

The **QCDNUM** evolution program

Michiel Botje BAT kick-off meeting 26-11-20

- QCDNUM is available from the web site https://www.nikhef.nl/~h24/qcdnum
- QCDNUM has two user interfaces
 - <u>Out-of-the-box</u> for standard unpolarised, polarised and time-like evolution up to NNLO
 - <u>Toolbox</u> for customised DGLAP and convolution of pdfs with coefficient functions
- Stable version qcdnum-17-00
 - Latest version 17-00/08 released at 01-05-2017
 - Fortran only
- Development version qcdnum-17-01/xx
 - Latest version 17-01/15 released at 31-10-2019
 - Out-of-the-box with much increased functionality and C++ interface (~stable)
 - Toolbox still in Fortran only
 - Plans to re-vamp toolbox providing both Fortran and C++ interface/classes
- My recommendation is to use qcdnum-17-01/xx and always upgrade to latest version

QCDNUM-17-00/08

Download

How to install

Write-up

Example jobs

Release history

QCDNUM-17-01/15

Download

How to install

Write-up

C++ interface

Example jobs

Tutorial

Release history

Whats next

QCDNUM

Known bugs

Talks

Contact

What's in the box

- Very detailed write-up
- Source code
 - MBUTIL pool of utility routines
 - QCDNUM out-of-the-box evolution + toolbox
 - ZMSTF zero-mass structure functions F_2 , F_L and xF_3
 - HQSTF heavy quark stfs in 3-flavour FFNS
- Easy installation with autotools

```
gunzip qcdnum170115.tar.gz
tar -xvf qcdnum170115.tar
cd qcdnum-17-01-15
./configure
make
make install
```

• Example codes, for instance,

cd qcdnum-17-01-15/run
./runtest example (Fortran)
./runtest exampleCxx (C++)

Takes only a few minutes on a MACBOOK to install QCDNUM and run your first NNLO evolution

Numerical method in a nutshell

- Solve DGLAP numerically on a log $x-\mu^2$ grid
- Based on linear or quadratic spline interpolation on multiple equidistant grids
- Convolution integrals become weighted sums with weights calculated only once at program initialisation
- Evolution step becomes a lower triangular $n \times n$ matrix equation solved by forward substitution
- This matrix roll-up is the only $O(n^2)$ calculation in the whole program; everything else is O(n)
- The O(n) algorithms + Fortran + in-house memory manager make for very fast code
- C++ interface does not slow-down QCDNUM

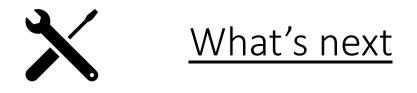
Fast?

- Standard timing test: mimic QCD fit by 1000 evolutions and 2×10^6 structure function calculations in NNLO
- On a MacBook Pro 2018 this takes 6.2 sec
- QCDNUM is probably the fastest QCD evolution program on the market

What QCDNUM gives you

- Evolution of unpolarised (NNLO), polarised (NLO) or time-like (NLO) pdfs
- Weights for all # of flavours, all orders and all evolution types simultaneous in memory + disk dump/read
- Supports FFNS, VFNS with dynamic or intrinsic heavy flavours
- Can vary renormalisation and factorisation scales
- Can hold up to 24 different pdf sets in memory
- Possibility to read pdf sets from an external source into memory
- Fast interpolation routines, also providing singlet/non-singlet decompositions (useful for structure functions)
- Toolbox provides customised DGLAP evolution and convolution in mass-less and generalised mass schemes
- Compact user interface; here we show an evolution of a full pdf set (13 pdfs) in 9 lines of code:

1	call	QCINIT(6,' ')	Initialise
2	call	GXMAKE(xmin,1,1,nxin,nx,iosp)	x-Grid
3	call	GQMAKE(qq,wt,2,nqin,nq)	mu2-Grid
4	call	<pre>FILLWT(ityp,id1,id2,nw)</pre>	Compute weights
5	call	SETORD(iord)	LO, NLO, NNLO
6		SETALF(as0,r20)	Input alpha_s
7	call	<pre>SETCBT(nfin,iqc,iqb,0)</pre>	FFNS, VFNS, thresholds
8	call	<pre>EVOLFG(ityp,func,def,iq0,eps)</pre>	Evolve all PDFs
9	call	ALLFXQ(ityp,x,q,pdf,0,1)	Interpolate all PDFS



- <u>Thread support</u> (parallel processing) using openmp
 - Fortran code is written such that pdf evolution can run on multiple processors
 - Proof of principle running Mickey-Mouse evolution on 4 processors

PDF	1 NF	6 evolved up in thread	0
PDF	2 NF	6 evolved up in thread	Θ
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

- Tricky to implement but may become huge time saver in fitting and Monte-Carlo applications
- <u>Re-design toolbox</u>
 - In-house memory manager in QCDNUM not unlike C++ memory management
 - New C++ interface:
 - Memory objects in Fortran become memory objects of C++ classes
 - Management routines in Fortran become member functions of C++ classes
 - First step: new Fortran/C++ memory manager is working OK, but not yet implemented
- <u>Bring polarised and time-like evolution up to NNLO</u> (now NLO only)