Transverse Momentum Resummation

Alexander Huss

August 29, 2023

Contents

1 Introduction

In the lectures we have seen a brief overview of the q_T resummation formalism for the Drell-Yan process. We will have a closer look at the main results here and highlight some features.

2 q_T resummation

In the leading double-logarithmic approximation, we have found the result

$$\frac{1}{\sigma_0}\frac{\mathrm{d}\sigma}{\mathrm{d}q_T^2} = \int_0^\infty \mathrm{d}b\,\frac{b}{2}\,J_0(q_T b)\,\exp\Big[-\frac{\alpha_s}{2\pi}\,C_F\,\ln^2(Q^2b^2)\Big]\,, \tag{1}$$

where we have completely ignored effects from subleading logarithms, the running of the strong coupling, and parton distributions functions. Nontheless, this simple formula already allows us to inspect some important features of q_T resummation.

3 Implementation

3.1 Python

The integral is a bit nasty because of the oscillating behaviour of the Bessel funtion J_0 so we need to adapt the `scipy.integrate` settings a little bit. Despite that, the implementation is straighforward:

```python
#!/usr/bin/env python
import sys
from math import pi, exp, log, log10, ceil, floor
from scipy.special import jv  # Bessel function of the 1st kind
from scipy.integrate import quad
import numpy as np

alpha_s: float = 0.118


def res_integrand(b: float, QT: float, Q: float, CX: float) -> float:
    # b0: float = 2. * exp(-0.57721566490153286061)
    # blim: float = 5.  # should be > 1/Lambda_QCD ~ 5
    # bs2: float = b**2 * blim**2 / (b**2 + blim**2)
    # return (b / 2.) * jv(0, b * QT) * exp(
    #     -alpha_s / (2. * pi) * CX * log(Q**2 * bs2 / b0**2 + 1.)**2)
    return (b / 2.) * jv(0, b * QT) * exp(
        -alpha_s / (2. * pi) * CX * log(Q**2 * b**2)**2)


if __name__ == "__main__":
    if len(sys.argv) < 3:
        raise RuntimeError("I expect at least two arguments:  Q [g|q]")
    Q = float(sys.argv[1])  # the hard scale
    pow_low = -4
    pow_upp = floor(log10(Q)) # ceil(log10(Q/2.))
    if sys.argv[2].lower() == "q":
        CX = 4. / 3.
    elif sys.argv[2].lower() == "g":
        CX = 3.
    else:
        raise RuntimeError("unrecognised parton: {}".format(sys.argv[2]))

    if len(sys.argv) >= 4:
        alpha_s = float(sys.argv[3])

    if len(sys.argv) >= 5:
        nsteps = int(sys.argv[4])
    else:
        nsteps = 51

    # print("# qt dSigQT2_val dSigQT2_err")
    for qt in np.logspace(pow_low, pow_upp, nsteps):
        val, err = quad(res_integrand,
                        0.,
                        np.inf,
                        args=(qt, Q, CX),
                        epsabs=0.,
                        epsrel=1e-3,
                        limit=50000)
        print("{}  {} {}".format(qt, val, err))
```

And we can generate some data files for Drell-Yan and Higgs production

```
python main.py  91 q > data_dy.dat
python main.py 125 g > data_h.dat
```

3.2 Mathematica

To cross-check the numerics, we can use a simple Mathematica implementation

```
dSigQT2[qt_] := Module[{cf = 4/3, as = 0.118, q = 91},
  NIntegrate[b/2 BesselJ[0, b*qt] Exp[-(as/(2 Pi)) cf Log[q^2 b^2]^2], {b, 0, Infinity}]
]
datQT2 = Table[{qt, dSigQT2[qt]}, {qt, 10^Range[-4, 2, 0.1]}]
Export["mma_dy.dat", datQT2, "Table", "FieldSeparators" -> " "]
```

4 Playground

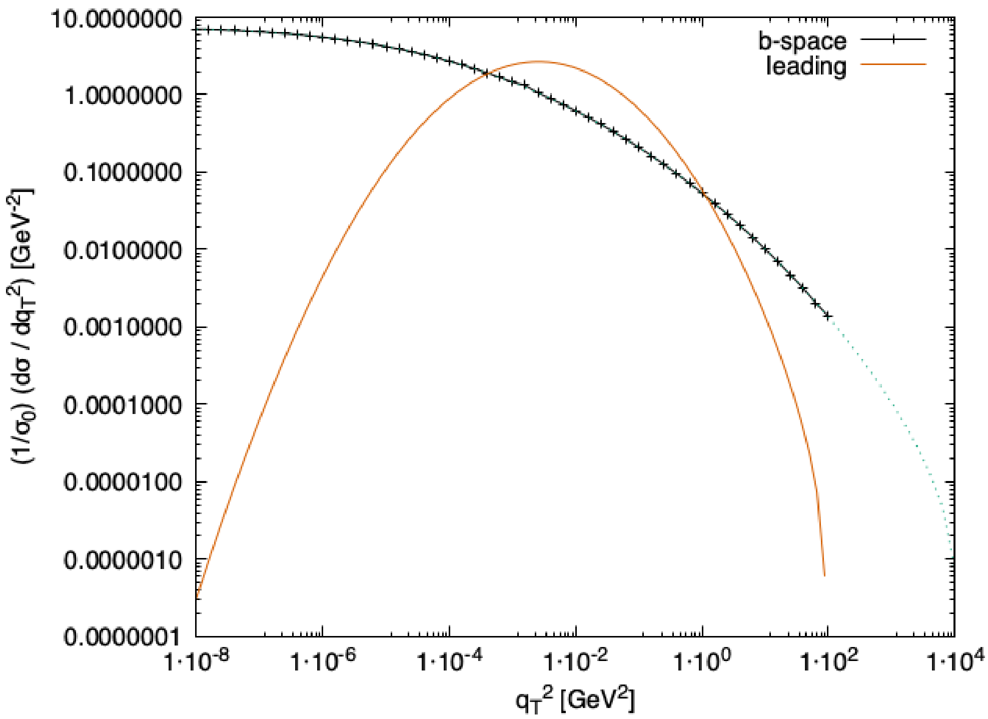
4.1 Low-q_T^2 behaviour

Let us have a look at the analytic expression for $\mathrm{d}\sigma/\mathrm{d}q_T^2$ in the leading double-logarithmic approximation in momentum space.

$$\frac{1}{\sigma_0}\frac{\mathrm{d}\sigma}{\mathrm{d}q_T^2} = \frac{\alpha_s}{\pi}\,C_F\,\frac{\ln(Q^2 q_T^2)}{q_T^2}\exp\left[-\frac{\alpha_s}{2\pi}\,C_F\,\ln^2(Q^2 q_T^2)\right]. \tag{2}$$

This expression can be obtained from Eq. (1) by systematically expanding the Fourier transform or alternatively by naively resumming the emissions *without* the transverse momentum conservation constraint.

We can compare this expression with the numerically evaluated b-space formula from above:



What is important to notice here is that the leading expression shows a strikingly different behaviour in the small q_T^2 limit compared to the b-space formula. The physical interpretation is quite clear: the leading term corresponds to restricting *all* gluon emissions to have k_T below the gauge-boson transverse momentum q_T. This gives a suppression at low q_T that is stronger than any power and as a consequence, the sub-leading effect suddenly becomes the leading one. In this situation, the small-q_T region is not restricted to only soft gluon emissions but instead by multiple gluon emissions that can individually have $k_T > q_T$ but they *balance out* in the azimuthal plane. By formulating the resummation in impact parameter space, this feature is automatically incorporated in the prediction.

This non-vanishing intercept in $\mathrm{d}\sigma/\mathrm{d}q_T^2$ for $q_T \to 0$ is a very important feature of trensverse momentum resummation. In fact, we can compute what this intercept is

$$\frac{1}{\sigma_0}\frac{\mathrm{d}\sigma}{\mathrm{d}q_T^2}\bigg|_{q_T=0} = \frac{\pi}{2Q^2}\,\frac{\mathrm{e}^{\overline{\frac{\pi}{2\alpha_s C_F}}}}{\sqrt{2\alpha_s C_F}}\,. \tag{3}$$

We have also superimposed a dotted line obtained from the Mathematica implementation, which is in good agreement so numerics appear to be under good control. Note that in my experimentations, Vegas appeared to struggle quite a bit with the oscillating behaviour of the integrand.

4.2 Transverse Momentum Distributioins

We now look at the transverse momentum distrubution, given by

$$\mathrm{d}\sigma/\mathrm{d}q_T = 2q_T\ \mathrm{d}\sigma/\mathrm{d}q_T^2\,, \tag{4}$$

and which does vanish in the $q_T \to 0$ limit like a power.