TeVJet: a general framework for the calculation of jet observables in NLO QCD

Mike Seymour on behalf of Chris Tevlin

University of Manchester

4th HERA and the LHC workshop
Wednesday 28 May 2008

Overview

- Introduction - the Dipole Subtraction Method
- TeVJet Design
- New calculations - including new processes
- PhaseSpace class - integrable singularities
- Example results
- Summary & Outlook

The Dipole Subtraction Method

The problem is that different terms must be integrated over different phase space volumes. The general idea is to write the NLO contribution as:

$$\begin{aligned}\sigma^{\mathrm{NLO}} &= \int_{\mathrm{m+1}} d\sigma^{\mathrm{R}} + \int_{\mathrm{m}} d\sigma^{\mathrm{V}} + \int_{\mathrm{m}} d\sigma^{\mathrm{C}} \\ &= \int_{\mathrm{m+1}} \left[d\sigma^{\mathrm{R}} - d\sigma^{\mathrm{A}}\right] + \int_{\mathrm{m+1}} d\sigma^{\mathrm{A}} + \int_{\mathrm{m}} d\sigma^{\mathrm{V}} + \int_{\mathrm{m}} d\sigma^{\mathrm{C}}\end{aligned}$$

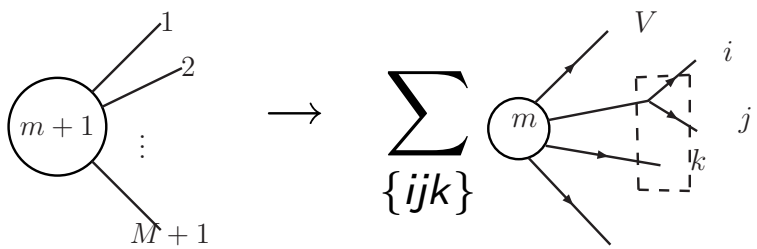
Where the 'subtraction term' can be easily integrated analytically and combined with the virtual and collinear terms:

$$\begin{aligned}\sigma^{\mathrm{NLO}} &= \int_{\mathrm{m+1}} \left[d\sigma^{\mathrm{R}}(p) - d\sigma^{\mathrm{A}}(p)\right] + \int_{\mathrm{m}} \left[d\sigma^{\mathrm{V}}(p) + d\sigma^{B}(p) \times \mathbf{I}\right] \\ &\quad + \int_0^1 dx \int_{\mathrm{m}} d\sigma^{\mathrm{B}}(xp) \times (\mathbf{P} + \mathbf{K})(x)\end{aligned}$$

The subtraction term

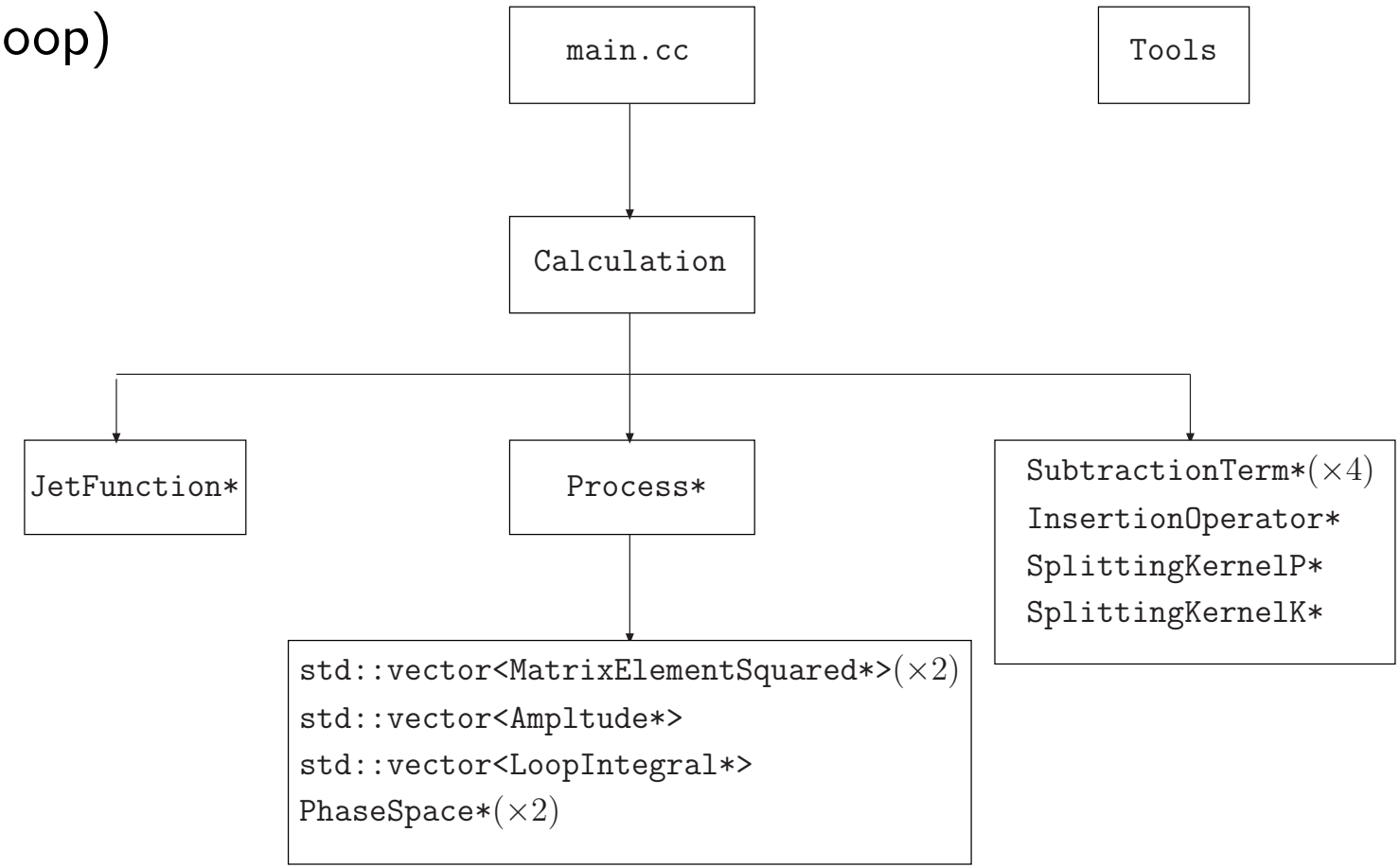
Need to construct a subtraction term that:

- Matches the divergences of the Real $|\mathcal{M}|^2$.
- Smoothly extrapolates away from these singular regions such that it obeys a factorisation over the whole of phase space.
- Must be simple enough to be analytically integrable (in d dimensions) over the one parton subspace containing the divergence [Phase Space constraints].



TeVJet Design - structure of the code

Aim: automate construction of all process-independent parts
(user still has to construct process-dependent matrix elements:
– Born, including colour and spin projections
– NLO tree
– NLO loop)



TeVJet Design - example main program


```
//standard lib headers
#include <iostream>
#include <cmath>

//my lib headers that are always needed
#include "Tools.h"
#include "Calculation.h"
#include "JetFunction.h"
#include "Process.h"

//process specific user headers
#include "ProcEEto3jet.h"
#include "UserEEto3jet.h"

int main(int argc, char** argv) {

  long           seed = 12345;
  std::string filename="EEto3jet_12345";

  //set the SM parameters
  TeVJet::Tools::getInstance()->setEWconvention(3);
  TeVJet::Tools::getInstance()->setWmass(80.41);
  TeVJet::Tools::getInstance()->setZmass(91.188);
  TeVJet::Tools::getInstance()->setGfermi(1.16639*pow(10,-5));
  TeVJet::Tools::getInstance()->setAlphasMZ(0.118);

  //set some parameters
  int       Nborn = 1000;//no of 'events'
  int       Nvirt = 1000;
  int       Nreal = 1000;

  double CMenergy = TeVJet::Tools::getInstance()->getZmass();//GeV
  bool       useZ = false;

  double Ebeam1   = CMenergy/2;
  double Ebeam2   = CMenergy/2;

  double fren = 1.0;

  TeVJet::Process     * proc;
  TeVJet::JetFunction * jet;
```



```
  //process specific stuff
  proc = new TeVJet::ProcEEto3jet(CMenergy,useZ);
  jet  = new TeVJet::UserEEto3jet(filename,CMenergy,useZ);
  //process specific ends

  TeVJet::Calculation calc(proc,jet);
  //user can set stuff here
  calc.setNborn(Nborn);
  calc.setNreal(Nreal);
  calc.setNvirt(Nvirt);
  calc.setSeed(seed);
  calc.setEbeam1(Ebeam1);
  calc.setEbeam2(Ebeam2);

  calc.setRenScaleFactor(fren);

  //initialize stuff
  calc.Initialize();

  //calculate born level
  calc.BornLoop();

  //if required calculate NLO contribution
  if(calc.m_proc->DoNLO()){
    calc.RealLoop();
    calc.VirtLoop();
    if(calc.m_proc->HadronBeams()){
      calc.ConvLoop();
    }
  }

  delete jet;
  jet=0;
  delete proc;
  proc=0;

  return 0;
}
```

How to include a new process

- Need to write a class which inherits from the Process class
- This includes the (usual) process-dependent information:
  - Square of the matrix elements for the Born level and real diagrams:

$$|\mathcal{M}_{m;a,\ldots}|^2 =_{m;a,\ldots} \langle 1,\ldots,m;a,\ldots|1,\ldots,m;a,\ldots\rangle_{m;a,\ldots}. \quad (1)$$

  - The Born level Amplitude projected onto the helicity space of each external gluon
  - The necessary colour algebra to calculate the colour correlated Born Level amplitude for each combination of emitter and spectator:

$$|\mathcal{M}^{I,J}_{m;a,\ldots}|^2 =_{m;a,\ldots} \langle 1,\ldots,m;a,\ldots|\mathbf{T}_I.\mathbf{T}_J|1,\ldots,m;a,\ldots\rangle_{m;a,\ldots}, \quad (2)$$

  - The 1-loop renormalized Amplitude
- Also need to know the scales (μ_R, μ_F) etc.

Matrix Element Squared

The user must provide code to return $|\mathcal{M}|^2$ for each sub-process which contributes to the Born level and real terms.

- TeVJet links to the HELAS library which can be used to calculate these
- This could be fully automated (c.f. MADGRAPH) in future

Born level Amplitude

Recall that in the soft and collinear limits the factorization is not exact - there are colour and spin correlations.

- In order to match the soft limits TeVJet requires user to provide the necessary colour algebra to calculate the colour-correlated amplitude for all emitters/spectators:

$$|\mathcal{M}^{I,J}_{m;a,\ldots}|^2 = {}_{m;a,\ldots}\langle 1,\ldots,m;a,\ldots|\mathbf{T}_I.\mathbf{T}_J|1,\ldots,m;a,\ldots\rangle_{m;a,\ldots}, \tag{3}$$

- For the collinear limits TeVJet needs the Born level amplitude projected onto the spin space of each external gluon, i.e $\mathcal{A}^\mu$ where

$$\mathcal{A} = \epsilon^*_\mu \mathcal{A}^\mu \tag{4}$$

- This could also be fully automated in future.

1-loop Amplitude

- Using dimensional regularization the renormalized one-loop matrix element in the $\overline{\mathrm{MS}}$ scheme is typically of the form

$$|\mathcal{M}|^2_{1-loop} = \frac{\alpha_S}{2\pi}\frac{1}{\Gamma(1-\epsilon)}\left(\frac{4\pi\mu^2}{Q^2}\right)^{\epsilon}\left\{\frac{A}{\epsilon^2}+\frac{B}{\epsilon}+C+\mathcal{O}(\epsilon)\right\}, \tag{5}$$

  where μ is the dimensional regularization scale (which we set equal to the renormalization scale throughout), and Q is some scale relevant to the process..
- The convention in `TeVJet` is to extract a factor of

$$\frac{1}{\Gamma(1-\epsilon)}\left(\frac{4\pi\mu^2}{Q^2}\right)^{\epsilon}, \tag{6}$$

  from the insertion operator and the one-loop matrix element.
- The user must provide the reference scale Q and the finite part of the 1-loop amplitude with this pre-factor extracted.
- In the future this could be the only process-dependent input required for a new process.

Phase space generation

The phase space generation in TeVJet uses the dipole phase space factorization:

$$d\Pi_{m+1}(\{p\}; Q) = d\Pi_m(\{\tilde{p}\}; Q) \cdot \mathcal{J}\, dydzd\phi$$

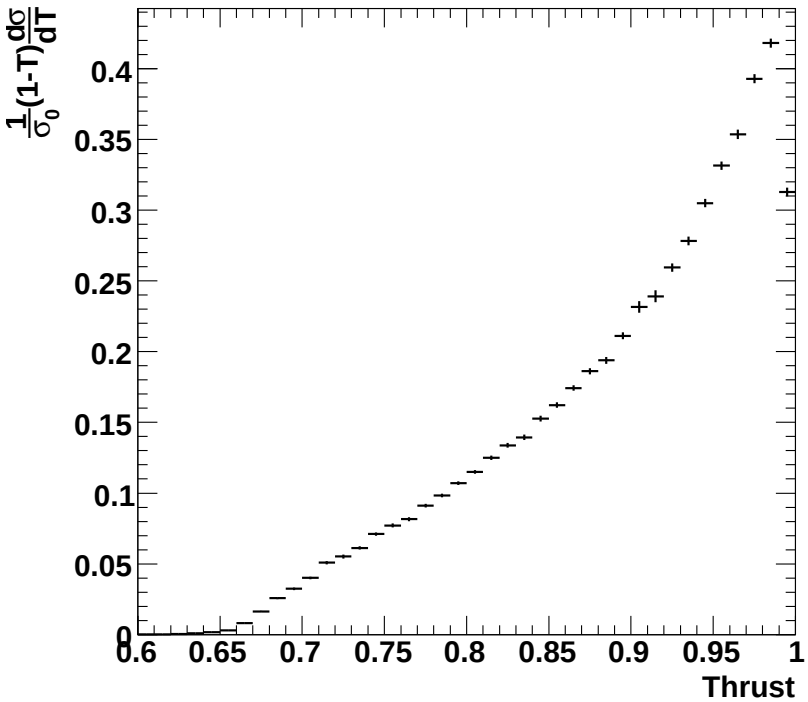
to generate an $m+1$ body configuration from an m body configuration.

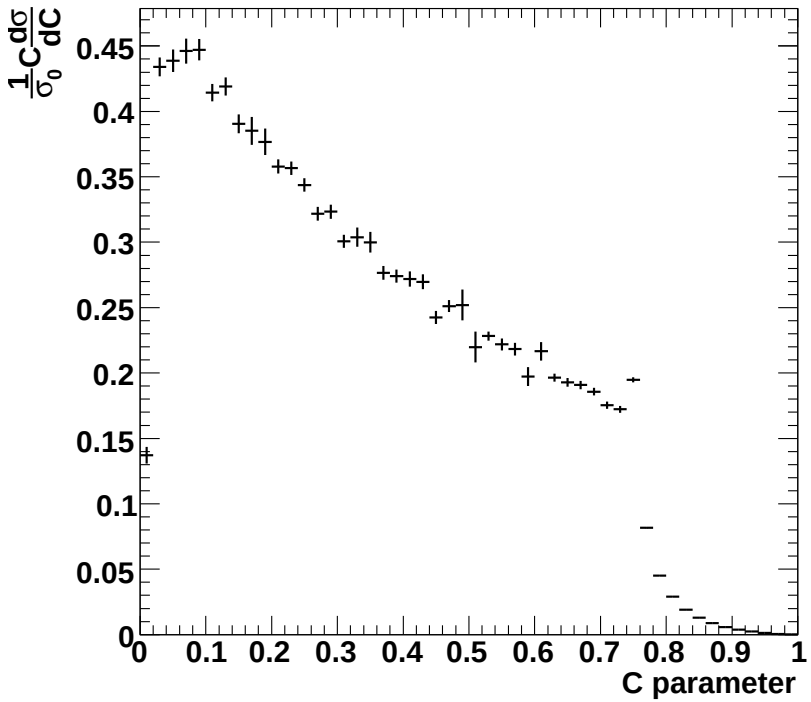
- The 'real minus subtraction term' typically contains (integrable) square-root singularities in the dipole variables as $y \longrightarrow 0$ and $z \longrightarrow 0, 1$
- TeVJet uses a multichannel phase space generation introducing a Jacobian factor of the form

$$\mathcal{J} = \frac{8\sqrt{y_{ij,k}}\sqrt{\widetilde{z}_i(1-\widetilde{z}_i)}}{\sqrt{\widetilde{z}_i} + \sqrt{1-\widetilde{z}_i}}, \tag{7}$$

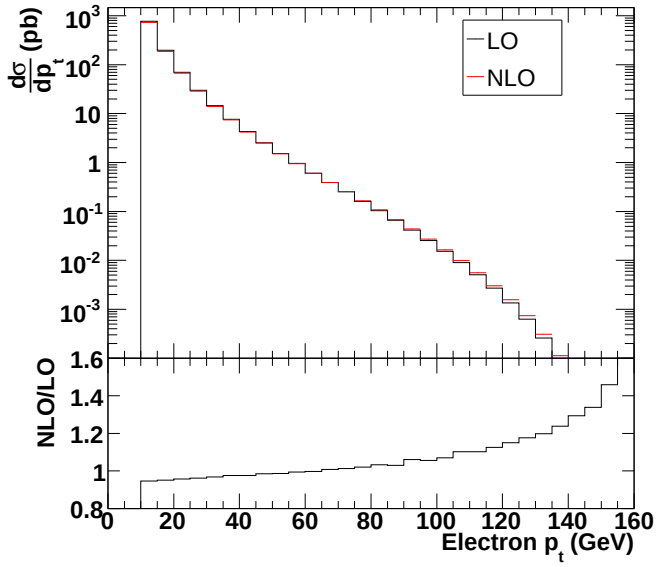
for each channel.

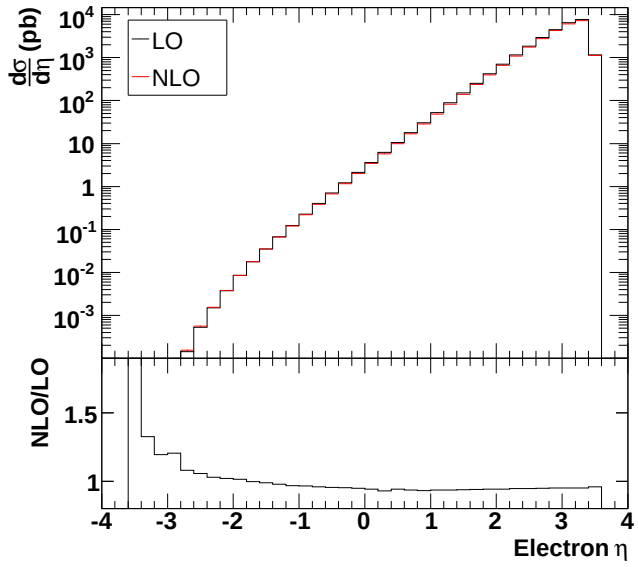
TeVJet Results - 3jet event shapes

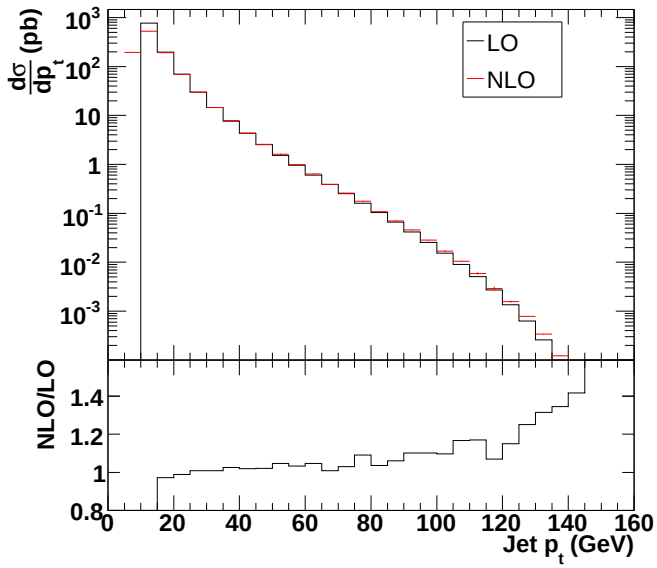


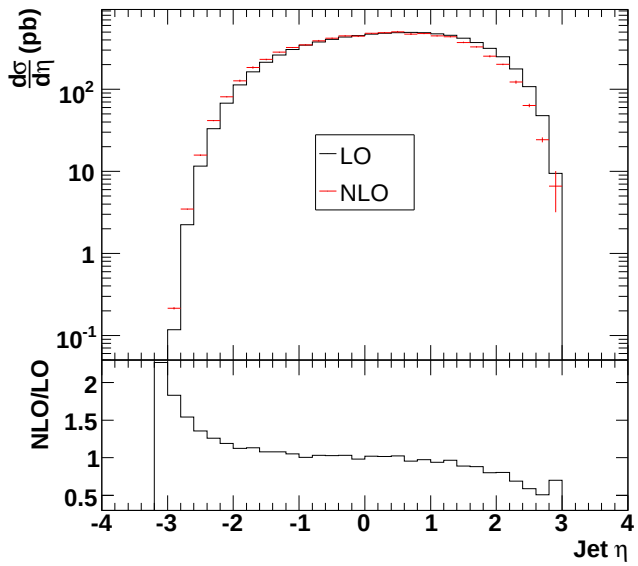


TeVJet Results - $ep \longrightarrow e + 1\text{jet}$









Summary & Outlook

- TeVJet is a direct implementation of the dipole subtraction method
- Inclusion of new processes fairly straight forward
- Could be made easier by full automation of Matrix element and Amplitude code (colour algebra)
- Full working version for the case of massless particles
- Currently including all mass effects ($pp \longrightarrow t\bar{t}$ cross section in progress)