EMCH 721: Homework 03

Instructor: Dr. Victor Giurgiutiu
Student: J.C. Vaught

Table of Contents

Problem Context

Consider an aircraft wing undergoing ground and flight testing at airspeed U (Figure 1). The wing is fixed at one root and free at the tip. It has straight elastic, mass, and aerodynamic axes located at points P, C, and Q, respectively. The mass offset from P to C is x_{CP}; the aerodynamic offset from P to Q is x_{QP}. The wing executes small-amplitude plunge $w(z,t)$ (positive downward) and pitch $\phi(z,t)$ (positive nose-up, clockwise) oscillations.

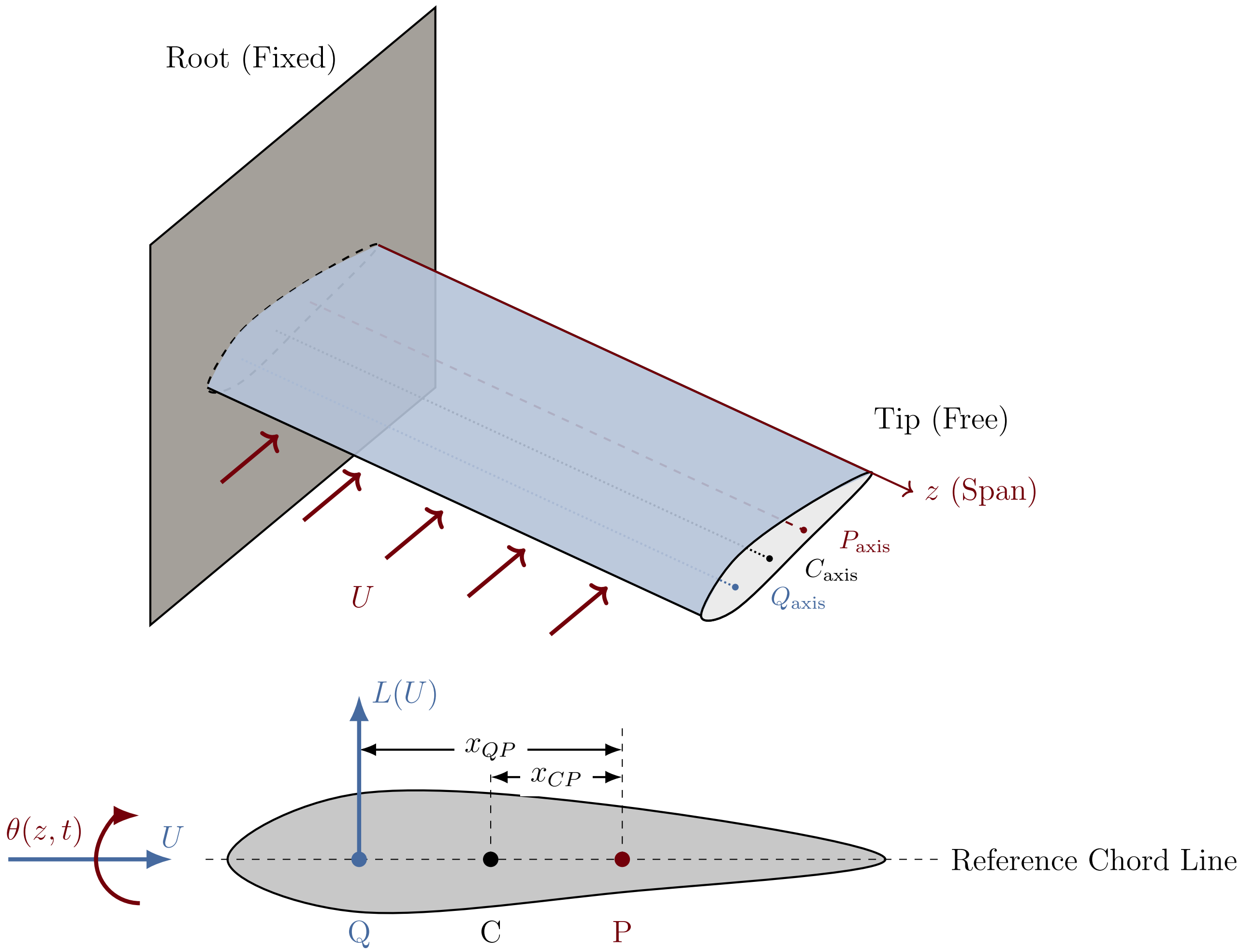


Figure 1: Overview of the rectangular test wing with spanwise axes P, C, and Q, and the incoming flow U. Cross-section shows offsets x_{CP} and x_{QP}.

Notes

Submission Notes

- Always display input data.
- Show work and relevant comments.
- Attach MATLAB codes to receive partial credit where applicable.
- Report numerical results to three significant digits (use four if the first digit is 1).
- Problems solved beyond the minimum requirements may receive bonus points.
- Challenge problems are optional and count toward extra credit.

Input Data

Input Data Summary

Parameter	**Value**
Air density ρ	1.225 kg/m^3
Chord c	0.45 m
Wingspan l	2.5 m
Mass m	3.2 kg
Mass moment of inertia I_0	0.055 $\text{kg} \cdot \text{m}^2$
Plunge frequency f_h	1.8 Hz
Pitch frequency f_θ	5.3 Hz
Static offset x_{CP}	$-10\%\,c$
Aerodynamic offset x_{QP}	$35\%\,c$

A.1 Flexural vibration of a cantilever beam of length l, with section properties EI, m, I_0

(a) Recall the formulae for flexural frequencies and modeshapes of a fixed-free beam.

(b) Calculate and display the roots γl of the characteristic equation for $N_w = 4$.

(c) Calculate and display the flexural stiffness EI such that the fundamental flexural frequency matches the plunge frequency of the rigid wing section and display the resulting EI value and units.

(d) Calculate and display the wavenumbers, natural frequencies in rad/s and Hz.

(e) Calculate and plot the modeshapes.

Step

Step 1: Governing ODE (Euler–Bernoulli)
Since a uniform, slender beam obeys the small-deflection Euler–Bernoulli equation, separating variables with $w(x,t) = \phi(x)e^{i\omega t}$ turns time derivatives into the factor $-\omega^2$, giving

$$EI\,\phi''''(x) = m\,\omega^2\,\phi(x)$$

and therefore the spatial ODE

$$\phi'''' - \delta^4\phi = 0, \qquad \delta^4 = \frac{m\omega^2}{EI}.$$

Step

Step 2: Boundary conditions (fixed–free)
Since the root is clamped, displacement and slope vanish at $x = 0$:

$$\phi(0) = 0, \qquad \phi'(0) = 0.$$

Because the tip is free, bending moment and shear must vanish at $x = l$:

$$\phi''(l) = 0, \qquad \phi'''(l) = 0.$$

Applying these four conditions to the general solution enforces the characteristic relation

$$\cos\beta_j \cosh\beta_j = -1, \quad \beta_j = \delta_j l.$$

Step

Step 3: Mode-shape coefficients (from tip compatibility)
Since the two free-end conditions couple the integration constants, solving them yields

$$B_j = \frac{\sinh\beta_j - \sin\beta_j}{\cosh\beta_j + \cos\beta_j}, \qquad A_j = \frac{1}{\sqrt{l}} \text{ (chosen for unit modal mass).}$$

Results

Flexural natural frequencies

Roots of $\cos\beta\cosh\beta = -1$ give $\beta_1 = 1.875,\ \beta_2 = 4.694,\ \beta_3 = 7.855,\ \beta_4 = 10.996, \ldots$

$$\omega_j = \beta_j^2\sqrt{\frac{EI}{m\,l^4}}, \qquad f_j = \frac{\omega_j}{2\pi}.$$

Results

Flexural modeshapes

$$\phi_j(x) = A_j\Big[(\cosh\delta_j x - \cos\delta_j x) - B_j(\sinh\delta_j x - \sin\delta_j x)\Big],$$
$$\delta_j = \frac{\beta_j}{l}, \qquad B_j = \frac{\sinh\beta_j - \sin\beta_j}{\cosh\beta_j + \cos\beta_j}, \qquad A_j = \frac{1}{\sqrt{l}}.$$

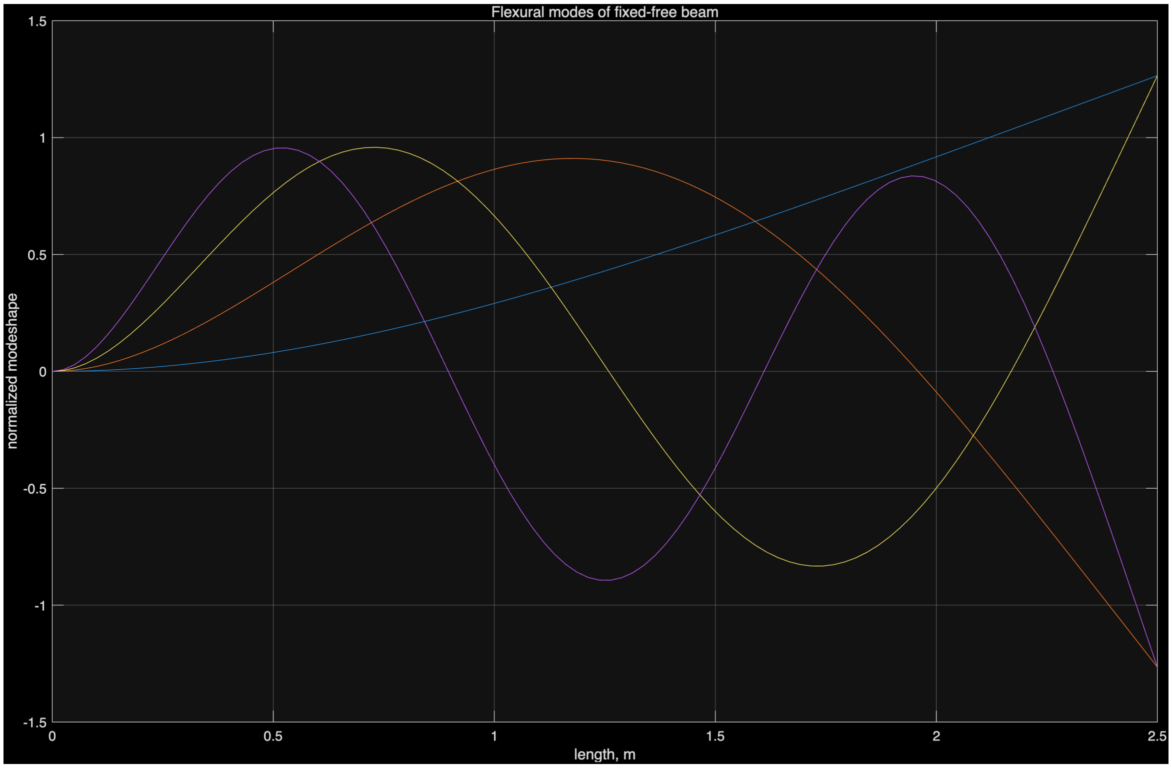
The first four β_j above provide the modeshapes for the required $N_w = 4$.

Results

```
Section A1: flexural vibration of a fixed-free beam
 (a) students should recall relevant formulae from
 in-class instruction and class notes
 (b) NW=4
 roots of flexural characteristic equation =
    1.8751    4.6941    7.8548   10.9955
 (c) flexural stiffness EI =1293 N*m^2/m
 (d) flexural wavenumbers, natural freq. in rad/s and Hz
 gW,rad/m wW,rad/s fW,Hz =
    0.7500   11.3097    1.8000
    1.8776   70.8769   11.2804
    3.1419  198.4573   31.5855
    4.3982  388.8972   61.8949
 (e) please see plot
```

Shows A1(b)–(d): characteristic roots for $N_w = 4$, the matched flexural stiffness EI, flexural wavenumbers, and the corresponding natural frequencies in rad/s and Hz.

Results



Part A.1(e): plot of the first four cantilever flexural modeshapes used in the A.1(e) discussion.

A.2 Torsional vibration of a cantilever beam of length l, with section properties GJ, m, I_0

(a) Recall the formulae for torsional frequencies and modeshapes of a fixed-free beam.

(b) Calculate and display the roots γl of the characteristic equation for $N_\phi = 3$.

(c) Calculate and display the torsional stiffness GJ such that the fundamental torsional frequency matches the pitch frequency of the rigid wing section and display the resulting GJ value and units.

(d) Calculate and display the wavenumbers, natural frequencies in rad/s and Hz.

(e) Calculate and plot the modeshapes.

Step

Step 1: Governing torsion ODE
Since uniform Saint–Venant torsion transmits torque as $GJ\,\phi'(z,t)$, equilibrium of a spanwise slice gives

$$GJ\,\phi''(z,t) = I_0\,\ddot{\phi}(z,t).$$

Assuming $\phi(z,t) = \Phi(z)e^{i\omega t}$ yields the spatial equation

$$\Phi''(z) + \gamma^2\Phi(z) = 0, \qquad \gamma^2 = \frac{I_0\omega^2}{GJ}.$$

Step

Step 2: Fixed–free boundary conditions
Since the root is clamped, the twist vanishes:

$$\Phi(0) = 0.$$

Since the tip is free (zero applied torque), the Saint–Venant torque must vanish:

$$GJ\,\Phi'(l) = 0 \;\Rightarrow\; \Phi'(l) = 0.$$

With the general solution $\Phi(z) = A\cos\gamma z + B\sin\gamma z$, the clamp sets $A = 0$, and the free-tip condition requires

$$B\gamma\cos(\gamma l) = 0 \;\Rightarrow\; \cos(\gamma l) = 0.$$

Thus the admissible wavenumbers satisfy $\gamma_j l = (2j-1)\frac{\pi}{2}$.

Results

Torsional natural frequencies

$$\gamma_j l = \frac{(2j-1)\pi}{2}, \qquad \omega_j = \gamma_j \sqrt{\frac{GJ}{I_0}}, \qquad f_j = \frac{\omega_j}{2\pi} = (2j-1)\left(\frac{1}{4l}\right)\sqrt{\frac{GJ}{I_0}}.$$

$$\gamma_1 l = 1.571, \quad \gamma_2 l = 4.712, \quad \gamma_3 l = 7.854, \qquad C_\phi \equiv \sqrt{\tfrac{GJ}{I_0}}, \quad f_j = (2j-1)\frac{C_\phi}{4l}.$$

Results

Torsional modeshapes

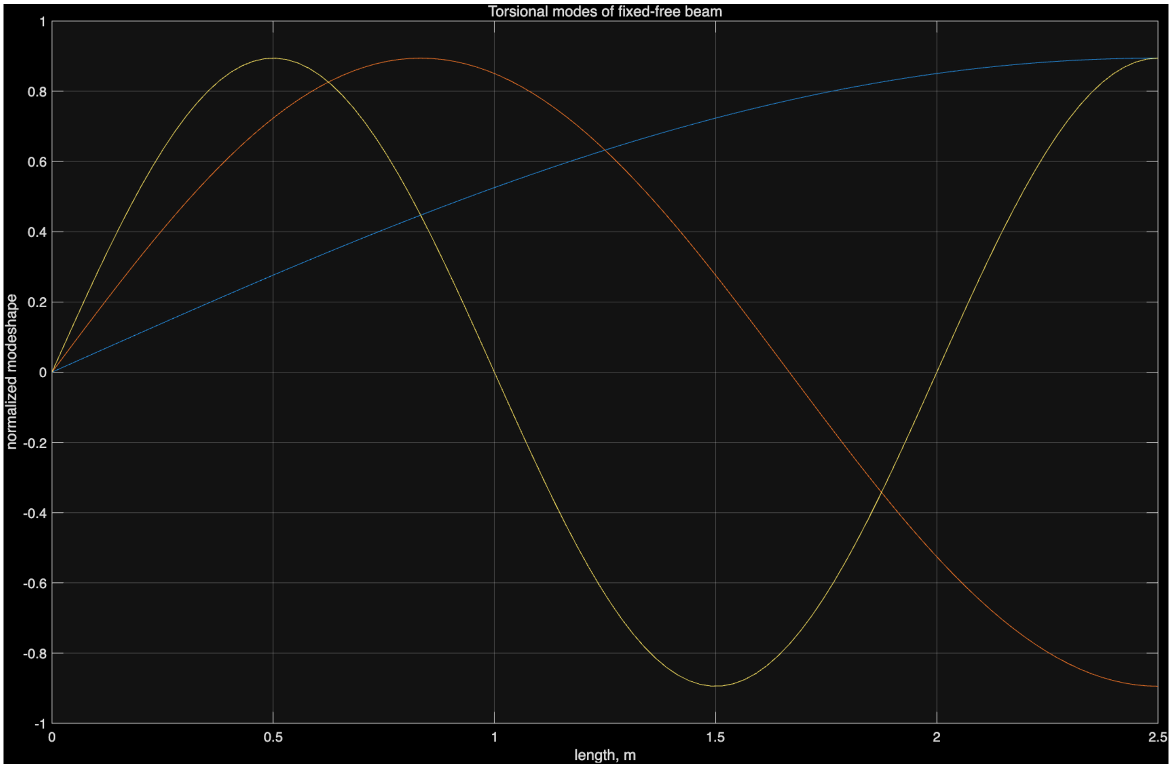
$$\Phi_j(z) = B_j \sin(\gamma_j z), \qquad B_j = \sqrt{\frac{2}{l}} \text{ (unit modal mass)}, \qquad \gamma_j = \frac{(2j-1)\pi}{2l}.$$

Results

```
 HW03 structural dynamics flutter analysis -- JC Vaught
 input data
 air density rho=1.225kg/m^3
 c=0.5m, m=3.2kg/m, I0=0.0550kg*m^2/m, L=2.5m
 static offset xCP=-10.0%, -0.0450m
 aerodynamic offset xQP=35.0%, 0.1575m
 rigid body frequencies fh=1.8Hz, ft=5.3Hz
 wing span L=2.5 m

Section A2: torsional vibration of a fixed-free beam
 (a) students should recall relevant formulae from
 in-class instruction and class notes
 (b) NPhi=3
 roots of torsional characteristic equation =
    1.5708    4.7124    7.8540
 (c) torsional stiffness GJ =154.5 N*m^2/m
 (d) torsional wavenumbers, natural freq. in rad/s and Hz
 gPhi,rad/m wPhi,rad/s fPhi,Hz =
    0.6283   33.3009    5.3000
    1.8850   99.9026   15.9000
    3.1416  166.5044   26.5000
 (e) please see plot
```

Shows A.2(b)–(d): characteristic roots for $N_\phi = 3$, matched torsional stiffness GJ, torsional wavenumbers, and natural frequencies in rad/s and Hz.

Results



Part A.2(e): first three cantilever torsional modeshapes used in the A.2(e) discussion.

B. Aeroelastic equations of a straight wing of length l, and section properties EI, GJ, m, I_0, x_{CP}, x_{QP}

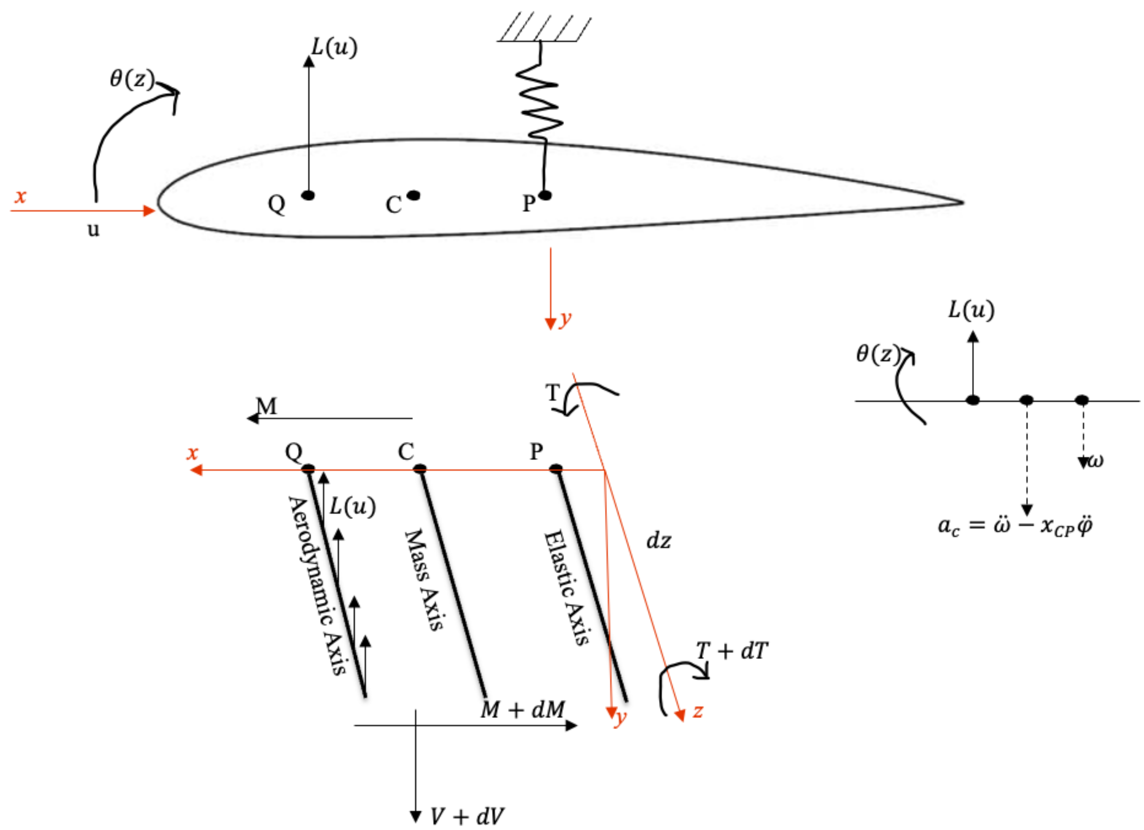
(a) Draw the free body diagram of an infinitesimal element of span dz.

(b) Write the plunge and pitch equations of motion as a set of two PDEs in space z and time t.

(c) Eliminate the time dependency by assuming solution proportional to e^{st} and cast the equations of motion as a set of two ODEs in space z.

(d) Apply normal mode expansion with N_w flexural modes and N_ϕ torsional modes and recast the equations of motion as an algebraic matrix eigenvalue problem of dimension $N = N_w + N_\phi$ that depends on airspeed U.

Results



Part B(a): free-body diagram showing elastic axis P, mass axis C (x_{CP} aft), aerodynamic axis Q (x_{QP} aft), lift $L(U)$, pitching moment, internal shear V, and torque T over dz.

Step

Set conventions and kinematics

Since z points spanwise from the root and $w(z,t)$ is positive downward, a nose-up twist $\phi(z,t)$ is positive clockwise (as in Part A). The mass center C is offset by x_{CP} from the elastic axis P, so its vertical acceleration is $\ddot{w} - x_{CP}\ddot{\phi}$. The aerodynamic center Q is offset by x_{QP} from P.

Step

Internal force/torque relations

Since Euler–Bernoulli bending and Saint–Venant torsion apply,

$$M = -EI\,w'', \qquad V = \frac{dM}{dz} = EI\,w''', \qquad T = GJ\,\phi', \qquad \frac{dT}{dz} = GJ\,\phi''.$$

These relations let us replace dV and dT by derivatives of w and ϕ.

Step

Aerodynamic linearization

Since small perturbations are assumed, the sectional lift is linearized as

$$L(U) = L_0(U)\,\phi,$$

acting upward at Q; the associated aerodynamic moment about P is $L_0(U)\,x_{QP}\,\phi$ (nose-up positive).

Step

Part B(b): plunge equilibrium (force in y)

Since internal shear varies as $V(z+dz) = V + dV$ and $V = EI\,w'''$, and the mass center acceleration is $\ddot{w} - x_{CP}\ddot{\phi}$, summing forces in y over dz gives

$$V - (V + dV) + L(U)\,dz + m\,dz\big(\ddot{w} - x_{CP}\ddot{\phi}\big) = 0.$$

Since $dV/dz = EI\,w''''$, dividing by dz yields the plunge PDE

$$m\,\ddot{w} - m x_{CP}\,\ddot{\phi} + EI\,w'''' + L_0(U)\,\phi = 0.$$

Step

Part B(b): pitch equilibrium (moment about elastic axis P)
Since torsion varies as $T(z+dz) = T + dT$ with $T = GJ\,\phi'$, and lift acts at Q giving $L_0(U)x_{QP}\phi$, taking moments about P over dz:

$$T - (T + dT) - mx_{CP}\,dz\,\ddot{w} + I_p\,dz\,\ddot{\phi} - L_0(U)\,x_{QP}\,\phi\,dz = 0.$$

Using $dT/dz = GJ\,\phi''$ and dividing by dz gives the pitch PDE

$$-mx_{CP}\,\ddot{w} + I_p\,\ddot{\phi} - GJ\,\phi'' - L_0(U)\,x_{QP}\,\phi = 0.$$

Results

Part B(b): coupled plunge–pitch PDEs

$$m\,\ddot{w} - mx_{CP}\,\ddot{\phi} + EI\,w'''' + L_0(U)\,\phi = 0, \qquad -mx_{CP}\,\ddot{w} + I_p\,\ddot{\phi} - GJ\,\phi'' - L_0(U)\,x_{QP}\,\phi = 0.$$

Step

Part B(c): assume harmonic time factor e^{st}
Since $w(z,t) = \hat{w}(z)e^{st}$ and $\phi(z,t) = \hat{\phi}(z)e^{st}$, we replace $\dot{} \to s$ and obtain ODEs in z:

$$ms^2\hat{w} - mx_{CP}s^2\hat{\phi} + EI\,\hat{w}'''' + L_0(U)\,\hat{\phi} = 0,$$

$$-mx_{CP}s^2\hat{w} + I_ps^2\hat{\phi} - GJ\,\hat{\phi}'' - L_0(U)x_{QP}\,\hat{\phi} = 0.$$

Results

Part B(c): time-harmonic spanwise ODEs

$$ms^2\hat{w} - mx_{CP}s^2\hat{\phi} + EI\,\hat{w}'''' + L_0(U)\,\hat{\phi} = 0, \qquad -mx_{CP}s^2\hat{w} + I_ps^2\hat{\phi} - GJ\,\hat{\phi}'' - L_0(U)x_{QP}\,\hat{\phi} = 0.$$

Step

Part B(d): normal-mode expansion and projections

Since the uncoupled mode families are orthonormal, expand

$$w(z,t) = \sum_{j=1}^{N_w} \eta_j^w(t) W_j(z), \qquad \phi(z,t) = \sum_{j=1}^{N_\phi} \eta_j^\phi(t) \Phi_j(z).$$

Project the s-domain ODEs onto W_p and Φ_p; using $\int_0^l W_p W_q \, m \, dz = \delta_{pq}$ and $\int_0^l \Phi_p \Phi_q \, dz = \delta_{pq}$ gives

$$[m^{ww}]\,\ddot{\eta}^w + [m^{w\phi}]\,\ddot{\eta}^\phi + [k_s^{ww}]\,\eta^w + [k_A^{w\phi}(U)]\,\eta^\phi = 0,$$

$$[m^{\phi w}]\,\ddot{\eta}^w + [m^{\phi\phi}]\,\ddot{\eta}^\phi + [k_s^{\phi\phi}]\,\eta^\phi + [k_A^{\phi\phi}(U)]\,\eta^\phi = 0.$$

Here

$$[m^{ww}]_{pq} = m \int_0^l W_p W_q \, dz, \quad [m^{w\phi}]_{pq} = -m x_{CP} \int_0^l W_p \Phi_q \, dz,$$

$$[m^{\phi\phi}]_{pq} = I_p \int_0^l \Phi_p \Phi_q \, dz,$$

$$[k_s^{ww}]_{pq} = EI \int_0^l W_p'''' W_q \, dz, \quad [k_s^{\phi\phi}]_{pq} = -GJ \int_0^l \Phi_p'' \Phi_q \, dz,$$

$$[k_A^{w\phi}(U)]_{pq} = \int_0^l L_0(U)\, \Phi_q W_p \, dz, \quad [k_A^{\phi\phi}(U)]_{pq} = -\int_0^l L_0(U)\, x_{QP}\, \Phi_q \Phi_p \, dz.$$

Results

Part B(d): eigenvalue form

Assuming e^{st} for modal coordinates yields the algebraic problem

$$\left[s^2 \begin{bmatrix} [m^{ww}] & [m^{w\phi}] \\ [m^{\phi w}] & [m^{\phi\phi}] \end{bmatrix} + \begin{bmatrix} [k_s^{ww}] & 0 \\ 0 & [k_s^{\phi\phi}] \end{bmatrix} + \begin{bmatrix} [k_A^{w\phi}(U)] \\ [k_A^{\phi\phi}(U)] \end{bmatrix} \right] \begin{bmatrix} \eta^w \\ \eta^\phi \end{bmatrix} = \mathbf{0}.$$

C. Ground vibration test (GVT) simulation for $N_w = 4$, $N_\phi = 3$

Problem Assumption

Assume $U = 0$.

(a) Display the number of modes N_w, N_ϕ of the uncoupled problems and calculate and display the total number of modes N of the coupled problem.

(b) Calculate and display the coupled flexural and torsional frequencies and eigenvectors. Discuss your results.

(c) Calculate and plot the coupled modeshapes and discuss your results.

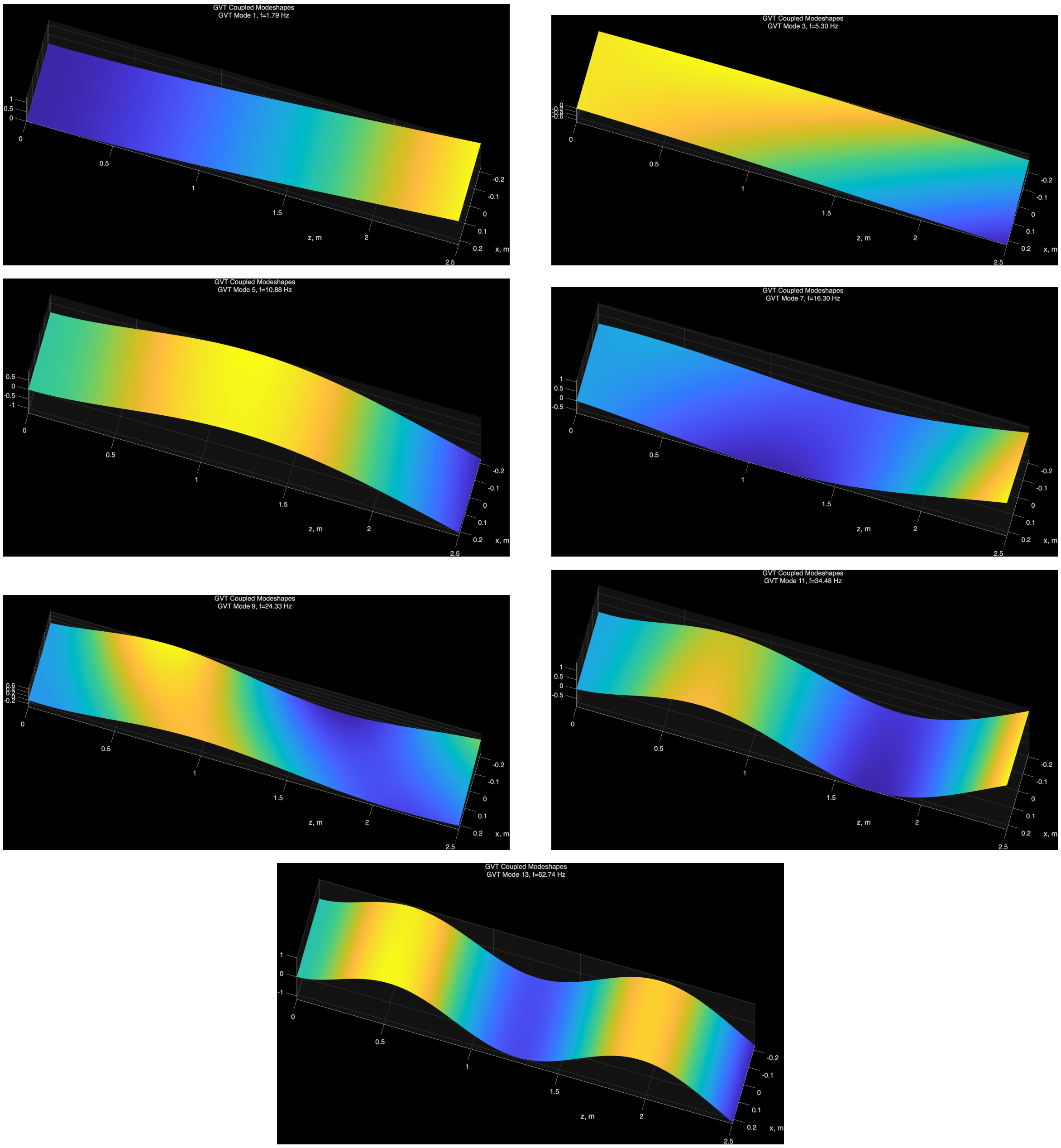
Results

```
HW03 structural dynamics flutter analysis -- JC Vaught
input data
air density rho=1.225kg/m^3
c=0.5m, m=3.2kg/m, I0=0.0550kg*m^2/m, L=2.5m
static offset xCP=-10.0%, -0.0450m
aerodynamic offset xQP=35.0%, 0.1575m
rigid body frequencies fh=1.8Hz, ft=5.3Hz
wing span L=2.5 m

Section C: GVT analysis
(a) NW=4, NPhi=3, N=7
(b) coupled GVT frequencies f, Hz =
   1.7873    1.7873    5.3047    5.3047   10.8756   10.8756   16.3014   16.3014   24.3264   24.3264   34.4780   34.4780   62.7435   62.7435
GVT eigenvectors =
   1.0000    1.0000   -0.3894   -0.3894    0.0738    0.0738    0.0924    0.0924   -0.0264   -0.0264   -0.0009   -0.0009    0.0049    0.0049
   0.0001    0.0001    0.0285    0.0285    1.0000    1.0000   -0.5851   -0.5851    0.2575    0.2575   -0.0313   -0.0313   -0.0161   -0.0161
   0.0000    0.0000    0.0006    0.0006    0.0038    0.0038    0.0584    0.0584    0.3798    0.3798    1.0000    1.0000    0.0704    0.0704
   0.0000    0.0000    0.0001    0.0001    0.0000    0.0000    0.0025    0.0025    0.0272    0.0272   -0.0355   -0.0355    1.0000    1.0000
   0.0409    0.0409    1.0000    1.0000   -0.1282   -0.1282    0.0220    0.0220   -0.0209   -0.0209   -0.0140   -0.0140   -0.0137   -0.0137
  -0.0012   -0.0012    0.0055    0.0055    0.2717    0.2717    1.0000    1.0000   -0.1813   -0.1813   -0.1330   -0.1330   -0.0242   -0.0242
   0.0001    0.0001   -0.0004   -0.0004   -0.0267   -0.0267    0.0611    0.0611    1.0000    1.0000   -0.4931   -0.4931   -0.1702   -0.1702
students should write their own discussion
(c) please see plots

success! HW03_tors_flex_flutter_EXAMPLE finished successfully
```

Part C(a)–(b): summary of uncoupled counts ($N_w = 4$, $N_\phi = 3$, $N = 7$), coupled frequencies/eigenvalues, and key numerical outputs from the GVT simulation at $U = 0$.

Results

Part C(c): coupled modeshapes ($N = 7$)



Coupled flexural/torsional shapes from the GVT case ($U = 0$) for modes 1–7 in order of increasing frequency.

Results

Part C discussion

Modes 1–2 are predominantly flexural; modes 3–4 show bending–torsion coupling; modes 5–7 are torsion-dominated. The slight shifts from the uncoupled frequencies reflect weak coupling at $U = 0$, consistent with the EI/GJ tuning in Parts A.1 and A.2. These mixed modes identify which shapes will interact first as airspeed grows toward flutter in Part D.

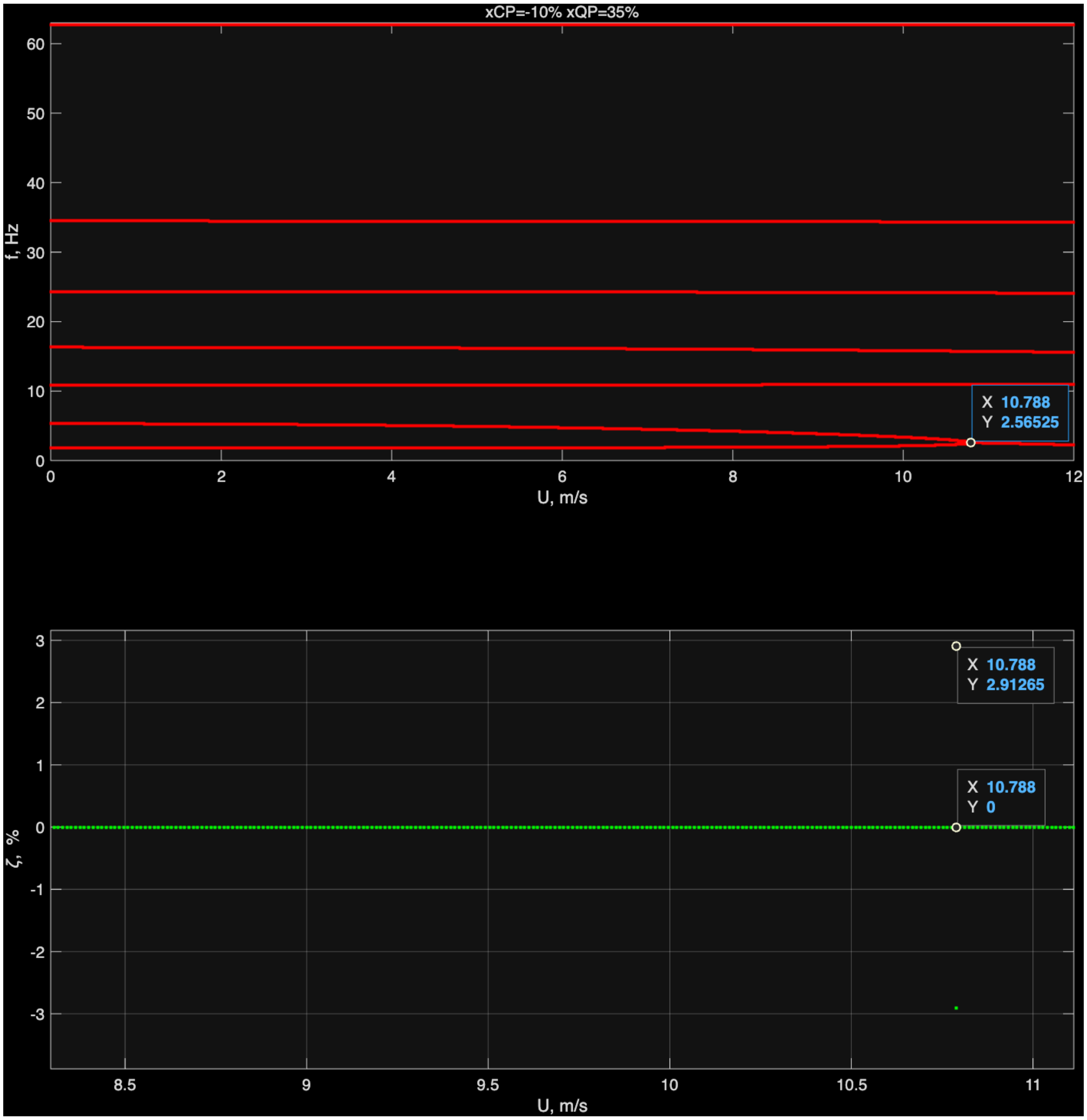
D. Flutter eigenvalue analysis for $N_w = 4$, $N_\phi = 3$

(a) Let airspeed $U = 0, \ldots, 12$ m/s with 1001 steps. Plot frequencies and damping vs airspeed and find the flutter speed U_F on the plots. Recall from HW01 the rigid airfoil flutter speed U_F^{rigid} and discuss the results comparatively.

(b) Calculate frequencies, damping, and eigenvectors around flutter speed, i.e., at $U = (1-\varepsilon)U_F$, U_F, $(1+\varepsilon)U_F$, $\varepsilon = 1\%$, and discuss your results.

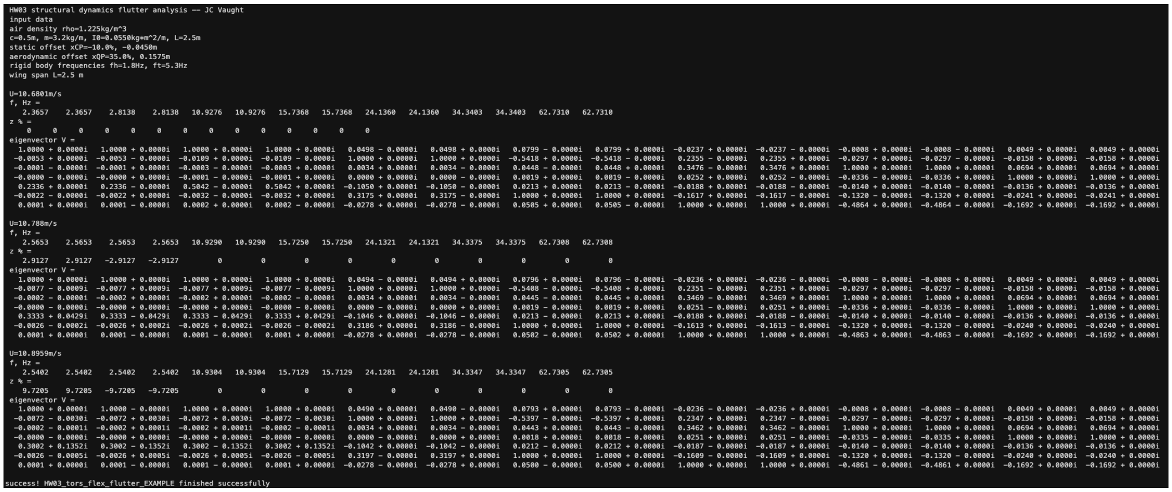
(c) Plot flutter modeshapes at U_F and discuss your results.

Results



Part D(a): frequency and damping trends vs airspeed U (0–12 m/s) used to read flutter onset.

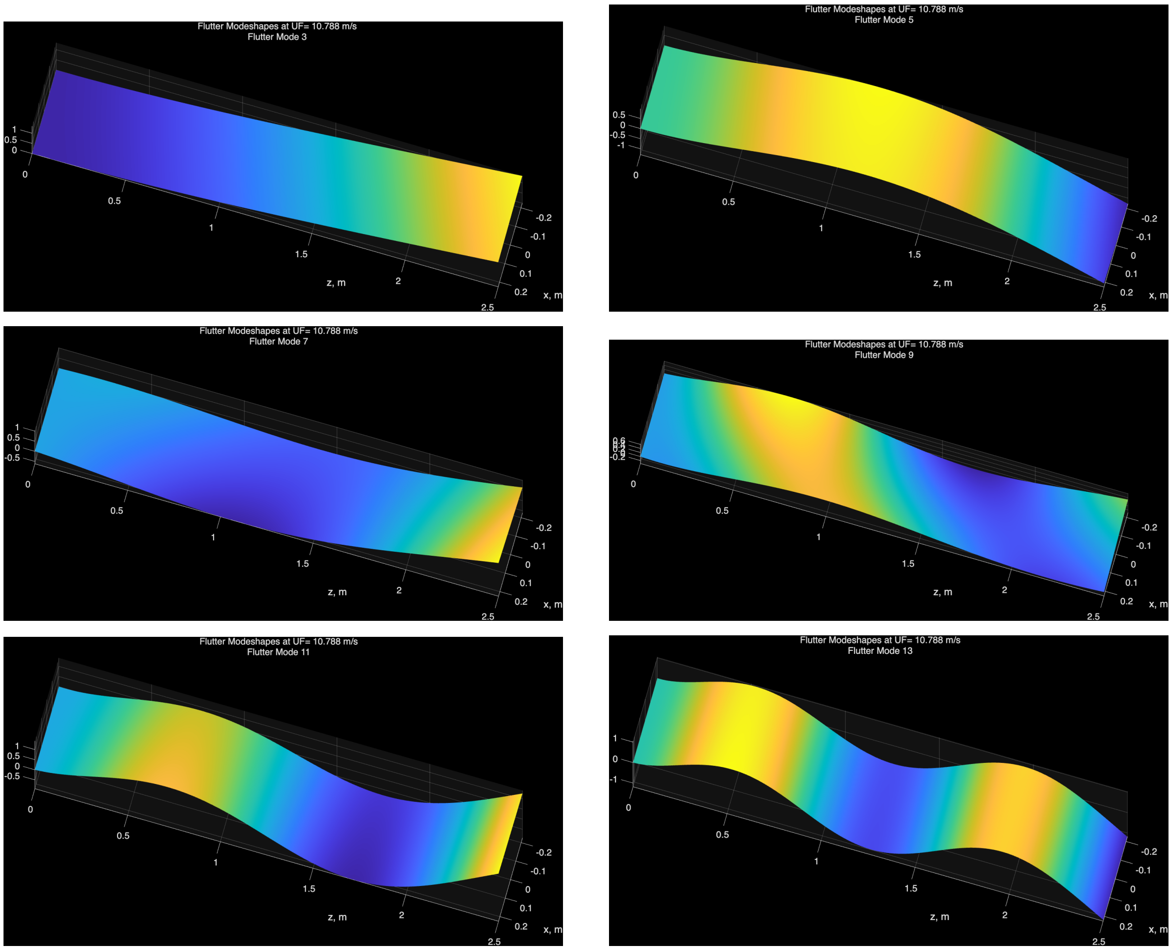
Results



```
HW03 structural dynamics flutter analysis -- JC Vaught
input data
air density rho=1.225kg/m^3
c=0.5m, m=3.2kg/m, I0=0.0550kg*m^2/m, L=2.5m
static offset xCP=-10.0%, -0.0450m
aerodynamic offset xQP=35.0%, 0.1575m
rigid body frequencies fh=1.8Hz, ft=5.3Hz
wing span L=2.5 m

U=10.6801m/s
f, Hz =
   2.3657    2.3657    2.8138    2.8138   10.9276   10.9276   15.7368   15.7368   24.1360   24.1360   34.3403   34.3403   62.7310   62.7310
z % =
    0     0     0     0     0     0     0     0     0     0     0     0     0     0
eigenvector V =
  1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   0.0498 - 0.0000i   0.0498 + 0.0000i   0.0799 - 0.0000i   0.0799 + 0.0000i  -0.0237 + 0.0000i  -0.0237 - 0.0000i  -0.0008 + 0.0000i  -0.0008 - 0.0000i   0.0049 + 0.0000i   0.0049 + 0.0000i
 -0.0053 + 0.0000i  -0.0053 - 0.0000i  -0.0109 + 0.0000i  -0.0109 - 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.5418 + 0.0000i  -0.5418 - 0.0000i   0.2355 - 0.0000i   0.2355 + 0.0000i  -0.0297 + 0.0000i  -0.0297 - 0.0000i  -0.0158 + 0.0000i  -0.0158 + 0.0000i
 -0.0001 - 0.0000i  -0.0001 + 0.0000i  -0.0003 - 0.0000i  -0.0003 + 0.0000i   0.0034 + 0.0000i   0.0034 - 0.0000i   0.0448 - 0.0000i   0.0448 + 0.0000i   0.3476 - 0.0000i   0.3476 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   0.0694 + 0.0000i   0.0694 + 0.0000i
 -0.0000 - 0.0000i  -0.0000 + 0.0000i  -0.0001 - 0.0000i  -0.0001 + 0.0000i   0.0000 + 0.0000i   0.0000 - 0.0000i   0.0019 + 0.0000i   0.0019 - 0.0000i   0.0252 + 0.0000i   0.0252 - 0.0000i  -0.0336 - 0.0000i  -0.0336 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i
  0.2336 + 0.0000i   0.2336 - 0.0000i   0.5042 - 0.0000i   0.5042 + 0.0000i  -0.1050 + 0.0000i  -0.1050 - 0.0000i   0.0213 + 0.0000i   0.0213 - 0.0000i  -0.0188 + 0.0000i  -0.0188 - 0.0000i  -0.0140 + 0.0000i  -0.0140 - 0.0000i  -0.0136 + 0.0000i  -0.0136 + 0.0000i
 -0.0022 - 0.0000i  -0.0022 + 0.0000i  -0.0032 - 0.0000i  -0.0032 + 0.0000i   0.3175 + 0.0000i   0.3175 - 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.1617 + 0.0000i  -0.1617 - 0.0000i  -0.1320 - 0.0000i  -0.1320 + 0.0000i  -0.0241 + 0.0000i  -0.0241 + 0.0000i
  0.0001 + 0.0000i   0.0001 - 0.0000i   0.0002 + 0.0000i   0.0002 - 0.0000i  -0.0278 + 0.0000i  -0.0278 - 0.0000i   0.0505 + 0.0000i   0.0505 - 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.4864 + 0.0000i  -0.4864 - 0.0000i  -0.1692 + 0.0000i  -0.1692 + 0.0000i

U=10.788m/s
f, Hz =
   2.5653    2.5653    2.5653    2.5653   10.9290   10.9290   15.7250   15.7250   24.1321   24.1321   34.3375   34.3375   62.7308   62.7308
z % =
   2.9127    2.9127   -2.9127   -2.9127         0         0         0         0         0         0         0         0         0         0
eigenvector V =
  1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   0.0494 - 0.0000i   0.0494 + 0.0000i   0.0796 + 0.0000i   0.0796 - 0.0000i  -0.0236 + 0.0000i  -0.0236 - 0.0000i  -0.0008 - 0.0000i  -0.0008 + 0.0000i   0.0049 + 0.0000i   0.0049 + 0.0000i
 -0.0077 - 0.0009i  -0.0077 + 0.0009i  -0.0077 + 0.0009i  -0.0077 - 0.0009i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.5408 - 0.0000i  -0.5408 + 0.0000i   0.2351 - 0.0000i   0.2351 + 0.0000i  -0.0297 + 0.0000i  -0.0297 - 0.0000i  -0.0158 + 0.0000i  -0.0158 + 0.0000i
 -0.0002 - 0.0000i  -0.0002 + 0.0000i  -0.0002 + 0.0000i  -0.0002 - 0.0000i   0.0034 + 0.0000i   0.0034 - 0.0000i   0.0445 - 0.0000i   0.0445 + 0.0000i   0.3469 - 0.0000i   0.3469 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   0.0694 + 0.0000i   0.0694 + 0.0000i
 -0.0000 - 0.0000i  -0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 + 0.0000i   0.0019 - 0.0000i   0.0019 + 0.0000i   0.0251 - 0.0000i   0.0251 + 0.0000i  -0.0336 + 0.0000i  -0.0336 - 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i
  0.3333 + 0.0429i   0.3333 - 0.0429i   0.3333 - 0.0429i   0.3333 + 0.0429i  -0.1046 + 0.0000i  -0.1046 - 0.0000i   0.0213 - 0.0000i   0.0213 + 0.0000i  -0.0188 + 0.0000i  -0.0188 - 0.0000i  -0.0140 + 0.0000i  -0.0140 - 0.0000i  -0.0136 + 0.0000i  -0.0136 + 0.0000i
 -0.0026 - 0.0002i  -0.0026 + 0.0002i  -0.0026 + 0.0002i  -0.0026 - 0.0002i   0.3186 + 0.0000i   0.3186 - 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.1613 + 0.0000i  -0.1613 - 0.0000i  -0.1320 + 0.0000i  -0.1320 - 0.0000i  -0.0240 + 0.0000i  -0.0240 + 0.0000i
  0.0001 + 0.0000i   0.0001 - 0.0000i   0.0001 - 0.0000i   0.0001 + 0.0000i  -0.0278 + 0.0000i  -0.0278 - 0.0000i   0.0502 - 0.0000i   0.0502 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.4863 + 0.0000i  -0.4863 - 0.0000i  -0.1692 + 0.0000i  -0.1692 + 0.0000i

U=10.8959m/s
f, Hz =
   2.5402    2.5402    2.5402    2.5402   10.9304   10.9304   15.7129   15.7129   24.1281   24.1281   34.3347   34.3347   62.7305   62.7305
z % =
   9.7205    9.7205   -9.7205   -9.7205         0         0         0         0         0         0         0         0         0         0
eigenvector V =
  1.0000 + 0.0000i   1.0000 - 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   0.0490 + 0.0000i   0.0490 - 0.0000i   0.0793 + 0.0000i   0.0793 - 0.0000i  -0.0236 - 0.0000i  -0.0236 + 0.0000i  -0.0008 + 0.0000i  -0.0008 - 0.0000i   0.0049 + 0.0000i   0.0049 + 0.0000i
 -0.0072 - 0.0030i  -0.0072 + 0.0030i  -0.0072 + 0.0030i  -0.0072 - 0.0030i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.5397 - 0.0000i  -0.5397 + 0.0000i   0.2347 + 0.0000i   0.2347 - 0.0000i  -0.0297 - 0.0000i  -0.0297 + 0.0000i  -0.0158 + 0.0000i  -0.0158 + 0.0000i
 -0.0002 - 0.0001i  -0.0002 + 0.0001i  -0.0002 + 0.0001i  -0.0002 - 0.0001i   0.0034 + 0.0000i   0.0034 - 0.0000i   0.0443 + 0.0000i   0.0443 - 0.0000i   0.3462 + 0.0000i   0.3462 - 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   0.0694 + 0.0000i   0.0694 + 0.0000i
 -0.0000 - 0.0000i  -0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 + 0.0000i   0.0018 + 0.0000i   0.0018 - 0.0000i   0.0251 + 0.0000i   0.0251 - 0.0000i  -0.0335 - 0.0000i  -0.0335 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i
  0.3002 + 0.1352i   0.3002 - 0.1352i   0.3002 - 0.1352i   0.3002 + 0.1352i  -0.1042 + 0.0000i  -0.1042 - 0.0000i   0.0212 + 0.0000i   0.0212 + 0.0000i  -0.0187 - 0.0000i  -0.0187 + 0.0000i  -0.0140 + 0.0000i  -0.0140 + 0.0000i  -0.0136 + 0.0000i  -0.0136 + 0.0000i
 -0.0026 - 0.0005i  -0.0026 + 0.0005i  -0.0026 + 0.0005i  -0.0026 - 0.0005i   0.3197 - 0.0000i   0.3197 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.1609 - 0.0000i  -0.1609 + 0.0000i  -0.1320 - 0.0000i  -0.1320 - 0.0000i  -0.0240 + 0.0000i  -0.0240 + 0.0000i
  0.0001 + 0.0000i   0.0001 - 0.0000i   0.0001 - 0.0000i   0.0001 + 0.0000i  -0.0278 - 0.0000i  -0.0278 + 0.0000i   0.0500 - 0.0000i   0.0500 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i  -0.4861 - 0.0000i  -0.4861 + 0.0000i  -0.1692 + 0.0000i  -0.1692 + 0.0000i

success! HW03_tors_flex_flutter_EXAMPLE finished successfully
```

Part D(b): zoomed view around U_F highlighting the first coalescing root and sign change in damping.

Results



Part D(c): flutter modeshapes/eigenvectors near U_F for the critical pair and higher modes.

Results

Part D discussion

The frequency/damping map in D(a) shows a single flexural–torsional pair converging as U increases; D(b) confirms the damping crosses zero at the same U_F, marking flutter onset. The mode snapshots in D(c) reveal that the critical pair mixes the first torsional with a higher flexural component—consistent with the weak but nonzero coupling seen in Part C. Because the damping slope near U_F is steep, small aerodynamic or stiffness changes will shift U_F noticeably, so the GVT correlation (Parts A–C) is essential for credible flutter prediction.

E.1 Flexural vibration analysis (extra credit)

Derive the equations of flexural vibration for frequencies and modeshapes.

Step

E.1 Step 1: Euler–Bernoulli governing PDE
Since slender wings obey Euler–Bernoulli theory, the transverse displacement $w(z,t)$ satisfies

$$EI\,w''''(z,t) + m\,\ddot{w}(z,t) = 0$$

with EI, m uniform along $0 \le z \le l$.

Step

E.1 Step 2: fixed–free boundary conditions
Since the root is clamped: $w(0,t) = 0$, $w'(0,t) = 0$. Since the tip is free: $w''(l,t) = 0$, $w'''(l,t) = 0$.

Step

E.1 Step 3: separation of variables
Since $w(z,t) = \hat{w}(z)e^{st}$, substitution gives the spatial ODE

$$\hat{w}'''' - \beta^4\,\hat{w} = 0, \qquad \beta^4 = \frac{ms^2}{EI}.$$

Applying the four BCs yields the characteristic relation

$$\cosh(\beta l)\cos(\beta l) + 1 = 0.$$

Results

E.1 Roots of $\cosh\beta l\cos\beta l + 1 = 0$

$$\beta_1 l = 1.8751, \quad \beta_2 l = 4.6941, \quad \beta_3 l = 7.8548, \quad \beta_4 l = 10.9955, \ldots$$

These agree with classical tabulations (e.g., Timoshenko & Young, *Vibration Problems in Engineering*, 5th ed.).

Step

E.1 Step 4: mode shapes and normalization

Since tip compatibility enforces

$$B_j = \frac{\sinh\beta_j - \sin\beta_j}{\cosh\beta_j + \cos\beta_j},$$

the jth mode shape is

$$W_j(z) = A_j\Big[(\cosh\beta_j z/l - \cos\beta_j z/l) - B_j(\sinh\beta_j z/l - \sin\beta_j z/l)\Big],$$

with $A_j = 1/\sqrt{l}$ to set unit modal mass: $\int_0^l mW_jW_k\,dz = \delta_{jk}$.

Results

E.1 Flexural natural frequencies

$$\omega_j = \beta_j^2\sqrt{\frac{EI}{m\,l^4}}, \qquad f_j = \frac{\omega_j}{2\pi}.$$

Results

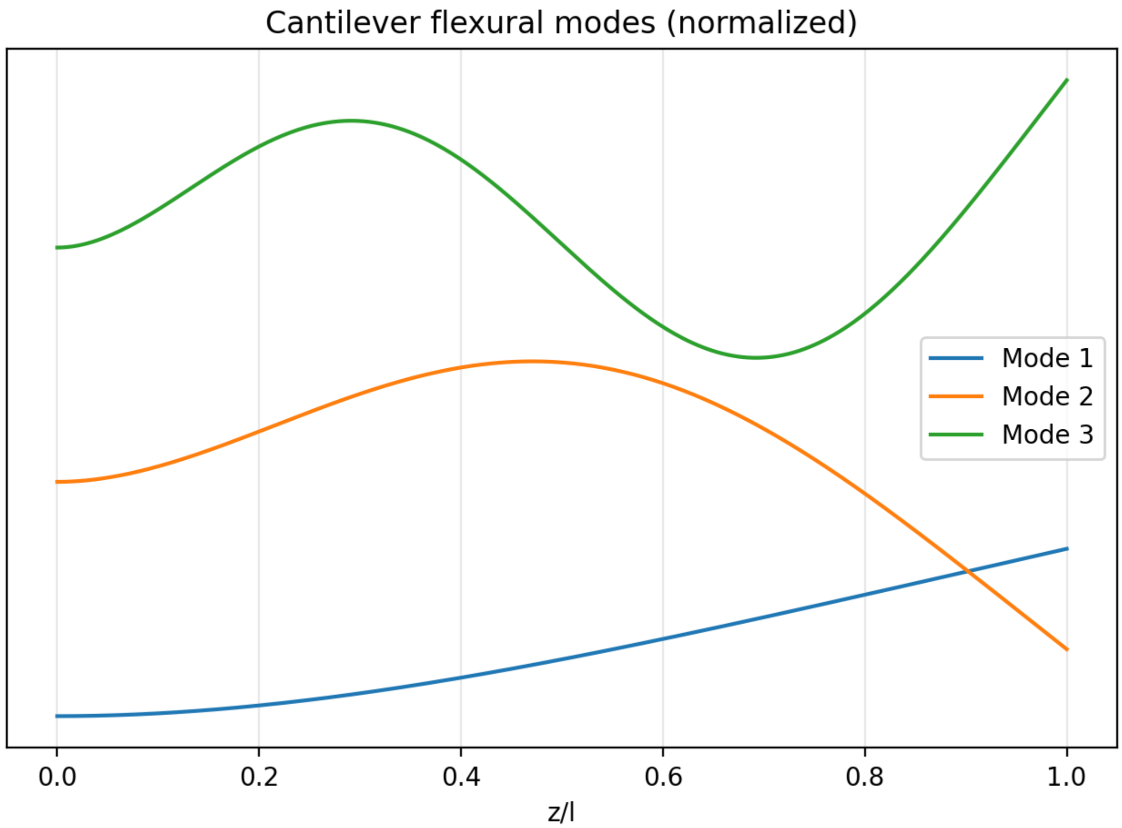
E.1 Python spot–check (unit beam, $l = 1$, $EI = m = 1$)

```
flexural betas: [1.87510407 4.69409113 7.85475744 10.99554073]
```

$$f_1 = \frac{\beta_1^2}{2\pi} = 0.560, \qquad f_2 = 2.645, \qquad f_3 = 6.864.$$

Computed with mpmath findroot on $\cosh\beta\cos\beta + 1 = 0$ (script available on request).

Results



E.1 visualization: first three cantilever flexural modes (normalized, offset for clarity).

E.2 Torsional vibration analysis (extra credit)

Derive the equations of torsional vibration for frequencies and modeshapes.

Step

E.2 Step 1: Saint–Venant torsion PDE

Since uniform shafts twist without warping, the twist $\phi(z,t)$ obeys

$$GJ\,\phi''(z,t) = I_0\,\ddot{\phi}(z,t),$$

where I_0 is the polar mass moment per unit span.

Step

E.2 Step 2: fixed–free boundary conditions

Since root is clamped: $\phi(0,t)=0$. Since tip is free (zero torque): $\phi'(l,t)=0$.

Step

E.2 Step 3: separation and eigenvalue condition

Since $\phi(z,t)=\hat{\phi}(z)e^{st}$, spatial equation $\hat{\phi}''+\gamma^2\hat{\phi}=0$ with $\gamma^2 = I_0 s^2/(GJ)$. Applying BCs gives $\cos(\gamma l)=0 \Rightarrow \gamma_j l = (2j-1)\pi/2$.

Results

E.2 Torsional modes and frequencies

$$\Phi_j(z) = B_j \sin(\gamma_j z), \qquad B_j = \sqrt{\frac{2}{l}}, \qquad \gamma_j = \frac{(2j-1)\pi}{2l},$$

$$\omega_j = \gamma_j\sqrt{\frac{GJ}{I_0}}, \qquad f_j = \frac{\omega_j}{2\pi}.$$

Fixed–free torsional spectrum follows directly from $\cos\gamma l = 0$ (see Blevins, *Formulas for Natural Frequency and Mode Shape*).

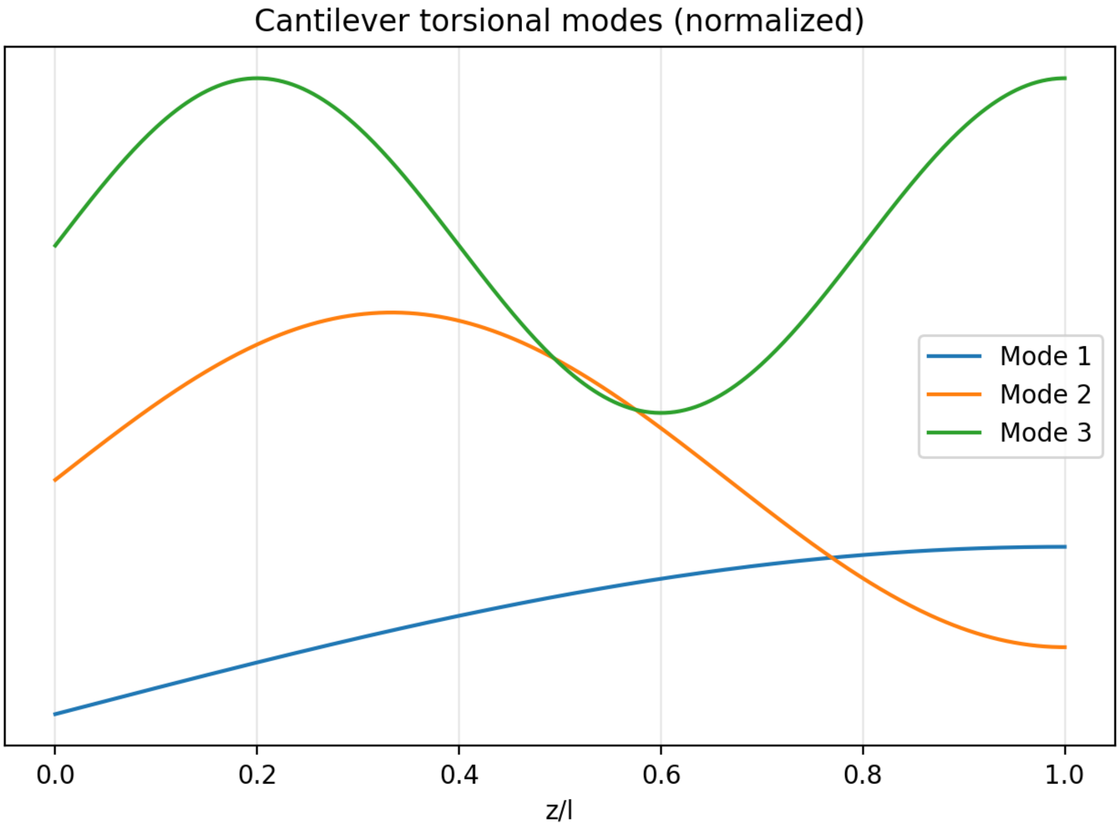
Results

E.2 Python spot–check (unit rod, $l=1$, $GJ=I_0=1$)

```
torsion gammas*l: [1.57079633 4.71238898 7.85398163]
```

$$f_1 = \frac{\gamma_1}{2\pi} = 0.250, \qquad f_2 = 0.750, \qquad f_3 = 1.250.$$

Results



E.2 visualization: first three cantilever torsional modes (normalized).

Appendix A: MATLAB Source Listings

```
%{
HW03 EXAMPLE
Torsion-flexure flutter of fixed-free wing
%}

%% Magic numbers overview (edit here first)
% Initialization: mm=1e-3 (m/mm), deg=180/pi (rad->deg), tol=1e-10 (zero cut)
% Aerodynamics: rho=1.225 (kg/m^3), a1=2*pi (lift-curve slope)
% Geometry/structure: c=0.4 m, m=3 kg, I0=0.0288 kg*m^2, fh=2 Hz, ft=5 Hz,
%   CPratio=-0.10, QPratio=0.30, L=2 m, scale=8, NW=4, NPhi=3
% Speed sweep: Ustart=0 m/s, Uend=12 m/s, UFrigid=8.244 m/s, UFread=8.34 m/s
% Discretization/animation: Nz=100, NU=1001, Nx=1e2, Nc=5, Nt=1e2, zmax=1.2,
%   VideoWriter FrameRate=10, Quality=100
% Feature toggles: ifA1flexure, ifA2torsion, ifGVT, ifFlutter, ifZoom,
%   ifFlutterModes, ifdisplayfV, ifanimation

%% initialization
clc                              % clear command window
clear                            % clear workspace
close all                        % close all plots
format compact
set(0,'DefaultFigureWindowStyle','docked')

nfig = 1;
mm   = 1e-3; deg = 180/pi;
tol  = 1e-10;                   % tolerance for discarding machine zero

%% DEFINE AERODYNAMIC PARAMETERS
rho = 1.225;                     % air density, 1.225 kg/m3
a1  = 2*pi;                      % ideal lift curve slope value

%% INPUT DATA
% ----- data for EXAMPLE -----
c       = 0.45;                   % airfoil chord, m
m       = 3.2;                     % mass, kg
I0      = 0.055;                % moment of inertia about the center of mass, kg*m
    ^2
fh      = 1.8;                     % plunge frequency, Hz
ft      = 5.3;                     % pitch frequency, Hz
CPratio = -10e-2;                % static offset as % of chord
QPratio = 35e-2;                 % aerodynamic offset ratio as % of chord
```

```
L       = 2.5;                        % span, m
scale   = 8;                        % scale up factor for plunge displ
NW      = 4;                        % number of flexural modes
NPhi    = 3;                        % number of torsional modes
Ustart  = 0;
Uend    = 12;                       % flutter range
UFrigid = 10.654;                    % flutter speed of rigid airfoil model from HW01
UFread  = 10.788;                     % flutter speed of structural dynamics wing
   model read on the plot

%% CALCULATED DATA
xCP     = CPratio*c;                % static offset value
xQP     = QPratio*c;                % aerodynamic offset calculated with % of chord
Ip      = I0+m*xCP^2;               % moment of inertia about the elastic center, kg*m
   ^2/m

% DISPLAY GIVEN DATA
display(' HW03 structural dynamics flutter analysis -- JC Vaught')
display(' input data')
fprintf(' air density rho=%4.3fkg/m^3 \n',rho)
fprintf(' c=%2.1fm, m=%2.1fkg/m, I0=%5.4fkg*m^2/m, L=%2.1fm \n',c,m,I0,L)
fprintf(' static offset xCP=%2.1f%%, %5.4fm \n',CPratio*100,xCP)
fprintf(' aerodynamic offset xQP=%2.1f%%, %5.4fm \n',QPratio*100,xQP)
fprintf(' rigid body frequencies fh=%2.1fHz, ft=%2.1fHz \n',fh,ft)
fprintf(' wing span L=%0.1f m \n', L)

%% choose what to do, to plot, and to display
% ifA1flexure = 0; % do not plot W modeshapes
ifA1flexure  = 0; % plot W modeshapes
ifA2torsion  = 0; % do not plot Phi modeshapes

ifGVT        = 0; % perform GVT analysis - Part C

ifFlutter    = 0; % perform flutter analysis - Part D.a

ifZoom       = 0; % provide calcuaotionjs/table - Part D.b

ifFlutterModes = 0; % plot flutter modes - Part D.c

ifdisplayfV  = 0; % do NOT display freq. and modeshapes

ifanimation  = 1; % do NOT animate GVT and flutter modeshapes
```

```matlab
%% CALCULATE UNCOUPLED ANGULAR FREQUENCIES wt, wh
wt = 2*pi*ft;
wh = 2*pi*fh;

%% discretize beam length
Nz = 100;
zL = linspace(0,L,Nz);
%% Section A1: FLEXURAL FREQUENCIES AND MODESHAPES
% calculate flexural eigenvalues
gL_guess = zeros(NW,1);
gL       = zeros(NW,1);
beta     = zeros(NW,1);

D = @(x)(cos(x)+1/cosh(x));      % D=0 equation to solve
for jW = 1:NW
    gL_guess(jW) = (2*jW-1)*pi/2;                % initial guess
    zW           = fzero(D,gL_guess(jW));        % solve equation
    gL(jW)       = zW;                           % store gL=gamma*L
end

% calculate wave number, angular frequency in rad/s, freq. in Hz
gW = gL/L;                         % flexural wave number
aW = sqrt(wh)/gW(1);               % flex const aW to match first freq fW(1)=fh
EI = m*aW^4;                       % flexural stiffness EI of the wing, N*m^2/m
wW = gW.^2*aW^2;                   % angular frequency, rad/sec
fW = wW/2/pi;                      % frequency in Hz

% Calculate and plot modeshapes
W    = zeros(Nz,NW);
AW   = zeros(1,NW);
beta = zeros(1,NW);
for jW = 1:NW
    AW(jW)   = 1/sqrt(L);
    beta(jW) = (sinh(gL(jW))-sin(gL(jW)))/(cosh(gL(jW))+cos(gL(jW)));
    W(:,jW)  = AW(jW)*(cosh(gW(jW)*zL)-cos(gW(jW)*zL)...
    -beta(jW)*(sinh(gW(jW)*zL)-sin(gW(jW)*zL)));
end

if ifA1flexure
    display(' ')
    display('Section A1: flexural vibration of a fixed-free beam')
    display(' (a) students should recall relevant formulae from')
```

```
    display(' in-class instruction and class notes')
    fprintf(' (b) NW=%1.0f \n',NW)
    display(gL',' roots of flexural characteristic equation')
    fprintf(' (c) flexural stiffness EI =%4.0f N*m^2/m \n',EI)
    display(' (d) flexural wavenumbers, natural freq. in rad/s and Hz')
    display([gW,wW,fW], ' gW,rad/m wW,rad/s fW,Hz')
    display(' (e) please see plot')
    figure; plot(zL,W); grid;
    title('Flexural modes of fixed-free beam', 'FontWeight', 'normal')
    xlabel('length, m'); ylabel('normalized modeshape')
end % ifplotW ends here

%% Section A2: TORSIONAL FREQUENCIES AND MODESHAPES
% torsional frequencies and modeshapes
gL  = zeros(NPhi,1);
cPhi = 2*L/pi*wt;                        % torsional wave speed
Phi = zeros(Nz,NPhi); gPhi = zeros(NPhi,1); B = zeros(NPhi,1);
for jPhi = 1:NPhi
    gL(jPhi)   = (2*jPhi-1)*pi/2;       % roots of torsional characteristic
   equation
    gPhi(jPhi) = gL(jPhi)/L;            % torsional wavenumber
    APhi(jPhi) = sqrt(2/L);             % mode amplitude, torsion
    Phi(:,jPhi)= APhi(jPhi)*sin(gPhi(jPhi)*zL); % modeshapes, torsion
    wPhi(jPhi) = cPhi*gPhi(jPhi);       % angular frequencies rad/s
end
GJ  = I0*cPhi^2;                         % torsional stiffness GJ of the wing, N*m
   ^2/m
fPhi= wPhi/(2*pi);                       % freq. in Hz
if ifA2torsion
    display(' ')
    display('Section A2: torsional vibration of a fixed-free beam')
    display(' (a) students should recall relevant formulae from')
    display(' in-class instruction and class notes')
    fprintf(' (b) NPhi=%1.0f \n',NPhi)
    display(gL',' roots of torsional characteristic equation')
    fprintf(' (c) torsional stiffness GJ =%4.1f N*m^2/m \n',GJ)
    display(' (d) torsional wavenumbers, natural freq. in rad/s and Hz')
    display([gPhi,wPhi',fPhi'], ' gPhi,rad/m wPhi,rad/s fPhi,Hz')
    display(' (e) please see plot')
    figure ; plot(zL,Phi);grid;
    title ('Torsional modes of fixed-free beam', 'FontWeight', 'normal')
    xlabel('length, m'); ylabel('normalized modeshape')
end % ifplotPhi ends here
```

```
%% Calculate modal matrices
N        = NW+NPhi;              % total number of modes
wW2      = wW.^2;
mWW      = m*diag(ones(NW,1));
kS_WW    = diag(wW2)*m;
wPhi2     = wPhi.^2;
mPhiPhi  = Ip*diag(ones(NPhi,1));
kS_PhiPhi= diag(wPhi2)*I0;
mWPhi    = zeros(NW,NPhi);
kS_WPhi  = zeros(NW,NPhi);
kS_PhiW  = zeros(NPhi,NW);
for pPhi = 1:NPhi
    for qW = 1:NW
        modePhi = @(x) APhi(pPhi)*sin(gPhi(pPhi)*x);
        modeW   = @(x) AW(qW)*((-cos(gW(qW)*x)+beta(qW)*sin(gW(qW)*x)...
                    +(1-beta(qW))/2*exp(gW(qW)*x)+(1+beta(qW))/2*exp(-gW(qW)*x)))
   ;
        Int     = @(x) modePhi(x).*modeW(x);
        mWPhi(qW,pPhi) = -m*xCP*integral(Int,0,L);
    end
end
mPhiW = mWPhi';

%% DEFINE STRUCTURAL MATRICES
% structural mass matrix
MS=[mWW mWPhi ;
mPhiW mPhiPhi];
% structural stiffness matrix
KS=[kS_WW kS_WPhi ;
kS_PhiW kS_PhiPhi];

%% Section C: GVT
if ifGVT
    display(' ')
    display(' Section C: GVT analysis')
    fprintf(' (a) NW=%1.0f, NPhi=%1.0f, N=%1.0f \n',NW,NPhi,N)
    %% CALCULATE EIGENVALUES:
    % use polyeig to get eigenvectors V and eigenvalues s
    [V_raw,s_raw] = polyeig(KS,0,MS);
    V_raw(1:NW,:)=V_raw(1:NW,:)*scale; % scale up plunge displ
    [V,s]=sort_norm_eig(V_raw,s_raw);
    %% EXTRACT FREQ. IMAG PART OF s
```

```
f=abs(imag(s))/(2*pi); % frequencies
%% DISPLAY FREQUENCY
display(f,' (b) coupled GVT frequencies f, Hz')
%% DISPLAY EIGENVECTORS
display(real(V),' GVT eigenvectors')
display(' students should write their own discussion')
display(' (c) please see plots')
%% 3D plotting of the GVT coupled modes v(x,z)
Nx=1e2; x=linspace(c/2,-c/2,Nx); % define x range
% select mode to plot
N=NW+NPhi;
Nmax=2*N; % max number of modes to plot
% Nmax=3; % for debugging
for mode=1:2:Nmax
    titleModeGVT=['GVT Mode ' num2str(mode) ', f=' num2str(f(mode),'%0.2f') '
 Hz' ];
    Vxz=zeros(Nx,Nz);
    for i=1:Nx
        Vxz(i,:)=W*V(1:NW,mode)-x(i)*Phi*V(NW+1:NW+NPhi,mode);
    end
    Vxz = real(Vxz); % eigenvectors are complex; use physical (real) part for
 plotting
    figure;
    surf(zL,x,Vxz); % surf plot
    % view(0,90) % top view
    % view(45,60) % rotated view
    view(16,84) % rotated view VG
    zlim([-2 2])
    set(gca,'Ydir','reverse')
    % set(gca,'Ydir','normal')
    shading interp
    axis equal
    xlim([0 L]);
    % ylim([-c/2 c/2]);
    xlabel('z, m'); ylabel('x, m');
    set(gca, 'FontSize', 12);
    tl=title({'GVT Coupled Modeshapes'; titleModeGVT});
    tl.FontWeight='normal'; tl.FontSize=12;
    %% 3D animation of the GVT coupled modes v(x,z)
    if ifanimation
        Nc=5; Nt=1e2; t=linspace(0,Nc/f(mode),Nt); % define time range over 5
 cycles
        animation = VideoWriter(titleModeGVT);
```

```
            animation.FrameRate = 10; %% time interval between two frame
            animation.Quality =100; open(animation);
            Vxz=zeros(Nx,Nz);
            zmax=1.2;
            for k=1:Nt
                for i=1:Nx
                    Vxz(i,:)=(W*V(1:NW,mode)-x(i)*Phi*V(NW+1:NW+NPhi,mode))...
                    *exp(1i*2*pi*f(mode)*t(k));
                end
                figure(100) % figure(100) is used to generate the animation
                surf(zL,x,real(Vxz)); % surf plot
                % view(45,60) % rotated view
                view(16,84) % rotated view VG
                set(gca,'Ydir','reverse')
                shading interp
                axis equal
                xlim([0 L]); ylim([-c/2 c/2]); zlim([-zmax zmax]);
                xlabel('z, m'); ylabel('x, m');
                set(gca, 'FontSize', 12);
                tl=title({'GVT Coupled Modeshapes'; titleModeGVT});
                tl.FontWeight='normal'; tl.FontSize=12;
                thisFrame = getframe(gcf);
                writeVideo(animation, thisFrame);
            end
            close(animation);
        end % ifanimation ends here
    end % mode loop ends here
end % ifGVT ends here

%% FLUTTER ANALYSIS section
if ifFlutter
    ifplot = 1;                                  % plot flutter diagram

    %% DEFINE AERODYNAMIC LIFT FUNCTION
    a1 = 2*pi;                                   % ideal lift curve slope value
    L0 = @(UU) rho*UU^2/2*c*a1;                  % Lift function for a generic speed
   UU

    %% DEFINE AIRSPEED RANGE
    NU = 1001;
    U  = linspace(Ustart, Uend, NU);             % airspeed range
    UF = UFread;
    if ifZoom
```

```
        eps = 1e-2;
        U   = [1-eps 1 1+eps]*UF;            % zoom around UF
    end
    if ifFlutterModes
        U = UF;                              % plot flutter modes at flutter speed
    end
    NU = length(U);
    if NU < 10
        ifplot = 0;
    end

    %% LOOP OVER ALL AIR SPEEDS
    r     = zeros(2*N,NU);
    v     = zeros(N,2*N,NU);
    f     = zeros(2*N,NU);
    z     = zeros(2*N,NU);
    sigma = zeros(2*N,NU);
    roots = zeros(2*N,NU);

    for jU = 1:NU
        %% DEFINE FLUTTER MATRICES
        MA      = zeros(N,N);                     % aerodynamic mass matrix MA=0
        kA_WW   = zeros(NW,NW);
        kA_PhiW = zeros(NPhi,NW);
        for pPhi = 1:NPhi
            for qW = 1:NW
                modePhi = @(x) APhi(pPhi)*sin(gPhi(pPhi)*x);
                modeW   = @(x) AW(qW)*((-cos(gW(qW)*x)+beta(qW)*sin(gW(qW)*x)...
                        +(1-beta(qW))/2*exp(gW(qW)*x)+(1+beta(qW))/2*exp(-gW(qW)*
x)));
                Int     = @(x) modePhi(x).*modeW(x);
                kA_WPhi(qW,pPhi) = L0(U(jU))*integral(Int,0,L);
            end
        end
        for pPhi = 1:NPhi
            for qPhi = 1:NPhi
                modePhi = @(x,j) APhi(j)*sin(gPhi(j)*x);
                Int     = @(x) modePhi(x,pPhi).*modePhi(x,qPhi);
                kA_PhiPhi(qPhi,pPhi) = -L0(U(jU))*xQP*integral(Int,0,L);
            end
        end

        % aerodynamic stiffness matrix
```

```
    KA = [kA_WW kA_WPhi ;
          kA_PhiW kA_PhiPhi];
    M  = MS+MA;                      % system mass matrix
    K  = KS+KA;                      % system stiffness matrix

    %% CALCULATE EIGENVALUES:
    % use polyeig to get eigenvectors V and eigenvalues s
    [V_raw,s_raw] = polyeig(K,0,M);
    V_raw(1:NW,:) = V_raw(1:NW,:)*scale; % scale up plunge displ
    [V,s] = sort_norm_eig(V_raw,s_raw);

    %% EXTRACT DAMPING AND FREQ. FROM REAL AND IMAG PARTS OF s
    ff  = abs(imag(s))/(2*pi);           % frequencies
    zz  = -real(s)./abs(s); zz=zz.*(abs(zz)>tol); % damping
    sig = real(s); sig=sig.*(abs(sig)>tol);

    %% STORE EIGENVALUES AND EIGENVECTOR
    r(:,jU)      = s;
    v(:,:,jU)    = V;
    f(:,jU)      = ff(:);
    z(:,jU)      = zz(:);
    sigma(:,jU)  = sig(:);
end

%% DISPLAY FREQ, DAMPING AND MODESHAPES
if ifZoom
    for jU = 1:NU
        display(' ')
        display([' U=' num2str(U(jU)) 'm/s']);
        display(f(:,jU)',' f, Hz')
        display(z(:,jU)'*100,' z %')
        display(v(:,:,jU),' eigenvector V');
    end
end

%% PLOT FREQUENCY AND DAMPING VS AIRSPEED
if ifplot
    figure
    subplot(2,1,1);
    plot(U,f,'.r');
    title(['xCP=' num2str(CPratio*1e2) '% ' ...
           'xQP=' num2str(QPratio*1e2) '%'],'FontSize', 10, 'FontWeight', '
normal')
```

```
        xlabel('U, m/s'); ylabel('f, Hz');
        fmax=ceil(max(max(f))); ylim([0 fmax]);
        xlim([U(1) U(NU)]);
        hold on
        subplot(2,1,2);
        plot(U,z*1e2,'.g');
        xlim([U(1) U(NU)]);
        xlabel('U, m/s'); ylabel('\zeta, %');
        ymax=15; ylim([-ymax ymax]);
        grid on
        hold on
    end

    %% coupled flutter mode plotting v(x,z)
    if ifFlutterModes
        Nx=1e2; x=linspace(c/2,-c/2,Nx); % define x range
        N=NW+NPhi;
        Nmax=2*N; % max number of modes to plot
        for mode=1:2:Nmax
            titleModeFlutter=['Flutter Mode ' num2str(mode) ];
            Vxz=zeros(Nx,Nz);
            for i=1:Nx
                Vxz(i,:)=W*V(1:NW,mode)-x(i)*Phi*V(NW+1:NW+NPhi,mode);
            end
            Vxz = real(Vxz); % keep only real part for visualization
            figure;
            surf(zL,x,Vxz); % surf plot
            view(16,84) % rotated view VG
            zlim([-2 2])
            set(gca,'Ydir','reverse')
            shading interp
            axis equal
            xlim([0 L]);
            xlabel('z, m'); ylabel('x, m');
            set(gca, 'FontSize', 12);
            tl=title({['Flutter Modeshapes at UF= ' num2str(UF,'%0.3f') ' m/s'];
titleModeFlutter});
            tl.FontWeight='normal'; tl.FontSize=12;

            %% 3D animation of the coupled modes v(x,z)
            if ifanimation
                Nc=5; Nt=1e2; t=linspace(0,Nc/f(mode),Nt); % define time range
over 5 cycles
```

```
                animation = VideoWriter(titleModeFlutter);
                animation.FrameRate = 10; %% time interval between two frame
                animation.Quality =100; open(animation);
                Vxz=zeros(Nx,Nz);
                zmax=1.2;
                for k=1:Nt
                    for i=1:Nx
                        Vxz(i,:)=(W*V(1:NW,mode)-x(i)*Phi*V(NW+1:NW+NPhi,mode))
   ...
                            *exp(1i*2*pi*f(mode)*t(k));
                    end
                    figure(100) % figure(100) is used to generate the animation
                    surf(zL,x,real(Vxz)); % surf plot
                    view(16,84) % rotated view VG
                    set(gca,'Ydir','reverse')
                    shading interp
                    axis equal
                    xlim([0 L]); ylim([-c/2 c/2]); zlim([-zmax zmax]);
                    xlabel('z, m'); ylabel('x, m');
                    set(gca, 'FontSize', 12);
                    tl=title({['Flutter Modeshapes at UF= ' num2str(UF)];
   titleModeFlutter});
                    tl.FontWeight='normal'; tl.FontSize=12;
                    thisFrame = getframe(gcf);
                    writeVideo(animation, thisFrame);
                end
                close(animation);
            end % ifanimation ends here
        end % mode loop ends here
    end % ifFlutterModes loop ends here
end % ifFlutter ends here

%% finish
display(' ')
display (['success! ' mfilename ' finished successfully'])
```

```
function [X_sorted_normalized,e_sorted] = sort_norm_eig(X,e)
%{
X(N,Ne) = matrix of Ne eigenvectors each of N dofs
e(Ne) = row of Ne eigenvalues
Procedure:
sort eigenvalues e in magnitude order and stores into es
reorder the eigenvector X and stores into Xs
normalize the sorted eigenvectors Xs to get Xsn
```

```
such that the largest element in each eigenvector is = 1
%}
N=size(X,1); Ne=size(X,2); % pick sizes N, Ne
e_abs=abs(e); % pick up abs values
% [~,Is]=sort(e_abs,'descend'); % sort in descending order
[~,Is]=sort(e_abs,'ascend'); % sort in ascending order
% Is contains the sorted indices
%% store sorted eigenvalues and eigenvectors
e_sorted=zeros(1,Ne); Xs=zeros(N,Ne);
for ne=1:Ne; e_sorted(ne)=e(Is(ne)); Xs(:,ne)=X(:,Is(ne)); end %
% display(e_sorted,'sorted eigenvalues')
% display(Xs,'sorted eigenvectors')
%% normalize eigenvectors to make +ve the largest element in each column
Xabs=abs(Xs);
[~,IX]=max(Xabs,[],1);
% allocate full N-by-Ne so we can store every sorted eigenvector
X_sorted_normalized=zeros(N);
for j=1:Ne;
    scale=sign(Xs(IX(j),j))*max(abs(Xs(:,j)));
    X_sorted_normalized(:,j)=Xs(:,j)/scale;
end

end % function ends her
```