Recent Development on Dynamic Computer Experiments

C. Devon Lin[1]

Joint work with Ru Zhang[1] and Pritam Ranjan[2]
[1]Queen's University
[2]Indian Institute of Management Indore

Aug 7, 2017, Banff Workshop on Latest Advances in the Theory and Applications of Design and Analysis of Experiments

Outline

- Large-scale dynamic computer experiments
- Proposed SVD-based Gaussian process models
- Inverse problem
- Concluding remarks

Dynamic Computer Simulators

Computer simulators with time-varying outputs

Dynamic Computer Simulators

Computer simulators with time-varying outputs

- Climate change model (Bhattacharya, 2007)
- Rainfall-runoff model (Conti et al., 2009)
- Marrow stem cell kinetics model (Morris, 2012; Jones et al., 1991)
- Vehicle suspension system (Bayrri et al., 2007)
- Hydrological saturated path model (logSPMs) (Liu and West, 2007)
- Sheffield Dynamic Global Vegetation Model (Conti and O'Hagan, 2010)
- The Lyon-Fedder-Mobary (LFM) model (Pratola et al., 2013)
- Volcanic pyroclastic flow model (Gu and Berger, 2016)

Dynamic Computer Simulators

- Systems evolving over time

$$Y_t = f(z_t, Y_{t-1}), t = 1, \ldots, T \tag{1}$$

where Y_t and Y_{t-1} represent the state vector at time-step t and $t-1$, respectively, $\mathbf{z}_t = (\mathbf{x}, \mathbf{w}_t)$ denotes the time-independent input $\mathbf{x}$ and the forcing inputs $\mathbf{w}_t$ at time-step t

Dynamic Computer Simulators

- Systems evolving over time

$$Y_t = f(z_t, Y_{t-1}), t = 1, \ldots, T \tag{1}$$

  where Y_t and Y_{t-1} represent the state vector at time-step t and $t-1$, respectively, $\mathbf{z}_t = (\mathbf{x}, \mathbf{w}_t)$ denotes the time-independent input $\mathbf{x}$ and the forcing inputs $\mathbf{w}_t$ at time-step t

- Systems generating time series outputs
  - the training data inputs $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
  - at time t, the outputs $\mathbf{y}_t = (y(\mathbf{x}_1, t), \ldots, y(\mathbf{x}_N, t))^T$

Dynamic Computer Simulators

- Systems evolving over time

$$Y_t = f(\mathbf{z}_t, Y_{t-1}), t = 1, \ldots, T \tag{2}$$

  where Y_t and Y_{t-1} represent the state vector at time-step t and $t-1$, respectively, $\mathbf{z}_t = (\mathbf{x}, \mathbf{w}_t)$ denotes the time-independent input $\mathbf{x}$ and the forcing inputs $\mathbf{w}_t$ at time-step t

- Systems generating time series outputs
  - the training data inputs $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
  - at time t, the outputs $\mathbf{y}_t = (y(\mathbf{x}_1, t), \ldots, y(\mathbf{x}_N, t))^T$

Existing Methods

- Many single-output emulator: emulate the T outputs separately, each via a single-output emulator (Conti and O'Hagan, 2009)

Existing Methods

- Many single-output emulator: emulate the T outputs separately, each via a single-output emulator (Conti and O'Hagan, 2009)
- Multi-output emulator: (a) multivariate GP models (Conti and O'Hagan, 2009); (b) represent time series outputs as linear combinations of a fixed set of basis such as singular vectors or wavelet basis and impose Gaussian process models on the linear coefficients (Higdon et al., 2008)

Existing Methods

- Many single-output emulator: emulate the T outputs separately, each via a single-output emulator (Conti and O'Hagan, 2009)
- Multi-output emulator: (a) multivariate GP models (Conti and O'Hagan, 2009); (b) represent time series outputs as linear combinations of a fixed set of basis such as singular vectors or wavelet basis and impose Gaussian process models on the linear coefficients (Higdon et al., 2008)
- Time-input emulator: include time as an extra input (Kennedy and O'Hagan, 2001)

Existing Methods

- Many single-output emulator: emulate the T outputs separately, each via a single-output emulator (Conti and O'Hagan, 2009)
- Multi-output emulator: (a) multivariate GP models (Conti and O'Hagan, 2009); (b) represent time series outputs as linear combinations of a fixed set of basis such as singular vectors or wavelet basis and impose Gaussian process models on the linear coefficients (Higdon et al., 2008)
- Time-input emulator: include time as an extra input (Kennedy and O'Hagan, 2001)
- Time-varying autoregression (TVAR) models with GP residuals (Liu and West, 2007)

Large-scale Computer Experiments

Computer Experiments with massive outputs

Large-scale Computer Experiments

Computer Experiments with massive outputs

- Mutli-step interpolator (Haaland and Qian, 2011)
- Fixed rank kriging (Cressie and Johannesson, 2008)
- Compactly supported covariance (Kaufman et al., 2011)
- Local Gaussian process (Gramacy and Apley, 2015; Gramacy et al., 2015; Gramacy and Haaland, 2016)

Local Gaussian Process

- Data points far from an untried input $\mathbf{x}$ have vanishingly small influence on the predictive distribution (assuming the usual choices of covariance).

Local Gaussian Process

- Data points far from an untried input $\mathbf{x}$ have vanishingly small influence on the predictive distribution (assuming the usual choices of covariance).
- Nearest neighbourhood (NN) approach selects data close to the input location for prediction such that the selected input locations are distributed as uniformly as possible around the location for prediction.

Local Gaussian Process

- Data points far from an untried input **x** have vanishingly small influence on the predictive distribution (assuming the usual choices of covariance).
- Nearest neighbourhood (NN) approach selects data close to the input location for prediction such that the selected input locations are distributed as uniformly as possible around the location for prediction.
- Emery (2009) built a local neighborhood by sequentially including data that make the Kriging variance decrease more.

Local Gaussian Process

- Data points far from an untried input **x** have vanishingly small influence on the predictive distribution (assuming the usual choices of covariance).
- Nearest neighbourhood (NN) approach selects data close to the input location for prediction such that the selected input locations are distributed as uniformly as possible around the location for prediction.
- Emery (2009) built a local neighborhood by sequentially including data that make the Kriging variance decrease more.
- Gramacy and Apley (2015) uses a sequential greedy algorithm and an optimality criterion for finding a non-trivial local neighbourhood set.

Large-scale Dynamic Computer Experiments

- Computer experiments with time series outputs at a large number of inputs (ten of thousands)

Large-scale Dynamic Computer Experiments

- Computer experiments with time series outputs at a large number of inputs (ten of thousands)
- The existing methods for dynamic computer experiments cannot handle thousands of data points

Large-scale Dynamic Computer Experiments

- Computer experiments with time series outputs at a large number of inputs (ten of thousands)
- The existing methods for dynamic computer experiments cannot handle thousands of data points
- It is not trivial to extend the exiting methods for large-scale computer experiments to accommodate the times series outputs

Large-scale Dynamic Computer Experiments

- Computer experiments with time series outputs at a large number of inputs (ten of thousands)
- The existing methods for dynamic computer experiments cannot handle thousands of data points
- It is not trivial to extend the exiting methods for large-scale computer experiments to accommodate the times series outputs
- Thus, it calls for a new method for emulating large-scale dynamic computer simulators

The Proposed Approach

- The building blocks: singular value decomposition (SVD), and the local surrogate idea

The Proposed Approach

- The building blocks: singular value decomposition (SVD), and the local surrogate idea
- A sequential greedy algorithm to construct the local neighbourhood set

The Proposed Approach

- The building blocks: singular value decomposition (SVD), and the local surrogate idea
- A sequential greedy algorithm to construct the local neighbourhood set
- An optimality criterion to search for next point

Model Formulation

- For N training points, let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T$ be the $N \times q$ input matrix, and $\mathbf{Y} = [\mathbf{y}(\mathbf{x}_1), \ldots, \mathbf{y}(\mathbf{x}_N)]$ be the $L \times N$ matrix of time series response

Model Formulation

- For N training points, let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T$ be the $N \times q$ input matrix, and $\mathbf{Y} = [\mathbf{y}(\mathbf{x}_1), \ldots, \mathbf{y}(\mathbf{x}_N)]$ be the $L \times N$ matrix of time series response
- The SVD on $\mathbf{Y}$ gives

$$\mathbf{Y} = \mathbf{U}\mathbf{D}\mathbf{V}^T = \sum_{i=1}^{k} d_i \mathbf{u}_i \mathbf{v}_i^T \tag{3}$$

where

- $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_k]$ is an $L \times k$ column orthogonal matrix of left singular vectors, $k = \min\{N, L\}$
- $\mathbf{D} = \mathrm{diag}(d_1, \ldots, d_k)$ is a $k \times k$ diagonal matrix of singular values sorted in decreasing order
- $\mathbf{V}$ is an $N \times k$ column-orthogonal matrix of right singular vectors

Model Formulation

The SVD-based GP model assumes that

$$\mathbf{y}(\mathbf{x}) = \sum_{i=1}^{p} c_i(\mathbf{x})\mathbf{b}_i + \varepsilon \tag{4}$$

where $\mathbf{b}_i = d_i\mathbf{u}_i \in \mathcal{R}^L$, $c_i(\mathbf{x}) \sim \mathrm{GP}(0, \sigma_i^2 K_i(\cdot,\cdot;\theta_i))$, for $i = 1,\ldots,p$, and $\varepsilon \sim N(0, \sigma^2\mathbf{I}_L)$. The number of significant singular values, p in (4), is determined empirically by the cumulative percentage criterion

$$p = \min\left\{ m : \frac{\sum_{i=1}^{m} d_i}{\sum_{i=1}^{k} d_i} > \gamma \right\},$$

where γ is a prespecified threshold of explained variation ($\gamma \in (0.9, 0.98)$)

SVD-based GP Models

- Higdon et al. (2008) used SVD-based GP models for computer model calibration with high-dimensional outputs

SVD-based GP Models

- Higdon et al. (2008) used SVD-based GP models for computer model calibration with high-dimensional outputs
- The formulation reduces the problem of building an emulator that maps from $(0,1)^q$ to $\mathcal{R}^L$ to building p independent, univariate GP models for each c_i

SVD-based GP Models

- Higdon et al. (2008) used SVD-based GP models for computer model calibration with high-dimensional outputs
- The formulation reduces the problem of building an emulator that maps from $(0,1)^q$ to $\mathcal{R}^L$ to building p independent, univariate GP models for each c_i
- Higdon et al. (2008) employ a fully Bayesian analysis via Markov chain Monte Carlo (MCMC), which is computationally intensive in our applications

Empirical Bayesian Inference

- The parameters
  - σ^2: the error variance
  - σ_i^2: the process variance, for $i = 1, \ldots, p$
  - $\theta_i = (\theta_{i1}, \ldots, \theta_{iq})$: the correlation parameters in Gaussian correlation function

Empirical Bayesian Inference

- The parameters
  - σ^2: the error variance
  - σ_i^2: the process variance, for $i = 1, \ldots, p$
  - $\theta_i = (\theta_{i1}, \ldots, \theta_{iq})$: the correlation parameters in Gaussian correlation function
- The priors:
  - $[\sigma^2] \sim \text{IG}\left(\frac{\alpha}{2}, \frac{\beta}{2}\right)$
  - $[\sigma_i^2] \sim \text{IG}\left(\frac{\alpha_i}{2}, \frac{\beta_i}{2}\right), i = 1, \ldots, p$
  - $[\frac{1}{\theta_{ij}}] \sim \textit{Gamma}(3/2, a)$, for $i = 1, \ldots, p$ and $j = 1, \ldots, q$, where the scale parameter a is chosen such that maximum squared distance among any two points of the design matrix lies at the position of 95% quantile (Gramacy, 2015)

Empirical Bayesian Inference

The approximate predictive distribution of $\mathbf{y}(\mathbf{x}_0|\mathbf{Y})$:

$$\pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{Y}) \approx \pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*, \hat{\sigma}^2, \hat{\Theta}) \approx \mathcal{N}\big(\mathbf{B}\hat{\mathbf{c}}(\mathbf{x}_0|\mathbf{V}^*, \hat{\Theta}), \mathbf{B}\Lambda(\mathbf{V}^*, \hat{\Theta})\mathbf{B}^T + \hat{\sigma}^2\mathbf{I}_L\big). \tag{5}$$

where

- $\hat{\mathbf{c}}(\mathbf{x}_0|\mathbf{V}^*, \hat{\Theta}) = [\hat{c}_1(\mathbf{x}_0|\mathbf{v}_1, \hat{\theta}_1), \ldots, \hat{c}_p(\mathbf{x}_0|\mathbf{v}_p, \hat{\theta}_p)]^T$
- $\Lambda(\mathbf{V}^*, \hat{\Theta}) = \mathrm{diag}\big(\hat{\sigma}_1^2(\mathbf{x}_0|\mathbf{v}_1, \hat{\theta}_1), \ldots, \hat{\sigma}_p^2(\mathbf{x}_0|\mathbf{v}_p, \hat{\theta}_p)\big)$.

- $[\sigma^2|Y]$ follows the inverse Gamma distribution $\mathrm{IG}((NL+\alpha)/2, (r^T r+\beta)/2)$, and

$$\hat{\sigma}^2 = \underset{\sigma^2}{\operatorname{argmax}}\, \pi(\sigma^2|Y) = \frac{1}{NL+\alpha+2}\left(r^T r+\beta\right). \tag{6}$$

- $\hat{\theta}_i = \underset{\theta_i}{\operatorname{argmax}}\, \pi(\theta_i|\mathbf{v}_i), i = 1,\ldots,p.$

$$\pi(\theta_i|\mathbf{v}_i) \propto |K_i|^{-\frac{1}{2}} \left(\frac{\beta_i+\psi_i}{2}\right)^{-(\alpha_i+N)/2} \pi(\theta_i), \tag{7}$$

Empirical Bayesian Inference

$$\pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*, \Theta, \sigma^2) \approx \mathcal{N}\left(\mathbf{B}\hat{\mathbf{c}}(\mathbf{x}_0|\mathbf{V}^*, \Theta), \mathbf{B}\Lambda(\mathbf{V}^*, \Theta)\mathbf{B}^T + \sigma^2\mathbf{I}_L\right), \quad (8)$$

To compute (8), we make use of

- the distribution of $[\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*, \Theta, \sigma^2, c_1(\mathbf{x}_0), \ldots, c_p(\mathbf{x}_0)]$
- the distribution of $[c_i(\mathbf{x}_0)|\mathbf{v}_i, \theta_i]$

Lemma

(Gelman et al., 2014) Suppose $[\boldsymbol{y}|\beta,\sigma^2] \sim \mathcal{N}(\boldsymbol{X}\beta,\sigma^2\boldsymbol{I}_n)$ *and* $[\beta] \sim \mathcal{N}(\boldsymbol{b},V)$*, where* $\boldsymbol{y} \in \mathbb{R}^n$*,* $\beta,\boldsymbol{b} \in \mathbb{R}^m$*,* $\boldsymbol{X}$ *is an* $n \times m$ *matrix, and* V *is an* $m \times m$ *positive definite covariance matrix. Then,* $[\boldsymbol{y}|\sigma^2] \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{b},\boldsymbol{X}V\boldsymbol{X}^T+\sigma^2\boldsymbol{I}_n)$.

- given $[\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*,\Theta,\sigma^2,c_1(\mathbf{x}_0),\ldots,c_p(\mathbf{x}_0)]$
- given $[c_i(\mathbf{x}_0)|\mathbf{v}_i,\theta_i]$
- derive $[\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*,\Theta,\sigma^2]$

Empirical Bayesian Inference

The conditional distribution of $c_i(\mathbf{x}_0)$ given $(\mathbf{v}_i, \theta_i)$ are independent non-central t with $N + \alpha_i$ degrees of freedom, for $i = 1, \ldots, p$, i.e.,

$$[c_i(x_0)|\mathbf{v}_i, \theta_i] \sim t_{N+\alpha_i}\left(\hat{c}_i(\mathbf{x}_0|\mathbf{v}_i, \theta_i), \hat{\sigma}_i^2(\mathbf{x}_0|\mathbf{v}_i, \theta_i)\right), \tag{9}$$

with the location parameter

$$\hat{c}_i(\mathbf{x}_0|\mathbf{v}_i, \theta_i) = k_i^T(\mathbf{x}_0)K_i^{-1}\mathbf{v}_i, \tag{10}$$

with $k_i(\mathbf{x}_0) = [K(\mathbf{x}_0, \mathbf{x}_1; \theta_i), \ldots, K(\mathbf{x}_0, \mathbf{x}_N; \theta_i)]^T$, and the scale parameter

$$\hat{\sigma}_i^2(\mathbf{x}_0|\mathbf{v}_i, \theta_i) = \frac{(\beta_i + \psi_i)\left(1 - k_i^T(\mathbf{x}_0)K_i^{-1}k_i(\mathbf{x}_0)\right)}{\alpha_i + N}. \tag{11}$$

- The posterior distribution of $c_i(\mathbf{x}_0)|\mathbf{v}_i, \theta_i$

$$\pi\left(c_i(\mathbf{x}_0)|\mathbf{v}_i, \theta_i\right) \approx \mathcal{N}\left(\hat{c}_i(\mathbf{x}_0|\mathbf{v}_i, \theta_i), \hat{\sigma}_i^2(\mathbf{x}_0|\mathbf{v}_i, \theta_i)\right). \tag{12}$$

Local SVD-based GP Models

- The full SVD-based GP model that uses all N training points involves with inverting an $N \times N$ correlation matrix and the corresponding computing time is $O(N^3)$.

Local SVD-based GP Models

- The full SVD-based GP model that uses all N training points involves with inverting an $N \times N$ correlation matrix and the corresponding computing time is $O(N^3)$.
- The local SVD-based GP model uses n $(\ll N)$ points for approximating the predicted response at an arbitrary $\mathbf{x}_0$

Local SVD-based GP Models

- The full SVD-based GP model that uses all N training points involves with inverting an $N \times N$ correlation matrix and the corresponding computing time is $O(N^3)$.
- The local SVD-based GP model uses n ($\ll N$) points for approximating the predicted response at an arbitrary $\mathbf{x}_0$
- The *naive local SVD-based GP model* uses n nearest neighbour points based on the Euclidean distance

Local SVD-based GP Models

- The full SVD-based GP model that uses all N training points involves with inverting an $N \times N$ correlation matrix and the corresponding computing time is $O(N^3)$.
- The local SVD-based GP model uses n ($\ll N$) points for approximating the predicted response at an arbitrary $\mathbf{x}_0$
- The *naive local SVD-based GP model* uses n nearest neighbour points based on the Euclidean distance
- The proposed *greedy local SVD-based GP model* sequentially finds the n neighbour points with the aim of reducing the prediction error

Greedy Local SVD-based GP Model

- Let k denote the current number of points in the neighbourhood set, $\mathbf{X}^{(k)}$ and $\mathbf{X} \backslash \mathbf{X}^{(k)}$ be the sets of selected and unselected (remaining) training points, respectively, and $\hat{\Theta}^{(k)} = \{\hat{\theta}_1^{(k)}, \ldots, \hat{\theta}_p^{(k)}\}$ be the estimated correlation parameters.

Greedy Local SVD-based GP Model

- Let k denote the current number of points in the neighbourhood set, $\mathbf{X}^{(k)}$ and $\mathbf{X} \backslash \mathbf{X}^{(k)}$ be the sets of selected and unselected (remaining) training points, respectively, and $\hat{\Theta}^{(k)} = \{\hat{\theta}_1^{(k)}, \ldots, \hat{\theta}_p^{(k)}\}$ be the estimated correlation parameters.
- The next follow-up point is chosen as

$$x_{k+1}^* = \underset{x \in X \backslash \mathbf{x}^{(k)}}{\operatorname{argmin}} \; J(\mathbf{x}_0, \mathbf{x}),$$

J-Criterion

$$J(\mathbf{x}_0, \mathbf{x}) = \mathrm{E}\left\{ \mathrm{E}\left[\left\| \mathbf{y}(\mathbf{x}_0) - \hat{\mathbf{y}}(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}) \right\|^2 \middle| \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right] \middle| \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right\}, \tag{13}$$

with

$$\begin{aligned} \hat{y}\left(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}\right) &= \mathrm{E}\left[y(x_0) \middle| \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right] \\ &= \mathbf{B}^{(k)} \hat{c}\left(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}\right), \end{aligned}$$

Proposition

Suppose the expectations in (13) and (14) are taken with respect to the approximate predictive distribution (8). Then, for any $\boldsymbol{x} \in \boldsymbol{X} \backslash \boldsymbol{X}^{(k)}$

$$J(\boldsymbol{x}_0, \boldsymbol{x}) = (\hat{\sigma}^{(k)})^2 L + \sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2 \left(x_0 | \boldsymbol{x}, \boldsymbol{v}_i^{(k)}, \hat{\theta}_i^{(k)} \right), \tag{14}$$

where $d_i^{(k)}$ *is the* i*-th largest singular value of* $\boldsymbol{Y}^{(k)}$*,*

$$\hat{\sigma}_i^2 \left(\boldsymbol{x}_0 | \boldsymbol{x}, \boldsymbol{v}_i^{(k)}, \hat{\theta}_i^{(k)} \right) = \frac{\rho_i^{(k)}(\boldsymbol{x}_0, \boldsymbol{x})}{\alpha_i + k} \left(\beta_i + \frac{\alpha_i + k}{\alpha_i + k - 1} \psi_i^{(k)} \right),$$

$$\rho_i^{(k)}(\mathbf{x}_0, \mathbf{x}) = 1 - \tilde{k}_i(\mathbf{x}_0, \mathbf{x})^T \tilde{K}_i^{-1}(\mathbf{x}) \tilde{k}_i(\mathbf{x}_0, \mathbf{x}),$$
$$\psi_i^{(k)} = (\mathbf{v}_i^{(k)})^T (\mathbf{K}_i^{(k)})^{-1} \mathbf{v}_i^{(k)},$$
$$\tilde{k}_i(\mathbf{x}_0, \mathbf{x}) = [K(\mathbf{x}_0, \mathbf{x}_1^{(k)}; \hat{\theta}_i^{(k)}), \ldots, K(\mathbf{x}_0, \mathbf{x}_k^{(k)}; \hat{\theta}_i^{(k)}), K(\mathbf{x}_0, \mathbf{x}; \hat{\theta}_i^{(k)})]^T,$$
$$\tilde{K}_i(\mathbf{x}) = \begin{bmatrix} \mathbf{K}_i^{(k)} & \mathbf{k}_i^{(k)}(\mathbf{x}) \\ \mathbf{k}_i^{(k)}(\mathbf{x})^T & 1 \end{bmatrix},$$

with $\mathbf{v}_i^{(k)}$ being the i-th column of $\mathbf{V}^{*(k)}$, for $i = 1, \ldots, p_k$, $\mathbf{x}_j^{(k)}$ being the j-th point of $\mathbf{X}^{(k)}$ for $j = 1, \ldots, k$, $\mathbf{K}_i^{(k)}$ being a $k \times k$ matrix with $K(\mathbf{x}_j^{(k)}, \mathbf{x}_l^{(k)}; \hat{\theta}_i^{(k)})$, as the (j, l)-th entry, and $\mathbf{k}_i^{(k)}(\mathbf{x}) = [K(\mathbf{x}, \mathbf{x}_1^{(k)}; \hat{\theta}_i^{(k)}), \ldots, K(\mathbf{x}, \mathbf{x}_k^{(k)}; \hat{\theta}_i^{(k)})]^T$.

J-Criterion

- Finding x_{k+1}^* by minimizing the J-criterion in Proposition 1 is equivalent to obtaining

$$x_{k+1}^* = \underset{x \in X \setminus \mathbf{x}^{(k)}}{\operatorname{argmin}} \left[\sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2 \left(x_0 | \mathbf{x}, \mathbf{v}_i^{(k)}, \hat{\theta}_i^{(k)} \right) \right]. \tag{15}$$

J-Criterion

- Finding x^*_{k+1} by minimizing the J-criterion in Proposition 1 is equivalent to obtaining

$$x^*_{k+1} = \underset{x \in X \setminus \mathbf{x}^{(k)}}{\operatorname{argmin}} \left[\sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2 \left(x_0 | \mathbf{x}, \mathbf{v}_i^{(k)}, \hat{\theta}_i^{(k)} \right) \right]. \tag{15}$$

- This J-criterion is a generalization of the active learning Cohn (ALC) criterion (Cohn et al., 1996)

$$\int_{\mathbf{x}} \left[\int_y \left(\hat{y}(\mathbf{x}) - y(\mathbf{x}) \right)^2 dP(y|\mathbf{x}) \right] dP(\mathbf{x}),$$

J-Criterion

- Finding x^*_{k+1} by minimizing the J-criterion in Proposition 1 is equivalent to obtaining

$$x^*_{k+1} = \underset{x \in X \setminus \mathbf{x}^{(k)}}{\operatorname{argmin}} \left[\sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2 \left(x_0 | \mathbf{x}, \mathbf{v}_i^{(k)}, \hat{\theta}_i^{(k)} \right) \right]. \tag{15}$$

- This J-criterion is a generalization of the active learning Cohn (ALC) criterion (Cohn et al., 1996)

$$\int_{\mathbf{x}} \left[\int_y \left(\hat{y}(\mathbf{x}) - y(\mathbf{x}) \right)^2 dP(y|\mathbf{x}) \right] dP(\mathbf{x}),$$

- We use L_2 norm discrepancy instead of the squared error.

Simulations

- We compare the performance of the three methods *full svdGP*, *naive local svdGP*, and *greedy local svdGP*

Simulations

- We compare the performance of the three methods *full svdGP*, *naive local svdGP*, and *greedy local svdGP*
- The performance is evaluated by comparing

$$\text{NMSPE}(x) = \frac{\sum_{t=1}^{L} \left(y_t(x) - \hat{y}_t(x)\right)^2}{\sum_{t=1}^{L} \left(y_t(x) - \bar{y}(x)\right)^2}, \tag{16}$$

where $y_t(x)$ is the real (scalar) response for input x at time t, $t = 1, \ldots, L$, $\hat{y}_t(x)$ is the corresponding model prediction, and $\bar{y}(x) = \sum_{t=1}^{L} y_t(\mathbf{x})/L$.

Example 1 (Forrester et al., 2008)

$$f(x,t) = (x_1 t - 2)^2 \sin(x_2 t - x_3), \tag{17}$$

where $x = (x_1, x_2, x_3)^T \in [4,10] \times [4,20] \times [1,7]$, and $t \in [1,2]$ is on a 200-point equidistant time-grid.

Example 1 (Forrester et al., 2008)

$$f(x,t) = (x_1 t - 2)^2 \sin(x_2 t - x_3), \tag{17}$$

where $x = (x_1, x_2, x_3)^T \in [4,10] \times [4,20] \times [1,7]$, and $t \in [1,2]$ is on a 200-point equidistant time-grid.

- The training data is generated using a 10000-point random Latin hypercube design (LHD)

Example 1 (Forrester et al., 2008)

$$f(x,t) = (x_1 t - 2)^2 \sin(x_2 t - x_3), \tag{17}$$

where $x = (x_1, x_2, x_3)^T \in [4,10] \times [4,20] \times [1,7]$, and $t \in [1,2]$ is on a 200-point equidistant time-grid.

- The training data is generated using a 10000-point random Latin hypercube design (LHD)
- The test data is generated using a 2000-point random LHD

Example 1 (Forrester et al., 2008)

$$f(x,t) = (x_1 t - 2)^2 \sin(x_2 t - x_3), \tag{17}$$

where $x = (x_1, x_2, x_3)^T \in [4,10] \times [4,20] \times [1,7]$, and $t \in [1,2]$ is on a 200-point equidistant time-grid.

- The training data is generated using a 10000-point random Latin hypercube design (LHD)
- The test data is generated using a 2000-point random LHD
- For local svdGP, use $n = 20, 40$, and $n_0 = \lceil n/4 \rceil, \lceil n/2 \rceil$

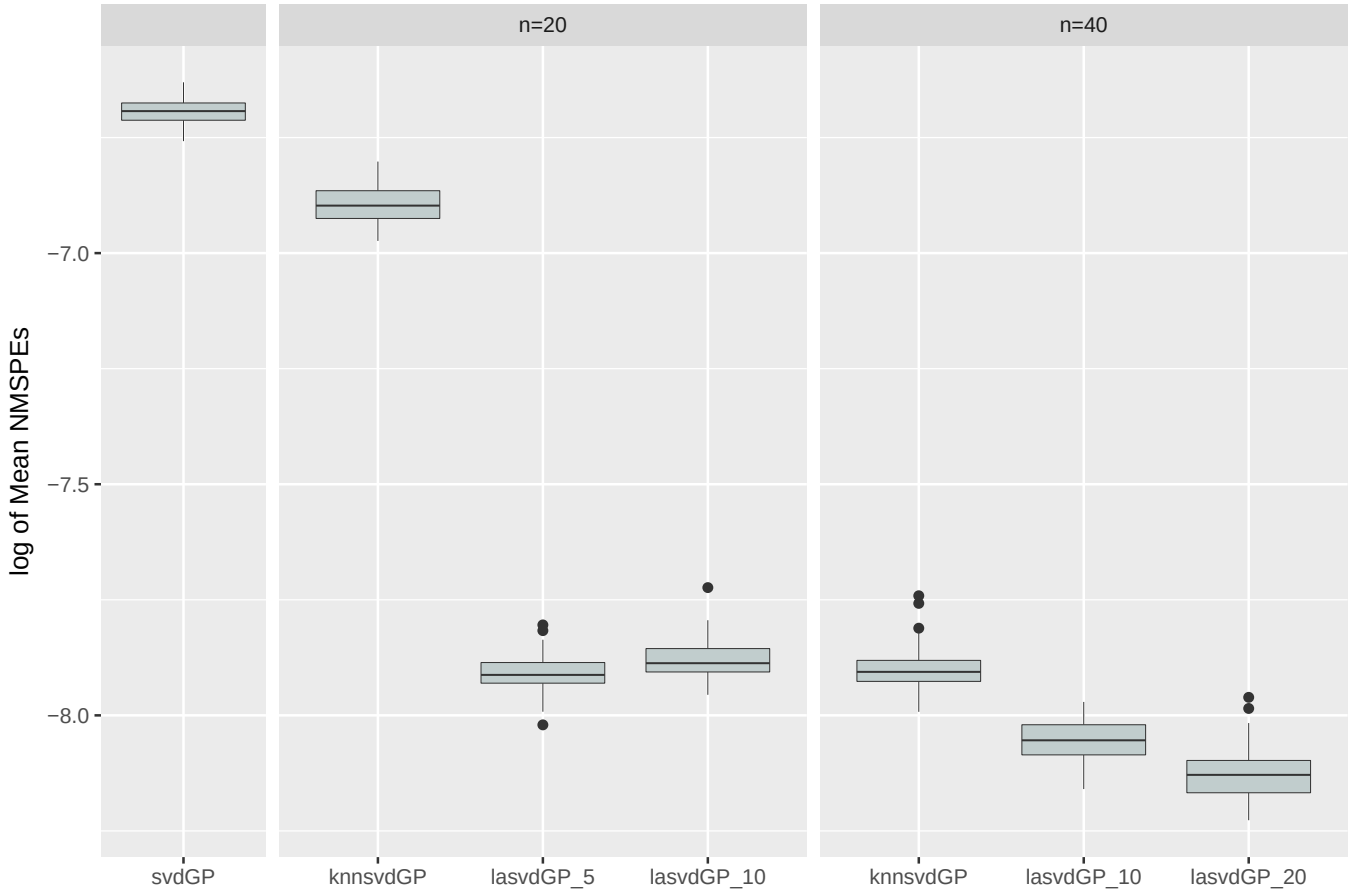


Figure: Boxplots of the log of mean NMSPE computed from 2,000 test points over 50 simulations for the computer simulator (17).

Example 2 (Bliznyuk et al.,2008)

$$f(x,t) = \frac{M}{\sqrt{Dt}} \exp\left(\frac{-s^2}{4Dt}\right) + \frac{M}{\sqrt{D(t-\tau)}} \exp\left(-\frac{(s-L)^2}{4D(t-\tau)}\right) I(\tau < t), \tag{18}$$

where $x = (M, D, L, \tau, s)^T$, M denotes the mass of pollutant spilled at each location, D is diffusion rate in the channel, L is location of the second spill, and τ is time of the second spill, $t \in [0.3, 60]$ is on a regular 200-point equidistant time grid,
$x \in [7, 13] \times [0.02, 0.12] \times [0.01, 3] \times [30.01, 30.295] \times [0, 3]$, .

Example 2

- The training data is generated using a 10000-point random LHD

Example 2

- The training data is generated using a 10000-point random LHD
- The test data is generated using a 2000-point random LHD

Example 2

- The training data is generated using a 10000-point random LHD
- The test data is generated using a 2000-point random LHD
- For local svdGP, use $n = 30, 50$, and $n_0 = \lceil n/4 \rceil, \lceil n/2 \rceil$

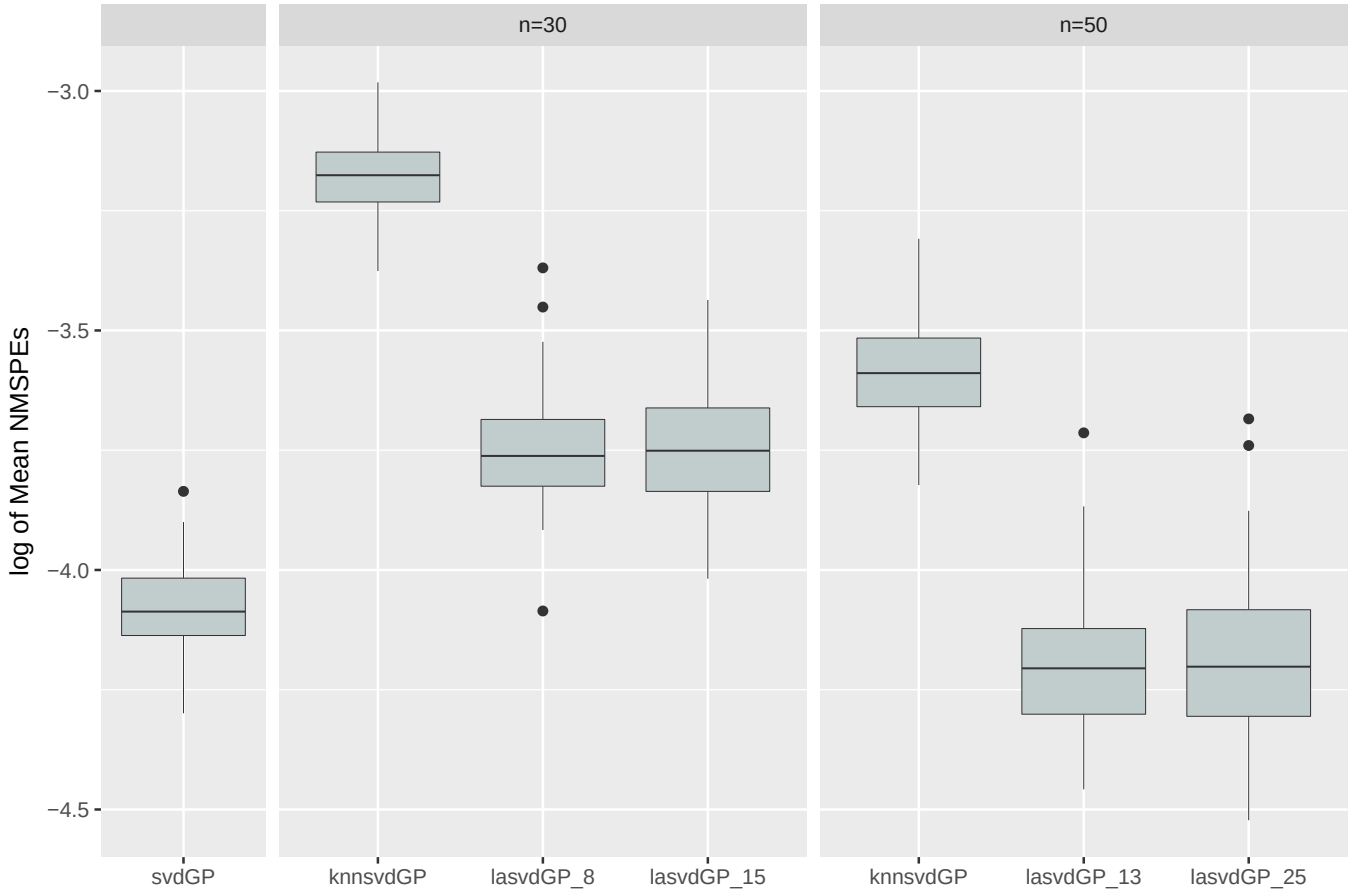


Figure: Boxplots of the log of mean NMSPE computed from 2,000 test points over 50 simulations for the computer simulator (19).

Example 3 (TDB simulator)

- The two-delay blowfly (TDB) model (Teismann et al., 2009) simulates European red mites (ERM) population dynamics under predator-prey interactions in apple orchards via numerically solving the Nicholson's blowfly differential equation (Gurney et al., 1980).

Example 3 (TDB simulator)

- The two-delay blowfly (TDB) model (Teismann et al., 2009) simulates European red mites (ERM) population dynamics under predator-prey interactions in apple orchards via numerically solving the Nicholson's blowfly differential equation (Gurney et al., 1980).
- The TDB model takes eleven input variables (e.g., death rates for different stages, fecundity, hatching time, survival rates, and so on) and returns the time series of ERM population evolutions at three stages, i.e., eggs, juveniles and adults.

Example 3 (TDB simulator)

- The two-delay blowfly (TDB) model (Teismann et al., 2009) simulates European red mites (ERM) population dynamics under predator-prey interactions in apple orchards via numerically solving the Nicholson's blowfly differential equation (Gurney et al., 1980).
- The TDB model takes eleven input variables (e.g., death rates for different stages, fecundity, hatching time, survival rates, and so on) and returns the time series of ERM population evolutions at three stages, i.e., eggs, juveniles and adults.
- $N = 20,000$ training and $M = 10,000$ test points are generated using random LHDs for the emulation of this process.

Example 3 (TDB simulator)

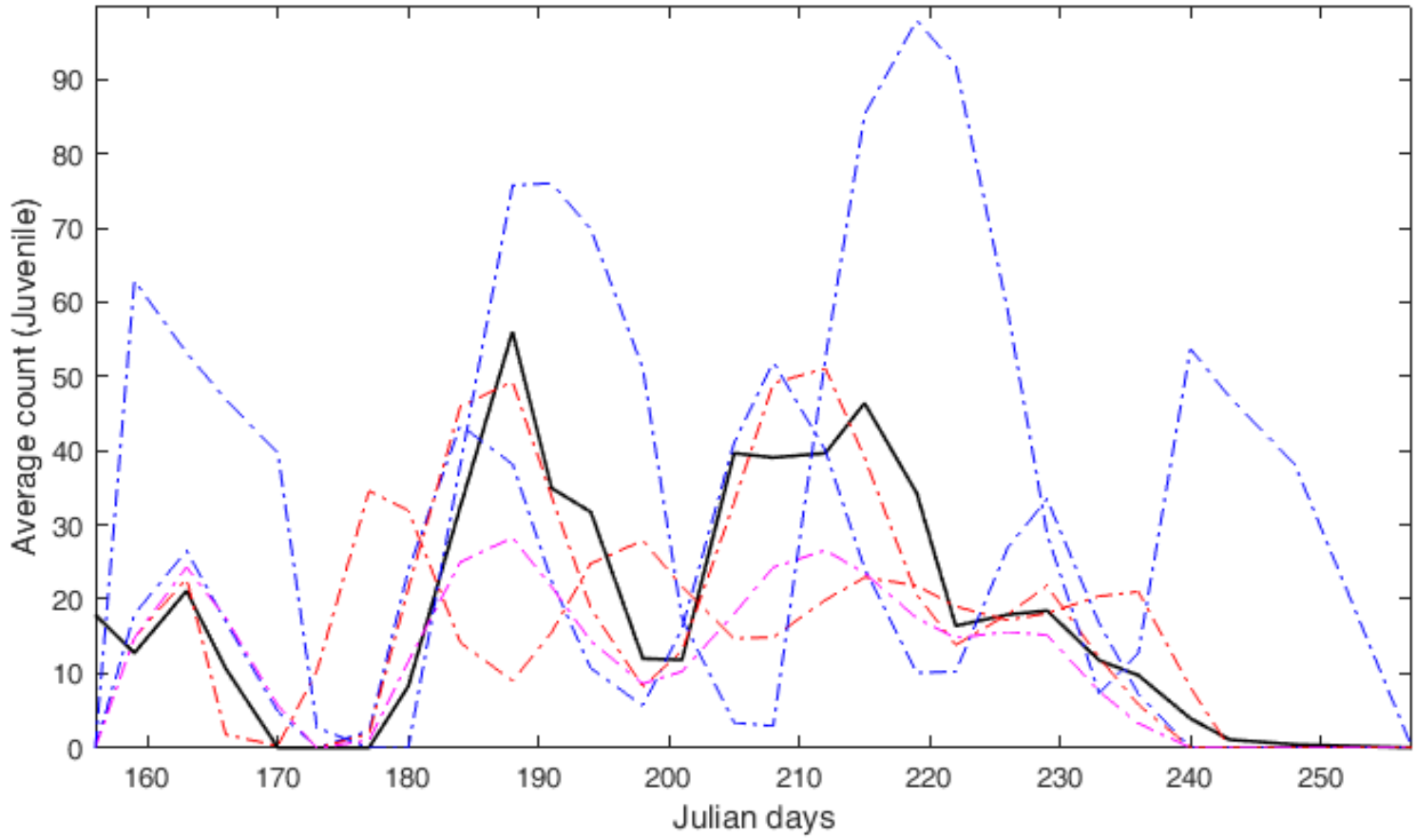
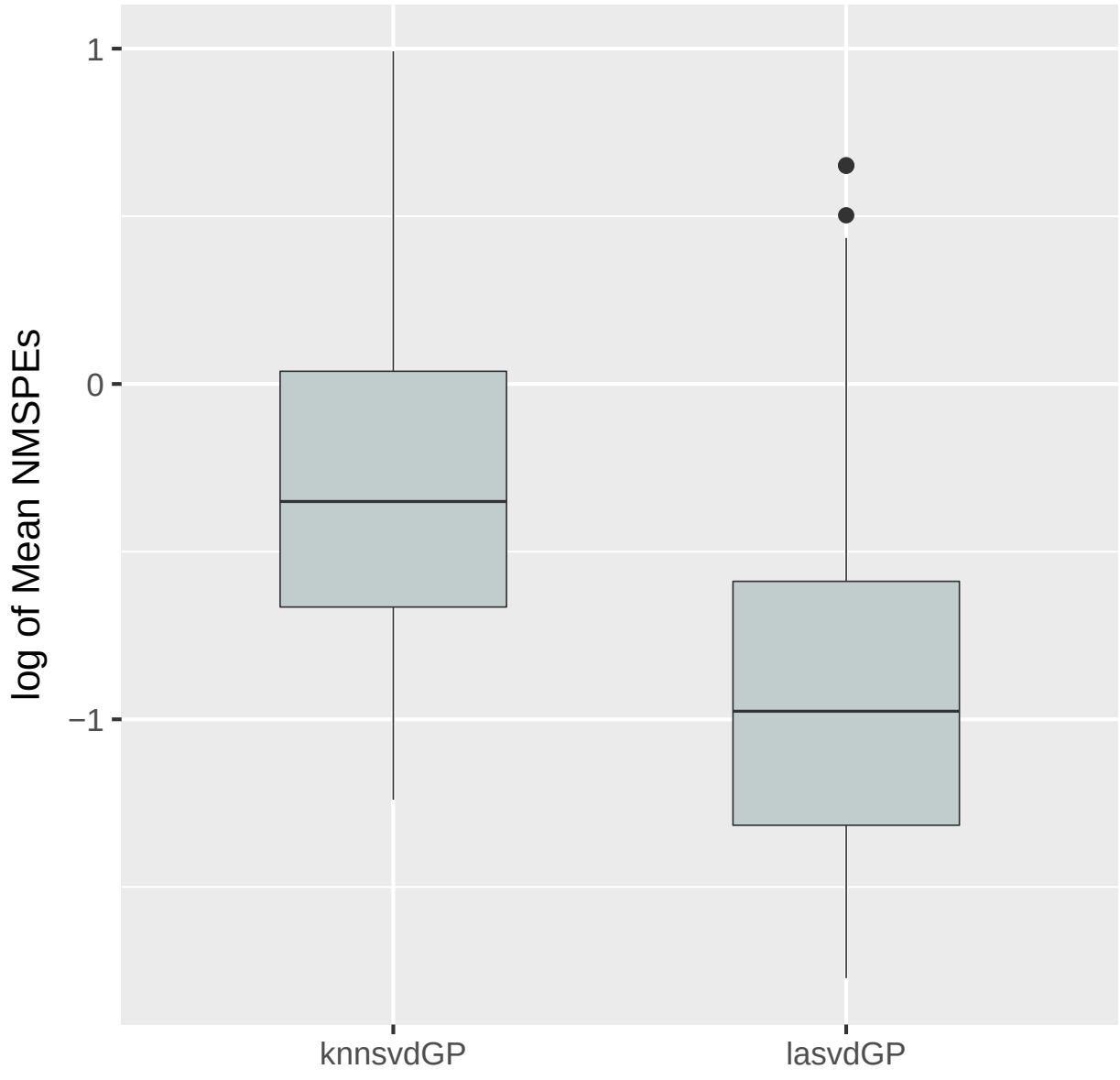


Figure: Juvenile ERM population dynamics as outputs of the TDB model at five different inputs. Black solid curve shows the field data and the other curves show the TDB outputs.

Example 3 (TDB simulator)



Inverse Problem

- Take a q-dimensional input $\mathbf{x}$ and return a time series response $\mathbf{y}(\mathbf{x}) = \{y(\mathbf{x}, t), t = 1, \ldots, L\}$ and consider a target field observation $\mathbf{z}_0$

Inverse Problem

- Take a q-dimensional input $\mathbf{x}$ and return a time series response $\mathbf{y}(\mathbf{x}) = \{y(\mathbf{x}, t), t = 1, \ldots, L\}$ and consider a target field observation $\mathbf{z}_0$
- The inverse problem refers to searching the input $\mathbf{x}$ such that the computer simulator $y(\mathbf{x}, t) \approx z_0(t)$ for all $t = 1, \ldots, L$.

Inverse Problem

- Take a q-dimensional input $\mathbf{x}$ and return a time series response $\mathbf{y}(\mathbf{x}) = \{y(\mathbf{x}, t), t = 1, \ldots, L\}$ and consider a target field observation $\mathbf{z}_0$
- The inverse problem refers to searching the input $\mathbf{x}$ such that the computer simulator $y(\mathbf{x}, t) \approx z_0(t)$ for all $t = 1, \ldots, L$.
- Consider the following model, for $t = 0.5, 0.52, \ldots, 2.5$ and $x \in [0, 1]$,

$$y(x, t) = \frac{sin(10\pi t)}{2t} + |t - 1|^{(2+4x)}, \tag{19}$$

and let the true field data correspond to $x_0 = 0.5$.

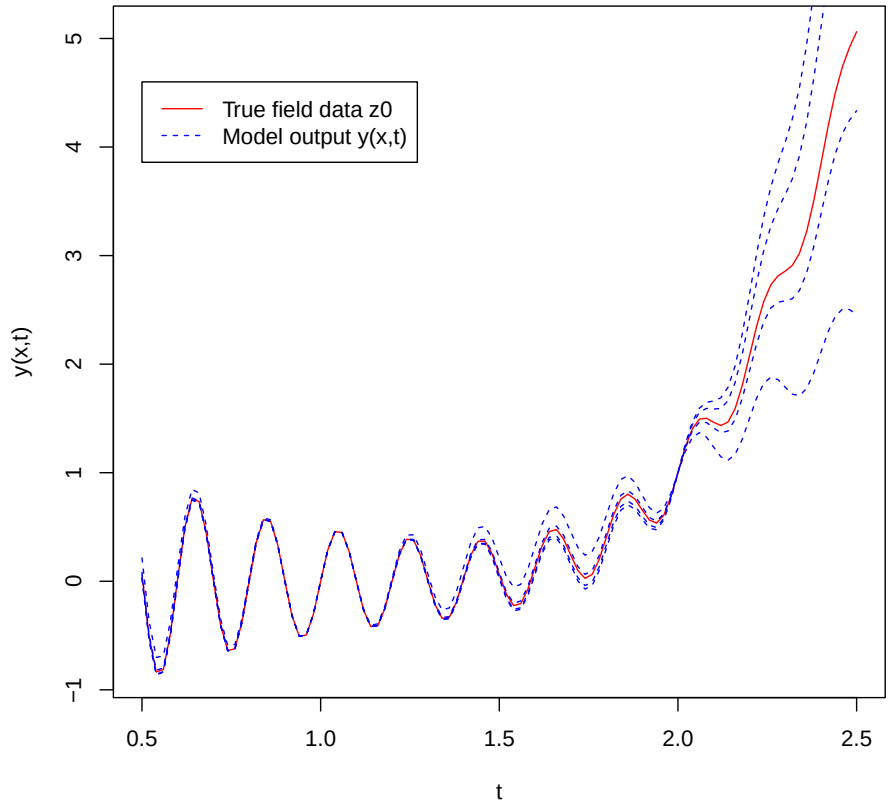


Figure: A few computer model outputs and the true field data for the computer simulator (19).

Existing Methods

- Scalarization by likelihood ratio statistic (Pratola et al., 2013)

$$\Delta(\mathbf{t}_i) = -2\log\left(\frac{L_r^*(\delta(\mathbf{t}_i))}{L_u^*(\delta(\mathbf{t}_i))}\right), \tag{20}$$

where $\delta(\mathbf{t}_i) = \mathbf{Y}^f - \mathbf{Y}_i^c$, $\mathbf{Y}^f = \eta(\theta) + \varepsilon$, and $\mathbf{Y}_i^c = \eta(\mathbf{t}_i)$, $\mathbf{t}_i$ is the ith input setting of calibration parameters.

Existing Methods

- Scalarization by likelihood ratio statistic (Pratola et al., 2013)

$$\Delta(\mathbf{t}_i) = -2\log\left(\frac{L_r^*(\delta(\mathbf{t}_i))}{L_u^*(\delta(\mathbf{t}_i))}\right), \tag{20}$$

  where $\delta(\mathbf{t}_i) = \mathbf{Y}^f - \mathbf{Y}_i^c$, $\mathbf{Y}^f = \eta(\theta) + \varepsilon$, and $\mathbf{Y}_i^c = \eta(\mathbf{t}_i)$, $\mathbf{t}_i$ is the ith input setting of calibration parameters.

- Scalarization by L_2 discrepancy (Ranjan et al., 2016).

$$w(\mathbf{x}) = ||\mathbf{y}(\mathbf{x}) - \mathbf{z}_0|| = \sqrt{\frac{1}{L}\sum_{t=1}^{L} |\mathbf{y}(\mathbf{x}, t) - \mathbf{z}_0(t)|^2} \tag{21}$$

  and find the minimum of $w(\mathbf{x})$.

Proposed Approach

- Fit an SVD-based GP model to the training data $\mathbf{Y} = \mathbf{UDV}^T$

Proposed Approach

- Fit an SVD-based GP model to the training data $\mathbf{Y} = \mathbf{UDV}^T$
- Estimate the input $\mathbf{x}^*$ for the inverse problem by minimizing the expected square L_2 discrepancy, that is,

$$\begin{aligned}
\hat{\mathbf{x}}^* &= \arg\min_{\mathbf{x}} E[||\mathbf{z}_0 - \mathbf{y}(\mathbf{x})||_2^2 | \mathbf{z}_0, \mathbf{Y}] \\
&= \arg\min_{\mathbf{x}} \sum_{i=1}^{p} d_i^2 [(\hat{c}_i(\mathbf{x}|\mathbf{v}_i, \hat{\theta}_i) - \hat{c}_{z_o,i})^2] + \sum_{i=1}^{p} d_i^2 [\hat{\sigma}_i^2(\mathbf{x}|\mathbf{v}_i, \hat{\theta}_i)].
\end{aligned}$$

Proposed Approach

- Fit an SVD-based GP model to the training data $\mathbf{Y} = \mathbf{UDV}^T$
- Estimate the input $\mathbf{x}^*$ for the inverse problem by minimizing the expected square L_2 discrepancy, that is,

$$\begin{aligned}\hat{\mathbf{x}}^* &= \arg\min_{\mathbf{x}} E[||\mathbf{z}_0 - \mathbf{y}(\mathbf{x})||_2^2 | \mathbf{z}_0, \mathbf{Y}] \\ &= \arg\min_{\mathbf{x}} \sum_{i=1}^{p} d_i^2 [(\hat{c}_i(\mathbf{x}|\mathbf{v}_i, \hat{\theta}_i) - \hat{c}_{z_o,i})^2] + \sum_{i=1}^{p} d_i^2 [\hat{\sigma}_i^2(\mathbf{x}|\mathbf{v}_i, \hat{\theta}_i)].\end{aligned}$$

- The first term represents the L_2 distance between the posterior mean of the simulator response at $\mathbf{x}$ and the projection of $\mathbf{z}_0$ on $\mathcal{M}(\mathbf{B})$ with $\mathbf{B} = \mathbf{U}^*\mathbf{D}^*$; the second term represents the model uncertainty at $\mathbf{x}$.

Sequential Designs

Choose the follow-up design points by sequentially maximizing the criterion

$$\begin{aligned} \mathbf{x}_{new} &= \underset{\mathbf{x}}{\operatorname{argmax}}\, J_l(\mathbf{x}) \\ &= \underset{\mathbf{x}}{\operatorname{argmax}} \sum_{i=1}^{p} d_i^2 [\xi \hat{\sigma}_i^2(\mathbf{x}|\mathbf{v}_i, \hat{\theta}_i) - (\hat{c}_i(\mathbf{x}|\mathbf{v}_i, \hat{\theta}_i) - \hat{c}_{\mathbf{z}_0,i})^2] \end{aligned}$$

where ξ is a tuning parameter which controls the emphasis on reducing the model uncertainty.

Simulation Study

- Consider the inverse problem using one-shot designs and sequential designs

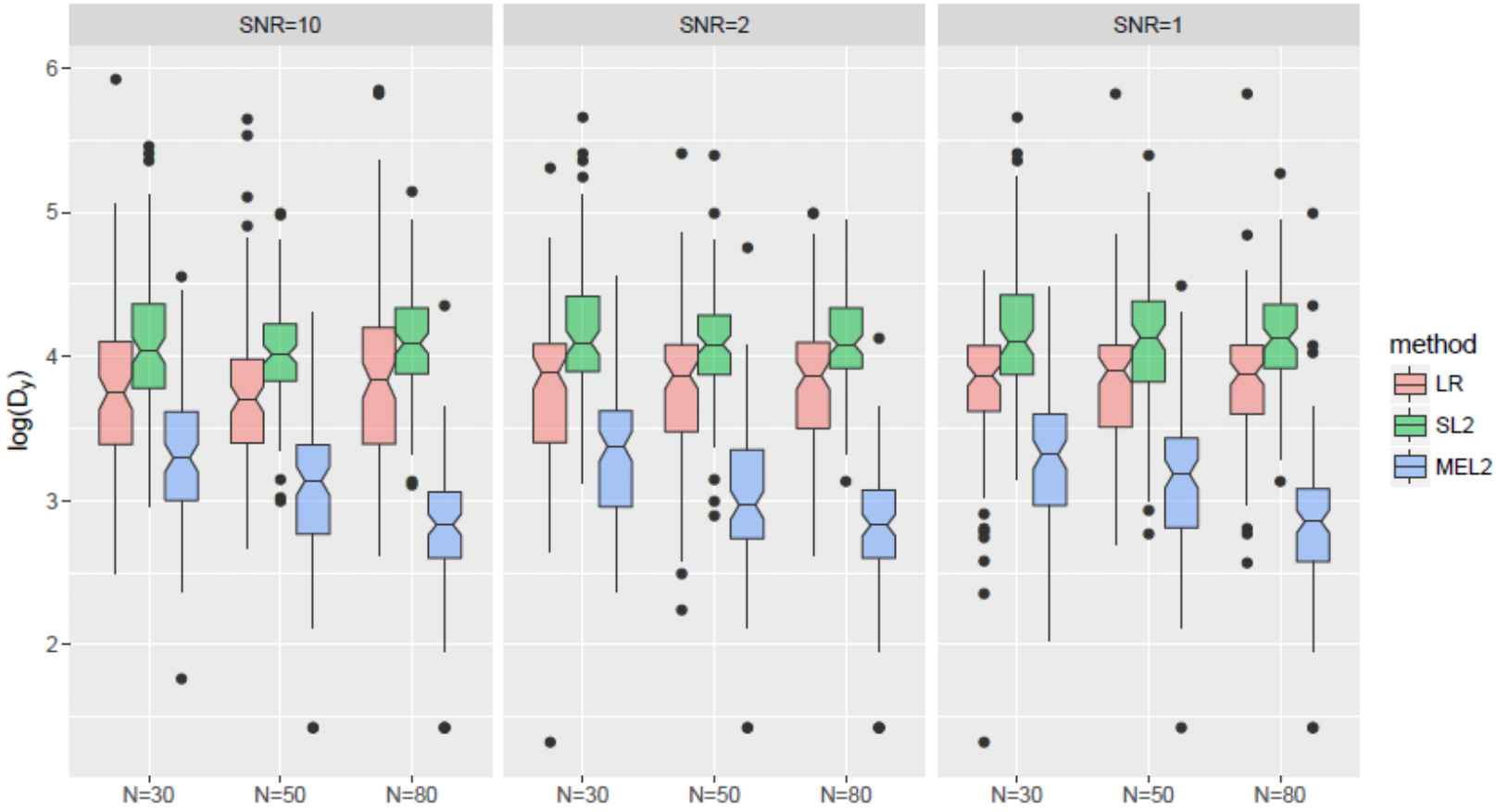
Simulation Study

- Consider the inverse problem using one-shot designs and sequential designs
- Compare LR (Pratola et al.,2013), SL2 (Ranjan et al., 2016), and MEL2 (the proposed approach)

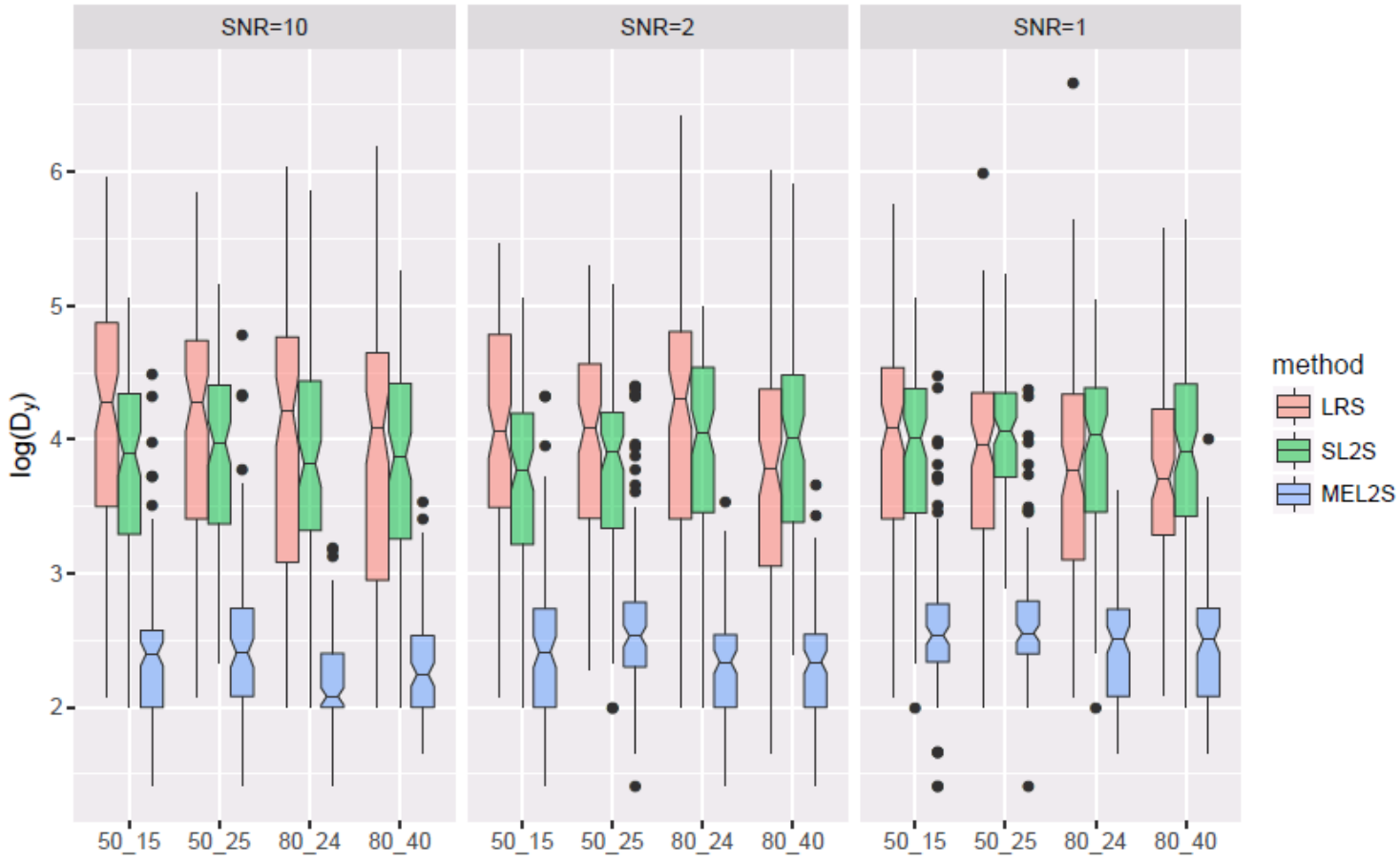
Simulation Study

- Consider the inverse problem using one-shot designs and sequential designs
- Compare LR (Pratola et al.,2013), SL2 (Ranjan et al., 2016), and MEL2 (the proposed approach)
- For a pre-specified $\mathbf{x}^*$, compare using $D_{\mathbf{y}} = ||\mathbf{y}(\hat{\mathbf{x}}^*) - \mathbf{y}(\mathbf{x}^*)||_2$.

Consider again the environmental model studied by Bliznyuk et al. (2008).



Consider again the environmental model studied by Bliznyuk et al. (2008).



Concluding Remarks

- Proposed greedy local SVD-based GP models for large-scale dynamic computer experiments

Concluding Remarks

- Proposed greedy local SVD-based GP models for large-scale dynamic computer experiments
- Proposed a new design criterion to determine local neighbour sequentially

Concluding Remarks

- Proposed greedy local SVD-based GP models for large-scale dynamic computer experiments
- Proposed a new design criterion to determine local neighbour sequentially
- Introduced a new design criterion to solve the inverse problem in dynamic computer experiments

Thank you! Q&A.