eistoolbox

for

MathWorks® MATLAB

A toolbox for batch fitting of Electrochemical Impedance Spectroscopy data to equivalent circuit models

User Guide

Juan J. Montero-Rodríguez

Version 0.1-93

Contents

Chapter 1

Introduction

eistoolbox is a toolbox for MATLAB® used for batch fitting Electrochemical Impedance Spectroscopy (EIS) data to equivalent circuits models.

In EIS the **impedance** of a sample is observed and recorded at different frequencies, and the impedance is stored as a Bode diagram, with magnitude and phase (polar format). Results can also be stored as real and imaginary parts (rectangular format) and plotted as a Nyquist diagram.

This impedance is commonly fitted to circuit models, selected depending on the physical characteristics of the measured system. The result is a list of **circuit elements** such as resistors, capacitors and inductors, and a list of **parameter values**.

The problem is how to calculate these parameters. This is often achieved using **nonlinear optimization**, and some of the most famous algorithms for fitting EIS data are based on the Complex Nonlinear Least Squares (CNLS) algorithm such as the Levenberg-Marquardt. However there are many algorithms for this end, and they are difficult to implement as they require advanced mathematics.

The present **eistoolbox** is a collection of algorithms for nonlinear optimization, and a graphical user interface for batch fitting of multiple EIS curves.

Chapter 2

Theory

2.1 Impedance Spectroscopy

Electrochemical Impedance Spectroscopy (EIS) measures the complex impedance of a sample as a function of the frequency. The experimental results are stored in two possible formats: polar coordinates (magnitude and phase) or rectangular coordinates (real and imaginary).

$$Z(f) = R + jX \tag{2.1}$$

where R is the resistance and X is the reactance of the sample.

The real part of the impedance is proportional to the resistivity, and the imaginary part is proportional to the permittivity. Both parameters can be calculated directly from the measurements, considering the exact geometry of the electrodes and measurement setup. For parallel plate electrodes, the following equations apply:

$$R = \frac{\rho L}{A} \tag{2.2}$$

$$C = \frac{\epsilon A}{D} \tag{2.3}$$

The capacitive reactance is given by

$$X_C = \frac{1}{2\pi f C} \tag{2.4}$$

Substituting (2.3) and (2.4) into (2.1) results in the following equation, which describes the impedance in terms of the resistivity and permittivity of the sample between parallel electrodes:

$$Z(f) = \frac{\rho L}{A} + \frac{1}{2\pi f}\frac{D}{\epsilon A} \tag{2.5}$$

Impedance Spectroscopy is also referred as Dielectric Spectroscopy, because it gives information about the dielectric properties of the measured sample.

2.2 Kramers-Kronig Transforms

Experimental data can be validated using the Kramers-Kronig Transforms.

2.3 Fitting algorithms

2.3.1 Levenberg-Marquardt

Chapter 3

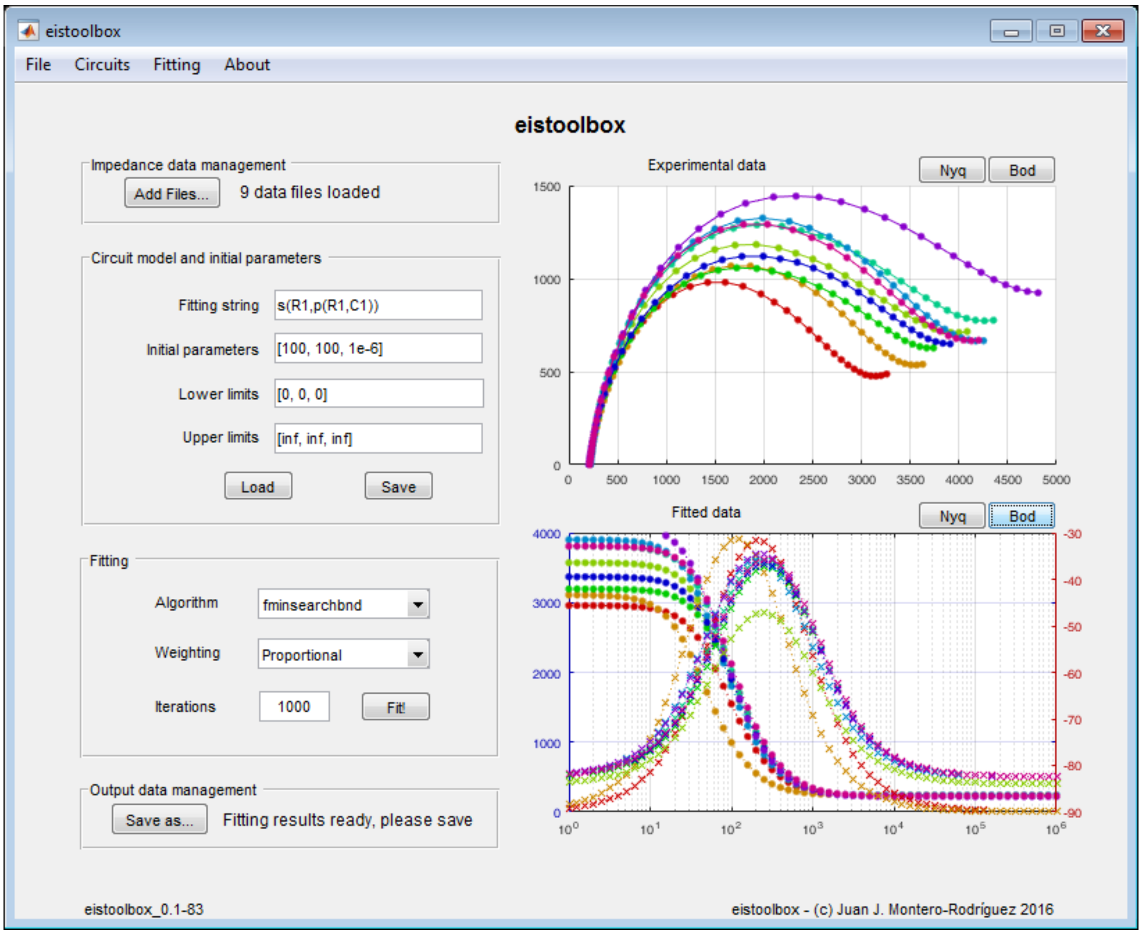
Quick Start Guide

Steps for batch fitting impedance data curves:

1. Add files using the "Add file..." button (supported .CSV or .DTA)
2. Write the circuit string and fitting parameters (circuit string formatting, see Chapter 3)
3. Click the "Fit" button
4. Save the results using the "Save..." button

3.1 Graphical User Interface

3.1.1 Main Window



Impedance data management

Add Files... 9 data files loaded

Circuit model and initial parameters

Fitting string	s(R1,p(R1,C1))
Initial parameters	[100, 100, 1e-6]
Lower limits	[0, 0, 0]
Upper limits	[inf, inf, inf]

Fitting

Algorithm	fminsearchbnd
Weighting	Proportional
Iterations	1000

Output data management

Save as... Fitting results ready, please save

eistoolbox_0.1-83

eistoolbox - (c) Juan J. Montero-Rodríguez 2016

3.1.2 Results Window

The output parameters after the fitting are displayed as a table, in the same order as they appear in the circuit string.

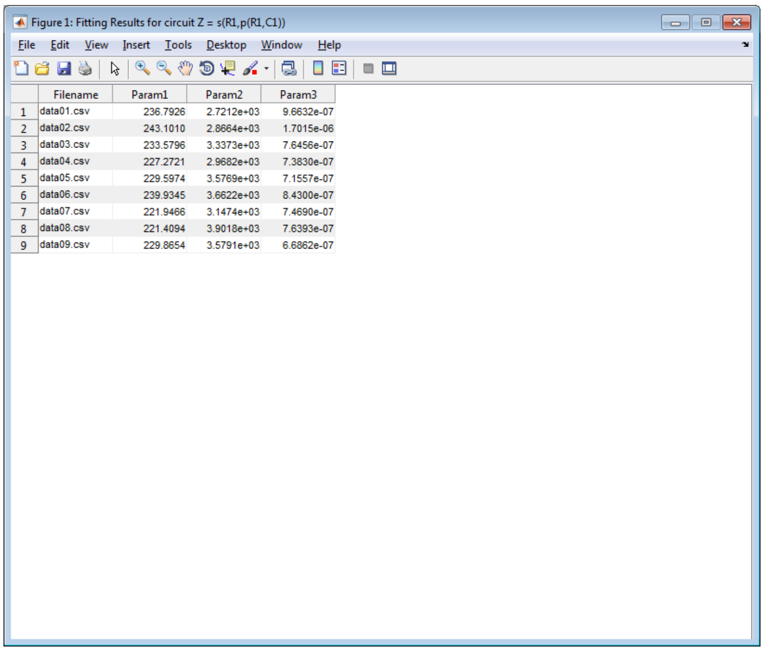


Figure 1: Fitting Results for circuit Z = s(R1,p(R1,C1))
File Edit View Insert Tools Desktop Window Help

	Filename	Param1	Param2	Param3
1	data01.csv	236.7926	2.7212e+03	9.6632e-07
2	data02.csv	243.1010	2.8664e+03	1.7015e-06
3	data03.csv	233.5796	3.3373e+03	7.6456e-07
4	data04.csv	227.2721	2.9682e+03	7.3830e-07
5	data05.csv	229.5974	3.5769e+03	7.1557e-07
6	data06.csv	239.9345	3.6622e+03	8.4300e-07
7	data07.csv	221.9466	3.1474e+03	7.4690e-07
8	data08.csv	221.4094	3.9018e+03	7.6393e-07
9	data09.csv	229.8654	3.5791e+03	6.6862e-07

3.1.3 Correlations Window

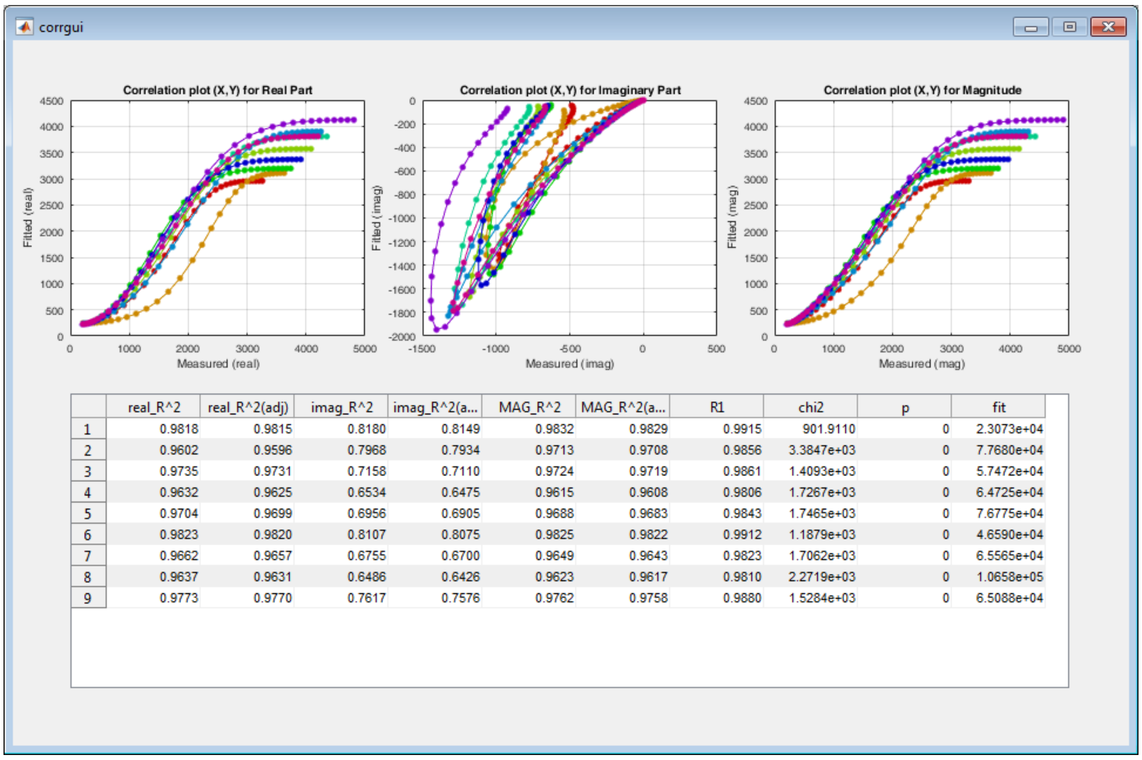
This window is useful to validate the fitting results.

There are three correlations implemented: real, imaginary and magnitude.

- The real correlation coefficient (rsq_real) is obtained by linear regression from the plot of the fitted vs. measured real part of the impedance.
- The imaginary correlation coefficient (rsq_imag) is obtained by linear regression from the plot of the fitted vs. measured imaginary part of the impedance.
- The magnitude correlation coefficient (rsq_MAG) is obtained by linear regression from the plot of the fitted vs. measured magnitude of the impedance.

The plots themselves are presented as individual figures. For a perfect fit, the three plots are straight lines where y(x)=x.

Additionally, the fitting results are compared with the original data files by using the Pearson's Chi-square Test of Goodness-of-Fit. This information includes the following parameters: R, chi2, p, fit. These parameters are further explained in Chapter 5.


corrgui
Correlation plot (X,Y) for Real Part
Fitted (real)
Measured (real)
Correlation plot (X,Y) for Imaginary Part
Fitted (imag)
Measured (imag)
Correlation plot (X,Y) for Magnitude
Fitted (mag)
Measured (mag)

	real_R^2	real_R^2(adj)	imag_R^2	imag_R^2(a...	MAG_R^2	MAG_R^2(a...	R1	chi2	p	fit
1	0.9818	0.9815	0.8180	0.8149	0.9832	0.9829	0.9915	901.9110	0	2.3073e+04
2	0.9602	0.9596	0.7968	0.7934	0.9713	0.9708	0.9856	3.3847e+03	0	7.7680e+04
3	0.9735	0.9731	0.7158	0.7110	0.9724	0.9719	0.9861	1.4093e+03	0	5.7472e+04
4	0.9632	0.9625	0.6534	0.6475	0.9615	0.9608	0.9806	1.7267e+03	0	6.4725e+04
5	0.9704	0.9699	0.6956	0.6905	0.9688	0.9683	0.9843	1.7465e+03	0	7.6775e+04
6	0.9823	0.9820	0.8107	0.8075	0.9825	0.9822	0.9912	1.1879e+03	0	4.6590e+04
7	0.9662	0.9657	0.6755	0.6700	0.9649	0.9643	0.9823	1.7062e+03	0	6.5565e+04
8	0.9637	0.9631	0.6486	0.6426	0.9623	0.9617	0.9810	2.2719e+03	0	1.0658e+05
9	0.9773	0.9770	0.7617	0.7576	0.9762	0.9758	0.9880	1.5284e+03	0	6.5088e+04

Chapter 4

Equivalent Circuits

Experimental data is fitted to equivalent circuit models. The models are designed to describe the interfaces, chemical processes and boundaries of the measured setup.

4.0.1 Circuit Elements

The elements present in the software are described in Table 4.1:

Table 4.1: Equivalent circuit elements and their MATLAB implementation.

Symbol	Element	Equation	MATLAB expression
R1	Resistor	$Z(f) = R$	`z=p*ones(size(f))`
C1	Capacitor	$Z(f) = 1/j2\pi fC$	`z=1./(1i*2*pi*f*p)`
L1	Inductor	$Z(f) = j2\pi fL$	`z=1i*2*pi*f*p`
E2	Constant Phase Element	$Z(f) = 1/p_1(j2\pi f)^{p_2}$	`z=1./(p(1)*(1i*2*pi*f).^p(2))`

The Warburg element can be obtained with a Constant-Phase Element by setting $p_2 = 0.5$

4.0.2 Circuit String Syntax

Circuits can be built using series and parallel combinations of the elements in Table 4.1, using the series and parallel operators s() and p(). These operators can contain any number of elements, separated by commas.

The number next to the element letter is the number of free parameters for this element. For a capacitor (C1) the only free parameter is the capacitance. For the constant-phase element (E2) the free parameters are p1 and p2.

Common mistake: Do not write the circuit elements as s(R1,R2,R3,C4...). The elements cannot be written as labels. Instead, write s(R1,R1,R1,C1...).

4.0.3 Circuit Files (.ckt)

The circuit string, initial parameters and boundary conditions can be stored and loaded from a circuit file with the .ckt extension. This file is read by MATLAB line-to-line. The file should include only four lines with the following content:

Line 1: circuit string

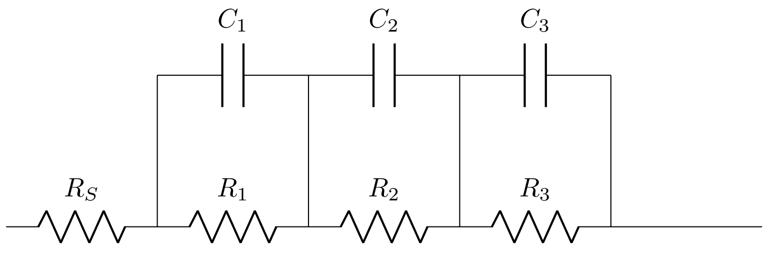
Line 2: initial parameters, sorted in the same order as they appear in the circuit string

Line 3: lower boundary conditions (LB)

Line 4: upper boundary conditions (UB)

Check the folder 'examples_circuits' for more examples.

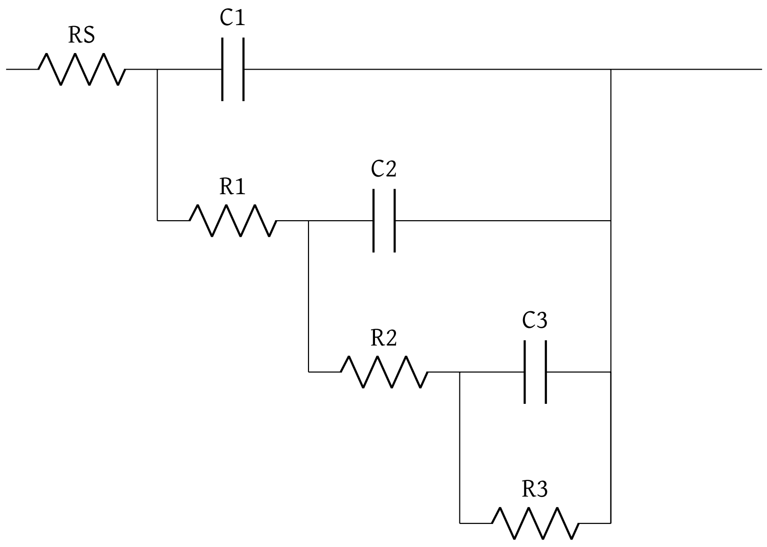
Example 1: Voigt model in the form R+R//C+R//C+R//C



Recommended circuit file
s(R1,p(R1,C1),p(R1,C1),p(R1,C1))
[100,100,1e-6,100,1e-6,100,1e-6]
[0,0,0,0,0,0,0]
[inf,inf,inf,inf,inf,inf,inf]

Notes
The elements appear in the circuit string in the following order: RS,R1,C1,R2,C2,R3,C3
The initial parameters, LB and UB have the exact same order

Example 2: Ladder circuit in the form: ((((R//C)+R)//C)+R)//C+R



Recommended circuit file
s(p(s(p(s(p(R1,C1),R1),C1),R1),C1),R1)
[100,1e-6,100,1e-6,100,1e-6,100]
[0,0,0,0,0,0,0]
[inf,inf,inf,inf,inf,inf,inf]

Notes
The elements appear in the circuit string in the following order: R3,C3,R2,C2,R1,C1,RS
The initial parameters, LB and UB have the exact same order

Chapter 5

Algorithms

This chapter includes the mathematical description of all the algorithms implemented in the program.

5.1 Weighting types

The fitting algorithms of this toolbox reduce the following distance function (Andrzej Lasia 2014).

$$dist = S = \sum_{i=1}^{n} \left(w_i'[Z_i' - Z_{i,calc}']^2 + w_i''[Z_i'' - Z_{i,calc}'']^2 \right) \tag{5.1}$$

The parameters w_i' and w_i'' are the weighting values, computed by one of the following methods:

Unit weighting	$w_i' = w_i'' = 1$
Modulus weighting	$w_i' = w_i'' = 1/\|Z\|^2$
Proportional weighting	$w_i' = 1/(Z')^2$ and $w_i'' = 1/(Z'')^2$
Statistical weighting	$w_i' = 1/(\sigma_i')^2$ and $w_i'' = 1/(\sigma_i'')^2$

5.2 Minimization algorithms

5.2.1 fminsearchbnd

To minimize the distance function, the toolbox uses the **fminsearchbnd** algorithm.
This algorithm is called in MATLAB to minimize the distance function of Eq. (5.1).

- Levenberg-Marquardt
- Nelder-Mead
- BFGS
- Powell

5.3 Statistics

The overall quality of the fitting can be determined by comparing the original measured data file (expected values) to the simulated values using the fit results (observed values).

Currently the software implements two methods for this comparison: linear regression and Pearson's chi-square test of goodness-of-fit.

5.3.1 Linear regressions

Real of fitted vs Real of measured

Imag of fitted vs Imag of measured

MAG of fitted vs MAG of measured

5.3.2 Chi-square

The Chi-square parameter is calculated with the following equation (Andrzej Lasia 2014).

$$\chi^2 = \sum_{i=1}^{n} \left(\left[\frac{Z'_i - Z'_{i,calc}}{\sigma'_i} \right]^2 + \left[\frac{Z''_i - Z''_{i,calc}}{\sigma''_i} \right]^2 \right)$$

This parameter depends on the number of points.

Usually it is divided by the number of degrees of freedom $\nu = 2N - m$

$$\chi^2_v = \frac{chi^2}{\nu} = \frac{chi^2}{2N - m}$$

N is the number of frequencies: there are 2N measured impedance points (N real and N imaginary)

m is the number of adjustable parameters in the model

5.3.3 Test F

For comparing two different variances (Lasia 2014)

$$F = \frac{\sigma_1^2}{\sigma_2^2}$$

5.3.4 T-test

For importance of regression parameters (Lasia 2014)

$$t = p/s_p$$

where p is the value of the parameter and s_p its standard deviation

5.4 Error estimates for individual parameters

ToDo: This is not yet implemented. It depends on the optimizer functions.

Chapter 6

Licenses for included software

6.1 Zfit

The original file was released in 2005 and it is available here:

https://de.mathworks.com/matlabcentral/fileexchange/19460-zfit

6.2 fminsearchbnd

This function was written by John D'Errico and published on MathWorks MATLAB Central under an open-source license. The original file can be downloaded at https://de.mathworks.com/matlabcentral/fileexchange/8277-fminsearchbnd--fmi

The function is based on fminsearch and includes the possibility of using boundary conditions, such as the lower and upper limits for the individual circuit parameters.