Analysis of Electronic Circuits using PySpice and Scipy

Raghuttam Hombal and N. S. Ashokkumar

Automation and Simulation

PySpice

Simulate electronic circuit using Python
and the Ngspice / Xyce simulators

PySpice

Simulating the circuit

PySpice is a Python module which interface Python to the Ngspice and Xyce circuit simulators.

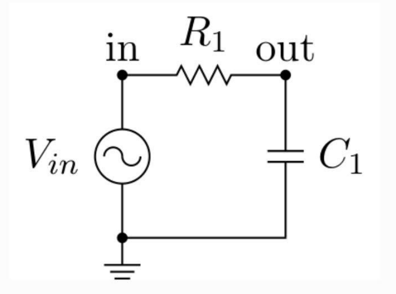
Cited: [Fabrice Salvaire - PySpice]

R-C Filter

1. Few circuits are already available in pyspice.netlist, others can be added through generated Netlist.

2. Parameters are fed into the function and simulation is started.

```
import PySpice.Logging.Logging as Logging
logger = Logging.setup_logging()

####################################################################################################

from PySpice.Plot.BodeDiagram import bode_diagram
from PySpice.Spice.Netlist import Circuit
from PySpice.Unit import *

####################################################################################################

#f# circuit_macros('low-pass-rc-filter.m4')

circuit = Circuit('Low-Pass RC Filter')

circuit.SinusoidalVoltageSource('input', 'in', circuit.gnd, amplitude=1@u_V)
R1 = circuit.R(1, 'in', 'out', 1@u_kΩ)
C1 = circuit.C(1, 'out', circuit.gnd, 1@u_uF)

#r# The break frequency is given by :math:`f_c = \frac{1}{2 \pi R C}`

break_frequency = 1 / (2 * math.pi * float(R1.resistance * C1.capacitance))
print("Break frequency = {:.1f} Hz".format(break_frequency))
#o#

simulator = circuit.simulator(temperature=25, nominal_temperature=25)
analysis = simulator.ac(start_frequency=1@u_Hz, stop_frequency=1@u_MHz, number_of_points=10,  variation='dec')
```

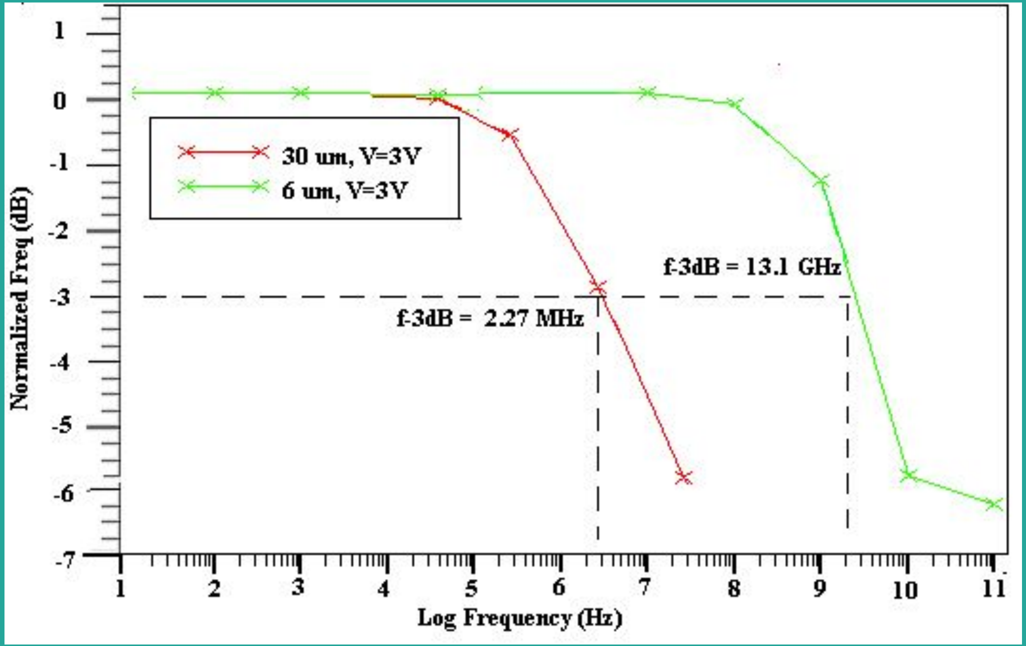


Analysis

3dB Frequency from
Frequency Response Curve

Significance of 3dB Frequency

The low & high cut-off frequency, at which the power is reduced to one-half of the maximum power and the range between the two is the bandwidth of the signal.



From: High performance silicon lateral PIN photodiode by Tasirin, S. & Menon, P Susthitha & Ahmad, Ibrahim & Abdullah, Siti Fazlili. (2013)

Optimize (Curve Fit)

- Process of constructing a curve, or mathematical function, that has the best fit to a series of data points, possibly subject to constraints.

```python
def func(x, a, b):
    return a*np.exp(b * x)
def fittingCurve(xD,yD):
    popt, pcov = curve_fit(funcPoly, xD, yD)
    return popt
def f(x,popt):
    return funcPoly(x,*popt)
```

```python
def funcPoly(x,a,b,c,d,e):
    return a+b*x+c*x**2+d*x**3+e*x**4
```

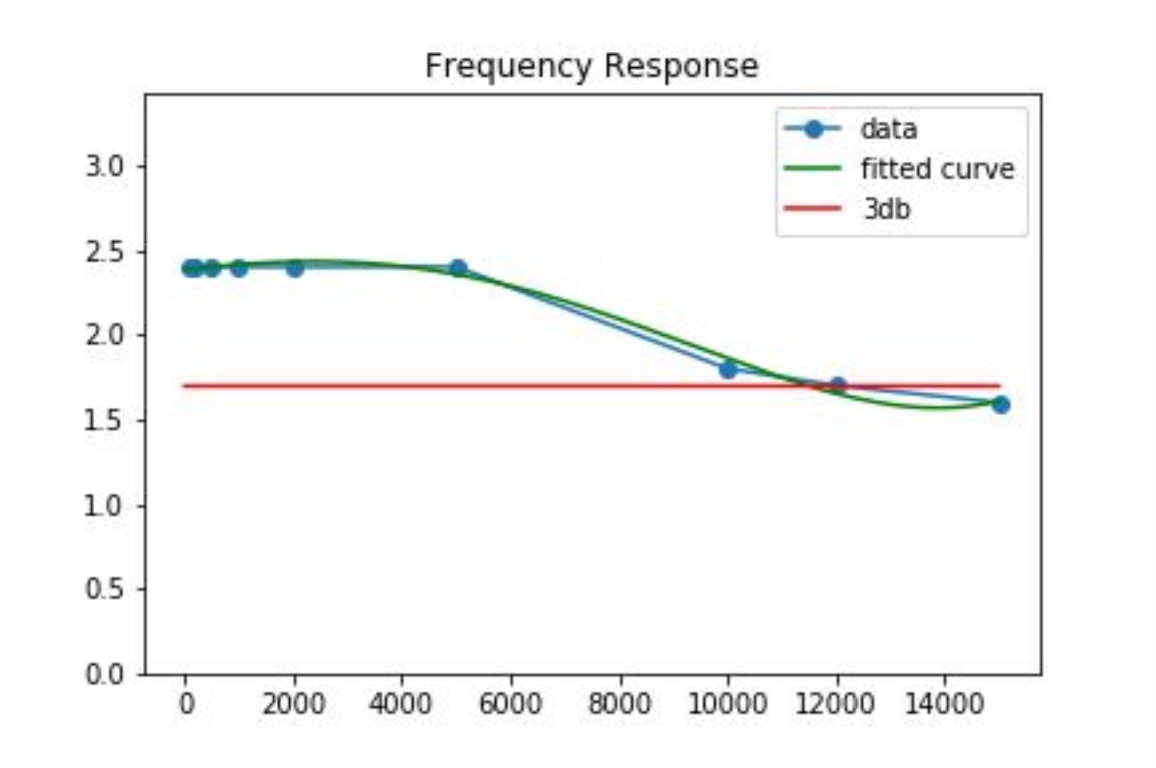
Documentation available in the website: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html

Approach to determine the 3dB frequency

Approach to the problem:

- Take an initial frequency (independent value) and longer ‘dx’
- Check if that satisfies the required dependent value
- Depending on the status increment the ‘dx’ or make it smaller.

```
def findx(y0,popt):
    x0,dx=0,10
    approx=y0-f(x0,popt)
    flag=True
    while dx>1e-5:
        if approx<0:#What decides to calculate up or down
            x0=x0+dx
        else:
            x0=x0-dx
            dx=dx/10
            x0=x0+dx

        approx=y0-f(x0,popt)
    return x0
```




```
The 3dB frequency is estimated to be =  11501.316509999999
```

Developed using Jupyter Notebook

Advantages

- Scope to change Parameters
- Lesser Processing capacity
- Wider range of applications

Conclusion

It was just an example

File Available at ‘github’:
https://github.com/group4pgs/frequencyResponseAnalysis

References

- Fabrice Salvaire - PySpice Documentation
- Scipy Optimize Curve Fit - Scipy Documentations

THANK YOU

Any Questions?