Bayesian online multi-task learning of Gaussian processes

Gianluigi Pillonetto, Francesco Dinuzzo and Giuseppe De Nicolao

***Abstract*—Standard single-task kernel methods have been recently extended to the case of multi-task learning in the context of regularization theory. There are experimental results, especially in biomedicine, showing the benefit of the multi-task approach compared to the single-task one. However, a possible drawback is computational complexity. For instance, when regularization networks are used, complexity scales as the cube of the overall number of training data, which may be large when several tasks are involved. The aim of this paper is to derive an efficient computational scheme for an important class of multi-task kernels. More precisely, a quadratic loss is assumed and each task consists of the sum of a common term and a task-specific one. Within a Bayesian setting, a recursive on-line algorithm is obtained, that updates both estimates and confidence intervals as new data become available. The algorithm is tested on two simulated problems and a real dataset relative to xenobiotics administration in human patients.**

***Index Terms*—collaborative filtering; multi-task learning; mixed effects model; kernel methods; regularization; Gaussian processes; Kalman filtering; pharmacokinetic data**

I. INTRODUCTION

Standard multidimensional regression deals with the reconstruction of a scalar function from a finite set of noisy samples, see e.g. [1], [2], [3]. When the simultaneous learning of several functions (tasks) is considered the so-called multi-task learning problem arises. The main point is that measurements taken on a task may be informative with respect to the other ones.

A typical multi-task problem is found in the analysis of biomedical data when experiments performed on several patients belonging to a population are analyzed. Usually, the individual responses share some common features so that data from a subject can help reconstructing also the responses of other individuals. The so-called population analysis is widely applied in pharmacokinetics (PK) and pharmacodynamics (PD) [4]. In this field, a *parametric* modelling approach based on compartmental models is mostly employed [5], [6]. The widely used NONMEM software traces back to the seventies [7], [8], whereas more sophisticated approaches include also Bayesian MCMC algorithms [9], [10]. More recently, semi-parametric and *nonparametric* approaches were developed for the population analysis of PK/PD and genomic data [11], [12], [13], [14], [15].

In the machine learning literature, the term multi-task learning has been popularized by [16]. Further investigations demonstrated the potential advantage of multi-task approaches against those that learn the single functions separately (single-task approach) [17], [18]. Another research issue has to do with the determination, within a Bayesian setting, of the amount of information needed to learn a task when it is simultaneously learned with several other ones [19]. Recently, vector-valued Reproducing Kernel Hilbert Spaces (RKHS) [20] were used to derive multi-task regularized kernel methods [21].

Among the open research questions listed in [21], there are the development of on-line multi-task learning schemes and the reduction of computational complexity. *On-line* multi-task learning concerns the recursive processing of examples that are made available in real-time. As for the second question, namely computational complexity, multi-task methods suffer from the problem of requiring much more operations than single-task ones. For instance, when using kernel methods with quadratic loss function (regularization networks), complexity scales with the cube of the overall number of examples, whereas each single-task problem scales with the cube of its examples. As observed in [21], a substantial improvement is possible when all the k tasks share the same n inputs and the multi-task kernel has a suitable structure, in which case complexity can be reduced to $O(kn^3)$. Along this direction, an $O(kn^3)$ algorithm for regularization networks in the longitudinal case has been recently developed [22].

A number of works on multi-task learning have addressed the case of several single-task problems sharing the same kernel. Then, the availability of multiple training sets is exploited to learn a better kernel, e.g. using the EM algorithm [23], [24], [25]. This kind of problems (learning several tasks with the same kernel) arises also in multi-class classification [26] and functional data analysis [27], [28]. The common feature of all these methods is that, once the kernel has been determined, the overall learning problem boils down to solving a set of single-task problems. Conversely, along the line of [21], [14], [29], [30], we adopt a more cooperative perspective in which, also for a given kernel choice, all training sets contribute to the reconstruction of each single task. This cooperative scheme is obtained assuming a quadratic loss and kernels which are the sum of a common term and a task-specific one. In particular, we derive a recursive algorithm that updates the estimates as new examples become available. On-line methods developed in single-task contexts [31], [32] rely upon sparse representations of Gaussian models, obtained, for example, replacing the posterior distribution with a simpler parametric description. In the present paper, conversely, computational efficiency is achieved without neither introducing approxima-

G. Pillonetto is with Dipartimento di Ingegneria dell'Informazione, University of Padova, Padova, Italy.

F. Dinuzzo is with Dipartimento di Matematica, University of Pavia, Pavia, Italy.

G. De Nicolao is with Dipartimento di Informatica e Sistemistica, University of Pavia, Pavia, Italy.

tions nor imposing constraints on the location of inputs but just exploiting the possible presence of repeated locations. The algorithm relies on a Bayesian reformulation of the problem and efficient formulas for the confidence intervals are also worked out. Part of the overall scheme can be viewed as a Kalman filter for a system with growing state dimension.

The paper is organized as follows. In section II, the multi-task learning problem is stated within a Bayesian framework. In Section III, the algorithmic core of the recursive scheme is derived. In Section IV and V, an efficient algorithm which solves the on-line multi-task learning problem is worked out, while in Section VI, simulated and real pharmacokinetic/biological data are used to test the computational scheme. Conclusions then end the paper. The Appendix A contains some technical results used in the paper while in Appendix B an extension of the proposed algorithm is discussed.

II. Preliminaries

In this section, kernel-based multi-task learning is briefly reviewed. In particular, the problem is introduced according to [21]. We take this deterministic approach as a starting point, then showing that the problem can be given a probabilistic Bayesian formulation. Further, the specific class of multi-task problems addressed in the paper is introduced within such Bayesian setting. Finally, some useful notation is given. Throughout the paper, boldface letters will be used to denote scalar or vector functions.

A. A brief review of kernel-based multi-task learning

Consider a set of k task functions $\mathbf{f}_j : X \mapsto \mathbb{R}$ where X, a compact set in $\mathbb{R}^d$, is an input space common to all tasks. For the j-th task, the following n_j examples are available

$$D_j := \left\{(x_{1j}, y_{1j}), \dots, (x_{n_j j}, y_{n_j j})\right\}.$$

The overall number of examples is $n^k = \sum_{j=1}^{k} n_j$. The aim is to jointly estimate all the unknown functions $\mathbf{f}_j$ starting from the overall dataset

$$D^k := \bigcup_{j=1}^{k} D_j.$$

Following [21], let the vector-valued function $\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_k]$ belong to an RKHS $\mathcal{H}$ with norm $\|\cdot\|_{\mathcal{H}}$, associated with the multi-task kernel $K((x_1,p_1),(x_2,p_2))$, with $x_i \in X$, $1 \le p_i \le k$, $i = 1,2$. According to the regularization approach, $\mathbf{f}$ can be estimated by minimizing the functional

$$J(\mathbf{f}) = \sum_{j=1}^{k}\sum_{i=1}^{n_j}(y_{ij} - \mathbf{f}_j(x_{ij}))^2 + \gamma\|\mathbf{f}\|^2_{\mathcal{H}}.$$

In the above expression, the sum of squares penalizes solutions which are not adherent to experimental evidence. Further, γ is the so-called regularization parameter which controls the balance between the training error and the solution regularity measured by $\|\mathbf{f}\|^2_{\mathcal{H}}$. The so-called *representer theorem* provides the regularization network expression of the minimizer of J (see e.g. [33], [34]):

$$\hat{\mathbf{f}}_p(x) = \sum_{j=1}^{k}\sum_{i=1}^{n_j} c_{ij} K\left((x,p),(x_{ij},j)\right), \quad p = 1,\dots,k, \quad (1)$$

where the weights $\{c_{ij}\}$ of the network are the solution of the following linear system of equations

$$\sum_{j=1}^{k}\sum_{i=1}^{n_j}\left[K((x_{ip},p),(x_{ij},j)) + \gamma\delta_{iq}\delta_{jp}\right]c_{ij} = y_{qp}, \quad (2)$$

where $p = 1,\dots,k$, $q = 1,\dots,n_p$ and δ_{ij} is the Kroenecker delta.

B. Problem formulation in a Bayesian setting

For future developments, it will be useful to define the following vectors

$$\begin{array}{rclrcl} y_j & := & [y_{1j}\dots y_{n_j j}]^T & y^k & := & [y_1^T \dots y_k^T]^T \\ c_j & := & [c_{1j}\dots c_{n_j j}]^T & c^k & := & [c_1^T \dots c_k^T]^T. \end{array}$$

According to the above notation, a variable with a subscript, e.g. x_j, indicates a vector associated with the j-th task, whereas a variable with a superscript, e.g. x^k, indicates the vector obtained by stacking all the vectors x_j of the first k tasks. In addition, in the sequel E indicates the expectation operator while I denotes the identity matrix of proper size. Given two random column vectors u and v, let

$$\begin{array}{rcl} cov[u,v] & := & E[(u - E[u])(v - E[v])^T], \\ Var[u] & := & E[(u - E[u])(u - E[u])^T]. \end{array}$$

Moreover, $\mathbf{N}(\mu,\Sigma)$ denotes the multinormal density with mean μ and autocovariance Σ. We also recall the following Lemma on the conditional distribution of Gaussian vectors, see e.g. [35], or Section 3.1 in [36].

Lemma 1: Let u, v be two random vectors. If

$$\left(\begin{array}{c} u \\ v \end{array}\right) \sim \mathbf{N}(0,\Sigma), \qquad \Sigma = \left(\begin{array}{cc} \Sigma_{uu} & \Sigma_{uv} \\ \Sigma_{vu} & \Sigma_{vv} \end{array}\right),$$

then

$$u|v \sim \mathbf{N}(\Sigma_{uv}\Sigma_{vv}^{-1}v, \Sigma_{uu} - \Sigma_{uv}\Sigma_{vv}^{-1}\Sigma_{vu}).$$

■

Hereafter, the following relation is assumed to hold

$$y_{ij} = \mathbf{f}_j(x_{ij}) + \epsilon_{ij}, \quad (3)$$

where the variables $\{\epsilon_{ij}\}$ are mutually independent and identically distributed, $\forall i,j$, with

$$\epsilon_{ij} \sim \mathbf{N}(0, \sigma^2_{ij}).$$

A Bayesian paradigm is adopted and the tasks $\mathbf{f_p(x)}$ will be regarded as realizations of Gaussian random fields. Let

$$\boldsymbol{\xi}^k_p(x) := E\left[\mathbf{f}_p(x)|D^k\right], \quad \boldsymbol{V}^k_p(x) := Var\left[\mathbf{f}_p(x)|D^k\right],$$

$$\boldsymbol{r}^k_p(x) := cov\left[\mathbf{f}_p(x), y^k\right].$$

Note that $\boldsymbol{\xi}^k_p$ is just the Bayes estimate of the p-th task whereas $\boldsymbol{V}^k_p$ is the associated posterior variance. The following proposition exploits the correspondence between Gaussian processes and RKHS, see e.g. [37]. It provides a link between regularization networks associated with a multi-task kernel and Bayesian estimation of Gaussian random fields.

Assumption 2: Assume that $\{\mathbf{f}_j\}_{j=1}^k$ are zero-mean Gaussian random fields, independent of $\epsilon_{ij}, \forall i, j$, with covariances

$$\mathit{cov}\left[\mathbf{f}_p(x_1), \mathbf{f}_q(x_2)\right] = K((x_1, p), (x_2, q)), \quad p, q = 1, \ldots, k.$$

Proposition 3: Under Assumption 2 and assuming $\sigma_{ij}^2 = \gamma$, the posterior mean $\boldsymbol{\xi}_p^k(x)$ is given by eqs. (1)-(2).

Proof: According to Lemma 1,

$$\boldsymbol{\xi}_p^k(x) = \boldsymbol{r}_p^k(x)\left(V^k\right)^{-1} y^k,$$

where, in view of eq. (3) and the given assumptions,

$$\begin{aligned} V^k &= \begin{bmatrix} V_{11} & \cdots & V_{1k} \\ \vdots & \ddots & \vdots \\ V_{k1} & \cdots & V_{kk} \end{bmatrix} + \gamma I, \\ V_{pq}(i,j) &:= K((x_{ip}, p), (x_{jq}, q)), \quad V_{pq} \in \mathbb{R}^{n_p \times n_q}. \end{aligned}$$

Moreover,

$$\begin{aligned} \boldsymbol{r}_p^k(x) &= \mathit{cov}\left[\mathbf{f}_p(x), \left[\mathbf{f}_1(x_{11}) \cdots \mathbf{f}_k(x_{n_k k})\right]\right] \\ &= \left[K((x,p),(x_{11},1)) \cdots K((x,p),(x_{n_k k}, k))\right]^T. \end{aligned}$$

Letting $c^k := \left(V^k\right)^{-1} y^k$ it easily follows that $\boldsymbol{\xi}_p^k(x)$ coincides with $\hat{\mathbf{f}}_p$ given in eqs. (1)-(2). ∎

In the following assumption, we introduce the specific class of multi-task models which will be the focus of the paper. The key feature is the decomposition of tasks into a global component and a local one. The former accounts for similarity among the tasks whereas the latter describes the individual differences.

Assumption 4: For each j and $x \in X$,

$$\mathbf{f}_j(x) = \overline{\mathbf{f}}(x) + \widetilde{\mathbf{f}}_j(x),$$

where $\overline{\mathbf{f}}$ and $\widetilde{\mathbf{f}}_j$ are zero-mean Gaussian random fields. In addition, it is assumed that $\{\epsilon_{ij}\}$, $\overline{\mathbf{f}}$ and $\widetilde{\mathbf{f}}_j$ are all mutually independent.

∎

Under Assumptions 2 and 4, it follows that there exist kernels $\overline{K}$ and $\widetilde{K}_j$, $j = 1, \ldots, k$, such that

$$K((x_1, p), (x_2, q)) = \overline{\lambda}^2 \overline{K}(x_1, x_2) + \delta_{pq} \widetilde{\lambda}^2 \widetilde{K}_p(x_1, x_2),$$

where

$$\left\{ \begin{array}{lcl} \overline{\lambda}^2 \overline{K}(x_1, x_2) & = & \mathit{cov}\left[\overline{\mathbf{f}}(x_1), \overline{\mathbf{f}}(x_2)\right], \\ \widetilde{\lambda}^2 \widetilde{K}_p(x_1, x_2) & = & \mathit{cov}\left[\widetilde{\mathbf{f}}_p(x_1), \widetilde{\mathbf{f}}_p(x_2)\right], \end{array} \right. \tag{4}$$

with $\overline{\lambda}^2$ and $\widetilde{\lambda}^2$ being scale factors that will be typically estimated from data, see Section V.

Assumption 4 extends the model described in Section 3.1.1 of [21] to nonlinear multi-task kernels. If $\overline{\lambda} = 0$, all the tasks are learnt independently of each other. Conversely, $\widetilde{\lambda} = 0$, implies that all the tasks are actually the same. In fact, we are assuming that each task is given by the sum of an average function $\overline{\mathbf{f}}$, hereafter named *average task*, and an *individual shift* $\widetilde{\mathbf{f}}_j(x)$ specific for each task [14].

Assuming homoskedastic noise in eq. (3) so that $\sigma_{ij}^2 = \sigma^2$, it is not difficult to see that by rescaling the triple $\left(\sigma^2, \overline{\lambda}, \widetilde{\lambda}\right)$ with the same constant, the task estimates do not change so that it would seem that there is some redundancy. However, all three parameters are needed in a truly Bayesian setting, because such a scaling affects both the computation of the marginal likelihood and the derivation of confidence intervals.

When examples from k tasks are available and Proposition 3 is used, it would seem that the computational complexity scales with the cube of the total number n^k of examples, that is the cost of solving (2). The rest of the paper is devoted to derive a more efficient numerical scheme that exploits the specific structure of the problem stemming from Assumption 4. Furthermore, the goal is to perform estimation in an online manner, as formalized below.

Problem 5: Assume that the dataset D^k, associated with the first k tasks, is given. In addition, suppose that a new set of examples D_{k+1}, relative to the $(k+1)$-th task, becomes available. Then,

1) Compute efficiently $\boldsymbol{\xi}_j^k(x)$ and $\mathbf{V}_j^k(x), \quad j = 1, \ldots, k$
2) By recursion, compute efficiently $\boldsymbol{\xi}_j^{k+1}(x)$ and $\mathbf{V}_j^{k+1}(x), \quad j = 1, \ldots, k+1$

C. Additional notation

Let

$$\begin{array}{lcllcl} x_j & := & [x_{1j} \ldots x_{n_j j}]^T & x^k & := & [x_1^T \ldots x_k^T]^T \\ \epsilon_j & := & [\epsilon_{1j} \ldots \epsilon_{n_j j}]^T & \epsilon^k & := & [\epsilon_1^T \ldots \epsilon_k^T]^T \\ \overline{f}_j & := & [\overline{f}_{1j} \cdots \overline{f}_{n_j j}]^T & \overline{f}^k & := & [\overline{f}_1^T \cdots \overline{f}_k^T]^T \\ \widetilde{f}_j & := & [\widetilde{f}_{1j} \cdots \widetilde{f}_{n_j j}]^T & \widetilde{f}^k & := & [\widetilde{f}_1^T \cdots \widetilde{f}_k^T]^T. \end{array}$$

where $\overline{f}_{ij} := \overline{\mathbf{f}}(x_{ij})$ and $\widetilde{f}_{ij} := \widetilde{\mathbf{f}}_j(x_{ij})$. Let also

$$\begin{array}{lcllcl} S_j & := & Var\left[\epsilon_j\right] & S^k & := & Var\left[\epsilon^k\right] \\ \overline{V}_j & := & Var\left[\overline{f}_j\right] & \overline{V}^k & := & Var\left[\overline{f}^k\right] \\ \widetilde{V}_j & := & Var\left[\widetilde{f}_j\right] & \widetilde{V}^k & := & Var\left[\widetilde{f}^k\right]. \end{array}$$

In the training set, there might be repeated input locations. As it will be seen in the following, exploiting these repetitions is essential in order to improve computational complexity of the multi-task learning algorithm. For this purpose, it is useful to introduce the *condensed* vector $\breve{x}^k$ whose components are the distinct elements (i.e. with no repetitions) of the set $\bigcup_{j=1}^k \bigcup_{i=1}^{n_j} \{x_{ij}\}$. For example, if $x_1 = [1,2,3]^T$, $x_2 = [1,3,6]^T$, then $x^2 = [1,2,3,1,3,6]^T$ and $\breve{x}^2 = [1,2,3,6]^T$. It is important to notice that x^k has dimension $n^k = \sum_{j=1}^k n_j$, while the dimension of $\breve{x}^k$, denoted by $\breve{n}^k$, can be much smaller. Let C^k and $\breve{C}^k$ be the binary matrices such that

$$\breve{x}^k = C^k x^k \qquad x^k = \breve{C}^k \breve{x}^k.$$

The condensed vector of samples of the average task is defined by

$$\breve{f}^k = C^k \overline{f}^k,$$

and has the same dimension as $\breve{x}^k$. Viceversa, if the condensed vector $\breve{f}^k$ is given, its full version is obtained using $\breve{C}^k$, i.e.

$$\overline{f}^k = \breve{C}^k \breve{f}^k$$

Let $\breve{C}_j$ be the sub-matrix of $\breve{C}^k$ such that

$$\overline{f}_j = \breve{C}_j \breve{f}^k,$$

where the dependence of $\breve{C}_j$ on k is omitted to simplify the notation. Finally, let $\breve{f}_k$ be such that

$$\breve{f}^k = \begin{pmatrix} \breve{f}^{k-1} \\ \breve{f}_k \end{pmatrix}.$$

In other words, $\breve{f}_k$ is the sub-vector of $\breve{f}^k$ associated with the k-th task. Note that, if the k-th task does not bring any new input location, then $\breve{f}_k$ is an empty vector. Let also

$$\begin{array}{rclrcl} \breve{V}^k & := & Var\left[\breve{f}^k\right] & \breve{r}_p^k & := & cov\left[\breve{f}_p, \breve{f}^k\right] \\ \breve{\xi}^{k_1|k_2} & := & E\left[\breve{f}^{k_1}|D^{k_2}\right] & \breve{V}^{k_1|k_2} & := & Var\left[\breve{f}^{k_1}|D^{k_2}\right]. \end{array}$$

where $k_1, k_2 \in \mathbb{N}$. Finally, using the above notation, the following equations hold

$$y_j = \breve{C}_j \breve{f}^k + \widetilde{f}_j + \epsilon_j, \tag{5}$$

$$y^k = \breve{C}^k \breve{f}^k + \nu^k, \tag{6}$$

where $\nu^k := \widetilde{f}^k + \epsilon^k$ is independent of $\overline{\mathbf{f}}$.

III. Recursive estimation of the sampled average task

As it will become clear in the sequel, the posterior mean $\breve{\xi}^{k|k}$ and the posterior variance $\breve{V}^{k|k}$ represent the two key quantities to be propagated in order to compute efficiently $\boldsymbol{\xi}_j^k(x)$, that is to solve Problem 5. These two quantities represent the point estimate and the corresponding uncertainties on the condensed input points. The aim of this section is to derive the recursive update formulas for $\breve{\xi}^{k|k}$ and $\breve{V}^{k|k}$. Once such posterior of the *sampled* average task $\overline{f}^k$ is available, the estimates of the *functions* $\{\mathbf{f}_j\}$ will be computed as discussed in Section IV. In other words, the first step consists in learning the values of the average and individual tasks in correspondence of the available inputs. It will be shown that such estimates are sufficient to reconstruct the entire functions all over the input space.

Proposition 6: $\breve{\xi}^{k|k}$ and $\breve{V}^{k|k}$ can be recursively updated according to the following three steps.

1) Initialization:

$$\begin{array}{rcl} A_1 & = & \breve{V}^1 + \widetilde{V}^1 + S^1, \\ \breve{\xi}^{1|1} & = & \breve{V}^1 A_1^{-1} y_1, \\ \breve{V}^{1|1} & = & \breve{V}^1 - \breve{V}^1 A_1^{-1} \breve{V}^1. \end{array}$$

2) Task update (predictor):

$$\begin{array}{rcl} H_k & = & \breve{r}_{k+1}^k (\breve{V}^k)^{-1}, \\ \breve{\xi}^{k+1|k} & = & \begin{pmatrix} I \\ H_k \end{pmatrix} \breve{\xi}^{k|k}, \end{array} \tag{7}$$

$$\breve{V}^{k+1|k} = \breve{V}^{k+1} - \begin{pmatrix} I \\ H_k \end{pmatrix} \left(\breve{V}^k - \breve{V}^{k|k}\right) \begin{pmatrix} I & H_k^T \end{pmatrix} \tag{8}$$

3) Measurement update (corrector):

$$A_{k+1} = \breve{C}_{k+1} \breve{V}^{k+1|k} \breve{C}_{k+1}^T + \widetilde{V}_{k+1} + S_{k+1}, \tag{9}$$

$$B_{k+1} = \breve{V}^{k+1|k} \breve{C}_{k+1}^T, \tag{10}$$

$$\breve{\xi}^{k+1|k+1} = \breve{\xi}^{k+1|k} + B_{k+1} A_{k+1}^{-1} \left(y_{k+1} - \breve{C}_{k+1} \breve{\xi}^{k+1|k}\right), \tag{11}$$

$$\breve{V}^{k+1|k+1} = \breve{V}^{k+1|k} - B_{k+1} A_{k+1}^{-1} B_{k+1}^T. \tag{12}$$

Proof: Exploiting Lemma 1, one has

$$\begin{array}{rcl} \breve{\xi}^{1|1} & = & cov\left[\breve{f}^1, y_1\right] \left(Var\left[y_1\right]\right)^{-1} y_1, \\ \breve{V}^{1|1} & = & \breve{V}^1 - cov\left[\breve{f}_1, y_1\right] \left(Var\left[y_1\right]\right)^{-1} cov\left[\breve{f}_1, y_1\right]^T. \end{array}$$

Using the equation $y_1 = \breve{f}^1 + \widetilde{f}_1 + \epsilon_1$ and the independence assumptions, one immediately obtains

$$\begin{array}{rcl} cov\left[\breve{f}_1, y_1\right] & = & \breve{V}^1 \\ Var\left[y_1\right] & = & \breve{V}^1 + \widetilde{V}^1 + S^1 = A_1. \end{array}$$

Passing now to the predictor step, to derive (7), we project $\breve{f}^{k+1}$ first onto the space generated by $\breve{f}^k$ and D^k and then onto D^k, that is

$$\breve{\xi}^{k+1|k} = E\left[E\left[\breve{f}^{k+1}|\breve{f}^k, D^k\right]|D^k\right]. \tag{13}$$

Using point (a) of Lemma 11, we obtain

$$E\left[\breve{f}^{k+1}|\breve{f}^k, D^k\right] = E\left[\breve{f}^{k+1}|\breve{f}^k\right] \tag{14}$$

so that

$$E\left[\breve{f}^{k+1}|\breve{f}^k, D^k\right] = cov\left[\breve{f}^{k+1}, \breve{f}^k\right] \left(\breve{V}^k\right)^{-1} \breve{f}^k = \begin{pmatrix} I \\ H_k \end{pmatrix} \breve{f}^k.$$

Finally,

$$\breve{\xi}^{k+1|k} = E\left[\begin{pmatrix} I \\ H_k \end{pmatrix} \breve{f}^k | D^k\right] = \begin{pmatrix} I \\ H_k \end{pmatrix} \breve{\xi}^{k|k},$$

which proves eq. (7). To obtain eq. (8), recall from eq. (6) that

$$y^k = \breve{C}^k \breve{f}^k + \nu^k,$$

with ν^k independent of $\breve{f}^{k+1}$. Then, eq. (8) follows from Lemma 13, with

$$z = \breve{f}^{k+1}, \qquad \eta = \breve{f}^k, \qquad v = \nu^k,$$

$$y = y^k, \qquad F = \breve{C}^k, \qquad U = \breve{V}^{k+1},$$

$$V = \breve{V}^k, \qquad \Gamma = \begin{pmatrix} \breve{V}^k \\ \breve{r}_{k+1}^k \end{pmatrix}, \qquad \Sigma_v = Var\left[\nu^k\right].$$

Finally, let us consider the measurement update. Notice that, by Lemma 12,

$$E\left[y_{k+1}|D^k\right] = \breve{C}_{k+1} \breve{\xi}^{k+1|k}.$$

Then, letting

$$A_{k+1} := Var\left[y_{k+1}|D^k\right], \qquad B_{k+1} := cov\left[\breve{f}^{k+1}, y_{k+1}|D^k\right],$$

by Lemma 1 we obtain equations (11) and (12). Expressions (9) and (10) for A_{k+1} and B_{k+1} follow by applying Lemma 12. ■

The major difference between Proposition 6 and a Kalman filter is that the dimension of the state $\breve{f}^k$ (i.e. the number of distinct input locations up to the first k tasks) can increase. This nontrivial issue is handled by means of the projection Lemma 13 in the derivation of the predictor step.

IV. Solution of the online multi-task learning problem

In the previous section, efficient recursive formulas have been derived for the estimation of the task functions sampled in correspondence of the input locations x^k. In this section, the estimate of $\mathbf{f}_j(x)$ is extended to the whole input space. In addition, confidence intervals are provided. In Appendix B the proposed numerical scheme is also extended in order to process new measurements associated with an existing task.

A. Task estimation

The next proposition shows that $\boldsymbol{\xi}_j^k(x)$ admits a representation in terms of a multi-task regularization network whose weight vector can be efficiently updated online as the number of tasks, and associated examples, increase. In particular, given k tasks, the complexity of the proposed algorithm scales as $O(k(\breve{n}^k)^3)$, where $\breve{n}^k$ is the number of distinct inputs. Recall that $\breve{n}^k$ may well be much smaller than the overall number of examples n^k.

Proposition 7: Under assumption 4, the posterior mean coincides with eq.(1)-(2) and is given by the multi-task regularization network

$$\boldsymbol{\xi}_j^k(x) = \overline{\lambda}^2 \sum_{i=1}^{\breve{n}^k} a_i \overline{K}(x, \breve{x}_i^k) + \widetilde{\lambda}^2 \sum_{i=1}^{n_j} b_{ij} \widetilde{K}(x, x_{ij}),$$

where $\overline{K}$ and $\widetilde{K}$ are defined in eq. (4) and the weights are

$$\begin{aligned} a &= \left(\breve{V}^k\right)^{-1} \breve{\xi}^{k|k}, \\ b_j &= \left(\widetilde{V}_j + S_j\right)^{-1} \left(y_j - \breve{C}_j \breve{\xi}^{k|k}\right). \end{aligned}$$

Proof: Let

$$\overline{\boldsymbol{\xi}}^k(x) := E\left[\overline{\mathbf{f}}(x)|D^k\right], \qquad \widetilde{\boldsymbol{\xi}}_j^k(x) := E\left[\widetilde{\mathbf{f}}_j(x)|D^k\right].$$

Then,

$$\boldsymbol{\xi}_j^k(x) = \overline{\boldsymbol{\xi}}^k(x) + \widetilde{\boldsymbol{\xi}}_j^k(x).$$

Following the same reasonings as in the second part of the proof of Proposition 6, in particular using eqs. (13,14) with $\breve{f}^{k+1}$ replaced by $\overline{\mathbf{f}}(x)$, one obtains

$$\overline{\boldsymbol{\xi}}^k(x) = cov\left[\overline{\mathbf{f}}(x), \breve{f}^k\right] \left(\breve{V}^k\right)^{-1} \breve{\xi}^{k|k},$$

so that, recalling the definition of $\overline{K}$, the expression for a is obtained. To compute $\widetilde{\boldsymbol{\xi}}_j^k(x)$, we first project $\widetilde{\mathbf{f}}_j(x)$ onto the space spanned by $\overline{f}_j$ and D^k, and then onto D^k. We have

$$\widetilde{\boldsymbol{\xi}}_j^k(x) = E\left[E\left[\widetilde{\mathbf{f}}_j(x)|\overline{f}_j, D^k\right]|D^k\right].$$

Exploiting point (b) of Lemma 11, and recalling (5), one obtains

$$\begin{aligned} E\left[\widetilde{\mathbf{f}}_j(x)|\overline{f}_j, D^k\right] &= E\left[\widetilde{\mathbf{f}}_j(x)|\overline{f}_j, D_j\right] = E\left[\widetilde{\mathbf{f}}_j(x)|\widetilde{f}_j + \epsilon_j\right] \\ &= cov\left[\widetilde{\mathbf{f}}_j(x), \widetilde{f}_j\right] \left(\widetilde{V}_j + S_j\right)^{-1} \left(y_j - \overline{f}_j\right), \end{aligned}$$

where the last equality follows from Lemma 1. Finally, by projecting $\left(y_j - \overline{f}_j\right)$ onto D^k, we have

$$\widetilde{\boldsymbol{\xi}}_j^k(x) = cov\left[\widetilde{\mathbf{f}}_j(x), \widetilde{f}_j\right] \left(\widetilde{V}_j + S_j\right)^{-1} \left(y_j - \breve{C}_j \breve{\xi}^{k|k}\right),$$

which, recalling the definition of $\widetilde{K}$, completes the proof. ■

B. Computation of confidence intervals

Assume that data relative to the first k tasks have been already processed and that $\breve{V}^{k|k}$ has been computed by means of Proposition 6. Given an arbitrary input location x, obtaining confidence intervals for $\mathbf{f}_j(x)$ calls for the computation of the posterior variances $\mathbf{V}_j^k(x)$. To this aim, define

$$\phi_j := \begin{bmatrix} f_j \\ \mathbf{f}_j(x) \end{bmatrix} \quad \overline{\phi}_j := \begin{bmatrix} \overline{f}_j \\ \overline{\mathbf{f}}_j(x) \end{bmatrix} \quad \widetilde{\phi}_j := \begin{bmatrix} \widetilde{f}_j \\ \widetilde{\mathbf{f}}_j(x) \end{bmatrix},$$

so that $\phi_j = \overline{\phi}_j + \widetilde{\phi}_j$. Letting $P = \begin{pmatrix} I_{n_j} & 0 \end{pmatrix}$, one has

$$y_j = P\phi_j + \epsilon_j.$$

Define the following unconditional moments

$$\overline{V}_{\phi_j} = Var\left[\overline{\phi}_j\right] \quad \widetilde{V}_{\phi_j} = Var\left[\widetilde{\phi}_j\right] \quad \overline{r}_{\phi_j}^k = cov\left[\overline{\phi}_j, \breve{f}^k\right]$$

as well as the following conditional ones

$$\overline{M}_j = Var\left[\overline{\phi}_j|D^k\right] \quad M_{-j} = Var\left[\phi_j|D_{-j}^k\right] \quad M_j = Var\left[\phi_j|D^k\right]$$

where D_{-j}^k is the training set containing all collected data but those regarding D_j, i.e. $D_{-j}^k = D^k \setminus D_j$. Notice that

$$\mathbf{V}_j^k(x) = [M_j]_{n_j+1, n_j+1}$$

where $[\cdot]_{i,j}$ denotes the (i,j) entry of a matrix. Since the random vector $\overline{\phi}_j$ conditional on D^k is correlated with $\widetilde{\phi}_j$, it is convenient to first calculate M_{-j}, as described in the next lemma whose proof is reported in Appendix. In fact, this permits to obtain immediately $cov\left[\phi_j, y_j|D_{-j}^k\right]$ and $Var\left[y_j|D_{-j}^k\right]$, thus simplifying the computation of the confidence interval, as described in Proposition 9.

Lemma 8: It holds that

$$M_{-j} = \left(\overline{M}_j^{-1} - P^T\left(\widetilde{V}_j + S_j\right)^{-1} P\right)^{-1} + \widetilde{V}_{\phi_j} \tag{15}$$

where

$$\overline{M}_j = \overline{V}_{\phi_j} - \overline{r}_{\phi_j}^k \left((\breve{V}^k)^{-1} \breve{V}^{k|k} (\breve{V}^k)^{-1} - (\breve{V}^k)^{-1}\right) \overline{r}_{\phi_j}^{kT} \tag{16}$$

Confidence intervals are finally provided by the following proposition.

Proposition 9:

$$M_j = M_{-j} - M_{-j} P^T \left(P M_{-j} P^T + S_j\right)^{-1} P M_{-j} \tag{17}$$

Proof: It holds that

$$cov\left[\phi_j, y_j|D^k_{-j}\right] = M_{-j}P^T \qquad (18)$$

$$Var\left[y_j|D^k_{-j}\right] = PM_{-j}P^T + S_j \qquad (19)$$

In addition, by Lemma 1

$$\begin{aligned} Var\left[\phi_j|D^k\right] &= M_{-j} - cov\left[\phi_j, y_j|D^k_{-j}\right]\left(Var\left[y_j|D^k_{-j}\right]\right)^{-1} \\ &\times \; cov\left[\phi_j, y_j|D^k_{-j}\right]^T. \end{aligned}$$

Using eqs. (18,19), eq. (17) is finally obtained. ■

Remark 10: The issue of confidence intervals is what makes the real difference between the kernel-based machine learning approach and the Bayesian one. A similar situation is found in the literature on smoothing splines [37]: point estimates are usually worked out as the solution to Tikhonov-type variational problems without necessarily referring to prior distributions. However, when coming to the computation of confidence intervals, the established literature [37] resorts to Bayesian formulas even though hyperparameters may be estimated by GCV minimization. In fact, computation of confidence intervals that propagates only the measurement error, without accounting for prior uncertainty on the unknown function, neglects the bias introduced by regularization. At present, the Bayesian approach appears to be a simple yet effective way to account for all type of uncertainties. Of course, care must be taken in the choice of the prior distribution in order to obtain realistic intervals.

V. Estimation of unknown hyperparameters via maximum marginal likelihood

Many learning problems involve a vector θ of unknown hyperparameters which have to be estimated from data. For example, assuming homoskedastic noise in eq. (3), that is $\sigma^2_{ij} = \sigma^2$ and recalling eq. (4), in our model the unknown hyperparameters can be grouped into the vector

$$\theta = \left(\begin{array}{ccc} \sigma^2 & \overline{\lambda}^2 & \tilde{\lambda}^2 \end{array}\right).$$

Moreover, θ may also include further hyperparameters characterizing the kernels $\overline{K}$ and $\widetilde{K}$. For instance, if $\overline{K}(x_1, x_2) = e^{-\|x_1-x_2\|^2/c}$, the positive scalar c may be regarded as a further unknown.

Hyperparameter estimation is here addressed by exploiting the developed Bayesian setting. In particular, we resort to the so-called Empirical Bayes approach (see e.g. [38], [3]) where, first, hyperparameters are estimated via marginal likelihood maximization (for alternative deterministic approaches see [39], [40] and see also [41] for a discussion about regularization and Bayesian methods for hyperparameters tuning). Then, in order to reconstruct the task functions, the maximum likelihood estimates are plugged into the formulas derived in the previous sections. Assuming that k tasks are available, θ is estimated as

$$\theta^{ML} = \arg\min_{\theta} J(y^k, \theta),$$

$$J(y^k, \theta) := \log[\det(Var[y^k|\theta])] + (y^k)^T Var[y^k|\theta]^{-1} y^k,$$

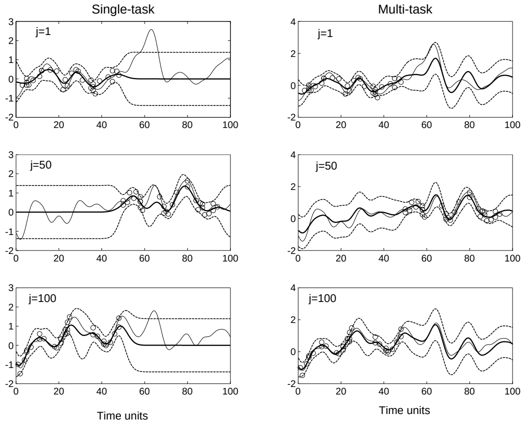

Fig. 1. Simulated data: comparison between single and multi-task learning. *Left* True $\mathbf{f}_j$ (thin line) and single-task estimates (thick line) with 95% confidence intervals (dashed lines) *Right* True $\mathbf{f}_j$ (thin line) and multi-task estimates $E\left[\mathbf{f}_j|y^{100}\right]$ (thick line) with 95% confidence intervals (dashed lines).

where, apart from a constant term, J is equal to the opposite of the logarithm of the likelihood $\mathbf{p}\left(y^k|\theta\right)$. Such objective function can be efficiently evaluated for any value of θ. In fact, the joint likelihood $\mathbf{p}\left(y^k|\theta\right)$ can be written in terms of conditional normal densities $\mathbf{p}(\cdot|\cdot)$ as follows

$$\mathbf{p}\left(y^k|\theta\right) = \mathbf{p}\left(y_1|\theta\right)\prod_{i=2}^{k}\mathbf{p}\left(y_i|D^{i-1}, \theta\right).$$

Recall that $A_i(\theta) := Var\left[y_i|D^{i-1}, \theta\right]$. Then, it holds that $\left(-\log\mathbf{p}\left(y^k|\theta\right)\right)$ is equal to

$$\begin{aligned} \alpha \;&+\; \frac{1}{2}\sum_{i=1}^{k}\log\det A_i(\theta) \\ &+\; \frac{1}{2}\sum_{i=1}^{k}\left(y_i - \breve{C}_i\xi^{i|i-1}(\theta)\right)^T A_i^{-1}(\theta)\left(y_i - \breve{C}_i\xi^{i|i-1}(\theta)\right) \end{aligned}$$

where $D^0 := \emptyset$ and α is a constant we are not concerned with. For any value of θ, $\xi^{i|i-1}$ and A_i can be determined by the recursive formulas in Proposition 6, see eqs. (7-9). Thus, an efficient evaluation of $J(y^k, \theta)$ is possible.

VI. Numerical examples

In this section, we apply the new multi-task algorithm to two simulated benchmarks and a pharmacological experiment.

A. Simulated data

This example is constructed by generating multiple tasks $\mathbf{f}_j$ that are realizations of longitudinal Gaussian processes. More precisely, $\mathbf{f}_j(x) = \overline{\mathbf{f}}(x) + \widetilde{\mathbf{f}}_j(x), x \in [0, 100]$, where $\overline{\mathbf{f}}(x)$ is the average task and $\widetilde{\mathbf{f}}_j$, $j = 1, \ldots, 100$, are the individual shifts. Gaussian-shaped auto-covariances are assumed:

$$\begin{aligned} cov\left[\overline{\mathbf{f}}(x_1), \overline{\mathbf{f}}(x_2)\right] &= e^{-\frac{(x_1-x_2)^2}{25}} \\ cov\left[\widetilde{\mathbf{f}}_j(x_1), \widetilde{\mathbf{f}}_j(x_2)\right] &= 0.25e^{-\frac{(x_1-x_2)^2}{25}} \qquad j = 1, 2, ..., 100 \end{aligned}$$

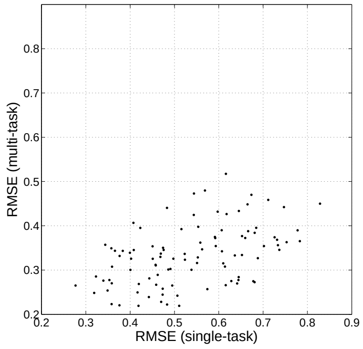

Fig. 2. Simulated data: comparison between single and multi-task learning. Scatterplot of $RMSE_j^{ST}$ and $RMSE_j^{MT}$.

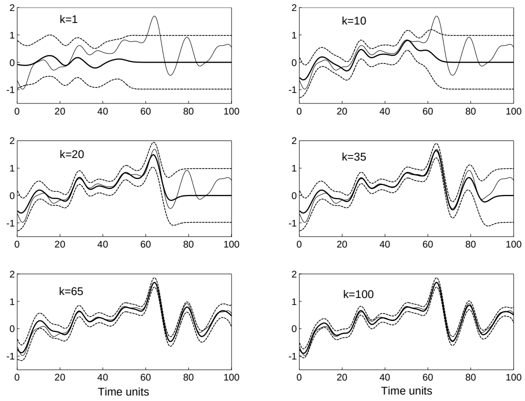

Fig. 3. Simulated data: comparison between single and multi-task estimation of the average task. True $\overline{\mathbf{f}}$ (thin line) and its estimate (thick line) for increasing values of k with 95% confidence intervals (dashed lines).

The average task curve is generated by drawing a single realization from the distribution of $\overline{\mathbf{f}}$, while 100 realizations of the shifts are independently drawn from the distribution of $\widetilde{\mathbf{f}}_j$. As for the inputs x_{ij}, $j = 1, \dots, 100$, they are integers randomly drawn from subsets N_j of $N = \{1, \dots, 100\}$. More precisely, for each task index j, 30 inputs x_{ij}, $i = 1, \dots, 30$ are drawn from a discrete uniform distribution having support $N_j = \{j, \dots, j \oplus 50\} \subset N$, where $\oplus$ denotes the mod-100 sum operator. Note that for each task there exists an input region $N \setminus N_j$ (a sampling "hole") where no data are collected, thus requiring nontrivial extrapolation. The outputs were generated according to eq. (3) with $\sigma_{ij}^2 = 0.4, \forall i, j$.

First, all tasks were estimated according to a single-task learning procedure. In other words, each task f_j was estimated using all and only the pairs (x_{ij}, y_{ij}), $i = 1, \dots, 30$. Note that the single-task estimate is obtained as a special case of the multi-task one by forcing $\overline{\lambda}^2 = 0$ in the formulas throughout the paper. The left panels of Fig. 1 show the results obtained in 5 tasks, together with their 95% confidence intervals. As expected, the tasks are poorly estimated in correspondence with the sampling holes due to the lack of information. Then, all tasks were estimated according to the multi-task approach presented in the paper: each task f_j was estimated using the complete dataset D^{100}. The right panels of Fig. 1 show the estimates and confidence intervals obtained in the same 5 tasks as in the left panels. By comparing left and right panels one can appreciate the benefit brought by the multi-task approach. In particular, the estimate uncertainty decreases in correspondence with the sampling holes. The advantage of multi-task learning can be also appreciated by looking at Fig. 2 that reports the $RMSE$ (Root Mean Square Error) for both single and multi-task estimates. The multi-task $RMSE_j^{MT}$ for the j-th task was defined as

$$RMSE_j^{MT} = \sqrt{\frac{1}{100} \int_0^{100} \left(\mathbf{f}_j(x) - \boldsymbol{\xi}_j^{100}(x) \right)^2 dx},$$

and the single-task $RMSE_j^{ST}$ was defined in a similar way. Finally, letting $R_j := \frac{RMSE_j^{MT}}{RMSE_j^{ST}}$ measure the $RMSE$ reduction when passing from single-task to multi-task estimation, the average R_j value over the 100 tasks was equal to 0.67.

Next, we consider iterative online multi-task learning for what concerns the average task $\overline{\mathbf{f}}$. More precisely, the estimates $E[\overline{\mathbf{f}}(x)|D^k]$ for $k = 1, \dots, 100$ were computed using the recursions derived in Section III. In Fig. 3 we display the true function $\overline{\mathbf{f}}(x)$ and its estimate, together with 95% confidence intervals, for some increasing values of k. For small values of k, no measurements are available in the rightmost part of X, which explains the shape of confidence intervals that get larger on the right. As k increases, incoming information is efficiently exploited in order to improve the estimate and reduce the size of confidence bounds. Not surprisingly, for $k = 50$, the estimate is already satisfactory since the whole domain X has been sampled. Finally, notice that, in this example, $n^{100} = 3000$, while $\breve{n}^{100} = 100$. Thus, without the method of the present paper, the multi-task learning problem would call for the solution of a system of 3000 linear equations. Conversely, by the new method, the solution is obtained by solving a sequence of linear systems whose order are always less than 100.

B. Real pharmacokinetic data

Multi-task learning was applied to a data set related to xenobiotics administration in 27 human subjects, see [42] and Section 5.2 in [14]. In the fully sampled dataset, 8 samples were collected in each subject at $0.5, 1, 1.5, 2, 4, 8, 12, 24$ hours after a bolus administration. Data are known to have a 10% coefficient of variation, i.e. $\sigma_{ij}^2 = (0.1 y_{ij})^2$. The 27 experimental concentration profiles are displayed in Fig. 4, together with the average profile. Given the number of subjects, such average profile can be regarded as a reasonable estimate of the average task $\overline{\mathbf{f}}$. The whole dataset, consisting of 216 pairs (x_{ij}, y_{ij}), $i = 1, \dots, 8$, $j = 1, \dots, 27$, was split in a training and a test set. In particular, for training we consider a sparse sampling schedule with only 3 measurements per subject, randomly chosen within the 8 available data. Let

$$W(t_1, t_2) = \frac{t_1 t_2 \min\{t_1, t_2\}}{2} - \frac{(\min\{t_1, t_2\})^3}{6}.$$

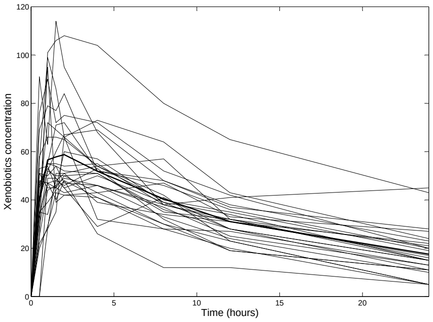

Fig. 4. Real pharmacokinetic data: xenobiotics concentrations after a bolus administration in 27 human subjects obtained by linearly interpolating noisy samples: average (thick) and individual profiles.

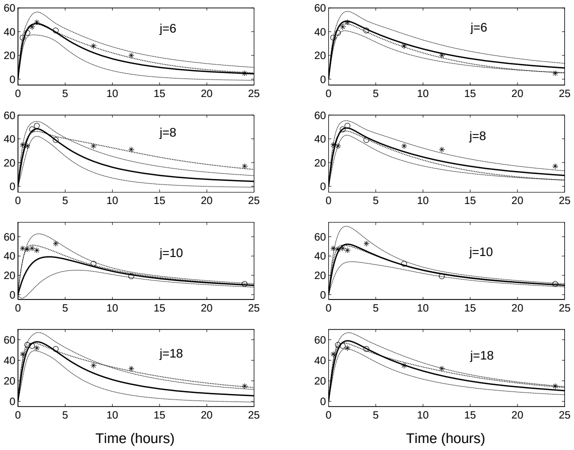

Fig. 5. Real pharmacokinetic data: single task (left) and multi-task (right) estimates (thick line) of 4 representative subjects with 95% confidence intervals (dashed lines) using only three data (circles) for each of the 27 subjects. The other five ”unobserved” data (asterisks) are also plotted. Dotted lines denote the estimates obtained by using the full sampling grid.

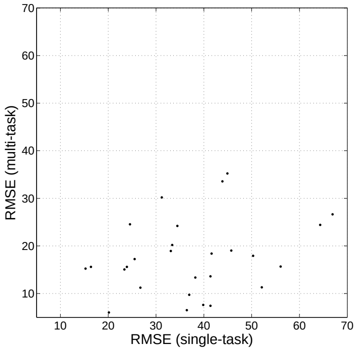

Fig. 6. Real pharmacokinetic data: comparison between single and multi-task learning. Scatterplot of $RMSE_j^{ST}$ and $RMSE_j^{MT}$.

denote the autocovariance of an integrated Wiener process having zero initial conditions at $t = 0$ and unitary intensity. With reference to (4), it is assumed that

$$\overline{K}(x_1, x_2) = \widetilde{K}_j(x_1, x_2) = W(h(x_1), h(x_2)), \tag{20}$$

$$h(x) = \frac{1}{1 + x/\beta}. \tag{21}$$

The aim of the transformation $h(x)$, originally introduced in [14], is to account for the non-stationary nature of pharmacological responses. In fact, in these experiments there is a greater variability for small values of t, followed by an asymptotic decay to zero. Due to the structure of $h(x)$, it follows that the prior variances of both $\overline{\mathbf{f}}$ and $\widetilde{\mathbf{f}}_j$ tend to zero as t goes to infinity. In particular, recalling that $\overline{\mathbf{f}}$ and $\widetilde{\mathbf{f}}_j$ are assumed to be zero-mean, this implies $\overline{\mathbf{f}}(+\infty) = \widetilde{\mathbf{f}}_j(+\infty) = 0$. Following [14], the parameter β was set equal to 3.0. To account for the fact that the initial plasma concentration is zero, a zero variance virtual measurement in $t = 0$ was added for all tasks.

According to the Empirical Bayes approach described in Section V, the hyperparameters, i.e. $\overline{\lambda}^2$ and $\widetilde{\lambda}^2$, were estimated via likelihood maximization. The left and right panels of Fig. 5, display results obtained by using the single-task and the multi-task approach, respectively. In particular, we display the data and the estimated curves with their 95% confidence intervals. In addition, each panel shows the estimates obtained by employing full sampling: it is apparent that the multi-task estimates are closer to these reference curves. One can also notice a good predictive capability with respect to the other five “unobserved” data. In this respect, let I^f and I_j^r denote the full and reduced sampling grid in the j-th subject. Define also the set $I_j = I^f \diagdown I_j^r$, whose cardinality is 5. For each subject we computed the quantity

$$RMSE_j^{MT} = \sqrt{\frac{\sum_{i \in I_j} (y_{ij} - \boldsymbol{\xi}_j^{27}(x_{ij}))^2}{5}}$$

as well as the single-task $RMSE_j^{MT}$defined in a similar way. Fig. 6 compares the $RMSE$ of single-task and multi-task estimates. The average $RMSE$ ratio defined as in the previous subsection was equal to 0.54.

Notice that the number of training inputs $\breve{n}^{27} = 8$ is about ten times smaller than the number of training examples $n^{27} = 81$. Therefore, the algorithm proposed in this paper enjoys about a 1000-fold reduction of computational effort with respect to formulas in [14].

In this experiment, single and multi-task learning provide similar results when full sampling is used. However, it is worth stressing that in real pharmacokinetic experiments such full sampling is quite an exception, i.e. very few data per subject are typically available. Thus, the experiment shows that multi-task learning proves effective in these realistic situations.

C. Simulated glucose data

Multi-task learning was finally applied to reconstruct glucose profiles in plasma during an intravenous glucose tolerance test (IVGTT) in which a glucose dose is injected in plasma at the beginning of the experiment [43]. Simulated data were generated by using the minimal model of glucose kinetics (MM) [44] which, since its inception in the late seventies, has been used in hundreds of papers to describe glucose and insulin dynamics after a glucose perturbation [43]. In particular, during an IVGTT, MM equations are:

$$\begin{cases} \dot{G}(t) = -\left[S_G + X(t)\right] G(t) + G_b S_G + \frac{u(t)}{V} \\ \dot{X}(t) = -p_2 \left[X(t) - S_I(I(t) - I_b)\right] \\ G(0) = G_b, \quad X(0) = 0 \end{cases} \tag{22}$$

In (22), $G(t)$ $(mgdl^{-1})$ and $I(t)$ (μUml^{-1}) are glucose and insulin concentration in plasma, respectively, G_b and I_b are glucose and insulin baseline values before glucose perturbation, respectively, S_I, S_G, p_2 and V are the MM parameters. Finally, $u(t)$ is ideally a Dirac delta centered in 0 with area equal to the injected glucose dose.

A log-normal probability density function for MM parameters was derived by exploiting the estimates reported in Table 1 of [45] obtained by 16 IVGTT experiments of length 240 minutes performed in normal subjects (see [45] for details). A continuous-time Gaussian prior for $I(t)$ was derived by first estimating via cubic smoothing splines the 16 insulin profiles using insulin plasma samples collected during the same experiments. Then, the sample mean and autocovariance of $I(t)$ was computed from the estimated time-courses. One thousand synthetic subjects were randomly generated from the prior distribution of model parameters and insulin profile. In particular, G_b was fixed to 120 $(mgdl^{-1})$. Furthermore, to account for the fact that in real experiments the injected dose is not an ideal Dirac delta, u was assigned a Gaussian profile, with support only on the positive axis, SD randomly drawn from a uniform distribution on the interval $[0, 1]$ min and area equal to 300 (mg).

Let Ω, expressed in minutes, be the set containing 30 sampling instants $\{t_k\}$ given by

$$\Omega = \left\{ \begin{array}{c} 1,2,3,4,6,8,10,12,14,16,18,20,25 \\ 30,35,40,45,50,60,70,80,90,100 \\ 120,140,160,180,200,220,240 \end{array} \right\}$$

We assume that in any of the 1000 subjects only 5 glucose measurements are available, being collected at different input locations extracted from Ω. To be more specific, we divided Ω in 5 subgrids, given by $\{1,2,3,4,6,8\}$, $\{10,12,14,16,18,20\}$ and so on. Then, the sampling grid relative to a subject is defined by randomly drawing one input location from each of the 5 subgrids. Measurements were then corrupted by a white normal noise with a 5% coefficient of variation, a value which is assumed known during the learning process. Glucose data were pre-processed by first subtracting the basal value from each profile. In addition, to account for the fact that the initial plasma concentration is zero, as in Section VI-B a zero variance virtual measurement in $t = 0$ was added for all tasks.

The proposed multi-task learning algorithm was tested on the 1000 synthetic subjects. The kernels reported in eq. (20)-(21) were adopted with β in eq. (21) set to 30. The Empirical Bayes approach described in Section V was used to estimate hyperparameters $\overline{\lambda}^2$ and $\widetilde{\lambda}^2$ via marginal likelihood maximization. Fig. 7 plots the estimated average glucose profile in plasma. The left and right panels of Fig. 8 show results obtained in 4 representative subjects by using the single-task and the multi-task approach, respectively. They display the data, the estimated curves with their 95% confidence intervals and the true glucose profile. One can notice that the multi-task estimates are closer to truth, with confidence intervals being much narrower and more informative than those obtained by the single-task approach.

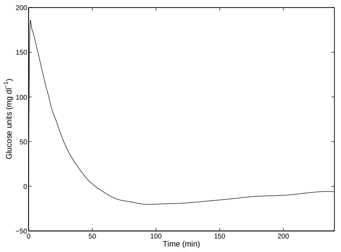

Fig. 7. Simulated glucose data: estimated average curve obtained by multi-task approach applied to 1000 IVGTT responses

TABLE I
SIMULATED GLUCOSE DATA: AVERAGE $RMSE$ RATIO AS A FUNCTION OF THE NUMBER OF MEASUREMENTS COLLECTED IN EACH SUBJECT

# of measurements per subject	5	10	15	30
Average R_j	0.23	0.33	0.46	0.49

The multi-task $RMSE_j^{MT}$ for the j-th task was defined as

$$RMSE_j^{MT} = \sqrt{\frac{1}{240}\int_0^{240} \left(\mathbf{f}_j(x) - \boldsymbol{\xi}_j^{1000}(x)\right)^2 dx},$$

and the single-task $RMSE_j^{ST}$ was defined in a similar way. Fig. 9 compares the $RMSE$ of the 1000 single-task and multi-task estimates. Remarkably, the average $RMSE$ ratio was equal to 0.23.

In Table 1 we also report the average $RMSE$ ratios obtained by increasing the number of measurements collected in any subject by means of subgrids of Ω defined by using the same rationale previously adopted, e.g. when 10 samples are taken, the 10 subgrids are given by $\{1,2,3\}$, $\{4,6,8\}$ and so on. It is interesting to notice that in this case, even when 30 measurements per subject are used, multi-task estimator performs much better than the single-task one.

Finally, notice that, without the method of the present paper, this problem would call for inverting matrices whose size is 30000 × 30000 when dealing with 30 measurements per subject, while the algorithm proposed in this paper returns the solution by solving a sequence of linear systems whose order never exceeds 30.

VII. CONCLUSIONS

The simultaneous learning of multiple tasks may significantly improve learning performances when limitations are imposed on the number and/or locations of samples collected in each single task. However, a potential drawback is the computational complexity involved by the joint processing of the whole dataset. To make an example, when using

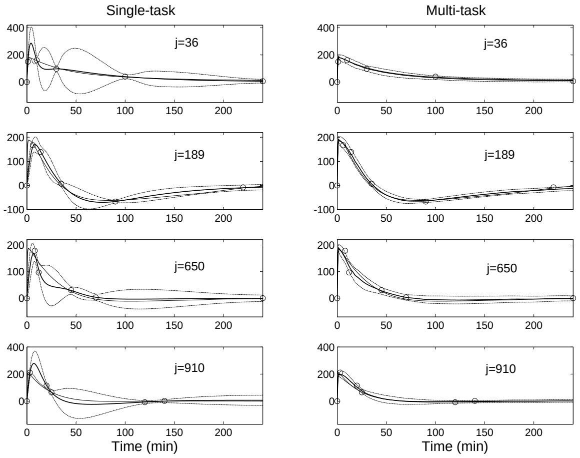

Fig. 8. Simulated glucose data: comparison between single and multi-task learning. *Left* True $\mathbf{f}_j$ (thin line) and single-task estimates (thick line) with 95% confidence intervals (dashed lines) *Right* True $\mathbf{f}_j$ (thin line) and multi-task estimates $E\left[\mathbf{f}_j|y^{1000}\right]$ (thick line) with 95% confidence intervals (dashed lines).

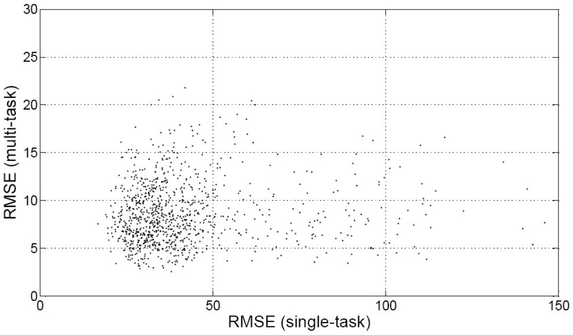

Fig. 9. Simulated glucose data: comparison between single and multi-task learning. Scatterplot of $RMSE_j^{ST}$ and $RMSE_j^{MT}$.

regularized kernel methods with quadratic loss functions, the number of operations scales with the cube of the overall number of examples. In the present paper, this computational problem has been addressed for a class of multi-task learning problems, in which each single task is modeled as the sum of an average function common to all tasks and an individual shift specific for each task. The problem has been given a Bayesian formulation under the assumption that the unknown tasks are Gaussian random fields.

The main contribution of the paper is a recursive learning scheme that efficiently updates estimates and variances exploiting the possible presence of repeated input samples. In addition to being interesting in its own, the on-line algorithm has the potential to greatly reduce the computational effort and memory occupation, especially when the number of distinct inputs is much smaller than the overall number of examples. The new algorithm has been tested on two simulated benchmarks and a set of real pharmacokinetic data.

It would be interesting to investigate the existence of efficient numerical implementations also for other classes of multi-task kernels. We conjecture that substantial computational gains can be obtained only for classes of kernels exhibiting rather particular structures. The one considered in this paper, albeit specific, has practical relevance. In fact, the decomposition of individual tasks as the sum of an average and an individual shift has already been successfully employed in biomedical data analysis [12], [15], [14].

ACKNOWLEDGMENT

This research has been partially supported by FIRB Project "Learning theory and application", and by the PRIN Projects "New Methods and Algorithms for Identification and Adaptive Control of Technological Systems" and "Artificial Pancreas: physiological models, control algorithms and clinical test". Data described in [45] were downloaded from the website of the Resource Facility for Population Kinetics (http://www.rfpk.washington.edu).

APPENDIX A: TECHNICAL LEMMAS

Lemma 11: We have:

(a)

$$E\left[\bar{\mathbf{f}}(x)|\breve{f}^k, D^k\right] = E\left[\bar{\mathbf{f}}(x)|\breve{f}^k\right], \forall x \in X.$$

In particular,

$$E\left[\breve{f}^{k+1}|\breve{f}^k, D^k\right] = E\left[\breve{f}^{k+1}|\breve{f}^k\right].$$

(b)

$$E\left[\tilde{\mathbf{f}}_j(x)|\bar{f}_j, D^k\right] = E\left[\tilde{\mathbf{f}}_j(x)|\bar{f}_j, D_j\right].$$

Proof:

Point (a) follows by showing that

$$\mathbf{p}\left(\bar{\mathbf{f}}(\cdot)|\breve{f}^k, D^k\right) = \mathbf{p}\left(\bar{\mathbf{f}}(\cdot)|\breve{f}^k\right).$$

In fact,

$$\begin{aligned}\mathbf{p}\left(\bar{\mathbf{f}}(\cdot)|\breve{f}^k, D^k\right) &= \mathbf{p}\left(\bar{\mathbf{f}}(\cdot)|\breve{f}^k, e^k\right) = \frac{\mathbf{p}\left(\bar{\mathbf{f}}(\cdot), \breve{f}^k, e^k\right)}{\mathbf{p}\left(\breve{f}^k, e^k\right)} \\ &= \frac{\mathbf{p}\left(\bar{\mathbf{f}}(\cdot), \breve{f}^k\right)\mathbf{p}\left(e^k\right)}{\mathbf{p}\left(\breve{f}^k\right)\mathbf{p}\left(e^k\right)} = \mathbf{p}\left(\bar{\mathbf{f}}(\cdot)|\breve{f}^k\right).\end{aligned}$$

As for point (b), it follows by showing that

$$\mathbf{p}\left(\tilde{\mathbf{f}}_j(\cdot)|\bar{f}_j, D^k\right) = \mathbf{p}\left(\tilde{\mathbf{f}}_j(\cdot)|\bar{f}_j, D_j\right).$$

In fact,

$$\begin{aligned}\mathbf{p}\left(\tilde{\mathbf{f}}_j(\cdot)|\bar{f}_j, D^k\right) &= \frac{\mathbf{p}\left(\tilde{\mathbf{f}}_j(\cdot), y^k, \bar{f}_j\right)}{\mathbf{p}\left(y^k, \bar{f}_j\right)} \\ &= \frac{\mathbf{p}\left(y^k|\tilde{\mathbf{f}}_j(\cdot), \bar{f}_j\right)\mathbf{p}\left(\tilde{\mathbf{f}}_j(\cdot), \bar{f}_j\right)}{\mathbf{p}\left(y^k, \bar{f}_j\right)}\end{aligned}$$

Now, let y_{-j}^k denote the vector containing all collected data but those regarding y_j, i.e. $y_{-j}^k = y^k \setminus y_j$, and let D_{-j}^k be defined in a similar way. Then,

$$\begin{aligned}\mathbf{p}\left(y^k|\tilde{\mathbf{f}}_j(\cdot), \bar{f}_j\right) &= \mathbf{p}\left(y_j|\tilde{\mathbf{f}}_j(\cdot), \bar{f}_j, D_{-j}^k\right)\mathbf{p}\left(y_{-j}^k|\tilde{\mathbf{f}}_j(\cdot), \bar{f}_j\right) \\ &= \mathbf{p}\left(y_j|\tilde{\mathbf{f}}_j(\cdot), \bar{f}_j\right)\mathbf{p}\left(y_{-j}^k|\bar{f}_j\right)\end{aligned}$$

(where the last equality exploits the independence assumptions) so that we obtain

$$\begin{aligned}
\mathbf{p}\left(\widetilde{\mathbf{f}}_j(\cdot)|\overline{f}_j, D^k\right) &= \mathbf{p}\left(y^k_{-j}|\overline{f}_j\right) \frac{\mathbf{p}\left(y_j|\widetilde{\mathbf{f}}_j(\cdot),\overline{f}_j\right)\mathbf{p}\left(\widetilde{\mathbf{f}}_j(\cdot),\overline{f}_j\right)}{\mathbf{p}\left(y^k,\overline{f}_j\right)} \\
&= \mathbf{p}\left(y^k_{-j}|\overline{f}_j\right) \frac{\mathbf{p}\left(\widetilde{\mathbf{f}}_j(\cdot),y_j,\overline{f}_j\right)}{\mathbf{p}\left(y^k|\overline{f}_j\right)\mathbf{p}\left(\overline{f}_j\right)} \\
&= \frac{\mathbf{p}\left(y^k_{-j}|\overline{f}_j\right)}{\mathbf{p}\left(y^k_{-j}|\overline{f}_j\right)\mathbf{p}\left(y_j|\overline{f}_j\right)} \frac{\mathbf{p}\left(\widetilde{\mathbf{f}}_j(\cdot),y_j,\overline{f}_j\right)}{\mathbf{p}\left(\overline{f}_j\right)} \\
&= \frac{\mathbf{p}\left(\widetilde{\mathbf{f}}_j(\cdot),y_j,\overline{f}_j\right)}{\mathbf{p}\left(y_i|\overline{f}_j\right)\mathbf{p}\left(\overline{f}_j\right)} = \mathbf{p}\left(\widetilde{\mathbf{f}}_j(\cdot)|\overline{f}_j, D_j\right).
\end{aligned}$$

■

Lemma 12: We have

$$\begin{aligned}
Var\left[y_{k+1}|D^k\right] &= \breve{C}_{k+1}\breve{V}^{k+1|k}\breve{C}^T_{k+1} + \widetilde{V}_{k+1} + S_{k+1}, \\
cov\left[\breve{f}^{k+1}, y_{k+1}|D^k\right] &= Var\left[\breve{f}^{k+1}|D^k\right]\breve{C}^T_{k+1}, \\
E\left[y_{k+1}|D^k\right] &= \breve{C}_{k+1}\breve{\xi}^{k+1|k}.
\end{aligned}$$

Proof: It suffices to exploit eq. (5), replacing y_{k+1} with $\breve{C}_{k+1}\breve{f}^{k+1} + \widetilde{f}_{k+1} + \epsilon_{k+1}$, and recall the independence assumptions. ■

The following lemma is an extension of Lemma 1 in the Appendix of [14]. It can also be seen as a special case of Lemma 1 in [31]. It is worth remarking that, differently from the statement in [14], here the symbol z denotes a vector (in place of a scalar) and the weaker condition $V > 0$, $\Sigma_v > 0$ (in place of $\Sigma > 0$) is invoked. Nevertheless, the proof is completely analogous and is therefore omitted.

Lemma 13: Let y, v and η be random vectors and F be a matrix such that

$$y = F\eta + v,$$

Let also $V > 0, \Sigma_v > 0$,

$$\begin{bmatrix} z \\ \eta \\ v \end{bmatrix} \sim \mathbf{N}(0,\Sigma), \qquad \Sigma = \begin{bmatrix} U & \Gamma & 0 \\ \Gamma^T & V & 0 \\ 0 & 0 & \Sigma_v \end{bmatrix}.$$

Then,

$$Var[z|y] = Var[z|\eta] + Var[E[z|\eta]|y],$$

where

$$\begin{aligned}
Var[z|\eta] &= U - \Gamma V^{-1}\Gamma^T, \\
Var[E[z|\eta]|y] &= \Gamma V^{-1} Var[\eta|y] V^{-1}\Gamma^T, \\
Var[\eta|y] &= \left(F^T\Sigma_v^{-1}F + V^{-1}\right)^{-1}.
\end{aligned}$$

Proof of Lemma 8:
It holds that

$$cov\left[\overline{\phi}_j, y_j|D^k_{-j}\right] = Var\left[\overline{\phi}_j|D^k_{-j}\right]P^T, \quad (23)$$

$$Var\left[y_j|D^k_{-j}\right] = PVar\left[\overline{\phi}_j|D^k_{-j}\right]P^T + \widetilde{V}_j + S_j \quad (24)$$

$$Var\left[\phi_j|D^k_{-j}\right] = Var\left[\overline{\phi}_j|D^k_{-j}\right] + Var\left[\widetilde{\phi}_j\right]. \quad (25)$$

Then, the following relation holds

$$\begin{aligned}
Var\left[\overline{\phi}_j|D^k\right] &= Var\left[\overline{\phi}_j|D^k_{-j}\right] - cov\left[\overline{\phi}_j, y_j|D^k_{-j}\right] \\
&\times \left(Var\left[y_j|D^k_{-j}\right]\right)^{-1} cov\left[\overline{\phi}_j, y_j|D^k_{-j}\right]^T \\
&= Var\left[\overline{\phi}_j|D^k_{-j}\right] - Var\left[\overline{\phi}_j|D^k_{-j}\right]P^T \\
&\times \left(PVar\left[\overline{\phi}_j|D^k_{-j}\right]P^T + \widetilde{V}_j + S_j\right)^{-1} PVar\left[\overline{\phi}_j|D^k_{-j}\right] \\
&= \left(\left(Var\left[\overline{\phi}_j|D^k_{-j}\right]\right)^{-1} + P^T\left(\widetilde{V}_j + S_j\right)^{-1}P\right)^{-1},
\end{aligned}$$

where the second equality makes use of eqs. (23,24) while the last one exploits the matrix inversion lemma, see e.g. [36]. Then, eq. (15) is obtained using eq. (25). Finally, to obtain eq. (16), consider eq. (6) and notice that $Var\left[\overline{\phi}_j|D^k\right]$ can be obtained by resorting to Lemma 13 with the following assignments:

$$y = y^k, \qquad z = \overline{\phi}_j, \qquad \eta = \breve{f}^k, \qquad v = \nu^k, \qquad F = \breve{C}^k.$$

Appendix B: Processing new measurements associated with a previous task

We consider now a situation where data from k distinct tasks have been already processed and additional examples relative to the j-th task, $j \leq k$, become available. In order to extend the computational scheme to such a case, it is useful to denote by y_j^+ the vector of new output data associated with the j-th task and by x_j^+ the vector containing the corresponding input values, whose dimension is n_j^+. For the sake of simplicity, we assume that $\breve{x}^k$ and x_j^+ do not have common elements. Let $\breve{f}^{k^+}$ denote the vector whose components are the elements of the set $\{\overline{\mathbf{f}}(x), x \in \breve{x}^{k^+}\}$ where $\breve{x}^{k^+} = \breve{x}^k \bigcup x_j^+$. Let also $\overline{f}_j^+$ indicate the vector whose components are $\{\overline{\mathbf{f}}(x), x \in x_j^+\}$, while $\widetilde{f}_j^+$ is the vector with components $\{\widetilde{\mathbf{f}}_j(x), x \in x_j^+\}$. Letting ϵ_j^+ denote the noise vector affecting y_j^+:

$$\begin{bmatrix} y_j \\ y_j^+ \end{bmatrix} = \begin{bmatrix} \overline{f}_j \\ \overline{f}_j^+ \end{bmatrix} + \begin{bmatrix} \widetilde{f}_j \\ \widetilde{f}_j^+ \end{bmatrix} + \begin{bmatrix} \epsilon_j \\ \epsilon_j^+ \end{bmatrix}$$

Let also

$$y^{k^+} = \begin{bmatrix} y^k \\ y_j^+ \end{bmatrix}$$

while D^{k^+} is the training set given by the union of D^k and the new input-output pairs defined by x_j^+ and y_j^+. Since data y_j have already been considered in the previous steps, the estimate $\breve{\xi}^{k^+|k^+}$ is computed according to eqs. (7)-(12) by replacing

- the superscript "$k+1$" with k^+ (e.g. $\breve{f}^{k+1}$ is replaced by $\breve{f}^{k^+}$ and so on)
- $\breve{r}^k_{k+1}$ with $cov\left[\overline{f}_j^+, \breve{f}^k\right]$
- $\breve{V}^{k+1}$ with

$$\breve{V}^{k^+} := Var\begin{bmatrix} \breve{f}^k \\ \overline{f}_j^+ \end{bmatrix}$$

- $\widetilde{V}_{k+1}$ and S_{k+1} with $Var\left[\widetilde{f}_j^+\right]$ and $Var\left[\epsilon_j^+\right]$, respectively
- $\breve{C}_{k+1}$ with the matrix $\breve{C}_{k+}$ such that

$$E\left[\overline{f}_j^+ | D^{k^+}\right] = \breve{C}_{k+}\breve{\xi}^{k^+|k}$$

- y_{k+1} with y_j^+

Then, if $q \neq j$,

$$\boldsymbol{\xi}_q^{k^+}(x) = \overline{\lambda}^2 \sum_{i=1}^{\breve{n}^k+n_j^+} a_i \overline{K}(x, \breve{x}_i^{k^+}) + \widetilde{\lambda}^2 \sum_{i=1}^{n_j} b_{iq}\widetilde{K}(x, x_{iq}),$$

else

$$\boldsymbol{\xi}_q^{k^+}(x) = \overline{\lambda}^2 \sum_{i=1}^{\breve{n}^k+n_j^+} a_i \overline{K}(x, \breve{x}_i^{k^+}) + \widetilde{\lambda}^2 \sum_{i=1}^{n_j+n_j^+} b_{iq}\widetilde{K}(x, x_{iq}),$$

where

$$\begin{aligned} a &= \left(\breve{V}^{k^+}\right)^{-1} \breve{\xi}^{k^+|k^+} \\ b_q &= \begin{cases} \left(\widetilde{V}_q + S_q\right)^{-1} \left(y_q - \breve{C}_q \breve{\xi}^{k^+|k^+}\right), & q \neq j \\ \left(\widetilde{V}_q^+ + S_q^+\right)^{-1} \left(\begin{bmatrix} y_q \\ y_q^+ \end{bmatrix} - \breve{C}_q \breve{\xi}^{k^+|k^+}\right), & q = j. \end{cases} \end{aligned}$$

and

$$\widetilde{V}_q^+ := Var\begin{bmatrix} \widetilde{f}_q \\ \widetilde{f}_q^+ \end{bmatrix} \qquad S_q^+ := Var\begin{bmatrix} \epsilon_q \\ \epsilon_q^+ \end{bmatrix}$$

REFERENCES

[1] T. Poggio and F. Girosi. Networks for approximation and learning. In *Proc. IEEE*, volume 7, pages 1481–1497, 1990.

[2] D. Barry. Nonparametric Bayesian regression. *The Annals of Statistics*, 14:934–953, 1986.

[3] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[4] L. B. Sheiner. The population approach to pharmacokinetic data analysis: rationale and standard data analysis methods. *Drug Metabolism Reviews*, 15:153–171, 1994.

[5] M. Davidian and D. M. Giltinan. *Nonlinear Models for Repeated Measurement Data*. Chapman and Hall, New York, 1995.

[6] J. A. Jacquez. *Compartmental analysis in biology and medicine*. Ann Arbor: University of Michigan Press, 1985.

[7] L. B. Sheiner, B. Rosenberg, and V. V. Marathe. Estimation of population characteristics of pharmacokinetic parameters from routine clinical data. *J. Pharmacokin. Biopharm.*, 5(5):445–479, 1977.

[8] S. Beal and L. Sheiner. *NONMEM User's Guide*. NONMEM Project Group, University of California, San Francisco, 1992.

[9] J. C. Wakefield, A. F. M. Smith, A. Racine-Poon, and A. E. Gelfand. Bayesian analysis of linear and non-linear population models by using the Gibbs sampler. *Applied Statistics*, 41:201–221, 1994.

[10] D. J. Lunn, N. Best, A. Thomas, J. C. Wakefield, and D. Spiegelhalter. Bayesian analysis of population PK/PD models: general concepts and software. *J. Pharmacokinet. Pharmacodyn.*, 29(3):271–307, 2002.

[11] K. E. Fattinger and D. Verotta. A nonparametric subject-specific population method for deconvolution: I. description, internal validation and real data examples. *Journal of Pharmacokinetics and Biopharmaceutics*, 23:581–610, 1995.

[12] P. Magni, R. Bellazzi, G. De Nicolao, I. Poggesi, and M. Rocchetti. Nonparametric AUC estimation in population studies with incomplete sampling: a Bayesian approach. *J. Pharmacokin. Pharmacodyn.*, 29(5/6):445–471, 2002.

[13] M. Neve, G. De Nicolao, and L. Marchesi. Nonparametric identification of pharmacokinetic population models via Gaussian processes. In *Proc. of 16th IFAC World Congress, Praha, Czech Republic*, 2005.

[14] M. Neve, G. De Nicolao, and L. Marchesi. Nonparametric identification of population models via Gaussian processes. *Automatica*, 97(7):1134–1144, 2007.

[15] F. Ferrazzi, P. Magni, and R. Bellazzi. Bayesian clustering of gene expression time series. In *Proc. of 3rd Int. Workshop on Bioinformatics for the Management, Analysis and Interpretation of Microarray Data (NETTAB 2003)*, pages 53–55, 2003.

[16] R. Caruana. Multi-task learning. *Machine Learning*, 28:41–75, 1997.

[17] S. Thrun and L. Pratt. *Learning to learn*. Kluwer, 1997.

[18] B. Bakker and T. Heskes. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research*, (4):83–99, 2003.

[19] J. Baxter. A Bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, (28):7–39, 1997.

[20] C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, 2005.

[21] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *J. Machine Learning Research*, 6:615–637, 2005.

[22] G. Pillonetto, G. De Nicolao, M. Chierici, and C. Cobelli. Fast algorithms for nonparametric population modeling of large data sets. *Automatica*, (to appear).

[23] A. Schwaighofer, V. Tresp, and K. Yu. Learning Gaussian process kernels via hierarchical Bayes. In *Advances in Neural Information Processing Systems 17*, volume 17, pages 1209–1216, 2005.

[24] N. D. Lawrence and J. C. Platt. Learning to learn with the informative vector machine. In *Proceedings of the International Conference in Machine Learning*, volume 69, page 65, 2004.

[25] K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 1012–1019, 2005.

[26] M. Seeger and M. I. Jordan. Sparse Gaussian process classification with multiple classes. Technical Report 661, Department of Statistics, University of California, Berkeley, 2004.

[27] J. O. Ramsay and B. W. Silverman. *Functional data analysis*. Springer-Verlag, 1997.

[28] J. O. Ramsay and C. J. Dalzell. Some tools for functional data analysis (with discussion). *Journal of the Royal Statistical Society, Series B*, 53:539–572, 1991.

[29] M. Neve, G. De Nicolao, and L. Marchesi. Nonparametric identification of population models: An mcmc approach. *IEEE Trans. on Biomedical Engineering*, 55:41–50, 2008.

[30] Z. Lu, T. Leen, Y. Huang, and D. Erdogmus. A reproducing kernel Hilbert space framework for pairwise time series distances. pages 624–631, 2008.

[31] L. Csató and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

[32] M. Opper. *Online Learning in Neural Networks*, chapter A Bayesian Approach to Online Learning. Cambridge University Press, 1998.

[33] G. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation of stochastic processes and smoothing by splines. *Ann. Math. Stat.*, 41:495–502, 1979.

[34] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Proceedings of the Annual Conference on Computational Learning Theory*, pages 416–426, Portland, OR, USA, 2001.

[35] A. N. Shiryaev. *Probability*. Springer, New York, NY, USA, 1996.

[36] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, N.J., USA, 1979.

[37] G. Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, 1990.

[38] J. S. Maritz and T. Lwin. *Empirical Bayes Method*. Chapman and Hall, 1989.

[39] A. Argyriou, C.A. Micchelli, and M. Pontil. Learning convex combinations of continuously parametrized basic kernels. In *Proc. of COLT 2005*, pages 338–352, 2005.

[40] A. Argyriou, R. Hauser, C.A. Micchelli, and M. Pontil. A dc algorithm for kernel selection. In *Proc. of of the 23rd International Conference on Machine learning*, pages 41–48, 2006.

[41] T. Evgeniou, M. Pontil, and O. Toubia. A convex optimization approach to modeling heterogeneity in conjoint estimation. *Marketing Science*, 26:805–818, 2007.

[42] M. Rocchetti and I. Poggesi. Comparison of the Bailer and Yeh methods using real data. In *The population approach: Measuring and managing variability in response, concentration and dose. Brussels, Belgium: European Cooperation in the Field of Scientific and Technical Research, European Commission*, pages 385–390, 1997.

[43] R. N. Bergman. Minimal model: perspective from 2005. *Hormone research*, 64:8–15, 2006.

[44] R. N. Bergman, Y. Z. Ider, C. R. Bowden, and C. Cobelli. Quantitative estimation of insulin sensitivity. *Am. J. Physiol. 236 (Endocrinol. Metab. Gastrointest. Physiol.)*, 5:E667–E677, 1979.

[45] P. Vicini and C. Cobelli. The iterative two-stage population approach to IVGTT minimal modeling: improved precision with reduced sampling. *Am. J. Physiol. Endocrinol. Metab.*, 280(1):179–186, 2001.