QCDNUM Status and Plans

Michiel Botje

xFitter external workshop

Orsay, March 9, 2022

QCDNUM-17-00/08

Download
How to install
Write-up
Example jobs
Release history

QCDNUM-18-00/00

Download
How to install
Write-up
C++ interface
Example jobs
Release history

Packages

MBUTIL
WSTORE
ZMSTF
HQSTF
SPLINT

QCDNUM

Known bugs
Talks
Contact

QCDNUM releases

- QCDNUM-17-01/15 : latest release October 31, 2019
  - No change in QCDNUM code since then
- QCDNUM-18-00/00 : released yesterday March 8, 2022
  - Fully backward compatible with 17-01/15
  - Some fixes of warnings/errors compiler CGG 11.2.0
  - All QCDNUM code is now interfaced to C++ (toolbox was missing)
  - All write-ups are now collected in the doc directory
  - See README for a few changes in running test jobs
  - Added ZMSTFIJ routine in ZMSTF (structure function on grid point)
  - SPLINT package to create cubic interpolation splines
  - WSTORE memory manager for future QCDNUM (now used in SPLINT)
  - Stable release: only bug fixes in the future

New memory package wstore

- ✓ Routine to convert 1-dim double precision array into a wstore (formatting)
- ✓ Routines to create table-sets and populate them with one or more n-dim tables
- ✓ Routines to clone, copy, disk dump and read tables and table-sets
- ✓ Object fingerprinting (equal fingerprint = equal object structure)
- ✓ Easy and fast navigation through linked-list structure
- ✓ Each object has a tag-field to store attributes
- ✓ Can build object hierarchies by storing addresses (pointers) in the tag-fields
- ✓ Hooks to create very fast iterators and address functions
- ✓ Successfully implemented in SPLINT

Cubic interpolation splines with SPLINT

- Piecewise cubic polynomial that agrees with some $f(x)$ at a set of node points x_i
- Node points in SPLINT are a selection of QCDNUM evolution grid points
- Function $f(x_i)$ is provided by the user
- Can do 1-dim and 2-dim splines
- Uses WSTORE memory manager to dynamically generate spline objects

```
ssp_spinit(nuser);                          //initialise
ia = isp_s2make(istepx,istepq);             //new spline object
ssp_s2fill(ia,sfun,rs);                     //sfun → spline
val = dsp_funs2(ia,x,q,ichk);               //compute spline
val = dsp_ints2(ia,x1,x2,q1,q2);            //integrate spline
```

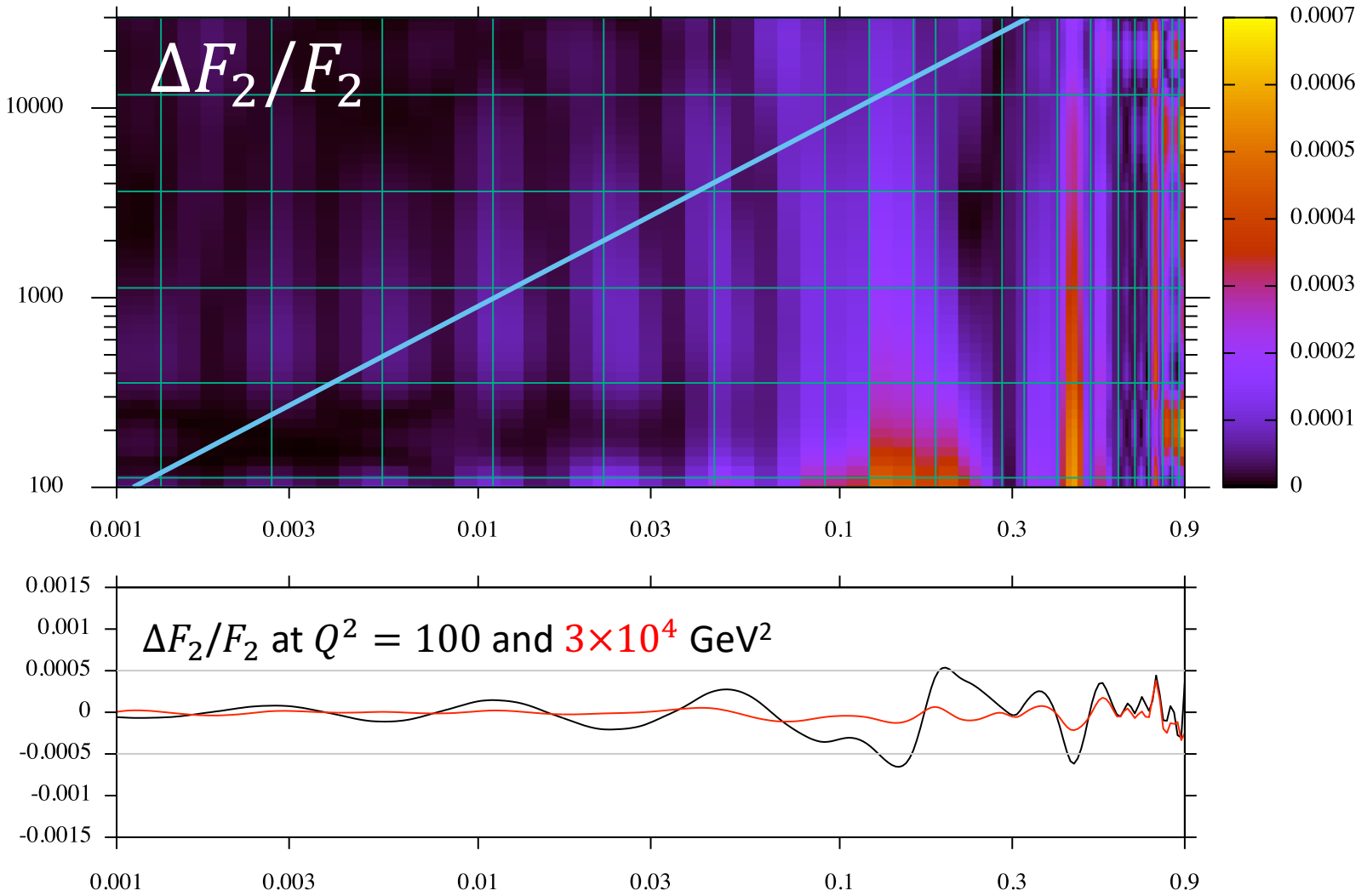
- Take every nth evolution grid-point as node of the spline (`istep`)
- The memory address `ia` is an array index and not a C++ pointer

Developments in 2021 for the BAT group

- Bayesian Analysis Tools group led by Allen Caldwell (Munich)
- Bayesian pdf fits using DESY data at $x > 10^{-3}$ and $100 < Q^2 < 3{\times}10^4$ GeV^2
- Needs integration of cross-section to predict event numbers in 430 x-Q^2 bins
- Wrote new add-on package SPLINT providing very fast integration of cubic splines
- The spline integration is presently implemented in the BAT analysis code
- BAT uses the JULIA language → QCDNUM now also becomes available in JULIA
- Will show some results of the QCDNUM/SPLINT tuning for BAT
- Complete story can be found on the talks page of the QCDNUM web site

BATUNE: proton $\Delta F_2/F_2$ on a 25×7 node grid

- Spline F_2 on a coarse grid and compare to F_2 on a very dense grid
- Step-x = 5 and step-q = 10 gives 25×7 nodes
- $\Delta F_2/F_2 < 5\times10^{-4}$ almost everywhere
- Node tuning also OK for splines of F_L and xF_3



BATUNE: Structure function splines for xsec (25×7 nodes)

- F_L for uptype and downtype $\sum(q+\bar{q})$
- F_2 for uptype and downtype $\sum(q+\bar{q})$
- xF_3 for uptype and downtype $\sum(q-\bar{q})$

3 msec for 6
25×7 splines

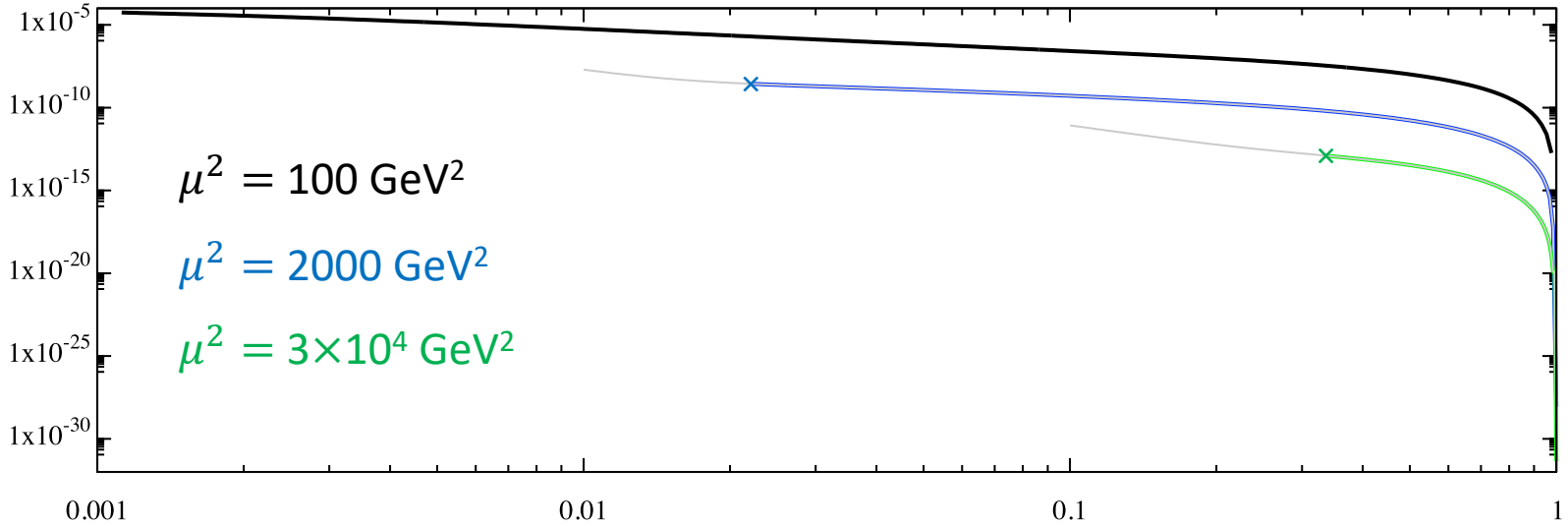
Simplified cross-section

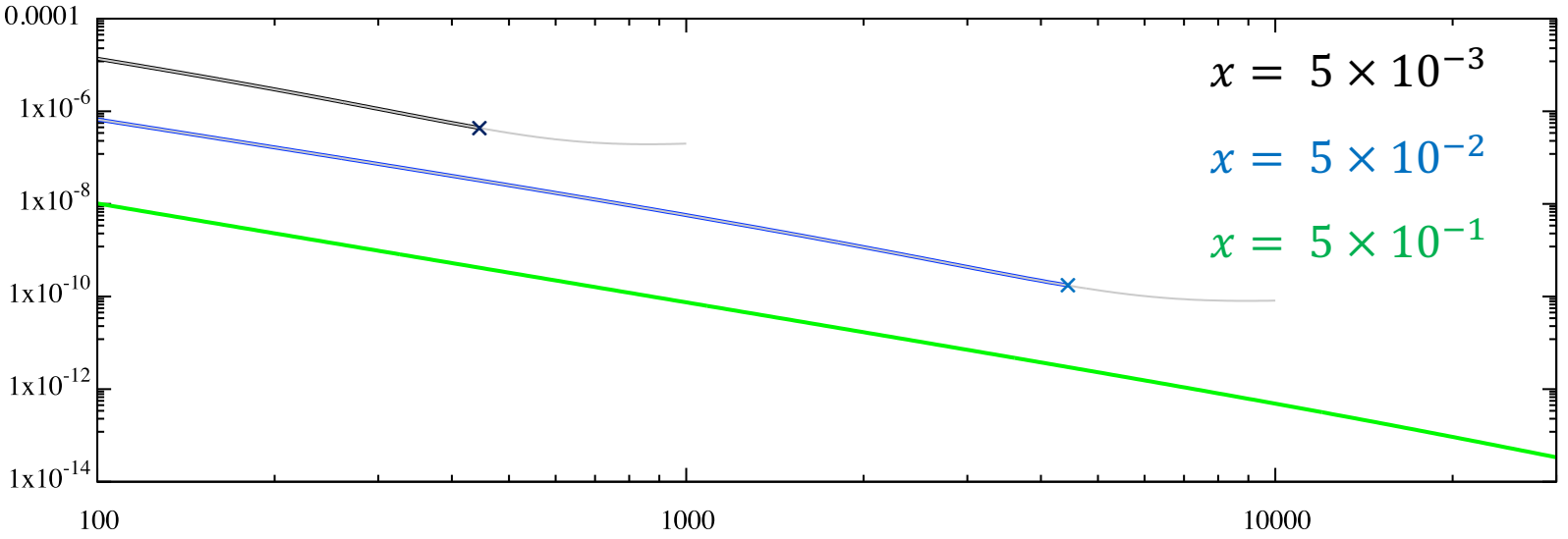
$$\begin{aligned} F_{2,\mathrm{L}} &= \tfrac{1}{9}F_{\mathrm{dtype}} + \tfrac{4}{9}F_{\mathrm{utype}} \\ xF_3 &= 0 \end{aligned} \qquad \begin{aligned} y &= Q^2/xs \\ Y_\pm &= 1 \pm (1-y)^2 \end{aligned}$$

$$\frac{\mathrm{d}^2\sigma}{\mathrm{d}x\mathrm{d}Q^2} = \frac{2\pi\alpha^2}{xQ^4}\left[Y_+F_2 - y^2F_{\mathrm{L}} + Y_-\,xF_3\right]$$

BATUNE: compute (simplified) cross-section

- Very steep fall-off over several orders of magnitude (cannot use reduced xsec for integration)
- Such steep dependence cannot be splined accurately on a coarse node-grid
- The grey curves show that the cross-section extrapolates smoothly beyond the kinematic cut → no further action required
- Next: find optimal nodes for cross-section spline



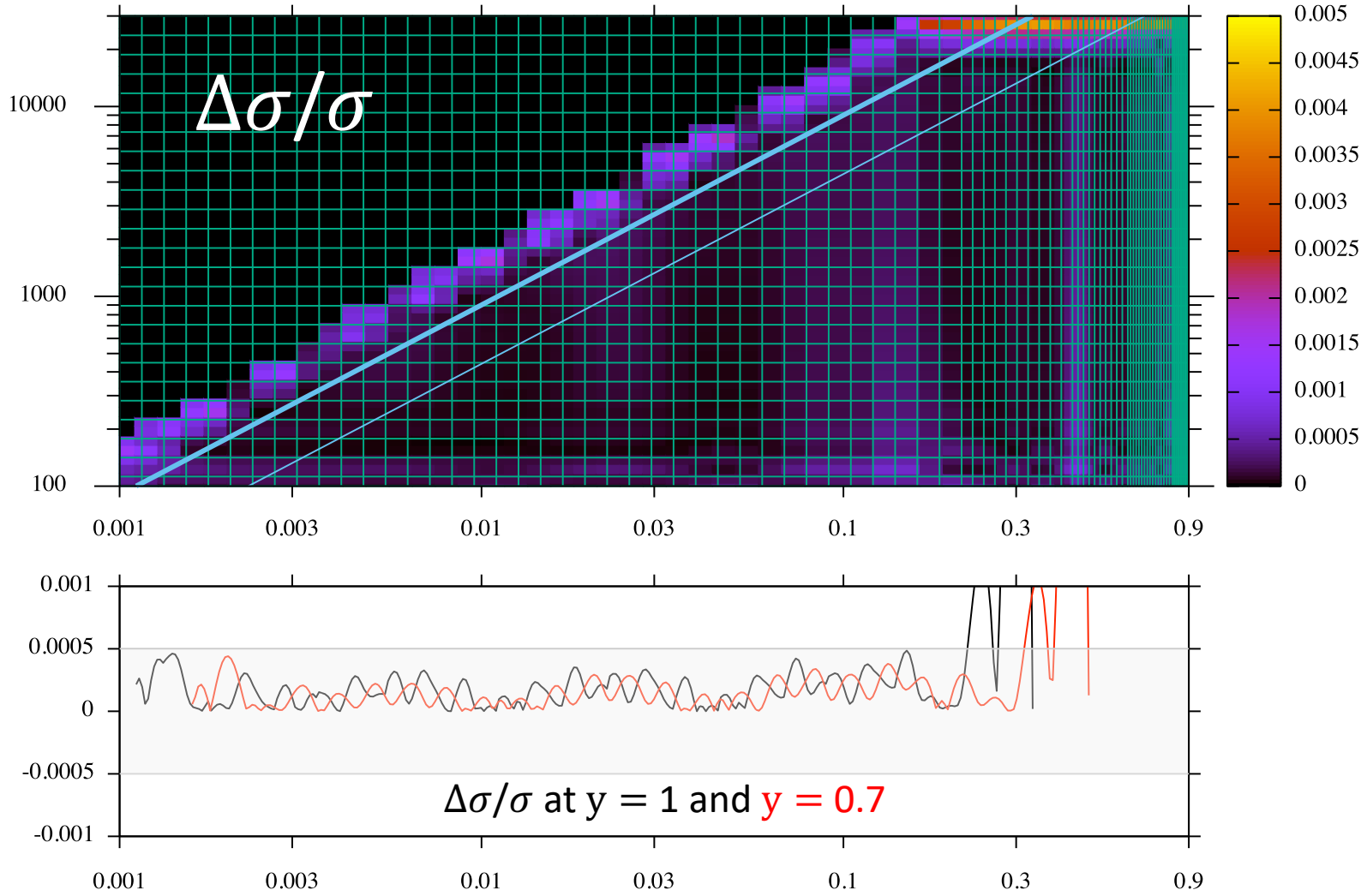


Cross-section spline with 100×25 nodes

- Set step-x = 1 and step-q = 2
- Filling cut at $\sqrt{s}$ = 370 GeV
- $\Delta\sigma/\sigma \lesssim 5\times10^{-4}$ along the kinematic cut ($y = 1$)

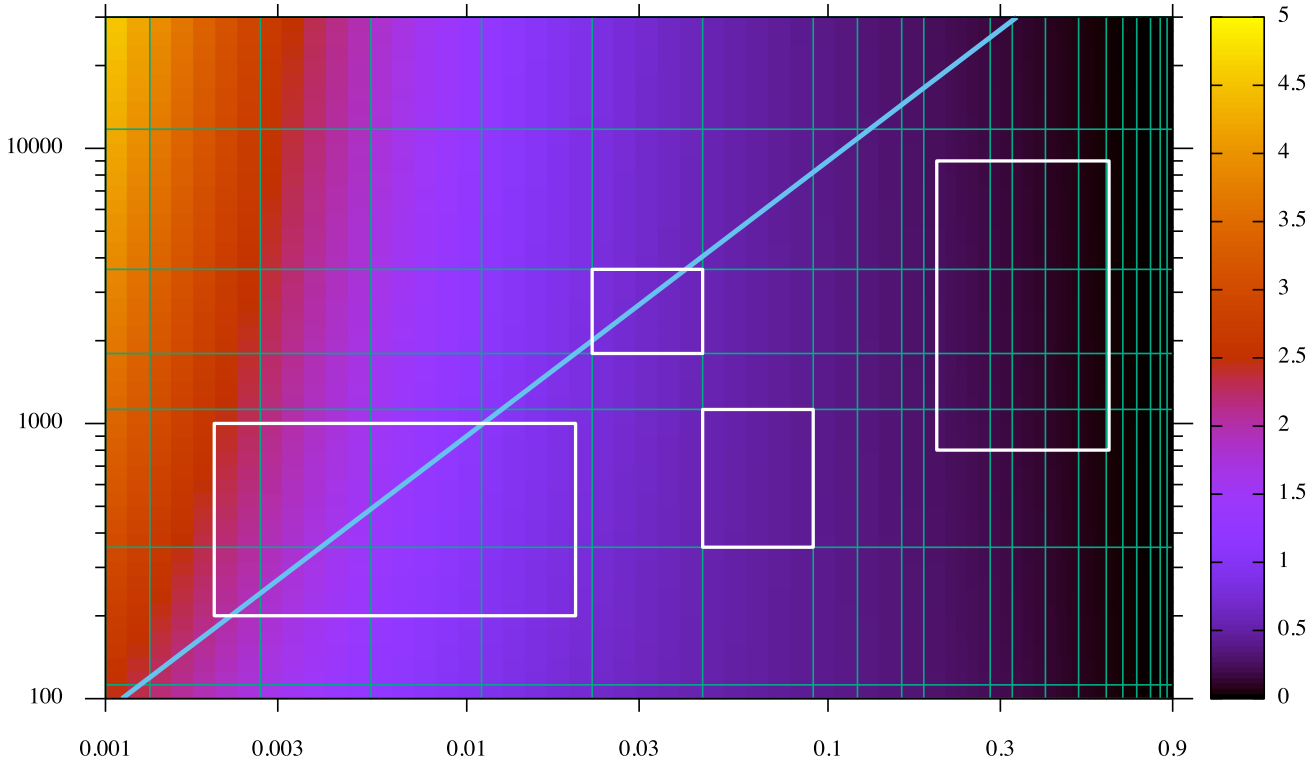
Timing (optimised code)

	n_x	n_q	t [ms]
Evolution	100	50	3.6
6 Stf splines	22	7	2.9
	100	50	4.5
Xsec spline	100	25	2.2
	50	25	1.2



Check of the 2-dimensional spline integration

- Extend a 1-dim n-point Gauss quadrature routine to 2-dim and compare to SPLINT
- For the test, integrate F_2 over single node-bins or arbitrary rectangles, with and without crossing the kinematic limit
- Relative difference $\lesssim 10^{-9}$ in all cases
- SPLINT is a factor 300 faster than Gauss



QCDNUM performance in BAT analysis

- SPLINT casts QCDNUM results into 2-dim cubic splines
- Needs some grid tuning to optimise speed versus accuracy
- BAT code: Evolution → 6 structure function splines → cross-section spline with $\Delta\sigma/\sigma < 5\times10^{-4}$ → integrate 430 bins
- Integration correctly handles bins crossed by kinematic limit

	n_x	n_q	t [ms]
Evolution	100	50	3.6
6 Stf splines	22	7	2.9
Xsec spline	100	25	2.2
Integration			0.8

- Splines can also be a nice CPU saver in bulk computations
- Timing test: 1000 F_2 in NNLO → measure CPU time needed
- Loop with point-by-point F_2 from ZMSTF is very slow
- ZMSTF list processing very fast but not always easy to implement
- Loop with F_2 from spline → easy and 4 times faster than list

Routine	Mode	t [s]
ZMSTF	loop	0.437
ZMSTF	list	0.004
SPLINT	loop	0.001

Stable version QCDNUM-18-00/00 and beyond

QCDNUM	Same as 17-01/15 but with a few minor fixes and toolbox C++ interface
MBUTIL	Added several new routines and C++ interfaces
WSTORE	New memory manager to be implemented in QCDNUM-19
SPLINT	New cubic interpolation spline package (already uses WSTORE)
ZMSTF	Added grid-point routine ZMSTFIJ 3x faster than ZMSTFUN
HQSTF	No changes
Distribution	See README for a few changes in running test jobs All write-ups are now collected in /doc directory

- Base new QCDNUM-19 code on WSTORE memory manager
- Implies large code re-organisation with memory objects very much like C++ objects
- Memory objects will be highly customisable → easy to adapt QCDNUM to your needs
- Release of QCDNUM-19 will take a while …