Electron Cloud in Steel Beam Pipe vs Titanium Nitride Coated and Amorphous Carbon Coated Beam Pipes in Fermilab's Main Injector

Michael Backfish

Submitted to the faculty of the University Graduate School in partial fulfillment of the requirements for the degree Master of Science in the Department of Physics, Indiana University,

April, 2013

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Master of Arts.

Master's Thesis Committee

Dr. Shyh-Yuan Lee, Chairperson

Dr. Robert Zwaska

Dr. Rex Tayloe

Dr. Mike Snow

Acknowledgement

I would like to acknowledge Bob Zwaska, my primary advisor at Fermilab and Cheng-Yang Tan, both of whom participated in this Electron Cloud research and reviewed this thesis. I would also like to acknowledge my committee chair, Shyh-Yuan Lee, and the members of the thesis review committee. I would like to thank Susan Winchester, Dan Johnson, Bob Mau and Mary Kohler for making my participation in USPAS possible. Acknowledgement also goes to Kevin Duel for the necessary engineering. I am forever indebted to my wife, Libby Backfish, because without her support this would not have been possible.

Electron Cloud in Steel Beam Pipe vs Titanium Nitride Coated and Amorphous Carbon Coated Beam Pipes in Fermilab's Main Injector

Michael Backfish

This paper documents the use of four retarding field analyzers (RFAs) to measure electron cloud signals created in Fermilab's Main Injector during 120 GeV operations. The first data set was taken from September 11, 2009 to July 4, 2010. This data set is used to compare two different types of beam pipe that were installed in the accelerator. Two RFAs were installed in a normal steel beam pipe like the rest of the Main Injector while another two were installed in a one meter section of beam pipe that was coated on the inside with titanium nitride (TiN). A second data run started on August 23, 2010 and ended on January 10, 2011 when Main Injector beam intensities were reduced thus eliminating the electron cloud. This second run uses the same RFA setup but the TiN coated beam pipe was replaced by a one meter section coated with amorphous carbon (aC). This section of beam pipe was provided by CERN in an effort to better understand how an aC coating will perform over time in an accelerator. The research consists of three basic parts: (a) continuously monitoring the conditioning of the three different types of beam pipe over both time and absorbed electrons (b) measurement of the characteristics of the surrounding magnetic fields in the Main Injector in order to better relate actual data observed in the Main Injector with that of simulations (c) measurement of the energy spectrum of the electron cloud signals using retarding field analyzers in all three types of beam pipe.

Contents

1 Introduction

Electron cloud (hereafter ecloud) buildup inside a beam pipe can be triggered by three known mechanisms in proton synchrotons. First, high energy particle beams can interact with residual gases within the vacuum chamber and trigger the release of electrons. Second, synchrotron radiation can interact with the chamber wall and through the photoelectric effect cause the release of more electrons. Finally, actual beam loss can interact with the chamber wall causing the release of secondary electrons. These non-relativistic electrons, created by any one of these mechanisms, can be accelerated transversely by the electromagnetic field of the beam. Their newly acquired transverse momentum can cause secondary electron emission when they collide with the vacuum chamber wall. With proper bunch spacing, this larger group of electrons can receive another electromagnetic kick from the next bunch of protons and cause even more secondary emissions from the vacuum chamber wall. This avalanche process can build up high enough electron densities to cause instabilities in the primary proton beam. These instabilities have been observed in beams at PSR (Proton Storage Ring) [1], RHIC (Relativistic Heavy Ion Collider) [2], SNS (Spallation Neutron Source) [3], SPS (Super Proton Synchrotron) [4], LHC (Large Hadron Collider) [5], and was first observed in Fermilab's Main Injector in 2006 [6] using an RFA designed by Richard Rosenberg of ANL (Argonne National Laboratory) [7].

2 Main Injector

The Main Injector is a 2 mile long ring with an injection energy of 8 GeV and an extraction energy of either 120 GeV for neutrino and anti-proton production or 150 GeV for injection into the Tevatron. Figure 1 shows a bird's eye view of the Fermilab site and MI-52 where the ecloud measurement setup is located.

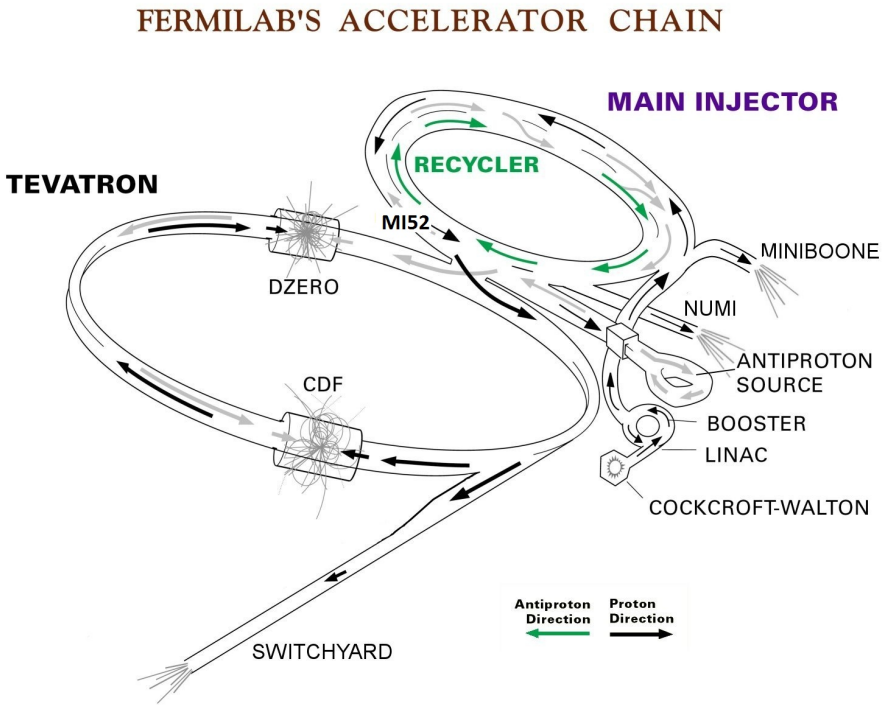


Figure 1: Fermilab's Accelerator Chain. MI52 is the location of the Ecloud setup.

The highest intensities in the machine are currently reached during mixed mode slip stacking pulses, which are repeated every 2.2 s. Mixed mode implies that both protons destined for the Numi (Neutrinos from the Main Injector) experiment and antiproton production are loaded into the machine during the same cycle. Slip stacking is an RF manipulation that allows the bucket to be filled twice before acceleration, thus increasing the total beam intensity [8]. For more information on slip stacking

Table 1 Main Injector Parameters	
Energy	8–120 GeV
Circumference	3319.4 m
RF frequency	52.8–53.1 MHz
Bunch Intensity	12×10^{10} (slip-stacked)
Bunch Spacing	19 ns
Bunch Length	1–10 ns @ 95%
Beam Admittance	40π mm·mrad
Beam Emittance	15π mm·mrad
Beam Pipe Inner Diameter	
steel	149.2 mm
TiN	149.2 mm
aC	155.0 mm

see Appendix A. During mixed mode slip stacking proton intensities can reach levels up to 45×10^{12} particles. The basic Main Injector parameters used for this study can be found in Table 1.

3 The Ecloud Measurement Setup

The secondary emission yield (SEY) of the vacuum chamber wall is the primary factor in the buildup of eclouds. While the ecloud has not limited beam operations in the Main Injector, this may change as intensities are increased to meet the needs of future projects. In this paper, the focus will be on how different beam pipe coatings affect the production of secondary electrons. A setup that consists of two identical sections of beam pipe, one coated with a test material and the other with no coating, was installed in a straight section of the Main Injector in 2009. Four retarding field analyzers (RFAs) were installed to directly measure the electron flux within these sections of beam pipe. RFA1, RFA2 and RFA3 use a new design modeled to improve collector sensitivity by 20% [9] while RFA4 uses the original Rosenberg design that was used in the MI in 2006 [6]. This analysis only uses RFA1, RFA2 and RFA3. The beam pipe and detectors can be seen in Figure 2. A close-up view of an improved RFA can be seen in Figure 3.

The RFAs directly measure the number of electrons that exit through slots cut into the beam pipe. When the electrons exit the beam pipe, they encounter the electric field from a grid, which is a fine mesh screen that can be charged to between 0 and -500 V. Beyond the grid, the electrons are absorbed by a detector cup that is connected to a 40 dB preamplifier and a low-pass filter with a 3 dB attenuation point at 3 kHz [9]. These electronics can be turned on or off so that either the raw signal or the amplified signal can be monitored. By increasing the grid voltage, only the electrons with energies exceeding the grid potential are detected. Therefore, energy spectrum measurements can be made by varying the grid potential. L. McCuller found the efficiency of the RFA to be 90% with at least -20 V on the grid. With potentials of a lesser magnitude, secondary emissions off the collector decrease the collection efficiency [10].

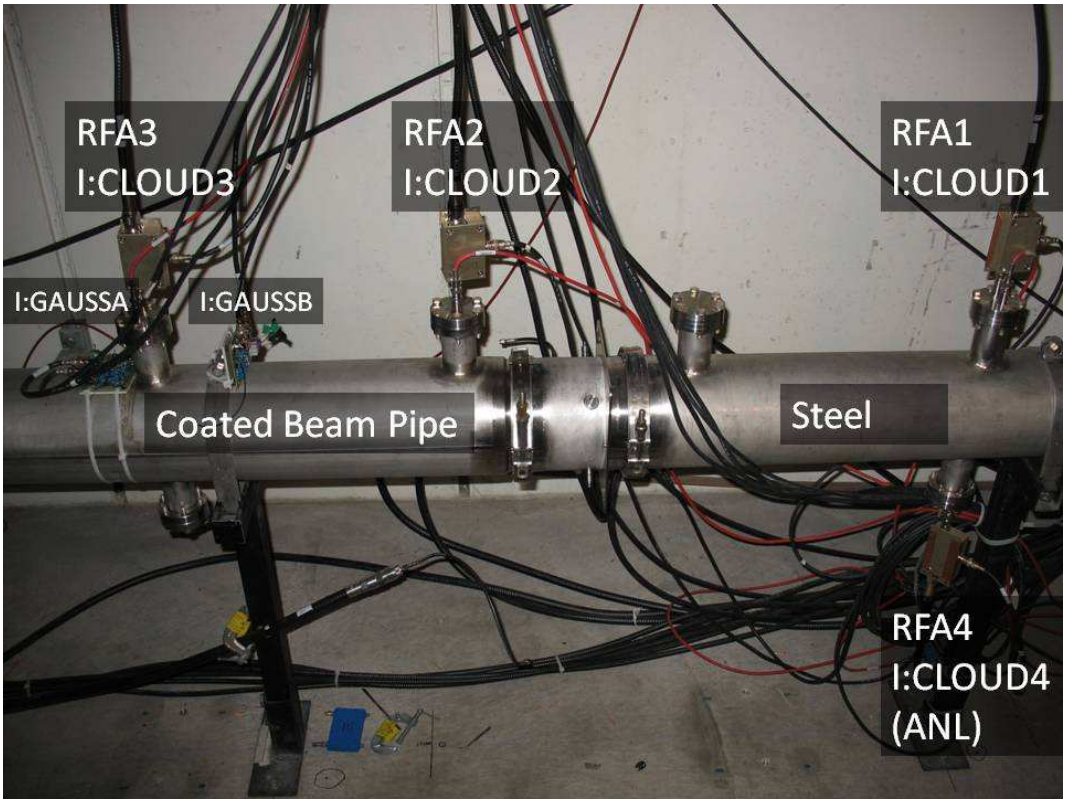


Figure 2: The ecloud measurement setup in the Main Injector. The primary setup consists of four RFAs. The beam pipe is 6" in diameter and the coated and uncoated sections are each 1 meter long. The setup is located in a straight section as to avoid electron confinement from magnets. Stray magnetic fields are analyzed in section 5.

The power supply for the RFA grids and the signal from the RFAs are all incorporated into Fermilab's control system. The power supply has remote controls with analog readbacks, which along with the RFA signals are used as inputs into a multiplexed analog to digital converter (MADC). This allows the ecloud signals to be time correlated to the Main Injector ramp and is essential for correlating an RFA signal with the beam intensity using the pre-existing DC Current Transformer. The control system is also used to record data for each individual Main Injector pulse. The gap between the grid and the detector in the RFA breaks down when the potential on the

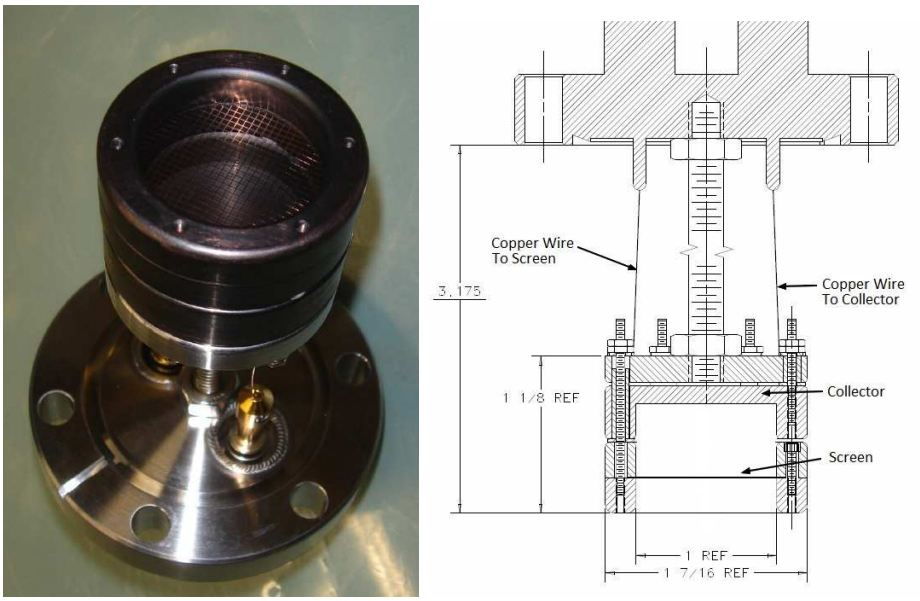


Figure 3: Retarding Field Analyzer

grid exceeds −500 V. Based on detector efficiency and detector break down, the grid voltages are limited to a range between −20 V and −500 V.

4 Experimental Procedures

Conditioning is the process where the bombarding electrons change the surface chemistry of the beam pipe. As the beam pipe conditions its secondary emission yield becomes lower. With a lower secondary emission yield comes a lower ecloud signal. Thus by tracking changes in the relationship between the Main Injector beam intensity and the ecloud signals, the rate of conditioning of the coated beam pipes and the steel pipe can be compared.

Due to limitations in the amount of data that can be written to file with the present control system, only one point from each RFA is logged after a Main Injector timing event. The time is chosen to obtain the maximum magnitude of the signal that each of the RFAs detects as illustrated in Figure 4. This data point is then saved for each of the Main Injector beam cycles. By coupling this data with full traces taken from a much smaller subset of data, the net charge deposited into the beam pipe can be calculated.

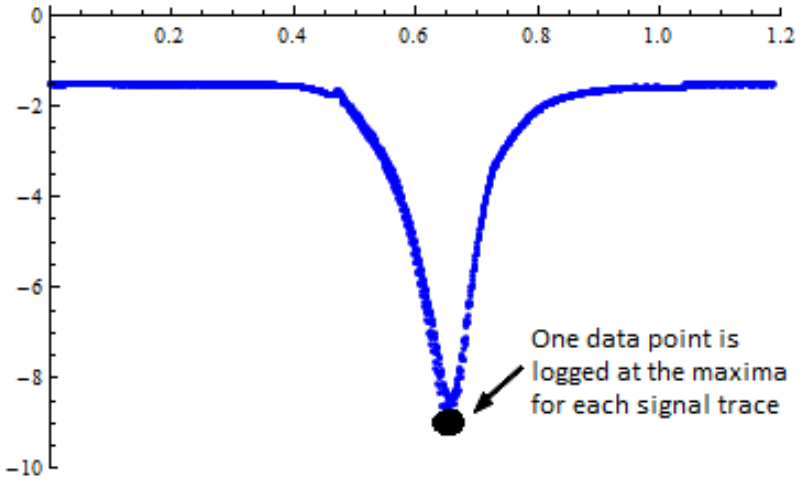


Figure 4: This figure shows a full signal trace with the logged data point marked at the point of maximum ecloud.

4.1 Timing Jitter

Data collection was complicated by a problem inherent in the controls system. The procedure described above requires each signal to be logged when the signal is at its maximum strength, which is the minimum value of the the negative signal. As illustrated in Figure 5, the logged data point is not always at this minimum value. Experts report that there is a problem in the controls system when multiple requests occur at once, a scheduled upgrade will fix this in the future. To ensure that this offset was indeed inherent to the data collection system, the Main Injector bend bus was logged in a similar manner. Since the bend bus is consistent from cycle to cycle one would expect a perfect logged signal to not have this jitter, but it did. To work around this problem, the data taken daily is visually analyzed to determine if the jitter is occurring. Any timing jitter will result in data with a smaller signal strength for the same Main Injector intensity. If the visual observation shows the two characteristic signals, an initial Mathematica fit of the data is performed. An offset is manually adjusted and added to this initial data. The offset fit is used to filter the poorly fit data. An example of the filtering can be seen in Figure 6 and in Appendix F.4. This method of filtering data is used throughout this analysis.

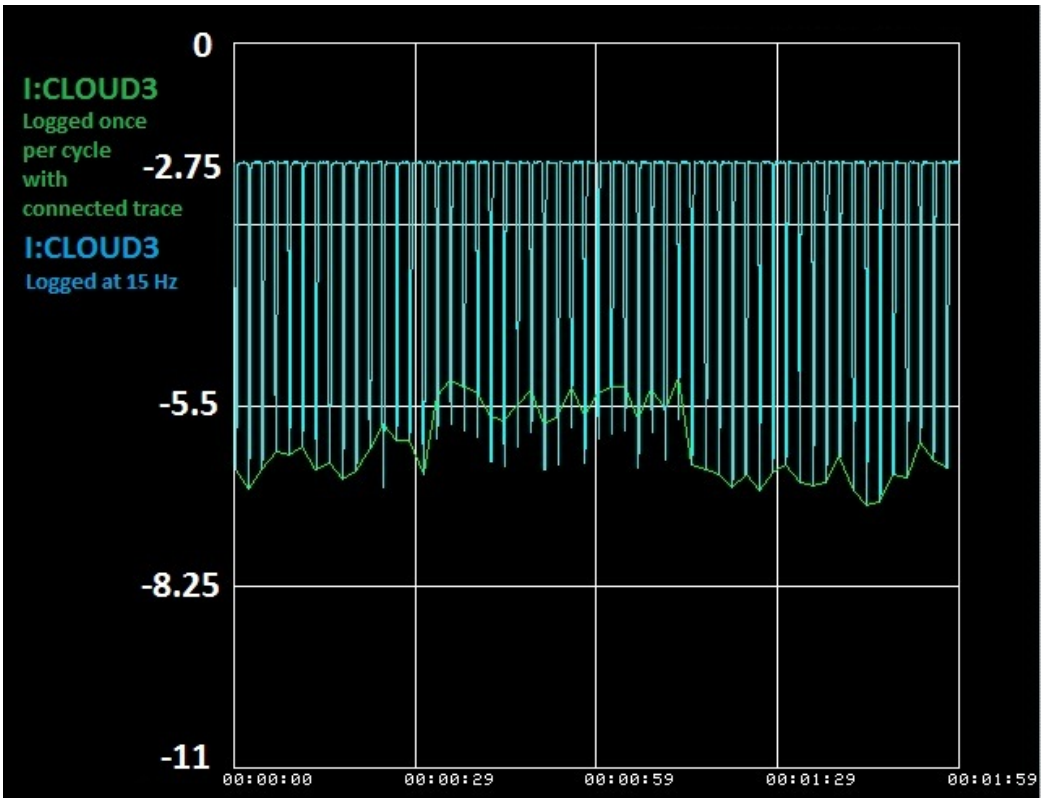


Figure 5: The data shown is from the same device both logged at 15 Hz and on an event timed to catch the maximum signal strength. It is plotted over 54 beam pulses. The individual points are connected by lines and are plotted in green. The 15 Hz signal is plotted in blue. Logged points do not always match up to the maximum signal strength shown on the 15 Hz green trace due to inherent problems in data collection.

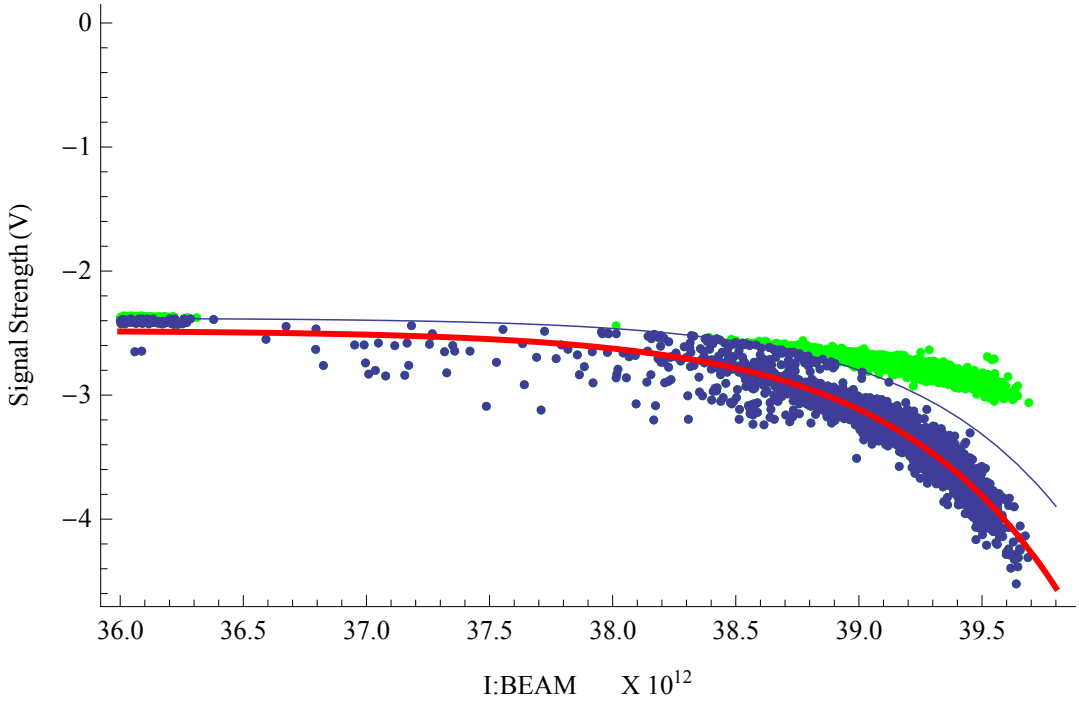


Figure 6: To compensate for the data that is not logged at the maximum signal strength, the data is first fit as a whole. An offset is added to this fit, shown by the blue curve. The data above this curve, shown in green, is removed. The remaining data, shown in blue, is fit a second time without the invalid data. The final fit is shown in red.

4.2 Comparing TiN coated Beam Pipe with Steel

On September 11, 2009, Main Injector beam operations began following the Fermilab summer shutdown. Proton beam intensities started low with non-slipstacked beam. The first ecloud signals were seen on the detectors when slipstacking operations began on September 15. Initial signal strengths were quite strong and exceeded the 100 mV maximum that the preamps are able to handle. This early data was taken without the low pass filter and preamps on. Initial signals in this configuration can be seen in Figure 7. This data shows that while the ecloud signals are smaller in the TiN coated beam pipe, they are not eliminated. The signals seem to be roughly half the size of the detected signals in the stainless steel beam pipe.

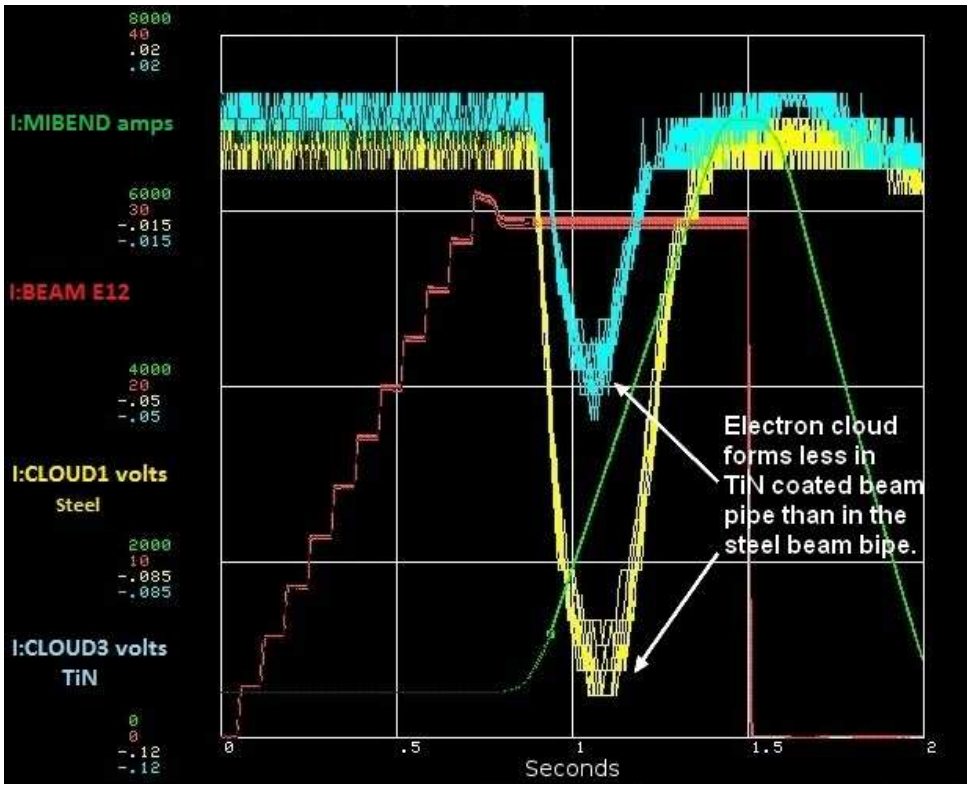


Figure 7: Initial detector signals displayed without amplification. I:CLOUD1 (steel), shown in yellow, is displayed with I:CLOUD3 (TiN), shown in blue.

In order to better characterize the differences between the TiN coated and standard steel beam pipes, their conditioning over time must be understood. In Figure 8 the Main Injector beam intensity, I:BEAM, is plotted with detector signals, I:CLOUD1 (RFA1 steel), I:CLOUD2 (RFA2 TiN), and I:IP521C (an ion pump at the same location) over a period of just under four days. Both detectors are plotted on the same scale to give a direct comparison of conditioning. Over this short time frame, both surfaces condition at a similar rate. As would be expected, the vacuum improves as the beam pipe conditions and the RFA signals decrease. By September 25, the signals have decayed sufficiently as a result of conditioning, permitting the preamps to be turned on for all the RFAs. Since I:CLOUD4, or RFA4, is a different type of detector, only I:CLOUD1 (RFA1), I:CLOUD2 (RFA2), and I:CLOUD3 (RFA3) are used for making comparisons. In Figure 9, the signals from I:CLOUD2 (TiN) and I:CLOUD3 (TiN) are no longer half the height of I:CLOUD1 (steel). This implies that the two types of beam pipe have conditioned at different rates.

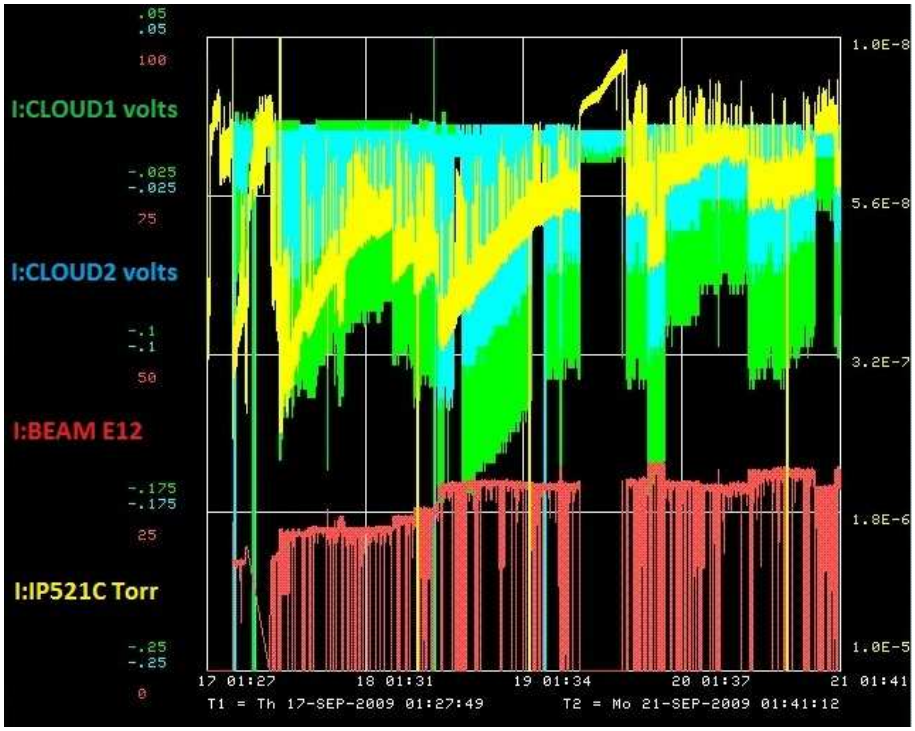


Figure 8: Conditioning over time can be monitored by monitoring the signal maxima for both steel and TiN coated beam pipes. The green trace is I:CLOUD1 (steel), while the blue trace is I:CLOUD2 (TiN), both datalogged at the maximum signal intensity and displayed on the same voltage scale. The yellow trace is an ion pump in the region and the red trace is the Main Injector beam intensity. These plots illustrate that both are conditioning at similar rates over these time periods.

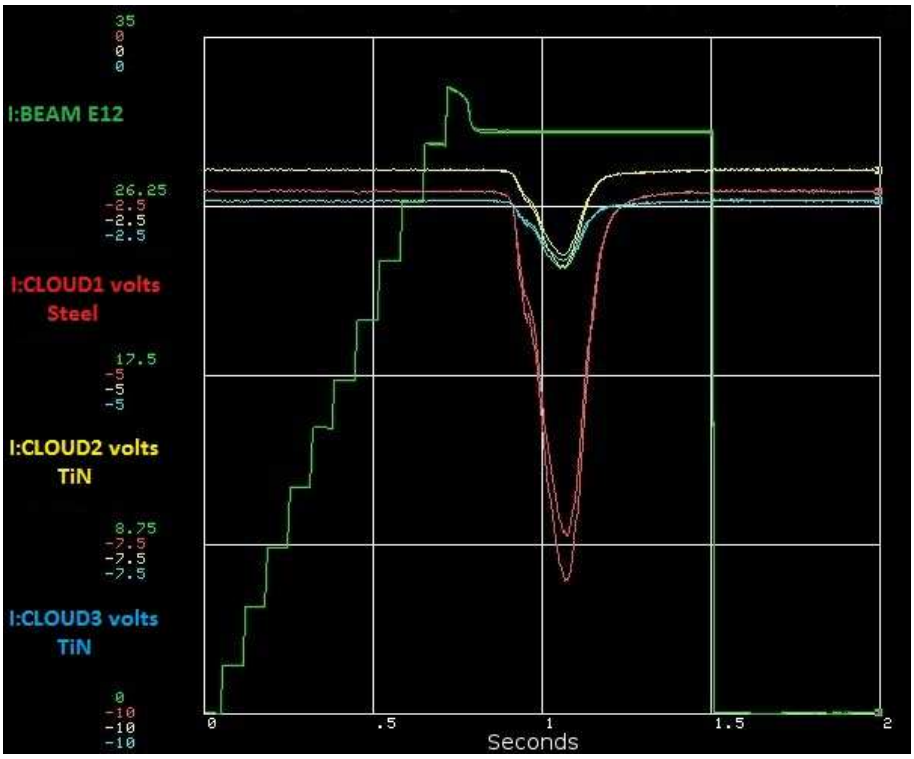


Figure 9: The first comparison with all of the preamps on illustrates that the TiN signals, plotted here in blue and yellow, are smaller than the detector signal located in the steel beam pipe, plotted here in red.

Small variations in Main Injector beam intensity result in significant changes in ecloud signals. To properly characterize signals, it is essential to correlate each detector signal with the Main Injector beam intensity. The Main Injector usually operates with a period of 2.2 seconds between beam cycles. By logging each RFA signal at the point in the ramp where the maximum ecloud signal is detected, and coupling that signal maxima with a proton beam intensity data point, the relationship between signal strength and beam intensity can be characterized. A plot showing the correlation between signal strength and beam intensity can be seen in Figure 10. Here all the detectors are plotted on the same scale with -28 V on the grid.

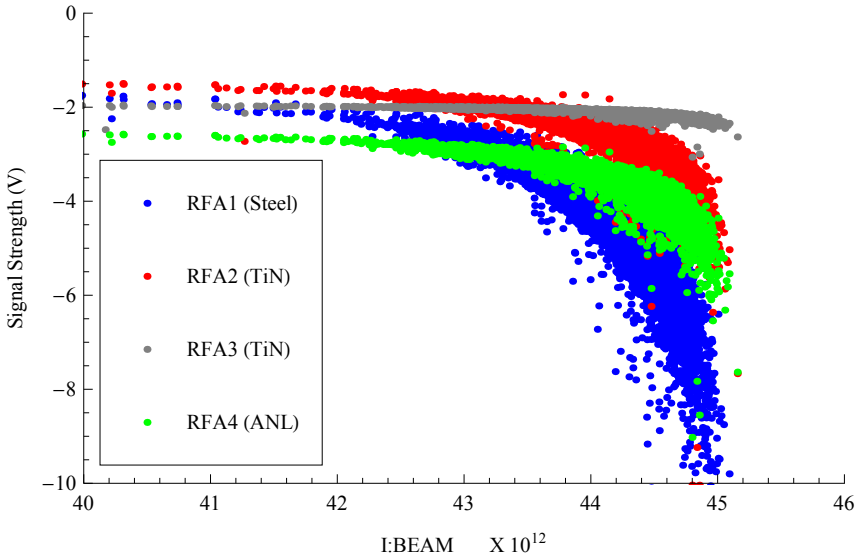


Figure 10: This plot shows the data for the four Ecloud detectors from March 1, 2010. The peak signal strengths are plotted vs the proton beam intensity (I:BEAM). This data was taken after six months of conditioning for the TiN and steel beam pipes.

By plotting the signal strength as a function of beam intensity, changes in this relationship can be tracked over time. Equation (1) was chosen as an empirical fit for the relationship. The ecloud signals are expected to saturate at some point as the intensities increase, thus the equation is only representative of current data. In this

equation, V represents the maximum signal strength that the RFA signal reaches. The variable z represents the voltage offset from the preamp which has a small drift over time. The terms a and x_0 are fit variables and x is the independant variable, which is the beam intensity. The purpose of this equation is to track the x_0 value. When x is equal to x_0 our exponential term becomes e^0 and thus $V(x; a, x_0, z) = z - 1$. Thus x_0 represents the Main Injector beam intensity where the ecloud signal contributes -1 V to the total signal. By tracking this benchmark over time, the conditioning of the steel beam pipe versus the coated beam pipes can be compared. Figure 11 illustrates $V(x)$ fitted to actual data.

$$V(x; a, x_0, z)[V] = z[V] - (1\ [\mathrm{V}]) \times e^{a(x-x_0)} \tag{1}$$

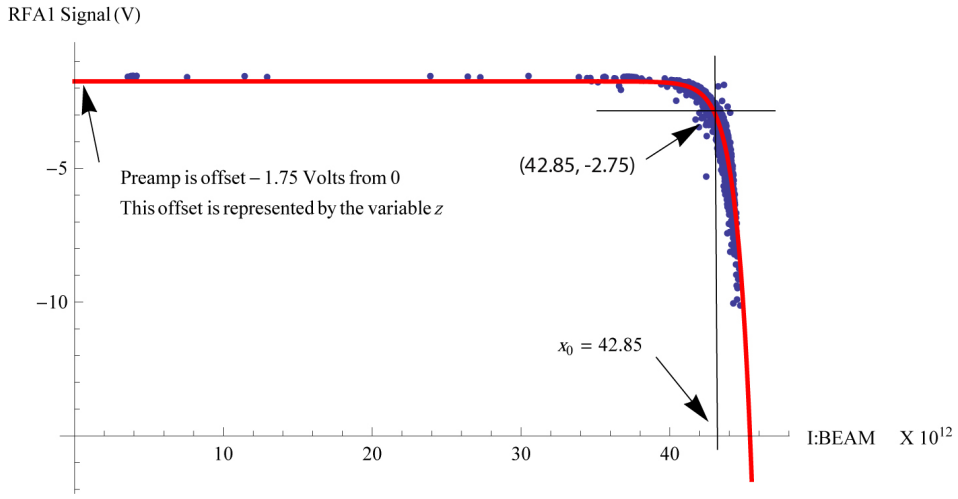


Figure 11: In the image above, a Main Injector beam intensity of 42.85×10^{12} particles is required to produce a -1 V difference in the peak detector signal. By daily tracking this -1 V benchmark, the beam pipe conditioning is tracked over time.

The first step in the analysis was to establish an x_0 term daily. The ecloud signals do not always fall within the range of our preamps, thus some data must be ignored. As a result, some of the x_0 data are extrapolated based on values from days with

data. Figure 12 shows all of the experimental data for this -1 V term without the extrapolated data. Mathematica was used for fitting Equation (1) to the daily data and for finding the x_0 value. This program is listed in Appendix F.1.

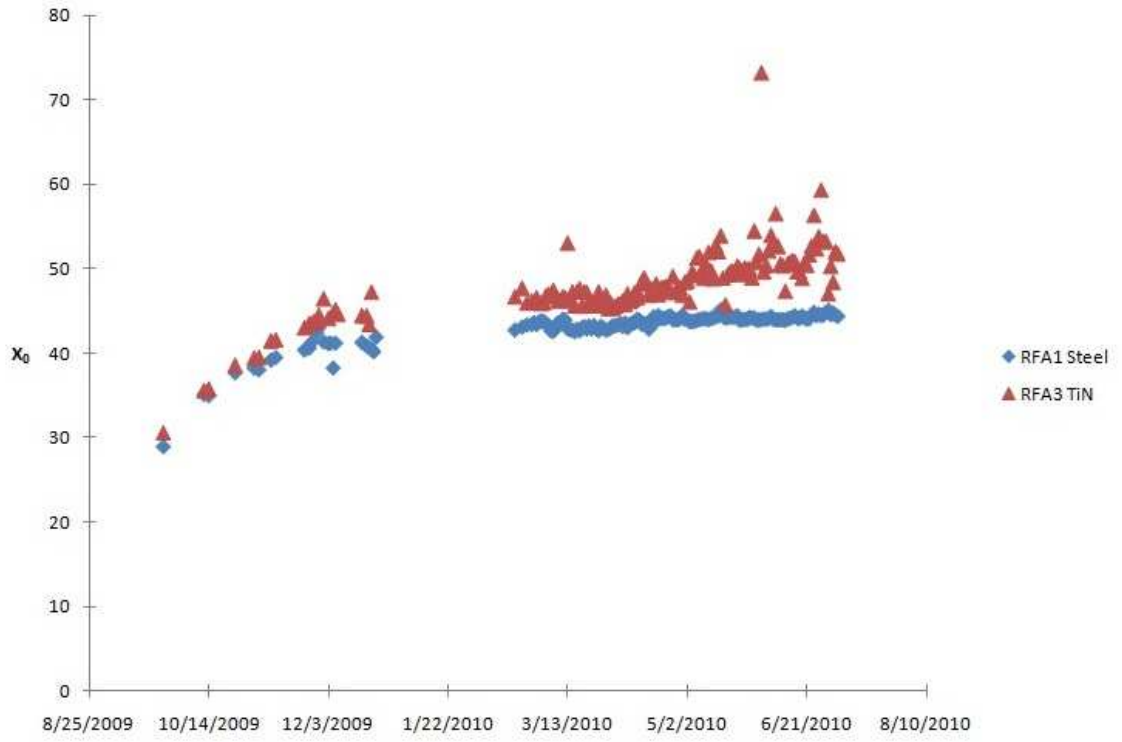


Figure 12: The vertical axis of this plot is the beam intensity where the ecloud signal exceeds the benchmark -1 V signal. Two different fitting methods were used for the z value which is the signal offset. From the start of the run to April 30, the variables were all fitted together. From May 1 on, the z value was first fitted using only data at low Main Injector intensities where there was no ecloud. This value was then used for fitting the rest of the data. Note, the error becomes larger on RFA3's signal as the ecloud becomes small due to conditioning as illustrated by the larger deviation from the mean later in the plot. This error due to small signals is illustrated in Figure 13.

These data provide daily relationships between the Main Injector beam intensity and the signal maxima associated with each detector in the exponential form of Equation (1). The proton intensity is logged for every Main Injector beam pulse. Some

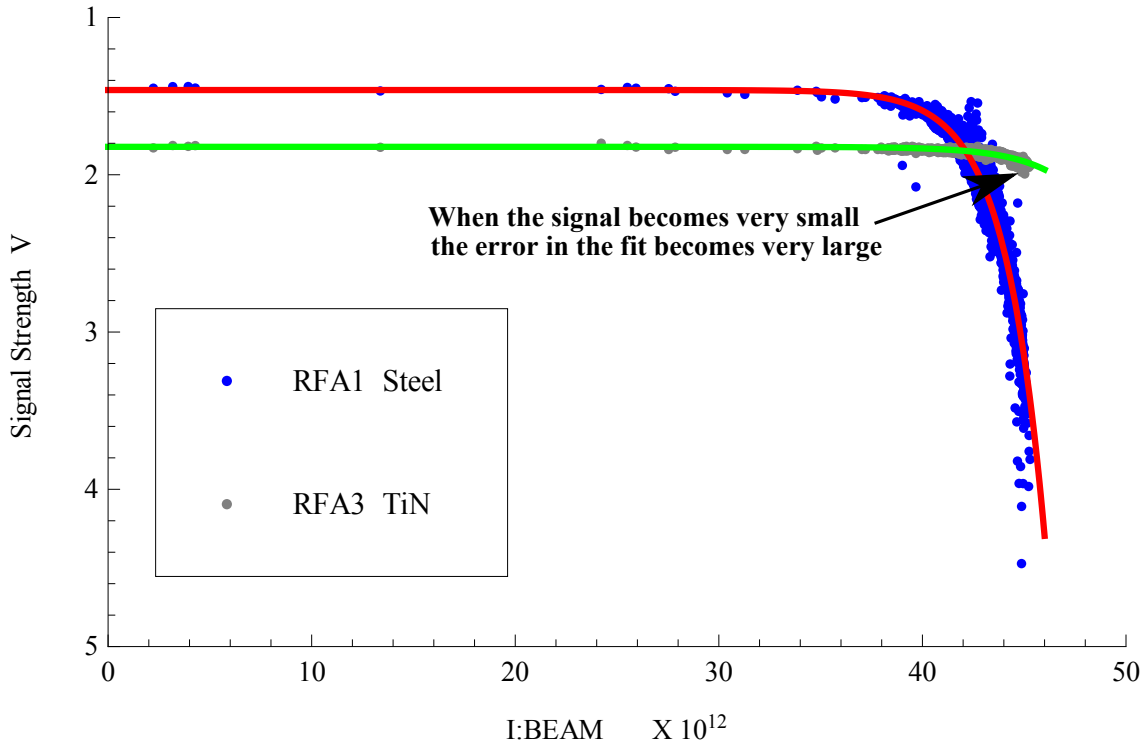


Figure 13: The blue data shows a very strong signal while the gray data signal is only slightly above background. As the ecloud signal becomes weak the data fit becomes worse.

ecloud data is missing due to preamp range limits and data acquisition problems. Using the daily relationships between Main Injector proton intensity and ecloud detector maximum signal strength, ecloud maxima data are extrapolated for every Main Injector cycle and each ecloud detector. These extrapolated voltage maxima, V_{max}, used in Equation (2), are required to recreate the full ecloud signal traces for every Main Injector pulse.

Lorentzian functions, shown in Equation (2), are then fitted to small samples of detailed signals as illustrated in Figure 14. These fits are purely empirical and do not relate to the physics of the ecloud, just the shape of the signal trace. These fitted samples are used to obtain λ in the Lorentzian equation. The variable t_{max}

is the time when the Lorentzian function is at the peak, and is the time the ecloud signal maxima are datalogged. The variable z is known for each day from the offset found in the daily fits. The variable V from Equation (1) is equal to V_{max} from Equation (2) when t is equal to t_{max}. The variable A relates to the amplitude of the Lorentzian function and can be solved by using the other known variables. Using this method, Lorentzian functions representing the signal for every Main Injector pulse are obtained. The program used for fitting Lorentzian functions to actual data is listed in Appendix F.2.

$$V_{max}(t; t_{max}, z, A, \lambda) = z - \frac{A}{[(t - t_{max})^2 + \lambda]} \tag{2}$$

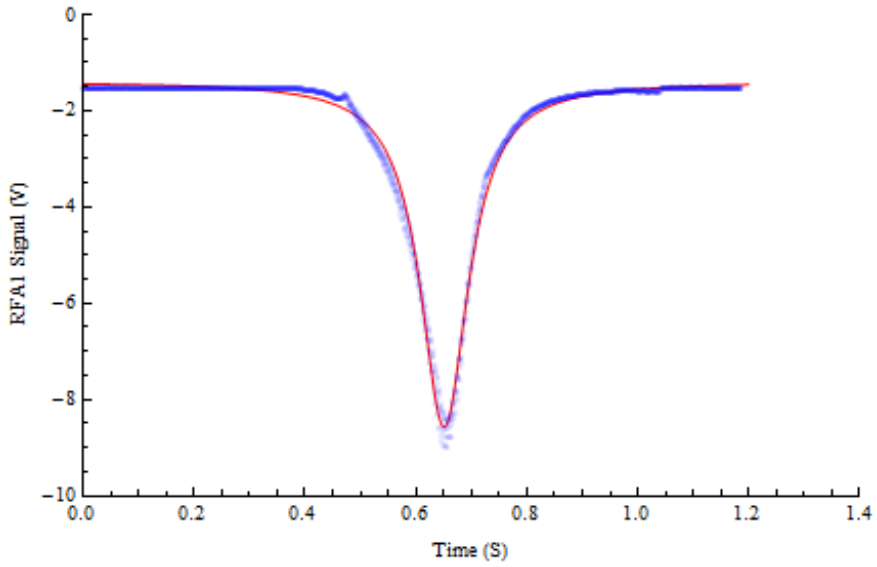


Figure 14: This shows the signal from RFA1 (Steel) fitted with a Lorentzian function.

The Lorentzian representation of the signal is integrated to find a value that is proportional to the total absorbed charge on the wall of the beam pipe at the location of the detectors. The absorbed charge, in the units of e/cm^2, is found by accounting for the RFA efficiency, beam pipe slit area, and preamp gain. This information is

then used to find a daily integrated charge per cm^2. The program used to determine the daily integrated charge is listed in Appendix F.3. When x_0, the -1 V benchmark, is plotted against the number of electrons absorbed per cm^2, the condition rates can be compared. Figure 15 illustrates the data comparing RFA1 (steel beam pipe) with RFA3 (TiN coated beam pipe). The red data points show that after almost ten months of running the steel beam pipe is still conditioning. The blue points show a hook shape that starts pointing up. This is due to the integrated charge becoming very small. Consequently, the fit becomes poor due to the small signals as was illustrated earlier in Figure 13. Figure 15 shows that the TiN signals became very small after about 4×10^{16} electrons per cm^2 were absorbed. This illustrates that with no increase in Main Injector beam intensity, after enough conditioning the ecloud does not form.

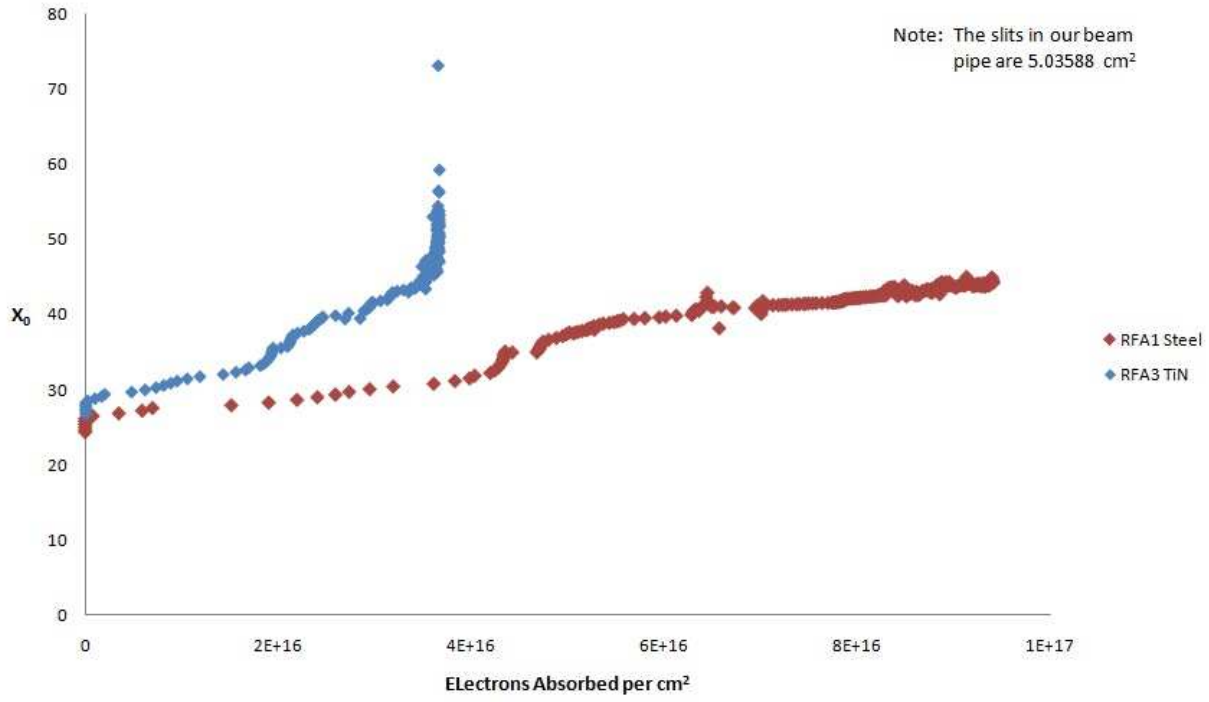


Figure 15: This plot illustrates the beam pipe conditioning plotted against the integrated charge. The steel beam pipe is still conditioning at the end of this run. The ecloud is no longer forming in the TiN lined beam pipe. When the signals get very small, the fit to the data becomes more error prone. This results in the sharp rise in the x_0 value, which is the beam intensity where the detector signal peaks at -1 V benchmark. It should be noted that while ecloud is not seen in the TiN beam pipe at the proton intensities for this time frame, there is no reason to believe that the ecloud would not still form at higher intensities.

4.3 Comparing aC coated beam pipe with both steel and TiN

During the 2010 summer shutdown at Fermilab, the TiN coated beam pipe was replaced with an aC coated beam pipe sent from CERN. It was installed and first saw electron signals in late September. Although signals were initially seen during pbar production from the Main Injector (which only uses two batches of MI beam), the analysis begins when Numi and pbar operations began running in the standard 11 batch slip stacking mode. This allows a direct comparison to the data obtained from the previous run. Figure 16 compares the ecloud signals from fourteen days into the first TiN run and five days into the second aC run. Five days into the aC run the signals were significantly smaller than they were fourteen days into the TiN run. This is a result of a quicker increase in MI intensities in 2010 versus 2009. This is illustrated in Figure 17 and is evidence for the need to compare conditioning based on absorbed charge rather than time.

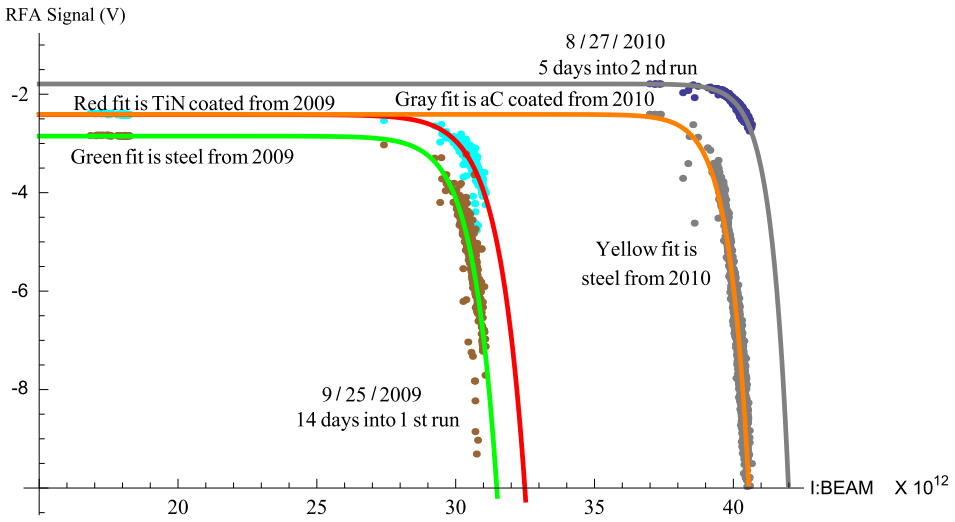


Figure 16: Large differences in conditioning time between the first run and the second run are evident in the detector signals.

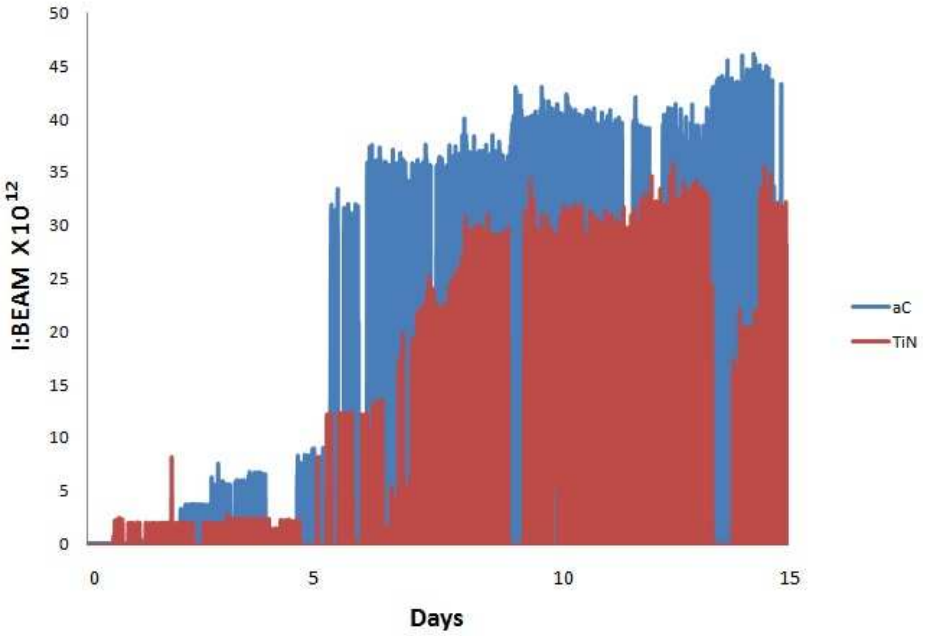


Figure 17: A comparison of Main Injector beam intensities for both runs. A quicker increase in proton intensities during the aC run resulted in faster beam pipe conditioning.

Initial signals from the aC coated beam pipe exhibited a double hump structure that had not been seen in earlier detectors (illustrated in Figure 18). As the beam pipe conditioned, this second dip became smaller, and the basic signal started behaving more like the steel signal. This double hump, and general shape, are not yet explained by either theory or simulation [11].

Once again, the Main Injector beam intensity is tracked where the ecloud contribution to the signal is -1 V. This x_0 benchmark is tracked over time for RFA1, RFA2 and RFA3 in Figure 19. As expected, the aC coated beam pipe starts out with signals much weaker than steel. The steel signals are initially too strong for the preamps thus limiting a direct initial comparison using RFA1. A vacuum leak near the downstream end of the installation resulted in a massive change in the apparent conditioning of the aC beam pipe on October 4, 2010. The leak is seen on both RFA2 (aC) and RFA3

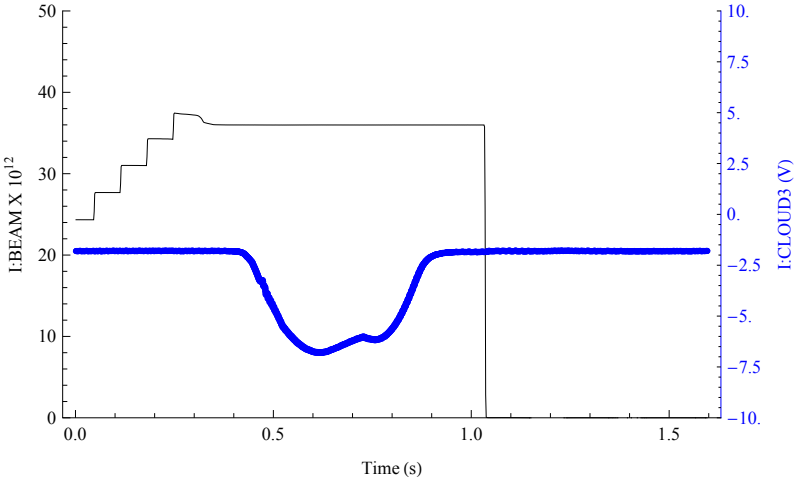


Figure 18: The detector located in the aC coated beam pipe (I:CLOUD3) initially had two peaks in contrast to the single hump of the steel beam pipe signals. This difference is yet to be explained.

(aC). RFA3 is the closest to the leak and was influenced most as seen in the large dip shown in Figure 19. Figure 20 shows a comparison of the conditioning of RFA1, RFA2 and RFA3 versus absorbed charge. In order to make this comparison, values for RFA1 from 8/23/2010 to 9/4/2010 have been constructed from its relationship to both RFA4 and RFA3, because there was no data from RFA1.

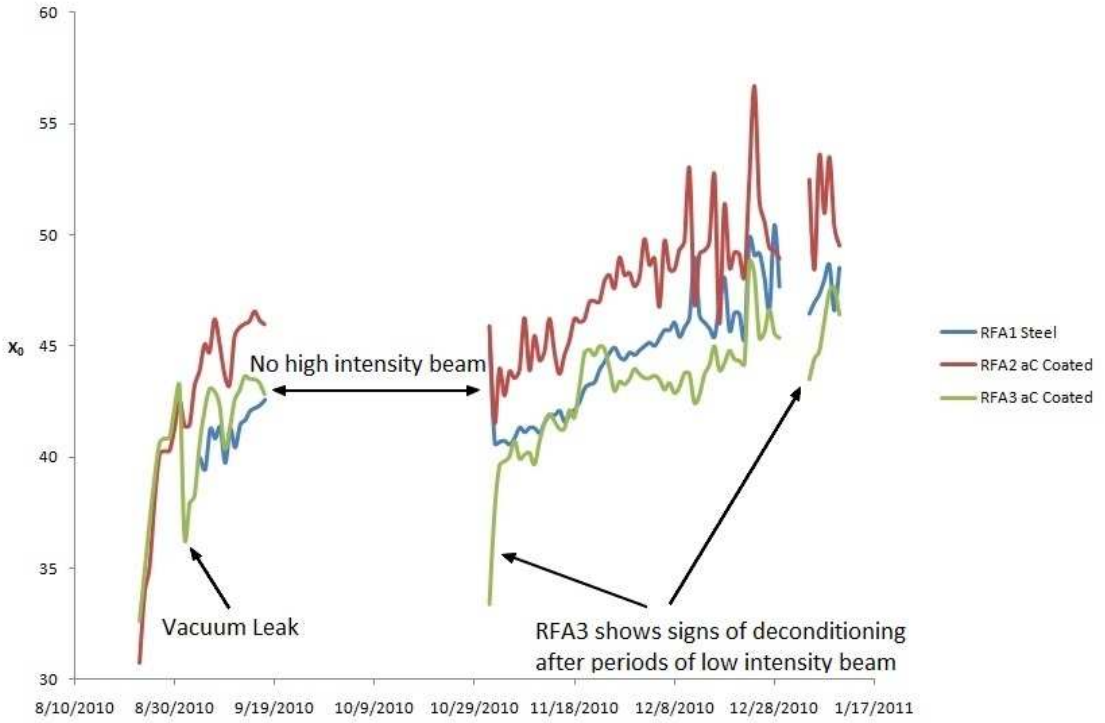


Figure 19: Initially, RFA1 (steel) detector signals were too strong which saturated the preamp, thus there is no data until September 4.

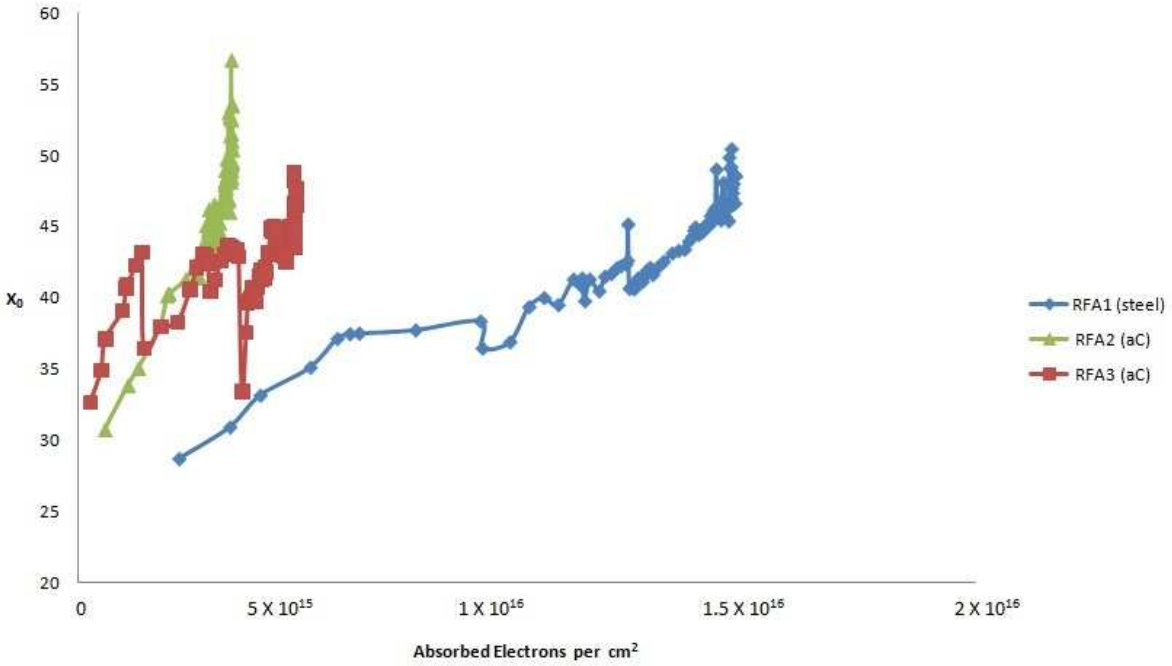


Figure 20: The x_0 value is the beam intensity in the Main Injector that results in a -1 V signal contribution from the ecloud when the cloud is at its peak. By plotting this benchmark versus the absorbed electrons per cm^2 one can better compare the conditioning of both steel and aC coated beam pipes. The data was taken from 8/23/2010 to 1/10/2011.

5 The Effects of Stray Magnetic Field From the Magnet Buses

Simulations suggest that the ecloud build up in the Main Injector should not decay away before the beam is extracted, which is contrary to what has been seen experimentally [11]. It was suggested that interference from magnetic fields from the buses could be a parameter that was not accounted for in the simulations. To test this hypothesis the coated section of beam pipe was wrapped in two layers of mu metal. Figure 21 shows the background fields in the tunnel measured with no current on any of the buses. Figure 22 shows the beam pipe wrapped in mu metal. Figure 23 shows the RFA signals before and after the mu metal wrap was intalled on the beam pipe. The change in the RFA signals after the mu metal wrap was installed shows that stray magnetic fields do influence the signals.

Two identical Hall probes were built to measure the stray fields in the tunnel. The 3-D magnetic probes, Probe A and Probe B, which can be seen in Figure 21 were initially installed to measure the fields on the top of the beam pipe. All three axes were added in quadrature to give a magnitude. The signal is sent to the control system and logged. These signals are named I:GAUSSA and I:GAUSSB. I:GAUSSA measured (4.9 ± 0.6) gauss at 95% confidence. I:GAUSSB measured (5.6 ± 0.4) gauss at 95% confidence. The procedure and data used for calibration of the hall probes can be found in Figure 36 in Appendix E. To better characterize the fields the Hall probes were moved to the bottom of the beam pipe. After this I:GAUSSA measured (6.4 ± 0.6) gauss at 95% confidence. I:GAUSSB measured (5.9 ± 0.4) gauss at 95% confidence. Simulations suggest that a field of 10 gauss may be enough to distort the formation of the ecloud [11]. Stray fields are not expected to affect the detector efficiency until the fields are much larger than 10 gauss [10]. The

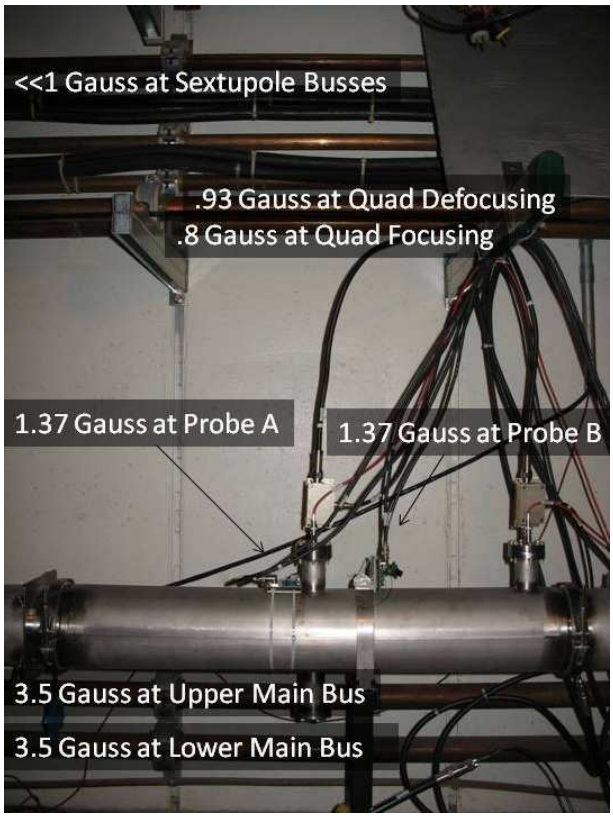


Figure 21: These are the residual magnetic fields measured in the tunnel.

measurements of the stray fields determined that they were indeed large enough to require more characterization. As a result, a third hall probe was built that allowed the measurement of each of the axes independantly. The probe was positioned, as illustrated in Figure 22, with the X-axis as the horizontal axis with the positive direction pointing radially outward from the center of the Main Injector. The Y-axis is the vertical axis with the positive direction pointing up. The Z-axis is the longitudinal axis with the positive direction pointing in the direction of proton travel. This data in Figure 24 shows that the field is almost exclusively in the negative Y direction. The vertical field reaches values as high as -5.8 gauss. This measurement independently verifies the earlier hall probe measurements.

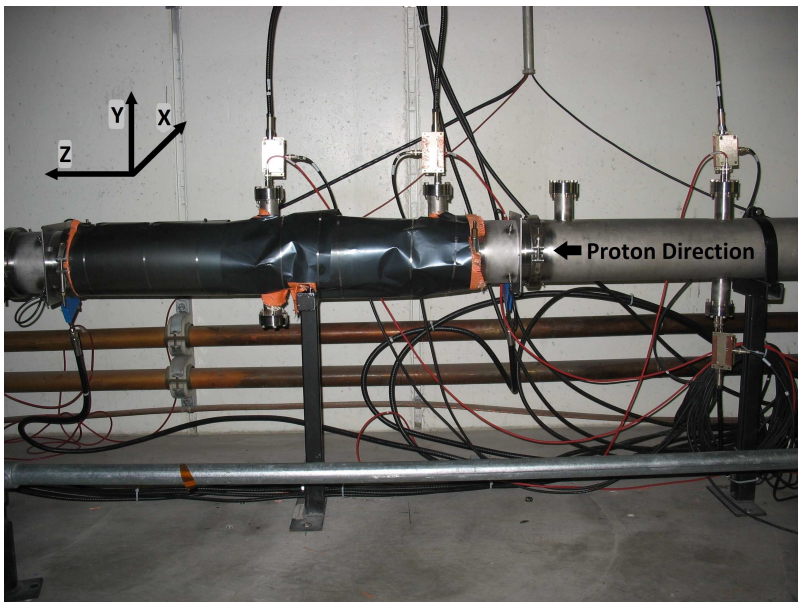
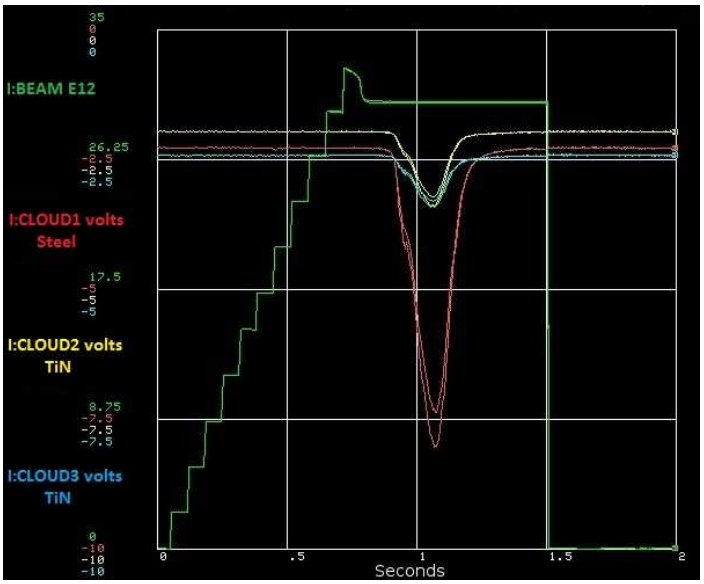
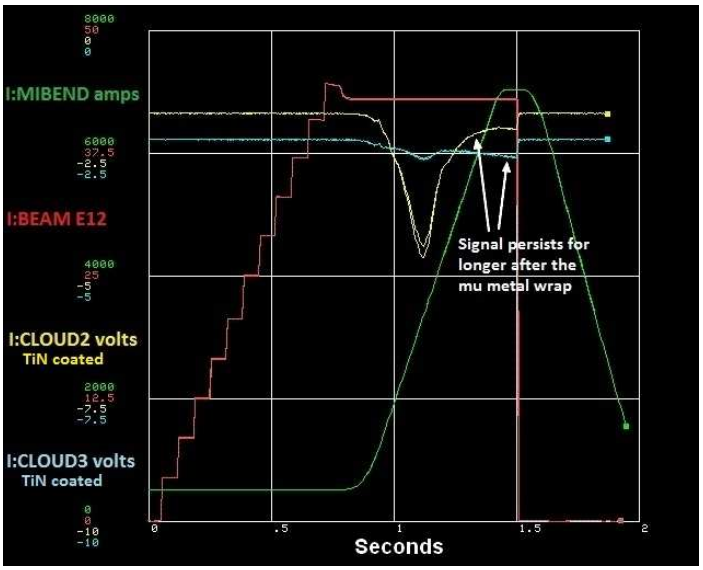


Figure 22: The TiN coated section of beam pipe was wrapped in mu metal in order to compare signals with and without magnetic shielding.



(a) Before mu metal wrap



(b) RFA2 after the mu metal wrap

Figure 23: To determine if stray magetic fields influence RFA signals, a mu metal wrap was placed around the beam pipe. The shielding changes the shape of the detector signals.

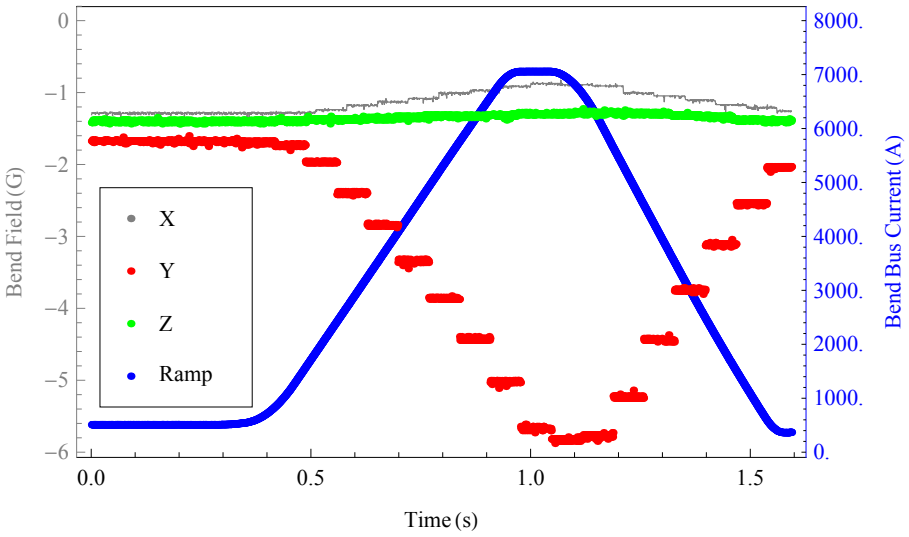


Figure 24: Stray magnetic fields reach −5.8 gauss in the vertical direction as the bend bus ramps. Stray fields are small in the other axes.

6 Measurement of Electron Energy Spectrum

The research described thus far used a grid bias of −20 V. By scanning the grid voltages from −20 V to −400 V in −20 V increments, the energy spectrum of the ecloud can be measured. Variations in beam intensities and machine conditions directly influence the ecloud buildup from cycle to cycle. Larger data samples are taken at each potential increment to reduce these variations. Data is saved at the point in time where the signals are the strongest. This data point is coupled with the proton beam intensity and the grid bias. Over a small range of beam intensities, a quadratic function can be used to empirically fit the detector data. This fit is taken for each grid voltage and then used to evaluate the function at specific beam intensities. This gives a signal strength for each grid voltage. The difference between the signal strength at the last value, (−400 V), and the signal strength at the starting value, (−20 V), represents the total signal strength. The difference between the signal strength at −20 V and −40 V represents the change in signal as a result of that change in grid voltage. Since the grid works as an energy filter that blocks out any electrons with energies less than that potential, this signal can be used to represent the number of electrons being blocked with energies between 20 eV and 40 eV. By taking this difference and dividing it by the total signal strength, a value is obtained that represents the fractional signal that is lost from changing the voltage from −20 V to −40 V on the grid. The energy spectrum of the ecloud, seen in Figure 25, is obtained by plotting the fractional signal versus the electron energy for all grid potentials between −20 V and −400 V in −20 V increments. The program used for this analysis can be found in Appendix F.4.

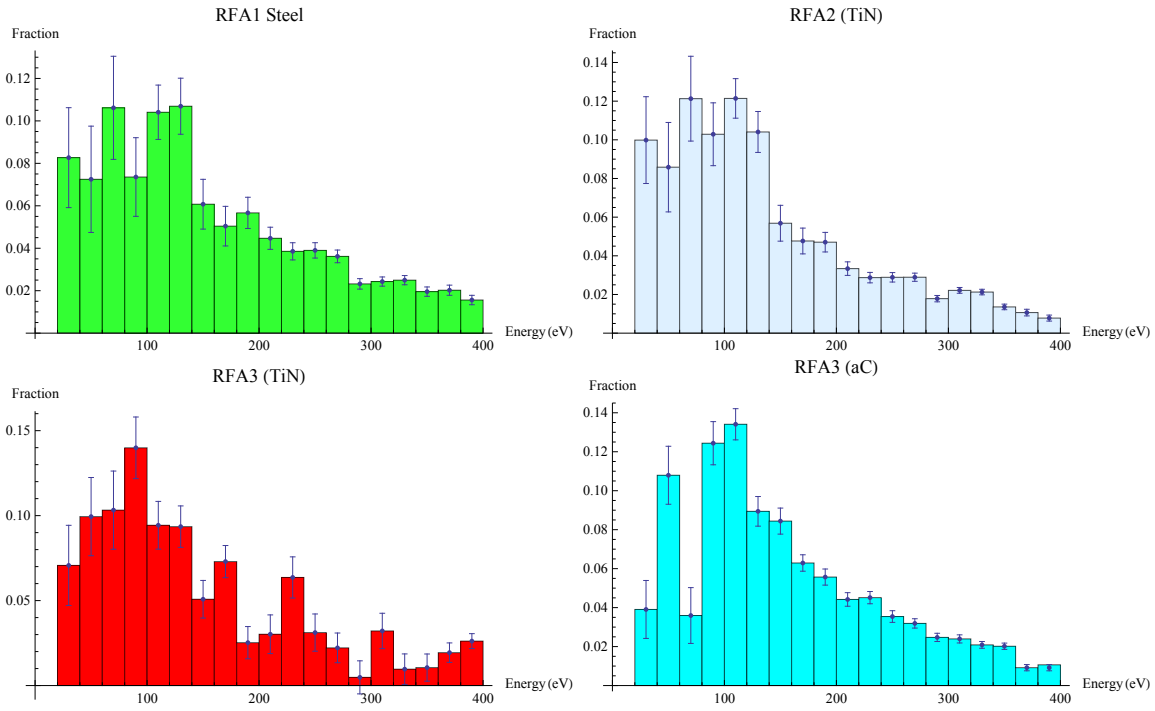


Figure 25: Fractional energy spectrum measurements are obtained for steel, TiN and aC. Both the steel and TiN measurements were evaluated with a proton intensity of 42.5×10^{12} in the Main Injector. The aC measurement was evaluated at a proton intensity of 40.00×10^{12} particles per cycle. Due to continually changing Main Injector conditions, data sets are gathered in periods of two to four hours and are displayed here with an error of one sigma. A notable peak is always present between 80 and 120 eV.

7 Conclusion

These results will be used for determining whether coatings will be needed to mitigate possible instabilities arising from eclouds in the Main Injector because of higher intensities needed for Project-X. From the results thus far, it seems that TiN and aC are comparable in their ability to mitigate ecloud. A fortuitous vacuum leak at the test location suggests that aC might not be as robust as needed. Both TiN and aC perform better than steel. Following the vacuum leak there was a period of time where the aC was much worse than the steel. However, it quickly conditioned to a better state than the steel.

8 Acknowledgement

Operated by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the United States Department of Energy.

9 References

[1] R.J. Macek, A. Browman, D. Fitzgerald, R. McCrady, F. Merrill, M M. Plum, T. Spickermann, T.S. Wang, J. Griffin, K.Y. Ng, D. Wildman, K. Harkay, R. Kustom, and R. Rosenberg. Electron proton two-stream instability at the psr. In *Proceedings of PAC01*, 2001.

[2] W. Fischer, M. Blaskiewicz, H. Huang, H. C. Hseuh, V. Ptitsyn, T. Roser, P. Thieberger, D. Trbojevic, J. Wei, S.Y. Zhang, and U. Iriso. Electron cloud observations and cures in rhic. In *Presented at the ECL2 Workshop CERN, Switzerland*, 2007.

[3] J. Wei, M. Blaskiewicz, J. Brodowski, P. Cameron, D. Davino, A. Fedotov, P. He, H. Hseuh, Y.Y. Lee, H. Ludewig, W. Meng, D. Raparia, J. Tuozzolo, S.Y. Zhang, A. Aleksandrov, S. Cousineau, V. Danilov, S. Henderson, M. Furman, M. Pivi, R. Macek, and N. Catalan-Lasheras. Electron-cloud mitigation in the spallation neutron source ring. In *Proceedings of PAC03*, 2003.

[4] C. Yin Vallgren, G. Arduini, J. Bauche, S. Calatroni, P. Chiggiato, K. Cornelis, P. Costa Pinto, B. Henrist, E. Métral, H. Neupert, G. Rumolo, E. Shaposhnikova, and M. Taborelli. Amorphous carbon coatings for the mitigation of electron cloud in the cern super proton synchrotron. *Physical Review Special Topics - Accelerators and Beams*, 14(071001), 2011.

[5] G. Rumolo, G. Arduini, V. Baglin, H. Bartosik, N. Biancacci, P. Baudrenghien, G. Bregliozzi, P. Chiggiato, S. Claudet, R. De Maria, J. Esteban-Möller, M. Favier, C. Hansen, W. Höfle, J. M. Jimenez, V. Kain, G. Lanza, K. Li, H. Maury Cuna, E. Métral, G. Papotti, T. Pieloni, F. Roncarolo, B. Salvant, E. Shaposhnikova, R. Steinhagen, L. Tavian, D. Valuch, W. Venturini Delsolaro, F. Zimmermann, U. Iriso, O. Domínguez, E. Koukovini-Platia, N. Mounet,

C. Zannini, and C. Bhat. Electron cloud observation in lhc. In *Proceedings of IPAC11*, 2011.

[6] Xiaolong Zhang, Alex Zuxing Chen, Weiren Chou, Bruce M. Hanna, King Yuen Ng, Jean-Francois Ostiguy, Linda Valerio, and Robert Miles Zwaska. Electron cloud studies at tevatron and main injector. In *Proceedings of PAC07*, 2007.

[7] R.A. Rosenberg and K.C. Harkay. A rudimentary electron energy analyzer for accelerator diagnostics. *Nucl. Instrum. Methods*, 453(3):507–513, 2000.

[8] K. Seiya, T. Berenc, B. Chase, J. Dey, I. Kourbanis, and J. Reid. Multi-batch slip stacking in the main injector at fermilab. In *Proceedings of PAC07*, 2007.

[9] C.Y. Tan, K.L. Duel, and R. Zwaska. An improved retarding field analyzer for electron cloud studies. In *Proceedings of PAC09*, 2009.

[10] Lee McCuller. Simulations and testing of retarding field analyzers for electron cloud monitoring. `http://www.illinoisacceleratorinstitute.org/2009Program/student_papers/lee_mcculler.pdf`, 2009. Presented as summer internship project.

[11] Paul L. G. Lebrun, Panagiotis Spentzouris, John R. Cary, Peter Stoltz, and Seth A. Veitzer. Accurate simulation of the electron cloud in the fermilab main injector with vorpal. In *Proceedings of IPAC10*, 2010.

[12] R. E. Kirby and F. K. King. Secondary electron emission yields from PEP-II accelerator materials. *Nucl. Instrum. Meth.*, A469:1–12, 2001.

[13] M. T. F. Pivi et al. Secondary Electron Yield and Rectangular Groove Chamber Tests in PEP II. Prepared for Particle Accelerator Conference (PAC 07), Albuquerque, New Mexico, 25-29 Jun 2007.

[14] Cristina Yin Vallgren. Low secondary electron yield carbon coatings for electron cloud mitigation in modern particle accelerators. CERN-THESIS-2011-063.

[15] S. Calatroni, P. Chiggiato, P. Costa Pinto, D. Hynds, M. Taborelli, and C. Yin Vallgren. Amorphous-carbon thin films for the mitigation of electron clouds in particle accelerators. In *Proceedings of HHH-2008*, 2008.

[16] Dave Capista. Beam measurements with and without coatings in the fermilab main injector. Presented at the Anti Ecloud Workshop at Cern in October, 2009.

[17] Bk precision 2880b instruction manual. BK Precisions Corp., 2006.

[18] Cheng-Yang Tan. High gain 3 khz lpf for rfa. Presented at Electron Cloud Working Group Meeting, 2008.

[19] In memoriam: Tom droege. Fermilab Today, Feb 18, 2008.

[20] Ametes magnetic field sensor - 3 axis datasheet. Ametes, 2011.

[21] Micromag3 3-axis magnetic sensor module datasheet. PNI Corporation, 2005.

[22] K. Seiya, T. Berenc, B. Chase, J. Dey, I. Kourbanis, and J. Reid. Progress in multi-batch slip stacking in the fermilab main injector and future plans. In *Proceedings of PAC09*, 2009.

[23] C. Gattuso, A. Braun, D. Morris, R. Spayde, B. Worthel, B. Evanger, and P. Karns. Accelerator concepts. `http://www-bdnew.fnal.gov/operations/rookie_books/Concepts_v3.6.pdf`, 2010. Fermilab Operations Rookie Book.

A Multi-Batch Slip Stacking

The Main Injector RF system runs at 52.8 MHz at injection with a harmonic number of 588. This means that at any given time there are 588 stable regions, or buckets, where beam can exist. The Main Injector is loaded with 8 GeV beam from the Booster which is extracted from the Booster at 52.8 MHz. The Booster has a harmonic number of 84. When fully loaded, only 80 of the Booster's 84 buckets contain beam as space is required for the extraction kicker magnetic field to rise. These 80 bunches of beam makeup up one Booster batch of beam. 588 divided by 84 is 7, therefore 7 batches of 84 bunch booster beam are required to fill all of the Main Injector's 588 buckets. As with the Booster, part of the Main Injector must remain empty to allow time for the kicker magnetic field to rise, therefore there is only room for 6 Booster batches in 1 Main Injector cycle [23].

The Main Injector has 18 RF cavities and amplifier stations that are used to accelerate beam. The RF system can be used as two independant systems with different frequencies. Using 3 stations, a booster batch is injected and captured on the central frequency of the first RF system. This batch is then decelerated to follow a different path in the machine. Another batch can then be injected and captured on the central frequency of the second RF system which consists of 3 different RF stations. Since the two batches have different frequencies and energies they follow different orbits in the machine and slip passed each other without interfering. Each RF system frequency is then moved to the same final central frequency. By turning on the rest of the RF when the different bunches line up longitudinally, the independant bunches can be captured in single RF buckets [8].

Eleven batch multi-batch slipstacking is described in Figure 26. Eleven batch mode starts by injecting 5 booster batches into the Main Injector using the first RF system. These 5 batches are then decelerated. Next, 5 more batches are injected onto

the central frequency of the second RF system. These follow a different orbit than the first and do not interfere with the first. The two independant RF system frequencies are then brought together, and the 12 remaining RF stations are brought up to full gradient at the central frequency thereby capturing 10 batches in the buckets for 5. A final non-slip stacked batch is then injected to make 11 Booster batches in the Main Injector [22].

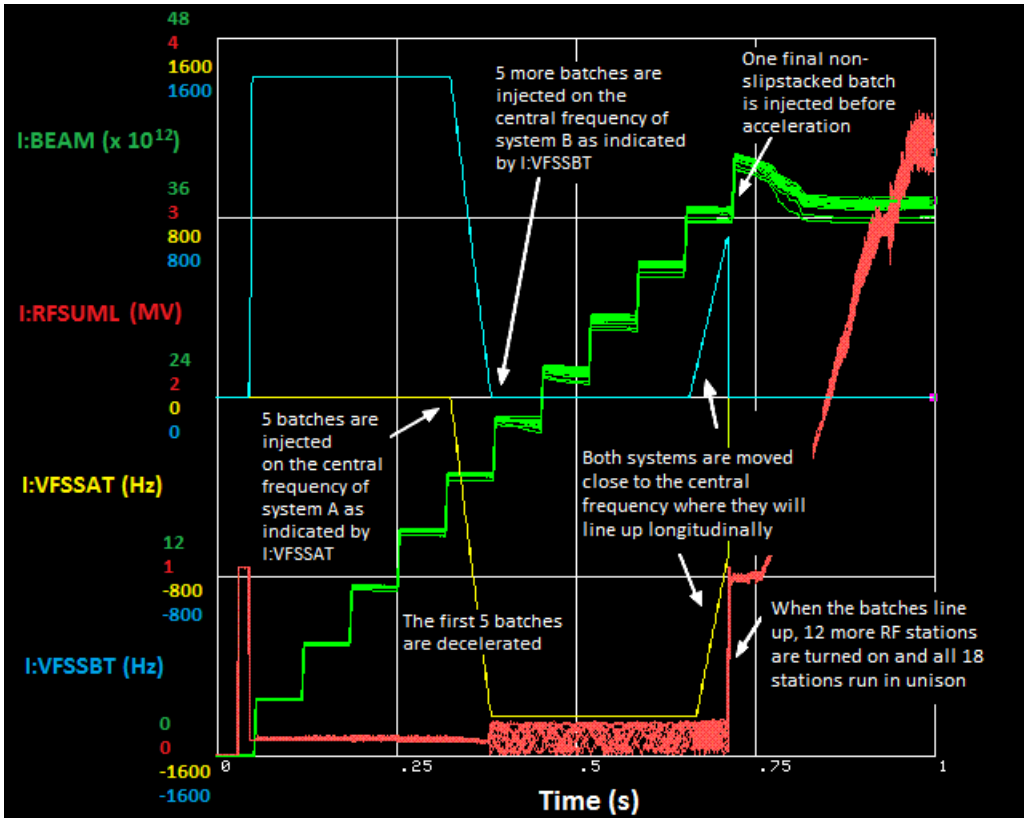


Figure 26: The green trace above is the Main Injector beam intensity. Each step is representative of one batch of beam injected from the Booster. The blue and yellow traces indicate the frequency at which each independant RF system is operating. The red trace is the sum of all of the RF station voltages. The discreet increase in this trace just before the .75 s mark is indicative of the point when all of the RF is turned on to capture two batches in one set of buckets. Eleven batch multi-batch slip stacking is utilized to increase Main Injector Intensities by a factor of 1.5 [8].

B Bunch Spacing

The Main Injector's primary mode of running during this study was in mixed mode as described in the Main Injector section above. In this mode beam destined for both Numi and Pbar production was loaded into the Main Injector in the same cycle. This mode required two different kicker gaps, or empty buckets that leave space for the kicker rise time. During times in which Pbar was not able to take beam or did not need as much beam, the machines were ran in a mode with Numi alone, hereafter Numi-only. In this mode, all of the beam was destined for one location, therefore there was only one kicker gap. The bunch structures as measured by a Sampled Bunch Display are illustrated in Figure 27. It was found that the bunch configuration caused significant changes in the ecloud formation; particularly, the extra kicker gap during mixed mode damped the ecloud formation. Figure 28 illustrates a time period in which both mixed mode and Numi-only mode were being alternated every other pulse. Since Numi-only mode does not include the beam destined for Pbar production, the Numi only signal consists of data with lower intensity but larger ecloud signals. This proves that bunch arrangement in the Main Injector can be used to dampen the ecloud formation. Since Numi-only was not the primary mode used for this study, only the Mixed mode signals were analyzed. These studies of bunch structure show that there are potential improvements to be made without the requirements of additional coatings. These methods will require more analysis, but were outside of the approved studies for this era of running.

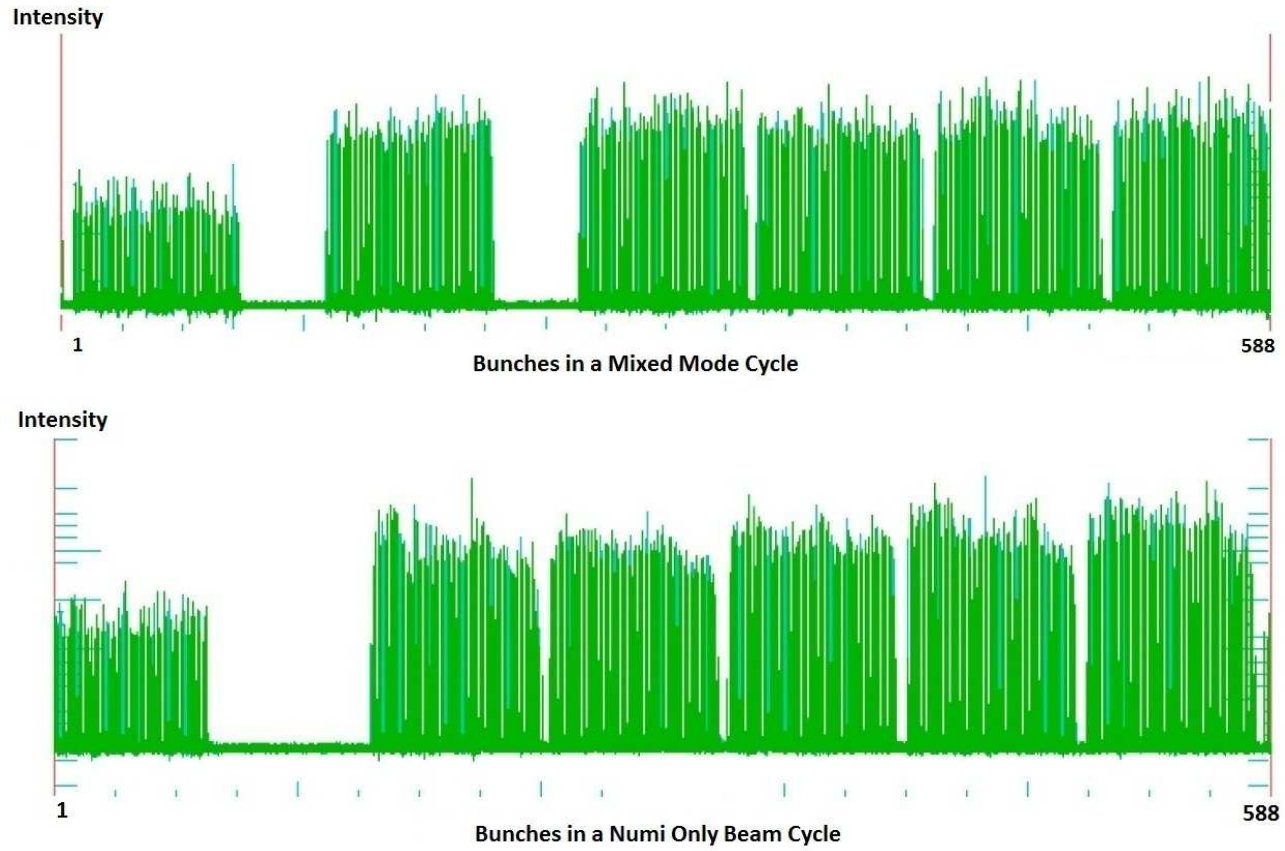


Figure 27: The top plot illustrates all of the bunches in the Main Injector during a mixed mode cycle. There are two kicker gaps present in this plot versus the lower one which has one larger kicker gap. This figure was provided by Dave Capista and Kiyomi Seiya [16].

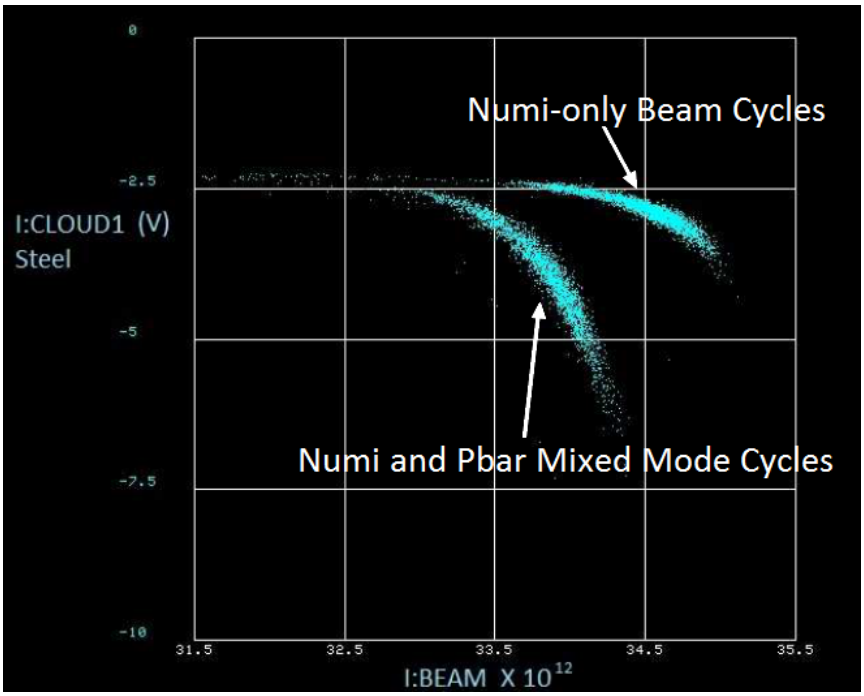


Figure 28: Different bunch structure within the Main Injector causes significant changes in ecloud formation. The higher intensity group of Numi-only data above has less ecloud signal, while the lower intensity group of mixed mode data has stronger signals. The bunch arrangements used to create these signals can be seen in Figure 27.

C Preamp and Filter Characterization

A 3 KHz 8 pole Butterworth Filter is used in combination with a 40 dB radiation hardened amplifier to transmit the signal from the collector into the Fermilab control system. Figure 29 shows both the high gain filter and schematic. Since any object to be used in a particle accelerator must be able to withstand the radiation levels encountered, a radiation hardened op-amp was used to increase the life expectancy of the preamp. Both time and frequency domain measurements were taken to characterize the high gain filters before taking data, after the first Titanium Nitride run and after the final amorphous Carbon run. Figure 30 shows the initial and final measurements for the preamp used in RFA1. Figure 31 shows the measurements for RFA2, and Figure 32 shows those for RFA3. These figures show that the preamp behavior did not change throughout the time frame documented in this thesis.

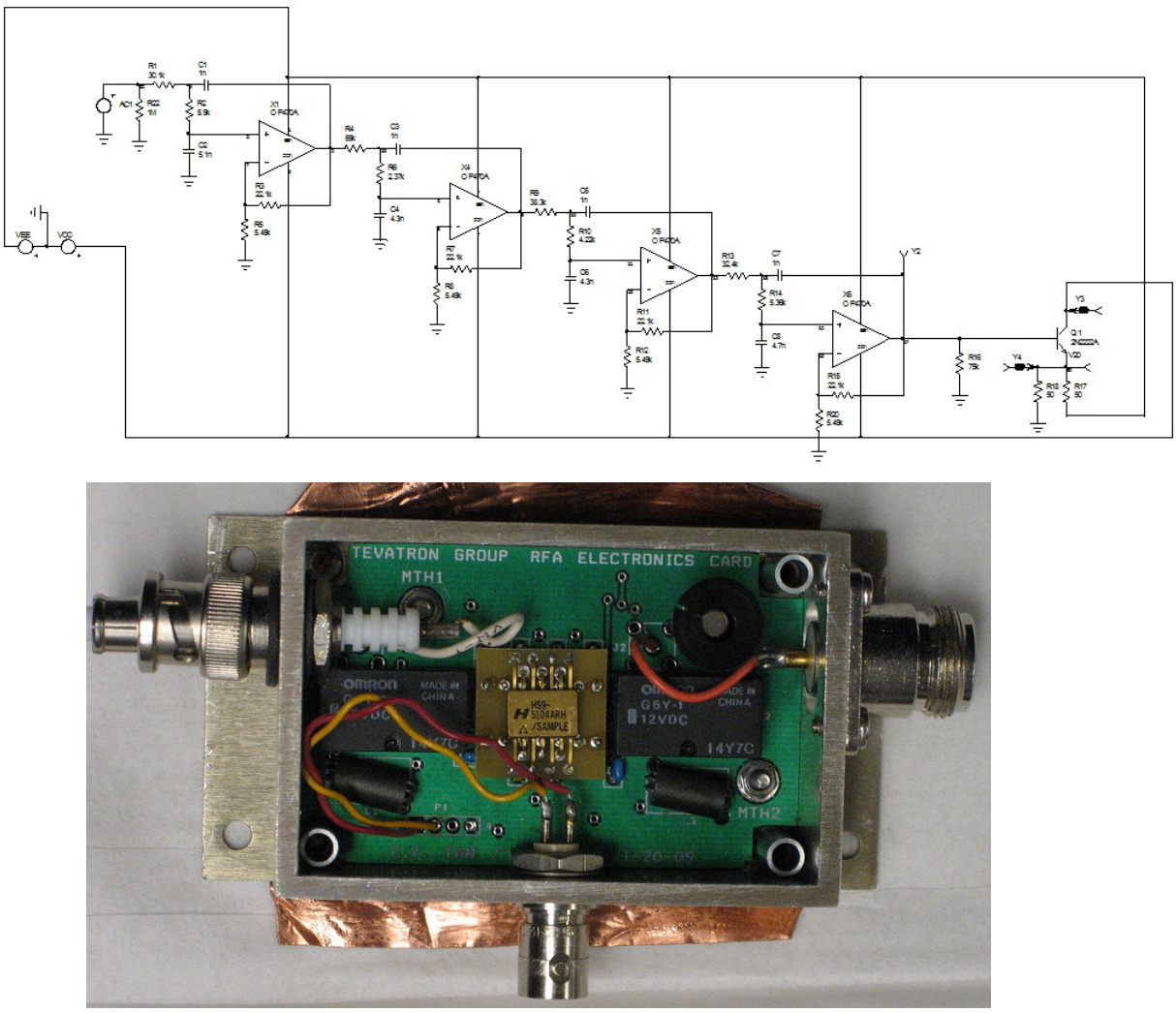
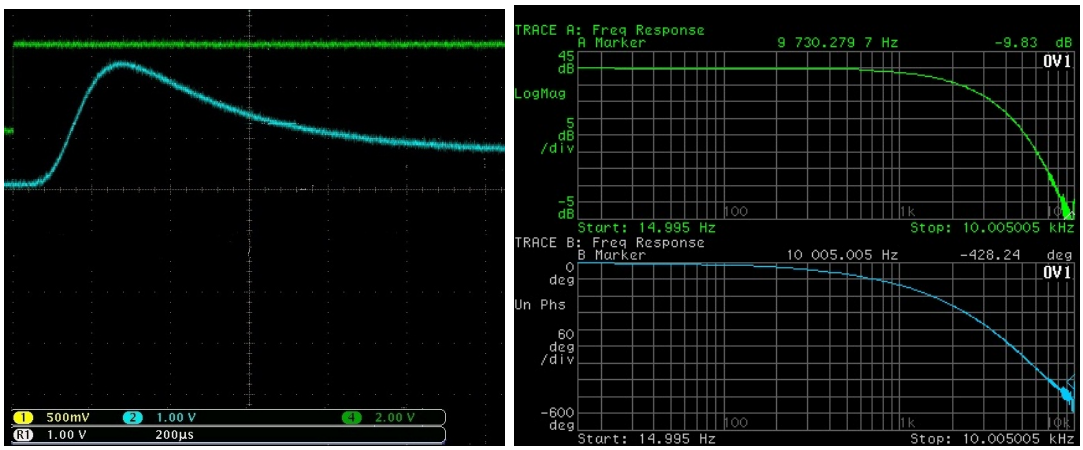
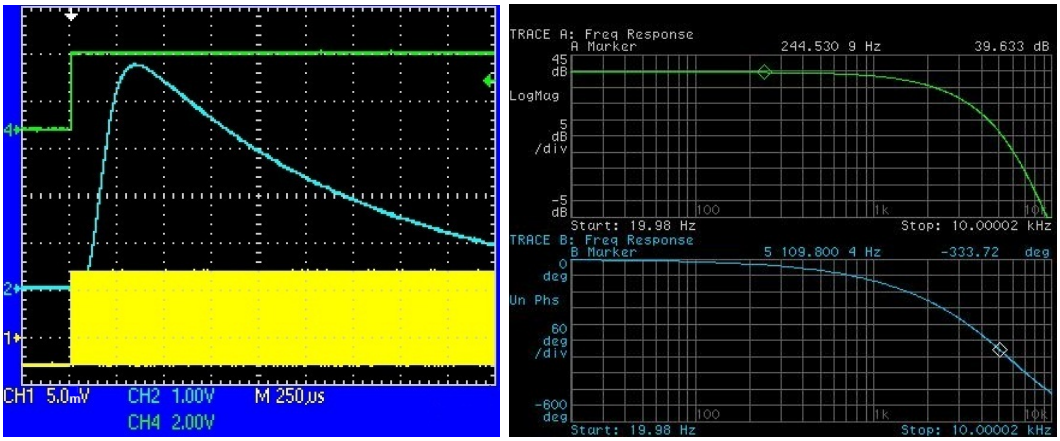


Figure 29: A 3 KHz 8 pole Butterworth filter is used in combination with a 40 dB radiation hardened amplifier. Top photo courtesy of C.Y. Tan [18].

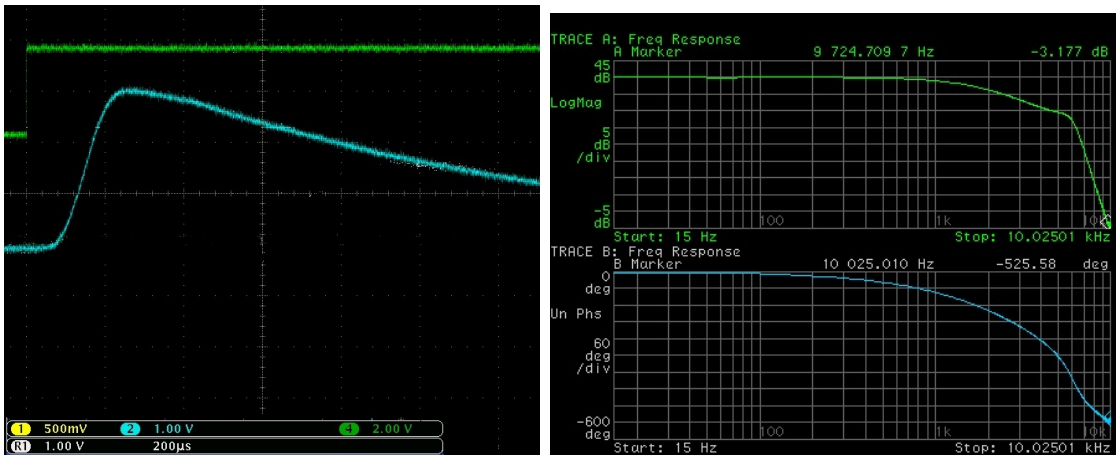


(a) Initial RFA1 Preamp Characterization

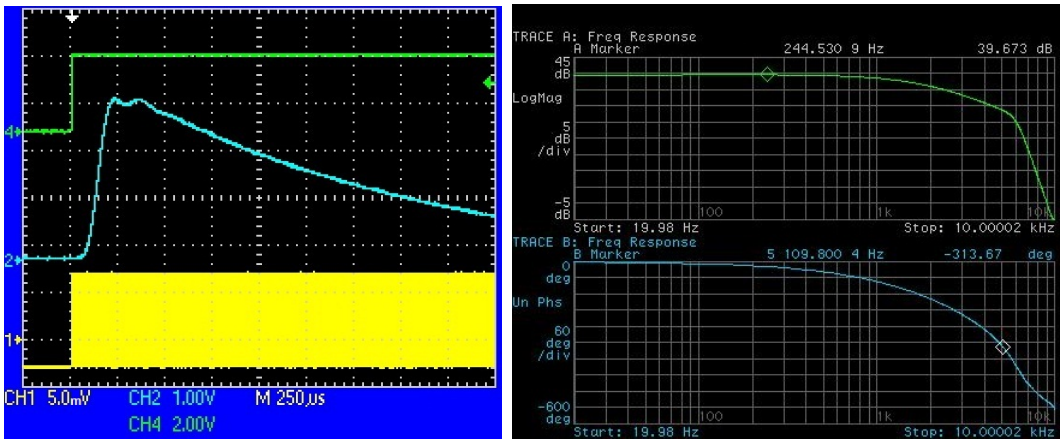


(b) Final RFA1 Preamp Characterization

Figure 30: Shown above are the initial and final measurements for RFA1. The time domain traces on the left were taken by sending a burst of 80 KHz square waves with an amplitude of 25 mV into the grid of the RFA. This was then detected in the collector of the RFA through the preamp and filter. It should be noted that the initial measurements were taken on a test bench with a short cable run. The final measurements were taken from the service building through a longer cable run. The green trace is the burst trigger. The blue trace is the signal filtered through the preamp. The yellow trace in the bottom plot shows the 80 KHz signal. The overshoot in the time domain trace is a result of the added inductance, capacitance and resistance from the long cable. The plots on the right show the initial and final frequency spectrum measurements both taken on a test bench. Both initial and final measurements look consistent, which shows that there was no damage to the amplifiers.

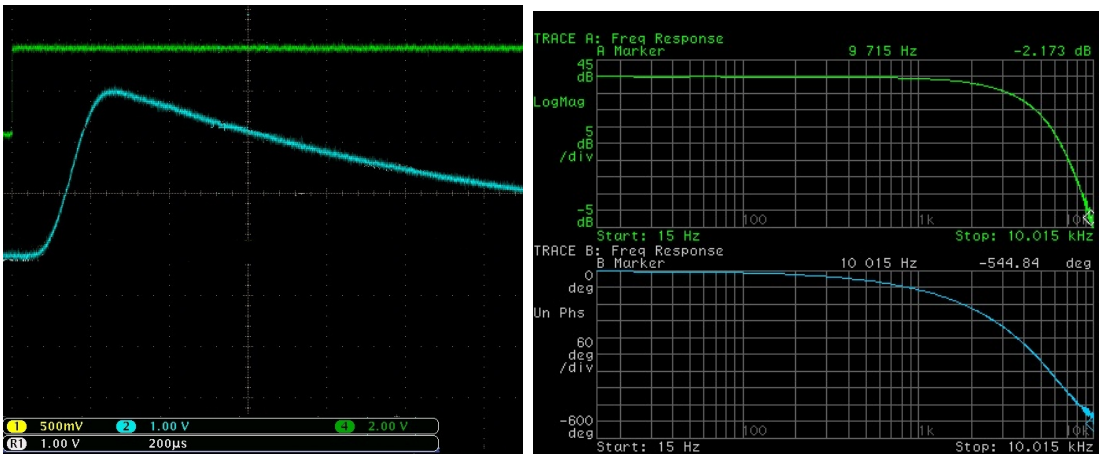


(a) Initial RFA2 Preamp Characterization

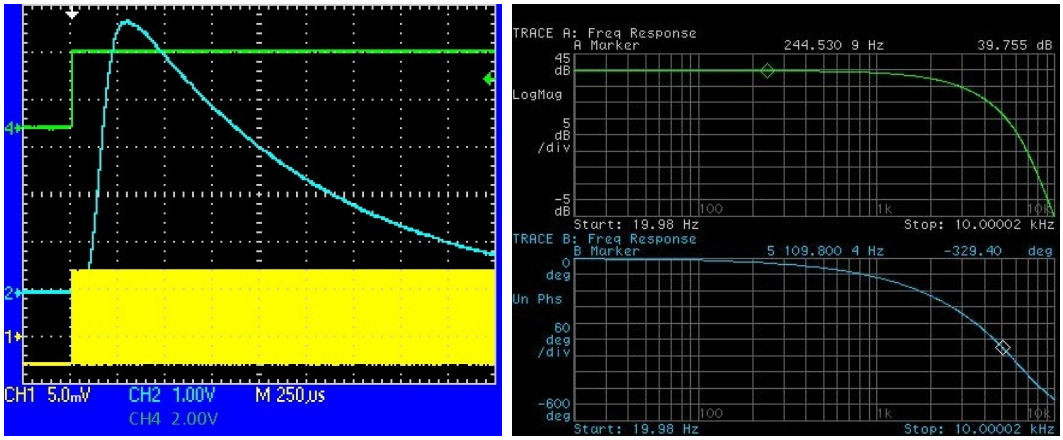


(b) Final RFA2 Preamp Characterization

Figure 31: Shown above are the initial and final measurements for RFA2. The time domain traces on the left were taken by sending a burst of 80 KHz square waves with an amplitude of 25 mV into the grid of the RFA. This was then detected in the collector of the RFA through the preamp and filter. It should be noted that the initial measurements were taken on a test bench with a short cable run. The final measurements were taken from the service building through a longer cable run. The green trace is the burst trigger. The blue trace is the signal filtered through the preamp. The yellow trace in the bottom plot shows the 80 KHz signal. The overshoot in the time domain trace is a result of the added inductance, capacitance and resistance from the long cable. The plots on the right show the initial and final frequency spectrum measurements both taken on a test bench. Both initial and final measurements look consistent, which shows that there was no damage to the amplifiers.



(a) Initial RFA3 Preamp Characterization



(b) Final RFA2 Preamp Characterization

Figure 32: Shown above are the initial and final measurements for RFA3. The time domain traces on the left were taken by sending a burst of 80 KHz square waves with an amplitude of 25 mV into the grid of the RFA. This was then detected in the collector of the RFA through the preamp and filter. It should be noted that the initial measurements were taken on a test bench with a short cable run. The final measurements were taken from the service building through a longer cable run. The green trace is the burst trigger. The blue trace is the signal filtered through the preamp. The yellow trace in the bottom plot shows the 80 KHz signal. The overshoot in the time domain trace is a result of the added inductance, capacitance and resistance from the long cable. The plots on the right show the initial and final frequency spectrum measurements both taken on a test bench. Both initial and final measurements look consistent, which shows that there was no damage to the amplifiers.

D Grid Power Supply

In order to effectively use the RFAs a DC bias must be applied to each of the retarding grids. To obtain this bias, one power supply runs in parallel to all four RFAs. The power supply is a "Droege" power supply, named after its designer, Tom Droege [19]. The power supply used can be seen in Figure 33, and has been altered to make its output range 0-2500 Volts versus the normal Droege output of 0-10,000 Volts. This power supply is a switching power supply and was chosen because of its remote control capabilities. It can be controlled by a basic unipolar Camac card and its output can be monitored through an MADC. A B&K Precision 2880B digital volt meter was used to calibrate the power supply. The voltmeter is accurate to within .2 V at 500 V [17]. The initial measurement and those taken after one year confirmed the power supply read-backs in our controls system to be within 1 V at 150 V and 3 V at 500 V. The beam pipe conditioning comparisons were all performed with −20 V on the grid. The energy scans performed to measure the energy spectrums of the ecloud are done in −20 V increments to −500 V. An error of just over 1 V to −150 V and just over 3 V at −500 V is quite acceptable. It was found that with voltages greater than −500 V the the gap between the grid and the detector started to break down, and the signal was negatively impacted. This limits the retarding field to grid potentials with a magnitude less than 500 V .

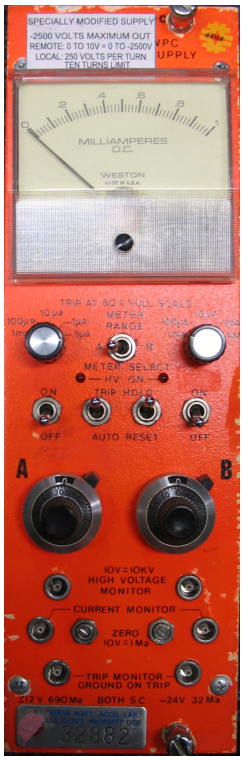


Figure 33: Droegie Power Supply

E Magnetic Probe Calibration

Three different magnetic probes were built and calibrated to measure the magnetic field at multiple locations near the retarding field analyzers. The Helmholtz coils shown in Figure 34 were used to provide a uniform magnetic field for calibration. This field was measured using a Lakeshore model 460 three dimensional Gaussmeter shown in Figure 34 and each of the three probes for signal characterization.

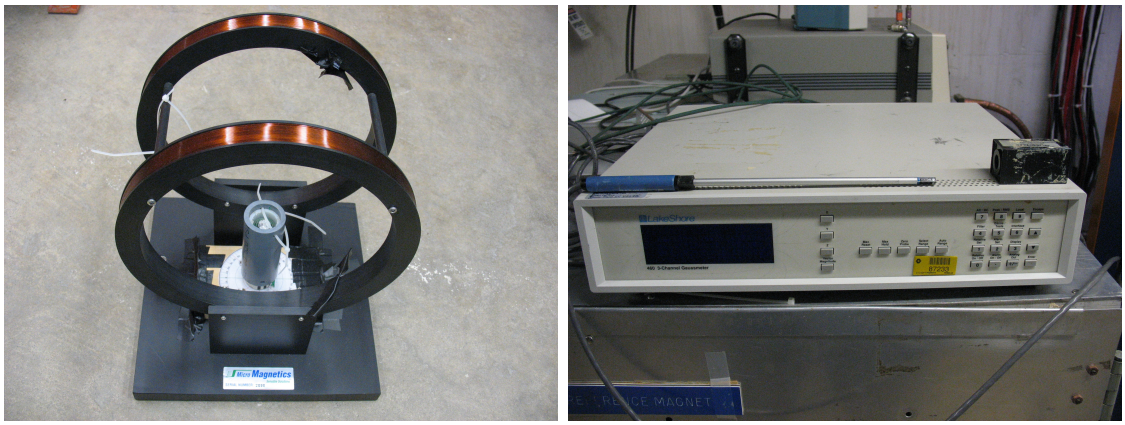


Figure 34: A current regulated power supply powering these coils provided the magnetic field which was used with this Lakeshore Gauss probe to determine the proper scale factors to convert the new probe signals from Volts to Gauss.

E.1 Three Dimensional Analog Hall Probe

The first magnetic field measurements in the tunnel were taken with two hall probes that were designed by C.Y. Tan and assembled at Fermilab. Each probe used a three axis MFS-3A magnetic field sensor built by Ametes shown in Figure 35 [20]. Each was calibrated using the Helmholtz coils and the Lakeshore three dimensional Gaussmeter. These probes send the net field measurement added in quadrature to an analog to digital converter. There were not enough cables running from the tunnel

to the service buildings to measure each axis individually. Figure 36 shows the data points measured on the existing hall probe relating the net signal in Volts to the measured field in Gauss. This data is shown with a linear fit along with linear fits of the upper and lower 95% confidence intervals. Equation (3) is used to convert the signal from Probe A from Gauss to Volts while Equation (4) is used for Probe B.

$$B(x)[G] = 10.93 \times x[V] - .04 \tag{3}$$

$$B(x)[G] = 12.85 \times x[V] - .04 \tag{4}$$

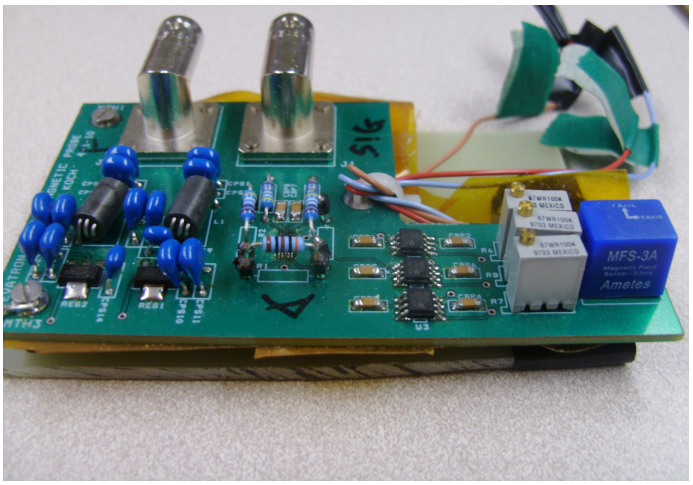


Figure 35: Probe A

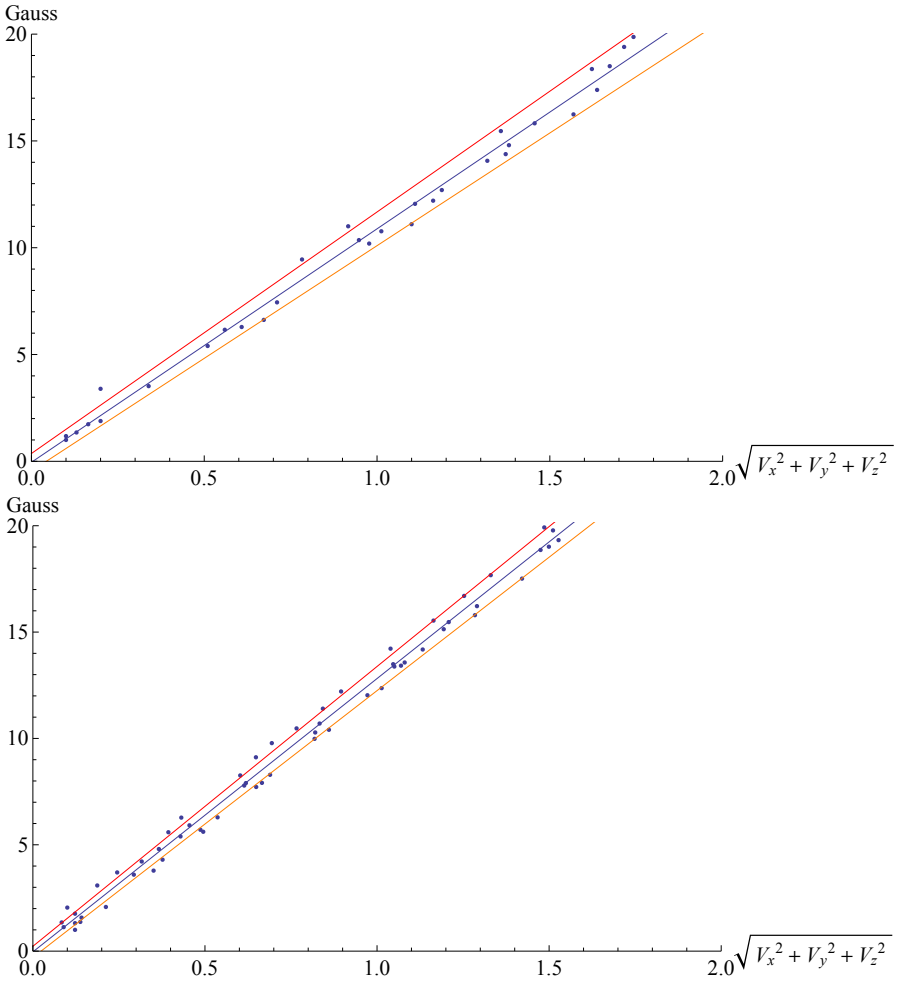


Figure 36: The linear equations shown in Equation (3) and Equation (4) are derived from experimental comparisons and are used to convert the voltage readbacks from the probes to a net magnetic field in Gauss. The above plots show measured data points with the linear fit and the upper and lower 95% confidence intervals.

E.2 Three Dimensional Hall Probe With Digital Selection of Axes

The initial net magnetic field measurements were high enough to warrant more research. As a result the a three axis system was built and calibrated using a MicroMag 3-Axis Magnetic Sensor Module [21]. This module is shown in Figure 37. Each axis of this module was calibrated using Helmholtz coils and a Lakshore three dimensional Gauss meter. This module enabled the digital selection of axis, thus allowing directional measurement of each magnetic field. Figures 38, 39 and 40 show measured data with fitted functions that relate the measured signal in Volts to the magnetic field in Gauss as measured by the Lakeshore Gauss probe. In this method the signals were first fitted as a functions of the magnetic fields. This method was used because the fitting algorithms worked better in Mathematica. The equations found were then solved for the magnetic fields as functions of the signals detected in all three axes of the new probe. The equations are shown in Equations (5) through (10).

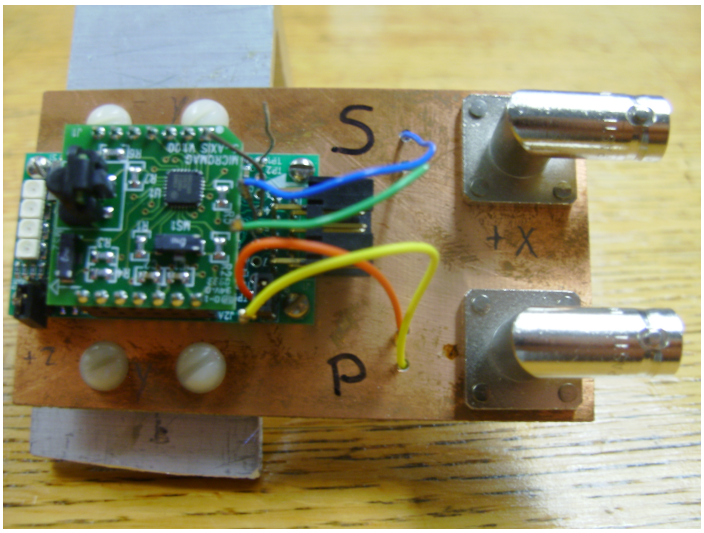


Figure 37: The MicroMag3 3-axis Magnetic Sensor Module provided both individual axis and net magnetic field data.

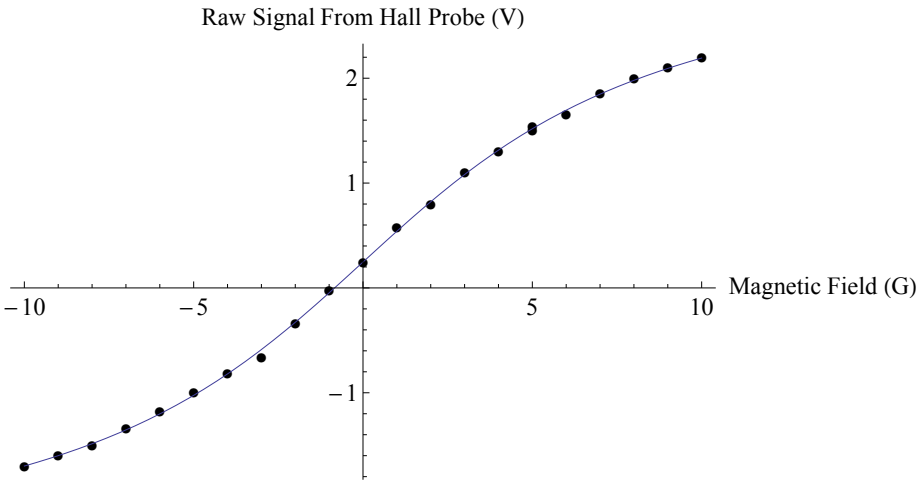


Figure 38: The black dots are experimental data with the Magnetic field measured by the Lakshore Gauss probe in the X axis of the plot and the signal in Volts from the X axis of the digitally controlled Gauss probe on the Y axis of the plot. Equation (5) is a function fit to this data. Equation (6) is the solution of Equation (5) for the dependant variable and is used for converting the voltage readback from the tunnel to the magnetic field in Gauss. This method of first fitting the signal as a function of the magnetic field was used because the fitting algorithms worked better in Mathematica.

$$S(B_X)[V] = 1.99 \times \arctan\left[.15 \times B_X[G]\right] + .25 \tag{5}$$

$$B_X(S)[G] = -6.71 \times \tan\left[.12 - .50 \times S[V]\right] \tag{6}$$

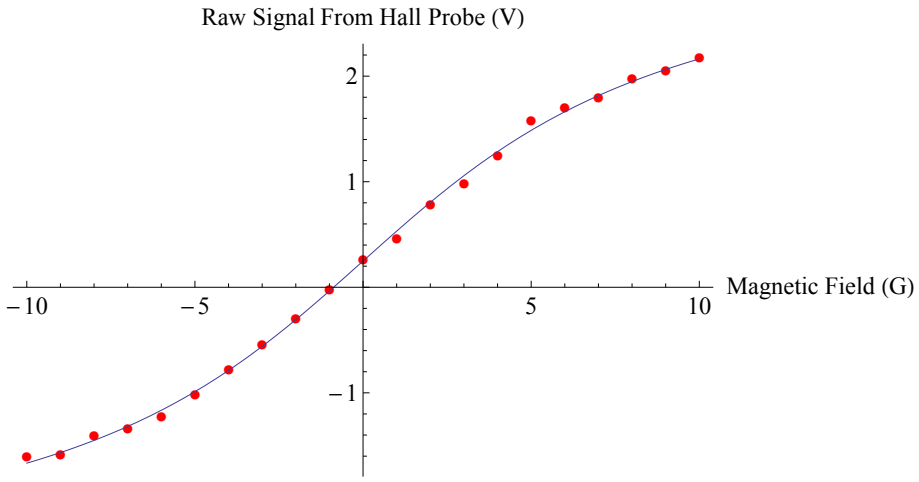


Figure 39: The red dots are experimental data with the Magnetic field measured by the Lakshore Gauss probe in the X axis of the plot and the signal in Volts from the Y axis of the digitally controlled Gauss probe on the Y axis of the plot. Equation (7) is a function fit to this data. Equation (8) is the solution of Equation (7) for the dependant variable and is used for converting the voltage readback from the tunnel to the magnetic field in Gauss.

$$S(B_Y)[V] = 1.99 \times \arctan\left[.14 \times B_Y[G]\right] + .25 \tag{7}$$

$$B_Y(S)[G] = -6.98 \times \tan\left[.13 - .50 \times S[V]\right] \tag{8}$$

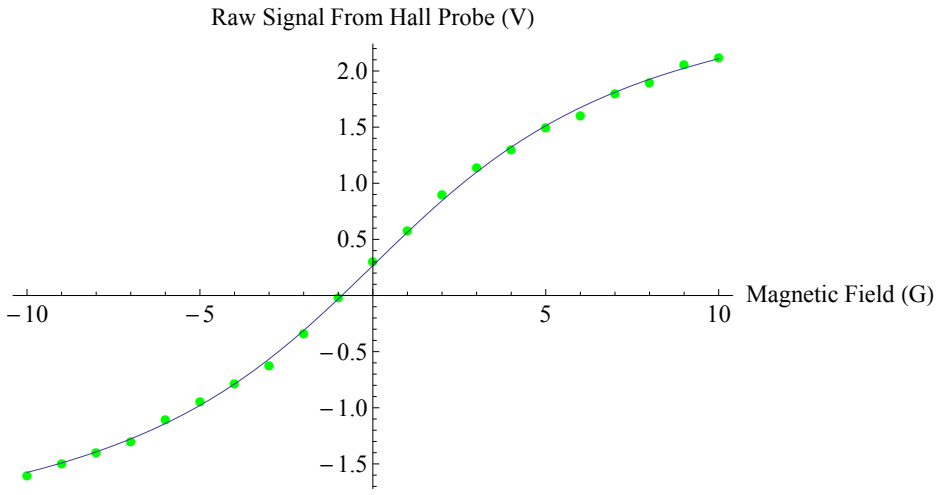


Figure 40: The green dots are experimental data with the Magnetic field measured by the Lakshore Gauss probe in the X axis of the plot and the signal in Volts from the Y axis of the digitally controlled Gauss probe on the Y axis of the plot. Equation (7) is a function fit to this data. Equation (8) is the solution of Equation (7) for the dependant variable and is used for converting the voltage readback from the tunnel to the magnetic field in Gauss.

$$S(B_Z)[V] = 1.78 \times \arctan\left[.17 \times B_Z[G]\right] + .27 \tag{9}$$

$$B_Z(S)[G] = -5.94 \times \tan\left[.15 - .56 \times S[V]\right] \tag{10}$$

F Mathematica Programs

F.1 Program for Fitting Daily Intensity Dependant Data

Daily Fit Parameters

In[512]:=
```
day = 6;

dir = Directory[] <> "/Ecloud/Ecloud Carbon Coated/D44/nov " <> ToString[day] <> "/";
i = 1;

table1 = ReadList[dir <> "nov " <> ToString[day] <> ".txt",
   {Number, Number, Number, Number, Number, Number, Number, Number, Number, Number}];
table2 = Table[{0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {i, Length[table1] - 1}];

VList = { 20, 40, 60, 80, 100, 120, 140, 160,
   180, 200, 220, 240, 260, 280, 300, 320, 340, 360, 380, 400};

parameters = Table[{0, 0, 0, 0}, {i, 32}];

(*Here we just create and 0 some of the tables to be used later*)
p = 1;
rfa1 = Table[{0, 0}, {p, Length[VList]}];
rfa2 = Table[{0, 0}, {p, Length[VList]}];
rfa3 = Table[{0, 0}, {p, Length[VList]}];
rfa4 = Table[{0, 0}, {p, Length[VList]}];
ClearAll[a, b, x, s, z, a1, b1, y, q];

ClearAll[x, lm2c, f, a, b, lm2a, lm2b, lm2d];
lower = .1;
 upper = 52;
 value = 25;
(*This s is what I will use to trim the function....I
 can use a different s to fit the function*)
s = 2.1;
os = 2;
os1 = .01;
Δos = ((os - os1) / Length[VList]);
t = 1;

Do[{
  min = -2 + -VList[[t]];
  max = 2 + -VList[[t]];
  j = 1;
  i = 1;
  For[i = 1, i ≤ Length[table1], i++,
   If[table1[[i, 10]] > min && table1[[i, 10]] < max &&
       table1[[i, 1]] > lower && table1[[i, 1]] < upper,
     table2[[j, 1]] = table1[[i, 1]];
     table2[[j, 2]] = table1[[i, 2]];
     table2[[j, 3]] = table1[[i, 3]];
     table2[[j, 4]] = table1[[i, 4]];
```



```
     table2[[j, 5]] = table1[[i, 5]];
     table2[[j, 6]] = table1[[i, 6]];
     table2[[j, 7]] = table1[[i, 7]];
     table2[[j, 8]] = table1[[i, 8]];
     table2[[j, 9]] = table1[[i, 9]];
     table2[[j, 10]] = table1[[i, 10]];
     j++
    ];
],

j = 1;
For [i = 1, i ≤ Length[table1], i++,
    If[table2[[j, 1]] ≠ 0,
        j++;
       ];
    ];
k = 1;
cloud1 = Table[{0, 0}, {k, j - 1}];
cloud2 = Table[{0, 0}, {k, j - 1}];
cloud3 = Table[{0, 0}, {k, j - 1}];
cloud4 = Table[{0, 0}, {k, j - 1}];

For[k = 1, k < j, k++,
    cloud1[[k]] = {table2[[k, 1]], table2[[k, 2]]};
    cloud2[[k]] = {table2[[k, 3]], table2[[k, 4]]};
    cloud3[[k]] = {table2[[k, 5]], table2[[k, 6]]};
    cloud4[[k]] = {table2[[k, 7]], table2[[k, 8]]};
    ];

Sort[cloud1, #1[[1]] < #2[[1]] &];
Sort[cloud2, #1[[1]] < #2[[1]] &];
Sort[cloud3, #1[[1]] < #2[[1]] &];
Sort[cloud4, #1[[1]] < #2[[1]] &];

(*Here FindFit is used to do an initial fit to the unfiltered
 data and make that function equal to lm2a, lm2b,
 lm2c and lm2d where these correspond to RFA1, 2 ,3, and 4*)
f = FindFit[cloud1, a + (-b / (10^30)) * s^(x), {a, b}, x];
a1 = a /. f;
b1 = b /. f;
lm2a[x_] := a1 + (-b1 / (10^30)) * s^(x);

z = 1;
cloud1a = Table[{0, 0}, {z, Length[cloud1] - 1}];
cloud2a = Table[{0, 0}, {z, Length[cloud2] - 1}];
cloud3a = Table[{0, 0}, {z, Length[cloud3] - 1}];
cloud4a = Table[{0, 0}, {z, Length[cloud4] - 1}];

(*Next any data that is outside of some offset of the initial fit is discarded.
   Since lm2 is a quadratic function fit to the data one can just evaluate it
  at some x value. The first term is used in each of the data points in the
```



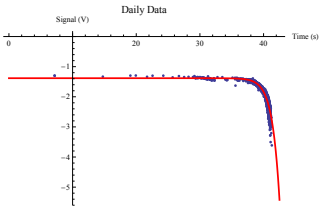
```
   cloud data.  When the fuction is evaluated for its x term,
  the Y term will either be greater than or less than the function plus an
   offset.  If it is less than then it is kept placed in the new table cloud1a.

    "&& cloud1[[i,2]]>-10" was added to
   get rid of any overdriven signals that are not real data*)

  i = 1;
  j = 1;
  For[i = 1, i < Length[cloud1], i++,
   If[(lm2a[cloud1[[i, 1]]] +os ) > cloud1[[i, 2 ]] &&
       cloud1[[i, 2]] > -10 && cloud1[[i, 1]] > 1,
     cloud1a[[j]] = cloud1[[i]];
     j++;
      ];
    ];
  (*Iterate through the new table until
   {0,0} value is seen which which signals the end of the data*)

  l = 1;
  i = 1;
  For [i = 1, i < Length[cloud1], i++,
      If[cloud1a[[l, 1]] ≠ 0,
          l++;
         ];
       ];


  (*
  The new cloud1a, cloud2a, cloud3a, and cloud4a tables are the same length as the
   original cloud1 tables and thus there are some 0 values for unpopulated
   parts of the table at the end.  Get rid of these by creating a new table
   using the iterators from the For loops above and populating the cloud1a,
  2a, 3a, 4a data into new tables called cloud1b, 2b, 3b and 4b.
  *)

  n = 1;

  cloud1b = Table[{0, 0}, {n, l - 1}];
  n = 1;
  For[n = 1, n < l, n++,
    cloud1b[[n]] = {cloud1a[[n, 1]], cloud1a[[n, 2]]};
     ];

  os = os - Δos;
  table2 = Table[{0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {i, Length[table1] - 1}];},
 {t, 1, 1, 1}
]
```


In[612]:=
```
data = ListPlot[cloud1b, PlotRange → All,
    PlotLabel → "Daily Data", AxesLabel → {"Time (s)", "Signal (V)"}];
Clear[a, b, x, x0, z];
zos = -1.3829726059113925`;
f = FindFit[cloud1b, zos - e^(a*(x-x0)), {a, x0}, x]
fit = Plot[zos - e^(a*(x-x0)) /. f, {x, 0, 42.5}, PlotRange → All, PlotStyle → {Red, Thick}];
(*zos is the offset found from data with no Ecloud signal*)
zos
Show[data, fit]
(*Write the fit parameters for that day to array "parameters"*)
parameters[[day, 1]] = day;
parameters[[day, 3]] = zos;
parameters[[day, 2]] = a /. f;
parameters[[day, 4]] = x0 /. f;
```

Out[615]= {a → 0.874931, x0 → 40.9042}

Out[617]= -1.38297

Out[618]=



F.2 Program for Fitting a Lorentzian Function to Signal

Fit a Lorentzian Function to the Beam Signal and Correlate it With Beam Intensity

```
dir = Directory[] <> "/Ecloud/Snapshot Data/27apr2010 $8E/";
beam = ReadList[dir <> "I_BEAM.txt", {Number, Number}];
cloud1 = ReadList[dir <> "I_CLOUD1.txt", {Number, Number}];
```

Find the average beam intensity.
Iterate through the I:BEAM signal to determine how many datapoints are taken between 1 and 1.1 seconds. This determines the length of the next table.

```
j = 1;
For[i = 1, i ≤ Length[beam], i++,
     If[beam[[i, 1]] > .6 && beam[[i, 1]] < 1,
       j++,
      ]];
beamint = Table[{0, 0}, {j - 1}];
(*A new table is made consisting of
 I:BEAM values from 1 to 1.1 to be used for finding the average beam intensity*)
j = 1;
For[i = 1, i ≤ Length[beam], i++,
     If[beam[[i, 1]] > .6 && beam[[i, 1]] < 1,
            beamint[[j]] = beam[[i]];
           j++;

      ]];
(*Next find the average, but since the Mean function will give the average of both
 terms in the list a new variable is made to assign the second list term to*)
avg = Mean[beamint];
avgbeam = avg[[2]]
```

```
44.8709
```

Now find a Lorentzian fit to the detector signal

```
table1 = Table[{0, 0}, {i, Length[cloud1] - 1}];
For[i = 1, i ≤ Length[cloud1] - 1, i++,
 table1[[i, 2]] = cloud1[[i, 2]];
 table1[[i, 1]] = cloud1[[i, 1]];
   ]
```



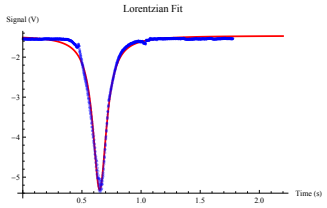
```
p1 = ListPlot[table1, PlotRange → All, PlotStyle → RGBColor[0, 0, 1, .3]];
table1[[1]]
```

```
{0.00087, -1.54906}
```

```
Clear[a, x0, λ, z]
fit = FindFit[table1, z - a / ((x - x0)^2 + λ), {a, x0, λ, z}, x]
```

```
{a → 0.0189071, x0 → 0.650201, λ → 0.00490512, z → -1.46556}
```

```
fit1 = Plot[ z - a / ((x - x0)^2 + λ) /. fit, {x, 0, 2.2}, PlotLabel → "Lorentzian Fit",
   AxesLabel → {"Time (s)", "Signal (V)"}, PlotRange → All, PlotStyle → {Red, Thick}];
Show[fit1, p1]
```



F.3 Program for Calculating Daily Integrated Charge

RFA1 Charge Integration

Signal = $z - e^{[a(x-x_0)]}$

parameters[[n]]={day of month, a, z, x_0}

In[64]:=
```
dir = Directory[] <> "/Ecloud/Charge Sum exp/RFA1/";
dir2 = Directory[] <> "/Ecloud/Ecloud/IBEAM/";
parameters = ReadList[dir <> "mar.txt", {Number, Number, Number, Number}];
ibeam = ReadList[dir2 <> "mar.txt", {Number, Number}];
```

V=Signal in Volts datalogged at max value for detector.
in our case it is .652 seconds from the $8E or ealier data was taken on the $8D.
$V_{max} = z + -e^{[a(x_{beam}-x_0)]}$
z represent the readback through the MADC when no electron cloud is present. This is the offset from 0 of the detector signal when no beam is present.
a is a value found for each day. Based on actual data and extrapolation for days when we did not save the data. An equation is found for each day.
x_{beam} here is the beam intensity which we have datalogged since the start of the experiment.
Signal = $z - e^{[a(x-x_0)]}$
parameters[[n]]={day of month, a, z, x_0}

In[68]:=
```
it = 1;
tableVmax = Table[{0, 0} , {it, Length[ibeam]}];
tableCharge = Table[{0, 0}, {it, Length[parameters]}];
i = 1;
For[i = 1, i < Length[ibeam] + 1, i++,
 tableVmax[[i, 1]] = ibeam[[i, 1]];
   For[j = 1, j < Length[parameters] + 1, j++,
     If[ibeam[[i, 1]] == parameters[[j, 1]],
   tableVmax[[i, 2]] = parameters[[j, 3]] - e^(parameters[[j,2]] (ibeam[[i,2]]-parameters[[j,4]])) ;
   Break[]
        ]]]
```

Next we find the form of a lorentzian equation that best fits our data.
$v = z - \frac{A}{[(x-v_{max})^2+\lambda]}$
We found our z to be equal to -2.02431 from the average of all of the values found.

λ = 0.0027658 is used based on one snapshot sample taken on Mar 23. Several more averages of this value will improve the analysis.

V_{max} is the value for our maximum signal obtained by datalogging our detector signal .652 seconds from the $8E.

A is unknown, but can be calculated since we record Vmax at time .652

Since we are integrating from 0 to 1 contribution from the base signal will be z*1 or just z. Thus by subtracting z we will get the total charge integrated from 0 to 1
The integration of the lorentzian function with the charge contribution from the offset subtracted is shown below.

In[73]:=
```
v[x_] := z - A/((x - .651)^2 + λ)
```

In[74]:=
```
integral = Integrate[v[x], {x, 0, 1}] - z
```

Out[74]= -0.278297

Now to calculate A from z and Vmax for every day
A = (z - tableVmax[[37]]) * ((.651 - .651)^2 + λ)
A=(z - tableVmax[[i,2]])*λ

In[75]:=
```
z = -2.12197;
Clear[A];
λ = .0027658;
it = 1;
sum = Table[0, {it, 31}];
charge = Table[0, {it, 31}];
i = 1;

  For[i = 1, i ≤ Length[tableVmax], i++,
       (*Here we manually line up our parameters aray with the day since
   parameters[[1]] is not always for the first of sept but instead the 12*)
       A = (parameters[[tableVmax[[i, 1]], 3]] - tableVmax[[i, 2]]) * (λ);
      (*v[x_]:= z-A/((x- .651)^2+λ);
       we saved time by calc integral earlier*)
        If[tableVmax[[i, 1]] == 1,
          charge[[1]] = -55.35975871515078` A;
          sum[[1]] = charge[[1]] + sum[[1]];
          ];
       If[tableVmax[[i, 1]] == 2,
          charge[[2]] = -55.35975871515078` A;
          sum[[2]] = charge[[2]] + sum[[2]];
          ];
        If[tableVmax[[i, 1]] == 3,
          charge[[3]] = -55.35975871515078` A;
          sum[[3]] = charge[[3]] + sum[[3]];
          ];
       If[tableVmax[[i, 1]] == 4,
          charge[[4]] = -55.35975871515078` A;
          sum[[4]] = charge[[4]] + sum[[4]];
          ];
       If[tableVmax[[i, 1]] == 5,
          charge[[5]] = -55.35975871515078` A;
          sum[[5]] = charge[[5]] + sum[[5]];
          ];
        If[tableVmax[[i, 1]] == 6,
```



```
          charge[[6]] = -55.35975871515078` A;
          sum[[6]] = charge[[6]] + sum[[6]];
          ];
       If[tableVmax[[i, 1]] == 7,
          charge[[7]] = -55.35975871515078` A;
          sum[[7]] = charge[[7]] + sum[[7]];
          ];
       If[tableVmax[[i, 1]] == 8,
          charge[[8]] = -55.35975871515078` A;
          sum[[8]] = charge[[8]] + sum[[8]];
          ];
        If[tableVmax[[i, 1]] == 9,
          charge[[9]] = -55.35975871515078` A;
          sum[[9]] = charge[[9]] + sum[[9]];
          ];
       If[tableVmax[[i, 1]] == 10,
          charge[[10]] = -55.35975871515078` A;
          sum[[10]] = charge[[10]] + sum[[10]];
          ];
        If[tableVmax[[i, 1]] == 11,
          charge[[11]] = -55.35975871515078` A;
          sum[[11]] = charge[[11]] + sum[[11]];
          ];
        If[tableVmax[[i, 1]] == 12,
          charge[[12]] = -55.35975871515078` A;
          sum[[12]] = charge[[12]] + sum[[12]];
          ];
       If[tableVmax[[i, 1]] == 13,
          charge[[13]] = -55.35975871515078` A;
          sum[[13]] = charge[[13]] + sum[[13]];
          ];
      If[tableVmax[[i, 1]] == 14,
          charge[[14]] = -55.35975871515078` A;
          sum[[14]] = charge[[14]] + sum[[14]];
          ];
       If[tableVmax[[i, 1]] == 15,
          charge[[15]] = -55.35975871515078` A;
          sum[[15]] = charge[[15]] + sum[[15]];
          ];
       If[tableVmax[[i, 1]] == 16,
          charge[[16]] = -55.35975871515078` A;
          sum[[16]] = charge[[16]] + sum[[16]];
          ];
```

```
If[tableVmax[[i, 1]] == 17,
       charge[[17]] = -55.35975871515078` A;
       sum[[17]] = charge[[17]] + sum[[17]];
      ];
 If[tableVmax[[i, 1]] == 18,
       charge[[18]] = -55.35975871515078` A;
       sum[[18]] = charge[[18]] + sum[[18]];
      ];
 If[tableVmax[[i, 1]] == 19,
       charge[[19]] = -55.35975871515078` A;
       sum[[19]] = charge[[19]] + sum[[19]];
      ];
If[tableVmax[[i, 1]] == 20,
       charge[[20]] = -55.35975871515078` A;
       sum[[20]] = charge[[20]] + sum[[20]];
      ];
 If[tableVmax[[i, 1]] == 21,
       charge[[21]] = -55.35975871515078` A;
       sum[[21]] = charge[[21]] + sum[[21]];
      ];
 If[tableVmax[[i, 1]] == 22,
       charge[[22]] = -55.35975871515078` A;
       sum[[22]] = charge[[22]] + sum[[22]];
      ];
 If[tableVmax[[i, 1]] == 23,
       charge[[23]] = -55.35975871515078` A;
       sum[[23]] = charge[[23]] + sum[[23]];
      ];
If[tableVmax[[i, 1]] == 24,
       charge[[24]] = -55.35975871515078` A;
       sum[[24]] = charge[[24]] + sum[[24]];
      ];
 If[tableVmax[[i, 1]] == 25,
       charge[[25]] = -55.35975871515078` A;
       sum[[25]] = charge[[25]] + sum[[25]];
      ];
 If[tableVmax[[i, 1]] == 26,
       charge[[26]] = -55.35975871515078` A;
       sum[[26]] = charge[[26]] + sum[[26]];
      ];
 If[tableVmax[[i, 1]] == 27,
       charge[[27]] = -55.35975871515078` A;
       sum[[27]] = charge[[27]] + sum[[27]];
```



```
      ];
 If[tableVmax[[i, 1]] == 28,
       charge[[28]] = -55.35975871515078` A;
       sum[[28]] = charge[[28]] + sum[[28]];
      ];
If[tableVmax[[i, 1]] == 29,
       charge[[29]] = -55.35975871515078` A;
       sum[[29]] = charge[[29]] + sum[[29]];
      ];
 If[tableVmax[[i, 1]] == 30,
       charge[[30]] = -55.35975871515078` A;
       sum[[30]] = charge[[30]] + sum[[30]];
      ];
 If[tableVmax[[i, 1]] == 31,
       charge[[31]] = -55.35975871515078` A;
       sum[[31]] = charge[[31]] + sum[[31]];
      ];
]
```

The table "sum" consists of daily integrated charge for one month where the first number in the table corresponds to the first day of the month.

In[83]:=
```
Export["feb charge.dat", sum];
```

F.4 Program for Plotting the Energy Spectrum

Ecloud Energy Spectrum Measurements

RFA2 TiN

This reads in the initial data taken from D44 (the data logger program). The data is each detector signal and the grid voltage plotted against I:BEAM. The data is exported to an excel file. I then use excel to remove any duplicate I:BEAM value sets because it confuses the *Mathematica* fitting functions. Headings are also removed and then it is saved to a text file. In this cas 1 084a.txt.

This program can divide the data into multiple groups of data for analysis, though we ultimately only use 1 group.

This first part of the program reads in 1 data set and assigns it to table1x. This table has 1 extra entry.....table1x[[i,1]] is a group number. We divide our data in into the number of groups designated by the variable "groups" just below the ClearAll command. This is all that needs changed for the rest of the program will do individual calculations for each group.

In[640]:=
```
ClearAll["Global`*"]
groups = 1;
dir = Directory[] <> "/Ecloud/Ecloud/Ecloud scans from MCR/05dec2009 on 8D/";
table1 = ReadList[dir <> "1 084.txt",
   {Number, Number, Number, Number, Number, Number, Number, Number, Number}];
(*Make sure number of elements in VLIST iis the number of times you loop*)
VList = { 20, 40, 60, 80, 100, 120, 140, 160,
   180, 200, 220, 240, 260, 280, 300, 320, 340, 360, 380, 400};

ix1 = 1;
ix2 = 1;
ix3 = 1;
ix4 = 1; (*this is used to increment the table
 used to store the final information.  2 big table whose
 first entries will be one through the value of group*)

table4rfa1number = Table[{0, 0, 0, 0}, {i , groups * Length[VList] - 1}];
table4rfa2number = Table[{0, 0, 0, 0}, {i , groups * Length[VList] - 1}];
table4rfa3number = Table[{0, 0, 0, 0}, {i , groups * Length[VList] - 1}];
table4rfa4number = Table[{0, 0, 0, 0}, {i , groups * Length[VList] - 1}];
(*This is the final table that will have a first entry that represents
 the group number, a second entry that is the the voltage increment
 and a third entry that is the fractional energy lost when changing
 from from voltage to the next the 4th entry will be the error*)
q = 1;
j = 1;
table1x = Table[{0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {q, Length[table1] }];
For[i = 1, i ≤ Length[table1], i++,
 table1x[[i]] = {j, table1[[i, 1]], table1[[i, 2]],
   table1[[i, 3]], table1[[i, 4]], table1[[i, 5]], table1[[i, 6]],
   table1[[i, 7]], table1[[i, 8]], table1[[i, 9]], table1[[i, 10]]};
    j++;
    If[j ≥ groups + 1, j = 1, j = j];
   ]
```

Main Function

```
Needs["ErrorBarPlots`"]
For[number = 1, number < groups + 1, number++,
 table2 = Table[{0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {i, Length[table1x] - 1}];
 p = 1;
 rfa1 = Table[{0, 0}, {p, Length[VList]}];
 rfa2 = Table[{0, 0}, {p, Length[VList]}];
 rfa3 = Table[{0, 0}, {p, Length[VList]}];
 rfa4 = Table[{0, 0}, {p, Length[VList]}];

 rfa1error = Table[0, {p, Length[VList]}];
 rfa2error = Table[0, {p, Length[VList]}];
 rfa3error = Table[0, {p, Length[VList]}];
 rfa4error = Table[0, {p, Length[VList]}];
```



```
rfa1u = Table[{0, 0}, {p, Length[VList]}];
rfa2u = Table[{0, 0}, {p, Length[VList]}];
rfa3u = Table[{0, 0}, {p, Length[VList]}];
rfa4u = Table[{0, 0}, {p, Length[VList]}];

rfa1l = Table[{0, 0}, {p, Length[VList]}];
rfa2l = Table[{0, 0}, {p, Length[VList]}];
rfa3l = Table[{0, 0}, {p, Length[VList]}];
rfa4l = Table[{0, 0}, {p, Length[VList]}];

(*Fit a quadratic to the data taken with MI intensities between
 39 and 44 then evaluate using these fits for MI intensity of 42*)
lower = 40;
 upper = 44;
 value = 42.5;

(*Data is evaluated at each grid voltage.  This data often has a timing jitter
 that causes two distinct traces. Data is logged at the maximum value thus
 any timing jitter will cause a decrease in signal.  With this in mind,
data is filtered.  The next parameters are used for filtering.  The function
 is initially fit then add the parameter os to the function and get rid of all
 of the data points with values greater than (since the signal is negative)
 that function.  The os value must change for each grid voltage as to not
 accidentally get rid of data that you want so I look at the data with (-20 V)
 on the grid and look at the data with the highest potential (-400 V).  By
 finding the difference between os (-20 V Offset) and osl (-400 volt offset)
 and dividing it by the Length[Vlist] (which is the number of voltages scanned)
 we get Δos which can be used to increment os as you change the grid
 voltage (or the energy you are scanning.  I then check each fit
   visually and vary the offsets to ensure the proper data is filtered.
  *)

os = .30;
osl = .0018;
osSave = os;
osInitial = os;
Δos = ((os - osl) / Length[VList]);
t = 1;

Do[{
   (*The next While loops were added to provide a means of supplying my
    own chosen offset (variable = os) in the event that the meachanism I
    use for changing the offset does not work for a particular data set*)
  myVariable = VList[[t]];

  While[myVariable == 180,
   os = .06;
   Break[];
  ];
  While[myVariable == 200,
```



```
   os = .12;
   Break[];
  ];
  While[myVariable == 220,
   os = .055;
   (*os=osSave-Δos;*)
   Break[];
  ];

  While[myVariable == 240,
   os = .02;
   Break[];
  ];

  While[myVariable == 260,
   os = .045;
   Break[];
  ];

  While[myVariable == 280,
   os = .022;
   Break[];
  ];

  While[myVariable == 300,
   os = .022;
   Break[];
  ];

  While[myVariable == 320,
   os = .022;
   Break[];
  ];

  While[myVariable == 340,
   os = .022;
   Break[];
  ];

  While[myVariable == 360,
   os = .025;
   Break[];
  ];

  While[myVariable == 380,
   os = .022;
   Break[];
  ];

  While[myVariable == 400,
   osSave = os;
```

```
 os = .018;
 Break[];
];

While[myVariable ≠ 220 && myVariable ≠ 180 &&
  myVariable ≠ 200 && myVariable ≠ 220 && myVariable ≠ 240 &&
  myVariable ≠ 260 && myVariable ≠ 280 && myVariable ≠ 300 && myVariable ≠ 320 &&
  myVariable ≠ 340 && myVariable ≠ 360 && myVariable ≠ 380 && myVariable ≠ 400,
 os = osSave;
 Break[];
];
(*The min and max values allow us to use
  a range of grid voltages in this case I am using from -16 to -
 24 rather than just -20 V.  I then write these values into table2*)
min = -4 + -VList[[t]];
max = 4 + -VList[[t]];
j = 1;
i1 = 1;
For[i1 = 1, i1 ≤ Length[table1x], i1++,
 If[table1x[[i1, 11]] > min && table1x[[i1, 11]] < max &&
    table1x[[i1, 2]] > lower && table1x[[i1, 2]] < upper,
   table2[[j, 1]] = table1x[[i1, 2]];
   table2[[j, 2]] = table1x[[i1, 3]];
   table2[[j, 3]] = table1x[[i1, 4]];
   table2[[j, 4]] = table1x[[i1, 5]];
   table2[[j, 5]] = table1x[[i1, 6]];
   table2[[j, 6]] = table1x[[i1, 7]];
   table2[[j, 7]] = table1x[[i1, 8]];
   table2[[j, 8]] = table1x[[i1, 9]];
   table2[[j, 9]] = table1x[[i1, 10]];
   table2[[j, 10]] = table1x[[i1, 11]];
   j++
  ];
];

j = 1;
For [i = 1, i ≤ Length[table1], i++,
    If[table2[[j, 1]] ≠ 0,
        j++;
       ];
     ];
k = 1;
cloud1 = Table[{0, 0}, {k, j - 1}];
cloud2 = Table[{0, 0}, {k, j - 1}];
cloud3 = Table[{0, 0}, {k, j - 1}];
cloud4 = Table[{0, 0}, {k, j - 1}];

For[k = 1, k < j, k++,
    cloud1[[k]] = {table2[[k, 1]], table2[[k, 2]]};
    cloud2[[k]] = {table2[[k, 3]], table2[[k, 4]]};
    cloud3[[k]] = {table2[[k, 5]], table2[[k, 6]]};
```



```
    cloud4[[k]] = {table2[[k, 7]], table2[[k, 8]]};
   ];
(*I sorted from least to greates because linear model fit needed this I thought*)
Sort[cloud1, #1[[1]] < #2[[1]] &];
Sort[cloud2, #1[[1]] < #2[[1]] &];
Sort[cloud3, #1[[1]] < #2[[1]] &];
Sort[cloud4, #1[[1]] < #2[[1]] &];

(*My naming convention is not ideal.  I
 believe I initially used a linear fit rather than quadratic,
thus I called it lm1 and this quadratic fit lm2.  THen to add to the confusion,
I first did it all with RFA1 which is lm2 and
 then called RFA2 lm2b rfa3 lm2c and rfa4 lm2d   *)
lm2 = LinearModelFit[cloud1, {x^2, x}, x]; (*CLOUD1 *)
lm2b = LinearModelFit[cloud2, {x^2, x}, x]; (*CLOUD2 *)
lm2c = LinearModelFit[cloud3, {x^2, x}, x]; (*CLOUD3 *)
lm2d = LinearModelFit[cloud4, {x^2, x}, x]; (*CLOUD4 *)

(*Now I make a new table cloud1a etc and zero it
 out.  Once I have filtered the data caused by the timing jitter
 I will assign it to cloud1a, cloud2a, cloud3a and cloud4a *)
z = 1;
cloud1a = Table[{0, 0}, {z, Length[cloud1] - 1}];
cloud2a = Table[{0, 0}, {z, Length[cloud2] - 1}];
cloud3a = Table[{0, 0}, {z, Length[cloud3] - 1}];
cloud4a = Table[{0, 0}, {z, Length[cloud4] - 1}];

i = 1;
j = 1;
For[i = 1, i < Length[cloud1], i++,
 If[(lm2[cloud1[[i, 1]]] + os) > cloud1[[i, 2]], (*By adding the fitted data plus
    the offset I can use this a benchmark of jittered data to get rid of  *)
   cloud1a[[j]] = cloud1[[i]];
   j++;
    ];
  ];

i = 1;
k = 1;
For[i = 1, i < Length[cloud2], i++,
 If[(lm2b[cloud2[[i, 1]]] + os) > cloud2[[i, 2]],
   cloud2a[[k]] = cloud2[[i]];
   k++;
    ];
  ];

i = 1;
l = 1;
For[i = 1, i < Length[cloud3], i++,
 If[(lm2c[cloud3[[i, 1]]] + os) > cloud3[[i, 2]],
   cloud3a[[l]] = cloud3[[i]];
```



```
   l++;
    ];
  ];

i = 1;
m = 1;
For[i = 1, i < Length[cloud4], i++,
 If[(lm2d[cloud4[[i, 1]]] + os) > cloud4[[i, 2]],
   cloud4a[[m]] = cloud4[[i]];
   m++;
    ];
  ];

j = 1;
(*
How many data points are in each data set,
because the length of the tables does not get totally filled.  By searching for
 the 0 value one can find how many data points were written to the table with the
 purpose being ultimately to create a new table that no longer has all of these 0s
*)
i = 1;
For [i = 1, i < Length[cloud1], i++,
    If[cloud1a[[j, 1]] ≠ 0,
        j++;
       ];
     ];

k = 1;
i = 1;
For [i = 1, i < Length[cloud2], i++,
    If[cloud2a[[k, 1]] ≠ 0,
        k++;
       ];
     ];

l = 1;
i = 1;
For [i = 1, i < Length[cloud3], i++,
    If[cloud3a[[l, 1]] ≠ 0,
        l++;
       ];
     ];

m = 1;
i = 1;
For [i = 1, i < Length[cloud4], i++,
    If[cloud4a[[m, 1]] ≠ 0,
        m++;
       ];
     ];
```



```
n = 1;
(*Define new tables cloud1b, cloud2b,
cloud3b and cloud4b my filtered data (filtered because of the timing jitter)
 will be written to these tables and these tables will not include any
 of the extra 0 values at the end (they will be the correct length  *)
cloud1b = Table[{0, 0}, {n, j - 1}];
cloud2b = Table[{0, 0}, {n, k - 1}];
cloud3b = Table[{0, 0}, {n, l - 1}];
cloud4b = Table[{0, 0}, {n, m - 1}];

n = 1;
For[n = 1, n < j, n++,
   cloud1b[[n]] = {cloud1a[[n, 1]], cloud1a[[n, 2]]};
    ];

n = 1;
For[n = 1, n < k, n++,
   cloud2b[[n]] = {cloud2a[[n, 1]], cloud2a[[n, 2]]};
    ];

n = 1;
For[n = 1, n < l, n++,
   cloud3b[[n]] = {cloud3a[[n, 1]], cloud3a[[n, 2]]};
    ];
n = 1;

For[n = 1, n < m, n++,
   cloud4b[[n]] = {cloud4a[[n, 1]], cloud4a[[n, 2]]};
    ];

(*After all of that,  the data is finally filtered properly and
 is ready for the actual fitting that will be used for analysis*)
lm3 = LinearModelFit[cloud1b, {x^2, x}, x];
lm3b = LinearModelFit[cloud2b, {x^2, x}, x];
lm3c = LinearModelFit[cloud3b, {x^2, x}, x];
lm3d = LinearModelFit[cloud4b, {x^2, x}, x];

lm3p = lm3["MeanPredictionBands"];
lm3bp = lm3b["MeanPredictionBands"];
lm3cp = lm3c["MeanPredictionBands"];
lm3dp = lm3d["MeanPredictionBands"];

rfa1[[t, 1]] = VList[[t]];
rfa1[[t, 2]] = lm3[value];
rfa1u[[t, 1]] = VList[[t]];
rfa1u[[t, 2]] = lm3p[[2]] /. x → value;
rfa1l[[t, 1]] = VList[[t]];
rfa1l[[t, 2]] = lm3p[[1]] /. x → value;
rfa1error[[t]] = lm3[value] - lm3p[[2]] /. x → value;
(*Subtract lower confidence interval value from the
 measured value to come up with the error for each data point. *)
```

```
  (*Assign values to variables to be used in the next script*)
  rfa2[[t, 1]] = VList[[t]];
  rfa2[[t, 2]] = lm3b[value];
  rfa2u[[t, 1]] = VList[[t]];
  rfa2u[[t, 2]] = lm3bp[[2]] /. x → value;
  rfa2l[[t, 1]] = VList[[t]];
  rfa2l[[t, 2]] = lm3bp[[1]] /. x → value;
  rfa2error[[t]] = lm3b[value] - lm3bp[[2]] /. x → value;

  rfa3[[t, 1]] = VList[[t]];
  rfa3[[t, 2]] = lm3c[value];
  rfa3u[[t, 1]] = VList[[t]];
  rfa3u[[t, 2]] = lm3cp[[2]] /. x → value;
  rfa3l[[t, 1]] = VList[[t]];
  rfa3l[[t, 2]] = lm3cp[[1]] /. x → value;
  rfa3error[[t]] = lm3c[value] - lm3cp[[2]] /. x → value;


  rfa4[[t, 1]] = VList[[t]];
  rfa4[[t, 2]] = lm3d[value];
  rfa4u[[t, 1]] = VList[[t]];
  rfa4u[[t, 2]] = lm3dp[[2]] /. x → value;
  rfa4l[[t, 1]] = VList[[t]];
  rfa4l[[t, 2]] = lm3dp[[1]] /. x → value;
  rfa4error[[t]] = lm3d[value] - lm3dp[[2]] /. x → value;

  pic1 = ListPlot[cloud2, PlotRange → All, PlotStyle → Opacity[0.6]];
  pic2 = ListPlot[cloud2b, PlotRange → All, PlotStyle → Red];
  pic3 = Plot[lm3b[x], {x, 41, 44}, PlotRange → All];
  pic4 = Graphics[{ PointSize[Large], Black, Point[{value, lm3b[value]}]}];
  pic5 = Plot[lm2b[x] + os, {x, 41, 44}, PlotRange → All, PlotStyle → Green];
  pic6 = Plot[lm2b[x], {x, 41, 44}, PlotRange → All, PlotStyle → Gray];
  Print[Show[pic1, pic2, pic3, pic4, pic5, PlotLabel →
     "Fit at -" <> ToString[VList[[t]]] <> " Volts With Offset " <> ToString[os] ]];

  osSave = (osSave - Δos);
  os = (os - Δos);

  table2 = Table[{0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {i, Length[table1] - 1}];},
 {t, 1, Length[VList], 1}
];

i = 1;
(*Here tables are created for storing the the signal strength
     difference for 1 grid potential to the next titled ΔRFA1....and
  knowing that the error will be a difference of two errors from the
  data above tables are made titled errornewrfa1-
 4.  These are to store the error which consists of a difference between 2
  error signals.  The difference between the error for one grid potential
  and the next divided by the total signal strength when there is only -
```



```
  20 V on the grid is found.  Since charge is proportional to Voltage this will
   give us the fraction of cloud signal lost as you go from one grid potential
   to the next.  This is written to the table  errornewrfa1, 2, 3, and 4.  *)
ΔRFA1 = Table[{0, 0}, {i, Length[rfa1] - 1}];
errornewrfa1 = Table[{0}, {i, Length[rfa1] - 1}];

ΔRFA2 = Table[{0, 0}, {i, Length[rfa2] - 1}];
errornewrfa2 = Table[{0}, {i, Length[rfa2] - 1}];

ΔRFA3 = Table[{0, 0}, {i, Length[rfa3] - 1}];
errornewrfa3 = Table[{0}, {i, Length[rfa3] - 1}];

ΔRFA4 = Table[{0, 0}, {i, Length[rfa4] - 1}];
errornewrfa4 = Table[{0}, {i, Length[rfa4] - 1}];

For[i = 1, i ≤ Length[rfa1] - 1, i++,
 (*As an example,
 here we will start by writing the 20V data to temp1 and the 40 V data to temp2 so
     we can subtract temp1 from temp2 to get the difference in signal strength as
     we change the grid voltage from 20 to 40 and writes it to ΔRFA1[[i]].  The
     next iteration then calculates for grid potential 40 to 60 and so on.....*)
 temp1 = rfa1[[i]];
 temp2 = rfa1[[i + 1]];
 ΔRFA1[[i]] = temp2 - temp1;
 (*Our new error is found by taking the square root of the
   sum of the squares of the two error signals that we are subtracting
   or more clearly when  x=y+-z then δx=√((δz)^2+(δy)^2)   where δ imples error in  *)
 errornewrfa1[[i]] = Sqrt[ (rfalerror[[i]])^2 + (rfalerror[[i + 1]])^2];
];
For[i = 1, i ≤ Length[rfa2] - 1, i++,
 temp1 = rfa2[[i]];
 temp2 = rfa2[[i + 1]];
 ΔRFA2[[i]] = temp2 - temp1;
 errornewrfa2[[i]] = Sqrt[ (rfa2error[[i]])^2 + (rfa2error[[i + 1]])^2];
];
For[i = 1, i ≤ Length[rfa3] - 1, i++,
 temp1 = rfa3[[i]];
 temp2 = rfa3[[i + 1]];
 ΔRFA3[[i]] = temp2 - temp1;
 errornewrfa3[[i]] = Sqrt[ (rfa3error[[i]])^2 + (rfa3error[[i + 1]])^2];
];
For[i = 1, i ≤ Length[rfa4] - 1, i++,
 temp1 = rfa4[[i]];
 temp2 = rfa4[[i + 1]];
 ΔRFA4[[i]] = temp2 - temp1;
 errornewrfa4[[i]] = Sqrt[ (rfa4error[[i]])^2 + (rfa4error[[i + 1]])^2];
];
(*By summing all of the signal differences we
 get a charge total value which we write to these parameters*)
Qtotrfa1 = 0;
```

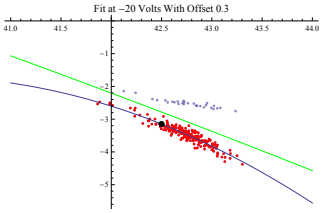


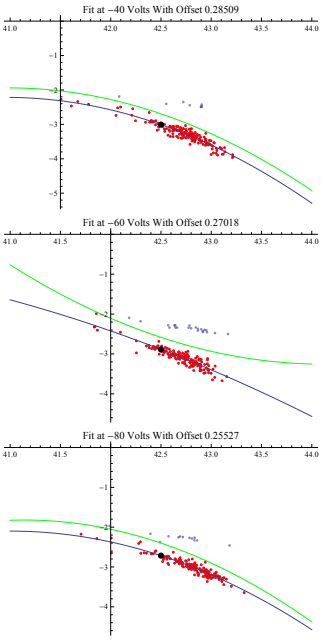
```
Qtotrfa2 = 0;
Qtotrfa3 = 0;
Qtotrfa4 = 0;

For[i = 1, i ≤ Length[rfa1] - 1, i++,
  Qtotrfa1 = ΔRFA1[[i]][[2]] + Qtotrfa1];

For[i = 1, i ≤ Length[rfa2] - 1, i++,
  Qtotrfa2 = ΔRFA2[[i]][[2]] + Qtotrfa2];

For[i = 1, i ≤ Length[rfa3] - 1, i++,
  Qtotrfa3 = ΔRFA3[[i]][[2]] + Qtotrfa3];

For[i = 1, i ≤ Length[rfa4] - 1, i++,
  Qtotrfa4 = ΔRFA4[[i]][[2]] + Qtotrfa4];

(*Next we make a table that consists of 2 terms,
the first being the grid potential and the second being the fractional change
 in the signal strength as you change the grid potential to the next value*)
For[i = 1, i ≤ Length[rfa1] - 1, i++,
 table4rfa1 =
  Table[{ΔRFA1[[i]][[1]], ΔRFA1[[i]][[2]] / Qtotrfa1}, {i, Length[rfa1] - 1}]];
(*Here using errornewrfa1 we calculate the new upper 95% confidence interval*)
For[i = 1, i ≤ Length[rfa1] - 1, i++,
 table4urfa1 = Table[{ΔRFA1[[i]][[1]],
    (ΔRFA1[[i]][[2]] + errornewrfa1[[i]]) / Qtotrfa1}, {i, Length[rfa1] - 1}]];
(*Here using errornewrfa1 we calculate the new lower 95% confidence interval*)
For[i = 1, i ≤ Length[rfa1] - 1, i++,
 table4lrfa1 = Table[{ΔRFA1[[i]][[1]],
    (ΔRFA1[[i]][[2]] - errornewrfa1[[i]]) / Qtotrfa1}, {i, Length[rfa1] - 1}]];

For[i = 1, i ≤ Length[rfa2] - 1, i++,
 table4rfa2 =
  Table[{ΔRFA2[[i]][[1]], ΔRFA2[[i]][[2]] / Qtotrfa2}, {i, Length[rfa2] - 1}]];

For[i = 1, i ≤ Length[rfa2] - 1, i++,
 table4urfa2 = Table[{ΔRFA2[[i]][[1]],
    (ΔRFA2[[i]][[2]] + errornewrfa2[[i]]) / Qtotrfa2}, {i, Length[rfa2] - 1}]];

For[i = 1, i ≤ Length[rfa2] - 1, i++,
 table4lrfa2 = Table[{ΔRFA2[[i]][[1]],
    (ΔRFA2[[i]][[2]] - errornewrfa2[[i]]) / Qtotrfa2}, {i, Length[rfa2] - 1}]];

For[i = 1, i ≤ Length[rfa3] - 1, i++,
 table4rfa3 =
  Table[{ΔRFA3[[i]][[1]], ΔRFA3[[i]][[2]] / Qtotrfa3}, {i, Length[rfa3] - 1}]];

For[i = 1, i ≤ Length[rfa3] - 1, i++,
 table4urfa3 = Table[{ΔRFA3[[i]][[1]],
    (ΔRFA3[[i]][[2]] + errornewrfa3[[i]]) / Qtotrfa3}, {i, Length[rfa3] - 1}]];
```

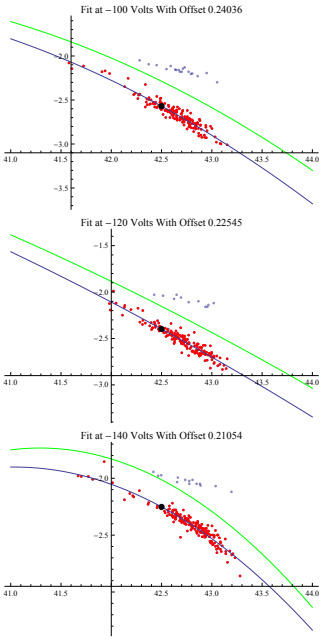


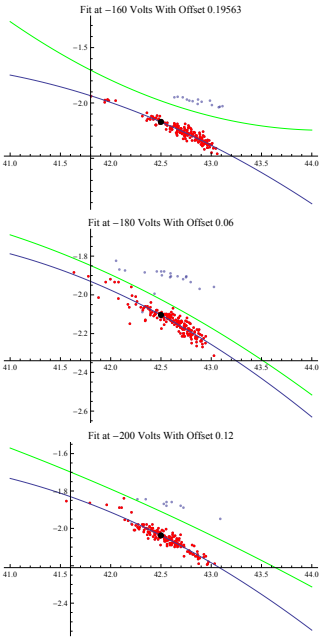
```
For[i = 1, i ≤ Length[rfa3] - 1, i++,
 table4lrfa3 = Table[{ΔRFA3[[i]][[1]],
    (ΔRFA3[[i]][[2]] - errornewrfa3[[i]]) / Qtotrfa3}, {i, Length[rfa3] - 1}]];

For[i = 1, i ≤ Length[rfa4] - 1, i++,
 table4rfa4 =
  Table[{ΔRFA4[[i]][[1]], ΔRFA4[[i]][[2]] / Qtotrfa4}, {i, Length[rfa4] - 1}]];

For[i = 1, i ≤ Length[rfa4] - 1, i++,
 table4urfa4 = Table[{ΔRFA4[[i]][[1]],
    (ΔRFA4[[i]][[2]] + errornewrfa4[[i]]) / Qtotrfa4}, {i, Length[rfa4] - 1}]];

For[i = 1, i ≤ Length[rfa4] - 1, i++,
 table4lrfa4 = Table[{ΔRFA4[[i]][[1]],
    (ΔRFA4[[i]][[2]] - errornewrfa4[[i]]) / Qtotrfa4}, {i, Length[rfa4] - 1}]];

i = 1;
rfa1Error = Table[{0, 0}, {i, Length[rfa1] - 1}];
i = 1;
For[i = 1, i ≤ Length[rfa1] - 1, i++,
 rfa1Error[[i]] = (table4urfa1[[i, 2]] - table4rfa1[[i, 2]])
];

i = 1;
rfa2Error = Table[{0, 0}, {i, Length[rfa2] - 1}];
i = 1;
For[i = 1, i ≤ Length[rfa2] - 1, i++,
 rfa2Error[[i]] = (table4urfa2[[i, 2]] - table4rfa2[[i, 2]])
];

i = 1;
rfa3Error = Table[{0, 0}, {i, Length[rfa3] - 1}];
i = 1;
For[i = 1, i ≤ Length[rfa3] - 1, i++,
 rfa3Error[[i]] = (table4urfa3[[i, 2]] - table4rfa3[[i, 2]])
];

i = 1;
rfa4Error = Table[{0, 0}, {i, Length[rfa4] - 1}];
i = 1;
For[i = 1, i ≤ Length[rfa4] - 1, i++,
 rfa4Error[[i]] = (table4urfa4[[i, 2]] - table4rfa4[[i, 2]])
];


(*This was set up to divide the data into groups,
we did not use this functionality in the final analysis*)

ix1 = 1;
For[i = 1, i ≤ Length[table4rfa1], i++,
 table4rfa1number[[ix1, 1]] = number;
```

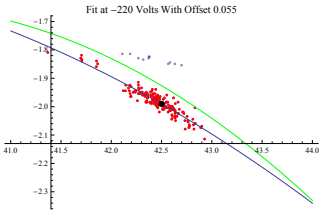
```
 table4rfa1number[[ix1, 2]] = table4rfa1[[i]][[1]];
 table4rfa1number[[ix1, 3]] = table4rfa1[[i]][[2]];
 table4rfa1number[[ix1, 4]] = rfa1Error[[i]];
 ix1++;
];

ix2 = 1;
For[i = 1, i ≤ Length[table4rfa1], i++,
 table4rfa2number[[ix2, 1]] = number;
 table4rfa2number[[ix2, 2]] = table4rfa2[[i]][[1]];
 table4rfa2number[[ix2, 3]] = table4rfa2[[i]][[2]];
 table4rfa2number[[ix2, 4]] = rfa2Error[[i]];
 ix2++;
];

ix3 = 1;
For[i = 1, i ≤ Length[table4rfa3], i++,
 table4rfa3number[[ix3, 1]] = number;
 table4rfa3number[[ix3, 2]] = table4rfa3[[i]][[1]];
 table4rfa3number[[ix3, 3]] = table4rfa3[[i]][[2]];
 table4rfa3number[[ix3, 4]] = rfa3Error[[i]];
 ix3++;
];

ix4 = 1;
For[i = 1, i ≤ Length[table4rfa1], i++,
 table4rfa4number[[ix4, 1]] = number;
 table4rfa4number[[ix4, 2]] = table4rfa4[[i]][[1]];
 table4rfa4number[[ix4, 3]] = table4rfa4[[i]][[2]];
 table4rfa4number[[ix4, 4]] = rfa4Error[[i]];
 ix4++;
];

]
```

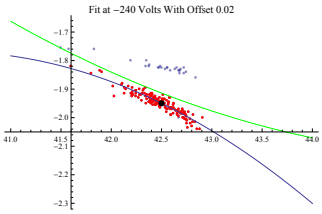


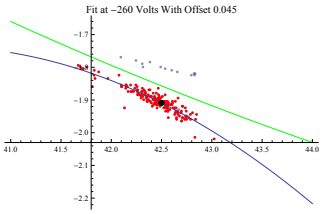


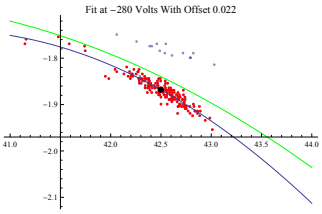


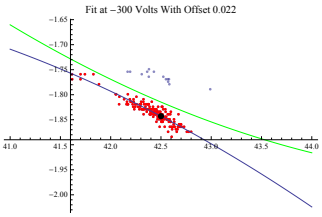


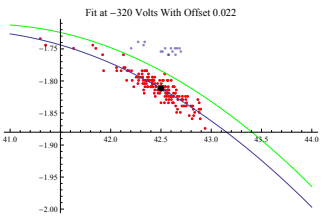

Fit at −220 Volts With Offset 0.055

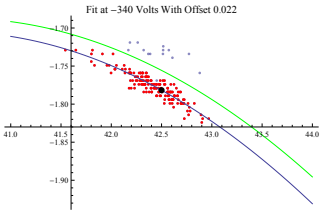

Fit at −240 Volts With Offset 0.02

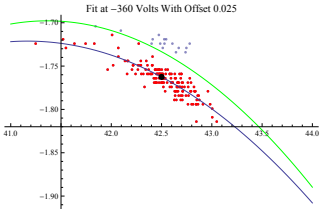

Fit at −260 Volts With Offset 0.045

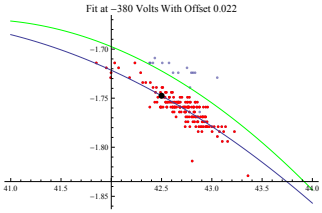

Fit at −280 Volts With Offset 0.022

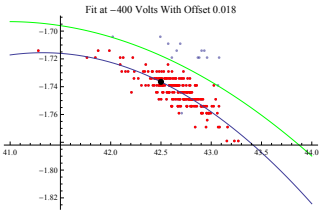

Fit at −300 Volts With Offset 0.022


Fit at −320 Volts With Offset 0.022


Fit at −340 Volts With Offset 0.022


Fit at −360 Volts With Offset 0.025


Fit at −380 Volts With Offset 0.022

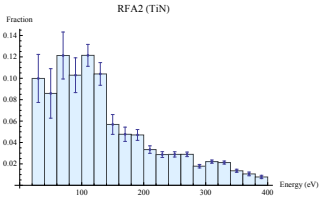

Fit at −400 Volts With Offset 0.018

Run this to write data to final tables

```
inew = 1;

table4rfa1Finally = Table[{0, 0}, {inew, (Length[VList] - 1)}];
rfa1ErrorFinally = Table[{0}, {inew, (Length[VList] - 1)}];

For[j = 1, j ≤ Length[VList] - 1, j++,
  table4rfa1Finally[[j, 1]] = table4rfa1number[[j, 2]];
  table4rfa1Finally[[j, 2]] = table4rfa1number[[j, 3]];
  rfa1ErrorFinally[[j]] = table4rfa1number[[j, 4]];
 ];


inew = 1;

table4rfa2Finally = Table[{0, 0}, {inew, (Length[VList] - 1)}];
rfa2ErrorFinally = Table[{0}, {inew, (Length[VList] - 1)}];

For[j = 1, j ≤ Length[VList] - 1, j++,
  table4rfa2Finally[[j, 1]] = table4rfa2number[[j, 2]];
  table4rfa2Finally[[j, 2]] = table4rfa2number[[j, 3]];
  rfa2ErrorFinally[[j]] = table4rfa2number[[j, 4]];
 ];


inew = 1;

table4rfa3Finally = Table[{0, 0}, {inew, (Length[VList] - 1)}];
rfa3ErrorFinally = Table[{0}, {inew, (Length[VList] - 1)}];

For[j = 1, j ≤ Length[VList] - 1, j++,
  table4rfa3Finally[[j, 1]] = table4rfa3number[[j, 2]];
  table4rfa3Finally[[j, 2]] = table4rfa3number[[j, 3]];
  rfa3ErrorFinally[[j]] = table4rfa3number[[j, 4]];
 ];
inew = 1;
table4rfa4Finally = Table[{0, 0}, {inew, (Length[VList] - 1)}];
rfa4ErrorFinally = Table[{0}, {inew, (Length[VList] - 1)}];

For[j = 1, j ≤ Length[VList] - 1, j++,
 table4rfa4Finally[[j, 1]] = table4rfa4number[[j, 2]];
 table4rfa4Finally[[j, 2]] = table4rfa4number[[j, 3]];
 rfa4ErrorFinally[[j]] = table4rfa4number[[j, 4]];
]
```


Make Plot with Error bars

```
plot3 = Show[RectangleChart[Join[{{20, 0}}, table4rfa2Finally],
    AxesLabel → {"Energy (eV)", "Fraction"}, PlotLabel → "RFA2 (TiN)",
    ChartStyle → LightBlue, BarSpacing → {0, 0}]];
i = 1;
plot4 =
  ErrorListPlot[{{{30, table4rfa2Finally[[1, 2]]}, ErrorBar[rfa2ErrorFinally[[1]]]},
    {{50, table4rfa2Finally[[2, 2]]}, ErrorBar[rfa2ErrorFinally[[2]]]},
    {{70, table4rfa2Finally[[3, 2]]}, ErrorBar[rfa2ErrorFinally[[3]]]},
    {{90, table4rfa2Finally[[4, 2]]}, ErrorBar[rfa2ErrorFinally[[4]]]},
    {{110, table4rfa2Finally[[5, 2]]}, ErrorBar[rfa2ErrorFinally[[5]]]},
    {{130, table4rfa2Finally[[6, 2]]}, ErrorBar[rfa2ErrorFinally[[6]]]},
    {{150, table4rfa2Finally[[7, 2]]}, ErrorBar[rfa2ErrorFinally[[7]]]},
    {{170, table4rfa2Finally[[8, 2]]}, ErrorBar[rfa2ErrorFinally[[8]]]},
    {{190, table4rfa2Finally[[9, 2]]}, ErrorBar[rfa2ErrorFinally[[9]]]},
    {{210, table4rfa2Finally[[10, 2]]}, ErrorBar[rfa2ErrorFinally[[10]]]},
    {{230, table4rfa2Finally[[11, 2]]}, ErrorBar[rfa2ErrorFinally[[11]]]},
    {{250, table4rfa2Finally[[12, 2]]}, ErrorBar[rfa2ErrorFinally[[12]]]},
    {{270, table4rfa2Finally[[13, 2]]}, ErrorBar[rfa2ErrorFinally[[13]]]},
    {{290, table4rfa2Finally[[14, 2]]}, ErrorBar[rfa2ErrorFinally[[14]]]},
    {{310, table4rfa2Finally[[15, 2]]}, ErrorBar[rfa2ErrorFinally[[15]]]},
    {{330, table4rfa2Finally[[16, 2]]}, ErrorBar[rfa2ErrorFinally[[16]]]},
    {{350, table4rfa2Finally[[17, 2]]}, ErrorBar[rfa2ErrorFinally[[17]]]},
    {{370, table4rfa2Finally[[18, 2]]}, ErrorBar[rfa2ErrorFinally[[18]]]},
    {{390, table4rfa2Finally[[19, 2]]}, ErrorBar[rfa2ErrorFinally[[19]]]} }];

Show[plot3, plot4]
(*Always show parameters so plots can be recreated properly*)
lower
 upper
 value
groups
os
osl
```




```
40

44
```



```
42.5

1

0.00309

0.0018
```