MINT Tutorial

Outline

- Directory Structure, compilation etc
- Named Variables
- The Minuit Interface

Before you start

- Make sure you've got access to root. Type

```
root -l &
```

- If that starts root, you're fine - just stop root (by typing `.q`) and go to the next page. Otherwise you will need set root up. That's not easy at CERN.

- This is one less than elegant way that works:

```
setenvDaVinci

source $DaVinci_release_area/DAVINCI/DAVINCI_v20r2/Phys/DaVinci/
v20r2/cmt/setup.csh
```

Directory Structure, compilation, etc

- Different groups of classes are put into some directory, say NamedParameter
- All source code is stored under the src/ directory. All header files are under Mint/ and all test files are under test/ in a mirror directory structure to src/
- Each directory contains a subdirectory called src with the source code. t Testing code, is stored under test/ . Only the code in the test directories contains a main() function and can be compiled.
- To compile, go the test directory of interest and type make.

Make your own

- The code that you have now will certainly change in the near future.
- To make this less painful it would be wise not to put any of your own code into any existing directory (at least not code that you want to keep).
- Make your own directory:

```
mkdir myOwn
mkdir myOwn/src
mkdir myOwn/test
cd myOwn/test
cp ../../NamedParameter/test/Makefile .
```

Named Parameters

- A NamedParameter can be initialised from a file. By default, parameters are read from stdin.
- Declare like this:

```
NamedParameter<double> myDble("myDble");
NamedParameter<int>    Nevents("Nevents");
NamedParameter<string> ampName("Amplitude Name");
```

- Initialise in a file like this:

```
myDble 27.0
Nevents 280
“Amplitude Name”  “B to D pi”
```

- You can also initalise values in the C++ source code, and specify a parameter file to read from instead of stdin - see the task on the next slide.

NamedParameters Hands-On

- Go to test/Mint/NamedParameter/test/
- Look at testNamed.cpp - it contains the code.
- Look at testNamed.txt - it contains the parameter initialisations.
- type "`make`" to compile the code (if you get incomprehensible errors, try "`make clean`" first, and then again "`make`")
- type: `./testNamed  < testNamed.txt`
- See if you understand the output. Add your own parameter. Try initialising it in the code. Try reading it from sth else than stdin.

The Minuit Interface

- Minimiser is the Minuit interface. In fact, Minimiser inherits from TMinuit, so you can use all TMinuit command.
- The difference is mainly how fit parameters are passed to Minuit. This is done with the class FitParameter
- FitParameter is a kind of NamedParameter and you initialise your fit parameters in a file.
- Otherwise you use your FitParameters in your likelihood like any other double.
- You do not need to keep track of how parameters are passed around in Minuit.

Minimiser Hands-On

- Go to test/Mint/PdfAndLogL/test
- type make
- run the code with toyFit < toyFit.txt
- Read the code and the output file. Understand what's going on.
- Add a likelihood scan to the code
- Write your own toy fit, say for a Gaussian.

Old-style Scanning code

```
Double_t arglist[100] = {0};
Int_t ierflg=0;
TFile fscan("scanTau.root", "RECREATE");
arglist[0]=0;
mini.mnexcm("SCAN", arglist, 1, ierflg);
TGraph *gr1 = (TGraph*)mini.GetPlot();
if(0 == gr1){
  cout << " didn't get plot " << endl;
}else{
  cout << "got plot!" << endl;
  fscan.cd();
  gr1->Write();
}
fscan.Close();
```

(the point of this exercise is to demonstrate that you can use TMinuit's functionality from MINT's Minimiser)

New-Style Scanning

```
* Assuming you start with your ASCII text file where you
* initialise your FitParameter (say its name is x) like
* this:

* Name            fix?    value            error    min      max

  x               0       1                0.1

* Now just add a line <parameter name>_Scan <from> <to>
* like this:

  x_Scan -0.5 0.5
```

Optional extras

- If you want to know how to write a likelihood w/o using Neg2LL, try out the code in Minimiser/test.
- Run ./testMinimiser < testMinimiser.txt
- Check out main() to see how to use NamedParameters to modify the behaviour of the program.