# Very brief talk on :
# In 4h from newbie to 1000 of ready-to-analyze $Z' \rightarrow tT$ events

Johannes Erdmann
Yale University

# What this talk is about

- Z' → tT benchmark analysis with Chris Pollard (Duke), Tobias Golling (Yale) and Aaron Effron (Yale)

- as so many others :
  - I know deadline is approaching, but busy with many other things …

  - we're using the 3 TeV Z' sample produced by Sergey Chekanov
  - what if we urgently need a 4 TeV sample ?
    - how much effort is it ? → better try now than 3 days before deadline

- had 4 spare hours yesterday and wanted to see if I can
  (1) set up the framework
  (2) produce 1000 Pythia Z' events
  (3) pass them through Delphes
  (4) make an invariant mass plot in ROOT

- Disclaimer : This is not a tutorial, just my very limited personal experience !

# (1) set up the framework

- started here
  https://atlaswww.hep.anl.gov/asc/wikidoc/doku.php?id=snowmass2013:howto_d3
- 200 MB tar-ball containing everything and instructions [ for linux ]
  (Delphes, Pythia, HepMC, Herwig++, fastjet, cards, generation scripts …)

- 1st surprise: tar-ball link broken → fixed immediately by Sergey
- downloaded to Yale cluster
- 2nd surprise: several packages didn't compile out-of-the-box
  - might be my setup / the machine / s.th. else (but failed also on lxplus)
  - needed to tweak build script a bit to get those packages I needed compiled
    (Delphes, fastjet, HepMC, HepMC_slim, Pythia)
  → will report back to Sergey
  - time estimate ~ 2h

- good news : from then on straight-forward !

# (2) produce 1000 Pythia Z' events

- Package provides folders for processes already generated

```
> cd run_zprime
> export LD_LIBRARY_PATH=$PWD/../HepMC/lib:$LD_LIBRARY_PATH
> export LD_LIBRARY_PATH=$PWD/../fastjetinstall/lib:$LD_LIBRARY_PATH
> make
> ./analysis.exe
```

- ran w/o problems
- time estimate : negligible

- Pythia configuration file : `analysis.ini`
  (1000 events @ 14 TeV)

- Produced HepMC output

# (3) pass them through Delphes

- run card : http://cmssw.cvs.cern.ch/cgi-bin/cmssw.cgi/UserCode/spadhi/Snowmass/Cards/

```
##############################
# Electron tracking efficiency - ID
##############################

module Efficiency ElectronTrackingEfficiency {
  set InputArray ParticlePropagator/electrons
  set OutputArray electrons

  # set EfficiencyFormula {efficiency formula as a function of eta and pt}
  # tracking efficiency formula for electrons
  set EfficiencyFormula {                                                (pt <= 0.1)    * (0.00) + \
                                 (abs(eta) <= 1.5) * (pt > 0.1    && pt <= 1.0)    * (0.85) + \
                                 (abs(eta) <= 1.5) * (pt > 1.0    && pt <= 1.0e2)  * (0.97) + \
                                 (abs(eta) <= 1.5) * (pt > 1.0e2)                  * (0.99) + \
                (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 0.1    && pt <= 1.0)    * (0.85) + \
                (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 1.0    && pt <= 1.0e2)  * (0.90) + \
                (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 1.0e2)                  * (0.95) + \
                (abs(eta) > 2.5)                                                    * (0.00)}
}
```

- PU file MinBias100K_14TeV.pileup from http://red-gridftp11.unl.edu/Snowmass/

```
> cd delphes; ln -s MinBias100K_14TeV.pileup MinBias.pileup
```

- running Delphes :

http://www.snowmass2013.org/tiki-index.php?page=Energy_Frontier_FastSimulation

```
> ./DelphesHepMC delphes_card_Snowmass_50PileUp.tcl <out.root> <in.hepmc>
```

- changed to saving all truth particles (except for just some as it's the default)

```
add Branch Delphes/allParticles Particle GenParticle
#add Branch StatusPid/filteredParticles Particle GenParticle
```

- ran w/o problems, time estimate : O(20 min.), output : ROOT file

# (4) make an invariant mass plot in ROOT

- managed to install fastjet and Delphes on my macbook
  (only issue : 32bit compatibility for fastjet)

```
./configure --prefix=<install dir.> CXXFLAGS=-O2 -m32 -g -Wall CFLAGS=-O2 -m32 -g
-Wall --build=i386 --host=i386
```

- How to read the ntuple ?
  - followed ex. https://cp3.irmp.ucl.ac.be/projects/delphes/wiki/WorkBook/QuickTour

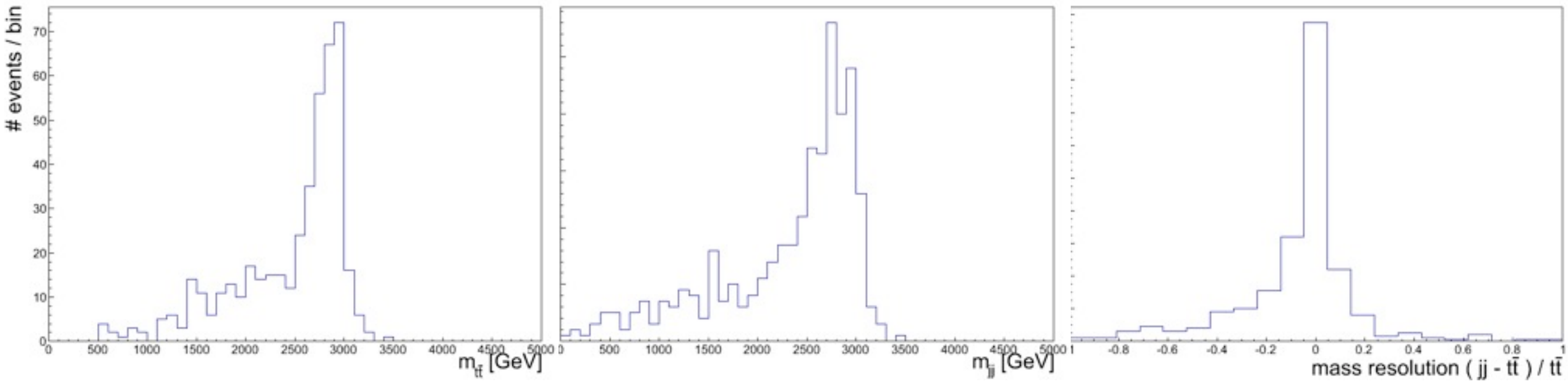
  - be aware that the Delphes ntuple is not flat (it's actually convenient)
  - class reference :
    https://cp3.irmp.ucl.ac.be/projects/delphes/wiki/WorkBook/RootTreeDescription
  - or just :

```
> root -l <file.root>
root [0] Delphes->Print()
```

```
*.....................................................................*
*Br  137 :Jet.PT     : Float_t PT[Jet_]                               *
*Entries :     1000 : Total  Size=      33598 bytes  File Size  =     29040 *
*Baskets :        4 : Basket Size=      11776 bytes  Compression=   1.14     *
*.....................................................................*
*Br  138 :Jet.Eta    : Float_t Eta[Jet_]                              *
*Entries :     1000 : Total  Size=      33606 bytes  File Size  =     29700 *
*Baskets :        4 : Basket Size=      11776 bytes  Compression=   1.11     *
*.....................................................................*
*Br  139 :Jet.Phi    : Float_t Phi[Jet_]                              *
*Entries :     1000 : Total  Size=      33606 bytes  File Size  =     29506 *
*Baskets :        4 : Basket Size=      11776 bytes  Compression=   1.12     *
*.....................................................................*
*Br  140 :Jet.Mass   : Float_t Mass[Jet_]                             *
*Entries :     1000 : Total  Size=      33614 bytes  File Size  =     29321 *
*Baskets :        4 : Basket Size=      11776 bytes  Compression=   1.13     *
*.....................................................................*
```

# (4) make an invariant mass plot in ROOT

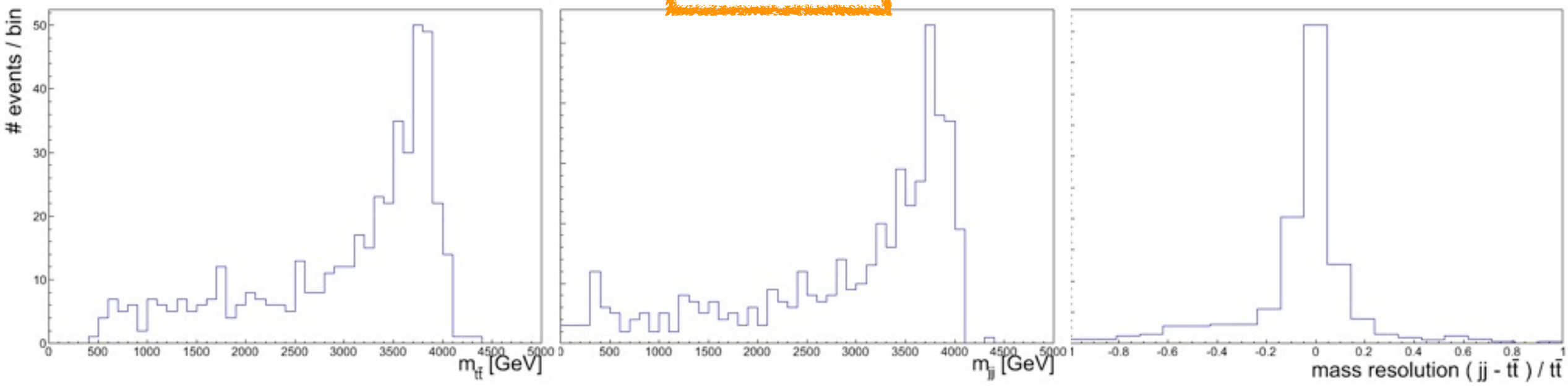
```
gSystem->Load("bin/Delphes-3.0.8/libDelphes.so");

TChain chain("Delphes");
chain.Add("output.root");
ExRootTreeReader * treeReader = new ExRootTreeReader(&chain);

TClonesArray * branchJet = treeReader->UseBranch("Jet");
TClonesArray * branchParticle = treeReader->UseBranch("Particle");
[...]
for(int iEntry(0), nEntries(treeReader->GetEntries()); iEntry < nEntries; ++iEntry) {
  treeReader->ReadEntry(iEntry);

  for (int iP(0), nP(branchParticle->GetEntries()); iP < nP; ++iP) {
    GenParticle * p = (GenParticle*)branchParticle->At(iP);
    TLorentzVector v;
    v.SetPtEtaPhiM(p->PT, p->Eta, p->Phi, p->Mass);
    [...]
  }
  [...]
  Jet * jet1 = (Jet*)branchJet->At(0);
  Jet * jet2 = (Jet*)branchJet->At(1);

  TLorentzVector v1;
  v1.SetPtEtaPhiM(jet1.PT, jet1.Eta, jet1.Phi, jet1.Mass);
  [...]
```



# (4) make an invariant mass plot in ROOT

```
gSystem->Load("bin/Delphes-3.0.8/libDelphes.so");

TChain chain("Delphes");
chain.Add("output.root");
ExRootTreeReader * treeReader = new ExRootTreeReader(&chain);

TClonesArray * branchJet = treeReader->UseBranch("Jet");
TClonesArray * branchParticle = treeReader->UseBranch("Particle");
[...]
for(int iEntry(0), nEntries(treeReader->GetEntries()); iEntry < nEntries; ++iEntry) {
  treeReader->ReadEntry(iEntry);

  for (int iP(0), nP(branchParticle->GetEntries()); iP < nP; ++iP) {
    GenParticle * p = (GenParticle*)branchParticle->At(iP);
    TLorentzVector v;
    v.SetPtEtaPhiM(p->PT, p->Eta, p->Phi, p->Mass);
    [...]
  }
  [...]
  Jet * jet1 = (Jet*)branchJet->At(0);
  Jet * jet2 = (Jet*)branchJet->At(1);

  TLorentzVector v1;
  v1.SetPtEtaPhiM(jet1.PT, jet1.Eta, jet1.Phi, jet1.Mass);
  [...]
```

**4 TeV**



# (4) make an invariant mass plot in ROOT

```
gSystem->Load("bin/Delphes-3.0.8/libDelphes.so");

TChain chain("Delphes");
chain.Add("output.root");
ExRootTreeReader * treeReader = new ExRootTreeReader(&chain);

TClonesArray * branchJet = treeReader->UseBranch("Jet");
TClonesArray * branchParticle = treeReader->UseBranch("Particle");
[...]
for(int iEntry(0), nEntries(treeReader->GetEntries()); iEntry < nEntries; ++iEntry) {
  treeReader->ReadEntry(iEntry);

  for (int iP(0), nP(branchParticle->GetEntries()); iP < nP; ++iP) {
    GenParticle * p = (GenParticle*)branchParticle->At(iP);
    TLorentzVector v;
    v.SetPtEtaPhiM(p->PT, p->Eta, p->Phi, p->Mass);
    [...]
  }
  [...]
  Jet * jet1 = (Jet*)branchJet->At(0);
  Jet * jet2 = (Jet*)branchJet->At(1);

  TLorentzVector v1;
  v1.SetPtEtaPhiM(jet1.PT, jet1.Eta, jet1.Phi, jet1.Mass);
  [...]
```

**It was pretty straight-forward to get started from basically nothing !**

**4 TeV**