surcharge_wall

Unknown Author

February 11, 2014

In [1]:
```
%pylab inline
from scipy.integrate import trapz,quad,romberg,dblquad
```
```
Populating the interactive namespace from numpy and matplotlib
```

1 Navfac DM7.02

The following formula is the classic one from the Navfac DM7.02 manual for the calculation of the horizontal stress on a wall caused by a line load. The second function integrates the first one to provide a formula for the area load.

$$\sigma_x = \begin{cases} \frac{P}{H} \cdot \frac{0.2 \cdot n}{(0.16+n^2)^2} & m \leq 0.4 \\ \frac{P}{H} \cdot \frac{0.28 \cdot m^2 \cdot n}{(m^2+n^2)^2} & m > 0.4 \end{cases}$$

$$m = \frac{x}{h} \quad n = \frac{z}{h}$$

In [2]:
```
def LL_navfac(P,x,H,z):
    m = x / H
    n = z / H
    if m<=0.4:
        return (P/H)*0.20*n/(0.16+n**2)**2
    else:
        return (P/H)*1.28*m**2*n/(m**2+n**2)**2
```

In [3]:
```
def SL_navfac(q,x1,x2,H,z):
    return quad(lambda x: LL_navfac(q,x,H,z),x1,x2)[0]
```

2 Boussinesq ν = 0.5

The same principle as above but with the classical Boussinesq formula:

$$\sigma_x = \frac{2P}{\pi} \frac{x^2 z}{(x^2+z^2)^2}$$

In [4]:
```
def LL_boussinesq(P,x,H,z):
    return (2.0*P/pi)*x**2*z/(x**2+z**2)**2
```

In [5]:
```
def SL_boussinesq(q,x1,x2,H,z):
    return quad(lambda x: LL_boussinesq(q,x,H,z),x1,x2)[0]
```

3 Original boussinesq with ν parameter

The original Boussinesq equation with poisson coefficient.

$$\sigma_x = \frac{Q}{2 \cdot \pi} \cdot \left(\frac{3 \cdot x^2 \cdot z}{R^5} - (1 - 2 \cdot \nu) \cdot \left(\frac{x^2 - y^2}{R \cdot r^2 \cdot (R + z)} + \frac{y^2 \cdot z}{R^3 \cdot r^2} \right) \right)$$

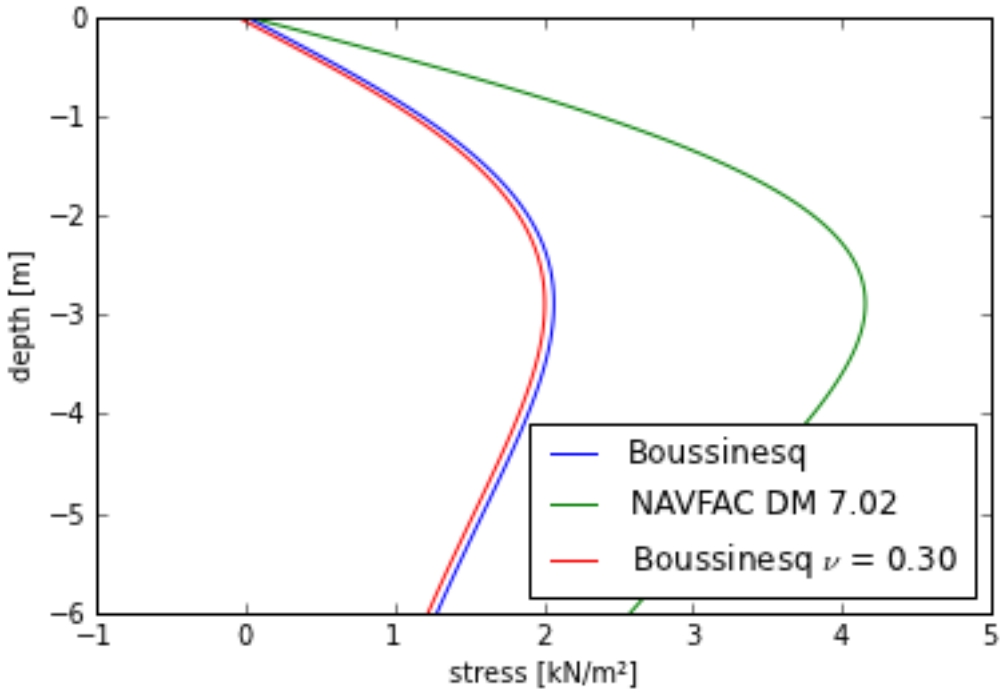
In [15]:
```
def b(Q, x, y, z, nu):
    R = sqrt(x**2+y**2+z**2)
    r = sqrt(x**2+y**2)
    return Q/(2*pi)*(3*x**2*z/R**5-(1-2*nu)*((x**2-y**2)/ \
           (R*r**2*(R+z))+y**2*z/(R**3*r**2)))

def LL_boussinesq_orgin(P,x,H,z,nu):
    return quad(lambda y:  b(P, x, y, z, nu), -100.0,100.0)[0]

#NOTE: function is way to slow
def SL_boussinesq_orgin(q,x1,x2,H,z,nu):
    #return quad(lambda x: LL_boussinesq_orgin(q,x,H,z,nu),x1,x2)[0]
    return dblquad(lambda y, x: b(q, x, y, z, nu), x1, x2, \
                   lambda x: -50.0, lambda x: 50.0)[0]
```

4 Example calculation: Point load

Wall height: 6.0 m Line load: 50 kN/m located 5 m behind the wall.

In [7]:
```
P = 50.0  #load 50 kN/m
x = 5.0 # 5 m from wall
H = 6.0 # 6 m heigth wal
nu =0.3
z = np.arange(0.0,H+H/100.0, H/100.0)
sigma1 = [LL_boussinesq(P,x,H,i) for i in z]
sigma2 = [LL_navfac(P,x,H,i) for i in z]
sigma3 = [LL_boussinesq_orgin(P,x,H,i,nu) for i in z]

xlabel(u"stress [kN/m$^2$]")
ylabel("depth [m]")

plot(sigma1,-z,label="Boussinesq")
plot(sigma2,-z,label="NAVFAC DM 7.02")
plot(sigma3,-z,label="Boussinesq $\\nu$ = %.2f" % nu)

legend(loc=4)
```

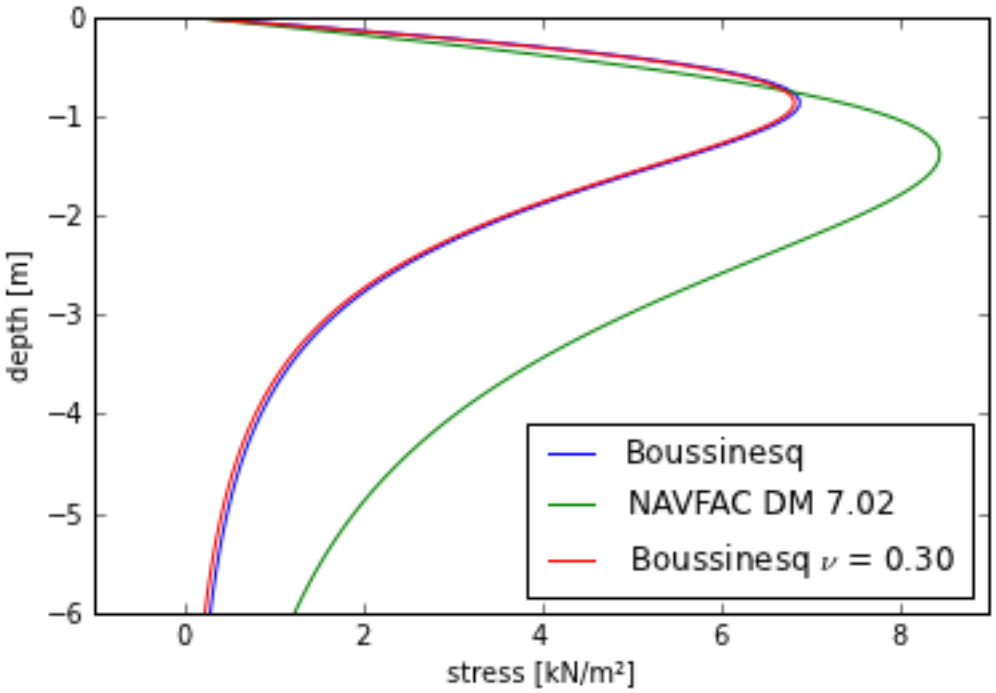
Out [7]:
```
<matplotlib.legend.Legend at 0x3341350>
```



The NAVFAC equations clearly uses a factor 2 for the mirror effect which was suggested by Midlin. (if m>0.4) The reults from the NAVFAC formula are closer to the Boussinesq ones for a load closer to the wall:

In [10]:
```
P = 50.0  #load 50 kN/m
x = 1.5 # 1.5 m from wall
H = 6.0 # 6 m heigth wal
nu =0.3
z = np.arange(0.0,H+H/100.0, H/100.0)
sigma1 = [LL_boussinesq(P,x,H,i) for i in z]
sigma2 = [LL_navfac(P,x,H,i) for i in z]
sigma3 = [LL_boussinesq_orgin(P,x,H,i,nu) for i in z]

xlabel(u"stress [kN/m$^2$]")
ylabel("depth [m]")

plot(sigma1,-z,label="Boussinesq")
plot(sigma2,-z,label="NAVFAC DM 7.02")
plot(sigma3,-z,label="Boussinesq $\\nu$ = %.2f" % nu)

legend(loc=4)
```

Out [10]:
```
<matplotlib.legend.Legend at 0x4067690>
```



5 Area load

The same comparison is done for an uniform area load, infinitly extended behind the wall. For example if we have a load of 50 kN/m^2 and a soil with $\phi = 30$ we would except the following at rest pressure on the wall:

$$\phi = 30 \cdot \frac{\pi}{180} = 0.524$$

$$K_0 = 1 - \sin(\phi) = 0.500$$

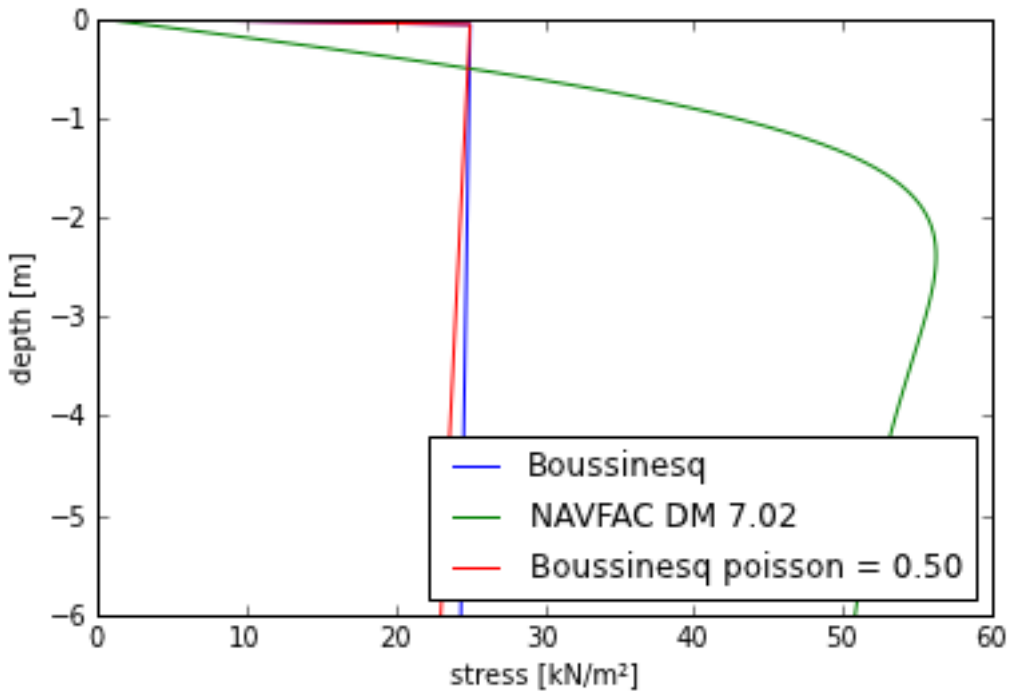
$$\sigma_x = 50.0 \cdot K_0 = 25.000$$

We can do the same with the Boussinesq and the NAVFAC formula:

In [16]:
```
Q = 50.0  #load 50 kN/m^2
H = 6.0
x1 = 0.0 # load starting behind the wall
x2 = 50*H # 'infinitly' extendend
nu =0.50
z = np.arange(0.0,H+H/100.0, H/100.0)
sigma1 = [SL_boussinesq(Q,x1,x2,H,i) for i in z]
sigma2 = [SL_navfac(Q,x1,x2,H,i) for i in z]
sigma3 = [SL_boussinesq_orgin(Q,x1,x2,H,i,nu) for i in z]

xlabel(u"stress [kN/m^2]")
ylabel("depth [m]")

plot(sigma1,-z,label="Boussinesq")
plot(sigma2,-z,label="NAVFAC DM 7.02")
plot(sigma3,-z,label="Boussinesq poisson = %.2f" % nu)

legend(loc=4)
```

```
<matplotlib.legend.Legend at 0x7f5530b40dd0>
```

Out [16]:



Conclusion: The classical Boussinesq formula gives 25 kN/m^2 lateral presure whilest the NAVFAC formula gives more than the double value.