrix in presence of non-negativity constraints, and we extend it to the more general case of box constraints. Some important implementation issues are discussed in Section 4.3. Finally, the results of an extensive numerical experience are presented in Section 5, showing the effectiveness of the proposed approach on the system identification problem, also with respect to other recent solvers. Our conclusions are offered in Section 6.

Notation. In the following the symbol $\mathrm{Tr}(\cdot)$ indicates the matrix trace and $\det(\cdot)$ the matrix determinant. We shall deal with real random vectors whose (possibly conditional) measure, will always be absolutely continuous w.r.t. the Lebesgue measure and will thus admit a density p. We shall denote with $p(v)$ the density of v (always w.r.t. the Lebesgue measure), $p(v|w)$ the conditional density of v given w. Densities may depend upon some parameters (say x), in which case we shall use subscripts such as $p_x(v)$ or $p_x(v|w)$.

2. Problem statement and model derivation. We shall consider the following problem: given a finite data record $\{u(t), y(t)\}_{t=1}^{N}$ from system (1.1), find an estimator of the impulse response h. This is clearly an ill-posed inverse problem since the unknown h is an infinite dimensional object. As customary in the literature on inverse problems [5] this can be tackled using Tikhonov regularization. Equivalently the (infinite dimensional) unknown h can be modeled as a Gaussian process [44]. We shall follow this second route since it provides a natural way to introduce estimators of the *regularization (hyper)parameters* through the marginal likelihood. We refer the reader to [39] and references therein for some recent work in support of this approach.

In order to avoid theoretical issues related to dealing with infinite dimensional unknowns, chiefly the complication of introducing probability densities for infinite

dimensional objects, the unknown impulse response $\{h(k)\}_{k\in\mathbb{Z}}$ is truncated to a *finite dimensional*, yet arbitrarily long vector. This approximation is always possible (within any arbitrary accuracy) since the impulse response of a finite dimensional linear systems $\{h(k)\}_{k\in\mathbb{Z}^+}$ decays exponentially fast as a function of the index k. In addition, since only N data points are available, no information could ever be obtained from data on the "tail" of the impulse response for $k \geq N$. Thus the model (1.1) can be rewritten as

$$y(t) = \phi(t)^T\theta + e(t), \quad t = n+1, \cdots, N \quad \theta \in \mathbb{R}^n \tag{2.1}$$

where $\phi(t) = (u(t-1), u(t-2), \cdots, u(t-n))^T$ and $\theta \in \mathbb{R}^n$ is the vector whose components are the system impulse response coefficients.
Note that, depending on the "true" underlying system, n can be arbitrarily large, so that estimating θ in the model (2.1) is still an ill-conditioned inverse problem. We stress that this truncation is inessential; by resorting to *Reproducing Kernel Hilbert Space* theory one can deal with the original infinite dimensional problem, see [41, 40].
Equation (2.1) can be represented in matrix form as

$$Y = \Phi\theta + E, \tag{2.2}$$

where $Y = (y(n+1), y(n+2), \ldots, y(N))^T$, $\Phi = (\phi(n+1)^T, \phi(n+2)^T, \ldots, \phi(N)^T)^T$ and $E = (e(n+1), e(n+2), \ldots, e(N))^T$. Since the noise affecting the data is white and Gaussian distributed with zero mean and variance σ^2, then Y conditioned on θ is Gaussian, $Y|\theta \sim \mathcal{N}(\Phi\theta, \sigma^2 I_{N-n})$, and thus has conditional density

$$p_{\sigma^2}(Y|\theta) = (2\pi\sigma^2)^{-(N-n)/2} e^{-\frac{\|Y-\Phi\theta\|_2^2}{2\sigma^2}}.$$

We further model θ as a Gaussian random vector, independent of E, i.e.

$$\theta \sim \mathcal{N}(\theta^{ap}, P(\nu)), \qquad p_\nu(\theta) = (2\pi)^{-n/2} \det(P(\nu))^{-1/2} e^{-\frac{1}{2}(\theta-\theta^{ap})^T P(\nu)^{-1}(\theta-\theta^{ap})},$$

where $P(\nu)$ is the prior covariance parametrized by the hyperparameter vector $\nu \in \mathbb{R}^m$. Typical examples of prior covariance $P(\nu)$ will be given in Section 2.1; suffices here to say that the number of hyper parameters m is typically "small" w.r.t. to the number of data points (from a few units to a few tens). For convenience of notation we shall define $x = (\nu^T, \sigma^2)^T$. From (2.2) it follows that Y is the linear combination of independent Gaussian random vectors and, therefore, the *marginal likelihood* $p_x(Y)$, i.e. the marginal of Y obtained integrating $p_x(Y, \theta) = p_{\sigma^2}(Y|\theta)p_\nu(\theta)$ w.r.t. θ, is still a multivariate normal with mean $\Phi\theta^{ap}$ and covariance matrix

$$\Sigma(x) = \Phi P(\nu)\Phi^T + \sigma^2 I_{N-n}, \quad x = (\nu^T, \sigma^2)^T \tag{2.3}$$

Using Bayes' Theorem we can compute the posterior density of θ given Y

$$p_x(\theta|Y) = \frac{p_{\sigma^2}(Y|\theta)p_\nu(\theta)}{p_x(Y)}, \tag{2.4}$$

which still depends on the unknown hyperparameters x. There are typically two approaches to deal with the unknown hyperparameters x. The first is the so called *Full Bayes approach*: a prior distribution (possibly uninformative) for the hyperparameters is postulated which allows to integrate them out. The second, which we consider in this paper, is the so-called *Empirical Bayes approach* [37]: a point estimate $\widehat{x}$ of the

hyperparameters x is found and then the posterior (2.4) is computed with x fixed to its point estimate $\hat{x}$. In this paper $\hat{x}$ is obtained following the maximum likelihood (ML) approach:

$$\widehat{x} = \underset{x\in\Omega}{\operatorname{argmax}}\ p_x(Y) = \underset{x\in\Omega}{\operatorname{argmin}}\ f(x), \tag{2.5}$$

where Ω is some suitable subset of $\mathbb{R}^{m+1}$ and

$$\begin{aligned} f(x) =& -2\log p_x(Y) - (N-n)\log(2\pi) \\ =& \log\det(\Sigma(x)) + (Y-\Phi\theta^{ap})^T\Sigma(x)^{-1}(Y-\Phi\theta^{ap}). \end{aligned}$$

After a solution $\widehat{x}$ of problem (2.5) has been found, the maximum a posteriori (MAP) estimate $\widehat{\theta}$ of θ, which is equal to the posterior mean for symmetric densities, can be computed:

$$\begin{aligned} \widehat{\theta} :=& \underset{\theta}{\operatorname{argmax}}\ p_{\hat{x}}(\theta|Y) = \underset{\theta}{\operatorname{argmin}}\ -2\log(p_{\hat{x}}(\theta|Y)) \\ =& (\Phi\theta + \hat{\sigma}^2 P(\hat{\nu})^{-1})^{-1}(\Phi^T Y + \hat{\sigma}^2 P(\hat{\nu})^{-1}\theta^{ap}), \end{aligned} \tag{2.6}$$

where in the last equality the fact that $P(\hat{\nu})$ is symmetric has been used.
Unless strong prior knowledge is available, the *a priori* mean θ^{ap} is set to zero, thus estimating the impulse response coefficients θ requires going through the following steps:

1. solve the nonconvex, constrained optimization problem (2.5) where

$$f(x) = Y^T\Sigma(x)^{-1}Y + \log\det(\Sigma(x)) \tag{2.7}$$

   and $\Sigma(x) \in \mathbb{R}^{(N-n)\times(N-n)}$ is defined in (2.3)
2. compute the corresponding impulse response coefficients setting $\theta^{ap} = 0$ in (2.6):

$$\hat{\theta} = (\Phi\Phi^T + \hat{\sigma}^2 P(\hat{\nu})^{-1})^{-1}\Phi^T Y\ .$$

2.1. Kernel matrices. Several kernel matrices have been introduced in the recent years to model impulse responses of dynamical systems. Perhaps the major breakthrough has been the observation that the kernel has to capture structural properties of dynamical systems [41, 21, 18], such as the fact that for linear systems described by difference/differential equations, the impulse response is a linear combination of exponentially decaying functions [29]. In order to do so, the seminal paper [41] has introduced the family of *stable-spline* kernels; the most used kernels in this family are the *stable-spline kernel of order 1*, called also tuned/correlated (TC) kernel [17]:

$$P^{TC}_{k,j}(\nu) = c\cdot\min(\mu^k, \mu^j), \quad k,j = 1,...,n, \tag{2.8a}$$

and the *stable-spline kernel of order 2*:

$$P^{SS}_{k,j}(\nu) = c\begin{cases} \frac{\mu^{2k}}{2}\left(\mu^j - \frac{\mu^k}{3}\right) & k \geq j \\ \frac{\mu^{2j}}{2}\left(\mu^k - \frac{\mu^j}{3}\right) & k < j \end{cases}, \quad k,j = 1,...,n, \tag{2.8b}$$

where $\nu = (c, \mu)^T$, $c \geq 0$, $\mu \in [0, 1)$. Soon after [41] several other papers appeared where different families of kernels have been introduced [41, 40, 17, 42], among which the diagonal/correlated (DC) kernel

$$P^{DC}_{k,j}(\nu) = c\mu^{(k+j)/2}\rho^{|k-j|}, \quad k, j = 1, ..., n, \tag{2.8c}$$

where $\nu = (c, \mu, \rho)^T$, $c \geq 0$, $\mu \in [0, 1)$, $\rho \in (-1, 1)$. As discussed in [15], and further elaborated upon in [18], these kernels alone may not well represent impulse responses obtained by linear combination of exponentially decaying functions when the decay rates vary widely; see e.g. Example 2.1 in [15]. For this reason the paper [15] introduces a family of *multiple kernels*, which take the form

$$P(\nu) = \sum_{i=1}^{m} \nu_i P_i, \tag{2.9}$$

where $P_i \in \mathbb{R}^{n\times n}$ are given fixed symmetric and positive semidefinite matrices and the coefficients $\nu_i \geq 0$ $(i = 1, \ldots, m)$ play the role of scale factors.
Here, as in [15], the "alphabet" of kernels P_i is chosen from one of the kernels (2.8) over a suitable grid of hyperarameters (ρ, μ, c). All the kernel choices listed above correspond to an optimization problem (2.5)–(2.7) with box-type constraints.

Remark. In our approach, σ^2 is treated as an optimization variable as suggested in [15]. As an alternative, the noise variance σ^2 can be estimated from the data using a high order (and thus low bias) ARX model (linear regressions) as suggested in [27, 34]; some care needs to be taken to avoid overfitting. In this case only ν, which corresponds to the first m components of x, would have to be optimized using the marginal likelihood.

3. Problem features. In this section we describe some properties of the optimization problem (2.5)–(2.7). We first need to introduce some notation, defining the objective function as

$$f(x) = f_0(x) + f_1(x), \tag{3.1}$$

with

$$f_0(x) = Y^T\Sigma(x)^{-1}Y, \qquad f_1(x) = \log\det(\Sigma(x)), \quad \forall x \in \mathbb{R}^{m+1}.$$

The i–th component of the gradient of $f_0(x)$ and $f_1(x)$ can be expressed as

$$\nabla_i f_0(x) = -Y^T\Sigma(x)^{-1}\frac{\partial\Sigma(x)}{\partial x_i}\Sigma(x)^{-1}Y \tag{3.2}$$

$$\nabla_i f_1(x) = \mathrm{Tr}\left(\Sigma(x)^{-1}\frac{\partial\Sigma(x)}{\partial x_i}\right) \tag{3.3}$$

where

$$\frac{\partial\Sigma(x)}{\partial x_i} = \begin{cases} \Phi\dfrac{\partial P(\nu)}{\partial\nu_i}\Phi^T, & i = 1, ..., m \\ I_{N-n}, & i = m+1. \end{cases}$$

Moreover, the element (i,j) of the Hessian matrix $\nabla^2 f(x)$, for $i,j=1,...,m+1$, is given by $\nabla^2_{ij} f(x) = \nabla^2_{ij} f_0(x) + \nabla^2_{ij} f_1(x)$, where

$$\nabla^2_{ij} f_0(x) = Y^T \Sigma(x)^{-1} \left(\frac{\partial \Sigma(x)}{\partial x_j} \Sigma(x)^{-1} \frac{\partial \Sigma(x)}{\partial x_i} - \frac{\partial^2 \Sigma(x)}{\partial x_i \partial x_j} + \frac{\partial \Sigma(x)}{\partial x_i} \Sigma(x)^{-1} \frac{\partial \Sigma(x)}{\partial x_j} \right) \Sigma(x)^{-1} Y,$$

$$\nabla^2_{ij} f_1(x) = \mathrm{Tr} \left(-\Sigma(x)^{-1} \frac{\partial \Sigma(x)}{\partial x_j} \Sigma(x)^{-1} \frac{\partial \Sigma(x)}{\partial x_i} + \Sigma(x)^{-1} \frac{\partial^2 \Sigma(x)}{\partial x_i \partial x_j} \right).$$

When $P(\nu)$ is the multiple kernel (2.9), then $f_0(x)$ and $f_1(x)$ are convex and concave, respectively (see [14]). In this case, since $\frac{\partial P(\nu)}{\partial \nu_i} = P_i$, $i = 1, \ldots, m$, is positive semidefinite, the gradient of the objective function has the following interesting property

$$\nabla f_0(x) \leq 0, \quad \nabla f_1(x) > 0, \quad \forall x \in \mathbb{R}^m \tag{3.4}$$

when $\sigma^2 > 0$. The first inequality is straightforward since $\Sigma(x)$ is positive definite and $\Phi P_i \Phi^T$ is positive semidefinite for all $i = 1, ..., m$, while the second one is a direct consequence of Lemma II.1 in [32]. Moreover, the objective function satisfies

$$\lim_{t \to +\infty} f(tx) = +\infty \tag{3.5}$$

for all $x > 0$, where $t \in \mathbb{R}$, so that its level sets are bounded, see [15, §III.B].
Observe that (3.4) and (3.5) are, in general, not true when the kernel $P(\nu)$ nonlinearly depends on its parameter ν, as in (2.8); in this case it is not even ensured that $f_0(x)$ and $f_1(x)$ are convex and concave, respectively.

4. Optimization method. In this section we describe the optimization method we propose to solve (2.5). We focus on first order methods based on gradient projection, which are particularly suited when the constraints are simple. The main objection in the use of first order methods is that their convergence rate is, in general, linear. However, introducing some clever choices to define the descent direction, they are able to compute a medium accuracy solution with a small number of iterations. Such acceleration strategies are implemented in the scaled gradient projection (SGP) method [12], which applies to any problem of the form

$$\min_{x \in \Omega} f(x), \tag{4.1}$$

where $\Omega \subseteq \mathbb{R}^p$ is a closed convex set, and employs a double scaling of the negative gradient direction through a positive scalar parameter α_k and a positive definite matrix D_k, both iteration dependent. The general scheme of SGP is summarized in Algorithm 1. To motivate the introduction of the scaling matrix, one can think, for example, to the Newton's method, which actually scales the gradient direction with the inverse Hessian, while other practical choices for α_k and D_k are described in the following sections.
In order to define a descent direction at Step 3, i.e. a vector $\Delta x^{(k)}$ such that $\nabla f(x^{(k)})^T \Delta x^{(k)} < 0$, the projection at Step 2 is computed with respect to the norm induced by the inverse of the scaling matrix D_k, i.e. it is defined as

$$\Pi_{\Omega, D_k^{-1}}(z) = \arg\min_{x \in \Omega} (x - z)^T D_k^{-1} (x - z).$$

Thus, even if any positive definite matrix is allowed, the most practical choice for D_k consists in a diagonal matrix with positive diagonal entries. Once defined the

descent direction at Step 3, an Armijo backtracking loop computes the steplength λ_k to guarantee the sufficient decrease of the objective function [7, §2.2.1], i.e.

$$f(x^{(k)} + \lambda_k \Delta x^{(k)}) \leq f(x^{(k)}) + \beta \lambda_k \nabla f(x^{(k)})^T \Delta x^{(k)}. \tag{4.2}$$

Algorithm 1 Scaled gradient projection (SGP) method

Choose the starting point $x^{(0)} \in \Omega$, set the parameters $\beta, \gamma \in (0,1)$, $0 < \alpha_{min} < \alpha_{max}$, $0 < L_{min} < L_{max}$ and fix a positive integer M.

FOR $k = 0, 1, 2, \ldots$ DO THE FOLLOWING STEPS:
- STEP 1. Choose the parameter $\alpha_k \in [\alpha_{min}, \alpha_{max}]$ and the diagonal scaling matrix D_k such that $L_{min} \leq (D_k)_{ii} \leq L_{max}$, $i = 1, \ldots, p$;
- STEP 2. Projection: $z^{(k)} = \Pi_{\Omega, D_k^{-1}}(x^{(k)} - \alpha_k D_k \nabla f(x^{(k)}))$;
- STEP 3. Descent direction: $\Delta x^{(k)} = z^{(k)} - x^{(k)}$;
- STEP 4. Set $\lambda_k = 1$;
- STEP 5. Backtracking loop:
  - IF $f(x^{(k)} + \lambda_k \Delta x^{(k)}) \leq f(x^{(k)}) + \beta \lambda_k \nabla f(x^{(k)})^T \Delta x^{(k)}$ THEN
    - go to Step 6;
  - ELSE
    - set $\lambda_k = \gamma \lambda_k$ and go to Step 5.
  - ENDIF
- STEP 6. Set $x^{(k+1)} = x^{(k)} + \lambda_k \Delta x^{(k)}$.

END

The Armijo condition (4.2) is crucial for the proof of the following general convergence result, which can be found in [12, Theorem 2.1] (see also [8, Theorem 4.2]).

THEOREM 1 *Let $\{x^{(k)}\}$ be the sequence generated by applying the SGP algorithm to problem (4.1). Then, every accumulation point x^* of the sequence $\{x^{(k)}\}$ is a constrained stationary point, that is*

$$\nabla f(x^*)^T (x - x^*) \geq 0, \quad \forall\, x \in \Omega.$$

We remark that all the iterates generated by SGP belong to the set $\Omega_0 = \{x \in \Omega : f(x) \leq f(x^{(0)})\}$. When $f(x)$ is defined as in (2.7) and $P(\nu)$ has the form (2.9), we recall that (3.5) holds: this implies that Ω_0 is bounded and, thus, the sequence $\{x^{(k)}\}$ admits at least one limit point.
Observe that Theorem 1 holds without convexity assumptions and for any bounded choice of the stepsize α_k and scaling matrix D_k. This freedom of choice can be exploited to significantly improve the practical performances of SGP. In the following we describe the main strategies for the selection of these parameters.

4.1. Stepsize selection rules. Once a scaling matrix D_k has been defined, a well performing choice of the stepsize parameter is the variant of the Barzilai–Borwein rules proposed in [12]. The rationale behind this idea consists in computing the stepsize α_k so that the matrix $\alpha_k D_k$ approximates in a quasi–Newton sense the inverse Hessian of the objective function. In practice, α_k is computed as the solution of one of the following minimization problems:

$$\min_{\alpha \in \mathbb{R}} \|\alpha D_k r^{(k-1)} - w^{(k-1)}\|, \qquad \min_{\alpha \in \mathbb{R}} \|r^{(k-1)} - (\alpha D_k)^{-1} w^{(k-1)}\|, \tag{4.3}$$

where $r^{(k-1)} = x^{(k)} - x^{(k-1)}$ and $w^{(k-1)} = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$. The solutions of the minimum problems in (4.3) are given by

$$\alpha_k^{BB1} = \frac{{r^{(k-1)}}^T D_k^{-1} D_k^{-1} r^{(k-1)}}{{r^{(k-1)}}^T D_k^{-1} w^{(k-1)}}, \quad \alpha_k^{BB2} = \frac{{r^{(k-1)}}^T D_k w^{(k-1)}}{{w^{(k-1)}}^T D_k D_k w^{(k-1)}}, \tag{4.4}$$

and, from the computational point of view, they can be computed in $\mathcal{O}(p)$ operations, where p is the number of variables. Actually, the scalar products ${r^{(k-1)}}^T D_k^{-1} w^{(k-1)}$ and ${r^{(k-1)}}^T D_k w^{(k-1)}$ may be negative, leading to negative values in formula (4.3). If this occurs, we set $\alpha_k^{BB1} = \alpha_{max}$ and $\alpha_k^{BB2} = \alpha_{max}$ respectively: this choice is based on the observation that the $(k-1)$-th iterate lies in a region where the objective function might have a negative curvature (if $D_k = I$ and f is convex, both the scalar products are non-negative). Thus, taking a long step along the negative gradient could help to go away from a nonoptimal stationary point.
It is well known by the recent literature that the best performances are achieved by adaptively alternating the two rules, with a thresholding to keep the stepsize within the prefixed interval $[\alpha_{min}, \alpha_{max}]$ (see Step 1 in Algorithm 1). In our implementation we adopt the alternation strategy detailed below:

IF ${r^{(k-1)}}^T D_k^{-1} w^{(k-1)} \leq 0$ THEN
  $\alpha_k^{(1)} = \alpha_{max}$;
ELSE
  $\alpha_k^{(1)} = \min\left\{\alpha_{max}, \max\left\{\alpha_{min}, \alpha_k^{BB1}\right\}\right\}$;
ENDIF
IF ${r^{(k-1)}}^T D_k w^{(k-1)} \leq 0$ THEN
  $\alpha_k^{(2)} = \alpha_{max}$;
ELSE
  $\alpha_k^{(2)} = \min\left\{\alpha_{max}, \max\left\{\alpha_{min}, \alpha_k^{BB2}\right\}\right\}$;
ENDIF
IF $\alpha_k^{(2)}/\alpha_k^{(1)} \leq \tau_k$ THEN
  $\alpha_k = \min\left\{\alpha_j^{(2)},\ j = \max\left\{1, k - M_\alpha\right\}, \ldots, k\right\}$; $\tau_{k+1} = \tau_k \cdot 0.9$;
ELSE
  $\alpha_k = \alpha_k^{(1)}$; $\tau_{k+1} = \tau_k \cdot 1.1$;
ENDIF

where M_α is a prefixed non-negative integer and $\tau_1 \in (0, 1)$. The alternating rule described above has been proposed for unconstrained, strictly convex quadratic problems in [26], where the authors investigate the related theoretical properties and numerically show that this alternation of the two BB rules allows to better capture the spectral properties of the Hessian matrix. Successively, an adaptation of the alternating rule in [26] has been proposed in [12] and employed also in several applications of SGP to different convex, nonlinear, constrained problems [8, 9, 10, 11]. In this paper we adopt the same rule also for the nonlinear, nonconvex, constrained problem described in Section 3.

4.2. Choice of the scaling matrix. Unlike the stepsize selection rules, the scaling matrix choice is strictly related to the specific structure of problem (4.1) and it depends on both the objective function and the constraints. In particular, the constraints of problem (2.5) are lower bounds when $P(\nu)$ is the multiple kernel (2.9) or box constraints when the kernels (2.8) are selected.
In this section we review the split gradient idea described in [6, 31] for lower bound

constraints and we extend such approach to general box constraints. To introduce the split gradient idea, we consider first the non-negatively constrained problem

$$\min_{x\geq 0} f(x) \tag{4.5}$$

whose first order optimality conditions are given by

$$x\nabla f(x) = 0; \quad x \geq 0; \quad \nabla f(x) \geq 0, \tag{4.6}$$

where the equality and inequalities are componentwise. If the gradient of $f(x)$ admits a decomposition like the following one

$$\nabla f(x) = V(x) - U(x) \ \text{ with } V(x) > 0,\ U(x) \geq 0, \tag{4.7}$$

then equality (4.6) writes also as the fixed point equation $x = xU(x)/V(x)$. This formulation is related to the corresponding fixed point method

$$x^{(k+1)} = x^{(k)} \frac{U(x^{(k)})}{V(x^{(k)})}, \tag{4.8}$$

whose convergence properties are not well studied, but which has the capability to preserve positivity when the initial point is positive and $U(x) > 0$ whenever $x > 0$. Several methods in signal and image processing (e.g. Lucy-Richardson/expectation minimization [45], iterative space reconstruction algorithm [20]) and statistical learning (Lee-Seung algorithm for non-negative matrix factorization [33]) actually have exactly this multiplicative form (see also [28, 38]).
With a simple algebra the multiplicative method (4.8) results in

$$x^{(k+1)} = \ x^{(k)} \frac{U(x^{(k)}) - V(x^{(k)}) + V(x^{(k)})}{V(x^{(k)})} = \ x^{(k)} - \frac{x^{(k)}}{V(x^{(k)})} \nabla f(x^{(k)}),$$

which corresponds to a scaled gradient iteration. These considerations suggest to define the scaling matrix for SGP as

$$(D_k)_{ii} = \min\left(\max\left(L_{min}, \frac{x_i^{(k)}}{V_i(x^{(k)})}\right), L_{max}\right). \tag{4.9}$$

More in general, for lower bound constraints $x \geq l$, $l \in \mathbb{R}^p$, the following scaling matrix

$$(D_k)_{ii} = \min\left(\max\left(L_{min}, \frac{x_i^{(k)} - l_i}{V_i(x^{(k)})}\right), L_{max}\right) \tag{4.10}$$

can be motivated using similar arguments as above.
This choice of the scaling matrix, combined with a suitable choice of the stepsize α_k, leads the SGP method to very good performances on ill-posed/ill-conditioned inverse problems approached by the Bayesian paradigm as convex, non-negatively constrained optimization problems [12, 43, 51, 52].
We propose to use the scaling (4.10) also on problem (2.5) with the multiple kernel (2.9), even if the objective function is nonconvex. In this case, recalling (3.4), the gradient of the objective function has the natural decomposition (4.7) with

$$V(x) = \nabla f_1(x) \quad \text{and} \quad U(x) = -\nabla f_0(x). \tag{4.11}$$

4.2.1. Gradient splitting strategy for box constraints. In this section we consider a box constrained problem

$$\min_{l\le x\le u} f(x) \tag{4.12}$$

where $l, u \in \mathbb{R}^p \cup \{+\infty, -\infty\}$ ($l_i = -\infty$, $u_i = +\infty$ means that x_i is unbounded below or above respectively), and we propose a scaling strategy also for this case. Driven by the considerations made in the previous section, the generalization to box constraints consists in finding a positive diagonal scaling matrix D_k such that $x^{(k)} - D_k\nabla f(x^{(k)})$ is feasible, i.e.

$$l_i \le x_i^{(k)} - (D_k)_{ii}\nabla_i f(x^{(k)}) \le u_i, \quad i = 1, \dots, p.$$

Then, to design an appropriate scaling, we should consider the sign of the gradient at the current iterate to devise which constraints could be violated taking a step along the negative gradient direction. To this end, we define the following sets of indices

$$\mathcal{I}_1 = \{i : l_i > -\infty \text{ and } u_i < +\infty\}, \quad \mathcal{I}_2 = \{i : l_i = -\infty \text{ and } u_i < +\infty\},$$
$$\mathcal{I}_3 = \{i : l_i > -\infty \text{ and } u_i = +\infty\}, \quad \mathcal{I}_4 = \{i : l_i = -\infty \text{ and } u_i = +\infty\},$$

to identify which variables are bounded below and/or above and which are unbounded. Then, we define the following vector

$$\tilde{d}_i(x^{(k)}) = \begin{cases} \dfrac{u_i - x_i^{(k)}}{U_i(x^{(k)})} & \text{if } i \in \mathcal{I}_1 \text{ and } \nabla_i f(x^{(k)}) \le 0 \text{ or } i \in \mathcal{I}_2 \\ \dfrac{x_i^{(k)} - l_i}{V_i(x^{(k)})} & \text{if } i \in \mathcal{I}_1 \text{ and } \nabla_i f(x^{(k)}) > 0 \text{ or } i \in \mathcal{I}_3 \\ 1 & \text{if } i \in \mathcal{I}_4 \end{cases} \tag{4.13}$$

based on a gradient decomposition of the form

$$\nabla f(x) = V(x) - U(x), \quad V(x) > 0,\ U_i(x) > 0. \tag{4.14}$$

Indeed, $\nabla_i f(x^{(k)}) \le 0$ implies $0 < V_i(x^{(k)}) \le U_i(x^{(k)})$ and, as a consequence, $x_i^{(k)} \le x_i^{(k)} - \tilde{d}_i(x^{(k)})\nabla_i f(x^{(k)}) \le u_i$. On the other side, $\nabla_i f(x^{(k)}) > 0$ if and only if $V_i(x^{(k)}) \ge U_i(x^{(k)}) > 0$, which yields $l_i \le x_i^{(k)} - \tilde{d}_i(x^{(k)})\nabla_i f(x^{(k)}) \le x_i^{(k)}$.
Finally, the diagonal entries of the scaling matrix are defined as

$$(D_k)_{ii} = \min\left(\max\left(L_{min}, \tilde{d}_i(x^{(k)})\right), L_{max}\right). \tag{4.15}$$

For an objective function of the form $f(x) = f_0(x) + f_1(x)$ a possible general rule to define $U(x^{(k)})$ and $V(x^{(k)})$ in (4.13) can be devised in the following way. When $\nabla_i f(x) > 0$, then $\nabla_i f_1(x) > -\nabla_i f_0(x)$ and we define

$$V_i(x) = \begin{cases} \nabla_i f_0(x) & \text{if } \nabla_i f_1(x) < 0 \\ \nabla_i f_1(x) & \text{if } \nabla_i f_1(x) \ge 0 \text{ and } \nabla_i f_0(x) < 0 \\ \nabla_i f(x) + \zeta & \text{otherwise} \end{cases}, \tag{4.16}$$
$$U_i(x) = V_i(x) - \nabla_i f(x)$$

for some $\zeta > 0$. Similarly, when $\nabla_i f(x) \leq 0$, then $\nabla_i f_1(x) \leq -\nabla_i f_0(x)$ and we set

$$U_i(x) = \begin{cases} -\nabla_i f_1(x) & \text{if } \nabla_i f_0(x) > 0 \\ -\nabla_i f_0(x) & \text{if } \nabla_i f_0(x) < 0 \text{ and } \nabla_i f_1(x) > 0 \\ \zeta - \nabla_i f(x) & \text{otherwise} \end{cases}, \tag{4.17}$$
$$V_i(x) = \nabla_i f(x) + U_i(x).$$

It is easy to verify that definitions (4.16) and (4.17) lead to a gradient decomposition with the property (4.14). Moreover, this choice of the scaling matrix reduces to (4.10) in presence of lower bounds only.
We adopt the scaling strategy (4.15) associated to the decomposition (4.16)–(4.17) for problem (2.5) when the kernel is given by (2.8).

4.3. Algorithm implementation and complexity. Each SGP iteration requires the objective function (3.1) and gradient (3.2)–(3.3) at the current point $x^{(k)} = ({\nu^{(k)}}^T, \sigma_k^2)^T$, which is the more relevant computational burden of the whole algorithm. If the Armijo condition (4.2) is not satisfied with $\lambda_k = 1$, more function evaluations are needed.
Thus, the practical performances of the algorithm also relies on the implementation of the gradient and objective function computation. On the other side we should take into account the severe ill-conditioning possibly affecting the matrices $P(\nu^{(k)})$ and $\Sigma(x^{(k)})$. In our implementation we implicitly assume that $n \ll N$, which is quite realistic, and we devise an algorithm for the computation of $f(x^{(k)})$ and $\nabla f(x^{(k)})$ with complexity $\mathcal{O}(n^3)$ which is detailed below.
We consider the approach proposed in [16] for objective function and gradient evaluations, which is based on the Cholesky factorization of $P(\nu^{(k)}) = \mathcal{L}_k \mathcal{L}_k^T$, at a cost of $\mathcal{O}(n^3)$. Then, the Cholesky factorization of the matrix $\sigma_k^2 I_n + \mathcal{L}_k^T \Phi^T \Phi \mathcal{L}_k = S_k S_k^T$ is also computed. Finally, the objective function is evaluated with the formula

$$f(x^{(k)}) = (\|Y\|^2 - \|S_k^{-1} \mathcal{L}_k^T \Phi^T Y\|^2)/\sigma_k^2 + (N-n)\log \sigma_k^2 + 2\log |S_k|. \tag{4.18}$$

The Cholesky factors S_k and $\mathcal{L}_k$ can be reused for the computation of $\Sigma(x^{(k)})^{-1}$ and, then, of the gradient as follows. Omitting for simplicity the dependency of $P(\nu^{(k)})$ and $\Sigma(x^{(k)})$ from $\nu^{(k)}$ and $x^{(k)}$ and applying the Sherman-Morrison-Woodbury formula we obtain

$$\Sigma^{-1} = (\sigma_k^2 I_{N-n} + \Phi P \Phi^T)^{-1} = \frac{1}{\sigma_k^2} I_{N-n} - \frac{1}{\sigma_k^2} \Phi (\sigma_k^2 P^{-1} + \Phi^T \Phi)^{-1} \Phi^T.$$

Finally, by observing that

$$(\sigma_k^2 P^{-1} + \Phi^T \Phi)^{-1} = (\sigma_k^2 \mathcal{L}_k^{-T} \mathcal{L}_k^{-1} + \Phi^T \Phi)^{-1} = \mathcal{L}_k S_k^{-T} S_k^{-1} \mathcal{L}_k^T$$

it follows that

$$\Sigma^{-1} = \frac{1}{\sigma_k^2} I_{N-n} - \frac{1}{\sigma_k^2} \Phi \mathcal{L}_k S_k^{-T} S_k^{-1} \mathcal{L}_k^T \Phi^T. \tag{4.19}$$

Taking into account of (4.19), if we set

$$\tilde{\Phi} = \Phi^T \Phi, \quad \tilde{Y} = \Phi^T Y, \quad Z_k = \mathcal{L}_k S_k^{-T} S_k^{-1} \mathcal{L}_k^T, \quad M_k = \Phi^T \Sigma^{-1} \Phi = \tilde{\Phi} - \tilde{\Phi} Z_k \tilde{\Phi}, \tag{4.20}$$

then (3.2) can be computed as

$$\nabla_i f_0(x^{(k)}) = q^T \frac{\partial P}{\partial \nu_i} q, \quad \text{with } q = \Phi\Sigma^{-1}Y = \frac{1}{\sigma_k^2}(I_n - \tilde{\Phi}Z_k)\tilde{Y} \tag{4.21a}$$

for $i = 1, ..., m$ and

$$\begin{aligned}
\nabla_{m+1} f_0(x^{(k)}) &= \|\Sigma^{-1}Y\|^2 \\
&= \|Y\|^2/\sigma_k^4 - 2Y^T\Phi\mathcal{L}_k S_k^{-T} S_k^{-1}\mathcal{L}_k^T\Phi^T Y/\sigma_k^4 + \\
&\quad + Y^T\Phi\mathcal{L}_k S_k^{-T} S_k^{-1}(\mathcal{L}_k^T\Phi^T\Phi\mathcal{L}_k) S_k^{-T} S_k^{-1}\mathcal{L}_k^T\Phi^T Y/\sigma_k^4 \\
&= \|Y\|^2/\sigma_k^4 - 2Y^T\Phi\mathcal{L}_k S_k^{-T} S_k^{-1}\mathcal{L}_k^T\Phi^T Y/\sigma_k^4 + \\
&\quad + Y^T\Phi\mathcal{L}_k S_k^{-T} S_k^{-1}(S_k S_k^T - \sigma_k^2 I_n) S_k^{-T} S_k^{-1}\mathcal{L}_k^T\Phi^T Y/\sigma_k^4 \\
&= \|Y\|^2/\sigma_k^4 - \|S_k^{-1}\mathcal{L}_k^T\tilde{Y}\|^2/\sigma_k^4 - \|S_k^{-T}S_k^{-1}\mathcal{L}_k^T\tilde{Y}\|^2/\sigma_k^2.
\end{aligned} \tag{4.21b}$$

On the other side, recalling (3.3), we have

$$\begin{aligned}
\nabla_i f_1(x^{(k)}) &= \mathrm{Tr}\left(\Sigma^{-1}\Phi\frac{\partial P}{\partial \nu_i}\Phi^T\right) = \mathrm{Tr}\left(\Phi^T\Sigma^{-1}\Phi\frac{\partial P}{\partial \nu_i}\right) \\
&= \frac{1}{\sigma_k^2}\mathrm{Tr}\left(M_k\frac{\partial P}{\partial \nu_i}\right), \quad i = 1, ..., m
\end{aligned} \tag{4.22a}$$

$$\begin{aligned}
\nabla_{m+1} f_1(x^{(k)}) &= \mathrm{Tr}(I_{N-n} - \Phi\mathcal{L}_k S_k^{-T} S_k^{-1}\mathcal{L}_k\Phi^T)/\sigma_k^2 \\
&= \left(\mathrm{Tr}(I_{N-n}) + \mathrm{Tr}(\mathcal{L}_k\Phi^T\Phi\mathcal{L}_k S_k^{-T} S_k^{-1})\right)/\sigma_k^2 \\
&= \left(\mathrm{Tr}(I_{N-n}) + \mathrm{Tr}((S_k S_k^T - \sigma_k^2 I_n) S_k^{-T} S_k^{-1})\right)/\sigma_k^2 \\
&= (N-2n)/\sigma_k^2 + \mathrm{Tr}(S_k^{-T} S_k^{-1}).
\end{aligned} \tag{4.22b}$$

The main difference between our approach for gradient computation and the analogous one described in [16, Section 5] is that we avoid to explicitly compute the matrix $P^{-1} = \mathcal{L}_k^{-T}\mathcal{L}_k^{-1}$, which is very ill-conditioned.
The previous formulae for gradient computation clearly hold when $P(\nu^{(k)})$ does not reduce to the zero matrix: since the latter case can occur at some iteration k, for sake of completeness we report the whole procedure in Algorithm 2.

Remark. The computation of the Hessian matrix can also be performed with a complexity of $\mathcal{O}(n^3)$, without need of further factorizations but with at least m additional matrix-matrix products of size $n \times n$, as detailed in the following. Developing the formulae for the entries of the Hessian matrix given in Section 3, for $i, j = 1, ..., m$, we can set $\nabla^2_{ij} f_0(x) = a_{ij} - b_{ij} + a_{ji}$, where

$$\begin{aligned}
a_{ij} &= Y^T\Sigma^{-1}\Phi\frac{\partial P}{\partial \nu_j}\Phi^T\Sigma^{-1}\Phi\frac{\partial P}{\partial \nu_i}\Phi^T\Sigma^{-1}Y = q^T\frac{\partial P}{\partial \nu_j}M_k\frac{\partial P}{\partial \nu_i}q \\
b_{ij} &= Y^T\Sigma^{-1}\frac{\partial^2\Sigma}{\partial x_i \partial x_j}\Sigma^{-1}Y = q^T\frac{\partial^2 P}{\partial \nu_i \partial \nu_j}q
\end{aligned}$$

with q defined as in (4.21a). Moreover we have

$$\begin{aligned}
\nabla^2_{i,m+1} f_0(x^{(k)}) &= \tilde{q}\frac{\partial P}{\partial \nu_i}q, \quad i = 1, ..., m \\
\nabla^2_{m+1,m+1} f_1(x^{(k)}) &= Y^T\Sigma^{-3}Y \\
&= \frac{1}{\sigma_k^6}\left(\|Y\|^2 - 3\tilde{Y}Z_k\tilde{Y} + 3\tilde{Y}^T Z_k\tilde{\Phi}Z_k\tilde{Y} - \tilde{Y}^T Z_k\tilde{\Phi}Z_k\tilde{\Phi}Z_k\tilde{Y}\right)
\end{aligned}$$

Algorithm 2 Objective function and gradient evaluation

Preprocessing: compute $\|Y\|^2$, $\tilde{\Phi} = \Phi^T\Phi$ and $\tilde{Y} = \Phi^T Y$.

FOR ANY $x^{(k)} = ({\nu^{(k)}}^T, \sigma_k^2)^T$, $k = 1, 2, ...$ DO THE FOLLOWING STEPS:

STEP 1. Compute $P(\nu^{(k)})$.

STEP 2. IF $P(\nu^{(k)}) \neq 0$ THEN

2.1 Compute the Cholesky factorization $P(\nu^{(k)}) = \mathcal{L}_k\mathcal{L}_k^T$;

2.2 Compute $Q_k = \sigma_k^2 I_n + \mathcal{L}_k^T\tilde{\Phi}\mathcal{L}_k$;

2.3 Compute the Cholesky factorization $Q_k = S_k S_k^T$, Z_k and M_k as in (4.20);

2.4 Compute $f(x^{(k)})$ by formula (4.18);

2.5 Compute $\nabla_i f_0(x^{(k)})$ and $\nabla_i f_1(x^{(k)})$ by means of (4.21) and (4.22) for $i = 1, ..., m+1$.

ELSE

2.6 Compute $f(x^{(k)}) = \frac{\|Y\|^2}{\sigma_k^2} + (N-n)\log(\sigma_k^2)$;

2.7 Compute $\nabla_i f_1(x^{(k)}) = \frac{1}{\sigma_k^2}\mathrm{Tr}\left(\tilde{\Phi}\frac{\partial P(\nu^{(k)})}{\partial x_i}\right)$ and $\nabla_i f_0(x^{(k)}) = -\frac{1}{\sigma_k^4}\tilde{Y}^T\frac{\partial P(\nu^{(k)})}{\partial x_i}\tilde{Y}$ for $i = 1, ..., m$; $\nabla_{m+1} f_1(x^{(k)}) = (N-n)/\sigma_k^2$; $\nabla_{m+1} f_0(x^{(k)}) = -\|Y\|^2/\sigma_k^4$;

ENDIF

STEP 3. Compute $\nabla f(x^{(k)}) = \nabla f_0(x^{(k)}) + \nabla f_1(x^{(k)})$.

END

where $\tilde{q} = \Phi^T\Sigma^{-2}Y = (I_n - \tilde{\Phi}Z_k)^2\tilde{Y}/\sigma_k^4$. As concerns the Hessian of f_1, exploiting the matrix trace properties, for $i, j = 1, ..., m$ we have $\nabla^2_{ij} f_1(x) = g_{ij} - e_{ij}$, where

$$e_{ij} = \mathrm{Tr}\left(\Sigma^{-1}\frac{\partial\Sigma}{\partial x_j}\Sigma^{-1}\frac{\partial\Sigma}{\partial x_i}\right) = \mathrm{Tr}\left(M_k\frac{\partial P}{\partial\nu_j}M_k\frac{\partial P}{\partial\nu_i}\right)$$
$$g_{ij} = \mathrm{Tr}\left(\Sigma^{-1}\frac{\partial^2\Sigma}{\partial x_i\partial x_j}\right) = \mathrm{Tr}\left(M_k\frac{\partial^2 P}{\partial\nu_i\partial\nu_j}\right)$$

with M_k defined as in (4.20), and

$$\nabla^2_{i,m+1} f_1(x^{(k)}) = \mathrm{Tr}\left(\Phi^T\Sigma^{-2}\Phi\frac{\partial P}{\partial\nu_i}\right) = \mathrm{Tr}\left(\tilde{M}_k\frac{\partial P}{\partial\nu_i}\right), \quad i = 1, ..., m$$
$$\nabla^2_{m+1,m+1} f_1(x^{(k)}) = \mathrm{Tr}(\Sigma^{-2}) = \frac{1}{\sigma_k^4}\left(N - n - 2\mathrm{Tr}(\tilde{\Phi}Z_k) + \mathrm{Tr}(\tilde{\Phi}Z_k\tilde{\Phi}Z_k)\right)$$

with $\tilde{\Phi}$, Z_k defined as in (4.20) and $\tilde{M}_k = (I_n - \tilde{\Phi}Z_k)\tilde{\Phi}/\sigma_k^4$. Observe that e_{ij} requires the explicit computation of the matrices $M_k\frac{\partial P}{\partial\nu_i}$, $i = 1, ..., m$, with a complexity of $\mathcal{O}(mn^3)$. For the multiple kernel (2.9), where m typically is of order of tenths, the Hessian computation is a quite expensive task. It is worth stressing that the computation of the matrix product $M_k\frac{\partial P}{\partial\nu_i}$ is not needed for (4.22a), since the well known formula $\mathrm{Tr}(AB) = \mathrm{vec}(A)^T\mathrm{vec}(B)$, where $\mathrm{vec}(\cdot)$ indicates the vectorization of a matrix by stacking its elements columnwise, can be applied.

5. Numerical experience. We consider the test sets described in [15, Section V.A], containing 1000 simulated data records $\{y(t), u(t)\}_{t=1}^N$:

- D1: $N = 210$, output SNR = 10;
- D2: $N = 210$, output SNR = 1;
- D3: $N = 500$, output SNR = 10;
- D4: $N = 500$, output SNR = 1.

The estimated model order is set to $n = 100$ for all simulations.
We consider two sets of test problems. In the first one, we choose $P(x)$ as the multiple kernel (2.9), where the 'basis' matrices P_i are chosen as follows:

- [DC-M]: $P_i = P^{DC}(1, \mu_i, \rho_i)$ where P^{DC} is the DC kernel defined in (2.8c) and (μ_i, ρ_i) are points of the grid

$$\{0.1i : 1 \leq i \leq 9\} \times \{-0.95, -0.65, -0.35, 0.35, 0.65, 0.95\}$$

  so that $m = 54$;
- [TCSS-M]: $P_i = P^{TC}(1, \mu_i^{TC})$, $i = 1, ..., 21$ where P^{TC} is defined in (2.8a) and $\mu_i^{TC} \in \{0.05i : 2 \leq i \leq 15\} \cup \{0.81 + 0.02i : 0 \leq i \leq 6\}$, $P_{21+i} = P^{SS}(1, \mu_i^{SS})$, $i = 1, ..., 8$ where P^{SS} is defined in (2.8b) and $\mu_i^{SS} \in \{0.8 + 0.02i : 0 \leq i \leq 7\}$. In this case we have $m = 29$.

The matrices P_i, $i = 1, ..., m$ are extremely ill-conditioned: indeed, the average condition number is about 10^n. As concerns the choice of m, we performed several tests also with finer grids and we observed similar behaviours of the algorithms with no significant improvements in the quality of the estimated impulse response coefficients.
In the second set of problems, we consider the following cases:

- [DC] $P(x)$ is the DC kernel (2.8c) with $x = (c, \mu, \rho)^T$, where $c \geq 0$, $0.72 \leq \mu \leq 0.99$, $-0.99 \leq \rho \leq 0.99$;
- [TC] $P(x)$ is the TC kernel (2.8a) with $x = (c, \mu)^T$, where $c \geq 0$, $0.7 \leq \mu \leq 0.99$;
- [SS] $P(x)$ is the SS kernel (2.8b) with $x = (c, \mu)^T$, where $c \geq 0$, $0.7 \leq \mu \leq 0.99$.

We choose the lower bounds on the 'μ' variable according to [16], with the aim to impose a reasonable upper bound to the condition number of $P(x)$.
The quality of the estimated models $\hat{\theta}$ is evaluated by the coefficient

$$W(\hat{\theta}) = 100 \left(1 - \sqrt{\frac{\sum_{i=1}^n |\theta_i^* - \hat{\theta}_i|^2}{\sum_{i=1}^n |\theta_i^* - \bar{\theta}|^2}} \right), \quad \bar{\theta} = \frac{1}{n} \sum_{i=1}^n \theta_i^*, \tag{5.1}$$

where θ_i^* are the true impulse response coefficients and $\hat{\theta}_i$ the estimated ones computed by formula (2.6).

5.1. SGP parameters setting. The SGP parameters have been set as follows: $\beta = 10^{-4}$, $\gamma = 0.4$, $\alpha_{min} = 10^{-7}$, $\alpha_{max} = 10^2$, $L_{min} = \zeta = 10^{-5}$, $L_{max} = 10^{10}$, $M_\alpha = 3$, $\tau = 0.5$. The initial point $x^{(0)}$ is the vector of all ones for the multiple kernels, while we set $x^{(0)} = (0.5, 0.5, 0.8, 0.5)^T$ for the DC kernel and $x^{(0)} = (0.5, 0.8, 0.5)^T$ for the TC and SS kernels. The initial stepsize α_0 is set to 1.
Since SGP is a projection method, it can occur that some of the iterates lay on the boundary of the feasible set. This may create some trouble, since for $x_{m+1} = \sigma^2 = 0$ the matrix $\Sigma(x)$ in (2.3) may become singular. For these reasons, we constrain the $(m+1)$-variable to be greater or equal to some positive constant. Then, we actually consider a problem of the form (4.12) where $l \in \mathbb{R}^{m+1}$, $u \in \mathbb{R}^{m+1} \cup \{+\infty\}$, with $l_{m+1} = 10^{-2}$, $u_{m+1} = +\infty$, and $l_i = 0$, $u_i = +\infty$, $i = 1, ..., m$ for the multiple kernels DC-M and TCSS, $l_1 = 0$, $l_2 = 0.72$, $l_3 = -0.99$, $u_1 = +\infty$, $u_2 = 0.99$, $u_3 = 0.99$ for the kernel DC and $l_1 = 0$, $l_2 = 0.7$, $u_1 = +\infty$, $u_2 = 0.99$ for the kernels TC and SS.
We experimentally observed that the constraint on σ^2 is never active at the solution of (4.12) with $l_{m+1} = 10^{-2}$: we experienced also smaller values, down to 10^{-8}, but we did not observe significant differences in the results and in the algorithms performance. As an alternative, this lower bound can be safely set to a fraction (say between one

tenth to one hundredth) of a preliminary estimate of the noise variance which can be obtained, for instance, as discussed in [41, 40].
We include also the following stopping criterion for the iterates:

$$f(x^{(k)}) - f(x^{(k+1)}) < \tau |f(x^{(k+1)})| \tag{5.2}$$

with $\tau = 10^{-9}$. Indeed, we experienced different values of τ, ranging from 10^{-11} to 10^{-7} and we observed that no significant improvements in accuracy are obtained with smaller tolerance values. A maximum number of 5000 iterations is also imposed.

5.2. Scaling matrix impact. In order to show the significant influence of the scaling strategy in the convergence behaviour of gradient methods, we compare Algorithm 1 with the scaling proposed in Section 4.2 (SGP) with the same algorithm without scaling (GP, $D_k = I$) on some instances of the whole test sets described above. Both algorithms adopt the same adaptive alternation of the Barzilai–Borwein rules (4.4) described in Section 4.1 and have all the other parameters set as described in Section 5.1.
As further benchmark, we consider also the affine scaling ciclic Barzilai–Borwein method (AS-CBB) proposed in [28], which consists in a diagonally scaled gradient method whose iteration is given by $x^{(k+1)} = x^{(k)} + \lambda_k d^{(k)}$ where

$$d_i^{(k)} = \frac{1}{\alpha_k + |\nabla_i f(x^{(k)})|/X_i(x^{(k)})} \nabla_i f(x^{(k)}), \qquad X_i(x) = \begin{cases} u_i - x_i & \text{if } \nabla_i f(x) \le 0 \\ l_i - x_i & \text{otherwise} \end{cases}$$

with the convention $0 \cdot \infty = 1/\infty = 0$. In particular, $\alpha_{c\ell+i} = \max(\alpha_{min}, 1/\alpha_{c\ell+1}^{BB1})$, $i = 1, ..., L_c$ for some fixed cycle length parameter L_c and λ_k is computed by a nonmonotone Armijo-type backtracking procedure. When f is twice continuously differentiable, any limit point of the sequence generated by AS-CBB is a stationary point [28, Theorem 4.1]; moreover, the authors also shows the local R-linear convergence to non degenerate local minimum satisfying the second order optimality conditions [28, Theorem 7.1]. In our experiments, we set $L_c = 4$ and the nonmonotone Armijo parameter (M in formula (2.2) in [28]) equal to 8.
The plots in Figure 1 are obtained by: a) running the Matlab function `fmincon` to get a reference value f^* for a minimum of f; b) running each algorithm and computing the relative difference between f^* and the current estimate $f(x^{(k)})$ at each iterate.
A significantly faster decrease of the objective function value is observed for SGP, with respect to the number of function evaluations, together with a smoother and faster improvement of the estimated impulse response, measured by means of the fit parameter defined in (5.1). In practice, after the very first SGP iterations, a good estimate of the impulse response is obtained.
The comparison between SGP, GP and AS-CBB gives information about the relative behaviour of scaled gradient methods (SGP, AS-CBB) with respect to a non scaled one (GP) and also about the importance of the scaling matrix choice (SGP versus AS-CBB), which, as observed before, leads to very different performances.

5.3. SGP results and performance assessment. In Tables 1 and 2 we summarize the results obtained by applying SGP on the test sets described above. For each dataset, we report the average fit (5.1), the average number of iterations ('it'), the average number of function evaluations ('nf') and the average computational time in seconds ('t'). The whole experimentation has been performed with the Matlab implementation of SGP described in the previous section, running on a server with a dual Intel Xeon QuadCore E5620 processor at 2,40 GHz, 12 Mb cache and 18 Gb

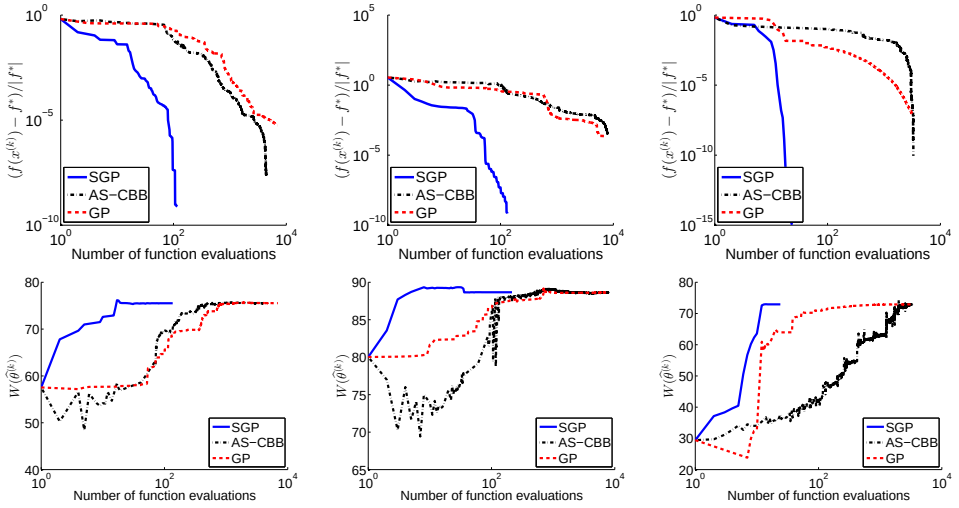

FIG. 1. *Comparison of SGP, GP and AS-CBB with respect to the number of function evaluations on three instances of the test problems (left: multiple kernel DC-M, dataset D3; middle: multiple kernel TCSS-M, dataset D1; right: kernel SS, dataset D2). First row: relative difference from the reference minimum value. Second row: fit parameter* (5.1).

of RAM under Matlab2010b. The accuracy of the results in terms of the fit parameter (5.1) is coherent with the results reported in [15]. To evaluate the effectiveness of SGP, we compare it to other state-of-the-art methods, such as the optimization algorithms implemented in the Matlab `fmincon` function `sqp`, `interior-point` and `trust-region-reflective`, which is the default one, denoted by 'sqp', 'ip' and 'tr' respectively in the tables. We point out that `fmincon` has a further algorithm option, `active-set`, which is however not suited for the considered problems since it may produce infeasible iterates outside the objective function domain.
The optimization parameters for `fmincon` are the default ones except `TolFun` which has been set to 10^{-9}, while the same functions described in Section 4.3 and employed also by SGP have been exploited for objective function and gradient evaluations.
The `sqp` option correspond to a BFGS approximation of the Hessian matrix, while `interior-point` and `trust-region-reflective` admit also a user supplied Hessian instead of an automatically computed approximation of it. With the suffix '-h' we indicate that the exact Hessian was also provided to the solvers. For sake of brevity, in Tables 1 and 2 we only report the case with the best average computational time. Indeed, as observed at the end of Section 4.3, the Hessian computation is quite costly in the multiple kernel case, so that the time needed for computing it is not balanced by the reduction of the iteration number that one expects when uses the exact second order information. For example, with the DC-M kernel on the dataset D1, the average iterations and function evaluations numbers were 23 and 25 respectively for the `interior-point` with the exact Hessian, but the corresponding average computational time was 11.04 seconds. Some instances of the plots of the relative difference from the minimum value and the fit parameter (5.1) as functions of the execution time for the different strategies are shown in Figure 2. For the multiple kernel case we also consider the method recently proposed in [15], which is based on a Minimization-Majorization (MM) approach. In practice, this method solves a sequence of convex optimization subproblems whose objective function is obtained by

		DC-M					TCSS-M				
		SGP	MM	fmincon			SGP	MM	fmincon		
				sqp	ip	tr			sqp	ip	tr
D1	fit	84.4	84.4	84.4	84.4	84.4	84.4	84.4	84.4	84.4	84.4
	it	104	12	44	78	323	123	12	42	72	97
	nf	137	-	121	95	324	156	-	110	88	98
	t	0.74	16.5	0.83	1.02	112.45	0.58	9.13	0.51	0.60	12.68
D2	fit	63.2	63.1	63.2	63.1	63.1	63.6	63.6	63.6	63.6	63.6
	it	76	11	49	114	126	94	11	50	99	87
	nf	103	-	121	134	127	129	-	116	118	88
	t	0.55	14.6	0.83	1.35	44.00	0.46	7.89	0.54	0.79	11.41
D3	fit	87.6	87.6	87.5	85.7	87.6	88.7	88.8	88.8	88.6	88.8
	it	86	11	52	99	179	127	11	52	90	129
	nf	115	-	131	119	180	163	-	131	108	130
	t	0.62	16.5	0.90	1.25	62.48	0.60	8.80	0.60	0.73	16.89
D4	fit	74.9	74.8	74.8	71.5	74.8	76.6	76.7	76.6	76.5	76.6
	it	76	10	70	180	131	95	10	68	139	97
	nf	103	-	160	204	132	126	-	156	160	98
	t	0.55	14.3	1.09	2.00	45.58	0.46	7.96	0.72	1.06	12.70

TABLE 1

Results obtained by SGP, fmincon (with three different algorithm options - see text for details) and MM on multiple kernels DC-M and TCSS-M. For each dataset, we report the average fit (5.1)*, the average number of iterations ('it'), the average number of function evaluations ('nf') and the average computational time in seconds ('t').*

linearizing the concave term. Each subproblem, which can be formulated as a second order cone program, is then solved by an especially tailored interior point method. The theoretical convergence properties of the MM method (see [47, Theorem 4]) are substantially identical to that stated in Theorem 1: every limit point of the sequence is stationary. For the multiple kernel (2.9), property (3.5) of the objective function only guarantees the existence of limit points. We adopt the MM Matlab implementation provided by the authors, which exploits the CVXOPT package, a Python module for convex optimization [1]. The numerical comparison with MM has been carried out with Python 2.7.1 installed and with the ATLAS library compiled and optimized for our architecture. All methods are initialized with the same starting point.

In order to give a more intuitive insight of the comparison among the different solvers, in Figure 3 we also report the performance profiles [23] obtained by grouping the test problems according to the kernel type, multiple or single. Given a test set $\mathcal{P}$ and a set of solvers $\mathcal{S}$, let us denote by $t_{p,s}$ the computational time required by solver $s \in \mathcal{S}$ to solve the problem $p \in \mathcal{P}$. Then, the performance ratio is defined as $\rho_{p,s} = t_{p,s}/\min\{t_{p,s}, s \in \mathcal{S}\}$. When a solver s does not succeed on a problem p, the corresponding ratio $t_{p,s}$ is set to a value ρ_{max} such that $\rho_{p,s} \leq \rho_{max}$ for all $p \in \mathcal{P}$ and $s \in \mathcal{S}$ and $\rho_{p,s} = \rho_{max}$ if and only if a failure occurred. The performance profile of the solver $s \in \mathcal{S}$ is $p_s(\xi) = \text{size}\{p \in \mathcal{P} : \rho_{p,s} \leq \xi\}/\text{size}\{\mathcal{P}\}$, for a given $\xi \in \mathbb{R}$. The quantity $p_s(\xi)$ expresses the probability that a performance ratio $\rho_{p,s}$ lies within a factor of ξ of the best possible ratio.

From Tables 1 and 2 and Figure 3 we can observe what follows:

- in general, all the considered methods provide solutions with comparable accuracy, measured in terms of the fit parameter (5.1). Some differences in accuracy could be due to the fact that problem (2.5) is nonconvex and, then, different algorithms can be attracted by different stationary points; however, the overall results are satisfactory;
- in presence of simple constraints, a first order method as SGP, equipped

		DC				TC				SS			
		SGP	fmincon			SGP	fmincon			SGP	fmincon		
			sqp	ip-h	tr-h		sqp	ip-h	tr-h		sqp	ip-h	tr-h
D1	fit	83.2	83.2	83.2	83.2	82.5	82.5	82.5	82.5	77.6	77.5	77.1	76.2
	it	124	27	21	136	19	18	13	17	43	30	21	225
	nf	168	88	30	137	23	61	22	18	62	81	31	226
	t	0.44	0.29	0.25	1.03	0.06	0.19	0.14	0.11	0.16	0.27	0.22	1.36
D2	fit	60.3	58.0	60.2	60.1	60.4	60.0	60.6	60.5	52.2	49.8	52.7	50.9
	it	59	30	19	42	20	24	15	16	29	29	21	96
	nf	77	86	26	43	23	70	22	17	38	75	28	97
	t	0.21	0.28	0.22	0.34	0.06	0.21	0.14	0.10	0.10	0.25	0.20	0.59
D3	fit	87.9	87.8	87.8	87.8	87.6	87.6	87.6	87.6	87.2	87.1	87.1	86.6
	it	109	29	23	141	21	19	14	18	40	33	24	262
	nf	148	90	30	142	25	63	24	19	55	86	34	263
	t	0.39	0.30	0.25	1.08	0.07	0.20	0.14	0.11	0.15	0.29	0.23	1.58
D4	fit	74.7	74.5	74.8	74.7	74.7	74.6	74.7	74.7	71.7	70.5	72.2	70.2
	it	78	37	25	66	19	27	17	16	32	34	23	172
	nf	104	101	30	67	23	75	26	17	41	83	31	173
	t	0.27	0.35	0.29	0.52	0.06	0.24	0.16	0.10	0.11	0.28	0.22	1.04

TABLE 2

Results obtained by SGP and fmincon (with three different algorithm options - see text for details) on DC, TC, SS kernel matrices. For each dataset, we report the average fit (5.1)*, the average number of iterations ('it'), the average number of function evaluations ('nf') and the average computational time in seconds ('t').*

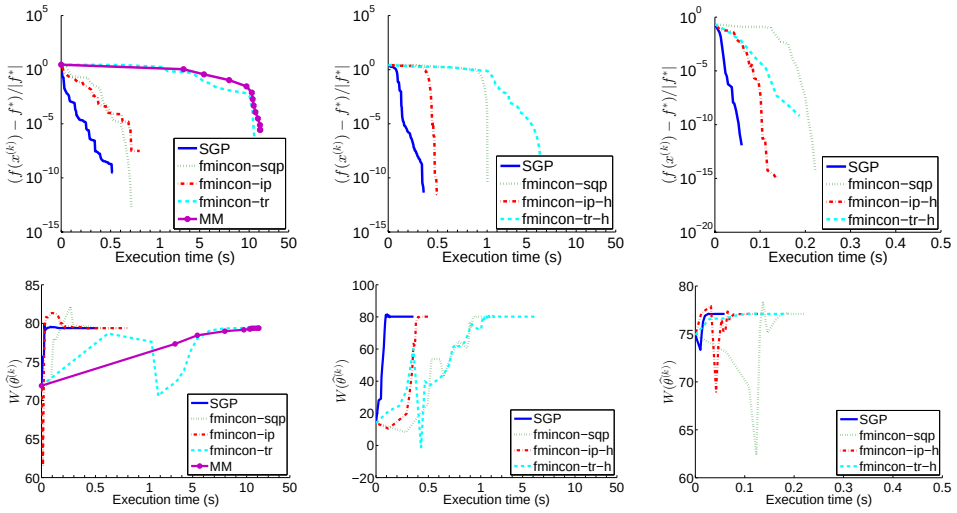

FIG. 2. *Comparison of SGP and fmincon (with three different algorithm options) with respect to the execution time on three instances of the test problems (left: multiple kernel DC-M, dataset D1; middle: kernel SS, dataset D1; right: kernel TC, dataset D4). The MM algorithm is also shown in the multiple-kernel case. First row: relative difference from the reference minimum value. Second row: fit parameter* (5.1).

with a suitable combination of a scaling matrix and a steplength parameter, is competitive with more sophisticated and highly optimized second order methods, as the ones implemented in the `fmincon` Matlab function;

- the high flexibility of SGP allows to overcome some limits of state-of-the-art schemes as the MM approach and be applied also when the objective function is not a difference of convex functions.

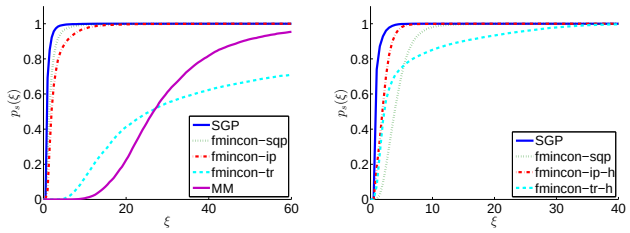

FIG. 3. *Performance profiles. Left: multiple kernels DC-M and TCSS-M. Right: kernels DC, TC and SS.*

6. Concluding remarks. In this paper we have considered linear system identification in the Bayesian framework. A key step is the estimation of the hyperparameters describing the Bayesian prior, which leads to the nonconvex, nonlinear, bound constrained optimization problem (2.5). Our aim was to analyze problem (2.5) from a numerical point of view, proposing also an especially tailored version of SGP for its solution and presenting the results of an extensive numerical experimentation comparing several state-of-the-art algorithms. Our analysis, together with the experimental results, aims to give new insights about the numerical issues related to the considered application and also about gradient projection methods and related scaling techniques. The numerical results, depicted in Figure 3, show that the proposed method obtains the overall best performances in terms of time.
The main contributions of the paper are summarized below. From the optimization point of view:

- we proposed a new split gradient approach for bound constrained optimization;
- the numerical experience shows that scaling techniques are useful to improve the performances of the gradient projection method also on nonconvex problems. The improvements obtained with the proposed approach are observed with respect to the nonscaled version of the same method and also with respect to a gradient method based on a different scaling technique;
- the combination of the proposed scaling technique with a suitable steplength selection rule makes SGP competitive with second-order methods, as the ones implemented in the `fmincon` Matlab function.

From the application point of view:

- we provide an $\mathcal{O}(n^3)$ algorithm to evaluate the objective function and gradient of problem (2.5) and an $\mathcal{O}(mn^3)$ algorithm for Hessian computation;
- we also provide an extensive numerical experimentation with the Matlab `fmincon` function, devising the most convenient algorithm options.

As concluding remarks, we point out that one of the main strength of the proposed approach is the capability to provide a good estimate of the impulse response coefficients after very few iterations, without need of the second order information, which, especially in the multiple kernel case (2.9), is quite costly to compute. We believe that the good performances of SGP rely on the fact that the proposed scaling technique takes into account of the problem structure, that is both the objective function and the constraints. On the other side, this is also the main difficulty to the generalization of SGP: indeed, the scaling technique is especially tailored for box constraints and the extension to more general constraints is not straightforward. This issue will be addressed in our future work, which will consider also a wider range of problems

arising in machine learning and system identification where sparse Bayesian learning ideas can be applied, e.g. the identification of multi-input, multi-output systems, where also automatic variable selection needs to be performed, or the basis selection problem in the context of machine learning.

REFERENCES

[1] M. Andersen, J. Dahl, and L. Vandenberghe, *Cvxopt: Python software for convex optimization, version 1.1.6*. Available at http://cvxopt.org, 2013.

[2] A. Aravkin, J. Burke, A. Chiuso, and G. Pillonetto, *Convex vs non-convex estimators for regression and sparse estimation: the mean squared error properties of ARD and GLasso*, J. Mach. Learn. Res., 15 (2014), pp. 217–252.

[3] M. Ayazoglu and M. Sznaier, *An algorithm for fast constrained nuclear norm minimization and applications to systems identification*, in IEEE CDC, 2012, pp. 3469–3475.

[4] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, *Multiple kernel learning, conic duality, and the smo algorithm*, in Proceedings of the Twenty-first International Conference on Machine Learning, ICML '04, New York, NY, 2004, ACM, pp. 41–48.

[5] M. Bertero, *Linear inverse and ill-posed problems*, Adv. Electron. El. Phys., 75 (1989), pp. 1–120.

[6] M. Bertero, H. Lantéri, and L. Zanni, *Iterative image reconstruction: a point of view*, in Mathematical Methods in Biomedical Imaging and Intensity-Modulated Radiation Therapy (IMRT), Y. Censor, M. Jiang, and A. K. Louis, eds., Birkhauser-Verlag, Pisa, Italy, 2008, pp. 37–63.

[7] D. P. Bertsekas, *Nonlinear programming*, Athena Scientific, Belmont, MA, 2nd ed., 1999.

[8] S. Bonettini, *Inexact block coordinate descent methods with application to non-negative matrix factorization*, IMA J. Numer. Anal., 31 (2011), pp. 1431–1452.

[9] S. Bonettini, A. Cornelio, and M. Prato, *A new semiblind deconvolution approach for Fourier-based image restoration: an application in astronomy*, SIAM J. Imaging Sci., 6 (2013), pp. 1736–1757.

[10] S. Bonettini, G. Landi, E. Loli Piccolomini, and L. Zanni, *Scaling techniques for gradient projection-type methods in astronomical image deblurring*, Int. J. Comput. Math., 90 (2013), pp. 9–29.

[11] S. Bonettini and M. Prato, *Nonnegative image reconstruction from sparse Fourier data: a new deconvolution algorithm*, Inverse Probl., 26 (2010), 095001.

[12] S. Bonettini, R. Zanella, and L. Zanni, *A scaled gradient projection method for constrained image deblurring*, Inverse Probl., 25 (2009), 015002.

[13] G. Box, G. M. Jenkins, and G. Reinsel, *Time series analysis: forecasting & control*, Prentice Hall, Englewood Cliffs, NJ, 3rd ed., 1994.

[14] S. Boyd and L. Vandeberghe, *Convex optimization*, Cambridge University Press, Cambridge, UK, 2004.

[15] T. Chen, M. S. Andersen, L. Ljung, A. Chiuso, and G. Pillonetto, *System identification via sparse multiple kernel-based regularization using sequential convex optimization techniques*, IEEE T. Automat. Contr., 59 (2014), pp. 2933–2945.

[16] T. Chen and L. Ljung, *Implementation of algorithms for tuning parameters in regularized least squares problems in system identification*, Automatica, 49 (2013), pp. 2213–2220.

[17] T. Chen, H. Ohlsson, and L. Ljung, *On the estimation of transfer functions, regularizations and Gaussian processes - revisited*, Automatica, 48 (2012), pp. 1525–1535.

[18] A. Chiuso, T. Chen, L. Ljung, and G. Pillonetto, *On the design of multiple kernels for nonparametric linear system identification*, IEEE CDC 2014, submitted.

[19] A. Chiuso and G. Pillonetto, *A Bayesian approach to sparse dynamic network identification*, Automatica, 48 (2012), pp. 1553–1565.

[20] M. E. Daube-Witherspoon and G. Muehllehner, *Iterative image space reconstruction algorithm suitable for volume ECT*, IEEE T. Med. Imaging, 5 (1986), pp. 61–66.

[21] F. Dinuzzo, *Kernels for linear time invariant system identification*, CoRR, abs/1203.4930 (2012).

[22] T. Doan, R. Litterman, and C. A. Sims, *Forecasting and conditional projection using realistic prior distributions*, Econmet. Rev., 3 (1984), pp. 1–100.

[23] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with performance profiles*, Math. Progam., Ser. A, 91 (2002), pp. 201–213.

[24] D. Donoho, *Compressed sensing*, IEEE T. Infor. Theory, 52 (2006), pp. 1289–1306.

[25] M. Fazel, H. Hindi, and S. P. Boyd, *A rank minimization heuristic with application to minimum order system approximation*, in Proceedings of the American Control Conference, vol. 6, 2001, pp. 4734–4739.
[26] G. Frassoldati, G. Zanghirati, L. Zanni, *New adaptive stepsize selections in gradient methods*, J. Ind. Manage. Optim., 4 (2008), pp. 299–312.
[27] G. C. Goodwin, M. Gevers, and B. Ninness, *Quantifying the error in estimated transfer functions with application to model order selection*, IEEE T. Automat. Contr., 37 (1992), pp. 913–928.
[28] W. W. Hager, B. A. Mair, and H. Zhang, *An affine-scaling interior-point CBB method for box-constrained optimization*, Math. Program., Ser. A, 119 (2009), pp. 1–32.
[29] T. Kailath, *Linear systems*, Prentice Hall, Englewood Cliffs, NJ, 1980.
[30] G. Kitagawa and H. Gersh, *A smoothness priors-state space modeling of time series with trend and seasonality*, J. Am. Stat. Assoc., 79 (1984), pp. 378–389.
[31] H. Lanteri, M. Roche, and C. Aime, *Penalized maximum likelihood image restoration with positivity constraints: multiplicative algorithms*, Inverse Probl., 28 (2002), pp. 1397–1419.
[32] J. B. Lasserre, *A trace inequality for matrix product*, IEEE T. Automat. Contr., 40 (1995), pp. 1500–1501.
[33] D. D. Lee and H. S. Seung, *Learning the parts of objects by non-negative matrix factorization*, Nature, 401 (1999), pp. 788–791.
[34] L. Ljung, *System identification - Theory for the user*, Prentice-Hall, Upper Saddle River, NJ, 2nd ed., 1999.
[35] L. Ljung, H. Hjalmarsson, and H. Ohlsson, *Four encounters with system identification*, Eur. J. Control, 17 (2011), pp. 449–471.
[36] D. J. C. MacKay, *Bayesian non-linear modelling for the prediction competition*, in ASHRAE Transactions, vol. 100, Pt.2, ASHRAE, 1994, pp. 1053–1062.
[37] J. S. Maritz and T. Lwin, *Empirical Bayes method*, Chapman and Hall, London, UK, 1989.
[38] M. Merritt and Y. Zhang, *Interior-point gradient method for large-scale totally nonnegative least squares problems*, J. Optimiz. Theory App., 126 (2005), pp. 191–202.
[39] G. Pillonetto and A. Chiuso, *Tuning complexity in regularized kernel-based regression and linear system identification: the robustness of the marginal likelihood estimator*, Automatica, submitted.
[40] G. Pillonetto, A. Chiuso, and G. De Nicolao, *Prediction error identification of linear systems: a nonparametric Gaussian regression approach*, Automatica, 47 (2011), pp. 291–305.
[41] G. Pillonetto and G. De Nicolao, *A new kernel-based approach for linear system identification*, Automatica, 46 (2010), pp. 81–93.
[42] G. Pillonetto, F. Dinuzzo, T. Chen, G. De Nicolao, and L. Ljung, *Kernel methods in system identification, machine learning and function estimation: a survey*, Automatica, 50 (2014), pp. 657–682.
[43] M. Prato, R. Cavicchioli, L. Zanni, P. Boccacci, and M. Bertero, *Efficient deconvolution methods for astronomical imaging: algorithms and IDL-GPU codes*, Astron. Astrophys., 539 (2012), A133.
[44] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*, The MIT Press, Cambridge, MA, 2006.
[45] L.A. Shepp and Y. Vardi, *Maximum likelihood reconstruction for emission tomography*, IEEE T. Med. Imaging, 1 (1982), pp. 113–122.
[46] T. Soderstrom and P. Stoica, *System identification*, Prentice Hall, London, UK, 1989.
[47] B. K. Sriperumbudur and G. R. G. Lanckriet, *On the convergence of the concave-convex procedure*, Adv. Neural Inf. Process. Syst., 22 (2009), pp. 1391–1407.
[48] R. Tibshirani, *Regression shrinkage and selection via the LASSO*, J. Roy. Stat. Soc. B, 58 (1996), pp. 267–288.
[49] M. Tipping, *Sparse Bayesian learning and the relevance vector machine*, J. Mach. Learn. Res., 1 (2001), pp. 211–244.
[50] D. P. Wipf, B. D. Rao, and S. Nagarajan, *Latent variable Bayesian models for promoting sparsity*, IEEE T. Infor. Theory, 57 (2011), pp. 6236–6255.
[51] R. Zanella, P. Boccacci, L. Zanni, and M. Bertero, *Efficient gradient projection methods for edge-preserving removal of Poisson noise*, Inverse Probl., 25 (2009), 045010.
[52] R. Zanella, G. Zanghirati, R. Cavicchioli, L. Zanni, P. Boccacci, M. Bertero, and G. Vicidomini, *Towards real-time image deconvolution: application to confocal and STED microscopy*, Sci. Rep., 3 (2013), 2523.
[53] Y. Zhu, *Multivariable system identification for process control*, Elsevier Science, New York, NY, 2001.