Interpolation grids in NNLO with NNLOJET

D. Britzger, K. Rabbertz, C. Gwenlan,
A. Huss, M. Sutton, J. Pires, et al.

Max-Planck-Institut für Physik
(Werner-Heisenberg-Institut)

Technique of interpolation grids

Motivation

- Any phenomenological study relies on a sufficiently fast repetition of theoretical predictions

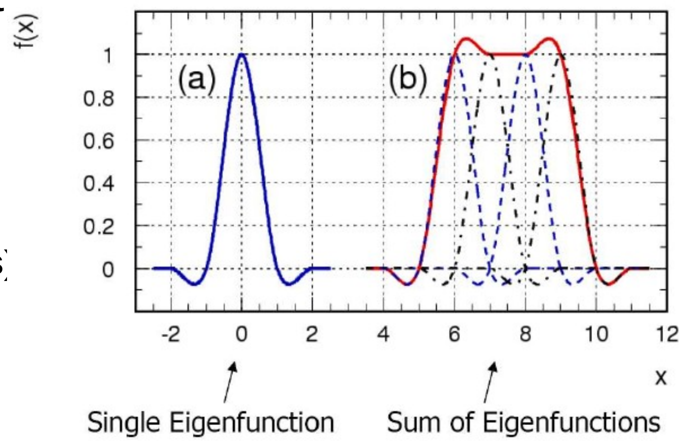
→ higher-order pQCD predictions are often 'slow'

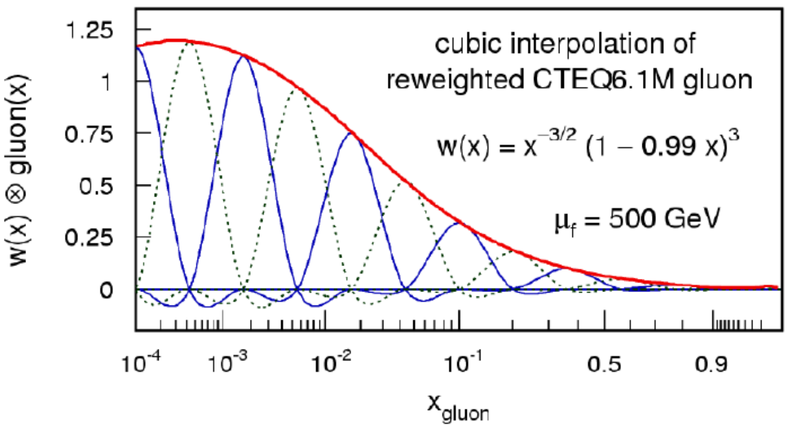
fastNLO and APPLgrid concept

- Introduce interpolation kernel (unit operator; lagrange polynomials)
  Set of functions f(x) around n discrete x-nodes
- Single PDF is replaced set of interpolation kernels

$$f_a(x) \cong \sum_i f_a(x_i) \cdot E^{(i)}(x)$$

- Improve interpolation by reweighting PDF





Scale dependence

- Similar interpolation procedure also for scales
- Two approaches implemented
  - 'fixed scales':
    μ_R and μ_F are specified during grid generation
  - 'flexible scale':
    Coefficients are fully independent on μ_R and μ_F

News on fastNLO and APPLgrid

APPLgrid

- NNLO scale variations
- ep grids validated

fastNLO

- Somewhat improved speed for convolution of *ep* tables
- a few minor bug fixes

APPLgrid and fastNLO

- New converter: fastNLO ↔ APPLgrid
  conversion based on evaluation of grid and filling new grid (by M. Sutton)
  Advantage: fastNLO flexible scale tables with scale flexibility, but a bit slower
  → APPLgrid with smaller files, once scale is set
  In principle:
    also fastNLO (flex) → fastNLO (fixed scale) conversion feasible
    also APPLgrid → APPLgrid feasible, for instance for faster but less accurate grid

The Applfast project

- Started as common project of NNLOJET, APPLgrid, and fastNLO authors at QCD@LHC in London
- Interface between NNLOJET and fast grid technology - APPLgrid and fastNLO
- Aimed to be the least obtrusive as possible for both ends of the interface
- Intended to be reusable by other theory programs

Outcome of productive meetings

- Careful validation of all ingredients in NNLO
- Thorough cross checks and benchmarks between APPLgrid and fastNLO
- new developments for fastNLO & APPLgrid
- development of more standardised interfaces, common code, etc...

The nnlo-bridge interface

Standardised interface between generator code and grid codes

- Generator code calls hook functions
  → <u>If</u> linked with grid interface, then these functions generate the grids
- initialise grid for any 1D histogram

```
extern void (*appl_initptr)( const int& igrid,
                              const int& id,
                              const std::string& gridname,
                              const int& nbins,
                              const double* bins );
```

- Fill 1D grid & pass list of weights (scale-independent and scale dependent)

```
extern "C" void  fill_grid( const int & igrid,
                            const int & id,
                            const double & obs,
                            const double * wt,
                            const double * coeff )
```

- Write grid to file

```
extern "C" void  write_grid( const int & igrid,
                             const int & id,
                             const double & norm )
```

- All additional details passed by 'hook' functions: scales, x-values, process id, etc...

Features of Applfast, fastNLO, APPLgrid

- Automatic detection of requested process
- Development of technique to get 'scale-indpendent weights' from scale-dependent calculation → now standard in NNLOJET
- Caching of events prior to 'fill'
- Accumulation of different subprocesses for equivalent phase space space point
- Automatised optimisation and mapping of sub-channels

0	163 177 191 205 206 245 246 285 301 317 347 377 391 405 435 449 450 489 490 529 530 569 593 617 647 677 707 737 767 899 905 906 935 941 942	$(d, \bar{d}) + (s, \bar{s}) + (b, \bar{b})$
1	164 178 192 207 208 247 248 286 302 318 348 378 392 406 436 451 452 491 492 531 532 570 594 618 648 678 708 738 768 900 907 908 936 943 944	$(u, \bar{u}) + (c, \bar{c})$
2	165 179 193 209 210 249 250 287 303 319 349 379 393 407 437 453 454 493 494 533 534 571 595 619 649 679 709 739 769 909 917 918 945 953 954	$(\bar{d}, d) + (\bar{s}, s) + (\bar{b}, b)$
3	166 180 194 211 212 251 252 288 304 320 350 380 394 408 438 455 456 495 496 535 536 572 596 620 650 680 710 740 770 910 919 920 946 955 956	$(\bar{u}, u) + (\bar{c}, c)$
4	167 181 195 229 230 269 270 289 305 321 351 381 395 409 439 481 482 521 522 561 562 585 586 609 610 621 651 681 711 741 771 817 818 861 862 901 902 937 938	$(d, g) + (s, g) + (b, g)$
5	168 182 196 231 232 271 272 290 306 322 352 382 396 410 440 483 484 523 524 563 564 587 588 611 612 622 652 682 712 742 772 819 820 863 864 903 904 939 940	$(u, g) + (c, g)$
6	169 183 197 233 234 273 274 293 309 323 353 383 397 411 441 473 474 513 514 553 554 577 578 601 602 623 653 683 713 743 773 801 802 845 846 889 890 925 926	$(g, d) + (g, s) + (g, b)$
7	170 184 198 235 236 275 276 294 310 324 354 384 398 412 442 475 476 515 516 555 556 579 580 603 604 624 654 684 714 744 774 803 804 847 848 891 892 927 928	$(g, u) + (g, c)$
8	171 172 185 186 199 200 325 326 355 356 385 386 399 400 413 414 443 444 629 630 659 660 689 690 719 720 749 750 779 780 797 798 799 800 841 842 843 844 885 886 887 888 921 922 923 924	(g, g)
9	173 187 201 237 238 277 278 295 311 327 357 387 401 415 445 477 478 517 518 557 558 581 582 605 606 625 655 685 715 745 775 805 806 849 850 893 894 929 930	$(g, \bar{d}) + (g, \bar{s}) + (g, \bar{b})$
10	174 188 202 239 240 279 280 296 312 328 358 388 402 416 446 479 480 519 520 559 560 583 584 607 608 626 656 686 716 746 776 807 808 851 852 895 896 931 932	$(g, \bar{u}) + (g, \bar{c})$
11	175 189 203 241 242 281 282 297 313 329 359 389 403 417 447 485 486 525 526 565 566 589 590 613 614 627 657 687 717 747 777 833 834 877 878 913 914 949 950	$(\bar{d}, g) + (\bar{s}, g) + (\bar{b}, g)$
12	176 190 204 243 244 283 284 298 314 330 360 390 404 418 448 487 488 527 528 567 568 591 592 615 616 628 658 688 718 748 778 835 836 879 880 915 916 951 952	$(\bar{u}, g) + (\bar{c}, g)$
13	213 253 335 365 423 457 497 537 635 665 695 725 755 785 813 821 822 857 865 866	$(d, \bar{d}) + (d, \bar{s}) + (d, \bar{b}) + (s, \bar{d}) + (s, \bar{s}) + (s, \bar{b}) + (b, \bar{d}) + (b, \bar{s}) + (b, \bar{b})$
14	214 254 336 366 424 458 498 538 636 666 696 726 756 786 814 858	$(d, \bar{u}) + (d, \bar{c}) + (s, \bar{u}) + (s, \bar{c}) + (b, \bar{u}) + (b, \bar{c})$
15	215 255 337 367 425 459 499 539 637 667 697 727 757 787 815 859	$(u, \bar{d}) + (u, \bar{s}) + (u, \bar{b}) + (c, \bar{d}) + (c, \bar{s}) + (c, \bar{b})$
16	216 256 338 368 426 460 500 540 638 668 698 728 758 788 816 823 824 860 867 868	$(u, \bar{u}) + (u, \bar{c}) + (c, \bar{u}) + (c, \bar{c})$
17	217 257 331 361 419 461 501 541 631 661 691 721 751 781 809 853	$(d, d) + (d, s) + (d, b) + (s, d) + (s, s) + (s, b) + (b, d) + (b, s) + (b, b)$
18	218 258 332 362 420 462 502 542 632 662 692 722 752 782 810 854	$(d, u) + (d, c) + (s, u) + (s, c) + (b, u) + (b, c)$
19	219 259 333 363 421 463 503 543 633 663 693 723 753 783 811 855	$(u, d) + (u, s) + (u, b) + (c, d) + (c, s) + (c, b)$
20	220 260 334 364 422 464 504 544 634 664 694 724 754 784 812 856	$(u, u) + (u, c) + (c, u) + (c, c)$
21	221 261 343 373 431 465 505 545 643 673 703 733 763 793 829 873	$(\bar{d}, \bar{d}) + (\bar{d}, \bar{s}) + (\bar{d}, \bar{b}) + (\bar{s}, \bar{d}) + (\bar{s}, \bar{s}) + (\bar{s}, \bar{b}) + (\bar{b}, \bar{d}) + (\bar{b}, \bar{s}) + (\bar{b}, \bar{b})$
22	222 262 344 374 432 466 506 546 644 674 704 734 764 794 830 874	$(\bar{d}, \bar{u}) + (\bar{d}, \bar{c}) + (\bar{s}, \bar{u}) + (\bar{s}, \bar{c}) + (\bar{b}, \bar{u}) + (\bar{b}, \bar{c})$
23	223 263 345 375 433 467 507 547 645 675 705 735 765 795 831 875	$(\bar{u}, \bar{d}) + (\bar{u}, \bar{s}) + (\bar{u}, \bar{b}) + (\bar{c}, \bar{d}) + (\bar{c}, \bar{s}) + (\bar{c}, \bar{b})$
24	224 264 346 376 434 468 508 548 646 676 706 736 766 796 832 876	$(\bar{u}, \bar{u}) + (\bar{u}, \bar{c}) + (\bar{c}, \bar{u}) + (\bar{c}, \bar{c})$
25	225 265 339 369 427 469 509 549 639 669 699 729 759 789 825 837 838 869 881 882	$(\bar{d}, d) + (\bar{d}, s) + (\bar{d}, b) + (\bar{s}, d) + (\bar{s}, s) + (\bar{s}, b) + (\bar{b}, d) + (\bar{b}, s) + (\bar{b}, b)$
26	226 266 340 370 428 470 510 550 640 670 700 730 760 790 826 870	$(\bar{d}, u) + (\bar{d}, c) + (\bar{s}, u) + (\bar{s}, c) + (\bar{b}, u) + (\bar{b}, c)$
27	227 267 341 371 429 471 511 551 641 671 701 731 761 791 827 871	$(\bar{u}, d) + (\bar{u}, s) + (\bar{u}, b) + (\bar{c}, d) + (\bar{c}, s) + (\bar{c}, b)$
28	228 268 342 372 430 472 512 552 642 672 702 732 762 792 828 839 840 872 883 884	$(\bar{u}, u) + (\bar{u}, c) + (\bar{c}, u) + (\bar{c}, c)$
29	291 307 573 597 897 933	$(d, d) + (s, s) + (b, b)$
30	292 308 574 598 898 934	$(u, u) + (c, c)$
31	299 315 575 599 911 947	$(\bar{d}, \bar{d}) + (\bar{s}, \bar{s}) + (\bar{b}, \bar{b})$
32	300 316 576 600 912 948	$(\bar{u}, \bar{u}) + (\bar{c}, \bar{c})$

Z+jets NNLO subprocesses:
794 processes mapped to 33 parton luminosities

Workflow for NNLO grids

1. Preprocessing: O(10 h)
- Check of interpolation quality
- Short test jobs to check interpolation settings (& optimise if necessary)

2. NNLOJET Warm-up: O(100 h)
- Vegas integration optimisation
- 1 long (multi-core) job per process

3. APPLgrid/fastNLO Warm-up: O(100 h)
- Adapt x- and scale-grids to accessed phase space (exact strategy differs between APPLgrid & fastNLO)
- Only phase space provided from NNLOJET → significant speed-up

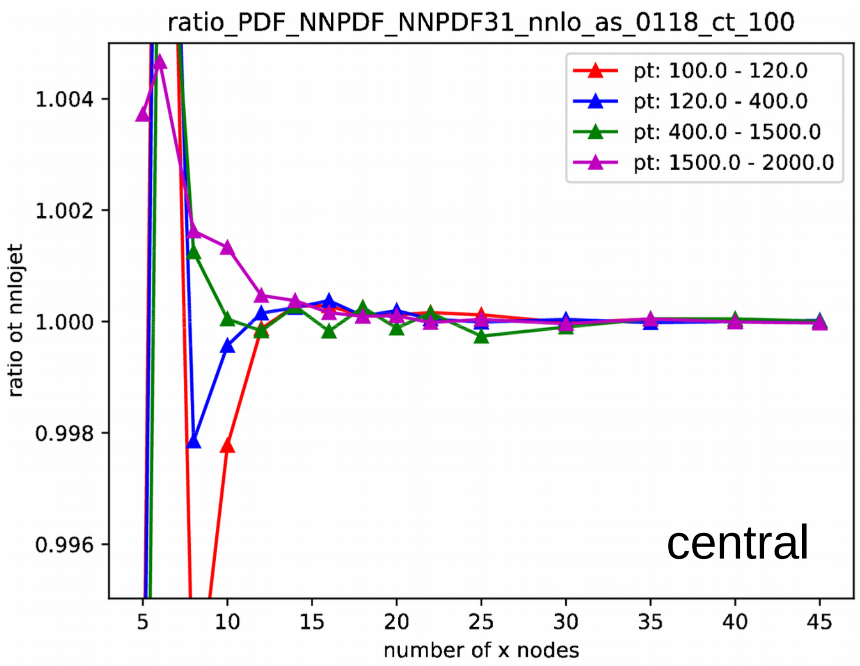
4. Interpolation grid production: O(250k h)
- Thousands of parallel jobs

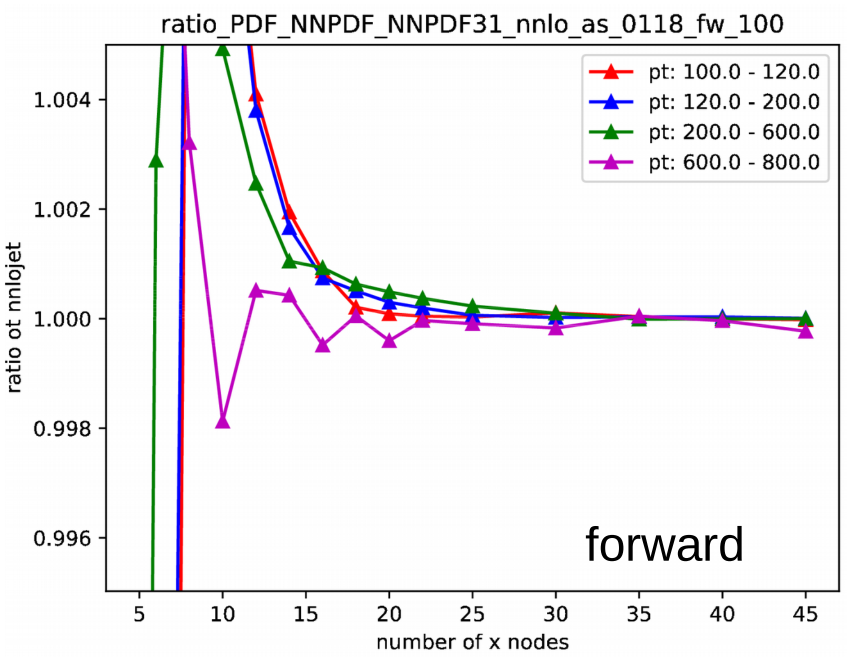
5. Postprocessing:
1) Statistical evaluation and combination of all single NNLOJET file O(h)
2) Merge grid-files with weights from step 1) O(40 h)
3) MergeJob to combine all grids and estimate statistical uncertainty O(min)

6. Validate, validate, and validate O(y)

Validation I

Grid closure
- Define σ(grid)/σ(nnlojet)
- Test case: ATLAS Inclusive jets @ 13TeV, R=0.6, LO
- Grid closure depends on number of x-nodes for PDF interpolation (see x-axis)





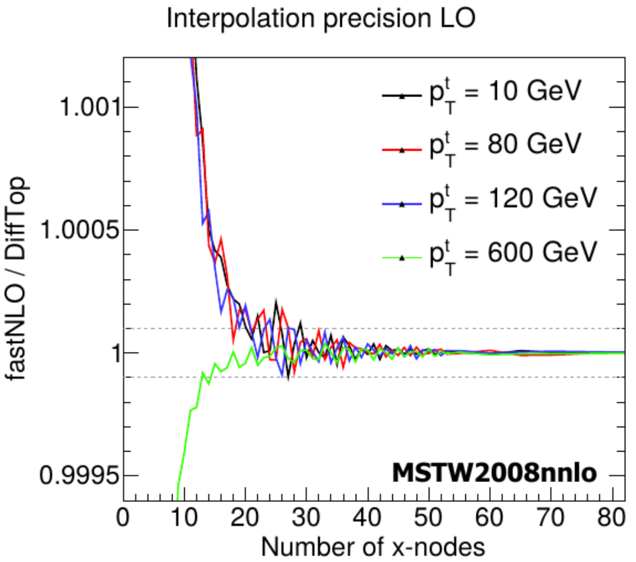
- Result: No systematic bias → only numerical fluctations
- As expected: Forward more difficult than central;
  low-pT more difficult than high-pT (due to large x-range and less nodes per dx)
- 0.1% accuracy typically with about 16-20 nodes

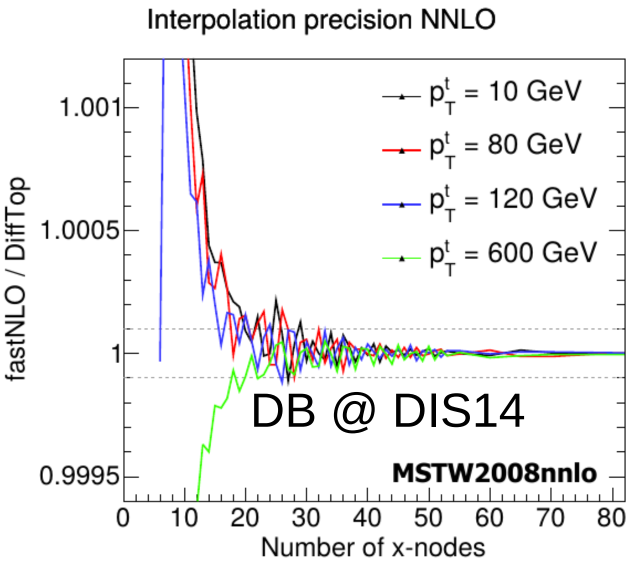
Interpolation accuracy

- Accuracy for fastNLO flexible scale
- aNNLO predictions from DiffTop (Moch, Guzzi)
- Interpolation accuracy depends on number of nodes for the x-grid
- Sub-permille level easily possible
- It is always a trade-off between:

  file-size/speed vs. numerical precision

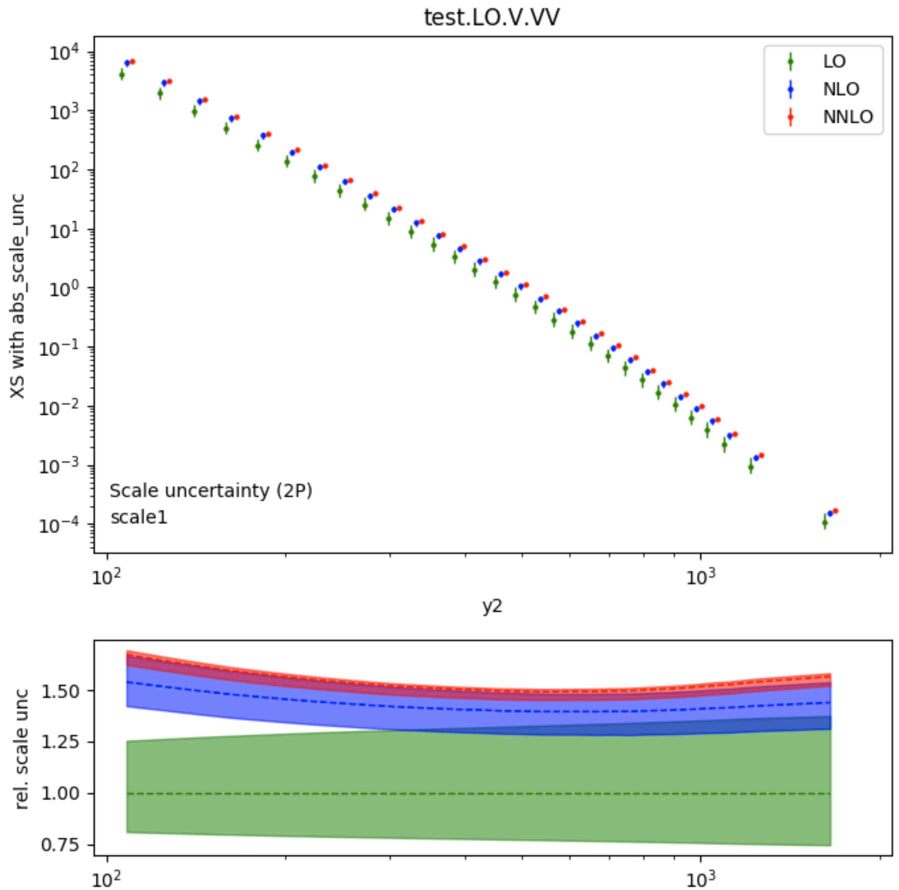
- Caveats:
  A 'dynamic' scale involves an additional grid: negligible deterioration of closure → but still increase in file size by factor 4-16
- 'unpleasant' binning (e.g. huge bins and small bins), or very high y data (high-x convolutes with low-x) is more challenging, due to rapid change of valence PDFs



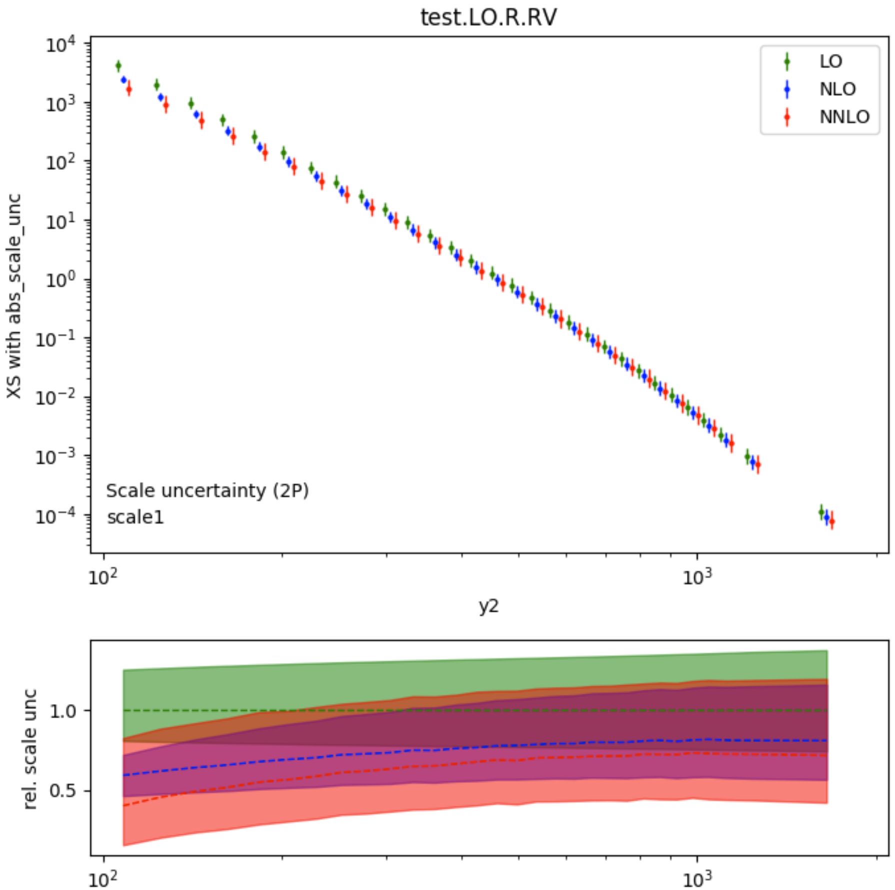


Cross sections with selected contributions

LO + V + VV



LO + R + RV



Scale variations

In NNLO scale variations...

- μ_R variation straight forward (although lenghty)
- μ_F variation computationally more intense than in NLO

Different options for scale variations are available

'Fixed scale'

- Choice for $\mu_{R/F}$ set during grid generation
- Scale variation done by:
  μ_R) apply formulae for μ_R variation (make use of lower order grids)
  μ_F) either generate grids for pre-defined μ_F factors (0.5,1,2), or
  perform variation with Hoppet

'Flexible scale'

- Store scale-independet weights in (additional) grids

$$\omega(\mu_R,\mu_F) = \omega_0 + \underbrace{\log(\mu_R^2)\omega_R + \log(\mu_F^2)\omega_F}_{\text{log's for NLO}} + \underbrace{\log^2(\mu_R^2)\omega_{RR} + \log^2(\mu_F^2)\omega_{FF} + \log(\mu_R^2)\log(\mu_F^2)\omega_{RF}}_{\text{additional log's in NNLO}}$$

- Pro: scales can be chosen rather flexible w/o re-calculation of grid
- Con: larger files and slower convolution, since more grids have to be evaluated

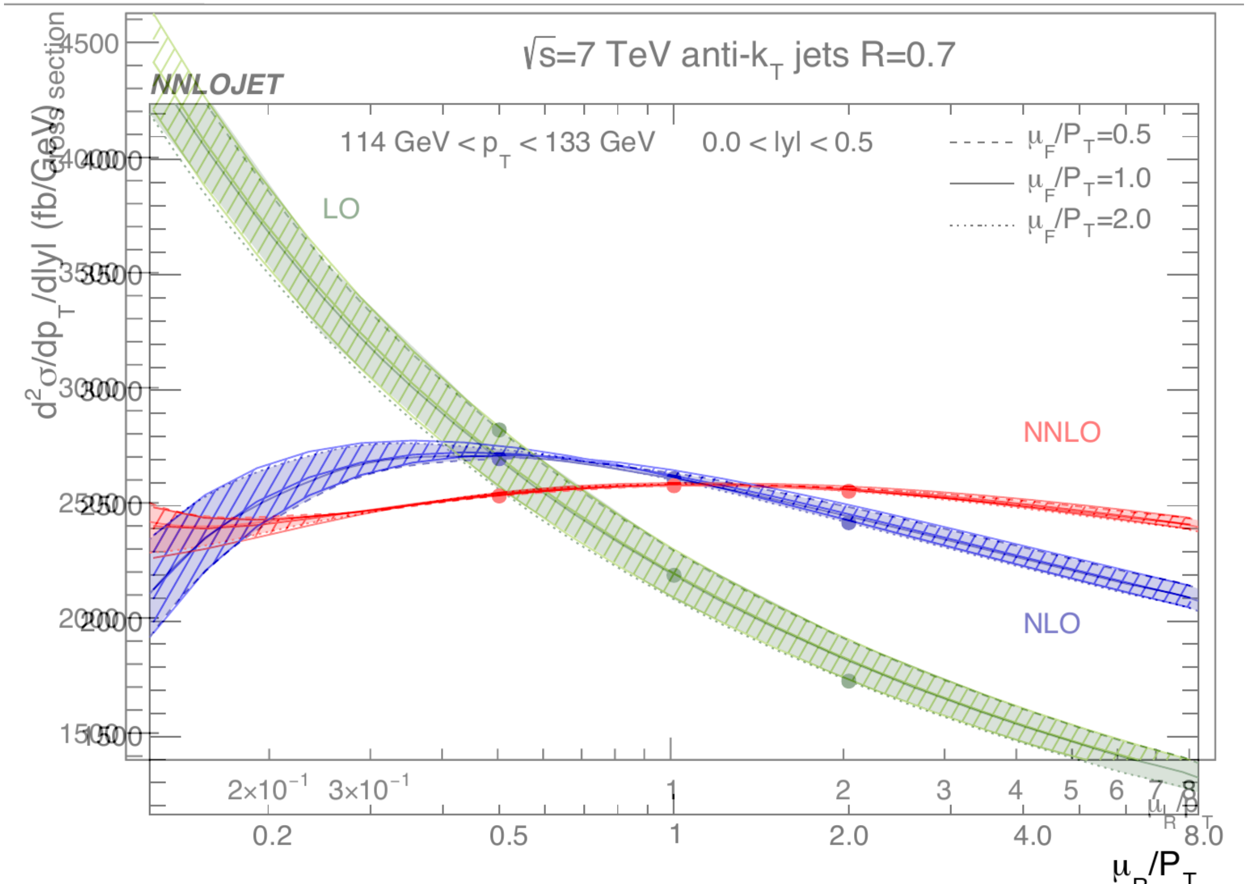
The full scale variation contribution at NNLO

$$\begin{aligned}
&\sigma(\mu_R,\mu_F,\alpha_s(\mu_R),L_R,L_F) = \\
&\left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^2 \hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes f_j(\mu_F) \\
&+\left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^3 \hat\sigma_{ij}^{(1)} \otimes f_i(\mu_F) \otimes f_j(\mu_F) \\
&\quad +L_R \left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^3 2\beta_0\,\hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes f_j(\mu_F) \\
&\quad +L_F \left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^3 \Big[-\hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes \left(P_{jk}^{(0)} \otimes f_k(\mu_F)\right) \\
&\qquad\qquad -\hat\sigma_{ij}^{(0)} \otimes \left(P_{ik}^{(0)} \otimes f_k(\mu_F)\right) \otimes f_j(\mu_F)\Big] \\
&+\left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^4 \hat\sigma_{ij}^{(2)} \otimes f_i(\mu_F) \otimes f_j(\mu_F) \\
&\quad +L_R \left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^4 \left(3\,\beta_0\,\hat\sigma_{ij}^{(1)} + 2\,\beta_1\,\hat\sigma_{ij}^{(0)}\right) \otimes f_i(\mu_F) \otimes f_j(\mu_F) \\
&\quad +L_R^2 \left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^4 3\,\beta_0^2\,\hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes f_j(\mu_F) \\
&\quad +L_F \left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^4 \Big[-\hat\sigma_{ij}^{(1)} \otimes f_i(\mu_F) \otimes \left(P_{jk}^{(0)} \otimes f_k(\mu_F)\right) \\
&\qquad\qquad -\hat\sigma_{ij}^{(1)} \otimes \left(P_{ik}^{(0)} \otimes f_k(\mu_F)\right) \otimes f_j(\mu_F) \\
&\qquad\qquad -\hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes \left(P_{jk}^{(1)} \otimes f_k(\mu_F)\right) \\
&\qquad\qquad -\hat\sigma_{ij}^{(0)} \otimes \left(P_{ik}^{(1)} \otimes f_k(\mu_F)\right) \otimes f_j(\mu_F)\Big] \\
&\quad +L_F^2 \left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^4 \Big[\hat\sigma_{ij}^{(0)} \otimes \left(P_{ik}^{(0)} \otimes f_k(\mu_F)\right) \otimes \left(P_{jl}^{(0)} \otimes f_l(\mu_F)\right) \\
&\qquad\qquad +\frac{1}{2}\hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes \left(P_{jk}^{(0)} \otimes P_{kl}^{(0)} \otimes f_l(\mu_F)\right) \\
&\qquad\qquad +\frac{1}{2}\hat\sigma_{ij}^{(0)} \otimes \left(P_{ik}^{(0)} \otimes P_{kl}^{(0)} \otimes f_l(\mu_F)\right) \otimes f_j(\mu_F) \\
&\qquad\qquad +\frac{1}{2}\beta_0\,\hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes \left(P_{jk}^{(0)} \otimes f_k(\mu_F)\right) \\
&\qquad\qquad +\frac{1}{2}\beta_0\,\hat\sigma_{ij}^{(0)} \otimes \left(P_{ik}^{(0)} \otimes f_k(\mu_F)\right) \otimes f_j(\mu_F)\Big] \\
&\quad +L_F L_R \left(\frac{\alpha_s(\mu_R)}{2\pi}\right)^4 \Big[-3\,\beta_0\,\hat\sigma_{ij}^{(0)} \otimes f_i(\mu_F) \otimes \left(P_{jk}^{(0)} \otimes f_k(\mu_F)\right) \\
&\qquad\qquad -3\,\beta_0\,\hat\sigma_{ij}^{(0)} \otimes \left(P_{ik}^{(0)} \otimes f_k(\mu_F)\right) \otimes f_j(\mu_F)\Big]
\end{aligned}$$

Validation II: scale variations

Scale variations in LO, NLO and NNLO

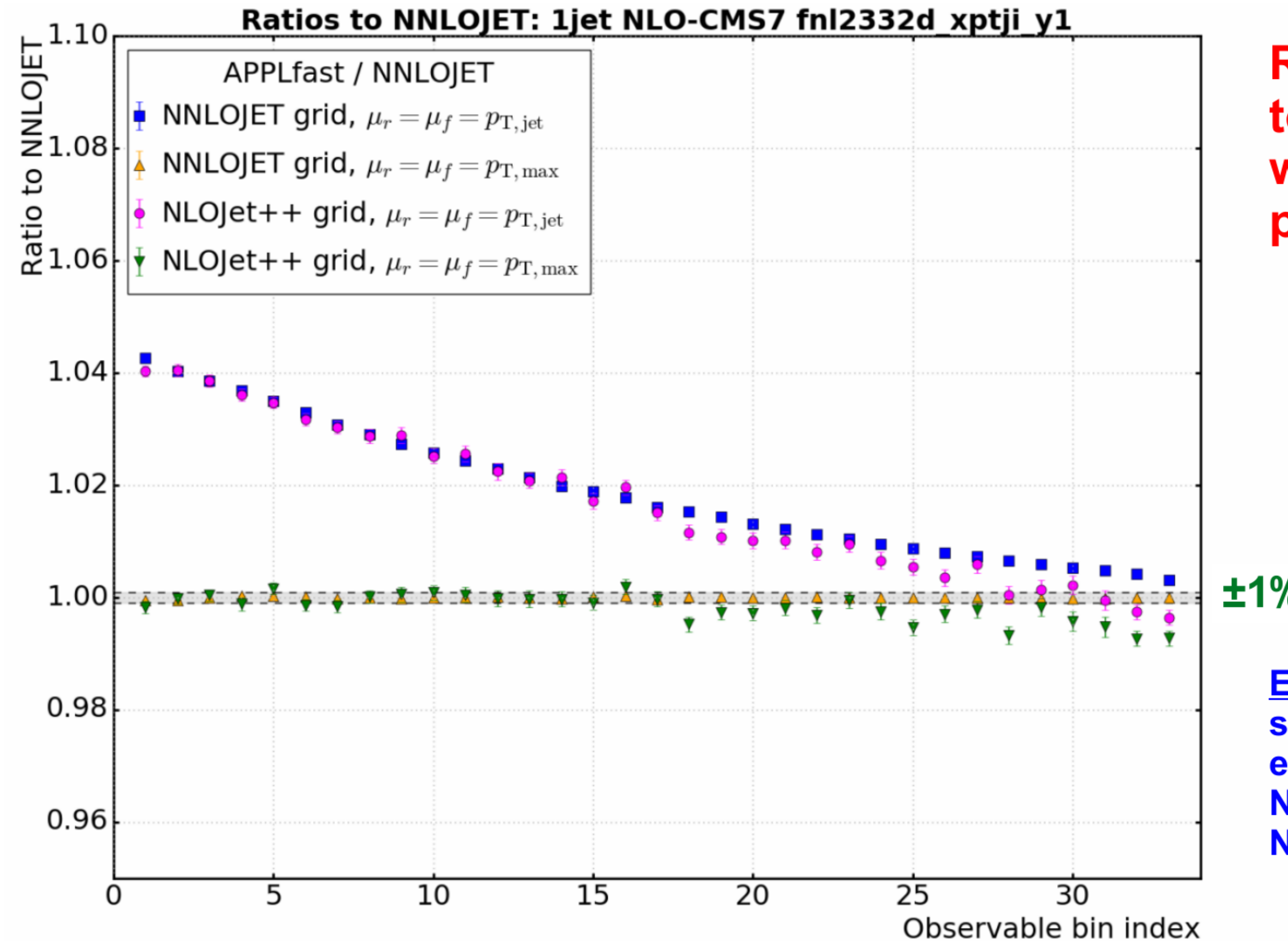
- underlying histogram from NNLOJET publication
- overlayed histogram from freshly produced NNLO grid



Excellent reproduction of native NNLOJET calculation

Validation: (N)NLOJET vs. nlojet++

- Inclusive jets in pp

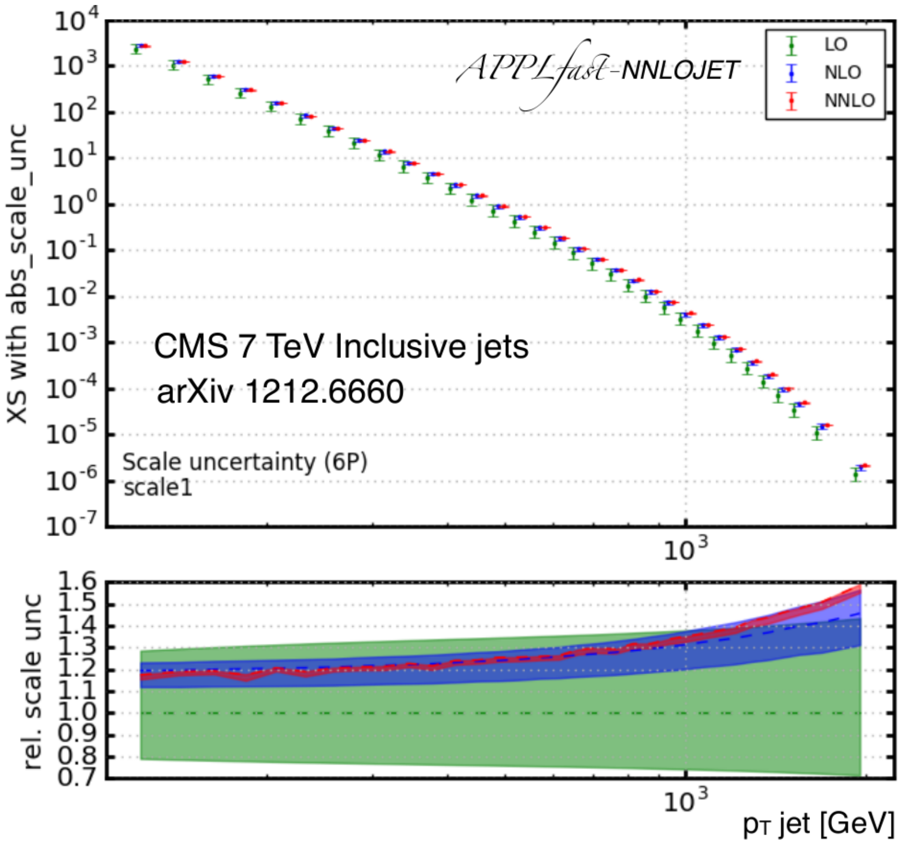


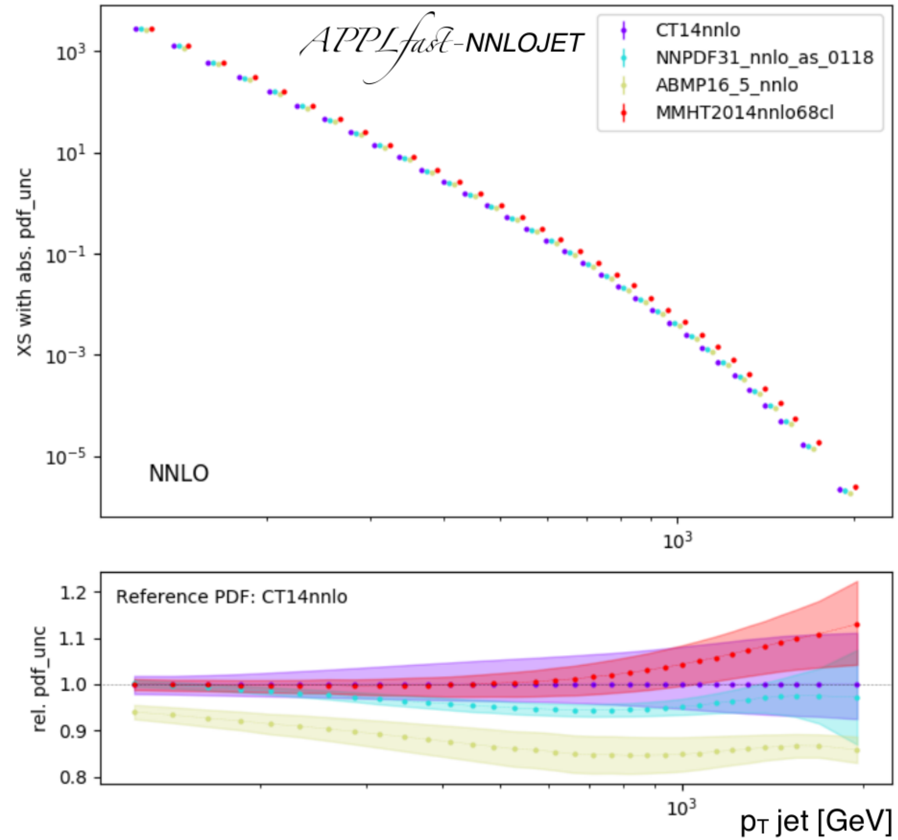
Ratio always to NNLOJET with scale $p_{T,max}$

Validation also has been done for 'ep' jet grids

Error bars:
stat. uncertainty estimate from NNLOJET & NLOJet++

Full NNLO grids





- Grid closure O(0.1 per mille): more accurate than numerics of published data table
- PDF error determinations, and PDF fits are reasonably fast
  Scale variations very fast for all scale-variation concepts

grid distribution

Ploughshare: preface

Preface:
Increasingly, all PDF fitters and experiments use fast interpolation grid

- <u>Many</u> grids are available from <u>many different</u> groups …
- These grids commonly require a large amount of CPU and personal effort to produce
- not always, the code for grid generation itself is public (e.g. NNLO top production, NNLOJET)
- LHC/HERA Experiments produce grids in conjuction with the APPLGrid/fastNLO developers

  → these do not have common repositories for their grids...

In view of the new 'NNLO grids'

- significant more commitment in terms of CPU resources is required
- It is still not clear, which grids are available, and from whom they can be obtained

→ it would be useful if there were some way to share the grids….

Ploughshare: project description

Project objective

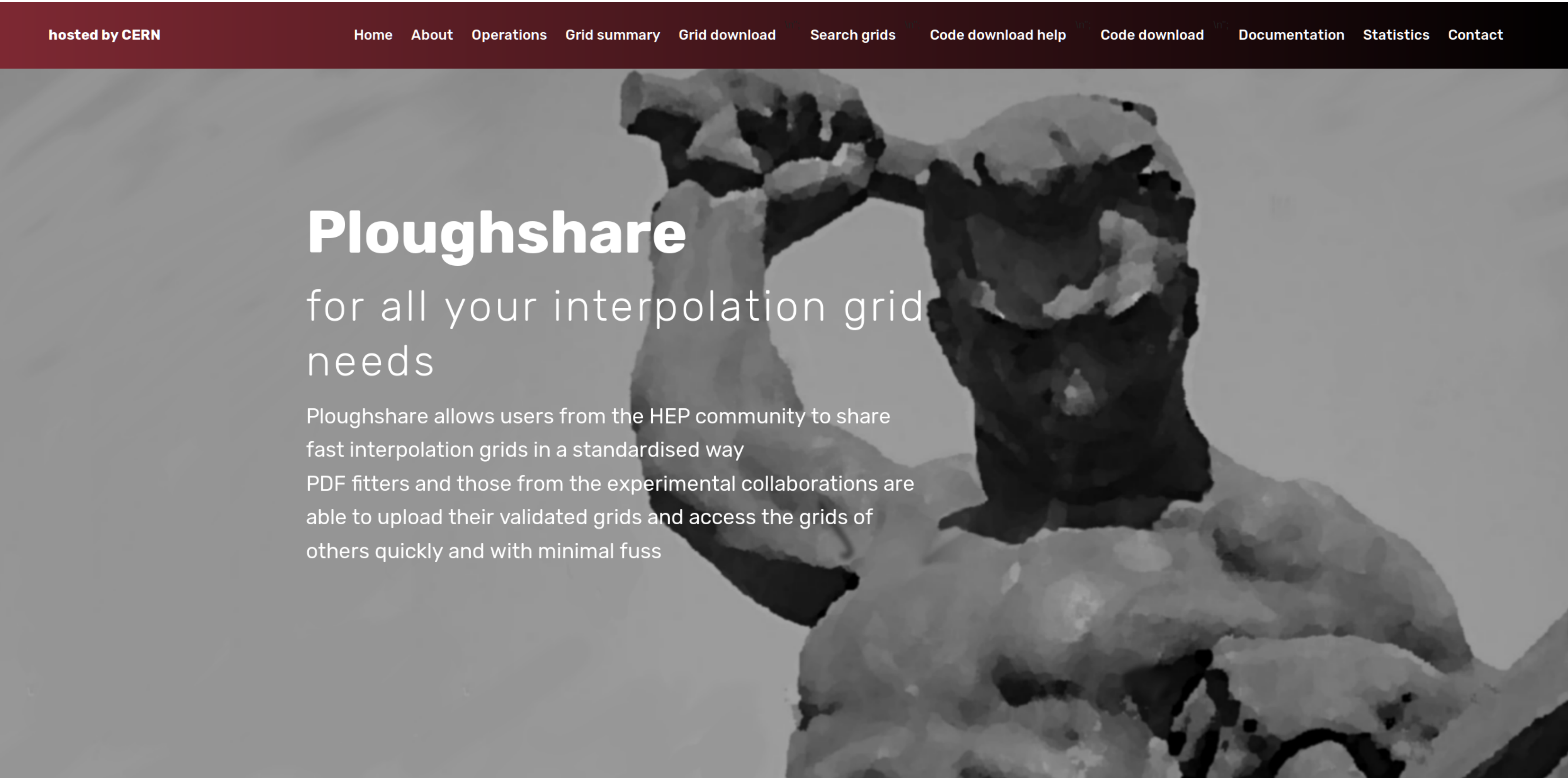
- provide a standardised grid repository
  and distribution for all available grids from all groups

Common repository

- grids are available from a single webpage
  - standardised meta-tags
  - searchable webpage
- Possibility to commit new grids through a standardised interface (for registered persons)
  - Very convenient, and easy to 'commit' updates...
- not only http download available:
  basic utility library (C++,Fortran) is available:
  → Automatic download, handling, and caching of grids within any program

http://ploughshare.web.cern.ch
Main developer: Mark Sutton

http://ploughshare.web.cern.ch



Presently publically available grids

Grid summary

- Experiment, collision, energy
- Process: ZJ, jets, Z0, ...
  we try to follow a common naming convention
- Calculation, generator
  NNLOJET, nlojet++, MCFM, aMCfast…
- Group
  the provider of the grid
- arXiv
  standard reference for the publication
- Download link
  APPLgrid, fastNLO

As of today: a few example grids…
but ploughshare is now fully operational!

Home About Operations Grid summary Grid download Search grids Code download help

Summary of available grid datasets

Here are all the uploaded and available grids. The grid database can be searched using the search facility here

Experiment	Collision	Energy	Process	Calculation	Group	arxiv	tgz file direct link
ATLAS	pp	7 TeV	1jet-R06-dev-fn	NNLOJET	APPLfast	1410.8857	applfast-atlas-1jet-r06-dev-fn-arxiv-1410.8857
ATLAS	pp	8 TeV	ZJ-dev-ap	NNLOJET	APPLfast	1512.02192	applfast-atlas-zj-dev-ap-arxiv-1512.02192
ATLAS	pp	8 TeV	ZJ-dev-fn	NNLOJET	APPLfast	1512.02192	applfast-atlas-zj-dev-fn-arxiv-1512.02192
CMS	pp	7 TeV	1jet-ptj-dev	NNLOJET	APPLfast	1212.6660	applfast-cms-1jet-ptj-dev-arxiv-1212.6660
H1	ep	319 GeV	ptj-dev	NNLOJET	APPLfast	1406.4709	applfast-h1-ptj-dev-arxiv-1406.4709
ATLAS	pp	7 TeV	dijets	NLOjet++	ATLAS	1112.6297	atlas-atlas-dijets-arxiv-1112.6297
ATLAS	pp	7 TeV	dijets	NLOjet++	ATLAS	1312.3524	atlas-atlas-dijets-arxiv-1312.3524
ATLAS	pp	7 TeV	incljets	NLOjet++	ATLAS	1009.5908v2	atlas-atlas-incljets-arxiv-1009.5908v2
ATLAS	pp	7 TeV	incljets	NLOjet++	ATLAS	1112.6297	atlas-atlas-incljets-arxiv-1112.6297
ATLAS	pp	2.76 TeV	incljets	NLOjet++	ATLAS	1304.4739	atlas-atlas-incljets-arxiv-1304.4739
ATLAS	pp	7 TeV	incljets	NLOjet++	ATLAS	1410.8857	atlas-atlas-incljets-arxiv-1410.8857
ATLAS	pp	7 TeV	Wpm	MCFM	ATLAS	1109.5141	atlas-atlas-wpm-arxiv-1109.5141
ATLAS	pp	7 TeV	Wpm	MCFM	ATLAS	1612.03016	atlas-atlas-wpm-arxiv-1612.03016
ATLAS	pp	7 TeV	Z0	MCFM	ATLAS	1109.5141	atlas-atlas-z0-arxiv-1109.5141
ATLAS	pp	7 TeV	Z0	MCFM	ATLAS	1612.03016	atlas-atlas-z0-arxiv-1612.03016
CMS	pp	13 TeV	dijets	NLOjet++	fastNLO	1703.09986	fastnlo-cms-dijets-arxiv-1703.09986
CMS	pp	2.76 TeV	incjets	NLOjet++	fastNLO	1512.06212	fastnlo-cms-incjets-arxiv-1512.06212
CMS	pp	13 TeV	incjets	NLOjet++	fastNLO	1605.04436	fastnlo-cms-incjets-arxiv-1605.04436
H1	dis	920 GeV	jets	DISENT	fastNLO	1406.4709	fastnlo-h1-jets-arxiv-1406.4709

Full analysis records

Measurement of inclusive jet and dijet production in pp collisions at $\sqrt{s} = 7$ TeV using the ATLAS detector

Inclusive jet and dijet cross sections have been measured in proton-proton collisions at a centre-of-mass energy of 7 TeV using the ATLAS detector at the Large Hadron Collider. The cross sections were measured using jets clustered with the anti-kT algorithm with parameters R=0.4 and R=0.6. These measurements are based on the 2010 data sample, consisting of a total integrated luminosity of 37 inverse picobarns. Inclusive jet double-differential cross sections are presented as a function of jet transverse momentum, in bins of jet rapidity. Dijet double-differential cross sections are studied as a function of the dijet invariant mass, in bins of half the rapidity separation of the two leading jets. The measurements are performed in the jet rapidity range |y|<4.4, covering jet transverse momenta from 20 GeV to 1.5 TeV and dijet invariant masses from 70 GeV to 5 TeV. The data are compared to expectations based on next-to-leading order QCD calculations corrected for non-perturbative effects, as well as to next-to-leading order Monte Carlo predictions. In addition to a test of the theory in a new kinematic regime, the data also provide sensitivity to parton distribution functions in a region where they are currently not well-constrained.

journal: Phys.Rev. D86 (2012) 014022 (doi: 10.1103/PhysRevD.86.014022)
arxiv: 1112.6297
inspire: https://inspirehep.net/record/1082936
HepData: https://hepdata.net/record/ins1082936

link to full tarball

links to specific hep data table

Experiment	Physics process	Beam energy	Calculation	direct link	
ATLAS	pp	7 TeV	NLOjet++	atlas-atlas-dijets-arxiv-1112.6297 tarball	
				atlas-atlas-dijets-arxiv-1112.6297-xsec000.root	:: Table 19 : d2sigma/dm_{12}dy* [pb/TeV], Anti-kT R=0.4, 0.0<y*<0.5
				atlas-atlas-dijets-arxiv-1112.6297-xsec001.root	:: Table 20 : d2sigma/dm_{12}dy* [pb/TeV], Anti-kT R=0.4, 0.5<y*<1.0
				atlas-atlas-dijets-arxiv-1112.6297-xsec002.root	:: Table 21 : d2sigma/dm_{12}dy* [pb/TeV], Anti-kT R=0.4, 1.0<y*<1.5
				atlas-atlas-dijets-arxiv-1112.6297-xsec003.root	:: Table 22 : d2sigma/dm_{12}dy* [pb/TeV], Anti-kT R=0.4, 1.5<y*<2.0
				atlas-atlas-dijets-arxiv-1112.6297-xsec004.root	:: Table 23 : d2sigma/dm_{12}dy* [pb/TeV], Anti-kT R=0.4, 2.0<y*<2.5
				atlas-atlas-dijets-arxiv-1112.6297-xsec005.root	:: Table 24 : d2sigma/dm_{12}dy* [pb/TeV], Anti-kT R=0.4, 2.5<y*<3.0
				atlas-atlas-dijets-arxiv-1112.6297-xsec006.root	:: Table 26 : d2sigma/dm_{12}dy* [pb/TeV], Anti-kT R=0.4, 3.5<y*<4.0

links to individual grids

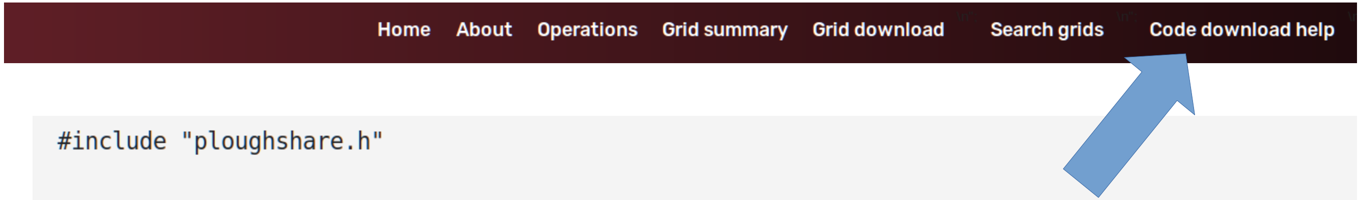
optional grid desciption

- Full title and abstract
- Links to …
  - Journal paper (DOI)
  - Preprint
  - Inspire
  - HepData
  - Table with all available grids

- This information is determined automatically from the required preprint ID when you upload grids
- For grids with no corresponding paper, a dummy arrive number arxiv: 0000.00000 can be used
- Users will be able to provide an additional HTML fragment in the tgz file if they require
- In the case of no available preprint the HTML fragment will be used as the analysis record …

Ploughshare: API

Ploughshare API

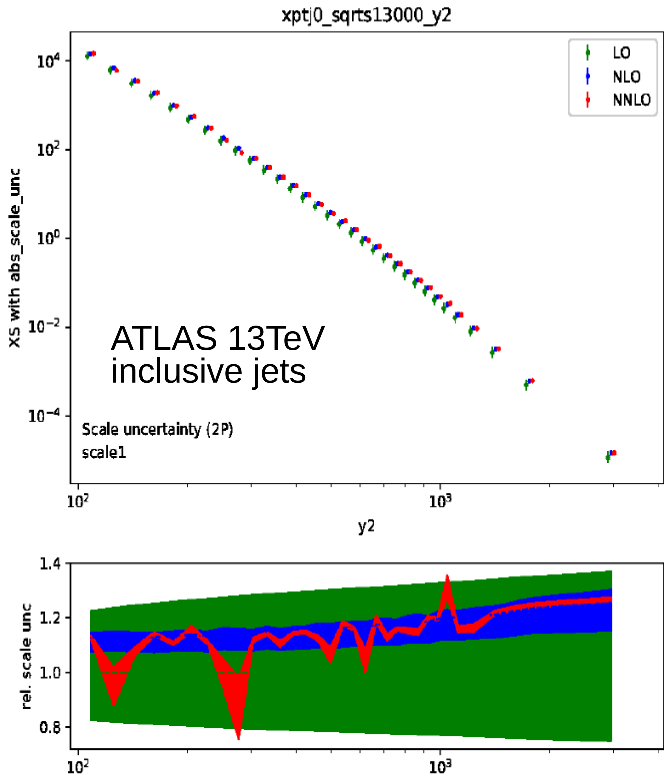


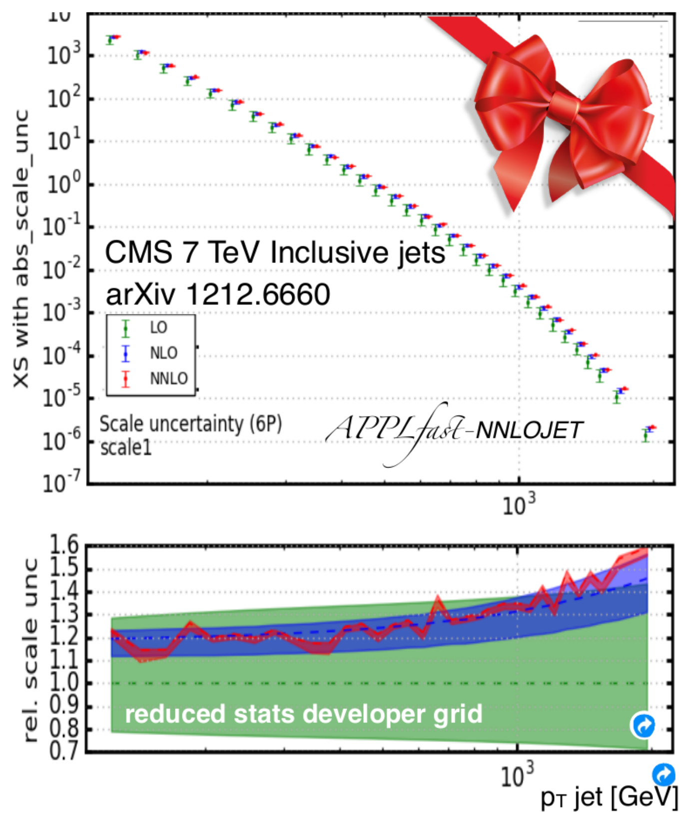
- Grid download, management and caching from ploughshare webpage
- Fortran and C++ interface
- More details: https://indico.cern.ch/event/761343/contributions/3226735/attachments/1770707/2877252/sutt-ploughshare-pdf4lhc.pdf
- Ploughshare API could be an interesting option for xfitter: works for APPLgrid and fastNLO

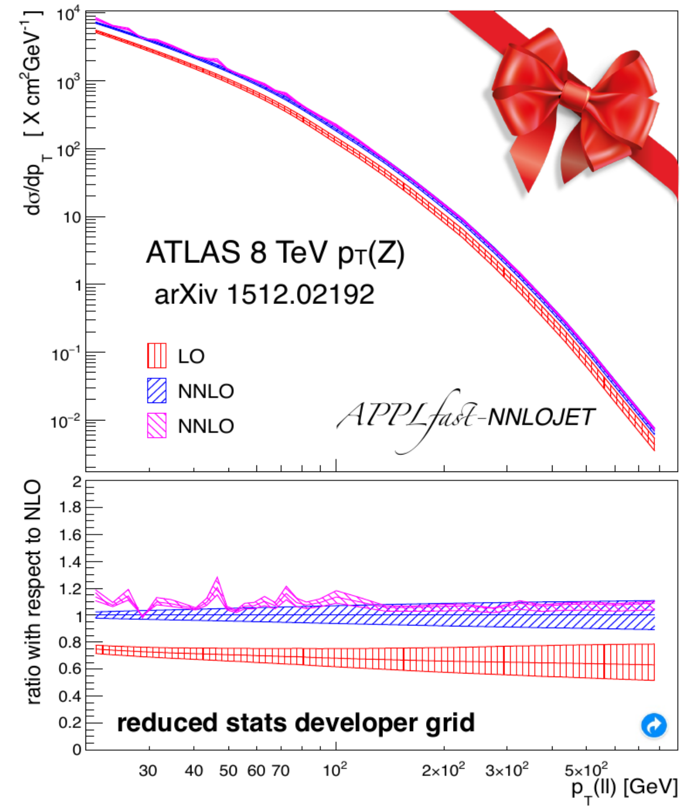
```
#include "ploughshare.h"

...

  ploughshare p;

  p.fetch("atlas-atlas-wpm-arxiv-1612.0.0.13")

  std::cout << "dataset grids: " << p.grids("atlas-atlas-wpm-arxiv-1612.0.0.13") << std::endl;

  std::vector grids = p.grids("atlas-atlas-z0-arxiv-1109.5141");

  for ( size_t i=0 ; i<grids.size() ; i++ ) {

    /// open grids as normal ...
    appl::grid g(grids[i]);

    ...

  }
```

```
ploughshare::dataset_t  ds = p.dataset("atlas-atlas-dijets-arxiv-1312.3524");
std::cout << "dataset path:  " << ds.path() << std::endl;
std::cout << "dataset grids: " << ds.grids() << std::endl;
```

Publically available 'dev'-grids







For development and testing purposes

- Grids for a number of processes are made available (see following slides)
- Grids have by purpose limited statistics, or missing channels
  → only reasonable for technical developments

Summary

APPLgrid and fastNLO grids in NNLO

- grids production is ongoing

ep → jets

- grids for all inclusive-jet and dijet cross sections at HERA available
  → they will be made public ~ spring 2019

pp → jets

- grids are being produced…
- first full stat. grids are currently validated
  - scale variations
  - closure with 'native' NNLOJET calculation
  - 'closure' of NNLOJET calculation itself is also tested

→ low-stat. grids are presently publically available for testing → We need feedback !

pp → anything else (Z,Z+jets,...)

- Grids can be produced on request

Ploughshare may be used for distribution of grids

New conversion tool has been developed to convert between formats

- Storage of scale-independent weights enable full scale flexibility also in NNLO
  - Additional logs in NNLO

$$\omega(\mu_R,\mu_F) = \underbrace{\omega_0 + \log(\mu_R^2)\omega_R + \log(\mu_F^2)\omega_F}_{\text{log's for NLO}} + \underbrace{\log^2(\mu_R^2)\omega_{RR} + \log^2(\mu_F^2)\omega_{FF} + \log(\mu_R^2)\log(\mu_F^2)\omega_{RF}}_{\text{additional log's in NNLO}}$$

  - Store weights: w_0, w_R, w_F, w_{RR}, w_{FF}, w_{RF} for order α_s^{n+2} contributions
- Advantages
  - Renormalization and factorization scale can be varied *independently* and by *any* factor
    - No time-consuming 're-calculation' of splitting functions in NLO necessary
  - Only small increase in amount of stored coefficients
- Implementation
  - *Two* different observables can be used for the scales
    - e.g.: H_T and $p_{T,max}$
    - or e.g.: p_T and $|y|$
    - …
  - *Any function* of those *two observables* can be used for calculating scales

Detailed closure tests

Example: ATLAS 13TeV incl. jets, R=0.6

- Closure of individual contributions
  - R (real,NLO)
  - VV (double-virtual, NNLO)

  - RRa (double-real (part), NNLO)
    2.5e8 events, >50k CPU hours,
    3T output files, 2560 jobs