An Example to Run Quadrature-Based Uncertainty Quantification Package for MFIX

X. Hu
xhu@iastate.edu
Department of Chemical and Biological Engineering
Iowa State University
Ames, IA, USA

A. Passalacqua
albertop@iastate.edu
Department of Mechanical Engineering
Iowa State University
Ames, IA, USA

R. O. Fox
rofox@iastate.edu
Department of Chemical and Biological Engineering
Iowa State University
Ames, IA, USA

December 3rd, 2013

1 Introduction

This document provides an example of using the non-intrusive quadrature-based uncertainty quantification (QBUQ) package implemented in MFIX. The package is developed in two parts: pre-processing and post-processing modules. In the pre-processing module, the space of the uncertain input parameters is sampled first, quadrature weights and nodes are obtained, and corresponding MFIX input file mfix.dat for each sample is generated. Then a set of MFIX simulations can be performed to obtain quadrature abscissas. In the post-processing module, moments of the system response can be estimated. Then low order statistics, including mean, variance, skewness, and kurtosis can be obtained, and the probability distribution function (PDF) of the system response at the specific location can be reconstructed. In the following parts, MFIX tutorial case fluidBed1 with uncertain particle size is used as an example case to illustrate the QBUQ procedure.

2 An example to run QBUQ package

The QBUQ modules are written in python3. The example shown here is using python3 (version 3.2.3) with following packages: numpy (version 1.7.1), scipy (version 0.12.0), sympy (version 0.7.2). Simulations are run using MFIX-2013-1, and bash shell is used for shell scripts.

MFIX tutorial case fulidBed1 is used as an example case in this section. It is a 2D bubbling fluidized bed reactor with a central jet. The uncertain parameter in this example is particle diameter, distributed uniformly from 300 μm to 500 μm. To use the QBUQ package, create a new run directory, and put the following directory and files in your run directory: directory "qbuq_package", python3 files "pre_qbuq.py" and "post_qbuq.py", and bash script files "run_serial" and "extract_data". You also need to provide a basic mfix.dat file in the run directory to use for generating input mfix.dat file for each sample.

2.1 Pre-processing module

The pre-processing module samples the space of uncertain parameters (particle diameter here), generates quadrature weights and nodes, and create corresponding mfix input files. Python3 file "pre_qbuq.py" is the main script,

and corresponding functions are in subpackage "pre_processing" under package "qbuq_package". To run the module, first go to your *run directory* in command window, then type:

python3 pre_qbuq.py

The program asks you to choose number of uncertain parameters. Only univariate and bivariate problems are implemented for now. For our case, the only uncertain parameter is particle diameter, so type **1** to select "1 – Univariate problem".

Then the program asks if the distribution of the variable is known. Type **y** for yes here since the particle diameter in our example is uniformly distributed. In the next prompt, type **1** to choose "Uniform distribution".

Next the program asks how many samples you need to generate. Here we use **10** samples in the example. Then you are asked for the range of your uniformly distributed uncertain parameter. Remember, in the basic MFIX input file mfix.dat, the units system we use is "*cgs*". Therefore, our range of particle diameter is 0.03 cm to 0.05 cm. Type **0.03** as the minimum value and **0.05** as the maximum value.

So far the first step, sample the space of uncertain parameter is finished. The program says a text file containing quadrature weights and nodes are generated successfully. The name of this file is "quadrature_weights_nodes.txt", in which the first row shows quadrature weights, and the second row shows quadrature nodes.

Then the program starts to generate MFIX input files for each sample. First a case sensitive head of the run_name is asked. Then the run_name for each mfix.dat file will be this head followed by a number. In our case, type **Test**, and with increasing particle size, the run_name will be Test0 to Test9.

Next the program asks how many keywords in mfix.dat file use the value of the uncertain parameter. Please count carefully to include all keywords related to the uncertain parameter. In our case, only one keyword "*D_p0*" is related to particle diameter. Type **1** here, and type **D_p0** for the next prompt. *Note:* The keywords typed here should be exactly the same with those in the basic mfix.dat file because these keywords will be searched and their values will be replaced by the quadrature nodes. If the keywords cannot match, the values may not be replaced.

Then the program searches keywords "*run_name*" and "*D_p0*" in the basic mfix.dat file and replaces their values with the new run_name and quadrature node value, respectively. The MFIX input file mfix.dat generated for each

sample is stored in separated directories named by their run_name. A text file named "*list_of_cases.txt*" is also generated, which contains run_name of each sample in the order of increasing quadrature nodes so that in post-processing part quadrature weights and nodes are corresponded.

Now the pre-processing part of QBUQ procedure is finished. Program says MFIX input files are generated successfully. In the run directory, you now have 2 more text files, "quadrature_weights_nodes.txt" and "list_of_cases.txt", and 10 more folders named from Test0 to Test9, each one containing a MFIX input file mfix.dat. The only difference of these input files is the value of *D_p0*. Next is how to use the shell script "run_serial" to run these simulations in serial.

2.2 Running simulations in serial

A bash script "run_serial" is written to run generated MFIX simulations in serial. In this example, when compile MFIX, serial mode of execution for each sample is selected. To run MFIX in DMP or SMP mode, please change commands in this file accordingly. If you want to submit your jobs on a cluster through a batch queue system, please consult with your system administrator.

To use the script "run_serial", in the run directory, type:

bash run_serial MFIX_model_directory

Here the "MFIX_model_directory" is the path of your MFIX model directory, such as "/home/mfix/model".

Then the program prompts the initial compilation of MFIX. The options used in this initial compilation will be used for all the simulations generated in 2.1. The options used in this example are following:

- Mode of execution: [1] Serial
- Level of optimization: [3] Level 3 (most aggressive)
- Option to re-compile source files in run directory: [1] Do not force re-compilation
- Compiler selection: [1] GNU (gfortran) version 4.3 and above

After the initial compilation, the same options are used repeatedly for all simulations. From Test0 to Test9, simulations are compiled and run in the order listed in text file "list_of_cases.txt". The screen outputs of compilation and

run for each simulation are redirected to log files "compile.log" and "run.log", respectively. Once all the simulations are completed, time-averaged quantities can be extracted, and the post-processing module of QBUQ can be used.

2.3 Post-processing module

The post-processing module first extracts time-averaged quantities with a bash shell script "extract_data" by calling MFIX post-processing program "post_mfix". Then time-averaged results can be used for QBUQ post-processing module "post_qbuq.py". The module can perform 2 tasks: estimate moments and low order statistics (mean, variance, skewness, and kurtosis), and reconstruct the PDF of system response. Functions used in this module are in subpackage "post_processing" under package "qbuq_package".

2.3.1 Extracting time-averaged data

To extract time-averaged quantities, a basic text file need to be provided in the run directory to use as the input file for MFIX post-processing program "post_mfix". Then the bash shell script "extract_data" will replace the run_name automatically and extract the data for each simulation. In this example, 2 basic text files are provided in the run directory: "post_all.txt" extracts time-averaged quantities on the whole computational domain which will be used for estimation of moments and low order statistics, and "post_point.txt" extracts data at one specific location which will be used for reconstruction of the PDF. The following parts show what conditions are used in these two post_mfix input files for MFIX post-processing program . *Note:* When name the file stored the extracted data, a .txt file extension must be added. Only .txt files can be read by the post_qbuq module.

post_all.txt

- RUN_NAME to post_procss > Test (*Note:* This name will be replaced automatically with the run_name of each simulation.)
- Enter menu selection > 1 - Examine/print data
- Write output using user-supplied precision? (T/F) > f
- Time:(0.000, 0.000) > 0, 2
- Time average? (N) > y

- Variable: (EP_g) > EP_g
- I range: (1, 1) > 1, 9
- Average or sum over I? (N) > n
- J range: (1, 1) > 1, 102
- Average or sum over J? (N) > n
- K range: (1, 1) > 1, 1
- File: (*) > EP_g_all.txt (*Note:* Must use .txt file here. The post_qbuq module only reads .txt files.)
- Time:(0.000, 0.000) > -1
- Enter menu selection > 0 - Exit POST_MFIX

Only gas volume fraction "*EP_g*" is extracted on the whole computational domain as an example. Other time-averaged quantities can be extracted by changing or adding conditions in the file accordingly.

post_point.txt

- RUN_NAME to post_procss > Test
- Enter menu selection > 1 - Examine/print data
- Write output using user-supplied precision? (T/F) > f
- Time:(0.000, 0.000) > 0, 2
- Time average? (N) > y
- Variable: (EP_g) > EP_g
- I range: (1, 1) > 7, 7
- J range: (1, 1) > 51, 51
- K range: (1, 1) > 1, 1
- File: (*) > EP_g.txt (*Note:* Must use .txt file here. The post_qbuq module only reads .txt files.)
- Time:(0.000, 0.000) > 0, 2
- Time average? (N) > y
- Variable: (EP_g) > P_g

- I range: (1, 1) > 7, 7
- J range: (1, 1) > 51, 51
- K range: (1, 1) > 1, 1
- File: (*) > P_g.txt
- Time:(0.000, 0.000) > 0, 2
- Time average? (N) > y
- Variable: (EP_g) > V_s
- I range: (1, 1) > 7, 7
- J range: (1, 1) > 51, 51
- K range: (1, 1) > 1, 1
- File: (*) > V_s.txt
- Time:(0.000, 0.000) > -1
- Enter menu selection > 0 - Exit POST_MFIX

Gas volume fraction "*EP_g*", gas pressure "*P_g*" and vertical solid velocity "*V_s*" are extracted at a specific location. Conditions in the file can be changed or added accordingly.

To use the bash shell script "extract_data" to extract time-averaged data for all simulations, in the run directory, type:

bash extract_data post_file post_mfix_directory

Here the "post_file" is the file used as the input for MFIX post-processing program "post_mfix". In this example, it can be either "post_all.txt" or "post_point.txt". The "post_mfix_directory" is the path of your MFIX post_mfix directory, such as "/home/mfix/post_mfix". We run the script twice using both "post_all.txt" and "post_point.txt" to extract gas volume fraction "*EP_g*" on the whole computational domain, and gas volume fraction "*EP_g*", gas pressure "*P_g*" and vertical solid velocity "*V_s*" at a specific location for all 10 simulations. Once the process is completed, these data can be used for post_qbuq module.

2.3.2 Estimation of moments and low order statistics

Two tasks can be performed by the QBUQ post-processing module, estimation of moments and low order statistics, and reconstruction of the PDF of system response. Here first shows estimation of moments and low order statistics. To run the module, in the run directory, type:

python3 post_qbuq.py

The program asks you to select your job first. Type **1** to to choose "1 – Calculate low order statistics".

Then the program asks for the file name of the quantity of interest. In this example, it is the file name used in file "post_all.txt", "EP_g_all.txt". So type **EP_g_all.txt** to continue. Next the program asks how many samples we have. Type **10** for 10 samples in this example.

Then we are asked for the highest order of moments. In order to estimate low order statistics including mean, variance, skewness, and kurtosis, at least fourth order moment is needed. So type **4** here to continue.

Next the program uses the time-averaged data of each simulation extracted in 2.3.1 as quadrature abscissas and the quadrature weights generated in 2.1 to estimate up to fourth order moments of gas volume fraction "*EP_g*" on the whole computational domain and calculate mean, variance, skewness, and kurtosis as well. Runtime warning maybe displayed here because when calculate skewness and kurtosis, the standard deviation as the denominator may become 0 at those locations, which results in "NaN" or "Inf" for skewness and kurtosis. These warning do not effect the results at other locations.

Once the program is completed, 5 more text files are generated in the run directory: "moments.txt", "mean.txt", "variance.txt", "skewness.txt", and "kurtosis.txt". Make sure these files are stored and renamed correctly before using the post_qbuq module again. These files will be rewritten when the module is used for other quantities of interest.

2.3.3 Reconstruction of the PDF

The other task the QBUQ post-processing module can perform, reconstruction of the PDF of system response, is shown here. PDF of three quantities, gas volume fraction "*EP_g*", gas pressure "*P_g*" and vertical solid velocity "*V_s*", at a specific location are reconstructed using extended quadrature

method of moments (EQMOM)[1, 3]. The following parts describe the reconstruction of the PDF of these three quantities one by one.

2.3.3.1 Gas volume fraction "*EP_g*"

Type the following command to run the QBUQ post-processing module.

python3 post_qbuq.py

Type **2** to select "2 – Reconstruction the probability distribution function". Then enter **1** to choose β-EQMOM for *EP_g*.

Then the program asks for the number of nodes to use. Maximum 5 nodes can be used. Type **4** for our example. Next the program asks for 4 ratios of minimum weights to maximum weights (rmin), which are parameters used in adaptive Wheeler algorithm[2]. For $rmin(0)$, type **0**; type $\mathbf{1e-12}$ for $rmin(1)$; for $rmin(2)$, enter $\mathbf{1e-8}$; enter $\mathbf{1e-4}$ for $rmin(3)$. In the next prompt, the program asks to enter the minimum distance between distinct nodes (eabs), which is also a parameter used in adaptive Wheeler algorithm[2]. Enter $\mathbf{1e-10}$ here.

Next type **10** for we have in total 10 simulations in this example. Then for the file name of quantity of interest, type **EP_g.txt**.

In β-EQMOM, a bounded interval is needed. In the post_qbuq module, the bounded interval can either use the minimum to maximum value of the data or be set up manually. In this example, type **1** to use minimum to maximum value of the data as the interval.

Then the program uses β-EQMOM to reconstruct the PDF of gas volume fraction. Once it is completed, information of σ, weights, nodes on $[0, 1]$, and nodes on bounded interval is shown on screen. Three more text files are generated in the run directory. File "betaEQMOM_sigma.txt" keeps the value of σ. In file "betaEQMOM_weights_nodes.txt", first row shows weights of each EQMOM node, second row gives value of each EQMOM node on $[0, 1]$, and the third row shows the value of each EQMOM node on the bounded interval. File "data_set_for_betaEQMOM.txt" contains 10 values used to reconstruct the gas volume fraction, each of which is the value of gas volume fraction of that simulation at the specific location.

For your reference, Table 1 gives the results shown on the author's screen.

sigma	0.00523497997459	
weights	0.46426887803	0.284369847782
	0.217899463915	0.033461810273
nodes	0.00956478087672	0.0894838372351
on $[0, 1]$	0.629186940922	0.999750406862
nodes	0.415734483248	0.425965720843
on $[a, b]$	0.495058512177	0.542498047086

Table 1: Reconstruction results of gas volume fraction using β-EQMOM.

2.3.3.2 Gas pressure "*P_g*"

The procedure of using post_qbuq module to reconstruct the PDF of gas pressure P_g at the location is similar to the procedure used in 2.3.3.1 except that some different options are selected. Conditions used in the module to reconstruct the PDF of gas pressure are as follows.

- Type **python3 post_qbuq.py** to run the module.
- Type **2** to select reconstruction of the PDF.
- Type **2** to use γ-EQMOM to reconstruct the PDF of gas pressure.
- Enter **5** as the number of EQMOM nodes.
- For the parameters rmin in adaptive Wheeler algorithm, the first four are the same with 2.3.3.1. Type $\mathbf{1e-4}$ for the fifth one $rmin[4]$.
- Use $\mathbf{1e-10}$ again as eabs in adaptive Wheeler algorithm.
- Enter **10** for 10 samples.
- The file name of gas pressure in this example is **P_g.txt**.
- Select **2** to set the lower bound for γ-EQMOM manually.
- Type **200** as the lower bound.

Once the reconstruction is completed, 3 more files are generated in the run directory: "gammaEQMOM_sigma.txt", "gammaEQMOM_weights_nodes.txt", and "data_set_for_gammaEQMOM.txt". Table 2 gives results on author's screen for your reference.

2.3.3.3 Vertical solid velocity "*V_s*"

sigma	19.606992775		
weights	0.177257589525	0.27693370017	0.181902294417
	0.150325247342	0.213581168547	
nodes	283.874714748	1536.18411219	3175.10056231
on $[0, \infty)$	4271.74179277	6774.60334977	
nodes	483.874714748	1736.18411219	3375.10056231
on $[a, \infty)$	4471.74179277	6974.60334977	

Table 2: Reconstruction results of gas pressure using γ-EQMOM.

The procedure of using post_qbuq module to reconstruct the PDF of vertical solid velocity *V_s* at the location is slightly different from the procedure used in 2.3.3.1 and 2.3.3.2, since 2-node Gaussian EQMOM is used[1]. The procedure is as follows.

- Type **python3 post_qbuq.py** to run the module.
- Type **2** to select reconstruction of the PDF.
- Type **3** to use 2-node Gaussian EQMOM to reconstruct the PDF of vertical solid velocity.
- Enter **10** for 10 samples.
- The file name of vertical solid velocity in this example is **V_s.txt**.

Three more files are generated in the run directory once the reconstruction is completed: "GaussianEQMOM_sigma.txt", "GaussianEQMOM_weights_nodes.txt", and "data_set_for_GaussianEQMOM.txt". Table 3 gives results on author's screen for your reference.

sigma	1.93623130711	
weights	0.694653795117	0.305346204883
nodes	-5.1032997211	3.10647288573

Table 3: Reconstruction results of vertical solid velocity using 2-node Gaussian EQMOM.

Now this document gives a complete example of using QBUQ package for a MFIX case with one uncertain parameter.

References

[1] C. Chalons, R.O. Fox, and M. Massot. A multi-Gaussian quadrature method of moments for gas-particle flows in a LES framework. In *Proceedings of the Summer Program*, pages 347 – 358. Center for Turbulence Research, Stanford University, 2010.

[2] C. Yuan and R. O. Fox. Conditional quadrature method of moments for kinetic equations. *Journal of Computational Physics*, 230(22):8216–8246, 2011.

[3] C. Yuan, F. Laurent, and R. O. Fox. An extended quadrature method of moments for population balance equations. *Journal of Aerosol Science*, 51:1–23, 2012.