Sweep cambered main sail

Zuzanna Wieczorek, Grzegorz Gruszczyński

Jun 09, 2023

CONTENTS

```python
# varaibles.py for jupyter
import os
import numpy as np
import time

half_wing_span = 8
sweep_angle_deg = 5.
chord_length = 4
AoA_deg = 8.
mgirths =  np.array([0.00, 1./8, 1./4, 1./2, 3./4, 7./8, 1.00])
mchords = np.array([chord_length]* len(mgirths))


output_args = {
    'case_name': 'my_case_name',  # get name of the current file
    'case_dir': os.path.abspath(''), # get dir of the current file
    'name': os.path.join("results_example_jib_and_mainsail_vlm",␣
↪time.strftime("%Y-%m-%d_%Hh%Mm%Ss")),
    'file_name': 'my_fancy_results', # name of xlsx excel file
}

solver_args = {
    'n_spanwise':  5,  # No of control points (above the water) per␣
↪sail, recommended: 50
    'n_chordwise': 5, # No of control points (above the water) per␣
↪sail, recommended: 50
    'interpolation_type': "linear",  # either "spline" or "linear"
    'LLT_twist': "real_twist",  # defines how the Lifting Line␣
↪discretize the sail twist.
}

conditions_args = {
    'leeway_deg': 0.,    # [deg]
    'heel_deg': 0.,      # [deg]
    'SOG_yacht': 1.,   # [m/s] yacht speed - speed over ground␣
↪(leeway is a separate variable)
    'tws_ref': 1.0,     # [m/s] true wind speed
    'alpha_true_wind_deg': AoA_deg,   # [deg] true wind angle (with␣
↪reference to course over ground) => Course Wind Angle to the boat␣
↪track = true wind angle to centerline + Leeway
    'reference_water_level_for_wind_profile': -0.,  # [m] this is␣
↪an attempt to mimick the deck effect
    # by lowering the sheer_above_waterline
    # while keeping the wind profile as in original geometry
    # this shall be negative (H = sail_ctrl_point - water_level)
    'wind_exp_coeff': 0.,  # [-] coefficient to determine the␣
↪exponential wind profile
    'wind_reference_measurment_height': 10.,  # [m] reference␣
↪height for exponential wind profile
```

(continues on next page)

(continued from previous page)

```
    'rho': 1.,  # air density [kg/m3]
    'wind_profile': 'flat', # allowed: 'exponential' or 'flat' or
↪'logarithmic'
    'roughness': 0.05, # for logarithmic profile only
}


rig_args = {
    'main_sail_luff': half_wing_span / np.cos(np.deg2rad(sweep_
↪angle_deg)),  # [m]
    'jib_luff': 10.0,  # [m]
    'foretriangle_height': 11.50,  # [m]
    'foretriangle_base': 3.90,  # [m]
    'sheer_above_waterline': 1.2,#[m]
    'boom_above_sheer': 1.3, # [m],
    'rake_deg': 90. + sweep_angle_deg,  # rake angle [deg]
    'mast_LOA': 0.,  # [m]
    'sails_def': 'main', # definition of sail set, possible: 'jib'␣
↪or 'main' or 'jib_and_main'
}


# INFO for camber:
# First digit describing maximum camber as percentage of the chord.
# Second digit describing the distance of maximum camber from the␣
↪airfoil leading edge in tenths of the chord.
main_sail_args = {
    'girths' : mgirths,
    'chords': mchords,
    'centerline_twist_deg': 0*mgirths,
    'camber': 10*np.array([0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.
↪01]),
    'camber_distance_from_luff': np.array([0.5, 0.5, 0.5, 0.5, 0.5,␣
↪0.5, 0.5]),
}


jgirths = np.array([0.00, 1./4, 1./2, 3./4, 1.00])
jib_sail_args = {
    'centerline_twist_deg': 0*(10+5)  + 0*15. * jgirths,
    'girths': jgirths,
    'chords': 0* np.array([3.80, 2.98, 2.15, 1.33, 0.5]),
    'camber': 0*np.array([0.01, 0.01, 0.01, 0.01, 0.01]),
    'camber_distance_from_luff': np.array([0.5, 0.5, 0.5, 0.5, 0.
↪5]), # starting from leading edge
}

# REFERENCE CSYS
# The origin of the default CSYS is located @ waterline level and␣
↪aft face of the mast
```

(continues on next page)

(continued from previous page)

```
# The positive x-coord: towards stern
# The positive y-coord: towards leeward side
# The positive z-coord: above the water
# To shift the default CSYS, adjust the 'reference_level_for_moments
↪' variable.
# Shifted CSYS = original + reference_level_for_moments
# As a results the moments will be calculated around the new origin.

# yaw_reference [m] - distance from the aft of the mast towards␣
↪stern, at which the yawing moment is calculated.
# sway_reference [m] - distance from the aft of the mast towards␣
↪leeward side. 0 for symmetric yachts ;)
# heeling_reference [m] - distance from the water level,  at which␣
↪the heeling moment is calculated.
csys_args = {
    'reference_level_for_moments': np.array([0, 0, 0]),  # [yaw_
↪reference, sway_reference, heeling_reference]
}

# GEOMETRY OF THE KEEL
# to estimate heeling moment from keel, does not influence the␣
↪optimizer.
# reminder: the z coord shall be negative (under the water)
keel_args={
    'center_of_lateral_resistance_upright': np.array([0, 0, -1.0]),␣
↪ # [m] the coordinates for a yacht standing in upright position
}
```

```
import shutil
from pySailingVLM.rotations.csys_transformations import CSYS_
↪transformations
from pySailingVLM.yacht_geometry.hull_geometry import HullGeometry
from pySailingVLM.results.save_utils import save_results_to_file
from pySailingVLM.solver.panels_plotter import display_panels_xyz_
↪and_winds
from pySailingVLM.results.inviscid_flow import InviscidFlowResults
from pySailingVLM.solver.vlm import Vlm
from pySailingVLM.runner.sail import Wind, Sail
from pySailingVLM.runner.container import Output, Rig, Conditions,␣
↪Solver, MainSail, JibSail, Csys, Keel

from pySailingVLM.solver.panels_plotter import plot_cp
```

```
import numpy as np
from pySailingVLM.solver.coefs import get_vlm_Cxyz

C_results = []
a_vlm_results = []
```

(continues on next page)

(continued from previous page)

```
out = Output(**output_args)
conditions = Conditions(**conditions_args)
solver = Solver(**solver_args)
main = MainSail(**main_sail_args)
jib = JibSail(**jib_sail_args)
csys = Csys(**csys_args)
keel = Keel(**keel_args)
rig = Rig(**rig_args)
csys_transformations = CSYS_transformations(conditions.heel_deg,␣
↪conditions.leeway_deg, v_from_original_xyz_2_reference_csys_
↪xyz=csys.reference_level_for_moments)


w = Wind(conditions)
s = Sail(solver, rig, main, jib, csys_transformations)
sail_set = s.sail_set
myvlm = Vlm(sail_set.panels, solver.n_chordwise, solver.n_spanwise,␣
↪conditions.rho, w.profile, sail_set.trailing_edge_info, sail_set.
↪leading_edge_info)

height = 1.0
sails_Cxyz = myvlm.get_Cxyz(w, height)

# enumerate through sails
# in this example we have only main
print(f"Cxyz for {rig.sails_def}")
for idx, c in enumerate(sails_Cxyz):
    print(f"C[{idx}]: {c}")

hull = HullGeometry(rig.sheer_above_waterline, rig.foretriangle_
↪base, csys_transformations, keel.center_of_lateral_resistance_
↪upright)
inviscid_flow_results = InviscidFlowResults(sail_set, csys_
↪transformations, myvlm)
inviscid_flow_results.estimate_heeling_moment_from_keel(hull.center_
↪of_lateral_resistance)
```

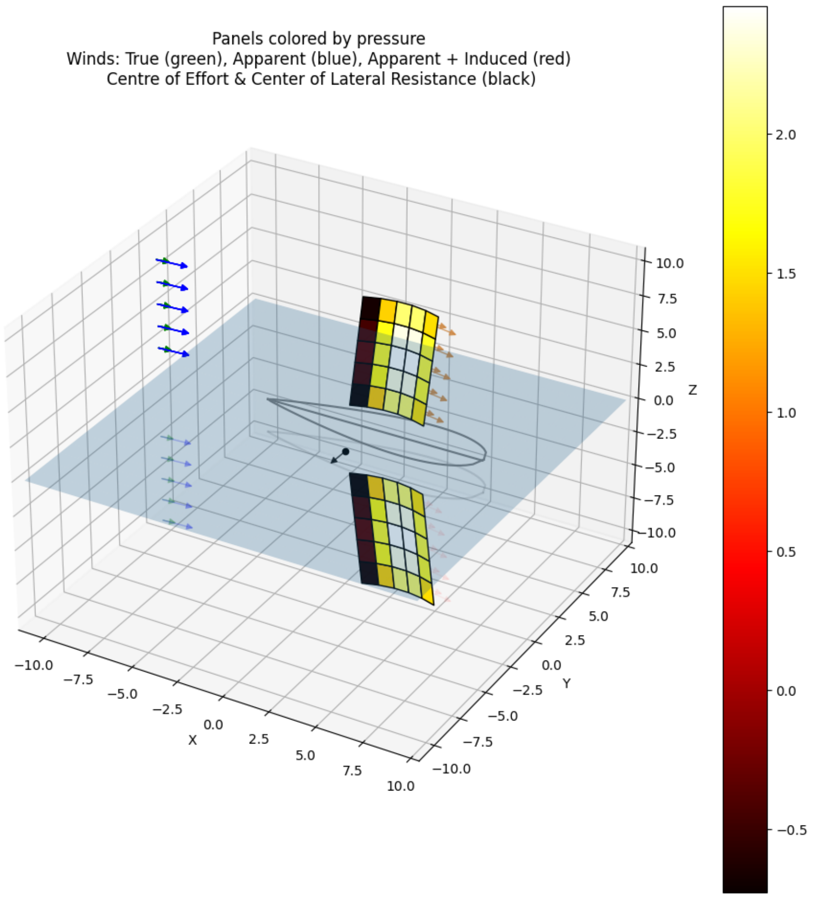
```
Cxyz for main
C[0]: [ 0.06028863  2.83686792 -0.00527457]
```

```
%matplotlib widget
print("Preparing visualization.")
display_panels_xyz_and_winds(myvlm, inviscid_flow_results, myvlm.
↪inlet_conditions, hull, show_plot=True, show_apparent_induced_
↪wind=True) # add show_apparent_induced_wind=True for apparent +␣
↪induced wind
df_components, df_integrals, df_inlet_IC = save_results_to_
↪file(myvlm, csys_transformations, inviscid_flow_results, s.sail_
↪set, out.name, out.file_name)
```

(continues on next page)

(continued from previous page)

```
Preparing visualization.
```




```
print(f"-----------------------------------------------------------------
↪-")
print(f"Notice:\n"
  f"\tThe forces [N] and moments [Nm] are without profile drag.\n"
  f"\tThe the _COG_ CSYS is aligned in the direction of the yacht␣
↪movement (course over ground).\n"
  f"\tThe the _COW_ CSYS is aligned along the centerline of the␣
↪yacht (course over water).\n"
  f"\tNumber of panels (sail s.sail_set with mirror): {s.sail_set.
↪panels.shape}")

df_integrals
```

```
----------------------------------------------------------------
Notice:
        The forces [N] and moments [Nm] are without profile drag.
        The the _COG_ CSYS is aligned in the direction of the␣
↪yacht movement (course over ground).
        The the _COW_ CSYS is aligned along the centerline of␣
↪the yacht (course over water).
        Number of panels (sail s.sail_set with mirror): (50, 4,␣
↪3)
```

```
                              Quantity       Value
0               F_main_sail_total_COG.x    0.988981
1               F_main_sail_total_COG.y   46.536280
2               F_main_sail_total_COG.z   -0.086525
3                   F_sails_total_COG.x    0.988981
4                   F_sails_total_COG.y   46.536280
5                   F_sails_total_COG.z   -0.086525
6                   F_sails_total_COW.x    0.988981
7                   F_sails_total_COW.y   46.536280
8                   F_sails_total_COW.z   -0.086525
9                    M_keel_total_COG.x  -46.536280
10                   M_keel_total_COG.y    0.988981
11                   M_keel_total_COG.z    0.000000
12            M_keel_total_COW.x (heel)  -46.536280
13           M_keel_total_COW.y (pitch)    0.988981
14    M_keel_total_COW.z (yaw - JG sign)   -0.000000
15             M_keel_total_COW.z (yaw)    0.000000
16              M_main_sail_total_COG.x -302.514742
17              M_main_sail_total_COG.y    6.902970
18              M_main_sail_total_COG.z  137.446498
19                  M_sails_total_COG.x -302.514742
20                  M_sails_total_COG.y    6.902970
21                  M_sails_total_COG.z  137.446498
22           M_sails_total_COW.x (heel) -302.514742
23          M_sails_total_COW.y (pitch)    6.902970
24   M_sails_total_COW.z (yaw - JG sign) -137.446498
25            M_sails_total_COW.z (yaw)  137.446498
26                        M_total_COG.x -349.051021
27                        M_total_COG.y    7.891951
28                        M_total_COG.z  137.446498
29                 M_total_COW.x (heel) -349.051021
30                M_total_COW.y (pitch)    7.891951
31        M_total_COW.z (yaw - JG sign) -137.446498
32                 M_total_COW.z (yaw)  137.446498
```

```
plot_cp(sail_set.zero_mesh, myvlm.p_coeffs, out.name)
```