MCFM@NNLO

Downloadable from **mcfm.fnal.gov**

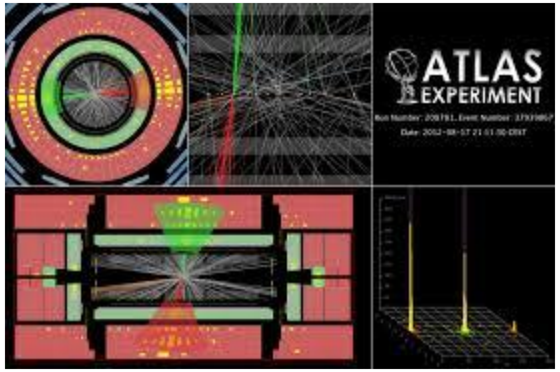
- A Multi-Threaded Version of MCFM, J.M. Campbell, R.K. Ellis, W. Giele, 2015
- Color singlet production at NNLO in MCFM, R. Boughezal, J.M. Campbell, R.K. Ellis, C. Focke, W. Giele, X. Liu, F. Petriello, C. Williams, 2016

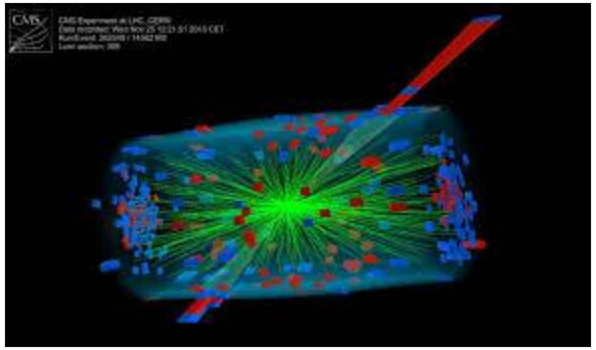
*The publicly available code MCFM v8.0 provides predictions for color singlet production at Next-to-Next-to Leading Order. The code runs in parallel using a hybrid openMP/MPI version of Vegas enabling MCFM to get accurate distributions in a few hours of runtime on moderate clusters of a few hundred computing cores. Included processes in this version are **W, Z, H, WH, ZH** and **di-photon** production.*

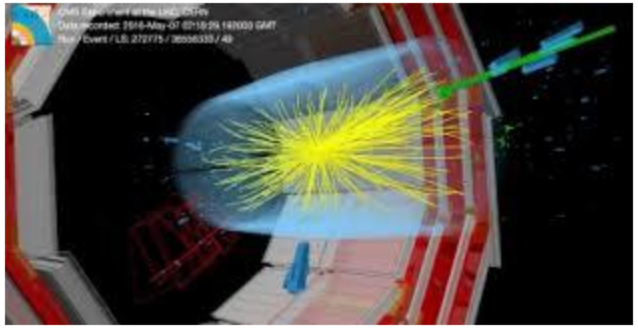
LoopFest 2016, Buffalo

Introduction

- The LHC will transition over time into more and more precision measurements.
- This requires precise theoretical predictions of signal and background to compare with.
- One of the improvements on the predictions comes from NNLO fixed order parton level Monte Carlo's
- These can be combined with Shower Monte Carlo's to construct tools which should be able to match future precision measurements.







NNLO Monte Carlo's

- More and more NNLO Monte Carlo's are being implemented:
  - *PP →V*: *Catani, Cieri, Ferrera, Florian, Grazzini (2009)*
    Li, Petriello (2012) →FEWZ
    Boughezal, Campbell, Ellis, Focke, Giele, Liu, Petriello, Williams (2016) → MCFM v8.0
  - *PP → H*: *Catani, Grazzini, Sargsyan (2007, 2008, 2013)*→HNNLO
    Anastasiou, Duhr, Dulat, Herzog, Mistlberger (2015)→up to N^3LO
    Boughezal, Campbell, Ellis, Focke, Giele, Liu, Petriello, Williams (2016) → MCFM v8.0
  - *PP →WH*: *Ferrera, Grazzini, Tramontana (2014)*
    Campbell, Ellis, Williams (2016) → MCFM v8.0
  - *PP →ZH*: *Ferrera, Grazzini, Tramontana (2015)*
    Campbell, Ellis, Williams (2016) → MCFM v8.0
  - *PP →di-photon*: *Catani, Cieri, de Florian, Ferrera, Grazzini (2012)*
    Campbell, Ellis, Li, Williams (2016) → MCFM v8.0

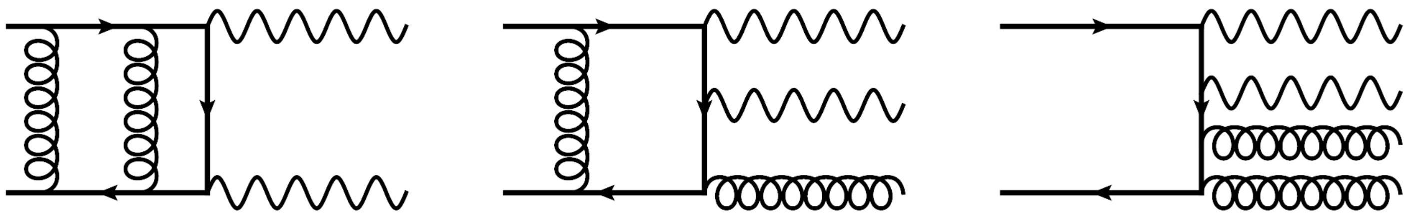
NNLO Monte Carlo's

- *PP →H + jet*: *Chen, Gehrmann, Glover, Jaquier (2015,2016)*
  Boughezal, Caola, Melnikov, Petriello (2015)
  Boughezal, Focke, Giele, Liu, Petriello (2015)→ to be in MCFM v8.1
- *PP →W+jet*: *Boughezal, Focke, Liu, Petriello (2015)* → to be in MCFM v8.1
- *PP →Z+jet*: *Gehrmann-de Ridder, Gehrmann, Glover, Huss, Morgan (2015)*
  Boughezal, Campbell, Ellis, Focke, Giele, Liu, Petriello (2015)
  → to be in MCFM v8.1
- *PP->H (WBF): Cacciari, Dreyer, Karlberg, Salam, Zanderighi (2015)*
- *H →bb: Anastasiou, Herzog, Lazopoulos (2012)*
  Del Duca, Duhr, Somogyi, Tramontano, Troscanyi (2015)
- *PP →ZZ*: *Cascioli, Gehrmann, Grazzini, Kallweit, Maierhofer, von Manteuffel, Pozzorini, Rathlev, Trancredi, Weihs (2014)*
  Caola, Melnikov, Rontsch, Trancredi (2015)

NNLO Monte Carlo's

- *PP →WW*: *Gehrmann, Grazzini, Kallweit, Maierhofer, von Manteuffel, Pozzorini, Rathlev, Tracredi (2014)*
  Caola, Melnikov, Rontsch, Trancredi (2015)
- *PP →V+photon: Grazzini, Kallweit, Rathlev (2016)*
- *PP →tt*: *Czakon, Fiedler, Heymez, Mitov (2016)*
  Abelof, Gehrmann-de Ridder, Majer (2015)
- *PP →single top*: *Brucherseifer, Caola, Melnikov (2014)*
- *Top decay*: *Brucherseifer, Caola, Melnikov (2013)*
- *PP →Di-Jets*: *Currie, Gehrmann-de Ridder, Gehrmann, Glover, Pires, Wells (2014)*

NNLO calculations: Virtual Corrections



- Foremost we need two-loop matrix elements of which most of the *2→2* are known by now.
- The ones used in MCFM v8.0:
  - *PP→V*: *Hamberg, van Neerven, Matsuura (1991)*
  - *PP→H*: *Harlander, Kilgore (2003)*
  - *PP→VH*: *Brein, Djouadi, Harlander (2004);*
    Brein, Harlander, Zirke (2013)
  - *PP→di-photon*: *C. Anastasiou, E.W.N. Glover, Tejeda-Yeomans (2002)*

NNLO Calculations: Bremsstrahlung

- A very important component for Monte Carlo's are the bremsstrahlung matrix elements
  - We need **a fast** and **accurate** NLO Monte Carlo prediction extending into the soft/collinear regions in phase space.
  - MCFM provides validated and precise predictions for this using analytic matrix elements developed and used for over almost 2 decades.
  - MCFM has hundreds of processes already build in at NLO, re-using these to upgrade the Monte Carlo to NNLO is a real time saver.

MCFM for the Tevatron and the LHC, Campbell and Ellis, 2010

Soft/Collinear Radiation (1)

- Treating the soft/collinear singularities at NNLO extended from local subtraction methods developed at NLO
  - Sector decomposition: *Anastasiou, Melnikov, Petriello (2003)*
  - Antenna subtraction: *Gehrmann-De Ridder, Gehrmann, Glover (2005)*
  - Colorful NNLO: *Del Duca, Somogyi, Trocsanyi (2005)*
  - Sector-improved subtraction: *Czakon (2010)*
- However, slicing methods greatly simplified the Monte Carlo's and allowed for re-use of existing NLO Monte Carlo's such as MCFM to calculate bremsstrahlung:
  - Above slicing cut → Re-use NLO MCFM (fast and validated)
  - Below slicing cut → Use two-loop matrix elements + analytic slicing function

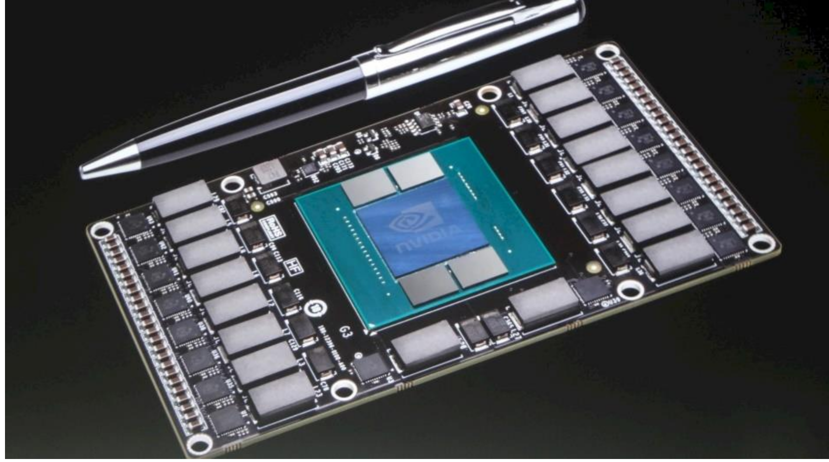
Soft/Collinear Radiation (2)

- Implementation of slicing:
  - In MCFM we use “non-local jettiness subtraction”. This is a slicing method with the advantage that all analytic slicing functions have been calculated already.
    - W-boson production in association with a jet at next-to-next-to-leading order in perturbative QCD, R. Boughezal, C. Focke, X. Liu and F. Petriello, 2015
    - N-jettiness Subtractions for NNLO QCD Calculations, J. Gaunt, M. Stahlhofen, F.J. Tackmann and J.R. Walsh, 2015
  - This is similar in methodology to q_t-subtraction, however it also works for jet final states.
    - An NNLO subtraction formalism in hadron colliders and its applications to Higgs boson production at the LHC, S. Catani and M. Grazzini, 2007

Use of Parallelism

- Performing the numerical integration of the single and double bremsstrahlung contributions will require a lot of computer power
- Using parallel computing to efficiently harvest all resources of a computer cluster is desirable
- Parton Monte Carlo's use VEGAS which is a perfect vehicle for parallel programming
  - Vegas is an adaptive Monte Carlo based on importance sampling
  - Each sweep in VEGAS evaluates a certain number of independent events
  - After a sweep all results are collected and the grid is optimized to reduce the weight fluctuations.
  - The new grid is used to optimize the integration for the new sweep of events

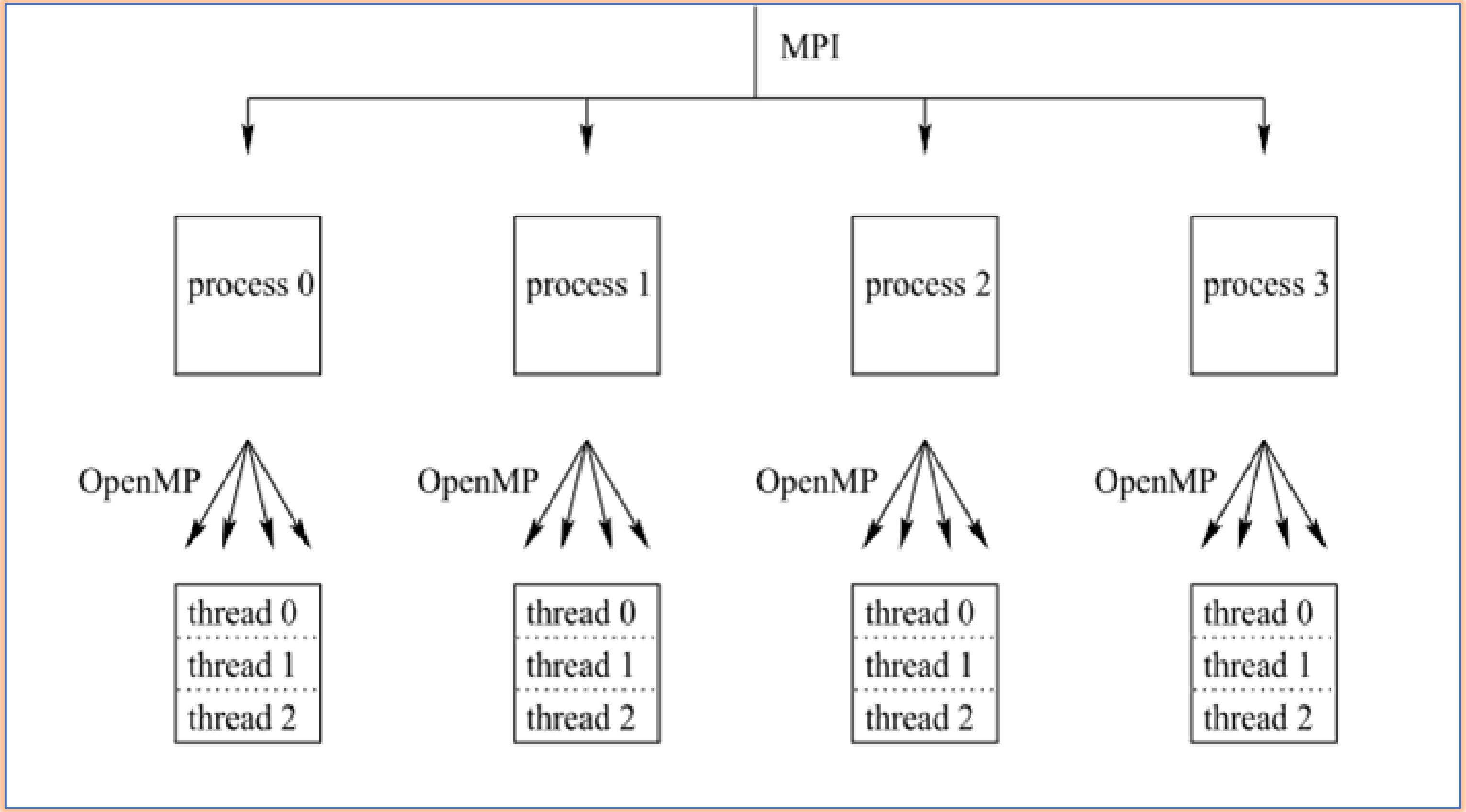
Shifting Sands of Hardware



- Used to programming in a non-parallel manner with a glut of memory favoring storing over on the fly recalculation.
- Technology is moving rapidly to “many integrated core” architecture.
- To take advantage of these new developments, parallel programming is needed (openMP, CUDA, openACC,…).
- Optimal methods to do Monte Carlo programs can shift in non-intuitive manners due to shifts in technology:
  - Limiting off-chip memory → PDF grid vs PDF evolution
  - Preference for brute force simple methods → slicing vs subtraction
- With careful programming one can ultimately run the algorithm on GPU’s

Hybrid openMP/MPI

- Two levels of parallelism:
  - OpenMP: For use on single processor to use the multiple cores on the processor. Uses a unified memory. Time consuming to develop, easy to use
  - MPI: Connects the different processors. Communication through messaging over e.g. infiniband. Quick to develop, harder to use.

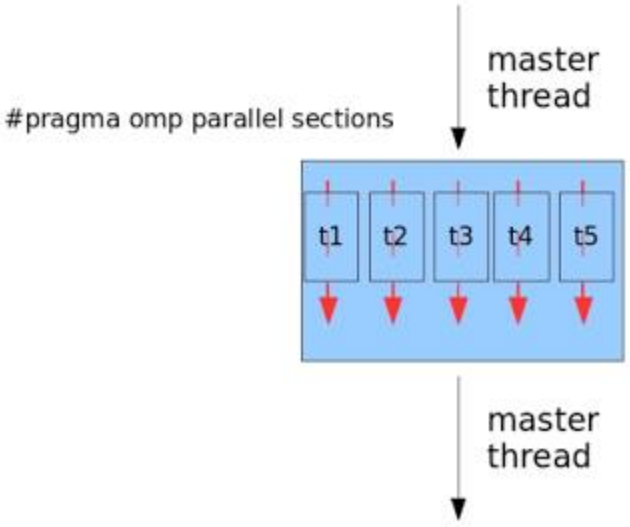


With the advance of MIC architecture with unified memory openMP/openACC will become more and more important in the future.

openMP

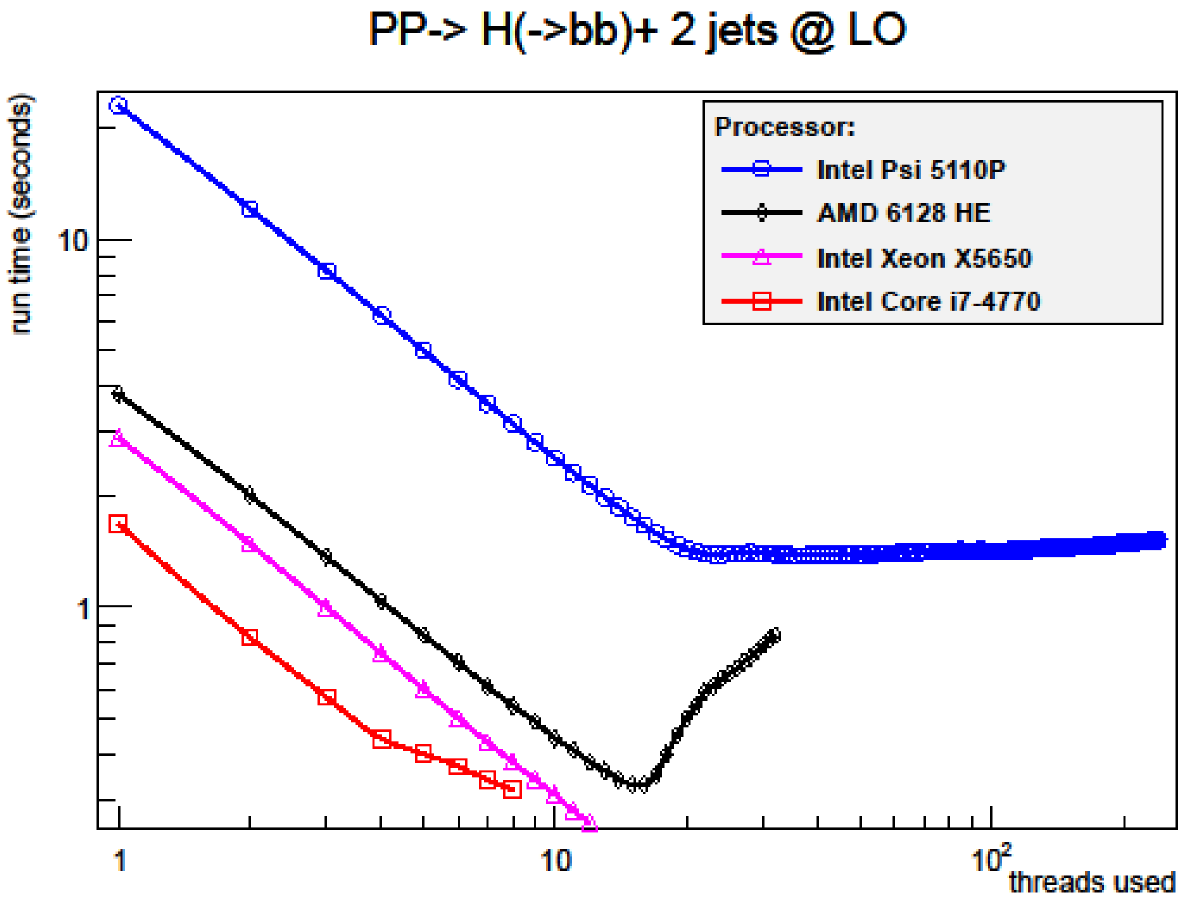
A Multi-Threaded Version of MCFM, J.M. Campbell, R.K. Ellis, W. Giele, 2015

- openMP can be used to parallelize a program using the available cores on a single motherboard.
- All cores see the same memory and cache (per processor).
- Shared memory leads to great speedups but also makes development/debugging harder.
- openMP is part of Intel and GNU compilers. It uses available cores without user interference. It is actively pushed by Intel for their MIC architecture (synergy between hardware and openMP)
- Simple pragma's are added to existing code (which are comments for non-openMP compilations)



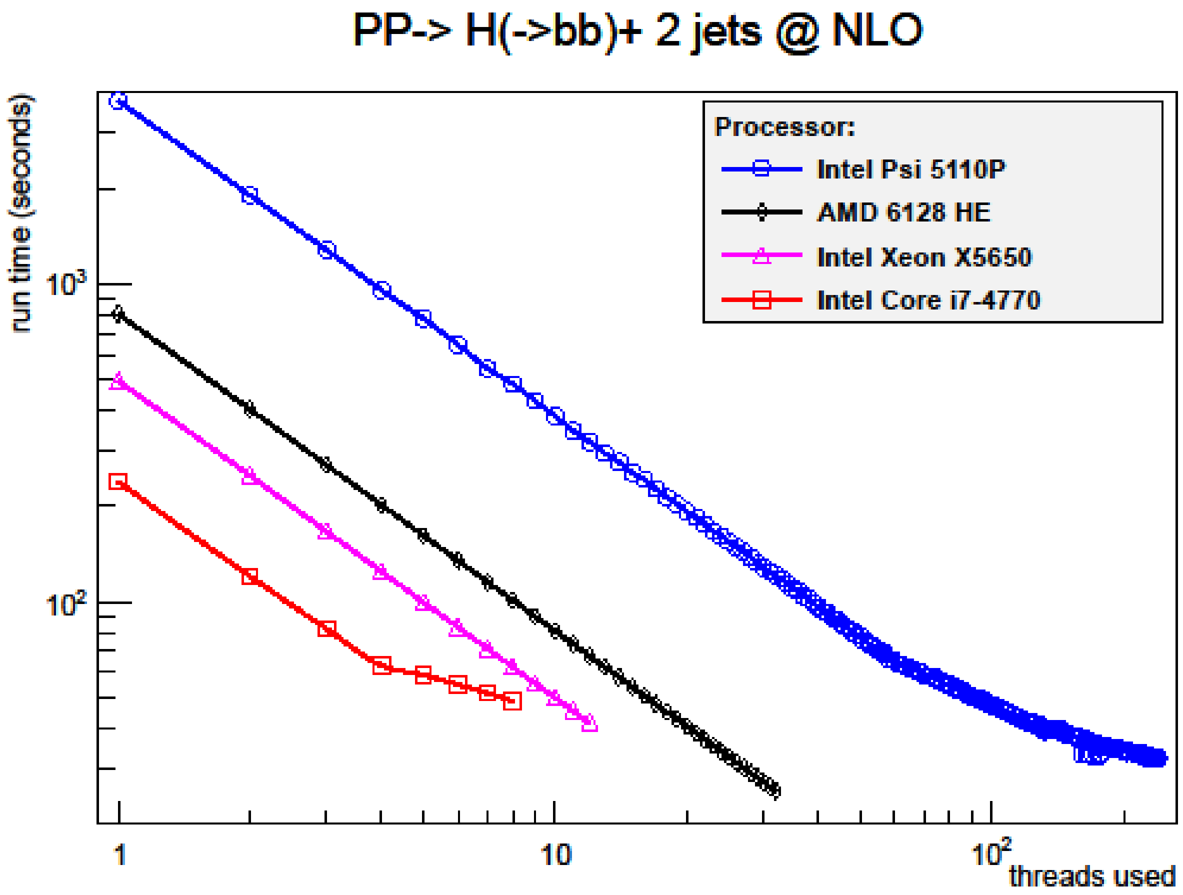
MCFM 7:openMP at LO

- Used *4x1,000+10x10,000* VEGAS events.
- Under perfect circumstances one would expect the run time to go as $t_n = t_1/n$.
- Deviations from that due to:
  - Hyper-threading
  - Memory bound limits because we do not calculate enough (memory transfer time dominates over computational time)
  - Traffic jams: We have excessive memory transfer between cache and main memory
  - Starvation: Number of events per thread low



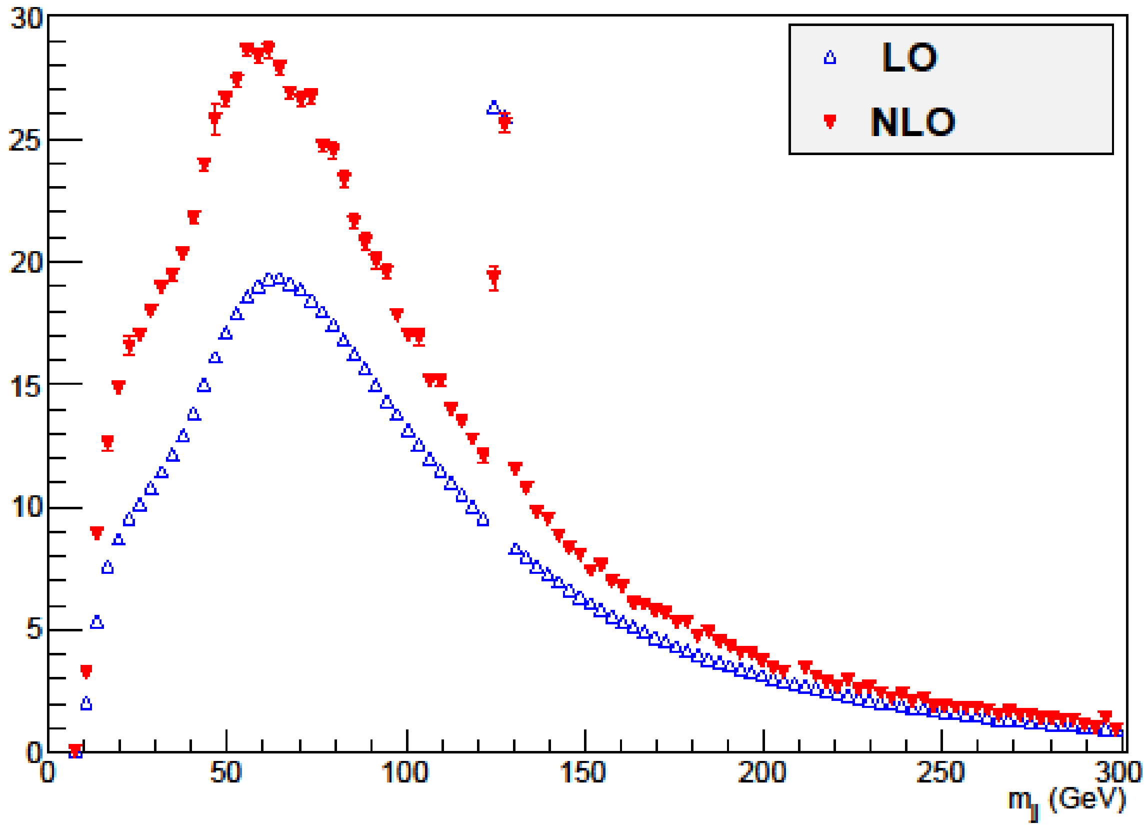
MCFM 7:openMP at NLO

- Used *4x1,000+10x10,000* VEGAS events
- Because the calculations are more complicated all memory bound issues have disappeared.
- openMP allows for doing the NLO phenomenology in on a workstation. No cluster required.
- Except for hyper threading the scaling is nearly perfect.



MCFM 7: NLO on a workstation

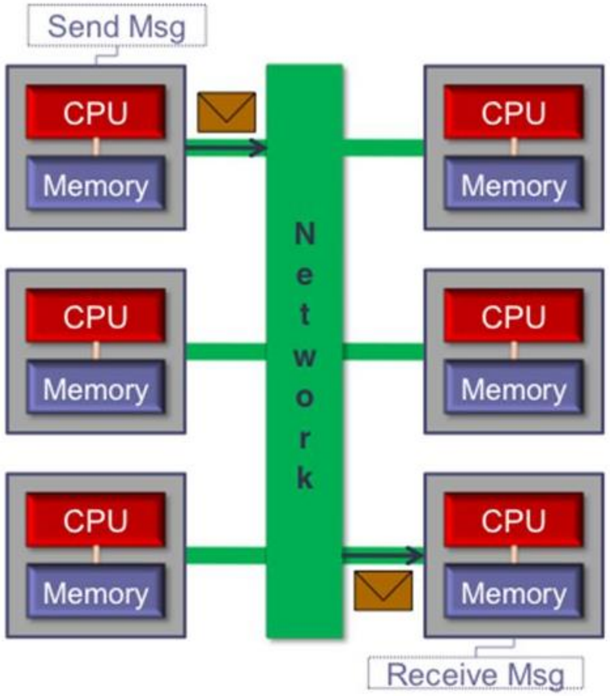
- The dijet invariant mass distribution (5 GeV bins) for *PP → H(→bb) + 2 jets* at NLO
- Used *4x1,500,000+10x15,000,000* VEGAS events
- LO: 12 min on *2x6* core dual Intel Xeon X5650
- NLO: 22 hours on the *4x8* core quad AMD Opteron system



Hybrid openMP/MPI

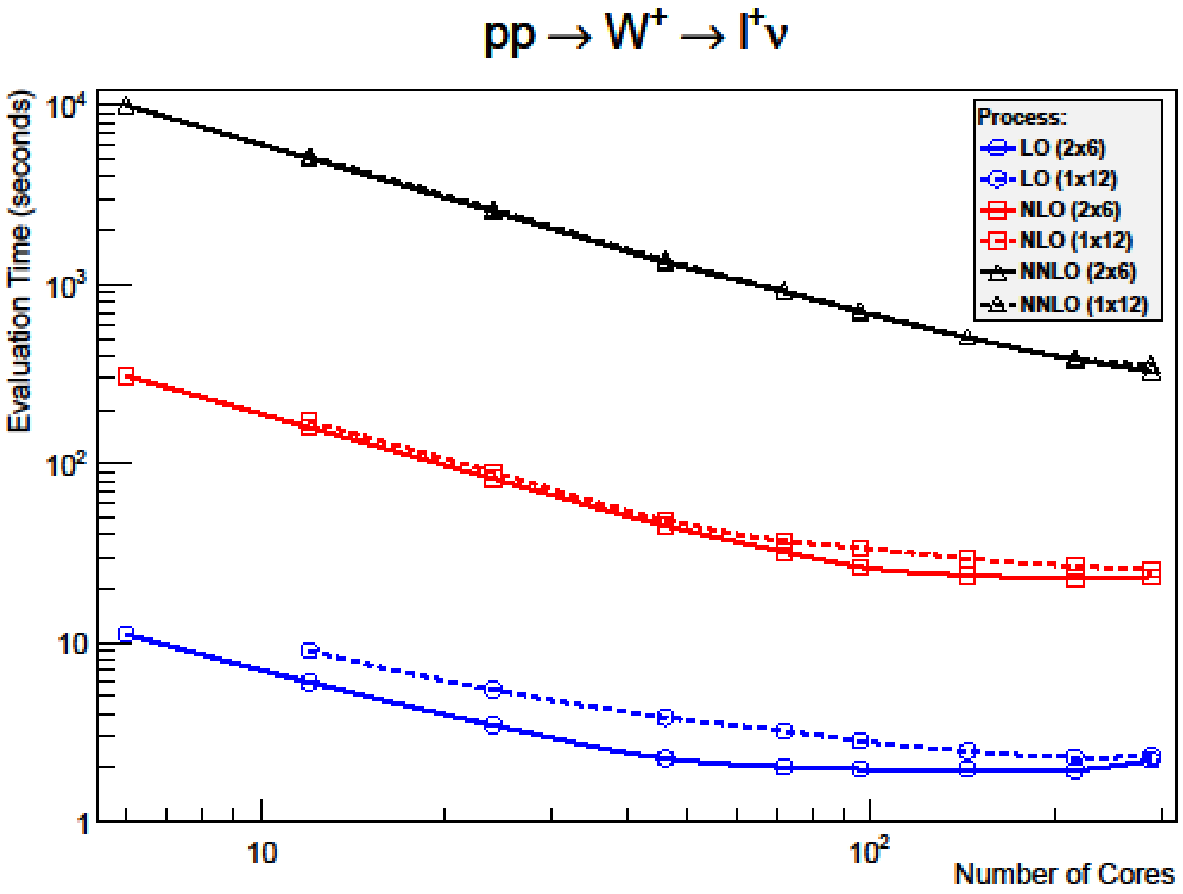
- Higgs boson production in association with a jet at NNLO using jettiness subtractions,
  R. Boughezal, C. Focke, W. Giele, X. Liu, F. Petriello, 2015
- Color singlet production at NNLO in MCFM,
  R. Boughezal, J.M. Campbell, R.K. Ellis, C. Focke, W. Giele, X. Liu, F. Petriello, C. Williams, 2016

- MPI can be used to parallelize the program over multiple processors, each with its own memory/cache.
- Communication between the different processors is slow and should be minimized.
- Easy to program, though one has to add explicit program statements. So program will not compile/run outside MPI.
- Harder to use, need some software to submit jobs in queues. No MPI standard.



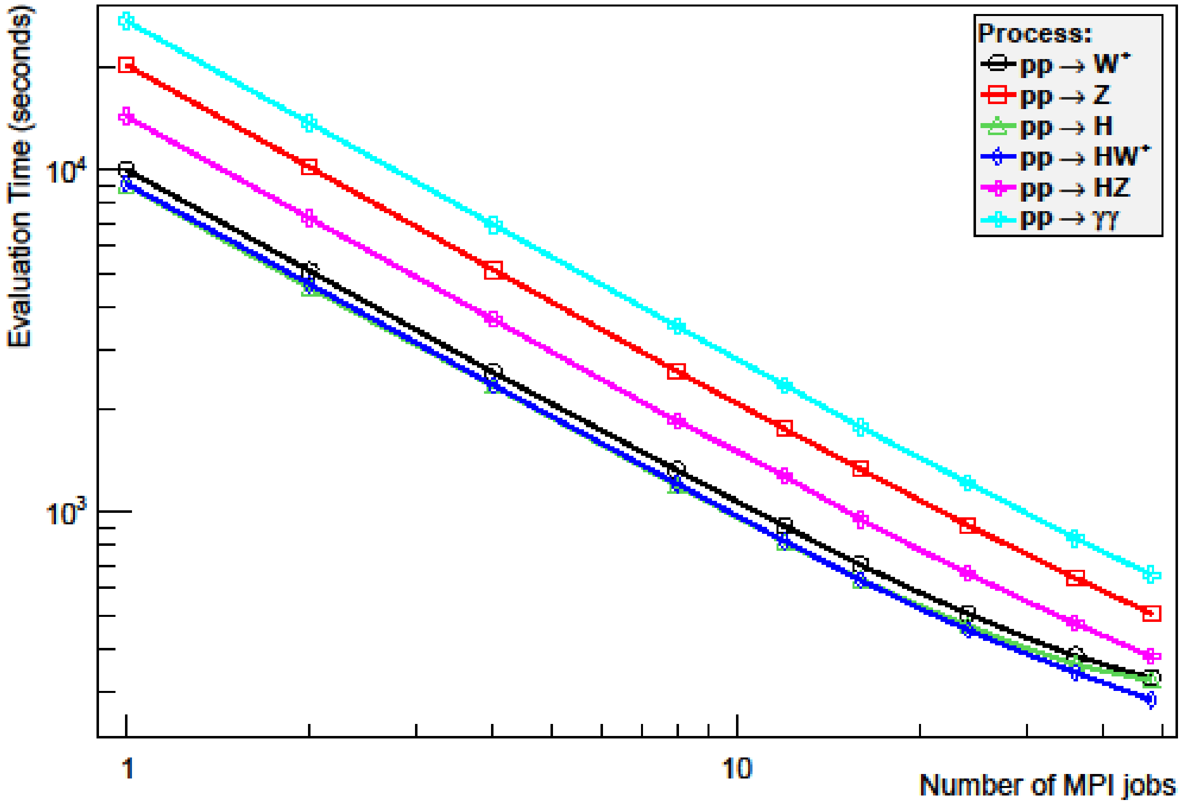
MCFM 8: Thread scaling (1)

- Runtime of *pp→W⁺* for LO/NLO/NNLO from 1 up to 288 cores
- The cluster consists of 24 nodes, each containing 2 processors of 6 cores
- Two running modes:
  - 1 MPI job per node: 1x12 (divided cache)
  - 2 MPI jobs per node, i.e. 1 MPI job per processor: 2x6
- Used *4x100,000+10x1,000,000* Vegas events
- LO/NLO stopped scaling above 50/100 cores → Memory dominated regime.
- 1x12 runs slower than 2x6 because openMP does not have to sync cache between the 2 processors in the 2x6 case.



MCFM 8: Thread scaling (2)

- The NNLO scaling for all singlet processes included in MCFM 8.0 as a function of the number of MPI jobs
- Used *4x100,000+10x1,000,000* Vegas events
- Each MPI job is one processor with 6 cores
- Only the *PP→H* shows the onset of non-scaling at 48 MPI jobs.
- All other processes can be speed up efficiently using a larger cluster



MCFM 8: Timing

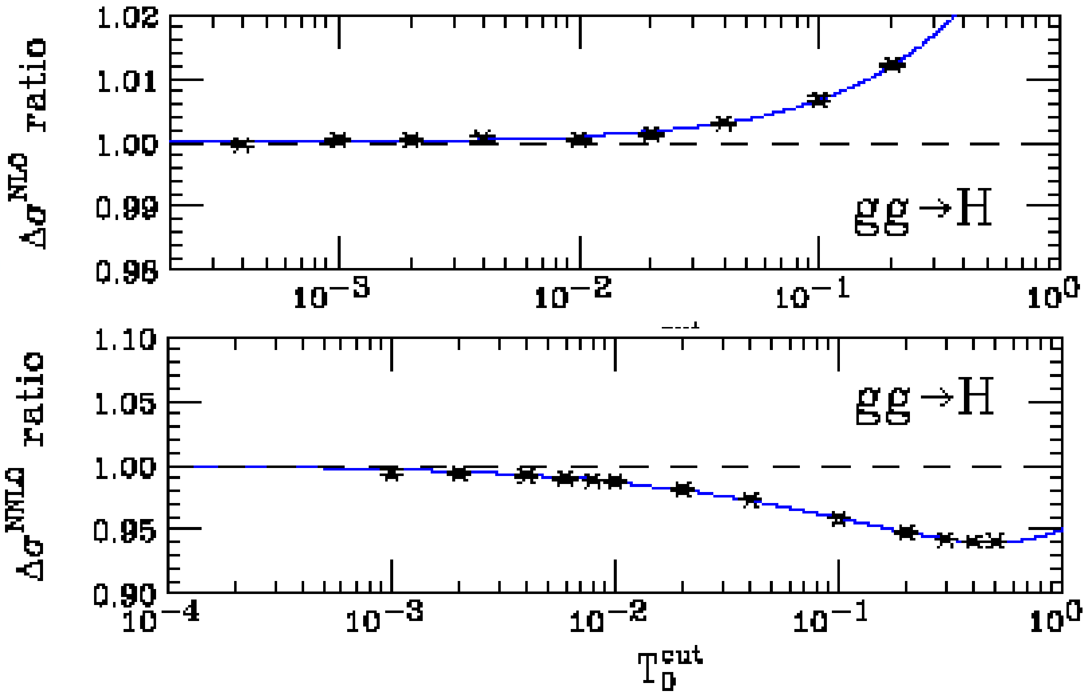
$\mathcal{T}_0^{cut}$	W^+	Z	H	HW^+	HZ	$\gamma\gamma$
0.001	2% (1397)	0.9% (2770)	0.05% (1256)	10% (1263)	6% (1939)	0.4% (3706)
0.005	0.7% (1358)	0.4% (2701)	0.04% (1234)	3% (1238)	2% (1906)	0.2% (3661)
0.01	0.5% (1356)	0.2% (2677)	0.04% (1214)	2% (1222)	1% (1847)	0.15% (3585)
0.05	0.2% (1315)	0.08% (2572)	0.04% (1197)	0.6% (1206)	0.4% (1841)	0.09% (3492)
0.1	0.09% (1307)	0.05% (2526)	0.04% (1186)	0.3% (1186)	0.2% (1847)	0.08% (3427)
0.5	0.04% (1266)	0.04% (2356)	0.04% (1176)	0.1% (1150)	0.09% (1768)	0.07% (3376)

Table 10. The relative statistical precision (in percentages) on the $pp \to W^+ \to l^+\nu$, $pp \to Z \to l^+l^-$, $pp \to H \to \gamma\gamma$, $pp \to H + W^+ \to \gamma\gamma + l^+\nu$ $pp \to H + Z \to \gamma\gamma + l^+l^-$ and $pp \to \gamma\gamma$ cross sections at NNLO as a function of $\mathcal{T}_0^{cut}$ (in GeV) using $4 \times 2 \times 6$ cores. Also given in brackets is the evaluation time (in seconds).

- Note the runtime is for only 48 cores using *4x100,000+10x1,000,000* Vegas events.
- As can be seen the value of the slicing cut greatly affects the achieved statistical precision.
- The choice of the value of the jettiness slicing cut is important
  - Large: the power correction contributes (incomplete cancelation)
  - Small: the statistical error is large

MCFM 8: curve fitting

Compared to ggh@nnlo



- The power corrections have a know functional form

$$\Delta\sigma^{NLO}_{\text{jettiness}}(\mathcal{T}_0^{cut}) = \Delta\sigma^{NLO} + c \times \left(\frac{\mathcal{T}_0^{cut}}{Q}\right) \times \log\left(\frac{\mathcal{T}_0^{cut}}{Q}\right) \quad \text{(+ polynomial)}$$

$$\Delta\sigma^{NNLO}_{\text{jettiness}}(\mathcal{T}_0^{cut}) = \Delta\sigma^{NNLO} + c_3 \times \left(\frac{\mathcal{T}_0^{cut}}{Q}\right) \times \log^3\left(\frac{\mathcal{T}_0^{cut}}{Q}\right) + c_2 \times \left(\frac{\mathcal{T}_0^{cut}}{Q}\right) \times \log^2\left(\frac{\mathcal{T}_0^{cut}}{Q}\right)$$

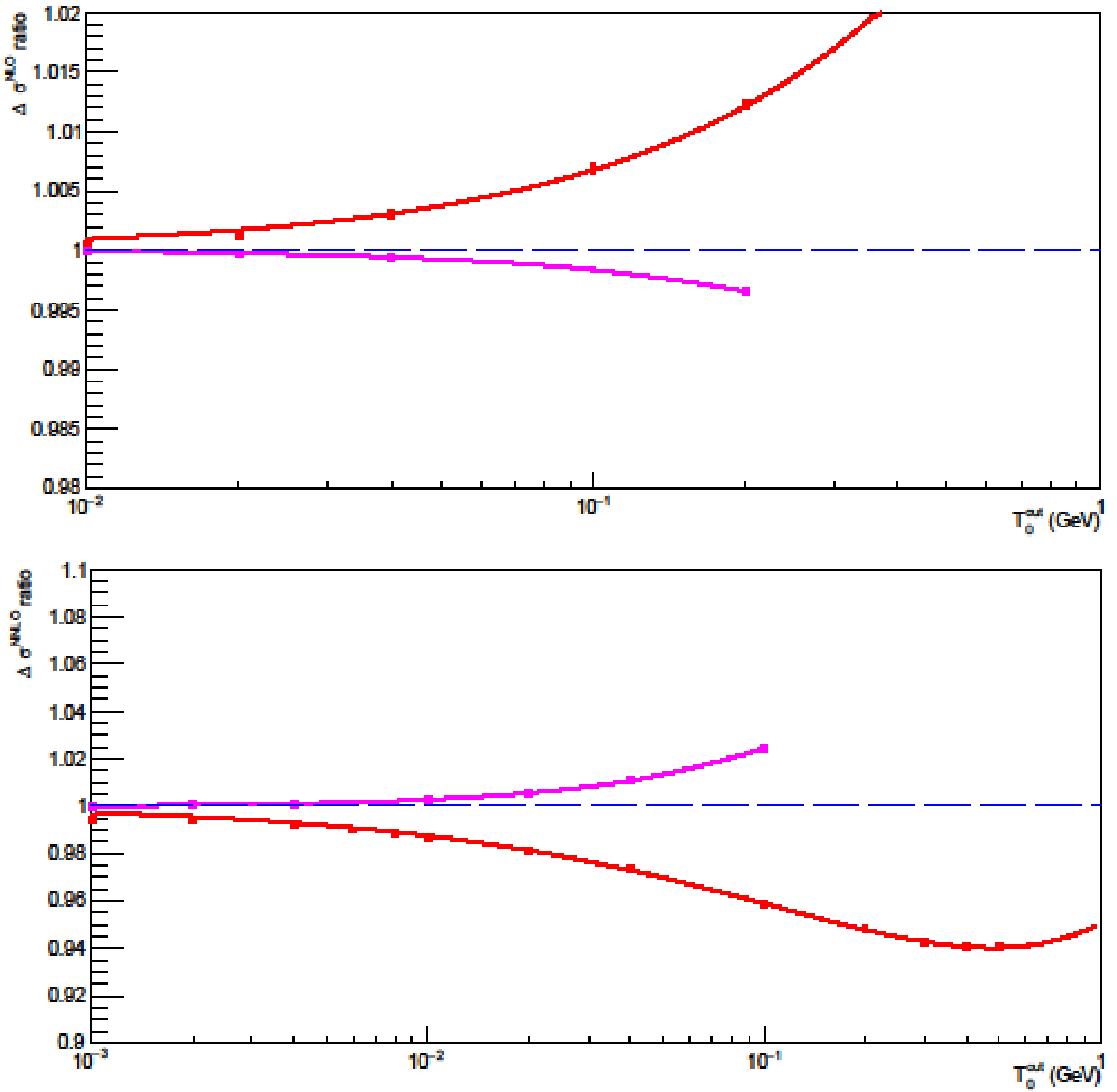
N-jettiness Subtractions for NNLO QCD Calculations,
J. Gaunt, M. Stahlhofen, F.J. Tackmann and J.R. Walsh, 2015

- MCFM is designed to produce the (differential) cross section for a given slicing cut
- One can run for a handful of cut values and fit the parameter to find the extrapolation to zero.
- This can be fully automated in e.g. root or Mathematica with a per bin fit including the statistical error
- The fit does not only give the parameters but also goodness of fit, indicating if more terms in the expansion are required.

NLO : $c_0 = 1.000$; $c_1 = -1.17$
NNLO: $c_0 = 0.998$; $c_3 = 0.324$; $c_2 = 1.30$

Improving slicing

- The leading power correction can be calculated and added to MCFM
  R. Boughezal, X. Liu, F. Petriello, 2016
- A preliminary study has been performed for *pp→H*.
- This should be included in a future release of MCFM, allowing one to choose much larger values of the cut.
- This holds great promise, especially if one can calculate more and more of the coefficients.

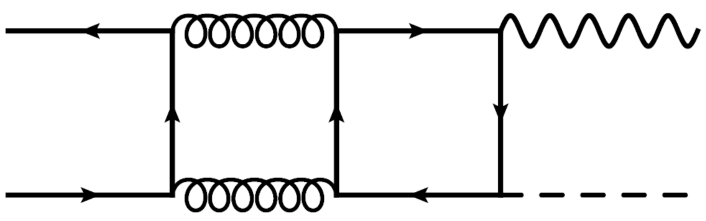


Phenomenology

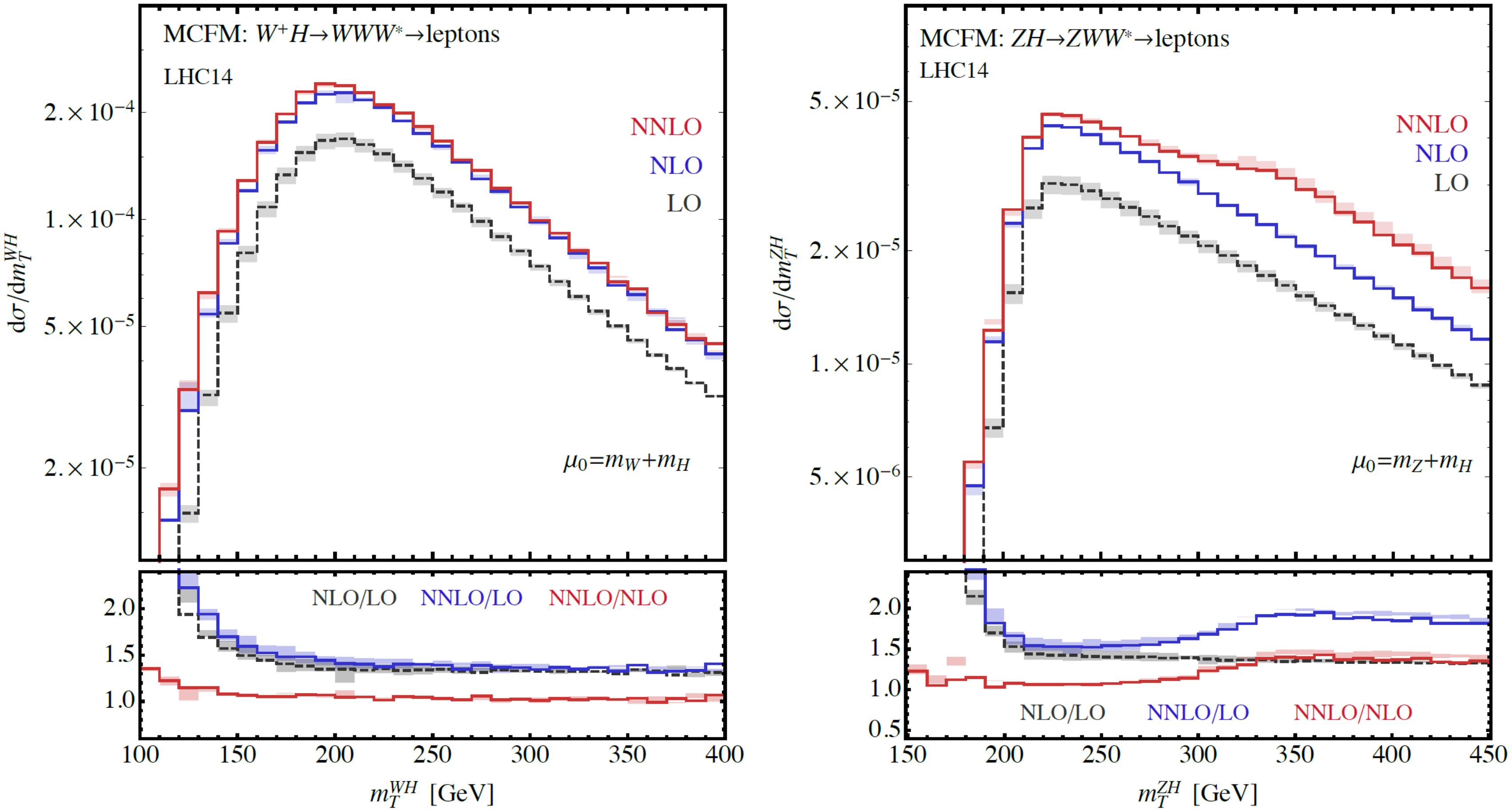
- In the end, the discussed technical aspects of MCFM are to a large degree hidden.
- One can use it out of the box, without much thought.
- openMP will by default use all available threads on a cluster node.
- MPI (if used) will work when submitting one job/cluster node
- The slicing scale is set within the program, depending on the precision specified.

Phenomenology: *PP →VH →V+V'V'*

- Demonstrates the MCFM set-up with jettiness slicing for a high dimensional phase space (6 particle final state)
- Shows importance of NNLO to describe the phenomenology
- p_T^l>25 GeV; $|\eta_l|$<2.5; E_t^{miss}>20 GeV
- While corrections to inclusive cross sections are order percent at NNLO, the effects can be large in local phase space regions: e.g. top loop threshold.



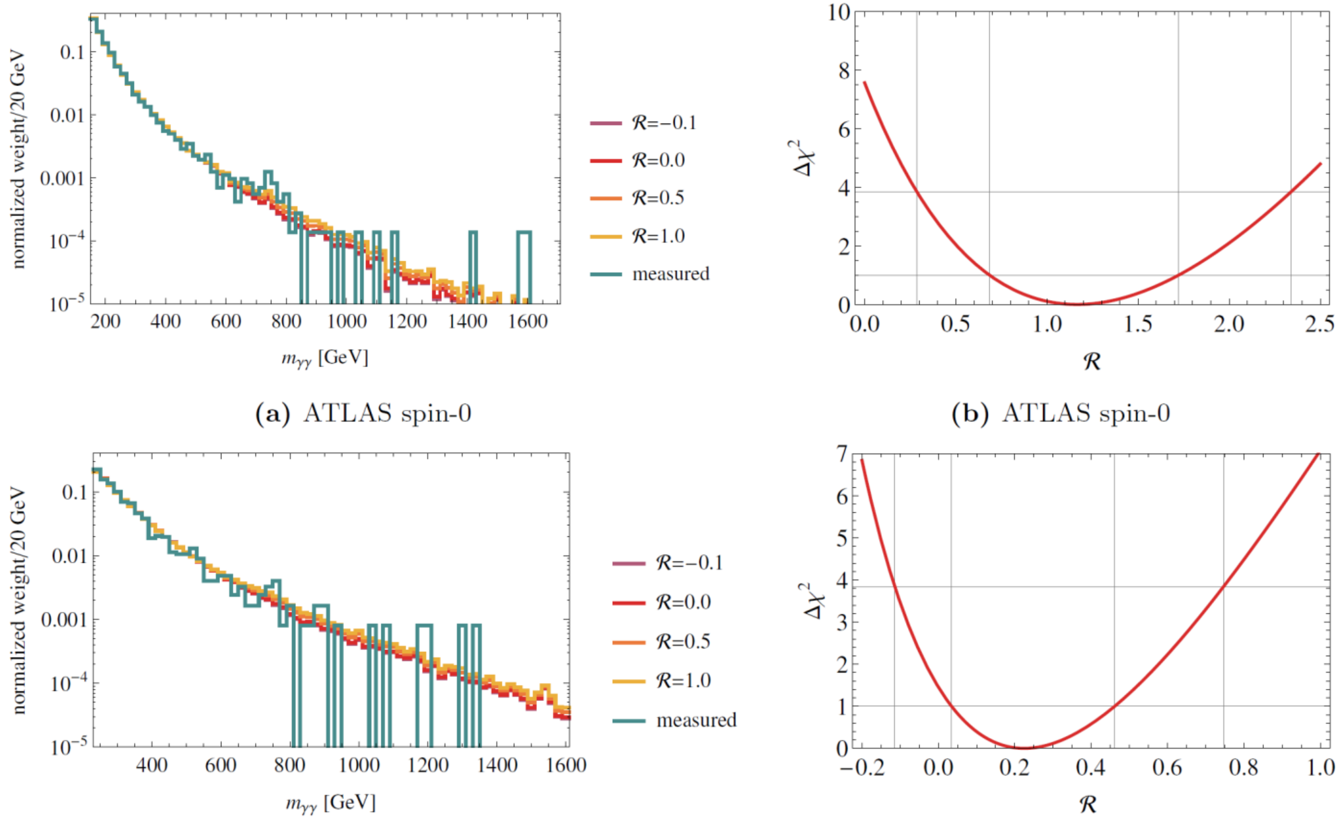
Associated production of a Higgs boson at NNLO, Campbell, Ellis, Williams (2015).



Phenomenology: *PP → di-photon*

On the challenge of estimating diphoton backgrounds at large invariant mass, Kamerik, Perez, Schlaffer, Weiler (2016).

- Shows one can download and use NNLO MCFM without any help from the authors.
- Shows that with NNLO precise predictions can be made for LHC phenomenology



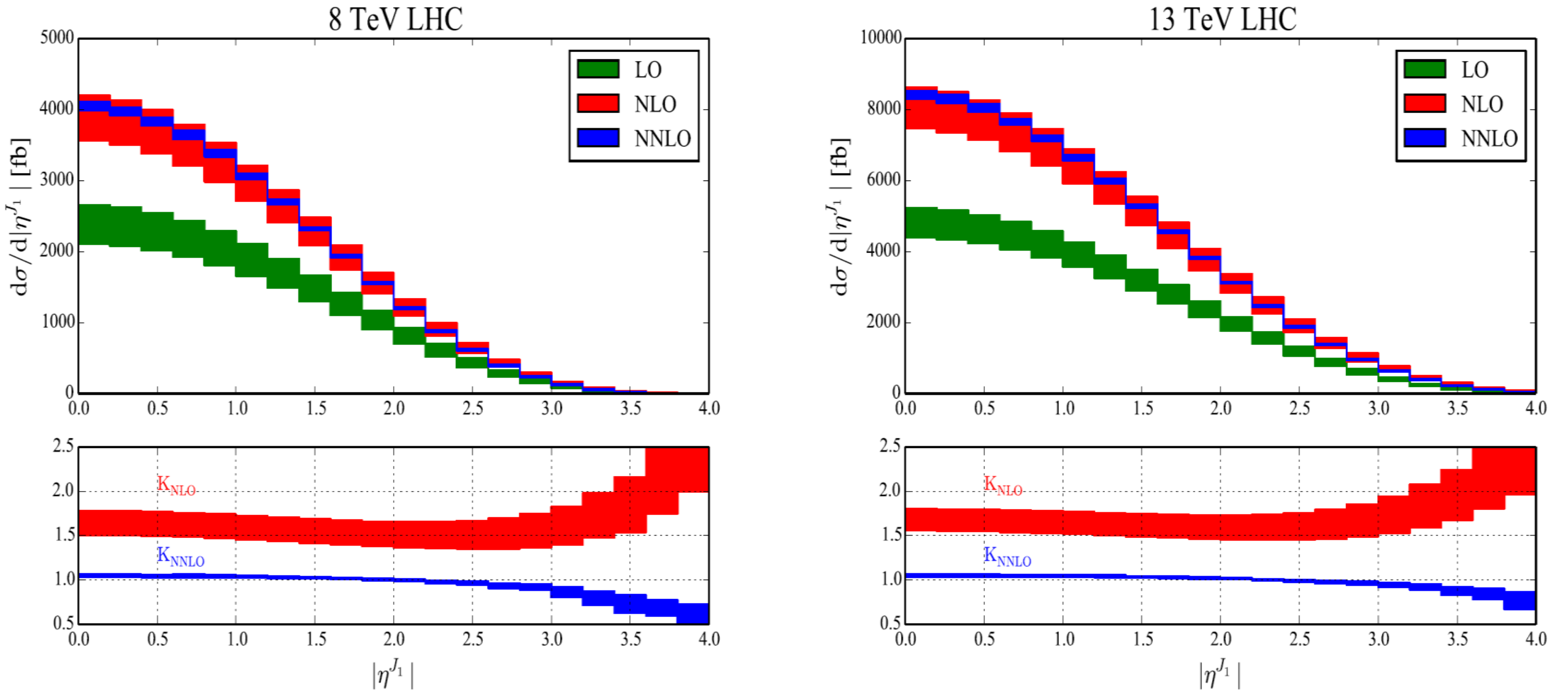
(a) ATLAS spin-0 (b) ATLAS spin-0

(c) CMS EBEB (d) CMS EBEB and EBEE combined

Phenomenology: *PP →Z+jet*

Phenomenology of the Z-boson plus jet process at NNLO,
Boughezal, Liu, Petriello (2016)

- The next version of NNLO MCFM will have more complicated final states including a jet.
- *PP →Z+jet* result from non-public version.
- This particular plot shows the importance of NNLO for large rapidity jet production
- *p_T^J>100 GeV; $|\eta_J|$<4.4*
  p_T^l>25 GeV; $|\eta_l|$<2.5
  71 GeV<M_{ll}<111 GeV



Outlook

- MCFM v8 is available for download from mcfm.fnal.gov. It will produce usable NNLO results on desktops & small clusters.
- The next step is including $pp \rightarrow X + jet$ into the public version of MCFM.
- Many other projects in the works to further develop MCFM@NNLO.
- The rapidly emerging technologies will keep pushing us into different numerical models for doing the calculations