DEFENDER : HOUSE LEVEL TOOL DEVELOPMENT (D1.3-6)

DE04 – Detailed functional/technical specification

As Built Review December 2022

V 2.3

carbontrust.com

+44 (0) 20 7170 7000

Published in the UK: 2022

Revision History

Name	Notes	Author	Date
1.0	Initial framework	Jane Wilson	30 May 2022
1.1	Reviewers' comments incorporated	Joshua Cooper	12 August 2022
1.2	Additional review comments incorporated	Joshua Cooper	16 August 2022
1.3	Minor changes	Ben Robertson	17 August 2022
2.0	Final for submission	Jane Wilson	18 August 2022
As built update 2.1	Commentary on actual delivered vs original specification	Faye Schuster	21 Decemeber 2022
As built update 2.2	Updates to above	Faye Schuster	23 February 2023
As built update 2.3	Updates	Faye Schuster	8 March 2023

Authors

Name	Position	Date
Joshua Cooper	Hildebrand	30 May 2022

Approvals

Name	Position	Date
Nick Devine	WPD – Innovation Engineer	27/03/23

Table of Contents

Abbreviations

Term	Description
DFES	Distribution future energy scenario
DNO	Distribution Network Operator
DNOA	Distribution Network Options Assessment
EE	Energy efficiency
EHV	Extra High Voltage
ESA	Electricity Supply Area
EUI	Energy Use Intensity
GUID	Global Unique Identifier
HTC	Heat Transfer Coefficient
LV	Low Voltage
MDI	Maximum Demand Indicator
sFTP	Secure File Transfer Protocol
SMETS	Smart Metering Equipment Technical Specifications
SQL	Structured Query Language
DFES	Distribution future energy scenario

1. Introduction to the 'As Built' Update

As part of project completion, Hildebrand have performed a review of the initial Detailed Functional Technical Specification to compare original intention versus final, approved, deliverable.

Commentary is added against any element that varied from original intent. Where the final version has replaced what was in the original specification, the content of the original specification has been deleted.

To distinguish the commentary it is in dark red, prefaced by 'As built'.

2. Introduction

2.1. Purpose

Work package 1.1 of Workstream 1 of the DEFENDER project develops the capability for simulating historical and future power demands, taking into account different energy efficiency measures.

A database of domestic heat pump operational and performance characteristics will be created and combined with the inputs below to create a model which will then be trained on Hildebrand's datasets. The model will be capable of deriving before and after building fabric retrofit heat demand and quantify this as a historical and future ADMD and load profiles, including transference of gas demand to electricity demand through the heat pump performance dataset.

Inputs to the model include:

- Hildebrand's smart meter database of domestic gas and electricity consumption data
- Heat Transfer Coefficients (HTC) from the SMETER project which used BEIS validated algorithms to calculate building thermal performance from smart meter data
- Carbon Trust's Building Decarbonisation Options Appraisal Tool, which can determine the impact of energy efficiency and heating measures on the building's heat loss factor (U value)
- The Energy Performance of Buildings Register, which contains EPCs for homes in England and Wales, which include heating and fabric data amongst other useful information
- Local weather data from weather stations sourced from METAR feeds

2.2. Background

Document D0.1 Profiling tool specification and design document, signed off by Nick Devine of WPD on 23 May 2022, identified the data sources and flows to be used in the profiling tool and underpins this detailed functional and technical specification.

That document has been updated as part of this workstream and is re-issued at D1.1-2 Revised scoping document and high-level tool specification.

This document is the detailed functional and technical specification for the House Level functions; this activity and its outputs will inform the second part of tool development for network planning (D1.2-1 Network planning detailed functional and technical design).

There are two end user tools that will have user interfaces:

1. Household Profiler – the tool specified in this document that creates and maintains archetypes
2. Network Planner – the tool (covered in a separate detailed specification) that applies household archetypes as a population representing the homes that are attached to a segment of the distribution network.

2.3. Objectives

For the design of the tool the high-level objectives that are taken into consideration are:

1. To produce representative house level load profiles used in future demand scenarios
2. Parameterise model to consider changes in load profile based on energy efficiency measures
3. Heating demand to be captured and the translation into electricity demand based on heating technology
4. Separate representation of electricity base load for the house profile
5. Save both the resulting data file and the parameters that generated the data and export to csv
6. Ability to update the models in the future

2.4. Assumptions

The following are the subset of project assumptions relevant to the technical design at the present time.

Assumption status:

- Identified – The assumption has been identified but has not been validated.
- Validated – The assumption has been validated. The validation source and validation date should be stated.
- Dismissed – The assumption is not valid or is no longer relevant.

ID	Description	Status
A-1	Data coverage is good over all of the archetypes	Validated
A-2	Weather stations that are used are representative of the conditions experienced at the meter point	Validated
A-3	Time period 2019 – pre-Covid; largely office working. Time period 2020 - 2021 – Covid: largely home working Time period 2022 - Covid new normal: 3 days a week office working	Validated
A-4	Data coverage between EPC and smart meter properties is good	Validated. 28/04/22
A-5	Retrofit measures are those available today, there is no forecasted change in insulation or window technology	Identified
A-6	Buildings fabric will not regress once a retrofit modification is made, e.g. insulation lasts for the forecast window	Validated
A-7	Half hourly energy (kilowatt hours) accurately represents the power demand for that period (kilowatts)	Identified
A-8	Cooling is not included in this model	Identified
A-9	Retrofit measures have been installed to best practice standards	Identified

2.5. Dependencies

The following are the subset of project issues relevant to the technical design at the present time.

Assumption status:

- Open – The issue has been identified
- Resolved – A dependency has been met

ID	Description	Impact	Status
D-1	Model to be stand-alone (WPD IT environment restrictions)	No server required	Resolved
D-2	'As built': Added post completion. Model outputs to be delivered in csv, json and Sincal format (see Item 35, Appendix 2, D1.3-7 UAT report)		Resolved

2.6. Risks and Issues

The following are the subset of project issues relevant to the technical design at the present time.

Assumption status:

- Open – The issue has been identified
- Resolved – A resolution has been accepted for the issue

ID	Description	Status
I-1	Diversity can be captured when generating load profiles or at a later stage before the profile is loaded into scenario analysis. Design needs to be clear when diversity is considered.	Open
I-2	Parameters are stochastic and therefore multiple running of the tool will not always give the same result.	Open
I-3	Weather assumptions for long term predictions may need to consider changing climate	Open

2.7. Design Decisions

The following table lists design decisions and their rationale.

ID	Description	Impact	Rationale
DD-1	Single page interaction for web that can be run locally in a browser without an Internet connection	UI framework does not need to consider mobile interactions or browser connecting back to a server	Used in browser environment to export files locally. Local loading of saved settings so that a server is not required.
DD-2	Where possible, easy to use selection criteria with ranges are used for describing house parameters.	May lose some fine grain control of model parameters	Complicated technical input is probably difficult for most end users of the tool, assumptions will have to be used to translate from high-level combination to the reference values. As a part of UAT, the typical user will give feedback as to the suitability of the selection criteria – i.e. are they easy for those users to use
DD-3	Results downloaded to files that are stored locally	Portable; sharing must be done with the files	No server

3. Model Inputs

This section explains the data input assumptions and the technical design of the model and generation of data to be exported.

Models can also be thought of as the archetype, along with the inputs that drive the generation of profile data. Options that are selected can be saved, along with the sample profile, so that different models are formally defined.

3.1. Smart meter data

Model data input was collected from a dataset of approximately 5,140 homes from across the UK. The dataset consists of:

1. Measured energy consumption: Electricity and gas meter readings from UK SMETS meters at 30-minute intervals; date ranges are from 2018 – 2022.
2. Building properties: Descriptions of buildings from EPC for the properties being metered. This provides building size, fuel type and existence of low carbon technology at the time of assessment. Not all buildings have current EPCs so there will be low thousands of buildings where we will have that information.
3. Weather data: An hourly weather file based on data recorded from the nearest weather station to the metering point, the hourly data will be resampled to 30-minute data

Smart meter data contains an address in the form of street, city, postcode and in most cases a UPRN (unique UK property reference) that is joined to EPC data and used to find the nearest weather station. UPRNs will be latitude/longitude geocoded for finding the nearest weather station using a simple shortest distance calculation.

3.1.1. Measured energy consumption

Energy consumption is used to train the model. Electricity consumption from smart meters is used to establish the base load for estimating the future total electricity demand. Gas consumption is used to estimate the heat transfer co-efficient (HTC) that, when combined with the selection of heating technology, will be additional electricity demand.

From the data set available, the time periods where high-quality data is available is from late 2019 up to Spring 2022. Figure 1 and Figure 2 show coverage of the energy consumption, broken down by property type and EPC rating.

In the data, the early periods show that not all house archetypes are represented. This is due to fewer meter installs/data capture for that time period.

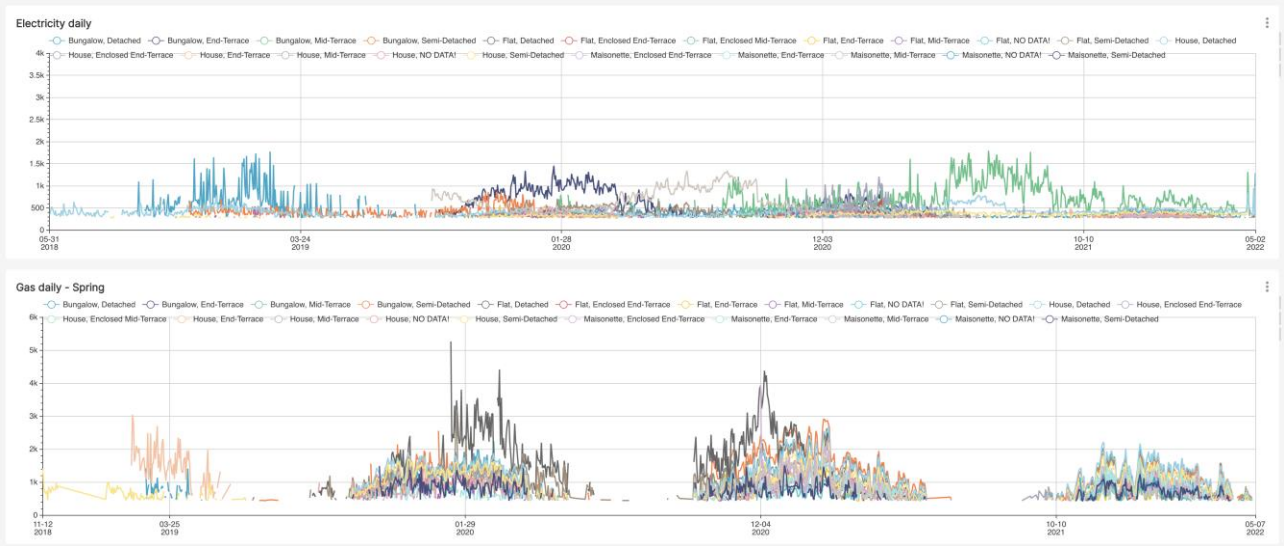

Figure 1. Source smart meter data (daily energy in kWh) for gas and electricity, sufficient population for analysis is in the 2019-2020, 2020-2021 and 2021-2022 heating seasons

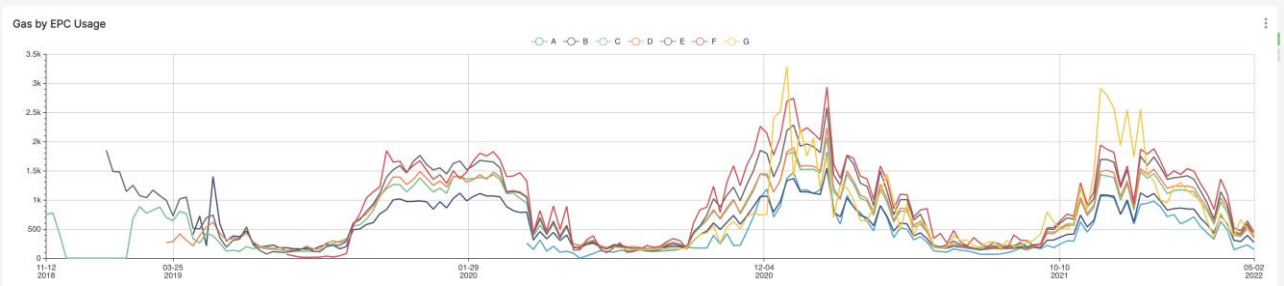

Figure 2. Average gas use by EPC rating, showing poorly rated homes with more consumption of energy

3.1.2. Building properties

From the EPC data, there are dimensions defining a property. The model will be trained on these dimensions so that they can be used to select the household building properties for the generation of new data. The smart meter data model can be sampled on like for like dimensions.

Built Form	Property Type	Construction Age Band			
'As built' Final building properties, as that's the data we had access to – selected in tool –'house parameters' by user as below:					
• End-Terrace • Mid-Terrace • Semi-Detached • Bottom Floor Flat • Mid Floor Flat • Top Floor Flat	• House • Flat	• before 1930 • after 1930	**Floor Type:** • Solid • Suspended • Other premises below	**Roof Type:** • Flat • Pitched • Other premises above	**Wall Type** • Cavity Wall • Solid Wall

Table 1. Selection criteria for building properties

3.1.3. Weather data

Each of the smart meters is found in a particular location. The weather station latitude and longitude is geocoded to be able to join to the smart meter location for training the models. As a reference to what has been used to train the models, the 99 automated surface observing system (ASOS) locations that provided historical weather data are found in Appendix A.

When selecting location for data generation, a Local Authority will be used rather than a station ID. The nearest weather station servicing that Local Authority will be in a data lookup. From that weather station identifier, the historical data will then feed the model to generate temperature profiles. In the future, that lookup can be replaced with typical meteorlogical year (TMY) values. Note, the functionality allows for specific temperature values to be supplied instead if there are more specific data required.

3.2. Determining house archetypes

The homes in the dataset will be grouped into house archetypes based on the characteristics provided in the EPC data shown in section 3.1.2. For each house archetype, three Bayesian calibrations will be completed.

1. **HTC:** Bayesian calibration for HTC is accomplished through the selection of a prior probability distribution that we believe to represent HTCs that we would find across the building property

categories. For instance, we would assume a normal distribution (or Gamma distribution for the case of only positive values) with a middle point for the mean, a wide variance and constrain the minimum and maximum to plausible values. Data then is presented to the prior distribution and the model function to estimate a posterior distribution based on a Markov Chain Monte Carlo or Hamiltonian Monte Carlo sampling algorithm. The execution of these steps requires a probabilistic programming library that carries out the posterior estimation; we will use PyMC for this. With PyMC, other methods of calculating the posterior can be tried for faster computation, for instance variational inference which may be required as the data set grows in time.

2. **Heat demand profiles:** Based on the gas smart meter data available for each house type, a heat demand profile can be calculated, detailing the timing of heat demand throughout the day. A similar approach will be followed to the HTC calibration, utilising PyMC.

3. **Base electricity load profiles:** There should be some homes that already have electric heating as well as large loads like EV and unusually low loads due to PV within the sample population. We will run multiple models to detect and segregate homes that have these characteristics before determining normal base loads. A similar approach will be followed to the HTC calibration, utilising PyMC.

Convergence will be based on how closely the model represents the data through an error rate. The error limits will be selected for calibration based on literature at 5%. HTC error will be assessed on the basis of energy use intensity, or EUI, calculated as the sum of gas consumption in kilowatt-hours per square meter of floor area. Likewise base load of electricity will have a percentage error of electricity consumption in kilowatt-hours per square meter of floor area.

3.2.1. Heat transfer co-efficient (HTC)

The HTC calculation is more formally represented by the following:

$$Q_{day} = htc \cdot \max(T_s - T_{ext}, 0)$$

Equation 1 Relationship of gas energy to temperature with the heat transfer co-efficient expressing the heat loss of the building fabric

The observed gas heating energy, Q_{day} is a function of the external temperature T_{ext} and the desired internal temperature, T_s (or set point for a thermostat). The external temperature T_{ext} is also an observed value, provided by the weather data above. The random variable htc or heat transfer co-efficent is the amount of energy required to raise the temperature by one degree. Therefore its units are kWh / K. T_s is also considered a random variable as the set point is not directly observable.

For each building archetype, the heat loss co-efficient and temperature set point will be probability distribution functions.

Using daily energy values and average temperatures for the day in the model training step is thought to be more robust as it smooths out lag in the heating system.

Note, when the Q*day* is generated, the T*ext* profile (48 values) is used to produce the half-hour energy reading which differs from the daily values used in the training step.

3.2.2. Heat demand profiles

Heat demand in the model was trained to be at the time the energy is used, therefore a direct interpretation of the gas consumption timeseries defines the shape of the heating requirement through the day. The estimated demand profile is a structural time series, generated by the heat demand and heating technology with each 30-minute period being represented.

Seasonal differences are represented in the external temperature. For data generation, it is assumed that it will be warm enough (above the temperature setpoint) to not trigger heating demand. The volume of the demand is captured in 3.2.1. using the HTC as the amount of energy required to raise the temperature.

Any occupancy assumptions (estimated from the base electricity load) will be available to the heat demand profile logic, however they may not be considered at this time.

3.2.3. Electricity demand profiles

Electricity demand in the model is observed, therefore influences on electricity consumption are the hidden variables that can be estimated. Base load (overnight or stand-by load), occupancy (use of appliances) and number/type of appliances used are not directly observed. The relationship is:

$$E = M_{app} \cdot occ + E_{base}$$

With three random variables, M_{app} as the appliance multiplier, occ is an occupancy factor and E_{base} is the contribution of the base load.

The seasonal issues with this data are from changes of occupancy (more time inside during winter) and use of electrical appliances (lighting increase in winter). Air conditioning has not been modelled.

3.3. Heating technology

To model the electricity requirements of the heat pump, a performance model of kWh / K along with an ideal kW running power is used. This will determine the duration of runtime to meet the heat demand as a saw tooth curve with constant running power while in use. From the heat demand profile and the HTC, the energy requirement for a given 30-minute period can be determined.

More advanced models could be used, such as Energy Plus simulator, but that would require additional modelling parameters that may not be available, as well as a server environment to run the Energy Plus software.

The important assumption here is that the heat pump electricity demand is deterministic based on the heating demand calculated in section 3.2.2.

Other heating technologies can be considered - direct electric and night storage can be expressed.

3.4. Carbon Trust efficiency model

The efficiency model is driven by a set of building fabric parameters, that when combined express the overall HTC of the building. For this model low level, the following can be changed:

Floor insulation – range of values that translate into change in W/m^2*K for the U value

Glazing – range of values, single, double, triple, etc that translate into change in W/m^2*K for the U value

Roof insulation – range of values that translate into change in W/m^2*K for the U value

Thermal bridge factor – range of values that translate into change in W/m^2*K for the U value

These values are applied to the size of the property and assumed wall to window ratios to impact the overall HTC. The new HTC is used to inform the heat demand volume with the demand profile.

For clarity, these values are changed to run a future scenario, for example, a home that has new roof insulation installed where it was not before. Note, information on the above parameters can be determined from EPC data.

4. Model outputs

'As built': Models renamed to 'Profiles' in the final tool, as requested by Carbon Trust

A set of outputs from the house-level modelling will be delivered that will include the following items.

4.1. Demo profile output from historic data (D1.1-1)

An aggregated profile of one or two feeders created from Hildebrand's historic smart meter data, for validation against WPD data to sense check Hildebrand's database.

The demand profile will visually look like **Error! Reference source not found.** with the ability to export as W att hours. For instance the peak value on the graph is 343 Wh or 0.343 kWh for that half hour period. The assumption is that the peak for the half hour is the maximum, which we believe is appropriate when the tool is being used to analyse across a number of homes.

The CSV file will have a header row with the field definitions and then can have rows underneath with the model reference (GUID linked to model) and date (example of time of year). An example output is:

'As built': **Final csv ouput**

- Heating and base consumption values added as separate rows so the data can be seen separately
- Optional archetype parameters can be included in CSV- see Item 41, Appendix 2, D1.3-7 UAT report

date,guid,name,consumption_type,units,00:00,00:30,01:00,01:30,02:00,02:30,03:00,03:30,04:00,04:30,05:00,05:30,06:00,06:30,07:00,07:30,08:00,08:30,09:00,09:30,10:00,10:30,11:00,11:30,12:00,12:30,13:00,13:30,14:00,14:30,15:00,15:30,16:00,16:30,17:00,17:30,18:00,18:30,19:00,19:30,20:00,20:30,21:00,21:30,22:00,22:30,23:00,23:30

Wednesday/Week 4/January, 6a18bb3f-9ce9-4446-b44c-9a3bd770a67f, model-example, Base Consumption, wh, 291.53813732735944,232.6954173258825,220.2916306708184,226.7556663894735,237.31128846810333,228.86699822579737,224.7394336936557,172.02513867585748,228.96150851799365,166.03089400404377,195.7766999034848,241.31541232025938,247.5023428167711,309.51584899369357,346.9124657568763,333.7276162556949,341.51892652796994,328.1566177049101,380.7345563265486,318.44939827347474,334.82690699130274,331.4709318470128,337.35194097053613,333.26536328786545,400.9106041679176,315.4999496899122,345.63818988739763,344.26607800256505,342.97903217476176,296.30882376765015,318.17544764110625,335.10580613527003,357.09365068928446,387.47296635807004,509.7299959952825,505.10425397468913,543.2546197384329,501.7028534939988,500.6854447753804,457.66312962907017,458.5035569994624,426.4534423101559,485.96234357283396,371.5845549936043,426.73610240185747,334.6112931680112,317.57075518864775,256.66421721754455,

Wednesday/Week 4/January, 6a18bb3f-9ce9-4446-b44c-9a3bd770a67f, model-example, Heating Consumption, wh, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

Wednesday/Week 4/January, 6a18bb3f-9ce9-4446-b44c-9a3bd770a67f, model-example,Temperature, ˚C, 5.2,5.2,5.1,5.1,5,5,5,5,4.9,4.9,4.9,4.9,4.8,4.8,4.8,4.8,4.8,4.8,5.2,5.2,6,6,6.8,6.8,7.3,7.3,7.7,7.7,7.7,7.7,7.6,7.6,7.3,7.3,6.8,6.8,6.4,6.4,6.2,6.2,6.1,6.1,6,6,5.8,5.8,5.7,5.7,

Figure 3 'As built': Sample export CSV file from a run of the model tool

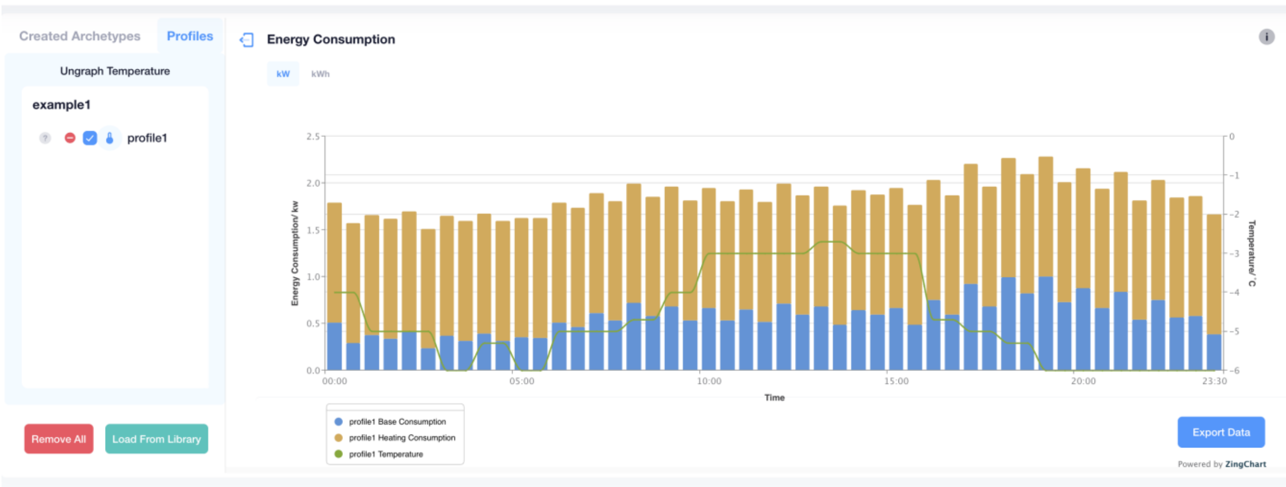

Figure 4 'As built': Final data output

'As built' Final Profile graph outputs (Fig 4)

- Base consumption and heating consumption is now shown on the graph as well as temperature – an improvement we thought to show heating loads from the base loads.
- Hovering over the legend at the bottom allows the user to see the values of consumption.
- Temperature can be viewed by selecting the thermometer symbol next to the Profile in the graph window

4.2. Models for house archetypes

The tool will allow for the run for a house archetype with that archetype being able to be named and saved. At a high level the archetypes are comprised of:

- Building characteristics
- Selection of electric heating technology (e.g. ASHP or storage heater)
- Base heat demand profile (mean and variance)
- Non-heating electricity load profile (mean and variance)
- Heating electricity load profiles for the base scenario and with each energy efficiency measure and each agreed combination of measures (mean and variance).

These will be provided as comma separated value (CSV) files with file names or columns describing the building archetype, retrofit measure applied (if any) and the identifier of the representative day. Load profiles will be specified based on:

- Weekend or weekday
- Season - Winter, Summer, Intermediate Cold and Intermediate Warm
- Half hour time period

In all cases there will be 48 data points per day expressed in average kilowatt per half hour.

The detailed model parameters will be captured and saved. They can be loaded back in the tool as well. The file format is JSON and will have a structure that is an array of models so that all the models in a “library” can be saved, shared and loaded.

'As built': Below is an example of final output. Final JSON output changed to:

- Base consumption and heating consumption values separated


```
"models": [
        {
        "guid":"6a18bb3f-9ce9-4446-b44c-9a3bd770a67f",
        "name":"model-example",
        "archetypeGuid":"d30715da-ee0f-47dc-9842-c97beb01d966",

        "weather":{
                "date": {
                "day":"Wednesday",
                "month":"January",
                "week":"Week 4"
                },
                "temperature":[
                5.2,5.2,5.1,5.1,5,5,5,5,4.9,4.9,4.9,4.9,4.8,4.8,4.8,4.8,4.8,4.8,5.2,5.2,6,6,6.8,6.8,7.3,7.3,7.7,7.7,7.7,7.7,7.7,7.6,7.6,7.3,7.
                3,6.8,6.8,6.4,6.4,6.2,6.2,6.1,6.1,6,6,5.8,5.8,5.7,5.7
                ],
                "location":"Ashfield"
```

},
"data":{
"baseConsumption": [
291.53813732735944,232.6954173258825,220.2916306708184,226.7556663894735,237.31128846810333,228.86699822579737,224.7394336936557,172.02513867585748,228.96150851799365,166.03089400404377,195.7766999034848,241.31541232025938,247.5023428167711,309.51584899369357,346.9124657568763,333.7276162556949,341.51892652796994,328.1566177049101,380.7345563265486,318.44939827347474,334.82690699130274,331.4709318470128,337.35194097053613,333.26536328786545,400.9106041679176,315.4999496899122,345.63818988739763,344.26607800256505,342.97903217476176,296.30882376765015,318.17544764110625,335.10580613527003,357.09365068928446,387.47296635807004,509.7299959952825,505.10425397468913,543.2546197384329,501.7028534939988,500.6854447753804,457.66312962907017,458.5035569994624,426.4534423101559,485.96234357283396,371.5845549936043,426.73610240185747,334.6112931680112,317.57075518864775,256.66421721754455],
"heatingConsumption":[
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
]}
}
]

Figure 5 'As built' Actual model library file with saved parameters

5. User interface

The DEFENDER tool is used by a WPD employee to interactively generate one day of half hourly profile reads for electricity consumption, specifically 48 data points. The tool visually shows the data in the form of a graph. Up to 3 output profiles can be visually compared as separate series on the graph.

'As built': More than 3 output profiles can be visually compared on the graph in case users want to view more than 3.

Models can be saved and loaded as files, see 4.2 for detailed JSON format and data series exported for use in modeling tools as CSV. Model runs can be done for longer periods of time or with new temperature assumptions i.e. one year, although they are not visualized.

This section presents how a user would interact to input the parameters of the model and produce output files.

5.1. Single page application

The user interface will run in a web browser with four main sections.

Models – the list of models, using the name given by the user if an existing archetype was used. Up to three models can be selected, which will generate series on the graph and be visualised.

Data – visual presentation of the data (48 half hourly data points) and control of the export of that data; up to 3 series can be shown side by side; raw data can be imported from a CSV if there is a need to compare.

Generating a New Archetype [Options] – the sub-areas of the options direct the input of assumptions, first house parameters, second weather parameters, third heating technology and fourth the energy efficiency parameters. Generating a graph will create a new working series on the graph, it can be visually compared with the other saved models. Once the model is complete it can be saved in the library for future recall. A new entry on the model list is made as a result.

Model Library (separate section from top-level navigation) - ability to select pre-existing archetypes that act as presets and the weather assumptions (saved temperature profiles) that were used to generate them. An exploration page that helps organize models and search based on characteristics.

'As built' : Final outcome – Profile Library and Archetype Library are separate pages to manage Profiles and Archetypes separately.

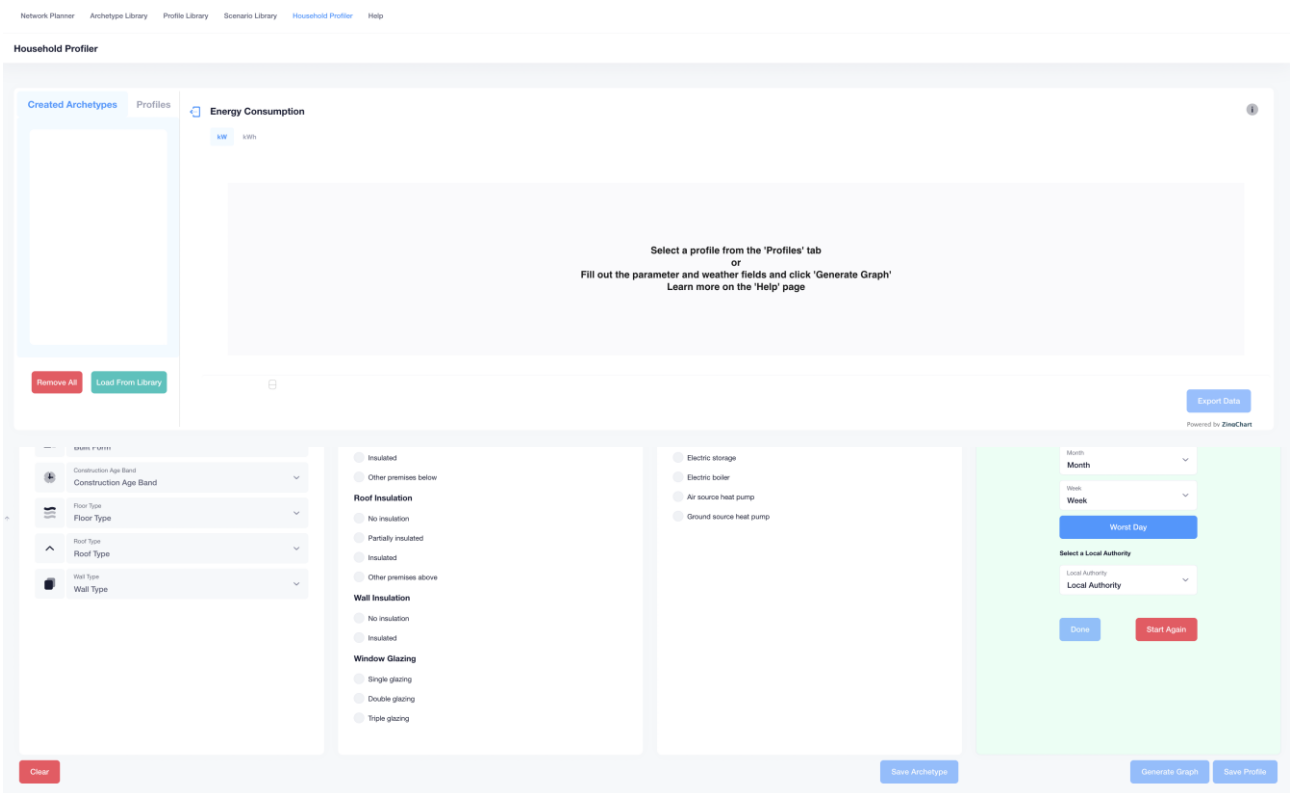

Figure 6 'As built' Final main user interface for the house profile tool, delivered as a single page web application

'As built': Final outcome of Household Profiler (Fig 6)

- Save Archetype button added to save Archetype to the library without generating a Profile. Archetypes from this library can be loaded into the Profiler and will be under 'Created Archetypes'
- Separate sections for 'Created Archetypes' and 'Profiles'. Profiles can be made from the 'Created Archetypes' section by selecting the Archetype under 'Created Archetypes' and adding weather parameters and saving.
- Worst Day can be selected to select the coldest day in 20 years - button added for ease

5.2. Running the model

The tool is built from a series of modules, each relating to an input parameter group. This approach will allow the underlying logic within each module to be updated if required without impacting the other sections of the tool.

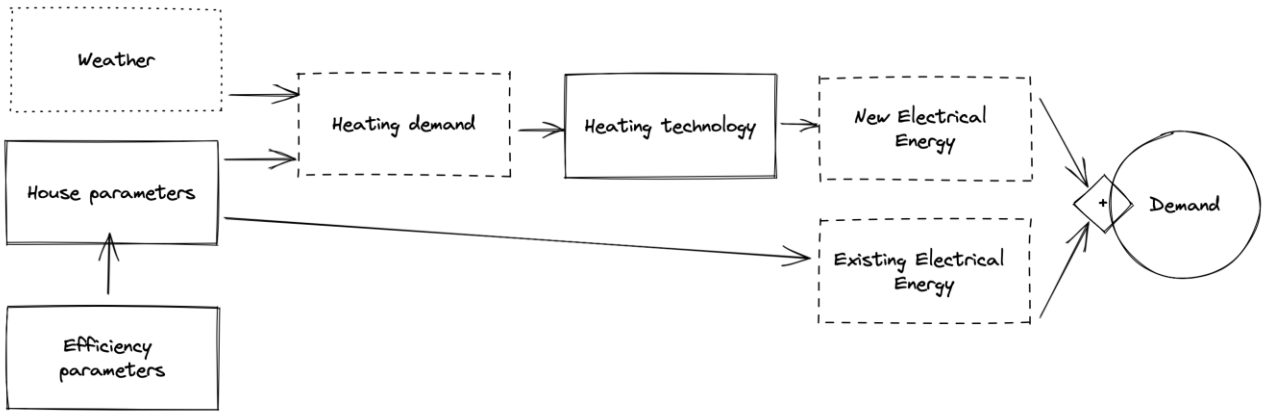

Figure 7. Flow diagram of data generation, the solid lines indicate direct control of parameters, light dashed (weather) is somewhat controlled by date and location selection, and heavy dashed are model computations that have the assumptions built in

5.3. User Steps

The user may want to use the tool to create new archetypes, recall previous models or run an archetype with different temperature profile assumptions.

If the user wants to look at a model that is in the model library, then it can be selected from the model menu. Upon selection, the graph will show the profile and the parameters that have been set so that they can be inspected.

'As built': In the final deliverable the user should load the models from the Profile library into the Household Profiler to view them. This is so the user can manage the Profiles in the library and choose what to display in the graph/graph window if they have a large number of Profiles in the library.

If the user is creating a new archetype/model, they must fill in values for the options section. Each section is like a step in creating the model where the step is shown as completed once the mandatory field selection is made. For instance a typical journey could be:

Step 1 – Select the parameters describing the house

Step 2 – Select the weather assumptions by either selecting a day and location (temperature profiles are retrieved from stored data) or upload a specific temperature profile

Step 3 – Select energy efficiency interventions to be applied, these will alter the HTC values

Step 4 – Select a heating technology to apply, this will translate the heating demand (HTC times

temperature) into energy consumption

Step 5 – Press “Generate graph” to calculate and see the output of the selections

Step 6 – Iterate options and generate graphs until use is satisfied

Step 7 – Recall an older model to compare on the graph with the new options selected

Step 8 – Press “Generate model” to save to the model library, it will then appear in the list

'As built': Final outcome of steps:

- Extra steps added where Archetypes can be saved separately under ‘Created Archetypes’ and they can be used as a ‘base’ that users can add weather parameters to create a Profile.

6. System Interactions

The system interactions are driven from the user interface and chaining together the data flows to produce the output of energy profile data. Unlike client-server interactions the interactions will be between software modules that are implemented in application programming interfaces (APIs) between Javascript classes executing in the browser.

There is an interaction with the computer's local filesystem to save and load models and results.

Note, the Javascript classes for the model execution can be run through node.js in a server environment if a GUI is not required.

6.1. Models

Models can also be thought of as the archetype, along with the inputs that drive the generation of profile data. Options that are selected can be saved, along with the sample profile, so that different archetypes are formally defined.

The Model area lists the models that have been loaded from a model.json file. When selected the Model is visualised on the profile graph.

6.1.1. Save models

On saving the models, a model.json file is created and a file dialogue is opened to save the model.json file on the user's local computer. The file is plain text and can be emailed, shared on a server, or uploaded to a sharing platform like Github.

The last run of the data is saved as a profile object.

6.1.2. Load models

After a model.json file has been created, it can be loaded back into the tool for the next session, or if it has been shared can be loaded by someone else. Once the models are loaded they appear on the list.

The last run of the model data is loaded as well.

6.1.3. Create model

Accessible from the Options section, a model can be created with a name. The GUID would have been assigned from the "Generate data" step, but if this is the first interaction after setting the parameters, i.e. the model has not been generated, "Generate data" will be run before "Create model".

The GUID is cleared after the "Create model" so that any new changes in the options are saved under a new GUID.

The name of the model is user definable, it does not need to be unique.

Note, there is no "update model", changes

```
Class Models {
        public models[] Model
        function add(),
        function remove(),
        function save(),
        function import(filename File)
}

Class Model {
        public guid GUID,
        public name String,
        public data Data,
        public options Options,
        function create(),
        function export()
}
```

6.2. Data

Profile data that is generated from the Model definition is shown in the data section. The “Current” data series shows the model that has not yet been saved.

Once the “Create model” has occurred then the name of the model is taken on by the label.

The class data has an instance of a model including a copy of the energy profile of half hourly values. To support the graph and series, up to four data objects can be in memory and bound to the graph.

Once models are saved, they are in a model library. New profiles can be generated with new temperature data in the Network Planning module to be described in a separate deliverable.

```
Class Profile {
        public startdate Date,
        public demand[48] <Float>
}

Class Data {
        public model Model,
        public baselineProfile BaselineElectricity,
        public heatingProfile HeatingDemand,
        function profile() Profile, // add baseline and heating
        function export(),
        function import(filename File)
}
```

6.3. Options

Generally there are input options provided by the user interface to generate energy consumption profiles. These options are formalised as class structures. An effort to use meaningful variable names has been made to help with documentation.

Classes can use data from other classes when they are referenced as the variable type, for instance the Household class defines the parameters of interest for a house and those are used in the BaseElectricity class as a named variable household. They can then be accessed from reading the next level down, e.g. household.builtForm.

Enumerated or Enum typed variables will be a pick list that comes from a known set such as Table 1.

6.3.1. Household parameters

Household parameters are specified using select boxes to constrain input. The archetype is driven from these selections. From the archetype the Heat Demand (HTC, temperature set point) and Base Electricity (occupancy, multiplier, base load) random variables are looked up within the trained data set. By construct, each of the different household parameter combinations will have random variables with different values. As an example the expected value of the HTC for an insulated roof will be higher than for a non-insulated roof; the variation in HTC may be greater for older buildings than newer ones due to variations in building conditions.

'As built': Final outcome

```
Class Household {
      public builtForm Enum,
      public propertyType Enum,
      public energyRating Enum,
      public mainFuel Enum,
      public constructionAgeBand Enum,
      public roofType Enum
}
```

6.3.2. Weather

Weather class to hold the temperature assumptions for running the model. The location, date, and stationId can be used to load data from historical sources. The populate method will get the data associated with the weather station for that day.

If using a file for populating temperatures, then the load method and a filename would be used.

The target object that will be populated is the temperature, that is a typed array of 48 values with a start date.

```
Class Weather {
    public date Date,
    public location Location,
    public temperature Profile,
    private stationId Enum, // ASOS weather station id
    function populate(), // populate the temperature from asos model
    function load(filename File), // load temperature profile from external file
}
```

6.3.3. Baseline electricity

Baseline electricity use is the non-heating load electricity consumption for the 48 half hour periods of the day. Parameters are replicated so that this software class can be run stand alone from the other model components. This gives modularity to the design if a different (or multiple) estimation techniques are adopted in the future.

It is the application developer's responsibility to make sure that the input parameters are the same between all the estimator classes.

The calculate method populates the profile variable, which is the public accessor for the profile data whereas, internal to the model are baseload and occupancy series that hold copies and should not be directly accessed as there is no guarantee that those are working variables.

Samples will mainly be used internally in the model to populate the random variables and draw samples from the distributions. It can take an optional parameter that specifies how many samples to take, normalising the output when displaying a single household profile or if a histogram plot is required within a user interface. This is useful to show the shape of the probability density.

The probability density function (pdf) can be directly accessed through the accessor function. The pdf() call returns the class of function (normal, poisson, etc) and its parameters (mean, variance, etc) in an array, one entry for each random variable in the model.

```
Class BaselineElectricity {
    public household Household, // household object defining the parameters lookup parameters
    public date Date, // for seasonal reference, day light hours
    public location Location, // for day light hours
    public profile Profile, // container for the result data
    private occ[48] Number, // occupancy estimate
    private multiple Number, // multiplier for occupancy to energy
    private baseload[48] Number, // baseload
    function calculate(), // populates profile
    function sample(), // populate the random variables
    function pdf(), // return probability distribution function
}
```

6.3.4. Heating demand

The heating demand, expressed in kWh of energy, is based on random variables for the heat transfer co-efficient, household temperature set point, and an input of external temperatures (Weather.)

The occupancy series has been modelled for future use; it is mainly to identify waste rather than driving the heating demand. The theory is the occupancy could be determined through analysis of the electricity use and then compared to times when heating is on while no one is home. This becomes useful to apply to various heating technologies.

The calculate method populates the profile variable, which is the public accessor for the profile data whereas, internal to the model are baseload and occupancy series that hold copies and should not be directly accessed as there is no guarantee that those are working variables.

Samples will mainly be used internally in the model to populate the random variables and draw samples from the distributions. It can take an optional parameter that specifies how many samples to take, normalising the output when displaying a single household profile or if a histogram plot is required within a user interface. This is useful to show the shape of the probability density.

The probability density function (pdf) can be directly accessed through the accessor function. The pdf() call

returns the class of function (normal, poisson, etc) and its parameters (mean, variance, etc) in an array, one entry for each random variable in the model. In this case, the random variables are HTC and t*set*.

```
Class HeatingDemand {
    public household Household, // household object defining the parameters lookup parameters
    public weather Weather, // weather assumptions, including external temperature profile
    public profile Profile, // container for the result demand data, when I want heat
    private heatingTechnology <<HeatingTechnology>>, // instance of one of the heating techs
    private energyEfficiency EnergyEfficiency, // factors that change htc
    private occ[48] Number, // occupancy estimate
    private htc Number, // heat transfer co-efficient
    private tset Number, // temperature set point
    function calculate(), // populates profile
    function sample() // populates htc, tset from probability distribution function
    function pdf(), // return probability distribution function
}
```

6.3.5. Heating technology

Heating technology classes are built from a base class with particular heating technologies extending the data model. The extensions are for parameters that are particular to the technology, whereas the base class presents the profile outputs in a common way.

The calculation methods are different for each technology but named the same for ease of use. Other technologies could be added in the future as long as they inherit from the base class and contain a specific implementation of the calculate function that relates to the technology.

```
Class <<HeatingTechnology>> {
    public name String, // Allows for familiar name
    public demand HeatingDemand, // demand object
    public fuel Enum, // Gas, Elec, Oil
    public profile Profile, // container for the result energy data
    private classname typeof(), // which class is the instance
    function calculate(), // populates profile

}

Class ElectricHeating extends HeatingTechnology {
    private size Number, // size of heater in kW
    private number Number, // number of heaters if known
    function calculate(), // populates profile
}

Class HeatPumpHeating extends HeatingTechnology {
    private size Number, // size of heater in kW
    private cop Number, // efficiency
    private radiator Enum, // underfloor, small, large
    function calculate(), // populates profile
}


Class GasHeating extends HeatingTechnology {
    private size Number, // size of boiler in kW
    private efficiency Number, // efficiency
    private radiator Enum, // underfloor, small, large
    function calculate(), // populates profile
}
```

6.3.6. Energy efficiency

The energy efficiency interventions that can be applied to an archetype. They will produce an enhanced HTC, taking the starting HTC and applying changes. The process of applying changes is through the calculate method.

The household starting values and assumptions are held in the household object. This can be set from the loadHouse method or assigned directly.

```
Class EnergyEfficiencyPackage {
    public enhancedHtc Number, // Adjusted Htc with EE measures
    public htc Number, // Starting htc
```

```
    public household HouseHold, // the loaded household to drive basic uvalues
    public windows Enum, // type of windows
    private uWindow Number, // reference value for insulating quality of windows
    private wtowratio Number, // wall to window ratio
    public wall Enum, // wall insulation selector
    private uWall Number, // reference value for insulating quality of wall
    public roof Enum, // roof insulation selector
    private uRoof Number, // reference value for insulating quality of roof
    public floor Enum, // floor insulation selector
    private uFloor Number, // reference value for insulating quality of floor
    public thermalbridge Enum, // type of thermal bridging
    private uThermalBridge Number, // reference value for insulating quality of bridges
    function loadHouse() Household, // load from higher object, like demand
    function calculate(), // populates uvalues and enhancedHtc from public inputs
}
```

7. Access and security

The tool will be open sourced and there is no restriction on use. Github will be the repository for release.

There are no security implications of the tool as there is no network element or wider system impact. The code is distributed as Javascript and will run within the browser's security.

Note, in response to question about Python, Python is not being used as there is not a way to execute Python browser side, therefore a Python source code with files would have to be supplied, with limited user interface capabilities. All libraries are self contained. The Javascript can be called from an API in addition to local execution. Javascript code will be open source and is source code by nature.

Access will be from opening an index.html within a folder. The folder will contain the following:

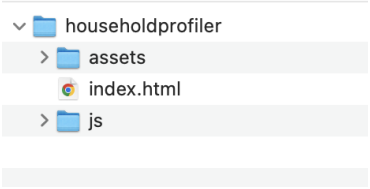

Figure 8. Directory structure of the files with the index.html as the launch file from the file system

8. Software Modules

The software will run on browsers. Chrome and Edge will be the targets tested with Chrome used for development.

8.1. Angular

A framework for coding single page applications that uses Typescript as the source language, transpiled into a distributable set of Javascript files. Metronic components will be used for page layout and selectors using metronic-ng.The compile settings in Angular will be for a single page. Tools for minification will be used. Asset and Javascript folders will contain the main code base.

8.2. Visual Components

The visual framework Metronic will be used. It has support Angular Material, Bootstrap layout and amcharts. Line charts are the preferred visual when there are multiple series to compare.

8.3. Data Generation

Custom classes as shown above in Systems Interactions. Zebra library will be used for CSV exports. Amcharts used for visualisation.

8.4. Generative Models

WebPPL libraries will be used for sampling of the random variables. The probability distributions and parameters describing them will be done offline in PyMC3.

TensorflowJS may be used for matrix/vector manipulation with the added advantage it could be used for data generation in the future.

PyMC3 can be used to draw new samples, although the focus is on Javascript and Tensorflow is also supported within the Python.

8.5. Static Data

JSON flat files will be used for static model data such as reference values, weather, stationId lookups, etc.

9. Appendix A. Weather Stations

StationID	latitude	longitude
EGAA	54.6633	-6.2258
EGAC	54.6135	-5.8735
EGAE	55.0428	-7.1611
EGBB	52.4602	-1.7576
EGBE	52.3697	-1.4797
EGBJ	51.8923	-2.1608
EGCC	53.3537	-2.275
EGCK	53.1042	-4.3403
EGCN	53.4747	-1.0044
EGDM	51.1618	-1.7546
EGDR	50.0843	-5.2571
EGDX	51.4059	-3.4348
EGDY	51.0063	-2.6428
EGEC	55.4372	-5.6864
EGEO	56.4635	-5.3997
EGFF	51.3967	-3.3433
EGGD	51.3827	-2.7191
EGGP	53.3333	-2.85
EGGW	51.8747	-0.3683
EGHC	50.1028	-5.6706
EGHE	49.9144	-6.2958
EGHH	50.7794	-1.8362
EGHI	50.9499	-1.3585
EGHQ	50.4406	-4.9954
EGJA	49.7061	-2.2147
EGJB	49.4331	-2.5981
EGJJ	49.2096	-2.1943
EGKA	50.8353	-0.2943
EGKB	51.3308	0.0325
EGKK	51.1481	-0.1903
EGLC	51.5053	0.0553
EGLF	51.28	-0.7727
EGLL	51.4785	-0.4614
EGMC	51.5714	0.6956
EGMD	50.9561	0.9392
EGMW	56.3592	-1.4941
EGNC	54.9387	-2.8088
EGNH	53.7744	-3.0394
EGNJ	53.5744	-0.3508
EGNM	53.8618	-1.6653
EGNO	53.7451	-2.8831
EGNR	53.1747	-2.9872
EGNS	54.0833	-4.6333
EGNT	55.0375	-1.6917
EGNV	54.5092	-1.4294
EGNX	52.8311	-1.3281
EGOM	55.05	-2.55
EGOP	51.7167	-4.3667
EGOS	52.7947	-2.6647
EGOV	53.2527	-4.5365
EGOW	53.5816	-3.0555
EGPA	58.9536	-2.9014
EGPB	59.8789	-1.2956
EGPC	58.4589	-3.0931
EGPD	57.2049	-2.2053
EGPE	57.5425	-4.0475
EGPF	55.8719	-4.4331
EGPH	55.95	-3.35
EGPI	55.6819	-6.2567
EGPK	55.5094	-4.5867
EGPL	57.4811	-7.3628
EGPM	60.4328	-1.2961
EGPN	56.4525	-3.0258
EGPO	58.2156	-6.3311
EGPU	56.5	-6.8806
EGQA	57.8249	-3.9555
EGQK	57.6456	-3.5635
EGQL	56.3774	-2.8624
EGQM	55.4209	-1.6012
EGQS	57.7114	-3.3235
EGSC	52.205	0.175

EGSH	52.6333	1.3
EGSS	51.8812	0.2232
EGTC	52.0722	-0.6167
EGTE	50.7372	-3.4058
EGTK	51.8369	-1.32
EGUB	51.6202	-1.0987
EGUL	52.4093	0.561
EGUN	52.3619	0.4864
EGUO	51.4391	-2.2864
EGUW	52.1239	0.9573
EGVA	51.6822	-1.79
EGVN	51.7585	-1.578
EGVO	51.2391	-0.945
EGVP	51.1394	-1.5686
EGWU	51.55	-0.4167
EGXC	53.0939	-0.1729
EGXE	54.2969	-1.5331
EGXP	53.3069	-0.5481
EGXS	53.4749	0.1521
EGXT	52.6114	-0.4612
EGXU	54.0453	-1.2511
EGXV	53.8758	-0.435
EGXW	53.1754	-0.5233
EGXZ	54.2055	-1.3821
EGYD	53.0317	-0.5047
EGYE	52.9622	-0.5616
EGYH	52.8733	0.1385
EGYM	52.6483	0.5671