In [59]:
```
os.system("date")
```

```
Mon May 20 12:51:51 PM EDT 2024
```

Out[59]: 0

Example plots with the IPython notebook

So I just use the lhapdf library for loading the PDFs and plot them

1. Imports and defs

Here are couple of modules that I usually load. The important one if lhapdf of course. Note the path to the python lhapdf library has to be added via the sys.path.append command

In [1]:
```
import os
import sys
import glob
import numpy as np
from math import *
import matplotlib.pyplot as plt
```

In [2]:
```
%matplotlib inline
```

This is the key to import the lhapdf library

In [3]:
```
os.system("python --version")
```

```
Python 3.10.12
```

Out[3]: 0

In [4]:
```
import sys
sys.path.insert(0,'/home/vboxuser/LHAPDF/LHAPDF-6.5.3/wrappers/python/bui
#sys.path.insert(0,'/home/vboxuser/Software/deps/lhapdf/lib/python3.10/si
```

In [5]:
```
import lhapdf
print(lhapdf.version())
```

```
6.5.3
```

In [6]:
```
lhapdf. __file__
```

Out[6]: '/home/vboxuser/LHAPDF/LHAPDF-6.5.3/wrappers/python/build/lhapdf/lhapdf.so'

In [7]:
```
print(lhapdf.paths())
```

```
['/home/vboxuser/Software/deps/lhapdf/share/LHAPDF']
```

In [8]:
```
#lhapdf.pathsAppend('/home/olness/Dropbox/PYTHON/LHAPDF/xFitterPDF/')
#lhapdf.pathsAppend('/home/olness/Dropbox/PYTHON/LHAPDF/xFitterPDF/')
os.listdir('./PDFDIR/')
```

Out[8]:
```
['NNPDF31_nnlo_as_0118',
 'nCTEQ15FullNuc_208_82',
 'lhapdf.conf',
 'pdfsets.index',
 'nCTEQ15FullNuc_56_26',
 'NNPDF40_nnlo_as_01180',
 'MSHT20an3lo_as118',
 'ABMP16als118_5_nnlo',
 'HERAPDF20_NNLO_VAR',
 'rsFree',
 'CT18ANNLO',
 'rsFixed',
 'nCTEQ15FullNuc_1_1',
 'NNPDF30_nlo_as_0118',
 'nCTEQ15FullNuc_184_74',
 'MSHT20nnlo_as118',
 'PDF4LHC21_40',
 'CT18NNLO',
 'HERAPDF20_NNLO_EIG']
```

In [9]:
```
#lhapdf.pathsAppend('/home/olness/Dropbox/PYTHON/LHAPDF/xFitterPDF/')
#lhapdf.pathsAppend('/home/olness/Dropbox/PYTHON/LHAPDF/xFitterPDF/')
os.listdir('/home/vboxuser/Software/deps/lhapdf/share/LHAPDF')
```

Out[9]:
```
['NNPDF31_nnlo_as_0118',
 'nCTEQ15FullNuc_208_82',
 'lhapdf.conf',
 'pdfsets.index',
 'nCTEQ15FullNuc_56_26',
 'NNPDF40_nnlo_as_01180',
 'MSHT20an3lo_as118',
 'ABMP16als118_5_nnlo',
 'HERAPDF20_NNLO_VAR',
 'rsFree',
 'CT18ANNLO',
 'rsFixed',
 'nCTEQ15FullNuc_1_1',
 'NNPDF30_nlo_as_0118',
 'nCTEQ15FullNuc_184_74',
 'MSHT20nnlo_as118',
 'PDF4LHC21_40',
 'CT18NNLO',
 'HERAPDF20_NNLO_EIG']
```

In [10]:
```
cwd = os.getcwd()
cwd
```

Out[10]:
```
'/home/vboxuser/PYTHON'
```

In [11]:
```
lhapdf.pathsAppend('./PDFDIR/')
```

In [12]:
```
print(lhapdf.paths())
```

```
['/home/vboxuser/Software/deps/lhapdf/share/LHAPDF', './PDFDIR/', '/home/v
boxuser/Software/deps/lhapdf/share/LHAPDF']
```

In [13]:
```
os.listdir('./PDFDIR/')
```

Out[13]:
```
['NNPDF31_nnlo_as_0118',
 'nCTEQ15FullNuc_208_82',
 'lhapdf.conf',
 'pdfsets.index',
 'nCTEQ15FullNuc_56_26',
 'NNPDF40_nnlo_as_01180',
 'MSHT20an3lo_as118',
 'ABMP16als118_5_nnlo',
 'HERAPDF20_NNLO_VAR',
 'rsFree',
 'CT18ANNLO',
 'rsFixed',
 'nCTEQ15FullNuc_1_1',
 'NNPDF30_nlo_as_0118',
 'nCTEQ15FullNuc_184_74',
 'MSHT20nnlo_as118',
 'PDF4LHC21_40',
 'CT18NNLO',
 'HERAPDF20_NNLO_EIG']
```

In [14]:
```
print(lhapdf.paths())
```

```
['/home/vboxuser/Software/deps/lhapdf/share/LHAPDF', './PDFDIR/', '/home/v
boxuser/Software/deps/lhapdf/share/LHAPDF']
```

In [15]:
```
os.listdir('/usr/local/share/LHAPDF')
```

Out[15]:
```
['lhapdf.conf',
 'pdfsets.index',
 'SEE_Software\\deps\\lhapdf\\share\\LHAPDF.txt']
```

In [16]:
```
print("Path at terminal when executing this file")
print(os.getcwd() + "\n")
```

```
Path at terminal when executing this file
/home/vboxuser/PYTHON
```

In [17]:
```
print(glob.glob("*"))
```

```
['Read_LHAPDF.pdf', 'Read_LHAPDF.ipynb', 'PDFDIR', 'bottom05.pdf']
```

In [18]:
```
print(lhapdf.paths())
```

```
['/home/vboxuser/Software/deps/lhapdf/share/LHAPDF', './PDFDIR/', '/home/v
boxuser/Software/deps/lhapdf/share/LHAPDF']
```

1.1 Functions and PDFs

Let's load all the PDFs sets that we need and in particular the nCTEQ15 full lead, iron and proton (deuterium).

In [19]:
```
proton_ncteq = lhapdf.getPDFSet('nCTEQ15FullNuc_1_1')
```

In [20]:
```
xf0 = lhapdf.getPDFSet('nCTEQ15FullNuc_1_1')
xf0=xf0.mkPDFs()
xf0[0].xfxQ(2,0.1,10.)
```

```
LHAPDF 6.5.3 loading all 1 PDFs in set nCTEQ15FullNuc_1_1
nCTEQ15FullNuc_1_1, version 1; 1 PDF members
```

Out[20]: 0.6504072208540947

In [ ]:

In [21]:
```
xf0 = lhapdf.getPDFSet('rsFixed')
xf0=xf0.mkPDFs()
xf0[0].xfxQ(2,0.1,10.)
```

```
LHAPDF 6.5.3 loading all 1 PDFs in set rsFixed
rsFixed, version 1; 1 PDF members
```

Out[21]: 0.66114855119381

In [22]:
```
xf1 = lhapdf.getPDFSet('rsFree')
xf1=xf1.mkPDFs()
xf1[0].xfxQ(2,0.1,10.)
```

```
LHAPDF 6.5.3 loading all 1 PDFs in set rsFree
rsFree, version 1; 1 PDF members
```

Out[22]: 0.6627243695017107

In [23]:
```
xf2 = lhapdf.getPDFSet('nCTEQ15FullNuc_1_1')
xf2=xf2.mkPDFs()
xf2[0].xfxQ(2,0.1,10.)
```

```
LHAPDF 6.5.3 loading all 1 PDFs in set nCTEQ15FullNuc_1_1
nCTEQ15FullNuc_1_1, version 1; 1 PDF members
```

Out[23]: 0.6504072208540947

In [24]:
```
xf3 = lhapdf.getPDFSet('NNPDF30_nlo_as_0118')
xf3=xf3.mkPDFs()
xf3[0].xfxQ(2,0.1,10.)
```

```
LHAPDF 6.5.3 loading all 101 PDFs in set NNPDF30_nlo_as_0118
```

Out[24]: 0.6200919115217465

```
NNPDF30_nlo_as_0118, version 2; 101 PDF members
```

In [25]:
```
xf4 = lhapdf.getPDFSet('CT18NNLO')
xf4=xf4.mkPDFs()
xf4[0].xfxQ(2,0.1,10.)
```

Out[25]: 0.6372213241191952

```
LHAPDF 6.5.3 loading all 59 PDFs in set CT18NNLO
CT18NNLO, version 1; 59 PDF members
```

In [26]:
```
xvalues = np.logspace(-3,0,100)
qvalues = np.logspace(0,2,500)
```

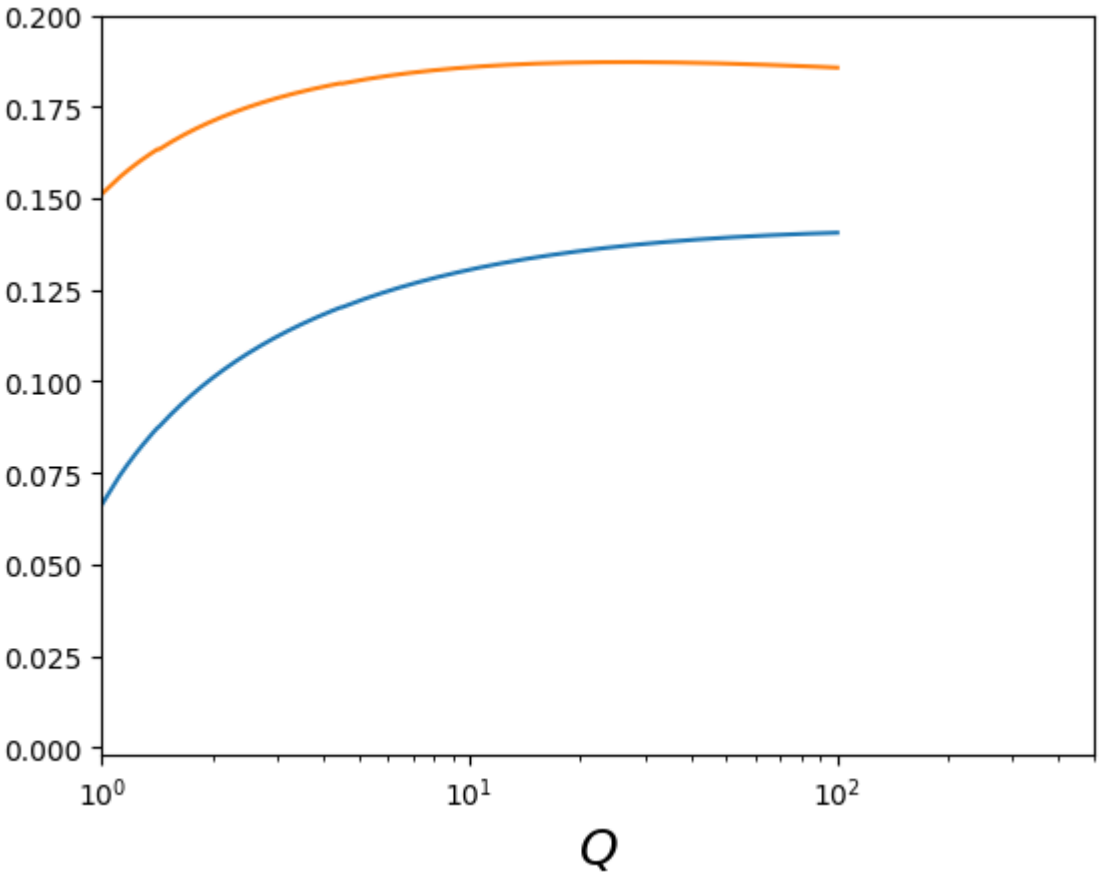
In [ ]:

In [ ]:

Make some plots

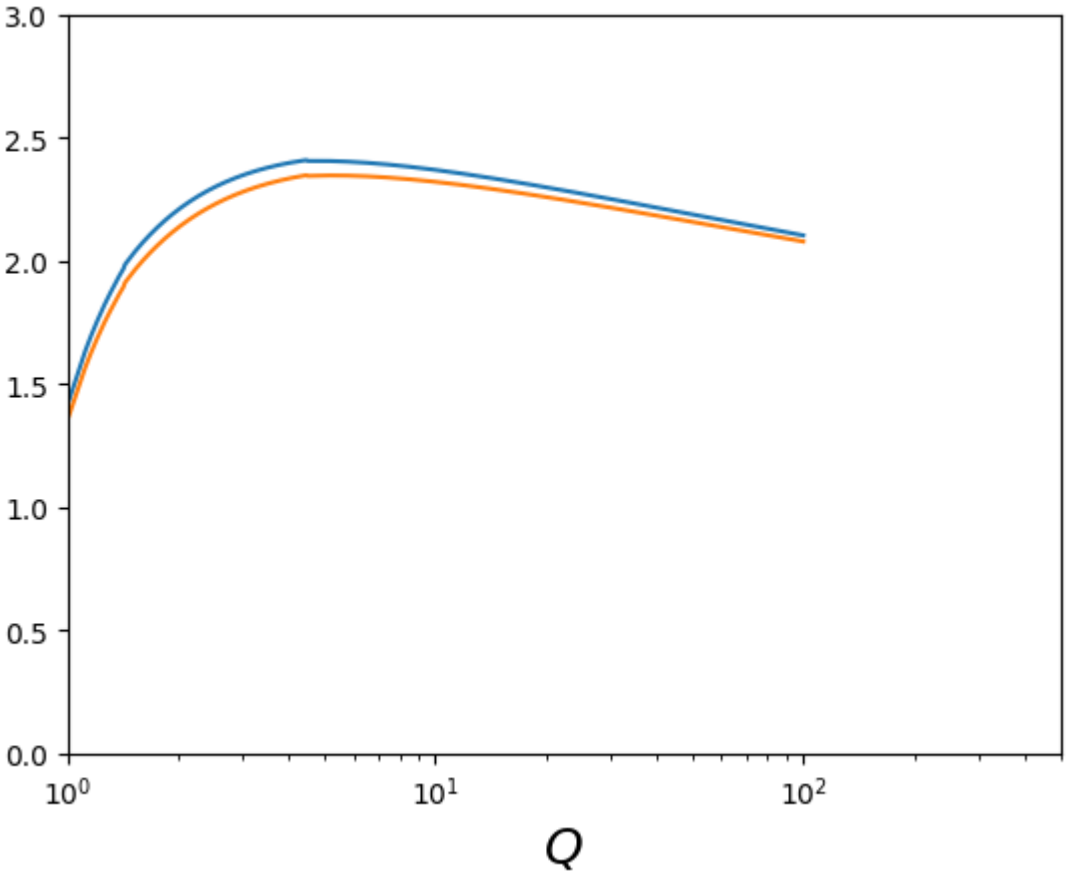
In [27]:
```
%matplotlib inline
```

In [28]:
```
istr=3;
ic=4;
ib=5;
x0=0.05;
sFix=[xf0[0].xfxQ(istr,x0,qq) for qq in qvalues]
sFree=[xf1[0].xfxQ(istr,x0,qq) for qq in qvalues]

plt.plot(qvalues,sFix)
plt.plot(qvalues,sFree)
plt.xscale("log")


plt.ylim((-0.002,0.20))
plt.xlim((1,500))


plt.xlabel('$Q $',fontsize=18)

fig = plt.gcf()

#fig.show()

# show(block=False)

fig.savefig('bottom05.pdf')
```

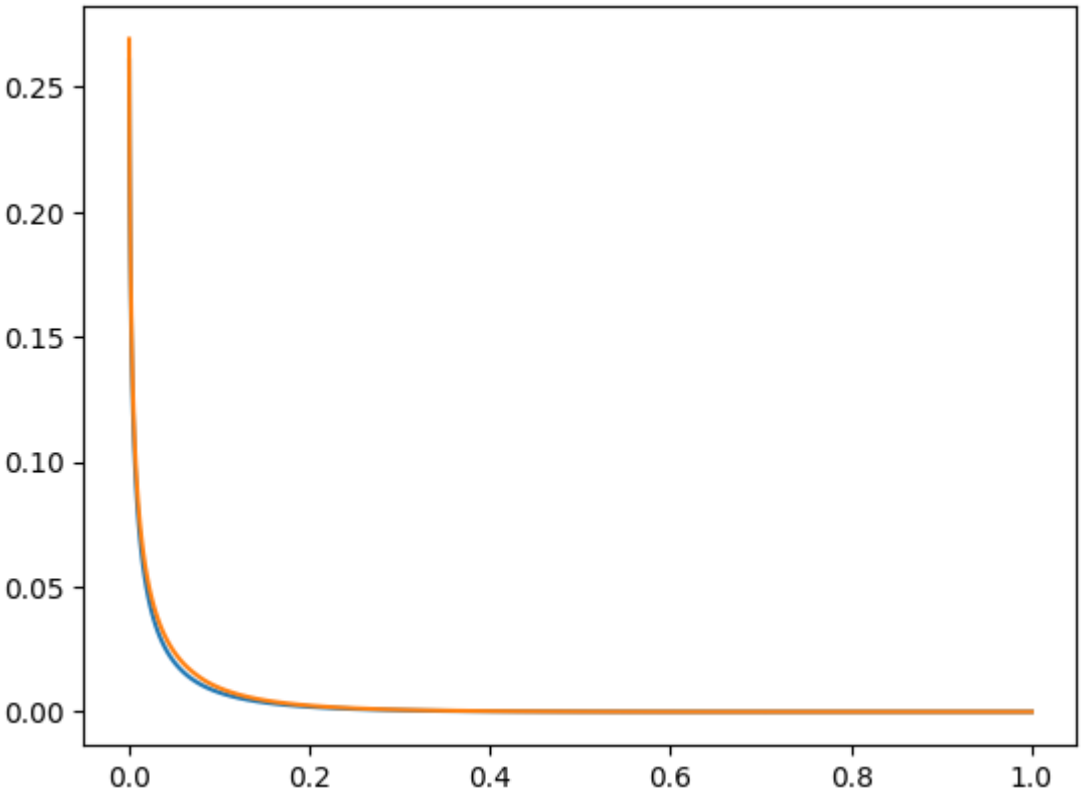


In [29]:
```
iglu=0;
istr=3;
ic=4;
ib=5;
```

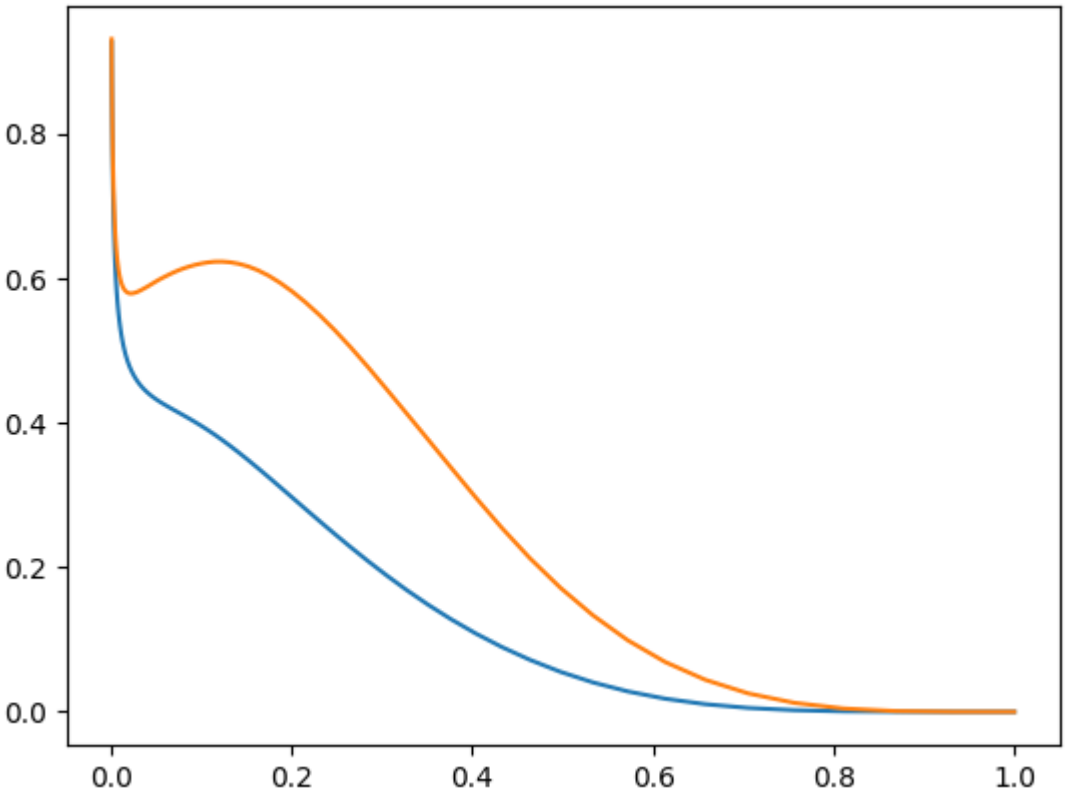
```
x0=0.05;
sFix=[xf0[0].xfxQ(iglu,x0,qq) for qq in qvalues]
sFree=[xf1[0].xfxQ(iglu,x0,qq) for qq in qvalues]

plt.plot(qvalues,sFix)
plt.plot(qvalues,sFree)
plt.xscale("log")


plt.ylim((-0.002,3.0))
plt.xlim((1,500))


plt.xlabel('$Q $',fontsize=18)

fig = plt.gcf()

#fig.show()

# show(block=False)

fig.savefig('bottom05.pdf')
```



In [30]:
```
s1=[xf2[0].xfxQ(5,xx,10.) for xx in xvalues]
s2=[xf3[0].xfxQ(5,xx,10.) for xx in xvalues]

plt.plot(xvalues,s1)
plt.plot(xvalues,s2)
plt.show()
```

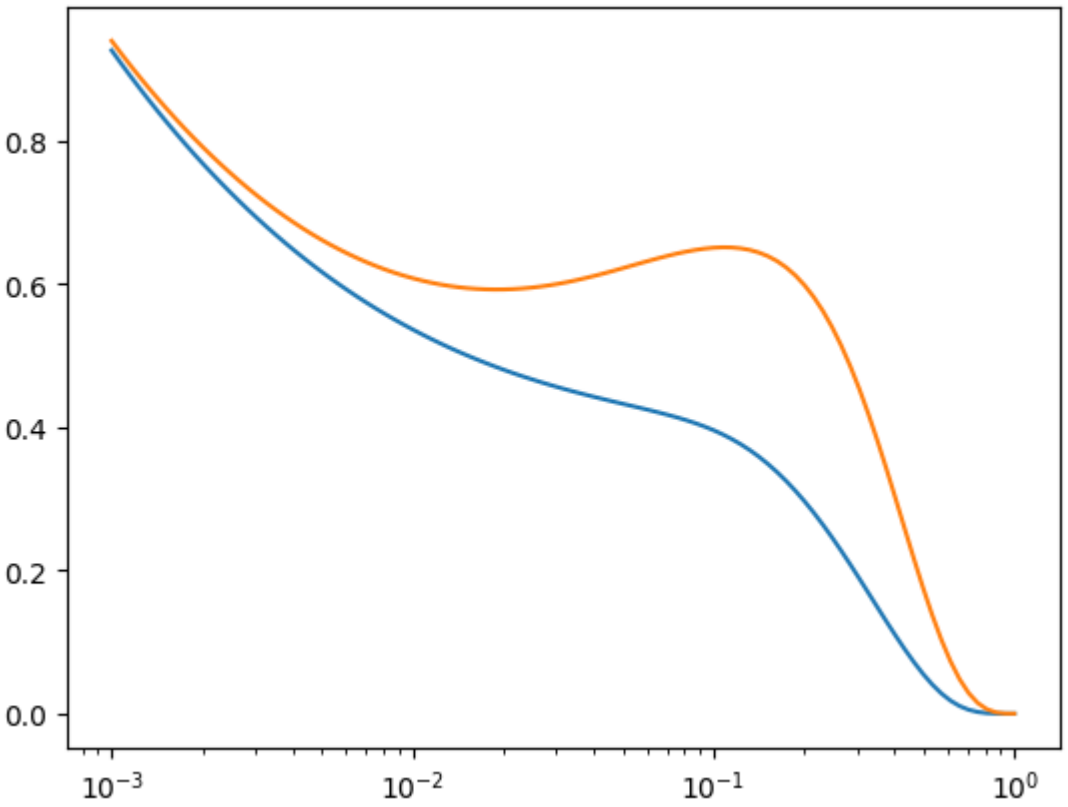


In [31]:
```
s1=[xf2[0].xfxQ(1,xx,10.) for xx in xvalues]
s2=[xf3[0].xfxQ(2,xx,10.) for xx in xvalues]

plt.plot(xvalues,s1)
plt.plot(xvalues,s2)
plt.show()
```

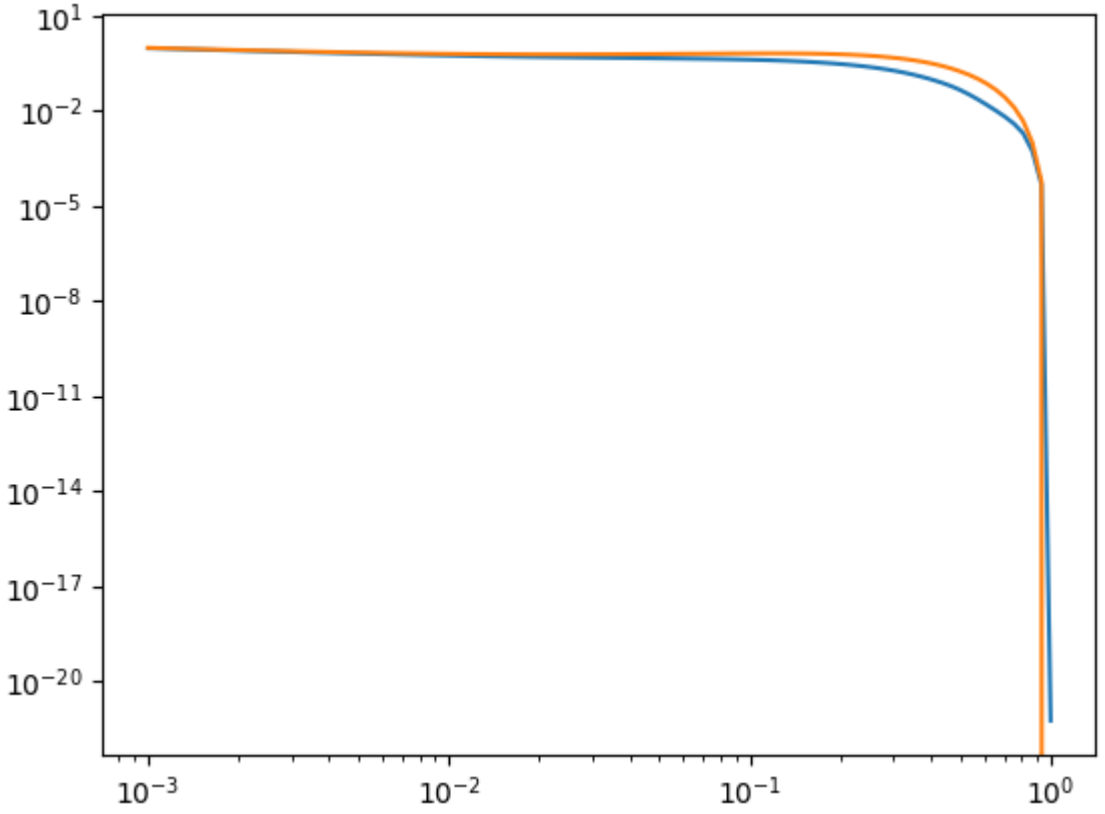


In [32]:
```
s1=[xf2[0].xfxQ(1,xx,10.) for xx in xvalues]
s2=[xf2[0].xfxQ(2,xx,10.) for xx in xvalues]
```

```
plt.xscale('log')

plt.plot(xvalues,s1)
plt.plot(xvalues,s2)
plt.show()
```



In [33]:
```
s1=[xf3[0].xfxQ(1,xx,10.) for xx in xvalues]
s2=[xf3[0].xfxQ(2,xx,10.) for xx in xvalues]

plt.xscale('log')
plt.yscale('log')

plt.plot(xvalues,s1)
plt.plot(xvalues,s2)
plt.show()
```



In [34]:
```
xf1[0].xfxQ(5,0.0001,4.5)
```

Out[34]:
```
-0.09360836147540727
```

In [35]:
```
t = np.arange(0.1, 1.0, 0.02)

t
```

Out[35]:
```
array([0.1 , 0.12, 0.14, 0.16, 0.18, 0.2 , 0.22, 0.24, 0.26, 0.28, 0.3 ,
       0.32, 0.34, 0.36, 0.38, 0.4 , 0.42, 0.44, 0.46, 0.48, 0.5 , 0.52,
       0.54, 0.56, 0.58, 0.6 , 0.62, 0.64, 0.66, 0.68, 0.7 , 0.72, 0.74,
       0.76, 0.78, 0.8 , 0.82, 0.84, 0.86, 0.88, 0.9 , 0.92, 0.94, 0.96,
       0.98])
```

In [36]:
```
# t = np.arange(0.1, 1.0, 0.02)
# s=xf1[0].xfxQ(2,t,10.)
# plot(t, s)
```

In [37]:
```
proton_tmp = lhapdf.getPDFSet('NNPDF30_nlo_as_0118')
```

In [38]:
```
proton_tmp = proton_tmp.mkPDFs()[0]
```

```
LHAPDF 6.5.3 loading all 101 PDFs in set NNPDF30_nlo_as_0118
NNPDF30_nlo_as_0118, version 2; 101 PDF members
```

In [39]:
```
# Let's load the proton
proton_ncteq = lhapdf.getPDFSet('nCTEQ15FullNuc_1_1')
proton_ncteq = proton_ncteq.mkPDFs()[0]
# Let's load the Full lead pedf sets
fulllead = lhapdf.getPDFSet('nCTEQ15FullNuc_208_82')
fullleads = fulllead.mkPDFs()
# Let's load th efull Iron
fulliron = lhapdf.getPDFSet('nCTEQ15FullNuc_56_26')
fullirons = fulliron.mkPDFs()
```

```
LHAPDF 6.5.3 loading all 1 PDFs in set nCTEQ15FullNuc_1_1
nCTEQ15FullNuc_1_1, version 1; 1 PDF members
LHAPDF 6.5.3 loading all 33 PDFs in set nCTEQ15FullNuc_208_82
nCTEQ15FullNuc_208_82, version 1; 33 PDF members
LHAPDF 6.5.3 loading all 33 PDFs in set nCTEQ15FullNuc_56_26
nCTEQ15FullNuc_56_26, version 1; 33 PDF members
```

In [40]:
```
# Let's just plot them now as an example. We first define the Q, and x va
```

In [41]:
```
xvalues = np.logspace(-3,0,100)
Qval = [5,10]
# select the flavour
flav = 2
```

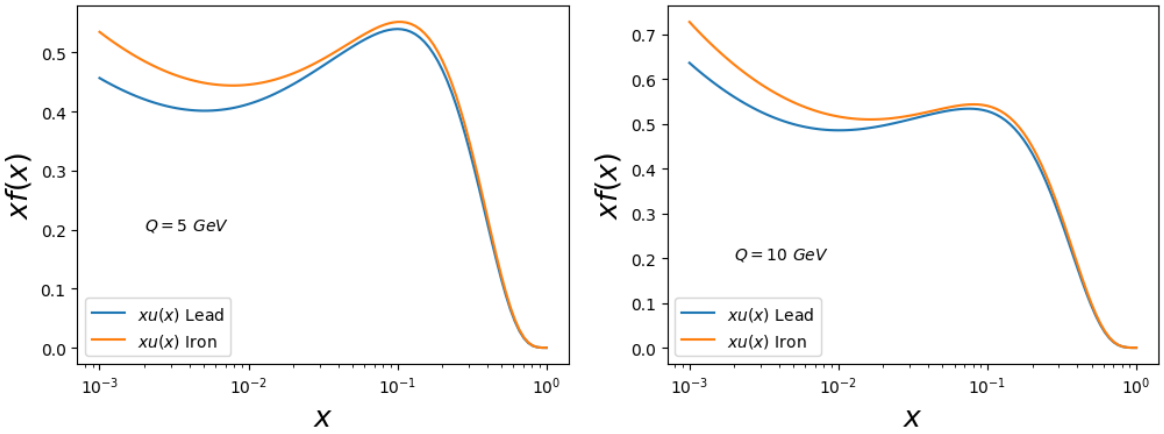
In [42]:
```
# Let's calculate the PDFs on each one of these points
fulllead_x_vals = { qq: [fullleads[0].xfxQ(2, xx, qq) for xx in xvalues]
fulliron_x_vals = { qq: [fullirons[0].xfxQ(flav, xx, qq) for xx in xvalue
```

In [43]:
```
%matplotlib inline
```

In [44]:
```
%pylab inline
```

```
%pylab is deprecated, use %matplotlib inline and import the required libra
ries.
Populating the interactive namespace from numpy and matplotlib
/usr/local/lib/python3.10/dist-packages/IPython/core/magics/pylab.py:166:
UserWarning: pylab import has clobbered these variables: ['degrees', 'gamm
a', 'frexp', 'radians', 'tanh', 'fmod', 'copysign', 'sinh', 'sin', 'lcm',
'isfinite', 'pow', 'log2', 'gcd', 'remainder', 'isnan', 'modf', 'e', 'ldex
p', 'isinf', 'isclose', 'nan', 'floor', 'tan', 'sqrt', 'trunc', 'ceil', 'e
xp', 'cos', 'prod', 'fabs', 'nextafter', 'pi', 'expm1', 'cosh', 'inf', 'lo
g10', 'log1p', 'log', 'hypot']
`%matplotlib` prevents importing * from pylab and numpy
  warn("pylab import has clobbered these variables: %s"  % clobbered +
```

In [45]:
```
# create a new figure
figure(1,figsize=(12,4))

# loop over Q values
for iq,qq in enumerate(Qval):
    # divide it into two subplot vertically
    subplot2grid((1,2),(0,iq))
    semilogx(xvalues, fulllead_x_vals[qq], label='$xu(x)\ \mathrm{{Lead}}
    semilogx(xvalues, fulliron_x_vals[qq], label='$xu(x)\ \mathrm{{Iron}}
    legend(loc=3)
    text(2e-3,0.2, "$Q={}\ GeV$".format(qq))
    xlabel('$x$',fontsize=18)
    ylabel('$xf(x)$',fontsize=18)
```



In [46]:
```
# Python program to demonstrate working
# of map.

# Return double of n
def addition(n):
    return n + n

# We double all numbers using map()
numbers = (1, 2, 3, 4)
result = map(addition, numbers)
print(list(result))
```

```
[2, 4, 6, 8]
```

In [47]:
```
def calcerrorpdf(PdfSet,iq,xx, Q, SymAsym='Asym',valence=False):
    """
    • Returns the Obervables with errors.
    • Pdfset must contain the central value in zeroth position
    • Pdfset then contains in odd position (1,3,5,...) the + error vector
    • PDFset contains in even  positions (2,4,...) the - error vectors.
    """

    def DFfuncp(x):
        return max([x[0]-Fs0,x[1]-Fs0,0])**2
    def DFfuncm(x):
        return max([Fs0-x[0],Fs0-x[1],0])**2
    def DFsym (x):
        return (x[1]-x[0])**2

    def Observable (pdfset):
        return pdfset.xfxQ(iq,xx,Q)

    def Observable_valence (pdfset):
        return pdfset.xfxQ(iq,xx,Q)-pdfset.xfxQ(-iq,xx,Q)



    if not(valence):
        #get the central value
        Fs0 = Observable(PdfSet[0])
        #Get the plus and minus sets
        Fsp = map(Observable,PdfSet[1::2])
        Fsm = map(Observable,PdfSet[2::2])
    else :
        Fs0 = Observable_valence(PdfSet[0])
        Fsp = map(Observable_valence,PdfSet[1::2])
```

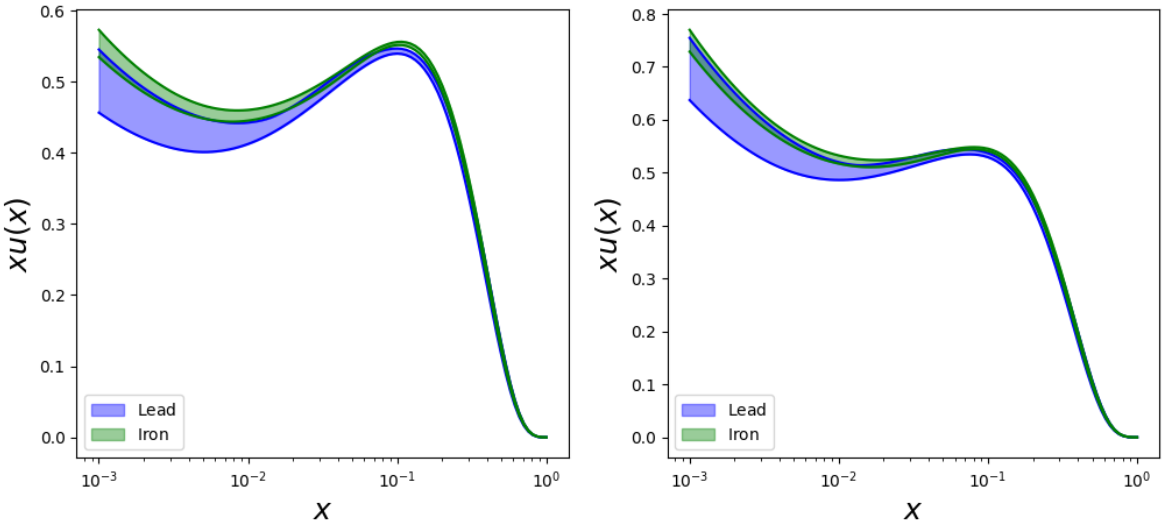
```
        Fsm = map(Observable_valence,PdfSet[2::2])

    #zip the two vectors returns (f1[0],f2[0]),(f1[1],f2[1]).... so usefu
    Fscombined = zip(Fsp,Fsm)

    if SymAsym == 'Asym':
        DFp,DFm = sqrt(sum(list(map(DFfuncp,Fscombined)))),sqrt(sum(list(
        return DFp,DFm,Fs0
    elif SymAsym == 'sym':
        DFp  = 0.5 * sqrt(sum(list(map(DFsym,Fscombined))))
        return DFp, DFp, Fs0
```

In [48]:
```
fullleads[0].xfxQ(1,0.1,10.)
```

Out[48]: 0.5809487232475684

In [49]:
```
# Let's add some errors
lead_error = {qq: np.array([calcerrorpdf(fullleads, flav, xx, qq) for xx
iron_error = {qq: np.array([calcerrorpdf(fullirons, flav, xx, qq) for xx
```

In [50]:
```
figure(2, figsize(12,5))
for iq, qq in enumerate(Qval):
    subplot2grid((1,2),(0,iq))
    #####
    #LEAD
    #####
    # plot central + Deltap
    semilogx(xvalues, fulllead_x_vals[qq]+lead_error[qq][:,0],color='b')
    #plot central - Delta m
    semilogx(xvalues, fulllead_x_vals[qq]-lead_error[qq][:,1],color='b')
    # fill in between
    fill_between(xvalues, fulllead_x_vals[qq] + lead_error[qq][:,0], full
    ######
    # IRON
    ######
    # plot central + Deltap
    semilogx(xvalues, fulliron_x_vals[qq]+iron_error[qq][:,0],color='g')
    #plot central - Delta m
    semilogx(xvalues, fulliron_x_vals[qq]-iron_error[qq][:,1],color='g')
    # fill in between
    fill_between(xvalues, fulliron_x_vals[qq] + iron_error[qq][:,0], full
    p1 = Rectangle((0,0),0,1,color='b',alpha=0.4)
    p2 = Rectangle((0,0),0,1,color='g',alpha=0.4)
    legend([p1,p2],['$\mathrm{Lead}$','$\mathrm{Iron}$'],loc=3)
    xlabel('$x$',fontsize=18)
    ylabel('$xu(x)$',fontsize=18)
```



In [ ]:

In [ ]:

Play with Integration

In [51]:
```
from scipy import integrate
```

In [52]:
```
# sFix=[xf0[0].xfxQ(istr,x0,qq) for qq in qvalues]
# sFree=[xf1[0].xfxQ(istr,x0,qq) for qq in qvalues]
```

In [53]:
```
xvalues = np.logspace(-3,0,100)
xvalues=xvalues[:-1]
print( len(xvalues) )
print( xvalues[0::10] )
```

```
99
[0.001      0.00200923 0.00403702 0.00811131 0.01629751 0.03274549
 0.06579332 0.13219411 0.26560878 0.53366992]
```

In [54]:
```
xf0[0].xfxQ(3,0.1,5)
```

Out[54]:
```
0.06396391472409065
```

In [55]:
```
istr=3
q0=1.4
sFix=[xf0[0].xfxQ(istr,xx,q0) for xx in xvalues]
dbFix=[xf0[0].xfxQ(-1,xx,q0) for xx in xvalues]
ubFix=[xf0[0].xfxQ(-2,xx,q0) for xx in xvalues]

sFree=[xf1[0].xfxQ(istr,xx,q0) for xx in xvalues]
dbFree=[xf1[0].xfxQ(-1,xx,q0) for xx in xvalues]
ubFree=[xf1[0].xfxQ(-2,xx,q0) for xx in xvalues]

print( len(sFix) , len(xvalues) )
print( sFix[0::20] )
```

```
99 99
[0.20579890141608911, 0.16608209702147544, 0.1262576859409809, 0.075113879
5872819, 0.013859900322095598]
```

In [56]:
```
print( len(sFix) , len(xvalues) )
```

```
99 99
```

In [57]:
```
sInt0=integrate.simpson(sFix, x=xvalues)
dbInt0=integrate.simpson(dbFix, x=xvalues)
ubInt0=integrate.simpson(ubFix, x=xvalues)
print(sInt0, dbInt0, ubInt0)
print(2.0*sInt0/(dbInt0+ubInt0))
```

```
0.015703466659388297 0.028751734603037038 0.03329210432038654
0.5062055131298371
```

In [58]:
```
sInt1=integrate.simpson(sFree, x=xvalues)
dbInt1=integrate.simpson(dbFree, x=xvalues)
ubInt1=integrate.simpson(ubFree, x=xvalues)
print(sInt1, dbInt1, ubInt1)
print(2.0*sInt1/(dbInt1+ubInt1))
```

```
0.027587341926872434 0.017188489825357298 0.03139653720696964
1.1356314326435053
```

In [ ]:
```
```