# ecx Documentation

***Release 0.1.0***

**M. Skocic**

**May 13, 2023**

# CONTENTS:

CHAPTER

ONE

# GETTING STARTED

## 1.1 ecx

*ecx* is a Fortran library providing the formulas for electrochemistry. It also provides a API for the C language. A shared and a static library *libiapws* are compiled (f2008+) with the Fortran and C headers. The static and shared libraries can be installed in order to be included in Fortran or C programs.

The compilation was tested on Linux (Debian), MacOS and Windows.

Sources: https://github.com/MilanSkocic/ecx

### 1.1.1 How to install

### 1.1.2 Dependencies

```
gcc>=4.6 or msvc>=14

gfortran>=4.6 or ifort>=18

cmake>=3.10
```

### 1.1.3 License

```
GNU General Public License v3 (GPLv3)
```

## 1.2 pyecx

Python wrapper around the Fortran ecx library. Follow the installation instructions. for compiling and installing the library.

For now,the wrapper must be compiled on Linux, windows and MacOS platforms after installing the ecx library using the compiler that was used to compile your python interpreter.

### 1.2.1 How to install

```
pip install pyecx
```

### 1.2.2 Dependencies

```

```

### 1.2.3 License

```
GNU General Public License v3 (GPLv3)
```

## 1.3 Examples

### 1.3.1 Example in Fortran

```fortran
program example_in_f
    use iso_fortran_env
    use ecx_eis
    implicit none

    real(real64) :: w(3) = [1.0d0, 1.0d0, 100.0d0]
    real(real64) :: r = 100.0d0

    print *, ecx_eis_zr(w,r)


end program
```

### 1.3.2 Example in C

```c
#include <stdio.h>
#include <stdlib.h>
#include "ecx_eis_capi.h"


int main(void){

    double w = 1.0;
    double r = 100.00;
    size_t n = 1;
    ecx_cdouble z = ecx_cbuild(0.0,0.0);

    ecx_capi_zr(&w, r, n, &z);

    printf("%f I%f \n", creal(z), cimag(z));

    return EXIT_SUCCESS;
}
```

### 1.3.3 Example in Python

```python
import numpy as np
from pyecx import eis
import matplotlib.pyplot as plt

R = 100
C = 1e-6
w = np.logspace(6, -3, 100)

zr = np.asarray(eis.zr(w, R))
zc = np.asarray(eis.zc(w, C))
zrc = zr*zc / (zr+zc)
print("finish")

fig = plt.figure()
ax = fig.add_subplot(111)

ax.set_aspect("equal")
ax.plot(zrc.real, zrc.imag, "g.", label="R/C")

ax.invert_yaxis()

plt.show()
```

CHAPTER

TWO

# ELECTROCHEMISTRY - THEORETICAL BACKGROUND

## 2.1 EIS

### 2.1.1 Introduction

Frequency dispersion measurements (or impedance spectroscopy) have become a common technique for the study of mass and charge transport in electrochemical systems. With the availability of automated high quality frequency response analysis systems immittance (i.e. impedance or admittance) measurements can be obtained in fairly easy way [1].

The advantage of measurements taken in the frequency domain over measurements in the time domain (i.e. pulse or step response measurements) is that the frequency response earl be described analytically, using an equivalent circuit as model. Time domain analysis often requires the approximation of complex functions, e.g. infinite summations of exponential functions. The circuit elements represent the various (macroscopic) processes involved in the transport of charge and mass. The dispersion relations for most equivalent circuit elements are very simple Barsoukov and Macdonald [1], Orazem and Tribollet [2].

If the (complex) immittance diagrams show distinct features, which can easily be related to specific subcircuits of the equivalent circuit model, analysis become quite simple. Often this can be accomplished by graphical means, using a compass and a ruler. However, if the time constants of the respective subcircuits are close together, or if elements with a fractional (e.g. Warburg, or a CPE-type element, a more sophisticated analysis procedure is needed. As the variation of one circuit parameter can influence large parts of the frequency dispersion, all parameters must be adjusted simultaneously in order to obtain the optimum fit to the data [3, 4, 5]

### 2.1.2 Black Box Approach

- Assume a black box with two terminals.
- One applies a voltage and measures the current response (or visa versa) as show in Fig. **??**
- Signal can be dc or periodic with frequency math:*f*, or angular frequency $\omega = 2\pi f$ as shown in Fig. **??**

**with:** $0 \leq \omega < \infty$**:**

- Voltage: $V(\omega) = V_0 \cdot e^{j\omega t}$
- Current: $I(\omega) = I_0 \cdot e^{j(\omega t-\phi)}$

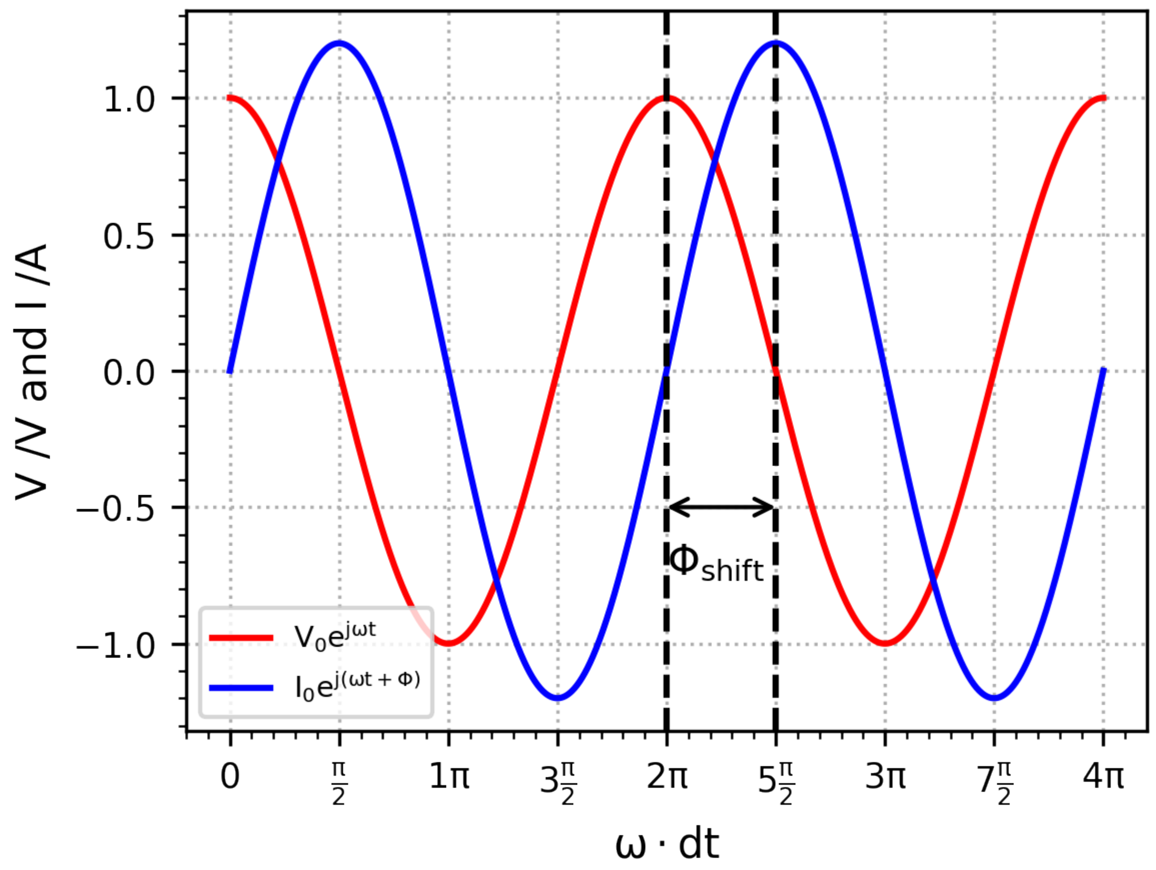
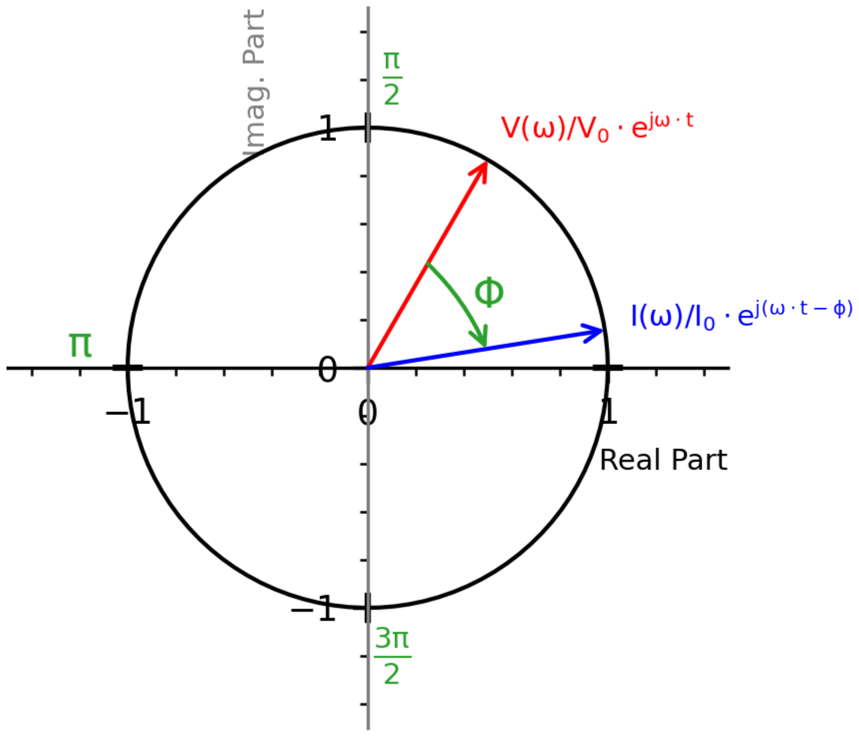


Fig. 2.1.1: EIS AC Waves



Imag. Part
$\frac{\pi}{2}$
$V(\omega)/V_0 \cdot e^{j\omega \cdot t}$
$\Phi$
$I(\omega)/I_0 \cdot e^{j(\omega \cdot t - \phi)}$
$\pi$
1
0
−1
0
1
Real Part
−1
$\frac{3\pi}{2}$

Fig. 2.1.2: Trigonometric Circle

### 2.1.3 What is EIS?

The impedance is determined from the imposed voltage/current and the measured current/voltage through the Ohm's law:

$$Z(\omega) = \frac{V(\omega)}{I(\omega)} = \frac{V_0}{I_0} e^{j\phi} = Z_0 e^{j\phi}$$

Therefore: * Resistive behavior: $ReZ = Z_0 \cdot \cos\phi$ * Capacitive/Inductive behavior $ImZ = Z_0 \cdot \sin\phi$

Sometimes, the complex admittance can also be used which is defined as the inverse of the complex impedance

$$Y(\omega) = \frac{1}{Z(\omega)}$$

### 2.1.4 Representation

The impedance $Z(\omega)$ can be represented in two different ways as shown in:

1. **Bode plot**: shows the phase shift and magnitude changes in the applied frequency ranges as shown in Fig. **??** and Fig. **??**
2. **Nyquist plot**: represents the real and imaginary parts of $Z(\omega)$ using cartesian coordinates as shown in Fig. **??**

The Bode plot has great advantages for observing phase margins in which the system becomes unstable (violent phase or magnitude changes). Therefore, it is useful for the study of sensors, filters, and transistors in electronic devices.

The Nyquist plot provides insight into the possible mechanism or governing phenomena in an equivalent circuit model system. Among these two types of representations, the Nyquist plot is more often used to analyze the characteristics of electrochemical processes.

### 2.1.5 Series and Parallel Connections

- Series connection: $Z_1 - Z_2 - \ldots - Z_n$ *= Z_{eq} = sum Z_i*
- Parallel connection: $Z_1/Z_2/\ldots/Z_n$ *= Z_{eq} = left( sum frac{1}{Z_i} right)^{-1}*

### 2.1.6 Equivalent Circuit Models

- The circuit model for EIS consists of a combination of electrical circuit elements:
  - ideal elements: resistors (R), capacitors (C) and inductors (L)
  - nonideal capacitor-like element: Constant Phase Element (CPE or $Q$)
  - diffusion elements: semi-infinite Warburg ($W$), Finite Length Warburg ($W_\delta$ or $O$) and Finite Space Warburg ($W_m$ or $T$)
- The circuit model represents the entire system of the electrochemical cell and therefore the aim is to construct an optimal circuit model that is physically meaningful and minimizes the number of variables.

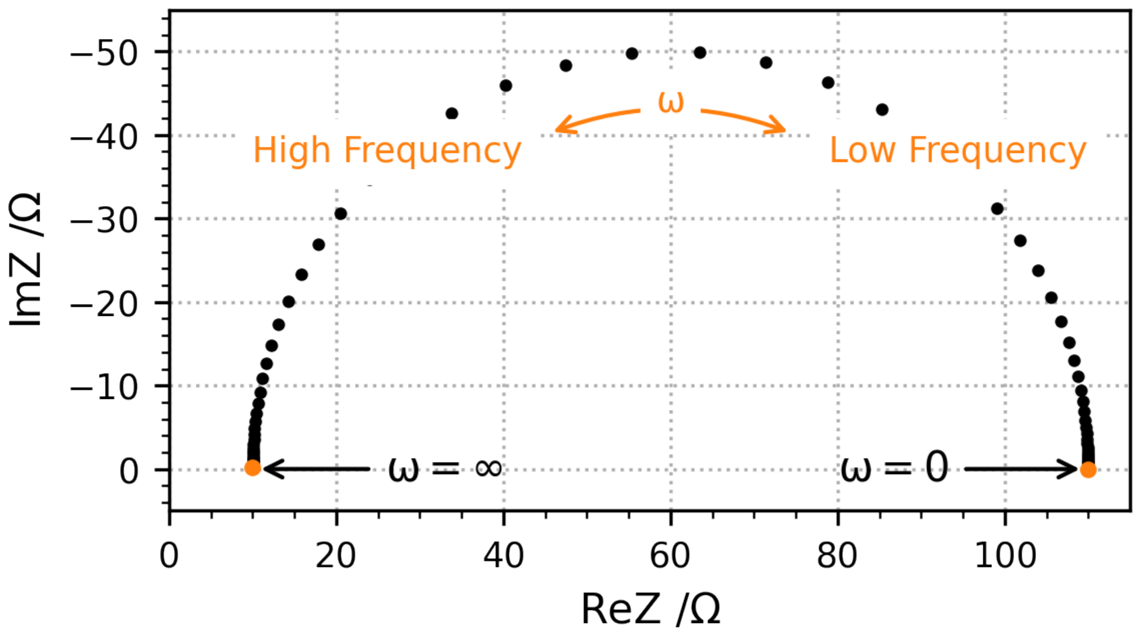


Fig. 2.1.3: Nyquist Representation

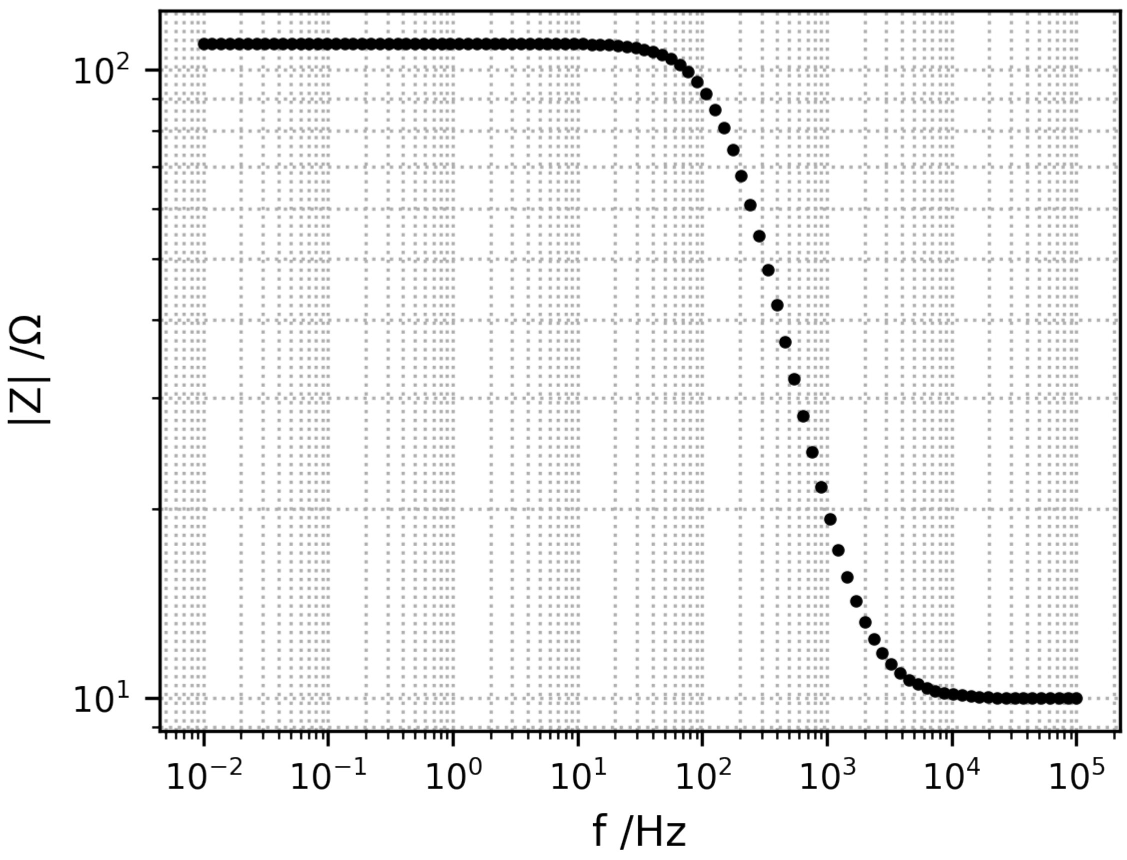


Fig. 2.1.4: Bode Modulus Representation

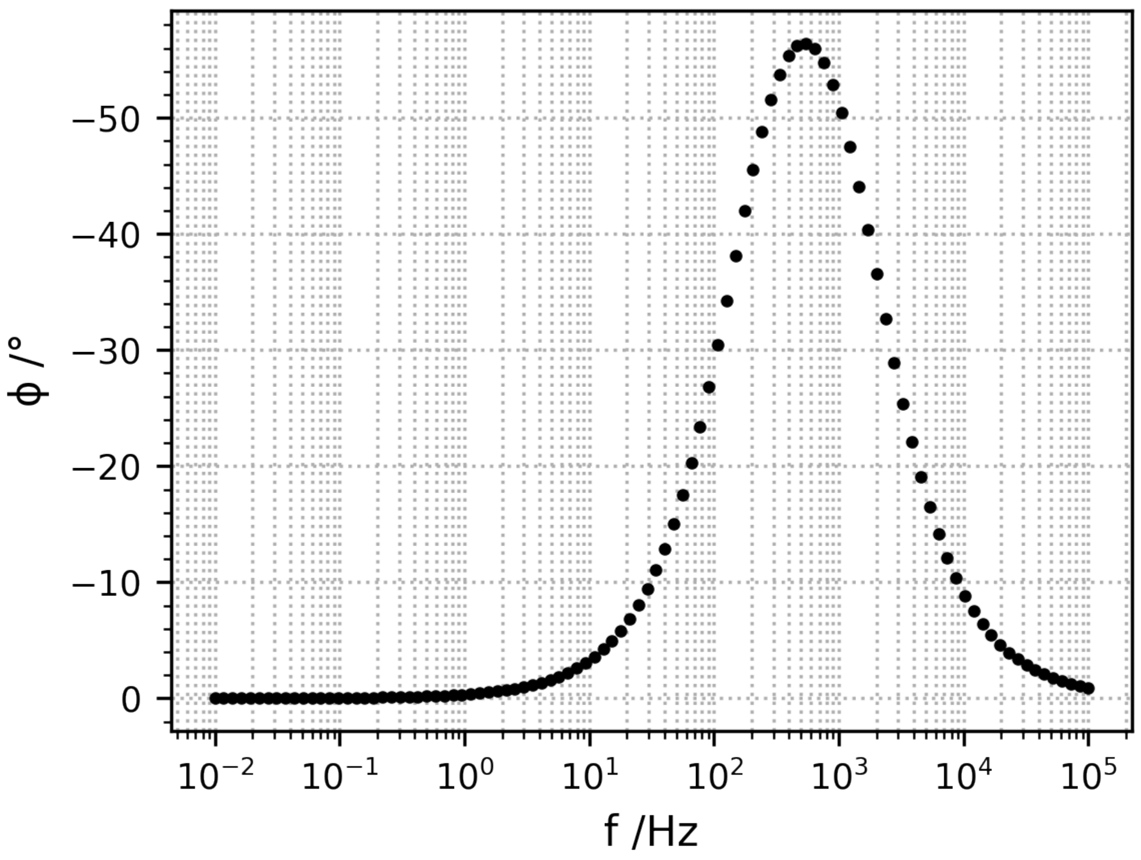


Fig. 2.1.5: Bode Phase Representation

### 2.1.7 Circuit elements

The differents circuit elements available with their string representation are listed here and their Nyquist representation is shown in Fig. **??**. In order to be recognized by the string parser each element must start the one or two letters defined below and can be followed by a name. The measurement model element needs an additional parameter which the number of Voigt elements defined after an underscore.

- R[name]: $Z(\omega) = R$
- C[name]: $Z(\omega) = \frac{1}{jC\omega}$
- L[name]: $Z(\omega) = jL\omega$
- W[name]: $Z(\omega) = \frac{\sigma}{\sqrt{\omega}} \cdot (1 - j)$
- Wd[name]: $Z(\omega) = \frac{R_\delta \cdot \tanh\left(\sqrt{j\omega\tau}\right)}{\sqrt{j\omega\tau}}$
- Wm[name]: $Z(\omega) = \frac{R_m \cdot \coth\left(\sqrt{j\omega\tau}\right)}{\sqrt{j\omega\tau}}$
- Q[name]: $\frac{1}{Q(jw)^\alpha}$
- M[name]_[n]: $Z(\omega) = R_0 + \sum_{k=0}^{k=n} \frac{R_k}{1+jR_kC_k\omega}$
- G[name]: $Z_G(\omega) = G \cdot (K_g + i\omega)^{-n_g}$

Inductor and Finite Space Warburg are rarely encountered in corrosion studies.

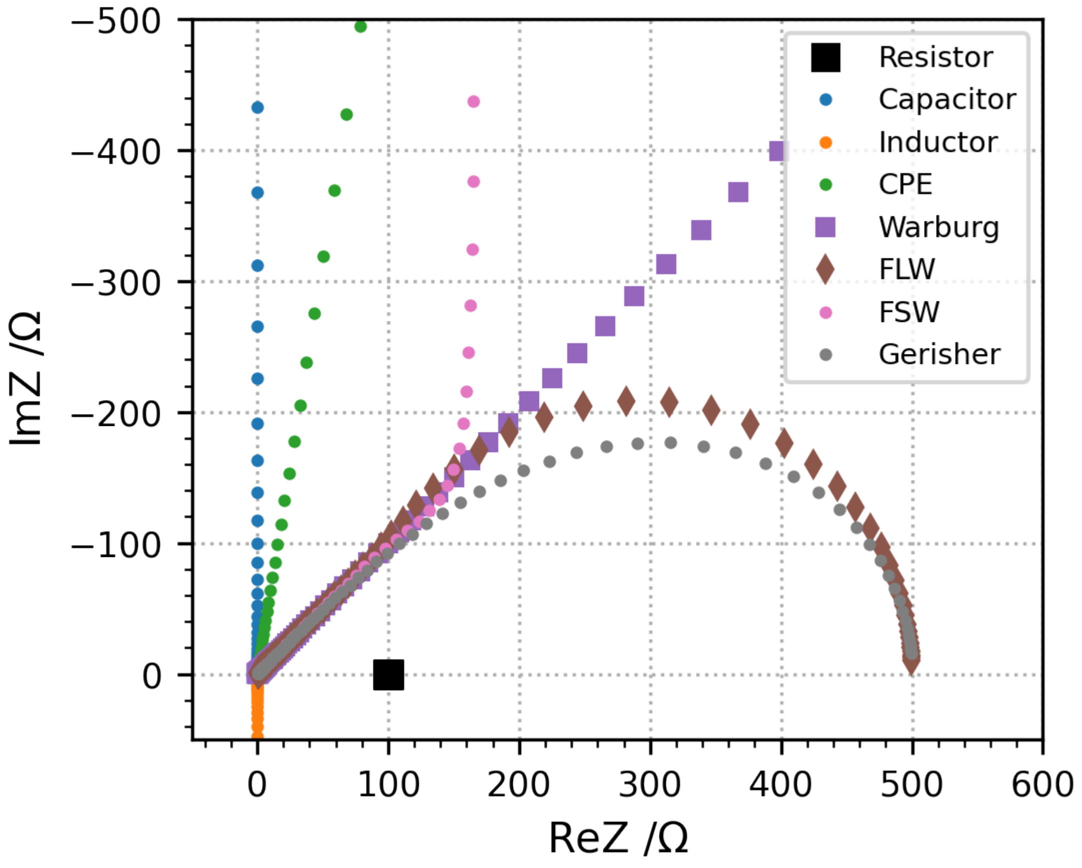


Fig. 2.1.6: Circuit Elements

### 2.1.8 Link between circuit elements and physical parameters

- Resistors can be linked to resistivity or kinetics:
  - $R = \frac{\rho \cdot d}{A}$
  - $R = \frac{RT}{FAj_0(\alpha_a+\alpha_c)} = \frac{RT}{AF^2k^0K_c(\alpha_a+\alpha_c)}$
- Capacitors can be linked to layer thickness
  - $C = \frac{\epsilon\epsilon_0 A}{d}$
- FS/FL Warburg element can be linked to diffusion coefficient and layer thickness:
  - $R = \frac{RT}{AF^2\sqrt{2}} \cdot \frac{d}{D \cdot C^*}$
  - $\tau = \frac{d^2}{D}$
  - $\sigma = \frac{R}{\sqrt{2\tau}}$

where,

- $R$: resistance $[\Omega]$
- $\rho$: resistivity $[\Omega \cdot m]$
- $d$: thickness $[m]$
- $A$: Area $[m^2]$
- $j_0$: exchange current density $[A \cdot m^{-2}]$
- $k^0$: kinetics constant $[m \cdot s^{-1}]$
- $K_c$: concentration factor $[mol \cdot m^{-3}]$
- $\alpha_a$: anodic transfer coefficient
- $\alpha_c$: cathodic transfer coefficient
- $C^*$: bulk concentration of the diffusing species $[mol.m^{-3}]$

### 2.1.9 Simplified Randles Circuit

Reflects electrochemical reaction controlled only by kinetics as shown in Fig. **??**

- $R_{el} + R_{ct}/C_{dl}$
- $R_{el}$: electrolyte resistance
- $R_{ct}$: charge transfer resistance
- $C_{dl}$: double layer capacitance

### 2.1.10 Randles Circuit

Reflects electrochemical reaction controlled by kinetics and diffusion as shown in Fig. **??**

- $R_{el} + R_{ct}/C_{dl}$
- $R_{el}$: electrolyte resistance
- $R_{ct}$: charge transfer resistance
- $C_{dl}$: double layer capacitance
- $W$: semi-infinite diffusion

Example for the Nyquist plot of the Randles circuit.

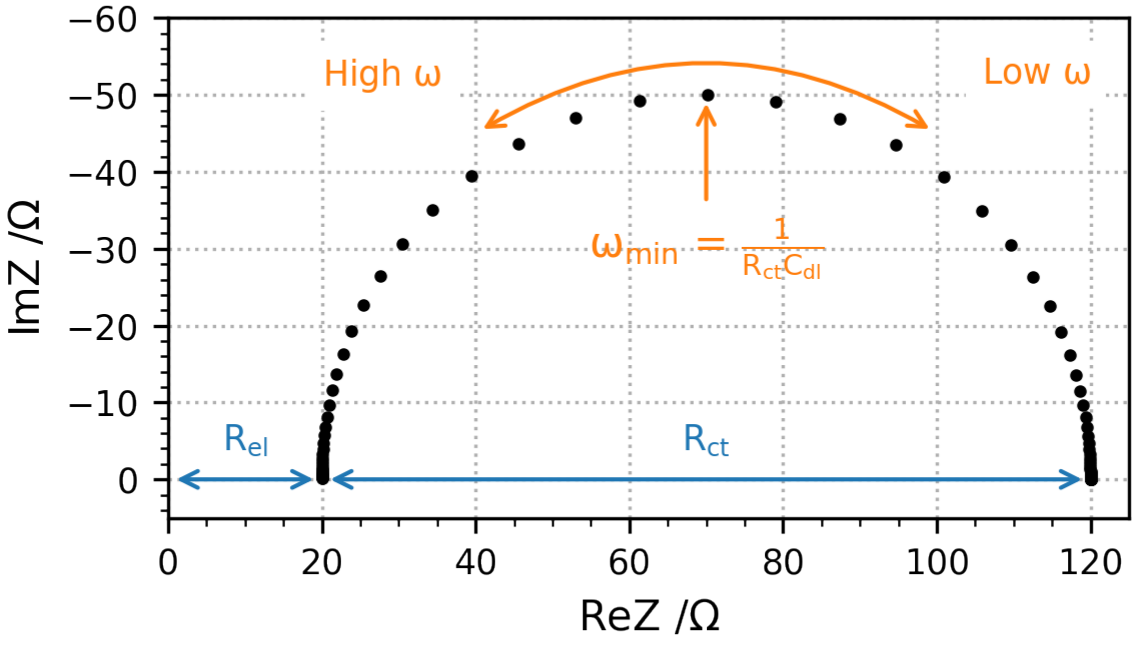


Fig. 2.1.7: Simplified Randles

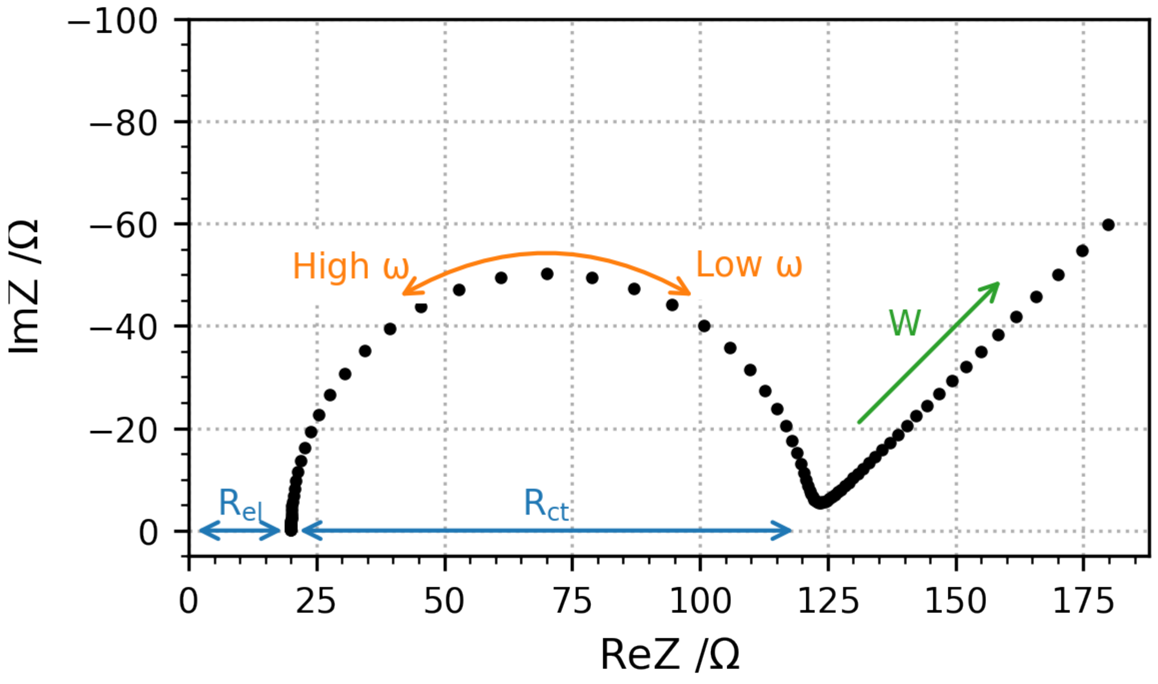


Fig. 2.1.8: Randles

```
Rel = 20.0
R = 100
C = 1e-5
W = 15.0

c = skx.eis.ElectrochemicalCircuit.from_string("Rel+(Rct+W)/Cdl")
c.set_parameter_values({"Rel": Rel, "Rct": R, "Cdl": C, "W": W})

Z = c(xi)
ReZ = Z.real
ImZ = Z.imag
modZ = np.absolute(Z)
phase = np.angle(Z, deg=True)

fig = plt.figure()
ax = fig.add_subplot(111)
ax.set_aspect("equal")
ax.set_xlabel("ReZ /$\Omega$")
ax.set_ylabel("ImZ /$\Omega$")


ax.plot(ReZ, ImZ, "k.", ms=4)




ax.text(s=r"High $\omega$", x=0+Rel, y=-55, va="top", ha="left", backgroundcolor="w",␣
↪color="C1", fontsize="small")
ax.text(s=r"Low $\omega$", x=100+Rel, y=-55, va="top", ha="right", backgroundcolor="w
↪", color="C1", fontsize="small")
ax.annotate(text="", xy=(20+Rel, -45), xytext=(80+Rel, -45),
            arrowprops=dict(arrowstyle="<->", connectionstyle="arc3, rad=0.3", color=
↪"C1"))

ax.annotate(text="", xy=(0, 0), xytext=(Rel, 0), arrowprops=dict(arrowstyle="<->",␣
↪color="C0"))
ax.text(s="$R_{el}$", x=Rel/2, y=-5, va="center", ha="center", color="C0", fontsize=
↪"small")

ax.annotate(text="", xy=(Rel, 0), xytext=(Rel+R, 0), arrowprops=dict(arrowstyle="<->",
↪ color="C0"))
ax.text(s="$R_{ct}$", x=Rel+R/2, y=-5, va="center", ha="center", color="C0", fontsize=
↪"small")


ax.annotate(text=r"", xy=(Rel+R+40, -50), xytext=(Rel+R+10, -20), color="C2",
            arrowprops=dict(arrowstyle="->", color="C2"))
ax.text(s="$W$", x=Rel+R+20, y=-40, va="center", ha="center", color="C2", fontsize=
↪"small")



ax.set_xlim(0,)
ax.set_ylim(-100, 5)

ax.invert_yaxis()
```

### 2.1.11 Differential Impedance analysis

The differential Impedance Analysis (DIA) is based on the use of a Local Operator Model (LOM) which is a equivalent circuit for a simple Faradic reaction but has direct meaning with the experimental spectrum that is being analyzed.

The LOM operator corresponds to the equivalent circuit Rads-(R/C).

The procedure of the structural and parametric identification can be described by the following steps:

- scanning with the LOM throughout the whole frequency range with a scanning window of a single frequency
- parametric identification of the LOM parameters at every working frequency
- Frequency analysis of the LOM parameters' estimates

#### Scanning with the LOM operator

The impedance of the LOM operator is defined as shown in Eq.**??**:

$$Z_{LOM} = R_{ads} + \frac{R}{1+T^2\omega^2} - j\frac{\omega RT}{1+T^2\omega^2} \tag{2.1.1}$$

#### Parametric idenfication of the LOM parameters

The objective is to identify the LOM parameters $P_j = Rads, R, C, T$.

First the effective resistance and the effective inductance are expressed:

$$R_{eff} = ReZ = R_{ads} + \frac{R}{1+T^2\omega^2}$$

$$L_{eff} = -ImZ/\omega = \frac{RT}{1+T^2\omega^2}$$

Derivatives of the effective resistance and inductance are:

$$\frac{dR_{eff}}{d\omega} = -R\frac{2\omega T^2}{(1+T^2\omega^2)^2}$$

$$ImZ = -L_{eff}\cdot\omega$$

$$\frac{dImZ}{d\omega} = -\frac{ImZ}{dL_{eff}}\frac{dL_{eff}}{d\omega} = -\omega\frac{dL_{eff}}{d\omega}$$

$$\frac{dL_{eff}}{d\omega} = -RT\frac{2T^2\omega}{(1+T^2\omega^2)^2} = -\frac{dImZ}{d\omega}\frac{1}{\omega}$$

Expression of the LOM parameters $P_i$:

$$T(\omega) = \frac{\frac{dL_{eff}}{d\omega}}{\frac{dR_{eff}}{d\omega}} = \frac{dL_{eff}}{dR_{eff}}$$

$$R(\omega) = -\frac{dR_{eff}}{d\omega}\cdot\frac{(1+T^2\omega^2)^2}{2\omega T^2}$$

$$R_{ads}(\omega) = R_{eff}(\omega) - \frac{R}{1+T^2\omega^2}$$

$$C(\omega) = \frac{T}{R}$$

### Temporal analysis

The temporal analysis computes the logarithmic values of the LOM parameters $L_j = a, r, c, t$ with respect to $\nu$ as defined in Eq.**??**:

$$\begin{aligned} L_j = \log_{10} P_j = \log_{10} Rads, \log_{10} R, \log_{10} C, \log_{10} T \\ L_j = a, r, c, t \\ \nu = \log_{10} \frac{1}{\omega} \end{aligned} \tag{2.1.2}$$

### Differential temporal analysis

The differential temporal analysis computes the derivatives $d_j$ of $L_j$ with respect to $\nu$ as defined in Eq.**??**

$$d_j = \frac{dL_j}{d\nu} = da, dr, dc, dt \tag{2.1.3}$$

### Spectral analysis

The spectral analysis is obtained by accumulating frequency bands with approximatively equal values of the parameters $L_j$. The amplitude of the individual spectral line $S_{j,l}$ can be expressed as shown in Eq.**??**.

$$\begin{aligned} S_{j,l} = \sum_1^N B(L_{j,i}) \\ B(L_{j,i}) = w_0 \text{ if } l < L_{j,l} < l + s \\ B(L_{j,i}) = 0 \text{ otherwise} \\ w_0 = N_{frequencies}/N_{decades} \end{aligned} \tag{2.1.4}$$

The spectral line is expressed in dB.

### Differential spectral analysis

The differential spectral analysis is obtained by accumulating frequency bands with approximatively equal values of the parameters $d_j$. The amplitude of the individual spectral line $S_{j,l}$ can be expressed as shown in Eq.**??**.

$$\begin{aligned} S_{j,l} = \sum_1^N B(d_{j,i}) \\ B(d_{j,i}) = w_0 \text{ if } l < d_{j,l} < l + s \\ B(d_{j,i}) = 0 \text{ otherwise} \\ w_0 = N_{frequencies}/N_{decades} \end{aligned} \tag{2.1.5}$$

The spectral line is expressed in dB.

An example with a simple RC circuit:

```
f = np.logspace(5, -2, 100)
w = 2*np.pi*f
xi = (w,)

c = skx.eis.ElectrochemicalCircuit.from_string("Rel+R/C")

values = {"Rel":10.0, "R":100, "C": 1e-5}
c.set_parameter_values(values)

```

(continues on next page)

(continued from previous page)

```
Z = c(xi)

ReZ = Z.real
ImZ = Z.imag
modZ = np.absolute(Z)
phase = np.angle(Z, deg=True)

fig = plt.figure()
ax = fig.add_subplot(111)
ax.set_aspect("equal")
ax.set_xlabel("ReZ /$\Omega$")
ax.set_ylabel("ImZ /$\Omega$")


ax.plot(ReZ, ImZ, "k.", ms=4)


ax.invert_yaxis()
```

```
dia = skx.eis.DifferentialImpedanceAnalysis(f, ReZ, ImZ)

plt.figure()
plt.xlabel(r"$\nu$")
plt.ylabel("$L_j$")
for key, values in dia.temporal_analysis().items():
    x = values["v"]
    y = values["L"]
    plt.plot(x, y, label = key)
plt.legend()

plt.figure()
plt.xlabel(r"$\nu$")
plt.ylabel("$d_j$")
for key, values in dia.differential_temporal_analysis().items():
    x = values["v"]
    y = values["d"]
    plt.plot(x, y, label = key)
plt.legend()

plt.figure()
plt.xlabel(r"$L_j$")
plt.ylabel("I /dB")
for key, values in dia.spectral_analysis(nbins=20).items():
    x = values["bins"]
    y = values["I"]
    plt.bar(x, y, width=dia.sa_width, label = key)
plt.legend()
name = 'EIS-DIA-sa.png'
plt.savefig('./figures/' +  name , dpi=DPI, format="png")

plt.figure()
plt.xlabel(r"$d_j$")
plt.ylabel("I /dB")
for key, values in dia.differential_spectral_analysis(nbins=20).items():
    x = values["bins"]
    y = values["I"]
```

(continues on next page)

media/EIS-DIA-RC.png

Fig. 2.1.9: Simple RC

(continued from previous page)

```
    plt.bar(x, y, width=dia.dsa_width, label = key)
plt.legend()
```

media/EIS-DIA-ta.png

Fig. 2.1.10: Temporal Analysis

media/EIS-DIA-dta.png

Fig. 2.1.11: Differential Temporal Analysis

media/EIS-DIA-sa.png

Fig. 2.1.12: Spectral Analysis

media/EIS-DIA-dsa.png

Fig. 2.1.13: Differential Spectral Analysis

### 2.1.12 References

Barsoukov and Macdonald [1], Orazem and Tribollet [2], Boukamp [3], Bevington and Robinson [4], Press *et al.* [5], Stoynov and Vladikova [6]

## 2.2 PEC

The `skelectrox.pec.core` module aims at simplifying the generation and fitting of data from photoelectrochemical measurements.

### 2.2.1 Introduction

Photo-electrochemistry characterizations are used to study films at macroscopic, mesoscopic, and microscopic scales. The latter advances were used to support (photo-)electrochemical studies of the electronic and optical properties of passive films and oxidized metals, and of their interfaces with electrolytes, providing informations on the nature and structure of these materials and to use properties such as the oxidation behaviour of a metallic substrate.

Basically, two kinds of curves are recorded in the course of photoelectrochemical characterization experiments, photocurrent voltammograms and photocurrent energy spectra. In photocurrent voltammograms, photocurrents are measured as a function of the potential, $V$, applied to the semiconducting electrode, at a given photon energy, $E = h\nu$. In photocurrent energy spectra, photocurrents are recorded, at a given applied potential, V, as a function of the photon energy, $E$. The analysis of the shapes of photocurrent voltammograms may allow to obtain informations such as the semiconducting type of the material, the energy of the surface band levels, the presence of macroscopic defects inducing photogenerated electron–hole pairs recombinations.

However, despite attempts to refine the Gartner-Butler model by taking into account surface or volume recombination, a complete description of the photocurrent voltammograms remains difficult, for the latter developments make use of a high number of adjustable parameters, most of them being very difficult to assess. The analysis of the photocurrent energy spectra is intended to identify the chemical nature of the material constituting the semiconducting electrode, through the value of their bandgap energies, $E_g$ as, on the one hand, bandgap energy values have been reported in the literature for numerous compounds, and as, on the other hand, bandgap values may be estimated from thermodynamic extensive atomic data. Practically, photocurrent energy spectra are usually analyzed by means of linear transforms to take benefit of the fact that, using the simplified form of the Gartner–Butler model, the quantum yield, $\eta$, of the photocurrent is proportional to the light absorption coefficient as shown in Eq.**??**.

In such conditions, $\eta$, obeys to the following relationship:

$$(\eta * E)^{1/n} = K(E - E_g) \tag{2.2.1}$$

where $K$ is a constant (things other than $E$ being equal), $E_g$ is the bandgap energy of the semiconductor, and $n$ depends on the band to band transition type, $n = 1/2$ for an allowed direct transition, and $n = 2$ for an allowed indirect transition. Direct transitions are rarely observed in more or less disordered thin oxide films.

### 2.2.2 Fitting of the Photocurrent Energy Spectra

Linear transformations were successfully performed for oxides made of one or two constituents. However, for complex oxide scales formed of several p-type and n-type phases, the complete description of the photocurrent energy spectra could not be achieved, and only semi-quantitative and/or partial informations could be obtained on the oxides present in the scales.

As $I_{PH}^*$ is measured under modulated light conditions and thus actually is a complex number, the real and the imaginary parts of the photocurrent should be considered simultaneously when analyzing and fitting the photocurrent energy spectra, rather than their modulus as shown in Eq.**??**.

$$\begin{aligned} I^*_{PH} &= |I^*_{PH}|\cos\theta + j|I^*_{PH}|\sin\theta \\ I^*_{PH} &= \sum_{i}^{i=N} J_{PH,i}\cos\theta_i + j\sum_{i}^{i=N} J_{PH,i}\sin\theta_i \end{aligned} \tag{2.2.2}$$

where $J_{PH,i}$ and $\theta_i$ represent the modulus and phase shift, respectively, of the photocurrent issued from the ith semiconducting constituent of the oxide layer. For thin semiconducting films, the space charge regions are low compared to penetration depth of the light. $J_{PH,i}$ may thus be expected, at a given applied potential, to follow the simplified form of the Gartner–Butler model.

where $E_{g,i}$ and $K_i$ represent the energy gap and a value proportional to $K$ ($I^*_{PH}$ * is proportional to but not equal to $\eta$) for the ith semiconducting constituent.

An example of the modulus and the phase computed with 4 semiconducting phases:

| Names | Values |
|---|---|
| K1 | +4.00e-05 |
| O1 | -5.00e+01 |
| Eg1 | +1.70e+00 |
| n1 | +2.00e+00 |
| K2 | +6.00e-05 |
| O2 | +1.30e+02 |
| Eg2 | +2.40e+00 |
| n2 | +2.00e+00 |
| K3 | +5.00e-05 |
| O3 | +1.40e+02 |
| Eg3 | +2.80e+00 |
| n3 | +2.00e+00 |
| K4 | +9.00e-05 |
| O4 | -5.00e+01 |
| Eg4 | +3.50e+00 |
| n4 | +2.00e+00 |

### 2.2.3 References

Petit *et al.* [7], Morrison [8], Memming [9], Stimming [10], Butler and Ginley [11], Butler [12] examples/figures/PEC-4SC_Mod.png

Fig. 2.2.1: PEC Modulus

`examples/figures/PEC-4SC_Phase.png`

Fig. 2.2.2: PEC Phase

# CHAPTER THREE

# RELEASE NOTES

## 3.1 ecx 0.1.0 Release Note

### 3.1.1 Changes

- Implementation of eis + C API
- Python wrappers for eis.
- Documentation with sphinx.

### 3.1.2 Download

ecx

pyecx

### 3.1.3 Contributors

Milan Skocic

### 3.1.4 Commits

Full Changelog: https://github.com/MilanSkocic/pyecx/compare/. . . .0.1.0

CHAPTER

# FOUR

# AUTOGENERATED DOCUMENTATION

## 4.1 ecx

### 4.1.1 Fortran

#### EIS

EIS Module.

namespace **ecx_eis**

Module containing functions and subroutines for Electrochemical Impedance Spectroscopy.

##### Functions

`pure elemental complex(real64) function, public ecx_eis_zr (w, r)`

Compute the complex impedance for a resistor.

**Parameters**

- `w` – **[in]** Angular frequencies in rad.s^-1.
- `R` – **[in]** Resistance in Ohms.

**Returns**

Z Complex impedance in Ohms.

`pure elemental complex(real64) function, public ecx_eis_zc (w, c)`

Compute the complex impedance for a capacitor.

**Parameters**

- `w` – **[in]** Angular frequencies in rad.s^-1.
- `C` – **[in]** Capacitance in Farad.

**Returns**

Z Complex impedance in Ohms.

`pure elemental complex(real64) function ecx_eis_zl (w, l)`

Compute the complex impedance for an inductor.

**Parameters**

- `w` – **[in]** Angular frequencies in rad.s^-1.
- `L` – **[in]** Inductance in Henry.

**Returns**
Z Complex impedance in Ohms.

`pure elemental complex(real64) function ecx_eis_zcpe (w, q, a)`

Compute the complex impedance for a CPE.

**Parameters**

- `w` – **[in]** Angular frequencies in rad.s^-1.
- `Q` – **[in]** Resistance in S.s^-a
- `a` – **[in]** CPE exponent

**Returns**
Z Complex impedance in Ohms.

`pure elemental complex(real64) function ecx_eis_w (w, s)`

Compute the complex impedance for a semi-infinite Warburg.

**Parameters**

- `w` – **[in]** Angular frequencies in rad.s^-1.
- `s` – **[in]** Pseudo-Resistance in Ohms.s^(1/2).

**Returns**
Z Complex impedance in Ohms.

`pure elemental complex(real64) function ecx_eis_flw (w, r, tau)`

Compute the complex impedance for a finite length warburg.

**Parameters**

- `w` – **[in]** Angular frequency in rad.s^-1.
- `r` – **[in]** Resistance in Ohms.
- `tau` – **[in]** Characteristic time in s.

**Returns**
Z Complex impedance in Ohms.

`pure elemental complex(real64) function ecx_eis_fsw (w, r, tau)`

Compute the complex impedance for a finite space warburg.

**Parameters**

- `w` – **[in]** Angular frequency in rad.s^-1.
- `r` – **[in]** Resistance in Ohms.
- `tau` – **[in]** Characteristic time in s.

**Returns**
Z Complex impedance in Ohms.

`pure elemental complex(real64) function ecx_eis_g (w, g, k)`

Compute the complex impedance of the Gerisher element.

**Parameters**

- `G` – Pseudo-Resistance in Ohms.s^(1/2).
- `K` – Offset in rad.s^-1.
- `w` – Angular frequency in rad.s^-1.

**Returns**
Z Complex impedance in Ohms.

### Variables

`real(real64), parameter pi = 4.0d0*datan(1.0d0)`

PI constant.

## PEC

PEC module.

namespace **`ecx_pec`**

Module containing functions and subroutines for PhotoElectrochemistry.

### Functions

`pure elemental real(real64) function ecx_pec_alpha (hv, eg, n)`

Compute the not scaled absorbance coefficient.

**Parameters**

- **`hv`** – **[in]** Light energy in eV.
- **`Eg`** – **[in]** Bandgap in eV.
- **`n`** – **[in]** Exponent for direct (1/2) or indirect transition (2)

**Returns**
alpha energy in eV.

`pure elemental real(real64) function ecx_pec_deg2rad (theta)`

Converts degrees to rad.

**Parameters**
**`theta`** – **[in]** Angle in degrees.

**Returns**
phase Angle in rad.

`pure elemental complex(real64) function ecx_pec_iph (hv, k, eg, theta, n)`

Compute the complex photocurrent.

**Parameters**

- **`hv`** – **[in]** Light energy in eV.
- **`K`** – **[in]** Scaling factor for absorbance in .
- **`Eg`** – **[in]** Bandgap in eV.
- **`theta`** – **[in]** Phase in degrees.
- **`n`** – **[in]** Transition type: n=1/2 for direct transition and n=2 for indirect transition

**Returns**
iph Complex photocurrent.

#### Variables

`real(real64), parameter pi   = 4.0d0*datan(1.0d0)`

PI constant.

`real(real64), parameter c   = speed_of_light_in_vacuum`

Speed of light in m/s.

### 4.1.2 C API

#### Common headers

- *ecx_types.h*

```
/**
 * @file ecx_types.h
 * @brief Type C header for th ecx library.
 * @details Compatibilty layer for handling complex numbers with MSC.
 * It is imported by all headers for each submodule.
 */
#ifndef ECX_TYPES_H
#define ECX_TYPES_H

#include <complex.h>
#if _MSC_VER
typedef _Dcomplex ecx_cdouble;
#define ecx_cbuild(real, imag) (_Cbuild(real, imag))
#else
typedef double _Complex ecx_cdouble;
#define ecx_cbuild(real, imag) (real+I*imag)
#endif

#endif
```

- *ecx.h*

```
/**
 * @file ecx.h
 * @brief Main C header for th ecx library.
 * @details Includes all the headers for each submodule.
 */
#ifndef ECX_H
#define ECX_H

#include "ecx_eis_capi.h"
#include "ecx_pec_capi.h"

#endif
```

### 4.1.3 EIS headers

```
/**
 * @file ecx_eis_capi.h
 * @brief EIS C header for th ecx library.
 * @details Complex impedance
 */
#ifndef ECX_EIS_CAPI_H
#define ECX_EIS_CAPI_H
#include "ecx_types.h"

extern void ecx_capi_zr(double *w, double R, size_t n, ecx_cdouble *Z);
extern void ecx_capi_zc(double *w, double C, size_t n, ecx_cdouble *Z);

#endif
```

namespace **ecx_eis_capi**

#### Functions

**pure subroutine ecx_capi_zr (w, r, n, z)**

Compute the complex impedance for a resistor.

**Parameters**

- **w** – **[in]** Angular frequencies in rad.s^-1 as 1d-array.
- **R** – **[in]** Resistance in Ohms.
- **n** – **[in]** Size of w and Z.
- **Z** – **[out]** Complex impedance in Ohms as 1d-array.

**pure subroutine ecx_capi_zc (w, c, n, z)**

Compute the complex impedance for a capacitor.

**Parameters**

- **w** – **[in]** Angular frequencies in rad.s^-1 as 1d-array.
- **C** – **[in]** Resistance in Ohms.
- **n** – **[in]** Size of w and Z.
- **Z** – **[out]** Complex impedance in Ohms as 1d-array.

namespace **iso_c_binding**

### 4.1.4 PEC headers

```
/**
 * @file ecx_pec_capi.h
 * @brief EIS C header for th ecx library.
 */
#ifndef ECX_PEC_CAPI_H
#define ECX_PEC_CAPI_H
#include "ecx_types.h"


#endif
```

## 4.2 pyipaws

# CHAPTER FIVE

# BIBLIOGRAPHY

# CHAPTER SIX

## INDICES AND TABLES

- genindex
- modindex
- search