

Notes on the **FSRoot** Package

Ryan Mitchell

February 24, 2019

Abstract

FSRoot is a set of utilities to help manipulate information about different Final States (FS) produced in particle physics experiments. The utilities are built around the CERN **ROOT** framework. This document provides an introduction to **FSRoot**.

Contents

1	Installation and Initial Setup	1
2	Conventions	2
3	Basic Histograms	2
4	Fitting Utilities	3

1 Installation and Initial Setup

Instructions for installation and initial setup:

(1) Download the source:

```
> git clone https://github.com/remitch66/FSRoot.git FSRoot
```

(2) Set the location of **FSRoot** in your login shell script (e.g. `.cshrc`):

```
setenv FSROOT [xxxxx]/FSRoot
```

(3) Also probably add the **FSRoot** directory to `$DYLD_LIBRARY_PATH` and `$LD_LIBRARY_PATH`. This allows you to compile code including **FSRoot** functions. For example:

```
setenv DYLD_LIBRARY_PATH $DYLD_LIBRARY_PATH\: $FSROOT
setenv LD_LIBRARY_PATH $LD_LIBRARY_PATH\: $FSROOT
```

(4) There is usually a `.rootrc` file in your home directory that ROOT uses for initialization. Add lines like these to `.rootrc`, which tell ROOT the location of FSRoot:

```
Unix.*.Root.DynamicPath: .:$(FSROOT):$(ROOTSYS)/lib:
Unix.*.Root.MacroPath:   .:$(FSROOT):
```

(5) Now when you open ROOT, the FSRoot utilities should be loaded and compiled – you should see a message saying "Loading the FSRoot Macros" along with the output of the compilation. The loading and compiling is done in the file `rootlogon.C`. This file also sets up default styles, which are not essential. Since these might conflict with styles you have defined elsewhere, it could be worthwhile to tweak or remove these.

2 Conventions

Some FSRoot operations on variables within a ROOT TTree assume a particular format within the tree. Four-vectors are assumed to take the form:

`[AB]EnP[CD]`, `[AB]PxP[CD]`, `[AB]PyP[CD]`, `[AB]PzP[CD]`

where `[CD]` is a particle label (often "1", "2", "3", "2a", etc., but also "CM" or "B" or other) and `[AB]` labels the type of four-vector (for example, "R" for raw or "K" for kinematically fit or "MC" for Monte Carlo, etc.). The FSRoot code does not assume anything about the conventions for `[CD]` and `[AB]`.

Variable names for run numbers, event numbers, and the χ^2/dof from a kinematic fit are also hardcoded in a couple of places:

`Run`, `Event`, `Chi2DOF`

Maybe a future version of FSRoot could make these formats customizable.

3 Basic Histograms

Basic histogram functions are provided by the `FSHistogram` class in the `FSBasic` directory. Like many other functions within FSRoot, the functions within the `FSHistogram` class are static member functions, so there is never a need to deal with instances of `FSHistogram`.

The basic functionality is through the `FSHistogram::getTH1F` and `FSHistogram::getTH2F` classes. Here are example uses that can be run from either the ROOT command line or from a macro (see `Examples/Intro/intro.C`). The first draws a 1d histogram and the second draws a 2d histogram. The third argument is the variable to plot; the fourth holds the number of bins and bounds.

```

> FSHistogram::getTH1F(FileName,TreeName,"Chi2DOF","(60,0.0,6.0)","")->Draw();
> FSHistogram::getTH2F(FileName,TreeName,"Chi2DOF:Event",
    "(100,0.0,1.0e6,60,0.0,6.0)","")->Draw("colz");

```

Cuts can be added in the fifth argument:

```

> FSHistogram::getTH1F(FileName,TreeName,"Chi2DOF",
    "(60,0.0,6.0)","Chi2DOF<5.0")->Draw();

```

The variable and cut arguments can contain shortcuts. For example, `MASS(I,J)` is expanded into the total invariant mass of particles I and J, where I and J are not necessarily numbers (they are the [CD] in Sec. 2). Characters in front of `MASS` (for example) are prepended to the variable names (the [AB] in Sec. 2). This is all done in the member function `expandVariable`, which can also be used on its own to explicitly see what it is doing. See `FSHistogram::expandVariable` for more options beyond `MASS` (for example, `RECOILMASS`, `MOMENTUM`, `COSINE`, etc.). Examples:

```

> FSHistogram::getTH1F(FileName,TreeName,"MASS(1,2)",
    "(60,0.0,6.0)","Chi2DOF<5.0&&MASS(1,2)>1.0")->Draw();
> cout << FSHistogram::expandVariable("MASS(1,2)+MASS(2,3)") << endl;
> cout << FSHistogram::expandVariable("ABMASS(1,C)") << endl;
> cout << FSHistogram::expandVariable("RECOILMASS(CM;D,2)") << endl;

```

Histograms are automatically cached so they are made only once. To save histograms at the end of a session, use the function:

```

> FSHistogram::dumpHistogramCache();

```

To read in the cache at the beginning of a session:

```

> FSHistogram::readHistogramCache();

```

To clear a cache from memory during a session:

```

> FSHistogram::clearHistogramCache();

```

To see more verbose output during a session:

```

> FSControl::DEBUG = true;

```

4 Fitting Utilities

The fitting utilities (contained in the directory `FSFit`) work, but are still under development. See the examples in the directory `Examples/Fitting` to get the general idea.