Parton Distribution Functions

Alexander Huss

July 22, 2023

Contents

1 Introduction

We will call the LHAPDF library through python to evaluate parton distribution functions and play around with them a bit.

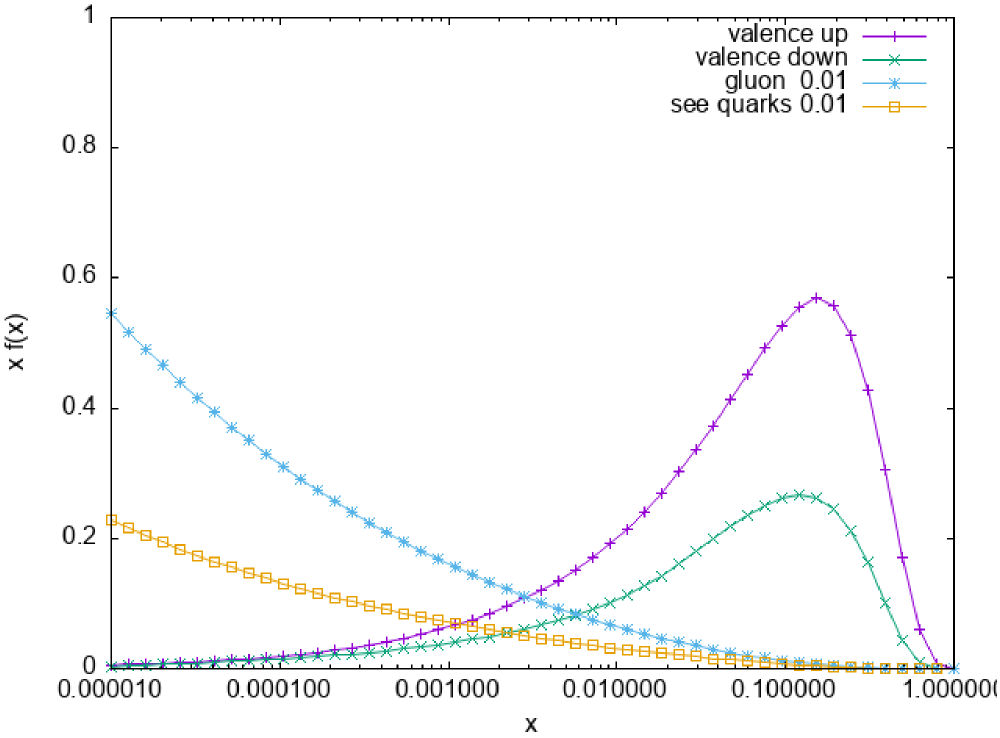
2 Evaluating PDFs

Let's have a look at one specific PDF set `PDF4LHC21_40`

```python
import lhapdf
import numpy as np

lhapdf.setVerbosity(0)
pdf = lhapdf.mkPDF("PDF4LHC21_40", 0)
if qlog:
    xs = [x for x in np.logspace(-5, 0, 50)]
else:
    xs = [x for x in np.linspace(0, 1, 50)]
res = np.empty([len(xs),6])
for ix, x in enumerate(xs):
    fac = 1. #1./x
    res[ix,0] = x
    res[ix,1] = q
    res[ix,2] = fac*(pdf.xfxQ(2, x, q) - pdf.xfxQ(-2, x, q))  # valence up-quark
    res[ix,3] = fac*(pdf.xfxQ(1, x, q) - pdf.xfxQ(-1, x, q))  # valence down-quark
    res[ix,4] = fac*(pdf.xfxQ(0, x, q))  # gluon (or `21`)
    res[ix,5] = fac*(
        2.*pdf.xfxQ(-2, x, q)
```

```
        +2.*pdf.xfxQ(-1, x, q)
        +pdf.xfxQ(3, x, q)+pdf.xfxQ(-3, x, q)
        +pdf.xfxQ(4, x, q)+pdf.xfxQ(-4, x, q)
        +pdf.xfxQ(5, x, q)+pdf.xfxQ(-5, x, q)
    )  # sum over all se quarks
return res
```



3 The quantum numbers of the proton

The qunatum numbers of the proton are determined by the valence quarks, p=(uud). Given that PDFs are *number densities* for the constituent partons, the following flavour sum rules hold:

$$\int_0^1 \mathrm{d}x \Big(f_{\mathrm{u}|\mathrm{p}}(x) - f_{\bar{\mathrm{u}}|\mathrm{p}}(x) \Big) = 2, \qquad \int_0^1 \mathrm{d}x \Big(f_{\mathrm{d}|\mathrm{p}}(x) - f_{\bar{\mathrm{d}}|\mathrm{p}}(x) \Big) = 1, \tag{1}$$

$$\int_0^1 \mathrm{d}x \Big(f_{\mathrm{q}|\mathrm{p}}(x) - f_{\bar{\mathrm{q}}|\mathrm{p}}(x) \Big) = 0 \quad \forall q \notin \{\mathrm{u},\, \mathrm{d}\} \tag{2}$$

Let's see if these hold for one of the global PDF sets:

```
import lhapdf
import math
import scipy
lhapdf.setVerbosity(0)
pdf = lhapdf.mkPDF("PDF4LHC21_40", 0)
```

```
res = list()
for id,label in [(1,"d"),(2,"u"),(3,"s"),(4,"c"),(5,"b")]:
    int_res = scipy.integrate.quad(lambda x : (pdf.xfxQ(id, x, q)-pdf.xfxQ(-id, x, q))/x, 0, 1, limit=1000, epsrel=1e-4)
    #print(int_res)
    res.append( (label, int_res[0]) )
return res
```

d	0.9895191932413574
u	1.9514579315015579
s	0.01509780263458491
c	7.592269630105822e-05
b	-0.0008936063440324684

4 Momentum sum rules

The parton a carries a momentum fraction x_a of the parent hadron, $p_a^\mu = x_a\, P^\mu$. Therefore, the momentum density associated witht that parton is given by $\$\mathrm{x}_{\mathrm{a\backslash,fa|\ H}}(\mathrm{x_a})$. Since the sum over all parton momenta must sum back up to the parent hadron one, the PDF sets satisfy a *momentum sum rule* (in $\overline{\mathrm{MS}}$):

$$\sum_a \int_0^1 \mathrm{d}x_a\; x_a\, f_{a|H}(x_a) = 1 \tag{3}$$

Let's see how the momenta are distributed accross different flavours

```
import lhapdf
import math
import scipy
lhapdf.setVerbosity(0)
pdf = lhapdf.mkPDF("PDF4LHC21_40", 0)
res = list()
int_sum = [0.,0.]
for id,label in [(-5,"bb"),(-4,"cb"),(-3,"sb"),(-2,"ub"),(-1,"db"),(0,"g"),(1,"d"),(2,"u"),(3,"s"),(4,"c"),(5,"b")]:
    int_res = scipy.integrate.quad(lambda x : pdf.xfxQ(id, x, q), 0, 1, limit=100)
    int_sum[0] += int_res[0]
    int_sum[1] += int_res[1]**2
    #print(int_res)
    res.append( (label, "{:.0f}%".format(int_res[0]*100.)) )
res.append( ("SUM", "{:.0f}%".format(int_sum[0]*100.)) )
return res
```

bb	1%
cb	2%
sb	3%
ub	4%
db	4%
g	47%
d	11%
u	22%
s	3%
c	2%
b	1%
SUM	100%

So the gluon actually carries almost 50% of the proton's momentum! The up quark, with $\sim$ 20%, has the second largect contribution, followed by the down-quark \$~\$half the size of the up (which makes sense as p=(uud)).