Efficient Variational Gaussian Processes Initialization via Kernel-based Least Squares Fitting

Xinran Zhu[1], Jacob R. Gardner[2], David Bindel[1]
[1]Cornell University, (xz584, bindel)@cornell.edu
[2]University of Pennsylvania, jacobrg@seas.upenn.edu

Abstract

Stochastic variational Gaussian processes (SVGP) scale Gaussian process inference to large datasets through inducing points and stochastic training. However, the training process involves hard multimodal optimization, and often suffers from slow and suboptimal convergence when initializing inducing points directly from training data. We provide a better initialization of inducing points from kernel-based least squares fitting. We show empirically that our approach consistently reaches better prediction performance. The total time cost of our method, including initialization, is comparable to the standard SVGP training.

1 Introduction

Gaussian processes (GP) are a popular probabilistic learning framework, especially when inference with uncertainty estimation is necessary [12, 17, 26, 27, 31]. However, exact GP inference costs $\mathcal{O}(n^3)$ in computation and $\mathcal{O}(n^2)$ in storage for n training points. Stochastic variational Gaussian processes (SVGP) have recently shown success in scaling up GP inference on large datasets [8]; the fundamental idea lies in variational GP inference [28]. Variational GPs introduce a small set of *inducing points* and corresponding *inducing values* as a "fake" training dataset, and assume that the inducing data is sufficient for inference. The variational GP model is then trained by minimizing the distance between the true GP posterior and the approximate GP posterior, which involves optimizing the variational Evidence Lower Bound (ELBO) [9, 10]. Furthermore, the SVGP model decomposes the log likelihood term of ELBO into a sum over training labels, thereby enabling stochastic optimization. However, the stochastic training of variational ELBO, which contains model hyperparameters, inducing parameters, and variational distribution parameters, involves hard multimodal optimization problems which have many local minima, and often suffers from slow and suboptimal convergence.

We propose a better initialization method of the inducing points, and get a better initialization of the variational parameters as well as kernel hyperparameters as by-products. The key observation is that the kernel-based least squares function approximation shares the same mean predictor formulation as variational GPs. In the kernel-based least squares function approximation setting, the regularized least squares error can be viewed as a function of the inducing points via variable projection. Therefore, optimizing inducing points boils down to solving a nonlinear least squares problem, which can be solved by standard numerical methods. We demonstrate the efficiency of our initialization in SVGP training by comparing to standard initialization methods, such as random subsampling and K-means initialization, on a variety of real datasets. With our initialization, we show better prediction performance consistently on various real datasets. The total time cost, including initialization computation, is comparable to the standard SVGP training.

2022 NeurIPS Workshop on Gaussian Processes, Spatiotemporal Modeling, and Decision-making Systems.

2 Background

A Gaussian process (GP) is a distribution over function values, denoted as $f \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, where μ is the mean function and k is the covariance function [22]. We assume familiarity with GPs and briefly introduce them for notational clarity. Given data points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^{n \times d}$ and function observations $\mathbf{f} = \{f(\mathbf{x}_i)\} \in \mathbb{R}^n$, a GP prior assumes a multivariate normal distribution $\mathbf{f} \sim \mathcal{N}(\mu_{\mathbf{X}}, K_{\mathbf{XX}})$, where $\mu_{\mathbf{X}}, K_{\mathbf{XX}}$ are the mean values and covariance matrix at data $\mathbf{X}$. Conditioning on noisy observations $\mathbf{y} = \mathbf{f} + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, the posterior distribution at a new data point x^* is $p(\mathbf{f}^*|\mathbf{y}) = \mathcal{N}(\mu^*, \Sigma^{**})$, where $\mu^* = \mu(\mathbf{x}^*) + K_{\mathbf{x}^*\mathbf{X}}(K_{\mathbf{XX}} + \sigma^2 I)^{-1}(\mathbf{y} - \mu_{\mathbf{X}})$ and $\Sigma^{**} = k(\mathbf{x}^*, \mathbf{x}^*) - K_{\mathbf{x}^*\mathbf{X}}(K_{\mathbf{XX}} + \sigma^2 I)^{-1} K_{\mathbf{x}^*\mathbf{X}}^T$. If we assume a prior mean of zero, the posterior mean μ^* becomes $\mu^* = K_{\mathbf{x}^*\mathbf{X}}(K_{\mathbf{XX}} + \sigma^2 I)^{-1}\mathbf{y} = K_{\mathbf{x}^*\mathbf{X}}\mathbf{c} = \sum_{i=1}^n c_i k(\mathbf{x}^*, \mathbf{x}_i)$, where $\mathbf{c} = (K_{\mathbf{XX}} + \sigma^2 I)^{-1}\mathbf{y}$. We generally estimate model hyperparameters such as kernel lengthscale l and noise σ by Maximum Likelihood. The log marginal likelihood function [22] can be optimized by standard numerical solvers such as LBFGS [20] with a cost of $\mathcal{O}(n^3)$ flops for each evaluation.

2.1 Stochastic Variational GP (SVGP)

Variational GPs improve scalability of standard GPs by introducing a set of *inducing points* $\mathbf{U} = \{\mathbf{u}_i\}_{i=1}^m$ with associated inducing values $\mathbf{g} = \{g_i\}_{i=1}^m$ to represent function values at $\mathbf{U}$ under the same GP prior assumption [28, 9, 8]. Therefore, the inference at a new point $\mathbf{x}^*$ becomes $p(\mathbf{f}^*|\mathbf{y}) = \int p(\mathbf{f}^*|\mathbf{f}, \mathbf{g}) p(\mathbf{f}|\mathbf{g}, \mathbf{y}) p(\mathbf{g}|\mathbf{y}) d\mathbf{f} d\mathbf{g}$. If we assume the inducing data is sufficient for inference, i.e., $\mathbf{f}^*$ and $\mathbf{f}$ are independent conditioning on $\mathbf{g}$, we have $p(\mathbf{f}^*|\mathbf{y}) \approx \int p(\mathbf{f}^*|\mathbf{g}) p(\mathbf{g}|\mathbf{y}) d\mathbf{g}$. Further assuming a variational distribution $q(\mathbf{g}) \sim \mathcal{N}(\mathbf{m}, \mathbf{S})$ that approximates the posterior $p(\mathbf{g}|\mathbf{y})$, the inference for the function value $\mathbf{f}^*$ at $\mathbf{x}^*$ is

$$p(\mathbf{f}^*|\mathbf{y}) \approx q(\mathbf{f}^*) = \mathcal{N}(K_{\mathbf{x}^*\mathbf{U}} K_{\mathbf{UU}}^{-1}\mathbf{m}, K_{\mathbf{x}^*\mathbf{x}^*} + K_{\mathbf{x}^*\mathbf{U}}(B - K_{\mathbf{UU}}^{-1}) K_{\mathbf{U}\mathbf{x}^*}), \tag{1}$$

where $B = K_{\mathbf{UU}}^{-1}\mathbf{S}K_{\mathbf{UU}}^{-1}$. The variational distribution $q(\mathbf{g}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$ is then learned by maximizing the variation ELBO [9, 10], which is a lower bound on the log marginal likelihood:

$$\log p(\mathbf{y}) \geq \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}|\mathbf{f})] - \text{KL}[q(\mathbf{g})||p(\mathbf{g})], \tag{2}$$

where $q(\mathbf{f}) = \int p(\mathbf{f}|\mathbf{g}) q(\mathbf{g}) d\mathbf{g}$, and $\text{KL}[q(\mathbf{g})||p(\mathbf{g})]$ is the Kullback-Leibler (KL) divergence [14]. SVGP further decomposes ELBO as a sum of loss over training labels and therefore enables stochastic gradient descent (SGD) [24] training:

$$\text{ELBO}_{\text{SVGP}} = \sum_{i=1}^n \left\{ \log \mathcal{N}(y_i | \mu_{\mathbf{f}}(\mathbf{x}_i), \sigma^2) - \frac{\sigma_{\mathbf{f}}(\mathbf{x}_i)^2}{2\sigma^2} \right\} - \text{KL}\left[q(\mathbf{g})||p(\mathbf{g})\right],$$

where $\mu_{\mathbf{f}}(\mathbf{x}_i), \sigma_{\mathbf{f}}(\mathbf{x}_i)^2$ are the predicted mean and variance from Eq. 1, respectively. To achieve heteroscedastic modeling and improve predictive variances, the Parametric Gaussian Process Regressor (PPGPR) [11] proposes a similar stochastic ELBO loss for SVGP:

$$\text{ELBO}_{\text{PPGPR}} = \sum_{i=1}^n \log \mathcal{N}(y_i | \mu_{\mathbf{f}}(\mathbf{x}_i), \sigma^2 + \sigma_{\mathbf{f}}(\mathbf{x}_i)^2) - \text{KL}[q(\mathbf{g})||p(\mathbf{g})].$$

We empirically apply our initialization method on both $\text{ELBO}_{\text{PPGPR}}$ (SVGP) and $\text{ELBO}_{\text{PPGPR}}$ (PPGPR) training in Section 4. Given inducing point locations, the analytical optimal variational distribution can be obtained by differentiating Eq. 2 [28]:

$$q^*(\mathbf{g}) = \mathcal{N}(\tilde{\mathbf{m}}, \tilde{\mathbf{S}}) = \mathcal{N}(\sigma^{-2} K_{\mathbf{UU}} \Sigma K_{\mathbf{UX}} \mathbf{y}, K_{\mathbf{UU}} \Sigma K_{\mathbf{UU}}), \tag{3}$$

where $\Sigma = (K_{\mathbf{UU}} + \sigma^{-2} K_{\mathbf{UX}} K_{\mathbf{XU}})^{-1}$. Plugging $\tilde{\mathbf{m}}$ back into the variational GP inference in Eq. 1, the predicted mean is

$$\mu_{GP} = K_{\mathbf{x}^*\mathbf{U}}(\sigma^2 K_{\mathbf{UU}} + K_{\mathbf{UX}} K_{\mathbf{XU}})^{-1} K_{\mathbf{UX}} \mathbf{y} := K_{\mathbf{x}^*\mathbf{U}} \mathbf{c}_{GP}. \tag{4}$$

3 Kernel-based Least Squares

Given data points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^{n \times d}$ and noisy observations $\mathbf{y} = \{y_i\} \in \mathbb{R}^n$, a kernel-based function approximation can be formed as $s(\mathbf{x}) = \sum_{i=1}^m c_i k(\mathbf{x}, \mathbf{u}_i)$ [1, 30], where $k(\cdot, \cdot)$ is a positive definite kernel function, and the centers of the kernel basis functions $\mathbf{U} = \{\mathbf{u}_i\}_{i=1}^m$ is a smaller set of points than the training data $\mathbf{X}$ [1, 30], same as inducing points. For a given set of inducing points $\mathbf{U}$, the coefficients $\mathbf{c}$ are solved from a least squares problem:

$$\underset{\mathbf{c} \in \mathbb{R}^m}{\text{minimize}} \, \mathcal{L}(\mathbf{c}) = \underset{\mathbf{c} \in \mathbb{R}^m}{\text{minimize}} \{ \|K_{\mathbf{XU}}\mathbf{c} - \mathbf{y}\|^2 + \sigma^2 \|\mathbf{c}\|_{K_{\mathbf{UU}}}^2 \}, \tag{5}$$

where σ is the regularization parameter, and $\|\mathbf{c}\|^2_{K_{\mathbf{UU}}} = \mathbf{c}'K_{\mathbf{UU}}\mathbf{c}$. The resulting coefficients $\mathbf{c}_{LS}$ from solving Eq. 5 match with SVGP in Eq. 4:

$$\mathbf{c}_{LS} = (\sigma^2 K_{\mathbf{UU}} + K_{\mathbf{UX}}K_{\mathbf{XU}})^{-1}K_{\mathbf{UX}}\mathbf{y} = \mathbf{c}_{GP}.$$

Furthermore the prediction of kernel-based approximation at a new point $\mathbf{x}^*$ also matches with the optimal mean predictor of variational GPs in Eq. 4: $s(\mathbf{x}^*) = K_{\mathbf{x}^*\mathbf{U}}\mathbf{c}_{LS} = K_{\mathbf{x}^*\mathbf{U}}\mathbf{c}_{GP} = \mu_{GP}$. The key observation of the shared mean predictor motivates the proposed initialization method. If $\mathbf{U}$ and kernel hyperparameters θ can be estimated from the least squares approximation, coefficients $\mathbf{c}_{LS}$ can be solved, and then the optimal variational mean $\tilde{\mathbf{m}}$ can be computed:

$$\tilde{\mathbf{m}} = \sigma^{-2}K_{\mathbf{UU}}\Sigma K_{\mathbf{UX}}\mathbf{y} = K_{\mathbf{UU}}(\sigma^2 K_{\mathbf{UU}} + K_{\mathbf{UX}}K_{\mathbf{XU}})^{-1}K_{\mathbf{UX}}\mathbf{y} = K_{\mathbf{UU}}\mathbf{c}_{GP} = K_{\mathbf{UU}}\mathbf{c}_{LS}.$$

If the inducing points $\mathbf{U}$ and kernel hyperparameters θ are good enough to obtain a good mean predictor $\tilde{\mathbf{m}}$, we expect to improve training of variational GPs using these parameters ($\mathbf{U}$, θ and $\tilde{\mathbf{m}}$) as an initialization. See Appendix A for more details of the proposed initialization method.

3.1 Solving for inducing points $\mathbf{U}$ and kernel hyperparameters θ

If we incorporate inducing points $\mathbf{U}$ and kernel hyperparameters θ as unknown variables, Eq. 5 can be equivalently reformulated as

$$\underset{\mathbf{c},\mathbf{U},\theta}{\operatorname{minimize}}\, \mathcal{L}(\mathbf{c},\mathbf{U},\theta) = \underset{\mathbf{c},\mathbf{U},\theta}{\operatorname{minimize}} \left\| \begin{bmatrix} K_{\mathbf{XU}} \\ \sigma L^T_{\mathbf{UU}} \end{bmatrix} \mathbf{c} - \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix} \right\|^2,$$

where $L_{\mathbf{UU}}$ is the Cholesky factor of $K_{\mathbf{UU}}$. To solve for $\mathbf{U}$ and θ first, we use the idea of *variable projection* [7, 21], which was introduced to solve such nonlinear least squares problems where part of the parameters are linear and can be separated from other nonlinear ones [7, 21]. Let $\bar{\mathbf{y}} = [\mathbf{y}; \mathbf{0}]$ and $A = [K_{\mathbf{XU}}; \sigma L^T_{\mathbf{UU}}]$, then the solution is $\mathbf{c}_{LS} = A^\dagger \bar{\mathbf{y}}$ by fixing $\mathbf{U}$ and θ, where $A^\dagger = (A^T A)^{-1}A^T$. Putting the solution $\mathbf{c}_{LS}$ back gives the *projected* problem

$$\underset{\mathbf{U},\theta}{\operatorname{minimize}}\, \mathcal{L}_p(\mathbf{U},\theta) = \underset{\mathbf{U},\theta}{\operatorname{minimize}}\, \|(I - AA^\dagger)\bar{\mathbf{y}}\|^2 := \|r(\mathbf{U},\theta)\|^2, \tag{6}$$

which minimizes the residual $r(\mathbf{U},\theta) = (I - AA^\dagger)\bar{\mathbf{y}}$ as a function of $\mathbf{U}$ and θ and can be solved as a nonlinear least squares problem. Any standard numerical methods apply and we use the Levenberg-Marquardt (LM) algorithm [15, 19]. Specifically, let $\mathbf{p} = (\mathbf{U}, \theta)$ be the variables, and $J_r(\mathbf{p})$ be the Jacobian of $r(\mathbf{p})$. The LM algorithm solves a regularized least squares problem at the k-th iteration:

$$\mathbf{p}^{k+1} = \mathbf{p}^k - (J_r^T(\mathbf{p}^k)J_r(\mathbf{p}^k) + \lambda^2 D_k)^{-1}J_r(\mathbf{p}^k)r(\mathbf{p}^k),$$

where D_k is a scaling matrix, which may be an identity [15] or a diagonal one with column norms of $J_r(\mathbf{p}^k)$ at diagonal [19]. For analytical derivation of $J_r(\mathbf{p})$, see Appendix B. After solving for $\mathbf{U}$ and θ, we then compute for $\tilde{\mathbf{m}}$ as aforementioned, and we use $\mathbf{U}$, θ and $\tilde{\mathbf{m}}$ during the initialization of an SVGP model. Other (hyper)parameters will be initialized by the default way.

4 Experiments

In this section we empirically evaluate the performance of our initialization on both SVGP and PPGPR, denoted as SVGP-lm and PPGPR-lm respectively. We compare to three baseline methods of inducing points initialization: random subsampling from training data, K-means clustering [16, 18] and pivoted Cholesky [4, 3]. For our LM implementation, we use centers from K-means clustering as initialization. We use SVGP and PPGPR models implemented in GPyTorch [6] and use a prior zero mean and a Squared Exponential (SE) kernel. Our experiments were accelerated on a single GPU and code is available upon request.

We consider eight univariate regression UCI datasets [5], with the number of training data ranging from 11250 to 47706 and input dimensions ranging from 8 to 20. We use 500 or 800 inducing points based on training data size. We use an Adam optimizer [13] and separately tune the learning rate for each method on each dataset.

Results were averaged over 10 random train/test/validation splits. As shown in Table 1, SVGP-lm yields significantly lower RMSE than all baselines on all datasets, and so does PPGPR-lm. As

Table 1: Testing RMSE on eight univariate regression datasets (lower is better). Results are averaged over 10 random train/validation/test splits.

	Pol	Elevators	Bike	Kin40k	Protein	Keggdir	Slice	Keggundir
SVGP-random	0.3009	0.3806	0.4421	0.1853	0.5168	**0.08832**	0.1412	0.1218
SVGP-kmeans	0.2995	0.3823	0.4403	0.1836	0.514	0.0886	0.1405	0.1217
SVGP-pivchol	0.3012	0.3808	0.4393	0.1802	0.5177	0.08872	0.1412	0.1217
SVGP-lm	**0.2187**	**0.3703**	**0.3336**	**0.1526**	**0.486**	0.08909	**0.1239**	**0.12**
PPGPR-random	0.3201	0.3931	0.6243	0.2886	0.5102	0.09304	0.229	0.1249
PPGPR-kmeans	0.3229	0.3925	0.6246	0.286	0.5108	**0.08997**	0.2287	0.1245
PPGPR-pivchol	0.3323	0.393	0.6286	0.2892	0.5208	0.09009	0.234	0.1248
PPGPR-lm	**0.2947**	**0.3747**	**0.4499**	**0.2441**	**0.4907**	0.09014	**0.2123**	**0.124**

Table 2: Testing NLL on eight univariate regression datasets (lower is better). Results are averaged over 10 random train/validation/test splits.

	Pol	Elevators	Bike	Kin40k	Protein	Keggdir	Slice	Keggundir
SVGP-random	0.2971	0.4538	0.6112	-0.1945	0.7652	-1.028	-0.4666	-0.6884
SVGP-kmeans	0.2869	0.4577	0.607	-0.197	**0.7584**	-1.026	-0.4743	-0.6869
SVGP-pivchol	0.2875	0.4538	0.6039	-0.229	0.7639	-1.025	-0.4692	-0.6894
SVGP-lm	**0.1211**	**0.4289**	**0.358**	**-0.3675**	0.7959	**-1.031**	**-0.4843**	**-0.6986**
PPGPR-random	-0.1474	0.3785	-0.5554	-0.8402	**0.5654**	-1.628	-1.126	-1.899
PPGPR-kmeans	-0.1596	0.3778	-0.5517	-0.8395	0.5578	**-1.642**	**-1.132**	**-1.908**
PPGPR-pivchol	**-0.1629**	0.3745	-0.5194	**-0.8515**	0.5777	-1.633	-1.113	-1.897
PPGPR-lm	-0.1384	**0.3572**	**-0.6666**	-0.8178	0.5784	-1.617	-1.111	-1.902

shown in Table 2, SVGP-lm consistently shows the best NLL on seven out of eight datasets; while PPGPR-lm only shows comparable results compared to baselines. This shows that our initialization helps finding a better local minimum for the multimodal optimization problem during training. The total time cost of SVGP-lm (including LM and training) is comparable to the standard SVGP training with other baseline initialization methods. See Appendix C for more details such as error bars on regression results, time cost comparison and ablation study.

5 Related Work

The selection and optimization of inducing points is one of the most crucial and hardest parts in variational GP training. The original work of Titsias [28] treated the inducing points as variational parameters to avoid overfitting. Many works have explored better ways of optimizing inducing points during the training. Rossi et al. [25] treated inducing points as model parameters with priors and posteriors and proposed a fully Bayesian treatment of both inducing points and GP hyperparameters. Similarly, Uhrenholt et al. [29] placed a point process prior on the inducing points to select a good number of inducing points in a probabilistic way. Burt et al. [2] provided an asymptotic bound on the KL divergence, which helps with selecting number of inducing points. Our work, combining the ideas of kernel-based least squares fitting and variational GP inference, however, is orthogonal to those works and can be applied to any training methods as an initialization.

6 Conclusion

SVGP enables GP inference on large datasets by combining inducing points methods and stochastic training. However, the stochastic ELBO loss function is highly multimodal with many local minima and training often suffers from slow and suboptimal convergence. Our initialization through kernel-based least squares fitting solves for a good set of inducing points, and yields good initial values for the variational parameters as well as kernel hyperparameters as a by-product. Empirically, our initialization helps SVGP and PPGPR find better local minima and obtain better prediction performance under a comparable total time cost. Our initialization approach can be combined with any other training improvement of SVGP.

References

[1] Martin D Buhmann. *Radial basis functions: theory and implementations*, volume 12. Cambridge university press, 2003.

[2] David Burt, Carl Edward Rasmussen, and Mark Van Der Wilk. Rates of convergence for sparse variational gaussian process regression. In *International Conference on Machine Learning*, pages 862–871. PMLR, 2019.

[3] David R Burt, Carl Edward Rasmussen, and Mark van der Wilk. Convergence of sparse variational inference in gaussian processes regression. *arXiv preprint arXiv:2008.00323*, 2020.

[4] Laming Chen, Guoxin Zhang, and Eric Zhou. Fast greedy map inference for determinantal point process to improve recommendation diversity. *Advances in Neural Information Processing Systems*, 31, 2018.

[5] Dheeru Dua and Casey Graff. UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science, 2017. URL `http://archive.ics.uci.edu/ml`.

[6] Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[7] Gene H Golub and Victor Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal on numerical analysis*, 10(2):413–432, 1973.

[8] James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, page 282–290, Arlington, Virginia, USA, 2013. AUAI Press.

[9] James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. In *Artificial Intelligence and Statistics*, pages 351–360. PMLR, 2015.

[10] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(5), 2013.

[11] Martin Jankowiak, Geoff Pleiss, and Jacob Gardner. Parametric Gaussian process regressors. In *International Conference on Machine Learning*, pages 4702–4712. PMLR, 2020.

[12] David E Jones, David C Stenning, Eric B Ford, Robert L Wolpert, Thomas J Loredo, Christian Gilbertson, and Xavier Dumusque. Improving exoplanet detection power: Multivariate gaussian process models for stellar activity. *arXiv preprint arXiv:1711.01318*, 2017.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

[14] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[15] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944.

[16] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[17] Marco Lorenzi, Gabriel Ziegler, Daniel C Alexander, and Sebastien Ourselin. Efficient gaussian process-based modelling and prediction of image time series. In *International Conference on Information Processing in Medical Imaging*, pages 626–637. Springer, 2015.

[18] J MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297, 1967.

[19] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[20] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[21] Dianne P O'Leary and Bert W Rust. Variable projection for nonlinear least squares problems. *Computational Optimization and Applications*, 54(3):579–593, 2013.

[22] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[23] CE. Rasmussen and CKI. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, USA, 2006.

[24] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

[25] Simone Rossi, Markus Heinonen, Edwin Bonilla, Zheyang Shen, and Maurizio Filippone. Sparse gaussian processes revisited: Bayesian approaches to inducing-variable approximations. In *International Conference on Artificial Intelligence and Statistics*, pages 1837–1845. PMLR, 2021.

[26] Peter Schulam and Suchi Saria. Reliable decision support using counterfactual models. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[27] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[28] Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

[29] Anders Kirk Uhrenholt, Valentin Charvet, and Bjørn Sand Jensen. Probabilistic selection of inducing points in sparse gaussian processes. In *Uncertainty in Artificial Intelligence*, pages 1035–1044. PMLR, 2021.

[30] Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

[31] Zhengkun Yi, Roberto Calandra, Filipe Veiga, Herke van Hoof, Tucker Hermans, Yilei Zhang, and Jan Peters. Active tactile object exploration with gaussian processes. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4925–4930. IEEE, 2016.

A Details of the Initialization Method

In this section, we discuss other details of our initialization method, including the initialization of the variational covariance matrix as a by-product and treatment of the noise variance parameter.

A.1 Uncertainty estimation

One of the strengths of GPs is the uncertainty estimation, which the kernel-based function approximation is not capable of. However, a given set of inducing points $\mathbf{U}$ and kernel hyperparameters θ are sufficient to compute the optimal variational covariance $\tilde{\mathbf{S}}$ in Eq.3 as a by-product:

$$\tilde{\mathbf{S}} = K_{\mathbf{UU}} \Sigma K_{\mathbf{UU}},$$

where $\Sigma = (K_{\mathbf{UU}} + \sigma^{-2} K_{\mathbf{UX}} K_{\mathbf{XU}})^{-1}$. Unlike the variational mean, which could then guarantee an equally good mean prediction of variational GP, there is no guarantee of how the $\tilde{\mathbf{S}}$ would perform in uncertainty estimation using a good set of $\mathbf{U}$ and θ from kernel-base approximation. However, in our empirical study, initializing the variational covariance by the computed optimal convariance $\tilde{\mathbf{S}}$ helps reduce both the training and testing NLL a lot. So we also incorporate the computed $\tilde{\mathbf{S}}$ in our initialization scheme.

A.2 The σ parameter

The σ parameter plays an important role in matching the mean predictor. In the kernel-based least squares function approximation setting, the σ is a regularization parameter in the least squares problem in Eq. 5. While in the variational GP setting, the σ parameter stands for the standard deviation of the observation noise ϵ, where the noise $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ is assumed to be Gaussian. Though mathematically equivalent in the two mean predictor formulations, the treatment of the σ parameter is usally different in these two settings. In a least squares setting, the regularization parameter σ is not solved within the minimization problem in Eq. 5, but there are various standard ways of selecting a good value for regularization parameter σ. While in a GP setting, the noise variance σ^2 is usually optimized together with all model hyperparameters such as kernel hyperparameters θ via Maximum Likelihood [23]. In our initialization, we use fixed σ in the kernel-based least squares fitting setting, and then use the same σ as an initialization of the variational GP.

B Jacobian of the projected problem

In this section we derive the Jacobian $J_r(\mathbf{p})$ of the residual $r(\mathbf{p})$ in the projected problem

$$\underset{\mathbf{p}}{\text{minimize}}\, \mathcal{L}_p(\mathbf{p}) = \underset{\mathbf{p}}{\text{minimize}}\, \|(I - AA^\dagger)\bar{\mathbf{y}}\|^2 := \|r(\mathbf{p})\|^2.$$

Note that $r(\mathbf{p}) = (I - AA^\dagger)\bar{\mathbf{y}}$, where only the matrix A contains $\mathbf{p}$, so it boils down to finding the Jacobian of the residual projector $P = I - AA^\dagger$ with respect to $\mathbf{p}$.

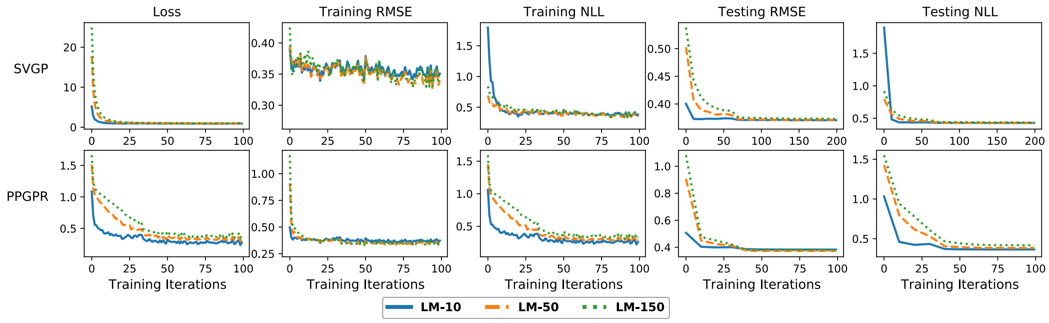

Figure 1: A performance comparison of using different number of LM iterations on the Elevators dataset. LM-X indicates that the initialization uses X LM iteration.

Table 3: Standard error of testing RMSE, averaged over 10 random train/validation/test splits.

	Pol (1e-3)	Elevators (1e-3)	Bike (1e-3)	Kin40k (1e-3)	Protein (1e-3)	Keggdir (1e-3)	Slice (1e-3)	Keggundir (1e-3)
SVGP-random	1.39	1.59	3.47	1.63	1.27	1.01	1.42	0.87
SVGP-kmeans	1.30	1.74	3.42	1.93	1.49	0.97	1.25	0.90
SVGP-pivchol	1.62	1.68	3.13	1.67	1.40	0.98	1.40	0.89
SVGP-lm	1.56	1.89	2.90	0.91	1.15	1.06	1.00	0.96
PPGPR-random	2.13	2.13	4.37	4.90	1.20	1.32	1.81	0.89
PPGPR-kmeans	2.22	2.17	4.59	4.71	1.35	0.96	3.10	0.91
PPGPR-pivchol	2.44	2.20	4.36	5.02	1.39	0.90	1.94	0.95
PPGPR-lm	4.17	2.23	12.0	0.61	1.29	8.85	2.05	0.88

Here we use the *variational notation* to denate derivatives of matrices. For example, δS represent infinitesimal changes to a vector (or matrix) S, and the Jacobian of S can be readily obtained from δS by applying δS to each single dimension of the variables that we differentiate with respect to.

Using the definition of pseudoinverse, we can derive

$$\begin{aligned}\delta P &= -(\delta A)A^\dagger - (A^\dagger)^T(\delta A)^T + (A^\dagger)^T(A^T\delta A + (\delta A)^T A)A^\dagger \\ &= -P(\delta A)A^\dagger - (A^\dagger)^T(\delta A)^T P,\end{aligned}$$

Therefore, for the residual we have

$$\delta r = (\delta A)\bar{y} = -P(\delta A)c - (A^\dagger)^T(\delta A)^T r_0,$$

where $\mathbf{c} = A^\dagger\bar{\mathbf{y}}$ and $r_0 = \mathbf{y} - A\mathbf{c}$. Given the economy QR decomposition of $A = QR$ and observe that $(A^\dagger)^T = QR^{-T}$ we have

$$\begin{aligned}\delta r &= (I - QQ^T)(\delta A)c - QR^{-T}(\delta A)^T r_0 \\ &= -(\delta A)c + QQ^T(\delta A)c - QR^{-T}(\delta A)^T r_0.\end{aligned}$$

C Experiment Details

Regression Details In this section, we provide more detailed results of our empirical evaluation. Table 3 includes standard errors of RMSE results, and Table 4 includes standard errors of NLL results. Table 5 presents the total time cost comparison on SVGP. The time cost of SVGP-lm includes both LM initialization and the subsequent SVGP training. As shown in Table 5, the total time cost of SVGP-lm is comparable to, and sometimes less than, the standard SVGP-random training. We only compare SVGP-lm with SVGP-random because: a) Other baselines share similar time cost as SVGP-random and b) PPGPR shares simliar time cost as the SVGP counterparts.

Table 4: Standard error of testing NLL, averaged over 10 random train/validation/test splits.

	Pol (1e-3)	Elevators (1e-3)	Bike (1e-3)	Kin40k (1e-3)	Protein (1e-3)	Keggdir (1e-3)	Slice (1e-3)	Keggundir (1e-3)
SVGP-random	3.11	3.90	6.59	0.39	2.27	8.09	7.14	6.29
SVGP-kmeans	2.89	4.43	6.50	3.89	2.57	9.35	6.22	6.60
SVGP-pivchol	3.61	4.30	6.08	3.82	2.48	9.24	7.02	6.62
SVGP-lm	7.21	4.68	18.5	3.57	4.13	9.21	4.67	6.18
PPGPR-random	4.49	5.83	12.8	11.8	7.26	29.4	2.89	19.9
PPGPR-kmeans	4.65	6.22	10.4	14.1	6.55	21.5	2.71	16.2
PPGPR-pivchol	5.74	5.64	12.0	5.78	6.56	16.1	3.20	22.1
PPGPR-lm	9.88	5.05	23.2	2.92	3.00	21.7	3.14	27.9

Table 5: Total time cost (min) comparison on SVGP. PPGPR has similar time cost results.

	Pol	Elevators	Bike	Kin40k	Protein	Keggdir	Slice	Keggundir
SVGP-random	3.5	2.5	2.5	15.4	18.2	19.0	21.2	25.1
SVGP-lm	3.5	1.3	5.1	11.2	8.00	10.9	19.6	26.4

Ablation Study on LM Iterations We perform ablation study on different number of LM iterations, and compare how it affects the performance of SVGP-lm and PPGPR-lm. Figure 1 shows the training on the Elevators dataset using results from different LM iterations. On this dataset, we see that using 10 LM iterations is sufficient and increasing the number of LM iterations to 50 or 150 does not significantly increase the training performance.