SUPPLEMENTARY MATERIAL

A Simulation functions

A.1 Schaffer function

The two-dimensional "fourth" Schaffer function (Surjanovic and Bingham, 2013) is defined as

$$f(x_1, x_2) = 0.5 + \frac{\cos^2\left(\sin(|x_1^2 - x_2^2|)\right) - 0.5}{\left[1 + 0.001\left(x_1^2 + x_2^2\right)\right]^2}.$$

We use the restricted domain $X \in [-2, 2]^2$.

A.2 G-function

The G-function (Marrel et al., 2009) is defined in d-dimension over the unit cube $X \in [0, 1]^d$ as

$$f(\mathbf{x}) = \prod_{i=1}^{d} \frac{|4x_i - 2| + a_i}{1 + a_i} \quad \text{where} \quad a_i = \frac{i-2}{2} \quad \text{for all} \quad i = 1, \ldots, d.$$

B Performance metrics

B.1 Prediction error

Let $\mu^\star$ represent posterior mean predictions at n_p testing locations. Let y^{true} represent the corresponding observed values. Root mean squared error (RMSE) and root mean squared prediction error (RMSPE) are respectively defined as

$$\text{RMSE} = \sqrt{\frac{1}{n_p}\sum_{i=1}^{n_p}\left((\mu_i^\star - y_i^{\text{true}})^2\right)} \qquad \text{RMSPE} = \sqrt{\frac{1}{n_p}\sum_{i=1}^{n_p}\left(\left(100 * \frac{\mu_i^\star - y_i^{\text{true}}}{y_i^{\text{true}}}\right)^2\right)}$$

B.2 Uncertainty quantification

Given a Gaussian posterior predictive distribution with predicted mean $\mu^\star$ and point-wise standard deviations $\sigma^\star$, the continuous rank probability score (CRPS; Gneiting and Raftery, 2007) is defined as

$$\text{CRPS}\left(y^{\text{true}} \mid \mu^\star, \sigma^\star\right) = \frac{1}{n_p}\sum_{i=1}^{n_p}\left[\sigma_i^\star\left(2\phi(z_i) + z_i\left(2 * \Phi(z_i) - 1\right) - \frac{1}{\sqrt{\pi}}\right)\right] \quad \text{for} \quad z_i = \frac{y_i^{\text{true}} - \mu_i^\star}{\sigma_i^\star}$$

where ϕ is the standard Gaussian pdf and Φ is the standard Gaussian cdf.

C Derivations

C.1 Partitioned matrix inverse

The inverse of a partitioned matrix follows (Barnett, 1979):

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}^{-1} \quad \text{where} \quad \begin{aligned} A_{11} &= \left(B_{11} - B_{12}B_{22}^{-1}B_{21}\right)^{-1} \\ A_{12} &= -B_{11}^{-1}B_{12}\left(B_{22} - B_{21}B_{11}^{-1}B_{12}\right)^{-1} \\ A_{21} &= -B_{22}^{-1}B_{21}\left(B_{11} - B_{12}B_{22}^{-1}B_{21}\right)^{-1} \\ A_{22} &= \left(B_{22} - B_{21}B_{11}^{-1}B_{12}\right)^{-1} \end{aligned}$$

C.2 Vecchia posterior predictive moments

We aim to predict $\mathcal{Y}$ at locations $\mathcal{W}$ conditioned on observed Y and W. We assume a zero-mean Gaussian process prior distribution,

$$\begin{bmatrix} Y \\ \mathcal{Y} \end{bmatrix} \sim \mathcal{N}_{n+n_p}\left(0, \ \Sigma_{\text{stack}}\right) \quad \text{where} \quad \Sigma_{\text{stack}} = \Sigma\left(\begin{bmatrix} W \\ \mathcal{W} \end{bmatrix}\right) = \begin{bmatrix} \Sigma(W) & \Sigma(W,\mathcal{W}) \\ \Sigma(\mathcal{W},W) & \Sigma(\mathcal{W}) \end{bmatrix}.$$

Under the Vecchia-approximation, we decompose the precision matrix using the sparse upper-lower Cholesky decomposition, with entries populated according to (9),

$$U_{\text{stack}} = \begin{bmatrix} U_w & U_{w,\mathcal{W}} \\ 0 & U_{\mathcal{W}} \end{bmatrix} \quad \text{such that} \quad \Sigma_{\text{stack}} = \left(U_{\text{stack}}U_{\text{stack}}^\top\right)^{-1} = \left(\begin{bmatrix} U_wU_w^\top + U_{w,\mathcal{W}}U_{w,\mathcal{W}}^\top & U_{w,\mathcal{W}}U_{\mathcal{W}}^\top \\ U_{\mathcal{W}}U_{w,\mathcal{W}}^\top & U_{\mathcal{W}}U_{\mathcal{W}}^\top \end{bmatrix}\right)^{-1}.$$

We aim to find a closed-form solution to the posterior predictive moments (3),

$$\mathcal{Y} \mid Y, W \sim \mathcal{N}_{n_p}\left(\mu^\star, \Sigma^\star\right) \quad \text{for} \quad \begin{aligned} \mu^\star &= \Sigma(\mathcal{W},W)\Sigma(W)^{-1}Y \\ \Sigma^\star &= \Sigma(\mathcal{W}) - \Sigma(\mathcal{W},W)\Sigma(W)^{-1}\Sigma(W,\mathcal{W}), \end{aligned}$$

which can avoid dense covariance matrices by instead relying on elements of the sparse U_{stack}. The simplification of $\Sigma^\star$ follows directly from the partitioned matrix inverse (App. C.1),

$$U_{\mathcal{W}}U_{\mathcal{W}}^\top = \left(\Sigma(\mathcal{W}) - \Sigma(\mathcal{W},W)\Sigma(W)^{-1}\Sigma(W,\mathcal{W})\right)^{-1} \quad \Longrightarrow \quad \Sigma^\star = \left(U_{\mathcal{W}}U_{\mathcal{W}}^\top\right)^{-1}.$$

The calculation of $\mu^\star$ first involves simplification of $\Sigma(W)$ and $\Sigma(\mathcal{W},W)$, again using partitioned matrix inverses (App. C.1):

$$\begin{aligned} \Sigma(W) &= \left(U_wU_w^\top + U_{w,\mathcal{W}}U_{w,\mathcal{W}}^\top - U_{w,\mathcal{W}}U_{\mathcal{W}}^\top\left(U_{\mathcal{W}}U_{\mathcal{W}}^\top\right)^{-1}U_{\mathcal{W}}U_{w,\mathcal{W}}^\top\right)^{-1} \\ &= \left(U_wU_w^\top\right)^{-1} \\ \Sigma(\mathcal{W},W) &= -\left(U_{\mathcal{W}}U_{\mathcal{W}}^\top\right)^{-1}U_{\mathcal{W}}U_{w,\mathcal{W}}^\top\left(U_wU_w^\top + U_{w,\mathcal{W}}U_{w,\mathcal{W}}^\top - U_{w,\mathcal{W}}U_{\mathcal{W}}^\top\left(U_{\mathcal{W}}U_{\mathcal{W}}^\top\right)^{-1}U_{\mathcal{W}}U_{w,\mathcal{W}}^\top\right)^{-1} \\ &= -\left(U_{\mathcal{W}}U_{\mathcal{W}}^\top\right)^{-1}U_{\mathcal{W}}U_{w,\mathcal{W}}^\top\left(U_wU_w^\top\right)^{-1}. \end{aligned}$$

Together, these yield

$$\begin{aligned}\mu^\star &= \Sigma(\mathcal{W}, W)\Sigma(W)^{-1}Y \\ &= -\left(U_\mathcal{W}U_\mathcal{W}^\top\right)^{-1}U_\mathcal{W}U_{w,\mathcal{W}}^\top\left(U_wU_w^\top\right)^{-1}U_wU_w^\top Y \\ &= -\left(U_\mathcal{W}^\top\right)^{-1}U_{w,\mathcal{W}}^\top Y.\end{aligned}$$

D Conditioning set size

Here we evaluate Vecchia-DGP (DGP VEC) and scaled Vecchia-GP (GP SVEC) models on the two-dimensional Schaffer function while varying the conditioning set size, m. The set-up is identical to that of Section 5.1, with the exception that we worked with $n = 1,000$ and $m \in \{5, 10, 25, 50, 100\}$. Resulting RMSE and CRPS from 20 MC repetitions are shown in Figure 7.

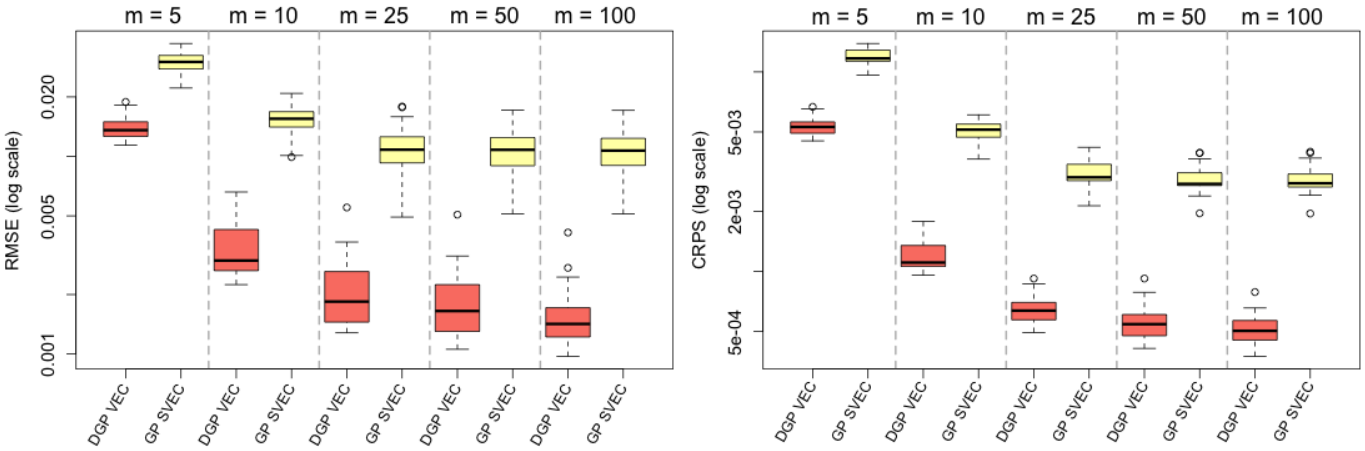

Figure 7: RMSE (left) and CRPS (right) for fits to the 2d Schaffer function as conditioning set size (m) increases (same m used for training and prediction). Boxplots represent the spread of 20 repetitions.

As expected, increasing the size of the conditioning set improves predictive accuracy for both models. However, the benefits of conditioning on more points diminishes beyond $m = 25$. While present in both models, this "leveling off" effect appears more starkly in the stationary GP SVEC than the non-stationary DGP VEC. Table 1 reports computation time (in minutes on a 16-core hyperthreaded, Intel i9 CPU at 3.6GHz) for each model/m configuration. The optimization-based inference of the GP SVEC model is fast enough that larger m do not incur additional computational costs, practically speaking. The sampling based nature of DGP VEC reveals the cubic costs of larger conditioning sets ($\mathcal{O}(nm^3)$). These cubic costs, and the minimal predictive improvements realized with larger m, motivate our choice of $m = 25$ for all other exercises and as the default setting in our software.

Model	$m = 5$	$m = 10$	$m = 25$	$m = 50$	$m = 100$
DGP VEC	3.16	4.19	8.91	25.80	92.09
GP SVEC	0.02	0.01	0.01	0.02	0.07

Table 1: Computation times in minutes for 2d Schaffer function exercise of Figure 7.

E Additional simulations

E.1 Simulation with noise

As an example of a noisy simulation, we re-create the Monte Carlo exercise for the G-function (Marrel et al., 2009) with the addition of additive Gaussian noise $\epsilon_i \sim \mathcal{N}(0, 0.01^2)$. The set-up is equivalent to that of Section 5.1, except that each model is now tasked with estimating a noise parameter (i.e. `true_g = NULL` in `deepgp`). The DGP HMC and GP SVEC models have built-in capability to estimate noise. The DGP DSVI model does not, so we simply fix the noise parameter to the true variance ($g = 0.01$). To account for the extra challenge of distinguishing signal from noise, we double the data sizes to $n \in \{6000, 10000, 14000\}$ and $n_p = 10000$. Code to reproduce these results is available in our github repository.

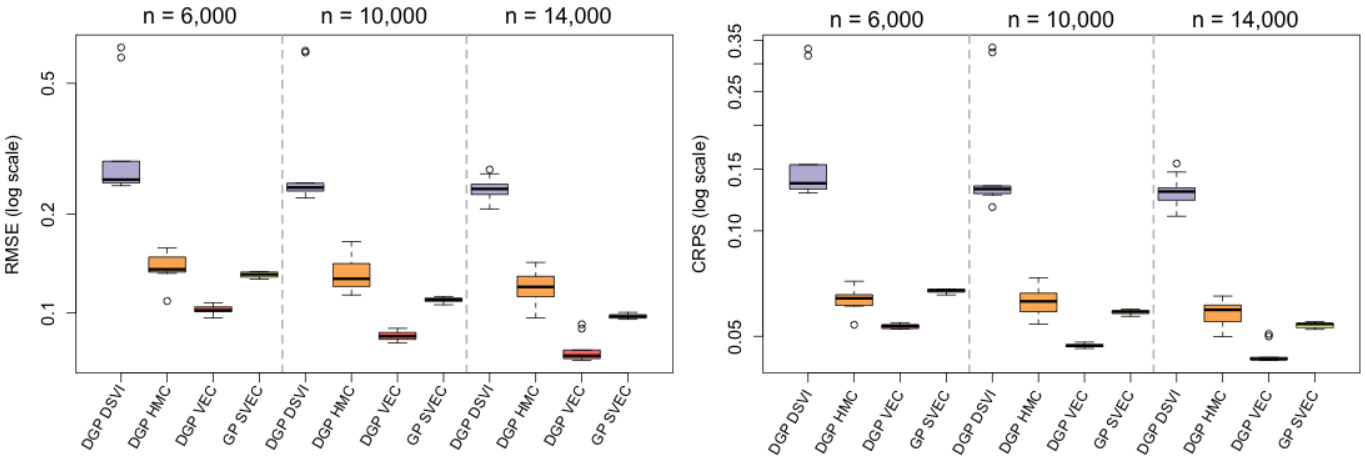

Figure 8: RMSE (left) and CRPS (right) on log scales for the 4d G-function observed with Gaussian white noise. Boxplots represent the spread of 10 MC repetitions.

The results resemble Figure 5, suggesting the addition of noise does not affect the comparative efficacy of the models. DGP HMC and GP SVEC perform similarly, the former benefiting from the flexibility of DGP layers and the latter benefiting from the Vecchia approximation (as compared to inducing points). The DGP VEC model outperforms across the board. When matched by training/testing data, DGP VEC had lower RMSE and CRPS than each of these comparators in 30/30 trials.

E.2 Higher dimensional simulation

To display functionality in higher dimension, we again re-create the Monte Carlo exercise for the G-function, this time expanding to $d = 6$. The set-up is equivalent to that of Section 5.1. Higher dimension demands larger data sizes; we entertain random LHS training designs of size $n \in \{10000, 20000, 30000\}$ and LHS testing designs of size $n_p = 20000$.

As in the lower dimensional exercises, DGP VEC outperforms on both prediction error and uncertainty quantification. Both DGP DSVI and DGP HMC are limited by inducing point approximations which are especially blurry in high dimensions. We note that the DGP HMC model is at a slight disadvantage since it estimates a noise parameter – the poor fits from this model may be over-estimating the noise.

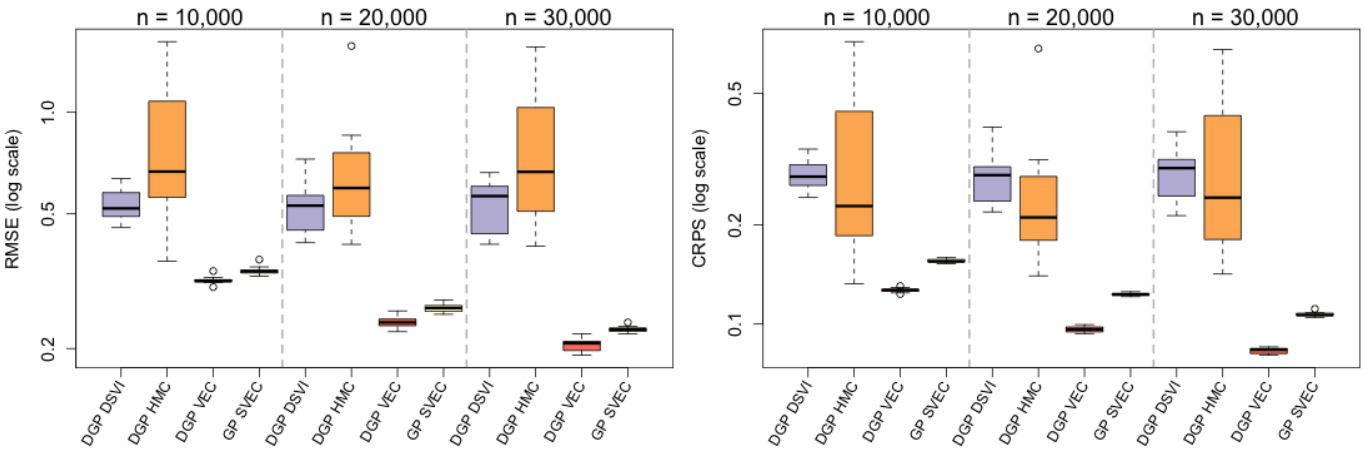

Figure 9: RMSE (left) and CRPS (right) on log scales for the 6d G-function. Boxplots represent the spread of 10 MC repetitions.

F Computation times

The following tables report computation times (in minutes) for the simulation exercises of Section 5 and Supp. E. We report the computation time for a single Monte Carlo exercise; times for each randomized trial are similar. All times were recorded on a 16-core hyperthreaded, Intel i9 CPU at 3.6GHz. The compute times of MCMC methods are directly tied to the number of MCMC samples collected. DGP VEC models sampled 3000 iterations for the Schaffer function, 3000 for the G-function, and 2000 for the satellite simulation. DGP HMC models utilized the default of 10,000 iterations. Un-approximated models (DGP FULL and GP) were not run for larger data sizes due to the cubic computational costs.

Model	**N = 100**	**N = 500**	**N = 1,000**
DGP DSVI	0.17	0.50	0.97
DGP HMC	0.58	1.45	2.43
DGP FULL	1.71	194.61	1605.94
DGP VEC	2.08	5.12	8.16
DGP VEC noU.	1.95	5.04	8.27
GP	0.03	0.55	3.48
GP SVEC	0.04	0.08	0.07

Table 2: Computation times in minutes for the Schaffer function exercises of Section 5.1 (Figure 4).

Model	N = 3,000	N = 5,000	N = 7,000
DGP DSVI	3.10	5.11	7.14
DGP HMC	7.59	12.21	17.35
DGP VEC	39.70	64.38	89.06
GP	807.33	N/A	N/A
GP SVEC	0.05	0.09	0.09

Table 3: Computation times in minutes for the G-function exercises of Section 5.1 (Figure 5).

Model	N = 10,000	N = 50,000	N = 100,000
DGP DSVI	12.00	59.13	109.20
DGP HMC	26.72	26.29	26.47
DGP VEC	190.57	809.75	1606.15
GP SVEC	0.17	0.18	0.18

Table 4: Computation times in minutes for the satellite simulation exercises of Section 5.2 (Figure 6).

Model	N = 6,000	N = 10,000	N = 14,000
DGP DSVI	6.15	10.17	14.22
DGP HMC	14.69	24.94	25.19
DGP VEC	84.85	132.41	179.44
GP SVEC	0.09	0.09	0.09

Table 5: Computation times in minutes for the noisy G-function exercise of Supp. E.1 (Figure 8).

Model	N = 10,000	N = 20,000	N = 30,000
DGP DSVI	11.73	21.82	35.04
DGP HMC	25.91	26.05	26.14
DGP VEC	235.49	425.43	623.89
GP SVEC	0.43	0.34	0.34

Table 6: Computation times in minutes for the 6d G-function exercise of Supp. E.2 (Figure 9).