QCDNUM Status and Plans

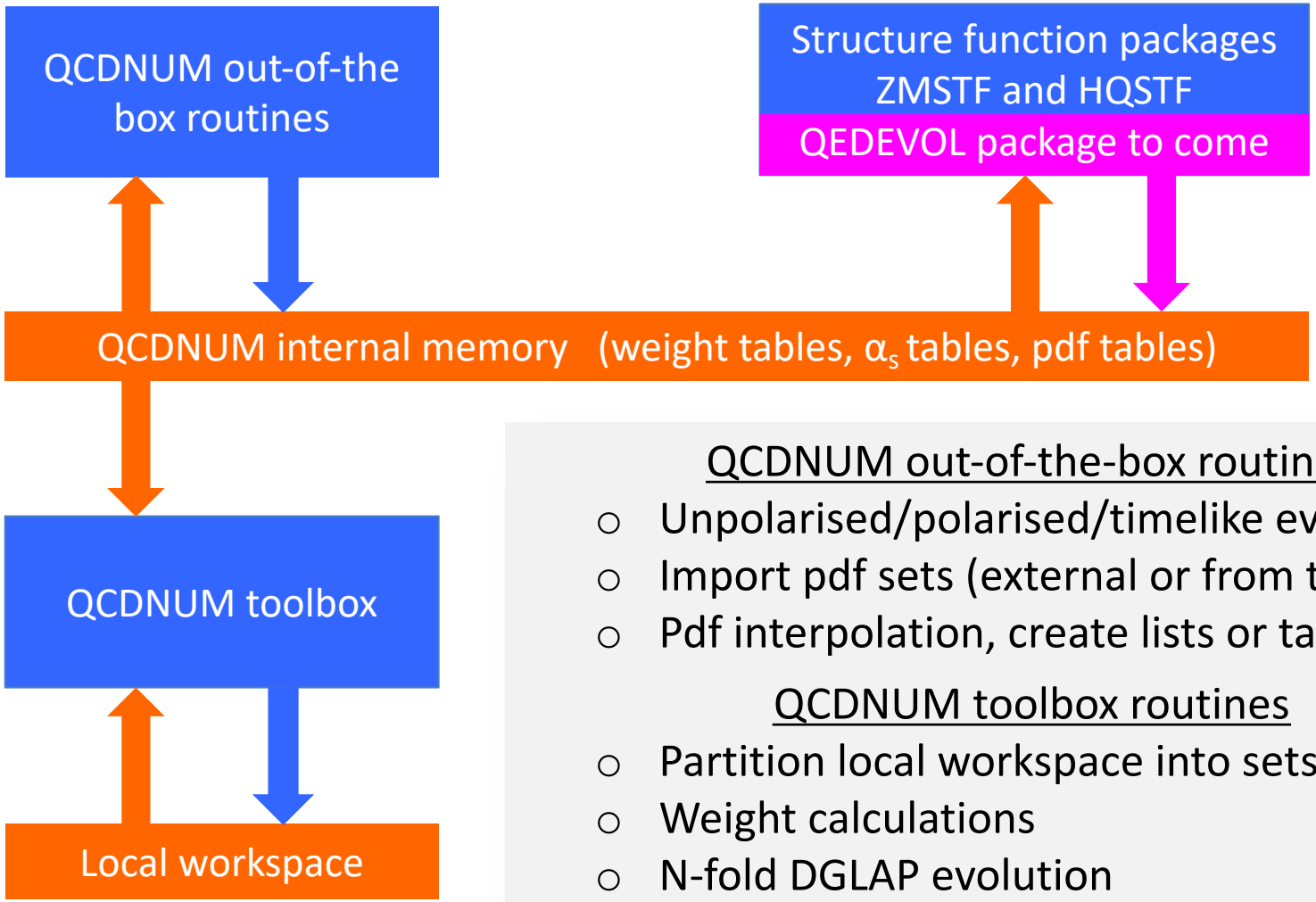
Michiel Botje

Nikhef, Amsterdam

xFitter users meeting

Oxford March 21, 2017

QCDNUM Program Structure



QCDNUM out-of-the-box routines

- Unpolarised/polarised/timelike evolution
- Import pdf sets (external or from toolbox)
- Pdf interpolation, create lists or tables

QCDNUM toolbox routines

- Partition local workspace into sets of tables
- Weight calculations
- N-fold DGLAP evolution
- Convolution tools, fast convolution engine

Is QCDNUM fast?

- Mimic a fit by 1000 evolutions and 10^6 F_2 and F_L calculations
  - VFNS NNLO on a 5-fold 100×60 x-Q^2 grid in the HERA kinematic range
  - MacBook Pro (2012) with 2.5 GHz Intel core i5 and 8 GB RAM
  - Code compiled with O3 optimisation and w/o array boundary check

10^3 evolutions	2×10^6 structure functions	total
3.7 secs	4.0 secs	7.7 secs

- Heavy quark contributions to $F_{2,L}$ with HQSTF take about the same CPU as the light quark structure functions with ZMSTF

☺ Yes, QCDNUM is pretty fast

QCDNUM Releases

Website
https://www.nikhef.nl/~h24/qcdnum

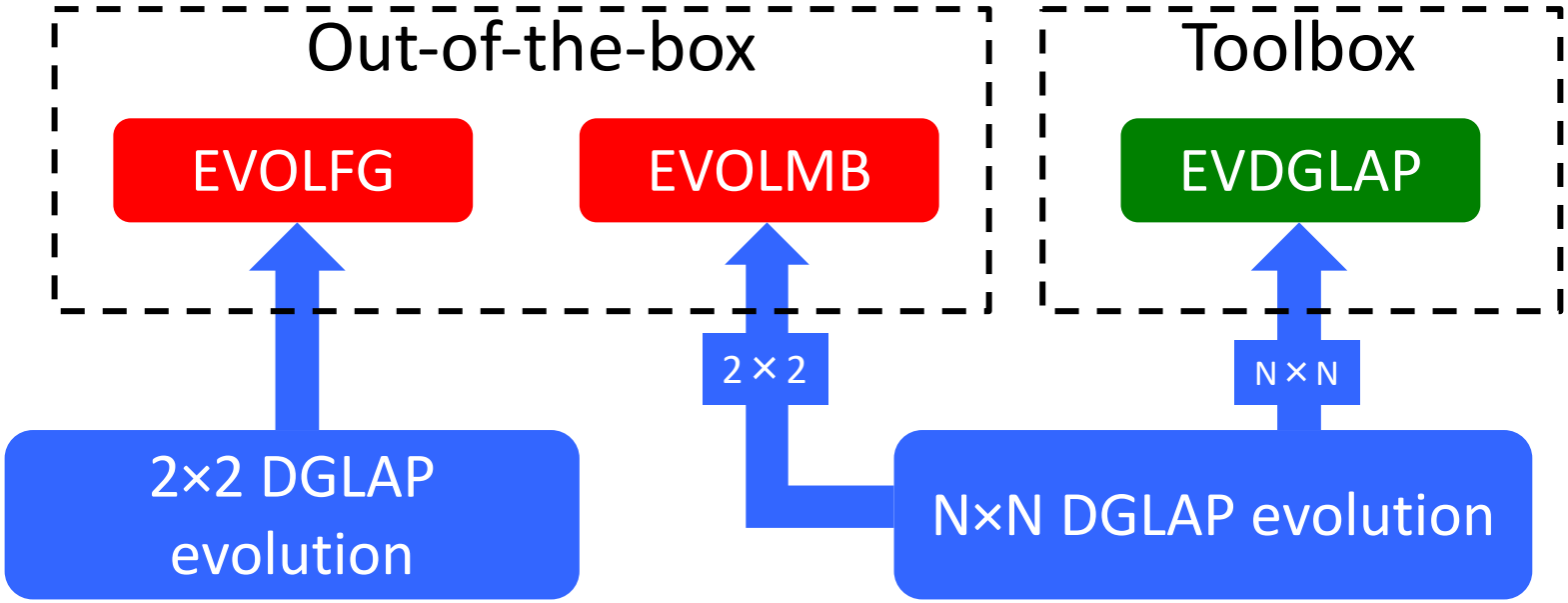
- 17-00/07: Stable release
  - Bug fix in singlet time-like evolution (Feb 2016)

arXiv:1602.08383

- 17-01/13: Pre-release on the road to QCDNUM-18-00
  - Suite of toolbox routines for N-fold DGLAP evolution
  - Routine to copy toolbox pdf set to internal memory
  - Imported pdf sets can have pdfs beyond qluon and quarks
  - Can store pdf sets with different evolution parameters
  - QCDNUM steering with datacards
  - New very fast pdf interpolation routines
  - C++ interface (written by Valerio, not yet fully tested)
- Upcoming release 17-01/14
  - New out-of-the-box evolution routine (testversion)
  - First steps towards thread support via OpenMP

Coming soon: QCDNUM-17-01/14

- EVOLMB: new out-of-the-box evolution routine



- EVOLMB is fully backward compatible with EVOLFG and will ultimately replace it

NB: N×N refers to the dimension of the splitting function matrix for N-fold evolution

EVOLMB versus EVOLFG

```
call EVOLMB(itype,func,def,iq0,epsi)
```

`itype`	Select polarised/unpolarised/time-like
`func(j,x)`	Input pdfs $f_j(x)$ at the input scale `iq0`
`def(i,j)`	Contribution of (anti)quark flavour `i` to input pdf `j`
`iq0`	Starting scale of the evolution
`epsi`	Output smoothness indicator to detect oscillations

- EVOLMB has the same argument list as EVOLFG
  - `itype` now allows you to select the output pdf set
  - `iq0` can be anywhere within the grid or cuts

NEW

Output pdf set selection in EVOLMB

- Evolution type selection via `itype` allows for direct storage of evolved pdfs into any pdf set [1-24]

1	Evolve unpolarised pdfs in `iset=1`
2	Evolve polarised pdfs in `iset=2`
3	Evolve fragmentation fcs in `iset=3`
`10*iset+itype`	Evolve 1, 2 or 3 and store in `iset` NEW

- Thus `itype=52` stores polarised pdfs in set 5, etc.
- This provides an alternative to copying pdf set 1, 2, 3 to another set with PDFCPY

Start scale in EVOLMB

- For EVOLFG the VFNS start scale must be below the charm threshold: evolution always starts at nf = 3
- For EVOLMB the VFNS start scale can be anywhere inside the grid: evolution can start at nf = 3, 4, 5, 6 NEW
- This allows for evolution with intrinsic heavy flavours
- Back-evolution of a heavy pdf over the threshold:
  - Subtract the discontinuity (if any) and set the heavy quark pdf to $f(x, \mu_h)$ for all $\mu \leq \mu_h$ (with option to set f to zero)
- When you start the evolution at a threshold `iqh`:

  `iq0 = +iqh` : start with nf above the threshold
  `iq0 = -iqh` : start with nf below the threshold

Thresholds in EVOLFG and EVOLMB

```
call SETCBT( nfix, iqc, iqb, iqt )
```

Scheme	nfix	Thresholds	
FFNS	3/4/5/6	No flavour thresholds	
VFNS	0	iqc	Boundary nf = 3/4
		iqb	Boundary nf = 4/5 must be iqb ≥ iqc+2
		iqt	Boundary nf = 5/6 must be iqt ≥ iqb+2

Threshold settings in the VFNS

- EVOLFG evolution must start at nf = 3 so that the set of thresholds put inside the grid must always include iqc
- EVOLMB has no start point restriction so that one may put any single threshold, or any two consecutive thresholds, or all three thresholds

NEW

New in EVOLMB: thread support

- At fixed nf, the 2nf singlet and nonsinglet evolutions can in principle be run in parallel (up to 12 evolutions)
- OpenMP directives are not yet implemented but by design EVOLMB should be thread-safe (EVOLSG is not)
- Proof of principle in MickeyMouse code: fake evolution of 6 pdfs distributed over 4 threads on my MacBook

```
PDF  1 NF  6 evolved up in thread  0
PDF  2 NF  6 evolved up in thread  0
PDF  3 NF  6 evolved up in thread  1
PDF  4 NF  6 evolved up in thread  1
PDF  5 NF  6 evolved up in thread  2
PDF  6 NF  6 evolved up in thread  3
```

- Might become quite a CPU saver when it all works …

EVOLMB issues after its first release

- QCDNUM plague: spline oscillation when back-evolving
  - Presently cured by iterative back-evolution
  - No problem when back-range is small < iqc for EVOLFG
  - For EVOLMB the back-range is not restricted so that the iterative solution needs to be reviewed for large back-ranges
- QCDNUM feature: speed
  - EVOLMB is now about a factor of two slower than EVOLFG
  - Have to investigate timing & speedup of EVOLMB
- EVOLMB will replace EVOLFG when all this is resolved

To come: QEDEVOL package

- QCDNUM QCD-QED is already implemented in xFitter and Renat sent me the standalone version of the code
- Turn this into a package with basically two routines:

`QEDWGTS`	To be called instead of `FILLWT`
`QEDEVOL`	To be called instead of `EVOLFG`

- To do in addition:
  - Provide C++ interface
  - Provide thread support
  - Provide small write-up
  - Include the package in the QCDNUM distribution with proper reference to Renat's work

QCDNUM todo list

- Finalise EVOLMB and release QCDNUM-17-01/14
- And then, not in order of priority:
  - Turn Renats code into a QEDEVOL package
  - Implement cuts
  - Fully test C++ interface to QCDNUM
  - Improve numerical stability of backward evolution
  - More attention to OpenMP thread-safe code in QCDNUM
  - Toolbox: more flexibility through user-defined functions
  - Upgrade polarised and time-like evolution to NNLO