PSUADE Short Manual
(Version 1.4d)

Charles Tong
Center for Applied Scientific Computing
Lawrence Livermore National Laboratory
Livermore, CA 94551-0808 [1]
April, 2012

[1]This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract No. DE-AC52-07NA27344.

1 Introduction

PSUADE (Problem Solving environment for Uncertainty Analysis and Design Exploration) is a software package for various uncertainty quantification (UQ) activities such as uncertainty assessment (UA), sensitivity analysis (SA), parameter study, numerical optimization, etc. It consists of three major components: a suite of sampling methods, a job execution environment, and a collection of analysis tools. This document describes how to set up and use these UQ tools. The mathematics of the sampling and analysis methods will be presented in the theory manual.

1.1 A Quick Start

Follow the instruction in this section and you should be able to build and run PSUADE (on a simple example) in less than 5 minutes (depending on the speed of your hardware) on a Linux-based system.

1. untar the PSUADE_v1.4d.tar
2. cd to PSUADE_v1.4d
3. Execute the install_psuade script
4. Use make to create the 'psuade' executable in the 'bin' directory
5. cd to the Examples/Bungee directory
6. Compile the example: cc -o simulator simulator.c -lm
7. Run PSUADE: ../../bin/psuade psuade.in

What you have just done are to build the PSUADE executable and perform uncertainty analysis on a simple example. At the end PSUADE prints out the summary of statistics and all input and output data have been stored in the psuadeData file. Later on in this document more details about how to create PSUADE input files (psuade.in in this case) will be given.

1.2 PSUADE Capabilities

PSUADE supports non-intrusive uncertainty quantification through sampling (it does have a few features to support semi-intrusive methods). Some of the available sampling methods are:

- Monte Carlo and quasi-Monte Carlo
- Latin hypercube and orthogonal arrays
- Morris one-at-a-time, its variants, and other screening designs

- Central composite, factorial and fractional factorial
- Fourier Amplitude Sampling Test (FAST)
- Other space-filling designs
- Support several popular input probability density functions

These sample points are then evaluated by running the user simulation codes. PSUADE provides a mechanism to accomplish this via a runtime environment which performs the following tasks when invoked:

- Write the values of a sample points to a file (say, param.in)
- Call the user code (provided by users in the PSUADE input file) with param.in
- The user code is expected to read in the sample data, run the application, produce some output quantities and write them to an output file (say, param.out). Thus, the user code can be a simple program such as simulator.c in some of our examples or a complex super-script performing postprocessing, actual model evaluation and postprocessing.
- PSUADE detects the presence of param.out and reads in the outputs.
- PSUADE moves on to the next sample point and continues until all sample points have been processed (PSUADE can process multiple sample points at the same time using the asynchronous mode).
- Finally, PSUADE reads in all sample data and analyzes them based on the user requests given in the PSUADE input file.

PSUADE supports many types of analysis such as

- Response surface analysis (MARS, polynomial regression, Gaussian process, etc.)
- Parameter screening
- Hypothesis testing
- Correlation analysis
- Main effect (first order sensitivity) analysis with or without input inequality constraints
- Interaction (second order sensitivity) analysis with or without input inequality constraints
- Group and total sensitivity analysis
- Numerical optimization

There are other advanced features in PSUADE which are under active research and are not described in this document.

2 Installation

As described in the last section, installation of PSUADE on Linux-based systems is straightforward. Simply type the following at the command prompt:

```
[Linux] install_psuade
```

where you will be asked to select the platform. PSUADE supports several platforms other than Linux-based platforms, but they have not been thoroughly tested. Afterward, the PSUADE executable can be built using:

```
[Linux] make
```

and the PSUADE executable will be created in the bin directory.

3 Using PSUADE

PSUADE operates in one of the two modes: batch and command line modes.

3.1 Batch Mode

In batch mode, PSUADE interacts with users via a few files. At the first level, an PSUADE input file (called psuade.in here) has to be created and run via

```
[Linux] psuade psuade.in
```

This psuade.in file should begin with the keyword PSUADE as the first line and should have 5 subsections following by the last line having the keyword END. The formats of the subsections are described next (for an example, read the psuade.in file in the Examples/Bungee directory).

3.1.1 The Input Section

The INPUT section allows the users to specify the number of inputs, their names, their range, and their distributions. Specifically, it is enclosed in an **INPUT** block. An example is given as follows:

```
INPUT
   dimension = 4
   variable 1 X1  =   0.0   1.0
   variable 2 X2  =   0.0   1.0
   variable 3 X3  =   0.0   1.0
   variable 4 X4  =   0.0   1.0
   variable 5 X5  =   0.0   1.0
   PDF 1 T   0.0   1.0
```

```
   PDF 2 N   0.0   1.0
   PDF 3 L   0.0   1.0
   PDF 4 W   0.1   1.0
   PDF 5 G   0.1   1.0
END
```

In this example the number of inputs is 4, their names are X1, X2, X3, and X4 (notice that the variable indices are 1-based), and their lower and upper bounds are all 0 and 1, respectively. The probability density distribution (PDF) for the inputs are optional (the default is uniform U). If the PDF is either normal (N) or lognormal (L), the mean and standard deviation also have to be provided. If the PDF is triangular (T), the mean and width are to be provided (joint PDF for normal and lognormal distributions are also supported). Other distributions available in Version 1.4d on are Gamma (G), Beta (B), Exponential (E), and Weibull (W) distributions.

A word of caution: When distributions other than uniform is specified, the lower and upper bounds after each variable declaration will be used to bound the sample inputs. Hence, it is important to specify the lower and upper bounds appropriately so that the prescribed distributions have nonzero densities in the regions. Another note is that these distributions will be applied to sample generation when the PSUADE_IO section does not exist in the PSUADE file (that is, it is not a data file). Otherwise, the distributions will only be used in subsequent statistical analyses.

3.1.2 The Output Section

The OUTPUT section is similar to but simpler than the INPUT section. Here only the output dimension and the names of the output variables are needed. For example, given as follows:

```
OUTPUT
   dimension = 3
   variable 1 Y1
   variable 2 Y2
   variable 3 Y3
END
```

3.1.3 The Method Section

The METHOD section specifies the selected sampling method and additional information on sampling. An example is given below.

```
METHOD
   sampling = LH
   num_samples = 600
   num_replications = 60
   num_refinements = 0
```

```
   randomize
END
```

In this example, the sampling method is Latin hypercube, the sample size has been set to 600, and no refinement is used (refinement is an advanced feature for adaptive sampling and is described in detail in the theory manual). When the number of replications is larger than 1, it is called replicated Latin hypercube which is useful for global sensitivity analysis. In this example, with 60 replications, the number of levels for the Latin hypercube samples is $600/60 = 10$. Also, the randomize flag has been turned on to tell the sampling method that random perturbation should be added to the sample.

Some of the other sampling methods available are (refer to the theory manual for more details):

```
MC    - Monte Carlo
LPTAU - a quasi-random sequence
FACT  - full factorial design
MOAT  - Morris one-at-a-time screening
LH    - Latin hypercube
OA    - Orthogonal Array
OALH  - Orthogonal Array-based Latin hypercube
FAST  - Fourier Amplitude Sampling Test (FAST)
BBD   - Box Behnken design
PBD   - Plackett Burman design
FF4   - Fractional factorial of resolution IV
FF5   - Fractional factorial of resolution V
FF5   - Fractional factorial of resolution V
CCI4  - Central composite (circumscribed) of resolution V
CCI5  - Central composite (circumscribed) of resolution V
SPARSEGRID
```

3.1.4 The Application Section

The APPLICATION section sets up the user-provided simulation executable and other runtime parameters. An example is given below.

```
APPLICATION
   driver = ./testmain
   opt_driver = NONE
   max_parallel_jobs = 1
   max_job_wait_time = 1000000
END
```

Here driver points to the executable to be used for function evaluations. opt_driver points to the executable for numerical optimization. Again, the user code can just be a simple

program or a complex super-script performing postprocessing, actual model evaluation and postprocessing. The user code can also be a PSUADE data file itself, as will be shown later.

After the creation of a sample based on information from the INPUT, OUTPUT and METHOD sections, PSUADE proceeds with launching the jobs. If the max_parallel_jobs is set to 1, the sequential mode is turned on. In this mode, PSUADE schedules the evaluation of the user-provided function by sequencing from sample point 1 onward. To run job i, PSUADE first creates an input parameter file (called psuadeApps.in.i). This file contains in its first line the input dimension, followed by the values of the input parameters for the i-th sample point. PSUADE then calls driver with two parameters (for example, for the sample point 9)

```
./testmain psuadeApps.in.9 psuadeApps.out.9
```

The driver program is expected to take the input parameters from the psuadeApps.in.9 file, do whatever is needed, and write the outputs to the psuadeApps.out.9 file. An example of the content of an output file (3 output variables) is:

```
3.12
15.9
100.4
```

If max_parallel_jobs is set to a number larger than 1, then the asynchronous job scheduling mode is turned on. In this mode, multiple psuadeApps.in.i files are created simultaneously, and driver is called max_parallel_jobs times simultaneously. max_job_wait_time is used for fault detection and recovery.

Some of the other options in this section are:

```
launch_only         - launch all jobs without waiting for results
limited_launch_only - same as launch_only but only launch max_parallel_jobs job
gen_inputfile_only  - generate all sample files only (no code runs)
```

3.1.5 The Analysis Section

The ANALYSIS section specifies what type of analysis is desired and the parameters used in the analysis. An example is given below (which computes the different moments of the sample on output number 1).

```
ANALYSIS
   analyzer method = Moment
   analyzer threshold = 5.000000e-04
   analyzer outputID  = 1
   analyzer rstype = MARS
   #optimization method = bobyqa
   #optimization num_local_minima = 1
   #optimization use_response_surface
```

```
   #optimization num_fmin = 1
   #optimization fmin = 0
END
```

Other available analysis methods are:

```
MainEffect     - sensitivity indices
TwoParamEffect - second order sensitivity indices
RSFA           - response surface analysis (curve fitting)
MOAT           - Morris one-at-a-time screening analysis
Correlation    - correlation analysis
Integration    - numerical integration using the data points
FAST           - Fourier Amplitude Sampling Test analysis
FF             - fractional factorial main and interaction analyses
PCA            - principal component analysis
RSMSobol1      - response surface-based first order sensitivity analysis
RSMSobol2      - response surface-based first second sensitivity analysis
RSMSobolG      - response surface-based group main effect analysis
RSMSobolTSI    - response surface-based total sensitivity analysis
```

When response surfaces are used together with the selected analysis method, the response surface type has to be specified. The available response surface types are:

```
MARS            - multi-variate adaptive regression splines (by Friedman)
MarsBag         - bootstrapped MARS
linear          - linear regression
quadratic       - second order polynomial
cubic           - third order polynomial
quartic         - fourth order polynomial
Legendre        - Legendre polynomial of different orders
user_regression - user-specified polynomial
ANN             - artificial neural network (SNNS)
GP1             - Gaussian process (Tpros)
GP2             - Gaussian process (by Carl Rasmussen)
SVM             - support vector machine
```

In addition, for performing numerical optimization, a few related options have to be specified (the commented lines from the above example). In the example, the selected optimization method is bobyqa by Michael Powell. Other available optimization methods are crude (a simple examination of all sample outputs and select the minimum one), minpack (an external optimization package), cobyla (an external optimization package), and sm/mm (space-mapping and manifold-mapping methods by David E. Ciaurri). num_local_minima tells PSUADE how many local minimum to search. If user_response_surface is used, the sample data will first be used to create a response surface before searching for the optimal points.

num_fmin tells PSUADE the number of optimal points to be expected so that PSUADE can decide to stop when this has been achieved. Also, users can also tell PSUADE what the optimal value is via fmin.

3.2 Command Line Mode

PSUADE allows users to interactively perform some of the analyses. Command line mode is activated by just calling

```
[Linux] psuade
```

without any argument. Some of the available commands in the command line mode are (most of the commands can be found by the help command):

```
load <filename>  (load a data file, e.g. psuadeData)
splot            (generate scatter plot in matlab)
me               (main effect study + matlab plot)
rs               (2-input response surface in Matlab)
rs3              (3-input response surface in Matlab)
rs3m             (3-input response surface in Matlab (movie)
rsi              (2-input response surface intersections in Matlab)
rsi3             (3-input response surface intersections in Matlab)
rscheck          (Check quality of response surface with MARS)
quit
help
```

For example, after you have completed a set of runs, a PSUADE data file will be created (say, the file name is psData). To create scatter plots for the data in the command line mode, do:

```
[Linux] psuade
*** ******************************************************** ***
*** Welcome to PSUADE (version 1.4d)                        ***
*** ******************************************************** ***
PSUADE - A Problem Solving environment for
         Uncertainty Analysis and Design Exploration
psuade> load psData
psuade> splot
matlabsp.m is now available for scatter plots.
psuade> quit
[Linux]
```

You can now use Matlab to display the scatter plot.

4 Examples

PSUADE provides many tools for answering many questions with uncertainty quantification. For example, given a computational model simulating some physical processes,

1. How to assess the impact of parameter uncertainties on the variabilities of some output of interest? (uncertainty analysis)
2. How do available experimental data alter the outcome of an uncertainty analysis? (conditional analysis)
3. How to identify a small subset of parameters accounting for most of the output variabilities ? (parameter screening)
4. How to quantify the impact of a particular subset of parameters on the output uncertainties ? (global sensitivity analysis)
5. How to construct a relationship between some input parameters to the model and the output of interest? (response surface modeling)
6. How to find the parameter values that best fit the available experimental data ? (calibration, parameter estimation)
7. How to search for the parameter values that give the best model performance? (optimization)
8. How to process, manipulate and visualize the uncertainty data?
9. How to formulate and perform hypothesis testing?

In the following we provide a few examples to show in more details how to set up and run PSUADE. PSUADE has many other advanced features for handling complex multi-physics models.

4.1 Confidence Intervals

Let the function be given by:

$$Y = F(X_1, X_2) = X_1 + X_2 + X_1X_2 + X_1^2, \quad X_i \in [0, 1]$$

and we would like to compute the few moments of this function assuming the inputs are uniformly distributed. We select Latin hypercube sampling with a sample size of 1000. The jobs are to be run in sequential mode. The corresponding PSUADE input file psuade.in is:

```
PSUADE
INPUT
   dimension = 2
   variable 1 X1  =   0.0   1.0
   variable 2 X2  =   0.0   1.0
END
OUTPUT
   dimension = 1
   variable 1 Y1
END
METHOD
   sampling = LH
   num_samples = 1000
   randomize
END
APPLICATION
   driver = ./testmain.py
END
ANALYSIS
   analyzer method = Moment
   diagnostics 3
END
END
```

The driver program can be in any language provided that it is executable. Our example uses Python to simulate the above function (testmain.py:

```
#!/usr/local/bin/python
import string
import sys
infile  = open(sys.argv[1], "r")
lineIn  = infile.readline()
ncols   = string.split(lineIn)
n       = eval(ncols[0])
lineIn  = infile.readline()
ncols   = string.split(lineIn)
X1      = eval(ncols[0])
lineIn  = infile.readline()
ncols   = string.split(lineIn)
X2      = eval(ncols[0])
infile.close()
Y       = X1 + X2 + X1 * X2 + X1 * X1
outfile = open(sys.argv[2], "w")
```

```
outfile.write("%e \n" % Y)
outfile.close()
```

After these files have been prepared (These files can be found in the Examples/UserExample directory. Make sure the python link in the testmain.py file is correct, and that testmain.py has execute permission), run PSUADE with:

```
[Linux] psuade psuade.in
```

and, at the completion of the runs, the moment information will be displayed and the psuadeData file will also be created for further analysis.

4.2 Basic Regression Analysis

Suppose we use the above function again and we would like to perform a quadratic regression analysis. We can reuse the data in the psuadeData file from the last exercise (let's copy this file to psData so that it will not be overwritten by PSUADE). To do so, the ANALYSIS section in psData has to be modified to:

```
PSUADE
... data ...
INPUT
...
END
OUTPUT
...
END
METHOD
...
END
APPLICATION
...
END
ANALYSIS
   analyzer method = RSFA
   analyzer rstype = quadratic
END
END
```

This modified file is to be run with PSUADE via:

```
[Linux] psuade psData
```

and regression analysis results (the regression coefficients and R^2) will be displayed.

4.3 Screening for Important Inputs

A useful design and analysis tool in PSUADE is parameter screening using the Morris method, which is an effective variable selection method when the number of inputs is large (say, > 15). The corresponding PSUADE input file should be set up as (say, for a 20-dimension problem in the Examples/MOATTest directory):

```
PSUADE
INPUT
   dimension = 20
   variable  1 X1  =   0.0   1.0
   variable  2 X2  =   0.0   1.0
   ...
   variable 20 X20 =   0.0   1.0
END
OUTPUT
   dimension = 1
   variable 1 Y1
END
METHOD
   sampling = MOAT
   num_samples = 210
   randomize
END
APPLICATION
   driver = ./morris_simulator
END
ANALYSIS
   analyzer method = MOAT
   diagnostics 3
END
END
```

Here the sample size should be a multiple (usually 10) of $K + 1$ where K is the number of inputs. The driver program can be constructed in a similar manner as before (and thus is not to be given here). Again, PSUADE is launched with this input file and screening results will be displayed at completion.

Alternatively, the analysis can be performed interactively by (again psuadeData has been created and has been copied to the psData file):

```
[Linux] psuade
*** ********************************************************* ***
*** Welcome to PSUADE (version 1.4d)                          ***
*** ********************************************************* ***
```

```
PSUADE - A Problem Solving environment for
         Uncertainty Analysis and Design Exploration
psuade> load psData
load complete : nSamples = 210
                nInputs  = 20
                nOutputs = 1
psuade> moat
... (MOAT results) ...
...
Would you like to create screening diagram ? (y or n) y
Please enter your preferred screening diagram file name : screen.m
MOAT screening diagram matlab file = screen.m
psuade> quit
[Linux]
```

Thereafter, you can launch Matlab and run screen to view the Morris screening diagram (scatter and bootstrap plots can also be generated).

In summary, the selection of screening methods depends on many factors such as parameter sizes and knowledge about the function. We list here a few options and their assumptions:

PBD design - useful if function is linear in the parameters and there is no parameter interactions.

FF design - useful if function is linear in the parameters and there are only two-parameter interactions.

Morris design - general nonparametric method.

Delta Test - general nonparametric method (use MC or quasiMC).

Sum-of-tree Test - general nonparametric method (use MC or quasi-MC).

Response surface-based - use any space-filling sample to create approximate response surfaces and run global sensitivity analysis or use ranking analysis if MARS is used.

4.4 Main Effect Analysis

Main effect analysis studies the first order sensitivities of individual input parameter based on variance decomposition. The sensitivity indices can be computed using replicated Latin hypercube sampling with the main effect analysis, the FAST sampling and analysis, or direct numerical integration using response surfaces. In the following example, we describe the use of the replicated Latin hypercube approach.

```
PSUADE
INPUT
```

```
   dimension = 2
   variable  1 X1  =   0.0   1.0
   variable  2 X2  =   0.0   1.0
END
OUTPUT
   dimension = 1
   variable 1 Y1
END
METHOD
   sampling = LH
   num_samples = 1000
   num_replications = 50
   randomize
END
APPLICATION
   driver = ./testmain.py
END
ANALYSIS
   analyzer method = MainEffect
END
END
```

Here the sample size is 1000 based on 50 replications of Latin hypercube with $1000/50 = 20$ levels. To run this analysis, go to Examples/UserExample and issue the following:

```
[Linux] psuade psuadeME.in
```

At the conclusion of the analysis, main effect statistics will be displayed. More information will be displayed if the diagnostics level is increased.

There are other ways to compute main effects such as:

- create response surface and run Sobol analysis
- use any sampling scheme and run main effect ('me') analysis
- if the data fits polynomials well, use Legendre response surface analysis which will output main effects.

In addition, two-parameter Sobol analysis and also total sensitivity index analysis are availalble via response surfaces. If you have large sample and it is not convenient to use response surface, main effect, two-way interaction, and total effect can still be computed using the me, ie, and tsi commands. PSUADE also provides a bootstrapped analysis rs_qsa to compute the additional error bars for the Sobol indices. In addition, for total sensitivity indices, the extended Fourier Amplitude Sampling Test (FAST) can be used with the FAST analyzer.

4.5 Response Surface Analysis

Since main effect analysis requires many model evaluations to ensure sufficient accuracy, it may not feasible for computationally expense functions. However, when the input-output relationship is well-behaved (namely, smoothly varying), response surface methods can be used to create a cheap surrogate for the actual function. In the following, we describe how to use PSUADE to create response surfaces and how to use response surface for quantitative analysis. The first step in creating a response surface is to select a sampling method. Typically, a space-filling sample such as quasi-Monte Carlo or maxi-min Latin hypercube is adequate (PSUADE also has advanced capabilities to create response surfaces using adaptive sampling techniques). The input file (psuadeRS.in in the Examples/UserExample directory) for sampling is:

```
PSUADE
INPUT
   dimension = 2
   variable  1 X1  =   0.0   1.0
   variable  2 X2  =   0.0   1.0
END
OUTPUT
   dimension = 1
   variable 1 Y1
END
METHOD
   sampling = LPTAU
   num_samples = 300
END
APPLICATION
   driver = ./testmain.py
END
ANALYSIS
   diagnostics 1
END
END
```

Here we choose the LP-τ quasi-Monte Carlo method based on the Sobol' sequence with a sample size of 300. This sample is then evaluated via

```
[Linux] psuade psuadeRS.in
```

At the conclusion of the runs, a file named psuadeData will be created which contains the data set (inputs and outputs). Next, rename this file to, for example, psData. In the next step, we check to to see if this data set can be fit with some surface-fitting schemes. A popular scheme is the multivariate adaptive regression splines (MARS) developed by

Jerome Friedman. There are many other options to select from. For example, users can specify a functional forms by providing a function evaluation python script together with the user_regression option. Information on the corresponding file format can be obtained by just using the user_regression option.

To check the goodness of the fit, we launch the PSUADE command line mode and the details of a session is given below:

```
[Linux] psuade
*** ********************************************************* ***
*** Welcome to PSUADE (version 1.4d)                          ***
*** ********************************************************* ***
PSUADE - A Problem Solving environment for
         Uncertainty Analysis and Design Exploration
psuade> load psData
load complete : nSamples = 300
                nInputs  = 2
                nOutputs = 1
psuade> ana_expert
psuade> rscheck
Which response surface tool to use ?
Available response surface tools:
0. MARS
1. Linear regression
.....
12. MARS with bootstrap aggregating (bagging)
Enter you choice ? 0
.....
RSFA: L  0: interpolation error on training set =
L1n error = 2.431e-04 (unscaled), 3.540e-04(scaled)
Avg error = -4.150e-08 (unscaled), 4.574e-05(scaled)
RMS error = 5.048e-04 (unscaled), 9.115e-04(scaled)
Max error = 5.338e-03 (unscaled, BASE=1.004e+00)
Max error = 6.774e-03 (  scaled, BASE=1.333e-01)
Based on 300 training points (total=300).
Perform cross validation ? (y or n) y
Enter the number of groups to validate : (2 - 300) 30
.....
.....
psuade> quit
[Linux]
```

The command ana_expert (analysis expert mode) tells PSUADE to ask if cross validation is to be performed. rscheck first asks for which response surface tool to select, whether

transformation should be done to the inputs and outputs (say no), and then it displays the interpolation error statistics (error checking on the original data set). If cross validation is selected, the number of groups is to be specified (that is, divide the sample into k groups, hold out one group at a time and compile prediction error statistics). Subsequently, cross validation will be performed, a summary of error statistics will be displayed and an error file (RSFA_CV_ERR.m) will be created to be plotted with Matlab.

If rscheck shows that the sample is not good enough to represent the model input-output relationship (the scaled RMS error or the Max error is large), more sample points should be added to the original sample until sufficient accuracy is attained. PSUADE provides two commands to perform sample refinement: refine and a_refine in the command line mode. The former approximately doubles the sample size by uniformly adding more sample points in the parameter space, while the latter refines based on some local error measure.

Once the sample and the corresponding response surface scheme are deemed to be satisfactory, the data set (say, again in the psData file), it can be used for main effect analysis. The setup is the same as in the last section, except that the definition of driver is different in this case (this is the psuadeRSME.in file in Examples/UserExample):

```
PSUADE
INPUT
   dimension = 2
   variable  1 X1  =   0.0   1.0
   variable  2 X2  =   0.0   1.0
END
OUTPUT
   dimension = 1
   variable 1 Y1
END
METHOD
   sampling = LH
   num_samples = 1000
   num_replications = 50
   randomize
END
APPLICATION
   driver = psData
END
ANALYSIS
   analyzer method = MainEffect
END
END
```

Again, simple run the following command

```
[Linux] psuade psuadeRS.in
```

and main effect results will be displayed.

More advanced features for main effect analysis include the handling of correlated inputs (governed by some inequality conditions).

4.6 Numerical Optimization

Let the function for numerical optimization be the two-dimensional Rosenbrock function:

$$Y = 100(X_2 - X_1^2)^2 + (1 - X_1)^2, \quad X_i \in [-2, 2].$$

The PSUADE input file for numerical optimization can be constructed as follow (here bobyqa is a public domain software developed by Michael Powell):

```
PSUADE
INPUT
   dimension = 2
   variable  1 X1  =   -2.0   2.0
   variable  2 X2  =   -2.0   2.0
END
OUTPUT
   dimension = 1
   variable 1 Y1
END
METHOD
   sampling = FACT
   num_samples = 9
END
APPLICATION
   opt_driver = ./simulator
END
ANALYSIS
   optimization method = bobyqa
   optimization num_local_minima = 3
   optimization max_feval = 10000
   optimization tolerance = 1.0e-4
   optimization print_level = 2
END
END
```

This analysis first creates a 3×3 factorial sample. The 9 sample points are evaluated and the 3 (since num_local_minima $= 3$ points with the lowest output values are selected as the starting points for a multi-start optimization. The maximum number of function evaluation is set to be 10000 and the termination tolerance is set to be $1e-4$. The driver points to an executable called simulator. Again, a PSUADE data file such as psData can be used instead.

Users can also specify their own initial points which have the same format as in the PSUADE_IO section in the psData file.

The above example is located in the Examples/OptRosenbrock directory. Simply compile the simulator.c file and then run psuade psuadeBobyqa.in to see optimization in action.

There are other advanced features in optimization such as avoiding repeated function evaluations (this is very useful for restart in the case when the function evaluation is expensive).

4.7 Bayesian Calibration

PSUADE also provides a Bayesian method for stochastic optimization or calibration. Since this method uses the expensive Markov Chain Monte Carlo (MCMC) algorithm to perform the inversion, it currently only supports the function being the response surfaces. The steps in performing this analysis are:

1. generate a sample for building response surfaces,
2. validate the response surfaces and add more points if needed,
3. now you have an evaluated sample in a file called psData.

Next, launch PSUADE in command line mode, load the sample data and issue the 'rsmcmc' command as follow:

```
[Linux] psuade
psuade> load psData
psuade> rsmcmc
....
```

You wil be asked to provide a script providing information on how to construct the likelihood function. An example is:

```
PSUADE_BEGIN
2
1  2.0 0.1
2  5.0 1.0
PSUADE_END
```

The number in the second line of this file should match the number of outputs in the psData file. The third line indicates that the first output has a mean of 2.0 and a standard deviation of 0.1. The fourth line corresponds to the same information for the second output. After this, 'rsmcmc' will run its course and at the end two files will be created: matlabmcmc.m and matlabmcmc2.m which show the posterior distributions. There are other features in this command which can be enabled by turning on the ana_expert mode.