## A. Details on PPGPR-MFD

The regularizer we use for PPGPR-MFD is given by

$$\begin{aligned}\mathcal{L}_{\mathrm{reg}} &= \beta_{\mathrm{reg}}\Big\{\log\mathcal{N}(\boldsymbol{m},\mathbf{K}^{\mu}) - \mathrm{KL}(\mathcal{N}(\mathbf{0},\mathbf{S}) \mid \mathcal{N}(\mathbf{0},\mathbf{K}^{\sigma}))\Big\} \\ &\to -\tfrac{\beta_{\mathrm{reg}}}{2}\Big\{\mathrm{Tr}\,\mathbf{S}(\mathbf{K}^{\sigma})^{-1} + \boldsymbol{m}^{\mathrm{T}}(\mathbf{K}^{\mu})^{-1}\boldsymbol{m} + \\ &\qquad \log\det\mathbf{K}^{\sigma} + \log\det\mathbf{K}^{\mu} - \log\det\mathbf{S}\Big\}\end{aligned} \tag{23}$$

with $\mathbf{K}^{\sigma} \equiv \mathbf{K}(\mathbf{Z}_{\sigma},\mathbf{Z}_{\sigma})$ and $\mathbf{K}^{\mu} \equiv \mathbf{K}(\mathbf{Z}_{\mu},\mathbf{Z}_{\mu})$, and where we have dropped irrelevant constants. Here $\mathbf{Z}_{\mu}$ and $\mathbf{Z}_{\sigma}$ are the decoupled inducing point locations used to define the mean function and variance function, respectively. This regularizer can be viewed as the sum of two multivariate Normal KL divergences, one of which involves a Dirac delta distribution (with divergent terms dropped).

As discussed in Sec. 3.4, the mean and variance functions for PPGPR-MFD are given by

$$\mu_{\mathbf{f}}(\mathbf{x}_i) = \mathbf{k}_i^{\mu T}(\mathbf{K}^{\mu})^{-1}\boldsymbol{m} \tag{24}$$

and

$$\sigma_{\mathbf{f}}(\mathbf{x}_i)^2 = \tilde{\mathbf{K}}_{ii}^{\sigma} + \mathbf{k}_i^{\sigma T}(\mathbf{K}^{\sigma})^{-1}\mathbf{S}(\mathbf{K}^{\sigma})^{-1}\mathbf{k}_i^{\sigma} \tag{25}$$

where the various kernels in Eqn. 24 and Eqn. 25 share the same hyperparameters and $\mathbf{S}$ is diagonal.

## B. Time and Space Complexity

We briefly describe the time and space complexity of the main algorithms for scalable GP regression discussed in this work. We note that our complexity analysis is similar to that of most sparse Gaussian process methods, such as SVGP (Hensman et al., 2013).

**Training complexity.** Training a VFITC model requires optimizing Eqn. 15, and training PPGPR models requires optimizing Eqn. 17. First, we note that the $\mathrm{KL}(q(\mathbf{u})|p(\mathbf{u}))$ term is the $KL$ divergence between two multivariate Gaussians, which requires $\mathcal{O}(m^3)$ time complexity and $\mathcal{O}(m^2)$ space complexity. For the remaining terms, the main computational bottleneck in both of these equations is computing the term $\mathbf{K}_{MM}^{-1}$. This is usually accomplished by computing its Cholesky factor $\mathbf{L}$, which takes $\mathcal{O}(m^3)$ computation given $m$ inducing points. After the Cholesky factor has been computed, all matrix solves involving $\mathbf{K}_{MM}^{-1}$ require only $\mathcal{O}(m^2)$ computation. The three main terms in both objective functions ($\mu_{\mathbf{f}}(\mathbf{x}_i)$, $\tilde{\mathbf{K}}_{ii}$, and $\mathbf{k}_i^T\mathbf{K}_{MM}^{-1}\mathbf{S}\mathbf{K}_{MM}^{-1}\mathbf{k}_i$) each require a constant number of matrix solves for each data point. All together, the total time complexity of each training iteration is therefore $\mathcal{O}(m^3 + bm^2)$, where $b$ is the number of data points in a minibatch. The space complexity is $\mathcal{O}(m^2 + bm)$ — the size of storing $\mathbf{K}_{MM}$, its Cholesky factor, and all $\mathbf{k}_i$ vectors.

**Prediction complexity.** The predictive distributions for VFITC and PPGPR are given by Eqn. 11. Again, the terms $\mu_{\mathbf{f}}(\mathbf{x}_i)$ and $\sigma_{\mathbf{f}}(\mathbf{x}_i) = \tilde{\mathbf{K}}_{ii} + \mathbf{k}_i^T\mathbf{K}_{MM}^{-1}\mathbf{S}\mathbf{K}_{MM}^{-1}\mathbf{k}_i$ require a constant number of matrix solves with $\mathbf{K}_{MM}^{-1}$ for each $\mathbf{x}_i$. However, we can cache and re-use the Cholesky factor of $\mathbf{K}_{MM}^{-1}$ for all predictive distribution computations, since (after training) we are not updating the inducing point locations or the kernel hyperparameters. Therefore, each predictive distribution has a time complexity of $\mathcal{O}(m^2)$ after the one-time cost of computing/caching the Cholesky factor. It is also worth noting that predictive means can be computed in $\mathcal{O}(m)$ time by caching the vector $\mathbf{K}_{MM}^{-1}\mathbf{m}$ vector. The space complexity is also $\mathcal{O}(m^2)$ (the size of the Cholesky factor).

Note that each of our proposed variants in Section 3.4 have the same computational complexity, as each variant simply modifies the form of $\mathbf{S}$ which is not the computational bottleneck. We do note that the MFD variant requires roughly double the amount of computation and storage, as we are storing/performing solves with two $\mathbf{K}_{MM}$ matrices (one for the predictive means and one for the predictive variances). Finally, we note that the whitening parameterization (see Appendix E) has the same computaitonal/storage complexity, as it also requires computing/storing a Cholesky factor.

## C. Experimental Details

We use zero mean functions and Matérn kernels with independent length scales for each input dimension throughout. All models and experiments are implemented using the GPyTorch framework (Gardner et al., 2018) and the Pyro probabilistic programming language (Bingham et al., 2019).

### C.1. Univariate regression

We use the Adam optimizer for optimization with an initial learning rate of $\ell = 0.01$ that is progressively decimated over the course of training (Kingma and Ba, 2014). We use a mini-batch size of $B = 10^4$ for the Buzz, Song, 3droad and Houseelectric datasets and $B = 10^3$ for all other datasets. We train for 400 epochs except for the Houseelectric dataset where we train for 200 epochs. Except for the Exact results, we do 10 train/test/validation splits on all datasets (always in the proportion 15:3:2, respectively). In particular for the Exact results we do 3 train/test/validation splits on the smaller datasets and one split for the two largest (3Droad and Song). All datasets are standardized in both input and output space; thus a predictive distribution concentrated at zero yields a root mean squared error of unity. We use $M = 1000$ inducing points initialized with kmeans. In the case of OD-SVGP and PPGPR-MFD we use $M = 1000$ inducing points for the mean and $M = 1000$ inducing points for the (co)variance. We use the validation set to determine a small set of hyperparameters. In particular for SVGP, OD-SVGP, and VFITC we search over $\beta_{\mathrm{reg}} \in \{0.1, 0.3, 0.5, 1.0\}$. For $\gamma$-Robust we search over $\{1.01, 1.03, 1.05, 1.07\}$ (with $\beta_{\mathrm{reg}} = 1$). For all PPGPR variants we search over $\beta_{\mathrm{reg}} \in \{0.01, 0.05, 0.2, 1.0\}$. For MAP we fix $\beta_{\mathrm{reg}} = 1$.

### C.2. PPGPR Ablation

The experimental procedure for the results reported in Sec. 5.2 follows the procedure described in Sec. C.1.

### C.3. DKL calibration

MC-Dropout has two hyperparameters that must be set by hand: a dropout proportion $p$ and a prior variance inflation term $\tau$. These were set by temporarily removing a portion of the training data as a validation set and performing a small grid search on each dataset over $p \in [0.05, 0.1, 0.15, 0.2, 0.25]$ and $\tau \in [0.05, 0.1, 0.25, 0.5, 0.75, 1.0]$. For PPGPR-MFD, $\beta \in [0.05, 0.2, 0.5, 1.0]$ was chosen in a similar fashion. The datasets for each of the three cities contain 30 features that encode various aspects of a trip like origin and destination location, time of day and week, as well as various rudimentary routing features.

For all three methods, we use the same five layer fully connected neural network, with hidden representation sizes of $[256, 256, 128, 128, 64]$ and ReLU nonlinearities. We use the Adam optimizer and use an initial learning rate of $\ell = 0.01$, which we drop by a factor of $0.1$ at $100$ and $150$ epochs. We train for 200 epochs for all three methods. For the GP methods, we use $M = 1024$ inducing points, initialized by randomly selecting training data points and passing them through the initial feature extractor.

## D. Additional Experimental Results

In Fig. 6-9 we depict summary results for all the experiments in Sec. 5.1-5.2 in the main text. In particular in Fig. 9 we depict Continuous Ranked Probability Scores (Gneiting and Raftery, 2007). We also include a complete compilation of our results in Table 3.

**The effect of $\beta_{\mathrm{reg}}$** We find empirically that PPGPR is robust to the value of $\beta_{\mathrm{reg}}$. On the five smallest UCI datasets, the test RMSE varies by no more than 5% (relative) and the LL by no more than 0.05 nats as we vary $\beta_{\mathrm{reg}}$ from 0 to 1. This is not unexpected, since we are always in the regime $M \ll N$.

**Comparison to Raissi et. al.** We do not include a full comparison to the method in Raissi et al. (2019) because we find that it is outperformed by all the other baselines we consider. In particular while this method can achieve middling RMSEs, on many datasets it achieves very poor log likelihoods. For example, using the authors' implementation[14] we find a test NLL of $\sim 26$ nats on the Elevators dataset and $\sim 6$ nats on the Pol dataset. This performance can be traced to the incoherency of the two-objective approach adopted by this method. In particular the logic of the derivation makes it unclear whether the noise term in the kernel should be included during test time; this ambiguity is reflected in the authors' code,[15] where this contribution is by default commented out. While the NLL performance can be improved by including this term, the larger point is that this approach does not provide a coherent account of function uncertainty.

**In-Sample Comparison to Exact GPs** In this section, we compare the fit of an exact GP, of SVGP, and of PPGPR to samples drawn from a Gaussian process prior with a Matern kernel and a Periodic kernel. For the draw from the Matern kernel, we use a lengthscale of 0.1 and an outputscale of 1. For the periodic kernel, we use a period length of 0.2 and outputscale of 1. For both kernels, we draw the function on the range $[0, 1]$, and fit all three models starting from default hyperparameter initializations. For the periodic kernel case, we consider an extrapolation task. The results are presented in Fig 10. We observe that all three methods are capable of performing the extrapolation task using the periodic kernel, and in general depict qualitatively similar results. This verifies that–in very simple cases–much of the model performance can depend on the choice of kernel rather than the particular training scheme.

## E. Whitened Sparse Gaussian Process Regression

The hyperparameters and variational parameters of the models can be learned by directly optimizing the objective functions in Eqn. 3 (for MAP), in Eqn. 5 (for SVGP), in Eqn. 15 (for VFITC), and Eqn. 17 (for PPGPR). In practice, we modify these objective functions using a transformation proposed by (Matthews, 2017). The "whitening transformation" is a simple change of variables

$$\mathbf{u}' = \mathbf{\Lambda}_{MM}^{-1}\mathbf{u}$$

where $\mathbf{\Lambda}_{MM}$ is a matrix such that $\mathbf{\Lambda}_{MM}\mathbf{\Lambda}_{MM}^{\top} = \mathbf{K}_{MM}$. (Typically, $\mathbf{\Lambda}_{MM}$ is taken to be the Cholesky factor of $\mathbf{K}_{MM}$.) Intuitively, this transformation is advantageous because it reduces the number of changing terms in the objective functions. In whitened coordinates the prior $p(\mathbf{u}')$ is constant: $p(\mathbf{u}') = \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_{MM}^{-1}\mathbf{K}_{MM}\mathbf{\Lambda}_{MM}^{-\top}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$.

[14]`https://github.com/maziarraissi/ParametricGP`

[15]See line 177 in `parametric_GP.py`.

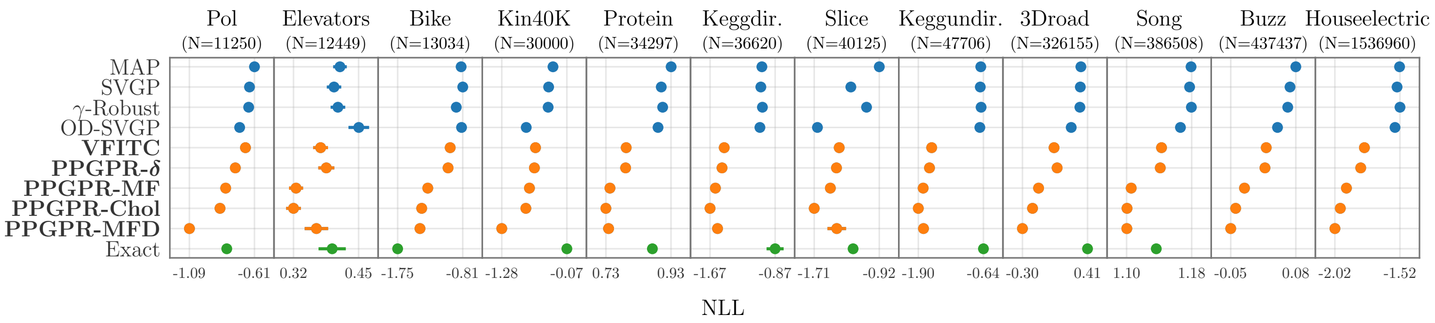

*Figure 6.* We compare test NLLs for the various methods explored in Sec. 5.1-5.2 in the main text (lower is better). Results are averaged over ten random train/test/validation splits. Here and throughout error bars depict standard errors.

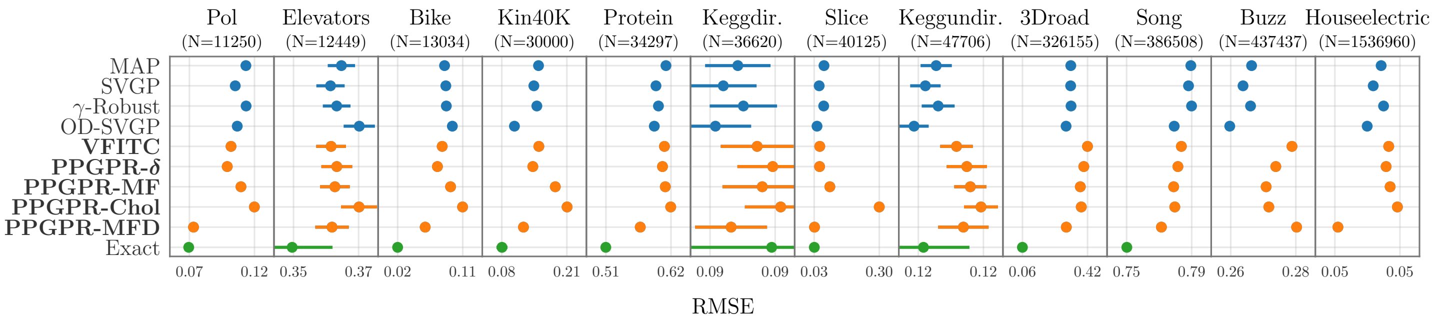

*Figure 7.* We compare test RMSEs for the various methods explored in Sec. 5.1-5.2 in the main text (lower is better). Results are averaged over ten random train/test/validation splits.

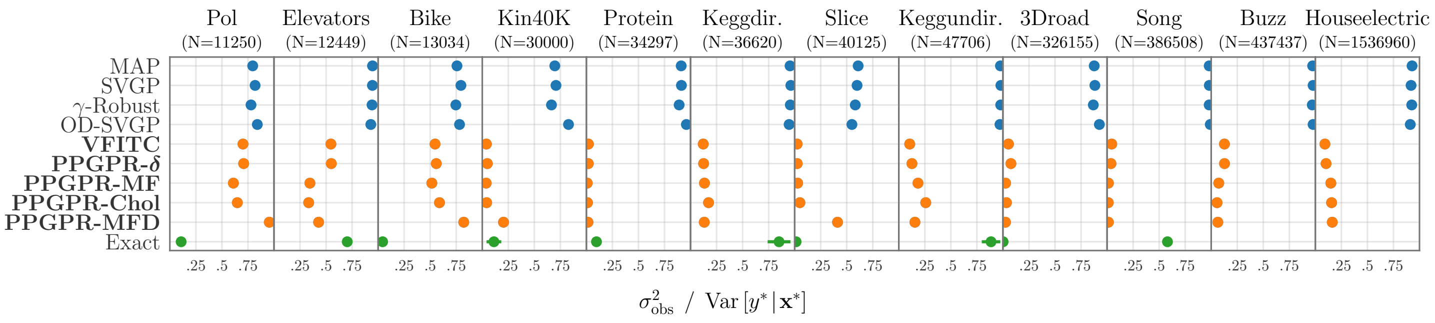

*Figure 8.* We compare the mean fraction of the predictive variance that is due to the observation noise for the various methods explored in Sec. 5.1-5.2 in the main text. Results are averaged over ten random train/test/validation splits.

Incorporating this transformation into Eqn. 3 gives us

$$\log p(\mathbf{y}, \mathbf{u}' | \mathbf{X}, \mathbf{Z}) \geq \mathbb{E}_{p(\mathbf{f}|\mathbf{u}')} \left[ \log p(\mathbf{y}|\mathbf{f}) + \log p(\mathbf{u}') \right]$$
$$= \sum_{i=1}^{n} \log \mathcal{N}(y_i | \mathbf{k}_i^T \boldsymbol{\Lambda}_{MM}^{-1} \mathbf{u}', \sigma_{\text{obs}}) - \tfrac{1}{2\sigma_{\text{obs}}} \text{Tr } \tilde{\mathbf{K}}_{NN}$$
$$- \tfrac{1}{2} \|\mathbf{u}'\|_2^2 - \tfrac{M}{2} \log(2\pi).$$

Importantly, the prior term $p(\mathbf{u}')$ in the modified objective does not depend on the inducing point locations $\mathbf{Z}$ or kernel hyperparameters. In all experiments we use similarly modified objectives for better optimization. For MAP and PPGPR-$\delta$, we directly optimize the whitened variables $\mathbf{u}'$. For SVGP, VFITC, PPGPR-Chol, and PPGPR-MF, the whitened variational distribution is given as $q(\mathbf{u}') = \mathcal{N}(\boldsymbol{\Lambda}_{MM}^{-1} \boldsymbol{m}, \boldsymbol{\Lambda}_{MM}^{-1} \mathbf{S} \boldsymbol{\Lambda}_{MM}^{-1})$. We parameterize the mean with a vector $\boldsymbol{m}' = \boldsymbol{\Lambda}_{MM}^{-1} \boldsymbol{m}$ and we parameterize the covariance with a lower triangular matrix $\mathbf{L}\mathbf{L}^\top = \boldsymbol{\Lambda}_{MM}^{-1} \mathbf{S} \boldsymbol{\Lambda}_{MM}^{-1}$. The same transformation is applied to the OD-SVGP covariance variational parameters—referred to as the $\beta$ parameters by Salimbeni et al. (2018).

*Table 3.* A compilation of all the results from Sec. 5.1-5.2. For each metric and dataset we bold the result for the best performing scalable method. Errors are standard errors.

| Metric | Dataset | MAP | SVGP | $\gamma$-Robust | OD-SVGP | VFITC | PPGPR-$\delta$ | PPGPR-MF | PPGPR-Chol | PPGPR-MFD | Exact |
|---|---|---|---|---|---|---|---|---|---|---|---|
| NLL | Pol | $-0.615 \pm 0.007$ | $-0.651 \pm 0.005$ | $-0.657 \pm 0.007$ | $-0.723 \pm 0.006$ | $-0.681 \pm 0.005$ | $-0.755 \pm 0.009$ | $-0.825 \pm 0.005$ | $-0.866 \pm 0.005$ | $-\mathbf{1.090} \pm \mathbf{0.009}$ | $-0.817 \pm 0.001$ |
| | Elevators | $0.412 \pm 0.007$ | $0.401 \pm 0.007$ | $0.409 \pm 0.007$ | $0.448 \pm 0.010$ | $0.376 \pm 0.007$ | $0.386 \pm 0.008$ | $\mathbf{0.329} \pm \mathbf{0.007}$ | $\mathbf{0.324} \pm \mathbf{0.007}$ | $0.368 \pm 0.011$ | $0.398 \pm 0.013$ |
| | Bike | $-0.830 \pm 0.010$ | $-0.807 \pm 0.007$ | $-0.901 \pm 0.013$ | $-0.824 \pm 0.009$ | $-0.989 \pm 0.009$ | $-1.019 \pm 0.008$ | $-1.314 \pm 0.007$ | $-1.402 \pm 0.004$ | $-\mathbf{1.426} \pm \mathbf{0.010}$ | $-1.750 \pm 0.014$ |
| | Kin40K | $-0.333 \pm 0.002$ | $-0.414 \pm 0.002$ | $-0.423 \pm 0.002$ | $-0.830 \pm 0.004$ | $-0.658 \pm 0.001$ | $-0.678 \pm 0.002$ | $-0.770 \pm 0.002$ | $-0.837 \pm 0.002$ | $-\mathbf{1.284} \pm \mathbf{0.005}$ | $-0.075 \pm 0.001$ |
| | Protein | $0.931 \pm 0.003$ | $0.902 \pm 0.003$ | $0.906 \pm 0.004$ | $0.892 \pm 0.006$ | $0.796 \pm 0.004$ | $0.794 \pm 0.004$ | $\mathbf{0.747} \pm \mathbf{0.008}$ | $\mathbf{0.735} \pm \mathbf{0.007}$ | $\mathbf{0.743} \pm \mathbf{0.008}$ | $0.875 \pm 0.002$ |
| | Keggdir. | $-1.032 \pm 0.017$ | $-1.045 \pm 0.017$ | $-1.026 \pm 0.021$ | $-1.057 \pm 0.018$ | $-1.494 \pm 0.014$ | $-1.520 \pm 0.013$ | $-1.601 \pm 0.016$ | $-\mathbf{1.667} \pm \mathbf{0.028}$ | $-1.575 \pm 0.015$ | $-0.870 \pm 0.052$ |
| | Slice | $-0.921 \pm 0.002$ | $-1.267 \pm 0.003$ | $-1.076 \pm 0.006$ | $-1.673 \pm 0.013$ | $-1.409 \pm 0.004$ | $-1.442 \pm 0.004$ | $-1.516 \pm 0.004$ | $-\mathbf{1.713} \pm \mathbf{0.004}$ | $-1.438 \pm 0.057$ | $-1.240 \pm 0.004$ |
| | Keggundir. | $-0.694 \pm 0.006$ | $-0.704 \pm 0.006$ | $-0.688 \pm 0.007$ | $-0.712 \pm 0.006$ | $-1.643 \pm 0.016$ | $-1.685 \pm 0.018$ | $-1.807 \pm 0.018$ | $-\mathbf{1.901} \pm \mathbf{0.021}$ | $-1.801 \pm 0.013$ | $-0.643 \pm 0.008$ |
| | 3Droad | $0.336 \pm 0.002$ | $0.330 \pm 0.002$ | $0.325 \pm 0.001$ | $0.231 \pm 0.014$ | $0.045 \pm 0.003$ | $0.079 \pm 0.004$ | $-0.122 \pm 0.002$ | $-0.188 \pm 0.002$ | $-\mathbf{0.297} \pm \mathbf{0.003}$ | $0.408$ |
| | Song | $1.180 \pm 0.001$ | $1.178 \pm 0.001$ | $1.181 \pm 0.001$ | $1.168 \pm 0.001$ | $1.145 \pm 0.001$ | $1.143 \pm 0.001$ | $1.109 \pm 0.001$ | $\mathbf{1.104} \pm \mathbf{0.001}$ | $\mathbf{1.103} \pm \mathbf{0.001}$ | $1.139$ |
| | Buzz | $0.080 \pm 0.002$ | $0.069 \pm 0.001$ | $0.064 \pm 0.001$ | $0.044 \pm 0.002$ | $0.022 \pm 0.002$ | $0.020 \pm 0.002$ | $-0.020 \pm 0.001$ | $-0.037 \pm 0.001$ | $-\mathbf{0.047} \pm \mathbf{0.001}$ | — |
| | Houseelectric | $-1.524 \pm 0.001$ | $-1.543 \pm 0.001$ | $-1.521 \pm 0.001$ | $-1.559 \pm 0.002$ | $-1.794 \pm 0.001$ | $-1.822 \pm 0.001$ | $-1.931 \pm 0.001$ | $-1.978 \pm 0.001$ | $-\mathbf{2.020} \pm \mathbf{0.003}$ | — |
| RMSE | Pol | $0.115 \pm 0.001$ | $0.107 \pm 0.002$ | $0.115 \pm 0.002$ | $0.109 \pm 0.001$ | $0.104 \pm 0.002$ | $0.101 \pm 0.001$ | $0.111 \pm 0.002$ | $0.121 \pm 0.002$ | $\mathbf{0.077} \pm \mathbf{0.001}$ | $0.074 \pm 0.001$ |
| | Elevators | $\mathbf{0.364} \pm \mathbf{0.002}$ | $\mathbf{0.360} \pm \mathbf{0.003}$ | $\mathbf{0.362} \pm \mathbf{0.002}$ | $0.370 \pm 0.003$ | $\mathbf{0.360} \pm \mathbf{0.003}$ | $\mathbf{0.362} \pm \mathbf{0.003}$ | $\mathbf{0.362} \pm \mathbf{0.003}$ | $0.370 \pm 0.003$ | $\mathbf{0.361} \pm \mathbf{0.003}$ | $0.347 \pm 0.007$ |
| | Bike | $0.086 \pm 0.002$ | $0.088 \pm 0.002$ | $0.089 \pm 0.002$ | $0.097 \pm 0.002$ | $0.083 \pm 0.002$ | $0.076 \pm 0.001$ | $0.094 \pm 0.002$ | $0.111 \pm 0.002$ | $\mathbf{0.060} \pm \mathbf{0.001}$ | $0.022 \pm 0.001$ |
| | Kin40K | $0.156 \pm 0.001$ | $0.147 \pm 0.001$ | $0.152 \pm 0.001$ | $\mathbf{0.109} \pm \mathbf{0.001}$ | $0.156 \pm 0.001$ | $0.145 \pm 0.001$ | $0.188 \pm 0.002$ | $0.211 \pm 0.003$ | $0.126 \pm 0.001$ | $0.084 \pm 0.005$ |
| | Protein | $0.610 \pm 0.002$ | $0.594 \pm 0.002$ | $0.598 \pm 0.002$ | $0.591 \pm 0.003$ | $0.607 \pm 0.002$ | $0.604 \pm 0.002$ | $0.609 \pm 0.002$ | $0.618 \pm 0.002$ | $\mathbf{0.569} \pm \mathbf{0.002}$ | $0.514 \pm 0.003$ |
| | Keggdir. | $\mathbf{0.087} \pm \mathbf{0.001}$ | $\mathbf{0.086} \pm \mathbf{0.001}$ | $\mathbf{0.088} \pm \mathbf{0.001}$ | $\mathbf{0.085} \pm \mathbf{0.001}$ | $0.089 \pm 0.001$ | $0.090 \pm 0.001$ | $0.089 \pm 0.002$ | $0.090 \pm 0.001$ | $\mathbf{0.087} \pm \mathbf{0.001}$ | $0.090 \pm 0.004$ |
| | Slice | $0.071 \pm 0.001$ | $0.051 \pm 0.001$ | $0.070 \pm 0.003$ | $0.043 \pm 0.001$ | $0.054 \pm 0.001$ | $0.053 \pm 0.001$ | $0.095 \pm 0.001$ | $0.298 \pm 0.004$ | $\mathbf{0.032} \pm \mathbf{0.001}$ | $0.031 \pm 0.003$ |
| | Keggundir. | $0.121 \pm 0.001$ | $\mathbf{0.120} \pm \mathbf{0.001}$ | $0.121 \pm 0.001$ | $\mathbf{0.119} \pm \mathbf{0.001}$ | $0.123 \pm 0.001$ | $0.123 \pm 0.001$ | $0.124 \pm 0.001$ | $0.125 \pm 0.001$ | $0.123 \pm 0.001$ | $0.119 \pm 0.002$ |
| | 3Droad | $0.329 \pm 0.001$ | $0.329 \pm 0.001$ | $0.331 \pm 0.001$ | $\mathbf{0.303} \pm \mathbf{0.004}$ | $0.424 \pm 0.001$ | $0.403 \pm 0.005$ | $0.383 \pm 0.002$ | $0.389 \pm 0.001$ | $\mathbf{0.304} \pm \mathbf{0.001}$ | $0.057$ |
| | Song | $0.788 \pm 0.001$ | $0.786 \pm 0.001$ | $0.788 \pm 0.001$ | $0.778 \pm 0.001$ | $0.782 \pm 0.001$ | $0.780 \pm 0.001$ | $0.777 \pm 0.001$ | $0.778 \pm 0.001$ | $\mathbf{0.770} \pm \mathbf{0.001}$ | $0.750$ |
| | Buzz | $0.265 \pm 0.001$ | $0.261 \pm 0.000$ | $0.264 \pm 0.001$ | $\mathbf{0.256} \pm \mathbf{0.001}$ | $0.281 \pm 0.001$ | $0.275 \pm 0.001$ | $0.271 \pm 0.001$ | $0.272 \pm 0.001$ | $0.283 \pm 0.001$ | — |
| | Houseelectric | $0.052 \pm 0.000$ | $0.051 \pm 0.000$ | $0.052 \pm 0.000$ | $0.050 \pm 0.000$ | $0.053 \pm 0.000$ | $0.052 \pm 0.000$ | $0.053 \pm 0.000$ | $0.054 \pm 0.000$ | $\mathbf{0.046} \pm \mathbf{0.000}$ | — |
| CRPS | Pol | $0.065 \pm 0.001$ | $0.061 \pm 0.000$ | $0.062 \pm 0.001$ | $0.059 \pm 0.000$ | $0.060 \pm 0.000$ | $0.057 \pm 0.001$ | $0.057 \pm 0.000$ | $0.057 \pm 0.000$ | $\mathbf{0.040} \pm \mathbf{0.000}$ | $0.051 \pm 0.000$ |
| | Elevators | $0.200 \pm 0.001$ | $0.198 \pm 0.001$ | $0.199 \pm 0.001$ | $0.203 \pm 0.001$ | $0.197 \pm 0.001$ | $0.198 \pm 0.001$ | $\mathbf{0.193} \pm \mathbf{0.001}$ | $\mathbf{0.195} \pm \mathbf{0.001}$ | $\mathbf{0.195} \pm \mathbf{0.002}$ | $0.195 \pm 0.003$ |
| | Bike | $0.047 \pm 0.000$ | $0.049 \pm 0.000$ | $0.043 \pm 0.000$ | $0.051 \pm 0.001$ | $0.042 \pm 0.000$ | $0.040 \pm 0.000$ | $0.037 \pm 0.000$ | $0.037 \pm 0.001$ | $\mathbf{0.028} \pm \mathbf{0.000}$ | $0.019 \pm 0.000$ |
| | Kin40K | $0.088 \pm 0.000$ | $0.082 \pm 0.000$ | $0.082 \pm 0.000$ | $0.056 \pm 0.000$ | $0.074 \pm 0.000$ | $0.071 \pm 0.000$ | $0.077 \pm 0.000$ | $0.079 \pm 0.000$ | $\mathbf{0.050} \pm \mathbf{0.000}$ | $0.093 \pm 0.000$ |
| | Protein | $0.337 \pm 0.001$ | $0.326 \pm 0.001$ | $0.326 \pm 0.001$ | $0.317 \pm 0.001$ | $0.318 \pm 0.001$ | $0.317 \pm 0.001$ | $0.310 \pm 0.001$ | $0.310 \pm 0.001$ | $\mathbf{0.288} \pm \mathbf{0.001}$ | $0.293 \pm 0.001$ |
| | Keggdir. | $0.038 \pm 0.000$ | $0.037 \pm 0.000$ | $0.036 \pm 0.000$ | $0.037 \pm 0.000$ | $0.033 \pm 0.000$ | $0.032 \pm 0.000$ | $0.031 \pm 0.000$ | $\mathbf{0.031} \pm \mathbf{0.000}$ | $0.031 \pm 0.000$ | $0.046 \pm 0.002$ |
| | Slice | $0.044 \pm 0.000$ | $0.031 \pm 0.000$ | $0.037 \pm 0.000$ | $0.021 \pm 0.000$ | $0.029 \pm 0.000$ | $0.028 \pm 0.000$ | $0.032 \pm 0.000$ | $0.060 \pm 0.001$ | $\mathbf{0.014} \pm \mathbf{0.000}$ | $0.029 \pm 0.000$ |
| | Keggundir. | $0.052 \pm 0.000$ | $0.051 \pm 0.000$ | $0.050 \pm 0.000$ | $0.051 \pm 0.000$ | $0.038 \pm 0.000$ | $0.037 \pm 0.000$ | $0.036 \pm 0.000$ | $\mathbf{0.035} \pm \mathbf{0.000}$ | $0.036 \pm 0.000$ | $0.056 \pm 0.001$ |
| | 3Droad | $0.178 \pm 0.000$ | $0.177 \pm 0.000$ | $0.175 \pm 0.000$ | $0.163 \pm 0.002$ | $0.191 \pm 0.001$ | $0.185 \pm 0.002$ | $0.170 \pm 0.001$ | $0.167 \pm 0.000$ | $\mathbf{0.138} \pm \mathbf{0.000}$ | $0.142$ |
| | Song | $0.441 \pm 0.000$ | $0.440 \pm 0.000$ | $0.441 \pm 0.000$ | $0.434 \pm 0.000$ | $0.434 \pm 0.000$ | $0.433 \pm 0.000$ | $0.426 \pm 0.000$ | $0.425 \pm 0.000$ | $\mathbf{0.422} \pm \mathbf{0.000}$ | $0.420$ |
| | Buzz | $0.141 \pm 0.000$ | $0.140 \pm 0.000$ | $0.139 \pm 0.000$ | $0.135 \pm 0.000$ | $0.141 \pm 0.000$ | $0.140 \pm 0.000$ | $0.136 \pm 0.000$ | $0.135 \pm 0.000$ | $\mathbf{0.134} \pm \mathbf{0.000}$ | — |
| | Houseelectric | $0.027 \pm 0.000$ | $0.027 \pm 0.000$ | $0.027 \pm 0.000$ | $0.026 \pm 0.000$ | $0.025 \pm 0.000$ | $0.025 \pm 0.000$ | $0.024 \pm 0.000$ | $0.024 \pm 0.000$ | $\mathbf{0.022} \pm \mathbf{0.000}$ | — |

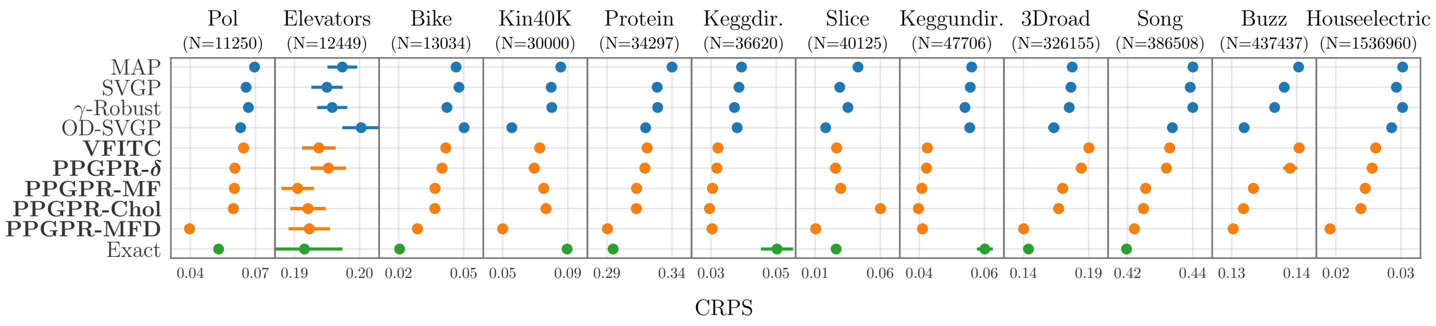

*Figure 9.* We compare test continuous ranked probability scores (CRPS) for the various methods explored in Sec. 5.1-5.2 (lower is better). Results are averaged over ten random train/test/validation splits.

As in the original work, the mean variational parameters (the $\gamma$ parameters) are not re-parameterized.

We apply a similar whitening transformation to the decoupled variant of our method (PGPR-MFD). Given inducing points $\mathbf{Z}_\mu$ and $\mathbf{Z}_\sigma$ for the mean and variance functions respectively, we optimize the transformed parameters

$$\boldsymbol{m}' = \boldsymbol{\Lambda}^{(\mu)-1}_{MM}\boldsymbol{m}, \quad \mathbf{S}' = \boldsymbol{\Lambda}^{(\sigma)-1}_{MM}\mathbf{S}\boldsymbol{\Lambda}^{(\sigma)-\top}_{MM},$$

where $\boldsymbol{\Lambda}^{(\mu)}_{MM}$ and $\boldsymbol{\Lambda}^{(\sigma)}_{MM}$ are the Cholesky factor of the $\mathbf{Z}_\mu$ and $\mathbf{Z}_\sigma$ kernel matrices. $\mathbf{S}'$ is constrained to be a diagonal matrix because of the mean-field approximation.

## F. Connection to Direct Loss Minimization

In a series of papers the authors of (Sheth and Khardon, 2016; 2017; 2019) consider approximate inference from the angle of regularized loss minimization. For Bayesian models of the general form

$$p(\mathbf{y}, \mathbf{u}) = p(\mathbf{u}) \prod_{i=1}^{N} p(y_i|\mathbf{u}, \mathbf{x}_i) \tag{26}$$

and (approximate posterior) distributions $q(\mathbf{u})$ Sheth and Khardon investigate PAC-Bayes-like bounds for the Bayes risk

$$r_{\text{Bayes}}[q(\mathbf{u})] \equiv \mathbb{E}_{p_{\text{data}}(\mathbf{x},y)}\left[-\log \mathbb{E}_{q(\mathbf{u})} p(y|\mathbf{u},\mathbf{x})\right] \tag{27}$$

and the Gibbs risk

$$r_{\text{Gibbs}}[q(\mathbf{u})] \equiv \mathbb{E}_{p_{\text{data}}(\mathbf{x},y)}\mathbb{E}_{q(\mathbf{u})}\left[-\log p(y|\mathbf{u},\mathbf{x})\right] \tag{28}$$

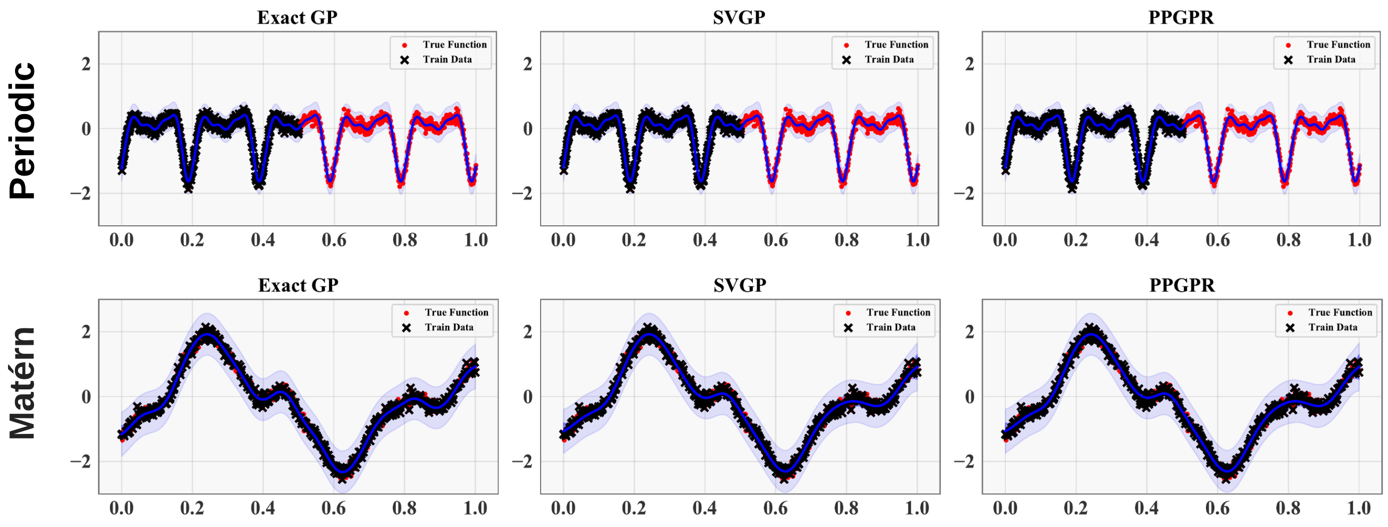

*Figure 10.* We compare exact GPs, SVGP, and PPGPR from functions drawn from a Gaussian process prior using a periodic kernel (top) and Matern kernel (bottom).

In particular, under some restrictions on the family of multivariate Normal distributions $q(\mathbf{u})$, they establish a bound on a smoothed log loss variant of Bayes risk for an inference setup that corresponds to Variational FITC (Sec. 3.1).[16] Additionally, under similar assumptions an analogous bound for an inference setup that corresponds to PPGPR-Chol is established in (Sheth and Khardon, 2019). Moreover, in (Sheth and Khardon, 2016) the authors provide some empirical evidence for the good performance of these two objective functions.

Importantly, our parametric modeling perspective allows us more flexibility than is allowed by Direct Loss Minimization as explored by Sheth and Khardon. Indeed much of the predictive power of approximate GP regressors arguably comes from the sensible behavior of the parametric family of mean and variance functions that is induced by a particular kernel and associated inducing point locations $\mathbf{Z}$—for example, the variance $\sigma_{\mathbf{f}}(\mathbf{x})^2$ increases as $\mathbf{x}$ moves away from $\mathbf{Z}$. In particular the maximum likelihood approach we adopt in Sec. 3.2 allows us to consider parametric families of predictive distributions that are *not* of the form $\mathbb{E}_{q(\mathbf{u})}\mathbb{E}_{p(f|\mathbf{u},\mathbf{x})}\left[p(y|f)\right]$ for some distribution $q(\mathbf{u})$. This is in fact the case for our best performing method, PPGPR-MFD, which freely combines mean and variance functions that are parameterized by distinct inducing point locations $\mathbf{Z}_\mu$ and $\mathbf{Z}_\sigma$. Indeed, while we have not done so in our experiments, this perspective allows us to entirely decouple $\mu_{\mathbf{f}}(\mathbf{x})$ and $\sigma_{\mathbf{f}}(\mathbf{x})^2$ by introducing separate kernels for each.

[16]See corollary 12 in (Sheth and Khardon, 2017).

## G. Connection to stochastic EP and the BB-$\alpha$ objective

Suppose we want to approximate the distribution

$$p(\omega) = \frac{1}{Z} p_0(\omega) \prod_{i=1}^{N} f_n(\omega) \tag{29}$$

where $Z$ is an unknown normalizer. In the prototypical context of Bayesian modeling $p_0(\omega)$ would be a prior distribution and $f_n(\omega)$ would be a likelihood factor for the $n^{\text{th}}$ datapoint. Expectation propagation (EP) is a broad class of algorithms that can be used to approximate distributions like that in Eqn. 29 (Minka, 2004). In the following we give a brief review of a few variants of EP and describe a connection to the Predictive objective defined in Eqn. 17 in the main text.

Li et al. (2015) propose a particular variant of EP called Stochastic EP that reduces memory requirements by a factor of $N$ by tying (i.e. sharing) factors together. In subsequent work Hernández-Lobato et al. (2016) present a version of Stochastic EP that is formulated in terms of an energy function, the so-called BB-$\alpha$ objective $\mathcal{L}_\alpha$, which is given by

$$\mathcal{L}_\alpha = -\frac{1}{\alpha} \sum_{i=1}^{N} \log \mathbb{E}_{q(\omega)} \left[ \left( \frac{f_n(\omega) p_0(\omega)^{1/N}}{q(\omega)^{1/N}} \right)^\alpha \right] \tag{30}$$

where $q(\omega)$ is a so-called cavity distribution and $\alpha \in [0,1]$. As shown in (Li and Gal, 2017), this can be rewritten as

$$\mathcal{L}_\alpha = R_\xi(\tilde{q}||p_0) - \tfrac{1}{\alpha} \sum_{i=1}^{N} \log \mathbb{E}_{\tilde{q}} \left[ f_n(\omega)^\alpha \right] \tag{31}$$

where $\tilde{q}(\omega)$ is defined by the equation

$$q(\omega) = \frac{1}{Z_q} \tilde{q}(\omega) \left( \frac{\tilde{q}(\omega)}{p_0(\omega)} \right)^{\frac{\alpha}{N-\alpha}} \tag{32}$$

and where $\xi \equiv \frac{N}{N-\alpha}$ and $R_\xi(\tilde{q}||p_0)$ is a Rényi divergence. Li and Gal (2017) then argue that, under suitable conditions, we have that as $\frac{\alpha}{N} \to 0$ this becomes

$$\mathcal{L}_\alpha \to \mathrm{KL}(q||p_0) - \tfrac{1}{\alpha} \sum_{i=1}^{N} \log \mathbb{E}_q \left[ f_n(\omega)^\alpha \right] \tag{33}$$

For the particular choice $\alpha = 1$ (so that we require $N \to \infty$) this then becomes

$$\mathcal{L}_{\alpha=1} \to \mathrm{KL}(q||p_0) - \sum_{i=1}^{N} \log \mathbb{E}_q \left[ f_n(\omega) \right] \tag{34}$$

The similarity of Eqn. 34 and Eqn. 17 is now manifest.

For further discussion of EP methods in the context of Gaussian Process inference we refer the reader to (Bui et al., 2017).