Jet mass predictions in dijet events with PYTHIA8

Reproduction of results published in JHEP 11 (2018) 113 by the CMS collaboration

Ola Lelek, Mees van Kampen

General settings for the predictions:

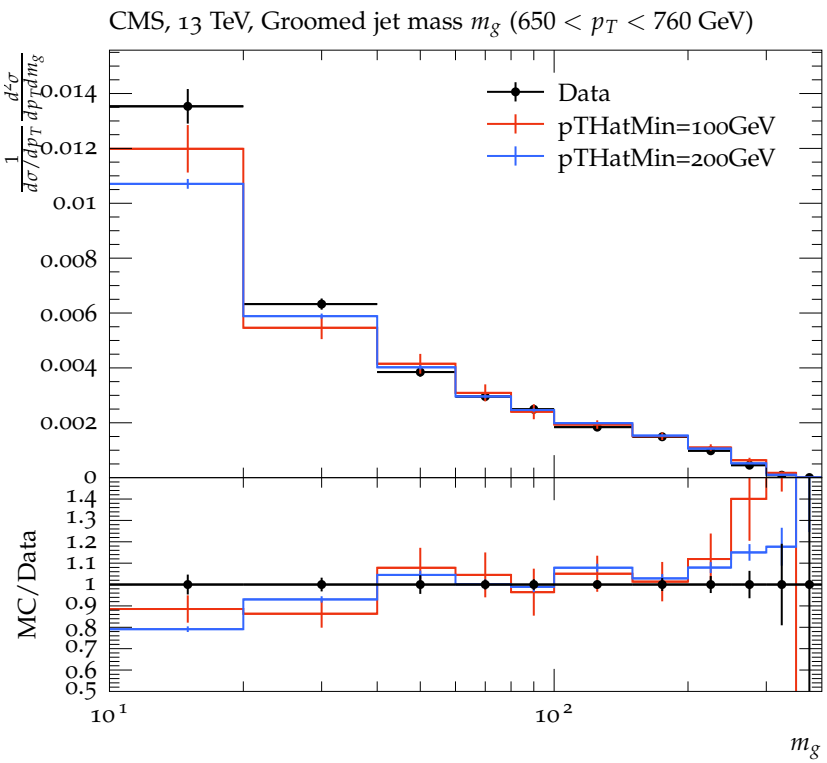
- HardQCD=on $\rightarrow$ 2 jets produced in hard interaction
- Tune and hard limits based on CMS analysis (CMS_2018_I1682495) arXiv:[1807.05974]: CUETP8M1 tune
- Centre of mass energy: 13 TeV
- Jet grooming with the "soft-drop" algorithm

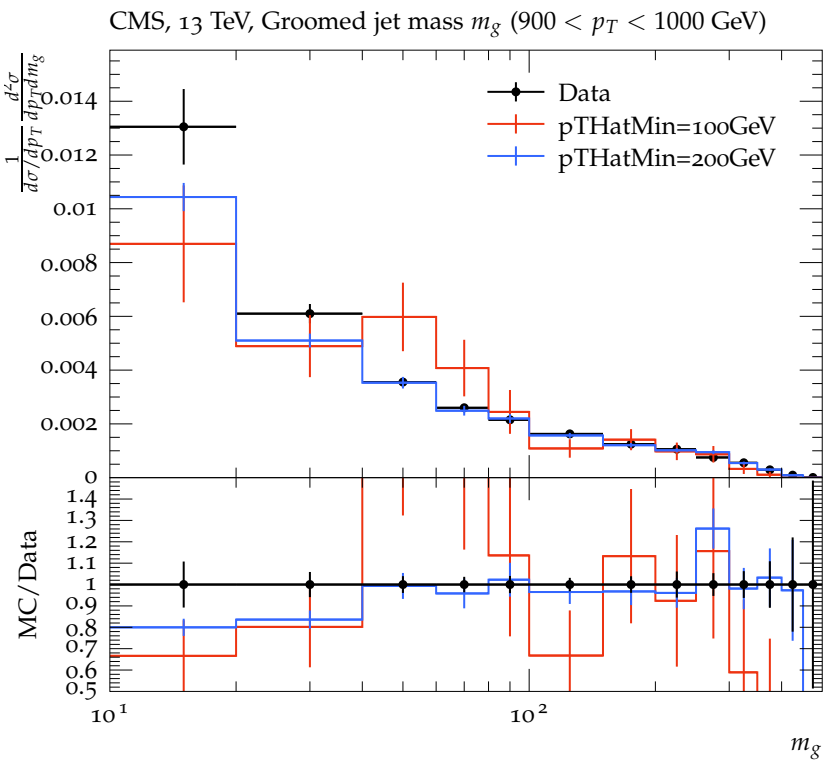
Tested and solved:

- Dependence on lower cut of heavy boson transverse momentum *pTHatMin*
- Test effect from: ISR - FSR - Hadronization - MPI
- Issue: normalization for ungroomed jets with Rivet2.7.0 plugin is wrong
- Difference of Rivet versions

pTHatMin dependence

This only influences the precision / statistics.

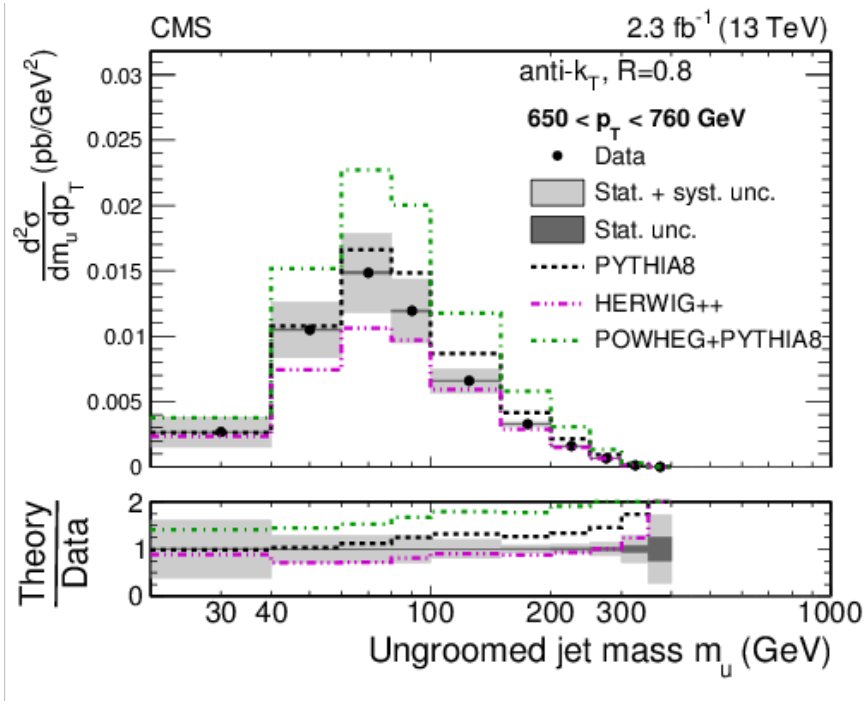


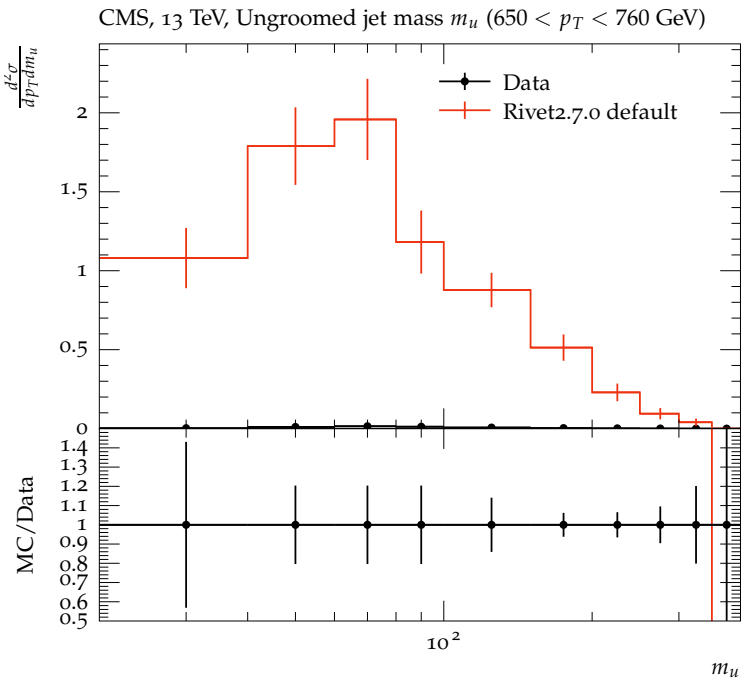


We use *pTHatMin* = 200 GeV, as is done in the paper.

Normalization issue: Rivet2.7.0 - Ungroomed

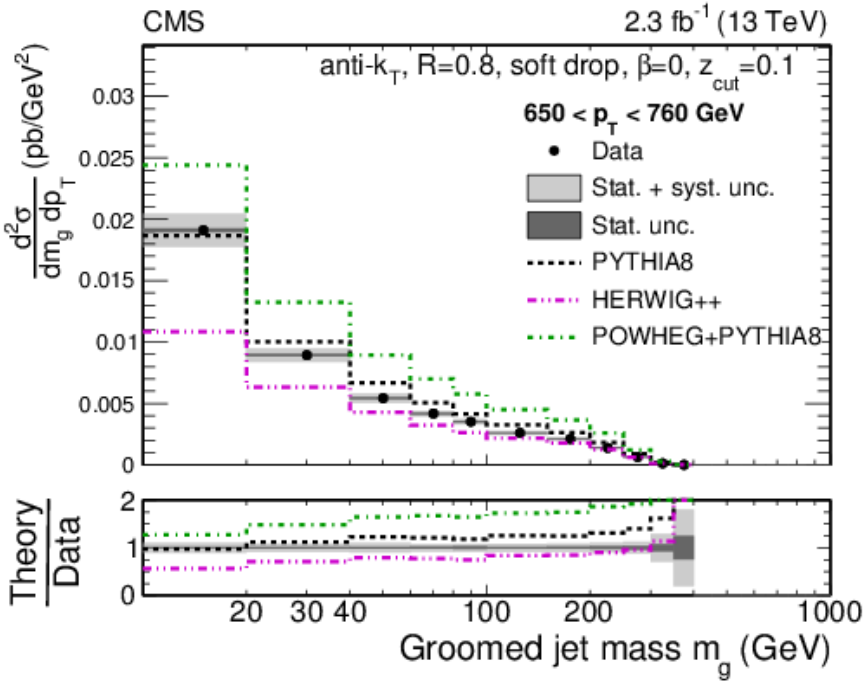
Plots with **absolute cross sections** were not normalized correctly in the default code of Rivet2.7.0. It can be seen by the large overshoot for data.

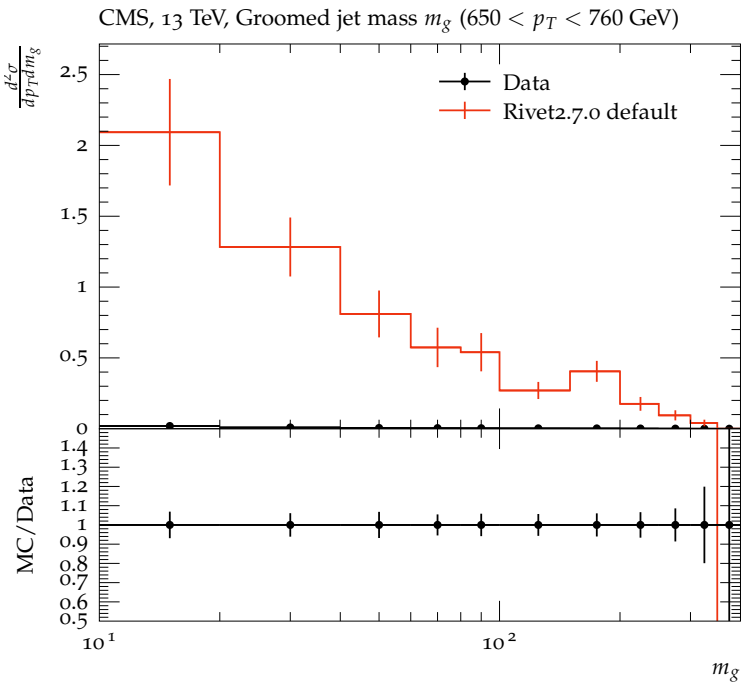




Normalization issue: Rivet2.7.0 - Groomed

Here the same overshoot of data occurs. Now the jet mass of groomed jets is shown.





Code difference Rivet2.7.0 - Rivet3.1.2

The origin of this incorrect normalization has been found in the analysis code. Differences between the Rivet2.7 plugin and the Rivet3.1 plugin are:

- the events for absolute cross section were weighted with an event weight
- the bins of absolute cross section were not divided by the binwidth

Rivet3.1.2

```
    // Find the appropriate pT bins and fill the histogram
    const size_t njetBin0 = findPtBin(j0.pt()/GeV);
    const size_t njetBin1 = findPtBin(j1.pt()/GeV);
    if (njetBin0 < N_PT_BINS_dj && njetBin1 < N_PT_BINS_dj) {
      for ( size_t jbin = 0; jbin < N_CATEGORIES; jbin++ ){
        _h_ungroomedJetMass_dj[njetBin0][jbin]->fill(j0.m()/GeV);
        _h_ungroomedJetMass_dj[njetBin1][jbin]->fill(j1.m()/GeV);
      }
    }

    // Now run the substructure algs...
    fastjet::PseudoJet sd0 = _softdrop(j0);
    fastjet::PseudoJet sd1 = _softdrop(j1);
    // ... and repeat
    if (njetBin0 < N_PT_BINS_dj && njetBin1 < N_PT_BINS_dj) {
      for ( size_t jbin = 0; jbin < N_CATEGORIES; jbin++ ){
        _h_sdJetMass_dj[njetBin0][jbin]->fill(sd0.m()/GeV);
        _h_sdJetMass_dj[njetBin1][jbin]->fill(sd1.m()/GeV);

      }
    }
  }


  /// Normalise histograms etc., after the run
+--  2 regels: void finalize() {
    for (size_t i = 0; i < N_PT_BINS_dj; ++i) {
      normalize(_h_ungroomedJetMass_dj[i][1]);
      normalize(_h_sdJetMass_dj[i][1]);
    }
    // Normalize the absolute cross section histograms to xs * lumi.
    for (size_t i = 0; i < N_PT_BINS_dj; ++i) {
      scale(_h_ungroomedJetMass_dj[i][0],   crossSection()/picobarn / sumOfWeights() / (ptBins_dj[i+1]-ptBins_dj[i]
      scale(_h_sdJetMass_dj[i][0],          crossSection()/picobarn / sumOfWeights() / (ptBins_dj[i+1]-ptBins_dj[i]

    }
  }
  //@}
```

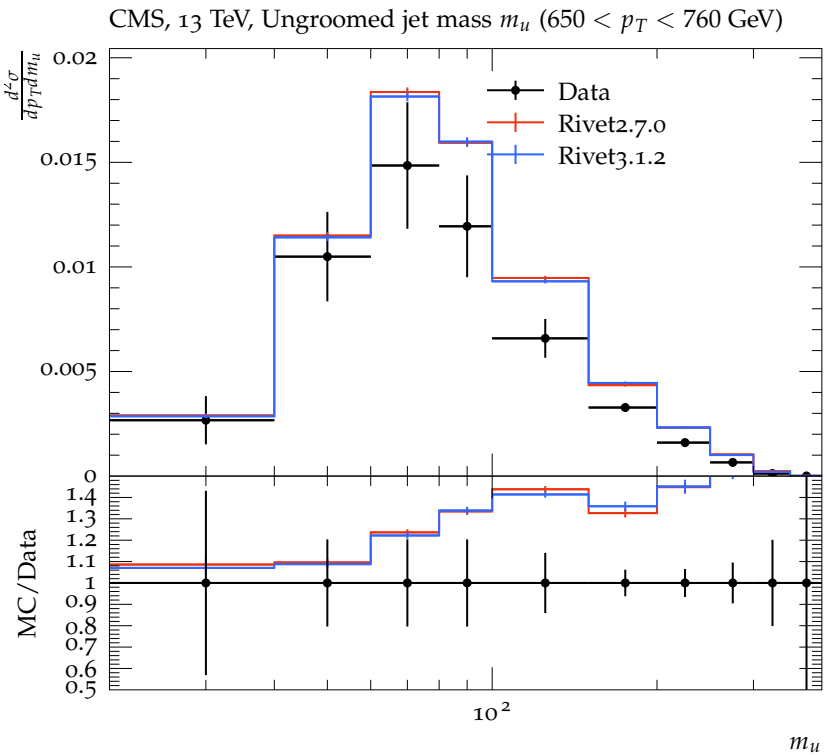
Rivet2.7.0

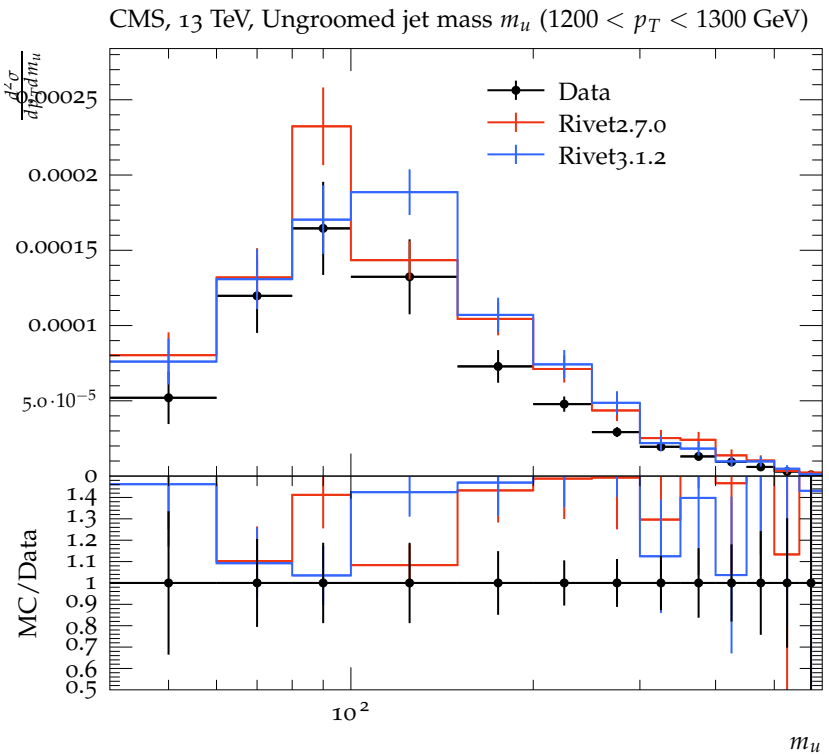
```
    // Find the appropriate pT bins and fill the histogram
    const size_t njetBin0 = findPtBin(j0.pt()/GeV);
    const size_t njetBin1 = findPtBin(j1.pt()/GeV);
    if (njetBin0 < N_PT_BINS_dj && njetBin1 < N_PT_BINS_dj) {
      for ( size_t jbin = 0; jbin < N_CATEGORIES; jbin++ ){
        //_h_ungroomedJetMass_dj[njetBin0][jbin]->fill(j0.m()/GeV);
        //_h_ungroomedJetMass_dj[njetBin1][jbin]->fill(j1.m()/GeV);
        _h_ungroomedJetMass_dj[njetBin0][jbin]->fill(j0.m()/GeV, weight);
        _h_ungroomedJetMass_dj[njetBin1][jbin]->fill(j1.m()/GeV, weight);
      }
    }

    // Now run the substructure algs...
    fastjet::PseudoJet sd0 = _softdrop(j0);
    fastjet::PseudoJet sd1 = _softdrop(j1);
    // ... and repeat
    if (njetBin0 < N_PT_BINS_dj && njetBin1 < N_PT_BINS_dj) {
      for ( size_t jbin = 0; jbin < N_CATEGORIES; jbin++ ){
        //_h_sdJetMass_dj[njetBin0][jbin]->fill(sd0.m()/GeV);
        //_h_sdJetMass_dj[njetBin1][jbin]->fill(sd1.m()/GeV);
        _h_sdJetMass_dj[njetBin0][jbin]->fill(sd0.m()/GeV, weight);
        _h_sdJetMass_dj[njetBin1][jbin]->fill(sd1.m()/GeV, weight);
      }
    }
  }


  /// Normalise histograms etc., after the run
+--  2 regels: void finalize() {
    for (size_t i = 0; i < N_PT_BINS_dj; ++i) {
      normalize(_h_ungroomedJetMass_dj[i][1]);
      normalize(_h_sdJetMass_dj[i][1]);
    }
    // Normalize the absolute cross section histograms to xs * lumi.
    for (size_t i = 0; i < N_PT_BINS_dj; ++i) {
      //  scale(_h_ungroomedJetMass_dj[i][0],   crossSection()/picobarn / sumOfWeights() / (ptBins_dj[i+1]-ptBins_dj
      //  scale(_h_sdJetMass_dj[i][0],          crossSection()/picobarn / sumOfWeights() / (ptBins_dj[i+1]-ptBins_dj
      scale(_h_ungroomedJetMass_dj[i][0],   crossSection()/picobarn / sumOfWeights());
      scale(_h_sdJetMass_dj[i][0],          crossSection()/picobarn / sumOfWeights());
    }
  }
  //@}
```

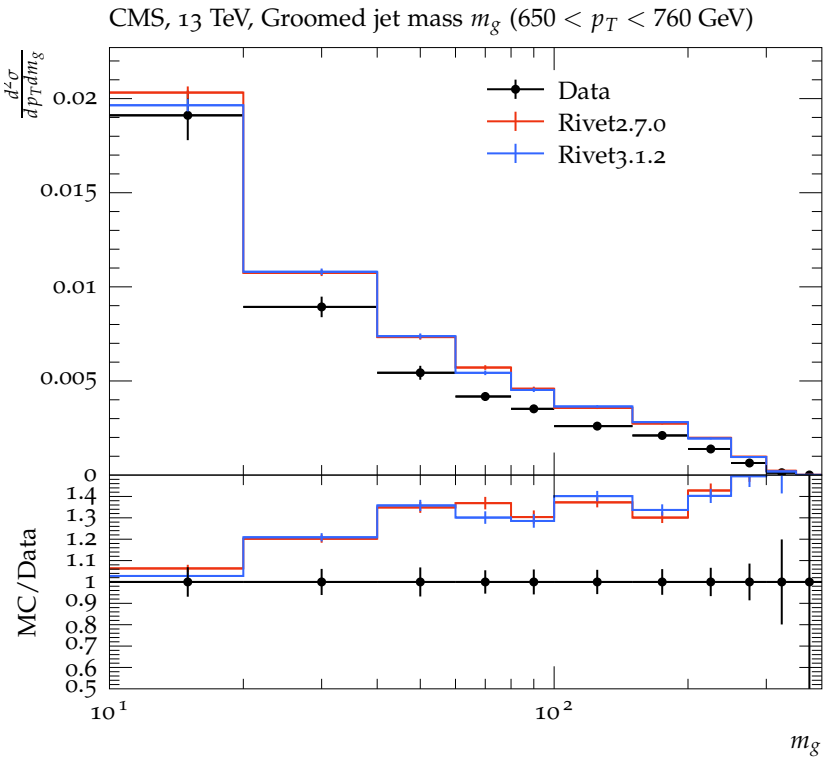
With changes in the .cc file in Rivet2.7.0, the normalization issue is solved. Also using a newer Rivet version solves the problem with absolute cross sections:

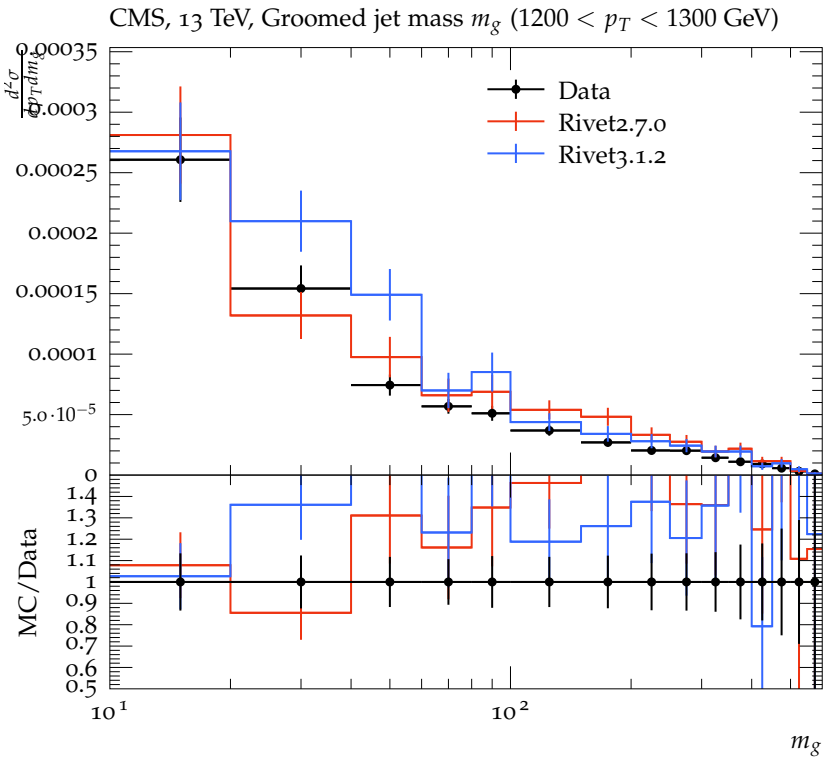




Rivet2.7.0 - Rivet3.1.2: Groomed

With changes in the .cc file in Rivet2.7.0, the normalization issue is solved. Also using a newer Rivet version solves the problem with absolute cross sections:

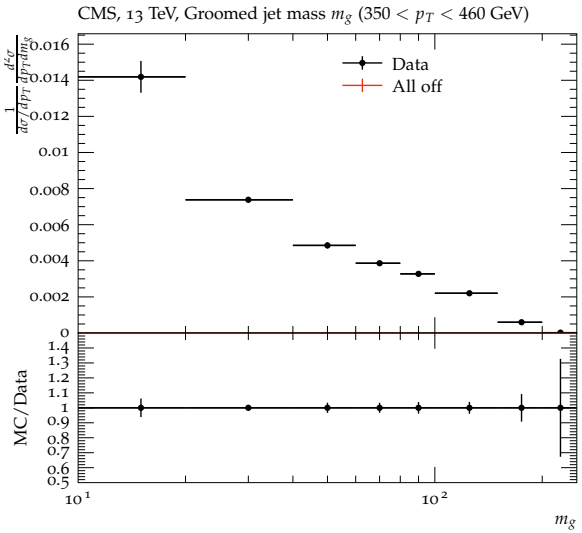


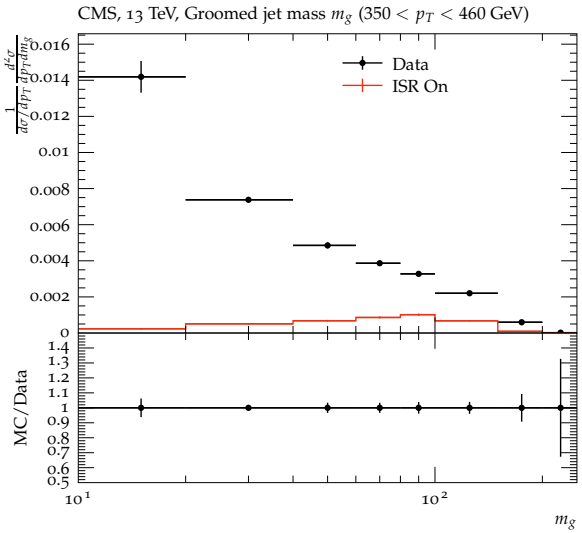


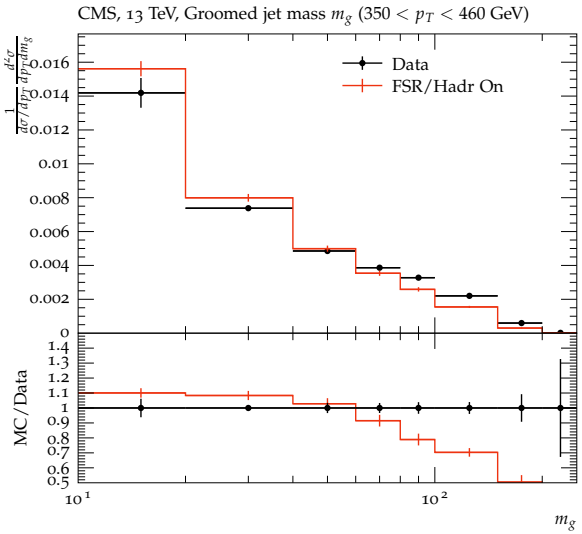
Check influence of ISR - FSR - Hadronization and MPI

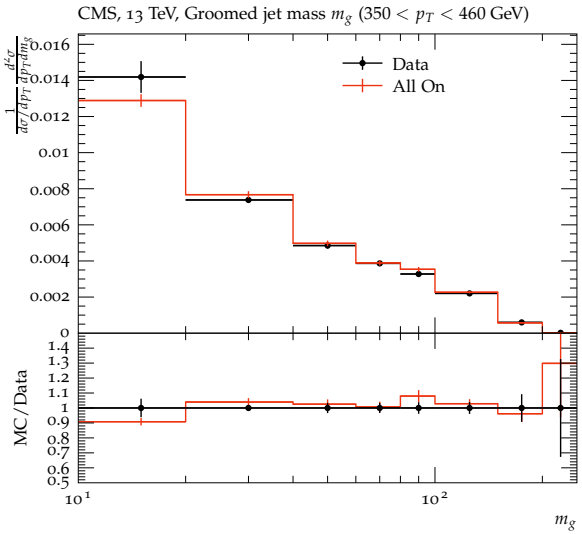
Add showers one by one ($350 < p_T < 460$ GeV)

The influence of parton showers, hadronization models and multi-parton interaction is investigated.



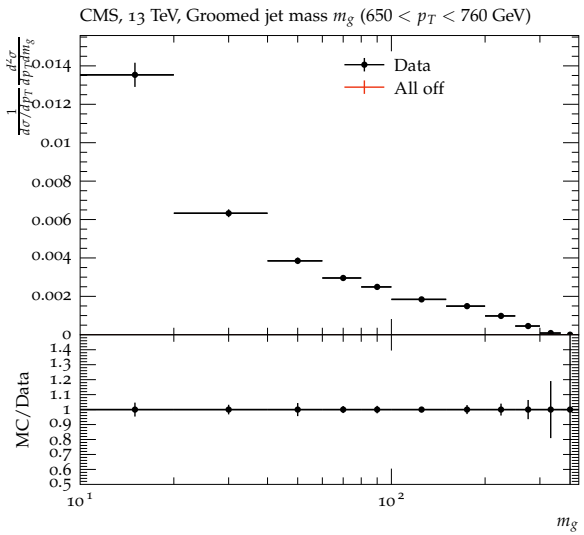


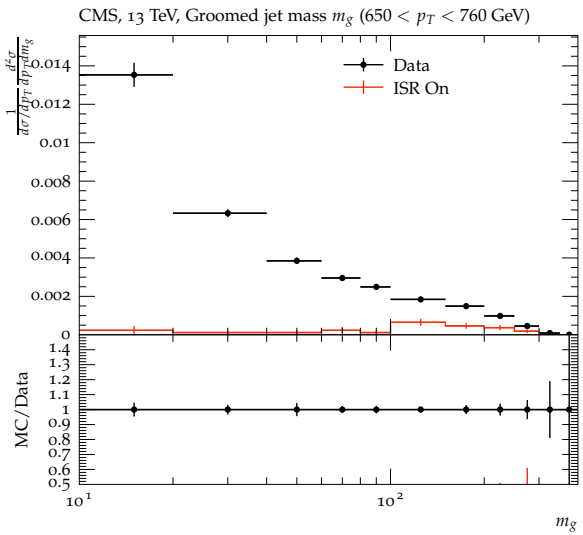


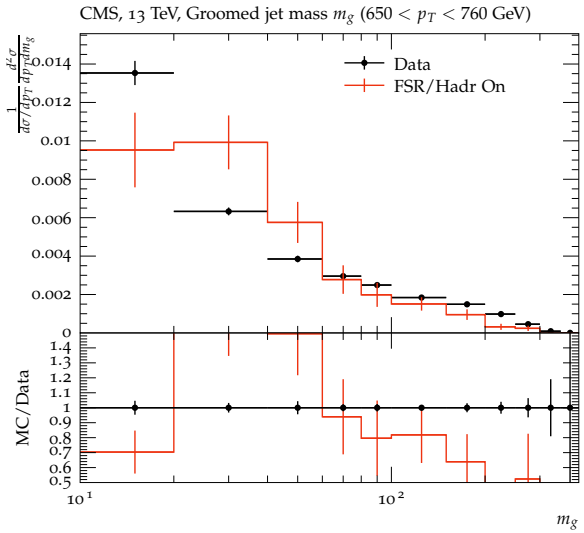


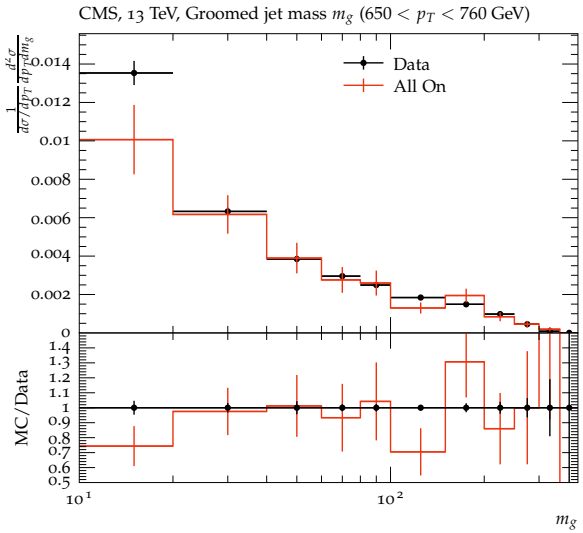
Add showers one by one ($650 < p_T < 760$ GeV)

We conclude that the final state radiation is of most influence on the results.



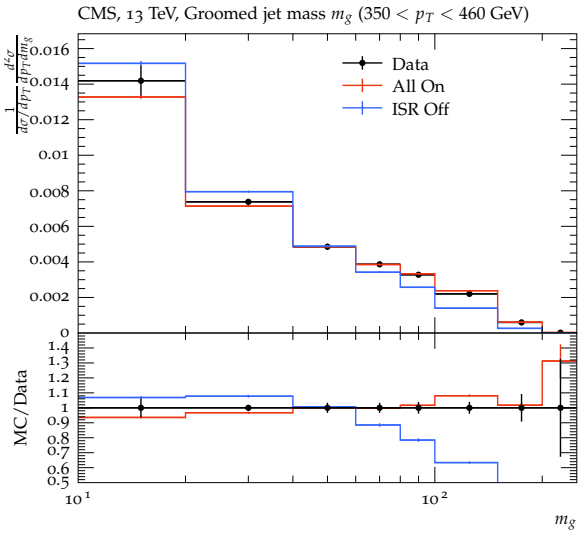


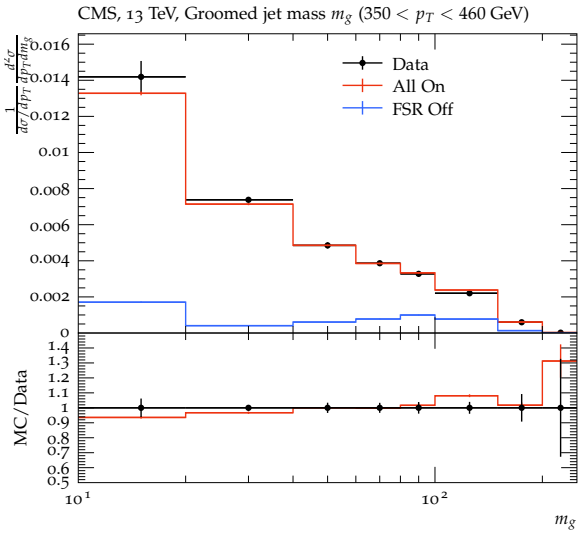


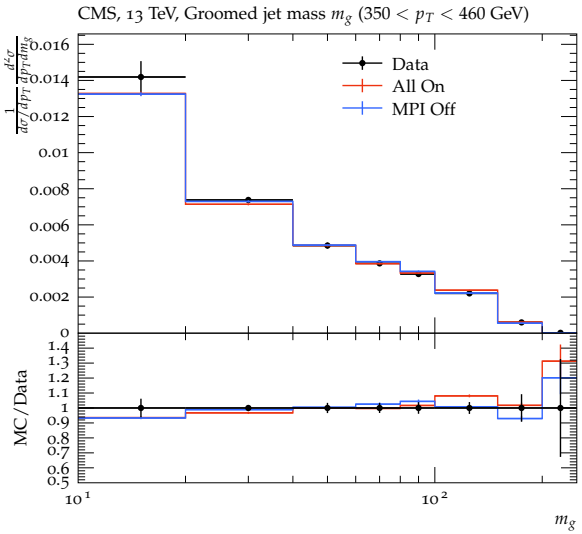


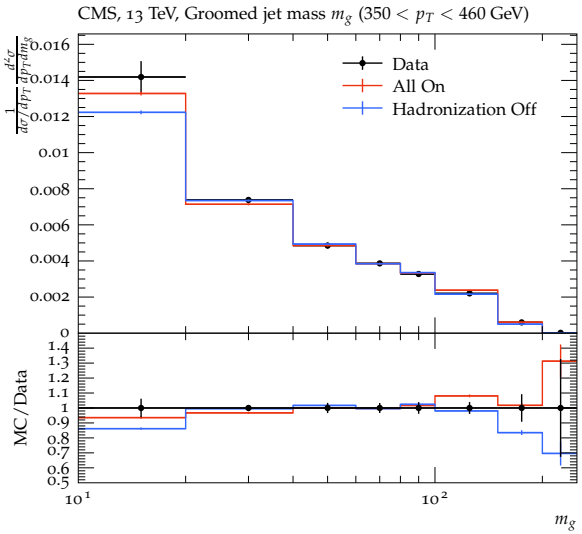
Turn off one shower ($350 < p_T < 460$ GeV)

The influence of parton showers, hadronization models and multi-parton interaction is investigated.



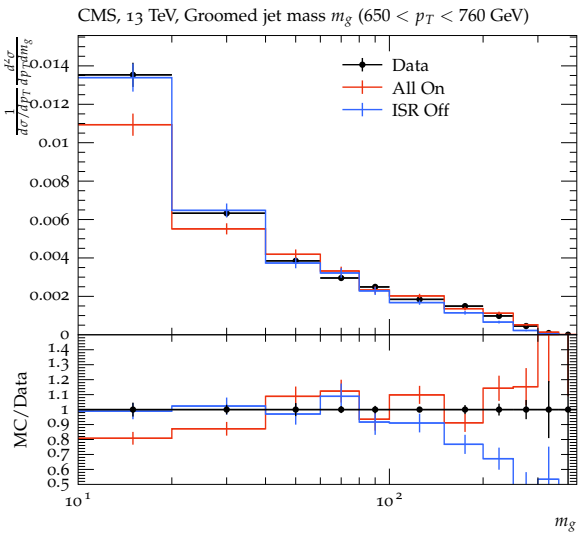


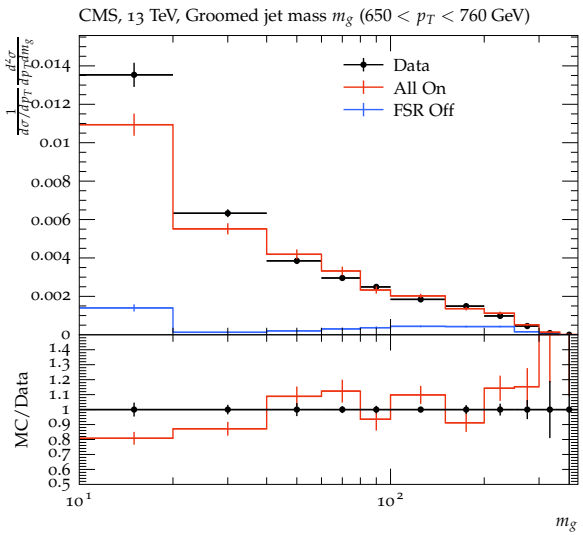


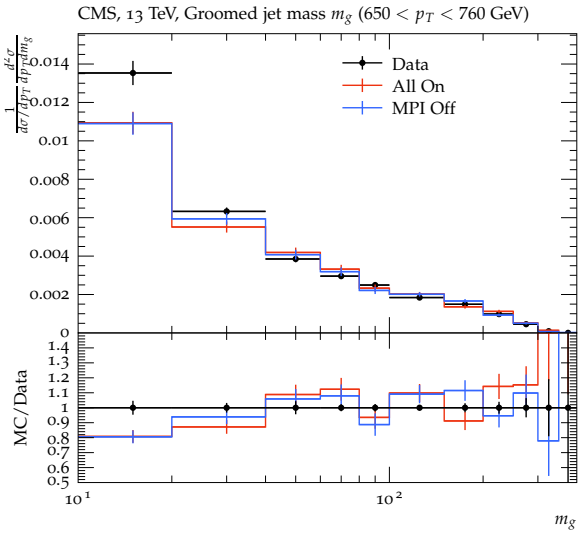


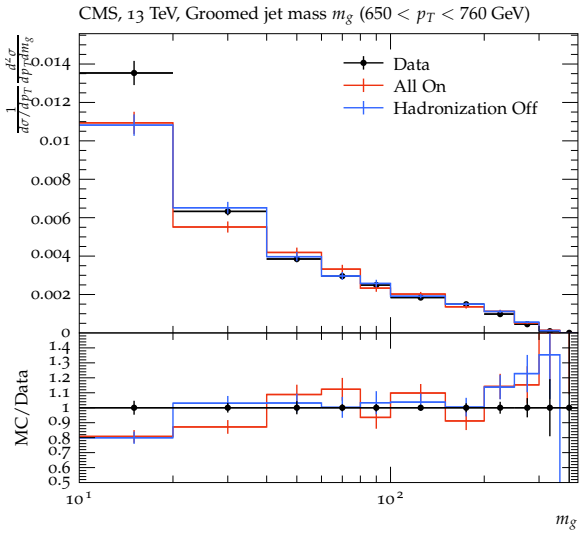
Turn off one shower ($650 < p_T < 760$ GeV)

We conclude that the final state radiation is of most influence on the results.









Final results for jet mass in dijet events with PYTHIA8.2.44

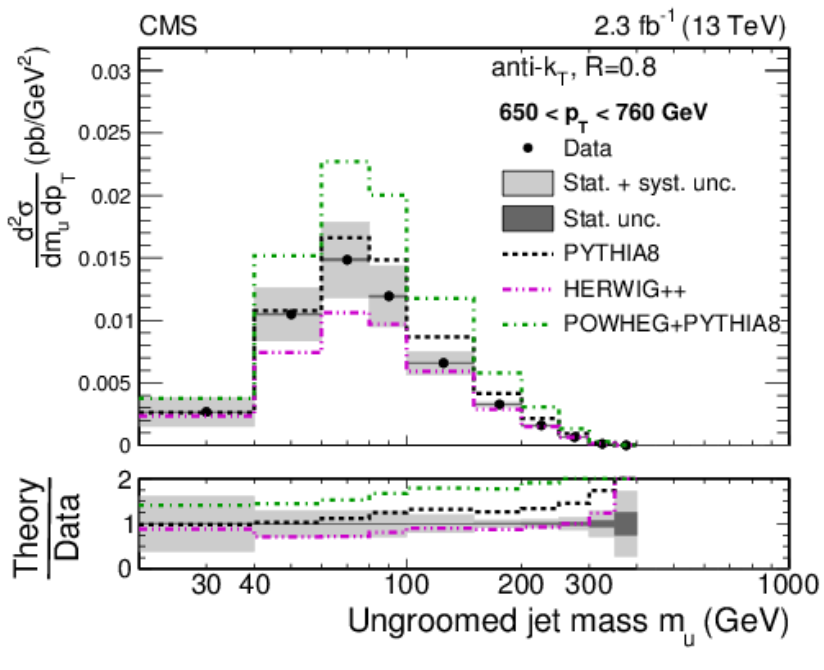
Final predictions with PYTHIA8

- #events = 10,000,000
- Rivet3.1.2
- Plugin: CMS_2018_I1682495
- pTHatMin = 200 GeV
- with ISR, FSR, MPI and Hadronization
- CUETP8M1 tune

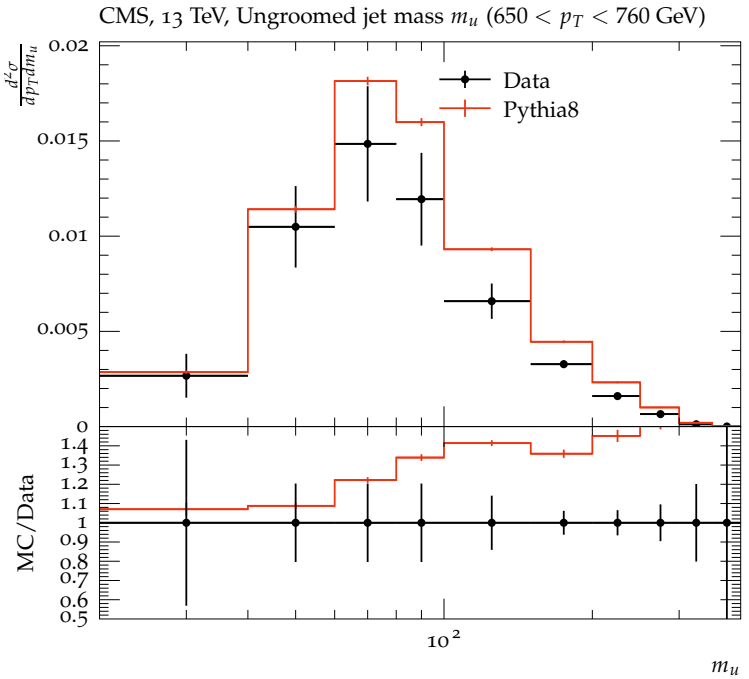
Comparison with CMS paper CMS_2018_I1682495

Absolute cross section
$650 < p_T < 760$ GeV
Ungroomed

Paper

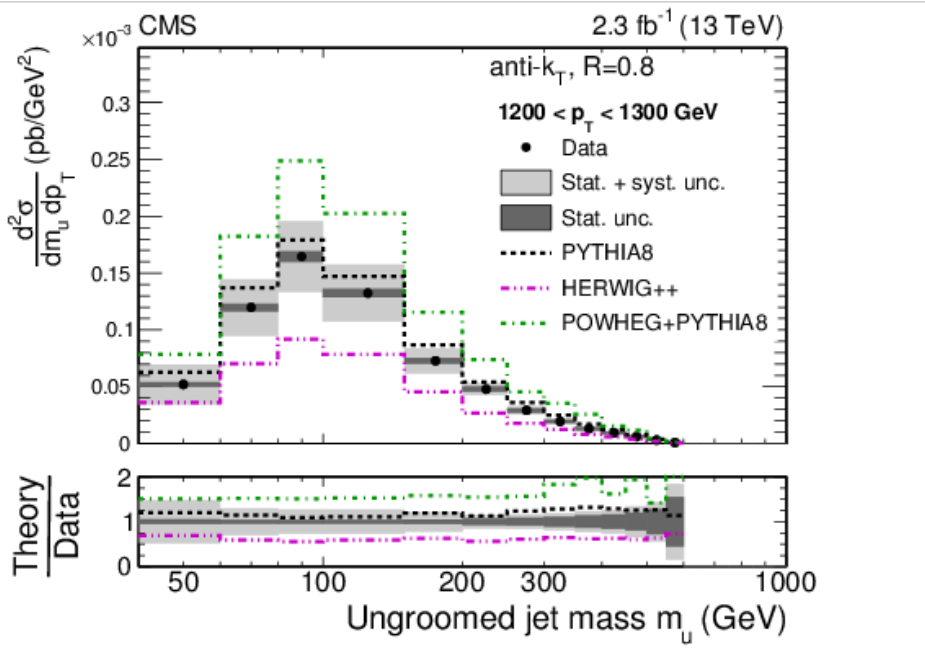


PYTHIA8

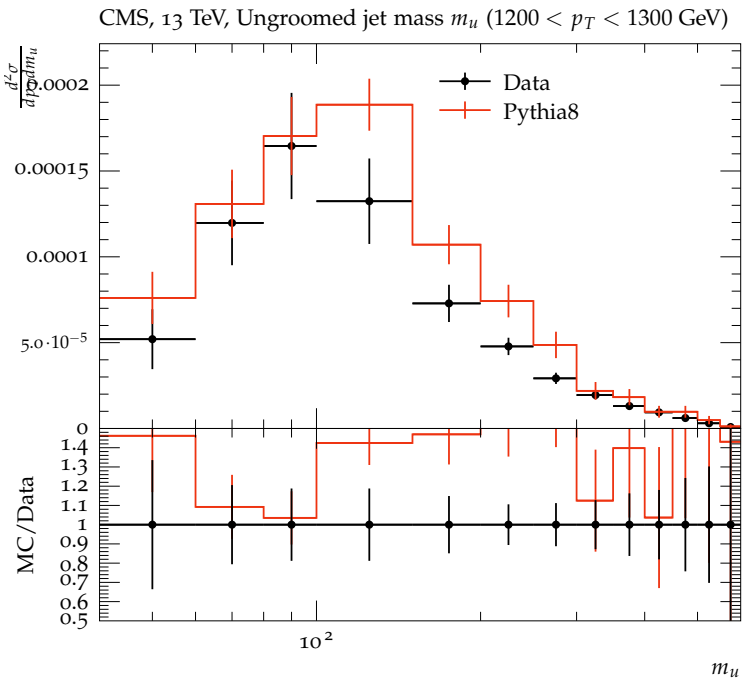


$1200 < p_T < 1300$ GeV
Ungroomed

Paper

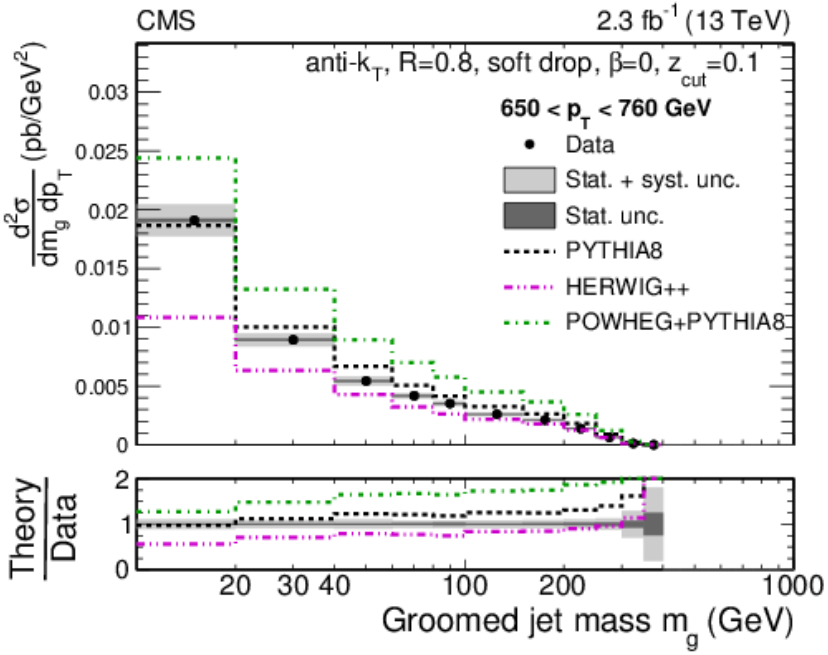


PYTHIA8

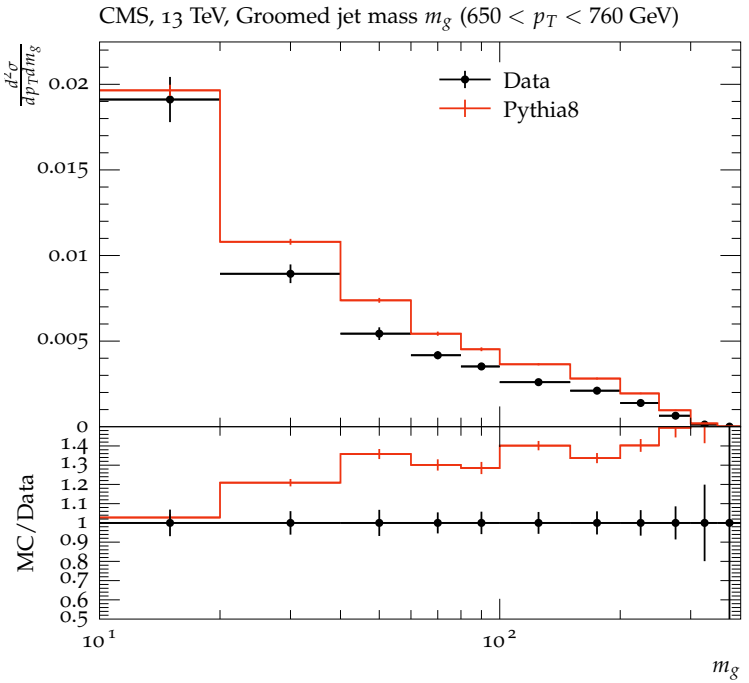


$650 < p_T < 760$ GeV
Groomed

Paper

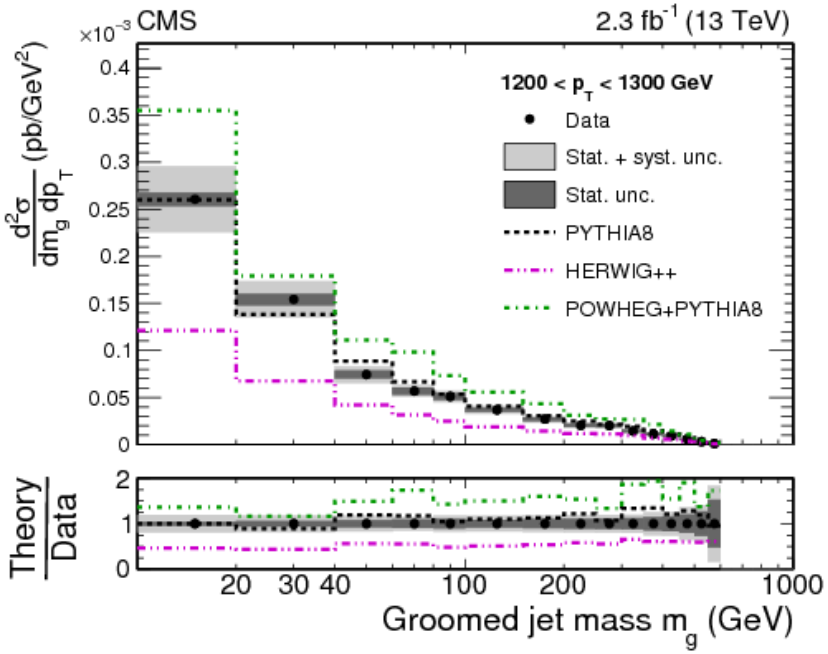


PYTHIA8

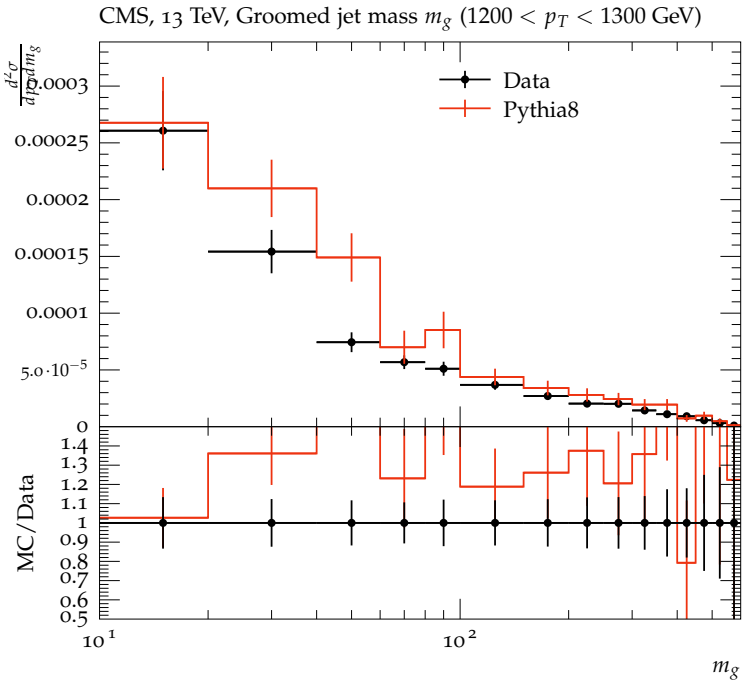


$1200 < p_T < 1300$ GeV
Groomed

Paper

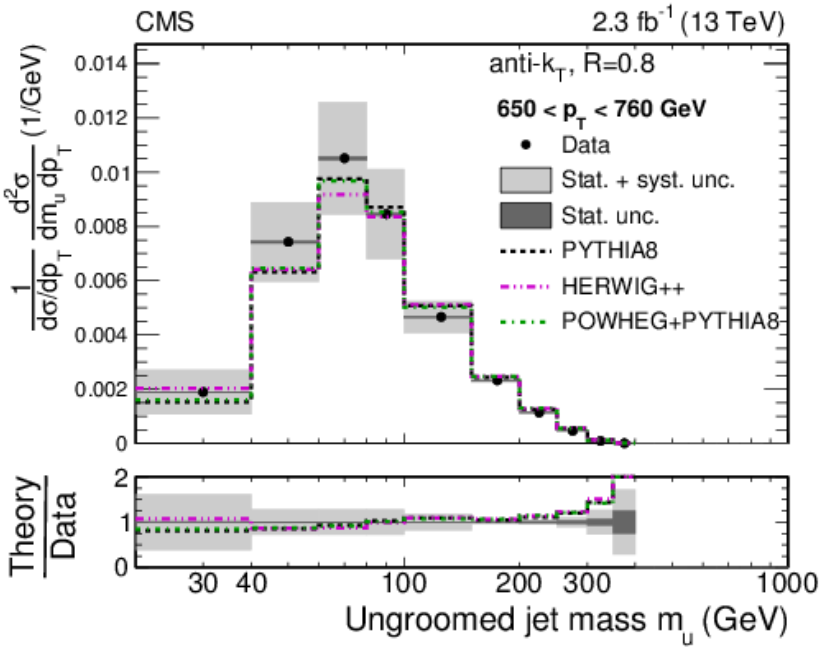


PYTHIA8

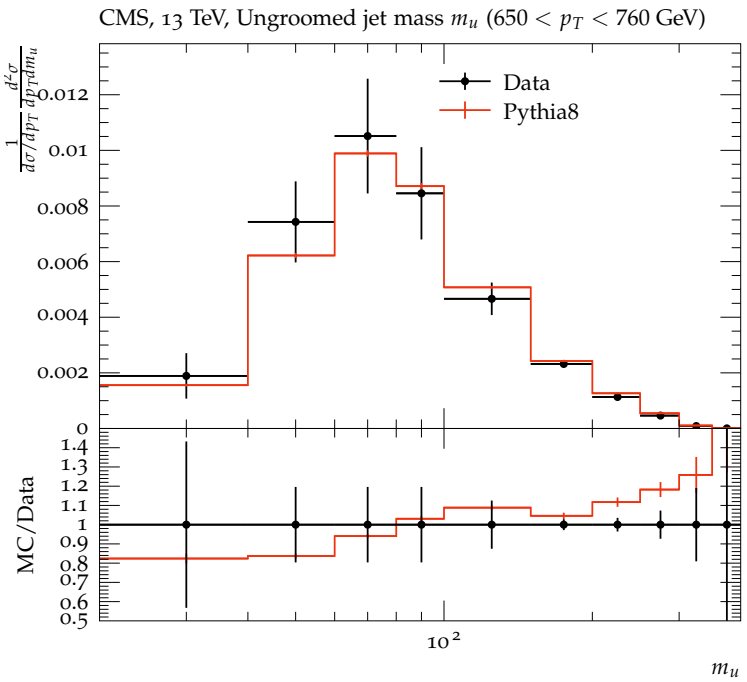


$650 < p_T < 760$ GeV
Ungroomed

Paper

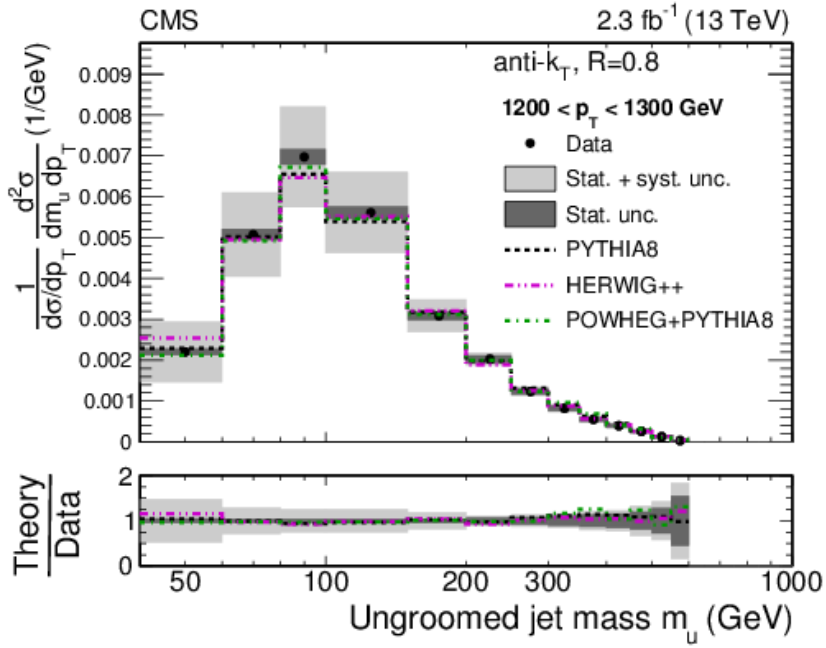


PYTHIA8

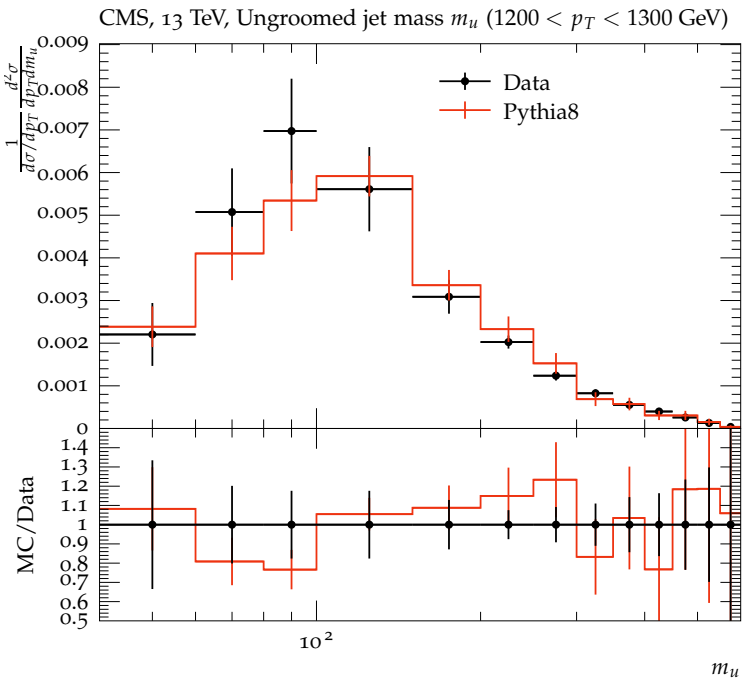


$1200 < p_T < 1300$ GeV
Ungroomed

Paper

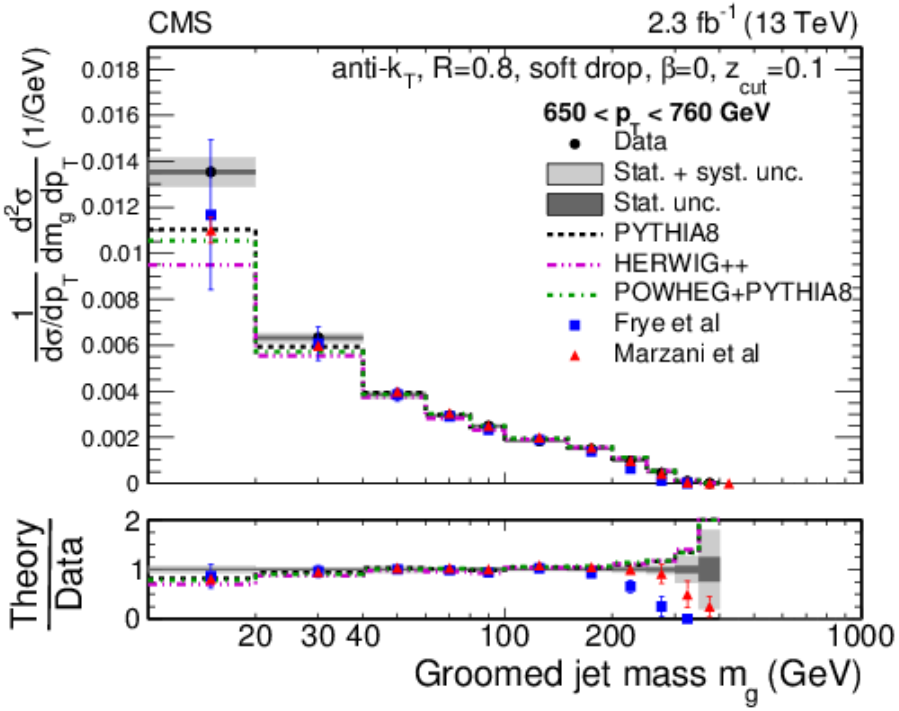


PYTHIA8

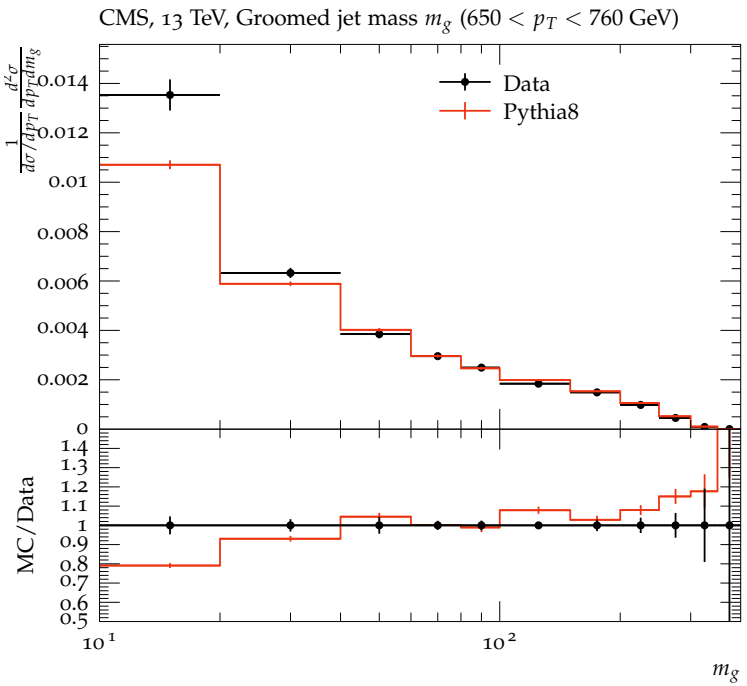


$650 < p_T < 760$ GeV
Groomed

Paper

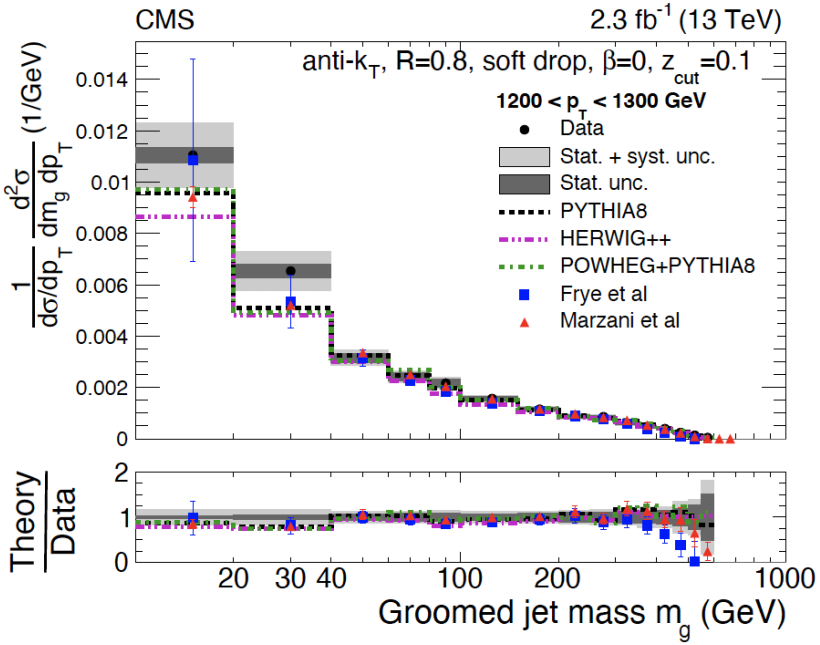


PYTHIA8

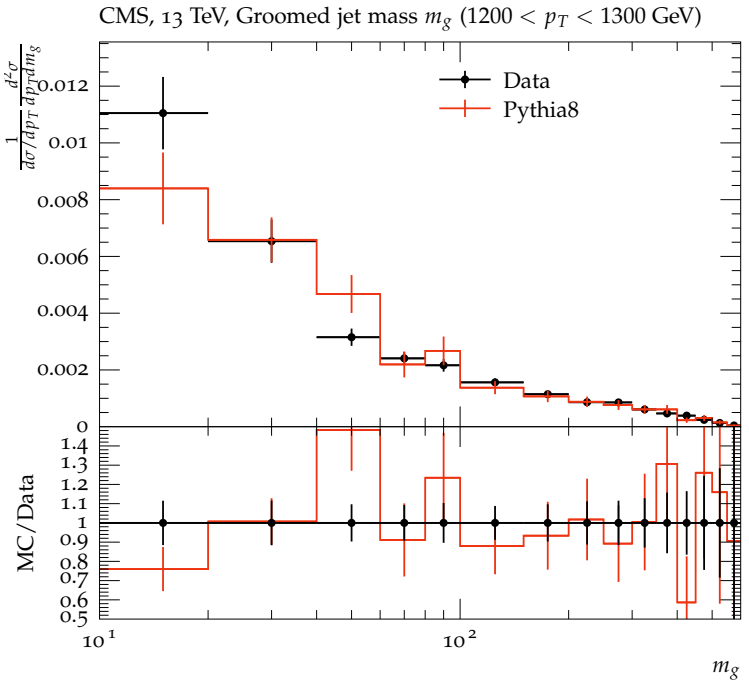


$1200 < p_T < 1300$ GeV
Groomed

Paper



PYTHIA8



Bins with deviations: by eye comparison

The predictions done with Pythia8 by CMS and by ourselves seem to differ in some mass bins. This comparison is done by eye.

Bins that deviate from CMS results

Plot	#bin with deviation
Absolute, ungroomed, 650 < pT < 760 GeV	-
Absolute, ungroomed, 1200 < pT < 1300 GeV	4, 6
Absolute, groomed, 650 < pT < 760 GeV	2
Absolute, groomed, 1200 < pT < 1300 GeV	2, 3, 5
Normalized, ungroomed, 650 < pT < 760 GeV	-
Normalized, ungroomed, 1200 < pT < 1300 GeV	2, 3
Normalized, groomed, 650 < pT < 760 GeV	-
Normalized, groomed, 1200 < pT < 1300 GeV	2, 3