



QCDNUM Status and Plans

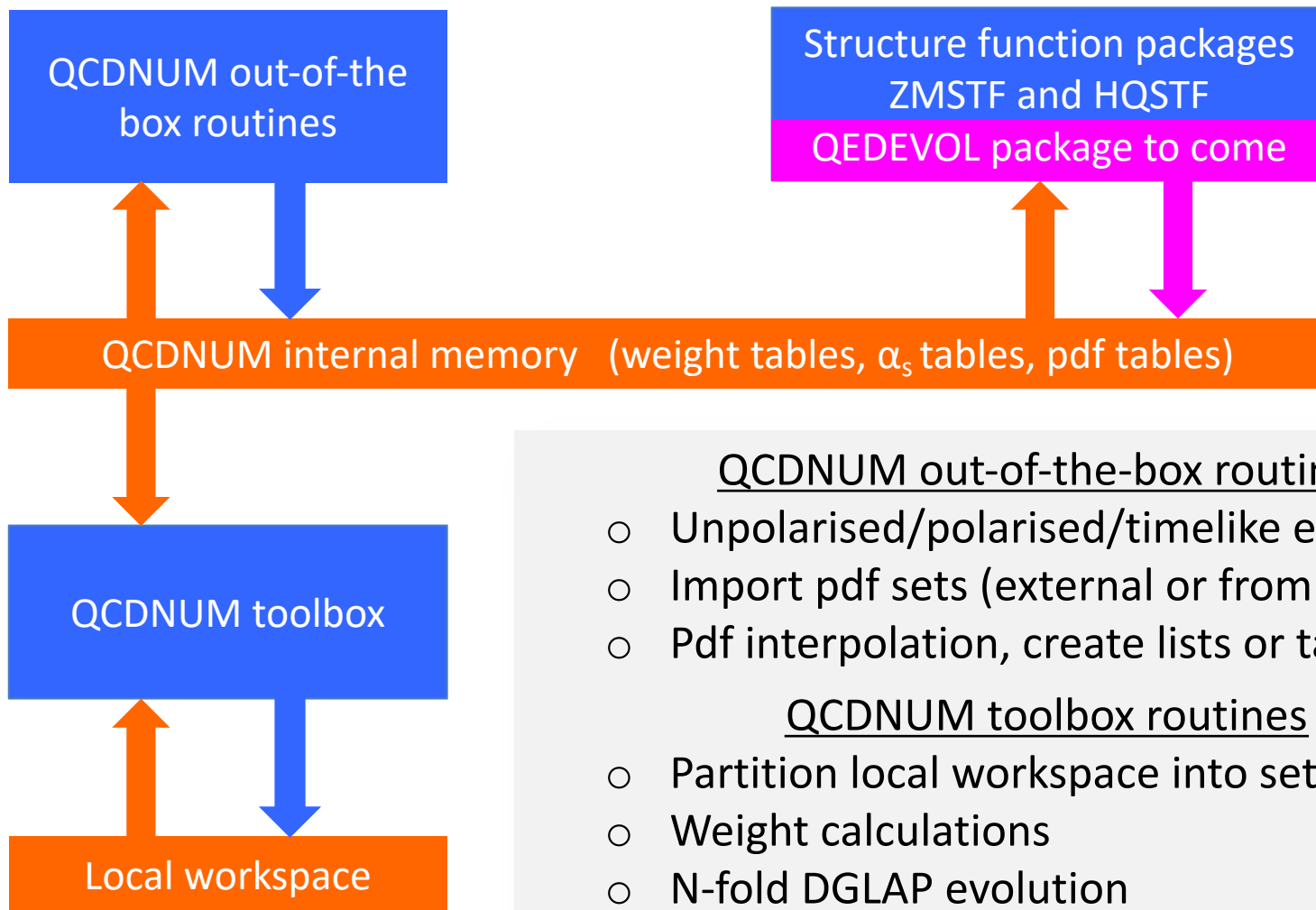
Michiel Botje

Nikhef, Amsterdam

xFitter external workshop

Krakow March 6, 2018

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

QCDNUM releases

Website

<https://www.nikhef.nl/~h24/qcdnum>

- [17-00/08: Stable release](#)

- Bug fix in singlet time-like evolution (Feb 2016)

arXiv:1602.08383

- [17-01/XX: Pre-releases on the road to QCDNUM-18-00](#)

- Suite of toolbox routines for N-fold DGLAP evolution
- Imported pdf sets can have pdfs beyond gluon and quarks
- Can store pdf sets with different evolution parameters
- New very fast pdf interpolation routines

- [17-01/14: Released 21-Dec-2017](#)

- C++ interface for out-of-the-box, ZMSTF and HQSTF

arXiv:1712.08162

- [17-01/15: Almost ready](#)

- New out-of-the-box evolution routine with intrinsic heavy flavours
- New routine to set cuts in the kinematic plane

C++ or FORTRAN77, thats the question

- QCDNUM is fast mainly because of
 - Code written in FORTRAN77
 - Very fast addressing by an in-house memory manager
- Recoding in C++ would be a huge effort and most likely will lead to quite some speed penalties
- Decide not to recode but to provide C++ interface
- Interface initially written by Valerio Bertone (thanks!)
- Fully operational in 17-01/14 except for the toolbox

C++ interface to QCDNUM

- Very easy to use, basically,

```
call SUB(arguments)
```



```
QCDNUM::sub(arguments);
```

- Documented in the write-up and in [ArXiv:1712.08162](https://arxiv.org/abs/1712.08162)

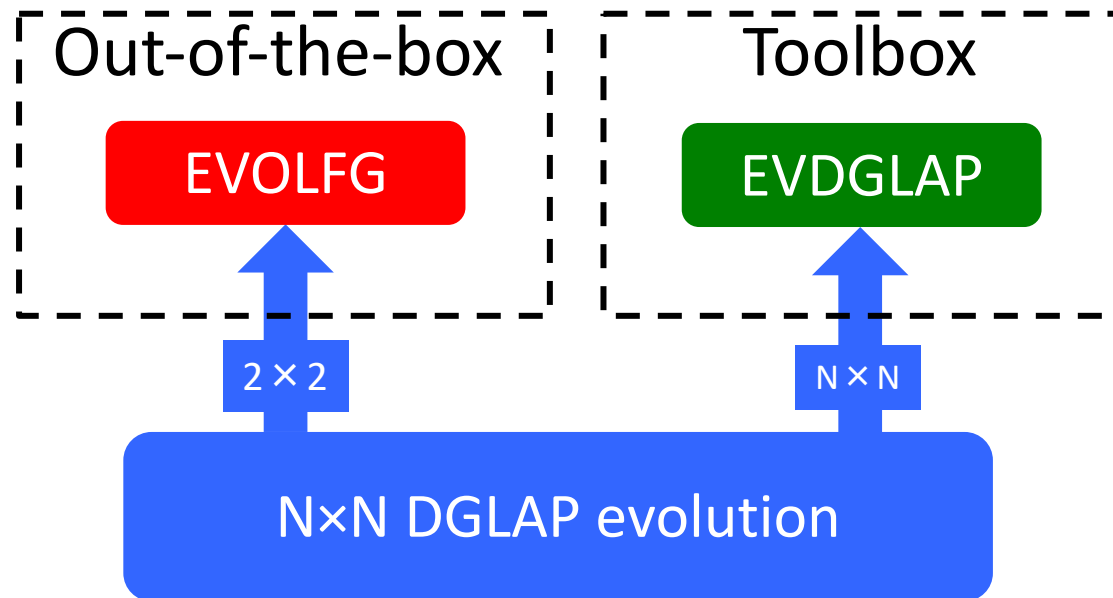
A C++ interface to QCDNUM

V. Bertone*, M. Botje†

NIKHEF, Science Park, Amsterdam, the Netherlands

QCDNUM-17-01/15 almost done

- Replaced old 2-fold DGLAP routine by an interface to the toolbox N-fold evolution routine



- Had to speed-up N-fold DGLAP by factor 2
- Present speed penalty only 10-20% 😊

New EVOLFG routine

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

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

Same argument list as before but more flexible

NEW

- `itype` now allows you to select the output pdf set
- `iq0` can be anywhere within the grid or cuts
- `func` accepts parameterisation of intrinsic heavy flavours

Output pdf set selection in EVOLFG

- 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 EVOLFG

- Previously the VFNS start scale had to be below the charm threshold: evolution always started at $nf = 3$
- Now the VFNS start scale can be anywhere inside the grid: evolution can start at $nf = 3, 4, 5, 6$ NEW
- When you start the evolution at a threshold iqh :
 - $iq0 = +iqh$: Start with nf above the threshold
Do the matching in backward evolution
 - $iq0 = -iqh$: Start with nf below the threshold
Do the matching in forward evolution

Reminder

Avoid QCDNUM back evolution as much as possible

Scheme selection and threshold settings

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

Scheme	nfix	Thresholds
FFNS	3/4/5/6	No flavour thresholds
VFNS	0	<i>iqc</i> Boundary nf = 3/4
		<i>iqb</i> Boundary nf = 4/5 must be $iqb \geq iqc+2$
		<i>iqt</i> Boundary nf = 5/6 must be $iqt \geq iqb+2$

Settings in SETCBT more flexible than before

- **EVOLFG** has no start point restriction so that one may put any single threshold, or any two consecutive thresholds, or all three thresholds NEW
- **nfix** = 0 : select VFNS with dynamic heavy flavours
- **nfix** = 1 : select VFNS with intrinsic heavy flavours (see later) NEW

Input pdfs for n_f active flavours

- Input pfs are parameterised in `func(j, x)`

```
function func(j, x)
  if(j.eq. 0) func = gluon(x)
  if(j.eq. 1) func = pdf01(x)
  ..
  if(j.eq.12) func = pdf12(x)
  return
end
```

- Called for the $j = 0, \dots, 2n_f$ input pdfs at μ_0^2
- Parameterisations for $j > 2n_f$ are ignored

Flavour composition of input pdfs

- Contribution of flavour i to f_j is given in $\text{def}(i, j)$

i	\bar{t}	\bar{b}	\bar{c}	\bar{s}	\bar{u}	\bar{d}	g	d	u	s	c	b	t
$j = 1$	6	5	4	3	3	3	·	3	3	3	4	5	6
2	6	5	4	3	3	3	·	3	3	3	4	5	6
3	6	5	4	3	3	3	·	3	3	3	4	5	6
4	6	5	4	3	3	3	·	3	3	3	4	5	6
5	6	5	4	3	3	3	·	3	3	3	4	5	6
6	6	5	4	3	3	3	·	3	3	3	4	5	6
7	6	5	4	4	4	4	·	4	4	4	4	5	6
8	6	5	4	4	4	4	·	4	4	4	4	5	6
9	6	5	5	5	5	5	·	5	5	5	5	5	6
10	6	5	5	5	5	5	·	5	5	5	5	5	6
11	6	6	6	6	6	6	·	6	6	6	6	6	6
12	6	6	6	6	6	6	·	6	6	6	6	6	6

- **EVOLFG** takes $2n_f \times 2n_f$ sub-matrix and ignores the rest
- Independent pdf input requires non-singular sub-matrix

NEW

VFNS with intrinsic heavy flavours

- Set `nfix = 1` in upstream call to `SETCBT`
- Provide parameterisation in `func(j, x)` for $j > 2n_f$
- Specify flavour composition in `def(i, j)` for $j > 2n_f$
 - Accepts only a linear combination of `h` and `hbar` without admixture of other flavours
 - You can set all coefficients to zero to turn heavy flavour off
- Input provides scale-independent density below threshold and evolution start point at threshold

NB: dynamic heavy flavours (`nfix = 0`) are zero below threshold

Matching with intrinsic heavy flavours

- More complicated since heavy quark enters the game at NLO

$$\begin{aligned}
 \Delta g &= a_s A_{gh} \otimes h^+ + a_s^2 \{ A_{gq} \otimes q_s + A_{gg} \otimes g \} \\
 \Delta h^+ &= a_s A_{hh} \otimes h^+ + a_s^2 \{ A_{hq} \otimes q_s + A_{hg} \otimes g \} \\
 \Delta h^- &= a_s A_{hh} \otimes h^- \\
 \Delta q_i^\pm &= a_s^2 A_{qq} \otimes q_i^\pm
 \end{aligned}$$

BMSN (NNLO)

- Code presently OK for forward but not for reverse matching
- Problem: the QCDNUM pdf basis maximally decouples DGLAP but not the matching equations
- Solution: transform to a basis that better decouples the ME
- Documented in write-up Appendix C, code still to be written

Cuts in the kinematic plane

```
call SETLIM(ixmin, iqmin, iqmax, dum)
```

- Set kinematic cuts for
 - Next evolution by **EVOLFG** or **EVDGLAP** (toolbox)
 - Next pdf import by **EXTPDF**
- To release cut set parameter to zero (or outside grid)
- Useful for speeding-up pdf fits
 - In χ^2 loop limit evolution to kinematic range of the data
 - After convergence evolve, only once, on full grid
- Useful to define kinematic range of imported pdf set

New in EVOLSG: thread-safety

- 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 the new **EVOLSG** routine should be thread-safe
- 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 ...

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:

<code>QEDWGTS</code>	To be called instead of <code>FILLWT</code>
<code>QEDEVOL</code>	To be called instead of <code>EVOLFG</code>

- 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 matching and release QCDNUM-17-01/15
- And then, not in order of priority:
 - Put OpenMP directives in EVOLSG for parallel running
 - Upgrade polarised and time-like evolution to NNLO
 - Turn Renats code into a QEDEVOL package
 - Toolbox: more flexibility through user-defined functions
 - Investigate numerical stability of backward evolution