QCDNUM Status and Plans

Michiel Botje
Nikhef, Amsterdam

HERAFitter developers meeting

Heidelberg, March 3, 2015

QCDNUM upgrades towards v17-01

- The convolution toolbox of v17-00 is upgraded to a combined evolution/convolution toolbox
- With this new toolbox you can feed-in your own splitting functions, perturbative expansion coefficients, and do n-fold coupled evolutions
- The convolution engine for zero-mass or generalised mass structure functions is unchanged since v17-00
- Present beta release is QCDNUM-17/0g (31-03-2015)

Already before V17-01/0g

- Can store more types of table in the local workspace

$$\text{v17.00} \rightarrow \text{one set of weight tables}$$

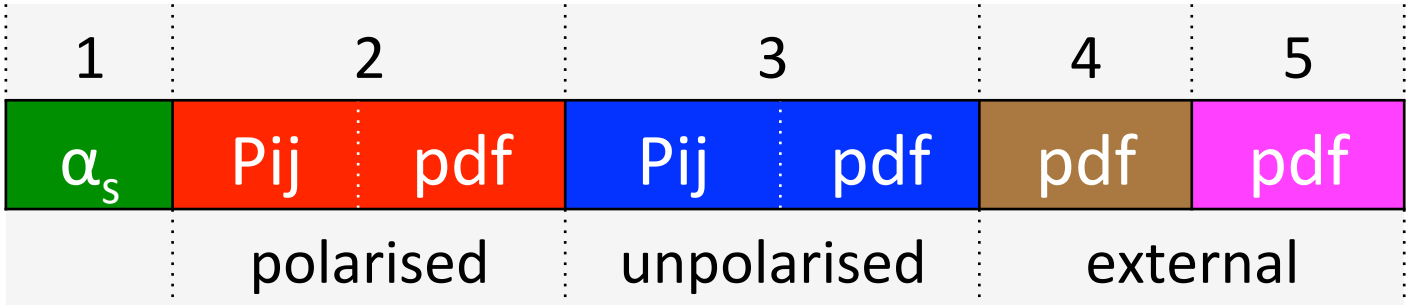
$$\text{v17.01} \rightarrow \text{one or more sets of} \begin{cases} \text{weight tables} \\ \alpha_s \text{ tables} \\ \text{pdf tables} \end{cases}$$

- Suite of toolbox routines for coupled n×n evolution

Subroutine or function	Description
`EVFILLA ( w,id,func)`	Fill α or α_s table
`EVGETAA ( w,id,iq,nf,ithresh )`	Get value of α or α_s
`EVDGLAP ( w,iw,ia,if,s,m,n,iq,nf,e )`	Coupled evolution
`EVPDFIJ ( w,id,ix,iq,ichk )`	Pdf at grid point
`EVPLIST ( w,id,x,qmu2,pdf,n,ichk )`	Pdf interpolation
`EVTABLE ( w,id,x,nx,q,nq,table,ichk )`	Pdf interpolation
`EVFCOPY ( w,id,def,iset )`	Copy to internal memory

Already before V17-01/0g: table sets

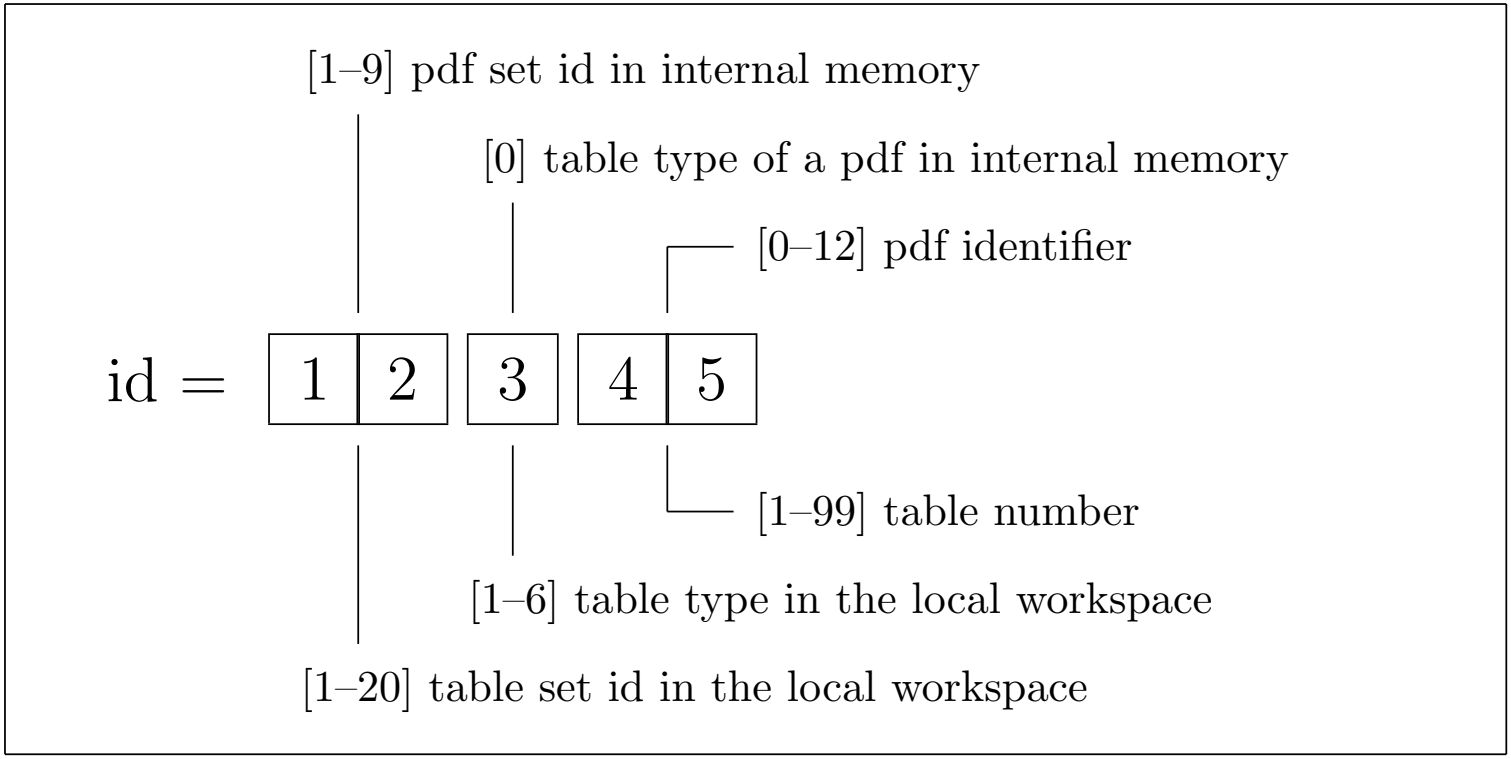
- A table set is created by a call to `maketab`
- Multiple calls to `maketab` do create multiple sets
- Easy to create layouts like the one below with 5 sets



- The use of table sets can greatly simplify your code
- For example, a routine evolves both polarised and unpolarised pdfs simply by switching the table set identifier from 2 to 3

Already before V17-01/0g: global table ids

- Table attributes are table set identifier, table type, table number and memory location (local workspace or internal)
- All this can be encoded in a global table identifier



- Makes it easy to access tables in internal or local memory

New in current release V17-01/0g

- Improved control over a (small) bias in the n×n coupled evolution with `evdglap`
- QCDNUM steering with datacards

Coupled evolution in the VFNS

- `Evdglap` evolves with fixed n_f up to next threshold

```
call EVDGLAP ( ..., start, ..., n, iq, ... )
```

argument	value on entry	value on exit
`start`	start pdf	pdf at next threshold
`iq`	start scale	next threshold

- Easy to implement threshold loop in the VFNS

```
start = pdf at iq0
call EVDGLAP ( ..., start, ..., n, iq, ... )
start = start + discontinuity at threshold
call EVDGLAP ( ..., start, ..., n, iq, ... )
..
```

- Pdfs in `start` array are a composite of pdfs evolved on subgrids: this causes a small bias in the evolution

Cure: internal transfer mode

- Subgrid-by-subgrid internal transfer of pdfs

```
call EVDGLAP ( ..., start, ...,-n, iq, ... )
```

argument	value on entry	value on exit
start	discontinuity at threshold	zero
iq	threshold	next threshold

- Still easy to implement threshold loop in the VFNS

```
start = pdf at iq0
call EVDGLAP ( ..., start, ..., +n, iq, ... )
start = discontinuity at threshold
call EVDGLAP ( ..., start, ..., -n, iq, ... )
..
```

- Probably a non-issue since bias is really small as far as I can see, but `evdglap` users should check

New in current release 17-01/0g: datacards

- Here is a datacard file to set-up a QCDNUM evolution:

```
' SETLUN   6                              '
' GXMAKE   3  100   1   1.D-4            '
' GQMAKE       60   2   2.     1.D4      '
' FILLWT   1                              '
' SETORD   3                              '
' SETALF   0.364   2.                     '
' SETCBT   0       3.   25.    1.D11     '
```

- Calls to QCDNUM routines can now be replaced by a datacard read

```
..
call qcards( usub, 'example.dcards', 0 )
itype = 1
q0    = 3.5
call evolfg( itype, func, def, iqfrmq(q0), eps )
..
```

QCDNUM has a set of 13 predefined keys

- `call qcbook ( 'List', ' ' )`

```
 1   SETLUN   QKEY
 2   SETVAL   QKEY
 3   SETINT   QKEY
 4   GXMAKE   QKEY
 5   GQMAKE   QKEY
 6   FILLWT   QKEY
 7   SETORD   QKEY
 8   SETALF   QKEY
 9   SETCBT   QKEY
10   MIXFNS   QKEY
11   SETABR   QKEY
12   SETCUT   QKEY
13   QCSTOP   QKEY
```

- These keys cover all calls needed to set-up an evolution
- The evolution itself can be driven by a user defined card

Add user defined card: ' EVOLFG 1 3.5 '

- `call qcbook ( 'Add', 'EVOLFG' )`

```
 1  SETLUN  QKEY
 2  SETVAL  QKEY
 3  SETINT  QKEY
 4  GXMAKE  QKEY
 5  GQMAKE  QKEY
 6  FILLWT  QKEY
 7  SETORD  QKEY
 8  SETALF  QKEY
 9  SETCBT  QKEY
10  MIXFNS  QKEY
11  SETABR  QKEY
12  SETCUT  QKEY
13  QCSTOP  QKEY
14  EVOLFG  USER
```

- When it sees a user keyword, the routine qcards calls the subroutine usub (provided by the user) to process the datacard

‘ EVOLFG 1 3.5 ‘

- `call qcards ( usub, ‘example.dcards’, 0 )`
- When the user key `EVOLFG` is encountered, the keyword and the parameter list are passed to `usub` for further processing

```
01      subroutine USUB ( key, nk, par, np, fmt, nf, ierr )
02
03      character*(*) key, par, fmt
04      external func
05      common /pass/ def(-6:6,12)
06      if(key .eq. 'EVOLFG') then
07        read(par,fmt,err=100,end=100) itype, q0
08        call evolfg( itype, func, def, iqfrmq(q0), eps )
09        return
10      endif
11  100 return
12       end
```

Yes, QCDNUM generates the FORTRAN format descriptor of your parameter list

- In this way you can write code to drive <u>everything</u> with datacards

New in next release V17-01/0h : Cuts

- Kinematic cuts can be a nice time saver in QCD fits
  - In the χ^2 loop, evolve only in the kinematic range of the data
  - After convergence open the cuts and evolve, only once, over the full range of the pdf set to be published
- In V17-00 a cut invalidates all pdfs in internal memory so that they have to be re-evolved with the new set of cuts
- In V17-01/0h, each pdf will know about its own range of validity so that different pdfs can have different cuts
- A cut will then only affect the pdfs evolved downstream (by `evolfg` or `evdglap`) and not anymore those evolved earlier or those imported from an external source

New in next release V17-01/0h : Intrinsic charm

- Allow the starting scale to be above the charm threshold in the VFNS (forbidden in V17-00)
- Cannot evolve backward over the threshold to $n_f - 1$
- Easy to implement by (internally) setting a cut at μ_c^2
- This would not be possible with the general cuts on all pdfs of V17-00, hence the effort to make the cuts local
- This also opens the interesting possibility to make all evolution parameters (order, α_s , etc.) a pdf attribute

Under development: toolbox tutorial

- Step-by-step building of your own evolution code (in LO)

E.1 How to partition a workspace

E.2 How to calculate weight tables

E.3 How to fill the α_s table

E.4 Singlet/gluon and non-singlet evolution

E.5 How to construct the singlet/non-singlet basis pdfs

E.6 Your own interpolation routine

E.7 How to compute a structure function

E.8 Make it robust and user-friendly

- The last two sections are still missing
- Fortran code in the testjobs directory and also on the web

To do after V17-01/0h

- Some bells and whistles (mainly toolbox error management)
- Finalise the tutorial
- Review the example jobs (also make them datacard driven)

- Release V17-01/00

- Make all evolution parameters a pdf attribute (?)
- Review time-like evolution of fragmentation functions
- Better handle on spline oscillation in backward evolution