# RF Network Analysis of the WEST ICRH Antenna with the Open-Source Python scikit-rf Package

Julien Hillairet

# RF Network Analysis of the WEST ICRH Antenna with the Open-Source Python scikit-rf Package

Julien Hillairet[a)]

*CEA, IRFM, F-13108 St-Paul-Lez-Durance, France*

[a)]*Corresponding author: julien.hillairet@cea.fr*

**Abstract.** `Scikit-rf` is an open-source Python package developed for RF/Microwave engineering. The package provides a modern, object-oriented library for network analysis and calibration which is both flexible and scalable. Besides offering standard microwave network physics and operations, it is also capable of advanced operations such as interpolating between an individual set of networks or deriving network statistical properties. The package also allows direct plotting of rectangular plots, Smith Charts or automated uncertainty bounds. In this paper, the `scikit-rf` package is used to simulate the WEST ICRH antennas. The antenna is modelled by connecting the various elements that compose it, separately full-wave modelled. Tunable elements, such as the matching capacitors, can be either created from ideal lump components or from interpolating full-wave calculations performed at various capacitance configurations. Finally, a numerical antenna model of a WEST ICRH antenna can be used offline or online by using operating-system-level virtualization services.

## INTRODUCTION

Modern science is founded on hypothesis testing and statistical significances of its derived results. Reproducing experiments, experimental or numerical, is the reason why scientists can gain confidence in their conclusions. However, during the last decades, the number of codes and libraries developed at an individual or a laboratory scale only increased. When these tools are not open-sourced, it leads to an obvious reproducibility problem as one should only rely on authors claims. To conform to the necessity of reproducibility in science, software sources used in physical and engineering researches should ideally be made open when possible [1]. The case of RF network manipulation and analysis is no different.

`Scikit-rf` [2] is a Python package produced for RF/Microwave engineering [3]. The package is licensed under the Berkeley Software Distribution (BSD) licence and is actively developed by volunteers on GitHub. The package provides a modern, object-oriented library for RF network analysis and calibration. Besides offering standard microwave network operations, such as reading/writing Touchstone files (`.sNp`), connecting or de-embedding N-port networks, frequency/port slicing, concatenation or interpolations, it is also capable of advanced operations such as Vector Network Analyzer (VNA) calibrations, time-gating, interpolating between an individual set of networks, deriving network statistical properties and supports Virtual Instrument for direct communication to VNAs. The package also allows straightforward plotting of rectangular plots (dB, mag, phase, group delay, etc), Smith Charts or automated uncertainty bounds. As the package is developed in Python, it makes it naturally compatible with the rich and modern scientific Python libraries [4], for example `scikit-learn` for machine learning tasks [5] or `PlasmaPy` [6] for plasma physics. Using Jupyter notebooks documents [7], modelling approaches and results can be shared and even directly reproduced using tools such as Binder [8] [9] or Google Colab [10], by other researchers (or future self) few months or years after work had been made.

In this paper, some functionalities of the `scikit-rf` package are presented through the example of modelling the WEST (Tungsten-W Environment in Steady-state Tokamak) Ion Cyclotron Resonance Heating (ICRH) antenna. The antenna circuit is made by connecting the various elements that compose it, separately full-wave modelled or using equivalent lumped models, like done with proprietary codes (i.e. owned by the person or the firm who has developed it) in [11, 12] or with commercial software such as ANSYS Designer in [13]. This paper is accompanied by Jupyter notebooks illustrating in more details the code leading to the results presented here [14].

## WEST ANTENNA MATCHING CAPACITORS

The three WEST ICRH antennas are load tolerant resonant structures equipped with internal vacuum matching capacitors. These capacitors are tuned before the antenna operation or are real-time controlled to maintain low reflected power toward the RF sources [13]. In order to forge a relevant numerical model of the antenna, it is thus needed to

modify the capacitance of these matching capacitors. At least two approaches can be used: (i) using an equivalent lumped model of the capacitor or (ii) interpolating the capacitor s-parameters from full-wave simulations (if available). The challenge with the former is to use a relevant model to capture the RF behaviour of the capacitor in its operational environment, where parasitic capacitances/inductances may arise from surrounding elements and are mostly unknown. The main difficulty with the later is to design a full-wave parametrized model geometrically accurate, assuming dimensions and materials of the capacitor are known. Both approaches can be implemented with `scikit-rf` as the package is able to compute s-parameters of RF circuits made from ideal lumped elements (resistor/capacitor/inductor, in series/shunt configuration) or to interpolate between N-port networks with same frequencies [15]. In the example below, both approaches are mixed: the capacitor (from COMET), which has been previously simulated with ANSYS HFSS for few different capacitance configurations in [16], is used to extract the lumped element parameters. Once fitted to an equivalent model, the capacitor networks can be driven as a function of their capacitances, while s-parameter interpolation only depend on of geometrical parameter (like the distance between capacitor electrodes).

A N-port electrical network is described in `scikit-rf` with the `Network` object, defined from its network parameters (s, z, y-parameters), port characteristic impedances and frequency information. It can be created from a file or directly from data. Importing all the touchstone files located in directory into a collection of `Network` in a `NetworkSet` object and plotting the real part of the $S_{21}$ parameter for each network for a subset of the frequency range is straightforward [17]:

```
import skrf as rf
capas_set = rf.NetworkSet.from_dir('S_Matrices/Antenna/Capacitor')
capas_set['20-80 MHz'].plot_s_re(m=1, n=0)  # Indexing starts at 0 in Python
```

Assuming the `NetworkSet` contains $M$ networks coming from a parameter sweep of $M$ values described in a `param_values` array, creating an interpolated network for a new value is achieved with:

```
ntw = capas_set.interpolate_from_network(param_values, new_value)
```

Figure 1(b) illustrates the interpolation of $S_{11}$ and $S_{21}$ at a given frequency for intermediate values of the swept parameter, here related to the distance between capacitor electrodes (noted $D_{cylinders}$). Obviously, the interpolation precision increases with the number of networks in the set.

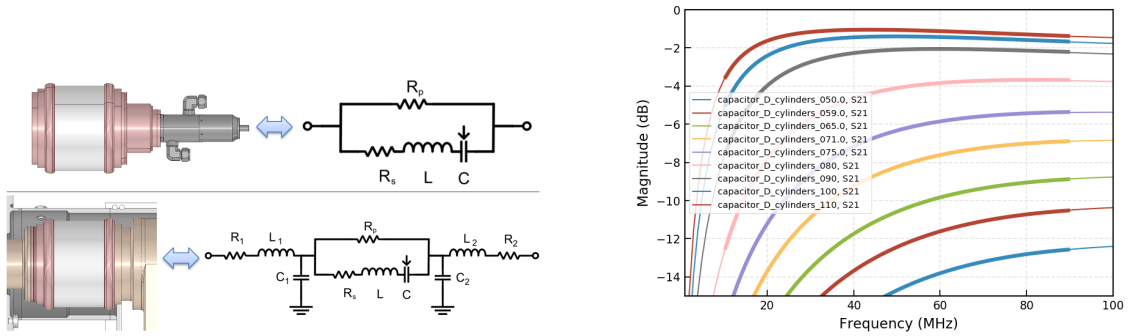
The equivalent lumped model proposed by the capacitor manufacturer (COMET) in the capacitor documentation consists in a RLC circuit (in parallel with a resistance $R_p$). However, in the case of the WEST ICRH antenna, the capacitor is inserted inside the antenna and is surrounded by the outer conductor elements, creating additional stray capacitances and inductances. In this situation, its turns out that the equivalent capacitance of the setup is modified [16]. The RF capacitance can nevertheless be extracted from full-wave calculation using a refined equivalent lumped model with additional shunt capacitances and RL elements around the capacitor itself to model the proximity of the outer conductor housing and propagation effects [18] (Figure 1(a)). Such electrical network can be constructed with `scikit-rf` with the following function:

```
def eq_capa(C, R, L, R1, C1, L1):  # here we neglect the parallel resistance R_p
    freq = rf.Frequency(start=5, stop=95, npoints=101, unit='MHz')
    line = rf.media.DefinedGammaZ0(frequency=freq, z0=50)  # ideal transmission line media
    pre  = line.resistor(R1) ** line.inductor(L1) ** line.shunt_capacitor(C1)
    post = line.shunt_capacitor(C1) ** line.resistor(R1) ** line.inductor(L1)
    capa = line.resistor(R) ** line.inductor(L) ** line.capacitor(C)
    return pre ** cap ** post  # ** is the scikit-rf network cascading operator
```

Using the standard optimization toolbox shipped in the `scipy` package, the model parameters can be fitted for each full-wave network. Once done, full-wave references and lumped model can be compared. The Figure 1(b) shows the calculated $S_{21}$ parameter which fit nice with the reference full-wave values. For most capacitance configurations, the error is below -40 dB. Taking constant lumped parameters ($R_s, L_s, C_1 = C_2, R_1 = R_2, L_1 = L_2$) but $C$, the still matches well the full-wave reference except for the lowest range of capacitance values, in a region where the relationship between the electrodes distance and the capacitance is no more linear. The model can thus be used to determine a realistic relationship between the RF capacitance and the capacitor configuration (driven by motors).

## Antenna RF Circuit

In `scikit-rf`, there is at least two ways of building a RF circuit made of various networks, both illustrated in the accompanying notebooks. The first method consists in connecting networks one by one using `Network` methods.




(a) equivalent lumped models of a WEST antenna matching capacitor, outside and inside the antenna.

(b) Real part of $S_{21}$ between the reference from full-wave calculations (thin lines) and the lumped model (thick lines).

FIGURE 1: Equivalent lumped model of the WEST ICRH antenna capacitor.

A second method consists in defining all the connections between networks and ports at once time, then use the `Circuit` object to build the resultant network. While the later method can be more verbose than the former, it has some advantages when the circuit has many connections. `Circuit` uses a systematic solving method giving both internal (for each circuit connections) and external (for ports) s-parameters for an arbitrary number of networks and interconnections and independently of the circuit excitation [19]. By imposing excitations at ports, it allows obtaining voltages/currents inside the circuit.

The Figure 2 schematizes a half WEST ICRH antenna, where the impedances connected to the capacitors model the antenna loading. The associated `Circuit` is built from the code example given in Figure 3. In this example, each network has been previously imported or created. The Figure 4 illustrates the voltage and current at the capacitors, extracted from the circuit for a specified excitation. The full antenna, which consists in two adjacent similar circuits can be built similarly. `Circuit` allows visualizing the circuit graph using the `networkx` package [20], which is useful to check that the intended circuit is the one expected (such as connections, port numbers and characteristic impedances). The Figure 5 illustrates the full antenna setup loaded by a 4-port networks modelling the coupling of the antenna front-face to a plasma.

## CONCLUSION

The RF circuit of the WEST ICRH antennas has been modelled using the open-source Python package `Scikit-rf`. The package provides a modern, object-oriented library for network analysis and calibration which is both flexible and scalable. As the package is developed in Python, it makes it naturally compatible with the rich and modern scientific Python libraries, testing and code coverage frameworks and reproducible modelling approaches using on-line virtualization services. The WEST ICRH antenna is modelled by connecting the various elements that compose it, separately full-wave modelled. Tunable elements, such as the matching capacitors, can be either created from ideal lump components or from interpolating full-wave calculations performed at various capacitance configurations. Being open-source, using `scikit-rf` to create the antenna numerical model with reproducible work-flow adds confidence in the fact that future users could use/maintain and extend it.

## ACKNOWLEDGMENTS

The author would like to thank Alex Arsenovic and the scikit-rf developers [21] for the scikit-rf package and their support. The capacitor full-wave model used to interpolate and extract lumped parameters had been made by W.Helou [16].

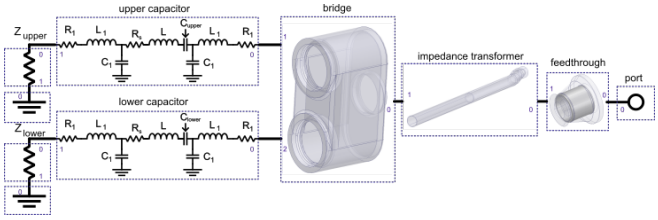



FIGURE 2: Half Antenna schematic.

```python
connections = [
  [(port1, 0), (window, 0)],
  [(window, 1), (impedance_transformer, 0)],
  [(impedance_transformer, 1), (bridge, 0)],
  [(bridge, 1), (capa_upper, 0)],
  [(bridge, 2), (capa_lower, 0)],
  [(capa_upper, 1), (load_upper, 0)],
  [(capa_lower, 1), (load_lower, 0)],
  [(load_upper, 1), (ground_upper, 0)],
  [(load_lower, 1), (ground_lower, 0)]
]
# create the Circuit object and associated network
crt = rf.Circuit(connections)
ntw = crt.network  # here a 1-port.
```

FIGURE 3: `scikit-rf Circuit` setup of an half-antenna.

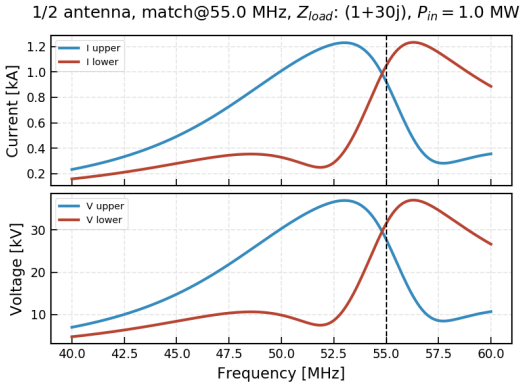



FIGURE 4: Voltages and currents at the matching capacitor for a defined power excitation, antenna loading and matching conditions.

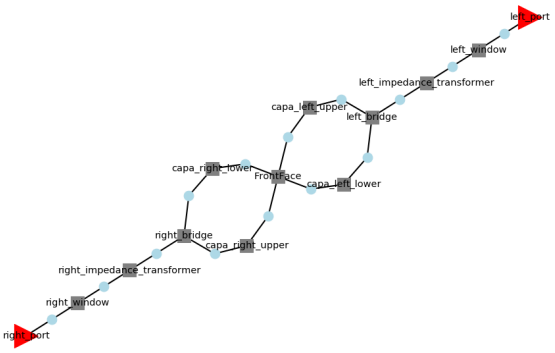



FIGURE 5: Circuit graph of a complete antenna.

## REFERENCES


1. D. C. Ince, L. Hatton, and J. Graham-Cumming, Nature **482**, 485–488 (2012).
2. Http://www.scikit-rf.org.
3. A. Arsenovic, H. Forsten, and M. e. a. Bauwens, (2018), 10.13140/RG.2.2.10056.57606.
4. K. J. Millman and M. Aivazis, Computing in Science and Engineering **13**, 9–12 (2011).
5. F. Pedregosa, G. Varoquaux, and e. a. Gramfort, Journal of Machine Learning Research **12**, 2825–2830 (2011).
6. PlasmaPy Community, N. A. Murphy, and A. J. e. a. Leonard, (2018), 10.5281/zenodo.1238132.
7. T. Kluyver, B. Ragan-Kelley, and F. e. a. Pérez, Physical Review **16**, 87–90 (2016).
8. Https://mybinder.org/.
9. Project Jupyter, Matthias Bussonnier, and Jessica Forde et al., "Binder 2.0 - Reproducible, interactive, sharable environments for science at scale," in *Proceedings of the 17th Python in Science Conference*, edited by Fatih Akici, David Lippa, Dillon Niederhut, and M. Pacer (2018), pp. 113 – 120.
10. Https://colab.research.google.com/.
11. F. Durodié, P. Dumortier, and W. e. a. Helou, **070013**, p. 070013 (2015).
12. W. Helou, P. Dumortier, and F. e. a. Durodié, "SIDON: A simulator of radio-frequency networks. Application to WEST ICRF launchers," in *AIP Conference Proceedings*, Vol. 1689 (2015) p. 070004.
13. J. Hillairet, P. Mollard, and Y. e. a. Zhao, "Ion cyclotron resonance heating systems upgrade toward high power and CW operations in WEST," in *AIP Conf. Proc. 1689, 070005*, Vol. 1689 (2015).
14. J. Hillairet, Jupyter notebooks in github: https://github.com/jhillairet/RFPPC2019/, may 2019.
15. Similar to the `NPort_Multi` component in ANSYS Designer.
16. W. Helou, "Design and operation of antennas at the ion cyclotron and lower hybrid range of frequencies for nuclear fusion reactors," Ph.D. thesis, Aix-Marseille University 2018.
17. Due to space constraints, the code snippets are complete minimal example but does not necessary reproduce all figure details such as line styles, markers or labels. Readers should refer to associated notebooks for the full code listing.
18. The parallel resistor $R_p$ can be neglected in the tens of MHz frequency range considered here.
19. P. Hallbjörner, Microwave and Optical Technology Letters **38**, 99–102jul (2003).
20. A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring Network Structure, Dynamics, and Function using NetworkX," in *Proceedings of the 7th Python in Science Conference* (2008), pp. 11–15.
21. Https://github.com/scikit-rf/scikit-rf/graphs/contributors.