Data Analysis Techniques for Continuous Gravitational Wave Searches

Joseph Bayley

Submitted in fulfilment of the requirements for the
Degree of Doctor of Philosophy

School of Physics and Astronomy
College of Science and Engineering
University of Glasgow

February 2017

Abstract

Contents

List of Tables

List of Figures

Acknowledgements

Declaration

Acronyms

A | B | C | E | F | G | L | M | N | P | R | S | U

A

ASD amplitude spectral density. vii, 114

B

BBH binary black hole. 1, 6, 7, 67

BNS binary neutron star. 1, 6, 8, 67, 68, 132

C

CBC compact binary coalescence. 5–8, 67, 73, 74

CFS Chandrasekhar, Friedman and Schutz. 9

CMB cosmic microwave background. 7

CNN convolutional neural network. v–vii, 26, 73–75, 80, 81, 84, 87–90, 93–100, 102–111, 132, 133

CPU central processing unit. v, 102–104

CW continuous gravitational wave. v–vii, 1, 5, 6, 8, 17, 23–27, 29, 55, 60, 63, 65, 67, 69–71, 73, 75, 76, 80, 84, 87, 90, 93, 96, 100, 102, 103, 105, 106, 108, 111–114, 116, 118, 131–134, 139, 140, 142

E

EM electromagnetic. 7, 74

EOS equation of state. 6, 8

F

FFT fast Fourier transform. vii, 46, 48, 68, 74, 75, 117, 118, 131, 132, 134, 136, 139

G

GPU graphics processing unit. v, 102–104

GR general relativity. 1

GRB gamma ray burst. 7

GW gravitational-wave. vi, 1, 5–11, 13, 14, 17–19, 24, 28–30, 33, 37, 51, 73, 74, 90, 97, 112–118, 126, 135

L

LIGO Laser Interferometer Gravitational-wave Observatory. vi, vii, 1, 11–15, 17, 24, 26, 28–30, 50, 55, 65, 67, 68, 72, 73, 87, 90, 93, 105, 114, 115, 117, 119, 123–126, 128, 130–133

LISA laser interferometer space antenna. 6, 8

M

MCMC Markov-Chain Monte Carlo. 22, 54

MDC mock data challenge. vii, 26, 48, 53–56, 58, 59, 62–65, 69, 70, 91, 95, 96, 100, 102

MSU million standard units. 26

N

NSBH neutron star black hole. 6

P

PEM physical environment monitor. 116

PSD power spectral density. 33, 46, 47, 49–51, 53, 74, 90, 97, 99, 114, 134, 135

R

RMS root median square. 52, 54–56, 70

S

SFT short Fourier transform. v, 24, 25, 29, 33, 37, 38, 40–42, 44, 46, 48, 50, 52, 55, 57, 67, 69–71, 93, 94, 99, 102–104, 118, 119, 123–127, 132

SNR signal-to-noise-ratio. 17, 24, 40–42, 44, 46, 48, 51, 52, 54–61, 65, 70, 71, 90, 93, 95–101, 105–107, 111, 132, 134–136, 140, 141

SSB solar system barycenter. 18

U

UCD up, centre or down. 32, 35, 36, 39, 40

Chapter 1

Introduction

Gravitational-waves (GWs) were first predicted in 1915 as a consequence of Einstein's general theory of relativity [1]. They are theorised as ripples in the fabric of space-time. The first observational evidence that GWs exist came from observations of the Hulse-Taylor binary [2, 3]. This observation of a pulsar in a binary system showed that the periastron was reached slightly early after each orbit, implying that the pulsars orbit was decreasing with time. If the separation of two orbiting objects is decreasing then the system must be losing energy. The loss in energy matched the general relativity (GR) prediction which assumed the energy was lost to GWs. This gave hope of GWs existence and helped lead the way to designing instruments which could directly detect them. The first direct detection of gravitational waves was made in 2015 when the two Laser Interferometer Gravitational-wave Observatory (LIGO) detectors in the US [4] identified a signal from a binary black hole (BBH) system. This was not only the first observation of a GW but gave information on a yet unobserved astrophysical system. This has since been followed by many more detections of BBH signals involving LIGO and Virgo including [5, 6]. In 2017 the LIGO detectors observed the first binary neutron star (BNS) system [7] which had a corresponding electromagnetic counterpart. This allowed verification of the source from optical counterparts and started the era of multi-messenger astronomy. These detections opened up the field of gravitational wave astronomy, where many more detections are expected to give more information on the universe and objects within it.

As well as searching for BBH and BNS signals, there are many efforts to detector other types of GW signals. This thesis focuses on efforts to search for a particular type of GW which are thought to originate from rapidly rotating neutron stars. In chapters 1 and 2 I will review introductory material. This includes a general introduction to the generation of GWs in Sec. 1.1 and their sources in Sec. 1.2. I will then introduce instruments used to detect GW in Sec. 1.3. In Chapter 2 I will introduce the general model for continuous gravitational waves (CWs) and current methods used to detect them. Chapters 3, 4 and 5 will go into detail about techniques developed by the author to search for CW signals.

Finally I will summarise this work and discuss future developments in chapter 6.

1.1 Gravitational waves

In general relativity, gravity is thought of as the curvature of space-time and matter moves according to this curvature. The matter in the universe also has an effect on the curvature of the space-time. The larger the mass of matter the more the space-time is distorted. Space-time can generally be described by Einstein's field equations

$$G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}. \tag{1.1}$$

where $G_{\mu\nu}$ is the Einstein tensor and $T_{\mu\nu}$ is the stress-energy tensor. The stress energy tensor describes the mass-energy in the universe, where its components contain information on the density of energy and momentum. The Einstein tensor contains information on the curvature of the universe. This can be derived directly from the metric tensor $g_{\mu\nu}$ which describes the geometry of the universe. Einstein's equations then explain how the curvature of space-time changes with the mass-energy within it. In empty space one can assume that the geometry of space-time is flat, i.e. there is no curvature to space-time. The metric tensor for this can then be defined as

$$g_{\mu\nu} = \eta_{\mu\nu} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{1.2}$$

Each index of this matrix refers to a space-time dimension, i.e. $x^0 = t$, $x^1 = x$, $x^2 = y$ and $x^3 = z$. Measuring a distance dx in space-time can be different for different observers, therefore, one needs a measure which is invariant for every observer. This is the space-time interval ds, also known as the line element, between two 'events' in space-time. This is defined as

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu. \tag{1.3}$$

As in Einstein's notation this is a sum over the indices μ and ν. Equation 1.3 can be thought to describe the space-time 'distance' between the two events. For flat space-time, $\eta_{\mu\nu}$, this can then be written as

$$ds^2 = -c^2 dt^2 + dx^2 + dy^2 + dz^2. \tag{1.4}$$

The Einstein equations Eq. 1.1 then demonstrate how the curvature of space-time $G_{\mu\nu}$ depends on the matter and energy distribution $T_{\mu\nu}$ within it.

A gravitational wave can be described as a ripple in this space time. The simplest way to visualise this is just a small time dependent change to the flat space-time metric $\eta_{\mu\nu}$. In linearised theory of gravity, the space-time metric $g_{\mu\nu}$ can be defined as

$$g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}, \tag{1.5}$$

where $\eta_{\mu\nu}$ is the metric for flat space-time and $h_{\mu\nu}$ is some perturbation, where $|h_{\mu\nu}| \ll 1$ [8]. In this linearised theory the perturbations to the metric tensor are assumed to be small, therefore, Einstein's field equations can be solved such that the solution is a plane wave. More information on this derivation can be found in [8, 9]. By using $g_{\mu\nu}$ from Eq. 1.5, we can write the linearised Einstein equations as

$$\Box h_{\mu\nu} = -16\pi T_{\mu\nu}, \tag{1.6}$$

where $\Box$ is the d'Alembert operator which in flat space is defined by

$$\Box = -\frac{1}{c^2}\frac{\partial^2}{\partial t^2} + \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \tag{1.7}$$

In empty space there is no matter, therefore, all the components of the stress energy tensor are zero, i.e. $T_{\mu\nu} = 0$. This allows Eq. 1.6 to be reduced to

$$\Box h_{\mu\nu} = 0, \tag{1.8}$$

Which is of course the wave equation. This follows the same form as in electrodynamics and the general plane-wave solutions have the form

$$h_{\mu\nu} = A_{\mu\nu} e^{ik_\alpha x^\alpha}, \tag{1.9}$$

where each component of $h_{\mu\nu}$ is a sinusoid traveling along vector k_α with amplitude $A_{\mu\nu}$ [10]. At this point the set of equations are not simple; the symmetric tensor $A_{\mu\nu}$ has 10 independent components. This can be greatly simplified by choosing a different gauge where the metric perturbation is both transverse and traceless (TT) [8]. This is just a choice of coordinate system which does not change any current assumptions **JOE: read more about gauges**. A traceless metric is one where the sum of the diagonal elements are 0 and a transverse metric is when the oscillations are perpendicular to the direction of travel. This gauge imposes two conditions: one is that $h_{\mu\nu}$ is traceless, i.e. that the sum of the diagonal elements are 0 and the other is that $h_{\mu\nu}$ is transverse. The transverse element means that the oscillations of the wave happen perpendicular to the direction of travel. At this point we can choose that the wave is traveling in the z direction which means that $k = (\omega, 0, 0, k)$. By then adopting the TT gauge there are only two unique

components to the metric such that the perturbation is

$$h_{\mu\nu} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & h_+ & h_\times & 0 \\ 0 & h_\times & -h_+ & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} e^{i(kt-wt)}. \tag{1.10}$$

The two unique components are then the two polarisations of gravitational waves, h_+ and $h_\times$. The affect of each of the polarisations on a ring of test particles can be seen in Fig. 1.1 where the gravitational wave is travelling out of the page.

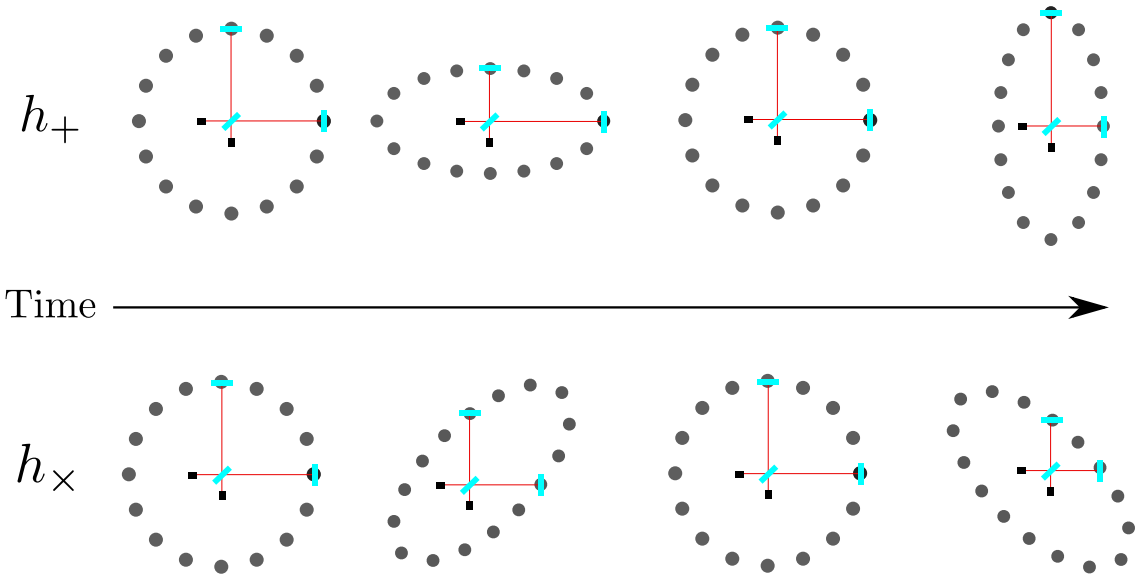


Figure 1.1: Shows how the plus and cross polarisations affect a ring of test particles. This assumes the wave is travelling out of the page and the effects have been greatly exaggerated. This also shows an example of how this effects the test masses of an interferometer. This will be described in more detail in Sec. 1.3.

Generating gravitational waves

To generate gravitational waves we go back to Eq. 1.6 where we include the stress-energy term on the right hand side. Following the derivation in [8], one can find that the gravitational wave amplitude is related to the second moment of the mass distribution. The second moment of the mass distribution $I_{\mu\nu}$is defined as

$$I_{\mu\nu}(t) = \int \rho(t, \mathbf{x}) x^\mu x^\nu d^3x, \tag{1.11}$$

where ρ is the mass density, and x_i and x_j are the coordinates [8]. This is the quadrupole moment tensor without the trace subtracted. The gravitational wave amplitude is then

$$h_{\mu\nu} = \frac{2}{r}\frac{d^2 I_{\mu\nu}(t-r)}{dt^2}, \tag{1.12}$$

where r is the distance from the source [9]. This has a slight modification in the TT gauge, see [8], however, has the same relationship between the mass quadrupole and the GW amplitude. This shows that for a GW to be generated, the second derivative of the mass quadrupole moment is needed. A mass quadrupole moment only exists when the mass distribution is not spherically symmetric. Therefore, a mass which is asymmetric and accelerating will produce a GW.

Systems which will produce detectable GWs are generally rapidly rotating high mass systems which have some asymmetry around their rotation axis. The sources of these GW will be described in the following section.

1.2 Sources and signals

There are many potential sources for GW. The expected sources can be split into 3 general categories based on their signal type: Transient, Stochastic and CWs. These categories are chosen based on the length of the signal and how well modelled the signal is. Figure 1.2 shows an example of each of the signals and their category. In the sections that follow, I will give an overview of the potential sources of each of these signal categories and their wave-forms.

1.2.1 Transient

Transient sources of GWs are short duration signals which are, depending on the source, observable from milliseconds to tens of seconds in current ground based detectors frequency band. Some of these sources, particularly CBC sources, will emit signals for a much longer time, however these are at a lower frequency and not observable by current ground based detectors detectors. Transient signals can be further split into two categories based on how well they are modelled. CBCs have well modelled wave-forms and bursts are generally from un-modelled or unknown sources.

Compact Binary Coalescence

CBCs originate from the in-spiral and merge of two compact objects which are gravitationally bound. The objects inspiral as they lost energy to the radiation of gravitational waves. Dependent on the masses and distances of the two objects, the gravitational waves

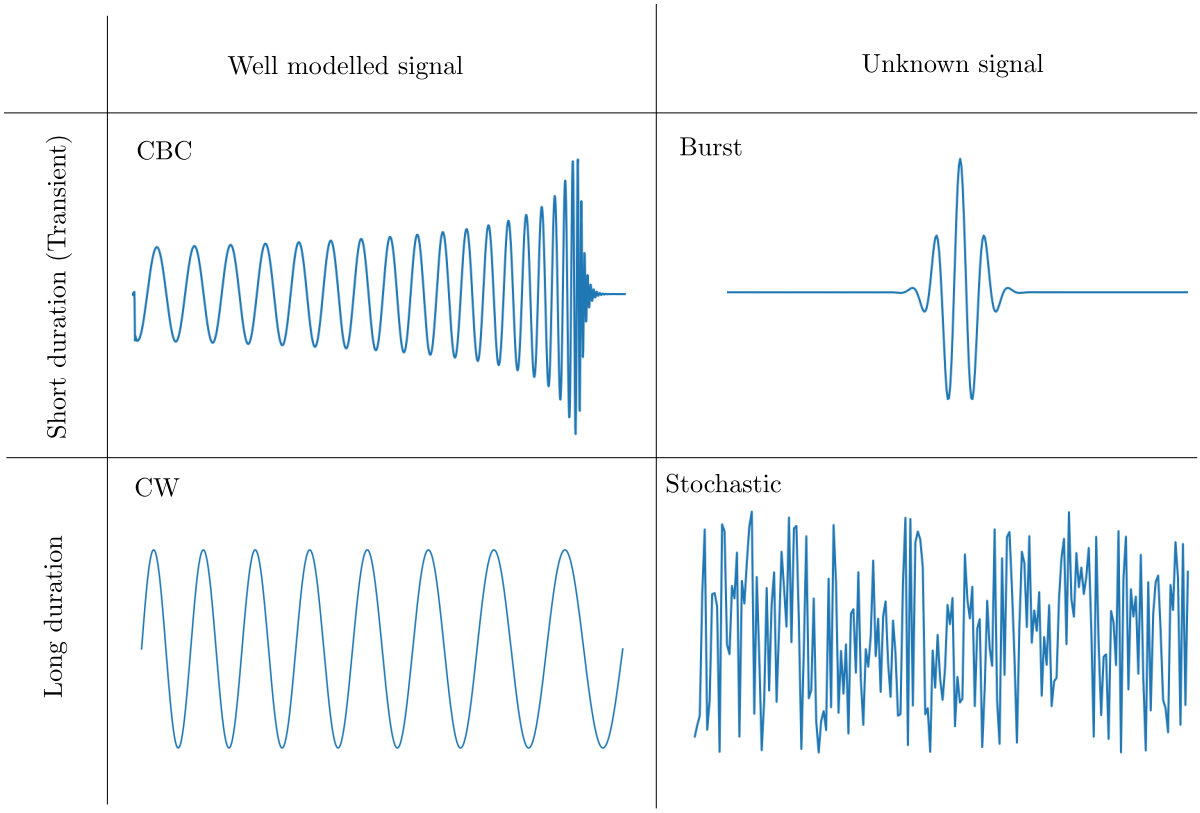


Figure 1.2: Each GW signal type can be categorised based on its signal length and how well the signal is modelled. Transient signals which are short duration in the ground based detectors band, include both well modelled compact binary coalescence (CBC) signals and unknown Burst signals. Long duration signals include well modelled CW signals and unknown Stochastic signals.

generated by the system can be detected by ground based detector such as LIGO [11] and Virgo [12]. In fact, the only detections to date have been of this type; these are summarised in [13].

JOE: More qualatative, See Grahams comment The compact objects referred to here are either either black holes or neutron stars. There are generally three types of CBC source: BBH, BNS and neutron star black hole (NSBH). The general structure of the waveform is the same for each of these and follows a 'chirp' where the GW frequency increases with time until merger. An example of this is shown in Fig. 1.2. For higher mass systems such as BBH these signals are detectable by ground based detectors for $< 1\,s$. For lower mass systems such as BNS they can be detected for longer periods $\mathcal{O}(10)$s. The lower frequency parts of the signal (i.e. earlier times) aim to be detected by future space based detectors such as laser interferometer space antenna (LISA) [14].

In systems which have a neutron star, during the in-spiral and merger phase, the neutron star can deform due tidal interactions between the objects [15]. This becomes useful as it will affect the generated waveform and can help place limits on and determine the equation of state (EOS) for the dense matter in a neutron star [16]. BNS systems also

offer a way do observe objects in multiple different channels, or what is known as multi-messenger astronomy. This is where the object can be viewed in the electromagnetic (EM) spectrum as well as in gravitational waves. This offers much in the field of astronomy as it can aid in the measurement of the Hubble constant [17]. Observations of BBH systems can also give information on how black holes and BBHs form, more details on this can be found in [18, 19].

Burst

Burst sources are also short duration however, are un-modelled or difficult to model, in the sense that the exact waveform of the signal is unknown. There are two possible reasons for the lack in knowledge of the waveform: the physics of the system is too complicated to model in a reasonable amount of time or there is no model of the source. As there is no model to generate waveforms, burst searches cannot use matched filtering as in CBC searches. Rather burst searches look for short bursts in power which is coincident between multiple detectors [20, 21]. There are a number of systems which could potentially emit a short duration burst signals. These include core collapse supernovae [22], gamma ray bursts (GRBs) [23], cosmic strings [24] and other unknown sources. Detecting GW from one of these sources could offer more insight into the processes which cause core collapse supernovae.

As burst searches are un-modelled, they are sensitive to almost any signal which is coherent between detectors. This allows them to also search for signals from CBC as well as any GW signal from an unknown source.

1.2.2 Stochastic

The stochastic background has no signal model, however, is expected to be a persistent source of GW in the background of the detector. The stochastic background is the incoherent sum of many unresolved GW signals. The source of these signals can be anything from cosmological sources such as cosmic strings to CBC signals. These signals can be thought of as the GW analogue of the cosmic microwave background (CMB). The signal is assumed to be isotropic such that it can be observed at any point on the sky [25]. As the stochastic background is noise-like it is very difficult to distinguish from noise within a single detector [25]. Therefore, searches for the stochastic background correlate signals between multiple detectors [26, 25]. When detected, these signals may be able to offer insights into the early universe and its formation.

1.2.3 Continuous waves

CWs are long duration signals which can be well modelled. The signals last for times greater than the observation runs of ground based detectors and in general have a fixed or slowly varying frequency. There are a number or potential sources of CWs including a CBC signal earlier in the inspiral phase. Before the final stages of the inspiral of a CBC signal, the objects orbiting at much slower non-relativistic speeds. They therefore emit a GW with a slowly varying frequency and can remain in this phase for millions of years. This signal however, is at lower frequency than ground based detectors can detect, therefore space based detectors such as LISA [14] are expected to observe this type of CW.

The primary source for many CWs searches is rapidly rotating neutron stars with spin periods ranging from $\sim 10^{-3} - 10$ s [27]. Neutron stars originate when a massive star $\sim 11 - 20 M_{\odot}$ collapses and are the remnant of this collapse, they are objects with incredibly high density and are highly magnetised with field strengths of $10^{8} - 10^{15}$ G [28]. They have masses around $1.4 - 2\ M_{\odot}$ contained in a star with radius of ~ 10 km. Despite many observations in the electromagnetic spectrum and a large amount of research, these objects are not well understood. A key part or neutron stars which is not understood is the EOS. A review of the current understanding can be found in [29]. The EOS relates quantities such as the pressure and density of a neutron star and dictates how the neutron star matter behaves. Observations of GWs from neutron stars can place limits on the EOS of this type of matter. These observations have already been made in the form of BNS mergers [7]. However, independent observations of rapidly rotating neutron stars can add to this understanding by placing limits on the deformability of the star and therefore the EOS.

For a neutron star to emit a gravitational wave it needs to have some asymmetry in its mass distribution around its rotation axis, this follows from Eq. 1.12. There are a number of different mechanisms which could cause this and emit GWs, some of these are reviewed in [30, 31, 32, 33]. Here I will summarise two main theories: Neutron star mountains and neutron star oscillations.

Mountains

One of more likely mechanisms for detectable GW emission from neutron stars is from 'mountains' on the surface of the star. These are permanent deformations of the crust which are non axisymmetric, i.e. the deformation is not symmetric around the rotation axis.

This deformation or asymmetry can be quantified by the ellipticity ϵ of the neutron

star. This is defined using the principal moments of inertial

$$\epsilon = \frac{I_{xx} - I_{yy}}{I_{zz}}, \tag{1.13}$$

where I_{zz}, I_{xx}, I_{yy} are the principal moment of inertia. This is when the star is rotating around the z axis so I_{zz} is along the rotation axis.

There are a number of theories which describe the origin of this axisymmetry. If the pulsar is in a binary system and accreting material from its companion star, the material can be funneled towards the magnetic poles by the magnetic field, thereby causing a hot spot [32]. This 'hot spot' could cause a deformation on the surface of the star which is not axisymmetric. The magnetic stresses from strong magnetic fields within the star, could potentially also cause non axisymmetric deformations to the star []. Finally the spin down of the pulsar itself could cause stresses in the crust of the star until the point of breaking, its then after this break which could leave a distortion in the crust [34]. More details on the signal waveform of this type of GW and methods to search for it will be explained in Sec. 2.

Neutron star oscillations

JOE: Probably shorten this section, as may not know enough to answer questions on There are a number of oscillation modes within a star such as f-modes, p-modes and r-modes [34]. Each of these waves are oscillations in the star similar to oscillations in the earth which measured in terrestrial seismology. The difference between these modes is the restoring force bringing the perturbed state back to equilibrium. For example, gravity is the restoring force for f-modes where the oscillations happen in the crust of the star. The more promising of these for gravitational wave emission and detection is the r-mode [33]. These are oscillations in the neutron superfluid part of the star, where the restoring force is the Coriolis effect from the rotation of the star. Figure 1.3 shows an highly exaggerated view of a neutron star with an oscillation mode travelling in each direction. If these modes are excited in a non-rotating star, then they will emit GW where the GW carries away angular momentum [35]. For the mode travelling in a clockwise direction, this angular momentum is positive and the mode travelling anti-clockwise the angular momentum is negative. Therefore, the GW is taking away either positive or negative angular momentum (adding angular momentum) depending on the direction of rotation. The emission of GW damps the modes and the magnitude of the perturbation decreases making them extremely difficult to detect. Now if the neutron star is rotating, this can lead to an effect called the Chandrasekhar, Friedman and Schutz (CFS) instability [36, 37]. As the rotation speed of the neutron star increases, there are two different effects on the modes travelling in opposite directions. For the mode travelling anti-clockwise with the stars rotation, the

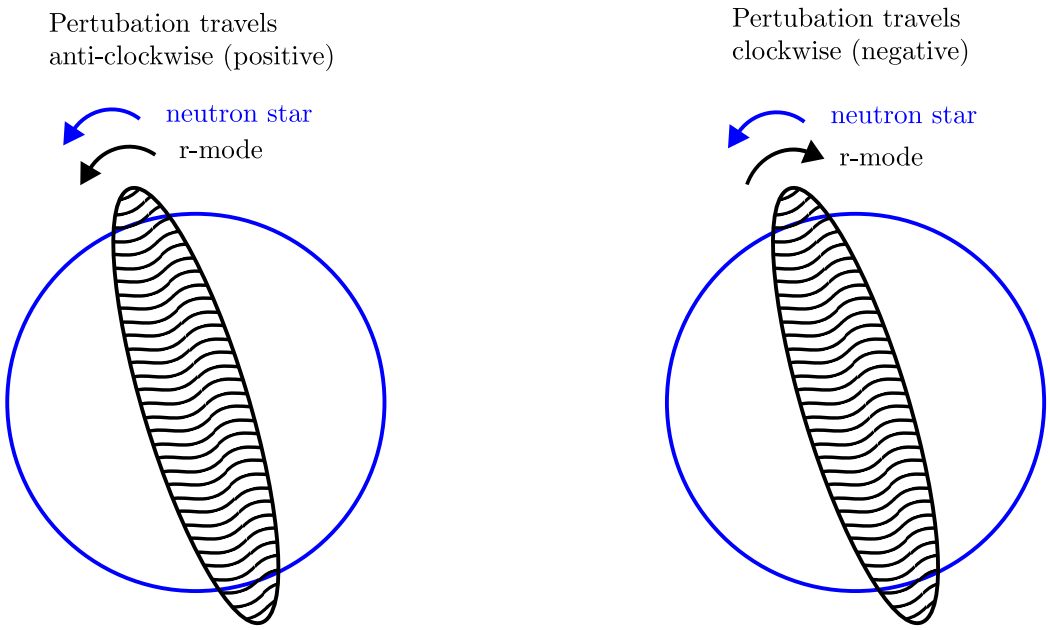


Figure 1.3: The r-modes can travel in either direction in the star. in the case when the r-mode is moving clockwise and the neutron star is moving anti-clockwise, if the neutron star is rotating fast enough it can cause unstable emission of GW. This image was reconstructed from [35].

mode will appear to be travelling faster, therefore, will emit more GW taking away more angular momentum. This means that this mode will be damped more rapidly. The interesting affect is for the mode travelling clockwise, opposite to the neutron stars rotation. At a certain rotation rate, the mode will be 'frozen' from the observers perspective and no GW will be emitted. As the rotation rate increases further, the mode will appear to travel anti-clockwise to an observer, i.e. the mode is dragged in the opposite direction by the stars rotation. Here it is key to remember that this mode had negative angular momentum as in the neutron stars frame it is still travelling clockwise. As the mode rotates its emits positive angular momentum, which is then subtracted from the modes negative angular momentum. The magnitude of the angular momentum then increases such that more GWs are released. This effect causes the amplitude of the oscillation to grow and therefore become unstable. Therefore, a neutron star is unstable to GW emission if it is rotating sufficiently fast [33]. For a more detailed view on how r-modes generate GW see [38, 35]

1.3 Detectors

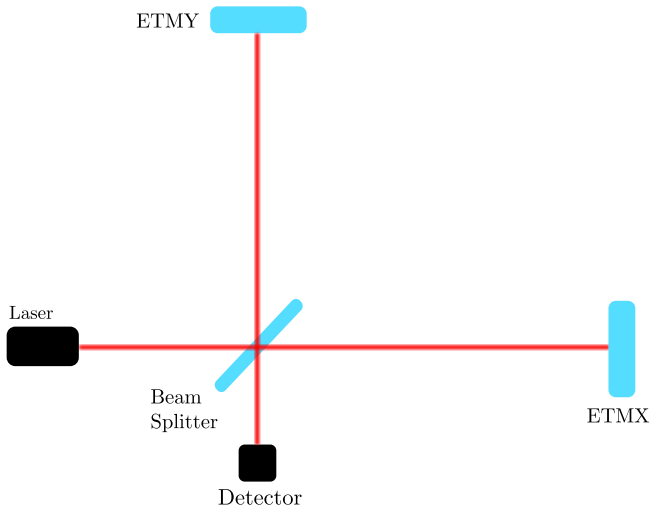
The indirect detection of gravitational waves from the Hulse-Taylor binary pulsar system left little doubt that GW existed. The real challenge was to design an instrument which could directly detect gravitational waves. There were a number of different proposed methods for the design of the instrument, notably: resonant bar detectors, both ground based and space based interferometers, pulsar timing arrays and cosmic microwave background (CMB) detectors. The first resonant bar detector was designed and built by Joseph Weber [39]. These are large cylinders of metal which should resonate as a gravitational wave passes by. There are a few different designs of this type of detector, including an omni-directional design [40]. Pulsar timing arrays aim to use the accurate arrival time of pulses from millisecond pulsars to measure GW [41]. As a GW passes between the pulsar and the observer, the arrival time of the pulses should change by $\mathcal{O}(10)$ ns [41]. Whilst a detection has not been made using pulsar timing arrays, these methods are still in use. Cosmic microwave background detectors aimed to look for evidence of gravitational waves in the polarisation's of the CMB [42]. These use the a range of detectors to look at the CMB however, are yet to confirm a detection of a GW signal. The most commonly known design of a GW detector is the ground based interferometer, these made the first detection of GW in 2015 [4]. These are the focus of this section as the analysis that will follow uses data from the LIGO detectors in the USA [43, 11] and Virgo detector in Italy [12, 44].

1.3.1 Laser Interferometers

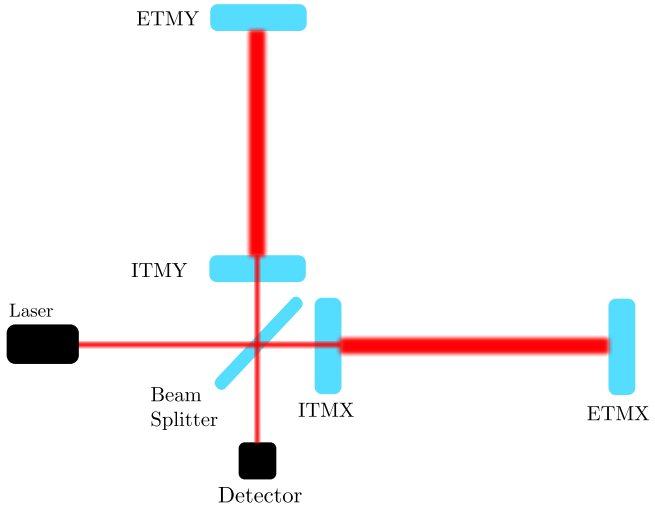
Laser interferometers use the interference of light to measure a length with high precision. The majority of this section will focus on ground based interferometers such as LIGO and Virgo [11, 12]. A simple design of an interferometer is shown in Fig. 1.4a. A laser beam is fired at a beam splitter which splits the light equally down two perpendicular arms. Each of these beams is reflected from a mirror at the end of either arm. The light then returns to the beam splitter where the two beams are combined and sent to a photo-detector. At the output, there is an interference pattern between the two beams. If the length of one of the arms is changed then the interference pattern will change as the phase of one beam changes with respect to the other. The phase difference of the light can be related to the wavelength of the light, λ_l and the length of the detectors arms L by

$$\Delta\phi \sim \frac{\Delta L}{\lambda_l}, \tag{1.14}$$

where $\Delta\phi$ is the phase change and ΔL is the difference in the arm lengths. An interferometer can then measure small changes in the mirrors position. This can be used in gravitational wave detection as the mirrors at the end of each arm of the interferometer



(a) Simple interferometer.



(b) Fabry-Perot interferometer.

Figure 1.4: LIGO uses interferometry to make a detection. This has many additional features, one of which is shown above. ETMY and ETMX refer to the end test masses, which are just mirrors at the end of the interferometer arms. The beam splitter splits the Laser beam equally to each arm, this them recombines the beams back to the detector. ITMY and ITMX refer to the internal test masses, these create a Fabry-Perot cavity in the interferometers arms which can build up laser power.

can be treated as 'free' test masses. Figure 1.1 shows the effect of a GW on free test masses. One can see this by looking at the proper separation between two test masses. If we place two test masses along the x axis with separation L_c where a gravitational wave is travelling along the z axis, the proper distance between them is given by

$$L = \int_0^{L_c} \sqrt{g_{xx}} dx, \tag{1.15}$$

where $g_{xx} = 1 + h_+(t)$ is the combination of Eq. 1.5 and Eq. 1.10. As the metric perturbation is small is can be can be expanded to first order, i.e. $\sqrt{g_{xx}} = \sqrt{1 + h_+(t)} \approx 1 + \frac{1}{2}h_+(t)$. The proper distance is then

$$\begin{aligned} L &\approx \int_0^{L_c} 1 + \frac{1}{2}h_+(t) dx \\ &\approx L_c + L_c \frac{1}{2} h_+(t). \end{aligned} \tag{1.16}$$

From Eq. 1.16 on can see that in this configuration the plus polarisation of the gravitational wave causes the separation between the two test masses to oscillate [8]. This can then be oscillation can be expressed as a fractional length change

$$\frac{\delta L}{L} \approx \frac{1}{2} h_+. \tag{1.17}$$

In the interferometer, this affect changes the lengths of the two arms which therefore changes ΔL in Eq. 1.14. The change of the interference pattern with time is then related to the GW. In practice the position of the two mirrors are held in position such that the interference pattern does not change. The GW is then related to the readout of the mirrors control system.

If one looks at Eq. 1.17 then if the mirrors at the end of the arms (ETMX and ETMY) are are placed further from the beam splitter, i.e. L is increased, then the length change of the arms due to the gravitational wave will be greater. This means that increasing the length of the detectors arms increases the sensitivity of the interferometer. A method to achieve a similar affect without physically increasing the arm length is to use a Fabry-Perot cavity [11], this is shown in Fig. 1.4b. This is where a semi-transparent mirror is placed between the beam splitter and end mirror in each arm (ITMX and ITMY). Light enters this cavity and reflects back and forth between the two mirrors (ITMX and ETMX) a number of times before returning to the beam splitter. This increases the time the light spends in one arm and is equivalent to increasing the arm length. Actual ground based GW detectors such as LIGO [43] and Virgo [12] are much more complicated than described above. They use many techniques to increase the sensitivity some of which are outlined in [11, 43]. Many of these techniques are designed to reduce non-astrophysical affects on

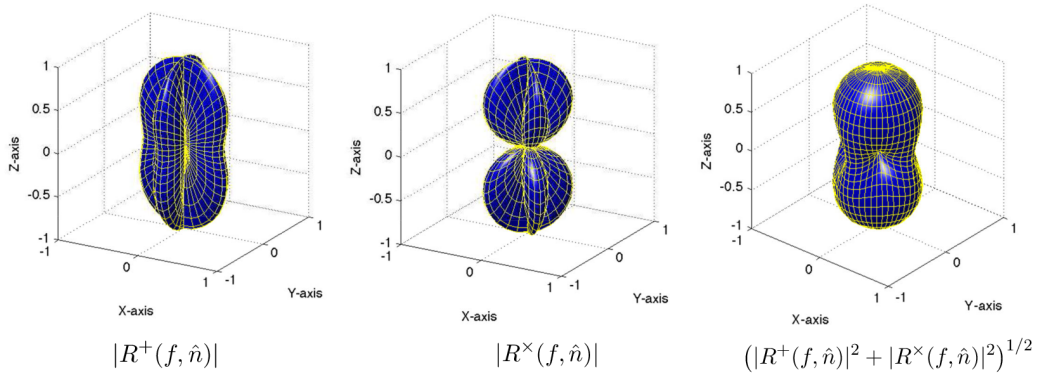


Figure 1.5: The antenna response is shown as in [26] for both the plus and cross polarisations and their average. The detectors arms lie on the x and y axis in the above plots.

the detector, some of these effects and solutions are listed in Sec. 1.3.1.

Detector response

An important factor to know when using detector data to search for astrophysical signals is the detectors antenna pattern. This measures how sensitive the detector is to different directions. An example of the antenna response for LIGO is in Fig. 1.5 where the detectors arms lie on the x and y axis of the image. This is clear when thinking about how a gravitational wave affects the test masses as in Fig. 1.1. As the GW is transverse to its propagation, when the detector is face on to the source, there will be a maximum change in the arm lengths and therefore a maximum sensitivity. In the same way the sensitivity will be at a minimum when edge to the source.

Noise sources

To increase the sensitivity of the LIGO detectors, any effect on the output of the interferometer which is not astrophysical ideally needs to be reduced. This involves understanding what causes certain noise features in the detector, and how the affect of these can be limited. Within the detector, there are many sources of noise. Some of these noise sources and how they limit the detectors strain sensitivity are all shown in Fig. 1.6 from [11]. Here I will summarise some of the limiting sources and also sources which become useful for

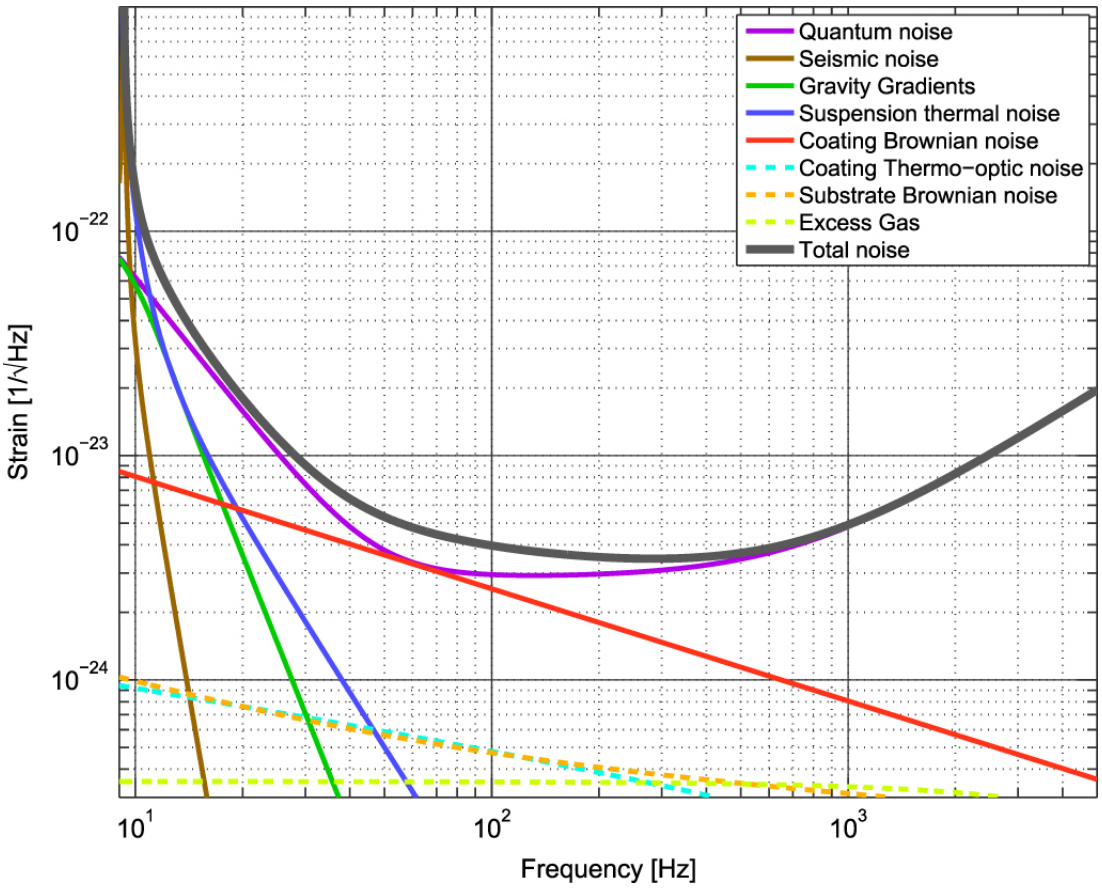


Figure 1.6: The different noise sources affect the sensitivity of the LIGO detectors at different frequencies. This shows the various sources how the affect the noise curve [11].

understanding later sections.

Seismic noise This originates from vibrations in the earth which can be from effects such as the earths seismic activity or anthropogenic sources. This affects lower lower frequencies of LIGOs spectrum. This can be oscillations such as earthquakes or ocean waves. Seismic waves cause the mirrors to oscillate and induce a change in the length of the arm. This is reduced by having multi stage suspensions in the detectors which filter out the seismic oscillations [45].

Coating noise This is in general due to two main factors, the thermal noise of the coating and Brownian noise. The Brownian noise is from the mechanical dissipation in the coating and the thermal noise is due to thermal dissipation. The Brownian noise is the dominant factor as shown in Fig. 1.6. These effects are limited by using different coatings on the mirrors [46].

Quantum noise Quantum noise is a fundamental limit due to the statistical uncertainty of counting photons. This limits the sensitivity at many frequencies.There are methods to reduce this include squeezing of the light [47].

Electronics noise **JOE: technical noise maybe** Whilst this is not shown in Fig. 1.6, this becomes important to searches described later. These have a different effect on the detector which is more narrowband frequency lines. This is generated by the digital and analogue electronics that are used to measure the signal.

There are also many other sources of noise in the detector which I have not listed. However, these are often not the limiting cases of noise or are not relevant to this thesis. In Sec. 5 I will go into more detail about specific noise sources in the detector known as instrumental lines and how they can be monitored and potentially removed.

Chapter 2

Searching for continuous gravitational waves

Continuous gravitational waves have particular challenges when it comes to their detection. The nature of CW are that they are long duration, this means that they would be observed for the entirety of a detectors observing runs. The signals also have an intrinsically small amplitude which is below the noise floor of current ground based detectors such as LIGO. This means the for a detection, the entire observing runs data will be needed to accumulate enough signal-to-noise-ratio (SNR) for a signal to be observed. Given that LIGO samples as ~ 16 kHz (generally downsampled to ~ 4 kHz) this leaves a huge amount of data which needs to be searched through. As will be described in Sec. 2.3.1 and 2.3.2, this requires a large amount of computational resources to perform these searches. For some types of search, the parameter space can also be very large, this only adds to the computational time and in some cases makes it infeasible.

Whilst I have described the potential sources of the signal and its approximate signal type in Sec. 1.2.3,to perform a search the wave-form of a signal and how it is observed is needed. In this section I will go into more detail on the 'mountain' model in Sec. 1.2.3 and its wave-form description. This model is then used in various search methods for CW signals. In Sec. 2.3 I will overview a subset of current searches for CW signals. Sec. 2.4 explains the motivation for the majority of the work in this thesis.

2.1 Continuous signal model

The model of a GW signal from a pulsar is relatively simple, it is a quasi-sinusoidal signal. This means that the signal is a sinusoid with a slowly varying frequency. One reason for the slow variance in the frequency is due to the energy loss to GW as the pulsar spins down. Here the signal is modelled to originate from an isolated triaxial neutron star rotating around a principal axis. The parameters of each pulsar can be split into two sections: the

Doppler components $(\alpha, \delta, \boldsymbol{f})$ and its amplitude components $(\psi, \phi_0, \iota, h_0, \theta)$. This ignores any orbital parameters which would be present if the star was in a binary systems and higher order frequency derivatives. They are defined as follows: the sky positions α and *delta* refer to the right ascension and declination. $\boldsymbol{f}$ refers to the source frequency and its derivatives. ψ and ϕ_0 and h_0 are the GW polarisation, initial phase and amplitude respectively. ι is the inclination angle which is how much the source is tilted relative to the observer. θ is the 'wobble angle' or the angle between the rotation axis and the symmetry axis of the neutron star.

The definition of the GW from a neutron star here follows that in [31, 48, 49]. The amplitude of the GW can be defined as

$$h(t) = F_+(t)h_+(t) + F_\times(t)h_\times(t), \tag{2.1}$$

where $h_+, h_\times$ are the plus and cross polarisations functions as in Eq.1.10 and $F_+, F_\times$ are the antenna pattern functions to the two polarisations. These are defined by

$$\begin{aligned} h_+(t) &= h_0 \frac{1 + \cos^2(\iota)}{2} \cos(\Phi(t)) \\ h_\times(t) &= h_0 \cos(\iota) \sin(\Phi(t)) \end{aligned} \tag{2.2}$$

The plus and cross polarised components then depend on the GW amplitude h_0, the inclination angle of the source ι and the phase evolution of the GW. Here I have chosen to assume a small wobble angle θ, however, this is included in [48]. The phase of the wave $\Phi(t_{\rm SSB})$ at the solar system barycenter (SSB) can be defined as

$$\Phi(t_{\rm SSB}) = \phi_0 + 2\pi \left[f(t_{\rm SSB} - t_0) + \frac{1}{2}\dot{f}(t_{\rm SSB} - t_0)^2 + ... \right]. \tag{2.3}$$

This consists of an initial phase ϕ_0 which is the phase at time t_0, the frequency of the signal f and its derivative $\dot{f}$ at time t_0. Here we show the phase to second order, however, this can be easily extended if necessary. The time at the SSB $t_{\rm SSB}$ can be transformed to the time t at the detector by

$$t_{\rm SSB} = t - \frac{\mathbf{r}_d \cdot \mathbf{k}}{c} + \delta_t. \tag{2.4}$$

Here $\mathbf{r}_d$ is the position of the detector with reference to the SSB, $\mathbf{k}$ is a unit vector in the direction of the source. This essentially takes into account the Doppler shift of the signal due to the movement of the detector, i.e. as the earth rotates and orbits the sun. c is the speed of light and δt is extra corrections from the Einstein, Binary and Shapiro delay [].

The amplitudes h_0 in Eq. 2.2 are defined by

$$h_0 = \frac{16\pi^2 G}{c^4}\frac{\epsilon I f^2}{r}, \tag{2.5}$$

where G is the gravitational constant, c is the speed of light, ϵ is the ellipticity of the star, f is the sum of the frequency of rotation of the star and the frequency of precession, r is the distance to the star and I_{zz} is the moment of inertia with respect to the rotation axis z. The ellipticity of the star ϵ is a measure of the distortion of the star around its rotation axis and is defined by

$$\epsilon = \frac{I_{xx} - I_{yy}}{I_{zz}}, \tag{2.6}$$

where I_{xx}, I_{yy} and I_{zz} are the moments of inertia for each axis.

In Eq. 2.1, $F_+(t)$ and $F_\times(t)$ are the antenna pattern functions of the detector. These describe how sensitive a detector is to a particular location on the sky at any given time. The amplitude of the signal will vary dependent on the orientation and location of the detector relative to the source. This is described in Sec. 1.3 and the response to sky location is shown in Fig. 1.5. These components are defined in [48] as

$$\begin{aligned} F_+(t) &= \sin\zeta[a(t)\cos(2\psi) + b(t)\sin(2\psi)], \\ F_\times(t) &= \sin\zeta[b(t)\cos(2\psi) - a(t)\sin(2\psi)], \end{aligned} \tag{2.7}$$

where ζ is the angle between the arms of the detectors, ψ is the polarisation angle of the GW and $a(t)$ and $b(t)$ are defined in [48] and relate the sky location to the orientation of the detector at a given time. A full derivation of this can be found in [48] where each of these terms are expanded.

Eq. 2.1 - 2.7 then describe the amplitude and phase evolution of a signal at a given detector location and orientation.

2.2 Bayes Theorem

A key part in understanding the different methods to search for GW or any data analysis, is understanding probability and statistics. This gives understanding of the random processes underlying all measured quantities. Whilst there are generally two approaches to statistics: Frequentist and Bayesian, here I will focus on the Bayesian approach.

2.2.1 Basic probability

Initially I will define some basic concepts of probability. We can define the probability of some event A as $p(A)$ where probabilities follow $0 \leq p(A) \leq 1$ and some other event B which has a probability $p(B)$ and follows $0 \leq p(B) \leq 1$.

Union A union is the probability of either and event A happening or event B happening. This is written as, $p(A \cup B)$.

Intersection An intersection is then the probability that both and event A and an event B happens. This is written as $p(A \cap B)$.

Independent and dependent Events If the events A and B are independent, i.e. the event A does not affect the outcome of event B, then

$$p(A \cap B) = p(A)p(B). \tag{2.8}$$

However, if the event A is dependent on event B, i.e. the event A affects event B or vice versa, then the joint probability of both events is

$$p(A \cap B) = p(A)p(B \mid A) = p(B)p(A \mid B). \tag{2.9}$$

Here $p(B \mid A)$ means the probability of event B happening given that event A has happened.

Conditional probability Conditional probability arises from situations where the outcome of one event will affect the outcome of future events. The definition of this arises from the the dependent events defined above in Eq. 2.9

$$p(A \mid B) = \frac{p(A \cap B)}{p(B)}. \tag{2.10}$$

Bayes Theorem Bayes theorem can then be defined using conditional probabilities. i.e we can use

$$p(A \mid B) = \frac{p(A \cap B)}{p(B)} \quad \text{and} \quad \mathrm{p(B \mid A)} = \frac{\mathrm{p(A \cap B)}}{\mathrm{p(A)}} \tag{2.11}$$

such that

$$p(B)p(A \mid B) = p(A)p(B \mid A) \tag{2.12}$$

and this is rearranged to Bayes theorem

$$p(A \mid B) = \frac{p(A)p(B \mid A)}{p(B)} \tag{2.13}$$

2.2.2 Bayesian Inference

We can take Bayes theorem from Sec. 2.2.1 and apply it to a problem which involves inferring some parameters from some model. Here we can relabel the events A and B with

the data $\boldsymbol{d}$ and the parameters $\boldsymbol{\theta}$ of some model I. Equation 2.2.2 then becomes

$$p(\boldsymbol{\theta} \mid \boldsymbol{d}, I) = \frac{p(\boldsymbol{\theta}, I)p(\boldsymbol{d} \mid \boldsymbol{\theta}, I)}{p(\boldsymbol{d} \mid I)} \tag{2.14}$$

where each of the components are assigned names: $p(\boldsymbol{\theta} \mid \boldsymbol{d})$ is the posterior distribution, $p(\boldsymbol{\theta})$ is the prior distribution, $p(\boldsymbol{d} \mid \boldsymbol{\theta})$ is the likelihood and $p(\boldsymbol{d})$ is the evidence.

Posterior The posterior distribution describes the probability of a parameter θ in some model I given some data d. For many problems this is the distribution which is most useful as it informs you the most likely set of parameters of you model given some observation.

Prior The Prior distribution is a key part of Bayesian statistics. This distribution describes any information which you have prior to the observation. This is a distribution defined by the user, where you define a distribution of the parameters based on what you expect to be true.

Likelihood The likelihood is where the observation is included in the calculation. This tells you how probable it is to get the observed data d given the model I with the set of parameters θ.

Evidence The evidence is the probability of the data itself given the choice of model. This is found by integrating the likelihood over all possible values of $\boldsymbol{\theta}$ weighting them by our prior belief of that value of $\boldsymbol{\theta}$. This known as a marginal distribution and is defined by,

$$p(\boldsymbol{d} \mid I) = \int p(\boldsymbol{\theta}, I)p(\boldsymbol{d} \mid \boldsymbol{\theta}, I)d\boldsymbol{\theta}. \tag{2.15}$$

Bayes theorem then gives a description of the probability distribution of some parameters in a model given some observation. Often when using Bayesian statistics the aim is to find posterior distribution of parameters. There are very few cases where this can be calculated analytically, therefore, numerical methods are almost always used to find the posterior. This can be difficult to calculate numerically especially in problems where the parameters space has many dimensions. The most difficult part to calculate is the evidence in Eq. 2.15, this involves calculating an integral over all possible parameters. There is however, a way around having to calculate this. For any given mode I, the evidence $p(\boldsymbol{d} \mid I)$ is the same for any set of parameters $\boldsymbol{\theta}$ in Eq. 2.14. The evidence is then just a normalisation factor for the posterior distribution. When different models are not being compared, and we assume the model I to be true, we no longer need to calculate the evidence. The posterior distribution can then be found by sampling

$$p(\boldsymbol{\theta} \mid \boldsymbol{d}, I) \propto p(\boldsymbol{\theta}, I)p(\boldsymbol{d} \mid \boldsymbol{\theta}, I). \tag{2.16}$$

To find this posterior you could then calculate (sample) the value for every point in parameter space. This however, is very computationally expensive and often the posterior distribution is located in a small area in parameter space. Therefore, the majority of the time is sampling a area of parameter space where the posterior is close to zero, and this is not particularly useful. A method titled Markov-Chain Monte Carlo (MCMC) was proposed [50] to deal with this issue, more information on this can be found in [51, 52]. This builds up the posterior distribution by randomly jumping around in the parameter space. It starts by calculating the posterior value for a particular point in parameter space. Then it will randomly jump to another parameter space point. The posterior can then be calculated again, if the posterior value is higher then the jump is 'accepted'. This just means that the parameter values of this point are stored. If the posterior value is lower than the previous step then the jump is accepted with some probability. This means that there is a random chance that a value lower than the current is accepted. As the accepted positions aim for areas where the posterior is higher, MCMC does not waste time calculating areas in parameter space of low posterior values. The samples which were accepted then build the posterior distribution.

In certain situations it can be useful to calculate the evidence in Eq.2.15. For example, if there are two different models which could represent the data, the evidence can be used to determine which of the two models is more likely. This is known as a Bayes factor where two models I_1 and I_2 are compares and is defined as

$$B = \frac{p(\boldsymbol{d} \mid I_1)}{p(\boldsymbol{d} \mid I_2)} \tag{2.17}$$

This then requires the calculation of the evidence. To estimate the evidence efficiently a method known as Nested sampling can be used, this is explained in detail in [53, 54]. By calculating the Bayes factor, which is similar to a likelihood ratio, one can find the posterior odds of a particular model by using,

$$\frac{p(I_1 \mid \mathbf{d})}{p(I_2 \mid \mathbf{d})} = \frac{p(I_1)}{p(I_2)} \frac{p(\mathbf{d} \mid I_1)}{p(\mathbf{d} \mid I_2)}. \tag{2.18}$$

The can be written as *posterior odds* = *prior odds* $\times$ *Bayes factor*. This is then a comparison of how likely different models are given some observation.

The methods described above then provide a way to estimate parameters of a model given some data. Also this provides a way to compare different models given some observation. In following sections the methods described above are used to estimate various parameters.

2.3 Continuous wave searches

There are many different methods to search for continuous gravitational waves. They can be split into three general categories: Targeted searches, Directed searches and All-sky searches. The main difference between these different categories is the amount which is known about the source prior to the search. For targeted searches the sky position (α, δ) and rotation frequency are known from electromagnetic observations, i.e. X-ray, radio or γ-ray. Directed searches have information on the sky position (α, δ) but not the rotation frequency. For all-sky searches, there is no prior knowledge of the pulsar, therefore, is a search for unknown pulsars. In general the searches in each of these categories use two distinct techniques: Fully coherent searches and Semi-coherent searches

2.3.1 Fully coherent

A Fully coherent search generally uses a pre generated waveform which follows the model described in Sec. 2.1. This contains all the phase information of the signal. The set of parameters which generated the waveform which 'matches' the best can then be considered as the optimum set of parameters given the data. This is known as a matched filter [] and is used in CW searches in [49].

The matched filter maximises the signal to noise ratio for a given filter, in this case the filter is our CW model. The matched filter used for CW models is defined in [55] and it titled the $\mathcal{F}$-statistic. This maximises a likelihood with respect to the parameters. If one assumes that the noise n is Gaussian and zero mean, the data x can be written as

$$x(t) = n(t) + h(t). \tag{2.19}$$

The likelihood can then be written as

$$\log \Lambda = (x \mid h) - \frac{1}{2}(h \mid h) \tag{2.20}$$

where the product $(x \mid y)$ is defined as

$$(x \mid y) = 4\mathcal{R} \int_{-\infty}^{\infty} \frac{\tilde{x}^{\mathrm{X}}(f)\tilde{y}^{\mathrm{X}*}(f)}{S^{X}(f)} \mathrm{d}f. \tag{2.21}$$

This is fully expanded into the $\mathcal{F}$-statistic in [48], however, it is this likelihood function which is maximised.

Targeted searches look for a specific pulsar which has been observed in the electromagnetic spectrum. These observations give information such as the sky position and the frequency evolution of the source. Using knowledge of the earths position around the sun, which is well known, one can use the accurate sky position and frequency of a known

source to find its phase evolution in Eq. 2.3. This mean that for this type of search one can maximise the likelihood with respect to the parameters h_0, ϕ_0, ι and ψ. Another method which uses templates is described in [49], this uses a Bayesian approach.

This type of search can take long periods of time. This is due both to the size of the parameter space and the amount of data which needs to be searched. CW searches need long observation times in order to accumulate the required SNR for detection. Therefore, most searches use data from an entire LIGO observing run which can last for $\mathcal{O}(1)$ years. Given that the sampling rate for the GW channel is 16 kHz usually downsampled to ~ 4 kHz, the quantity of data is large.

Whilst the fully coherent matched filter searches have methods to reduce the computational time for known sources, in all-sky and directed searches, this type of search is no feasible. This is because all-sky and directed searches have a wider parameter space, therefore, enough templates need to be made to sufficiently cover the large parameter space. This task quickly becomes impossible for coherent matched filtering for an entire observing run due to the amount of time needed. This problem led to the development of semi-coherent searches which will be introduced in the next section.

2.3.2 Semi coherent

Semi-coherent searches offered a solution to searching over large parameters spaces and large amounts of data. As is directed and all-sky searches the phase evolution of the source is not known, one cannot use a coherent search for the entire observing run. It may however, be possible to approximately describe the phase for a shorter length of time known as the coherence time, T_{coh}. The general idea of a semi-coherent search is to break the data-set into smaller section which each can be analysed coherently. The coherent analysis can use the matched filter as described in Sec. 2.3.1 or another method such as a Fourier transform. The results from each of these individual sections can the be combined incoherently using various methods which will be summarised later. This method can greatly reduce the time taken for the analysis depending on the coherence length, however, will always come with some loss in sensitivity.

There are many different types of semi-coherent search which use various methods to incoherently combine the coherently analysed results. I will summarise some of these searches below, some of these searches were summarised and compared in [56]. Many of these searches use a set of 1800s long Fourier transforms as the input data, known as short Fourier transforms (SFTs). This is a default for many all-sky CW searches, where it assumes that the signal remains within one frequency bin during that 1800s.

Stack-slide Stack uses a set of Fourier transforms of the data known as SFTs, specifically it uses the power spectrum of these. Each of the separate Fourier transforms

(segments) is shifted up or down relative to the others to account for the Doppler modulation of the source. The power from each can then be stacked. More explanation of this can be found in [57, 58]

Hough The Hough transform is based on the stack-slide algorithm. The main difference is that the detection statistic for each segment is assigned a weight of 0 or 1 depending if it crossed a detection threshold. The Hough transform can the create a 'Hough map' which gives a view of the data in parameter space. This approach is explained in greater detail in [59, 60]. This method has been applied in two main ways known as Sky Hough [59]and Frequency Hough [60, 61].

Einstein@Home Einstein at home uses the $\mathcal{F}$-statistic mentioned above in various stages. It has a hierarchical structure where it starts with a coarse parameter space with shorter coherence times. This search then provides a list of candidates from this run in coarse parameter space. The parameter space is then more finely sampled around the parameters of the candidates and this process is repeated. The search can also increases the coherence length when searching around given candidates to improve the sensitivity of the search. This algorithm has many additions which are explained in more detail in [62, 63, 56] This provides the most sensitive all-sky CW search, however, uses a large amount of computing power. This is achieved by using a distributed computing projects, more details can be found at [64].

Time domain $\mathcal{F}$-statistic The time domain $\mathcal{F}$-statistic splits the data into narrowband segments of length $\sim$ 2 days [56]. Then a coherent search using the $\mathcal{F}$-statistic is applied to each of these segments. Values of this statistic above a threshold are stored. Coincidences are then found in each segment, where candidates are selected best on a given threshold. This is explained in greater detail in [65, 56].

Powerflux Powerflux uses a standard set of 1800s SFTs. For each point in parameter space, the power in this set of SFTs along the frequency track is recorded. This power is then weighted depending on the antenna pattern and noise of the detector. In longer stretches of $\sim$ 1 month, the weighted power is summed. Any point in parameter space which produces high power in each of these stretches is identified as a potential signal. This search can then be repeated around each candidate with a finer resolution in parameter space. This is explained in more detail and tested in [66, 56, 67]

Viterbi The Viterbi algorithm [68] has been used in [69, 70, 71, 72, 73] to search for a CWs with randomly wandering spin frequency. This algorithm was applied to specific sources, where the $\mathcal{F}$-statistic is used on short duration segments which are then incoherently combined using the Viterbi algorithm.

Table 2.1: From [56], shows the computational cost for the first 4 months of advanced LIGO for each search. One million standard units (MSU), where one standard unit is one core-hour on a standard core. ‘ Expected computational costs of searches using the first four months of advanced LIGO data with each search pipeline. These estimates are for a different data observing time from that of the MDC, and do not cover the same parameter space as each other or the MDC. The Einstein@Home searches uses the computing resources of the Einstein@Home project and is designed to run for 6 - 10 months in the Einstein@Home grid.’

Pipeline	Expected runtime of O1 search
Powerflux	6.8 MSU
Time domain $\mathcal{F}$-statistic	1.6 MSU
Frequency Hough	0.9 MSU
Sky Hough	0.9 MSU
Einstein@Home	100-170 MSU

Each of these searches uses a large computational cost. In [56] a mock data challenge (MDC) was conducted to compare the sensitivity of some of the searches, where an expected runtime for and O1 search was presented. Result in O1 for some of these searches can now be found in [74]. The results of this are shown in Tab.2.1. Even the fastest of these searches takes close to 1 million core-hours to search through four months of data. This is one of the larger current issued in CW searches as this is a slow process and running computing clusters can be costly.

2.4 Motivation

The searches described in Sec. 2.3 are computationally expensive, where the fastest takes $\sim$1 million core-hours to search through 4 months of O1 data. Many of these searches use well-modelled signals to compare to the data. This leads to the parameter space having to be finely sampled such that it is sufficiently covered. The motivation for much of the work then follows from these points. The aim was to develop searches which used minimal computational resources and could work outside of the CW model in Sec. 2.1. This thesis then outlines algorithms which can reduce this computational time of searches for CW whilst retaining as much sensitivity to CW signals as possible. There are two main sections which follow, Sec. 3 will outline an essentially un-modelled CW search method which uses the Viterbi algorithm. Sec 4 will then outline an method which used machine learning, specifically convolutional neural networks (CNNs) as both its own CW search and an extension to the search described in Sec. 3. Following chapters then explain applications of these searches.

Chapter 3

SOAP: A generalised application of the Viterbi algorithm to searches for continuous gravitational-wave signals.

The SOAP search is a semi-coherent CW search algorithm that aims to reduce the computational time needed to find a potential signal. The algorithm looks though narrow-banded time-frequency spectrograms of data to find the 'most probable track' in frequency through it. This 'most probable track' is then the most likely track which a pulsars frequency would follow. The motivation of the search is simple, if we looked at a frequency band in a spectrogram as in Fig. 3.1, we could find every possible randomly wandering track from a starting frequency bin to an end frequency bin. For each of these tracks the sum of the spectrogram power along the track can be found such that for each track there is a single value. Figure 3.1 shows a histogram of a subset of these values, where the main distribution is from tracks which are through noise. Signals which lie outside this are then tracks which follow features which are not noise like. The track which gives the maximum sum of spectrogram power is the least noise-like and therefore, can be taken as most likely to be from some signal. In Fig. 3.1 the optimum track in red shows a statistic value of ~ 1700 which is far outside the main distribution of summed powers. The red track follows that an injected signal. This demonstrates that the sum of the spectrogram power along a track which follows a signal is outside the distribution of tracks which randomly walk through noise. Therefore, it can be assumed that if the frequency track with the highest sum of spectrogram power is found, then the corresponding track is most likely to follow a signal. Given that in the example in Fig. 3.1, the spectrogram has 180 frequency bins M and 400 time segments N. After each segment the track has T possible options to jump to (in this case it is 180 options), The total number of possible tracks is MT^N. For this spectrogram this value is $\sim 2.3 \times 10^{904}$, this is an unreasonable number of tracks to possibly calculate. This is where the Viterbi algorithm [68] is useful as it can efficiently find the track which

gives the maximum sum of power. For an equivalent search the Viterbi algorithm would have to do TMN calculations for find the optimum track. A description of this method is in the following sections.

The majority of this chapter that follows has been reviewed and published as in [75]. The exceptions are work in Sec. 3.9, Sec. 3.11, Sec. 3.12 and Sec. 3.13 which is supplementary material. This was work done by the author under the supervision of Prof. Graham Woan and Dr. Chris Messenger.

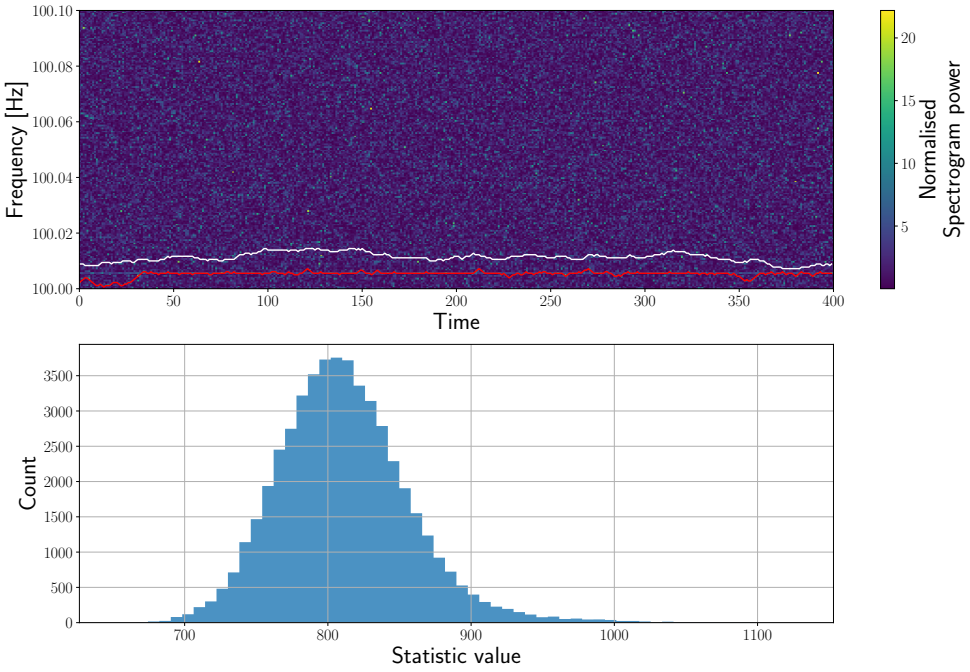


Figure 3.1: This figure shows an example of a time-frequency spectrogram which is typical of LIGO data which is searched through. Here an instrumental line has been injected at 100.006 Hz. The white track shows a random walk track though this spectrogram whereas the red line shows the track which gives the highest sum of detector power. The second panel shows a histogram of a subset of all paths which can be found through the given spectrogram from start to finish. This is a subset as the total number of paths is too large to calculate. The value of the statistic which comes from the optimal path is $\sim$ 1700. This is much larger than any of the random tracks in our subset and much larger than the mean of all tracks.

3.1 Introduction

One of the main targets for current ground based GW detectors, including LIGO [43, 11] and Virgo [44, 12], are sources of continuous gravitational waves. These are long-duration,

quasi-monochromatic sinusoidal signals that are well-modelled by a Taylor series expansion in the signal phase. A likely source of such signals are rapidly spinning non axisymmetric neutron stars. A number of possible emission mechanisms are outlined in [76, 77].

These types of GWs are expected to give strain amplitudes that are significantly below the detector's noise spectral density, and need sensitive search algorithms for detection. The most sensitive method is to use a coherent matched filter which requires knowledge of the waveform beforehand such that it can be coherently correlated with the data. This approach is used in searches for gravitational signals from known pulsars such as [49, 78, 48, 79, 80]. For broad parameter space searches, where the parameters of the signal are unknown, a large number of template waveforms must be used to sufficiently cover the parameter space. This approach rapidly becomes computationally impractical as the search space grows, so semi-coherent search methods have been developed to deliver the maximum overall sensitivity for a given computational cost. Semi-coherent searches break the data up into sections of either time or frequency and perform a coherent analysis on these sections separately. These intermediate results can then be recombined incoherently in a number of different ways to form the final search result outlined in [81, 82] and references therein.

The analysis that we present here is known as SOAP [83] and is based on the Viterbi algorithm [68]. The algorithm models a process that has a discrete number of states at discrete time steps, and computes the set of states which gives the highest probability (suitably defined) given the data. Our implementation of SOAP is intended as a stand-alone search which is naturally non-parametric and has broad applications to both searches for known signal types and signals which have an unknown frequency evolution. The algorithm works in time-frequency plane, where our 'states' are represented by the time and frequency coordinates of a potential signal. We can then find the most probable set of frequencies a possible signal could have, i.e., we can find the most probable track in frequency as a function of time. This is not the first application of the Viterbi algorithm to GW data. Another variant of the algorithm [84] has recently been used, amongst other applications, as part of a CW search to track a pulsar with randomly wandering spin frequency [69, 70, 71, 72, 73]. We develop an alternative version which is aimed to be applied more generally to search for any long duration signals using just SFTs.

In the next section we will describe the Viterbi algorithm and the basic SOAP implementation to GW time-frequency data. We then describe additional features to the algorithm, including the use of data from multiple detectors. As well as this we describe methods used to ignore instrumental effects in the data, such as incoherently summing data and a 'line aware' statistic. In the final section as well as a test of the computational cost of the search, we show results of a search performed on datasets of increasing complexity: Gaussian noise with no gaps (i.e., contiguous in time), Gaussian noise with gaps

simulating real data more accurately, and finally real LIGO data taken during the sixth science run.

3.2 Viterbi algorithm

The Viterbi algorithm is an efficient method for determining the most probable set of states (a single 'track' of steps on the time-frequency plane) in a Markov model dependent on data, where the model has a discrete number of states at each step. Rather than computing the probability of every possible track and selecting the most probable, the algorithm maximises this probability after every discrete step. As a result, a partial track which cannot ultimately be the most probable is rejected before the next step is calculated, and only a fraction of all possible tracks need to be computed to find the one that is most probable.

In this work we apply the Viterbi algorithm to a GW strain time-series to find the most probable track of a single variable-frequency signal in the noisy data. We divide the time series into N equal-length and contiguous segments $\boldsymbol{x}_j$, defining the set $D \equiv \{\boldsymbol{x}_j\}$. The 'states' in the model correspond to the frequencies a signal could have in each segment. A 'track' is a list of such frequencies $\boldsymbol{\nu} \equiv \{\nu_j\}$, where ν_j is the frequency in the segment $\boldsymbol{x}_j$.

Our objective is to calculate the most probable track given the data, i.e., the track that maximises $p(\boldsymbol{\nu} \mid D)$. Using Bayes theorem, this posterior probability can be written as

$$p(\boldsymbol{\nu} \mid D) = \frac{p(\boldsymbol{\nu})p(D \mid \boldsymbol{\nu})}{p(D)}, \tag{3.1}$$

where $p(\boldsymbol{\nu})$ is the prior probability of the track, $p(D \mid \boldsymbol{\nu})$ is the likelihood of the track (i.e., the probability of the data given the track) and $p(D)$ is the model evidence (or marginalised likelihood).

The Viterbi algorithm treats the track as the result of a Markovian process, such that the current state depends only on the previous state. It is therefore useful to split the track's prior into a set of transition probabilities such that

$$\begin{aligned} p(\boldsymbol{\nu}) &= p(\nu_{N-1}, \ldots, \nu_1, \nu_0) \\ &= p(\nu_{N-1} \mid \nu_{N-2})p(\nu_{N-2} \mid \nu_{N-3}) \ldots p(\nu_1 \mid \nu_0)p(\nu_0) \\ &= p(\nu_0) \prod_{j=1}^{N-1} p(\nu_j \mid \nu_{j-1}), \end{aligned} \tag{3.2}$$

where $p(\nu_0)$ is the prior probability that the signal in the first time step has a frequency ν_0 and $p(\nu_j \mid \nu_{j-1})$ is the prior 'transition' probability for ν_j given the frequency at the last step was ν_{j-1}.

The noise in each of the segments can be treated as independent, so the likelihood

component in Eq. 3.1 can be factorised as

$$p(D \mid \boldsymbol{\nu}) = \prod_{j=0}^{N-1} p(\boldsymbol{x}_j \mid \nu_j), \tag{3.3}$$

where $p(\boldsymbol{x}_j \mid \nu_j)$ is the likelihood of our signal having a frequency ν_j in the jth segment.

Using Eq. 3.1,3.2 and 3.3, the posterior probability is then

$$p(\boldsymbol{\nu}|D) = \frac{p(\nu_0)p(\boldsymbol{x}_0|\nu_0)\prod_{j=1}^{N-1} p(\nu_j|\nu_{j-1})p(\boldsymbol{x}_j|\nu_j)}{\sum_S \left\{ p(\nu_0)p(\boldsymbol{x}_0|\nu_0)\prod_{j=1}^{N-1} p(\nu_j|\nu_{j-1})p(\boldsymbol{x}_j|\nu_j) \right\}}, \tag{3.4}$$

where in the denominator we must sum over all possible tracks S. We require the specific track, or set of frequencies, $\hat{\boldsymbol{\nu}}$ that maximises the posterior probability. Therefore, as the denominator in Eq. 3.4 is a sum over all possible tracks, the track which maximises the posterior is the same track which maximises the numerator on the right-hand side of Eq. 3.4, i.e.,

$$p(\hat{\boldsymbol{\nu}}|D) \propto \max_{\boldsymbol{\nu}} \left[p(\nu_0)p(\boldsymbol{x}_0|\nu_0)\prod_{j=1}^{N-1} p(\nu_j|\nu_{j-1})p(\boldsymbol{x}_j|\nu_j) \right]. \tag{3.5}$$

This track also maximises the log of the probability and can be written as,

$$\begin{aligned} \log p(\hat{\boldsymbol{\nu}}|D) = \max_{\boldsymbol{\nu}} \Big\{ & \log p(\nu_0) + \log p(\boldsymbol{x}_0|\nu_0) \\ & \sum_{j=1}^{N-1} \Big[\log p(\nu_j|\nu_{j-1}) + \log p(\boldsymbol{x}_j|\nu_j) \Big] \Big\} + \text{const.} \end{aligned} \tag{3.6}$$

The Viterbi algorithm finds the most probable track $\hat{\boldsymbol{\nu}}$ by calculating the quantities in Eq, 3.6 for each frequency at each time step. In the following sections we explain how this is achieved in practice.

3.3 The transition matrix

An important concept when using the Viterbi algorithm is the'transition matrix' T, which is defined as the matrix that stores the prior log-probabilities $\log p(\nu_j \mid \nu_{j-1})$. These transition probabilities depend only on the size and direction of the transition, and in our case correspond to a jump in frequency when moving from the $(j-1)$th to the jth state. It is within the transition matrix that we impose some loose model constraints. For example it is usual in the time-frequency plane for frequencies to only have discrete

values (frequency bins) and a track might only be allowed to move by one bin in each time step, restricting it to a up, centre or down (UCD) transition or 'jump' or equivalently setting the size of the first dimension of the transition matrix $n_1 = 3$. We can also impose that the transition probabilities are independent of the current track location in frequency, i.e. $p(\nu_j \mid \nu_{j-1}) = p(\nu_{j+k} \mid \nu_{j+k-1})$. This leads to the transition matrix containing only three numbers, corresponding to the three prior log-probabilities that the track was in the corresponding UCD frequency bin at the previous time step. These numbers are chosen to reflect the prior probability of a frequency deviation in the track and depend on the class of signals that one wishes to detect. For the majority of examples that follow, a symmetric transition matrix is used, i.e. the probability of a transition up a frequency bin is equal to the probability of a transition down a frequency bin. This allows us to parameterise the one dimensional transition matrix with a single value, this value is the ratio of the probability of a transition to the same frequency bin, to either up or down a frequency bin.

In later sections we will consider more complex situations in which the transition matrix describes the prior probability associated with sequences of even earlier transitions ('memory') and the case where there are multiple detectors. In these cases the number of dimensions of the transition matrix can grow substantially to account for the extra complexity of the problem.

3.4 Single detector

We will first consider the simple case of a single dataset D, generated by a single gravitational wave detector, and consider only a one-dimensional transition matrix. We will make use of discrete Fourier transforms so that frequencies, and hence the track frequencies, are also discrete. These frequencies will be indexed by k and therefore $\nu_j \to \nu_{j,k} = k(j)\Delta f$ where $\Delta f = 1/T$ is the frequency bin width for a segment of duration T.

The Viterbi algorithm determines the most probable track on the time-frequency plane by calculating the value of Eq. 3.6 for every discrete Fourier frequency, incrementally in time. In other words, at each time segment it finds the most probable earlier track which ends at each particular frequency. On reaching the final segment it can look back to identify the most probable track connecting segment 1 to segment N.

There are two main components to Eq. 3.6: the transition probabilities $p(\nu_j \mid \nu_{j-1})$ and the likelihoods $p(\boldsymbol{x}_j \mid \nu_j)$. The transition probabilities are pre-calculated and stored in a transition matrix according to Sec. 3.3 above. To calculate the likelihood we follow the approach of [85] which gives, under the assumption of a single sinusoidal signal in additive

Gaussian noise in data segment $\boldsymbol{x}_j$,

$$p(\boldsymbol{x}_j \mid \nu_{j,k}, \sigma_{j,k}, I) \propto \exp\left[C(\nu_{j,k})\right]. \tag{3.7}$$

where $C_{j,k}(\nu_{j,k})$ is the Schuster periodogram normalised to the noise variance at frequency $\nu_{j,k}$ of segment j. This is equivalent to the log-likelihood, and is defined as

$$C(\nu_{j,k}) \equiv C_{j,k} = \frac{1}{\sigma_{j,k}^2}\frac{1}{N_\mathrm{s}}\left|\sum_{r=0}^{N_\mathrm{s}-1} x_{j,r}\mathrm{e}^{i\nu_{j,k}t_r}\right|^2, \tag{3.8}$$

where N_s is the number of data points in each segment and t_r is the time corresponding to $x_{j,r}$, the rth sample in the jth data segment. $\sigma_{j,k}^2$ is the noise variance and is calculated as an estimate of the noise power spectral density (PSD) in the kth sample and the jth data segment. It is worth noting at this point that it is also possible to write this as a likelihood ratio, and therefore write out detection statistic as a log-odds ratio, however, we will discuss this in more depth in Sec. 3.8. The log-likelihoods of each segment can be calculated at discrete frequencies before running the algorithm by computing the power spectra for each segment from discrete Fourier transforms of the data. In the GW field these standard data forms are known as SFTs.

The Viterbi algorithm records two quantities for each frequency and time bin: The first, $V_{j,k}$, contains the value defined by Eq. 3.6, which is the log-probability of the most probable path ending in position j, k. The second, $A_{j,k}$, is the transition, or 'jump', used to achieve the most probable path. The algorithm can be divided into three main sections: initialisation, iteration and identification. These three sections are described in pseudo-code in Alg. 3.1 and a simple demonstration of the algorithm at work is shown in Fig. 3.2.

Initialisation The two parts of Eq. 3.6, $\log p(\nu_0)$ and $\log p(\boldsymbol{x}_0 \mid \nu_0)$, must be computed before the main recursive part of the algorithm can start. Therefore, the initialisation section (lines 5–8) in Alg. 3.1 calculates the first column in the lower panel of Fig. 3.2. A priori, there is no preferred initial frequency, so we take the log-prior $\log p(\nu_{0,k})$ to be uniform over the complete frequency range. As a result, this is does not affect the maximisation for any jump, therefore, can be omitted from the calculation. We then use the pre-calculated log-likelihood values $C_{0,k}$ to fill the track probabilities $V_{0,k}$. There is no previous position to jump from in this case, so the transition probabilities are irrelevant and $A_{0,k}$ are set to zero.

Iteration The main part of the calculation is the sum in Eq. 3.6. Lines 11–16 in Alg. 3.1 calculate the most probable tracks that end at each frequency bin for each segment

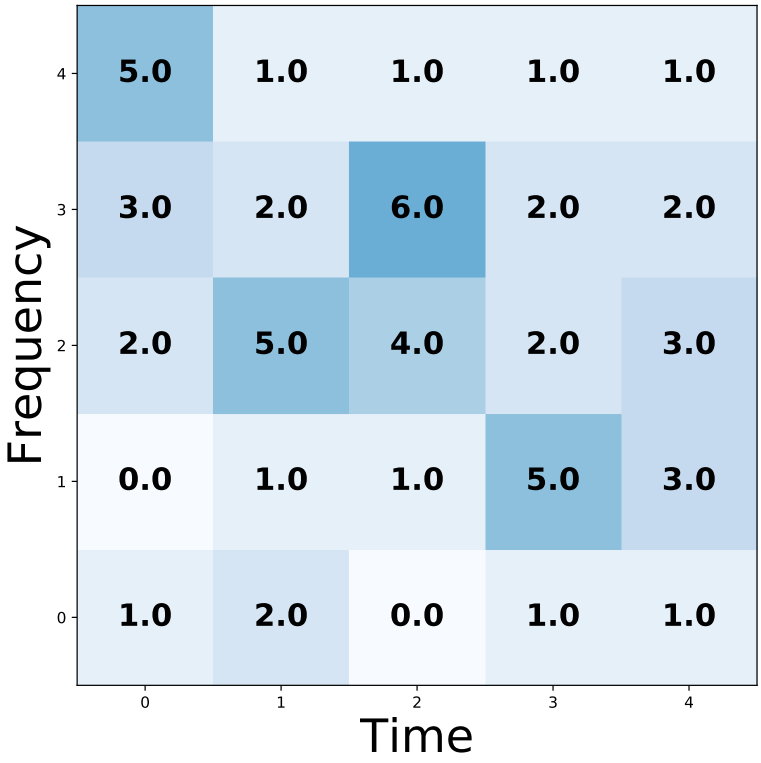
Input: C, T {log-likelihood,transition matrix}
Output: $\hat{\boldsymbol{\nu}}$, V, A {most probable track, track probabilities, jumps}

Initialisation
for Frequency $(\nu_{0,k})$, $k = 0 \rightarrow M-1$ **do**
  $V_{0,k} = C_{0k}$
  $A_{0,k} = 0$
end for

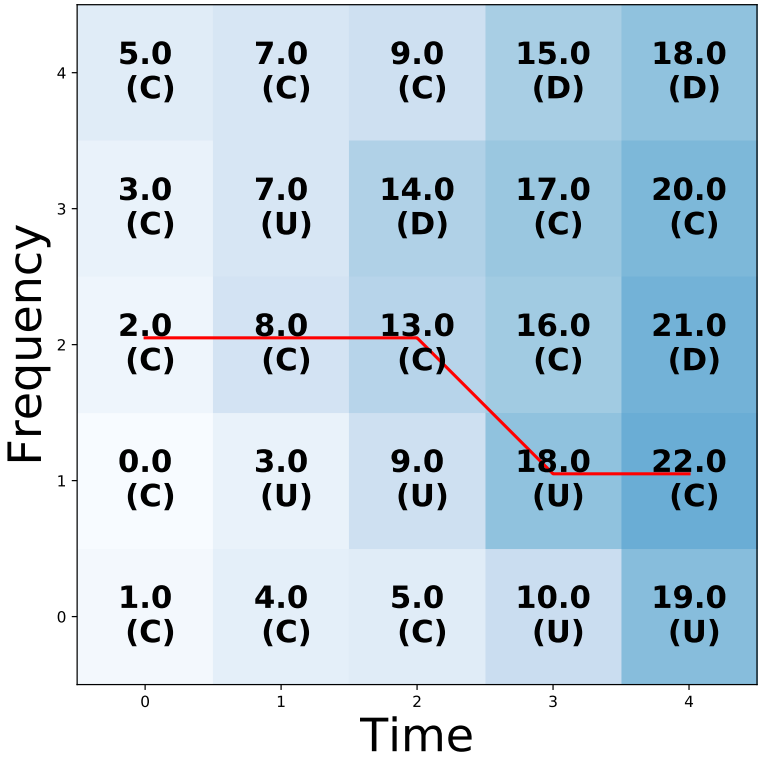
Iteration
for Segment, $j = 0 \rightarrow N-1$ **do**
  for Frequency $(\nu_{j,k})$, $k = 0 \rightarrow M-1$ **do**
    $V_{j,k} = \max_i(C_{j,k} + T_i + V_{j-1,j+i})$
    $A_{j,k} = \underset{i}{\text{argmax}}(C_{j,k} + T_i + V_{j-1,j+i})$
  end for
end for

Identification
$\hat{\nu}_{N-1} = \text{argmax}_k(V_{N-1,k})$
for Segment, $j = N-1 \rightarrow 0$ **do**
  $\hat{\nu}_j = \hat{\nu}_{j+1} + A_{j,\nu_{k+1}}$
end for

ALGORITHM 3.1: The Viterbi algorithm in pseudo-code. N is the number of segments, M is the number of frequency bins in each segment. Here the maximisations over i run between $\pm(n_1-1)/2$ where n_1 is the size of the transition matrix. The values from Eq. 3.6 are stored in V, and the jumps are stored in A. The most probable track is denoted by $\hat{\boldsymbol{\nu}}$.



(a) The input data



(b) The log-probabilities, jumps, and most probable path

Figure 3.2: Fig. 3.2a shows the observed data, i.e the log-likelihood values $C_{j,k}$. Fig. 3.2b shows the calculated log-probabilities $V_{j,k}$. $A_{j,k}$ is shown in parentheses, where the UCD components correspond to $i = [-1, 0, 1]$ respectively. The red line shows the path that gives the maximum probability. The transition matrix for the UCD jumps is $[0, 1, 0]$ and corresponds to the un-normalised prior log-probabilities of these jumps occurring.

by using

$$V_{j,k} = \max_i (C_{j,k} + T_i + V_{j-1,k+i}), \tag{3.9}$$

where i is the size and direction of the jump. For example, in Fig. 3.2 columns 1–4 are calculated in order using Eq. 3.9, where it maximises over three possible previous positions in frequency. These positions are the frequency bins UCD of the current position. The size and direction of the jump, i, which gives the maximum probability is then saved to $A_{j,k}$. These are shown in parentheses below the log-probabilities in Fig. 3.2 where UCD correspond to values of $i = [-1, 0, 1]$ respectively.

Identification The final stage of the algorithm identifies the most probable track. This is done by initially finding the highest log-probability values in the final time segment, $\max_k(V_{N-1,k})$ (line 19 in Alg. 3.1). In the lower panel of Fig. 3.2 this is located at position $j, k = 4, 1$ with $V_{4,1} = 22$. To find the track which corresponds to this, the values in A_{jk} are followed backwards from this position (lines 20–21). For example, in Fig. 3.2 the final position is $j, k = 4, 1$ and $A_{j,k} = \text{Center} = 0$, this means that at the previous segment the most probable track was at position $j, k = 4-1, 1+0 = 3, 1$. At this time $A_{3,1} = R = 1$, therefore, the next track element is at $j, k = 3-1, 1+1 = 2, 2$. This then continues until $j = 0$ whereupon these retraced positions constitute the most probable track, highlighted in red in Fig. 3.2.

The most probable track is the one traced backwards from the highest probability final segment frequency position. However, tracks can also be traced back from any of the end-frequency positions, returning the most probable track conditional on a given final position. Such tracks should not be confused with the being equal to the second, third, fourth, etc. most probable tracks. Information regarding the rankings and properties of all possible tracks (excluding the most probable and conditionally most probable tracks) is lost during the maximisation procedures computed at each stage in the algorithm – a necessary consequence of the algorithm's speed and efficiency.

3.5 Multiple detectors

If there are Q detectors operating simultaneously we have Q sets of data which can be combined appropriately to provide input to the Viterbi search described above. We must also modify the allowed transitions encoded within the transition matrix to take account of the extra prior constraints that are now available.

The received instantaneous frequency of a given astrophysical signal will be nearly the same for all ground-based GW detectors, and our algorithm should be sensitive to tracks that show this consistency in frequency. However there *will* be small differences between the frequencies measured at detectors that are not co-located, due to differential Doppler

shifts caused by Earth rotation. As a result the signal could fall in different frequency bins at each detector.

To account for these small differences in signal tracks in each detector, we reference the observed tracks to a third (pseudo) detector located at the centre of the Earth which would be insensitive to Earth spin. The signal frequencies in each real detector are then allowed to vary within a certain number of frequency bins from the track in the reference detector. In the examples that follow, we only consider the possibilities that the track in each real detector is no more that one frequency bin away from the reference track. We can tune the length of the SFTs to ensure this is a valid assumption. As well as differences in signal frequency, due to antenna patterns and other effects, the measured signal amplitude may differ between the detectors. In the following example we assume that the signal has the same amplitude in each detector, however, in Sec. 3.8 we discuss the case where they differ.

We will now show how the algorithm in Sec. 3.4 can be modified to handle a two-detector network (i.e., $Q = 2$), however any number of detectors can easily be accommodated. In the two detector case the joint probability of two (real) tracks, $\nu^{(1)}$ and $\nu^{(2)}$, and the geocentric track ν, given the data, is

$$\begin{aligned} p(\nu, \nu^{(1)}, \nu^{(2)}|D^{(1)}, D^{(2)}) \propto p(\nu)p(\nu^{(1)}, \nu^{(2)}|\nu) \\ p(D^{(1)}|\nu^{(1)})p(D^{(2)}|\nu^{(2)}), \end{aligned} \tag{3.10}$$

where $D^{(1)}$ and $D^{(2)}$ represent the data from the two detectors. The main difference between this and that described in Sec. 3.4 is that the track probabilities $V_{j,k}$ are stored for the geocentric pseudo-detector. The main iterative calculation (defined for the single detector case in Eq. 3.9) now becomes

$$V_{j,k} = \max_{i,l,m}(C^{(1)}_{j,k+l} + C^{(2)}_{j,k+m} + T_{i,l,m} + V_{j-1,k+i}), \tag{3.11}$$

where $C^{(1)}$ and $C^{(2)}$ refer to the log-likelihoods in detectors 1 and 2 respectively and the transition matrix T is an $n_1 \times n_2 \times n_3$ matrix, where n_1 dimension refers to the jump from the previous time step, n_2 and n_3 refer to the relative frequency positions in each real detector. The transition matrix is now three-dimensional and holds the prior log-probabilities of $p(\nu)$ and $p(\nu^{(1)}, \nu^{(2)}|\nu)$. We now need to maximise over three indices: i, l and m. The index i refers to the size and direction of the jump at the geocentre (as before). The indices l and m refer to the number of frequency bins by which the two real tracks deviate from the geocentre track. For example, if the most probable track in the geocentred detector is in bin $j, k = 5, 12$ and the values of $i, l, m = 0, -1, 1$, then detector 1 is in position $j, k = 5, 11$ and detector 2 is in position $j, k = 5, 13$ and the geocentred track was in the position $j, k = 4, 12$ at the previous time step. As a result, the track at

the geocentre is only affected by Doppler modulations from the Earth's orbit whereas the tracks in the real detectors include Doppler modulations from the Earth's spin.

At every time step the frequency bin position for each real detector is forced to be within n_l or n_m bins of the track in the geocentred detector, where n_l and n_m depend on how much each detector could possibly be Doppler shifted. As mentioned previously, we only consider the case where $n_l = 1$ and $n_m = 1$, allowing the track from each real detector to be at most one frequency bin away from the geocentred track position. While we tune the SFT length to keep this condition for different frequencies, it is also possible to tune the values of n_l and n_m to get a similar effect. The implementation of the multi-detector algorithm is similar to the single detector case described in Sec. 3.4. However in the single detector case there is only a single variable to be maximised over for each time-frequency bin. This variable is the frequency jump from the position in the previous segment. For the multi-detector case there are at least three variables to be maximised over: the probability of the jump, i, at the geo-centre and the probability of the signal being in the surrounding positions in each on Q real detectors, $l, m, \ldots$. The values of $i, l, m, \ldots$ are then saved to $A_{j,k}$ and are ultimately used to reconstruct the most probable consistent tracks in each real detector.

As in Sec. 3.4, there are three main sections: Initialisation, iteration, and the identification. For the multi-detector case each element is modified as follows.

Initialisation The first-row calculation (lines 5–8) in Alg. 3.1, are now modified to additionally maximise over the real detector track positions l and m. For each time-frequency bin the maximum sum of the log-likelihoods is saved together with the frequency locations of the corresponding tracks in the real detectors. The index $i = 0$ is kept constant as there is no previous position.

Iteration To process the subsequent time segments, lines 13–14 in Alg. 3.1 are modified to account for two (or more) detectors. Line 13 of Alg. 3.1 is changed to calculate Eq. 3.11, the log-probability of a track at the geocentre ending in bin j, k given that signal is in the real detector positions of $j, k+l$ and $j, k+m$. Line 14 is then modified so that $A_{j,k}$ stores the jump values, i, and the real detector positions, l and m, which returned the highest probability.

Identification The most probable track is identified in the same way as for the single detector case, first by finding the maximum value in the final time step of $V_{j,k}$ (line 19 in Alg. 3.1). The track at the geocentre can then be found by iteratively following the jump values stored in $A_{j,k}$ back from this position. The track in each of the real detectors is determined by using the values of l and m indices also stored in $A_{j,k}$ to find the relative position of the track in each real detector compared to the geocentre.

This method can be extended to more than two detectors by including additional datasets and expanding the corresponding number dimensions of the maximisation procedures in the iterative steps.

3.6 Memory

In this section we extend the basic Viterbi algorithm to improve its sensitivity to non-stochastic signals where there is some knowledge of its frequency evolution. We do this by including a form of 'memory' and this extension applies to both the single and multiple-detector cases. Rather than considering only the previous step in our decision-making process, we now include the previous $m+1$ steps and expand the transition matrix to include these values. A memory of $m = 0$ therefore corresponds to the methods described in previous sections. With a non-zero memory the transition matrix can a-priori make certain sequences of jumps more probable and assign different prior probabilities for these jump sequences e.g., 'up then centre' may be less preferable to 'centre then centre'. As a result we can increase the chance of the most probable track matching an expected astrophysical signal. In a single detector search with a memory of $m = 1$, if we only allow UCD transitions, then for every frequency bin we save 3 values. These are proportional to the log-probabilities of a track coming from a UCD bin in the previous time step, where the maximisation is over the corresponding UCD bins two time steps back. Equation 3.11 then is then modified to,

$$V_{j,k,s} = \max_{h}(C_{j,k} + T_{s,h} + V_{j-1,k+s,k+s+h}), \tag{3.12}$$

where s and h refer to the UCD jumps at the time step $j-1$ and $j-2$ respectively. Similar to the previous two sections, the algorithm is split into three parts: initialisation, iteration, and the track identification:

Initialisation The initialisation process needs to populate the first $m+1$ steps before the main iteration can start. At the first time step, the elements $V_{0,k,s}$ are set to the log-likelihoods $C_{0,k}$ as in Sec. 3.4. There is no previous time step, so the element s is not relevant. At the second time step, $V_{1,k,s}$ is calculated using Eq. 3.12, where there is no maximisation over h, it is assumed to be 0, or a center jump. As there is no data before $j = 0$, the maximisation at this point will always return the jump which has the largest prior probability, which in this case is a center jump. Therefore, the maximisation returns the same value for all frequency bins and can be set to a center jump.

Iteration For all following time steps the values for each element of $V_{j,k,s}$ in Eq. 3.12 are calculated. This quantity is proportional to the log-probability of the track ending

in time-frequency bin j, k, which was in the previous position of $j-1, k+s$. The corresponding value of h that maximised the log-probability of the track is recorded in $A_{j,k,s}$.

Identification The most probable track is identified in a similar way to the non-memory cases, by finding the highest-valued last element, $V_{N-1,k,s}$. The values of s and h are then followed back to find the most probable track. As an example, let us assume the most probable track finishes in bin $j, k, s = 10, 5, 0$, where the value of m is $A_{10,5,0} = 1 =$ up. The previous position is then $j, k, s = 10-1, 5+s, m = 10-1, 5+0, 1 = 9, 5, 1$ with a value $A_{9,5,1} = 0 =$ Center, and the next track position is $j, k, s = 9-1, 5+1, 0 = 8, 6, 0$ etc. The values of j, k along this track describes most probable path.

The number of elements over which one must search increases rapidly with memory length, and has a strong impact on the computational cost of the analysis. For the single detector Viterbi approach the number of calculations made is $3 \times N \times M$ if we only allow UCD jumps, where N and M are the number of time are frequency bins respectively. When memory is included this increases to $3^{m+1} \times N \times M$.

3.7 Summed input data

In this section a method of incoherently-summing a set of SFTs to increase the SNR of a signal in a segment is outlined. To be more precise, it is actually the log-likelihoods which are summed, i.e. the quantity in Eq. 3.8. We can write the new summed set of data F_j as,

$$F_j = \sum_{i}^{N_s} C_{i,k} \tag{3.13}$$

where N_s is the number of SFTs to sum together and the log-likelihood $C(\nu_{i,k})$ is defined in Eq. 3.8. We can see this is possible by looking at Eq. 3.7, where we can use the product of likelihoods,

$$\begin{aligned} p(D \mid \nu) &\propto p(x_1, x_2 \ldots x_n \mid \nu) \\ &\propto p(x_1 \mid \nu) \ldots p(x_n \mid \nu) \\ &\propto \exp\left(\sum_i C_{j,k}\right). \end{aligned} \tag{3.14}$$

If the data contains gaps where the detector was not observing, then we fill the gaps in the power spectrum with a constant value which is the expectation value of the log-likelihood. The procedure of filling in the gaps of the data is completed before any summing. There-

fore, the data should have the same mean regardless of how much real data is in each sum. In the examples that follow, we sum the SFTs over the length of one day.

The main motivation for summing the data is to increase the SNR of a signal in the segments. The risk is that a signal can move between adjacent frequency bins during a day. To reduce this risk, we choose the frequency bin width such that it is more likely that a signal will be contained within a single frequency bin that cross a bin edge. In practice, to ensure that this is true, the segment or SFT length and the number of segments which are summed can be tuned for each search. As well as increasing the SNR, summing over one day should average out the antenna pattern. This means that the log-likelihood value in any bin should be more similar between detectors, however, there is still some variation due to the sky localisation and polarisation.

This also has two main effects on the transition matrix, the first is that as each segment of data is now one day long, a jump between frequency bins is far more likely, therefore, the transition matrix elements are modified to account for this. The second is that as the data is averaged over one day, the signal should remain is the same frequency bin between detectors, therefore, there is no longer a need for the multi-dimensional transition matrix described in Sec. 3.5.

The volume of the data is also reduced by a factor of $1/N_s$, therefore, the time taken for the algorithm to run is also reduced by the same factor.

3.8 Line-aware statistic

The multiple-detector algorithm described in Sec. 3.5 returns the most probable track of a common signal assumed to be in Gaussian noise. As a consequence the algorithm will return large values of the log-likelihood even if there are inconsistent values of SFT power between the detectors, either from non-Gaussian noise or because the signal is not equally strong in the two detectors. However a signal with unequal power in the two detectors is more likely to be a non-Gaussian instrumental line than an astrophysical signal. The line-aware statistic described in this section is designed to make the search more robust to such instrumental artefacts within realistic non-Gaussian data whilst maintaining sensitivity to astrophysical signals.

For most of the analysis examples presented here we use data which is the incoherent sum of 30-minute normalised SFTs over a day (described in more detail in Sec. 3.7). As a result the effects of the detector antenna patterns and of differential Doppler shifts are significantly reduced, and any signal should have a broadly similar summed log-likelihood in the same frequency bin in each detector. The statistic can then be modified such that we expect a similar log-likelihood in each detector.

We first consider the model of Gaussian noise with no signal present. Within a sin-

gle summed segment, the likelihood of Gaussian noise at frequency ν is given by a χ^2 distribution,

$$p(F_j|\nu_j, M_{\rm N}, I) = \frac{1}{2^{d/2}\Gamma(d/2)} F_j^{d/2-1} \exp\left\{\frac{F_j}{2}\right\} \tag{3.15}$$

where F_j is the frequency domain power summed over sub-segments within a single day, as described in Sec. 3.7 and d is the number of degrees of freedom, equal to twice the total number of summed SFTs. $M_{\rm N}$ represents the model that the data is simply Gaussian noise. In the presence of a signal (model $M_{\rm S}$), the power should follow a non central χ^2 distribution in which the non-centrality parameter λ is the square of the SNR, ($\lambda = \rho^2_{\rm opt}$), i.e.

$$\begin{aligned} p(F_j|\nu_j, \lambda, M_{\rm S}, I) = \frac{1}{2}\exp\left\{-\frac{F_j+\lambda}{2}\right\}\left(\frac{F_j}{\lambda}\right)^{d/4-1/2} \\ I_{d/2-1}\left(\sqrt{\lambda F_j}\right). \end{aligned} \tag{3.16}$$

If a signal is present we therefore expect the SFT powers in both detectors to follow Eq. 3.16. Assuming for the moment that the noise variance is the same in both, we can determine the evidence for model $M_{\rm S}$ by marginalising over λ,

$$\begin{aligned} p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, M_{\rm S}, I) = \int_0^\infty p(\lambda, w_{\rm S}) \\ p(F_j^{(1)}|\nu_j, \lambda, M_{\rm S}, I) p(F_j^{(2)}|\nu_j, \lambda, M_{\rm S}, I) d\lambda. \end{aligned} \tag{3.17}$$

We set the prior on λ to be an exponential distribution of width w, this is done somewhat arbitrarily as we expect the majority of signals to have a low SNR. This distribution follows,

$$p(\lambda, w) = \exp\left(\frac{-\lambda}{w}\right). \tag{3.18}$$

On the other hand, if an instrumental line is present in one of the detectors we expect to see signal-like power in that detector and noise-like power in the other. The evidence for this 'line' model ($M_{\rm L}$) is therefore

$$\begin{aligned} p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, M_{\rm L}, I) = \int_0^\infty p(\lambda, w_{\rm L}) \\ \Big[p(F_j^{(1)}|\nu_j, M_{\rm N}, I) p(F_j^{(2)}|\nu_j, \lambda, M_{\rm S}, I) \\ + p(F_j^{(1)}|\nu_j, \lambda, M_{\rm S}, I) p(F_j^{(2)}|\nu_j, M_{\rm N}, I)\Big] d\lambda, \end{aligned} \tag{3.19}$$

The third option to consider is the simple case of approximately Gaussian noise in both

of the detectors,

$$p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, \lambda, M_{\rm G}, I) = p(F_j^{(1)} \mid \nu_j, M_{\rm G}, I) \\ p(F_j^{(2)} \mid \nu_j, M_{\rm G}, I). \tag{3.20}$$

The posterior probability of model $M_{\rm GL}$, which contains the probability of Gaussian noise or Gaussian noise with a line in one detector, (taken as mutually exclusive) is

$$p(M_{\rm GL} \mid F_j^{(1)}, F_j^{(2)}, \nu_j, I) = p(M_{\rm G} \mid F_j^{(1)}, F_j^{(2)}, \nu_j, I) \\ + p(M_{\rm L} \mid F_j^{(1)}, F_j^{(2)}, \nu_j, I), \tag{3.21}$$

where we assume that $M_{\rm G}$ and $M_{\rm L}$ are mutually exclusive.

We can now find the posterior odds ratio for the presence of a signal over noise or a line,

$$\begin{aligned} O_{\rm SGL}(F_j^{(1)}, F_j^{(2)} \mid \nu_j) &= \frac{p(M_{\rm S} \mid F_j^{(1)}, F_j^{(2)}, \nu_j)}{p(M_{\rm GL} \mid F_j^{(1)}, F_j^{(2)}, \nu_j)} = \frac{p(M_{\rm S} \mid F_j^{(1)}, F_j^{(2)}, \nu_j)}{p(M_{\rm G} \mid F_j^{(1)}, F_j^{(2)}, \nu_j) + p(M_{\rm L} \mid F_j^{(1)}, F_j^{(2)}, \nu_j)} \\ &= \frac{p(M_{\rm S}) p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm S}, \nu_j)}{p(M_{\rm G}) p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm G}, \nu_j) + p(M_{\rm L}) p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm L}, \nu_j)} \\ &= \frac{p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm S}, \nu_j) p(M_{\rm S}) / p(M_{\rm G})}{p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm G}, \nu_j) + p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm L}, \nu_j) p(M_{\rm L}) / p(M_{\rm N})} \end{aligned} \tag{3.22}$$

In practice it is convenient to use the log odds ratio,

$$\begin{aligned} \log \left[O_{\rm SGL}(F_j^{(1)}, F_j^{(2)}) \right] &= \log \left[p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm S}) \right] \\ &\quad - \Big[\log \Big(p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm G}) \\ &\quad + p(F_j^{(1)}, F_j^{(2)} \mid M_{\rm L}) p(M_{\rm L}) / p(M_{\rm G}) \Big) \Big] \end{aligned} \tag{3.23}$$

As we are only interested in the maximum of $\log \left[O_{\rm SGL}(F_j^{(1)}, F_j^{(2)}) \right]$, the factor $\log [p(M_{\rm S})/p(M_{\rm G})]$ can be dropped from the expression.

In this version of the Viterbi algorithm, rather than storing a value proportional to the log-probabilities as in Sec. 3.5, here we store a value proportional to the log-odds ratio. Here we take the log-odds ratio defined in Eq. 3.23, which is the log-odds of a signal having a similar power in each detector, and add the log-prior odds $p(\boldsymbol{\nu} \mid M_S)/(p(\boldsymbol{\nu} \mid M_N) + p(\boldsymbol{\nu} \mid M_L))$ which is the log-prior or any particular track. By assuming that the track transitions for the line and noise model are equally probable for any jump, we set the denominator of

the prior-odds is a constant b. This then means Eq. 3.11 is modified to,

$$\begin{aligned}\hat{V}_{i,j} = \max_{k,l,m} \big(T_{k,l,m} + b + V_{i-1,j+k} \\ + \log\left[O_{\mathrm{SGL}}\left(F_j^{(1)}, F_j^{(2)}\right)\right]\big),\end{aligned} \tag{3.24}$$

where $\hat{V}$ refers to a log-odds ratio. The maximised statistic now has three tune-able parameters: the width, w_S in Eq. 3.18, on the prior for a signal SNR squared, $p_{\mathrm{S}}(\lambda)$, the width, w_L of the prior in the case of a line, $p_{\mathrm{L}}(\lambda)$, and the ratio of the prior on the line and noise models, $p(M_{\mathrm{L}})/p(M_{\mathrm{G}})$. These parameters are optimised for each search, where we initially estimate the SNR of a signal we hope to be sensitive to in each time slice, then use this as a guide for the width of the signal prior. This is then repeated for an expected line SNR and this is used for the width of the line prior. The ratio of line and noise models runs in the range 0 to 1, we set this limit as we do not expect an instrumental line to be as likely as Gaussian noise in any particular frequency bin.

This line-aware statistic can be applied in a more powerful way when we use multiple detectors and is similar to the approach in [86]. The multiple-detector algorithm described in Sec. 3.5 returns the most probable track of a common signal assumed to be in Gaussian noise. As a consequence the algorithm will return large values of the log-likelihood even if there are inconsistent values of SFT power between the detectors, either from non-Gaussian noise or because the signal is not equally strong in the two detectors. However a signal with unequal power in the two detectors is more likely to be a non-Gaussian instrumental line than an astrophysical signal. The line-aware statistic described in this section is designed to make the search more robust to such instrumental artefacts within realistic non-Gaussian data whilst maintaining sensitivity to astrophysical signals.

For most of the analysis examples presented here we use data which is the incoherent sum of 30-minute normalised SFTs over a day (described in more detail in Sec. 3.7). As a result the effects of the detector antenna patterns and of differential Doppler shifts are significantly reduced, and any signal should have a broadly similar summed log-likelihood in the same frequency bin in each detector. The statistic can then be modified such that we expect a similar log-likelihood in each detector.

In a similar way to the single-detector case, we can write out the evidence for each of the three models as follows. If a signal is present we therefore expect the SFT powers in both detectors to follow Eq. 3.16. Assuming for the moment that the noise variance is the same in both, we can determine the evidence for model M_{S} by marginalising over λ,

$$\begin{aligned}p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, M_{\mathrm{S}}, I) = \int_0^\infty p(\lambda, w_{\mathrm{S}}) \\ p(F_j^{(1)}|\nu_j, \lambda, M_{\mathrm{S}}, I) p(F_j^{(2)}|\nu_j, \lambda, M_{\mathrm{S}}, I) d\lambda.\end{aligned} \tag{3.25}$$

We set the prior on λ the same as in the single detector case in Eq. 3.18. In this case, if an instrumental line is present in one of the detectors we expect to see signal-like power in that detector and noise-like power in the other. The evidence for this 'line' model (M_{L}) is therefore

$$\begin{aligned} p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, M_{\mathrm{L}}, I) = \int_0^\infty p(\lambda, w_{\mathrm{L}}) \\ \Big[p(F_j^{(1)}|\nu_j, M_{\mathrm{N}}, I)p(F_j^{(2)}|\nu_j, \lambda, M_{\mathrm{S}}, I) \\ +p(F_j^{(1)}|\nu_j, \lambda, M_{\mathrm{S}}, I)p(F_j^{(2)}|\nu_j, M_{\mathrm{N}}, I)\Big]\, d\lambda, \end{aligned} \tag{3.26}$$

The third option is the simple case of approximately Gaussian noise in both of the detectors,

$$\begin{aligned} p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, \lambda, M_{\mathrm{G}}, I) = p(F_j^{(1)} \mid \nu_j, M_{\mathrm{G}}, I) \\ p(F_j^{(2)} \mid \nu_j, M_{\mathrm{G}}, I). \end{aligned} \tag{3.27}$$

We can now find the posterior odds ratio for the presence of a signal over noise or a line by following the same steps as in Eq. 3.22. Once again we write this as a log-odds ratio,

$$\begin{aligned} \log\left[O^{(2)}_{\mathrm{S/GL}}(F_j^{(1)}, F_j^{(2)})\right] = \log\left[p(F_j^{(1)}, F_j^{(2)} \mid M_{\mathrm{S}})\right] \\ - \Big[\log\Big(p(F_j^{(1)}, F_j^{(2)} \mid M_{\mathrm{G}}) \\ +p(F_j^{(1)}, F_j^{(2)} \mid M_{\mathrm{L}})p(M_{\mathrm{L}})/p(M_{\mathrm{G}})\Big)\Big] \end{aligned} \tag{3.28}$$

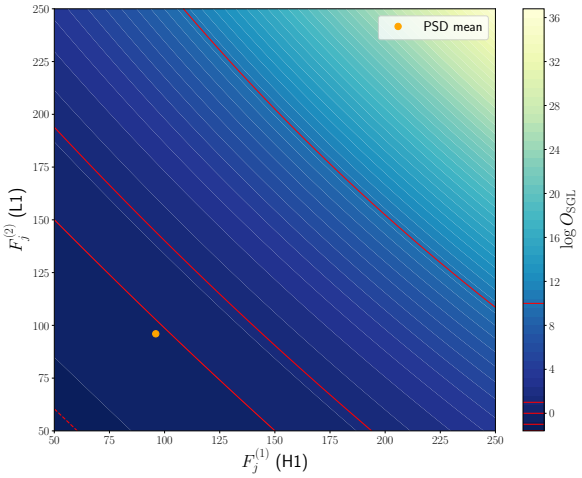
The factor $\log\left[p(M_{\mathrm{S}})/p(M_{\mathrm{G}})\right]$ can again be dropped from the expression.

For the multi-detector case we then modify Eq. 3.11 to,

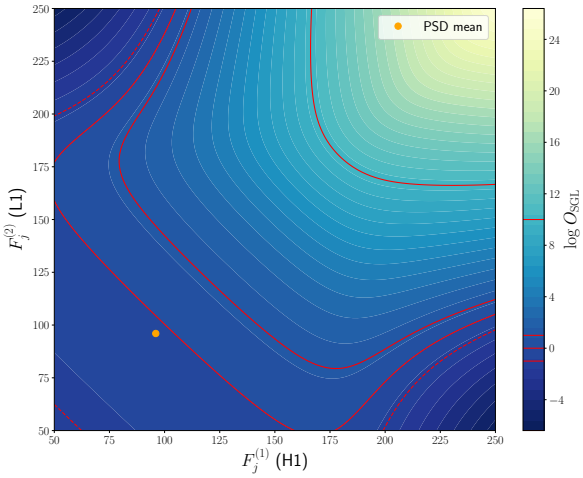
$$\begin{aligned} \hat{V}_{i,j} = \max_{k,l,m}\big(T_{k,l,m} + b + V_{i-1,j+k} \\ + \log\left[O^{(2)}_{\mathrm{S/GL}}\left(F_j^{(1)}, F_j^{(2)}\right)\right]\big), \end{aligned} \tag{3.29}$$

where $\hat{V}$ refers to a log-odds ratio. This is then optimised over the same three parameters as the single detector case.

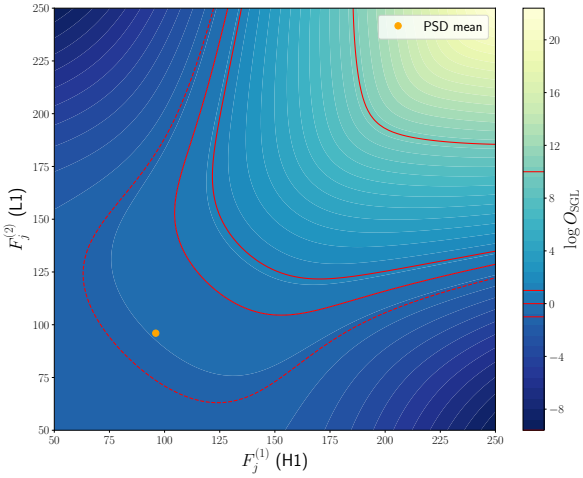
Fig. 3.3 shows an example of the output of the statistic in Eq. 3.28 for different fast Fourier transform (FFT) powers F.



(a) This shows the line-aware statistic as a function of it input from each detector. This example is for parameters $p(\lambda, w_{\mathrm{S}}) = 4$, $p(\lambda, w_{\mathrm{L}}) = 0$ and $p(M_{\mathrm{L}})/p(M_{\mathrm{G}}) = 0$. So the line part of the statistic is not operating.So the line part of the statistic is not operating.



(b) This shows the line-aware statistic as a function of it input from each detector. This example is for parameters $p(\lambda, w_{\mathrm{S}}) = 4$, $p(\lambda, w_{\mathrm{L}}) = 5$ and $p(M_{\mathrm{L}})/p(M_{\mathrm{G}}) = 0.03$. Here we include the line part of the statistic.



(c) This shows the line-aware statistic as a function of it input from each detector. This example is for parameters $p(\lambda, w_{\mathrm{S}}) = 4$, $p(\lambda, w_{\mathrm{L}}) = 5$ and $p(M_{\mathrm{L}})/p(M_{\mathrm{G}}) = 1$. Here the effect of lines is expected to be larger than the previous panel on the search. Therefore, the statistic forces the two detectors to have more similar power.

Figure 3.3: Lookup tables using the line aware statistic in Eq. 3.29. The PSD mean is the expected mean of a χ^2 distribution with 48 degrees of freedom, i.e. the expected power from out summed spectrograms F_j.

3.9 Line aware statistic for consistent amplitude

In Sec. 3.8 the 'line aware' statistic was designed to penalise high SFT powers in a single detector and reward powers which have a similar SNR. This is often a useful statistic to use when the detectors have similar sensitivities, however, this is not always the case. During an observing run of a gravitational wave detector, their sensitivity will vary with time due to fluctuating or new noise sources, or upgrades which increase the sensitivity. A change in the sensitivity, or noise floor, affects the SNR of a possible signal in the data, i.e. a lower noise floor results in a higher SNR. In this section the above 'line aware' statistic is modified to account for the difference in sensitivities of the detectors. The statistic then highlights areas of consistent amplitude between detectors as opposed to consistent SNR.

There are two main factors which are taken into account when determining how sensitive a detector is in a particular time interval: the PSD of detector and the duty cycle. The PSD of the detector is a measure of how sensitive the detector is at that time and the duty cycle is the fraction of time in a given interval that the detector was collecting data. A decrease in the duty cycle and an increase in the PSD will decrease the SNR and vice-versa. To search for consistent amplitude Eq.3.24 is modified by weighting each detector by its PSD and duty cycle.

The definition of SNR is taken from [55] as,

$$\rho_0^2 = \frac{h_0^2 T}{2S}(\alpha_1 A + \alpha_2 B + \alpha_3 C), \tag{3.30}$$

where ρ_0 is the optimal SNR, h_0 is the signal amplitude, T is the time of observation, S is the noise PSD and the terms in brackets include the antenna pattern of the detector. The signal with amplitude h_0 will be the same amplitude at both detectors (H and L), therefore we can relate the SNR in each detector by,

$$\rho_L^2 = \frac{\rho_H^2 S_H T_L}{S_L T_H}\frac{(\alpha_1 A_L + \alpha_2 B_L + \alpha_3 C_L)}{(\alpha_1 A_H + \alpha_2 B_H + \alpha_3 C_H)}. \tag{3.31}$$

For the majority of the analysis that follows, the SFTs are summed over one day, this is explained in greater detail in Sec. 3.7. The components in the above equation which have the form $(\alpha_1 A + \alpha_2 B + \alpha_3 C)$, account for the antenna pattern of the detector as the earth rotates. These can be approximated to be the same for the two detectors H1 and L1 as we average out the daily modulation by summing SFTs. Therefore we can simplify the above Eq. 3.31 to,

$$\rho_L^2 \approx \frac{\rho_H^2 S_H T_L}{S_L T_H} = l\rho_H^2. \tag{3.32}$$

This then gives a factor $l = S_H T_L / S_L T_H$ which relates the SNR of each detector, where S and T are values that are known for a given data-set prior to running the search.

This ratio of SNRs can be included in the integral over SNR for the signal model in Eq. 3.25 as follows,

$$p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, M_{\mathrm{S}}, I) = \int_0^\infty p(\lambda, w_{\mathrm{S}}) p(F_j^{(1)} | \nu_j, \lambda, M_{\mathrm{S}}, I) p(F_j^{(2)} | \nu_j, l\lambda, M_{\mathrm{S}}, I) d\lambda. \quad (3.33)$$

Similarly, the line model in Eq. 3.26 can be modified as,

$$\begin{aligned} p(F_j^{(1)}, F_j^{(2)} \mid \nu_j, M_{\mathrm{L}}, I) = \int_0^\infty p(\lambda, w_{\mathrm{L}}) \Big[& p(F_j^{(1)} | \nu_j, M_{\mathrm{N}}, I) p(F_j^{(2)} | \nu_j, l\lambda, M_{\mathrm{S}}, I) \\ & + p(F_j^{(1)} | \nu_j, \lambda, M_{\mathrm{S}}, I) p(F_j^{(2)} | \nu_j, M_{\mathrm{N}}, I) \Big] d\lambda. \end{aligned} \quad (3.34)$$

Fig. 3.4 shows an example of the values of the statistic described in Eq. 3.34 plotted against a range of FFT powers from each detector. This demonstrates how the statistic accounts for a difference in sensitivity between detectors by allowing the FFT power, or effectively SNR, to vary more.

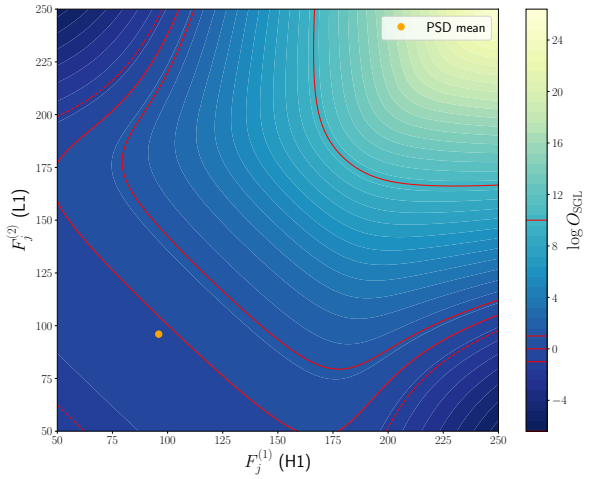
In Fig. 3.4 we show slices of the line-aware statistic with consistent amplitude for different values of l in Eq. 3.32. Figure 3.4a shows a slice where the SNR and duty cycle of the two detectors is the same, this is symmetric in the line-aware statistic as in Sec. 3.8. The asymmetry in Fig. 3.4c demonstrates how as the sensitivity of one detector (L1) increases compared to (H1) the line-aware statistic allows for lower powers in H1 with corresponding higher powers in L1.

3.10 Testing the algorithm

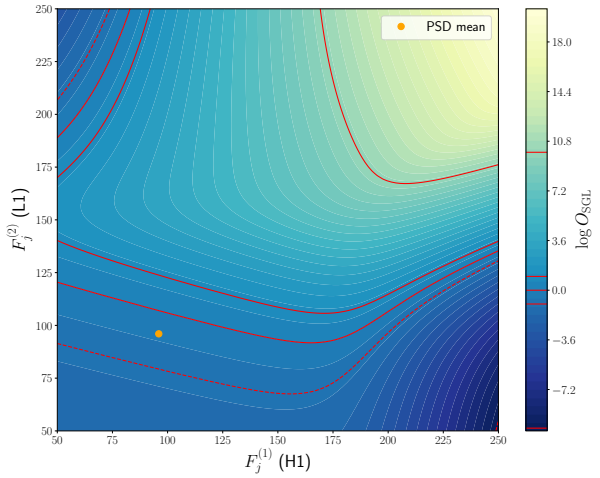
The sensitivity of the algorithm was tested by searching for artificial signals from isolated pulsars added to three types of noise-like data: continuous Gaussian noise, Gaussian noise but with periods of missing data, and real detector data (the S6 MDC [56]). The S6 MDC refers to a standardised set of simulated signals which are injected into real data, this set is also what is used for the injections into the two Gaussian noise cases. We describe each of the tests in more detail in Sec. 3.10.1,3.10.2 and 3.10.3, but several common pre-processing steps are performed before running these datasets through the Viterbi algorithm:

1. We read SFTs generated from 1800 s stretches of data in 2 Hz bands between 100 and 200 Hz. The SFTs length is chosen to ensure that any signal is likely to be contained within the width of a single frequency bin during the length of one day, rather than being split across the bin edges (see below).

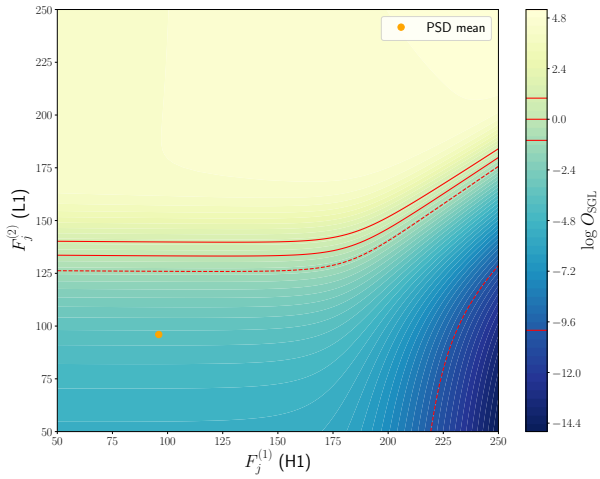
2. We estimate the noise PSD for each SFT by calculating a running median over frequency using LALSuite code `XLALSFTtoRnmed` [87], this includes a bias factor to



(a) This shows the line-aware statistic with consistent amplitude as a function of it input from each detector. This example is for an equal sensitivity and equal duty cycle for each of the detectors, i.e. $l = 1$.



(b) This shows the line-aware statistic with consistent amplitude as a function of it input from each detector. This example has L1 with a greater sensitivity and/or duty cycle than H1 where $l = 0.55$.



(c) This shows the line-aware statistic with consistent amplitude as a function of it input from each detector. This example has L1 with a greater sensitivity and/or duty cycle than H1 where $l = 0.1$.

Figure 3.4: Lookup tables using the line aware statistic for consistent amplitude as in Sec. 3.9. Each of these use the parameters $p_s(\lambda) = 4$,$p_l(\lambda) = 5$ and $p(M_L)/p(M_G) = 0.03$. The PSD mean is the expected mean of a χ^2 distribution with 48 degrees of freedom, i.e. the expected power from out summed spectrograms F_j.

convert this to the mean and has a width of 100 bins. We then normalise the SFT by dividing it by its running median, giving the noise-like parts of the spectrum a mean power of approximately one.

3. The SFTs are then summed over one day, as described in Sec. 3.7. The signal parameters are chosen so that within the frequencies of the search, the signal will not fall in more than two frequency bins over this period.

   The differential Doppler shift of a signal seen at two detector sites due to the Earth's rotation $\Delta f_{\mathrm{rot}}^{(1,2)}$ is simply

   $$\Delta f_{\mathrm{rot}}^{(1,2)} = \frac{(\boldsymbol{v}^{(1)} - \boldsymbol{v}^{(2)}) \cdot \hat{\boldsymbol{s}}}{c} f_0, \tag{3.35}$$

   where $\boldsymbol{v}^{(1,2)}$ is the velocity of detector $1, 2$ in an inertial reference frame, f_0 is the instantaneous signal frequency in the frame, $\hat{\boldsymbol{s}}$ is the unit vector in the direction of the source and c is the speed of light. The maximum difference in frequency seen by the two LIGO detectors is

   $$\Delta f_{\mathrm{rot}} \approx 6.5 \times 10^{-7} f_0, \tag{3.36}$$

   so the frequency measured from a source in the equatorial plane with $f_0 = 200\,\mathrm{Hz}$ will differ by up to $1.3 \times 10^{-4}\,\mathrm{Hz}$ in the two detectors. This is ~ 4 times smaller than the frequency bin width of 1800 s SFTs ($5.6 \times 10^{-4}\,\mathrm{Hz}$), so signals at frequencies lower than this are likely to appear in the same frequency bin in the two detectors. Therefore, whilst at higher frequencies we still allow the signal to be in different frequency bins between the detectors, in the following searches, we do not allow this.

4. The data is then split into 0.1 Hz sub-bands which are overlapping by 0.05 Hz. These were chosen to ensure that signals are contained within a sub-band over the year. On these timescales the important contributions to the frequency evolution are the spin-down rate of the pulsar and the Doppler shift due to the earth orbit. To investigate the doppler shift, we can look at a signal at 200 Hz, using Eq. 3.35 we can calculate the maximum shift in frequency due to the earths orbit as,

   $$\Delta f_{\mathrm{orbit}} = \frac{2\pi R_o}{T_o} \frac{1}{c} f_0 \approx 9.9 \times 10^{-5} f_0, \tag{3.37}$$

   where T_{o} and R_{o} are the earth orbit time and radius. This gives a maximum doppler shift of 0.019 Hz, this is a $\sim 1/5$ of the width of a sub-band, therefore, is more likely to be totally contained within a sub-band than crossing over the edge. To account for the cases where the signal frequency crosses over the edge of a sub-band, the sub-bands overlap by 0.05 Hz so that the majority of the signals should be completely

contained within at least one of the sub-bands. To investigate the spin-down of the pulsar, we look at the length of data, $T = 4.05 \times 10^7$ s and we choose a sub-band width of 0.1 Hz. For a signal to drift over the width of a whole sub-band we would need f-dot of,

$$\frac{df}{dt} > \left| \frac{-0.1}{4.05 \times 10^7} \right| = 2.4 \times 10^{-9} \text{Hz/s}. \tag{3.38}$$

The majority of the injections that follow satisfy this condition, signals which are greater than this, and therefore drift over multiple bands, are vetoed from the search.

5. The two detector Viterbi algorithm is then run using the line aware statistic (see Sec. 3.8). There are 4 parameters which we optimise in this search. The transition probabilities, where we have one parameter τ which is the ratio of the probability of going straight to the probability of going either up or down. Due to the averaging procedure, the signals received at each detector are forced to follow a common track which is equal to the 'imaginary' detectors track. The other three parameters, w_S, w_L and $\mathrm{p(M_L)/p(M_N)}$, are described in Sec. 3.8.

6. The algorithm then returns the most probable track though the data, and the value $\propto$ the log-odds in the final time step, i.e., the maximum final value, $\max_j(V_{N,j})$, in Eq. 3.29, which is then our detection statistic.

As an example of what the algorithm returns, Fig. 3.5 shows the tracks in the two detectors, H1 and L1. This also shows the log-odds ratio of ending in any frequency bin, i.e., all the elements in Eq. 3.29. In this figure, each time segment of the odds ratios have been normalised such that the sum of the odds ratios is 1.

In the following tests there are two main quantities which we use to determine the sensitivity. These are sensitivity depth $\mathcal{D}$ and the optimal SNR ρ. The sensitivity depth, $\mathcal{D}$, is defined in [88] as,

$$\mathcal{D}(f) = \frac{\sqrt{S_h(f)}}{h_0}, \tag{3.39}$$

where $S_h(f)$ is the single-sided noise PSD and h_0 is the GW amplitude. The optimal SNR is defined as,

$$\rho^2 = \sum_X 4\Re \int_0^\infty \frac{\tilde{h}^X(f)\tilde{h}^{X*}(f)}{S^X(f)} df, \tag{3.40}$$

where X indexes the detectors and $\tilde{h}(f)$ is the Fourier transform of the time series of the signal $h(t)$. This expression is defined in [55] for a double-sided PSD and we have defined it for the more common single-sided case.

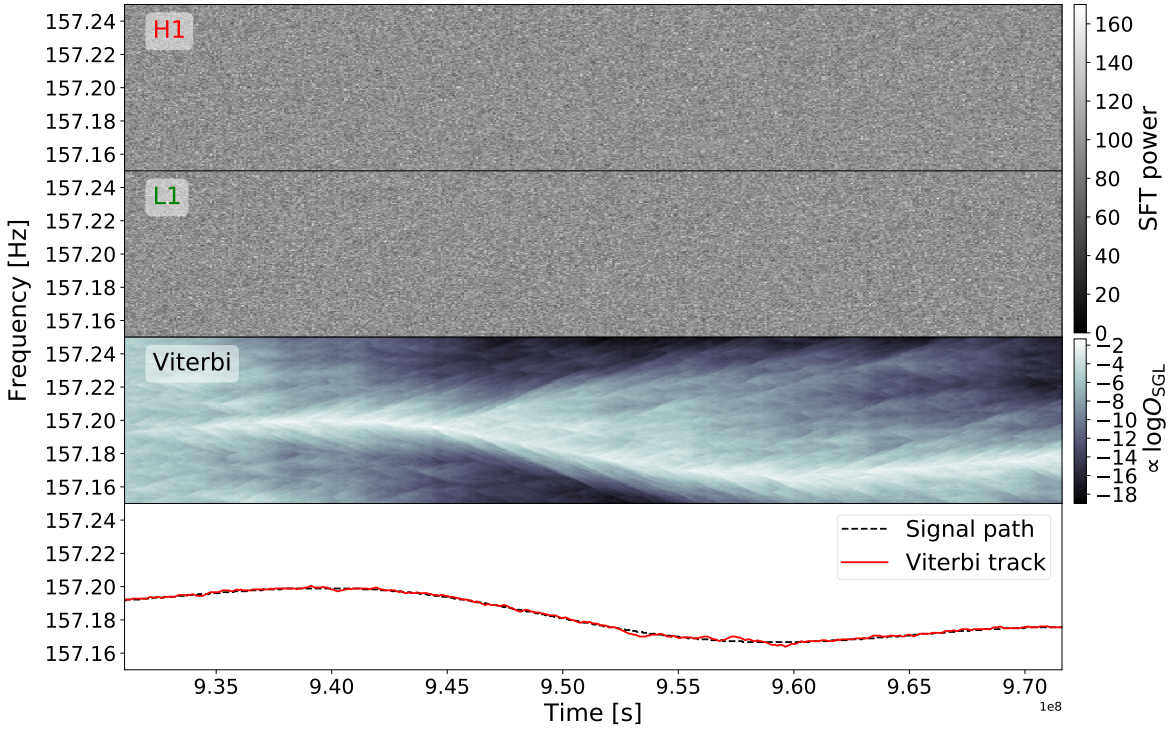


Figure 3.5: The results that the SOAP algorithm returns from an injection with an optimal SNR of 90, i.e., the SNR in H1 is 64 and the SNR in L1 is 62. The signal is injected into Gaussian noise, where the 1800 s SFTs have been summed over 1 day. The top panel shows a simulation of summed SFTs from H1, the second panel shows the same for L1, the third panel shows the values proportional to the log-odds ratios in Eq. 3.29. The log-odds have been normalised such that the sum of all the odds ratios in every time bin are equal to 1. The bottom panel shows the injected signal track (black dotted) and the track found in the 'imaginary' detector by the two-detector SOAP search with the line-aware statistic (red), both of these tracks are at the geo-centre. In this case the root median square (RMS) of the difference between the Viterbi track and injected signal track was ∼1 bin, where 1 bin is 0.00056 Hz wide.

3.10.1 S6 injections into gapless Gaussian noise

The first test involves injecting signals into Gaussian noise. The power spectrum of a Gaussian noise time-series follows a χ^2 distribution with two degrees of freedom, therefore, as we search through the power spectrum, we generate spectrograms which follow a χ^2 distribution. These spectrograms are 0.1 Hz wide and are set at 0.05 Hz intervals between 100 Hz and 200 Hz. The bins are 1./1800 Hz wide and 1800s long, where the total length of data is the same as S6, i.e., ~ 1.3 years. We then generate the signals, where the pulsars parameters are fixed to the same values as the injections in the S6 MDC in this band, these values are outlined in [56].

The values of f_0 for the injections were not always centered in a sub-band, therefore a number of sub-bands contained only part of the injected signal. These sub-bands were ignored as they contaminated the signal statistics and only the sub-band which contained the whole signal was accepted. This reduced the number of sub-bands from 2000 to 1762 with the removal of 238 sub-bands containing only part of a signal. This set also includes signals that drift across multiple sub-bands due to their high spin-down rate. Only two signals were removed due to their spin-down values, which were $> 5 \times 10^{-9}$ Hz/s, these were the two hardware injections in the 100-200 Hz band.

For each injection the SOAP algorithm returns the detection statistic described in Sec. 3.8 and 3.10. We calculate a false alarm rate, which is the fraction of bands that have no injection that do exceed a given threshold. This is set to 1% and is used as a detection threshold. We then take all of the bands and if they pass the threshold we set them as detected, i.e., 1, and if they do not they are set as not detected, i.e., 0. This then leaves us with a set of binomial data, where the efficiency curves later in the paper are sigmoids which have been fitted to this. The sigmoid follows,

$$s(x; x_0, k) = \frac{1}{1 - \exp\left(-k(x - x_0)\right)}. \tag{3.41}$$

The fit is done by sampling the posterior, i.e.,

$$p(x_0, k \mid b) \propto p(x_0, k) p(x \mid x_0, k), \tag{3.42}$$

where $p(x_0, k)$ is the prior and we set to a flat prior and $p(x \mid x_0, k)$ is the likelihood function which is defined by,

$$p(\bar{x} \mid x_0, k) = \prod_{j=0} \frac{n!}{k!(n-k)!} s(x_j \mid x_0, k)^k \\ (1 - s(x_j \mid x_0, k))^{n-k}. \tag{3.43}$$

To plot the efficiency curves and lower and upper error bounds, we sample Eq. 3.42 using

Table 3.1: Table shows the ranges of the search parameters and their optimised values for injections into gapless Gaussian noise, Gaussian noise with gaps and the S6 MDC. For gapless Gaussian noise and Gaussian noise with gaps, there are 10 parameter values spaced linearly between the limits. For the S6 MDC the parameters, τ, w_{L} and w_{S} were distributed in log space between the limits and $p(M_L)/p(M_N)$ is distributed uniformly.

	τ	w_S	w_L	$p(M_L)/p(M_N)$
		Gapless Gaussian		
limits	[1.0,1.3]	[0.1,5.0]	None	0.0
optimised	1.1	2.06	None	0.0
		Gaussian with gaps		
limits	[1.0,1.3]	[0.1,5.0]	None	0.0
optimised	1.1	2.06	None	0.0
		S6 MDC		
limits	[1.0,1.1]	[0.1,5.0]	[0.1,6.0]	[0.0,1.0]
optimised	1.00000001	4.0	5.0	0.0387

MCMC and then take the mean and the 5th and 95th percentiles respectively for each point in SNR or depth and plot these. Figure 3.6a and 3.6c then show the efficiency curves for the analyses plotted against the signals optimal SNR and depth respectively. The parameters of the search and their optimised values are shown in Tab. 3.1. Where we set the prior on the line model to 0 as this part is irrelevant to this search due to the lack of lines in the data.

From this we can determine that in Gaussian noise without gaps, the Viterbi algorithm can detect to an SNR of ~ 60 and a depth of $\sim 33\,\mathrm{Hz}^{-1/2}$ with 95% efficiency at a 1% false alarm.

Fig. 3.6b and 3.6d, show the RMS of the difference between the injected signal track and the track found by Viterbi for SNR and sensitivity depth respectively. This shows that at SNR of 60, where we are detecting signals with a 95% efficiency, the signals have a mean RMS of ~ 2 frequency bins. Here one bin width is 0.00056 Hz therefore, we have and RMS of ~ 0.0012 Hz.

3.10.2 S6 injections into Gaussian noise with gaps

In the second test, we attempt to more closely mirror the S6 MDC [56] in two stages. The first uses the same injection method as Sec. 3.10.1 however, removes the SFTs where there are gaps in S6. The second uses the same injection method again including gaps, however, uses a different value for the noise floor for each SFT, this is calculated for each band and

SFT from S6 data.

Both detectors in S6 had a duty cycle of ∼50% [89], which means that there are sections of time where there is no data in either one or both detectors. In the sections where one detector is observing but the other is not, the multi detector statistic will not behave correctly as it only has access to data from a single detector. In these sections we switch from using the multi-detector statistic to the single-detector statistic using the same parameters, these are both defined in defined in Sec. 3.8.

The process of removing sub-bands and generating efficiency curves is the same as in Sec. 3.10.1.

We set a 1% false alarm rate and generate an efficiency curve for SNR and depth in Fig. 3.6a and Fig. 3.6c respectively. From these efficiency plots we can see to an SNR of ~ 72 or a depth of $\sim 13\,\mathrm{Hz}^{-1/2}$ at a 95% confidence with a false alarm of 1%.

The parameters of the search which were optimised and their optimised values are shown in Tab. 3.1.

In Fig. 3.6b and 3.6d show the RMS of the difference between the injected signal track and the track found by Viterbi for SNR and sensitivity depth respectively. This shows that at SNR of 72, where we are detecting signals with a 95% efficiency, the signals have a mean RMS of ~ 10 frequency bins (0.0056 Hz).

3.10.3 Tests on the S6 MDC

For a more direct comparison to other CW searches and to see how the algorithm performs with real data, we test the two detector SOAP algorithm using the S6 MDC. We focus this search on the 100-200 Hz band, there are two main reasons for this, one being that this is LIGOs most sensitive band and the other is that for much higher frequencies the signal will drift over larger frequency ranges, therefore, our SFT length will have to be changed. Here the 1800 s SFTs are split as in Sec. 3.10, whereafter normalisation, the data is split into 0.1 Hz wide sub-bands overlapping by 0.05 Hz.

The two detector SOAP algorithm using the line-aware statistic in Sec. 3.8 is then run on each sub-band under the assumption that the detectors have the same sensitivity. For this search we have four parameters which we optimise, the ranges and optimised values are shown in Tab. 3.1.

As in Sec. 3.10.2, only the sub-bands which contained the entire frequency evolution of the signal were selected. Out of the 2000 sub-bands, 238 were removed due the sub-band only containing part of the signals frequency evolution. The main difference between the analysis for Gaussian noise and real data is that the real data is contaminated with instrumental lines. This means that whilst the techniques described in Sec. 3.8 reduce the number of contaminated bands with a high statistic value, there are still instrumental lines which are coincident between the detectors and which could not be removed with these

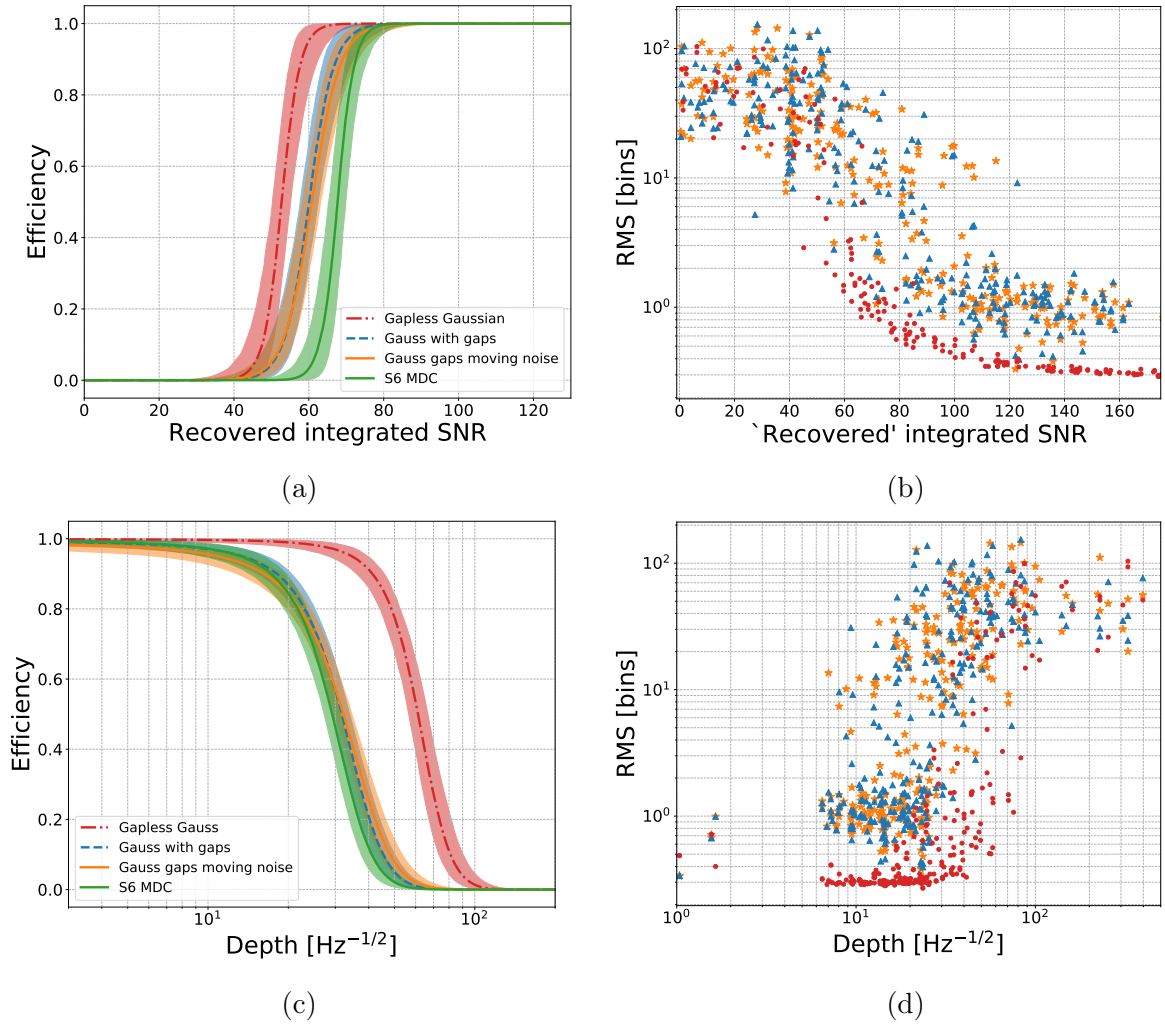


Figure 3.6: Panels 3.6a and 3.6c show the detection efficiency as a function of SNR and depth respectively. Here SNR is the the integrated SNR which we would expect to recover from the available data. The four curves refer to injections into gapless Gaussian noise (red), Gaussian noise with gaps in data, where the noise floor is either fixed (blue-dashed) or it is moving with time (orange) in the same way as the S6 MDC and injections into real data i.e., the S6 MDC. In the gapless Gaussian noise case, the recovered integrated SNR refers to the SNR the injection would have if it had the same amount of data as in the cases with gaps. The curves are made by fitting a sigmoid Eq. 3.39 to binomial detection data with a 1% false alarm rate, as explained in Sec. 3.10.1, the error bounds are the 5% and 95% intervals. At 95% efficiency and a 1% false alarm rate, this shows we can detect to an SNR of ~ 60 and a sensitivity depth of $\sim 34\,\mathrm{Hz}^{-1/2}$ for gapless Gaussian noise and an SNR of ~ 69 and 72 and a sensitivity depth of $\sim 13\,\mathrm{Hz}^{-1/2}$ and $\sim 10\,\mathrm{Hz}^{-1/2}$ for the Gaussian with gaps case with fixed noise floor and moving noise floor respectively. For the S6 MDC we can detect an SNR of ~ 74 and a sensitivity depth of $\sim 13\,\mathrm{Hz}^{-1/2}$. Panels 3.6b and 3.6d show the RMS of the difference between the injected signal track and the track found by SOAP as a function of SNR and sensitivity depth respectively. This is shown in units of bins where each bin is 0.00056 Hz wide.

techniques. Within the data there are large number of lines at integer Hertz, which are seen in coincidence between the two detectors, these are thought to originate from digital electronics [90]. Therefore the frequency bins ± 1 bin of each integer frequency in Hertz were removed and filled with the expectation value of the noise. To remove instrumental effects at other frequencies, the sub-bands which gave values of our statistic above a chosen threshold were investigated by eye. In this case 344 sub-bands were investigated, and any which were contaminated were vetoed. From these 344 sub-bands, 193 were removed from the analysis. The predominant feature in the bands which were removed were broad spectral features which lasted the whole run. Therefore, out of the 2000 sub-bands which are searched over, a total number of 431 sub-bands were removed.

The process to calculate the efficiency curves is the same as in Sec. 3.10.2 and 3.10.1.

Fig. 3.6c and Fig. 3.6a show the efficiency curves for SNR and depth respectively. These show that we can detect and SNR of ~ 74 and a sensitivity depth of $\sim 13\,\mathrm{Hz}^{-1/2}$ with an efficiency of 95% at a false alarm of 1%. These results can then be compared to other searches in the S6 MDC comparison paper [56]. Whilst we only search in the 100 - 200 Hz range, the closest comparison in [56] is the test in the 40 - 500 Hz range, such as in Fig. 4 in [56]. Here our algorithm sits roughly in the middle of all other searches in terms of sensitivity.

3.11 Optimisation of Line-aware statistic.

For the above searches we used optimised versions on the line aware statistic, however, we have yet to explain how this was optimised. The aim is to find the best parameters for any given search; the four parameters are τ, w_{S}, w_{L} and $p(M_{\mathrm{L}})/p(M_{\mathrm{G}})$. We find the optimum values empirically by running the entire search for each parameter value that needs to be tested. This is possible as the search is relatively fast, this will be explained in Sec. 3.14. The line aware statistic is time consuming to calculate, therefore, to reduce the computational time, it is pre-calculated and placed into lookup tables such that it is calculated once and called many times. These lookup tables were calculated for values of the SFT power summed over one day F. The summed SFT power is in the range 1 to 400 in each of the detectors as shown in Fig. 3.3. The four parameters ranges were chosen based on what we expect to see. For example we expect a signal to have a small SNR therefore, the range of w_{S} is between 0.1-10 in the S6 case. We expect the instrumental lines to have a larger SNR therefore, w_{L} ranges between 0.1-20 in the S6 case. Each of the parameter ranges is shown in Tab. 3.2.

We can then use a measure of the sensitivity of that search and pick the set of parameters lookup table which gives the highest sensitivity. We measure the sensitivity by taking the value of SNR which is at 80% efficiency at 1% false alarm.

Table 3.2: Table shows the ranges of the search parameters and their optimised values for injections into Gaussian noise and the S6 MDC. For Gaussian noise there are 30 parameter values spaced linearly between the limits. For the S6 MDC the transition matrix parameters, τ, had three values space between the limits. This is because the search is relatively insensitive to this parameter. The parameters w_{L}, w_{S} and $p(M_{\mathrm{L}})/p(M_{\mathrm{G}})$ had 10 parameters distributed in linearly between the limits.

	τ	w_S	w_L	$p(M_{\mathrm{L}})/p(M_{\mathrm{G}})$
Gaussian noise				
limits	[1.0,1.1]	[0.1,7.0]	None	0.0
S6 MDC				
limits	[1.0,1.1]	[0.1,10.0]	[0.1,20.0]	[0.0,0.3]

3.11.1 Gaussian noise simulations

For injections into Gaussian noise, we know that there are no instrumental lines, therefore, we do not need to optimise over the 'lines' part of the statistic and can set the parameter $p(M_{\mathrm{L}})/p(M_{\mathrm{G}})$ to zero which renders the parameter $p_L(\lambda)$ redundant. This then reduces the complexity of the problem by leaving us with only two parameters to optimise over, τ and $p_S(\lambda)$. Whilst this optimisation was partially done in Sec. 3.10, with the result in Tab. 3.1, this is repeated more completely here. The parameters were optimised in the range shown in Tab. 3.2.

For each point in Fig. 3.7, the entire SOAP search was run using the corresponding parameters as input. Efficiency curves are then generated for each of these runs and the values for the SNR at 80% efficiency are recorded. Fig. ur3.7 then shows the 80% efficiency for each parameter value. There appears not to be any single value which gives an optimum, however, the dark stripe in Fig. 3.7 running from the bottom left to the red cross is the combinations of parameters which give the best result. The point where the red lines cross is the parameters used in previous searched in Sec. 3.10.1 and 3.10.2. This falls on the line where the algorithm performs best. Venturing far from this 'optimum' line does not change the results a great deal as the SNR does not change much. The search is then not particularly sensitive to choice of parameters in Gaussian noise.

3.11.2 S6 MDC injections

As the S6 MDC data-set is real detector data, there are many examples of instrumental lines. This is where we expect the line-aware statistic to have the greatest effect in rejecting lines. Here all four parameters are optimised over in the ranges described in Tab. 3.1. This greatly increases the number of lookup tables which need to be generated and therefore the number of times the search needs to be run. Figure 3.8 shows the projections in parameter

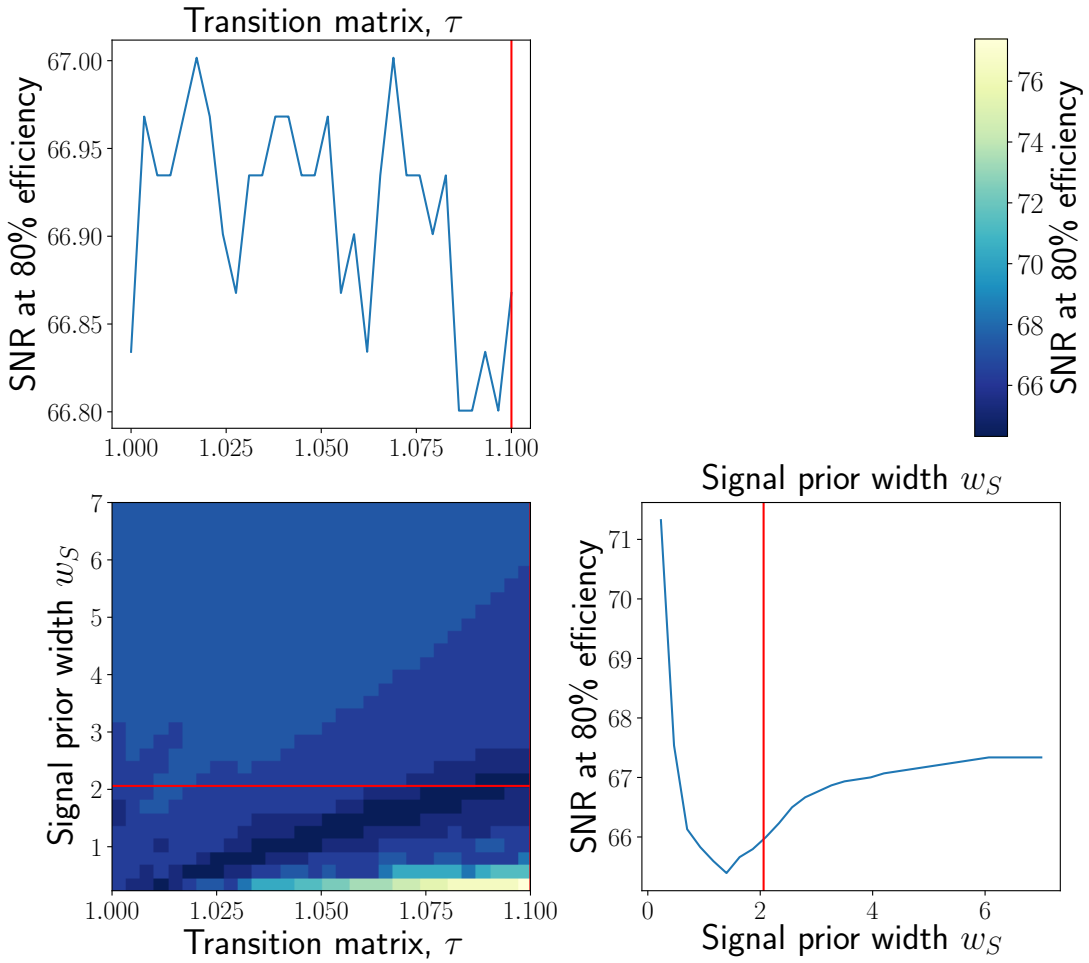


Figure 3.7: In Gaussian noise the transition matrix parameter τ and the width of the prior on the signal case w_{S} were optimised. The key part to remember when reading this plot is that the lower the value of SNR the better the search has performed. Therefore darker blue areas are when the search performed better. This map shows that there is a line in parameter space where the search performed best. Also in Gaussian noise, the search is not that sensitive to the choice of parameter. The red lines on here shows the parameters used in the searches in this section.

space for each of the parameters where the values are the SNRs at 80% efficiency. These projections are made by taking the mean **CHRIS: if you're taking the mean then these aren't slices, they're projections or marginalisations. Why are you taking the mena anyway?** across the parameters not shown in the figure. The range chosen for the transition matrix parameter τ is small with only 3 point, this is because the parameter is expected to have little impact on the result. Fig .3.8 demonstrates that changing this parameter does not have much effect on the final sensitivity. The parameter w_{S} appears to have a clear location around a value of 3 where the sensitivity is at a minimum. This is not far from the value of the parameter which is used for all other searches in this thesis (the red line). The parameters w_{L} and $p(M_{\mathrm{L}})/p(M_{\mathrm{G}})$ show that the sensitivity drops substantially as the values increase. In these examples the optimum value may be out of

the range chosen here. The sensitivity of this search could be improved slightly by running with a wider parameter space in w_{L} and $p(M_{\mathrm{L}})/p(M_{\mathrm{G}})$. The optimisation procedure has a much larger computational cost than the search itself. Therefore, as one of the searches strengths is its speed, and that the sensitivity change is relatively insensitive to the choice of parameters, see Fig. 3.11.2, the search on a wider parameter space was not run. The results in Sec. 3.10 show that the current values of the parameters (red lines in Fig. 3.8) are sufficiently sensitive when compared to other searches. To choose optimum parameters for this search, the minimum point for each parameter was chosen. This is the minimum in each of the diagonal plots in Fig .3.8. These values are,

$$\begin{aligned} \tau &= 1.0 \\ w_S &= 3.0 \\ w_L &= 20.0 \\ p(M_L)/p(M_G) &= 0.2879. \end{aligned} \tag{3.44}$$

CHRIS: We've discussed that the obvious thing to do is to take the global minimum value. I don't think you are doing this. You are taking the marginalised values. You do not explain why you do this.

To visualise how these better optimised parameters are to the values used in Sec. 3.10 we can compare how they perform on a simulated signal in a time-frequency spectrogram. In Fig. 3.9 one of the detectors contains a narrow instrumental line and both detectors contain a CW signal. In the Gaussian noise optimised case the search is looking for high power in both detectors which the strong instrumental line satisfied when the astrophysical signal is weak. The S6 optimised statistic looks for more consistent SNR in each of the detectors. The lines labelled Line-aware and Old Line-aware refer to parameters in Eq. 3.44 and Tab. 3.1 respectively. This demonstrates how the 'line aware' statistic improves the robustness of the algorithm against non astrophysical signals when correct parameters are chosen. Figure 3.9 shows the results for a specific example, however, Sec. 3.11.2 demonstrates how the search is not particularly sensitive to choice of parameters when tested on many examples.

Comparison of sensitivity

To compare the different parameters of the line aware statistic, three parameter sets were used. The optimised values for the Gaussian noise case and the S6 MDC case above are used alongside the values optimised in Tab. 3.1. These three parameter sets are run on the S6 MDC simulations between 40 and 500 Hz to compare their sensitivity. The process of running on the S6 MDC and generating the efficiency curves is the same as in Sec. 3.10. Fig. 3.10 shows the efficiency curves at a 10% false alarm rate. This, along with Fig. 3.7

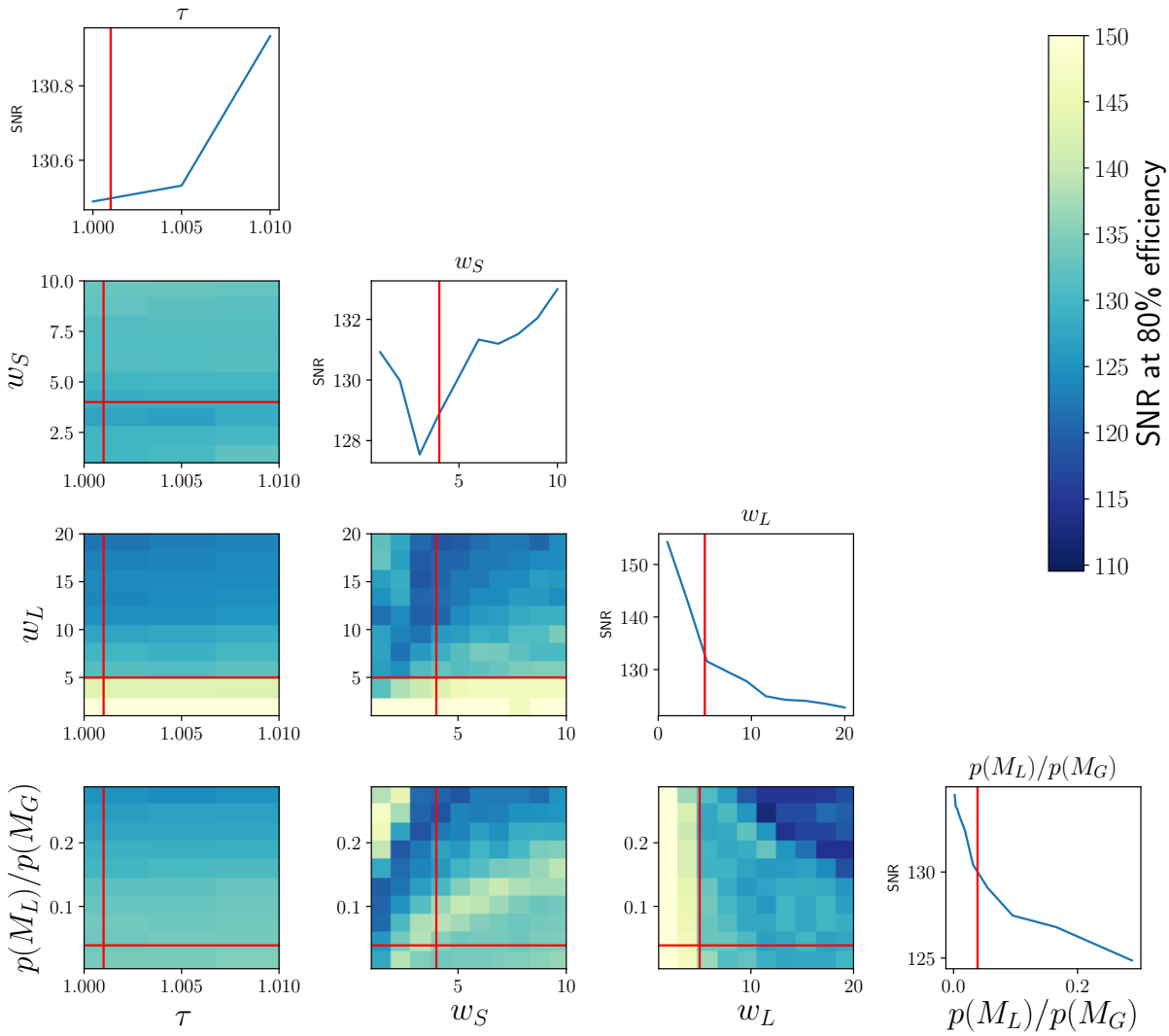


Figure 3.8: When using real S6 data, all four parameters of the search were optimised over on simulations in real data. The plot above shows the SNR at 80% efficiency for each of the parameters where the ranges are in Tab. 3.2. Lower values of SNR mean the search is performing better. The red lines show the parameters used in the searches in this section and the sections that follow. Whilst this does not seem optimal, the search does not underperform much using the current choice of parameters (red).

and 3.8 shows that the search is relatively insensitive to the choice of parameters.

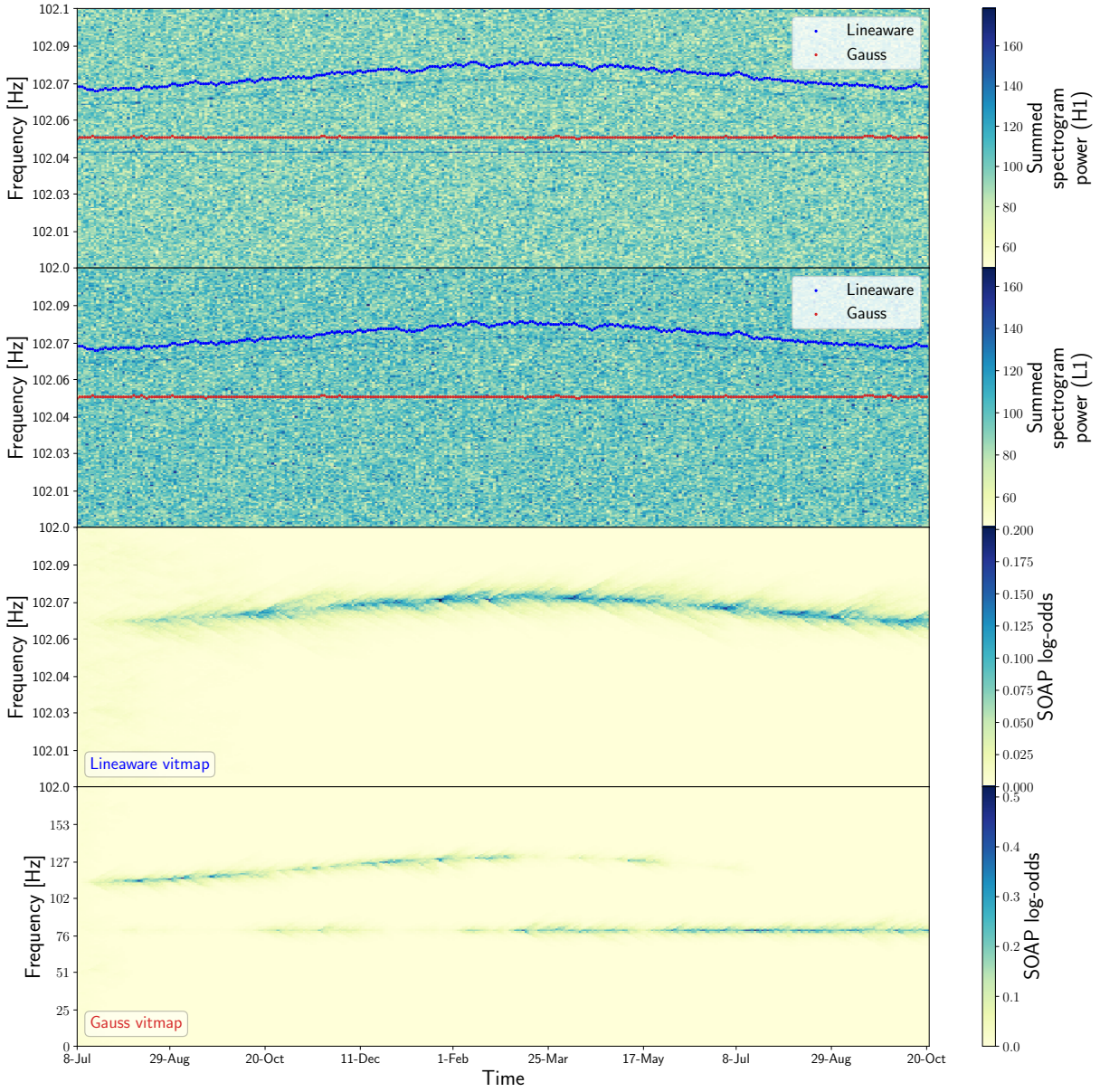


Figure 3.9: The spectrograms of H1 and L1 which contain a CW simulation are shown in the first and second panel respectively. The top panel (H1) also contains a narrow spectral line at $\sim$ 102.5 Hz. The Viterbi tracks from the SOAP search with Line-aware statistic optimised for S6 and Gaussian noise are shown in blue and red respectively. These are shifted up by 10 frequency bins to allow the underlying feature to be identified in the spectrogram. The third and fourth panel show the Viterbi map for the statistic values optimised in the S6 MDC and Gaussian noise respectively. The Viterbi map for the S6 MDC statistic shows areas of high log-odds around the signal track. The Viterbi map for the Gaussian noise optimised statistic shows high log-odds around the signal and the instrumental line and identified the track along the instrumental line to be the most significant.

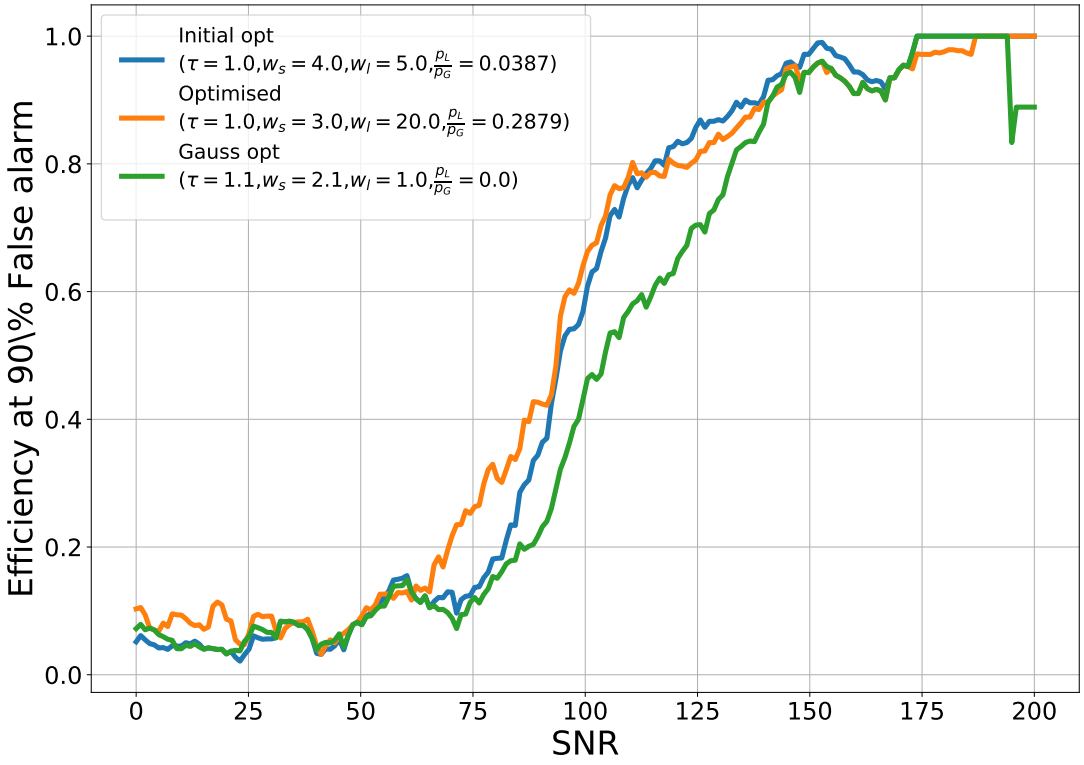


Figure 3.10: The sensitivity can be compared for three sets of parameters of the line aware statistic. These are the sets optimised for Gaussian noise and the S6 MDC in Sec. 3.11 above (Optimised and Gauss opt) and the values used in Tab. 3.1 (Initial opt).

3.12 Sensitivity with frequency

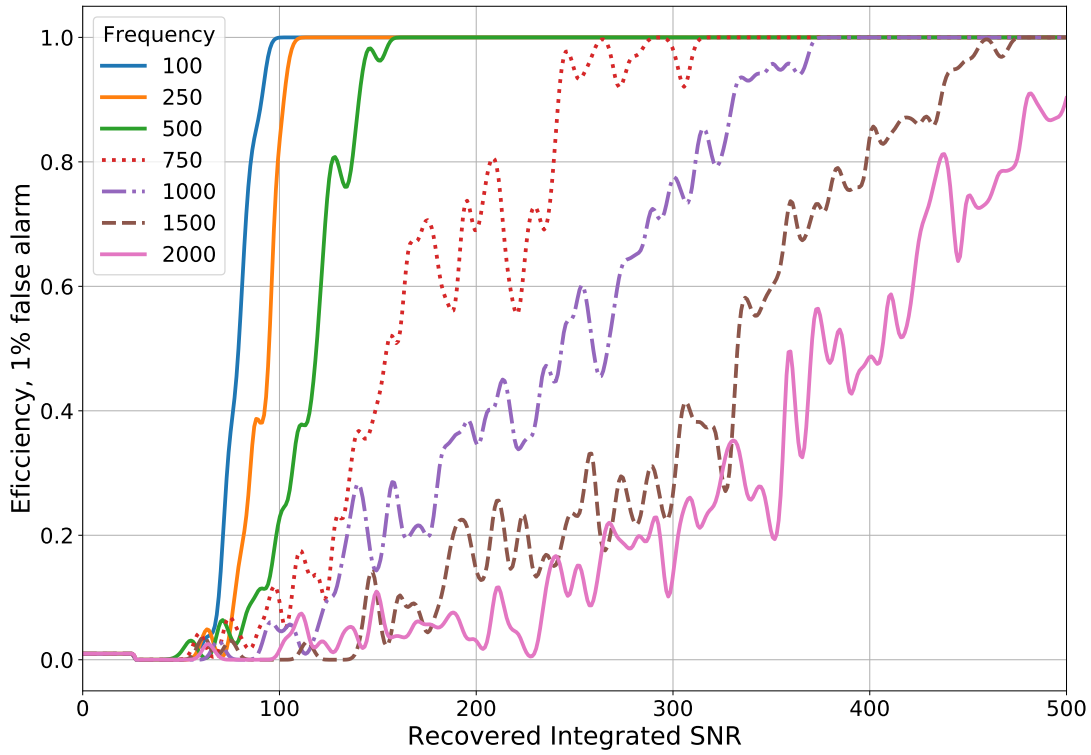
The tests in Sec. 3.10 on Gaussian noise and S6 data was conducted in the range from 100-200 Hz. This was chosen to be within the most sensitive band of LIGO as this is where a signal is most likely to be discovered. However, signals can appear at much higher frequencies also. Therefore, it is important to see how the sensitivity of the search varies with the frequency.

For this test we simulated CW signals in Gaussian noise with no gaps in data. The injections used the same source parameters as in the S6 MDC [56] and the tests above. This has the exception that the integrated recovered SNR of the signal is sampled uniformly between 50 and 500. These injections were then made at frequencies of 100, 250, 500, 750, 1000, 1500 and 2000 Hz, where the band width is 2 Hz. i.e. the simulations were in frequency bands 100-102 Hz, 250-252 Hz etc. The setup of the search was the same as in the above sections. Here each sub-band is 0.1 Hz wide, and the parameters of the SOAP search were as in Tab. 3.1.

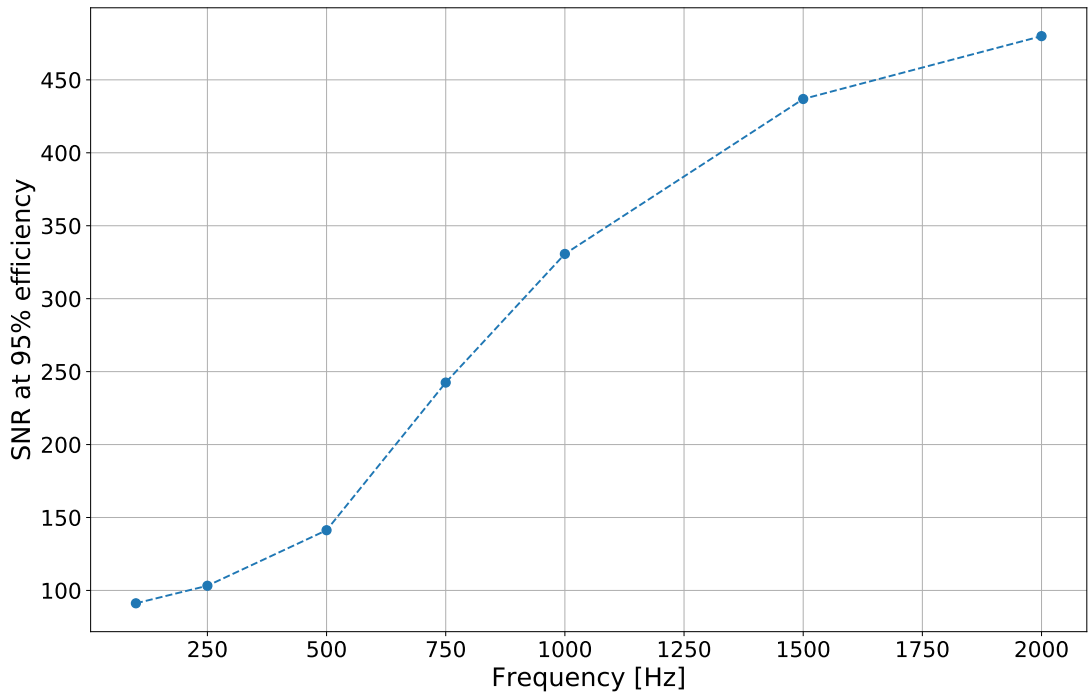
Figure 3.11 shows the resulting efficiency curves from each of these tests. This is for a 1% false alarm rate, which means that 1% of sub-bands which contained no injection crossed the detection threshold. This plot shows how the sensitivity of the search drops as the frequency increases. This is perhaps unfair to the algorithm as we used the setup of the search which has been optimised for the range 100-200 Hz. Optimising the search means choosing the parameters of SOAP, the key parameter which will affect this is the transition matrix. As the simulated signals frequency is increased, the scale of the Doppler modulation will also increase. This means that at higher frequencies the signal is more likely to jump more than a single frequency bin. The current setup of the search does not allow this size of jump and therefore, would struggle to identify this type of track. The other main factor which will decrease this sensitivity is the sub-band width of 0.1 Hz. As the signal frequency increases the scale of the Doppler modulation will increase as shown in Eq. 3.37. For example at 1000 Hz, the Doppler shift is $\sim$ 0.1 Hz, the signal is then more likely to not be fully contained within a frequency band. Therefore, the search can not accumulate all of the injected SNR The search in its current state however, does lose sensitivity as the signal frequency increases.

3.13 Searching for non-CW sources

Whilst SOAP was designed to search for sources of CWs, when set up correctly, it can be applied to searching for other signal types. This is because the search is essentially un-modelled, and in its simplest form looks for tracks of high power in a time-frequency spectrogram. Therefore, if the signal can be represented in a spectrogram, then it can be searched for using SOAP. Ideally the signal will also live on a single track and span the



(a)



(b)

Figure 3.11: The sensitivity of the SOAP search in this configuration decreases as the frequency of the pulsar increases. This setup of data for the search however, was optimised for the 100-200 frequency band and can be changed for different frequencies. 3.11a shows the efficiency curves with 1% false alarm rate for each frequency. 3.11b shows the values from the efficiency curves at 90% efficiency.

time length of the spectrogram.

CBC signals cover a wide frequency range and are very short signals in LIGO detectors compared to CWs. The longest of these are BNS signals which are generally detectable for $\infty\prime$ s. In previous SOAP searches the default length of SFT has been 1800 s, however ,this is not suitable when searching for CBC signals. We then need a shorter time base for each of the SFTs. In the following examples the SFTs are overlapping in time, this allows us to achieve the desired time and frequency resolution. However, this means that the SFTs are no longer independent and out Bayesian formalism is not technically correct.

Fig. 3.12 shows an example of the two detector SOAP search running on data +2s and -5 s of the GW170817 merger time [7]. The spectrograms show the SFTs power spectra divided by their running median. Each SFT is 0.2 s long and are overlapping by 90% (0.189 s), this gives a frequency resolution of 5 Hz. Figure 3.12 shows how SOAP identifies a track which follows that of the BNS and the Viterbi map shows a clear excess of log probability where the signal lies. Whilst this may not be an optimal setup for this particular search, it demonstrates that SOAP can identify a BNS signal within the data. The example in Fig. 3.12 show the result between 20 and 520 Hz, however the SOAP search returns the same track for a wider band-width up to 2000 Hz. Some work has been done on a variant of the Viterbi algorithm to search for a post-merger remnant of the GW170817 merger [91].

Searching for BBH systems becomes more difficult as they are in the LIGO band for a short period of time ($< 1s$). To see the evolution of the signal in a spectrogram, the time resolution required means that the frequency bins are too large to see the signals evolution. There are other time-frequency representations which we would use such as the Q transform to allow us to search for these signals. However, the bins here are not independent, therefore the Bayesian formalism described in this chapter is no longer technically true. The SOAP search can still be run on the Q transform however, this is shown for GW150914 [4] in Fig. 3.13.

In both Fig. 3.13 and 3.12, the signal does not last for the entire length of the time-frequency band,

This section serves as a demonstration that it is possible to use SOAP for other types of search, it has more flexibility than the majority of examples in this thesis show. This is an area of work which would benefit from further investigation.

CHRIS: You could expand upon this subsection. One thing to address is the issue that the signal does not last for the entire length of the window. What could you say about that? You could make the argument that SOAP makes the TF track far more visible that a spectrogram and other algoritjhms could be developed to search the vitmaps?

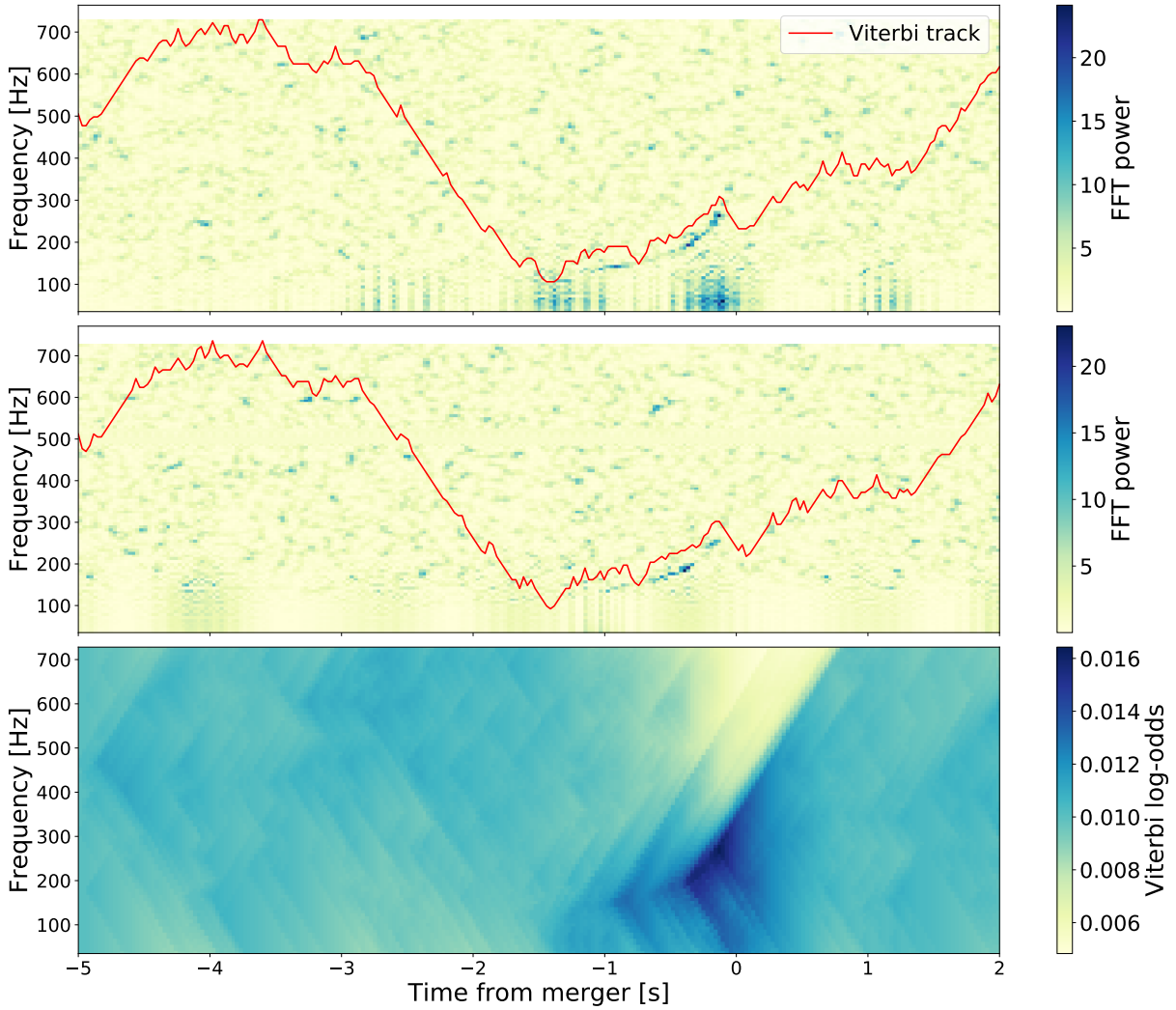


Figure 3.12: The SOAP search was run on a spectrogram of LIGO data +2 and -5 seconds around the merger of GW170817 [7]. The top two panels show the spectrograms from LIGOs H1 and L1 detectors respectively. Each FFT in the spectrograms are 0.2 s long and are overlapping by 0.189 s (95%). The red track shows the output Viterbi track shifted up by 50 Hz so that the signal can bee seen in the spectrogram. The final panel shows the Viterbi map output. The BNS signal here is GW170817 [7], where the coalescence is at a time of 0. The Viterbi map shows a area of higher log-odds along the path of the BNS signal.

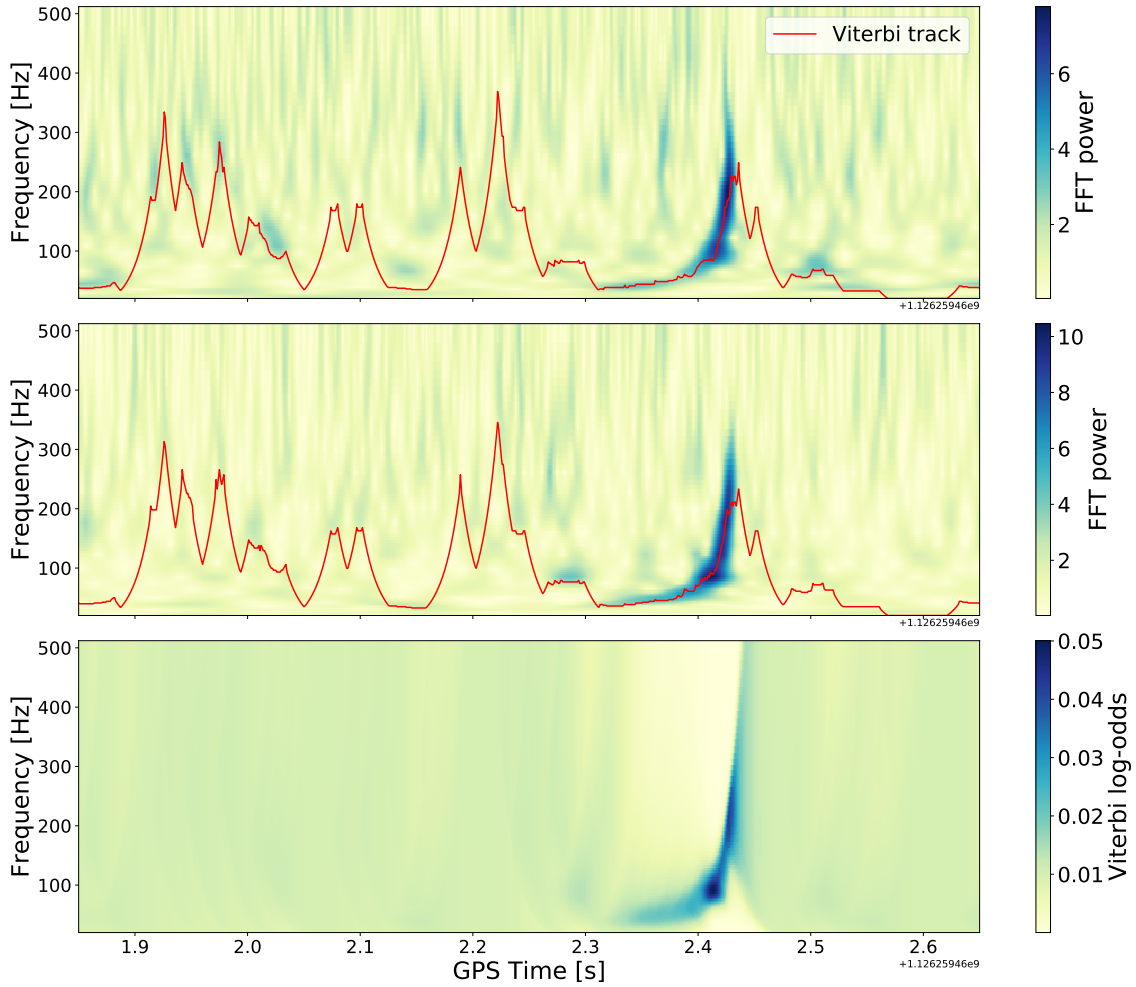


Figure 3.13: The Q transform is taken around GW150914 [4]. The top two panels who the Q transform for H1 and L1 respectively, where the red track is the Viterbi track identified in each of the transforms. The final panel then shows the output Viterbi map for this data. The two tracks follow the frequency evolution of the BBH signal as is sweeps through the band and the Vitebri map shows areas of large log-odds where the BBH signal has high Q transform power.

3.14 Computational cost

One of the main strengths of this search is the drastically reduced computational cost when compared to other current CW searches. The scaling of the computing cost can be estimated for a single detector by looking at the number of calculations that need to be made. The number of calculations for a single detector search, $N^{(1)}_{\text{calcs}}$ is,

$$N^{(1)}_{\text{calcs}} = n_1^m NM, \tag{3.45}$$

where n_1 is the size of the transition matrix, N is the number of SFTs, M is the number of frequency bins and m is the amount of memory described in Sec. 3.6. Where the computing cost scales linearly with the number of frequency bins and SFTs. In the following test we ignore 'memory' and look at the time taken for the single detector search where the time taken to read and save data is ignored. Here the data is the same size as the S6 MDC for a single detector search and the search is over a 0.1 Hz band, where we set $n_1 = 3$. This test, and the following test, was run locally on a MacBook Air with a 1.3 GHz Intel Core i5 processor .We can then write the time taken ,T, as,

$$T = 0.56\,\text{sec}\left(\frac{\text{N}}{22538}\right)\left(\frac{\text{M}}{180}\right)\left(\frac{\text{N}_{\text{bands}}}{1}\right), \tag{3.46}$$

where N_{bands} is the number of different frequency bands. For the multiple, Q, detector case, we can then generalise Eq. 3.45 and write the number of calculations $N^{(Q)}_{\text{calcs}}$ as,

$$N^{(Q)}_{\text{calcs}} = NMn_1^m \prod_{q=1}^{Q} n_{q+1}, \tag{3.47}$$

where n_1 is the first dimension of the transition matrix, Q is the number of detectors and n_{q+1} is the size of the transition matrix element which refers to detector q. For our tests we set $n_1 = n_{q+1} = 3$ and use 2 detectors i.e., $Q = 2$ which each have the same size data as the previous test. The actual time taken to run however, depends on the version of the algorithm which is run. For example, including the line aware statistic slows the search slightly. For the two detector case where two SFT powers are summed,

$$T_{\text{sum-power}} = 1.35\text{s}\left(\frac{\text{N}}{22538}\right)\left(\frac{\text{M}}{180}\right)\left(\frac{\text{N}_{\text{bands}}}{1}\right). \tag{3.48}$$

The same search now including the line aware statistic, which is implemented using a lookup table, changes this to,

$$T_{\text{line-aware}} = 25.7\text{s}\left(\frac{\text{N}}{22538}\right)\left(\frac{\text{M}}{180}\right)\left(\frac{\text{N}_{\text{bands}}}{1}\right). \tag{3.49}$$

Other searches, excluding Einstein@home which takes on the order of months to run (> 100 million core-hours [56]), take $1-10$ million core-hours [56]. Running the line-aware statistic search should take ~ 14 core-hours to run between 100 and 200 Hz, not including the generation of data.

3.15 Discussion

3.15.1 Paper discussion

In this paper we describe an application of the Viterbi algorithm, called SOAP, to search for continuous sources of gravitational waves. This paper outlines the method and derives the statistics behind the method in a consistent Bayesian formalism. It then presents the results from the first set of tests of sensitivity for the SOAP algorithm on three separate datasets.

We tested SOAP on a set of fake isolated pulsar signals in the 100 – 200 Hz range, based on 1800s SFTs summed over one day. The three datasets that included these signals comprised continuous Gaussian noise, Gaussian noise but with temporal gaps corresponding to LIGO dead times in the S6 data run, and real data, i.e., the S6 MDC. Although a major attraction of SOAP is its sensitivity to a wide range of signal types, in the tests above it was optimised to detect isolated pulsar signals below 100 Hz with low spin-down to offer a comparison with other CW searches. From these tests, by setting a 95% efficiency and a false alarm of 1%, we found that in the case of continuous Gaussian data we could detect a signal with an optimal SNR of ~ 60 and a depth of $\sim 33\,\mathrm{Hz}^{-1/2}$ with an RMS of the difference between the injected and Viterbi track being ~ 2 frequency bins (0.0012 Hz). When gaps were introduced into the data to simulate S6 we could detect a signal with an SNR ~ 72 and a depth of $\sim 10\,\mathrm{Hz}^{-1/2}$, with an RMS of ~ 10 bins (0.0056 Hz). The drop in sensitivity here is simply because there is $\sim 50\%$ less data compared to the previous case. Finally, in the S6 MDC we could detect a signal with an SNR ~ 74 and a depth of $\sim 13\,\mathrm{Hz}^{-1/2}$. These real data contain non-Gaussian artefacts such as instrumental lines and this causes a further drop in sensitivity. Whilst not a full comparison to other searches in the S6 MDC [56], as we only tested on a subset of the bands, this search has a sensitivity which is comparable to some other CW searches, however offers a massive increase in speed.

We chose the specific frequency band to search over as the data which we used, i.e., the summed data, becomes less effective at frequencies much higher than 200 Hz, and using the parameters of our simulations, signals can spread over many frequency bins in a day, reducing sensitivity further, however this can be mitigated by using shorter SFTs or performing their summation over 12 (rather than 24) hours.

The methods described in this paper present a basic approach for gravitational-wave

signal searches using SOAP. However there are several further developments that could increase its sensitivity. Some of these are outlined below:

One of the main features which reduces the sensitivity of the search is non-gaussianities within the data, namely instrumental lines. Although we have a statistic which penalises these features, in some cases it will also penalise a strong signal. For example, when the amplitude of the noise floor is high for one detector or the duty factor is lower, the signal will appear more like an instrumental line to this statistic. We hope to improve the search statistic in the future by searching for consistent amplitudes as opposed to consistent SNR, i.e, the statistic will take the amplitude of the noise floor and the duty factor into account.

One variation of this method which has been described in this paper is 'memory', which is where the tracks jump in frequency is determined by the previous n jumps. This has yet to be fully tested, however, we expect that this will increase our sensitivity to signals where have a better idea of their frequency evolution. This however, comes at a cost in computational time which we can estimate given Eq. 3.47 in Sec. 3.14.

Further additions to the search include using the Fourier transform of the SFT power along the Viterbi track as a detection statistic. If the Viterbi track follows that from an astrophysical signal, then we should see the effects of the antenna pattern in this Fourier transform as a peak at half a sidereal day. If the track follows something which is not astrophysical then this should not be seen this peak in this Fourier transform. This only applies to the search directly on the SFTs not the summed data, as the antenna pattern variations will have been averaged out in the summing.

As well as searching for astrophysical signals, SOAP can also be used to search for and identify instrumental lines. Here we use single detector data, or multiple channels from a single detector, to identify quasi-monochromatic features on the data for further study.

Whilst this paper presents initial tests on sensitivity, further tests will be needed for a full comparison to other CW search methods. This search, however, aims to look for signals which may not follow the standard frequency evolution and is intended to return potentially interesting candidates for a more sensitive followup.

3.15.2 Extra work discussion

Sections 3.9, 3.11, 3.12 and 3.13 are additional to the published work in [75]. In Sec. 3.9, the line-aware statistic was developed to be able to search through detectors with unequal sensitivities and duty cycles.

In Sec. 3.11 the optimisation of the line-aware statistic was described. This optimised the parameters of the line aware statistic in a case where the data does and does not contain instrumental aretefacts. This revealed that there are large areas of parameter space where the search performs at or close to its optimum. As the optimisation procedure is time consuming, this allows the same set of line-aware statistic parameters to be run on

multiple observing runs withough having to re-optimise the parameters.

The tests conducted in Sec. 3.10 were all in the sensitive frequency band of LIGO. To see how the sensitivity of SOAP varied with Sec. 3.12 th

CHRIS: You could add to this section with a discussion about the extra things you've added in the Sec. 3.9, Sec. 3.11, Sec. 3.11.3 and Sec. 3.11.4.

Chapter 4

Machine learning for continuous wave searches

Machine learning is a term which has been around since the 1950s, and is a subset of what is known as artificial intelligence. This is a field which aims to use computers to learn information without being given explicit instructions. With the increase in available computing power in recent years along with the easier access to large data-sets, machine learning has become a much more accessible technique. One of the techniques in particular is a method known as deep learning which uses deep neural networks. These have been used extensively in classification problems as well as many others. Neural networks wave a large range of applications, and have gained increasing popularity for use in GW data analysis problems.

This chapter aims to give an overview of neural networks, specifically CNNs and their application to a CW search. The majority of this section is written in a paper which is yet to be published, however, is aimed to be submitted soon. The differences are, Sec. 4.3, 4.4 and 4.5 which describe the operation of neural networks have more detail in this thesis than in the paper draft. Sec.4.10 is additional material which is currently not included in the paper draft.

4.1 Introduction

Gravitational wave detectors such as LIGO [43, 11] and VIRGO [12, 44] search for a number of different targets. Some targets such as CBCs have been observed [7, 5, 4], however, other primary sources such as CWs are yet to be observed. CWs are well modelled quasi-sinusoidal signals with a duration much longer than observing times of detectors. The source of these signals is thought to be rapidly rotating neutron stars which can emit GW if there is some asymmetry around its rotation axis. This can be caused by various mechanisms as described in [76]. These signals have small amplitude, which if detected

will be below the noise PSD of the detector. Therefore, sensitive search algorithms are needed to find the signals. These algorithms generally fall into three categories: Targeted, directed, and all-sky searches, listed in order of how much is known a priori about the source from EM observations.

In targeted searches the sky position, frequency, and its derivatives are assumed to be well known, in directed searches only the sky position is known and in all-sky searches the sky position and frequency of the source is unknown. The most sensitive of these are targeted searches which use coherent matched filtering [49, 48]. These use template waveforms which are generated using the information already known about the source, then correlated this with the data. Directed and all-sky searches have a much broader parameter space to search, therefore, many templates are needed to sufficiently cover the parameter space. Using the coherent matched filter for broader parameter space searches becomes unfeasible due to the amount of computing time that is needed. This led to the development of semi-coherent searches where the data is divided up into smaller segments which can be analysed separately and then the results can be recombined incoherently using various methods [82, 81]. Semi-coherent searches result in a trade off between sensitivity and computing time.

The analysis here is presented mainly as an addition to an existing semi-coherent search algorithm titled SOAP [75]. This is a fast and largely un-modelled search which finds tracks of high FFT power in time-frequency spectrograms. When applied to multiple detectors using a line-aware statistic, SOAP looks for frequency bins which have both a high power and are similar in each detector. This means that at a given frequency at a given time, SOAP will penalise frequencies where the FFT power is largely different in each detector. The algorithmic details summarised in Sec. 3.

One effect which limits the sensitivity of SOAP and many other GW searches is noise artefacts known as 'instrumental lines'. These can be anything from long duration fixed frequency or wandering lines to shorter duration fixed frequency transients. There are certain types of instrumental line which the SOAP search can struggle to distinguish from an astrophysical signal even with the development of a 'line aware' statistic in [75]. Currently the method used to reduce the effect of these lines is to manually look at the SOAP output and the spectrograms for each sub-band to determine whether the sub-band is contaminated by instrumental effects. This process is slow, requires a lot of human input and is subject to human error. When the search runs over a larger bandwidth, it will no longer be practical to look through all bands.

We aim to automate how the search deals with instrumental lines by using CNNs. These have been used extensively in image classification and we explain this in more detail in Sec. **??**. CNNs have already been shown to detect gravitational wave signals from CBCs in [92, 93, 94] and other deep learning techniques have been used in searching

for CW signals in [95].

In Sec.4.2 we will summarise the basics of how the SOAP search works. In Sec. 4.3, 4.4 and 4.5 we explain how CNNs operate and how they are trained. Sec. 4.6 will describe how this is applied to an CW search. This includes the structure of the CNN in Sec. 4.6.1 and the entire search from raw data to results in Sec. 4.8. Finally in Sec. 4.9 we show the results from this search and compare to similar analyses.

4.2 SOAP

SOAP [75] is an un-modelled search for long duration signals which is based on the Viterbi algorithm [68]. In its most simple form SOAP analyses a spectrogram to find the time-frequency track which gives the highest sum of FFT power. If a signal is present and sufficiently loud then this is the track which is most likely to come from some signal. In [75] the algorithm has been expanded to search through multiple detectors as well as including a statistic to penalise artefacts in the data from the instrument as opposed to from an astrophysical source.

Fig. 4.1 shows an example of the spectrogram data which is searched through and the outputs of SOAP; the three main output components are the frequency track, the Viterbi statistic and the Viterbi map.

Viterbi track The Viterbi track is the most probable track through time-frequency data given a choice of statistic.

Viterbi statistic The Viterbi statistic is the sum of the individual statistics along the Viterbi track. In the analysis that follows, the 'line-aware' Viterbi statistic is used. This is the sum of the log-odds ratios, $p_{\text{signal}}/(p_{\text{line}} + p_{\text{noise}})$ along the track. This is defined in more detail in [75]

Viterbi map The Viterbi map value of the Viterbi statistic for every time-frequency bin in the spectrogram. This represents the most probable track which ends in any time-frequency bin. In the Viterbi maps, each time slice is normalised individually, i.e., each vertical slice has been normalised such that the sum of their exponentiated values is equal to 1. This way each pixel in the image can be interpreted as a value related to the log-probability that there is a signal in the bin at that time.

To determine whether an simulated astrophysical signal has been detected, in [75] we used the Viterbi 'line aware' statistic alone described above. The 'line-aware' statistic reduced the affect of instrumental lines on the analysis, however the 'line-aware' statistic is still contaminated by certain types of line. For example, the statistic is affected by broad wandering lines as they offer high power tracks in both detectors. To reduce the

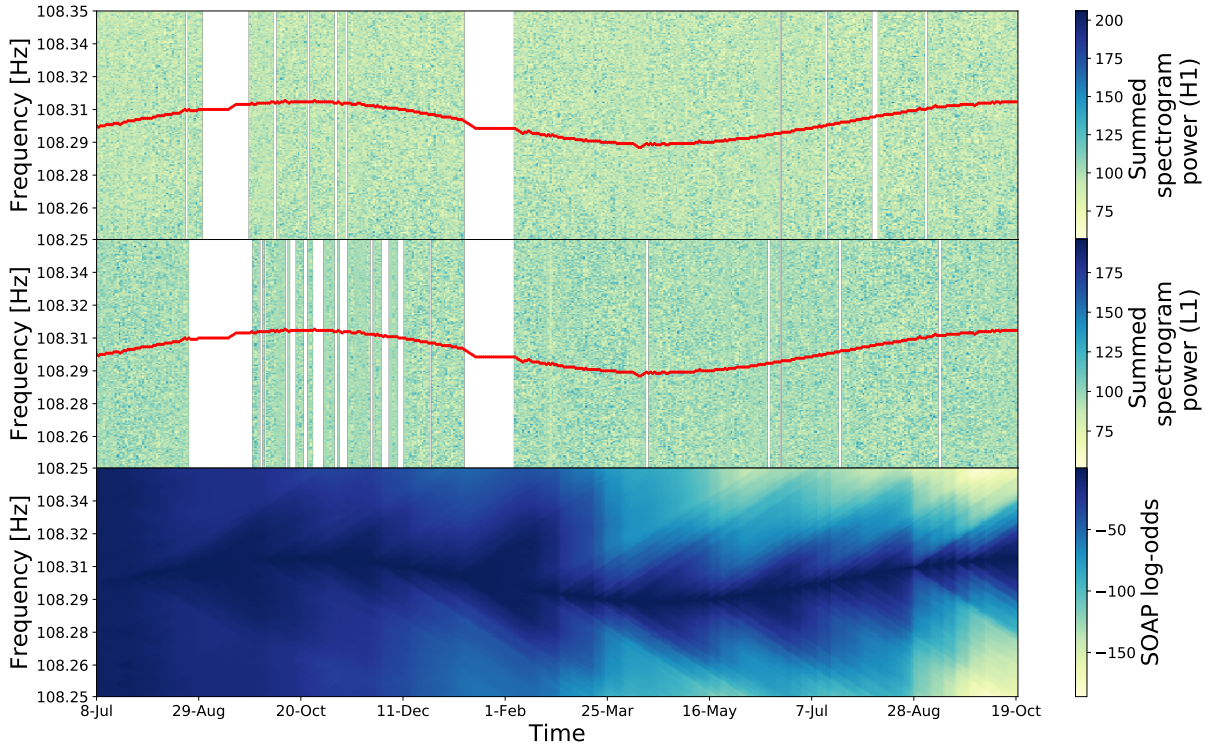


Figure 4.1: This plot shows the inputs and outputs of the SOAP search. The top two panels are time-frequency spectrograms which have been pre-processed as described in Sec. 4.8. This data is a 0.1 Hz wide frequency band from the S6 observing run [] **JOE: reference** which includes a string simulated CW signal. The white areas in the spectrograms are gaps in data when the detector was not operating. These both have the optimal track found by SOAP overlaid. The bottom panel shows the normalised Viterbi map, the intensity of a pixel in this image relates to the log-probability that a track ends in and particular frequency bin at a given time.

effect of these instrumental lines, we looked through the spectrograms and Viterbi maps of individual bands by eye as in Fig. 4.1. Bands which appeared to be contaminated were then removed from the search.

In this search the spectrograms and the Viterbi map contained extra information over the Viterbi statistic. We aim to utilise this in addition to the Viterbi statistic to replace the process of removing contaminated bands 'by eye' and therefore automating the search. A useful tool which can be used to classify this extra information is convolutional neural networks.

4.3 Neural networks

Throughout this section I will summarise one machine learning technique known as a neural network. Neural networks, as the name may suggest, were developed as a way for a computer to mimic neurons in the brain. To understand why this would be useful, one can

try to design an algorithm to identify hand written digits. This seems like a simple task as a brain can complete with ease. However, writing a traditional algorithm to perform this same task is very difficult. The algorithm would have to identify a particular shape which has a huge amount of variation. Neural networks offer a way to deal with this problem as they can be trained on large datasets, similar to how a human brain is 'trained'. In the lifetime of a brain, many examples of different hand written digits are seen. For each new example the brain 'updates' itself based on the new version of the observed digit. This process is replicated in a neural network where the algorithm can be updated for each example, with the goal of correctly identifying a digit. A neural network has many parameters which can be modified or 'trained', it is these parameters which are updated after each new example of a digit. These parameters are grouped into objects called neurons, many combinations of neurons can then be used to build a neural network.

4.3.1 Neurons

Neurons are the building blocks of any neural network. They perform simple operations on any number of input values and then output a single value. The output o of a neuron is defined by the equation

$$o = f\left(b + \sum_{i=1}^{N} w_i x_i\right), \tag{4.1}$$

where b is the bias, x_i is an input value with a corresponding weight w_i, f is the activation function, o is the output and N is the number of inputs. Here the inputs $\mathbf{x}$ represents either the data which is input, in the example above this is the pixels in the digits image, or the output of another neuron. The weights $\mathbf{w}$ then represent the importance of each data point to this neuron. The bias b is then just an extra factor which can shift the data by a fixed value. The activation function f is then a function which can have many forms, in the simplest case in a neuron known as a 'perceptron', it provides a cut where any value above a given threshold is 1 and any below is 0, this will be explained in more detail in Sec. 4.3.3.

In the example in Fig. 4.2 I have shows a neuron which has 4 input variables, or 4 input data points. When a network is trained the weights and the bias are updated to better represent the input data. This training procedure is explained in more detail in Sec. **??**. Many neurons are then used in combination with each other to develop a neural network which can be applied to more complex problems.

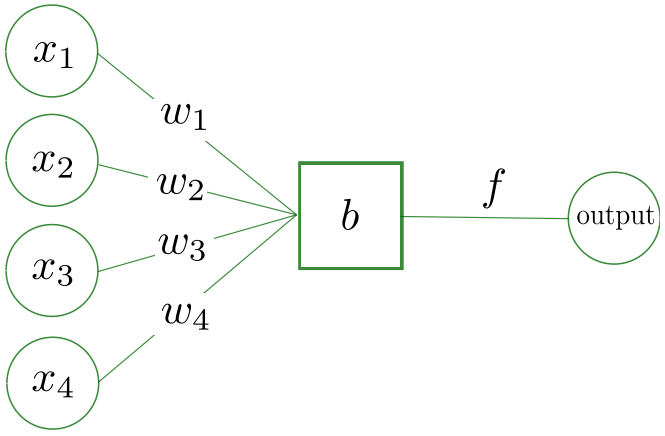


Figure 4.2: Basic neuron showing the four input parameters x_n and their corresponding weights w_n. These are multiplied and summed as in Eq. 4.1. A bias b is added and this value is passed through and activation function f to the output.

4.3.2 Network structure

The structure of a neural network is defined by the user and there is no set way to design a network. However, the general layout of a neural network is defined by structures called layers, sometimes known as fully connected layers. These are rows of N neurons which all take the same input such that there is N output values. An example of a simple neural network is shown in Fig. 4.3. The first layer is the input layer, this is just the data points from an input example. In the example of hand drawn digits, this would be the pixels from the image of the digit. The final layer represents the information that you intend the network to extract from the input data. In the hand drawn digit example, this could have 10 output neurons corresponding to each digit 0-9. Each of these outputs is then a value which is related to the probability of that digit being present in the image.

When designing a network, the user will have a defined input layer size from the data and a desired number of output neurons which represents, for a classification example, the number of output classes. The number of hidden layers and the number of neurons in those hidden layers can be arbitrarily changed. In general if the data contains more complex information the size or complexity of the network will need to be increased for it to be able to extract the information.

4.3.3 Activation functions

The activation function transforms the sum of the data and weights as in Eq. 4.1. The most simple activation function is to set a threshold where for any number above this the output is 1 otherwise the output is zero. However, this type of activation known as a perceptron does not always perform well in neural networks. Activations functions

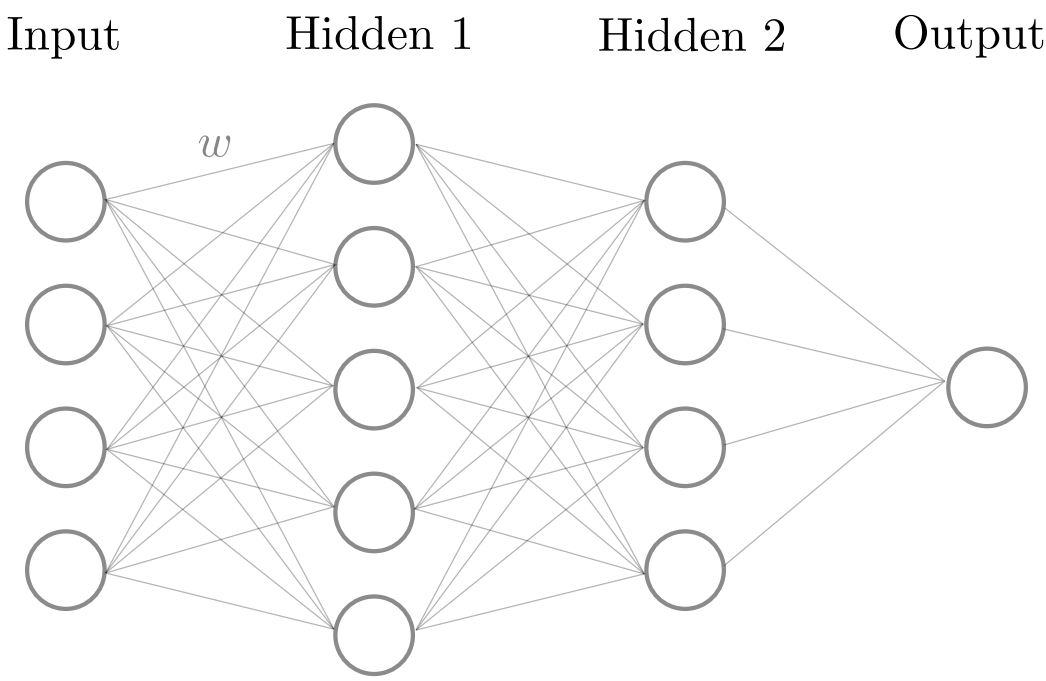


Figure 4.3: A neural network is structured with layers. Each of the circles in these layers are neurons as described in Sec. 4.3.1 and Fig. 4.2. The networks contain an input layer which is usually the data which you would like to analyse. Then this passes to a number of 'hidden' layers, in the above diagram there are two. Hidden layers are just layers which exist between the input and the output. The output layer is then the desired output, above I have chosen a single neuron as output. This is such that the network could classify the input to a value between 0 and 1. Every neuron in a layer is connected to the output of all neurons in the previous layer.

are generally non-linear, this reflects the non-linearity of real world problems and allows networks to learn that. A linear activation function means that any number of layers in a network is equivalent to a single layer network. Another property which is desired in activation function is that it is continuously differentiable. This is to allow algorithms such as gradient descent to optimise the network [96]. There are many choices when defining this in the network, some of the available options are shown in Fig. 4.4 [96]. One of the more commonly used activation function is the LeakyRELU function, this is explained in more detail in [97]. In the work that follows we use the LeakyRELU function and the sigmoid function. The sigmoid function is used on the output such that values are constrained between 0 and 1, the LeakyRELU is used everywhere else.

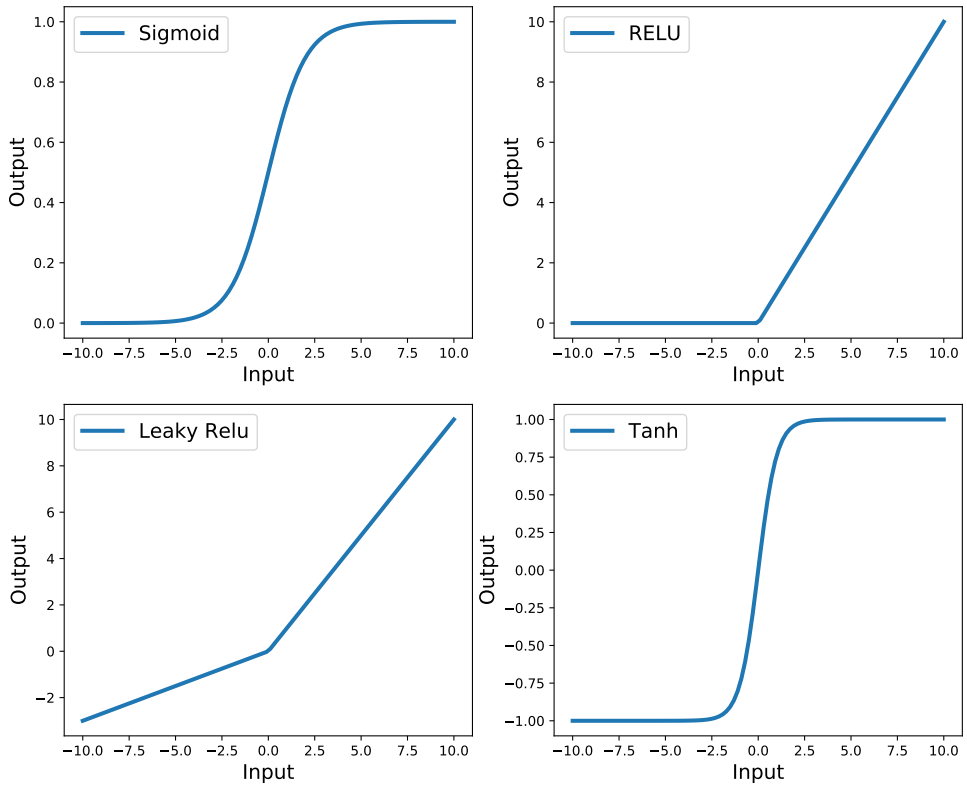


Figure 4.4: There are many different activations function which are used, any function can be defined for this however, a subset of the more commonly used functions is shown here.

4.4 Convolutional Neural Networks

CNNs are a different type of deep neural network than in Sec. 4.3, they are primarily used in image processing and recognition [98, 99, 100, 101]. A CNN has a similar goal to a full connected neural network; it is designed to take in data, identify different features within that data and classify what those features or combinations of those features mean. In the context of this work the input data is a time-frequency spectrogram which may contain a simulated CW signal. The output is then a single number gives a probability that a signal is present. A CNN can learn how to identify features by being trained on many examples of the input data which has a label. For example, an input spectrogram with a simulated CW signal would be labelled to have a signal. Given the set of training examples, the many parameters of the CNN can be updated such that it gives the best result for any new image. This process is the same as neural networks in Sec. 4.3 and will be described in greater detail in Sec. 4.5.

The key features of CNNs which distinguish them from ordinary neural networks is some additional types of layers including: Convolutional layers and max pooling layers.

4.4.1 Convolutional layers

Convolutional layers have some similarities to fully connected layers as described in Sec. 4.3.2. The main difference being how the weights are applied to the inputs. If we assume that the input to the network is some image, then a fully connected neural network would flatten this image and apply Eq. 4.1 to the input pixels. This involves having a separate weight for each of the input pixels in an image. A convolutional layer however, filters the image and outputs a filtered image of the same size (the image can be a different size it depends how the layer was set up). This convolution is defined by

$$O_{i,j} = f\left(\sum_m \sum_n F_{m,n} x_{i-m,j-n}\right), \tag{4.2}$$

where O is the output image, x is the input image, F is the convolutional filter and f is the activation function. The weights of the filter $F_{m,n}$ are what are updated when the network is trained. Figure 4.5 shows an example of a 6x6 image and the results of filtering the image using Eq. 4.2 with two different filters F. In this case the network has 4 parameters for each filtered image which can be updated as opposed to the 36 which a full connected network would have for a single neuron.

Figure 4.5 demonstrates how a filter which matches a feature in an image can highlight that particular feature. i.e. the diagonal line in the bottom left of the input is enhanced by Filter 1, which matches that feature. When this type of layer is trained, the weights of the filter are updated. After training the filter weights should then ideally match the feature which is intended to be extracted from the image.

A convolutional layer has a number of different hyper-parameters which can be varied when setting up a CNN. Below I list each of the adaptable parameters and what they do.

Filter size The filter size is the size and shape of the convolutional filter. In Fig. 4.5 we use a filter size of 2×2. The filter does not have to be square, however must be less than the dimensions of the image.

Number of filters The number of filters can be any value. The convolutional layer will output the same number of filtered images as there are filters. In Fig. 4.5 we use two filters and therefore, the output of the layer is two images.

Activation function The activation function is generally kept the same for each of the layers, however this can be set here. The different types have been explained in Sec. 4.3.3 and are applied as in Eq. 4.2.

Stride A normal convolutional layer applies a filter by multiplying by a filter, then shifting over by one pixel and repeating. Applying a stride mean rather than shifting by one

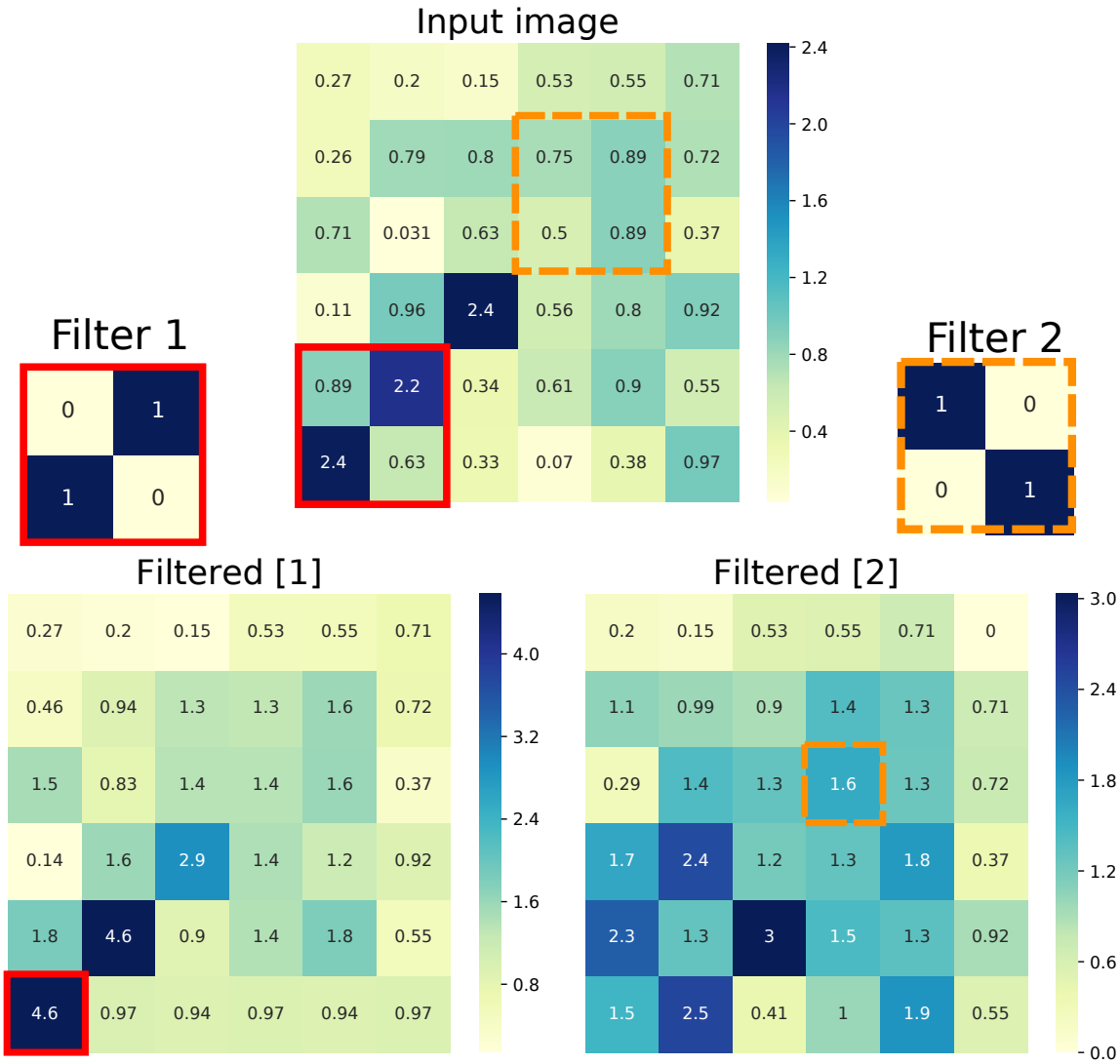


Figure 4.5: Convolutional filters can be designed to 'pick out' certain features within an image. In this simple example above, the first filter (filter 1) matches the diagonal line in the bottom left of the input better than filter 2. The output filtered image the exaggerates this filter. The coefficients of the filter i.e. $F_{m,n}$ in Eq. 4.2, in this are set to ones and zeros. These are the weights which are trained by the network. In this an cases that follow, to get the same size image in the output as the input, the image is padded with zeros. I this image it was necessary to pad above and to the right of the image. The output of a convolutional layer is then the filtered images above after a bias and activation function have been applied.

pixel, one shifts by a number greater than one. This reduces the size of the output by the same factor of stride. i.e. if you skip one pixel (a stride of 2) then the image will be half the size on output. This has a similar affect to max-pooling which we describe in Sec. 4.4.2.

The convolutional layers can reduce the number of updatable parameters used in each network compared to an equivalent fully connected network. However, the output of a convolutional layer is a number of images which are potential the same size as the input. This has potentially increased the size of the parameter space for the next layer. To decrease this a type of layer known as max-pooling is used.

4.4.2 Max pooling layers

Max pooling layers are designed to reduce the size of the problem whilst holding on to as much important information as possible. These do not contain any trainable parameters. The idea of this layer is relatively simple, it reduces the image size by taking the maximum value in a region of a given size. Fig. 4.6 shows the output of the first filtered image in Fig. 4.5. The image is then reduced by a 2×2 max pooling layer. The output of max-pooling Then shows a large value in the bottom left, this is where the input image matched the filter in Fig. 4.5. This demonstrates how the max-pooling layer can hold on to important information whilst reducing the image size.

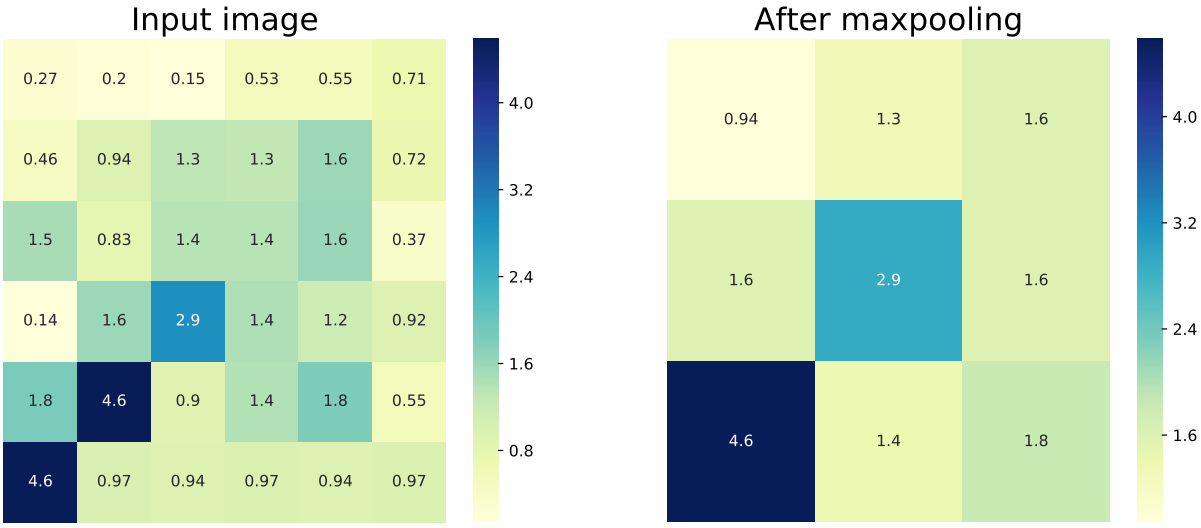


Figure 4.6: Max pooling layers aim to reduce the size of am image whilst retaining important information within the original image. Above shows an example where a 2×2 max-pooling layer is used on the output of Filter 1 in Fig. 4.5. This retains the information that the input image matches the filter in the bottom left.

n

4.4.3 CNN structure

CNNs are usually structured such that they can extract larger features from an input image, then the outputs from this are passed on to be classified. The 'feature extraction' part of the network consists of the convolutional layers and the max-pooling described in Sec. 4.4. The outputs of the final max-pooling layer are then flattened and used as the input to a fully connected network. This fully connected network the classifies these outputs into a number of classes. Figure 4.7 shows an example of the layout. Here an input image which is the same as in previous examples is passed onto a single convolutional layer with two different filters. The output of two filtered images is passed to a max-pooling layer. The two max-pooled images are flattened into 18 input neurons, this then passes through a fully connected network to a single output neuron. This shows a simple example, however, there are many hyper-parameters of the network which can be changed. These include: the number of filters in a convolutional layer, the number of convolutional layers and max-pooling layers, the number of hidden layers in the fully connected section and the number of neurons in the hidden layers. This example also shows the network being classified to a single output as this is how we use CNNs for the following work.

4.5 Training

Once the structure of the network is decided, the network needs to be trained. This means that the weights and bias' for every neuron and filter need to be updated such that the neural network gives a useful output. For this work we will classify input time-frequency spectrograms into a signal or noise class using a single output neuron. This neuron outputs a value in the range [0,1] by using a sigmoid activation function. In our case the CNN is trained using a process called supervised learning. In supervised learning, the class of each input example is know. For example, we assign a label of 1 when the input is a time-frequency spectrogram which includes a simulated CW signal. Similarly a time-frequency spectrogram with no simulated signal is assigned a label of 0. In general when training neural networks this way, the performance of the network can be improved by increasing the number of input examples which are shown to the network. This stops the network from over-fitting to specific examples. Instead it should generalise to the full input and learn the underlying features within the data.

4.5.1 Loss function

Initially each of the training examples is propagated though the network to its single output value which lies between 0 and 1. Using a loss function, this output is then be compared to the label of the input data which is either 0 or 1. There are many types of

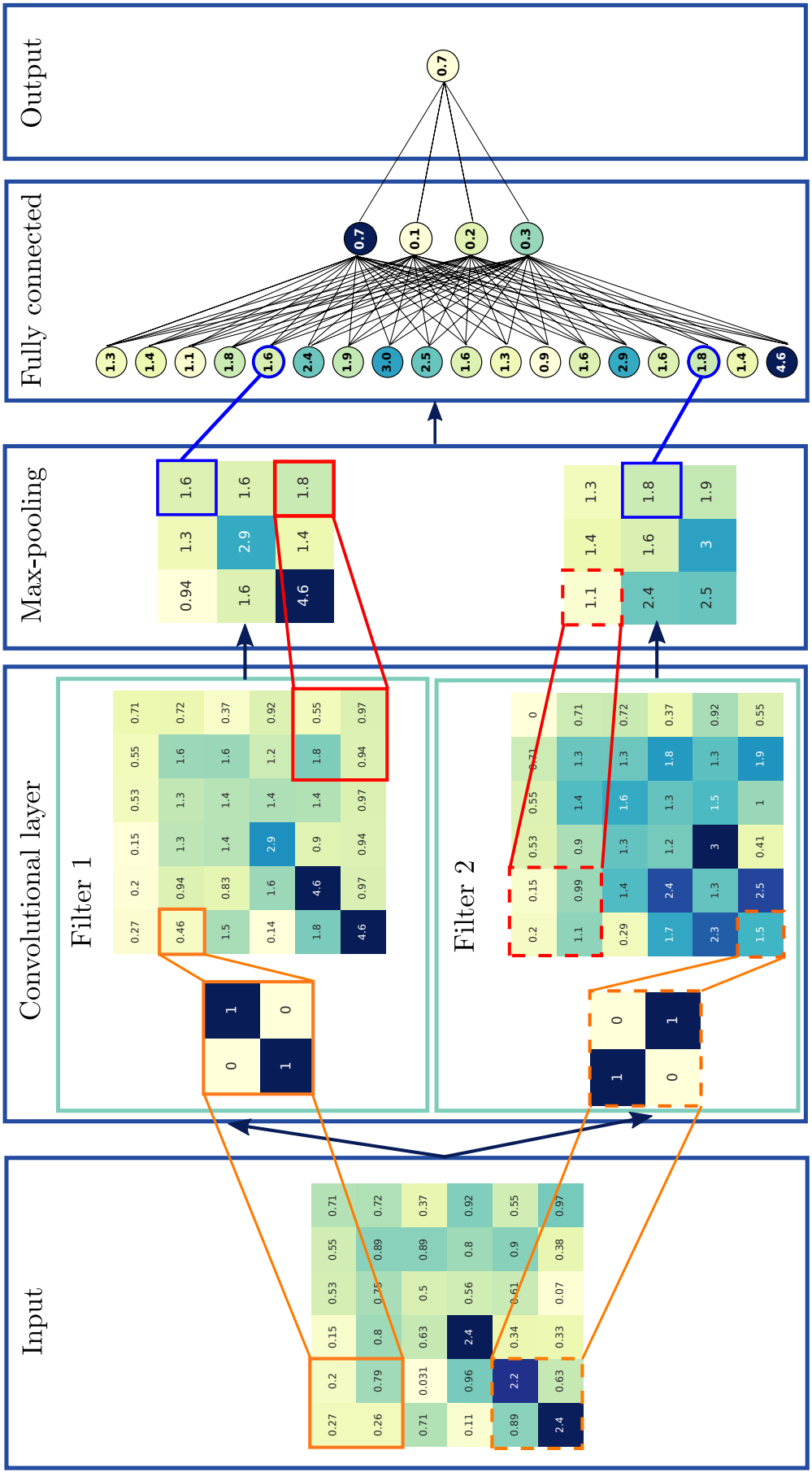


Figure 4.7: Convolutional neural networks consist of two broad sections, the 'feature extraction' part which is the convolutional and max-pooling layers, and the classification part which is the fully connected part of the network. This diagram shows a simple example of an image passing through a single convolutional layer with two filters, a single max-pooling layer and a simple fully connected network with a single hidden layer consisting of 4 neurons. The values in this diagram omit the use of any activation function such that the values are easier to follow. In a real network the activation functions are important.

loss function which can be used, this depends on the type of problem which one wants to solve. As we are classifying between two classes in out networks, the loss function, L, is the binary crossentropy defined as

$$L = -\frac{1}{N}\sum_{i}^{N} y_i \log(p_i) + (1 - y_i) \log(1 - p_i), \tag{4.3}$$

where p is the networks predicted output which has any value in the range $[0, 1]$ and y is the true output which has binary labels 0 or 1. This is calculated as the sum over all training examples. The loss function is minimised when the output matches the 'truth'. This tells the neural network how close to the truth this output is. The weights and bias' of the neural network can be updated based on the value of this loss function. The process of updating the weights and other parameters is called back-propagation, and typically uses a form of gradient descent [102]. Back-propagation uses the derivative of the loss function with respect to a weight to update that weight. If changing that weight in a particular direction decreases the loss function, then the weight will be updated in that direction. The size of the change of the weight value is related to the change in the size loss function. This means that the weights can be updated to minimise the loss function and therefore improve the performance of the network.

4.5.2 Training procedure

The training procedure entails passing a set of training examples through the network a number of times. Once the entire training data set has been passed through the network (forward pass) and the weights have been updated accordingly (back propagation), the training has completed one epoch. If the data was passed and the weights were updated a single time, the loss may decrease by is likely not at a minimum. Passing the data through again may move the weights to a lower loss. This process is repeated a number of times to try and find the minimum loss. When training there the value of the loss at each epoch is monitored, the trend of the loss of the training set should always decrease. In general subset of the training data is set aside and not used in the training procedure, this is known as validation data. After each epoch the value of the loss for this validation set can be measured, i.e. all the validation data is passed though the network. This can be used to monitor the training of the network. If the validation loss begins to increase then this is a sign that the network is over-fitting to the training data-set.

4.6 Application to CW search

The aim for this work is to use a CNN to classify LIGO data into one of two classes: signal or noise. Here the signal class refers to a CW signal from an isolated neutron star as described in Sec. 2.1. Noise then refers to anything else which appear in the data, from Gaussian noise to instrumental artefacts. In Sec. 3.10 to reduce the effect of instrumental artefacts, each of the search sub-bands was analysed by eye to determine if a sub-band was contaminated. Sub-bands which contained an artefact were then removed from the search. This is a time consuming process. The main goal of the CNN approach is to automate this part of the search. This section will describe how we design the network to extract features and distinguish signals from instrumental artefacts. We will then present results form searches in a range of LIGO observing runs which include: S6, O1 and O2.

4.6.1 Network structure

In this section the structure of the networks which are used in this analysis are described. There are three main inputs of data for each CNN: spectrograms, Viterbi maps and the Viterbi statistic. Each of these are different representations of the raw detector data. In this analysis we train a separate CNN for each of these inputs and then a further three which use these combinations of inputs: Viterbi map + spectrogram, Viterbi map + Viterbi statistic and Viterbi map + Viterbi statistic + spectrogram. In all of the layers excluding the output layer of each CNN, the activation functions in Eq. 4.2 and 4.1 are defined by a function titled 'leakyRELU' [97]. For our output neuron a sigmoid function is used as an activation function such that the output is limited between 0 or 1. For a given input a CNN can then output a value between 0 and 1. When the output value is closer to 1, the input is more likely to contain a signal. The output value can then be treated as a detection statistic. The structure of the network is shown in Fig. 4.8 and is explained below.

Viterbi statistic This is the simplest of the networks and will give the exact same result as the Viterbi statistic on its own. This is a single neuron which takes in the Viterbi statistic applies a weight and bias and then passes through a sigmoid function.

Viterbi map The Viterbi map CNN takes in a down-sampled Viterbi map of size (156,89), this is described more in Sec. 4.7.3. This CNN consists of two convolutional layers and 3 fully connected layers. The first layer has 8 filters which have a size of 5×5 pixels, the second layer has 8 filters with a size of 3×3 pixels. After each of these layers we use a max-pooling layer with a size of 8×8 pixels. This then passed into three fully connected layers which all have 8 neurons and used leakyRELU activation functions. Finally these lead to an output neuron which uses a sigmoid function.

Spectrogram The spectrogram CNN takes in a down-sampled spectrograms of size (156,89), this is described more in Sec. 4.7.3. This CNN has an identical structure as the Viterbi map CNN, however, takes two channels as input. The two channels are the spectrograms of two different detectors.

The next three networks are constructed from combinations of the previous described CNNs.

Viterbi map and spectrogram To combine the spectrogram and Viterbi map network, we remove the final output neuron and its 8 weights from each of the networks. The outputs from each network is then 8 neurons. These can be combined to a single sigmoid neuron which has 16 new weights.

Viterbi map and Viterbi statistic In this network we combine the Viterbi statistic with the Viterbi map. As before, this uses the pre-trained Viterbi map and Viterbi statistic CNNs. The output sigmoid neuron and corresponding weights are removed from each network. The 8 neurons from the Viterbi map network and the single neuron from the Viterbi statistic network are then combined to a single neuron with 9 new weights.

Viterbi map, Viterbi statistic and spectrogram This combination takes all component CNNs from above. As before the final sigmoid output and the corresponding weights from each network are removed. The 8 neurons from the Viterbi map and spectrograms CNNs and the single neuron from the Viterbi statistic are then joined into a single output neuron with 17 new weights.

When combining CNNs we use a process called transfer learning [103]. This uses the pre-trained weights of the networks as a starting point to continue training. In our examples we found that we could fix the weights inside the pre-trained networks and just train the final 16 output weights from the neurons as in Fig. 4.8. These combinations of networks were chosen as the different representations of the data should contain slightly different information on the input. For example, the Viterbi statistic contains no information on the structure of the track in the data and the Viterbi maps lost some information about lines in the band. The addition of the spectrograms aimed to include even more information about this piece of data. Where when each of these are combined, the CNN should be able to pick to important information from each of these representations.

4.7 Data generation

To train the CNNs we need to generate many examples of data, this include the three data products above: Time-frequency spectrograms, Viterbi maps and the Viterbi statistic.

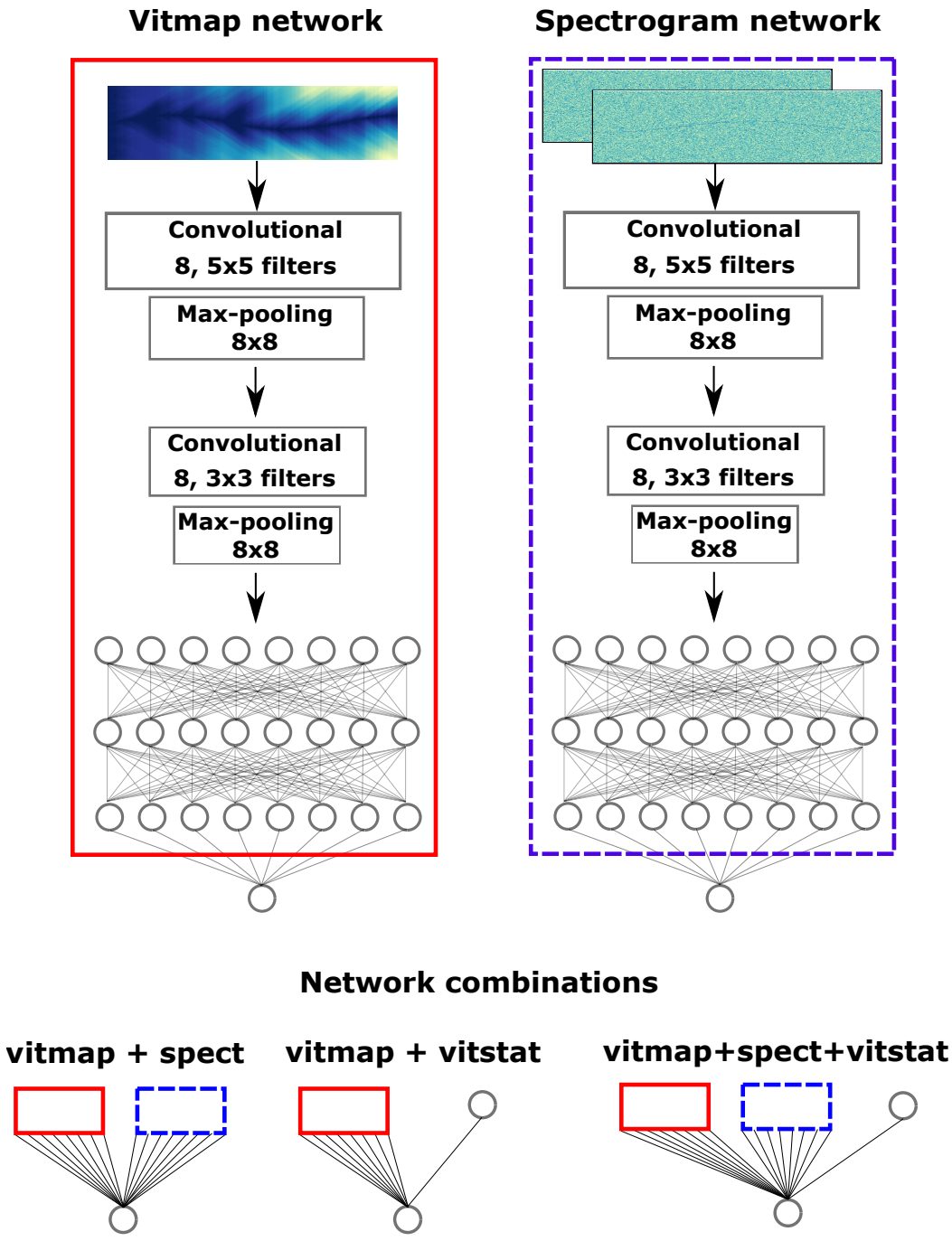


Figure 4.8: The structure of the Viterbi map and spectrogram CNNs used in this analysis are the same, with the difference that the spectrogram takes two images as input. They each use two convolutional layers and 3 fully connected layers before they're output to a single neuron which represents the probability of belonging to the signal class. The Viterbi statistic network is a single neuron that transforms the statistic into a number between 0 and 1 representing the probability of belonging to the signal class. For the combinations of networks, we remove the final output neuron and its 8 weights, i.e. we take the part inside the red or blue box. The 8 outputs from each network are then combined to a single neuron with 16 new weights.

When a CNN is trained it needs to see examples of all possible features which could appear in the data. This include, Gaussian noise, non-Gaussian artefacts and CW signals. As non-Gaussian artefacts are difficult to simulate, it is possible to use the non-Gaussian artefacts in real data as part of the training set. Therefore, for the majority of the analysis that follows, the time-frequency spectrograms which are used to generate the Viterbi data are from real detector data. The exact observing runs used will be explained in Sec. **??**.

For the analysis that follows there are three main sets of data: training data, test data and search data. Training data uses a set of augmented (see Sec. **??**) time-frequency spectrograms containing simulated signals and is used to train each of the networks. Test data is a separate set of simulations in time-frequency spectrograms which are not augmented. These are used to generate efficiency curves and test the network. Search data does not contain any simulated signal injections and is used to search for real signals within the data.

When training and testing a network it is important that the networks are not trained and tested on the same data. Otherwise the CNNs can learn specific features of the training data and not the underlying distribution of features. To avoid this, the spectrograms are split into 0.1 Hz wide sub-bands where alternating bands are designated as 'odd' or 'even'. This means that bands starting with 100.1,100.3 are odd and 100.2,100.4 are even etc. The networks can then be trained on the odd bands and tested on the even bands and vice versa. This then means that each time we want to search over data, we will have two final networks. One which will be run on odd bands and a separately trained network which is run on even bands.

4.7.1 Signal simulations

To inject the simulated signals into real data we generate a random set of signal parameters which are drawn from prior distributions defined in Table **??**. The SNR of each simulation is then uniformly distributed between 50 and 150. Where the SNR is the integrated 'recovered' SNR. This is calculated for each time segment using the definition of optimal SNR in [55], the total SNR is then the sum of the squares of these. The GW amplitude h_0 is scaled based on the noise PSD to achieve this SNR. The power spectrum of the signal can then be simulated in each time segment of a time-frequency spectrogram. This is done by assuming that the spectrogram is χ^2 distributed. The the antenna pattern functions are taken into account for the given source parameters and detector such that the SNR for each time segment is calculated. This SNR is spread over neighbouring frequency bins dependent on its location in frequency. The power spectrum values can then be drawn from a non-central χ^2 distribution with the non centrality parameter equal to the square of the SNR. Each signal is simulated in two detectors: LIGOs H1 and L1. The SNRs reported below are then the sum of the squares of the SNRs from each detector.

Table 4.1: Table shows the upper and lower limits over which each signal parameter was randomized. The parameters $\alpha, \sin(\delta), f, \log\left(\dot{f}\right), \cos(\iota), \phi_0, \psi$ were sampled uniformly in the ranges specified in the table. The frequencies f_l and f_u refer to the lower and upper frequency of the band that each signal is injected into. Excluding the distribution of frequencies f, all the injections parameters are sampled from the same distributions as the S6 MDC [56].

	α [rad]	$\sin(\delta)$ [rad]	f [Hz]	$\log_{10}\left(\dot{f}[\mathrm{Hz/s}]\right)$	$\cos\iota$ [rad]	ϕ [rad]	ψ [rad]
lower bound	0	-1	$f_\mathrm{l} + 0.25$	-9	-1	0	0
upper bound	2π	1	$f_\mathrm{u} - 0.25$	-16	1	2π	$\pi/2$

4.7.2 Augmentation

To train a neural network, many examples of data from each class are needed to avoid over-fitting. In our case when we use data between 40-500 Hz, splitting the data into 0.1 Hz wide sub-bands does not give enough data for the networks to be trained effectively. Therefore, using a technique called data augmentation [104, 105] we can artificially increase the number of training examples. Augmentation is when data is transformed such that, to the network, it appears to be 'new' data. For example, by shifting a time-frequency band up and down in frequency, this appears to be a new realisation of noise which we can then inject a simulated signal into. This would double the size of the training data-set and reduce the likelihood of over-fitting to the training data.

The augmentations are applied to the spectrograms from each of the detectors. The augmentations that are used on each sub-band are: reversing the data in time, flipping the data in frequency, rolling the data in time by a small number of segments and shifting the data in frequency by a small number of bins. As we use real data, there are gaps in time where the detectors were not operating. We preserve the location of these gaps when augmenting the data. When shifting the data in frequency, we shift each band up and down by 30 frequency bins (0.016 Hz) and up and down by 60 frequency bins (0.032 Hz). When rolling the data in time, we roll each sub-band by 100 time segments (100 days). Fig. 4.9 shows examples of the original data, a flip in frequency, a roll in time and a flip in time. For each frequency shift, we flip the sub-band in time and frequency and roll the sub-band in time. This then gives us 3 transformations for each of the 4 frequency shifts, which including the original data gives 20 times the number of training examples.

4.7.3 Downsampling

One further issue for our data sets are their size. The spectrograms we use have a large number of pixels within them. This means that as the spectrograms are passed through the network, there are a large number of computations. Both this number of computations and

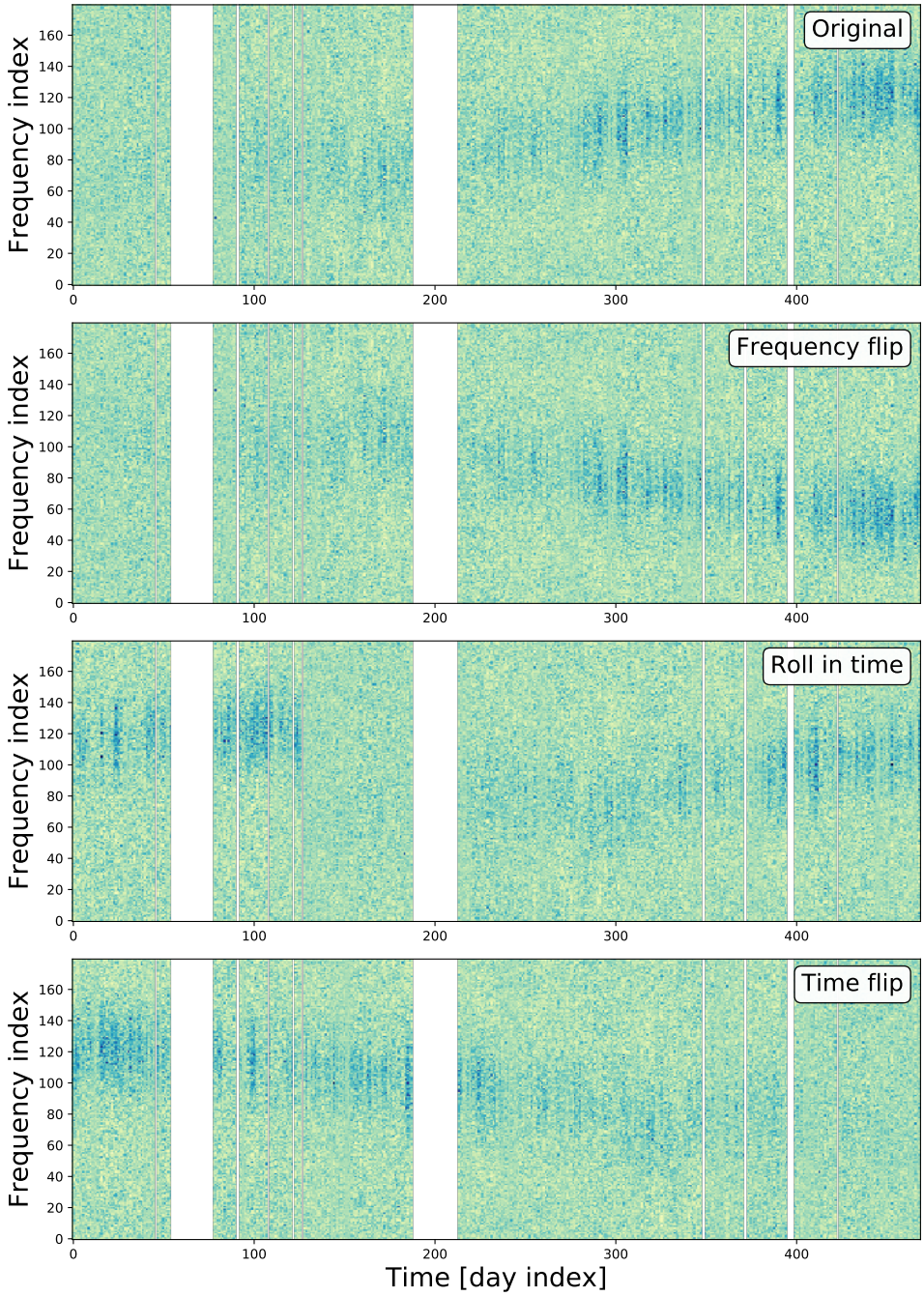


Figure 4.9: The data is transformed by flipping the data in frequency (panel 2), rolling the data in time by 100 bins (panel 3) and flipping the data in time (panel 4). The original summed spectrogram is show in panel 1. Simulated signals can then be injected using this data as noise. The plots above show a broad wandering line to demonstrate the changes to the data when it is augmented, however, the majority of sub-bands contain almost Gaussian noise.

the memory requirements of the GPU mean that training a network with a large number of data points takes longer. We implement a few methods to reduce the size of the data: summing time segments of spectrograms and down-sampling these summed spectrograms.

The spectrograms are summed over one day, i.e., every 48 time segments, as in [75]. This should increase the SNR for a given signal within a given time-frequency bin assuming that the signal remains within the frequency bin for the majority of the time segment. To reduce the size of the data further, the package 'resize' from scikit-image [106] is used, this uses interpolation to resize the summed spectrograms to a size of (156,89) [time segments,frequency bins]. This size was defined based on the summed spectrograms of the S6 data-set. This is 1/3 the number of summed segments in time, 1/2 the number of segments in frequency. The down-sampling is applied to the spectrograms and vitmap. In [75] we demonstrated that summing spectrograms can increase the speed and sensitivity of our search. When down-sampling the image, we found that reducing the amount of data had a small affect on the sensitivity of the CNNs used.

4.8 Search pipeline

In previous sections each component of the search pipeline has been described, however, described below is how each component fits together. Figure 4.10 shows a flow diagram of the pipeline. The pipeline is run in three different ways: training the CNN, testing the search and running a search on real data.

1. **SFTs** Generate 1800s long SFTs from detector time-series data. SFTs of this length are a standard set for CW searches which are continuously generated during observing runs by members of the LIGO collaboration.

2. **Normalising** The SFTs are then divided by their running median with a window width of 100 frequency bins. If we assume the resulting SFTs to be χ^2 distributed, we can apply a correction factor using LALSuite code `XLALSFTtoRngmed` [87] such that their power spectrum has a mean of ~ 1. By then multiplying this by 2, the noise like parts of the spectrum are χ^2 distribution with two degrees of freedom.

3. **Narrowbanding** The computational efficiency can be improved if the data is split into narrow bands. This is because the analysis can be completed on each band in parallel on separate CPU nodes. In this search the spectrograms are split into 2.1 Hz wide bands every 2 Hz, i.e. 100.0-102.1, 102.0-104.1 etc. The bands are 2.1 Hz wide as the analysis on each node will further split the data into 0.1 Hz wide sub-bands. The overlap then allows the sub-band from 1.95-2.05 to be calculated on a node. This band size was chosen based on the available computational memory at the time.

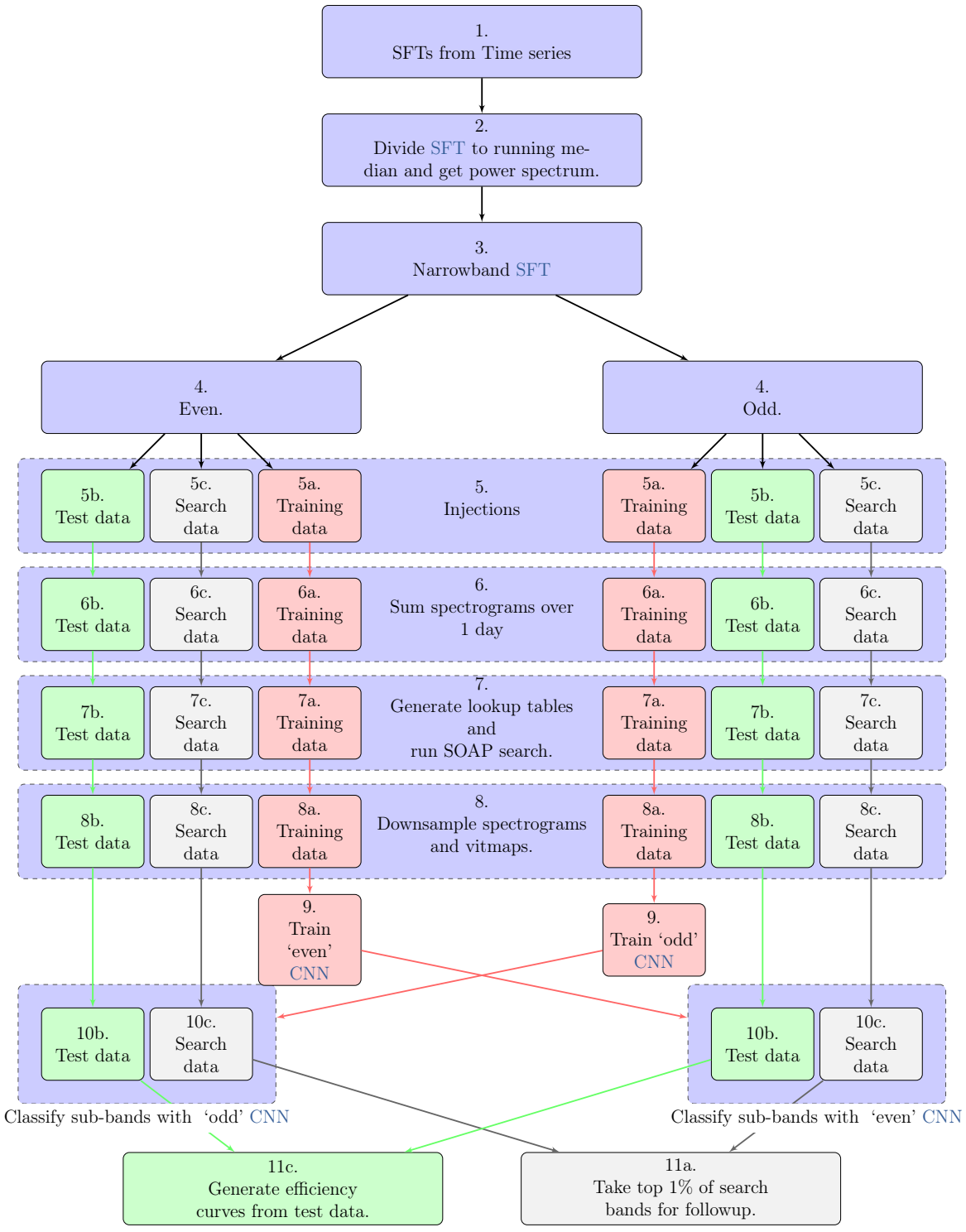


Figure 4.10: This diagram shows the SOAP pipeline from start to finish. There are three main sections: Training (red), Testing (green) and Searching (grey) for both the odd and even bands. The blue sections mean that the same operations is done in all cases.

4. **Band splitting** A CNN should not be trained on the same data that it will be tested on. For this reason, each of the 0.1 Hz wide sub-bands are split into 'odd' or 'even' bands. A CNN can then be trained on even bands and tested on odd bands and vice versa.

5a. **Training data generation** To generate training data the process is the same as described in Sec. 4.7. Each of the 0.1 Hz sub-bands is 'augmented' as in Sec. 4.7.2. For each of the augmented bands, the data is duplicated such that there is a second copy of every augmented band. In the copied set of bands, signals are injected into them with SNRs in the range 50-150. This gives us and example for a noise class and a signal class. There are two of these sets, one for 'even' bands and one for 'odd'.

5b. **Test data generation** For test data, signals following the parameters in Tab. 4.1 are injected in to 50% of the 0.1 Hz sub-bands. These signal have and SNR in the range 20-200. The SNR range here is wider than the training set as a method to test how the trained networks perform s on a wider range od SNRs. Here we again have a set for 'odd' and a set for 'even'.

5c. **Search data** This data is generated such that we can search for a real signal. The sub-bands described in part 4 are now overlapping by 0.05 Hz. This means that if there is an astrophysical signal it should be fully contained within at least one sub-band. We do assume that a signals frequency does not drift by more than 0.1 Hz, which is assumed to be true for isolated neutrons stars < 500 Hz. There are both 'odd' and 'even' versions of this search data.

6. **Summing spectrogram** As in [75] the spectrograms are summed over one day, i.e., every 48 time segments (1 day) of the spectrogram are summed. This is done separately for each of the 6 data-sets (3 for 'odd', 3 for 'even').

7. **Generate lookup tables and run SOAP search** Before the SOAP search is run, the line-aware statistic lookup tables need to be generated as in [75]. Then for each of the 6 data-sets (3 for 'odd', 3 for 'even') the SOAP search is run separately.

8. **Down-sample data** At this stage there are four elements which are saved for each of the 6 data-sets. The two spectrograms, the Viterbi maps and the Viterbi statistic. The spectrograms and the Viterbi maps are down-sampled to a size of (156×89) using interpolation from scikit-image's resize [106]. This size was chosen based on the S6 MDC data-set, where this is $1/3$ the length in time and $1/2$ the width in frequency of the summed spectrograms. This was chosen such that the CNNs trained efficiently and still achieved a reasonable sensitivity.

9. Train Networks The down-sampled training data is then used to train a CNNs. One CNN is trained on 'odd' bands and a different CNN with the same structure is trained on 'even' bands.

10b. Run search on test data The trained CNNs from part 9 are then used to classify each sub-band in the test data with injections, this returns a statistic on the range $[0, 1]$. The close the value is to 1 the more likely it is from an astrophysical signal, therefore, the statistic can be interpreted as an estimate of the probability of a signal being present. Here the CNN trained on the 'odd' bands is tested using the 'even' bands and vice versa. The algorithms are run on this test data to asses the sensitivity of the analysis.

10c. Run search on real data The trained CNNs from part 9 are then used to classify each sub-band in the search data, this returns a statistic in $[0, 1]$. This statistic is the same as in part 10b. Once again the CNN trained on the 'odd' bands is tested using the 'even' bands and vice versa.

11a. Signal candidates The signals which have a statistic in the top 1% can be taken as potential candidates. This can then potentially be followed up with other CW search methods.

11c. Efficiency curves The output statistics from the test data-set (11b.) can be plotted against SNR to see how the network classified signals with the SNR of the injection. This can potentially be extended to other signal parameters also. Then the efficiency curves can be generated, this is described in further detail in Sec. 4.9.1.

4.9 Results

The networks described in Sec. 4.6.1 were trained and tested on four different data-sets: the S6 MDC as in [75, 56], our own injections into O2 data, Gaussian noise which had the same gaps and noise floor as the S6 data-set, and our own injections into real S6 data. Each of the searches use training and testing data in the frequency range of 100-400 Hz, except the S6 MDC which uses data in the range 40-500 Hz for testing and training.

4.9.1 Sensitivity

To investigate the sensitivity of the pipeline we use two measures: the sensitivity depth $\mathcal{D}$ [55] and optimal SNR ρ [88] which are both defined in [75]. The sensitivity depth is defined as

$$\mathcal{D}(f) = \frac{\sqrt{S_h(f)}}{h_0}, \tag{4.4}$$

where $S_h(f)$ is the single-sided noise PSD and h_0 is the GW amplitude. The optimal SNR is defined as,

$$\rho^2 = \sum_X 4\Re \int_0^\infty \frac{\tilde{h}^X(f)\tilde{h}^{X*}(f)}{S^X(f)} df, \tag{4.5}$$

where X indexes the detectors and $\tilde{h}(f)$ is the Fourier transform of the time series of the signal $h(t)$. This expression is defined in [55] for a double-sided PSD and we have defined it for the more common single-sided case.

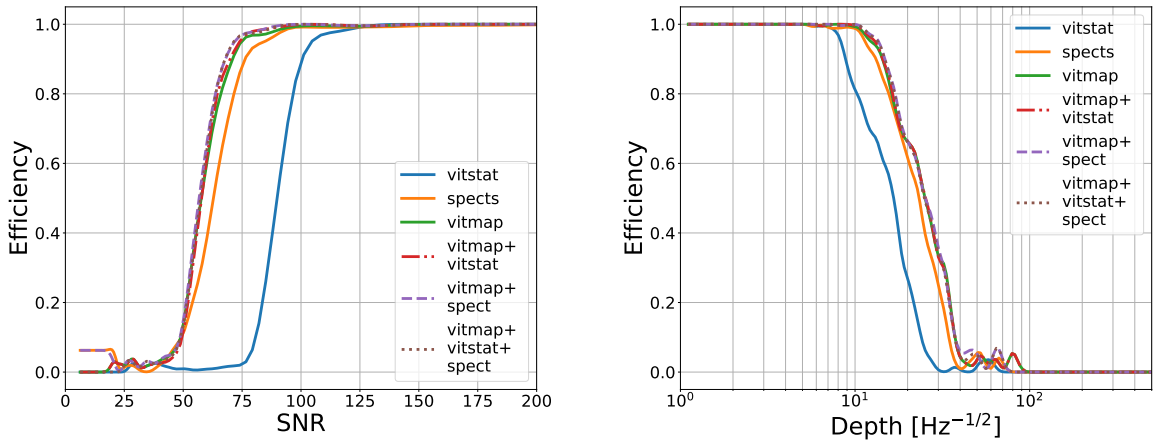
The sensitivity curves shown in Fig. 4.12,4.13 and 4.14 were generated using a 1% false alarm rate, where the false alarm threshold is the value of our statistic where 1% of sub-bands which do not contain an injection exceed that value. This is then used as a detection threshold. The efficiency is defined as the fraction of events which exceed the false alarm rate for any given SNR. The SNR is sampled uniformly between the range 20-200 as described in Sec. 4.8. Therefore, we do not have multiple simulations for a discrete SNR but adopt a different approach. Instead, one can define some window around a point in SNR and count the fraction of statistics which exceed the false alarm threshold within that window. We define the window as a Gaussian with a standard deviation of 2, this is wide enough to contain enough injections at a given SNR to achieve a reliable value. The efficiency curves y are then be calculated using,

$$y(\rho) = \frac{\sum_i H(O_i - O^{1\%})\mathcal{G}(\rho_i; \mu = \rho, \sigma = 2)}{\sum_i \mathcal{G}(\rho_i; \mu = \rho, \sigma = 2)}, \tag{4.6}$$

where O_i is the output statistic from the CNN, $O^{1\%}$ is the statistic value corresponding to a 1% false alarm rate, H is the Heaviside st ep function which has a value of 1 for positive input arguments and 0 for negative arguments. The SNR if a simulation with output O_i is defined in Eq. 4.6 using ρ_i. The current location in SNR is then ρ. The window is a Gaussian with a mean of the current SNR and a standard deviation of 2, $\mathcal{G}(\rho_i, \mu = \rho, \sigma = 2)$. The sensitivity curves for each of the described data-sets are shown in Figs. 4.12,4.13 and 4.14.

O1

For the first test, injections were made into the O1 data-set as in Sec. 4.7 between 100 Hz and 400 Hz. Then each of the 6 networks described in Sec. **??** were trained and tested on this data. Figure 4.11 shows the sensitivity curves for this test for both SNR and sensitivity depth for each of the 6 networks. Focusing on Fig. 4.11a, the least sensitive, i.e. furthest to the right, of the CNNs is the Viterbi statistic (vitstat), this is expected as we know that the Viterbi statistic is sensitive to instrumental lines. The spectrogram CNN has an improved sensitivity over the Viterbi statistic, this importantly does not involve the SOAP search but is run entirely on down-sampled and summed spectrograms. Whilst this



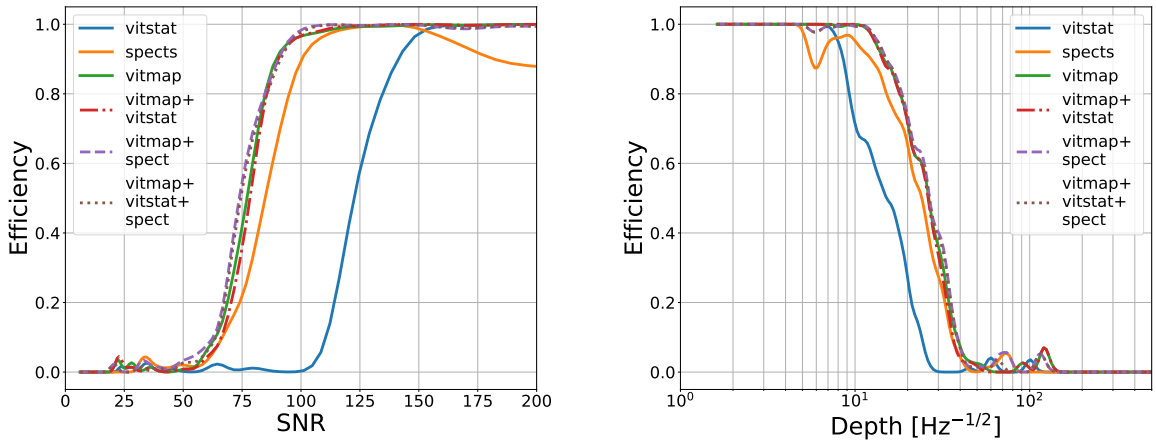
(a) (b)

Figure 4.11: In the O1 data-set, each of the six CNNs were tested. The efficiency plots above are for a 1% false alarm rate.

network is approaching the most sensitive of the examples in Fig. 4.11, and with further efforts may reach it, this network takes ~ 10 times the amount of training time. This will be explained in more detail in Sec. 4.9.2. The remaining networks achieved almost the same sensitivity. The vitmap network however, is the fastest of these to train and is used as an input for all of these remaining networks. For the O1 data-set we show that with a false alarm of 1% the Viterbi map CNN achieves a sensitivity of SNR 73 and sensitivity depth of 12 $\mathrm{Hz}^{-1/2}$ with 95% efficiency. The SNR here should not be compared between different runs as this is the integrated 'Recovered' SNR. Therefore, observing runs, such as O1, which were shorted will appear to have a greater sensitivity when they in fact do not.

O2

For the first test, injections were made into the O2 data-set as described in Sec. 4.7 between 100 Hz and 400 Hz. Each of the 6 networks described in Sec. 4.6.1 were then trained and tested on this data. Figure 4.12 shows the sensitivity curves for this test for both SNR and sensitivity depth for each of the 6 networks. Focusing on Fig. 4.12a, the least sensitive of the CNNs is the Viterbi statistic (vitstat). This is expected as we know that the despite the line-aware aspect of the Viterbi statistic, it can still confuse some instrumental lines with an astrophysical signal. The spectrogram CNN has an improved sensitivity over the Viterbi statistic, this importantly does not involve the SOAP search but is run entirely on down-sampled and summed spectrograms. This network is approaching the most sensitive of the examples in Fig. 4.12. With further investigation involving perhaps different network structures or larger or higher resolution data sets, this could potentially



(a) (b)

Figure 4.12: In the O2 data-set, each of the six CNNs were tested. The efficiency plots above are for a 1% false alarm rates. Fig. 4.12a shows the efficiency of the search as a function of SNR and Fig.4.12b shows the efficiency as a function of sensitivity depth. The efficiency here is a measure of the fraction of events which exceed the 1% false alarm probability for any given SNR. These plots both show the sensitivity of the Viterbi statistic is far below that of the different CNNs. The others are grouped with a similar sensitivity.

reach the same sensitivity as other networks. However, this network takes ~ 10 times the amount of training time compared to the Viterbi map network. This is explained in more detail in Sec. 4.9.2. The remaining four networks all achieve a similar sensitivity, each of these networks contain the Viterbi map (vitmap) as one of their inputs or their only input. Therefore, it is assumed that the dominating effect on the sensitivity originated from the Viterbi maps. In the following tests the focus will be on the Viterbi map CNN as in all cases this is among the most sensitive. For the O2 data-set we show that with a false alarm probability of 1% the Viterbi map CNN achieves a sensitivity of SNR 95 and sensitivity depth of 12 $\mathrm{Hz}^{-1/2}$ with 95% efficiency. In Fig. 4.12a the sensitivity of the spectrogram CNN drops after an SNR of 150. This is most likely due to the training set containing simulations between and SNR of 50 and 150, therefore, has not seen signal simulations of higher SNR. The dip in sensitivity in Fig. 4.12b at lower depths is from the same origin as Fig. 4.12a.

Gaussian noise

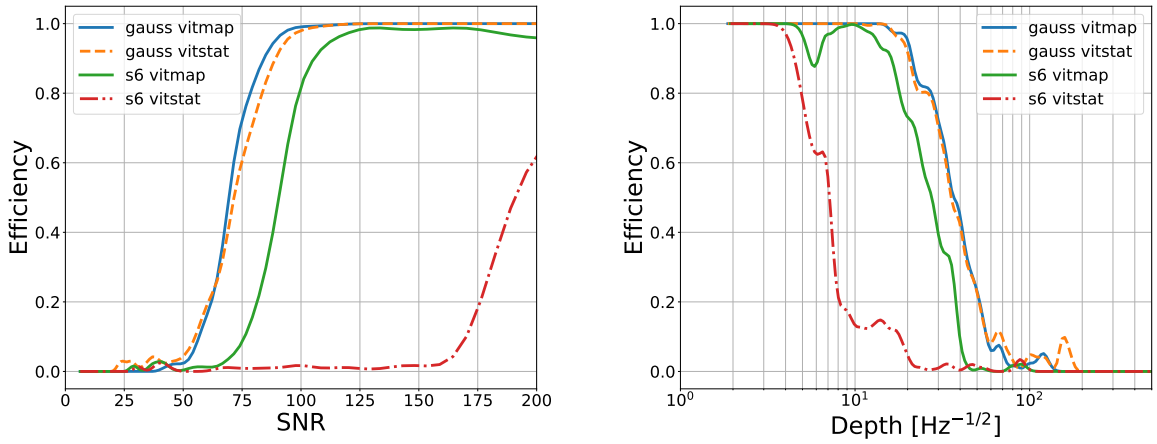
The second test involves using simulations in Gaussian noise and comparing this to simulations in S6 data. For this test we replicate the S6 data-set without including instrumental artefacts such as lines. We included the same gaps in data as S6 and the noise floor of S6 was replicated by scaling the SNR of an injection in any given SFT by an estimate of its PSD. Figure 4.13 shows the SNR and depth sensitivity curves for the Viterbi statistic

and Viterbi map CNN for both the Gaussian noise run with S6 gaps and for injections into the S6 data-set. In the Gaussian noise data-set the curves for both statistics, Viterbi map CNN and the Viterbi statistic, show very similar results, this is to be expected as the main use of the CNN was to reduce the effect of instrumental lines, for which there are none in this data set. The advantage of using the Viterbi maps in a CNN becomes clear when it is tested on simulations into real S6 data with many instrumental lines. The two curves corresponding to simulations real S6 data in Fig. 4.13a show the sensitivity as a function of SNR in these tests. It becomes clear here that the Viterbi map CNN reduces the effect of instrumental lines and therefore increases the searches sensitivity to SNR. A similar feature can be seen in Fig.4.13b where the use of an CNN greatly increases the sensitivity.

These tests on S6 data also show that the effect of instrumental lines was far greater in this run than in O2. This is shown in Fig. 4.12a where the separation between the Viterbi statistic curves and the Viterbi map curves is much smaller than the S6 curves in Fig. 4.13a. For simulations into Gaussian noise following S6 gaps we show that with a false alarm of 1% the Viterbi map CNN achieves a sensitivity of SNR 85 and sensitivity depth of 20 $\mathrm{Hz}^{-1/2}$ with 95% efficiency. For injections into real S6 data the search achieves a sensitivity of SNR 115 and sensitivity depth of 11 $\mathrm{Hz}^{-1/2}$ with 95% efficiency and 1% false alarm. We can also see from Fig. 4.13a that the sensitivity of the vitmap CNN in Gaussian noise with S6 gaps is better than in real S6 data. There are then still some artefacts in real data which reduce the sensitivity, these could potentially be non Gaussian artefacts such as weak instrumental lines.

S6

The final test was set up to again use the S6 data-set, however, in this case we use a standard set of injections in the S6 MDC [56] to compare directly to other CW search pipelines. In Fig. 4.14 we show the results of the sensitivity curves from these injections. In both Fig. **??** and **??** the sensitivity curves are substantially more noisy than in Fig. 4.12 or 4.13, this is mainly due to the size of the testing set. The standard set of simulations in Fig. 4.14 contained $\sim$ 900 signal simulations between 40 and 500 Hz where the majority of these signals are distributed between an SNR of 0 and 150. Figures 4.12 and 4.13 are generated using 2300 simulations between 40 and 500 Hz and SNRs of 20 and 200 as described in Sec. 4.8. Figure **??** shows the direct comparison in depth of the results in [56] with the results from the SOAP search with the Viterbi map CNN. This shows that we achieve a sensitivity consistent with that of other semi-coherent searches with the exception of the Einstein@home search [107]. Whilst we are not at the most sensitive end of these searches, the SOAP and CNN search offers a greatly reduced computational cost. This will be explained in more detail in Sec. 4.9.2. This particular test was limited to searching



(a) (b)

Figure 4.13: We compared the sensitivity of this search on simulations on real S6 data (s6) to simulations in Gaussian noise (gauss). Figure 4.13a shows the efficiency of the search as a function of SNR and Fig. 4.13b shows the efficiency as a function of sensitivity depth. The efficiency is the fraction of events which exceed the 1% false alarm threshold for a given SNR or depth. The Gaussian noise injections included the same gaps in data as the S6 data set. The SNR of the simulated signal in Gaussian noise was adjusted based on the noise floor of S6. In the Gaussian noise simulations the searches achieve an efficiency of 90% with 1% false alarm at an SNR ~ 85 and ~ 90 for the Viterbi map and Viterbi statistic respectively. In the real S6 noise simulations the searches achieve an efficiency of 90% with 1% false alarm at an SNR ~ 108 and > 200 for the Viterbi map and Viterbi statistic respectively.

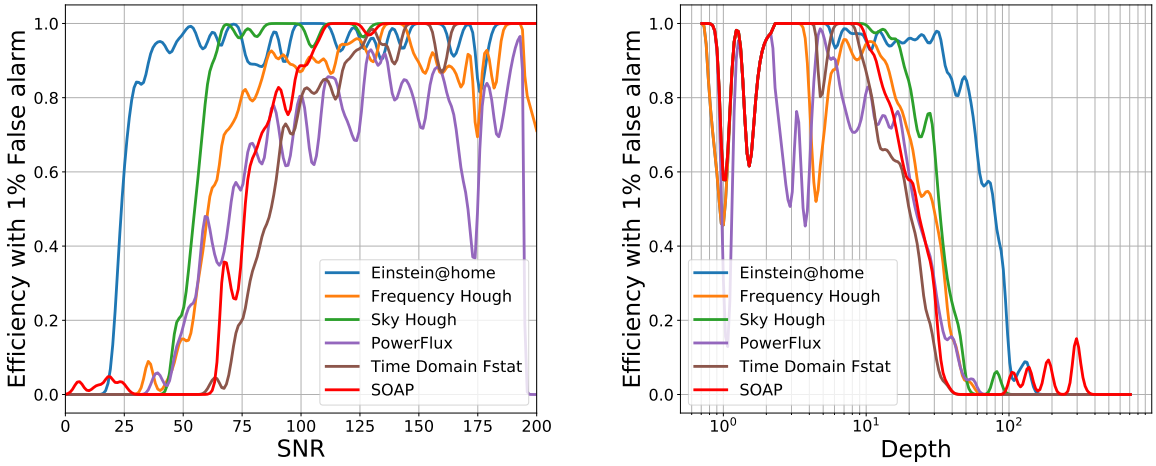


Figure 4.14: To compare the SOAP and CNN search to other existing CW searches, we used a standard set of injections used in the S6 MDC [56]. We have taken the list of detected pulsars for each search from this paper [56] and replotted using the method in Sec. 4.9.1 to compare the sensitivities to the SOAP + CNN search. This includes results for all pulsar simulations between 40 and 500 Hz. The efficiency curves are generated with a 1% false alarm probability.

for isolated neutron stars, however, unlike some other semi-coherent searches such as Einstein@Home [62] or the time domain $\mathcal{F}$-statistic [65], SOAP has a lot of flexibility in the type of signal which it can search for. The inclusion of the CNN does introduce some dependency of the search on the model as the training set for the CNN contained simulations of isolated pulsars. However, this is not a limitation of the method but of the training set as, for example, a new training set using a different signal model could be generated. For tests in the S6 MDC we show that with a false alarm of 1% the Viterbi map CNN achieves a sensitivity in SNR of ~ 90 and sensitivity depth of ~ 16 Hz$^{-1/2}$ with 95% efficiency.

4.9.2 Computational time

A key property of any CW search is the computational time taken for the search to run. Table 4.2 shows the timings for different sections of the search when run on the S6 dataset. This can be split into three main sections: data generation, CNN training and CNN testing. To get from raw SFTs to results with this search, the majority of the computational time taken is in the data generation step. The timings shown Tab. 4.2 are for the S6 observing run where each section is run on a single central processing unit (CPU) or graphics processing unit (GPU), however, in practice the generation of the data is generally run on multiple CPUs on a computing cluster. The training and testing of a

CNN is done on a singleGPU, this substantially decreases the training time compared to a CPU due to the parallel nature of neural networks.

One can start from raw SFTs from the S6 dataset without any trained networks, where this dataset has 22538, 1800 s long SFTs and we search between 40-500 Hz, i.e. 828000 frequency bins. This search would have a total computing time of ~ 386 hours on a single CPU and GPU. However, the majority of this time is taken generating the appropriate data. The generation of training, testing and search data can be easily parallelised, where in practice this is split over 200 CPUs so that it takes ~ 2 hours instead of ~ 355 hours. Rather than taking ~ 364 hours, the generating data section then takes ~ 11 hours in real time. After this parallelisation, if one was given S6 data without any trained networks, the search would then take approximately 13 hours to get an efficiency curve and a list of candidates. In this case I assume that only the Viterbi map network is trained and tested based on the conclusions from Sec. 4.9.

The computational cost could be reduced further if a network had been trained on a previous observing run. This would mean that the generating of the training data and the training of the network may not be needed. This would reduce the total run time on S6 to ~ 9.5 hours. However, this does not drastically reduce the run time as the majority of the time is spent narrow-banding the SFTs which is not run in parallel.

To reduce the time taken to generate results at the end of an observing run, one could narrowband the SFTs periodically as the data is taken during an observing run. This would allow the results to be generated within ~ 3.5 hours of the end of the run. SFTs generated on a regular basis would allow results to be generated during an observing run. This could be done, for example, on a weekly basis by adding 7 days of pixels to a spectrogram, then retraining a CNN and generating results.

The computational cost of this search is small when compared to other existing CW searches. In [56] the expected computational cost for the first 4 months of O1 for each search is shown, where the fastest search takes 0.9 million core-hours (Hough searches) and the slowest is $100 - 170$ million core-hours (Einstein@Home). The equivalent cost of the SOAP + CNN search is $\sim 100 - 200$ core-hours which is $\sim 5 - 10$ thousand times faster.

Table 4.2: The approximate timings were measured for each part of the search. This table shows the timings for training and testing starting from raw SFTs. This is the results from S6 which is the longest run we tested. We used S6 data in the frequency range between 40-500 Hz and it had a length 22538 SFTs with time base of 1800 s. In the training,testing and search data sections we averaged the SFTs over one day such that we had 469 time segments as input to the CNNs. The data generation times here are for a single CPU however, in reality this will be split across many separate CPUs. The training and testing is completed on a single GPU.

Generating data on single CPU

	Time [hrs]
Narrow-banding	~ 9
Training data	~ 240
Testing data	~ 75
Search data	~ 40

Training CNNs on single GPU

	Training time [hrs]	Loading time [hrs]
Viterbi statistic	0.03	0.2
Viterbi map	0.8	0.7
spectrogram	9	1
Viterbi map + Viterbi statistic	1	0.7
Viterbi map + spectrogram	1.4	1.6
Viterbi map + Viterbi statistic + spectrogram	1.5	2

Testing CNNs on real data on GPU

	Testing [s]	Loading [s]
All CNNs	5	$60 - 160$

4.10 Sensitivity with the size of dataset

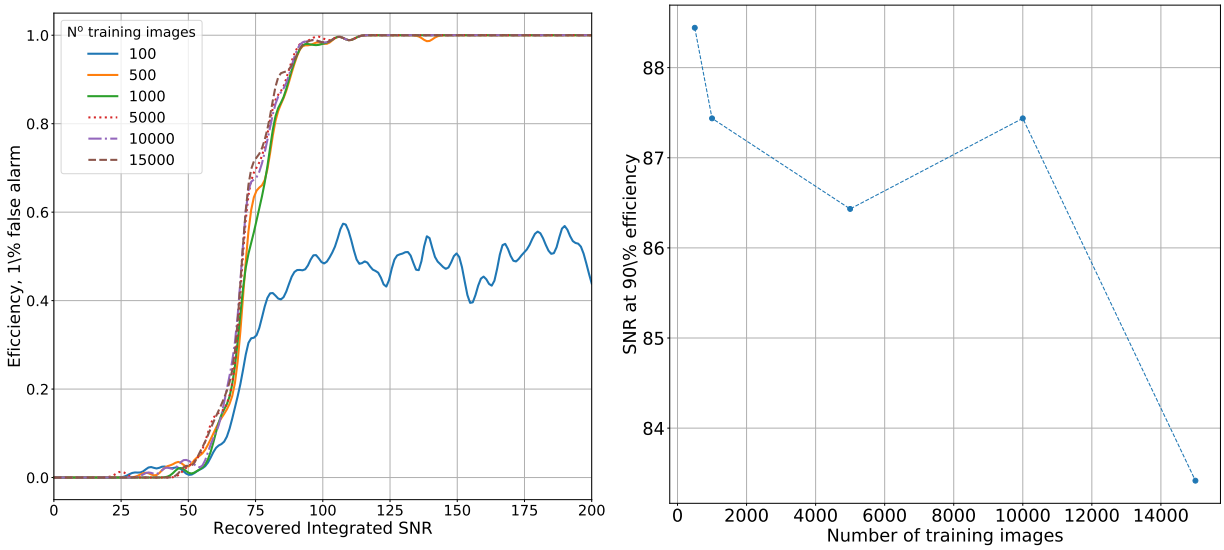
When training a CNN, the general rule is that the more data the better. More data can limit effects such as over-training mentioned in Sec. 4.5, and can increase the CNNs sensitivity. To investigate how the sensitivity of the search changes with the number of training examples, the Viterbi map (vitmap) network in Sec.4.9 was trained using a range of sizes of the training set. These networks are then tested on a separate dataset of a fixed size to see how they perform. This was repeated for two data-sets: CW simulations in Gaussian noise and simulations in LIGOs O1 data-set. For both of these cases six different networks were trained, these used: 100, 500, 1000, 5000, 10000 and 15000 Viterbi maps as their training datasets. Each of these different sizes of training data is a randomly selected subset of the training data used in Sec. 4.9.1. The test data for each was the same entire tests sets as in Sec. 4.9.1.

Figure 4.15a shows that in the Gaussian noise case, the majority of the networks performed with approximately the same sensitivity. This is with the exception of the network which was trained with 100 input Viterbi maps. Figure 4.15b shows that the sensitivity does appear to decrease, however, this is only by an SNR of ~ 5. The implication of this is that the information in the Viterbi maps is relatively easy to extract when simulations are in Gaussian noise. In this case the network is trying to distinguish Gaussian noise from a simulated CW signal. Therefore, one would expect this to be an easier problem for the network to solve compared to trying to distinguish an CW signal from instrumental lines and Gaussian noise.

When simulating signals in real O1 data, many of the sub-bands will contain instrumental lines. The noise class for the CNN then contains many variations compared to the Gaussian noise case. This is a harder challenge for the CNN as it increases the size of the parameter space. Because of this, one would expect the network to need many more training examples to be able to achieve a similar sensitivity to Gaussian noise. In Fig. 4.16a, one can see that using 100 training examples is not enough for the CNN to achieve any sensitivity at any SNR. This means that the CNN cannot classify any injection as detected using this number of training examples. Figure 4.16b seems to agree that real data poses a harder problem as the sensitivity drastically increases as the number of training examples is increased. Therefore, more training examples are needed for the network to perform well on real data.

4.11 Network Visualisation

Neural network are generally hard to visualise due to the large number or parameters in the network that have to be varied. However, there are methods which can be used to see how input data is affected by the network. This can be useful to see how the network

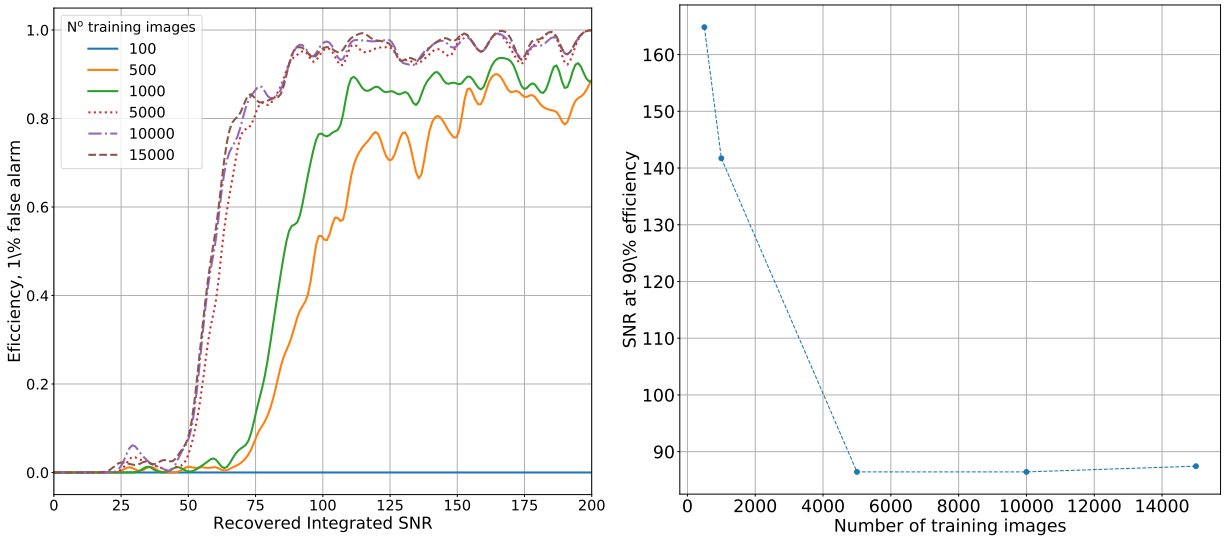


(a) (b)

Figure 4.15: The amount of training data needed for an CNN to perform well depends on the problem. Here I show how the sensitivity changes as a function of the number of training examples for simulations in Gaussian noise. Figure 4.15a shows the efficiency curves for each of the different data-set sizes and Fig. 4.15b shows the values of SNR at 90% efficiency as the data size increases. This shows that the increase in data size causes a slight increase in the sensitivity of the search. The efficiency curve for 100 training examples is not shows in Fig. 4.15b as it does not reach the 90% efficiency mark.

performs when given certain types of data and gives some insight into how the networks work.

One way to visualise the network is to pass a piece of data through the network and see how each of the layers transforms the input. Figs. 4.17, 4.18 and 4.19 show an example of this. The layout is equivalent to that in Fig. 4.8. In these figures, the outputs from each of the layers in the CNN are shown. The first being the convolutional layers, followed by their max-pooling layers. The final fully connected layers are illustrated with connecting lines. The final neuron is then the value of the statistic which we use for the above analysis. Fig. 4.17 shows an input of a Viterbi where the corresponding time-frequency spectrograms contained a strong CW signal. In this figure the next layer is the first convolutional layer, where 8 filtered Viterbi maps can be seen. This is then reduced in size by a max-pooling layer, followed by another convolutional and max-pooling layer. After this point, the figures can be compared and it becomes obvious that different neurons light up when the input is part of the signal class or when it in the noise class.



(a) (b)

Figure 4.16: Here I show how the sensitivity changes as a function of the number of training examples for simulations in O1 data. Figure 4.16a shows the efficiency curves for each of the different data-set sizes and Fig. 4.16b shows the values of SNR at 90% efficiency as the data size increases. This shows that the increase in data size causes a large increase in the sensitivity of the CNN. The efficiency curve for 100 training examples is not shows in Fig. 4.16b as it does not reach the 90% efficiency mark.

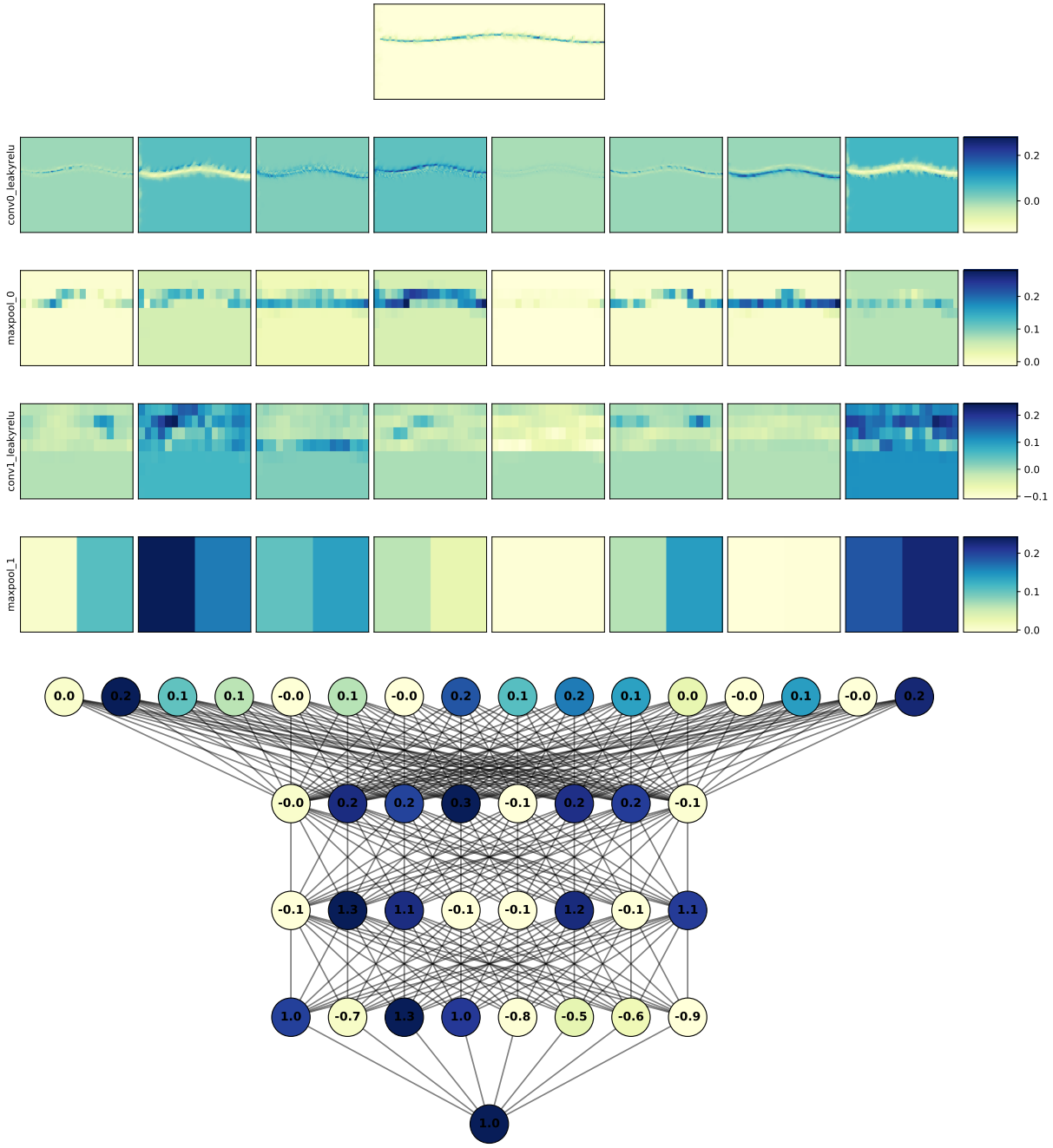


Figure 4.17: The CNN can be visualised by seeing how a piece of data is passed through the network. This figure shows how the image is processed in each layer of the network. The input here is a Viterbi map which results from the SOAP algorithm. The input to SOAP here is a time-frequency spectrogram which contains a strong CW signal. This then passes through 8 convolutional filters, then 8 max-pooling layers, then another set of 8 convolutional and max-pooling layers.the values are then flattened and it passes through 3 layers of 8 fully connected neurons.The final output neuron here is a 1 indicating that this is likely to contain a signal.

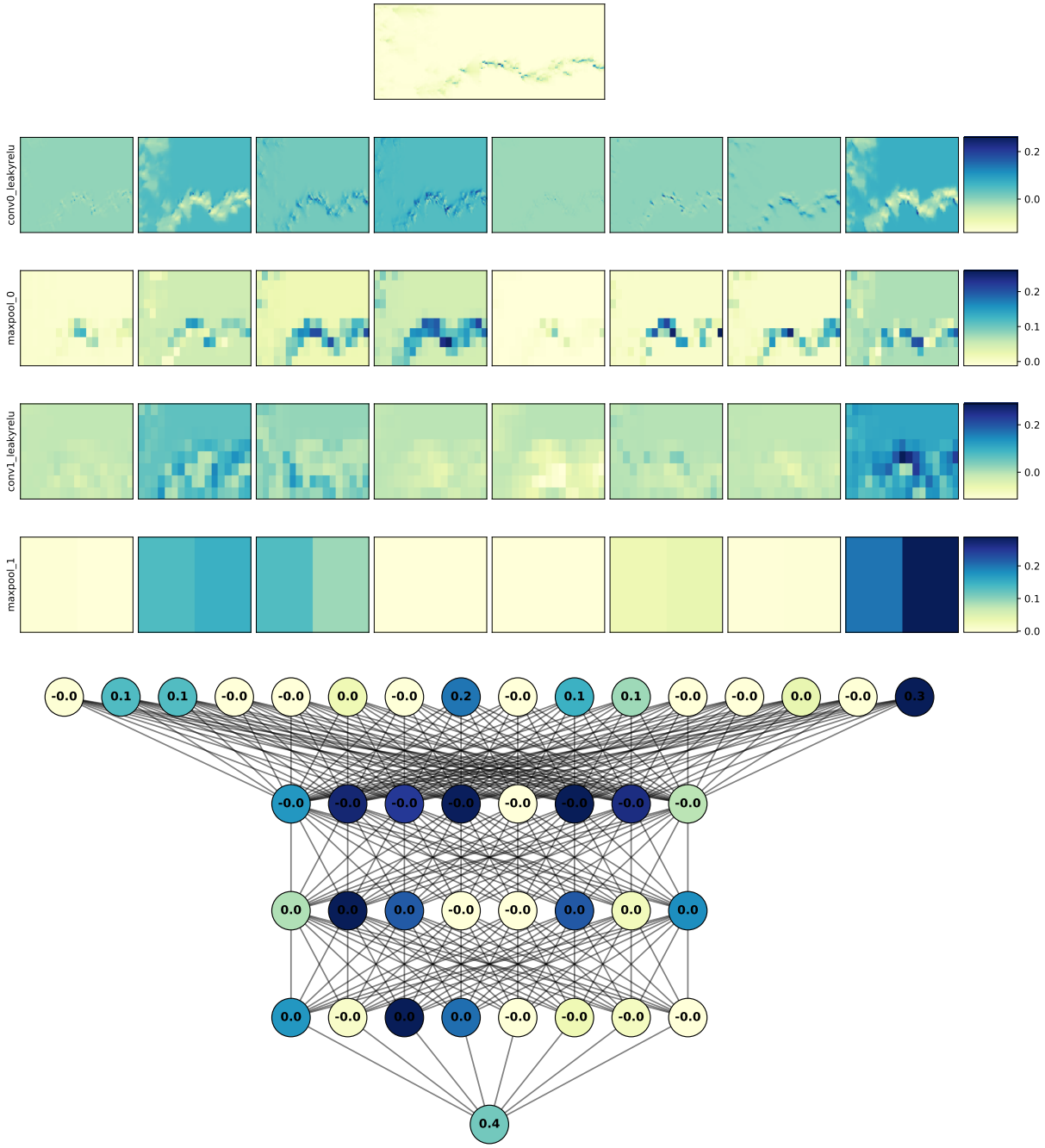


Figure 4.18: This visualisation of the Viterbi map CNN shows the input of a Viterbi map where the time-frequency spectrogram contains an instrumental line. Here the output neuron has a value of 0.4, far below that in Fig. 4.17.

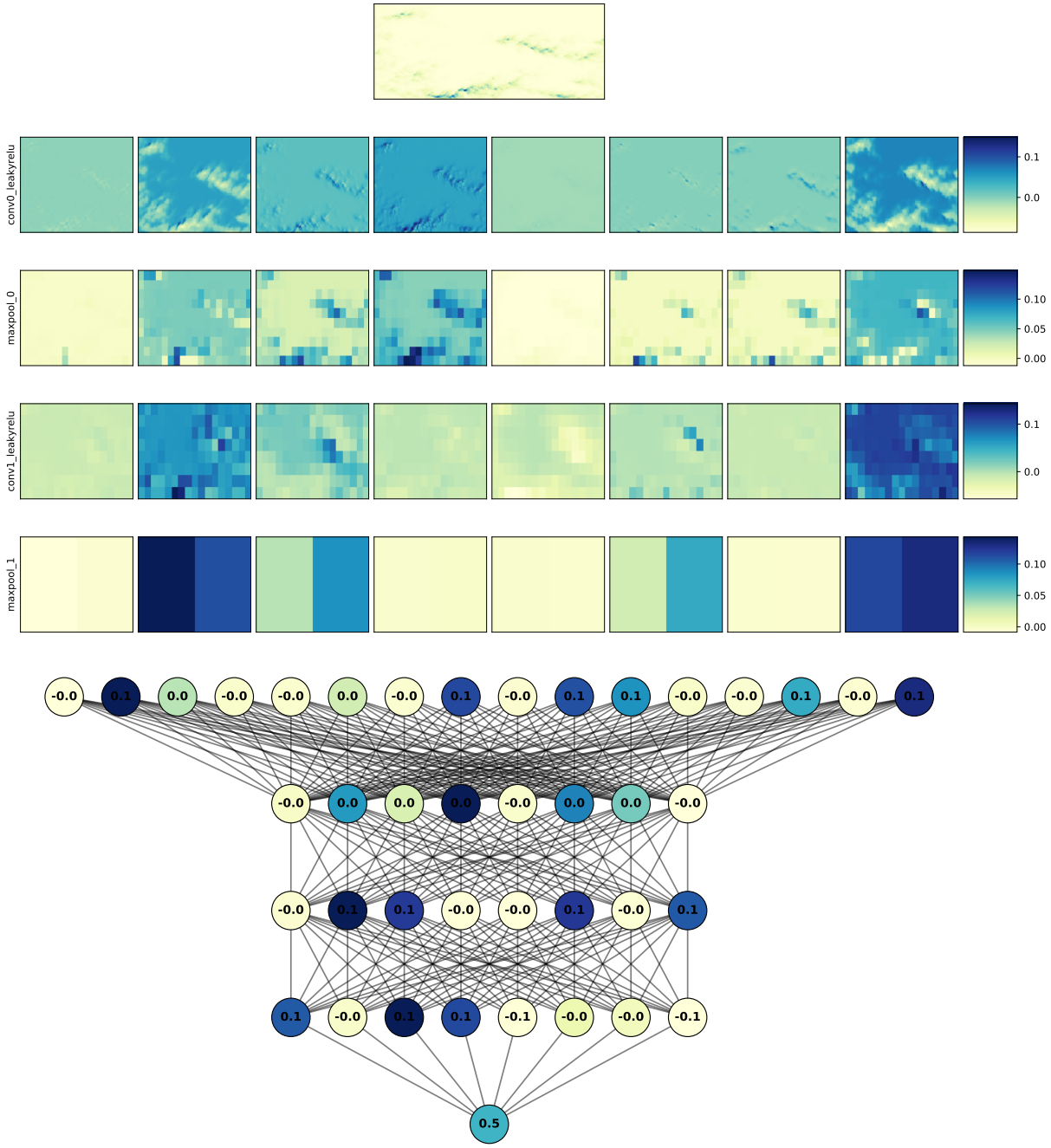


Figure 4.19: This visualisation of the Viterbi map CNN shows the input of a Viterbi map where the time-frequency spectrogram contains just Gaussian noise. The output here is similar to the case where an instrumental line is injected, i.e. the output is much less than 1.

4.12 Summary

In this paper we summarise an extension of the SOAP algorithm [75]. The extension makes use of a CNN to limit the effect of instrumental lines in a search for sources of CW. The SOAP search has a number of outputs for a given input spectrogram, in this paper we focus on using two of the outputs: the Viterbi statistic and the Viterbi map. The Viterbi statistic has previously been used as a measure of whether there is an astrophysical signal in a given frequency band as in [75]. The Viterbi maps are output maps with the same shape as the input spectrogram, these however give a value related to the probability that a signal is in any time-frequency bin. The aim of the CNN approach is to use both the Viterbi maps and spectrograms as input images to more effectively classify each frequency band to either having an astrophysical signal or not. This would then remove then need to manually look through frequency bands and remove those which are contaminated with non-astrophysical (instrumental) features.

We tested 6 separate CNNs which take in some combination of the three representations of the input data: the Viterbi statistic, the Viterbi map and normalised spectrograms. The aim of using different input data types is that each would provide a different representation of the same information, this had the potential to increase the sensitivity of the search. The tests found that the CNN which uses the Viterbi map alone as input was more sensitive than any other which used a single data type as input. Each of the CNNs that used a combination of input data types had a similar sensitivity to the Viterbi map CNN, therefore, we concluded that the Viterbi map provides the most useful information when detecting a signal. Given that the main aim of this paper was to reduce the effect of instrumental lines on the SOAP search, in Gaussian noise data (with no such lines), the CNN search should achieve a similar sensitivity to the Viterbi statistic alone. The tests in Gaussian noise with S6 gaps showed that at a 95 % efficiency and a 1% false alarm rate the Viterbi statistic and Viterbi map achieved a sensitivity of SNR 95 and 90 respectively. When the same test was run in real S6 data at a 95 % efficiency and a 1% false alarm rate the Viterbi statistic and Viterbi map achieved corresponding sensitivities of SNR 300 and 120 respectively. This demonstrates that the Viterbi map approach has a much larger effect when used on real data due to the presence of many instrumental lines.

These tests were once again repeated using a standard set of injections into S6 data such that a direct comparison can be made with other CW search pipelines. At a 95% efficiency and a 1% false alarm rate the Viterbi map CNN achieved a sensitivity of SNR ~ 90 and sensitivity depth ~ 14 Hz$^{-1/2}$. We have shown that the SOAP + CNN approach can achieve a similar sensitivity to other semi-coherent CW search algorithms but with a greatly reduced computational cost.

This search also offers a lot of flexibility in the signal type which can be searched, in the above examples the focus is on isolated neutron stars such that a comparison can be

made to other CW searches, however, this search is un-modelled. By changing the input parameters of the search, different signal types can be searched over, and in future work we aim to test its ability to identify other sources of GW such as neutron stars in binary systems. Further to this, we aim to apply a more advances Bayesian analysis to enable parameter estimation of some parameters of the signal. These parameters would then provide crucial information for a deeper followup by fully coherent pipelines.

Chapter 5

Detector Characterisation with SOAP

When searching for GW signals, it is important to understand the origins of noise artefacts in the detector data which does not originate from an astrophysical source. A large fraction of GW search algorithms, including SOAP above, assume that the detectors noise follows a Gaussian distribution. However, the detectors contain artefacts which do not follow this distribution. These artefacts can negatively affect many searches for GWs as they can be easily mistaken for a real GW signal. Some of the potential sources of these artefacts have been mentioned in Sec. 1.3.1. There are many different classes of artefact, including: glitches, which are short duration broad band bursts in power and instrumental lines, which are long duration narrow-band signals. To conduct a reliable search there are two main tasks which are necessary for detector characterisation. The first is identifying the artefact such that any search knows which frequency bands and time segments are contaminated. The search can then address that section of data, this could mean removing that section of data or use more sophisticated techniques to deal with the artefact [108]. The second task is to find the source of the artefact. If the source of the artefact is found, it can potentially be removed or limited for future data runs.

The focus of this chapter is on instrumental lines and how they affect CW searches. Sec. 5.1 will introduce different sub-classes of instrumental line and how each of them affects a CW search. Sec. 5.2 will outline how these artefacts are detected and monitored, and describe current tools used for this task. Sec. 5.3 will describe how the CW search algorithm introduces in Sec. 3 can be used to search for instrumental lines. Finally Sec. 5.4 will show the outputs of the search and this is displayed for ease of use.

5.1 Instrumental lines

Instrumental lines have the general structure that they are persistent noise artefacts. There are many classes of instrumental line spanning a range from narrow, fixed frequency spectral artefacts to broader (< 0.1 Hz) features which have a time varying frequency

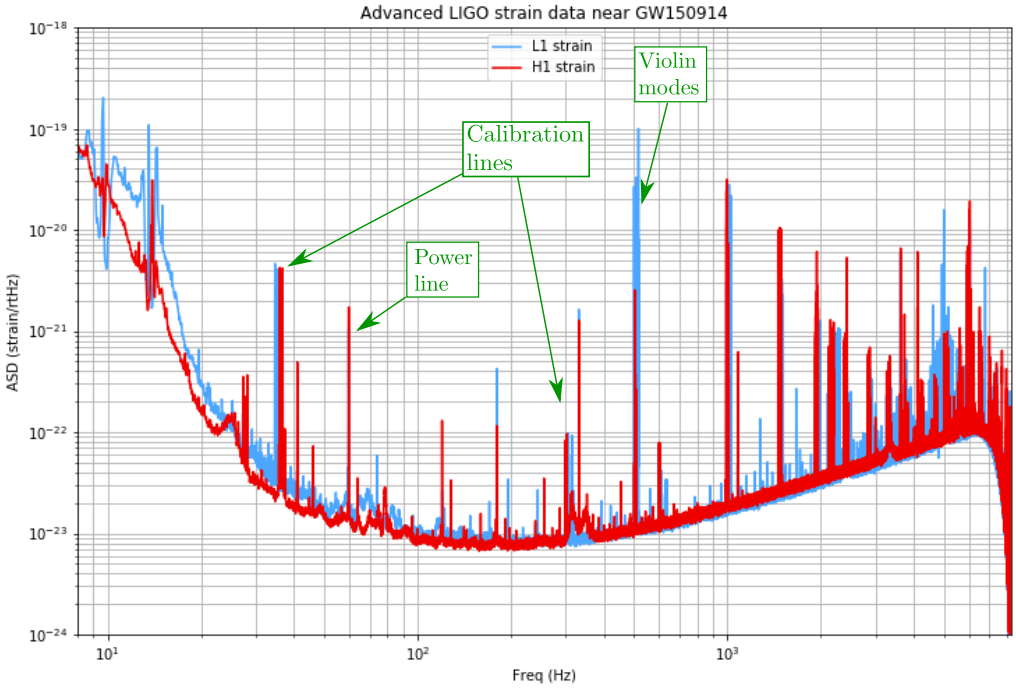


Figure 5.1: There are many features in the LIGO detectors averaged ASD. This image was taken from [109] where I have annotated some of the stronger lines. The power line from the mains in the USA is at 60 Hz. Some of the calibration lines are around 30 Hz, 331 Hz and 1083 Hz. The various violin modes of the suspensions are at 300, 500, 600 and 900 Hz where I have only marked the 500 Hz mirror suspension modes [109].

known as wandering lines. For many of these lines, it is difficult to distinguish them from an astrophysical signal. They affect search methods in two main ways. They can cause the search to produce outliers which are then considered as GW candidates. Extra efforts then have to be made to analyse these outliers further. If the line is close to the GW signal in frequency, then it can conceal the power of the GW, or if it is overlapping, the search becomes very difficult. It is therefore crucial to understand the structure and origin of these lines when performing a search for GW, specifically CW and stochastic searches **JOE: explain why stochastic**.

Some instrumental lines are clearly visible when looking at a amplitude spectral density (ASD) or PSD of the LIGO detectors. Figure 5.1 shows the ASD for LIGOs Hanford and Livingston detectors during their first observing run (O1) [109]. This clearly shows some peaks which are associated with strong lines, some have been labelled. There are however, many more weaker line which become visible when spectra are averaged over longer times. The ASD in Fig. 5.1 shows the time averaged spectra the GW channel of the LIGO detectors. The GW channel is the error signal on the mirrors whilst they are being held

on a resonance, i.e. how much the mirrors have to be moved to keep the same intensity at the output of the interferometer. The lines seen in the spectrum are not from any GW and are usually from ground based sources. To see the lines in the GW channel, they must be coupled in via some mechanism. There are a number of ways in which this happens which are outlined in [110]. This includes coupling via shared power and grounds. When different components share the same power supplies, if a component draws power with a given period, then the voltage will decrease repeatedly at this frequency. Another component which shares this same power supply can then also see this drop in voltage and this can potentially become visible in a recorded output. Another mechanism is coupling through magnetic fields, this is common when cables are close to each other, the magnetic field in one can affect the other, therefore, coupling noise between different systems. Coupling can also occur though a physical connection, known as mechanical coupling, for example the resonances of the suspension fibers which couple directly into the mirrors.

Many of the spectral lines seen in the frequency spectrum, Fig. 5.1 are fundamental to the design of the detector. These cannot be eliminated, therefore need to be understood such that their effect on searches is minimised. Some of the strongest of these lines are listed below:

Power line The power line harmonics are fundamental to the detector and originate from the mains supply in the USA. These lines exist at 60 Hz which is the frequency of the mains alternating current [89]. European detectors Virgo and GEO have a power line at 50 Hz instead of 60 Hz.

Violin modes The violin modes are associated with the suspensions of the mirrors and the beam splitter in the detector. These are designed to have a narrow frequency spectrum such that they contaminate as small a part of the spectrum as possible. These are the lines around 500 Hz for the mirrors and 300, 600 and 900 Hz for the beam-splitter [109] in Fig. 5.1.

Calibration lines The mirrors of the LIGO detectors, are held at a resonance of the cavity in the arms of the interferometer. This then requires a feedback loop to hold them at resonance as a GW passes the detector. Calibration lines are used to calibrate this feedback loop such that the arm length changes are accurate [111, 90].

In [112], techniques are used to counter the affect of power lines and calibration lines on searches.

Along with the fundamental lines of the detector which cannot be removed at the source, there are a large number of other lines whose source has been found and can be removed. Many of these are from mechanisms described earlier such as shared power

supplies or grounds. These can be removed by, for example, using a different power supply for different systems. See [110] for a full investigation into the mitigation of these lines.

These lines have large effect on all searches for GWs both if the astrophysical signals frequencies overlap with the frequency of the line and can cause outliers. Long duration searches for CWs are particularity sensitive to this type of artefact. As described in Sec. 2, CWs are long duration signals with a slowly varying frequency. In the case of an isolated neutron star, the signal which is searched for is narrow-band and a fixed frequency which is Doppler modulated by the earths rotation and orbit. For certain areas of parameter space, such as close to the poles, the astrophysical signal of an isolated neutron star can appear very similar to a narrow band fixed frequency instrumental line. Many of these lines can be mitigated by using multiple detectors data. If a signal appears in a one detector and not the others, then it is likely that the signal is from an instrumental line and not an astrophysical source. These contaminated frequency bands can either be removed or a statistic similar to that described in Sec. 3.8 or [86] can be used to limit their effect. However, there are many examples of instrumental line which appear at the same or similar frequencies in multiple detectors. These pose a real challenge to some CW searches, and require a lot of investigation to limit their affect.

5.2 Identifying and monitoring instrumental lines

JOE: mention wiki page somewhere, eg : https://wiki.ligo.org/CW/O3aLineCleaningI

When a detector is running, it is very important to identify instrumental lines and monitor them. This can then lead to either locating the origin of the line such that it can be removed, or allowing it to be flagged for other search algorithms. The lines affect searches in the GW channel, this is the output of the detector which GW are observed and is the data using is previous chapters. This is the channel where the affect of lines is intended to be minimised.

As well as the GW channel, the detector records many different channels known as auxiliary channels. These channels monitor many components of the detector, and importantly are not sensitive to GWs. Many of the channels useful for line searches are the physical environment monitors (PEMs). These include sensors such as seismometers measuring the ground motion, temperature sensors, magnetometers etc. These channels can be very useful in identifying the source of an instrumental line. The main goal is to reduce the number of artefacts in the GW such that it is as close to Gaussian noise as possible. If an artefact shows up in the GW channel in coincidence with one of the PEM then this is an indicator that the artefact originates from something related to that PEM. For example, say a magnetometer located near some electronics has a long duration line in its spectrum at the same frequency as the main GW channel. This indicates that

noise from this piece of electronics is somehow coupling into the detector. One can then investigate that piece of electronics further to see how it couples in.

There are a number of tools which are used to monitor these spectral lines, along with a team of people which regularly look though the results, a summary of these for the first two observing runs of LIGO can be found in [110]. Some of the tools used are described below.

Fscan Fscan [90] takes FFTs of the raw detector data, typically these are 1800s long. This is done for all of the auxiliary channels as well as the GW channel. The FFTs are then averaged over a day and time-frequency spectrograms are generated. After known lines such as Violin modes and power lines are subtracted, noise lines can be identified. A threshold can be set and anything in the spectrograms which exceed this threshold are flagged as a line. These can then be compared across multiple different channels. More detail on how the lines are identified can be found in [90].

Coherence Coherence searches for the coherence between different channels and different detectors. This is similar to searches for stochastic gravitational waves. More detail of how this works can be found in [110].

Finetooth Many of the instrumental lines found are part of combs. These are repeating structures where the harmonics (teeth) of a line make up the comb which is defined by the start frequency and tooth spacing. Finetooth is a tool which identifies and monitors these combs [113].

NoEMi NoEMi uses FFTs and identifies peaks within the spectrum. It can then find coincidences with auxiliary channels and label is there are overlapping peaks with the GW channel. This can then tracks the lines with time. All of the information is stored in a database where more detail on its operation can be found in [114].

These tools offer many ways for the detector characterisation team to identify instrumental lines and hunt for their source. A summary of these efforts for the advanced LIGO data can be found in [110], or the LIGO wiki page `https://wiki.ligo.org/DetChar/O3LinesCombsInvestigations`. The following sections describe how the SOAP search described in Sec. 3 can be used as an extra tool to aid in the identification and monitoring of instrumental lines.

5.3 Identifying and cleaning lines with SOAP

The SOAP search has been run on a number of observing runs to search for CW. One of the major factors which limited the sensitivity of the search is the presence of instrumental lines within the data. Many of the potential candidates which SOAP returned could be identified as an instrumental lines An example of once of these candidates can be seen in Fig. 5.2. Figure 5.2 shows a broad and wandering line in a single detector causing the SOAP search to mistake it for an astrophysical signal. This is because the SOAP line aware statistic from Sec. 3.8 find areas of higher power which are consistent between detectors according to the parameters of the statistic. These types of line are difficult to mitigate in the astrophysical search, however, this has a side effect of being useful to identify the instrumental lines themselves. In this section, I will explain the setup of the search to identify instrumental lines.

Whilst it is useful to run searches on auxiliary channels when trying to identify the lines source, the aim of this search was to flag potential lines. Therefore, this search currently only runs on the GW channel where they flagged lines can be investigated further. In the future this could be modified to search multiple channels to find coincidences between them. Sec. 3 described how multiple detectors can be used to increase the sensitivity of a search for CWs. However, when searching for instrumental lines, the aim is to have the reverse effect, therefore, a simple search is run separately on each detector. This then removed any of the statistics developed in Sec. 3.8 and uses the 'normalised' SFT power as the statistic in the SOAP search. The single detector search then has one parameter to vary, the transition matrix parameter. This governs how probable the frequency track is to transition up straight or down a frequency bin. In this search we are aiming to find any non Gaussian artefacts, therefore, we allow an equal probability for the track to jump in any direction, however, limit it to change by one frequency bin after each time segment. Whilst the astrophysical search summed the SFTs over one day as shown in Fig. 5.2, for the line search, we assume that instrumental lines are stronger than astrophysical signals and therefore should be identifiable in the raw 1800s long SFTs. The Fscan search described above generates SFTs every day of varying lengths, this include 1800 s long FFTs. For this line search, we split the 1800 s long Fscan FFTs into 0.2 Hz wide sub-bands and run the single detector search on each sub-band. This then returns the same outputs as described in Sec. 3 and Sec. 4: the frequency track (Viterbi track), a Viterbi map and a Viterbi statistic. Here the Viterbi statistic is just the sum of the SFT power along the frequency track. The line search then outputs plots as shown in Fig. 5.3, 5.4, 5.5 and 5.6. These plots and the equivalent plots for other sub-bands can then allow each sub-band to be classified into containing an instrumental artefact or not. An example of the SOAP line search being run on the H1 detector on the same instrumental line but with slightly wider frequency band as Fig. 5.2 can be seen in Fig. 5.3 This shows how the line search

identifies the same instrumental line albeit at a higher resolution and can flag it for further investigation.

Fig. 5.4 shows an example of the spectrogram of a sub-band which does not contain any signal or spectral artefacts but is Gaussian distributed noise. The track in the top panel of this plots shows the power spectrum including detector gaps from 1800 s SFTs in LIGO Hanford's data from its second observing run (O2). The Viterbi track is overlaid and the structure of the track indicates that there is no signal present, this is because the track appears to be randomly wandering and has a large spread over the entire band. The Viterbi map plot does not show much structure and contains areas of low log-probability. Using this information along with the Viterbi statistic result, this particular sub-band can be considered to not contain an instrumental artefact.

Fig. 5.5 demonstrates the same figure as before but now with a strong and narrow instrumental line in the sub-band. The Viterbi track now clearly indicates that there is a narrow spectral artefact present in the sub-band. The track does not have much spread over the band and stays at an approximately fixed frequency. The Viterbi map plot then shows that there are clear areas around the track which are of high log-probability. The areas of white in the Viterbi maps area areas where the probability of a signal falls to zero. These pieces of information along with the large value of the Viterbi statistic indicate that there is a narrow spectral artefact within the sub-band.

Fig. 5.6 once again shows the equivalent plots to Fig. 5.4 and 5.5 but now contains some wandering spectral artefact. This is some line which wanders in frequency as is moves though the band. This can be seen in the frequency track, which here does not have much spread, however, the frequency of the potential signal changes with time. There are also areas where the signal could have potentially switched to a separate spectral artefact within the same band. Fig. 5.6 shows this discrete jump around January.

The combination of the Viterbi statistic, Viterbi map and spectrograms are then used to flag this sub-band as containing some sort of instrumental artefact. The example plots here show some specific examples, however, there is a lot of variation in the types of lines and features which appear in all of the SOAP outputs. This line search then works by initially ranking all of the Viterbi statistics, the largest values will be most likely from an instrumental artefact. Figure **??** shows an example of a histogram of the ranked statistics from O2. The sub-bands which give statistic values which lie outside the main distribution can then be viewed for further investigation. The high values outside the distribution are currently defined rather arbitrarily, this could either be something like the top 10% of sub-bands or could involve looking through the ranked list until signals begin to look like noise. The top results which are deemed to come from an instrumental line can be compared to the known line lists from the other line tools mentioned in Sec. 5.2. Any new lines can be investigated further by methods described in [110] using the tools in Sec. 5.2. A list of lines

which have been categorised by these searches can be found here in https://ldas-jobs.ligo-wa.caltech.edu/~evan.goetz/CW/O3aLines/H1/index.html and lines which have not been categorised in https://ldas-jobs.ligo-wa.caltech.edu/~evan.goetz/CW/O3aLines/H1/Unidentified/index.html. The two line lists above can then be compared to the output of the SOAP line search.

When running this search SOAP identified lines which did not appear in other line lists, therefore, offers a method to search for weaker lines. **JOE: actually check this**

JOE: need to mention why this is good as extra tool, i.e. searching for wandering lines, identifies track of very weak lines etc, things that other searches cannot do

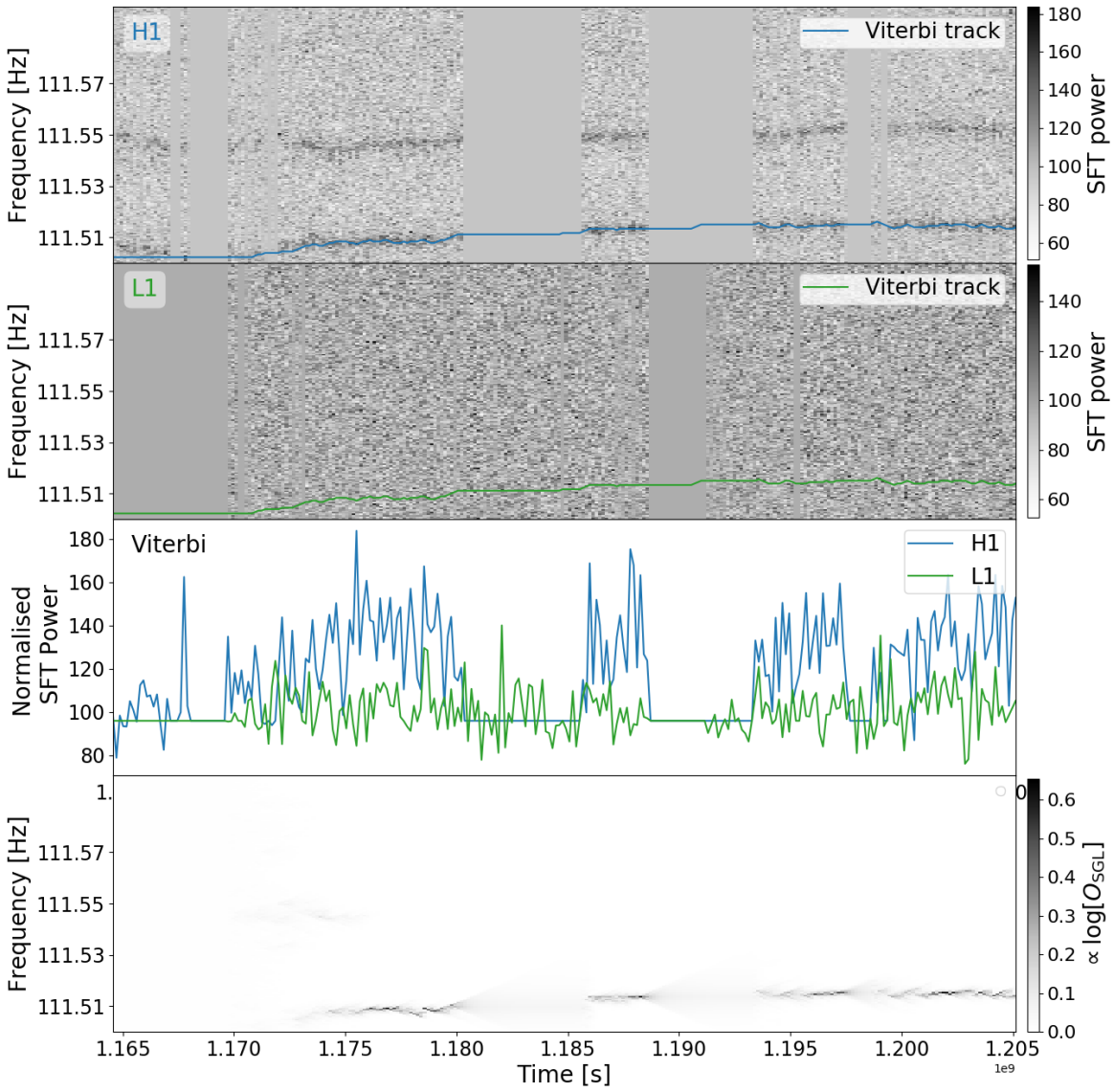


Figure 5.2: Broader instrumental lines are generally weaker and can be mistaken for an astrophysical signal by the SOAP astrophysical search. Above an example of two broad line in the H1 detector is shown, where SOAP has identified one of the. The top two panels show the summed spectrograms from the H1 and L1 panels with the optimal Viterbi track overlaid. The third panel shows the normalised spectrogram power along the Viterbi track for each of the detectors. The final panel shows the Viterbi map for this frequency band.

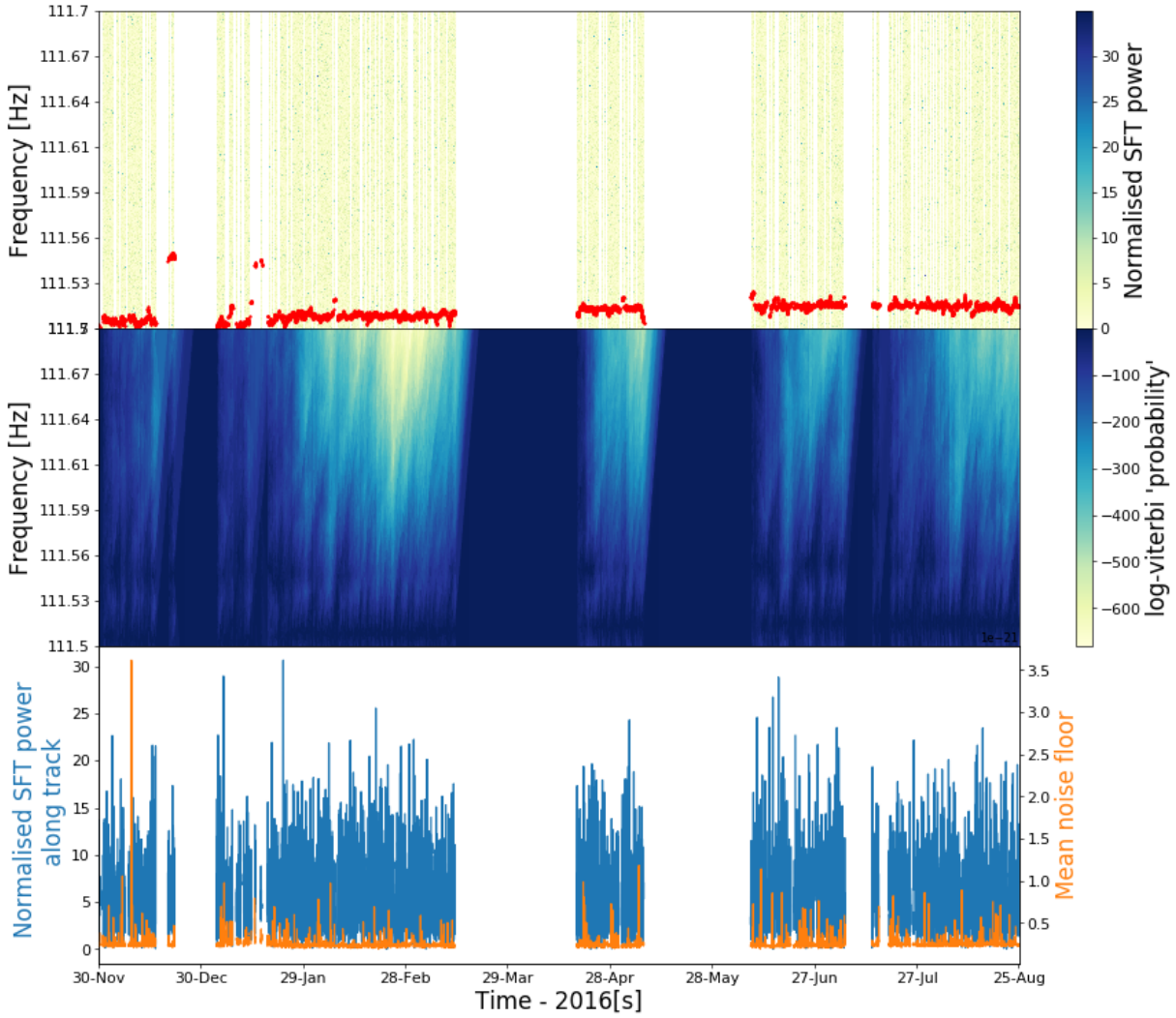


Figure 5.3: The SOAP line search is run on a similar frequency band to Fig. 5.2. This returns a similar track however at a resolution of 1800 s rather than one day.

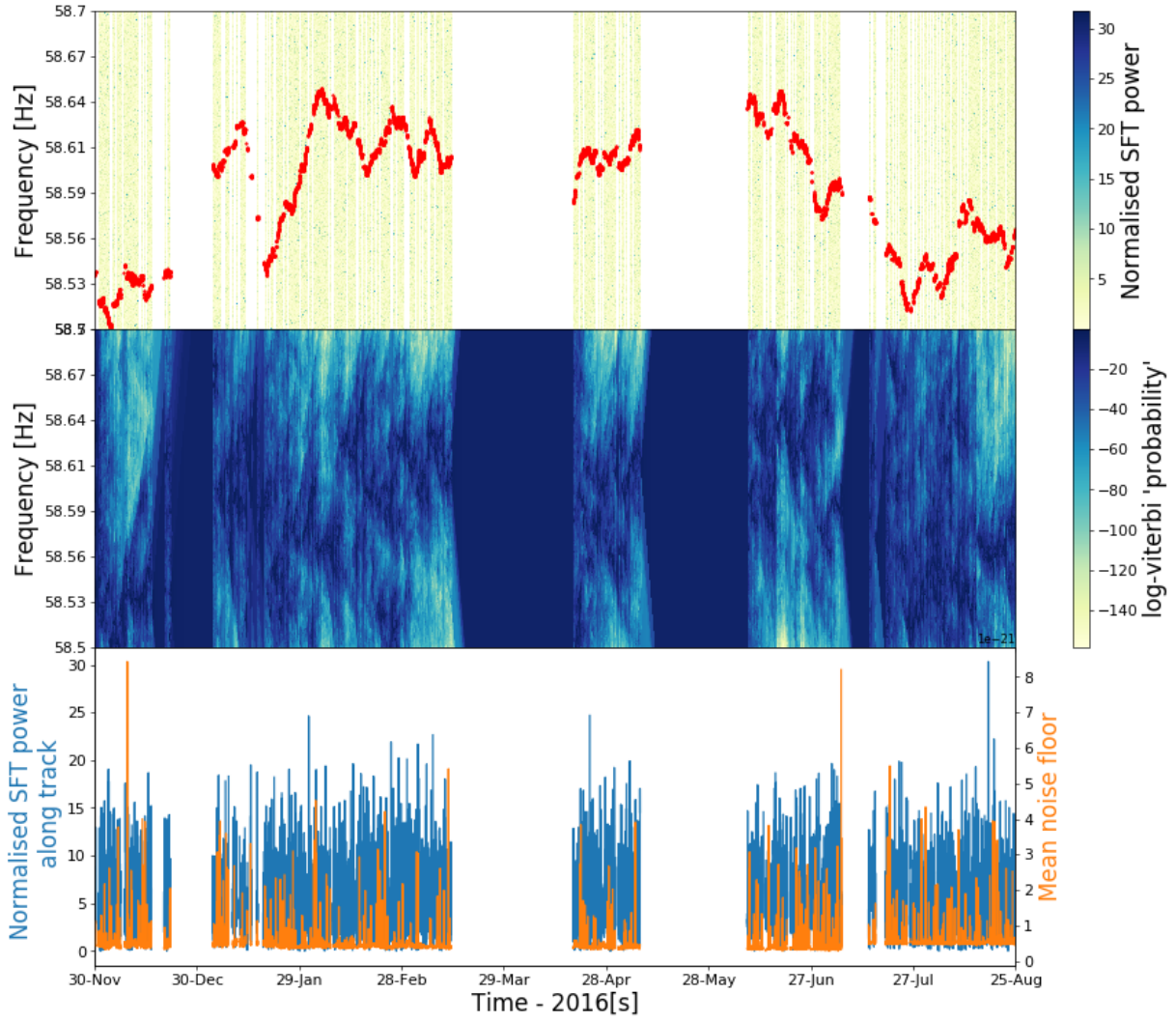


Figure 5.4: The SOAP search outputs three main quantities, the Viterbi maps, the Viterbi track and the Viterbi statistic. The Viterbi track is shown above overlaid onto the 1800s SFT power spectrum including the detector gaps for LIGOs Hanford detector (H1) in its second observing run (O2) []. This track is an indicator as to what type of signal the track is following. The above track indicated that this is just noise. The returned Viterbi statistic is also consistent with that of noise. The Viterbi map is another visualisation of the sub-band, how to interpret this has been explained in previous sections. However, here there does no appear to be a clear signal. The final panel is a way to visualise how the SFT power changes along the Viterbi track. Also on this plot is an estimate of the mean noise floor for this band to visualise how the sensitivity of the detector changed over the course of the run.

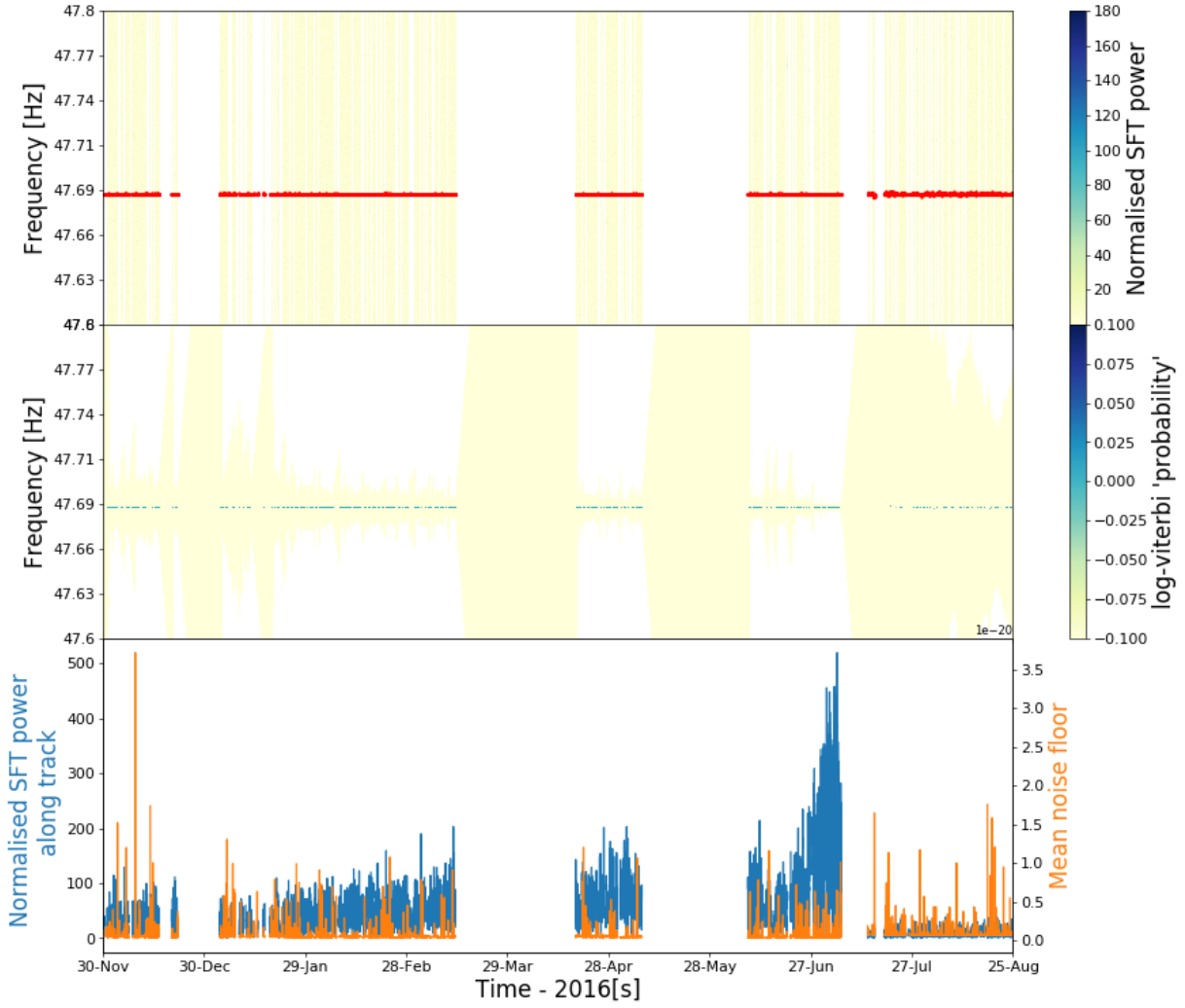


Figure 5.5: The equivalent plot as in Fig. 5.4 can be made when there is a narrow spectral artefact in the band. The above is again results from LIGOs Hanford detector (H1) in its second observing run (O2) using 1800s SFT power spectrum. In this there is a narrow spectral line at $\sim$ 47.69 Hz. The Viterbi track then follows this line of high power. The Viterbi map has much higher values for the log-probability in this line case compared to the noise case, this is an indicator some real signal. The probability in the Viterbi maps drops to zero in some areas due to the strength of the instrumental line.

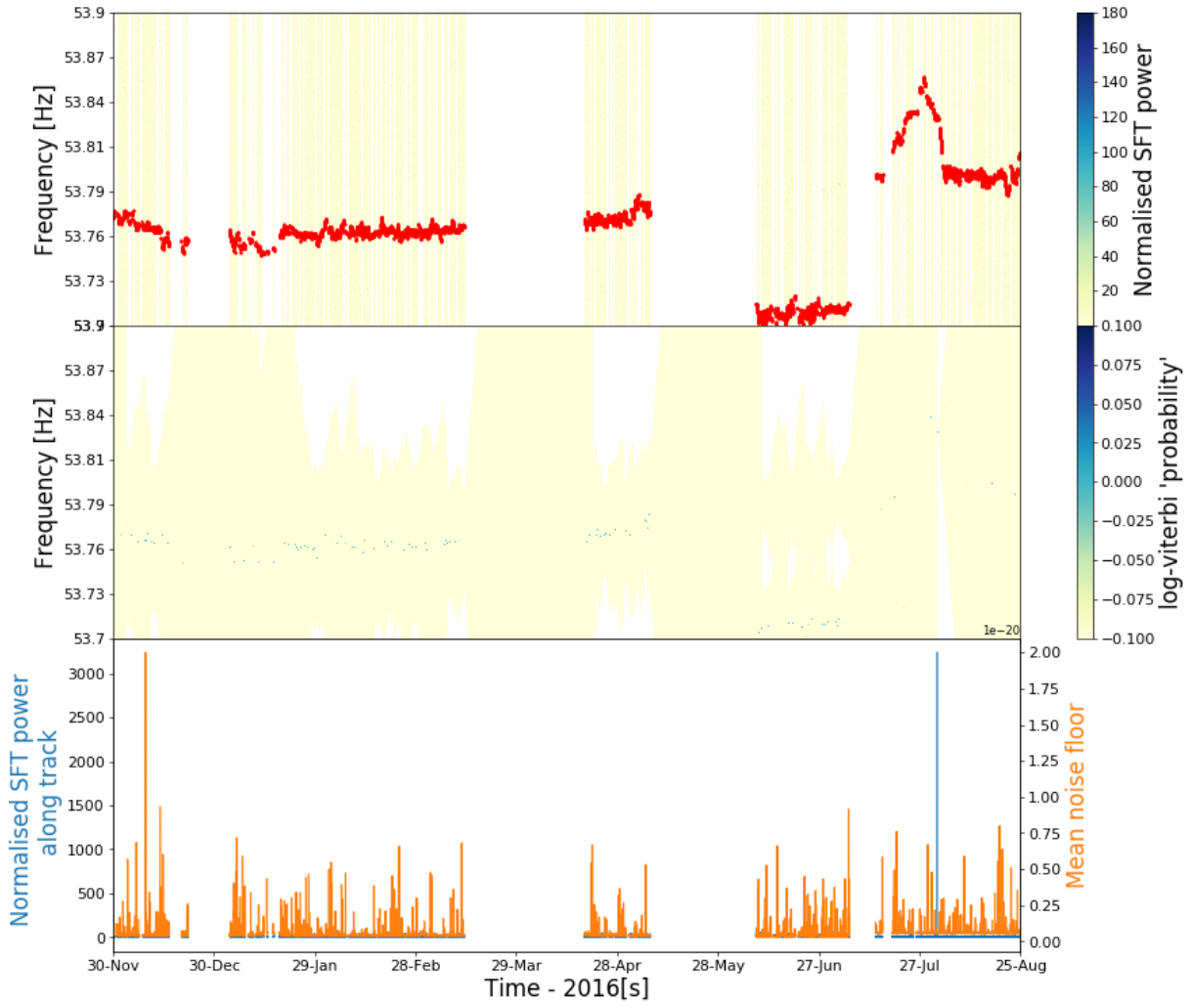


Figure 5.6: The equivalent plot as in Fig. 5.4 can be made when there is a wandering spectral line. The above is again results from LIGOs Hanford detector (H1) in its second observing run (O2) using 1800s SFT power spectrum. This shows how some spectral lines do not have a fixed frequency bin can wander through the band. These are especially hard to track and monitor. The Viterbi track here shows is clearly different from the noise case in Fig. 5.4 as the track is more tightly concentrated around some areas of power.

5.4 Summary pages

Summary pages are an important tool when searching for instrumental lines. There is such a large amount of data both in frequency and time space, and channels to search though when looking for instrumental lines. Summary pages distill this data such that only the important information is shown. This enables line to be identified easily when looking through sub-bands. The criteria when designing summary pages is that they are easy to navigate and the important information is shown in a clear and concise way. How we display this information will be explained later in this section. These summary pages exist for the above searches in [115] where this is only accessible by LIGO members.

For the SOAP search summary pages were generated for each observing run and for the two LIGO detectors. This was done for various timescales: for the entire observing run and separately for each month. This allows the variation of a line to be observed for the entire length and also artefacts on shorter timescales to be observed. Once the detector, observing run and timescale is set, the band is split into 0.2 Hz wide sub-bands. The Viterbi search with a flat transition matrix and using the summed SFT power as the statistic. A flow diagram of how the SOAP search works for instrumental line searches can be found in Fig. 5.7. These stages are as follows:

1. **SFTs from time series** The SFTs are generated for the GW output channel. This is done by the Fscan search, therefore we do not repeat this process. Currently the search only runs on the GW channel, however, in the future could be made to run on others.

2. **Divide SFT by running median** In this stage each SFT is divided by its running median which is 100 bins wide. The running median takes each SFT and applies a window of 100 bins, where the median of these 100 frequency bins is taken. This window then slides over the SFT producing an 'filtered' SFTwhich should exclude outliers.

3. **Narrow-band SFT** The SFT is then split into 0.2 Hz wide sub-bands for the SOAP search to run on. These smaller bands are chosen as the SOAP search pull information on the most likely track, therefore, smaller bands are not contaminated by areas of high power in neighboring frequency bands.

4. **Run SOAP and generate plots** This stage runs the SOAP search with a flat transition matrix probability and generates plots as shown in Fig. 5.4.

5. **Generate summary page** Finally the summary pages are built which take all of the bands and puts them in a table. This table can be ordered by the value of the Viterbi statistic, or can be searched for particular frequency bands.

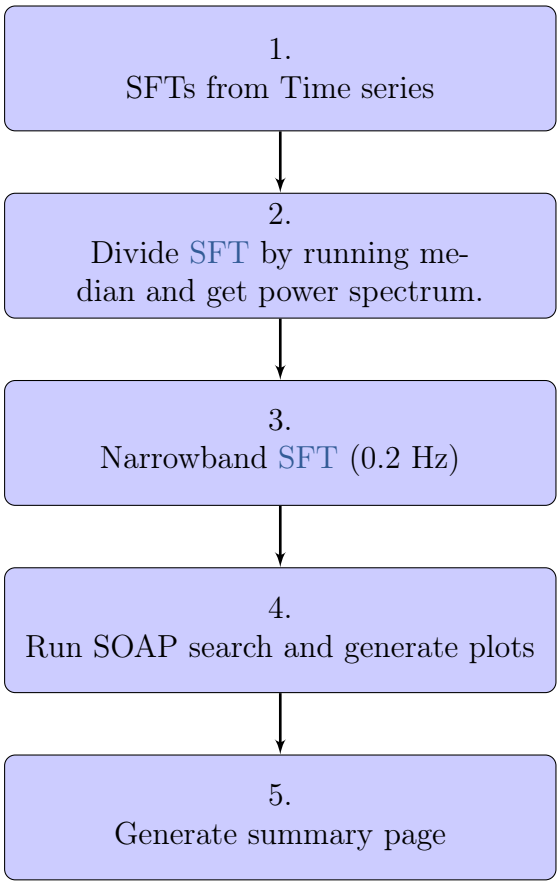


Figure 5.7: The SOAP search for instrumental lines is simpler than other searches. A simple version of the search is run separately for each detector, where the raw SFTs are divided by their running median, narrow-banded and then the search is run.

An example of a summary page is shown in Fig. 5.8. This has been annotated showing how to navigate the page. There are generally two separate parts to the page: selecting the observing run and frequencies, and viewing the outputs. The observing run is selected at the top of the page, where currently this has only ben run on O2 and O3. From this menu the detector can be selected, currently only LIGOs H1 and L1 detectors are present. The selection of frequencies and viewing of outputs then happens on this page. The key information of each page is the plots shown in Figs. 5.3 - 5.6. These clearly show the data with a number of representations of the data. They show the time frequency spectrograms of the data, the output Viterbi tracks which identify the most probable frequency track, the Viterbi maps which allow the probability of a signal as a function of the time and frequency bin to be viewed. Finally they show the spectrogram power along the Viterbi track with the mean noise floor of the detector during the observing run. This should provide enough information to determine whether an instrumental line in present. To navigate each page, the left panel contains a calendar where the start and end times of a result can be selected. Currently there are pre-defined times which can be selected from. This allows a line to be investigated for a shorter or longer time period. This can be useful when a new instrumental line appears and it needs to be investigated only from when the line appears. Below this in the left panel of the page there is a table where each cell is one of the sub-bands which was searched through This allows individual frequency bands of interest to be searched for as well as the table to be limited between different frequencies. The table contains four columns: the frequency of the sub-bands, the Viterbi statistic, the σ from the mean of all the sub-band Viterbi statistics and extra information. The first two columns are self explanatory, the frequency range of the sub-band and the resulting Viterbi statistic from that sub-band. The table can be ordered by the Viterbi statistic such that only the highest values are investigated. The σ from the mean is found by approximating the distribution of the Viterbi statistics as a Gaussian. A Gaussian is then fit to the distribution using a simple least squares, each statistic then has a multiple of σ away from the mean of this distribution. This is an approximate calculation to give a scale of how significant the statistic in that sub-band is. The final column contains any extra information which exists for that particular frequency range. For example, this is filled with other line list information which has been collected from other search methods. The loud features such as Violin modes can then be marked. This means that these particular bands are likely to have been investigated already allowing this search to focus on any extra instrumental lines.

The summary pages are then hoped to be useful alongside the other tools for searching for instrumental lines. **JOE: more**

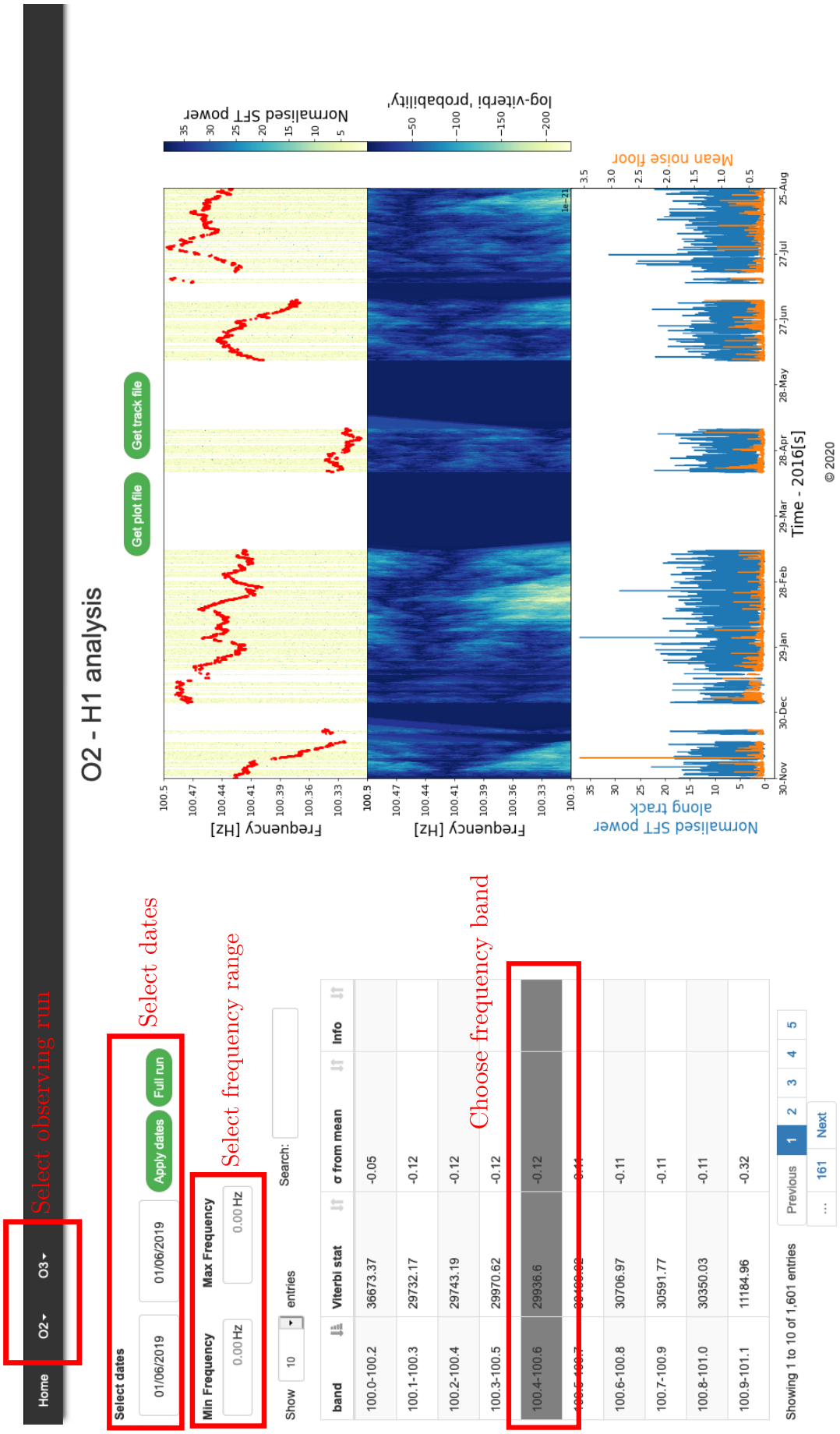


Figure 5.8: The summary pages are made for each observing run (in this case just O2 and O3). The range of times can then be selected from a set of start and end times. This is in general the entire observing run and monthly runs of this search. These pages can be found at [115]

5.5 Armadillo

Armadillo was a project which aimed to develop a diagnostics tool to be used at the LIGO detectors.

- LIGO has large control systems
- these can be generally split into two parts: Analogue and digital control systems
- Armadillo was a tool that focusses on the digital control system
- Having an understanging of what happens to a signal as it passes through the control system is important it can help identify source of glitches
- one way to do this is to look at the transfer funtion of a possible signal path
- Armadillo aimed to find a simple way to return the transfer function between any two points in the control system.

To find the transfer function between any two points in the digital control system, one needs to know all the possible paths a signal could take between those two points and each of the component filters which lie along the paths. The models of the digital control system for LIGO are stored in Simulink diagrams []. These generally

- May components are need to explain a transfer function
- first step is to load all the models from simulink diagrams
- these contain of many blocks and connections see Fig ....
- each block can be a single filter or a submodel which contains many more filters
- there is then a large number of filters and possible paths
- for each filter, one needs to load the 'FOTON' filter files which contain the coefficients
- there are also blocks which are matricies which only allow a signal to pass along a certain path at a given time
- These are defined int EPICS by the user, and their values at any time are stored
-

Chapter 6

Summary

This thesis outlines the current state of searches for CWs, and describes new techniques which have been developed to address some of the challenges in this type of search. Whilst CWs have not yet been detected, they are expected to originate from rapidly rotating neutron stars which are not symmetric around their rotation axis. The expected signal is a quasi-sinusoidal signal which lasts for times longer than LIGOs observing runs. The large observation times associated with this signal mean that methods that search for these signals need to run on large amounts of data. For many of these searches this large observations time along with the large parameter space, mean the computational time needed to complete a search is large. Sec. 2 outlines some of these methods and highlights the needed computing time. The new methods described in this thesis address the challenge of reducing the computational cost of searching for CWs.

Chapter 3 describes a search algorithm entitled SOAP which is un-modelled. This is based on the Viterbi algorithm which was designed to find the most likely set of states through a discrete Markov process. This has been utilised to search through time-frequency spectrograms for tracks in frequency which could originate from an astrophysical signal. Here the different states are the frequency bins in each given time segment. The algorithm can be constrained such that any track can only change by a given number of frequency bins at each time segment. This is governed by a 'transition matrix' described in Sec. 3.3 and can focus the search on an expected astrophysical signal. The search then returns the frequency track which gives the highest value for the sum of some statistic. There are a number of statistics which were developed, the simplest being the FFT power in a given frequency bin. This search though a single time-frequency spectrogram was then extended to search through multiple detectors, i.e. multiple time-frequency spectrograms. An astrophysical signal should have a similar high FFT power in both detectors, therefore, the search was modified to look for consistent high powers. This simple statistic of the FFT power encountered problems when frequency tracks of high FFT power originated from instrumental affects as opposed to astrophysical ones. Specifically, the search is con-

taminated with instrumental lines which are long duration narrow band spectral artefacts. In an attempt to mitigate the effect of these instrumental lines, a Bayesian statistic was developed in Sec. 3.8. This effectively down-weights FFT powers which appear to be from instrumental lines, i.e. very high values or values which are high in only one detector, and rewards similar low values of the FFT power. SOAP then searches for consistent SNR between multiple detectors. However, these multiple detectors can have different sensitivities, therefore, the SNR of the same astrophysical signal can be different in each detector. In Sec. 3.9, the Bayesian statistic was modified to search for consistent signal amplitudes, i.e. consistent values of h_0, by incorporating the detector noise floor and fraction of observation data in each segment. These statistics had a set of parameters which were optimised to injections into each data-set. The statistic increased the sensitivity of the network when it is run of real LIGO data. **JOE: write section in body about this**

SOAP was then tested on three main data-sets containing simulations of CW signals form isolated neutron stars. The data-sets include: Gaussian noise, Gaussian noise but with temporal gaps corresponding to time the detector was off in LIGOs S6 data run, and in real S6 data, this was from a standard set generated to compare CW searches sensitivity to isolated neutron stars. In this test we achieved a sensitivity similar to other CW searches. **JOE: more here** However, the computational cost of this search is orders of magnitude less than others. SOAP is also not limited to search for isolated neutron stars, as it searches for track of high power, it is mostly un-modelled and can search for many signal types. Sec. 3.13 shows an example of the search identifying the first BNS signal. SOAP still has limitations however, the line aware statistic reduced the affect of instrumental lines, however, many contaminated frequency bands were still manually removed in the analysis. Sec. 4 aims to address this problem.

There are a number of additions which we aim to add to this search in the future. For example, there are additional statistics, such as using the Fourier transform of the detector power along the track in the SFTs. If an astrophysical signal is present then the effects of the antenna patters should be seen. Future work on this also includes using the output of the SOAP search to estimate parameters of the source.

The next piece of work aimed to follow on from the SOAP algorithm using machine learning. One of the main challenges in the above search was the contamination of frequency bands with instrumental lines At this stage many bands were manually investigated and removed from the analysis if they were deemed to to be contaminated. This is a time consuming process and becomes impractical when searching over larger bandwidths. Therefore, we needed a way to classify these bands into containing a signal or not. A common tool in machine learning is deep neural networks. These have been used extensively in image recognition and classification. Deep neural networks, specifically CNNs are a tool well suited to the challenge above. Sec. 4 contains details of how neural

networks are structured and how they operate on a given input. This is followed by an explanation of how a CNN operates. CNNs are well suited to the task above as they were designed to take an image as input. The data we are trying to classify in this work is time-frequency spectrograms, these can be thought of as images. A CNN can identify features within this image and extract useful information from them, such as whether an astrophysical signal is present in the data. An key part of using neural networks is training them this is described in detail in Sec. 4.5. Training a network involves showing it many examples of data which is labelled. This means in our examples, that a time-frequency spectrogram which contains an astrophysical signal is labelled to contain a signal and a spectrogram with noise or an instrumental line is labelled as noise. The many parameters of the network can then be updated such that given the set of training data, when any new example is shown to the network it returns a desired value. Training data-sets are generally very large, this allows the weights to be updated without over fitting. We then designed CNNs which took in three main types of data: down-sampled time-frequency spectrograms of LIGO data, down-sampled output Viterbi maps and the output Viterbi statistic. The Viterbi maps and Viterbi statistic are different representations of the time-frequency spectrograms which are output from SOAP. The time-frequency spectrograms and Viterbi maps were downsampled to reduce the amount of data passed through the CNN and speed up the training time. There were then 6 main networks which took these data types and combinations of them as inputs. The networks return a statistic which ranges between 0 and 1, and is used as a detection statistic. Each of the 6 networks was tested on simulations into four data-sets: real LIGO data observing runs O1, O2 ,and S6 and Gaussian noise. In each of these tests the CNN which contributed most to the sensitivity was the network which took the Viterbi map as input. Therefore, for most results this is what was used. This showed that applying a CNN to the output Viterbi maps of the SOAP search eliminates the need to manually remove the contaminated bands. This method achieved the same sensitivity as the SOAP search alone, whilst reducing the time needed for the entire search. In this section a complete comparison the other all-sky CW searches was conducted. A standard set of simulations is isolated neutron stars in LIGOs S6 data-set was used. Where the signals which had frequencies in the range of 40-500 Hz were compared. In this test we found that we sit amongst other all-sky searches, however, can run with a computational time orders of magnitude faster.

The majority of the thesis uses techniques to reduce the affect of instrumental artefacts on the data, however, given that SOAP can identify these features well, we also aimed to use SOAP as a search for instrumental lines. Within the detector there are many sources of noise. Particualr features in the noise which are narrowband can be a large problem for CW searches.

Appendix A

Continuous gravitational wave injections

In this section I outline how we inject a CW signal into data. This can generally be done in two different ways: simulating a signal in the time domain and injecting into time domain noise or simulating the signal in the frequency domain. The searches described in Sec. 3 and Sec. 4 only use output power spectra] Generating the time series and performing a Fourier transform or generating the signal in the frequency domain is time consuming. In this section I will outline how I simulate the power spectrum of CW signals and inject them into a PSD. This should greatly improve the speed of data generation.

A.1 Signal SNR

To inject into a spectrogram the power spectrum of the signal will need to be simulated. In our injection we do not have access to a time-series, therefore, we do not simulate the signal in in the time series or complex frequency domain. Instead, the SNR of the signal can be estimated in given frequency bins and injected straight into the power spectrum.

It can be shown that the PSD of Gaussian noise with zero mean and unit variance is a χ^2 distribution with 2 degrees of freedom. Therefore, if we want to generate a spectrogram for Gaussian noise, we just generate a two dimensional array of values distributed as χ^2 with two degrees of freedom. Assuming that there is some sinusoidal signal with a given SNR within a Gaussian noise time-series with zero mean and unit variance, the FFT power in a particular frequency bin can be estimated using a non-central χ^2 distribution with 2 degrees of freedom, where the non centrality parameter is the square of the SNR. To calculate the SNR in a given frequency bin the equation in [55] for optimal SNR was used

$$\rho(0)^2 = \frac{1}{2} h_0^2 T S^{-1} \left[\alpha_1 A + \alpha_2 B + \alpha_3 C\right], \tag{A.1}$$

where h_0 is the GW amplitude, T is the total observing time is seconds, S^{-1} is the mean PSD noise floor. The values of α are then defined in [55] by

$$\begin{aligned}
\alpha_1 &= \frac{1}{4}\left(1+\cos^2(\iota)\right)^2\cos^2(2\psi)+\cos^2(\iota)\sin^2(2\psi)\\
\alpha_2 &= \frac{1}{4}\left(1+\cos^2(\iota)\right)^2\sin^2(2\psi)+\cos^2(\iota)\cos^2(2\psi)\\
\alpha_3 &= \frac{1}{4}\left(1-\cos^2(\iota)\right)^2\sin(2\psi)\sin^2(2\psi)
\end{aligned} \tag{A.2}$$

where ψ is the gravitational wave phase and ι is the inclination angle of the source. The values of A, B and C in Eq. A.1 are functions which represent the time averages antenna patterns, they are defined by

$$\begin{aligned}
A &\equiv \langle a^2\rangle\\
B &\equiv \langle b^2\rangle\\
C &\equiv \langle ab\rangle,
\end{aligned} \tag{A.3}$$

where a and b are the antenna pattern functions defined in [48] as

$$\begin{aligned}
a(t) = &\frac{1}{16}\sin 2\gamma(3-\cos 2\lambda)(3-\cos 2\delta)\cos[2(\alpha-\phi_r-\Omega t)]\\
&-\frac{1}{4}\cos 2\gamma\sin\lambda(3-\cos 2\delta)\sin[2(\alpha-\phi_r-\Omega t)]\\
&+\frac{1}{4}\sin 2\gamma\sin 2\lambda\sin 2\delta\cos[\alpha-\phi_r-\Omega t]\\
&-\frac{1}{2}\cos 2\gamma\cos\lambda\sin 2\delta\sin[\alpha-\phi_r-\Omega t]\\
&+\frac{3}{4}\sin 2\gamma\cos^2\lambda\cos^2\delta,
\end{aligned} \tag{A.4}$$

$$\begin{aligned}
b(t) = &\cos 2\gamma\sin\lambda\sin\delta\cos[2(\alpha-\phi_r-\Omega t)]\\
&+\frac{1}{4}\sin 2\gamma(3-\cos 2\lambda)\sin\delta\sin[\alpha-\phi_r-\Omega t]\\
&+\cos 2\gamma\cos\lambda\cos\delta\cos[\alpha-\phi_r-\Omega t]\\
&+\frac{1}{4}\sin 2\gamma\sin 2\lambda\cos\delta\sin[\alpha-\phi_r-\Omega t].
\end{aligned}$$

Hereγ is the orientation of the detectors arms, λ is the latitude of the detectors site, α and δ are the right ascension and declination of the GW, ϕ_r is a deterministic phase defining the position of the earth and Ω is the rotational angular velocity of the earth. This takes into account the antenna pattern modulation of the signal as the earth rotates the sun and orbits the earth. We then have a description of the SNR of a signal with a set of parameters for any given time and duration. This however does not describe how the

SNR of a signal varies with frequency which we need for spectrogram injections.

A.2 SNR with frequency

If one takes a sinusoidal signal and the takes the Fourier transform of that, then this should be a delta function at the frequency of the sinusoid. However, in the real world this sinusoid has a limited length and one will instead take the FFT of that signal, the signal will then be broken into discrete frequency bins. If the sinusoids frequency falls at the center of a frequency bin then the entire power of the signal will be contained within that frequency bin. However, if it is not perfectly centered on a frequency bin, then the power of the signal will begin to be spread over surrounding frequency bins. The aim is then to simulate how the signal is spread over surrounding frequency bins. If one has a sinusoid with a finite length, this is equivalent to taking an infinitely long sinusoid and convolving it rectangular window (box). The frequency response is then the Fourier transform of the sinusoid convolved with the Fourier transform of the box window. The Fourier transform of a box window is a sinc and of an infinitely long sinusoid is a delta function. The resulting Fourier transform is then a sinc function. One can write this down mathematically by writing the signal as

$$n(t) = \exp\{i2\pi f_0 t\}, \tag{A.5}$$

where f_0 is the signals frequency. The Fourier transform for this for a finite length of time which ranges between $-T/2 < t < T/2$ can be written as

$$\begin{aligned}
\tilde{n}(f) &= \int_{-T/2}^{T/2} \exp\{-i2\pi(f-f_0)t\}dt \\
&= \frac{1}{-i\pi(f-f_0)}\left(\exp\{-i\pi(f-f_0)T\} + \exp\{-i\pi(f-f_0)T\}\right) \\
&= \frac{2\sin\left(\pi(f-f_0)T\right)}{\pi(f-f_0)} \\
&= T\mathrm{sinc}\left(\pi(f-f_0)T\right)
\end{aligned} \tag{A.6}$$

One can verify this by taking the power spectrum of a finite length sinusoid, and plotting the square of a sinc function on top as shown in Fig. A.1.

For a sinusoid which has some frequency derivative the Fourier transform changes slightly. Similarly to above one can start with the definition of a signal with a constant frequency derivative such that

$$f = f_0 + \dot{f}t, \tag{A.7}$$

where f is its frequency, f_0 is the center frequency, $\dot{f}$ is its frequent derivative and t is

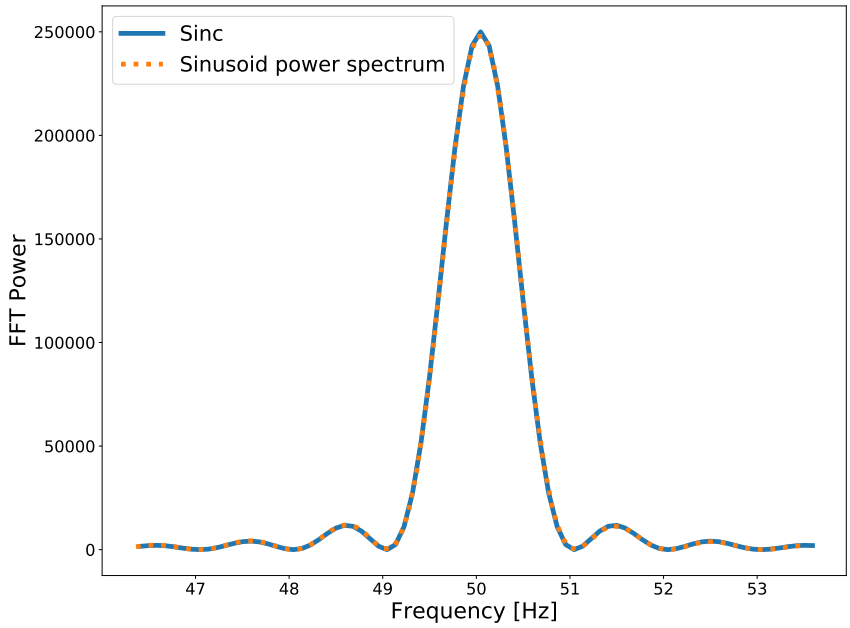


Figure A.1: If one takes the power spectrum of a sinusoid with finite length, this returns the same distribution as the square of a sinc function.

time. The signal can then be written as

$$\begin{aligned} n(t) &= a \exp\left[2\pi \int_0^t f(t)dt + \phi\right], \\ &= a \exp\left[2\pi(f_0 t + \frac{\dot{f}}{2}t^2) + \phi\right], \end{aligned} \tag{A.8}$$

where ϕ is some extra phase. In cases that follow we will have a finite length of data, we can center this around a time of zero; this is equivalent to applying a box window the signal. If we define the length of our signal as T we can say that the signal $n(t) = 0$ outside of $-T/2 \leq t \leq T/2$ [116]. The Fourier transform can then be written as

$$\begin{aligned} \tilde{n}(f) &= \frac{a}{2} \int_{-T/2}^{T/2} n(t) \exp\{-i2\pi t f\} dt \\ &= \frac{a}{2} e^{i\phi} \int_{-T/2}^{T/2} e^{i2\pi\left(\frac{\dot{f}}{2}t^2 + f_0 t - ft\right)} dt \\ &= \frac{a}{2} e^{i\phi} \left[I\right]. \end{aligned} \tag{A.9}$$

The integral can then be written as

$$\begin{aligned} I &= \int_{-T/2}^{T/2} \exp\left\{ i2\pi \left(\frac{\dot{f}}{2}t^2 + f_0 t - f \right) \right\} dt \\ &= \int_{-T/2}^{T/2} \exp\left\{ i2\pi \left(\frac{\dot{f}}{2}t^2 + (f - f_0)t \right) \right\} dt \\ &= \int_{-T/2}^{T/2} \exp\left\{ i\frac{\pi}{2}2\dot{f} \left(t^2 + 2\frac{(f - f_0)}{\dot{f}}t \right) \right\} dt \\ &= \int_{-T/2}^{T/2} \exp\left\{ i\frac{\pi}{2}2\dot{f} \left[\left(t + \frac{(f - f_0)}{\dot{f}} \right)^2 - \left(\frac{(f - f_0)}{\dot{f}} \right)^2 \right] \right\} dt \\ &= \exp\left\{ -i\frac{\pi}{2}2\dot{f} \left(\frac{(f - f_0)}{\dot{f}} \right)^2 \right\} \int_{-T/2}^{T/2} \exp\left\{ i\frac{\pi}{2}2\dot{f} \left[\left(t + \frac{(f - f_0)}{\dot{f}} \right)^2 \right] \right\} dt \end{aligned} \tag{A.10}$$

We can then substitute using

$$v = \sqrt{2\dot{f}} \left(t - \frac{(f - f_0)}{\dot{f}} \right) \tag{A.11}$$

where

$$dv = \sqrt{2\dot{f}} dt. \tag{A.12}$$

The integral then becomes

$$I = \exp\left\{ -i\frac{\pi}{2}2\dot{f} \left(\frac{(f - f_0)}{\dot{f}} \right)^2 \right\} \int_{-V_l}^{V_u} \exp\left\{ i\frac{\pi}{2}v^2 \right\} dv \tag{A.13}$$

When $t = -T/2$ and $t = T/2$ we can define the limits of the integral V_l and V_u respectively as

$$\begin{aligned} V_l &= \sqrt{2\dot{f}} \left(\frac{T}{2} + \frac{(f - f_0)}{\dot{f}} \right) \\ V_u &= \sqrt{2\dot{f}} \left(\frac{T}{2} - \frac{(f - f_0)}{\dot{f}} \right). \end{aligned} \tag{A.14}$$

As the $\sin(v^2)$ function is symmetric, this integral can be split up further such that

$$\begin{aligned} I = \exp\left\{ -i\frac{\pi}{2}2\dot{f} \left(\frac{(f - f_0)}{\dot{f}} \right)^2 \right\} \frac{1}{\sqrt{2\dot{f}}} \\ \cdot \left[\int_0^{V_u} \exp\left\{ i\frac{\pi}{2}v^2 \right\} dv + \int_0^{V_l} \exp\left\{ i\frac{\pi}{2}v^2 \right\} dv \right]. \end{aligned} \tag{A.15}$$

One can then use the Fresnel $S(x)$ and $C(x)$ defined by

$$\begin{aligned} C(x) &= \int_0^x \cos\frac{\pi}{2}t^2 dt \\ S(x) &= \int_0^x \sin\frac{\pi}{2}t^2 dt \end{aligned} \tag{A.16}$$

The Fourier transform is then

$$\tilde{n}(f) = \frac{a}{2\sqrt{2\dot{f}}} \exp\left\{ i(\phi - \pi\frac{f-f_0}{\dot{f}}) \right\} \left[C(V_l) + C(V_u) + i(S(V_l) + S(V_u)) \right], \tag{A.17}$$

where the power spectrum is then

$$|\tilde{n}(f)|^2 = \frac{a^2}{4|\dot{f}|} \left[(C(V_l) + C(V_u))^2 + (S(V_l) + S(V_u))^2 \right] \tag{A.18}$$

By taking a simple signal, we can verify that this is correct. Figure A.2 demonstrates the FFT of a simple signal and the power spectrum estimation using Eq. A.18.

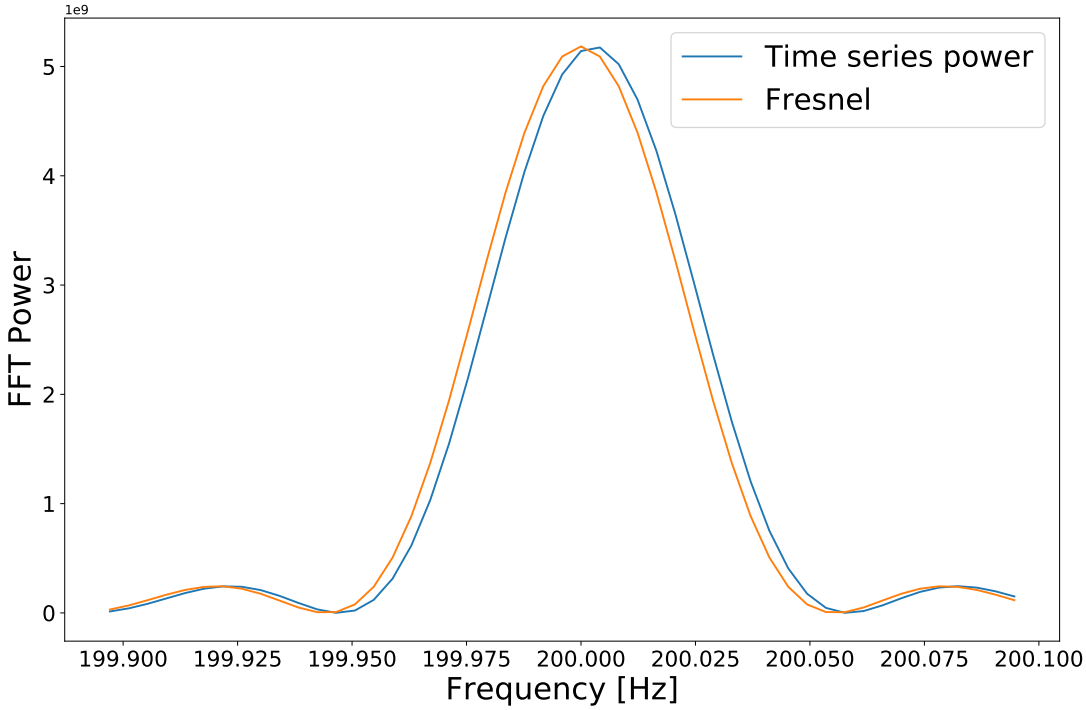


Figure A.2: The Fresnel integral form of the power spectrum above can then be compared to a numerically calculated power spectrum from Eq. A.8. This looks very similar to the sinc form in Fig. A.1, this is because the frequency derivative that are common for CW sources are very small $\sim 1e^{-9}$. **JOE: need to fix so they actually match, currently about 1 frequency bin off not sure why**

The values of this for the center location of each frequency bin surrounding the signal frequency can then be calculated for each time segment. A full spectrogram of a loud

signal ($\sim$ 1000 SNR) can be seen in Fig. A.3, A.4, A.5 and A.6 to demonstrate the signal simulations. The signal frequency for each time segment can be found using the model described in Sec. 2.1. Figure A.3 shows an example of a signal of fixed frequency which is simulated at the center of a frequency bin. When the signals frequency is then moved

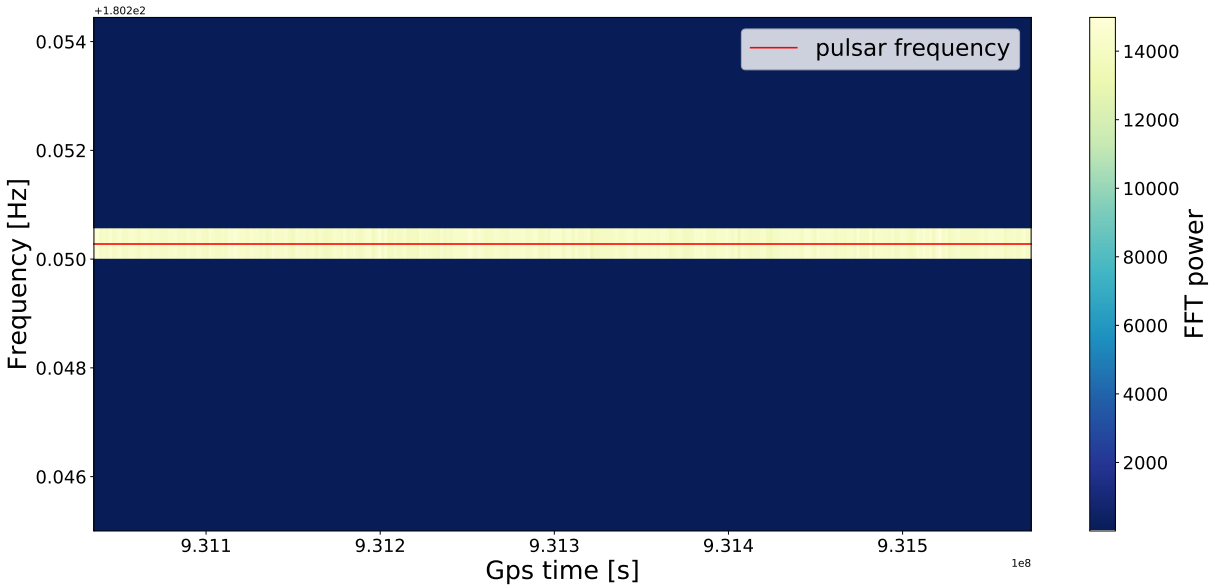


Figure A.3: By simulating a signal with a frequency in the center of a band, all of the signals power is contained within a single frequency bin. This shows an example of this kind of simulation in a spectrogram. The red line is the signals frequency evolution.

to the edge of a bin, the power can be seen to be distributed evenly between the two surrounding frequency bins. This can be seen in Fig. A.4. The doppler shift of a signal can then be added such that the frequency changes with time. This is shown in Fig. A.5. Finally Fig. A.6 shows the Doppler modulation and the antenna pattern modulation of a CW signal.

These simulations in the power spectrum greatly increased the speed of data generation when compared to simulating the signal in the time-series and taking their Fourier transform.

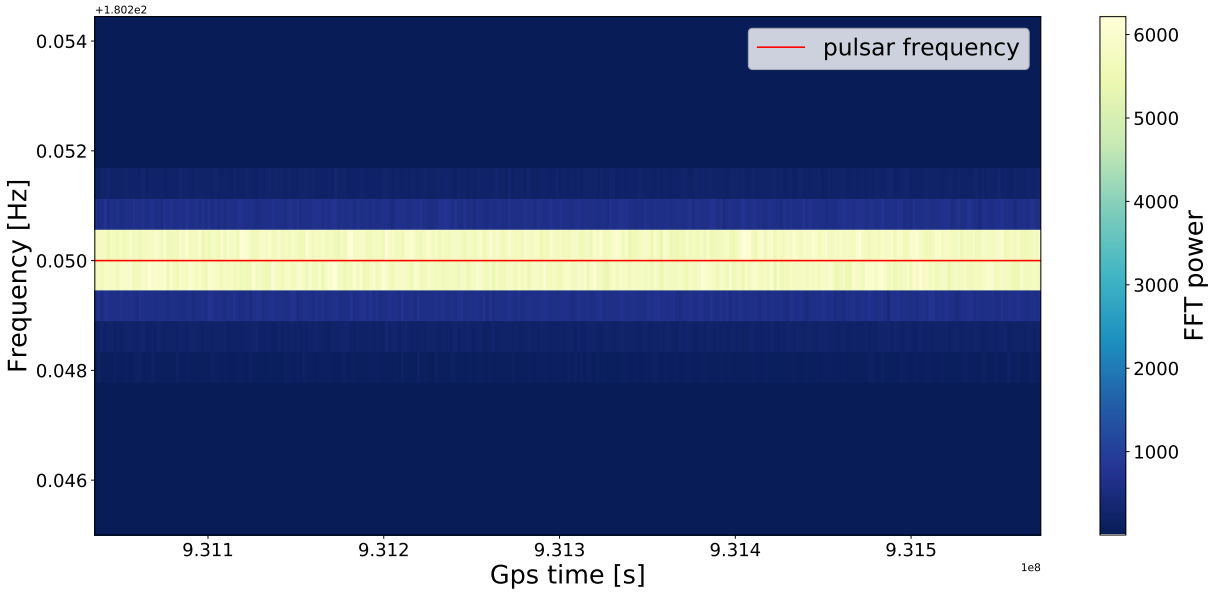


Figure A.4: By placing the signal at the edge of a frequency bin, the power is distributed around surrounding frequency bin, this can be seen above. The red line is the signals frequency evolution.

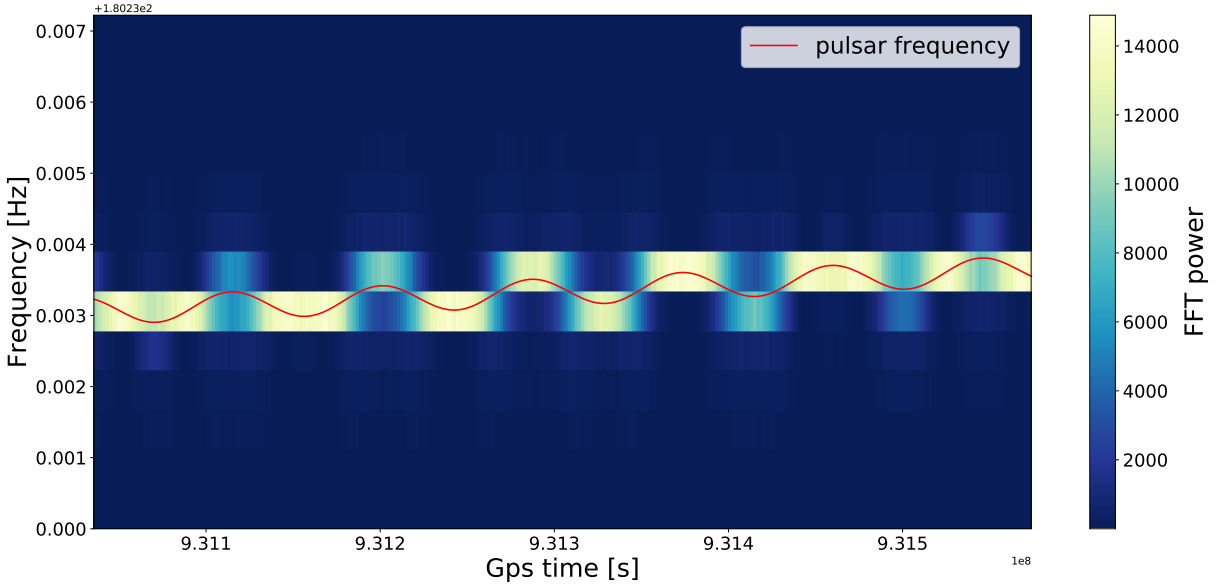


Figure A.5: Including the Doppler shift of a signal due to the earths rotation and orbit cause the signal to be modulated in frequency. This also causes a modulation in the SNR of the signal in and single frequency bin as it moves between bins. The red line is the signals frequency evolution.

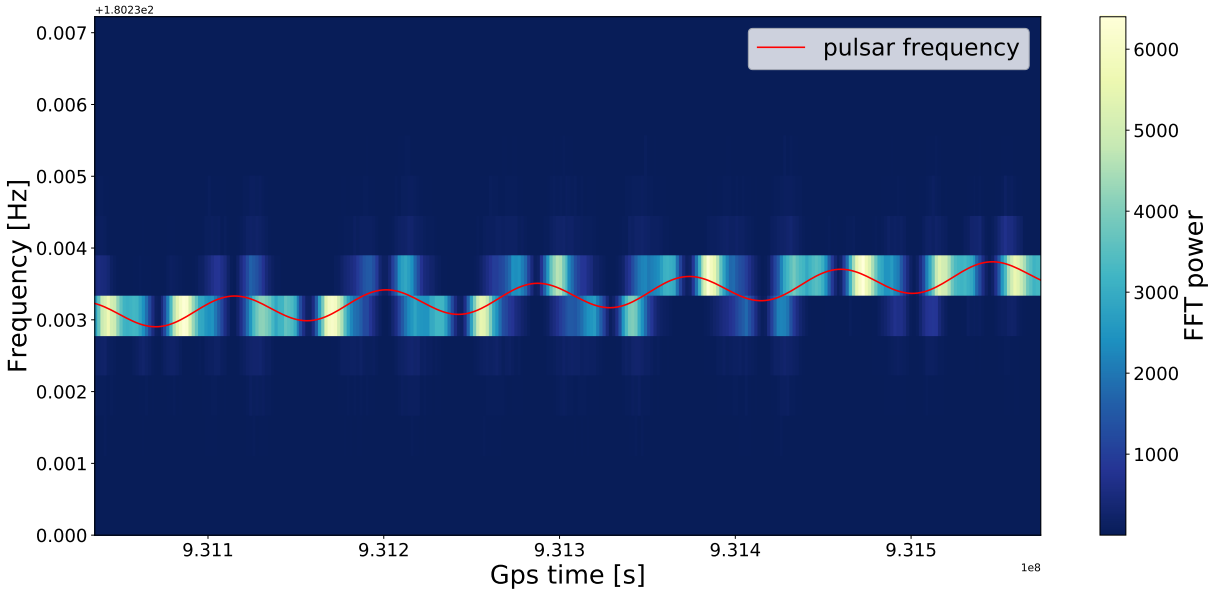


Figure A.6: The antenna pattern modulation can also be included to completely simulate a potential CW signal in a spectrogram. The red line is the signals frequency evolution.

Bibliography

[1] A. Einstein. "Die Grundlage Der Allgemeinen Relativitätstheorie [AdP 49, 769 (1916)]". en. In: *Annalen der Physik* 14.S1 (2005), pp. 517–571. ISSN: 1521-3889. DOI: 10.1002/andp.200590044. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/andp.200590044 (visited on 02/26/2020).

[2] Joel M. Weisberg, Joseph H. Taylor, and Lee A. Fowler. "Gravitational Waves from an Orbiting Pulsar". In: *Sci. Am.* 245.4 (Oct. 1981), pp. 74–82. ISSN: 0036-8733. DOI: 10.1038/scientificamerican1081-74.

[3] J. M. Weisberg and J. H. Taylor. "Relativistic Binary Pulsar B1913+16: Thirty Years of Observations and Analysis". In: (July 2004). URL: http://arxiv.org/abs/astro-ph/0407149.

[4] B. P. Abbott et al. "Observation of Gravitational Waves from a Binary Black Hole Merger". In: *Phys. Rev. Lett.* 116.6 (Feb. 2016), p. 061102. ISSN: 10797114. DOI: 10.1103/PhysRevLett.116.061102. URL: https://link.aps.org/doi/10.1103/PhysRevLett.116.061102.

[5] B. P. P. Abbott et al. "GW170814: A Three-Detector Observation of Gravitational Waves from a Binary Black Hole Coalescence". In: *Phys. Rev. Lett.* 119.14 (Oct. 2017), p. 141101. ISSN: 10797114. DOI: 10.1103/PhysRevLett.119.141101. URL: https://link.aps.org/doi/10.1103/PhysRevLett.119.141101.

[6] The LIGO Scientific Collaboration et al. "GW190425: Observation of a Compact Binary Coalescence with Total Mass $\backslashsim 3.4 M_{\backslashodot}$". In: (Jan. 2020). URL: http://arxiv.org/abs/2001.01761.

[7] B. P. P. Abbott et al. "GW170817: Observation of Gravitational Waves from a Binary Neutron Star Inspiral". In: *Phys. Rev. Lett.* 119.16 (Oct. 2017), p. 161101. ISSN: 10797114. DOI: 10.1103/PhysRevLett.119.161101. URL: https://link.aps.org/doi/10.1103/PhysRevLett.119.161101.

[8] Eanna E. Flanagan and Scott A. Hughes. "The Basics of Gravitational Wave Theory". In: *New J. Phys.* 7 (Jan. 2005). DOI: 10.1088/1367-2630/7/1/204. URL: http://arxiv.org/abs/gr-qc/0501041%20http://dx.doi.org/10.1088/1367-2630/7/1/204.

[9] Alexandre Le Tiec and Jérôme Novak. "Theory of Gravitational Waves". In: *An Overview of Gravitational Waves*. WORLD SCIENTIFIC, May 2016, pp. 1–41. ISBN: 978-981-314-175-9. DOI: 10.1142/9789813141766_0001. URL: https://www.worldscientific.com/doi/abs/10.1142/9789813141766_0001 (visited on 03/19/2020).

[10] Collin Capano. "Searching for Gravitational Waves from Compact Binary Coalescence Using LIGO and Virgo Data". In: *Phys. - Diss.* (Dec. 2011). URL: https://surface.syr.edu/phy_etd/114.

[11] J. Aasi et al. "Advanced LIGO". In: *Class. Quantum Gravity* 32.7 (Apr. 2015), p. 074001. ISSN: 13616382. DOI: 10.1088/0264-9381/32/7/074001. URL: http://stacks.iop.org/0264-9381/32/i=7/a=074001?key=crossref.20895763c84bce3f8929251031b2475c.

[12] F. Acernese et al. "Advanced Virgo: A Second-Generation Interferometric Gravitational Wave Detector". In: *Class. Quantum Gravity* 32.2 (Jan. 2015), p. 024001. ISSN: 13616382. DOI: 10.1088/0264-9381/32/2/024001. URL: http://stacks.iop.org/0264-9381/32/i=2/a=024001?key=crossref.f3779cf21eab5f6631c6650a20708

[13] LIGO Scientific Collaboration and Virgo Collaboration et al. "GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs". In: *Phys. Rev. X* 9.3 (Sept. 2019), p. 031040. DOI: 10.1103/PhysRevX.9.031040. URL: https://link.aps.org/doi/10.1103/PhysRevX.9.031040 (visited on 03/25/2020).

[14] Karsten Danzmann and the LISA study team. "LISA: Laser Interferometer Space Antenna for Gravitational Wave Measurements". en. In: *Class. Quantum Grav.* 13.11A (Nov. 1996), A247–A250. ISSN: 0264-9381. DOI: 10.1088/0264-9381/13/11A/033. URL: https://doi.org/10.1088%5C%2F0264-9381%5C%2F13%5C%2F11a%5C%2F033 (visited on 02/26/2020).

[15] Éanna É. Flanagan and Tanja Hinderer. "Constraining Neutron-Star Tidal Love Numbers with Gravitational-Wave Detectors". In: *Phys. Rev. D* 77.2 (Jan. 2008), p. 021502. DOI: 10.1103/PhysRevD.77.021502. URL: https://link.aps.org/doi/10.1103/PhysRevD.77.021502 (visited on 03/31/2020).

[16] Ian Harry and Tanja Hinderer. "Observing and Measuring the Neutron-Star Equation-of-State in Spinning Binary Neutron Star Systems". en. In: *Class. Quantum Grav.* 35.14 (June 2018), p. 145010. ISSN: 0264-9381. DOI: 10.1088/1361-6382/aac7e3. URL: https://doi.org/10.1088%5C%2F1361-6382%5C%2Faac7e3 (visited on 02/26/2020).

[17] The LIGO Scientific Collaboration and The Virgo Collaboration et al. "A Gravitational-Wave Standard Siren Measurement of the Hubble Constant". en. In: *Nature* 551.7678 (Nov. 2017), pp. 85–88. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature24471. URL: http://www.nature.com/articles/nature24471 (visited on 02/26/2020).

[18] Michael Zevin et al. "Constraining Formation Models of Binary Black Holes with Gravitational-Wave Observations". en. In: *ApJ* 846.1 (Sept. 2017), p. 82. ISSN: 0004-637X. DOI: 10.3847/1538-4357/aa8408. URL: https://doi.org/10.3847%5C%2F1538-4357%5C%2Faa8408 (visited on 02/26/2020).

[19] Ilya Mandel and Alison Farmer. "Merging Stellar-Mass Binary Black Holes". In: *arXiv:1806.05820 [astro-ph, physics:gr-qc]* (June 2018). URL: http://arxiv.org/abs/1806.05820 (visited on 02/26/2020).

[20] Neil J. Cornish and Tyson B. Littenberg. "Bayeswave: Bayesian Inference for Gravitational Wave Bursts and Instrument Glitches". In: *Class. Quantum Gravity* 32.13 (July 2015), p. 135012. ISSN: 13616382. DOI: 10.1088/0264-9381/32/13/135012.

[21] S. Klimenko et al. "A Coherent Method for Detection of Gravitational Wave Bursts". In: *Class. Quantum Gravity*. Vol. 25. IOP Publishing, June 2008, p. 114029. DOI: 10.1088/0264-9381/25/11/114029.

[22] Christian D. Ott. "The Gravitational Wave Signature of Core-Collapse Supernovae". In: *Class. Quantum Gravity* 26.6 (Sept. 2008). DOI: 10.1088/0264-9381/26/6/063001. URL: http://arxiv.org/abs/0809.0695%20http://dx.doi.org/10.1088/0264-9381/26/6/063001.

[23] J. Aasi et al. "Search for Gravitational Waves Associated with γ-Ray Bursts Detected by the Interplanetary Network". In: *Phys. Rev. Lett.* 113.1 (June 2014), p. 011102. ISSN: 10797114. DOI: 10.1103/PhysRevLett.113.011102.

[24] Thibault Damour and Alexander Vilenkin. "Gravitational Radiation from Cosmic (Super)Strings: Bursts, Stochastic Background, and Observational Windows". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 71.6 (Mar. 2005), pp. 1–13. ISSN: 15502368. DOI: 10.1103/PhysRevD.71.063510.

[25] Nelson Christensen. "Stochastic Gravitational Wave Backgrounds". In: *Reports Prog. Phys.* 82.1 (Nov. 2018). DOI: 10.1088/1361-6633/aae6b5. URL: http://arxiv.org/abs/1811.08797%20http://dx.doi.org/10.1088/1361-6633/aae6b5.

[26] Joseph D. Romano. "Searches for Stochastic Gravitational-Wave Backgrounds". In: (Aug. 2019). URL: http://arxiv.org/abs/1909.00269.

[27] R. N. Manchester et al. "The Australia Telescope National Facility Pulsar Catalogue". en. In: *AJ* 129.4 (Apr. 2005), p. 1993. ISSN: 1538-3881. DOI: 10.1086/428488. URL: https://iopscience.iop.org/article/10.1086/428488/meta (visited on 03/26/2020).

[28] Sushan Konar. "Magnetic Fields of Neutron Stars". In: *J. Astrophys. Astron.* 38.3 (Sept. 2017). DOI: 10.1007/s12036-017-9467-4. URL: http://arxiv.org/abs/1709.07106%20http://dx.doi.org/10.1007/s12036-017-9467-4.

[29] James M. Lattimer and M. Prakash. "The Equation of State of Hot, Dense Matter and Neutron Stars". In: *Phys. Rep.* 621 (Dec. 2015), pp. 127–164. DOI: 10.1016/j.physrep.2015.12.005. URL: http://arxiv.org/abs/1512.07820%20http://dx.doi.org/10.1016/j.physrep.2015.12.005.

[30] Kostas Glampedakis and Leonardo Gualtieri. "Gravitational Waves from Single Neutron Stars: An Advanced Detector Era Survey". In: (Sept. 2017), pp. 673–736. DOI: 10.1007/978-3-319-97616-7_12. URL: http://arxiv.org/abs/1709.07049%20http://dx.doi.org/10.1007/978-3-319-97616-7_12.

[31] Keith Riles. "Recent Searches for Continuous Gravitational Waves". In: *Mod. Phys. Lett. A* 32.39 (Dec. 2017), p. 1730035. ISSN: 0217-7323, 1793-6632. DOI: 10.1142/S021773231730035X. URL: http://arxiv.org/abs/1712.05897 (visited on 02/20/2020).

[32] B. Haskell et al. "Detecting Gravitational Waves from Mountains on Neutron Stars in the Advanced Detector Era". In: *Mon. Not. R. Astron. Soc.* 450.3 (July 2015), pp. 2393–2403. ISSN: 0035-8711. DOI: 10.1093/mnras/stv726. URL: http://academic.oup.com/mnras/article/450/3/2393/1056611/Detecting-gravitational-waves-from-mountains-on.

[33] Paul D. Lasky. "Gravitational Waves from Neutron Stars: A Review". In: *Publ. Astron. Soc. Aust.* 32 (Aug. 2015). DOI: 10.1017/pasa.2015.35. URL: http://arxiv.org/abs/1508.06643%20http://dx.doi.org/10.1017/pasa.2015.35.

[34] Werner Becker, ed. *Neutron Stars and Pulsars*. Vol. 357. Astrophysics and Space Science Library. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. ISBN: 978-3-540-76964-4. DOI: 10.1007/978-3-540-76965-1. URL: http://link.springer.com/10.1007/978-3-540-76965-1.

[35] Ian Jones. *The CFS Instability*. URL: http://www.personal.soton.ac.uk/dij/cfs.html.

[36] S. Chandrasekhar. "Solutions of Two Problems in the Theory of Gravitational Radiation". In: *Phys. Rev. Lett.* 24.11 (Mar. 1970), pp. 611–615. DOI: 10.1103/PhysRevLett.24.611. URL: https://link.aps.org/doi/10.1103/PhysRevLett.24.611 (visited on 02/29/2020).

[37] J. L. Friedman and B. F. Schutz. "Secular Instability of Rotating Newtonian Stars". In: *The Astrophysical Journal* 222 (May 1978), pp. 281–296. ISSN: 0004-637X. DOI: 10.1086/156143. URL: http://adsabs.harvard.edu/abs/1978ApJ...222..281F (visited on 02/29/2020).

[38] Benjamin J. Owen et al. "Gravitational Waves from Hot Young Rapidly Rotating Neutron Stars". In: 58 (1998), pp. 1–15. DOI: 10.1103/PhysRevD.58.084020. URL: http://arxiv.org/abs/gr-qc/9804044%5C%7B%5C%%5C%7D0Ahttp://dx.doi.org/10.1103/PhysRevD.58.084020.

[39] J. Weber. "Observation of the Thermal Fluctuations of a Gravitational-Wave Detector". In: *Phys. Rev. Lett.* 17.24 (Dec. 1966), pp. 1228–1230. ISSN: 00319007. DOI: 10.1103/PhysRevLett.17.1228.

[40] A. De Waard et al. "MiniGRAIL, the First Spherical Detector". In: *Class. Quantum Gravity*. Vol. 20. IOP Publishing, May 2003, S143. DOI: 10.1088/0264-9381/20/10/317.

[41] George Hobbs and Shi Dai. "Gravitational Wave Research Using Pulsar Timing Arrays". In: *Natl. Sci. Rev.* 4.5 (Sept. 2017), pp. 707–717. ISSN: 2095-5138. DOI: 10.1093/nsr/nwx126. URL: https://academic.oup.com/nsr/article/4/5/707/4566535.

[42] P. A.R. Ade et al. "Constraints on Primordial Gravitational Waves Using Planck, WMAP, and New BICEP2/ Keck Observations through the 2015 Season". In: *Phys. Rev. Lett.* 121.22 (Nov. 2018), p. 221301. ISSN: 10797114. DOI: 10.1103/PhysRevLett.121.221301.

[43] B. P. Abbott et al. "LIGO: The Laser Interferometer Gravitational-Wave Observatory". In: *Reports Prog. Phys.* 72.7 (July 2009), p. 076901. ISSN: 00344885. DOI: 10.1088/0034-4885/72/7/076901. URL: http://stacks.iop.org/0034-4885/72/i=7/a=076901?key=crossref.669f7c34d7d28503fd7f84b53db6048f.

[44] F. Acernese et al. "Status of Virgo". In: *Class. Quantum Gravity* 25.11 (June 2008), p. 114045. ISSN: 02649381. DOI: 10.1088/0264-9381/25/11/114045. URL: http://stacks.iop.org/0264-9381/25/i=11/a=114045?key=crossref.62c81a5e3c6d5978d66477c5fb155f0c.

[45] F. Matichard et al. "Seismic Isolation of Advanced LIGO: Review of Strategy, Instrumentation and Performance". en. In: *Class. Quantum Grav.* 32.18 (Aug. 2015), p. 185003. ISSN: 0264-9381. DOI: 10.1088/0264-9381/32/18/185003. URL: https://doi.org/10.1088%5C%2F0264-9381%5C%2F32%5C%2F18%5C%2F185003 (visited on 03/07/2020).

[46] Matthew Robert Abernathy et al. "An Overview of Research into Low Internal Friction Optical Coatings by the Gravitational Wave Detection Community". en. In: *Materials Research* 21 (0). ISSN: 1516-1439. DOI: 10.1590/1980-5373-mr-2017-0863. URL: http://www.scielo.br/scielo.php?script=sci_abstract%5C&pid=S1516-14392018000800203%5C&lng=en%5C&nrm=iso%5C&tlng=en (visited on 03/07/2020).

[47] J. Aasi et al. "Enhanced Sensitivity of the LIGO Gravitational Wave Detector by Using Squeezed States of Light". en. In: *Nature Photon* 7.8 (Aug. 2013), pp. 613–619. ISSN: 1749-4893. DOI: 10.1038/nphoton.2013.177. URL: https://www.nature.com/articles/nphoton.2013.177 (visited on 03/07/2020).

[48] B. Schutz. "Data Analysis of Gravitational-Wave Signals from Spinning Neutron Stars: The Signal and Its Detection". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 58.6 (Aug. 1998), p. 063001. ISSN: 15502368. DOI: 10.1103/PhysRevD.58.063001.

[49] Réjean J. Dupuis and Graham Woan. "Bayesian Estimation of Pulsar Parameters from Gravitational Wave Data". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 72.10 (Nov. 2005), p. 102002. ISSN: 15507998. DOI: 10.1103/PhysRevD.72.102002. URL: https://link.aps.org/doi/10.1103/PhysRevD.72.102002.

[50] Nicholas Metropolis et al. "Equation of State Calculations by Fast Computing Machines". In: *J. Chem. Phys.* 21.6 (June 1953), pp. 1087–1092. ISSN: 00219606. DOI: 10.1063/1.1699114. URL: http://aip.scitation.org/doi/10.1063/1.1699114.

[51] Don van Ravenzwaaij, Pete Cassey, and Scott D. Brown. "A Simple Introduction to Markov Chain Monte–Carlo Sampling". In: *Psychon. Bull. Rev.* 25.1 (Feb. 2018), pp. 143–154. ISSN: 15315320. DOI: 10.3758/s13423-016-1015-8.

[52] Sanjib Sharma. "Markov Chain Monte Carlo Methods for Bayesian Data Analysis in Astronomy". In: *Annu. Rev. Astron. Astrophys.* 55.1 (Aug. 2017), pp. 213–259. ISSN: 0066-4146. DOI: 10.1146/annurev-astro-082214-122339.

[53] John Skilling. "Nested Sampling for General Bayesian Computation". In: *Bayesian Anal.* 1.4 (2006), pp. 833–860. ISSN: 19360975. DOI: 10.1214/06-BA127.

[54] Joshua S Speagle. "Dynesty: A Dynamic Nested Sampling Package for Estimating Bayesian Posteriors and Evidences". In: *Mon. Not. R. Astron. Soc.* (Apr. 2019). DOI: 10.1093/mnras/staa278. URL: http://arxiv.org/abs/1904.02180%20http://dx.doi.org/10.1093/mnras/staa278.

[55] Reinhard Prix. "Search for Continuous Gravitational Waves: Metric of the Multi-detector F-Statistic". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 75.2 (Jan. 2007), p. 023004. ISSN: 15507998. DOI: 10.1103/PhysRevD.75.023004. URL: http://arxiv.org/abs/gr-qc/0606088%20http://dx.doi.org/10.1103/PhysRevD.75.023004%2010.1103/PhysRevD.75.069901%20https://link.aps.org/doi/10.1103/PhysRevD.75.023004.

[56] S. Walsh et al. "Comparison of Methods for the Detection of Gravitational Waves from Unknown Neutron Stars". In: *Phys. Rev. D* 94.12 (Dec. 2016), p. 124010. ISSN: 24700029. DOI: 10.1103/PhysRevD.94.124010. URL: https://link.aps.org/doi/10.1103/PhysRevD.94.124010.

[57] Patrick R. Brady and Teviet Creighton. "Searching for Periodic Sources with LIGO. II. Hierarchical Searches". In: *Phys. Rev. D* 61.8 (Feb. 2000), p. 082001. DOI: 10.1103/PhysRevD.61.082001. URL: https://link.aps.org/doi/10.1103/PhysRevD.61.082001 (visited on 02/25/2020).

[58] Curt Cutler, Iraj Gholami, and Badri Krishnan. "Improved Stack-Slide Searches for Gravitational-Wave Pulsars". In: *Phys. Rev. D* 72.4 (Aug. 2005), p. 042004. DOI: 10.1103/PhysRevD.72.042004. URL: https://link.aps.org/doi/10.1103/PhysRevD.72.042004 (visited on 02/25/2020).

[59] Badri Krishnan et al. "Hough Transform Search for Continuous Gravitational Waves". In: *Phys. Rev. D* 70.8 (Oct. 2004), p. 082001. DOI: 10.1103/PhysRevD.70.082001. URL: https://link.aps.org/doi/10.1103/PhysRevD.70.082001 (visited on 02/25/2020).

[60] F. Antonucci et al. "Detection of Periodic Gravitational Wave Sources by Hough Transform in Thefversus \textbackslashskew6\textbackslashdot f Plane". en. In: *Class. Quantum Grav.* 25.18 (Sept. 2008), p. 184015. ISSN: 0264-9381. DOI: 10.1088/0264-9381/25/18/184015. URL: https://doi.org/10.1088%5C%2F0264-9381%5C%2F25%5C%2F18%5C%2F184015 (visited on 02/25/2020).

[61] Pia Astone et al. "Method for All-Sky Searches of Continuous Gravitational Wave Signals Using the Frequency-Hough Transform". In: *Phys. Rev. D* 90.4 (Aug. 2014), p. 042002. DOI: 10.1103/PhysRevD.90.042002. URL: https://link.aps.org/doi/10.1103/PhysRevD.90.042002 (visited on 02/25/2020).

[62] Avneet Singh et al. "Results of an All-Sky High-Frequency Einstein@Home Search for Continuous Gravitational Waves in LIGO's Fifth Science Run". In: *Phys. Rev. D* 94.6 (Sept. 2016), p. 064061. DOI: 10.1103/PhysRevD.94.064061. URL: https://link.aps.org/doi/10.1103/PhysRevD.94.064061 (visited on 02/26/2020).

[63] Maria Alessandra Papa et al. "Hierarchical Follow-up of Subthreshold Candidates of an All-Sky Einstein@Home Search for Continuous Gravitational Waves on LIGO Sixth Science Run Data". In: *Phys. Rev. D* 94.12 (Dec. 2016), p. 122006. DOI: 10.1103/PhysRevD.94.122006. URL: https://link.aps.org/doi/10.1103/PhysRevD.94.122006 (visited on 02/26/2020).

[64] *Einstein@Home*. URL: https://einsteinathome.org/ (visited on 02/26/2020).

[65] J. Aasi et al. "Implementation of an \textdollar\textbackslashmathcal$\lbrace$F$\rbrace$\t Statistic All-Sky Search for Continuous Gravitational Waves in Virgo VSR1 Data". en. In: *Class. Quantum Grav.* 31.16 (Aug. 2014), p. 165014. ISSN: 0264-9381. DOI: 10.1088/0264-9381/31/16/165014. URL: https://doi.org/10.1088%5C%2F0264-9381%5C%2F31%5C%2F16%5C%2F165014 (visited on 02/26/2020).

[66] J. Abadie et al. "All-Sky Search for Periodic Gravitational Waves in the Full S5 LIGO Data". In: *Phys. Rev. D* 85.2 (Jan. 2012), p. 022001. DOI: 10.1103/PhysRevD.85.022001. URL: https://link.aps.org/doi/10.1103/PhysRevD.85.022001.

[67] LIGO Scientific Collaboration and Virgo Collaboration et al. "Comprehensive All-Sky Search for Periodic Gravitational Waves in the Sixth Science Run LIGO Data". In: *Phys. Rev. D* 94.4 (Aug. 2016), p. 042002. DOI: 10.1103/PhysRevD.94.042002. URL: https://link.aps.org/doi/10.1103/PhysRevD.94.042002 (visited on 02/26/2020).

[68] Andrew J. Viterbi. "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm". In: *IEEE Trans. Inf. Theory* 13.2 (Apr. 1967), pp. 260–269. ISSN: 15579654. DOI: 10.1109/TIT.1967.1054010. URL: http://ieeexplore.ieee.org/document/1054010/.

[69] L. Sun et al. "Hidden Markov Model Tracking of Continuous Gravitational Waves from Young Supernova Remnants". In: (Oct. 2017). DOI: 10.1103/physrevd.97.043013. URL: https://arxiv.org/abs/1710.00460.

[70] S. Suvorova et al. "Hidden Markov Model Tracking of Continuous Gravitational Waves from a Binary Neutron Star with Wandering Spin. II. Binary Orbital Phase Tracking". In: *Phys. Rev. D* 96.10 (Nov. 2017), p. 102006. DOI: 10.1103/PhysRevD.96.102006. URL: https://link.aps.org/doi/10.1103/PhysRevD.96.102006.

[71] B. P. Abbott et al. "Search for Gravitational Waves from Scorpius X-1 in the First Advanced LIGO Observing Run with a Hidden Markov Model". In: *Phys. Rev. D* 95.12 (June 2017), p. 122003. DOI: 10.1103/PhysRevD.95.122003. URL: https://link.aps.org/doi/10.1103/PhysRevD.95.122003.

[72] B. P. Abbott et al. "Search for Gravitational Waves from a Long-Lived Remnant of the Binary Neutron Star Merger GW170817". In: (2018).

[73] Ling Sun and Andrew Melatos. "Application of Hidden Markov Model Tracking to the Search for Long-Duration Transient Gravitational Waves from the Remnant of the Binary Neutron Star Merger GW170817". In: *arXiv e-prints*, arXiv:1810.03577 (Oct. 2018), arXiv:1810.03577.

[74] LIGO Scientific Collaboration and Virgo Collaboration et al. "All-Sky Search for Periodic Gravitational Waves in the O1 LIGO Data". In: *Phys. Rev. D* 96.6 (Sept. 2017), p. 062002. DOI: 10.1103/PhysRevD.96.062002. URL: https://link.aps.org/doi/10.1103/PhysRevD.96.062002 (visited on 02/28/2020).

[75] Joe Bayley, Graham Woan, and Chris Messenger. "SOAP: A Generalised Application of the Viterbi Algorithm to Searches for Continuous Gravitational-Wave Signals". In: (Mar. 2019). DOI: 10.1103/PhysRevD.100.023006. URL: http://arxiv.org/abs/1903.12614%20http://dx.doi.org/10.1103/PhysRevD.100.023006.

[76] Reinhard Prix. "Gravitational Waves from Spinning Neutron Stars". In: *Neutron Stars Pulsars*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 651–685. DOI: 10.1007/978-3-540-76965-1_24. URL: http://link.springer.com/10.1007/978-3-540-76965-1_24.

[77] Benjamin J. Owen. "Probing Neutron Stars with Gravitational Waves". In: (2009).

[78] P. Astone et al. "A Method for Detection of Known Sources of Continuous Gravitational Wave Signals in Non-Stationary Data". In: *Class. Quantum Gravity* 27.19 (Oct. 2010). ISSN: 02649381. DOI: 10.1088/0264-9381/27/19/194016.

[79] The LIGO Scientific Collaboration et al. "First Search for Gravitational Waves from Known Pulsars with Advanced LIGO". In: 12 (2017). ISSN: 1538-4357. DOI: 10.3847/1538-4357/aa677f. URL: http://arxiv.org/abs/1701.07709%5C%7B%5C%%5C%7D0Ahttp://dx.doi.org/10.3847/1538-4357/aa677f.

[80] B. P. Abbott et al. "Searches for Gravitational Waves from Known Pulsars at Two Harmonics in 2015-2017 LIGO Data". In: (2019).

[81] Teviet Creighton. "Searching for Periodic Sources with LIGO. II. Hierarchical Searches". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 61.8 (Feb. 2000), p. 082001. ISSN: 15502368. DOI: 10.1103/PhysRevD.61.082001. URL: https://link.aps.org/doi/10.1103/PhysRevD.61.082001.

[82] B. P. Abbott et al. "All-Sky Search for Continuous Gravitational Waves from Isolated Neutron Stars Using Advanced LIGO O2 Data". In: *Phys. Rev. D* 100.2 (July 2019), p. 024004. ISSN: 2470-0010. DOI: 10.1103/physrevd.100.024004. URL: https://link.aps.org/doi/10.1103/PhysRevD.100.024004.

[83] David R. Ellis. *Snakes on a Plane*. Action, Crime, Thriller. IMDb ID: tt0417148 event-location: USA. Aug. 2006.

[84] S Suvorova et al. "Hidden Markov Model Tracking of Continuous Gravitational Waves from a Neutron Star with Wandering Spin". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 93.12 (2016), pp. 1–17. ISSN: 15502368. DOI: 10.1103/PhysRevD.93.123009.

[85] G Larry Bretthorst. "Bayesian Spectrum Analysis and Parameter Estimation". In: *Springer-Verlag*. 1988, p. 220. ISBN: 0-387-96871-7.

[86] David Keitel et al. "Search for Continuous Gravitational Waves: Improving Robustness versus Instrumental Artifacts". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 89.6 (2014), pp. 1–19. ISSN: 15502368. DOI: 10.1103/PhysRevD.89.064023.

[87] LIGO Scientific Collaboration. "LIGO Algorithm Library - LALSuite". In: (2018). DOI: 10.7935/GT1W-FZ16.

[88] Berit Behnke, Maria Alessandra Papa, and Reinhard Prix. "Postprocessing Methods Used in the Search for Continuous Gravitational-Wave Signals from the Galactic Center". In: *Phys. Rev. D - Part. Fields, Gravit. Cosmol.* 91.6 (Mar. 2015), p. 064007. ISSN: 15502368. DOI: 10.1103/PhysRevD.91.064007. URL: https://link.aps.org/doi/10.1103/PhysRevD.91.064007.

[89] J. Aasi et al. "Characterization of the LIGO Detectors during Their Sixth Science Run". In: *Class. Quantum Gravity* 32.11 (2015). ISSN: 13616382. DOI: 10.1088/0264-9381/32/11/115012.

[90] Michael Coughlin, the Ligo Scientific Collaboration, and the Virgo Collaboration. "Noise Line Identification in LIGO S6 and Virgo VSR2". In: *Journal of Physics: Conference Series* 243.1 (2010), p. 012010. URL: http://stacks.iop.org/1742-6596/243/i=1/a=012010.

[91] B. P. Abbott et al. "Search for Gravitational Waves from a Long-Lived Remnant of the Binary Neutron Star Merger GW170817". en. In: *ApJ* 875.2 (Apr. 2019), p. 160. ISSN: 1538-4357. DOI: 10.3847/1538-4357/ab0f3d. URL: https://iopscience.iop.org/article/10.3847/1538-4357/ab0f3d (visited on 04/01/2020).

[92] Hunter Gabbard et al. "Matching Matched Filtering with Deep Networks for Gravitational-Wave Astronomy". In: *Phys. Rev. Lett.* 120.14 (Apr. 2018), p. 141103. ISSN: 10797114. DOI: 10.1103/PhysRevLett.120.141103. URL: https://link.aps.org/doi/10.1103/PhysRevLett.120.141103.

[93] Daniel George and E. A. Huerta. "Deep Learning for Real-Time Gravitational Wave Detection and Parameter Estimation: Results with Advanced LIGO Data". In: *Phys. Lett. Sect. B Nucl. Elem. Part. High-Energy Phys.* 778 (Mar. 2018), pp. 64–70. ISSN: 03702693. DOI: 10.1016/j.physletb.2017.12.053. URL: https://www.sciencedirect.com/science/article/pii/S0370269317310390?via%5C%3Dihub.

[94] Timothy D. Gebhard et al. "Convolutional Neural Networks: A Magic Bullet for Gravitational-Wave Detection?" In: *Phys. Rev. D* 100.6 (Sept. 2019), p. 063015. ISSN: 24700029. DOI: 10.1103/PhysRevD.100.063015. URL: https://link.aps.org/doi/10.1103/PhysRevD.100.063015.

[95] Christoph Dreissigacker et al. "Deep-Learning Continuous Gravitational Waves". In: *Phys. Rev. D* 100.4 (Aug. 2019), p. 044009. ISSN: 24700029. DOI: 10.1103/PhysRevD.100.044009. URL: https://link.aps.org/doi/10.1103/PhysRevD.100.044009.

[96] Chigozie Nwankpa et al. "Activation Functions: Comparison of Trends in Practice and Research for Deep Learning". In: *arXiv:1811.03378 [cs]* (Nov. 2018). URL: http://arxiv.org/abs/1811.03378 (visited on 03/30/2020).

[97] Andrew L. Maas. "Rectifier Nonlinearities Improve Neural Network Acoustic Models". In: 2013.

[98] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. "Deep Learning". In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 14764687. DOI: 10.1038/nature14539. URL: http://www.nature.com/articles/nature14539.

[99] Yann LeCun et al. "Gradient-Based Learning Applied to Document Recognition". In: *Proc. IEEE* 86.11 (1998), pp. 2278–2323. ISSN: 00189219. DOI: 10.1109/5.726791. URL: http://ieeexplore.ieee.org/document/726791/.

[100] Alexander Waibel et al. "Phoneme Recognition Using Time-Delay Neural Networks". In: *IEEE Trans. Acoust.* 37.3 (Mar. 1989), pp. 328–339. ISSN: 00963518. DOI: 10.1109/29.21701. URL: http://ieeexplore.ieee.org/document/21701/.

[101] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-

classification-with-deep-convolutional-neural-networks.pdf (visited on 02/27/2020).

[102] Diederik P. Kingma and Jimmy Lei Ba. "Adam: A Method for Stochastic Optimization". In: *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.* (Dec. 2015). URL: http://arxiv.org/abs/1412.6980.

[103] L Y Pratt. *Discriminability-Based Transfer between Neural Networks*. Tech. rep.

[104] Simard Patrice et al. *Tangent Prop: A Formalism for Specifying Selected Invariances in Adaptive Networks*. 1991. URL: https://papers.nips.cc/paper/536-tangent-prop-a-formalism-for-specifying-selected-invariances-in-an-adaptive-network.

[105] Henry S. Baird. "Document Image Defect Models". In: *Struct. Doc. Image Anal.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 546–556. DOI: 10.1007/978-3-642-77281-8_26. URL: http://www.springerlink.com/index/10.1007/978-3-642-77281-8_26.

[106] Stéfan Van Der Walt et al. "Scikit-Image: Image Processing in Python". In: *PeerJ* 2014.1 (June 2014), e453. ISSN: 21678359. DOI: 10.7717/peerj.453. URL: https://peerj.com/articles/453.

[107] B. P. Abbott et al. "Results of the Deepest All-Sky Survey for Continuous Gravitational Waves on LIGO S6 Data Running on the Einstein@Home Volunteer Distributed Computing Project". In: *Phys. Rev. D* 94.10 (Nov. 2016), p. 102002. ISSN: 24700029. DOI: 10.1103/PhysRevD.94.102002. URL: https://link.aps.org/doi/10.1103/PhysRevD.94.102002.

[108] Chris Pankow et al. "Mitigation of the Instrumental Noise Transient in Gravitational-Wave Data Surrounding GW170817". In: *Phys. Rev. D* 98.8 (Oct. 2018), p. 084016. DOI: 10.1103/PhysRevD.98.084016. URL: https://link.aps.org/doi/10.1103/PhysRevD.98.084016 (visited on 03/18/2020).

[109] *GW Open Science Center*. URL: https://www.gw-openscience.org/o1speclines/ (visited on 02/29/2020).

[110] P. B. Covas et al. "Identification and Mitigation of Narrow Spectral Artifacts That Degrade Searches for Persistent Gravitational Waves in the First Two Observing Runs of Advanced LIGO". In: *Phys. Rev. D* 97.8 (Apr. 2018), p. 082002. ISSN: 24700029. DOI: 10.1103/PhysRevD.97.082002.

[111] D. Tuyenbayev et al. "Improving LIGO Calibration Accuracy by Tracking and Compensating for Slow Temporal Variations". en. In: *Class. Quantum Grav.* 34.1 (Dec. 2016), p. 015002. ISSN: 0264-9381. DOI: 10.1088/0264-9381/34/1/015002. URL: https://doi.org/10.1088%5C%2F0264-9381%5C%2F34%5C%2F1%5C%2F015002 (visited on 02/28/2020).

[112] Derek Davis et al. "Improving the Sensitivity of Advanced LIGO Using Noise Subtraction". en. In: *Class. Quantum Grav.* 36.5 (Feb. 2019), p. 055011. ISSN: 0264-9381. DOI: 10.1088/1361-6382/ab01c5. URL: https://doi.org/10.1088%5C%2F1361-6382%5C%2Fab01c5 (visited on 02/28/2020).

[113] Ansel Neunzert. *Daily Comb Summary*. URL: https://ldas-jobs.ligo-wa.caltech.edu/~aneunzert/O3combs/combtracking/summary.html (visited on 03/23/2020).

[114] T Accadia et al. "The NoEMi (Noise Frequency Event Miner) Framework". In: *J. Phys.: Conf. Ser.* 363 (June 2012), p. 012037. ISSN: 1742-6596. DOI: 10.1088/1742-6596/363/1/012037. URL: http://stacks.iop.org/1742-6596/363/i=1/a=012037?key=crossref.4b8bfd9320b240a51529da2d97549ab6 (visited on 02/28/2020).

[115] Joseph Bayley. *Home*. URL: https://ldas-jobs.ligo.caltech.edu/~joseph.bayley/soap/lines.php (visited on 03/02/2020).

[116] T. Misaridis and J.A. Jensen. "Use of Modulated Excitation Signals in Medical Ultrasound. Part II: Design and Performance for Medical Imaging Applications". In: *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control* 52.2 (Feb. 2005). Conference Name: IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, pp. 192–207. ISSN: 1525-8955. DOI: 10.1109/TUFFC.2005.1406546.