

# MATLAB Parametric Empirical Kriging (MPeRk) User's Guide

Gang Han and Thomas J. Santner

May 14, 2008

MPeRk (Parametric Empirical Kriging) is MATLAB program written for predicting the output from a computer experiment having quantitative inputs. The program implements several user specified choices of the empirical Best Linear Unbiased Prediction. The program

- gives MLE or REML estimates of the correlation parameters for the Cubic, Gaussian, Power Exponential correlation functions based on user-specified linear regression and stationary Gaussian stochastic process models,
- estimates the mean and variance of the Gaussian Stochastic process,
- computes the Best Linear unbiased predictions and associated prediction errors models,
- optionally perform cross validation estimation on the training data.

This manual provides instructions for running the program. Section 1 introduces the Gaussian stochastic process models that are fit. Section 2 describes the MATLAB files used by MPeRk and the inputs/outputs of the program. Section 3 illustrates the program with examples. Section 4 discusses some technical details of the program.

## 1 Introduction

MPeRk views the output from a computer experiment,  $y(\mathbf{x})$ , as a realization from a Gaussian stochastic process

$$Y(\mathbf{x}) = \mathbf{f}^\top(\mathbf{x})\boldsymbol{\beta} + Z(\mathbf{x}), \quad \mathbf{x} \in [0, 1]^d \subset \mathcal{R}^d, \quad (1)$$

where  $\mathbf{f}(\mathbf{x})$  is a known vector of regression functions (usually the constant 1 and possibly other monomials of  $\mathbf{x}$ ),  $\boldsymbol{\beta}$  is an unknown vector of regression parameters,  $Z(\mathbf{x})$  is a zero-mean Gaussian stochastic process with unknown variance  $\sigma^2$  and correlation function  $R(\cdot|\boldsymbol{\xi})$  with correlation parameters  $\boldsymbol{\xi}$ . For two inputs  $\mathbf{x}_1$  and  $\mathbf{x}_2$ ,

$$Cov(Z(\mathbf{x}_1), Z(\mathbf{x}_2)) = \sigma^2 R(\mathbf{x}_1 - \mathbf{x}_2|\boldsymbol{\xi}),$$

where  $R(\cdot|\boldsymbol{\xi})$  is allowed to be one of the three correlation functions:

### 1. Gaussian correlation

$$R(\mathbf{x}_1 - \mathbf{x}_2 | \boldsymbol{\xi}) = \prod_{i=1}^d \exp[-\theta_i (x_{1,i} - x_{2,i})^2], \quad (2)$$

where  $\boldsymbol{\xi} = \boldsymbol{\theta} = (\theta_1, \dots, \theta_d)$  and  $\theta_i \geq 0$  for  $i \in \{1, \dots, d\}$ .

### 2. Product Power Exponential correlation

$$R(\mathbf{x}_1 - \mathbf{x}_2 | \boldsymbol{\xi}) = \prod_{i=1}^d \exp[-\theta_i |x_{1,i} - x_{2,i}|^{p_i}], \quad (3)$$

where  $\boldsymbol{\xi} = (\boldsymbol{\theta}, \mathbf{p}) = (\theta_1, \dots, \theta_d, p_1, \dots, p_d)$  with  $\theta_i \geq 0$  and  $0 < p_i \leq 2$ .

### 3. Cubic correlation

$$R(\mathbf{x}_1 - \mathbf{x}_2 | \boldsymbol{\xi}) = \prod_{i=1}^d R(x_{1,i} - x_{2,i} | \theta_i), \quad (4)$$

$$R(x_{1,i} - x_{2,i} | \theta_i) = \begin{cases} 1 - 6(x_{i,1} - x_{i,2})^2 / \theta_i^2 + 6|x_{i,1} - x_{i,2}|^3 / \theta_i^3, & \text{if } |x_{i,1} - x_{i,2}| \leq \theta_i/2, \\ 2(1 - |x_{i,1} - x_{i,2}| / \theta_i)^3, & \text{if } \theta_i/2 \leq |x_{i,1} - x_{i,2}| \leq \theta_i, \\ 0, & \text{if } \theta_i \leq |x_{i,1} - x_{i,2}|, \end{cases} \quad (5)$$

where  $\boldsymbol{\xi} = \boldsymbol{\theta} = (\theta_1, \dots, \theta_d)$  and  $\theta_i \geq 0$ .

Estimates of the correlation parameters  $\boldsymbol{\xi}$  are obtained by either maximum likelihood (ML) or restricted maximum likelihood (REML). Optimizations of the correlation parameters are carried out by the MATLAB function `fmincon`. Some details of `fmincon` are given in Section 4.

After estimating the model parameters, MPErK can predict an unknown output  $Y(\mathbf{x}_0)$  given the training data  $\mathbf{Y}^n = (Y(\mathbf{x}_1), \dots, Y(\mathbf{x}_n))$ . Assume that

$$\begin{pmatrix} Y(\mathbf{x}_0) \\ \mathbf{Y}^n \end{pmatrix} \sim N_{n+1} \left[ \begin{pmatrix} \mathbf{f}_0^\top \\ \mathbf{F} \end{pmatrix} \boldsymbol{\beta}, \sigma^2 \begin{pmatrix} 1 & \mathbf{r}_0^\top \\ \mathbf{r}_0 & \mathbf{R} \end{pmatrix} \right], \quad (6)$$

where

- $\mathbf{f}_0 = \mathbf{f}(\mathbf{x}_0)$  is the  $q \times 1$  vector of regression functions of  $Y(\mathbf{x}_0)$ ,
- $\mathbf{F} = (f_j(\mathbf{x}_i))$  is the  $n \times p$  matrix of regression functions for the training data,
- $\mathbf{r}_0 = (R(\mathbf{x}_0 - \mathbf{x}_1 | \boldsymbol{\xi}), \dots, R(\mathbf{x}_0 - \mathbf{x}_n | \boldsymbol{\xi}))^\top$  is the  $n \times 1$  vector of correlation of  $\mathbf{Y}^n$  with  $Y(\mathbf{x}_0)$ ,
- $\mathbf{R} = (R(\mathbf{x}_i - \mathbf{x}_j | \boldsymbol{\xi}))$  is the  $n \times n$  matrix of correlation among the  $\mathbf{Y}^n$ .

The predictor is obtained by plugging estimated correlation parameters using the BLUP of  $y(\mathbf{x}_0)$ , which is

$$\widehat{\mathbf{Y}}(\mathbf{x}_0) = [\mathbf{f}_0^\top (\mathbf{F}^\top \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^\top \mathbf{R}^{-1} + \mathbf{r}_0^\top \mathbf{R}^{-1} (\mathbf{I}_n - \mathbf{F} (\mathbf{F}^\top \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^\top \mathbf{R}^{-1})] \mathbf{Y}^n. \quad (7)$$

## 2 Job files and inputs/outputs

### 2.1 Job files

The MPErK program is `perk.m`, which uses the inputs described in Section 2.2 and returns a output structure as explained in Section 2.3. Next we list the MATLAB programs used by `perk.m` and describe briefly their function.

- Programs used to estimate the correlation parameters
  1. `cormatexp.m` calculates the Gaussian or Power Exponential correlation function.
  2. `cubiccorrfn.m` calculates the cubic correlation function.
  3. `estgaussiancor.m` estimates the  $\theta_i$  parameters for the Gaussian correlation, (2).
  4. `estexponentcor.m` estimates the  $\theta_i$  and  $p_i$  parameters for the product power exponential correlation function, (3).
  5. `estcubiccor.m` estimates the  $\theta_i$  parameters for the cubic correlation function, (5).
  6. `calicatMLE.m` calls `llik1d.m` to calculate the likelihood.
  7. `calicatREML.m` calls `llik1d.m` to calculate the restricted likelihood.
  8. `llik1d.m` calculates the likelihood/the restricted likelihood.
- Programs used for Prediction and Cross Validation are `cormatexp.m`, `cubiccorrfn.m`, and `correxp.m`.

### 2.2 MPErK Inputs

We list the possible values and default settings of all the inputs next. Note that MPErK requires only 'X' and 'Y'. The other inputs are optional.

- 'X' =  $\mathbf{x}$ , an  $n \times p$  matrix of inputs for training input values (required)
  - $n$  == number of training data cases
  - $p$  == number of the computer code input for each experiment
- 'Y' =  $\mathbf{y}$ , an  $n \times 1$  vector of real-valued outputs of computer code at the training data sites (required)
- 'CorrelationEstimationMethod'  $\in$  {'MLE', 'REML'} where
  - 'MLE' indicates the Maximum Likelihood (ML) estimation of  $\xi$  is to be determined
  - 'REML' indicates the REstricted Maximum Likelihood (REML) estimation of  $\xi$  is to be determined
  - **Default:** 'CorrelationEstimationMethod' = 'REML'.
- 'CorrelationFamily'  $\in$  {'PowerExponential', 'Gaussian', 'Cubic'} where

- ‘PowerExponential’ is the product power exponential correlation function,
- ‘Gaussian’ is the Gaussian correlation function (the product power exponential family with powers equal to 2),
- ‘Cubic’ is the product cubic correlation function,
- **Default:** ‘CorrelationFamily’ = ‘PowerExponential’.
- ‘XPred’ =  $x_{pred}$ , an  $n_{pred} \times p$  matrix of  $n_{pred}$  new inputs at which predictions are to be computed
  - $n_{pred}$  is the number of the desired predictions,
  - **Default:** no predictions are requested.
- ‘RegressionModel’ =  $\mathbf{F}$ , an  $n \times q$  matrix of regression functions describing the mean of the fitted GaSP, i.e.,  $E(\mathbf{Y}^n) = \mathbf{F} \times \boldsymbol{\beta}$ , where
  - $q$  is the number of regression parameters,
  - **Default:** an  $n \times 1$  vector with elements 1.
- ‘PredRegressionModel’ =  $\mathbf{F}_{pred}$ , an  $n_{pred} \times q$  matrix of regression coefficients for prediction inputs ‘XPred’.
  - **Default:** an  $n_{pred} \times 1$  vector with elements 1.
- ‘CrossValidation’:  $\in \{‘Yes’, ‘No’\}$ 
  - ‘Yes’ indicates that the program does cross validation of the training data and saves the results in the matrix `output.cv`,
  - ‘No’ indicates that the program does NOT perform cross validation,
  - **Default:** ‘CrossValidation’ = ‘No’.
- ‘InitNumber’ =  $I$ , a positive integer number in  $\{1, 2, \dots, 200\}$  used to specify the number of the starting points in the search for the maximizer of the likelihood/restricted likelihood. The program generates  $200 \times p$  starting points by creating a Latin Hypercube design with  $200 \times p$  points, where  $p$  is the number (or dimension) of the correlation parameters. The program then computes the likelihood for each of the  $200 \times p$  choices and then picks the ‘InitNumber’  $\times p$   $\boldsymbol{\xi}$ s having the largest likelihood values at which to run the optimization routine `fmincon`. After running `fmincon`, the maximizer of the likelihood from is treated as the estimated correlation parameters used by the program.
  - **Default:**  $I = 10$ .
- ‘UseCorModel’ = `pout`, a structure containing model parameters to be used in this run. The program can directly predict the outputs and conduct cross validation.
  - **Default:** there is no input for ‘UseCorModel’.

## 2.3 MPErK outputs

The output of MPErK is a MATLAB structure with five objects. The command of the program is of the form

```
output = perk('X',x,'Y',...),
```

where `output` is a structure having five objects

1. `output.job`,
2. `output.corparms`,
3. `output.est`,
4. `output.pred`,
5. `output.cv`.

`output.job` is a structure that describes the data, the model, and the estimation method. The objects of `output.job` are

1. `output.job.CorrelationFamily` returns the type of correlation function (PowerExponential/Gaussian/Cubic).
2. `output.job.CorrelationEstimationMethod` returns the estimation method, which is 'MLE' / 'REML'.
3. `output.job.X` is an  $n \times p$  training data inputs.
4. `output.job.Y` is an  $n \times 1$  training data outputs.
5. `output.job.Ranges` is a  $p \times 2$  matrix having the ranges of the inputs. The  $i$ th row of `output.job.Ranges` consists of the minimum and maximum of the inputs in dimension  $i$ .
6. `output.job.XPred` is an  $n_{pred} \times p$  prediction inputs.
7. `output.job.RegressionModel` =  $\mathbf{F}$ .
8. `output.job.PredRegressionModel` =  $\mathbf{F}_{pred}$ .

`output.corparms` is a structure that contains the estimated correlation parameters.

1. `output.corparms.theta` =  $\boldsymbol{\theta}$  for the Gaussian/ Power Exponential/ Cubic correlation function.
2. `output.corparms.power` is  $(p_1, \dots, p_d)$  for the Power Exponential correlation; `output.corparms.power` is the vector  $(2, 2, \dots, 2)$  for the Gaussian correlation; `output.corparms.power` =  $\mathbf{Inf}$  for the Cubic correlation.

`output.est` is a structure that contains the following estimated quantities.

1. `output.est.loglik` is the maximized likelihood/restricted likelihood value (up to an additive constant  $-\frac{1}{2}\log 2\pi$ ).
2. `output.est.beta` is the  $q \times 1$  vector of the MLE or REML of the regression parameters  $\beta$ .
3. `output.est.invr` is an  $n \times n$  estimated inverse correlation matrix  $inv(\hat{\mathbf{R}})$  corresponding to the training data inputs and the estimated correlation parameters.
4. `output.est.sigma2` is the MLE or REML estimation of  $\sigma$ : `output.est.sigma2` =  $\frac{1}{n}(\mathbf{y}^n - \mathbf{F}\hat{\beta})^\top \mathbf{R}^{-1}(\mathbf{y}^n - \mathbf{F}\hat{\beta})$  for MLE; `output.est.sigma2` =  $\frac{1}{n-p}(\mathbf{y}^n - \mathbf{F}\hat{\beta})^\top \mathbf{R}^{-1}(\mathbf{y}^n - \mathbf{F}\hat{\beta})$  for REML.

`output.pred` contains the predictions and their estimated standard deviations. Note: `output.pred` does not exist unless 'XPred' is specified.

1. `output.pred.ypreds`: an  $n_{pred} \times 1$  vector containing the predictions.
2. `output.pred.se`: an  $n_{pred} \times 1$  vector containing the standard errors of the predictions.

`output.cv` is a structure that contains the leave-one-out cross validation predictions, standard errors, and residuals. Note: `output.cv` exists only if 'CrossValidation' = 'Yes'.

1. `output.cv.ypreds`: an  $n \times 1$  vector having the leave one out predictions.
2. `output.cv.se`: an  $n \times 1$  vector having the standard errors of the predictions.
3. `output.cv.resids`: an  $n \times 1$  vector having the cross validation residuals.

### 3 Examples

We describe two examples demonstrating the use of MPErK. The training data in the two examples are generated from the Branin function

$$y(x_1, x_2) = (x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6)^2 + 10(1 - \frac{1}{8\pi}) \cos(x_1) + 10,$$

for  $(x_1, x_2) \in [-5, 10] \times [0, 15]$  in  $\mathcal{R}^2$ . In Example 1, input variables to the Branin function have been scaled to  $[0, 1]^2$  prior to the use of MPErK.

#### 3.1 Example 1

- Goal: predict  $y(\cdot)$  for the Branin function at `xpred` based on the Power Exponential correlation, MLE, and the constant mean (i.e.,  $\mathbf{F} = \mathbf{1}_{21 \times 1}$ ); conduct cross validation.

- Data:

- Training data input with 21 input points obtained from a Maximin Latin Hypercube Design.

```
x=[ 0.83333333 0.40476193
    0.40476193 0.26190473
    0.97619047 0.54761907
    0.64285713 0.30952380
    0.50000000 0.97619047
    0.11904760 0.16666667
    0.54761907 0.02380953
    0.02380953 0.45238093
    0.07142860 0.83333333
    0.73809527 0.11904760
    0.88095240 0.73809527
    0.78571427 0.92857140
    0.30952380 0.07142860
    0.21428573 0.35714287
    0.35714287 0.69047620
    0.92857140 0.21428573
    0.16666667 0.64285713
    0.69047620 0.59523807
    0.59523807 0.78571427
    0.26190473 0.88095240
    0.45238093 0.50000000];
```

– Training data outputs Y:

```
y = [35.80951
      14.86287
      31.41880
      19.87899
      141.88566
      99.43335
      3.88973
      97.47380
      6.27060
      19.85914
      95.50587
      181.74214
      49.39445
      23.13762
      43.09524
      2.82392
      3.61474
      75.79100
      104.11175
      43.33586
      23.39797];
```

– Prediction requested at xpred:

```
xpred = [.03333  .03333
          .03333  .96667
          .50000  .50000
          .96667  .03333
          .96667  .96667];
```

- The MATLAB command:

```
output1 = perk('X',x,...
               'Y',y,...
               'CorrelationEstimationMethod','MLE',...
               'CorrelationFamily','PowerExponential',...
               'Xpred',xpred,...
               'CrossValidation','Yes');
```

- The output

- output structure:

```
output1 =  
  job: [1x1 struct]  
  corparms: [1x1 struct]  
  est: [1x1 struct]  
  pred: [1x1 struct]  
  cv: [1x1 struct]
```

- The original job specifications:

```
output1.job =  
  CorrelationFamily: 'PowerExponential'  
  CorrelationEstimationMethod: 'MLE'  
  X: [21x2 double]  
  Y: [21x1 double]  
  Ranges: [2x2 double]  
  XPred: [5x2 double]  
  RegressionModel: [21x1 double]  
  PredRegressionModel: [5x1 double]
```

- Estimated correlation parameters:

```
[output1.corparms.theta, output1.corparms.power]=  
  7.7521    2.0000  
  0.5028    2.0000
```

- Miscellaneous estimated quantities:

```
output1.est =  
  loglik: -65.0905  
  beta: 196.4929  
  inv_R: [21x21 double]  
  sigma2: 2.2482e+04
```

- Predictions and their standard errors:

```
[output1.pred.ypreds, output1.pred.se] =  
  206.7318    9.7680  
   7.4123    3.4646  
  24.4282    0.3152  
   5.1921    4.3077  
  135.2228   13.1852
```

- Leave-one-out cross validation predictions, their standard errors, and their residuals:

```
output1.cv =  
  preds: [21x1 double]  
  se:    [21x1 double]  
  resid: [21x1 double]  
  
[output1.cv.ypreds, output1.cv.se, output1.cv.resids] =  
  36.9259    3.6393   -1.1164  
  13.1337    2.0604    1.7292  
  28.1294    9.2858    3.2894  
  17.7324    2.3783    2.1466  
 139.2493    6.0951    2.6363  
  91.5428    7.5190    7.8906  
  16.3134    7.0031  -12.4237  
  96.6479   12.7942    0.8259  
  17.5079   11.3540  -11.2373  
  19.3533    6.8131    0.5059  
  97.2159    4.2875   -1.7100  
 175.3577    6.4358    6.3845  
  47.4345    5.5906    1.9599  
  27.6480    2.7981   -4.5104  
  41.7896    2.2456    1.3057  
   2.1535    9.1332    0.6704  
  -0.1582    3.4865    3.7730  
  76.4293    2.0260   -0.6383  
 104.5654    2.4147   -0.4536  
  45.8473    4.8841   -2.5114  
  24.4108    1.4741   -1.0128
```

## 3.2 Example 2

- Goal: predict  $y(\cdot)$  for the Branin function at `xpred` based on the Cubic correlation, REML, and the process mean  $E[Y(x_1, x_2)] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$ .

- Data

– Training data inputs

```
x = [ 7.5000    6.0714  
      1.0714    3.9286  
      9.6429    8.2143  
      4.6429    4.6429  
      2.5000   14.6429  
     -3.2143    2.5000  
      3.2143    0.3571  
     -4.6429    6.7857  
     -3.9286   12.5000  
      6.0714    1.7857  
      8.2143   11.0714  
      6.7857   13.9286  
     -0.3571    1.0714  
     -1.7857    5.3571  
      0.3571   10.3571  
      8.9286    3.2143  
     -2.5000    9.6429  
      5.3571    8.9286  
      3.9286   11.7857  
     -1.0714   13.2143  
      1.7857    7.5000];
```

– Training data outputs

```
y = [ 35.80951  
      14.86287  
      31.41880  
      19.87899  
      141.88566  
      99.43335  
      3.88973  
      97.47380  
      6.27060  
      19.85914  
      95.50587  
      181.74214  
      49.39445  
      23.13762  
      43.09524  
      2.82392  
      3.61474  
      75.79100  
      104.11175  
      43.33586  
      23.39797];
```

– Prediction requested at inputs

```
xpred = [-4.5000    0.5000  
         -4.5000   14.5001  
          2.5000    7.5000  
          9.5000    0.5000  
          9.5000   14.5001];
```

– Regression functions for the training data

```
F = [ones(21,1) x(:,1) x(:,2) diag(x(:,1)*(x(:,2)))'];
```

– Regression functions for the prediction inputs

```
Fpred = [ones(21,1) xpred(:,1) xpred(:,2) ...  
         diag(xpred(:,1)*(xpred(:,2)))'];
```

- The MATLAB command

```
output2 = perk('X',x,'Y',y,...  
              'CorrelationEstimationMethod','REML',...  
              'CorrelationFamily','Cubic','Xpred',xpred,...  
              'RegressionModel',F,'PredRegressionModel',...  
              Fpred);
```

- The output

- The original job specification

```
output2.job =  
  CorrelationFamily: 'Cubic'  
  CorrelationEstimationMethod: 'REML'  
    X: [21x2 double]  
    Y: [21x1 double]  
  Ranges: [2x2 double]  
    XPred: [5x2 double]  
  RegressionModel: [21x4 double]  
  PredRegressionModel: [5x4 double]  
    Xpred: [5x2 double]
```

- Estimated correlation parameters

```
output2.corparms =  
  scale: [2x1 double]  
  smoothness: Inf  
output2.corparms.scale =  
  18.5003  
  43.8506
```

- Miscellaneous estimated quantities

```
output2.est =  
  loglik: -57.3626  
  beta: [4x1 double]  
  inv_R: [21x21 double]  
  sigma2: 1.1360e+04  
output2.est.beta =  
  227.0622  
  -24.3519  
  -5.0811  
  2.0273
```

– Predictions and their standard errors

```
output2.pred =  
  preds: [5x1 double]  
  se: [5x1 double]  
[output2.pred.preds output2.pred.se] =  
 214.6024  14.3075  
   3.3216  10.8944  
  23.8426   3.7067  
 -19.0383  14.1916  
 153.1121  15.7334
```

## 4 Program Notes and Technical Details

This MPErK and an earlier version written in C (C-PERK) both provide predictions of the output from a computer code. We compare this MPErK and the C-PERK for Example 1 in Section 3.1. We list the estimates and predictions from both programs for Example 1.

- Likelihoods:

```
MPErK    -65.0905  
C-PERK   -65.0905
```

- MLEs of the correlation parameters:

MPErK			
Case	Range	Power	
1	7.7527	2.00000	
2	0.5028	2.00000	
C-PERK			
Case	Range	Power	2 - Power
1	7.75228	2.00000	0.00000
2	0.50277	2.00000	0.00000

- MLE of  $\beta$ :

```
MPErK    196.4929  
C-PERK   196.4891
```

- MLE of  $\sigma^2$ :

```
MPErK    22,474.0  
C-PERK   22,480.5
```

- Predictions and estimated standard errors:

MPeRK		
Case	EBLUP	Standard Error
1	206.7318	9.7680
2	7.4123	3.4646
3	24.4282	0.3152
4	5.1921	4.3077
5	135.2228	13.1852
C-PeRK		
Case	EBLUP	Standard Error
1	206.730725	9.7681924
2	7.412122	3.4646872
3	24.428226	0.3152540
4	5.191758	4.3078362
5	135.222015	13.1853379

Both C-PeRK and MPeRK provide nearly identical estimations and predictions in this example.

MPeRK uses `fmincon` function to estimate the unknown correlation parameters based on a user specified number of starting points. The option of `fmincon` is the *simplex* search with tolerance level 0.0001. The MATLAB command is

```
options = optimset('Display','off','LargeScale','off',...
                  'Simplex','on','TolFun',.0001);
```

The default settings are used for all other function options. Detailed explanations of the settings and all possible values of each option could be found in the MATLAB help file for functions `optimset` and `fmincon`.

`fmincon` uses lower and upper bounds for  $\theta_i$  for all  $i = 1, \dots, d$ . For the product power exponential and Gaussian correlation functions, the lower bound for  $\theta_i$  is  $-\log(0.99^{1/d})/M_i$  and the upper bound is  $-\log(0.1^{1/d})/m_i$ , where  $d$  is the number of inputs,  $M_i$  is the largest Euclidean  $L_2$  distance between two training data inputs in  $i$ th dimension, and  $m_i$  is the smallest non-zero  $L_2$  distance between two training data inputs in  $i$ th dimension. For the cubic correlation function, the lower bound for  $\theta_i$  is 0.00001 and the upper bound is  $100/d$ .

## Reference

T. J. Santner, B. J. Williams and W. I. Notz (2003). *The Design and Analysis of Computer Experiments*, Springer Verlag, New York.