PalmSens

PyPalmSens

Impedance Spectroscopy with Python

Webinar (20 min) + Q&A (20 min)

What is PyPalmSens?

Open-source Python library

- manage PalmSens instruments
- automate experiments
- analyze data

```
pip install pypalmsens
```

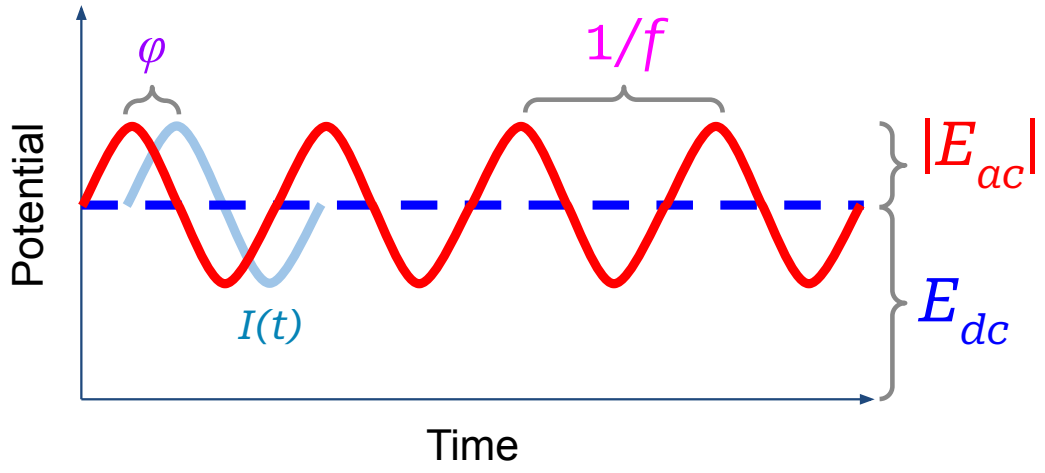
Platform support

Windows, MacOS, Linux (arm/x64)

```python
import pypalmsens as ps

method = ps.ChronoAmperometry(
    interval_time=0.1,
    potential=1.0,
    run_time=5.0,
)
measurement = ps.measure(method)
```

Impedance Spectroscopy



Spectroscopy: vary ***f***

Potentiostatic: apply ***E*** ➡ measure ***I***

Galvanostatic: apply ***I*** ➡ measure ***E***

Applied potential

$$E(t) = \underbrace{E_{dc}}_{\text{bias}} + \underbrace{|E_{ac}|}_{\text{amplitude}} \sin(\underbrace{\omega}_{\substack{\text{angular}\\\text{frequency}}} t)$$

Measured current

$$I(t) = I_{dc} + |I_{ac}| \sin(\omega t + \underbrace{\varphi}_{\text{phase shift}})$$

Impedance

$$Z(\omega) = \frac{E(\omega)}{I(\omega)} = Z' + jZ''$$

$$\omega = 2\pi \underbrace{f}_{\text{frequency}}$$

Why Impedance Spectroscopy?

Field	The question EIS answers	Typical band
Corrosion	how fast, and is the coating still holding?	10 mHz – 1 MHz
Batteries	which resistance grew (SEI)?	1 mHz – 1 MHz
Fuel cells / PV	ohmic, kinetic or mass-transport loss?	10 mHz – 100 kHz
Biomedical	is the tissue/cell barrier intact?	100 Hz – 1 MHz
Food & environment	is it fresh, wet, fouled?	10 Hz – 1 MHz
Materials	bulk, grain boundary, or contact?	1 Hz – 10 MHz

Impedance spectroscopy with PyPalmSens

```python
from pypalmsens import (
    ElectrochemicalImpedanceSpectroscopy    as EIS,
    GalvanostaticImpedanceSpectroscopy      as GEIS,
    FastImpedanceSpectroscopy               as FastEIS,
    FastGalvanostaticImpedanceSpectroscopy as FastGEIS,
)
from pypalmsens import MixedMode  # impedance stage
```

} *fixed frequency, 1 ms latency*

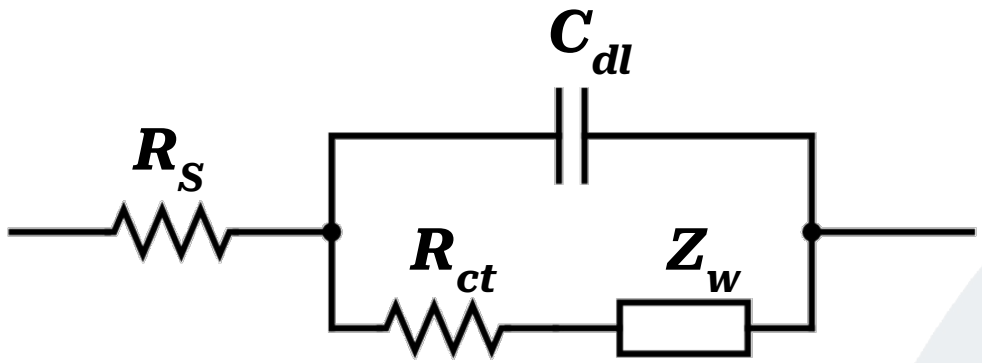
Supported instruments

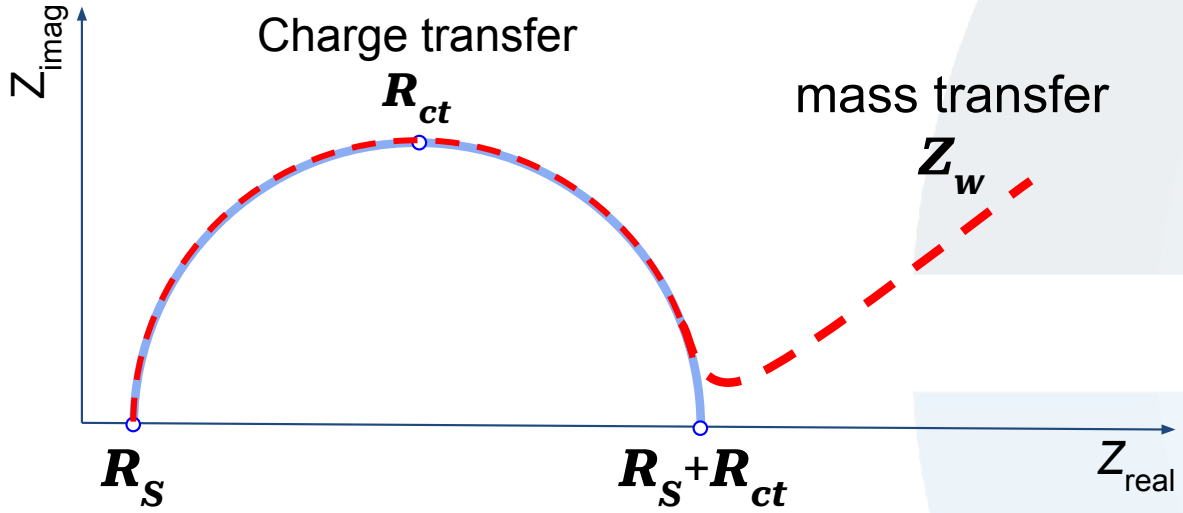
- Nexus
- EmStat 4 series
- PalmSens 4 series
- EmStat Pico
- Sensit Smart / Wearable
- PalmSens 3

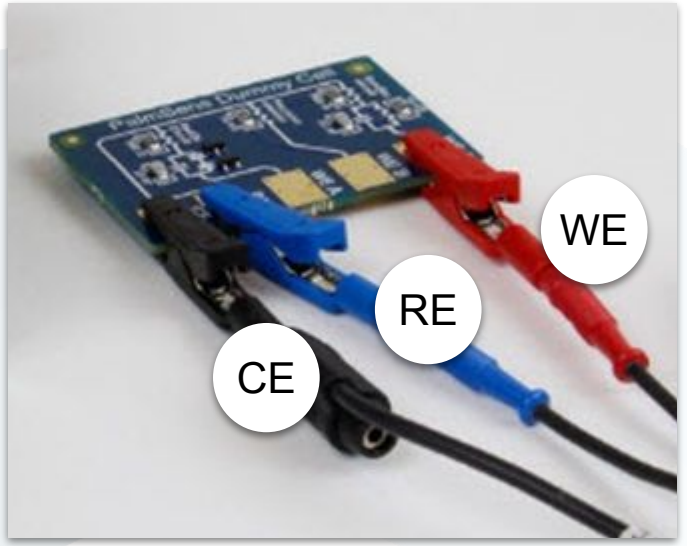


EmStat4T

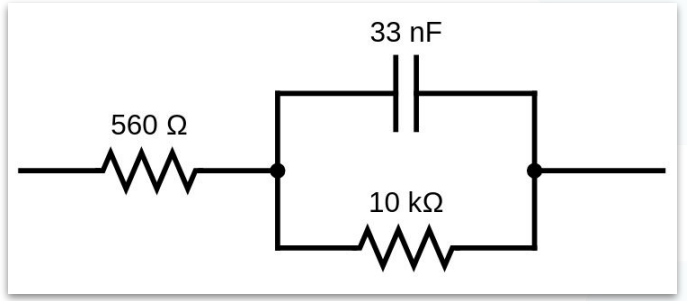
Randles Circuit







PalmSens Dummy cell



Simplified Randles Circuit

Demo: EIS in 3 lines

```python
import pypalmsens as ps

method = ps.ElectrochemicalImpedanceSpectroscopy(
    dc_potential=0.0,     # V
    ac_potential=0.01,    # 10 mV RMS
    n_frequencies=51,
    min_frequency=10,
    max_frequency=200_000,
    max_equilibration_time=3.0,  # s
    min_sampling_time=0.5,       # s
    general={'power_frequency': 50},
)

measurement = ps.measure(method, callback=print)
```

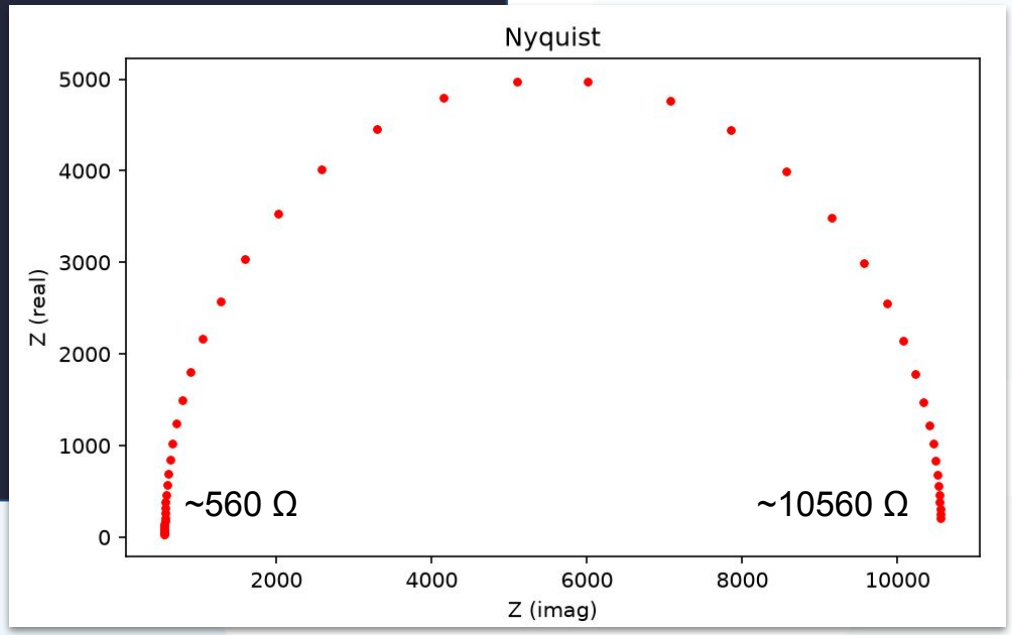
Demo: plot and export data

```
eis = measurement.eis_data[0]

# export to CSV
df = eis.dataset.to_dataframe()
df.to_csv('eis.csv', columns=['Frequency', 'ZRe', 'ZIm'])

# make Nyquist plot
import matplotlib.pyplot as plt

plt.plot(df['ZRe'], df['ZIm'], 'r.')
plt.xlabel('Z (real)')
plt.ylabel('Z (imag)')
plt.title('Nyquist')
plt.show()
```



Galvanostatic EIS

```python
import pypalmsens as ps

method = ps.GalvanostaticImpedanceSpectroscopy(
    applied_current_range='10uA',
    dc_current=0.0,
    ac_current=0.01,
    n_frequencies=51,
    min_frequency=20.0,
    max_frequency=100_000,
    max_equilibration_time=3.0,  # s
    min_sampling_time=0.5,       # s
    general={'power_frequency': 50},
)

measurement = ps.measure(method, callback=print)
```

Measurement modes

Scan frequency (default)

```
ps.ElectrochemicalImpedanceSpectroscopy(
    frequency_type='scan',
    n_frequencies=51,
    min_frequency=20,
    max_frequency=200_000,
)
```

Fixed frequency

```
ps.ElectrochemicalImpedanceSpectroscopy(
    frequency_type='fixed',
    fixed_frequency=10_000,    # Hz
)
```

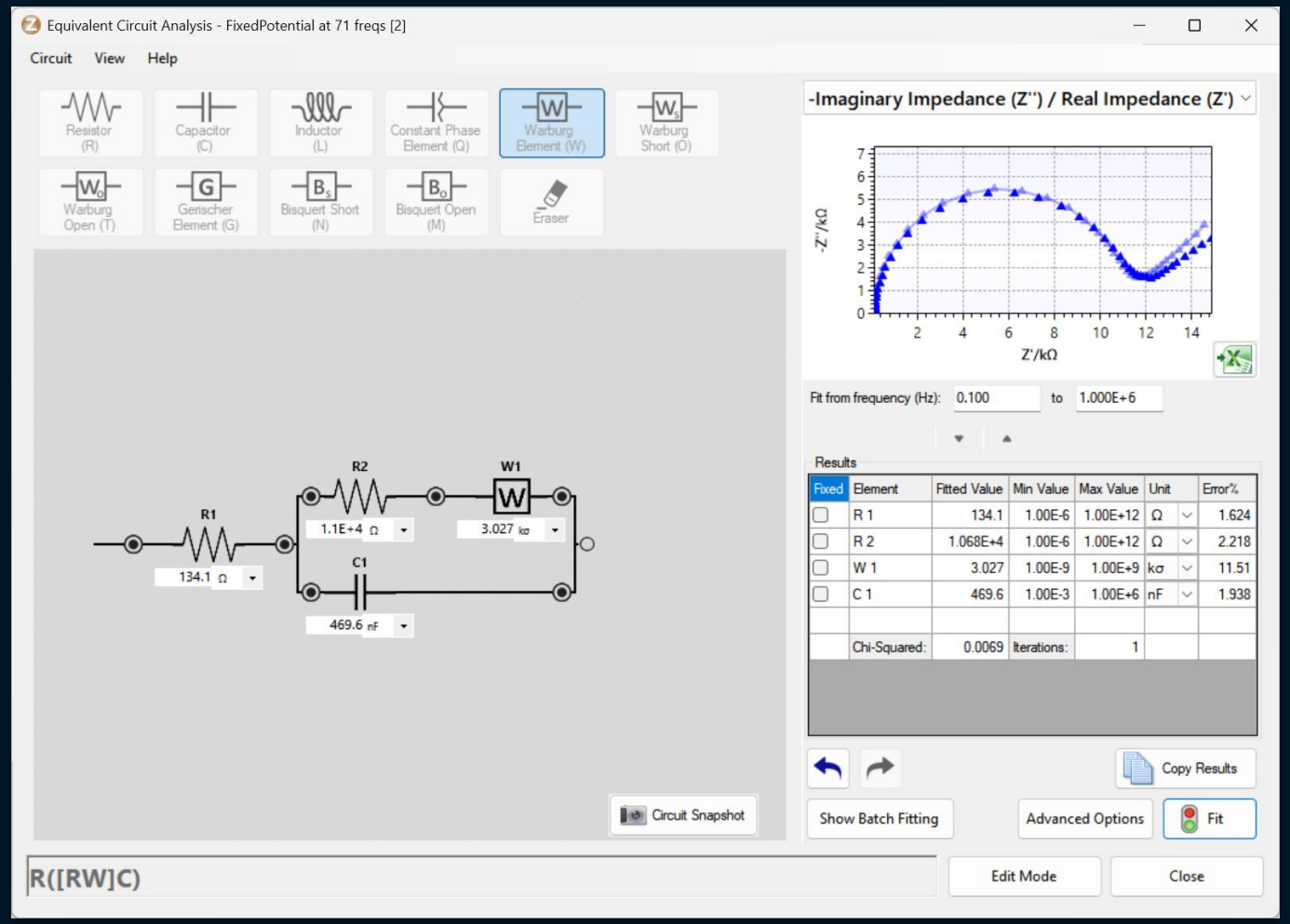
Repeat over time

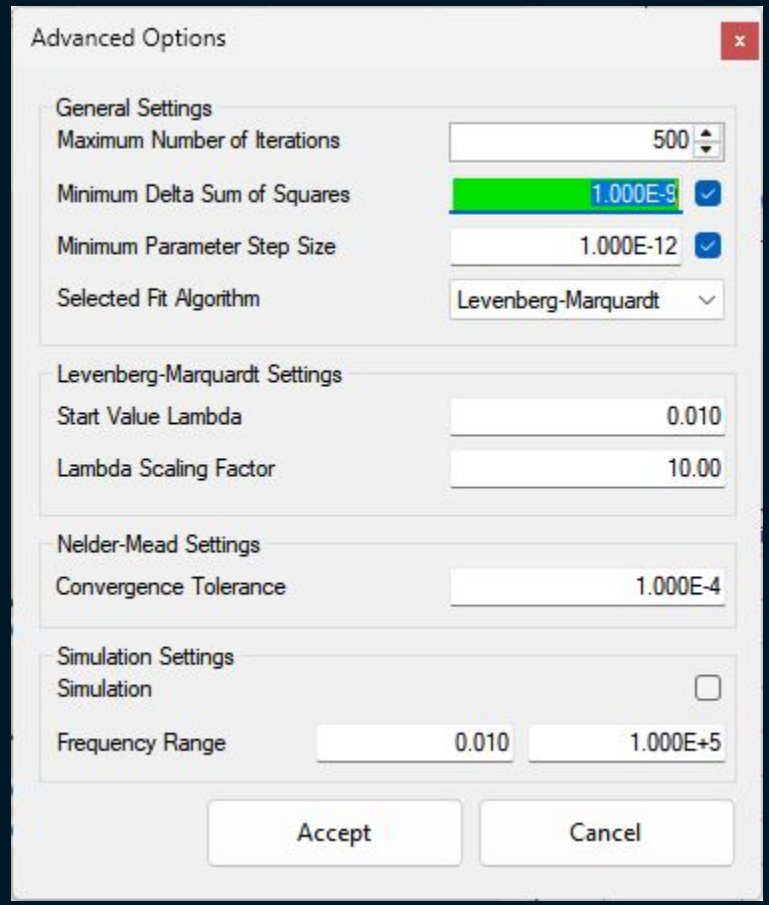
```
ps.ElectrochemicalImpedanceSpectroscopy(
    scan_type='time',
    run_time=600,    # s
)
```

Repeat over potential range

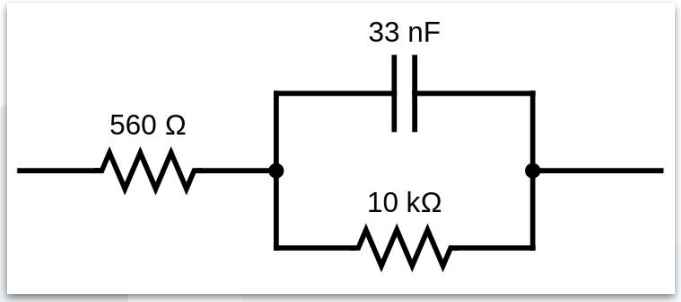
```
ps.ElectrochemicalImpedanceSpectroscopy(
    scan_type='potential',
    begin_potential=-0.5,    # V
    end_potential=0.5,       # V
    step_potential=0.1,      # V
)
```

Equivalent circuit fitting





Demo: Equivalent circuit fit



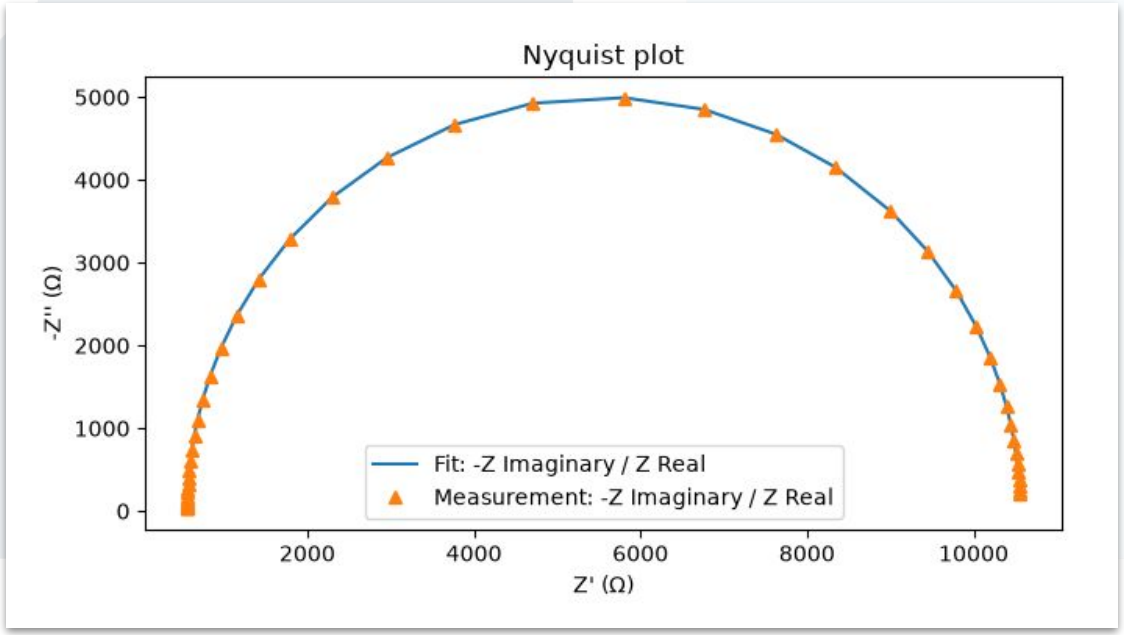

```
eis = measurement.eis_data[0]

cdc = 'R(RC)'  # circuit description code
model = ps.fitting.CircuitModel(cdc=cdc)
result = model.fit(eis)

result.plot_nyquist(eis)
result.plot_bode(eis)

print(result.parameters)
# [558.35, 9998.43, 3.32e-08]


print(result.error)  # %
# [0.103, 0.097, 0.114]
```



Set starting parameters

```python
# set fit algorithm
model = ps.fitting.CircuitModel(cdc=cdc, algorithm='leastsq')

# set starting parameters
result = model.fit(eis, parameters=[10000, 10000, 0])

# re-fit parameters
result = model.fit(eis, parameters=result.parameters)

# fixing parameters
parameters = model.default_parameters()
parameters[0].value = 560  # Ohm
parameters[0].fixed = True
result = model.fit(eis, parameters=parameters)
```

impedance.py

Community Python package for reproducible electrochemical impedance spectroscopy (EIS) analysis

```
pip install impedance
```

- Pure Python
- Scikit-learn like API
- Open-source (MIT licence)
- Community-driven

- Preprocessing
- Data validation (Kramers-Kronig)
- Equivalent circuit fitting
- Visualizations

impedancepy.readthedocs.io

Demo: impedance.py (1)

```python
import numpy as np
from impedance.models.circuits import CustomCircuit

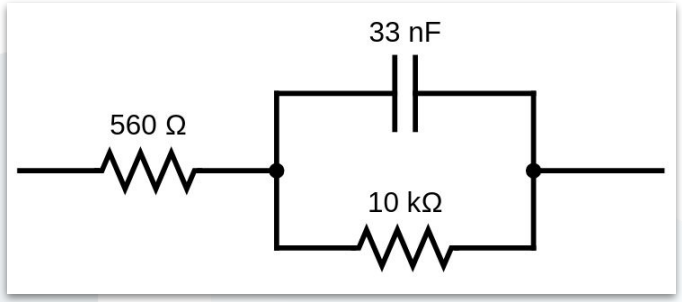
eis = measurement.eis_data[0]

F = np.array(eis.dataset['Frequency'])
Zre = np.array(eis.dataset['ZRe'])
Zim = np.array(eis.dataset['ZIm'])

Z = Zre - 1j * Zim

circuit = CustomCircuit(
    'R0-p(R1,C1)',
    initial_guess=[1000, 1000, 0]
)

ret = circuit.fit(F, Z)
```



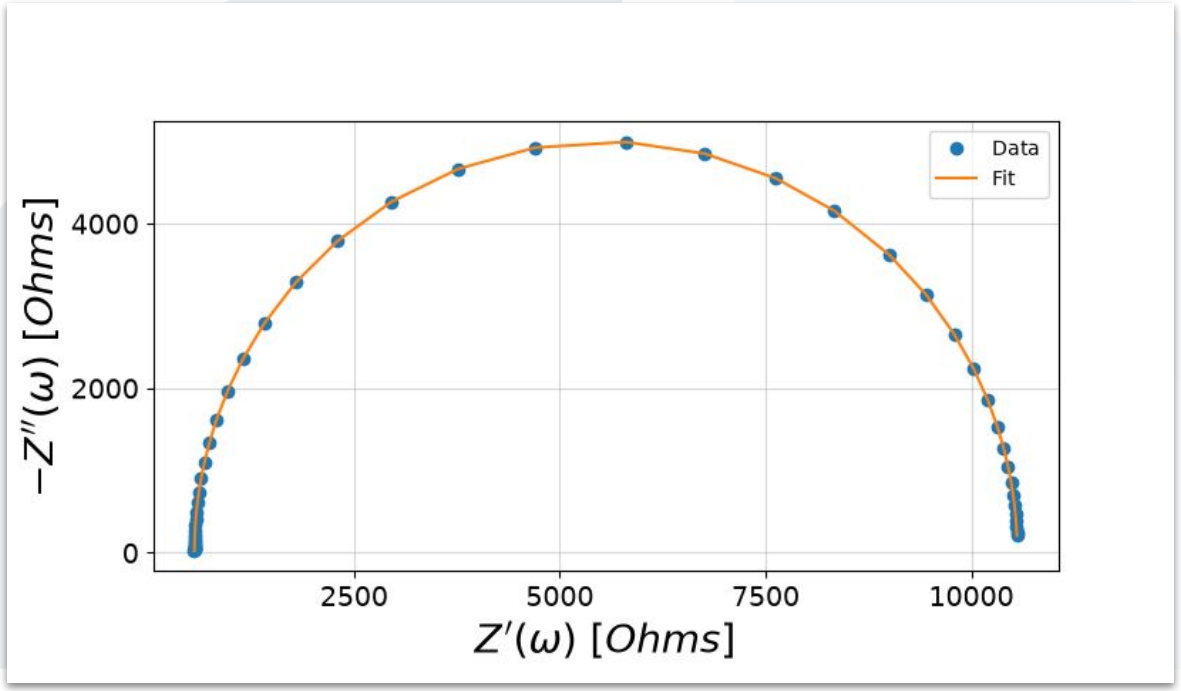

```python
print(ret)
# Circuit string: R0-p(R1,C1)
# Fit: True
#
# Initial guesses:
#      R0 = 1.00e+03 [Ohm]
#      R1 = 1.00e+03 [Ohm]
#      C1 = 0.00e+00 [F]
#
# Fit parameters:
#      R0 = 5.59e+02  (+/- 4.18e-01) [Ohm]
#      R1 = 1.00e+04  (+/- 6.39e-01) [Ohm]
#      C1 = 3.32e-08  (+/- 6.59e-12) [F]
```

Demo: impedance.py (2)

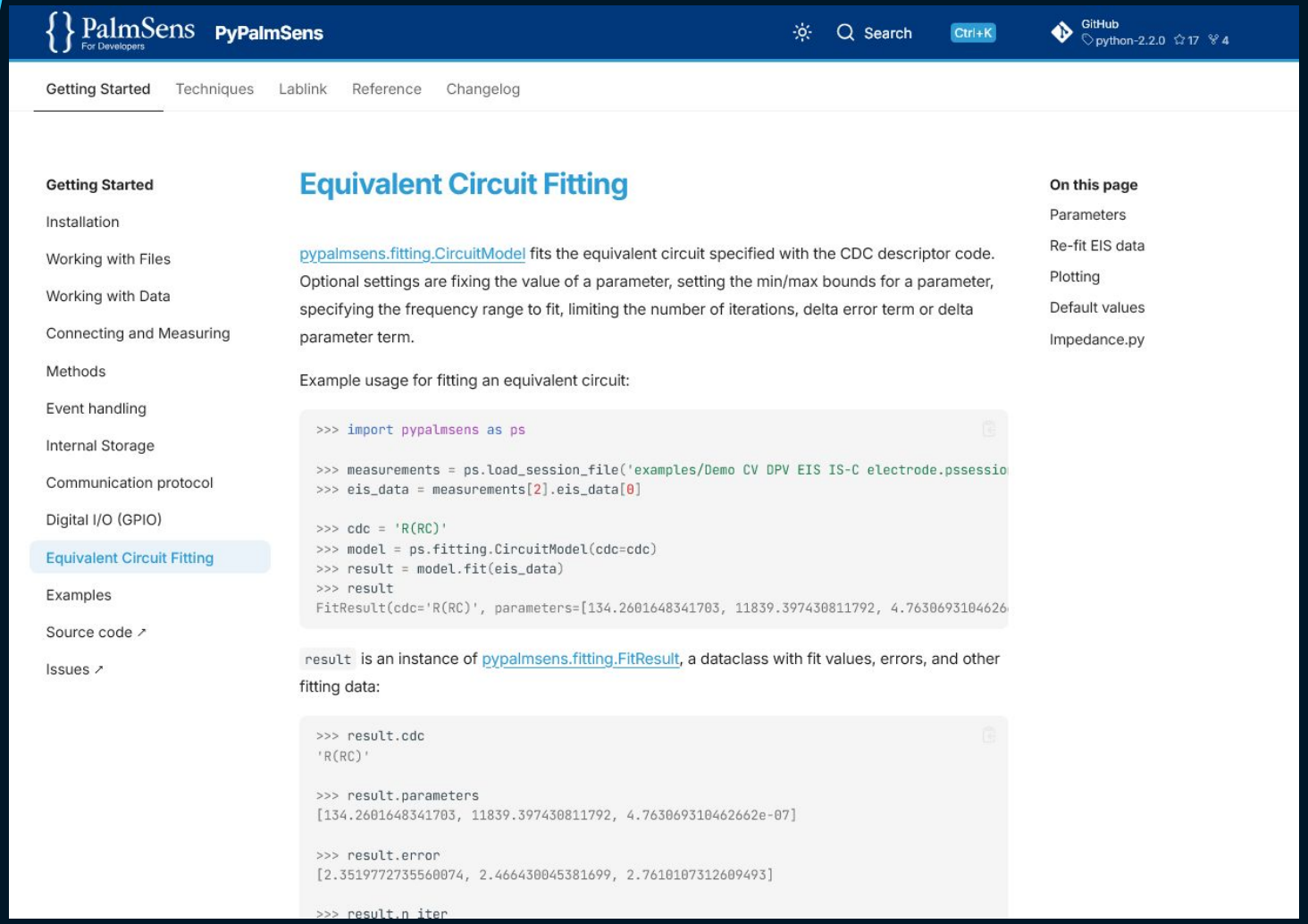
```
from impedance.visualization \
    import plot_nyquist

Z_fit = circuit.predict(F)

fig, ax = plt.subplots()
plot_nyquist(Z, fmt='o', ax=ax)
plot_nyquist(Z_fit, fmt='-', ax=ax)
plt.legend(['Data', 'Fit'])
plt.show()
```



More information



dev.palmsens.com/python/latest

Questions, issues, feedback?

support@palmsens.com

Python for PalmSens Webinar

- **Session I** - Getting started (Oct 7, 2026)
- **Session II** - Impedance Spectroscopy (Nov 4, 2026)
- **Session III** - Advanced topics: Become a pro (Dec 2, 2026)

Circuit description codes + element library

CDC	Circuit
R(RC)	Simplified Randles (no diffusion)
R(RQ)	With constant-phase element (real electrodes)
R(RWQ)	+ Warburg
R([RT]Q)	+ finite-length Warburg