JUPYTER NOTEBOOKS FOR COMPUTER-BASED LABORATORIES ON POWER SYSTEM DYNAMICS AND CONTROL

Federico Milano and Guðrún Margrét Jónsdóttir

School of Electrical, Electronic and Communications Engineering
University College Dublin (IRELAND)
federico.milano@ucd.ie, gudrun.jonsdottir@ucdconnect.ie

Abstract

The paper describes the author's experience with laboratory activities based on Jupyter Notebook for the module “Power System Dynamics and Control” in the academic year 2017/2018 at University College Dublin. Jupyter is an open-source project that started in 2014 based on IPython and has quickly become popular in the scientific and academic community. Jupyter Notebooks are basically interactive webpages, that can be easily and efficiently designed for testing live code, embedding narrative text and visualizing results. The paper shows that using Jupyter Notebooks for teaching can improve the students learning experience and minimize the need to know how to interact with a Unix server and, thereby effectively increasing the ability of the students to run more tests and simulations. An example of the lab activities proposed in the module as well as the point of view of the teaching assistant that helped run the laboratories and students' feedback are provided in the paper.

Keywords: power system dynamics, power system control, simulation, Jupyter Notebook, bash shell, computer-based laboratory.

1 INTRODUCTION

The first author is the developer of a Python-based software suite, called Dome, for power system modelling and dynamic analysis. Dome has been utilized by the first author for research since 2009 and for education from 2013 to date at University College Dublin. The advantages and drawbacks of Dome for the laboratory activities of modules on power system modelling, control and stability analysis are discussed in [1-5]. The software package itself is outlined in [6]. Dome is an “old-style” Unix programme, which does not provide any graphical interface and has to be run from the command-line through a not very user-friendly Unix terminal.

While in [6], the didactic benefits of using Unix terminals are thoroughly discussed, it is a fact that the current generation of students is not used to terminals and the learning time to master such a tool requires a significant part of the laboratory activities. At UCD, the agenda of undergraduate students of the Electric Energy Systems programme is more packed than ever, with many lab activities, internships and workshops. Any tool that allows reducing the learning curve time and helps student focus on the specific matter of each module is thus to be welcome.

Jupyter Notebook is an open-source project that started in 2014 based on IPython and has quickly become popular in the scientific and academic community [7]. While IPython was limited to the utilization of the Python language, Jupyter Notebook can leverage several different programming and scripting languages, including the bash shell, which is the interface utilized in the laboratories discussed in this paper. A notebook is basically an interactive webpage, that can be easily and efficiently designed for testing live code, embedding narrative text and visualizing results. The term “computational narrative” has been coined for these virtual notebooks and summarizes well the feature of Jupyter Notebook.

The contributions of the paper are as follows.

- A description of the key features of Jupyter Notebook for the interaction with the software tool Dome and the preparation of computer-based laboratory activities on power system dynamics and control.
- A complete example of a laboratory activity that focuses on the control of renewable resources, frequency-controlled loads and energy storage devices.

- The experience and comments of the teaching assistant that set up and run the laboratories of the module “Power System Dynamics and Control”.
- The feedback of the undergraduate students that attended the module “Power System Dynamics and Control” in the academic year 2017/18.

The paper is organized as follows. Section 2 briefly describes the software tools, namely Dome and Jupyter Notebook, that have been used in all labs. The interface between Dome and Jupyter Notebook is also discussed in this section. Section 3 outlines the modules and the laboratory activities that have been carried out. Sections 4 and 5 presents the teaching assistant experience and students' feedback on the labs, respectively. Finally, Section 6 draws relevant conclusions and suggests future work directions.

2 OUTLINES OF THE SOFTWARE TOOLS

The software tools considered in this section are two: Dome and Jupyter Notebook. Dome is a Python-based command-line based software developed by the first author in the last 10 years for power system modelling, stability analysis and control prototyping. This tool is currently utilized by the first author and all his collaborators [6]. Dome is also the software tool utilized in the lab activities of the Power Systems Dynamics and Control module [1-5]. Jupyter Notebook is an interactive, web-based interface based on Python programming language that has become very popular for teaching and laboratory activities [7].

Here below we first briefly describe each tool and then discuss their integration and the main concepts on which the laboratories activities have been designed.

2.1 Dome

Dome provides about 50 parsers for input data, including most popular power system formats such as PSS/E and GE and SIMPOW formats; more than 900 devices ranging from standard power flow models, synchronous machines, AVRs and other basic controllers to a variety of wind turbines, energy storage devices and distributed energy sources; 10 analysis tools including standard power flow analysis as well as three-phase unbalanced power flow, continuation power flow, OPF, time domain simulation, electromagnetic transients, eigenvalue analysis, short circuit analysis, equivalencing procedures and load admission control strategies for smart grids; and 10 output formats, including LaTeX, Excel, and 2D and 3D visualization tools.

Despite the vastness of the tools and models provided, Dome remains fundamentally a light tool. Thanks to “modularity” and “laziness” (see [6] for details), only needed devices and routines are loaded at run-time. These are generally about 1% of Dome modules. Hence, the project can grow without affecting performance. Modularity and laziness has also the advantage of allowing parallel development of new modules: if a beta-version of a new function is broken, all users that are not using that function can continue using Dome uninterrupted. Moreover, no forking is necessary as different versions of the same module can coexist.

The key aspect of Dome that can be exploited for educational purposes is the possibility of using Dome with different levels of expertise: undergraduate, master, Ph.D. and experienced researcher. At the undergraduate level, no knowledge of the internal functioning of the software is required. However, since Dome does not have a user interface, students still have to learn basic Unix commands and the interaction with a Unix server (e.g., ssh and sftp clients). In the experience of the authors, this can take up a significant part of the time students dedicate to the lab activities. The web-based interface provided by Jupyter Notebook can help reduce this time, as discussed below.

2.2 Jupyter Notebook

According to the main webpage [7], “*Jupyter Notebook, is an open-source web application that allows you to create and share documents that contain live code, equations, visualization and narrative text.”*

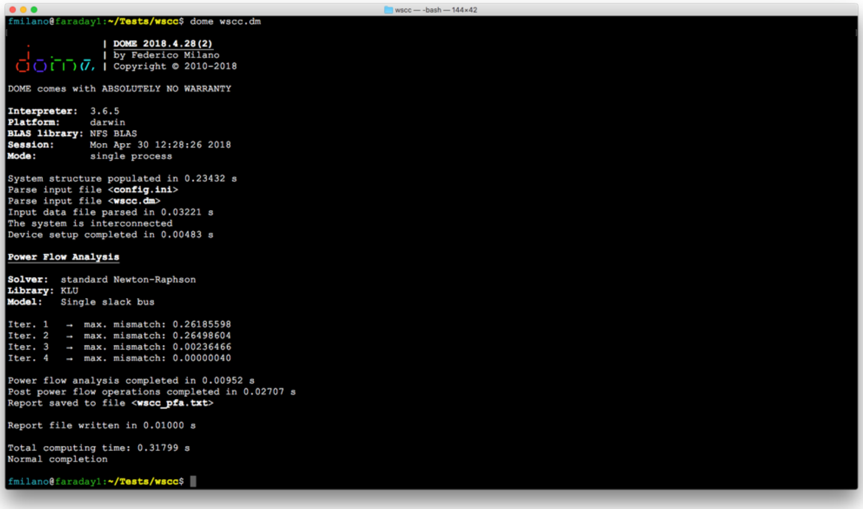
The installation of this software tool basically requires only the Python language, the Apache web server and a few Python packages which come with most important Linux distributions. The authors have installed Jupyter Notebook on one of their Linux servers running Fedora 25.

The simplicity of Jupyter makes it an ideal tool for teaching and, in fact, the number of successful stories of the deployment of such a tool in academia have become very numerous in recent years. One of the key features is the fact that a Jupyter Notebook is relatively easy to extend as it is based on the Python language. So much so that there are entire conferences dedicated to Jupyter applications and developments (e.g., JupyterCon 2018 will be held in New York, in August 2018).

The main idea behind this tool is that one can mix text and equations with live code which can be changed and executed interactively by the user. By default, such a code is Python, but there are many extensions that allow utilizing other popular languages, such as Java and Julia. The Unix Bash shell script language has been recently added to the list of supported environments. This is the key feature for the of Jupyter Notebook with Dome, as it is discussed in the following section.

2.3 Integration of Dome within the Jupyter Environment

While Dome is written in the Python language, it actually runs through a standard Unix terminal, as shown in Figure 1. Of course, Dome also can be imported as a standard Python package through an interactive session of the Python interpreter. However, this utilization requires an expert knowledge of Dome as well as the knowledge of the Python programming language. This was an obstacle that prevented early attempts to utilize Jupyter Notebooks for lab activities based on Dome.



```
fmilano@faraday1:~/Tests/wscc$ dome wscc.dm

                  | DOME 2018.4.28(2)
                  | by Federico Milano
                  | Copyright © 2010-2018

DOME comes with ABSOLUTELY NO WARRANTY

Interpreter:  3.6.5
Platform:     darwin
BLAS library: NFS BLAS
Session:      Mon Apr 30 12:28:26 2018
Mode:         single process

System structure populated in 0.23432 s
Parse input file <config.ini>
Parse input file <wscc.dm>
Input data file parsed in 0.03221 s
The system is interconnected
Device setup completed in 0.00483 s

Power Flow Analysis

Solver:  standard Newton-Raphson
Library: KLU
Model:   Single slack bus

Iter. 1   →  max. mismatch: 0.26185598
Iter. 2   →  max. mismatch: 0.26498604
Iter. 3   →  max. mismatch: 0.00236466
Iter. 4   →  max. mismatch: 0.00000040

Power flow analysis completed in 0.00952 s
Post power flow operations completed in 0.02707 s
Report saved to file <wscc_pfa.txt>

Report file written in 0.01000 s

Total computing time: 0.31799 s
Normal completion

fmilano@faraday1:~/Tests/wscc$
```

Figure 1. Standard command-line utilization of Dome from a Unix terminal.

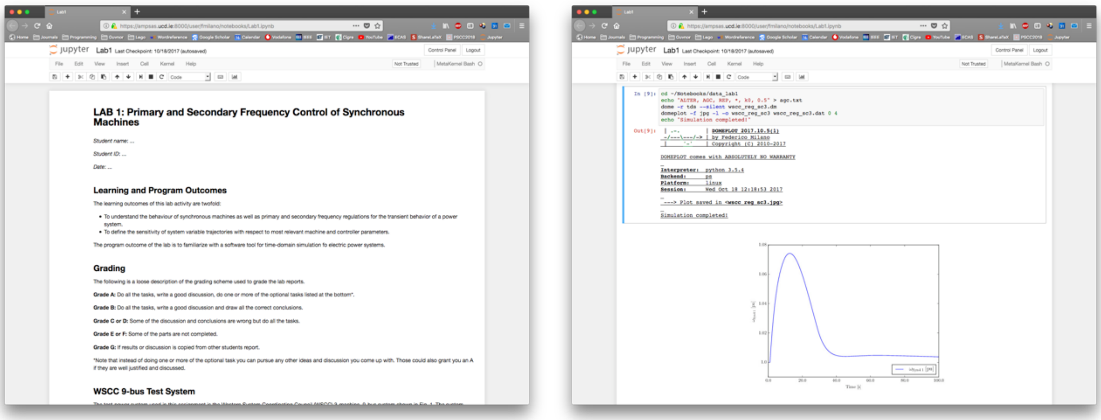
This issue has been solved by utilizing the Jupyter extension that provides support for the Bash shell scripting language and that allows a seamless integration of the notebook with Unix terminals. The steps are as follows.

- First, a folder with relevant Dome data files is created on the server. This folder will also contain Dome results and plots.
- Each data file is modified to include a custom file that modifies some of it parameters. This is possible thanks to the flexibility of the Dome data format that permits the inclusions of special macros to modify the behaviour with which Dome parses the data file itself.
- A small script is written in the Jupyter Notebook that creates “on the fly” the modifications to be included in the data files and finally runs Dome and, when required, plots relevant quantities.

An example of script included in the Jupyter Notebook is shown below:

```
In:  cd ~/Notebooks/data_lab1
     echo "ALTER, Tg1, REP, *, R, 0.05" > turb.txt
     dome -r tds --silent wscc_reg.dm
     domeplot -f jpg -l -o wscc_tg wscc_reg.dat 0 4 5 6
     echo "Simulation completed!"
```

The first line of the script moves the current shell to the relevant folder where the data files are located. The second line modifies a parameter of the file – in this case, the droops of all turbine governors of the synchronous machines. The third line calls Dome and solves the time domain simulation. The fourth line plots the rotor speeds of the machines. The last line is printed only if no errors stopped the execution of the script and serves to understand if the simulation has completed successfully. Figure 2 shows a typical look of the Jupyter Notebook and a section where the script is executed and the results shown.


Lab1
https://ampsas.ucd.ie:8000/user/fmilano/notebooks/Lab1.ipynb
jupyter
Lab1 Last Checkpoint: 10/18/2017 (autosaved)
Control Panel
Logout
File
Edit
View
Insert
Cell
Kernel
Help
Not Trusted
MetaKernel Bash
Code

LAB 1: Primary and Secondary Frequency Control of Synchronous Machines

Student name: ...

Student ID: ...

Date: ...

Learning and Program Outcomes

The learning outcomes of this lab activity are twofold:

- To understand the behaviour of synchronous machines as well as primary and secondary frequency regulations for the transient behavior of a power system.
- To define the sensitivity of system variable trajectories with respect to most relevant machine and controller parameters.

The program outcome of the lab is to familiarize with a software tool for time-domain simulation fo electric power systems.

Grading

The following is a loose description of the grading scheme used to grade the lab reports.

Grade A: Do all the tasks, write a good discussion, do one or more of the optional tasks listed at the bottom*.

Grade B: Do all the tasks, write a good discussion and draw all the correct conclusions.

Grade C or D: Some of the discussion and conclusions are wrong but do all the tasks.

Grade E or F: Some of the parts are not completed.

Grade G: If results or discussion is copied from other students report.

*Note that instead of doing one or more of the optional task you can pursue any other ideas and discussion you come up with. Those could also grant you an A if they are well justified and discussed.

WSCC 9-bus Test System

```
In [9]: cd ~/Notebooks/data_lab1
        echo "ALTER, AGC, REP, *, k0, 0.5" > agc.txt
        dome -r tds --silent wscc_reg_sc3.dm
        domeplot -f jpg -l -o wscc_reg_sc3 wscc_reg_sc3.dat 0 4
        echo "Simulation completed!"

Out[9]:  | .-.          | DOMEPLOT 2017.10.5(1)
        -/---\---/->  | by Federico Milano
         |      '-'   | Copyright (C) 2010-2017

        DOMEPLOT comes with ABSOLUTELY NO WARRANTY

        Interpreter:  python 3.5.4
        Backend:      ps
        Platform:     linux
        Session:      Wed Oct 18 12:18:53 2017

        ---> Plot saved in <wscc_reg_sc3.jpg>

        Simulation completed!
```

Figure 2. Utilization of Dome integrated into a Jupyter Notebook.

The simple script above has several advantages with respect to the execution of Dome directly on a Unix terminal. The main feature is that the students can easily test several values of the parameters and visualize the effect of such changes all from the same web page. When utilizing a remote connection to a server, results have to be transferred from the server to the client and one has to open several windows to run the simulations. In fact, one needs a terminal connected to the server where to write the commands, another terminal where to modify the data files, an SFTP client to transfer file and an image viewer to open the plots (see [2] for more details on the terminal-based utilization of Dome). Hence, in the same time, the students can solve many more simulations than utilizing the terminal.

Another advantage is the fact that the text of the lab is also part of the same webpage. Students just need to follow the flow of the notebook, run the required simulations, add any other simulations that they wish to solve, and finally fill the notebook with their comments and conclusions. Then, at the end of the lab, the students just need to save their work on a file. No report has thus to be written from scratch. This again, allows the students to have more time to solve simulations and elaborate their comments on the work done.

Since all results are saved on the server and the main documents to be evaluated are Notebook files with the same structure, the duty of the evaluator is greatly simplified as they can focus exclusively on the sections where the students were requested to add their comments.

The Appendix illustrates the discussion above through a lab activity proposed to the students.

3 MODULE CONTENTS AND LABORATORY ACTIVITIES

This section provides a brief description of the programme of the module Power Systems dynamics and control (PSDC) as thought at the 4th stage of the BE and ME programmes of Electrical Energy Systems of University College Dublin by the authors. Reference books of PSDC are [8-11].

3.1 Module Contents

PSDC introduces the main control requirements of the most important devices that compose a HV transmission system. The focus of the module is on the dynamic behaviour of power systems. This includes frequency control, voltage control and auxiliary controllers aimed to improve the stability of the network. All topics are explained both theoretically and with simulation-based examples.

The module is divided into three parts.

- Part I: Modelling and control of the synchronous machine. Dynamic model of the machine, Park transformation and per-unit equations. Primary and secondary controls including automatic voltage regulators, turbine and turbine governors, under and over-excitation limiters, and power system stabilizers. Synchronous machine secondary controls including automatic generator controllers and secondary voltage regulators.
- Part II: Transformer and FACTS device controllers, including under-load tap changers and phase shifters. VSC model and controls. Control of shunt and series FACTS devices and HVDC-links.
- Part III: Distributed energy sources control with particular emphasis on models and controllers of wind turbines and energy storage devices (MPPT, voltage control, frequency control, etc.).

The learning outcomes of the module are: basic concepts of power system frequency and voltage control; knowledge of control systems of all principal devices for high voltage transmission systems; and practical examples based on numerical simulations.

3.2 Laboratory Activities

PSDC includes 1 introductory lecture of up to 2 hours and 4 lab activities, 2 hours each. The activities are intended to develop the following skills: critical analysis of the results; ability to solve problems; ability to identify the key aspects of a certain phenomenon; and the ability to understand the response of a power system based on simulation results. The latter is a crucial skill that the student has to achieve. Based on this skill, they are able to decide whether the results that they have obtained are reasonable or not and, if not, to find out why. After each laboratory the students have to submit a report which is evaluated based on the consistency of the organization of the matter, clarity of exposition, completeness of results and correctness of conclusions.

The lab activities of PSDC are: (i) inertial response, primary and secondary frequency regulation of synchronous machines; (ii) automatic voltage regulation of synchronous machines and power system stabilizers; (iii) voltage regulation of under-load tap changers and FACTS devices (namely SVC and TCSC); and (iv) frequency control through non-synchronous devices.

In previous years, when lab activities were solved on standard Unix terminals, the PSDC modules required 4 hours of tutorials to teach the student how to use Unix terminals and Dome (see a thorough discussion in [2]). The integration of Dome within Jupyter Notebook consistently reduces the need of learning the details of both tools. In the current setup, only one 2-hour tutorial is deemed necessary. This tutorial explains the basic functioning of Jupyter notebook (few minutes) and the few Bash commands and Dome options required to run all simulations (see the example in Section 2.3).

Jupyter Notebook allows thus to cut by half the time required to prepare the students to solve the activities required in the following labs. It is important to note, however, that the learning time is reduced because the students do *not* need to learn how to use a Unix terminal but just how to run the simulations through the webpage. Thus, the students can focus exclusively on the object of the labs but learn less.

4 TEACHING ASSISTANT'S EXPERIENCE

The second author of this paper has attended the module when laboratory activities were based exclusively on Unix terminals as discussed in [2] and is currently a PhD student developing her research based on Dome. In 2017 she became the Teaching Assistant (TA) of the module PSDC and helped develop the laboratory activities based on Jupyter Notebooks. For example, the lab activity reported in Appendix A.2 was proposed and entirely designed by the second author. She is thus in the unique position of having carried out all labs using both Unix and Jupyter Notebook approaches.

Until 2017 the PSDC labs were all done on the Unix terminal. A big part of the questions the students had for the then TA in the module were related to using the Unix terminal. Even though two introductory labs were used to familiarize the students with the Unix environment, the utilization of the command line was an ongoing issue throughout the course. A big part of the time students dedicated to the lab, in fact, was dedicated to solving issues stemming from using the terminal. As a consequence, the students had less time to focus on actually grasping the concepts the lab was supposed to be teaching.

Comparing these labs to the labs in the school year 2017-18, which are based in Jupyter Notebook, where much of the coding is already laid out for the students, there has been a significant shift in the time the students dedicated to the labs and the amount of questions the students had for the TA. The amount of software support needed by the students dropped significantly allowing most of the students questioning to be related to the actual technical topics of the lab. However, the extra time the students had compared to previous year students to focus on the lab activities did not fully translate into better reports than the years before. This was mainly due to the apparent “simplicity” of the Jupyter Notebook which led less motivated students to do just the minimum effort to complete the assigned tasks.

5 STUDENTS' FEEDBACK ON LABORATORY ACTIVITIES

This section presents and discusses the feedback provided by the students regarding the utilization of Dome and Jupyter Notebooks during the laboratory activities of the module PSDC of the academic year 2017/18. The discussion below is based on direct observation during laboratory activities by the TA.

The general attitude of the students towards using the Jupyter Notebook with Dome was positive. They could work independently and got a hang of the environment quickly. Some students however had an issue with having to do the report in the Jupyter Notebook environment, as they found it limiting. Clearly, the notebooks can be customized as much as needed, but the students were reluctant to do so, at least at the beginning of the module. Another issue the more motivated students had was that they found it difficult to do more advanced simulations because they didn’t fit in with the predefined structure of the notebooks. Students that want to do more, in fact, have to learn also how utilize the command line version of Dome and customize the scripts that solve the simulations. This was just a byproduct of the approach based only on Unix terminals, but it is an additional effort when utilizing Jupyter Notebook.

To try to fix the issues above, the time in the introduction lab could be put to better use. This year the introductory lab was mostly based on previous years introduction lab on Dome. Much of that information is no longer required for the students. Taking into account the feedback from the students and their reports we would like to redefine the introduction lab to focus on more practical things related to Dome and on what is required of the students in relation to the simulations and results. Additionally, it is important to put some focus on teaching the students how to take full advantage of the Jupyter Notebook environment when it comes to writing, equations and displaying results.

6 CONCLUSIONS

The paper shows that using Jupyter Notebook for teaching can improve the students learning experience by minimizing the need to know how to interact with a Unix server and thereby effectively increasing the ability of the students to run more tests and simulations. The structure of Jupyter notebooks, however, while it does not preclude motivated students from actually taking advantage of the flexibility of the Unix terminal, requires them to pay an extra effort to master both environments. Overall, we believe that Jupyter Notebook offers a versatile solution that can seamlessly be adapted to the skills and the interests of the students. Future work will focus on improving the deep learning of the students while retaining the simplicity of the Jupyter Notebook approach.

APPENDIX

A.1 Lab on Frequency Control on Non-synchronous Machines

This appendix shows the laboratory activity on the inertia emulation and primary frequency control as provided by non-synchronous generation (wind turbines), energy storage devices and thermostatically controlled loads proposed to the students of PSDC. Typical simulation results are shown in Figures 3 and 4 at the end of Sections A.1.4 and A.1.5, respectively.

A.1.1 Learning Outcomes

The purpose of this lab activity is to get familiar with the dynamic behaviour of power systems with inclusion of non-synchronous generation and the impact on frequency dynamics of the regulators of non-synchronous devices.

- To understand the effect on steady-state and transient operation of the frequency controllers of wind turbines, energy storage devices and thermostatically controlled loads.
- To understand the different information obtained with deterministic and stochastic simulations.

- To understand future challenges for the control of low inertia power systems.

The program outcome of the lab is to familiarize with a software tool for time-domain simulation of electric power systems.

A.1.2 Grading

The following is a loose description of the grading scheme used to grade the lab reports.

Grade A: Do all the tasks, write a good discussion, do one or more of the optional tasks listed at the bottom*.

Grade B: Do all the tasks, write a good discussion and draw all the correct conclusions.

Grade C or D: Some of the discussion and conclusions are wrong but do all the tasks.

Grade E or F: Some of the parts are not completed.

Grade G: If results or discussion is copied from another student's report.

*Note that instead of doing one or more of the optional task you can pursue any other ideas and discussion you come up with. Those could also grant you an A if they are well justified and discussed.

A.1.3 Modified WSCC 9-bus Test System

The system is a modified version of the WSCC 9-bus system presented in [10]. The following changes have been made to the system:

- The capacity of the synchronous generator at Bus 2 is reduced by 100 MW.
- A wind power plant is connected to the system at Bus 7 through a two-winding transformer with the power capacity 100 MW. The wind turbine model used is a Double-Fed Induction Generator.

A.1.4 Deterministic Analysis

Consider the modified WSCC system in Fig. 2 with d-q axis machine models, AVRs and turbine governors and consider a loss of load at bus 5 occurring at *t* = 1 s. Determine the effect on frequency variations in the following scenarios.

A.1.4.1 Scenario 0

This is the base-case scenario that utilizes the data provided in [10] and that includes only conventional frequency regulation provided by the turbine governors of synchronous machines (*wscc_reg_orig.dm*).

A.1.4.2 Scenario 1

Consider the modified WSCC 9-bus system (*wscc_reg_wind.dm*). Compare the results to the results obtained for Scenario 0. Can you improve the behaviour in Scenario 1 by changing the droops of the turbine governors?

```
In:  cd ~/Notebooks/data_lab4
     echo "ALTER, Tg1, REP, *, R, 0.05" > sc1.txt
     dome --silent -r tds wscc_reg_orig.dm
     dome --silent -r tds wscc_reg_wind.dm
     domeplot --silent -f jpg -l --ylabel "$\omega_{\rm Syn4} {\rm 1\;[pu(Hz)]}$" -o
     wscc_reg_sc1     wscc_reg_original.dat     wscc_reg_original_a.lst     0     4
     wscc_reg_wind.dat wscc_reg_wind_a.lst 0 4
```

A.1.4.2 Scenario 2

Consider the modified WSCC 9-bus system with turbine governor and frequency control on the wind turbine (*wscc_reg_wind_wcon.dm*). Discuss the effect of varying the droop (*R*) of the wind frequency control (*WindFreq*).

```
In:  cd ~/Notebooks/data_lab4
     echo "ALTER, WindFreq, REP, *, R, 0.05" > sc2.txt
     dome -r tds --silent wscc_reg_wind.dm
     dome -r tds --silent wscc_reg_wind_wcon.dm
     domeplot --silent -f jpg -l -o wscc_reg_sc2 --ylabel "$\omega_{\rm Syn4} {\rm
     1\;[pu(Hz)]}$" wscc_reg_wind.dat wscc_reg_wind_a.lst 0 4 wscc_reg_wind_wcon.dat
     wscc_reg_wind_wcon_a.lst 0 4
```

A.1.4.3 Scenario 3

Consider the modified WSCC 9-bus system with turbine governors and a general storage system (*Stordyn3*) providing frequency control (*wscc_reg_wind_stcon.dm*). The storage device is located at Bus 8. Discuss the effect of varying the gains of the frequency control (K_{ppc} and K_{ipc}), the initial stored energy (E_0) and the time constant of the active power dynamics (T_p).

```
In:  cd ~/Notebooks/data_lab4
     echo "ALTER, Stordyn3, REP, 1, Kppc, 13" > sc3.txt
     echo "ALTER, Stordyn3, REP, 1, Kipc, 20" >> sc3.txt
     echo "ALTER, Stordyn3, REP, 1, E0, 2.88" >> sc3.txt
     echo "ALTER, Stordyn3, REP, 1, Tp, 0.0026" >> sc3.txt
     dome -r tds --silent wscc_reg_wind.dm
     dome -r tds --silent wscc_reg_wind_stcon.dm
     domeplot --silent -f jpg -l -o wscc_reg_sc3 --ylabel "$\omega_{\rm Syn4} {\rm
     1\;[pu(Hz)]}$"       wscc_reg_wind.dat       wscc_reg_wind_a.lst       0       4
     wscc_reg_wind_stcon.dat wscc_reg_wind_stcon_a.lst 0 4
```

A.1.4.4 Comparison

Compare Scenarios 2-3. Is any method for frequency control more preferable than the other?

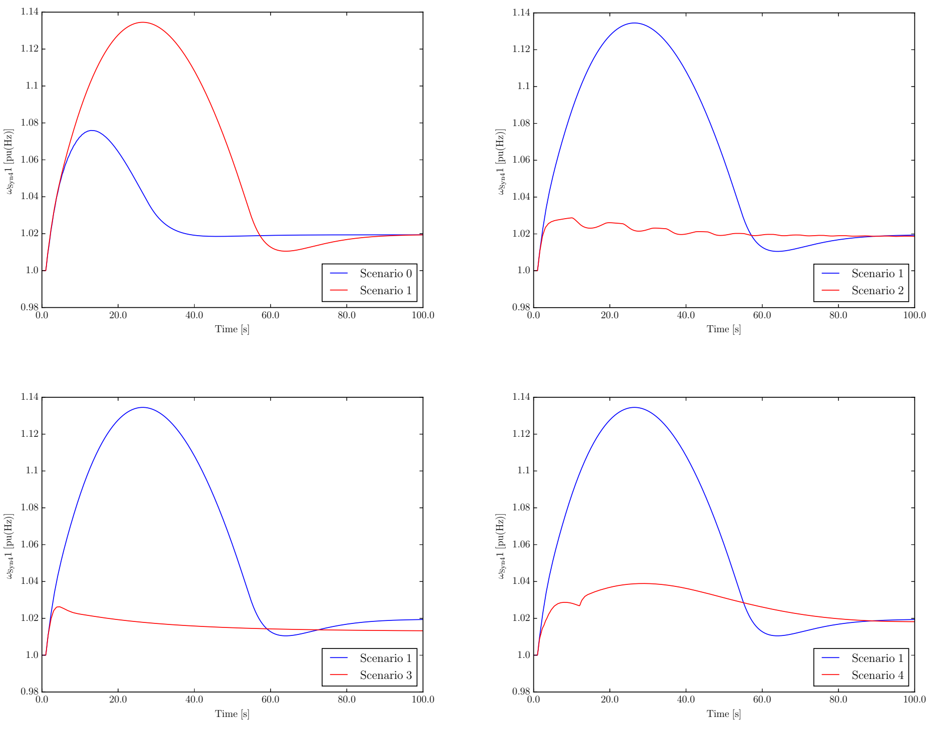


Figure 3. Typical results of the deterministic analysis. The plots show the time response of the rotor speed of the synchronous machine connected at bus 1 for different scenarios.

A.1.5 Stochastic Analysis

In this case the modified WSCC 9-bus system is considered with stochastic wind speed and stochastic load (*PQmr*). The system is simulated for 10,000 s and the distribution of the frequency for Scenarios 1-4.

A.1.5.1 Scenario 1

Consider the modified WSCC 9-bus system with turbine governors (*wscc_reg_winds.dm*).

```
In:   cd ~/Notebooks/data_lab4
      dome -r tds --silent wscc_reg_winds.dm
      domestat --pdf --silent --xmin=0.985 --xmax=1.015 wscc_reg_winds.dat 0 110
      convert wscc_reg_winds_pdf.eps hist_sc1.jpg
```

A.1.5.2 Scenario 2

Consider the modified WSCC 9-bus system with turbine governor and frequency control on the wind turbine (*wscc_reg_wind_wcon.dm*). Discuss the effect of varying the droop (*R*) of the wind frequency control (*WindFreq*).

```
In:   cd ~/Notebooks/data_lab4
      echo "ALTER, WindFreq, REP, *, R, 0.05" > sc2b.txt
      dome --silent -r tds wscc_reg_winds_wcon.dm
      domestat --silent --pdf --xmin=0.985 --xmax=1.015 wscc_reg_winds_wcon.dat 0 113
      convert wscc_reg_winds_wcon_pdf.eps hist_sc2.jpg
```

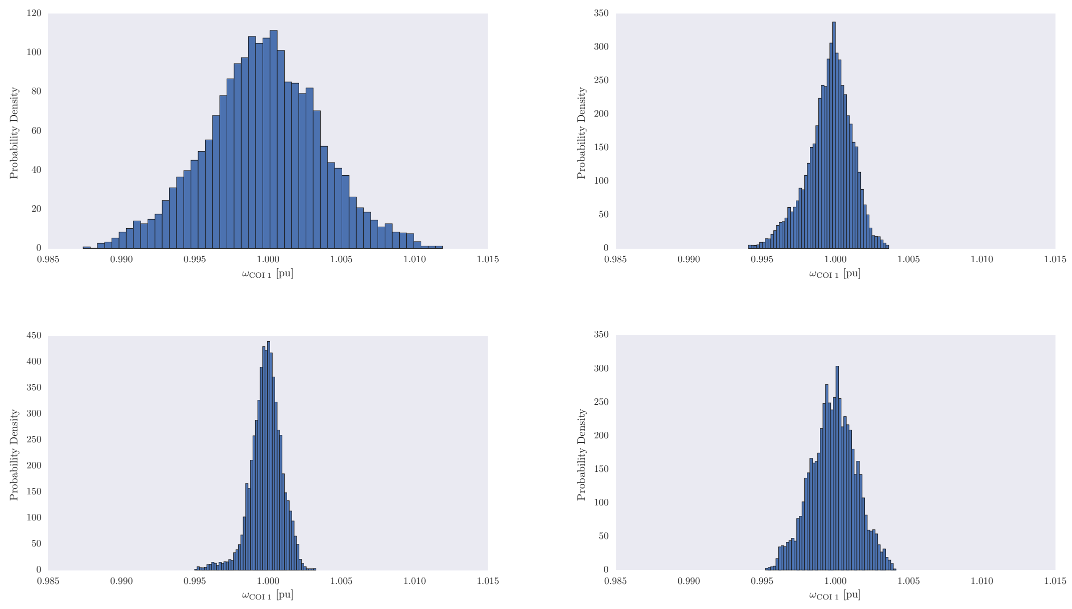


Figure 4. Typical results of the stochastic analysis. The plots show the distribution of the frequency of the centre of inertia of the system for different scenarios. Upper-left panel: no control; upper-right panel: wind turbine frequency control; lower-left panel: energy storage control; and lower-right panel: load frequency control.

A.1.5.3 Scenario 3

Consider the modified WSCC 9-bus system with turbine governors and a general storage system providing frequency control (*Stordyn3*, *wscc_reg_wind_stcon.dm*). The storage device is located at Bus 8. The storage device is located at Bus 8. Discuss the effect of varying the time constant of the active power dynamics (T_p).

```
In:   cd ~/Notebooks/data_lab4
      echo "ALTER, Stordyn3, REP, 1, Tp, 0.0026" > sc3b.txt
      dome --silent -r tds wscc_reg_winds_stcon.dm
      domestat --silent --pdf --xmin=0.985 --xmax=1.015 wscc_reg_winds_stcon.dat 0 116
      convert wscc_reg_winds_stcon_pdf.eps hist_sc3.jpg
```

A.1.5.4 Scenario 4

Consider the modified WSCC 9-bus system, with thermostatically controlled loads (*ThlFreq*, *wscc_reg_wind_locon.dm*). 33% of the loads at Bus 5, 6 and 8 are controlled. Discuss the effect of varying the gain of the frequency controller (K_f) of the thermostatically controlled loads.

```
In:  cd ~/Notebooks/data_lab4
     echo "ALTER, ThlFreq, REP, *, Kf, 10" > sc4b.txt
     dome -r tds --silent wscc_reg_winds_locon.dm
     domestat --pdf --silent --xmin=0.985 --xmax=1.015 wscc_reg_winds_locon.dat 0 119
     convert wscc_reg_winds_locon_pdf.eps hist_sc4.jpg
```

A.1.6 Suggested Optional Activities

- Repeat Scenario 1 reducing the inertia of the synchronous machines.
- Repeat Scenario 1, both deterministic and stochastic case using turbine governor with a deadband (*Tg1db*).
- Repeat Scenario 3 in the stochastic analysis and consider the effect of varying the initial stored energy (E_0) close to in the range from E_{min} to E_{max}.

You are welcome to pursue any other analysis that you consider relevant for frequency control.

AKNOWLEDGEMENTS

This material is based upon works supported by Science Foundation Ireland, under Investigator Programme, Grant No. SFI/15/IA/3074.

REFERENCES

[1] F. Milano, and R. Zárate-Miñano. Using Python for the development of Electrical Engineering Projects. ICERI, Madrid, Spain, 14-16 November 2011.

[2] F. Milano, Experience of Unix Terminal-based Labs for Undergraduate Modules on Power System Analysis, EDULEARN, Barcelona, Spain, 7-9 July 2014.

[3] F. Milano, L. Vanfretti, Role of Non-commercial Software in Undergraduate Electric Energy Systems Programmes, EDULEARN, Barcelona, Spain, 6-8 July 2015.

[4] F. Milano, Problem Solving through Limit-case Analysis: Experience in Electrical Engineering Programmes, EDULEARN, Barcelona, Spain, 4-6 July 2016.

[5] F. Milano, Thinking Nonlinearly: Experience in Teaching Power System Stability Analysis in Engineering Programmes, EDULEARN, Barcelona, Spain, 4-6 July 2016.

[6] F. Milano, A Python-based Software Tool for Power System Analysis. PES General Meeting 2013, Vancouver, Canada, 21-25 July 2013.

[7] Jupyter Project, available at http://jupyter.org

[8] P. Kundur, Power System Stability and Control. New York: McGraw- Hill, 1994.

[9] P. M. Anderson, A. A. Fouad, *Power System Control and Stability*, 2nd Edition, IEEE Press, John Wiley Interscience, Hoboken, 2003.

[10] P. W. Pai, M. A. Sauer, *Power System Dynamics and Stability*, Prentice Hall, Upper Saddle River, 1998.

[11] F. Milano, *Power System Modelling and Scripting*. London: Springer-Verlag, 2010.