1 Computing Weights from ten Digitized Samples

1.1 Computational Method

1.2 Megan's Branch:

1.2.1 Where to Find Needed Files and Directories

This branch of the directory can be found at the following link: **Megan's Branch**

Details on all files contained in this directory can be found below.

1.2.2 Determining Weights for 10 digitized samples :Calculate.cpp

This is a file for extracting weights for reconstruction of pulse shape from 10 digitized data points.

The script reads a text file with 14 tab delineated data points and used the user specified points to calculate corresponding weights accordingly. Other files required to run this code are ComputeWeights.cpp and ComputeWeights.h which can all be found in the following directory: **Compute Weights**

In addition to those files the machine used will also need the CLHEP directory installed. If the CLHEP directory is needed, use the following link for documentation and installation help: **CLHEP Documentation**

Within the file 'calculate.cpp' the user is able to define the data file they wish to use. This can be done by the user being prompted for the file name inside the terminal or by hardcoding the path of the data file into the script. Data files are currently located in the following directory: **Data Files**.
Original data can also be found at: **Data Files on lxplus** but access to lxplus is required.

1.2.3 Running the File in Terminal

To run these files in terminal run the following command:

```
clang++ -std=c++11 -stdlib=libc++ -L/Users/meganstark/Computation\*/
PHYS7321_Project/sourcecode/bin/lib/ -I/Users/meganstark/Computation
\*/PHYS7321_Project/sourcecode/bin/ -lCLHEP-Matrix-2.3.4.5
Calculate.cpp ComputeWeights.cpp
```

Below is an example:

```
clang++ -std=c++11 -stdlib=libc++ -L/Users/meganstark/Computation\*/
PHYS7321_Project/sourcecode/bin/lib/ -I/Users/meganstark/Computation
\*/PHYS7321_Project/sourcecode/bin/ -lCLHEP-Matrix-2.3.4.5
Calculate.cpp ComputeWeights.cpp
```

This command will create an output file called 'a.out' in the current directory. To run this file use the following command in the terminal:

```
./a.out
```

then run this file and saves output file that includes weights also depending on the set verbosity will output the variables noted above in the terminal. The output file is currently saving files to the following directory:
and is of the following format:

1.2.4 TimeWeights.py

This script plots the output data of Calculate.cpp.

1.2.5 CalculateWeights.py

This file is a less robust version of the same calculation in python. It is convenient however, since a single script is able to run make the calculations as well as plot the results.

2 Computing Weights From Alpha and Beta Parameters

2.1 Determining Ten Samples from the Pulse

2.1.1 Determining Ten Samples in Root Terminal

User will need to input the following variables: alpha, beta, t0, amplitude
Values for these variables can be found below.
$/Users/meganstark/Computation*/PHYS7321_Project/Data/CrystalParams.txt$
This file containes these four variables for each of the $\approx 75k$ crystals in the ECAL.

The format of the file is shown below:

```
Crystal ID#  Amplitude T0     Alpha    Beta
838861313    0.240791  121.44 1.18942  40.1906
```

User will also need to input the number if iterations i which is the number of time shifted samples that will be determined. Each time the code iterates over i, the sample is taken from original sample location + 1 x-unit. For example, the 10th iteration will be taken from original sample location + 10.

To use the Fit Alpha Beta function in order to determine the 10 samples for a given alpha beta value can be done using the following commands in the terminal. In order to do this you must have ROOT installed on your computer.

If ROOT has not been installed documentation on ROOT can be found here: **ROOT Documentation**.
I used HomeBrew to install ROOT on my Mac OS. Homebrew can be found here: **HomeBrew**.

To open ROOT use the following command in the terminal:

```
root -l  //open root
```

Then the function needs to be defined, here TF1 is a predefined data type in ROOT. When defining the function the final two imput variables must be changed. These are the minimum and maximum variables on the x axis.

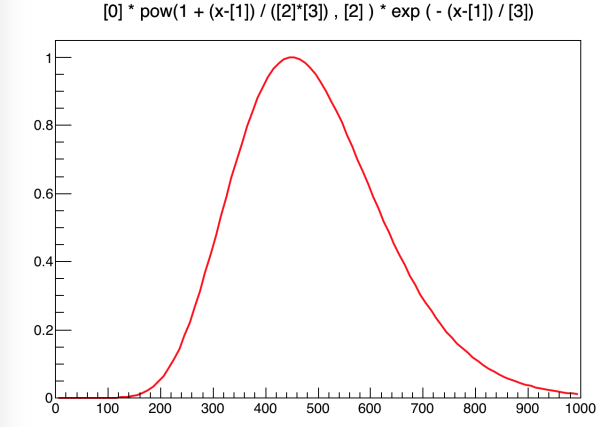
```
TF1 *function_alphabeta = new TF1("function_alphabeta","[0] *
pow(1 + (x-[1]) / ([2]*[3]) , [2] ) * exp ( - (x-[1]) / [3])",73,40*10.5);
// definition of the alpha beta function:
// TF1("function name" , function definition, xmin, xmax);
// Here, [0], [1], [2], and [3] are the needed parameters;
```

Next, the user need to set the parameters as mentioned above. These parameters should be from a single crystal (single line) from crystalparams.txt and set the following parameters with the following code, where within the parentheses is the parameter number, and then the parameter value. Parameter 0 is the amplitude, parameter 1 is t_0, parameter 2 is Alpha, and parameter 3 is Beta.

```
function_alphabeta->SetParameter (0, 0.240791);    // A
function_alphabeta->SetParameter (1, 121.44);   // t_0
function_alphabeta->SetParameter (2, 1.18942);   // alpha
function_alphabeta->SetParameter (3, 40.1906);   // beta
```

In order to see the pulse one should use the following command:

```
function_alphabeta->Draw()
```



If one is not able to see the pulse in its entirety (see image before for example) The one must change the range on the x axis. To do this you should use the following command:

```
function_alphabeta->SetRange(71, 400) // setting range of x axis in the
// entire pulse is not able to be seen
function_alphabeta->Draw() // Make sure to draw the function again
// to know for sure you have the proper xmin and x max
```

You will then want to redraw the pulse and continue to follow this process until the entire pulse can be seen in the plot.

Next we will want to create an output file for the 10 samples being extracted will be saved to. This is done using the following command:

```
  std::chrono::system_clock::time_point tp = std::chrono::system_clock::now();
  std::chrono::system_clock::duration dtn = tp.time_since_epoch();
  std::stringstream ss;
  ss << "output/" << "new17" << ".txt";
  std::ofstream output_file(ss.str());
  // creating output folder for the sample results to be saved to

std::vector<double> d;
```

// determining samples for i iterations //each iteration the sample is taken from the ith unit to the left. This is for the calculation of time shifting .

```
for(int i = 1;i<20;i++){
    for(int j = 0;j<90*9+i;){
        double tmp = function_alphabeta->Eval(i + j);
        d.push_back(tmp);
        j = j + 90;
    }
}
```

//saving to tab delineated output file with name given above

```
for(int k = 0; k<d.size(); k++){
    int lineLength = k % 10;
    if(lineLength < 9){
```

```
        output_file << d[k] << "\t";
    } else {
        output_file << d[k] << "\n";
    }
}
```

From these commands a single file will be created with 10 samples saved for each iteration over i (moving one x units to the left).
This file will be saved to the following directory: $/Users/meganstark/Computation*/PHYS7321_Project/output/$
The file will be given the name titled above, here it is"$new17.txt$" User should change the file name.

The saved file will have the following format:

```
for i = 0:  sample1 sample 2 sample 3...
```

Each line in the file should contain 10 tab delineated values, there are the10 samples from that pulse. This file can then be used as an input for the method mentioned above in order to determine the weights for the 10 samples. See section 2.2 for more details.

2.1.2 Script to Determine 10 Samples: Untitled.cpp

$Untitled.cpp$ reads a text file with tab delineated crystal parameters, and uses the user specified points to calculate corresponding samples of the created pulse. Other files required to run this code are:
which can all be found in the following directory: **Compute Weights**

User defined input variables for the script: $xmin$, $xmax$, number of iterations i , and the file CrystalParams.txt located in the following directory:

```
/Users/meganstark/Computation*/PHYS7321_Project/Data/CrystalParams.txt
```

format of file:

```
Crystal ID#  Amplitude T0       Alpha    Beta
838861313    0.240791  121.44   1.18942  40.1906
```

To build this file use the following command:

```
clang++ -std=c++11 -stdlib=libc++ -v `root-config --cflags --glibs --libs`
-L/Users/meganstark/Computation\*/PHYS7321_Project/sourcecode/bin/lib/ -I/Users
/meganstark/Computation\*/PHYS7321_Project/sourcecode/bin/root/hist/hist/inc/
-I/Users/meganstark/Computation\*/PHYS7321_Project/sourcecode/bin/root/include/
-I/Users/meganstark/Computation\*/PHYS7321_Project/sourcecode/bin/root
-lCLHEP-Matrix-2.3.4.5 /Users/meganstark/Computation\*/PHYS7321_Project/src/
ComputeWeights/untitled.cpp -o /Users/meganstark/Computation\*/PHYS7321_Project/a.out
```

The executable file will be located in:

```
PHYS7321_Project/
```

and will be named

```
a.out
```

The following command can then be used to execute the file:

```
./a.out
```

Executing this file will will create an output file in the following directory: $Users/meganstark/Computation*/PHYS7321_Project/output/$
The saved file will have the following format:

```
for i = 0:  sample1 sample 2 sample 3...
```

Each line in the file should contain 10 tab delineated values, there are the10 samples from that pulse. This file can then be used as an input for the method mentioned above in order to determine the weights for the 10 samples. See section 2.2 for more details.

2.2 Determining Weights From Alpha Beta Parameters: $CalculateAlphaBeta.cpp$

$CalculateAlphaBeta.cpp$ reads a text file with 10 tab delineated data points and uses the user specified points to calculate corresponding weights accordingly. Other files required to run this code are ComputeWeights.cpp and ComputeWeights.h which can all be found in the following directory: **Compute Weights**
input: file of data samples determined using Untitled.cpp or the Code noted above for the root terminal
running: following code to build:

```
clang++ -std=c++11 -stdlib=libc++ -L/Users/meganstark/Computation*/PHYS7321_Project/
sourcecode/bin/lib/ -I/Users/meganstark/Computation*/PHYS7321_Project/sourcecode/bin/
-lCLHEP-Matrix-2.3.4.5 /Users/meganstark/Computation*/PHYS7321_Project/src/
ComputeWeights/Calculate_AlphaBeta.cpp /Users/meganstark/Computation*
/PHYS7321_Project/src/ComputeWeights/ComputeWeights.cpp -o /Users/meganstark/
Computation*/PHYS7321_Project/a.out
```

will give an executable a.out in following directory:
this output will have the following form:
Execute this file using following command: ./a.out
In turn, this will give you an output file of the following name with the following format

2.3 Plotting Results: $PlotAlphaBetaWeights.py$

Input variables:
User will need to input the file with weights, located on line 27 :

```
with open('new18_weights.txt', 'r') as f:
```

Here it is opening

```
  new18_{}weights.txt
```

This file will plot the samples, weights, and amplitude of signal located in the input file.

3 References

- **Reconstruction of the signal amplitude of the CMS electromagnetic calorimeter**