Manual

space

Stochastic Process Analysis of Computer Experiments

(c) 1997-2001 Matthias Schonlau

matt@schonlau.net

Last Updated : March 26, 2001

Contents

Chapter 1

Introduction

space is a program for the analysis of data with Gaussian stochastic processes. It is most suitable for the analysis of computer experiments (deterministic functions), but it also allows for random error. Its major functions are :

- fitting the model (*Fit*),
- cross validating the model fit (CrossValidate)
- predicting new design sites using the fitted model (Predict),
- visualizing main and joint effects (MainEffects, JointEffects),
- Global minimization in several stages (MinimizeStage): The code suggests a specified number of design sites at each stage. The function can then be evaluated off - line at these design sites. The new function evaluations are fed back to space for the next stage.
- Minimization with supplied function of the function to be minimized. The code generates a single design site, waits for the design site to be evaluated by a supplied function, space generates the next design site, etc., until a convergence criterion is satisfied.

Several responses can be specified, it is possible to minimize one response variable subject to simple constraints on other response variables. Currently, rhe program does not handle

categorical variables, missing values or constraints on x -variables (however, constraints on x-variables might be handled by specifying a second response that represents the x-variable constraint).

The program is time consuming and therefore you may not want to use it with functions that are inexpensive to evaluate. The program is excessively time consuming if the data consists of more than a few hundred points.

The code runs both under various Unix platforms and under Windows.

Chapter 2 contains a brief tutorial that gives an example for most of the above major functions. Chapter 3 is a reference section with a description of the individual commands.

Appendix A contains a list of all commands as well as defaults to parameters.

Appendix B contains a list of warnings that *space* issues when encountering ill-conditioning of matrices. The normal user will not need to use this Appendix.

Chapter 2

Tutorial

We will use the Branin function to illustrate the use of "space" throughout this tutorial.

The Branin function (Törn and Žilinskas 1989) is

$$f(x_1, x_2) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10. \qquad (2.1)$$

The x ranges are $-5 \leq x_1 \leq 10$ and $0 \leq x_2 \leq 15$. The function has three global minima.

21 points were generated from a Latin Hypercube design. Their coordinates can be seen in Table 2.2. (Note : The Latin Hypercube design itself is not part of *space*). We analyze the data with the program shown in Table 2.1.

The program is executed with the command file as its first argument. For example, if the command file is called 'input.br', then it can be run typing

```
space input.br
```

This will cause the output log to appear on the terminal. The following command causes the output to be saved in the file 'output.br' :

```
space input.br output.br
```

space can also be used interactively. Type

```
space
```

and enter one command line at a time at the prompts. For help on what commands are available type 'help' or just 'h'.

After a comment line that is ignored by *space*, the next two program lines in Table 2.1 read the data into the matrices X and Y. The words or symbols in any command line may be separated by an arbitrary number of blanks. An example for the format for the matrices in the input files ('xstart.mat' and 'ystart.mat') is given in Table 2.2. Each observations appears on a new line, the number of blanks within a line is arbitrary. Further details are given in the reference section.

The command "Fit" causes all model parameters to be estimated. "Tries" specifies the number of attempts to estimate the model parameters. The best estimates (according to the maximum likelihood criterion) are kept and can subsequently be written out to several files (here 'beta21.mat', 'corpar21.mat' and ydescrip21.mat').

Next we are trying to assess the fit by means of cross validation. One point at a time is left out of the model and the response is predicted from the remainder. The output matrix (here 'cv.mat') contains cross validated response values, cross validated standard errors and cross validated expected improvements. The output is usually used to construct diagnostic plots.

The next command line causes predictions to be made at a set of points specified by an input file (here 'xpred.mat'). The input file follows the same format conventions as seen before in 2.2. An output file is produced (here 'ypred.mat') containing the predicted values and the corresponding standard errors.

Experiments are often done in several stages. We are now interested in designing one set of points (corresponding to one stage) aimed at finding the global minimum over a certain range. The range has to be specified (here in 'xdescrip.mat'). The file 'xdescrip.mat' can be seen in Table 2.3. In this example we choose to design a stage with 10 points.

# Branin test program	everything after '#' is a comment
X < xstart.mat	# read the X matrix from 'xstart.mat'
Y < ystart.mat	# read the Y matrix from 'ystart.mat'
Tries = 3	# number of attempts to fit the model
Fit	# fit the model (i.e. estimate all parameters)
	# Write out parameters :
RegressionModel > beta21.mat	# beta (Intercept of a Regression Model)
StochasticProcessModel > corpar21.mat	# alpha's and theta's
YDescription > ydescrip21.mat	# StochasticProcessVariance, n, etc.
CrossValidate > cv.mat	# CrossValidation: write cross validated response,
	# se's and expected improvements to 'cv.mat'
Predict < xpred.mat > ypred.mat	# Prediction at points given in 'xpred.mat'
	# write predicted values and se's to 'ypred.mat'
XDescription < xdescrip.mat	# read the range over which to minimize
MinimizeStage 10 > xdes.test	# design a stage with 10 points for minimization.
	# write X matrix including 10 points to 'xdes.test'
quit	# quit space

Table 2.1: Tutorial Program: Fitting, CrossValidation, Prediction and Minimization for the Branin Function

several lines of comments here ——————— Case x1 x2 ———————			some other lines of comments here ———- Case y ———-	
1	4.75000	12.75000	1	140.21015
2	-0.50000	10.50000	2	31.90971
3	2.50000	0.75000	3	6.62761
4	-4.25000	12.00000	4	15.31100
5	-1.25000	14.25000	5	49.73576
6	-5.00000	3.00000	6	214.00494
7	7.75000	0.00000	7	13.02623
8	3.25000	9.75000	8	57.57799
9	9.25000	4.50000	9	5.24649
10	-3.50000	9.00000	10	18.25489
11	6.25000	7.50000	11	60.56853
12	10.00000	8.25000	12	29.47461
13	4.00000	5.25000	13	16.32078
14	8.50000	11.25000	14	93.41968
15	5.50000	2.25000	15	18.00527
16	1.00000	6.75000	16	20.08260
17	-2.00000	1.50000	17	73.24144
18	-2.75000	6.00000	18	29.78705
19	7.00000	15.00000	19	207.97739
20	0.25000	3.75000	20	22.76390
21	1.75000	13.50000	21	106.09233

Table 2.2: Branin function : xstart.mat and ystart.mat

Case	Variable	Min	Max
1	x1	-5	10
2	x2	0	15

Table 2.3: Branin Function : xdescrip.mat

Chapter 3

Reference Manual

3.1 General Commands

#

Usage: # comment line

Comment sign. Everything after it is ignored.

Database

Usage: Database

Information about parameter values. Some parameters are only printed when different from the default values. Default values are given in Appendix A.

Help

Usage: Help

Generates the information in Appendix A.

Quit

Usage: Quit

Exit from space program.

3.2 In- and Output of Matrices

3.2.1 Syntax for Matrices

space reads in and outputs matrices. A file that contains a matrix must look as follows:

```
Comments
and more
comments
------------------------------
 Case          x1          x2
------------------------------
    1    -0.20000     0.20000
    2    -1.80000    -1.40000
    3     1.00000    -2.00000
```

Everything before the first line that starts with '–' is considered a comment and ignored by the program. A line with labels ('Case' may be the first label) follows. Another line that starts with '–' follows the line with labels. An arbitrary number of data lines follows. Extra blank lines are allowed.

The Case column may be omitted. The numbering of the Cases has no influence on the program.

3.2.2 Output on terminal screen

All matrices in this section can be read in and written out. The matrix is written out to the filename specified, unless the filename specified is 'screen' or 's'. In that case, the matrix is written onto the terminal screen. This is mainly useful for interactive use.

3.2.3 Note for Windows users

Windows interprets the file type '.mat' as a short cut to an access database. For some reason the effect is that files with this file type cannot be opened. It may be easiest to name the matrices differently, such as '.mt' instead of 'mat'.

3.2.4 Specific Matrices

X

Usage: X < {filename}

Usage: X > {filename}

The X matrix has to be read before any of the other matrices are read.

y

Usage: y < {filename}

Usage: y > {filename}

The number of rows in the y matrix has to match the number of rows in the x-matrix.

StochasticProcessModel

Usage: StochasticProcessModel < {filename}

Usage: StochasticProcessModel > {filename}

Specification of the stochastic process model (alpha's and theta's). Example:

```
----------------------------------------
 Case    Variable     theta.y     alpha.y
----------------------------------------
    1          x1    0.024708    0.000000
    2          x2    0.000345    0.000000
```

RegressionModel

Usage: RegressionModel < {filename}

Usage: RegressionModel > {filename}

Specification of the regression model. Currently, the regression model may only contain an intercept ("Term 1" in example below). Example :

```
-----------------------------------------
 Case    Variable        Term  Coefficients
-----------------------------------------
    1           y           1    1318.535335
```

XDescription

Usage: XDescription < {filename}

Usage: XDescription > {filename}

Information about x variables. Currently only "Min" and "Max" available. "Min" and "Max" specify the Minimization range. Example :

```
--------------------------------
 Case Variable Min        Max
--------------------------------
  1     x1      -5     10
  2     x2      0       15
```

YDescription

Usage: YDescription < {filename}

Usage: YDescription > {filename}

Information about y variables. See example below for fields. There is no particular order to the fields nor do all fields have to appear all the time. For example, when reading in usually only "Variable", "StochasticProcessVariance" (and sometimes "ErrorVariance") are used.

The field "Constraint" indicates whether the corresponding response variable is constrained. The actual values of the constraint can be specified in the fields "Min" and /or "Max". Example :

```
-----------------------------------------------------------
```

```
 Case    Variable  LogLikelihood  StochasticProcessVariance
-------------------------------------------------------------
    1           y     -67.331803                 42154.457537
    2           y2     33.0     15.2
-------------------------------------------------------
 Case  ErrorVariance   NumberObs Constraint Min Max
-------------------------------------------------------
    1       0.000000          21  FALSE      NA  NA
    2       0.000000          21  TRUE       -3  4
```

3.3 Functions

Fit

Purpose: Maximum Likelihood estimation of the Parameters.

Usage: Fit

Required: X and Y

Optional Input: RegressionModel, CorrelationParameter, YDescription

Optional Commands: Tries, RandomSeed, RandomError

Effect: Changes RegressionModel, CorrelationParameter, YDescription

Maximum Likelihood estimation is done (Tries) times from different starting points. RandomSeed is the seed value to generate random numbers (the selection of starting points has a random component). RandomError specifies whether the problem is deterministic or not (TRUE by default). When optional Matrices are specified, the starting point for the first MLE is taken from the RegressionModel, CorrelationParameter and (for the stochastic Process variance) from YDescription.

CrossValidate

Purpose: leave-one-out cross validations of the data given by X and Y.

Usage: CrossValidate > outputfile

Required: Model has to have been fitted before

Effect: Outputs CrossValidations into outputfile

Outputfile is usually used for constructing diagnostic plots.

Predict

Purpose: Prediction at a given set of sites

Usage: Predict < inputfile > outputfile

Required: Model has to have been fitted before

Effect: Outputs Predictions into outputfile

Input File contains the prediction sites.

MainEffects

Purpose: Outputting data for the visualization of Main effects

Usage: MainEffects > outputfile

Required: Model has to have been fitted before

Optional Commands: GridSize

Effect: Outputs (Gridsize) levels for each variable into outputfile

After the generation of the output file the effects have to be plotted.

JointEffects

Purpose: Outputting data for the visualization of Joint effects

Usage: JointEffects > outputfile

Required: Model has to have been fitted before

Optional Commands: GridSize

Effect: Outputs (Gridsize $\times$ Gridsize) levels for each pair of variables into outputfile

After the generation of the output file the effects have to be plotted.

Minimize

Purpose: Find the global Minimum of a function with few function evaluations

Usage: Minimize

Required: one of (ObjectiveFunction, SystemCommandObjectiveFunction), Xdescription (Min and Max), YDescription (only if constrained minimization is desired)
Optional Commands: StoppingCriterion, StoppingAbsoluteTolerance, StoppingRelativeTolerance, GridSize, MinimumSample, MinimizationTarget, MleMaxObs, MleFrequency, MlePercentage, GlobalExpectedImprovement
Effect: X, Y, RegressionModel, CorrelationParameter, YDescription are updated
Bugs: The variable to be minimized must be the first variable in Y

The minimization range must be specified in Xdescription. If minimization subject to constraints on additional responses is desired, then the constraints must be specified in YDescription.

For the Stopping criterion, StoppingAbsoluteTolerance and StoppingRelativeTolerance can be specified. Only one of MleFrequency and MlePercentage can be used.

In order for the minimization to proceed, there need to be a way of letting *space* know how to compute the y's (responses). That is done either with the command ObjectiveFunction or with SystemCommandObjectiveFunction.

Sampling one point at a time can be very time consuming - especially if the parameters are reestimated after every point. There are a number of options that avoid reestimation after every point to make it less time consuming: MleMaxObs, MleFrequency / MlePercentage.

MinimizeStage

Purpose: Perform one stage of a function minimization in stages
Usage: MinimizeStage {number of points} > outputfile
Required: Model has to be fitted before , Xdescription (Min and Max)
Optional Commands: GlobalExpectedImprovement MinimizationTarget
Effect: Both the old design (X) and the new design are output in outputfile
Bugs: The variable to be minimized must be the first variable in Y

The range in Xdescription must be specified. The number of points to be designed in this stage is given by {number of points} .

3.4 Options

GlobalExpectedImprovement

Usage: GlobalExpectedImprovement = {non-negative integer number }

The greater GlobalExpectedImprovement, the more global the search for minimization. The choice GlobalExpectedImprovement = 0 yields the probability of improvement, the choice of GlobalExpectedImprovement = 1 (the default) yields expected improvement. Currently up to GlobalExpectedImprovement = 10 is supported. For higher values of GlobalExpectedImprovement the tolerance stopping criterion does not work very well in that the stopping rule will not kick in for a long time. The choice of GlobalExpectedImprovement also impacts StoppingAbsoluteTolerance and StoppingRelativeTolerance.

GridSize

Usage: GridSize = {positive integer number }

GridSize is used to determine the number of levels of each x variable for MainEffects and JointEffects. It is not possible to specify a different number of levels for each x variable.

MinimumSample

Usage: MinimumSample = {number}

During minimization with Minimize the minimization will continue sampling (overwriting any stopping rule) until at least a total of MinimumSample observations (including initial points) have been sampled.

MinimizationTarget

Usage: MinimizationTarget = {number}

By default the algorithm samples where the (global) expected improvement over the best value found so far is maximized. MinimizationTarget changes that value. That is, the algorithm samples where the expected improvement over MinimizationTarget is maximized.

By default MinimizationTarget is $min(y_1, ..., y_n)$. MinimizationTarget must be set to a value smaller than $min(y_1, ..., y_n)$.

MinimizationTarget also affects StoppingRelativeTolerance.

MleFrequency

Usage: MleFrequency = {number}

During minimization with Minimize the mle is updated every MleFrequency observations. At most one of MleFrequency and MlePercentage can be used.

MleMaxObs

Usage: MleMaxObs = {number}

During minimization with Minimize the mle is not updated when there are more than MleMaxObs observations. This is sometimes needed when the mle would take too much time and is unlikely to change much anyways.

MlePercentage

Usage: MlePercentage = {number}

During minimization with Minimize the mle is updated every time the number of observations since the last mle has increased by at least MlePercentage per cent. At most one of MleFrequency and MlePercentage can be used.

ObjectiveFunction

Usage: ObjectiveFunction = {function name }

Specifies the function to be minimized for the command Minimize. The function must read a file produced by space called space.temp which contains information about the number of x- variables and the actual values of the x's to be evaluated. For example, for three x-variables where $x_1 = .3$, $x_2 = 12.1$ and $x_3 = -.9$ space.temp takes the following form :

```
3  .3 12.1 -.9
```

Numbers are separated by at least one space. The objective function must compute the response(s) for these x's and output it back into space.temp. For example, if there are two responses which yield 2.4 and 25.2 for the above x-values, then space.temp should look like

```
3  .3 12.1 -.9 2.4 25.2
```

The order of the x- and y- variables must be the same as input into X and Y. Only one of ObjectiveFunction and SystemCommandObjectiveFunction can be used.

ObjectiveFunction is an executable taking space.temp as an argument:

'functionname' space.temp .

RandomError

Usage: RandomSeed = {TRUE / FALSE }

Specifies whether the problem is deterministic (i.e. no random error) or not.

RandomSeed

Usage: RandomSeed = {number}

Specifies a seed for the random number generator. The seed should be an integer number between 1 and 100.

StoppingAbsoluteTolerance

Usage: StoppingAbsoluteTolerance = {number}

Specifies a stopping rule for the minimization. Minimization will stop when

$$\text{Criterion} < \text{StoppingAbsoluteTolerance}$$

where the criterion is determined through the choice of the GlobalParameter.

The criterion is

Probability of Improvement	$P(I)$	if GlobalParameter $= 0$
Expected Improvement	$E(I)$	if GlobalParameter $= 1$
Generalized Expected Improvement	$(E(I^p))^{1/p}$	if GlobalParameter > 1

where for brevity we here p for GlobalParameter is used.

The minimization will stop when the StoppingAbsoluteTolerance condition or the StoppingRelativeTolerance is fulfilled.

StoppingRelativeTolerance

Usage: StoppingRelativeTolerance = {number}

Specifies a stopping rule for the minimization. Minimization will stop when

$$\frac{\text{Criterion}}{\text{MinimizationTarget}} < \text{StoppingAbsoluteTolerance.}$$

The criterion is determined through the choice of the GlobalParameter (see also StoppingAbsoluteTolerance). By default, the MinimizationTarget is the minimal found function value, $min(y_1, y_2, \ldots, y_n)$.

The minimization will stop when the StoppingAbsoluteTolerance condition or the StoppingRelativeTolerance is fulfilled.

SystemCommandObjectiveFunction

Usage: SystemCommandObjectiveFunction = { command line }

Specifies the function to be minimized for the command Minimize in the same way ObjectiveFunction does, except that SystemCommandObjectiveFunction is not an executable. The system function call simply "command line".

Tries

Usage: Tries = { integer number }

Number of attempts to perform maximum likelihood estimation (MLE) of the parameters. This commands affects Fit and Minimize.

Appendix A

List of all Commands

List of valid commands:

```
IN/OUTPUT:
(X|y|RegressionModel|StochasticProcessModel|range) (<|>) ([filename]|screen)
(R|F|YDescription|XDescription)                    (<|>)  ([filename]|screen)

FUNCTIONS:
Fit
CrossValidate                  {< xfilename yfilename} > ([filename]|screen)
Predict                         < {[filename]|screen} > {[filename]|screen}
Minimize
MinimizeStage  {number}         > {[filename]|screen}
MainEffects                     > {[filename]|screen}
JointEffects                    > {[filename]|screen}

OPTIONS:
GridSize                        = {number} #default is 10
GlobalExpectedImprovement       = {number} #default is 1
MinimumSample                   = {number} #default is 0
```

```
MleFrequency                    = {number} #default is 1
MleMaxObs                       = {number} #default is 200
MlePercentage                   = {number} #absent by default
MinimizationTarget              = {number} #equals min(y_1 ... y_n) by default
ObjectiveFunction               = {name of executable}
RandomError                     = {TRUE/FALSE}
RandomSeed                      = {number} #default is 10
SystemCommandObjectiveFunction  = {system command for function}
StoppingCriterion               = {quantile/tolerance} #default is tolerance
StoppingAbsoluteTolerance       = {number} #default is 0.001
StoppingRelativeTolerance       = {number} #default is 0.001
StoppingQuantile                = {number} #default is 0.01
Tries                           = {number} #default is 3
CriterionOptimStart             = {random|displacement|distanceneighbour}
                                  #default is distanceneighbour
CriterionOptimMethod            = {simplex|powell} #default is simplex
CriterionOptimTries             = {number} #default is 2*n

OTHER:
check_allocation (as first command only)
#{comment}
Info
Database
Quit
```

Appendix B

Warnings

*Type T*1

Correlation Matrix is singular (1msg per singularity) (in $calc.augmR$)

*Type T*2

Correlation Matrix is singular (1msg per singularity) (decompose in krigfit)

*Type T*3

Correlation Matrix is singular (1msg per singularity) (in $Kr.calc.augmR$)

*Type T*4

Numerical difficulties in computing the log likelihood. $(Krig - mle)$

*Type T*5

warning only

Bibliography

[1] Schonlau, M. (1997). *Computer Experiments and Global Optimization*. Ph.D. thesis, University of Waterloo.

[2] Jones, D., Schonlau, M., Welch, W., (1998) "Efficient Blobal Optimization of Expensive Black-Box Functions". *Journal of Global Optimization* 00:1-39 (to appear)

[3] Schonlau, M., Welch, W., Jones, D., "Global Versus Local Search in Constrained Optimization of Computer Models", in: *New Developments and Applications in Experimental Design*, N. Flournoy, W.F. Rosenberger, and W.K. Wong (editors), Institute of Mathematical Statistics. (to appear)

[4] Schonlau, M., Welch, W., Jones, D. (1997), "A Data-Analytic Approach to Bayesian Global Optimization". *Proceedings of the Section on Physical and Engineering Sciences*, American Statistical Association, 186-191.

[5] Schonlau, M., Welch, W. (1996). "Global Optimization with Nonparametric Function Fitting". *Proceedings of the Section on Physical and Engineering Sciences*, American Statistical Association, 183-186.

[6] Schonlau, M., Hamada, M., Welch, W. (1995). "Nonparametric Function-Fitting To Suggest Nonlinear Parametric Models". *Proceedings of the Section on Physical and Engineering Sciences*, American Statistical Association, 262-267