nTuples and applications

Daniel Maître

with Gudrun Heinrich and Mark Johnson

IPPP Durham

Outline

- nTuples
  - NLO
  - NNLO
- Strong coupling constant from high multiplicity processes

n-Tuple files [arXiv:1310.7439]

- High multiplicity NLO calculations are computationally intensive
- It would not be possible to rerun a high multiplicity W+jet calculation every time a new interesting observable comes up
- Matrix elements are expensive, while
  - Jet clustering
  - Observables
  - PDF evaluation

  are relatively cheap
- Store each matrix element, PS point and the information necessary to change the factorisation and renormalisation scales in large files we call n-Tuple files
- We use ROOT file as storage

NLO nTuple files [arXiv:1310.7439]

- Advantages
  - One can change the analysis cuts, add observables
  - Cheap scale variation and PDF errors (otherwise extremely expensive)
  - Easy communication between theorists and experimenters
  - No need for specific know-how of the tool which produced the NLO calculation
  - Easier to “endorse” an event file than a program
- Disadvantage
  - Large files
  - Generation cuts need to be loose enough to accommodate many analysis → efficiency cost

nTupleReader library [arXiv:1310.7439]

- We provide a C++ library to facilitate the use of the n-Tuple files
- Allows:
  - Change of factorisation and renormalisation scales
  - Change of pdf (from LHAPDF set), including error sets
- Has a Python interface
- Template for a customised implementation
- Available on hepforge

  https://blackhat.hepforge.org/trac/wiki/BlackHatSherpaNtuples

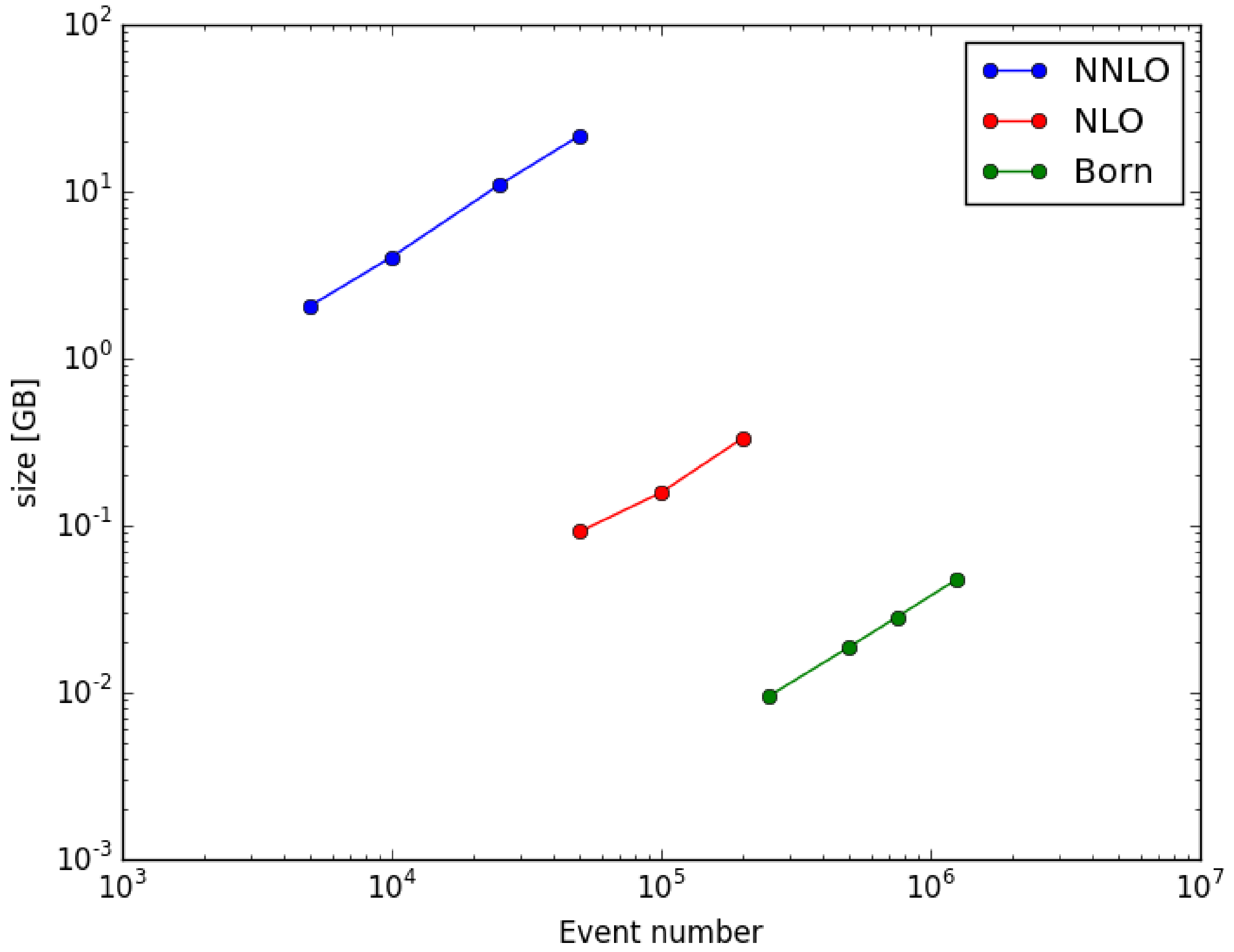
nTuples for NNLO

- nTuples have proven useful for NLO
- Can they be as useful for NNLO?
- Same advantages and same disadvantages but amplified:
  - Programs are more complex
  - Larger files:
    - Many more pieces in the calculation
    - More logarithm coefficients
- Main question: is the size reasonable?

Experiment in Les Houches

- Used EErad3 [Gehrmann-De Ridder, Gehrmann, Glover, Heinrich; arXiv:0710.0346] to generate NNLO ntuples for $e^+e^- \rightarrow$ 3 jets
- Storage structure very similar to NLO nTuples
- Easier than hadronic :
  - No initial state information
  - Strong coupling dependence trivial (no real need to store coefficients of logarithms)

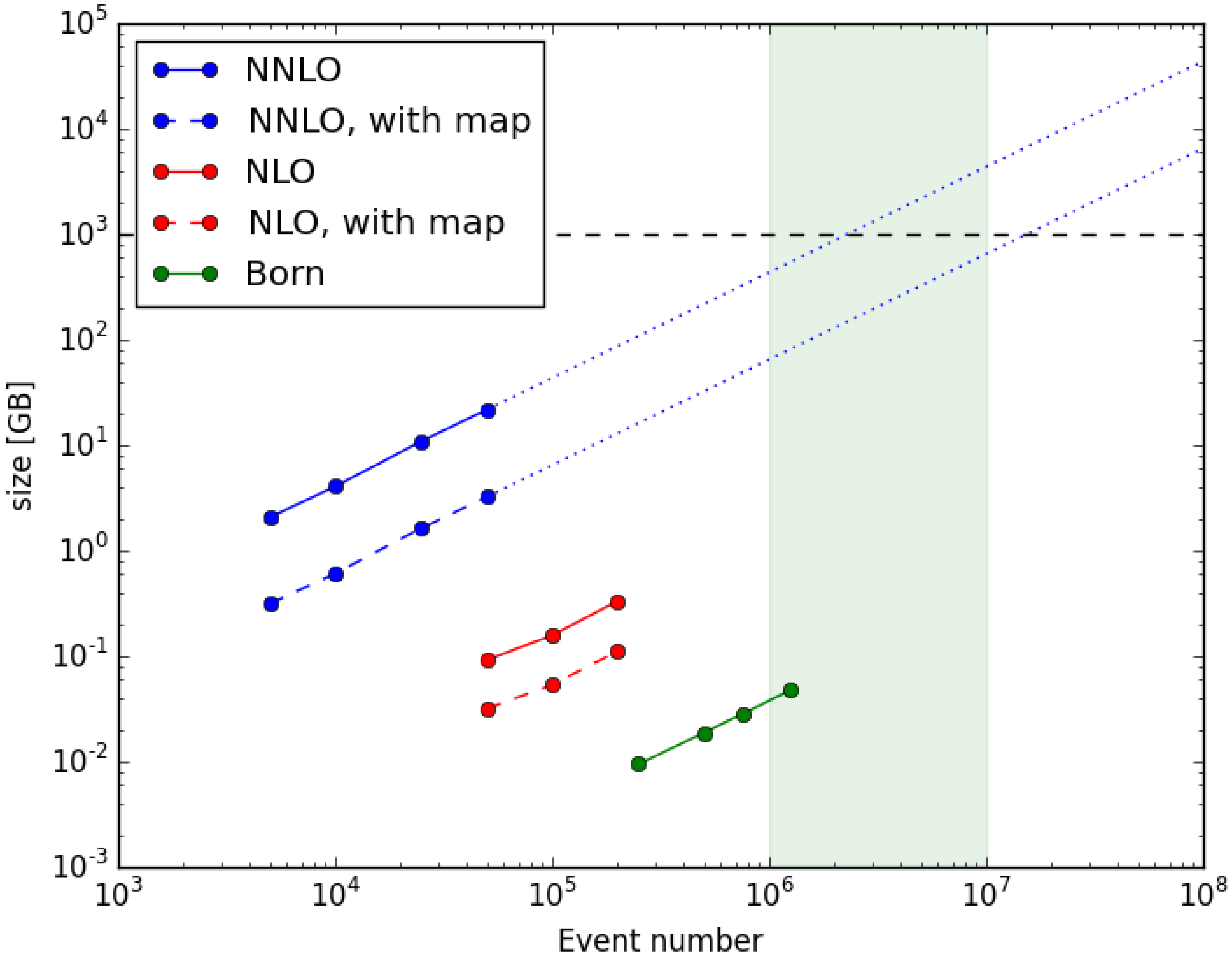
Size vs Event number



Using mapping information

- The most space-consuming part is the double real part
  - More final state momenta
  - Need much statistics because of subtraction terms
- For each real-real phase-space point we have many subtraction terms
- Each of them has a different set of momenta given by a $(n+2) \rightarrow n$ or $(n+1) \rightarrow n$ map
- We can save much space if we simply record the mapping that was used instead of the momenta
- The downside is that
  - there is more calculation at the moment of reading the nTuple
  - More coupling between nTuple file and code that produced it

Extrapolated file size



File sizes

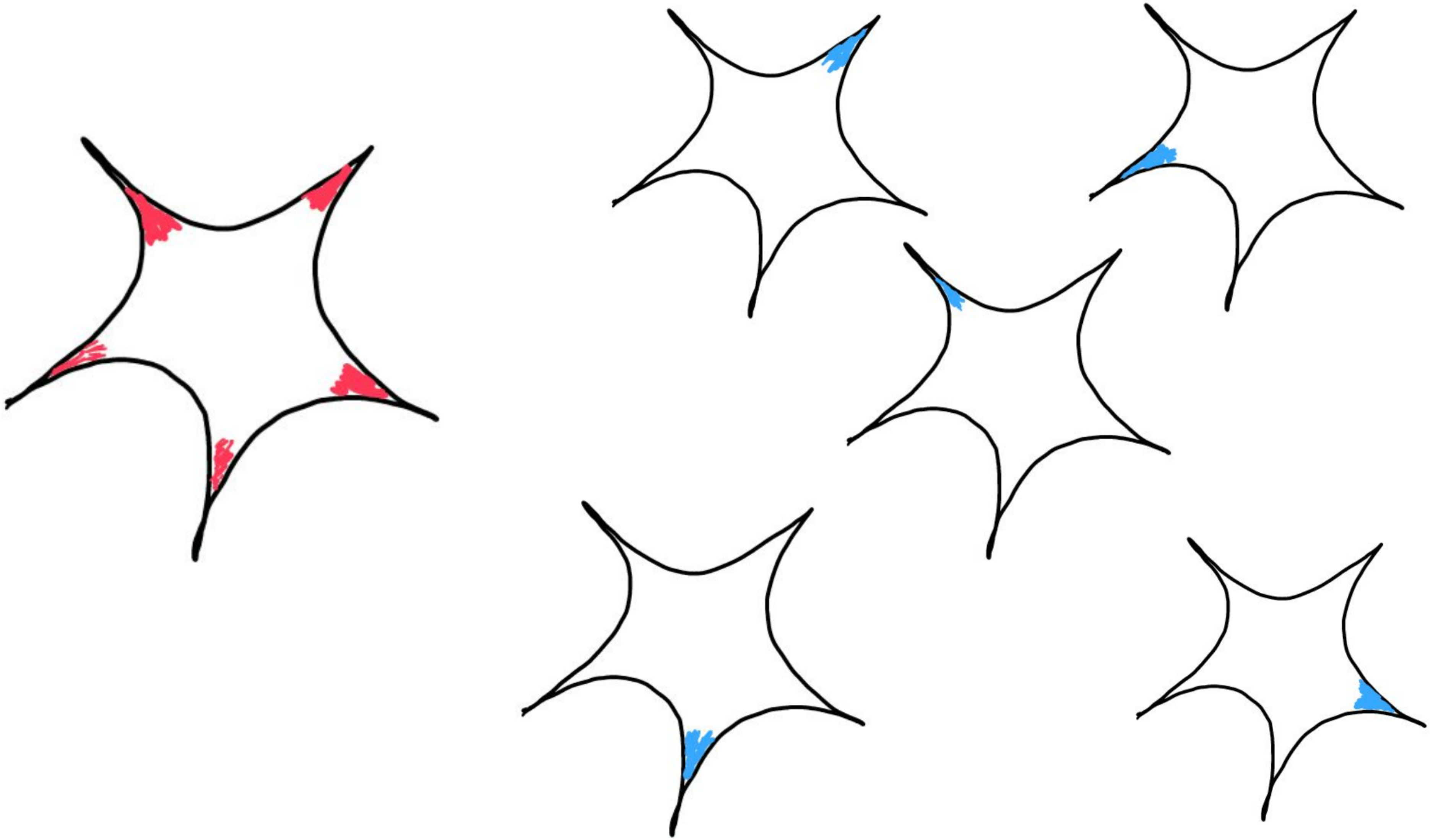
- EErad3 “true” NNLO calculation with NNLO subtraction
- Processes calculated with the q_T subtraction or n-jettiness methods look more like two NLO calculations → could be more suitable for nTuple production
- Current programs are optimized for CPU usage, the trad-offs between “unweighting” and run time can be different if one wants to optimise the storage size

Hadronic initial states

- Work in progress to extend NNLO nTuples to hadronic initial states
- Working prototype (I can reproduce weights) for DIS setting
- Currently working on hadron-hadron

More effective storage

- Real phase space has many divergent regions
- Each is made finite by subtraction terms



More effective storage

- The cancellation between real matrix element and subtraction in the divergent part of the phase space requires a lot of statistics
- Much less is needed where the subtraction term is finite
- At NLO subtraction terms are routinely cut off where they are not needed.
- The subtraction term is integrated analytically as a function of the cut-off
- It is much harder to do at NNLO

More effective storage

- To optimise the storage one can separate the integration in two parts, one where the cancellations happen and one free of cancellations
- The former requires large statistics but the latter does not
- No analytical integration needed, just to run the program once above and once below the cut-off, with different statistics

NNLO nTuple files

- Trade offs

  - How intrusive

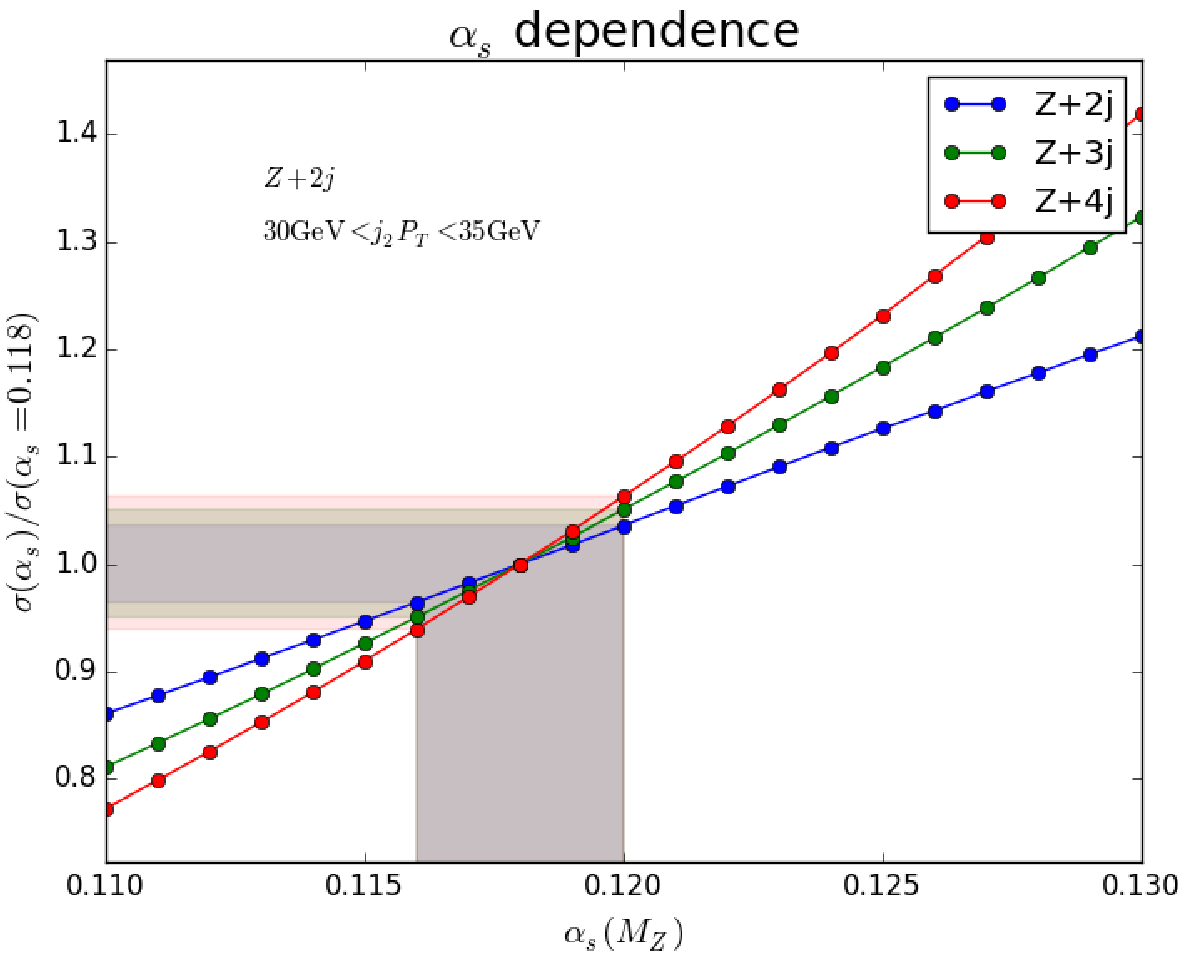
    (almost) no modification | introduce cut-off | reorganise PS integration

  - How much calculation at read time/coupling to original code

    no use of mapping information | use mapping information

Strong coupling determination

- Strong coupling determination are usually done using low multiplicity processes
  - High statistics
  - Well studied
  - Can have NNLO predictions
- High multiplicity processes have
  - A steeper coupling constant dependence
  - Less statistics



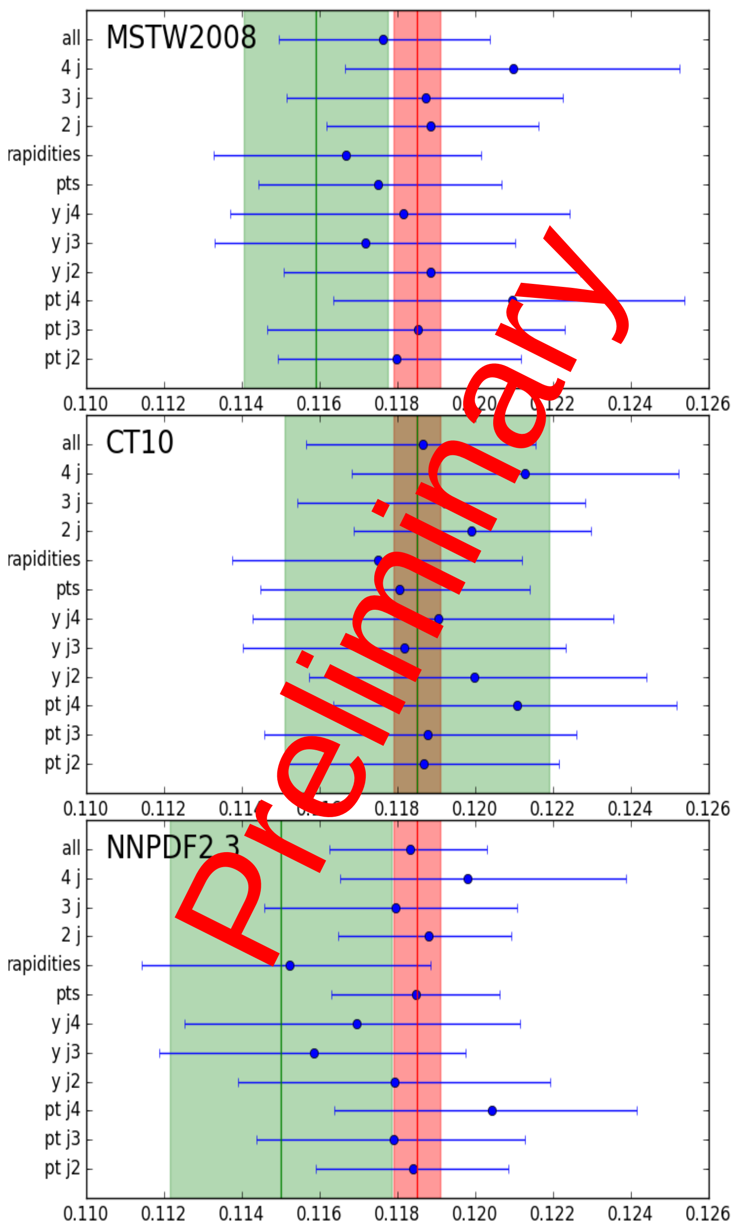
Strong coupling determination

- Here we look at the coupling constant dependence in Z+2,3,4 jets
- Compare BlackHat+Sherpa NLO results with Atlas measurement at 7 TeV [arXiv:1304.7098]
- To do this one needs to calculate the NLO theory prediction for several values of the coupling constant, for several pdfs and for all pdf sets to estimate the pdf uncertainty

  ~ 30’000 histograms!
- It is too much even with nTuples
- We used the BH+Sherpa public nTuples to create fastNLO tables [Britzger, Rabbertz, Sieber, Stober, Wobisch; arXiv:1208.3641]

Strong coupling determination

- The high multiplicity processes are not quite competitive
- High multiplicity processes are a good place to investigate scale setting issues between predictions and pdfs
- No scale variation



Red: World average
Green: CMS inclusive jets [arXiv:1410.6765]

FastNLO vs nTuple

- FastNLO table much more efficient
- Less flexibility
  - Fixed distributions
  - Limited number of scales
- Missing statistical error+correlation
- I don’t know how to estimate the parametrisation error

Conclusion

- nTuples could be a viable option for NNLO
  - Storage can be made more efficient at the cost of read-time speed or changes to the original code
  - Can help disseminate NNLO results
- FastNLO tables can be more convenient for dedicated use