OPEN SCIENCE BUILDING STOCK MODELLING: AN EXAMPLE USING GBXML, OPENBUILDING AND ENERGYPLUS

Steven K. Firth, Bianca Howard, and Jonathan A. Wright
School of Architecture, Building and Civil Engineering
Loughborough University, UK

ABSTRACT

Open Science is the process of scientific study using open, reproducible and reusable datasets, models and methods. This paper explores the potential for Open Science in the field of Building Stock Modelling and investigates the benefits of this approach for the Building Simulation research community. An Open Science workflow is described for running EnergyPlus simulations on multiple buildings that could be used as the calculation engine of a future building stock model. The stages of the workflow are: 1) the original building descriptions are held as gbXML files; 2) simulation runs using EnergyPlus are carried out on the buildings; 3) the simulation results, combined with the gbXML data, are stored in a graph data structure for analysis; and 4) the results are analysed to create research findings. This is achieved using gbXML files, Python scripts, Jupyter notebooks and the 'openbuilding' Python package. The entire process is made available on GitHub.

In this paper, the workflow is demonstrated using an example stock of 72 typical UK detached houses. The paper concludes with a discussion of the benefits, challenges and limitations of Open Science approaches for building stock modelling. The choice of input datasets and simulation methods is discussed, as well as the options for making both the results and methods of stock modelling studies available for the wider research community. Techniques for improving the reproducibility and the reusability of datasets and methods are also presented.

INTRODUCTION

Kraker et al. (2011) in their paper 'The case for an open science in technology enhanced learning' propose a vision for Open Science based on four instruments:

1. Open Access - publishing the results of academic research as freely available on the public internet
2. Open Data - publishing the datasets collected in the research process, without restricting their use
3. Open Source - making software developed in the research available under an open license
4. Open Methodology - sharing the methodology of a study, and the tools used for data collection and analysis

These combine to create an Open Science approach where each stage of the research process is made available to the wider research community. This ambition is recognised at the highest levels of scientific thinking, such as the first recommendation of the 'Science as an Open Enterprise' report by the UK's Royal Society:

> 'Scientists should communicate the data they collect and the models they create, to allow free and open access, and in ways that are intelligible, assessable and usable for other specialists in the same or linked fields wherever they are in the world. Where data justify it, scientists should make them available in an appropriate data repository. Where possible, communication with a wider public audience should be made a priority, and particularly so in areas where openness is in the public interest.' (Royal Society, 2012)

This paper considers Open Science in the context of building stock modelling studies. Building stock modelling is the academic field of using computer models to predict the energy and environmental performance of large groups of buildings. The research objectives of this work are:

1. To demonstrate a Building Simulation workflow suitable for building stock modelling
2. To develop the workflow as an Open Science process with open, reproducible and reusable datasets and methods.

The concepts of Open Data and Open Methodology for building stock modelling are presented. Open Data involves the publishing of the underlying datasets in public, open-access repositories. However, this in itself is not sufficient. It also involves arranging, structuring and documenting the datasets in such a way that they are understandable and useable to others, including those who have no knowledge with the original data collection. It is this structuring of the datasets that is the challenge in Open Data working, and this is accomplished through the role of data schemas. Data schemas define how individual pieces of information in a dataset should be defined, presented and linked together. By utilising a well-defined and structured data schema, a dataset will be both understandable to others and the data format much more amenable to analysis.

Open Methodology approaches apply to the analysis of simulated building performance datasets to produce meaningful insights and results. For Open Science, the entire process of data cleaning, data processing, new variable creation, statistical analysis and final creation of graphs and tables need to be completely documented and reproducible. In practice, this means that computer programming scripts should be used to develop analysis algorithms to carry out these data analysis stages. No manual process, such as copy and pasting, should be done as these are difficult to document effectively and for others to reproduce with certainty. Analysis algorithms should be clear, readable and as simple as possible. If published in support of an academic publication, they should enable other researchers to create identical results of the publication (such as the plots and tables) from the underlying datasets.

STOCK MODEL WORKFLOWS

Table 1 shows the processes used in an existing stock model, the Community Domestic Energy Model (Firth et al., 2009). This model was chosen as a typical example of existing stock model types, which utilises large-scale government datasets and simple steady-state energy models to make predictions for groups of domestic buildings. The table shows the four stages that are used to move from the initial, original datasets to the final model predictions and analysis. For each stage, a commentary is given on the suitability of the CDEM approaches in terms of an Open Science workflow.

The limitations observed in Table 1 lead to the following recommendations for a building stock modelling workflow that would follow Open Science principles:

1. The original, externally collected datasets (such as Government surveys) should be publically available for all to access or, failing this, available under license to as many end-users as possible. Without the original data being available, it will be impossible for the building stock model to be used, tested and reproduced by other researchers.
2. A model- and software-neutral description of the buildings within the stock should first be created. This would allow this description to be published as Open Data files and clearly define the characteristics of the buildings that have been modelled. This should be done using a well-structured, interoperable data format, such as the Green Building XML format (gbXML, 2018), and the process should use well-documented computer code.
3. The process of converting the building descriptions into model input files for the simulation stage should be clearly defined using well-documented computer code.
4. The simulation engine used to calculate the energy and environmental performance of the buildings should either be: a) existing Open Source software that is available to end-users; or b) a custom simulation engine that is created using well-documented computer code and published as Open Source software.
5. The simulation results, often time-series data, should be stored in a well-structured and interoperable data format. The raw and processed simulation results could be published as part of the Open Science workflow.
6. The final analysis of the results, including producing tables and plots for publication, should be done using well-documented, understandable and reproducible computer code. This will document exactly how publication outputs have been produced, and enable other researchers to build on these methods.

In this paper, an example of an Open Science building stock modelling workflow is presented. All stages of this workflow are achieved with Open Science methods. The initial data is held in gbXML format, the established industry-standard data format for interoperable building data. All analysis steps are written in Python code and documented in Jupyter notebooks for reproducibility and reusability. Both the datasets and the methods are made publically available on GitHub to demonstrate the potential for Open Science publishing (GitHub, 2018). The computer code used for this paper is designed to be open, understandable and replicable by using the following approach:

- The code is written in Python, this language being chosen as it was primarily designed to be easy to read by others (Python, 2018). It is also widely used by the data science community.
- The Python scripts are placed in a Jupyter notebook, which allows both code and markup documentation to be placed in the same document, to enable the code to be fully documented in-situ (Project Jupyter, 2018).
- The scripts make use of the 'openbuilding' Python package, an open-source Python package for Building Simulation (openbuilding, 2018).
- All Jupyter notebooks are available to view on the GitHub repository for this paper (uSim_2018, 2018).

Table 1: Stages of the Community Domestic Energy Model (CDEM) and reflections on an Open Science approach

Stage	Description of CDEM model approach	Reflections for an Open Science approach
1. Original datasets sourced	Original data included the English House Condition Survey 2001 that held data on over 16,000 English homes. This data is only available to academic researchers via the UK Data Archive.	All stock models are likely rely on datasets collected by other parties, as data collection of large numbers of buildings is an expensive process. However, the final stock models will only be truly Open Science if these third-party datasets are available to the wider research community.
2. Original data converted to model inputs	The model input data for 49 housing archetypes were created. This was done manually and the model input data was placed in an Excel workbook.	This is an important step as many defaults and assumptions can occur at this stage, for example an estimation of a wall construction based on knowledge of the building age only. This inferencing should be done using well-documented computer code to make this process reproducible and the assumptions clearly available for all to see. It would also be preferable if the original data were first converted into a model-neutral description of the buildings, so that other models could also be tested and used. By converting directly to a single specific model input file, important parameters and other information may be excluded.
3. Simulations run and results generated	The BREDEM-8 calculation engine was used. This is a steady-state monthly energy model that was implemented in an Excel workbook. Results were calculated using formulae cells and were stored in the same Excel workbook.	Using formulae cells in Excel workbooks is potentially reproducible as the calculation steps are shown. However they are typically very hard for another research to follow and understand, and can be poorly documented. A better approach would be a series of small, modular functions in computer code that would both demonstrate the modelling process and be easily reusable by other researchers.
4. Analysis of results and findings generated	Results of model runs were manually copied into further Excel workbooks and used to create graphs and tables for publications.	Using manual processes to copy, analyse and display the results can work well, but any manual steps mean that the process becomes difficult to follow and/or involves very detailed documentation of the steps involved. Again, using computer code to create the final tables and plots will make this step understandable and reproducible to others.

DATA COLLECTION AND BASE CASE GBXML FILES

In this step, a gbXML file of a detached UK house is created. This file is then modified to produce 72 variants of the detached home with different building orientations, one for every 5° point on the compass. Each variant is saved as an individual gbXML file.

Firstly, the description of the detached house in the Allen and Pinney report on 'Standard dwellings for modelling' was taken as the foundation for the detached house model used in this work (Allen and Pinney, 1990). Using the details provided by Allen and Pinney, a 3D computer model of the house was developed in the building simulation software IES VE-PRO (Figure 1). The constructions and material properties of the walls, roofs, floor, internal surfaces, windows and doors were also included in the model. Given that the Allen and Pinney report is a paper-based report, with floor plan and section sketches, the inputting of this data into the IES software was carried out manually and is not, in itself, completely replicable. However, the assumptions and judgements made in this process are fully visible in the exported gbXML file and it is this exported file that is considered the start of the Open Science workflow.

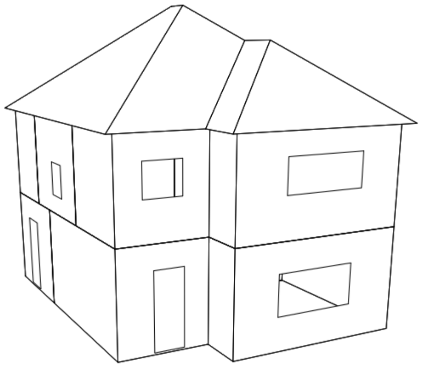

Figure 1: 3D render of the base case building – a UK detached house

The finished IES model was then exported to a gbXML file. This is an XML (Extensible Markup Language) file that has been structured according to the gbXML schema (gbXML, 2018). gbXML is a common data exchange format in the Building Simulation community, with many Building Simulation and CAD software packages having the ability to import and export gbXML files. Here further additions were made to the gbXML file exported from IES. This is necessary as the gbXML export from IES only provides geometry and construction details. These additions are done using computer code so that the process complies with the Open Science approach. In this case, the gbXML file exported from IES was imported into an openbuilding 'GbxmlGraph' class. The GbxmlGraph object was manipulated to make the following changes to the gbXML data:

- Individual zones are assigned for each space in the dwelling.
- Electric room heaters are placed in each room except the roof space and are controlled by an individual thermostat set at 21°C for 24 hours a day from 1st January to 30th April and 1st October to 31st December.
- An infiltration rate of 0.5 air changes per hour is set for each room.
- Internal heat gains were set as constant, always-on values for each room except the roof space. For each room the values used are:
  - number of people: 2
  - people heat gain: 90W/person
  - lighting heat gain: 0.5W/m^2
  - appliance heat gains: 3W/m^2
- A loop was used to modify the building orientation (stored in the 'CADModelAzimuth' gbXML element) to values between 0 and 355 in steps of 5, equivalent to 72 orientations for every 5° point on the compass.

Once complete the 72 modified objects were saved as new gbXML files. An excerpt of final gbXML file, for the building with an orientation of 0° (due North) is shown in Figure 2. The XML contains elements (such as '<gbXML>','<campus>'), attributes (such as 'temperatureUnit="C"', and text (such as 'detached_house'). The excerpt shows a Building with an id 'detached_house' and the first Space (or room) of the building. The Space is a dining room with internal heat gains, infiltration and a floor area of 13.1369m^2. The remainder of the file, not shown in Figure 2, provides information on other aspects of the building such as the coordinates of the building surfaces and openings.

```
<gbXML temperatureUnit="C" lengthUnit="Meters" areaUnit="SquareMeters"   volumeUnit="CubicMeters"
 useSIUnitsForResults="true" version="0.37">
   <Campus id="campus-1">
      <Name>detached_house</ns0:Name>
      <Location>
         <CADModelAzimuth>0.000000</ns0:CADModelAzimuth>
      <Location>
      <Building id="detached_house" buildingType="Unknown">
         <BuildingStorey id="GROUP_1">
            <Name>Ground Floor</ns0:Name>
            <Level>0.000000</ns0:Level>
         </BuildingStorey>
         <Space id="DINING_ROOM" conditionType="HeatedAndCooled" buildingStoreyIdRef="GROUP_1"
            peopleScheduleIdRef="schedule-alwaysOn" lightScheduleIdRef="schedule-alwaysOn"
            equipmentScheduleIdRef="schedule-alwaysOn" zoneIdRef="Zone-DINING_ROOM">
            <Name>DINING_ROOM</ns0:Name>
            <Area>13.136900</ns0:Area>
            <Volume>32.842250</ns0:Volume>
            <AirChangesPerHour>0.5</ns0:AirChangesPerHour>
            <PeopleNumber unit="NumberOfPeople">0.2</ns0:PeopleNumber>
            <PeopleHeatGain unit="WattPerPerson" heatGainType="Total">90</ns0:PeopleHeatGain>
            <LightPowerPerArea unit="WattPerSquareMeter">0.5</ns0:LightPowerPerArea>
            <EquipPowerPerArea unit="WattPerSquareMeter">3</ns0:EquipPowerPerArea>
         </Space>
         ...
      </Building>
      ...
   </Campus>
</gbXML>
```

Figure 2: Excerpt of the final gbXML file for one of the 72 base case buildings, in this case the detached house with an orientation (CADModelAzimuth) of 0°.

MODEL DEVELOPMENT

72 EnergyPlus simulations were then run on the 72 detached house variants, based on the 72 gbXML files as described in the previous section (EnergyPlus, 2018). Converting gbXML to the EnergyPlus input data files (.idf files) is a BIM-to-BEM process, where BIM refers to Building Information Modelling and BEM refers to Building Energy Modelling. As EnergyPlus idf files require more detailed information that is available or can be stored in the gbXML file (such as simulation time step and additional material properties), any BIM-to-BIM approach will require assumptions to be made about a number of the idf parameters. Different tools or processes may make different assumptions during the conversion and it is important that these assumptions are available for inspection.

Here the ‘EnergyPlusModel’ class in the openbuilding package is used to import the gbXML files, create the EnergyPlus idf files and run the EnergyPlus simulations. The results are saved as the standard EnergyPlus file set, with the time series results recorded in the .eso and .csv files. The assumption made can be viewed by looking at the mapping methods in the code for different gbXML element types (‘Building’, ‘Space; etc.) and by visually inspecting the resulting idf files. The simulations were run for a full year and were based on the Birmingham, UK, weather data available for download from the EnergyPlus website (the epw file ‘GBR_Birmingham.035340_IWEC.epw’).

A new data structure is also created at this point. The building information in the gbXML files and EnergyPlus results are combined into a graph data structure, using the ‘BIMGraph’ class (i.e. a Building Information Model graph) in openbuilding. This graph is created for use in the analysis of the simulation results as a well-formed and logical data structure that holds input and output data of the simulation process. It also provides a software-neutral format for storing building information data that could potentially enable different simulation engines to be run on the same dataset.

Figure 3 shows a visualisation of the BimGraph for one of the building variants, the detached house with a building orientation of 0°. This is a graph data structure. Graphs, in computer science, are a data structure that consists of nodes (entities that can contain properties) and edges (the relationships between the nodes). In Figure 3 the nodes are shown as coloured circles and they represent the objects that make up the detached house. Objects such as ‘Building’, ‘Space’, ‘Surface’ and ‘Opening’ are shown. The edges are shown by lines and they represent the relationships between the objects. So, for example, if a room is bounded by 6 surfaces (such as 4 walls, a floor and a roof) then this is shown on the graph as a Space node with edge connections to 6 Surface nodes. This provides a strong logic in the data structure aligning with the relationships found in the real world between these different objects. The well-structured and logical data represented in the graph means that the computer code and analysis algorithms written to develop insights and findings from the dataset become simpler to write and easier to read, with relatively few lines of code needed in many cases.

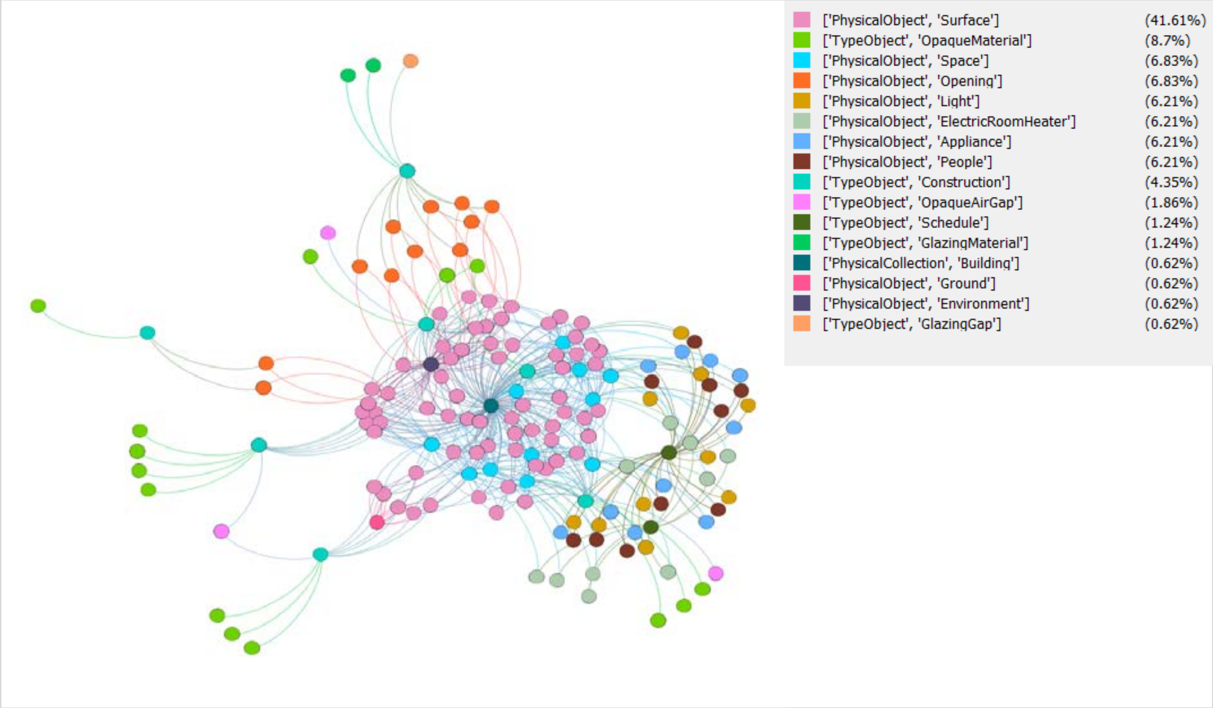
['PhysicalObject', 'Surface'] (41.61%)
['TypeObject', 'OpaqueMaterial'] (8.7%)
['PhysicalObject', 'Space'] (6.83%)
['PhysicalObject', 'Opening'] (6.83%)
['PhysicalObject', 'Light'] (6.21%)
['PhysicalObject', 'ElectricRoomHeater'] (6.21%)
['PhysicalObject', 'Appliance'] (6.21%)
['PhysicalObject', 'People'] (6.21%)
['TypeObject', 'Construction'] (4.35%)
['TypeObject', 'OpaqueAirGap'] (1.86%)
['TypeObject', 'Schedule'] (1.24%)
['TypeObject', 'GlazingMaterial'] (1.24%)
['PhysicalCollection', 'Building'] (0.62%)
['PhysicalObject', 'Ground'] (0.62%)
['PhysicalObject', 'Environment'] (0.62%)
['TypeObject', 'GlazingGap'] (0.62%)

Figure 3: Visualisation of the BimGraph object for the detached house building at 0° orientation. This image is created using the Gephi software (Gephi, 2018).

RESULTS – SUMMERTIME TEMPERATURES

This section analyses the results of the 72 BimGraph objects that were created using the methods described above. The 72 BimGraphs hold the gbXML input data and the EnergyPlus results data (i.e. the time series data available in the .eso files) for each of the 72 building variants. The analysis was fully written in computer code and the algorithms documented to support the Open Science approach. This was done using Python code within Jupyter notebooks, and the final notebooks are made available on GitHub (uSim_2018, 2018).

Figure 4 shows the hourly mean internal air temperatures for the 72 building variants. The hourly air temperatures are shown for a single day, July 7th, to represent the performance of the buildings on a typically hot summer day. For each of the 72 building orientations, the internal air temperatures are shown in a different coloured line with the colours based on a gradient relating to the angle from due south. For a building orientation of 0°, the living room is located on the south side of the house with a window looking due south. Thus the living temperatures in Figure 4 appear higher when the building itself is orientated toward the north (orientation = 0°) and lower when the building is orientation to the south (orientation = 180°), due to the increase in solar heat gains through the living room window. The peak temperatures occur at around orientation = 50° or orientation = 310°. This is likely because the solar radiation may strike the window at a smaller angle of incidence at these orientations, thereby increasing the solar heat gain through the window.

Figure 5 shows the maximum hourly internal air temperatures for the different rooms in the base case building. This is the highest hourly internal temperature from the 8760 internal air temperatures that were predicted throughout the year. The maximum hourly air temperature values are plotted against the building orientation for the 72 building variants. Each room in the building is plotted as a line with a different colour. The results show that the dining room has the highest maximum internal air temperatures of all room (around 35°C) and these occur at building orientations around 180°. This occurs as the dining room has a large glazed opening (a set of patio doors) which face due south at this building orientation. The roof has the lowest maximum internal air temperatures (around 26°C) which are relatively independent of orientation. This occurs because the roof space does not have a window and thus no solar gains that could be impacted by building orientation. Figure 5 shows the potentially large impact that building orientation can have on the maximum air temperatures, and thus the overheating, of different building spaces.

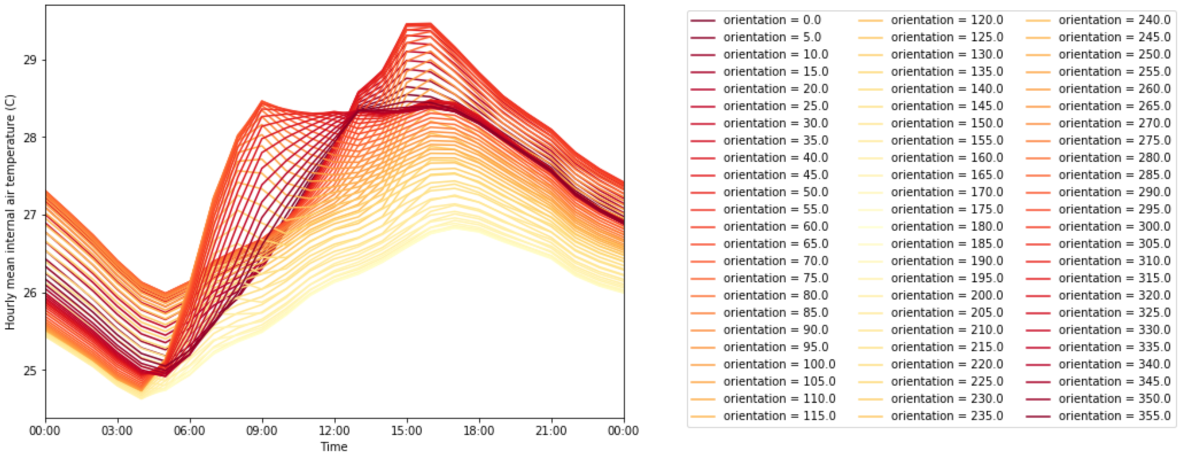

Figure 4: Living room hourly mean internal air temperature for the base case building with varying building orientations for a hot summer day (July 7th)

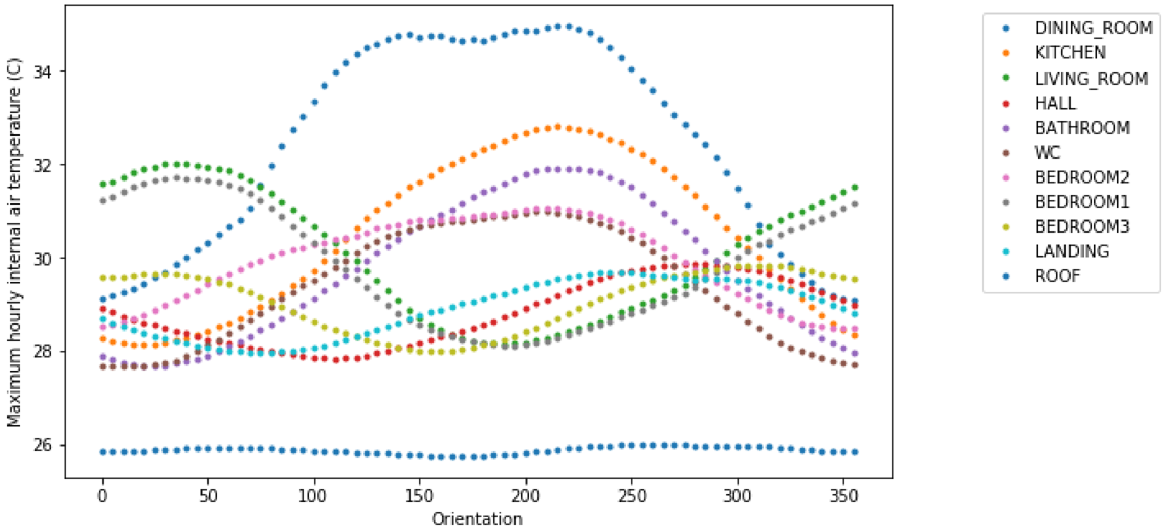

Figure 5: Building orientation vs. maximum hourly internal air temperature (i.e. a single value for the year) for each room in the base case building

The results here are intended as examples of the potential of the Open Science workflow. The Python code and Jupyter notebooks used to generate Figure 4 and Figure 5 are provided on GitHub and can be used as a reference by other researchers to both replicate these plots or to adapt the methods used for new research studies (uSim_2018, 2018). It should be noted that due to the well-structured and logic data of the Open Science workflow, both these plots can be created in less than 20 lines of Python code.

DISCUSSION

There are clear benefits to the building stock modelling community from working in an Open Science manner. Tool and methods developed from one stock model could be easily shared and reused by other researchers, thus reducing much of the duplication of effort that currently exists in the field. New researchers to the area could quickly learn and understand how the models are working, and have more time to advance the field itself. Open Science also improves the robustness of the models themselves and the results generated from the models' use. Any errors or omissions may be detected by the wider community and be quickly corrected. It is important that the results of stock modelling studies, which are often intended for informing national policy, are as error-free as possible.

However, a number of challenges to Open Science stock modelling still exist. One of the main limitations is in the availability of the original datasets on which the models are built. Given that stock modelling researchers will rarely collect their own datasets (as this would involve large-scale and costly surveys), most stock models are constructed on existing, third-part datasets. If the modellers have accessed a third-party dataset through an exclusive arrangement that is not available to the wider research community, then their models and results cannot be understood, tested, reused or improved upon by others. The ideal solution here is to both campaign to improve the accessibility of third-party datasets (at least so they are available to the research community) and to use non-exclusive datasets wherever possible.

CONCLUSIONS

This work has demonstrated an Open Science workflow for building stock models. A case study presented the results of the workflow using an example of 72 detached house models. The workflow used gbXML files as an Open Science dataset and used EnergyPlus to run simulations on the house models. The workflow was automated using the openbuilding Python package.

The key conclusions of this work are:

1. It is possible to develop and run building stock models according to Open Science principles. This paper presents one approach to an Open Science building stock model workflow (using gbXML, openbuilding and EnergyPlus) but other approaches are also possible.
2. The gbXML data format provides an industry-standard platform-neutral approach to sharing building information that is well suited for Open Science building stock models.
3. Python scripts and Jupyter notebooks, particularly if they are hosted in a public repository such as GitHub, can provide a clear, understandable, replicable and reusable record of the analysis algorithms used in an Open Science building stock model. These can then be easily reproduced or enhanced by other researchers.

REFERENCES

Allen, E. and Pinney, A. 1990. Standard Dwellings for Modelling: Details of Dimensions, Construction and Occupancy Schedules. Building Environmental Performance Analysis Club (BEPAC) Report, Building Research Establishment, Watford, UK

EnergyPlus. 2018. EnergyPlus is a whole building energy simulation program that engineers, architects, and researchers use to model both energy consumption, https://energyplus.net/

Firth S.K., Lomas K.J. and Wright A.J. 2009. 'Targeting household energy-efficiency measures using sensitivity analysis', Building Research & Information, 38: 1, 25 — 41, http://dx.doi.org/10.1080/09613210903236706

gbXML. 2018. gbXML: an industry supported schema for sharing building information between disparate building design software tools, available at http://www.gbxml.org/

Gephi. 2018. The Open Graph Viz Platform, https://gephi.org/

GitHib, 2018. The world's leading software development platform, https://github.com/

Project Jupyter. 2018. Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages, http://jupyter.org/

Kraker P., Leony D., Reinhardt W. and Beham G. 2011. The case for an open science in technology enhanced learning, Int. J. Technology Enhanced Learning, Vol. 3, No. 6, pp 643-654

openbuilding. 2018. https://github.com/openbuilding/openbuilding

Python. 2018. The official home of the Python Programming Language, https://www.python.org/

Royal Society. 2012. Science as an open enterprise, The Royal Society Science Policy Centre report 02/12, ISBN: 978-0-85403-962-3.

uSim_2018, 2018. Analysis for the uSim 2018 conference - Urban Energy Simulation - Glasgow, 29-30 Nov 2018, GitHub repository, https://github.com/stevenkfirth/uSim_2018