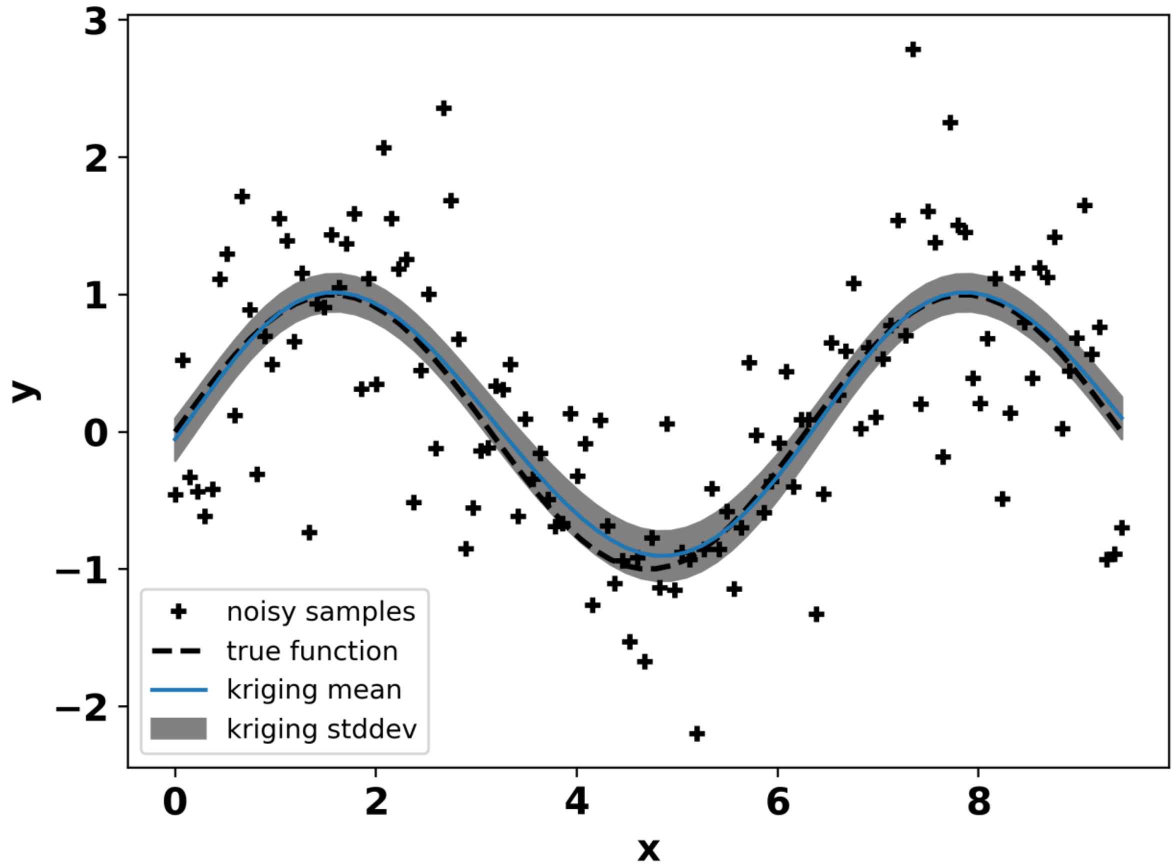
Kriging
a.k.a. Gaussian Process Regression(GPR)

Yubo “Paul” Yang, Algorithm Interest Group, Jan. 17 2019



What is kriging?

Kriging is an interpolation method.

Kriging minimizes the variance of prediction error at data points.

Kriging provides uncertainty estimates to its predictions.

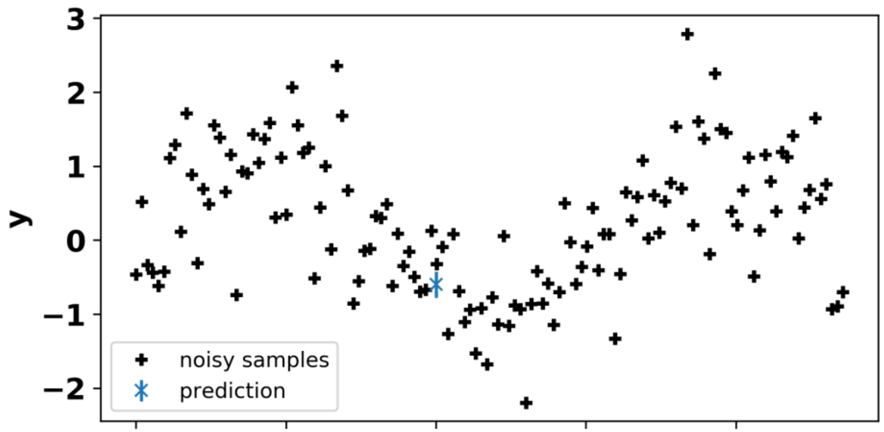
Problem:
Given samples of a random scalar field, predict value at un-sampled location.

Solution:
Design unbiased estimator with minimum prediction error variance.

$$\hat{y}_0 = \boldsymbol{w}^T \boldsymbol{y}$$

$$E[\hat{y}_0] = E[y_0]$$

Minimize $V[\hat{y}_0 - y_0]$



Why krige?

Interpolation is generally useful.

Error estimates on predictions.

Gold!

Photo by Sharon McCutcheon on Unsplash

How to krige? : ordinary

Derive ordinary kriging equation: assume stationary mean $E[y_i] = \mu, \forall i$

$$\hat{y}_0 = \boldsymbol{w}^T \boldsymbol{y} = \sum_{i=1}^{N} w_i \, y_i$$

1. Design unbiased estimator $E[\hat{y}_0] = E[y_0] = \mu$

$$E[\hat{y}_0] = E\left[\sum_{i=1}^{N} w_i \, y_i\right] = \sum_{i=1}^{N} w_i \, E[y_i] = \mu \sum_{i=1}^{N} w_i$$

$$\boxed{\sum_{i=1}^{N} w_i = 1}$$

2. Minimize prediction error variance $V[\hat{y}_0 - y_0]$

$$V[y_0 - \hat{y}_0] = Cov[y_0, y_0] - 2Cov[y_0, \hat{y}_0] + Cov[\hat{y}_0, \hat{y}_0]$$

$$Cov[y_0, \hat{y}_0] = Cov\left[y_0, \textstyle\sum_{i=1}^{N} w_i \, y_i\right] = \textstyle\sum_{i=1}^{N} w_i Cov[y_0, y_i] = \boldsymbol{w}^T \boldsymbol{d} \qquad d_i \equiv Cov[y_0, y_i]$$

$$Cov[\hat{y}_0, \hat{y}_0] = \boldsymbol{w}^T \boldsymbol{C} \, \boldsymbol{w} \qquad C_{ij} \equiv Cov[y_i, y_j]$$

$$\boxed{V[y_0 - \hat{y}_0] = C_{00} - 2\boldsymbol{w}^T \boldsymbol{d} + \boldsymbol{w}^T \boldsymbol{C} \, \boldsymbol{w}}$$

[1] Wikipedia
[2] Kriging Example
[3] Ordinary Kriging by MSU Ashton Shortridge

How to krige? : ordinary

Ordinary kriging equation: assume stationary mean $E[y_i] = \mu, \forall i$

minimize $\boxed{V[y_0 - \hat{y}_0] = C_{00} - 2\boldsymbol{w}^T\boldsymbol{d} + \boldsymbol{w}^T\boldsymbol{C}\,\boldsymbol{w}}$

With the constant $\boxed{\sum_{i=1}^{N} w_i = 1}$

Solve ordinary kriging equation using Lagrange multiplier λ

minimize $V[y_0 - \hat{y}_0] - 2\lambda(\boldsymbol{1}^T\boldsymbol{w} - 1)$

$$\left[\begin{array}{ccc|c} C_{11} & \cdots & C_{1n} & 1 \\ \vdots & & \vdots & \vdots \\ C_{1n} & \cdots & C_{nn} & 1 \\ \hline 1 & \cdots & 1 & 0 \end{array}\right] \left[\begin{array}{c} w_1 \\ \vdots \\ w_n \\ \hline \lambda \end{array}\right] = \left[\begin{array}{c} C_{10} \\ \vdots \\ C_{n0} \\ \hline 1 \end{array}\right]$$

$$\begin{bmatrix} \boldsymbol{C} & \boldsymbol{1} \\ \boldsymbol{1}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \lambda \end{bmatrix} = \begin{bmatrix} \boldsymbol{d} \\ 1 \end{bmatrix}$$

[1] Wikipedia
[2] Kriging Example
[3] Ordinary Kriging by MSU Ashton Shortridge

How to krige? : ordinary

Ordinary kriging equation: assume stationary mean $E[y_i] = \mu, \forall i$

$$\begin{bmatrix} \boldsymbol{C} & \boldsymbol{1} \\ \boldsymbol{1}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \lambda \end{bmatrix} = \begin{bmatrix} \boldsymbol{d} \\ 1 \end{bmatrix}$$

solution

$$\lambda = \frac{\boldsymbol{1}^T \boldsymbol{C}^{-1} \boldsymbol{d} - 1}{\boldsymbol{1}^T \boldsymbol{C}^{-1} \boldsymbol{1}} \qquad \boldsymbol{w} = \boldsymbol{C}^{-1}(\boldsymbol{d} - \lambda \boldsymbol{1})$$

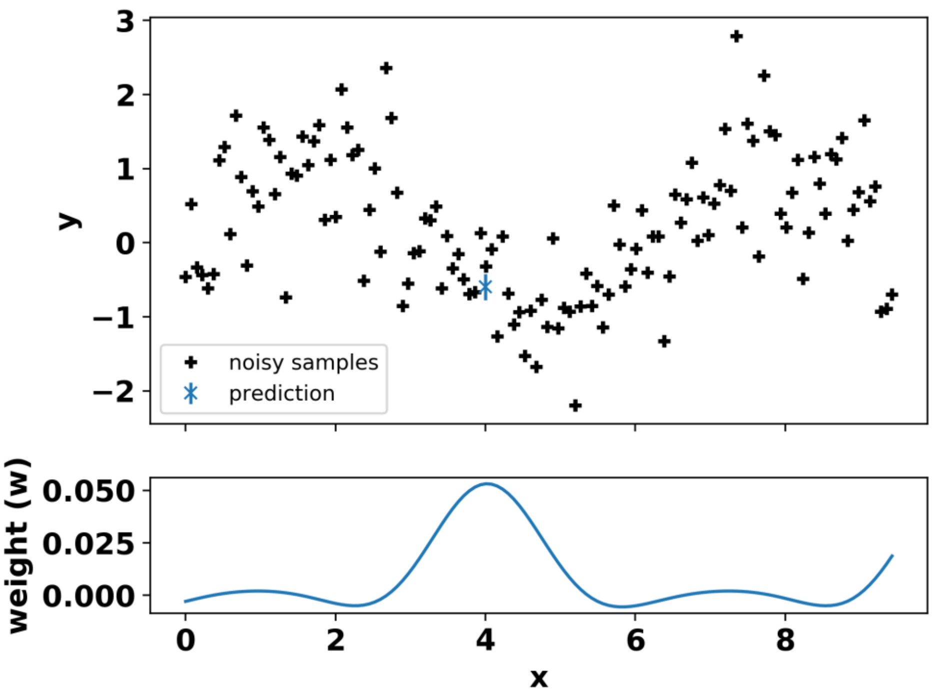
implementation

$$E[\hat{y}_0] = \boldsymbol{w}^T \boldsymbol{y}$$

$$V[\hat{y}_0 - y_0] = C_{00} - \boldsymbol{w}^T \boldsymbol{d} - \lambda$$

```python
def ordinary_krig(x0, x, y, myc, invc):
  from scipy.linalg import cho_solve
  dvec = [myc(x0, x1) for x1 in x]
  # calculate Lagrange multiplier
  ovec = np.ones(len(dvec))
  nume = np.dot(np.dot(dvec, invc), ovec) - 1
  deno = np.dot(np.dot(ovec, invc), ovec)
  lam = nume/deno
  # calculate weights
  wvec = np.dot(invc, dvec-lam*ovec)
  if not np.isclose(wvec.sum(), 1):
    raise RuntimeError('weight constraint failed')
  ym = np.dot(wvec, y)
  sig2 = myc(x0, x0)-np.dot(dvec, wvec)-lam
  ye = np.sqrt(sig2)
  return ym, ye, wvec
```

result



[1] Wikipedia
[2] Kriging Example
[3] Ordinary Kriging by MSU Ashton Shortridge

How to krige? : simple

Simple kriging: assume $E[y_i] = 0, \forall i \Rightarrow$ no constraint on weights

$$\begin{bmatrix} \boldsymbol{C} & \boldsymbol{1} \\ \boldsymbol{1}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \lambda \end{bmatrix} = \begin{bmatrix} \boldsymbol{d} \\ 1 \end{bmatrix} \quad \text{becomes} \quad \boldsymbol{C}\boldsymbol{w} = \boldsymbol{d}$$

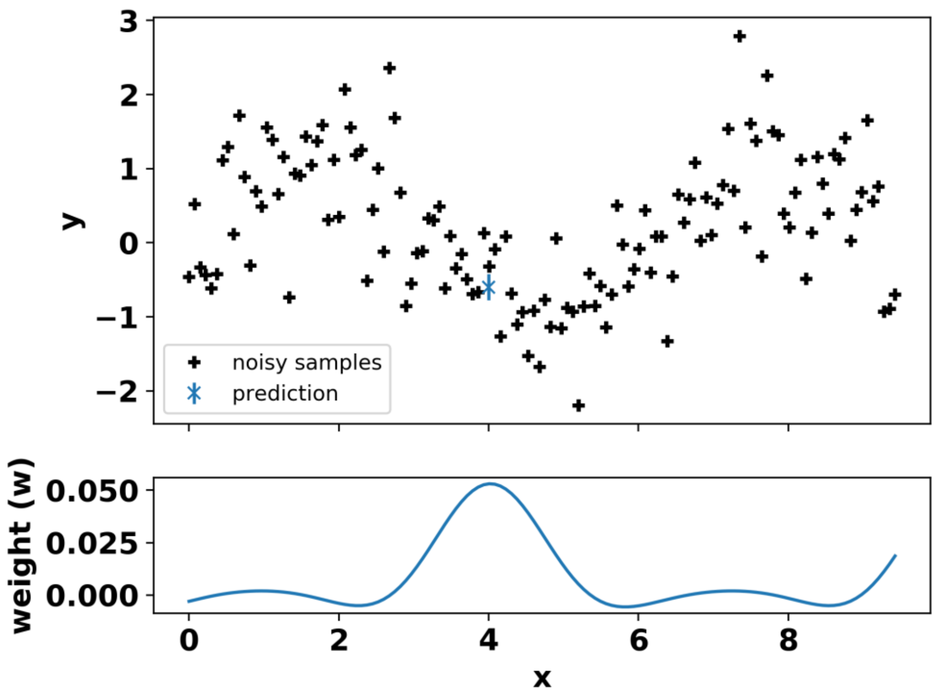
implementation

$$E[\hat{y}_0] = \boldsymbol{w}^T \boldsymbol{y}$$

$$V[\hat{y}_0 - y_0] = C_{00} - \boldsymbol{w}^T \boldsymbol{d}$$

```python
def simple_krig(x0, x, y, myc, lmat):
  from scipy.linalg import cho_solve
  dvec = [myc(x0, x1) for x1 in x]
  wvec = cho_solve((lmat, True), dvec)
  ym = np.dot(wvec, y)
  sig2 = myc(x0, x0)-np.dot(dvec, wvec)
  ye = np.sqrt(sig2)
  return ym, ye, wvec
```

result



The secret sauce: correlation function

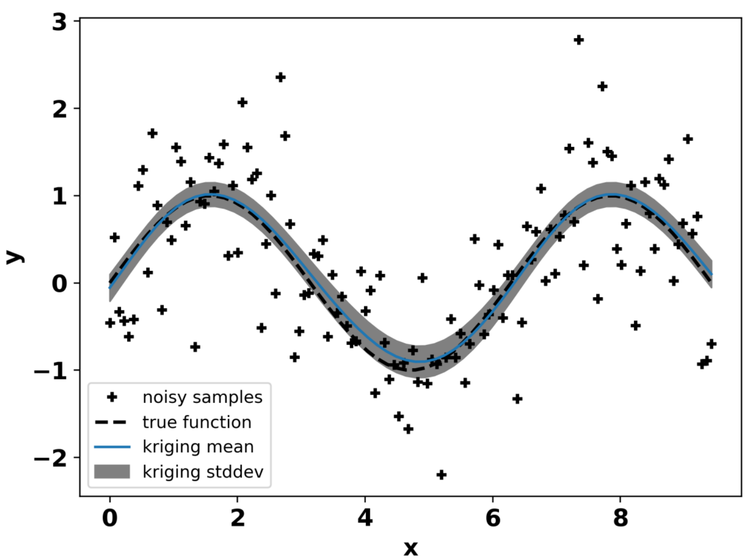
The correlation function $Cov(\boldsymbol{x}_1, \boldsymbol{x}_2)$ should capture covariance in data. (CM people think g(**r**))

$Cov(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is used to build the $\boldsymbol{C}$ matrix and the $\boldsymbol{d}$ vector.

$Cov(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is related to the so-called variogram by $\gamma(\boldsymbol{x}_1, \boldsymbol{x}_2) = Cov(\boldsymbol{x}_0, \boldsymbol{x}_0) - Cov(\boldsymbol{x}_1, \boldsymbol{x}_2)$

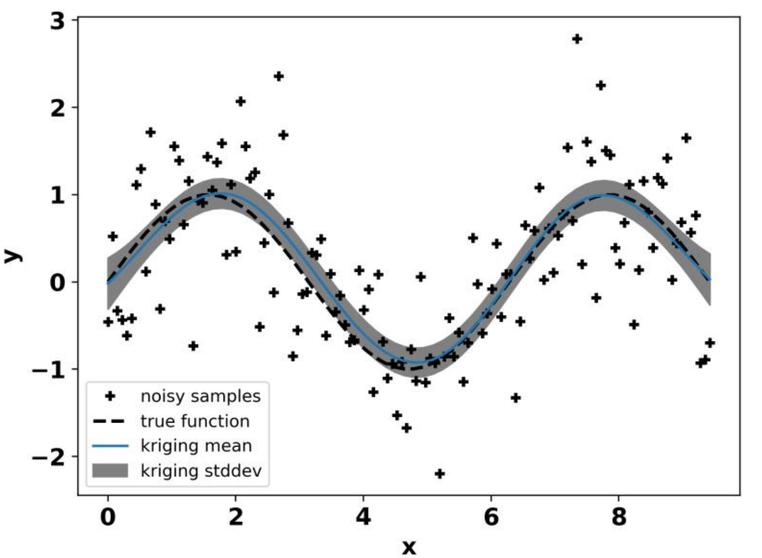
Exponential sine squared

$$Cov(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp\left(-2\left(\sin\frac{\left(\frac{\pi}{T}|\boldsymbol{x}_1 - \boldsymbol{x}_2|\right)}{L}\right)^2\right)$$



Squared exponential

$$Cov(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp\left(-\frac{(\boldsymbol{x}_1 - \boldsymbol{x}_2)^2}{2L^2}\right)$$

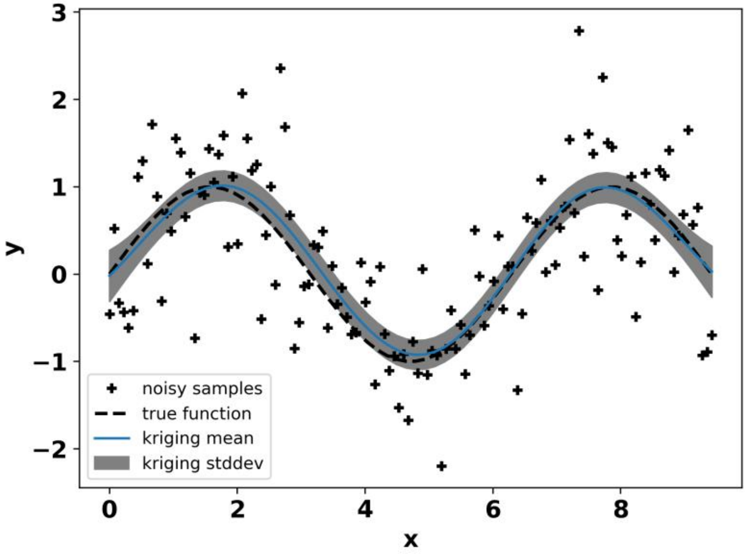


The secret sauce: correlation function

A kriging expert knows how to choose the correlation function **form** and *parameters*.

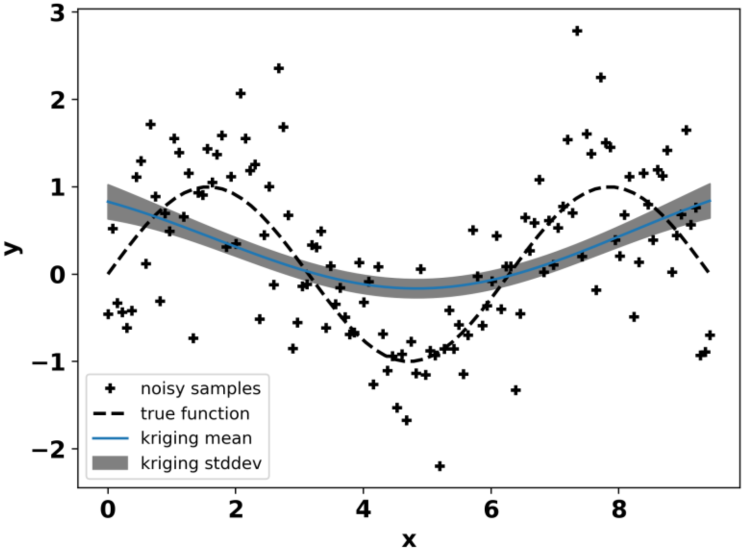
Squared exponential L=1.5

$$Cov(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp\left(-\frac{(\boldsymbol{x}_1 - \boldsymbol{x}_2)^2}{2L^2}\right)$$



Squared exponential L=5.0

$$Cov(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp\left(-\frac{(\boldsymbol{x}_1 - \boldsymbol{x}_2)^2}{2L^2}\right)$$



Historical Review

Kriging was used for time series analysis back in the 1940s.

Kriging got its name from Danie G. Krige’s master thesis for predicting the location of gold deposits in South Africa in 1960.

Kriging is used extensively in geostatistics and meteorology.

Kriging was reformulated in the context of Baysian inference in the late 1990s.

Kriging is now known as Gaussian process regression.
The choice of correlation function is phrased as a machine learning problem.

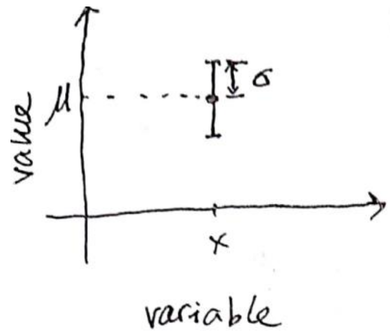
[1] Wikipedia
[2] Chapter 2.8 RW2006

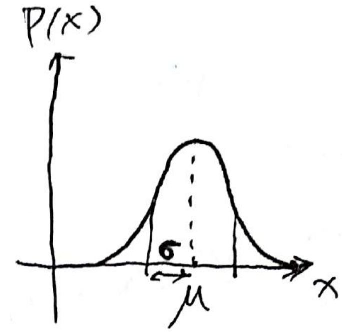
Gaussian Process

Gaussian process is the generalization of multivariate distribution to infinite variables.

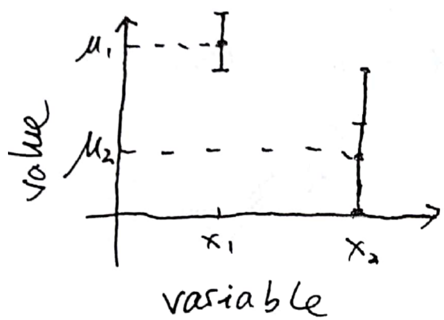
Gaussian process is probability distribution over functions.

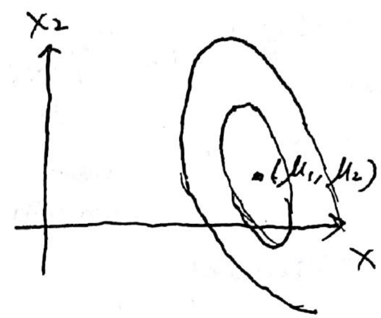
Gaussian variable
Normal distribution



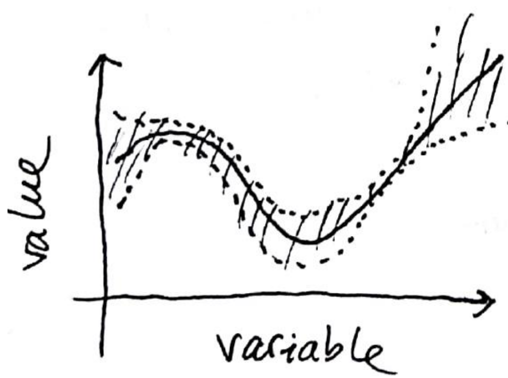


Gaussian vector
Multivariate distribution



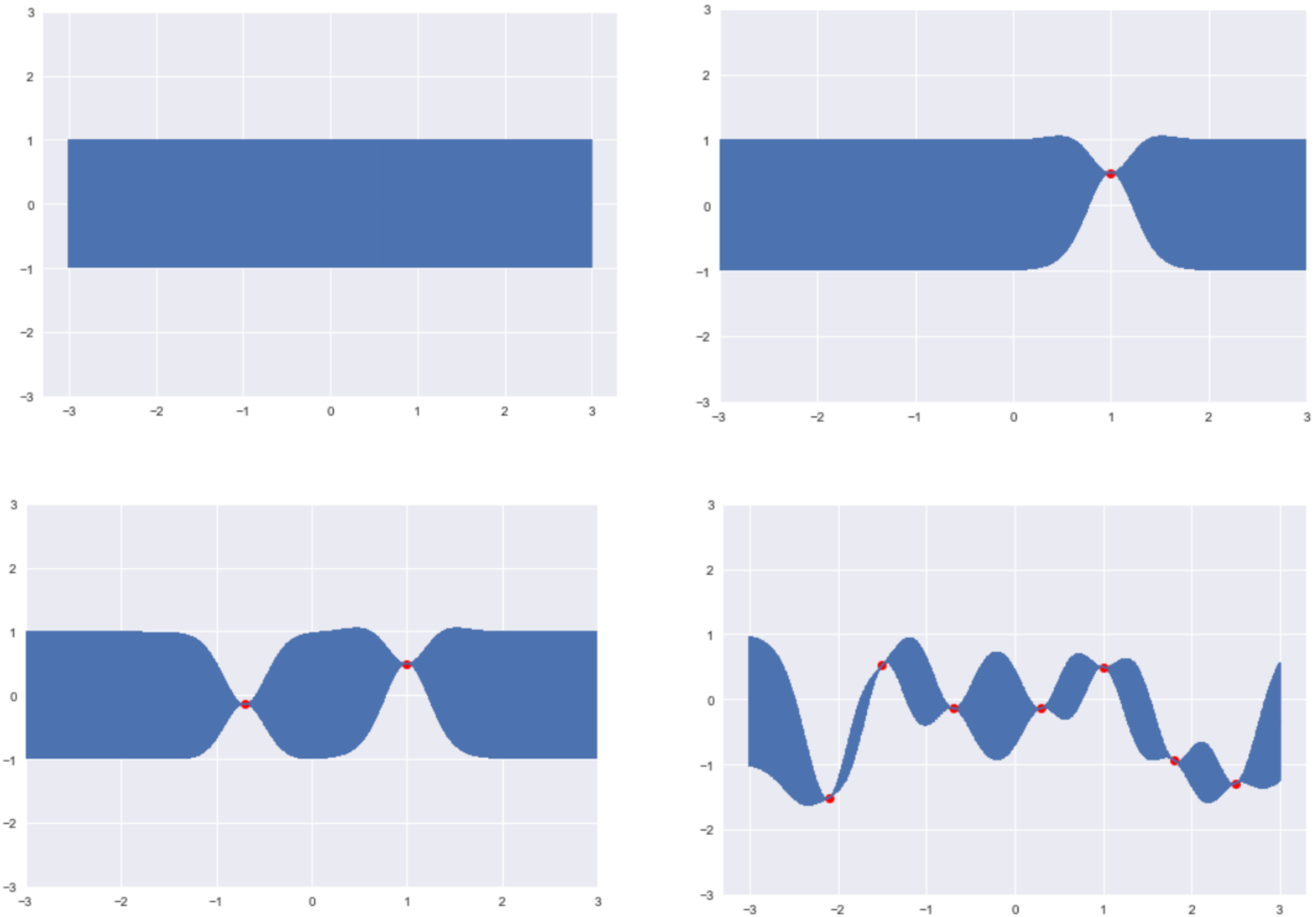


Gaussian process



[1] Chapter 2.2 RW2006

Gaussian Process Regression



[1] [Chris Fonnsbeck blog]

Recent Applications

[1] A. P. Bartok et. al., “Machine Learning a General-Purpose Interatomic Potential for Silicon,” Phys. Rev. X **8**, 041048 (2018).

[2] A. Kamath et. al., “Neural networks vs Gaussian process regression for representing potential energy surfaces: A comparative study of fit quality and vibrational spectrum accuracy,” J. Chem. Phys. **148**, 241702 (2018).

[3] A. Denzel and J. Kastner, “Gaussian Process Regression for Transition State Search,” J. Chem. Theory Comput., 14 (11), pp 5777-5786 (2018).

[4] G. Schmitz and O. Christiansen, “Gaussian process regression to accelerate geometry optimization relying on numerical differentiation,” J. Chem. Phys. **148**, 241704 (2018).

Conclusions

Kriging is a minimal-variance unbiased interpolation algorithm.

Kriging result depends critically on the choice of correlation function (variogram).

Kriging outputs a Gaussian process.
Recently combined with Baysian inference and machine learning.

Problem:
Given samples of a random scalar field, predict value at un-sampled location.

Solution:
Design unbiased estimator with minimum prediction error variance.

$$\hat{y}_0 = \boldsymbol{w}^T \boldsymbol{y}$$

$$E[\hat{y}_0] = E[y_0]$$

Minimize $V[\hat{y}_0 - y_0]$