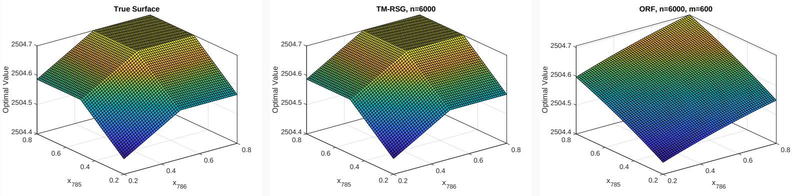
Sample and Computationally Efficient Simulation Metamodeling in High Dimensions

Xiaowei Zhang (HKU Business School)

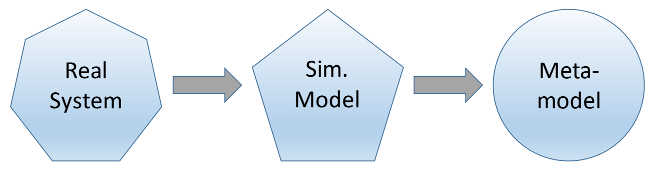
December 5, 2020

Joint work with Liang Ding (Texas A&M University)

Surface Fitting in 816 Dimensions



Metamodeling



- Simulation models are often computationally expensive
- Metamodel: statistical model for simulation input-output relationship
    - A.K.A. surrogate model
    - Run simulation at a small number of design points
    - Predict responses with the fitted statistical model

Comparison with Regression

	Metamodeling	Regression
Statistical Model	Linear/Nonparametric	Linear/Nonparametric
Typical Design	Fixed	Random
Typical Noise	Heteroscedastic	Homoscedastic
Replications	Yes	No

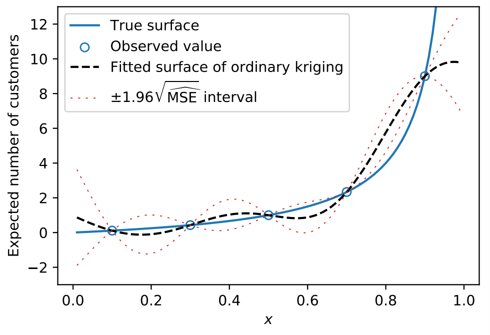
Stochastic Kriging

- Gaussian process regression
- Response surface is a sample path of a GP with kernel $k(\boldsymbol{x}, \boldsymbol{x}')$

$$\mathsf{Y}(\cdot) \sim \mathsf{GP}(0, k(\cdot, \cdot))$$

- Take samples at design points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$
- SK predictor

$$\hat{\mathsf{Y}}_n(\boldsymbol{x}) = \boldsymbol{k}^\intercal(\boldsymbol{x})(\boldsymbol{K} + \Sigma)^{-1}\bar{\boldsymbol{Y}}$$



Critical Specifications

The prediction accuracy of SK depends on

(i) choice of kernel

(ii) choice of experimental design

Typical Kernels

- Gaussian kernel

$$k_{\text{Gauss}}(\boldsymbol{x}, \boldsymbol{x}') := \exp\left(-\left\|\boldsymbol{\theta}^\intercal(\boldsymbol{x} - \boldsymbol{x}')\right\|^2\right)$$

- Matérn kernel

$$k_{\text{Matérn}(\nu)}(\boldsymbol{x}, \boldsymbol{x}') := \frac{1}{2^{\nu-1}\Gamma(\nu)} \left(\sqrt{2\nu}\left\|\boldsymbol{\theta}^\intercal(\boldsymbol{x} - \boldsymbol{x}')\right\|\right)^\nu K_\nu\left(\sqrt{2\nu}\left\|\boldsymbol{\theta}^\intercal(\boldsymbol{x} - \boldsymbol{x}')\right\|\right)$$

Typical Kernels

- Gaussian kernel

$$k_{\mathrm{Gauss}}(\boldsymbol{x}, \boldsymbol{x}') := \exp\left(-\left\|\boldsymbol{\theta}^\intercal(\boldsymbol{x} - \boldsymbol{x}')\right\|^2\right)$$

- Matérn kernel

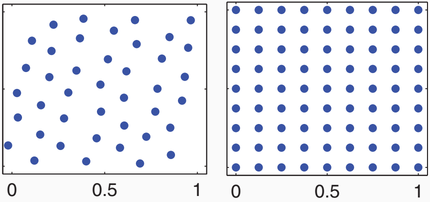
$$k_{\mathrm{Matérn}(\nu)}(\boldsymbol{x}, \boldsymbol{x}') := \frac{1}{2^{\nu-1}\Gamma(\nu)} \left(\sqrt{2\nu}\left\|\boldsymbol{\theta}^\intercal(\boldsymbol{x} - \boldsymbol{x}')\right\|\right)^{\nu} K_\nu\left(\sqrt{2\nu}\left\|\boldsymbol{\theta}^\intercal(\boldsymbol{x} - \boldsymbol{x}')\right\|\right)$$

- Generalized integrated Brownian field (Salemi et al. 2019, *OR*)
    - Motivation: model smoothness separately for each dimension

$$k(\boldsymbol{x}, \boldsymbol{x}') = \prod_{j=1}^{d} k_j(x_j, x_j')$$

    - On dimension j is a Brownian motion integrated ℓ_j times

Typical Experimental Designs



- Latin hypercube design (LHD)
  - Pros: ease of use; 1-d projections onto are evenly dispersed
  - Con: superiority only holds for large sample sizes when d is large
- Lattice design: Cartesian product of one-dimensional designs
  - Pro: $\boldsymbol{K}^{-1} = \bigotimes_{j=1}^{d} \boldsymbol{K}_j^{-1}$ for tensor-product kernels
  - Con: excessive sample size when design space when d is large

Curse of Dimensionality

- Sample complexity: n grows exponentially in d
- Computational complexity: $\mathcal{O}(n^3)$

Curse of Dimensionality: Sample Complexity

- Sample paths of a GP with Matérn(α) kernel form a *Sobolev space*
  - α represents the smoothness
- Minimax-optimal rate for estimating Y via noisy samples: $n^{-\alpha/(2\alpha+d)}$
- So, sample complexity for achieving an δ-error is $\delta^{-(2+d/\alpha)}$
- Practical situation could be even worse due to model misspecification
  - SK is specified with Matérn(ν) kernel
  - Convergence rate: $\mathcal{O}(n^{-\min(\alpha,\nu)/(2\nu+d)})$

Combact the Curse with Smoothness?

- Given a large *d*, the convergence rate is fast with a large α
- Why don't we set/assume $\alpha = \infty$?
  - Infinitely differentiable response surface is rare
  - E.g., the max function often appears in queueing, inventory, FE models

Curse of Dimensionality: Computational Complexity

- Computing $(\boldsymbol{K} + \Sigma)^{-1}$ requires $\mathcal{O}(n^3)$
- Subsampling to construct a low-rank approximation: $\mathcal{O}(\ell^2 n)$
- Lu et al. (2020, *OR*), but huge literature in machine learning and statistics

This Work

- "New" kernel: Tensor Markov (TM) kernels
- New exponential design: Random sparse grid (RSG) designs
- Convergence rate: "weakly" dependent of d
- Fast, exact computation

TM Kernels

- Tensor-product form: $k(\boldsymbol{x}, \boldsymbol{x}') = \prod_{j=1}^{d} k_j(x_j, x_j')$
- Each k_j corresponds to a Gauss-Markov process
  - Brownian motion: $k(x, x') = x \wedge x'$
  - Stationary OU process: $k(x, x') = \exp(-\theta|x - x'|)$

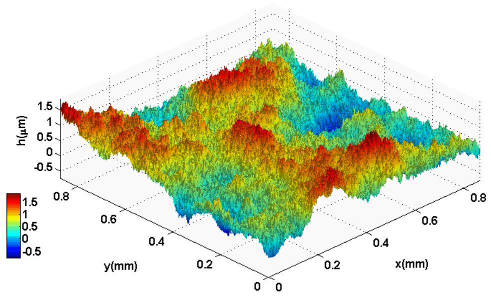
Markov Properties

- Facilitate computation
- One-dimensional case: $\boldsymbol{K}^{-1}$ is tri-diagonal
  - Assume $\mathcal{X} = [0, 1]$ and $x_i = \frac{i}{n+1}$, $i = 1, \ldots, n$
  - Brownian motion: $k(x, y) = \min(x, y)$

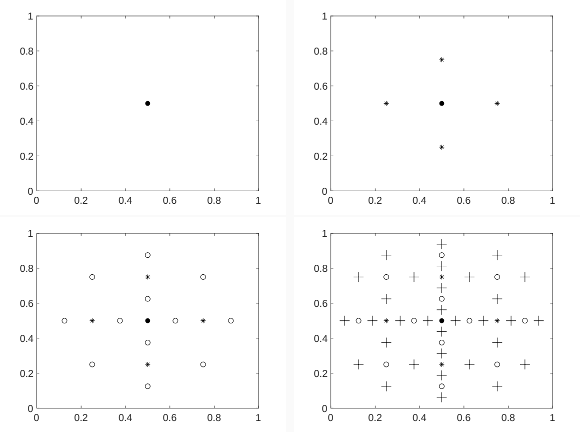
$$\boldsymbol{K}^{-1} = (n+1) \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{pmatrix}$$

Nowhere-differentiable Sample Paths

- Time-changed Brownian field



Classical Sparse Grids



- Controlled by a level parameter τ

Dimension d	Full Grid	Sparse Grid of Level 4
1	15	15
2	225	49
5	759,375	351
10	5.77×10^{11}	2,001
20	3.33×10^{23}	13,201
50	6.38×10^{58}	182,001

Pro and Con of Classical Sparse Grids

- Pro: Fast computation of $\boldsymbol{K}^{-1}$ for tensor-product kernels (Plumlee, 2014, *JASA*)

Pro and Con of Classical Sparse Grids

- Pro: Fast computation of $\boldsymbol{K}^{-1}$ for tensor-product kernels (Plumlee, 2014, *JASA*)

Level τ	$d = 2$	$d = 5$	$d = 10$	$d = 20$	$d = 50$
2	5	11	21	41	101
3	17	71	241	881	5,201
4	49	351	2,001	13,201	182,001
5	129	1,471	13,441	154,881	4,867,201

- Con: Inflexible to use, fast computation only available for complete SGs
  - n must be coincide with the sample size of some level τ

Random Sparse Grids

- Find τ such that n falls between $\mathcal{X}_\tau^{\mathsf{SG}}$ and $\mathcal{X}_{\tau+1}^{\mathsf{SG}}$
- Random sampling on $\mathcal{X}_{\tau+1}^{\mathsf{SG}} \setminus \mathcal{X}_\tau^{\mathsf{SG}}$
- $\mathcal{X}_n^{\mathsf{RSG}} := \mathcal{X}_\tau^{\mathsf{SG}} \cup \mathcal{A}$
- Fast computation of $\boldsymbol{K}^{-1}$ only for TM kernels

Convergence Rates: Kriging

- Model well-specified: true surface is a GP with tensor Markov kernel

$$\max_{\boldsymbol{x} \in [0,1]^d} \mathbb{E}[(\widehat{Y}_n(\boldsymbol{x}) - Y(\boldsymbol{x}))^2] = \mathcal{O}\left(n^{-1}(\log n)^{2(d-1)}\right)$$

Convergence Rates: Kriging

- Model well-specified: true surface is a GP with tensor Markov kernel

$$\max_{\boldsymbol{x}\in[0,1]^d} \mathbb{E}[(\widehat{\mathrm{Y}}_n(\boldsymbol{x}) - \mathrm{Y}(\boldsymbol{x}))^2] = \mathcal{O}\left(n^{-1}(\log n)^{2(d-1)}\right)$$

- If true surface is a GP with Matérn(α) and SK uses the same kernel, then

$$\max_{\boldsymbol{x}\in[0,1]^d} \mathbb{E}[(\widehat{\mathrm{Y}}_n(\boldsymbol{x}) - \mathrm{Y}(\boldsymbol{x}))^2] = \mathcal{O}\left(n^{-2\alpha/(2\alpha+d)}\right).$$

Model Mis-specification

- Smooth surface, rough kernel
- True surface is a GP with a tensor product kernel that is smoother

$$\max_{\boldsymbol{x}\in[0,1]^d} \mathbb{E}[(\widehat{\mathsf{Y}}_n^{\mathrm{mis}}(\boldsymbol{x}) - \mathsf{Y}^*(\boldsymbol{x}))^2] = \mathcal{O}\left(n^{-2}(\log n)^{3(d-1)}\right).$$

Convergence Rates: Stochastic Kriging

- Model well-specified:

$$\mathcal{O}\left(n^{-1}(\log n)^{2(d-1)} + (\log n)^d \max_{1\le i\le n} m_i^{-1/2}\sigma(\boldsymbol{x}_i)\right).$$

- Model mis-specified:

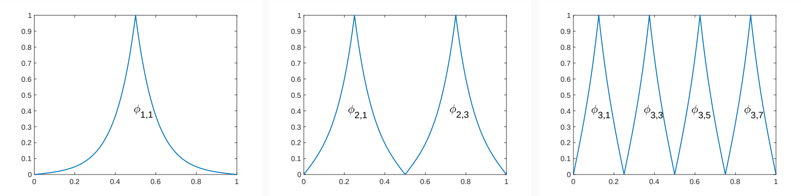
$$\mathcal{O}\left(n^{-2}(\log n)^{3(d-1)} + (\log n)^d \max_{1\le i\le n} m_i^{-1/2}\sigma(\boldsymbol{x}_i)\right).$$

Key for Proof: Orthogonal Expansions

- Expansion for TM kernels

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sum_{\tau=1}^{\infty} \sum_{(\boldsymbol{l},\boldsymbol{i}):\boldsymbol{c}_{\boldsymbol{l},\boldsymbol{i}} \in \mathcal{X}_{\tau}^{\mathrm{SG}}} \frac{\phi_{\boldsymbol{l},\boldsymbol{i}}(\boldsymbol{x})\phi_{\boldsymbol{l},\boldsymbol{i}}(\boldsymbol{x}')}{\|\phi_{\boldsymbol{l},\boldsymbol{i}}\|_{\mathcal{H}_k}^2}$$

- $\phi_{\boldsymbol{l},\boldsymbol{i}} \in [0,1]$
- $\|\phi_{\boldsymbol{l},\boldsymbol{i}}\|_{\mathcal{H}_k}^2 \asymp 2^{|\boldsymbol{l}|}$



- Expansion for GP with TM kernels

$$\mathsf{Y}(\boldsymbol{x}) = \sum_{\tau=1}^{\infty} \sum_{(\boldsymbol{l},\boldsymbol{i}):\boldsymbol{c}_{\boldsymbol{l},\boldsymbol{i}} \in \mathcal{X}_{\tau}^{\mathsf{SG}}} \frac{\phi_{\boldsymbol{l},\boldsymbol{i}}(\boldsymbol{x})}{\|\phi_{\boldsymbol{l},\boldsymbol{i}}\|_{\mathcal{H}_k}} Z_{\boldsymbol{l},\boldsymbol{i}}$$

- Observing $\{\mathsf{Y}(\boldsymbol{c}_{\boldsymbol{l},\boldsymbol{i}}) : \boldsymbol{c}_{\boldsymbol{l},\boldsymbol{i}} \in \mathcal{X}_{\tau}^{\mathsf{SG}}\}$ equals observing $\{Z_{\boldsymbol{l},\boldsymbol{i}} : \boldsymbol{c}_{\boldsymbol{l},\boldsymbol{i}} \in \mathcal{X}_{\tau}^{\mathsf{SG}}\}$

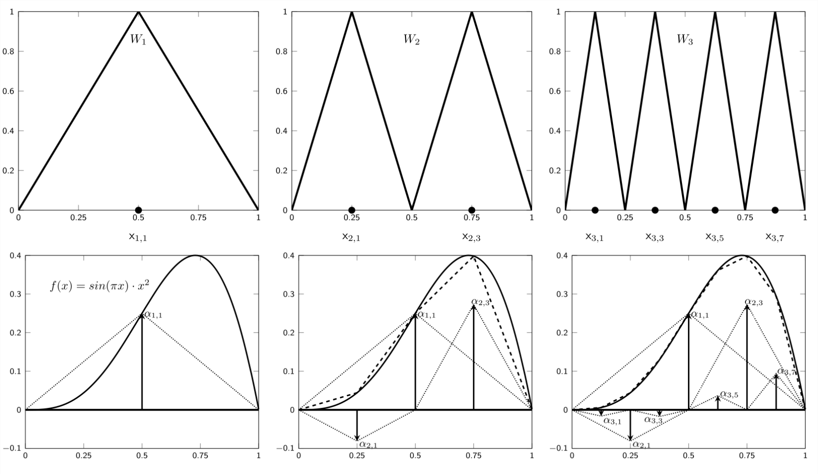


Figure 1: Brumm and Scheidegger (2017, *Econometrica*)

$$
\begin{aligned}
\mathbb{E}[(\hat{\mathsf{Y}}_n(\boldsymbol{x}) - \mathsf{Y}(\boldsymbol{x}))^2] &= \sum_{\tau=\ell+1}^{\infty} \sum_{(\boldsymbol{l},\boldsymbol{i}):\boldsymbol{c}_{\boldsymbol{l},\boldsymbol{i}} \in \mathcal{X}_{\tau}^{\mathsf{SG}}} \frac{\phi_{\boldsymbol{l},\boldsymbol{i}}^2(\boldsymbol{x})}{\|\phi_{\boldsymbol{l},\boldsymbol{i}}\|_{\mathcal{H}_k}^2} \\
&\leq \sum_{\tau=\ell+1}^{\infty} \sum_{(\boldsymbol{l},\boldsymbol{i}):\boldsymbol{c}_{\boldsymbol{l},\boldsymbol{i}} \in \mathcal{X}_{\tau}^{\mathsf{SG}}} \frac{1}{\|\phi_{\boldsymbol{l},\boldsymbol{i}}\|_{\mathcal{H}_k}^2} \\
&\asymp \sum_{|\boldsymbol{l}| > \ell+d-1} 2^{-|\boldsymbol{l}|} \\
&= \mathcal{O}(2^{-\ell} \ell^{d-1}) \\
&= \mathcal{O}\left(n^{-1} (\log n)^{2(d-1)}\right)
\end{aligned}
$$

Fast Computation

- $\boldsymbol{K}^{-1}$ can be expressed as

$$\boldsymbol{K}^{-1} = \begin{array}{c} \begin{array}{cc} |\mathcal{X}_\tau^{\mathsf{SG}}| \text{ dim.} & \quad \tilde{n} \text{ dim.} \end{array} \\ \begin{pmatrix} \square & \square \\ \square & \boldsymbol{D} \end{pmatrix} \end{array}$$

- Each block can be computed efficiently
- $\boldsymbol{K}^{-1}$ is sparse: Proportion of nonzero entries: $\mathcal{O}(n^{-1}(\log n)^{2d})$

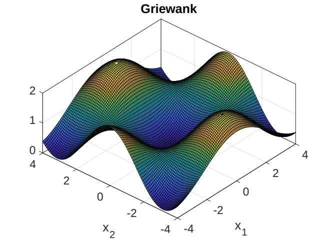
- $\boldsymbol{K}^{-1}$ can be expressed as

$$\boldsymbol{K}^{-1} = \begin{array}{c} \begin{array}{cc} |\mathcal{X}_\tau^{\mathsf{SG}}| \text{ dim.} & \tilde{n} \text{ dim.} \end{array} \\ \begin{pmatrix} \square & \square \\ \square & \boldsymbol{D} \end{pmatrix} \end{array}$$

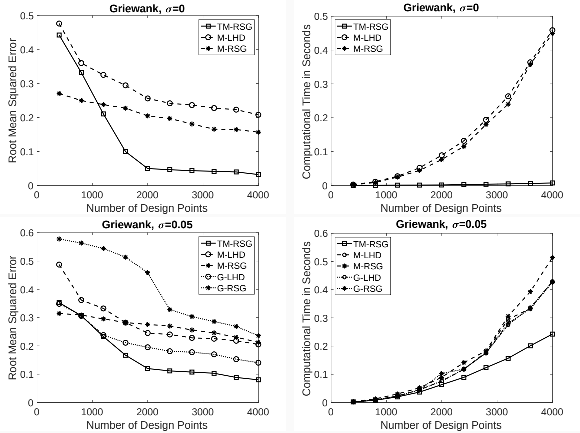
- Each block can be computed efficiently
- $\boldsymbol{K}^{-1}$ is sparse: Proportion of nonzero entries: $\mathcal{O}(n^{-1}(\log n)^{2d})$
- Compting stochastic kriging: Woodbury matrix identity

$$(\boldsymbol{K} + \Sigma)^{-1} = \Sigma^{-1} - \Sigma^{-1}(\boldsymbol{K}^{-1} + \Sigma^{-1})^{-1}\Sigma^{-1},$$

Griewank Function in 10 Dimensions



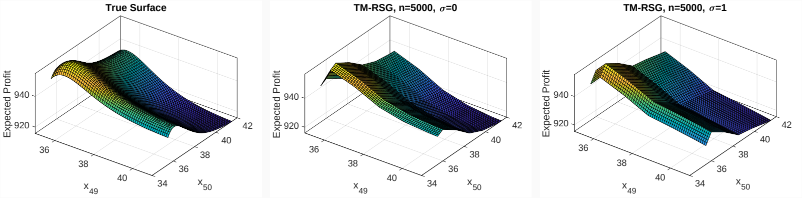
$$y_{\text{Griewank}}(\boldsymbol{x}) = \sum_{j=1}^{d} \frac{x_j^2}{4000} - \prod_{j=1}^{d} \cos\left(\frac{x_j}{\sqrt{t}}\right) + 1, \quad \boldsymbol{x} \in [-4, 4]^d,\ d = 10$$

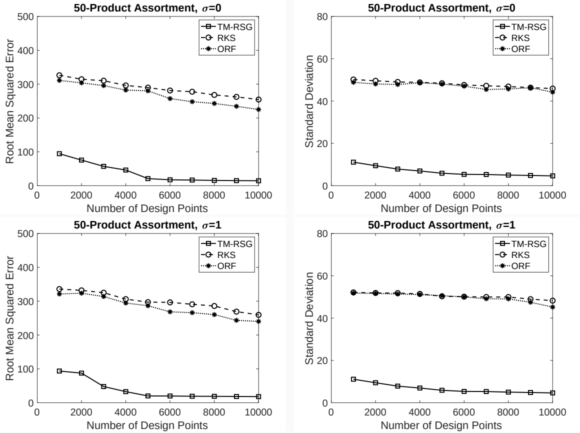

Griewank, σ=0
Root Mean Squared Error
Number of Design Points
TM-RSG
M-LHD
M-RSG
Griewank, σ=0
Computational Time in Seconds
Number of Design Points
TM-RSG
M-LHD
M-RSG
Griewank, σ=0.05
Root Mean Squared Error
Number of Design Points
TM-RSG
M-LHD
M-RSG
G-LHD
G-RSG
Griewank, σ=0.05
Computational Time in Seconds
Number of Design Points
TM-RSG
M-LHD
M-RSG
G-LHD
G-RSG

A Product Assortment Problem

- Aydin and Porteus (2008, *OR*)
- Design variable ($d = 50$): price vector
- Response surface: expected profit

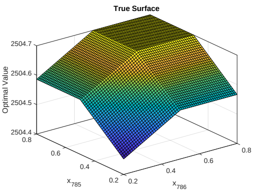
$$y(\boldsymbol{x}) = \frac{1}{2(b-a)} \sum_{j=1}^{d} \left[(b-a) \left(\frac{x_j - c_j}{x_j} \right) + a \right]^2 Q_j^2(\boldsymbol{x})$$

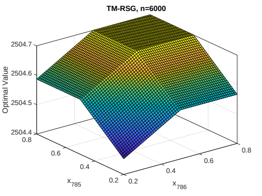


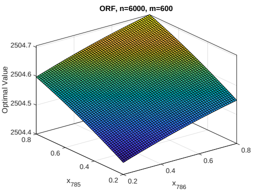

50-Product Assortment, σ=0
Root Mean Squared Error
Number of Design Points
TM-RSG
RKS
ORF
50-Product Assortment, σ=0
Standard Deviation
Number of Design Points
TM-RSG
RKS
ORF
50-Product Assortment, σ=1
Root Mean Squared Error
Number of Design Points
TM-RSG
RKS
ORF
50-Product Assortment, σ=1
Standard Deviation
Number of Design Points
TM-RSG
RKS
ORF

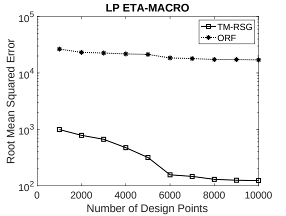
A Large Linear Program

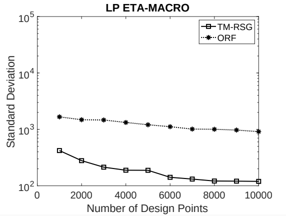
- Decision variable ($d = 816$): coefficient vector of the objective function
- Response surface: optimal value











Summary

- Curse of dimensionality
- Tensor Markov kernels
- Random sparse grids
- **Sample** efficiency: convergence rate suffers little from curse of dimensionality
- **Computational** efficiency: exact computation from sparse structure of $\boldsymbol{K}^{-1}$