in Aerospace

by Alex Kenan

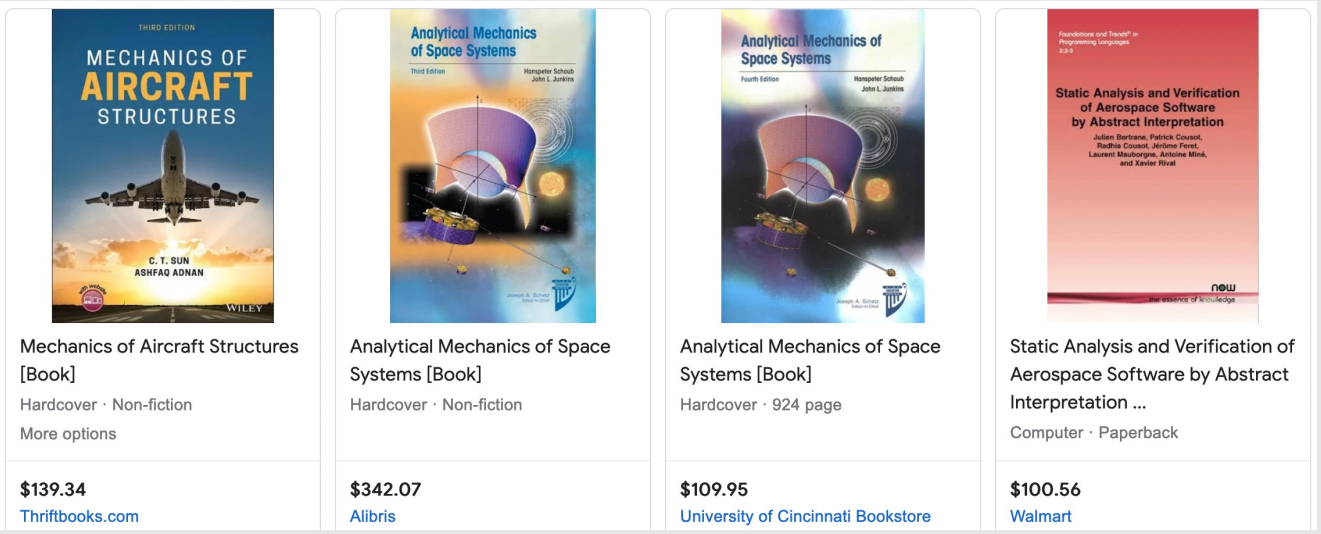
Presented to National Aerospace Center (NLR) on 13 January 2022

Agenda

1. About me & benefits of Python
2. How I use Python
3. How industry can use Python
4. Python libraries to highlight

About Me

- Aerospace Engineer
  - Worked in operations support engineering and propulsion engineering
  - Use Python for data querying, web scraping, modeling, and more
- Author of *Python for Mechanical and Aerospace Engineering*
  - Existing literature on Python for engineering was either 700-page textbooks that were about numerical methods, linear algebra, or matrices, or were 200-page condensed PhD theses



Benefits of Python

- Another tool for the toolbox
  - Often described as “rarely the best tool for the job, but almost always second-best”
- FREE!
- Identical (or better) in capability to MATLAB
- Fast development time with readable code. If speed matters, bottleneck code portions can be written in C for faster runtime
- Open-source nature promotes code sharing
- Large development community
- Other scientific communities are already using Python
  - Astrophysics
  - Chemistry
  - Biology
  - Meteorology

Agenda

1. About me & benefits of Python
2. How I use Python
3. How industry can use Python
4. Python libraries to highlight

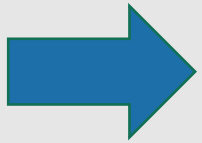
How I've used Python – Reports (Pweave)

```
#' ### Calculations

# Constant altitude range from page 20 of Perf3.pdf
constant_alt_range = (2/ct)*sqrt(2/(rho*S))*CLCDmax1_5*(sqrt(W0)-sqrt(W1))


# cruise climb: R = integral from W1 to W0 ((V/ct)*(L/D)*1/W)dW from page 18 of Perf3.pdf
constant_speed_range = V*CLCDmax/ct*log(W0) - V*CLCDmax/ct*log(W1)

#' ### Output
#+echo=False
print('Constant altitude range is {:.4E} ft or {:.0f} miles'.format(constant_alt_range,
                                                                   floor(constant_alt_range/5280)))
print('Constant speed range is {:.4E} ft or {:.0f} miles'.format(constant_speed_range,
                                                                 constant_speed_range/5280))
print('The pilot should use a cruise climb (constant velocity climb) because it yields a greater range')
```



- Create reports that show calculations and capture output
- Works natively with Python and is "compiled" with one bash command

```
V = 337.562  # ft/s (200 kts)
W0 = 6000  # lb
S = 184  # ft^2
Wf = 500  # lb
W1 = W0 - Wf
K = 0.057
Cd0 = 0.02
CLCDmax1_5 = (3/4)*(1/(3*K*Cd0**3))**(1/4)  # CL^(1/2)/CD MAX
CLCDmax = sqrt(1/(4*Cd0*K))  # CL/CD MAX = L/D MAX
ct = 0.836/3600
rho = 12.67E-4  # slug/ft^3
```

Calculations

```
# Constant altitude range
constant_alt_range =(2/ct)*sqrt(2/(rho*S))*CLCDmax1_5*(sqrt(W0)-sqrt(W1))


# cruise climb: R = integral from W1 to W0 ((V/ct)*(L/D)*1/W)dW
constant_speed_range = V*CLCDmax/ct*log(W0) - V*CLCDmax/ct*log(W1)
```

Output

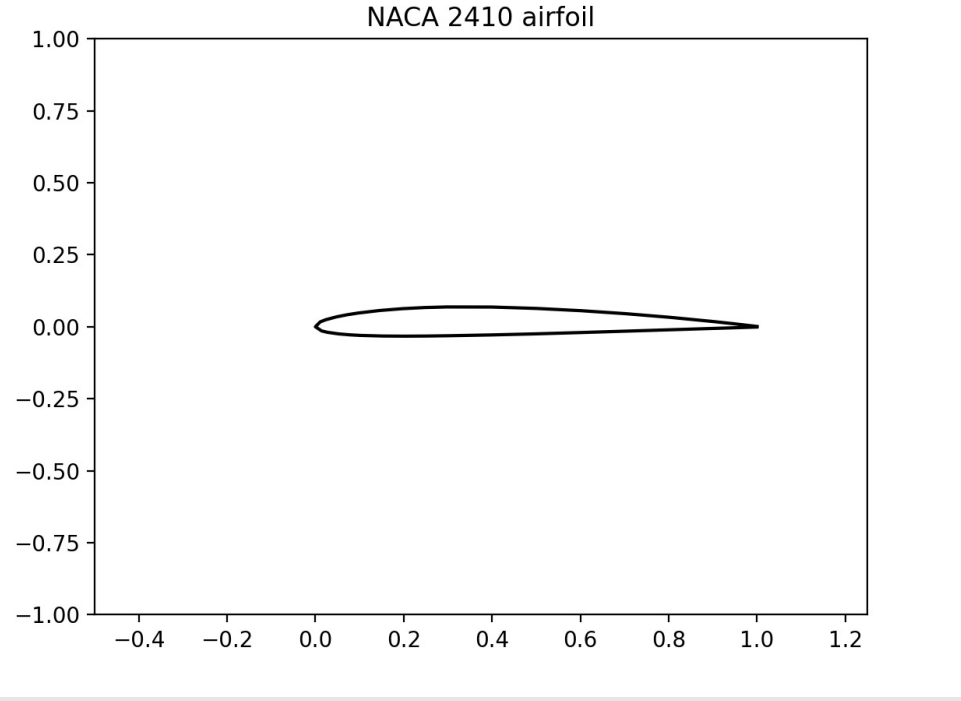
```
Constant altitude range is 1.8243E+06 ft or 345 miles
Constant speed range is 1.8730E+06 ft or 355 miles
The pilot should use a cruise climb (constant velocity climb) because it yields a gr
```

How I've used Python - Plotting

```
try:
    x_values, y_values, title = get_airfoil_coords(air_foil)

    plot_airfoil(x_values, y_values, title)
except NameError:
    print('{} not in UIUC database, try again!'.format(air_foil))
```

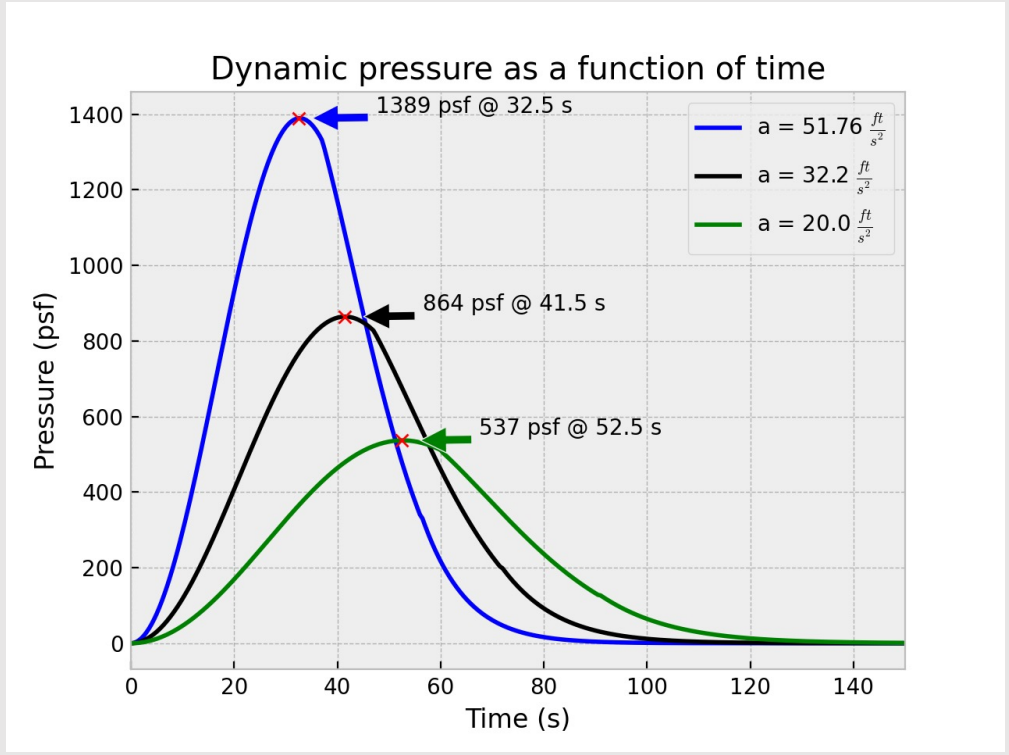
- Get airfoil name by either hard-coding it into the program, or by asking for user input
- Get airfoil coordinates from University of Indiana at Urbana-Champaign (UIUC) website
- Plot airfoil with normalized coordinates



How I've used Python - Modeling

```
plt.style.use('bmh')
y_values = []
x_values = np.arange(0.0, 550.0, 0.5)
for elapsed_time in x_values:
    #
    '''
    Acceleration is the average acceleration
    to go from 0 ft/s to 26,400 ft/s
    (18,000 mph) in 8.5 minutes = 51.764705882 ft/s^2
    '''
    accel = 51.764705882
    alt = altitude(elapsed_time, accel)
    # Dynamic pressure q = 0.5*rho*V^2 = 0.5*density*velocity^2
    q = 0.5 * density(alt) * velocity(elapsed_time, accel) ** 2
    y_values.append(q)

plt.plot(x_values, y_values, 'b-',
         label=r"a = 51.76 $\frac{ft}{s^2}$")
max_val = max(y_values)
ind = y_values.index(max_val)
```



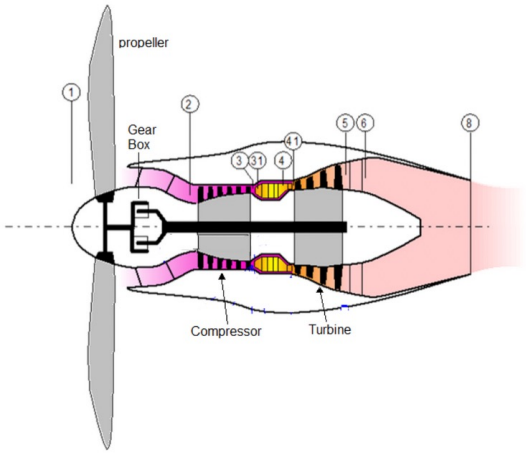
- Model dynamic pressure as a function of time for a rocket launch

Agenda

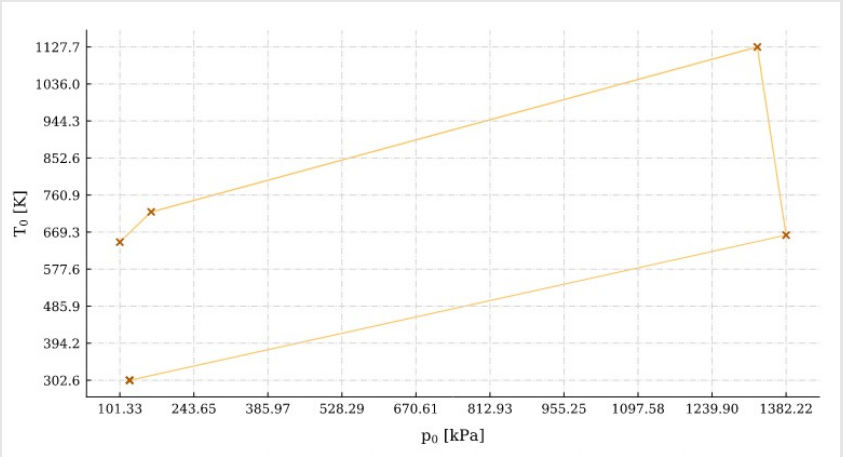
1. About me & benefits of Python
2. How I use Python
3. How industry can use Python
4. Python libraries to highlight

Notable Python Packages - Huracan

Huracan is an open source, 0-dimensional, object-oriented airbreathing engine modelling package for preliminary analysis and design of airbreathing engines, convergence/divergence, and educational purposes. It is capable of modelling engines with an arbitrary number of components connected by an arbitrary number of shafts. It allows for a single combustion chamber per stream, reheating, intercooling and the addition of electrical system power requirements. Multiple-stream systems can be modelled, as well as splitting (such as the bypass flow of a turbofan) and mixing streams (such as in the nozzle of a mixed exhaust turbofan). Among the outputs are T-S plots, p-V plots, H-p plots, and T-p plots for your specific configuration.



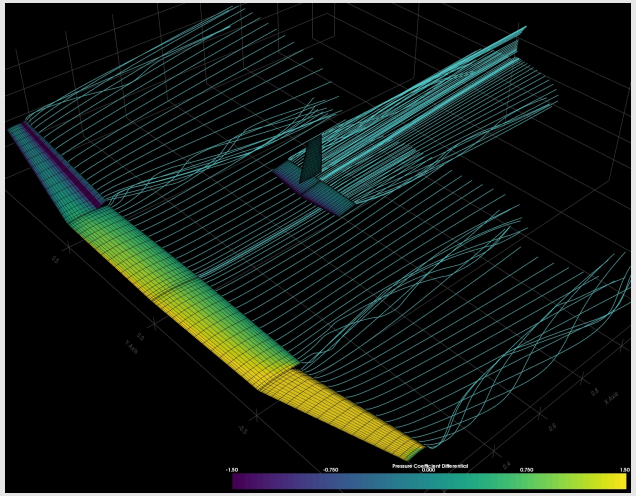

```
0.pr    Prop
          T0 = 302.558          [K]
          p0 = 120,192.996      [Pa]
0.il    Inlet
          T0 = 302.558          [K]
          p0 = 120,192.996      [Pa]
0.cp    Compressor
          T0 = 661.837          [K]
          p0 = 1,382,219.449    [Pa]
0.cc    Combustion chamber
          T0 = 1,130.000        [K]
          p0 = 1,326,930.671    [Pa]
0.tb    Turbine
          T0 = 721.509          [K]
          p0 = 162,412.892      [Pa]
0.nz    Nozzle
          T0 = 645.787          [K]
          p0 = 101,325.000      [Pa]
```



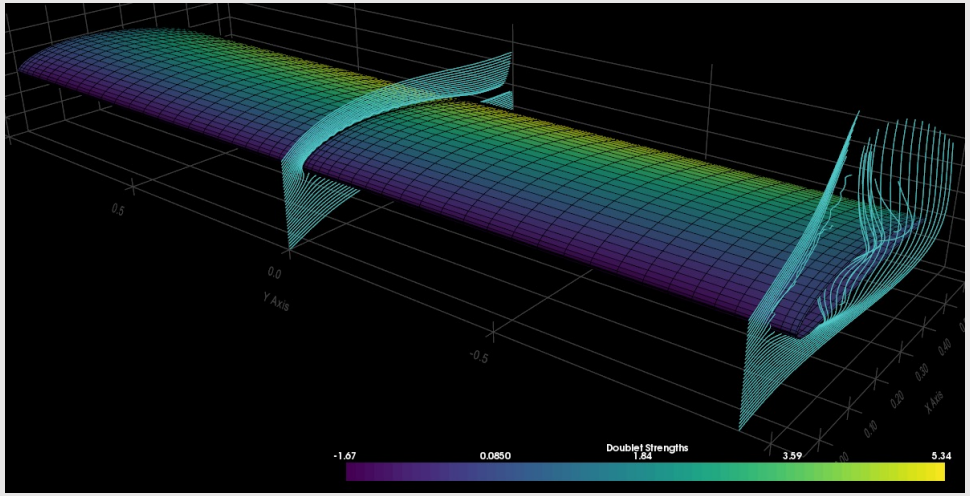
The inspiration for the project lies in traditional thermodynamic plant diagrams, and similar architectures are used in well known proprietary tools such as GasTurb and NLR's GSP.

Notable Python Packages - AeroSandbox

AeroSandbox is an optimization suite that combines the ease-of-use of familiar NumPy syntax with the power of modern automatic differentiation. This automatic differentiation dramatically improves optimization performance on large problems: design problems with tens of thousands of decision variables can be solved in seconds on a laptop. AeroSandbox contains dozens of end-to-end-differentiable aerospace physics models, allowing you to simultaneously optimize an aircraft's aerodynamics, structures, propulsion, mission trajectory, stability, and more.



VLM simulation of a glider, aileron deflections of +/- 30°. Runtime of 0.35 sec on a typical laptop (i7-8750H).

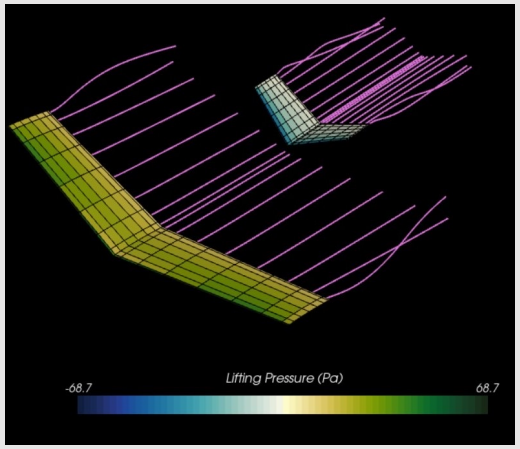


Panel simulation of a wing (extruded NACA2412, α=15°, AR=4). Note the strong three-dimensionality of the flow near the tip.

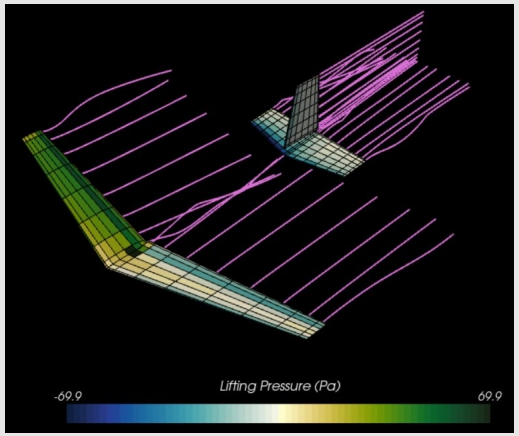
Notable Python Packages – Ptera Software

Ptera is an extension of AeroSandbox developed specifically for flapping wing flight. It currently supports three solvers: a steady horseshoe vortex lattice method (VLM), a steady ring VLM, and an unsteady ring VLM (UVLM). The software produces visualization outputs as well as lift/drag vs time and roll/pitch/yaw moments vs time.

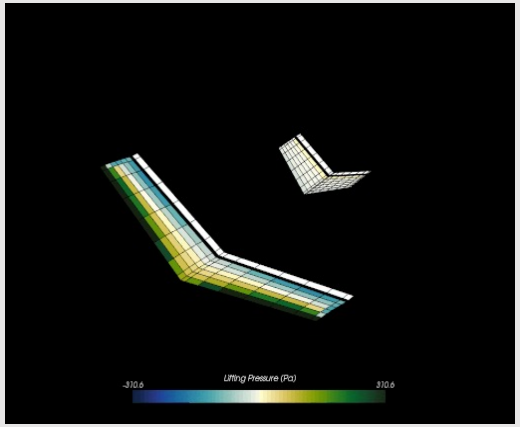
Horseshoe VLM



Steady Ring VLM



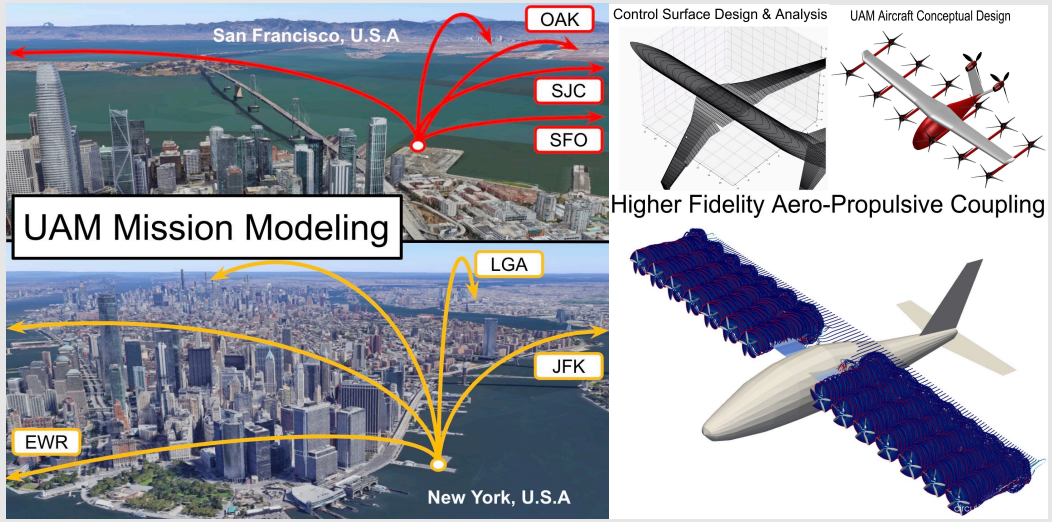
Unsteady Ring VLM



This is a GIF in PowerPoint. Click here to view

Notable Python Packages - SUAVE

Suave is a fully functioning, conceptual aircraft design tool for designing various parts of aerospace vehicles and missions.

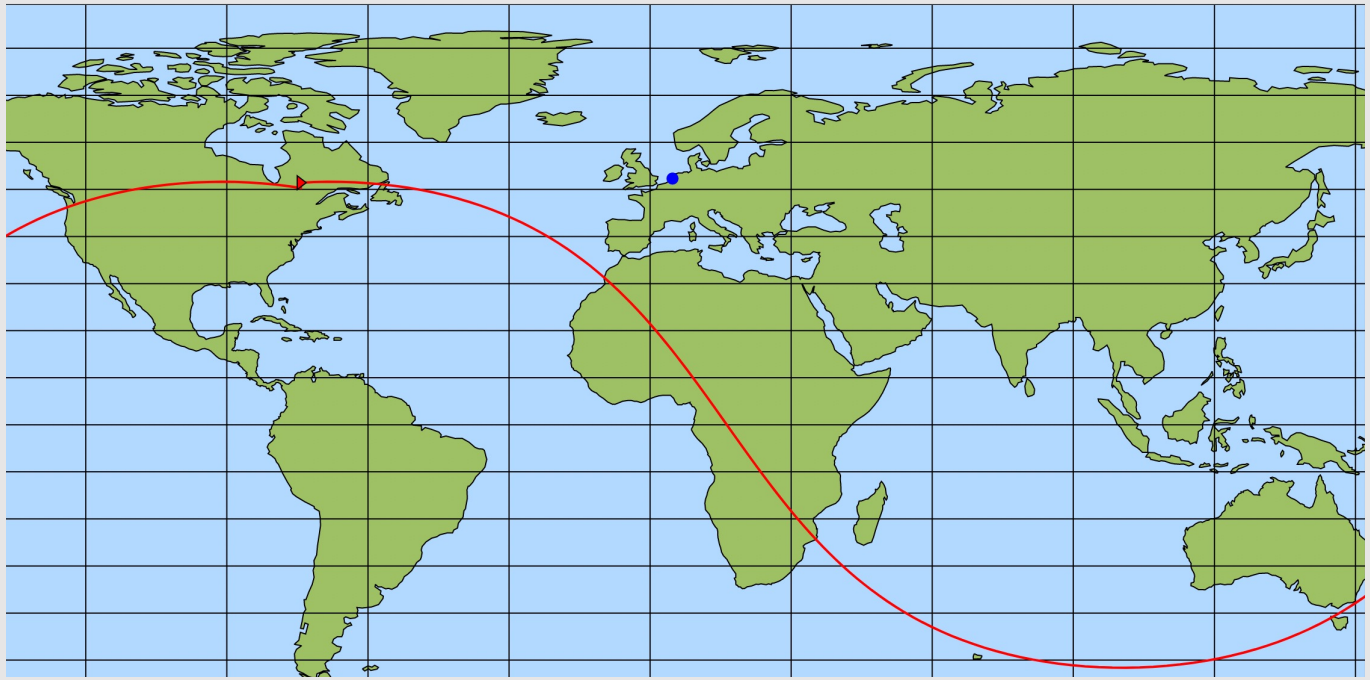


- Segment-based mission design (taxi, takeoff, climb, cruise, descent, landing, etc.)
- Max takeoff weight (MTOW) sizing for tube-and-wing, blended-wing body, and human-powered aircraft
- Subsonic and supersonic aerodynamic modeling
- Static and dynamic stability modeling
- Aircraft performance estimates (MTOW, takeoff length, range, endurance, etc.)
- Propulsion modeling for turbines and electrically ducted fans
- Energy networks for battery, fuel-cell, and solar-panel based vehicles
- External observer modeling for airframe and engine noise contribution
- Import and export capabilities to third party software like OpenVSP

SUAVE Optimization Paper

Notable Python Packages - Poliastro

Poliastro is a library for astrodynamics and orbital mechanics (and it handles quantities with units!)



A select sample of its features:

- Trajectory plotting
- Analytical and numerical orbit propagation
- Hohmann and bielliptic maneuvers computation
- Initial orbit determination (Lambert problem)
- Planetary ephemerides (using SPICE kernels via Astropy library)
- Computation of Near-Earth Objects (NEOs)
- Coordinate frame transformations

Poliastro uses Plotly to create visualizations, which renders in a web browser, further enhancing interactivity

Others

- Libraries to connect with simulation software like X-Plane 11, Microsoft Flight Simulator (FSX/MSFS 2020), Prepar3d, etc.
- Turbulence detection algorithms
- Basic computational fluid dynamics (CFD) models
- Orbital mechanics tutorials
- scikit-aero - Aeronautical engineering calculations in Python

Agenda

1. About me & benefits of Python
2. How I use Python
3. How industry can use Python
4. Python libraries to highlight

Important Python Libraries

You might have to create your own Python library for the work that you do. Here are the major Python libraries in several different areas of engineering and computer science that can be used as the building blocks for a custom program or library:

- Advanced math: NumPy, SciPy, SymPy
- Data visualization: Matplotlib, Seaborn, Plotly (and many others)
- Data processing: Pandas, openpyxl, csv
- Web scraping: Requests, BeautifulSoup4
- Creating standalone applications: py2exe, py2app, cx_freeze
- Creating user interfaces (GUIs): Tkinter, PyQt5, wxPython, Kivy (and many others)
- Creating websites: Flask, Django, FastAPI, web2py
- Machine learning: TensorFlow, scikit-learn, Keras, PyTorch
- Reports: Pweave