User manual for DEC*i*M: Program for the Determination of Equivalent Circuit Models

Hendrik P. Rodenburg and Peter Ngene
Utrecht University

March 1, 2024

Table of contents

1 Introduction

1.1 Equivalent circuit modelling

The analysis of (electrochemical) impedance spectra usually involves fitting a model to the impedance data. In most cases, such a model takes the form of an equivalent circuit model, *i.e.* an electrical circuit whose impedance is similar to that of the sample. Equivalent circuit models may contain many different circuit elements, including capacitors, inductors, resistors, and more complex elements such as Warburg elements and constant phase elements, which can be arranged in many different ways. The circuit elements typically represent physical processes, and the values of the elements' parameters inform on the properties of the measured system. For instance, a resistor can represent the resistance of an electrolyte. If the dimensions of the electrolyte are known and its resistance can be determined, then its conductivity can be calculated. Similarly, a capacitor can represent a double layer capacitance, which can be useful for determining the electrochemically active surface area of an electrode. However, extracting the value of a parameter such as a resistance or capacitance usually requires the entire impedance spectrum to be modelled, which can be challenging for more complex systems. In this manual, a program for the Determination of Equivalent Circuit Models (DEC*i*M) is presented, which can be used to model impedance spectra.

1.2 DEC*i*M's purpose

DEC*i*M is designed to allow both expert and non-expert users to adequately model impedance spectra. The program offers manual and automatic fitting, three different methods for creating equivalent circuit models, data validation, and different options for visualization. For non-expert users, there is a simple workflow consisting of drawing a circuit or typing a circuit string followed by manual fitting and a quick 'simple refinement'. For expert users, there is Z-HIT data validation, and there is the option to define a custom model (which can be a transmission line model or even a non-circuit model). Additionally, one may separately refine parameters in frequency ranges of choice with weighting schemes of choice. During and after data processing, there are three different types of plots available for visualization, with the option of including multiple data sets for comparison. Finally, the data and model can be saved to a result file which is human-readable and can be loaded by DEC*i*M.

2 Installation

2.1 Prerequisites

DEC*i*M is written in Python[1] and requires Python 3.10 or a more recent version of the Python interpreter. In addition, the following packages are required:

- NumPy[2]
- SciPy[3]
- Matplotlib[4]
- tkinter (Python Standard Library)
- functools (Python Standard Library)
- copy (Python Standard Library)
- webbrowser (Python Standard Library)

2.2 Installing DEC*i*M

Installing DEC*i*M requires 'DEC*i*M.py' and all 'ecm_*.py' modules to be placed in the same folder. These modules are the following files:

- ecm_circuits.py
- ecm_custom_models.py
- ecm_datastructure.py
- ecm_file_io.py
- ecm_fit.py
- ecm_helpers.py
- ecm_history.py
- ecm_manual.py
- ecm_plot.py

- ecm_user_input.py
- ecm_zhit.py

In addition, a file containing circuit presets, 'ecm_presets.decim_circuits' should be present.

2.3 Starting DEC*i*M

After the installation has been completed, DEC*i*M can be started by – for example – typing 'python DEC*i*M.py' on the command line and pressing RETURN/ENTER.

3 Equivalent circuit model selection

Before data can be loaded into DEC*i*M, it is necessary to define an equivalent circuit model. There are multiple ways to do this: typing a circuit string, choosing a circuit preset, drawing a circuit or defining a custom model. Once an equivalent circuit model has been chosen, data can be loaded and the model parameters' values can be determined.

3.1 Typing a circuit

A circuit can be described in text as a circuit string following a notation similar to that of Boukamp.[5] In DEC*i*M, the symbols R, L, C, Q, O, S, and G are used to refer to the resistor, inductor, capacitor, constant phase element, open Warburg, short Warburg and Gerischer elements, respectively. They can be connected in series by placing them between curly brackets '{}' or in parallel by placing them between parentheses '()'. Two examples are given in Figure 1.

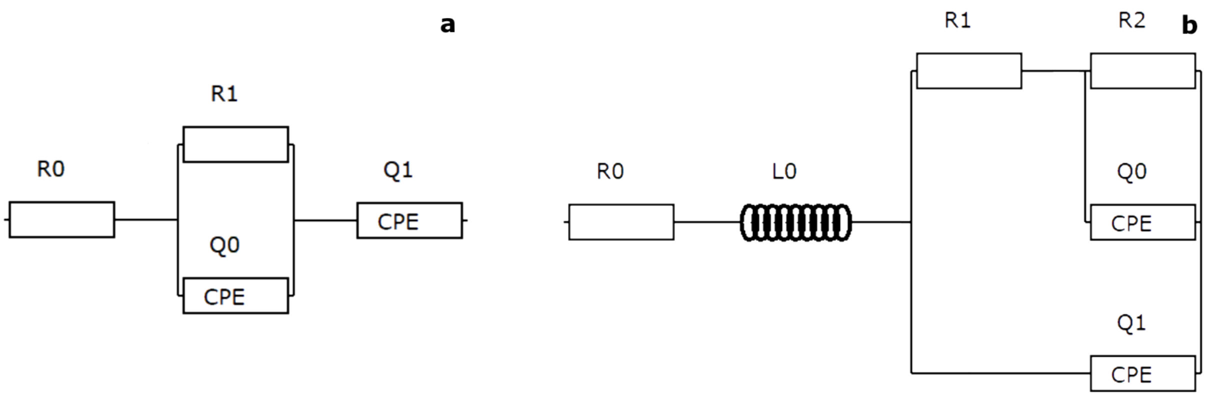

Figure 1: Two example circuits. a) $\{R_0(R_1Q_0)Q_1\}$; b) $\{R_0L_0(Q_1\{R_1(R_2Q_0)\})\}$.

The circuit typing input box can be accessed via the 'Circuit' menu as 'Type circuit'. A new window will be opened in which the circuit string canbe typed. When the window is closed, the text is read and validated. If the text is a valid circuit string, then the circuit is updated.

3.2 Circuit presets

Circuit strings can also be typed in the circuit_presets.decim_circuits text file. One circuit string can be entered on each line. On program startup, DEC*i*M reads this file and places every valid circuit string in the 'Circuit' menu, from where the circuit can be selected.

3.3 Drawing a circuit

It is possible to draw a circuit diagram in DEC*i*M by clicking the circuit button below the slider or by going to the 'Circuit' menu and clicking 'Draw circuit'. A new window will open in which on the left, there are some controls, and on the right, there is a drawing canvas. In the top left, the connection mode (series or parallel) is indicated. Drawing in series mode will autmatically connect elements drawn to the left or right of each other on the canvas, whereas drawing in parallel mode will automatically connect elements drawn above or below each other. This way, the elements are grouped into so-called *units*: groups of elements connected in parallel or series. When elements that were drawn in different modes need to be connected, their units must be merged by activating merge mode. This will create a new unit into which the smaller units are merged. The connection mode determines how the smaller units are connected.

To draw a new element or to merge units, you must click on the drawing canvas. If merge mode is enabled and there are units that can be merged, the highlighted units will be merged. If merge mode is disabled, clicking on the canvas will place the element on the canvas at the mouse position (first element) or at the closest available '+' position (all other elements). To illustrate the drawing process, an example in which the RL(RQ) circuit is drawn, is given below in Figures 2–11.

Figure 2: First, switch to series mode by clicking 'Series/parallel mode', then select a resistor from the element dropdown.

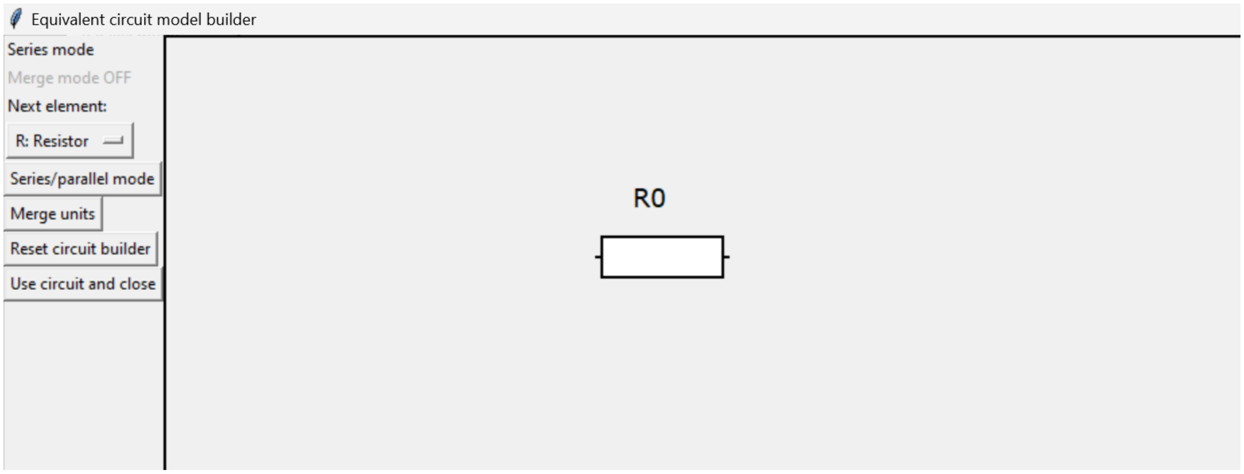

Figure 3: Draw the resistor by clicking on the canvas.

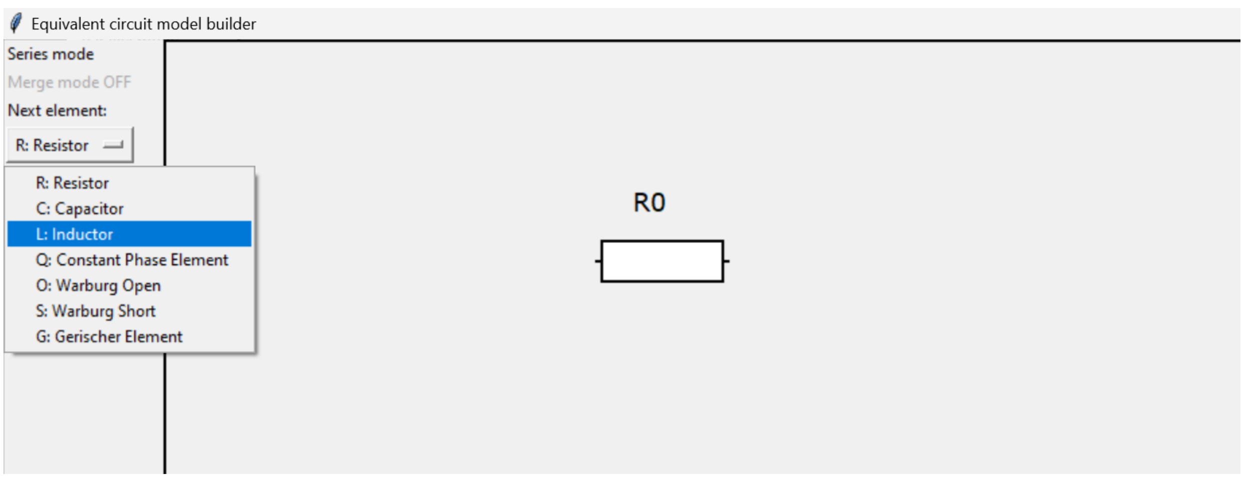

Figure 4: Select an inductor and draw it on the canvas by clicking to the right of the resistor.

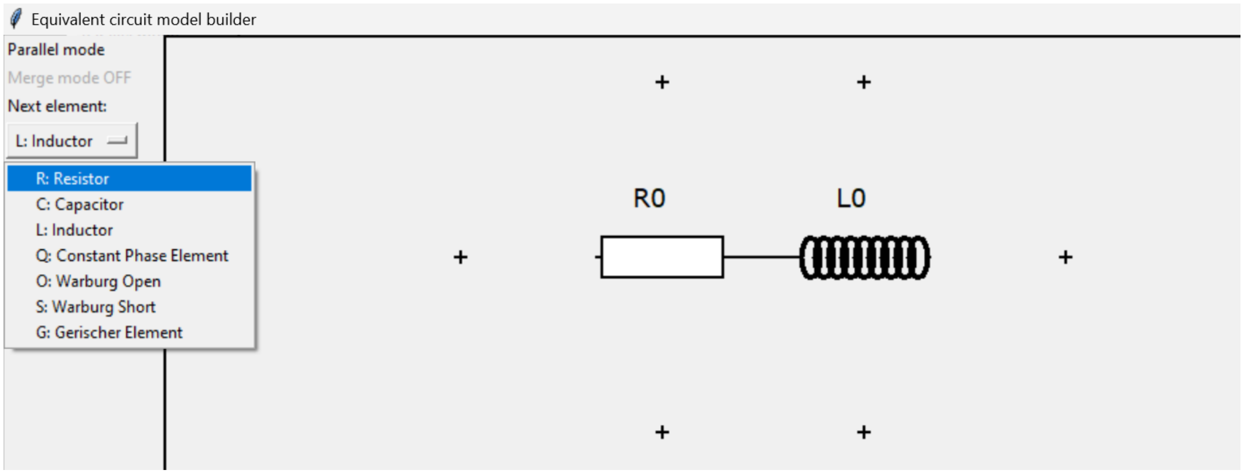

Figure 5: Switch to parallel mode and select a new resistor.

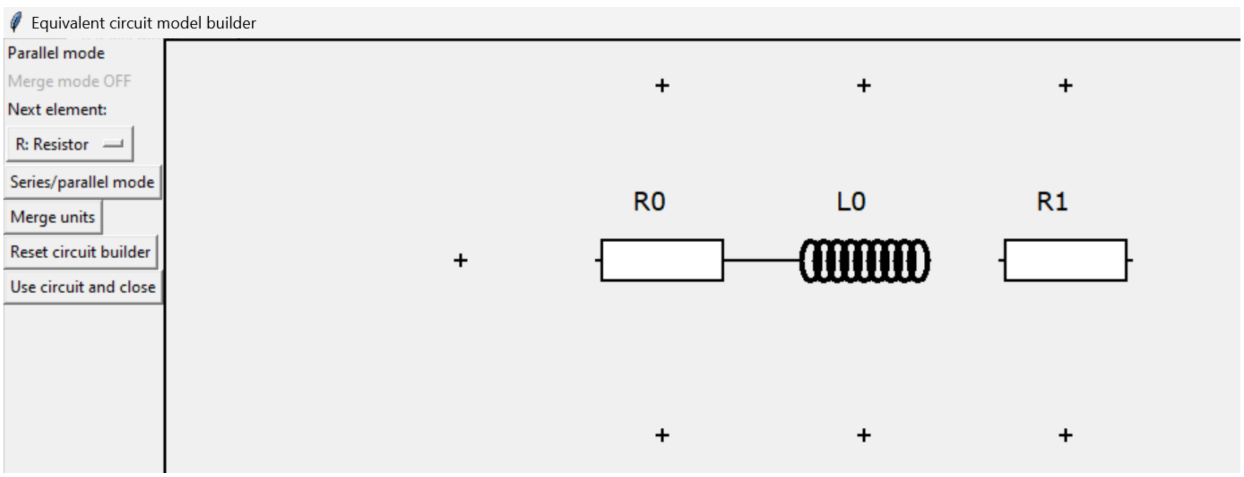

Figure 6: Draw the resistor by clicking to the right of the inductor. The resistor is in a new parallel unit and is not connected to the other elements, which are in a series unit.

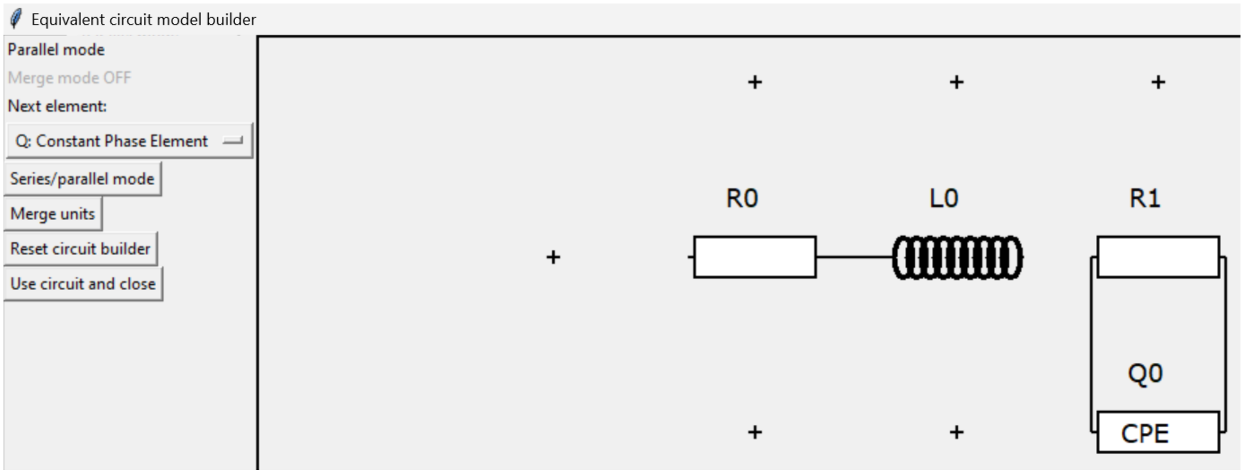

Figure 7: Select a constant phase element from the element dropdown and draw it below the new resistor. It will be automatically added to the same parallel unit, and the connections will be drawn.

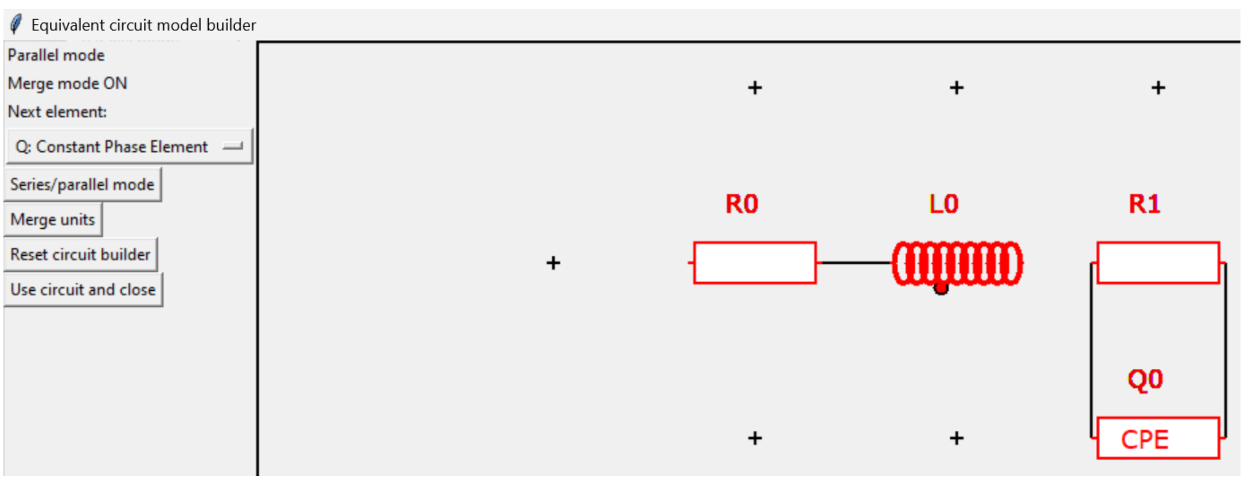

Figure 8: Enable merge mode by clicking 'Merge units' and move the mouse to the red dot. Click on or near it to try to merge the units.

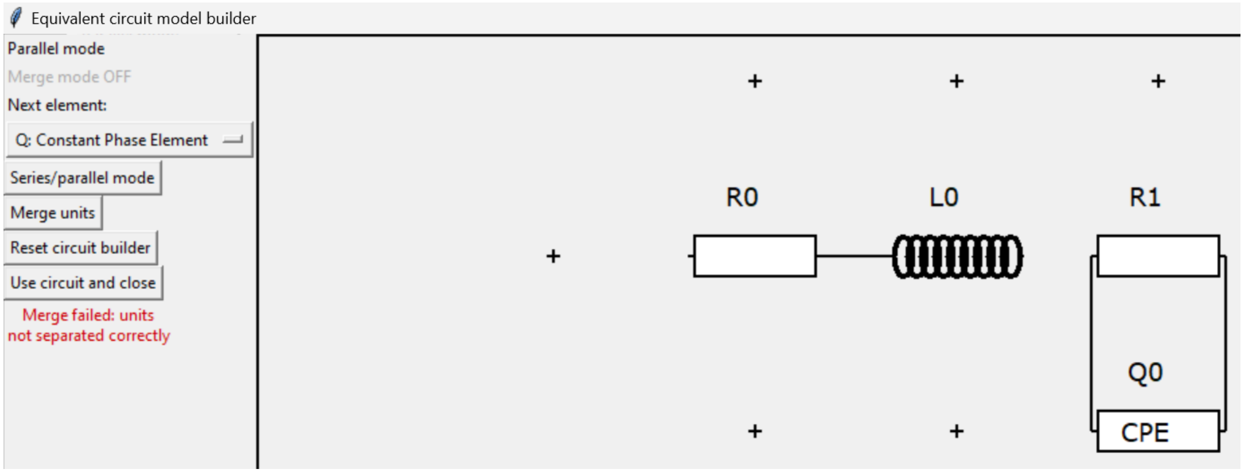

Figure 9: The units are not merged because they are only horizontally separated. To merge two units in parallel, they must be vertically separated. Similarly, merging two units in series requires them to be horizontally separated.

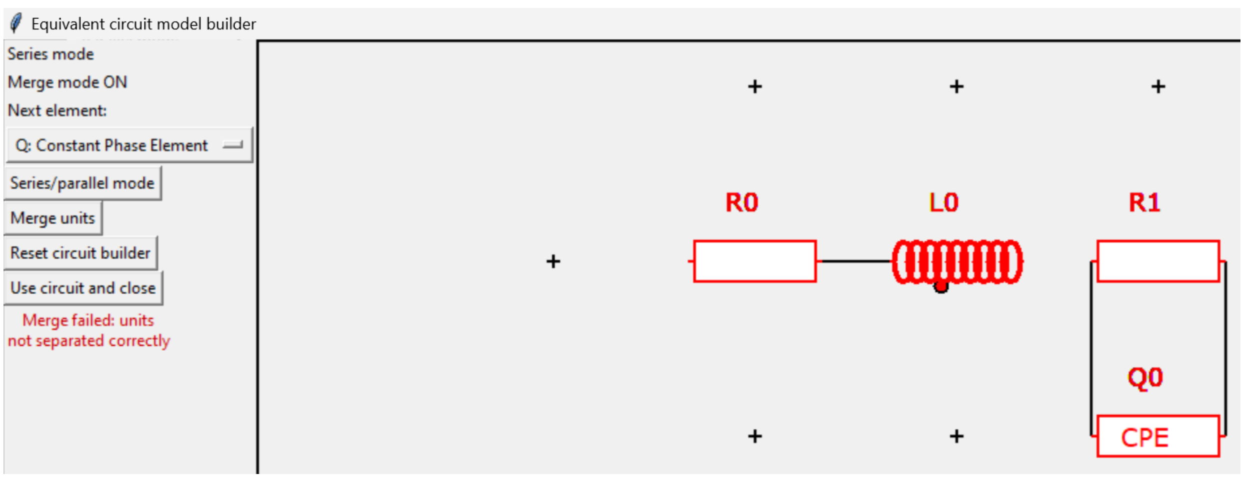

Figure 10: Try again in series mode.

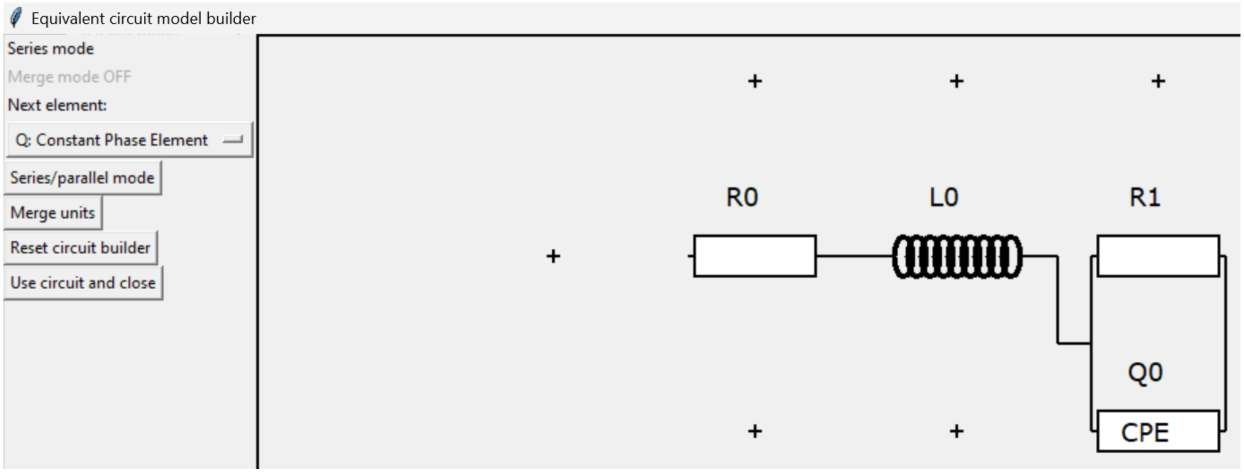

Figure 11: The two units are now merged and the circuit is complete.

With the circuit made, click ‘Use circuit and close’ to select it as the model for the analysis.

3.4 Custom models

Custom models can be defined in ‘ecm_custom_models.py’. Custom models are functions that return the impedance based on the fit parameters, but do not necessarily involve any circuit diagram. This is useful for different physical models of the impedance, or for transmission line models, which cannot be defined using the circuit typing or drawing interfaces. A custom model can be created as follows:

1. Create a function Z(fp, freq), where fp is a list of fit parameters and freq a NumPy array of frequencies. The function should return a NumPy array of complex impedances.
2. Add the function to the custom_model_diagrams dictionary as *‘name’: (‘diagram’, Z)*, where ‘diagram’ is a circuit string that contains as many parameters as the model uses.

Once the model has been defined, it can be accessed from the ‘Circuit’ menu, where the name entered into the custom_model_diagrams dictionary will be displayed.

4 Loading and saving files with impedance data

4.1 Data files

Data files can be loaded from the ‘File’ menu in the menu bar at the top of the program window. DECiM expects data files to be text files consisting of columns separated by whitespace characters (tabs, spaces) or commas. The first line can be a header specifying the quantities and units; DECiM will skip any lines that do not consist solely of numbers, separators and line breaks. On every line, the first value should be a linear frequency (f, **not** $\omega = 2\pi f$) in units of Hz. The second value should be the real impedance component Re$[Z]$ in units of Ω and the third value should be the imaginary impedance component Im$[Z]$ in units of Ω. The imaginary component should **not** be multiplied with -1; it is not Z'' as shown in complex plane plots.

Data files can only be loaded by DEC*i*M, not saved. Even if the data are tranformed with Z-HIT, it is not possible to save a new data file. Only a result file (.recm2) may be saved.

4.2 Result files

Result files (.recm2 file extension) may be saved and loaded by DEC*i*M, via the 'File' menu. They contain more information than data files: besides data, also limited statistical information, model parameters and points to plot the model are provided. The structure of a result file is as follows:

1. '>CIRCUIT DEFINITION' header.
2. Text describing the circuit diagram (circuit string).
3. '>MODEL PARAMETERS' header.
4. List of model parameters' names, values and indices; the latter are important for fitting.
5. '>STATISTICAL DATA' header.
6. List of number of parameters, number of frequencies, degrees of freedom, observation-to-parameter ratio and the proprotionally weighted sum of the squares S_v (a measure of the goodness of fit).
7. '>IMPEDANCE DATA' header.
8. 'Frequency (Hz), Re(Z) / Ohm, Im(Z) / Ohm' header.
9. Impedance data as described by the above header.
10. '>IMPEDANCE FIT' header.
11. 'Frequency (Hz), Re(Z) / Ohm, Im(Z) / Ohm' header.
12. 500 points of impedance that describe the model, formatted as described by the above header.

Result files are designed to be human-readable and to be a useful step towards preparing publication-quality figures. It is not necessary to evaluate the impedance using the circuit diagram and parameter values after the result file is generated. The model can simply be plotted from the data points below the '>IMPEDANCE FIT' header.

5 Data validation

5.1 Z-HIT transform

For data validation, DEC*i*M provides the Z-HIT transform[6] in the 'Calculate menu'. Clicking 'Perform Z-HIT transform' will open a new window with complex plane and Bode plots in the upper half and four entry boxes and three buttons in the lower half. With the entry boxes, the desired frequency range for data validation can be selected, as well as the settings for the first part of the calculation. The buttons can be used to start the calculation and to close with or without accepting the result.

The algorithm for the Z-HIT transform is:

1. Fit a spline to interpolate the phase data. Via the entry boxes, you can set how many points are used to plot the spline relative to the number of data points, as well as the smoothness of the spline. The lower the number in the smoothness box, the more the spline will follow the data. The default value of 0.3 is often too smooth; 0.03 or even 0.01 can work better. For a more complete description, see the SciPy[3] documentation for the UnivariateSpline.[7]
2. Calculate $\ln|Z|$ from the phase spline.
3. Determine the constant of integration in the Z-HIT transform.
4. Display the result.

If you only want to inspect the Z-HIT transform, you should close the window with the 'Reject and close button'. If you wish to use the transformed data instead of the measured data, click 'Accept and close'. Note that the number of transformed data points will be made equal to the number of measured data points upon accepting the transformation.

6 Manual parameter adjustment

The user interface of the main DEC*i*M window contains controls for manual parameter adjustment below the plots. These controls allow the selection of parameters, modification of parameters via a slider and directly setting a parameter's value by typing it. The controls used for this are shown in Figure 12.

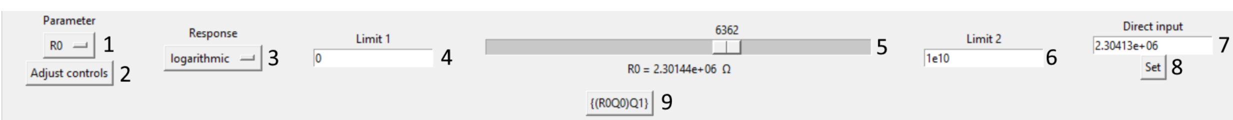

Figure 12: The controls for manually changing parameters' values. 1) Parameter dropdown. 2) Slider controls adjustment button. 3) Slider response dropdown. 4) First slider limit (can be upper or lower limit). 5) Slider. 6). Second slider limit. 7) Direct input box. 8) Parameter value set button. 9) Circuit selection button.

6.1 Parameter dropdown

The parameter dropdown is an option menu that, when clicked, displays a list of all parameters in the current equivalent circuit model. Clicking a parameter in this list makes it adjustable by the slider and 'Direct input' box. If the equivalent circuit model is changed, the parameter dropdown is automatically updated.

6.2 Slider controls

After a parameter has been chosen, it can be adjusted by the slider. It is, however, important to set the lower and upper limits of the parameter first. For example, resistors typically take values between 0 Ω and 10^{10} Ω, while capacitors normally take values between 1 F and 10^{-12} F. These limits can be set in the 'Limit 1' and 'Limit 2' text boxes. However, note that for values between

0 and 1 on a log-scaled axis, 'Limit 1' should be 0, not 1; 'Limit 2' should be the lower limit (*e.g.* 10^{-12}). After the limits have been set correctly, the slider response can be set with the menu under 'Response'. Usually, a logarithmic response is appropriate, but in the case of constant phase element (CPE) exponents, a linear response is often better.

To save time in setting up the slider limits and response, the 'Adjust controls' button can be used to automatically set up the slider controls. The limits and slider response will be set to typical values for the selected parameter. For any parameter which has units of Ω (R, O, S, and G), 'Limit 1' will be 0, 'Limit 2' will be 10^{10} and the response will be logarithmic. For C, Q, and L, 'Limit 1' will be 0, 'Limit 2' will be 10^{-12} and the response will be logarithmic. For the second parameter of O, S, and G (k, l, and m, respectively), 'Limit 1' will be 0, 'Limit 2' will be 10^6 $s^{-1/2}$ and the response will be logarithmic. Finally, for a CPE exponent n, 'Limit 1' will be 0, 'Limit 2' will be 1 and the response will be linear.

6.3 Slider and Set button

With the slider controls ready, the value of the selected parameter can be changed through the slider in three different ways:

1. Dragging the slider with the left mouse button pressed. The slider will move quickly.
2. Holding the left mouse button while the cursor is to the left or right of the button inside the slider. The slider will move slowly.
3. Clicking at any point on the slider bar with the right mouse button. The slider will immediately move to the indicated position.

The value of the parameter will be automatically updated and displayed below the slider. The plots will also be updated.

As an alternative to the slider, the parameter value may also be updated with the 'Direct input' text box. To do this, first enter the desired value of the parameter and then press the 'Set' button. The parameter's value will immediately be changed to the value you entered and the plots will be updated.

6.4 Guidelines for manual fitting

Manual fitting can be difficult and time-consuming, especially for more complex spectra. Fortunately, there is a systematic approach that works for many spectra. It works as follows:

1. Set all capacitors C and CPEs Q to 1 F, CPE exponents n to 1, inductors L to 10^{-12} H, and resistors R to 1 Ω. Set Warburg and Gerischer elements O, S, and W to 1 Ω as well and don't touch the parameter influencing the time constant (k, l, and m).
2. Estimate the resistors' values from the ends of the semicircles in the complex plane and the heights of the $|Z|$ plateaus in the Bode plot.
3. Decrease the C and Q values of the capacitors and CPEs that are connected in parallel with the resistors until semicircles start to appear. Make sure the phase and amplitude are fitted well.

4. For CPEs, adjust the exponent n until the fit is as close as possible.
5. Increase L until the fit no longer improves.
6. For Warburg and Gerischer elements, estimate O, S, or G from the real part of the electrode response. Then carefully adjust k, l, or m until the electrode response is fitted well.
7. Fine-tune all parameters.

7 Automatic model refinement

In many cases, it is useful to automatically refine the parameters obtained by manual fitting. DEC*i*M includes two options for this: the simple and advanced refinement. These can be found in the 'Calculate' menu in the menu bar.

7.1 Simple refinement

The simple refinement refines all parameters without applying any weights. It is mainly useful for simple circuits. Only the frequency range can be controlled; this is done with the 'Set simple refinement frequency range' option in the 'Calculate' menu. After clicking 'Refine solution (simple)', the refinement will immediately begin, and the text 'Refining...' will be displayed in the complex plane plot. Do not click anywhere inside the window while the refinement is running; this could cause DEC*i*M to become unresponsive. However, the refinement is usually fast, as its length is fixed to 10000 iterations. After the refinement is completed, the parameters and plots are automatically updated.

7.2 Advanced refinement

The advanced refinement can be accessed via the 'Advanced refinement...' option in the 'Calculate' menu. Clicking this option will launch a new window with three plots and three columns of controls. The plots are the residuals, the complex plane plot and the Bode plot of the impedance spectrum. The controls are shown in Figure 13.

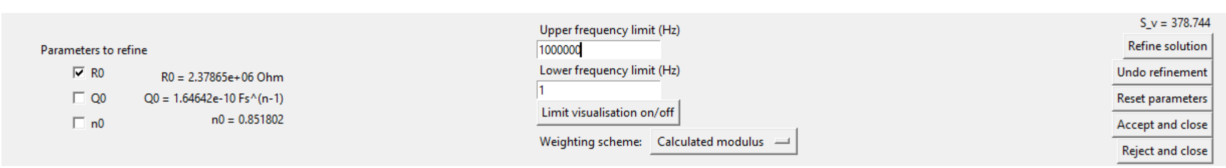

Figure 13: The refinement controls.

It is possible to select and deselect parameters to be refined, to set the frequency range and to choose a weighting scheme. The frequency limits can be shown in the residuals and Bode plots with the 'Limit visualisation on/off' button. The available weighting schemes are 'Unit' (no weights), 'Observed modulus' (modulus weighting based on measured data), 'Calculated modulus' (modulus weighting based on model), 'Observed proportional' (proportional weighting based on measured data), and ‘Calculated proportional’ (proportional weighting based on model). Unit weighting (the default) is fast, but not as accurate as modulus or proportional weighting. Proportional weighting is slow and can run into problems when the imaginary impedance component is close to 0. Therefore, modulus weighting is recommended, with modulus weighting based on the model (‘Calculated modulus’) typically being the best performing option.

Once all desired parameters have been selected, the limits have been set and a weighting scheme has been chosen, the solution can be refined with the ‘Refine solution’ button. If the result is not as desired, then the parameters can be reset with ‘Undo refinement’ to go back to the previous refinement result, or ‘Reset parameters’ to go back to the parameters as they were when the window was launched. If no satisfactory result can be obtained at all or you wish to cancel the refinement, then ‘Reject and close’ will close the window and discard the refinement result. If, however, an acceptable result is obtained, ‘Accept and close’ will update the model parameters in the main part of DEC*i*M and update the plots there.

7.3 Undoing refinements

In case a simple or advanced refinement leads to an undesirable result, the refinement can be undone with the ‘Undo refinement’ button in the ‘Calculate’ menu. It is possible to go back more than once; after every refinement, the parameters are saved and can be recovered. This is not done for slider movements, as these are more easily undone by hand.

8 Plotting options

By default, DEC*i*M displays two plots: a complex plane plot and a Bode plot that shows both the amplitude $|Z|(\omega)$ and the phase $\phi(\omega)$. These can be customized in various different ways.

8.1 Plotting multiple data sets

Multiple data sets can be plotted via the ‘History’ menu. If one data set is loaded, then saved with ‘Save current dataset and sample dimensions to history’ and then a new data set is loaded, the old data set can be plotted alongside the new one with ‘Plot or remove non-interactive dataset (max. 3)’. This can be done with up to three additional data sets. Additionally, you can switch between different data sets for analysis with ‘Select other dataset’; this is equivalent to loading a result file, but then from RAM.

8.2 Marking specific frequencies

Frequencies that are integer powers of ten (*e.g.* 0.1 Hz, 10 Hz, 100 kHz, etc.) can be highlighted in the complex plane plot with the ‘Mark frequencies that are integer powers of 10’ option under ‘Plot’. The points belonging to the frequencies in question (or those with the best matching frequencies for frequencies that are not present exactly) will be circled and the frequencies will be indicated in text beside them. Clicking the marking option again will turn all of this off again.

8.3 Switching between Bode and real/imaginary admittance plots

The Bode plot can be changed to a plot in which the real and imaginary admittance components (Y', Y'') are shown. Choosing 'Toggle Bode amplitude+phase impedance/real+imaginary admittance' in the 'Plot' menu will result in the (Y', Y'') plot being displayed.

8.4 Other plotting options

There are several other options in the 'Plot' menu. These are:

- 'Toggle data visibility': turn measured data on/off in both plots.
- 'Toggle model visibility': turn model curve(s) on/off in both plots.
- 'Reset view': reset the view limits of both plots.
- 'Toggle amplitude/real admittance log scale': change the axis scaling of $|Z|$ or Y' from linear to logarithmic or vice versa.
- 'Toggle phase/imaginary admittance log scale': change the axis scaling of ϕ or Y'' from linear to logarithmic or vice versa.
- 'Toggle amplitude/real admittance visibility': turn $|Z|$ or Y' on/off.
- 'Toggle phase/imaginary admittance visibility': turn ϕ or Y'' on/off.

There is also a toolbar below the plots (which also features in the Z-HIT and refinement windows). This allows zooming, panning, adjusting the shape of the plots and the saving of image files.

9 Other functions

9.1 Apex frequencies

The 'Get apex frequencies' option in the 'Calculate' menu will open a new window in which the frequencies at which maxima in the complex plane plot are found. The frequencies given here are angular frequencies $\omega = 2\pi f$, not linear frequencies f (which are displayed everywhere else in DECiM).

9.2 Help menu

The 'Help' menu has only two options:

- 'Show instructions', which will display brief instructions for how to use the program.
- 'Open manual', which will open the PDF manual in a web browser.

10 Copyright information

DEC*i*M is distributed under the MIT license.

10.1 DEC*i*M license

11 Citing

11.1 Scientific publication

Please cite the publication referenced on the DEC*i*M download page if you publish work in which you made use of DEC*i*M for data analysis.

1. Python Software Foundation. Python 3, 2023.
2. Harris, C. R.; Millman, K. J.; Walt, S. J. van der; Gommers, R.; Virtanen, P.; Cournapeau, D.; et al. Array programming with NumPy. *Nature* **2020**, *585*(7825), 357–362.
3. Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **2020**, *17*, 261–272.
4. Hunter, J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* **2007**, *9*(3), 90–95.
5. Boukamp, B. A. *Equivalent Circuit Users Manual*; 1989.
6. Schiller, C. A.; Richter, F.; Gülzow, E.; Wagner, N. Validation and evaluation of electrochemical impedance spectra of systems with states that change with time. *Phys. Chem. Chem. Phys.* **2001**, *3*, 374–378.

7. SciPy contributors. SciPy API reference: UnivariateSpline, 2023.