Genetic Programming: A Novel Method for Neutrino Analysis

Kaeli Hughes
Advisor: Professor Amy Connolly
April 2017

Abstract

In this project we have investigated how genetic programming, a form a machine learning, can improve the analysis of data from the Antarctic Impulsive Transient Antenna (ANITA), a balloon experiment searching for ultra-high-energy (UHE) neutrinos that interact in the ice in Antarctica. Discovering these UHE neutrinos will unlock new information about the universe and will lead the way into a new era of neutrino astronomy. ANITA, like many astroparticle experiments, relies heavily on being able to differentiate between signal events and background noise. This project has taken advantage of genetic programming algorithms to effectively model the anthropogenic backgrounds.

Genetic programming takes advantage of an evolutionary style of function generation, in which prospective functions meant to describe a dataset are populated and tested in sets called "generations", with the best fitting functions populating the next generation of functions. One such program that implements a genetic programming algorithm is called Karoo GP, a program written by Kai Staats, a scientist currently working with the Laser Interferometer Gravitational-Wave Observatory (LIGO). Karoo GP was designed for the Square Kilometer Array (SKA) radio experiment and has been used by the LIGO Collaboration. In addition, it outputs the fitting algorithms as analytical functions, which would easily allow us to include it in the analysis.

In this research project, Karoo GP was used to model the data from the ANITA experiment. Events in the data were characterized by two variables, the signal-to-noise ratio (SNR) and the correlation between the voltage peaks from different antennas. At the moment, the best model from Karoo GP does not adequately describe the data, and will not be used in further analysis. However, we suggest ways in which the algorithm can be improved, and it is possible that Karoo GP will allow for better optimization of the neutrino flux limit in the future.

Acknowledgements

I would first like to thank my research advisor, Professor Amy Connolly, who has mentored me for four years and without whom this research project would not be possible. In addition, I would like to thank everyone in my research group at Ohio State, in particular Patrick Allison, Carl Pfendner, Jordan Hanson, Brian Dailey, Sam Stafford, Oindree Banerjee, and Brian Clark, for always answering my questions and being incredibly helpful, especially in the beginning of my research career. I would like to thank Kai Staats, for providing the code for Karoo GP and coming to Ohio State in person to talk about my project. Finally, a special thanks to Amy Connolly, Brian Clark, Oindree Banerjee, Lucas Beaufore, and Richard Hughes for reading through and editing multiple versions of this paper. Without all of these people, this work would not have been possible.

Table of Contents

List of Figures

1.0. Introduction to Ultra High Energy (UHE) Neutrinos

1.1. Neutrino Astronomy

Neutrinos are light, neutral subatomic particles that are produced throughout the universe in a multitude of interactions. There are three “flavors” of neutrinos, and although the exact masses of each flavor are unknown, the current upper bound on the mass of the electron neutrino is 2 eV [10], which is about 250,000 times smaller than the mass of the electron. At the highest energies, above 10^{18} eV, it is theorized that neutrinos could be produced from ultra high energy (UHE) sources or from interactions between cosmic rays and the Cosmic Microwave background [9]. Neutrinos are ideal messengers to study the UHE regime, because they are neutral and will travel directly from their source without being bent by magnetic fields. In addition, the low cross section of neutrinos (between $10^{-32.5}$ and $10^{-30.7}$ cm^2 in the 10^{18} -10^{21} eV energy range [11]) makes it possible for them to travel long distances without interacting.

However, the qualities of neutrinos that make them ideal messengers also make them very difficult to detect. In addition, the expected UHE neutrino flux is expected to be approximately 1 neutrino per square kilometer per year per steradian [6], which means that the events are both rare and difficult to detect. To combat these issues, neutrino experiments have been searching for interactions in an unusual location: the ice in Antarctica.

1.2. Why Antarctica is a practical detector

When neutrinos interact with matter, they produce a shower of secondary particles in which the charged particles of the shower are traveling faster than the speed of light in ice. This in turn causes photons to radiate outward [1]. Wavelengths on the length scale of the shower are coherently emitted in the radio regime, creating a broadband signal. This effect is known as the Askaryan Effect [1].

Antarctic ice in particular is well suited to see these interactions, because ice is radio-transparent, meaning that the radio signals created by these interactions are capable of traveling long distances, up to ~1 km [5]. Antarctica also has a low amount of continuous wave (CW) contamination from things like radios and communication devices due to the low number of people living on the continent. In addition, Antarctica has over one million square kilometers of ice available as a target medium. Therefore, if an experiment chose to survey a large portion of the ice, it maximizes the possibility of observing an event.

1.3. The Antarctic Impulsive Transient Antenna (ANITA)

One experiment that is taking advantage of the Antarctic ice to detect UHE neutrinos is called the Antarctic Impulsive Transient Antenna (ANITA). ANITA is a balloon experiment that flies for approximately 30 days during the Antarctic summer and records potential radio signals over one million cubic kilometers of ice [2]. This experiment has flown 4 times since 2006, most recently in December 2016, and it hopes to detect the UHE neutrinos that interact in the ice there.

As in many experiments, one of the most important parts of the analysis is differentiating between signal and background events. The next chapter will discuss the previous analysis work done.

2.0. Previous Analysis Work

The research project presented here builds off of the thesis work done by Dr. Brian Dailey, a recent graduate of Ohio State's Physics PhD program. His thesis [3] focused on re-analyzing the data from the ANITA-II flight, and one of the novel methods presented in his thesis was binning the candidate neutrino signals into equal area bins of Antarctic ice. The main goal of analysis for the ANITA experiment is to cut away noise while maximizing the number of neutrinos that may have been seen. This thesis did this optimization on a per-bin basis, allowing the cuts to be specifically designed based on the expected noise levels in each area of Antarctica. This will improve the ability to cut background events out of the data. An example of the Antarctic bins is shown below in Figure 1:

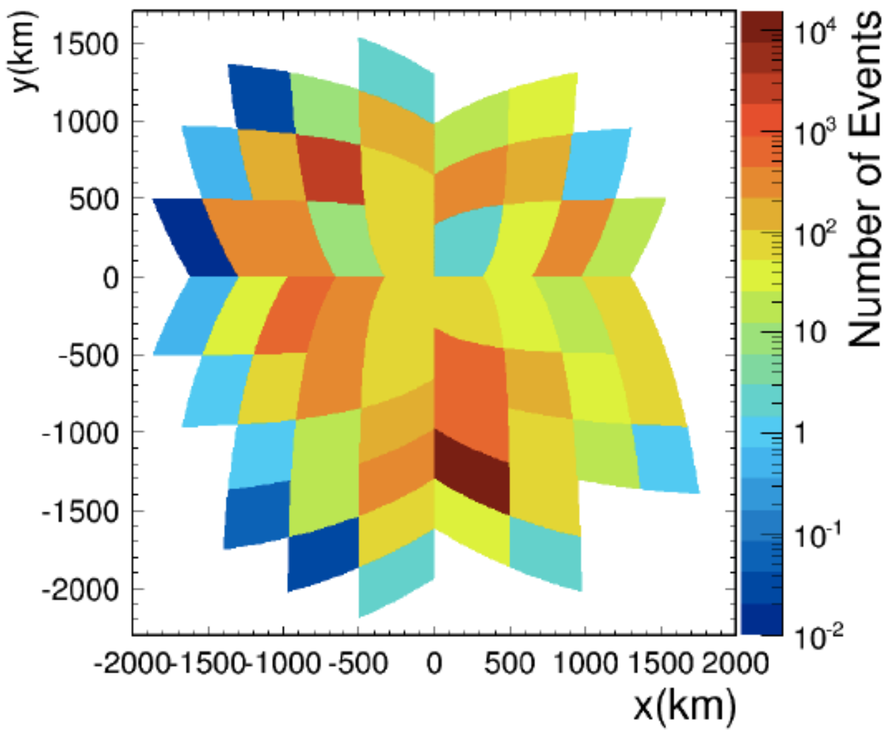

Figure 1: A plot of background events from the 10% sample, binned into equal area bins of Antarctic ice. From Brian Dailey [3]

For each Antarctic bin, a rotated cross correlation cut was optimized, which is best illustrated in Figure 2:

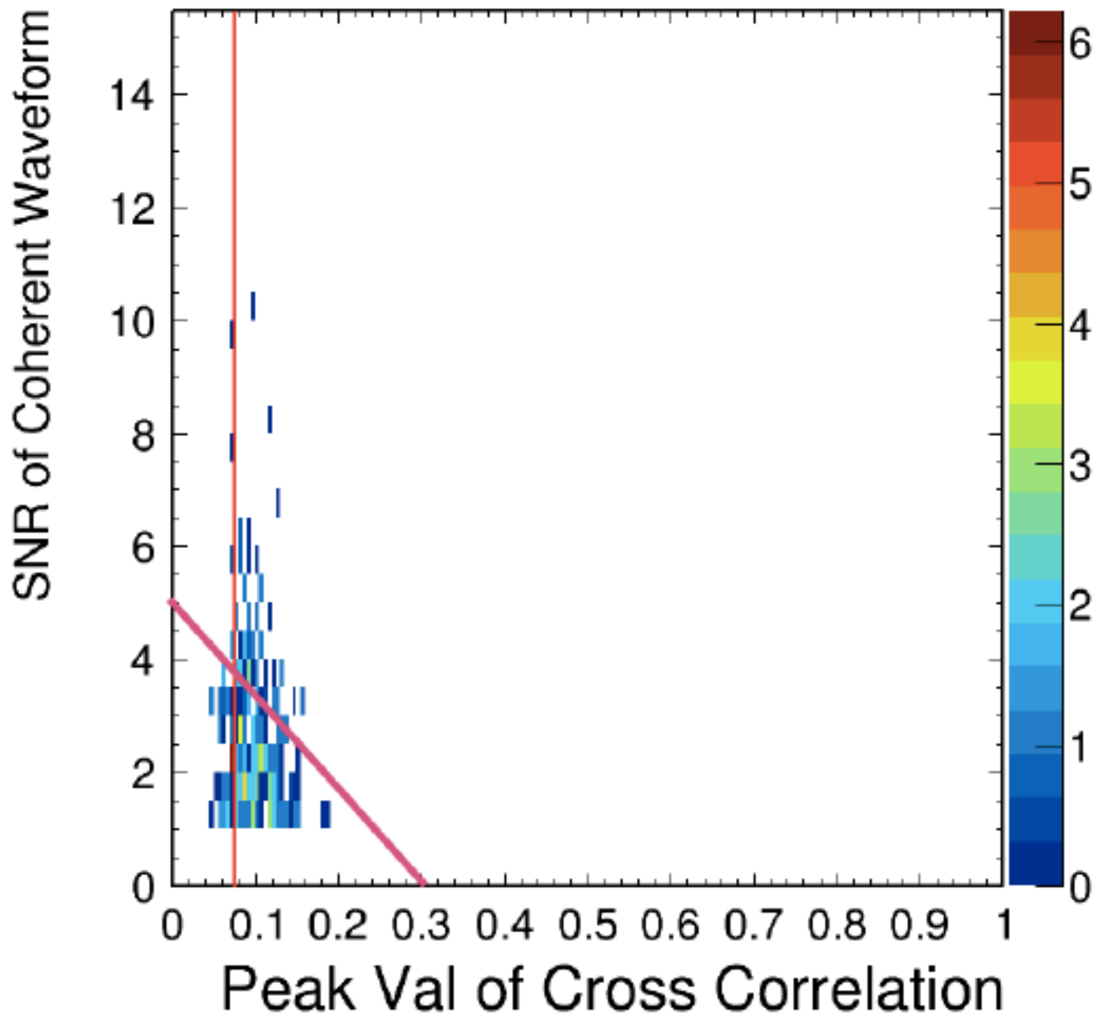

Figure 2: A plot of the Signal to Noise Ratio (SNR) vs. Cross Correlation for a particular Antarctic bin. The rotated cross correlation cut is the oblique line in red. From Brian Dailey [3]

The rotated cross correlation cut uses two variables: the signal to noise ratio (SNR) and the cross correlation of the signal. By choosing a y-intercept and a slope, an oblique line will cut the data points into two categories: those that are removed (below the line) and those that pass (above the line). After the slope is chosen, the y-intercept is increased, and each time the y-intercept increases, the number of events that pass the cut are counted. Based on the log likelihood distribution and the p-value of the chosen slope and y-intercept, this information can be used to find the best y-intercept and slope. More information on this process can be found in [3].

This analysis method is fairly trial and error based, with the best result found by trying various slopes and y-intercepts and seeing which one results in the best fit. It also assumed the best distinction between the populations is a straight line, which may or may not be true. The method presented in this paper instead works to find a more thorough method of modeling the background events. In particular, the goal is to use a machine learning algorithm to model the background with a three dimensional function based on the same two variables that were important for Dailey's work: SNR and cross correlation. Because machine learning can be incredibly powerful, it is thought that utilizing it will allow a more complete model of the background to be realized. This model can then be carried through the next analysis steps and potentially result in a better limit being set on the neutrino flux.

3.0. Introduction to Machine Learning

3.1. Motivation behind Machine Learning

Many particle physics experiments, especially those conducted within colliders, create their own background events and are able to then model the background events that they see. The controlled environment and the sheer number of events make it straightforward to cut out background events from extraneous particles. However, even in this environment, machine learning techniques such as boosted decision trees are utilized in order to eliminate backgrounds.

The ANITA experiment has a very different background to model, due to a high level of anthropogenic noise. Antarctica is an active site for many experiments, and because people use radio as a method of communication fairly regularly, activity throughout the continent can affect the quality of the data that is collected. Because of this, a more sophisticated method of modeling the background is necessary to filter out background events. It is this reasoning that has led to machine learning.

3.2. Machine Learning Packages

Before beginning to model the background, two potential machine learning algorithms were investigated to determine which one would be the right fit. Each of them are discussed below:

3.2.1. The Toolkit for Multi Variable Analysis (TMVA)

TMVA is a machine learning algorithm that is provided as part of the ROOT software package, which is commonly used as a data analysis and plotting program for many high energy particle experiments [8]. TMVA takes advantage of the vast libraries that ROOT has created, and uses them to create multiple methods of analyzing data. Included in the TMVA package are linear and functional fitting algorithms, boosted decision trees, neural networks, and many other well known machine learning tools. TMVA is capable of solving two types of machine learning problems, called “regression” and “classification”. Regression problems try to create an fit that will model a distribution of data, while classification problems try to distinguish between two separate categories of data.

One benefit of TMVA is that it is well developed and has many different machine learning algorithms available. However, it requires many specific inputs that must be set correctly in order to begin analysis, which is prohibitive for rapidly beginning analysis. In addition, it will not output its algorithm in a functional form, except in specific scenarios which require the functional form to be determined by the user beforehand. Getting a randomly generated function from the machine learning algorithm is one of the main goals of this project, because the functional form will allow further mathematical analysis in the next steps of the project. Because of this, a different machine learning program was investigated, called Karoo GP.

3.2.2. Karoo GP

Karoo GP is a genetic programming algorithm written by Kai Staats, a former researcher with the Square Kilometer Array (SKA) and current researcher with LIGO. The program tries to emulate an evolutionary style of function growth, in order to find the best function that fits a certain scenario. In order to do this, sets of functions called "generations" are randomly produced based on expressions fed to Karoo GP by the user. Popular expressions include addition, subtraction, multiplication, division, exponentials, square roots, logarithms, and trigonometric functions. Each function can include many combinations of the expressions listed above. For example, in order to fit data with three independent variables, called "a", "b", and "c", the function tree shown in Figure X might be produced:

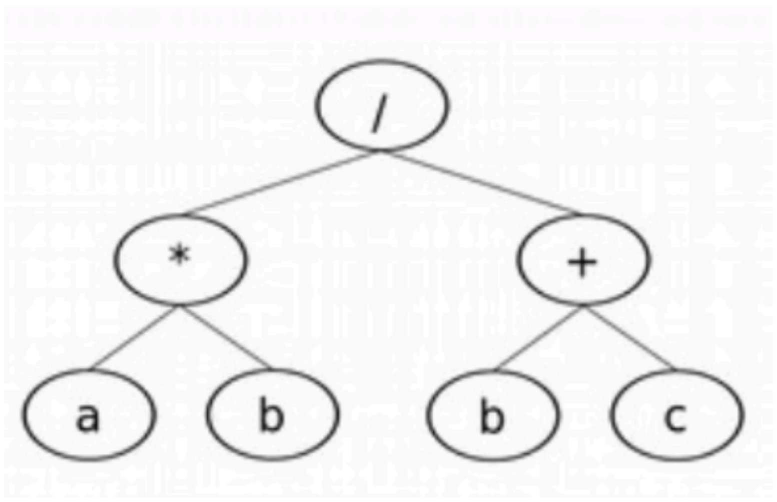

Figure 3: A function tree with depth 2 using simple arithmetic functions: (a*b)/(b+c). From [7]

Each function is tested by calculating a fit score that describes how well the data matches the function. For regression problems, the fit score is the difference squared of the predicted answer from Karoo and the expected answer from the data; thus, a low fitness score is considered a better fit for the function. For classification problems, the fitness score is simply the total number of the data points that are classified correctly, so a high fitness score means that the function fits the data correctly.

The fitness scores are calculated for each function in a generation. After the generation is created, Karoo GP will select a random subset of those functions to face off in what is referred to as a "tournament selection". In a tournament selection, the function with the best fitness score will go on to parent the next generation. This tournament selection process is repeated until there are enough parents to populate the next generation.

There are four main ways that functions can parent the next generation. The first is "reproduction", in which the function that wins the tournament selection is exactly copied into the next generation. The second is "point mutation", in which a single data element of a function is swapped out for another data element. For example, a function of the form a*b/(b+c) might undergo a point mutation and become a*c/(b+c). The third method is a branch mutation, in which an entire branch of a function is changed. An example of this would be changing the function a*b/(b+c) into a function of the form a*b/(a-c), in which the second half of the function has completely changed. Finally, there is the "crossover" method, in which two functions are selected through the tournament selection and their branches are switched. An example of this process is shown in Figure 4:

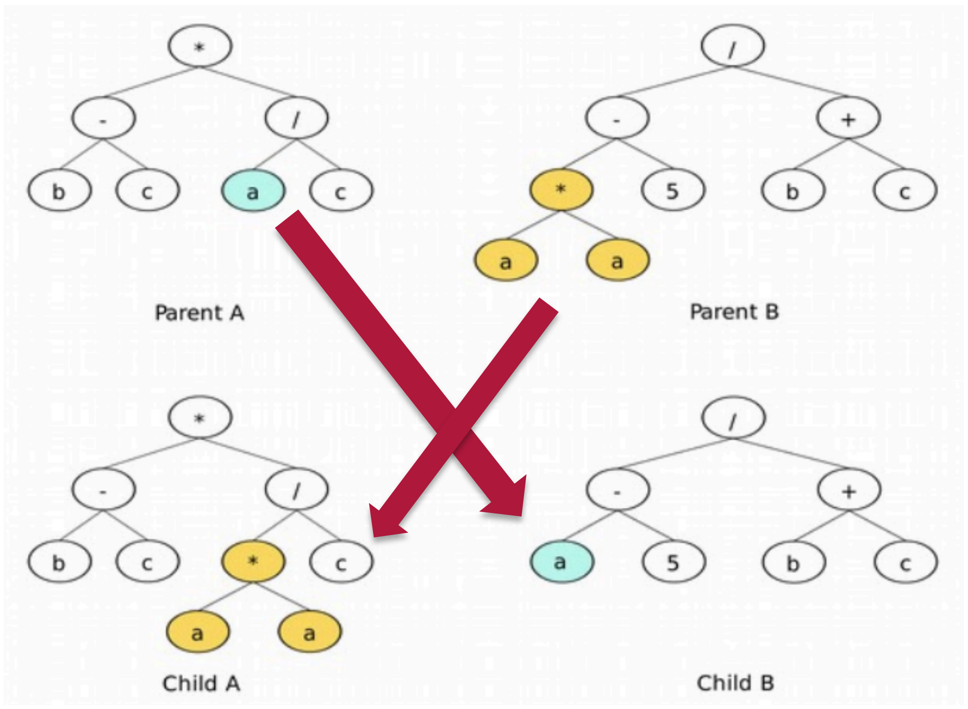

Figure 4: An example of a "crossover" method of parenting new functions. From [7]

In order to successfully create a model, the data sheet fed to Karoo GP must be formatted correctly. Karoo GP accepts data in the form of a CSV file, in which the header of each column is the name of the variable, and the final column is the answer that you expect. For regression problems, this final column is the value of the function that Karoo is trying to guess. For classification problems, the final column is either "0" if it is of one species, and "1" if it is of the other. The CSV file can include as many independent variables as necessary, although more variables will lead to an increase in run time.

The CSV file can also be modified to include two other components: constants and features. Constants allow the function to become more specific, and make it easier for Karoo GP to find offsets in the data. By adding five constants (specifically 0.1, 0.2, 0.3, 0.4, and 0.5) to the top of the CSV file, Karoo GP will randomly select and combine them in the same way that it treats the data variables. This actually allows Karoo GP to find very detailed and specific constants, which significantly improves the fit of the function.

The other component that can be included in the CSV file is called a feature, which is a combination of variables and constants. Features are selected after running through Karoo GP multiple times and determining which expressions are commonly used in the best functions. For example, if Karoo GP keeps coming up with a function that includes "$a^2 + 3$", the expression "$a^2 + 3$" can be added as a new variable in the CSV file. This way, Karoo GP can more easily find the expressions that are a good model for the data.

Depending on the complexity of the problem, it can take Karoo GP anywhere from 10 to 100 generations before it settles on an answer to a given problem. One of the benefits to genetic programming is that the answer it finds is always a little different, which allows continual improvements to be made each time it is run over a dataset. However, Karoo GP also has some drawbacks. At the moment, it is not equipped to put constants within the expressions; for example, it can find "sin(x)" but not "sin(x+2)". This can make it difficult to fit data that has a peak that is not centered at the origin. This feature was deeply ingrained into the structure of Karoo GP and would have required an extensive rewrite of the hard code; in the interest of time, this was not changed. In addition, if the first generation of Karoo GP is not seeded correctly, it will get stuck in a loop of poorly fitting functions that never really improve.

However, even with some noticeable negatives, it was decided that Karoo GP would fit the purposes of this project.

4.0. Methodology

This project focused on a particular Antarctic bin from Brian's thesis, Bin 3048, as a first test to see whether genetic programming would be capable of modeling the background at all. The contents of the bin are shown below in Figure 5:

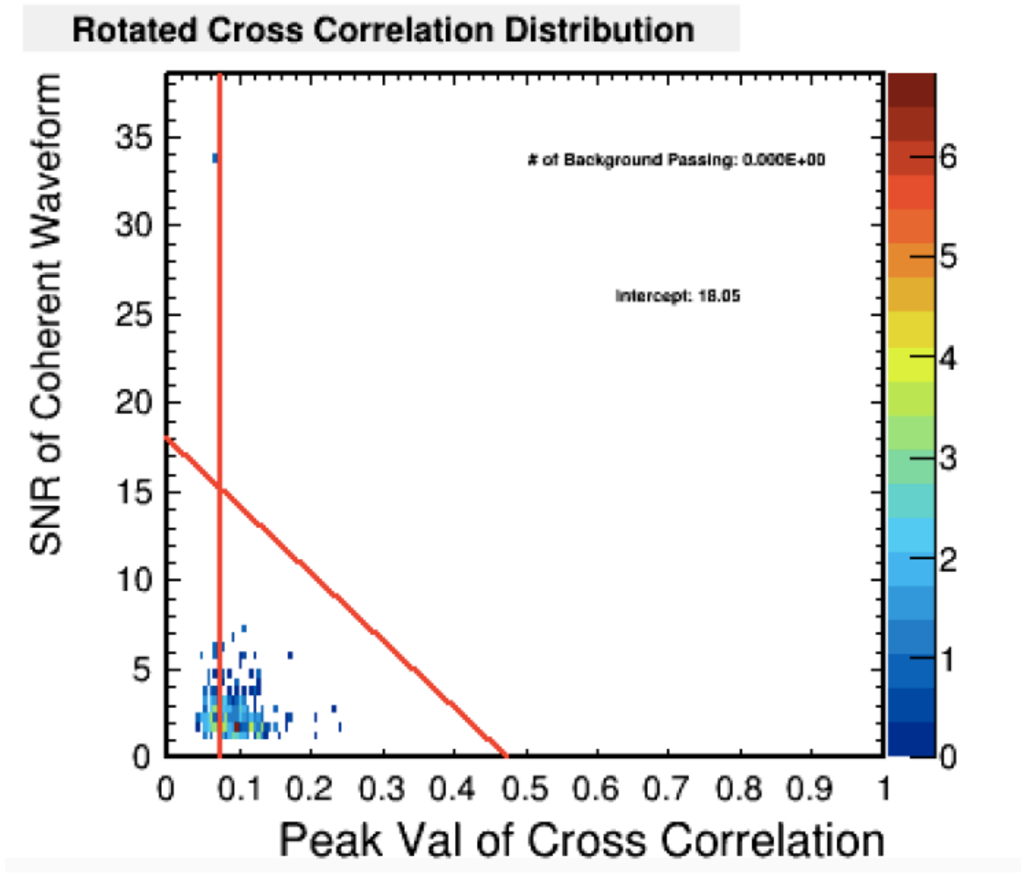

Figure 5: A plot showing the SNR vs. the Cross Correlation for Antarctic Bin 3048. From [3]

In order to use the regression algorithm in Karoo GP, there needs to be an "answer" for Karoo GP to test against. In other words, a value for the background must be provided so that Karoo GP can check its answer as it tried to come up with a function that describes the background. Therefore, the data was binned, so that Karoo GP could try to guess the bin content as the predicted value for the background. Below in Figure X, the same data from Figure X is repeated, except this time the color refers to the number of events in the bin, and not the weight.

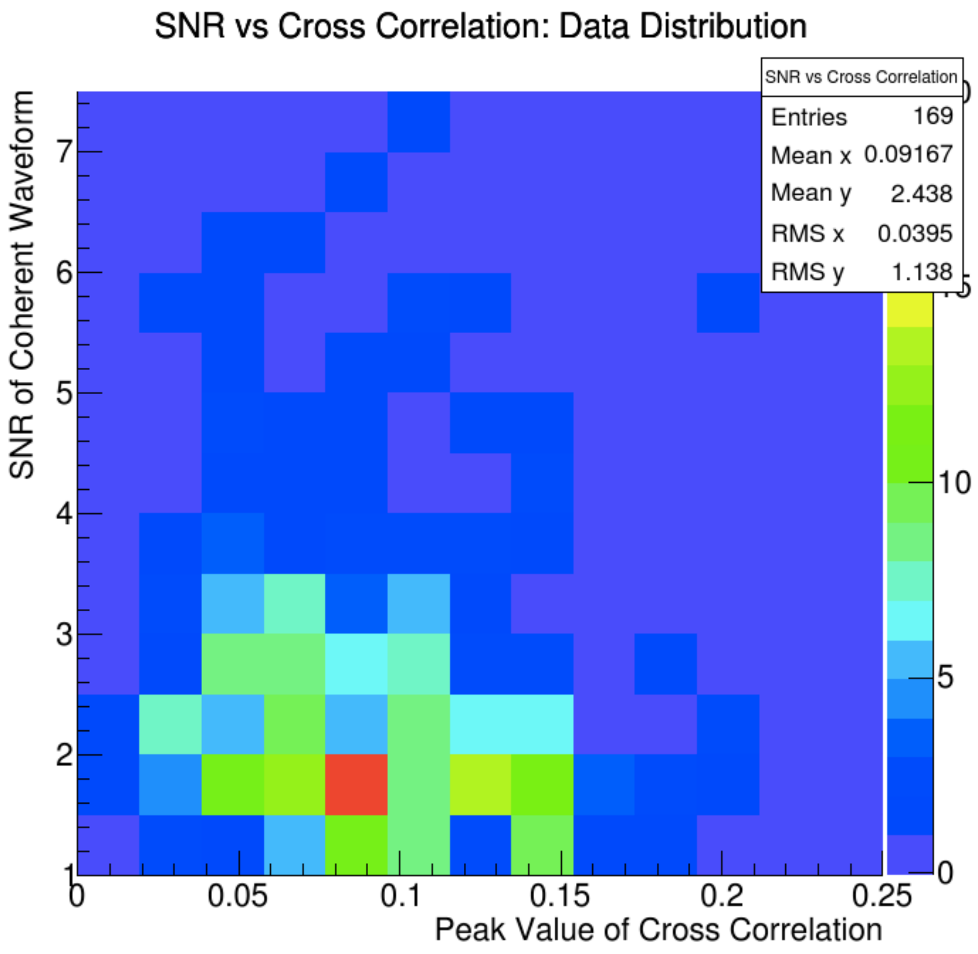

Figure 6: Binned Data Distribution from Antarctic Bin 3048

For the first attempt at fitting this data, the data was sent through Karoo GP exactly as pictured above. There were 270 events binned into 169 bins. After running Karoo GP multiple times, the best-fit function was the following:

$$Background\ Bin\ Value = pval * e^{-pval} * \cos(pval)\,/\,\sin(pval) - \cos(SNR)$$

The following figures show this function visually:

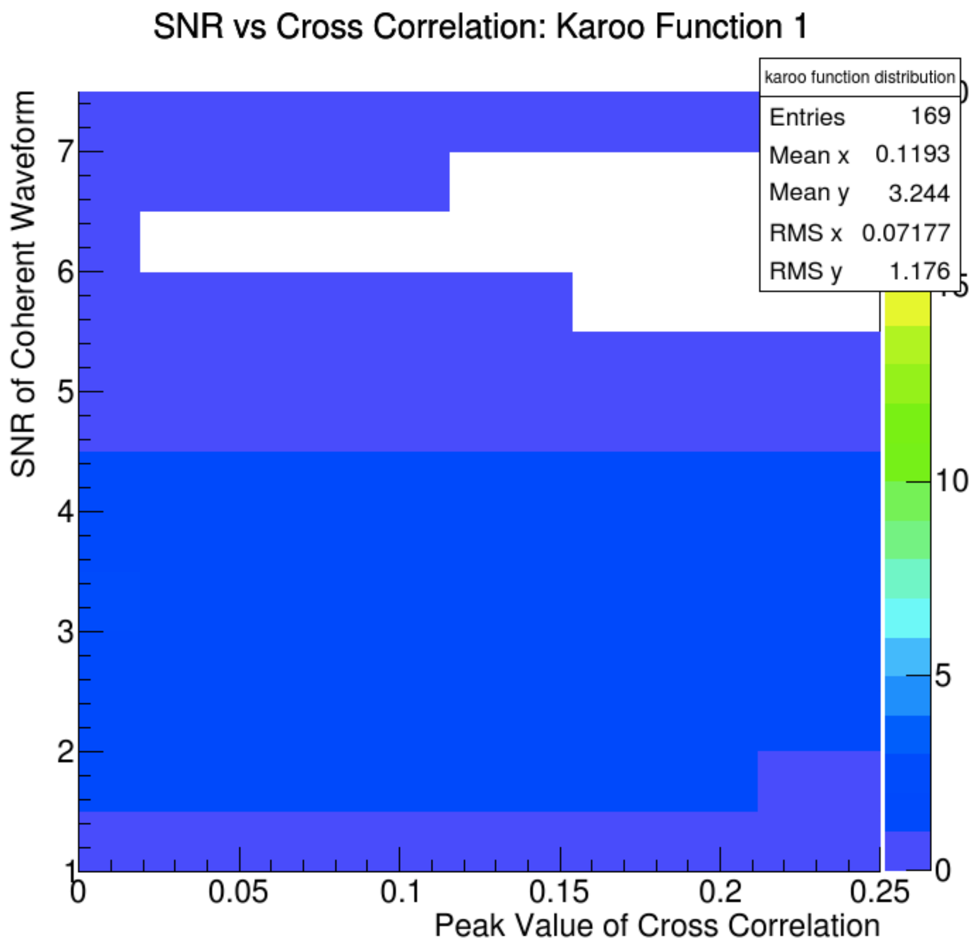

Figure 7: Plot of Predicted Bin Size from Karoo GP Function #1. The color axis in this figure is intentionally identical to Figure 6 for easy comparison.

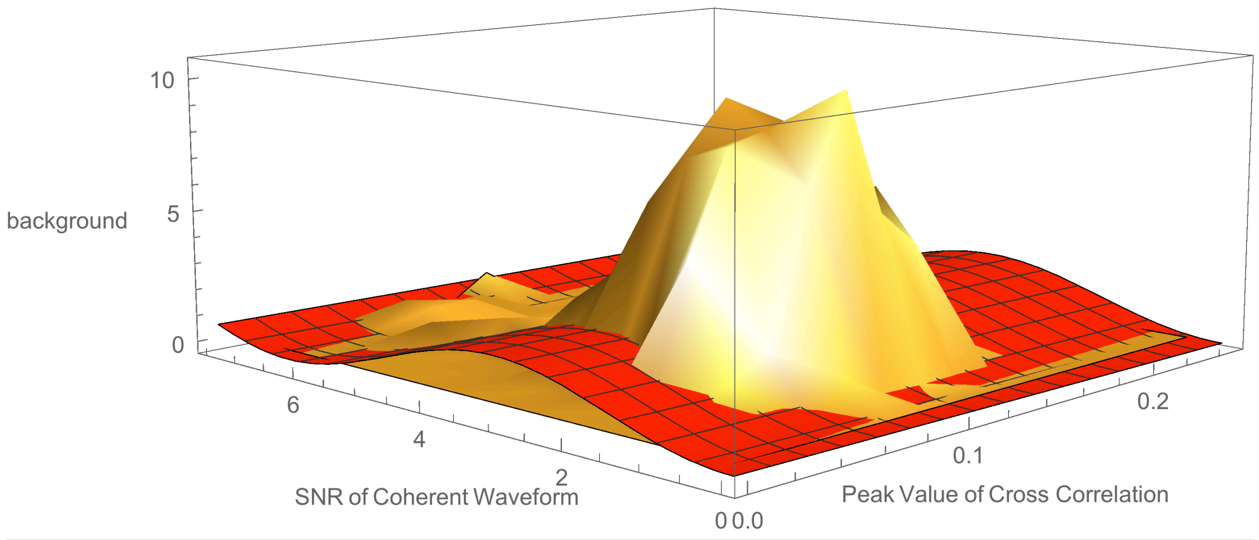

Figure 8: Comparison of the data (yellow) and the Karoo GP function (red).

Clearly, this function does a poor job of fitting to the data. The peak is only present in one of the variables, and the peak in the function never reaches the same height as the data does. In order to help Karoo GP find a better fit, some modifications were made to the data:

1. Increase the range of the Cross Correlation value: normally the cross correlation value is measured from 0 to 1. However, this makes it incredibly difficult for Karoo GP to correctly find a function that has any noticeable change along the x-axis, because most functions are not diverse enough in this range. Therefore, the cross correlation values were multiplied by 30 so that the range of the cross correlation matched the range of the SNR.

2. Take the logarithm of the number of events: This will solve two problems. Firstly, the peak of the logarithm will be smaller than the peak of the regular data, which could potentially make it easier for Karoo to fit, as the fit function does not have to have such a drastic change in a small range. Secondly, it is possible that during the first round, the multitude of empty bins that were fed to Karoo GP made it more likely to find a fit that was close to zero at all points. One of the side effects of taking the logarithm means that the bins with zero events become bins with negative infinity events instead. Because of this, the zero bins are not included in further analysis.

3. Adding in Constants and Features: In Chapter 3 the advantages of constants and features are discussed. Five constants were added to the data: 0.1, 0.2, 0.3, 0.4, and 0.5. In addition, various features were added as well. They are:

- SNR+pval
- SNR+3
- SNR-1.5
- pval+3

- pval-1.5

These features were chosen based on the fact that Karoo GP cannot do addition within the expressions. This is a very basic way to incorporate some of the features of addition within the expression, although this is an area in which improvement could continue to yield even better results.

After the data was reformatted, the distribution looked like Figure 9. This is mostly the same as Figure 6, except the Cross Correlation axis goes to 7 instead of 0.25, and the color axis is the log of the answer from before. The white boxes correspond to boxes that had zero events.

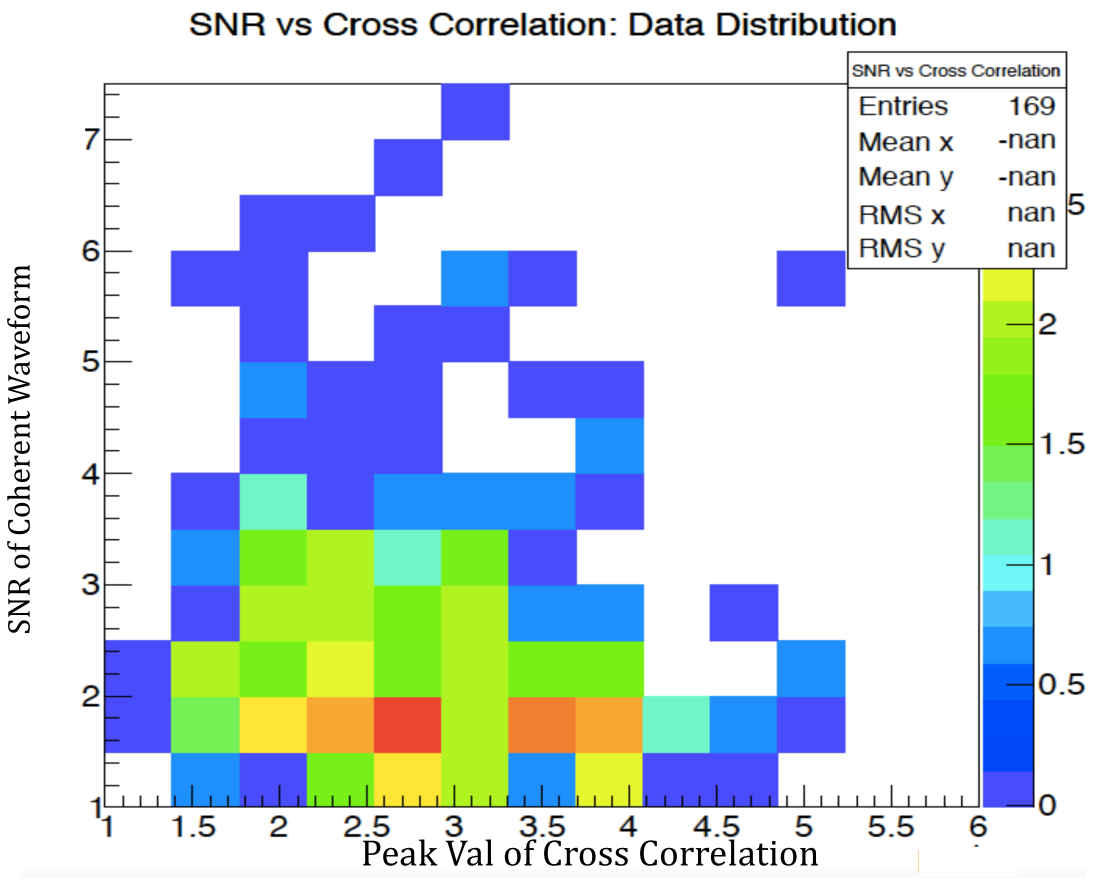

Figure 9: Background Event Distribution after changing range and taking the logarithm of the bin content.

Before feeding this data into Karoo GP, one other small adjustment was made. Because Karoo GP struggles with adding constants within expressions, it is hard for it to find peaks that are centered at random points in space. Therefore, the data was moved so that the approximate peak was at the origin. To accomplish this. 2.5 was subtracted from every value of the Cross Correlation, and 2 was subtracted from every value of the SNR.

5.0. Results

After reformatting the data as outlined above, Karoo GP was run over the data multiple times. Out of those runs, the best function is as follows:

$$Background\ Bin\ Value = 0.29 * \sin(pval + 3) - \cos(SNR + 3) * \cos(pval + SNR) - 0.819 * \cos(SNR + 3) + 0.40 - 0.71 * e^{-SNR-3} * \cos(pval + 3) - 0.88 * e^{-SNR} * e^{-SNR-3} - e^{-pval-3} * \cos(pval + 3) * \cos(SNR + 3) - 2.0 * e^{-pval-3} * \cos(pval + 3) - 0.59 * e^{-pval-3} - 0.61 * e^{-pval} * \cos(pval + 3) * \cos(pval + SNR)$$

"pval"=Peak Value of Cross Correlation

"SNR"= Signal to Noise Ratio

This function, when plotted, looks like this:

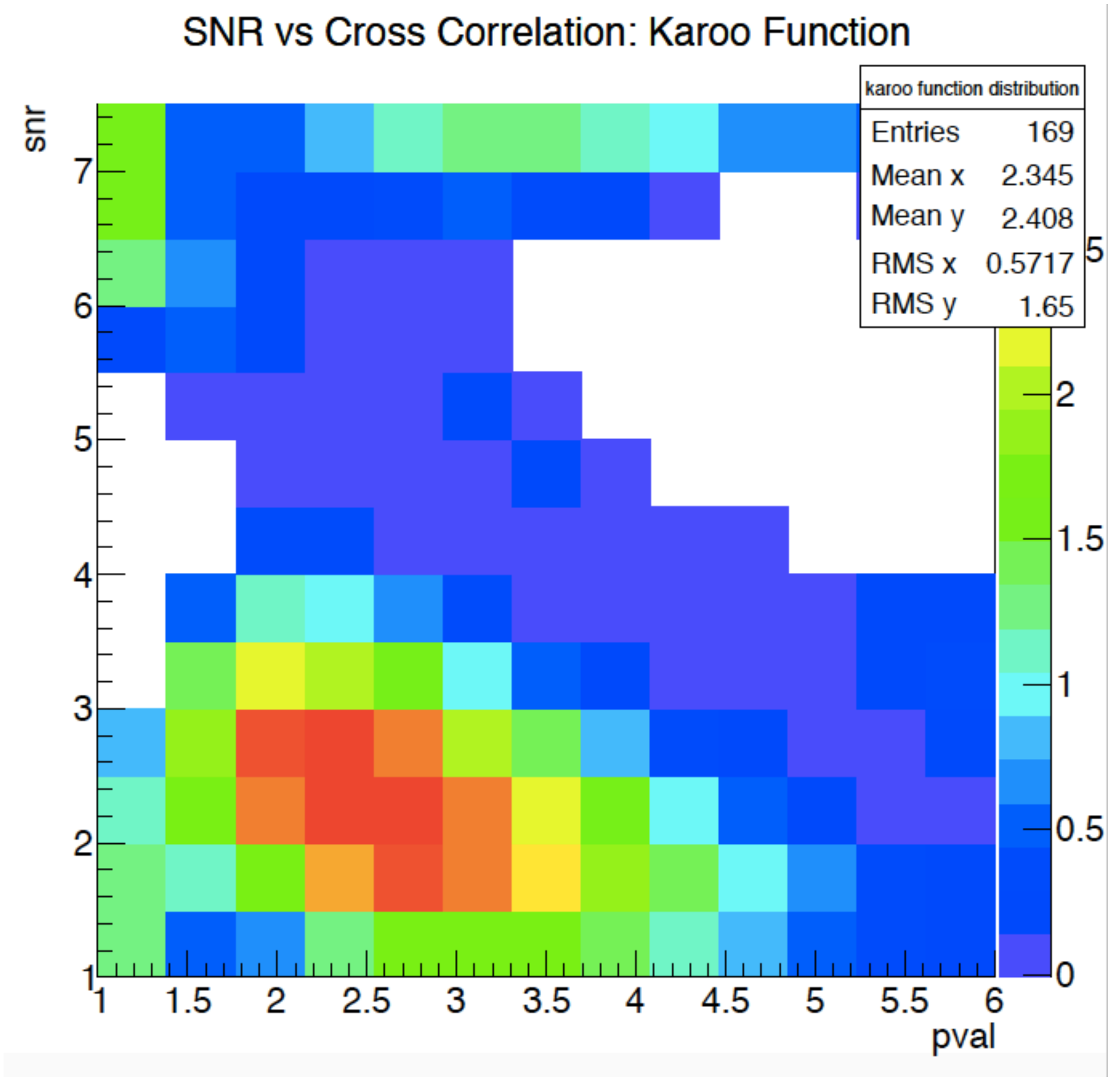

Figure 10: Logarithmic Karoo Function 2.0

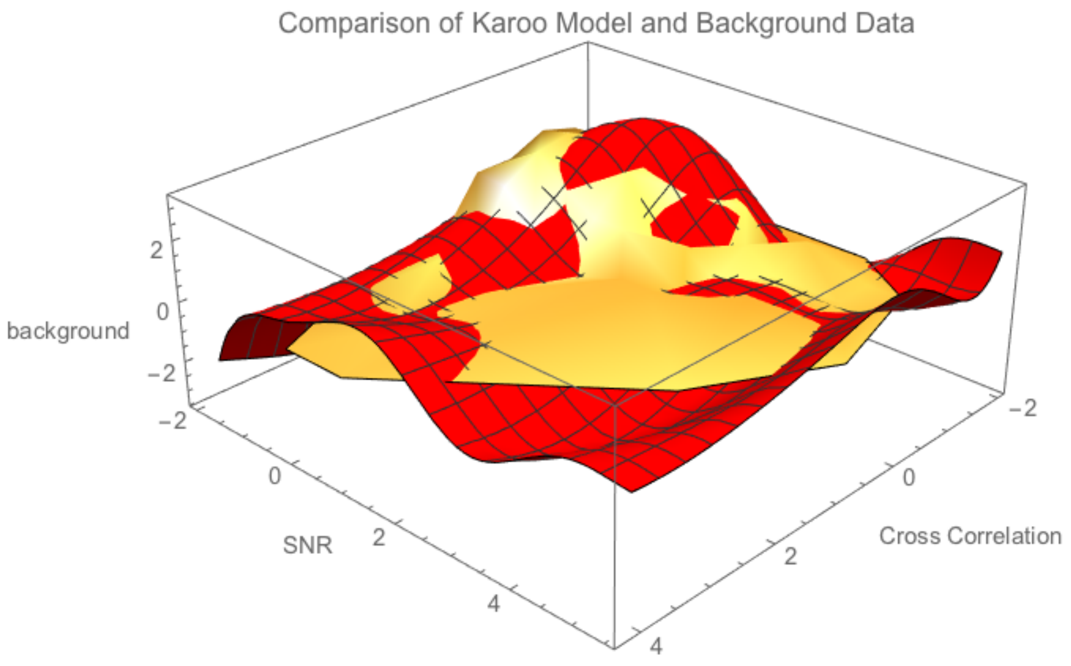

Figure 11: Comparison between background data (yellow) and Karoo GP function (red).

Qualitatively, this looks fairly similar to the data. The peak is in the correct location, and the function is clearly dependent on both the cross correlation value and the SNR. However, at the top left corner of Figure 10 there is a slight peak, which is unexpected in a model of background. Based on these factors, there are good and bad elements to this fit, and a concrete method of comparison must be utilized before making any conclusions.

In order to measure how good of a fit this function is, a log-likelihood ratio analysis was done. From the Particle Data Group article on statistics [4], the equation for this is:

$$-2\ln\lambda(\boldsymbol{\theta}) = 2\sum_{i=1}^{N}\left[\mu_i(\boldsymbol{\theta}) - n_i + n_i \ln\frac{n_i}{\mu_i(\boldsymbol{\theta})}\right]$$

In this equation, μ_i is the value from the model, and n_i is the value from the data. It can be used to test the likelihood that the model that Karoo GP found is a good model for the background data it was built from. In order for this likelihood value to mean something, the likelihood value must also be calculated from pseudo data that is modeled after the Karoo GP model. This was done by generating "pseudo-experiments" that had the same Poisson statistics as the Karoo GP model, and then using that pseudo data in the likelihood equation above as n_i. Thousands of pseudo-experiments were created and the likelihood distribution, along with the likelihood value from the actual data, is shown below in Figure 12:

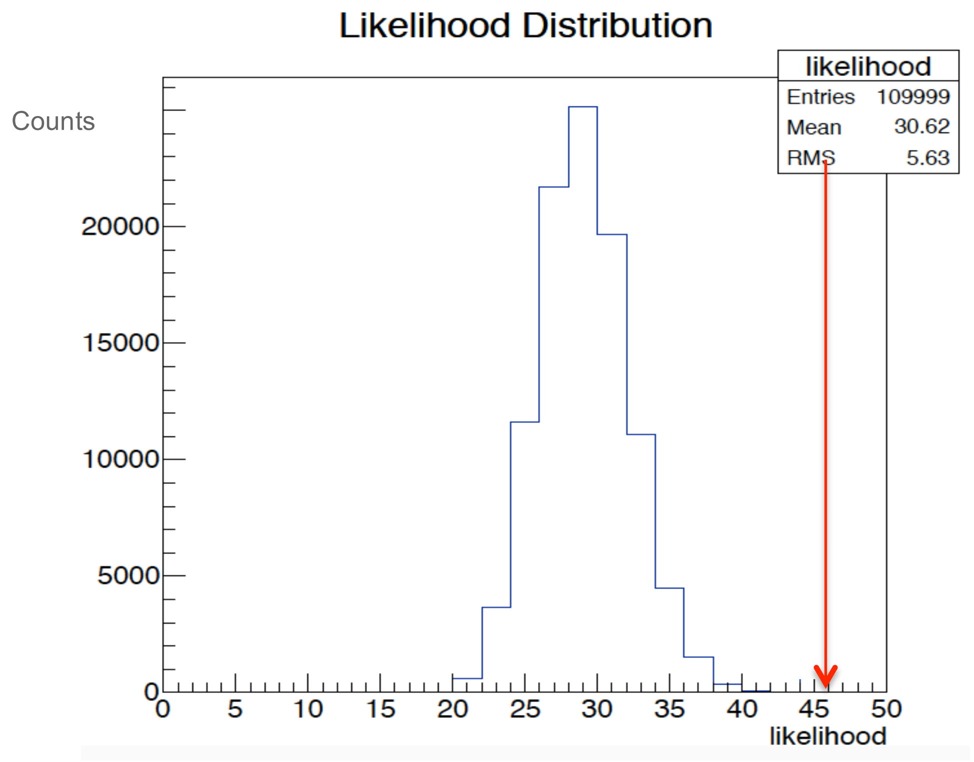

Figure 12: Likelihood Distribution for Best Karoo GP Model (blue) and likelihood compared to data (red)

As is clear from Figure 12, the Karoo GP model falls outside of the likelihood distribution, which means it is unlikely that this Karoo GP function is a good model of the data. However, this result has been noticeably improved based on the additional data formatting that happened, which is a signal that in the future Karoo GP will continue to improve.

6.0. Conclusion

Summary

A genetic programming algorithm called Karoo GP was used to model the 10% background sample from a specific area of Antarctica. Two variables, SNR and the peak value of cross correlation, were used to build a three dimensional function that described the shape of the background. Multiple improvements were made on the way that the data was input into Karoo GP. First, the logarithm of the bin values was used, and second, the data was centered on the origin. Both of these modifications show remarkable improvement in the fit of the data; however, even the best fitting model was not a good match for the distribution of the data.

Future Directions

There are multiple ways in which this project could continue to improve. Something that would have an effect right away would be modifying Karoo GP so that it could accept constants within functions, in order to handle expressions like sin(x+3). While this problem may seem simple to overcome, it actually requires a fairly extensive rewrite of how Karoo GP handles its constants. It is very likely that improving this would significantly improve the complexity and accuracy of the Karoo GP models.

In addition to new Karoo GP functions, it would be interesting to try other ways of formatting the data. For example, moving the origin to the peak of the data allowed the best fit out of any of the runs; perhaps translating to a different origin, or rotating where the axes are with respect to the data would yield interesting results.

If this method is improved upon and yields a success for ANITA down the line, it is very possible that the same types of methods could be used for the Askaryan Radio Array (ARA), another UHE neutrino experiment that is located in Antarctica. It is clear that machine learning holds significant power in the analysis world, although more work must be done to utilize it to its full potential.

References

[1] *Observation of the Askaryan Effect: Coherent Microwave Cherenkov Emission from Charge Asymmetry in High Energy Particle Cascades,* Phys.Rev.Lett. 86 (2001) 2802-2805, arXiv:hep-ex/0011001

[2] *The Antarctic Impulsive Transient Antenna Ultra-high Energy Neutrino Detector Design, Performance, and Sensitivity for 2006-2007 Balloon Flight.* Astropart.Phys.32:10-41,2009, *arXiv:0812.1920*

[3] *Analysis of the second flight of ANtarctic Impulsive Transient Antenna with a focus on filtering techniques.* Dr. Brian Dailey. PhD Dissertation, 2017

[4] K.A. Olive *et al.* (Particle Data Group), Statistics. Chin. Phys. C, **38**, 090001 (2014)

[5] ARA Collaboration. *Constraints on the Ultra-High-Energy Neutrino Flux from Gamma-Ray Bursts from a Prototype Station of the Askaryan Radio Array*. Astropart.Phys. 88 (2017) 7-16 arXiv:1507.00100v2

[6] Spencer R. Klein, for the IceCube Collaboration. IceCube: A Cubic Kilometer Radiation Detector. IEEE Trans.Nucl.Sci.56:1141-1147,2009. arXiv:0807.0034

[7] Kai Staats. Karoo GP User Guide: Genetic Programming in Python. kstaats.github.io/karoo_gp/

[8] Rene Brun and Fons Rademakers, ROOT - An Object Oriented Data Analysis Framework, Proceedings AIHENP'96 Workshop, Lausanne, Sep. 1996, Nucl. Inst. & Meth. in Phys. Res. A 389 (1997) 81-86. See also root.cern.ch/

[9] V. S. Beresinsky and G. T. Zatsepin, Phys. Lett. B 28, 423 (1969)

[10] https://arxiv.org/pdf/0909.2104.pdf

[11] https://arxiv.org/pdf/1102.0691.pdf