EFFICIENT GAUSSIAN PROCESS BASED ON BFGS UPDATING AND LOGDET APPROXIMATION [1]

W. E. Leithead [*,**,2] **Yunong Zhang** [**] **D. J. Leith** [*]

[*] *Hamilton Institute, National University of Ireland, Maynooth, Co. Kildare, Ireland*
[**] *Dept. of Electronic & Electrical Engineering, University of Strathclyde, Glasgow, U.K.*

Abstract: Gaussian process (GP) is a Bayesian nonparametric regression model, showing good performance in various applications. However, its hyperparameter-estimation procedure suffers from numerous covariance-matrix inversions of prohibitively $O(N^3)$ operations. In this paper, we propose using the quasi-Newton BFGS $O(N^2)$-operation formula to update recursively the inverse of covariance matrix at every iteration. As for the involved log det computation, a power-series expansion based approximation and compensation scheme is proposed with only $50N^2$ operations. A number of numerical tests are performed based on the 2D-sinusoidal regression example and the Wiener-Hammerstein identification example. It is shown that by using the proposed implementation, more than 80% $O(N^3)$ operations are eliminated, and the speedup of $5 \sim 9$ can be achieved.

Keywords: Gaussian process regression, matrix inverse, power series expansion, approximation, compensation

1. INTRODUCTION

The idea of using Gaussian processes for regression has been investigated for a long time; e.g., Mardia and Marshall (1984). Recently, Gaussian processes have been applied to various engineering examples; e.g., Williams and Barber (1998), Leith *et al* (2002), Solak *et al* (2003), Leithead *et al* (2003). Roughly speaking, given some noisy measured data $\mathbb{D} = \{(x_i, t_i)\}_{i=1}^N$, where input vector $x_i \in \mathbb{R}^L$ with L denoting the dimension of input space, output vector $t_i \in \mathbb{R}$, and N is the number of given data points (possibly $\geqslant 1000$), a probabilistic description of the mapping from x to the measured quantity is required. That probabilistic description is a Gaussian process. Some training procedure has to be employed to determine the particular Gaussian process, given the data. Once the Gaussian process has been determined, it may then be used to make predictions at new input points, e.g., t_{N+1} corresponding to x_{N+1}.

The Gaussian process is characterized by the covariance matrix $C(\Theta) \in \mathbb{R}^{N\times N}$, which is a function of a few hyperparameters Θ like length scales and variance scales. In addition to Monte Carlo approach, another standard and general practice for estimating Θ is termed maximum likelihood estimation (MLE); i.e., to minimize the following negative log-likelihood function of $\mathbb{D}$:

[1] This work was supported by Science Foundation Ireland grant, 00/PI.1/C067, and by the EPSRC grant, GR/M76379/01.
[2] Tel.: +44-141-548-2408, Fax: +44-141-548-4203; Emails: w.leithead@eee.strath.ac.uk, ynzhang@ieee.org.

$$\mathcal{L}(\Theta) = \frac{1}{2}\log\det C + \frac{1}{2}\mathbf{t}^T C^{-1}\mathbf{t} \qquad (1)$$

with $\mathbf{t} := [t_1, t_2, \cdots, t_N]$. As the likelihood is in general nonlinear and multimodal, efficient optimization routines usually entail the following gradient information:

$$2\frac{\partial \mathcal{L}}{\partial \Theta_i} = \mathrm{tr}\left(C^{-1}\frac{\partial C}{\partial \Theta_i}\right) - \mathbf{t}^T C^{-1}\frac{\partial C}{\partial \Theta_i}C^{-1}\mathbf{t} \quad (2)$$

with $i = 1, 2, \cdots, L+2$, and $\mathrm{tr}(\cdot)$ denoting the trace operation of a matrix. During the standard MLE model-tuning procedure via (1) and (2), numerous evaluations of $\log\det$ and C^{-1} may be required with prohibitive $O(N^3)$ operations.

Efficient implementation for matrix-inverse problems has been tried in the last decade; e.g., Skilling (1993), Gibbs and MacKay (1997), Williams and Barber (1998), and Seeger (2000) to name a few. Specifically, Skilling (1993) proposed solving for $C^{-1}\mathbf{t}$ based on its conversion to a quadratic-programming problem and using conjugate gradient (CG) methods. Gibbs and MacKay implemented this $O(N^2)$-operation approach in the GP context by minimizing $Q(u) = u^T Cu/2 - u^T\mathbf{t}$ via the Polack-Ribiere CG method. In summarizing this technique, Seeger (2000) pointed out that it is suitable for large sparse matrices with some eigenstructure conditions. Theoretically-speaking, for quadratic programs like $Q(u)$, the CG algorithm converges to true minimum $C^{-1}\mathbf{t}$ after N steps, depending on the condition number of C (Nocedal, 1992). Furthermore, it is observed that the approximation accuracy of $\mathrm{tr}\{C^{-1}(\partial C/\partial\Theta_i)\}$ by using randomized trace estimator is sometimes not sufficiently good in the GP context, especially when only a few seeds are involved. The less favorable performance was also noted in Williams and Barber (1998). In light of $\mathrm{tr}(AB) = \sum_i^N\sum_j^N a_{ij}b_{ji}$ being N^2-operation for known matrices A and B, and also motivated by the above observations, we may have to compute and store C^{-1} for an efficient yet accurate implementation of Gaussian processes. In addition, the $O(N^2)$-operation $\log\det$ approximation is worth investigating, as it is an important part of Wolfe condition for guaranteeing the global convergence and robust performance of optimization routines (Nocedal, 1992; More and Thuente, 1994; Paciorek, 2003).

In this paper, firstly, we propose to approximate explicitly and recursively the inverse of covariance matrix, denoted by $\tilde{C}$, in an $O(N^2)$ fashion by applying the quasi-Newton BFGS method to solving $Q(u)$ at every iteration of the GP model tuning. The approximation accuracy and performance is guaranteed carefully by monitoring if $\mathrm{tr}(\tilde{C}C) \approx N$, otherwise $\tilde{C}$ is reassigned to be C^{-1} before continuing with the model-tuning procedure. Secondly, the power series expansion is used to approximate $\log\det C$ in an $O(N^2)$-operation manner. Three compensation schemes are proposed to further improve the approximation accuracy and computation efficiency of $\log\det$ computation; namely, 1) trace seeds selection and compensation, 2) power-series truncation error compensation, and 3) geometric-series based re-estimation. The detailed proof (Zhang and Leithead, 2004) is omitted due to space limitation. Thirdly, a number of numerical experiments are performed based on two GP regression examples. It is shown that more than 80% $O(N^3)$-operations are eliminated, and a speedup of $5 \sim 9$ can be achieved over the standard GP implementation.

2. GAUSSIAN PROCESS REGRESSION

Now let us return to the general model of data $\mathbb{D}$, $t_i = y(x_i) + v_i$ where v_i is the noise on $y(x_i)$, $i \in \{1, 2, \cdots, N\}$. The data vector $\mathbf{t}$ is defined here as a Gaussian process, a collection of random variables $\mathbf{t} = [t(x_1), t(x_2), \cdots]$ for any set of input $\mathbf{x}$, having the following Gaussian joint distribution under standard zero-mean assumption

$$P(\mathbf{t}|\mathbf{x}, \Theta) \propto \exp\left(-\frac{1}{2}\mathbf{t}^T C^{-1}(\Theta)\mathbf{t}\right).$$

In the above equation, the ij-th entry of covariance matrix $C(\Theta)$ is a scalar covariance function $c_{ij}(\Theta)$, and $\propto$ stands for equality up to a normalizing factor. The conditional Gaussian distribution of t_{N+1} is thus:

$$P(t_{N+1}|\mathbb{D}, \Theta, x_{N+1}) \propto \exp\left(-\frac{(t_{N+1} - \bar{t}_{N+1})^2}{2\sigma_{N+1}^2}\right)$$

where predictive mean $\bar{t}_{N+1} := \kappa C^{-1}(\Theta)\mathbf{t}$ with row vector $\kappa := [c(x_1, x_{N+1}; \Theta), c(x_2, x_{N+1}; \Theta), \cdots, c(x_N, x_{N+1}; \Theta)]$, and the predictive variance $\sigma_{N+1}^2 := c(x_{N+1}, x_{N+1}; \Theta) - \kappa C^{-1}(\Theta)\kappa$. Evidently, given $\mathbb{D}$, the covariance function/matrix (or simply, the hyperparameters Θ) dominates the Gaussian process model. The following covariance function is often used by most researchers:

$$c_{ij} = \alpha\exp\left(\frac{-\|x_i - x_j\|_D^2}{2}\right) + \upsilon\delta_{ij} \qquad (3)$$

where δ_{ij} is the Kronecker delta. Matrix $D := \mathrm{diag}(d_1, d_2, \cdots, d_L)$ characterizes a set of length scales corresponding to each input, and the set of hyperparameters $\Theta := [\alpha, d_1, d_2, \cdots, d_L, \upsilon] \in \mathbb{R}^{L+2}$ with each being nonnegative.

Given the specific form of $c_{ij}(\Theta)$, we have to estimate the distribution of hyperparameters Θ from training data $\mathbb{D}$, in order to make an ideal prediction of t_{N+1}. It is common practice to minimize (1) to obtain the most probable values of hyperparameters for prediction. From the optimization point of view, given specific $\mathbb{D}$ and $c_{ij}(\Theta)$,

the Gaussian process regression actually becomes an unconstrained nonlinear optimization problem depicted in (1) and (2) over hyperparameters Θ. Furthermore, we can see that hyperparameters Θ are always updated iteratively, and in most situations, gradually/slightly. In the standard GP implementation, however, covariance matrix C is inverted directly with $O(N^3)$ operations (like, by Gaussian elimination with partial pivoting), without fully utilizing this iterative nature of Θ.

3. BFGS UPDATE OF C^{-1}

To avoid $O(N^3)$ operations, the CG methods were used to minimize $Q(u) = u^T Cu/2 - u^T \mathbf{t}$ at every iteration of the GP model tuning (Gibbs and MacKay, 1997). Evidently, true minimum $u^* = C^{-1}\mathbf{t}$ because of

$$\nabla Q(u^*) := \left.\frac{dQ(u)}{du}\right|_{u=u^*} = (Cu - \mathbf{t})|_{u=u^*} = 0.$$

The CG solution could be efficient. But, motivated by the necessity of explicit C^{-1} for accurate $\mathrm{tr}\{C^{-1}(\partial C/\partial\Theta_i)\}$ computation, we will apply the quasi-Newton Broyden-Fletcher-Goldfarb-Shanno (BFGS) method to minimizing $Q(u)$ during which covariance inverse $\tilde{C}$ is also updated iteratively.

For solving unconstrained convex quadratic programs like $Q(u)$, if $C^{-1}(u_k)$ is computable with k denoting the kth iteration, the Newton's method of iteration form $u_{k+1} = u_k - C^{-1}(u_k)\nabla Q(u_k)$ is the most powerful algorithm, requiring the fewest number of iterations (here, normally 1) and capable of giving the most accurate answers (Nocedal, 1992). If the evaluation of $C^{-1}(u_k)$ is costly, the central idea of using quasi-Newton methods is to use an approximation $\tilde{C}_k$ instead of $C^{-1}(u_k)$. The approximation $\tilde{C}_k$ is normally updated in a general additive form, like, $\tilde{C}_{k+1} = \tilde{C}_k + \Delta\tilde{C}_k$ where $\Delta\tilde{C}_k$ is an updating matrix. The following BFGS update formula, implementable with only $2N^2$ operations, is generally considered to be the most effective (Nocedal, 1992; MathWorks, 2003):

$$\Delta\tilde{C}_k = \left(1 + \frac{q_k^T\tilde{C}_k q_k}{q_k^T p_k}\right)\frac{p_k p_k^T}{p_k^T q_k} - \frac{p_k q_k^T\tilde{C}_k + \tilde{C}_k q_k p_k^T}{q_k^T p_k}$$

where $p_k := u_{k+1} - u_k$ and $q_k := \nabla Q(u_{k+1}) - \nabla Q(u_k)$. As quasi-Newton methods always include a line search subroutine, the kth iteration of minimizing $Q(u)$ is given as Algorithm 1.

In case of static optimization problems of $Q(u)$ with constant C, the initial guess of approximation $\tilde{C}$, denoted by $\tilde{C}_0$, can be chosen as the identity matrix I. However, for dynamic optimization problems like minimizing $Q(u)$ repeatedly in our GP context, we have to adapt the initialization and termination of quasi-Newton BFGS methods to our GP context. See Algorithm 2.

Algorithm 1. The kth iteration
compute search direction $\varsigma = -\tilde{C}_k\nabla Q(u_k)$
perform line-searches along ς for stepsize η
update solution vector $u_{k+1} = u_k + \eta\varsigma$
update $\tilde{C}_{k+1}$ from $\tilde{C}_k$ with $\Delta\tilde{C}_k$
update the iteration number, $k = k + 1$.

Algorithm 2. The lth MLE epoch
$\Theta_{(l)}$ is updated and then $C(\Theta_{(l)})$
... ...
if $l = 0$
  $\tilde{C}_{(l)} := C^{-1}(\Theta_{(l)})$ and $u^*_{(l)} := \tilde{C}_{(l)}\mathbf{t}$
else
  $k = 0$, $\tilde{C}_0 = \tilde{C}_{(l-1)}$ and $u_0 = u^*_{(l-1)}$
  LOOP
    invoke Algorithm 1 for $\tilde{C}_{(l)}$, $u^*_{(l)}$
    check (4) for terminating the loop
  CONTINUE
  check (5) for usability of $\tilde{C}_{(l)}$
  if not usable
    restart $\tilde{C}_{(l)}$ with $C^{-1}(\Theta_{(l)})$
  end
end
evaluate (1) and (2) using $\tilde{C}_{(l)}$ and $u^*_{(l)}$

3.1 Initialization of Algorithm 1

Review the iterative-update nature of Θ and $C(\Theta)$ in the GP model-tuning procedure. For example, in the lth epoch of MLE optimization, $\Theta_{(l)}$ has been updated from $\Theta_{(l-1)}$ via an optimization step, and $C(\Theta_{(l)})$ is thus changed from $C(\Theta_{(l-1)})$ slightly. When using quasi-Newton method to handle $C(\Theta_{(l)})$, we can re-use the result of the last MLE epoch as the initial guess of the next MLE epoch. For better accuracy and efficiency, in the starting MLE epoch of $l = 0$, $\tilde{C}_{(0)} := C^{-1}(\Theta_{(0)})$ and $u^*_{(0)} := \tilde{C}_{(0)}\mathbf{t}$. Note that here, an iteration is defined as the computation of a search direction and its following line search, while an epoch is defined as the evaluation of an objective function and its gradient like (1) and (2). Specifically, assume that by applying the quasi-Newton BFGS method in the $(l-1)$th MLE epoch, we have obtained the optimal solution $u^*_{(l-1)}$ to the minimization problem of $Q_{(l-1)}(u)$ with good approximation $\tilde{C}_{(l-1)}$ for $C^{-1}(\Theta_{(l-1)})$. When proceeding to the lth MLE epoch with updated $\Theta_{(l)}$ and $C(\Theta_{(l)})$, we re-apply the quasi-Newton BFGS method to solving $Q_{(l)}(u)$ and $\tilde{C}_{(l)}$, but using the assignments $k = 0$, $u_0 = u^*_{(l-1)}$, $\tilde{C}_0 = \tilde{C}_{(l-1)}$ as the initialization of Algorithm 1.

3.2 Termination of Algorithm 1

Another important detail of the quasi-Newton BFGS method is to check for convergence and make termination rules of Algorithm 1. The following termination rules are tailored for our GP context:

$$\|\nabla Q(u_k)\|_\infty \leqslant \epsilon_1 \quad \text{or} \quad m \geqslant m^+, \tag{4}$$

where the threshold $\epsilon_1 := 0.01/N$, and the number of N^2 operations involved in Algorithm 2 is monitored by m that should not exceed its upper bound $m^+ := 100$.

3.3 Restarts Technique

The resultant $\tilde{C}_{(l)}$ via Algorithm 2 is in general accurate enough for $\mathrm{tr}\{C^{-1}(\partial C/\partial\Theta_i)\}$ approximation. But, sometimes $\tilde{C}_{(l)}$ may become "corrupted" and unusable, because of round-off errors, too large steps of $\Theta_{(l)}$ from $\Theta_{(l-1)}$, or other inaccuracies. In light of the nature of the involved trace computations, the following N^2-operation criterion are proposed to check the approximation accuracy and usability of $\tilde{C}_{(l)}$:

$$\left|\frac{\mathrm{tr}(\tilde{C}_{(l)}C(\Theta_{(l)})) - N}{N}\right| \leqslant 10^{-4}. \tag{5}$$

If criterion (5) is satisfied, that means the approximation $\tilde{C}_{(l)}$ is accurate enough and usable for $\mathrm{tr}\{\tilde{C}(\partial C/\partial\Theta_i)\}$ computations. Otherwise, $\tilde{C}_{(l)}$ is not usable and we may have to find alternative ways to remedy the failure; e.g., to use the restarts technique of simply setting $\tilde{C}_{(l)} := C^{-1}(\Theta_{(l)})$. It is worth mentioning that using restarts for $\tilde{C}_{(l+1)}$ is also inspired by the previous research on conjugate-gradient methods (Nocedal, 1992).

4. APPROXIMATE LOGDET

The other problem to be solved in evaluating (1) is to approximate $\log\det C$ with $O(N^2)$ operations. C has to be first transformed to a matrix of eigenvalues less than 1:

$$\log\det C = N\log(\bar{c}) + \log\det A \tag{6}$$

where $\bar{c} := \|C\|_\infty$ and $A := C/\bar{c}$. Then, by Skilling (1993), we know $\log\det A = \mathrm{tr}(\log A)$ and

$$\log(A) = -B - B^2/2 - \cdots - B^w/w - \cdots$$

where $B := I - A$. Thus, $\log\det A = -\mathrm{tr}(B) - \mathrm{tr}(B^2)/2\cdots - \mathrm{tr}(B^w)/w\cdots$ To calculate $\log\det A$ with only $O(N^2)$ operations via the above, the randomized trace estimator can be used instead of calculating B^w explicitly.

Proposition 1. (Barry and Pace, 1999) For matrix $B \in \mathbb{R}^{N\times N}$ and integer w, $\mathcal{E}(s^TB^ws/s^Ts) = \mathrm{tr}(B^w)/N$, where $\mathcal{E}(s^TB^ws/s^Ts)$ is the mean of s^TB^ws/s^Ts and $s \in \mathcal{N}_N(0, I)$. The Gaussian random vector, s, is also termed the seed. □

The simple w-term power-series approximation of $\log\det A$ is thus

$$\log\det A \approx -N\mathcal{E}\left(\sum_{i=1}^{w}\frac{s^TB^is}{is^Ts}\right). \tag{7}$$

However, this approximation may not work well if applied directly to dense covariance matrices in our GP context. The error is mainly from two procedures, i.e., the truncation error in power series expansion (throughout the paper, $w = 30$) and the trace estimation error (only a few seeds or even one used). To make the approximation practically usable, three levels of compensation schemes are proposed as in the following propositions to remedy the approximation error of $\log\det A$.

4.1 Trace-Seed Selection and Compensation

To use only one-seed trace approximation, the first two power-series terms, $\mathrm{tr}(B)$ and $\mathrm{tr}(B^2)$ respectively of $O(N)$ and $O(N^2)$ operations, are fully exploited to compare and compensate the higher-order trace estimations in (7).

The algorithm is as follows. Step 1) generate a seed vector $s \sim \mathcal{N}_N(0, I)$. Step 2) compute the difference Δ_1 between $Ns^TBs/(s^Ts)$ and $\mathrm{tr}(B)$. Step 3) compute the difference Δ_2 between $Ns^TB^2s/(s^Ts)$ and $\mathrm{tr}(B^2)$. Step 4) calculate the weighted difference $\Delta_1 + \Delta_2/2$. Step 5) repeat Steps 1-4 for h times, and save the smallest weighted-difference workspace. Step 6) calculate $\varrho = \Delta_2/\Delta_1$, and compensate $\log\det A$ by the ensuing Proposition 2.

Proposition 2. Given s with trace-estimation errors Δ_1 and Δ_2, the seed compensation for $\log\det A$ in (7) is $-(\Delta_1/\varrho)\log(1-\varrho)$ if $\varrho < 1$ and $\Delta_1/(1-\varrho/2)$ otherwise, where $\varrho = \Delta_2/\Delta_1$, $\Delta_1 = Ns^TBs/(s^Ts) - \mathrm{tr}(B)$, and $\Delta_2 = Ns^TB^2s/(s^Ts) - \mathrm{tr}(B^2)$. □

4.2 Power-Series Truncation Compensation

For the case of the eigenvalue of B near 1, the power series converges very slowly, and the truncation part after the wth term may be relatively very large. The efficient and robust remedy to the one-seed small-w power-series approximation of general dense matrices is to compensate $-N\sum_{i=w+1}^{\infty}s^TB^is/(is^Ts)$ as in the following proposition.

Proposition 3. By calculating the $(w-1)$th and wth power-series terms respectively as $\Lambda_{w-1} = Ns^TB^{w-1}s/(s^Ts)$ and $\Lambda_w = Ns^TB^ws/(s^Ts)$, the power-series truncation-error compensation for $\log\det A$ of (7) is $\Lambda_w\{\log(1-\bar{\lambda}_w)+\sum_{i=1}^{w}\bar{\lambda}_w^i/i\}/\bar{\lambda}_w^w$, where $\bar{\lambda}_w := \Lambda_w/\Lambda_{w-1} < 1$. □

4.3 Geometric-Series Based Re-estimation

Based on the compensation in Propositions 2 and 3, we have a finite sequence during the approxi-

Table 1. Thirty Numerical Tests of Example 1

	$N = 484$	$N = 961$	$N = 1444$	$N = 1936$	$N = 2500$	$N = 2916$
EPO (RST)	160 (28)	135 (25)	95 (15)	100 (16)	90 (15)	75 (12)
ITE (N^2)	2 (9)	3 (12)	2 (9)	2 (8)	2 (8)	2 (8)
SPEEDUP	5.71	5.40	6.33	6.25	6.00	6.25
EPO (RST)	75 (11)	80 (12)	70 (11)	75 (12)	80 (13)	75 (12)
ITE (N^2)	1 (6)	1 (7)	1 (7)	2 (8)	2 (8)	2 (8)
SPEEDUP	6.82	6.67	6.36	6.25	6.15	6.25
EPO (RST)	70 (11)	135 (22)	90 (14)	80 (13)	95 (16)	90 (15)
ITE (N^2)	1 (7)	1 (5)	1 (7)	1 (7)	2 (7)	1 (6)
SPEEDUP	6.36	6.14	6.43	6.15	5.94	6.00
EPO (RST)	80 (14)	80 (13)	85 (13)	90 (15)	85 (13)	90 (15)
ITE (N^2)	2 (11)	2 (9)	2 (8)	2 (10)	2 (9)	2 (8)
SPEEDUP	5.71	6.15	6.54	6.00	6.54	6.00
EPO (RST)	135 (22)	140 (25)	80 (13)	75 (12)	90 (14)	85 (14)
ITE (N^2)	1 (6)	1 (6)	1 (7)	2 (8)	2 (8)	2 (7)
SPEEDUP	6.14	5.60	6.15	6.25	6.43	6.07

Table 2. Thirty Numerical Tests of Example 2

	$N = 500$	$N = 1000$	$N = 1500$	$N = 2000$	$N = 2500$	$N = 3000$
EPO (RST)	154 (19)	224 (26)	154 (19)	126 (14)	119 (14)	147 (17)
ITE (N^2)	2 (8)	1 (5)	1 (6)	1 (4)	1 (7)	1 (6)
SPEEDUP	8.11	8.62	8.11	9.00	8.50	8.65
EPO (RST)	161 (20)	147 (18)	140 (17)	189 (24)	175 (22)	133 (16)
ITE (N^2)	2 (8)	1 (5)	1 (6)	1 (3)	1 (6)	1 (5)
SPEEDUP	8.05	8.17	8.24	7.88	7.95	8.31
EPO (RST)	203 (25)	154 (18)	119 (14)	119 (14)	154 (19)	133 (16)
ITE (N^2)	1 (7)	1 (7)	1 (6)	1 (4)	1 (6)	1 (5)
SPEEDUP	8.12	8.56	8.50	8.50	8.11	8.31
EPO (RST)	182 (22)	189 (24)	133 (16)	133 (16)	168 (21)	154 (19)
ITE (N^2)	2 (8)	1 (5)	1 (6)	1 (4)	1 (5)	1 (5)
SPEEDUP	8.27	7.88	8.31	8.31	8.00	8.11
EPO (RST)	147 (18)	98 (11)	133 (16)	140 (17)	140 (17)	126 (15)
ITE (N^2)	1 (7)	1 (6)	1 (6)	1 (4)	1 (6)	1 (6)
SPEEDUP	8.17	8.91	8.31	8.24	8.24	8.40

mation as w terms sum up; i.e., the g-term result of $\log\det A$ is in (8) and its limiting value is given by Proposition 4.

Proposition 4. By calculating the last three values of Γ_g in (8), i.e., $g = (w-2), (w-1), w$, the limiting value of Γ_w can be estimated as follows based on geometric series:

$$\lim_{w\to\infty} \Gamma_w = \Gamma_{w-2} + \frac{(\Gamma_{w-1} - \Gamma_{w-2})^2}{2\Gamma_{w-1} - \Gamma_{w-2} - \Gamma_w},$$

where for index g varying from 1 to w,

$$\begin{aligned}\Gamma_g := &- N\left(\sum_{i=1}^{g} \frac{s^T B^i s}{i s^T s}\right) \\ &+ \Lambda_g \bar{\lambda}_g^{-g}\left(\log(1-\bar{\lambda}_g) + \sum_{i=1}^{g} \frac{\bar{\lambda}_g^i}{i}\right) \qquad (8) \\ &+ \begin{cases} -\Delta_1 \log(1-\varrho)/\varrho & \text{if } \varrho < 1, \\ \Delta_1/(1-\varrho/2) & \text{otherwise.} \end{cases}\end{aligned}$$

□

Remarks. Evidently, the total computational cost for approximating $\log\det C$ is $(w+2h)N^2$ operations in terms of the leading order (as $w = 30$ and $h = 10$, thus $50N^2$). The geometric-series re-estimation can be done twice or multiple times, or multiplied by a coefficient between 1.0 and 1.1. A large number of numerical tests (totally 50531 randomly-generated matrices of dimensions 484 through 3000) have shown that the average approximation error is only 0.0887, and that the proposed scheme gives a consistent and robust trend of true $\log\det C$ values.

5. NUMERICAL TESTS

Numerical experiments are performed based on two Gaussian-process regression examples; i.e., the 2D-sinusoidal regression example and the Wiener-Hammerstein identification example as depicted in Leithead *et al* (2003). Data sets of size N around 500, 1000, 1500, 2000, 2500 and 3000 are tested five times with random initial conditions. The results are tabulated that include the numbers of MLE epoches, restarts, BFGS iterations and N^2 operations on average for replacing explicit C^{-1}. As shown in the tables (specifically, the last line of every cell), the speedup is a simple estimate of the ratio of the number of N^3 operations involved in optimization using exact inverse to the N^3 operations using the proposed approach.

Example 1. The underlying 2D sinusoidal function is of form $y(x) = \beta_0 \sin(\beta_1 x^{(1)} + \beta_2 x^{(2)})$.

The GP model is trained via the proposed efficient implementation, starting with random $\Theta_{(0)}$. Thirty tests are illustrated in Table 1, where EPO (RST) denotes the numbers of MLE epoches and restarts (with the latter in brackets), ITE (N^2) denotes the numbers of BFGS iterations and N^2 operations (with the latter in brackets) for approximating C^{-1}, and SPEEDUP denotes the estimated speedup ratio corresponding to that trial. On average, for this example, the number of BFGS iterations and N^2 operations required for approximating C^{-1} are respectively 2 and 8. The average speedup is 6.17, and 83% $O(N^3)$ operations are eliminated.

Example 2. Consider a transversal Wiener-Hammerstein system. The system dynamics of interest is $y_i = 0.3(H_1R)^3 + 0.165(H_3R)^3$, $i = 1, 2, \cdots, N$, where the reformulated input over time instants τ_i is defined as $R = [r(\tau_i) \quad r(\tau_{i-1}) \quad r(\tau_{i-2}) \quad r(\tau_{i-3})]^T$ (i.e., $L = 4$), and H_1 and H_3 are defined as [0.9184, 0.3674, 0, 0] and [0, 0, 0.9184, 0.3674], respectively. The output in response to a random input is measured for $0.1N$ seconds with sampling interval 0.1 and Gaussian noise of $v = 0.2$. The GP model is trained via the proposed efficient implementation. Table 2 shows thirty tests, where the meaning of the symbols is the same as in Table 1. On average, for this example, the number of BFGS iterations and N^2 operations for approximating C^{-1} are respectively 1 and 6. The average speedup is 8.29, and 88% $O(N^3)$ operations are eliminated. By carefully using criteria like (4) and (5) and the restarts technique, the proposed efficient implementation is also accurate and robust in the sense that no optimization failures is encountered during the GP model-tuning procedure.

6. CONCLUSIONS

By exploiting the iterative nature of covariance-matrix updating, this paper has proposed using the $2N^2$-operation quasi-Newton BFGS formula to approximate the inverse of covariance matrix recursively. Being another important part of the MLE hyperparameters estimation, the efficient computation of $\log\det$ has also been investigated in a $50N^2$-operation manner. The examples have substantiated that the proposed quasi-Netwon BFGS and power-series approximation/compensation scheme is efficient, accurate and robust for general GP hyperparameters estimation.

REFERENCES

Barry, R. P., & Pace, R. K. (1999). Monte Carlo estimate of the log determinant of large sparse matrices. *Linear Algebra and its Application, 289*: 41-54.

Gibbs, M. N., & MacKay, D. J. C. (1997). Efficient implementation of Gaussian process. *Technical Report*, Cavendish Laboratory, Cambridge University.

Leith, D. J., Leithead, W. E., Solak, E., & Murray-Smith, R. (2002). Divide and conquer identification using Gaussian process priors. *Proceedings of the 41st IEEE Conference on Decision and Control, 1*: 624-629.

Leithead, W. E., Solak, E., & Leith, D. J. (2003). Direct identification of nonlinear structure using Gaussian process prior models. *Proceedings of European Control Conference*, Cambridge.

Mardia, K. V., & Marshall, R. J. (1984). Maximum likelihood estimation for models of residual covariance in spatial regression. *Biometrika, 71*: 135-146.

More, J. J., & Thuente, D. J. (1994). Line search algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software, 20*: 286-207.

Nocedal, J. (1992). Theory of algorithms for unconstrained optimization. *Acta Numerica*, 199-242, Cambridge University Press.

Paciorek, C. J. (2003). Nonstationary Gaussian processes for regression and spatial modelling. *Ph.D. thesis, 3*: 107-108, Department of Statistics, Carnegie Mellon University.

Seeger, M. (2000). Skilling Techniques for Bayesian Analysis. *Technical Report*. Institute for Adaptive and Neural Computation, University of Edinburgh.

Skilling, J. (1993). Bayesian numerical analysis. *Physics and Probability* (Grandy, W. T., & Milonni P., Eds.), Cambridge University Press, 203-221.

Solak, E., Murray-Smith, R., Leithead, W. E., Leith, D. & Rasmussen, C. E. (2003). Derivative observations in Gaussian process models of dynamic systems. *Advances in Neural Information Processing Systems, 15*: 1033-1040, MIT Press.

The MathWorks, Inc. (2003). *Optimization Toolbox User's Guide*, Version 2.3.

Williams, C. K. I., & Barber, D. (1998). Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*: 1342-1351.

Zhang, Y., & Leithead, W. E. (2004). Approximate implementation of logarithm of matrix determinant in Gaussian processes. *Journal of Statistical Computation and Simulation*, under review.