

Seddighi, M, Allanson, D, Rothwell, G and Takrouri, K

Study on the use of a combination of IPython Notebook and an industry-standard package in educating a CFD course

<http://researchonline.ljmu.ac.uk/id/eprint/13062/>

Article

Citation (please note it is advisable to refer to the publisher's version if you intend to cite from this work)

Seddighi, M, Allanson, D, Rothwell, G and Takrouri, K (2020) Study on the use of a combination of IPython Notebook and an industry-standard package in educating a CFD course. Computer Applications in Engineering Education. ISSN 1061-3773

LJMU has developed **LJMU Research Online** for users to access the research output of the University more effectively. Copyright © and Moral Rights for the papers on this site are retained by the individual authors and/or other copyright owners. Users may download and/or print one copy of any article(s) in LJMU Research Online to facilitate their private study or for non-commercial research. You may not engage in further distribution of the material or use it for any profit-making activities or any commercial gain.

The version presented here may differ from the published version or from the version of the record. Please see the repository URL above for details on accessing the published version and note that access may require a subscription.

For more information please contact researchonline@ljmu.ac.uk

Study on the use of a combination of IPython Notebook and an industry-standard package in educating a CFD course

Mehdi Seddighi  | David Allanson  | Glynn Rothwell  | Khaled Takroui 

Department of Maritime and Mechanical Engineering, Liverpool John Moores University, Liverpool, UK

Correspondence

Mehdi Seddighi, Department of Maritime and Mechanical Engineering, Liverpool John Moores University, James Parson building, Byrom street, Liverpool, L3 3AF, UK.

Email: m.seddighi@ljmu.ac.uk and mehdiseddighi@gmail.com

Abstract

It is common that industry-standard packages are used in teaching professional engineering courses in final-year undergraduate and postgraduate levels. To improve the competency of students in using such professional packages, it is important that students develop a good understanding of theoretical/fundamental concepts used on the packages. However, it is always a challenge to teach theoretical/fundamental concepts in the computational-related courses. The teaching of such subjects can be improved by the use of advanced open-source web applications. The present research proposes an approach based upon the combination of Jupyter Notebook and an industry-standard package to teach an applied, computationally related course. We investigate the use of backward design and a novel tool called IPython (Jupyter) Notebook to redesign a postgraduate Computational Fluid Dynamics (CFD) course. IPython Notebook is used to design a series of integrated lecture slides and tutorial tasks, and also one of the assignments for the blended-learning-based, semester-run, CFD course. The tool allows the implementation of backward curriculum design and a learn-by-doing approach in redesigning the course. The materials produced were used on the first part of the course which contributed 40% towards the course's final mark and delivered the fundamental concepts of CFD over the first half of the semester. The remaining 60% of the mark was based on a final project from the materials taught on using an industry-standard CFD package in solving complex CFD problems during the second half of the semester. It was shown that the IPython environment is a very useful tool which provides learning-by-doing practices allowing students to have a coherent integrated lecture, tutorial, and assignment material in a highly interactive way. It improved (a) students' engagement in teaching complex theoretical concepts, (b) students satisfaction of the course and (c) students performance in working with the industry-standard package over the second half of the semester.

KEYWORDS

backward design, Computational Fluid Dynamics (CFD), computational-related courses, IPython (Jupyter) Notebook, learn-by-doing, pedagogy

[Correction updated after publication on dated 02 June 2020: in Table 1, the second row (first cell under "Options") "Strongly" was corrected to "MEng/MSc"]

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2020 The Authors. *Computer Applications in Engineering Education* published by Wiley Periodicals LLC

1 | INTRODUCTION

1.1 | Background

The education of engineers is generally targeted to (a) prepare students for research, (b) prepare graduates for employment in the engineering industry, (c) prepare engineering/science/numerically literate citizens for society and (d) provide an intellectually stimulating education [1]. It is widely acknowledged that academic performance in Science, Technology, Engineering, Mathematics (STEM) subjects plays a pivotal role in sustaining economic growth and advancing technological innovations. However, a diminishing interest of young people to pursue STEM subjects increasingly becomes a serious concern for many nations such that the issue is sometimes called the STEM crisis or national crisis [17], for example, Australia [6], Canada [25], India [13], Ireland [11], Japan [23], UK [7,26] and the US [18]. In the case of the UK, it is expected to become a more challenging issue after Brexit [5]. It seems a pedagogical reform may be needed to improve motivation, participation, retention and graduation of students in STEM subjects. Among the important elements to achieve this are leveraging active inclusive learning environments and innovative teaching pedagogy. The learning environment can be improved by introducing new technologies in blended formats into the teaching that combine classroom, online and community-based learning opportunities.

Backward Design is an approach that proposes an innovative way of designing curriculum, pedagogical and education research [15]. On the traditional approach, the pedagogy deals with the coverage and activity-oriented design. In contrast, the backward design identifies the goal of teaching the course first, and then appropriate curriculum, assessment, and finally learning activities for study are designed. In summary, the approach involves the following three stages: (a) identify desired results, (b) determine acceptable evidence of successful learning and (c) plan and design learning experience and activities.

IPython (Jupyter) Notebook is a technology which allows an interactive web environment, in which one can create and share documents that contain live code, rich text, mathematics, plots and rich media (see Figure 1a). Another interesting feature of the platform is that it is open-source and the code can be run in multiple programming languages (e.g., Python, Julia, MATLAB, etc). Hence, the users would not necessarily need to learn a new programming language, rather they can run the codes with an available kernel of their choice. The use of the platform has increased significantly in the last few

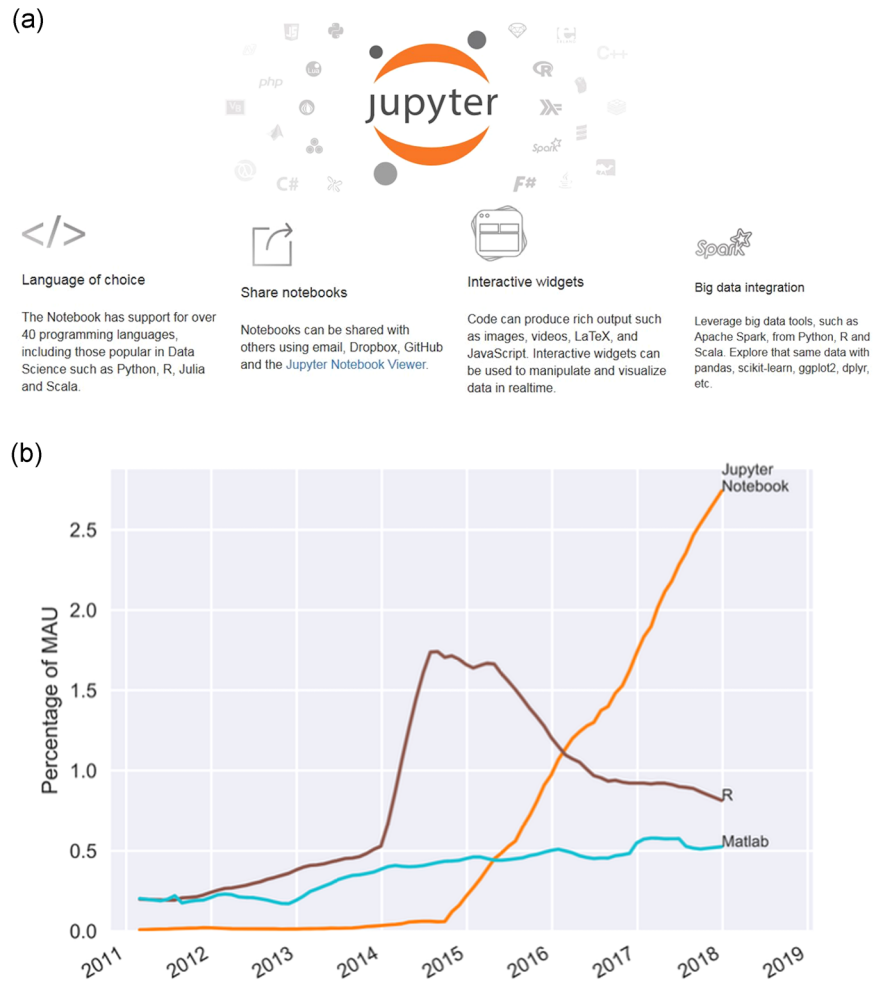
years in comparison with other popular computing packages (e.g., Matlab and R; see Figure 1b).

Given the variety of toolsets which can be used within the platform, it provides an interactive and exploratory environment for teaching, tutorials, as well as assignments of computing-related teaching. Moreover, computer codes can be embedded in books; hence, a reader can interactively change the content of the code [9]. The following provides a brief summary of some of the examples in which IPython Notebook has been used for teaching.

Chemical and geoscience engineering [24] used IPython Notebook in teaching an electronic structure to undergraduate physical chemistry and materials science courses. An integrated environment was developed for the theory and programming code that converts real space vectors to reciprocal space vectors. The authors proposed IPython Notebook as an effective tool particularly for teaching the conversion of real space vectors to reciprocal space vectors which is necessary for more elaborate electronic structure calculations. Moreover, a systematic pedagogical framework was proposed for teaching this tool to students in physical chemistry and materials science courses. Another very successful experience of using IPython Notebook in teaching is given by Jacobs et al. [14] to teach computer programming to geoscientists.

Using anonymized student marks and feedback data over the academic years between 2010 and 2014, the authors concluded that the tool is a very useful environment to improve teaching programming. *Computational biology* [19] developed a flipped-classroom approach for an undergraduate course on computational biology, by making the use of IPython Notebook. A new classroom was also designed for the course to improve the engagement of the students in the class. *Economics, business and finance* [22] used the IPython Notebook environment to develop some interactive tools for teaching economic models and cash management courses. The authors studied an illustrative example of the application of the approach on a cash management course. They discussed relevant questions raised by the students in the classroom. The authors claimed that the tool could improve the teaching of the subject (particularly those with quantitative models) through promoting a learn-by-doing approach. Moreover, it could connect theory to practice by enabling students to implement their own decision support tools. *Data literacy*. IPython Notebook was used by Wermelinger [27] for teaching data literacy at the Urban Data School. The author indicated that "It is a no-brainer, when you think about it, even for primary, because you have your code, inputs and

FIGURE 1 (a) IPython (Jupyter) Notebook; (b) monthly active users (MAU) of IPython Notebook and other two popular computing tools [10]



outputs all in one window, and can step through code execution instead of (or as well as) using more conventional tools like Trinket, Thonny or IDLE". *Physics & Astronomy, Mathematics*. A very useful series of IPython quantum mechanics lectures are provided by Dawes [8] by making the use of QuTiP (Quantum Toolbox in Python) package. Bowler [3] used IPython Notebook to teach physics to Physics & Astronomy students at University College London. A number of useful IPython Notebooks were produced for the third year quantum mechanics course. The Notebook was found to be an interactive tool. Ramz [21] developed a platform to improve teaching statistics through IPython. As per the feedback which the authors got from the learners, the content of the ICT course was sufficient, short and accurate.

Computational Fluid Dynamics (CFD) is one of the pioneering IPython Notebook teaching materials for STEM courses and was delivered by Barba and Forsyth [1]. A step-by-step approach towards computational solution of Navier–Stokes equations, as one of the most important and complex equations in fluid dynamics, was provided using 12 Jupyter Notebooks. As per the

feedback from quite a few academics around the world who have used the tool, the work was not only very successful in teaching this particular subject but has inspired for many other Jupyter course developments, including the present project.

1.2 | Present work

MSc and final-year MEng students in Maritime and Mechanical Engineering at LJMU took a course called Advanced CFD. For the academic year 2018–2019, a total of 61 students were enrolled in the class. The course aims to (a) explore the underlying theory of CFD industry-standard codes; (b) investigate performance and reliability of CFD codes in engineering applications and (c) enable students to have a critical approach towards the appraisal of CFD predictions. The present research aims at introducing a novel approach for the pedagogical design of the CFD course. A framework called *Backward Design* is used and new state-of-the-art Jupyter Notebook technology is embedded into the course.

ILO1: <i>Appreciate</i> the theory underpinning commercial CFD codes.
ILO2: <i>Critically evaluate</i> the output from a CFD analysis.
ILO3: <i>Set up and validate</i> efficient and accurate CFD models of a range of <i>simple engineering flows</i> under steady and unsteady conditions.
ILO4: <i>Set up and validate</i> an efficient and accurate CFD model of a <i>complex flow</i> (steady or unsteady) regime.

FIGURE 2 Intended learning outcomes (ILOs) of the CFD course

2 | METHOD

This section describes the structure of the CFD course and also the approach used to redesign the course. Figure 2 shows the intended learning outcomes (ILOs) of the course. Although the first two ILOs are focused on understanding the fundamental concepts about CFD, the last two ILOs deal with teaching and enhancement of students ability to carry out an appropriate CFD simulation using an industry-standard CFD package.

2.1 | Course's old design

The *old design* refers to the academic year 2017–2018, with a total number of 53 students registered in the CFD course. The teaching and learning materials included a series of PowerPoint lecture slides and tutorial sheets (see Figure 3a).

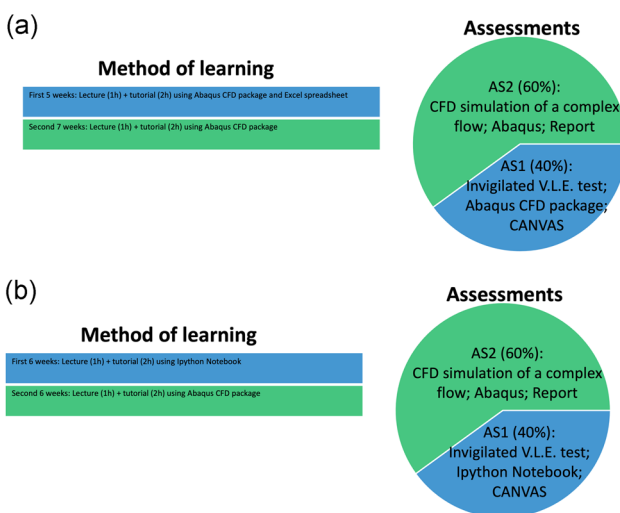


FIGURE 3 Method of learning and assessments of the course; (a) old design; (b) new design

The theoretical aspects of CFD are covered in the lectures, with students having the opportunity to practice CFD simulation of flow via a range of tutorials and assignments:

- First-half of the course: 4-week-tutorial sessions to do tutorials on CFD simulations using an industry-standard software, and 1 week to practice several approaches of CFD simulations of steady advection-diffusion problems, using an Excel spreadsheet. An invigilated CANVAS virtual learning environment (VLE) test (AS1; weighted 40% towards the course's final mark) was taken in Week 6, concerning CFD simulations of a simple model using the CFD industry-standard software.
- Second-half of the course: 3 weeks of tutorial sessions to do further tutorials on CFD simulations using the industry-standard software. A project concerning CFD simulation of a complex flow was then introduced to the students and they worked on the project over the last 3 weeks of the course. For the second assignment (AS2; weighted 60% towards the course's final mark), students submitted a scientific report on the CFD project at the end of the semester.

2.2 | Course's new design

Course feedback from both the students and staff on the *old design* indicated that (a) students were struggling (and less interested) to understand the theoretical aspects of CFD; (b) there was no appropriate assessment for such aspects of the course (the first two ILO's in Figure 2). This was a problem because the understanding of the theoretical aspects is imperative to help students to achieve ILOs.

Inspired by the improvements of the pedagogical practice obtained from the literature discussed in Section 1, we redesigned the course for the subsequent academic year, 2018–2019, in which a total number of 61 students were registered in the course. The major changes were made to the first part of the course, in which complex CFD theories were taught (i.e., first-half of the course in Figure 3b). Accordingly, we (a) used a framework of *Backward Design* and (b) embedded some *Jupyter Notebooks*, to integrate lecture slides, tutorials and assignments in an interactive environment.

Feedback discussed in the class with students, examination results and comments from other CFD instructors whom the authors were in contact with has been used to “qualitatively” evaluate the performance of the implemented approach.

3 | HOW ARE THE BACKWARD DESIGN AND JUPYTER NOTEBOOK APPLIED?

This section discusses details of the ways in which *Backward Design* is applied to the course, and also the structure of the *Jupyter* notebooks and how they are delivered. Three *Backward Design* principles, for curriculum design, are followed in redesigning the course:

Step 1: Identify desired results. This concerns what the lecturer expects students to achieve when they have completed the course. Hence, this step may characterise the ILOs of the course. We found the ILO's associated with the course's old design as a series of very appropriate results could be expected from the students who take this course. Hence, no change was made to the ILOs (Figure 2).

Step 2: Determine acceptable evidence of successful learning. This determines the method used to assess what students have learnt from the course and whether they have accomplished the learning outcomes. Hence, this step may characterise the type of coursework, assessment and exam. The first assignment on the course's old design consisted of the use of the CFD industry-standard software to simulate a simple model, without in-depth knowledge of the theory underpinning commercial CFD codes. Therefore, it was unable to fully assess students particularly for the first two ILO's. The assignment was replaced with the new first assignment which was an open-book, invigilated, CANVAS VLE test. Five multi-choice questions (selected response) together with three open-ended questions (constructed-response) were designed on CANVAS. The questions required students to run Jupyter scripts to generate results, plot relationships in IPython or Excel spreadsheets (the files can also be submitted) and comment on the findings.

As feedback, the Jupyter Notebook scripts were returned to students, with marks and feedback, and answers were discussed in class to enhance learning.

Step 3: Plan and design the learning experience and activities. This characterises the activities that students need to do during the course to prepare for the course assessments. This step characterises tutorials, homeworks and any other lecture material that will be used over the lecture and tutorial sessions. We designed a series of Jupyter Notebooks each of which consists of an introduction to the theoretical concepts, programming codes, and tasks. Figure 4 shows three snippets for different parts of one of the IPython Notebooks, which concerns the finite volume method for steady advection-diffusion problems with zero source term (Figure 4a). Note that the snippets demonstrate just a section of each

part of the notebook. Below are some details about the structure of the notebooks:

- Introduction to the theoretical concepts. The notebook starts with some descriptions as an introduction to the theoretical aspects; Markdown cells are used for this part (Figure 4a). This component of the document is generally delivered during a 1-hr lecture session in each week.
- Programming code. The programming part (Figure 4b) is embedded using code cells and contains the mathematical theories which are discussed in the first part. It is mainly delivered during 2-hr tutorial sessions during the week. Considering the fact that some students started the course with limited (or even no) experience of programming, the code is presented in a step-by-step manner. Moreover, some comments are added next to each line, to further describe the line. For instance, for the notebook shown in Figure 2, the codes are presented under four steps, namely (a) Step 1: grid generation; (b) Step 2: set-up coefficients from discretisation; (c) Step 3: solution of the matrix equation; (d) Step 4: plot the solution.
- Tasks. This is the part which outlines the tasks that students are required to practice during the tutorial session and self-study time (Figure 4c).

3.1 | How the Jupyter Notebooks were delivered to the students

3.1.1 | Week 1

An introduction session was delivered to students, as a crash course on basics and how to run the Jupyter Notebook, during the first tutorial session of the course (Figure 5). Despite being short, the introduction was prepared on a step-by-step basis, and this helped students (in particular those who had limited knowledge of programming skills) to get into the topic. It is believed that it helped to prepare and expose students to the techniques and tools built into the course. To get students fully prepared for running CFD codes in Jupyter, they were also asked to practice the Introduction notebook on their self-study time over the coming week, before the Week 2 session.

3.1.2 | Weeks 2–6

We went through one Jupyter Notebook each week. The theoretical concepts (which were implemented as the first part of the Jupyter Notebook) were discussed in a

(a)

Finite Volume Method for Steady Advection (Convection)-Diffusion Problems with zero source term

Dr Mehdi Seddighi, Dr David Allanson

Department of Maritime and Mechanical Engineering, Liverpool John Moores University

v1.1 April 2018

Outline of the lecture

- Governing equations for steady 1-D heat advection and conduction
- Steady 1-D Advection Diffusion Problem with *Central Difference* scheme
- Steady 1-D Advection Diffusion Problem with *Upwinding* scheme.

1. Governing equations for steady 1-D heat advection and conduction

Governing equation for general transport equation (from lecture note 2):

$$\frac{d}{dt}(\rho\phi) + \text{div}(\rho\phi u) = \text{div}(\Gamma \text{grad}\phi) + S_\phi \quad (\text{Eq. 1})$$

In steady diffusion problems (lecture note 3; Tutorial 1):

$$\text{div}(\Gamma \text{grad}\phi) + S_\phi = 0 \quad (\text{Eq. 2})$$

(b)

2.2. Implementation into the solver

Step 1: Grid generation

Step 2: Set-up Coefficients from discretisation

In [32]:

`clear all`

In [33]:

```
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
import matplotlib.pyplot as plt
import numpy as np
import time
%matplotlib inline

# Parameters
nx = 5 #number of cells in x-direction
nt = 100000 # Maximum number of iterations
xmin = 0 # Minimum x location [Units: m]
xmax = 1.0 # Maximum x location [Units: m]
T_L = 500 # Temp. at left boundary [Units: K]
T_R = 100 # Temp at right boundary [Units: K]
gam = 0.1 # Diffusion Coefficient [Units: kg.m^-1.s^-1]
A = 1 # Cross-sectional area [Units: m^2]
rho = 1 # Density [Units: kg.m^-3]
U = 0.1 # Velocity of material - +ve for motion right to left (ie from cell 1 towards cell nx) [Units: m.s^-1]

dx = (xmax-xmin)/(nx) # Cell width [Units: m]
D = gam*A/(dx) # Represents the effect of Diffusion associated with the moving material [Units: kg.m^2.s^-3.]
F = rho*U*A # Represents the mass Flux associated with the moving material [Units: kg.s^-1]
Peclet = F/D # Cell Peclet number - ratio of advective to diffusive transport rates (Is this dimensionless)

# The definitions of Peclet Number shown here assume a specific heat capacity of 1 m^2.s^-2.K^-1.

#Set the Coefficients
a_W = np.zeros(nx) # Initialise the a_W coefficients array (initially set to 0)
a_E = np.zeros(nx) # Initialise the a_E coefficients array (initially set to 0)
a_P = np.zeros(nx) # Initialise the a_P coefficients array (initially set to 0)
```

FIGURE 4 Three snippets of one of the Jupyter Notebooks developed during the project; (a) Introduction; (b) Programming code; (c) Tasks

1-hr morning lecture session which was held in a lecture theatre. It was then followed by 2-hr, afternoon, practical session during which students were asked to follow the tasks outlined in the notebook.

3.1.3 | Week 7

A 3-hr session was devoted to assessing the students on the first part of the course using an open-book invigilated VLE test, details of which are described earlier in Section 3.

4 | FEEDBACK ON THE STUDENT EXPERIENCE

An anonymous survey was used to collect students' views and feedback to the implemented approach, though it may not give an accurate measure of actual learning. The survey contented six questions and was designed using Vevox. It was released 2 weeks from the end of the course and kept open for 1 week. Table 1 shows the questions and also the options provided for each question.

(c)

Task 1: Study effect of cell Peclet number on accuracy and validity of the Central Difference Approach. Use the default parameters set in the code unless otherwise instructed.

Task 1.1.

For a model containing five cells, change U to the following values 0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 1.6 m/s and record the following results: cell Peclet number, magnitude of the residual, the number of iterations required, the minimum and maximum temperatures in the model. Whilst undertaking this task make sketches of the resulting temperature field and its relationship to the analytical solution.

Task 1.2.

With regard to the results gained: How well does the central differencing scheme agree with the analytical result and when is it at its most accurate? Why does the residual reported by the solver not give any clear indication or warning as to the discrepancy between the numerical result and the analytical result? Why is the code reporting that the answer is within the tolerance (Tol) set?

Task 1.3.

Change the mesh spacing (dx) to half that used above. Re-run the test case above and determine what happens to the accuracy and validity of the results. Continue increasing the velocity U and hence determine the point when the model fails?

Task 1.3.

What is the cost of keeping the model from failing by reducing dx ?

FIGURE 4 (Continued)

The total number of students in the class was 61 from which 30 participated in the survey. The results of the first five questions of the survey are shown in Figure 6. Dominant participation of MEng students (83%) compared with that of the MSc (17%; Figure 6a) is consistent with the fact that MSc students were 25% of the total number of students on the course. A good student's satisfaction is demonstrated (see

Figure 6b-e) for (i) using the blended approach by embedding IPython Notebook in the first part of the course, (ii) lecturers attempt to engage students in the lectures, and (iii) overall satisfaction of the module. Moreover, in response to the last questions, 11 students provided descriptive comments. Some of the comments in relation to the structure of the module and IPython Notebook are as below:

- **Unicode Identifiers (Works with Python3 and Julia)**

To get α variable, type $\backslash alpha$ Then Press Tab.

```
In [2]:  $\alpha$  = 1 # Type \alpha and press Tab
 $\beta$  = 2 #\beta
 $\lambda$  = 0 #\lambda
 $\Lambda$  = 0 #\Lambda
 $\alpha$  = 0 #\mfraka - Notice it is not monospaced - Thinner
 $Q$  = 0 #\msansQ - Notice it is not monospaced - Wider
 $\alpha$  /  $\beta$ 
```

Out [2]: 0.5

- **Text Editor**

A multilingual text editor is added with highlighting for many programming and scripting languages.

- **Build-it Terminal** You can start a terminal from jupyter to access your machine. Thanks to the development team for making it black background with white text.
- **Jupyter Drive** You can share jupyter notebooks from your Google Drive using the new [Jupyter Drive](#). This allows you share jupyter notebooks like NBViewer with all the access control that Google Drive provides.

2. Running Jupyter Notebook on your home pc/Mac

- **Download and install Jupyter Notebook and python version 3.5 or 3.7 (leave this step if you have already got Jupyter and Python 3.5 installed on your machine):** <http://jupyter.readthedocs.org/en/latest/install.html> [link checked on 07 October 2019]
 - Make sure you do not install version 2.x python as the notebook will not work with these versions of python.
 - If you are new to Python/IPython and haven't got any earlier versions of Python on the machine, you can install IPython via "[installing Anaconda](#)". [link checked on 07 February 2019]
- **Gerate a new folder on your machine, and download, into this folder, the .ipynb and all other files from corresponding folder on Blackboard.**
 - Make sure that i) there is no space or dashed symbol (underscore symbol is ok) in the folder's name; ii) the name of the files downloaded in the folder, will remain the same as the above. If they not, rename the files accordingly.
- **Run ipython notebook** __

FIGURE 5 A snippet of the introduction session

TABLE 1 Survey's questions and options

Question	Options
1. Are you an MSc or MEng student?	MEng/MSc
2. I found IPython Notebook very useful in learning key fundamental concepts in CFD	Strongly disagree/Disagree/Neither agree nor disagree/Agree/Strongly agree
3. I like the blended structure of the module, that is, first-half mainly on learning fundamental concepts in CFD using IPython, and the second-half on using a general-purpose CFD commercial package	Strongly disagree/Disagree/Neither agree nor disagree/Agree/Strongly agree
4. The lecturer provided opportunities for class participation	Strongly disagree/Disagree/Neither agree nor disagree/Agree/Strongly agree
5. How satisfied were you with the level of teaching?	Not at all satisfied/Not very satisfied/Slightly satisfied/Very Satisfied/Extremely satisfied
6. Please provide any further feedback below	

- *"The use of IPython notebook was an excellent way to learn the fundamental concepts of CFD and was a very useful and enjoyable aspect of the module".*
- *"Overall enjoyed the module, the lectures were informative and engaging and the tutorial helped increase the level of understanding".*
- *"I think this is one of the best analysis modules I have ever done! All members of staff who work on the module are so helpful and nice!"*
- *"One of the best-taught modules of the year, clear instructions and fed well into the in-class assessment".*

5 | DISCUSSIONS AND CONCLUSIONS

We have presented a novel framework for teaching an advanced CFD course. A successful model of curriculum planning in education, called *Backward Design*, has been used in redesigning the CFD course. The novel framework proposes to split the teaching weeks into two: (a) the first half of the semester consists of teaching primarily some important fundamental theories used in the industry-standard CFD packages. This includes doing tutorials and lectures, and eventually taking an open-book, invigilated assignment, using Jupyter Notebook; (b) over the second half of the semester, teaching and tutorial sessions devote to teaching/practising various aspects of CFD simulations using industry-standard packages. Finally, students are required to study a CFD problem using the package and submit a report on the project as their second assignment.

IPython (Jupyter) Notebook was used to design assessment and teaching materials for steps two and three of the *Backward Design* model, respectively. There is an increasing growth of embedding Jupyter Notebook in

various Engineering and Science subjects. The consensus conclusion by all the studies reviewed herein is that Jupyter Notebook is an incredible tool to provide *blended* learning and embed *learning-by-doing* into lectures, tutorials, as well as open-book, open-web, computer-based assignments. Moreover, its flexible programming section allows the learners to examine and debug their programming codes and make comments for the instructor on the notebook. In addition, the approach can help to promote active learning which involves students in doing things and thinking about the things they are doing [2]. Finally, Jupyter Notebook is starting to be considered as a great tool for use by publishers which gives the readers an opportunity to interactively change the content of the programming codes [9]. Specific conclusions which can be made from implementing the approach to CFD module at LJMU are as follows:

- *Assignments.* An open-book invigilated VLE test was designed to assess students for the first two ILO's of the course. Compared with the paper-based, the computer-based assignment provides faster, more accurate marking, and reduced test administration times, and make possible relatively low-cost scaling to large numbers of test-takers. Computer-based assignments could also be designed to meet the needs of special populations, including people with physical disabilities and people from-diverse cultural or linguistic backgrounds [16]. Due to the nature of the open-book, open-web assignment, students skills on higher levels of Bloom's taxonomy can be evaluated [12]. The assignment constructed for the CFD module was designed to comply with the requirement of assessment practice, namely *validity*, *reliability*, *rigour*, *probity* and *fairness* [20]. Fundamental theories used in the industry-standard package theory

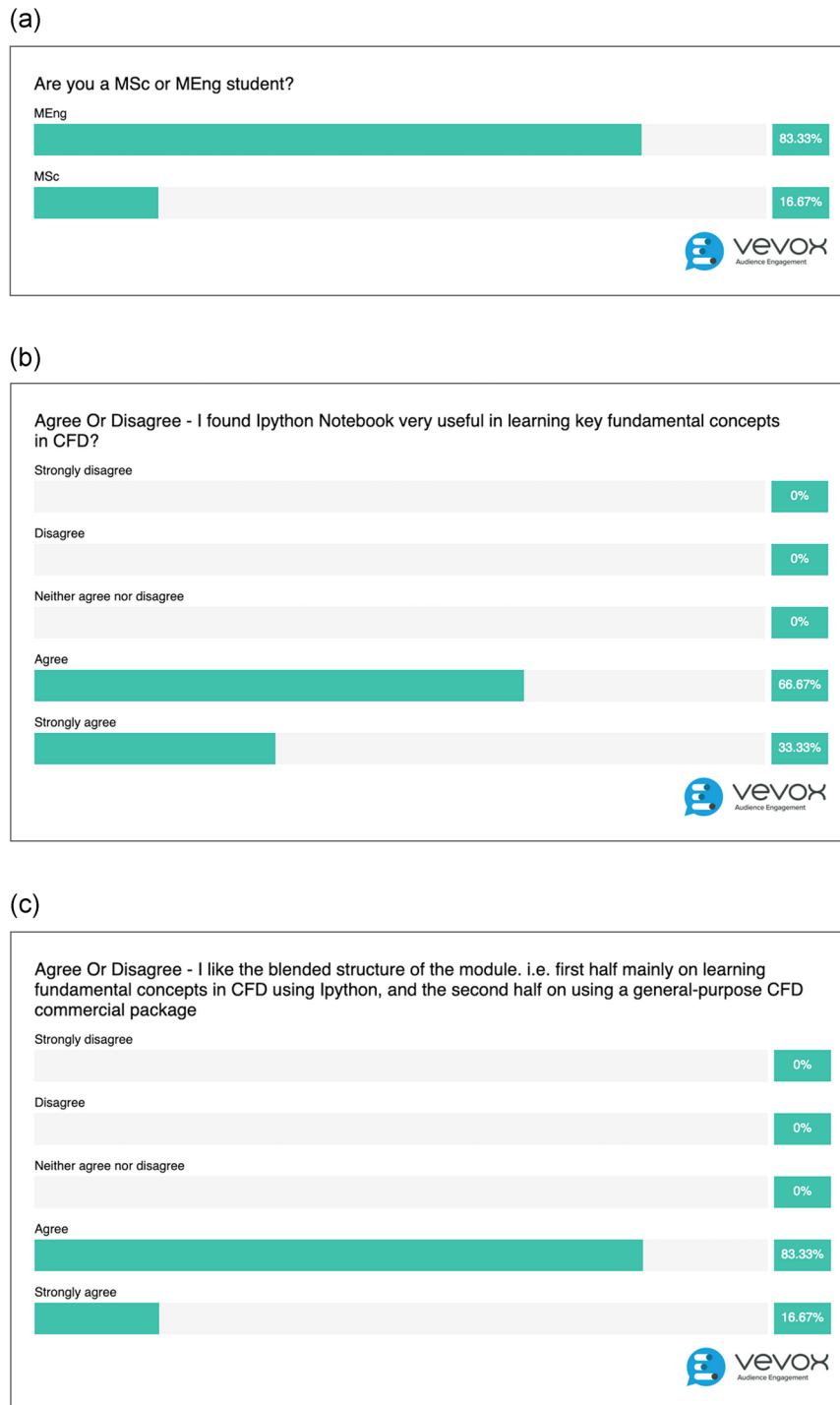
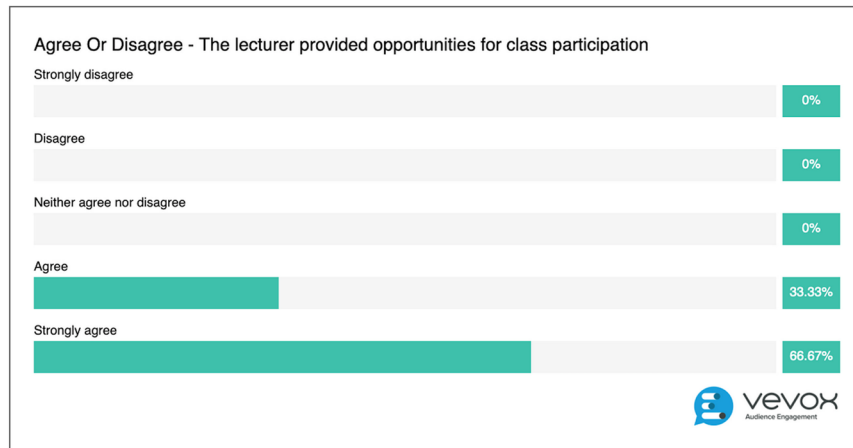


FIGURE 6 Results of the survey

assignment were made available, using the same IPython scripts which students had worked with for a few weeks. Hence, on the exam, students knew the code environment and could focus on analysis and interpretation of the results which were asked (*Validity*). All questions were either multiple choice or a specific, detailed pre-prepared model answers were used. This made the marking more consistent

(*Reliability*). The questions were encrypted with a password, but the password was announced to students at the beginning of the exam. Also, before the start of the exam, lecturers made sure that IT technicians had preloaded and opened IPython on all machines and they were all ready to use. We also provided a printed version of the questions and asked students to write their answers on the paper as well

(d)



(e)

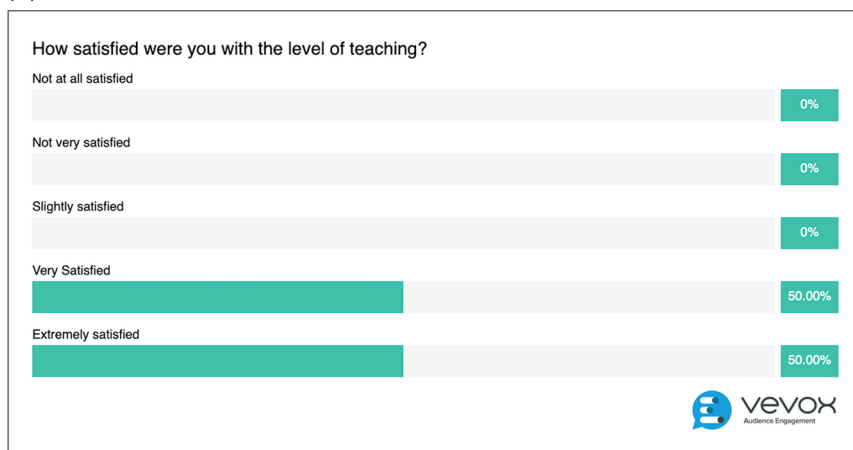


FIGURE 6 (Continued)

as entering them into VLE, to avoid possible interruption of the exam just in case CANVAS/internet connection went down (*Rigour, Probity and fairness*).

- *Lectures/tutorials.* IPython Notebook is a useful tool to design *learn-by-doing* teaching materials. It combines computations, writing and mathematics in a highly interactive environment. The Notebooks are designed in a manner that the taught materials can be integrated into the second part of the module on which the use of a commercial package in the simulation of more complex problems was taught. This helped us to teach some of the challenging theoretical/fundamental concepts that appeared when using some of the parameters in the industry-standard package, by providing a blended, learning-by-doing learning environment for the students:

(i) The concept/parameter was referred to a particular Jupyter notebook in which the embedded programming code contains the particular parameter, for a simple problem,

(ii) The particular notebook was projected on the screen and explanations were given while the lecturer running the notebook live.

(iii) In the computer-lab tutorial session, students were asked to run the annotated Jupyter Notebook themselves, and practice the effect of changing the parameter being investigated. Despite problem-solving with no guidance, it is believed that the worked examples, established within the IPython Notebook for the complex CFD concepts, provide learning improvements [4].

The improved mark for AS2 in the academic year 2018–2019 may suggest that when the students were running the industry-standard CFD package (in the second half of the course) to simulate real-life complex fluid dynamics problems, they had a better understanding (compared with the student cohort took the course in the previous year on the old design) of the parameters required to set-up a simulation in the package's solvers. The students'

survey and also comments from other instructors suggest that the implemented approach enhanced engagement of students in particular for teaching the fundamental theory of CFD. Moreover, compared to the old framework taught before, the class average mark on the second assignment, AS2, increased from 62% to 66%. Given the fact that the new design did not change much on the second part of the course, this can be an indication that student's understanding obtained from the first part on the theoretical concepts and CFD analysis may have helped to improve their performance for the second part as well. Also, it was possible to include more interests and aspirations of all of the students in the cohort, in designing the lecture, tutorial, and assignment. In addition, one of the most important benefits of using the approach was that the Jupyter notebooks made it much easier (and sensible to students) to explain a concept used in the industry-standard CFD codes by referring to some simple programming parts in the notebooks.

Finally, the tool is open-source, needs zero-install hosted service, and is compatible with various platforms.

- *Contribution of the present study to the literature.* The major contribution into the available literature from the present research is to propose an approach based upon the combination of a Jupyter Notebook and an industry-standard package in teaching an applied, computationally related course. Applied computationally related courses in engineering subjects are usually taught using either “programming-based” (e.g., using IPython Notebook, Matlab, Excel) or “commercial package-based” approaches. The former approach mainly focuses on teaching theoretical/fundamental aspects of the modelling but usually limited to considering simplified problems only. The latter approach, on the contrary, can consider the modelling of more complex problems but is generally lacking in the teaching of theoretical/fundamental aspects underpinning the packages. We have shown here that IPython Notebook is a great tool that can be used in combination of commercial packages in the teaching of computationally related courses. The results are presented for the implementation of the blended approach to a thermofluids course. However, we believe the proposed blended teaching method can be applied to many other Engineering subjects (e.g., Civil Engineering, Chemical Engineering) in which numerical courses are taught.
- *Suggestions for future work.* As discussed in Section 2, evaluation of the effectiveness of the implemented

approach was based on feedback obtained from students and other instructors and examination results. More quantitative data may have to be analysed to judge better the effectiveness of the proposed framework. Moreover, for the use of the developed assignment, it was a bit difficult to assess the engineering process necessary to reach the correct answer. So, perhaps some new-type questions should be designed to improve the assessment for that aspect.

ACKNOWLEDGEMENT


The authors gratefully acknowledge fruitful discussions with Dr Sean Malkeson and Dr Philip Carey during the course of the project.

ORCID

Mehdi Seddighi  <https://orcid.org/0000-0003-4941-5111>

David Allanson  <https://orcid.org/0000-0001-7532-7938>

Glynn Rothwell  <https://orcid.org/0000-0001-5799-2862>

Khaled Takroui  <https://orcid.org/0000-0002-0514-9125>

REFERENCES

1. L. Barba and G. Forsyth, *CFD Python: the 12 steps to Navier-Stokes equations*, J. Open Source Educ. **1** (2018), no. 9, 21.
2. C. C. Bonwell and J. A. Eison, *Active learning: Creating excitement in the classroom (ASHE-ERIC Higher Education Rep. No. 1)*, The George Washington University, School of Education and Human Development, 1991.
3. D. Bowler, *Investigating IPython notebook servers for teaching physics*, 2015, available at <http://blogs.ucl.ac.uk/eldg-collected/2015/10/07/investigating-ipython-notebook-servers-for-teaching-physics/>
4. O. Chen, S. Kalyuga, and J. Sweller, *Relations between the worked example and generation effects on immediate and delayed tests*, Learn. Inst. **45** (2016), 20–30.
5. B. Christian, *The UK's lack of STEM skills is a 'national crisis'—and it's only going to get worse after Brexit*, 2017, available at <https://www.wired.co.uk/article/stem-skills-and-space-industry>
6. P. Coutis, C. Cuthbert, and H. Macgillivray, *Bridging the gap between assumed knowledge and reality: A case for supplementary learning support in tertiary engineering mathematics*, 2002.
7. A. C. Croft, M. C. Harrison, and C. L. Robinson, *Recruitment and retention of students—An integrated and holistic vision of mathematics support*, Int. J. Math. Educ. Sci. Technol. **40** (2009), no. 1, 109–125.
8. A. M. C. Dawes, *Teaching quantum mechanics with python*, 2018, available at <https://github.com/amcdawes/QMLabs>
9. S. Downes, *Quantum leaps you can expect in teaching and learning in the digital age—A roadmap*, Teachonline.CA, 2017.
10. B. Frederickson, *Ranking programming languages by GitHub users*, 2018, available at <https://www.benfrederickson.com/ranking-programming-languages-by-github-users/>
11. O. Gill and J. O'Donoghue, *Service mathematics in Irish universities: Some findings from a recent study*, Adults Learn. Math. Int. J. **2** (2007), no. 2, 6–19.

12. P. J. Goodhew, *Teaching Engineering The School of Engineering's Active Learning Lab at The University of Liverpool*, 2014.
13. India Ministry of Science and Technology, *Annual report 2017-18*, Department of Science & Technology, 2018.
14. C. T. Jacobs et al., *Experiences with efficient methodologies for teaching computer programming to geoscientists*, J. Geosci. Educ. **64** (2016), no. 3, 183–198.
15. J. L. Jensen et al., *Using backward design in education research: A research methods essay*, J. Microbiol. Biol. Educ. **18** (2017), no. 3, 18.3.50.
16. J. A. Naglieri et al., *Psychological testing on the internet: New problems, old issues*, Am. Psychol. **59** (2004), no. 3, 150–162.
17. Y. Nakakoji and R. Wilson, *First-year mathematics and its application to science: Evidence of transfer of learning to physics and engineering*, 2018, p. 8.
18. National Research Council, *The mathematical sciences in 2025*, The National Academies Press, Washington, DC, 2013.
19. L. Nederbragt, *Introduction to computational modelling for the biosciences*, 2017, available at <https://flexblog.wordpress.com/2017/03/08/a-new-1st-semester-bachelor-course-introduction-to-computational-modelling-for-the-biosciences-%E2%80%8B/>
20. QAA, *Understanding assessment: Its role in safeguarding academic standards and quality in higher education*, 2012.
21. A. Ramz, *Developing a course for learning mathematics through programming*, Master's Thesis, Metropolia Ammattikorkeakoulu, 2016.
22. F. Salas-Molina and D. Pla-Santamaria, *Coding oriented learning in economics, business and finance*, Model. Sci. Educ. Learn. **11** (2018), no. 1, 55.
23. Science Council of Japan, *Nihon no Tenbou—Rigaku Kougaku Karano Teigen (Prospects of Japan: Suggestions from science and engineering)*, 2010.
24. M. N. Srnc, S. Upadhyay, and J. D. Madura, *Teaching reciprocal space to undergraduates via theory and code components of an IPython notebook*, J. Chem. Educ. **93** (2016), no. 12, 2106–2109.
25. A. Stokke, *What to Do About Canada's Declining Math Scores? (May 27, 2015)*, C.D. Howe Institute Commentary 427, 2015, available at SSRN: <https://ssrn.com/abstract=2613146> or <https://doi.org/10.2139/ssrn.2613146>
26. The Royal Society, *Vision for science and maths education*, 2017.
27. M. Wermelinger, *Nine ideas for teaching Computing at School from the 2017 CAS conference*, 2017, available at <https://duncan.hull.name/tag/mcq/>

AUTHOR BIOGRAPHIES



Mehdi Seddighi is a Senior Lecturer in Fluid Mechanics at Liverpool John Moores University. He is a MSc in Aerospace Engineering (Aerodynamics) and holds a PhD degree in Fluid Mechanics from University of Aberdeen. His research interests include unsteady turbulent flow, control of turbulent flow, hydrodynamics and marine renewable energy, using numerical and experimental techniques.

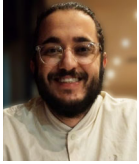
The numerical approach includes, a high-fidelity in-house DNS (direct numerical simulation) and LES (large eddy simulation) code, CHAPSim, which is developed initially by him and then further developed by other researchers under his supervision over the past decade. Mehdi is a Fellow of the Higher Education Academy and is interested developing novel techniques for teaching Thermofluids subject and Computational Fluid Dynamics courses at undergraduate and postgraduate levels.



David Allanson has held an academic post at LJMU since January 1998. Prior to that he was a member of the Advanced Manufacturing Technology research group and completed his PhD in 1995. During his time in the laboratory he was involved in the development of intelligent and adaptive control systems and the development of supporting process models. Since joining the academic staff, David has delivered modules over a range of engineering topics including Thermodynamics, Fluid Dynamics, Computational Fluid Dynamics (CFD) at undergraduate and postgraduate levels. He is currently interested exploring different techniques for the teaching of engineering CFD and encouraging the development of a robust approach to engineering analysis. David also spent some time working for National Nuclear Corporation (NNC) and worked in the commissioning division at Heysham 2 AGR where he worked on the testing of reactor instrumentation and safety circuits. David is currently Programme Leader for the Marine and Offshore Engineering MSc programme and assist with the management and running of the Mechanical and Marine Engineering undergraduate programmes.



Glynn Rothwell received his BSc degree in mechanical engineering from the University of Manchester in 1983. He subsequently spent some time in industry working on a variety of nuclear engineering projects. In 1991, Glynn was appointed as a senior lecturer in mechanical engineering at Liverpool John Moores University. He received his PhD from in 2003 for a thesis entitled “Fracture Toughness Determination Using Constraint Enhanced Sub-Sized Specimens”. His current position is Subject Head for Mechanical Engineering. Glynn is a Chartered Engineer and a Fellow of the Institution of Mechanical Engineers.



Khaled Takrouri is a PhD candidate at the Department of Maritime and Mechanical Engineering at Liverpool John Moores University. He is studying turbulence and drag reduction for flow over a backswimmer-inspired textured surface. Prior to that, he received a Master of Research degree in 2016 at University of Liverpool in Simulation in Aerospace Engineering where he studied forces on a ground of the wake of a helicopter.

How to cite this article: Seddighi M, Allanson D, Rothwell G, Takrouri K. Study on the use of a combination of IPython Notebook and an industry-standard package in educating a CFD course.

Comput Appl Eng Educ. 2020;1–13.

<https://doi.org/10.1002/cae.22273>