EIS Software

Arun Atwal

September 2022

Contents

1 Introduction

Welcome to the EIS (Electrochemical Impedance Spectroscopy) automated software! This program is designed to control temperature ramping and holding on a sample stage, and automatically run data sweeps from an impedance analyser at the correct points, whilst providing a user interface for control. As a result, users are able to input parameters at the start and leave the software to run the experiment for extended periods of time, or make changes midway through.

This software was developed as part of the *Faraday Undergraduate Summer Experience* program under the *Faraday Institute*, with I, the intern, working under the supervision of Dutton group, Quantum Materials in the Department of Physics, University Cambridge. I also worked closely worked with Driscoll Group, in the Department of Materials Science and Metallurgy. As a result, throughout this document I will refer to the specific setup I left behind, currently within the Maxwell Centre, for the benefit of those who will likely use it, although much of this work is expandable and applicable elsewhere.

Currently, two stages from *Linkam Scientific* are supported: the *HFS350EV-PB4* and the *TS1000-PB4*. As for impedance analysers, the *Biologic SP200*

potentiostat is fully compatible, and the *Solartron 1260A* is nearly fully compatible (the main drawback being the data formatting, which I did not change within the timeframe of my internship as the Solartron was removed, although this could easily be changed to match the Biologic output). The code is structured so that it would be straightforward to implement other stages and analysers, if so desired. Similarly, alternative techniques could be performed at the measurement points, although currently the software is designed to record Real and Imaginary Impedance components over a range of frequencies.

2 Running an experiment - Quick Start

If using the setup existing at the time of writing, the following instructions will allow you to collect your desired data. Otherwise, or if any problems arise, see the further details in later sections. This document focuses on the software operation only.

1. Setup your experiment
    - Set sample in the stage, connect the analyser and check water cooling if necessary.
    - See SOP document for more details.
    - **Note:** Multiple pieces of equipment can be connected, but **only one stage can be turned on AND connected**. *Linkam* software does not support handling multiple stages, so for safety reasons this risk should be avoided.
    - Disconnect from or close any software that otherwise controls the equipment. For the case of the *Linkam* stages and the *Biologic SP200*, this would be the *Link* software and *EC-Lab*.
2. Run `EISMeasure.py`
    - Run the program in a **64-bit** Python environment. Only Windows support is guaranteed.
    - If using the original lab PC, there is an Anaconda Powershell Prompt that will suffice. This can be navigated using the `ls` to display the contents of the current directory (folder), and the `cd` command to change the directory. Alternatively, the `dir` command replaces `ls` in an ordinary command prompt. As of now, the script is located in `C:\Users\EIS User\OneDrive\Desktop\Final_EIS`.
    - Once in the correct directory, enter `python EISMeasure.py` to run the program.

- The graphical user interface will now open. Select the desired stage and impedance analyser, and input an experiment name - duplicate names will be appended with numbers. The "virtual" options are for debugging and testing purposes only - do not run a combination of virtual and real equipment.

3. Input parameters

   - Selecting "Start" will present users with all the input parameters necessary to create a temperature ramp and perform measurement sweeps. Allowed input ranges are displayed to the left of entry boxes. The options are as follows:
     - **Starting Temperature:** The first temperature point for a measurement ramp. The temperature limits depend on the stage in use, and are displayed. Unless sub-room temperature cooling apparatus (such as liquid nitrogen) are connected to the low-temperature stage, do not enter any temperatures below the current room temperature.
     - **End Temperature:** The last temperature point for a measurement ramp. Can be higher or lower than the starting temperature, with the same qualifications applying.
     - **Rate:** The heating rate of the ramp between hold points, and the cooling rate as far as is possible.
     - **Temperature Interval:** The gap in temperature between different measurements within the ramp. For example, if starting at 60°C and ending at 100°C, with a temperature interval of 20°C, the ramps would hold and measure every 20°C, i.e. at 60, 80 and 100°C.
     - **Minimum Holdtime:** The length of time, in seconds, that the temperature will remain constant at the measurement point before an impedance scan is performed. Includes the start and end point of the ramp. "Minimum" as if the temperature is not yet steady or if the program is paused, this can be extended.
     - **Voltage Amplitude:** The amplitude, in mV, of the voltage waveform passed through the sample.
     - **Minimum Frequency:** The minimum frequency for which data is recorded. Especially low frequencies will result in long sweep times.
     - **Maximum Frequency:** The maximum frequency for which data is recorded.
     - **Points per decade:** The number of frequency points within an order of magnitude for which data is recorded, and hence the resolution of data in frequency space.
     - **Sweeps at T:** The number of measurement sweeps performed at each temperature hold point in the ramp. Set to 0 to imple-

ment a ramp that only adjusts temperature between measurement ramps.

  - **Sweep Delay:** The length of time, in seconds, between successive sweeps at the same temperature, if "Sweeps at T" is set to be greater than 1.

- Select "Create Ramp" to save a ramp with these parameters. If the inputs are invalid, a notification detailing the cause will appear and the ramp will not be saved. If desired, further ramps with different parameters can be input.
- Select "Clear All" to delete all ramps and start over. Select "Clear Prev" to only delete the last ramp saved, if a mistake was made. The total ramp counter displays how many ramps are currently saved.
- Hit finish to see a preview of the temperature profile when ready. Take care of any warning notifications that may appear.

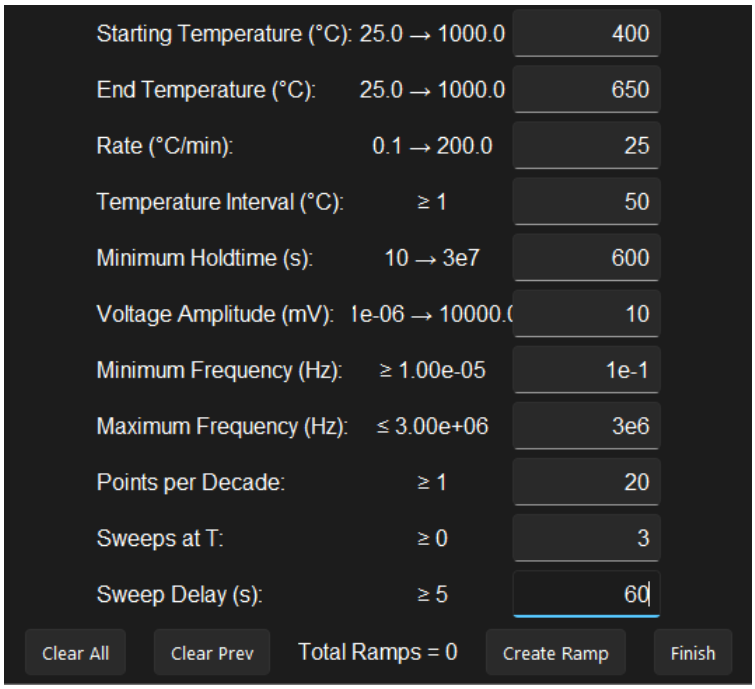

Figure 1: The data input page, with sensible values.

4. Run the experiment

   - A preview of the temperature profile will appear with an estimated time. If acceptable, select "begin experiment". Otherwise, select "start over" to clear all saved ramps and try again.

- A live temperature plot will appear, and the experiment will run. No further action is necessary as the stage will automatically cool upon completion. Experiment and stage details are updated in real time, as displayed in the pane to the left of the plot.
- There are 3 buttons that can be pressed in the lower left. "Stop" will cease the experiment and allow the stage to cool down; this button will update to "Finish" if the experiment is complete.
- "Pause" will indefinitely hold the temperature where it is and not perform a measurement. Do not leave the stage paused at a high temperature for extended periods of time. The same button becomes "Resume" to return to the experiment.
- "Edit Ramps" will function the same as pause, but allow new temperature ramps to be input, which will take effect immediately, overwriting the old ones. The total ramp counter will, at minimum, include the number of previous ramps input, even if they do not take effect.
- It is not possible to pause or edit ramps during a measurement sweep. It is viable when the stage is holding its temperature, but this will contribute to the holdtime. If the end of the holdtime is reached, the sweep will begin as soon as the program is resumed, unless the ramp point has been removed.
- To check if the stage is cool after the program has ended, connect via the *Link* software.

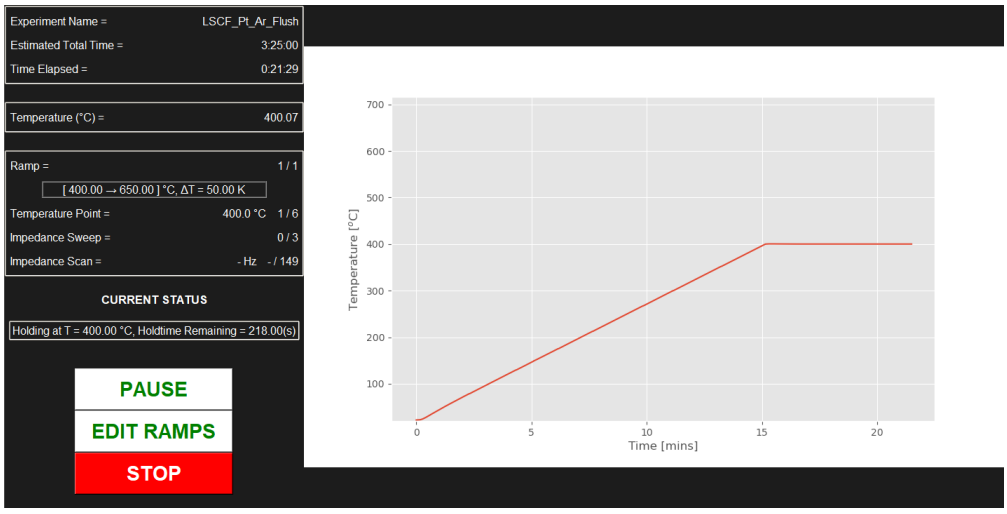

Figure 2: The main experiment page.

5. Collect data

- Data is saved within the `experiments` folder in the same directory as the Python script.
- There will be a folder within with the name of the experiment that was input at the start of the program. Inside this folder, experiment details, temperature profiles, and folders for each ramp will be included.
- Within each ramp folder, there will be folders for each temperature point measured. Inside these, text files with tab-separated data will exist for each sweep.
- There are 6 lines of header information about the measurement, followed by a 7th line with column headers, in each text file. This is the expected format for viewing the data in *ZView*. The column headers are Frequency/Hz, Re(Z)/Ohm, and Im(Z)/Ohm. The data here can be taken and interpreted as desired.

3 Troubleshooting

In the case of any code problems or missing files, the final version of the software I created can be found on my GitHub: `https://github.com/Arun-Atwal/Electrochemical-Impedance-Spectroscopy-Automation`

3.1 Linkam stages not connecting

- Ensure the stage is not connected in the *Link* software.
- Ensure only one stage is connected and turned on.
- Check the log text file in the `Final_EIS` directory.
- Ensure a 64-bit Python environment is being used.
- Ensure the `USBdriver` folder is present.
- Ensure `linkamSDK.dll` and `linkamSDK.pdb` are present. Use the version on my GitHub for guaranteed compatibility.
- Ensure there is a `linkam.lsk` license file in the directory. If it is missing, it can be obtained from older versions and potentially on my GitHub, unless I am required to remove it from there. The below point will likely need to be followed also.
- The log may regularly show the license has expired. This is a known issue: despite there being a valid license key, it "expires" every month or so. To resolve this, open the `linkam.lsk` file in notepad. Do not edit any of the contents - it will likely display as a rather cryptic set of symbols and a string of letters and numbers. If there are no letters or numbers, try a different text editor. Rename the file and re-save it as `linkam.lsk` as a .lsk, **not** a .txt data file. This "new" file will not be viewed as expired.

3.2 Biologic not connecting

- Ensure the analyser is not connected in the *EC-Lab* software.
- Ensure all relevant files are present in the `EC-Lab_Files` folder. If in doubt, replace the folder with an untouched version from the EC-Lab development package.
- Ensure the folder `kbio` is present, obtained from the EC-Lab development package.
- Alternatively, in case of an update, use the versions on my GitHub.

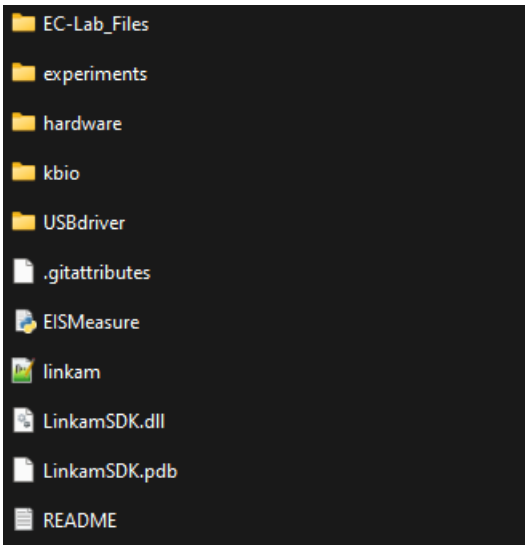

Figure 3: A complete directory for the software, exempting this document. Ensure all these files are present.

4 Code Structure and Dependencies

Should you wish to update the code, add functionality, or need to understand the file setup, the following information may be of use.

The main script is `EISMeasure.py`, containing the GUI and main experiment control. Throughout the code, analyser and stage objects are referred to. This leads to the importance of the `hardware` folder, currently containing `linkam.py`, `biologic.py, solartron.py` and `virtual.py`. Each of these contains classes referring to the specific analyser and/or stage they represent. If adding equipment, it is imperative you do so under this format, using the **same function**

names within the classes to control the physical hardware. This will allow the main script to call these general functions regardless of the equipment selected.

The necessary external dependencies are as follows:

Python

A 64-bit Python environment is required. Tested on version 3.8.3.

- numpy
- maptlotlib
- ctypes
- microscope
- sv_ttk, a third party GUI theme. See `https://github.com/rdbende/Sun-Valley-ttk-theme`
- pyvisa (only for Solartron use)

Linkam

- linkam.lsk, the license file. Obtained from *Linkam Scientific*. See troubleshooting for known errors.
- LinkamSDK.dll
- LinkSDK.pdb

The 64-bit version of these SDK files are required. They themselves have dependencies, however they should be included with standard versions of Windows.

Biologic SP200

For full functionality, the `EC-Lab_Files` folder taken from the *EC-Lab development package* is required. This contains a long list of `.ecc` techniques that various *Biologic* instruments may be able to perform. Crucial to the current operation are all 64-bit files that are not other `.ecc` techniques, `peis4.ecc`, and in a separate folder, the `kbio` scripts. See my GitHub for complete details.

Solartron 1260A

I personally did not work much with the Solartron system as it was obsolete by the time I started. If this functionality is to be revived, I will quote the instructions from the previous intern, Adam Alderton:

Assuming the `pyvisa` library is installed, the only dependency talking to the

Solartron unit has software wise the *NI-VISA* backend, which can be installed from here. The *NI-488.2* driver was also found to be useful, but perhaps optional from system to system (untested).

5 Issues and Future Suggestions

- The estimated time is currently rather dubious. The method employed for the *Biologic SP200* as of now is to rely on the lower frequency points as dominant, and multiply the reciprocal of the absolute lowest frequency by the points per decade and the "effective number of decades", which is essentially a fudge factor. This is currently set to 2, although future testing is required. The time appears to be broadly helpful regardless.
- Currently, manual flowmeters for gas control are in use. Theoretically, the gas analyser can be connected to the PC, so software controlled electric flowmeters would help immensely, although this would require more code.
- There is an optical temperature probe that can see into the stage near the setup, which has not been tested.
- It would be great to convert the program into a .exe for easier use. However, the `linkam.lsk` license file requires updating every month or so, making this impossible. It would be ideal to remove this issue.
- The program ceases to update any measurement information upon completion of the last sweep. This decision was made as the software may be running for hours before someone returns to it. However, it might be a good idea to display a non-logged temperature readout after regardless, and the opportunity to insert a new ramp at any point in the following cooling process.
- It would be easy to load other `.ecc` files that the *SP200* supports to add functionality.
- The code is well set-up to accept more analysers, and potentially stages, if the appropriate hardware scripts are written.
- Potentially live plots of data could be helpful. Adding to this, automatic addition of measurement points based on impedance values has been suggested.
- The *SP200* takes measurements using a variety of current ranges, which are automatically detected. In certain specific cases, users may want to control these manually, which is an option within *EC-Lab*, but not this software. This function could be added as a drop-down menu in the parameter input page.