iDQ: Statistical inference of non-gaussian noise with auxiliary degrees of freedom in gravitational-wave detectors

MACHINE
LEARNING
Science and Technology

CrossMark

OPEN ACCESS

RECEIVED
28 May 2020

REVISED
10 July 2020

ACCEPTED FOR PUBLICATION
31 July 2020

PUBLISHED
1 December 2020

PAPER

iDQ: Statistical inference of non-gaussian noise with auxiliary degrees of freedom in gravitational-wave detectors

Reed Essick[1], Patrick Godwin[2,3], Chad Hanna[2,3,4,5], Lindy Blackburn[6,7] and Erik Katsavounidis[7]

[1] Kavli Institute for Cosmological Physics, The University of Chicago, 5640 South Ellis Avenue, Chicago, Illinois, 60637, United States of America
[2] Department of Physics, The Pennsylvania State University, University Park, PA 16802, United States of America
[3] Institute for Gravitation and the Cosmos, The Pennsylvania State University, University Park, PA 16802, United States of America
[4] Department of Astronomy and Astrophysics, The Pennsylvania State University, University Park, PA 16802, United States of America
[5] Institute for CyberScience, The Pennsylvania State University, University Park, PA 16802, United States of America
[6] Center for Astrophysics, Harvard & Smithsonian, 60 Garden Street, Cambridge, MA 02138, United States of America
[7] LIGO, Massachusetts Institute of Technology, Cambridge, MA 02139, United States of America

E-mail: reed.essick@gmail.com

Keywords: gravitational waves, detector characterization, low-latency, real-time machine learning

Abstract

Gravitational-wave detectors are exquisitely sensitive instruments and routinely enable ground-breaking observations of novel astronomical phenomena. However, they also witness non-stationary, non-Gaussian noise that can be mistaken for astrophysical sources, lower detection confidence, or simply complicate the extraction of signal parameters from noisy data. To address this, we present iDQ, a supervised learning framework to autonomously detect noise artifacts in gravitational-wave detectors based only on auxiliary degrees of freedom insensitive to gravitational waves. iDQ has operated in low latency throughout the advanced detector era at each of the two LIGO interferometers, providing invaluable data quality information about each detection to date in real-time. We document the algorithm, describing the statistical framework and possible applications within gravitational-wave searches. In particular, we construct a likelihood-ratio test that simultaneously accounts for the presence of non-Gaussian noise artifacts and utilizes information from both the observed gravitational-wave strain signal and thousands of auxiliary degrees of freedom. We also present several examples of iDQ's performance with modern interferometers, showing iDQ's ability to autonomously reproduce known data quality monitors and identify noise artifacts not flagged by other analyses.

1. Introduction

Gravitational-wave (GW) detectors, like the advanced LIGO [1] and Virgo [2] interferometers (IFOs), are exquisitely sensitive machines. This sensitivity requires complex control schemes to isolate the instruments from their surroundings [3, 4] and detailed calibration to infer the correct astrophysical strain incident on the detectors [5]. Their success, including the first direct detection of GWs [6], the now routine detection of binary black hole coalescences [7], and the detection of coalescing neutron stars [8, 9], which enabled ground-breaking multi-messenger observations [10–12], is due to a combination of the detectors' sensitivity and advanced signal processing techniques.

However, several sources of noise still limit the detectors' sensitivity. The most fundamental is stationary Gaussian noise [13, 14], which can be completely characterized by a power spectral density (PSD) and describes the detectors' behavior reasonably well most of the time. Another common noise source, referred to as *non-Gaussian noise transients* (colloquially termed *glitches* [15]), manifests as bursts of excess power in the detectors above and beyond what would be expected from stationary Gaussian noise alone. Because this is also the hallmark of a GW signal, non-Gaussian noise transients can be mistaken for real GW signals if they occur simultaneously in multiple detectors and currently limit searches' sensitivity to many expected astrophysical signals (e.g. [7, 16, 17]). Throughout the advanced detector era, an extensive zoology has been developed to categorize and mitigate the impact of non-Gaussian noise transients. This includes examining the morphology of the noise transients themselves in GW strain data as well as searching for correlations between the noise transients and other degrees of freedom that are not sensitive to GWs. Many other works have explored the former [18–20]. We focus on the latter.

Information from the detectors is recorded in a set of discretely sampled timeseries, referred to as *channels*. These channels observe many different degrees of freedom within the interferometers, including control signals used to stabilize the device [3, 4, 21, 22] and monitors of the physical environment [23]. In total, there are more than 2×10^5 channels recorded at each LIGO detector, although only $\mathcal{O}(10^4)$ are sampled at frequencies high enough to be within the detectors' sensitive band. Typically the auxiliary features used within iDQ are derived from this subset of channels, which are sampled at $\gtrsim 256$ Hz. Each channel may witness a variety of possible noise sources, some of which may couple to the measurement of GWs and some of which may not. Because of the large number of channels, it is impractical to measure the couplings between all of them directly via targeted injection campaigns. Instead, we rely on statistical correlations between channels to infer the noise's source. If channels insensitive to GWs routinely glitch in close proximity to non-Gaussian transients within the GW channel, we infer that the transients in the GW channel are mostly likely due to terrestrial noise rather than being of astrophysical origin. Figure 1 depicts a probabilistic graphical model representing this inference.

iDQ [24, 25], a statistical framework for this inference, has operated throughout the advanced detector era and continues to provide robust, real-time measures of correlations between thousands of degrees of freedom within each detector and non-Gaussian noise in the GW channel. The speed and reliability of this information has proven invaluable for several GW detections (e.g. GW170817 [8] and examples in section 5). With the expected increases in detector sensitivity and corresponding elevated detection rates over the next few years [26], real-time data quality information will only become more important.

iDQ, first described in reference [27], was developed as an extension of reference [28] in preparation for the first observing run, which began in September 2015. Although there is a long history of algorithmic development in the field, including hierarchical veto application schemes based on approximations of the likelihood ratio [29], the Poisson significance of coincident noise [30], the percentage of time witnesses remove noise [31], and applications of more general machine learning algorithms [28, 32, 33], many of these algorithms fail to produce probabilistic statements about their predictions. Furthermore, such algorithms did not operate in real-time, a key component of searches for multi-messenger astrophysical events [34]. Additionally, GW interferometers are not stationary over long periods of time; the characteristics of the noise change. This means that correlations measured by any particular algorithm at any particular time may not generalize well to data recorded later, complicating the inference process. We note that there is a distinction between non-Gaussian noise and non-stationarity. For example, non-Gaussian noise may be described by stationary Poisson processes. In fact, many algorithms make this assumption [29, 30], although it is not guaranteed to be the case.

iDQ provides a framework in which any supervised learning algorithm can be run in real-time, and several common algorithms (e.g. Random Forests, Support Vector Machines, and Neural Networks, see [28]) are already included. It calibrates their output (real numbers between zero and one) into statistical statements about the confidence that non-Gaussian noise is present in the GW channel. This is accomplished via two-class classification, amenable to many machine learning algorithms. Additionally, iDQ automatically re-trains and re-calibrates the algorithms to capture non-stationarity within the detectors. This means that iDQ autonomously adapts to new sources of non-Gaussian noise within the detectors, identifying witnesses of previously unseen noise sources and flagging data as problematic without human intervention.

We describe iDQ's formalism in section 2, including our use of supervised learning in section 2.2. Section 3 describes how we structure the inference, including how we construct vectorized representations of a detector's state (section 3.2), how we train machine learning algorithms, including ways to extract feature importance from the trained models (section 3.3), as well as how iDQ manages cross-validation (section 3.4) and calibrates its predictions into probabilistic statements (section 3.5). A few examples of iDQ's performance are shown in section 5 and possible applications within searches are discussed in section 6, including a likelihood ratio test based on first-principles noise models which account for our imperfect knowledge of the presence of non-Gaussian noise in our detectors. We conclude in section 7.

2. Formalism

We couch our statistical inference as two-class classification, which we approach within a supervised learning framework. This produces predictions for the presence or absence of non-Gaussian noise. It is worth noting that this is not the only approach, and one could instead attempt to regress the full waveform of the non-Gaussian noise based on auxiliary degrees of freedom (e.g. [35, 36]). However, classification is more

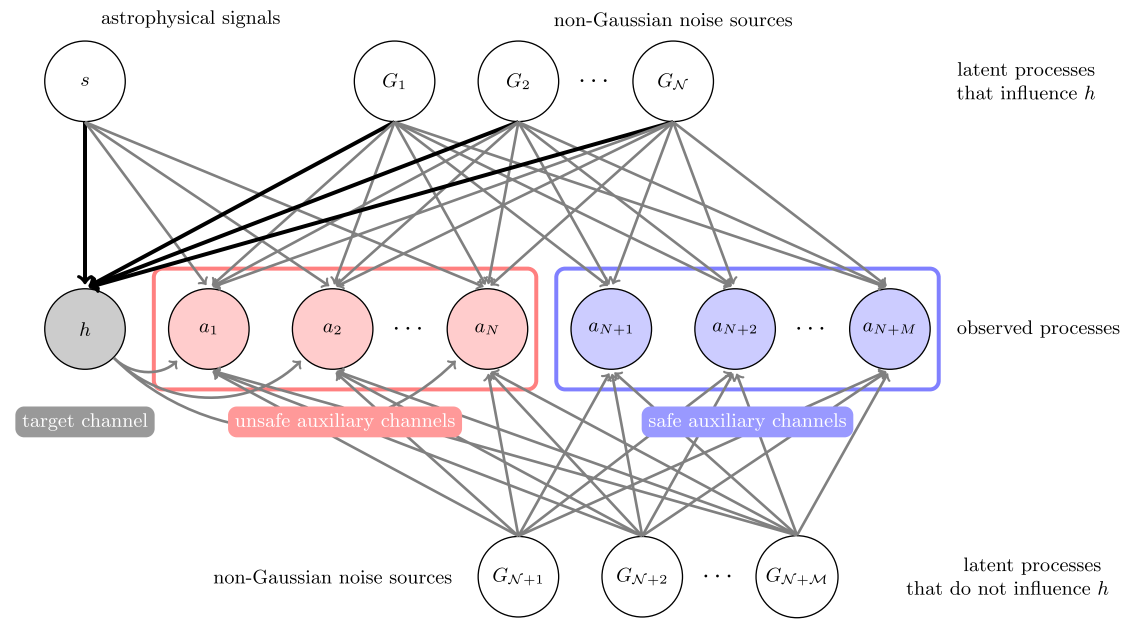

Figure 1. Probabilistic graphical model representing different sources of noise within GW interferometers. Each circle represents a different process, some of which are observed and some of which are not, and conditional dependencies are represented by directed arrows. We assume that each separate process additionally includes independent stationary additive Gaussian noise, which we omit for clarity. Latent processes, which we cannot observe directly, are shown as unshaded circles. The top row corresponds to processes that can influence the target channel (h), including astrophysical signals (s) and a subset of possible sources of non-Gaussian noise ($G_1 - G_\mathcal{N}$). Conditional dependencies from these channels to h (thick arrows) are how non-Gaussian noise couples into the target channel. The bottom row ($G_{\mathcal{N}+1} - G_{\mathcal{N}+\mathcal{M}}$) corresponds to possible sources of non-Gaussian noise that do not affect h but may affect auxiliary witnesses. Observed processes are shown in the middle row, which include h (*grey shaded circle*) as well as N *unsafe* auxiliary channels (*red shaded circles*), which may be influenced by s or h (or both), and M *safe* auxiliary channels (*blue shaded circles*), which only witness sources of non-Gaussian noise. If we marginalize over the latent processes in this graph, we will introduce conditional dependencies between h and the auxiliary channels as well as between different sets of auxiliary channels. iDQ only uses information from safe auxiliary channels to infer these induced correlations, thereby predicting the presence or absence of non-Gaussian noise in the target channel without observing the target channel itself.

tractable at this time, and we construct our inference in that framework using a vectorized representation of the detector's auxiliary state.

2.1. Statistical inference

First, a bit of nomenclature. The *target channel* $h(t)$ refers to the degree of freedom containing non-Gaussian noise we would like to identify, typically a proxy for the GW channel. This is shown in grey in figure 1. *Auxiliary channels* refer to all other channels. Because GW detectors are complicated devices that require active control, several auxiliary channels may be nearly identical to the target channel (i.e. contain signals derived from h used to control the interferometer). These are referred to as *unsafe auxiliary channels*, as they are likely to witness real GW signals, and it would be *unsafe* to use them to construct veto conditions as they could systematically veto real GW signals [31]. These are shown in red in figure 1. The remaining auxiliary channels, which are not sensitive to GWs, are referred to as *safe auxiliary channels* $\vec{a}(t)$, shown in blue in figure 1. Safety is typically determined through a series of hardware injections in which excitations are injected into the interferometer to mimic the effect of a real GW. Auxiliary channels which correlate strongly with the hardware injections may similarly correlate with real GW signals and are deemed unsafe. *iDQ only uses information from safe auxiliary channels*, although labels for supervised learning are derived from h.

To put this more formally, we assume the target channel is composed of three independent components

$$h(t) = n(t) + s(t) + g(t) \tag{1}$$

where n is stationary Gaussian noise, s is the astrophysical strain induced in the detector, and g represents non-Gaussian noise artifacts. We can only observe h and therefore model n, s, and g as latent processes which may or may not be correlated with other degrees of freedom. Furthermore, we assume n is stationary over timescales much longer than either s or g and can therefore be completely described by a single PSD.

Because we adopt a two-class classification scheme, we must define our classes. We take G to be the union of all possible types of non-Gaussian noise (G_i) without explicitly enumerating each class (as opposed to, e.g. GravitySpy's explicit multi-class classification [20]).

$$G = \bigcup_{i \in \substack{\text{glitch} \\ \text{classes}}} G_i \tag{2}$$

Typically, this is defined by a set of thresholds on h, such as signal-to-noise ratio (ρ) and a frequency range. For instance, one may target only loud, low-frequency noise relevant for high-mass binary black hole searches. G then consists of all time samples that correspond to h within these thresholds. The complement of G, referred to as C, corresponds to *clean* times when the detector does not display non-Gaussian noise such that

$$p(G) + p(C) = 1 \; \forall \; t \tag{3}$$

and

$$p(G \cap C) = 0 \; \forall \; t. \tag{4}$$

Furthermore clean states axiomatically imply $g(t) = 0 \; \forall \; t \in C$, in that there are no non-Gaussian noise transients within clean times.

Because we derive labels based on h instead of g, true signals may also fall within the thresholds defining G. However, the true signal rate is expected to be orders of magnitude less than the rate of non-Gaussian noise artifacts ($\lesssim 1/\text{day}$ as opposed to $\sim 1/\text{minute}$), and we do not expect GWs to significantly pollute our training set [8]. Furthermore, because we only use safe auxiliary channels, defined by their insensitivity to GW signals, we expect s to be independent of $\vec{a}$, and h containing GW signals is indistinguishable from h without GW signals based on $\vec{a}$ alone. Nonetheless, one could remove almost all true signals by removing any element of G that is coincident between multiple detectors [9]. This would also accidentally remove elements of G due to g that just happened to be coincident, but we expect the processes producing g to be independent in each detector [10]. Removing such a random subset of G would only decrease our sample size without biasing the training sets. However, because of the additional complexity associated with synchronizing processes running at geographically disparate locations, and the fact that the impact on our training sets is negligible, iDQ does not currently remove elements coincident between detectors from its training set.

Again, this is not the only way to construct the inference. Instead of classification, one could use the fact that ρ measures the probability that Gaussian noise alone could have produced h, and can therefore be used to estimate the probability that a non-Gaussian transient is present. Instead of classifying samples separated by hard thresholds, one could regress ρ directly or use ρ to define weighed training sets. We leave such extensions to future work, but note that similar model comparisons are implicit within our marginal-maximized likelihood ratio test (section 6.2).

iDQ infers the probability of the presence of non-Gaussian noise artifacts ($g \neq 0$) within h based on safe auxiliary channels. Specifically, iDQ estimates

$$\begin{aligned} p_G(t) &= p(G|\vec{a}(t)) \\ &= \frac{p(\vec{a}|G)p(G)}{p(\vec{a}|G)p(G) + p(\vec{a}|C)p(C)}. \end{aligned} \tag{5}$$

using supervised learning to estimate the likelihoods $p(\vec{a}|G)$ and $p(\vec{a}|C)$.

[8] We note that elevated detection rates expected with advanced detectors at design sensitivity and other planned detectors may cause this assumption to break down, necessitating further curation of training sets within our supervised learning framework.

[9] Note that a few key detections were essentially made with data from a single interferometer (e.g. GW170817 was initially detected as a a single-interferometer trigger at LIGO Hanford) and therefore requiring coincidences between detectors may not remove all astrophysical signals.

[10] The assumption of independent noise in each detector may break down in certain cases, like correlated magnetic noise due to Schumann resonances.

2.2. Supervised Learning as Dimensional Reduction

As described in section 1, GW interferometers monitor a large number of auxiliary degrees of freedom as discretely sampled timeseries recorded at different rates. iDQ represents the information in these auxiliary channels as a set of *features* extracted from each channel separately and compiled into an array of fixed dimension.

Several example feature extractors are described in reference [37, 38], and these generally rely on a wavelet decomposition to identify excess power beyond what is expected from stationary Gaussian noise alone. These feature extractors map discretely sampled timeseries into tabular data, such as the frequency, amplitude, and duration of non-Gaussian transients. iDQ constructs high-dimensional representations of the detector's auxiliary state based on this tabular data, typically recording $\mathcal{O}(5)$ features for each of $\mathcal{O}(10^3)$ auxiliary channels. Although efforts to extract better feature sets are on-going [39], iDQ implicitly assumes that features extracted in this way from each channel are sufficient statistics. Indeed, the wavelet decompositions at the core of many feature extractors form overcomplete bases and contain all information available in the original channel. There is also evidence that the precise algorithmic details of the feature extractor may not significantly impact the overall inference (see section 6.2 of reference [40]).

The Neyman-Pearson lemma [41] states that an optimal classification scheme orders samples by their likelihood ratio

$$\Lambda^G_C(\vec{a}) = \frac{p(\vec{a}|G)}{p(\vec{a}|C)} = \frac{p(C)}{p(G)}\left(\frac{p_G}{1-p_G}\right) \tag{6}$$

However we do not know the functional form of the likelihoods *a priori* and must estimate them from observed samples. Compounding this, the dimensionality of $\vec{a}$ can be very large, typically $\mathcal{O}(10^4)$ or more. This drives us to supervised machine learning as a way to approximate Λ^G_C and reduce the dimensionality to something tractable.

We therefore consider the main product of supervised learning algorithms, referred to as *classifiers*, to be a map from the high dimensional input space to the unit interval $\mathcal{M}(\vec{a}) : \mathcal{R}^{N\gg 1} \to \mathcal{R} \in [0,1]$. The precise functional form of the map is determined by the details of the algorithm and is unimportant for the rest of the inference, but we expect elements of G to be mapped to values near 1 and elements of C to be mapped to values near 0. We then construct a likelihood ratio in this lower dimensional space for each classifier separately, *de facto* estimating

$$p_G = \frac{p(\mathcal{M}(\vec{a})|G)p(G)}{p(\mathcal{M}(\vec{a})|G)p(G) + p(\mathcal{M}(\vec{a})|C)p(C)}. \tag{7}$$

Estimates of $p(\mathcal{M}(\vec{a})|G)$ and $p(\mathcal{M}(\vec{a})|C)$ are obtained by evaluating labeled samples with trained classifiers and directly modeling the resulting distributions (section 3.5 and appendix A).

Optimal classifiers will order samples according to Λ^G_C, so we expect Λ^G_C to be monotonic in the classifier's output $\mathcal{M}(\vec{a})$. Therefore iDQ also calculates the *efficiency* and *false alarm probability* as cumulative conditioned likelihoods integrated over classifier predictions

$$\begin{aligned} \text{efficiency} &= P(\mathcal{M}(\vec{a}) \geq r|G) \\ &= \int_r^1 dx\, p(\mathcal{M}(\vec{a}) = x|G) \end{aligned} \tag{8}$$

and

$$\begin{aligned} \text{false alarm probability} &= P(\mathcal{M}(\vec{a}) \geq r|C) \\ &= \int_r^1 dx\, p(\mathcal{M}(\vec{a}) = x|C) \end{aligned} \tag{9}$$

These cumulative statistics define the receiver operating characteristic (ROC) curve for a classifier, the standard metric for classification performance. Likelihood ratio tests optimize the efficiency at all false alarm probabilities.

We also note that iDQ can run multiple classifiers in parallel over the same data. Because each classifier produces a different map, and therefore may be able to better identify different subsets of glitches, we should be able to extract more information by combining classifiers. *De facto*, this would amount to using supervised learning to map a very high-dimensional space into a more manageable size, which may be amenable to direct modeling of the likelihood. This is the case for one-dimensional output from a single

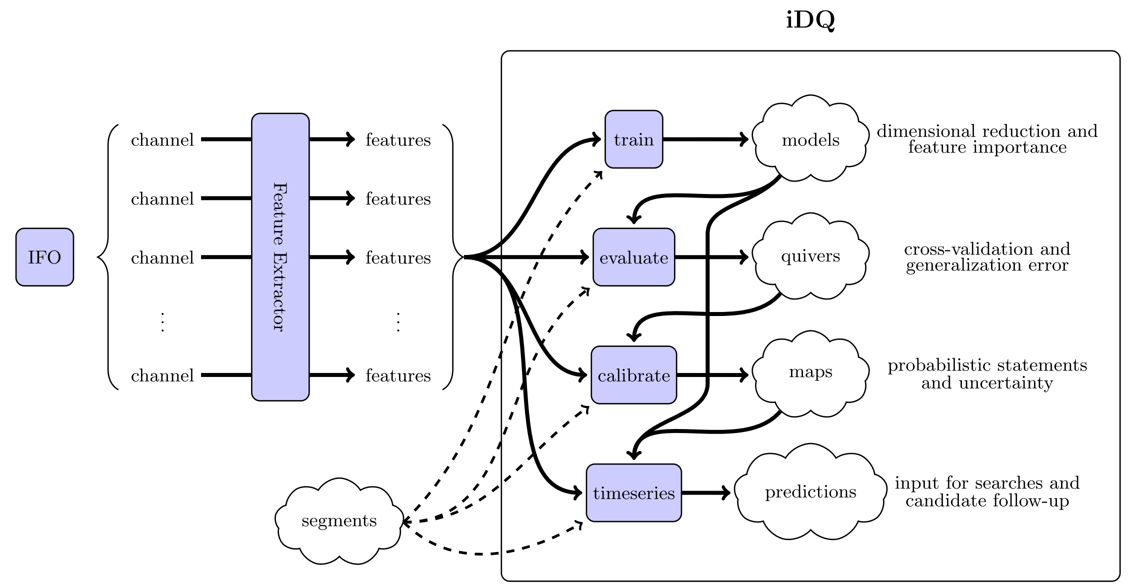
iDQ
IFO
channel
channel
channel
⋮
channel
Feature Extractor
features
features
features
⋮
features
train
evaluate
calibrate
timeseries
models
quivers
maps
predictions
segments
dimensional reduction and feature importance
cross-validation and generalization error
probabilistic statements and uncertainty
input for searches and candidate follow-up

Figure 2. iDQ's workflow. Channels are processed into feature sets $\vec{a}(t)$, which are then used in several asynchronous processes to train classifiers, calibrate their output, and produce real-time probabilistic predictions for the presence of non-Gaussian noise G in the target channel $h(t)$. The large block on the right identifies the steps within the iDQ pipeline, while the source of features and segments on the left identify input used at various points in the workflow.

classifier. Combining a handful of classifiers should not pose a more complicated conceptual issue, although running enough classifiers in parallel may require us to use supervised learning again to model the joint likelihood over their output. This type of *boosted classifier*, previously explored within iDQ [27], would again reduce the problem to likelihoods defined over a one-dimensional space. Although not yet implemented, such boosted classification schemes are an active area of research.

3. Decomposition of the statistical inference

iDQ divides the workflow into several asynchronous processes which communicate to share updated models and calibration. As figure 2 shows, features are generated for each IFO and retrieved by various processes (section 3.1). Vectorized representations of the detector's auxiliary state are constructed as needed in each process (section 3.2). Training (section 3.3) produces models for each classifier, which are then used in both evaluation (section 3.4) and timeseries production (section 3.6). The evaluated output is used to calibrate the model (section 3.5), and the resulting map transforms the low-latency predictions made during timeseries production into probabilistic measures with associated uncertainties. These measures, such as p_G, can then be ingested by GW searches in real-time.

3.1. Data discovery

iDQ relies upon an external source of features, typically taken to be tabular data denoting the location and properties of non-Gaussian transients in a set of channels. Although not strictly necessary (e.g. iDQ could ingest the raw timeseries directly from the detectors), we find that this preprocessing efficiently extracts the features relevant for our classification problem. Figure 2 represents these feature streams as directed arrows exiting the feature extractor and entering the iDQ workflow. Pragmatically, these are implemented as abstractions that manage data discovery and produce a consistent tabular output format regardless of the features' source, thereby simplifying any client interactions throughout the pipeline.

Most feature extractors operate in one of several wavelet domains, extracting excess power as collections of time-frequency pixels. Common choices are the Haar wavelet transform and the Q-transform [37]. However, the precise form of the feature extractor is unimportant beyond the fact that different wavelet transforms are able to better resolve different aspects of non-Gaussian noise transients. iDQ used features extracted with an implementation of the Haar transform (KleineWelle [42]) during the LIGOs' first two observing runs before switching to a low-latency implementation of the Q-transform during the third observing run [38]. iDQ can also use features extracted from multiple wavelet transforms simultaneously. What's more, not all feature extractors produce the same set of features, although all provide some measure of the transient's central time, duration, frequency content, and amplitude or significance (typically measuring how rare the transient would be in stationary Gaussian noise). iDQ allows users to choose the (sub)set of features, although typically at least the central time, amplitude, and frequency are used. reference [28] explored the relative importance of features in the Haar domain, finding the relative time offset and significance to be most important.

Samples from G and C are identified based on the features present in h. Specifically, any transient which meets the criteria for G (see section 2) is recorded as a *target time*. Samples from C, called *random times*, are drawn according to a Poisson process in stretches of data sufficiently far away from target times. These clean segments are defined by another set of thresholds on h, which are typically chosen to be slightly looser than the bounds defining target times. This creates an effective buffer between samples in G and C, which is believed to avoid threshold effects and improve classifiers' ability to distinguish the sets. At the end of the day, any time segment not declared clean is considered *dirty* (i.e. $t \in G$), and this dirty time is accounted for within the calibration's prior odds (section 3.5).

Additionally, iDQ gathers IFO-state information from a remote database [43]. Segments produced outside of iDQ record high-level state information about the detectors, such as when the IFOs record science-quality data, but without fine enough temporal resolution to flag subsecond non-Gaussian noise transients. iDQ polls the database for such segments, filtering the training and evaluation samples to retain only times within science-quality data. This may not be strictly necessary in all cases and can introduce additional latency before segments are available. Segment information is, therefore, optional (directed arrows in figure 2 are dashed instead of solid), but is almost always used in practice except during low-latency timeseries production. Timeseries production only applies existing models and calibration to IFO data. Therefore, it does not care whether the detectors are currently recording science-quality data.

3.2. Feature vector construction

Given a stream of features, we must still determine which features to use. Most, if not all, supervised classification schemes require consistent dimensionality in the input feature space ($\vec{a} \in \mathcal{R}^N$ with fixed N). Therefore iDQ must downselect features if too many auxiliary transients are nearby or fill in default values if no auxiliary transients are available. Mapping the streams of features from many auxiliary channels into an array of fixed dimension is referred to as *vectorization*, and is carried out on-the-fly as needed within the pipeline.

Although other types of features have been investigated in the literature, such as averaging over small neighboring time windows [33], iDQ implements the following vectorization scheme following reference [28], which was also employed in [32]. For each auxiliary channel, iDQ queries all transients in that channel with central times within some window surrounding the time of interest. If no auxiliary transients are available, default values are returned for all requested features, thereby denoting auxiliary channels that were inactive. Otherwise, iDQ will extract features from the loudest auxiliary transient (largest ρ) within the window. We note that this is not the only choice, and quiet transients in closer coincidence with the time of interest may be more relevant than louder transients further away [39]. Nonetheless, this *select-loudest* algorithm works well in practice [11] with coincidence windows ~ 100 ms.

The resulting *feature vectors* are collected into sets for training, evaluation, and timeseries generation. Each vector is labeled according to whether the time is associated with an element of either G or C based on h, thereby constituting a supervised learning training set. These labels are completely ignored during evaluation and timeseries generation. As implemented, each *feature vector* retains a reference to the data discovery abstraction used to retrieve the features (see section 3.1). In this way, classifiers can access the full feature set if desired, although most rely on the vectorization scheme articulated above. A notable exception is the Ordered Veto Lists algorithm (OVL [29]), which directly ingests the data discovery abstraction during training. This is done to avoid additional overhead associated with vectorization and is peculiar to the OVL algorithm, although this type of behavior is more broadly supported within iDQ.

3.3. Training

Given vectorized representations of the auxiliary features for each time of interest, iDQ then trains classifiers to separate the labeled samples. Again, classifiers are only given features extracted from *safe auxiliary channels* and cannot construct decision surfaces based on the vectors' G or C labels. The details of each classifier's training algorithm are specific to each classifier, and iDQ generally relies on external libraries for their implementation (e.g. support vector machines, random forests, and neural networks as implemented in

[11] There is some evidence that our feature vectors are relatively sparse (i.e. it is rare to have multiple coincident auxiliary transients in a single channel) and the relevant information encoded in the feature sets is simply whether or not there were any non-Gaussian transients in the auxiliary channel within the specified window. In that case, it is perhaps not surprising that the *select-loudest* algorithm retains all the relevant information.

scikit-learn [44], Keras [45], and XGBoost [46]). However, a few algorithms are implemented directly within iDQ, such as OVL [29].

In this way, iDQ uses supervised learning on labeled auxiliary feature vectors to generate maps from high-dimensional input spaces to a single scalar *rank*, a real number between zero and one. We note that vectorization itself also introduces dimensional reduction, as we extract features from at most a single auxiliary transient per channel, but beyond that it is the classifiers themselves that determine which features are relevant and which are not. This establishes a *model* for each classifier (α):

$$\mathcal{M}_\alpha : \vec{a} \in \mathcal{R}^N \rightarrow r_\alpha \in [0,1]. \tag{10}$$

Each classifier generates a separate model, and different models may be able to separate the training sets to different degrees. iDQ requires ranks to be within the unit interval, although this choice is arbitrary. The important aspect of the model is the relative ordering of samples, not the precise value of the rank. Therefore, any monotonic mapping from the unit interval to another range will preserve all relevant information.

Each classifier additionally manages internal cross-validation or provides techniques to prevent over-fitting (see appendix B for an example). Again, the details may be specific to each classifier, but iDQ also provides a cross-validation scheme independent of the classifiers themselves. Section 3.4 describes the various techniques used to evaluate a classifier's performance fairly, making sure to account for any generalization error associated with the derived models. Each trained model additionally records the range of data used during training as a unique *hash*. These hashes are used in evaluation (section 3.4) and timeseries generation (section 3.6) to track how data was manipulated as it progressed through the pipeline.

Some algorithms also support measures of *feature importance* within trained models. These are often related to the directional derivative of the model with respect to each auxiliary feature: $\partial\mathcal{M}/\partial a_i|_{\vec{a}}$. Features with larger directional derivatives tend to be more important, although each classifier's measure of feature importance may adopt different specific details (appendix B describes how OVL extracts the feature importance measures shown in figures 7 and 8). Not all classifiers provide this information. Many that do only provide global estimates averaged over all samples instead of the local estimates at a particular auxiliary vector. However, we leave further investigations of standardized measures of feature importance, such as reference [47], to future work.

3.4. Evaluation

Supervised learning relies on cross-validation to evaluate generalization errors. This typically consists of subdividing the data into distinct sets, one of which is used for training and the other to evaluate performance. iDQ supports two main ways to subdivide the data into different *bins* for cross-validation.

Acausal or *round-robin* binning divides the data into a sequence of small segments, mixing samples independently of their time ordering. Figure 3 demonstrates how segments are assigned to different data sets. Briefly, for N different bins with M segments per bin, iDQ generates $N_{\text{segs}} = N \times M$ segments of equal duration. Data from the first segment is assigned to the first bin, the second segment to the second bin, and so on. Data from the $(M+1)^{\text{th}}$ segment is assigned to the first bin, the $(M+2)^{\text{th}}$ segment to the second bin, and this repeats until all data has been assigned. We then train over all data outside of the ith bin to generate a model used to evaluate data inside the ith bin, repeating the procedure for all bins. This approach provides features drawn from consistent distributions in both training and evaluation, even if the instantaneous feature distributions change over time. With features drawn from consistent distributions in both training and evaluation, acausal binning measures the best performance that should ever be expected from a classifier.

Causal binning again divides the data into many small segments. Figure 4 shows this schematically. Each analysis then trains on a cumulative set of historical segments and uses the resulting model to evaluate the next segment, preserving the relative time ordering. Again, this is repeated for all segments, each using all historical data available during training. This allows users to investigate how detector non-stationarity affects their algorithm.

Each approach has its uses, and both produce ROC curves that measure classifiers' performance in a fair manner. Importantly, iDQ can simultaneously run multiple classifiers, guaranteeing they see identical data sets and that comparisons are as fair as possible. Evaluation also records each model's hash within each feature vector to maintain a record of how each feature vector was classified.

3.5. Calibration

While training and evaluation are the foundation of iDQ's supervised learning approach, calibrating the resulting ranks into probabilistic statements is of equal importance. iDQ does this by directly modeling the observed conditioned likelihoods for each classifier's rank: $p(r_\alpha|G)$ and $p(r_\alpha|C)$. Just like each classifier's model retains a hash to track provenance, each pair of conditioned likelihoods, or *calibration map*, records a unique hash to denote the evaluated samples from which it was generated.

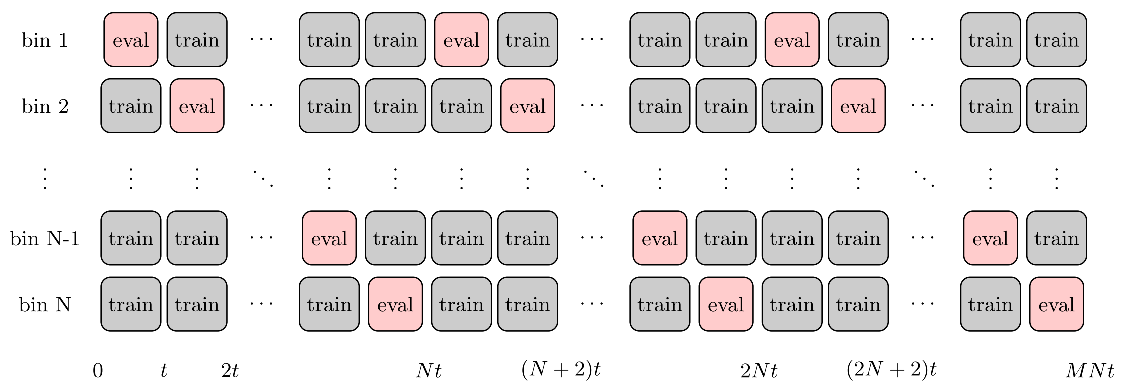

Figure 3. Schematic diagram showing data segmentation for acausal batch operation. Each row corresponds to one of N bins, and each column corresponds to one of $N \times M$ segments. Data from sequential segments are assigned in order to different bins, effectively giving each bin access to samples from throughout the entire analysis period during training. Evaluation sets from each bin remain disjoint from training sets, though, allowing for meaningful cross-validation of classifier performance.

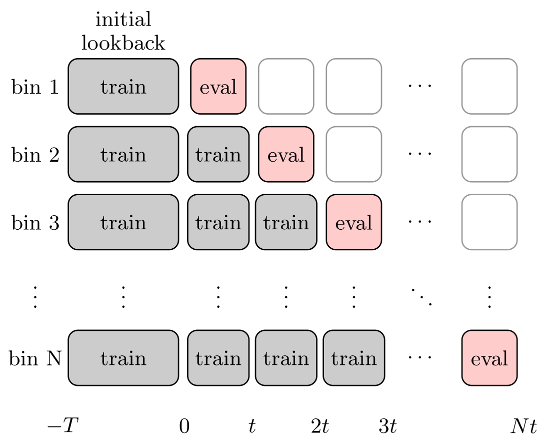

Figure 4. Schematic of data segmentation for causal batch analyses. Each row illustrates the data included in a single bin, and columns represent different segments of data. Each bin uses historical data cumulatively when training (grey shaded boxes), including an initial lookback period before the first evaluation segment (red boxes). Bins progressively train on more and more time-ordered data, thereby testing algorithmic sensitivity to detector non-stationarity in a similar way to what is experienced during streaming operation.

iDQ can model the conditioned likelihoods in two ways, related to different assumptions about the nature of ranks produced by a classifier's model. Both quantify the uncertainty in their estimates, and both work well in practice. Figure 5 shows how they correctly model non-trivial conditioned likelihoods. Calibration maps further estimate prior odds for G and C samples. We describe these procedures in more detail below.

3.5.1. Continuous calibration maps

If classifiers produce ranks that can take any real value within the unit interval, with no discrete lumps of probability so that

$$\lim_{\varepsilon \to 0} \left(\int_{r-\varepsilon}^{r+\varepsilon} dx\, p(x|X) \right) \propto \varepsilon \quad \forall\, r \in [0,1], \tag{11}$$

then it is appropriate to model the underlying distribution with a continuous kernel density estimate (KDE). iDQ implements a Gaussian kernel and dynamically optimizes the kernel's bandwidth to maximize a cross-validation likelihood quantifying how well the KDE reproduces the observed sample set. iDQ reflects observed samples around rank = 0 and 1 to avoid edge effects within the KDE while numerically enforcing proper normalization. Uncertainty in the KDE model for the true likelihood at each rank is modeled by a β-distribution (see appendix A). This procedure accurately predicts the observed variance in KDEs obtained from different realizations of sample sets drawn from the same underlying distributions, regardless of the true distribution.

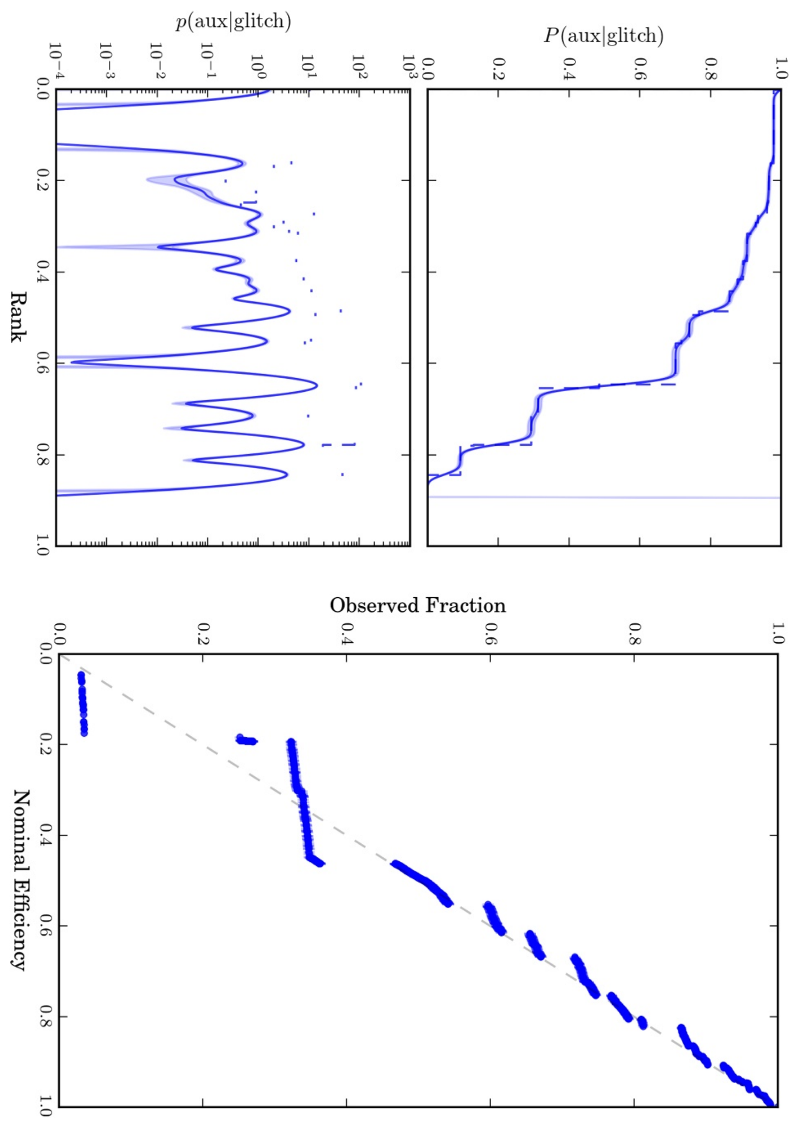

Figure 5. (*left*) Example calibration distribution representing $p(\mathcal{M}(\vec{a})|G)$ from a batch analysis during a period of elevated radio-frequency noise at the LIGO Hanford detector on Monday December 22, 2015 (see figure 9). Solid lines denote the optimized Gaussian KDE representing the distributions; shaded regions correspond to the expected error in that fit; and dashed lines show histograms of the observed ranks. A cumulative histogram is shown above the differential histogram as it more clearly shows the rounding effects of the finite bandwidth assumed within the KDE model. (*right*) The corresponding calibration coverage shows the approximate fraction of glitches assigned a particular nominal efficiency the calibration map. Perfect calibration corresponds to a diagonal line. The step-like behavior observed here comes from interpolation artifacts when assigning nominal efficiencies to glitches based on the regularly sampled timeseries.

3.5.2. Discrete calibration maps

If the trained model only produces a finite number of possible ranks, the resulting distribution may be better modeled as a weighed sum of δ-functions

$$p(r|X) = \sum_i w_i \delta(r - r_i) \quad \Bigg| \quad \sum_i w_i = 1 \tag{12}$$

equivalent to our continuous Gaussian KDE in the limit of vanishingly small bandwidths. Weights are estimated as the fraction of observed samples assigned to that rank, and uncertainty in the weights is again modeled as a β-distribution such that

$$p(w_i|n_i, N) \propto w_i^{n_i} (1 - w_i)^{N - n_i} \tag{13}$$

where n_i out of N total samples were assigned rank r_i.

3.5.3. Prior odds

While iDQ's conditioned likelihoods, and therefore the likelihood ratio Λ^G_C, do not depend on the prior odds between G and C, most applications instead rely on p_G (equation (5)). While p_G is monotonic in Λ^G_C, the exact value can be made arbitrarily large or small based on the prior odds assumed. This forces us to carefully consider our assumptions *a priori* about the relative frequencies of G and C. iDQ implements several choices, either allowing users to specify fixed prior odds, estimating them based on the relative fractions of samples within training sets, or estimating them based on the fraction of time declared clean within a training set. All these are based on the premise that the prior odds are approximately the ratio of the rates at which each type of sample occurs, adopting different techniques for approximating the rates of G (R_G) and C (R_C) samples.

As appropriate for inference over tabulated data, one approximation is $R_G/R_C \approx N_G/N_C$. This asks what the chance is that one would select either a G or C sample when randomly choosing an element of the fixed training set. Generally, N_G is set by the true R_G in the detector and the amount of time over which we collect samples. Similarly, N_C depends on the rate at which we generate clean samples, which is an arbitrary choice typically chosen to balance training accuracy and computational expense. While formally correct for tabulated data, and therefore useful in some applications, this model does not necessarily represent the prior odds relevant for timeseries production.

Alternatively, we can model $R_G/R_C \approx T_G/T_C = (T/T_C) - 1$ where T_C/T is the fraction of analysis time declared clean (alternatively, not dirty) when constructing the sample set. We measure T_C directly from the conditions defining C based on h (section 3.1). This approach models the relative frequency of times declared glitchy or clean, rather than selecting an element from tabular data, as is more appropriate for timeseries production.

3.6. Timeseries production

iDQ produces streaming estimates of statistical quantities as the culmination of training and calibration. These timeseries should be thought of as the main data product generated within iDQ and are the most applicable to GW searches. iDQ generates vectorized feature sets on a regular grid in time, typically sampled at $\geq$ 128 Hz. The regularly spaced vectors are then evaluated using a trained model, and the resulting array of ranks is calibrated into several statistical quantities using a calibration map. These quantities are then distributed in real-time to GW searches, and we discuss ways to incorporate them within searches in section 6.

Section 4 describes the differences between offline (*batch*) and online (*stream*) modes of operation in more detail, but both utilize asynchronous processes to manage training, evaluation, calibration, and timeseries production. Because some tasks require the output from other tasks, iDQ synchronizes them by polling for specific models and calibration maps from various repositories, referred to as *modular data servers* (MDSs). In this way, timeseries jobs can obtain the most relevant model and calibration map for any stretch of data. As with data discovery, iDQ does not depend on the particular implementation of a MDS as long as it allows tasks to *get* and *put* results through a consistent interface. *De facto*, most MDSs are implemented via local filesystems.

As noted in section 3.5, the prior odds assumed during calibration can have a significant impact on the interpretability of the resulting timeseries. In particular, we expect $p(G) \ll p(C)$ because the typical duration of non-Gaussian noise transients is usually much less than their separation: $\mathcal{O}(10^{-1}\,\text{sec}) \ll \mathcal{O}(10^{2}\,\text{sec})$.

Because of the ambiguity associated with the choice of prior odds, iDQ produces timeseries for multiple statistical quantiles besides p_G, including Λ^G_C, the conditioned survival functions (*efficiency* and *false alarm probability*), as well as the raw rank produced by the model. Access to Λ^G_C allows users to estimate p_G with whatever prior odds they choose. Furthermore, the *false alarm probability* approximates the amount of time discarded by the classifier, and therefore could be used to set a convenient working point. What's more, the calibration map also provides uncertainty estimates based on the finite calibration sample size.

4. Batch vs. stream modes

In addition to the decomposition described in section 3, iDQ supports two modes of operation related to how it synchronizes processes.

The offline, or *batch*, mode targets specific stretches of data and can support both *causal* and *acausal* binning schemes (section 3.4). Additionally, *batch* jobs run tasks synchronously within each bin, which is to say that training must complete before evaluation begins, evaluation must complete before calibration begins, and calibration must complete before timeseries are produced. However, separate bins are independent and can be processed in parallel. Batch analyses have loose latency requirements.

The online, or *stream*, mode instead runs in low-latency, typically producing timeseries within $\mathcal{O}(10^{-1}\,\text{sec})$ of receiving features, regardless of the wavelet transform used to generate those features. The

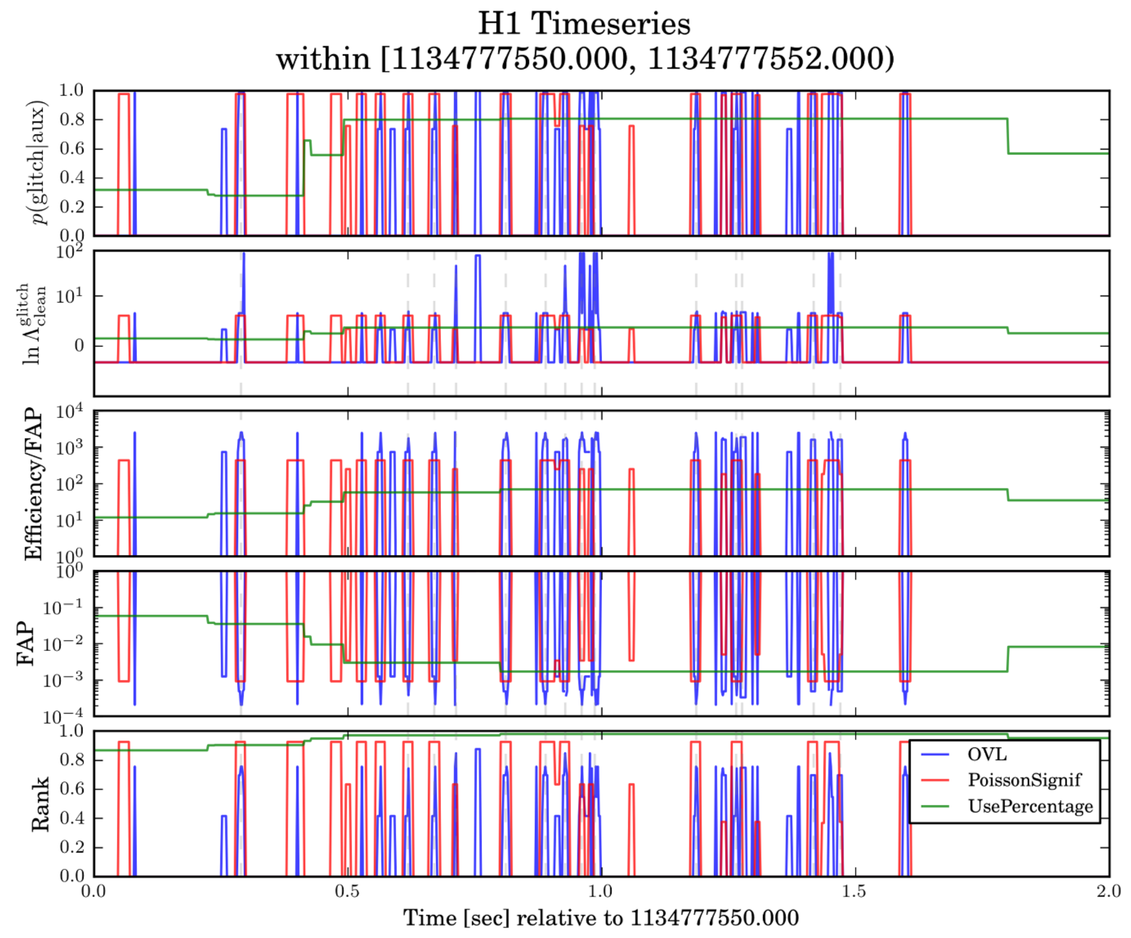

Figure 6. Example iDQ timeseries for OVL [29] with 3 different ranking metrics (see appendix B) from a batch analysis of $\mathcal{O}(10^2)$ sec during a period of elevated radio-frequency noise at the LIGO Hanford detector on Monday December 21, 2015. From the top to the bottom, we show the estimated probability of a glitch given the auxiliary data, the log likelihood-ratio between the glitch and clean models, the ratio of the detection efficiency to the false alarm probability (FAP), the FAP itself, and the raw rank produced by the classifier. For reference, central times for glitches derived from h are shown with light grey lines, occuring at a rate of $\sim$ 10 Hz at this time. We see that OVL correctly identifies the times at which glitches are present in h and that our prior ($p(C) \gg p(G)$) correctly drive p_G to small values when the likelihood is uninformative.

dominant source of latency for iDQ is feature generation and vectorization. The time required for feature extractors to process $\mathcal{O}(10^3)$ channels was recently reduced to $\sim$ 5 sec with a low-latency implementation of the Q-transform used during the third observing run [38], as opposed to $\sim$ 32 sec for the implementation of the Haar transform (KleineWelle [42]) used throughout the first two observing runs. Vectorization can also limit iDQ's latency, but the pipeline can generate vectors consisting of $\mathcal{O}(5)$ features for each of $\mathcal{O}(10^3)$ channels at rates above 128 Hz, which is typical of production configurations as most glitches have durations $\sim$ 100 ms. We also note that iDQ can use features extracted with multiple wavelet transforms simultaneously, if desired.

Online jobs process all data causally and manage tasks asynchronously. However, because some tasks require input from others before they can begin, the streaming iDQ pipeline will run small batch pipelines if models and/or calibration maps are not initially available for all classifiers. Once initial models and calibration maps are available, separate processes for training, evaluation, calibration, and timeseries production run in parallel and interact through put and get requests in MDSs (section 3.6). This means that re-training and re-calibration happen continuously, with a new data set defined and fed to classifiers as soon as they finish processing their previous sets. When each task begins processing a new data set, it polls the relevant MDS to obtain all the required data products without waiting for the asynchronous processes to complete. For example, several evaluation strides may use the same set of models because the training jobs generally take longer to complete than evaluation. Nonetheless, as soon as a training job completes, the evaluation jobs will automatically retrieve the new model.

Training jobs usually take the longest to complete, with runtimes of $\mathcal{O}$(hours), and it is possible, then, that trained models will respond relatively slowly to detector non-stationarity. However, if the model becomes out-of-date, the much faster evaluation and calibration jobs, which complete in $\mathcal{O}$(sec), will detect

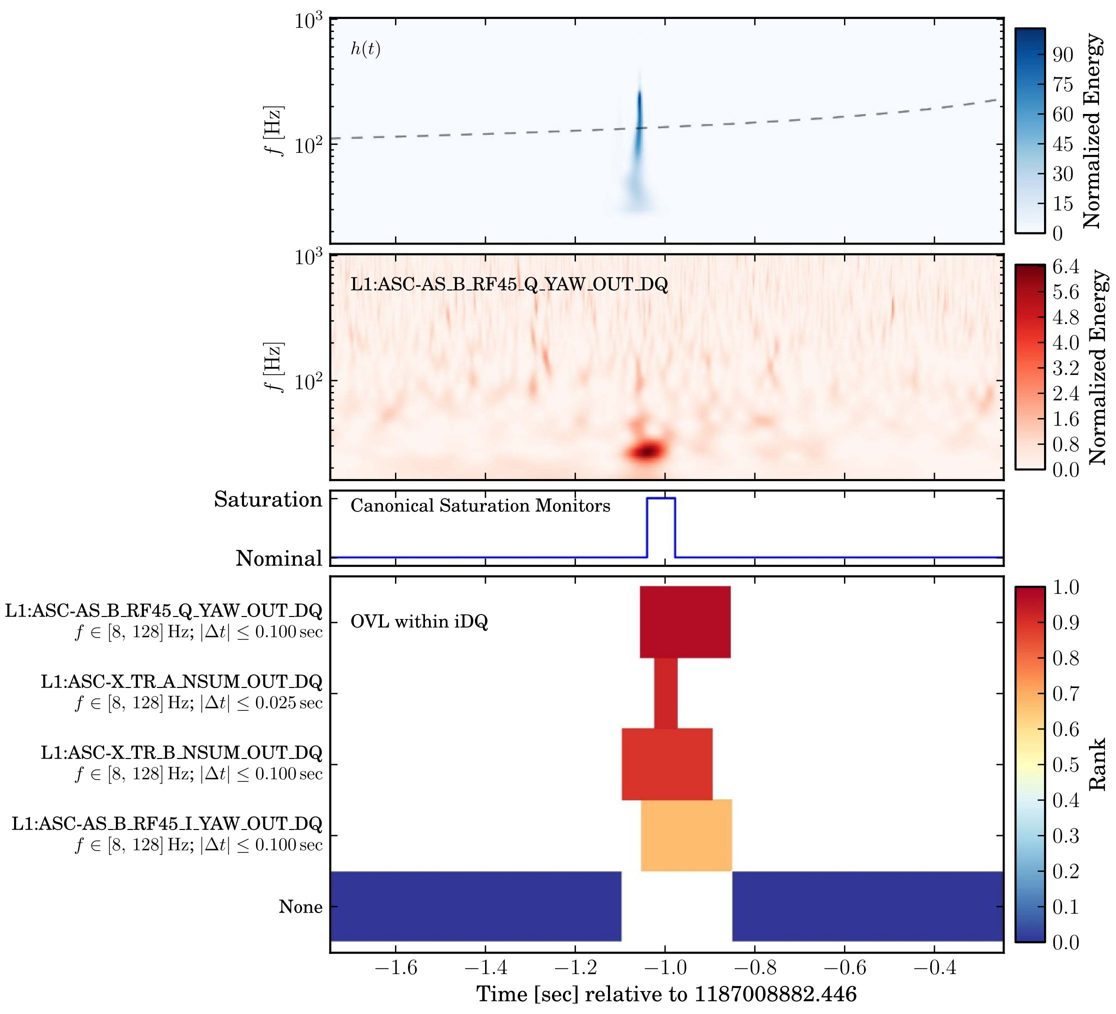

Figure 7. iDQ's low-latency predictions surrounding the non-Gaussian noise transient coincident with GW170 817 in the LIGO Livingston detector. (*top panel*) Time-frequency decomposition of the GW strain chosen to highlight the noise transient's short duration; GW170 817's inspiral track is shown for reference (*dashed line*), with times measured relative to the coalescence time. (*2nd panel*) Time-frequency representation of the top-ranked auxiliary witness active at this time, autonomously identified by iDQ as correlated with non-Gaussian noise without *a priori* knowledge about the type of noise present in the detectors. (*3rd panel*) A canonical saturation monitor, which identified the time as problematic. (*bottom panel*) OVL's feature importance, showing multiple veto configurations (appendix B) active in coincidence with the glitch. Color denotes OVL's rank, with rank$\rightarrow 1$ indicating high confidence in the presence of a glitch. OVL finds various *Alignment Sensing and Control* (ASC) auxiliary channels to be relevant, including the amount of light transmitted through the x-arm's end-station test mass (ASC-X_TR), which are partly responsible for mainining the interferometer's angular alignment during operation.

the decreased performance and update iDQ's probabilistic statements accordingly. For example, if a model suddenly cannot distinguish between G and C samples, the calibration jobs will update the conditioned likelihoods to $p(r|G) \sim p(r|C)$, and therefore return the prior odds.

Asynchronous, parallel processing lowers the latency required to produce timeseries at the cost additional complexity in data discovery. Because data may occasionally be dropped before it reaches iDQ, each process analyzes data in small strides and internally manages timeout logic, skipping data if it takes too long to arrive. In production, we generally find iDQ achieves duty cycles above 99\%, thereby essentially guaranteeing results will be reliably provided to GW searches in low-latency. In fact, iDQ's information was distributed at the same time as the calibrated GW strain during the third observing run.

While many analyses will utilize the batch workflow, we expect streaming processes to be the most relevant in the coming years as iDQ's predictions are incorporated further into low-latency GW searches. Indeed, the examples in figures 7 and 8 were all derived from streaming analyses. Section 6 enumerates a few other possible applications.

5. Examples

We present a few examples of iDQ's behavior with real detector data from the first two observing runs. First, and perhaps most importantly given the context, is the non-Gaussian noise artifact coincident with

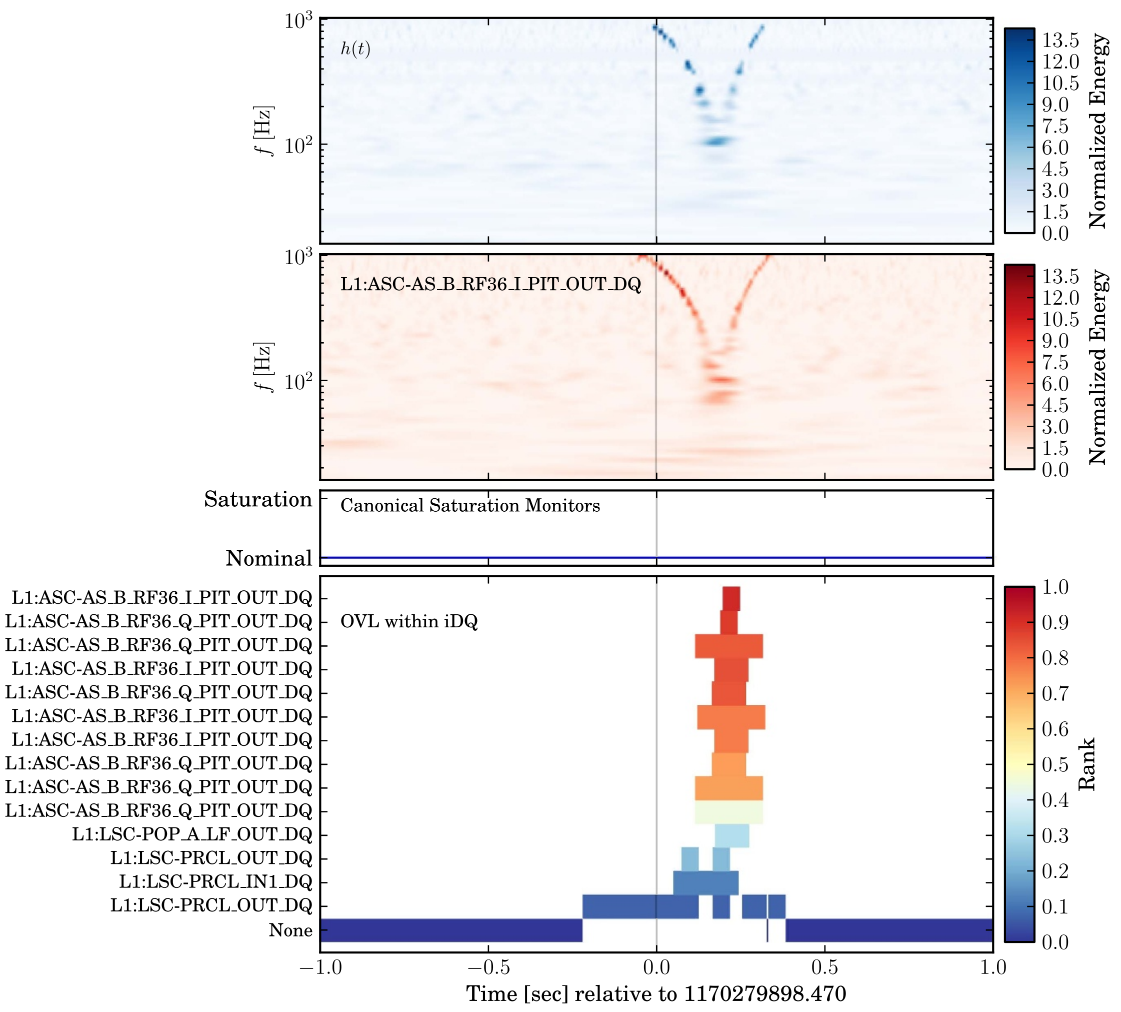

Figure 8. Example iDQ output surrounding a GW candidate identified by a search for unmodeled transients [48] on 4 February 2017. (*top panel*) A time-frequency representation of $h(t)$, showing the radio-frequency *whistle* in coincidence with the candidate. (*2nd panel*) A time-frequency representation of the highest-ranked auxiliary channel active at this time, showing a similar whistle. (*3rd panel*) Canonical saturation monitors, inactive during the whistle, as expected. Canonical monitors for whistles do not exist in low-latency and can often miss whistles even in high latency. (*bottom panel*) OVL's feature importance for active auxiliary channels. Color indicates OVL's rank, with rank$\rightarrow$ 1 corresponding to $p_G \rightarrow 1$. We see that iDQ clearly identified the whistle in low-latency while canonical monitors were silent or otherwise unavailable.

GW170 817 in the LIGO Livingston interferometer [49]. Although similar noise transients are witnessed several times per day in each IFO, their exact cause is not known. They are, however, often associated with saturations within the interferometric control systems, and monitors exist to flag such saturations. Figure 7 shows a time-frequency projection highlighting the noise transient's short duration, as well as the behavior of the canonical monitors for saturations. iDQ, at the same time and without prior knowledge of the existence of saturations or which auxiliary degrees of freedom correlate with noise in h, autonomously identified witnesses for such events and flagged the time as very likely to be a glitch in real-time. This information was automatically made available within 8 sec of the candidate being reported to the Gravitational Wave Candidate Event DataBase (GraceDB [50]), thereby informing decisions in real-time about the candidate's probability of being astrophysical in origin and the resulting announcement to the broader astronomical community [51]. GW170 817 serves as an example of how iDQ can independently identify noise sources already known to human analysts. Additionally, the witnesses iDQ identifies sometimes flag problematic time associated with loud noise transients that go unnoticed by more conventional monitors, including saturation-like glitches that happen to not saturate the control signals being monitored (e.g. [52]).

Pursuing this further, figure 8 presents a radio-frequency *whistle* identified as a possible GW candidate by oLIB, a search for unmodeled GW bursts [48], on 4 February 2017. iDQ vetoed this event within 7s. As with GW170817, iDQ autonomously developed witnesses for such noise and clearly identifies the time as glitchy. We note that canonical monitors for saturations did not flag this event, as expected because they monitor sources of different types of non-Gaussian noise, and iDQ's low-latency predictions were the only data quality products available at the time that could reject this candidate as noise without relying on the human

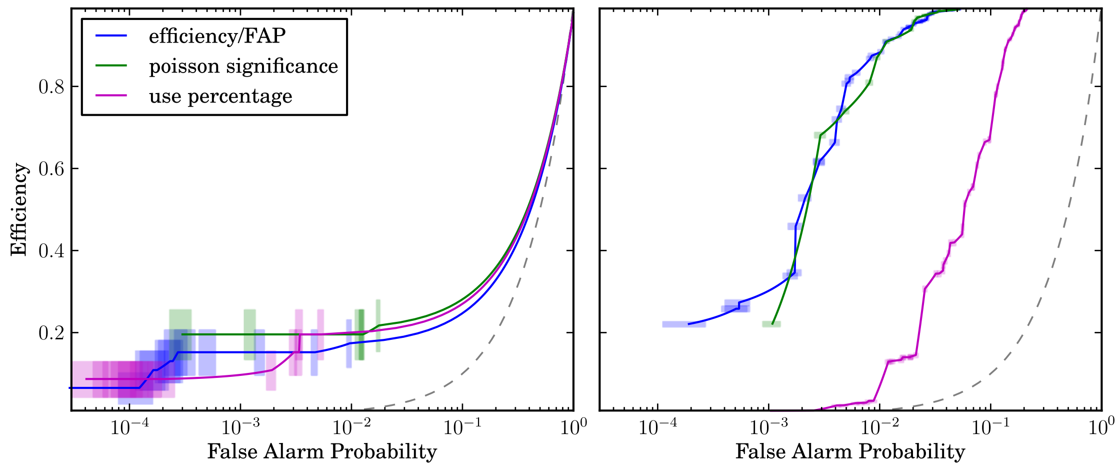

Figure 9. Example receiver operating characteristic (ROC) curves generated within iDQ relating the glitch detection efficiency to the false alarm probability. (*left*) ROC curves from a batch analysis of LIGO Hanford data from December 18–20, 2015 (~ 2 days, 101 glitches). (*right*) A batch analysis of LIGO Hanford data from December 22, 2015 (1 hour, 5504 glitches). Shaded boxes approximate 1-σ uncertainty from the counting experiment used to measure the ROC curves at the points where they are measured, with linear interpolation between. Grey dashed lines correspond to uninformative classifier, and colored lines correspond to different ranking metrics within OVL (appendix B.1). Each of these analyses used iDQ's internal acausal cross-validation scheme with two bins and two segments per bin, and these ROC curves therefore account for generalization error. These two examples represent approximate limits on the performance seen within iDQ, and primarily differ by the type of noise present within the detector. Nonstationary changes in non-Gaussian noise can be sudden and severe, necessitating automatic re-training and re-calibration.

inspection of the signal morphology in $h(t)$. Rejecting candidates from unmodeled transient searches based on $h(t)$ morphology itself is risky, and therefore iDQ's auxiliary witnesses provide much greater confidence the transient was of terrestrial origin. At the time of writing, no low-latency monitors exist for such radio-frequency whistles besides iDQ.

Figures 7 and 8 show two examples of noise transients that are typically witnessed well by the auxiliary degrees of freedom used within iDQ. Both were identified in low-latency the OVL algorithm ([29]; appendix B) running within iDQ's framework. We typically find that, in agreement with reference [28], OVL performs as well as, if not better than, more complex algorithms, and we primarily uses its predictions to identify noise in low-latency. In general, different classifiers may witness different noise sources, and combining the ranks from multiple classifiers (creating a *boosted classifier*) is an active area of research.

Figure 9 presents a few ROC curves showing iDQ's typical performance, also demonstrating iDQ's ability to run multiple classifiers simultaneously over the same data. We focus on OVL with 3 different choices for the ranking metric (appendix B.1). OVL, running within iDQ's framework, typically identifies ~ 10\%–40\% of non-Gaussian noise artifacts (KleineWelle [42] triggers between 32 and 2048Hz with $\rho \gtrsim 8$) at the cost of $\lesssim$ 0.1%–1\% false alarm probability. This depends on the mixture of noise sources present within the detectors, though, as a high fraction of well-witnessed noise will lead to correspondingly more impressive ROC curves. Figure 9 shows one such example from the LIGO Hanford detector a few days before GW151 226 [53]. An intense radio-frequency *glitch-storm* produced a large number of clearly witnessed noise transients, and OVL identified $\geq$ 90\% of them with $\leq$ 1\% false alarm probability. It is worth noting that, at that time, offline canonical monitors for radio-frequency noise had become less effective due to detector non-stationarity [17,]. iDQ automatically detected new witnesses without human intervention and retained a high glitch-detection efficiency in low-latency.

6. Applications within gravitational-wave searches

The optimal incorporation of probabilistic data quality information within GW searches remains largely unaddressed in the literature. In this section, we present a likelihood ratio test that incorporates imperfect knowledge of the presence of non-Gaussian noise within our detectors. We first review the current state of the field and how data quality information is often incorporated into searches, demonstrating how iDQ's products could be used within existing methodologies while discussing different approaches' relative advantages and drawbacks. We then formulate a search based on first-principles models of noise processes within detectors and the incorporation our imperfect knowledge of data quality based on both auxiliary and GW strain information.

6.1. Current Veto and Gating Strategies

Data quality products are currently applied within searches in two main ways. Data quality *vetoes* are applied after filtering, meaning after a search has produced a list of candidates. Usually, vetoes are specified as a list of segments and any candidate that falls within these segment is rejected. *Gating*, on the other hand, attempts to remove problematic data before filtering, thereby preventing false positives from appearing in candidate lists at any point. Importantly, both approaches assume *binary* data quality information. That is, the data is either declared clean or glitchy with complete certainty. iDQ extends this by providing probabilistic measures of data quality.

Let us begin with vetoes applied after candidates have already been identified. A naive approach is to perform a simple coincidence experiment between the search's candidates and data quality monitors. If, for example, iDQ's false alarm rate timeseries dips below a threshold anywhere within a coincidence window surrounding a GW candidate, this may indicate that a non-Gaussian noise artifact is present near the candidate, which may suggest it is of terrestrial origin. Other methods to identify problematic data based on auxiliary channels, often constructed by hand, are often used to define such veto segments. Of course, this depends on the precise way GW candidate reference times are recorded, as low-mass compact binary coalescences can sometimes coalesce several seconds after the non-Gaussian noise that caused the false alarm. In effect, this maps timeseries output into *binary veto segments* with a window and a threshold.

iDQ's output was used in this way with modest success during the first two observing runs [54], but vetoes suffer from several limitations. First, searching for extreme excursions in any of iDQ's timeseries within a coincidence window naturally introduces an additional trials factor. For example, if the coincidence window is longer than the typical separation between glitches, then the probability of obtaining $p_G \sim 1$ for at least one point in that window is almost surely 1. This complicates the statistical interpretation of iDQ's predictions, since, for example, what iDQ reports as the false alarm probability will not generally correspond to the false alarm probability of finding a large excursion within a large window. Furthermore, the mapping from threshold–window pairs to the effective false alarm probability will depend strongly on the quantity used, and there is no single obvious choice. One may threshold on the false alarm probability in an attempt to bound the probability of false alarms, but one may alternatively threshold on the likelihood ratio Λ^G_C as this may be more appropriate for likelihood ratio tests. This could be tuned by hand, but that then negates any efforts to calibrate model predictions within iDQ, thereby defeating the purpose of a large part of the pipeline. Nonetheless, the conceptual simplicity and ease of implementation make this attractive for practical applications. For this reason, veto segments remain a core data quality product within GW searches.

A perhaps more sophisticated approach to vetoing based on extremized timeseries is to incorporate the extremum value within a window as part of a likelihood ratio test. While this may remove some ambiguity about the effective false alarm probability (the likelihood ratio test will naturally account for the probability of seeing such extrema within *clean* data) and remove the need to tune thresholds by hand, the ambiguities associated with the choice of window and statistic still remain. Although properly constructed likelihood ratios should be able to simultaneously incorporate arbitrarily many window–statistic pairs, one quickly encounters the pragmatic issues with modeling high-dimensional probability distributions that led us to employ machine learning as dimensional reduction in the first place (section 2.2).

Reference [55] explores a somewhat simpler construction, in which the iDQ $\log \Lambda^G_C$ timeseries is maximized over 1-second windows, slightly transformed and then applied directly as a multiplicative factor to a likelihood ratio detection statistic. The transformation for iDQ's $\log \Lambda^G_C$ was empirically determined and fixed *a priori*, essentially assuming the functional form for the trials factor introduced by the maximization and how that modified the likelihood of a signal being present. While the assumed mapping may not be optimal, iDQ was found to moderately benefit current searches even with this simple approach.

Another approach is the idea of *gating* in some form, in that we should remove all problematic (glitchy) times before filtering, thereby removing the need to select a specific extremization procedure to veto *post hoc*. Heuristically, the logic is that candidates are generated with large ρ by glitches ringing up templates, and we can model the correlations between ρ and the presence of a glitch either explicitly within a likelihood ratio test over many variates or implicitly by removing the contribution of $h(t)$ due to a glitch from the matched filter response altogether.

Such *gating* schemes are now ubiquitous within the field and trivial to implement within white noise. However, IFOs generate colored noise and the presence of loud glitches can ring whitening filters, polluting surrounding data that would otherwise be unaffected by the noise-transient. This prompted the development of *inverse-Tukey window* gating [56, 57] as well as more complicated *in-painting* techniques designed to zero the filter response within a specified window after whitening [58]. We derive similar approaches to in-painting in section 6.2 from first-principles.

While gating mechanisms become more complex, we note that current approaches all rely on the same premise, that there is a predefined set of times declared glitchy that must be removed from the analysis. These gates are typically defined by extremization processes over monitors and hard thresholds, and therefore suffer from the same ambiguities in determining appropriate settings as *post hoc* veto segments, although without the ability to simultaneously consider multiple choices as would be possible in a likelihood ratio test. Nonetheless, it is sometimes the case that filtering artifacts from gating or the ambiguity in defining gates are preferable to the original noise transient.

One could define gating conditions based on iDQ timeseries, but again we must face the selection of thresholds with no more obvious metric than guessing and checking how this affects searches' sensitivities. One could be tempted to soften the hard thresholds with an *adaptive gate*, such that the matched filter response of a timeseries h with a filter f would be modified to

$$\rho_C(t) = \int d\tau h(t-\tau) f(\tau) p(C|\mathcal{M}(\vec{a}(\tau))) \tag{14}$$

in effect estimating ρ by probabilistically keeping the times expected to be clean based on auxiliary degrees of freedom. Again, while heuristically appealing, it is not clear that this approach is optimal. Indeed, it suffers from the same issues of whether to apply the adaptive gate before or after whitening the data as normal gates.

While GW searches have benefited from data quality information made available in the past (e.g. [17]), the issues associated with choosing or optimizing ad hoc prescriptions for applying that information beg the question of whether there is a self-consistent framework that would provide a natural motivation for a particular approach. We present such a framework, and additionally prescribe how the greater information available from probabilistic knowledge of data quality can be used without the need to cast that information into binary flags.

6.2. Optimal Searches with Imperfect Knowledge of non-Gaussian Noise

We now formulate the problem from a first-principles model of the noise processes within our detectors. As a reminder, we assume linear additive noise so that the detector output is given by $h = n + s + g$, where we only observe h and the auxiliary state $\vec{a}$, meaning we must marginalize over the unobserved latent processes n, s, and g. We begin by formulating probability distributions for detector noise in the target channel and auxiliary features conditioned on whether the IFO is in a *glitchy* or *clean* state.

In clean states, we assume the noise is generated by a stationary process, at least over timescale relevant to the filter, such that the autocorrelation function is given by

$$\mathcal{C}_{ij}(\tau) = \langle n(t_i) n(t_j = t_i + \tau) \rangle = \int df e^{2\pi i f \tau} S(f) \tag{15}$$

where $S(f)$ is the PSD. Adopting the Einstein summation convention, we then obtain

$$p(n|C) \propto \exp\left(-\frac{1}{2} n_j \mathcal{C}_{ij}^{-1} n_i\right) \tag{16}$$

by assuming Gaussianity. Furthermore, we expect n to be independent of $\vec{a}$ and declare $g = 0 \;\forall\; t \in C$ such that

$$p(n,s,g,h,\vec{a}|C) = p(n|C)p(\vec{a}|C)\delta(g)p(s)\delta(h-(n+s+g)) \tag{17}$$

where the astrophysical strain induced in the detector is assumed to be independent of the instantaneous detector behavior. We use iDQ's conditioned likelihood to model $p(\vec{a}|C) \sim p(\mathcal{M}(\vec{a})|C)$ and assume an astrophysically-motivated prior for signals $p(s)$.

In *glitchy* states, we still assume n is distributed as in clean times and that n is independent of ($\vec{a}$, s, g).

$$p(n,s,g,h,\vec{a}|G) = p(n|C)p(g|\vec{a},G)p(\vec{a}|G)p(s)\delta(h-(n+s+g)) \tag{18}$$

We note that the only difference is that we demand $p(g|C) = \delta(g)$ but leave $p(g|\vec{a},G)$ as a completely unknown function. Assuming that the set of data $\{g(t) \mid t \in G\}$ is distributed somehow, we can construct a combined likelihood spanning predefined labeling of both G and C samples as

$$p(n,s,g,h,\vec{a}) = p(n|C)\delta(h-(n+s+g))p(s)\left[\prod_{i\in C} p(\vec{a}_i|C)\delta(g_i)\right]\left[p(\{g_j|j\in G\}|\vec{a},G)\prod_{j\in G} p(\vec{a}_j|G)\right] \tag{19}$$

Marginalizing over latent processes yields

$$p(h,\vec{a}) = \int \mathcal{D}n\mathcal{D}s\mathcal{D}g\, p(n,s,g,h,\vec{a})$$

$$= \int \mathcal{D}s\, p(s) \prod_{i\in C} p(\vec{a}_i|C) \prod_{j\in G} p(\vec{a}_j|G) \int \mathcal{D}g \left(p(n = h - s - g|C) p(\{g_j|j \in G\}|\vec{a}, G) \prod_{i\in C} \delta(g_i) \right) \tag{20}$$

The result, in general, depends on how $g \in G$ is distributed, which is unknown. We note that assuming g is Gaussian-distributed with divergent variances, so that any g is equally likely, incurs a large Occam factor from the normalization. While not a problem when considering a single known permutation of which times are glitchy and which are clean, as was done in reference [58], this Occam factor can present issues when comparing different permutations, as we do below. Instead, we note that

$$\begin{aligned} \int \mathcal{D}g \left(p(n = h - s - g|C) p(\{g_j|j \in G\}|\vec{a}, G) \prod_{i\in C} \delta(g_i) \right) &\leq \max_{n_j \in G} \{p(n|C)\} \mid n_i = h_i - s_i \,\forall\, i \in C \\ &= \frac{1}{\sqrt{(2\pi)^N \det|\mathcal{C}|}} \exp\left(-\frac{1}{2} \sum_{i,j\in C} (h_i - s_i) \mathcal{C}^{-1}_{ij} (h_j - s_j) \right) \\ &\equiv \not{p}_C(n = h - s|\text{perm}) \end{aligned} \tag{21}$$

where $\not{p}_C$ is the distribution for n restricted to $n(t) \neq 0$ iff $t \in C$, which depends on the specific permutation of which times are clean and which are glitchy. This, then, implies

$$\begin{aligned} p(h, \vec{a}|\text{perm}) &= \int \mathcal{D}s\, p(s) \prod_{i\in C} p(\vec{a}_i|C) \prod_{j\in G} p(\vec{a}_j|G) \int \mathcal{D}g \left(\left[\prod_{i\in C} \delta(g_i) \right] p(\{g_j|j \in G\}|\vec{a}, G) p(n = h - s - g|C) \right) \\ &\leq \int \mathcal{D}s\, p(s) \prod_{i\in C} p(\vec{a}_i|C) \prod_{j\in G} p(\vec{a}_j|G)\ \not{p}_C(n = h - s|\text{perm}) \end{aligned} \tag{22}$$

which is not a proper distribution (i.e. normalizable) because of the maximization, but instead is an upper bound on the marginal likelihood for h and $\vec{a}$ given a permutation.

We, in effect, construct an inference only using times declared clean. This is necessary because we do not know how g is distributed within glitchy times. Alternatively, machine learning models that infer g directly from $\vec{a}$, or other assumptions about how g is distributed (e.g. [59]), would allow us to retain observations of $h \in G$ and marginalize over g directly. We note that maximized likelihood does not break the assumption of Gaussianity or stationarity; we simply restrict our selves to times known to be clean and treat glitchy times as if they were not observed. In spirit, then, this is similar to gating.

However, we only have probabilistic knowledge of which times are glitchy and which are clean based on h and $\vec{a}$. We therefore must marginalize over all permutations weighed by their relative prior probabilities. That is

$$p(h, \vec{a}) \leq \sum_{\text{perm}} \left(p(\text{perm}) \prod_{i\in C} p(\vec{a}_i|C) \prod_{j\in G} p(\vec{a}_j|G) \int \mathcal{D}s\, p(s)\ \not{p}_C(n = h - s|\text{perm}) \right)$$

with

$$p(\text{perm}) = p(C)^{N_C} p(G)^{N_G} \tag{23}$$

so that the resulting likelihood is

$$\begin{aligned} p(h, \vec{a}) &\leq \sum_{\text{perm}} \left(\prod_{i\in C} p(\vec{a}_i|C) p(C) \prod_{j\in G} p(\vec{a}_j|G) p(G) \int \mathcal{D}s\, p(s)\ \not{p}_C(n = h - s|\text{perm}) \right) \\ &= \sum_{\text{perm}} \left(p(\vec{a}|\text{perm}) p(\text{perm}) \int \mathcal{D}s\, p(s)\ \not{p}_C(n = h - s|\text{perm}) \right) \end{aligned}$$

and we remember that the specific sets C and G depend on the permutation. Note, beyond the conditioned likelihoods from iDQ, even non-trivial prior odds alone can affect the resulting inference. For example, this tells us that if $p(G) = p(C)/10$ and $p(\vec{a}|G) = p(\vec{a}|C)\ \forall\ t$, then one should marginalize over all possible covariance matrices that account for the prior knowledge that one glitch occurs for every 10 clean samples, on average.

Existing gating approaches assume exact data quality knowledge ($p(C|\vec{a})$ is either exactly 0 or exactly 1), thereby selecting a single permutation and a single noise model. In appendix D, we show how marginalization allowing for the presence of glitches, but without *a priori* knowledge of when the glitches occur, can effectively gate glitches automatically. This recovers and indeed expands upon current data quality approaches even without knowledge of auxiliary data. Proper prior odds also allow us to infer how many glitches are likely to be present and where they are likely to be, all without *a priori* knowledge beyond the relative frequencies of G and C samples.

With these marginal-maximized likelihoods in hand, we construct a likelihood ratio for the presence or absence of a signal in the data.

$$\Lambda^{S}_{!S} = \frac{\int \mathcal{D}s\, p(s) \sum_{\mathrm{perm}} p(\vec{a}|\mathrm{perm}) p(\mathrm{perm})\ \not{p}_C(n = h - s|\mathrm{perm})}{\sum_{\mathrm{perm}} p(\vec{a}|\mathrm{perm}) p(\mathrm{perm})\ \not{p}_C(n = h|\mathrm{perm})} \tag{24}$$

Appendix D.1 shows how the statistic behaves in the presence of glitches with either quiet or loud signals; in both cases it remains sensitive to GWs while being insensitive to the presence of glitches.

However, there is always the chance that our noise model remains insufficient to properly characterize IFO behavior, in which case other well-developed ad hoc signal consistency tests, such as χ^2 goodness-of-fit tests, could be combined with $\Lambda^S_{!S}$ within a larger likelihood ratio test, similar to how some searches currently include the matched filter ρ [56, 60], itself a proxy for the likelihood ratio in stationary Gaussian noise. If $\Lambda^S_{!S}$ is sufficient, additional variates like ad hoc χ^2 tests will not hurt the search's sensitivity. If it is not, they may continue to be vital.

Appendix D.2 shows how $\Lambda^S_{!S}$'s marginalization acts as an explicit signal consistency test without the need for additional ad hoc χ^2 tests. In effect, the marginalization over whether any particular data sample is declared glitchy or clean constitutes a signal consistency test conditioned on the rest of the clean data in that permutation: given the other clean data, is it more likely that the observed datum in question was generated by stationary Gaussian noise or a glitch?

The computational cost of direct marginalization may be prohibitively high, though, as the combinatorics of the number of different permutations grow exponentially with the length of the signal. Appendix D.3 discusses a few possible approximations that may render this more tractable.

We note that many pipelines may not assume the GW signal comes from a known template family and instead search for unmodeled transients [48, 61–63]. Such searches are often constructed by finding a maximum likelihood estimator for s and the corresponding maximum likelihood, subject to loose constraints on the GW waveform morphology or polarization to avoid degeneracies from underconstrained inferences. While we leave the explicit development of analogous searches to future work, we note that similar maximum likelihood techniques should work just as well with our marginal-maximized $\Lambda^S_{!S}$.

Similarly, one could formulate the probability of observing Gaussian noise in terms of a time-frequency decomposition of the data instead of the time-domain formulation provided here. In this case, one could construct an analogous marginal-maximized $\Lambda^S_{!S}$ where the individual time-frequency pixels were assigned glitch or clean labels, marginalizing over all permutations. Several parallel streams of probabilistic data quality, each targeting a specific frequency band, could be produced with parallel instances of the existing iDQ framework. Such a formulation is perhaps closer in spirit to the time-frequency glitch model considered in reference [59] than it is to gating, although it would only exacerbate any computational issues associated with direct marginalization already present in our time-domain formulation.

7. Discussion

We present a statistical learning framework to infer the presence of non-Gaussian noise within gravitational-wave detectors: iDQ. Using a supervised learning approach to classification, we decompose the inference into training, evaluation, calibration, and finally timeseries production. iDQ accounts for non-stationarity in the detectors by continuously re-training classifiers to autonomously detect and exploit new witnesses for non-Gaussian noise without human intervention. iDQ's framework can accommodate any supervised learning algorithm that operates on tabular data, and iDQ supports multiple modes of operation. Offline, or *batch* operation reproduces and expands upon previous functionality in the literature, fairly evaluating different algorithms' relative performance. Online, or *stream* operation manages multiple asynchronous processes to continuously re-train and re-calibrate statistical data quality inferences, producing robust probabilistic data quality information in real-time.

While some issues remain open, such as the best way to recover from detector non-stationarity (possibly through the use of weighted training sets) as well as the development and incorporation of novel feature sets and boosted classifiers, we show how iDQ has already proven invaluable within real GW searches. Specifically, we show several examples in which iDQ either autonomously reproduced the behavior of canonical data quality monitors without *a priori* knowledge of the type of noise in GW detectors or robustly identified noise sources otherwise unflagged by canonical monitors. Reference [55] describes how iDQ has already been included within some GW searches. We reiterate that iDQ has operated in low-latency throughout the entire advanced detector era and provided robust data quality information in low-latency for all detections to-date [7].

We also explore current methods of incorporating data quality information within GW searches, discussing their relative merits and drawbacks, and note that all rely on absolute knowledge of data quality. That is, most existing techniques implicitly require analysts to assume they know whether the detector is in a glitchy or clean state with absolute certainty at all times. iDQ moves beyond this assumption. We instead introduce a way to incorporate probabilistic data quality information that accounts for our imperfect knowledge of the presence or absence of non-Gaussian noise based on safe auxiliary channels alone. This approach presents several attractive features, automatically incorporating optimally located gates without *a priori* knowledge of where gates should be placed, as well as providing signal-consistency tests from first-principles noise models rather than ad hoc χ^2 tests. While we remark that the computational expense of direct marginalization over imperfect data quality information may be large, we also suggest several possible solutions, leaving their full development to future work.

With increasing detection rates and improved detector sensitivity, robust data quality information will only become more important over the next few years. Indeed, the importance of low-latency information for multi-messenger astronomy cannot be overstated. iDQ has provided robust low-latency probabilistic data quality information throughout the advanced detector era, and will continue to do so as GWs reveal new astrophysical phenomena in the most extreme environments found anywhere in the Universe.

Acknowledgments

The authors thank Ruslan Vaulin and acknowledge his vital contributions to earlier versions of iDQ. R E is supported at the University of Chicago by the Kavli Institute for Cosmological Physics through an endowment from the Kavli Foundation and its founder Fred Kavli. P G and C H are supported by the National Science Foundation (NSF) through OAC-1841480, OAC-1642391, and PHY-1454389. The authors also gratefully acknowledge the computational resources provided by the LIGO Laboratory and supported by NSF grants PHY-0757058 and PHY-0823459. Computations for this research were also performed on the Pennsylvania State University's Institute for Computational and Data Sciences Advanced CyberInfrastructure (ICDS-ACI).

Appendix A. Gaussian kernel density estimates

We review basic features of kernel density estimates (KDEs) and describe the particular one-dimensional Gaussian kernel implemented within iDQ. For convenience, iDQ adopts a fixed bandwidth (standard deviation) for all samples and imposes *reflecting boundary conditions* in order to avoid edge effects associated with the finite range of ranks ($r \in [0, 1]$). This is done by reflecting the samples across the bounds of their range so that

$$\{x_i\} \rightarrow \{x_i\} \oplus \{2X_{\mathrm{min}} - x_i\} \oplus \{2X_{\mathrm{max}} - x_i\}, \tag{25}$$

which forces the KDE's derivative to vanish at X_{min} and X_{max}.

We define a Gaussian kernel between two points (x, y) given a bandwidth (b) as

$$K(x, y; b) = \frac{1}{\sqrt{2\pi} b} e^{-(x-y)^2/2b^2}. \tag{26}$$

We consider a set of observed samples (x_i) with associated weights (w_i). The samples are assumed to be independently and identically distributed according to $p(x)$. iDQ assigns equal weights to each sample. Similarly, without loss of generality, we can set $\sum_i w_i = 1$, but this is not strictly necessary. We consider the following estimate for the probability density function $p(y)$ given b within the prior bounds $x_i \in [X_{\mathrm{min}}, X_{\mathrm{max}}]$.

$$\hat{p}(y|b, \{x_i\}) = \frac{1}{N} \sum_i w_i \left(K(y, x_i; b) + K(y, 2X_{\mathrm{min}} - x_i; b) + K(y, 2X_{\mathrm{max}} - x_i; b)\right) \tag{27}$$

$$\mathcal{N} = \int_{X_{\min}}^{X_{\max}} dy \sum_i w_i \left(K(y, x_i; b) + K(y, 2X_{\min} - x_i; b) + K(y, 2X_{\max} - x_i; b)\right) \tag{28}$$

where the observed samples x_i are explicitly reflected around the prior bounds. This is the basic estimator used within iDQ's continuous calibration maps (section 3.5.1) to model the conditioned likelihoods given observed evaluated sample sets. Furthermore, iDQ estimates the corresponding survival functions

$$\hat{P}(y|b, \{x_i\}) = \int_{y}^{X_{\max}} d\gamma\, \hat{p}(\gamma|b, \{x_i\}) \tag{29}$$

$$= \frac{1}{\mathcal{N}} \sum_i w_i \int_{y}^{X_{\max}} d\gamma \left(K(\gamma, x_i; b) + K(\gamma, 2X_{\min} - x_i; b) + K(\gamma, 2X_{\max} - x_i; b)\right) \tag{30}$$

with cumulative normal distributions computed during a single iteration over the sample set rather than numerical integration of $\hat{p}$. Given a bandwidth, iDQ generates a dense grid of ranks and evaluates these estimators once for each grid point. Calibration during timeseries production (section 3.6) can then be performed rapidly via linear interpolation without requiring repeated iteration over the sample set, which can be quite large: $\mathcal{O}(10^4)$. We now consider the choice of bandwidth and how to represent the uncertainty in our KDE representation given different realizations of the observed sample set. While all KDEs are biased estimators ($E[\hat{p}(y)] \neq p(y)$) because they smooth the true distribution according to $K(x, y; b)$, our bandwidth optimization procedure finds the best representation of p possible given our kernel. In practice, then, iDQ produces correct coverage (e.g. 50\% of samples have nominal survival functions $\leq$ 50\%) to within the expected statistical uncertainty for stationary distributions.

A.1. Bandwidth optimization

Common practice is to define a *leave-one-out* cross-validation likelihood and use this to optimize the bandwidth. We adopt the following likelihood

$$\log \mathcal{L}(b; \{x_i\}) = \frac{1}{\sum_i w_i} \sum_i w_i \log \left(\frac{1}{\sum_{j\neq i} w_j} \sum_{j \neq i} w_j K(x_i, x_j; b) \right) \tag{31}$$

As shown in reference [48], maximizing this likelihood is equivalent to minimizing the Kullback-Leibler divergence between the true distribution and our estimator, approximating an integral over the measure defined by the true distribution $p(x)$ with a Monte-Carlo integral over the observed sample set. The astute reader will note that we do not impose reflecting boundary conditions within $\log \mathcal{L}$. We expect the impact to be minor and the computational complexity is significantly lessened.

iDQ optimizes b separately for G and C samples through direct bisection searches within prespecified prior bounds. While this generates reliable estimators, we also note that $\log \mathcal{L}$ is often quite flat near its maximum and nearby bandwidths may produce similarly well behaved estimators. In principle, one could marginalize over the choice of bandwidth with respect to a prior

$$\hat{p}_{\mathrm{marg}}(y|\{x_i\}) = \int_{b_{\min}}^{b_{\max}} db\, p(b) \mathcal{L}(b; \{x_i\}) \hat{p}(y; b, \{x_i\}). \tag{32}$$

We expect marginalization to produce more robust estimators [64], although we find that using the b that maximizes $\log \mathcal{L}$ works well enough in practice and avoids the additional computational burden of direct numerical marginalization.

A.2. Representating an estimator's uncertainty as a β-distribution

Because iDQ has access to only a finite number of samples, $\hat{p}(x)$ will not perfectly reproduce $p(x)$. iDQ models this sample uncertainty with β-distributions. Let us consider the function

$$
\begin{aligned}
f(y;b,\{x_i\}) &= \hat{p}(y|b,\{x_i\})\frac{\sqrt{2\pi}b}{3} \\
&= \frac{\sqrt{2\pi}b}{3\sum_i w_i}\sum_i w_i\left(K(x_i,y;b)+K(2X_{\min}-x_i,y;b)+K(2X_{\max}-x_i,y;b)\right)
\end{aligned}
\tag{33}
$$

Because this statistic depends on the set of random variables x_i, it will also follow some distribution. We note that $f\in(0,1]$, and therefore a β-distribution is a good candidate for compactly representing the expected uncertainty. Specifically, we fit a β-distribution to estimates of the mean and variance of f, based on the observed sample set.

$$
\begin{aligned}
E[f(y;b)] &= \int\left(\prod_i dx_i p(x_i)\right) f(y;b,\{x_i\}) \\
&= \int\prod_i dx_i p(x_i)\frac{\sqrt{2\pi}b}{3\sum_j w_j}\sum_j w_j\left(K(x_j,y;b)+K(2X_{\min}-x_j,y;b)+K(2X_{\max}-x_j,y;b)\right) \\
&= \frac{\sqrt{2\pi}b}{3\sum_j w_j}\sum_j w_j\int dx_j p(x_j)\left(K(x_j,y;b)+K(2X_{\min}-x_j,y;b)+K(2X_{\max}-x_j,y;b)\right)\int\prod_{i\neq j}dx_i p(x_i) \\
&= \frac{\sqrt{2\pi}b}{3}\int dx p(x)\left(K(x,y;b)+K(2X_{\min}-x,y;b)+K(2X_{\max}-x,y;b)\right) \\
&\approx \frac{\sqrt{2\pi}b}{3\sum_j w_j}\sum_j w_j\left(K(x_j,y;b)+K(2X_{\min}-x_j,y;b)+K(2X_{\max}-x_j,y;b)\right)
\end{aligned}
\tag{34}
$$

where in the last line we approximate the integral over $p(x)$ as a weighed sum of our observed samples. We also calculate the second moment as

$$
\begin{aligned}
E\left[f^2\right] &= \int\left(\prod_i dx_i p(x_i)\right) f^2 \\
&= \left(\frac{\sqrt{2\pi}b}{3\sum_j w_j}\right)^2\Bigg[\left(\sum_j w_j^2\right)\int dx p(x)\left(K(x,y;b)+K(2X_{\min}-x,y;b)+K(2X_{\max}-x,y;b)\right)^2 \\
&\quad + \left(\sum_{j,k\neq j} w_j w_k\right)\left(\int dx p(x)\left(K(x_j,y;b)+K(2X_{\min}-x_j,y;b)+K(2X_{\max}-x_j,y;b)\right)\right)^2\Bigg] \\
&= \left(\frac{\sqrt{2\pi}b}{3\sum_j w_j}\right)^2\Bigg[\left(\sum_j w_j^2\right)\frac{1}{\sum_j w_j}\sum_i w_i(K(x_i,y;b)+K(2X_{\min}-x_i,y;b)+K(2X_{\max}-x_i,y;b))^2 \\
&\quad + \left(\left(\sum_j w_j\right)^2-\sum_j w_j^2\right)\left(\frac{1}{\sum_j w_j}\sum_i\left(K(x_i,y;b)+K(2X_{\min}-x_i,y;b)+K(2X_{\max}-x_i,y;b)\right)\right)^2\Bigg]
\end{aligned}
\tag{35}
$$

and thereby obtain the variance directly via $V[f(y;b)]=E\left[f^2\right]-E[f]^2$. In the case of equal weights with N samples, this yields

$$
\begin{aligned}
V[f(y;b)] = \frac{2\pi b^2}{N}\Bigg[&\frac{1}{N}\sum_i\left(K(x_j,y;b)+K(2X_{\min}-x_j,y;b)+K(2X_{\max}-x_j,y;b)\right)^2 \\
&-\left(\frac{1}{N}\sum_i\left(K(x_j,y;b)+K(2X_{\min}-x_j,y;b)+K(2X_{\max}-x_j,y;b)\right)\right)^2\Bigg].
\end{aligned}
$$

We note that $V_{\{x\}}[f(y;b)] \propto N^{-1} V_x[K(y,x;b)]$, as expected for a Fisher-efficient estimator. Now, the β-distribution defined by

$$\mathrm{Beta}(x;\alpha,\beta) \propto x^{\alpha-1}(1-x)^{\beta-1} \quad \Big| \quad x \in [0,1] \tag{36}$$

has mean and variance

$$E[x] = \frac{\alpha}{\alpha+\beta} \tag{37}$$

$$V[x] = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{38}$$

We choose α and β to reproduce $E[f]$ and $V[f]$. The distribution of our estimator $\hat{p}(y;b,\{x_i\})$ is therefore given by

$$\hat{p}(y;b,\{x_i\}) \sim \left(\frac{3}{\sqrt{2\pi b}}\right) \mathrm{Beta}(\alpha(b,\{x_i\}),\beta(b,\{x_i\})) \tag{39}$$

where the fit parameters are a function of the bandwidth and observed samples. We note that some estimates of the mean and variance have no corresponding solution in terms of α and β, typically corresponding to expectation values very close to 0 or 1. We therefore impose a minimum expectation value of $\sim 10^{-6}$ for numerical stability.

iDQ estimates the best-fit α and β parameters at every point on the dense grid over ranks used to compute $\hat{p}$, interpolating between neighboring grid points as needed. This provides rapid estimates of sampling uncertainty at the same time as point estimates within timeseries production. A similar procedure is used to model uncertainty in the cumulative distribution $\hat{P}(x)$. Sample uncertainty for the likelihood ratio Λ^G_C (equation (6)) is obtained via Monte-Carlo sampling from the β-distributions representing $\hat{p}(r|G)$ and $\hat{p}(r|C)$. Again, these uncertainty measures are computed once for each grid point and interpolated as needed.

Appendix B. Updates to the Ordered Veto List (OVL) algorithm

The Ordered Veto List (OVL, [29]) algorithm has been updated since originally published. These changes provide greater flexibility within the algorithm as well as parallelization and other computational optimizations. We refer readers to reference [29] for an introduction to OVL and focus only on the updates in what follows.

B.1. Veto performance metrics

First, we note that OVL is very similar in concept to both hVeto [30] and UPV [31], in that this class of algorithm develops a hierarchically applied list of *veto configurations*, each consisting of a single auxiliary channel, a significance threshold for triggers in that channel, and a symmetric time window used to construct veto segments around noise in the auxiliary channel. By direct optimization over many channel–threshold–window tuples, the algorithms identify a preferred order in which to apply the veto conditions. As discussed in reference [29], the ordering depends on the metric used to rank channel performance, and, indeed, this is the main difference between the original OVL implementation, hVeto, and UPV. Specifically, OVL originally used the ratio of the marginal efficiency to the marginal deadtime. That is, the fraction of remaining noise transients removed to the fraction of remaining time removed by the veto configuration. This behaves similarly to a likelihood ratio test and optimizes ROC curves, subject to algorithmic constraints. hVeto uses a measure of the Poisson significance of removing coincident noise transients that, as discussed in reference [29], favors veto conditions that remove many noise transients given a fixed efficiency–deadtime ratio. UPV orders configurations by the ratio of the number of target transients removed to the number of auxiliary transients present, thereby preferring more deterministic couplings. However, there are counterexamples (e.g. figure 9).

Because each metric has its own merits, the updated OVL algorithm now allows users to specify the metric used to rank configurations, thereby reproducing the behavior of the original OVL, hVeto, and UPV within a single framework. We note that OVL uses exact segment logic when constructing segments, unlike hVeto [30], and defines the use percentage slightly differently than UPV. OVL computes the use percentage as the ratio of target transients removed to the effective number of auxiliary transients present, defined as the quotient of the vetoed time associated with an entire veto configuration to the window used to construct the veto segments. The effective number of auxiliary transients, then, clusters nearby auxiliary noise to avoid overcounting if many neighboring auxiliary disturbances within a single channel produce nearly identical veto segments. Anecdotally, we find that ranking by either the efficiency-to-deadtime ratio or the use percentage routinely produce better ROC curves than ranking by the Poisson significance, in agreement with reference [29].

B.2. Training

OVL's training scheme has also been slightly updated, although it still remains largely as described in reference [29]. Specifically, OVL still trains via a nested iteration, evaluating veto configurations' performance hierarchically with a given ordering, pruning ineffective configurations, and re-ordering the list within each epoch. Pruning is done to avoid over-training and is accomplished by setting minima on various veto configuration performance metrics, like the efficiency–deadtime ratio or Poisson significance. The precise impact of these minima has not been quantified, but typically analysts select values to balance the loss in efficiency associated with restricting the *vetolist* to only the most exceptionally well-ranked veto configurations and the generalization error introduced by over-fitting, as efficient vetoes can sometimes correspond to statistically rare accidental coincidences within the training set that do not generalize well. For this reason, it is thought that pruning based on the Poisson significance has the largest impact on over-training.

OVL learns when it re-orders its list, as this places higher ranked veto conditions first. Because of the hierarchical nature of OVL, we take care to re-order the veto configurations to preserve as much information as possible. Specifically, within each re-ordering, we first sort the list to place high-threshold, small-window configurations first. All else being equal, these should produce better veto configurations. Only then do we order the configurations by their metrics, so that configurations with the same score are ordered to prefer high thresholds and short windows. Pragmatically, we find this makes a small but noticeable difference in the final ordering produced by the algorithm.

B.3. Ranks

Because iDQ requires classifiers to generate ranks within the unit interval and the metrics used within OVL typically span the positive real line, we map the metric scores (m) into ranks (r) according to

$$r = \frac{m}{\xi + m}, \tag{40}$$

where the scale (ξ) is fixed for each metric separately to account for their very different dynamic ranges seen with typical interferometric data. We note that this mapping is not unique, and other functional forms would accomplish the same task. However, equation (40) distributes the ranks more uniformly over the unit interval than some other mappings, and this can help iDQ's calibration build accurate representations of the resulting conditional likelihoods.

B.4. Feature importance

Finally, we would be remiss if we did not discuss OVL's notion of feature importance, one of the most attractive aspects of the algorithm besides its robust performance. Because OVL only considers a single auxiliary channel at a time and applies them in a specific order, it is straightforward to determine which auxiliary features (channel, threshold, and window) are responsible for OVL's predicted rank at any time. In this way, OVL reports which veto configurations were active as a function of time while simultaneously reporting their relative importances as their ranks. Indeed, feature importance as a function of time is shown in figures 7 and 8.

OVL also measures correlations between veto conditions. Specifically, it can report the intersection of segments created by each veto configuration as a symmetric matrix. Diagonal elements correspond to the time contained within of each set of veto segments separately. Nearly redundant veto configurations, then, will produce intersection times close to the times of each configuration separately. We note that OVL's training algorithm, by design, will remove redundant configurations and therefore reduce the amount of overlap in the list by applying configurations hierarchically, removing vetoed transients and time before proceeding to the next configuration. This, combined with pruning, will tend to select a single witness configuration out of sets of highly correlated configurations. Nonetheless, such 'covariance matrices' between veto configurations may prove useful when diagnosing the source of specific noise transients identified by the selected auxiliary witnesses. Figure B1 shows examples of correlations between the veto configurations used surrounding GW170817.

Appendix C. Synthetic feature generation

In addition to supporting multiple possible feature sources, each supplying distinct sets of features, iDQ can also generate synthetic data on-the-fly. This is done for testing purposes and to benchmark algorithms. Briefly, iDQ simulates an arbitrary number of stationary Poisson processes representing sources of noise. Each of these synthetic processes is described by a separate rate as well as distributions over frequency and ρ. Synthetic processes are then witnessed by user-specified sets of channels, and each witness records values scattered around the true value within the process (e.g. central times recorded in witness channels are Gaussian-distributed around the central times produced by the process, with separate standard deviations for each witness). In this way, the synthetic processes entangle the features witnessed by several channels, and noise in a single channel can be modeled by a separate process witnessed only by that channel. Each channel may witness multiple streams, generating arbitrarily complex correlations within the feature set. This implementation realizes the probabilistic graphical model depicted in figure 1.

Appendix D. Futher discussion of optimal searches

We now explore some attractive features and limiting cases of $\Lambda^{S}_{!S}$ (equation (24)) in order to build further intuition for how marginalization over probabilistic data quality information benefits searches. We note that if $p(C|\vec{a})$ is *binary*, that is we assume perfect knowledge of which data is clean and which is glitchy, only a single permutation retains non-trivial probability. Current gating schemes, then, are equivalent to assuming perfect knowledge of data quality within the detectors at all times, at best an exaggeration of the current state of the field since the sources of many non-Gaussian noise transients remain unknown (e.g. [40, 65]).

We also note that, assuming trivial conditioned likelihoods from iDQ ($p(r|C) = p(r|G)\ \forall\ r$), the weight assigned to each permutation is $p(\text{perm}) = p(C)^{N_C} p(G)^{N_G} = p(C)^{N_C}(1-p(C))^{N-N_C}$, which is just the binomial distribution with N trials, N_C successes, and a probability of success given by $p(C)$. This has the appealing interpretation of marginalizing over the number and placement of glitches given knowledge about their relative frequency but nothing else. Indeed, this is the most basic piece of data quality information that could be incorporated and, as we will see, it could already significantly improve search backgrounds.

D.1. Toy model

Let us consider a toy model of stationary white noise in three observed data. We assume constant prior odds for G vs. C, but otherwise assume $\vec{a}$ is uninformative. The marginal-maximized likelihood then becomes

$$\begin{aligned}
p(h,\vec{a}) = \frac{1}{(2\pi)^{3/2}\sigma^3}\Bigg[& p(C)^3 \exp\left(-\frac{|h_1-s_1|^2+|h_2-s_2|^2+|h_3-s_3|^2}{2\sigma^2}\right) \\
& + p(C)^2 p(G)\left(\exp\left(-\frac{|h_1-s_1|^2+|h_2-s_2|^2}{2\sigma^2}\right)\right. \\
& \left. + \exp\left(-\frac{|h_1-s_1|^2+|h_3-s_3|^2}{2\sigma^2}\right) + \exp\left(-\frac{|h_2-s_2|^2+|h_3-s_3|^2}{2\sigma^2}\right)\right) \\
& + p(C)p(G)^2\left(\exp\left(-\frac{|h_1-s_1|^2}{2\sigma^2}\right) + \exp\left(-\frac{|h_2-s_2|^2}{2\sigma^2}\right) + \exp\left(-\frac{|h_3-s_3|^2}{2\sigma^2}\right)\right) \\
& + p(G)^3\Bigg]
\end{aligned}$$

Now, let us further assume $t_1,\ t_3 \in C$ and $t_2 \in G$ for concreteness such that

$$h_1 - s_1 = n_1 \sim \sigma \tag{41}$$

$$h_2 - s_2 = n_2 + g_2 \gg \sigma \tag{42}$$

$$h_3 - s_3 = n_3 \sim \sigma \tag{43}$$

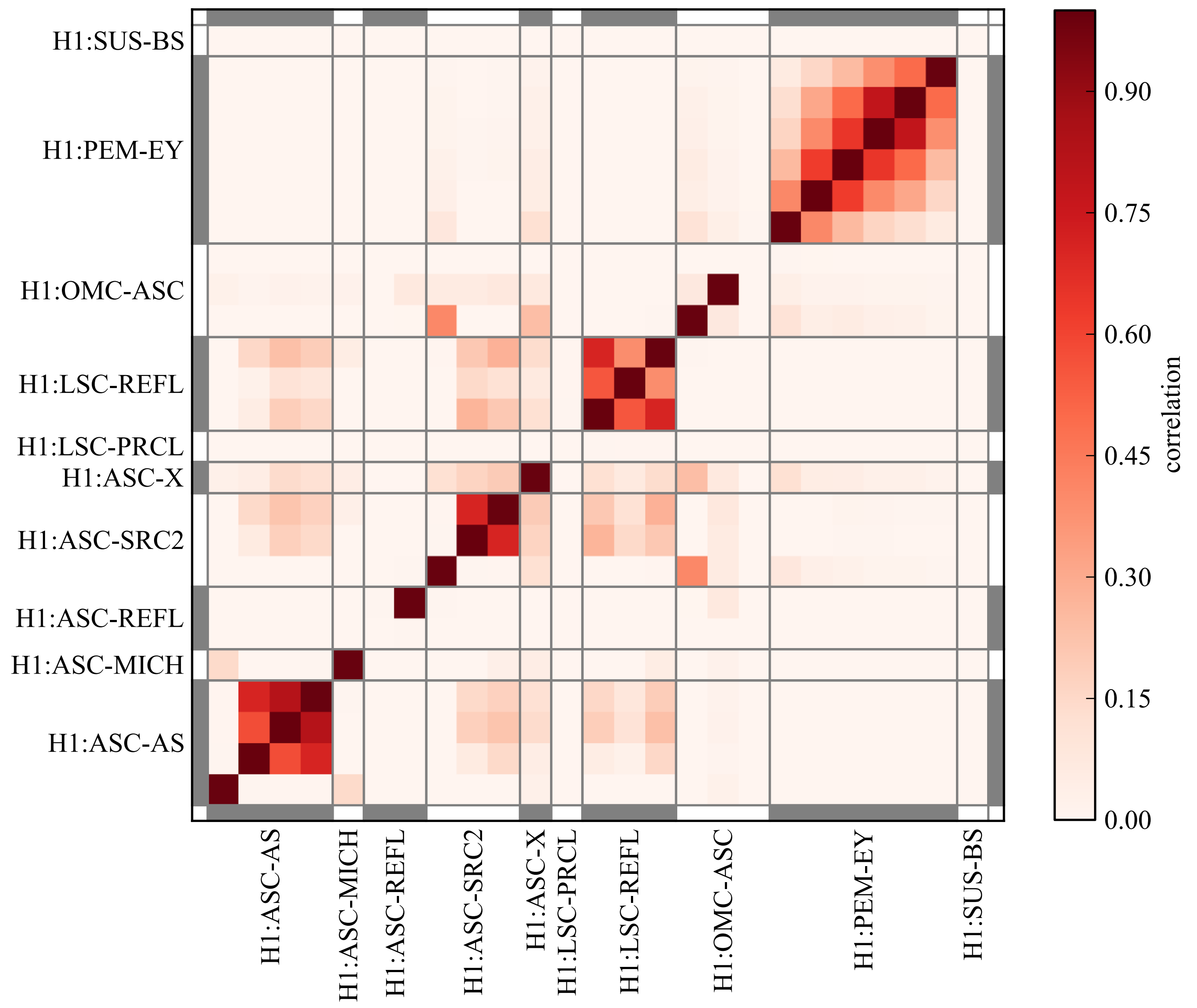

Figure B1. Example OVL correlation matrix showing the inter-relatedness of various veto conditions. Data is taken from the LIGO Hanford observatory on 16-18 August 2017 and the veto configurations included are based on the OVL vetolist used in low-latency during GW170 817. For brevity, we group veto configurations by their channel subsystems and order them alphabetically. Diagonal elements with zero correlation correspond to veto conditions that were not active at any time in the 2×10^5 seconds analyzed.

In this case, we obtain

$$\begin{aligned} p(h,\vec{a}) \approx \frac{1}{(2\pi)^{3/2}\sigma^3} \Bigg[& p(C)^3 \exp\left(-\frac{|g_2|^2}{2\sigma^2}\right) + p(C)^2 p(G)\left(\exp\left(-\frac{|n_1|^2+|n_3|^2}{2\sigma^2}\right) + 2\exp\left(-\frac{|g_2|^2}{2\sigma^2}\right)\right) \\ & + p(C)p(G)^2\left(\exp\left(-\frac{|n_1|^2}{2\sigma^2}\right) + \exp\left(-\frac{|g_2|^2}{2\sigma^2}\right) + \exp\left(-\frac{|n_3|^2}{2\sigma^2}\right)\right) + p(G)^3 \Bigg] \\ \approx \frac{1}{(2\pi)^{3/2}\sigma^3} \Bigg[& p(C)^3 e^{-|g_2|^2/2\sigma^2} + p(C)^2 p(G) e^{-(|n_1|^2+|n_3|^2)/2\sigma^2} + p(C)p(G)^2\left(e^{-|n_1|^2/2\sigma^2} + e^{-|n_3|^2/2\sigma^2}\right) \\ & + p(G)^3 \Bigg] \end{aligned}$$

If we now assume glitches are relatively rare *a priori* ($p(C)/p(G) \gg e^{-1/2}$), then the second term dominates over the third and fourth terms. Because the glitch is loud, we additionally have $e^{-|g_2|^2/2\sigma^2} \ll p(G)/p(C)$ and the second term also dominates the first term, yielding

$$p(h,\vec{a}) \approx \frac{p(C)^2 p(G)}{(2\pi)^{3/2}\sigma^3} \exp\left(-\frac{|h_1-s_1|^2+|h_3-s_3|^2}{2\sigma^2}\right) \tag{44}$$

which is equivalent to the what we would obtain if we knew the correct sample to gate *a priori*, even though we did not, up to a normalization constant. Specifically, the marginalization automatically detects the correct placement for gates based on the relative frequency of G and C samples and the data's consistency with Gaussian noise without any other *a priori* knowledge. We note that, if the signal is quiet ($h - s \sim h \sim \sigma$), then we obtain

$$\log\Lambda^S_{!S} \approx -\frac{|h_1-s_1|^2+|h_3-s_3|^2}{2\sigma^2} + \frac{|h_1|^2+|h_3|^2}{2\sigma^2} \tag{45}$$

exactly as expected for stationary white Gaussian noise with the glitch gated with precise *a priori* knowledge of the glitch's location. If instead the signal is loud, the inference is more complicated as the noise-only model may confuse what is really a loud signal with a loud glitch, although we have

$$\Lambda^S_{!S} \approx \left(\frac{p(C)}{p(G)}\right)^2 \exp\left(-\frac{|h_1-s_1|^2+|h_3-s_3|^2)}{2\sigma^2}\right) \sim \left(\frac{p(C)}{p(G)}\right)^2 \gg 1 \tag{46}$$

which is still large due to the prior odds and therefore still strongly in favor of a signal. While the full solution with more samples and colored noise is more challenging technically, is follows the same basic principles.

We again note that the benefits of marginalization seen within this toy model assume uninformative auxiliary information and simply accounts for the possibility that glitches exist within the detectors and our imperfect knowledge of data quality. Informative supervised learning models based on $\vec{a}$ with correct calibration, such as those provided by iDQ, can only further improve the inference. In the case of our toy model, this would simply add additional weight to the correct permutation that gated the second sample.

D.2. Signal consistency tests

We remark that signal consistency tests, like χ^2 goodness-of-fit, require the data to be consistent over several smaller, independent trials, looking at the distribution of $\sum_t \rho(t)^2$ rather than $(\sum_t \rho(t))^2$. We note that such χ^2 statistics are ad hoc and not uniquely defined. Therefore, there is no particular reason we should expect to derive the form of any such statistic from first-principles considerations. Nonetheless, we show that marginalization naturally defines a signal consistency requirement similar in spirit to, but different in detail from, existing χ^2 tests.

Let us further consider the model comparisons implicit with the marginalization over permutations. As an example, let us assume that the G or C assignments are known perfectly for all samples except one: h_k. The explicit marginalization over this single unknown sample is then

$$p(h,\vec{a}) = p(\vec{a}_k|C)p(C)\left(\frac{\exp\left(-\frac{1}{2}\sum_{i,j\in C+k}(h_i-s_i)\mathcal{C}_{ij}(h_j-s_j)\right)}{\sqrt{(2\pi)^N \det|\mathcal{C}|}}\right)$$

$$+p(\vec{a}_k|G)p(G)\left(\frac{\exp\left(-\frac{1}{2}\sum\limits_{i,j\in C}(h_i-s_i)\mathcal{C}_{ij}(h_j-s_j)\right)}{\sqrt{(2\pi)^N\det|\mathcal{C}|}}\right)$$

$$=p(\vec{a}_k|C)p(C)\ \not{p}_C(n=h-s)\left(\exp\left(-\sum_{i\in C}(h_i-s_i)\mathcal{C}^{-1}_{ik}(h_k-s_k)-\frac{1}{2}(h_k-s_k)^2\mathcal{C}^{-1}_{kk}\right)+\frac{p(G|\vec{a}_k)}{p(C|\vec{a}_k)}\right) \tag{47}$$

We note that

$$\exp\left(-\sum_{i\in C}(h_i-s_i)\mathcal{C}^{-1}_{ik}(h_k-s_k)-\frac{1}{2}(h_k-s_k)^2\mathcal{C}^{-1}_{kk}\right)=\sqrt{2\pi\frac{\det|\mathcal{C}|_{C+k}}{\det|\mathcal{C}|_C}}p(h_k-s_k|\{h_i-s_i\,\forall\, i\in C\}) \tag{48}$$

which is the likelihood of observing $(h_k - s_k)$ as part of the stationary Gaussian noise process conditioned on the observations of the rest of the Gaussian noise process known to be clean $(h_i - s_i\ \forall\ i \in C)$ multiplied by the prior volume allowed by the additional degrees of freedom. This is a signal consistency test that checks whether the k^{th} sample agrees with the signal seen in the other N_C samples. Marginalization compares this consistency test against the posterior odds that the k^{th} sample was a glitch based on $\vec{a}_k$, effectively placing a lower bound on the probability of seeing *any* $h_k - s_k$. Within likelihood ratio tests, this prevents the noise-only model from becoming vanishingly small in the presence of loud glitches, thereby preventing the likelihood ratio from diverging and rendering the search much less sensitive to glitches. Indeed, this is exactly the behavior seen in our toy model.

D.3. Computational cost of marginalization

We note that the marginalization over all permutations proposed within $\Lambda^S_{!S}$ is combinatorially expensive. Efficiently implementing such a calculation is an open problem, but we discuss a few possible solutions below.

First, one could Monte-Carlo integrate instead of performing the entire sum. However, Monte-Carlo integrals have variances that scale as

$$\mathrm{Var}\left[\frac{1}{N}\sum_i f_i\right]\sim\frac{1}{N}\mathrm{Var}\,[f] \tag{49}$$

The integral's variance, then, could be quite large in the presence of loud glitches as the variance between different permutations would be large. Such integrals may require many samples to converge.

Alternatively, one could sample from the sum in a scheme similar to the Metropolis-Hasting algorithm [66]. Jump proposals would consist of flipping the label of one sample from G to C or vice versa, much like an Ising spin model [67]. However, Markov-Chain Monte-Carlo estimates of marginal likelihoods are not without their own computational challenges and may not scale well in practice.

Regardless of the sampling procedure, we note that the number of possible permutations could be exponentially reduced by labeling small contiguous segments as G or C instead of labeling every time sample separately. This would greatly reduce the computational cost, perhaps to something tractable, but introduces issues of how to select the window size. Based on our considerations of model comparisons occurring within the marginalization, windows of comparable size to the stationary Gaussian noise's autocorrelation times may be appropriate, as this is the relevant timescale over which $h_k|h_{i\in C}$ becomes less in formed by $h_{i\in C}$ and therefore less stringent of a test.

We would need to compute the probability that there was a glitch present at any time within each segment. This coarse-graining should be straightforward, though, as

$$p(G\in\text{window}|\vec{a})=1-\prod_{t_i\in\text{window}}p(C|\vec{a}(t_i)) \tag{50}$$

A related approach would be to round the probabilities up or down based on some threshold. That is, if $p(C|\vec{a})$ is above some threshold, we only consider permutations where that sample is labeled C. Similarly, if $p(C|\vec{a})$ is below another threshold, we only consider permutations where that sample is labeled G, marginalizing over only the samples in the middle. In effect, this would define

$$p_{\mathrm{eff}}(C|\vec{a}) = \begin{cases} 1 & p(C|\vec{a}) \geq p_{\max} \\ p(C|\vec{a}) & p_{\min} < p(C|\vec{a}) < p_{\max} \\ 0 & p(C|\vec{a}) \leq p_{\min} \end{cases} \tag{51}$$

There is no single obvious choice of thresholds, however, so care would be needed. We note that this is similar to the *auto-gating* implemented within some existing searches [56, 60], although the thresholds are placed on ρ and are not currently determined by $p(G|\vec{a})/p(C|\vec{a})$. What's more, they ignore the conditioning on other data already declared clean. Indeed, this could be considered a conservative choice as the threshold to declare a sample clean based on the conditioned likelihood $h_k|h_{i\in C}$ should only be more stringent than a threshold derived without the observations of $h_{i\in C}$.

Such coarse-graining is likely to result in information loss and therefore less sensitive searches. However, the impact may be small enough and the computational speed-ups large enough to make this tractable, thereby improving GW search sensitivity compared to current approaches that do not marginalize over imperfect data quality information. reference [55] implements one such coarse-graining procedure, although they do not attempt to marginalize over probabilistic data quality and instead directly modify their likelihood ratio with a multiplicative factor that depends on iDQ's output. Even this simple approach already shows modest improvements in search sensitivity.

ORCID iD

Reed Essick https://orcid.org/0000-0001-8196-9267

References

[1] Aasi J *et al* 2015 Advanced LIGO *Class. Quantum Grav.* **32** 074001

[2] Acernese F *et al* 2015 Advanced Virgo: a second-generation interferometric gravitational wave detector *Class. Quantum Grav.* **32** 024001

[3] Matichard F *et al* 2015 Seismic isolation of Advanced LIGO: Review of strategy, instrumentation and performance *Class. Quant. Grav.* **32** 185003

[4] Graef Rollins J 2016 arXiv:1604.01456 [astro-ph.IM] Distributed State Machine Supervision for Long-baseline Gravitational-wave Detectors https://arxiv.org/abs/1604.01456)

[5] Viets A *et al* 2018 Reconstructing the calibrated strain signal in the Advanced LIGO detectors *Class. Quant. Grav.* **35** 095015

[6] Abbott B P *et al* 2016 Observation of Gravitational Waves from a Binary Black Hole Merger *Phys. Rev. Lett.* **116** 061102

[7] Abbott B P *et al* 2019 GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs *Phys. Rev.* X **9** 031040

[8] Abbott B P *et al* 2017 GW170817: Observation of Gravitational Waves from a Binary Neutron Star Inspiral *Phys. Rev. Lett.* **119** 161101

[9] Abbott B *et al* 2020 GW190425: Observation of a Compact Binary Coalescence with Total Mass ~ 3.4 Msun *Astrophys. J. Lett.* **892** L3

[10] Abbott B P *et al* 2017 Gravitational Waves and Gamma-Rays from a Binary Neutron Star Merger: GW170817 and GRB 170817A *Astrophys.* J. **848** L13

[11] Coulter D A *et al* 2017 Swope Supernova Survey 2017a (SSS17a), the optical counterpart to a gravitational wave source *Science* **358** 1556

[12] Goldstein A*et al* 2017 An Ordinary Short Gamma-Ray Burst with Extraordinary Implications: Fermi-GBM Detection of GRB170817A *Astrophys. J.* **848** L14

[13] Abbott B P*et al* 2016 Sensitivity of the Advanced LIGO detectors at the beginning of gravitational wave astronomy *Phys. Rev.* **D93** 112004

[14] Abbott B P*et al* 2018 Sensitivity of the Advanced LIGO detectors at the beginning of gravitational wave astronomy (Addendum) *Phys. Rev.* **D97** 059901

[15] Abbott B P *et al* 2020 A guide to LIGO-Virgo detector noise and extraction of transient gravitational-wave signals *Class. Quant. Grav.* **37** 055002

[16] Abbott B P *et al* 2019 All-sky search for short gravitational-wave bursts in the second Advanced LIGO and Advanced Virgo run *Phys. Rev.* D **100** 024017

[17] Abbott B P *et al* 2018 Effects of data quality vetoes on a search for compact binary coalescences in Advanced LIGO's first observing run *Class. Quantum Grav.* **35** 065010

[18] Powell J *et al* 2015 Classification methods for noise transients in advanced gravitational-wave detectors *Class. Quant. Grav.* **32** 215012

[19] Powell J et al 2017 Classification methods for noise transients in advanced gravitational-wave detectors II: performance tests on Advanced LIGO data *Class. Quant. Grav.* **34** 034002

[20] Zevin M *et al* 2017 Gravity Spy: integrating advanced LIGO detector characterization, machine learning, and citizen science *Class. Quantum Grav.* **34** 064003

[21] Mueller C L *et al* 2016 The Advanced LIGO Input Optics *Rev. Sci. Instrum.* **87** 014502

[22] Staley A 2015 Locking the advanced LIGO gravitational wave detector: with a focus on the arm length stabilization technique *PhD thesis*, Columbia U (https://doi.org/10.7916/D8X34WQ4)

[23] Effler A et al 2015 Environmental Influences on the LIGO Gravitational Wave Detectors during the 6th Science Run *Class. Quant. Grav.* **32** 035017

[24] Essick R and Godwin P 2018 iDQ Source Code (https://git.ligo.org/reed.essick/iDQ)

[25] Essick R and Godwin P 2018 iDQ Documentation (https://docs.ligo.org/reed.essick/iDQ)
[26] Abbott B P *et al* 2018 Prospects for observing and localizing gravitational-wave transients with Advanced LIGO, Advanced Virgo and KAGRA *Living Rev. Relativity* **21** 3
[27] Essick R 2017 *PhD Theses, MIT* (http://hdl.handle.net/1721.1/115024) Detectability of Dynamical Tidal Effects and the Detection of Gravitational-Wave Transients With LIGO
[28] Biswas R *et al* 062003 Application of machine learning algorithms to the study of noise artifacts in gravitational-wave data *Phys. Rev.* D **88** 2013
[29] Essick R *et al* 2013 Optimizing vetoes for gravitational-wave transient searches *Class. Quantum Grav.* **30** 155010
[30] Smith J R et al 2011 A hierarchical method for vetoing noise transients in gravitational-wave detectors *Class. Quantum Grav.* **28** 235005
[31] Isogai T*et al* and 2010 Used percentage veto for LIGO and Virgo binary inspiral searches *J. Physics: Conf. Series* **243** 012005
[32] Cavaglia M *et al* 2019 Finding the origin of noise transients in LIGO data with machine learning *Commun. Comput. Phys.* **25** 963–87
[33] Colgan R E *et al* 2019 Efficient Gravitational-wave Glitch Identification from Environmental Data Through Machine Learning *Phys. Rev.* D **101** 102003
[34] Abbott B *et al* 2017 Multi-messenger Observations of a Binary Neutron Star Merger *Astrophys.* J. **848** L12
[35] Vajente G 2020 Machine-learning nonstationary noise out of gravitational-wave detectors *Phys. Rev.* D **101** 042003
[36] Ormiston R *et al* 2020 Noise Reduction in Gravitational-wave Data via Deep Learning *Phys. Rev. Res.* **2** 033066
[37] Chatterji S et al 2004 Multiresolution techniques for the detection of gravitational-wave bursts *Class. Quantum Grav.* **21** S1809
[38] Godwin P 2020 Low-latency statistical data quality in the era of multi-messenger astronomy *PhD Thesis* Penn State University
[39] Essick R 2020 *in prep.* A Coincidence Null-Test for Poisson-Distributed Events
[40] Abbott B P *et al* 2016 Characterization of transient noise in Advanced LIGO relevant to gravitational wave signal GW150914 *Class. Quantum Grav.* **33** 134001
[41] Neyman J and Pearson E S 1933 On the Problem of the Most Efficient Tests of Statistical Hypotheses *Phil. Trans. Roy. Soc. Lond.* **A231** 289
[42] Blackburn L 2007 Kleine-Welle algorithm (https://dcc.ligo.org/public/0027/T060221/000/T060221-00.pdf)
[43] Fisher R *et al in prep.* DQSEGDB: A time-interval database for storing gravitational wave observatory metadata
[44] Pedregosa F *et al* 2011 Scikit-learn: Machine Learning in Python *J. Machine Learning Res.* **12** 2825
[45] Chollet F *et al* 2015 (https://keras.io) Keras
[46] Chen T and Guestrin C 2016 XGBoost: A Scalable Tree Boosting System *Proc. of the 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* pp 785–94
[47] Tulio Ribeiro M *et al* 2016 "Why Should I Trust You?": Explaining the Predictions of Any Classifier https://arxiv.org/abs/1602.04938)
[48] Lynch R *et al* 2017 Information-theoretic approach to the gravitational-wave burst detection problem *Phys. Rev.* D **95** 104046
[49] Pankow C *et al* 2018 Mitigation of the instrumental noise transient in gravitational-wave data surrounding GW170817 *Phys. Rev.* **D98** 084016
[50] Gracedb (https://gracedb.ligo.org/)
[51] Essick R 2017 (on behalf of the LIGO-Virgo Scientific Collaborations) (https://gcn.gsfc.nasa.gov/gcn3/21509.gcn3)
[52] Piotrzkowski B 2019 (on behalf of the LIGO-Virgo Scientific Collaborations) (https://gcn.gsfc.nasa.gov/gcn3/25442.gcn3)
[53] Abbott B P *et al* 2016 GW151226: Observation of Gravitational Waves from a 22-Solar-Mass Binary Black Hole Coalescence *Phys. Rev. Lett.* **116** 241103
[54] Abbott B P *et al* 2019 Low-latency Gravitational-wave Alerts for Multimessenger Astronomy during the Second Advanced LIGO and Virgo Observing Run *Astrophys.* J. **875** 161
[55] Godwin P *et al* 2020 *in prep.* Incorporation of Statistical Data Quality Information into the GstLAL Search Analysis
[56] Messick C *et al* 2017 Analysis Framework for the Prompt Discovery of Compact Binary Mergers in Gravitational-wave Data *Phys. Rev.* **D95** 042001
[57] Usman S A *et al* 2016 The PyCBC search for gravitational waves from compact binary coalescence *Class. Quantum Grav.* **33** 215004
[58] Zackay B *et al* 2019 Detecting Gravitational Waves in Data with Non-Gaussian Noise (arXiv:1908.05644)
[59] Cornish N J and Littenberg T B 2015 Bayeswave: Bayesian inference for gravitational wave bursts and instrument glitches *Class. Quantum Grav.* **32** 135012
[60] Sachdev S *et al* 2019 The GstLAL Search Analysis Methods for Compact Binary Mergers in Advanced LIGO's Second and Advanced Virgo's First Observing Runs (https://arxiv.org/abs/1901.08580)
[61] Klimenko S *et al* 2016 Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors *Phys. Rev.* D **93** 042004
[62] Sutton P J *et al* 2010 X-Pipeline: an analysis package for autonomous gravitational-wave burst searches *New J. Phys.* **12** 053034
[63] Thrane E *et al* 2011 Long gravitational-wave transients and associated detection strategies for a network of terrestrial interferometers *Phys. Rev.* D **83** 083004
[64] Abbott B P *et al* 2019 Constraining the *p*–Mode—*g*–Mode Tidal Instability with GW170817 *Phys. Rev. Lett.* **122** 061104
[65] Cabero M *et al* 2019 Blip glitches in Advanced LIGO data *Class. Quantum Grav.* **36** 155010
[66] Hastings W K 1970 Monte Carlo sampling methods using Markov chains and their applications *Biometrika* **57** 97
[67] Ising E 1925 Beitrag zur Theorie des Ferromagnetismus *Zeitschrift für Physik* **31** 253