PROCEEDINGS OF SCIENCE

Searching for Arbitrary Low-Energy Neutrino Transients with IceCube

The IceCube Collaboration[†]
[†] *http://icecube.wisc.edu/collaboration/authors/icrc17_icecube*
E-mail: rcross@icecube.wisc.edu, segev.benzvi@icecube.wisc.edu

The IceCube Neutrino Observatory, located in the deep ice at the South Pole, is designed to observe neutrinos above 1 TeV; however, it is also highly sensitive to low-energy neutrinos from a Galactic Core Collapse Supernova (CCSN). SNDAQ, an online data acquisition and trigger system designed to observe CCSNe neutrino bursts in real time, is running with 99% uptime. In its current implementation, the time windows used by the SNDAQ trigger are tuned to typical supernova simulations and the observed signal from SN1987A. However, no galactic supernovae have been observed with high neutrino statistics so far, and many simulations do not produce an actual explosion. Therefore, it is wise to define a trigger that is not biased by simulations. To improve the sensitivity of the trigger to a much wider range of models, as well as unusual hadronic physics or physics beyond the Standard Model, we have implemented a time-domain search using the Bayesian Blocks algorithm. This technique allows the data themselves to determine the natural timescale of excess counts above background. The Bayesian Blocks window makes the SNDAQ trigger more robust to uncertainties in CCSN neutrino emission models. In addition, it also allows for general sub-threshold transient searches. We describe the implementation and performance of the Bayesian Blocks trigger and discuss improvements in the sensitivity of IceCube to supernovae in the Galaxy and its nearest satellites.

Corresponding authors: Robert Cross[1,*] and Segev BenZvi[1]
[1]Department of Physics and Astronomy, University of Rochester, Rochester, NY, 14627, USA

35th International Cosmic Ray Conference — ICRC2017
10–20 July, 2017
Bexco, Busan, Korea

*Speaker.

© Copyright owned by the author(s) under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (CC BY-NC-ND 4.0).

PoS(ICRC2017)936

1. Core-Collapse Supernovae

Core-Collapse Supernovae (CCSNe) are among the most violent events in the Universe. CCSNe can be classified as any of Type Ib, Ic, or II, depending on the composition of the star. As the fuel within a massive star is synthesized into heavy nuclei, instabilities can occur which causes the star to violently collapse and then explode. A CCSN can put out 3×10^{53} erg in about 10 s [1], which is 250 times the energy the sun will emit over its entire lifetime. While Type Ia supernovae are caused by a runaway thermonuclear explosion and emit mostly electromagnetic radiation [2], all current models of CCSNe agree that roughly 99% of the gravitational energy of the collapsing star is released through neutrinos. On February 24, 1987, 24 neutrinos were detected around the world within a 13 second interval from `SN1987A` [3]. Hours later, the event was seen in the optical regime. Most of our current understanding of CCSNe is based upon the `SN1987A` neutrinos.

The detection of neutrinos from `SN1987A` prompted the development of CCSN theory, and the neutrinos from this event continue to be used to constrain models of core collapse. In addition, the scientific community has been building detectors waiting for the next supernova. It is estimated that only 3 galactic CCSNe happen every century [1], giving rise to thousands of neutrinos in a detector such as IceCube. With this information, we could measure the luminosity and time scales of the different stages of a core collapse and begin to differentiate between several CCSN models. In addition, we could combine pointing measurements from other detectors and localize the source. Finally, depending on the strengths and shapes of the signals we see, we could even probe the neutrino mass hierarchy [4].

The exact mechanisms behind CCSNe are currently unproven. The prevailing theory, based on 1D and 2D models, only tells us the broad sequence of events that occur during a CCSNe. The actual neutrino luminosity seen in a CCSN depends significantly on the mass of the progenitor star, the neutrino production model, and the distance to the source. Whatever method we use to detect a CCSN needs to handle a large range of unknowns.

2. Detection of Neutrinos with IceCube

The IceCube Neutrino Observatory [3] is a 1 km^3 water ice Cherenkov detector constructed at the geographic South Pole. The detector was constructed by drilling 2.5 km holes into the Antarctic ice sheet and lowering 86 cables (known as "strings") instrumented with photosensors into the ice in a hexagonal grid layout. Each string is spaced about 125 m apart and each contains 60 Digital Optical Modules (DOMs) vertically separated by 17 m, for a total of 5160 DOMs. The DOMs are located between 1450 m to 2450 m below the surface of the ice. Each DOM contains a 10" hemispherical Hamamatsu R7081 photomultiplier tube and a complete autonomous data acquisition system.

IceCube is designed to detect TeV neutrino events; however, the neutrinos expected from CCSNe have energies of $\mathcal{O}(10\,\mathrm{MeV})$. At these energies, the neutrinos will produce on average less than one photoelectron per DOM. However, with a sufficient number of low energy neutrinos interacting in the detector volume, a Galactic CCSN will produce a correlated rise in the dark rate of all of the DOMs.

PoS(ICRC2017)936

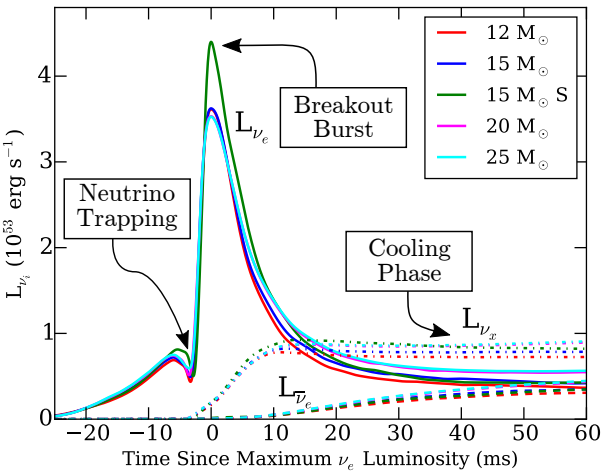

Figure 1: Neutrino luminosity of the breakout burst for five different simulated models before undergoing neutrino oscillations [4]. The solid lines show ν_e luminosity, the dashed lines show $\overline{\nu_e}$ luminosity, and the dot-dash lines show the luminosity of all other neutrino flavors. The point at which the core becomes dense enough to trap neutrinos can be seen as a small dip before the large spike in emissions at the breakout burst, after which neutrinos of all flavors are produced, and the overall luminosity decreases over the course of about 10 s as the remnant cools.

3. Supernova Trigger System

Because neutrinos from CCSNe are too low in energy to trigger the IceCube detector, we use the hit rates in the photomultipliers to search for supernova signals. The electronics in the DOMs count the number of times each DOM crosses a preset voltage threshold equivalent to $1/4$ of a photoelectron. The DOMs are sampled at 40 MHz, and the threshold crossings are counted over 2^{16} clock cycles, which produces a hardware based fundamental time bin of 1.6384 ms. These rates are transmitted to the supernova data acquisition system (SNDAQ) and are rebinned to 2 ms time bins for ease of further analysis [3].

The current implementation of the supernova trigger takes the 2 ms DOM hit rates and rebins the data further to various bin widths. The bin widths are optimized to detect signals that are consistent with current supernova models which predict neutrino emission on a variety of timescales. The time bases Δt are chosen to be 0.5 s, 1.5 s, 4.0 s and 10.0 s.

For each Δt, the hit rate for each DOM i, $r_i = N_i/\Delta t$, is calculated. Figure 2 shows the distribution of rates in a single DOM. The distribution of r_i has tails and can be described by the exponentially modified Gaussian distribution. For efficient calculation online, this is approximated as a Gaussian distribution.

The data are analyzed using a sliding window centered around the bin of interest. The structure of the sliding window is shown in Figure 3. To estimate the parameters of the Gaussian background, there is a moving interval of 300 s on either side of the bin of interest. From these 300 s regions, the expectation values of the rate and standard deviation $\langle r_i \rangle$ and $\langle \sigma_i \rangle$ are computed. There is also a 30 s exclusion zone between the background rate estimation windows and the bin of interest, so that very long-tailed signals do not influence the estimation of background rates.

The likelihood of a model which embodies a "collective rate deviation" that measures the

PoS(ICRC2017)936

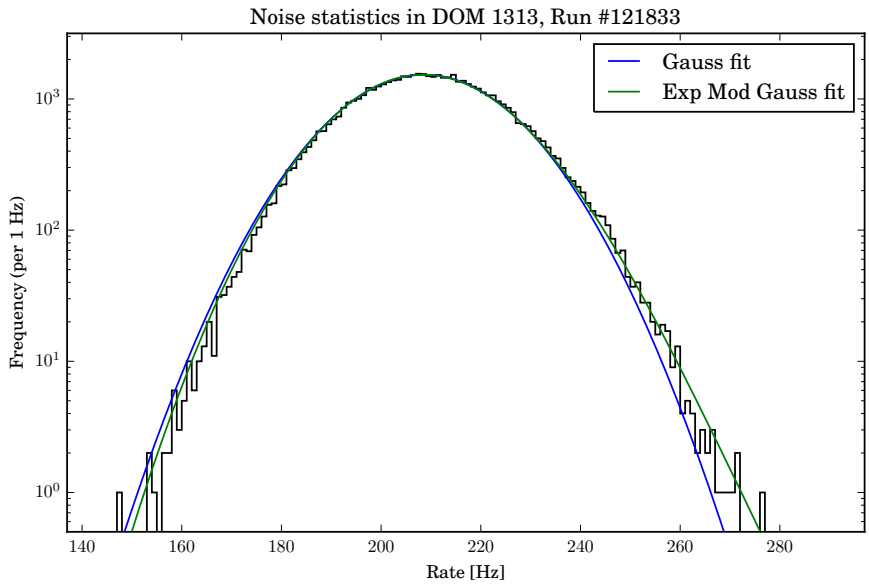

Figure 2: The distribution of the noise rate within a DOM. It is well described by the exponentially modified Gaussian distribution, but is approximated to a Gaussian to calculate likelihoods efficiently online.

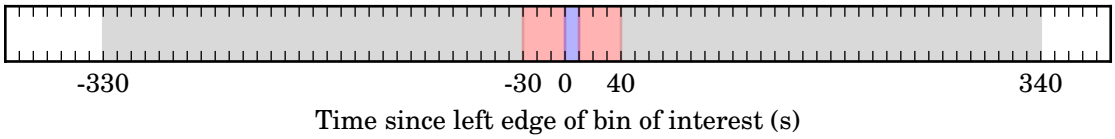

Figure 3: The structure of the sliding window in the current online analysis system. The blue zone in the center is the bin of interest. In this case, the time base Δt is 10 s. The red areas surrounding this bin represent the exclusion zone. These data are not used, so that heavy-tailed signals do not distort the background estimation. The grey areas show the data that are used to estimate the parameters for the distribution of background rates. The white areas show data outside the window. After the calculation, the window is shifted to the right by the timebase Δt.

isotropic rate deviation of all DOM noise rates r_i from their expectation values $\langle r_i \rangle$ with a DOM-dependent correction factor ε_i is

$$\mathscr{L}(\Delta\mu) = \prod_{i=1}^{N_{\mathrm{DOM}}} \frac{1}{\sqrt{2\pi}\,\langle \sigma_i \rangle} \exp\left(-\frac{[r_i - (\langle r_i \rangle + \varepsilon_i \Delta\mu)]^2}{2\,\langle \sigma_i \rangle^2} \right). \tag{3.1}$$

The likelihood is maximized for the $\Delta\mu$ parameter, and the "significance" of the bin is calculated as

$$\xi = \frac{\Delta\mu}{\sigma_{\Delta\mu}}. \tag{3.2}$$

4. Dynamic Histogramming

In this section, we discuss alternative methods to compute an optimal histogram binning which avoids the need to choose fixed values of Δt. These methods avoid the overhead of a template-based autocorrelation and are effective non-parametric approaches to observing an excess in real time.

PoS(ICRC2017)936

4.1 Methods

The act of binning data is inherently lossy since it removes information about individual data points and groups them together, making them indistinguishable. If the bin size for a histogram is chosen to be too small, the noise in the data can dominate, and the signal cannot be seen. If the bin size is chosen to be too large, then one may completely miss important features in the underlying signal.

There are many methods of determining the optimal bin width in a non-arbitrary way. Scott's rule assumes a data set with Gaussian uncertainties, and varies the bin width while minimizing the integrated mean square error [5]. The Freedman-Daiconis rule expands upon Scott's rule by relaxing the Gaussian assumption [6]. In 2006, Knuth introduced a method that maximizes the posterior probability when comparing the data to piecewise constant models with uniform bin sizes [7].

4.2 Bayesian Blocks

Bayesian Blocks (BBlocks) is a dynamic histogramming method by Scargle that extends Knuth's rule even further. Instead of assuming that the model will consist of uniform bin widths, BBlocks allows the bin width to vary. BBlocks allows as many or as few bins as are needed in order to maximize the likelihood [8, 9].

The advantages of this method are clear: nothing needs to be assumed about the signal width or shape, gaps in data are handled seamlessly, it can easily be extended to real-time triggering, and it is very fast and lightweight. No information is wasted in generating the optimal partition. This algorithm can be applied to the detection of any transient signal, which is commonly done in the context of astrophysics. As such, there is a Python implementation of the algorithm included in AstroPy [10].

4.2.1 The Algorithm

The BBlocks algorithm compares the likelihood of the data to fit any of the possible piecewise constant models, or *partitions*. A partition is composed of one or many *blocks*, separated by *change points*. A data point or measurement is called a *cell*. For simplicity, the first and last cells are considered change points.

The algorithm considers every possible partition of the data. There are 2^N ways that N cells can be rearranged into partitions. The optimal partition is found by maximizing the "fitness" of each partition. The fitness function of a block is the likelihood of the data fitting the block model. In order to find the optimal partition efficiently, the fitness function must be "block additive". In other words, the fitness function for a block must only depend on the data inside the block, and the fitness of a two block model must be the sum of the fitnesses of the two blocks independently.

The algorithm works by adding one data cell at a time and storing the best values of the fitness at every step. By doing this, data cells can be easily added and the best partitions and fitness values are updated. Once all data cells have been processed, the change points for the optimal partition are simply read out from the internal state of the algorithm. Figure 4 shows the evolution of the optimal partition as data cells are added. The algorithm adds and removes change points as needed while progressing through the data set.

PoS(ICRC2017)936

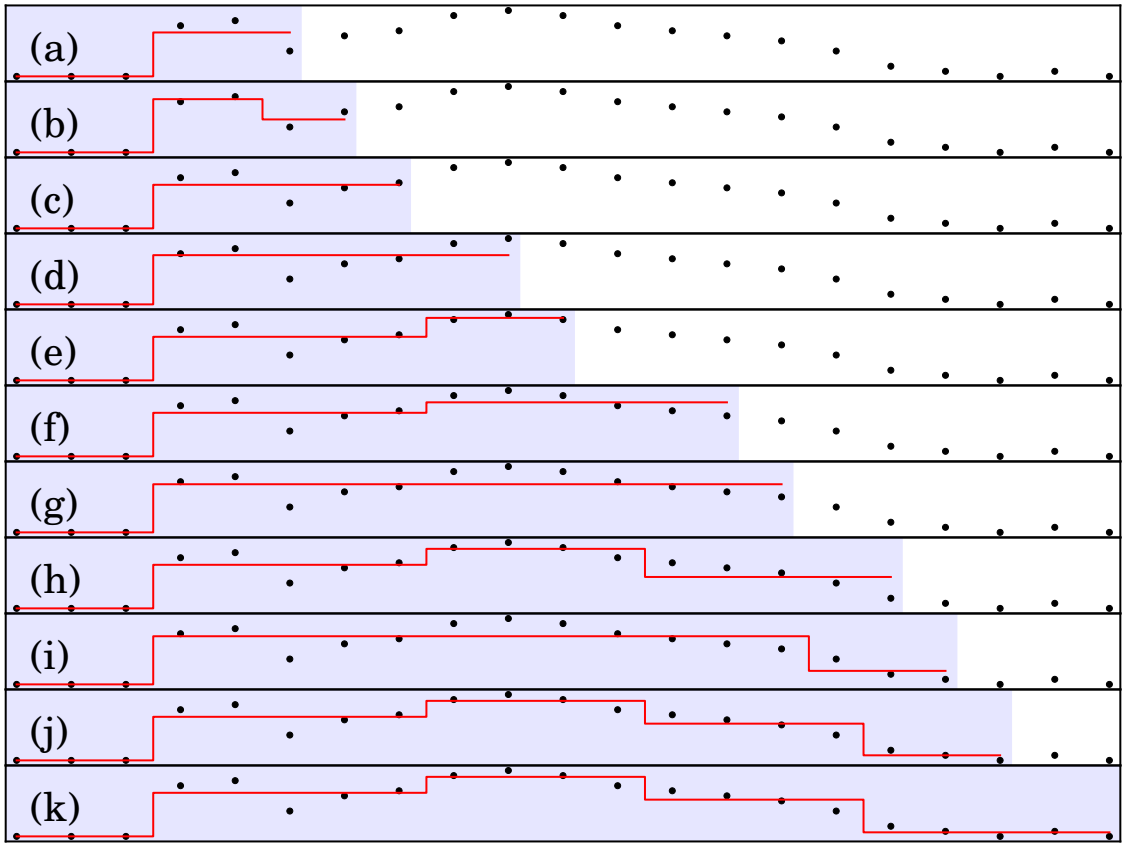

Figure 4: (a)-(k) shows the evolution of the best possible partition found by the Bayesian Blocks algorithm as data are added to the right hand side. Not all steps of the algorithm are shown. The dots show the data set used, generated by hand to showcase interesting behavior of the algorithm. The shaded area shows the data points included in the calculation at that step. The line shows the optimal partition found by the BBlocks algorithm using the included data points. At step (b), the algorithm finds that there is a secondary change point in the pulse, and at step (c), the change point is removed after another data point of information is added. You can see a similar thing happen as the algorithm goes from a 3-block model at step (i) to a 5-block model at step (j). The optimal piecewise constant model for the entire data set is found at the conclusion of the algorithm shown in (k).

By performing the search in this way, the algorithmic complexity of BBlocks is $\mathcal{O}(N^2)$. When you consider that there are 2^N choices of partitions, and that BBlocks gives an exact solution for finding the optimal partition, you can see how BBlocks can be used to detect a transient signal of any size or shape in a light curve [8, 9].

4.3 Block Fitness Function

A good choice of the block fitness function is the likelihood that the data fits the block model. For independent measurements that are approximately Gaussian, we can construct the likelihood of a given block [1].

[1] In the future, it would be interesting to implement an exponentially modified Gaussian fitness function into the BBlocks algorithm.

PoS(ICRC2017)936

The likelihood of a measurement n is

$$\mathscr{L}_n = \frac{1}{\sigma_n\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x_n-\lambda}{\sigma_n}\right)^2}, \tag{4.1}$$

where λ is the model amplitude of the block being tested. The block likelihood is therefore

$$\mathscr{L}^{(k)} = \prod_{n}^{\text{in block } k} \mathscr{L}_n = \frac{(2\pi)^{-\frac{N_k}{2}}}{\prod_m \sigma_m} e^{-\frac{1}{2}\sum_n\left(\frac{x_n-\lambda}{\sigma_n}\right)^2}. \tag{4.2}$$

Maximizing the likelihood, taking the log, and removing model independent constants [8, 9], we get

$$\log \mathscr{L}^{(k)}_{\max} = \frac{b_k^2}{4a_k}, \tag{4.3}$$

where $a_k = \frac{1}{2}\sum_n \frac{1}{\sigma_n^2}$ and $b_k = -\sum_n \frac{x_n}{\sigma_n^2}$.

4.4 Prior on Number of Blocks and Type I Errors

The number of blocks N_{blocks} is a free parameter in this algorithm, so we must choose a prior probability distribution for the parameter. By default, a uniform distribution is selected as the prior distribution on the number of blocks. This is not very useful, as it causes the algorithm to return a partition with one block per data cell, since adding more blocks to match the data would always increase the likelihood. To solve this problem, we must adopt a penalty for adding more blocks.

Scargle chooses to use a single parameter geometric prior for efficient calculations. After normalization, for N data cells, the prior is

$$P(N_{\text{blocks}}) = \frac{1-\gamma}{1-\gamma^{N+1}}\gamma^{N_{\text{blocks}}}. \tag{4.4}$$

With this choice, the log difference (ignoring the normalization, as this will be a constant in all models) in the prior is simply given by

$$\ln\left[\frac{P(K+1)}{P(K)}\right] = \ln\gamma. \tag{4.5}$$

This choice of γ effectively tunes the sensitivity of BBlocks, which can be very useful in a trigger. The rate at which the BBlocks algorithm will falsely identify a background fluctuation as a real change point is denoted p_0. By running the algorithm on large amounts of background-only simulated data, we can tune the "false positive" rate p_0 in terms of the parameter γ. Thus, BBlocks can be tuned to any desired false positive rate.

4.5 Applications to SNDAQ

Simulated data were generated from the DOM noise distributions with light curves from several CCSN models positioned at various distances from Earth. The data were analyzed using the BBlocks algorithm and the rate of true positive detections were recorded. Figure 5 shows the results of these trials.

PoS(ICRC2017)936

By raising the false positive rate p_0, we can effectively extend the detection "horizon" of IceCube to more distant sources. The cost is an increase in the rate of spurious detections, but this increase is tunable and predictable. Note that the sensitivity of the detector depends strongly on the CCSN model used.

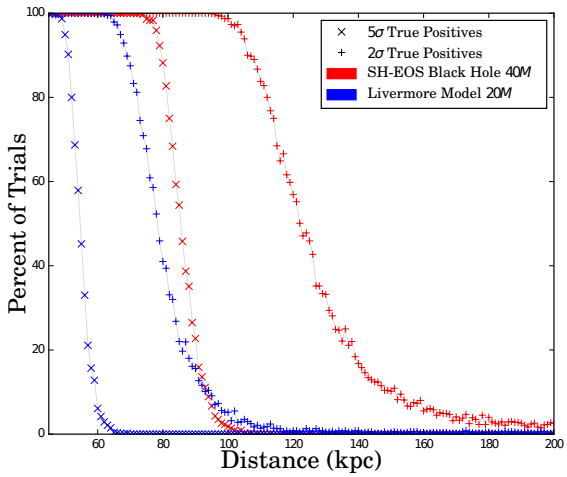

Figure 5: Plot showing the probability positively identifying a signal generated via simulated light curves at various distances. There are two curves for each model, each with the BBlocks algorithm tuned to two different values of p_0 corresponding to 5σ and 2σ significance that a detected edge is a false positive over a 500 s window. 1000 data sets were generated for each distance, and each data set was rebinned to 0.5 s.

5. Future Work

We have implemented a nonparametric algorithm which allows the data themselves to choose the optimal binning scheme based on statistically significant changes in the hit rate. This allows for a tunable error rate which can be used to increase sensitivity to distant objects, with the compromise of seeing more false positives.

Given that the Galactic CCSN rate is about one in every 30 years, it is also notable that our algorithm can be used to search for many kinds of astrophysical transients (such as Fast Radio Bursts) while we wait for the next supernova. This would be a considerable advantage over the existing real-time detection system.

We are currently implementing and deploying BBlocks in the live analysis system to provide real-time alerts. Additional technical changes are planned, such as using the exponentially modified Gaussian fitness function.

References

[1] S. M. Adams, C. S. Kochanek, J. F. Beacom, M. R. Vagins, and K. Z. Stanek, *The Astrophysical Journal* **778** (2013) 164.

[2] W. P. Wright, J. P. Kneller, S. T. Ohlmann, F. K. Roepke, K. Scholberg, and I. R. Seitenzahl, *Phys. Rev.* **D95** (2017) 043006.

[3] R. Abbasi et al., *Astronomy & Astrophysics* **535** (Nov., 2011) A109.

[4] J. Wallace, A. Burrows, and J. C. Dolence, *The Astrophysical Journal* **817** (2016) 182.

[5] D. W. Scott, *Biometrika* **66** (1979) 605–610.

[6] D. Freedman and P. Diaconis, *Probability theory and related fields* **57** (1981) 453–476.

[7] K. H. Knuth, *arXiv: physics/0605197* (May, 2006).

[8] J. D. Scargle, *The Astrophysical Journal* **504** (Sept., 1998) 405.

[9] J. D. Scargle, J. P. Norris, B. Jackson, and J. Chiang, *The Astrophysical Journal* **764** (Feb., 2013) 167.

[10] T. P. Robitaille et al., *Astronomy & Astrophysics* **558** (2013) 9.

PoS(ICRC2017)936