Data Format and Data Tools

S. Glazov, HERAFitter Meeting, CPPM, 13 Feb 2012.

Outline

- Overview
- `HERAFitter` from user point of view
- `HERAFitter` for developers.

HERAFitter: basic interface requirements

The software code is a mixture of C++ and Fortran codes. The core interfaces are provided in the Fortran part of the code.

- Use standard Fortran method to input information: `namelist` files.
- Central steering file to define input data, fitting parameters `steering.txt` containing several namelists.
- Inclusion of new data tables for existing processes should be possible without code recompilation. Data are provided as text files with a namelist header and the main body, as a table.
- Various models for treatment of correlated systematic uncertainties are available, steerable
- Inclusion of new theory in a standardized, modular way.
- Output contains basic text information to
  - Control consistency of the input data/fit parameters (error logging).
  - Report quality of the fit: χ^2, pulls, etc.
  - Report resulting PDFs: simple text and HERAGRID LHAPDF format.

Fortran Namlists

Input file:

```
&Data         ! Start with the &<Name>, comments start with "!"
   Name   = 'NC cross section'     ! Variables can be
                                   ! assigned directly
   NData = 145                     ! Type is defined in the fortran
                                   ! code

   ColumnType = 3*'Bin','Sigma',116*'Error' ! Arrays elements
                                   ! can be listed one after another
   DataInfo(2) = 'reduced'         ! or given explicitly.
&End
```

Fortran code:

```
C Data declaration part:
       integer NData
       character*32 Name,ColumnType(NColMax),DataInfo(NInfoMax)
       namelist/Data/Name,NData,ColumnType,

C Reading code ...
       open (51, file=fileName, status='old')
       read (51, nml=Data)
       close (51)
```

$\longrightarrow$ simple and flexible.

The `steering.txt` file

The `steering.txt` file controls behaviour of the program using several namelists.

- `&H1Fitter`: contains main steering cards, to be transferred to `QCDNUM`, to define PDF parametrization style, to specify data model for treatment of correlated uncertainties
- `&InFiles`: lists all input data files
- `&ExtraMinimisationParameters`: Non-PDF parameters to be minimized, e.g. α_S. The namelist can be repeated several times to add more parameters.
- `&Output`: Perform detailed PDF error analysis or not, specify text output parameters.
- `&Cuts`: Apply process dependent cuts.
- `&MCErrors`: Perform error analysis using Toy MC method
- `&lhapdf`: Use LHAPDF instead of QCDNUM, to calculate χ^2 for existing PDFs.
- `&HQScale`: Factorization scale for heavy flavours

Example 1: &InFiles namelist

```
&InFiles
  ! Number of intput files
    NInputFiles = 4

  ! Input files:
    InputFileNames = 'datafiles/H1ZEUS_NC_e-p_HERA1.0.dat',
                     'datafiles/H1ZEUS_NC_e+p_HERA1.0.dat',
                     'datafiles/H1ZEUS_CC_e-p_HERA1.0.dat',
                     'datafiles/H1ZEUS_CC_e+p_HERA1.0.dat',
```

(setting for the HERAPDF1.0 fit).

Data File Format

```
&Data
   Name   = 'NC cross section'     ! dataset label
   Reaction = 'NC e+-p'            ! reaction type - used to pick proper theory calculation
 ! Data table definitions:
   NData    =  145   ! 145 rows, corresponding to 145 data points
   NColumn =  120   ! 3 bins, sigma and 116 errors

 ! Layout of the data table columns: 3 bins, cross-section and 116 errors
 ! The following types are predefined: Bin, Sigma, Error and Dummy (case sensitive!)
  ColumnType = 3*'Bin','Sigma',116*'Error'

!   To treat error uncorrelately, then: first  is uncor, then the sys_i(i=1,114) -> 115 sources
             !     Bins       x-sec                           Errors
  ColumnName = 'x','Q2','y','reduced x-section','stat','uncor',110*'uncor',4*'ignore'

 ! options for bins: depends on Reaction type, for NC e+-p these are 'x', 'Q2' and 'y'
 ! options for the errors: 'stat', 'uncor', 'ignore' or different names for correlated uncertainties

 ! Extra information for the x-section calculation:
   NInfo     = 3
   DataInfo =  318.,        -1.,        0.
   CInfo     = 'sqrt(S)','e charge','e polarity'

   IndexDataset = 61

!   To take into account the correlations  uncomment below:
!   ColumnName = 'x', 'Q2', 'y', 'Sigma',
!                    'stat', 'uncor',  'h1', 'h2', 'h3', 'h4', 'h5', 'h6', 'h7', 'h8', 'h9', 'h10',
....
!                    'h101', 'h102', 'h103', 'h104', 'h105', 'h106', 'h107', 'h108', 'h109', 'h110',
!                    'hproc1', 'hproc2', 'hproc3',    ! procedural errors
!                    'hcor_lum',                      ! common lumi error

!   To treat the uncertainties as absolute use "false"
   Percent      = 116*true
&END
*  X         Q2         Y         Sigma      Stat       Uncor     h1      h2      h3    h4     h5     ...
0.0000320 1.5000000 0.5196785 0.8208551 1.1749068 3.2026455 0.0000 -0.4089 0.1332 0.0057 0.0228  ...
```

Data cuts

Reaction dependent cuts, for variables used to bin the data, can be applied directly in the `steering.txt`:

```
*
* Process dependent cuts
*
&Cuts

  !--------------------- NC ep  --------------------------

  ! Rule #1: Q2 cuts
   ProcessName(1)     = 'NC e+-p'
   Variable(1)        = 'Q2'
   CutValueMin(1)     = 3.5              ! cut data below Q^2=3.5 Ge
   CutValueMax(1)     = 1000000.0        ! no upper cut
...
```

At the moment it is impossible to apply cuts based on a function of bining variables without modifying the code.

Data Model

Data are reported differentially, with up to 10 dimension. The measurements are given with their statistical, systematic uncorrelated and correlated uncertainties.

Several χ^2 models are available, steerable by the `CHI2Style` parameter. For example, `CHI2Style = 'HERAPDF'` turns on default HERAPDF treatment:

$$\chi^2_{\mathrm{exp,cor}}(\mathbf{m}, \boldsymbol{b}) = \sum_i \frac{\left(m_i\left[1 - \sum_j \gamma_i^j b_j\right] - \mu_i\right)^2}{\delta^2_{i,\mathrm{unc}} m_i^2 + \delta^2_{i,\mathrm{stat}} \mu_i m_i} + \sum_j b_j^2 \, .$$

The minimization uses that for each iteration χ^2 has quadratic dependence on the nuisance parameters describing systematic uncertainties. Thus minimization vs **b** is performed analytically at each iteration while minimization vs PDF parameters is performed by the `MINUIT` package.

Control of systematic uncertainties

Types of systematic uncertainties are defined in data files. Three types are distinguished: statistical (`'stat'`), uncorrelated systematic (name contains `'uncor'` sub-string) and correlated systematic (any other name). Systematic uncertainties are correlated among different data files if they have the same name. After minimization, shifts and estimated reduction of uncertainty for the nuisance parameters corresponding to the systematic error sources are reported in the `output/Results.txt` file:

```
 Systematic shifts                        (shift)       (uncertainty)
    1 h1                                   0.0000   +/-    1.0000
    2 h2                                  -1.3537   +/-    0.8232
    3 h3                                   0.0933   +/-    0.9942
    4 h4                                  -0.5063   +/-    0.8667
    5 h5                                   0.3228   +/-    0.9906
    6 h6                                   0.2496   +/-    0.9277
```

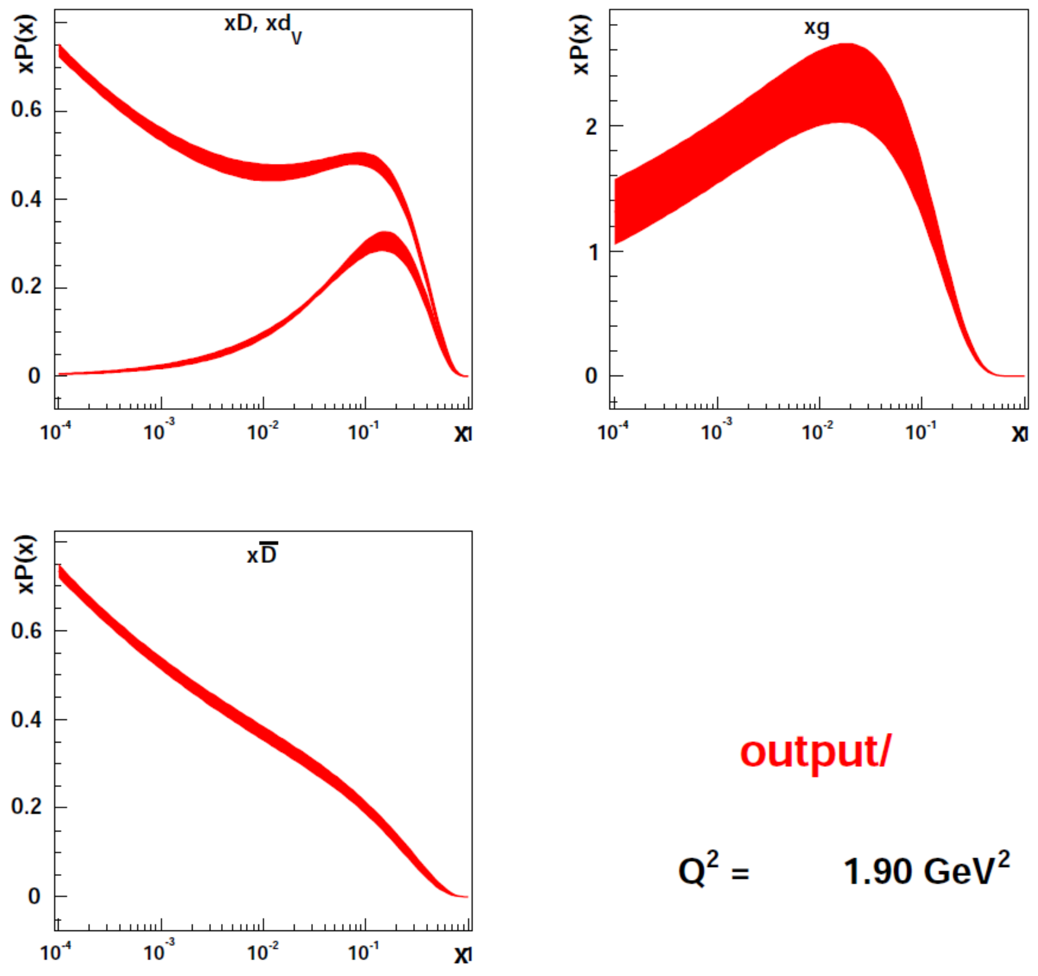
PDF uncertainties

The χ^2 minimization is based on `MINUIT` package. The minuit steering cards are included using `minuit.in.txt` file, they follow standard `MINUIT` syntax. `HESSE` reports improved error estimation and correlations among PDF parameters

The package includes extension of the `MINUIT` package from J.Pumplin which allows to perform detailed error analysis of PDFs and report full errors in terms of eigenvectors. This is steerable by `DoBands` namelist parameter.

In addition the fitter has code to perform error analysis using toy MC method. The method uses uncertainties as reported in the data tables and can generate replica around the actual data values, to study error propagation, and around the predicted theoretical values, to study biases from the χ^2 definition.

Example of “doBands” output



The text output files can be converted to `root` graphs/plots using `DrawResults`, `StoreGraphs` and `StoreGraphsMC` tools.

Relation between data and theory

The theory predictions for a given data file are controlled by the `Reaction` namelist variable which gives string name of the reaction.

At each iteration, a loop is run over the data files and the theory calculations are matched to the data inside `GetTheoryForDataset` subroutine:

```
elseif (DATASETREACTION(IDataSet).eq.'ttbar') then
  Call GetHathorXsection(IDataSet)
elseif ...
```

Additional information for the theory calculations can be given by given in data files using `NInfo` block, e.g.:

```
! Extra information for the x-section calculation:
  NInfo    = 4
  DataInfo =  318.,        -1.,   1.,   0.
  CInfo    = 'sqrt(S)','e charge','reduced','e polarity'
```

Overall behaviour of the theory calculations can be controlled in `steering.txt` file.

Theory module coding example

```
C
C Get indexes for Q2, x and y bins:
C
      idxQ2 = GetBinIndex(IDataSet,'Q2')
      idxX  = GetBinIndex(IDataSet,'x')
  ! -> index of abstract bins is determined by names
....
       do i=1,NDATAPOINTS(IDataSet)
  ! -> loop over all data points in a dataset
C Reference from the dataset to a global data index:
C
         idx =  DATASETIDX(IDataSet,i)
  ! -> get index in the global data array
C
C Local X,Y,Q2 arrays, used for QCDNUM SF caclulations:
C
         X(i)   = AbstractBins(idxX,idx)
  ! -> get x-bin value.
...
      enddo
```

How to add a new theory module

- Extra steering parameters can be added to the `&H1Fitter` namelist, in `read_steer.f` file. Add your own new namelist if needed.
- Add initialization call in `init_theory.f`, see subroutine `init_theory_datasets`.
- Add call to your theory module in `theory_dispatcher.f`, see `GetTheoryForDataset`
- Optionally, add pre-calculation at each iteration using `GetTheoryIteration` in `theory_dispatcher.f`.

Error logging

Recently `HERAFitter` got a simple error logging facility, based on an H1 tool. The code allows for 4 levels of errors (I,W,S,F), with a stearable printount, and sorted summary of the messages at the end of the run.

```
           *********************************
           ***       Error Summary       ***
           *********************************

Total number of logged errors:           9
Total number of errors not recorded:     0


List of errors sorted by severity level:

*-------------------------------------------------------------------------------
* Module |   Error |   Error |
*   Name |    Type |   Count |   Error Description
*-------------------------------------------------------------------------------

 Informational messages:
*-----------------------
 H1Fitter  12020501        1 I: steering.txt has been read successfully
 H1Fitter  12020502        1 I: data tables have been read successfully
 H1Fitter  12020503        1 I: theory modules initialised successfully
 H1Fitter  12020504        1 I: read minuit input params from file minuit.in.txt
 H1Fitter  12020515        4 I: FCN is called

 Warning messages:
*-----------------------
 H1Fitter  12020601        1 W: DUMMY version applgrids_dum called!
```

Building together with external code

The `HERAFitter` code uses `automake` tools to configure and build the package. Minimal coupling with external packages is preferred. Right now the pre-requested packages are high energy physics specific `CERNLIB, ROOT, QCDNUM` as well as `GSL, BLAS`. Additional packages, e.g. `APPLGRID` can be linked to as a configuration option:

```
autoreconf --install
./configure --enable-applgrid
make
make install
```

It is planed to remove dependence on `CERNLIB` for future builds. It is probably also desirable to remove dependence on `ROOT` for the minimal installation

HERAFitter for benchmarking

`HERAFitter` package can be used to benchmark various theory approaches:

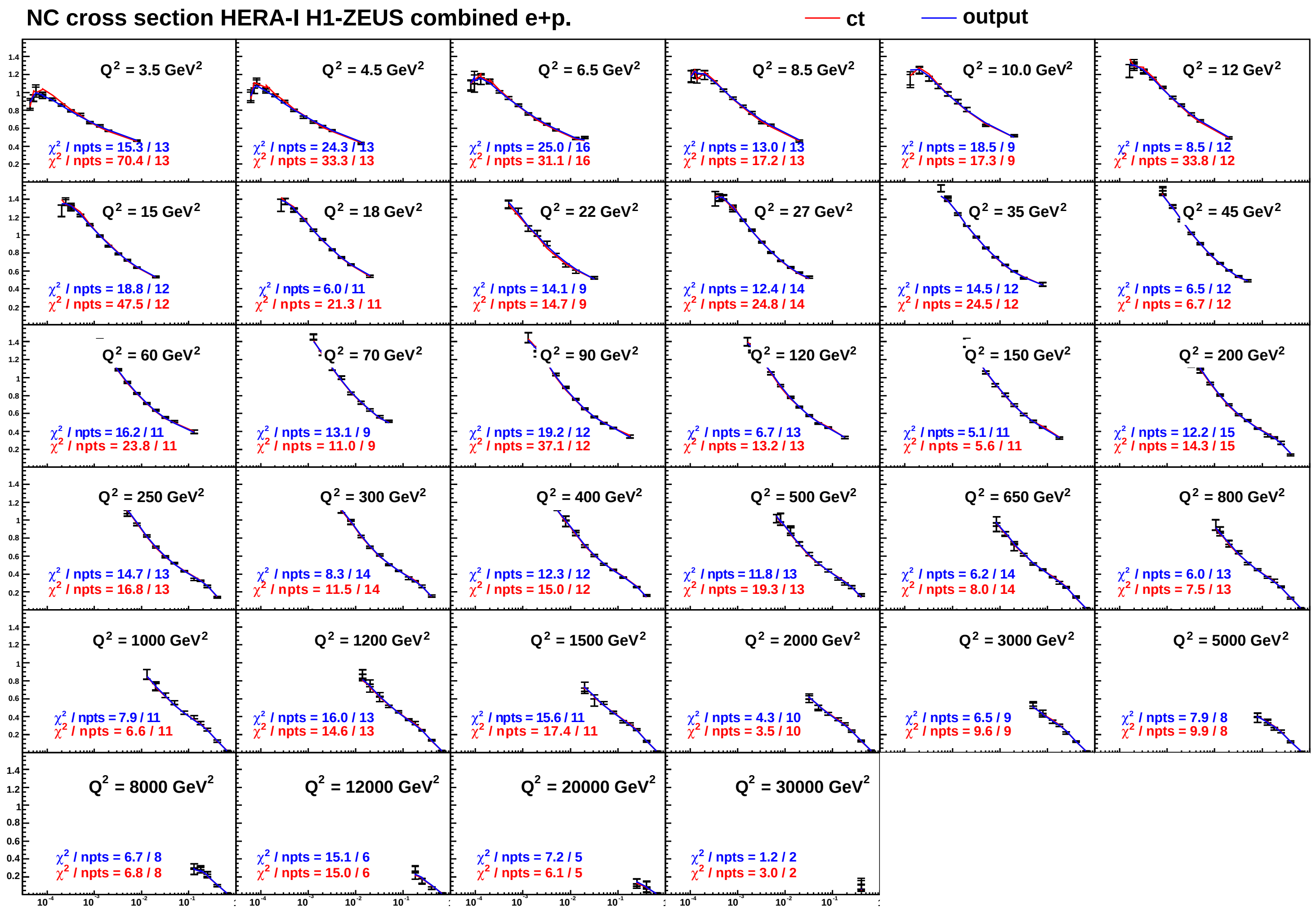
- Different theory calculation modules can be implemented, compared to each other
- Using LHAPDF interface, PDFs from variuos groups can be read in and combined with different theory calculations.

Example how to use CTEQ6.6 set:

```
&H1Fitter
  ...
  PDFStyle = 'LHAPDF'  ! or PDFStyle = 'LHAPDFQ0'
  ...
&End

&lhapdf
  LHAPDFSET  = 'cteq66.LHgrid' ! LHAPDF grid file
  ILHAPDFSET = 0               ! Set number withing PdfSet
&End
```

CTEQ6.6 vs HERA1.0 data



CTEQ6.6 using RT-scheme to calculate heavy flavour contribution, χ^2 calculated usign HERAPDF prescription.

Summary

- `HERAFitter` is a modular code with reduced depenence to external packages.
- Data files can be added without code recompilation.
- New theory modules can be added in a modular way.
- `HERAFitter` can be used for PDF benchmarking.

The code is open for further developments. Your input/suggestions are very welcome !