

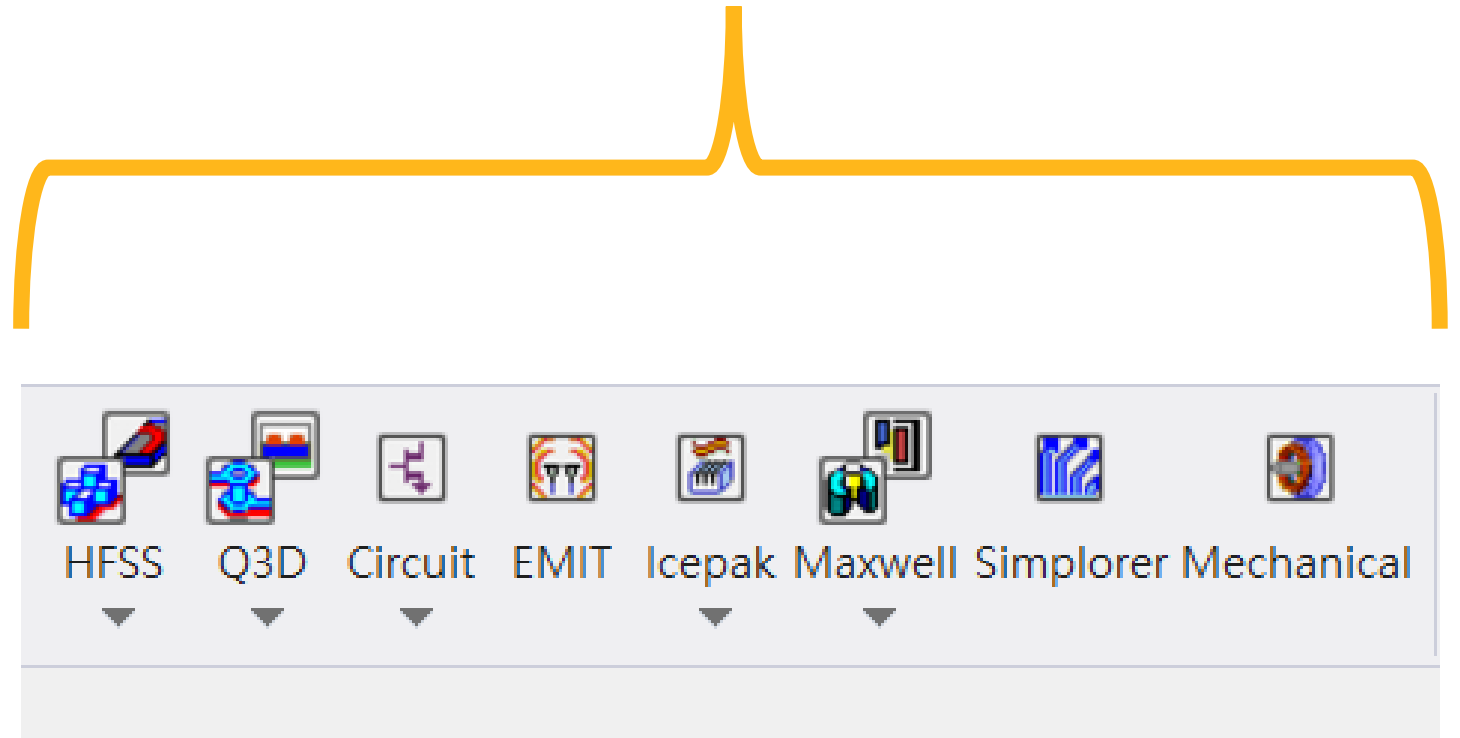
# AEDT Automation Development with PyAEDT

Lin, Ming Chih



# / Agenda

- PyAEDT Overview
- Examples Demo
- Installation
- IDE Introduction
- Scripting Demo



## 求解器設定

## 3D建模

## 激發源設定

## 境界設定

## 參數掃描

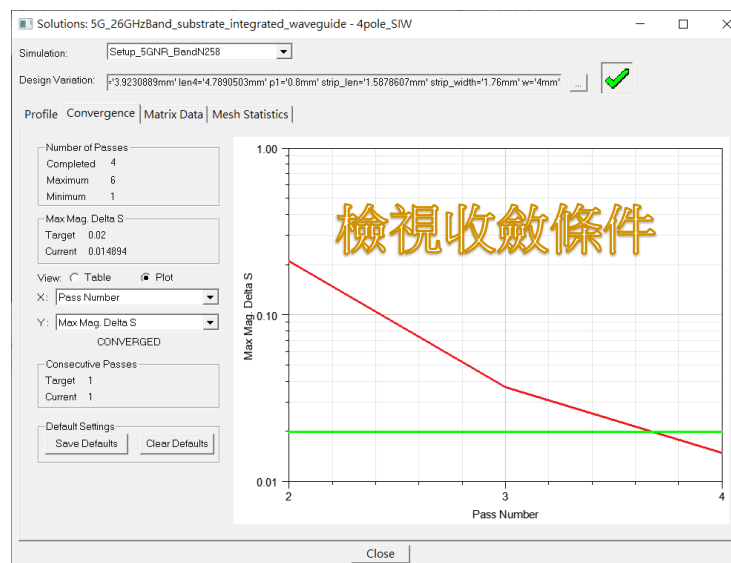
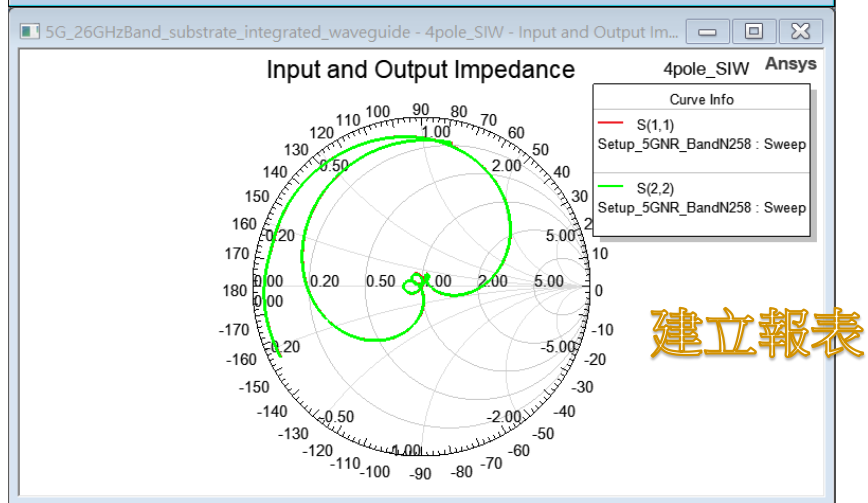
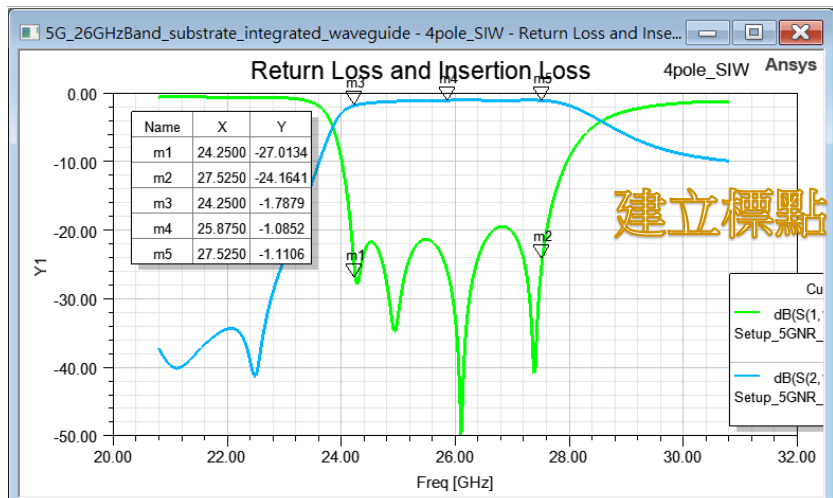
## 材料參數設定

## 收斂設定

## 初始網格設定

掃頻設定

# PostSim Setting in AEDT HFSS



Equivalent Circuit Export Options

Default Directory: C:\AnsysEM\AnsysEM21.2\Win64\Examples\HFSS\Filters

Format: PSpice (\*.lib)

Full Wave Spice

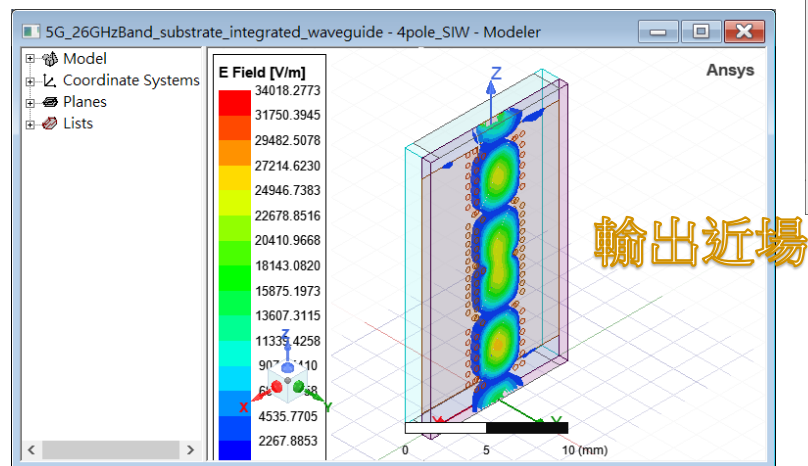
- ☒ Full Wave Spice Export
- Desired Fitting Error (Percent): 0.5
- Maximum Order: 10000
- ☒ Use Common Ground
- ☐ Enforce Passivity

Lumped Element Export (Low Bandwidth): 5G\_26GHzBand\_substrate\_integrated\_wavegui

Partial Fraction Expansion for Matlab (\*.m): 5G\_26GHzBand\_substrate\_integrated\_wavegui

OK Cancel

模型輸出



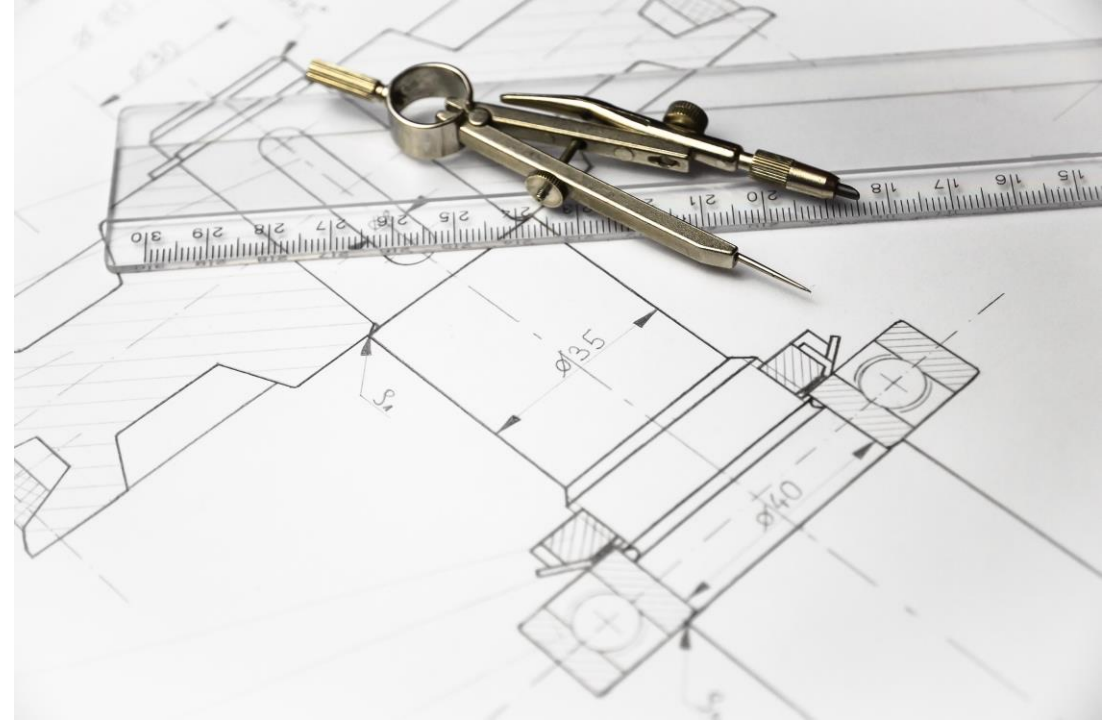
# / Typical GUI-Based Setting Flow

GUI is the native operation mode but there exist drawbacks:

- GUI license is long time occupied.
- Setting mistakes is easy to happen.
- Settings are distributed in dozens of windows, difficult to inspect.
- Comments of the settings are not supported.
- Hard to compare setting and data between projects and designs.
- Takes time to organize report and simulation data
- Advanced data processing is limited(e.g. Machine Learning)

# / Python + PyAEDT + Jupyter Notebook/Streamlit

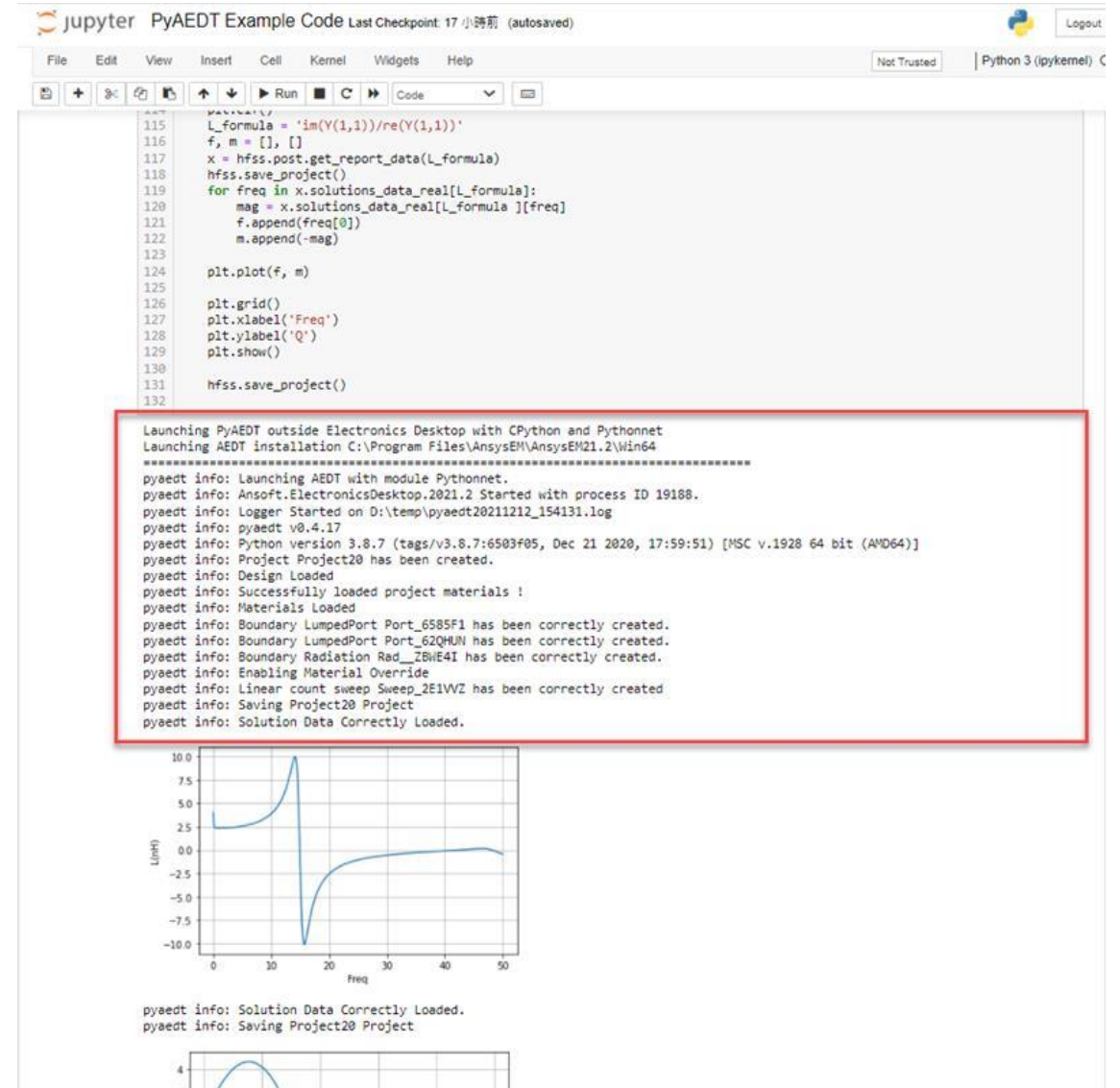
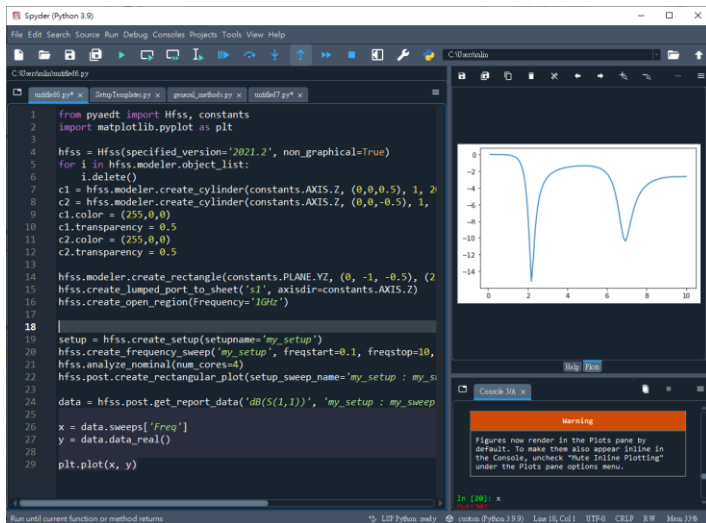
- Research and Design Optimization, like:
  - Design of Experiment
  - Variation and Yield Analysis
  - Data Processing
- Multi-Physics
  - Electrical-Thermal-Mechanical
  - Electrical-Optical
  - ...
- Automate Routine Jobs, like:
  - Model Extraction
  - Signal Process
- Interactive Training Material, like:
  - Mode Theory
  - Array Antenna Mechanism





# Examples

- Dipole Antenna Sweep
- Spiral Inductor Simulation
- PCB Model Extraction
- Package 3D Modeling



# PyAEDT Overview



# / What is PyAEDT

- PyAEDT is a Python library that interacts directly with the AEDT API to make scripting simpler for the end user. It uses an architecture that can be reused for all AEDT 3D products (HFSS, Icepak, Maxwell 3D, Q3D and Mechanical) as well as 2D tools and circuit tools like Nexxim and Simplorer.
- Finally, it provides scripting capabilities in Ansys layout tools like HFSS 3D Layout and EDB. Its class and method structures simplify operation for the end user while reusing information as much as possible across the API.



# Classical API

A quick and easy approach for automating a simple operation in the AEDT UI is to record and reuse a scripts. However, disadvantages of this approach are:

- Recorded code is dirty and difficult to read and understand.
- Recorded scripts are difficult to reuse and adapt.
- Complex coding is required by many global users of AEDT.
- No IDE can be used for code development.
- C-python modules can't be integrated.

# The Main Advantages of PyAEDT API

- Automatic initialization of all AEDT objects, such as desktop objects like the editor, boundaries, and so on
- Error management
- Log management
- Variable management
- Compatibility with IronPython and CPython
- Simplification of complex API syntax using data objects while maintaining PEP8 compliance.
- Code reusability across different solvers
- Clear documentation on functions and API
- Unit tests of code to increase quality across different AEDT versions



pyansys



全部

影片

新聞

圖片

購物

更多

工具

約有 11,400 項結果 (搜尋時間 : 0.36 秒)

<https://github.com/pyansys> ▾ [翻譯這個網頁](#)

## PyAnsys - GitHub

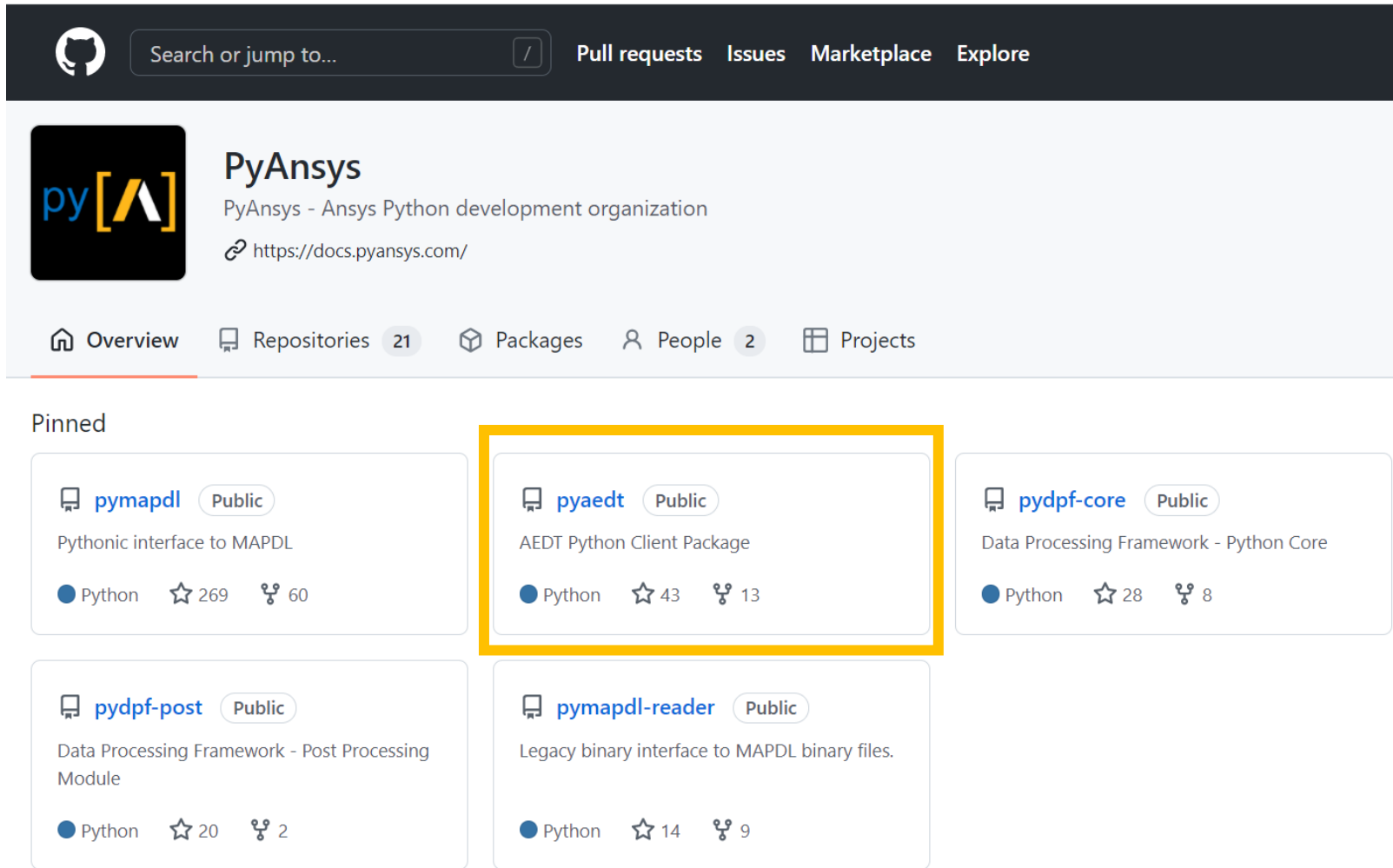
**PyAnsys** - Ansys Python development organization. **PyAnsys** has 21 repositories available.

Follow their code on GitHub.

[Pyansys/pyaedt: AEDT Python...](#) · [Pyansys/pydpf-post: Data...](#)

您曾多次瀏覽這個網頁。上次瀏覽日期：2021/12/25

# / Source Code in Github



The screenshot displays the GitHub profile for the PyAnsys organization. The header includes the GitHub logo, a search bar, and navigation links for Pull requests, Issues, Marketplace, and Explore. The organization's name, PyAnsys, is prominently displayed, along with its description as the Ansys Python development organization and its website URL, https://docs.pyansys.com/. Below the organization name, there are tabs for Overview, Repositories (21), Packages, People (2), and Projects. The 'Pinned' section lists five repositories: pymapdl, pyaedt, pydpf-core, pydpf-post, and pymapdl-reader. The repository 'pyaedt' is highlighted with a yellow border. Each repository card shows the repository name, its public status, a brief description, the programming language (Python), and the number of stars and forks.

| Repository     | Status | Description  | Language | Stars | Forks |
|----------------|--------|--|----------|-------|-------|
| pymapdl        | Public | Pythonic interface to MAPDL                        | Python   | 269   | 60    |
| pyaedt         | Public | AEDT Python Client Package                         | Python   | 43    | 13    |
| pydpf-core     | Public | Data Processing Framework - Python Core            | Python   | 28    | 8     |
| pydpf-post     | Public | Data Processing Framework - Post Processing Module | Python   | 20    | 2     |
| pymapdl-reader | Public | Legacy binary interface to MAPDL binary files.     | Python   | 14    | 9     |



# PyAEDT Documentation 0.4.18


## Introduction

PyAEDT is intended to consolidate and extend all existing functionalities around scripting for Ansys Electronics Desktop (AEDT) to allow reuse of existing code, sharing of best practices, and increased collaboration. PyAEDT is licensed under the [MIT License](#).

PyAEDT includes functionality for interacting with the following AEDT tools and Ansys products:

- HFSS and HFSS 3D Layout
- Icepak
- Maxwell 2D/3D and RMXprt
- Q3D/2DExtractor
- Mechanical

Installation User Guide **API Documentation** Examples Code Guidelines Contributing

 Search the docs ...

Initial Setup and Launching AEDT

AEDT Applications

Message Manager

AEDT Modules

EDB Modules

Modeler and Primitives

Mesh Classes

Material and Stackup

Setup

Postprocessing

Setup Templates

<https://aedtdocs.pyansys.com/API/index.html#>


## API Documentation

Welcome to PyAEDT API documentation. Use the search feature or click on the following links to view the API documentation.


- [Initial Setup and Launching AEDT](#)
  - [Initial Setup and Launching AEDT Locally](#)
  - [Initial Setup and Launching AEDT Remotely](#)
- [AEDT Applications](#)
  - [pyaedt.Desktop](#)
  - [pyaedt.Hfss](#)
  - [pyaedt.Q3d](#)
  - [pyaedt.Q2d](#)
  - [pyaedt.Edb](#)
  - [pyaedt.Maxwell2d](#)
  - [pyaedt.Maxwell3d](#)




# / API Example



Installation   User Guide   API Documentation   **Examples**   Code Guidelines   Contributing



 Search the docs ...

EDB Geometry Creation

Siwave Analysis from EDB Setup

IPC 2581 Export

5G linear array antenna

Fully parameterized design

Plot Nets with Matplotlib


Parametric Via Creation

Coordinate System Creation

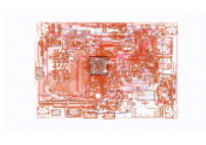
Optimetrics Setup

Polyline Creation

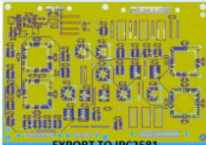
Advanced Far Field Postprocessing



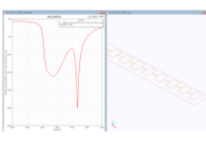
EDB Geometry  
Creation



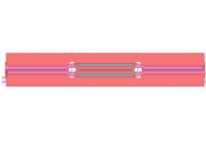
Siwave Analysis  
from EDB Setup




IPC 2581 Export



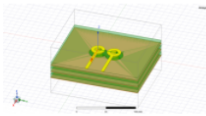
5G linear array  
antenna



Fully  
parameterized  
design



Plot Nets with



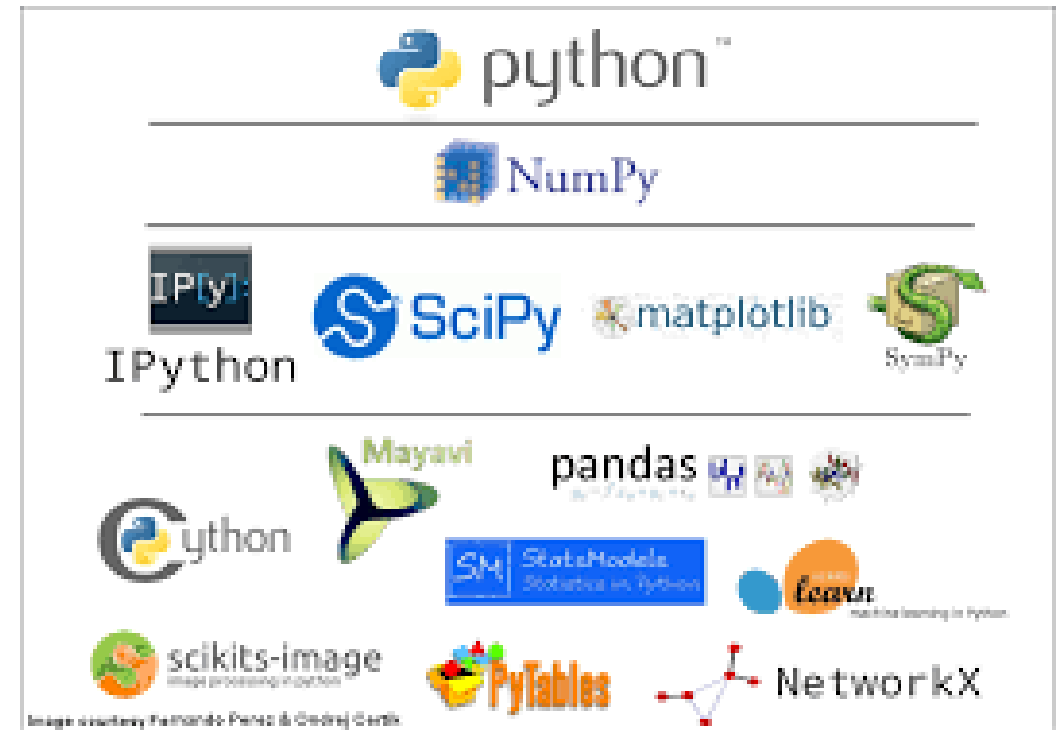
Parametric Via

# Installation of PyAEDT Module

# / Python安裝: <https://www.python.org/downloads/windows/>

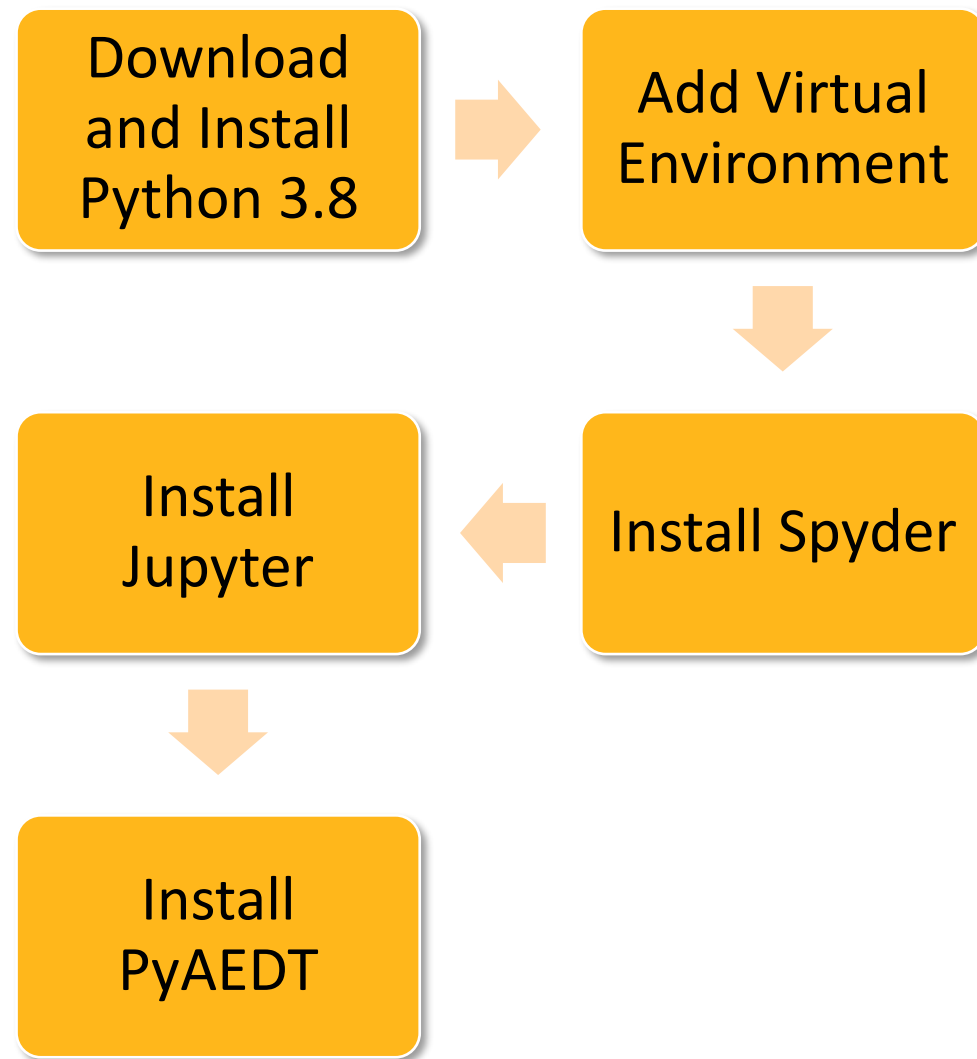
Note that Python 3.8.9 cannot be used on Windows XP or earlier.

- Download [Windows embeddable package \(32-bit\)](#)
- Download [Windows embeddable package \(64-bit\)](#)
- Download [Windows help file](#)
- Download Windows **installer** (32-bit)
- Download Windows **installer** (64-bit)



# / Installation Flow

- 下載並安裝Python套件
- 開啟Command視窗，建立虛擬環境
  - `.\python -m venv D:\demo_env\myenv`
- 啟動虛擬環境
  - `.\activate`
- 安裝Spyder IDE(Integrated Development Environment), Jupyter
  - `pip install spyder`
  - `pip install jupyter`
- 安裝pyaedt
  - `pip install pyaedt`

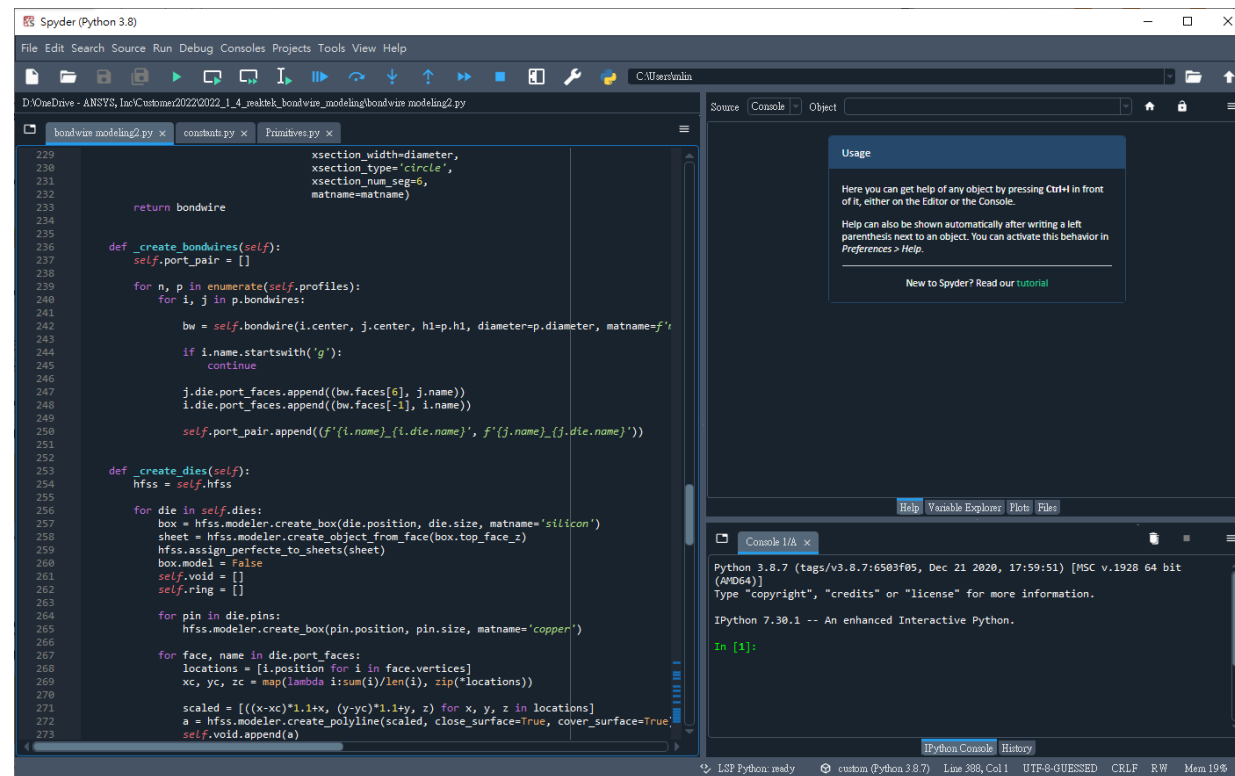


# Installation and Initial Setup

- PyAEDT consolidates and extends all existing capital around scripting for Ansys Electronics Desktop (AEDT), allowing re-use of existing code, sharing of best practices, and collaboration.
- This tool has been tested on HFSS, Icepak, and Maxwell 3D. It also provides basic support for EDB and Circuit (Nexxim).
- In addition to the runtime dependencies listed in the installation information, PyAEDT requires ANSYS EM Suite 2021 R1 or later.

# **Spyder IDE & Jupyter Notebook**

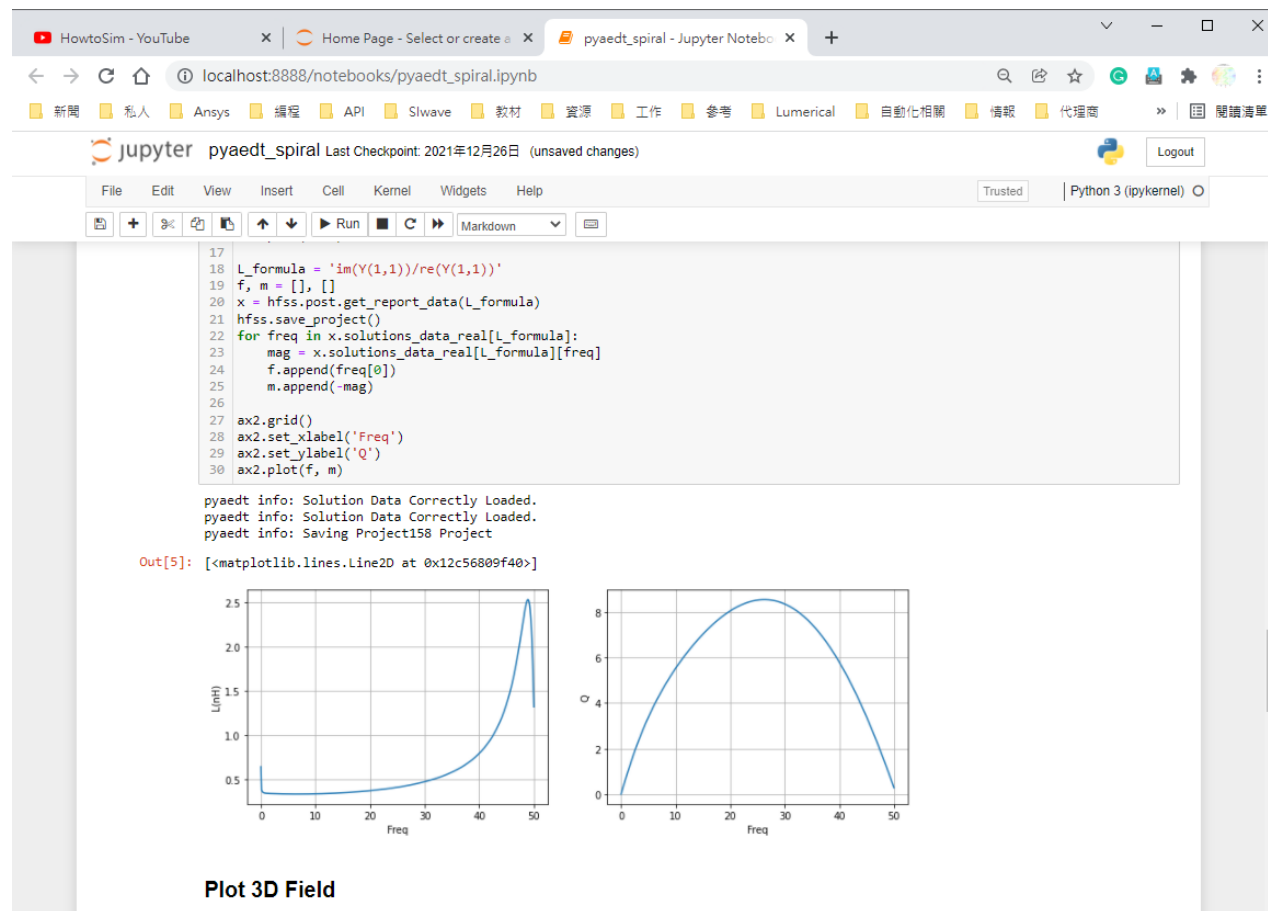
- 編輯器：
  - 具有函式和類檢視器
  - 代碼分析特性
  - 代碼補全
  - 直接跳入定義
- 互動視窗：
  - Python或IPython埠都在工作區可以調整和使用。支援對編輯器里的代碼直接除錯。。
- 文件瀏覽器：
- 在編輯器或埠中顯示任意類或函式呼叫的文件。
- 變數瀏覽視窗
- Matplotlib的圖表顯示視窗
- 歷史記錄





# Jupyter Notebook

- HTML格式混和支援多種資料型態
  - 程式碼
  - 文字，Markdown語法
  - 圖表輸出
- 以Cell為單位
- 可讀性高



# PyAEDT Classes

# Class v.s Object

- Class
  - Define
    - Attributes/Properties
    - methods
- Object
  - Access
    - Attributes
    - methods

# / 類別關係: Is-a v.s. Has-a

- Is-a
  - Inheritance(繼承)
  - Family
  - Module
  - One-way
- Has-a
  - Composition(組成)
  - Team
  - Application
  - Two-way

# Design Operation

# / Validate and Solve

- validate\_simple
  - `Hfss.validate_simple(logfile=None)`
- validate\_full\_design
  - `Hfss.validate_full_design(dname=None, outputdir=None, ports=None)`
- solve\_in\_batch
  - `Hfss.solve_in_batch(filename=None, machine='local', run_in_thread=False)`
- submit\_job
  - `Hfss.submit_job(clustername, aedt_full_exe_path=None, numnodes=1, numcores=32, wait_for_license=True, setting_file=None)`
- analyze\_nominal
  - `Hfss.analyze_nominal(num_cores=None, num_tasks=None, num_gpu=None, acf_file=None)`

# / 初始化與退出

```
from pyaedt import Hfss, constants
hfss = Hfss(specified_version='2021.2')
#%%
.....

#%%
hfss.save_project()
hfss.release_desktop()
```



# 模擬流程常用函式

- 屬性
  - hfss['length'] = '3mm'
- 材料
  - hfss.materials.add\_material()
- 物件
  - hfss.modeler.create\_box()
- 激發元
  - hfss.create\_lumped\_port\_to\_sheet()
- 邊界條件
  - hfss.create\_open\_region()
- 網格
  - hfss.mesh.assign\_initial\_mesh\_from\_slider()
- 模擬設定
  - hfss.create\_setup()
  - hfss.create\_linear\_count\_sweep()
  - hfss.analyze\_all()
- 生成報告
  - hfss.create\_lumped\_port\_to\_sheet()
- 輸出資料
  - hfss.post.get\_report\_data()
- 常數
  - constants.AXIS.X (Y, Z)
  - constants.PLANE.XY, (YZ, ZX)

# Example Code

```
from pyaedt import Hfss, constants
import matplotlib.pyplot as plt

hfss = Hfss(specified_version='2021.2', non_graphical=True)

c1 = hfss.modeler.create_cylinder(constants.AXIS.Z, (0,0,0.5), 1, 20, matname='copper')
c2 = hfss.modeler.create_cylinder(constants.AXIS.Z, (0,0,-0.5), 1, -20, matname='copper')

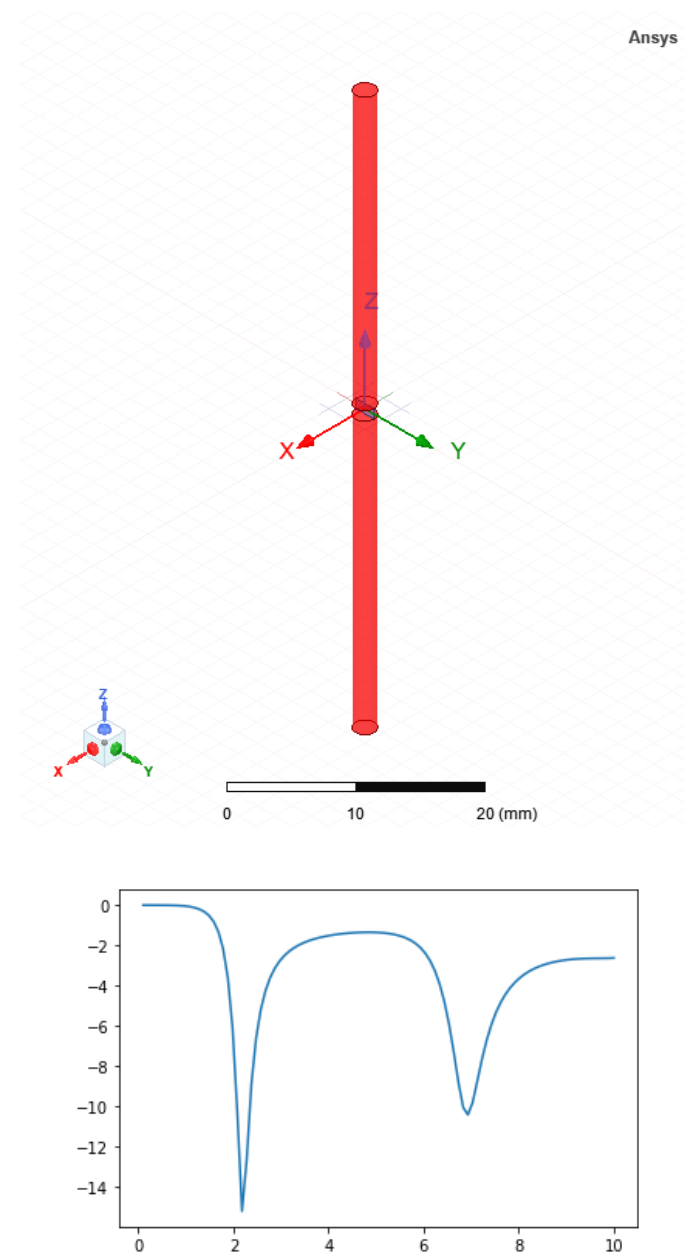
hfss.modeler.create_rectangle(constants.PLANE.YZ, (0, -1, -0.5), (2,1), 's1')
hfss.create_lumped_port_to_sheet('s1', axisdir=constants.AXIS.Z)
hfss.create_open_region(Frequency='1GHz')

setup = hfss.create_setup(setupname='my_setup')
hfss.create_frequency_sweep('my_setup', freqstart=0.1, freqstop=10, num_of_freq_points=101, sweepname='my_sweep')
hfss.analyze_nominal(num_cores=4)
hfss.post.create_rectangular_plot(setup_sweep_name='my_setup : my_sweep')

data = hfss.post.get_report_data('dB(S(1,1))', 'my_setup : my_sweep')

x = data.sweeps['Freq']
y = data.data_real()

plt.plot(x, y)
hfss.release_desktop()
```



# DDR Simulation

```
from pyaedt import Hfss3dLayout, Edb

edb = Edb()
edb.import_cadence_file("d:/demo/Galileo_G87173_204.brd")
edb.core_hfss.create_coax_port_on_component(['U1B5', 'U2A5'], ['M_DQ<0>', 'M_DQ<1>'])

edb.core_components.set_solder_ball('U1B5')
edb.core_components.set_solder_ball('U2A5')

edb.create_cutout(['M_DQ<0>', 'M_DQ<1>'],
                  ['GND'],
                  output_aedb_path='d:/demo/ddr.aedb',
                  open_cutout_at_end=False)

edb.close_edb()

h3d = Hfss3dLayout('d:/demo/ddr.aedb/edb.def')
setup1 = h3d.create_setup('setup1')
setup1.props['Frequency'] = '1GHz'
h3d.create_frequency_sweep('setup1', 'GHz', 0, 1, 11)
setup1.update()

h3d.analyze_all()
h3d.save_project()
```

 **Ansys**

