How to create and fill a tree
How to access and analyze a tree

Anna Simon

November 18, 2015

Creating a data tree

- Understand the structure of the event
- Declare variables that will store the event data

```
int multiplicity;
double energy[20];
```

- Declare and initialize an output ROOT file, a tree and branches that will be filled with the event data

```
TFile *fileOut = new TFile("data.root","RECREATE");
TTree *t = new TTree("t","simple root tree");
t->Branch("multi",&multiplicity,"multiplicity/I");
t->Branch("ene",energy,"energy[20]/F");
```

Creating a data tree

- Loop over the data file to populate the variables, execute the tree Fill() command for each event

```
while (!fileIn.eof()){
    cin >> energy[0] >> energy[1] .... ;
    multiplicity=0;
    for(int i=0; i<20; i++){
        if (energy[i]>0) multiplicity++;
    }
    t->Fill();
}
```

- Write the ROOT file to disk

```
fileOut->Write();
```

Writing a data analyzer

- Inspect the structure of the data tree (here two variables, one int and one float)
- Declare the variables that will store data from the tree
- Open the data file, create a pointer to the tree and set the branches addresses

```
int multiplicity;     // the identifier do not have to be the
float ene;     // same as the names of the branches
TFile *fileIn = new TFile("../read_spectra/data.root");
TTree *t = (TTree*)fileIn->Get("t");
t->SetBranchAddress("ene",ene); // name of the branch, address of the variable
t->SetBranchAddress("multi",&multiplicity);
```

- Create an output file and ROOT structures within it

```
TFile *fileOut = new TFile("histograms.root","RECREATE");
TH1F *hMult1 = new TH1F("hMult1","mult 1 events", 1000,0,1000);
TH1F *hMult2 = new TH1F("hMult2","mult 2 events", 1000,0,1000);
```

Writing a data analyzer

- Loop over the whole tree, read event-by-event to populate the variables

```cpp
int nEntries = t->GetEntries();
float totalEne
for (int i=0; i<nEntries; i++){
   t->GetEntry(i);
   for (int j=0; j<20; j++){
      totalEne = totalEne + ene[j] ;
   }

      if (multiplicity==1) hMult1->Fill(totalEne);
      if (multiplicity==2) hMult2->Fill(totalEne);
      totalEne=0; //zero your variables when done using them!
}
```

- Write the output file to disk

```cpp
fileOut->Write();
```