Gaussian Processes and Bayesian Optimization

Rui Tuo

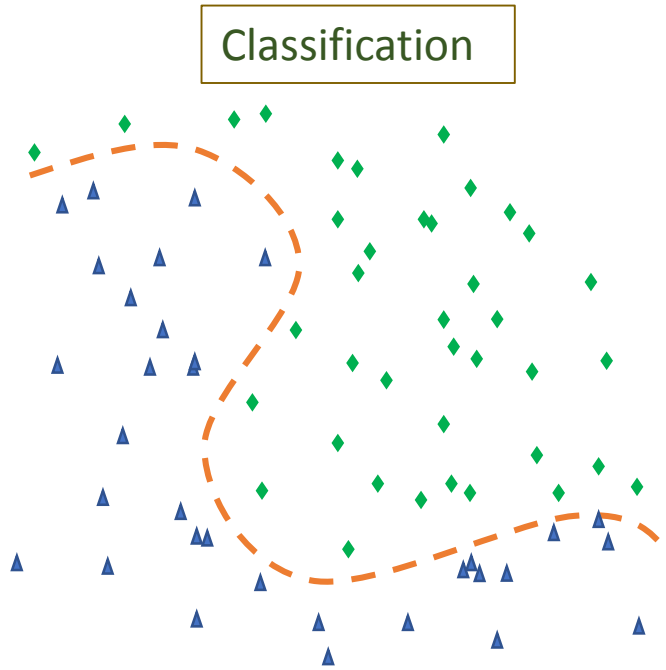
Wm Michael Barnes '64 Department of Industrial & Systems Engineering

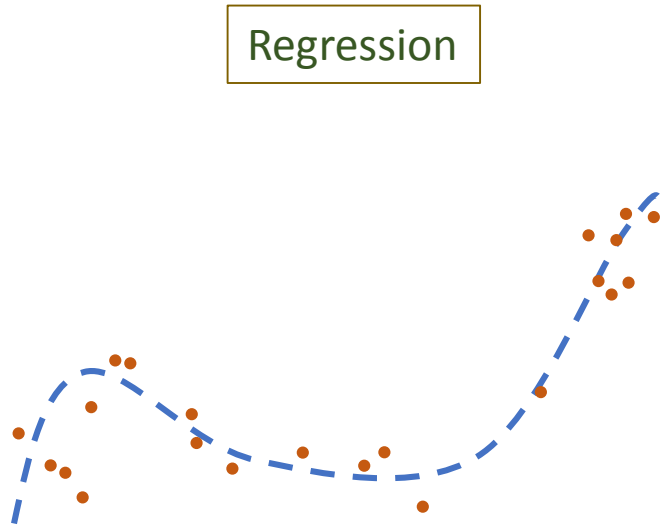
Texas A&M University

I. Gaussian process regression
II. Design of experiments for GP models
III. Nonstationary GP models in computer experiments
IV. Bayesian optimization

September 4th, 2020
TAMIDS, Texas A&M University

Supervised learning



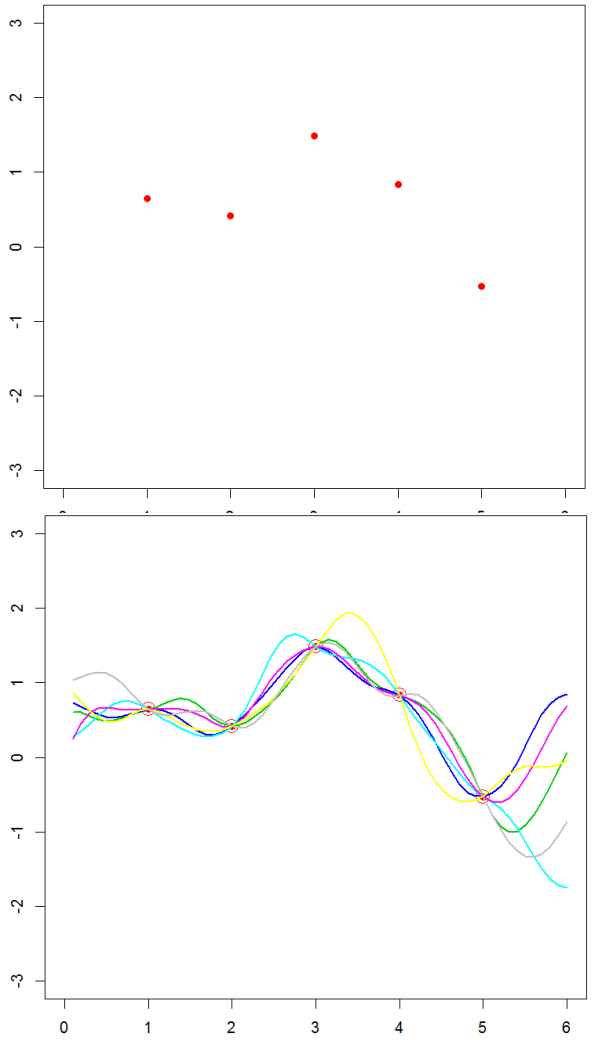


- In supervised learning (e.g., classification and regression) we want to find the underlying function (dashed curves) that represents data.
- How to represent a general function?

A Motivational Example: global optimization

- Global optimization for complex functions
  - Only limited evaluations are available.
- Problem: find x_0 such that
$$f(x_0) = \max_x f(x).$$
- Applications:
  - Engineering designs
  - Parameter calibration for FEA models
  - Optimal tuning for deep neural networks
- Challenge

  No information for untried points!!



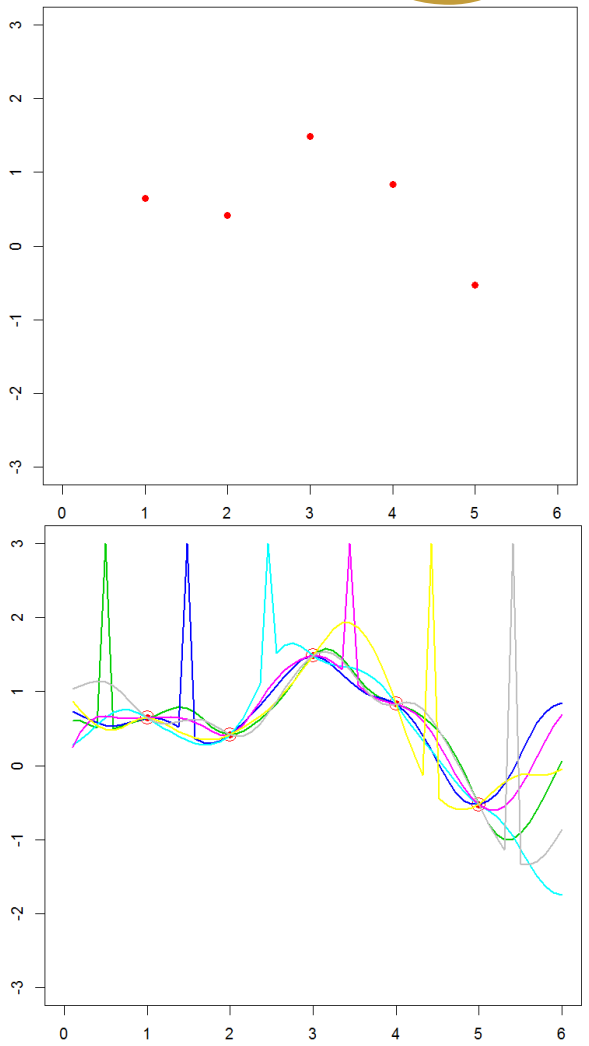
A Motivational Example: global optimization

- Global optimization for complex functions
  - Only limited evaluations are available.
- Problem: find x_0 such that
$$f(x_0) = \max_x f(x).$$
- Applications:
  - Engineering designs
  - Parameter calibration for FEA models
  - Optimal tuning for deep neural network
- Challenge

  No information for untried points!!

Q: Where is the problem?

A: Function space too large.

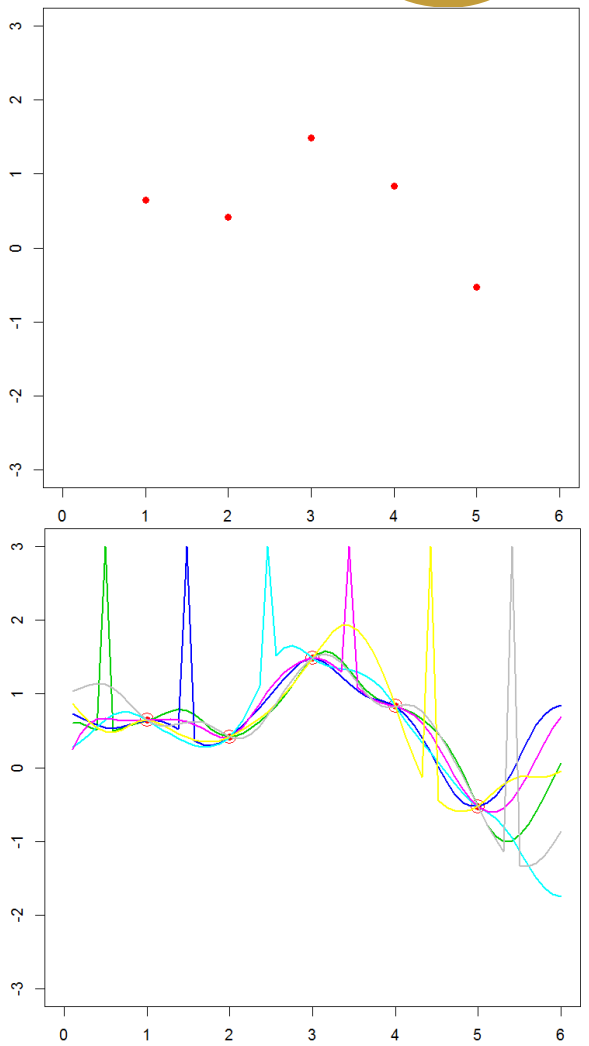


A Motivational Example: global optimization

- Global optimization for complex functions
  - Only limited evaluations are available.
- Problem: find x_0 such that
$$f(x_0) = \max_x f(x).$$
- Applications:
  - Engineering designs
  - Parameter calibration for FEA models
  - Optimal tuning for deep neural network
- Challenge

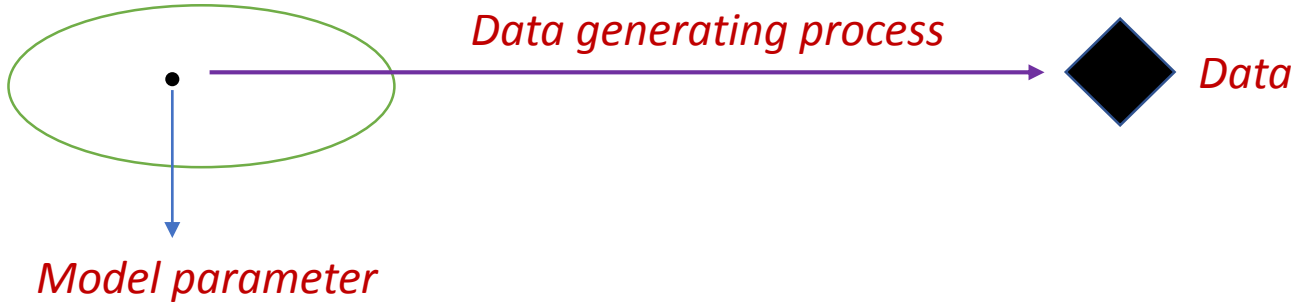
  No information for untried points!!

Solution: Restrict the functions of interest!

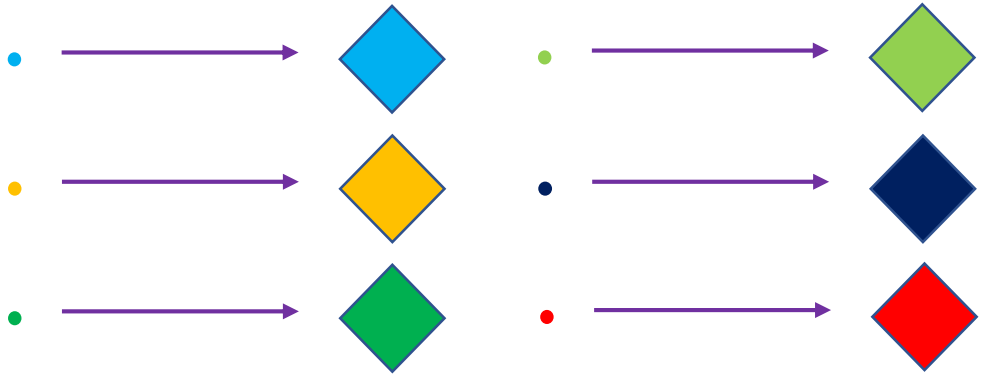


A Paradigm of Statistics

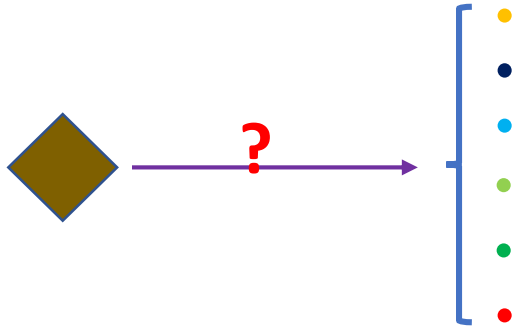
- Statistical model



- Forward problems. *Not statistics*

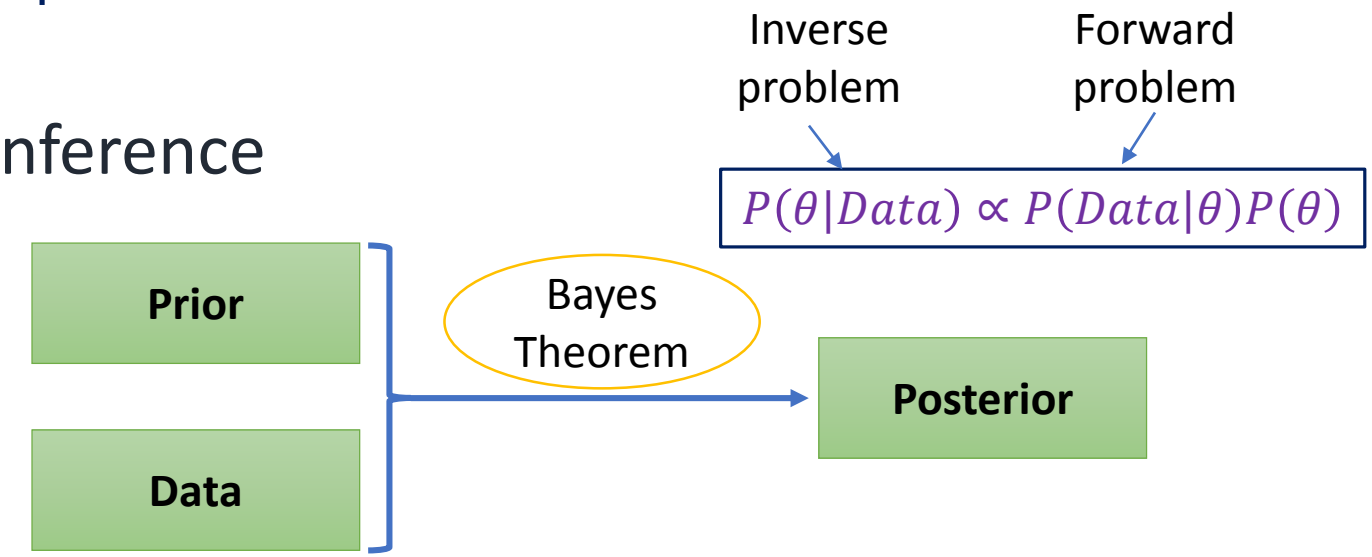


- Inverse problems. **This is statistics!**



Bayesian Nonparametrics

- Bayesian inference



- "Parametric" Bayes
  - Number of parameters is finite.
  - The prior is a distribution in a finite-dimensional space.
- Nonparametric Bayes
  - The unknown is a function (that is infinite dimensional).
  - The prior is a stochastic process.

Stochastic processes

- Rolling a die to get a number.
  - The outcome of a dice rolling is a *random number.*

- A stochastic process Z is a *random function.*
  - Each realization (a.k.a. sample path) of Z is a deterministic function.
  - Given x, $Z(x)$ is a random variable.
  - Here x is a d-dimensional vector.

Gaussian processes

- Ideal priors for continuous functions.
- To define a Gaussian process, we need:
  - Mean function $m(x)$.
  - Covariance function $C(x_1, x_2)$.
  - Denoted as $GP(m, C)$.
- $GP(m, C)$ has continuous sample paths if m and C are continuous.
- A GP with $m = 0$ is called *centered*.
- *Stationary* Gaussian processes
  - GP is centered and $C(x_1, x_2) = K(x_1 - x_2)$.
  - Probability structure is invariant in translation.
- Stationary GPs are commonly used priors. Why?

Correlation functions

- For stationary GPs, we parametrize

$$C(x_1, x_2) = \sigma^2 \Phi(x_1 - x_2),$$

with $\Phi(0) = 1$.

σ^2 is called the variance; Φ is called the correlation function.

- Commonly used correlation functions in 1D

➢ Gaussian correlation family

$$\Phi(x; \theta) = \exp\{-(\theta x)^2\}.$$

➢ θ is a scale parameter.

➢ Sample paths are infinitely differentiable.

➢ Matérn correlation family

$$\Phi(x; \theta, \nu) \propto |\theta x|^{\nu} K_{\nu}(2\sqrt{\nu}|\theta x|).$$

➢ K_{ν} is the modified Bessel function of the second kind.

➢ θ is a scale parameter.

➢ ν is the "smoothness parameter".

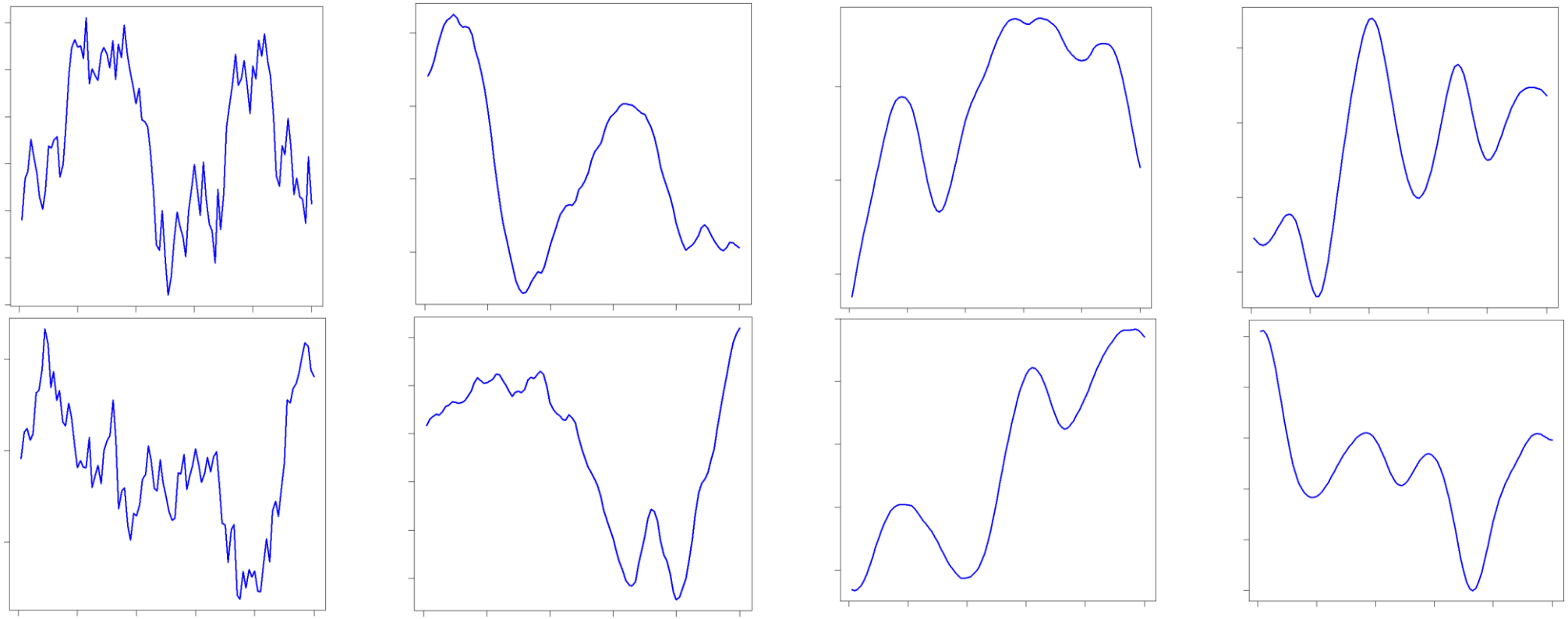
➢ The sample path smoothness is governed by ν.

Sample path comparison

Matérn

Gaussian

$\nu = 0.5$ $\nu = 1.5$ $\nu = 2.5$ $(\nu = \infty)$



Multi-dimensional correlation functions

- Two common strategies to construct d-dimensional correlations

1. Isotropic correlation:

$$\Phi(x) = \Phi_1(\|x\|),$$

where Φ_1 is a 1D Gaussian or Matérn correlation; $\|x\|$ is the Euclidean norm.

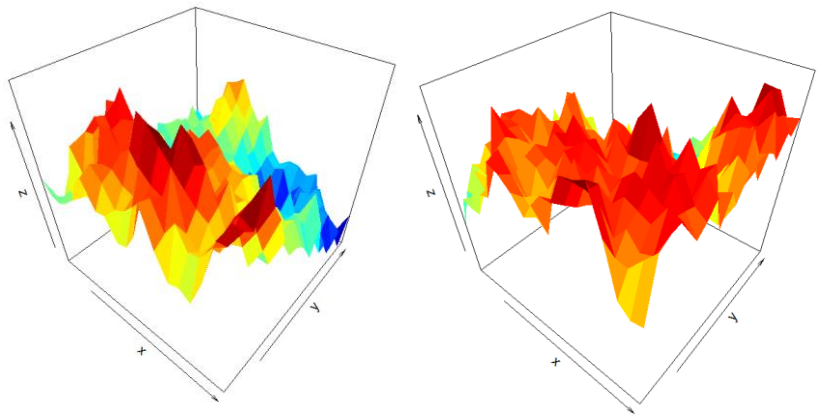
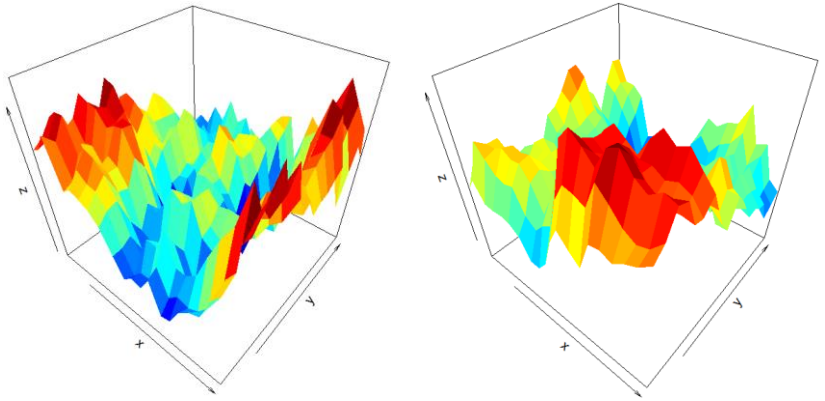
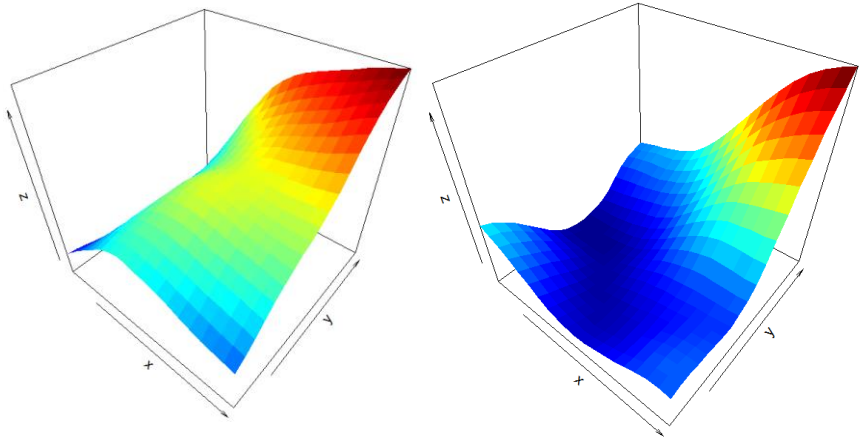
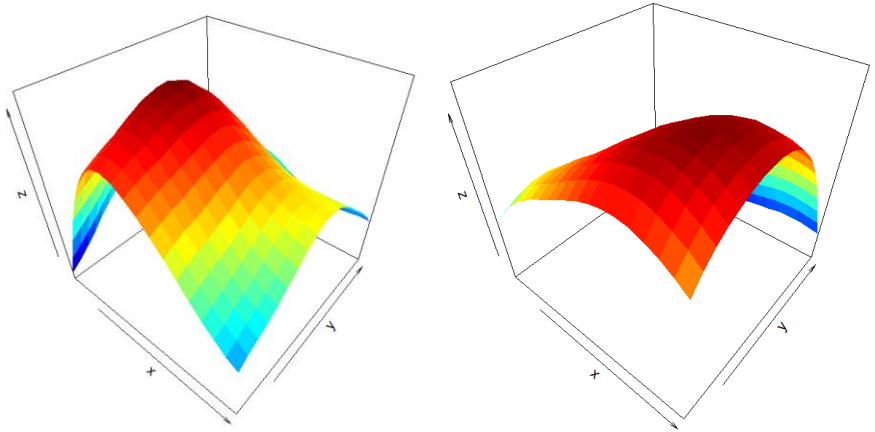
2. Product correlation:

$$\Phi(x) = \Phi_1(x_1) \cdots \Phi_d(x_d),$$

where $\Phi_1, \ldots, \Phi_d$ are 1D correlations, $x =: (x_1, \ldots, x_d)$.

An isotropic Gaussian kernel is also a product kernel.

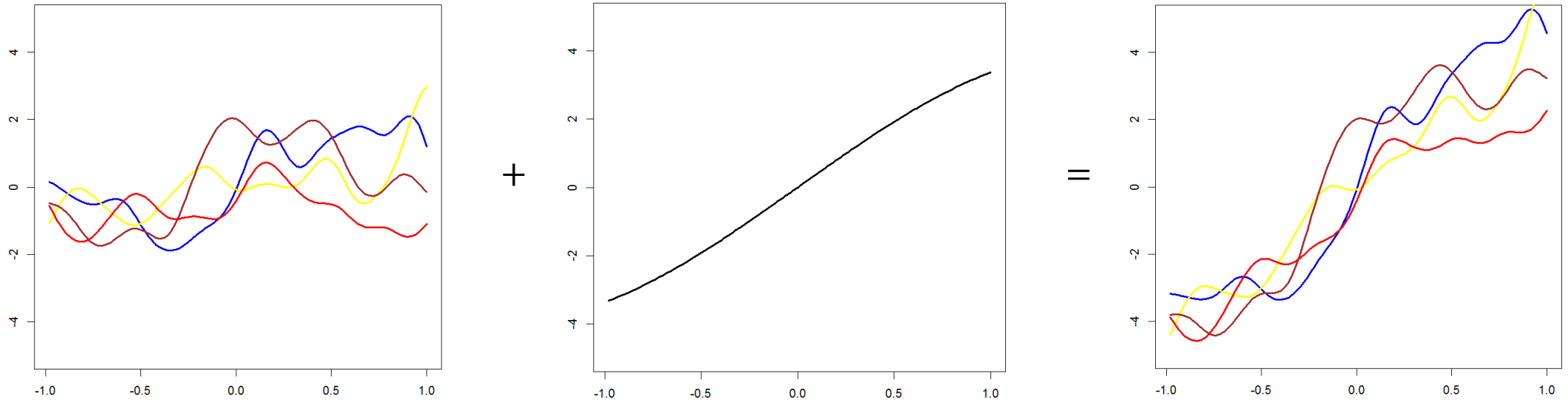
Sample paths

	Isotropic Matérn	Product Matérn
$\nu = 0.5$		
$\nu = 2.5$		

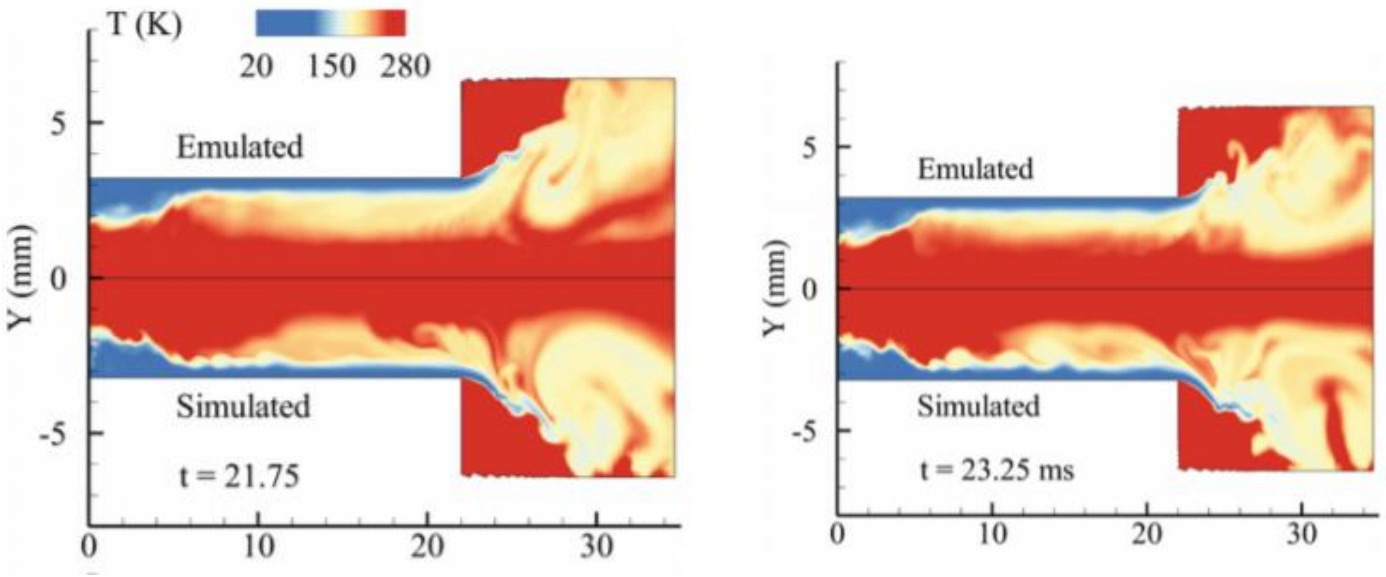
Non-centered Gaussian processes

- A non-centered GP is the sum of a centered GP and a deterministic function.

Centered GP Sample paths $m(x)$ Non-centered GP Sample paths



Machine learning with Gaussian process models



GP surrogate models for Large Eddy Simulations
Figure courtesy of Mak et al. (2018)

Gaussian process regression

- *Simple kriging*

$$y_i = f(x_i) + e_i,$$

with $f \sim GP(0, \sigma^2\Phi)$ and i.i.d. e_i's with $\mathbb{E}e_i = 0$ and $\mathbb{E}e_i^2 = \tau^2$.

- The goal is to reconstruct f based on the data.
- The estimator is denoted as $\hat{f}$.
- If $\tau^2 = 0$, $\hat{f}$ should interpolate f.

Multivariate normal distribution

- Multivariate normal (MVN) distribution is a generalization of $N(\mu, \sigma^2)$.
- To define an MVN random vector, we need
  - Mean vector μ;
  - Covariance matrix Σ.
- Probability density function

$$(2\pi)^{-\frac{n}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right\}.$$

- The conditional distribution of an MVN random vector given some of its entries is also MVN with
  - Condition mean: $\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(Y - \mu_2)$
  - Conditional covariance matrix: $\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{12}$

Prediction via Conditional Distributions

- Suppose $Z \sim GP(0, \sigma^2\Psi)$.
- Given design $X = (x_1, \dots x_n)$, data $Y = \left(Z(x_1), \dots Z(x_n)\right)^T$

For unobserved x, $Z(x)$ is normally distributed with

$$\mathbb{E}[Z(x)|Y] = r^T(x)K^{-1}Y$$
$$\mathrm{Var}[Z(x)|Y] = \sigma^2\left(1 - r^T(x)K^{-1}r(x)\right)$$

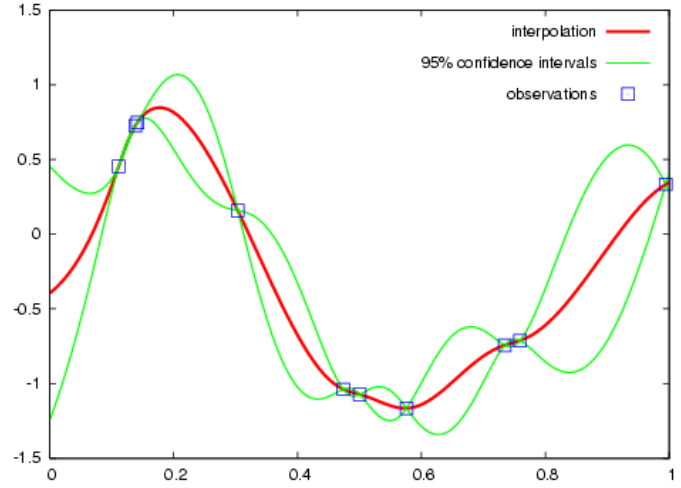
Interpolation Property

Uncertainty Quantification

where

$r(x) = \left(\Phi(x - x_1), \dots, \Phi(x - x_n)\right)^T$: correlation vector

$K = \left(\Phi(x_i - x_j)\right)_{ij}$: kernel matrix



- $\mathbb{E}[Z(x)|Y]$ naturally predict $Z(x)$ given the data.

Computational Challenges

- Predictive mean: $r^T(x)K^{-1}Y$.
- Training step: Solve for
$$u = K^{-1}Y.$$
- Prediction step: Input x; compute
$$\hat{f}(x) = \sum_{i=1}^{n} u_i \Phi(x - x_i)\,.$$
- Time complexity
  - $O(n^3)$ for training via Gaussian elimination;
  - $O(n)$ for prediction.
  - Both unacceptable for a huge n.
- K can be nearly singular when n is large.

Nugget effect

- To enhance numerical stability, we use
$$u = (K + \lambda I)^{-1} Y,$$
with a small $\lambda > 0$, say, 10^{-9}.

- λ is called a nugget term.

- The predictor is *no longer* an interpolant.

- This approach is equivalent to the predictor give the noisy data with
$$\sigma^2 = \lambda.$$

Universal kriging

- Model

$$f \sim GP\left(\mu(\cdot), \sigma^2 \Phi_\theta(\cdot,\cdot)\right).$$

  - $\mu(\cdot) = \sum \beta_j f_j(\cdot)$: linear combination of basis functions with unknown coefficients.

- Parameters can be estimated by MLE or Bayesian methods.

- Prediction can be done by plugging in the estimated parameters or a full Bayesian approach.

Maximum likelihood estimation

- Parameters of a universal kriging model
  - Regression coefficients β
  - Variance σ^2
  - Correlation parameters θ

- Estimate the parameters by maximizing the likelihood function

$$\left(\hat{\beta}, \hat{\sigma}^2, \hat{\theta}\right) = \operatorname{argmax} P(Y|\beta, \sigma^2, \theta)\,.$$

Multivariate normal distribution

- Maximization usually proceeds by a gradient descend algorithm.

Bayesian methods

❑Step 1: Choose a prior for $(\beta, \sigma^2, \theta)$.

❑Step 2: Use the Bayes rule to determine the posterior

$$P(\beta, \sigma^2, \theta|Y) \propto P(Y|\beta, \sigma^2, \theta) \times P(\beta, \sigma^2, \theta).$$

❑Step 3: Bayesian computation and inference

- Markov Chain Monte Carlo
- Variational inference

Why fit a deterministic function with a GP?

- Justification from a Bayesian perspective

  Regard the GP as a prior of the underlying function.

- Justification from a frequentist perspective

  GP regression, as a methodology, works for a family of problems.
  Regard the specific problem as a sample from the "population of problems".

- Justification from the approximation theory

  The approximation error is mathematically in control under mild conditions.

More supervised learning problems

- A general supervised learning problem:
  - Data: (x_i, y_i)
  - Underlying function f, supposed to be continuous.
  - Empirical loss:

$$Loss(f) \coloneqq \sum l\big(y_i, f(x_i)\big).$$

  - GP prior: $f \sim GP\big(\mu(\cdot), \sigma^2\Phi(\cdot)\big)$.
- Data augmentation
  - Given $z_i = f(x_i)$, the problem can be decomposed into two parts.

➢ Empirical loss:

$$Loss = \sum l(y_i, z_i).$$

➢ GP regression: $z_i = f(x_i)$.

Training methods

- Frequentist approach
  - Minimize the regularized loss function

$$\min_{Z,\beta,\sigma^2,\theta} \sum l(y_i, z_i) - \log LH(\beta, \sigma^2, \theta | Z) .$$

  - If $f \sim GP(0, \sigma^2 \Phi)$ with a known Φ, the above method is equivalent to a kernel learning method:

$$\min_{f} \sum l(y_i, z_i) + \lambda \| f \|_{\Phi}^2$$

- Bayesian posterior density

$$P(\beta, \sigma^2, \theta, Z | Y) \propto P(Y | Z) \times P(Z | \beta, \sigma^2, \theta) \times P(\beta, \sigma^2, \theta).$$

Example: GP-based logistic regression

- Classification problem: $y \in \{0,1\}$, input x is real-valued.
- Likelihood function given Z

$$P(Y|Z) = \prod \left(\frac{e^{z_i}}{1+e^{z_i}}\right)^{y_i} \left(\frac{1}{1+e^{z_i}}\right)^{1-y_i}.$$

- The posterior density is

$$P(\beta, \sigma^2, \theta, Z|Y) \propto P(Y|Z)P(Z|\beta, \sigma^2, \theta)P(\beta, \sigma^2, \theta).$$

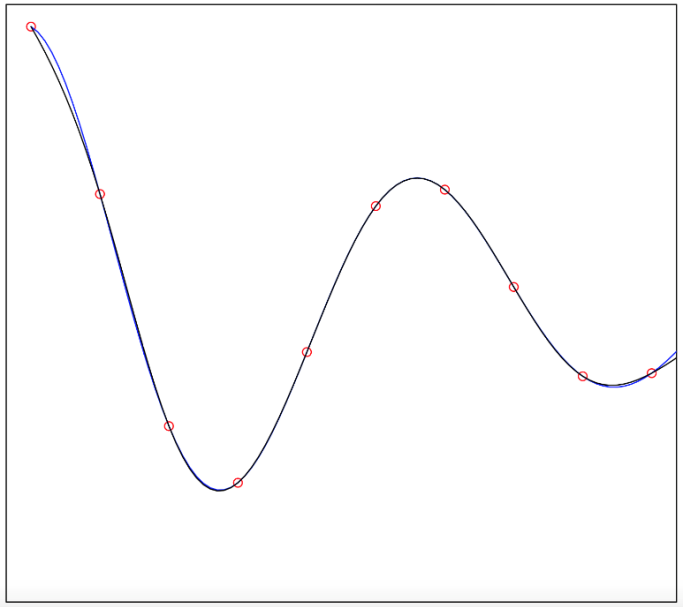
- Prediction at a new input x_{new}:
  - Step 1: sample z_{new} from the posterior distribution of $f(x_{new})$
  - Step 2: sample y_{new} from $P(y|z_{new}) = \left(\frac{e^{z_{new}}}{1+e^{z_{new}}}\right)^{y_{new}} \left(\frac{1}{1+e^{z_{new}}}\right)^{1-y_{new}}$.

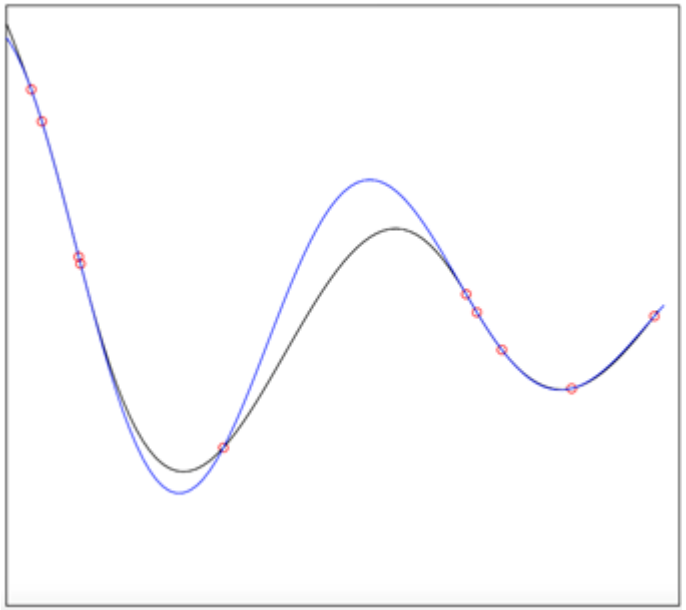
When to use GP models

- GP models are suitable under the following conditions
  1. Underlying function is smooth
  2. Data size is moderate
  3. Input dimension is not too high
  4. Signal-to-noise ratio is high
  5. Uncertainty quantification is of interest

- Typical areas and problems
  - Spatial statistics (GP is a natural tool to capture spatial-temporal correlation)
  - Bayesian optimization
  - Surrogate modeling for complex computer models

Design of experiments





Space-filling designs versus random designs

Design of Experiments

- The performance for GP models (as well as other methodologies) highly rely on the set of input points x of the training data.

- Goal of DoE: Choose the best input sets to run the experiment to maximize the prediction performance.

- Three principles of experimentation (suggested by R. A. Fisher)
  - Replication: Reducing inevitable random noise
  - Blocking: Removing effects of recognized nuisance variables
  - Randomization: Removing effects of unrecognized variables

- The above principles are not applicable for GP models

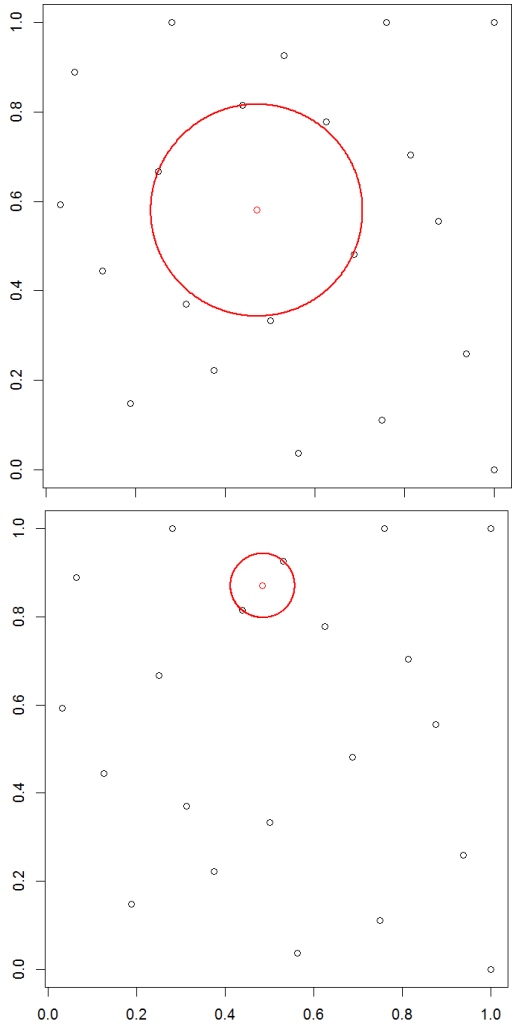
Experimental design strategies

- Geometric considerations
  - Space-filling designs
- Projection properties
  - Latin hypercube designs
- Tensor-product-based designs
  - Full grid designs
  - Sparse grid designs
- Optimal designs

Distance-based criteria [JMY90]

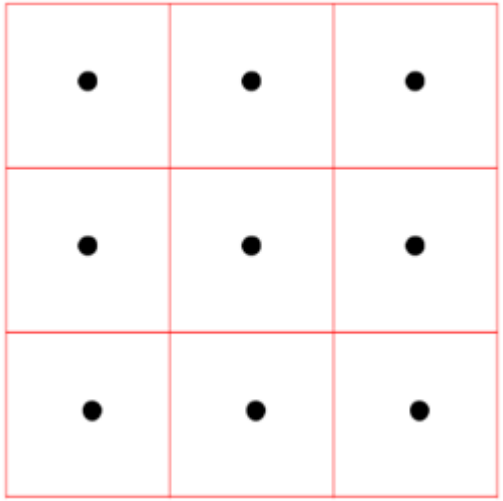
- Fill distance
  - $h_{X,\Omega} = \sup_{x \in \Omega} \min_{x_j \in X} \|x - x_j\|$.
  - Minimize $h_{X,\Omega}$ ➡ minimax distance design

- Separation distance
  - $q_X = \frac{1}{2} \min_{i \neq j} \|x_i - x_j\|$.
  - Maximize q_X ➡ maximin distance design.



(Full) Grid Designs

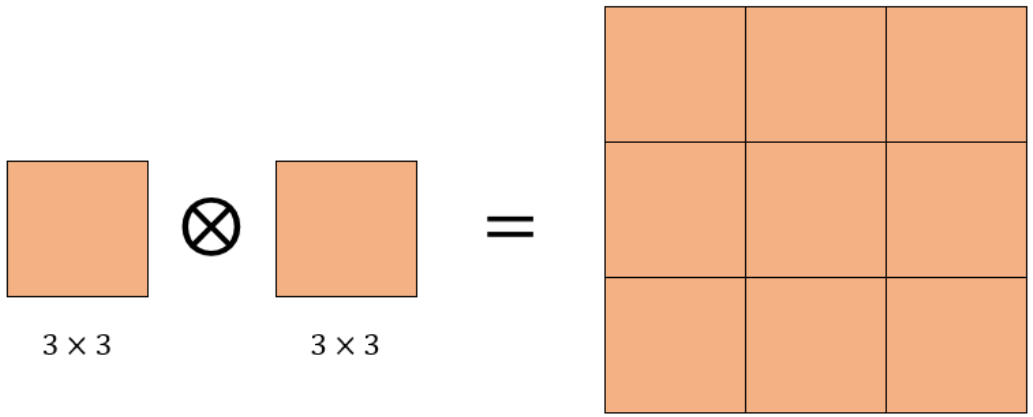
- A simple space filling design.
- Not necessarily a square (hypercube) design.
- Arisen naturally in certain problems,
  e.g., imaging, remote sensing, etc.
- Good accuracy for isotropic kernels.
- Less accurate for product (Matérn) kernels.
- Main reason: poor projection properties.

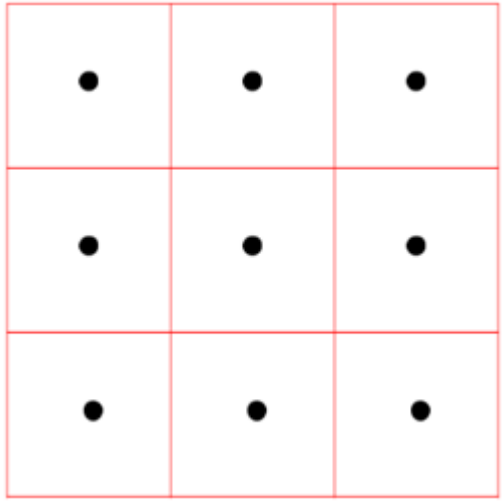


When projected onto 1D, only 3 points are left.

(Full) Grid Designs: computational advantages

- Two performance measures
  1. Prediction accuracy
  2. Computational efficiency
- Despite the accuracy deficiency, grid designs for product kernels enjoy computational advantages.
- The kernel matrix is a tensor product.





Kriging prediction with 9 input points

1. Direct Gaussian elimination
   Time complexity = $O(9^3)$.
2. Tensor product + Gaussian elimination
   Time complexity = $O(3^3)$.

Latin hypercube designs

- A d-dimensional grid design has n^d points
- A Latin hypercube design (LHD) is an n-point *subset* such that each row and column have exactly one point.
- There are $n!$ difference LHDs.
- Space-filling metrics are usually incorporated to choose the best LHDs.
  - E.g., minimax LHDs.

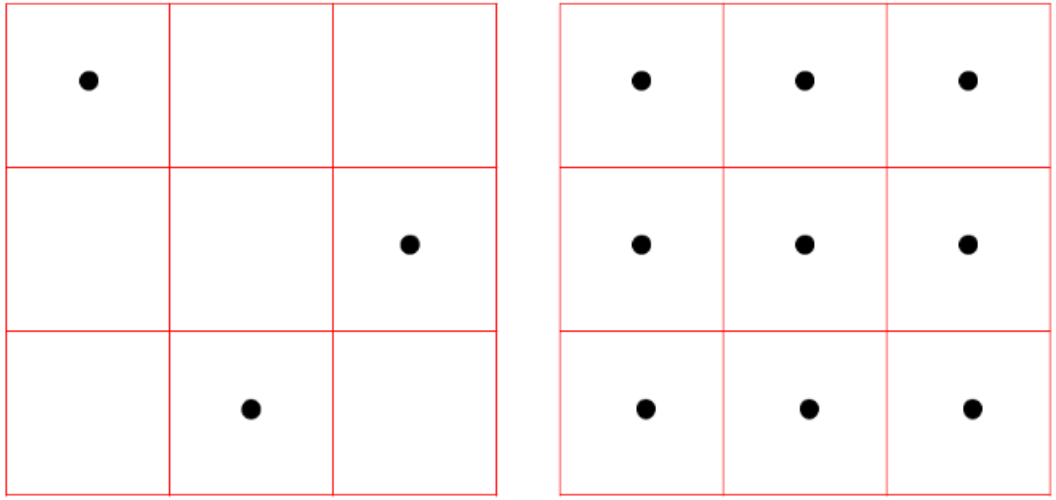


Fig. Latin hypercube design versus full grid design

Optimal designs

- Idea: Minimize a criterion function, usually related to a prediction error.

- Notation: D=design, $\hat{Y}_D$=kriging predictor given D.

- Integrated mean squared prediction error

$$IMSPE(D) = \int_{\Omega} \mathbb{E}\left[Y(x) - \hat{Y}_D(x)\right]^2 dx\,.$$

- Maximum mean squared prediction error

$$MMSPE(D) = \max_{x\in\Omega} \mathbb{E}\left[Y(x) - \hat{Y}_D(x)\right]^2.$$

Sparse grid designs

- Sparse grid designs provide a tradeoff between prediction accuracy and computational efficiency.

- Sparse grids
  - Suitably chosen subset of a full grid.
  - Better projection properties than full grids.
  - Matrix inversion can be done efficiently via the Smolyak algorithms.

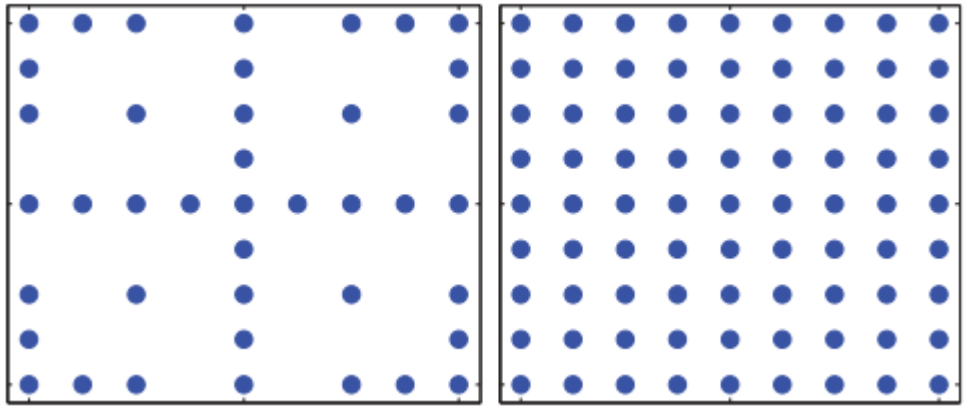


Fig. courtesy of [Plumlee14].
Sparse grid design versus full grid design.

GP models with nonstationary covariance

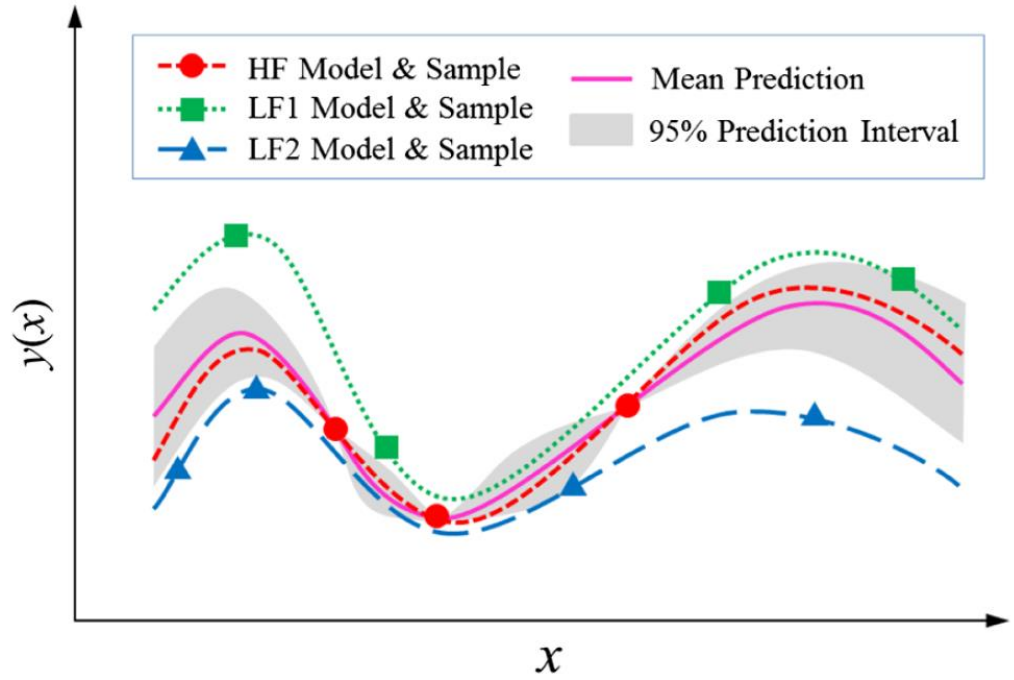
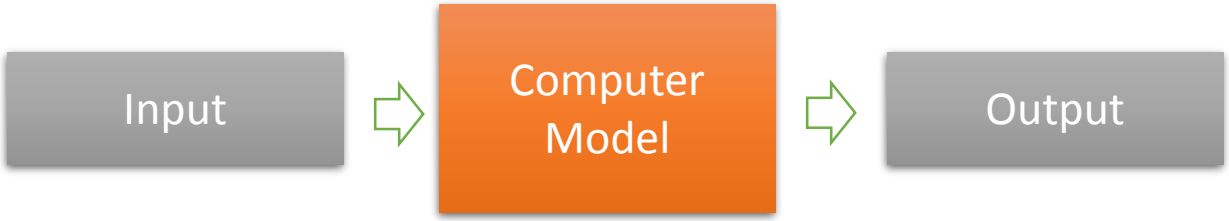


Figure courtesy of [CJYC17].

Nonstationary GPs can fuse experimental data from different sources.

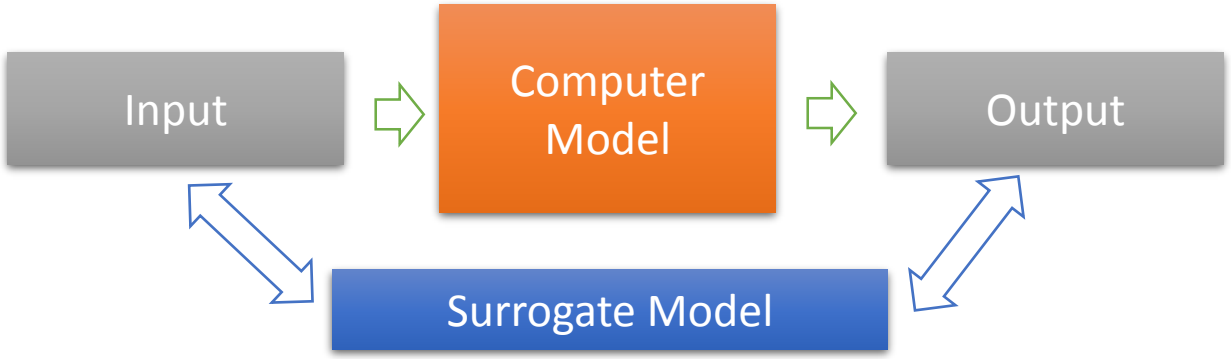
Computer Experiments

- Computer model is a complex black box function.



Deterministic; Expensive

- The aim of CE is to explore and reconstruct the function relationship between the input and the output.



Accurate; Fast

Multi-fidelity computer models

- Computer codes with different accuracy levels are available.
- Example: FEA with different mesh size.
- Properties:
  - High fidelity computer code is more accurate.
  - High fidelity computer code is also more costly.
- Goal: integrate CE outputs from different fidelities to improve the prediction.

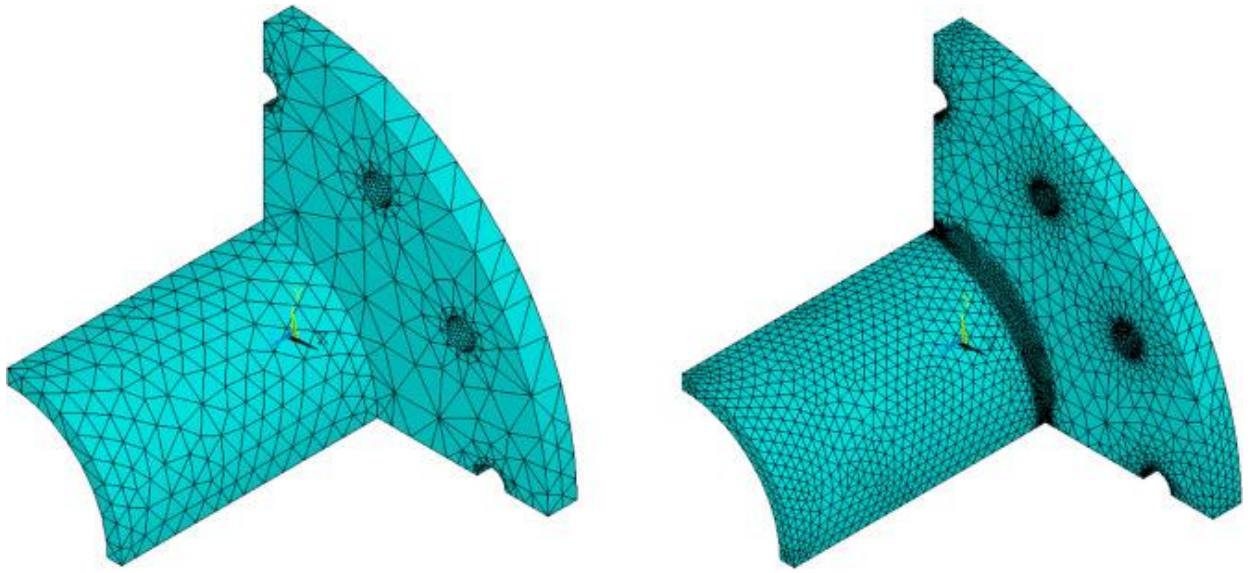


Figure courtesy of [TT17].

Autoregressive model

- Autoregressive model suggested by Kennedy and O'Hagan [KO00].

$$z_1(x) = \epsilon_1(x).$$
$$z_2(x) = z_1(x) + \epsilon_2(x).$$
$$\cdots$$
$$z_S(x) = z_{S-1}(x) + \epsilon_S(x).$$

  - z_t = computer output at fidelity level t, $t = 1, \ldots, S$. Accuracy increases in t.
  - Model ϵ_t as mutually independent GPs with stationary covariances.

Calibration of computer models

- Problem description
  - Both the computer code and the physical data are available
  - Computer code requires unknown input parameters (physical properties)
    - E.g, permeability, conductivity, etc.
- "Calibration is the activity of adjusting the unknown (calibration) parameters until the outputs of the (computer) model fit the *observed* data." [KO01].

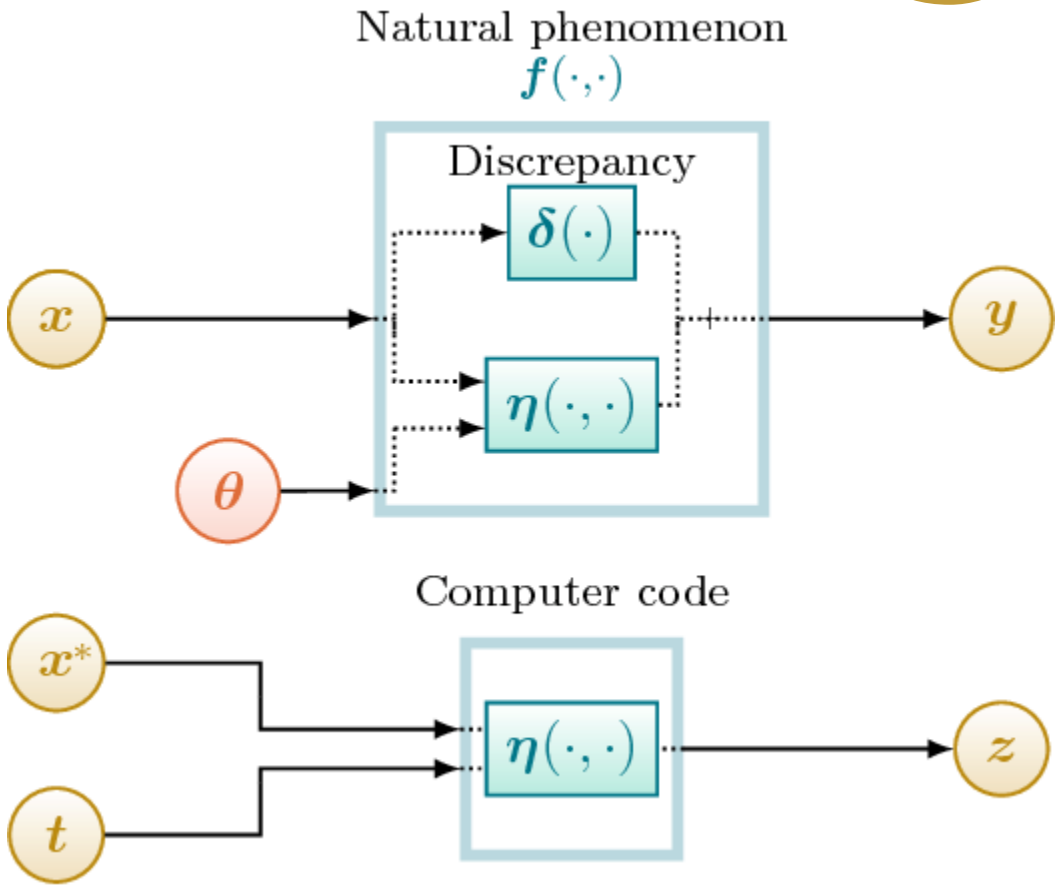


Figure courtesy of [MSM18].

Kennedy-O'Hagan approach [KO01]

- Model

$$y_i^p = \zeta(x) + \epsilon_i$$
$$\zeta(x) = \eta(x, \theta_0) + \delta(x),$$

  - y_i^p = ith physical observation;
  - ζ = the average physical response at input x, known as the true process;
  - η = computer output;
  - δ = discrepancy function (CE cannot perfectly mimic the physical process);
  - ϵ_i = random error corresponding to ith physical observation.
  - Model η and δ as independent GPs with stationary covariances.
- Estimating θ_0
  - Impose a prior for θ_0.
  - Use MCMC to obtain the posterior of θ_0.

Bayesian Optimization

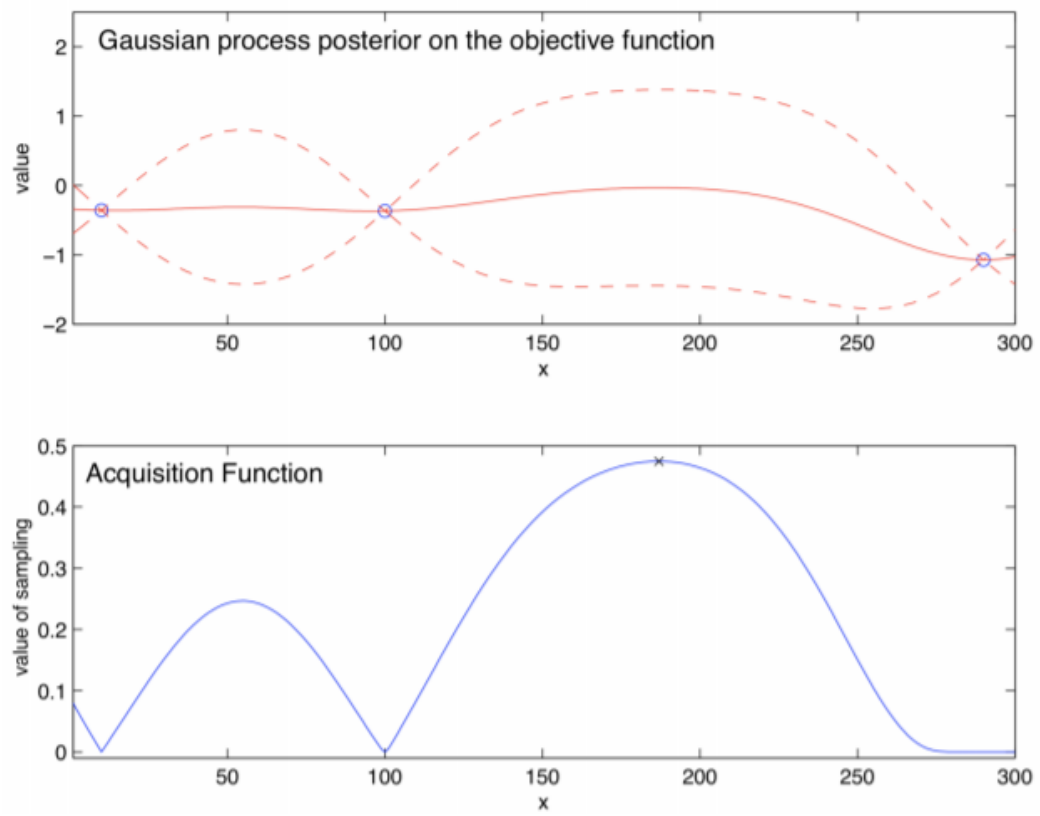


Figure courtesy of Frazier (2018).

Problem of interest

- Global optimization

$$\max_{x \in A} f(x).$$

- Bayesian optimization methodologies are mostly promising if
  - The input dimension is not too large, typically no more than 20.
  - The objective function f is continuous.
  - No known special structure of f, such as convexity.
  - f is expensive to evaluate.

E.g., How to best train our Ph.D students?

- Applications:
  - ❑Optimizing complex computer model outputs
  - ❑Reinforcement learning
  - ❑Architecture configuration in deep learning
  - ❑…

Sequential optimization

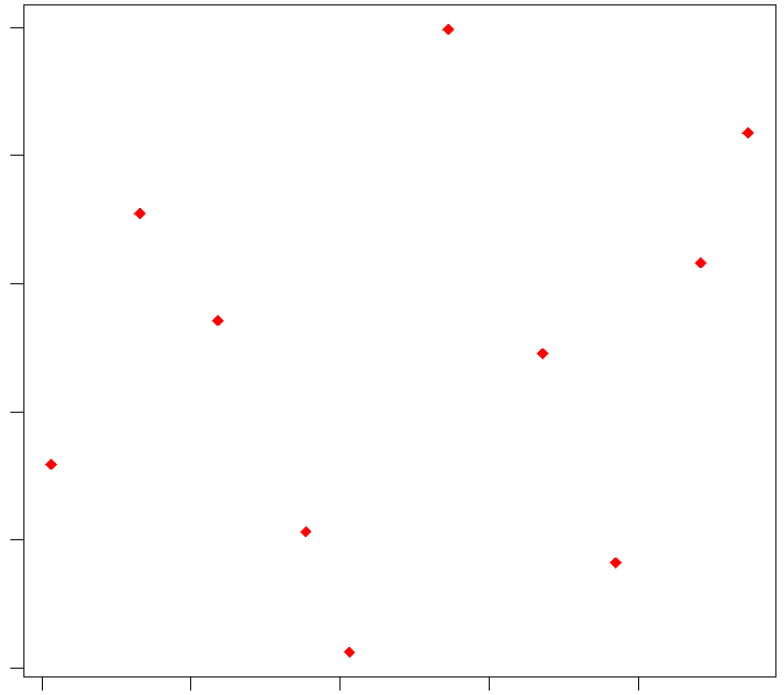
- Step 1:

  Choose a GP prior for f.

- Step 2:

  Choose an initial design, e.g., a maximin Latin-hypercube design.

  Evaluate f over the initial design.



Sequential optimization

- Step 1:

  Choose a GP prior for f.

- Step 2:

  Choose an initial design, e.g., a maximin Latin-hypercube design.

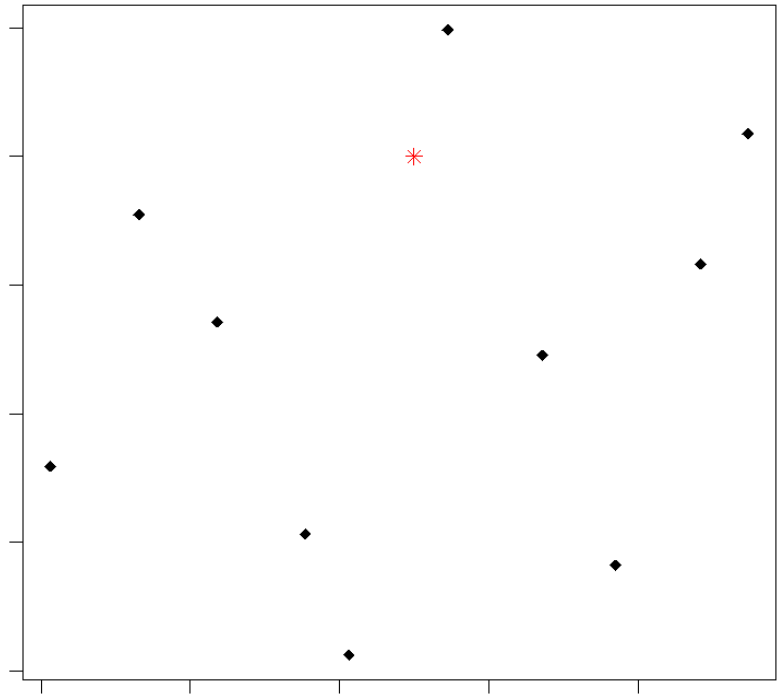
  Evaluate f over the initial design.

- Step 3:

  Update the posterior of the GP.

- Step 4:

  Determine the next point by optimize an acquisition function.



Sequential optimization

- Step 1:

  Choose a GP prior for f.

- Step 2:

  Choose an initial design, e.g., a maximin Latin-hypercube design.

  Evaluate f over the initial design.

- Step 3:

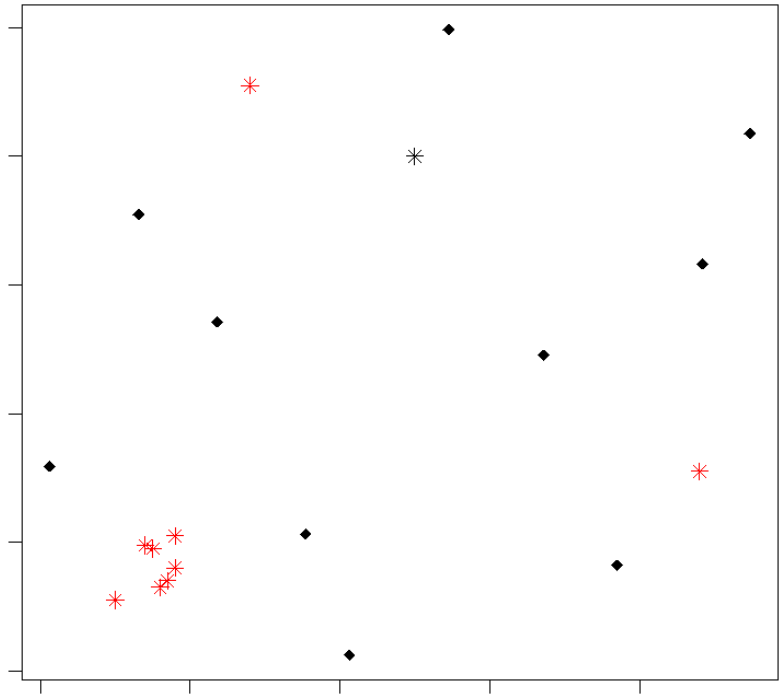
  Update the posterior of the GP.

- Step 4:

  Determine the next point by optimize an acquisition function.

- Step 5:

  Repeat Steps 3 & 4 until budget is used or accuracy level is met.



Acquisition Function

- Acquisition function is a function of input location.
  It also depends on the GP posterior.

- Denote the acquisition function by $a_n(x)$ given the first n inputs.

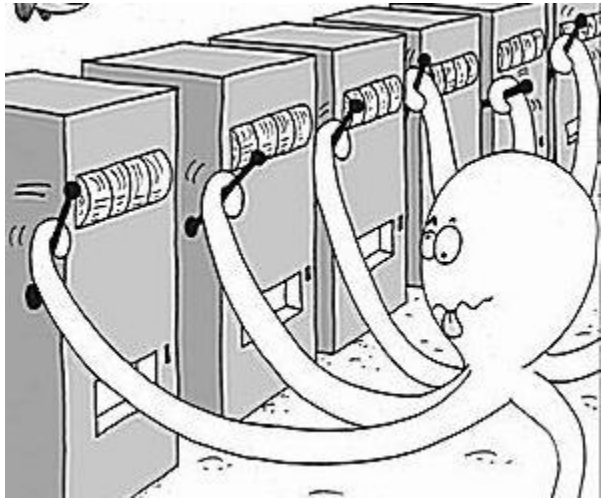
- Determine the next input as

$$x_{n+1} = \text{argmax}_x \, a_n(x) \, .$$

- Another global optimization is needed.
  But it is easier as a_n is less expensive.

Exploration versus Exploitation

- Multi-armed bandit
  - Exploitation
    Play the arm with the highest expected reward.
  - Exploration
    Play the arm with the highest uncertainty.

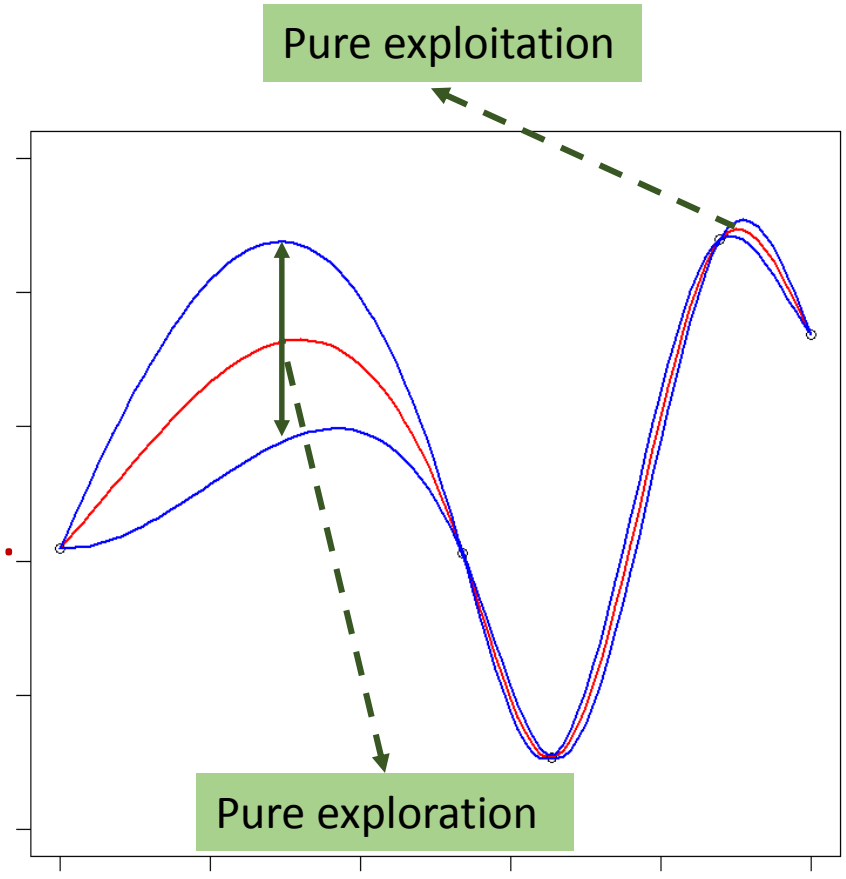


- Bayesian optimization
  - Exploitation
    Sample the point with the highest expected value.
  - Exploration
    Sample the point with the highest uncertainty.

Exploration versus Exploitation

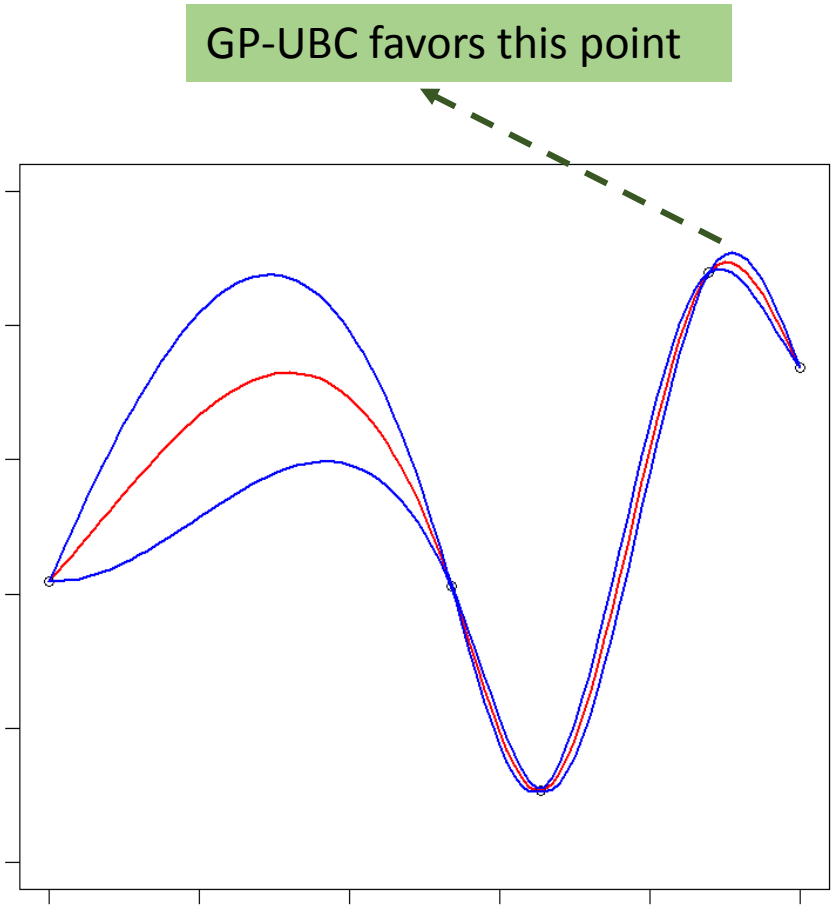
- Multi-armed bandit
  - Exploitation
    Play the arm with the highest expected reward.
  - Exploration
    Play the arm with the highest uncertainty.

- Bayesian optimization
  - Exploitation
    Sample the point with the highest expected value.
  - Exploration
    Sample the point with the highest uncertainty.



GP-UCB

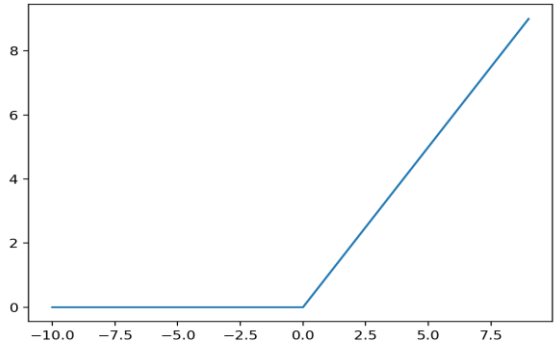
- An intuitive method to balance the exploitation and exploration.
- Consider the α-upper confidence bound, denoted as $UCB(\alpha)$.
  Blue line in the Figure.
- Acquisition function
$$a_n(x) = UCB(\alpha_n).$$
- UCB can be expressed as
$$UCB(\alpha_n) = \mu_n(x) + \beta_n^{\frac{1}{2}}\sigma_n(x).$$
- A theory is available to determine β_n.



Expected improvement

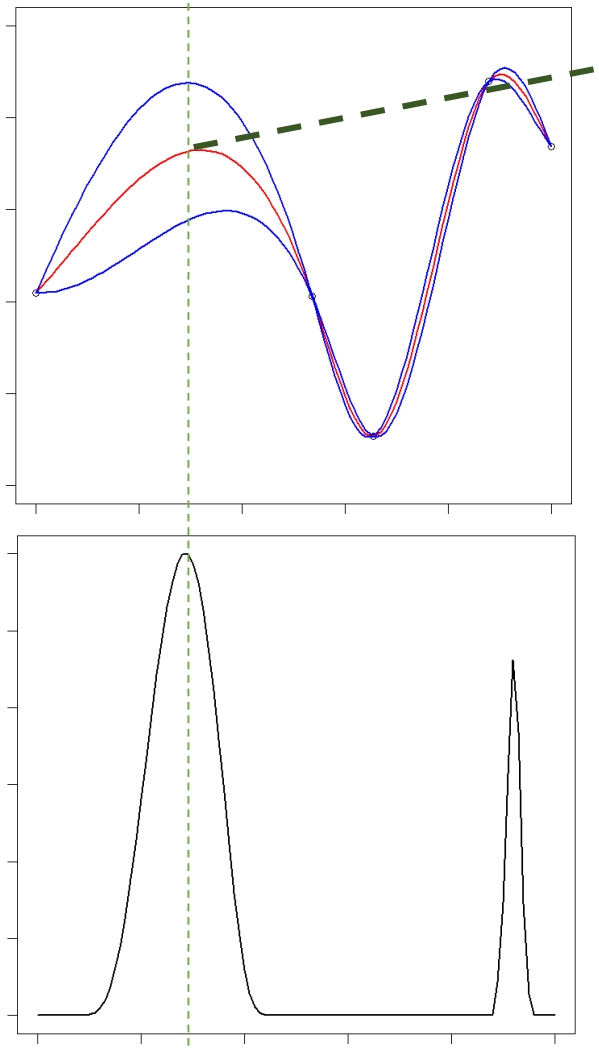
- Most commonly used acquisition function.
- Maximum value in the current observations = f_n^*.
- Improvement of a *potential* observation:

$$[f(x) - f_n^*]^+ = \begin{cases} f(x) - f_n^* & \text{if} \quad f(x) - f_n^* > 0; \\ 0 & \text{otherwise.} \end{cases}$$



This function is known as a *Rectifier* in Deep Learning.

Expected Improvement



EI favors this point

- Acquisition function, called the Expected Improvement:

$$\mathrm{EI}_n(x) := \mathbb{E}\left[[f(x) - f_n^*]^+ | \text{observations}\right].$$

- $\mathrm{EI}_n(x)$ can be expressed explicitly, and a function of $\mu_n(x)$ and $\sigma_n(x)$.
- EI does not rely on a tuning parameter.

Other Bayesian Optimization Criteria

- Probability of improvement
- Knowledge Gradient
- Entropy Search
- …

Conclusion

- Advantages of GP models
  - GP models enable uncertainty quantification.
  - GP models can accommodate complex data structure and prior information.
- Deficiencies of GP models
  - Computational issues when n is large.
    (This can be partially evaded by choosing appropriate designs.)
  - Cannot handle discontinuous response surfaces.

Thank you for attending the talk!

References

- [JMY90] Johnson, Mark E., Leslie M. Moore, and Donald Ylvisaker. "Minimax and maximin distance designs." *Journal of Statistical Planning and Inference* 26.2 (1990): 131-148.
- [Plumlee14] Plumlee, Matthew. "Fast prediction of deterministic functions using sparse grid experimental designs." Journal of the American Statistical Association 109.508 (2014): 1581-1591.
- [CJYC17] Chen, S., Jiang, Z., Yang, S., and Chen, W., “Multi-Model Fusion Based Sequential Optimization”, AIAA Journal, 55(1), 2017.
- [TT17] Thompson, M.K. and Thompson, J.M., 2017. *ANSYS mechanical APDL for finite element analysis*. Butterworth-Heinemann.
- [KO00] Kennedy, Marc C., and Anthony O'Hagan. "Predicting the output from a complex computer code when fast approximations are available." *Biometrika* 87.1 (2000): 1-13.
- [KO01] Kennedy, Marc C., and Anthony O'Hagan. "Bayesian calibration of computer models." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.3 (2001): 425-464.
- [MSM18] Marmin, Sébastien, and Maurizio Filippone. "Variational Calibration of Computer Models." *arXiv preprint* arXiv:1810.12177 (2018).
- [Plumlee17] Plumlee, M. Bayesian calibration of inexact computer models. *Journal of the American Statistical Association*, vol. 112, no. 519, pp. 1274-1285, 2017.