QCD at hadron colliders
Lecture 4: some main tools at LHC

Gavin Salam

CERN, Princeton & LPTHE/CNRS (Paris)

Maria Laach Herbtschule für Hochenenergiephysik
September 2010, Germany

If you work directly on LHC/Tevatron physics, what QCD tools will you run into?

1. Monte Carlo shower programs
2. Fixed order codes
3. Procedures to “merge” their predictions
4. Jet algorithms

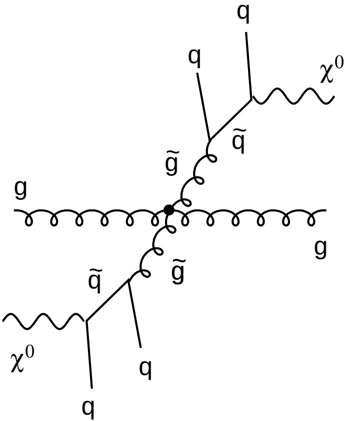
If you work directly on LHC/Tevatron physics, what QCD tools will you run into?

1. Monte Carlo shower programs
2. Fixed order codes
3. Procedures to "merge" their predictions
4. Jet algorithms

An example process

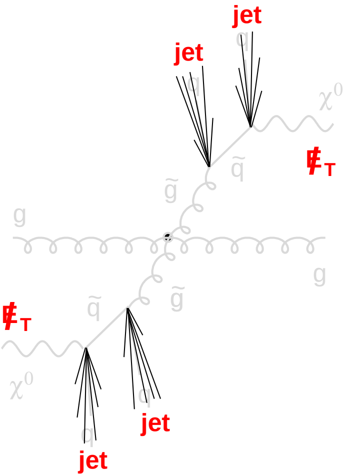
SUSY example: gluino pair production

Signal



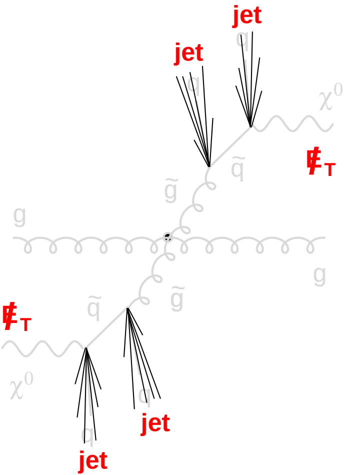
SUSY example: gluino pair production

Signal

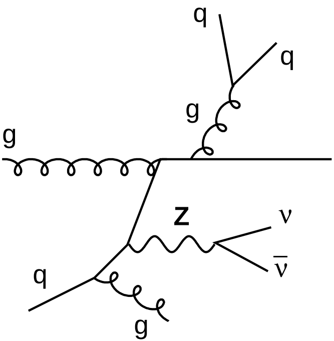


SUSY example: gluino pair production

Signal

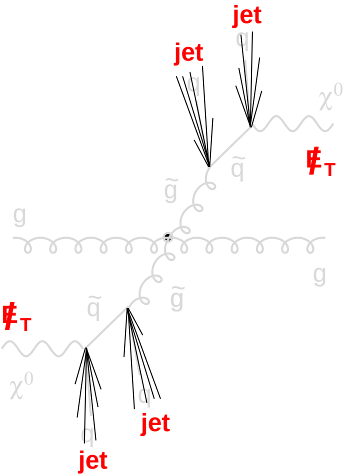


Background

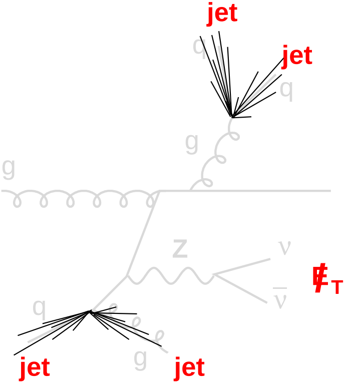


SUSY example: gluino pair production

Signal



Background

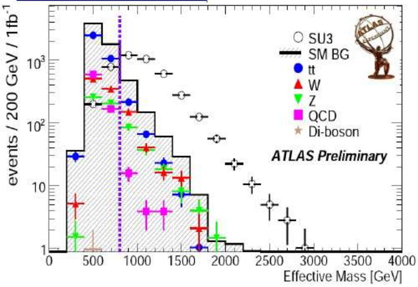


Example SUSY searches

Atlas selection [all hadronic]

- no lepton
- MET > 100 GeV
- 1^{st},2^{nd} jet > 100 GeV
- 3^{rd},4^{th} jet > 50 GeV
- MET / m_{eff} > 20%

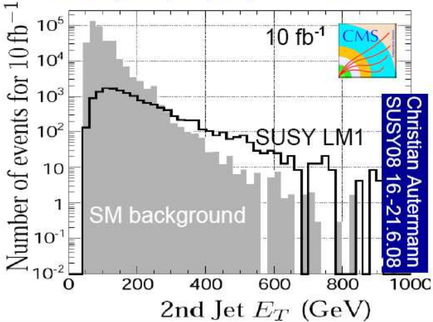
Christian Autermann
SUSY08 16.-21.6.08
4



CMS selection [leptonic incl.]

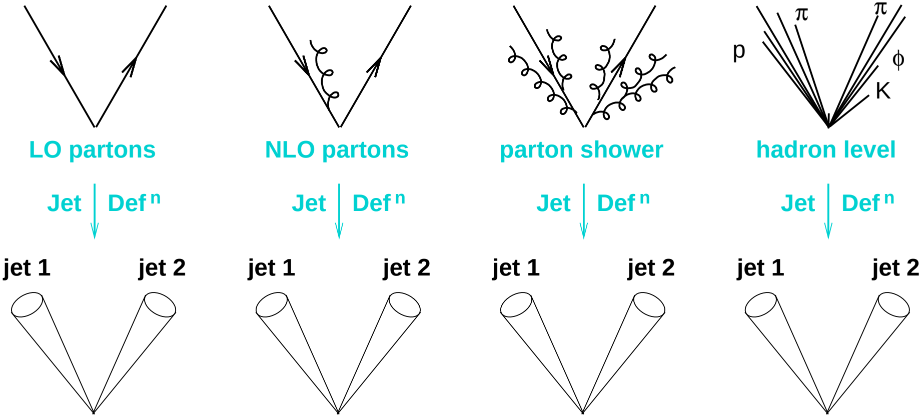
(optimized for 10fb^{-1}, using genetic algorithm)

- 1 muon pT > 30 GeV
- MET > 130 GeV
- 1^{st}, 2^{nd} jet > 440 GeV
- 3^{rd} jet > 50 GeV
- −0.95 < cos(MET,1^{st}jet)<0.3
- cos(MET,2^{nd}jet) < 0.85



Start with jet finding, because it's simple(st)

Jets as projections



LO partons | NLO partons | parton shower | hadron level

Jet Def^{n} | Jet Def^{n} | Jet Def^{n} | Jet Def^{n}

jet 1 jet 2 | jet 1 jet 2 | jet 1 jet 2 | jet 1 jet 2

Projection to jets provides “common” view of different event levels

But projection is not unique: we must define what we mean by a jet

The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1

The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

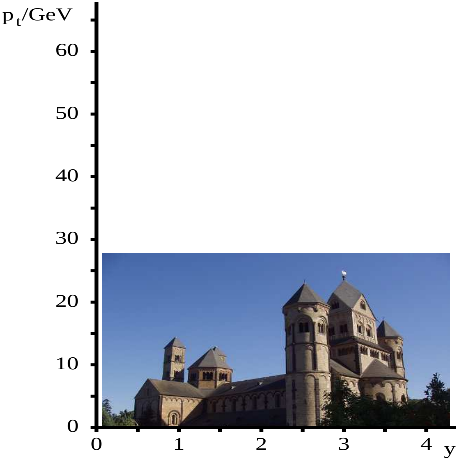
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

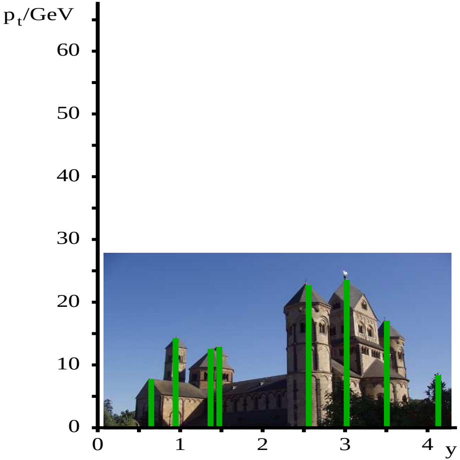
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

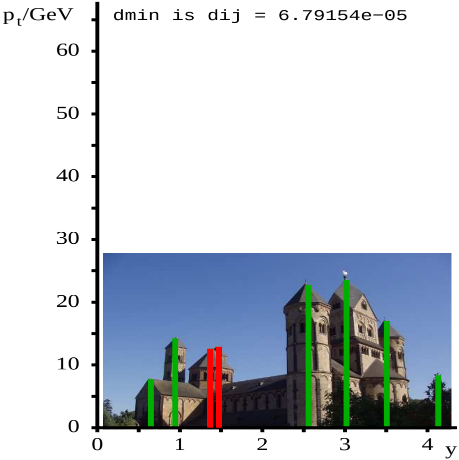
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

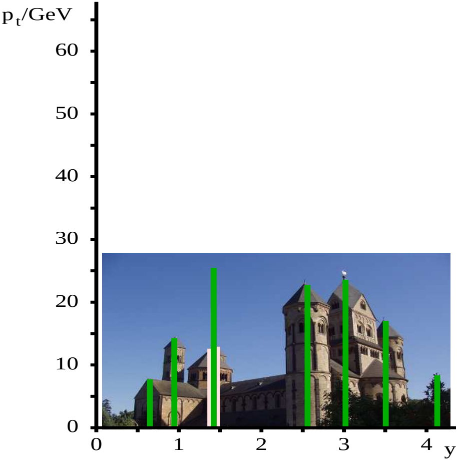
Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

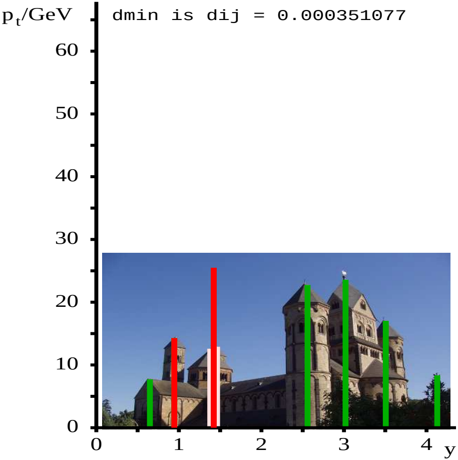
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

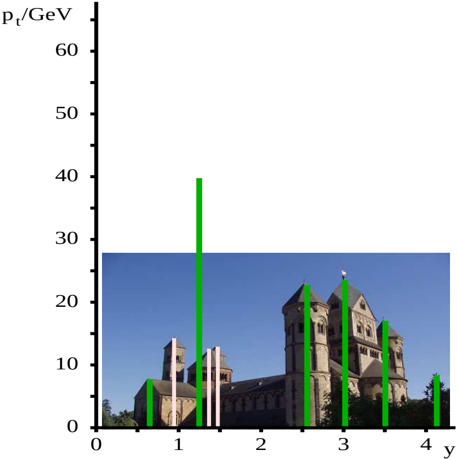
Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

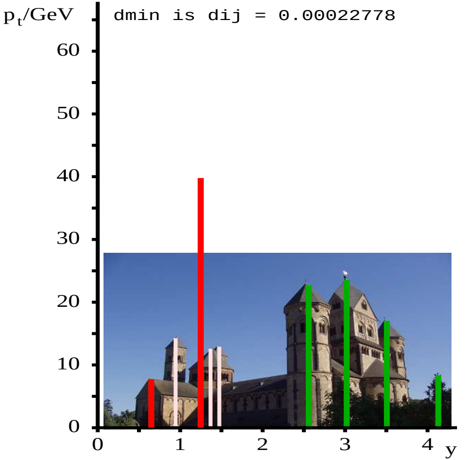
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

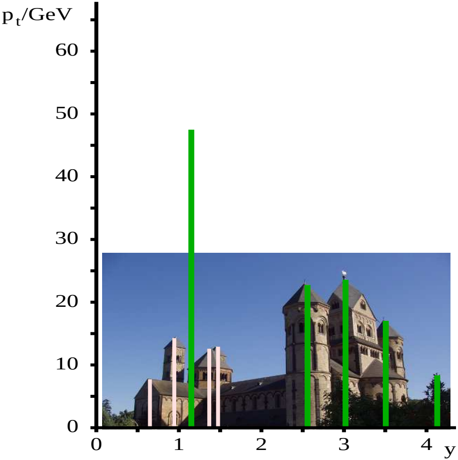
Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

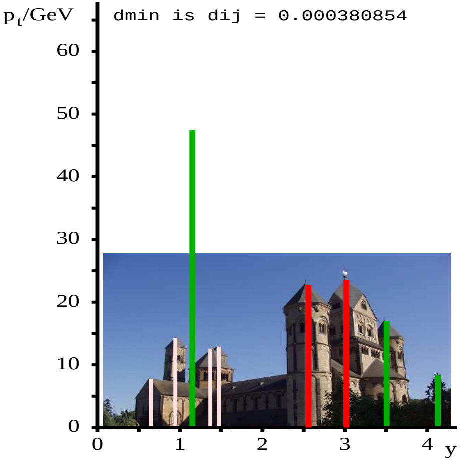
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

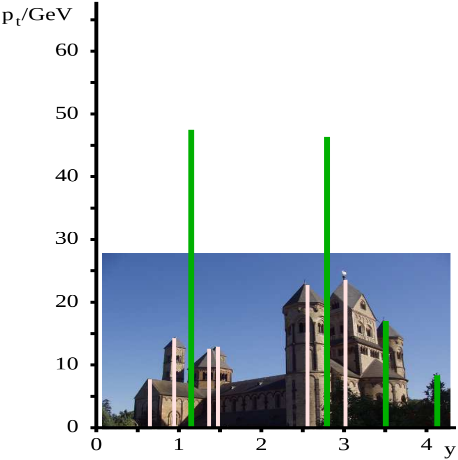
Define “distance” between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

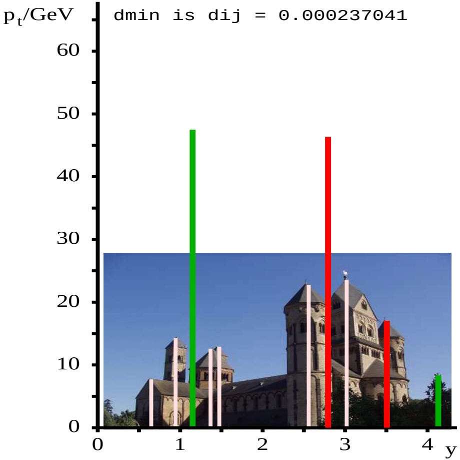
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

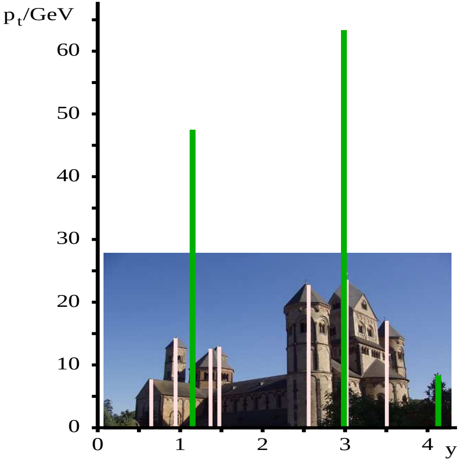
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

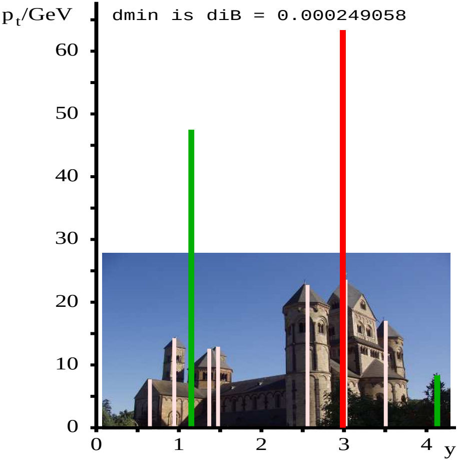
Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define “distance” between every pair of particles: [Cacciari, GPS & Soyez '08]

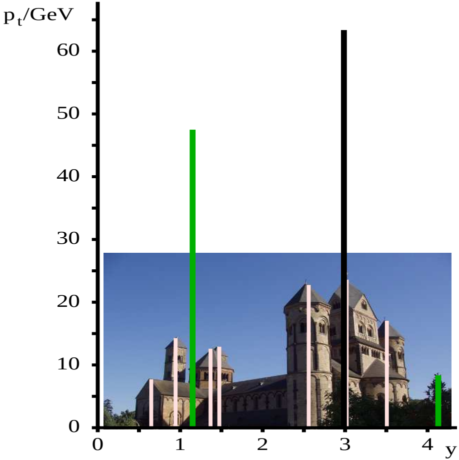
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

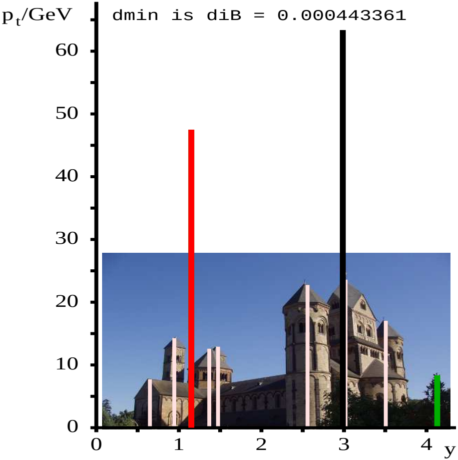
Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

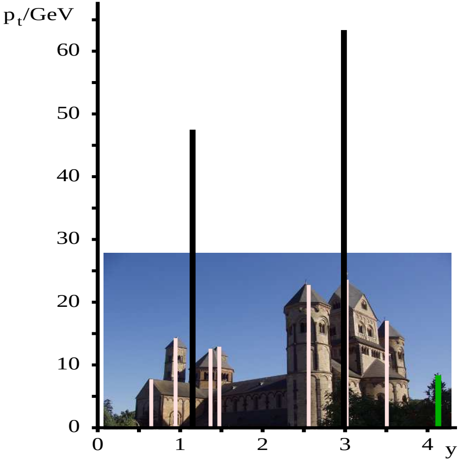
$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2}$$

$$\left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

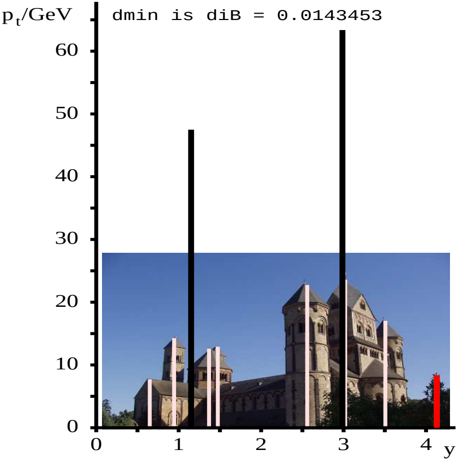
Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

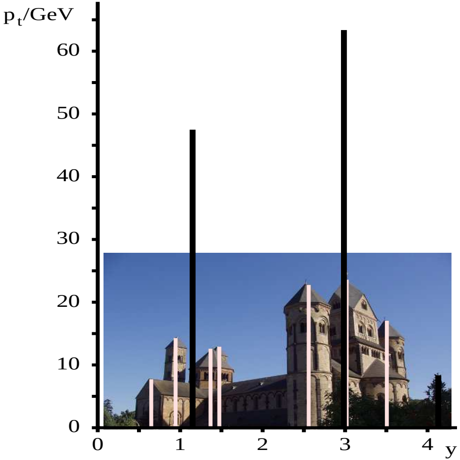
Define “distance” between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

Define a single-particle distance

$$d_{iB} = \frac{1}{p_{ti}^2}$$

1. Find the smallest of d_{ij} and d_{iB}
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define “distance” between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

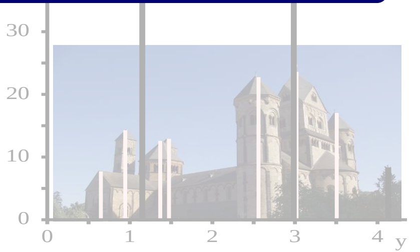
De

The algorithm involves two parameters:

1. R, the angular reach for the jets
2. A p_t threshold for the final jets to be considered relevant

1.

2. If it’s a d_{ij}, merge i and j into a single particle
3. If it’s a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



The anti-k_t jet algorithm

Define "distance" between every pair of particles: [Cacciari, GPS & Soyez '08]

$$d_{ij} = \frac{1}{\max(p_{ti}^2, p_{tj}^2)} \frac{\Delta R_{ij}^2}{R^2} \qquad \left[\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j^2)\right]$$

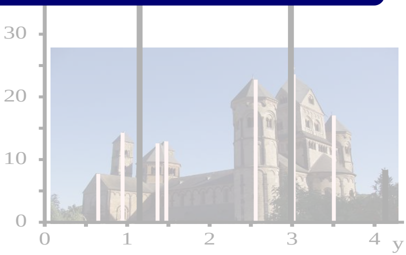
Def

The algorithm involves two parameters:

1. R, the angular reach for the jets
2. A p_t threshold for the final jets to be considered relevant

[It's the default algorithm for ATLAS & CMS]

1\.
2. If it's a d_{ij}, merge i and j into a single particle
3. If it's a d_{iB} call i a jet and remove it from list
4. Update all distances, go to step 1



What accuracy are our predictions?

It matters if we're say a signal is just an excess over expected backgrounds. . .

total X-section $e^+e^- \to Z \to$ hadrons

Start simply and look back at cross section for $e^+e^- \to Z \to$hadrons (at $\sqrt{s} \equiv Q = M_Z$).

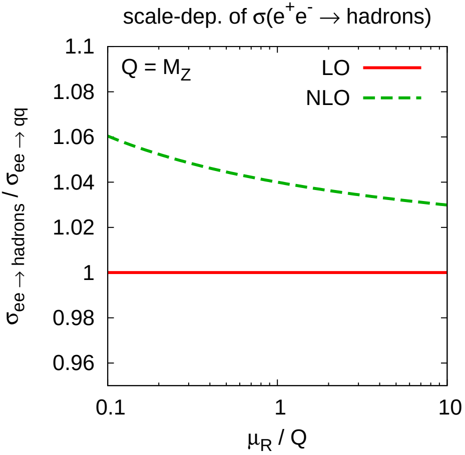
In lecture 1 we wrote:

$$\sigma_{tot} = \sigma_{q\bar{q}} \left(\underbrace{1}_{\text{LO}} + \underbrace{1.045 \frac{\alpha_s(Q)}{\pi}}_{\text{NLO}} + \underbrace{0.94 \left(\frac{\alpha_s(Q)}{\pi}\right)^2}_{\text{NNLO}} + \cdots \right)$$

Who told us we should we should write the series in terms of $\alpha_s(Q)$?

$Q = M_Z$ is the only physical scale in the problem, so not unreasonable.
But hardest possible gluon emission is $E = Q/2$. Should we have used $Q/2$?
And virtual gluons can have $E > Q$. Should we have used $2Q$?

Scale dependence



Start with the first order that "contains QCD" (NLO).

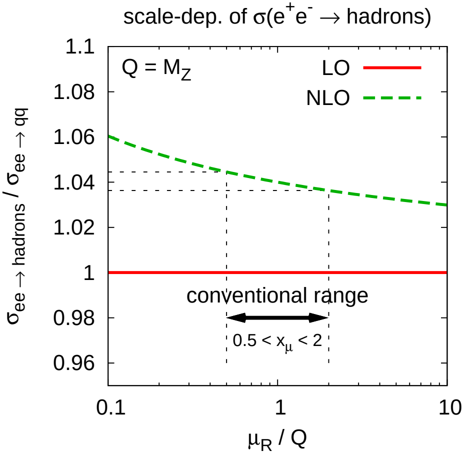
Introduce arbitrary **renormalisation scale** for the coupling, μ_R

$$\sigma^{\text{NLO}} = \sigma_{q\bar{q}}\,(1 + c_1\alpha_s(\mu_R)\,)$$

Result depends on the choice of μ_R.

Convention: the uncertainty on the result is the range of answers obtained for $Q/2 < \mu_R < 2Q$.

Scale dependence



Start with the first order that "contains QCD" (NLO).

Introduce arbitrary **renormalisation scale** for the coupling, μ_R

$$\sigma^{\text{NLO}} = \sigma_{q\bar{q}}\,(1 + c_1\alpha_s(\mu_R)\,)$$

Result depends on the choice of μ_R.

Convention: the uncertainty on the result is the range of answers obtained for $Q/2 < \mu_R < 2Q$.

Scale dependence (cont.)

Let's express results for arbitrary μ_R in terms of $\alpha_s(Q)$:

$$\sigma^{\mathrm{NLO}}(\mu_R) = \sigma_{q\bar{q}}\,(1 + c_1\,\alpha_s(\mu_R))$$
$$= \sigma_{q\bar{q}}\left(1 + c_1\,\alpha_s(Q) - 2c_1\,b_0 \ln\frac{\mu_R}{Q}\,\alpha_s^2(Q) + \mathcal{O}\left(\alpha_s^3\right)\right)$$

As we vary the renormalisation scale μ_R, we introduce $\mathcal{O}\left(\alpha_s^2\right)$ pieces into the X-section. I.e. generate some set of NNLO terms ~ uncertainty on X-section from missing NNLO calculation.

If we now calculate the full NNLO correction, then it will be structured so as to cancel the $\mathcal{O}\left(\alpha_s^2\right)$ scale variation

$$\sigma^{\mathrm{NNLO}}(\mu_R) = \sigma_{q\bar{q}}\left[1 + c_1\,\alpha_s(\mu_R) + c_2(\mu_R)\alpha_s^2(\mu_R)\right]$$

$$c_2(\mu_R) = c_2(Q) + 2c_1\,b_0 \ln\frac{\mu_R}{Q}$$

Remaining uncertainty is now $\mathcal{O}\left(\alpha_s^3\right)$.

Scale dependence (cont.)

Let's express results for arbitrary μ_R in terms of $\alpha_s(Q)$:

$$\sigma^{\mathrm{NLO}}(\mu_R) = \sigma_{q\bar{q}} \left(1 \right.$$

$$= \sigma_{q\bar{q}} \left(\right.$$

$$\alpha_s(\mu_R) = \frac{\alpha_s(Q)}{1 + 2b_0\,\alpha_s(Q)\,\ln \mu_R/Q}$$

$$= \alpha_s(Q) - 2b_0\,\alpha_s^2(Q)\,\ln \mu_R/Q + \mathcal{O}\left(\alpha_s^3\right)$$

As we vary the renormal

the X-section. I.e. generate some set of NNLO terms $\sim$ uncertainty on X-section from missing NNLO calculation.

If we now calculate the full NNLO correction, then it will be structured so as to cancel the $\mathcal{O}\left(\alpha_s^2\right)$ scale variation

$$\sigma^{\mathrm{NNLO}}(\mu_R) = \sigma_{q\bar{q}}\left[1 + c_1\,\alpha_s(\mu_R) + c_2(\mu_R)\alpha_s^2(\mu_R)\right]$$

$$c_2(\mu_R) = c_2(Q) + 2c_1\,b_0 \ln \frac{\mu_R}{Q}$$

Remaining uncertainty is now $\mathcal{O}\left(\alpha_s^3\right)$.

Scale dependence (cont.)

Let's express results for arbitrary μ_R in terms of $\alpha_s(Q)$:

$$\begin{aligned}\sigma^{\rm NLO}(\mu_R) &= \sigma_{q\bar q}\left(1 + c_1\,\alpha_s(\mu_R)\right) \\ &= \sigma_{q\bar q}\,\left(1 + c_1\,\alpha_s(Q) - 2c_1 b_0 \ln\frac{\mu_R}{Q}\,\alpha_s^2(Q) + \mathcal{O}\left(\alpha_s^3\right)\right)\end{aligned}$$

As we vary the renormalisation scale μ_R, we introduce $\mathcal{O}\left(\alpha_s^2\right)$ pieces into the X-section. I.e. generate some set of NNLO terms ~ uncertainty on X-section from missing NNLO calculation.

If we now calculate the full NNLO correction, then it will be structured so as to cancel the $\mathcal{O}\left(\alpha_s^2\right)$ scale variation

$$\sigma^{\rm NNLO}(\mu_R) = \sigma_{q\bar q}\,\left[1 + c_1\,\alpha_s(\mu_R) + c_2(\mu_R)\alpha_s^2(\mu_R)\right]$$

$$c_2(\mu_R) = c_2(Q) + 2c_1 b_0 \ln\frac{\mu_R}{Q}$$

Remaining uncertainty is now $\mathcal{O}\left(\alpha_s^3\right)$.

Scale dependence (cont.)

Let's express results for arbitrary μ_R in terms of $\alpha_s(Q)$:

$$\begin{aligned}\sigma^{\mathrm{NLO}}(\mu_R) &= \sigma_{q\bar{q}}\,(1 + c_1\,\alpha_s(\mu_R)\,) \\ &= \sigma_{q\bar{q}}\,\left(1 + c_1\,\alpha_s(Q) - 2c_1 b_0 \ln\frac{\mu_R}{Q}\,\alpha_s^2(Q) + \mathcal{O}\left(\alpha_s^3\right)\right)\end{aligned}$$

As we vary the renormalisation scale μ_R, we introduce $\mathcal{O}\left(\alpha_s^2\right)$ pieces into the X-section. I.e. generate some set of NNLO terms ∼ uncertainty on X-section from missing NNLO calculation.

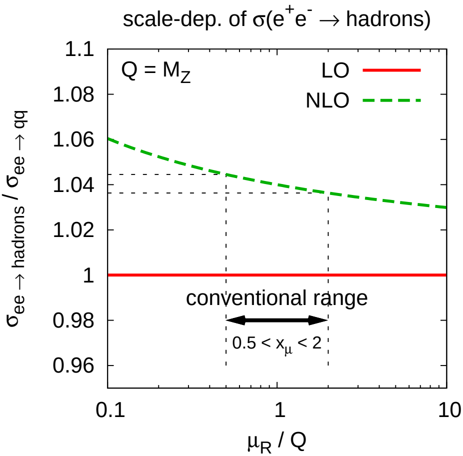
If we now calculate the full NNLO correction, then it will be structured so as to cancel the $\mathcal{O}\left(\alpha_s^2\right)$ scale variation

$$\sigma^{\mathrm{NNLO}}(\mu_R) = \sigma_{q\bar{q}}\,\left[1 + c_1\,\alpha_s(\mu_R) + c_2(\mu_R)\alpha_s^2(\mu_R)\,\right]$$

$$c_2(\mu_R) = c_2(Q) + 2c_1 b_0 \ln\frac{\mu_R}{Q}$$

Remaining uncertainty is now $\mathcal{O}\left(\alpha_s^3\right)$.

Scale dependence: NNLO



See how at NNLO, scale dependence is much flatter, final uncertainty much smaller.

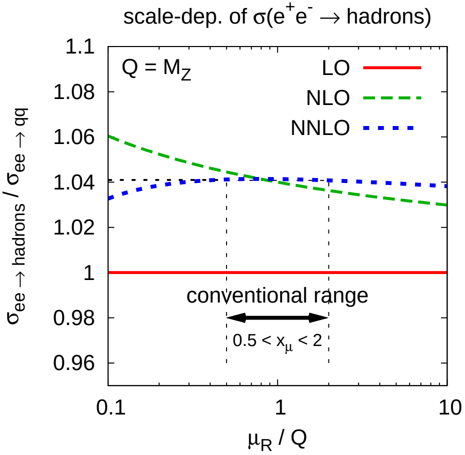
Because now we neglect only α_s^3 instead of α_s^2

Moral: not knowing exactly how to set scale → blessing in disguise, since it gives us handle on uncertainty.

Scale variation ≡ standard procedure
Often a good guide
Except when it isn't!

NB: if we had a large number of orders of perturbation theory, scale dependence would just disappear.

Scale dependence: NNLO



See how at NNLO, scale dependence is much flatter, final uncertainty much smaller.

Because now we neglect only α_s^3 instead of α_s^2

Moral: not knowing exactly how to set scale → blessing in disguise, since it gives us handle on uncertainty.

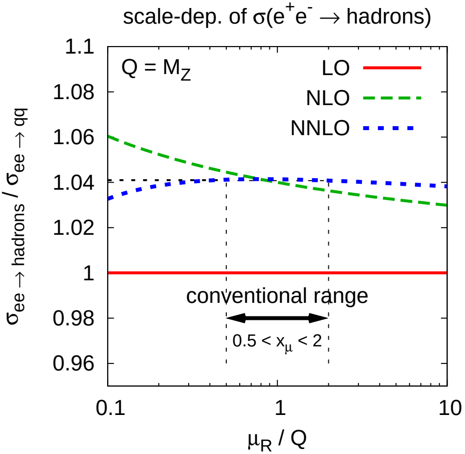
Scale variation ≡ standard procedure

Often a good guide

Except when it isn't!

NB: if we had a large number of orders of perturbation theory, scale dependence would just disappear.

Scale dependence: NNLO



See how at NNLO, scale dependence is much flatter, final uncertainty much smaller.

Because now we neglect only α_s^3 instead of α_s^2

Moral: not knowing exactly how to set scale → blessing in disguise, since it gives us handle on uncertainty.

Scale variation ≡ standard procedure
Often a good guide
Except when it isn't!

NB: if we had a large number of orders of perturbation theory, scale dependence would just disappear.

Now switch to looking at the Z cross section in pp

LO $pp \to Z$

$$\sigma^{\text{LO}}_{pp\to Z} = \sum_i \int dx_1 dx_2\, f_{q_i}(x_1, \mu_F^2)\, f_{\bar{q}_i}(x_2, \mu_F^2)\; \hat{\sigma}_{0,q_i\bar{q}_i\to Z}(x_1 p_1, x_2 p_2)\,,$$

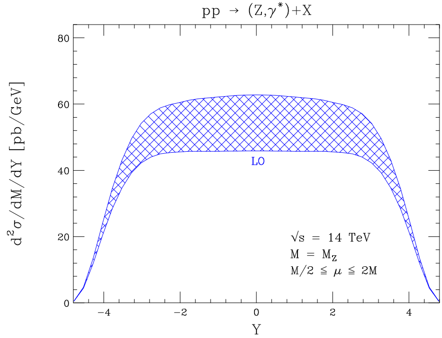
- $\sigma_{0,q_i\bar{q}_i\to Z} \propto \alpha_{EW}$, knows nothing about QCD like $\sigma_{e^+e^-\to Z}$
- But $\sigma_{0,q_i\bar{q}_i\to Z}$ depends on PDFs.
- We have to choose a **factorisation scale, μ_F**.
- Natural choice: $\mu_F = M_Z$, but one should vary it (just like the renorm. scale, μ_R, for α_s).

Plot shows $\sigma^{\text{LO}}_{pp\to Z}$ differentially as a function of rapidity (y) of Z. Band is uncertainty due to variation of μ_F.

LO $pp \to Z$

$$\sigma^{\mathrm{LO}}_{pp\to Z} = \sum_i \int dx_1 dx_2\, f_{q_i}(x_1, \mu_F^2)\, f_{\bar{q}_i}(x_2, \mu_F^2)\; \hat{\sigma}_{0,q_i\bar{q}_i\to Z}(x_1 p_1, x_2 p_2)\,,$$

- $\sigma_{0,q_i\bar{q}_i\to Z} \propto \alpha_{EW}$, knows nothing about QCD like $\sigma_{e^+e^-\to Z}$
- But $\sigma_{0,q_i\bar{q}_i\to Z}$ depends on PDFs.
- We have to choose a **factorisation scale, μ_F**.
- Natural choice: $\mu_F = M_Z$, but one should vary it (just like the renorm. scale, μ_R, for α_s).



$M_Z/2 \le \mu_F \le 2M_Z$

Plot shows $\sigma^{\mathrm{LO}}_{pp\to Z}$ differentially as a function of rapidity (y) of Z. Band is uncertainty due to variation of μ_F.

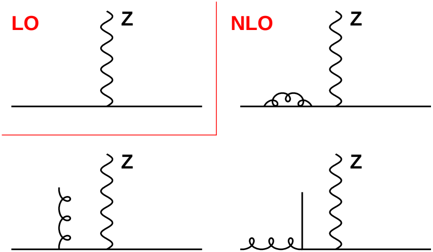
$pp \to Z + X$ at (N)NLO

$$\sigma^{\mathrm{NLO}}_{pp\to Z} = \sum_{i,j} \int dx_1 dx_2\, f_i(x_1, \mu_F^2)\, f_j(x_2, \mu_F^2)\, [\hat{\sigma}_{0,ij\to Z}(x_1, x_2) + \\ + \alpha_s(\mu_R)\hat{\sigma}_{1,ij\to Z}(x_1, x_2, \mu_F)]$$

- New channels open up ($gq \to Zq$)
- Now X-sct depends on renorm scale μ_R *and* fact. scale μ_F

  often vary $\mu_R = \mu_F$ together

  not necessarily "right"
- But $\hat{\sigma}_1$ piece cancels large LO dependence on μ_F
- At NNLO dependence on μ_R and μ_F is further cancelled



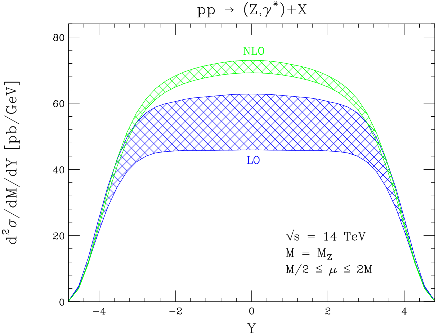
$pp \to Z + X$ at (N)NLO

$$\sigma^{\text{NLO}}_{pp\to Z} = \sum_{i,j} \int dx_1 dx_2\, f_i(x_1,\mu_F^2)\, f_j(x_2,\mu_F^2)\, [\hat{\sigma}_{0,ij\to Z}(x_1,x_2) + \\ + \alpha_s(\mu_R)\hat{\sigma}_{1,ij\to Z}(x_1,x_2,\mu_F)]$$

- New channels open up ($gq \to Zq$)
- Now X-sct depends on renorm scale μ_R *and* fact. scale μ_F

  often vary $\mu_R = \mu_F$ together

  not necessarily "right"
- But $\hat{\sigma}_1$ piece cancels large LO dependence on μ_F
- At NNLO dependence on μ_R and μ_F is further cancelled

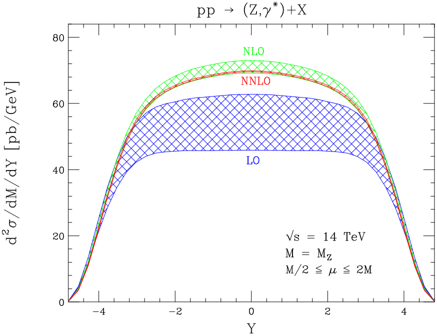


Anastasiou et al '03; $\mu_R = \mu_F$

$pp \to Z + X$ at (N)NLO

$$\sigma^{\text{NLO}}_{pp\to Z} = \sum_{i,j} \int dx_1 dx_2\, f_i(x_1,\mu_F^2)\, f_j(x_2,\mu_F^2)\, [\hat\sigma_{0,ij\to Z}(x_1,x_2) + \\ + \alpha_s(\mu_R)\hat\sigma_{1,ij\to Z}(x_1,x_2,\mu_F)]$$

- New channels open up ($gq \to Zq$)
- Now X-sct depends on renorm scale μ_R *and* fact. scale μ_F
  often vary $\mu_R = \mu_F$ together
  not necessarily "right"
- But $\hat\sigma_1$ piece cancels large LO dependence on μ_F
- At NNLO dependence on μ_R and μ_F is further cancelled



Anastasiou et al '03; $\mu_R = \mu_F$

Rules of thumb

In hadron-collider QCD calculations:

- Choose a sensible central scale for your process
- Vary μ_F, μ_R by a factor of two around that central value

- LO: good only to within factor of two

Despite $\alpha_s \simeq 0.1$

- NLO: good to within $10 - 20\%$
- NNLO: good to a few percent

The above rules fail if NLO/NNLO involve characteristically new production channels and/or large ratios of scales.

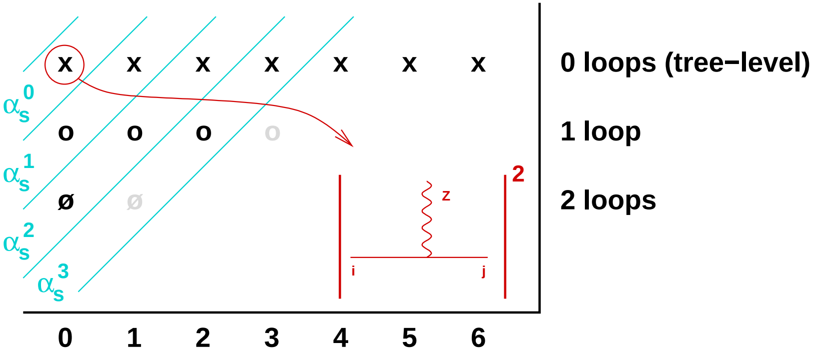
Diagrams / processes / orders

x	x	x	x	x	x	x	0 loops (tree−level)
o	o	o	o				1 loop
ø	ø						2 loops
0	1	2	3	4	5	6	

ij → Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the *p*-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

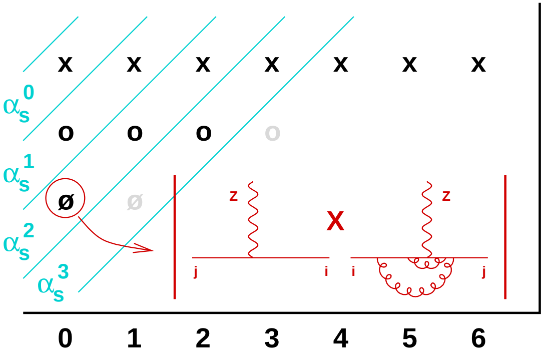
Diagrams / processes / orders



ij → Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders



0 loops (tree−level)

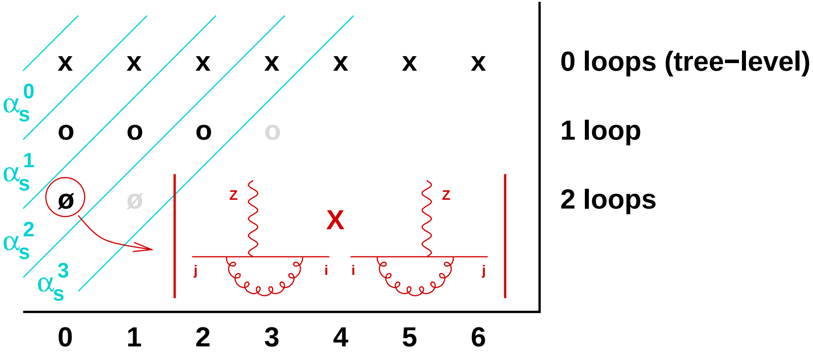
1 loop

2 loops

ij $\rightarrow$ Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

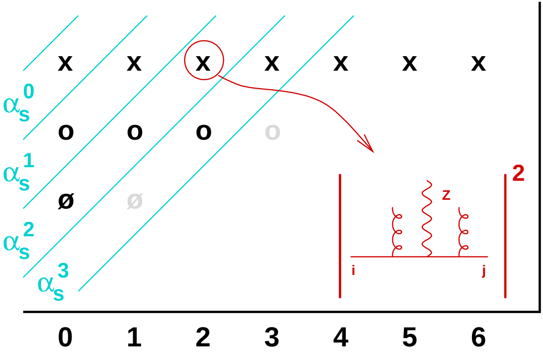
Diagrams / processes / orders



ij → Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the *p*-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders



0 loops (tree−level)

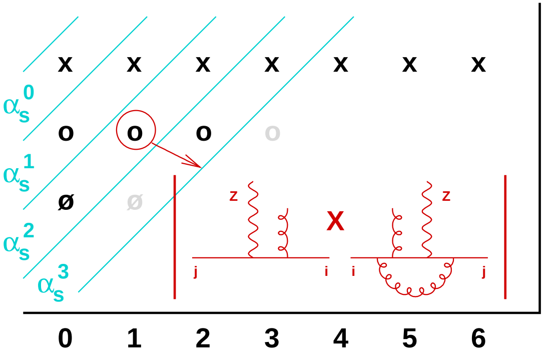
1 loop

2 loops

ij $\to$ Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders



0 loops (tree−level)

1 loop

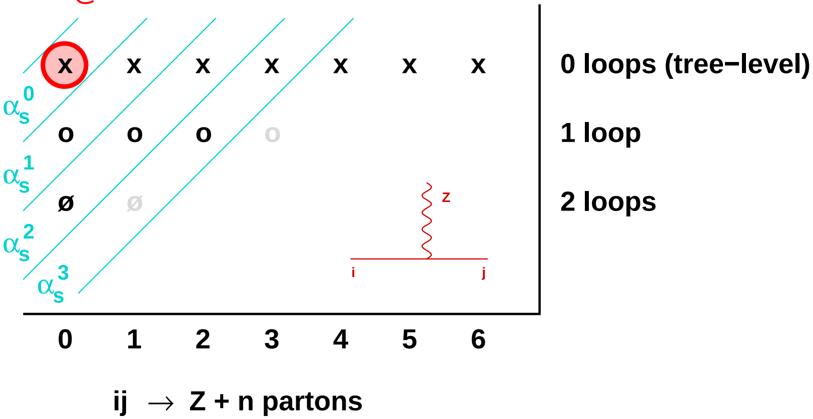
2 loops

ij $\rightarrow$ Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders

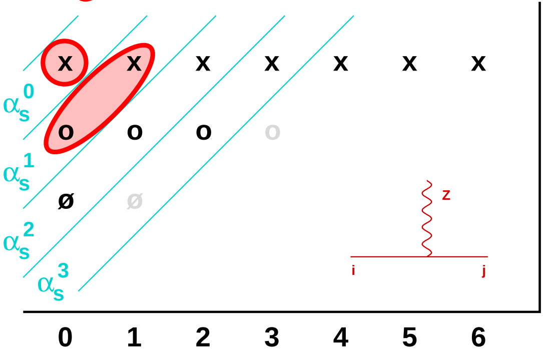
Z @ LO



The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders

Z @ NLO



0 loops (tree−level)

1 loop

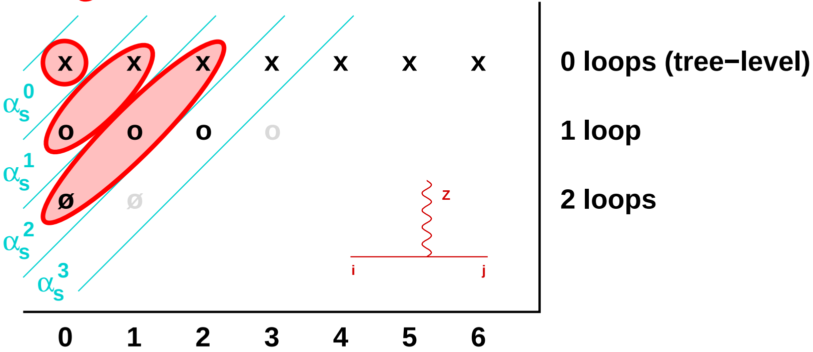
2 loops

ij → Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders

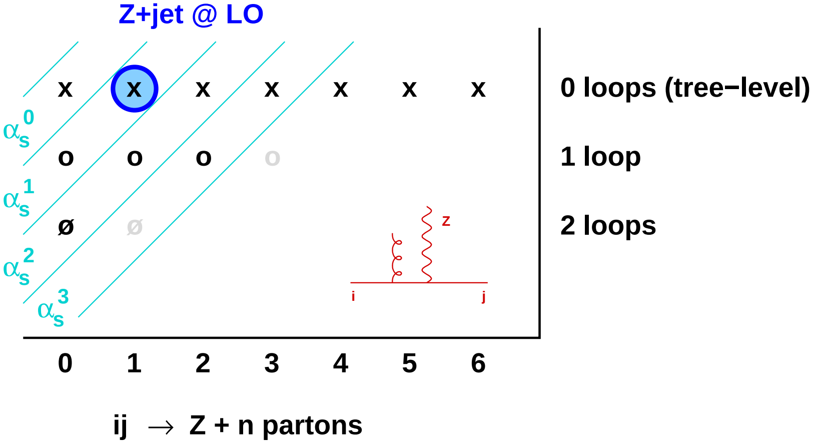
Z @ NNLO



ij $\to$ Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

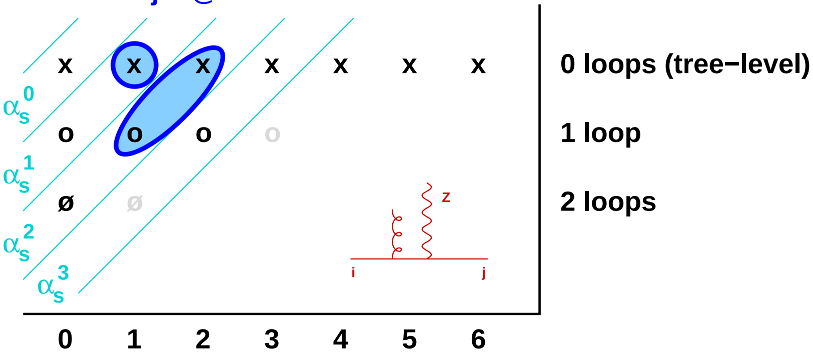
Diagrams / processes / orders



The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders

Z+jet @ NLO

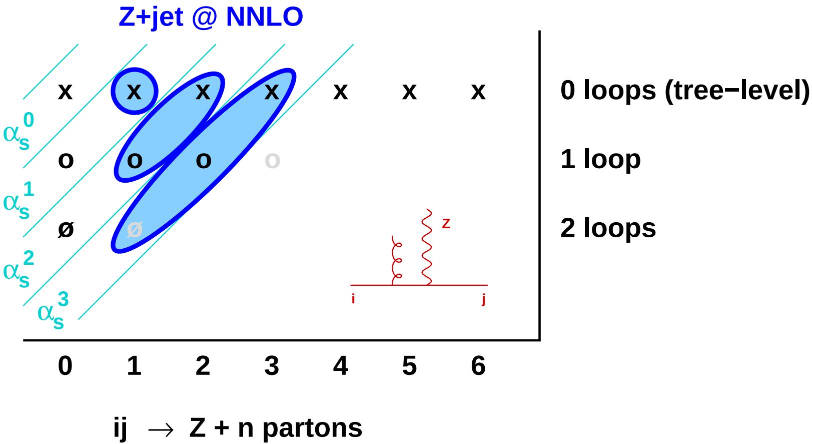


	0	1	2	3	4	5	6
0 loops (tree−level)	x	x	x	x	x	x	x
1 loop	o	o	o	o			
2 loops	ø	ø					

ij → Z + n partons

The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

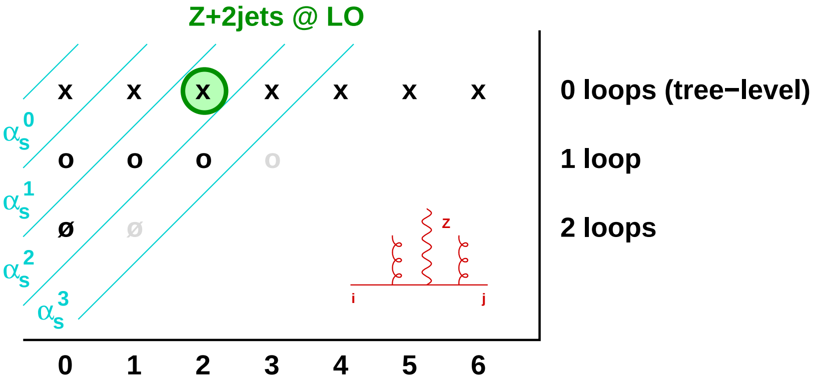
Diagrams / processes / orders



The bottleneck in getting N^{p}LO predictions is usually either the calculation of the *p*-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders

Z+2jets @ LO

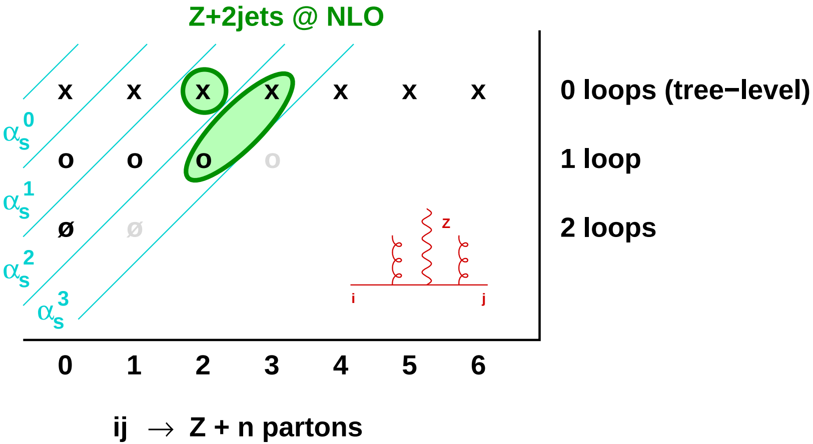


	0	1	2	3	4	5	6
0 loops (tree−level)	x	x	x	x	x	x	x
1 loop	o	o	o	o			
2 loops	ø	ø					

ij → Z + n partons

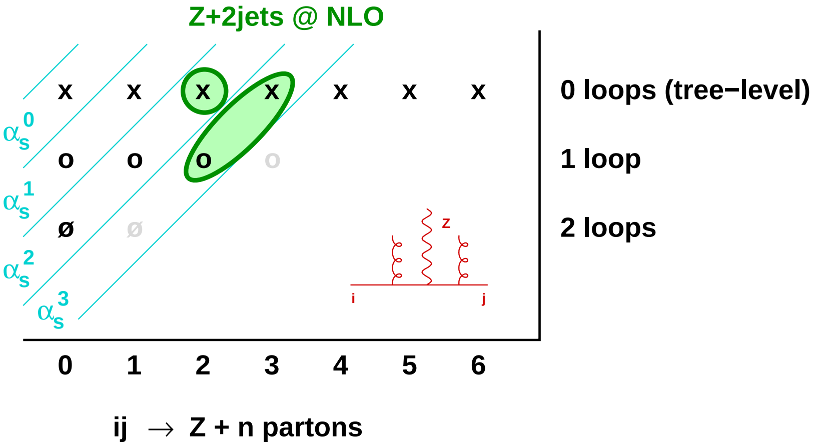
The bottleneck in getting N^{p}LO predictions is usually either the calculation of the *p*-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders



The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

Diagrams / processes / orders



The bottleneck in getting N^{p}LO predictions is usually either the calculation of the p-loop diagram, or figuring out how to combine (cancel) divergences between 2-loops, 1-loop & tree-level.

The limits of what we know

- Tree-level / LO: $\mathbf{2 \to 6 - 8}$

  ALPGEN, CompHep, Helac/Helas, Madgraph, Sherpa, Whizard
- 1-loop / NLO: $\mathbf{2 \to 3}$

  MCFM, NLOJet++, PHOX-family + various single-process codes

  Several 2 → 4 (and first 2 → 5) have appeared in past 18 months:

  Denner et al ($ttbb$), HELAC-NLO($ttjj, tt b\bar{b}$)

  Blackhat ($W/Z + 3j, W + 4j$), Rocket($W + 3j$)
- 2-loop / NNLO: $\mathbf{2 \to 1}$ (W,Z,H) FEWZ, FeHiP, HNNLO

Example of complexity of the calculations, for gg → N gluons:

Njets	2	3	4	5	6	7	8
# diags	4	25	220	2485	34300	5x10^5	10^7

Programs like Alpgen, Helac/Helas, Sherpa avoid Feynman diagrams and use methods that recursively build up amplitudes

The limits of what we know

- Tree-level / LO: **$2 \to 6 - 8$**

  ALPGEN, CompHep, Helac/Helas, Madgraph, Sherpa, Whizard
- 1-loop / NLO: **$2 \to 3$**

  MCFM, NLOJet++, PHOX-family + various single-process codes

  Several $2 \to 4$ (and first $2 \to 5$) have appeared in past 18 months:

  Denner et al ($ttbb$), HELAC-NLO($ttjj, tt b\bar{b}$)

  Blackhat ($W/Z + 3j, W + 4j$), Rocket($W + 3j$)
- 2-loop / NNLO: **$2 \to 1$** (W,Z,H) FEWZ, FeHiP, HNNLO

Example of complexity of the calculations, for gg → N gluons:

Njets	2	3	4	5	6	7	8
# diags	4	25	220	2485	34300	5x10^5	10^7

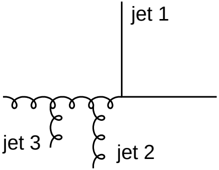
Programs like Alpgen, Helac/Helas, Sherpa avoid Feynman diagrams and use methods that recursively build up amplitudes

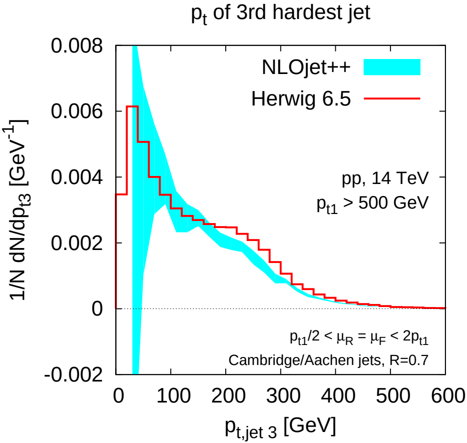
Fixed-order programs give controlled accuracy, but (partonic) final states and (at NLO, NNLO) divergent weights.

Monte Carlo Parton Shower programs give a “sensible” (hadronic) final state, with unit event weights, but ill-controlled accuracy.

How well do parton showers reproduce the LO/NLO results?

Multijet events





Generate hard dijet events, shower and hadronise them with Herwig.

Select events in which hardest jet has $p_t > 500$ GeV. Look at p_t distribution of 3rd hardest jet

- Herwig doesn't do too bad a job of reproducing high-p_t 3rd-jet rate

  But no uncertainty band

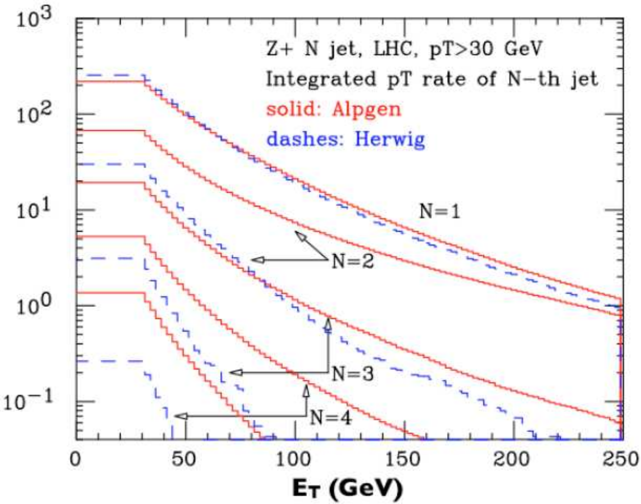
  Hard to know how trustworthy

  unless you also have NLO

- NLO does poor job at low p_t — large ratios of scales, $p_{t3}/p_{t1} \ll 1$, are dangerous in fixed-order calculations.

  higher-orders $\sim \alpha_s \ln \frac{p_{t1}}{p_{t3}} \sim 1$

$Z + N$ jets



Herwig: select $Z+1$ jet hard process.

Look at p_t distribution of jets with highest p_t, 2nd highest p_t, etc.

Compare to tree-level calculation

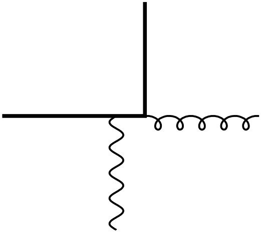
Mangano '08

Parton shower (Herwig) does very badly even just for 2nd jet.
Why is this so much worse than in the pure jet case?

- Tree-level (LO) gives decent description of multi-jet structure
- NLO gives good normalisation
- Parton-shower gives good behaviour in soft-collinear regions and fully exclusive final state.

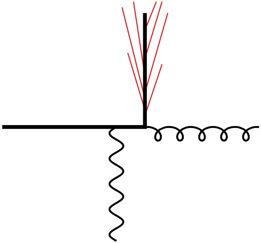
Can we combine the advantages of all three?
[Here we'll look at just Tree + Parton shower]

Add Z+1jet, Z+2jet + shower



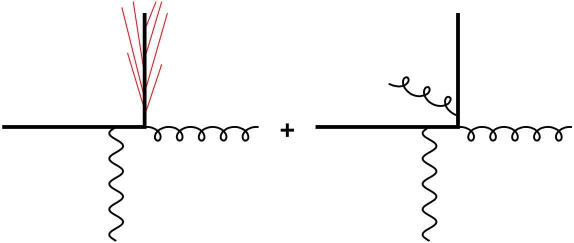
Z+parton

Add Z+1jet, Z+2jet + shower



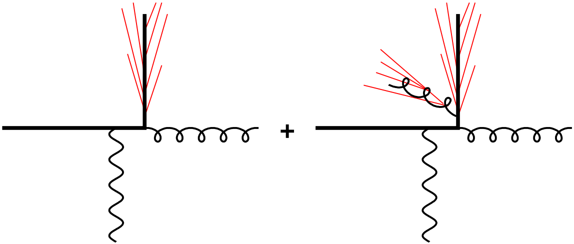
shower Z+parton

Add Z+1jet, Z+2jet + shower



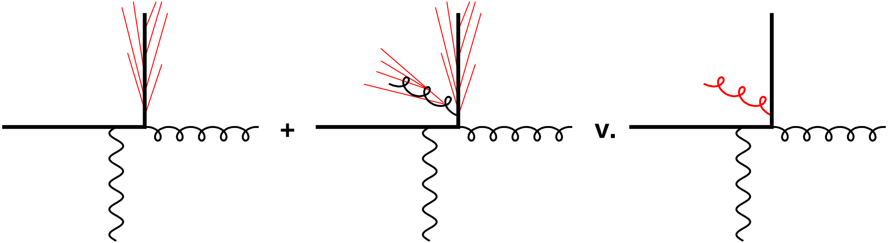
shower **Z+parton** **Z+2partons**

Add Z+1jet, Z+2jet + shower



shower Z+parton **shower Z+2partons**

Add Z+1jet, Z+2jet + shower

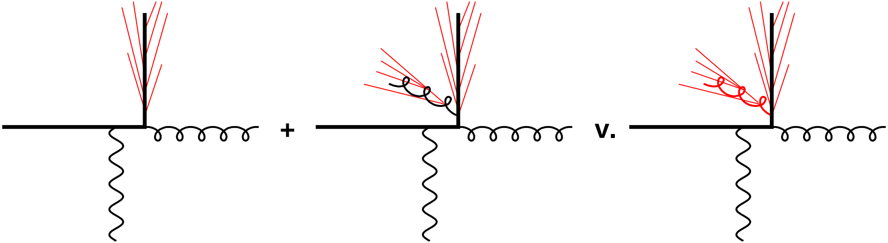


shower Z+parton

shower Z+2partons

shower of Z+parton
generates hard gluon

Add Z+1jet, Z+2jet + shower

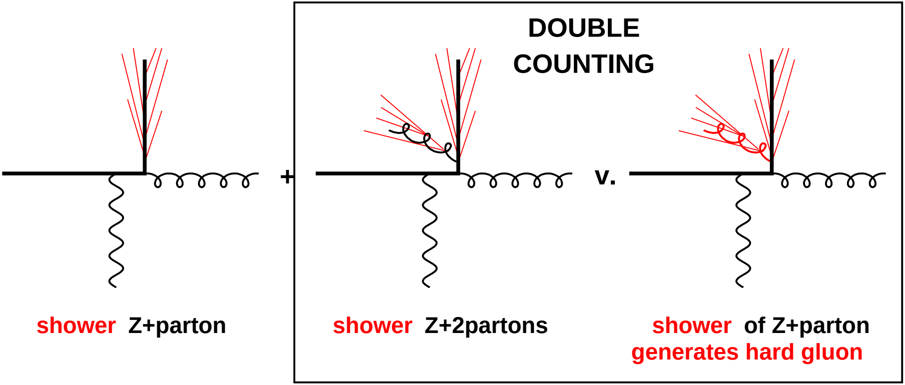


shower Z+parton

shower Z+2partons

shower of Z+parton generates hard gluon

Add Z+1jet, Z+2jet + shower



+

DOUBLE COUNTING

v.

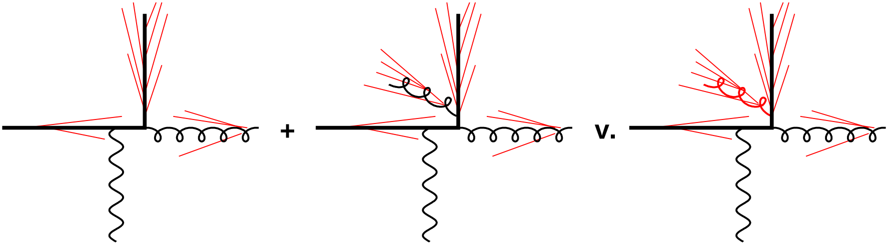
shower Z+parton

shower Z+2partons

shower of Z+parton generates hard gluon

Double counting + associated issues with virtual corrections are the main problems when merging PS + ME

"MLM" matching in a nutshell



shower Z+parton

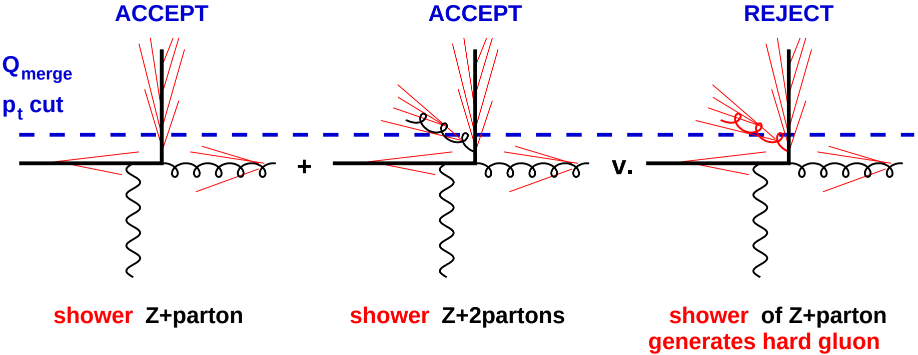
shower Z+2partons

shower of Z+parton
generates hard gluon

- Hard jets above scale Q_{merge} have distributions given by tree-level ME
- Rejection procedure eliminates "double-counted" jets from parton shower
- Rejection generates Sudakov form factors between individual jet scales

How well? Depends on details of PS. One of the weaker points of MLM

"MLM" matching in a nutshell


ACCEPT

ACCEPT

REJECT

Q_{merge}

p_t cut

+

v.

shower Z+parton

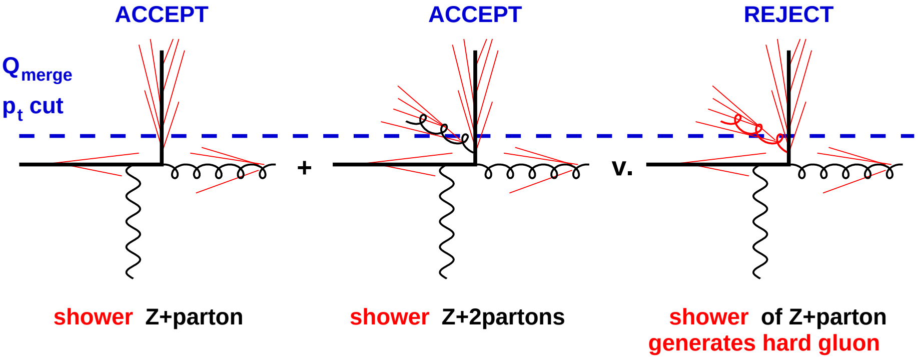
shower Z+2partons

shower of Z+parton
generates hard gluon

- Hard jets above scale Q_{merge} have distributions given by tree-level ME
- Rejection procedure eliminates "double-counted" jets from parton shower
- Rejection generates Sudakov form factors between individual jet scales

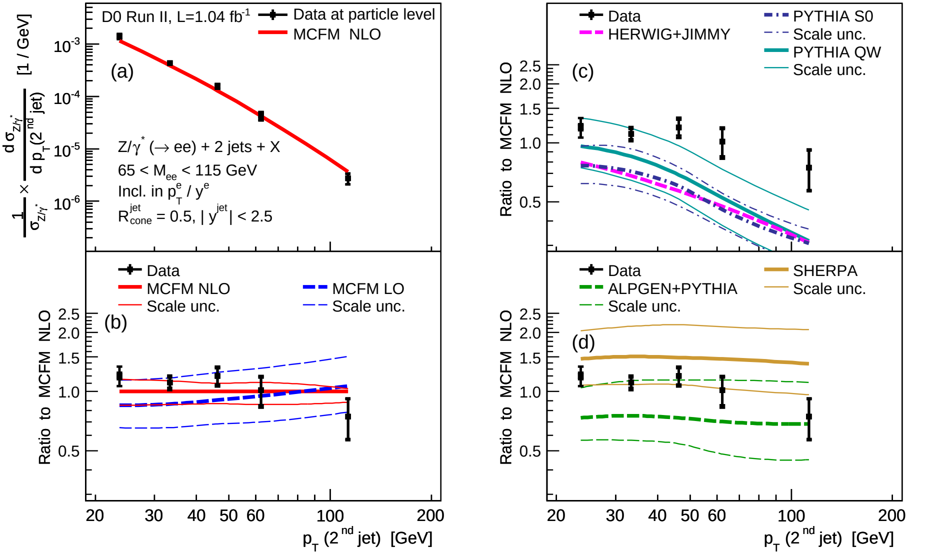
How well? Depends on details of PS. One of the weaker points of MLM

"MLM" matching in a nutshell



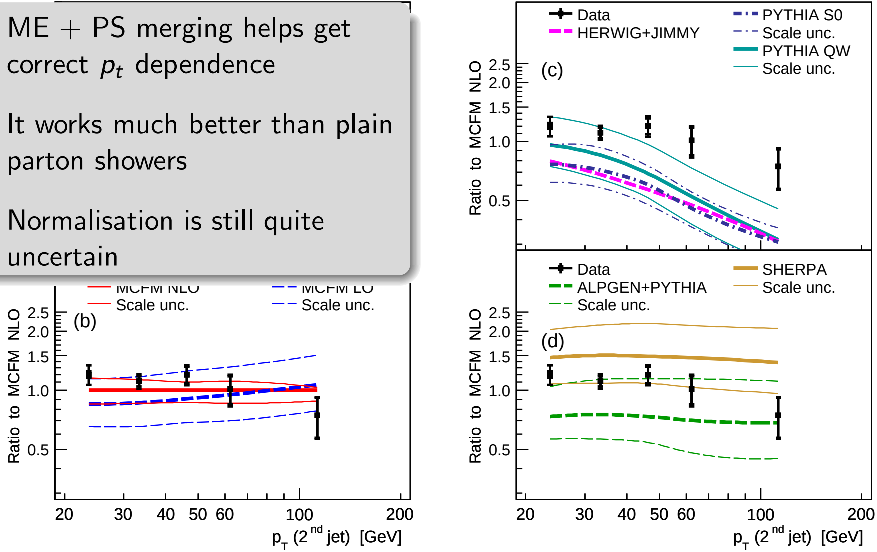
- Hard jets above scale Q_{merge} have distributions given by tree-level ME
- Rejection procedure eliminates "double-counted" jets from parton shower
- Rejection generates Sudakov form factors between individual jet scales
  How well? Depends on details of PS. One of the weaker points of MLM

Z + 2 jets


D0 Run II, L=1.04 fb^{-1}
Data at particle level
MCFM NLO
(a)
Z/γ^* (→ ee) + 2 jets + X
$65 < M_{ee} < 115$ GeV
Incl. in p_T^e / y^e
$R_{cone}^{jet} = 0.5$, $|y^{jet}| < 2.5$
$\frac{1}{\sigma_{Z/\gamma^*}} \times \frac{d\sigma_{Z/\gamma^*}}{d p_T (2^{nd}\text{ jet})}$ [1 / GeV]
10^{-3}
10^{-4}
10^{-5}
10^{-6}
(b)
Data
MCFM NLO
Scale unc.
MCFM LO
Scale unc.
Ratio to MCFM NLO
(c)
Data
HERWIG+JIMMY
PYTHIA S0
Scale unc.
PYTHIA QW
Scale unc.
(d)
Data
ALPGEN+PYTHIA
Scale unc.
SHERPA
Scale unc.
2.5
2.0
1.5
1.0
0.5
20
30
40
50
60
100
200
p_T (2^{nd} jet) [GeV]

Z + 2 jets

- ME + PS merging helps get correct p_t dependence
- It works much better than plain parton showers
- Normalisation is still quite uncertain



Conclusions

Conclusions

Over the course of these lectures we've seen some of the basic elements of QCD for hadron colliders.

We've slowly been approaching the frontiers of the subject:

- Can you do accurate matrix-element (loop) calculations for the multi-jet discovery signatures at LHC?

  Blackhat/Rocket/HELAC-NLO teams are making big advances on NLO
  NNLO is still very tough, basically only for $pp \to H/W/Z$

- How do you put together the soft/collinear approximation (parton showers) and exact exact matrix-element calculations?

  We've looked at tree-level + parton showers (need for cutoff is ugly)
  Also NLO + parton shower [MC@NLO, POWHEG, MENLOPS]

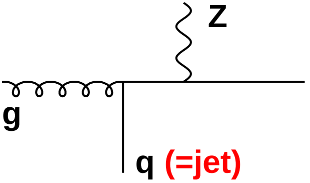
- How do you organise the information in an event to make signals emrge most clearly?

  Novel ways of using jets

EXTRAS

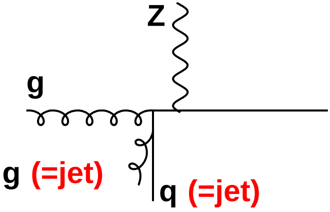
Why parton shower so poor for Z+jets?

Z + 1 jet



$\alpha_s \alpha_{EW}$

Z + 2 jets



Produced by parton shower

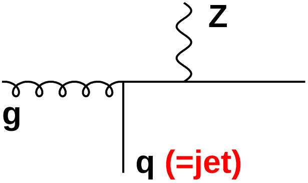
Parton showers generate starting from hard process you asked for.

Z/W + multijet production involves **two classes of hard process**

A. Z + recoil jet; **B.** dijets + emission of Z (missing from MC)

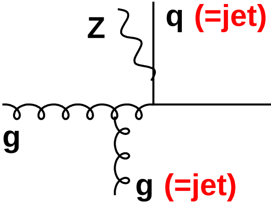
Why parton shower so poor for Z+jets?

Z + 1 jet



$\alpha_s \alpha_{EW}$

Z + 2 jets



Not produced by parton shower
enhanced at high p_t: $\alpha_s^2 \alpha_{EW} \ln^2 \frac{p_t}{M_Z}$

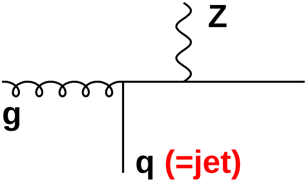
Parton showers generate starting from hard process you asked for.

Z/W + multijet production involves **two classes of hard process**

A. Z + recoil jet; **B.** dijets + emission of Z (missing from MC)

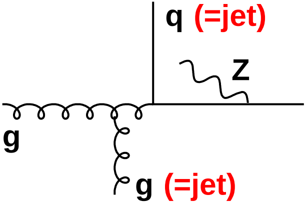
Why parton shower so poor for Z+jets?

Z + 1 jet



$\alpha_s \alpha_{EW}$

Z + 2 jets



Not produced by parton shower
enhanced at high p_t: $\alpha_s^2 \alpha_{EW} \ln^2 \frac{p_t}{M_Z}$

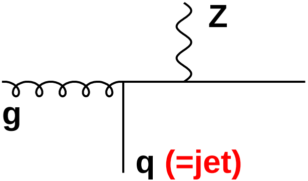
Parton showers generate starting from hard process you asked for.

Z/W + multijet production involves **two classes of hard process**

A. Z + recoil jet; **B.** dijets + emission of Z (missing from MC)

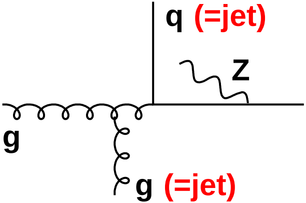
Why parton shower so poor for Z+jets?

Z + 1 jet



$\alpha_s \alpha_{EW}$

Z + 2 jets



Not produced by parton shower
enhanced at high p_t: $\alpha_s^2 \alpha_{EW} \ln^2 \frac{p_t}{M_Z}$

Parton showers generate starting from hard process you asked for.

Z/W + multijet production involves **two classes of hard process**

A. Z + recoil jet; **B.** dijets + emission of Z (missing from MC)