Lateral torsional buckling - LTBeamN example scrapbook

March 22, 2020

1 Preliminaries

Install *LTBeamN* available at https://www.cticm.com/logiciel/ltbeamn/

For the *jupyter notebook* do the following steps:

1. Install anaconda distribution available at https://www.anaconda.com/distribution/ ;
2. Go to *Anaconda prompt*, on *Windows*, or the terminal, on *MacOS/Linux*;
3. Put yourself in the directory where you have the notebook by typing "cd" followed by the path;
4. Type "jupyter notebook"

Import Python libraries

[1]:
```
import numpy as np # Numerical library
import pandas as pd # Data analysis library
import matplotlib.pyplot as plt # Plotting library
```

Function Definitions

[2]:
```
def singlySymmetricGeoProp(h,bf1,tf1,bf2,tf2,tw):
    """This function calculates the geometric properties of singly symmetric␣
 ↪cross-sections"""

    hw=h-tf1-tf2

    A=(bf2*tf2+tw*hw+bf1*tf1)

    zG=(bf2*tf2*tf2/2.+tw*hw*(hw/2.+tf2)+bf1*tf1*(h-tf1/2.))/
 ↪(bf2*tf2+tw*hw+bf1*tf1)

    ht=zG-tf2/2.
    hc=h-tf1/2.-zG

    # Shear center position - cf. slide 39 of course
    if (bf1**3*tf1-bf2**3*tf2)==0:
        zc=0.
    else:
        zc = (bf1**3*hc*tf1-bf2**3*ht*tf2)/(bf1**3*tf1-bf2**3*tf2)
```

```
    Iw = (hc+ht)**2*bf1**3*tf1*bf2**3*tf2/(12*(bf1**3*tf1+bf2**3*tf2)) #␣
↪Warping constant - cf. slide 39 of course

    K = 1/3.*(bf1*tf1**3+bf2*tf2**3+(hc+ht)*tw**3) # Torsional - cf. slide 39␣
↪of course

    # Strong Axis moment of inertia
    Iy= bf1*tf1**3/12.+bf1*tf1*hc**2 +\
        tw*hw**3/12.+hw*tw*(zG-(hw/2.+tf2/2.))**2+\
        bf1*tf1**3/12.+bf1*tf1*hc**2

    # Weak axis moment of inertia
    Iz= bf1**3*tf1/12.+\
        tw**3*hw/12.+\
        bf1**3*tf1/12.


    # Sectorial characteristic - cf. slide 39 of course
    beta = zc+1/(2.*Iy)*(ht*(bf2**3*tf2/12.+bf2*tf2*ht**2+ht**3*tw/4.)-\
                         hc*(bf1**3*tf1/12.+bf1*tf1*ht**2+hc**3*tw/4.))

    return zc,A,Iy,Iz,K,Iw,beta
```

Clarke and Hill's critical moment:

$$M_{cr} = \frac{C_1 \pi^2 E I_z}{k_v k_\varphi L^2} \left(\sqrt{(C_2 z_a + C_3 \beta)^2 + \frac{I_\omega}{I_z} \left(\frac{G K k_\varphi^2 L^2}{\pi^2 E I_\omega} + 1 \right)} + C_2 z_a + C_3 \beta \right) \tag{1}$$

[3]:
```
def clarkHill_criticalMoment(E, G, Iz, K, Iw, beta, za, L,
                              kphi=1.0,kv=1.0, C_1=1.0,C_2=1.0,C_3=1.0):
    """
    This function returns the critical moment as proposed by Clark and Hill -␣
↪cf. slide 42 of LTB course

    """

    M_cr=C_1*np.pi**2*E*Iz/(kv*kphi*L**2)*(\
              np.sqrt((C_2*za+C_3*beta)**2+Iw/Iz*(G*K*kphi**2*L**2/(np.
↪pi**2*E*Iw)+1))+\
                    (C_2*za+C_3*beta))

    return M_cr
```

2 Units and material properties

[4]:
```
E=210e3 #N/mm2
nu=0.3 # Poisson's ratio
fy=235 #N/mm2

G=E/(2*(1+nu))
```

3 Geometric properties

3.1 Bridge grider(BG)

Cross-section

[5]:
```
#Inputs in mm

h=1070.

bf1=450.
tf1=35.

bf2=450.
tf2=35.

tw=20.

zc_BG, A_BG, Iy_BG, Iz_BG, K_BG, Iw_BG, beta_BG =␣
 ↪singlySymmetricGeoProp(h,bf1,tf1,bf2,tf2,tw)

print('zc_BG = ',round(zc_BG,2),' mm')
print('A_BG = ',round(A_BG,2),' mm^4')
print('Iy_BG = ',round(Iy_BG,2),' mm^4')
print('Iz_BG = ',round(Iz_BG,2),' mm^4')
print('K_BG = ',round(K_BG,2),' mm^4')
print('Iw_BG = ',round(Iw_BG,2),' mm^6')
print('beta_BG = ',round(beta_BG,2),' mm')
```

```
zc_BG =  0.0  mm
A_BG =  51500.0  mm^4
Iy_BG =  10111904166.67  mm^4
Iz_BG =  532229166.67  mm^4
K_BG =  15622500.0  mm^4
Iw_BG =  142355759765625.0  mm^6
beta_BG =  0.0  mm
```

Element characteristics

[6]:
```python
L_BG=24000. #mm
```

3.2 Transverse beam (TB)

HEB300 - Cross-section

[7]:
```python
A_TB= 149.1e2 #mm2
Iy_TB= 25160e4 #mm4
Iz_TB= 8562e4 #mm4

K_TB= 189.1e4 #mm4
Iw_TB= 1687e9 #mm6
```

Element characteristics

[8]:
```python
L_TB=8000. #mm
```

4 LTBeamN analysis

In this section we perform some analysis for the *LTBeamN* software.

Firstly, we shall see how to setup some basic properties for the project. These include:

1. Defining the beam's length;
2. Defining the beam's material;
3. Defining the beam's cross-section.

After, we shall analyze the sensitivity of the critical moment depending on:

1. Boundary conditions;
2. Loading type;
3. Load position;
4. Lateral restraints;
5. Effect of axial load;

We shall build a table in order to keep track of our analyses:

[9]:
```python
analysisTable_header=['Case','Boundary Cond.','Loading','Restraints','Load␣
↪Position','Axial Load',
                      'M_cr_LTBeamN','M_cr_ClarkHill']

analysisTable=pd.DataFrame(columns=analysisTable_header)
```

Furthermore, when interpreting the results from *LTBeamN* we should always bear in mind that the program is, by default, computing a multiple of whatever loading is applied to the beam. In other words, say that you have a distributed load q applied to the beam, what the program is computing is **how many times do I have to multiply q in order to lose stability in the member**:

$$q_{cr} = \mu_{cr} q \tag{2}$$

In so far that the loading q produces a maximum moment of $M_{max,q}$ then the critical moment in this situation will be,

$$M_{cr} = \mu_{cr} M_{max,q} \tag{3}$$

4.1 Simply supported beam subjected to distributed load at shear center, warping free at ends - Case 1

[10]:
```
za=0.0 # loading applied at shear center

L_d=L_BG # Buckling length

caseCounter=1

Mcr_CH=clarkHill_criticalMoment(E, G, Iz_BG, K_BG, Iw_BG, beta_BG, za, L_d,
                                kphi=1.0,kv=1.0,
                                C_1=1.13,C_2=0.46,C_3=0.53)/1e6 #kN.m

analysisRow=[caseCounter,'Simple, Warp. Free','Dist. Load','No Inter. Lateral␣
↪Restraint','Shear Center',
             0.0,
             2058.,
             round(Mcr_CH)]

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[10]:
```
   Case       Boundary Cond.     Loading                      Restraints  \
0     1  Simple, Warp. Free  Dist. Load  No Inter. Lateral Restraint

   Load Position  Axial Load  M_cr_LTBeamN  M_cr_ClarkHill
0   Shear Center         0.0        2058.0          2083.0
```

4.2 Simply supported beam subjected to distributed load outside shear center, warping free at ends - Case 2

[11]:
```
za=355.0 # mm

L_d=L_BG # Buckling length
```

```
caseCounter+=1

Mcr_CH=clarkHill_criticalMoment(E, G, Iz_BG, K_BG, Iw_BG, beta_BG, za, L_d,
                                kphi=1.0,kv=1.0,
                                C_1=1.13,C_2=0.46,C_3=0.53)/1e6 #kN.m

analysisRow=[caseCounter,'Simple, Warp. Free','Dist. Load','No Lateral␣
↪Restraint','za=355mm',
             0.0,
             2438,
             round(Mcr_CH)]

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[11]:

```
   Case      Boundary Cond.     Loading                  Restraints  \
0     1  Simple, Warp. Free  Dist. Load  No Inter. Lateral Restraint
0     2  Simple, Warp. Free  Dist. Load        No Lateral Restraint

  Load Position  Axial Load  M_cr_LTBeamN  M_cr_ClarkHill
0  Shear Center         0.0        2058.0          2083.0
0      za=355mm         0.0        2438.0          2466.0
```

4.3 Simply supported beam subjected to distributed load outside shear center, warping restrained at ends - Case 3

[12]:

```
za=355.0 # mm

L_d=L_BG # Buckling length

caseCounter+=1

Mcr_CH=clarkHill_criticalMoment(E, G, Iz_BG, K_BG, Iw_BG, beta_BG, za, L_d,
                                kphi=0.5,kv=1.0,
                                C_1=1.13,C_2=0.46,C_3=0.53)/1e6 #kN.m

analysisRow=[caseCounter,'Simple, Warp. Fixed','Dist. Load','No Lateral␣
↪Restraint','za=355mm',
             0.0,
             3410,
             round(Mcr_CH)]
```

```
analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[12]:
```
   Case        Boundary Cond.     Loading                     Restraints  \
0     1    Simple, Warp. Free  Dist. Load  No Inter. Lateral Restraint
0     2    Simple, Warp. Free  Dist. Load         No Lateral Restraint
0     3   Simple, Warp. Fixed  Dist. Load         No Lateral Restraint

   Load Position  Axial Load  M_cr_LTBeamN  M_cr_ClarkHill
0   Shear Center         0.0        2058.0          2083.0
0       za=355mm         0.0        2438.0          2466.0
0       za=355mm         0.0        3410.0          3639.0
```

4.4 Simply supported beam subjected to distributed load outside shear center, warping free at ends and restraints, lateral displacements and rotations fixed at 8 meter intervals - Case 4

[13]:
```
za=355 # mm

L_d=8000 # Buckling length

caseCounter+=1

Mcr_CH=clarkHill_criticalMoment(E, G, Iz_BG, K_BG, Iw_BG, beta_BG, za, L_d,
                                kphi=1.0,kv=1.0,
                                C_1=1.00,  ##    <----- more like uniform␣
↪moment than a parabolic distribution
                                C_2=0.46,C_3=0.53)/1e6 #kN.m

analysisRow=[caseCounter,'Simple, Warp. Free','Dist. Load','@8m lateral Fixed␣
↪and rotation Fixed','za=355mm',
             0.0,
             13892,
             round(Mcr_CH)]

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[13]:
```
   Case        Boundary Cond.     Loading                               Restraints  \
0     1    Simple, Warp. Free  Dist. Load            No Inter. Lateral Restraint
0     2    Simple, Warp. Free  Dist. Load                   No Lateral Restraint
```

```
0    3  Simple, Warp. Fixed  Dist. Load                    No Lateral Restraint
0    4   Simple, Warp. Free  Dist. Load  @8m lateral Fixed and rotation Fixed

  Load Position  Axial Load  M_cr_LTBeamN  M_cr_ClarkHill
0  Shear Center         0.0        2058.0          2083.0
0      za=355mm         0.0        2438.0          2466.0
0      za=355mm         0.0        3410.0          3639.0
0      za=355mm         0.0       13892.0         13261.0
```

Let's see how well we are doing with this approximation by defining the Clark Hill approximation error as

$$Error_{CH} = \frac{M_{cr,CH} - M_{cr,LTBeam}}{M_{cr,LTBeam}} * 100 \tag{4}$$

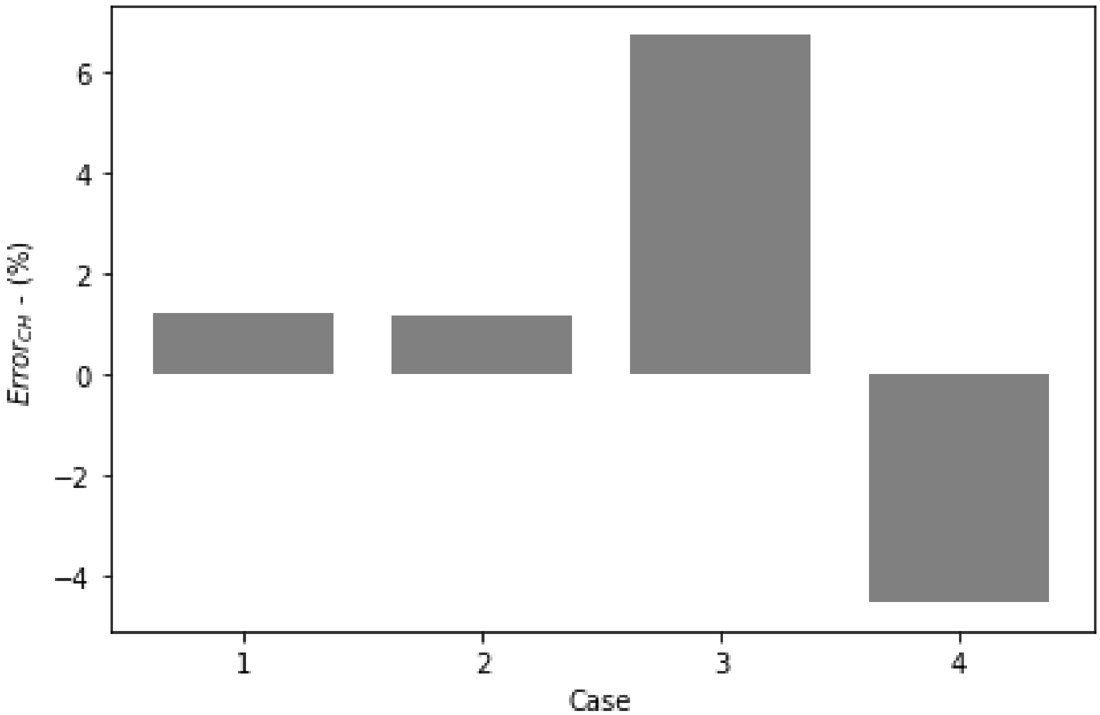
[14]:
```
analysisTable['Clark Hill Error pc.']=np.
↪round((analysisTable['M_cr_ClarkHill']-analysisTable['M_cr_LTBeamN'])\
                                                    /
↪analysisTable['M_cr_LTBeamN']*100. ,2)

analysisTable_header= analysisTable_header+['Clark Hill Error pc.']

analysisTable
```

[14]:
```
  Case        Boundary Cond.     Loading                           Restraints  \
0    1   Simple, Warp. Free  Dist. Load           No Inter. Lateral Restraint
0    2   Simple, Warp. Free  Dist. Load                  No Lateral Restraint
0    3  Simple, Warp. Fixed  Dist. Load                  No Lateral Restraint
0    4   Simple, Warp. Free  Dist. Load  @8m lateral Fixed and rotation Fixed

  Load Position  Axial Load  M_cr_LTBeamN  M_cr_ClarkHill  \
0  Shear Center         0.0        2058.0          2083.0
0      za=355mm         0.0        2438.0          2466.0
0      za=355mm         0.0        3410.0          3639.0
0      za=355mm         0.0       13892.0         13261.0

   Clark Hill Error pc.
0                  1.21
0                  1.15
0                  6.72
0                 -4.54
```

or in graphical form:

[15]:
```
plt.bar(analysisTable['Case'],analysisTable['Clark Hill Error pc.'],width=0.
↪75,color='0.5')
plt.xticks(np.array(analysisTable['Case'],dtype=int))
```

```
plt.xlabel('Case')
plt.ylabel(r'$Error_{CH}$ - (%)')
plt.tight_layout()
```



4.5 Simply supported beam subjected to distributed load outside shear center, warping free at ends and restraints, lateral displacements fixed and flexible rotations at 8 meter intervals - Case 5

Now, because the bridge girders depicted in the problem set are connected between each other, they share a common global buckling mode. Since they buckle together, the HEB300 provides some rotational stiffness to the bridge girder, but it's not completely fixed. This situation is very hard to take into account in calculating the critical moment in an approximate way with the Clark-Hill formula. To use this formula we would be forced to make simplifications (like the rotationally fixed one) which could to lead an overestimation of our critical moment. Let's see by how much. First let us estimate the rotational stiffness the HEB300 provides and then plug it in *LTBeam* to obtain a more refined critical moment.

The rotational stiffness of a simply supported beam loaded in pure bending by moments at the extremities can be shown to be equal to:

$$K_\theta = \frac{2EI}{L} \tag{5}$$

[16]:
```
2*E*Iy_TB/L_TB*1e-6 #kN.m/rad
```

[16]: 13209.0

[17]:
```
za=355 # mm

L_d=8000 # Buckling length

caseCounter+=1

Mcr_CH='-'

analysisRow=[caseCounter,'Simple, Warp. Free','Dist. Load','@8m lateral Fixed␣
 ↪and rotation flex.','za=355mm',
            0.0,
            13721,
            Mcr_CH,
            '-'] # <---- no error calculation

analysisTable=analysisTable.append(pd.
 ↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[17]:
```
  Case       Boundary Cond.     Loading                             Restraints  \
0    1   Simple, Warp. Free  Dist. Load         No Inter. Lateral Restraint
0    2   Simple, Warp. Free  Dist. Load                No Lateral Restraint
0    3  Simple, Warp. Fixed  Dist. Load                No Lateral Restraint
0    4   Simple, Warp. Free  Dist. Load  @8m lateral Fixed and rotation Fixed
0    5   Simple, Warp. Free  Dist. Load  @8m lateral Fixed and rotation flex.

  Load Position  Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.
0  Shear Center         0.0        2058.0           2083                 1.21
0      za=355mm         0.0        2438.0           2466                 1.15
0      za=355mm         0.0        3410.0           3639                 6.72
0      za=355mm         0.0       13892.0          13261                -4.54
0      za=355mm         0.0       13721.0              -                    -
```

4.6 Simply supported beam subjected to distributed load outside shear center, warping free at ends and restraints, lateral displacements fixed throughout the length and flexible rotations at 8 meter intervals - Case 6

[18]:
```
za=355 # mm

L_d=8000 # Buckling length
```

```
caseCounter+=1

Mcr_CH='-'

analysisRow=[caseCounter,'Simple, Warp. Free','Dist. Load','Laterally Fixed␣
↪throughout and rotation flex. @8m','za=355mm',
             0.0,
             15793,
             Mcr_CH,
             '-'] # <---- no CH error calculation

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[18]:

```
   Case        Boundary Cond.     Loading  \
0     1    Simple, Warp. Free  Dist. Load
0     2    Simple, Warp. Free  Dist. Load
0     3   Simple, Warp. Fixed  Dist. Load
0     4    Simple, Warp. Free  Dist. Load
0     5    Simple, Warp. Free  Dist. Load
0     6    Simple, Warp. Free  Dist. Load

                                          Restraints Load Position  \
0                        No Inter. Lateral Restraint  Shear Center
0                              No Lateral Restraint      za=355mm
0                              No Lateral Restraint      za=355mm
0                @8m lateral Fixed and rotation Fixed      za=355mm
0                @8m lateral Fixed and rotation flex.      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm

   Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.
0         0.0        2058.0           2083                 1.21
0         0.0        2438.0           2466                 1.15
0         0.0        3410.0           3639                 6.72
0         0.0       13892.0          13261                -4.54
0         0.0       13721.0              -                    -
0         0.0       15793.0              -                    -
```

4.7 Simply supported beam subjected to distributed load outside shear center, warping free at ends and restraints, lateral displacements fixed throughout the length and flexible rotations at 8 meter intervals, with axial load - Case 7

Now, here we have an ambiguous situation: what precisely do we mean by the buckling multiplier(μ_{cr}) when we have two independent load types, one producing flexural moments, and another axial load?

As mentioned previously, by default *LTBeam* multiplies whatever loading exists in the beam (M_{Ed}, N_{Ed}) proportionally. Let's call this option 1:

1.

$$M_{cr} = \mu_{cr} M_{Ed,max} \quad ; \quad N_{cr} = \mu_{cr} N_{Ed,max} \tag{6}$$

Another alternative, say option 2, would be to hold the axial load constant while increasing the moment until stability in the system is lost.

2.

$$M_{cr} = \mu_{cr} M_{Ed,max} \quad ; \quad N_{cr} = N_{Ed,max} \tag{7}$$

Yet another, option 3, would be the converse of option 2, that is to hold the moment constant while increasing the axial load until stability in the system is lost.

3.

$$M_{cr} = M_{Ed,max} \quad ; \quad N_{cr} = \mu_{cr} N_{Ed,max} \tag{8}$$

Which one you choose will depend on the load case, and hence the question, you want to solve.

Let's imagine a situation in which insufficient space was left at the extremities for the bridge to expand. In this situation, an increase in temperature will submit our cross-section to axial load. An estimation of this axial load can be made by:

$$N = EA\alpha\Delta T \tag{9}$$

[19]:
```
Delta_T=20 # °C
alpha=1e-5 # Coefficient of thermal expansion m/(m.°C)

E*A_BG*alpha*Delta_T*1e-3 #kN
```

[19]: 2163.0000000000005

Here I would be mostly interested in how much heavy traffic can the bridge girder support, in the presence of a 20°C temperature rise, before losing its stability.

[20]:
```
za=355 # mm

L_d=8000 # Buckling length

caseCounter+=1
```

```
Mcr_CH='-'

analysisRow=[caseCounter,'Simple, Warp. Free','Dist. Load','Laterally Fixed␣
↪throughout and rotation flex. @8m','za=355mm',
             'Constant 2160',
             14471,
             Mcr_CH,
             '-'] # <---- no CH error calculation

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[20]:

```
   Case        Boundary Cond.     Loading  \
0     1    Simple, Warp. Free  Dist. Load
0     2    Simple, Warp. Free  Dist. Load
0     3   Simple, Warp. Fixed  Dist. Load
0     4    Simple, Warp. Free  Dist. Load
0     5    Simple, Warp. Free  Dist. Load
0     6    Simple, Warp. Free  Dist. Load
0     7    Simple, Warp. Free  Dist. Load

                                         Restraints Load Position  \
0                        No Inter. Lateral Restraint  Shear Center
0                               No Lateral Restraint      za=355mm
0                               No Lateral Restraint      za=355mm
0                @8m lateral Fixed and rotation Fixed      za=355mm
0                @8m lateral Fixed and rotation flex.      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm

      Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.
0              0        2058.0           2083                 1.21
0              0        2438.0           2466                 1.15
0              0        3410.0           3639                 6.72
0              0       13892.0          13261                -4.54
0              0       13721.0              -                    -
0              0       15793.0              -                    -
0  Constant 2160       14471.0              -                    -
```

4.8 Simply supported beam subjected to distributed load outside shear center, warping free at ends and restraints, lateral displacements fixed throughout the length and flexible rotations at 8 meter intervals, with variable axial load - Case 8

Another question, for example, I would be interested in asking is: for a given traffic volume (fixed vertical load), how fast can the traffic go so that if all vehicles brake at the same time, the bridge girder will lose its stability? Since vehicle speed is directly related to its braking force, this is equivalent to scaling the braking forces on the bridge girder until the girder buckles.

[21]:
```
za=355 # mm

L_d=8000 # Buckling length

caseCounter+=1

Mcr_CH='-'

analysisRow=[caseCounter,'Simple, Warp. Free','Dist. Load','Laterally Fixed␣
↪throughout and rotation flex. @8m','za=355mm',
             'Variable -30828',
             2336,
             Mcr_CH,
             '-'] # <---- no CH error calculation

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[21]:
```
  Case       Boundary Cond.     Loading  \
0    1   Simple, Warp. Free  Dist. Load
0    2   Simple, Warp. Free  Dist. Load
0    3  Simple, Warp. Fixed  Dist. Load
0    4   Simple, Warp. Free  Dist. Load
0    5   Simple, Warp. Free  Dist. Load
0    6   Simple, Warp. Free  Dist. Load
0    7   Simple, Warp. Free  Dist. Load
0    8   Simple, Warp. Free  Dist. Load

                                       Restraints Load Position  \
0                        No Inter. Lateral Restraint  Shear Center
0                              No Lateral Restraint      za=355mm
0                              No Lateral Restraint      za=355mm
0                @8m lateral Fixed and rotation Fixed      za=355mm
0                @8m lateral Fixed and rotation flex.      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
```

```
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm

        Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.
0                0        2058.0           2083                 1.21
0                0        2438.0           2466                 1.15
0                0        3410.0           3639                 6.72
0                0       13892.0          13261                -4.54
0                0       13721.0              -                    -
0                0       15793.0              -                    -
0    Constant 2160       14471.0              -                    -
0  Variable -30828        2336.0              -                    -
```

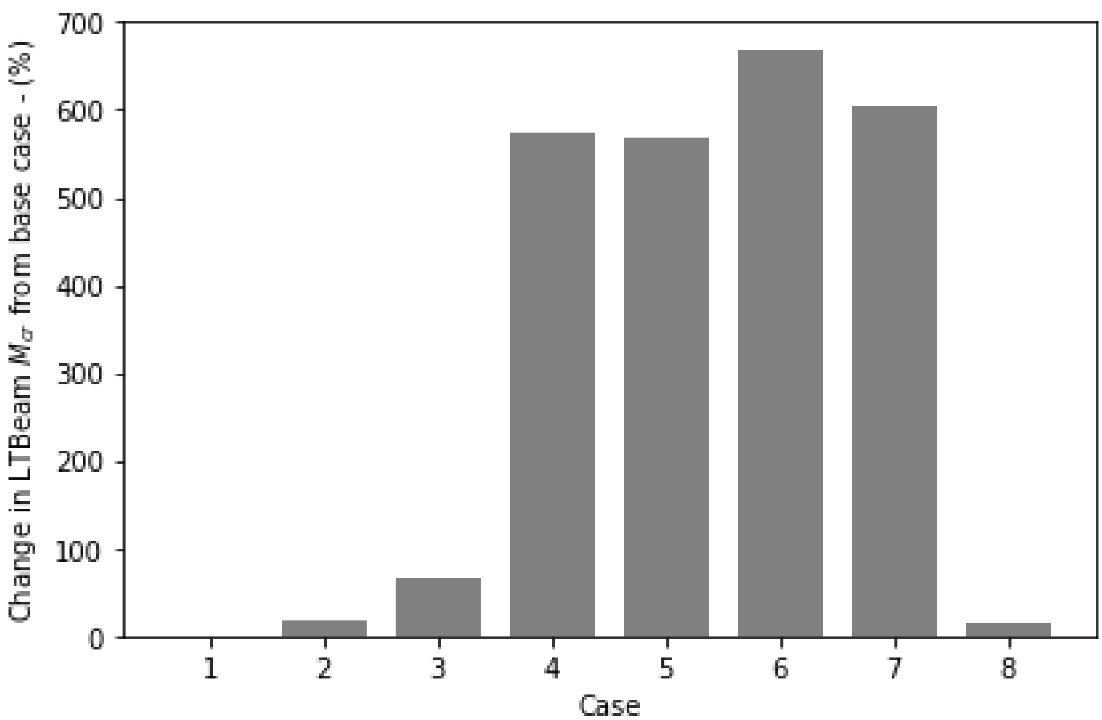
[22]:
```
analysisTable['Change Base pc.']=np.
 ↪round((analysisTable['M_cr_LTBeamN']-analysisTable['M_cr_LTBeamN'].iloc[0])\
                                              /analysisTable['M_cr_LTBeamN'].
 ↪iloc[0]*100. ,2)

analysisTable_header= analysisTable_header+['Change Base pc.']

analysisTable
```

[22]:
```
  Case        Boundary Cond.     Loading  \
0    1   Simple, Warp. Free  Dist. Load
0    2   Simple, Warp. Free  Dist. Load
0    3  Simple, Warp. Fixed  Dist. Load
0    4   Simple, Warp. Free  Dist. Load
0    5   Simple, Warp. Free  Dist. Load
0    6   Simple, Warp. Free  Dist. Load
0    7   Simple, Warp. Free  Dist. Load
0    8   Simple, Warp. Free  Dist. Load

                                          Restraints Load Position  \
0                       No Inter. Lateral Restraint   Shear Center
0                              No Lateral Restraint       za=355mm
0                              No Lateral Restraint       za=355mm
0              @8m lateral Fixed and rotation Fixed       za=355mm
0              @8m lateral Fixed and rotation flex.       za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm

        Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.  \
0                0        2058.0           2083                 1.21
0                0        2438.0           2466                 1.15
0                0        3410.0           3639                 6.72
0                0       13892.0          13261                -4.54
```

```
0                0       13721.0             -                      -
0                0       15793.0             -                      -
0     Constant 2160      14471.0             -                      -
0   Variable -30828       2336.0             -                      -

   Change Base pc.
0             0.00
0            18.46
0            65.69
0           575.02
0           566.72
0           667.40
0           603.16
0            13.51
```

[23]:

```
plt.bar(analysisTable['Case'],analysisTable['Change Base pc.'],width=0.
 ↪75,color='0.5')
plt.xticks(np.array(analysisTable['Case'],dtype=int))
plt.xlabel('Case')
plt.ylabel(r'Change in LTBeam $M_{cr}$ from base case - (%)')
plt.tight_layout()
```



4.9 Continuous beam on three supports, loaded from the top flange by a distributed load, intermediately braced @4m laterally and rotationally - Case 9

[24]:
```
caseCounter+=1

za=-535.

L_d=4000.

psi=40/-180

C_1_psi=min(1.75-1.05*psi+0.3*psi**2,2.35) #Salvadori/SIA263


Mcr_CH=clarkHill_criticalMoment(E, G, Iz_BG, K_BG, Iw_BG, beta_BG, za, L_d,
                                kphi=1.0,kv=1.0,
                                C_1=C_1_psi,C_2=0.46,C_3=0.53)/1e6 #kN.m; Be␣
↪carefull with C_2...

Mcr_LTBeam=86694.

Error_CH=round((Mcr_CH-Mcr_LTBeam)/Mcr_LTBeam*100.,2)

analysisRow=[caseCounter,'Cont., 3 Supp. , Warp. Free','Dist. Load','Laterally␣
↪and rotationally fixed @4m','za=-535mm',
             0.0,
             Mcr_LTBeam,
             Mcr_CH,
             Error_CH,
             'N.A.'] #  <---- Not applicable

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[24]:
```
  Case                 Boundary Cond.     Loading  \
0    1            Simple, Warp. Free  Dist. Load
0    2            Simple, Warp. Free  Dist. Load
0    3           Simple, Warp. Fixed  Dist. Load
0    4            Simple, Warp. Free  Dist. Load
0    5            Simple, Warp. Free  Dist. Load
0    6            Simple, Warp. Free  Dist. Load
0    7            Simple, Warp. Free  Dist. Load
0    8            Simple, Warp. Free  Dist. Load
0    9  Cont., 3 Supp. , Warp. Free  Dist. Load
```

```
                                       Restraints Load Position  \
0                       No Inter. Lateral Restraint   Shear Center
0                              No Lateral Restraint      za=355mm
0                              No Lateral Restraint      za=355mm
0               @8m lateral Fixed and rotation Fixed      za=355mm
0               @8m lateral Fixed and rotation flex.      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0               Laterally and rotationally fixed @4m     za=-535mm

       Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.  \
0               0        2058.0           2083                 1.21
0               0        2438.0           2466                 1.15
0               0        3410.0           3639                 6.72
0               0       13892.0          13261                -4.54
0               0       13721.0              -                    -
0               0       15793.0              -                    -
0    Constant 2160      14471.0              -                    -
0  Variable -30828       2336.0              -                    -
0               0       86694.0        47169.9               -45.59

  Change Base pc.
0               0
0           18.46
0           65.69
0          575.02
0          566.72
0           667.4
0          603.16
0           13.51
0            N.A.
```

4.10 Continuous beam on three supports, loaded from the top flange by a point load, intermediately braced @4m laterally and rotationally - Case 10

[25]:
```
caseCounter+=1

za=-535.

L_d=4000.

psi=133.33/-400.0

C_1_psi=min(1.75-1.05*psi+0.3*psi**2,2.35) #Salvadori/SIA263
```

```
Mcr_CH=clarkHill_criticalMoment(E, G, Iz_BG, K_BG, Iw_BG, beta_BG, za, L_d,
                                kphi=1.0,kv=1.0,
                                C_1=C_1_psi,C_2=0.46,C_3=0.53)/1e6 #kN.m; Be␣
↪carefull with C_2...
Mcr_LTBeam=97401.

Error_CH=round((Mcr_CH-Mcr_LTBeam)/Mcr_LTBeam*100.,2)

analysisRow=[caseCounter,'Cont., 3 Supp. , Warp. Free','Point Load','Laterally␣
↪and rotationally fixed @4m','za=-535mm',
             0.0,
             Mcr_LTBeam,
             Mcr_CH,
             Error_CH,
             'N.A.'] #  <---- Not applicable

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[25]:

```
   Case               Boundary Cond.     Loading  \
0     1            Simple, Warp. Free  Dist. Load
0     2            Simple, Warp. Free  Dist. Load
0     3           Simple, Warp. Fixed  Dist. Load
0     4            Simple, Warp. Free  Dist. Load
0     5            Simple, Warp. Free  Dist. Load
0     6            Simple, Warp. Free  Dist. Load
0     7            Simple, Warp. Free  Dist. Load
0     8            Simple, Warp. Free  Dist. Load
0     9  Cont., 3 Supp. , Warp. Free  Dist. Load
0    10  Cont., 3 Supp. , Warp. Free  Point Load

                                        Restraints Load Position  \
0                       No Inter. Lateral Restraint  Shear Center
0                              No Lateral Restraint      za=355mm
0                              No Lateral Restraint      za=355mm
0               @8m lateral Fixed and rotation Fixed      za=355mm
0               @8m lateral Fixed and rotation flex.      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0               Laterally and rotationally fixed @4m     za=-535mm
0               Laterally and rotationally fixed @4m     za=-535mm
```

```
        Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.  \
0                0        2058.0           2083                  1.21
0                0        2438.0           2466                  1.15
0                0        3410.0           3639                  6.72
0                0       13892.0          13261                 -4.54
0                0       13721.0              -                     -
0                0       15793.0              -                     -
0    Constant 2160       14471.0              -                     -
0  Variable -30828        2336.0              -                     -
0                0       86694.0        47169.9                -45.59
0                0       97401.0          50361                 -48.3

  Change Base pc.
0               0
0           18.46
0           65.69
0          575.02
0          566.72
0           667.4
0          603.16
0           13.51
0            N.A.
0            N.A.
```

4.11 Continuous beam on three supports, loaded from the bottom flange by a point load, intermediately braced @4m laterally and rotationally - Case 11

[26]:
```python
caseCounter+=1

za=535.

L_d=4000.

psi=133.33/-400.0

C_1_psi=min(1.75-1.05*psi+0.3*psi**2,2.35) #Salvadori/SIA263


Mcr_CH=Mcr_CH=clarkHill_criticalMoment(E, G, Iz_BG, K_BG, Iw_BG, beta_BG, za,␣
↪L_d,
                                kphi=1.0,kv=1.0,
                                C_1=C_1_psi,C_2=0.0,C_3=0.53)/1e6 #kN.m; C_2=0␣
↪--> load position does not matter

Mcr_LTBeam=97401.
```

```python
Error_CH=round((Mcr_CH-Mcr_LTBeam)/Mcr_LTBeam*100.,2)

analysisRow=[caseCounter,'Cont., 3 Supp. , Warp. Free','Point Load','Laterally␣
↪and rotationally fixed @4m','za=535mm',
            0.0,
            Mcr_LTBeam,
            Mcr_CH,
            Error_CH,
            'N.A.'] #  <---- Not applicable

analysisTable=analysisTable.append(pd.
↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

[26]:

```
   Case               Boundary Cond.     Loading  \
0    1            Simple, Warp. Free  Dist. Load
0    2            Simple, Warp. Free  Dist. Load
0    3           Simple, Warp. Fixed  Dist. Load
0    4            Simple, Warp. Free  Dist. Load
0    5            Simple, Warp. Free  Dist. Load
0    6            Simple, Warp. Free  Dist. Load
0    7            Simple, Warp. Free  Dist. Load
0    8            Simple, Warp. Free  Dist. Load
0    9  Cont., 3 Supp. , Warp. Free  Dist. Load
0   10  Cont., 3 Supp. , Warp. Free  Point Load
0   11  Cont., 3 Supp. , Warp. Free  Point Load

                                         Restraints Load Position  \
0                       No Inter. Lateral Restraint  Shear Center
0                              No Lateral Restraint      za=355mm
0                              No Lateral Restraint      za=355mm
0               @8m lateral Fixed and rotation Fixed      za=355mm
0               @8m lateral Fixed and rotation flex.      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
0               Laterally and rotationally fixed @4m     za=-535mm
0               Laterally and rotationally fixed @4m     za=-535mm
0               Laterally and rotationally fixed @4m      za=535mm

        Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.  \
0                0        2058.0           2083                 1.21
0                0        2438.0           2466                 1.15
0                0        3410.0           3639                 6.72
0                0       13892.0          13261                -4.54
```

```
0                0       13721.0              -                      -
0                0       15793.0              -                      -
0     Constant 2160       14471.0              -                      -
0   Variable -30828        2336.0              -                      -
0                0       86694.0        47169.9                 -45.59
0                0       97401.0          50361                  -48.3
0                0       97401.0        78625.7                 -19.28

  Change Base pc.
0               0
0           18.46
0           65.69
0          575.02
0          566.72
0           667.4
0          603.16
0           13.51
0            N.A.
0            N.A.
0            N.A.
```

4.12 Continuous beam on three supports, loaded from the bottom flange by a point load, intermediately braced @4m laterally and rotationally, and the top flange braced throughout - Case 12

[27]:

```python
caseCounter+=1

za=-535.

L_d=4000.

psi=133.33/-400.0

C_1_psi=min(1.75-1.05*psi+0.3*psi**2,2.35) #Salvadori/SIA263


Mcr_CH='-' # Out of scope of CH

Mcr_LTBeam=111048.

Error_CH='-' # out of scope of CH

analysisRow=[caseCounter,'Cont., 3 Supp. , Warp. Free','Point Load','Laterally␣
 ↪and rotationally fixed @4m, Top flange throughout','za=-535mm',
             0.0,
             Mcr_LTBeam,
```

```
            Mcr_CH,
            Error_CH,
            'N.A.'] #  <---- Not applicable

analysisTable=analysisTable.append(pd.
 ↪DataFrame([analysisRow],columns=analysisTable_header))

analysisTable
```

```
[27]:   Case              Boundary Cond.     Loading  \
     0     1           Simple, Warp. Free  Dist. Load
     0     2           Simple, Warp. Free  Dist. Load
     0     3          Simple, Warp. Fixed  Dist. Load
     0     4           Simple, Warp. Free  Dist. Load
     0     5           Simple, Warp. Free  Dist. Load
     0     6           Simple, Warp. Free  Dist. Load
     0     7           Simple, Warp. Free  Dist. Load
     0     8           Simple, Warp. Free  Dist. Load
     0     9  Cont., 3 Supp. , Warp. Free  Dist. Load
     0    10  Cont., 3 Supp. , Warp. Free  Point Load
     0    11  Cont., 3 Supp. , Warp. Free  Point Load
     0    12  Cont., 3 Supp. , Warp. Free  Point Load

                                           Restraints Load Position  \
     0                     No Inter. Lateral Restraint  Shear Center
     0                            No Lateral Restraint      za=355mm
     0                            No Lateral Restraint      za=355mm
     0             @8m lateral Fixed and rotation Fixed      za=355mm
     0             @8m lateral Fixed and rotation flex.      za=355mm
     0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
     0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
     0  Laterally Fixed throughout and rotation flex. @8m      za=355mm
     0             Laterally and rotationally fixed @4m     za=-535mm
     0             Laterally and rotationally fixed @4m     za=-535mm
     0             Laterally and rotationally fixed @4m      za=535mm
     0  Laterally and rotationally fixed @4m, Top flan…     za=-535mm

          Axial Load  M_cr_LTBeamN M_cr_ClarkHill Clark Hill Error pc.  \
     0             0        2058.0           2083                 1.21
     0             0        2438.0           2466                 1.15
     0             0        3410.0           3639                 6.72
     0             0       13892.0          13261                -4.54
     0             0       13721.0              -                    -
     0             0       15793.0              -                    -
     0    Constant 2160    14471.0              -                    -
     0  Variable -30828     2336.0              -                    -
     0             0       86694.0        47169.9               -45.59
```

```
0                 0      97401.0        50361                 -48.3
0                 0      97401.0      78625.7                -19.28
0                 0     111048.0            -                     -

  Change Base pc.
0               0
0           18.46
0           65.69
0          575.02
0          566.72
0           667.4
0          603.16
0           13.51
0            N.A.
0            N.A.
0            N.A.
0            N.A.
```

5 Some afterthoughts

5.1 Bending-moment and axial-load interaction in code verifications (SIA263)

Although *LTBeamN* allows us to calculate in a refined way the critical multiplier(μ_{cr}) associated with a moment and an axial load acting concurrently in our element (*e.g.* option 1, from before), code verifications impose a different approach. The code axial-load and moment interaction equation tries to simplify things by considering an approximate the interaction formula. Roughly reproducing the SIA263, the interaction formula(§5.1.9.1) looks like this:

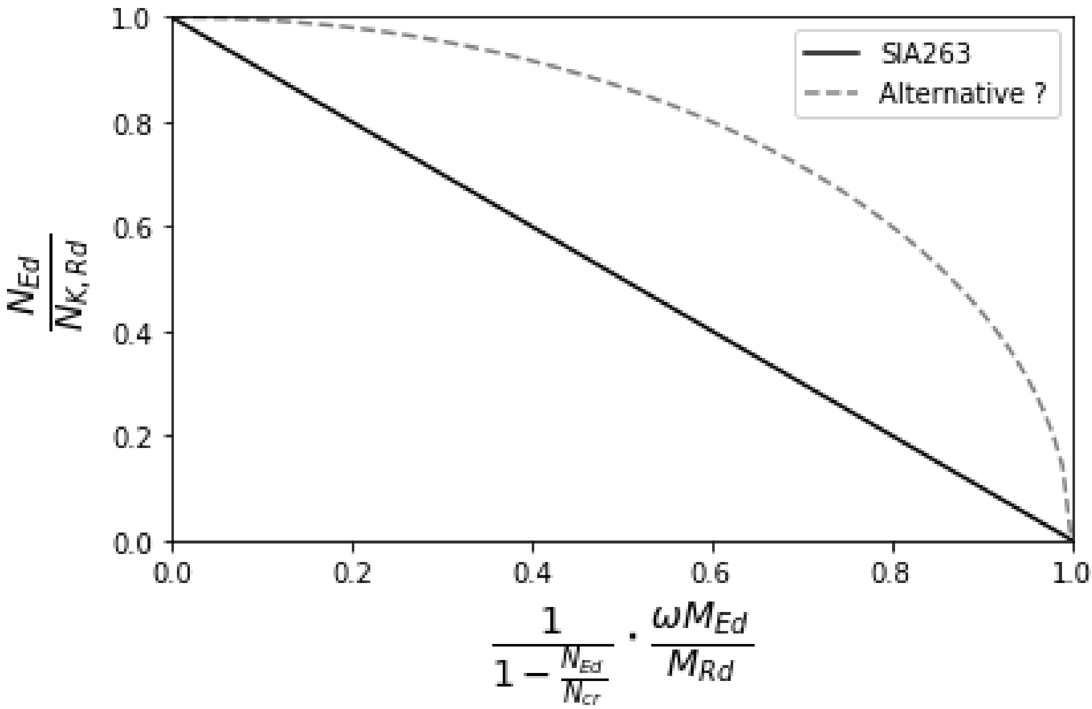
$$\frac{N_{Ed}}{N_{K,Rd}} + \frac{1}{1-\frac{N_{Ed}}{N_{cr}}} \cdot \frac{\omega M_{Ed}}{M_{Rd}} \leq 1.0 \tag{10}$$

with ω a factor to take into account the moment distribution in the element(not to be confused with the normalized sectorial coordinate in the last lecture); M_{Rd} the resisting moment, which should be $M_{D,Rd} = \chi_D M_{Rd}$ if lateral torsional buckling is not restrained; $N_{K,Rd}$ the axial resistance accounting for lateral buckling; N_{cr} the critical buckling load; and $1/(1-N_{Ed}/N_{cr})$ an amplification factor for 2nd order moments.

Adding to the complexity of the interaction axial load and bending moment interaction are other factors, like the distribution of residual stresses in the cross section, and imperfections in the member. Because of this complexity, the interaction formula is conservative in its shape, when one thinks of the other alternatives that could be considered. Here, however, is where experiments guide the decision and the data shows that a linear relation is the most appropriate one.

[28]:
```
rightSide=np.linspace(0,1,100)
leftSide_SIA263=1-rightSide
```

```
leftSide_alternative=(1-rightSide**2)**0.5 #Quadratic interaction between left␣
↪and right terms

plt.plot(rightSide,leftSide_SIA263,color='0.0',label='SIA263')
plt.plot(rightSide,leftSide_alternative,color='0.5',ls='--',label='Alternative ?
↪')
plt.xlim(0,1)
plt.ylim(0,1)
plt.xlabel(r'$\frac{1}{1-\frac{N_{Ed}}{N_{cr}}}\cdot\frac{\omega␣
↪M_{Ed}}{M_{Rd}}$',fontsize=20)
plt.ylabel(r'$\frac{N_{Ed}}{N_{K,Rd}}$',fontsize=20),
plt.legend(loc='upper right')
plt.tight_layout()
```



Aside from the shape of the interaction, the fact that we use an empirical formula to calculate the resistance of the interaction, gives way to multiple interpretations of how this formula should be used. An interesting discussion on this fact can be found in the *TGC-10*, §6.3.2 . There, it is recommended that, not only should the critical axial load and critical lateral torsional buckling loads be calculated independently of each other, but also that the moment diagram for the LTB calculation should be uniform. Paraphrasing what is stated in the *TGC*, two equations should be used to verify for the interaction:

$$\frac{N_{Ed}}{N_{K,Rd}} + \frac{1}{1 - \frac{N_{Ed}}{N_{cr}}} \cdot \frac{\omega M_{Ed,max}}{M_{D,Rd,min}} \leq 1.0 \tag{11}$$

$$M_{Ed,max} \leq M_{D,Rd}(C_1) \tag{12}$$

where, $M_{D,Rd,min} = M_{D,Rd}(C_1 = 1.0)$.

So, in conclusion, what should we use *LTBeamN* for:

1. Stick to cases without axial load interaction. If you end up using *LTBeamN* in these situations, always make the most conservative of assumptions (*e.g.*, uniform moment), and calculate the critical axial load and moment independently of each other;
2. Use *LTBeamN* whenever you have cases that you need to calculate the moment strength with the *LTB* reduction factor, when either the boundary conditions, load position, load distribution, and lateral restraints, start falling outside the scope of the simplified formulas. In short, if you start feeling uncomfortable about the assumptions you are using for C_1,C_2 and C_3, then you should move to *LTBeamN*.

$$\chi_D = \frac{1}{\Phi_D + \sqrt{\Phi_D^2 - \bar{\lambda}_D^2}} \leq 1 \quad ; \quad \Phi_D = 0.5\left(1 + \alpha_D\left(\bar{\lambda}_D - 0.4\right) + \bar{\lambda}_D^2\right) \quad ; \quad \bar{\lambda}_D = \sqrt{\frac{M_{Rd}}{M_{cr}}} \tag{13}$$

5.2 Python comments

The Python *Pandas* library gives us, as a bonus, the possibility of exporting the table we just made in LaTeX form:

[29]:
```
print(analysisTable.to_latex(index=False))
```

```
\begin{tabular}{lllllrlll}
\toprule
Case &                Boundary Cond. &     Loading &
Restraints & Load Position &       Axial Load &  M\_cr\_LTBeamN &
M\_cr\_ClarkHill & Clark Hill Error pc. & Change Base pc. \\
\midrule
   1 &           Simple, Warp. Free &  Dist. Load &                        No
Inter. Lateral Restraint &  Shear Center &                0 &        2058.0 &
2083 &                 1.21 &               0 \\
   2 &           Simple, Warp. Free &  Dist. Load &
No Lateral Restraint &      za=355mm &                0 &        2438.0 &
2466 &                 1.15 &           18.46 \\
   3 &          Simple, Warp. Fixed &  Dist. Load &
No Lateral Restraint &      za=355mm &                0 &        3410.0 &
3639 &                 6.72 &           65.69 \\
   4 &           Simple, Warp. Free &  Dist. Load &              @8m lateral
Fixed and rotation Fixed &      za=355mm &                0 &       13892.0 &
13261 &                -4.54 &          575.02 \\
```

```
   5 &           Simple, Warp. Free &  Dist. Load &               @8m lateral
Fixed and rotation flex. &      za=355mm &                0 &       13721.0 &
- &                    - &          566.72 \\
   6 &           Simple, Warp. Free &  Dist. Load &  Laterally Fixed throughout
and rotation flex. @8m &      za=355mm &                0 &       15793.0 &
- &                    - &           667.4 \\
   7 &           Simple, Warp. Free &  Dist. Load &  Laterally Fixed throughout
and rotation flex. @8m &      za=355mm &    Constant 2160 &       14471.0 &
- &                    - &          603.16 \\
   8 &           Simple, Warp. Free &  Dist. Load &  Laterally Fixed throughout
and rotation flex. @8m &      za=355mm &  Variable -30828 &        2336.0 &
- &                    - &           13.51 \\
   9 &  Cont., 3 Supp. , Warp. Free &  Dist. Load &               Laterally and
rotationally fixed @4m &     za=-535mm &                0 &       86694.0 &
47169.9 &               -45.59 &             N.A. \\
  10 &  Cont., 3 Supp. , Warp. Free &  Point Load &               Laterally and
rotationally fixed @4m &     za=-535mm &                0 &       97401.0 &
50361 &                -48.3 &             N.A. \\
  11 &  Cont., 3 Supp. , Warp. Free &  Point Load &               Laterally and
rotationally fixed @4m &      za=535mm &                0 &       97401.0 &
78625.7 &               -19.28 &             N.A. \\
  12 &  Cont., 3 Supp. , Warp. Free &  Point Load &  Laterally and rotationally
fixed @4m, Top flan… &     za=-535mm &                0 &      111048.0 &
- &                    - &            N.A. \\
\bottomrule
\end{tabular}
```

If you are an Excel person you can also export it to a *csv* format:

[30]:
```python
analysisTable.to_csv('LTBeamN_analysisTable.csv',index=False)
```

Feel free to play around with the code. Change, break things, and put them back together in a way that makes sense for you. Whenever you get stuck always remember that, at least in these situations, Google is your friend.

[ ]:
```python

```