RFF approx GP
Wenying Deng
4/24/2020

Last time we validated the ability of Random Fourier Feature in approximating Gaussian Process, specifically using RBF kernel (with length-scale 0.4 and variance 1). I computed the posterior covariance using the whole sample at once, since there is closed form relationship between the kernel and random Fourier Feature by Equation (2) from the RFF original paper [1]:

$$k(\mathbf{x}-\mathbf{y}) = \int_R^d p(\omega)e^{j\omega'(\mathbf{x}-\mathbf{y})}d\omega = E_\omega[\zeta_\omega(\mathbf{x})\zeta_\omega(\mathbf{x})^*].$$

These two weeks I tried to use Woodbury identity formula to update the covariance, to reduce time complexity. A GP (say f) under RFF approximation is written as $f = \Phi\beta$, where Φ are the Fourier Features and β is the regression coefficients, in this way we can get GP posterior by performing posterior inference on β, which in our case is the same as performing posterior inference for Bayesian linear regression.

Specifically, the posterior variance of β is $\Sigma_\beta = (\Phi^\top\Phi + \sigma^2\mathbf{I})^{-1}$. Given a mini-batch of Φ, we can update the covariance matrix using Woodbury identity formula:

$$\Sigma_{\beta,t+1} = \Sigma_{\beta,t} - \Sigma_{\beta,t}\Phi^\top(\mathbf{I} + \Phi\Sigma_{\beta,t}\Phi^\top)^{-1}\Phi\Sigma_{\beta,t}.$$

After we get the posterior variance Σ_β, the GP posterior for prediction mean $f_{new} = \Phi_{new}\beta$ is $\Phi_{new}\Sigma_\beta\Phi_{new}^\top$.

In this way we can reduce time complexity from $O(N^3)$ to $O(N)$, where N is sample size.

On the other hand, [2] proposed Orthogonal Random Features (ORF), which can significantly reduced the kernel approximation error. The idea of ORF is to impose orthogonality on the approximate kernel using RFF. Jeremiah also implemented ORF in the original RFF and we see that it indeed has better performance.

Figure 1 shows four metrics comparing GP, RFF and ORF, x-axes are training sample sizes. Dashed lines represent Population variance. The first two plots are small sample size (40-200), while the last two are large (200-1000). $\mathtt{rff\_dim} = 5000$ in the first and third subplots and $\mathtt{rff\_dim} = 10000$ in the second and fourth. For each sample size, I repeated for 10 times and take the average.

Figure 2 shows the random sample plots ($\mathtt{rff\_dim} = 5000$) of posterior predictive distribution: Black points are training samples, with blue curve the posterior mean prediction and the shaded area the 95% CI. ORF has similar performance with RFF, so here I compare ORF to GP in sample size 40, and compare RFF to GP in sample size 1000, with Woodbury and Population variances respectively.

Note that in terms of test log-likelihood and RMSE, GP, RFF-Population and ORF-Population always is slightly better than RFF-Woodbury and ORF-Woodbury, but they are similar. On the other hand, MPIW (mean prediction interval width) are almost the same across GP, RFF and ORF, meaning that we have correct implementation for the variance estimation. Their difference focuses on PICP (prediction interval coverage). Therefore it's possible that RFF-Woodbury and

ORF-Woodbury are not as good as GP, RFF-Population and ORF-Population for mean prediction. It makes sense because we used mini-batch update for Woodbury, and there's optimization error.

In Figure 2, I choose several random sample plots of posterior predictive distribution to see if that's the case. We see that RFF-Woodbury/ORF-Woodbury indeed cannot approximate GP well enough in terms of mean prediction, but RFF-Population/ORF-Population can.

Judging from Figure 1, I reduced $\mathtt{rff\_dim}$ to be $\sqrt{n} * \log(n) * 8$ and 1200 in Figure 3 [3]. Also, I added L2 penalty to MSE, so what we were minimizing is $(\mathbf{y} - \Phi\boldsymbol{\beta})^2 + \sigma^2|\boldsymbol{\beta}|^2$.

Figure 3 shows four metrics comparing GP, RFF and ORF, x-axes are training sample sizes. Dashed lines represent Population variance. The first two plots are small sample size (40-200), while the last two are large (200-1000). $\mathtt{rff\_dim} = \sqrt{n} * \log(n) * 8$ (the random Fourier features dimension are $(186, 253, 313, 367, 419, 467, 513, 556, 598)$ and $(598, 789, 958, 1111, 1253, 1386, 1512, 1632, 1747)$ respectively) in the first and third subplots and $\mathtt{rff\_dim} = 1200$ in the second and fourth. For each sample size, I repeated for 10 times and take the average.

Figure 4 shows the random sample plots ($\mathtt{rff\_dim} = \sqrt{n} * \log(n) * 8$, except for the third row) of posterior predictive distribution: Black points are training samples, with blue curve the posterior mean prediction and the shaded area the 95% CI. ORF has similar performance with RFF, so here I compare ORF to GP in sample size 40, and compare RFF to GP in sample size 1000, with Woodbury and Population variances respectively.

We see that after adding L2 penalty, PICP performs better for small sample size, but not for $n = 40$. ORF-Woodbury cannot fit $n = 40$ well even $\mathtt{rff\_dim} = 1200$ (the second and the third row of Figure 4). And if we compare the second row to the first/fourth row, Woodbury is still underfit.

On the other hand, $\mathtt{rff\_dim}$ doesn't seem to play an important role here. Additionally, there's no obvious difference between ORF and RFF.

Finally, since $\beta = (\Phi^\top\Phi + \sigma^2\mathbf{I})^{-1}\Phi^\top\mathbf{y} = \Sigma_\beta\Phi^\top\mathbf{y}$, I also updated $\Phi^\top\mathbf{y}$:

$$(\Phi^\top\mathbf{y})_{new} = (\Phi^\top\mathbf{y})_{prev} + \Phi_{batch}^\top\mathbf{y}_{batch},$$

to get $\hat{\beta}$. Then we can make prediction with $\hat{\beta}$ and the random Fourier Features, termed as "lnr". As noted before, $\mathtt{rff\_dim}$ doesn't seem to play an important role here. Additionally, there's no obvious difference between ORF and RFF. Here I chose RFF, and in Figure 5 and 6, I showed the cases when $\mathtt{rff\_dim} = 1200$.

Figure 5 shows four metrics comparing GP, LNR-Woodbury and LNR-Population, x-axes are training sample sizes. Dashed lines represent Population variance. The first row are small sample size (40-200), while the second are large (200-1000). For each sample size, I repeated for 30 times and take the average.

Figure 7 shows the random sample plots: Black points are training samples, with blue curve the posterior mean prediction and the shaded area the 95% CI.

We see that LNR-Woodbury performs similar to LNR-Population, which is exactly what we want. Therefore, we can proceed with LNR-Woodbury method from now on and maybe extend to multiple layers [4] later.

Next I will start working on variable selection using RFF method. And I think the tricky part is to compute derivative.

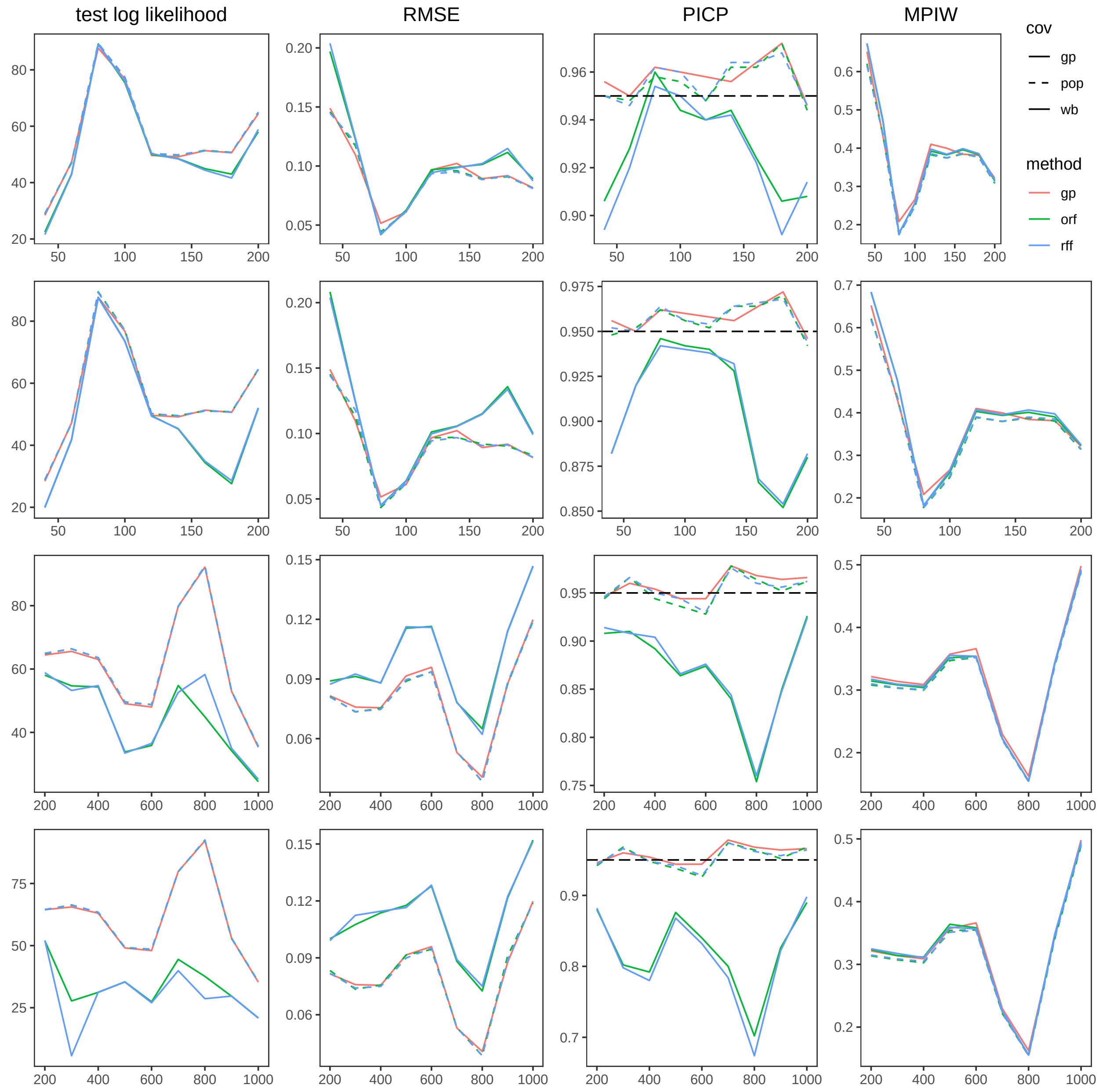

Figure 1: Four metrics comparing GP, RFF and ORF, x-axes are training sample sizes. Dashed lines represent Population variance. The first two plots are small sample size (40-200), while the last two are large (200-1000). `rff_dim` $= 5000$ in the first and third subplots and `rff_dim` $= 10000$ in the second and fourth.

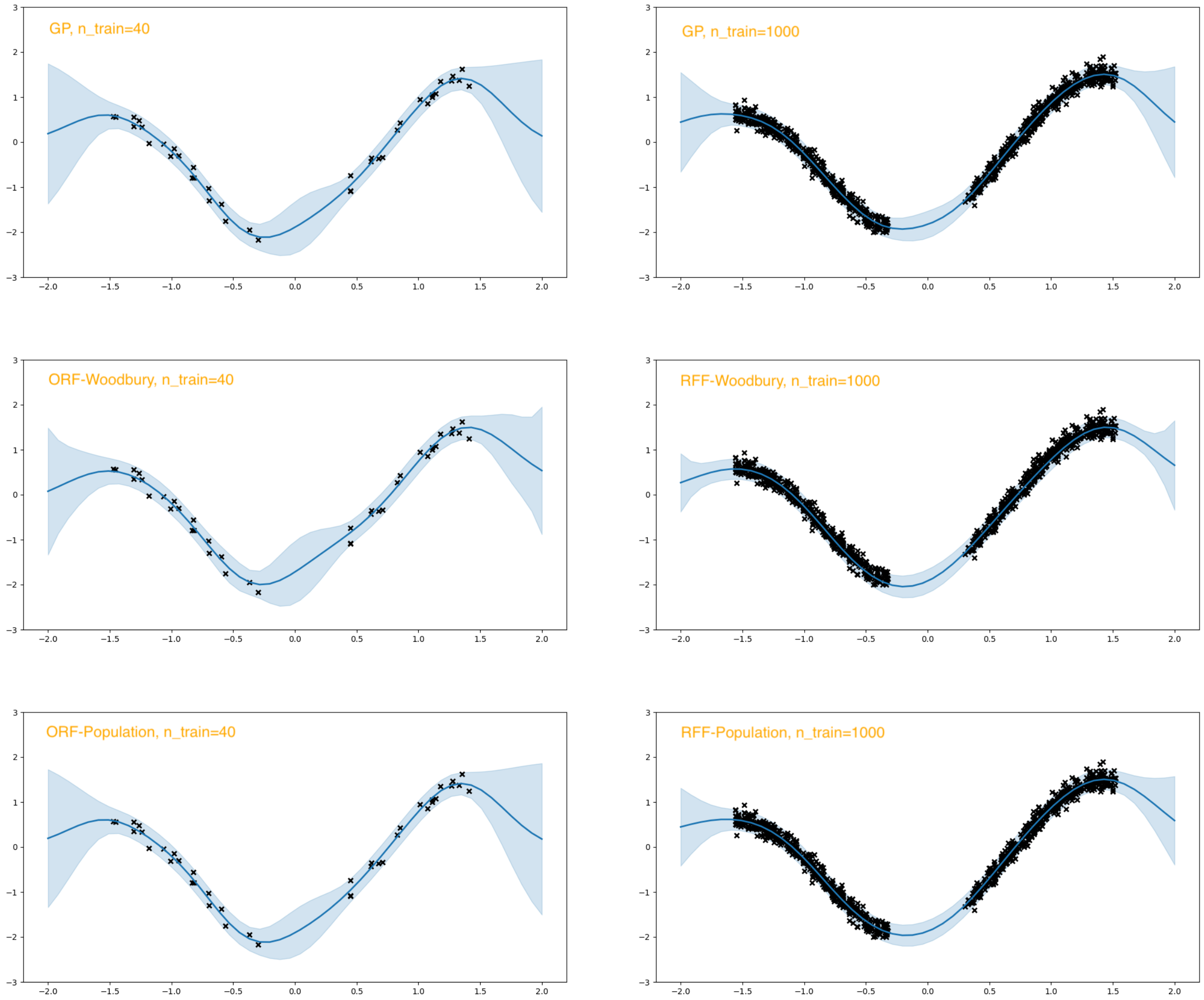

Figure 2: Sample plots of posterior predictive dist: Black points are training samples, with blue curve the posterior mean prediction and the shaded area the 95% CI.

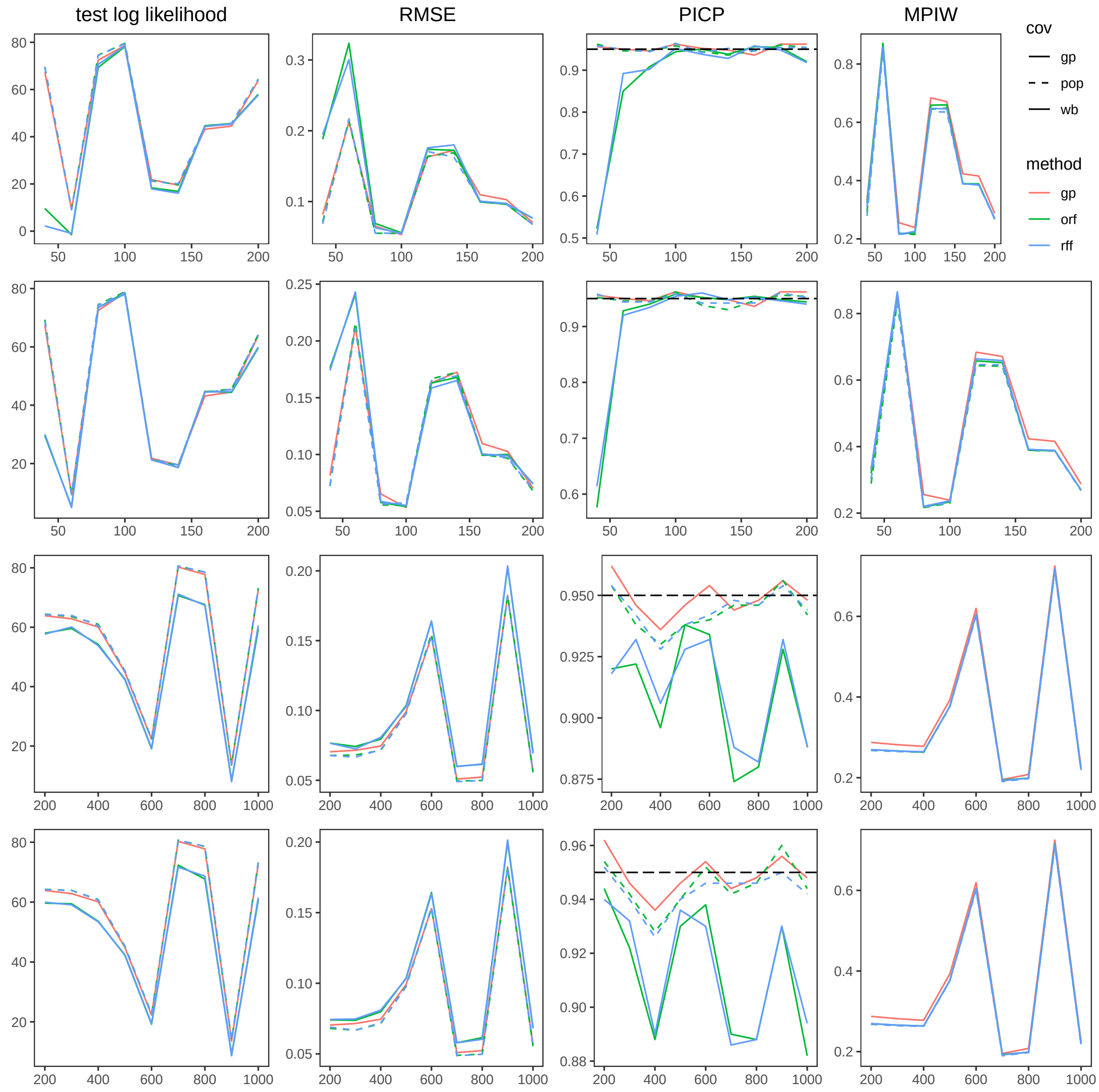

Figure 3: Four metrics comparing GP, RFF and ORF, x-axes are training sample sizes. Dashed lines represent Population variance. The first two plots are small sample size (40-200), while the last two are large (200-1000). $\texttt{rff\_dim} = \sqrt{n} * \log(n) * 8$ in the first and third subplots and $\texttt{rff\_dim} = 1200$ in the second and fourth.

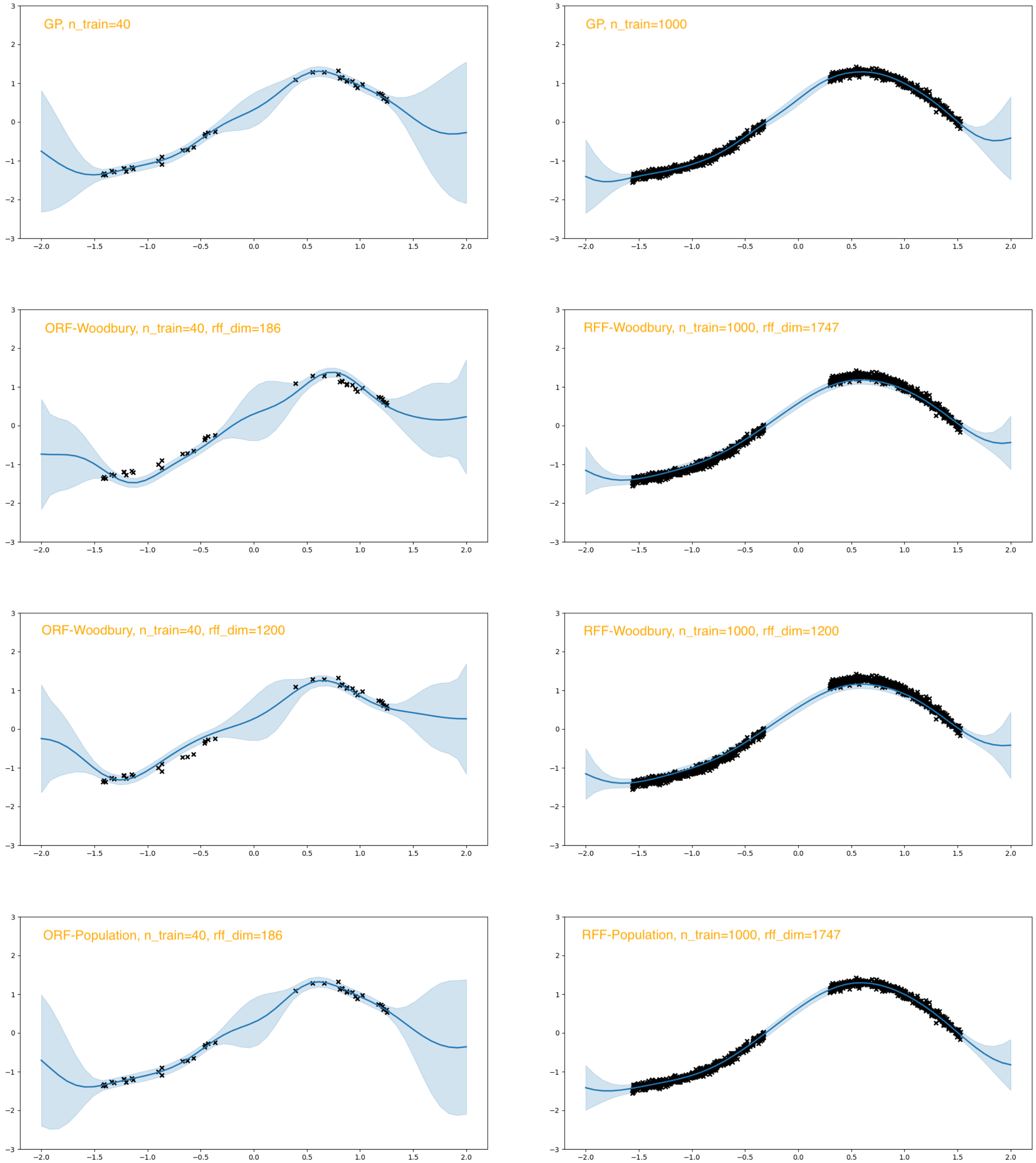

Figure 4: Sample plots of posterior predictive dist: Black points are training samples, with blue curve the posterior mean prediction and the shaded area the 95% CI.

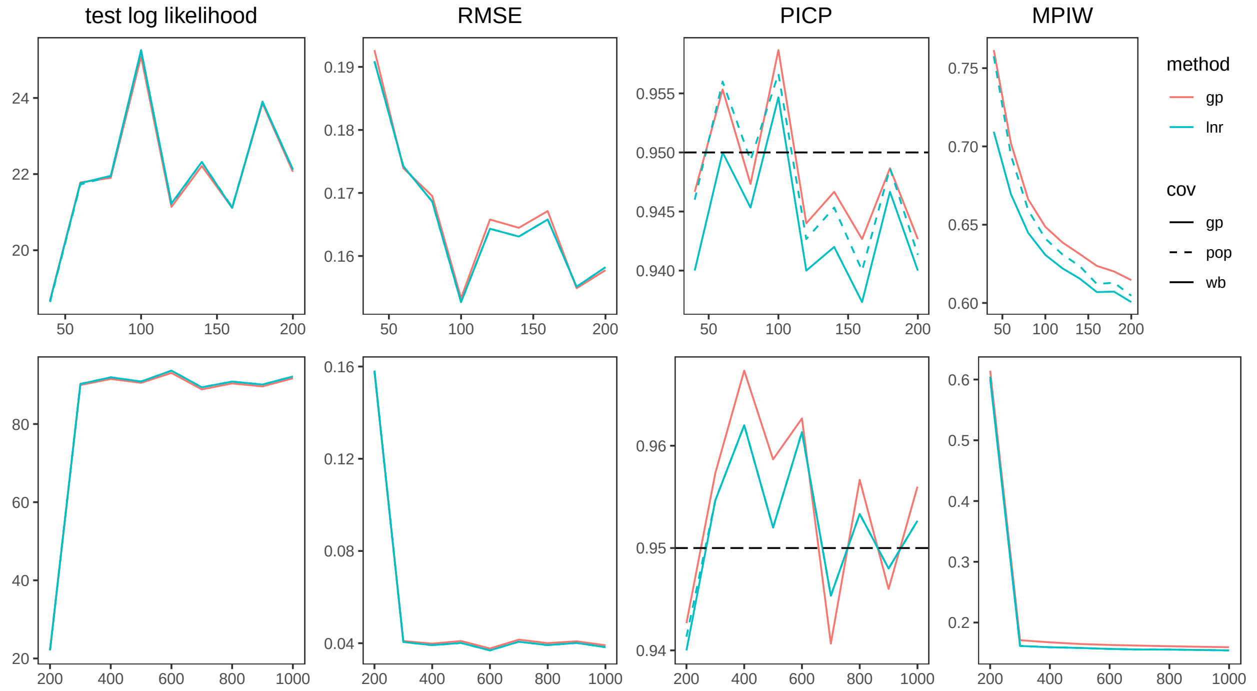

Figure 5: Dashed lines represent Population variance. The first row are small sample size (40-200), while the second are large (200-1000). `rff_dim` $= 1200$.

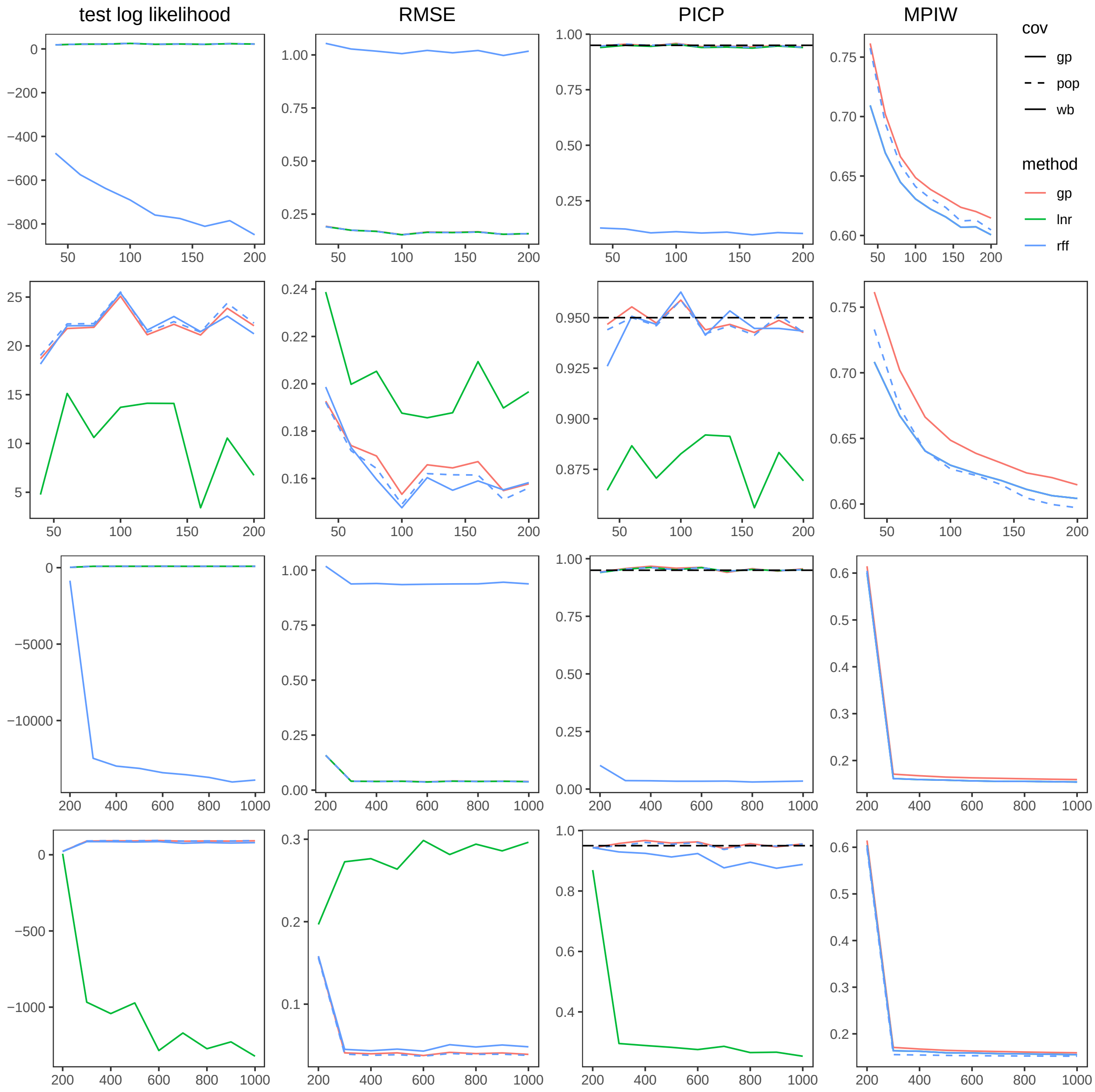

Figure 6: Dashed lines represent Population variance. The first two rows are small sample size (40-200), while the last two are large (200-1000). `rff_dim` $= 1200$. For loop was commented out in the first and third rows and not commented out in the second and fourth.

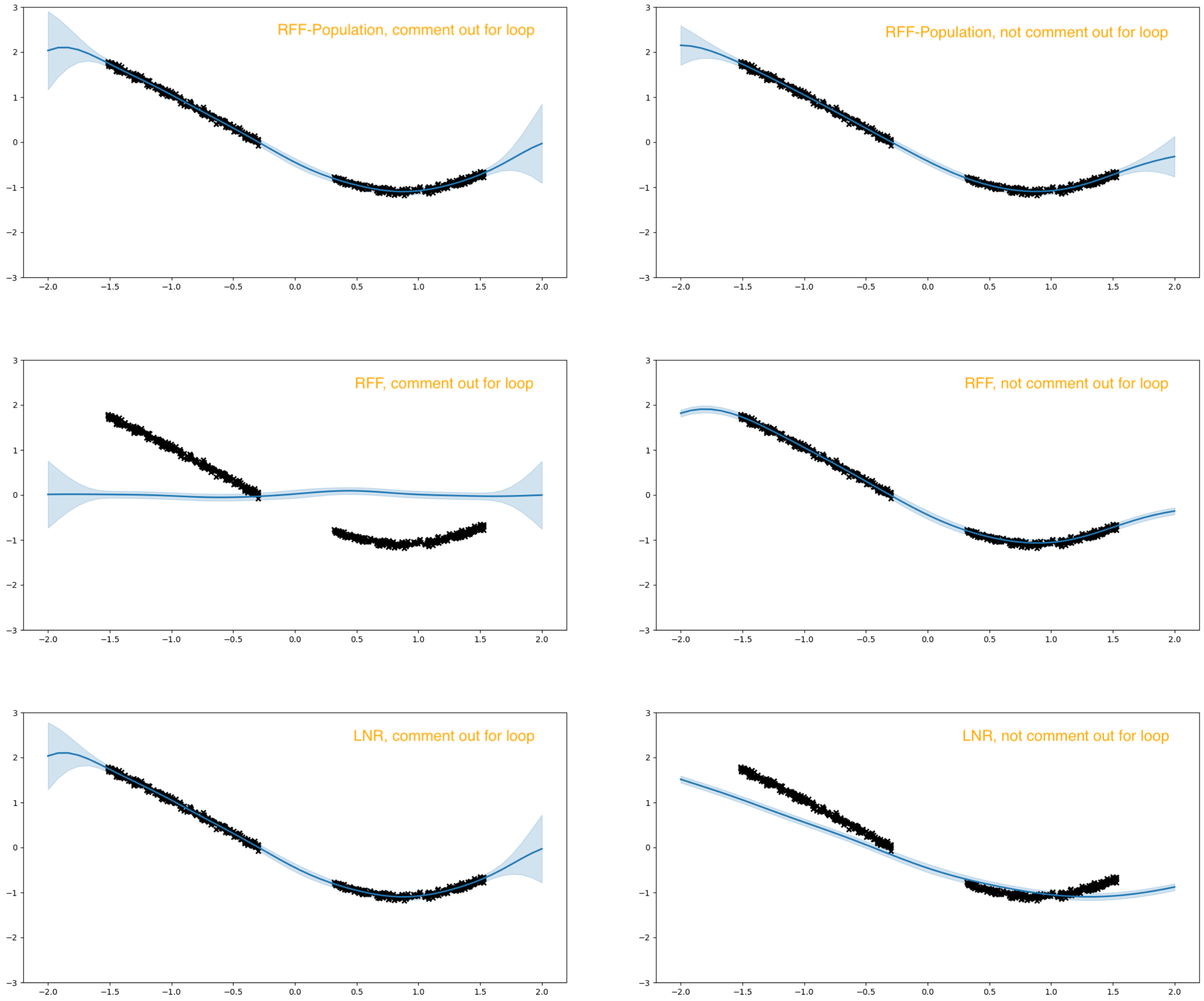

Figure 7: Sample plots of posterior predictive dist: Black points are training samples, with blue curve the posterior mean prediction and the shaded area the 95% CI, $\mathtt{n\_train} = 600$ and repeat for 30 times.

References

[1] Ali Rahimi and Ben Recht. Random Features for Large-Scale Kernel Machines. page 8.

[2] Felix Xinnan X Yu, Ananda Theertha Suresh, Krzysztof M Choromanski, Daniel N Holtmann-Rice, and Sanjiv Kumar. Orthogonal Random Features. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1975–1983. Curran Associates, Inc., 2016.

[3] Alessandro Rudi and Lorenzo Rosasco. Generalization Properties of Learning with Random Features. *arXiv:1602.04474 [cs, stat]*, January 2018. arXiv: 1602.04474.

[4] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. *arXiv:1806.07572 [cs, math, stat]*, February 2020. arXiv: 1806.07572.