Advance Publication by J-STAGE

Electrochemistry

The Electrochemical Society of Japan

Received: January 14, 2021
Accepted: January 19, 2021
Published online: January 26, 2021

https://doi.org/10.5796/electrochemistry.21-00010

Note

Electrochemistry, (in press)

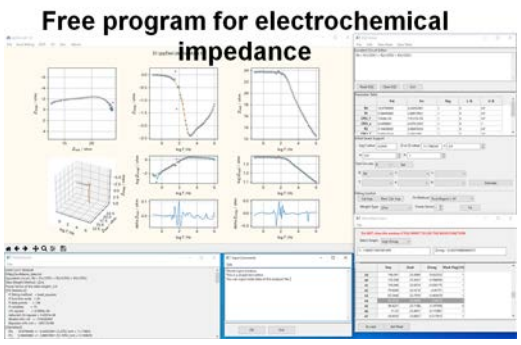
Free Analysis and Visualization Programs for Electrochemical Impedance Spectroscopy Coded in Python

Kiyoshi KOBAYASHI* **and Tohru S. SUZUKI**

Research Center for Functional Materials, National Institute for Materials Science, 1-2-1 Sengen, Tsukuba, Ibaraki 305-0047, Japan

* *Corresponding author: KOBAYASHI.kiyoshi@nims.go.jp*

K. Kobayashi orcid.org/0000-0001-9644-1879


Free program for electrochemical impedance

Free analysis and visualization programs for electrochemical impedance spectroscopy coded in Python

Kiyoshi Kobayashi Tohru S. Suzuki

Research Center for Functional Materials, National Institute for Materials Science

Sengen 1-2-1, Tsukuba, Ibaraki 305-0047, Japan

Corresponding Author: Kiyoshi Kobayashi

E-mail: KOBAYASHI.kiyoshi@nims.go.jp

Abstract

New programs for fitting using an equivalent circuit model and data visualization of electrochemical impedance spectroscopy were developed using Python and its open-source libraries. Executable programs have a graphical user interface and can run on a Microsoft Windows operating system without utilizing other platform applications. Many practical functions were implemented, e.g., supporting various file formats, graphical support functions to obtain a good initial guess, display interactive three-dimensional plots of the spectrum, a modified logarithmic Hilbert integral transform test, display multiple spectra data, etc. The developed executable programs were released by free of charge for use under the MIT license.

Key words: electrochemical impedance, free programs, fitting, visualization

1. Introduction

Electrochemical impedance spectroscopy is widely used to analyze various electrochemical systems such as batteries, fuel cells, solar cells, and corrosions.[1] Owing to the considerable increase in remote work caused by the current COVID-19 crisis, free analysis programs have gained importance because they enable scientists, engineers, and students to analyze data at different places from within their laboratories and offices using their personal computers. Although it is recommended to use a downloadable program available free of charge in such scenarios owing to its easy installation and software license agreements, free programs for electrochemical impedance analysis are lacking. This is attributed to reasons such several free programs being coded under a user-hostile interface (character-based user interface)[2-5] and the strict limitation of readable data format regardless of the graphical user interface implemented in the programs.[6-9] These program architectures are considerably inconvenient for users who aim to conduct spectrum analysis with data files output by a software used to control a potentio-galvanostat. Therefore, it is important to develop free programs to implement a user-friendly interface for electrochemical impedance analysis.

We developed electrochemical impedance programs and released them under a free software license. These programs were coded in Python[10] and its open-source libraries.[11-20] In this note, typical characteristics and various functions of our free programs are reported.

2. Common characteristics of developed programs

The main functions of the developed programs are: (1) model fitting and display of single spectrum data and (2) the display of multiple spectra data. The developed programs are named pyZwx and pyZMultiPlot, respectively (Fig.1). pyZwx is developed to implement a support function for obtaining a good initial guess[21]; this function was originally developed by the Igor Pro (Wavemetrics, USA) macro program, which requires the use of interactive graphs of single spectrum data.[21] Although pyZwx has similar function to the Igor Pro macro program,[21] other commercial platform-applications are not necessary to use. Therefore, pyZwx can run on personal computers operating on Microsoft Windows system.

Fig. 1

Visualization of multiple spectra data is necessary for users to confirm a relative change caused by a difference in measurement conditions. pyZMultiPlot was developed for this demand. Visualization program similar to pyZMultiPlot does not exist as free program operatable by graphic user interface (GUI).

Source codes of pyZwx and pyZMultiPlot can be executed on Python (versions 3.6 to 3.9) combined with the libraries listed in Table 1.[11-18] WinPython distribution[22] was used to develop these programs. The developed executable programs have been released under an MIT license,[23] and it is possible to download them from the web site.[24] Since no installer program is used, users can run these programs by copying them to a folder on their computer, USB memory, and any storage devices.

Table.

Spectrum analysis and visualization could be performed by all computers listed in Table 2; however, 30–40 s were necessary to boot pyZwx and pyZMultiPlot on computers with relatively low-performance central processing unit (CPU). The screen space feels constrained on a full-WXGA (1366 × 768 pixels) monitor and therefore, a monitor with resolution higher than full-HD (1920 × 1080 pixels) is preferred.

Table.

PyZwx and pyZMultiPlot can import spectrum data files output by ZPlot®,[25] Gamry Framework™,[26] EC-Lab® text format (mpt),[27] Versa Studio,[28] and IviumSoft.[29] Further, it is possible to import the fitted result file output by pyZwx. For the mpt including multiple spectra data, pyZwx and pyZMultiPlot have a function that allows them to separate the mpt into individual spectrum files. Plain text with a one-line header for columns is possible to import. The tab, space, and comma are used as delimiters.

3. Main function of pyZwx: Fitting using an equivalent circuit

3.1.Data visualization of single spectrum data

After importing data, eight plots are automatically displayed on the main window (Fig. 1(a)). For the impedance plot, the unit lengths of the real and imaginary parts of the impedance (Z_{real} and Z_{imag}) axes are set at the same value automatically; this setting is maintained when users change the main window size. The relationship between the axes is retained in the three-dimensional (3D) plot by taking the logarithm of the frequency (f) – Z_{real} – Z_{imag} plot. In addition, this 3D plot can be rotated using a mouse (Mov. S1 in the Supporting Information).

Mov. S1

3.2.Modified logarithmic Hilbert integral transform test

The modified logarithmic Hilbert integral transform test of the impedance spectrum (Z-HIT)[30-32] is similar to the Kramers–Kronig (KK) transform test.[33] The KK transform test is a checking test wherein an impedance spectrum is satisfied with causality, linearity, and stability.[33] Satisfying the KK transform relation between angular frequency (ω) versus the real part and ω versus the imaginary part of an impedance spectrum indicates that the impedance spectrum is solvable using an equivalent circuit model. The conventional KK transform test has one disadvantage: it is impossible to apply to an impedance spectrum wherein the imaginary part of the impedance diverges at high and low frequency limits.[27] This disadvantage implies the conventional KK transform test cannot be applied to impedance spectra collected from batteries and capacitors as the imaginary parts of these spectra diverge with decreasing frequency. Further, it is impossible to test the spectrum including the effect of an inductor.

Ehm and Schiller overcame this disadvantage by employing an empirical relationship based on the Hilbert integral transform.[30-32] The empirical relationship was that an impedance spectrum can be reproduced using the sum of an isolated resistor with a convoluted integral of the admittance of a constant phase element. By employing this empirical equation, this transform test can be applied to all types of spectra such as batteries and capacitors. This transformation was named *Z*-HIT.

An obvious deviation was observed when a numerically calculated spectrum including the resistor connected in parallel with the inductor was checked by *Z*-HIT. Thus, the reliability of *Z*-HIT is slightly low when a spectrum including an element of a resistor connected in parallel with an inductor is used. One example is shown in Fig. 2. The *Z*-HIT results cannot reproduce the calculated spectrum using the equivalent circuit presented in Fig. 2(a) because an obvious deviation appears between the calculated spectrum and the *Z*-HIT in the frequency region where the effect of the resistor connected to inductor in parallel is predominant on the total impedance spectrum (Fig. 2(b)).

Fig. 2

3.3 Data mask function

When the measured values show a systematic deviation from the *Z*-HIT at a frequency region, the corresponding data points are removed from the complex nonlinear least square calculation using an equivalent circuit model. Further, data convoluted with large noise are preferably removed from the complex nonlinear least square calculations.[34] Owing to this necessity, a data mask function is implemented. Users can easily set the data to be removed from the fitting via a mouse operation on the GUI (Mov. S.1). Furthermore, masked data are visualized on graphs by changing the color of the marker (Fig. 3).

Fig. 3

3.4.GUI support function to obtain a good initial guess

Similar to the program coded by the Igor Pro macro language,[21] initial guess values can be collected by a mouse click on only the log f – Z_{imag}, log f – ΔZ_{imag}, log f – Z_{real}, or log f – ΔZ_{real} graphs. By combining partial impedance elements, sampling data, and assigning parameter names, the initial guess values are calculated visually by the GUI operation. The assigned parameter values can be modified via GUI operation by checking graphs such as parameter manipulation (Mov. S1). This process is the same with the Igor Pro macro program.[21]

Mov. S1

3.5.Complex nonlinear least squares

By employing the LMFIT library[18] for complex nonlinear least square engine, the user can fit the model under various settings. Each parameter can be set various conditions; variable or fix, lower and upper limit values. Using these settings, empirical parameters such as the α of CPE is constrained between 0 and 1. These settings can be achieved on the equivalent circuit editor (Fig. 4). Further, the nine algorithms listed in Table 3[12] can be selected as complex nonlinear least squares. By default settings, the trust-region Levenberg-Marquardt algorithm is employed because this algorithm appears to be the most suitable for the impedance spectrum fitting.

Fig. 4

Table. 3

The fitting report includes various statistical information that is displayed after fitting. This information includes the values of optimized parameters and their standard errors, residual sum of squares (χ^2), variance of residuals (reduced χ^2), Akaike's information criterion, Bayesian information criterion, correlation factor between two parameters, and covariance matrix (Fig. 1). These are the “as is” output of LMFIT.[18] The values of optimized parameters and their standard errors are displayed automatically on the parameter table shown in the equivalent circuit editor. The values in the table can be copied to the memory of the computer (clipboard) and pasted on the other software if required.

3.6. Metadata input

Users can input additional text information about the sample, preparation process, measurement condition, and so on for each analysis through input memo window (Fig. 1). This window is opened after the completion of complex nonlinear least squares. The input memo window is a simple text editor; therefore, there is no limitation on the word count and line numbers. This text information is saved in the fitted result file.

3.7.Output analyzed data sheet

When the user saves the fitted result from the equivalent circuit editor window, a data sheet with a text format is created. An extension of the fitted result file is “fit.” This file includes the file name, equivalent circuit model, metadata input, parameter information, original spectrum data, data used for least squares, the calculated data, difference between measured and calculated spectra, masked data, and covariance matrix of fitted parameters. This data sheet can be used as training data for machine learning because all information is included in one file.

4. Function of pyZMultiplePlot: Program for visualization of multiple data

PyZwx can display single spectrum data only because of the implementation of the GUI support function to obtain a good initial guess as explained previously. Often, the user needs to obtain multiple spectral data in the same graphs to understand the change in spectrum based on measurement conditions, e.g., changes in temperature, applied voltage, concentration of contents, and so on. Therefore, a data viewer program—pyZMultiPlot—was developed. Using this software, multiple data files can be selected on the file selection dialog, and then multiple data are displayed in the same graphs (Fig. 1). The addition of data makes it possible to import data files after displaying graphs. Inversely, the removal of data makes it possible to exclude the data selection window by GUI operation.

5. Conclusions

Analysis and visualization programs for electrochemical impedance spectroscopy were developed using Python and its open-source libraries. Many convenient functions can be used by these programs through GUI operation. Various format data are supported by default and therefore, user can import data without pre-processing. In addition, practically useful Hilbert transform test is also implemented. This test can be applied to the spectra collected from batteries and capacitors.

Although many valuable functions were successfully implemented, the speed of graphical response is insufficient. Therefore, code optimization and/or change in the GUI and data visualization libraries are necessary to solve this problem.

Supporting Information: The supporting Information is available on the website at DOI: xxxxxxxxxxxxx.

Acknowledgement

These programs are released under the MIT license and therefore, these are possible to use by free of charge. pyZwx and pyZMultiPlot are possible to download from the web site,[24] and to use by copy only.

Part of this work was supported by JSPS KAKENHI Grant Numbers 17H01317 and 19H02804, and JST A-Step Grant Number JPMJTS1617, Japan.

References

1. E. Barsoukov and J. R. Macdonald, *Impedance Spectroscopy, Theory, Experiment, and Applications*, John Wiley & Sons, New Jersey, p. 185 (2018).
2. M. D. Murbach, B. Gerwe, N. Dawson-Elli, and L. Tsui, *J. Open Source Software*. **5**, 2349 (2020). DOI: 10.21105/joss.02349.
3. M. D. Murbach, B. Gerwe, N. Dawson-Elli, and L. Tsui, impydance.py, https://zenodo.org/record/3955199#.X9_5iBZUuUk. DOI: 10.5281/zenodo.3955199.
4. K. B. Knudsen, *ECS Meet. Abstr*. **MA2019-01**, 1937 (2019)
5. K. B. Knudsen, pyEis, https://zenodo.org/record/2532068#.X9_5ARZUuUk. DOI: 10.5281/zenodo.532068.
6. A. S. Bondarenko, G. A. Ragoisha, in *Progress in Chemometrics Research*, p. 89 (Ed. A. L. Pomerantsev, Nova Sci. Pub., New York, 2005).
7. A. S. Bondarenko, G. A. Ragoisha, EIS Spectrum Analyzer, http://www.abc.chemistry.bsu.by/vi/analyser/.

8. M. Źic, V. Subotić, S. Pereverzyev, and I. Fajfar, *J. Electroanal. Chem.* **866**, 114171 (2020). DOI: 10.1016/j.jelechem.2020.114171

9. J. R. Macdonald, LEVMW, https://jrossmacdonald.com/.

10. Python Software Foundation, python, https://python.org/.

11. The wxPthon Team, wxPython, https://www.wxpython.org/.

12. J. D. Hunter, *Comp. Sci. Eng.* **9**, 90 (2007). DOI: 10.1109/MCSE.2007.55.

13. The Matplotlib development team, matplotlib, https://matplotlib.org/.

14. C. R. Harris, K. J. Millman, S. fan J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, *Nature*, **585**, 357 (2020). DOI: 10.1038/s41586-020-2649-2.

15. NumPy Developers, Numpy, https://numpy.org/.

16. P. Virtanen1, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, *Nature Methods*. **17**, 261 (2020). DOI: 10.1038/s41592-019-0686-2.

17. SciPy developers, Scipy, https://www.scipy.org/.

18. N. Matthew, S. Till, A. Daniel B, I. Antonino, LMFIT, https://lmfit.github.io/lmfit-py/. DOI: 10.5281/zenodo.11813.

19. The Qt Company, Qt for Python, https://doc.qt.io/qtforpython/.

20. Riverbank Computing, pyQt5, https://www.riverbankcomputing.com/software/pyqt/.

21. K. Kobayashi and T. S. Suzuki, *Electrochem*. **88**, 39 (2020). DOI: 10.5796/electrochemistry.19-00058.

22. The WinPython Development Team, WinPython, https://winpython.github.io/.

23. Open source initiative, https://opensource.org/licenses/MIT.

24. K. Kobayashi, pyZwx, https://www.nims.go.jp/pyZwx/.

25. Scribner Associate, Zplot$^{®}$ and ZView$^{®}$, http://www.scribner.com/.

26. Gamry instruments, Gamry Framework^{TM}, https://www.gamry.com/.

27. Biologic, EC-Lab$^{®}$, https://www.biologic.net/.

28. Princeton Applied Research, Versa Studio, https://www.ameteksi.com/brands/princeton-applied-research/.

29. Ivium Technologies, IviumSoft, https://www.ivium.com/.

30. W. Ehm, http://zahner.de/pdf/zhitehm.pdf.

31. W. Ehm, H. Göhr, R. Kaus, B. Röseler, C. A. Schiller, *ACH-Models in Chem*. **137**, 145 (2000)

32. C. A. Schiller, F. Richter, E. Gülzow, and N. Wagner, *Phys. Chem. Chem. Phys*. **3**, 374 (2001). DOI: 10.1039/b007678n.

33. M. E. Orazem and B. Tribollet, *Electrochemical Impedance Spectroscopy*, John Wiley & Sons, New Jersey, p. 427 (2008).

34. K. Kobayashi, Y. Sakka, T. S. Suzuki, *J. Ceram. Soc. Jpn*. **124**, 943 (2016). DOI: 10.5796/electrochemistry.19-00058.

Table 1. Python libraries used for pyZwx and pyZMultiPlot.

Name	Function	Version	License
wxPython[11]	Toolkit of graphic user interface	4.1.1	wxWindows library license
Matplotlib[12,13]	Data visualization	3.3.3	matplotlib license
Numpy[14,15]	Scientific computing	1.19.3	BSD-3
Scipy[16,17]	Mathematics, science, and engineering computing	1.5.4	BSD-3
LMFIT[18]	Non-linear least-squares minimization	1.0.1	BSD-3

Table 2. Tested environments.

Operating system	Version	CPU and size of memory	Display resolution (pixels)
Windows 8.1 Pro	64 bit	Core i7-4500U, 8 GB	Full-HD (1920×1080)
Windows 10 Pro	32 bit	Celeron(R) CPU 2950M, 4 GB	Full-WXGA (1366×768)
Windows 10 Home	64 bit	Celeron(R) CPU 3855U, 8 GB	Full-WXGA (1366×768)
Windows 10 Pro	64 bit	Core i7-10810U, 16 GB	Wide Ultra-XGA (1920×1200)

Table 3. Selectable nonlinear least square algorithms.[18]

Name	Setting popup label	Setting
Trust-region Levenberg–Marquardt algorithm	Trust Region L–M	Default
Levenberg–Marquardt algorithm	L–M	Selectable
Nelder–Mead simplex algorithm	Nelder-Mead	Selectable
Modified Broyden–Fletcher–Goldfarb–Shanno algorithm	L–BFGS–B	Selectable
Powell algorithm	Powell	Selectable
Conjugate gradient algorithm	Conjugate Gradient	Selectable
COBYLA algorithm	COBYLA	Selectable
Broyden–Fletcher–Goldfarb–Shanno algorithm	BFGS	Selectable
Constrained trust-region algorithm	Constrained Trust–Region	Selectable

Figure captions

Figure. 1. Overview of (a) pyZwx after fitting and (b) pyZMultiPlot after reading data. Z_{real} and Z_{imag} are the real and imaginary parts of impedance. f represents frequency, and $|Z_{\mathrm{real}}|$ and $|Z_{\mathrm{imag}}|$ denote the absolute values of Z_{real} and Z_{imag}, respectively.

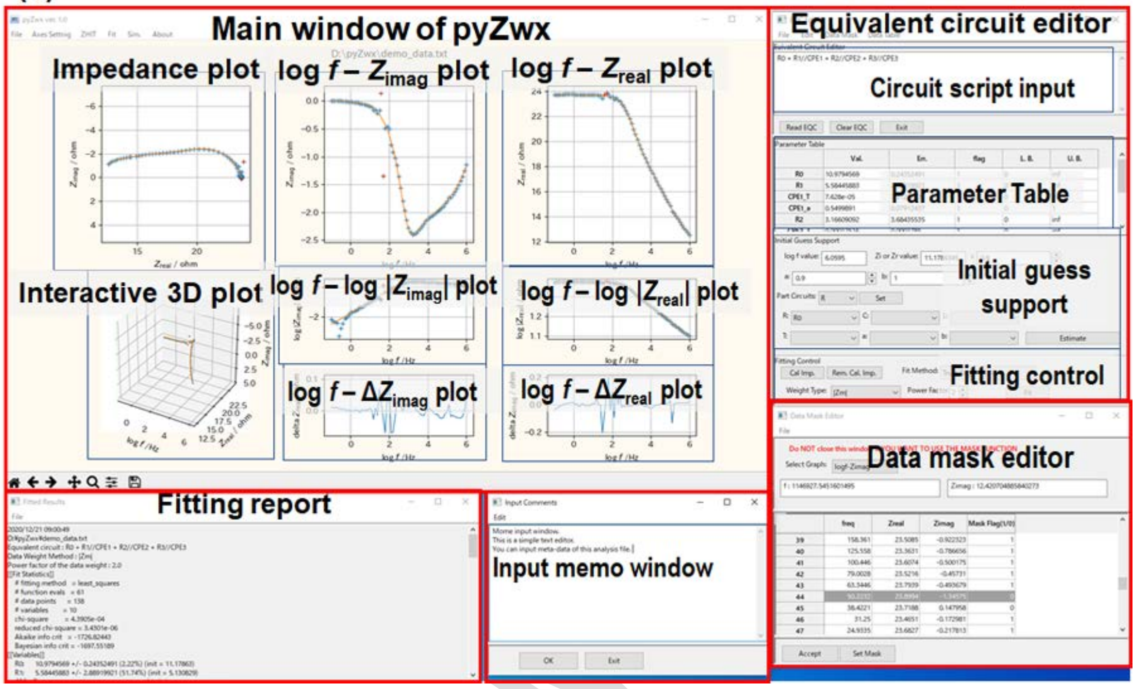
Figure. 2 Comparison of simulated spectrum and Z-HIT using simulated spectrum. The simulated spectrum is calculated using the model circuit shown by (a). The impedance plot of simulated spectrum (open circle) and Z-HIT result (red curve) is shown in (b). Obvious deviation observed in the dotted square region.

Figure. 3. Overview of pyZwx after setting the data mask. Blue and brown markers surrounded by red circles represent non masked and masked data points. Masked data points are removed from nonlinear least squares using an equivalent circuit model.

Figure. 4. Equivalent circuit editor window. Brief explanations of each column of the parameter table and popup of the initial guess support are indexed. The selector popup for nonlinear least squares algorithm is emphasized. Selectable algorithms are listed in Table 3.

Figure. 1. Overview of (a) pyZwx after fitting and (b) pyZMultiPlot after reading data. Z_{real} and Z_{imag} are the real and imaginary parts of impedance. f represents frequency, and $|Z_{real}|$ and $|Z_{imag}|$ denote the absolute values of Z_{real} and Z_{imag}, respectively.

(a)



R0 + R1//CPE1 + R2//CPE2 + R3//CPE3

	Val.	Err.	flag	L. B.	U. B.
R0	10.9794569	[illegible]	[illegible]	[illegible]	inf
R1	5.58445883	[illegible]	[illegible]	[illegible]	[illegible]
CPE1_T	7.628e-05	[illegible]	[illegible]	[illegible]	[illegible]
CPE1_a	0.5499891	[illegible]	[illegible]	[illegible]	[illegible]
R2	3.16609092	3.68435535	1	0	inf

log f value: 6.0595
Zi or Zr value: 11.178
a: 0.9
b: 1
Part Circuits: R
R: R0
Fit Method:
Weight Type: |Zm|
Power Factor:

```
2020/12/21 09:00:49
D:¥pyZwx¥demo_data.txt
Equivalent circuit : R0 + R1//CPE1 + R2//CPE2 + R3//CPE3
Data Weight Method : |Zm|
Power factor of the data weight : 2.0
[[Fit Statistics]]
    # fitting method   = least_squares
    # function evals   = 61
    # data points      = 138
    # variables        = 10
    chi-square         = 4.3905e-04
    reduced chi-square = 3.4301e-06
    Akaike info crit   = -1726.82443
    Bayesian info crit = -1697.55189
[[Variables]]
    R0:   10.9794569 +/- 0.24352491 (2.22%) (init = 11.17863)
    R1:   5.58445883 +/- 2.88919921 (51.74%) (init = 5.130829)
```

Memo input window.
This is a simple text editor.
You can input meta-data of this analysis file.

Do NOT close this window IF YOU WANT TO USE THE MASK FUNCTION
Select Graph: logf-Zimag
f : 1146927.5451601495
Zimag : 12.420704885840273

	freq	Zreal	Zimag	Mask Flag(1/0)
39	158.361	23.5085	-0.922323	1
40	125.558	23.3631	-0.786656	1
41	100.446	23.6074	-0.500175	1
42	79.0028	23.5216	-0.45731	1
43	63.3446	23.7939	-0.493679	1
44	50.2232	23.8994	-1.34575	0
45	38.4221	23.7188	0.147958	0
46	31.25	23.4651	-0.172981	1
47	24.9335	23.6827	-0.217813	1

(b)

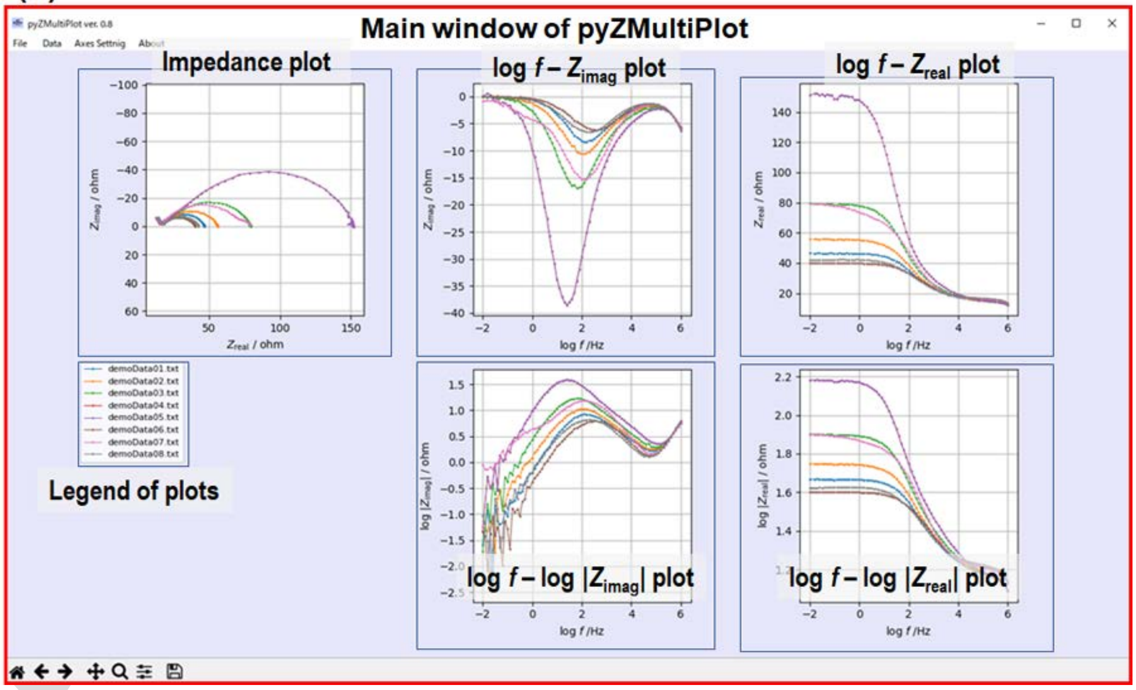
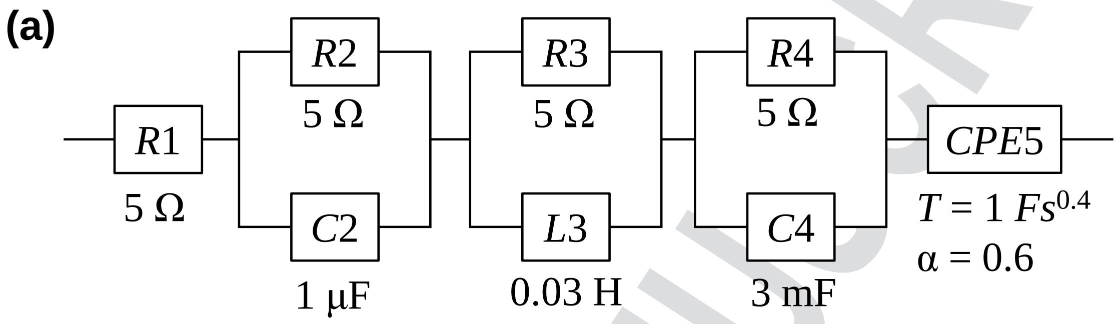


Figure. 2 Comparison of simulated spectrum and Z-HIT using simulated spectrum. The simulated spectrum is calculated using the model circuit shown by (a). The impedance plot of simulated spectrum (open circle) and Z-HIT result (red curve) is shown in (b). Obvious deviation observed in the dotted square region.



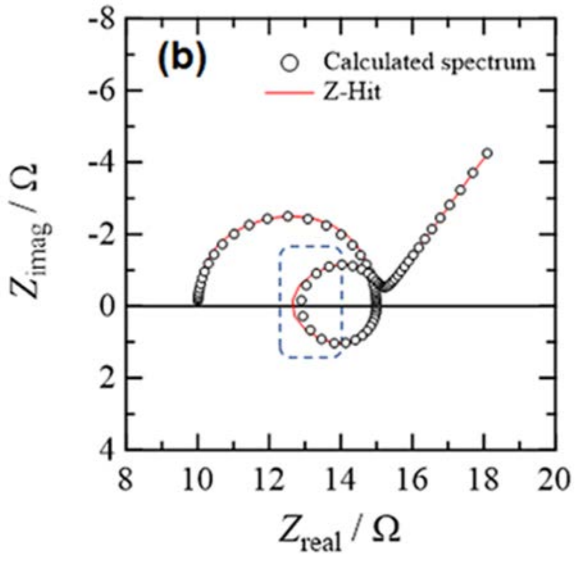


Figure. 3. Overview of pyZwx after setting the data mask. Blue and brown markers surrounded by red circles represent non masked and masked data points. Masked data points are removed from nonlinear least squares using an equivalent circuit model.

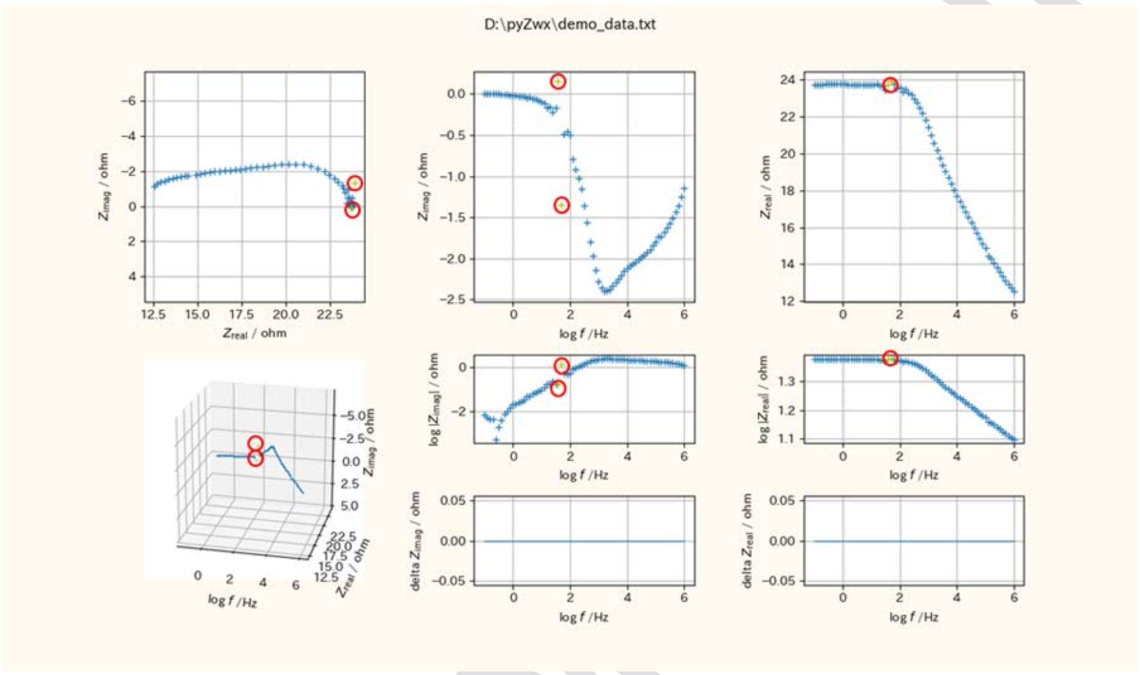
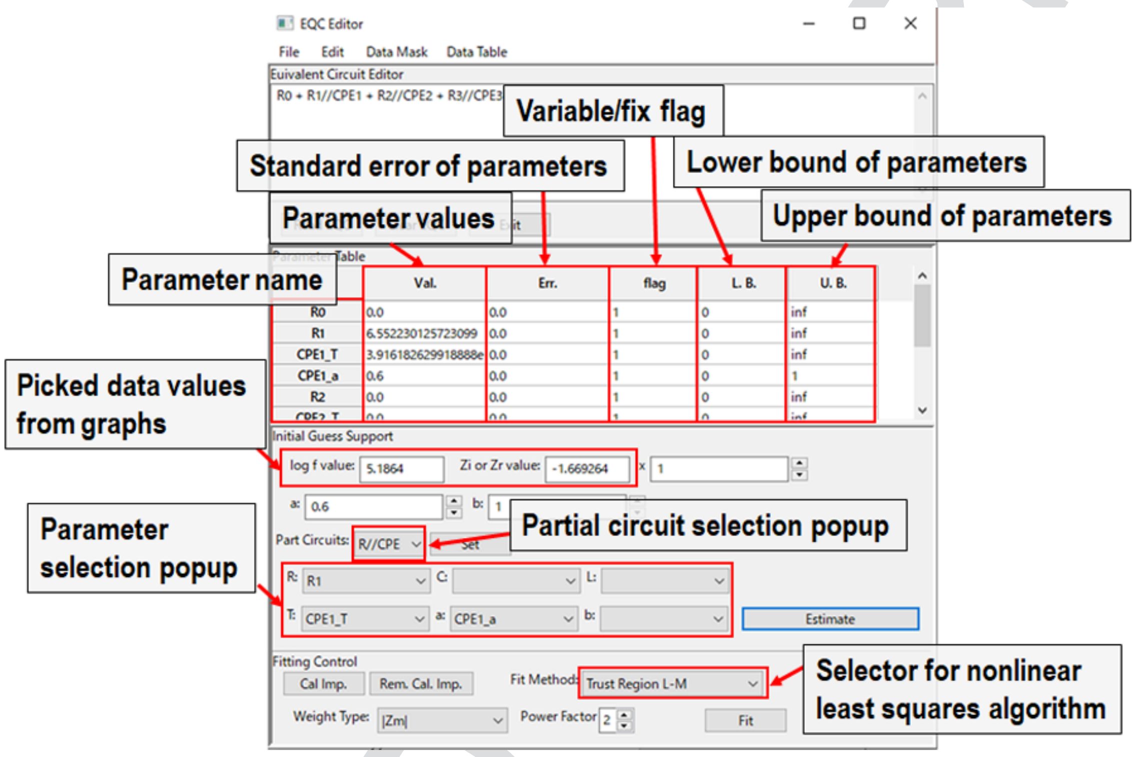


Figure. 4. Equivalent circuit editor window. Brief explanations of each column of the parameter table and popup of the initial guess support are indexed. The selector popup for nonlinear least squares algorithm is emphasized. Selectable algorithms are listed in Table 3.


EQC Editor
File
Edit
Data Mask
Data Table
Euivalent Circuit Editor
Variable/fix flag
Standard error of parameters
Lower bound of parameters
Parameter values
Upper bound of parameters
Parameter Table
Parameter name
Picked data values from graphs
Initial Guess Support
Parameter selection popup
Partial circuit selection popup
Set
Estimate
Fitting Control
Cal Imp.
Rem. Cal. Imp.
Fit
Selector for nonlinear least squares algorithm

R0 + R1//CPE1 + R2//CPE2 + R3//CPE3

	Val.	Err.	flag	L. B.	U. B.
R0	0.0	0.0	1	0	inf
R1	6.552230125723099	0.0	1	0	inf
CPE1_T	3.916182629918888e	0.0	1	0	inf
CPE1_a	0.6	0.0	1	0	1
R2	0.0	0.0	1	0	inf
CPE2_T	0.0	0.0	1	0	inf

log f value: 5.1864 Zi or Zr value: -1.669264 x 1

a: 0.6 b: 1

Part Circuits: R//CPE

R: R1 C: L:

T: CPE1_T a: CPE1_a b:

Fit Method: Trust Region L-M

Weight Type: |Zm| Power Factor 2