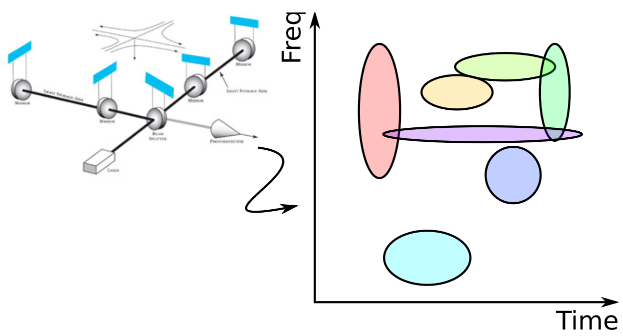
A Bayesian Coincidence Test for Noise Rejection in a Gravitational Wave Burst Search

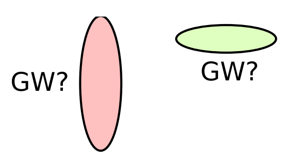
KIPP CANNON
LIGO, California Institute of Technology, Pasadena, CA (LIGO-G070881-00-Z)

Excess Power Burst Search

- Statistically significant features in the output of a gravitational wave antenna are recorded as a list of "events".

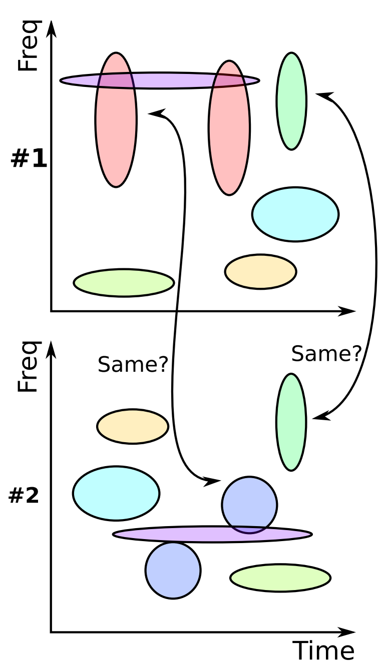


- Some events are the result of noise, and some (maybe) are a record of gravitational waves. But which are which?



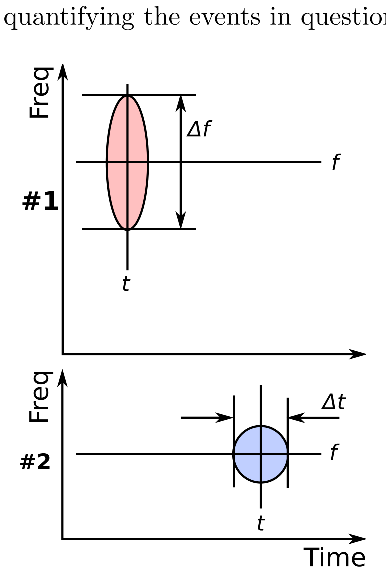
Coincidence, But How?

- One approach to rejecting noise: build 2nd, 3rd, 4th, etc., antennas and reject events that are not seen in all antennas together — retain only coincident events.
- But when has the same event been seen in the outputs of two different antennas?



Quantify a Potential Coincidence

- Begin by quantifying the events in question.



- Here, we imagine measuring the times, frequencies, bandwidths, and durations of each of a pair of events.

Standard Coincidence Tests

- The standard solution to the problem of identifying coincident events is to measure the amount of disagreement among the parameters of the events participating in the coincidence, and impose maximum allowed differences.
- For example (not realistic, for illustrative purposes only, do not attempt),

$$|t_1 - t_2| \leq 0.1\,\mathrm{s}$$
$$|f_1 - f_2| \leq 100\,\mathrm{Hz}$$
$$|\Delta t_1 - \Delta t_2| \leq 0.1\,\mathrm{s}$$
$$|\Delta f_1 - \Delta f_2| \leq 10\,\mathrm{Hz}$$

Problems

- It can take a lot of work to manually tune the coincidence thresholds.
- For example, the false alarm rate can be reduced by tightening any one threshold individually, but tightening some thresholds will result in more injections being lost than if others are tightened instead. (not all parameters are equally effective at discriminating noise from injections).

A Better Approach

- Imagine we can collect examples of coincident n-tuples that are known to be noise, and examples of n-tuples that are known to be signals, and then imagine that we have two unknown n-tuples and ask the question "which of these two n-tuples is most like the n-tuples is the signal pile?"
- Bayes' theorem answers this question,

$$P(T \in S|\vec{x}) = \frac{P(\vec{x}|T \in S)P(T \in S)}{P(\vec{x})}. \qquad (1)$$

T = an n-tuple, $\vec{x}$ = the parameters describing the n-tuple, S = the set of n-tuples known to be signals, N = the set of n-tuples known to be noise (to be seen below).

- Using $P(T \in N) = 1 - P(T \in S)$, it can be shown that $P(T \in S|\vec{x})$ is a monotonically increasing function of

$$\Lambda(\vec{x}) = \frac{P(\vec{x}|T \in S)}{P(\vec{x}|T \in N)}, \qquad (2)$$

the likelihood ratio.

- The n-tuple with the highest likelihood ratio value is the most signal-like n-tuple.

Procedure

- Begin with a standard, but loose, coincidence test to collect n-tuples.
- The standard burst search techniques of collecting n-tuples from time-slid data and n-tuples from software injections are used to collect examples of "noise" and "signals" respectively.
- Bin the parameters of the n-tuples, and measure the $P(\vec{x}|T \in N)$ and $P(\vec{x}|T \in S)$ distributions numerically.
- Using the parameters of each unknown n-tuple, evaluate and assign to the n-tuple its likelihood ratio, $\Lambda(\vec{x})$.
- Cut n-tuples, thresholding on $\Lambda(\vec{x})$.
- Cut to achieve a specific false alarm rate, or cut on the highest-ranked event for a loudest-event upper limit.
- Perform additional software injections, and assign likelihood ratios to them, apply the same cut, and measure efficiency.

Problems

- As the number of parameters being measured in each n-tuple increases, the total number of bins required for measuring the joint distribution becomes enormous.
- This problem can be addressed by identifying parameters that are statistically independent of one another, allowing the distribution density to be factored,

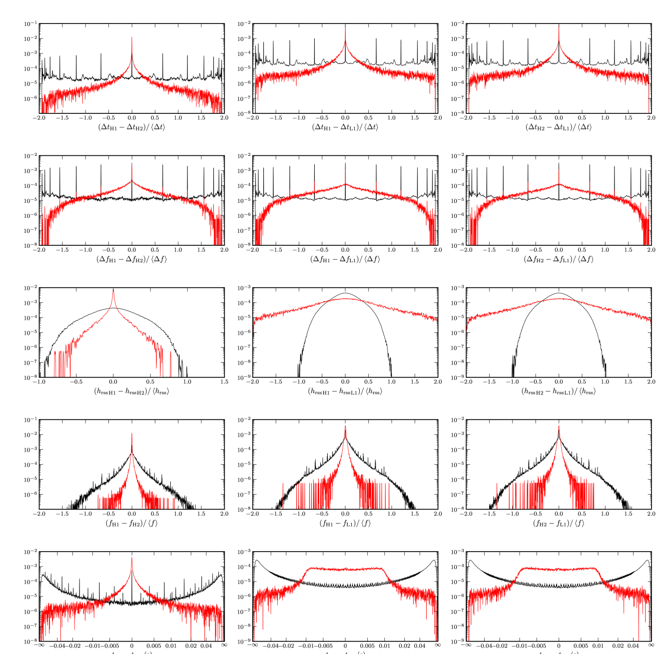
$$P(\vec{x}) \approx P(x_1)P(x_2)\cdots P(x_n). \qquad (3)$$

Advantages

- The technique is entire self-tuning, eliminating the laborious task of manually adjusting and exploring choices of coincidence cuts.
- The technique is not harmed by the inclusion of weak parameters: parameters that are poor discriminators of injection from noise.
- It is often the case that injections, or noise, or both cluster in disjoint regions of parameter space. A simple, single, threshold cannot reflect this property. The likelihood ratio technique can pick out events that fall in the good areas even when those areas are disjoint.

Example

- As an example of how things look in a "real" application, an excess power burst search was run on a network of three instruments at the locations and with the antenna patterns of the three LIGO antennas, but where the noise was simply stationary white Gaussian noise. Triggers were recovered and characterized by the four parameters given above as well as h_{rss}. Triple coincident events are, thus, characterized by 15 "deltas", which are happily found to be approximately statistically independent ((3) is true to within about 10%).



- Red = parameter distributions observed in n-tuples associated with software injections; Black = parameter distributions observed in noise (time slide) n-tuples.
- These distributions illustrate how the likelihood ratio technique is advantageous:
  - ▷ Some parameters are good discriminators of injections from noise, and some are not so good, but this is reflected in their distributions, and bad parameters contribute factors to the likelihood ratio that are close to 1.
  - ▷ In most cases, the intrinsic discreteness of the underlying trigger generator is revealed in the parameter distributions. Some parameters show clear, special, values where n-tuples, particularly noise n-tuples, cluster. The likelihood ratio technique naturally rejects n-tuples that fall in these special bins.