The recursive variational Gaussian approximation (R-VGA)

Marc Lambert, Silvere Bonnabel, Francis Bach

The recursive variational Gaussian approximation (R-VGA)

Marc Lambert
DGA/CATOD, Centre d'Analyse Technico-Opérationelle de Défense
& INRIA
marc-h.lambert@intradef.gouv.fr

Silvère Bonnabel
ISEA, Université de la Nouvelle-Calédonie
& MINES ParisTech, PSL University, Center for robotics
silvere.bonnabel@mines-paristech.fr

Francis Bach
INRIA - Ecole Normale Supérieure - PSL Research university
francis.bach@inria.fr

Abstract

We consider the problem of computing a Gaussian approximation to the posterior distribution of a parameter given N observations and a Gaussian prior. Owing to the need of processing large sample sizes N, a variety of approximate tractable methods revolving around online learning have flourished over the past decades. In the present work, we propose to use variational inference (VI) to compute a Gaussian approximation to the posterior through a single pass over the data. Our algorithm is a recursive version of the variational Gaussian approximation we have called recursive variational Gaussian approximation (R-VGA). We start from the prior, and for each observation we compute the nearest Gaussian approximation in the sense of Kullback-Leibler divergence to the posterior given this observation. In turn, this approximation is considered as the new prior when incorporating the next observation. This recursive version based on a sequence of optimal Gaussian approximations leads to a novel implicit update scheme which resembles the online Newton algorithm, and which is shown to boil down to the Kalman filter for Bayesian linear regression. In the context of Bayesian logistic regression the implicit scheme may be solved, and the algorithm is shown to perform better than the extended Kalman filter, while being far less computationally demanding than its sampling counterparts.

1 Introduction

Recent applications of probabilistic modeling and Bayesian inference involve large numbers of observations. In this setting, recursive algorithms that update the posterior distribution after each observation are desirable. In the present paper we seek a Gaussian approximation $q(\theta) = \mathcal{N}(\theta|\mu, P)$ to the posterior

$$p(\theta|y_1,\dots,y_N) = \frac{p_0(\theta)p(y_1,\dots,y_N|\theta)}{p(y_1,\dots,y_N)} = \frac{p_0(\theta)p(y_1,\dots,y_N|\theta)}{\int p_0(\theta)p(y_1,\dots,y_N|\theta)d\theta},$$

given N conditionally independent observations $y_1,\dots,y_N$ and a Gaussian prior p_0. Various methods exist for Gaussian approximation, for instance the Laplace approximation computes the Hessian of the log-posterior at the maximum a posteriori (MAP) to fit the covariance term, whereas moment matching methods estimate the mean and covariance matrix through direct integration [5]. Besides, variational inference (VI) has become widely used to approximate posteriors. Following this framework, we seek to minimize the KL divergence

between the true posterior and a Gaussian distribution $q(\theta) = \mathcal{N}(\theta|\mu, P)$ with parameters μ (mean vector) and P (covariance matrix) to be estimated in an online fashion. Since the observations are independent conditionally on θ we have:

$$\min_{\mu,P} \quad KL(q(\theta)||p(\theta|y_1,\ldots,y_N)) := \min_{\mu,P} \quad \mathbf{E}_q\left[\log \frac{q(\theta)}{p(\theta|y_1,\ldots,y_N)}\right]$$

$$= \min_{\mu,P} \quad \mathbf{E}_q\left[\log q(\theta) - \log p_0(\theta)\right] - \sum_{i=1}^{N} \mathbf{E}_q\left[\log p(y_i|\theta)\right] + \log p(y_1,\ldots,y_N).$$

The black box variational inference algorithm [19] attacks this VI optimization problem using Monte-Carlo samples $\theta_t \sim q$, combined with a Robbins-Monro stochastic gradient algorithm [20]. The natural gradient algorithm has also been used to improve the rate of convergence using second order information [13]. These algorithms need to parse the entire dataset at each step and online versions have been used on large size datasets to prevent memory overflow [22].

In this paper we introduce a recursive (online) version of variational Gaussian approximation we have called R-VGA. The method consists in letting $q_0 = p_0$ and then computing a Gaussian approximation $q_t(\theta) = \mathcal{N}(\theta_t|\mu_t, P_t)$ at each step $1 \leq t \leq N$ in the following recursive way. At step t, when observation y_t comes in, our approximation $q_t(\theta)$ exactly minimizes the Kullback-Leibler (KL) divergence to $\propto p(y_t|\theta)q_{t-1}(\theta)$, that is, the partial posterior based on the previous approximation q_{t-1} and latest observation y_t. After a single pass over the data, a Gaussian approximation $q(\theta) = q_N(\theta)$ to the full posterior based on a series of optimal approximations is thus obtained. Optimality comes at a price, though, as at each step fixed point equations need to be solved. These fixed point equations form a novel implicit update scheme which resembles the online Newton algorithm, except that it involves expectations over q_t instead of q_{t-1}. Note that, although they do not yield a readily implementable closed form solution to the optimisation problem, implicit schemes have recently gained interest owing to their inherent stability, see for instance [11] who considers an implicit scheme for online Newton descent applied to logistic regression [9].

The proposed R-VGA proves optimal when applied to Bayesian linear regression, and is then shown to boil down to the Kalman filter. When the observations belong to an exponential family and the model is linearized, we show also the R-VGA is algebraically equivalent to the extended Kalman filter and to the online natural gradient. This reminds of prior works of [17] and [7], that have made connections between these algorithms. In the present paper, the R-VGA is also applied to Bayesian logistic regression. We show the implicit update scheme may then be solved, and the R-VGA's numerical complexity is identical to the extended Kalman filter's, that is, $O(d^2N)$ where d is the dimension of parameter θ and N is the number of observations. Numerical experiments based on synthetic data illustrate that the R-VGA outperforms the extended Kalman filter (EKF) as well as other variants having similar computational cost. In some challenging cases it even beats the batch Laplace approximation in terms of divergence to the true posterior.

The paper is organized as follows. In Section 2, we introduce the recursive variational Gaussian approximation (R-VGA) scheme and derive an averaged version of the online Newton algorithm. We also show this second order algorithm can be reformulated in an Hessian free and derivative-free versions. In Section 3 we show that in the case where observations are Gaussian and linearly related to the hidden variable θ, the R-VGA is algebraically equivalent to the linear Kalman filter and the online Newton algorithm. We extend these results to the context of exponential family distributions with linearized models, and show the approximated explicit version of the R-VGA is then equivalent to the extended Kalman filter (EKF) and to the online natural gradient. In Section 4, R-VGA is applied to the logistic regression problem. We also introduce the quadratic Kalman filter as an alternative variant based on quadratic variational approximations, and recall the EKF equations for logistic regression. Finally numerical experiments allow for algorithm comparisons in Section 5.

2 The recursive variational Gaussian approximation (R-VGA)

The recursive variational approximation is detailed in Algorithm 1.

Algorithm 1 R-VGA

Require: $y_1, \ldots, y_N$, μ_0, P_0
  $q_0(\theta) \leftarrow \mathcal{N}(\theta|\mu_0, P_0)$
  for $t \leftarrow 1 : N$ **do**
    $\mu_t, P_t = \arg\min KL(\mathcal{N}(\theta|\mu_t, P_t)||p(y_t|\theta)q_{t-1}(\theta))$
    $q_t(\theta) \leftarrow \mathcal{N}(\theta|\mu_t, P_t)$
  end for
  return $q_N(\theta)$

Let Y_t denote the observations up to step t, that is, $Y_t = (y_1, \ldots, y_t)$, and q_t denote a Gaussian approximation for the distribution of θ at step t, that is, $q_t(\theta) = \mathcal{N}(\theta|\mu_t, P_t)$. The algorithm is based on the recursive update that consists in approximating at each step the distribution $p(\theta|Y_t)$ through the past Gaussian approximation $q_{t-1}(\theta)$. More precisely the recursive updates write:

$$
\begin{aligned}
&\mathbf{KL}(q_t(\theta)||p(\theta|Y_t)) := \int q_t(\theta) \log \frac{q_t(\theta)}{p(\theta|Y_t)} d\theta \\
&= \int q_t(\theta) \log \frac{q_t(\theta) p(y_t|Y_{t-1})}{p(y_t|\theta, Y_{t-1}) p(\theta|Y_{t-1})} d\theta \text{ (from Bayes' theorem on } (\theta, y_t)) \\
&= \int q_t(\theta) \log \frac{q_t(\theta) p(y_t|Y_{t-1})}{p(y_t|\theta) p(\theta|Y_{t-1})} d\theta \text{ (since } y_t \text{ is conditionally independent of } Y_{t-1} \text{ given } \theta) \\
&= \int q_t(\theta) \log \frac{q_t(\theta) p(Y_t)}{p(y_t|\theta) p(\theta|Y_{t-1}) p(Y_{t-1})} d\theta \text{ (multiplying by } \frac{p(Y_{t-1})}{p(Y_{t-1})}) \\
&\approx \int q_t(\theta) \log q_t(\theta) d\theta - \int q_t(\theta) \log p(y_t|\theta) d\theta - \int q_t(\theta) \log q_{t-1}(\theta) d\theta + \log p(Y_t) - \log p(Y_{t-1}).
\end{aligned}
$$

The latter approximation amounts to substituting $p(\theta|Y_{t-1})$ with the latest computed distribution $q_{t-1}(\theta)$. Based on this approximation, our goal is to compute at each step t the parameters μ_t and P_t of the variational distribution q_t which minimize the latter expression, that is,

$$
\underset{\mu_t, P_t}{\operatorname{argmin}} \quad \mathbf{E}_{q_t}[\log q_t(\theta) - \log q_{t-1}(\theta) - \log p(y_t|\theta)].
$$

We may compute the equations the minimizers must satisfy at each step t. In the next section we show these stationary equations can be written in a form that resembles the online Newton algorithm. Unlike the classical online Newton algorithm which uses a second order Taylor approximation as in the Laplace method, our algorithm is based on a variational approach which involves expectations.

2.1 The R-VGA as an averaged online Newton algorithm

Theorem 1 shows the optimal solution at iteration t of the R-VGA can be rewritten as an optimization step of an averaged version of the online Newton descent algorithm.

Theorem 1. *Suppose* $\log p(y|\theta)$ *is absolutely continuous with respect to* θ *and the observations* $y_1, \ldots, y_N$ *are independent conditionally on* θ*. Given a Gaussian prior distribution* q_0*, a sequence of Gaussian distributions*

$q_1, \ldots, q_t$ being solutions to the recursive variational Gaussian approximation (R-VGA) scheme of Algorithm 1 necessarily satisfy the following fixed point equations:

$$\mu_t = \mu_{t-1} + P_{t-1} \nabla_{\mu_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)] \tag{1}$$

$$P_t^{-1} = P_{t-1}^{-1} - 2\nabla_{P_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)]. \tag{2}$$

Those fixed point equations are equivalent to a second order form which may be viewed as an averaged version of the online Newton algorithm ("order 2 form"):

$$\mu_t = \mu_{t-1} + P_{t-1} \mathbf{E}_{q_t}[\nabla_\theta \log p(y_t|\theta)] \tag{3}$$

$$P_t^{-1} = P_{t-1}^{-1} - \mathbf{E}_{q_t}[\nabla_\theta^2 \log p(y_t|\theta)]. \tag{4}$$

Proof. We seek to minimize the quantity:

$$\min_{\mu_t, P_t} \ \mathbf{E}_{q_t}[\log q_t(\theta) - \log q_{t-1}(\theta) - \log p(y_t|\theta)]. \tag{5}$$

The critical point with respect to μ_t yields:

$$\begin{aligned}
&\nabla_{\mu_t} \mathbf{E}_{q_t}[\log q_t(\theta) - \log q_{t-1}(\theta) - \log p(y_t|\theta)] \\
&= \nabla_{\mu_t} \left(\frac{1}{2} \mu_t^T P_{t-1}^{-1} \mu_t - \mu_{t-1}^T P_{t-1}^{-1} \mu_t \right) - \nabla_{\mu_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)] \\
&= P_{t-1}^{-1} \mu_t - P_{t-1}^{-1} \mu_{t-1} - \nabla_{\mu_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)] = 0 \\
&\iff \mu_t = \mu_{t-1} + P_{t-1} \nabla_{\mu_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)].
\end{aligned} \tag{6}$$

The critical point with respect to P_t yields:

$$\begin{aligned}
&\nabla_{P_t} \mathbf{E}_{q_t}[\log q_t(\theta) - \log q_{t-1}(\theta) - \log p(y_t|\theta)] \\
&= \nabla_{P_t} \left(-\frac{1}{2} \log |P_t| + \frac{1}{2} \mathrm{Tr}(P_t P_{t-1}^{-1}) \right) - \nabla_{P_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)] \\
&= -\frac{1}{2} P_t^{-1} + \frac{1}{2} P_{t-1}^{-1} - \nabla_{P_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)] = 0 \\
&\iff P_t^{-1} = P_{t-1}^{-1} - 2\nabla_{P_t} \mathbf{E}_{q_t}[\log p(y_t|\theta)].
\end{aligned} \tag{7}$$

We have thus recovered equations (1) and (2). We operate now a change of variable: the derivative with respect to μ and P can be transformed into derivatives with respect to θ using the symmetry properties of the Gaussian distribution:

$$\nabla_\mu \mathcal{N}(\theta|\mu, P) = -\nabla_\theta \mathcal{N}(\theta|\mu, P) \tag{8}$$

$$\nabla_P \mathcal{N}(\theta|\mu, P) = \frac{1}{2} \nabla_\theta^2 \mathcal{N}(\theta|\mu, P). \tag{9}$$

Since $\log p(y|\theta)$ is absolutely continuous with respect to θ, we can interchange differentiation and integration and use the formula of integration by parts (known as the Bonnet & Price formulas [14] to recover equations (3) and (4):

$$\nabla_\mu \mathbf{E}_q[\log p(y|\theta)] = \int \log p(y|\theta) \nabla_\mu \mathcal{N}(\theta|\mu, P) d\theta \tag{10}$$

$$= \int \nabla_\theta \log p(y|\theta) \mathcal{N}(\theta|\mu, P) d\theta = \mathbf{E}_q[\nabla_\theta \log p(y|\theta)] \tag{11}$$

$$\nabla_P \mathbf{E}_q[\log p(y|\theta)] = \int \log p(y|\theta) \nabla_P \mathcal{N}(\theta|\mu, P) d\theta \tag{12}$$

$$= -\frac{1}{2} \int \nabla_\theta \log p(y|\theta) \nabla_\theta \mathcal{N}(\theta|\mu, P)^T d\theta = \frac{1}{2} \mathbf{E}_q[\nabla_\theta^2 \log p(y|\theta)], \tag{13}$$

which completes the proof. □

Equations (3) and (4) resemble the online Newton algorithm but where the iterates θ are averaged through expectations. Gradient averaging (over time) has attracted a lot of attention in optimization, see, e.g., [21]. However in our equations, the averaging consists of an expectation over the parameters we are currently estimating and leads to an implicit scheme. We will study this second order implicit scheme in the remainder of this paper and show it can be made explicit in the linear regression case, and can be solved for the logistic regression problem. Before that, the next corollary shows we can derive lower order schemes which are algebraically equivalent to the second order scheme.

Corollary 1.1. *The R-VGA updates, in particular the averaged version of the online Newton, can be rewritten in a Hessian-free version ("order 1 form"):*

$$\mu_t = \mu_{t-1} + P_{t-1}\mathbf{E}_{q_t}[\nabla_\theta \log p(y_t|\theta)] \tag{14}$$
$$P_t^{-1} = P_{t-1}^{-1} - P_t^{-1}\mathbf{E}_{q_t}[(\theta-\mu_t)\nabla_\theta \log p(y_t|\theta)^T], \tag{15}$$

and in a derivative-free version ("order 0 form"):

$$\mu_t = \mu_{t-1} + P_{t-1}P_t^{-1}\mathbf{E}_{q_t}[(\theta-\mu_t)\log p(y_t|\theta)] \tag{16}$$
$$P_t^{-1} = P_{t-1}^{-1} - P_t^{-1}\mathbf{E}_{q_t}[(\theta-\mu_t)(\theta-\mu_t)^T \log p(y_t|\theta)]P_t^{-1} + P_t^{-1}\mathbf{E}_{q_t}[\log p(y_t|\theta)]. \tag{17}$$

Proof. The proof is a direct consequence of Theorem 1. Rather than using integration by parts, we use (10) and (12):

$$\int \log p(y|\theta)\nabla_\mu \mathcal{N}(\theta|\mu,P)d\theta = \int \log p(y|\theta)P^{-1}(\theta-\mu)\mathcal{N}(\theta|\mu,P)d\theta \tag{18}$$
$$= P^{-1}\mathbf{E}_q[(\theta-\mu)\log p(y|\theta)] \tag{19}$$
$$\int \log p(y|\theta)\nabla_P \mathcal{N}(\theta|\mu,P)d\theta = \int \log p(y|\theta)(\frac{1}{2}P^{-1}(\theta-\mu)(\theta-\mu)^T P^{-1} - \frac{1}{2}P^{-1})\mathcal{N}(\theta|\mu,P)d\theta \tag{20}$$
$$= \frac{1}{2}P^{-1}\mathbf{E}_q[(\theta-\mu)(\theta-\mu)^T \log p(y|\theta)]P^{-1} - \frac{1}{2}P^{-1}\mathbf{E}_{q_t}[\log p(y|\theta)]. \tag{21}$$

Plugging this relation into the R-VGA equations (1) and (2), we recover the new derivative-free update equations of the theorem. The Hessian-free version is a variant where we use only one integration by parts in (12):

$$\int \log p(y|\theta)\nabla_P \mathcal{N}(\theta|\mu,P)d\theta = -\frac{1}{2}\int \log p(y|\theta)\nabla_\theta\nabla_\mu \mathcal{N}(\theta|\mu,P)^T d\theta \tag{22}$$
$$= \frac{1}{2}\int \nabla_\theta \log p(y|\theta)\nabla_\mu \mathcal{N}(\theta|\mu,P)^T d\theta \tag{23}$$
$$= \frac{1}{2}P^{-1}\mathbf{E}_q[(\theta-\mu)\nabla_\theta \log p(y|\theta)^T], \tag{24}$$

which completes the proof. □

These lower order versions open up for handling complex distributions $\log p(y|\theta)$ for which the Hessian or the gradient are difficult to compute or are not well-defined. The Hessian-free version provides a way to compute online the Hessian using only the gradient information. Online versions of Gauss-Newton methods approximate also the Hessian with a gradient, but in Equation (15) the iteration is exact, not approximated, while the scheme is implicit. Derivative-free optimization has been developed for stochastic optimisation of a non-smooth function [16]. If the function is smoothed with a Gaussian, integration by parts can be used to eliminate the gradient term. Stochastic derivative-free optimization is known to be much slower than the stochastic gradient counterpart because the smoothing process introduces a bias. Our derivative-free version

in Equation (16) is different since it introduces an adaptive step through the equations (17). However these equations are both implicit and not directly implementable.

All the equations we have derived until now do not provide explicit iterations. We will discuss now how to construct numerical schemes to implement them.

2.2 Discussion and closed-form approximations

We see the implementation of the main R-VGA fixed-point update equations (1)-(2) or any of its variants poses two main difficulties. First, the updates are implicit, i.e., the right-hand side depends on the parameters that one is seeking. Second, they require to compute an expectation over the distribution q_t. Let us start with the first issue, and devise an approximate explicit scheme.

Explicit approximation: A simple way to approximate the second order implicit scheme (3)-(4):

Implicit scheme (exact R-VGA)

$$\mu_t = \mu_{t-1} + P_{t-1}\mathbf{E}_{q_t}[\nabla_\theta \log p(y_t|\theta)], \tag{25}$$

$$P_t^{-1} = P_{t-1}^{-1} - \mathbf{E}_{q_t}[\nabla^2_\theta \log p(y_t|\theta)], \tag{26}$$

is to consider q_t to be close enough to q_{t-1} and to replace the expectation under q_t with the expectation under q_{t-1}, yielding the following

Explicit scheme (approximated R-VGA)

$$\mu_t = \mu_{t-1} + P_t\mathbf{E}_{q_{t-1}}[\nabla_\theta \log p(y_t|\theta)], \tag{27}$$

$$P_t^{-1} = P_{t-1}^{-1} - \mathbf{E}_{q_{t-1}}[\nabla^2_\theta \log p(y_t|\theta)]. \tag{28}$$

Note we have also changed P_{t-1} into P_t in the right hand side of (25), since this makes the above scheme optimal in the linear Gaussian case, see Section 3.1. In other cases it is suboptimal and the implicit scheme should be preferred.

Computation of the expectations: Even when using the explicit scheme, one needs to compute expectations under a variational distribution q which is the second issue we have raised. Most often they are analytically intractable, and various approaches have been advocated in the literature. A first approach is to use a Monte Carlo approximation. According to [8], in the particular case where θ is a scalar, the Monte Carlo estimator based on $\frac{1}{2}\mathbf{E}_q[\nabla^2 f(\theta)]$ has lower variance than the one based on $\frac{1}{2}\mathbf{E}_q[P^{-1}(\theta-\mu)\nabla_\theta f(\theta)^T]$, indicating the order 2 scheme shall be then preferred. A second approach to the computation of expectations is to use quadrature rules, that compute an approximation to the integral based on a finite number of so-called "sigma points". Such quadrature integrals have recently been advocated by [2] where a derivative-free form was also used in the context of batch variational inference.

In this paper we apply the R-VGA to both Bayesian linear and logistic regression. For logistic regression, we show we can circumvent both issues that have been raised, as solving the implicit scheme is amenable to a simple two-dimensional optimization problem, see Section 4.2), and regarding the computation of the expectations we derive analytical expressions using the inverse probit approximation of the logistic function, see Section 4.1. For linear regression we show in the next section the implicit equations can be rewritten as an explicit scheme.

3 Links with the Kalman filter, natural gradient descent, and online Newton

In the previous section, we have developed the R-VGA update equations and have shown these updates ressemble the averaged version of the online Newton scheme. We now describe in more detail this connection and show there is indeed an algebraic equivalence between the R-VGA scheme and the online Newton scheme in the linear case, or the nonlinear case under linearization assumptions. We establish also a connection with other second-order online algorithms like the Kalman filter or the natural gradient.

Let the stochastic loss function index at t be defined as:

$$\ell_t(\theta) = -\log p(y_t|\theta). \tag{29}$$

As before we suppose the observations to be independent conditionally on θ such that the likelihood writes $\log p(Y_t|\theta) = \sum_{i=1}^{t} \log p(y_i|\theta)$, so that the loss up to t shall be defined as:

$$L_t(\theta) = \sum_{i=1}^{t} \ell_i(\theta). \tag{30}$$

3.1 The linear regression case

We now assume the observations are Gaussian and are linearly related to θ such that $p(y_t|\theta) = \mathcal{N}(H_t\theta, R_t)$. In this case, the least mean squares cost function (30) for t observations writes:

$$\begin{aligned} L_t(\theta) = \sum_{i=1}^{t} \ell_i(\theta) &= \frac{1}{2}\theta^T(\sum_{i=1}^{t} H_i^T R_i^{-1} H_i)\theta - (\sum_{i=1}^{t} y_i R_i^{-1} H_i)\theta + \frac{1}{2}\sum_{i=1}^{t} y_i^T R_i^{-1} y_i \\ &= \frac{1}{2}\theta^T Q_t \theta - v_t^T\theta + C, \quad \text{where we let} \quad Q_t = \sum_{i=1}^{t} H_i^T R_i^{-1} H_i, \end{aligned} \tag{31}$$

where C is a constant with respect to θ. Assuming Q_t is invertible and the solution to the least mean squares problem is given by the normal equation $\theta_t^* = Q_t^{-1} v_t$, we can express θ_t^* as a function of θ_{t-1}^*, and we then recover the linear Kalman filter or equivalently the online Newton algorithm.

We prove now that in the simple case of linear regression the R-VGA updates of Theorem 1 are strictly equivalent to the linear Kalman filter and the online Newton algorithm. This is logical, as the posteriors are then Gaussian so that the KL divergence may be made equal to 0 at each step.

Theorem 2. *We suppose the observation model is Gaussian and linear $y_t = H_t\theta_t + \varepsilon_t$, with $\varepsilon_t \sim \mathcal{N}(0, R_t)$, and the prior is defined as $p_0 = q_0 = \mathcal{N}(\mu_0, P_0)$ and we define $Q_0 = P_0^{-1}$. Then, the R-VGA* (3)-(4)*:*

R-VGA

$$\mu_t = \mu_{t-1} - P_{t-1}\mathbf{E}_{q_t}[\nabla_\theta \ell_t(\theta)] \tag{32}$$

$$P_t^{-1} = P_{t-1}^{-1} + \mathbf{E}_{q_t}[\nabla_\theta^2 \ell_t(\theta)], \tag{33}$$

is algebraically equivalent to the linear Kalman filter for a stationary state θ defined as:

Kalman filter

$$\mu_t = \mu_{t-1} + K_t(y_t - H_t\mu_{t-1}) \tag{34}$$

$$K_t = P_{t-1}H_t^T(R_t + H_t P_{t-1} H_t^T)^{-1} \tag{35}$$

$$P_t = (\mathbb{I} - K_t H_t)P_{t-1}, \tag{36}$$

and to online Newton descent defined as:

Online Newton descent

$$\begin{aligned}\mu_t &= \mu_{t-1} - Q_t^{-1}\nabla\ell_t(\mu_{t-1})\\ Q_t &= Q_{t-1} + \nabla^2\ell_t(\mu_{t-1}),\end{aligned} \tag{37}$$

and to our

R-VGA (explicit scheme)

$$\begin{aligned}\mu_t &= \mu_{t-1} - P_t\mathbf{E}_{q_{t-1}}[\nabla_\theta\ell_t(\theta)],\\ P_t^{-1} &= P_{t-1}^{-1} + \mathbf{E}_{q_{t-1}}[\nabla^2_\theta\ell_t(\theta)].\end{aligned}$$

They all correspond to the optimal iterative algorithm for the least mean squares problem weighted by R_t. From a probabilistic viewpoint it means at each t the computed parameters satisfy $\mathcal{N}(\theta|\mu_t,P_t) \sim p(\theta \mid y_1,\dots,y_t)$.

The full proof is given in Appendix 7.1. The equivalence between the Kalman filter and online Newton is already well-known, see [3], but the equivalence with our variational approach is novel. Regarding this point, we note that the Kalman filter update (36) is known to write in information form $P_t^{-1} = P_{t-1}^{-1} + H_t R_t^{-1} H_t$ hence we immediately recover (33). Regarding (32), it is easily shown to rewrite in our context $\mu_t = \mu_{t-1} + P_{t-1}H_t^T R_t^{-1}(y_t - H_t\mu_t)$. Although the latter bears a resemblance to (34), the link is not straightforward and relies on the nontrivial but useful fact that one may rewrite the Kalman gain as $K_t = P_t H_t^T R_t^{-1}$.

The theorem shows how the issues inherent to the implicit scheme evaporate in the linear case, as we may arrive at expectations that involve the known distribution q_{t-1}. Moreover, it shows the algorithm is unbeatable in the linear Gaussian case: this is logical as it outputs a Gaussian that minimizes the KL divergence to a constant times $p(y_t|\theta)q_{t-1}(\theta)$, but the latter is Gaussian in the present case, so that the algorithm exactly recovers the partial posterior at each step.

3.2 Application to exponential families

When the likelihood stems from an exponential family, we may proceed along the same lines under linearization assumptions. This way, we may extend the results by [17] about the equivalence between the natural gradient and the extended Kalman filter (EKF) to a full equivalence with the proposed R-VGA for a generalized linear model (GLM). We consider in this section that the observations follow an exponential family distribution with a nonlinear model h, that is,

$$\begin{aligned}&p(y|\theta) = m(y)\exp(\eta(\theta)^T y - A(\eta(\theta)))\\ &\mathbf{E}(y|\theta) := \bar{y} = h(\theta) \text{ where } h \text{ is the model,}\\ &\eta = g(\bar{y}) = g(h(\theta)) \text{ where } g \text{ is the link function.}\end{aligned} \tag{38}$$

Assume now that at each step t of an online algorithm, the model h is approximated through a first-order Taylor expansion around the previous estimate μ_{t-1} :

$$h(\theta) \approx h(\mu_{t-1}) + \nabla_\theta^T h(\mu_{t-1})(\theta - \mu_{t-1}) = h(\mu_{t-1}) + H_t(\theta - \mu_{t-1}), \tag{39}$$

and the covariance is evaluated around the last mean $h(\mu_{t-1})$ and assumed independent of θ:

$$\mathrm{Cov}(y_t) = \mathbf{E}[(y_t - h(\mu_{t-1}))(y_t - h(\mu_{t-1}))^T] := R_t. \tag{40}$$

From the property of exponential families, the mean vector and the covariance matrix are deduced from the first and second derivative of the log-partition function $A(\eta)$, so we have the following relations:

$$\frac{\partial A(\eta_t)}{\partial \eta_t} = \bar{y}_t = h(\mu_{t-1}) + H_t(\theta - \mu_{t-1})$$
$$\text{and } \frac{\partial^2 A(\eta_t)}{\partial \eta_t^2} = \frac{\partial \bar{y}_t}{\partial \eta_t} = R_t.$$

The loss function under the latter linearizations writes:

$$\tilde{\ell}_t(\theta) = -\log p(y_t|\theta) = -\eta_t^T y_t + A(\eta_t) + C. \tag{41}$$

And its derivatives are approximated as:

$$\nabla_\theta \tilde{\ell}_t(\theta) = \frac{\partial \tilde{\ell}_t}{\partial \eta}\frac{\partial \eta}{\partial \bar{y}}\frac{\partial \bar{y}}{\partial \theta} = -H_t^T R_t^{-1}(y_t - h(\mu_{t-1}) - H_t(\theta - \mu_{t-1})) \tag{42}$$
$$\nabla_\theta^2 \tilde{\ell}_t(\theta) = H_t^T R_t^{-1} H_t. \tag{43}$$

We can then rewrite the update equation (4) as:

$$P_t^{-1} = P_{t-1}^{-1} + \mathbf{E}_{q_t}[H_t^T R_t^{-1} H_t] \tag{44}$$
$$= P_{t-1}^{-1} + H_t^T R_t^{-1} H_t, \tag{45}$$

and we find the same form that as in the linear case, and similar derivations may be applied to the update equation (3). Combined with the results from [17] we finally find:

Theorem 3. *For observations based on an exponential family* (38)*, under the approximations and assumptions* (39)-(40)*, that is, using the equalities* (42)-(43)*, one may prove that*

The linearized R-VGA
$$\mu_t = \mu_{t-1} - P_{t-1}\mathbf{E}_{q_t}[\nabla_\theta \tilde{\ell}_t(\theta)]$$
$$P_t^{-1} = P_{t-1}^{-1} + \mathbf{E}_{q_t}[\nabla_\theta^2 \tilde{\ell}_t(\theta)], \tag{46}$$

is algebraically equivalent to

The online Newton descent
given μ_0 and Q_0 supposed invertible:
$$\mu_t = \mu_{t-1} - Q_t^{-1}\nabla \tilde{\ell}_t(\mu_{t-1}) \tag{47}$$
$$Q_t = Q_{t-1} + \nabla^2 \tilde{\ell}_t(\mu_{t-1}), \tag{48}$$

and to the online natural gradient defined as:

The online natural gradient with learning rate $\frac{1}{t+1}$
given μ_0 and J_0 supposed invertible:
$$\mu_t = \mu_{t-1} - \frac{1}{t+1} J_t^{-1}\nabla_\theta \tilde{\ell}_t(\mu_{t-1})$$
$$J_t = \frac{t}{t+1}J_{t-1} + \frac{1}{t+1}\mathbf{E}_y[\nabla_\theta \tilde{\ell}_t(\mu_{t-1}) \times \nabla_\theta \tilde{\ell}_t(\mu_{t-1})], \tag{49}$$

as well as to the extended Kalman filter for a stationary state:

The extended Kalman filter
given μ_0 and P_0 supposed invertible:

$$\begin{aligned}
&\mu_t = \mu_{t-1} + K_t(y_t - h(\mu_{t-1})) \\
&H_t = \nabla_\theta^T h(\mu_{t-1}) \\
&K_t = P_{t-1}H_t^T(R_t + H_t P_{t-1} H_t^T)^{-1} \\
&P_t^{-1} = P_{t-1}^{-1} + H_t^T R_t^{-1} H_t
\end{aligned} \tag{50}$$

where the last line may alternatively be re-written $P_t = (\mathbb{I} - K_t H_t)P_{t-1}$.

The full proof has been moved to Appendix 7.2.

As for the linear case, it is shown in the proof in Appendix 7.2 that the implicit scheme is equivalent to the explicit scheme. However it is only true for the linearized version of the R-VGA. The equivalence between explicit and implicit scheme is lost for the exact R-VGA version.

Moreover the theorem shows that, for observations following an exponential family, all the mentioned algorithms including ours coincide under linearization assumptions. However, this is not true when considering the original nonlinear model. This is important, though, as experiments of Section 5 will prove that for the logistic loss the R-VGA outperforms the EKF, and equivalently the online natural gradient.

Remark 1. *The online natural gradient, in Equation* (49)*, uses an expectation under y whereas the exact R-VGA uses an expectation under θ. These expectations have nothing in common. For the natural gradient the expectation comes from the Fisher matrix definition and allows for the Gauss-Newton form for the Hessian. Indeed, the Gauss-Newton form is equivalent to the Hessian form only if we average under the observations y: $\mathbf{E}_y[\nabla_\theta \tilde{\ell}_t(\mu_{t-1}) \times \nabla_\theta \tilde{\ell}_t(\mu_{t-1})] = \mathbf{E}_y[\nabla^2_\theta \tilde{\ell}_t(\mu_{t-1})]$, otherwise the Hessian can be strongly biased as shown by [12]. For a GLM model (like the logistic regression problem) we can drop the expectation $\mathbf{E}_y$ without loosing precision if we introduce the covariance matrix R_t: $\mathbf{E}_y[\nabla^2_\theta \tilde{\ell}_t(\mu_{t-1})] = \nabla_\theta h(\mu_{t-1}) \times R_t^{-1} \times \nabla_\theta h(\mu_{t-1})$ as shown by [15] in the generalized Gauss-Newton framework. The detailed proof of this relation is recalled at the end of Appendix 7.2.*

4 Application to logistic regression

In this section, we apply the R-VGA to binary classification. We compare it to the extended Kalman filter and to the quadratic Kalman filter which is a variational variant using an upper bound on the logistic loss. In logistic regression, the observation are binary labels $y_t \in \{0,1\}$ associated to input variable $x_t \in \mathbf{R}^d$. The loss relies on the logistic function $\sigma(x) = \frac{1}{1+\exp(-x)}$ as:

$$\ell_t(\theta) = -\log p(y_t|\theta) = -y_t \log \sigma(x_t^T\theta) - (1-y_t)\log(1-\sigma(x_t^T\theta)).$$

4.1 The R-VGA for logistic regression

We need first to compute the expectations which appear in the R-VGA. Using the relation $\sigma' = (1-\sigma)\sigma$, the first and second derivatives of the logistic loss are given by:

$$\begin{aligned}
\nabla_\theta \ell_t(\theta) &= -(y_t - \sigma(x_t^T\theta))x_t, \\
\nabla^2_\theta \ell_t(\theta) &= x_t \sigma'(x_t^T\theta)x_t^T.
\end{aligned}$$

The Gaussian expectations of these derivatives read:

$$\mathbf{E}_{q_t}[\nabla_\theta \ell_t(\theta)] = -y_t x_t + \mathbf{E}_{q_t}[\sigma(x_t^T\theta)]x_t \tag{51}$$

$$\mathbf{E}_{q_t}[\nabla^2_\theta \ell_t(\theta)] = x_t \mathbf{E}_{q_t}[\sigma'(x_t^T\theta)]x_t^T. \tag{52}$$

To compute $\mathbf{E}_{q_t}[\sigma(x_t^T\theta)]$, we use the marginalized variable a along the vector x_t, i.e., $a = x_t^T\theta$, with distribution $\mathcal{N}(\mu_a, \sigma_a^2)$ where $\mu_a = x_t^T\mu_t$ and $\sigma_a^2 = x_t^T P_t x_t$, as proposed by [1]. Then we deduce:

$$\mathbf{E}_{q_t}[\sigma(x_t^T\theta)] = \int \sigma(x_t^T\theta) q_t(\theta) d\theta = \int_{-\infty}^{+\infty} \sigma(a)p(a)da.$$

For analytical tractability we approximate the logistic function with the inverse probit function as proposed by [1]:

$$\sigma(a) \approx \Phi(\lambda a) := \frac{1}{2}(1 + \mathrm{erf}(\frac{\lambda a}{\sqrt{2}}))$$
$$\text{with } \mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp(-u^2)du \text{ and } \lambda = \sqrt{\frac{\pi}{8}}.$$

Figure 1 shows a good match, indicating this choice is equivalent in practice to the logistic function.

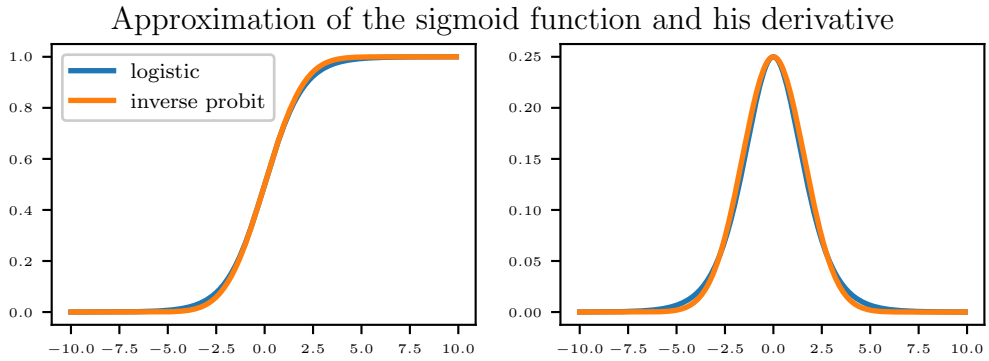

Figure 1: Comparison of the logistic function and the inverse probit function (left) ands their derivatives (right). Both match well, and shall be indifferently used in applications.

We obtain analytical expressions for the expectations:

$$\int_{-\infty}^{+\infty} \sigma(a)\mathcal{N}(a|\mu_a, \sigma_a^2)da \approx \int_{-\infty}^{+\infty}\int_{-\infty}^{\lambda a} \mathcal{N}(x|0,1)\mathcal{N}(a|\mu_a,\sigma_a^2)dxda$$
$$= \int_{-\infty}^{+\infty}\int_{-\infty}^{0} \mathcal{N}(x|-a,\lambda^{-2})\mathcal{N}(a|\mu_a,\sigma_a^2)dxda = \int_{-\infty}^{0}[\mathcal{N}(0,\lambda^{-2}) * \mathcal{N}(-\mu_a, \sigma_a^2)](x)dx$$
$$= \int_{-\infty}^{0} \mathcal{N}(x|-\mu_a, \sigma_a^2 + \lambda^{-2})dx = \Phi\left(\frac{\mu_a}{\sqrt{\sigma_a^2 + \lambda^{-2}}}\right) \approx \sigma\left(\frac{\mu_a}{\lambda\sqrt{\sigma_a^2 + \lambda^{-2}}}\right).$$

To compute $\mathbf{E}_{q_t}[\sigma'(x_t^T\theta)]$, we follow [6] where we let $\beta = \lambda^{-1} = \sqrt{8/\pi}$:

$$\int_{-\infty}^{+\infty} \sigma'(a)p(a)da \approx \int_{-\infty}^{+\infty} \mathcal{N}(a|0,\beta^2)p(a)da = \frac{1}{\beta\sigma_a 2\pi}\int_{-\infty}^{+\infty} \exp -\frac{1}{2}\left(-\frac{a^2}{\beta^2} - \frac{(a-\mu_a)^2}{\sigma_a^2}\right)da$$
$$= \frac{1}{\beta\sigma_a 2\pi}\exp\left(-\frac{1}{2}\frac{\mu_a^2}{\sigma_a^2+\beta^2}\right)\int_{-\infty}^{+\infty}\exp\left(-\frac{1}{2}\frac{\sigma_a^2+\beta^2}{\sigma_a^2\beta^2}(a-\frac{\mu_a\beta^2}{\sigma_a^2+\beta^2})^2\right)da$$
$$= \frac{1}{\beta\sigma_a 2\pi}\exp\left(-\frac{1}{2}\frac{\mu_a^2}{\sigma_a^2+\beta^2}\right)\sqrt{\frac{2\pi\sigma_a^2\beta^2}{\sigma_a^2+\beta^2}} = \frac{1}{\sqrt{2\pi(\sigma_a^2+\beta^2)}}\exp\left(-\frac{1}{2}\frac{\mu_a^2}{\sigma_a^2+\beta^2}\right)$$
$$= \mathcal{N}(0|\mu_a,\sigma_a^2+\beta^2) \approx \frac{\beta}{\sqrt{\sigma_a^2+\beta^2}}\sigma'\left(\frac{\mu_a\beta}{\sqrt{\sigma_a^2+\beta^2}}\right).$$

Coming back to (51)-(52), we find:

$$k_t = \frac{\beta}{\sqrt{x_t^T P_t x_t + \beta^2}}, \quad \mathbf{E}_{q_t}[\nabla_\theta \ell_i(\theta)] \approx -y_t x_t + \sigma(k_t x_t^T\mu_t)x_t, \quad \mathbf{E}_{q_t}[\nabla^2_\theta \ell_t(\theta)] \approx x_t k_t \sigma'(k_t x_t^T\mu_t)x_t^T. \tag{53}$$

As shown by Figure 1, the inverse probit approximation is a very good approximation. Gathering those results, the R-VGA now reads

$$k_t = \frac{\beta}{\sqrt{x_t^T P_t x_t + \beta^2}} \tag{54}$$
$$\mu_t = \mu_{t-1} + P_{t-1}x_t(y_t - \sigma(k_t x_t^T\mu_t)), \tag{55}$$
$$P_t^{-1} = P_{t-1}^{-1} + k_t\sigma'(k_t x_t^T\mu_t)x_t x_t^T. \tag{56}$$

We see we have obtained analytical expressions for the integrals involved, as in the linear case. However, the algorithm remains implicitly defined, and we will shortly show how to solve the problem. A prediction based on any obtained distribution $q \sim \mathcal{N}(\mu, P)$, for an input x_s may then be computed as:

$$\hat{y}_s = \mathbf{E}_q[\sigma(x_s^T\theta)] = \sigma\left(k_s x_s^T\mu\right) \tag{57}$$
$$\text{where } k_s = \frac{\beta}{\sqrt{x_s^T P x_s + \beta^2}}.$$

The variance of the prediction may also be computed as (following [6]):

$$\mathbf{Var}(\hat{y}_s) = \mathbf{E}_q[\sigma^2(x_s^T\theta)] - \mathbf{E}_q^2[\sigma(x_s^T\theta)] = \sigma(k_s x_s^T\mu)(1-\sigma(k_s x_s^T\mu))(1-k_s). \tag{58}$$

Further details about output prediction and uncertainty assessment of the R-VGA may be found in Appendix 7.3.

4.2 Solving the implicit scheme

Solving the implicit scheme (55)-(56) is amenable to a two-dimensional fixed point problem. Indeed, we see that as soon as the scalar quantities $x_t^T\mu_t$ and $x_t^T P_{t-1}x_t$ are known the scheme becomes explicit. To find those quantities, we consider the following change of variables:

unknown variables

$\alpha = x_t^T\mu_t$

$\nu = x_t^T P_t x_t$

parameters in the equations

$\alpha_0 = x_t^T\mu_{t-1}$

$\nu_0 = x_t^T P_{t-1}x_t.$

We also rewrite k_t in (54) as $k(\nu)$ to make the dependency clearly appear. By multiplying equation (55) by x_t^T on the left we find:

$$\alpha = -\nu_0 \sigma(\alpha k(\nu)) + \alpha_0 + \nu_0 y_t. \tag{59}$$

Applying the Woodbury formula to (56) we obtain:

$$P_t = P_{t-1} - P_{t-1} x_t \left(\frac{1}{k(\nu)\sigma'(k(\nu) x_t^T \mu_t)} + x_t^T P_{t-1} x_t \right)^{-1} x_t^T P_{t-1}. \tag{60}$$

Multiplying this equation by x_t^T on the left and x_t on the right we find:

$$\nu = \nu_0 - \nu_0^2 \left(\frac{1}{k(\nu)\sigma'(\alpha k(\nu))} + \nu_0 \right)^{-1} = \frac{\nu_0}{1 + \nu_0 k(\nu)\sigma'(\alpha k(\nu))}. \tag{61}$$

We end up with to a two-dimensional implicit scheme (59)-(61). To solve it we need to find the roots of $F_{\alpha_0,\nu_0,y}$ defined as:

$$F_{\alpha_0,\nu_0,y} : \mathbb{R}^2 \to \mathbb{R}^2 \tag{62}$$

$$(\alpha, \nu) \to (f(\alpha,\nu), g(\alpha,\nu)), \tag{63}$$

with:

$$f(\alpha,\nu) = \alpha + \nu_0 \sigma(\alpha k(\nu)) - \alpha_0 - \nu_0 y \tag{64}$$

$$g(\alpha,\nu) = \nu - \frac{\nu_0}{1 + \nu_0 k(\nu)\sigma'(\alpha k(\nu))}, \tag{65}$$

on the domain:

$$\alpha_0 + \nu_0(y-1) \le \alpha \le \alpha_0 + \nu_0 y \tag{66}$$

$$0 \le \nu_0 (1 - \frac{\nu_0}{4+\nu_0}) \le \nu \le \nu_0. \tag{67}$$

If need be, the domain may be further reduced, as shown in Appendix 7.4, where the shape of the functions f and g are also displayed, in Figure 12. To find the roots of $F_{\alpha_0,\nu_0,y}$, we have opted for the hybrid-Powel method [18] which converges quite fast for relatively moderate values of α_0 and ν_0 (typically below 10^{-4}). To enforce the fact that ν be positive, we change variables in the optimization scheme letting $\exp(\tilde{\nu}) = \nu$.

We obtain finally the following implicit updates:

R-VGA (implicit) for logistic regression

$$\alpha_0 = x_t^T \mu_{t-1}, \nu_0 = x_t^T P_{t-1} x_t, \tag{68}$$

$$\alpha, \nu \leftarrow \text{roots of } F_{\alpha_0,\nu_0,y_t} \text{ defined in } (62) - (65), \tag{69}$$

$$k_t = \frac{\beta}{\sqrt{\nu + \beta^2}}, \tag{70}$$

$$\mu_t = \mu_{t-1} + P_{t-1} x_t (y_t - \sigma(k_t \alpha)), \tag{71}$$

$$P_t = P_{t-1} - P_{t-1} x_t x_t^T P_{t-1} / (\frac{1}{k_t \sigma'(k_t \alpha)} + x_t^T P_{t-1} x_t). \tag{72}$$

4.3 Numerical complexity

To avoid inverting P_t as in (56), we have used Woodbury's formula in (72) to update P_t directly, as in the Kalman filter, along the lines of (60). At each step we see $O(d^2)$ operations are required, as soon as the quantities $x_t^T \mu_t$ and $x_t^T P_{t-1} x_t$ have been found, which amounts to computing a few iterations of a two-dimensional optimizer. This makes an overall computation of cost of order $O(d^2 N)$.

Note that, if one wants to enforce the positivity of matrix P_t, it is possible without increasing the overall computational cost to resort to a square-root implementation akin to the Kalman filter's, see [4].

4.4 Alternative algorithms

The proposed R-VGA algorithm is optimal at each step given the previous Gaussian approximation, that is, it minimizes the divergence between $q_t(\theta)$ and $p(y_t|\theta)q_{t-1}(\theta)$ assuming the inverse probit approximation of the logistic function. It may be compared with other variants, that is, our explicit scheme approximation, the well-known extended Kalman filter applied to logistic regression as presented by [17] and an online version of the variational approach to logistic regression of [10] we introduced hereafter.

4.4.1 The explicit scheme approximation

Our approximated explicit-scheme based R-VGA (27)-(28) may be readily applied using the analytical expressions (53) for the integrals replacing μ_t, P_t with μ_{t-1}, P_{t-1}:

R-VGA (explicit) for logistic regression

$$k_t = \frac{\beta}{\sqrt{x_t^T P_{t-1} x_t + \beta^2}} \tag{73}$$

$$\mu_t = \mu_{t-1} + P_t x_t (y_t - \sigma(k_t x_t^T \mu_{t-1})), \tag{74}$$

$$P_t = P_{t-1} - P_{t-1} x_t x_t^T P_{t-1} / (\frac{1}{k_t \sigma'(k_t x_t^T \mu_{t-1})} + x_t^T P_{t-1} x_t). \tag{75}$$

Albeit optimal in the context of linear regression, this scheme is no longer optimal in the non-linear case. However, We find in the sequel its performance is often close to the implicit scheme's, and better than the EKF's for the logistic regression problem. Comparisons are displayed in our experiments indeed.

4.4.2 The extended Kalman filter for logistic regression

The extended Kalman filter (EKF) is based on the linearization of the likelihood, and is equivalent to the natural gradient for the logistic regression, see [17]. We have shown in Theorem 3 that the extended Kalman filter is also equivalent to the linearized version of the R-VGA. Comparing both will indicate how the exact version behaves compared to the linearized one. To build the extended Kalman filter we need to compute the covariance of the observations R_t and the observation matrix H_t which is the Jacobian of the averaged observed value at the previous estimate. Following [17], those quantities write:

$$\begin{aligned}
&R_t = \mathrm{Cov}(y_t) = \sigma'(x_t^T \mu_{t-1}) = \sigma(x_t^T \mu_{t-1})(1 - \sigma(x_t^T \mu_{t-1})) \\
&\bar{y}_t = \sigma(x_t^T \theta) \approx \sigma(\mu_{t-1}) + H_t(\theta - \mu_{t-1}) \\
&\text{where } H_t = \frac{\partial \sigma(x_t^T \theta)}{\partial \theta}(\mu_{t-1}) = \sigma'(x_t^T \mu_{t-1}) x_t^T = R_t x_t^T,
\end{aligned}$$

leading to the following equations for the EKF:

$$
\begin{aligned}
R_t &= \sigma(x_t^T \mu_{t-1})(1-\sigma(x_t^T \mu_{t-1})) \\
H_t &= R_t x_t^T \\
K_t &= P_{t-1} H_t^T (R_t + H_t P_{t-1} H_t^T)^{-1} \\
\mu_t &= \mu_{t-1} + K_t(y_t - \sigma(x_t^T \mu_{t-1})) \\
P_t &= (\mathbb{I} - K_t H_t) P_{t-1}.
\end{aligned}
$$

Using the fact that $K_t = P_t H_t^T R_t^{-1}$ and $P_t^{-1} = P_{t-1}^{-1} + H_t^T R_t^{-1} H_t$ allows rewriting the equations in compact form as:

The extended Kalman filter (EKF) for logistic regression (76)

$$
\begin{aligned}
\mu_t &= \mu_{t-1} + P_t x_t (y_t - \sigma(x_t^T \mu_{t-1})), \\
P_t &= P_{t-1} - P_{t-1} x_t x_t^T P_{t-1} / (\frac{1}{\sigma'(x_t^T \mu_{t-1})} + x_t^T P_{t-1} x_t).
\end{aligned}
$$

The obtained formulas bear a strong resemblance to the R-VGA, but important differences shall be noted, though. Indeed, in the rightmost terms P_t and P_{t-1} are swapped and so are μ_t and μ_{t-1}. Besides, the factor $0 < k_t \leq 1$ which appears in the R-VGA is absent in the EKF. Those differences are not wholly surprising, as the extended Kalman filter linearizes the sigmoid function around the last estimate μ_{t-1} to compute the information matrix P_t^{-1}, whereas R-VGA truly minimizes the KL divergence. To this respect, the EKF may be viewed as an online version of the Laplace approximation.

4.4.3 The quadratic Kalman filter for logistic regression

The logistic loss has a sharp form and it is known that Laplace approximation can make the Hessian vanish. As a remedy [10] have proposed a quadratic upper bound for the logistic loss, easier to minimize. Following their approach, we construct first a lower bound for the sigmoid using the Legendre transform, introducing local tangents of slope η at points ξ (see [10] for more details):

$$
\sigma(\theta) \geq \sigma(\xi) \exp(\frac{\theta - \xi}{2} + \eta(\theta^2 - \xi^2)) \quad \forall \xi \in \mathbb{R} \tag{77}
$$

$$
\text{with } \eta = -\frac{1}{2\xi}(\sigma(\xi) - \frac{1}{2}). \tag{78}
$$

Giving this lower bound on the sigmoid, the upper bound for $-\log p(y|\theta)$ has a quadratic form in θ easier to minimize than the initial logistic loss (here x is the input as before):

$$
-\log p(y|\theta) \leq -x^T \theta y - \log \sigma(\xi) + \frac{x^T \theta + \xi}{2} - \eta((x^T \theta)^2 - \xi^2) = Q(\theta, \xi).
$$

This upper bound has also been used in the context of Gaussian processes using approximation with Polya-Gamma function (see [23], Appendix A.5). The lower and upper bound shapes are displayed in Figure 2 for dimensions one and two. In the present paper we propose to apply the R-VGA (3)-(4) to $Q(\theta, \xi)$, which yields:

$$
\mu_t = \mu_{t-1} - P_{t-1} \mathbf{E}_{q_t}[\nabla_\theta Q(\theta, \xi)] = \mu_{t-1} - P_{t-1}(-2\eta \mu_t x_t x_t^T + (\frac{1}{2} - y_t) x_t) \tag{79}
$$

$$
P_t^{-1} = P_{t-1}^{-1} + \mathbf{E}_{q_t}[\nabla_\theta^2 Q(\theta, \xi)] = P_{t-1}^{-1} - 2\eta x_t x_t^T. \tag{80}
$$

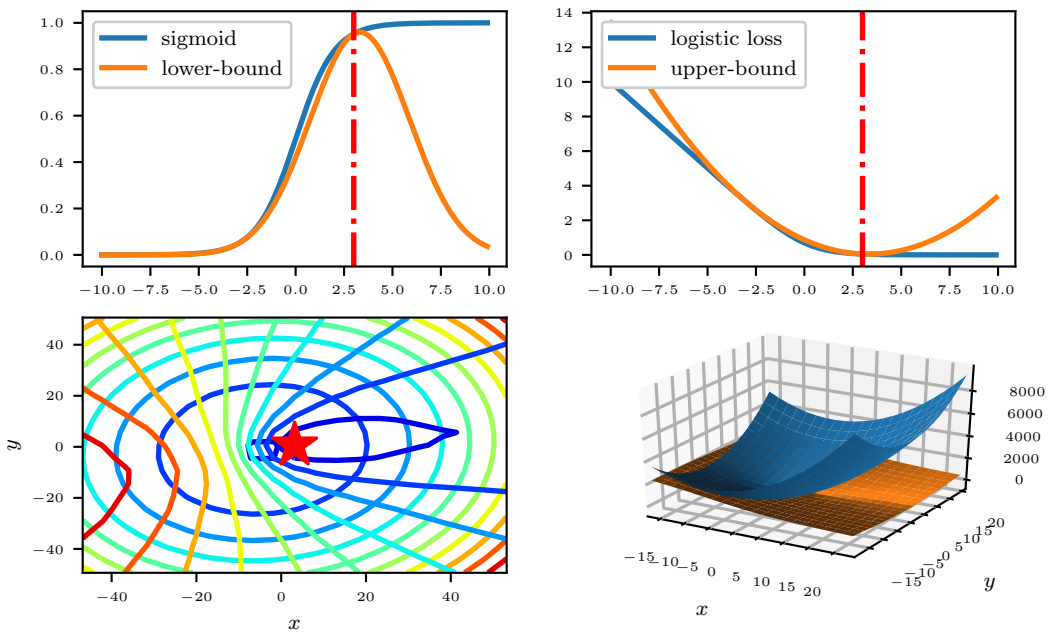

Figure 2: Quadratic lower bound on the logistic function (upper left) and upper bound on logistic loss (upper right). Upper bound for a two-dimensional problem (lower line).

The simplistic form of $Q(\theta, \xi)$ allows not only to obtain analytical expressions for the integrals but also to make the update explicit. Indeed, multiplying the equality (79) by P_{t-1}^{-1} on both sides, we find:

$$\mu_t = (P_{t-1}^{-1} - 2\eta x_t x_t^T)^{-1}(P_{t-1}^{-1}\mu_{t-1} - (\frac{1}{2} - y_t)x_t) = P_t(P_{t-1}^{-1}\mu_{t-1} + (y_t - \frac{1}{2})x_t), \tag{81}$$

which is the same form as the one found by [10]. This expression can be further rearranged to make a "Kalman gain" appear:

$$\mu_t = P_t((P_t^{-1} + 2\eta x_t x_t^T)\mu_{t-1} + (y_t - \frac{1}{2})x_t) \tag{82}$$

$$= \mu_{t-1} + P_t x_t(-2\eta)(\frac{y_t - \frac{1}{2}}{-2\eta} - x_t^T\mu_{t-1}) := \mu_{t-1} + K_t(\frac{y_t - \frac{1}{2}}{-2\eta} - x_t^T\mu_{t-1}). \tag{83}$$

We call the explicit algorithm we have just obtained the quadratic Kalman filter for logistic loss. Using Woodbury formulas and the Kalman filter formalism, we obtain the following update equations:

The quadratic Kalman filter (QKF) for logistic regression

$$
\begin{aligned}
&H_t = x_t^T \\
&\xi_t^2 = H_t(P_{t-1} + \mu_{t-1}\mu_{t-1}^T)H_t^T \qquad (84)\\
&R_t^{-1} = -2\eta = \frac{1}{\xi_t}(\frac{1}{1+e^{-\xi_t}} - \frac{1}{2}) \\
&K_t = P_{t-1}H_t^T(R_t + H_t P_{t-1} H_t^T)^{-1} \\
&\mu_t = \mu_{t-1} + K_t(R_t(y_t - \frac{1}{2}) - H_t\mu_{t-1}) \\
&P_t = (\mathbb{I} - K_t H_t)P_{t-1}, \qquad (85)
\end{aligned}
$$

where the update equation for ξ used in (84) is derived from [10] and can also be found by differentiating the KL divergence with respect to ξ.

5 Numerical experiments for logistic regression

We consider N synthetic inputs x_i equally distributed along two Gaussians with identical covariance matrix such that the distribution of the outputs $p(y_i|x_i)$ writes as a sigmoid function. Moreover we suppose the mean vectors are symmetric $\mu_1 = -\mu_2$, so that the linear separator passes through the origin and the constant term is null. The mean μ_1 and μ_2 of the Gaussian are separated by a distance $s = ||\mu_1 - \mu_2|| = ||2\mu_1||$ which is a free parameter. The algorithms are initialized with a prior distribution $q_0 = \mathcal{N}(\mu_0, \sigma_0^2\mathbf{I})$. To make the problem more challenging, we consider ill-conditioned covariance matrices of the form:

$$C = M^T \text{Diag}(1, 1/2^c, \ldots, 1/d^c)M, \qquad (86)$$

where M is a random unitary orthogonal matrix to ensure the covariance is not aligned with the axes and c is a free parameter. If $c = 0$ the covariance is "isotropic", ortherwise it is "ill-conditionned". Moreover, we normalize the random inputs vectors for each classes i such that $x_i \leftarrow \mu_i + (x_i - \mu_i)/||std(x_i)||$ assuming the data can be pre-processed. This allows for a better management of the separability s of the dataset since the norm of the inputs scale as $\sqrt{d}$ (in the "isotropic" case).

5.1 Comparison metrics computation

To assess the performance of the algorithms, we propose to compute the KL divergence between the posterior and its approximations over time. To do so we note the posterior may be written $p(\theta|Y_N) = \frac{\tilde{p}(\theta|Y_N)}{Z_p}$ where Z_p is the partition function and $\tilde{p}(\theta|Y_N) = \Pi_1^N p(y_t|\theta)p_0(\theta)$ is the unnormalized posterior. The divergence to the posterior of any Gaussian density q with covariance Q may be re-written:

$$
\begin{aligned}
KL(q(\theta)||p(\theta|Y_N)) &= \int q(\theta)\log\frac{q(\theta)}{p(\theta|Y_N)} = \int q(\theta)\log\frac{q(\theta)}{\tilde{p}(\theta|Y_N)} + \log Z_p \qquad (87)\\
&= -\int q(\theta)\log\tilde{p}(\theta|Y_N) - \frac{1}{2}\log|Q| - \frac{d}{2}(1+\log(2\pi)) + \log Z_p. \qquad (88)
\end{aligned}
$$

To plot the evolution of the KL, we propose to draw M independent samples $\theta_k \sim q$, leading to the Monte-Carlo approximation:

$$KL \approx -\frac{1}{M}\sum_{k=1}^{M}\log\tilde{p}(\theta_k|Y_N) - \frac{1}{2}\log|Q| - \frac{d}{2}(1+\log(2\pi)) + \log Z_p. \qquad (89)$$

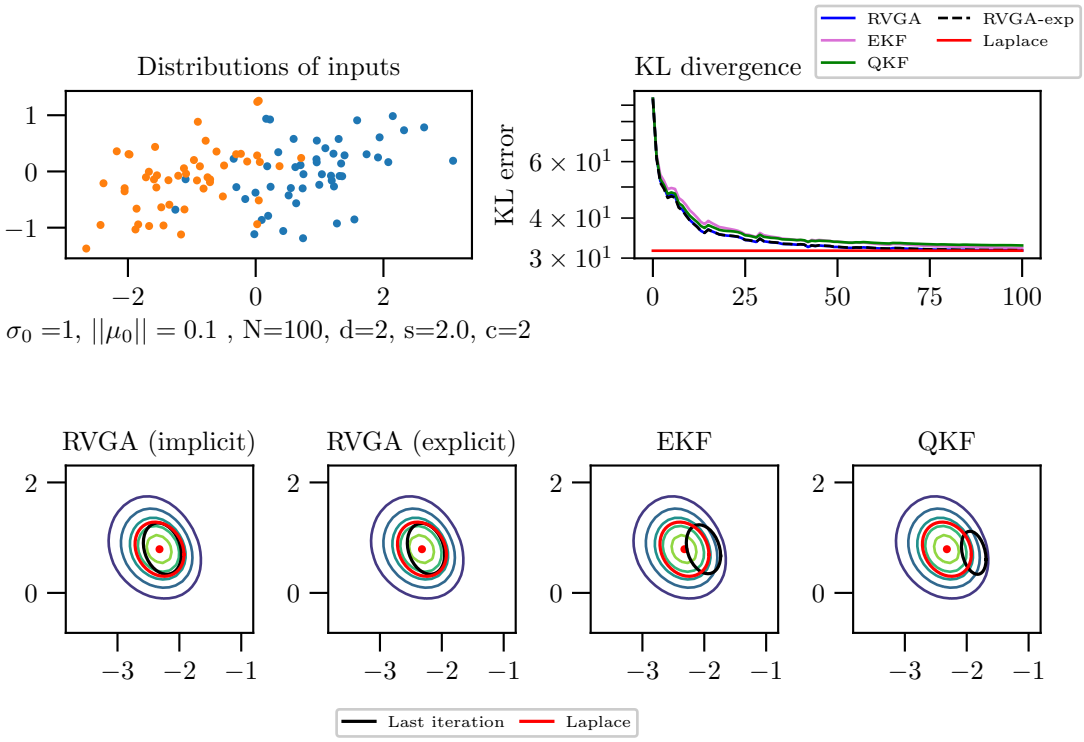

Figure 3: Case with $\sigma_0 = 1$ and $||\mu_0|| = 0.1$. Upper row: 2D synthetic dataset (left) and KL divergence. Lower row: Confidence ellipsoids for the R-VGA (left) vs EKF (middle) and QKF (right) at final time. The Batch Laplace covariance ellipsoid is shown in red and the contour lines of the true posterior are also displayed.

The normalization factor Z_p can take very low values in high dimension and may prove difficult to estimate. As we aim at comparing the algorithms, we may arbitrarily set it to 1 (hence the plotted KL can take higher values than the true KL). As it affects all KL evaluations in the same way, letting $Z_p = 1$ allows for relative comparisons.

5.2 Two-dimensional results

We begin with a two-dimensional parameter θ, that is, $d = 2$ and a sample size N of a few hundreds. This allows for plotting the confidence ellipsoids. In dimension two we use a grid to compute the true posterior and consider the batch Laplace approximation (see, e.g., [5]) as a baseline, even if it can be biased for asymmetric sharp posteriors.

The form of the posterior is sharp when the data are separable, that is when the parameter s reflecting separability is high. The posterior is sharper also when the prior has low confidence or equivalently when the problem is not regularized, that is σ_0 is large. In this case the maximum likelihood coincides with the maximum of the posterior. On the contrary, when σ_0 is very low, it dominates the observations, and the problem becomes approximately linear. The parameter μ_0 often represents our initial guess, and might be set to a wrong value by the user. The more disaligned with the separating hyperplane the more challenging for the algorithms. The extended Kalman filter proves very sensitive to this initial guess, whereas the R-VGA is much less sensitive to large μ_0 or σ_0.

We compare the R-VGA with the EKF and the variational approach based on a quadratic upper bound (QKF) detailed in the previous section. The ellipsoids of the covariance matrix at final time and the KL

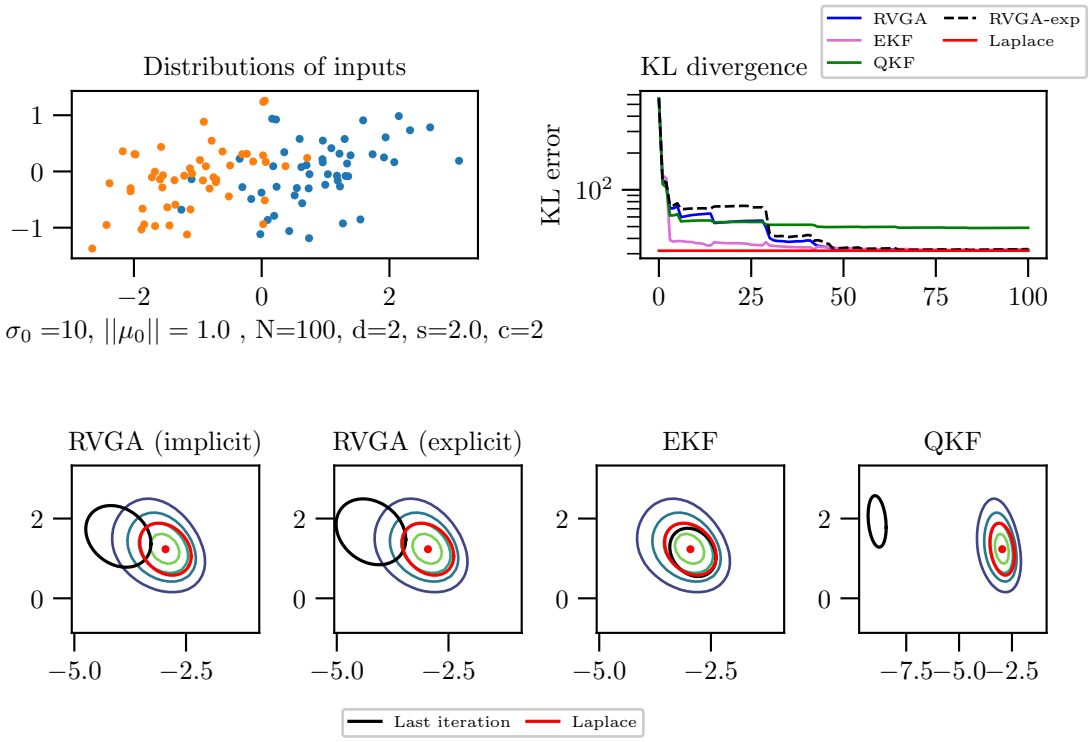

Figure 4: Same as Figure 3 in the case where $\sigma_0 = 10$ and $||\mu_0|| = 1$.

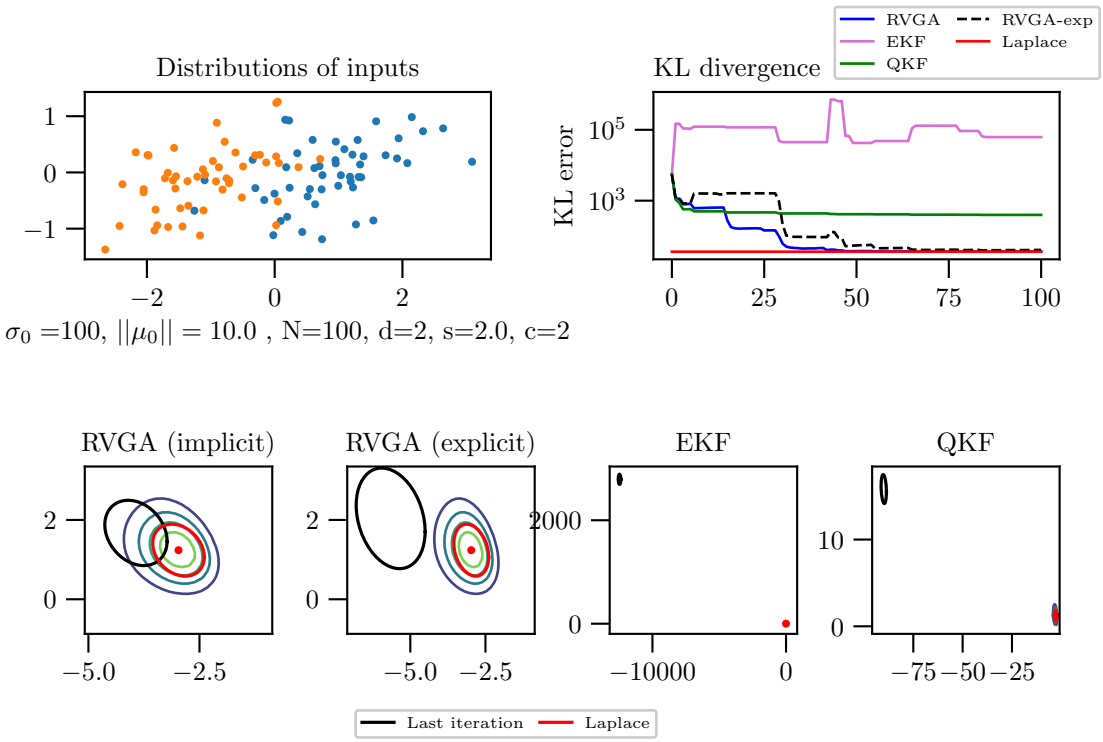

Figure 5: Same as Figure 3 in the case where $\sigma_0 = 100$ and $||\mu_0|| = 10$.

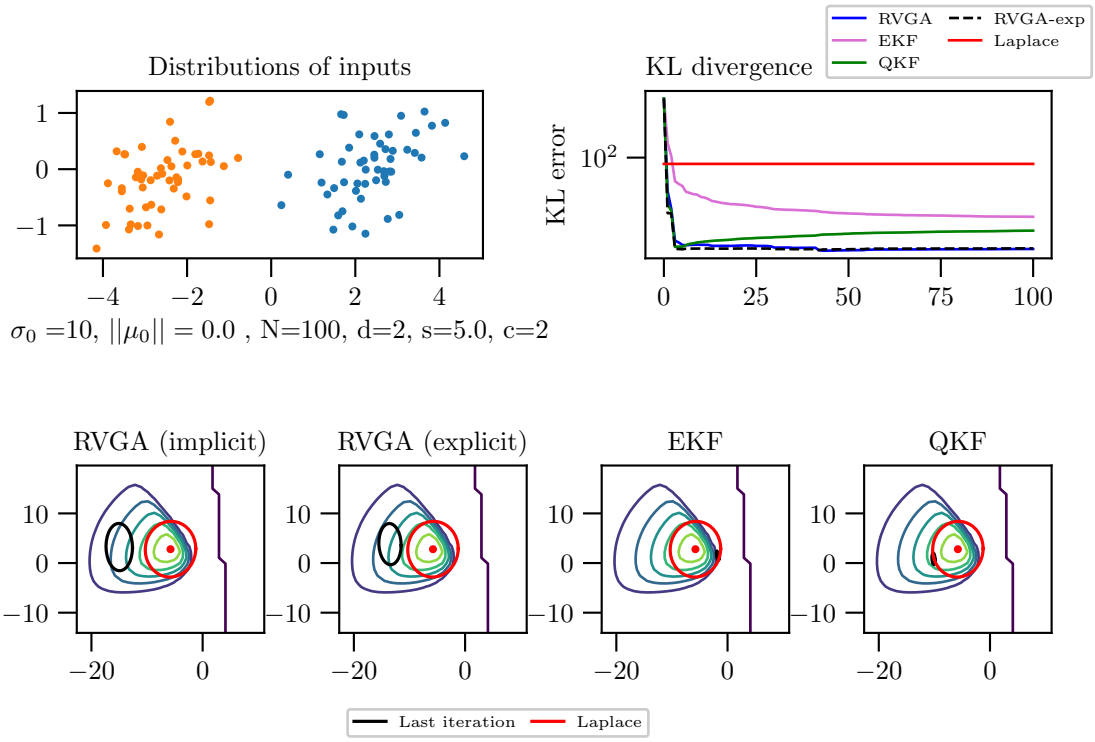

Figure 6: Same as Figure 3 in the harder case where the inputs distribution is ill-conditioned with μ_0 set to 0. The plots illustrate that the Gaussian Laplace approximation may be beaten in terms of KL divergence when the true posterior is asymmetric.

divergence are shown for different values of the prior parameters μ_0 and σ_0 (Figures 3 to 5) where we have supposed here μ_0 to be proportional to the unitary vector of $\mathbb{R}^d$. R-VGA appears more robust than EKF or QKF to initial guess. A harder case where the data are ill-conditioned is shown in Figure 6. More examples are shown in Appendix 7.5. We see that (exact) R-VGA, which is based on successive optimal Gaussian approximations, outperforms the EKF and QKF. The R-VGA explicit scheme yields results that are close to the R-VGA implicit scheme's but tends to degrade for large σ_0. However, a caveat that shall be borne in mind is that all the online recursive algorithms treat data one by one and then discard the data. This makes their use advantageous for large sample size, but inherently approximate.

5.3 Higher dimension results

In higher dimension we study the sensitivity to the dimension d of the parameter for a sharp prior encoded in σ_0 (Figure 7), and a flat prior (Figure 8). We see the R-VGA outperforms its Kalman-based counterparts. When the prior is weighty the discrepancies between algorithms look more moderate than when σ_0 is large, in which case differences are stark. We assess also the sensitivity to the separability of the dataset s (Figures 9 and 10). This parameter drives the sharpness of the posterior. We see that R-VGA outperforms the other Kalman variants and is even susceptible to beat the batch Laplace approximation because it avoids the region of near null probabilities whereas the Laplace method does not.

Sources: The sources of the code are available on Github on the following repository: `https://github.com/marc-h-lambert/Kalman4Classification`.

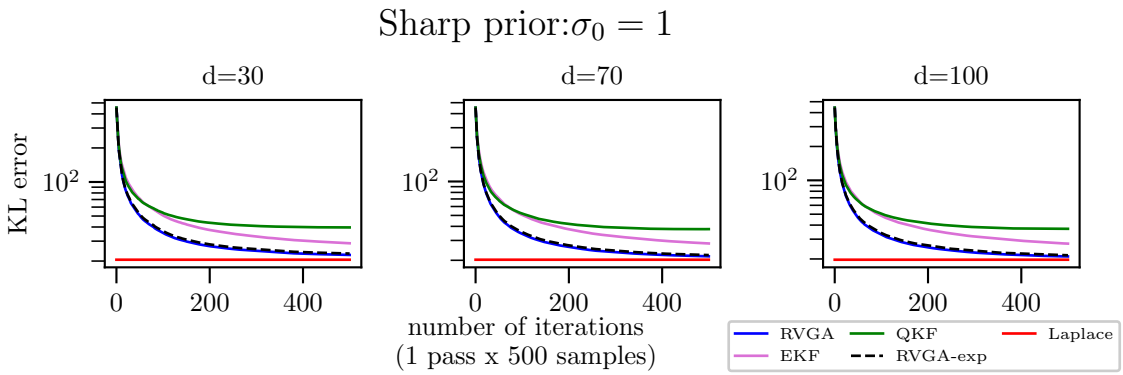

Figure 7: $\sigma_0 = 1$ and $\mu_0 = 0$. Evolution of the convergence for $d = 30$, 70 and 100 for $N = 500$ with an "isotropic" covariance.

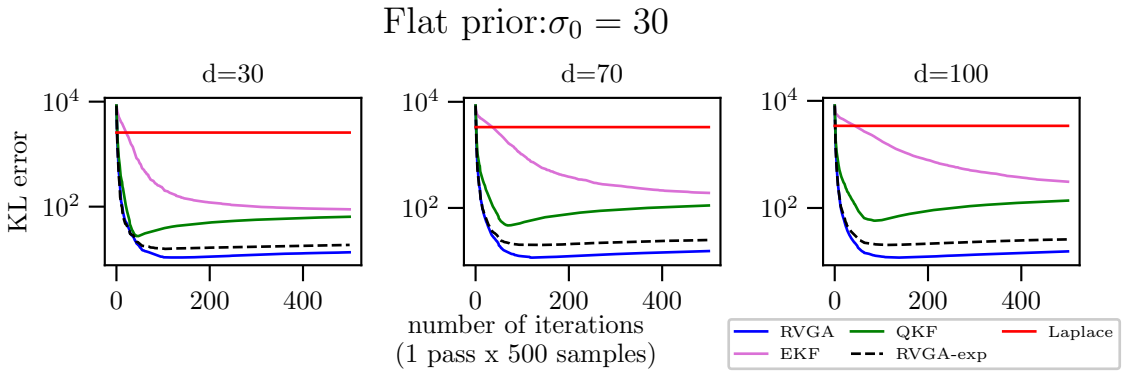

Figure 8: $\sigma_0 = 30$ and $\mu_0 = 0$. Evolution of the convergence for $d = 30$, 70 and 100 for $N = 500$ and an "isotropic" covariance.

6 Conclusion

We have shown how a recursive version of the variational Gaussian approximation leads to a new family of implicit optimization algorithms which generalize the extended Kalman filter and the natural gradient in the exponential family case. This recursive version allows for sequential processing of the data which is suited to large sample size as is characteristic of big data. Moreover, owing to the nature of the considered problem, that is, infer a posterior distribution, only one pass through the data is required. The algorithm performs a sequence of optimal Gaussian approximations, which comes at the price of an implicit update scheme. For linear regression R-VGA is optimal and shown to coincide with the Kalman filter. For the logistic regression problem, a computable form could be derived. In this case, the algorithm proves to scale relatively with the parameter dimension d, with an overall computation cost $O(d^2N)$. While having similar computational cost, R-VGA beats the extended Kalman filter or equivalently the online natural gradient, see [17], as well as the variational approach with quadratic upper bound we have proposed, inspired from [10]. When the posterior has a tall spike, though, we have observed the method fails to recover the true posterior, which is to be expected as it is based on a series of Gaussian approximations. However, this method does not require any tuning, and could be used to initialize more precise and computationally demanding MCMC methods to speed up their convergence. In the future, we would like to address issues related to high dimensional problems, and hence devise a low-rank version of R-VGA that scales linearly with d.

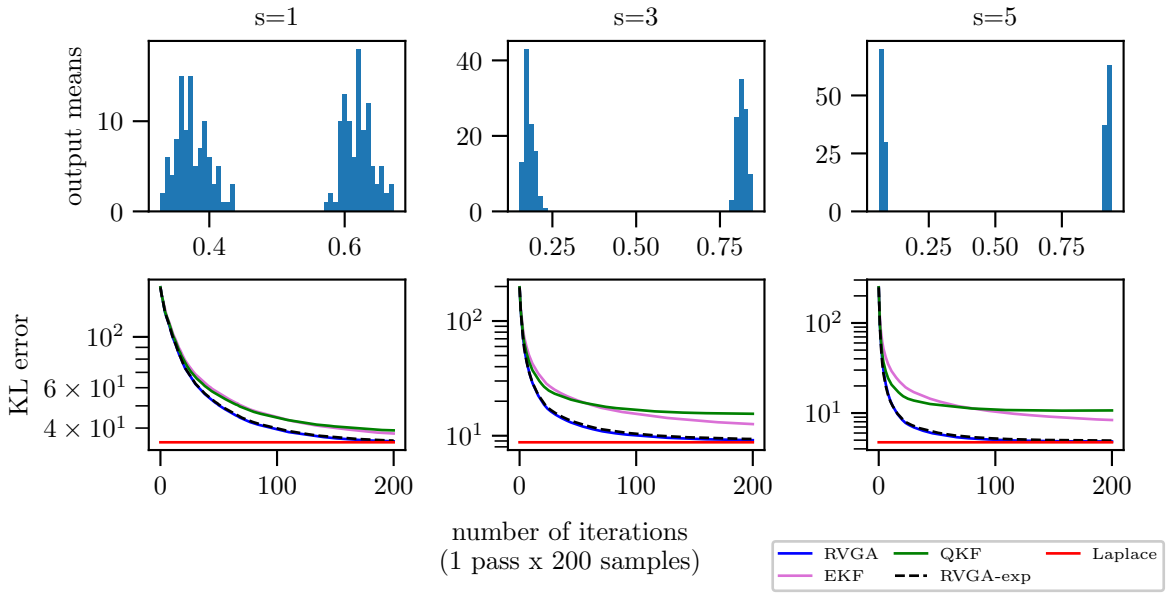

Figure 9: $\sigma_0 = 10$ and $\mu_0 = 0$. Evolution of the convergence with the separability of the inputs means s for $s = 1$,3 and 5. We have generated $N = 200$ inputs in dimension $d = 100$ and an "isotropic" covariance.

Acknowledgements

This work was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001(PRAIRIE 3IA Institute). We also acknowledge support from the French Defence procurement agency (DGA) and from the European Research Council (grant SEQUOIA 724063). The authors would like to thank Eric Moulines and Jean-Pierre Nouaille as well as Hadi Daneshmand for fruitful discussions related to this work.

References

[1] D. Barber and Christopher Bishop. Ensemble learning in Bayesian neural networks. In *Generalization in Neural Networks and Machine Learning*, pages 215–237, 1998.

[2] Timothy D. Barfoot, James R. Forbes, and David Yoon. Exactly sparse gaussian variational inference with application to derivative-free batch nonlinear state estimation. *arXiv preprint arXiv:1911.08333*, 2019.

[3] Dimitri P. Bertsekas. Incremental least squares methods and the extended kalman filter. *SIAM Journal on Optimization*, 6(3):807–822, 1996.

[4] G. J. Bierman. Measurement updating using the U-D factorization. In *Conference on Decision and Control including the Symposium on Adaptive Processes*, pages 337–346, 1975.

[5] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

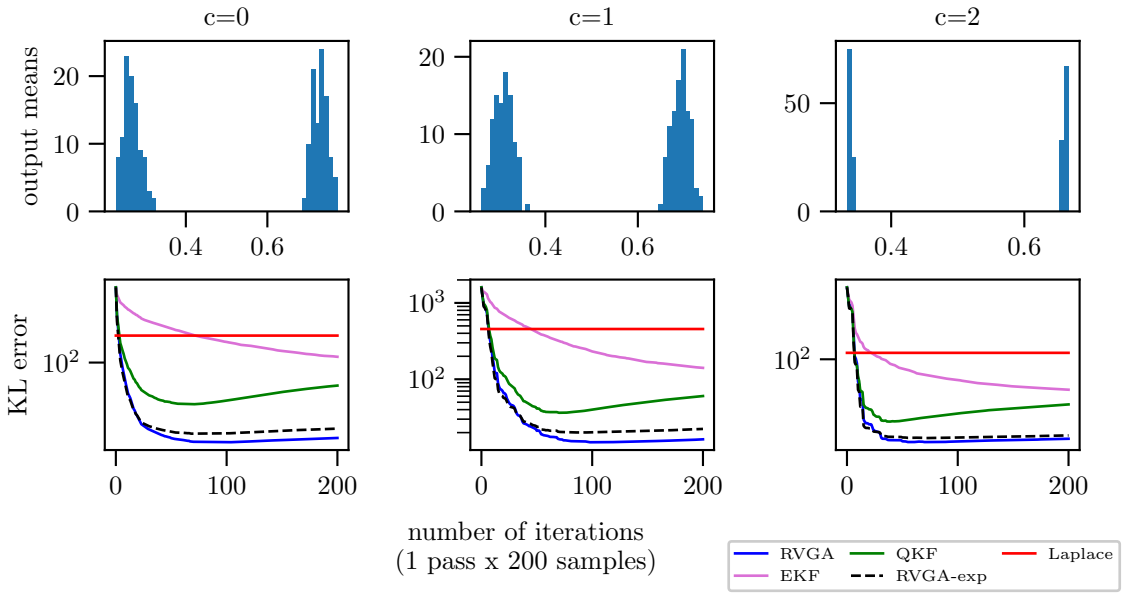

Figure 10: $\sigma_0 = 10$ and $\mu_0 = 0$. More challenging case of an ill-conditioned covariance matrix for the inputs. The separation of the data depends here to the geometry of the problem. We have generated $N = 200$ inputs in dimension $d = 100$.

[6] Jean Daunizeau. Semi-analytical approximations to statistical moments of sigmoid and softmax mappings of normal variables. *arXiv preprint arXiv:1703.00091*, 2017.

[7] Mohammad Emtiyaz Khan, Zuozhu Liu, Voot Tangkaratt, and Yarin Gal. Vprop: Variational inference using rmsprop. *arXiv*, pages arXiv–1712, 2017.

[8] Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 1, 2016.

[9] Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69:169–192, 2007.

[10] Tommi Jaakkola and Michael Jordan. A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, volume 82, 1997.

[11] Rémi Jézéquel, Pierre Gaillard, and Alessandro Rudi. Efficient improper learning for online logistic regression. *arXiv preprint arXiv:2003.08109*, 2020.

[12] Frederik Kunstner, Philipp Hennig, and Lukas Balles. Limitations of the empirical fisher approximation for natural gradient descent. In *Advances in Neural Information Processing Systems*, pages 4156–4167, 2019.

[13] Wu Lin, Mohammad Emtiyaz Khan, and Mark Schmidt. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. *arXiv preprint arXiv:1906.02914*, 2019.

[14] Wu Lin, Mohammad Emtiyaz Khan, and Mark Schmidt. Stein's lemma for the reparameterization trick with exponential family mixtures. *arXiv preprint arXiv:1910.13398*, 2019.

[15] James Martens. New insights and perspectives on the natural gradient method. *arXiv preprint arXiv:1412.1193*, 2014.

[16] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17:527–566, 2017.

[17] Yann Ollivier et al. Online natural gradient as a kalman filter. *Electronic Journal of Statistics*, 12:2930–2961, 2018.

[18] M. J. D. Powell. On nonlinear optimization since 1959. In *The Birth of Numerical Analysis*, pages 141–160. 2010.

[19] Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial intelligence and statistics*, pages 814–822. PMLR, 2014.

[20] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[21] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

[22] Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In *International conference on machine learning*, pages 1971–1979, 2014.

[23] Florian Wenzel, Théo Galy-Fajou, Christan Donner, Marius Kloft, and Manfred Opper. Efficient gaussian process classification using pòlya-gamma data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5417–5424, 2019.

7 Appendix

7.1 Proof of Theorem 2

Proof. In the case where p is a multivariate Gaussian distribution $p(y_t|\theta) \sim \mathcal{N}(y_t|H_t\theta, R_t)$, with H_t is the observation matrix, we have the relation:

$$\begin{aligned}
\log p(y_t|\theta) &= -\frac{1}{2}(y_t - H_t\theta)^T R_t^{-1}(y_t - H_t\theta) + C = -\ell_t(\theta) \\
\nabla_{\mu_t}\mathbf{E}_{q_t}[\log p(y_t|\theta)] &= H_t^T R_t^{-1}(y_t - H_t\mu_t) = -\nabla\ell_t(\theta) \\
-2\nabla_{P_t}\mathbf{E}_{q_t}[\log p(y_t|\theta)] &= H_t^T R_t^{-1} H_t = \nabla^2\ell_t(\theta),
\end{aligned}$$

this last relation gives directly the second R-VGA update equation, rewriting (2) as:

$$P_t^{-1} = P_{t-1}^{-1} + H_t^T R_t^{-1} H_t, \tag{90}$$

which is the information update equation. We then rewrite the first R-VGA update equation (1) as:

$$\begin{aligned}
&\mu_t = \mu_{t-1} + P_{t-1} H_t^T R_t^{-1} (y_t - H_t \mu_t) && (91)\\
&\iff \mu_t (\mathbf{I}_d + P_{t-1} H_t^T R_t^{-1} H_t) = \mu_{t-1} + P_{t-1} H_t^T R_t^{-1} y_t \\
&\iff \mu_t = (P_{t-1}^{-1} + H_t^T R_t^{-1} H_t)^{-1} P_{t-1}^{-1} (\mu_{t-1} + P_{t-1} H_t^T R_t^{-1} y_t) \\
&\iff \mu_t = \mu_{t-1} + (P_{t-1}^{-1} + H_t^T R_t^{-1} H_t)^{-1} (-H_t^T R_t^{-1} H_t \mu_{t-1} + H_t^T R_t^{-1} y_t) \\
&\iff \mu_t = \mu_{t-1} + P_t H_t^T R_t^{-1} (y_t - H_t \mu_{t-1}). && (92)
\end{aligned}$$

If we note the Kalman gain $K_t = P_t H_t^T R_t^{-1}$, we find the Kalman update equations for the state (36), indeed:

$$K_t = P_t H_t^T R_t^{-1} = (P_{t-1}^{-1} + H_t^T R_t^{-1} H_t)^{-1} H_t^T R_t^{-1} = P_{t-1} H_t^T (R_t + H_t P_{t-1} H_t^T)^{-1}, \tag{93}$$

where we have used the matrix formula: $(A^{-1} + B^T C^{-1} B)^{-1} B^T C^{-1} = A B^T (B A B^T + C)^{-1}$. The Kalman update equation for the covariance matrix (36) is then deduced from (90) using the Woodbury formula:

$$\begin{aligned}
& P_t^{-1} = P_{t-1}^{-1} + H_t^T R_t^{-1} H_t \\
\iff & P_t = P_{t-1} - P_{t-1} H_t^T (R_t + H_t P_{t-1} H_t^T)^{-1} H_t P_{t-1} \\
& \quad = (\mathbb{I} - K_t H_t) P_{t-1}. && (94)
\end{aligned}$$

Equivalence to linear Kalman filter have thus been proved.

The equation (92) can be rewritten to find, combined with (90), the online Newton descent. Indeed, if we pose $Q_t = P_t^{-1}$ as the estimation of the Hessian matrix up to the iteration t we find directly:

$$\mu_t = \mu_{t-1} - Q_t^{-1} \nabla \ell_t(\theta) \tag{95}$$

$$Q_t = Q_{t-1} + H_t^T R_t^{-1} H_t. \tag{96}$$

This proves the equivalence to the online Newton descent. Now it is well known the Kalman filter is optimal for the least mean squares problem, let us reformulate the proof to better show the connection to stochastic optimization.

Let us recall the form of the least mean squares cost function for t observations :

$$\sum_{i=1}^{t} \ell_i(\theta) = \frac{1}{2} \theta^T Q_t \theta - v_t^T \theta + C. \tag{97}$$

We express now the optimal $\theta_t^* = Q_t^{-1} v_t$ at time t in function of the optimal at time $t-1$:

$$\begin{aligned}
\theta_t^* &= Q_t^{-1} v_t \\
&= Q_t^{-1} (v_{t-1} + H_t^T R_t^{-1} y_t) \\
&= Q_t^{-1} (Q_{t-1} \theta_{t-1}^* + H_t^T R_t^{-1} y_t) \\
&= Q_t^{-1} ((Q_t - H_t^T R_t^{-1} H_t) \theta_{t-1}^* + H_t^T R_t^{-1} y_t) \\
&= \theta_{t-1}^* - Q_t^{-1} H_t^T R_t^{-1} H_t \theta_t^* + Q_t^{-1} H_t^T R_t^{-1} y_t \\
&= \theta_{t-1}^* + Q_t^{-1} H_t^T R_t^{-1} (y_t - H_t \theta_{t-1}^*) \\
&= \theta_{t-1}^* - Q_t^{-1} \nabla \ell_t(\theta) && (98)\\
&= \theta_{t-1}^* + K_t (y_t - H_t \theta_{t-1}^*). && (99)
\end{aligned}$$

The two last equations show that the recursive least mean squares estimate is found by both online Newton and the linear Kalman filter. □

7.2 Proof of Theorem 3

Proof. The proof is quite similar to the proof in the linear case. The update equation (3) can be rewritten using the same manipulations as in the linear case:

$$
\begin{aligned}
\mu_t &= \mu_{t-1} - P_{t-1}\mathbf{E}_{q_t}[\nabla_\theta \ell_t(\theta)] && (100)\\
&= \mu_{t-1} + P_{t-1}\mathbf{E}_{q_t}[H_t^T R_t^{-1}(y_t - h(\mu_{t-1}) - H_t(\theta - \mu_{t-1})] \\
&= \mu_{t-1} + P_{t-1}H_t^T R_t^{-1}(y_t - h(\mu_{t-1}) + H_t\mu_{t-1}) - P_{t-1}H_t^T R_t^{-1}H_t\mu_t \\
&\iff \\
\mu_t &= (\mathbf{I} + P_{t-1}H_t^T R_t^{-1}H_t)^{-1}(\mu_{t-1} + P_{t-1}H_t^T R_t^{-1}(y_t - h(\mu_{t-1}) + H_t\mu_{t-1})) \\
&= (P_{t-1}^{-1} + H_t^T R_t^{-1}H_t)^{-1}P_{t-1}^{-1}(\mu_{t-1} + P_{t-1}H_t^T R_t^{-1}(y_t - h(\mu_{t-1}) + H_t\mu_{t-1})) \\
&= \mu_{t-1} + (P_{t-1}^{-1} + H_t^T R_t^{-1}H_t)^{-1}(-H_t^T R_t^{-1}H_t\mu_{t-1} + H_t^T R_t^{-1}(y_t - h(\mu_{t-1}) + H_t\mu_{t-1})) \\
&= \mu_{t-1} + P_t H_t^T R_t^{-1}(y_t - h(\mu_{t-1})) \\
&= \mu_{t-1} - P_t\nabla_\theta \tilde{\ell}_t(\mu_{t-1}) \\
&= \mu_{t-1} - P_t\mathbf{E}_{q_{t-1}}[\nabla_\theta \tilde{\ell}_t(\mu_{t-1})] && (101)\\
&= \mu_{t-1} + K_t(y_t - h(\mu_{t-1})). && (102)
\end{aligned}
$$

From (45) and (101), we deduce that R-VGA is equivalent to its non-averaged version. From (45) and (102), we deduce that R-VGA is equivalent to the extended Kalman filter using the same formula (93) and (94) as in the linear case. The equivalence between the extended Kalman filter and the natural gradient is already known [17], we recall the main argument. In the update equation (45), the information matrix P_t^{-1} is of growing size as long as we observe new data. If we pose $J_t = \frac{1}{t+1}P_t^{-1}$, we can reformulate the update (45) as a moving average:

$$
\begin{aligned}
J_t &= \frac{t}{t+1}J_{t-1} + \frac{1}{t+1}H_t^T R_t^{-1}H_t && (103)\\
&= \frac{t}{t+1}J_{t-1} + \frac{1}{t+1}\mathbf{E}_y\left[\frac{\partial \log p(y|\theta)}{\partial \theta}\frac{\partial \log p(y|\theta)}{\partial \theta}^T\right]\Big|_{\mu_{t-1}} && (104)\\
&= \frac{t}{t+1}J_{t-1} - \frac{1}{t+1}\mathbf{E}_y\left[\frac{\partial^2 \log p(y|\theta)}{\partial \theta^2}\right]\Big|_{\mu_{t-1}} \\
&= \frac{t}{t+1}J_{t-1} + \frac{1}{t+1}\mathbf{E}_y\left[\nabla_\theta^2 \tilde{\ell}_t(\mu_{t-1})\right],
\end{aligned}
$$

which is the Fisher matrix update. The derivation from (103) to (104) is not obvious. [15] introduces it in the context of the generalized Gauss-Newton. To better understand where it comes from we rather use the proof

proposed in [17]. Using the relation $\frac{\partial \log p(y|\theta)}{\partial \eta} = y - \bar{y}$ which holds for exponential families, we can write:

$$
\begin{aligned}
H_t^T R_t^{-1} H_t &= \frac{\partial \bar{y}}{\partial \theta} R_t^{-1} \frac{\partial \bar{y}}{\partial \theta}^T \Big|_{\mu_{t-1}} = \frac{\partial \bar{y}}{\partial \theta} R_t^{-1} R_t R_t^{-1} \frac{\partial \bar{y}}{\partial \theta}^T \Big|_{\mu_{t-1}} \\
&= \frac{\partial \bar{y}}{\partial \theta} \frac{\partial \eta}{\partial \bar{y}} R_t \frac{\partial \eta}{\partial \bar{y}}^T \frac{\partial \bar{y}}{\partial \theta}^T \Big|_{\mu_{t-1}} = \frac{\partial \eta}{\partial \theta} R_t \frac{\partial \eta}{\partial \theta}^T \Big|_{\mu_{t-1}} \\
&= \frac{\partial \eta}{\partial \theta} \mathbf{E}_y \left[(y - \bar{y})(y - \bar{y})^T \right] \frac{\partial \eta}{\partial \theta}^T \Big|_{\mu_{t-1}} \\
&= \frac{\partial \eta}{\partial \theta} \mathbf{E}_y \left[\frac{\partial \log p(y|\theta)}{\partial \eta} \frac{\partial \log p(y|\theta)}{\partial \eta}^T \right] \frac{\partial \eta}{\partial \theta}^T \Big|_{\mu_{t-1}} \\
&= \mathbf{E}_y \left[\frac{\partial \log p(y|\theta)}{\partial \theta} \frac{\partial \log p(y|\theta)}{\partial \theta}^T \right] \Big|_{\mu_{t-1}}.
\end{aligned}
$$

□

7.3 Output uncertainty assessment

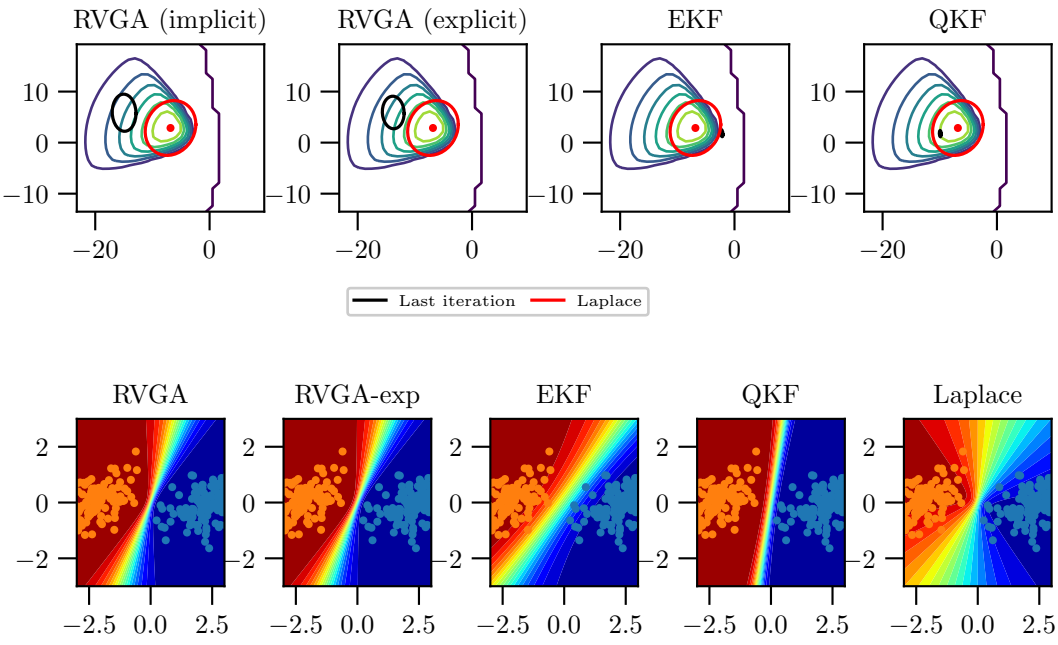

Figure 11: Iso-probabilities of the outputs in function of the inputs for the RVGA implicit and explicit schemes (left), the EKF, QKF (middle) and the Laplace (right). We have considered $\sigma_0 = 10$ and $\mu_0 = 0$ with $N = 300$ and $s = 5$.

Given an unseen input, a prediction based on the estimated distribution q_t, may either be obtained through the maximum a posteriori (MAP) estimate of the parameter, that is, $P[y|x] = \sigma(x^T \theta^*)$, or it may be obtained using the entire (Bayesian) distribution, that is, $P[y|x] = \mathbf{E}[y|x] = \mathbf{E}_{q \sim \mathcal{N}(\mu^*, P^*)}[\sigma(x^T \theta)] \approx \sigma(k x^T \mu^*)$ with

$k = \frac{\beta}{\sqrt{x^T P^* x + \beta^2}}$. Because of this, the latter is less confident than the former MAP based-approach. Indeed we have the following relation for the sigmoid:

$$|\mathbf{E}_q[\sigma(x)] - 0.5| < |\sigma(\mathbf{E}_q[x]) - 0.5| \text{ for } k < 1. \tag{105}$$

This relation comes from the fact the sigmoid is convex for $x < 0$ and concave for $x > 0$. The prediction based on the Bayesian approach are shown in figure 11 where we have drawn the iso-probabilities of the outputs in function of the inputs. On this separable data-set, the Laplace give low probabilities prediction for the unseen inputs whereas the QKF tends to predict with high probabilities. The RVGA give prediction probabilities between both of them.

7.4 Details on the fixed point method

The roots of F are the fixed point of the function $\tilde{F}$ defined by :

$$\tilde{F}_{\alpha_0,\nu_0,y} : \mathbb{R}^2 \to \mathbb{R}^2 \tag{106}$$

$$(\alpha, \nu) \to \tilde{F}(\alpha, \nu) = (\tilde{f}, \tilde{g}) \tag{107}$$

where:

$$\tilde{f}(\alpha, \nu) = -\nu_0 \sigma(\alpha k(\nu)) + \alpha_0 + \nu_0 y \tag{108}$$

$$\tilde{g}(\alpha, \nu) = \frac{\nu_0}{1 + \nu_0 k(\nu) \sigma'(\alpha k(\nu))}. \tag{109}$$

Fonction F is displayed in Figure 12. We found that $\tilde{F}$ is not contractive, so that fixed point iterations shall oscillate and do not converge, see Figure 13. However we can further restrict the admissible domain $[\alpha_{min}, \alpha_{max}] \times [\nu_{min}, \nu_{max}]$ in which the searched point lies as follows.

Coarse bounds are given by:

$$\alpha_{min} = \alpha_0 + \nu_0(y-1) \leq \alpha \leq \alpha_0 + \nu_0 y = \alpha_{max} \tag{110}$$

$$\nu_{min} = \nu_0 (1 - \frac{\nu_0}{4 + \nu_0}) \leq \nu \leq \nu_0 = \nu_{max}. \tag{111}$$

Using the fact that $0 \leq k(\nu_{max}) \leq k(\nu) \leq k(\nu_{min})$, the first inequality (110) gives:

$$a_1 \leq \alpha k(\nu) \leq a_2 \tag{112}$$

where:

$$\alpha_{min} \geq 0 \Rightarrow a_1 = \alpha_{min} k(\nu_{max}), \qquad \alpha_{min} < 0 \Rightarrow a_1 = \alpha_{min} k(\nu_{min}) \tag{113}$$

$$\alpha_{max} \geq 0 \Rightarrow a_2 = \alpha_{max} k(\nu_{min}), \qquad \alpha_{max} < 0 \Rightarrow a_2 = \alpha_{max} k(\nu_{max}) \tag{114}$$

And we find the following new bound for α:

$$\alpha_0 + \nu_0 y - \nu_0 \sigma(a_2) \leq \alpha \leq \alpha_0 + \nu_0 y - \nu_0 \sigma(a_1). \tag{115}$$

For the second inequality (111) we use (112) to bound :

$$b_1 \leq \sigma(\alpha k(\nu))(1 - \sigma(\alpha k(\nu))) = \sigma'(\alpha k(\nu)) \leq b_2, \tag{116}$$

where b_1 and b_2 depends on the sign of $a_1 a_2$:

$$a_1 a_2 > 0 \Rightarrow b_1 = \min(\sigma'(a_1), \sigma'(a_2)) \text{ and } b_2 = \max(\sigma'(a_1), \sigma'(a_2)) \tag{117}$$

$$a_1 a_2 \leq 0 \Rightarrow b_1 = \min(\sigma'(a_1), \sigma'(a_2)) \text{ and } b_2 = 1/4. \tag{118}$$

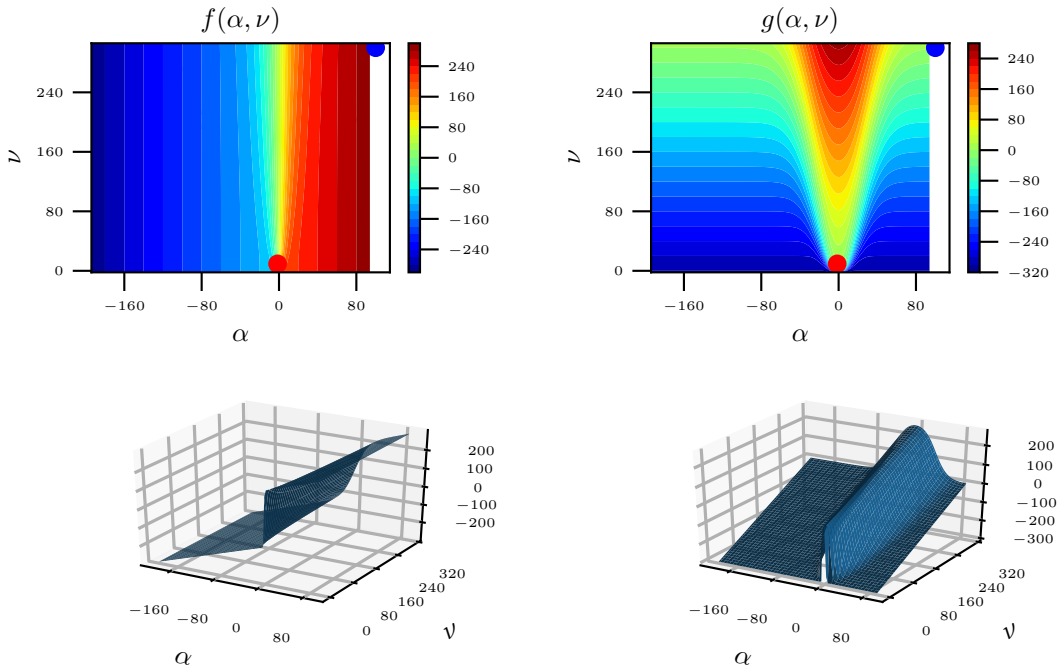

Figure 12: Values of the function $F_{\alpha_0,\nu_0,y}$ for $\alpha_0 = 100$, $\nu_0 = 300$ and $y = 0$. In the upper row, we show the initial value (α_0, ν_0) which is located on the bound of the domain (blue dot) and the optimal value (red dot).

And we find the following new bound for ν:

$$\frac{\nu_0}{1+\nu_0 k(\nu_{min})b_2} \leq \nu \leq \frac{\nu_0}{1+\nu_0 k(\nu_{max})b_1}. \tag{119}$$

This scheme can be iterated to restrict the search domain.

For moderate values of ν_0, that is moderately uncertain prior, the domain shrinks fast. But for highly uncertain priors it does not, as shown in Figure 13, hence the need to resort to a 2D optimisation algorithm.

7.5 Influence of separability of the dataset

We plot here results that reflect the sensitivity to the separability factor s with $s = 2$ (Figure 14), $s = 5$ (Figure 15) and $s = 10$ (Figure 16). The evolution of the ellipsoids over the iterations are also displayed. We see RVG-A consistently yields good performance.

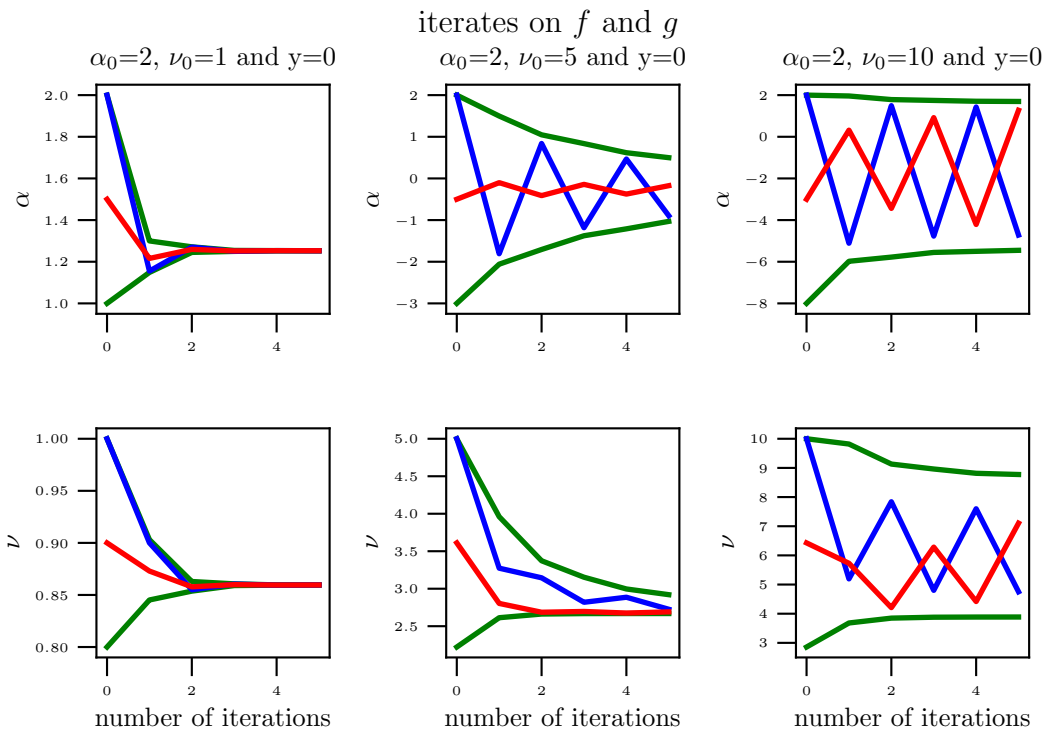

Figure 13: Bounds for the roots α, ν of F through the proposed iterative scheme (in green) for $\nu_0 = 1, 5$ and 10. Iteration on F for different initial conditions are shown in red and blue.

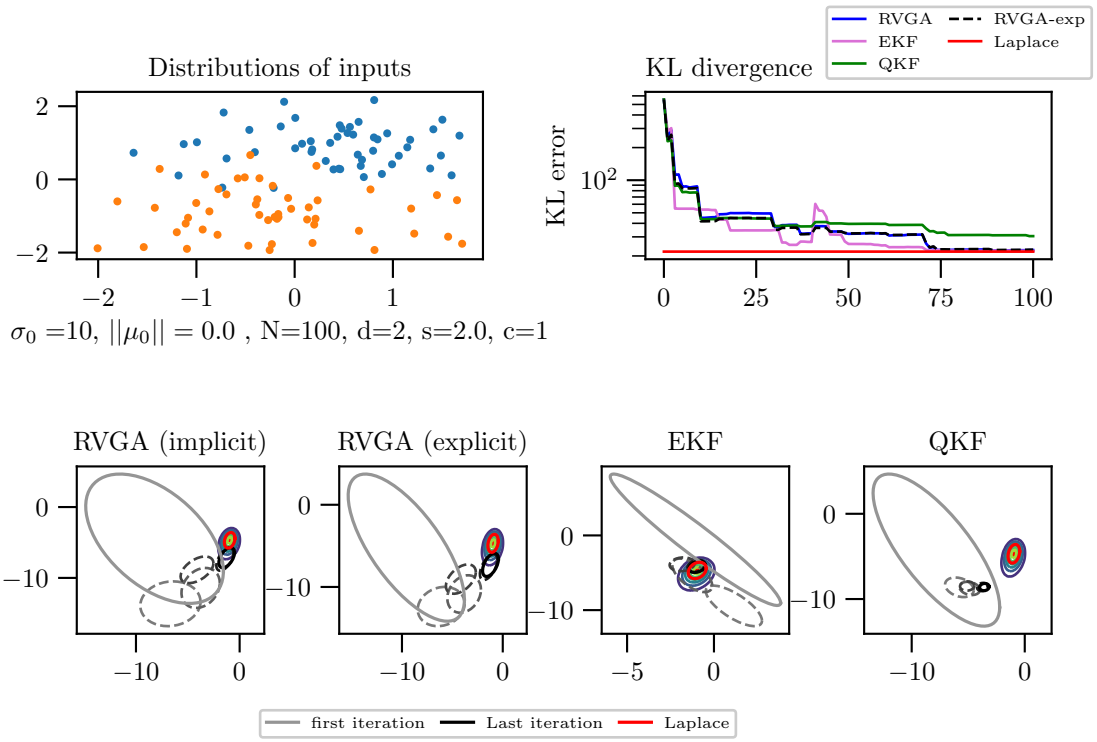

Figure 14: $s = 2$. Upper row: 2D synthetic data-set (left) and KL divergence. Lower row: Confidence ellipsoids for the RVGA implicit and explicit versions (left) and the EKF and QKF (right) at different times (final $t = N$ in black). The Batch Laplace covariance is shown in red and is considered accurate.

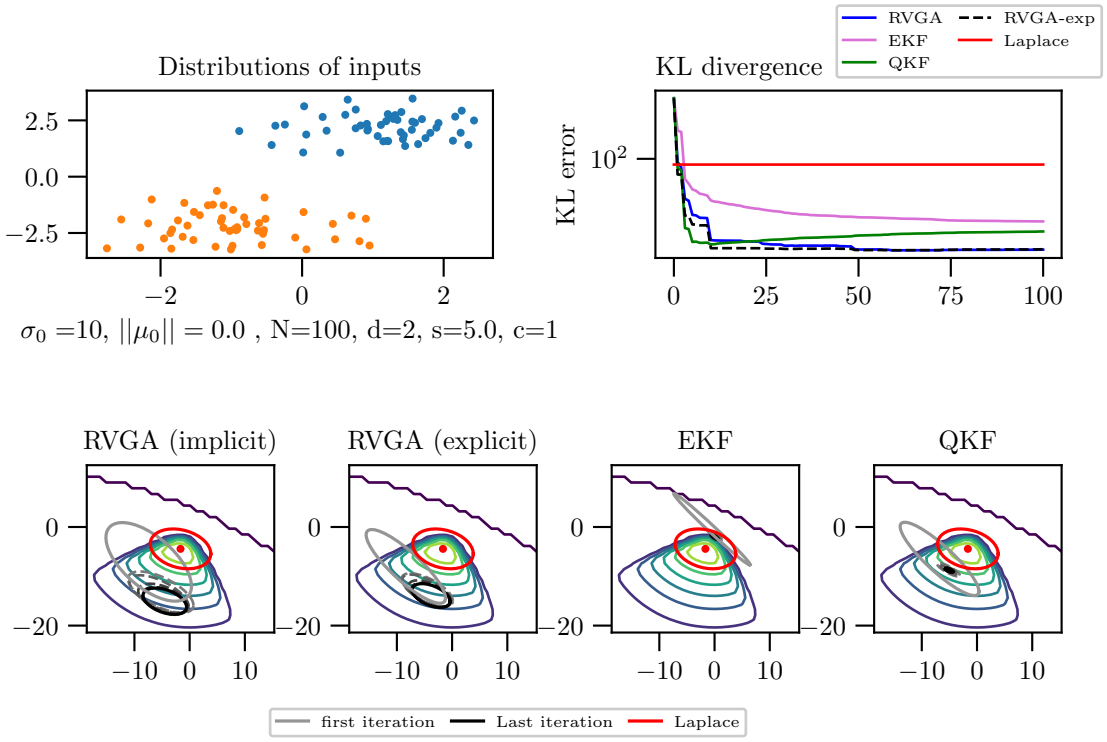

Figure 15: Same as Figure 14 in the case where $s = 5$.

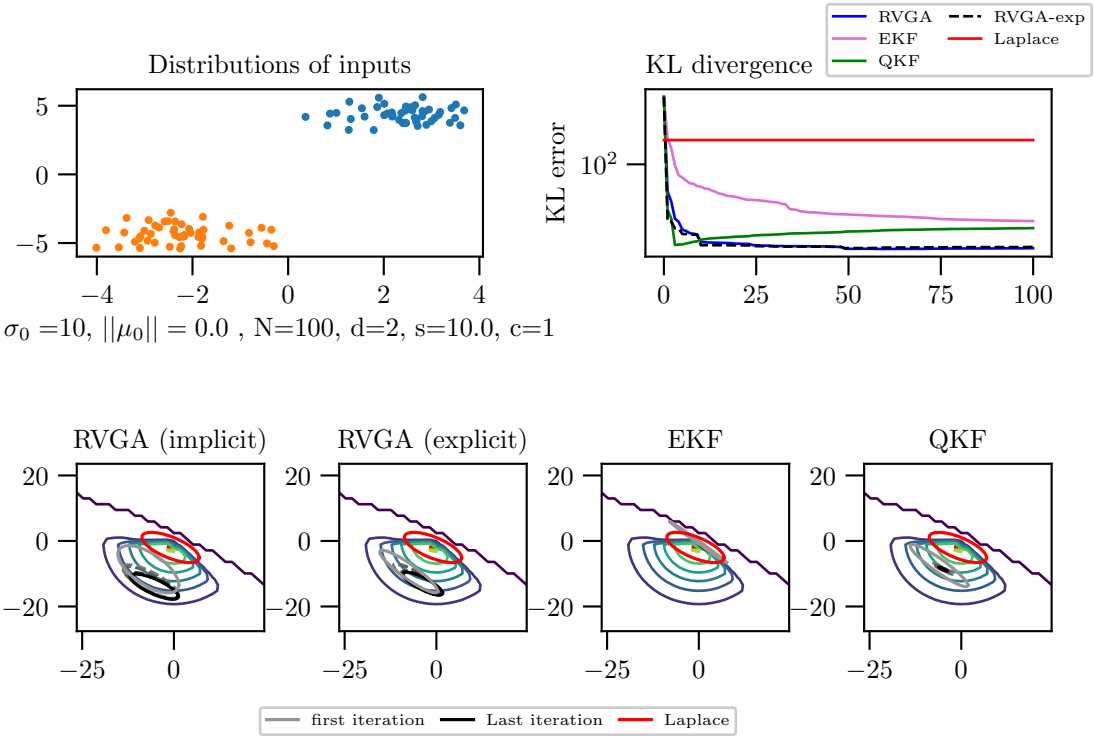

Figure 16: Same as Figure 14 in the case where $s = 10$.