QCDNUM Status and Plans

Michiel Botje

Nikhef, Amsterdam

xFitter external workshop

Minsk March 18, 2019

QCDNUM releases

- 17-01/14: Released 21-Dec-2017
  - C++ interface for out-of-the-box, ZMSTF and HQSTF

arXiv:1712.08162

- 17-01/15: Will be released today 
  - Out-of-the-box evolution routine with intrinsic heavy flavours
  - New out-of-the-box singlet/non-singlet evolution routine
  - New routine to set cuts in the kinematic plane
  - More flexibility in setting thresholds
  - Evolution start scale can be anywhere in μ^2
  - Pdf access not anymore restricted to those with current parameters

Backward compatible (almost)
Just install 17-01/15 and enjoy the added features

Scheme selection and threshold settings

```
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

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

Reminder: two μ^2 grids

- Internally a threshold is stored twice
  - With n_f above threshold
  - With n_f below threshold
- The user sees two grids
  - `iq = +iqh` : n_f above threshold (456-grid)
  - `iq = -iqh` : n_f below threshold (345-grid)
- To see charm matching discontinuity, for instance:

```
FVALIJ(iset,id,ix,+iqc,1) – FVALIJ(iset,id,ix,-iqc,1)
```

Updated EVOLFG routine

```
call EVOLFG(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

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

QCDNUM back evolution is iterative but reasonably accurate

To see what EVOLFG does exactly

```
call SETINT('edbg',1)
```

NEW

```
---------------- forward ----------------
EVOLVE iq1,2 =     9    9   nf =   3  start
JUMPNF iq1,2 =     9    9   nf =   3  4
EVOLVE iq1,2 =     9   18   nf =   4
JUMPNF iq1,2 =    18   18   nf =   4  5
EVOLVE iq1,2 =    18   44   nf =   5
JUMPNF iq1,2 =    44   44   nf =   5  6
EVOLVE iq1,2 =    44   60   nf =   6
---------------- reverse ----------------
EVOLVE iq1,2 =     9    1   nf =   3  start
```

iq0 = -iqc

3 -> 4 transition forward evolution

Here is an example showing what happens when we start the evolution at the charm threshold with 3 (-iqc) or 4 (+iqc) flavours

```
---------------- forward ----------------
EVOLVE iq1,2 =     9   18   nf =   4  start
JUMPNF iq1,2 =    18   18   nf =   4  5
EVOLVE iq1,2 =    18   44   nf =   5
JUMPNF iq1,2 =    44   44   nf =   5  6
EVOLVE iq1,2 =    44   60   nf =   6
---------------- reverse ----------------
EVOLVE iq1,2 =     9    9   nf =   4  start
JUMPNF iq1,2 =     9    9   nf =   4  3
EVOLVE iq1,2 =     9    1   nf =   3
```

iq0 = +iqc

4 -> 3 transition reverse evolution

New EVSGNS routine

```
call EVSGNS(itype,func,isns,n,iq0,epsi)
```
NEW

`isns(i)`	Type of evolution (singlet/gluon, non-singlet)
`n`	Number of evolutions

Creates a user-pdf set with n+1 pdfs

- It is not possible to match arbitrary sets of pdfs so that the evolution can only run in the FFNS or MFNS
- There is also a routine to import arbitrary pdf sets

```
call USRPDF(func,iset,n,offset,epsi)
```
NEW

Input pdfs for n_f active flavours

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

```
function func(j,x)

if(j.eq.-1) initialise func

if(j.eq. 0) func = gluon(x)
if(j.eq. 1) func = pdf01(x)
 ..
if(j.eq.12) func = pdf12(x)

return
end
```

Dummy call with `j = -1` NEW

- Called for the `j` = `0`, …, `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 `def(i,j)`

i	$\bar{t}$ -6	$\bar{b}$ -5	$\bar{c}$ -4	$\bar{s}$ -3	$\bar{u}$ -2	$\bar{d}$ -1	g 0	d 1	u 2	s 3	c 4	b 5	t 6
$j=1$	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

VFNS with intrinsic heavy flavours

- Set `nfix` = `1` in upstream call to `SETCBT` NEW
- 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
  - Row of zero's terminates the heavy flavour input
- Input provides scale-independent density below threshold and evolution start point at threshold

NB: input heavy flavours at μ_0^2 are <u>always</u> intrinsic

Matching with intrinsic heavy flavours

R.D. Ball et al.
PLB754, 49 (2016)
arXiv:1510.00009

- 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)

- Difficulty: the QCDNUM pdf basis maximally decouples DGLAP but not the matching equations
- Therefore transform to a basis that better decouples the ME
- Reverse matching is numerically unstable ⇒ iterative solution
- All this is documented in the write-up Appendix C and D

Cuts in the kinematic plane

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

NEW

- Set kinematic cuts for
  - Next evolution by EVOLFG or EVDGLAP (toolbox)
  - Next pdf import by EXTPDF
- To release cut set parameter to zero
- 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

Evolution parameter keys

- There are 12 parameters divided in 6 groups

1.	order (1)	3.	thresholds (4-7)	5.	cuts (10-12)
2.	α_s (2-3)	4.	scale (8-9)	6.	all (1-12)

- All settings are stored in a repository and given a version number (key): (un)equal keys means (un)equal parameters
- Handy for quick checks on parameter groups NEW
- E.g. pdf sets with the same threshold settings have

```
KEYGRP(iset1,3) == KEYGRP(iset2,3)
```

Summary of new subroutines

`WTFILE`	Maintain an up-to-date weight file on disk
`NFRMIQ`	Number of flavours vs μ^2 for a given pdf set
`SETLIM`	Set evolution limits
`GETLIM`	Get current limits
`KEYGRP`	Key (version number) of a parameter group
`EVSGNS`	Evolve an arbitrary set of singlet/non-singlet pdfs
`USRPDF`	Import an arbitrary pdf set
`IEVTYP`	Get evolution type of a pdf set
`FFPLOT`	Generate plot file (for gnuplot)

Pdf access

- PDF tables are somewhat complicated objects with a μ^2 subgrid structure, depending on the threshold settings
- Subgrids are handled by fast pointer-tables
- There used to be one pointer-table for the current settings which restricted pdf access to those settings only
- Now every pdf set has its own pointer-table so that access is not anymore restricted

Change of basis

- There are two changes in the basis definition
  - e_2 = u – d has changed to d – u
  - e_h = $h^{\pm}$ for heavy quarks below threshold
- This should not affect your code unless you use the basis pdfs instead of the flavour pdfs
- The reason for these changes is the introduction of a fast transformation algorithm, including transformation to a special basis for intrinsic heavy quark matching

Performance

- Standard timing test of 1000 evolutions and 2M structure function calculations (mimic a pdf fit)
- On a MacBook Pro 2018 this gives

5.7 sec	QCDNUM-17-01/14
6.2 sec	QCDNUM-17-01/15

- About 10% speed penalty for 17-01/15
- Modest price for a lot of increased flexibility

Accuracy of iterative reverse evolution

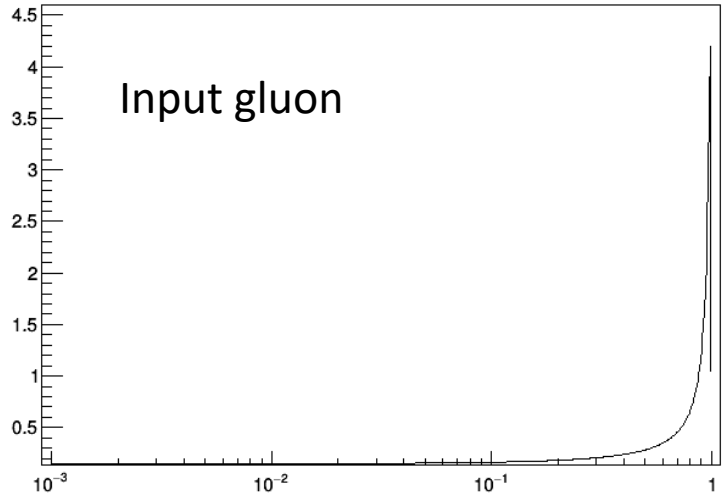
- Do NNLO VFNS evolution
- Evolve from 2 to 10^4 GeV^2 and then back to 2 GeV^2
- Compare forward and reverse evolution for all 13 basis pdfs

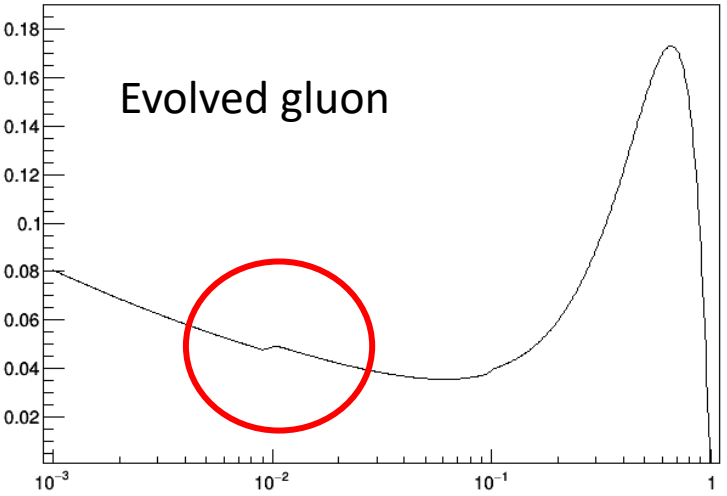
```
id =  0     dif =  0.0014
id =  1     dif =  0.0019
id =  2     dif =  0.0018
id =  3     dif =  0.0019
id =  4     dif =  0.0016
id =  5     dif =  0.0001
id =  6     dif =  0.0000
id =  7     dif =  0.0019
id =  8     dif =  0.0018
id =  9     dif =  0.0019
id = 10     dif =  0.0000
id = 11     dif =  0.0000
id = 12     dif =  0.0000
```

Small hiccup

- Pomeron evolution with input gluon peaked at very large x
- Multiple x-grids show up in the evolved gluon (and singlet)
- Will be improved in the next QCDNUM release

Thanks to Daniel Britzger for spotting the problem





EVOLSG threading with OpenMP

- 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 …

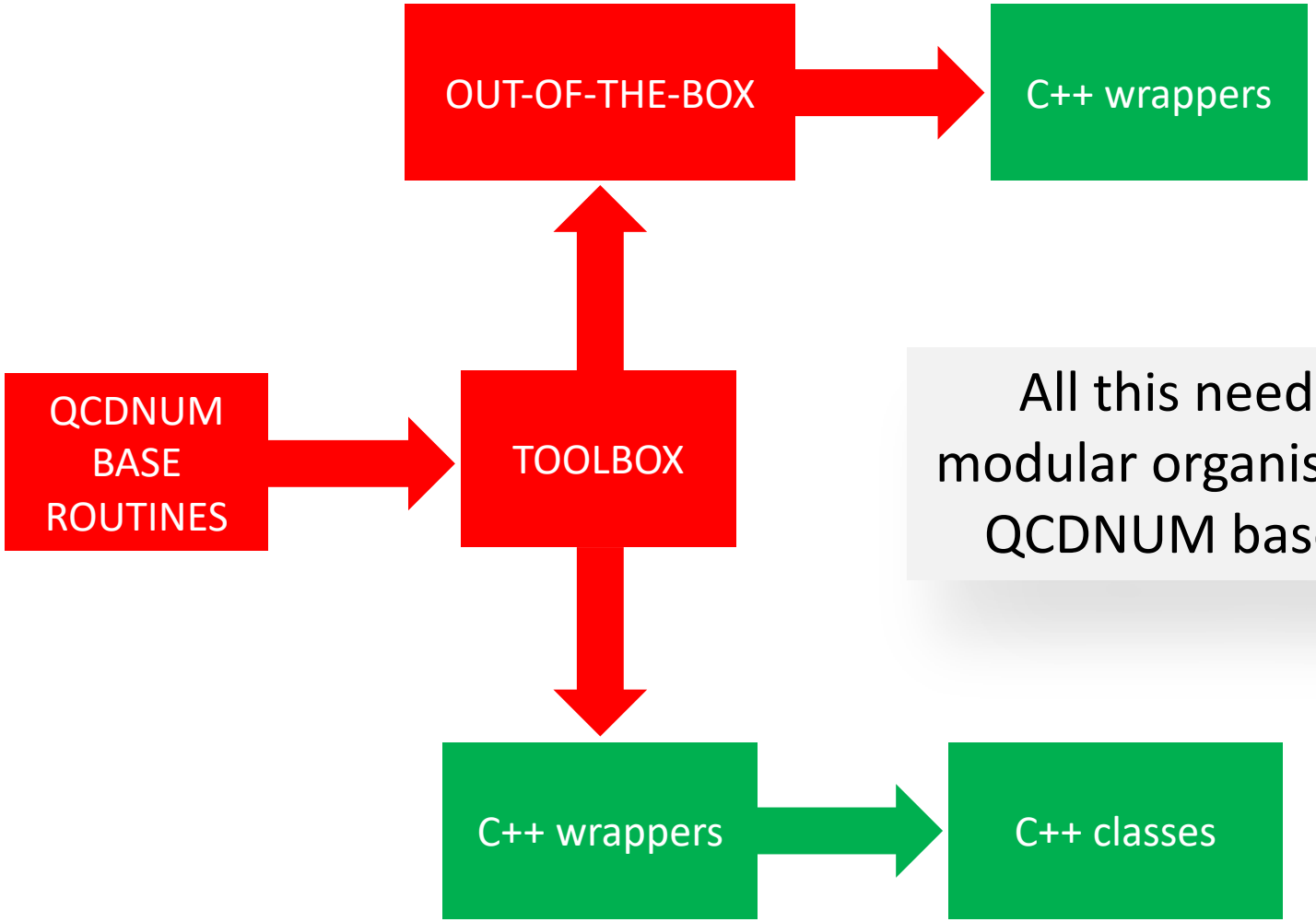
QCDNUM short-term todo list

- Put OpenMP directives in EVOLSG for parallel running
- Better handling of multiple x-grids
- Upgrade polarised and time-like evolution to NNLO
- Turn Renats QCD-QED code into a QEDEVOL package (this is after a re-design of the toolbox)

Long-term code development

- QCDNUM memory is a collection of table sets stored in a large linear array (workspace)
- Routines that operate on a set only care about the position (base address) of table sets in the workspace
- Thus we can have one large workspace with many sets (FORTRAN) or many workspaces with one set (C++)
- In this way table sets can become objects of a C++ class and C++ wrappers of FORTRAN routines can become member functions of that class

Long-term program structure



All this needs a more modular organisation of the QCDNUM base routines

- There are a lot of internal changes in 17-01/15
- Code passes all my tests
- But please stress-test the program in xFitter and report problems to me