
ezaero Documentation

Release 0.1.dev0

Pedro Arturo Morales Maries

Oct 09, 2020

CONTENTS:

1	Steady VLM module	3
2	Examples	7
2.1	Simple steady VLM demo	7
2.2	Dihedral angle effect	9
	Python Module Index	13
	Index	15

Under construction

ezaero (easy-aero) is an open source Python package oriented to implement numerical methods for Aerodynamics, such as the Vortex lattice Method for lifting surfaces.

STEADY VLM MODULE

The `ezaero.vlm.steady` module includes a Vortex Lattice Method implementation for lifting surfaces.

References

class `ezaero.vlm.steady.FlightConditions` (*ui: float = 100, aoa: float = 3.141592653589793, rho: float = 1*)

Container for the flight conditions.

ui

Free-stream flow velocity.

Type `float`

angle_of_attack

Angle of attack of the wing, expressed in radians.

Type `float`

rho

Free-stream flow density.

Type `float`

class `ezaero.vlm.steady.MeshParameters` (*m: int = 4, n: int = 16*)

Container for the wing mesh parameters.

m

Number of chordwise panels.

Type `int`

n

Number of spanwise panels.

Type `int`

class `ezaero.vlm.steady.Simulation` (*wing: ezaero.vlm.steady.WingParameters, mesh: ezaero.vlm.steady.MeshParameters, flight_conditions: ezaero.vlm.steady.FlightConditions*)

Simulation runner.

wing

Wing geometry definition.

Type `WingParameters`

mesh

Mesh specification for the wing.

Type *MeshParameters*

flight_conditions

Flight conditions for the simulation.

Type *FlightConditions*

plot_cl ()

Plot lift coefficient distribution on the wing.

plot_wing (**kwargs)

Generate 3D plot of wing panels, vortex panels, and panel control points.

run ()

Run end-to-end steady VLM simulation.

Returns Object containing the results of the steady VLM simulation.

Return type *SimulationResults*

```
class ezaero.vlm.steady.SimulationResults (dp: numpy.ndarray, dL: numpy.ndarray,  
                                           cl: numpy.ndarray, cl_wing: float, cl_span:  
                                           numpy.ndarray)
```

Container for the resulting distributions from the steady VLM simulation.

dp

Distribution of pressure difference between lower and upper surfaces.

Type *np.ndarray*, shape (m, n)

dL

Lift distribution.

Type *np.ndarray*, shape (m, n)

cl

Lift coefficient distribution.

Type *np.ndarray*, shape (m, n)

cl_wing

Wing lift coefficient.

Type *float*

cl_span

Spanwise lift coefficient distribution.

Type *np.ndarray*, shape (n,)

```
class ezaero.vlm.steady.WingParameters (root_chord: float = 1, tip_chord: float = 1, plan-  
                                         form_wingspan: float = 4, sweep_angle: float = 0,  
                                         dihedral_angle: float = 0)
```

Container for the geometric parameters of the wing.

root_chord

Chord at root of the wing.

Type *float*

tip_chord

Chord at tip of the wing.

Type *float*

planform_wingspan

Wingspan of the planform.

Type `float`

sweep_angle

Sweep angle of the 1/4 chord line, expressed in radians.

Type `float`

dihedral_angle

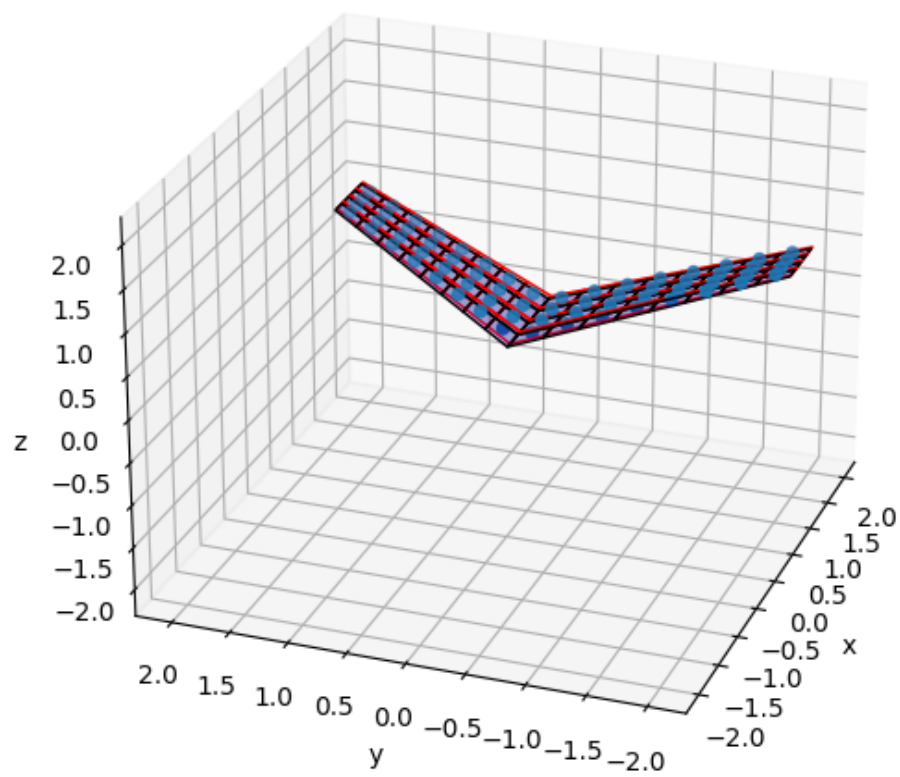
Dihedral angle, expressed in radians.

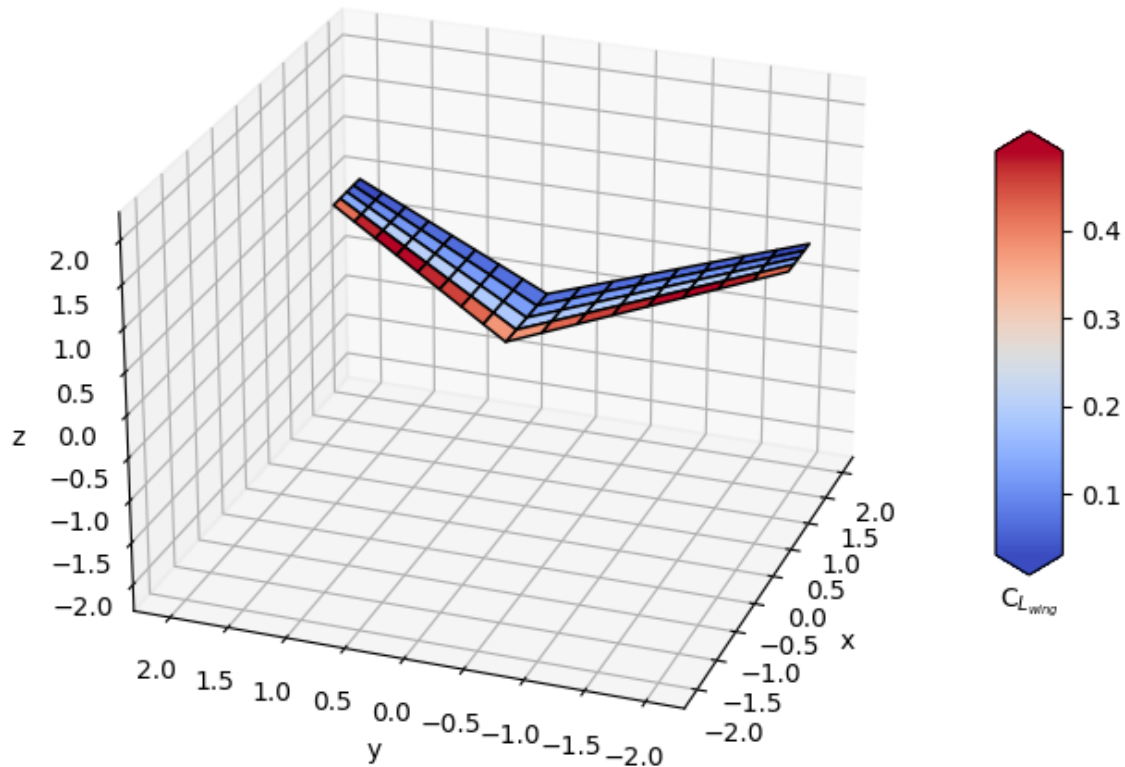
Type `float`

EXAMPLES

2.1 Simple steady VLM demo

Minimal example of simulation execution.





Out:

```
Wing lift coefficient: 0.20335518605804598
Elapsed time: 0.010579347610473633 s
```

```
import time

import matplotlib.pyplot as plt
import numpy as np

import ezaero.vlm.steady as vlm

# definition of wing, mesh and flight condition parameters
wing = vlm.WingParameters(
    root_chord=1,
    tip_chord=0.6,
    planform_wingspan=4,
    sweep_angle=30 * np.pi / 180,
    dihedral_angle=15 * np.pi / 180,
)
mesh = vlm.MeshParameters(m=4, n=16)
flcond = vlm.FlightConditions(ui=100, aoa=3 * np.pi / 180, rho=1.0)
```

(continues on next page)

(continued from previous page)

```
sim = vlm.Simulation(wing=wing, mesh=mesh, flight_conditions=flcond)

start = time.time()
res = sim.run()
print(f"Wing lift coefficient: {res.cl_wing}")
print(f"Elapsed time: {time.time() - start} s")

# plot wing panels, vortex panels, and collocation points
sim.plot_wing()
plt.show()

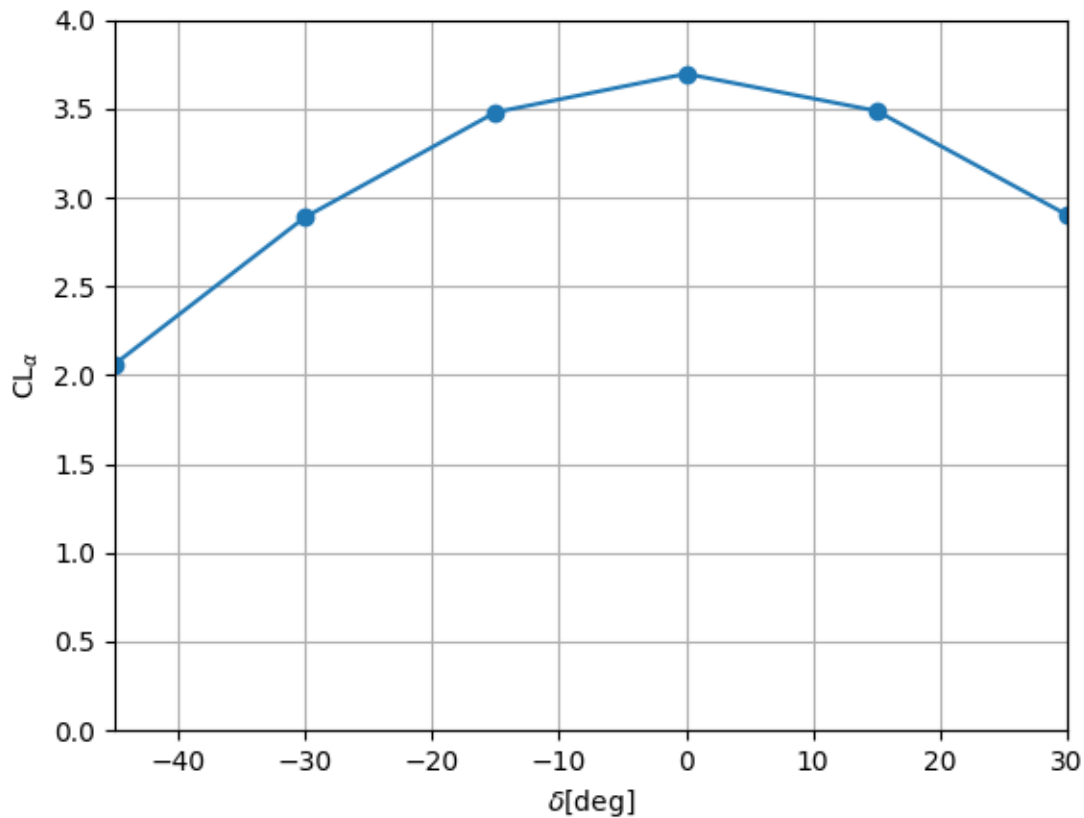
# plot cl distribution on wing
sim.plot_cl()
plt.show()
```

Total running time of the script: (0 minutes 0.891 seconds)

2.2 Dihedral angle effect

Effect of dihedral on the lift coefficient slope of rectangular wings.

2.2.1 References



Out:

```
Elapsed time: 1.1781010627746582 s
```

```
import time

import matplotlib.pyplot as plt
import numpy as np

import ezaero.vlm.steady as vlm

start = time.time()

# dihedral angles grid
deltas = np.array([-45, -30, -15, 0, 15, 30]) * np.pi / 180

# define mesh parameters and flight conditions
mesh = vlm.MeshParameters(m=8, n=30)
```

(continues on next page)

(continued from previous page)

```

# slope for each dihedral calculated using two flight conditions
flcond_0 = vlm.FlightConditions(ui=100.0, aoa=0.0, rho=1.0)
flcond_1 = vlm.FlightConditions(ui=100.0, aoa=np.pi / 180, rho=1.0)
cla_list = [] # container for the lift coefficient slope
for delta in deltas:
    # The figure in the book uses an aspect ratio of 4. It does not
    # correspond to the planform, but the "real" wingspan, hence we project
    # the wingspan with the dihedral angle
    bp = 4 * np.cos(delta)
    # define rectangular wing (same cr and ct), with no sweep (theta).
    wing = vlm.WingParameters(
        root_chord=1.0,
        tip_chord=1.0,
        planform_wingspan=bp,
        sweep_angle=0,
        dihedral_angle=delta,
    )
    res_0 = vlm.Simulation(wing=wing, mesh=mesh, flight_conditions=flcond_0).run()
    res_1 = vlm.Simulation(wing=wing, mesh=mesh, flight_conditions=flcond_1).run()
    d_cl = res_1.cl_wing - res_0.cl_wing
    d_alpha = flcond_1.aoa - flcond_0.aoa
    slope = d_cl / d_alpha * np.cos(delta) # project load
    cla_list.append(slope)

end = time.time()
elapsed = end - start

print("Elapsed time: {} s".format(elapsed))

fig = plt.figure()
plt.plot(deltas * 180 / np.pi, cla_list, "o-")
plt.xlabel(r"$\delta$[deg]")
plt.ylabel(r"$C_{L\delta}$")
plt.ylim(0, 4)
plt.grid()
plt.xlim(deltas.min() * 180 / np.pi, deltas.max() * 180 / np.pi)
plt.show()

```

Total running time of the script: (0 minutes 1.306 seconds)

PYTHON MODULE INDEX

e

`ezaero.vlm.steady`, 3

A

`angle_of_attack` (*ezaero.vlm.steady.FlightConditions* attribute), 3

C

`cl` (*ezaero.vlm.steady.SimulationResults* attribute), 4

`cl_span` (*ezaero.vlm.steady.SimulationResults* attribute), 4

`cl_wing` (*ezaero.vlm.steady.SimulationResults* attribute), 4

D

`dihedral_angle` (*ezaero.vlm.steady.WingParameters* attribute), 5

`dL` (*ezaero.vlm.steady.SimulationResults* attribute), 4

`dp` (*ezaero.vlm.steady.SimulationResults* attribute), 4

E

`ezaero.vlm.steady`
module, 3

F

`flight_conditions` (*ezaero.vlm.steady.Simulation* attribute), 4

`FlightConditions` (class in *ezaero.vlm.steady*), 3

M

`m` (*ezaero.vlm.steady.MeshParameters* attribute), 3

`mesh` (*ezaero.vlm.steady.Simulation* attribute), 3

`MeshParameters` (class in *ezaero.vlm.steady*), 3

module

`ezaero.vlm.steady`, 3

N

`n` (*ezaero.vlm.steady.MeshParameters* attribute), 3

P

`planform_wingspan`
(*ezaero.vlm.steady.WingParameters* attribute),
4

`plot_cl` (*ezaero.vlm.steady.Simulation* method), 4

`plot_wing` (*ezaero.vlm.steady.Simulation* method),
4

R

`rho` (*ezaero.vlm.steady.FlightConditions* attribute), 3

`root_chord` (*ezaero.vlm.steady.WingParameters* attribute), 4

`run` (*ezaero.vlm.steady.Simulation* method), 4

S

`Simulation` (class in *ezaero.vlm.steady*), 3

`SimulationResults` (class in *ezaero.vlm.steady*), 4

`sweep_angle` (*ezaero.vlm.steady.WingParameters* attribute), 5

T

`tip_chord` (*ezaero.vlm.steady.WingParameters* attribute), 4

U

`ui` (*ezaero.vlm.steady.FlightConditions* attribute), 3

W

`wing` (*ezaero.vlm.steady.Simulation* attribute), 3

`WingParameters` (class in *ezaero.vlm.steady*), 4