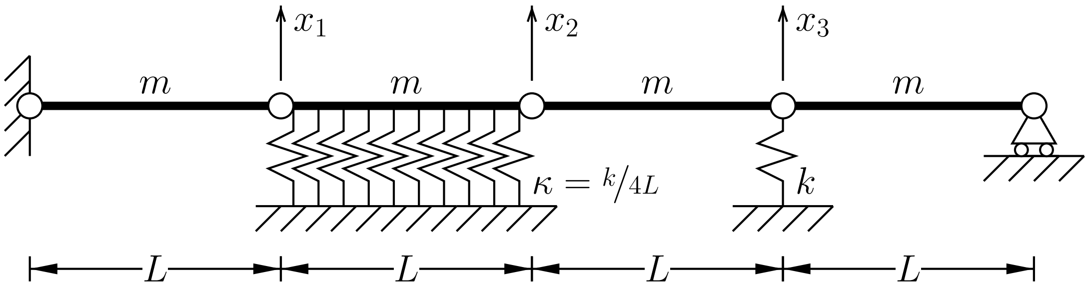
05_Rayleigh

May 23, 2014

1 Rayleigh estimates

```
display(SVG(filename='rigid.svg'))
```



For a single bar,

$$\dot{x}(s) = \dot{x}_1 + \frac{\dot{x}_{i+1} - \dot{x}_i}{L}\, s$$

and the total kinetic energy is

$$T = \frac{1}{2}\int_0^L \frac{m}{L}\dot{x}^2(s)\,\mathrm{d}s = \frac{1}{2}\left(\frac{1}{3}\dot{x}_i^2 + \frac{1}{3}\dot{x}_i\dot{x}_{i+1} + \frac{1}{3}\dot{x}_{i+i}^2\right) m$$

We compute, bar by bar, the contribution to twice the kinetic energy. The velocities of the first and last bar have to be written in terms of the fictitious coordinates x_0 and x_4 where $x_0 = x_4 = 0$

```
def index(i):
    return i*2, i*2+1, i*2+2

# We use x0, x1, x2, x3, x4, with the understanding that x0=x4=0
# we compute twice the kinetic energy storing the coefficients
# of the quadratic form in a sequence of length 9 --- note
# that the cross terms are restricted to adjacent DOFs

# Initially T2 is zero
#      x0x0, x0x1, x1x1, x1x2, x2x2, x2x3, x3x3, x3x4, x4x4
T2 = [    0,    0,    0,    0,    0,    0,    0,    0,    0]

# then for each bar  we add, in the correct positions, the contributions to T2
# due to the DOFs associated with the current bar
for i in (0,1,2,3):
    i11,i12,i22=index(i)
    T2[i11] += frac(1,3)
    T2[i12] += frac(1,3)
    T2[i22] += frac(1,3)
```

we print the coefficients, omitting the ones associated with either x_0 or x_4

```
print T2[2:-2]
```

```
[Fraction(2, 3), Fraction(1, 3), Fraction(2, 3), Fraction(1, 3), Fraction(2, 3)]
```

With the help of a helper function, we display the quadratic form for $2T$ and also for $2V$, that's so easy to determine that I leave its determination to the reader

```
def Lf(fr):
    n = fr.numerator
    d = fr.denominator
    if n>0:
        return r' + \frac{%d}{%d} '%( n,d) if d>1 else '%d'%(n,)
    elif n<0:
        return r' - \frac{%d}{%d} '%(-n,d) if d>1 else '%d'%(n,)
    else:
        return ' 0 '

s = (r'$$2 T /m= ' +
     Lf(T2[2])+'x_1^2' +
     Lf(T2[3])+'x_1x_2'+
     Lf(T2[4])+'x_2^2'+
     Lf(T2[5])+'x_2x_3'
    +Lf(T2[6])+'x_3^2$$')
display(Latex(s))
display(Latex(r'$$2V/k=\frac1{12}x_1^2+\frac1{12}x_1x_2+\frac1{12}x_2^2+x_3^2$$'))
```

$$2T/m = +\frac{2}{3}x_1^2 + \frac{1}{3}x_1x_2 + \frac{2}{3}x_2^2 + \frac{1}{3}x_2x_3 + \frac{2}{3}x_3^2$$

$$2V/k = \frac{1}{12}x_1^2 + \frac{1}{12}x_1x_2 + \frac{1}{12}x_2^2 + x_3^2$$

Using again fraction, we construct the nass matrix and the stiffness matrix equating the double matrix products to our previous results.

```
six = frac(1,6)
M = matrix((
            (4*six,six,0),
            (six,4*six,six),
            (0,six,4*six)
            ))
tw4 = frac(1,24)
one = frac(1)
K = matrix((
            (2*tw4,tw4, frac()),
            (tw4,2*tw4, frac()),
            (frac(),frac(),one)))
```

Let's store the stiffness inverse (a bit of work to have integer coefficients).

```
F = matrix(map(int,ravel(K.I))).reshape((3,3))
```

Our initial guess for the shape vector

```
# u = matrix('-10;10;1')
u = matrix('1;-1;0')
```

Our initial guess for the Rayleigh estimate,

```
n, = ravel(u.T*K*u)
d, = ravel(u.T*M*u)
print n, d, '\t\tRoo =', n/d,'=',1.*n/d
```

```
1/12 1                Roo = 1/12 = 0.0833333333333
```

for the second estimate, the previous denominator is the new numerator, and we have to compute a new denominator

```
n = d
d, = ravel(u.T*M*F*M*u)
print n, d, '\t\tRoi =', n/d,'=',1.*n/d
```

```
1 433/36                Roi = 36/433 = 0.0831408775982
```

and then the final estimate that was requested

```
n = d
d, = ravel(u.T*M*F*M*F*M*u)
print n, d, '\t\tRii =', n/d,'=',1.*n/d
```

```
433/36 7813/54                Rii = 1299/15626 = 0.0831306796365
```

To see if ours result is good, we have to cheat...

```
from scipy.linalg import eigh
eval, evec = eigh(K,M, eigvals=(0,0))
print "1st eigenvalue  =  ", eval[0]
print "1st eigenvector =", ravel(evec)
```

```
1st eigenvalue  =   0.0831297397226
1st eigenvector = [ 0.99565905 -0.99930337 -0.01465763]
```

The results are good, but I sort of cheated in the choice of the trial shape vector, as I already had an idea of the first eigenvector of the system.