Hacking CCFM

Gavin Salam
CERN — TH Division
& LPTHE, Jussieu, Paris
Some of the proposals put together with Hannes Jung

Lund Small-x Workshop, 2 March 2001

hacker, *noun* [Computer User High-Tech Dictionary]:

1. [. . .] A hacker can mean an expert programmer who finds special tricks for getting around obstacles and stretching the limits of a system.

2. To some people it means an unconventional programmer or one who is not formally trained, or one who jerry-rigs programs (making temporary fixes that are not well-done).

[Oxford English Dictionary, citing 1983 Byte May 298/1]: 'Hacker' seems to have originated at MIT. The original German/Yiddish expression referred to someone so inept as to make furniture with an axe, but somehow the meaning has been twisted so that it now generally connotes ...

Status of CCFM

CCFM embodies the *combination* of ordering in x and ordering in angles.

It was intended to give dominant contributions at large Q^2 and small x. It correctly reproduces:

✓ Leading single logs of x for cross section.

✓ Leading single logs of x for final state (double logs of x usually cancel).

✓ Leading double logs of Q^2 for final state.

These make it a *good starting point for an Event Generator.*

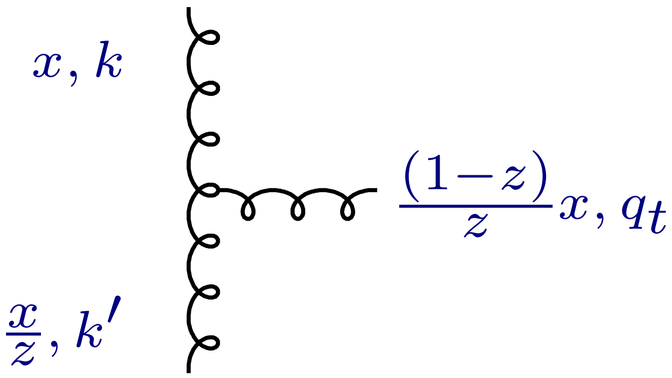
Various elements are missing. They are all formally subleading, but NLL BFKL suggests that in practice the following may be important:

✗ Running coupling.

✗ Full DGLAP splitting function.

✗ Correct anti-collinear behaviour (or symmetry).

The CCFM equation



Define also:

- ✗ $q = \dfrac{q_t}{1-z}$, rescaled transverse momentum
- ✗ p, which specifies maximum angle for branching

CCFM branching probability is:

$$\bar{\alpha}_s \frac{\Delta_S(p)}{\Delta_S(zq)} \left[\frac{1}{z}\Delta_{NS}(z,q,k) + \frac{1}{1-z}\right] \Theta(p - zq)$$

where the Sudakov form factor is

$$\ln \Delta_S(p) = -\int^{p^2} \frac{dq^2}{q^2} \int_0^1 \frac{dz}{1-z}\,\bar{\alpha}_s$$

and the non-Sudakov form factor is

$$\ln \Delta_{NS}(z,q,k) = -\int \frac{dq'^2}{q'^2} \int_z^1 \frac{dz}{z} \bar{\alpha}_s \Theta(q' - z'q)\Theta(k - q')$$

Importance of different pieces

$$\bar{\alpha}_{\mathsf{s}} \frac{\Delta_{\mathrm{S}}(p)}{\Delta_{\mathrm{S}}(zq)} \left[\frac{1}{z}\Delta_{\mathrm{NS}}(z,q,k) + \frac{1}{1-z}\right] \Theta(p - zq)$$

For a correct description (LLx) of the cross section, only the LH piece in the square brackets gives a leading contribution.

For the final state each piece, individually, gives rise to contributions $\alpha_{\mathsf{s}} \ln^2 x$. But such contributions *cancel* in the sum of the two pieces leaving at most contributions such as $\alpha_{\mathsf{s}} \ln x \ln Q$.

Running coupling

Hacks are constrained: they must give correct logarithmic behaviours in various limits.

NLLx corrections tell us that at small z correct scale for coupling is $q \simeq q_t$ (α_s Bremsstrahlung scheme). For z close to 1, we know from large-x resummations that we must use $\alpha_s(q_t)$. So simplest solution is to use $q_t = (1-z)q$ everywhere. This gives

$$\bar{\alpha}_s[(1-z)q] \, \frac{\Delta_S(p)}{\Delta_S(zq)} \left[\frac{1}{z}\Delta_{NS}(z,q,k) + \frac{1}{1-z}\right] \Theta(p - zq)$$

We must also modify the form factors to include the analogous running of the coupling (at least in regions where there are log or double-log divergences):

$$\ln \Delta_S(p) = -\int^{p^2} \frac{dq^2}{q^2} \int_0^1 \frac{dz}{1-z} \, \bar{\alpha}_s[(1-z)q]$$

and

$$\ln \Delta_{NS}(z,q,k) = -\int \frac{dq'^2}{q'^2} \int_z^1 \frac{dz}{z} \bar{\alpha}_s(q')\Theta(q' - z'q)\Theta(k - q')$$

(For z close to 1 there are no divergences here so we cannot say which of q or q_t is more appropriate).

Full DGLAP splitting function

Full DGLAP splitting function is

$$P_{gg}(z) = \frac{1}{z} + \frac{1}{1-z} - 2 + z(1-z)$$

So it is tempting to replace $1/(1-z)$ part of CCFM splitting function with

$$\frac{1}{1-z} \to \frac{1}{1-z} - 2 + z(1-z)$$

But this particular solution can give *negative branching probabilities*. One workaround is to modify also the small-z piece of the branching probability and use

$$\left[\left(\frac{1-z}{z} + \frac{z(1-z)}{2}\right)\Delta_{\mathrm{NS}}(z,q,k) + \left(\frac{z}{1-z} + \frac{z(1-z)}{2}\right)\right]$$

One has much freedom ($\rightarrow$ ambiguity) in distributing splitting function between the small and large-z terms.

Another requirement is that Sudakov form factor be modified:

$$\ln \Delta_{\mathrm{S}}(p) = -\int^{p^2} \frac{dq^2}{q^2} \int_0^1 dz \bar{\alpha}_{\mathrm{s}}[(1-z)q] \left(\frac{z}{1-z} + \frac{z(1-z)}{2}\right)$$

Symmetry, a.k.a. Kinematic Constraint

γ^*-proton scattering is identical to proton-γ^* scattering. So evolution up in Q^2 from proton to γ^* must give the same answer as going down in Q^2 from γ^* to proton.

Furthermore, during evolution, transverse scale goes up and down even if external scales (proton / γ^*) are identical — so the evolution must be done correctly in both directions.

Why is this non-trivial? When going up in scale, natural variable is Sudakov-plus component (x) — going down in scale, natural variable is Sudakov-minus component ($\bar{x}$). CCFM already involves three variables: x, angle and transverse momentum. Full treatment of an extra evolution variable would complicate things by an order of magnitude.

What happens if we ignore the problem? We get spurious collinear double logs. Illustrate by showing what happens in DIS if $\bar{x} = Q_0^2/s$ is used 'naively' instead of $x = Q^2/s$.

An evolution in $\bar{x}$ gives double logs

$$\left(\bar{\alpha}_s \ln \frac{Q^2}{Q_0^2} \ln \frac{1}{\bar{x}}\right)^n$$

Rewrite this result in terms of $\ln 1/\bar{x} = \ln 1/x + \ln Q^2/Q_0^2$:

$$\left(\bar{\alpha}_s \ln \frac{Q^2}{Q_0^2} \ln \frac{1}{x} + \bar{\alpha}_s \ln^2 \frac{Q^2}{Q_0^2}\right)^n$$

Illegal double collinear logs appear!

Double anti-collinear logs

For collinear evolution CCFM uses x: ✓
For anti-collinear evolution internally it uses $\sqrt{x\bar{x}}$: ✗

This means that double anti-collinear logs will be wrong. To see by how much we can compare to NLL corrections. In $\chi_1(\gamma)$ there is a piece

$$-\frac{1}{(1-\gamma)^3}\,.$$

This is a 'counterterm' which sorts out the anticollinear double logs when using the x as the main evolution variable.

CCFM implicitly contains NLL corrections. The $\gamma \to 1$ of χ_1^{ccfm} is

$$-\frac{1}{2}\frac{1}{(1-\gamma)^3}\,.$$

The factor of $1/2$ arises because $\sqrt{x\bar{x}}$ is internal evolution variable, rather than $\bar{x}$.

Kinematic constraint

Original CCFM, and then LDC, and Durham-Cracow BFKL include a ‘kinematic’ constraint:

$$\Theta\left(\frac{k^2}{k'^2} - z\right)$$

$i.e.$ in anticollinear region, $k^2 \ll k'^2$, effective evolution variable is $zk'^2/k^2 \simeq \bar{z}$. In LDC and BFKL this leads to correct anti-collinear double logs.

But straightforward insertion of this constraint into CCFM branching probability gives wrong answer for $\gamma \to 1$:

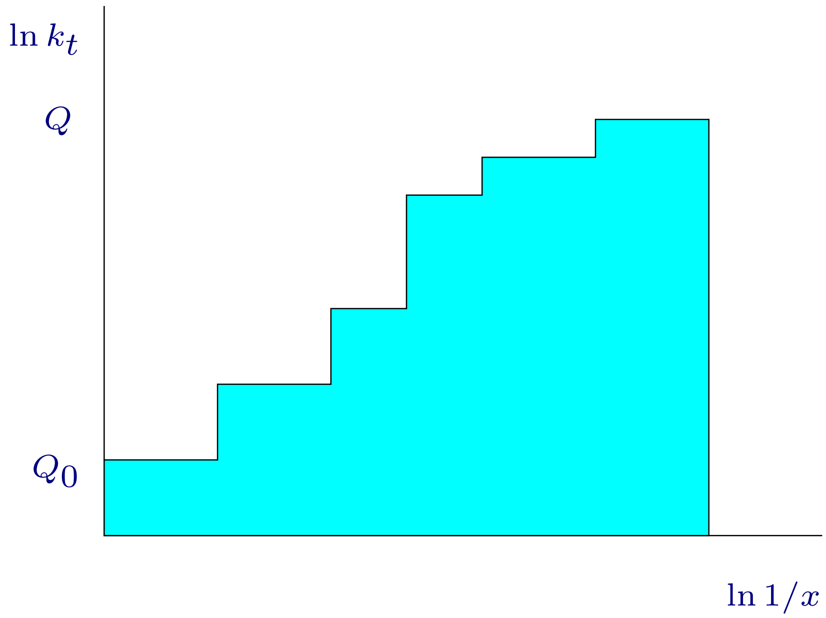
$$-\frac{3}{2}\frac{1}{(1-\gamma)^3}\,.$$

Reason is that constraint needs to to be taken into account in virtual corrections: at edge of kinematic constraint, $z = k^2/k'^2$ one has that the normal $\Delta_{\mathrm{NS}} \ll 1$, whereas we should have $\Delta_{\mathrm{NS}} = 1$, since the kinematic constraint tells us that there is 100% probability of there having been no branching. A solution is to change Δ_{NS} to

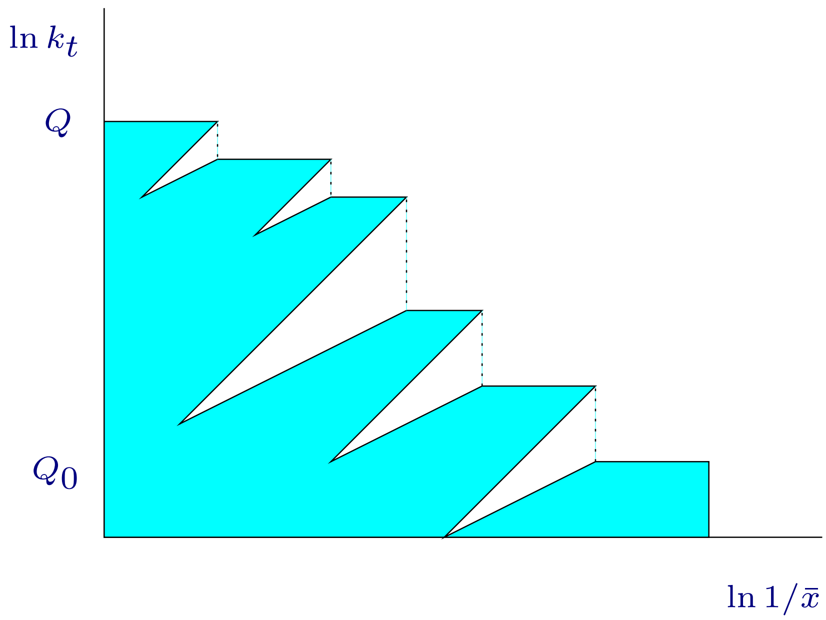
$$\ln \Delta_{\mathrm{NS}}(z,q,k) =$$
$$-\int \frac{dq'^2}{q'^2}\int_z^1 \frac{dz}{z}\bar{\alpha}_{\mathsf{s}}(q')\Theta(q'-z'q)\Theta(k-q')\Theta\left(\frac{k^2}{q^2}-z'\right)$$

Anticollinear properties of final state

For collinear evolution distribution of emissions typically looks as follows



For anti-collinear evolution, shown with $\bar{x}$ rather than x the correct answer should be symmetric to this. But CCFM with kinematic constraint actually gives



Important cross checks: NLL

The NLL corrections for small-x evolution of the cross section are known. We should be able to use these as a constraint on the evolution equation.

	$O\left(\bar{\alpha}_s^2\right)$
Full BFKL	-18.4
default CCFM (fixed coupling)	-9.2
CCFM with $\Theta(k-(1-z')q')$ in Δ_{NS}	~ -75

It is important to check the NLL corrections of any given eqn. because seemingly small modifications can change things completely.

Current methods for determining the NLL corrections of any given version of CCFM are clumsy, and sometimes specific to the given version of the equation. We need a simpler method: maybe some Monte Carlo integration method can provide a simple generalisable way of getting the desired information.

Summary

Status of different approaches to small-x evolution. Maybe discussion can help get rid of some question marks?

	BFKL	CCFM	LDC
X-section LLx	✓	✓	almost
X-section LLQ^2	B?	(✓)	(✓)?
final-state LLx	✓	✓	almost?
final-state DLQ^2	✗	✓	✓
sym. X-section DLQ^2	(✓)	(✓)	✓
sym. X-section SLQ^2	B?	✗?	(✓)?
sym. final-state DLQ^2	✗	✗	✓
all-orders IRC safe	✗	✓	✓

BFKL = exclusive formulation of BFKL

almost = not exact, but numerically similar
(✓) = not by default, but can be arranged
B = possible, but may break something else?
? = Don't know — you tell me!

- ✗ So far, hacking CCFM can solve some but not all problems.
- ✗ Non-logarithmic parts of hacks are *arbitrary* — we must cross-check with NLL corrections to ensure that we're not doing anything too stupid.
- ✗ Is CCFM a sufficiently flexible starting point?