In [ ]:
```python
import numpy as np
import math

# function expression
func = lambda x: x[0]**2 + x[1]**2 + x[2]**2
# Gradients
delf_deld = lambda x: 2.0*x[0]
delf_dels = lambda x: np.array([2.0*x[1], 2.0*x[2]])
delh_dels = lambda x: np.array([[2.0*x[1]/5.0, 2*x[2]/25.0],[1.0,-1.0]])
delh_deld = lambda x: np.array([[x[0]/2.0], [1.0]])
dfdd = lambda x: delf_deld(x) - np.matmul(np.matmul(delf_dels(x), np.linalg.inv(delh_dels(x))), delh_deld(x))

# error tolerance
err = 0.0001
# iter
k = 0

#line search
def linsrch(xvec, dfdd_vec):
    a = 1
    x1 = np.array([0.0,0.0,0.0])
    x1[0]= xvec[0]-a*dfdd_vec
    x1[1:3]= xvec[1:3] + a* np.transpose(np.matmul(np.matmul(np.linalg.inv(delh_dels(xvec)),delh_deld(xvec)),np.transpose([dfdd(xvec)])))
    while func(x1) > (func(xvec)-a*0.3*dfdd_vec**2):
        a = 0.5*a
        x1[0]= xvec[0]-a*dfdd_vec
        x1[1:3]= xvec[1:3] + a* np.transpose(np.matmul(np.matmul(np.linalg.inv(delh_dels(xvec)),delh_deld(xvec)),np.transpose([dfdd(xvec)])))
    return a

#solver algorithm
def solv(x):
    while np.linalg.norm(np.array([[x[0]**2/4.0+x[1]**2/5.0+x[2]**2/25.0-1.0], [x[0]+x[1]-x[2]]]))  > 0.001:
        s =  np.transpose([x[1:3]]) - np.matmul(np.linalg.inv(delh_dels(x)), np.array([[x[0]**2/4.0 + x[1]**2/5.0 + x[2]**2/25.0-1.0], [x[0]+x[1]-x[2]]]))
        x=np.append(x[0], s)
    return x

#Initial guess
x_k=np.array([0.00, 2.04, 2.04])


#minimizing the gradient
while np.linalg.norm(dfdd(x_k)) > err:
    # gradient
    dfdd_k = dfdd(x_k)
    # line search
    a_k = linsrch(x_k, dfdd_k)
    # d update
    d_k = x_k[0] - a_k * dfdd_k
    # s update
    s_k = x_k[1:3] + a_k * np.transpose(  np.matmul(np.matmul(np.linalg.inv(delh_dels(x_k)), delh_deld(x_k)),  np.transpose(dfdd_k)) )
    x0 = np.append(d_k,s_k)
    # solve for s to update x
    x_k = solv(x0)
    print('iter = '+str(k))
    print('(x1,x2,x3)= '+str(x_k))
    print('err= '+ str(np.linalg.norm(dfdd(x_k))))
    print('err reduction= '+ str(np.linalg.norm(dfdd(x_k))))
    print('')
    k += 1
```

```
iter = 0
(x1,x2,x3)= [-0.68        2.01617305  1.33617305]
err= 3.0235002827975617
err reduction= 3.0235002827975617

iter = 1
(x1,x2,x3)= [-1.43587507  1.55581318  0.11993811]
err= 1.1249473010876239
err reduction= 1.1249473010876239

iter = 2
(x1,x2,x3)= [-1.50618428  1.47107334 -0.03511094]
err= 0.6244024286014094
err reduction= 0.6244024286014094

iter = 3
(x1,x2,x3)= [-1.54520943  1.41852231 -0.12668712]
err= 0.28519642458457417
err reduction= 0.28519642458457417

iter = 4
(x1,x2,x3)= [-1.56303421  1.39348586 -0.16954834]
err= 0.11363187078634773
err reduction= 0.11363187078634773

iter = 5
(x1,x2,x3)= [-1.5701362   1.38310274 -0.18703346]
err= 0.04104096276123803
err reduction= 0.04104096276123803

iter = 6
(x1,x2,x3)= [-1.57270126  1.37929035 -0.19341091]
err= 0.01417360598630557
err reduction= 0.01417360598630557

iter = 7
(x1,x2,x3)= [-1.57358711  1.37796577 -0.19562133]
err= 0.004811805928974788
err reduction= 0.004811805928974788

iter = 8
(x1,x2,x3)= [-1.57388785  1.37751515 -0.19637269]
err= 0.0016237312123528191
err reduction= 0.0016237312123528191

iter = 9
(x1,x2,x3)= [-1.57398933  1.37736298 -0.19662635]
err= 0.0005467939034589087
err reduction= 0.0005467939034589087

iter = 10
(x1,x2,x3)= [-1.5740235   1.37731173 -0.19671178]
err= 0.00018400512394967095
err reduction= 0.00018400512394967095

iter = 11
(x1,x2,x3)= [-1.574035    1.37729448 -0.19674053]
err= 6.190617312595847e-05
err reduction= 6.190617312595847e-05
```