Fast Near-GRID Gaussian Process Regression

Yuancheng Luo **Ramani Duraiswami**
Department of Computer Science,
University of Maryland, College Park,
yluo1@umd.edu, ramani@umiacs.umd.edu

Abstract

Gaussian process regression (GPR) is a powerful non-linear technique for Bayesian inference and prediction. One drawback is its $O(N^3)$ computational complexity for both prediction and hyperparameter estimation for N input points which has led to much work in sparse GPR methods. In case that the covariance function is expressible as a *tensor product kernel* (TPK) and the inputs form a multidimensional grid, it was shown that the costs for exact GPR can be reduced to a sub-quadratic function of N. We extend these exact fast algorithms to sparse GPR and remark on a connection to *Gaussian process latent variable models* (GPLVMs). In practice, the inputs may also violate the multidimensional grid constraints so we pose and efficiently solve missing and extra data problems for both exact and sparse grid GPR. We demonstrate our method on synthetic, text scan, and magnetic resonance imaging (MRI) data reconstructions.

1 Introduction

Bayesian non-parametric methods such as Gaussian processes (GPs) have successfully been used for regression and classification. However prediction and hyperparameter training for GPR present a computational bottleneck when applied to large data sets. For N data points, the $O(N^3)$ cost stems from the solution of a linear system and the inversion of a general covariance matrix. This is problematic for even iterative methods as Magnus et al. (1952) that store the $O(N^2)$ entries of the covariance matrix.

Appearing in Proceedings of the 16^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2013, Scottsdale, AZ, USA. Volume 31 of JMLR: W&CP 31.

Recently, Saatci (2011) and Xu et al. (2012) showed that large computational savings were possible for exact GPR with a TPK covariance function $K(x_j, x_k) = \prod_{i=1}^{D} K_i(x_j, x_k)$ evaluated on multidimensional *gridded* inputs $x \in X$ and $X = X_1 \times X_2 \times \ldots \times X_D$ where $\times$ is the Cartesian outer product between sets $X_i \in \mathbb{R}^{m_i \times *}$. The direct result from these conditions is the efficient Kronecker tensor product (KTP) decomposition described in Van Loan (2000) of the covariance matrix $C = \otimes_{i=1}^{D} C_i$ where $C_i \in \mathbb{R}^{m_i \times m_i}$ where the total number of points is $N = \prod_{i=1}^{D} m_i$.

Several other works have handled lower dimensional KTP decompositions with different treatments of the noise term. In the noiseless case, Rougier (2008) showed that inference via this decomposition is fast. The noise term in Liutkus et al. (2011) was treated as an independent GP and removed from the formulation. In the constant noise case, Bonilla et al. (2008) computed low-rank approximations of the KTP covariance matrix for $D = 2$. The GPR_GRID algorithm from Saatci (2011) efficiently handles the general case of isotropic noise. The general SVD trick after KTP decomposition was also derived in Stegle et al. (2011) and Baldassarre et al. (2011). We note that sparse GPR methods such as GPML by Williams and Seeger (2000) and SPGP/GPSTUFF by Snelson and Ghahramani (2006) have $O(M^2N)$ complexity for latent input size $M << N$ but do not take advantage of this KTP decomposition and grid structure.

Contributions: We present a novel derivation of grid GPR under isotropic noise in section 2.1 and provide computational complexity costs for mean prediction and gradient descent, showing them to be subquadratic in section 2.2. Second, we remark on a connection between TPK covariance and multidimensional grid to GPLVM in Lawrence (2005) and extend the efficient computations to sparse GPR methods in sections 2.3 and 2.4 respectively. Third, we observe that the grid conditions are often violated in practice as a result of corrupted and off-grid inputs; we derive two methods for handling missing and extra data en-

tries from a multidimensional grid in sections 2.5 and 2.6. The additional costs of processing each entry are shown to be linear and quadratic w.r.t. the sizes of input and sparse input sets respectively. Furthermore, this relaxation generalizes standard and sparse GPR with TPK covariance functions as any input set can be contained or appended from a multidimensional grid. We compare the standard and sparse GPR to our grid and sparse grid methods on synthetic high-dimensional tensor products and real text scan and MRI data. Empirical runtime results for full, missing, and extra data cases are shown in section 3.

2 GPR Background

Formally, a GP is a set of random variables $X = [x_1, x_2, \ldots, x_N]$ such that any finite subset is jointly Gaussian. Let the random field $\mathbf{f} = [f(x_1), f(x_2), \ldots, f(x_N)]$ be a vector of random function values drawn from a $N-$variate Gaussian distribution specified by the GP prior mean $m(x)$ and covariance $K(x_i, x_j)$ functions given by

$$\begin{aligned} f(x) \sim GP(m(x), K(x_i, x_j)), \quad m(x) = 0, \\ K(x_i, x_j) = \mathbf{cov}\left(f(x_i), f(x_j)\right). \end{aligned} \tag{1}$$

For the general regression problem, let the model for a target observation y be given by

$$y = f(x) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2), \tag{2}$$

where the noise term ϵ is zero centered with constant variance σ^2. The joint distribution between training y and test outputs f_* under the prior is given by

$$\begin{aligned} &\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X,X) + \sigma^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right), \\ &K_{ff} = K(X,X), \quad \hat{K} = K_{ff} + \sigma^2 I, \\ &K_{f*} = K(X, X_*), \quad K_{**} = K(X_*, X_*), \end{aligned} \tag{3}$$

where X and X_* are the training and test inputs respectively. From Eq. 3 and marginalization over the function space $\mathbf{f}$, the test output conditioned on the test input, training data, and training inputs is normally distributed given by

$$\begin{aligned} P(f_*|X, y, X_*) &\sim \mathcal{N}(\bar{f}_*, \mathbf{cov}\left(f_*\right)), \\ \bar{f}_* = E[f_*|X, y, X_*] &= K_{f*}^T \hat{K}^{-1} y, \\ \mathbf{cov}\left(f_*\right) &= K_{**} - K_{f*}^T \hat{K}^{-1} K_{f*}. \end{aligned} \tag{4}$$

From Eq. 4, the predicted mean and covariance of test output f_* are fully specified by the covariance function K and training outputs y. Once a suitable covariance function K is selected, the hyperparmaeters can be optimized via the negative log-marginal likelihood and its partial derivative functions given by

$$\begin{aligned} -\log p(y|X) &= \frac{1}{2}\left(\log|\hat{K}| + y^T \hat{K}^{-1} y + N\log(2\pi)\right), \\ -\frac{\partial \log p(y|X)}{\partial \theta_i} &= \frac{1}{2}\left(\mathbf{tr}\left(\hat{K}^{-1} P\right) - y^T \hat{K}^{-1} P \hat{K}^{-1} y\right), \end{aligned} \tag{5}$$

where matrix $P = \partial \hat{K} / \partial \theta_i$. Computing $t = \hat{K}^{-1} y$, $t^T P t$, $\log|\hat{K}|$, and $\mathbf{tr}\left(\hat{K}^{-1} P\right)$ requires $\mathrm{O}(N^3)$ flops.

2.1 Formulation of Grid GPR

To establish notation, we denote the covariance matrices K_{f*}, K_{ff}, $\hat{K}$ from Eq. 3 as C_{f*}, C, and $\hat{C}$ respectively. Let matrix C and its partial derivative of w.r.t. parameter θ_π in the j^{th} covariance matrix be the KTPs of D covariance matrices given by

$$\begin{aligned} C &= C_1 \otimes C_2 \cdots \otimes C_D = \otimes_{i=1}^{D} C_i, \\ P &= \otimes_{i=1}^{j-1} C_i \otimes \frac{\partial C_j}{\partial \theta_\pi} \otimes_{l=j+1}^{D} C_l. \end{aligned} \tag{6}$$

The eigendecomposition trick for Kronecker products is as follows: The eigendecomposition of the real symmetric positive definite matrix $C_i = U_i Z_i U_i^T$ where matrices U_i and Z_i are its eigenvectors and diagonal matrix of eigenvalues respectively. Let $\partial C_j / \partial \theta_\pi = V_j W_j V_j^T$ have an analogous eigendecomposition. This allows large matrices C^{-1} and P in Eq. 6 to be expressed as a series of KTPs of smaller eigendecompositions given by

$$\begin{aligned} &U = \otimes_{i=1}^{D} U_i, \quad Z = \otimes_{i=1}^{D} Z_i, \quad C^{-1} = U Z^{-1} U^T, \\ &V = \otimes_{i=1}^{j-1} U_i \otimes V_j \otimes_{l=j+1}^{D} U_l, \\ &W = \otimes_{i=1}^{j-1} Z_i \otimes W_j \otimes_{l=j+1}^{D} Z_l, \quad P = V W V^T, \end{aligned} \tag{7}$$

where the total costs of the eigendecompositions are now $\mathrm{O}(\sum_{i=1}^{D} m_i^3)$ flops and $\mathrm{O}(\sum_{i=1}^{D} m_i^2)$ storage, rather than $\mathrm{O}(N^3)$ flops and $\mathrm{O}(N^2)$ storage.

A vector-KTP (VKTP) and the diagonal of a KTP are computed in in $\mathrm{O}(N)$ flops given by

$$C_{f*} = \otimes_{i=1}^{D} K_i(X_i, *), \quad \mathbf{diag}\left(C\right) = \otimes_{i=1}^{D} \mathbf{diag}\left(C_i\right), \tag{8}$$

where $*$ is set of D inputs. A Kronecker tensor-vector product (KTVP) between D-KTPs of matrices $C_i \in \mathbb{R}^{m_i \times \bar{m}_i}$ and vector y can also be obtained cheaply by decomposing matrix C into D-matrix products of three KTPs given by

$$\begin{aligned} M_i &= \prod_{j=1}^{i-1} m_j, \quad \bar{M}_i = \prod_{j=i+1}^{D} \bar{m}_j, \\ C &= \otimes_{i=1}^{D} C_i = \prod_{i=1}^{D} I_{M_i} \otimes C_i \otimes I_{\bar{M}_i}. \end{aligned} \tag{9}$$

Let $y = \mathbf{vec}(Y)$ be the vectorization of the matrix Y formed from stacking its columns. From Eq. 9, a single matrix-vector product is given by

$$\left((I_{M_i} \otimes C_i) \otimes I_{\bar{M}_i}\right) y = \mathbf{vec}(Y(I_{M_i} \otimes C_i^T)), \quad (10)$$

where matrix $Y \in \mathbb{R}^{\bar{M}_i \times \bar{m}_i M_i}$ and $I_{M_i} \otimes C_i^T$ is block diagonal. For grid GPR where $m_i = |X_i|$, $N = \prod_{i=1}^D m_i$ and $m_i = \bar{m}_i$ for all covariance matrices C_i, computing the block diagonal matrix-vector product in Eq. 10 requires $\mathrm{O}(m_i N)$ flops making the total cost of a KTVP $\mathrm{O}(N \sum_{i=1}^D m_i)$ flops and $\mathrm{O}(N + \sum_{i=1}^D m_i^2)$ storage for KTP matrices C_i and the solution vector.

2.2 Efficient Noisy Grid GPR

In the noiseless case, where matrix $\hat{C} = C$, the terms $t = C^{-1}y$ and $t^T P t$ are successive KTVPs. The log-determinant of matrix $\hat{C}$ is the log-sum of the diagonal of matrix Z in Eq. 7. The trace of matrix product $\hat{C}^{-1}P = \otimes_{i=1}^D C_i^{-1} P_i$ is the sum of its diagonal entries which is found via Eq. 8.

In the noisy case, where matrix $\hat{C} = C + \sigma^2 I$, we rely on the fact that the eigenvectors of the covariance matrix C can be decomposed into products of KTP eigenvectors for all matrices C_i; the noise constant is simply added to the diagonal KTP eigenvalue matrix Z given by

$$\hat{C}^{-1} = (C + \sigma^2 I)^{-1} = U(Z + \sigma^2 I)^{-1} U^T. \quad (11)$$

The inverse-matrix vector product $t = \hat{C}^{-1} y$ is computed via successive KTVPs in via Eq. 11 given by

$$t = \mathbf{KTVP}\left(U, \mathbf{diag}\left(Z + \sigma^2 I\right)^{-1} .* \mathbf{KTVP}\left(U^T, y\right)\right). \quad (12)$$

Computing the term $t^T P t$ follows a similar procedure. The log-determinant is directly found by factorizing the eigendecomposition of matrix C from Eq. 7 with the isotropic noise term $\sigma^2 I$ given by

$$\log|\hat{C}| = \sum_{i=1}^N \log\left(\mathbf{diag}\left(Z\right)_i + \sigma^2\right). \quad (13)$$

The trace is directly found using Eqs. 7, 11 and applying invariance under cyclic permutations given by

$$\begin{aligned} \mathbf{tr}\left(\hat{C}^{-1}P\right) &= \mathbf{diag}\left((Z + \sigma^2 I)^{-1}\right)^T \\ \mathbf{diag}\left(\otimes_{i=1}^{j-1} Z_i \otimes U_j^T V_j W_j V_j^T U_j \otimes_{l=j+1}^{D} Z_l\right), \end{aligned} \quad (14)$$

where diagonals are found via Eq. 8 within $\mathrm{O}(N + m_j^3)$ flops. Note that the full complexity of noisy grid GPR was not presented in Saatci (2011). Based on the above formulation, we can estimate that the total costs of prediction and hyperparameter training are given by

$$\mathrm{O}\left(\sum_{i=1}^D m_i^3 + N\sum_{i=1}^D m_i\right), \quad \mathrm{O}\left(N + \sum_{i=1}^D m_i^2\right), \quad (15)$$

flops and storage respectively. Both are minimized when each sub-matrix has size $m_i = N^{1/D}$. As a result, the asymptotic costs have a best-case sub-quadratic bound of $\mathrm{O}\big(D\left(N^{3/D} + N^{1+1/D}\right)\big)$ flops and $\mathrm{O}(N + DN^{2/D})$ storage.

2.3 Relation to GPLVM

In GPLVM, a mapping between a set of $\tilde{d}_l$ dimensional latent variables $X \in \mathbb{R}^{\tilde{N} \times \tilde{d}_l}$ and a set of $\tilde{d}$-dimensional observations $Y \in \mathbb{R}^{\tilde{N} \times \tilde{d}}$ can be determined by maximizing a GP likelihood with respect to variables X. GPLVM can be shown to be an inverse formulation of grid GPR as the two methods share similar log-marginal likelihood function formulations given by

$$\begin{aligned} \mathcal{L} &= -\frac{1}{2}\left(\tilde{d}\log|\hat{C}| + \mathbf{tr}\left(Y^T\hat{C}^{-1}Y\right) + N\log(2\pi)\right), \\ \mathbf{tr}\left(Y^T\hat{C}^{-1}Y\right) &= y^T\tilde{C}^{-1}y, \quad \tilde{d}\log|\hat{C}| = \log|\tilde{C}|, \end{aligned} \quad (16)$$

where vector $y = \mathbf{vec}(Y) \in \mathbb{R}^N$, $N = \tilde{N}\tilde{d}$, and $\tilde{C} = I_{\tilde{d}} \otimes C + \sigma^2 I_N$ by vectorization. This is interpreted as the addition of inputs $X_1 = [1, \ldots, \tilde{d}]^T$ and a leading Kronecker delta covariance function. In the case that the latent variables X are unconstrained or do not complete a multidimensional grid, GPLVM becomes grid GPR for $D = 2$ and the predicted mean and gradient computations have compact forms.

The inverse-matrix vector product $t = \tilde{C}^{-1}y$ is a discrete time Lyapunov or Sylvester equation given by

$$C^T T + \sigma^2 T = Y, \quad t = \mathbf{vec}(T), \quad (17)$$

in Sorensen and Zhou (2003), which has a standard solution in Bartels and Stewart (1972). The log-term is given by $\log|\tilde{C}| = \tilde{d}\sum_{i=1}^{\tilde{N}} \log\left(\mathrm{diag}(Z)_i + \sigma^2\right)$ while the gradient terms for $\tilde{P} = I_{\tilde{d}} \otimes P$ and $P = \partial\hat{C}/\partial\theta_i$ are given by

$$\begin{aligned} \mathbf{tr}\left(\tilde{C}^{-1}\tilde{P}\right) &= \mathbf{diag}\left(\left(I_{\tilde{d}} \otimes Z + \sigma^2 I_N\right)^{-1}\right)^T \\ \mathbf{diag}\left(I_{\tilde{d}} \otimes (U^T P U)\right), &\quad t^T\tilde{P}t = t^T\mathbf{vec}(P^T T). \end{aligned} \quad (18)$$

If the latent inputs are also be constrained to a multidimensional grid, then the covariance matrix decomposes into KTPs in addition to the leading identity-block matrix.

2.4 Grid GPR with Sparse Structures

A unified framework for sparse GPR Quinonero-Candela and Rasmussen (2005) was presented as a modification of the joint prior $p(f, f_*)$ assuming that the latent function values f are conditionally independent of a set of $M << N$ latent variables $\mathbf{u} = [u_1, \ldots, u_M]^T$ at locations $X^{(u)}$. The approximated joint priors $q(y, f_*)$, after marginalizing out the latent variables $\mathbf{u}$, share the common form given by

$$\begin{aligned} q(y, f_*) &\sim \mathcal{N}\left(0, \begin{bmatrix} \hat{Q} & Q_{f*} \\ Q_{*f} & c \end{bmatrix}\right), \\ \hat{Q} &= Q_{ff} + \wedge, \quad Q_{ff} = K_{fu} K_{uu}^{-1} K_{uf}, \end{aligned} \tag{19}$$

where matrices $K_{uu} = K(X^{(u)}, X^{(u)})$, $K_{fu} = K(X, X^{(u)})$, and $(\wedge, c)$ varies by sparse method. The modified log-marginal likelihood function and its gradient w.r.t. hyperparameter θ_i are similar to Eq. 5 with matrix $\hat{Q}$ replacing $\hat{K}$ and partial derivative matrix $P = \partial\hat{Q}/\partial\theta_i$. We show below that these sparse structures have an analogous efficient formulation when either input or latent variables are gridded. In particular, we extend grid GPR to the subset of regressors (SoR) and deterministic training conditional (DTC) methods that both have the common term $\wedge = \sigma^2 I$. For reference, their GP predicted mean and variance estimates are given by $q(f_*|y) = \mathcal{N}(\sigma^{-2} K_{*u} \Sigma K_{uf} y, c - Q_{**} + K_{*u} \Sigma K_{u*})$ where the economical Gram matrix is $\Sigma = (\sigma^{-2} K_{uf} K_{fu} + K_{uu})^{-1}$.

Case 1: For gridded inputs X and arbitrary latent inputs $X^{(u)}$, the rows of matrix K_{uf} are VKTPs computed from Eq. 8 and matrix products $K_{uf} y$ and $K_{uf} K_{fu}$ only require $\mathrm{O}(MN)$ and $\mathrm{O}(M^2 \sum_{i=1}^{D} m_i)$ flops with $\mathrm{O}(M^2)$ and $\mathrm{O}(M)$ storage respectively.

Case 2: For arbitrary inputs X and gridded latent inputs $X^{(u)}$, the matrix products $K_{uf} y$ and $K_{uf} K_{fu}$ are computed via outer-product decompositions $\sum_{i=1}^{N} K(X^{(u)}, x_i) y_i$ and $\sum_{i=1}^{N} K(X^{(u)}, x_i) K(x_i, X^{(u)})$ in $\mathrm{O}(MN)$ and $\mathrm{O}(NM^2)$ flops with $\mathrm{O}(M)$ and $\mathrm{O}(M^2)$ storage respectively.

Case 3: For gridded inputs X and gridded latent inputs $X^{(u)} = X_1^{(u)} \times X_2^{(u)} \times \ldots \times X_D^{(u)}$ where $m_i^{(u)} = |X_i^{(u)}|$ and $M = \prod_{i=1}^{D} m_i^{(u)}$, matrices K_{fu}, K_{uf}, and K_{uu} are expressible as KTPs and the low-rank decompositions as $K_{uf} K_{fu} = \otimes_{i=1}^{D} K_i^{(uf)} K_i^{(fu)}$. Matrix Σ can also be expressed as products of KTPs with diagonal scaling by the expansion of the inversion of matrix sums; we compute the eigendecompositions of KTP matrices $K_{uu} = \otimes_{i=1}^{D} U_i Z_i U_i^T$ where $U = \otimes_{i=1}^{D} U_i$ and $Z = \otimes_{i=1}^{D} Z_i$ followed by a second set of eigendecompositions of the KTP matrix $Z^{-1/2} U^T K_{uf} K_{fu} U Z^{-1/2} = \otimes_{i=1}^{D} \bar{U}_i \bar{Z}_i \bar{U}_i^T$. Both matrices K_{uu} and $\sigma^{-2} K_{uf} K_{fu}$ have KTP eigendecompositions which expresses matrix Σ as products of KTPs with diagonal scaling given by

$$\begin{aligned} &\Sigma = \sigma^2 \Omega (\bar{Z} + \sigma^2 I)^{-1} \Omega^T, \quad \Omega = U Z^{-1/2} \bar{U}, \\ &\bar{U} = \otimes_{i=1}^{D} \bar{U}_i, \quad \bar{Z} = \otimes_{i=1}^{D} \bar{Z}_i, \end{aligned} \tag{20}$$

and computed within $\mathrm{O}\left(\sum_{i=1}^{D} m_i^{(u)^2} (m_i^{(u)} + m_i)\right)$ operations and $\mathrm{O}\left(\sum_{i=1}^{D} m_i^{(u)} (m_i^{(u)} + m_i)\right)$ storage. While matrix Ω is not orthogonal and $(\bar{Z} + \sigma^2 I)^{-1}$ are not scaled eigenvalues of Σ, the determinant $|\Sigma|$ remains easy to compute as the product of orthogonal matrix determinants cancel to give the expression

$$\log|\Sigma| = \log \sigma^2 + \log|Z| - \log|(Z + \sigma^2 I)|. \tag{21}$$

We omit the remaining gradient terms produced in Quinonero-Candela (2004), which can be easily rearranged for efficient KTP operations.

2.5 Modifying Grid GPR for Missing Data

For missing observations y_r from vector y, denote its corresponding input set by $X^{(r)}$ of size R for row index r_i in matrix C. Note that removing a single row or column c_r from matrix C invalidates its KTP decomposition. However, a single row-column deletion within matrix $\hat{C}$ is given by

$$\begin{aligned} \hat{C} &= \begin{bmatrix} C_{11} & c_{1r} & C_{13} \\ c_{r1}^T & c_{rr} & c_{r3}^T \\ C_{31} & c_{3r} & C_{33} \end{bmatrix} + \sigma^2 I, \\ \bar{C} &= \begin{bmatrix} C_{11} + \sigma^2 I & 0 & C_{13} \\ 0^T & 1 & 0^T \\ C_{31} & 0 & C_{33} + \sigma^2 I \end{bmatrix}, \end{aligned} \tag{22}$$

after zeroing out the r^{th} row-column and replacing the diagonal entry with 1 in the resulting matrix $\bar{C}$. This equates the determinant of matrix $\bar{C}$ and the entries excluding the r^{th} row-column of the inverse matrix $\bar{C}^{-1}$ to that of a row-column deleted matrix C since matrix $\bar{C}$ can be permuted into dense and identity blocks. While this property holds true for multiple row-column deletions in Davis and Hager (2005), a transformation from matrix C to $\bar{C}$ can be expressed as a series of rank-1 updates given by

$$\begin{aligned} &\bar{C} = \hat{C} + aa^T - bb^T, \quad a = \sqrt{\frac{||\bar{c}_r||}{2}} \left(\frac{\bar{c}_r}{||\bar{c}_r||} + e_r\right), \\ &b = \sqrt{\frac{||\bar{c}_r||}{2}} \left(\frac{\bar{c}_r}{||\bar{c}_r||} - e_r\right), \quad \bar{c}_r = \begin{bmatrix} -c_{1r} \\ \frac{1 - c_{rr} - \sigma^2}{2} \\ -c_{3r} \end{bmatrix}, \end{aligned} \tag{23}$$

where vector e_r is the r^{th} column of the identity matrix. Multiple R row-columns are concatenated

into two rank-R updates $\bar{C} = \hat{C} + AA^T - BB^T$ for $A, B \in \mathbb{R}^{N\times R}$ where the columns of matrices A and B follow Eq. 23 for each vector $\bar{c}_r$ and zeroing out entries $A_{r,r+1:R}$ and $B_{r,r+1:R}$.

The inverse of a rank-R update of matrix $\hat{C}$ in Saigal (1993) is efficiently computed by the modified *Woodbury* formulation for diagonal matrix D given by

$$\begin{aligned}(\hat{C} - BB^T)^{-1} &= \hat{C}^{-1} + B^{(R)}DB^{(R)^T},\\ B^{(k)} &= \left[B_1^{(1)}, \ldots, B_k^{(k)}\right] \in \mathbb{R}^{N\times k},\\ D_{ii}^{(k)} &= (1- < B_i, B_i^{(i)} >)^{-1},\end{aligned} \tag{24}$$

and $\log|\hat{C} - BB^T| = \log|\hat{C}| - \log|D|$ for superscript iteration and subscript column index. The column update rule for matrix B is given by

$$B_{k+1}^{(k+1)} = \left(\hat{C}^{-1} + B^{(k)}D^{(k)}B^{(k)^T}\right) B_{k+1}. \tag{25}$$

Matrix $\bar{C}^{-1} = \left((\hat{C} - BB^T) + AA^T\right)^{-1}$ is computed by two rank-R updates in Eqs. 24 and 25 using matrices A, B from Eq. 23 given by

$$\begin{aligned}\bar{C}^{-1} &= \hat{C}^{-1} + B^{(R)}DB^{(R)^T} - A^{(R)}EA^{(R)^T},\\ A^{(k)} &= \left[A_1^{(1)}, \ldots, A_k^{(k)}\right] \in \mathbb{R}^{N\times k},\\ E_{ii}^{(k)} &= (1+ < A_i, A_i^{(i)} >)^{-1},\end{aligned} \tag{26}$$

with the column update rule for matrix A given by

$$A_{k+1}^{(k+1)} = \left(\hat{C}^{-1} + B^{(R)}DB^{(R)^T} - A^{(k)}E^{(k)}A^{(k)^T}\right) A_{k+1}, \tag{27}$$

and $\log|\bar{C}| = \log|\hat{C} - BB^T| - \log|E|$. Note that since matrix $\hat{C}^{-1}$ is expanded with Eq. 12, the costs of computing matrices $B^{(R)}$ and $A^{(R)}$ are $\mathrm{O}(R^2N + RN\sum_{i=1}^{D} m_i)$ flops and $\mathrm{O}(RN)$ storage from the KTVPs. Computing the inverse-matrix vector product $\bar{t} = \bar{C}^{-1}\bar{y}$ uses KTVPs followed by series of $N \times R$ sized matrix-vector products given by

$$\bar{t} = \left(\hat{C}^{-1} + B^{(R)}DB^{(R)^T} - A^{(R)}EA^{(R)^T}\right) \bar{y}, \tag{28}$$

where entries $\bar{t}_{i\in X^{(r)}} = \bar{y}_{i\in X^{(r)}} = 0$; setting these entries in $\bar{t}$ to zero gives a valid expression for the term $\bar{t}^T P \bar{t}$. The trace term $\mathbf{tr}\left(\bar{C}^{-1}P\right)$ is given by

$$\begin{aligned}\mathbf{tr}\left(\bar{C}^{-1}P\right) = \mathbf{tr}\left(\hat{C}^{-1}P\right) + \mathbf{tr}\left(DB^{(R)^T}PB^{(R)}\right)\\ - \mathbf{tr}\left(EA^{(R)^T}PA^{(R)}\right) - \sum_{i\in R} P_{ii},\end{aligned} \tag{29}$$

which requires Eq. 14 followed by the trace of two $R \times R$ matrices computed from $4R$ KTVPs each and the subtraction of the missing data diagonal entries of matrix P; the asymptotic costs remain unchanged from computing matrices $A^{(R)}$ and $B^{(R)}$.

2.6 Modifying Grid GPR for Extra Data

For input sets $X = \{X^{(s)}, X^{(c)}\}$ where set $X^{(s)}$ of size S contain points outside a multidimensional grid $X^{(c)}$, the inverse of its covariance matrix $\hat{K}$ by the block-matrix inversion lemma can be expressed as

$$\begin{aligned}\hat{K} &= \begin{bmatrix}\hat{H} & G^T\\ G & \hat{C}\end{bmatrix}, \quad \hat{H} = K(X^{(s)}, X^{(s)}) + \sigma^2 I,\\ G &= K(X^{(c)}, X^{(s)}), \quad \hat{C} = K(X^{(c)}, X^{(c)}) + \sigma^2 I,\\ \hat{K}^{-1} &= \begin{bmatrix}\bar{H} & -\bar{H}G^T\hat{C}^{-1}\\ -\hat{C}^{-1}G\bar{H} & \hat{C}^{-1}G\bar{H}G^T\hat{C}^{-1} + \hat{C}^{-1}\end{bmatrix},\end{aligned} \tag{30}$$

where matrix $\bar{H} = (\hat{H} - G^T\hat{C}^{-1}G)^{-1}$ and the columns of matrix G are VKTPs. As extra data size S grows, the cost of the matrix inversion $\bar{H}$ dominates with $\mathrm{O}(S^3)$ flops and can be interpreted as performing standard GPR over the arbitrary input set $X^{(s)}$.

Computing the inverse-matrix vector product $t = \hat{K}^{-1}y$ requires Eq. 12 followed by series of $N \times S$ sized matrix-vector products. Since matrix $\hat{C}$ is invertible, the block-determinant of matrix $\hat{K}$ follows Eq. 13 and

$$\log|\hat{K}| = \log|\hat{C}| - \log|\bar{H}|. \tag{31}$$

The gradient terms $t^T P t$ and $\mathbf{tr}\left(\hat{K}^{-1}P\right)$ are computed from the block-partial derivative of matrix $\hat{K}$ and block-matrix products

$$\begin{aligned}P &= \begin{bmatrix}\frac{\partial\hat{H}}{\partial\theta} & \frac{\partial G^T}{\partial\theta}\\ \frac{\partial G}{\partial\theta} & \frac{\partial\hat{C}}{\partial\theta}\end{bmatrix}, \quad \mathbf{tr}\left(\hat{K}^{-1}P\right) = \mathbf{tr}\left(\bar{H}P_{11}\right)\\ &+ \mathbf{tr}\left(\hat{C}^{-1}P_{22}\right) + \mathbf{tr}\left(G^T\hat{C}^{-1}(P_{22}\hat{C}^{-1}G - 2P_{21})\bar{H}\right),\end{aligned} \tag{32}$$

where the matrix product $\hat{C}^{-1}G$ is expanded using Eq. 12 and computed as $2S$ KTVPs.

If the extra data set $X^{(s)}$ is gridded, then the covariance matrix $\hat{H}$ and inverse have analogous Kronecker decompositions to that of matrix $\hat{C}$. Difficulties arise in handling the non-zero noise term σ as the eigenvectors of the block-matrix K are not computed and may not be expressible as a single KTP. In the noiseless case, matrix $\bar{H} = (H + G^T C^{-1} G)^{-1}$ can readily be expressed as products of KTPs with diagonal scaling via Eq. 20 by substituting matrices $H \to K_{uu}$, $G^T C^{-1} G \to K_{uf} K_{fu}$ and removing the σ term; all blocks in $\hat{K}^{-1}$ have KTP structures and the total costs are the sum of two grid GPRs for inputs $X^{(c)}$ and $X^{(s)}$.

3 Experiments

For reference, all experiments are done on an Intel i7-2630QM laptop running Matlab 2010 on 64-bit Win-

dows. All hyperparameters are trained using the natural gradient with resilient back-propagation (RPROP) as Igel and Toussaint (2005) for 10 iterations unless otherwise stated. RPROP locally rescales each hyperparameter via an online stepsize adaptation based the signs of the gradient evaluated once per iteration. The heuristic gives a fast convergence rate and prevents oscillatory behavior compared to standard gradient descent. Compared to nonlinear conjugate gradient, RPROP provides tractable run-time cost analyses due to the absence of line-search.

3.1 Synthetic Data

A D dimensional unit cube centered about the origin is uniformly partitioned by a multidimensional grid of inputs $X = X_1 \times \ldots \times X_D$ with linear spacing of size m along each dimension. The $N = m^D$ outputs y are assigned the Euclidean distance

$$X_j = \left\{ -\frac{1}{2} : \frac{1}{m-1} : \frac{1}{2} \right\}, \quad y_i = \sqrt{\sum_{j=1}^{D} x_{ij}^2}, \qquad (33)$$

from the origin and then corrupted by additive Gaussian white noise $\hat{y}_i = y_i + \mathcal{N}(0, \sigma^2)$.

In all experiments, we specify the well-known squared exponential covariance function $K_i(x_j, x_k) = \exp\left(-\frac{(x_j - x_k)^2}{2\theta_i^2}\right)$ for each dimension and the overall covariance function as $\mathbf{cov}\,(x_j, x_k) = \alpha^2 \prod_{i=1}^{D} K_i(x_j, x_k)$ since the underlying function is smooth. Training the global-scale and length-scale hyperparameters α and θ_i respectively follows the gradient descent of the log-marginal likelihood function. The predicted outputs $\bar{f}$ for inputs X are compared against the noiseless synthetic data via the root mean squared error (RMSE) $\sqrt{\sum_{i=1}^{N} (\bar{f}_i - y_i)^2 / N}$. A log-marginal likelihood of the final iteration and the total runtimes for hyperparameter training are recorded.

Table. 1 shows the results of standard, grid, SoR, and SoR grid GPs trained over the synthetic data $\hat{y}$ for fixed $D = 2$, $m = 32$, and $\sigma = .3$. The **bolded methods** mark our contributions. Note that the dimensions and grid size are kept small for standard GPR to fit in working memory. In both SoR and SoR grid GPR methods, the latent inputs are chosen to be an uniformly spaced multidimensional grid of length $\hat{M} = 4$ replacing m in Eq. 33. The runtime gains of the other methods over standard GPR are apparent. The small RMSE indicates that all methods including SoR GPR and SoR grid GPR approximations approach the true solution. SoR grid GPR runtimes remain similar to that of grid GPR for small m until higher dimensions D as seen in Figure. 1.

Table 1: GPR synthetic data reconstruction

Full Data	**RMSE**	**LH**	**Train sec**
STD-GPR	1.600e-002	222	65
GRID-GPR	1.600e-002	222	0.079
SoR-GPR	1.851e-002	249	0.74
SoR-GRID-GPR	1.851e-002	249	0.063
Missing Data			
STD-GPR	1.601e-002	219	58
GRID-GPR	1.601e-002	219	0.22
SoR-GPR	1.855e-002	247	0.84
SoR-GRID-GPR	1.855e-002	247	0.073
Extra Data			
STD-GPR	1.606e-002	224	66
GRID-GPR	1.606e-002	224	0.27
SoR-GPR	1.860e-002	252	0.79
SoR-GRID-GPR	1.860e-002	252	0.16

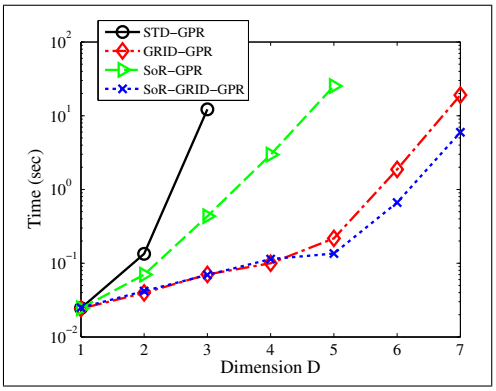

Figure 1: Runtimes for varying dimension D, fixed grid sizes $m = 8$, $\hat{M} = 2$.

In the missing data problem, 1% or $R = 10$ inputs are randomly removed from the synthetic set X with all else equal to the full data experiment. The resulting increase in RMSE and decline in log-marginal likelihood are indicative of the missing data. The runtime for SoR grid GPR remain an order magnitude faster than that of SoR GPR for increasing missing data entries R in Figure. 2. In the extra data problem, 1% or $S = 10$ inputs are randomly generated within the unit cube with all else equal to the full data experiment. The resulting decrease in RMSE and increase in log-marginal likelihood are indicative of added data. The runtimes for SoR grid slowly converge to that of SoR GPR for increasing extra data entries S in Figure. 2. The runtimes of exact grid GPR for both missing and extra data problems are predictably slower than

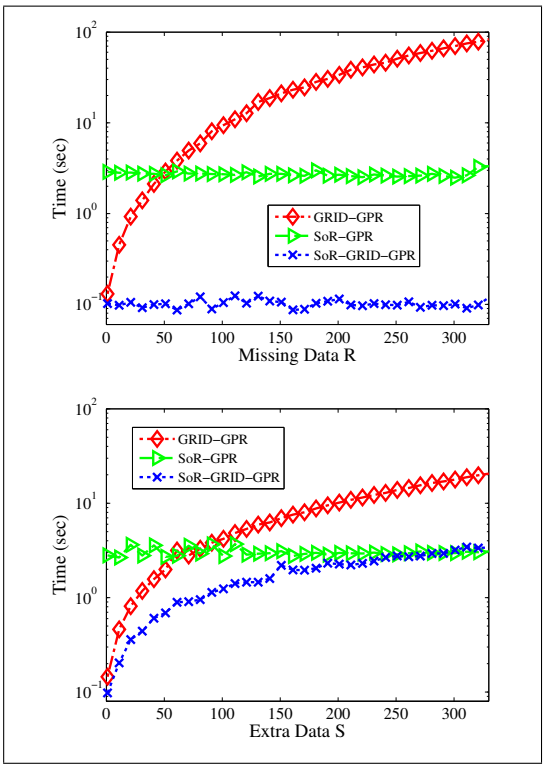

Figure 2: Runtimes for varying missing data size R (top, fixed dimension $D = 3$, grid sizes $m = 16$, $\hat{M} = 2$) and for varying extra data size S (bot, fixed dimension $D = 3$, grid sizes $m = 16$, $\hat{M} = 2$).

the approximation methods. The log-marginal likelihood difference between the two remain constant for increasing missing and extra data entry sizes.

3.2 MRI and Text Scan Data

For the small non-synthetic case, we choose a salt-pepper denoising example of imaged text data. The image is cropped to 32×32 px, scaled to $[0, 1]$ for the output, and inverted so that all methods can be compared without memory and runtime limitations. The row-columns of the image form the multidimensional grid of inputs $X = X_1 \times X_2$ where $X_1 = X_2 = [1 : m]^T$ and $m = 32$. The corrupted pixels shown in Figure. 4 are treated as a variable number R of inputs-outputs pairs that are missing and a GP is trained over the remaining points. We find that specifying the *Ornstein-Uhlenbeck* covariance function $K_i(x_j, x_k) = \exp\left(-\frac{|x_j - x_k|}{\theta_i}\right)$ along each dimension produces the lowest RMSE and largest log-marginal likelihood over the missing data after hyperparameter training and mean prediction. For both SoR and SoR grid methods, the latent inputs are chosen to be an uniformly spaced multidimensional grid of length $\hat{M} = 16$ spanning half the inputs. The performance and accuracy of the reconstructions are shown in Table. 2 after 50 training iterations for fixed $\sigma = .01$ across all methods. As a baseline, the results of averaging the known neighbors of the missing inputs are also reported. Both standard and grid GPR produce identical RMSE and likelihood estimates but with an order difference in the training time. The SoR methods are faster but inaccurate due to the constraint that latent inputs are fixed to a static multidimensional grid. Compared to the baseline neighbor averaging, standard and grid GPR produce visible improvements in the reconstructions as seen in Figure. 4.

Table 2: Small reconstruction, $R = 100$, $N = 1024$

Method	RMSE	LH	Train sec
STD-GPR	0.192	49	230
GRID-GPR	0.192	49	10
SoR-GPR	0.244	-1.18e5	8.3
SoR-GRID-GPR	0.244	-1.18e5	2.5
Neighbor Avg	0.231	-	-

For the large non-synthetic case, we choose multidimensional tensor MRI data from the 3-D Matlab MRI data set. A 2-D horizontal slice is cropped to 97×97 px and the range scaled to $[0, 1]$ for the output. Unlike the synthetic case, the MRI data seen in Figure. 3 are not smooth and so the OU covariance function specified across each dimension performs well. A similar multidimensional grid as the small case is specified over the inputs for $m = 97$ and several regions of interest considered *visual artifacts* are declared as missing data. This is closely related to the inpainting problem for reconstructing lost parts of an image. The first case considers four visual blotches on the scan as missing data, the second case a 3 px wide scanline, and the final case a set of randomized salt and pepper artifacts. The performance and accuracy of the reconstructions are shown in Table. 3 after 50 training iterations for fixed $\sigma = .01$. The results for both standard and SoR GPR methods are omitted due to runtime barriers.

Table 3: GRID-GPR large reconstruction, $N = 9409$

Miss Data (R)	RMSE	LH	Train sec
Blotch: 65	0.199	3809	54
Line: 201	0.136	3719.5	323
Random: 300	0.156	3738	735

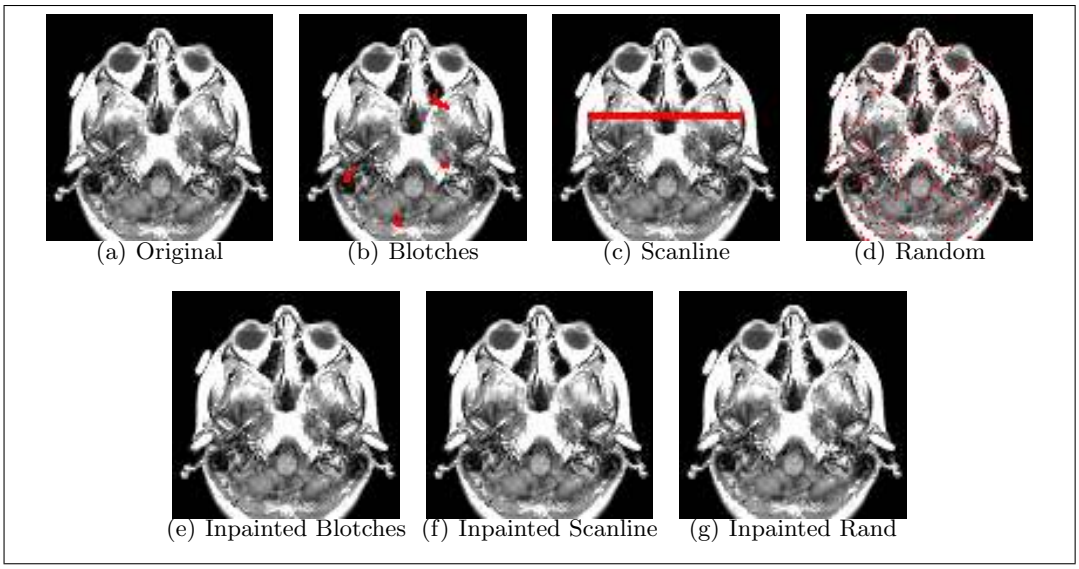

Figure 3: Grid GPR reconstruction of MRI brain slice for missing points in red

Both SoR methods did not converge for varying latent point intervals $\hat{M}$ due to the large variance in the output domain. When inference is performed without hyperparameter training, both SoR methods produce inaccurate results as the posterior variances may be large. Only the exact grid GPR produced interpretable reconstructions within a reasonable time.

These issues may be addressed in future works by allowing the coordinates of the latent multidimensional grid to vary or be trained. This has applications to GPLVM with latent variables constrained to a non-uniform multidimensional grid. Another possibility considers a domain decomposition of the tensor data where the size or density of each latent grid are allowed to vary. In turn, the missing data are partitioned and constrained to local boundaries such that the size of each sub-problem is reduced.

4 Conclusions

We have provided a derivation of multidimensional grid GPR and extended the computational savings to sparse GPR. A connection between grid GPR and GPLVM was made. Two problems for handling missing and extra data for the multidimensional grid were posed and solutions presented with computational costs proportional to the grid size. This allowed grid GPR to address near-multidimensional grid inputs more quickly than standard or non-grid methods. The savings were empirically verified on high-dimensional synthetic data for the full, missing, and extra data problems. Last, we demonstrated missing data reconstruction for denoising and inpainting applications to text scan and MRI data using our fast methods.

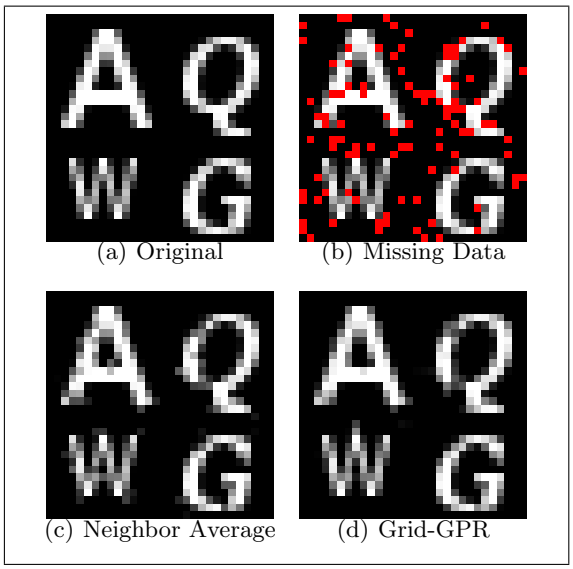

Figure 4: Grid GPR reconstruction of text scan for $R = 100$ missing points in red

Acknowledgments

This work was partially supported by the National Science Foundation (NSF grant IIS-1117716) and Office of Naval Research (MURI grant N00014-08-10638).

References

L. Baldassarre, L. Rosasco, A. Barla, and A. Verri. Multi-output learning via spectral filtering. Technical report, Massachusetts Institute of Technology, 2011.

R. H. Bartels and G. W. Stewart. Solution of the matrix equation AX + XB = C. *Comm. ACM*, 15: 820–826, 1972.

E.V. Bonilla, K.M.A. Chai, and C.K.I. Williams. Multi-task Gaussian process regression. *Advances in Neural Information Processing Systems*, 20:153–160, 2008.

T.A. Davis and W.W. Hager. Row modifications of a sparse Cholesky factorization. *SIAM. J. Matrix Anal. Appl.*, 26:621–639, 2005.

C. Igel and M. Toussaint. Rprop using the natural gradient. *Trends and Applications in Constructive Approximation. International Series of Numerical Mathematics*, 151:259–272, 2005.

N. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

A. Liutkus, R. Badeau, and G. Richard. Gaussian processes for underdetermined source separation. *IEEE Transactions on Signal Processing*, 59:3155–3167, 2011.

R. Magnus, Hestenes, and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

J. Quinonero-Candela. *Learning with Uncertainty - Gaussian Processes and Relevance Vector Machines*. PhD thesis, Technical University of Denmark, 2004.

J. Quinonero-Candela and C.E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

J. Rougier. Efficient emulators for multivariate deterministic functions. *Journal of Computational and Graphical Statistics*, 17:827–843, 2008.

Y. Saatci. *Scalable Inference for Structured Gaussian Process Models*. PhD thesis, University of Cambridge, 2011.

R. Saigal. On the inverse of a matrix with several rank one updates. Technical report, University of Michigan Ann Arbor, 1993.

E Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, 2006.

D.C. Sorensen and Y. Zhou. Direct methods for matrix Sylvester and Lyapunov equations. *Journal of Applied Math*, 6:277–303, 2003.

O. Stegle, C. Lippert, J. Mooij, N. Lawrence, and K. Borgardt. Efficient inference in matrix-variate Gaussian models with iid observation noise. In *Advances in Neural Information Processing Systems*, 2011.

F. Van Loan. The ubiquitous Kronecker product. *Journal of Computational and Applied Mathematics*, 123:85–100, 2000.

C.K.I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, 2000.

Z. Xu, F. Yan, and Y. Qi. Infinite Tucker decomposition: Nonparametric Bayesian models for multiway data analysis. In *International Conference on Machine Learning*, 2012.