

Init

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

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

```
In [3]: import sys
sys.path.insert(0, '/home/vboxuser/LHAPDF/LHAPDF-6.5.3/wrappers/python/bui
```

```
In [4]: import lhpdf
print(lhpdf.version())

6.5.3
```

```
In [5]: lhpdf.pathsAppend('./PDFDIR/')
```

```
In [6]: lhpdf.__file__

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

```
In [7]: xf0 = lhpdf.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[7]: 0.66114855119381
```

```
In [8]: xf1 = lhpdf.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[8]: 0.6627243695017107
```

```
In [9]: xvalues = np.logspace(-3,0,100)[: -1]
qvalues = np.logspace(0,2,500)
```

Exercises

```
In [10]: #all the flavor numbers we care about and their labels
flavors_num = [2,1,3,4,6,5,21]
flavors_name = ['u','d','s','c','t','b','gluon/10']

#pick a Q value
q0 = 10
```

```

#loop over all flavors (and gluon), and all x values; if we have a gluon
sfix_flav = [[xf0[0].xfxQ(flavi,x,q0)/10 if flavi == 21 else xf0[0].xfxQ(
sfree_flav = [[xf1[0].xfxQ(flavi,x,q0)/10 if flavi == 21 else xf1[0].xfxQ(

fig = plt.figure()
ax = plt.subplot(111) #this is needed for the legend later

#plot each flavor
for flavi in flavors_num:
    ax.plot(xvalues,sfix_flav[flavors_num.index(flavi)],label="%s (rsFixe
    ax.plot(xvalues,sfree_flav[flavors_num.index(flavi)],label="%s (rsFre

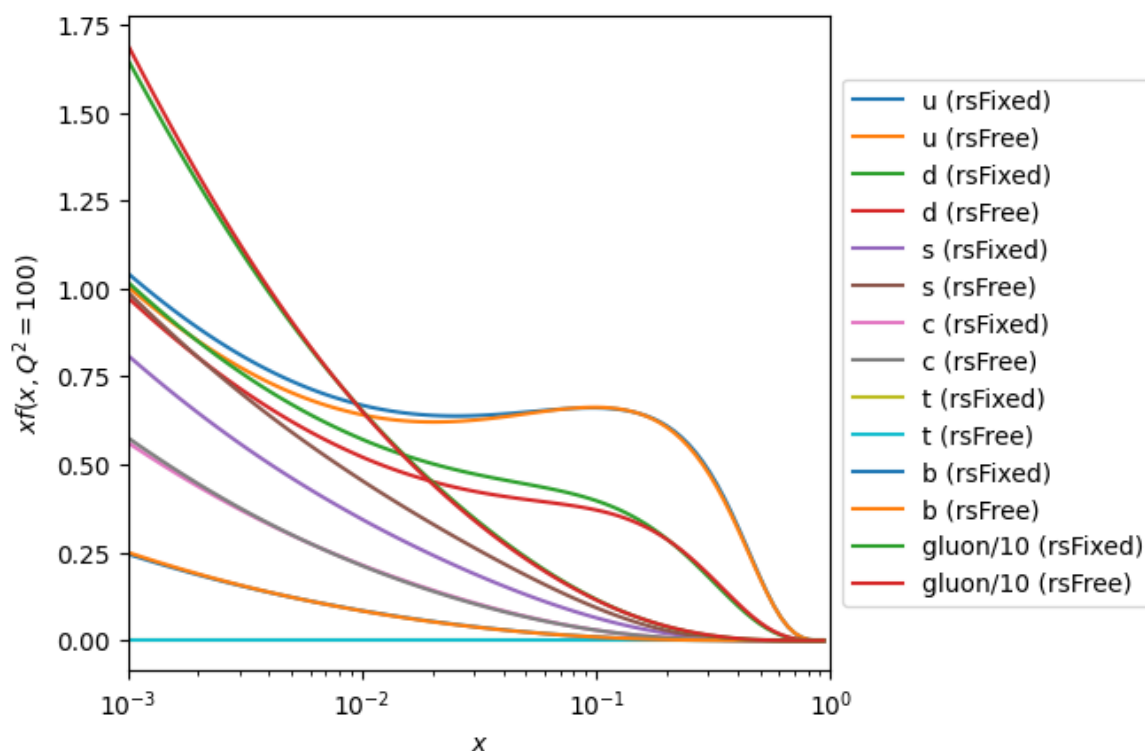
#rescale the plot and add a legend on the right
box = ax.get_position()
ax.set_position([box.x0, box.y0, 0.8*box.width, box.height])
ax.legend(loc="center left", bbox_to_anchor=(1,0.5))

plt.xscale("log")
plt.xlim((0.001,1))
plt.xlabel('$x$')
plt.ylabel('$xf(x,Q^2=%d)$'%q0**2)

fig = plt.gcf()

fig.savefig("step1.png",dpi=400)

```



```

In [11]: #same thing but now we fix a x0 and loop over all qvalues
flavors_num = [2,1,3,4,6,5,21]
flavors_name = ['u','d','s','c','t','b','gluon']

x0 = 0.05

#loop over all flavors and all q
sfix_flav = [[xf0[0].xfxQ(flavi,x0,q) for q in qvalues] for flavi in flav
sfree_flav = [[xf1[0].xfxQ(flavi,x0,q) for q in qvalues] for flavi in fla

```

```

#make the plot the same way as before
fig = plt.figure()
ax = plt.subplot(111)

for flavi in flavors_num:
    ax.plot(qvalues,sfix_flav[flavors_num.index(flavi)],label="%s (rsFixe
    ax.plot(qvalues,sfree_flav[flavors_num.index(flavi)],label="%s (rsFre

box = ax.get_position()
ax.set_position([box.x0, box.y0, 0.8*box.width, box.height])
ax.legend(loc="center left", bbox_to_anchor=(1,0.5))

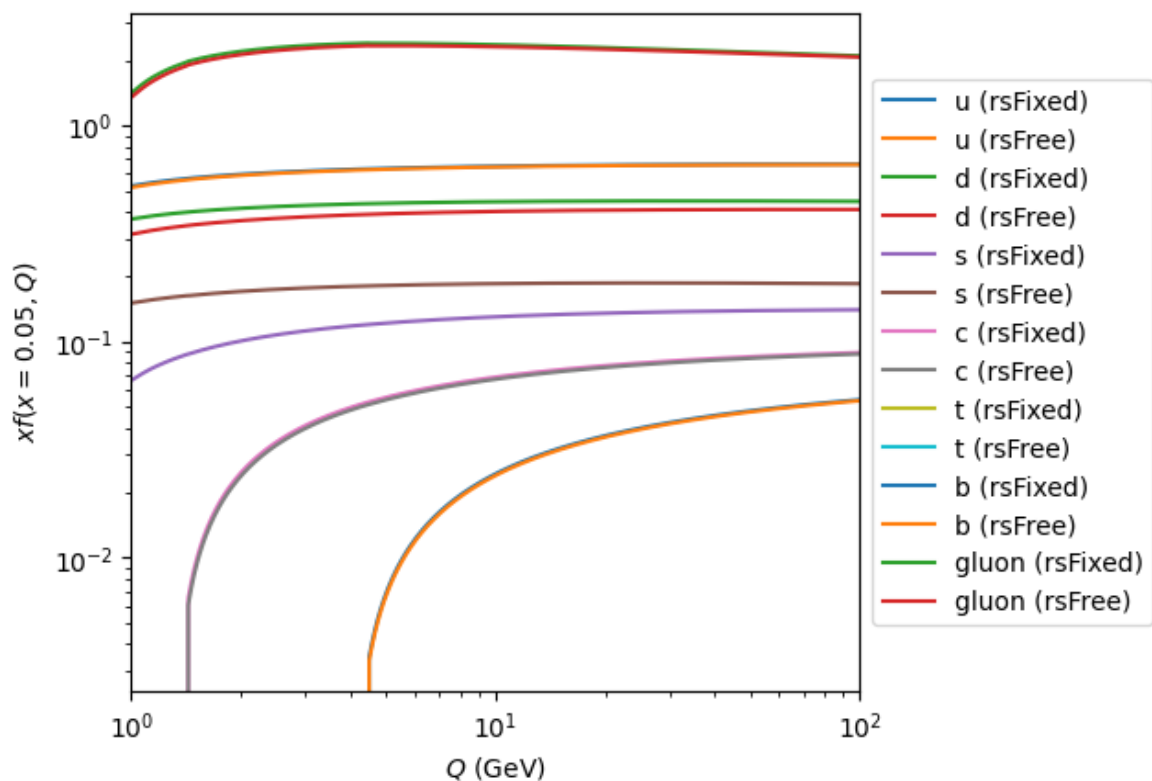
plt.xscale("log")
plt.xlim((1,100))
plt.xlabel('$Q$ (GeV)')

plt.yscale("log")
plt.ylabel('$xf(x=%0.2f,Q)$'%x0)

fig = plt.gcf()

fig.savefig("step2.png",dpi=400)

```



In [12]: `from scipy import integrate`

```

In [13]: #to account for ALL momentum fractions we also need to include the antiqu
flavors_num = [2,-2,1,-1,3,-3,4,-4,6,-6,5,-5,21]
flavors_name = ['u','ubar','d','dbar','s','sbar','c','cbar','t','tbar','b

#pick your favorite Q
q0 = 3

#get the data for all xvalues for each flavor
sfix_flav = [[xf0[0].xfxQ(flavi,x,q0) for x in xvalues] for flavi in flav

```

```

sfree_flav = [[xf1[0].xfxQ(flavi,x,q0) for x in xvalues] for flavi in fla

#integrate over x for each flavor
sfix_x_ints = [integrate.simpson(flav, x=xvalues) for flav in sfix_flav]
sfree_x_ints = [integrate.simpson(flav, x=xvalues) for flav in sfree_flav

#print the results
print("rsFixed:")
for i in range(len(flavors_name)):
    print(flavors_name[i].ljust(5), sfix_x_ints[i])
print("SUM ", sum(sfix_x_ints))

print()

print("rsFree:")
for i in range(len(flavors_name)):
    print(flavors_name[i].ljust(5), sfree_x_ints[i])
print("SUM ", sum(sfree_x_ints))

```

rsFixed:

```

u      0.31098274130018005
ubar   0.03423393915764831
d      0.14611104735896405
dbar   0.03042701454022303
s      0.019401748105255637
sbar   0.019466055348053825
c      0.006820348184701286
cbar   0.0068796466405982555
t      0.0
tbar   0.0
b      0.0
bbar   0.0
gluon  0.41783438752534985
SUM     0.9921569281609743

```

rsFree:

```

u      0.3073571183306357
ubar   0.03262684846536941
d      0.14252417724203995
dbar   0.020665417555012423
s      0.029384936532700603
sbar   0.029449925084195434
c      0.006801894300091693
cbar   0.006861822582834685
t      0.0
tbar   0.0
b      0.0
bbar   0.0
gluon  0.41707727253908455
SUM     0.9927494126319644

```

In [14]: *#do that again but for every single q in qvalues, and store the result in*

```

flavors_num = [2,-2,1,-1,3,-3,4,-4,6,-6,5,-5,21]
flavors_name = ['u','ubar','d','dbar','s','sbar','c','cbar','t','tbar','b'

#the results of the integrations, one array for each flavor
sfix_x_ints = [[] for _ in range(len(flavors_num))]
sfree_x_ints = [[] for _ in range(len(flavors_num))]

#for each q0...

```

```

for q0 in qvalues:
    #...do what we did in the previous example
    sfix_flav = [[xf0[0].xfxQ(flavi,x,q0) for x in xvalues] for flavi in
    sfree_flav = [[xf1[0].xfxQ(flavi,x,q0) for x in xvalues] for flavi in

    for flavi in range(len(flavors_num)):
        #put the results in the appropriate array bin
        #except now we also want a percentage so multiply everything by 1
        sfix_x_ints[flavi].append(100*integrate.simpson(sfix_flav[flavi],
        sfree_x_ints[flavi].append(100*integrate.simpson(sfree_flav[flavi]

#make the plot as usual
fig = plt.figure()
ax = plt.subplot(111)

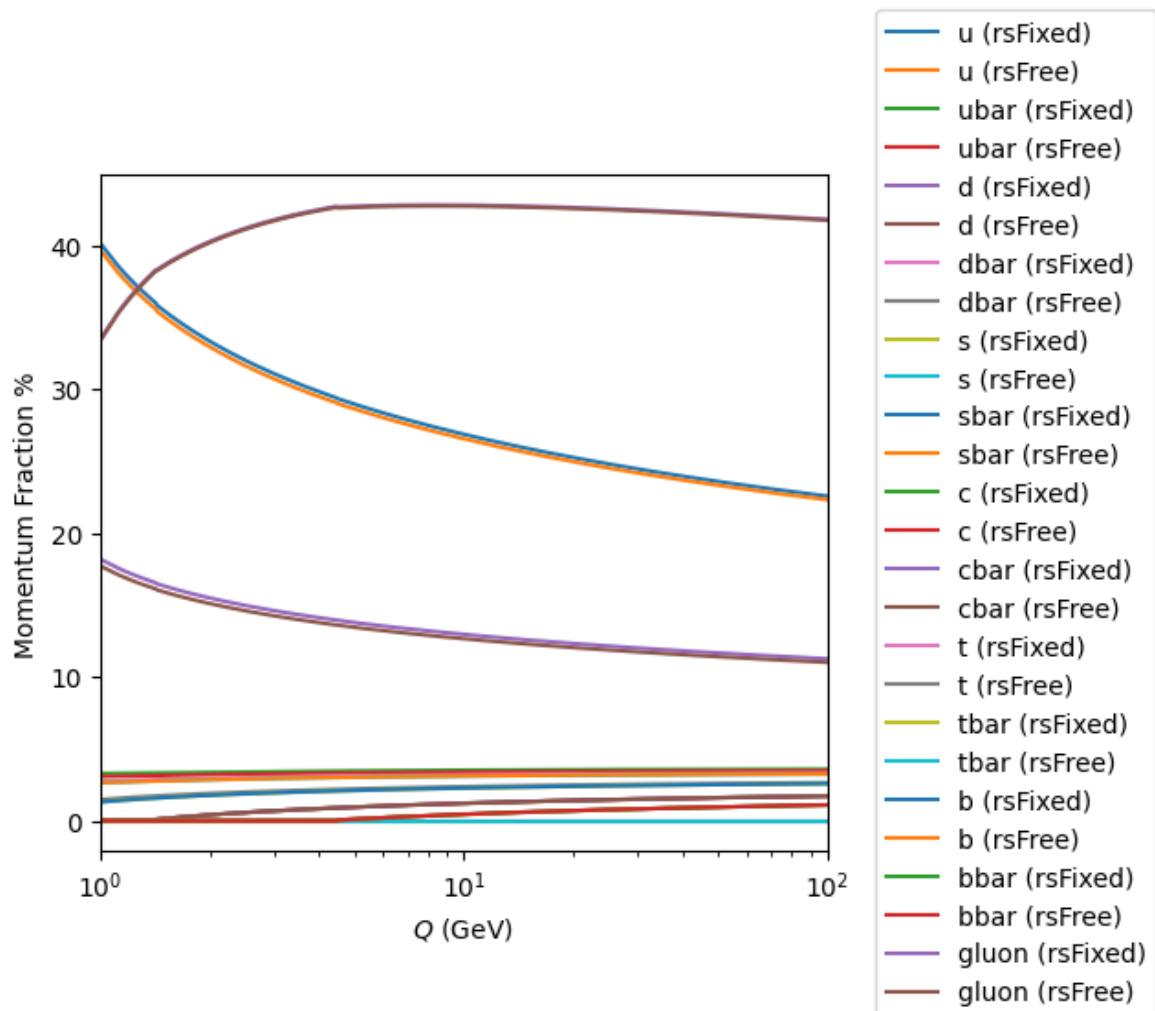
for flavi in flavors_num:
    ax.plot(qvalues,sfix_x_ints[flavors_num.index(flavi)],label="%s (rsFi
    ax.plot(qvalues,sfree_x_ints[flavors_num.index(flavi)],label="%s (rsF

box = ax.get_position()
ax.set_position([box.x0, box.y0, 0.8*box.width, box.height])
ax.legend(loc="center left", bbox_to_anchor=(1.05,0.5)) #legend has to mo

plt.xscale("log")
plt.xlim((1,100))
plt.xlabel('$Q$ (GeV)')

plt.ylabel('Momentum Fraction %')
fig.savefig("step4.png",dpi=400)

```



In []: