EV Rate Calculator

NOTE: THIS IS A DEVELOPMENT VERSION OF THIS TOOL

The EV Rate Calculator is a python-based tool that calculates the utility costs for charging a private electric vehicle (or fleet of electric vehicles on the same meter) or a public electric vehicle charging station. The main purpose of this tool is to enable cross-rate analyses of charging costs for any EV charging scenario. For example, it could be used to calculate the cost to operate a public DCFC charger across all supported utility rates.

 Support for the tool is limited. Please report bugs in the issues section of this repository. Current version: 0.2

If you use the EV Rate Calculator in your work, please cite it as follows: >Atlas Public Policy, 2023. EV Rate Calculator (v0.2). Washington, DC. www.github.com/AtlasPublicPolicy/ev-rate-calculator.

The tool uses utility rates from the OpenEI Utility Rate Database. THe accuracy of this tool is dependent on the accuracy of the data in the OpenEI database.

Functionality

The tool supports most commonly-used utility rate structures in the United States. * Flat or tiered volumetric (energy) rates * Time-of-use (TOU) rates * Flat or tiered Demand Charges * Time-of-use (TOU) Demand Charges

The tool does not support: * Rates with coincident demand charges * Rates with ratcheting demand charges * Rates where demand charges are assessed on a daily basis * Complex rates with interacting tier and TOU components

The tool is designed to be used from the command line and so use requires comfort with simple command line interfaces. Future versions of the tool may include a web-based interface for accessibility and ease of use.

The EV Rate Calculator does not simulate charging behavior or energy and power requirements for charging vehicles. Users must model or these aspects of EV charging separately, use real-world charging data, or otherwise provide charging energy and power curves. Users provide both a 24-hour hourly average energy use, and hourly peak power use for the charging scenario. The tool supports both single-inputs (one 24-hour input) or monthly inputs (twelve 24-hour inputs). See the Input Data section in the User Guide for more information on inputting data for the tool.

User Guide

This tool requires a basic understanding of command line interfaces and familiarity with Excel. It does not require any programming knowledge or experience. However, because the tool ships with minimal error handling built-in, some familiarity with Python is helpful for troubleshooting any edge-case errors that are not expressly handled by the tool.

Installation

1. Download the latest release of the tool from the releases page and unzip the file to a location of your choice on your computer.
2. Download and install the latest version of Python and install it on your computer. For instructions on how to install python, see the Python Documentation or one of the many tutorials available online for installing Python on your operating system such as this one. (*Note: This tool requires Python 3.9 or higher. It has not been tested on earlier versions of Python.*)
3. Run initial_setup.py to automatically install the required Python packages and configure the directory structure for the tool. To run the script, ensure that your OS has associated the .py file extension with the Python interpreter (python.exe in the python directory). Double-click the initial_setup.py file and let it run. You may also run the script from the command line by navigating to the directory where the script is located and running `python initial_setup.py`.

User Input Data

Open the rate_calculator_input_file.xlsx file in Excel. This file is the user-data input interface for the tool. The file contains two sheets: `Single` and `Monthly`. * The `Single` sheet is used for single-input scenarios, where the user provides a single 24-hour average energy use and a single hourly peak power use. * The `Monthly Inputs` sheet is used for monthly-input scenarios, where the user provides twelve 24-hour average energy uses and twelve hourly peak power uses. Users should use monthly scenarios when they expect that energy and peak power use varies month to month or seasonally.

It is only neccesary to fill out one of these sheets, depending on the type of input data you wish to use. The tool will ask which input to use when it is run and will ignore the other sheet. For whichever sheet you use, you must fill in values for all input cells (yellow) in the sheet. If no power or energy is used during a given hour, enter a zero in the input cell for that hour. Valid input values are any positive number or zero. Excel will not allow you to enter negative numbers in the input cells.

If the tool is run and any input cells are left blank, the tool will throw an error

and exit. Excel will change the color of the input cells to green when required input data is provided.

Input Guidance

- `Hourly Average Energy` use is the mean energy use (in kWh) for a given hour in any *Charging Day*. It should be an average of the energy use for that hour across all **charging days** in the month (or year if using single month). For example, if on days the vehicle charges, it consumes 100kWh between 1pm and 2pm, on half of the days it charges and on the other half it consumes 50kWh between 1pm and 2pm, the hourly average energy use for 1pm to 2pm is 75kWh. (*Note: The tool handles number of days charging in a month through a separate user input discussed later in the script and so users do not need to account for non-charging days when developing average energy inputs.*)
- `Hourly Peak Power` is the expected peak energy use for any 15-minute interval within a given hour for any given month (i.e., the maxiumum power draw that will occur for that hour for any charging session across the whole month). In a perfect scenario, where charging is perfectly predictable and consistent, this would be the same as the `Hourly Average Energy` use. However, in real-world scenarios, this is often not the case. For example, if a vehicle charges for 1 hour at 1pm on half of the days in a month and consumes 100kWh during that hour, and on the other half of the days in the month it charges for 1 hour at 1pm and consumes 50kWh during that hour, the hourly average energy use for 1pm to 2pm is 75kW, but the hourly peak power is 100kW. Moreover, in situations where charging is not consistent (such as public charging) the peak may be much higher (up to the maximum power capacity of the charger or group of chargers) than the average energy use.

For users that are not interested in TOU rates, it is acceptable to load all charging energy into the first (or any arbitrary) hour because the time incidence of energy consumption is not important (all other fields must be zero). However, that will result in incorrect result for TOU rates. The same is true for peak power fields for users that are not interested in calculating TOU demand charges. Additionally, if users are not interested in any demand charges, they may enter zero for all demand charge fields.

All input files must be saved in the user_input directory to be accessible to the tool, However, they may be saved with any name. The tool will allow you to select the input file you wish to use when it is run. Best practice is to save a new file with a unique name for each scenario you wish to run. You may choose to pass the input file name to the output file name to make it easier to identify which output file corresponds to which input file.

Running the Tool

Once you have specified your input file you will run the main.py script. To run the script, ensure that your OS has associated the .py file extension with the Python interpreter (python.exe in the python directory). Double-click the main.py file and let it run. You may also run the script from the command line by navigating to the directory where the script is located and running `python main.py`.

The tool will launch to a main menu with the options: 1. Run Rate Calculator 2. Refresh Cache 3. Open Input Workbook 4. Exit

Use the arrow keys to select the option you wish to run and press enter.

- `Run Rate Calculator` will launch the rate calculator dialog which provides configuration options for running the tool. See the Rate Calculator section for further instructions.
- `Refresh Cache` will refresh the cache of rate data from the OpenEI database. The tool will also automatically build the cache on the first run, so it is only necessary to refresh the cache if you wish to update the rate data.
- `Open Input Workbook` will open the input workbook in Excel. This is a shortcut to open any input workbook in the user_input directory.
- `Exit` will exit the tool.

Rate Calculator

Running the rate calculator will launch a dialog with the following configuration options * `Input File` - Select whether to use the default input file or a custom input file. If you select `Custom`, the tool will list all .xlsx files in the user_input directory and allow you to select the file you wish to use.

- `Rate List` - select what set or subset of rates to run the calculator against. Filtered rates are rates that are current, are residential, commercial, or industrial rates, and are not specialty rates (such as space heating rates). *Specialty rates are removed by keyword detection, so it is possible that some specialty rates may be included in the filtered rates list.* Users may also select subsets of the filtered rates for residential, commercial or industrial rates only. Additionally users may select all rates, which will run the calculator against all rates in the OpenEI database. This is useful for users that wish to subset rates based on other criteria after the tool has run. (*larger rate sets will take longer to run and save and will result in larger output files*)
- `Number of days` - select how many days per week (1-7) that vehicles are likely to charge. This determines how many weekday and weekends that charging takes place and the total amount of energy demanded in a given month. The tool assumes priority of weekend charging over weekday charging to take advantage of cheaper weekend rates (if those are available). For

example, a fleet that must charge 5 days a week will charge 4 days during weekdays and one charge over the weekend. For public charing, users should select 7 days per week.

- `Single or Monthly Charging Input` - select whether to use the single or monthly input sheet. If you select `Single`, the tool will use the `Single` sheet in the input workbook. If you select `Monthly`, the tool will use the `Monthly Inputs` sheet in the input workbook.
- `Output File Name` - Users have choice to select the default output file, a custom output file, or name the file after the input file name (with _output appended). If you select `Custom`, the tool will allow you to enter a custom file name. Only valid charachters will be allowed (invalid chars are automatically removed)

Once you have completed the dialog options the tool will present a summary of inputs and ask you to confirm that they are correct. If you select `Yes`, the tool will run the rate calculator and save the output file. If you select `No`, the tool will return to the dialog and allow you to change the options. If you select `Exit`, the tool will exit the dialog and return to the main menu.

The tool will run automatically, and will display a progress bars or in progress messages as it runs. Upon completion the data will be saved in the output file and the user is given the option to open the output file in Excel. Users can select to run the rate calculator again from the main menu or exit the tool.

Modules and other files

The program consists of four modules in addition to the main.py file and the initial_input.py file. The modules are: 1. inputFunctions.py - contains functions for reading and processing inputs 2. interfaceFunctions.py - contains the functions for the user interface and high level control of the program 3. calculatorFunctions.py - contains the functions for calculating rates 4. outputFunctions.py - contains the functions for assembling and writing output files

In addition to the modules, there is: 1. calculator_documenation.ipynb - that contains the documentation for the calculator functions in a Jupyter notebook format that can be viewed and run to test calculator methods 2. test_data.py - that contains test rate data for the calculator documentation notebook