Fiber Bragg Grating
Fiber Optic Sensors Measuring Strain & Temperature Changes at the CMS's Central Beam Pipe

Khalil El Achi

CERN Summer Project
CMS Safety Team

Under the supervision of
Dr. Francesco Fienga
Dr. Vincenzo Romano Marrazzo
Dr. Zoltan Szillasi
Dr. Noemi Beni

Meyrin, Switzerland

Table of Content

Abstract

This work tests the reliability and efficacy of the Fiber Bragg Grating (FBG) monitoring system installed around the Compact Muon Selonoid's (CMS) beam pipe. The FBGs work on measuring Temperature and Strain around the peripherals of the CMS's BP. The CMS is one of four detectors at the Large Hadron Collider (LHC) operated by CERN around the borders of France and Switzerland. A large particle accelerator that accelerates protons and heavy ions relativistically to high speeds close to the speed of light. Then collides two beams at 4 different detects laid out around the circular collider. ATLAS, LHCb, ALICE, and CMS. The CMS safety team works on systems inside the CMS BP to monitor temperature and strain changes induced in the beam pipe. In hindsight, the FBG sensors use Fiber Optic Sensor (FOS) technology with varying Bragg Gratings inside the glass core of the optical fiber. FBGs work by being exposed to UV light as the result of proton collisions. Then the light is thus reflected back at each grating. The reflected ray's wavelength changes when being subjected to different levels of temperature and strain. To test the efficacy of the interrogation system, temperature calibration is done and linear and cubic fittings were generated to estimate the temperature through the imputed current. A MATLAB code has been constructed to output the outlined calibration graphs and the fittings.
Keywords: Fiber Bragg Grating, Fiber Optic Sensor, CMS, LHC

Part I

Introduction to Fiber Bragg Grating (FBGs)

1 Utilization of FBGs

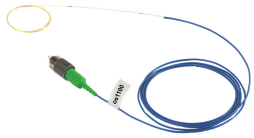

Figure 1: Fiber Optic Sensor with Bragg Gratings

To fabricate a monitoring system for the inner working of the CMS BP, seen in figure 2, the system must endure a harsh and arduous atmosphere. During the LHC run, the pipe will be subjected to extreme conditions of vacuum levels as low as $10^{-13}atm$ and high levels of radiation due to the constant proton-proton collisions, happening once every 25 milliseconds. These conditions render regular electric sensors failable and unfeasible to engender a monitoring system.

Luckily Fiber Optic Sensors (FOS), seen in figure 1, with Bragg Gratings are able to withstand such conditions and thus can be used to construct a lucrative and sufficient monitoring system.

1.1 Working Principle

Hundreds of FBG sensors have already been installed around the beam pipe and are functioning well by recording data acquisition for all three runs.

The basic notion relies on periodically modulating and varying the refractive index in the core element of the optical sensor. When UV light is shined at one of the ends of the sensor, each grating reflects back a certain wavelength from the initial light beam. Thus transmitting forward the remaining light beam. Under nominal and ambient conditions this wavelength is constant and can be referred to as

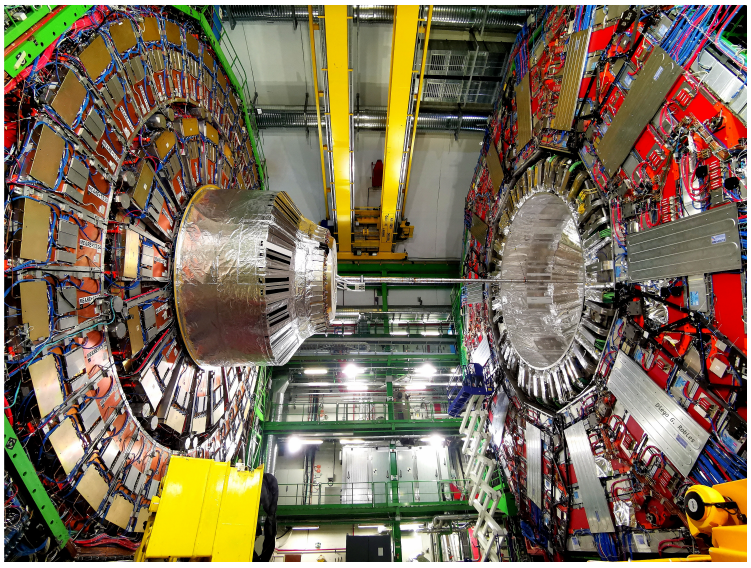

Figure 2: CMS's Beam Pipe

the Bragg wavelength λ_B. This principle is illustrated in figure 3 and relies on Wavelength Division Multiplexing.
when it is subjected to temperature and strain this wavelength is altered and shifts from the regular Bragg wavelength. The change and modulation can be used in mathematical relations that can thus predict the temperature around the FBG and the subsequent strain that was applied to it.

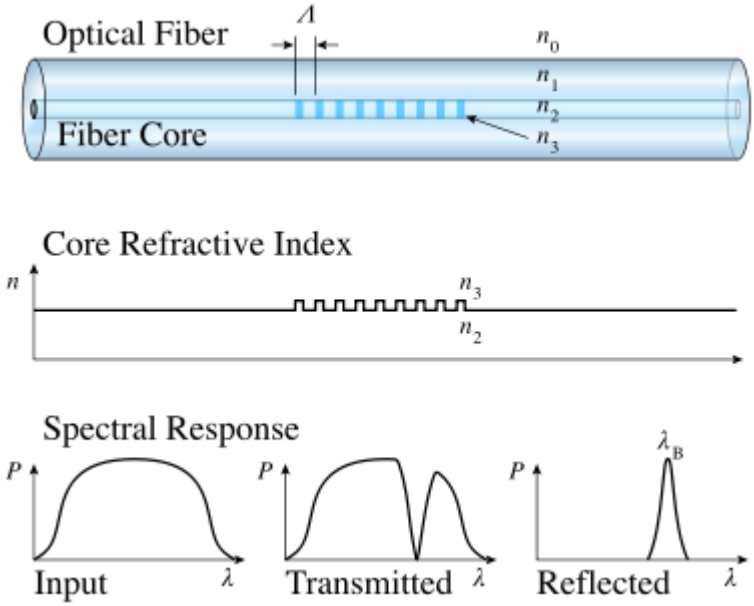

Figure 3: Working Principle of FBGs

1.2 Mathematical Model

A mathematical relation can be laid out to represent the ability of the FBG system to predict temperature and strain.
Initially the construct Bragg wavelength in ambient conditions can be represented as

$$\lambda_B = 2n_{effective}\Lambda \tag{1.1}$$

Where n_{eff} = is the effective refractive index and Λ is the grating pitch of the corresponding FBG.
Furthermore, using the difference in the omitted wavelength $\delta\lambda$ a relation can be constructed as such:

$$\frac{\Delta\lambda}{\lambda_B} = \underbrace{(1-P_e)\epsilon}_{\text{strain measure}} + \underbrace{[(1-P_e)\alpha+\xi]\Delta T}_{\text{temperature measure}} \tag{1.2}$$

Where P_e is the photoelastic constant, α is the coefficient of thermal expandability of metal and ξ is the coefficient of thermo-optics.
The first section of the equation can be used for strain measurement whilst the second part can be used to measure the alterations of temperature from the initial temperature.

Part II

Implementation on Beam Pipe

1.3 CMS's Beam Pipe

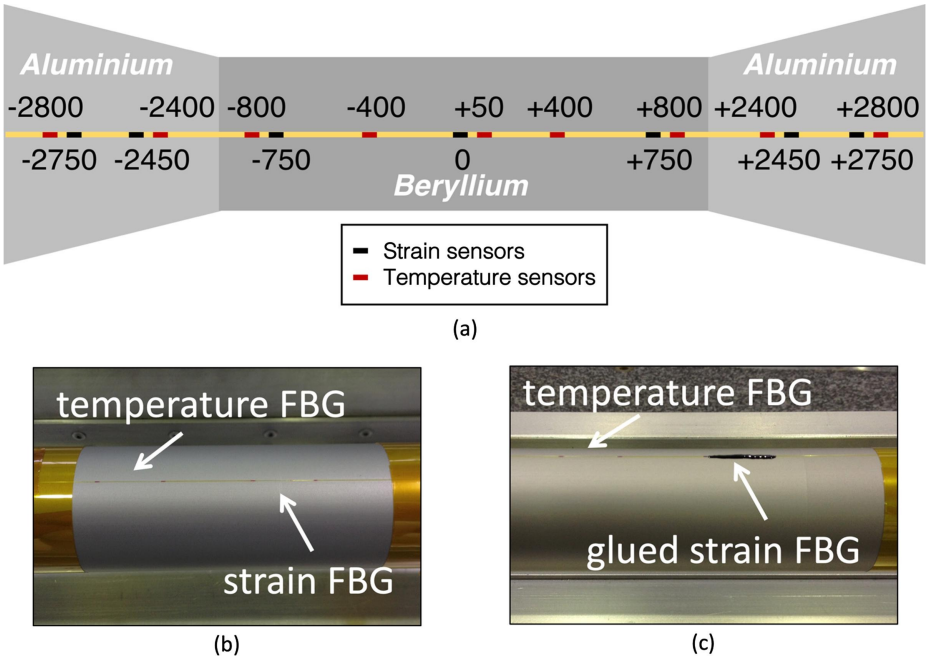

Figure 4: CMS's Beam Pipe Illustration

The localized area where the proton-proton collisions occur and where the two diverging beams coincide is in the CMS's central Beam Pipe, illustrated in figure 4. A 6-meter long tube, with a central 3-meter long gossamer Beryllium tube that has a diameter of 4.5 cm and is 0.8 mm thick. With 1.5-meter long Conical Aluminum end caps attached to either side of the tube. The BP must be engineered in a way to withstand the harsh condition subjected to the LHC from:

- Extreme Vaccum of $10^{-13} atm$
- Temperatures up to $220°C$ during bakeout
- High doses of radiation close to $10^5 - 10^6 Gy$ resulting from the proton-proton collision

- Large Magnetic field up to 3.8 T

It is crucial that the material has a low atomic number and doesn't become radioactive as it is exposed to high radiation levels. Furthermore, since the data and the particles resulting from the collisions are vital, any system and equipment placed inside the triggers and calorimeters must not interfere with the data and the results from these collisions.
Thus FBGs are a great recourse to fabricate a Temperature and Strain monitoring system due to their lightweight, radiation, and vacuum immunity.

1.4 Assembly on BP

The FBGs are glued around the beam pipe in several z-locations and then coupled with Aluminum foil around them to be kept in place. Figure 5 shows the assembly of the FBGs on a section of the BP.
It is vital that the glue can withstand the high levels of radiation of the cavern as well as the high temperature of 220°C during the bake-out.
Tests around the reliability of the glue have been done separately and epoxy encapsulate Stycast 2850 FT with catalyzer Catalyst 24 LV has shown to be an adequate fit in the final assembly on the beam pipe.

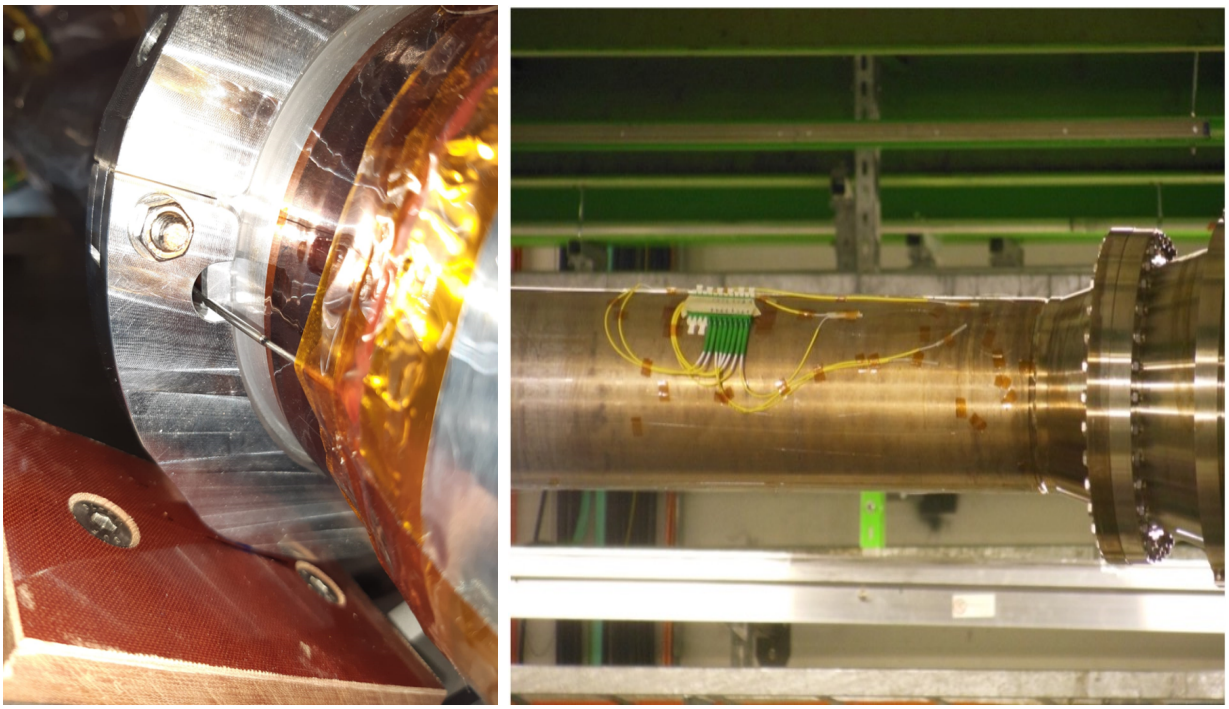

Figure 5: FBGs intalled on the BP

Part III

Calibration Test & Numerical Models

2 Interrogation System

The interrogation system is a prototype that works in testing and calibrating different Fiber Bragg Grating sensors using an arrayed waveguide (AWG) using channels 26 and 28 under the following setup in figure 6.
This prototype will be implemented in a Detector Safety System environment in the general CMS cavern as a Safety monitoring system. This system has been collecting data since January 2022 at the CERN site in Meyrin.
The setup builts on a temperature-controlled box with water in a chamber. The system can be calibrated in different ambient temperatures and has FBG sensors installed on the chamber.

- Initially A broadband source of light is fed into an optical circulator. Which acts as a monochromator and injects the beam into the FBGs.

- The reflected light is this fed into the AWG channels and a photodetector outputs an electrical current.

- A measurement.db file is appended into an SQLite database fosdsstest.sqlite with the following parameters:

    1. UnixTime of each measure
    2. The intended temperature the controlled box is set at
    3. The actual temperature of the chamber
    4. Current Outputted from FBG 26 channel
    5. Current Outputted from FBG 28 channel

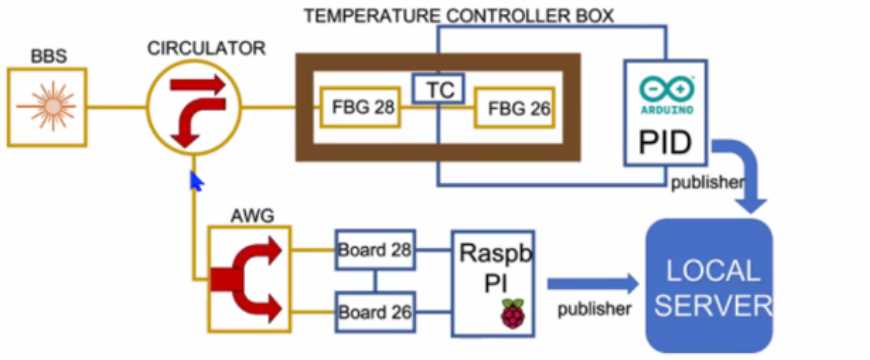

Figure 6: The Illustration of the Interrogation System

3 Temperature Calibration

This work endeavors to construct a relation, a numerical equation, between the Temperature and outputted current. Thus establishing a relationship that is capacitated to accurately predict the temperature using the current from the photodetector.
Many calibration tests were performed, the calibration test performed goes as follows:

a. Starting from 20 ℃, the temperature was increased by 1 ℃ every 30 mins.

b. The system overloads at 60 ℃ and goes back to 20 ℃ twice.

Calibration tests were previously made having 5 or 2 degrees variation but with the same wait time of 30 minutes so the system can calibrate well.

3.1 Connection Via Sqlite

The Calibration is first done by connecting to the pi@cmsraspifosdsstest.cern.ch server.
Initially the system can be closely monitored using the /home/pi/serialtest/sertialtest_thermo/attachTMux which shows a window of the current measurements and what is being appended into measurement.db.
Using the mcedit command, the runCalibration script can be edited. It first marks the waitTime which is the time separating the temperature change intervals.
A vector with the temperatures that must be set to is manually typed. Thus the vector is laid out as such $20\ 21 \cdots 59\ 60\ 59 \cdots 21\ 20\ 21 \cdots 59\ 60\ 59 \cdots 21\ 20$.
A for loop is then used to call the command /home/pi/DSSTest/set Temperature for each value in the vector after the specified waitTime.
When the calibration is ready to be executed the command /home/pi/DSSTest/runCalibration is called. Then the script selectDataintoCSV is edited with the updated UnixTime

which is the time when the runCalibration was executed.
selectDataintoCSV when called will collect the data points from section 2 and save them in a CSV file that is saved using this command Sqlite3 /home/pi/DSSTest/fosdsstest.sqlite </home/pi/DSSTest/selectDataintoCSV> filename.csv
Finally after the calibration is complete, the file can be thus downloaded using the command scp pi@cmsraspifosdsstest: /home/pi/filename/csv .

3.2 MATLAB Code

A coordinated MATLAB function was typed up to do the entire analysis and outputs all the material needed. The code can be seen here:

```
function [Time,current_Temperature, ...
    I_26,I_28,f26_1,f26_2,f28_1,f28_2]=OutputFromCSV(filename . . .
    ,lower_bound_26,upper_bound_26,lower_bound_28,upper_bound_28)
if nargin<2 || isempty(lower_bound_26) || isempty(upper_bound_26) ...
    || isempty(lower_bound_28) || isempty(upper_bound_28)
    lower_bound_26=0;
    upper_bound_26=70;
    lower_bound_28=0;
    upper_bound_28=70;
end
Table=readtable(filename); % converts the CSV inputs into a table
M=table2array(Table); % converts the table into an array

% Each column is deligated in an array

Unix_Time=M(:,1);
set_Temperature=M(:,2);
current_Temperature=M(:,3);

I_26=M(:,4);
I_28=M(:,5);

% Converting the Unix time to minutes

Date_Array=datestr(Unix_Time./86400 + datenum(1970,1,1));
l=length(Date_Array(:,1));
minute=[];
for i=1:l
    x=split(Date_Array(i,:));
    dt=datevec(datetime(x(2),'InputFormat','HH:mm:ss'));
```

```
        minute(i)=minutes(duration(dt(:,4:end)));
    end
    Time_interval = minute ./60;
    Time=0;
    for i=2:l
        if Time_interval(i)>Time_interval(i-1)
            timestep=Time_interval(i)-Time_interval(i-1);
        else
            timestep=(Time_interval(i)+24)-Time_interval(i-1);
        end
        Time(i)=Time(i-1)+timestep;
    end

    % Finding the average of the plateau

    temp_variation=[];
    j=1;
    for i=1:l-1
        if set_Temperature(i)≠set_Temperature(i+1)
            temp_variation(j+1)=set_Temperature(i);
            j=j+1;
        end
    end
    temp_variation(end+1)=20;
    averaged_26=[];
    averaged_28=[];
    for i=2:length(temp_variation)
        current_26=[];
        current_28=[];
        for j=1:l
            if set_Temperature(j)==temp_variation(i)
                current_26=[current_26;I_26(j)];
                current_28=[current_28;I_28(j)];
            end
        end
        plateau_26=[];
        plateau_28=[];
        for k=1:length(current_26) -1
            es_26=abs((current_26(k+1) - current_26(k))/ ...
               current_26(k+1)) *100;
            es_28=abs((current_28(k+1) - current_28(k))/ ...
               current_28(k+1)) *100;
            if es_26<1
                plateau_26=[plateau_26;current_26(k);current_26(k+1)];
```

```
71              elseif es_26≥1
72                  plateau_26=[];
73              end
74              if es_28<1
75                  plateau_28=[plateau_28;current_28(k);current_28(k+1)];
76              elseif es_28≥1
77                  plateau_28=[];
78              end
79          end
80          if isempty(plateau_26)
81              plateau_26(1)=current_26(end);
82          end
83          if isempty(plateau_28)
84              plateau_28(1)=current_28(end);
85          end
86          averaged_26=[averaged_26 mean(plateau_26)];
87          averaged_28=[averaged_28 mean(plateau_28)];
88      end
89
90      temp_variation=temp_variation(2:end);
91
92      % specifying the fit for FBG 26
93
94      exclude1= ...
            excludedata(averaged_26,temp_variation,'range',[lower_bound_26 ...
            upper_bound_26]);
95      f26_1=fit(averaged_26',temp_variation','poly1','Exclude',exclude1);
96      f26_2=fit(averaged_26',temp_variation','poly3');
97
98      % specifying the fit for FBG 28
99
100     exclude2= ...
            excludedata(averaged_28,temp_variation,'range',[[lower_bound_28 ...
            upper_bound_28]]);
101     f28_1=fit(averaged_28',temp_variation','poly1','Exclude',exclude2);
102     f28_2=fit(averaged_28',temp_variation','poly3');
103
104     plot(Time,current_Temperature','LineWidth',2)
105     xlabel('Time (h)')
106     ylabel('Temperature (C)')
107     grid on
108     figure;
109     plot(Time,I_26','LineWidth',2)
110     xlabel('Time (h)')
```

```
    ylabel('I FBG 26 (mA)')
    grid on
    figure;
    plot(Time,I_28','LineWidth',2)
    xlabel('Time (h)')
    ylabel('I FBG 28 (mA)')
    grid on
    figure;
    plot(I_26,current_Temperature)
    xlabel('I FBG 26 (mA)')
    ylabel('Temperature (C)')
    grid on
    figure;
    plot(I_28,current_Temperature)
    xlabel('I FBG 28 (mA)')
    ylabel('Temperature (C)')
    grid on
    figure;

    plot(f26_1,averaged_26,temp_variation,exclude1)
    xlabel('Averaged I FBG 26 (mA)')
    ylabel('Temperature (C)')
    title('Linear Fit with exclusion')
    grid on
    figure;
    plot(f28_1,averaged_28,temp_variation,exclude2)
    xlabel('Averaged I FBG 28 (mA)')
    ylabel('Temperature (C)')
    title('Linear Fit with exclusion')
    grid on
    figure;
    plot(f26_2,averaged_26,temp_variation)
    xlabel('Averaged I FBG 26 (mA)')
    ylabel('Temperature (C)')
    title('Cubic Fit for the entire curve')
    grid on
    figure;
    plot(f28_2,averaged_28,temp_variation)
    xlabel('Averaged I FBG 28 (mA)')
    ylabel('Temperature (C)')
    title('Cubic Fit for the entire curve')
    grid on

    end
```

3.2.1 Code Explained

Initially, the function takes the CSV file as input and adds the entire data in one Matrix. It then separates the matrix into smaller vectors, each for the data sets listed in section 2. The vector containing the Unix time is then transformed to a regular hourly vector starting at the Zeroth point at the beginning of the data measure.

3.2.2 Outputed Graphs

The code outputs the first three curves showing the variation of the Temperature, current outputted from the 26^{th} channel, and the current outputted from the 28^{th} channel.

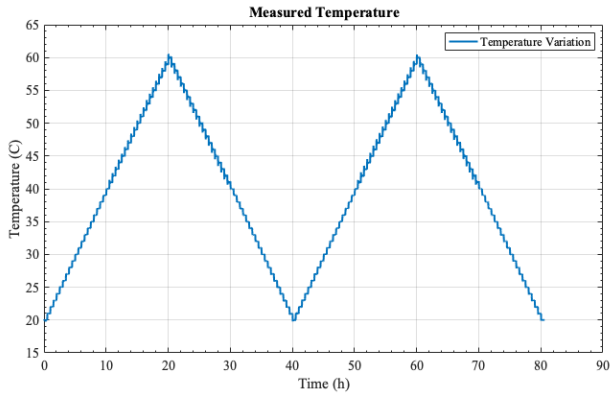

Figure 7: Temperature Measured across Time

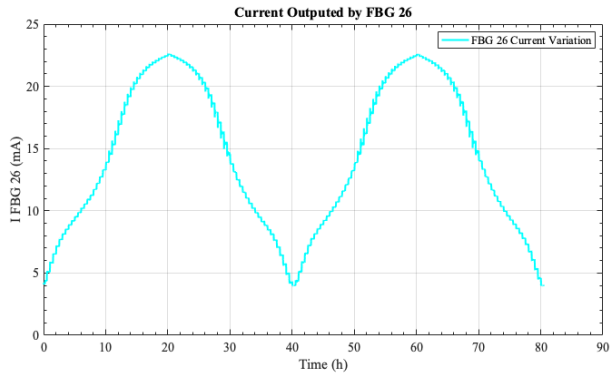

Figure 8: Current Outputted from the 26^{th} Channel across Time

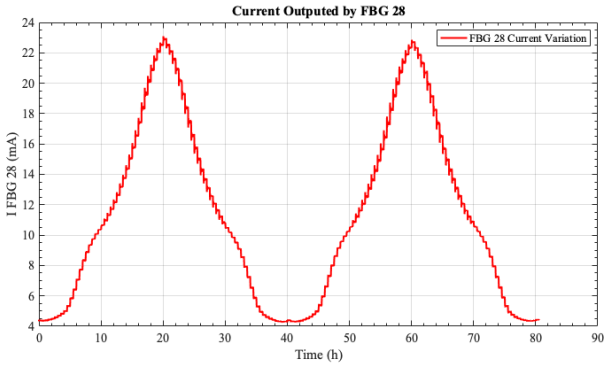

Figure 9: Current Outputted from the 28^{th} Channel across Time

As shown in figure 10 during the initial stage of the calibration the temperature and the current increase progressively and to the desired setting.

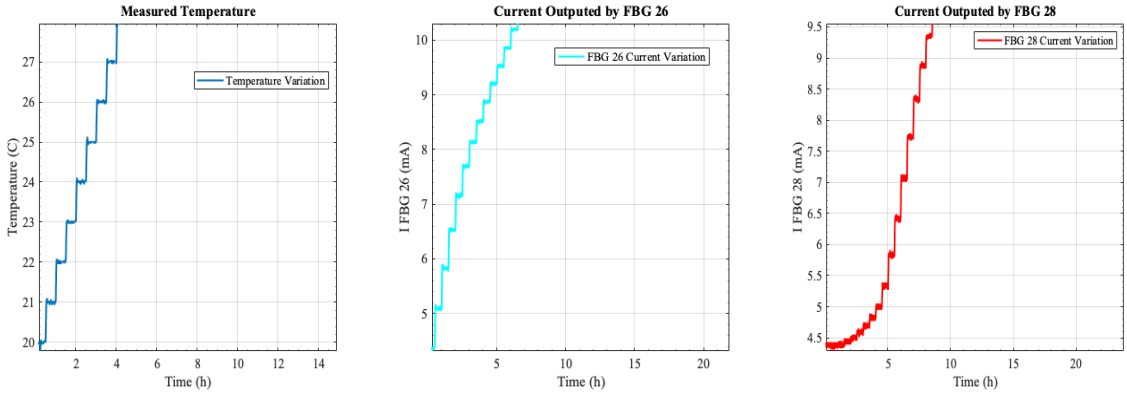

Figure 10: A zoomed section of the graphs

Whilst figure 11 shows that later in the calibration period when the temperature goes beyond $40°C$ the system tends to be overwhelmed and overshoot is seen. Where the system overestimates the intended temperature to be set. This can be seen in both the measured temperature and the outputted current.

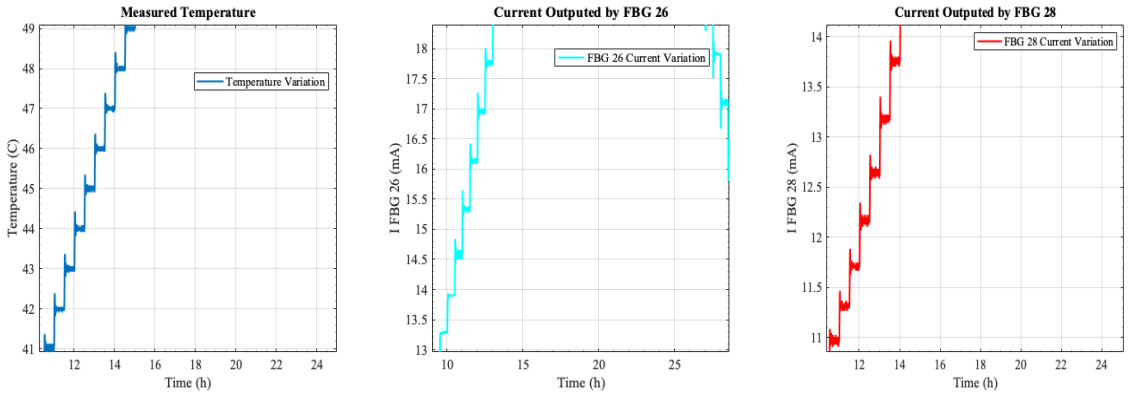

Figure 11: A zoomed section of the graphs foreshadowing the overshoot in Temperature and Current

This doesn't necessarily present a significant error since the system will still regulate over time and will fix the perspective overshoots and undershoots. But when we intend to pilot the variation of temperature in terms of the output current for channels 26 and 28, plotted in figure 12 the graphs look chaotic and messy. In adequate to construct linear or cubic fits on the curves. The chaoticness is represented in figure 13.

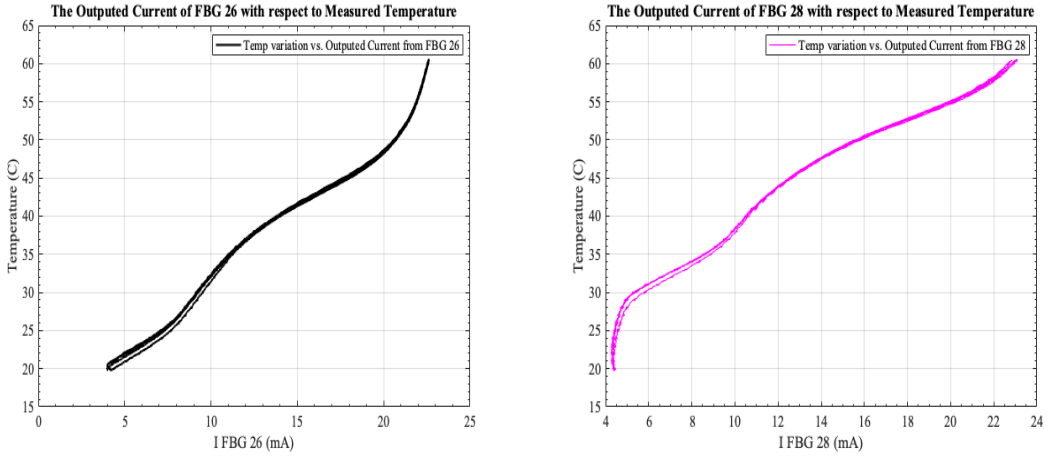

Figure 12: The Variation of Measured Temperature in terms of the Output Current

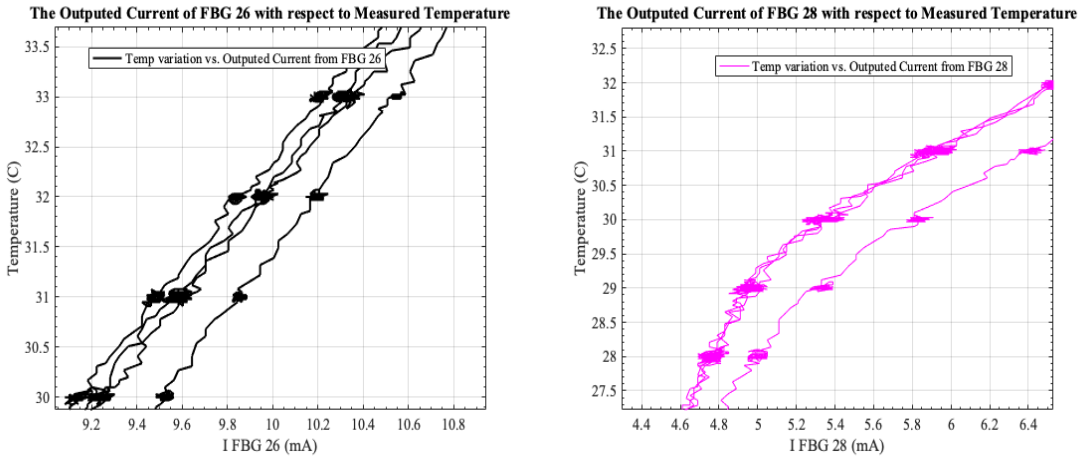

Figure 13: A zoomed in portion of the plotted graphs above illustrating their chaoticness

To fix the overshoot dilemma and make the data more elegant and discretized, the MATLAB code implements a plateau scanning technique. For each temperature set, all the outputted current is put in one vector, and the percentage difference between them is thus calculated. The percentage difference chosen to represent a plateau formation is 1%. When less than 1% separates the data points, the code saves it in a new vector. If suddenly the percentage difference goes above 1%, the vector is emptied, and the measure starts again. After all the plateau is scanned, all the values are averaged for each cycle. Then each temperature set has a discrete current assigned to it. Thus the graphs are transformed into more amenable and fittable graphs, seen in figure 14.

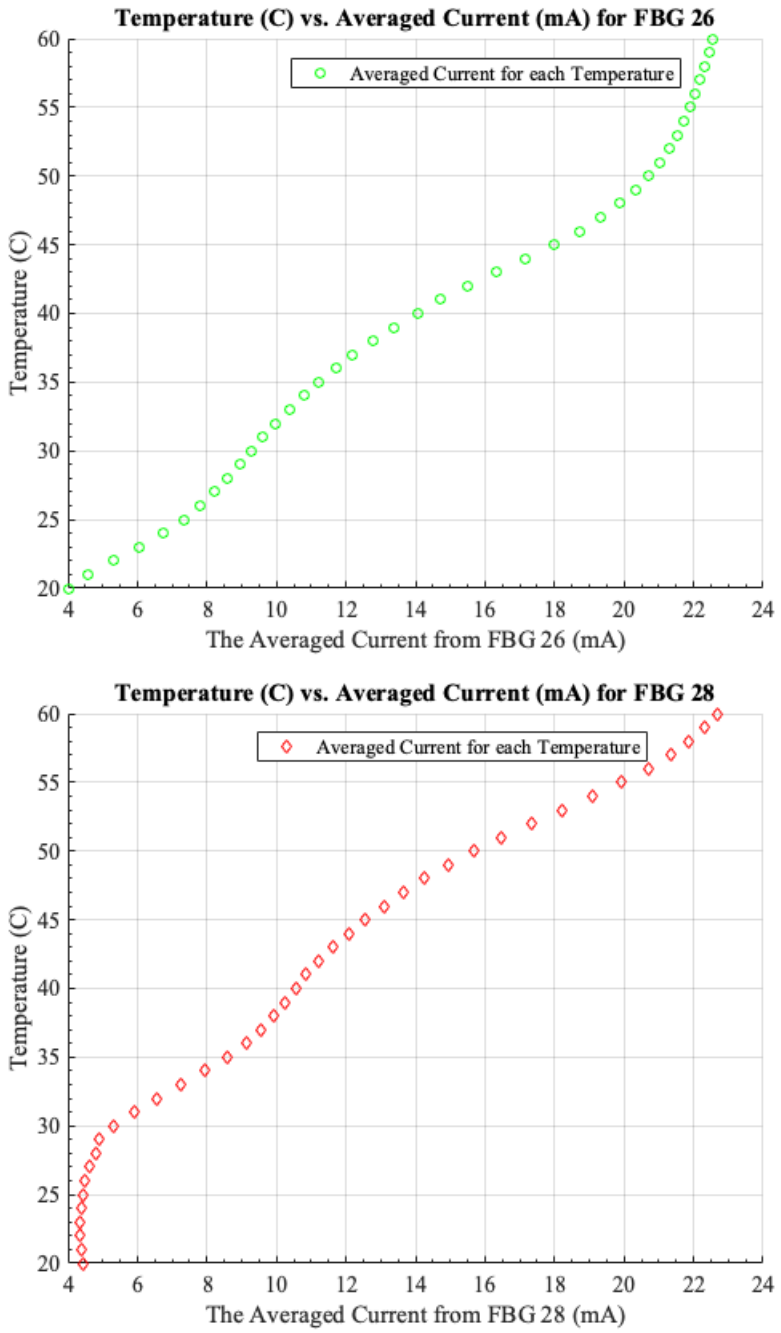

Figure 14: A Discretized Solution to the Temperature vs. Current Graphs

3.2.3 Constructing Linear and Cubic Fittings

After the graphs are discretized, linear and cubic fits can be constructed. Choosing sections from either graph exponential and logarithmic trends appears to exclude making a simple and elegant linear fit for the temperatures that lie on that trend. For the current outputted from the 26^{th} channel the data points above the range of 50 are excluded. Respectively,

the data points below the range of 30 are excluded in the current outputted from the 28^{th} channel.

The linear fits for each channel are illustrated in figure 15 and are as such:

$$T = 1.821\ I_{26} + 14.06 \tag{3.1}$$

$$T = 1.723\ I_{28} + 22.41 \tag{3.2}$$

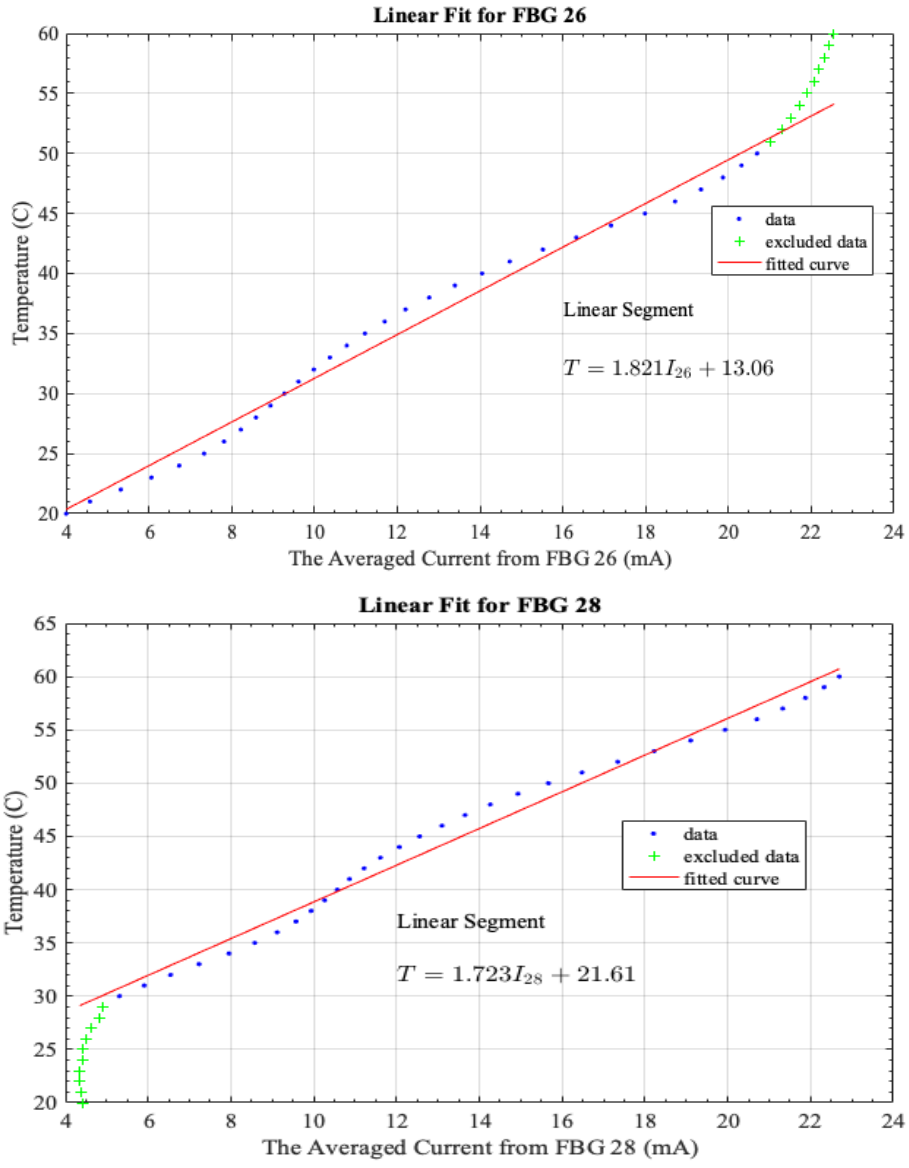

Figure 15: Linear Fits For Each Graph

Conversely, we can fit the entire data set in a cubic fit that encompasses the entire data

points. The relations are laid out as such:

$$T = 0.006268\ I_{26}^3 - 0.2503\ I_{26}^2 + 4.916\ I_{26} + 2.896 \tag{3.3}$$

$$T = 0.002338\ I_{28}^3 - 0.1476\ I_{28}^2 + 4.411\ I_{28} + 7.673 \tag{3.4}$$

The fits are illustrated in figure 16.

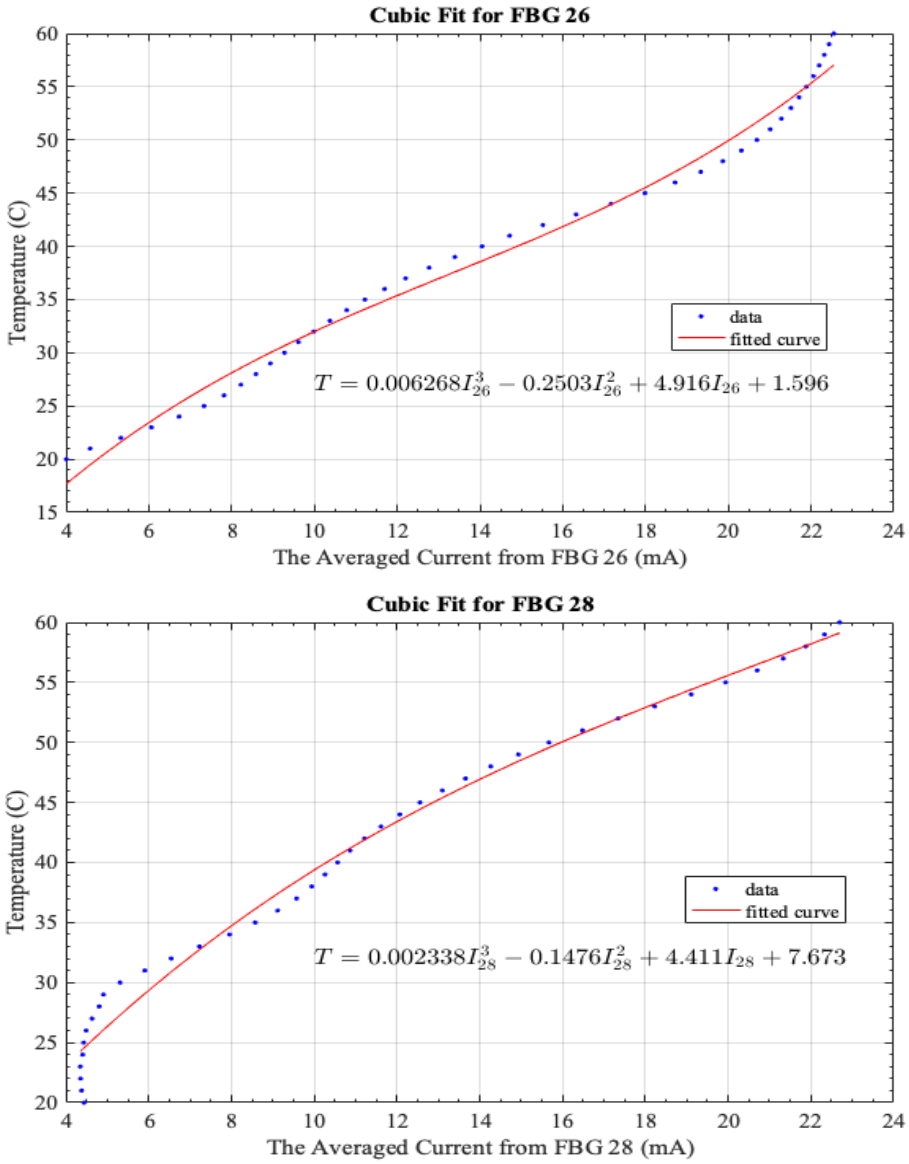

Figure 16: Cubic Fits For Each Graph

3.2.4 Testing the Reliability of Fittings

Another MATLAB code was built to test the efficacy of the fits. The code is as below:

```
T26_lin=1.821 * I_26 + 14.06;
T28_lin=1.723 * I_28 + 22.41;

T26_cubic=0.006268 * (I_26.^3) - 0.2503 * (I_26.^2) + 4.916 * I_26 ...
    + 2.896;
T28_cubic=0.002338 * (I_28.^3) - 0.1476 * (I_28.^2) + 4.411 * I_28 ...
    + 7.673;
T28_fifth=0.0002841 * (I_28.^5) - 0.01898 * (I_28.^4) + 0.4812 * ...
    (I_28.^3) - 5.783 * (I_28.^2) + 34.95 * I_28 - 51.80;
plot(Time',current_Temperature,'b')
set(gca,'Xminortick','on')
set(gca,'Yminortick','on')
set(gca,'fontname','times','FontSize',16)
set(gcf, 'Position',[0 0 1000 800]);
ylim([38.2 40.4])
xlim([0 100])
hold on
plot(Time',T26_lin,'g')
hold on
plot(Time',T28_lin,'y')
hold on
plot(Time',T26_cubic,'m')
hold on
plot(Time',T28_cubic,'r')
hold on
plot(Time',T28_fifth,'k')
xlabel('Time (h)')
ylabel('Temperature (C)')
title('Plotted Temperatures for each Model')
legend('Actual Temperature','FBG 26 Linear Model','FBG 28 Linear ...
    Model','FBG 26 Cubic Model','FBG 28 Cubic Model','FBG 28 Fifth ...
    Degree Model')
grid on
```

The system was set to $40°C$ for over 4 days. Then using the outputted current across these 4 days from either channel, they are substituted back into the 4 relations and plotted out to see how they compare to the actual measure. The plots can be seen in figure 17.

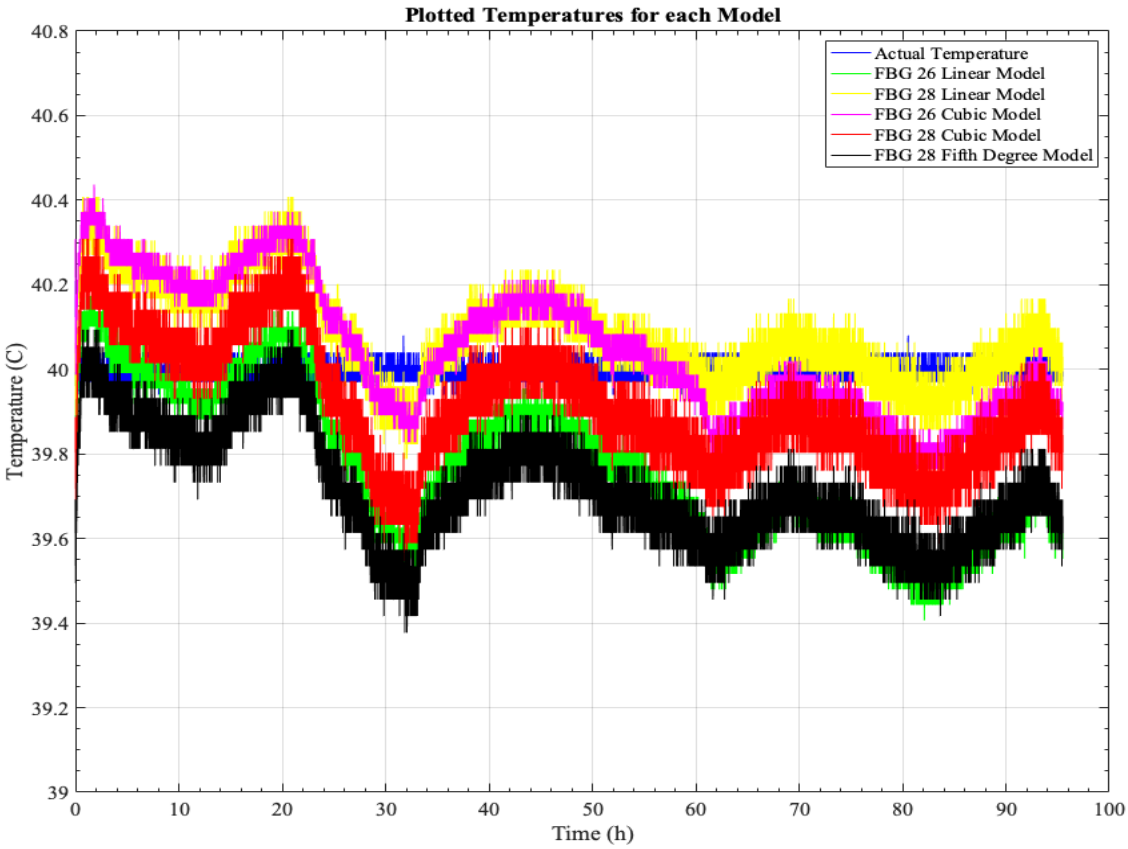

Figure 17: The predictive temperature of each of the 4 relations

The graphs show a great correlation with a very small margin of error that signifies that the fits are reliable and sufficient to be used as temperature sensors.
Furthermore, I went beyond and calculated what would a fifth-degree fit look like. The equation is as follows

$$T = 0.0002841\ I_{28}^5 - 0.01898\ I_{28}^4 + 0.4812\ I_{28}^3 - 5.783\ I_{28}^2 + 34.95\ I_{28} - 51.80 \qquad (3.5)$$

The output from all 5 equations shows a very close correlation to the actual measure. Which explains why we went for a linear model for the fits, to use simple and elegant equations that will have similar results as the other more complicated and longer fits for temperatures inside the selected range.

Part IV

Final Notes and Acknowledgement

It was an absolute honor and privilege to work closely with such an established group of physicists and engineers to build a reliable and very lucrative system. I want to express gratitude to my supervisor Dr. Francesco Fienga and Dr. Zoltan Szillasi and Dr. Noemi Beni welcoming me into the research group and explaining the background of the research team and providing me with access to the server that allowed me to control the interrogation system. A huge thank you goes to Dr. Vincenzo Marrazzo for directing and orienting me step by step throughout the entire Calibration work.
Also I want to thank Roberto Perruzza for constantly checking up on me and offering help and guidance on a personal level, and for Ali Karaki for providing pictures of the CMS's Beam pipe and the FBGs installed on it.
To sum up the Summer Student Program was truly a very engaging and fulfilling journey that I will cherish forevermore.

References

[1] Francesco Fienga, Vincenzo Romano Marrazzo, Sara Bennett Spedding, Zoltan Szillasi, Noemi Beni, Andrea Irace, Wolfram Zeuner, Austin Ball, Vittorio Giorgio Vaccaro, Benoit Salvant, Salvatore Buontempo, and Giovanni Breglio. Fiber bragg grating sensors as innovative monitoring tool for beam induced rf heating on lhc beam pipe. *Journal of Lightwave Technology*, 39(12):4145–4150, 2021.

[2] Francesco Fienga, Zoltan Szillasi, Noemi Beni, Andrea Irace, Andrea Gaddi, Wolfram Zeuner, Austin Ball, Salvatore Buontempo, and Giovanni Breglio. A fiber optic sensors monitoring system for the central beam pipe of the cms experiment. *Optics Laser Technology*, 120:105650, 2019.

[3] Ali Karaki. Ls2 report: Installation of the cms beam pipe. 2021.