LHAPDF 6

Andy Buckley, James Ferrando, Steve Lloyd,
Dave Mallows, Martin Ruefenacht, Karl Nordstrom
+Ben Page, Marek Schoenherr, Graeme Watt, Mike Whalley

PDF4LHC, CERN, 17 April 2013

LHAPDF has *finally* been rewritten in C++

. . . third time lucky!

The problem(s) with LHAPDF 5

- Fundamentally shared memory. Most sets' "common" block array workspaces are not really common.
  $\Rightarrow$ *huge* static memory requirements $\mathcal{O}(2\ \text{GB})$. Grid issues.
- Related: multiset, low-memory, etc. modes are hacks on top of fundamental situation.
  Speed, VMEM, flexibility, and correct operation all suffer depending on build-time configuration.
  - NMXSET determined at build-time: too restrictive for e.g. error set reweighting
  - Can get slow-down as num. member/set switches becomes large! (?)
  - Some functions don't (can't) respect multi-set indexing: alphaS, xMin, etc.
- Many different grid formats, and single-file sets add big parsing overhead (esp. for NNPDF).

LHAPDF 6 features

- **Ground-up rewrite (in C++) attempting to learn from and solve all these problems**
- **Key feature: dynamic allocation!** Allocate only what you use and no concurrency limitation
  - PDF member (1 PDF for each of several flavours) is fundamental object; `PDFSet` to be added for convenience
  - User takes control of memory. . . although our auto-management system could be made public
  - No more multi-set woes with alphaS, etc.
- Powerful "cascading" metadata system: system-, set-, and member-level info
  - uses standard YAML format (www.yaml.org)
  - fixes e.g. `Lam4/5` passing issues with LHAPDF5. Careful with backward compatibility!

LHAPDF 6 features (2)

So many features...

- **Maintainability: single PDF grid format and standard interpolators/extrapolators**
  - ipol/xpol specified at runtime via configuration metadata: flexible
  - one data file-per member data, dir per set: zero-overhead random member access, tarball distribution
  - arbitrary set of flavours supported, accessed by PDG ID code (so gluon is now 21, not 0^*, and photon flavour is trivial!)
  - distinct ipol grid blocks allow subgrids in Q.
  - ***releasing a new PDF no longer needs a new LHAPDF code release!***
  - also removes need for separate 100/1000 replica NNPDF sets...
- **Backward compatibility Fortran interface** (LHAPDF5/PDFLIB)
  - PYTHIA, Pythia8, Herwig++ tested so far. Sherpa (and some others) have a neat compatibility route
- Also new, full OO Python wrapper interface

Examples: system config

```
Verbosity: 1
Interpolator: logcubic
Extrapolator: nearest
ImplicitFlavorAction: return_zero
PwdInSearchPath: false
MZ: 91.2
MUp: 0.002
MDown: 0.005
MStrange: 0.10
MCharm: 1.29
MBottom: 4.19
MTop: 172.9
```

Examples: set info file

```
SetDesc: PDF fits using the standard CTEQ PDF evolution [...]
Authors: H.-L.Lai, M.Guzzi, J. Huston, Z.Li, P.M.Nadolsky, [...]
Reference: arXiv:1007.2241
NumMembers: 53
Flavors: [-5,-4,-3,-2,-1,1,2,3,4,5,21]
OrderQCD: 1
EvolutionNf: 5
ErrorType: hessian90
XMin: 1e-08
XMax: 1
Q2Min: 1.69
Q2Max: 1e+10
AlphaS_MZ: 0.117998
AlphaS_OrderQCD: 1
Lambda4: 0.326
Lambda5: 0.226
...
```

Examples: member data file

```
PdfType: central
Format: lhagrid1
---
1.000000e-08 1.214290e-08 1.474520e-08 1.790520e-08 ... [xs]
1.690000e+00 2.254442e+00 3.079814e+00 4.317128e+00 ... [Q2s]
-5 -4 -3 -2 -1 1 2 3 4 5 21                             [flavs]
0.000000e+00 0.000000e+00 5.253407e+00 6.215917e+00 ... [xfs]
0.000000e+00 1.868643e-01 5.367774e+00 6.316984e+00 ... [xfs...]
...
```

Examples: usage from C++

Single member:

```
#include "LHAPDF/LHAPDF.h"
...
LHAPDF::PDF* pdf = LHAPDF::mkPDF("CT10nlo", 0);
double xf_g = pdf->xfxQ(21, 1e-3, 126.0);
map<int, double> xfs = pdf->xfxQ(1e-3, 126.0);
size_t num_mems = pdf->numMembers();
delete pdf;
```

PDF set:

```
// (Using some nice C++11 features)
typedef unique_ptr<LHAPDF::PDF> PdfPtr;
vector<PdfPtr> pdfs;
for (size_t i = 0; i < num_mems; ++i)
  pdfs.push_back( PdfPtr(LHAPDF::mkPDF("CT10nlo", i)) );
for (const auto& p : pdfs) {
  double xf_g = p->xfxQ(21, 1e-3, 126.0);
```

`PDFSet` will be added to make this easier!

Examples: usage from Python

Single member:

```
>>> import lhapdf
>>> pdf0 = lhapdf.mkPDF("CT10nlo", 0)
>>> pdf0.xfxQ(21, 1e-3, 126)
31.199466144272378
```

PDF set:

```
>>> pdfs = [lhapdf.mkPDF("CT10nlo", i) for i in xrange(pdf0.numMembers)]
>>> len(pdfs)
52
>>> [pdf.xfxQ(21, 1e-3, 126) for pdf in pdfs]
[31.199466144272378, 31.10261967456719, ...
...
```

`PDFSet` will be added to make this easier!

Examples: C++ LHAPDF 5/6 compatibility

Use the `LHAPDF_MAJOR_VERSION` macro to handle C++ API differences:

```
#if defined LHAPDF_MAJOR_VERSION && LHAPDF_MAJOR_VERSION == 6

LHAPDF::PDF* pdf = LHAPDF::mkPDF("CT10nlo", 0);
cout << "xf_g = " << pdf->xfxQ(21, 1e-3, 126.) << endl;
delete pdf;

#else

LHAPDF::initPDFSet("CT10nlo", LHAPDF::LHGRID, 0);
cout << "xf_g = " << LHAPDF::xfx(x, 1e-3, 126.) << endl;

#endif
```

Memory

LHAPDF 5

```
$ size -B -d ~/heplocal/lib/libLHAPDF.so
text    data     bss     dec
1509082  142048 2039405376 2041056506
```

$\Rightarrow$ 1.5 MB functions, 140 kB data, **2 GB** uninitialised data!

LHAPDF 6

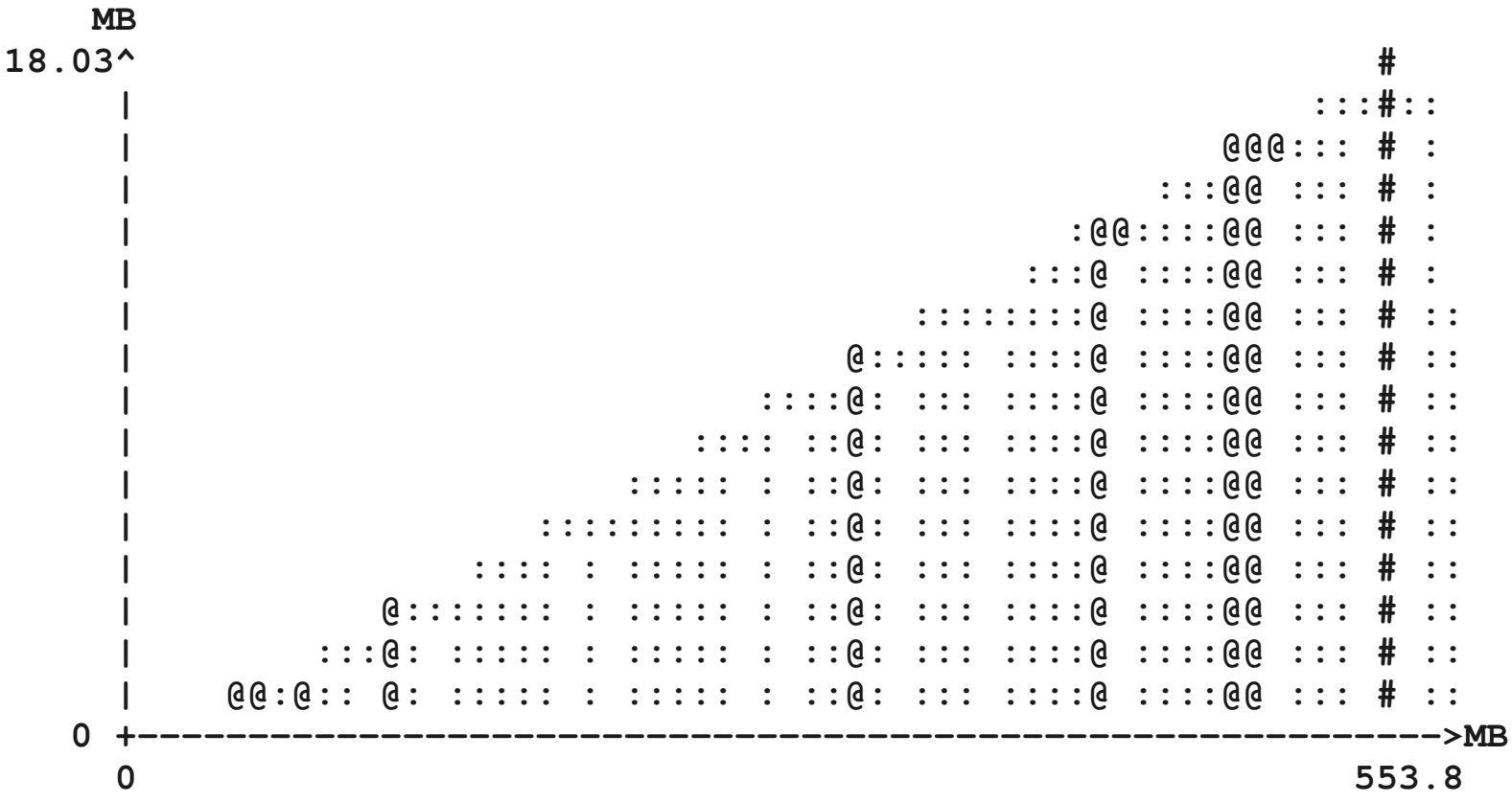
```
$ size -B -d ~/heplocal/lib/libLHAPDF.so
text    data     bss     dec
265310    8504    1552  275366
```

$\Rightarrow$ 2.6 kB functions, 8 kB data, **280 kB** uninitialised data!

WIN!

More on memory

From Valgrind's **`massif`** tool, for loading the whole CT10nlo set:

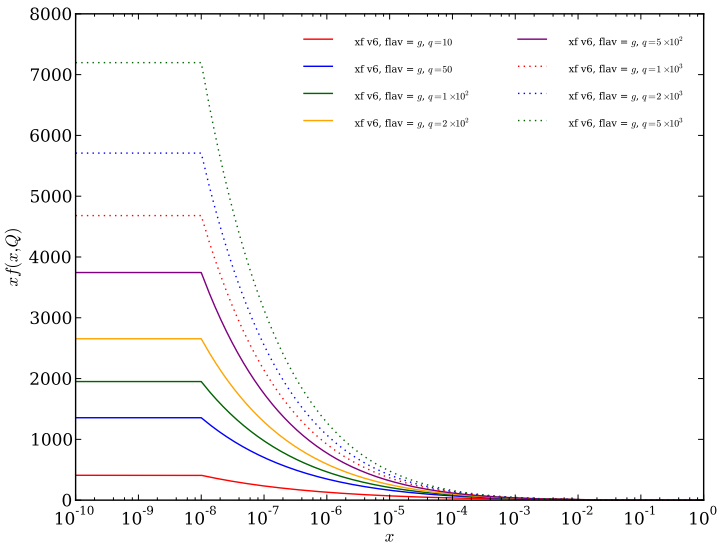


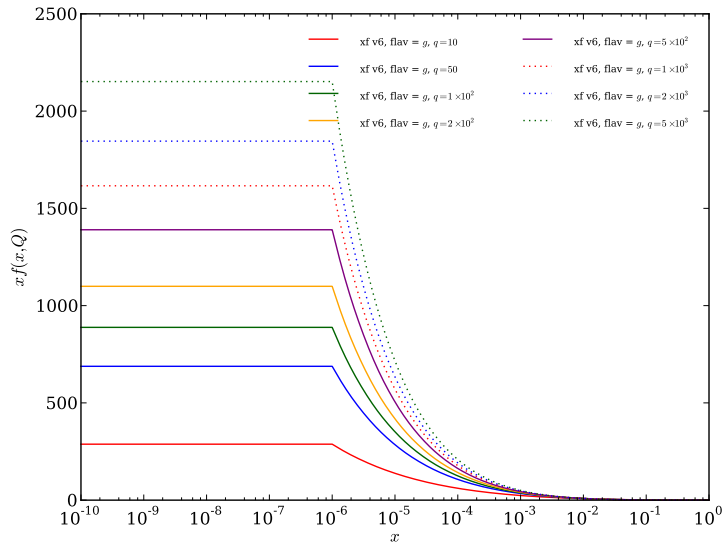
$\Rightarrow$ **$\sim$ 20 MB total.**

Set migration and validation

- We set a nominal LHA5 $\rightarrow$ 6 reproduction accuracy target of per-mille (1/1000)
- First sets for migration are CT10nlo and CTEQ6L1. Using original grid for CT10nlo, and log-bicubic ipol.

xf* vs. *x



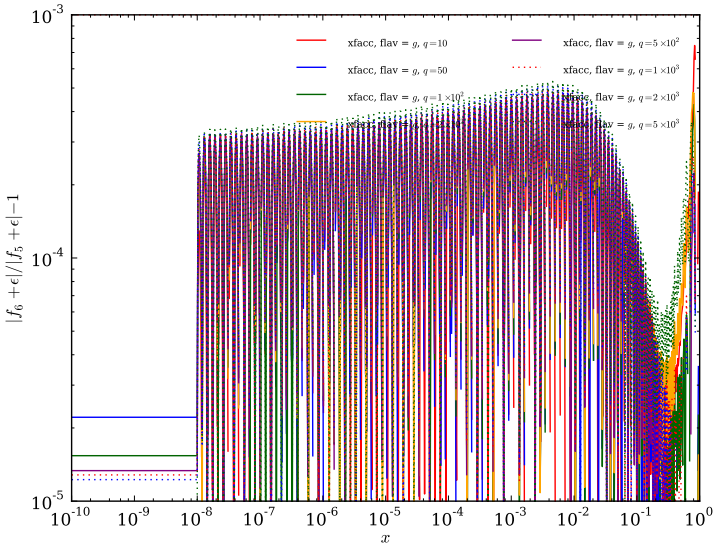


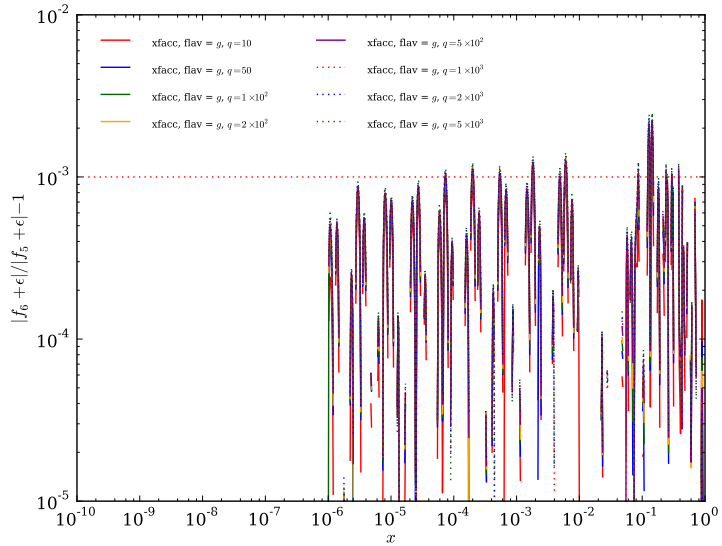
Sub-permille agreement everywhere in CT10nlo: grid spacing could even be *widened*! Difficult to get CTEQ6L1 to agree.

Set migration and validation

- We set a nominal LHA5 $\rightarrow$ 6 reproduction accuracy target of per-mille (1/1000)
- First sets for migration are CT10nlo and CTEQ6L1. Using original grid for CT10nlo, and log-bicubic ipol.

$\Delta xf/xf$ vs. x



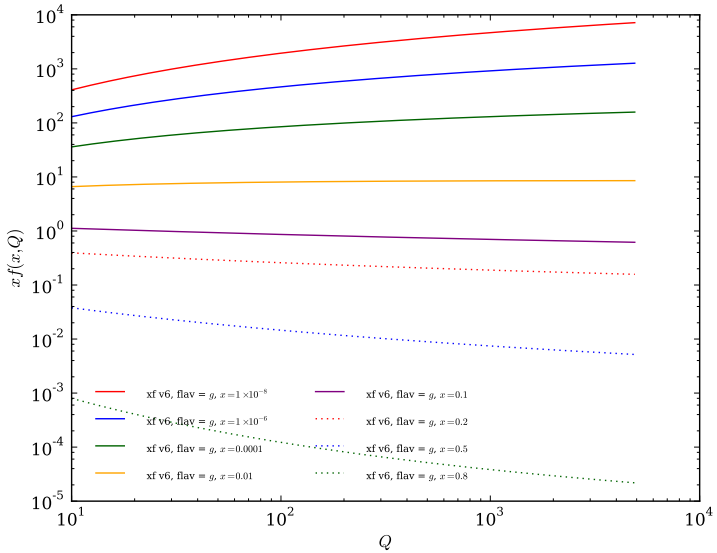


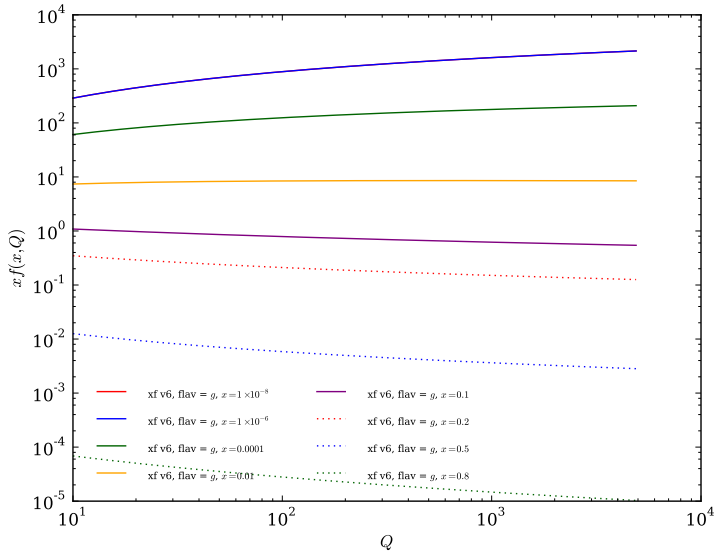
Sub-permille agreement everywhere in CT10nlo: grid spacing could even be *widened*! Difficult to get CTEQ6L1 to agree.

Set migration and validation

- We set a nominal LHA5 $\rightarrow$ 6 reproduction accuracy target of per-mille (1/1000)
- First sets for migration are CT10nlo and CTEQ6L1. Using original grid for CT10nlo, and log-bicubic ipol.

xf vs. Q



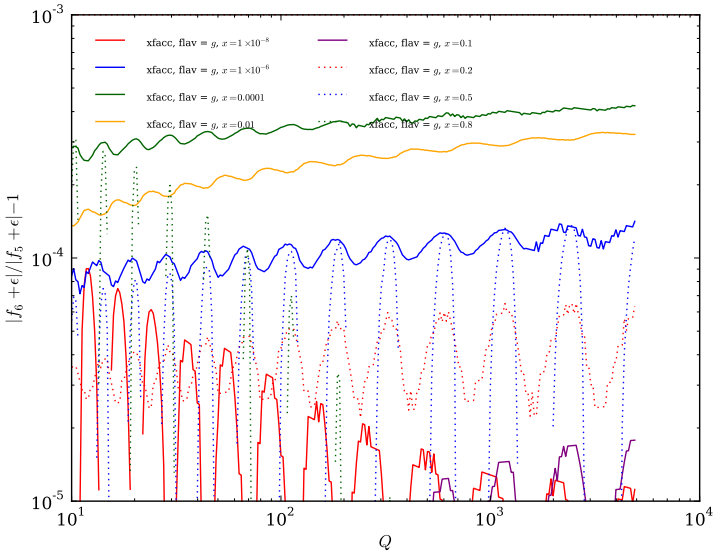


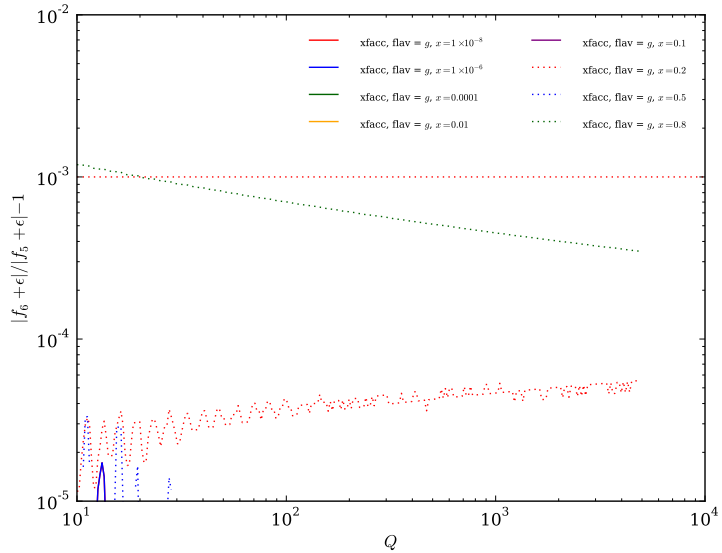
Sub-permille agreement everywhere in CT10nlo: grid spacing could even be *widened*! Difficult to get CTEQ6L1 to agree.

Set migration and validation

- We set a nominal LHA5 $\rightarrow$ 6 reproduction accuracy target of per-mille (1/1000)
- First sets for migration are CT10nlo and CTEQ6L1. Using original grid for CT10nlo, and log-bicubic ipol.

$\Delta xf/xf$ vs. Q





Sub-permille agreement everywhere in CT10nlo: grid spacing could even be *widened*! Difficult to get CTEQ6L1 to agree.

6.0.0beta1...and limitations

Beta #1 available to download now...or use direct from
`/afs/cern.ch/sw/lcg/experimental/lhapdf`

Limitations:

- No photon or nuclear PDFs yet (could be added: think about class design)
- Only grid PDFs (for now...and for forseeable future? Benefit from uniformity)
- Slightly larger files (CT10nlo is 28 MB vs. 21). But it's a generic format and we currently have no flavour-aliasing
- α_s system not yet ready/complete
- Feedback, suggestions and iteration of physics metadata needed ("RenFac", "Nf", "FlavorScheme", ...)

Summary and plans

- LHAPDF 6.0.0beta1 is available to try out now
- Complete rewrite should solve all (?) current limitations and offers new PDF release process, new possibilities for special members via metadata, etc.
- Backward compatibility with Fortran and with unique integer member IDs

Plans:

- Beta #2 in next 2 months. . . and another?
- Next migrations are MSTW2008 and NNPDF2.3. Requests/help? Intend full conversion?
- Improvements to high-x (and low-Q^2?) interpolation
- Profiling and performance improvement: caching, opportunistic parallelism?
- Versioning of data files? (Already an issue)
- Paper with full 6.0.0 release.