UTA HEP/D0/0001
Version 2.0
August 29, 2003

An Introduction to D0 Data Analysis at UTA

Barry S. Spurlock
The University of Texas at Arlington

Introduction

The goal of this document is to provide simple instructions that will give a novice the ability to start D0 data analysis at UTA. The documentation normally available on this subject is typically aimed at completeness. This type of documents is very useful to those that have a general idea about what they are doing. However, this is of limited use to the novice, who will suffer from having an excess of information available. This overload leads to a great deal of confusion. This document will attempt to bridge that gap, providing a concise introduction that will give a novice a general idea about what he (or she) will be doing. My motivation for writing this is that this is exactly the kind of document that I would have liked to have when I got started doing data analysis a short time ago.

I begin by explaining how to get access (accounts) to the computer resources both at UTA and Fermilab/D0. Then, I provide many of the basics that you will need to function in Linux. This is necessary since all of the analysis packages are designed for Linux. Next I discuss how to find data to analyze, and how to create a "dataset definition" that will allow you to access this information. Then I go over how to download "thumbnail" files from Fermilab, and how to convert them to root files. Once root files are available, I conclude the first section by showing how to use an existing program to generate histograms and explaining how to find them. I also suggest where one might look for more information on the variable used in root.

Obtaining Access to Linux Machines from a Windows PC

Data analysis at UTA can be performed on a number of computers operating in Linux. Consult with the HEP group's software specialist (currently Mark Sosebee [1]) about which machine to use, and set up an account on that machine. The software specialist will provide you with an ID and password that will enable you to log on to that particular computer. At the same time, you should make sure that you also get the IP address (node name) for that computer. For example, I began work on one of the farm computers (node name: heppc14.uta.edu). The immediate goal is to allow you to log on from the Windows PC at your workstation/desk. In order to do this you will need "Exceed" and "PuTTy". This software can also be obtained from the HEP software specialist. To log on to the machine do the following:

1) Run Exceed.

   An Exceed box will appear momentarily on your screen, then a small Exceed box will remain on the status bar at the bottom of your screen.

2) Run PuTTy.

A PuTTy configuration box will appear on your screen. For the host name, insert the IP address (node name) of the machine you intend to work on. Just below, select "SSH" (Secure Shell) for the protocol. This will automatically select the correct port. At this point it is convenient to save this information. Go to the box beneath "Saved Sessions" and type in a name (for example I named the session "heppc14"). Once you save the session, the session name will appear in the list below the "Saved Sessions" box. In the future you can then select the appropriate name and click the "Load" button. This will fill in the correct IP address and protocol, saving you from having to remember and type the address. Once the IP address and protocol have been entered, click on the "Open" button at the bottom of the screen. This opens up a window that will allow you to login. On top of the window you will see the computer's IP address. Below this will be a prompt with "login as:"

3) Login.

The curser will already be next to the line that says "login as:". Just type in the user name you acquired from the HEP software specialist, and hit enter. It will then ask you for your password. Type in the password you were given and hit 'enter'. It will display the date and time that you last logged in, followed by a prompt which begins with "your username" @ "the computer's IP address" followed by the name of the directory that you are currently working in. It ends with a prompt.

You are now ready to work. At this point (the first time you log on), you should change your password. To do this, type in the command "yppassword". It will ask you to verify your old password, and then ask you for the new password. It is worth noting that you can log in as many times as you desire, each time opening up a new window. Having more than one window is very useful as it allows you to use one window to modify your program while a second window is running root where the program will be executed. Jumping in and out of root is tedious.

Obtaining Access to Linux Machines from a Linux PC

As with the windows machine, you need to consult with the HEP group's software specialist to determine which machine to use and obtain the IP address name of that computer, log on ID, and password. Then logging in to the machine you will use is relatively easy. First, log on to the Linux PC at your workstation/desk. Then, at the prompt, type in "ssh" followed by a space and then the IP address of the computer you wish to work on. It will then ask you for your password. Type in the password you were given and hit 'enter'. It will display the date and time that you last logged in, followed by a prompt which begins with "your username" @ "the computer's IP address" followed by the name of the directory that you are currently working in. It ends with a prompt.

Getting Started in Linux

If you followed the instructions above, you are now ready to work. The next thing to do is some standard commands to set up things you will need. In order, run the following commands:

. /fnal/ups/etc/setups.sh
setup D0RunII p13.08.00
d0setwa
export DISPLAY=*IP address number*:0.

The first command (. /fnal/ups/etc/setups.sh) will make it possible for you to use software packages (setups) from D0.

The second command (setup D0RunII p13.08.00) instructs your computer to use a specific software release (in this case p13.08.00), and obtains access to the D0 software. The D0 software is continually modified, and new versions are given new numbers. For example, the release following p13.08.00 would be p13.09.00. Using p13.08.00 is acceptable, but it is best to use the most recent software release. To modify this command simply use the number of the most recent release in place of p13.08.00.

The third command (d0setwa) instructs your computer to "set working area". For our purposes the exact nature of this command is unimportant. What is important is that some of the D0 software will fail without having previously executed this command.

The fourth command (export DISPLAY=*IP address number*:0.) redirects the graphic output from the working computer to your desktop. You must, of course, replace the Italics with the actual IP address number from the computer at your workstation/desktop. In my case, the computer on my desk is HEPPC33, and 129.107.83.30 is its IP address number. Consequently, when I use the export command it looks like this:

export DISPLAY=129.107.83.30:0.

or you could do:

export DISPLAY=heppc33.uta.edu:0.

Linux Basics

At this point, I should introduce a few of the most common Linux commands that you will need to navigate through Linux.

Command	Results
cd *subdirectory*	Moves you into the chosen subdirectory (down the directory chain)
cd ..	Moves you into the higher directory (up the directory chain)
ls	Lists the files inside the current directory
ls –s	Lists the files inside the current directory and their size in kB
pwd	Shows the current directory chain.
rm *filename*	Removes (i.e. deletes) the chosen file.
rm –rf *subdirectory*	Removes (i.e. deletes) the chosen subdirectory
mkdir *subdirectory*	Creates a new subdirectory
mv *oldname newname*	Moves and/or renames a file. The names should include the entire directory chain (ex: /home/spurlock/*filename*). To select the current working directory use ". /" (ex: . /*filename*).
cp *oldname newname*	Copies a file. This works like "mv" above.

mozilla &	Opens up a web browser for internet access
↑	Places the most recently used command on the command line. Repeating this command will then call up the command used just before the most recent command. Repeated use can allow you to call up any recently used command.
ssh *username@IPaddress*	Commence working on a new computer. This will require you to login on that computer. For example, "ssh *spurlock@hepfm004.uta.edu*" would start me up on the computer named hep farm 4.
exit	Ends your session (i.e. logs you off).
gedit *filename*	Edit (modify) the file named *filename*

This should be enough Linux to get you started. You'll pick up more as you go along.

Fermilab D0 Accounts

Now that you have a bit of working knowledge of Linux, you need some data to work on. The data is at Fermilab, and in order to access that data you'll need to get a Fermilab ID and Fermilab D0 accounts. To get a Fermilab ID follow the instructions at:

http://www-d0.fnal.gov/computing/systems/id-ver.html

Getting the ID will involve filling out and submitting (mailing) a form, so this may take a few days to complete. Once you've got a Fermilab ID, you can open up D0 computing accounts. The forms to get a Fermilab D0 account can be found at:

http://www-d0.fnal.gov/computing/systems/d0_account.html

First, you must have a Fermilab wide account. Since the professors in the UTA HEP group make frequent trips to Fermilab, it is best if you request that one of them pick up your cryptocard. The most important account for our purposes is the D0 web access (listed under the heading "D0 Unix"). This will give you access to the data. If you plan on being at Fermilab at some point in the future, you should also get the D0 central analysis systems account (D0mino) and ClueD0 desktop account (also listed under the heading "D0 Unix").

Acquiring Data

The first step towards having data to work on is to find some data. I will show you how to look for 'skimmed' data from the physics working groups. After finding some data to work on, I will use SAM, a system used to access the data for D0. On the SAM web interface I start by creating a dataset definition, which stores a description of the desired data that can later be used to extract data files. Once the definition has been made, I will use some existing programs to bring data from Fermilab to UTA in the form of 'thumbnail' files. At this time analysis at UTA is being done on 'Root' files rather than 'thumbnail' files; consequently I will have to convert from thumbnail into root. Once the data is in the form of root files I can then begin writing code for data analysis.

As an example of the entire data analysis process I will do a search for Z bosons decaying into a pair of muons of opposite charge. I keep this in mind as I start to look for data sets from the physics working groups.

Finding Data Sets

This next part might seem a bit confusing, but I decided to trace step by step the links that will take you from a common starting place to a listing of data sets that can be used for this analysis. A similar type of hunt should lead you to data sets that will match the final state searched for in your analysis.

When trying to find information online about D0, the place to start is:

http://www-d0.fnal.gov/atwork/index.html

This is the major branching point for most online D0 activities. Of course if you prefer, you can find your way to this site from the Fermilab homepage without much difficulty. Since I am interested in Z bosons, a good place to look is in the WZ physics group. Under the heading "physics", you will find a link labeled "physics groups". This link will take you to the physics groups page:

http://www-d0.fnal.gov/Run2Physics/physics_groups.html

In particular I am looking for Z bosons, so I am most interested in the WZ physics group. So I use that link to reach:

http://www-d0.fnal.gov/Run2Physics/wz/

Looking through the links on this site I come across a link that reads "WZ datasets (data)" under the heading "Miscellaneous". This link will take us to the WZ data-skimming page:

http://www-d0.fnal.gov/Run2Physics/wz/d0-private/wzskim/#d0recoVersion

In data skimming, the group pulls out events that pass certain simple criteria based on the end products of an event and stores them in a separate data file. For example, the WZ group will pull all events that have some minimal signature in the D0 muon system. Since I am looking for Z to $\mu^-\mu^+$, this is the data I am interested in. On the bottom of the WZ group data-skimming page there are three sections. In the section entitled "Summary by output stream." I see a stream named "mu". Using this link takes us to:

http://www-d0.fnal.gov/Run2Physics/wz/d0-private/wzskim/WZskim-mu.html

This page summarizes the datasets based on which reconstruction software was used. At this time the latest version is p13.05.00. Choosing the link entitled "wzskim1305.mu" takes us to:

http://www-d0.fnal.gov/Run2Physics/wz/d0-private/wzskim/WZskim-v00-0001
/p13.05.00/wzskim1305.mu.html

Scrolling a short way down this page displays a number of gray boxes that begin with "WZskim-muStream ...". The titles on these gray boxes are the names of the data files themselves. With these names I will be able to make a dataset definition that I can use on the SAM system, which in turn will enable us to download the data. Either print this page out or copy down a few of the dataset names. I began with two data files:

WZskim-muStream-20030112-200818.raw_p13.05.00
WZskim-muStream-20030110-145203.raw_p13.05.00

Each of these files contains in excess of 40,000 events, and all of these events contain activity in the muon detector. Further along in our example, I came to the conclusion that two data files weren't enough, and this is where I came to find more. With these file names I can now move on to the SAM dataset definition editor.

Creating a SAM Dataset Definition

As I did before, I will take you through the series of links that will take you to the site that allows you to create a dataset definition. Once again I start at:

http://www-d0.fnal.gov/atwork/index.html

This time, under the section titles "Computing", I see a link titled "SAM". This link takes us to:

http://d0db.fnal.gov/sam/

In the left hand column of this page I then find a link called "Dataset Definition Editor", which takes us to:

http://d0db.fnal.gov/sam_project_editor/

Here I select "Web Interface", which takes us to:

http://d0db.fnal.gov/sam_project_editor/DatasetDefDoc.html

This page has a link to the "Sam Dataset Definition Editor". This link takes us to:

http://d0db.fnal.gov/sam_project_editor/DatasetEditor.html

This is (finally) the Sam Dataset Definition Editor. I want to make a new dataset definition, so I select the link labeled "create new" from the list in the left column. Our next step is to identify the data I want. In this case, I know the names of the data files that I am looking for; so I will use the names themselves to establish a data definition. Underneath the section that begins with "Dimension Query", there are three lines with three data entry slots in each row. The first column is labeled "Operator", the second "Dimension", and the third "Constraint Value". In the first row, under the "Dimension" column select "FILE_NAME", indicating that I wish to specify which the data files I want by name. Under "Constraint Value" enter the name of one of the data files that you've selected. By using all three rows you can specify three different data files to add to your definition. If you want to use more than one row, make sure to change

the first column from "and" to "or"; this will add the data files with the first name, "or" the second name, "or" the third name.

Once this is completed, click on the button below labeled "Translate Constraints". The SAM system then looks through the available data, adding any data files that match your query, in this case any of the files that you named as constraint values. The box at the bottom of the page returns the results of your query. It includes the number of files that meet your definition, the number of events, the average size of the files, and a list of the file names. The list of file names should, of course, coincide with the file names you entered. If you wish to add more data files to your definition, just use the "or" operator and type in the new file names as constraint values as you did before.

Once the data definition is satisfactory, you will need to save the definition. To do so you must enter some additional information at the top of the page. First, enter a name of your choice for the data definition. Then enter your Fermilab username. Finally, enter "dzero" under work group. Then click on the save definition button and you are through.

For my example, I used the two data files listed earlier and saved them under the definition name "spurlock001". Don't include too many files in your definition because it will take quite a long time to download, and if there is an error in transmission you could waste a lot of time. Once you've finished the data definition, you are ready to download the files.

Downloading Data Files

To download data files I used two programs obtained from Mark Sosebee. A version of these programs can currently be found in the directory:

/home/spurlock/sam

Unfortunately, you most likely do not have permission to access this directory. In that case, get a software specialist to copy the entire "sam" directory into your home area.

The first thing to do is to modify the python script "get_file.py". To do so, use the command

gedit get_file.py

As you look at the program, you will notice a block of text where each line starts with "#project_definition = ...". The symbol "#" in this case is used to denote comments, so the only line that is actually executed is the line without the symbol "#". This is the last line of the block, which reads:

project_definition = "*dataset definition name*"

Inside the quotes on this line you should change the dataset definition name to the name you used to save your dataset definition.

A few lines lower is a section of program that reads:

```
# The maximum number of files to get from sam
max_file_amt =5
```

This limits the number of files it will read from SAM. This is used to protect against inadvertently downloading great numbers of data files. Change the number "5" if you are planning on downloading more than five data files at a time.

Finally, near the bottom of the code is a line that reads:

```
os.system('cp %s /cache001_B/phys5326/zmumu/. '%filename)
```

This line determines the file(s)'s destination. In this case I moved the file to:

```
/cach001_B/phys5326/zmumu/
```

Modify this line by inserting the appropriate destination for your files.

One these modifications are complete, save the python code. Now you are ready to download the files. It is important to note that you must make sure that SAM is installed on the machine that will perform this task. If you are unsure, consult with a software specialist. To download, run the "run_project" program by using the command:

```
. /run_project
```

It will take some time to download, but upon completion you should have the appropriate data file(s) in your destination directory. Now that I have our thumbnail files, our next job is to convert them to root files.

Converting from Thumbnail to Root

To do this conversion I used a program named "TMBAnalyze_x", which I copied from Fermilab. It is important that the preliminary commands discussed earlier have been run. By this I mean:

```
. /fnal/ups/etc/setups.sh
setup D0RunII p13.08.00
d0setwa
export DISPLAY=IP address number:0.
```

To begin, move to the directory where you wish the files to be installed. This should be the same directory in which you stored your thumbnail files. Once you are in the right directory, copy the files from Fermilab using:

```
cp /d0dist/dist/releases/p13.08.00/tmb_analyze/rcp/runTMBTreeMaker.rcp ./
```

Once TMBTreeMaker.rcp has been installed you can convert your file using:

```
TMBAnalyze_x –rcp runTMBTreeMaker.rcp –log tmb.log –input_file ./filename
```

Here *filename* refers to the name of the thumbnail file you wish to convert. This will generate a root file named "tmb_tree.root". It is necessary to rename this file before converting the next thumbnail, as it too will create a root file named "tmb_tree.root". Once your thumbnail files have been converted to root, you are ready to do some analysis.

Data Analysis

The best way to start data analysis is by using an existing program. This will allow you to become accustomed to the variables used, which will be needed when you begin to program on your own. It is also convenient to modify existing programs to suit your needs. A good program to start with is located in my home directory. Once again you may need the assistance of one of the HEP group's software specialists in order to gain access to the program. It is located at:

/home/spurlock/TMBTree_bu.C

Copy this file into one of your directories. In addition to this, you will need to copy another program from Fermilab called "MakeTMBTreeClasses_so.C". To copy this file use the following commands:

ls –s /d0dist/dist/releases/p13.08.00/tmb_analyze/macros/

This will show a list of the files available including "MakeTMBTreeClasses_so.C". Copy this file to your current directory. Since the version you will be using is most likely not p13, after typing "releases/p" hit the tab key and it will fill in the correct version.

cp /d0dist/dist/releases/p13.08.00/tmb_analyze/macros/MakeTMBTreeClasses_so.C ./

Now you are ready to run the analysis program. To do so you run "root". Simply type "root" at the command line and hit enter. This will start root, giving you a prompt inside root. In order to get to the command prompt you will first have to click on the window again; this an irritating feature of root. Once you are at the root prompt you can run your program with the following commands:

```
root> .x MakeTMBTreeClasses_so.C
root> .O 0
root> TFile *f = new TFile ("filename")
root> .L TMBTree_bu.C
root> TMBTree_bu h
root> h.Loop()
```

There are a couple of things that I should note. First, in the command ".O 0", the first oval is the capital letter while the second oval is the number zero. Second, *filename* should be replaced by the name of the root file you wish to look at; this includes the necessary directory information. In my case, this could be:

root> TFile *f = new TFile ("/cache001_B/phys5326/zmumu/tmb_tree.root")

I should also note that this only analyzes one of your root files. It is possible to chain files together or instruct your program to analyze all the files in a list, but that will not be covered here. Another important tidbit is the command ".q", which "quits" root. Sometimes root starts misbehaving, and it is useful to quit and restart. Also, using ".q" is a much better choice than using "Ctrl-c", which also can end your root session. After using "Ctrl-c" root continues to run in the background using up computer resources.

When the analysis program finishes running it will open a graphic window containing a histogram. To view other histograms, I will need to open a "browser", using:

TBrowser k

The browser window will show three directory files. One of these directories should be the current directory (i.e. the directory that you were in before you launched Root). Upon opening this directory, you should see the root file(s) that you created. Click on each root file. This creates corresponding files under the "root files" directory. Once you've created the appropriate files in the "root files" directory, you can move into that directory to start viewing histograms.

From the "root files" directory you can open the root file that you wish to examine. In that directory will be a sub directory of similar name. Upon opening that directory, you will find it filled with a number of directories with each shown by a leaf icon. Each of these directories corresponds to objects in the calorimeter, and inside these directories, you will find a list of histograms also shown with leaf icons. For example, the leaf label "Muon" contains data extracted from the d0 Muon system. The histograms inside show the values of the variables used in the root files. For example "Muon._E" holds a histogram that shows the energy of all of the muons in the event.

The description/meaning of each of the variable names listed should be documented by the particle ID working groups and accessible via their web pages. Unfortunately, this documentation can be difficult to find. As an example of this type of documentation, the variables used in the muon section can be found in a document entitled, "Content of the p13 Muon thumbnail"[4]. It is also helpful to get a slightly better understanding of the relevant detector components. In this case I found a document entitled, "D0 Muon Identification Primer".

Multiple Data Files

There will be times when you want to run your analysis code on a number of data files simultaneously. The simplest way to do this is by linking the files together into a chain[8]. This is done from inside root. To do so you define a chain, and then add all of the desired files to the chain. Then you pass the chain as an argument when calling your analysis program.

```
root> .x MakeTMBTreeClasses_so.C
root> .O 0
root> TChain tt ("TMBTree")
root> tt.Add("filename_1")
root> tt.Add("filename_2")
:
root> .L TMBTree_bu.C
```

root> TMBTree_bu h
root> h.Loop()

The "tt.Add" command can be used as many times as desired, each time adding a new file to the chain. This allows any number of files to be added.

Writing to Data Files

At times it will be convenient to record some values in a data file. This is done in the analysis program. In the begining of the program (in the "include" section) insert the following line:

#include <iostream>

It gives you the option of using input/output streams. Then early in the program (before you want to start writing data and before the main loop) insert this line:

ofstream *varname* ("*filename*");

This creates a data file named *filename.* The variable *varname* is then used to direct output to the datafile. For example,

varname << *variable_1* << " " << *variable_2* << "text" << endl;

This line will write out the values of *variable_1* and *variable_2* and some text into one line of the data file named "*filename*".

Writing Events into a New File

It is a little more complicated to create a new root file and store events that pass some criterion, but at some point it will be desirable to cut down the size of the data you are analyzing. Near the beginning of the program (before your main loop) include the following statements:

TFile **file_var_name* = new TFile ("*newfilename*.root", "recreate");
TTree **tree_var_name* = fChain->CloneTree(0);

The first creates a new (empty) root file to hold our tree. The second line creates a tree variable that will be filled in the main loop. The argument of CloneTree, namely "0", associates the new tree to the newly created file. Then inside the main loop, events can be written into the new file with the following statement:

if (*condition*) *tree_var_name*->Fill( ):

The condition can be anything you desire, derived from the data of the current event. If the event passes the condition it will then be written into the new tree. Finally, at the end of the code, include the line:

file_var_name->Write( );

NOTE: Originally, we had some problems with this. When used to copy a file in entirety, the new file was somewhat smaller than the original, and when we tried to make a copy of that copy our program crashed. The crashed was caused because some of the branches were not defined in the header file, specifically "LeBob", "MCevtInfo", and "BCJets". When the header was correctly modified to initialize these extra branches, the program no longer crashed. The resulting copy is still slightly smaller than the original.

Bibliography

[1] Mark Sosebee, private communications

[2] "WZ group – skimmed data documentation (datasets)", D0 WZ Physics Group, http://www-d0.fnal.gov/Run2Physics/wz/d0-private/wzskim-v00-0001/p13.05.00 /wzskim1305.mu.html

[3] "D0 Muon Identification Primer", D0 Muon Algorithm and Identification Group, http://www-d0.fnal.gov/phys_id/muon_id/d0_private/muonid_primer.html

[4] "Content of the p13 Muon Thumbnail", D0 Muon Algorithm and Identification Group, D0 Note 4091 Version 1.4, Feb 03, 2003

[5] Shahnoor Habib, private communications

[6] Venkatesh Kaushik, private communications

[7] Jaehoon Yu, private communications

[8] http://root.cern.ch/root/roottalk/roottalk03/0115.html