Aerodynamics of Wings and Bodies

Homework 5

Lior Isakov (200928372)

Eden Shazar (204378608)

Fresh start

```
clc
close all
clear variables
```

Parameters

```
% user input
N = 300;      % number of tiles used per side of wing for VLM
% environment
T = 288;      % [K] - air temperature (chosen as ISA temperature at SL)
alpha = 0;    % [rad] - angle of attack
% missile
d = 1;        % [m] - missile body diameter (arbitrary)

% givens
M = 0.6;        % [-] - mach number
gamma = 1.4;    % [-] - specific heat ratio for air
R = 286;        % [m^2/(s^2*K)] - gas constant for air

% calculated parameters
% environment
a = sqrt(gamma*R*T);     % [m/s] - speed of sound
uoo = M*a;               % [m/s] - uniform flow speed
beta = sqrt(1 - M^2);    % [-]
% missile
r = d/2;                        % [m] - body radius
lN = 2.5*d;                     % [m] - length of nosecone
rN = d/2;                       % [m] - body radius at interface with nosecone
lW = 5.62*d;                    % [m] - lengthwise position of wing root chord
lT = lW + 1.56*d + 3.95*d;      % [m] - lengthwise position of wing root chord
RN = (lN^2 + rN^2)/(2*rN);      % [m] - radius of side of nosecone
VN = pi * ...                   % [m^3] - volume of nosecone
    integral(@(x) (sqrt(RN^2 - x.^2) - ...
                  (RN - rN)).^2, -lN, 0);
% wing
wingChordLeadingEdges = [0, 0;
                         1.56*d, 1.06*d];
wingChordLengths = [1.56*d, 1e-6];
% tail
```

```
tailChordLeadingEdges = [0, 0;
                         1.13*d, 0.75*d];
tailChordLengths = [1.38*d, 0.25*d];
```

Wing and tail

```
% wing and tail VLM definition
wing = FlatWing(wingChordLeadingEdges, wingChordLengths, N);
tail = FlatWing(tailChordLeadingEdges, tailChordLengths, N);

% VLM vortex calculations
tempAlpha = deg2rad(6);    % for lift line slope calculation only
wing.CalculateGammaValues(uoo, tempAlpha);
tail.CalculateGammaValues(uoo, tempAlpha);

% lift coefficients
cLWing = wing.AeroCoeffs(uoo);
cLTail = tail.AeroCoeffs(uoo);
cLAlphaWing = cLWing/tempAlpha;
cLAlphaTail = cLTail/tempAlpha;
clear tempAlpha

AREffectiveWing = beta*wing.aspectRatio;    % wing effective aspect ratio
AREffectiveTail = beta*tail.aspectRatio;    % wing effective aspect ratio

sWing = (wing.span + d)/2;    % maximum wing semispan including body
sTail = (tail.span + d)/2;    % maximum tail semispan including body

lambdaWing = wingChordLengths(2) / ...    % tail taper ratio
             wingChordLengths(1);
lambdaTail = tailChordLengths(2) / ...    % tail taper ratio
             tailChordLengths(1);
```

Nielsen method

```
vortexRelativeToRoot = pi/4*(sWing - r);    % approximate from lecture 5, page 6
vortexLateralPosition = vortexRelativeToRoot + r;
vortexVerticalPosition = 0;    % hard-coded as deltaWing = 0 and alpha = 0
lambdaTail = tail.chordLengths(2)/tail.chordLengths(1);
i = -1.75;    % chart 7.c (with lambdaTail approximated) from NACA 1307

% K coefficient functions
% nose
KN = 2*pi*rN^2/(wing.area*cLAlphaWing);
% wing in presence of body, equation 14 in NACA 1307 for slender-body KW(B)
KWBFunc = @(tau) 2/pi*((1 + tau.^4).*(1/2*atan(1/2*(1./tau - tau)) + pi/4) - ...
                        tau.^2.*((1./tau - tau) + 2*atan(tau))) ./ ...
                       (1 - tau).^2;
% body in presence of wing, equation 21 in NACA 1307 for slender-body KB(W)
KBWFunc = @(tau) ((1 - tau.^2).^2 - ...
```

```
                  2/pi*((1 + tau.^4).*(1/2*atan(1/2*(1./tau - tau)) + pi/4) - ...
                        tau.^2.*(1./tau - tau + 2*atan(tau)))) ./ ...
                 (1 - tau).^2;
% k coefficients are irrelevant for the excercise
% kWB appears in equation 19 in NACA 1307 for slender-body kW(B)
% kBW appears in equation 33 in NACA 1307 for slender-body kB(W)

% lift line slopes corrected with K coefficients
% partial configuration - body only
cLAlphaNose = KN*cLAlphaWing;
cLAlphaBody = cLAlphaNose;
% partial configuration - body and wing
KBW = KBWFunc(r/sWing);
KWB = KWBFunc(r/sWing);
cLAlphaBW = KBW*cLAlphaWing;
cLAlphaWB = KWB*cLAlphaWing;
cLAlphaBodyWing = cLAlphaNose + cLAlphaBW + cLAlphaWB;
% full configuration - body, wing and ttail
KBT = KBWFunc(r/sTail);
KTB = KWBFunc(r/sTail);
cLAlphaBT = KBT*(tail.area/wing.area)*cLAlphaTail;
cLAlphaTB = KTB*(tail.area/wing.area)*cLAlphaTail;
cLAlphaTV = cLAlphaWing*cLAlphaTail*KWB*i*(sTail - r) / ...
                (2*pi*tail.aspectRatio*vortexRelativeToRoot);
cLAlphaFull = cLAlphaNose + ...
              cLAlphaBW + cLAlphaWB + ...
              cLAlphaBT + cLAlphaTB + ...
              cLAlphaTV;

% centers of pressure - xBar/cR
% wing
xBarCRWing = 0.525;    % chart 11.c from NACA 1307
xBarCRWB = xBarCRWing;
xBarCRBW = 0.45;    % chart 16.g (with r/sWing approximated) from NACA 1307
% tail
xBarCRTail = 0.52;    % chart 11.c (with lambdaTail approximated) from NACA 1307
xBarCRTB = xBarCRTail;
xBarCRBT = 0.49;    % chart 16.g (with lambdaTail, r/sTail approximated) from NACA 1307

% center of pressure location relative to missile apex
% missile components
lNBar = lN*(1 - VN/(pi*rN^2*lN));
lWBBar = lW + xBarCRWB*wing.chordLengths(1);
lBWBar = lW + xBarCRBW*wing.chordLengths(1);
lTBBar = lT + xBarCRTB*tail.chordLengths(1);
lBTBar = lT + xBarCRBT*tail.chordLengths(1);
lTVBar = lTBBar;
lBVBar = lBTBar;    % cLBV = 0 because gammaM = 0
% partial configuration - body only
lBarBody = lNBar*cLAlphaNose/cLAlphaBody;
% partial configuration - body and wing
lBarBodyWing = (lNBar*cLAlphaNose + ...
                lWBBar*cLAlphaWB + ...
```

```
                 lBWBar*cLAlphaBW) / ...
                cLAlphaBodyWing;
% full configuration - body, wing and tail
lBarFull = (lNBar*cLAlphaNose + ...
            lWBBar*cLAlphaWB + ...
            lBWBar*cLAlphaBW + ...
            lTBBar*cLAlphaTB + ...
            lBTBar*cLAlphaBT +  ...
            lTVBar*cLAlphaTV) / ...
           cLAlphaFull;
```

Numerical results

```
% given values, calculated numerically
cLAlphaBodyGiven = 2.2;
cLAlphaBodyWingGiven = 13.55;
cLAlphaFullGiven = 17.5;
lBarBodyGiven = 2.6*d;
lBarBodyWingGiven = 5.8*d;
lBarFullGiven = 7.2*d;

% corrected lift line slopes for different reference area
sRefGiven = pi*d^2/4;    % given reference area - cross section of body
normalizationRatio = wing.area/sRefGiven;
cLAlphaBodyCorrected = cLAlphaBody*normalizationRatio;
cLAlphaBodyWingCorrected = cLAlphaBodyWing*normalizationRatio;
cLAlphaFullCorrected = cLAlphaFull*normalizationRatio;

% printed output
fprintf('============================================================\n')
fprintf('Nielsen method results for body, wing and tail configuration\n')
fprintf('============================================================\n')
fprintf('Setup parameters:\n')
fprintf('-----------------\n')
fprintf('Missile diameter: %d [m] (arbitrary)\n', d)
fprintf('Mach number: %.0f\n', M)
fprintf('Air temperature: %d [K]\n', T)
fprintf('Resulting airspeed: %.1f [m/s]\n\n', uoo)
fprintf('Method parameters:\n')
fprintf('------------------\n')
fprintf('(r/s)wing: %.4f\n', r/sWing)
fprintf('(r/s)tail: %.4f\n', r/sTail)
fprintf('Wing taper ratio: 0\n')
fprintf('Tail taper ratio: %.4f\n', lambdaTail)
fprintf('Wing effective aspect ratio: %.3f\n', AREffectiveWing)
fprintf('Tail effective aspect ratio: %.3f\n', AREffectiveTail)
fprintf('KN: %.4f\n', KN)
fprintf('KW(B): %.3f\n', KWB)
fprintf('KB(W): %.4f\n', KBW)
fprintf('KB(T): %.4f\n', KBT)
fprintf('KT(B): %.3f\n', KTB)
fprintf('fw - rw: %.4f [m]\n', vortexRelativeToRoot)
```

```matlab
fprintf('h/s: 0\n')
fprintf('(xBar/Cr)W(B): %.4f\n', xBarCRWB)
fprintf('(xBar/Cr)B(W): %.4f\n', xBarCRBW)
fprintf('(xBar/Cr)T(B): %.4f\n', xBarCRTB)
fprintf('(xBar/Cr)B(T): %.4f\n\n', xBarCRTB)
fprintf('Component lift line slopes:\n')
fprintf('--------------------------\n')
fprintf('CLAlphaN: %.4f\n', cLAlphaNose)
fprintf('CLAlphaWB: %.3f\n', cLAlphaWB)
fprintf('CLAlphaBW: %.3f\n', cLAlphaBW)
fprintf('CLAlphaTB: %.3f\n', cLAlphaTB)
fprintf('CLAlphaBT: %.3f\n', cLAlphaBT)
fprintf('CLAlphaTV: %.3f\n', cLAlphaTV)
fprintf('CLAlphaBV: 0\n\n')
fprintf('Component centers of pressure:\n')
fprintf('-----------------------------\n')
fprintf('lNBar: %.3f\n', lNBar)
fprintf('lWBBar: %.3f\n', lWBBar)
fprintf('lBWBar: %.3f\n', lBWBar)
fprintf('lTBBar: %.2f\n', lTBBar)
fprintf('lBTBar: %.2f\n', lBTBar)
fprintf('lTVBar: %.2f\n', lTVBar)
fprintf('lBVBar: %.2f\n\n', lBVBar)
fprintf('Lift line slopes per configuration:\n')
fprintf('----------------------------------\n')
fprintf('Result correction factor: %.3f\n', normalizationRatio)
fprintf('Body only configuration\n')
fprintf('Calculation: %.3f\n', cLAlphaBodyCorrected)
fprintf('Given: %.3f\n', cLAlphaBodyGiven)
fprintf('Error: %.3f [%%]\n\n', ...
        abs(cLAlphaBodyCorrected - cLAlphaBodyGiven)/cLAlphaBodyGiven*100);
fprintf('Body and wing configuration\n')
fprintf('Calculation: %.3f\n', cLAlphaBodyWingCorrected)
fprintf('Given: %.3f\n', cLAlphaBodyWingGiven)
fprintf('Error: %.3f [%%]\n\n', ...
        abs(cLAlphaBodyWingCorrected - cLAlphaBodyWingGiven) / ...
        cLAlphaBodyWingGiven*100);
fprintf('Full configuration\n')
fprintf('Calculation: %.3f\n', cLAlphaFullCorrected)
fprintf('Given: %.3f\n', cLAlphaFullGiven)
fprintf('Error: %.3f [%%]\n\n', ...
        abs(cLAlphaFullCorrected - cLAlphaFullGiven)/cLAlphaFullGiven*100);
fprintf('Centers of pressure per configuration:\n')
fprintf('-------------------------------------\n')
fprintf('Body only configuration\n')
fprintf('Calculation: %.3f\n', lBarBody)
fprintf('Given: %.3f\n', lBarBodyGiven)
fprintf('Error: %.3f [%%]\n\n', ...
        abs(lBarBody - lBarBodyGiven)/lBarBodyGiven*100)
fprintf('Body and wing configuration\n')
fprintf('Calculation: %.3f\n', lBarBodyWing)
fprintf('Given: %.3f\n', lBarBodyWingGiven)
fprintf('Error: %.3f [%%]\n\n', ...
```

```
        abs(lBarBodyWing - lBarBodyWingGiven)/lBarBodyWingGiven*100)
fprintf('Full configuration\n')
fprintf('Full configuration: %.3f\n', lBarFull)
fprintf('Given full configuration: %.3f\n', lBarFullGiven)
fprintf('Error: %.3f [%%]\n\n', ...
        abs(lBarFull - lBarFullGiven)/lBarFullGiven*100)
```

Plots

```
% plot parameters
lw = 1.2;    % line width
fs = 15;     % font size
load('colors.mat')

% FIGURE 1: geometry of wings and VLM tiles
figure
hold on
grid on
title('Geometry of wing, tail and $VLM$ tiles', 'FontSize', fs)
xlabel('$\frac{x}{d}$', 'FontSize', fs)
ylabel('$\frac{y}{d}$', 'FontSize', fs)
wing.PlotSelf('color', colors.blue, 'tiles');
tail.PlotSelf('color', colors.purple, 'tiles', 'z offset', 1);
zticks([0, 1])
zticklabels({'Wing', 'Tail'})
view(-68, 27)
axis image
hold off
```

VLM code is unchanged from homework 4