



Reshaping EV load profiles at  
an office by adopting smart  
charging strategies

*Manual*

Roeland in 't Veld





# Reshaping EV load profiles at an office by adopting smart charging strategies

Manual

Roeland in 't Veld





## Introduction

This manual gives an explanation to the charging controller used for testing at the Smart Energy Delivery (SEND) lab at Avans. This includes setting up the controller to work with MATLAB Simulink and the operation of the charging controller. Furthermore, the parameters that can be altered are presented.

## Installation

### Prerequisites

- MATLAB 2021b (with Simulink)
- Anaconda 3
- Pycharm community (with GIT)
- MatlabChargingController (from GIT)
- Python 3.8 (Standalone)
- iPOPT (using Ipopt.zip)

### Modules

- pandas
- numpy
- datetime
- requests
- io
- json
- pyomo

if any modules are missing Python will throw an error and the missing package needs to be installed.

### Connecting Python and MATLAB

#### 1. Installing MATLAB engine for Python

To install from the MATLAB folder, on Windows open CMD.exe and type:

```
cd "C:\Program Files\MATLAB\R2021b\extern\engines\python"
python -m pip install .
python -m pip instal matlabengine
```

note: if python is not found: the folder for python 3.8 is not added to the windows PATH variable. Make sure the option to add python to the PATH variable is selected on install or add it manually.

[https://nl.mathworks.com/help/matlab/matlab\\_external/install-the-matlab-engine-for-python.html](https://nl.mathworks.com/help/matlab/matlab_external/install-the-matlab-engine-for-python.html)

#### 2. Adding Python to MATLAB

Open MATLAB and in the terminal enter:

```
pyenv (Version="3.8")  
or  
pyenv (Version="executable")  
(where executable is the directory to python.exe)
```

[https://nl.mathworks.com/help/matlab/matlab\\_external/install-supported-python-implementation.html](https://nl.mathworks.com/help/matlab/matlab_external/install-supported-python-implementation.html)

## Installing Python modules

Open CMD.exe and type:

```
python -m pip install [module_name]
```

This has to be done for all modules listed.

If python is not found use the following command:

```
C:\Users\roela\AppData\Local\Programs\Python\Python38\python.exe -m pip install [module_name]
```

## Adding project to Pycharm from GIT

When installing Pycharm add a GIT account that has access to the MatlabChargingController GIT repository.

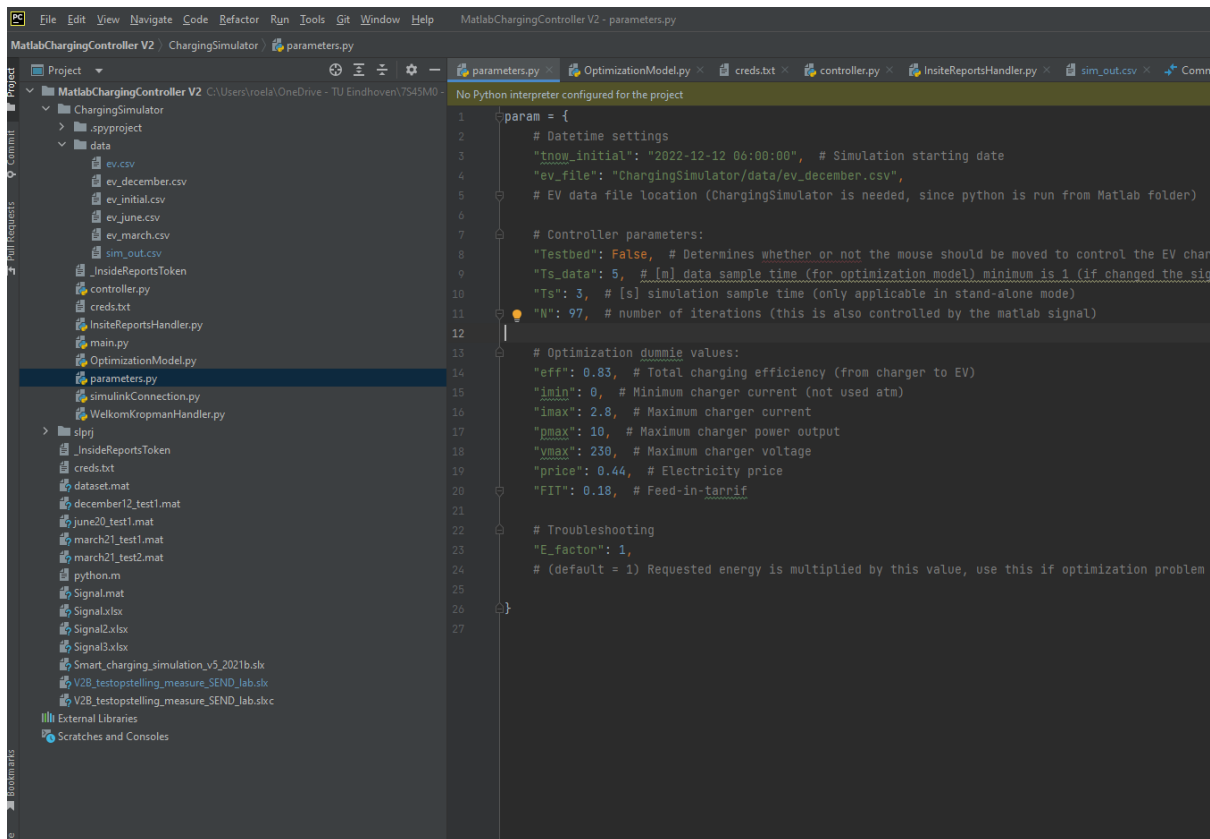
Create a New Project within Pycharm and choose **Get from Version Control**. Clone the MatlabChargingController to a desired directory.

<https://www.jetbrains.com/help/pycharm/set-up-a-git-repository.html#clone-repo>

# Configuration

## Parameters

Within Pycharm it is possible to alter the parameters for the optimization model. In parameters.py all the parameters are listed, and descriptions are included.

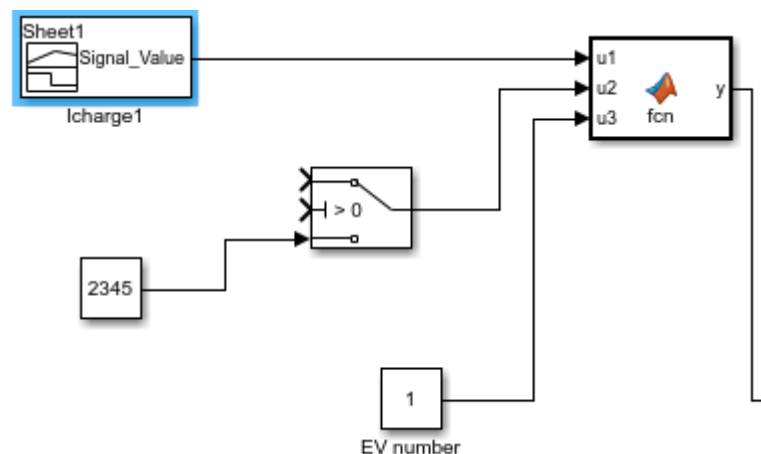


```
1 param = {
2     # Datetime settings
3     "tnow_initial": "2022-12-12 06:00:00", # Simulation starting date
4     "ev_file": "ChargingSimulator/data/ev_december.csv",
5     # EV data file location (ChargingSimulator is needed, since python is run from Matlab folder)
6
7     # Controller parameters:
8     "Testbed": False, # Determines whether or not the mouse should be moved to control the EV charging
9     "Ts_data": 5, # [m] data sample time (for optimization model) minimum is 1 (if changed the simulation time)
10    "Ts": 3, # [s] simulation sample time (only applicable in stand-alone mode)
11    "N": 97, # number of iterations (this is also controlled by the matlab signal)
12
13    # Optimization dummy values:
14    "eff": 0.83, # Total charging efficiency (from charger to EV)
15    "imin": 0, # Minimum charger current (not used atm)
16    "imax": 2.8, # Maximum charger current
17    "pmax": 10, # Maximum charger power output
18    "vmax": 230, # Maximum charger voltage
19    "price": 0.44, # Electricity price
20    "FIT": 0.18, # Feed-in-tariff
21
22    # Troubleshooting
23    "E_factor": 1,
24    # (default = 1) Requested energy is multiplied by this value, use this if optimization problem
25
26 }
27
```

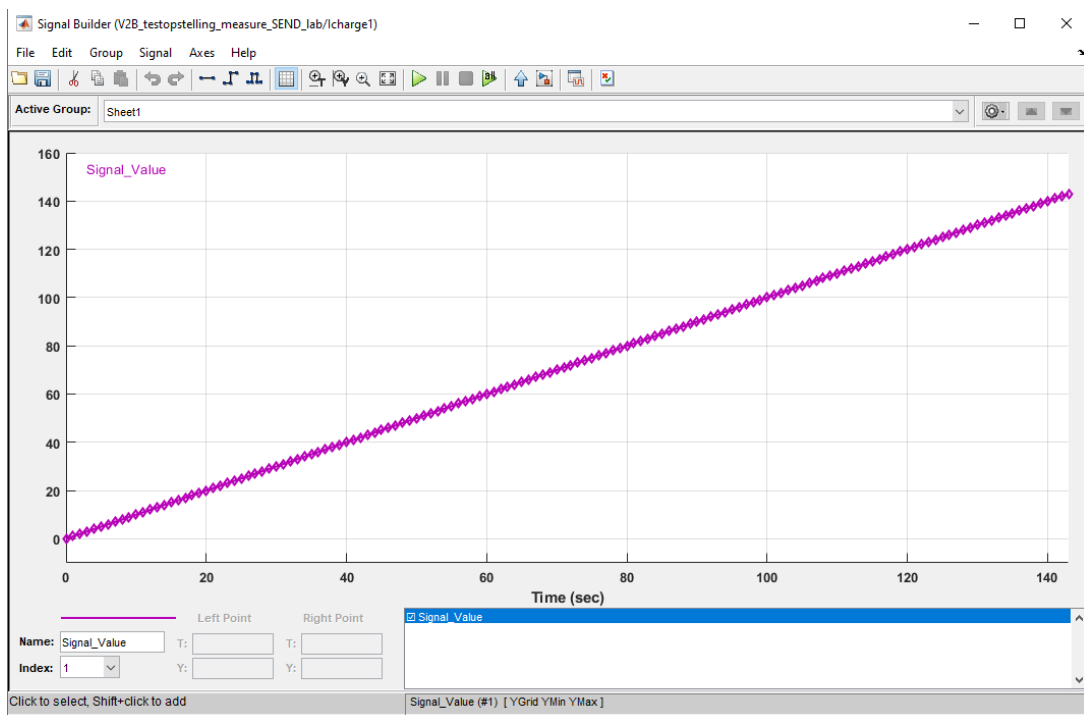
## MATLAB signal

If the sample time is altered in the parameters.py file, for instance from 5 to 15 minutes. The signal in the Simulink file needs to be changed. The signal is used to run the Python file every second and for each iteration. Going from 0 to 12 hours \* sample\_time - 1. So if the sample time is 5 minutes from 0 to 143 and if the sample time is 5 minutes from 0 to 47. This represents the daily operation of the charging controller for 12 hours long, running every  $n$  minute.

Double click signal value:



This opens the signal editor



Select File > Import from File

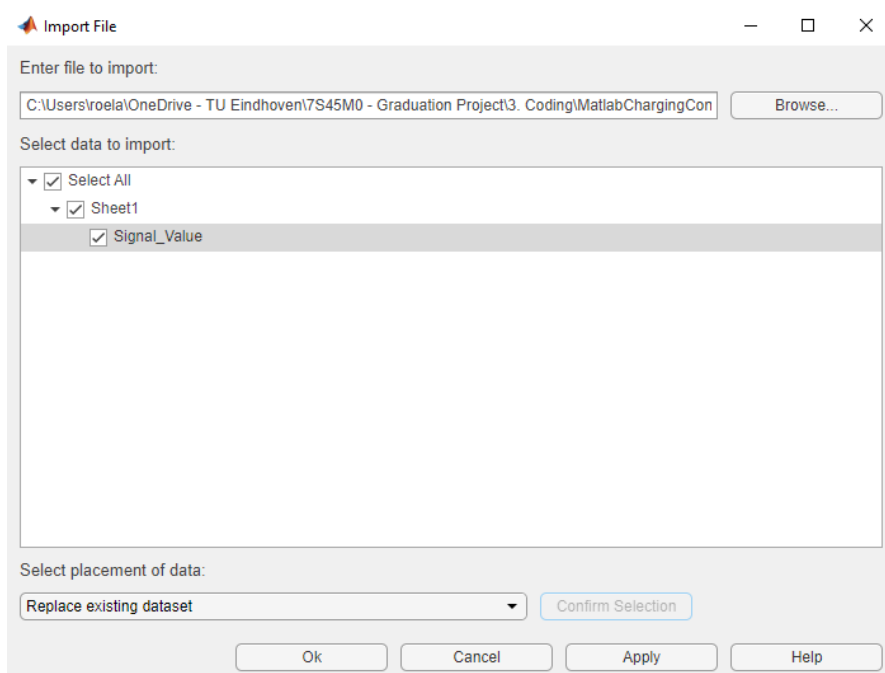
Click Browse...

Select either Signal\_5min.xlsx or Signal15min.xlsx

For select placement of data choose "Replace existing data"

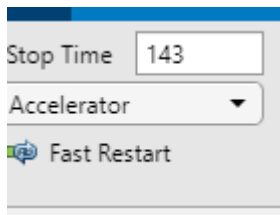
Click "Confirm Selection"

And finally "Apply" and choose "No import without saving".





Additionally, make sure the stop time is set to the number of iterations



In this case 143 for a sample time of 5 minutes. ( $12 \times 12 - 1$ )

## Matlab Time Steps

If the python model takes too long to compute and delays occur when sending the charging currents it is possible to change the discrete time steps in MATLAB.



Edit the block above in MATLAB and decrease the time steps from 1 to for example 0.5 or 0.25 seconds.

## Python AutoGUI (mouse mover)

If in parameters.py the Testbed parameter is set to True, the mouse will be moved to control the window of the EV charger on the SEND-lab testbed.

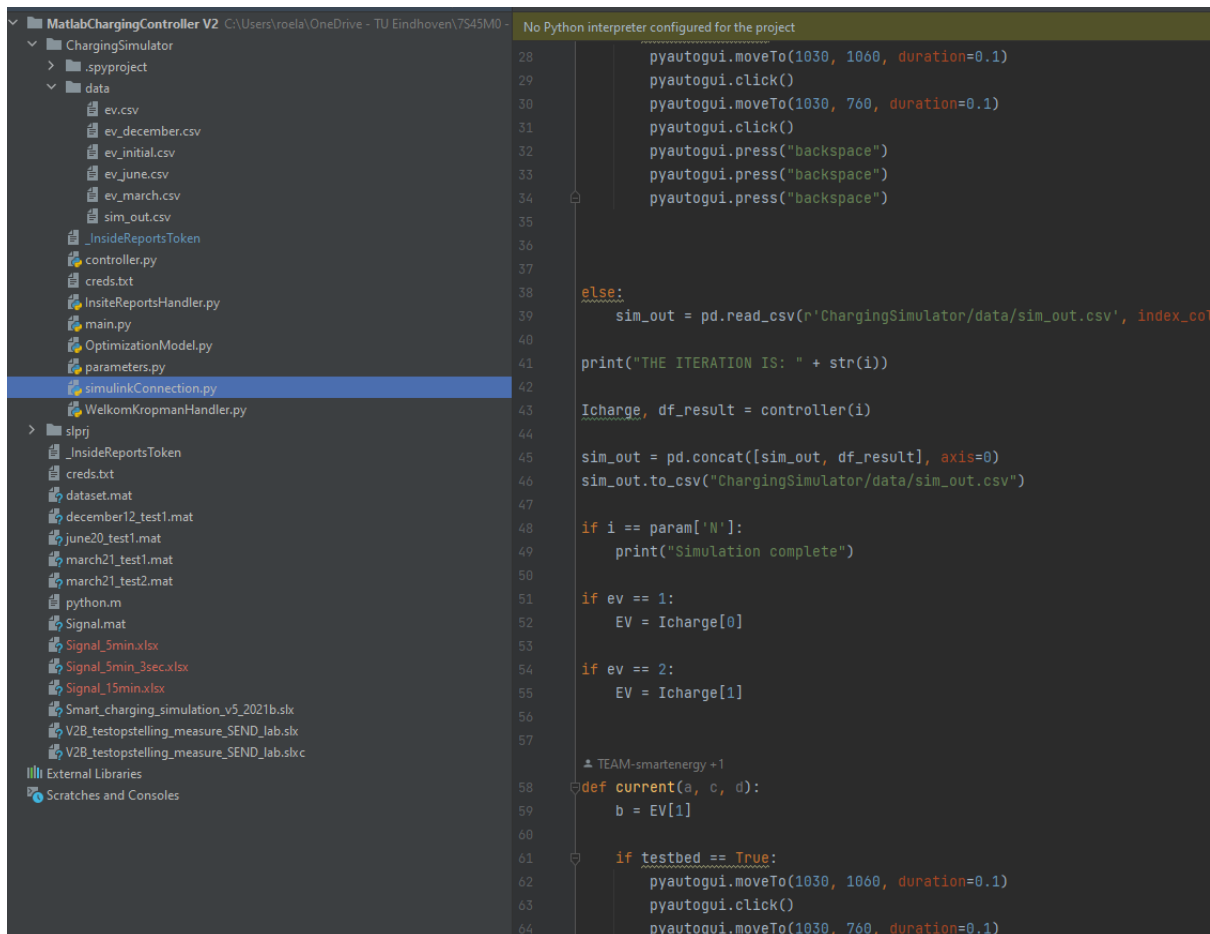
If the mouse does not move to the correct program on the taskbar. Alter the code in the SimulinkConnection.py file.

## Make sure to do this at both locations in the code

The first number is the width of the screen (out of 1920) and the second number is the height of the screen (out of 1080).

```
pyautogui.moveTo(1030, 1060, duration=0.1)
pyautogui.click()
pyautogui.moveTo(1030, 760, duration=0.1)
```

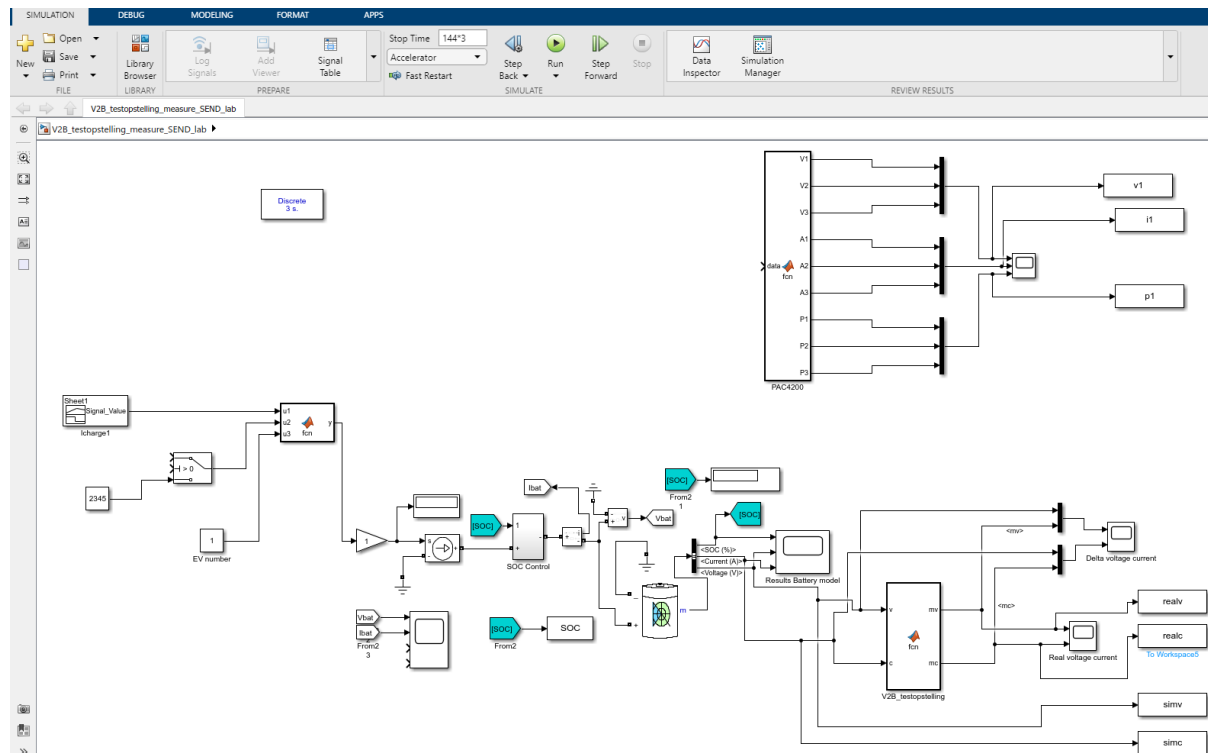
In the figure below the location of the code in the SimulinkConnection.py file is shown:



```
28     pyautogui.moveTo(1030, 1060, duration=0.1)
29     pyautogui.click()
30     pyautogui.moveTo(1030, 760, duration=0.1)
31     pyautogui.click()
32     pyautogui.press("backspace")
33     pyautogui.press("backspace")
34     pyautogui.press("backspace")
35
36
37
38 else:
39     sim_out = pd.read_csv(r'ChargingSimulator/data/sim_out.csv', index_col=0)
40
41     print("THE ITERATION IS: " + str(i))
42
43     Icharge, df_result = controller(i)
44
45     sim_out = pd.concat([sim_out, df_result], axis=0)
46     sim_out.to_csv("ChargingSimulator/data/sim_out.csv")
47
48     if i == param['N']:
49         print("Simulation complete")
50
51     if ev == 1:
52         EV = Icharge[0]
53
54     if ev == 2:
55         EV = Icharge[1]
56
57
58 TEAM-smartenergy +1
59 def current(a, c, d):
60     b = EV[1]
61
62     if testbed == True:
63         pyautogui.moveTo(1030, 1060, duration=0.1)
64         pyautogui.click()
65         pyautogui.moveTo(1030, 760, duration=0.1)
```

## Usage

Open V2B\_testopstelling\_measure\_SEND\_lab.slx in the MatlabChargingController folder. This will present the following window:



If no settings are altered the Simulink model can be run and the optimization model will run for each iteration.

By clicking on view diagnostics, it is possible to see each iteration result or view the error.

The results of the optimization model are stored in data/sim\_out.csv.

This file is updated each time the Simulink model is re-run. So make sure to copy this file before starting a new simulation if you want to save the data.

MatlabChargingController V2B_testopstelling_measure_SEND_lab.slx	
ChargingSimulator	
.spypoint	
data	
ev.csv	
ev_december.csv	
ev_initial.csv	
ev_june.csv	
ev_march.csv	
sim_out.csv	
InsideReportsToken	
controller.py	
creds.txt	
InsiteReportsHandler.py	
main.py	
OptimizationModel.py	
parameters.py	
simulinkConnection.py	
WelkomKropmanHandler.py	
slprj	
InsideReportsToken	
.csv files are supported in older Jetbrains IDEs	
1	patetime, EV1, PEV1, Current1, EV2, PEV2, Current2, EV3, PEV3, Current3, PEVTot,
2	2022-12-12 06:00:00, 0.0, 5.732976123743428e-12, 1.4009377053682729e-11, 0.
3	2022-12-12 06:05:00, 0.0, 5.732976123743464e-12, 1.400937705368274e-11, 0.
4	2022-12-12 06:05:00, 0.0, 5.732976123743464e-12, 1.400937705368274e-11, 0.
5	2022-12-12 06:10:00, 0.0, 5.732976123743428e-12, 1.4009377053682729e-11, 0.
6	2022-12-12 06:10:00, 0.0, 5.732976123743428e-12, 1.4009377053682729e-11, 0.
7	2022-12-12 06:15:00, 0.0, 5.73297612374343e-12, 1.400937705368274e-11, 0.0.
8	2022-12-12 06:15:00, 0.0, 5.73297612374343e-12, 1.400937705368274e-11, 0.0.
9	2022-12-12 06:20:00, 0.0, 5.732976123743466e-12, 1.4009377053682751e-11, 0.
10	2022-12-12 06:20:00, 0.0, 5.732976123743466e-12, 1.4009377053682751e-11, 0.
11	2022-12-12 06:25:00, 0.0, 5.732976123743571e-12, 1.4009377053682767e-11, 0.
12	2022-12-12 06:25:00, 0.0, 5.732976123743571e-12, 1.4009377053682767e-11, 0.
13	2022-12-12 06:30:00, 0.0, 5.732976123743571e-12, 1.4009377053682767e-11, 0.
14	2022-12-12 06:30:00, 0.0, 5.732976123743571e-12, 1.4009377053682767e-11, 0.
15	2022-12-12 06:35:00, 0.0, 5.732976123743496e-12, 1.4009377053682729e-11, 0.
16	2022-12-12 06:35:00, 0.0, 5.7329761237434955e-12, 1.4009377053682729e-11, 0.
17	2022-12-12 06:40:00, 0.0, 5.732976123743428e-12, 1.4009377053682729e-11, 0.
18	2022-12-12 06:40:00, 0.0, 5.732976123743428e-12, 1.4009377053682729e-11, 0.