Part 1: GP Models for Optimization

William J. Welch

University of British Columbia

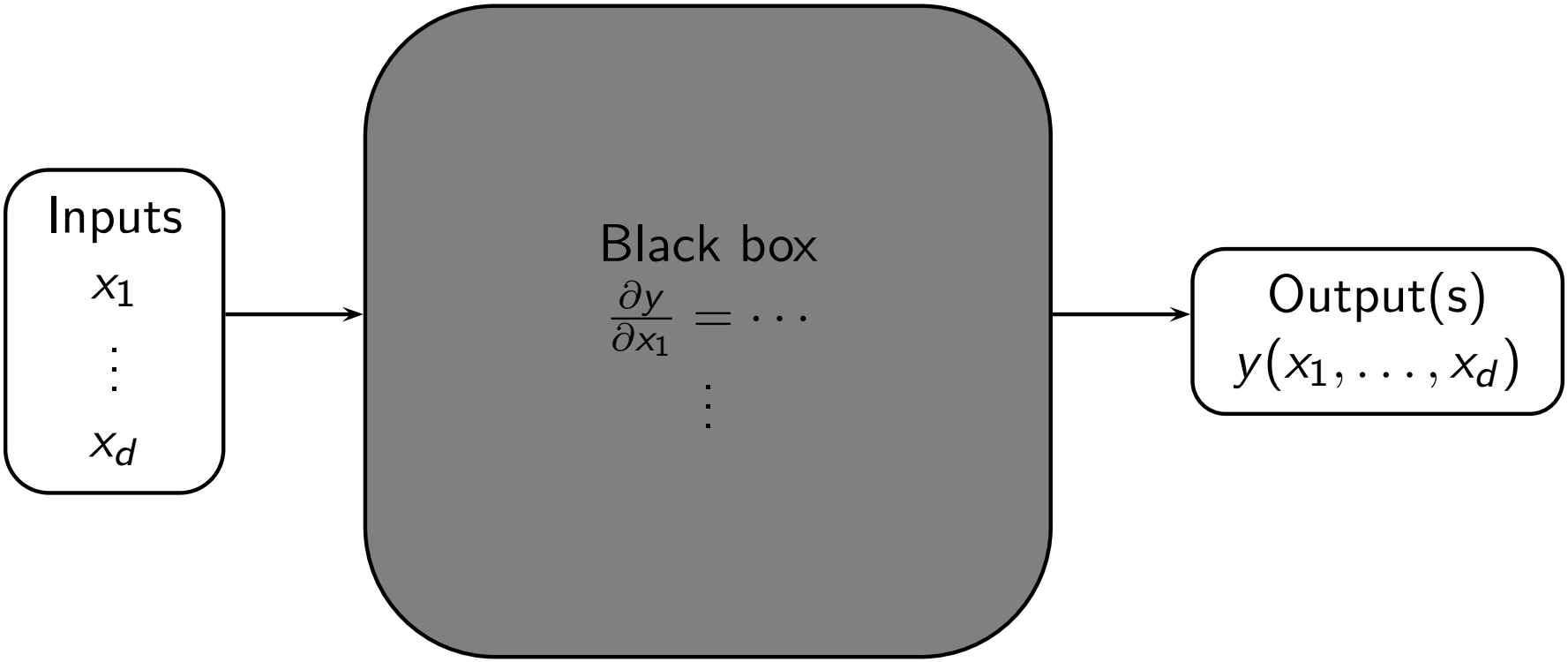
Adapted from materials prepared by Jerry Sacks (National Institute of Statistical Sciences) and Will Welch for various short courses.
Also featuring GaSP software developed by Will Welch and Yilin (Justin) Yang.

School on Artificial Intelligence for Materials Science in the Exascale Era, May 22, 2023

(Part 2 will be on Efficient Global Optimization)
Slides and files for the examples are at
https://www.stat.ubc.ca/~will/cap_roig.html

Outline of Topics

Computer Code



We will be thinking mainly about deterministic functions $y(\mathbf{x})$.

- But many of the arguments can be adapted if y is measured with random error.

Jerry Sacks 1931–2023

Pioneer of statistical design and analysis of "computer experiments", mentor, friend, ...

GaSP (Gaussian Stochastic Process) Package in R

Yilin (Justin) Yang

- We will use the GaSP package (Welch and Yang, 2021) to illustrate methods.

Install from CRAN if you want to run provided scripts.

Deterministic Codes: Why Statistics?

- Need for statistical prediction of functions representing complex science
  - Limited number of code runs n (few data)
  - Moderate to many inputs (high dimension)
  - We want to predict model output at untried inputs based on limited data
  - There will be uncertainty in predictions (key to use in optimization)
- Some other sources of uncertainty
  - Propagation of variation: variability in inputs induces uncertainty in outputs
  - Assess the fidelity of the model to reality (validation); needs field/experimental data which will be measured with error

  (Not considered further today.)
- Design
  - Where to run the computer code (an essential part of optimization)

What's the Problem?

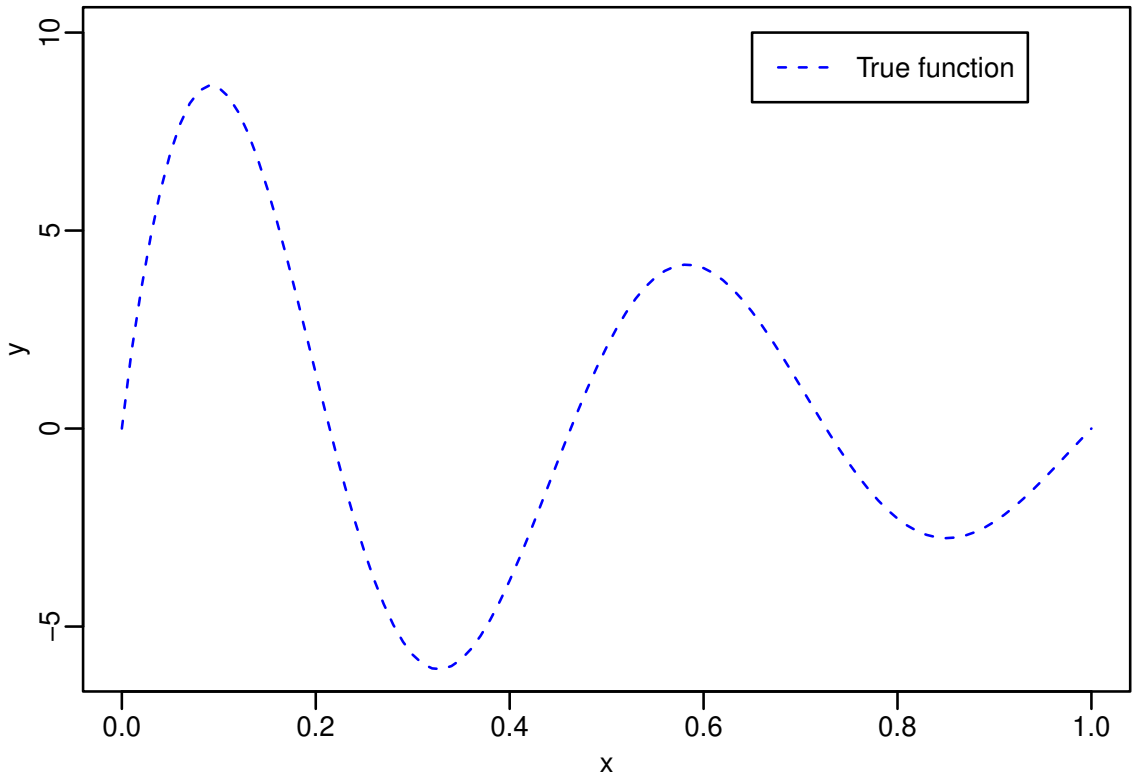
- Run the code at n locations $\mathbf{x}^{(i)}$ for $i = 1, \ldots, n$ in the input space.
- Now have data $\{\mathbf{x}^{(i)}, y(\mathbf{x}^{(i)})\}$ for $i = 1, \ldots, n$.
- Want to predict $y(\mathbf{x})$ at a new $\mathbf{x}$, a standard statistical question; also standard function approximation (no error).
- Don't know much about the function $y(\mathbf{x})$, and if we specify a class (like cubic splines) or use a deep neural network we need lots of data because of high dimensions.

Bayesian Strategy

Has a long history, but see Currin, Mitchell, Morris, and Ylvisaker (1991), Rasmussen and Williams (2006), Sacks, Welch, Mitchell, and Wynn (1989).

- Before collecting data (evaluating the function) we have a vague idea of $y(\mathbf{x})$'s properties and so think of $y(\mathbf{x})$ as random.
- Our prior belief or uncertainty about $y(\mathbf{x})$ is measured by a probability distribution.
- Collect data.
- Now update belief/uncertainty through the conditional distribution of $y(\mathbf{x})$ given the data. In particular, predict $y(\mathbf{x})$ at a new $\mathbf{x}$ as $E(y(\mathbf{x}) \mid \text{data})$.
- Conceptually: prior uncertainty + data $\Rightarrow$ updated (posterior) uncertainty (the Bayesian Paradigm)

Target Function (Treat it as Unknown)



Pretend we don't know this function, but we think of it as a Realization of a random function.

What is a Gaussian Process (GP)?

- A (deterministic) function $y(\mathbf{x})$ coded in a computer model is a "table" $\{\mathbf{x}, y(\mathbf{x})\}$.
- Graph the function by plotting the points of the table. (Plot $y(\mathbf{x})$ at a large number, N, of points, $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$ — a scatter diagram — and "connect the dots".)
- Suppose these values are the outcome of a random draw from some joint Gaussian distribution of random variables $y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)})$ and plot as above. We will get a realization of a Gaussian process (GP).
- Alternatively, think of the distribution of $y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)})$ as a prior distribution for the function values $y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)})$.

The Prior

- We will abuse notation and think of $y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)})$ at any N points as random.
- We will work solely with the multivariate normal (MVN) distribution for the $y(\mathbf{x}^{(i)})$.
- Each $y(\mathbf{x}^{(i)})$ has mean μ (can easily be generalized to μ varying according to a regression function)
- The covariance matrix is $\sigma^2\mathbf{R}$ where the correlation matrix

$$\mathbf{R} = \mathrm{Cor}(y(\mathbf{x}^{(i)}), y(\mathbf{x}^{(j)})) \quad (N \times N \text{ matrix})$$

  is specified and absolutely critical to the GP approach.
- Summary: $\mathbf{y} = (y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)}))^T$ is MVN$(\mu\mathbf{1}, \sigma^2\mathbf{R})$, i.e., has density

$$\frac{1}{(2\pi\sigma^2)^{N/2}(\det \mathbf{R})^{1/2}} \exp\left(-\frac{1}{2\sigma^2}(\mathbf{y}-\mu\mathbf{1})^T\mathbf{R}^{-1}(\mathbf{y}-\mu\mathbf{1})\right),$$

where $\mathbf{1}$ is an $N \times 1$ vector of 1's.

Example Correlation Functions in One or More Dimensions

The squared-exponential (Gaussian) correlation function is a popular choice.
Let $\mathbf{x}$ and $\mathbf{x}'$ be two sets of values for the input variables.
For $\theta > 0$:

- 1 dimension, $\mathbf{x} = x$

$$\mathrm{Cor}(y(x), y(x')) \equiv R(x, x') = \exp(-\theta|x - x'|^2)$$

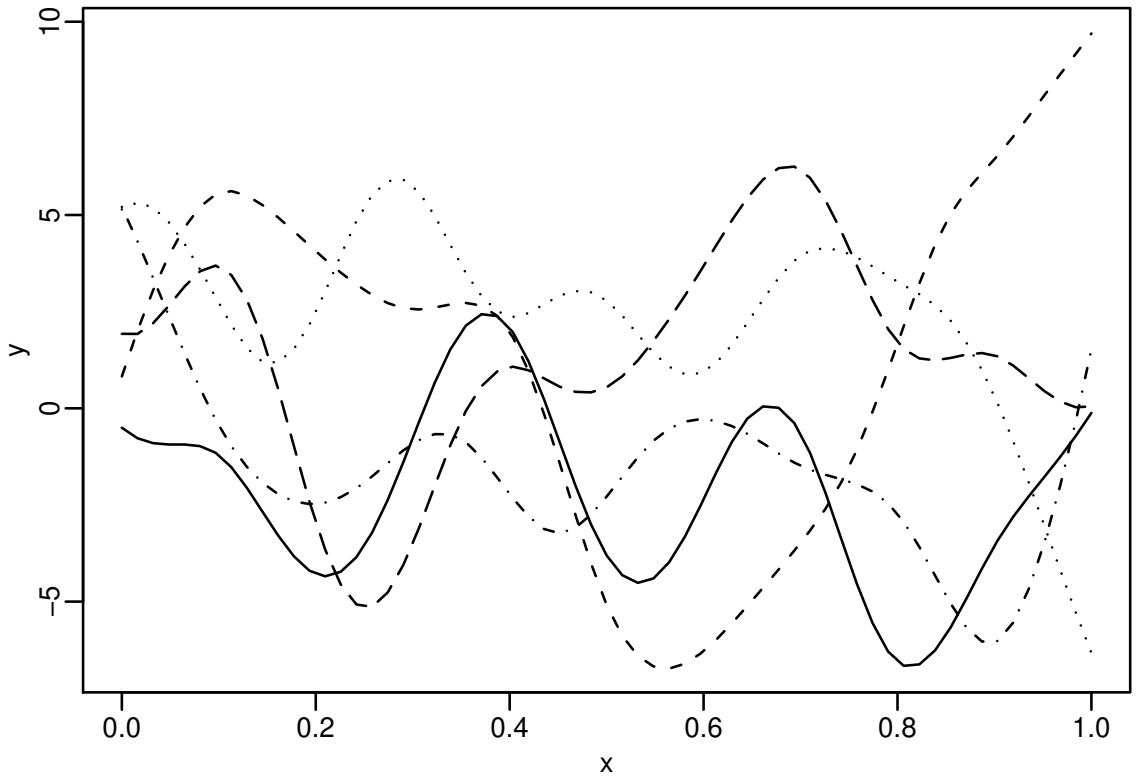
and

$$\mathbf{R} = [\exp(-\theta|x^{(i)} - x^{(j)}|^2)]$$

- 2 dimensions, $\mathbf{x} = (x_1, x_2)$

$$R(\mathbf{x}, \mathbf{x}') = \exp(-\theta_1|x_1 - x_1'|^2) \times \exp(-\theta_2|x_2 - x_2'|^2)$$

Some Realizations Using the Squared-Exponential Correlation Function



- We don't know the target function!

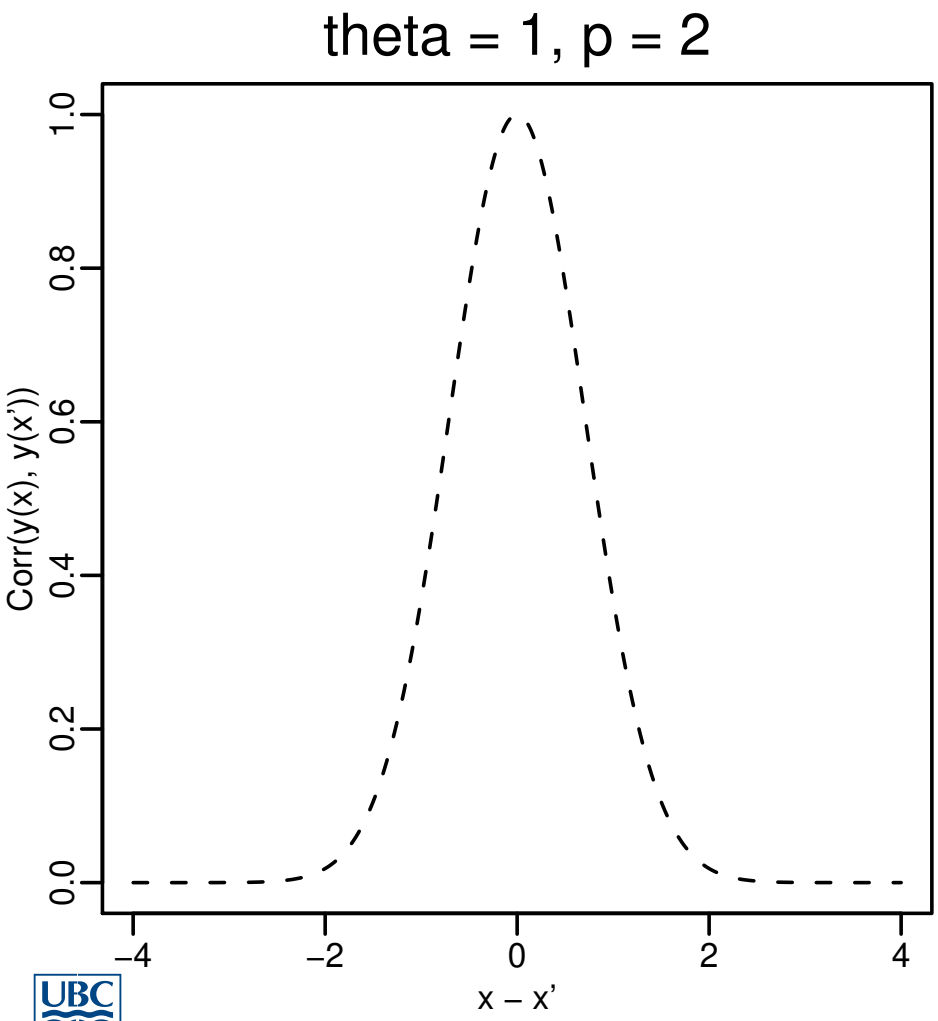
But we think it has qualitative properties like these functions.

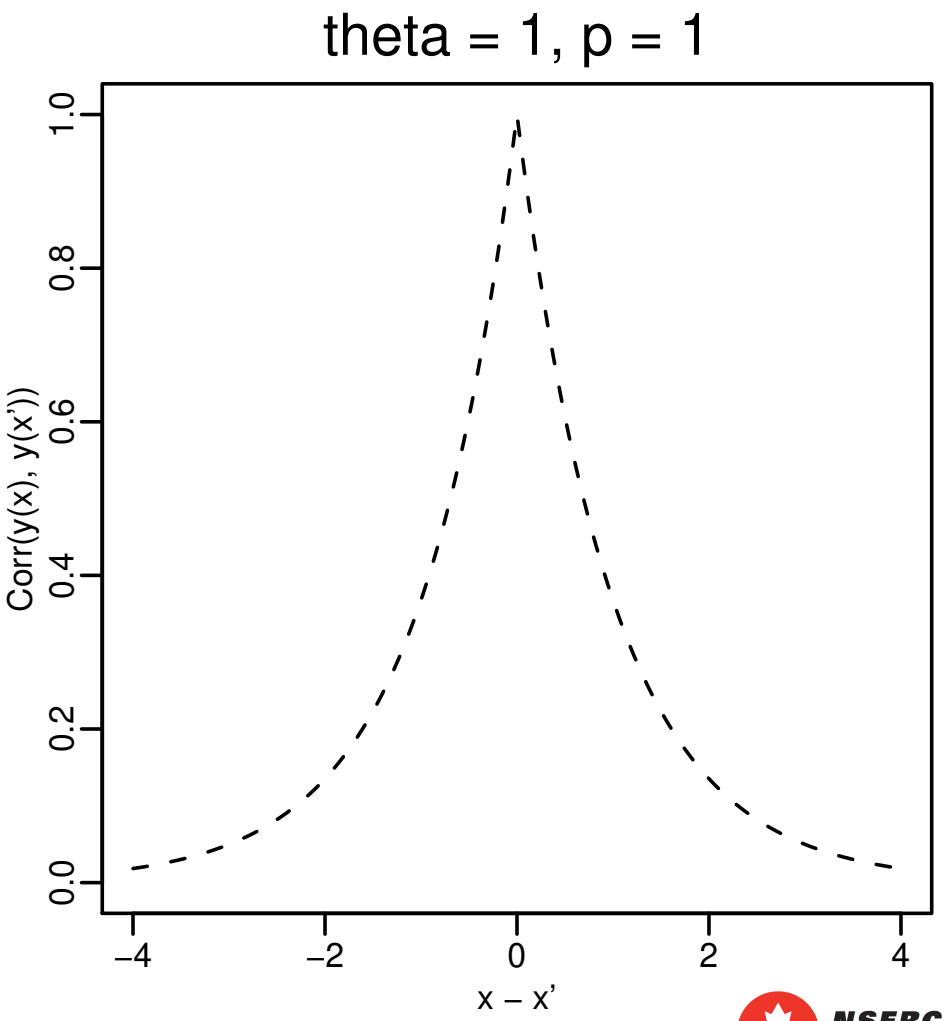
Power-Exponential Correlation Function

- A popular generalization of the squared-exponential correlation function is the power exponential.
- $R(\mathbf{x}, \mathbf{x}') = \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j'|^{2-\alpha_j})$.
- $\theta_j \geq 0$ controls the sensitivity of the GP w.r.t. x_j.
  - Larger θ_j gives smaller correlation, i.e., $y(\mathbf{x})$ and $y(\mathbf{x}')$ are less related in the x_j direction and the function is more complex.
  - $\theta_j = 0$ removes x_j (dimension reduction)
- The power $2 - \alpha_j$ (as used by GaSP) is often written as p_j.
- $\alpha_j \in [0, 1]$ affects the smoothness of the GP w.r.t. x_j.
  - $\alpha_j = 0$ (squared-exponential correlation) gives smooth realizations (with infinitely many derivatives).
  - $\alpha_j = 1$ ("exponential correlation function) gives much rougher realizations (good for continuous but non-differentiable functions).

GaSP also allows the Matérn correlation function with adjustable smoothness.

Power-Exponential Correlation Function





Why Do Properties of the Correlation Function Matter?

- The prediction function $y(\mathbf{x})$ for any new $\mathbf{x}$ is a linear combination of basis functions.
- Each basis function has the form of the correlation function.
- There is one basis function for each training data point.
- The prediction function has the same mathematical properties (smoothness) as the correlation function.
- More basis functions as n increases makes prediction automatically more complex with a larger sample.

Illustration: Damped Sin Wave

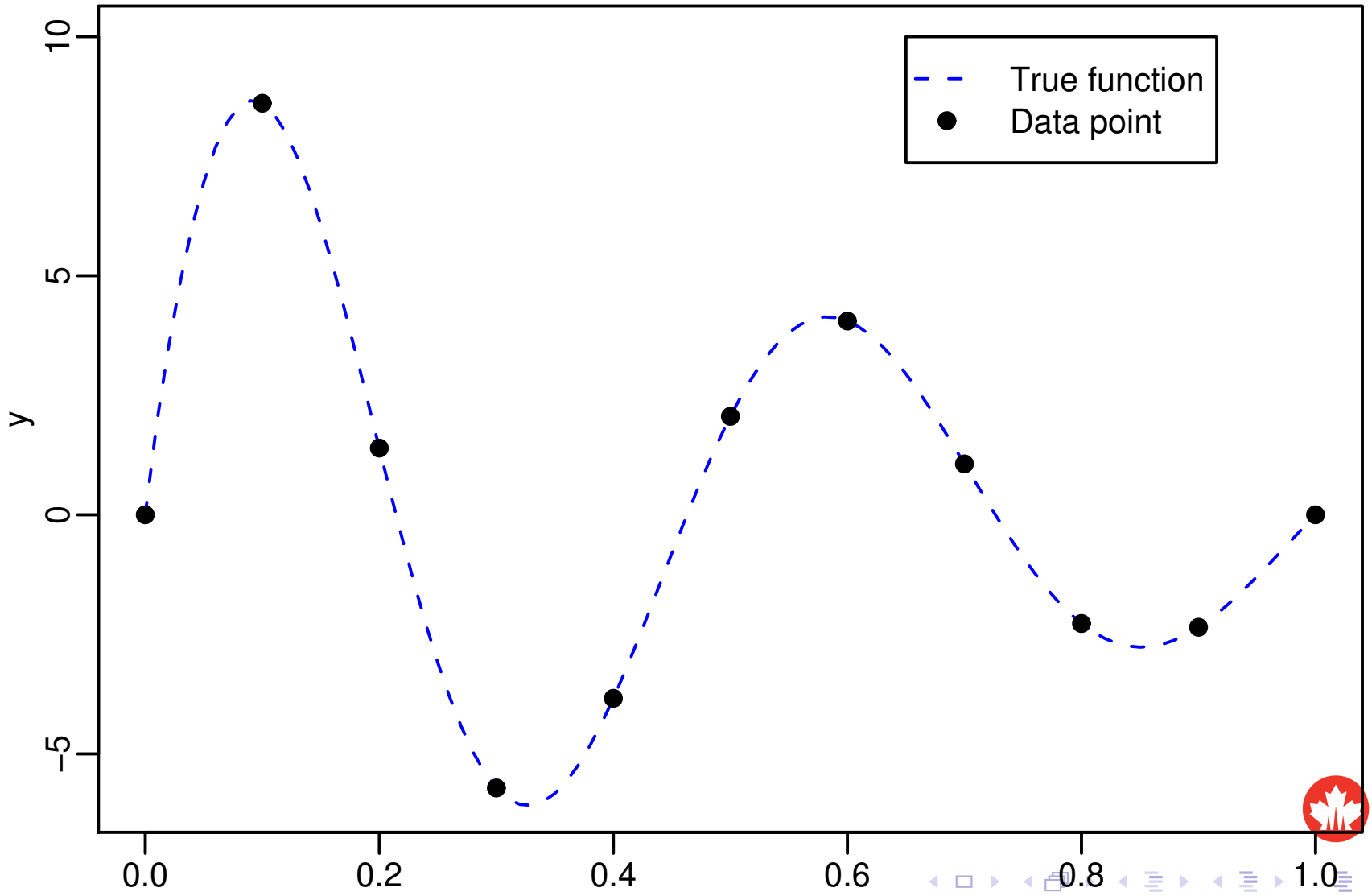
The "damped-sin" function

$$y(x) = 10\sin(4\pi x^{0.9})e^{-1.5x} \qquad (0 \leq x \leq 1)$$

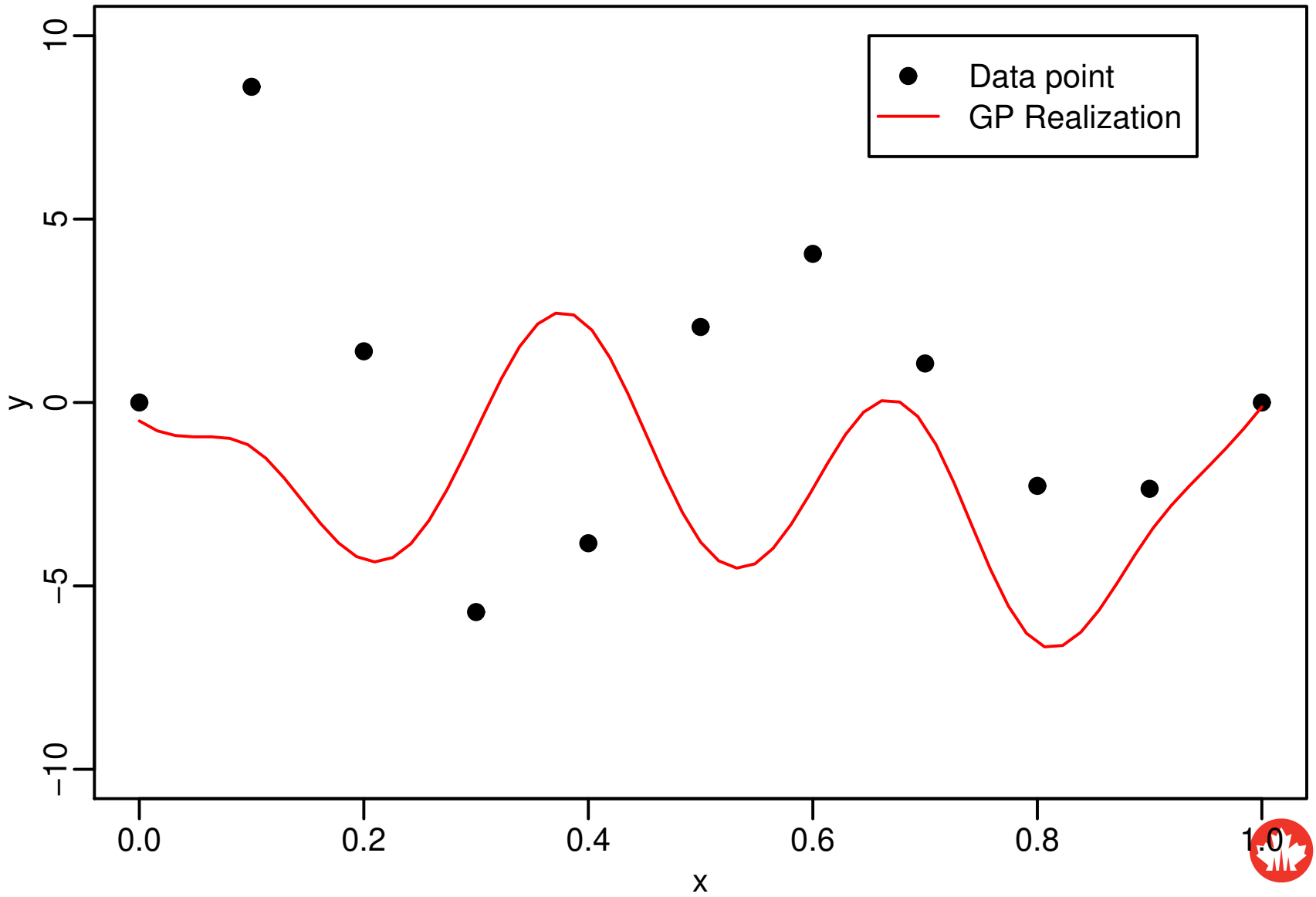
will be used to illustrate the key ideas in approximating a deterministic computer model.

- It is is highly nonlinear and hence complex.
- But it is simple:
  - It is measured without random variability (it represents a deterministic computer model).
  - $\mathbf{x}$ is only 1-d.
- We will see that the same methodolology extends to high-dimensional $\mathbf{x}$.

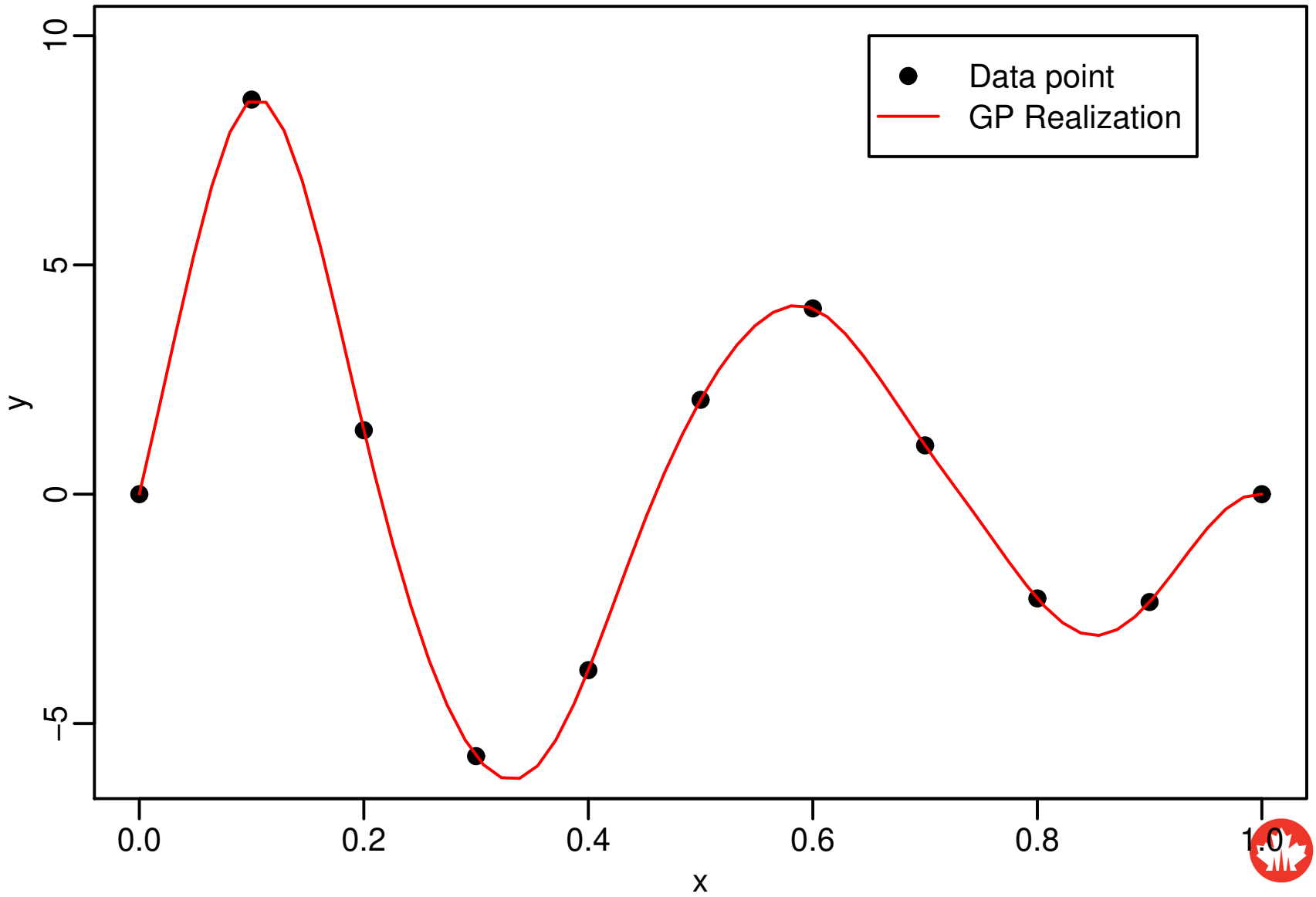
True Function Unknown in Practice! and 11 Runs of the "Code"



A Bad Realization (Inconsistent With Our Data)



A Good Realization (Consistent With Our Data)



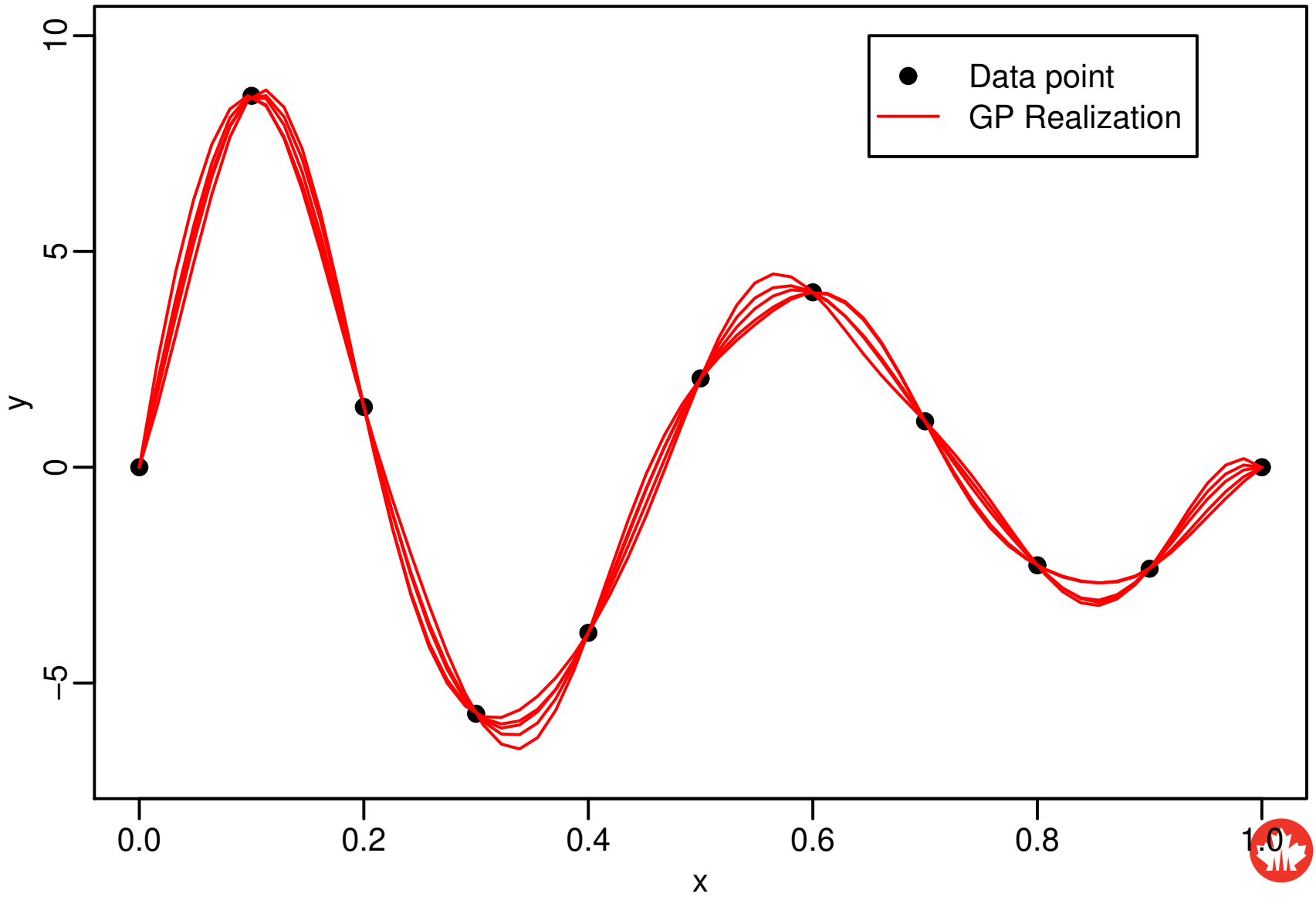
What are Good Realizations?

- $\{y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)})\}$ at any N "new" points (at which we want to predict) has a prior distribution determined through the MVN distribution.
- Get data $\{y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})\}$ by running the code at $n << N$ points (design points).
- Now have a posterior distribution of $y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)})$ (or any subset thereof) given the data. It is a conditional MVN distribution

$$\{y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(N)})\} \mid \{y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})\}.$$

- Good realizations are draws from this posterior distribution.
- Next slide has five such realizations for the damped sin example.

5 Realizations of the GP Conditional on the Data



Computing the Posterior Distribution

- In practice, we do not have to generate random realizations to predict the function.
- For simplicity, consider predicting y at any single new point, $\mathbf{x}$.
- Given the parameters $(\mu, \sigma^2, \theta, \ldots)$ of the GP, the posterior distribution of $y(\mathbf{x})$ conditional on the data is

$$y(\mathbf{x}) \mid \{y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})\} \sim \mathrm{N}(m(\mathbf{x}), s^2\mathbf{x})),$$

where

  - $m(\mathbf{x}) = \mu + \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1}(\mathbf{y} - \mu\mathbf{1})$ is the conditional mean, which provides a prediction (approximation) of $y(\mathbf{x})$
  - $s^2(\mathbf{x}) = \sigma^2(1 - \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1}\mathbf{r}(\mathbf{x}))$ is the conditional variance, which provides the variance of the prediction error.
  - $\mathbf{R} = R(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ $\quad (n \times n$ matrix$)$
  - $\mathbf{r}(\mathbf{x}) = R(\mathbf{x}^{(i)}, \mathbf{x})$ $\quad (n \times 1$ vector$)$
  - $\mathbf{1}$ is an $n \times 1$ vector of 1's.

NSERC CRSNG

Math?

According to the GP model,

- The $n+1$ random variables

$$y(\mathbf{x}), y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})$$

  have a multivariate normal distribution, and
- From the properties of the multivariate normal, the conditional distribution of

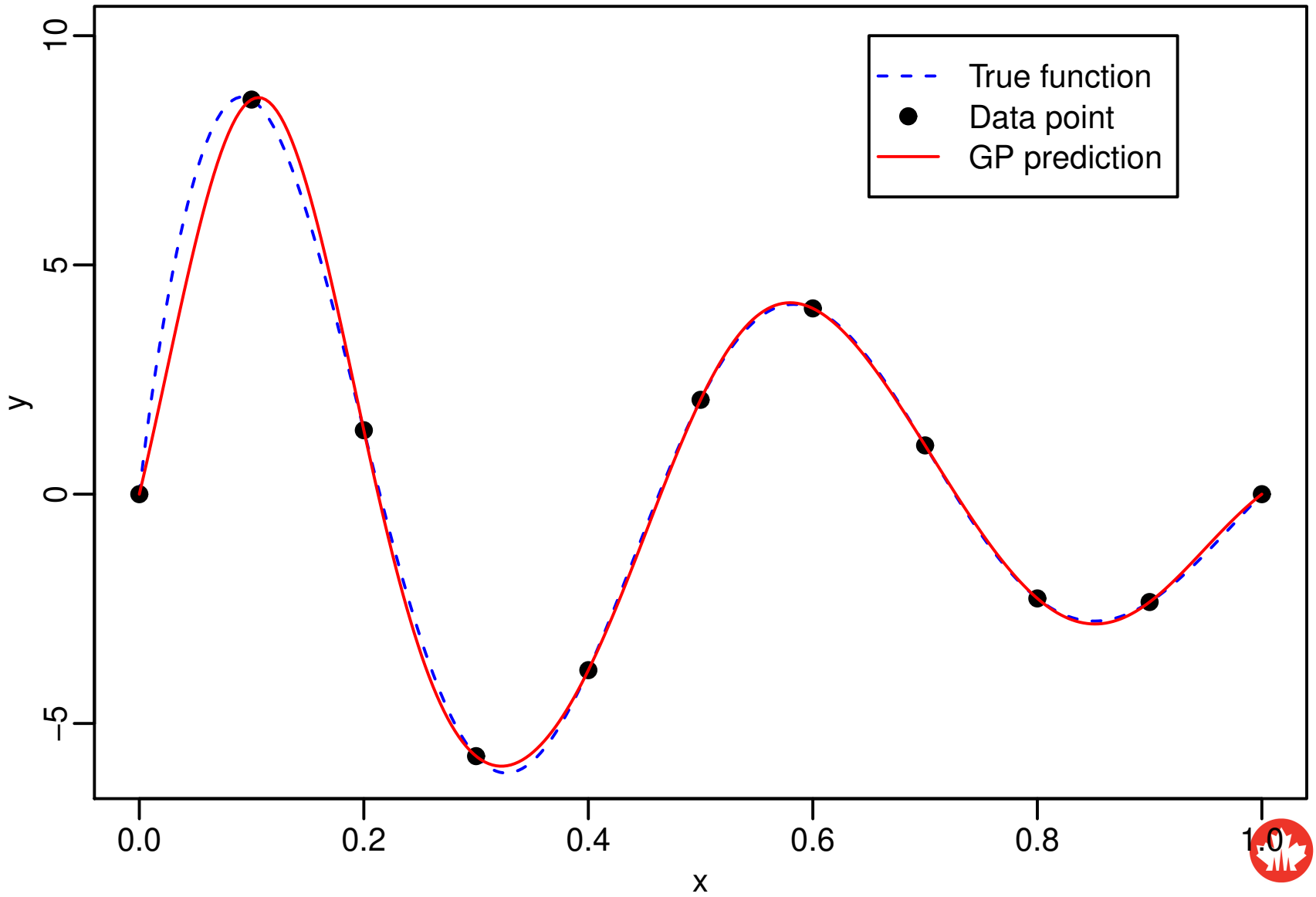
$$y(\mathbf{x}) \mid \{y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})\}$$

  is also normal (with the claimed mean and standard deviation).

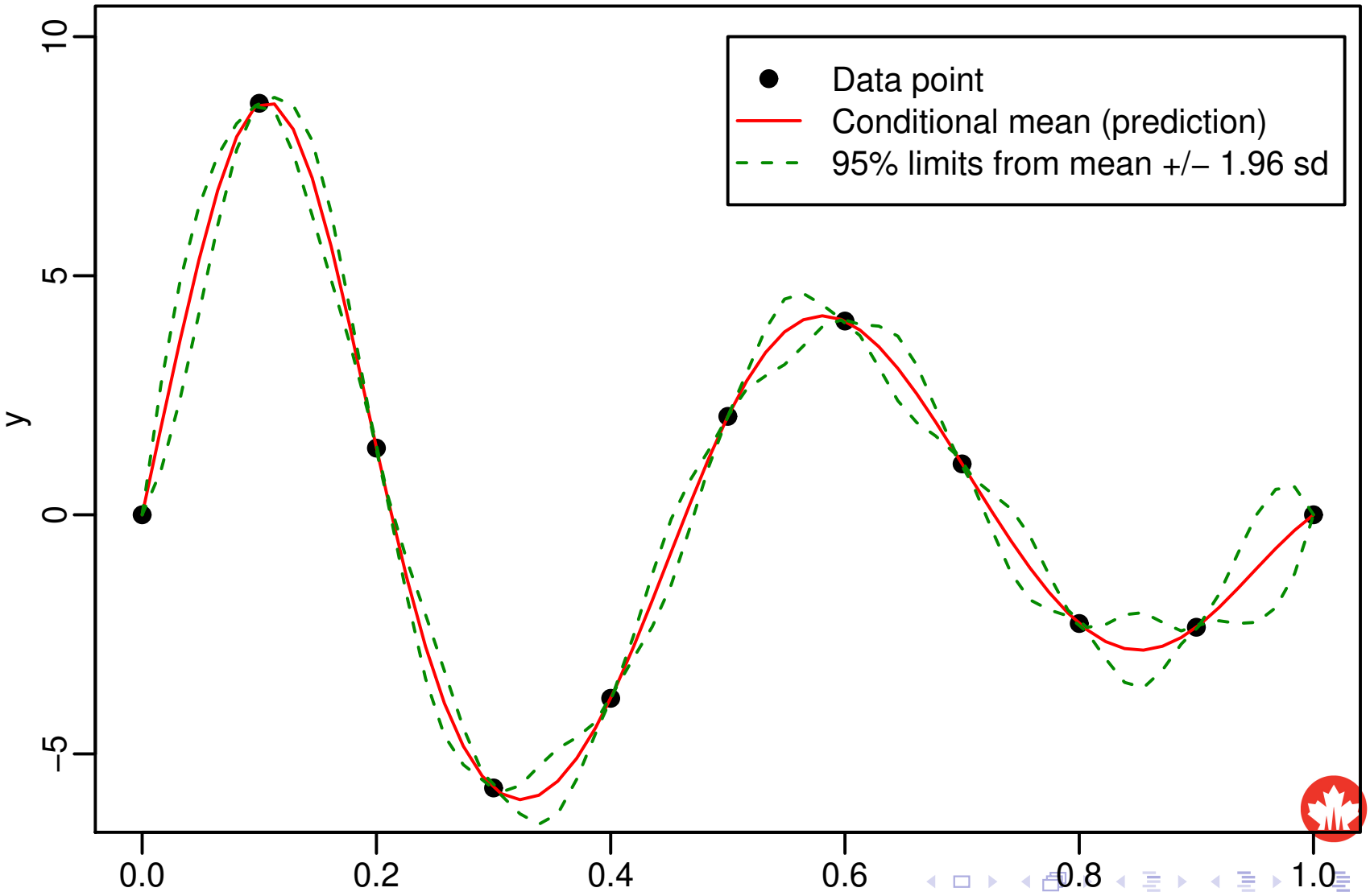
What About the Hyperparameters?

- The GP model says the training data $y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})$ are a sample from the multivariate normal distribution.
- Therefore we have a multivariate normal likelihood as a function of the hyperparameters $\mu, \sigma^2, \theta_1, \ldots, \theta_d, \ldots$
- Estimate the hyperparameters via maximum likelihood or Markov Chain Monte Carlo (MCMC).
- Over to the `GaSP` script in `dampsin.R` ...

Damped Sin: Prediction (Conditional Mean)



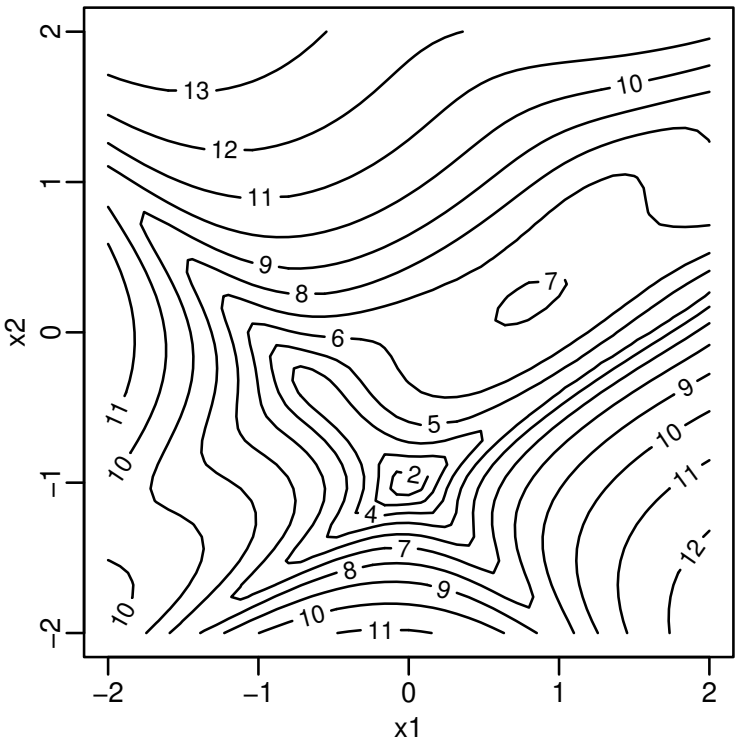
Damped Sin: Prediction With Conditional Standard Deviation



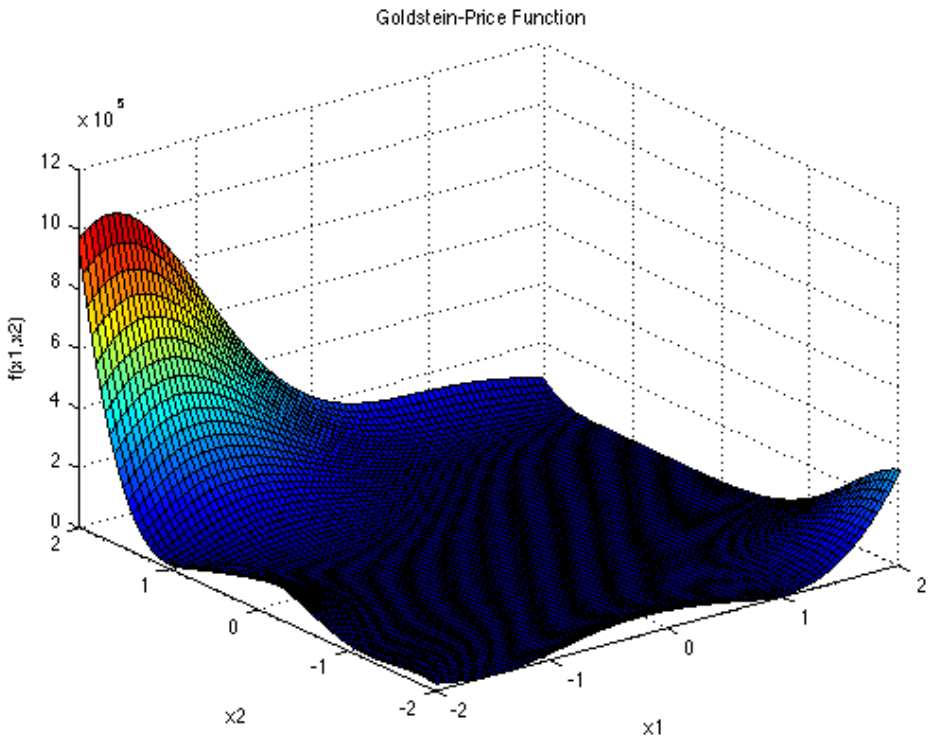
Initial Training Data for Bayesian Optimization

- Bayesian optimization is based on a GP statistical model.
- We need a starting model, trained with data from an initial design.
- How do we choose the initial design when $\mathbf{x}$ has dimension $d > 1$?

Goldstein-Price Function (To Be Optimized in Part 2)

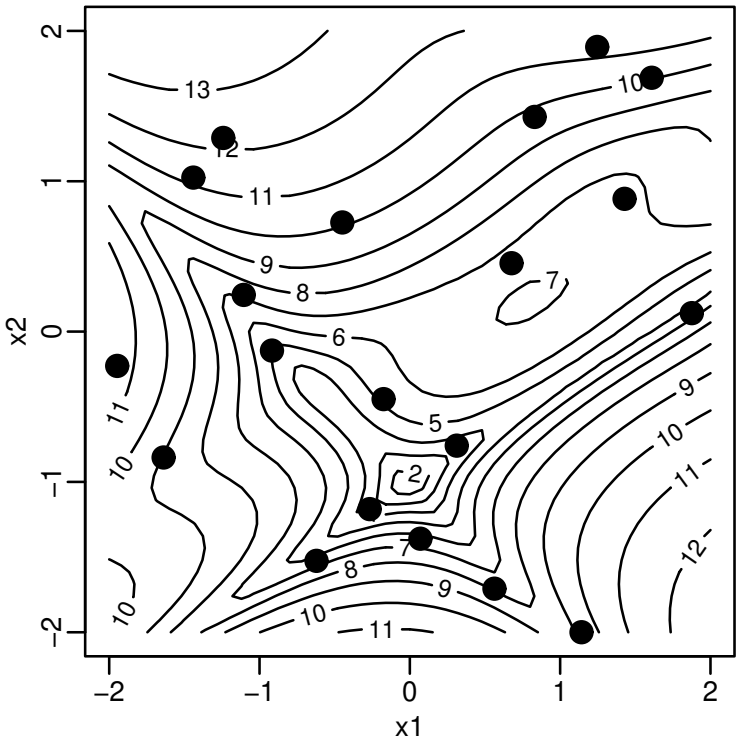


Contours in units of 10^5



https://www.sfu.ca/~ssurjano/goldpr.html

Latin Hypercube Sample

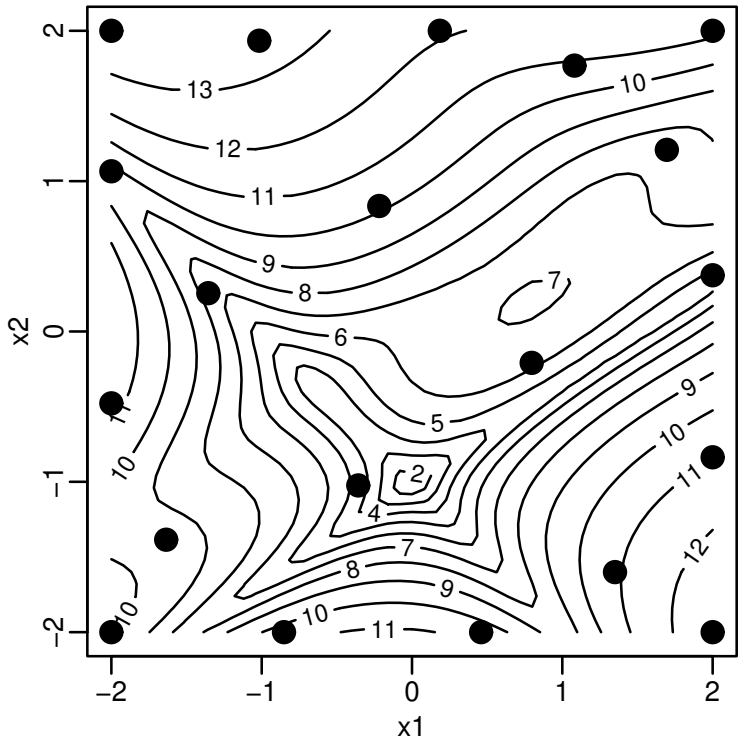


Don't know the contours in practice!

Spreads points in the input space by

- Putting x_1 on a grid; similarly x_2
- Combining the x_1 and x_2 values at random

Maximin-Distance Design

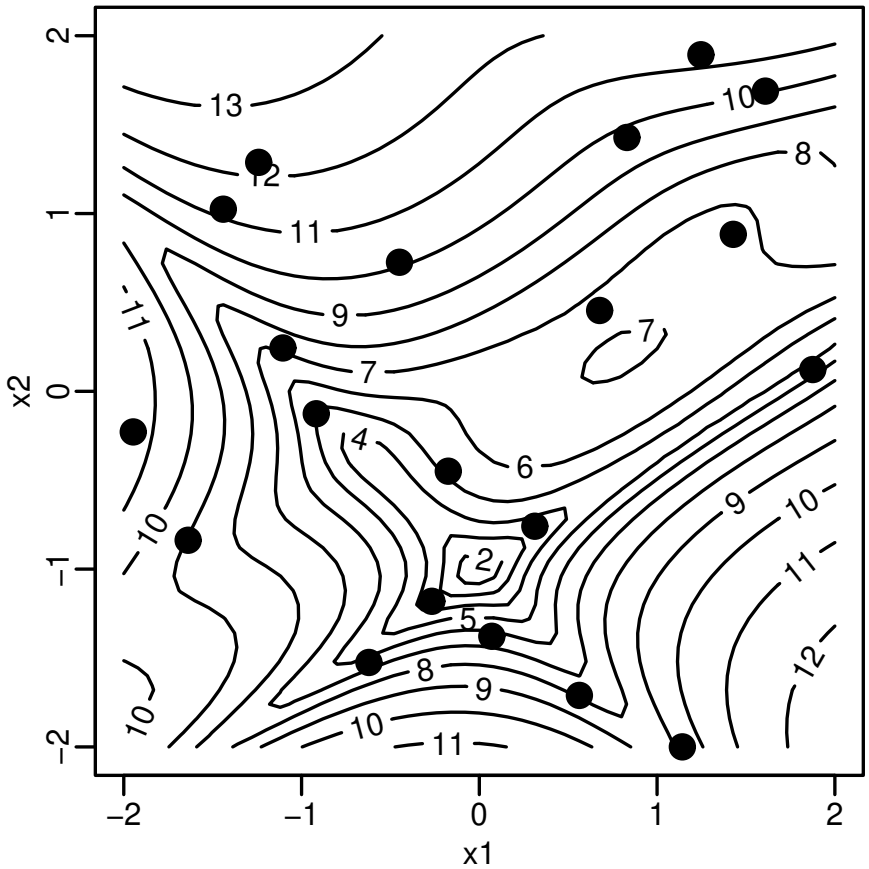


Spreads points in the input space by

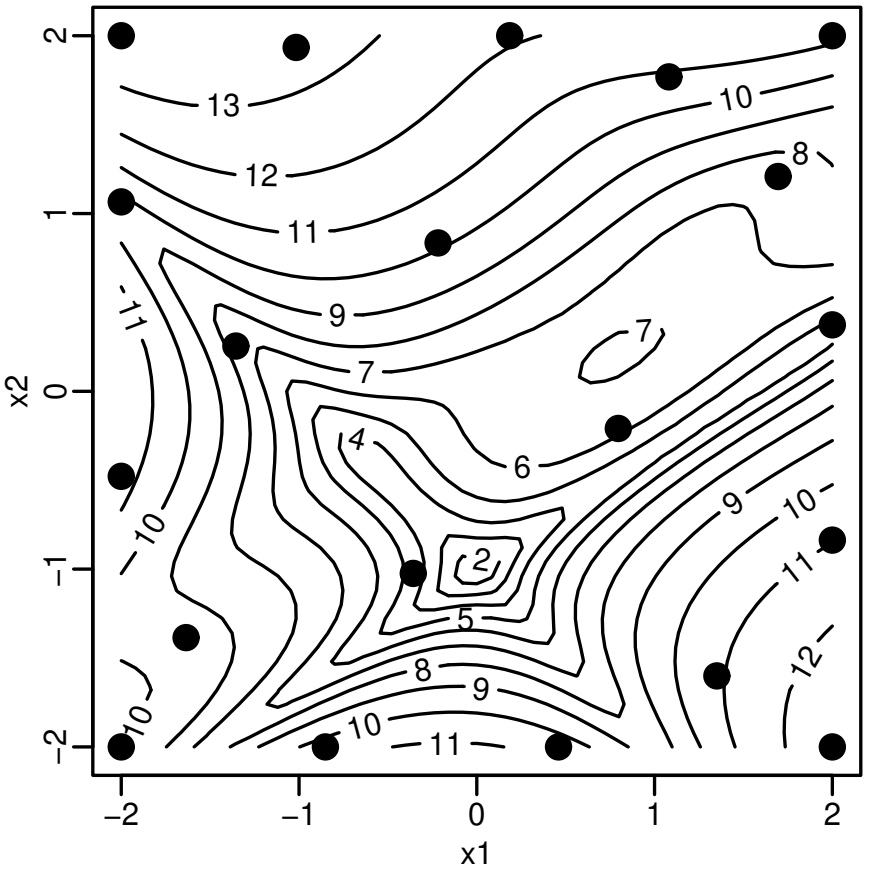
- Maximizing the minimum distance between training points

Space-Filling Designs

Latin hypercube sample



Maximin distance design



`randomLHS()` in R

`maximin()` in R

Space-Filling Designs

- There are many other variations!
- Don't worry too much about your choice, as long as it is "space-filling".
  - But best to avoid standard experimental designs like 2- or 3-level fractional factorials.
- Adaptive design, e.g., Bayesian optimization (Part 2), is where differences between methods matter.

Summary (Part 1)

- Treating the code input-output function as if it is a realization of a Gaussian process (GP) is a probability model for the unknown function.
- The probability model leads to statistical properties of predictions.
- We have a multivariate normal likelihood for the data; the correlation-function (hyper) parameters can be estimated by maximum likelihood or MCMC.
- Predict $y(\mathbf{x})$ by the mean of the conditional distribution given the data.
- An uncertainty measure comes from the conditional standard deviation.
- The model is flexible and data adaptive.
- Experimental designs should be space-filling.

References I

Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1991), "Bayesian Prediction of Deterministic Functions, With Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association*, 86, 953–963.

Rasmussen, C. E. and Williams, C. K. I. (2006), *Gaussian Processes for Machine Learning*, Cambridge, MA: The MIT Press.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989), "Design and Analysis of Computer Experiments," *Statistical Science*, 4, 409–423.

Welch, W. J. and Yang, Y. (2021), *GaSP: Train and Apply a Gaussian Stochastic Process Model*, R package version 1.0.4.

Dealing With Random Error

Suppose we observe

$$y(\mathbf{x}) + \text{random measurement noise.}$$

Simply model the data as a realization of prior for $y + \epsilon$, where ϵ is independent Gaussian error with mean zero and variance σ_ϵ^2.
In formulas replace

$$\begin{aligned}
\sigma^2 \quad &\text{with} \quad \sigma^2_{\text{Total}} = \sigma^2 + \sigma^2_\epsilon \\
\mathbf{R} \quad &\text{with} \quad \frac{\sigma^2}{\sigma^2_{\text{Total}}}\mathbf{R} + \frac{\sigma^2_\epsilon}{\sigma^2_{\text{Total}}}\mathbf{I}_{n\times n} \\
\mathbf{r}(\mathbf{x}) \quad &\text{with} \quad \frac{\sigma^2}{\sigma^2_{\text{Total}}}\mathbf{r}(\mathbf{x}),
\end{aligned}$$

where $\mathbf{I}_{n\times n}$ is an $n \times n$ identity matrix.