

UQ-PyL User Manual (Version 1)

Chen Wang (<u>wangchen@mail.bnu.edu.cn</u>)

Qingyun Duan (<u>qyduan@bnu.edu.cn</u>)

Beijing Normal University Beijing, China

Table of Contents

1 Introduction	3
1.1 A Quick Start	3
1.2 Available UQ-PyL Capabilities	3
1.2.1 Design of Experiment	3
1.2.2 Uncertainty Analysis	3
1.2.3 Sensitivity Analysis	3
1.2.4 Surrogate Modeling	3
1.2.5 Parameter Optimization	4
1.3 Overview about functionality of the UQ-PyL package	4
2 Installation	5
2.1 Dependencies	5
2.2 Detailed Installation	6
2.2.1 Windows platform	6
2.2.2 Linux platform	12
2.2.3 MacOS platform	
3 Using UQ-PyL	22
3.1 UQ-PyL Flowchart	23
3.2 UQ-PyL Main Frame	24
4 Examples	25
4.1 Sobol' g-function	25
4.1.1 Problem Definition	25
4.1.2 Design of Experiment	26
4.1.3 Uncertainty Analysis	
4.1.4 Sensitivity Analysis	35
4.2 SAC-SMA model	40
4.2.1 Problem Definition	40
4.2.2 Design of Experiment	46
4.2.3 Sensitivity Analysis	49
4.2.4 Surrogate Modeling	52
4.2.5 Parameter Optimization	54

1 Introduction

1.1 A Quick Start

UQ-PyL (<u>U</u>ncertainty <u>Q</u>uantification <u>Py</u>thon <u>L</u>aboratory) is a software platform for performing various uncertainty quantification (UQ) activities such as Design of Experiments (DoE), Uncertainty Analysis (UA), Sensitivity Analysis (SA), Surrogate Modeling and Parameter Optimization. This document describes how to set up problems and use these UQ methods to solve them through UQ-PyL. The mathematics of those UQ methods can be found in the separate theory manual.

We request that you cite the following paper when you report the results obtained by using the UQ-PyL software platform:

C. Wang, Q. Duan, Charles H. Tong, W. Gong, (2015), UQ-PyL – A GUI platform for uncertainty quantification of complex models. Under review for Environmental Modeling & Software.

1.2 Available UQ-PyL Capabilities

1.2.1 Design of Experiment

Full-Factorial design, Fractional-Factorial design, Plackett-Burman design, Box-Behnken design, Central-Composite design, Monte Carlo design, Latin Hypercube design, Symmetric Latin Hypercube design, Improved Distributed Hypercube design, Sobol' sequence, Halton sequence, Faure sequence, Hammersley sequence, Good Lattice Point.

1.2.2 Uncertainty Analysis

Statistical moments, Confidence interval, Hypothesis test.

1.2.3 Sensitivity Analysis

Morris One at A Time (MOAT), Derivative-based Global Sensitivity Measure (DGSM), Sobol' Sensitivity Analysis (Sobol'), Fourier Amplitude Sensitivity Test (FAST), Metamodel-based Sobol', Correlation analysis, Delta Moment-Independent Measure (Delta).

1.2.4 Surrogate Modeling

Polynomial, Generalized Linear Model, Regression Tree, Random Forest, Nearest Neighbors, Support Vector Machine, Gaussian Process.

1.2.5 Parameter Optimization

Shuffled Complex Evolution (SCE), Dynamically Dimensional Search (DDS), Adaptive Surrogate Modeling based Optimization (ASMO), Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Monte Carlo Markov Chain (MCMC).

1.3 Overview about functionality of the UQ-PyL package

1	initpy		
2	Doe/		
3	initpy	#	Ensure all needed files are loaded
4	mainpy	#	For GUI uses
5	box_behnken.py	#	Box-behnken design
6	СС.ру	#	CC design
7	central_composite.py	#	Central-composite design
8	Fl.py	#	F1 design
9	F2.py	#	F2 design
10	fast_sampler.py	#	FAST sensitivity analysis design
11	faure.py	#	Faure design
12	ff2n.py	#	Factorial design
13	finite_diff.py	#	DGSM sensitivity analysis design
14	frac_fact.py	#	Factorial design
15	full_fact.py	#	Full Factorial design
16	GH.py	#	GH design
17	GL.py	#	GL design
18	GLP.py	#	Good Lattic Point design
19	GP.py	#	GP design
20	halton.py	#	Halton Quasi-Monte Carlo design
21	hammersley.py	#	Hammersley Quasi-Monte Carlo design
22	ihs.py	#	ihs design
23	LG.py	#	LG design
24	lhs.py	#	Latin Hypercube design
25	monte_carlo.py	#	Monte Carlo design
26	morris_oat.py	#	Morris One at A Time design
27	plackett_burman.py	#	Plackett Burman design
28	saltelli.py	#	Sobol' sensitivity analysis design
29	sobol.py	#	Sobol' Quasi-Monte Carlo design
30	symmetric_LH.py	#	Symmetric Latin Hypercube design
31	analysis/		
32	initpy	#	Ensure all needed files are loaded
33	mainpy	#	For GUI uses

```
# Confidence Interval
34
     confidence.py
      correlations.py
                            # Correlation analysis
35
36
                            # Delta sensitivity analysis
     delta.py
                            # DGSM sensitivity analysis
37
      dgsm.py
      extended_fast.py  # FAST sensitivity analysis
38
                            # Hypothesis Test
39
      hypothesis.py
40
     moments.py
                            # Statistics moments method
                            # MOAT sensitivity analysis
41
     morris.py
      sobol_analyze.py  # Sobol' sensitivity analysis
sobol_svm.py  # Metamodel based sobol' anal
42
                            # Metamodel based sobol' analysis
43
  RSmodel/
44
45
      __init__.py
                            # Ensure all needed files are loaded
46
      main .py
                             # For GUI uses
47
     DT.py
                             # Decision Tree regression
48
                             # Gaussian Process regression
     gp.py
     kNN.py
                            # k-nearest neighbour regression
49
                             # MARS regression
50
     MARS.py
     polynomial.py
                            # Polynomial regression
51
52
     RF.py
                             # Random Forest regression
53
     SVR.py
                              # Support Vector Machine regression
54 optimization/
      __init_ .py
55
                             # Ensure all needed files are loaded
                             # For GUI uses
56
      main .py
57
      ASMO.py
                             # ASMO optimization
58
     DDS.py
                             # DDS optimization
                            # Genetic Algorithm optimization
59
     GA.py
                            # Particle Swarm Optimization
60
     PSO.py
                             # Simulated Annealing optimization
61
     SA.py
                              # SCE-UA optimization
62
     SCE.py
63 util/
64
      init .py
                            # Ensure all needed files are loaded
      discrepancy.py
                            # Compute discrepancy of design
65
```

2 Installation

2.1 Dependencies

UQ-PyL is an open-source package written in Python. It runs on all major platforms (Windows, Linux, MacOS). It requires some pre-installed standard Python packages:

 \Rightarrow Python version >= 2.7.6

♦ Numpy >= 1.7.1

- ♦ Scipy >= 0.16.0
- \diamond Matplotlib >= 1.4.3
- ♦ PyQt4
- ♦ Scikit-learn

2.2 Detailed Installation

2.2.1 Windows platform

For Windows platform, there is a software integrate Python and some common packages called Python(xy). It contains all the packages UQ-PyL needed. You can just install Python(xy) and UQ-PyL to run UQ analysis.

Step 1. Install Python(xy) software.

You can download "Python(xy)" from our website. Double click the Installation file to start installation.

😢 Python(x, y) 2.7.6	.O Setup	
python(x,y)	License Agreement Please review the license terms before installing Python(x, 2.7.6.0.	y)
Press Page Down to see the	rest of the agreement.	
Licensed under	Copyright © 2008 Pierre Raybaut r the GNU General Public License version 3	
Python(x,y) components are under their own copyright ar (x,y) software collection (i.e. under the terms of the <u>GNU</u> < <u>http://www.qnu.org/licens</u>	distributed as they were received from their copyright holder, id/or license, and without any linking with each other. Python the <i>collection</i> of software, libraries and documents) is licensed <u>General Public License version 3</u> <u>es/qpl.txt></u> .	ł
GNU GENERA Version 3, 29	L PUBLIC LICENSE June 2007	~
If you accept the terms of th agreement to install Python(ne agreement, click I Agree to continue. You must accept the x,y) 2.7.6.0.	•
Python(x,y), the Python Distrib	ution made by Scientists for Scientists	ncel

Click "I Agree" to continue.

۲	Python(x,y) 2.7.6.0 Setup – 🗆 🗙
python(x,y)	Choose Users Choose for which users you want to install Python(x,y) 2.7.6.0.
Select whether you want t computer. Click Next to co	to install Python(x,y) 2.7.6.0 for yourself only or for all users of this ntinue.
Install for anyon	e using this computer
◯ Install just for m	•
Python(x,y), the Python Dist	ribution made by Scientists for Scientists

Click "Next" to continue.

🤨 P	ython(x,y) 2.7.6.0 Setup 🛛 🗕 🗖 🗙				
python(x,y)	Choose Components Choose which features of Python(x,y) 2.7.6.0 you want to install.				
Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.					
Select the type of install:	Custom V				
Or, select the optional components you wish to install:	Python 2.7.6 Python Other Plugins				
Concernantical (72 DVD	Description				
Space required: 473.5MB	Position your mouse over a component to see its description.				
Python(x,y), the Python Distribution made by Scientists for Scientists —					
	< <u>B</u> ack <u>N</u> ext > Cancel				

Choose "Custom" type to install.

0	Python(x,y) 2.7.6.0 Setup –	×			
Choose Components Choose which features of Python(x,y) 2.7.6.0 you want to install.					
Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.					
Select the type of install:	Custom	~			
Or, select the optional components you wish to install:	Python ♥ Base Libraries 1.5.0-10 ♥ Base Python 1.9.2-24 ♥ setuptools 3.0-12 ♥ requests 2.2.1-1 ♥ html5lib 0.999-2 ♥ Description	~			
Space required: 473.5MB	Position your mouse over a component to see its description,				
Python(x,y), the Python Distribution made by Scientists for Scientists					
	< <u>B</u> ack <u>N</u> ext > C	ancel			

For "Python" option, you must check all the package UQ-PyL needed.

PyQt 4.9.6-4 NumPy 1.8.0-5 Scipy 0.13.3-6 Matplotlib 1.3.1-4 Scikit-learn 0.14.1-4 (**Please note: this one is not checked by default**) "

Click "Next" to continue.

😵 Python(x, y) 2.7.6.0 Setup				
Choose Install Location Choose the folder in which to install Python(x,y) 2.7.6.0.				
Setup will install all Python(x,y) components in the following folder. Installation folders of included packages may be customized (see previous page).				
To install in a different folder, click Browse and select another folder. Click Next to continue.				
- Puthon(x, y) Race Installation Directory				
C:\Program Files\pythonxy Browse Browse				
Space required: 535.7MB Space available: 25.0GB				
Python(x,y), the Python Distribution made by Scientists for Scientists				
< <u>B</u> ack <u>N</u> ext > Cancel				

Click "Next" to continue.

😢 Python (ж,у) 2.7.	6.0 Setup	
	ython(x,y)	Choose Start Menu Folder Choose a Start Menu folder for the Python(x,y) 2.7.6 shortcuts.	.0
Select the S can also en	itart Menu foldo ter a name to c	er in which you would like to create the program's shortcu reate a new folder.	ts. You
Python(x,y	0		
CMake 2.8 EndNote Foxit Softw Google Ch MathType MICAPS3.1 Microsoft 9 PyQt GPL Python 2.7 R	vare ome 浏览器 6 2)ffice iilverlight v4.8.1 for Pyth	on v2.7	
Do not o	reate shortcuts	5	
Python(x,y), t	ne Python Distr	ibution made by Scientists for Scientists	Cancel

Click "Install", then waiting for the installation process.

After installation, you executable python.exe file will be C:\Python27\python.exe. All the package will be in the C:\Python27\Lib\site-packages directory.

Step 2. Install UQ-PyL software

se a la companya de l	UQ-PyL Software - 🗆	×
	Installing UQ-PyL Software	^
		~
	Destination folder	
	Extraction progress	
	Extract Cancel	

Choose the default directory D:\ or your own path, then click "unzip" to continue.

:::	UQ-PyL Software 🗧 🗆	×
	Extracting files to D:\folder Extracting from UQ-PyL.exe	
	Extracting UQ-PyL\UQ\optimization\examples\ine.py	
	Extraction progress	
	Extract Cancel	

After unzip, there will be a shortcut on the desktop, double click the shortcut to start UQ-PyL software. If the shortcut doesn't work, please go to your install path, double

click the "main.pyw" file to start it.



5		UQ-PyL -	- Uncertainty Quantification	on Python Laboratory		- 🗆 🗙
<u>F</u> ile <u>A</u> bout						
Problem Definition	Design of Exp	periment	Uncertainty Analysis	Sensitivity Analysis	Surrogate Modelling	Optimi 2at
	Add Input Varia	bles				
_	Parameter Na	me:				
→ <u></u>						
	0.00	wer Bound:				÷
Input Variables	Parameter Upp	per Bound:				
	1.00					+
	Parameter Di Uniform	stribution:				-
			Add		Reset	
Driver Generator	-					
	- Show input vari	ables				
	Param	eter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	^
	1					
	2					
	3					
	4					
	5					~
	6				· · · · · · · · · · · · · · · · · · ·	
				to Parameter File		

2.2.2 Linux platform

Canopy is a globally recommended Python distribution. It contains Python and 100+ common built-it packages. It also contains all the package UQ-PyL used in one software. So you can install Canopy for all the dependences UQ-PyL needed. Please go to the official website (<u>https://www.enthought.com/products/canopy/</u>) for more information.

Step 1. Install Canopy software.

Canopy is a commercial software. However, it provide free use for academic usage. If you use Canopy for education or academic, you can download canopy-1.5.5-full-rh5-64.sh from our website or from Canopy official website. After downloading, you should install Canopy by steps below:

chmod 755 canopy-1.5.5-full-rh5-64.sh ./canopy-1.5.5-full-rh5-64.sh

```
Welcome to the Canopy 1.5.5 installer!
To continue the installation, you must review and approve the license term
agreement.
Press Enter to continue
>>> []
```

If you approve the license term, press Enter to continue

```
_____
                        ____
Canopy Product
                       License
                       Canopy Express Software License Agreement
Express
Basic & Professional Canopy Subscription License Agreement
                       Canopy Software License for Academic Use
Academic
Please review your applicable license carefully.
By installing or using a Canopy product you
signify your assent to and acceptance of the terms of the applicable license to Canopy. If
you do not accept the terms of the applicable
license, then you must not use the Canopy
products. Should you have any questions
regarding licensing, please contact us at
support@enthought.com.
ENTHOUGHT CANOPY EXPRESS
Software License Agreement
This Enthought Canopy Express Software License
Agreement (the ? . greement? . is between Enthought,
Inc., a Delaware corporation (? . nthought? . , and
the licensee subscriber who accepts the terms of
this Agreement (the ? . ustomer? . . The effective
Do you approve the license terms? [yes|no]
[no] >>> yes
```

Type "yes" then press Enter to continue.

```
Canopy will be installed to this location:
/home/quanjp/Canopy
* Press Enter to accept this location
* Press CTRL-C to abort
* or specify an alternate location. Please ensure that your location
contains only ASCII letters, numbers, and the following punctuation
chars: '.', '_', '-'
[/home/quanjp/Canopy] >>> /home/quanjp/swgfs/software/Canopy]
```

Type the path you want to install Canopy, then press Enter to continue.

```
Installing to /home/quanjp/swgfs/software/Canopy ... please wait
Must specify the vendor namespace for these files with --vendor
No directories in update-desktop-database search path could be processed and updated.
***
* Updating MIME database in /home/quanjp/.local/share/mime...
Wrote 2 strings at 20 - 44
Wrote aliases at 44 - 48
Wrote parents at 48 - 4c
Wrote literal globs at 4c - 50
Wrote suffix globs at 50 - 108
Wrote full globs at 108 - 10c
Wrote magic at 10c - 118
Wrote namespace list at 118 - 11c
***
done.
   You can run the Canopy graphical environment by running the script:
        /home/quanjp/swgfs/software/Canopy/canopy
   or by selecting 'Canopy' in your Applications menu.
   On your first run, your Canopy User Python environment will be initialized,
   and you will have the opportunity to make Canopy be your default Python
   at the command line. Details at support.enthought.com/forums
   Thank you for installing Canopy!
```

Complete to install Canopy.

Step 2: Setting up Canopy environment

Enter into the Canopy directory, for me is "/home/quanjp/swgfs/software/Canopy", you can see the file inside it.

[quanjp@login02 Canopy]\$ 11								
total 336								
drwxrwxr-x	3	quanjp	quanjp	32768	Jul	4	07:23	
drwxrwxr-x	2	quanjp	quanjp	32768	Jul	4	07:23	
-rw-rw-r	1	quanjp	quanjp	27157	Jul	4	06:41	_bootpy
-rwxr-xr-x	1	quanjp	quanjp	92	Jul	4	06:41	canopy
-rwxr-xr-x	1	quanjp	quanjp	92	Jul	4	06:41	canopy_cli
-rwxrwxr-x	1	quanjp	quanjp	401	Jul	31	14:06	canopy.desktop
-rw-rw-r	1	quanjp	quanjp	394	Jul	31	14:06	canopy-mime.xml
-rw-rw-r	1	quanjp	quanjp	9704	Jul	4	06:42	canopy.png
drwxrwxr-x	2	quanjp	quanjp	32768	Jul	4	07:23	
drwxrwxr-x	3	quanjp	quanjp	32768	Jul	4	07:23	

Run "./canopy" to setting up Canopy software

×	Canopy Environment Setup
	Canopy System and User environment locations Your Canopy environment will be installed in the location shown below. You may change it, if you wish to. What's this Canopy environment directory
	/home/quanjp/swgfs/software/Python Continue

Enter the Canopy environment directory, for me is "/home/quanjp/swgfs/software/Python", click "Continue" to continue. Your python installation will in this directory.

🔀 Canopy	×
Setting up your Canopy envi	ronment
	2%

After that, a dialogue will display,

Make Canopy your default Python environment?	×
Do you want to make Canopy your default Python environment?	
Yes (Recommended)	
This will give you direct access to Canopy Python, and to utilities like IPython, easy_install, nosetests, from your terminal / command prompt. Learn More	
C No	
Later on, if you want to make Canopy Python the default, you can do so from the preferences dialog. Warning - If you plan to manually specify the full path to Canopy Python, you must specify Canopy's "User" Python, rather than the Cano installation Python. Learn More	ру
Start using Canopy	

Choose "Yes", then click "Start using Canopy".



In "Package Manager" section, you can check what packages in your Python library now.

Actually, you can check your python installation in your python installation path. All "YourPythonPath/User/" files in (for me are is /home/quanjp/swgfs/software/Python/User/). The python executable file is in "YourPythonPath/User/bin/" and packages all the are installed in "YourPythonPath/User/lib/python2.7/site-packages/".

Step 3: Test your Python installation

If you have multiple python environment, please specific one. Usually, modify

your .bashrc file can do it. Add two sentence into your .bashrc file:

```
export PYTHON=/home/quanjp/swgfs/software/Python/User/bin export PATH=$PATH:$PYTHON:
```

Then enter command "source .bashrc" to make your .bashrc file renew.

Type "python" or "python2.7" command, if you can see "Enthought Canopy Python" that means you already accomplished the installation.

```
Enthought Canopy Python 2.7.9 | 64-bit | (default, Jun 30 2015, 22:40:22)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-55)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
```

You can check if all the packages UQ-PyL needed are already installed. Using "import" command, if no error messages that means you already have all the packages.



Step 4. Install UQ-PyL software

Download UQ-PyL Linux version, unzip the source code using command

tar -zxvf UQ-PyL_Linux.tar.gz

Then enter into the UQ-PyL directory

cd UQ-PyL_Linux

Enter command to run UQ-PyL

python main.pyw (or python2.7 main.pyw)

	UO-PvL	Uncertainty Quantific	ation Python Laborat	orv	_ 🗆 🗙
<u>File About</u>				,	
Problem Definition	Design of Experiment	Uncertainty Analysis	Sensitivity Analysis	Surrogate Modelling	Optimization
	Add Input Variables				
	Parameter Name:				
	Parameter Lower Bour	nd:			
	0.00				÷
Input Variables	Parameter Upper Bou	nd:			
	Parameter Distribution	P.			⊒
	Uniform				
Driver Generator		Add		Reset	
briver Generator	L				
	Show input variables				
	Parameter Name	Parameter Lower Bo	und Parameter Upp	er Bound Parameter D	Jistribution
	1				
	2				
	3				
	4				
	5				
	6				
			Save to Parameter Hie		
					- 🐣 📋 all

You can see the main page of UQ-PyL software.

2.2.3 MacOS platform

For MacOS platform, Canopy also has a MacOS version. You can download Canopy software and UQ-PyL MacOS version from our website. The installation process is very similar with Linux platform.

Step 1. Install Canopy software.

First, double click the .dmg file to start the installation.



Pull Canopy icon to Application folder.



Step 2: Setting up Canopy environment

Double click "Canopy" icon to start setting Canopy environment.

\mathbf{O}	Canopy Environment S	etup
Canopy Sy	stem and User environment l	ocations
Your Canop You may cha	environment will be installed in t inge it, if you wish to. <u>What's this</u>	he location shown below.
Canopy en	vironment directory	Change
/Users/wa	ngchen/Library/Enthought/(Canopy_64bit
		Continue

Write Canopy environment directory, click "Continue" to continue. Your python installation will be in this directory.

$\Theta \cap \Theta$	Canopy	
Setting up	o your Canopy environment	
After that, a dial	ogue will display,	

💿 Yes (F	Recommended)
This will g easy_insta	ive you direct access to Canopy Python, and to utilities like IPython, Il, nosetests, from your terminal / command prompt. <u>Learn More</u>
🔵 No	
Later on, i preference Canopy Py installatio	f you want to make Canopy Python the default, you can do so from the es dialog. Warning – If you plan to manually specify the full path to thon, you must specify Canopy's "User" Python, rather than the Canopy n Python. <u>Learn More</u>

Choose "Yes", then click "Start using Canopy".



Also, you can check your python installation in your python installation path. All files are in *"YourPythonPath/User/"* (for me is /Users/wangchen/Library/Enthought/Canopy_64bit/User/). The python executable file is in "YourPythonPath/User/bin/".

Step 3: Test your Python installation

If you have multiple python environment, please specific one. For MacOS you could add a line like this to the /etc/launchd.conf file

export PYTHONPATH=/Users/wangchen/Library/Enthought/Canopy_64bit/User/bin

Then enter command "source launchd.conf" to make your launchd.conf file renew.

Type "python" or "python2.7" command, if you can see "Enthought Canopy Python" that means you already accomplished the installation.

```
ouchenmatoMacBook-Pro:UQ-PyL_Linux wangchen$ python
Enthought Canopy Python 2.7.9 | 64-bit | (default, Jun 30 2015, 19:41:21)
[GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2335.6)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
```

Step 4. Install UQ-PyL software

Download UQ-PyL MacOS version, unzip the source code using command

tar -zxvf UQ-PyL_Mac.tar.gz

Then enter into the UQ-PyL directory

cd UQ-PyL_Mac

Enter command to run UQ-PyL

python main.pyw (or python2.7 main.pyw)

You can see the main page of UQ-PyL software.

_			1	1		
Prob	olem Definition	Design of Experim	ent Uncertainty Analys	is Sensitivity Analysis	Surrogate Modelling	Optimization
	Add Input Va	riables				
	Paramet	or Name:				
	raramet	er Name.				
	Paramet	er Lower Boun	d:			
	0.00					
Input Variables	Paramet	er Upper Boun	d:			
	1.00					
	Paramet	er Distribution	:			
	Uniform					
_						
			Add	11	Poset	
Driver Generator	Show input v	ariables	Add			
Driver Generator	Show input va	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	1
Driver Generator	Show input va	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	1
Driver Generator	Show input va	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	n
Driver Generator	Show input vi Pa 1 2 3	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	n
Driver Generator	Show input va	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	n
Driver Generator	Show input vi Pa 1 2 3 4 5	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	n
Driver Generator	Show input vi Pa 1 2 3 4 5 6	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	n
Driver Generator	Show input vi Pa 1 2 3 4 5 6	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	
Driver Generator	Show input vi Pa 1 2 3 4 5 6	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	
Driver Generator	Show input vo	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	1
Driver Generator	Show input v.	ariables rameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	

3 Using UQ-PyL

3.1 UQ-PyL Flowchart

Fig. 1 is the flowchart illustrating how UQ-PyL executes an UQ task. A typical task is carried out in three major steps: (1) model configuration preparation; (2) uncertainty propagation; and (3) UQ analysis. In the first step, the user specifies the model configuration information (i.e., parameter names, ranges and distributions), and the DoE information (i.e., the sampling techniques and sample sizes) to prepare for UQ exercise for a given problem. In the second step, the different sample parameter sets generated in the last step are fed into the simulation model (or mathematical function) to enable the execution of simulation model (function calculation). In the third step, a variety of UQ exercises are carried out, including UA, SA, surrogate modelling and parameter optimization.



Fig 1. UQ-PyL flowchart

3.2 UQ-PyL Main Frame

UQ-PyL is equipped with a Graphic User Interface (GUI) to facilitate execution of various functions, but it can also run as a script program in a batch mode. Fig. 2 shows the front page of UQ-PyL. Different tab widgets allow user to execute different steps of UQ process, including problem definition, DoE, UA, SA, surrogate modeling and parameter optimization. One may click on the desired tab by mouse and/or enter the required information via keyboard to perform various tasks. After a task is completed, the software generates tabular results and/or graphical outputs. The graphical outputs can be saved in a variety of formats, including .png, .bmp, .tiff or .pdf formats, among others.

		UQ-Py	Uncertainty Quantificatio	n Python Laboratory		- 🗆 🗙
<u>File</u> <u>A</u> bout						
Problem Definition	Desi	gn of Experiment	Uncertainty Analysis	Sensitivity Analysis	Surrogate Modelling	Optimi 2a
Input Variables	Add 1 Par ZPE Par 5.0 Par 350 Par Uni	Input Variables ameter Name: 20 ameter Lower Bound: 0 ameter Upper Bound: 00 ameter Distribution: form	Add		Reset	• •
Driver Generator						
	Show	input variables				
		Parameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	^
	1	UZTWM	10.00	300.00	Uniform	
	2	11704/64	5.00	450.00	11-26	
	2	UZEVVIVI	5.00	150.00	Uniform	
	3	UZK	0.10	0.75	Uniform	
	3	UZK PCTIM	0.10 0.00	0.75	Uniform Uniform	-
	2 3 4 5	UZK PCTIM ADIMP	0.10 0.00 0.00	0.75 0.10 0.20	Uniform Uniform Uniform	
	2 3 4 5 6	UZK PCTIM ADIMP	0.10 0.00 0.00	0.75 0.10 0.20	Uniform Uniform Uniform	- - -
	2 3 4 5 6	UZK PCTIM ADIMP	0.10 0.00 0.00 Save	0.75 0.10 0.20 to Parameter File	Uniform Uniform Uniform	~
	2 3 4 5 6	UZK PCTIM ADIMP	0.10 0.00 0.00 Save	0.75 0.10 0.20 to Parameter File	Uniform Uniform Uniform	~
	2 3 4 5 6	UZK PCTIM ADIMP	0.10 0.00 0.00 Save	0.75 0.10 0.20 to Parameter File	Uniform Uniform Uniform	v
	2 3 4 5 6	UZK PCTIM ADIMP	0.10 0.00 0.00 Save	0.75 0.10 0.20 to Parameter File	Uniform Uniform Uniform	v
	2 3 4 5 6	UZK PCTIM ADIMP	0.10 0.00 0.00 Save	0.75 0.10 0.20 to Parameter File	Uniform Uniform Uniform	~
	2 3 4 5 6	UZK PCTIM ADIMP	0.10 0.00 0.00 Save	0.75 0.10 0.20 to Parameter File	Uniform Uniform Uniform	~

Fig 2. Graphic User Interface of UQ-PyL

4 Examples

4.1 Sobol' g-function

4.1.1 Problem Definition

The expression of sobol' g-function is:

$$f(x) = \prod_{i=1}^{n} g_i(x_i)$$

where

$$g_i(x_i) = \frac{|4x_i - 2| + a_i}{1 + a_i}$$

The input parameter x_i is uniformly distributed within (0, 1), $a_i = \{0, 1, 4.5, 9, 99, 99, 99, 99\}$.

The model is implemented using Python and the parameter file is shown below:

Model file (UQ-PyL/UQ/test_functions/Sobol_G.py)

```
from __future__ import division
import numpy as np
# Non-monotonic Sobol' G Function (8 parameters)
# First-order indices:
# x1: 0.7165 77.30%
# x2: 0.1791 19.32%
# x3: 0.0237 2.56%
# x4: 0.0072 0.78%
# x5-x8: 0.0001 0.01%
def evaluate(values):
   a = [0, 1, 4.5, 9, 99, 99, 99, 99]
   Y = np.empty([values.shape[0]])
   for i, row in enumerate(values):
      Y[i] = 1.0
      for j in range(8):
         x = row[j]
         Y[i] *= (abs(4*x - 2) + a[j]) / (1 + a[j])
   return Y
```

Parameter file (UQ-PyL/UQ/test_functions/params/Sobol_G.txt)

x1 0.0 1.0 x2 0.0 1.0 x3 0.0 1.0 x4 0.0 1.0 x5 0.0 1.0 x6 0.0 1.0 x7 0.0 1.0 x8 0.0 1.0

Parameter file can also be generated from GUI of UQ-PyL:

Step 1: Enter "Parameter Name", "Parameter Lower Bound" and "Parameter Upper Bound", choose "Parameter Distribution";

Step 2: Click "Add" button to save this parameter information to table widget; Step 3: Enter every parameter's information, click "Save to Parameter File" button, choose the save path "UQ-PyL/UQ/test_functions/params/Sobol_G.txt".

S		UQ-PyL -	- Uncertainty Quantificatio	on Python Laboratory		- 🗆 🛛
File About						
Problem Definition	Desi	gn of Experiment	Uncertainty Analysis	Sensitivity Analysis	Surrogate Modelling	Optimi 🛿 🖈
	Add	Input Variables				
	Par	ameter Name:				
→ —	хЗ					
_	Par	ameter Lower Bound:	Choose parameter i	nformation		
Input Variables	D. C	ameter Upper Bound.				
input variables	1.0	0				•
	Par	ameter Distribution:				
	Uni	form				
			Add		Reset	
Driver Generator						
	- Show	innut variables				
	Ditor	Input variables				
		Parameter Name	Parameter Lower Bound	Parameter Upper Bound	Parameter Distribution	^
	1	x1	0.00	1.00	Uniform	_
	2	x2	0.00	1.00	Uniform	
	3	x3	0.00	1.00	Uniform	
	4					
	5					
	6					~
				to Parameter File		
			Click t	o save parameter file		

4.1.2 Design of Experiment

After problem definition, we do Design of Experiment, the experiment has three

steps:

- 1) Define parameter and model information;
- 2) Choose Design of Experiment method;
- 3) Generate script and run the script.

<u>19</u>		UQ-PyL Uncertainty Quantificati	on Python Laboratory			- 🗆 🛛
File /	About					
Pro	blem Definition De ad Model Information	esign of Experiment Uncertainty Analysis	Sensitivity Analysis	Surrogate	Modelling	Optimi 2at
c	Choose Parameter File:	D:/UQ-PyL/UQ/test_functions/perams/Sobol_G.txt D:/UQ-PyL/UQ/test_functions/Sobol_G.py			Choose Parame Choose Mode	eter File el File
Des	sign of Experiment Method-	Load parameter file and mode	al file			
	hoose DoE method: Lat	tin Hypercube				•
	[Latin Hyperoube Configuration				
		Choose different Latin Hypercube method:	 Random Latin Hypercube Center Latin Hypercube Maximin Latin Hypercube Center Maximin Latin Hypercube Correlation Latin Hypercube 	per cube cube		
		Number of Sample Points:	50			•
		Generate DoE Script				
		Execute DoE Script				
Sho	ow Design of Experiment Re:	sult				
c	Choose Result File:				Choose Res	ult File
	Display Result					

Step 1: Define parameter and model information

- ♦ Switch to "Design of Experiment" tab;
- Click "Choose Parameter File" button to choose "UQ-PyL/UQ/test_functions/params/Sobol_G.txt" file;
- Click "Choose Model File" button to choose
 "UQ-PyL/UQ/test_functions/Sobol_G.py" file.

2	UQ-PyL Uncertainty Quantifica	tion Python Laboratory	_ 🗆 ×
ile About			
Problem Definition	Design of Experiment Uncertainty Analysis	Sensitivity Analysis Surrogate	Modelling Optimila
Load Model Information			
Choose Parameter File	e: D:/VQ-PyL/VQ/test_functions/params/Sobol_G.txt		Choose Parameter File
Choose Model File:	D:/VQ-FyL/VQ/test_functions/Sobol_G.py		Choose Model File
Design of Experiment Metho	od		
Choose DoE method:	Latin Hypercube		
	Latin Mypercube Configuration		
	Choose different Latin Hypercube method:	🔘 Random Latin Hypercube	
		🖲 Center Latin Hypercube	
		🔵 Maximin Latin Hypercube	
		🔘 Center Maximin Latin Hypercube	
		Correlation Latin Hypercube	
	Number of Sample Points:	50	<u>-</u>
	Generate DoE Script	Choose DoE method and	
	Execute DoE Script	define number of sample p	oints
Show Design of Experiment	Result		
Choose Result File:			Choose Result File
Display Result			

Step 2: Choose DoE method

- Choose DoE method, like "Latin Hypercube", choose one specific Latin Hypercube method, like "Center Latin Hypercube";
- ♦ Set "Number of Sample Points", like: 50.

	UQ-PyL Uncertainty Quantifica	ion Python Laboratory	
le About			
Problem Definition	Design of Experiment Uncertainty Analysis	Sensitivity Analysis Surrogat	e Modelling Optimi1
Load Model Information —			
Choose Parameter Fil	e: D:/UQ-PyL/UQ/test_functions/params/Sobol_G.txt		Choose Parameter File
Choose Model File:	D:/UQ-PyL/UQ/test_functions/Sobol_G.py		Choose Model File
-Design of Experiment Meth	od		
Choose DoE method:	Latin Hypercube		•
	Latin Mypercube Configuration		
	Choose different Latin Hypercube method:	🔿 Random Latin Hypercube	
		Center Latin Hypercube	
		🔿 Maximin Latin Hypercube	
		🔿 Center Maximin Latin Hypercube	
		Correlation Latin Hypercube	
	Number of Sample Points:	50	•
	Generate DoE Script	Generate and run the	script
	Execute DoE Script		
•			
-Show Design of Experiment	Result		
Choose Result File:			Choose Result File
Display Result			

Step 3: Run for DoE results

- ♦ Click "Generate DoE Script" button to generate DoE script which contains information you just choose;
- ♦ Click "Execute DoE Script" button to run DoE script.

Then, UQ-PyL gives the tabular and graphic results of DoE:



The result automatically save in text files, the name of files including DoE method used and current time.

 model_output_latin2_2015_05_18_22_12_46.txt
 2015/5/18 22:12

 sample_output_latin2_2015_05_18_22_12_46.txt
 2015/5/18 22:12

This step can also implemented using python script:

Python script file (Sobol_G_DoE.py)

```
# Optional - turn off bytecode (.pyc files)
import sys
sys.dont write bytecode = True
from UQ.DoE import lhs
from UQ.test functions import Sobol G
from UQ.util import scale samples general, read param file, discrepancy
import numpy as np
import random as rd
# Set random seed (does not affect quasi-random Sobol sampling)
seed = 1
np.random.seed(seed)
rd.seed(seed)
# Read the parameter range file and generate samples
param file = './UQ/test functions/params/Sobol G.txt'
pf = read param file(param file)
# Generate samples (choose method here)
param values = lhs.sample(50, pf['num vars'], criterion='center')
res = discrepancy.evaluate(param values)
print res
# Samples are given in range [0, 1] by default. Rescale them to your
parameter bounds.
scale samples general(param values, pf['bounds'])
np.savetxt('Input Sobol\'.txt', param values, delimiter=' ')
# Run the "model" and save the output in a text file
# This will happen offline for external models
Y = Sobol_G.evaluate(param_values)
np.savetxt("Output Sobol\'.txt", Y, delimiter=' ')
```

4.1.3 Uncertainty Analysis

In this section, we do uncertainty analysis using UQ-PyL.

There are also three steps:

- 1) Define parameter and model information;
- 2) Do Design of Experiment or load Design of Experiment results;
- 3) Choose uncertainty analysis method and show the results.

	UQ-F	yL Uncertainty Quantification Python Laboratory	
e About Problem Definition Design	n of Experime	nt Uncertainty Analysis Sensitivity Analysis Surrogate	Modelling Optimil
Perform Design of Experiment			
Load parameter file:	D:/UQ-PyL/UQ/tes	t_functions/params/Sobol_G.txt	Choose Parameter File
Load Model File:	D:/UQ-PyL/UQ/tes	t_functions/Sobol_G.py	Choose Model File
Choose DoE method: Number of Sample Points:	Monte Carlo	Define parameter and model informa	tion
Execute DoE Script Choose Analysis Method			
Load parameter file:		D:/UQ-PyL/UQ/test_functions/params/Sobol_G.txt	Choose Parameter File
Load data file (input file,	output file):		Choose Input File
			Choose Output File
Basic Statistical Analysis 1	Methods:	Statistical Moments Methods 🗸 🗸	Show Results
Advanced Statistical Analys	is Methods:	Pearson/Spearman Correlations Analysis 🔹	Show Results

Step 1: Define parameter and model information

- \diamond Switch to "Uncertainty Analysis" tab;
- Click "Choose Parameter File" button to choose
 "UQ-PyL/UQ/test_functions/params/Sobol_G.txt" file;
- ♦ Click "Choose Model File" button to choose "UQ-PyL/UQ/test_functions/Sobol_G.py" file.

8	UQ-P	/L Unce	ertainty Qu	antificatio	on Python	Laborator	ry				- 🗆 🛛
File About											
Problem Definition Design	n of Experimen	t Unce	ertainty A	nalysis	Sensitiv	ity Analy	vsis	Surroga	te Mc	delling	Optimi 2at
Perform Design of Experiment											
Load parameter file:	D:/UQ-PyL/UQ/test	functions/	params/Sobol_	G. txt					Cl	hoose Param	eter File
Load Model File:	D:/UQ-PyL/UQ/test	functions/	Sobol_G. py							Choose Mod	el File
Choose DoE method:	Monte Carlo								-		
Number of Sample Points:	50							E			
				Loa	d Desigr		rimer	it result	S		
Choose Analysis Method		η: /IIΩ=Ρ ₁ ,ζ /	10/test funct	Loa	G Design	гог схре	rimer	it result	s	haase Perem	atar Rila
Choose Analysis Method Load parameter file: Load data file (input file,	output file):	D:/UQ-PyL/1 D:/UQ-PyL/1	VQ/test_funct sample output	LOa ions/params, latin2 2015	d Design /Sobol_G.txt 5 05 18 22 1:	2 46. txt	rimer	t result	SC1	hoose Paramo Choose Inpu	eter File 1t File
Choose Analysis Method Load parameter file: Load data file (input file,	output file):	D: /VQ-PyL/ D: /VQ-PyL/: D: /VQ-PyL/:	VQ/test_funct sample_output model_output_	LOa ions/params/ _latin2_2015 latin2_2015_	d Design /Sobol_G. txt 5_05_18_22_12 _05_18_22_12	2_46. txt	rimer	it result	S [1	hoose Param Choose Inp Choose Outp	eter File at File ut File
Choose Analysis Method Load parameter file: Load data file (input file, Basic Statistical Analysis)	output file): Methods:	D:/UQ-PyL/ D:/UQ-PyL/ D:/UQ-PyL/ Statistica	UQ/test_funct sample_output model_output_ l Moments Met	LOA ions/perams/ _latin2_2015 latin2_2015 hods	d Design /Sobol_G. txt 5_05_18_22_12 _05_18_22_12	2_46. txt 46. txt	rimer	t result	s (1 (1 (1) (1) (1) (1) (1) (1)	hoose Param Choose Inpu Choose Outp Show Res	eter File nt File ut File ults
Choose Analysis Method Load parameter file: Load data file (input file, Basic Statistical Analysis J Advanced Statistical Analys:	output file): Methods: is Methods:	D:/UQ-FyL/ D:/UQ-FyL/ D:/UQ-FyL/ Statistica Pearson/Sp	VQ/test_funot sample_output model_output_ l Moments Met earman Correl	LOA ions/params/ _latin2_2015 latin2_2015 hods ations Analy	d Design /Sobol_G.txt 5_05_18_22_12 _05_18_22_12 _vsis	2_46. txt _46. txt	rimer	t result	s (1 (1 	hoose Param Choose Inp Choose Outp Show Res Show Res	ater File at File ut File ults ults

Step 2: Load DoE results

- Click "Choose Input File" button to choose sample file you just generated, for example: "sample_output_latin2_2015_05_18_22_12_46.txt";
- Click "Choose Output File" button to choose model output file you just generated, for example: "model_output_latin2_2015_05_18_22_12_46.txt".

	UQ-F	yL Uncertainty Quantification Python Laboratory		_ □ ×
Problem Definition Desig	n of Experime	nt Uncertainty Analysis Sensitivity Analysis S	Surrogate	Modelling Optimi 🖈
Perform Design of Experiment				
Load parameter file:	D:/VQ-PyL/VQ/tes	_functions/params/Sobol_G.txt		Choose Parameter File
Load Model File:	D:/UQ-PyL/UQ/tes	_functions/Sobol_G.py		Choose Model File
Choose DoE method:	Monte Carlo		•	
Number of Sample Points:	50		•	
Generate DoE Script Execute DoE Script				
Choose Analysis Method		D:/VQ-FyL/VQ/test_functions/params/Sobol_G.txt		Choose Parameter File
Load data file (input file,	output file):	D:/UQ-PyL/sample_output_latin2_2015_05_18_22_12_46.txt		Choose Input File
		D:/UQ-FyL/model_output_latin2_2015_05_18_22_12_46.txt		Choose Output File
Basic Statistical Analysis	Methods:	Statistical Moments Methods	-	Show Results
Advanced Statistical Analys	sis Methods:	Pearson/Spearman Correlations Analysis	•	Show Results
		Define uncertainty analysis method and sh	ow result	ts
[💫 🚺 🕅				

Step 3: Choose uncertainty analysis method and show results

- ♦ Choose uncertainty analysis method, like "Statistical Moments Methods";
- ♦ Click "Show Results" button to show uncertainty analysis results.

UQ-PyL gives the tabular and graphic results:





This step can also implemented using python script:

Python script file (Sobol_G_UA.py)

```
# Optional - turn off bytecode (.pyc files)
import sys
sys.dont write bytecode = True
from UQ.DoE import lhs
from UQ.analyze import *
from UQ.test functions import Sobol G
from UQ.util import scale samples general, read param file, discrepancy
import numpy as np
import random as rd
# Set random seed (does not affect quasi-random Sobol sampling)
seed = 1
np.random.seed(seed)
rd.seed(seed)
# Read the parameter range file and generate samples
param file = './UQ/test functions/params/Sobol G.txt'
pf = read_param_file(param_file)
# Generate samples (choose method here)
param values = lhs.sample(50, pf['num vars'], criterion='center')
res = discrepancy.evaluate(param values)
print res
```

```
# Samples are given in range [0, 1] by default. Rescale them to your
parameter bounds.
scale_samples_general(param_values, pf['bounds'])
np.savetxt('Input_Sobol\'.txt', param_values, delimiter=' ')
# Run the "model" and save the output in a text file
# This will happen offline for external models
Y = Sobol_G.evaluate(param_values)
np.savetxt("Output_Sobol\'.txt", Y, delimiter=' ')
# Perform the sensitivity analysis/uncertainty analysis using the model
output
# Creatify which column of the cutput file to evaluate (param_d)
```

```
# Specify which column of the output file to analyze (zero-indexed)
moments.analyze('Output Sobol\'.txt', column=0)
```

4.1.4 Sensitivity Analysis

Next, we do sensitivity analysis using UQ-PyL. There are three steps:

- 1) Define parameter and model information;
- 2) Do specific Design of Experiment or load Design of Experiment results (Different sensitivity analysis method need different Design of Experiment method);
- 3) Choose sensitivity analysis method and show the results.

g UQ-F	yL Uncertainty Quantifica	tion Python Laboratory	_ 🗆 ×
File About			
Problem Definition Design of Experime	nt Uncertainty Analysis	Sensitivity Analysis Surrogate	e Modelling 🛛 Optimi 🕼
Perform Design of Experiment			
Load parameter file: D:/VQ-FyL/VQ/test_funct	ions/params/Sobol_G. txt		Choose Parameter File
Load Model File: D:/VQ-PyL/VQ/test_funct	ions/Sobol_G. py		Choose Model File
Design of Experiment Method	Define para	neter and model information	
Choose DoE method: Morris One at A Time	•		-
- Morries One At A Time(MDAT) Configuration		
*****) • The last of The last of the	
Number of Traject	sampie points – (dimension+i pries:	50	.
	enerate DoE Script		
	Execute DoE Script		
Choose Analysis Method			
		(1) 1	
Load parameter file:	D:/0Q-YyL/0Q/test_tunctions/para	ms/Sobol_G.txt	Choose Farameter File
Load data file (input file, output file):			Choose Input File
			Choose Output File
Sensitivity Analysis Method:	Morris		Show Results

Step 1: Define parameter and model information

- \diamond Switch to "Sensitivity Analysis" tab;
- Click "Choose Parameter File" button to choose
 "UQ-PyL/UQ/test_functions/params/Sobol_G.txt" file;
- Click "Choose Model File" button to choose
 "UQ-PyL/UQ/test_functions/Sobol_G.py" file.

ł	UQ-I	yL Uncertainty Quantifica	tion Python Laboratory	_ □ >
ile About				
Problem Definition	Design of Experime	nt Uncertainty Analysis	Sensitivity Analysis Surrogat	e Modelling Optimiia
-Perform Design of Experime	nt			
Load parameter file:	D:/VQ-PyL/VQ/test_funct	ions/params/Sobol_G. txt		Choose Parameter File
Load Model File:	D:/VQ-PyL/VQ/test_funct	ions/Sobol_G. py		Choose Model File
Design of Experiment Metho	d			
Choose DoE method:	Morris One at A Time			
	Morries One At A Time	MDAT) Configuration		
	**Number of total	sample points = (dimension+1) * Number of Trajectories	
	Number of Traject	mies:	50	\$
		enerate DoE Script		
		Execute DoE Script		
Change Anglanda Webbad	Do specific	Design of Experiment a OR	nd load results	
CHOOSE ANALYSI'S MELHOU	Load Desig	n of Experiment results	directly	
Load parameter file:		D:/VQ-PyL/VQ/test_functions/para	ns/Sobol_G. txt	Choose Parameter File
Load data file (input	file, output file):	D:/UQ-PyL/sample_output_morris_2	015_05_19_17_54_55. txt	Choose Input File
		D:/VQ-PyL/model_output_morris_20	15_05_19_17_54_55. txt	Choose Output File
Sensitivity Analys	is Method:	Morris	•	Show Results
🚞 🎸 📑				

Step 2: Do specific DoE for specific sensitivity analysis method. For example, we do Morris analysis in this chapter. Then load DoE results.

- ♦ Choose DoE method, for this experiment is "Morris One at A Time";
- ♦ Set "Number of Trajectoriese", for example: 50;
- ♦ Click "Generate DoE Script" button to generate script;
- ♦ Click "Execute DoE Script" button to run script and acquire DoE result;
- ♦ Load input/output file you just generated: 1) Click "Choose Input File" button to load sample file, for example

"UQ-PyL/sample_output_morris_2015_05_19_17_54_55.txt"; 2) Click "Choose Output File" button to load model output file, for example

"UQ-PyL/model_output_morris_2015_05_19_17_54_55.txt".

	UQ-PyL -	- Uncertainty Quan	tification I	ython Lab	oratory			
e About					_			
Problem Definition Design	n of Experiment	Uncertainty Anal	lysis 🔇	ensitivity	Analysis	Surrogate	Modelling	Optimi
Perform Design of Experiment								
Load parameter file: D:/VQ-D	PyL/WQ/test_functions,	'params/Sobol_G. txt					Choose Parame	eter File
Load Model File: D:/UQ-D	PyL/VQ/test_functions,	'Sobol_G. py					Choose Mode	el File
Design of Experiment Method								
Choose DoE method: Morris O	ne at A Time							•
Morri	es One At A Time(MOAT)	Configuration						
**N	umber of total sam	ole points = (dimens	sion+1) * N	umber of Tr	viectories			
Num	ber of Trajectorie	s:		50				
	Gener	ate DoE Script						
	Exect	ite DoE Script						
Cho	oose sensitivity	analysis method	and show	v results				
CHOOPE WUNTABLE WELDOD								
Load parameter file:	D:/	UQ-PyL/UQ/test_function	ıs/params/Sob	ol_G. txt			Choose Parame	eter File
Load data file (input file,	output file): D:/	VQ-PyL/sample_output_mo	rris_2015_05	_19_17_54_55.	txt		Choose Inpu	nt File
	D:/	VQ-PyL/model_output_mor	ris_2015_05_	19_17_54_55. t	ĸt		Choose Outp	ut File
Sensitivity Analysis Met	hod: Mor	ris				•	Show Res	ults
📺 🚺 🚺 📷								

Step 3: Choose sensitivity analysis method and show results

- ♦ Choose sensitivity analysis method, like "Morris";
- ♦ Click "Show Results" button to show sensitivity analysis results.

UQ-PyL gives the tabular and graphic results:





This step can also implemented using python script:

Python script file (Sobol_G_SA.py)

```
# Optional - turn off bytecode (.pyc files)
import sys
sys.dont write bytecode = True
from UQ.DoE import morris oat
from UQ.analyze import *
from UQ.test functions import Sobol G
from UQ.util import scale samples general, read param file
import numpy as np
import random as rd
# Set random seed (does not affect quasi-random Sobol sampling)
seed = 1
np.random.seed(seed)
rd.seed(seed)
# Read the parameter range file and generate samples
param file = './UQ/test functions/params/Sobol G.txt'
pf = read param file(param file)
# Generate samples (choose method here)
param_values = morris_oat.sample(50, pf['num_vars'], num_levels = 10,
grid jump = 5)
# Samples are given in range [0, 1] by default. Rescale them to your
```

```
parameter bounds.
scale_samples_general(param_values, pf['bounds'])
np.savetxt('Input_Sobol\'.txt', param_values, delimiter=' ')
# Run the "model" and save the output in a text file
# This will happen offline for external models
Y = Sobol_G.evaluate(param_values)
np.savetxt("Output_Sobol\'.txt", Y, delimiter=' ')
# Perform the sensitivity analysis/uncertainty analysis using the model
output
# Specify which column of the output file to analyze (zero-indexed)
morris.analyze(param_file, 'Input_Sobol\'.txt', 'Output_Sobol\'.txt',
column = 0)
```

4.2 SAC-SMA model

4.2.1 Problem Definition

The SAC-SMA is a rainfall-runoff model which has a highly non-linear, non-monotonic input parameter-model output relationship. There are sixteen parameters in the SAC-SMA model. Thirteen of them are considered tunable, and the other three parameters are fixed at pre-specified values according to Brazil (1988). Table 1 describes those parameters and their ranges.

No.	Parameter	Description	Range
1	UZTWM	Upper zone tension water maximum storage (mm)	[10.0, 300.0]
2	UZFWM	Upper zone free water maximum storage (mm)	[5.0, 150.0]
3	UZK	Upper zone free water lateral drainage rate (day ⁻¹)	[0.10, 0.75]
4	PCTIM	Impervious fraction of the watershed area (decimal	[0.0, 0.10]
		fraction)	
5	ADIMP	Additional impervious area (decimal fraction)	[0.0, 0.20]
6	ZPERC	Maximum percolation rate (dimensionless)	[5.0, 350.0]
7	REXP	Exponent of the percolation equation	[1.0, 5.0]
		(dimensionless)	
8	LZTWM	Lower zone tension water maximum storage (mm)	[10.0, 500.0]
9	LZFSM	Lower zone supplemental free water maximum	[5.0, 400.0]
		storage (mm)	
10	LZFPM	Lower zone primary free water maximum storage	[10.0,
		(mm)	1000.0]
11	LZSK	Lower zone supplemental free water lateral	[0.01, 0.35]
		drainage rate (day ⁻¹)	

12	LZPK	Lower zone primary free water lateral drainage rate	[0.001, 0.05]
		(day ⁻¹)	
13	PFREE	Fraction of water percolating from upper zone	[0.0, 0.9]
		directly to lower zone free water (decimal fraction)	
14	RIVA	Riverside vegetation area (decimal fraction)	0.30
15	SIDE	Ration of deep recharge to channel base flow	0.0
		(dimensionless)	
16	RSERV	Fraction of lower zone free water not transferrable	0.0
		to lower zone tension water (decimal fraction)	
		Table (Denomentary of SAC SMA model	

 Table 6. Parameters of SAC-SMA model

So we generate the parameter file (UQ-PyL/UQ/test_functions/params/SAC.txt) as: UZTWM 10 300 UZFWM 5 150 UZK 0.1 0.75 PCTIM 0 0.1 ADIMP 0 0.2 ZPERC 5 350 REXP 1 5 LZTWM 10 500 LZFSM 5 400 LZFPM 10 1000 LZSK 0.01 0.35 LZPK 0.001 0.05 PFREE 0 0.8

SAC-SMA model is an executable file on Windows or Linux or MacOS system. In order to using UQ-PyL, we need to generate a python driver to couple SAC-SMA model and UQ-PyL platform. The driver file can be generated automatically by UQ-PyL's GUI.

📓 UQ-PyL -	- Uncertainty Quantification Python Laboratory	_ 🗆 🛛
File About		
Problem Definition Design of Experiment	Uncertainty Analysis Sensitivity Analysis	Surrogate Modelling Optimilat
Generate Template File		
Load Model Input File:	D:/UQ-PyL/UQ/test_functions/SAC/ps_test01.sac	Choose Model Input File
Generate Template File		
Input Variables		
Generate Driver	Generate Template file	
Load Parameter File:	D:/UQ-PyL/UQ/test_functions/params/SAC.txt	Choose Parameter File
Load Model Input File:	D:/UQ-PyL/UQ/test_functions/SAC/ps_test01.sac	Choose Model Input File
Driver Generator Load Executable File:	D:/UQ-PyL/UQ/test_functions/SAC/mopexcal.exe	Choose Executable File
Generate Driver		

Step 1: Generate template file

- ♦ Choose "Problem Definition" tab, click on "Driver Generator" widget;
- Click "Choose Model Input File" to load model configuration file, for SAC model is "UQ-PyL/UQ/test_functions/SAC/ps_test01.sac";
- Click "Generate Template File" to generate model configuration template file, this file will be used in model driver file.

1	UQ-PyL -	- Uncertainty Quantification	on Python Laboratory			- 🗆 🛛
File About						
Problem Definition	Design of Experiment	Uncertainty Analysis	Sensitivity Analysis	Surroga	te Modelling	Optimi1at
	Generate Template File					
-> ==	Load Model Input File:	D:/UQ-FyL/UQ/test_functions/S		Choose Model Input File		
Input Variables	Generate Template File					
	Generate Driver					
	Load Parameter File:	D:/UQ-PyL/UQ/test_functions/p	oarams/SAC.txt		Choose Parameter	r File
	Load Model Input File:	D:/VQ-PyL/VQ/test_functions/S	GAC/ps_test01.sac		Choose Model Inpu	nt File
Driver Generator	Load Executable File:	D:/UQ-PyL/UQ/test_functions/S	GAC/mopexcal.exe		Choose Executabl	e File
	Generate Driver					
	Genera	ate Python driver file				

Step 2: Generate driver file

- Click "Choose Parameter File" to load model parameter file, for SAC model is "UQ-PyL/UQ/test_functions/params/SAC.txt";
- ♦ Click "Choose Model Input File" to load model configuration file, for SAC model is "UQ-PyL/UQ/test_functions/SAC/ps_test01.sac";
- ♦ Click "Choose Executable File" to load model executable file, for SAC model is "UQ-PyL/UQ/test functions/SAC/mopexcal.exe";
- ♦ Click "Generate Driver" button to acquire model driver file.

The driver file (UQ-PyL/UQ/test_functions/SAC.py) shows below:

```
appInputFiles = "ps test01.sac"
appInputTmplts = appInputFiles + ".Tmplt"
****
# FUNCTION: GENERATE MODEL INPUT FILE
#______
def
genAppInputFile (inputData, appTmpltFile, appInputFile, nInputs, inputName
s):
  infile = open(appTmpltFile, "r")
  outfile = open(appInputFile, "w")
  while 1:
     lineIn = infile.readline()
     if lineIn == "":
       break
     lineLen = len(lineIn)
     newLine = lineIn
     if nInputs > 0:
       for fInd in range(nInputs):
          strLen = len(inputNames[fInd])
          sInd = string.find(newLine, inputNames[fInd])
          if sInd >= 0:
             sdata = '%7.3f' % inputData[fInd]
             strdata = str(sdata)
             next = sInd + strLen
             lineTemp = newLine[0:sInd] + strdata + " " +
newLine[next:lineLen+1]
             newLine = lineTemp
             lineLen = len(newLine)
     outfile.write(newLine)
  infile.close()
  outfile.close()
  return
*****
# FUNCTION: RUN MODEL
#______
def runApplication():
  sysComm = "mopexcal.exe"
  os.system(sysComm)
  return
****
```

```
# FUNCTION: CALCULATE DESIRE OUTPUT
```

```
#_____
def getOutput():
  Qe = []
  Qo = []
  functn = 0.0
  ignore = 92
  I = 0
  outfile = open("ps_test01.sac.day", "r")
  for jj in range(ignore):
     lineIn = outfile.readline()
  while 1:
     lineIn = outfile.readline()
     if lineIn == "":
        break
     nCols = string.split(lineIn)
     Qe.append(eval(nCols[4]))
     Qo.append(eval(nCols[5]))
     functn = functn + (Qe[I] - Qo[I]) * (Qe[I] - Qo[I])
     I=I+1
  outfile.close()
  functn = functn/I
  functn = math.sqrt(functn)
  return functn
*****************
# MAIN PROGRAM
def evaluate(values):
  pf = read param file(controlFileName)
  for n in range(pf['num vars']):
     pf['names'][n] = 'UQ ' + pf['names'][n]
  Y = np.empty([values.shape[0]])
  os.chdir('D:/UQ-PyL/UQ/test functions/SAC')
  for i, row in enumerate(values):
     inputData = values[i]
genAppInputFile(inputData,appInputTmplts,appInputFiles,pf['num vars']
,pf['names'])
     runApplication()
     Y[i] = getOutput()
```

```
print "Job ID " + str(i+1)
return Y
```

4.2.2 Design of Experiment

We do Design of Experiment for SAC-SMA model:

😫 UQ-PyL Uncertainty Quantification Python Laboratory – 🗖	×
File About	
Problem Definition Design of Experiment Uncertainty Analysis Sensitivity Analysis Surrogate Modelling Opti	mi 2at
Choose Parameter File: D:/VQ-PyL/UQ/test_functions/params/SAC.txt Choose Parameter File	e
Choose Model File: D:/UQ-PyL/UQ/test_functions/SAC.py Choose Model File	
Design of Experiment Method Step1: Load parameter file and driver file	
Choose DoE method: Morris One at A Time	•
Morries One At A Time(MDAT) Configuration	
**Number of total sample points = (dimension+1) * Number of Trajectories	
Number of Trajectories: 20	
Generate DoB Script	
Execute DoE Script	
Step2: Choose Design of Experiment method and generate results	
Show Design of Experiment Result	
Choose Result File: Choose Result File	e
Display Result	

Step 1: Define parameter and model information

- ♦ Choose "Design of Experiment" tab;
- Load parameter file "UQ-PyL/UQ/test_functions/params/SAC.txt" and model file "UQ-PyL/UQ/test_functions/SAC.py" (for SAC model, it's the model driver file generated before).

Step 2: Choose DoE method and run the results

- Choose DoE method "Morris One at A Time" and set "Number of Trajectories" = 20;
- ♦ Click "Generate DoE Script" button and "Execute DoE Script" button to acquire DoE results.

UQ-PyL gives the tabular and graphic results:





This step can also implemented using python script:

Python script file (SAC_DoE.py)

```
# Optional - turn off bytecode (.pyc files)
import sys
sys.dont write bytecode = True
from UQ.DoE import morris oat
from UQ.test functions import SAC
from UQ.util import scale samples general, read param file, discrepancy
import numpy as np
import random as rd
# Set random seed (does not affect quasi-random Sobol sampling)
seed = 1
np.random.seed(seed)
rd.seed(seed)
# Read the parameter range file and generate samples
param file = './UQ/test functions/params/SAC.txt'
pf = read_param_file(param_file)
# Generate samples (choose method here)
param values = morris oat.sample(20, pf['num vars'], num levels = 10,
```

```
# Samples are given in range [0, 1] by default. Rescale them to your
parameter bounds.
scale_samples_general(param_values, pf['bounds'])
np.savetxt('Input_Sobol\'.txt', param_values, delimiter=' ')
# Run the "model" and save the output in a text file
# This will happen offline for external models
Y = SAC.evaluate(param_values)
np.savetxt("Output Sobol\'.txt", Y, delimiter=' ')
```

4.2.3 Sensitivity Analysis

grid jump = 5)

Then, we do sensitivity analysis for 13 parameters of SAC-SMA model:

📓 UQ	-PyL Uncertainty Quantification Python Laboratory	_ 🗆 🗙
File About		
Definition Design of Experiment Un Perform Design of Experiment	certainty Analysis Sensitivity Analysis Surrogate Modellin	ng Optimization 🖣 🕨
Load parameter file: D:/UQ-FyL/UQ/test_fun Load Model File: D:/UQ-FyL/UQ/test_fun	stions/parems/SAC.txt stions/SAC.py	Choose Parameter File Choose Model File
Design of Experiment Method	o1: Load parameter file and driver file	
Choose DoE method: Morris One at A Time		•
Morries One At A Tim **Number of tota Number of Trajec	e(MOAT) Configuration 1 sample points = (dimension+1) * Number of Trajectories tories: 50	T
	Generate DoE Script Execute DoE Script	
Choose Analysis Method	2: Load Design of Experiment results	
Load parameter file:	D:/UQ-FyL/UQ/test_functions/params/SAC.txt	Choose Parameter File
Load data file (input file, output file)	: D:/WQ-FyL/VQ/test_functions/SAC/semple_output_morris_2015_05_19_21_34_26.txt D:/WQ-FyL/VQ/test_functions/SAC/model_output_morris_2015_05_19_21_34_26.txt	Choose Input File Choose Output File
Sensitivity Analysis Method:	Morris V	Show Kesults
Step3: (Choose sensitivity analysis method and show results	
🔄 🎸 🚺 📷 💽		

Step 1: Define parameter and model information

- ♦ Choose "Sensitivity Analysis" tab;
- Load parameter file "UQ-PyL/UQ/test_functions/params/SAC.txt" and model file (driver file) "UQ-PyL/UQ/test_functions/SAC.py".

Step 2: Load DoE results

↓ Load DoE results, sample input file
 "UQ-PyL/UQ/test_functions/SAC/sample_output_morris_2015_05_19_21_34_2
 6.txt" and model output file
 "UQ-PyL/UQ/test_functions/SAC/model_output_morris_2015_05_19_21_34_26
 .txt".

Step 3: Choose sensitivity analysis method and show results

♦ Choose sensitivity analysis method "Morris" and click "Show Results" button to acquire sensitivity analysis results.

UQ-PyL gives the tabular and graphic results:



This step can also implemented using python script:

Python script file (SAC_SA.py)

```
# Optional - turn off bytecode (.pyc files)
import sys
sys.dont_write_bytecode = True
from UQ.DoE import morris_oat
from UQ.analyze import *
from UQ.test_functions import SAC
from UQ.util import scale_samples_general, read_param_file
import numpy as np
import random as rd
# Set random seed (does not affect quasi-random Sobol sampling)
seed = 1
np.random.seed(seed)
```

```
rd.seed(seed)
```

```
# Read the parameter range file and generate samples
param file = './UQ/test functions/params/SAC.txt'
pf = read param file(param file)
# Generate samples (choose method here)
param_values = morris_oat.sample(20, pf['num_vars'], num_levels = 10,
grid jump = 5)
# Samples are given in range [0, 1] by default. Rescale them to your
parameter bounds.
scale_samples_general(param_values, pf['bounds'])
np.savetxt('Input SAC.txt', param values, delimiter=' ')
# Run the "model" and save the output in a text file
# This will happen offline for external models
Y = Sobol G.evaluate(param_values)
np.savetxt("Output SAC.txt", Y, delimiter=' ')
# Perform the sensitivity analysis/uncertainty analysis using the model
output
# Specify which column of the output file to analyze (zero-indexed)
morris.analyze(param_file, 'Input_SAC.txt', 'Output_SAC.txt', column =
```

0)

4.2.4 Surrogate Modeling

UQ	PyL Uncertainty Quantification Python Laboratory	- 🗆 ×
File About Problem Definition Design of Experim Perform Design of Experiment	ent Uncertainty Analysis Sensitivity Analysis Surrogate	9 Modelling Optimilat
Load parameter file: D:/UQ-FyL/UQ/tes	t_functions/params/SAC.txt	Choose Parameter File
Load Model File: D:/UQ-FyL/UQ/tes	t_functions/SAC.py	Choose Model File
Choose DoE method: Monte Carlo	•	
Number of Sample Points: 200	•	
	Step1: Load parameter file a	nd driver file
Generate DoE Script		
Execute DoE Script		
	Step2: Load Design of Experiment results	
Choose Analysis Method		
Load parameter file:	D:/UQ-PyL/UQ/test_functions/params/SAC.txt	Choose Parameter File
Load data file (input file, output file)	D:/VQ-FyL/VQ/test_functions/SAC/sample_output_mc_2015_05_19_21_45_26.txt	Choose Input File
	D:/VQ-PyL/VQ/test_functions/SAC/model_output_mc_2015_05_19_21_45_26.txt	Choose Output File
Surrogate Model Method:	SVM 👻	Show Results
	Step3: Choose surrogate modeling method and show	results

Step 1: Define parameter and model information

- ♦ Choose "Surrogate Modeling" tab;
- Load parameter file "UQ-PyL/UQ/test_functions/params/SAC.txt" and model file (driver file) "UQ-PyL/UQ/test_functions/SAC.py".

Step 2: Load DoE results for surrogate modeling

- ♦ Choose DoE results, sample input file
 - "UQ-PyL/UQ/test_functions/SAC/sample_output_mc_2015_05_19_21_45_26.tx t" and model output file

```
"UQ-PyL/UQ/test_functions/SAC/model_output_mc_2015_05_19_21_45_26.txt".
```

Step 3: Choose surrogate modeling method and show results

- ♦ Choose surrogate modeling method "SVM";
- ♦ Click "Show Results" button to acquire surrogate modeling results.

UQ-PyL gives the tabular and graphic results:

W	Figure 1		×
1 🖓 🔇) 🕂 💣 🔚 🗸		
0.18	Cross validation results of SVM regression model		
0.16	•	-	
0.14		-	
0.12 U			
0.10 Square		-	
шеал Wean		-	
0.06		-	
0.04		-	
0.02			
	Support Vector Machine Regression		

This step can also implemented using python script:

Python script file (SAC_Surrogate.py)

```
# Optional - turn off bytecode (.pyc files)
import sys
sys.dont write bytecode = True
from UQ.DoE import monte carlo
from UQ.test functions import SAC
from UQ.util import scale_samples_general, read_param_file, discrepancy
import numpy as np
import random as rd
# Set random seed (does not affect quasi-random Sobol sampling)
seed = 1
np.random.seed(seed)
rd.seed(seed)
# Read the parameter range file and generate samples
param_file = './UQ/test_functions/params/SAC.txt'
pf = read_param_file(param_file)
# Generate samples (choose method here)
param values = monte carlo.sample(500, pf['num vars'])
```

```
# Samples are given in range [0, 1] by default. Rescale them to your
parameter bounds.
scale_samples_general(param_values, pf['bounds'])
np.savetxt('Input_SAC.txt', param_values, delimiter=' ')
# Run the "model" and save the output in a text file
# This will happen offline for external models
Y = SAC.evaluate(param_values)
np.savetxt("Output_SAC.txt", Y, delimiter=' ')
# Perform regression analysis using the model output
# Specify which column of the output file to analyze (zero-indexed)
```

model = SVR.regression('Input_SAC', 'Output_SAC', column = 0, cv = True)

	UQ-PyL Uncertainty Quantification Python Laboratory	
e About Definition Design of Experiment Load Data	t Uncertainty Analysis Sensitivity Analysis Surro	gate Modelling Optimization
Load Parameter File: D:/UQ-FyL/UQ/tes Load Model: D:/UQ-FyL/UQ/tes	_functions/params/SAC.txt _functions/SAC.py	Choose Parameter File Choose Model File
Choose Ontimization Method	Step1: Load parameter file and model o	driver
Optimization Method:	Shuffled Complex Evolution	▼ Show Results
Show Optimization Results	Step2: Choose optimization method and	show results
s 🔝 📷		

4.2.5 Parameter Optimization

Step 1: Define parameter and model information

- \diamond Choose "Optimization" tab;
- Load parameter file "UQ-PyL/UQ/test_functions/params/SAC.txt" and model file (driver file) "UQ-PyL/UQ/test_functions/SAC.py".

Step 2: Choose optimization method and show results

♦ Choose optimization method "Shuffled Complex Evolution" and click "Show Results" button to acquire optimization results.

UQ-PyL gives the tabular and graphic results:



This step can also implemented using python script:

Python script file (SAC_Optimization.py)

```
# Optional - turn off bytecode (.pyc files)
import sys
sys.dont_write_bytecode = True
import shutil
```

```
from UQ.optimization import SCE
from UQ.util import scale_samples_general, read_param_file, discrepancy
import numpy as np
import random as rd
# Read the parameter range file
param_file = './UQ/test_functions/params/SAC.txt'
bl=np.empty(0)
bu=np.empty(0)
pf = read_param_file(param_file)
for i, b in enumerate(pf['bounds']):
    bl = np.append(bl, b[0])
    bu = np.append(bu, b[1])
dir = './UQ/test_functions/'
shutil.copy(dir+'SAC.py', dir+'functn.py')
# Run SCE-UA optimization algorithm
```

```
SCE.sceua(bl, bu, ngs=2)
```