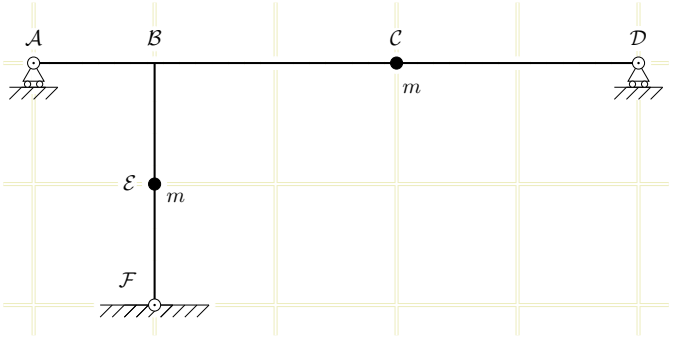
3 DOF System

Giacomo Boffi

1 3 DOF System

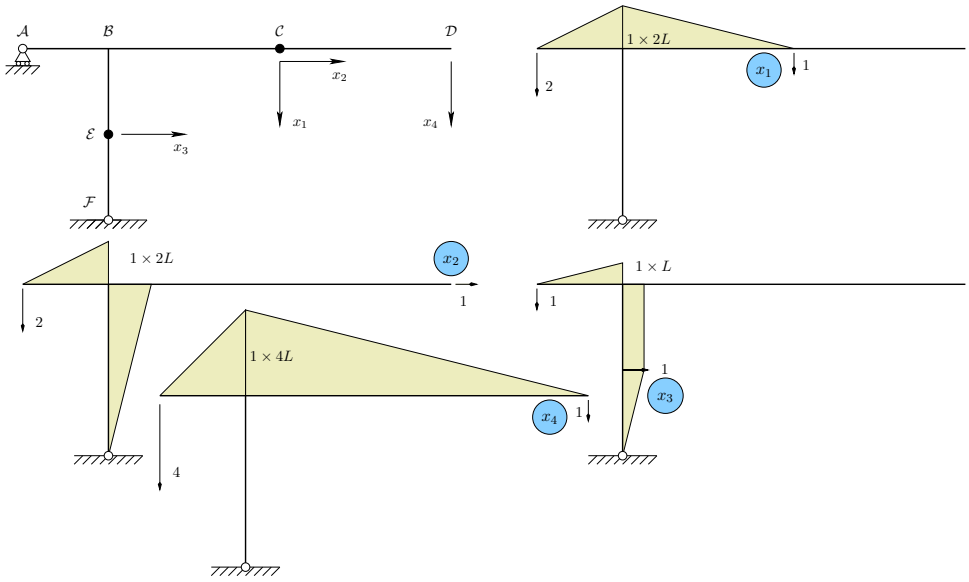


The 3 DOF Dynamic System

The structure above is statically overdetermined, to compute the stiffness matrix and, later, the influence matrix for the impressed displacement we have to introduce an additional degree of freedom, so that the system becomes statically determined.

Because we need to compute the influence matrix for the displacement of the roller in $\mathcal{D}$, we remove the roller and compute the flexibility matrix for the 4 DOF system (the 3 dynamic DOF's + the support displacement) using the PVD.

First of all, with an external unit load applied to each of the DOF's we compute (a) the reaction of the roller in $\mathcal{A}$ and (b) the bending moments:



Bending Moments for Different Load Conditions

We have 5 different fields, listed with the correct orientation:

1. $\overline{\mathcal{AB}}$, $0\le x\le L$
2. $\overline{\mathcal{CB}}$, $0\le x\le 2L$
3. $\overline{\mathcal{DC}}$, $0\le x\le 2L$
4. $\overline{\mathcal{FE}}$, $0\le x\le L$
5. $\overline{\mathcal{EB}}$, $0\le x\le L$

and for every field we write down, in `L`, the respective length and in each row of `M` (one row for each DOF) the 5 expression for the bending moment on each of the 5 fields, using the univariate polynomial class that is offered by `numpy`, e.g., `p((2,0))` means `2*x**1 + 0*x**0`:

```
L = [1, 2, 2, 1, 1]
M = [[p((2,0)), p((1,0)), p((0,0)), p((0,0)), p((0,0)),],
     [p((2,0)), p((0,0)), p((0,0)), p((1,0)), p((1,1)),],
     [p((1,0)), p((0,0)), p((0,0)), p((1,0)), p((0,1)),],
     [p((4,0)), p((1,2)), p((1,0)), p((0,0)), p((0,0)),],]
```

In the following `M` plays a double role, in one loop it represents the bending moments and in the other loop the curvatures; in the inner loop, for a fixed pair of moments and curvatures, we iterate over the different fields of integration, sum the diverse contributions and eventually we have one coefficient of the flexibility matrix.

That done, we invert the flexibility to have the stiffness matrix for the DOF augmented system.

```
F = array([[sum(poly_int(μ, χ, λ) for μ, χ, λ in zip(μ_s, χ_s, L))
            for χ_s in M] for μ_s in M])
K = np.linalg.inv(F)

platex('\\overline{\\boldsymbol_F}_=_\\frac{L^3}{6EJ}\\,',
       mat2lat(F*6,     fmt='%6.2f'), ',')
platex('\\overline{\\boldsymbol_K}_=_\\frac{3EJ}{532L^3}\\,',
       mat2lat(K*532/3, fmt='%6.1f'), '.')
```

$$\overline{\boldsymbol{F}} = \frac{L^3}{6EJ}\begin{bmatrix} 24.00 & 8.00 & 4.00 & 56.00 \\ 8.00 & 24.00 & 15.00 & 16.00 \\ 4.00 & 15.00 & 10.00 & 8.00 \\ 56.00 & 16.00 & 8.00 & 160.00 \end{bmatrix},$$

$$\overline{\boldsymbol{K}} = \frac{3EJ}{532L^3}\begin{bmatrix} 268.0 & -120.0 & 144.0 & -89.0 \\ -120.0 & 816.0 & -1192.0 & 20.0 \\ 144.0 & -1192.0 & 1856.0 & -24.0 \\ -89.0 & 20.0 & -24.0 & 37.0 \end{bmatrix}.$$

From the stiffness matrix, using partitioning, we have the structural stiffness matrix $\boldsymbol{K}_{ss}$ etc, the structural mass matrix is proportional to a 3-by-3 unit matrix,

```
Mss = np.eye(3)
platex('{\\boldsymbol_M}_{ss}_=__m\\,', mat2lat(Mss, fmt='%8.2f'), '.')
Kss, Ksg = K[:-1,:-1], K[:-1,-1:]
Kgs, Kgg = K[-1:,:-1], K[-1:,-1:]
platex('{\\boldsymbol_K}_{ss}_=_\\frac{3EJ}{532L^3}\\,',
       mat2lat(Kss*532/3,fmt='%8.2f'), ',\\qquad',
       '{\\boldsymbol_K}_{sg}_=_\\frac{3EJ}{532L^3}\\,',
       mat2lat(Ksg*532/3,fmt='%8.2f'), ',')
platex('{\\boldsymbol_K}_{gs}_=_\\frac{3EJ}{532L^3}\\,',
       mat2lat(Kgs*532/3,fmt='%9.4f'), ',\\qquad',
       '{\\boldsymbol_K}_{gg}_=_\\frac{3EJ}{532L^3}\\,',
       mat2lat(Kgg*532/3,fmt='%8.2f'), ',')
```

$$\boldsymbol{M}_{ss} = m\begin{bmatrix} 1.00 & 0.00 & 0.00 \\ 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 1.00 \end{bmatrix}.$$

$$\boldsymbol{K}_{ss} = \frac{3EJ}{532L^3}\begin{bmatrix} 268.00 & -120.00 & 144.00 \\ -120.00 & 816.00 & -1192.00 \\ 144.00 & -1192.00 & 1856.00 \end{bmatrix}, \qquad \boldsymbol{K}_{sg} = \frac{3EJ}{532L^3}\begin{bmatrix} -89.00 \\ 20.00 \\ -24.00 \end{bmatrix},$$

$$\boldsymbol{K}_{gs} = \frac{3EJ}{532L^3}\begin{bmatrix} -89.0000 & 20.0000 & -24.0000 \end{bmatrix}, \qquad \boldsymbol{K}_{gg} = \frac{3EJ}{532L^3}\begin{bmatrix} 37.00 \end{bmatrix},$$

eventually we compute the eigenvalues and the eigenvectors of the dynamic system, print them

```
om2, Psi = eigh(Kss, Mss)

platex('{\\boldsymbol\\Lambda}_=_\\omega_0^2\\,', mat2lat(om2[None,:]))
platex('{\\boldsymbol\\Psi}_=_', mat2lat(Psi))
```

$$\Lambda = \omega_0^2\begin{bmatrix} 0.18847 & 1.44102 & 14.94946 \end{bmatrix}$$

$$\Psi = \begin{bmatrix} 0.09677 & 0.99225 & -0.07791 \\ 0.83646 & -0.03865 & 0.54666 \\ 0.53942 & -0.11807 & -0.83372 \end{bmatrix}$$

as well as a check of the ortogonality of the eigenvectors:

```
platex('{\\boldsymbol\\Psi}^T_{\\boldsymbol_M}_{ss}_{\\boldsymbol\\Psi}_=_m\\,',
       mat2lat(Psi.T@Mss@Psi),',')
platex('{\\boldsymbol\\Psi}^T_{\\boldsymbol_K}_{ss}_{\\boldsymbol\\Psi}_=_\\frac{EJ}{L^3}\\,',
       mat2lat(Psi.T@Kss@Psi),'.')
```

$$\Psi^T \boldsymbol{M}_{ss} \Psi = m \begin{bmatrix} 1.00000 & 0.00000 & 0.00000 \\ 0.00000 & 1.00000 & 0.00000 \\ 0.00000 & 0.00000 & 1.00000 \end{bmatrix},$$
$$\Psi^T \boldsymbol{K}_{ss} \Psi = \frac{EJ}{L^3} \begin{bmatrix} 0.18847 & 0.00000 & -0.00000 \\ 0.00000 & 1.44102 & 0.00000 \\ -0.00000 & -0.00000 & 14.94946 \end{bmatrix}.$$

2 Modal Response for EQ Excitation

With

$$\boldsymbol{e} = \{0 \quad 1 \quad 1\}^T$$

the equation of motion can be written

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{K}\boldsymbol{x} = -\boldsymbol{M}\boldsymbol{e}\delta\omega_0^2 \sin\omega_0 t = -m\boldsymbol{e}\delta\omega_0^2 \sin\omega_0 t,$$

with $\boldsymbol{x} = \Psi\boldsymbol{q}$ and premultiplying each term by Ψ^T we have (taking into account the ortogonality relationships)

$$m\ddot{\boldsymbol{q}} + \frac{EJ}{L^3}\Lambda\boldsymbol{q} = -m\Psi^T\boldsymbol{e}\delta\omega_0^2 \sin\omega_0 t.$$

Simplifying and writing the individual equations,

$$\ddot{q}_i + \lambda_i^2\omega_0^2 q_i = -\boldsymbol{\psi}_i^T\boldsymbol{e}\delta\omega_0^2 \sin\omega_0 t = -\delta_i\omega_0^2 \sin\omega_0 t,$$

we have, for rest initial conditions, the individual integrals

$$q_i(t) = \frac{\delta_i}{1-\lambda_i^2}\left(\sin\omega_0 t - \frac{1}{\lambda_i}\sin\lambda_i\omega_0 t\right)$$

```
e = array((0.0, 1.0, 1.0))
Lambda = om2
lambda_i = sqrt(Lambda)
delta = 1.0
delta_i = Psi.T@e*delta
C_i = delta_i/(1.0-lambda_i**2)
# We have no use for the modal velocities (qv0) but we compute them all the same
qd0 = [lambda a, c=c, l=l:c*(sin(a)-sin(l*a)/l) for c, l in zip(C_i, lambda_i)]
qv0 = [lambda a, c=c, l=l:c*(cos(a)-cos(l*a)/1) for c, l in zip(C_i, lambda_i)]
```

For the free vibration response, we compute `sn` and `cn`, the sines and cosines at $\omega_0 t = 2\pi$ for the different modal frequencies, then the modal displacement and velocities at $\omega_0 t = 2\pi$ and eventually the coefficients, from the solution of

$$\begin{bmatrix} s_i & c_i \\ +\lambda_i c_i & -\lambda_i s_i \end{bmatrix} \begin{Bmatrix} A_i \\ B_i \end{Bmatrix} = \begin{Bmatrix} q_i(2\pi) \\ \dot{q}_i(2\pi) \end{Bmatrix}, \qquad s_i = \sin 2\lambda_i\pi,\ c_i = \cos 2\lambda_i\pi;$$

```
sn, cn = sin(2*pi*lambda_i), cos(2*pi*lambda_i)
d2pi, v2pi = -C_i*sn/lambda_i, C_i*(1-cn)
A_i, B_i = sn*d2pi+cn*v2pi/lambda_i, cn*d2pi-sn*v2pi/lambda_i

qd1 = [lambda a,A=A,B=B,l=l:   A*sin(l*a)+  B*cos(l*a) for A,B,l in zip(A_i,B_i,lambda_i)]
qv1 = [lambda a,A=A,B=B,l=l: l*A*cos(l*a)-l*B*sin(l*a) for A,B,l in zip(A_i,B_i,lambda_i)]
```

2.1 Analytic espressions of the q_i

We can compute the analytic expressions of the modal responses:

```
for i in (0, 1, 2):
    l = lambda_i[i]
    qqq = r'q_%d(t)_=_'%(i+1)
    beg = r'\begin{cases}'
    eq1 = r'%+f\,(\sin\omega_0t-\frac{1}{%f}\sin%f\,\omega_0t).&'%(C_i[i], l, l)
    in1 = r'\qquad0\le\omega_0t\le2\pi\\'
    eq2 = r'%+f\,\sin%f\omega_0t_%+f\,\cos%f\,\omega_0t,&'%(A_i[i],l,B_i[i],l)
    in2 = r'\qquad2\pi\le\omega_0t.'
    end = r'\end{cases}'
```

```
    platex(qqq,beg,eq1,in1,eq2,in2,end)
```

$$q_1(t)=\begin{cases}+1.695415(\sin\omega_0 t-\frac{1}{0.434134}\sin 0.434134\omega_0 t). & 0\le\omega_0 t\le 2\pi\\ -7.480879\sin 0.434134\omega_0 t-1.570455\cos 0.434134\omega_0 t, & 2\pi\le\omega_0 t.\end{cases}$$

$$q_2(t)=\begin{cases}+0.355355(\sin\omega_0 t-\frac{1}{1.200424}\sin 1.200424\omega_0 t). & 0\le\omega_0 t\le 2\pi\\ -0.205298\sin 1.200424\omega_0 t-0.281779\cos 1.200424\omega_0 t, & 2\pi\le\omega_0 t.\end{cases}$$

$$q_3(t)=\begin{cases}+0.020579(\sin\omega_0 t-\frac{1}{3.866453}\sin 3.866453\omega_0 t). & 0\le\omega_0 t\le 2\pi\\ -0.001766\sin 3.866453\omega_0 t+0.003960\cos 3.866453\omega_0 t, & 2\pi\le\omega_0 t.\end{cases}$$

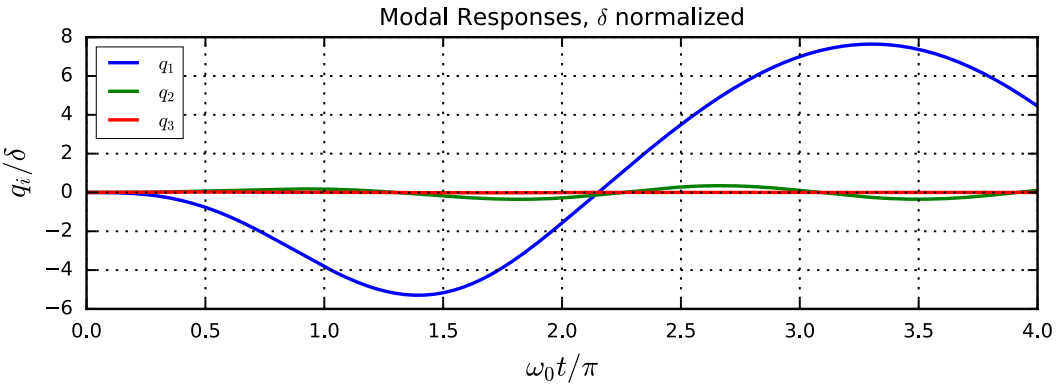
2.2 Plotting the q_i

First of all, we need a time axis, here `a` for adimensional time, $a=\omega_0 t$. The sampling is such that `h = pi/100`, and the dimension of the array, that contains both extremes, is `401` — Next, the array `q`, its size `3 x 401`, with the modal responses; we have to be careful because there are two different intervals of definition for the analityc definition of the response function.

```
a = linspace(0, 4*pi, 401)
q = array([np.where(a<pi*2, qd0[i](a), qd1[i](a)) for i in range(3)])
```

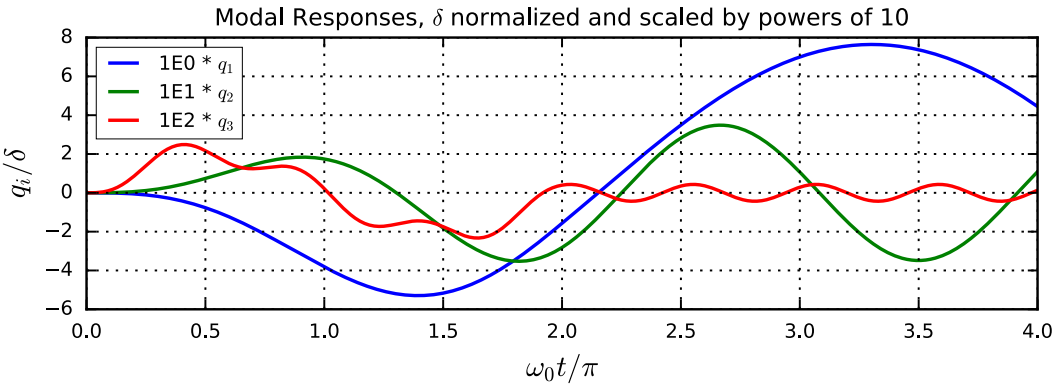
and here it is the actual plot, first without scaling, to appreciate the different amplitudes of the modal responses

```
lines = plt.plot(a/pi, q.T)
for i, l in enumerate(lines): l.set_label('$q_%d$'%(i+1))
plt.title('Modal_Responses,_$\\delta$_normalized')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$q_i/\delta$', fontsize='large')
plt.legend(loc='best');
```



and then with successive modes scaled by increasing powers of ten, to appreciate the details of the response for higher modes...

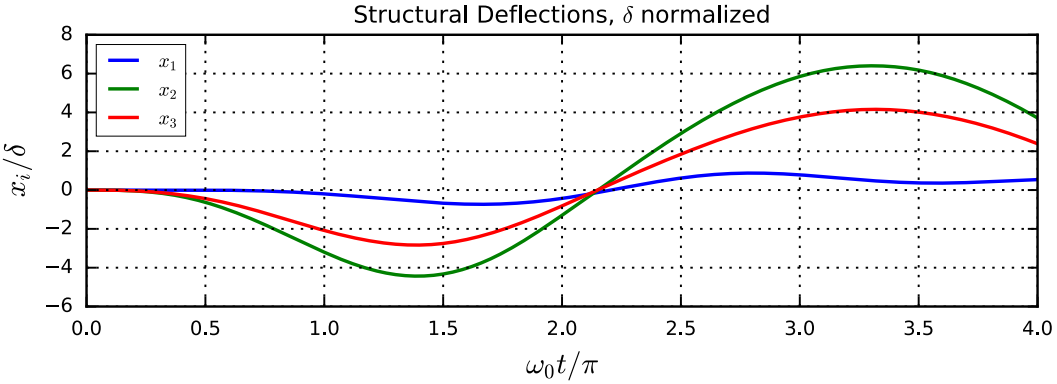
```
factors = array((1,10,100))
lines = plt.plot(a/pi, q.T*factors)
for i, l in enumerate(lines): l.set_label('1E%d_*_$q_%d$'%(i, i+1))
plt.title('Modal_Responses,_$\\delta$_normalized_and_scaled_by_powers_of_10')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$q_i/\delta$', fontsize='large')
plt.legend(loc='best');
```



2.3 Plotting the x_i

This is easy as the line of code `x = (Psi@q)` ...

```
x = (Psi@q)
lines = plt.plot(a/pi, x.T)
for i, l in enumerate(lines): l.set_label('$x_%d$'%(i+1))
plt.title('Structural_Deflections,_$\\delta$_normalized')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$x_i/\delta$', fontsize='large')
plt.legend(loc='best');
```



3 Numerical Integration

Using the prescribed *Constant Acceleration Algorithm*, we use the same time step used in the time array `a` to compute all the (matricial) constants, etc.

```
h = pi/100 # time step
dK = Kss + 4*Mss/h/h # the modified stiffness
dF = np.linalg.inv(dK) # ... and its inverse
# these are the v0, a0 coefficients to correct dp, theincremental load
dC, dM = 4*Mss/h, 2*Mss
f = np.where(a<2*pi, -sin(a), 0.0) # the load vector
```

Let's initialize the different vectors, as well as the containers with the output

```
# initial values
x0, v0, p0 = np.zeros(3), np.zeros(3), f[0]
a0 = np.linalg.inv(Mss) @ (e*p0-Kss@x0)

# initialize the deflections, velocities and accelerations
X, V, A = [x0], [v0], [a0]
```

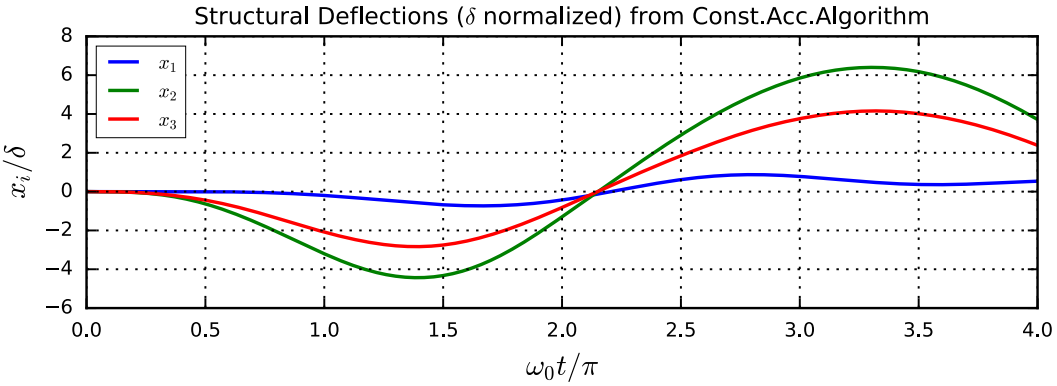
Finally, the integration proper

```
for p0, p1 in zip(f[:-1], f[+1:]):
    dx = dF@(e*(p1-p0)+dC@v0+dM@a0)
    x0 = dx + x0
    v0 = 2*dx/h - v0
    a0 = e*p1-Kss@x0 # I have omitted Mss^{-1}, that is a unit matrix
    X.append(x0), V.append(v0), A.append(a0)
```

Now, we have just to plot our results, looking at them and comparing with the analytic solution it seems that we have a good agreement,

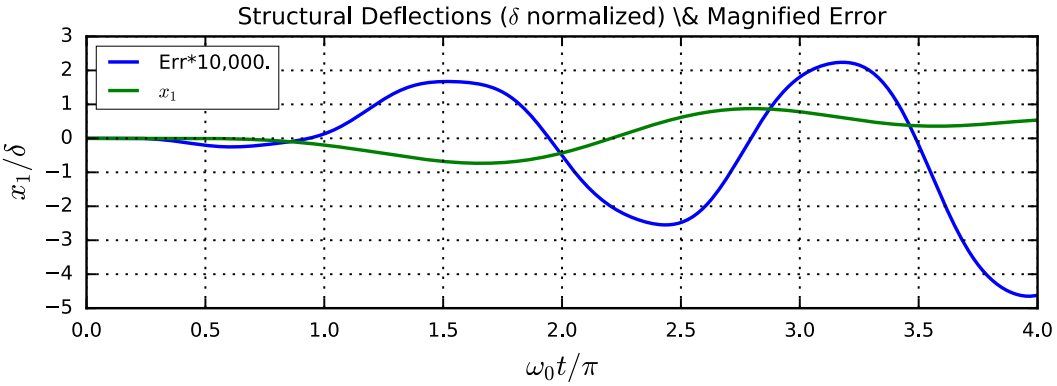
```
X = array(X)

lines = plt.plot(a/pi, X)
for i, l in enumerate(lines): l.set_label('$x_%d$'%(i+1))
plt.title('Structural_Deflections_($\\delta$_normalized)_from_Const.Acc.Algorithm')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$x_i/\delta$', fontsize='large')
plt.legend(loc='best');
```



but we can also plot just `x_1` and the difference between the analytic solution and the numerical one to appreciate how good is the numerical solution.

```
plt.plot(a/pi, 10000*(X[:,0]-x[0]), label='Err*10,000.')
plt.plot(a/pi, x[0], label='$x_1$')
plt.title('Structural_Deflections_($\\delta$_normalized)_\\&_Magnified_Error')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$x_1/\delta$', fontsize='large')
plt.legend(loc='best');
```



4 Support Motion

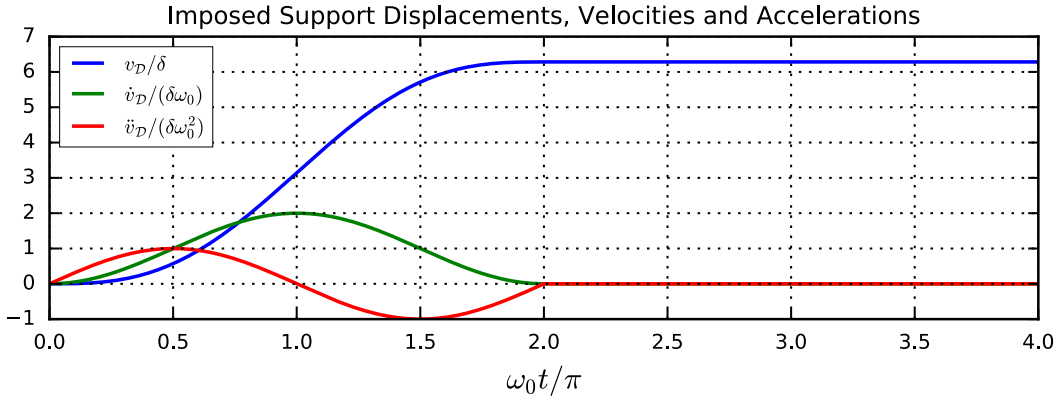
The structure is at rest when it is subjected to a vertical motion of the support in $\mathcal{D}$,

$$v_{\mathcal{D}}(t)=\delta\begin{cases}0 & \omega_0 t<0,\\ \omega_0 t-\sin\omega_0 t & 0\le\omega_0 t\le 2\pi,\\ 2\pi\omega_0 & 2\pi<\omega_0 t.\end{cases}$$

1. Plot $v_{\mathcal{D}}(t)$, $\dot{v}_{\mathcal{D}}(t)$ and $\ddot{v}_{\mathcal{D}}(t)$ in the interval $0\le\omega_0 t\le 4\pi$.
2. Compute and plot the total vertical displacement of the mass in $\mathcal{C}$, $v_{\mathcal{C}}(t)$ in the same time interval.

4.1 Plot of the displacements, velocities and accelerations.

```
plt.plot(a/pi, np.where(a<2*pi, a-sin(a), 2*pi), label=r'$v_{\cal D}/\delta$')
plt.plot(a/pi, np.where(a<2*pi, 1-cos(a),    0), label=r'$\dot{v}_{\cal D}/(\delta\omega_0)$')
plt.plot(a/pi, np.where(a<2*pi,   sin(a),    0), label=r'$\ddot{v}_{\cal D}/(\delta\omega_0^2)$')
plt.title('Imposed Support Displacements, Velocities and Accelerations')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.legend(loc='best');
```



4.2 Modal Response

Previously we had $\boldsymbol{e}=\{011\}^T$, now it is $\boldsymbol{e}=\{7/20\,2/20\,1/20\}^T$ (as can be seen from the last column of the 4×4 flexibility matrix) *and this is the only difference between this part of the problem and the previous one.*

```
e = F[:-1,-1]/F[-1,-1]
print(F*6,'/ 6 =\n\n',e*20, '/ 20 = e')
```

```
[[  24.    8.    4.   56.]
 [   8.   24.   15.   16.]
 [   4.   15.   10.    8.]
 [  56.   16.    8.  160.]]  / 6 = F

 [ 7.  2.  1.] / 20 = e
```

The equation of motion can be written

$$\boldsymbol{M}\ddot{\boldsymbol{x}}+\boldsymbol{K}\boldsymbol{x}=-\boldsymbol{M}\boldsymbol{e}\delta\omega_0^2\sin\omega_0 t=-m\boldsymbol{e}\delta\omega_0^2\sin\omega_0 t,$$

with $\boldsymbol{x}=\Psi\boldsymbol{q}$ and premultiplying each term by Ψ^T we have (taking into account the ortogonality relationships)

$$m\ddot{\boldsymbol{q}}+\frac{EJ}{L^3}\Lambda\boldsymbol{q}=-m\Psi^T\boldsymbol{e}\,\delta\omega_0^2\sin\omega_0 t.$$

Simplifying and writing the individual equations,

$$\ddot{q}_i+\lambda_i^2\omega_0^2 q_i=-\boldsymbol{\psi}_i^T\,\boldsymbol{e}\,\delta\omega_0^2\sin\omega_0 t=-\delta_i\omega_0^2\sin\omega_0 t,$$

we have, for rest initial conditions, the individual integrals

$$q_i(t)=\frac{\delta_i}{1-\lambda_i^2}(\sin\omega_0 t-\frac{1}{\lambda_i}\sin\lambda_i\omega_0 t)$$

```
delta_i = Psi.T@e*delta
C_i = delta_i/(1.0-lambda_i**2)

qd0 = [lambda a, c=c, l=l:c*(sin(a)-sin(l*a)/l) for c, l in zip(C_i, lambda_i)]
qv0 = [lambda a, c=c, l=l:c*(cos(a)-cos(l*a)/1) for c, l in zip(C_i, lambda_i)]

sn, cn = sin(2*pi*lambda_i), cos(2*pi*lambda_i)
d2pi, v2pi = -C_i*sn/lambda_i, C_i*(1-cn)
A_i, B_i = sn*d2pi+cn*v2pi/lambda_i, cn*d2pi-sn*v2pi/lambda_i

qd1 = [lambda a,A=A,B=B,l=l:   A*sin(l*a)+  B*cos(l*a) for A,B,l in zip(A_i,B_i,lambda_i)]
qv1 = [lambda a,A=A,B=B,l=l: l*A*cos(l*a)-l*B*sin(l*a) for A,B,l in zip(A_i,B_i,lambda_i)]
```

4.2.1 The modal responses, analytic expressions

```
for i in (0, 1, 2):
    l = lambda_i[i]
    qqq = r'q_%d(t) = '%(i+1)
    beg = r'\begin{cases}'
    eq1 = r'%+f\,(\sin\omega_0t-\frac{1}{%f}\sin%f\,\omega_0t).&'%(C_i[i], l, l)
    in1 = r'\qquad0\le\omega_0t\le2\pi\\'
    eq2 = r'%+f\,\sin%f\omega_0t_%+f\,\cos%f\,\omega_0t,&'%(A_i[i],l,B_i[i],l)
    in2 = r'\qquad2\pi\le\omega_0t.'
    end = r'\end{cases}'
    platex(qqq,beg,eq1,in1,eq2,in2,end)
```

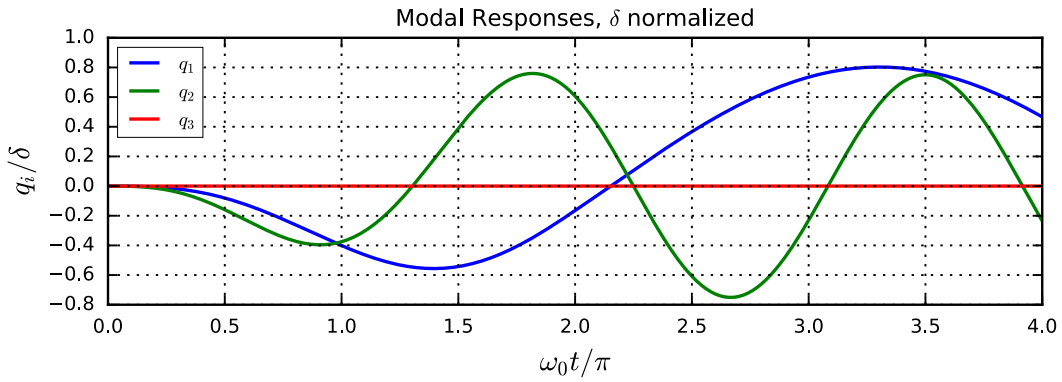
$$q_1(t)=\begin{cases}+0.178041(\sin\omega_0 t-\frac{1}{0.434134}\sin 0.434134\omega_0 t). & 0\le\omega_0 t\le 2\pi\\ -0.785592\sin 0.434134\omega_0 t-0.164919\cos 0.434134\omega_0 t, & 2\pi\le\omega_0 t.\end{cases}$$

$$q_2(t)=\begin{cases}-0.765321(\sin\omega_0 t-\frac{1}{1.200424}\sin 1.200424\omega_0 t). & 0\le\omega_0 t\le 2\pi\\ +0.442147\sin 1.200424\omega_0 t+0.606862\cos 1.200424\omega_0 t, & 2\pi\le\omega_0 t.\end{cases}$$

$$q_3(t)=\begin{cases}+0.001024(\sin\omega_0 t-\frac{1}{3.866453}\sin 3.866453\omega_0 t). & 0\le\omega_0 t\le 2\pi\\ -0.000088\sin 3.866453\omega_0 t+0.000197\cos 3.866453\omega_0 t, & 2\pi\le\omega_0 t.\end{cases}$$

4.2.2 The modal responses, plots

```
q = array([np.where(a<pi*2, qd0[i](a), qd1[i](a)) for i in range(3)])
lines = plt.plot(a/pi, q.T)
for i, l in enumerate(lines):
    l.set_label('$q_%d$'%(i+1))
plt.title('Modal Responses, $\\delta$ normalized')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$q_i/\delta$', fontsize='large')
plt.legend(loc='best');
```



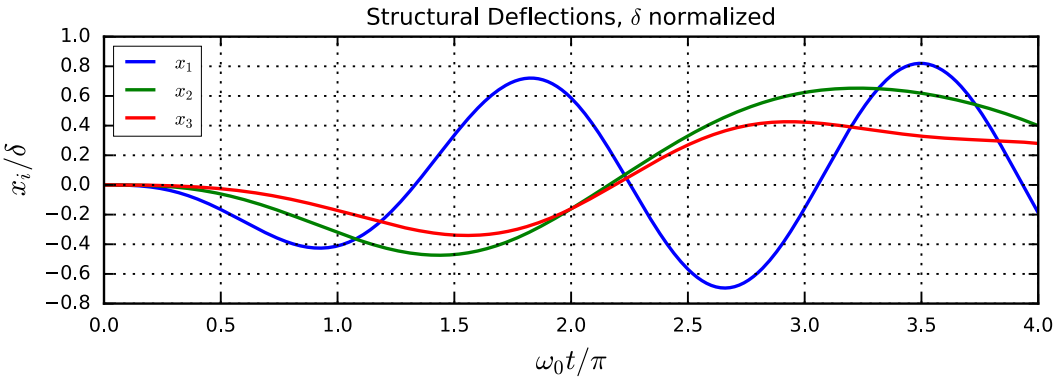
4.3 Structural response

We have to perform the following computations

1. from the modal response compute the dynamic component of structural displacements,
2. using the influence matrix, compute the static component,
3. summing the dynamic and the static components, compute the total displacements.

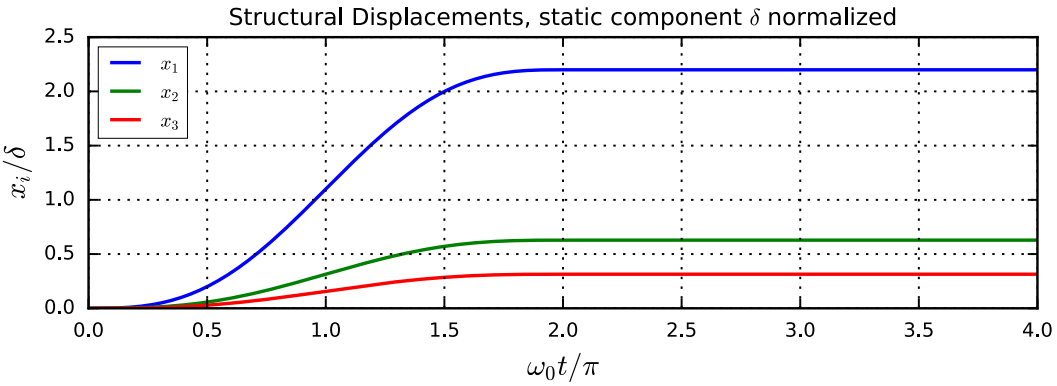
4.3.1 The dynamic component

```
xd = Psi@q
lines = plt.plot(a/pi, xd.T)
for i, l in enumerate(lines): l.set_label('$x_%d$'%(i+1))
plt.title('Structural Deflections, $\\delta$ normalized')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$x_i/\delta$', fontsize='large')
plt.legend(loc='best');
```



4.3.2 The static component

```
xs = e[:,None] * np.where(a<2*pi, a-sin(a), 2*pi)
lines = plt.plot(a/pi, xs.T)
for i, l in enumerate(lines): l.set_label('$x_%d$'%(i+1))
plt.title('Structural Displacements, static component $\\delta$ normalized')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$x_i/\delta$', fontsize='large')
plt.legend(loc='best');
```



4.3.3 Total displacements

```
xt = xd+xs
lines = plt.plot(a/pi, xt.T, linewidth=3)
for i, l in enumerate(lines): l.set_label('$x_%d$'%(i+1))
plt.plot(a/pi, xs[0], '--b', linewidth=1)
plt.plot(a/pi, xs[1], '--g', linewidth=1)
plt.plot(a/pi, xs[2], '--r', linewidth=1)
plt.title('Total Displacements, $\\delta$ normalized')
plt.xlabel(r'$\omega_0t/\pi$', fontsize='large')
plt.ylabel(r'$x_i/\delta$', fontsize='large')
plt.legend(loc='best');
```