Parton Showers

Alexander Huss

July 30, 2023

Contents

1 Introduction

We will investigate the emission probability of gluons off quarks and gluons and use that to implement a **very** simplified parton shower (only final state, primary branching, leading double-log, only virtuality q^2 and not proper kinematics, . . .).

2 Emission probability and the Sudakov form factor

In the leading double-log approximation (soft *and* collinear emission), we have seen in the lecture that the emission probability is given as

$$\mathrm{d}\omega_{X\to X+g} = 2\,\frac{\alpha_s}{\pi}\,C_X\,\frac{\mathrm{d}E}{E}\,\frac{\mathrm{d}\theta}{\theta}\,, \tag{1}$$

where E denotes the energy of the emitted gluon and θ the angle w.r.t. the parent particle. We denote the emitting particle by "X" and C_X is the associated colour factor. For quarks, $C_X = C_F = \frac{4}{3}$ and for gluons $C_X = C_A = 3$.

For any parton shower, we first need to fix the evolution variable w.r.t. which we want to generate emissions. To this end, we choose the virtuality q^2 associated with the emission for which we find

$$d\mathcal{P} = \frac{\alpha_s}{\pi}\,C_X\,\frac{\mathrm{d}q^2}{q^2}\,\ln\!\left(\frac{q^2}{Q_0^2}\right) \xrightarrow{\int\mathrm{d}q^2} \frac{\alpha_s C_X}{2\pi}\,\ln^2\!\left(\frac{q^2}{Q_0^2}\right) \tag{2}$$

where Q_0 denotes a cutoff below which emissions are considered unresolved. Note that this fixed-order result comes with a serious problem: For one α_s, we get two powers of a potentially large logarithm (the so-called "double logarithms" that appear frequently in higher-order calculations), a pattern that will continue to higher orders. For some representative values ($\alpha_s \sim 0.1$, $Q_0 \sim \Lambda_{\rm QCD} \sim 0.2\,{\rm GeV}$, $q \sim 100\,{\rm GeV}$), we quickly realize that the large logarithm compensates the small value of the coupling, giving rise to a non-converging expansion. In such situations, where we are sensitive to large logarithms, we need to re-arrange the perturbative expansion in such a way to "re-sum" these large logarithms to all orders.

To accomplish this, we define the so-called Sudakov form factor $\Delta(Q^2, q^2)$, which is the probability for *no resolved emissions* to happen between the evolution $Q^2 \to q^2$. It satisfies a differential equation reminiscent of radiative decay with a simple solution

$$\begin{aligned} \frac{{\rm d}\Delta(Q^2,q^2)}{{\rm d}q^2} &= \Delta(Q^2,q^2)\,\frac{{\rm d}\mathcal{P}}{{\rm d}q^2}\,, \\ \Delta(Q^2) \equiv \Delta(Q^2,Q_0^2) &= \exp\left\{-\frac{\alpha_s C_X}{2\pi}\,\ln^2\left(\frac{q^2}{Q_0^2}\right)\right\}\,, \end{aligned} \tag{3}$$

which now has the large logarithm in the exponent. This solution therefore accomplishes exactly what we wanted: sum up the problematic logarithms to all orders, and in doing so, tame the otherwise divergent behaviour ($Q_0 \to 0$). It turns out that we can use the Sudakov form factor to sample successive emissions (it's a Markovian process), which we discuss in the next section.

3 Implementation

With the Sudakov form factor at hand, we can easily iterate the sampling of emissions using the following steps:

1. set $Q = Q_{\rm start}$
2. draw a uniform random number r in the range [0, 1]
3. if $r < \Delta(Q^2)$, no resolvable emission can be generated ($< Q_0$): Terminate loop.
4. solve $r = \Delta(Q^2)/\Delta(Q_{\rm new}^2)$ for $Q_{\rm new}$, which is the new emission scale.
5. "generate" the emission at $Q_{\rm new}$, set $Q = Q_{\rm new}$ and go back to step 2.

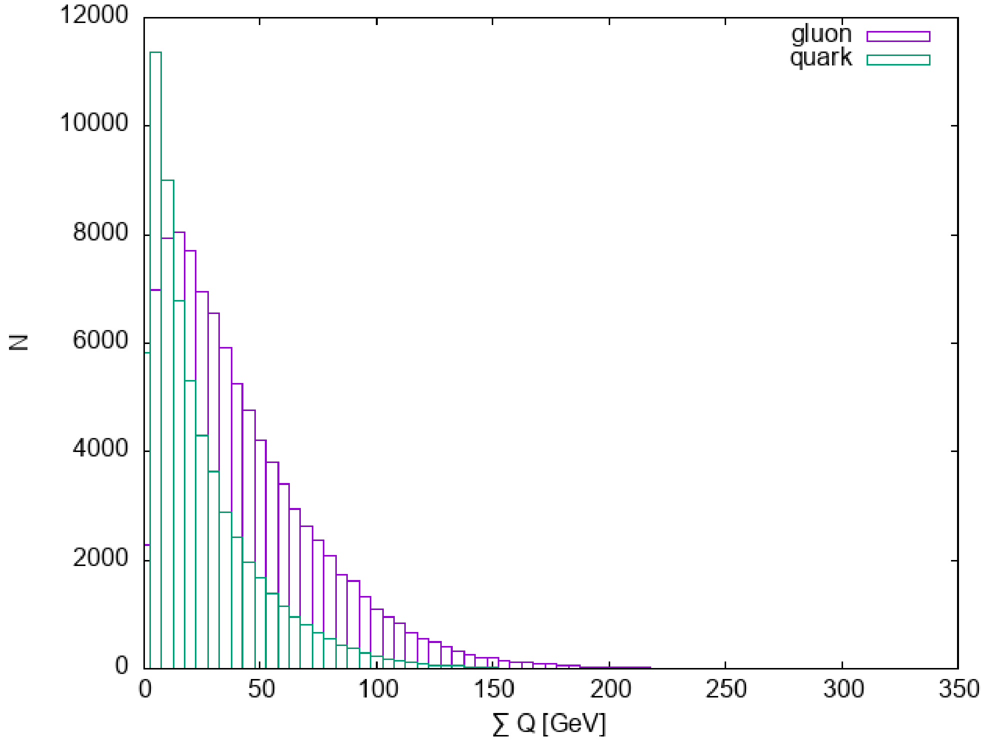
```python
#!/usr/bin/env python

import math
import random
import sys

random.seed(42)
alphas = 0.118

def generate_event(Q2_start: float, Q2_cutoff: float, CX: float):
  sudakov = 1.  # initialize Sudakov to the starting scale
```

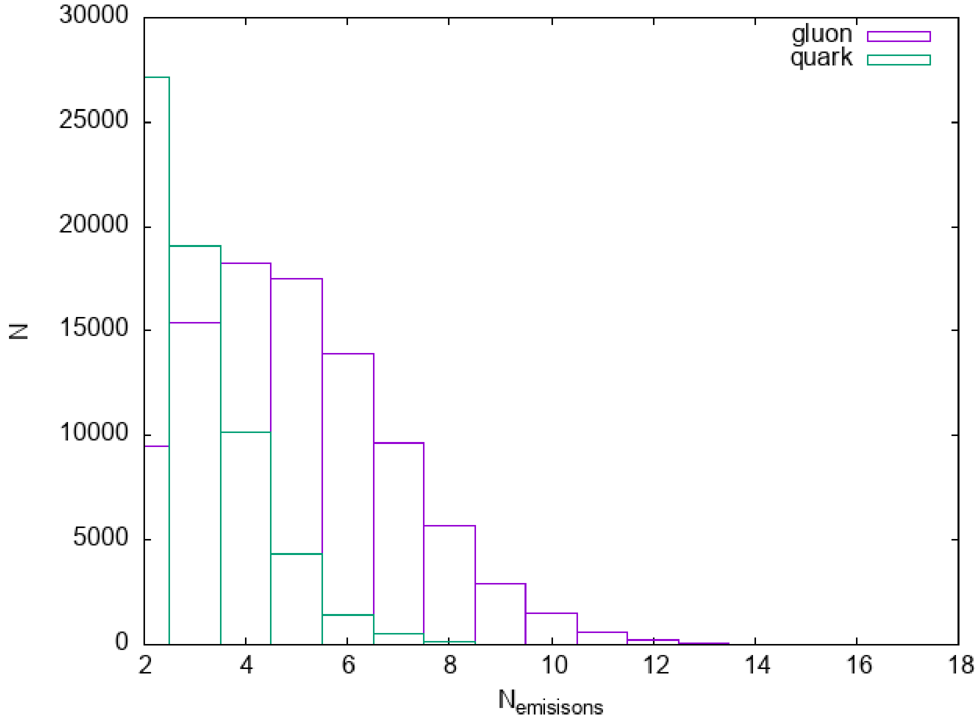
```
  fac = alphas*CX/(2.*math.pi)
  Qlist = []
  while True:
    r = random.uniform(0.,1.)
    sudakov *= r
    #> sudakov = exp( -[alphas*CX/(2.*pi)] * log^2[Q2/Q2_start] )
    #> determine Q2 from the associated sudakov
    L2 = - math.log(sudakov) / fac
    Q2 = Q2_start * math.exp(-math.sqrt(L2))
    if Q2 < Q2_cutoff:
      break
    Qlist.append( math.sqrt(Q2) )
  if len(Qlist) > 1:
    print("#summary2 {} {} {} {}".format(len(Qlist),sum(Qlist),Qlist[0],Qlist))

if __name__ == "__main__":
  if len(sys.argv) < 3:
    raise RuntimeError("I expect at least two arguments:  Q_start [g|q]")
  Q_start = float(sys.argv[1])  # the hard scale
  Q_cutoff = 1  # shower cutoff (PS stops -> hand over to hadronization)
  if sys.argv[2] == "q":
    CX = 4./3.  # quark
  elif sys.argv[2] == "g":
    CX = 3.     # gluon
  else:
    raise RuntimeError("unrecognised parton: {}".format(sys.argv[2]))
  if len(sys.argv) >= 4:
    alphas = float(sys.argv[3])
  if len(sys.argv) >= 5:
    nevents = int(sys.argv[4])
  else:
    nevents = 1000
  for i in range(nevents):
    print("# event {} [{} {} {} {} {}]".format(i,Q_start,sys.argv[2],CX,alphas,nevents))
    generate_event(Q_start**2, Q_cutoff**2, CX)
```

Let's use the implementation to generate some "events"

```
python main.py 100 g 0.118 100000 > data_g.dat
python main.py 100 q 0.118 100000 > data_q.dat
```

We can see that the all-order description damps the divergent behaviour of a pure fixed-order prediction for $Q \to 0$. Given $C_A > C_F$, we also see how a gluon generates more emissions than quarks. This property can be exploited to try and discriminate between "quark jets" and "gluon jets".





- To increase the amount of emissions, try out setting the strong coupling to $\alpha_s = 0.5$. How does the picture change?