Detection of minute-long Gravitational Wave transients using Deep Learning methods

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Science

by

Vincent BOUDART

under the supervision of

Prof. Maxime FAYS and **Prof. Jean-René** CUDELL

IFPA Group
Department of Astrophysics, Geophysics and Oceanography
Faculty of Science

UNIVERSITY OF LIÈGE

September 2023

Declaration of Authorship

I, Vincent BOUDART, declare that this thesis titled, “Detection of minute-long Gravitational Wave transients using Deep Learning methods” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Look up at the stars and not down at your feet. Try to make sense of what you see, and wonder about what makes the universe exist. Be curious."

Stephen Hawking

Abstract

On the 11th of February 2016, the Laser Interferometer Gravitational-wave Observatory (LIGO) and the Virgo collaboration jointly announced the first direct observation of a gravitational wave. Coming from a binary black hole merger, the detection was made possible thanks to match filtering techniques which require a collection of accurate waveform models. In opposition to compact binary mergers, minute-long transients include a wide range of astrophysical phenomena including accretion disk instabilities, fallback of matter onto neutron stars and magnetar flares. Because of their inherent complexity, most of these phenomena are poorly modeled, preventing the use of matched filtering methods. Such events are thus probed through the template-free excess-power method, consisting in searching for a local excess of power in the time-frequency (TF) space correlated between detectors. Their detection is mainly constrained by environmental and instrumental transient noises called glitches. Glitches contaminate burst searches, reducing the amount of useful data and limiting the sensitivity of current algorithms. It is therefore of primordial importance to distinguish them from potential burst signals. In this thesis, I propose to build a Convolutional Neural Network (CNN) in order to highlight both burst signals and glitches in TF maps. The CNN is designed to set apart groups of pixels resembling glitches, acting both as an anomaly detector and a classifier. A thorough analysis shows that our new algorithm is competitive with the current state-of-the-art burst pipelines while being much faster.

Chapter 1 introduces the framework of General Relativity needed for gravitational-wave detection. In Chapter 2, we give details about the detectors' architecture as well as the main sources of noise. Chapter 3 reviews the general methods implemented to detect bursts and provides extensive details about long-duration searches. An introduction to deep learning with a special focus on convolutional neural networks is found in Chapter 4. In Chapter 5, we present the method implemented in this thesis as well as some major improvements based on a former analysis. Chapter 6 is finally dedicated to the inclusion of our algorithm into an existing pipeline, called Pyxel. The results of our new pipeline to the Burst Benchmark Project are also presented and prospects for future work are envisaged.

Acknowledgements

This thesis has been a wonderful journey. As for any research paper, on the cover, there should not only be the name of the researcher but also all the names of the people who contributed to varying degrees and provided support and encouragement. I would first like to acknowledge my supervisor Prof. Maxime Fays and my co-supervisor Prof. Jean-René Cudell, who made this work possible. Their guidance and mentorship carried me through all the stages of this thesis. I am grateful for the fruitful and enlightening discussions we had during these 4 years.

I had the pleasure of working with Grégory Baltus, Atri Bhattacharya and Prasanta Char, who shared their experience and love of physics with me. I would like to thank them for their comments and suggestions and for the moments we spent together in and out of the office. I would also like to acknowledge Melissa Lopez, a Ph.D. student at the University of Utrecht, with whom I investigated and discussed various artificial intelligence aspects. I wish them all the best for the suite of their career.

I deeply believe that everyone performs better at work when his/her life is filled with friends and family. I could not have undertaken this journey without the support and love of my parents and my little sister. I am also deeply indebted to my friends from Liège for the board game nights and the good times we shared. Many thanks to Célia, Jean-François, Sarah, Thomas, Alice, Henri, Galdan and Gwen. This endeavor would not have been possible without my high-school friends, Estelle, Malicia and Max. I will be forever grateful for being understanding when I could not join their celebrations and dinners. Words cannot express how much their support is important to me.

I must definitely say a word about my beloved Sandrine, whose unconditional support and love helped me to be stronger. She was the one to endure my mood swings with a smile and for this, she deserves the biggest thank you (and probably 1 or 2 dinners).

I also want to give a special thanks to Rachel, who is the first friend I made when coming to Liège. She was there during the toughest times of this thesis and helped me to get over it. I hope I was able to return the favor to you in some way.

I could not write this thesis without expressing how my best friend, Corentin, influenced my work and attitude over the years. As we started competing for the highest grades in high school, we pulled each other up. We ended up becoming friends and sharing our passion for football. The way he behaved pushed me to always give the best out of me. I would definitely not be here today if I had not met him. Thank you for everything, Coco.

Lastly, I would like to show my deepest gratitude to Mrs Gilmaire, my high-school maths teacher. Her commitment and passion have always been an inspiration for me to work hard and be grateful for the chances we have in life. I will never forget her dedication to teaching and the values she passed on to me.

Finally, I want to thank again my parents for allowing me to pursue university studies. It was a childhood dream to learn about stars, space and time. This thesis comes as the culmination of years of dedication to learn how the universe is working. I can say that I am proud to have brought my little contribution to the field of Science.

Contents

List of Abbreviations

AI	**A**rtificial **I**ntelligence
ALBUS	**A**nomaly detection for **L**ong-duration **BU**rst **S**earches
BNS	**B**inary **N**eutron **S**tar
CA	**C**oherent **A**mplitude
CBC	**C**ompact **B**inary **C**oalaescence
CNN	**C**onvolutional **N**eural **N**etwork
DL	**D**eep **L**earning
EDT	**E**uclidean **D**istance **T**ransform
ELU	**E**xponential **L**inear **U**nit
FAR	**F**alse **A**larm **R**ate
GPS	**G**lobal **P**ositioning **S**ystem
GR	**G**eneral **R**elativity
GW	**G**ravitational **W**ave
H1	LIGO **H**anford detector
HRSS	**R**oot **S**um **S**quared strain amplitude (noted **h**)
L1	LIGO **L**ivingston detector
LSC	**L**IGO **S**cientific **C**ollaboration
MAE	**M**ean **A**bsolute **E**rror
MDC	**M**ock-**D**ata **C**hallenge
ML	**M**achine **L**earning
MLP	**M**ulti-**L**ayer **P**erceptron
MSE	**M**ean **S**quared **E**rror
NN	**N**eural **N**etwork
O3	**O**bserving run **3**
ReLU	**Re**ctified **L**inear **U**nit
RGB	**R**ed-**G**reen-**B**lue
SNR	**S**ignal-to-**N**oise **R**atio
TF	**T**ime-**F**requency
V1	**V**irgo detector

I dedicate this thesis to all the people who have supported me through my education, especially my friends and family. This achievement comes as the end of an adventure and the beginning of another.

Chapter 1

From General Relativity to Gravitational Waves

1.1 Fundamentals of General Relativity

1.1.1 Metric tensor and geodesic equations

In General Relativity (GR), the trajectory of free-falling objects in spacetime is described through the geodesic equations. Prior to considering the latter, let us introduce the definition of spacetime itself. In GR, spacetime unifies the three dimensions of space and one dimension of time into a single four-dimensional manifold. It is described by a second-order tensor known as the metric. In turn, the metric allows to define the notion of distance which is fundamental in order to characterize the motion of objects. If we consider two points in a manifold, infinitely close to each other, the distance ds between them is defined as:

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu \tag{1.1}$$

where $g_{\mu\nu}$ is the metric tensor and x^μ are coordinates of spacetime (t, x, y, z). Einstein's summation convention is used, meaning that there exists an implicit sum on repeated indices (here μ and ν).

The metric tensor leads to a calculation of the deformation of spacetime, i.e. it permits to evaluate the curvature of spacetime at every point of a manifold. If we consider a classical Euclidean space in 2 dimensions, the metric in cartesian coordinates reduces to:

$$g = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{1.2}$$

and the distance between two points of this space is expressed as:

$$ds^2 = \begin{pmatrix} dx & dy \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} = dx^2 + dy^2 \tag{1.3}$$

which corresponds to the Euclidean distance formula. Note that the time coordinate does not appear in the Euclidean metric, i.e. space and time are not coupled. Newton's laws of motion, built within the Euclidean formalism, therefore describe a universe in which time is absolute. In the case of special relativity, the Minkowski metric is used to describe the flat space in which space and time are associated:

$$g = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1.4}$$

Now that we know how to define a distance in a four-dimensional manifold, let us consider a particle moving freely under the influence of a gravitational field. The particle is said to be in free-fall since it is not subject to external forces. According to the Principle of Equivalence, stating that the inertial mass is equivalent to the gravitational mass, the particle must be at rest with respect to a freely falling coordinate system noted $X^\mu(x)$. Setting $T \equiv X^0$, the following equation is therefore satisfied locally [1]:

$$\frac{d^2X^\mu}{dT^2} = 0 \tag{1.5}$$

Using the chain rule of partial derivatives, this leads to:

$$\frac{dX^\mu}{dT} = \frac{dx^\nu}{dT}\frac{\partial X^\mu}{\partial x^\nu} \tag{1.6}$$

By differentiating one more time with respect to T,

$$\frac{d^2X^\mu}{dT^2} = \frac{d^2x^\nu}{dT^2}\frac{\partial X^\mu}{\partial x^\nu} + \frac{dx^\nu}{dT}\frac{dx^\alpha}{dT}\frac{\partial^2 X^\mu}{\partial x^\nu \partial x^\alpha} = 0 \tag{1.7}$$

Therefore:

$$\frac{d^2x^\nu}{dT^2}\frac{\partial X^\mu}{\partial x^\nu} = -\frac{dx^\nu}{dT}\frac{dx^\alpha}{dT}\frac{\partial^2 X^\mu}{\partial x^\nu \partial x^\alpha} \tag{1.8}$$

Multiplying both sides by $\frac{\partial x^\lambda}{\partial X^\mu}$ and using the product rule

$$\frac{\partial X^\mu}{\partial x^\nu}\frac{\partial x^\lambda}{\partial X^\mu} = \delta^\lambda_\nu, \tag{1.9}$$

we have:

$$\frac{d^2x^\lambda}{dT^2} = -\frac{dx^\nu}{dT}\frac{dx^\alpha}{dT}\left[\frac{\partial^2 X^\mu}{\partial x^\nu \partial x^\alpha}\frac{\partial x^\lambda}{\partial X^\mu}\right] \tag{1.10}$$

This gives the equation of motion:

$$\frac{d^2x^\lambda}{dT^2} + \Gamma^\lambda_{\nu\alpha}\frac{dx^\nu}{dT}\frac{dx^\alpha}{dT} = 0 \tag{1.11}$$

where $\Gamma^\lambda_{\nu\alpha}$ is the affine connection, defined by:

$$\Gamma^\lambda_{\nu\alpha} = \frac{\partial^2 X^\mu}{\partial x^\nu \partial x^\alpha}\frac{\partial x^\lambda}{\partial X^\mu} \tag{1.12}$$

Expression (1.11) settles the geodesic equations. Their solutions, known as geodesics are the generalization, in curved space, of a straight line in flat space. Just as in Newton's theory where free objects follow straight lines, in General Relativity free objects follow geodesics. Within equations (1.11), the affine connections appear, noted $\Gamma^\lambda_{\nu\alpha}$. These connections determine the gravitational force as seen from an external observer in a locally inertial frame of reference.

To understand the role of the connections $\Gamma^\lambda_{\nu\alpha}$ in GR truly, consider the surface of a sphere as a curved manifold as shown in the left panel of Fig. 1.1. At every point on this surface, we can define tangent spaces on which we can build vectors. If we want to compare these vectors, we need to transport one of the two vectors in the tangent plane of

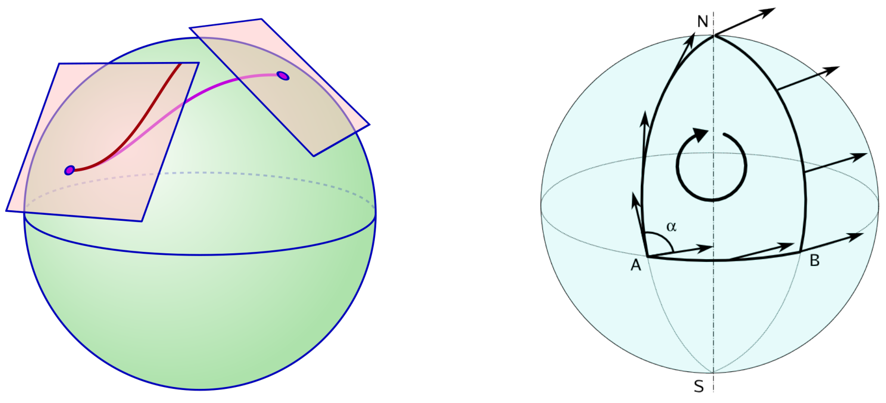

FIGURE 1.1: Left: Illustration of tangent planes defined at the surface of a sphere. The covariant derivative is used to compare vectors defined on different planes. Right: Geometrical representation of the parallel transport on a curved manifold. If we transport a vector from A to B, then from B to N and finally come back to A, the direction of the vector will be altered.

the other vector. For this, a mathematical tool exists and is called the covariant derivative:

$$\nabla_\mu V^\nu = \partial_\mu V^\nu + \Gamma^\nu_{\mu\rho} V^\rho \tag{1.13}$$

For a vector V, defined in one plane, the covariant derivative has a term containing the connection coefficients. The translation of the vector V along a path defined on the surface is not well defined and depends on the path along which the vector is translated. As an example, let us take a vector defined at a point A along the equator (see right panel of Fig. 1.1). If we transport the vector first along the equator to point B, then along a meridian to the north pole noted N, and finally transport it along another meridian back to A, the orientation of this vector will differ from the original vector. This is a direct consequence of the curvature of the surface: parallel transport does not yield the same result whether the vector lies on a flat or curved space. In the case of an Euclidean space, which is flat, the direction of the vector would be unaltered. The connection coefficients in (1.13) serve to express this change of direction and contain the information related to the curvature of the manifold.

As a first result, we obtained that the connections are responsible for the gravitational force in expression (1.11). Then, we concluded that these connections express the curvature of the manifold in the definition of the covariant derivative (1.13). Since the metric tensor, used to introduce the notion of distance, defines spacetime itself, it must be related to the connection coefficients. However, under some hypotheses[1] we can show that only one coefficient is compatible. It is expressed as:

$$\Gamma^\alpha_{\mu\nu} = \frac{1}{2} g^{\alpha\beta} (\partial_\mu g_{\nu\beta} + \partial_\nu g_{\mu\beta} - \partial_\beta g_{\mu\nu}) \tag{1.14}$$

Note that, even if the above expression involves the metric tensor, $\Gamma^\alpha_{\mu\nu}$ is not a tensor. This connection, called the Christoffel symbol, is of primordial importance in General Relativity. The Christoffel symbol associates the curvature of spacetime with the metric. If we look back at the geodesic equations (1.11), the curvature of spacetime itself is now responsible for the force of gravitation, which is the fundamental principle in General Relativity.

[1]The coefficient must be torsion-free ($\Gamma^\nu_{\mu\rho} = \Gamma^\nu_{\rho\mu}$) and the covariant derivative of the metric should be zero ($\nabla_\alpha g_{\mu\nu} = 0$), i.e. the metric should be preserved everywhere on the manifold.

1.1.2 Einstein's equations

In order to solve the equations of motion (1.11), we need to evaluate the Christoffel symbols, and consequently the metric tensor. For this, Einstein assumed that matter and energy are responsible for the curvature of spacetime. He connected the metric tensor $g_{\mu\nu}$ to the stress-energy tensor $T_{\mu\nu}$ via his famous Einstein's equations:

$$R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu} \tag{1.15}$$

where G is the Cavendish gravitational constant and c is the speed of light in vacuum. As the metric tensor is defined on a four-dimensional spacetime, the above expression should represent 16 equations: each of the four dimensions (t, x, y, z) affecting each of the other four. However, $R_{\mu\nu} = R_{\nu\mu}$, $g_{\mu\nu} = g_{\nu\mu}$ and $T_{\mu\nu} = T_{\nu\mu}$, i.e. the tensors are symmetric. Thus, only 10 equations need to be solved to derive the metric $g_{\mu\nu}$. Let us now detail each term of (1.15) individually.

The first term on the left hand side is the Ricci tensor. It is a contraction of the Riemann tensor:

$$R^{\rho}_{\theta\mu\nu} = \partial_\mu \Gamma^{\rho}_{\nu\theta} - \partial_\nu \Gamma^{\rho}_{\mu\theta} + \Gamma^{\rho}_{\mu\lambda}\Gamma^{\lambda}_{\nu\theta} - \Gamma^{\rho}_{\nu\lambda}\Gamma^{\lambda}_{\mu\theta} \tag{1.16}$$

This tensor characterizes the variation of a parallel-transported vector on an infinitesimal loop. If we transport a vector V^σ around a close loop defined by two vectors A^μ and B^ν as shown in Fig. 1.2, the expression of the change δV^ρ experienced by this vector is of the form [2]:

$$\delta V^\rho = (\delta a)\,(\delta b)\, A^\mu\, B^\nu\, R^{\rho}_{\sigma\mu\nu}\, V^\sigma \tag{1.17}$$

The Riemann tensor therefore contains information about the curvature of the manifold. In the case of a flat manifold, the change δV^ρ should be zero since the parallel-transported vector is identical to the initial vector. This imposes the Riemann tensor to be null for a flat manifold, and $R^{\rho}_{\sigma\mu\nu} \neq 0$ otherwise.

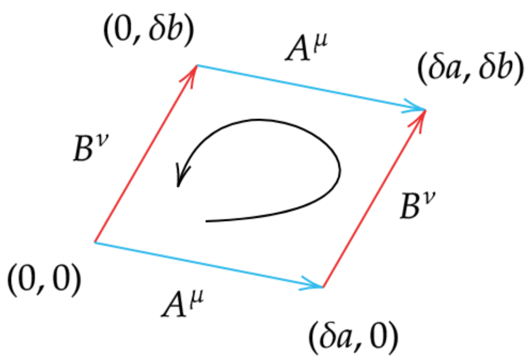

FIGURE 1.2: Infinitesimal loop defined by two vectors A^μ and B^ν.

From the Riemann tensor, we can build the Ricci tensor $R_{\mu\nu}$ and the Ricci scalar R that both appear in Einstein's equations (1.15). The first one is obtained by contraction of the Riemann tensor over its first and third indices:

$$R_{\mu\nu} = R^{\lambda}_{\mu\lambda\nu} \tag{1.18}$$

This latter expression can be further contracted and yields the Ricci scalar:

$$R = R^{\mu}_{\ \mu} \tag{1.19}$$

The Ricci tensor, defined through the Riemann tensor and appearing in Einstein's equations, can be interpreted as the measure of volume changes of an object [1]. It describes how much a volume element has changed under a gravitational field. Besides, the Ricci scalar, which appears as the second term in (1.15), is a local measure of the curvature of a manifold. It assigns a unique real number to each point of this manifold [2].

The third term on the left side of (1.15) contains the cosmological constant Λ. The latter considers the effects of the expansion of the Universe on the metric tensor. As Λ is very small and its effects are perceptible on very long time scales, this term will be neglected throughout this thesis. The implications of the cosmological constant in Einstein's equations are beyond the scope of this thesis and we refer to [1] for more details.

The term on the right-hand side of Einstein's equations contains the stress-energy tensor $T_{\mu\nu}$ that describes the energy density, momentum density and stress at that point in spacetime. It is defined as:

$$T_{\mu\nu} = \begin{pmatrix} u & \rho_x & \rho_y & \rho_z \\ \rho_x & P_{xx} & \sigma_{xy} & \sigma_{xz} \\ \rho_y & \sigma_{yx} & P_{yy} & \sigma_{yz} \\ \rho_z & \sigma_{zx} & \sigma_{zy} & P_{zz} \end{pmatrix} \tag{1.20}$$

where u represents the energy density, ρ_i are the momentum densities, P_{ii} stand for the pressures and σ_{ij} are the shear stresses. The element $T^{\mu\nu}$ therefore expresses how much energy in the μ direction flows into the ν direction at a given point in spacetime.

All in all, Einstein's equations tell us how spacetime (left-hand side) reacts to the matter (right-hand side) within it. Hence the expression "matter tells spacetime how to curve, and spacetime tells matter how to move" makes perfect sense. Einstein's equations need to be solved to determine the metric tensor $g_{\mu\nu}$, which in turn is used to predict the motion of objects in a gravitational field via the geodesic equations (1.11).

1.2 Gravitational Waves

1.2.1 Derivation from Einstein's equations

Once we have introduced the fundamental concepts and equations in General Relativity, let us try to solve Einstein's equations. Because of their non-linearity, finding analytical solutions is complex and tedious. However, if we consider a flat Minkowski spacetime, to which we add a small perturbation, it becomes possible to linearize Einstein's equations. This almost flat Minkowski spacetime admits the following metric:

$$g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu} + O(h^2_{\mu\nu}), \qquad |h_{\mu\nu}| << 1. \tag{1.21}$$

where $\eta_{\mu\nu}$ is the Minkowski metric (1.4) and $h_{\mu\nu}$ is a small correction due to a weak gravitational field. Note that, under the hypothesis of a weak field, one can raise and lower the indices by $\eta_{\mu\nu}$, i.e.:

$$\begin{aligned} h^{\alpha}_{\nu} &= \eta^{\alpha\mu} h_{\mu\nu} \\ h = h^{\mu}_{\mu} &= \eta^{\mu\nu} h_{\mu\nu} \end{aligned} \tag{1.22}$$

where h is the trace of the perturbation $h_{\mu\nu}$.

Injecting (1.21) into the expression of the Christoffel symbols (1.14) leads to:

$$\Gamma^{\alpha}_{\mu\nu} = \frac{1}{2}\left(\partial_\mu h_\nu^{\ \alpha} + \partial_\nu h_\mu^{\ \alpha} - \partial^\alpha h_{\mu\nu}\right) \tag{1.23}$$

which in turn gives the Riemann tensor via expression (1.16):

$$R^{\rho}_{\theta\mu\nu} = \frac{1}{2}\left(\partial_\mu \partial_\theta h_\nu^{\ \rho} + \partial_\nu \partial^\rho h_{\mu\theta} - \partial_\mu \partial^\rho h_{\nu\theta} - \partial_\nu \partial_\theta h_\mu^{\ \rho}\right) \tag{1.24}$$

The Ricci tensor can be obtained from the contraction of the Riemann tensor:

$$R_{\mu\nu} = \frac{1}{2}\left(\partial^\beta \partial_\mu h_{\nu\beta} + \partial_\nu \partial^\beta h_{\beta\mu} - \partial_\beta \partial^\beta h_{\mu\nu} - \partial_\mu \partial_\nu h\right) \tag{1.25}$$

for which a further contraction gives the Ricci scalar:

$$R = \partial^\alpha \partial^\beta h_{\alpha\beta} - \partial_\alpha \partial^\alpha h \tag{1.26}$$

Einstein's equations therefore write as:

$$\frac{1}{2}\left(\partial^\beta \partial_\mu h_{\nu\beta} + \partial_\nu \partial^\beta h_{\beta\mu} - \partial_\beta \partial^\beta h_{\mu\nu} - \partial_\mu \partial_\nu h\right) - \frac{1}{2}\eta_{\mu\nu}\left(\partial^\alpha \partial^\beta h_{\alpha\beta} - \partial_\alpha \partial^\alpha h\right) = \frac{8\pi G}{c^4} T_{\mu\nu} \tag{1.27}$$

To simplify the above expression further, let us define the tensor $\bar{h}_{\mu\nu} = h_{\mu\nu} - \frac{1}{2}\eta_{\mu\nu} h$ whose trace $\bar{h} = \eta^{\mu\nu}\bar{h}_{\mu\nu} = -h$. With this new definition, equation (1.27) gives:

$$\frac{1}{2}\left(\partial_\mu \partial^\beta \bar{h}_{\nu\beta} + \partial_\nu \partial^\beta \bar{h}_{\mu\beta} - \partial_\beta \partial^\beta \bar{h}_{\mu\nu} - \eta_{\mu\nu}\partial^\alpha \partial^\beta \bar{h}_{\alpha\beta}\right) = \frac{8\pi G}{c^4} T_{\mu\nu} \tag{1.28}$$

To go further, we must note that GR is a gauge theory. Contrarily to Newton's theory and special relativity, we do not need to work in an inertial frame of reference. We can thus freely choose the coordinate system in which we write Einstein's equations. By choosing the appropriate coordinate system (i.e. fixing the gauge), we can get rid of redundant degrees of freedom and thus obtain the simplest expressions possible. Here, we consider the Lorenz gauge, defined by the condition: $\partial^\mu \bar{h}_{\mu\nu} = 0$. Substituting this condition into (1.28) allows to discard the first, second and fourth term on the left-hand side, leading to:

$$\partial_\beta \partial^\beta \bar{h}_{\mu\nu} = -\frac{16\pi G}{c^4} T_{\mu\nu} \tag{1.29}$$

Equations (1.29) are known as the linearized Einstein equations. The choice of the Lorenz gauge, $\partial^\mu \bar{h}_{\mu\nu} = 0$, imposes 4 conditions ($\nu = 0, 1, 2$ or 3) that reduce the 10 independent components of the symmetric tensor $h_{\mu\nu}$ to 6.

In the absence of matter, they become:

$$\partial_\beta \partial^\beta \bar{h}_{\mu\nu} = 0 \tag{1.30}$$

The expression $\partial_\beta \partial^\beta = \Box$ is known as the d'Alembertian operator and the above equation is therefore a wave equation. The solution of (1.30) is then a wave traveling at the speed of light in vacuum, called a gravitational wave (GW). For example, a monochromatic plane wave propagating in a direction noted z would take the following form:

$$\bar{h}_{\mu\nu} = A_{\mu\nu} \cos(2\pi f\,(t - z/c) + \varphi_0) \tag{1.31}$$

where $A_{\mu\nu}$ is a tensor describing the amplitude of the wave, f represents the frequency of the wave and φ_0 is an initial phase. At this stage, $\bar{h}_{\mu\nu}$ is a symmetric tensor showing 6 independent components. The Lorenz gauge therefore determines a whole class of gauge transformations, so we need to further specify the coordinate transformation. Let us consider the transformation $x^\mu \to x^\mu + \xi^\mu(x)$. The metric transforms as [3]:

$$g_{\mu\nu}(x) \to g'_{\mu\nu}(x') = \frac{\partial x^\rho}{\partial x'^\mu}\frac{\partial x^\sigma}{\partial x'^\nu} g_{\rho\sigma}(x). \tag{1.32}$$

Using this transformation law, $\bar{h}_{\mu\nu}$ becomes

$$\bar{h}_{\mu\nu} \to \bar{h}'_{\mu\nu} = \bar{h}_{\mu\nu} - (\partial_\mu \xi_\nu + \partial_\nu \xi_\mu - \eta_{\mu\nu}\partial_\rho \xi^\rho). \tag{1.33}$$

The coordinate transformation $x^\mu \to x^\mu + \xi^\mu(x)$ cannot be chosen without restriction. Indeed, it has to fulfil the Lorenz condition $\partial^\mu \bar{h}_{\mu\nu} = 0$, which translates as:

$$\partial^\mu \bar{h}_{\mu\nu} \to (\partial^\mu \bar{h}_{\mu\nu})' = \partial^\mu \bar{h}_{\mu\nu} - \Box \xi_\mu \tag{1.34}$$

If the initial field $h_{\mu\nu}$ is such that $\partial^\mu \bar{h}_{\mu\nu} = 0$, then ξ^μ must be chosen so that $\Box \xi_\mu = 0$. This means that we can choose the functions $\xi_\mu(x)$ so as to impose 4 conditions on $\bar{h}_{\mu\nu}$ via the transformation (1.33). In particular we can choose ξ^0 and ξ^i such that the trace $\bar{h} = 0$ and $h^{0i} = 0$ respectively. Since the trace is zero, $\bar{h}_{\mu\nu} = h_{\mu\nu}$ and the Lorenz condition with $\mu = 0$ then reads

$$\partial^0 h_{00} + \partial^i h_{0i} = 0 \tag{1.35}$$

which further implies $\partial^0 h_{00} = 0$, i.e. h_{00} is constant in time. A time-independent term corresponds to the Newtonian potential of the source, which is the static part of the gravitational interactions. Therefore, as far as gravitational waves are concerned, $\partial^0 h_{00} = 0$ also means $h_{00} = 0$. We thus have set all four components $h_{0\mu} = 0$ and we are left only with the spatial components $h_{ij} \neq 0$.

The 4 conditions impose by the Lorenz gauge and the 4 conditions on $h_{0\mu} = 0$ define the transverse-traceless gauge or TT-gauge. By imposing this gauge, the symmetric matrix $\bar{h}_{\mu\nu}$ now has just 2 degrees of freedom to be shared between the non-zero spatial components h_{ij}. In the case of the monochromatic plane wave described in (1.31), it can take the form:

$$h^{TT}_{\mu\nu} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & h_+ & h_\times & 0 \\ 0 & h_\times & -h_+ & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \cos(2\pi f\,(t - z/c) + \varphi_0) \tag{1.36}$$

where the two degrees of freedom are denoted h_+ and $h_\times$. To understand their meaning, let us study what happens when this wave propagates through matter.

1.2.2 Effect on matter

In order to understand what is a gravitational wave, it is necessary to apprehend its effect on matter. For this, let us consider an almost flat Minkowski space in which a gravitational wave is propagating. The metric is therefore defined by $g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}^{TT}$, which gives:

$$g_{\mu\nu} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 + h_+ \cos(2\pi f(t - z/c) + \varphi_0) & h_\times \cos(2\pi f(t - z/c) + \varphi_0) & 0 \\ 0 & h_\times \cos(2\pi f(t - z/c) + \varphi_0) & 1 - h_+ \cos(2\pi f(t - z/c) + \varphi_0) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1.37}$$

Injecting the above metric into the definition of the interval ds (1.1), we obtain:

$$\begin{aligned} ds^2 = -c^2dt^2 &+ \big(1 + h_+ \cos(2\pi f(t - z/c) + \varphi_0)\big)dx^2 \\ &+ \big(1 - h_+ \cos(2\pi f(t - z/c) + \varphi_0)\big)dy^2 \\ &+ \big(2h_\times \cos(2\pi f(t - z/c) + \varphi_0)\big)dxdy \\ &+ dz^2 \end{aligned} \tag{1.38}$$

Although this expression seems complex, an intuitive explanation is easily accessible. Let us first consider the case where $h_\times = 0$. In that case, the term in $dxdy$ is dropped and the intervals in the x and y directions are no longer related. However, when $\cos(2\pi f(t - z/c) + \varphi_0)$ is positive, the interval along the x axis is stretched by a value $h_+ \cos(2\pi f(t - z/c) + \varphi_0)$ while the interval along the y axis is shortened by the exact same amount. The reverse happens when $\cos(2\pi f(t - z/c) + \varphi_0)$ is negative. This oscillating motion, seen in the top panel of Fig. 1.3, forms a $+$ sign if we superimpose the largest deformations at $T/4$ and $3T/4$. When dx and dy are inversely affected by a gravitational wave, the latter is said to be plus-polarized.

In the case where $h_+ = 0$, the interval ds can be written as:

$$\begin{aligned} ds^2 = -c^2dt^2 &+ dx^2 + dy^2 + dz^2 \\ &+ \big(2h_\times \cos(2\pi f(t - z/c) + \varphi_0)\big)dxdy \end{aligned} \tag{1.39}$$

We can observe that the only term that differs from the Minkowski space is the term in $dxdy$. To understand the effect of this term, let us rotate our coordinate system by 45°. In the new coordinates x' and y', the expression (1.38) becomes:

$$ds^2 = -c^2dt^2 + \big(1 + h_\times \cos(2\pi f(t - z/c))\big)dx'^2 + \big(1 - h_\times \cos(2\pi f(t - z/c))\big)dy'^2 + dz^2 \tag{1.40}$$

which is equivalent to a plus-polarization. In our (x, y) system of coordinates, the effect of $h_\times$ is therefore equivalent to tilting the effect of h_+ by 45°. The deformation induced on a ring of matter can be seen in the bottom panel of Fig. 1.3. The oscillating motion shows a $\times$ sign and $h_\times$ is therefore called the cross-polarization. Note that pure plus or cross radiation is typically received from systems whose plane motion is viewed edge-on. In nature, gravitational waves rather show a mix of plus- and cross-polarizations. They are said to be elliptically polarized.

1.2.3 Sources of gravitational waves

Just as electromagnetic waves are produced through the acceleration of charged particles, gravitational waves are produced via the acceleration of masses. However, where a dipole of charges is sufficient to generate electromagnetic waves, gravitational waves require at least a quadrupole moment. Monopoles and dipoles are absent as the result of mass and momentum conservation respectively. Einstein [4] derived the expression of the quadrupole formula by solving the linearized field equations with a far-away source

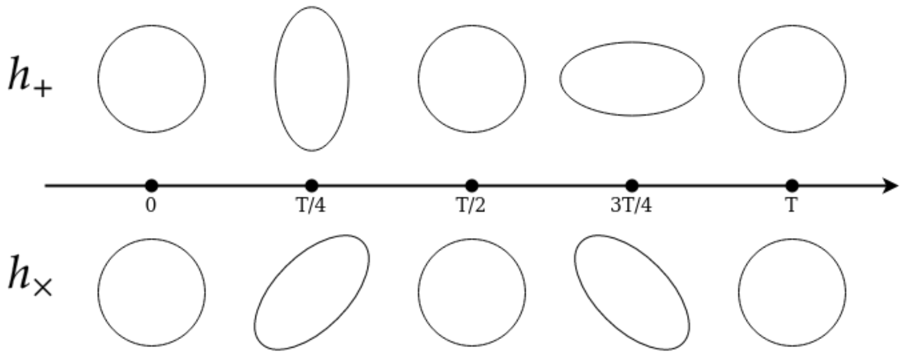

FIGURE 1.3: Effect of the plus- (top) and cross- (bottom) polarizations of a gravitational wave on a ring of matter over a period $T = 1/f$ of oscillations.

term, namely (1.29). His solution shows that the first multipole contribution to the wave amplitude comes from the second time derivative of the quadrupole moment of the source [5]:

$$h_{\mu\nu}^{TT} = \frac{2G}{c^4 r}\ddot{Q}_{\mu\nu}^{TT}(t - r/c) \tag{1.41}$$

where r stands for the distance from the source and $Q_{\mu\nu}^{TT}$ is the quadrupole moment evaluated in the TT-gauge at the retarded time $t - r/c$. This formula is accurate as long as the wavelength of the gravitational wave is much longer than the source size, which is valid for all the sources we will further consider.

In order to figure out which astrophysical phenomena are inclined to produce GWs, let us approximate the above expression. The quadrupole moment of a mass system is approximately equal to the mass that moves M times the square of the size of the system R [5]. The second time derivative is then proportional to:

$$\ddot{Q} \approx \frac{M R^2}{T^2} \approx Mv^2 \tag{1.42}$$

where v is the mean velocity of the source's non-spherical motion[2] and T can be interpreted as the period of the system. Injecting the latter expression into (1.41) leads to:

$$h \approx \frac{G}{c^4 r} Mv^2 \approx \frac{GM}{c^2}\frac{1}{r}\left(\frac{v}{c}\right)^2 \tag{1.43}$$

The first conclusion that can be drawn concerns the mass and velocity of the objects in the system. The latter need to be massive and relativistic to generate strong gravitational waves. A second remark can be raised by looking at the energy radiated over time, known as the GW luminosity L_{GW}. The luminosity can be obtained from the quadrupole moment via its third derivative according to:

$$L_{GW} = \frac{1}{5}\frac{G}{c^5}\langle \dddot{Q}_{\mu\nu}^{TT}\,\dddot{Q}_{TT}^{\mu\nu}\rangle \tag{1.44}$$

[2]Spherically symmetric motions do not yield to a quadrupole moment that is varying in time and thus do not produce gravitational waves.

where $\langle . \rangle$ denotes the average over several wavelengths[3]. Using (1.42), we can estimate L_{GW}:

$$L_{GW} \approx \frac{G}{c^5} \frac{Mv^2}{T} \frac{Mv^2}{T} \approx \frac{G}{c^5} \frac{M^2 v^4}{T^2} \approx Gc \left(\frac{M}{R}\right)^2 \left(\frac{v}{c}\right)^6 \tag{1.45}$$

As R appears in the denominator of (1.45), it is clear that compact systems are preferred to maximize the GW luminosity.

The approximation of h can also be written in terms of signal properties like the energy radiated E_{GW}, the frequency f_{GW} and the duration τ of the gravitational wave [5]:

$$h \approx 10^{-22} \left(\frac{E_{GW}}{10^{-4}\,[M_\odot c^2]}\right)^{1/2} \left(\frac{1\,[kHz]}{f_{GW}}\right) \left(\frac{1\,[ms]}{\tau}\right)^{1/2} \left(\frac{15\,[Mpc]}{r}\right) \tag{1.46}$$

This formula describes the amplitude of a GW emitted from the Virgo cluster (located at roughly 15 Mpc from our galaxy) during which the energy E_{GW} is radiated at a frequency of 1 kHz with a signal duration of 1 ms. This expression summarizes why gravitational waves are hard to detect. If we consider $h \approx 10^{-22}$, a kilometer-size ring of matter would be deformed by an amount:

$$dl = h\,L = 10^{-22}\,10^3 = 10^{-19} m \tag{1.47}$$

The changes in length induced by gravitational waves are therefore extremely faint. Detecting such a faint motion is similar to measuring the distance from the Earth to Neptune with a precision smaller than the thickness of a human hair. Although this task is extremely challenging, one very sophisticated technology has made it feasible: laser interferometry. Chapter 2 describes the inner workings of this scientific and engineering achievement.

From the approximate expression of h, which we will now call the strain, we established that relativistic compact objects are the ideal progenitors of GWs. However, a black hole moving fast along a straight trajectory does not produce GWs. The only necessary criterion is that the system has to show a quadrupole moment for which the second time derivative is not zero. This important result implies that mass systems showing large asymmetric motions of dense masses produce gravitational waves. This is the case for black holes and neutron stars in binary systems. The orbital motion of both objects leads to the production of GWs that extract energy from the system. In turn, this loss of energy leads to the progressive shrinkage of their orbit and eventually to their collision. These events are known as Compact Binary Coalescences (CBC). Fig. 1.4 shows an example of strain h for a collision of black holes associated to the orbital motion of the two objects. Note that although they are less dense than neutron stars, white dwarfs could also produce gravitational waves via binary mergers. They however remain undetectable for current-generation detectors.

Single neutron stars can also produce gravitational waves. If a neutron star is rapidly rotating and not perfectly symmetric about its rotation axis, it will emit a faint continuous GW signal. In order to generate a GW signal with h of the order of 10^{-25}, a 10-kilometer radius neutron star should exhibit an ellipticity of the order $10^{-8} - 10^{-7}$ [6]. Such ellipticities would lead to millimeter-high "mountains" at the surface of these dense objects.

[3]The energy carried by GWs cannot be localized within a wavelength. Instead, one can say that a certain amount of energy is contained in a region of space that extends over several wavelengths [5].

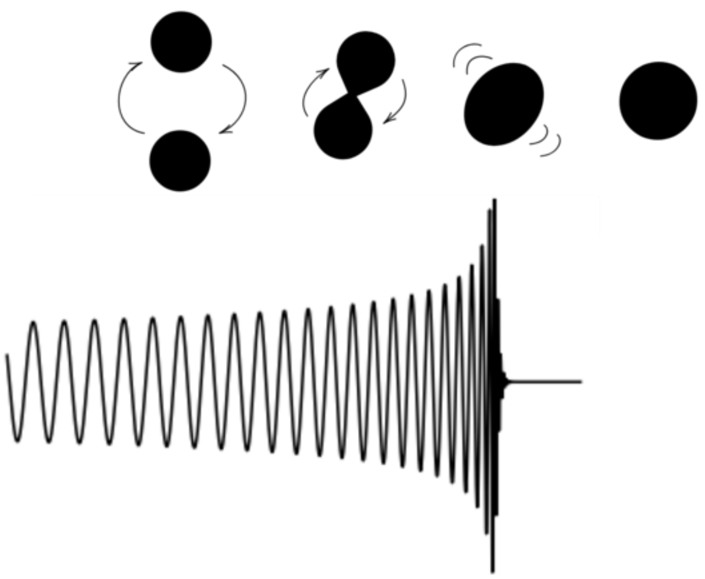

FIGURE 1.4: Representation of a binary black hole merger. The two objects orbit close to each other in the inspiral phase, then get deformed and collide in the merger phase. The final heavier black hole vibrates for a short time during the ringdown phase before returning to a quiet state.

Fig. 1.5 shows the generation process and the corresponding GW signal received on Earth.

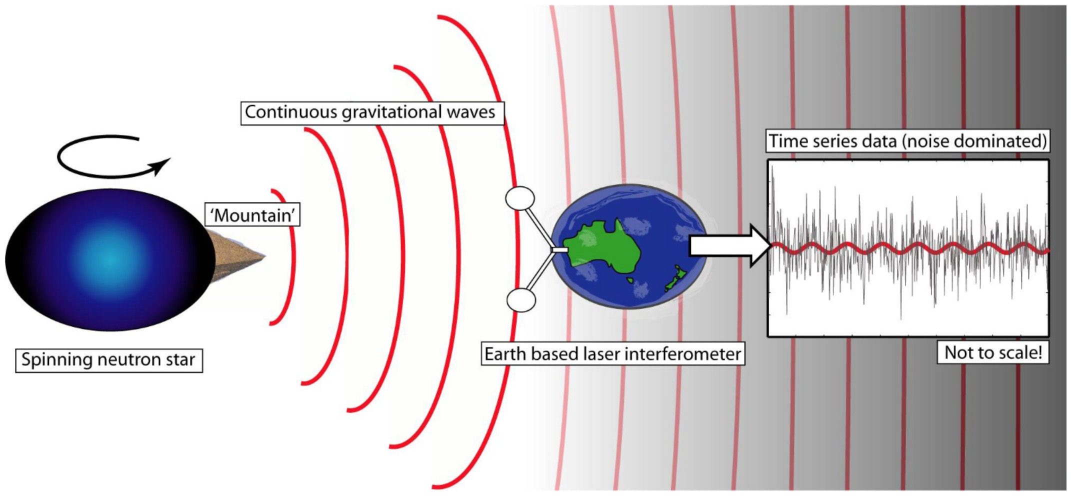
Continuous gravitational waves
Time series data (noise dominated)
'Mountain'
Spinning neutron star
Earth based laser interferometer
Not to scale!

FIGURE 1.5: Non-axisymmetric neutron star emitting continuous gravitational waves. The signal received on Earth is expected to have a quasi-monotonic frequency, equal to twice the rotational frequency of the star. Source: [7]

The third class of gravitational wave events concerns what we call the stochastic background. When a lot of CBC sources generate GWs that ultimately reach the Earth, the resulting signal is a mixture of h_+ and $h_\times$ polarizations arriving with different phases. Resolving each source independently is therefore impossible, leaving data analysts with detecting statistical properties of this background signal [8, 9]. Moreover, the nature of the sources contributing to the stochastic background depends also on the sensitive frequency band of the detectors. For example, stellar-mass binary black holes are expected to be partly responsible for the stochastic background of current-generation detectors [9]. Another GW background signal may arise from early universe sources. Among the possible candidates, we can cite phase transitions, cosmic strings and "preheating" mechanisms after the inflation phase [10, 11]. Detecting such events would have a deep impact on high-energy physics as well as early-universe cosmology.

The last category of GW signals includes all the transient signals different from CBC events. This class is called Burst. Sources such as spherically asymmetric supernovae, disk instabilities in the torus of matter around black holes or gamma-ray bursts are expected to produce burst gravitational waves. As the focus of this thesis is to detect burst signals, more details will be given in Chapter 3.

Chapter 2

Gravitational wave detectors

2.1 The Michelson interferometer

2.1.1 Taking advantage of GW polarizations

The detection of gravitational waves is challenging because of their faint amplitude, h being of the order of 10^{-22} for the strongest events. However, before discussing how we can achieve such sensitivity, let us introduce the basic principles of detection.

If we consider a pure plus-polarized gravitational wave passing through a ring of matter, the latter will be deformed into an ellipse (see Fig. 2.1). The oscillating motion can be detected by looking at the major and minor axes which make a plus sign. The situation is then similar to a gravitational wave passing through a "plus" sign. We thus have to monitor the length of both axes to check for GWs. The deformations induced by the gravitational wave are exactly the same on each side of the major and minor axes. We can therefore reduce the "plus" shaped detector into an "L" structure and save half the materials needed to build our detector. Of course, the orientation of the "L" is primordial in order to be sensitive to both h_+ and $h_\times$. This can be partially solved by building multiple detectors with distinct spatial orientations. More details will be given in Sect. 2.2. Now that we have our optimal shape, we need a way to measure precisely the change in length for the two arms. An 1880s technology is the first step towards this purpose: the Michelson interferometer.

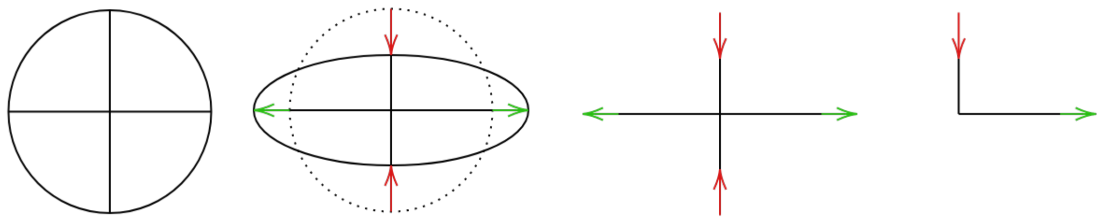

FIGURE 2.1: Representation of the effect of a plus-polarized GW on a ring of matter at phase $3\pi/4$. The red and green arrows respectively indicate contraction and stretching. From left to right, we simplify the design of our detector to end up with an "L" shape structure.

2.1.2 Experimental setup

A Michelson interferometer consists of an optical system capable of measuring interferences. A laser beam is sent through a beamsplitter where half the beam is reflected through a primary mirror while the other half is refracted and sent to another mirror. Both light beams are reflected back to the beamsplitter where they recombine. The resulting interference is then projected onto a screen. Fig. 2.2 shows the setup of the Michelson interferometer when Albert M. Michelson and Edward Morley achieved their famous experiment in 1887. The experiment aimed at detecting the luminous aether, a medium that was supposed to carry light waves. The negative result brought by Michelson and Morley provided strong evidence against the presence of such a medium and paved the way for special relativity.

In their original setup, Michelson and Morley positioned the two mirrors at the exact same distance from the light source. However, one of the two mirrors was movable and tilted with respect to the other mirror. This arrangement leads to the interferogram shown in Fig. 2.2. The interference pattern observed depends on whether or not the mirrors are tilted but also on the difference in the distances of the mirrors to the beamsplitter. If the paths of the beams that reach the detector differ by whole numbers of wavelengths, the constructive interference generates a strong output signal. For differences equal to an odd number of half wavelengths, the interference is destructive and the signal is close to zero.

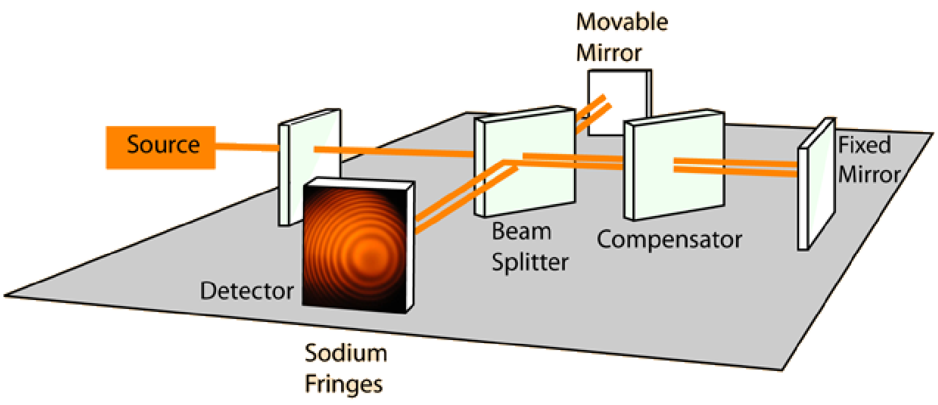

FIGURE 2.2: Original Michelson interferometer setup. Using a beamsplitter, a light beam is divided into two identical beams. Each beam is then sent to a mirror and reflected back to the original beamsplitter. Both beams recombine again to produce an interference pattern that can be seen on the detector screen. Source: [12].

By precisely tuning the difference in path length, we can force the destructive interference of the two light beams. Therefore, a GW detector including an adapted Michelson interferometer could be used to measure the change in length induced by a gravitational wave. In normal conditions, the difference in arms lengths is such that destructive interference takes place at the beamsplitter [3]. If a gravitational wave passes through the instrument, the wave will alternately stretch and compress each arm and the two beams do not annihilate each other anymore. The interferences at the screen could then reveal the passage of a GW.

2.1.3 Towards a real GW detector

Michelson interferometers can be tuned to detect the polarizations of gravitational waves. Moreover, their perpendicular design matches well with the optimal "L" shape derived in subsection 2.1.1. However, a simple Michelson interferometer is not sensitive enough to detect gravitational waves with $h = 10^{-22}$. Let us consider a simple interferometer in order to determine its limitations.

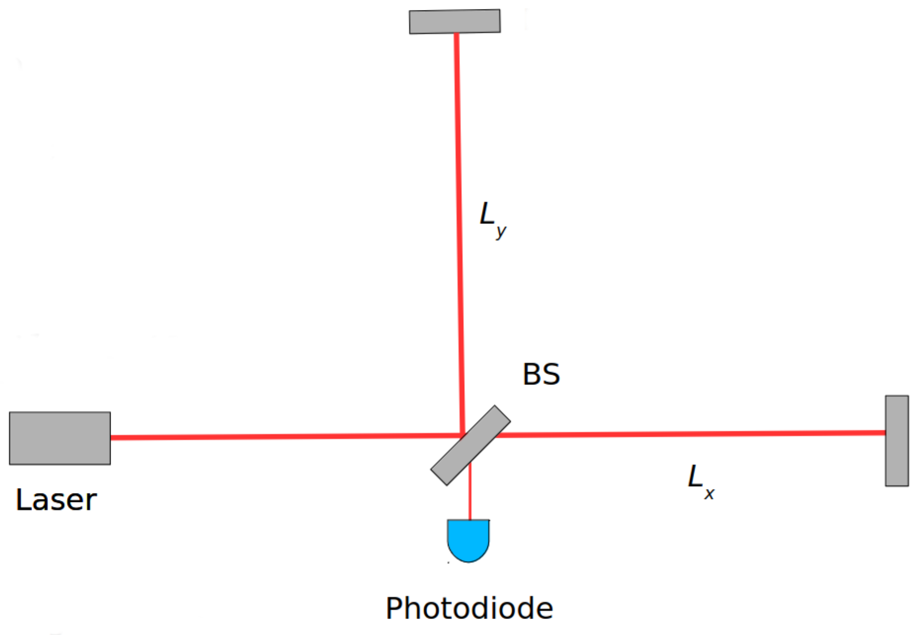

FIGURE 2.3: Setup of a basic Michelson interferometer adapted for gravitational wave detection. The detector screen has been replaced by a photodiode that converts the received light into a current. BS is the acronym for beamsplitter.

Fig. 2.3 shows a basic interferometer setup. A laser producing monochromatic near-infrared electromagnetic radiation is used as well as a beamsplitter (BS). Two perfectly reflecting mirrors are found at the end of the two arms while a photodiode is positioned at the output. The photodiode is a light-sensitive device that produces a current when it absorbs photons. The lengths of the two arms are noted L_x and L_y. For this perfect interferometer, the phase difference at the photodiode is equal to [13]:

$$\Delta\phi = \frac{4\pi}{\lambda}\left(L_x - L_y\right) \tag{2.1}$$

where λ is the wavelength of the monochromatic laser light. In modern interferometers, lasers have a wavelength of 1064 nm. A gravitational wave passing through the detector will cause a differential change in arm length. For example, the lengths of the arms are momentarily given by:

$$L_x = (1+h)\,L \quad \text{and} \quad L_y = (1-h)\,L \tag{2.2}$$

where h is the GW strain amplitude and L is the total length of the arms. Let us assume that we can build an interferometer with arms of 10 kilometers. Considering $h = 10^{-22}$, this yields to measure a phase difference of the order:

$$\Delta\phi = \frac{4\pi}{\lambda}2hL \approx \frac{10}{10^{-6}}\,10^{-22}\,10^{4} \approx 10^{-11} \tag{2.3}$$

A perfect Michelson interferometer is nonetheless impacted by quantum noise [14]. The major quantum noise limiting the sensitivity is the shot noise that originates from the quantum fluctuations in the laser beam. The noise in the phase difference induced by shot noise is given by [15]:

$$\delta\phi_{SN} = \sqrt{\frac{2\,h_{Plank}\,c}{\eta P \lambda}\Delta f} \tag{2.4}$$

with h_{Plank} represents the Plank constant, P is the power of the laser at the photodiode, Δf is the sensitive bandwidth of the interferometer and η stands for the efficiency of the photodiode to convert photons into electrons. Current generation detectors target a frequency bandwidth of the order of 1 kHz. Considering a perfect photodiode ($\eta = 1$), the power required to achieve the sensitivity in phase difference is given by:

$$P_{min} = \frac{2\,h_{Plank}\,c}{\Delta\phi^2 \lambda}\Delta f \approx \frac{10^{-34}\,10^{8}}{10^{-22}\,10^{-6}}\,10^{3} = 10^{5}\,[W] = 100\,[kW] \tag{2.5}$$

The laser systems used in GW detectors can deliver only several hundred watts of power. Most powerful lasers exist but they do not meet the required stability in phase and amplitude [16]. Transmitting hundreds of kilowatts of power through the interferometer would also cause significant thermal deformations of the optics due to absorption [16]. A solution could be to increase the arm length to enhance the phase difference to measure. However, building 100-kilometer-long arms is impossible because of terrain and cost constraints.

An answer to both the power and length limitations is found via Fabry-Perot cavities. A Fabry-Perot cavity is an optical cavity formed by two mirrors that achieves resonance of the light that is trapped inside. Two highly reflecting mirrors are positioned an integer number of half laser wavelengths away from one another, leading to the constructive interference of the light inside the cavity [17]. This constructive interference then amplifies the laser power in the cavity, maximizing the sensitivity of the interferometer to a change in the frequency or wavelength of the input laser. The effect of adding a Fabry-Perot cavity to the arms of the interferometer is similar to multiplying the arm lengths by a factor N_{eff} given by [13]:

$$N_{eff} = \frac{2}{\pi}F \quad \text{with} \quad F = \frac{\pi\sqrt{r_1\,r_2}}{1 - r_1 r_2} \tag{2.6}$$

where r_1 and r_2 are the reflectivities of the first and second mirrors forming the cavity and F is known as the finesse of the cavity. A higher finesse will lead to a more sensitive detector. Current detectors show a finesse in the range 500-1500 with reflectivities higher than 98% [18]. For a 4-kilometer-arm interferometer, the lower margin is equivalent to building a 1280-kilometer detector. There is nonetheless an upper limit to the finesse that is imposed by quantum noise. Indeed, a high finesse will lead to a small bandwidth which could prevent us from detecting some events [14]. The finesse of the cavities therefore determines the sensitive bandwidth of the interferometer, as seen in Fig. 2.4. It is important to point out that the strain has units $1/\sqrt{Hz}$. The strain in Fig. 2.4 is not the strain of the GW but rather the expected strain sensitivity of the detector as a function of the frequency. In this specific case, it is preferred to mention it as the amplitude spectral density of the detector. Further details will be given in Chapter 3 when addressing the burst search methods.

Fabry-Perot cavities are useful to build up the power that circulates in the arms of the interferometer. However, a part of this power is lost and leaves the interferometer towards the laser. Instead of 'wasting' that power, we can insert a semi-transparent power-recycling mirror to send the light back to the interferometer. A new resonant cavity is formed between this mirror and the rest of the interferometer so that little light comes back toward the laser. The effective laser power is then enhanced by a factor of 10 to 50 [15] which is called the recycling gain.

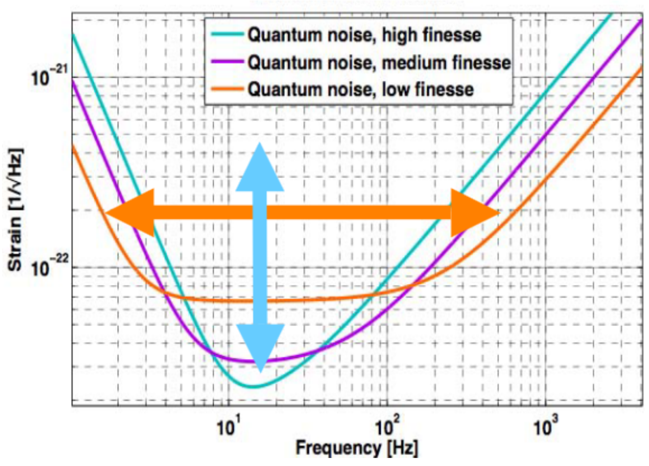

FIGURE 2.4: Sensitivity to quantum noise of a Fabry-Perot Michelson interferometer. A high finesse of the Fabry-Perot cavity leads to an improved sensitivity but also to a tighter bandwidth. Note that the strain has units $1/\sqrt{Hz}$. Source: [14].

The Fabry-Perot cavity is tuned to resonate at the particular frequency of the laser light. However, in the cavity, sidebands are generated on the light by gravitational wave signals interacting with the arms. These sidebands, carrying power, do not interfere destructively at the beamsplitter and therefore appear at the output. If a mirror is put ahead of the photodetector, these sidebands can be recycled back into the system where they resonate. Their amplitude is thus enhanced over a given bandwidth which can be tuned by choosing the adequate reflectivity of the mirror. This technique, called signal recycling, is particularly useful to narrow the detection bandwidth, which may be valuable in searches for continuous waves [19].

The final design of our interferometer adapted to gravitational wave searches can be seen in Fig. 2.5. This view constitutes the basic design on which the current generation GW detectors, namely LIGO and Virgo, have been built.

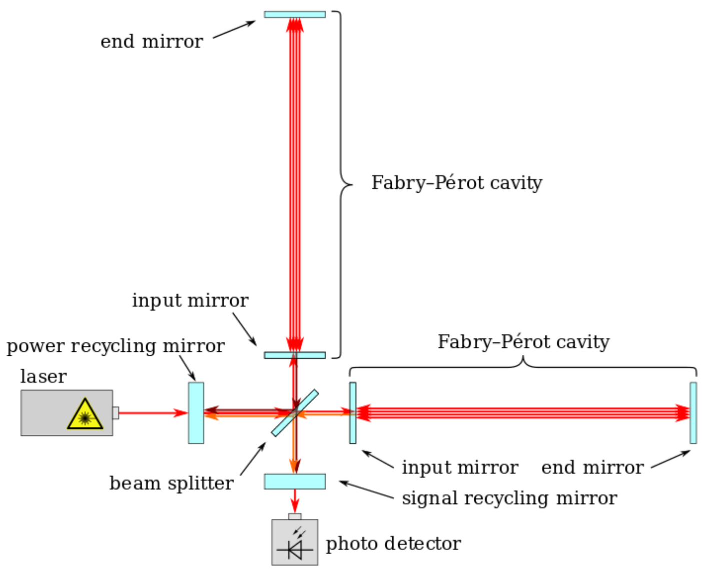

FIGURE 2.5: Advanced interferometer design showing Fabry-Perot cavities, a power recycling mirror and a signal recycling mirror. Source: [20]

2.2 Current generation of detectors

2.2.1 The LIGO, Virgo and KAGRA collaborations

On the 11th of February 2016, the Laser Interferometer Gravitational-wave Observatory (LIGO) and the Virgo collaboration jointly announced the first direct observation of a gravitational wave [21]. The 2017 Nobel Prize in Physics was awarded to three scientists that initiated the LIGO project, namely Rainer Weiss, Kip Thorne and Barry Barish for this exceptional discovery. LIGO consists of two "L" shaped interferometers with 4 km arms located 3000 km apart within the United States, one in Hanford Washington and the other in Livingston, Louisiana. The LIGO detectors are operated by the LIGO Scientific Collaboration (LSC). Funded in 1997, the collaboration is currently made up of more than 1500 scientists over 100 institutions spread over 18 countries worldwide.

In Europe, the European Gravitational-wave Observatory (EGO) is the operating center of the Virgo detector, a 3km interferometer situated in Cascina, Italy. EGO is partially funded by three agencies in France (CNRS), Italy (INFN) and in the Netherlands (Nikhef). More than 650 members work as a community on the development and operation of the Virgo detector, forming the Virgo collaboration.

The third observatory is an underground detector with 3 km arms located in Kamioka, Japan and called KAGRA. KAGRA is a project initiated by the Institute for Cosmic Ray Research (ICRR) at the University of Tokyo. Although it became operational in February 2020, the current sensitivity of the Japanese detector prevents it from actively participating in the detection of GW events. Developments of innovative technologies such as cryogenic mirrors are currently being improved to reach LIGO and Virgo operational sensitivities. More than 400 individuals from 15 countries compose the KAGRA collaboration.

LIGO and Virgo collaborations have agreed to share their data and to carry out joint analyses since 2010. The observational results of the numerous analyses are found in scientific papers whose authorship is shared by the two collaborations. In early 2021, the KAGRA collaboration joined its two sister collaborations. Together, they constitute the LIGO-Virgo-KAGRA (LVK) Collaboration. Fig. 2.6 shows the location of the GW detectors on Earth.

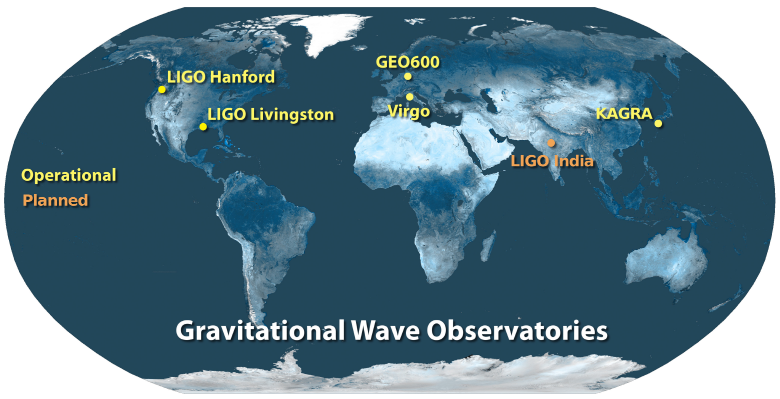

FIGURE 2.6: Gravitational wave detectors around the world. GEO600 is a 600-m long interferometer based in Hannover (Germany) that serves as a prototype and testing facility. LIGO India is a duplicate LIGO observatory that is expected to operate by 2030.

2.2.2 Observing runs and upgrade phases

The current GW detectors are the culmination of more than four decades of work. To achieve strain sensitivities of the order of $h = 10^{-22}$ and beyond requires to proceed step by step. That is why the interferometers are not continuously gathering data as it is the case for classical optical telescopes. Instead, the interferometers are turned off regularly to install new instruments contributing to increase the overall sensitivity of the detectors or to upgrade them. The periods where detectors are "on" are called observing runs (designated "O1", "O2", etc.) while the inactive intervals are known as upgrade (or commissioning) phases. Since the GW detectors are improved during the upgrade phases, their strain sensitivities always increase from one observing run to the following.

Fig. 2.7 presents the observing plans as well as the foreseen sensitivities from 2015 to the end of the current decade. Note that the sensitivities are not expressed in terms of the best strain sensitivity that can be achieved. Alternatively, the sensitivity is indicated with respect to the Binary Neutron Star (BNS) range of the detectors. The BNS range can be defined as the volume- and orientation-averaged distance at which the coalescence of a pair of 1.4 solar mass neutron stars gives a signal-to-noise ratio of 8 in a single detector [22]. It is important to note that a factor 2 improvement in the BNS range leads to a volume 8 times larger, and so to potentially 8 times more recorded signals.

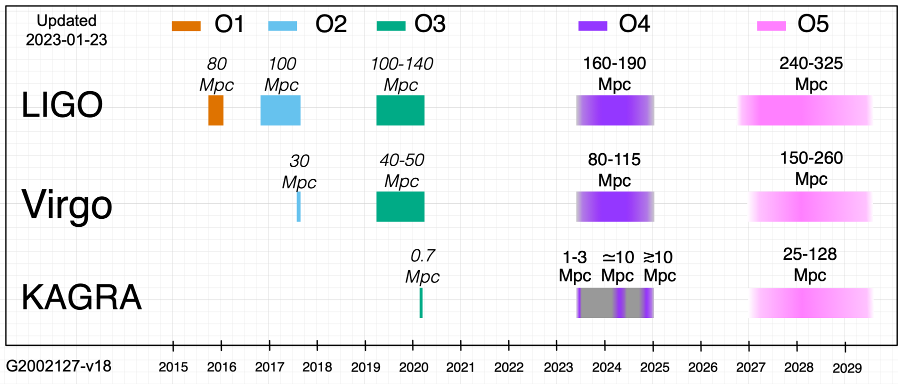

FIGURE 2.7: LIGO, Virgo and KAGRA observing plans. The O5 start date, duration and sensitivities are current projections that will likely be adjusted after O4. Credit: LIGO-Virgo-KAGRA collaborations.

To this date, 3 observing campaigns have been concluded. Over the first 3 observing runs, the data analysis pipelines have recorded a total of 90 gravitational wave events in the LIGO and Virgo data [23, 24, 25]. As the BNS ranges of the detectors have been significantly improved, most of the detections happened during O3. Fig. 2.8 shows the cumulative histogram of the detections with respect to the number of observing days. Among the 90 detections, 86 were Binary Black Hole mergers, 2 originated from BNS collisions [26, 27] and 2 came from Neutron Star-Black Hole events (NSBH) [28].

2.2.3 Sources of noise

The LIGO and Virgo detectors, whose latest status are known respectively as aLIGO [29] and Advanced Virgo [30], are limited by many sources of noise. Although they are improved versions of the Michelson interferometer (designated as Dual-Recycled Fabry-Perot Michelson interferometers), they still suffer from some limitations inherent to the original design.

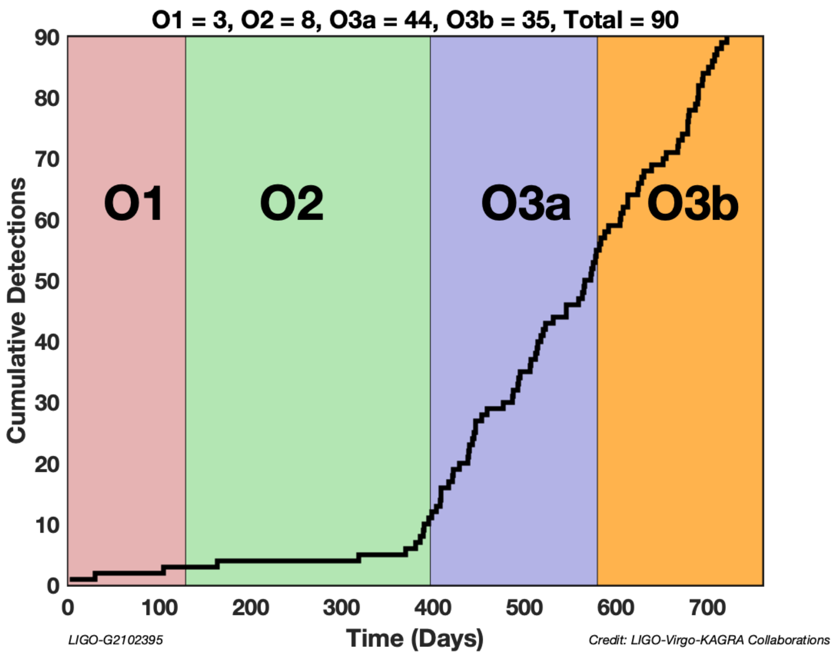

FIGURE 2.8: Cumulative number of detections as a function of the number of days in each run period. Note that the third observing run O3 has been split into 2 periods but no upgrades were performed between O3a and O3b.

Quantum noise

Quantum noise impacts the interferometer in two different ways, via shot noise and radiation pressure noise.

Shot noise arises from quantum fluctuations in the arrival of photons at the photodiode. The photons in the laser beam are not equally distributed, rather they follow a Poisson statistic. The photodiode therefore absorbs photons at an irregular rate, leading to a noisy output voltage as it can be seen in the left panel of Fig. 2.9. In current generation detectors, shot noise is the dominating source of noise at high frequencies [13].

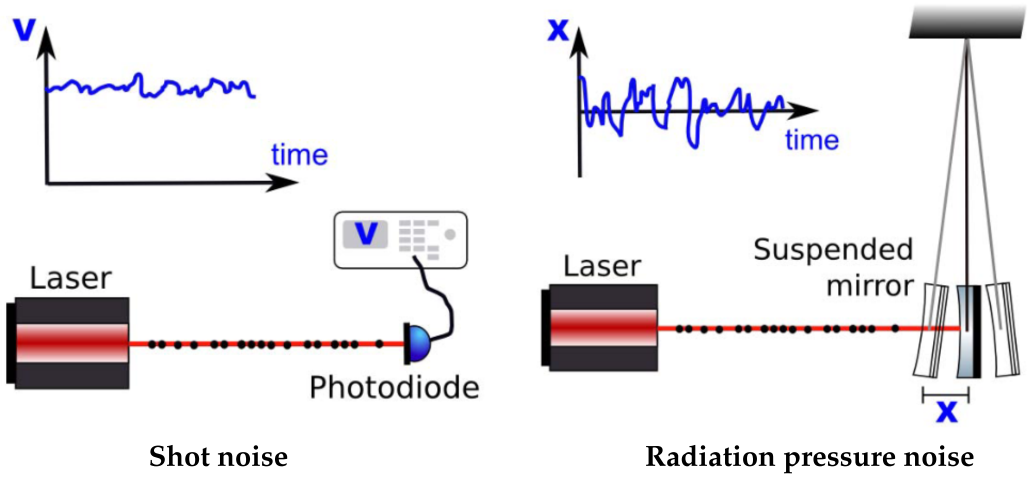

FIGURE 2.9: Quantum noise affecting the current generation detectors. Shot noise (left) leads to uncertainties in the output voltage while radiation pressure (right) induces undesired motions of the mirrors. Source: [14].

Radiation pressure is related to the impact of photons on the mirrors. When a photon hits one of the mirrors, it transfers a part of its momentum to the mirror causing the latter to move. In turn, this motion modifies the length of the cavity leading to a change in the phase of the laser light. The interference at the beamsplitter is therefore affected and a fake signal is recorded at the photodiode. The right panel of Fig. 2.9 shows a representation of this effect. The effects of radiation pressure are mostly prevailing in the low-frequency regime [13].

In an attempt to reduce shot noise, we can enhance the power of the laser [20]. By doing so, we increase the number of photons in direct proportion to the laser power. As the noise induced in the photon arrival rate is proportional to the square root of the power [14], the overall signal-to-noise ratio is therefore improved. However, as the power of the laser increases, the transfer of photon momentum to the mirrors also increases. This introduces more uncertainty in the length of the arm and cancels out any gain due to reduced shot noise. As both noises impact the interferometer sensitivity at different frequency regimes, the trade-off in the laser power causes quantum noise to shape the overall sensitivity of the detectors. Fig. 2.10 illustrates how shot noise and radiation pressure noise affects the sensitive bandwidth of the GW detectors.

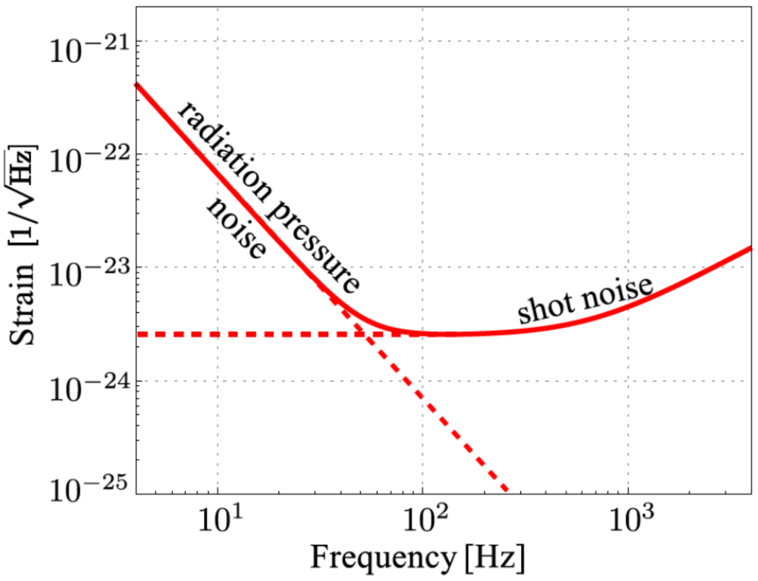

FIGURE 2.10: Theoretical sensitivity curve imposed by shot noise and radiation pressure noise for the aLIGO detectors. Shot noise limits the strain sensitivity of the detectors at high frequencies while radiation pressure effects are dominant at low frequencies. Source [31].

A new technique has recently been introduced into both LIGO and Virgo detectors to overcome the quantum limit imposed by radiation pressure and shot noises: squeezing. Both sources of noise arise from the quantum uncertainties in the laser light. Although photons from the laser are in a coherent state, there exist minimal uncertainties that impact both the phase and amplitude of the light. Phase uncertainties cause the rate of photons in the laser to be irregular, leading to shot noise in interferometers [32]. Amplitude uncertainties affect the transfer of momentum from photons to the mirrors and then show up as radiation pressure noise. Squeezing is a technique that allows to reduce the uncertainties either in phase (phase squeezing) or amplitude (amplitude squeezing) at the price of an increase in the uncertainties of the other property. Phase squeezing is therefore useful to reduce shot noise at high frequencies while amplitude squeezing improves the radiation pressure noise at low frequencies. Achieving both squeezed states at different frequencies is known as Frequency Dependent Squeezing (FDS). Fig. 2.11 shows the effect of squeezing on the theoretical noise curve in LIGO. FDS has been introduced into both the aLIGO [33] and Advanced Virgo detectors [34] prior to the third observing run. Overall, squeezing produced a sensitivity enhancement up to 8% in the BNS range of Advanced Virgo and 15% in aLIGO.

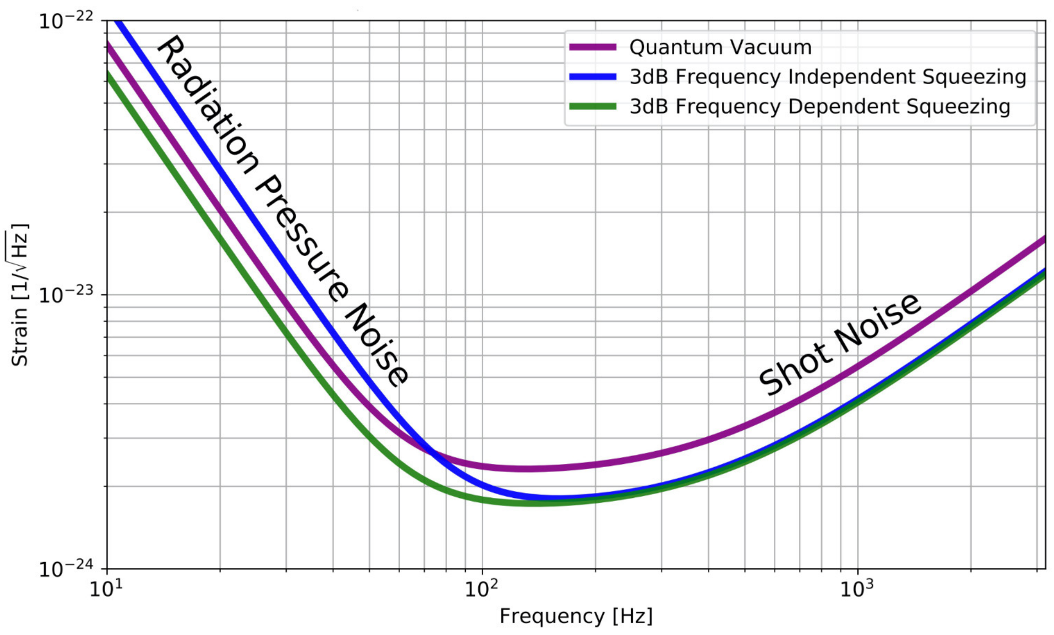

FIGURE 2.11: Effect of Frequency Dependent Squeezing on the quantum noise in LIGO. Source [32].

Displacement noise

Because LIGO and Virgo are built on the ground, they are susceptible to ground motion. GW detectors are so sensitive that they can detect cars on highways kilometers away as well as almost every earthquake that happens on Earth. Even though it is a remarkable achievement, these phenomena are sources of noise that cause unwanted vibrations of the mirrors, perturbing the search for gravitational waves. Mitigating them is therefore of primordial importance if we want to achieve the desired sensitivity. That is why LIGO and Virgo mirrors are hung by an assemblage of pendulums. Each pendulum has a different length and so a different resonance frequency. By stacking them one above the other, it is possible to reduce significantly the external perturbations in a broad range of frequencies, with the exception of the particular natural frequencies of the pendulums. The overall suspension system allows the mirrors to behave as effective free-falling masses along the axes of the laser beam. The design of the multi-stage pendulum of both LIGO and Virgo can be seen in Fig. 2.12. This passive structure works together with active stages using feedback control loops to achieve more than 10 orders of magnitude attenuation above 10 Hz.

Among the sources of noise, there is one that cannot be shielded: gravity gradients [15]. Gravity gradients arise from fluctuations in the gravitational field of the ground below the detectors. These fluctuations are usually caused by microseismic motion that modifies the local mass density by very tiny amounts. Wind, rain or human activities could also be

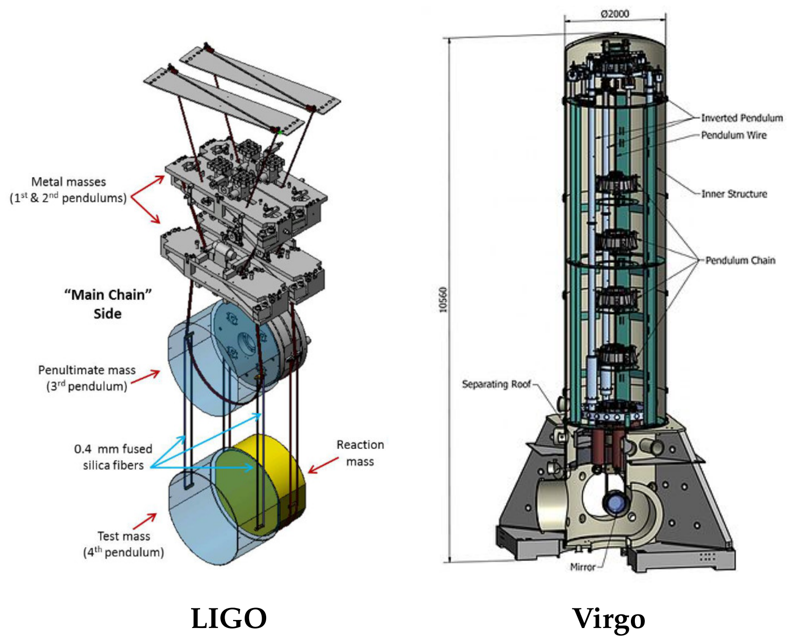

FIGURE 2.12: Schematic representation of the LIGO and Virgo multi-stage pendulums. The LIGO suspension system is composed of 4 stages while Virgo has 7 stages. The mirrors are suspended via fused silica fibers on the lowest level. Credit: Caltech/MIT/LIGO Lab and Virgo collaboration.

responsible for these density fluctuations. As the gravitational field in the vicinity of the detector changes, the orientation of the mirrors can be slightly impacted. The strain sensitivity noise due to gravity gradients is expected to be proportional to f^{-4} [15], which affects mainly the low-frequency regime. Building underground interferometers is a solution to minimize the impact of gravity gradients, as the density fluctuations are damped few hundreds of meters below the ground level.

Thermal noise

In the absence of any motion, thermal physics dictates that every object has a minimal state of energy depending on its temperature. Therefore, the strings holding the mirrors (i.e. fused silica fibers) can undergo thermal excitations of their natural vibration modes due to their inner temperature [20]. This leads the mirrors to vibrate at the resonant frequencies of the fibers (at 500 Hz and harmonics: 1000, 1500, 2000 Hz), called violin modes.

The mirror surface is also impacted by thermal noise in the form of Brownian motion. Indeed, the temperature of the coating's molecules causes the surface to vibrate, which in turn slightly affects the length of the cavity producing a change in the phase of the laser beam. The coating Brownian noise is responsible for the largest contribution to the overall detector noise by the optical components of the interferometers [29, 30].

Other sources of noise originate from the optics such as thermo-elastic noise and thermo-refractive noise. Thermo-elastic noise is the apparent expansion of the mirror coating into the probe beam causing a change in phase [35]. Thermo-refractive noise arises from the changes in both the refractive index and the width of the coating layers due to the temperature [35]. These two sources of noise are referred to as thermo-optic noise. In current generation detectors, the thermo-optic noise is one order of magnitude below the coating Brownian noise [35].

Residual gas noise

LIGO and Virgo are so sensitive that they are susceptible to exceptionally faint disturbances like sound vibrations and dust in the air. The former can refract the laser beam as well as cause parasite motion of the mirrors [20]. Dust sticks to the mirror surfaces and scatters the laser beam, decreasing the coherence of the laser. Even the impact of air molecules on the mirrors can introduce detectable motions. All these sources result in fluctuations in the path length and must be eliminated to achieve the desired sensitivity. One strategy allows to address all of them: operating the experiment in a vacuum. It consists in enclosing the laser beam in large vacuum tubes for which the air is evacuated down to a pressure of about 10^{-9} mBar. With respectively 7000 and 10000 cubic meters of vacuum, Virgo and LIGO are among the largest ultra-high vacuum chambers ever built [20]. The remaining excess gas in these chambers is taken into account in the noise budget although it is of the same order of magnitude as the thermo-optic noise.

Noise budget

After listing the primary source of noise in the GW detectors, we can estimate the overall sensitivity as a function of the frequency. This is done in Fig. 2.13 where the noise budget of the LIGO and Virgo interferometers are shown. As we can see, the sensitivity is shaped by the quantum noise, especially at high frequencies. Gravity gradients and seismic noise limit the sensitivity of the detectors at low frequencies while the thermal noise from the mirrors' coating has a significant impact at frequencies around 50 Hz to several hundreds of Hz. Both detectors achieve their best sensitivity around 300 Hz.

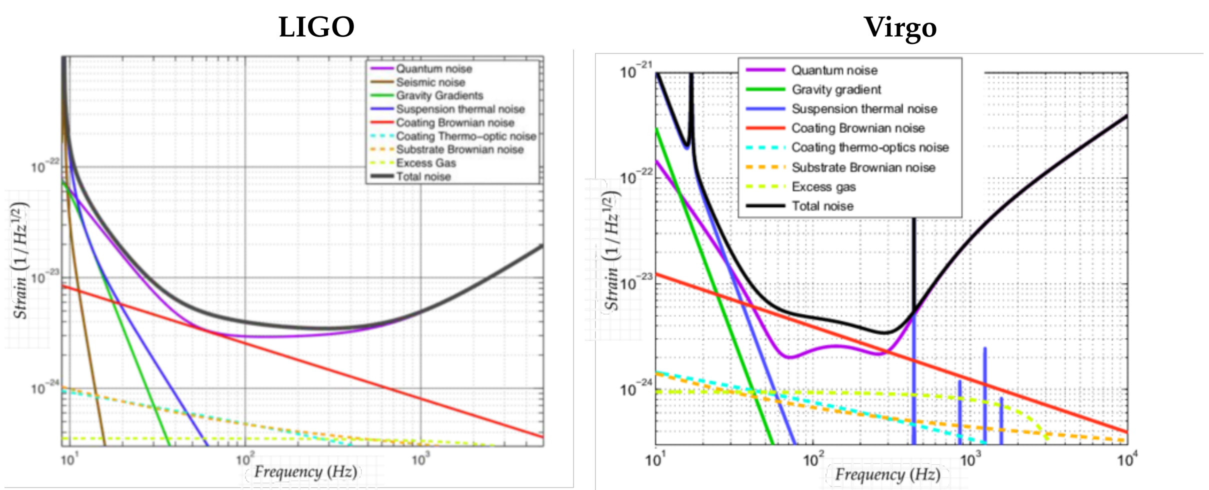

FIGURE 2.13: Design sensitivity of the LIGO and Virgo detectors. The black curve represents the total sensitivity, accounting for all the sources of noise. Quantum noise is the limiting noise at high frequencies while seismic and thermal noises affect the sensitivity in the low-frequency regime. Sources: [29, 30].

In practice, the sensitivity is spoiled by various technical noises such as instrument coupling to environmental noises (magnetic, seismic, acoustic, etc.), scattered lights in the cavity due to small defects at the surface of the mirrors or electrical readout noise [36]. The noises known as instrumental lines appear in the sensitivity as vertical peaks. The most prominent peaks arise at harmonics of the electric power line frequency (60 Hz for LIGO in the US, 50 Hz for Virgo in Europe), due to imperfect electromagnetic shielding and magnetic coupling to the mirror suspensions [37]. It is also important to mention the thermally excited violin modes of the silica fibers that induce peaks at 500 Hz and harmonics (1000, 1500, 2000 Hz). Both the LIGO and Virgo measured sensitivity (also known as amplitude spectral density) are shown in Fig. 2.14.

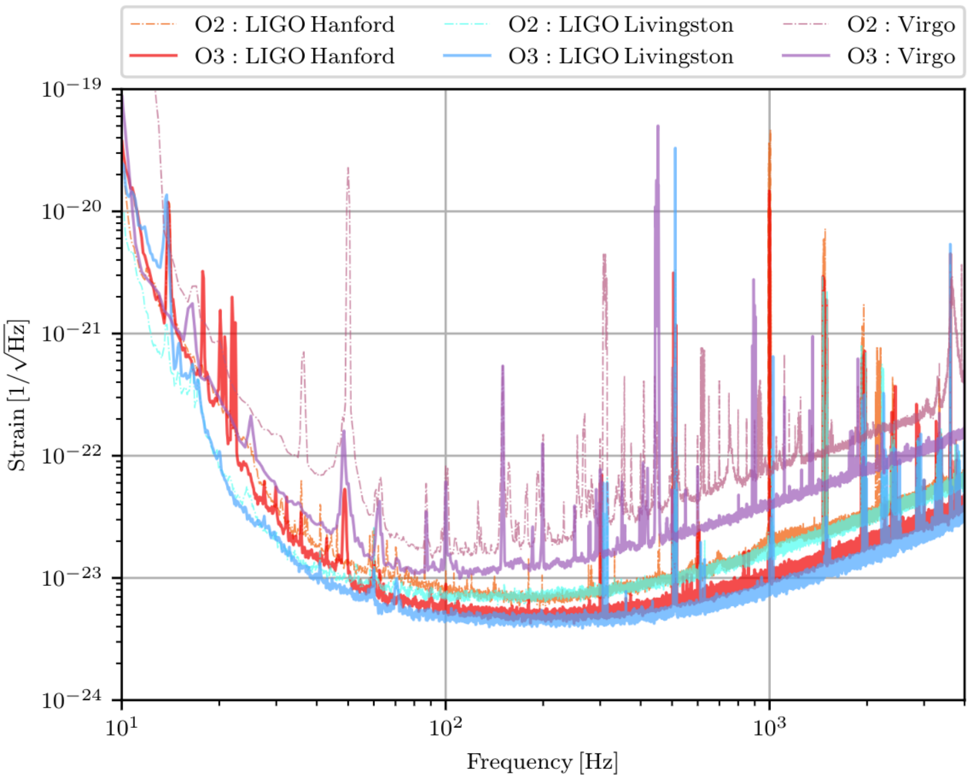

FIGURE 2.14: Representative amplitude spectral density of the LIGO and Virgo detectors for the O2 and O3 runs. The curves shown for O3 are taken from O3a. Source: [36].

2.2.4 Lock and duty cycle

The Fabry-Perot cavities installed in the current GW detectors allow to build up the power circulating in the arms exclusively when the laser light resonates. This holds for the other cavities such as those formed with the signal recycling mirror and the power recycling mirror [38]. For example, aLIGO has five length degrees of freedom that must be controlled in order for the interferometer to be operational. All these degrees of freedom, strongly coupled, need to be tuned simultaneously between 0.01% and 1% of a fringe [39], making the acquisition of the operating point very challenging. When all the cavities work on resonance, we say that the interferometer is "locked".

Because of the tight adjustment of the cavities, exceptional ground motions can cause LIGO and Virgo to lose the lock in one or more optical cavities. As a consequence, the whole detector suffers and the strain sensitivity drops, resulting in a reduction of the effective observation time. Earthquakes, microseisms, human activities and winds are all sorts of incidents that are considered as potential lock loss events [40]. Studies have shown a correlation between local ground motions and losses of the interferometer operating state [41, 42]. Fig. 2.15 shows the effect of repeated earthquakes in a 24-hour period that happened during the ninth engineering run[1] in 2015. We can clearly see the correlation between the peak ground motions (bottom) and the steep drop in the BNS range (top) of the LIGO Livingston detector.

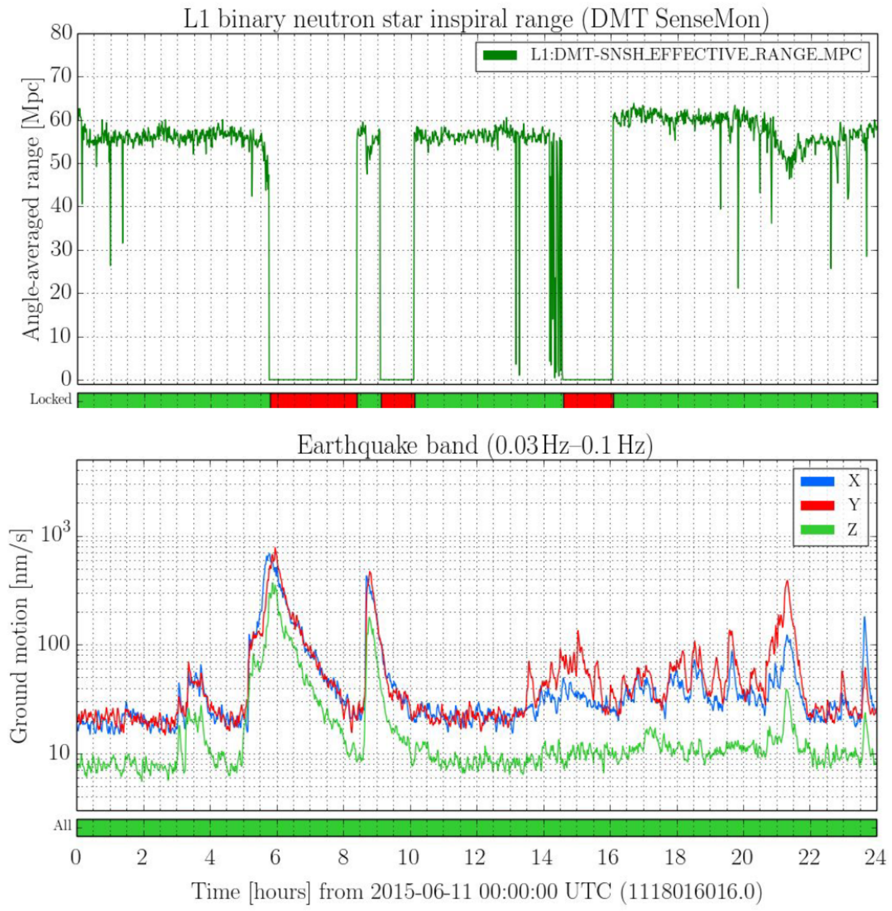

FIGURE 2.15: Top: BNS range of the Livingston detector on 11 June 2015. Bottom: Ground motion around the detector on the same day. The X and Y axes point in the direction of the two arms while the Z axis is directed from the ground to the sky. Source: [44].

The length of the cavities, as well as accurate measurement of the ground motions and other external factors, are recorded in the form of time series known as auxiliary channels [45]. By using the information retained in these channels, it is possible to infer which component of the detector is susceptible to causing a lock loss. The auxiliary channels are therefore vital to predict a likely lock loss and engage the detector in a "lock loss robust" state [45]. Together with active control loops, these channels help resume the operation much more quickly than the minimal hours-days otherwise required.

The time percentage during which the detectors are in observing mode is called the duty cycle. It is critical to have a high duty cycle to achieve a lot of gravitational wave detections. Most importantly, a network of detectors is primordial to confirm detections and localize the sources of the event, especially in the case of an electromagnetic counterpart [26]. Fig. 2.16 shows the operating status of the two LIGO detectors (Hanford H1, Livingston L1) and the Virgo detector for the third observing run O3. All of them are

[1]Engineering runs are periods where the detectors are operated as in an observing run. These runs are scheduled at various stages of installation or detector configuration to characterize the effect of the recent upgrades and to fix the remaining bugs in hardware and software [43].

locked more than 70% of the time, resulting in 80% observing coverage with two or three interferometers.

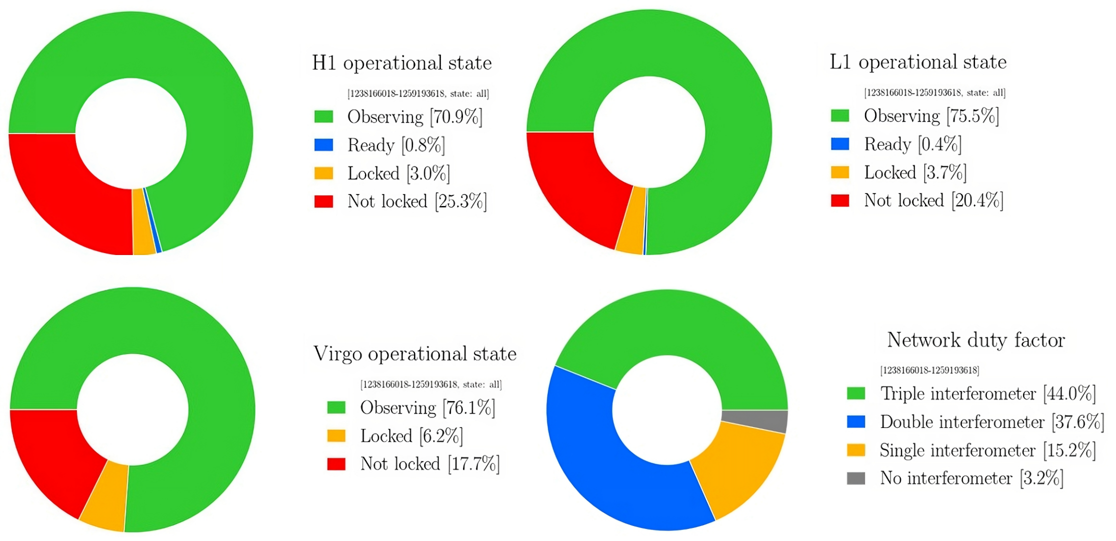

FIGURE 2.16: Percentage of time that the aLIGO and Advanced Virgo detectors spent in different operating modes during the third observing run. The overall network duty cycle is also shown in the bottom right panel. "Observing" refers to the periods where the detectors were locked and the data quality was sufficient for data analysis. "Locked" and "ready" refer respectively to the periods where the operation team achieved the lock and to the periods where some complementary maintenance was performed. Credits: LIGO Detector Characterization team.

2.2.5 Calibration

Once the lock is achieved, there is a need to convert the output of the photodetector into the strain $h(t)$. Since aLIGO and Virgo employ active feedback loops to keep the optical cavities in resonance, the calibration process should include models of the electronics used. Fig. 2.17 illustrates the feedback control as well as the calibration pipeline for the aLIGO detector. There are three main components to the feedback loop: the sensing function S, the digital filters applied D and the actuation function A. Since the digital filters are known with great precision, the main uncertainties in the calibration come from the sensing and actuation functions, S and A [46].

The feedback procedure works as follows. At first, the photodetector, which is an element of the sensing C, measures the residual differential arm length ΔL_{res}. The output d_{err} of the sensing function is then sent through digital filters D and applied to the differential arm length actuators through the actuation function A. These actuators are used to suppress ΔL_{crtl} from the desired quantity ΔL_{free} and therefore keep the interferometer locked. In order to get a faithful representation of ΔL_{free}, we model A and C via photon calibrators [48]. Photon calibrators (PC) are auxiliary lasers assemblies that send a pair of beams to the mirrors. These lasers, whose intensities are modulated at a known amplitude and frequency, allow to measure A and C as functions of frequency and yield to a displacement x_T^{PC} at the mirrors (via radiation pressure). Note that some sinusoidal forces are applied at very specific frequencies during the calibration process, leading to calibration lines appearing in the raw strain data. For O2 and onwards, these lines are removed before releasing the data [49, 50]. Once A and C are known, the strain h is obtained by

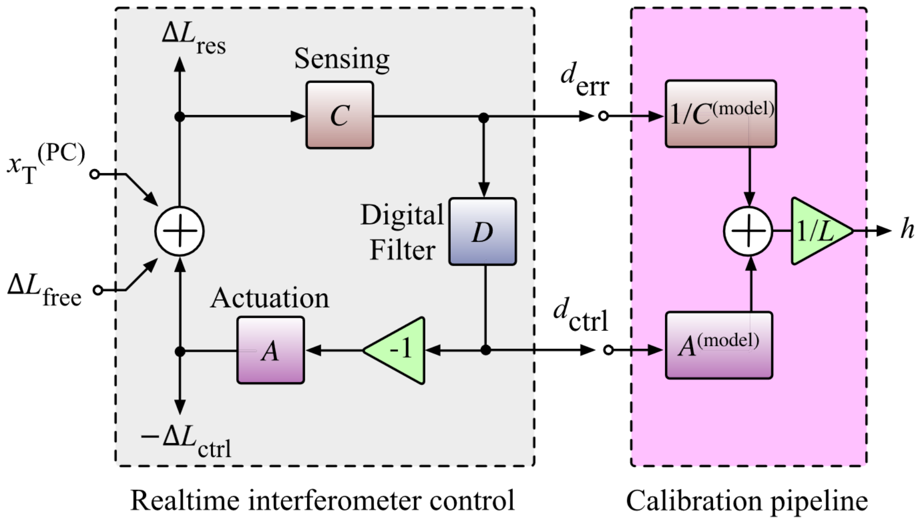

FIGURE 2.17: Control loop (left) and calibration procedure (right) of the Advanced LIGO detector. The sensing function, comprising the photodiode, provides the output d_{err} in response to the residual differential motion ΔL_{res}. The output is then passed through digital filters D, which produces a control signal d_{ctrl}. The actuation system A processes d_{ctrl} and acts on the optics by ΔL_{ctrl} to maintain cavity resonance. The sensing and actuation functions are modeled through the calibration pipeline using photon calibrators. The output of the calibration procedure is the GW strain data h(t). Source: [47].

dividing the true differential arm length by the length of the arm (4 km for LIGO, 3 km for Virgo) via the following expression [46]:

$$h(t) = \frac{1}{L}\left(\mathcal{C}^{-1} * d_{err}(t) + \mathcal{A} * d_{ctrl}(t)\right) \tag{2.7}$$

where $\mathcal{A}$ and $\mathcal{C}$ are time-domain filters derived respectively from the actuation and sensing functions $A(f)$ and $C(f)$.

The calibration procedure described above is applied to the photodiode output respectively 16384 times and 20000 times per second for LIGO and Virgo data. The strain $h(t)$ is therefore a time series sampled at 16384 Hz for LIGO and 20 kHz for Virgo [46]. It is fundamental to note that the strain $h(t)$ is only calibrated between 10 Hz and 5 kHz for Advanced LIGO and 10 Hz and 8 kHz for Advanced Virgo [50]. The reconstruction of the strain is not faithful at lower or higher frequencies.

2.2.6 Angular sensitivity

As described in subsection 2.1.1, interferometers take advantage of the polarization modes of gravitational waves in order to detect them. However, the sensitivity of the LIGO and Virgo detectors highly depends on the source location in the sky. Indeed, the interferometer response is maximized for GW propagating orthogonally to the plane of the arms and linearly polarized[2] [51] while any other angles of incidence or polarizations give a reduced response. It is possible to derive the sensitivity of the detector as a function of the direction of the GW. This sensitivity is known as the antenna pattern of the detector and is shown in

[2]We say that a gravitational wave is linearly polarized when it shows exclusively the plus- or cross-polarization.

Fig. 2.18. As we can see, the antenna pattern has a very peculiar peanut shape, indicating that GW detectors are more sensitive to GWs coming directly from above or below them.

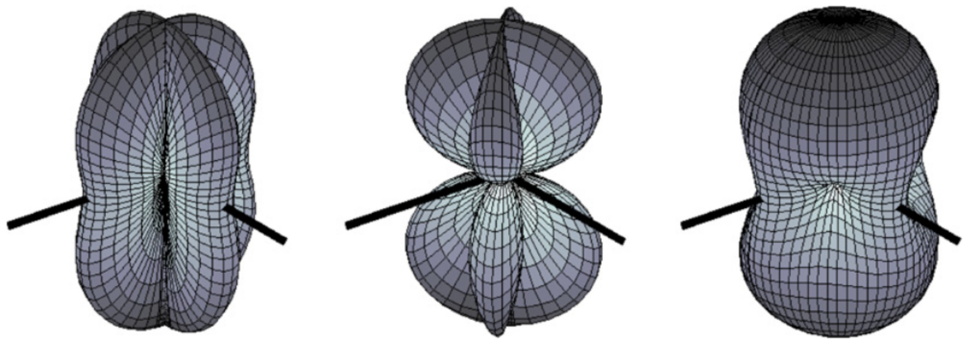

FIGURE 2.18: Antenna pattern for a L-shaped GW detector in the long-wavelength approximation (i.e. the wavelength of the GW is much larger than the size of the detector). The black lines indicate the orientation of the two arms with the beamsplitter at their center. The sensitivity for h_+, $h_\times$ are shown respectively in the left and middle panels. The right-most pattern is for the root mean square of h_+ and $h_\times$. Source: [51].

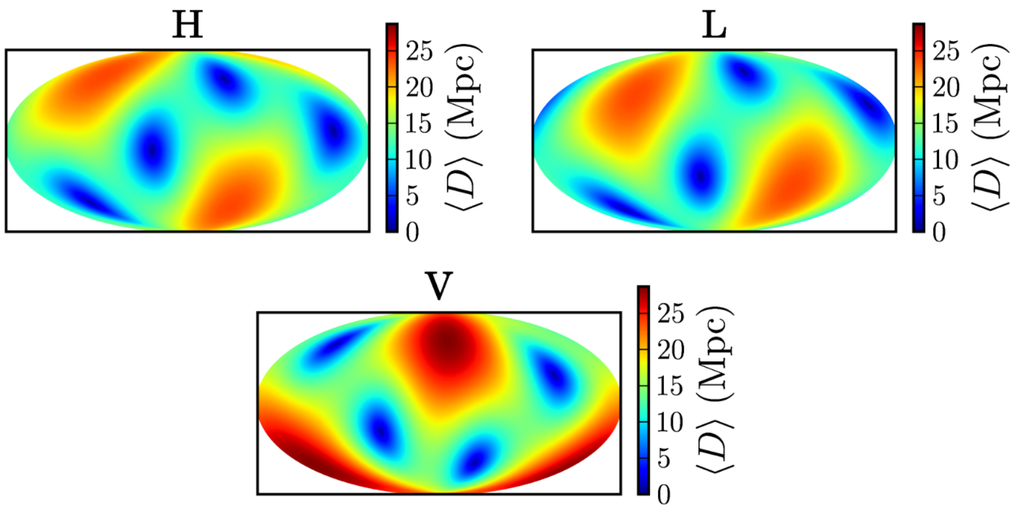
H
25
20
15
10
5
0
$\langle D \rangle$ (Mpc)
L
25
20
15
10
5
0
$\langle D \rangle$ (Mpc)
V
25
20
15
10
5
0
$\langle D \rangle$ (Mpc)

FIGURE 2.19: Angle-averaged sky maps of the LIGO Hanford (H), LIGO Livingston (L) and Virgo (V) detectors for their design sensitivities. The color scale indicates the BNS range, noted here $\langle D \rangle$. Source: [52].

When projected onto the sky, the antenna pattern shows the sky sensitivity of each detector. Fig. 2.19 represents the sky maps of the LIGO and Virgo detectors in their initial design state in spherical coordinates (Mollweide projection). We can clearly see the two preferred directions of each detector and the 4 blind spots corresponding to GWs propagating in the plane of the arms, as shown in Fig. 2.18. Another remark concerns the important role of the Virgo detector. In addition to confirming and localizing GW events, Virgo is undoubtedly more sensitive than LIGO to GWs coming from certain sky directions. This highlights the importance of having a network of detectors covering the whole sky [52]. This comment finds a confirmation in Fig. 2.20 where the antenna patterns $|F_+|$ and $|F_\times|$ are shown for the HL and HLV networks. The sky sensitivity of a network of 3 detectors has fewer blind spots than any network of 2 detectors, allowing a wider sky coverage. The current generation network's sensitivity will be further improved with KAGRA joining the fourth observing run.

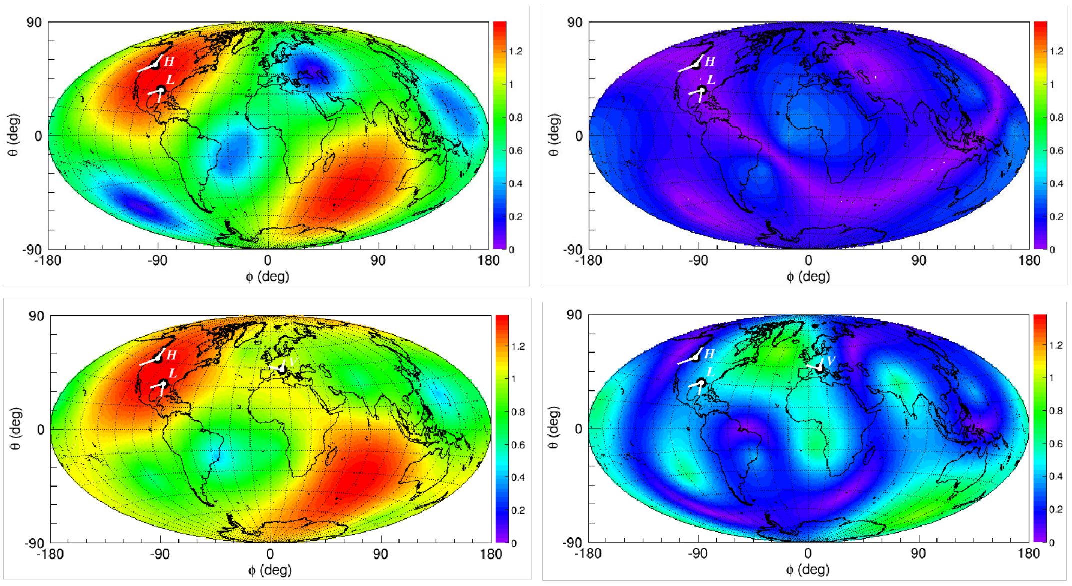

FIGURE 2.20: Antenna pattern $|F_+|$ (left) and $|F_\times|$ (right) in spherical Earth-based coordinates for two networks of detectors, HL (top) and HLV (bottom). When adding Virgo to the network formed by the two LIGO detectors, we assume all detectors share the same spectral sensitivity. The location and orientation of the LIGO and Virgo detectors are depicted in white. Source: [53].

2.2.7 Transient noise artefacts

Despite the attempts to model and mitigate the noise sources in the current GW detectors, the latter are still susceptible to instrumental and environmental disturbances that contaminate the strain data. Of particular concern are non-Gaussian transient noise artifacts, known as glitches, that occur at a very high rate ($\sim$ 1 per min in O2 [36]). Glitches usually last less than a second and can mimic high-mass binary black hole mergers as seen in Fig. 2.21. Moreover, they can also hinder the detection of longer signals such as in the case of GW170817 [26]. Although efforts have been carried out to mitigate the effect of glitches on GW searches [36], the sensitivity of burst searches is still limited by the presence of these noise transients [54, 55]. A thorough analysis of the methods employed in burst searches and their limitations is discussed in Chapter 3.

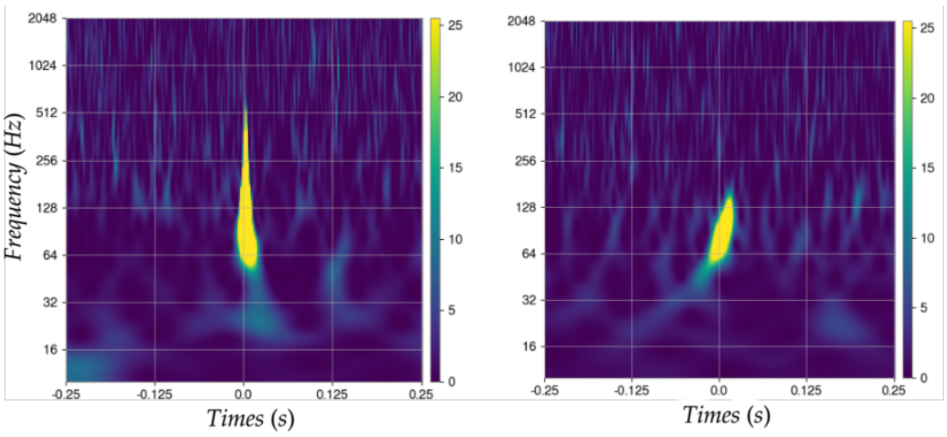

FIGURE 2.21: Time-frequency representations of a blip glitch (left) and a high-mass BBH event (right) produced via the Q-transform method. Source: [56].

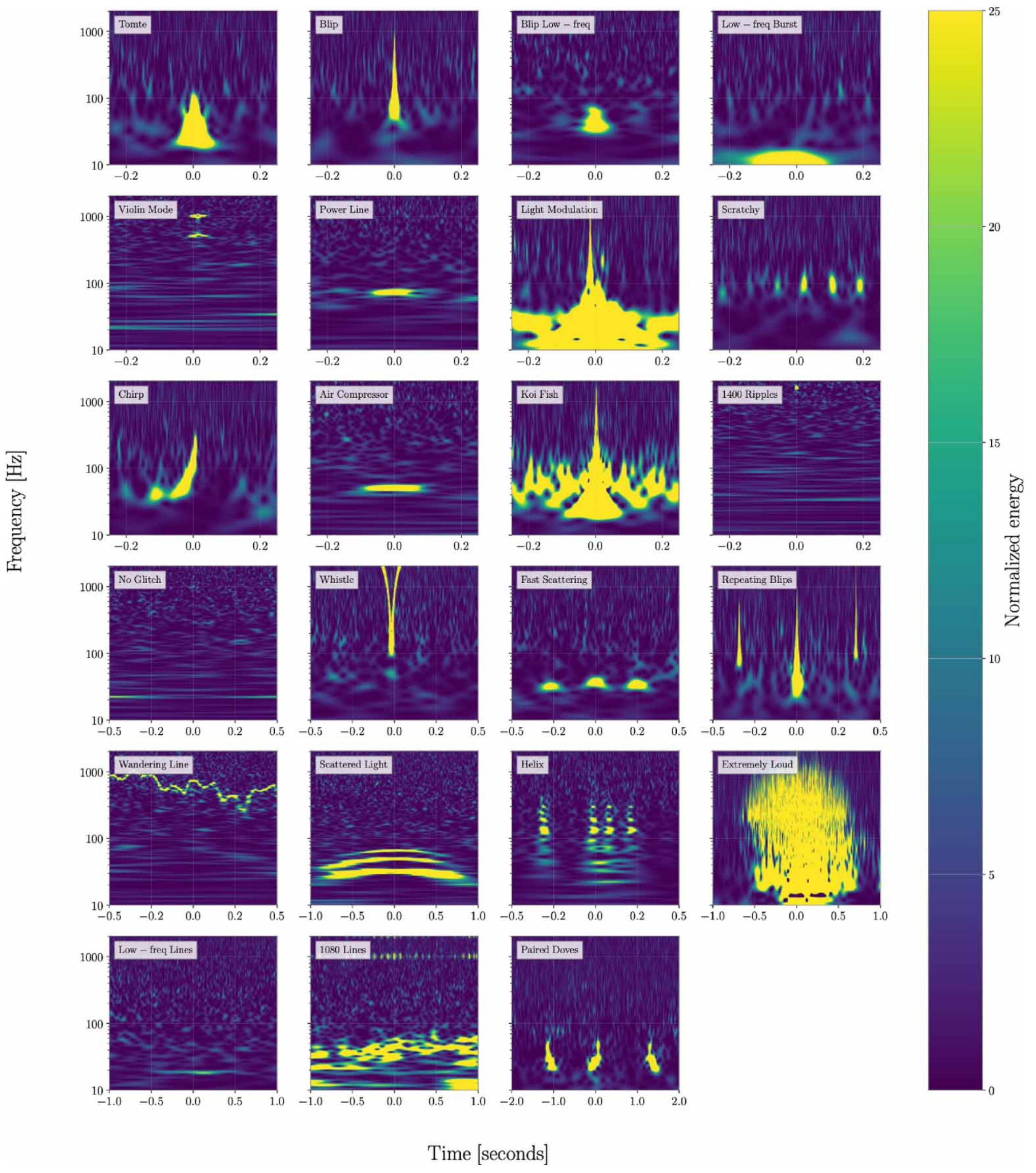

FIGURE 2.22: Time-frequency representations of the 23 glitch classes identified over the first 3 observing runs. The classes are grouped by the time window (0.5 s, 1 s, 2 s or 4 s) that best illustrates their features. Source: [57].

Glitches come in a wide variety of time-frequency morphologies, with new classes appearing as the sensitivity of the detector improves. Characterizing and identifying the cause of all the different classes seems to be a tedious and daunting task. A promising option is to build machine-learning techniques that would rapidly recognize them in the data streams. To this end, *Zevin et al.* [58] have developed a pioneer work via a crowd-sourcing initiative called *Gravity Spy*. Their method consists in using the classification labels assigned by volunteers in order to train a neural network to distinguish the existing glitch classes. Gravity Spy initially identified 20 glitch classes selected by detector characterization experts [58]. The training set was later refined to 23 classes [57], including a "No Glitch" class and a "Chirp" family. The former was added when no significant power is visible in the time-frequency representations. The "Chirp" class contains the time-frequency footprints of binary black hole mergers that have been detected so far [25]. An exhaustive list of the classes used is shown in Fig. 2.22.

With the goal of improving the classification and mitigation of glitches, Melissa Lopez and I have developed a deep-learning algorithm that generates glitch time series [56]. The method consists in training a neural network on glitch time-series reconstructions [59] to learn their intrinsic properties. Once the training is completed, the network is able to generate new unseen glitches sharing the properties of the real glitches. An open-source Python package implementing this algorithm has been developed [60]. The work is preliminary and only considers glitches from the "Blip" class. An example of the generation capability of our network called *gengli*, can be seen in Fig. 2.23. Note the resemblance between the generated glitch and the real blip glitch in the second-top left subfigure of Fig. 2.22.

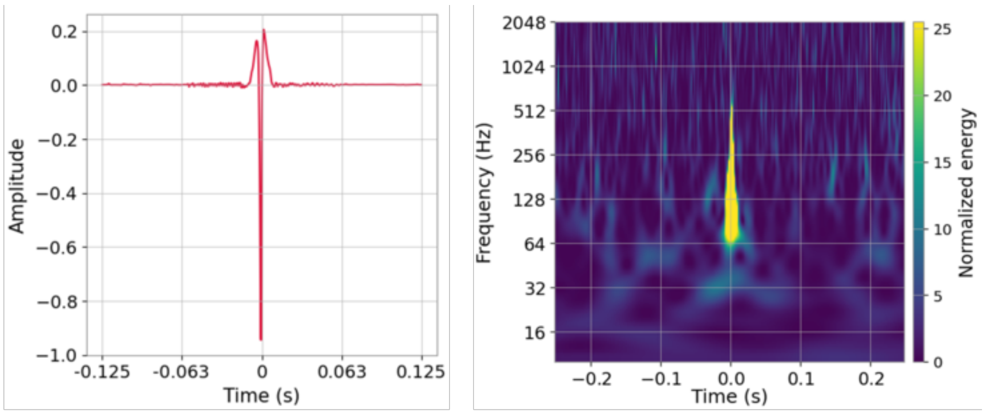

FIGURE 2.23: Time series (left) and time-frequency (right) representations of a glitch generated with gengli. The fake glitch has been injected into O2 real noise.

Chapter 3

Burst searches

3.1 General methods

3.1.1 Prerequisites and notations

Before describing the search methods for GW burst events, let us introduce the notation adapted from [3]. Consider a gravitational wave passing through an interferometric GW detector. The signal in the output port, after the calibration process, is a strain $s(t)$ given by[1]:

$$s(t) = n(t) + h(t) \tag{3.1}$$

where $n(t)$ is the noise and $h(t)$ is the strain of the gravitational wave. In the general case of an elliptically polarized wave, $h(t)$ is the sum of both polarizations h_+ and $h_\times$. However, the interferometer response is different for these two polarizations, depending on the orientation of the arms. Noting F_+ and $F_\times$ the antenna patterns shown respectively in the left and middle panels of Fig. 2.18, the strain response of the detector is expressed as:

$$h(t) = F_+ h_+(t) + F_\times h_\times(t) \tag{3.2}$$

We further make the assumption that the noise in the detector is not correlated with the signal. Mathematically, this requires imposing the correlation of $n(t)$ and $h(t)$ to be zero when averaged over several noise realizations [61]. The correlation between two signals is defined as:

$$R_{xy} = R(x(t), y(t)) = x(t) \otimes y(t) = \int_{-\infty}^{+\infty} x(\tau) y(t+\tau) d\tau \tag{3.3}$$

where τ is called displacement or lag. The condition for which $n(t)$ and $h(t)$ are uncorrelated therefore writes as:

$$\langle n(t) \otimes h(t) \rangle = 0 \tag{3.4}$$

where $\langle . \rangle$ denotes the average over multiple noise realizations.

It is important to note that the time domain representation of the GW signals is not the only representation. Expressions (3.1) and (3.2) can be transformed into the frequency domain. The Fourier transform is useful to go from the time domain to the frequency domain according to:

$$\tilde{x}(f) = \int_{-\infty}^{+\infty} x(t)\, e^{-i2\pi f t}\, dt \tag{3.5}$$

Note that this integral is written in the case of continuous signals. In practice, the strain $s(t)$ is evaluated thousands of times per second and we only know its value at discrete

[1]Note that we make the trivial assumption that the noise in the detector and the GW signal add linearly, which suggests that the detector responds linearly to GW signals ($h^2 << h$).

times. The discrete version of the Fourier transform is expressed as:

$$\tilde{x}_k = \sum_{j=0}^{N-1} x_j \, e^{-i2\pi jk/N} \tag{3.6}$$

where j is an integer number and x_j refers to $x(t_j)$ with $t_j = t_0 + j\,\Delta t$, Δt being the time interval between successive measurements with $\Delta t = 1/f_s$, with f_s the sampling frequency.

The correlation of two signals in the time domain can also be represented in the frequency domain. For this, we apply the Fourier transform to expression (3.3), resulting in the cross-spectral density (CSD) of the signal $x(t)$ with respect to $y(t)$:

$$CSD_{xy} = CSD(x(t), y(t)) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x(\tau) y(\tau + t) d\tau \, e^{-i2\pi ft} \, dt \tag{3.7}$$

A particular case of expression (3.7) is found when evaluating the correlation of a signal with itself, known as auto-correlation. The Fourier transform of the auto-correlation is known as the power spectral density (PSD), noted S and expressed as [61]:

$$S_x = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x(\tau) x(\tau + t) d\tau \, e^{-i2\pi ft} \, dt = \tilde{x}(f)\tilde{x}^*(f) = |\tilde{x}(f)|^2 \tag{3.8}$$

according to the Wiener-Khinchin theorem and $\tilde{x}^*(f) = \tilde{x}(f)$ for real signals. For practical purposes, the PSD measures the contribution to the total power of x made by each frequency component. Its square root, known as the amplitude spectral density (ASD), quantifies the root-mean-squared value of the signal at frequency f.

The definition of the PSD is remarkably helpful in order to characterize the noise $n(t)$. The PSD and the ASD of the noise are widely used to evaluate the impact of the different sources of noise on the output of the detector, as seen in Figures 2.10 and 2.13.

3.1.2 Maximum-likelihood analysis

The maximum-likelihood analysis consists in defining a rule that allows to discriminate between two hypotheses: the data contains only noise and the data contains noise plus a signal. The former hypothesis is noted H_0 and referred to as the null hypothesis. The alternative hypothesis is indicated as H_1.

In practice, the data are sampled at discrete times and $s(t)$ can be considered as a vector, noted $\boldsymbol{s}$. The antenna pattern functions F_+ and $F_\times$ then write as matrices that project each polarization in the frame of the detector. In matrix notation, expression (3.1) can therefore be written as:

$$\begin{aligned} \boldsymbol{s} &= \boldsymbol{n} + \boldsymbol{F}_+ \boldsymbol{h}_+ + \boldsymbol{F}_\times \boldsymbol{h}_\times \\ &= \boldsymbol{n} + \boldsymbol{F}\boldsymbol{h} \end{aligned} \tag{3.9}$$

where the contribution of both antenna patterns have been grouped into a single matrix $\boldsymbol{F}$.

Under each hypothesis, the data $\boldsymbol{s}$ can be viewed as the realization of a stochastic process [61] respectively associated to a probability $p(\boldsymbol{s}|H_0)$ and $p(\boldsymbol{s}|H_1)$. To distinguish between hypotheses, one can define the log-likelihood ratio [62]:

$$\mathcal{L} = 2 \log \left(\frac{p(\boldsymbol{s}|H_1)}{p(\boldsymbol{s}|H_0)} \right) \tag{3.10}$$

With the above definition, if $\mathcal{L} >> 1$, this means that the alternative hypothesis has a high probability of being true and the data cannot be explained by the noise only. They should therefore contain a signal. One can show that thresholding the log-likelihood ratio is the optimal strategy for detecting h, known as the Neyman-Pearson lemma [63].

To expand upon, let us assume that the noise n follows a Gaussian statistics. In such a case, the probability of measuring s in the absence of GW ($h = 0$) is given by:

$$p(\boldsymbol{s}|H_0) = \exp\Big(-\frac{1}{2}\boldsymbol{s}^T\boldsymbol{s}\Big) \tag{3.11}$$

while the probability of measuring s given a GW h is:

$$p(\boldsymbol{s}|H_1) = \exp\Big(-\frac{1}{2}(\boldsymbol{s}-\boldsymbol{Fh})^T(\boldsymbol{s}-\boldsymbol{Fh})\Big) \tag{3.12}$$

The log-likelihood ratio becomes [64]:

$$\mathcal{L} = \boldsymbol{s}^T\boldsymbol{Fh} + (\boldsymbol{Fh})^T\boldsymbol{s} \quad - \quad (\boldsymbol{Fh})^T(\boldsymbol{Fh}) \tag{3.13}$$

The first two terms in expression (3.13) depend on the data $\boldsymbol{s}$ and the signal $\boldsymbol{h}$. The last term is generally ignored since it does not depend on $\boldsymbol{s}$. Typical search methods consist in spanning many different signals $\boldsymbol{h}$ to maximize the likelihood for each detector. The collection of signals for which we evaluate expression (3.13) is called a template bank. Because this method matches the data with the appropriate templates, it is termed matched filtering [65]. In practice, the noise in the detectors is not Gaussian and the detection threshold is not applied to the likelihood ratio but rather to the signal-to-noise ratio (SNR), defined as [3]:

$$\rho^2 = 4\int_0^{+\infty}\frac{|\tilde{h}(f)|^2}{S_n(f)}df \tag{3.14}$$

where $\tilde{h}(f)$ is the Fourier transform of the template h and $S_n(f)$ is the PSD of the noise.

However, this method requires prior knowledge of the expected signals as well as an accurate way to model them. Since bursts are produced in highly turbulent and/or complex explosion mechanisms, simulations can take months to complete only one template on the most recent supercomputers. That template can be considered as a best guess, which might not reflect the complexity of the real GW emission event. We are therefore left with maximizing the likelihood, which is satisfied when:

$$\frac{\partial\mathcal{L}}{\partial h}\Big|_{h=\hat{h}} = 0 \tag{3.15}$$

with $\hat{h}$ being the best-fit waveform. The problem is linear and we can solve it by deriving expression (3.13) with respect to h, leading to:

$$\hat{h} = (\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T\boldsymbol{s} \tag{3.16}$$

In practice, $\boldsymbol{F}^T\boldsymbol{F}$ tends to be a singular matrix (i.e the determinant is close to zero) and the problem does not converge well [64]. An efficient technique to overcome this limitation has been proposed by *Klimenko et al.* [66]. It consists in projecting the antenna pattern of each detector $\boldsymbol{F}$ over a Dominant Polarization Frame (DPF) [67]. This new frame allows to remove the convergence problem that arises in expression (3.16) and to solve for each polarization h_+ and $h_\times$.

3.1.3 Time-Frequency analysis

Let us consider a signal $h(t)$ uncorrelated with the noise $n(t)$. We can evaluate the energy in the data s by computing its auto-correlation, averaged over several noise realizations:

$$\langle s(t) \otimes s(t) \rangle = \langle n(t) \otimes n(t) \rangle + 2\langle n(t) \otimes h(t) \rangle + \langle h(t) \otimes h(t) \rangle \tag{3.17}$$

Since the signal and the noise are uncorrelated, expression (3.4) is valid and the second term equals zero. As the energy carried by the signal is positive, $\langle h(t) \otimes h(t) \rangle > 0$, we can write:

$$\langle s(t) \otimes s(t) \rangle = \langle n(t) \otimes n(t) \rangle + \langle h(t) \otimes h(t) \rangle \geq \langle n(t) \otimes n(t) \rangle \tag{3.18}$$

i.e. the signal induces an excess of power in the data [68].

The above result has led researchers to consider time-frequency representations of the data. Assuming a signal that is well localized in a compact time-frequency box, it will cause an excess of power regardless of the form of the signal. This conclusion allows to develop robust and model-agnostic techniques based on image processing tools. Numerous time-frequency representations have been proposed to detect bursts: Wilson-Daubechies-Meyer transform [66, 69], Welch periodogram [70] and spherical harmonics [71].

It is important to mention that transient artifacts happening in the detector noise $n(t)$, i.e. glitches, also generate an excess of power in the data. Developing techniques to reject them is therefore fundamental to perform searches with low false-alarm rates. Among the proposed methods, we can cite the null-energy [72] or simply the use of Gravity Spy [58] in pre- or post-processing analysis.

As the above-mentioned methods aim at showing the excess of power caused by a signal in the data, their time-frequency representations are all different measures of the power in the data. However, if we decide to measure the absolute variation in power, we will be limited by the power contained in the noise. Indeed, the noise has a particular amplitude spectral density, mainly limited by quantum noise, thermal effects and displacement noise and contaminated by instrumental lines. The PSD, being the square of the ASD (see Fig. 2.13), reveals that most of the power is contained at low frequency or in the instrumental lines. It is therefore necessary to measure the relative excess of power to mitigate this effect. To this aim, *Cuoco et al.* [73] have developed a technique, called whitening, that consists in dividing $\tilde{h}(f)$ by the ASD of the noise $\sqrt{S_n(f)}$ at each frequency component. This procedure transforms the noise into a white noise, i.e. showing uniform distribution of power at all frequencies[2], resulting in a flat PSD. Note that in the expression of the SNR (3.14), the whitening is done implicitly each time we match one template with the noise. This is the consequence that most of our theory of detection is established in the framework of a stationary Gaussian white noise [73]. If we whiten the data, we approach the ideal case in which most of the detection algorithms have been developed.

Fig. 3.1 shows an example of a spectrogram generated on raw data as opposed to a spectrogram evaluated on the same data, but whitened. In the left panel, we can clearly identify the lines corresponding to the violin modes of the silica fibers as well as a peak of noise at low frequencies (< 20 Hz). It is also impossible to depict the injected signal. When the data is whitened, the excess power due to the signal is easily identified. We also recognize 4 broadband glitches that appear as vertical lines because of their short duration.

[2]In opposition, when a signal does not show the same power over all frequencies, it is said to be colored.

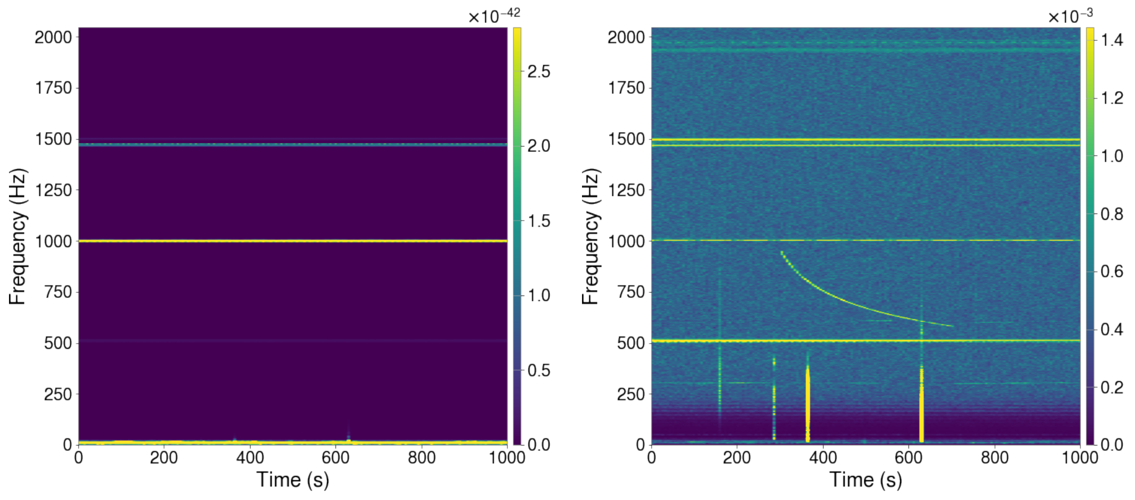

FIGURE 3.1: Spectrograms generated on H1 data from O3a. The data are either non-whitened (left) or whitened (right). In the non-whitened case, most of the power lies below 20 Hz and in the instrumental lines. When whitened, the excess of power due to the signal and the glitches is easily recognized. The frequency resolution is 2 Hz with time bins of 6 s. The GPS time at the start of the data is 1248306006 and the injected signal is a magnetar model [74].

3.1.4 Coincident searches

Even if statistics and methods have been defined to address the problem of glitches [58, 72], causing parasite excess of power in the data, the overwhelming amount of these transient artifacts still limits the sensitivity of burst searches. A way to solve this problem is to use multiple detectors in coincidence. The core concept is, if two or more detectors are far apart, their noise should be mostly uncorrelated[3]. The probability of an accidental coincidence is therefore very small while a GW signal should shake both detectors nearly at the same time. Note that the methods addressed in subsections 3.1.2 and 3.1.3 can be easily adapted to coincident searches by taking into account the time-of-flight between detectors $\Delta t = \boldsymbol{\Delta r} \cdot \boldsymbol{\Omega}/c$, where $\boldsymbol{r}$ is the position of the detector with respect to the center of the Earth, $\boldsymbol{\Delta r}$ the distance between them and $\boldsymbol{\Omega}$ is the sky localization of the GW signal.

When performing coincidences between detector data, it is important to orient the detectors according to their location. To confirm a signal, both detectors should be sensitive to roughly the same sky directions. As the location of these detectors affects their sky sensitivity, it is convenient to orient the arms to compensate for this difference in location [3]. That is why both LIGO detectors, H1 and L1, show approximately the same sky maps (see Fig. 2.20).

There exist two methods to carry coincident searches: single-detector or multi-detector methods. The former consists in analyzing separately the data streams from the detectors at first and searching for coincidences in a second time. In the time-frequency domain, we can build two separate TF maps, process them and require some consistency in the detected signals. For instance, we could require that the bandwidth of the two events have an overlap [3]. The multi-detector method rather combines the data from each detector

[3]To the exception of electromagnetic disturbances, that can propagate all over the globe.

to evaluate one or several statistics that will be used to search for GW signals. As an example of statistics, we can cite the coherence of two data streams from different detectors. Considering two signals $x(t)$ and $y(t)$, the coherence is evaluated via:

$$C_{xy}(f) = \frac{|CSD_{xy}(f)|^2}{S_x(f)\,S_y(f)} \tag{3.19}$$

where the cross-spectral density and the PSD of both signals are involved. In practice, expression (3.19) is evaluated at several sampled frequencies to produce a vector of coherence values versus frequency bins. To generate a full time-frequency array, we apply Welch's method [75] to successive time intervals of the original data streams. This is equivalent to repeatedly updating the coherence value and compiling this time evolution as a single map, known as a coherence spectrogram.

Any algorithm that aims at detecting GW signals needs to assess the number of false detections over a certain period of time, known as the false-alarm rate. These false alarms are most of the time due to random noise realizations or glitches contaminating the data. In order to assess the sensitivity of the pipeline to these noise artifacts, it is convenient to process a large quantity of data that does not contain GW signals, referred to as background. To estimate the background, and therefore make sure that there is no GW signal, we can shift the data streams that are analyzed. The procedure consists in shifting the data of one detector with respect to the other by a time step significantly longer than the time-of-flight [76]. The events captured by the detection algorithms are then all accidental coincidences arising from different noise realizations. Repeating this process with multiple time steps allows to simulate hundreds of years of coincident data with only months of data acquired by the GW detectors.

3.1.5 Sky localization

Typically we do not know the incident sky direction $\mathbf{\Omega}$ of a signal. However, the likelihood ratio depends on $\mathbf{\Omega}$, via the antenna patterns $\boldsymbol{F}_+$ and $\boldsymbol{F}_\times$, and via the arrival time delay between detectors Δt. The standard approach consists in repeating the maximum likelihood analysis over a grid of $\mathbf{\Omega}$ covering the entire sky [64, 66]. This procedure gives likelihood maps as seen in Fig. 3.2. The likelihood can be well localized in a unique small

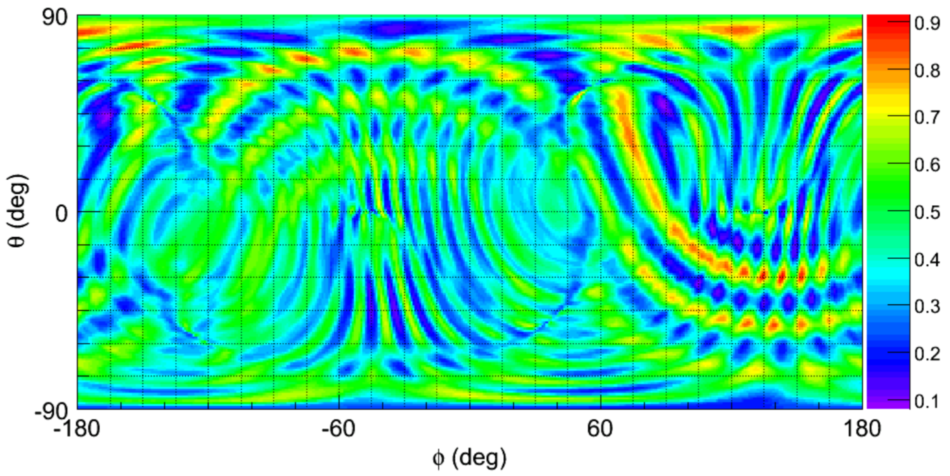

FIGURE 3.2: Likelihood sky map produced by coherent WaveBurst [66] for a Sine-Gaussian signal injected at $\mathbf{\Omega} = (\theta, \phi) = (-30^\circ, 144^\circ)$. Source: [77].

area or distributed over a large zone, itself split into several disjoint clusters. This ambiguity can be caused by several factors such as the signal strength or its morphology and is typical for networks with only three detectors [77]. Adding KAGRA and LIGO India to the current network of detectors will allow a wider sky coverage as well as an improved sky localization [77].

3.2 Long-duration searches

3.2.1 Astrophysical sources

Historically, gravitational wave transients have been classified into two classes: compact binary coalescences (CBC), whose models can be predicted, and bursts for which exact waveforms are inaccessible yet. The methods used to search for burst events and their associated challenges greatly depend on their durations. That is why bursts have been further decomposed into two types, short ($\leq$ 1 sec) and long ($>$ 1 sec). As this thesis focuses on the detection of long transients, only the potential progenitors of such signals will be described below. As a reminder, although some waveforms have been modeled, they mainly serve as test signals to compare algorithms and pipelines due to their known inaccuracies.

At the end of their lives, massive stars generally blow up into supernovae, giving birth to either a black hole or a neutron star. The remnant object could, under some conditions, be subject to fallback accretion of the matter originally ejected by the massive star. In the case of a neutron star, the fallback of matter can cause its collapse into a black hole, possibly producing gamma-ray bursts. When the incoming material forms a disk, the accretion can spin up the neutron star sufficiently to produce non-axisymmetric instabilities, leading to gravitational wave radiation [78]. When a black hole is the remnant of the massive star, the accretion of fallback material down to the Inner most Stable Circular Orbit (ISCO) can generate and sustain broadband gravitational wave emissions [79]. The subsequent models of GW emission are called *ISCOchirp* in this work.

Long duration bursts could also be produced in a torus of matter around a black hole. Catastrophic events such as hypernovae or black hole-neutron star coalescences are expected to result in a black hole surrounded by a disk or a torus of matter. Gravitational wave radiation could then originate from instabilities in the accretion disk or fragmentation of matter spiraling into the black hole [80, 81]. The waveforms referring to this type of event are denoted by the acronym *ADI* (adiabatic disk instabilities).

Supernovae and even binary neutron star mergers can lead to the formation of a highly magnetized rapidly rotating neutron star, known as magnetar. This newborn object could undergo secular instabilities, leading to gravitational wave losses which in turn would affect its spin-down [82]. The latter process has been proposed to account for plateaus observed in the light curves of gamma-ray bursts (GRB), and we mention the related waveform models as *GRBplateau* or *CM09*. Other mechanisms like distortions of the magnetar shape caused by very high internal magnetic fields ($> 10^{15}$ Gauss) are considered to be GW sources [74]. In such scenarios, the models are commonly denoted as *Magnetar* or *maXgnetar*.

Fig 3.3 shows some of the models used to test the methods developed in this work. The acronyms refer to the above-mentioned progenitor events with the exception of *ECBC* that refers to eccentric compact binary coalescences. Some of the CBC waveforms [83] are

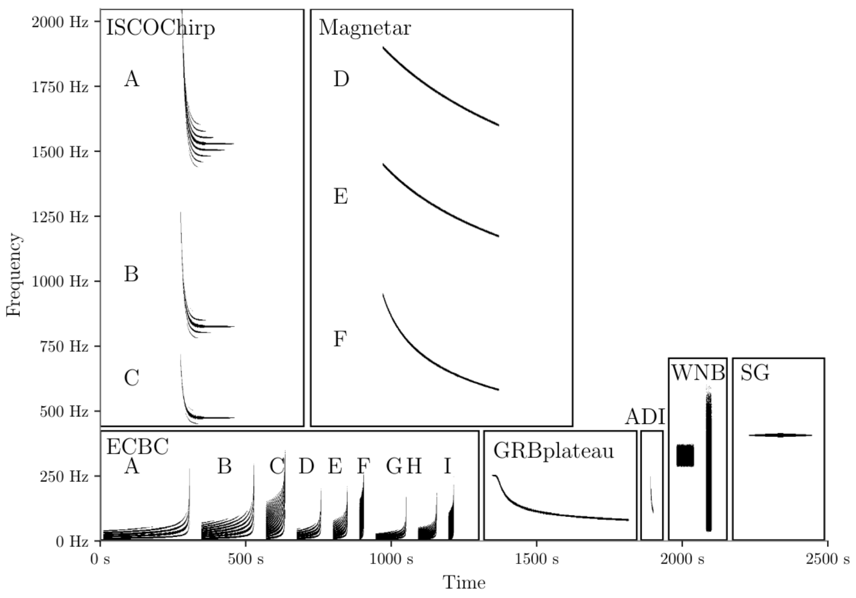

FIGURE 3.3: Time-frequency representation of some long-duration burst models. Note that the *ISCOchirp* waveforms have been shifted up by 50 Hz for readability. The different capital letters (from A to I) assigned to a waveform model refer to the same astrophysical phenomenon simulated with different parameters and/or different hypotheses. Source: [55].

indeed considered targets for burst searches. Note that *WNB* and *SG* stand respectively for White Noise Bursts and Sine-Gaussian. Even if they are not astrophysical signals, their time-frequency behavior serves as valuable tests for detection algorithms [55].

3.2.2 Expected amplitude and range

Now that we have an idea of the physical processes that can produce GW bursts, it is convenient to evaluate the distance to which LIGO and Virgo could detect them. For this let us express the SNR of the GW signal at the detectors as a function of the energy radiated by the source. By estimating the amount of energy carried away from different sources, we can derive the distance that would lead to a particular SNR in the detector data. First we write expression (3.14) as a function of the two polarizations h_+ and $h_\times$:

$$\rho^2 = 4 \int_0^{+\infty} \frac{|F_+\tilde{h}_+(f) + F_\times \tilde{h}_\times(f)|^2}{S_n(f)}\, df \tag{3.20}$$

In burst searches, the standard measure of the intensity of a signal is not the SNR, but rather the root-sum squared amplitude:

$$\begin{aligned} h_{rss}^2 &= \int_{-\infty}^{+\infty} \left(h_+^2(t) + h_\times^2(t)\right) dt \\ &= 2\int_0^{+\infty} \left(|\tilde{h}_+(f)|^2 + |\tilde{h}_\times(f)|^2\right) df \end{aligned} \tag{3.21}$$

where we have used Parseval's theorem and integrated over the physical positive frequencies. Note that the h_{rss} does not take into account the detector noise as in the definition of the SNR. Since the noise in the current GW detectors varies with time, two identical signals (i.e. with the same h_{rss}) arriving at different times could lead to different footprints in the time-frequency maps. This concern will be addressed in Chapter 5.

We can estimate the signal-to-noise ratio by assuming a narrowband signal with central frequency f_0. The power spectrum of the noise $S_n(f)$ can then be considered constant and equal to $S_n(f_0)$. In the case of an elliptically polarized wave, the two polarizations are independent stochastic time series and the terms $\tilde{h}_+\tilde{h}_\times^*$ can be dropped[4] [84]. The SNR can therefore be written as:

$$\begin{aligned}\rho^2 &= \frac{4}{S_n(f_0)}\int_0^{+\infty}\left(F_+^2\,|\tilde{h}_+(f)|^2 + F_\times^2\,|\tilde{h}_\times(f)|^2\right)df \\ &= \frac{\Theta^2}{S_n(f_0)}\int_0^{+\infty}\left(|\tilde{h}_+(f)|^2 + |\tilde{h}_\times(f)|^2\right)df \\ &= \frac{\Theta^2}{S_n(f_0)}\,h_{rss}^2\end{aligned} \tag{3.22}$$

where the dependence on the sky position, source orientation and polarization of the wave is contained in the angle factor Θ [84]. Now we need to relate the total energy emitted in gravitational waves E_{GW} to the h_{rss} amplitude at the detector. This expression can be found in [3]:

$$E_{GW} = \frac{\pi c^3}{2G}\int_{-\infty}^{+\infty} f^2\left[\int\left(|\tilde{h}_+(f)|^2 + |\tilde{h}_\times(f)|^2\right)dA\right]df \tag{3.23}$$

where we integrate over all frequencies of the signal and over the area A through which the total energy is flowing. Let us consider an isotropic emission from a source located at a distance r from the detectors. The two polarizations are therefore independent of the emission direction and the integral over dA is equal to $4\pi r^2$, leading to:

$$\begin{aligned}E_{GW} &= \frac{2\pi^2c^3r^2}{G}\int_{-\infty}^{+\infty} f^2\left(|\tilde{h}_+(f)|^2 + |\tilde{h}_\times(f)|^2\right)df \\ &\simeq \frac{2\pi^2c^3r^2}{G}\,f_0^2\int_{-\infty}^{+\infty}\left(|\tilde{h}_+(f)|^2 + |\tilde{h}_\times(f)|^2\right)df \\ &\simeq \frac{2\pi^2c^3r^2}{G}\,f_0^2\,h_{rss}^2\end{aligned} \tag{3.24}$$

where we have assumed a narrowband signal with central frequency f_0.

We can now express the signal-to-noise ratio as a function of the energy by replacing the expression of the h_{rss} with expression (3.24):

$$\rho^2 \simeq \Theta^2\,\frac{G}{\pi^2c^3}\,\frac{E_{GW}}{S_n(f_0)r^2f_0^2} \tag{3.25}$$

If we average over angles, a good estimate for Θ is $1/\sqrt{2}$ [84]. Rearranging the terms, we obtain the effective distance at which our current GW detectors are sensitive to burst events:

$$\mathcal{R}_{eff} \simeq \left(\frac{G}{2\pi^2c^3}\,\frac{E_{GW}}{S_n(f_0)f_0^2\rho_{det}^2}\right)^{1/2} \tag{3.26}$$

[4]This result is also valid in the case of a circularly or linearly polarized wave, where respectively the two polarizations are orthogonal and $\tilde{h}_\times = 0$ or $\tilde{h}_+ = 0$.

where ρ_{det} is the threshold at which the detection efficiency is 50%. For sine-Gaussian signals, this threshold is expected to be $\rho_{det} \simeq$ 20-30 [85, 86]. As we expect the effective range to be limited by the fainter signals, we take $\rho_{det} = 20$ for our further estimates. Taking the aLIGO and Advanced Virgo design sensitivity curves, it is possible to illustrate R_{eff} as a function of the frequency. Fig. 3.4 shows the effective range of the two GW detectors in two different cases, $E_{GW} = 10^{-6}\, M_{\odot}c^2$ or $E_{GW} = 10^{-2}\, M_{\odot}c^2$.

Fig. 3.4 can be used to determine the effective range for some of the astrophysical events described in subsection 3.2.1. In the case of magnetar giant flares, the amount of energy radiated into GWs is not expected to exceed $10^{-6}\, M_{\odot}c^2$ [87]. This leads to a range roughly equal to 100 kpc in the most sensitive band of the LIGO detector. The width of the Milky Way being 32 kpc, only galactic sources can be probed with current generation detectors. Instabilities in the accretion disk around black holes (as in *ADI*) are expected to emit GWs with $E_{GW} = 10^{-3} - 10^{-2}\, M_{\odot}c^2$, leading to an effective range of 10 to 100 Mpc in the most optimistic scenarios [88]. Such events can thus be detected as far as the Virgo cluster and even beyond. Secular instabilities in highly magnetized neutron stars (as modeled in *GRBplateau*) should produce GW energy of the order $E_{GW} = 10^{-2}\, M_{\odot}c^2$, and therefore be observable up to 100 Mpc [82]. Fallbacks of material down to the ISCO may induce turbulences and spin down of the black hole, leading to GWs with energy up to $10^{-1}\, M_{\odot}c^2$. This strong emission scenario shows an effective range of several Gpc [81].

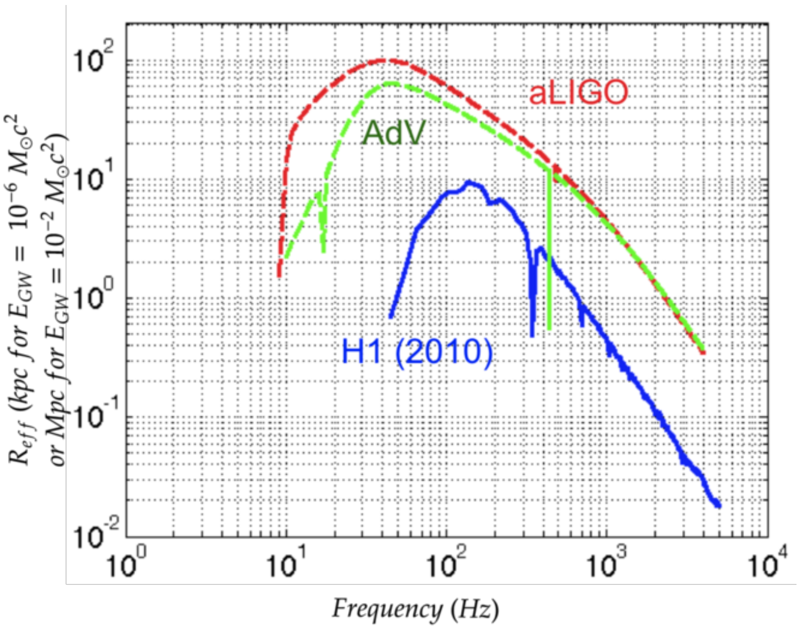

FIGURE 3.4: Effective range as a function of the frequency for the Advanced LIGO and Advanced Virgo detectors. The design sensitivity curves have been used to evaluate $S_n(f)$ and $\rho_{det} > 20$. For comparison, the effective range of the Hanford detector in 2010 is shown. Source: [89].

3.2.3 Detection pipelines

The search for long duration bursts is tackled by several groups within the LVK collaboration. The use of multiple pipelines, implementing various TF representations and clustering algorithms, provides redundancy as well as different sensitivities to different signal morphologies. The diversity of detection algorithms is therefore important to cover the whole TF space. The methods used to identify bursts in the TF images can be classified into two categories: seed-based or seedless. Seed-based methods aim at clustering pixels above a predefined threshold while seedless algorithms are processing pixels derived from generic models.

In O2 [90], the pipelines used were the long-duration configuration of coherent WaveBurst (cWB) [66], the two different versions of the Stochastic Transient Analysis Multi-detector Pipeline - All Sky (STAMP-AS), Zebragard and Lonetrack [91, 92], and X-SphRad [71]. In O3, X-SphRad has been dropped and only the 3 first pipelines performed the analysis of the LIGO-Virgo data [55]. For O4, PySTAMPAS [70] should incorporate and replace the two configurations of STAMP-AS. cWB and PySTAMPAS are both seed-based algorithms.

cWB [66] is based on a maximum likelihood approach. A time-frequency representation of the strain data is obtained via the Wilson-Daubechies-Meyer transform [69]. Then, the whitened TF series of all detectors are combined to build multi-resolution energy maps, which are used to identify clusters of excess power. The selected clusters constitute the burst events, from which cWB extracts the signal waveform, polarization and sky location by solving the inverse problem in expression (3.16). The search is performed on data where poor quality periods have been removed, in the frequency range of 24 - 2048 Hz. A threshold is applied on the null-energy criterion to reject glitches [72]. The triggers that survive this threshold, are then ranked according to their detection statistic.

PySTAMPAS [70] processes single-detector spectrograms built on the time series strain data. The TF maps have a duration of 500 s and a frequency band of 24 – 2048 Hz with a set of 4 resolutions ranging from 4 s × 0.25 Hz to 0.5 s × 2 Hz. A clustering algorithm identifies clusters of excess energy pixels in all individual spectrograms. The pixels from clusters obtained in a single spectrogram are then matched with the corresponding pixels in the other spectrograms to produce a coherent statistic. This coherent analysis is evaluated on multiple sky positions, which enables a precise reconstruction of the signal. PySTAMPAS also implement a glitch rejection procedure, similar to the null-energy criterion. Finally, post-processing steps such as vetoes and gating are used to deal with non-Gaussian events.

Unfortunately, long duration bursts have not been detected over the first three observing runs. With the planned improved sensitivity of the aLIGO and Advanced Virgo detectors, the probability of an event entering the effective range of the detectors is higher. Burst pipelines should therefore be ready to analyze the incoming data during O4. However, a single 1000-second block of data can take several minutes to be processed, which ultimately leads to delayed detection. This latency might affect the discovery of a potential electromagnetic counterpart. Moreover, the thresholds used to select the excess power pixels are affected by the noise in the detectors. Consequently, they might need to be modified from one observing run to the following. The same remark holds for the parameters used in the post-processing steps. In the end, a detection pipeline might need several weeks to months to be fine-tuned. This thesis aims at tackling both problems by developing a parameter-free tool that provides fast trigger identification, taking advantage of the speed of neural networks.

As building a full detection pipeline takes several years, we will not develop everything from scratch in this thesis. Indeed, this work originates from the early development of a new long-duration pipeline, called Pyxel (draft in progress), which has been built by Maxime Fays. Pyxel is a seed-based algorithm that processes coherence spectrograms. More precisely, after whitening the data from both detectors, it evaluates their coherence from which a TF map is built. The whitening procedure being imperfect, violin lines appear in the raw coherence spectrogram as in the right panel of Fig. 3.1. A normalization across the frequency bins is then performed to discard these horizontal lines. Pyxel finally applies a clustering method on the pixels that pass a certain threshold. The goal of this thesis is to replace the detection engine included in Pyxel while preserving the structure of the pipeline.

Chapter 4

Deep learning

4.1 Introduction to deep learning

4.1.1 General considerations

Deep learning, machine learning and artificial intelligence (AI) are terms often used interchangeably to characterize algorithms that are capable of mimicking human behavior. It is important to understand the key distinctions among them to use the appropriate term.

Deep learning is a subset of machine learning, which in turn is a subset of AI. More generally, AI refers to all the computer systems that are able to perform tasks that normally require human intelligence such as decision-making, visual perception or speech recognition. A pile of "if-else" statements, carrying out some computations if some conditions are fulfilled, is then considered an AI algorithm. The fundamental aspect that distinguishes machine learning from other AI programs, is the ability to modify itself when exposed to data. A machine learning algorithm is therefore dynamic and can adapt its inner parameters without being explicitly programmed to do so. The phase in which the machine learning model optimizes its parameters to achieve an objective is called the learning phase. The key differences between machine learning and deep learning are the methods used to complete the learning phase. Machine learning involves statistical models while deep learning is based on Artificial Neural Networks (ANNs). ANNs are a set of interconnected nodes that process and learn from data in a way that is inspired by the human brain. Neural networks usually pass input data through much more mathematical operations than other machine learning algorithms and are therefore more computationally intensive to train. Fig. 4.1 summarizes the relationship between AI, machine learning and deep learning.

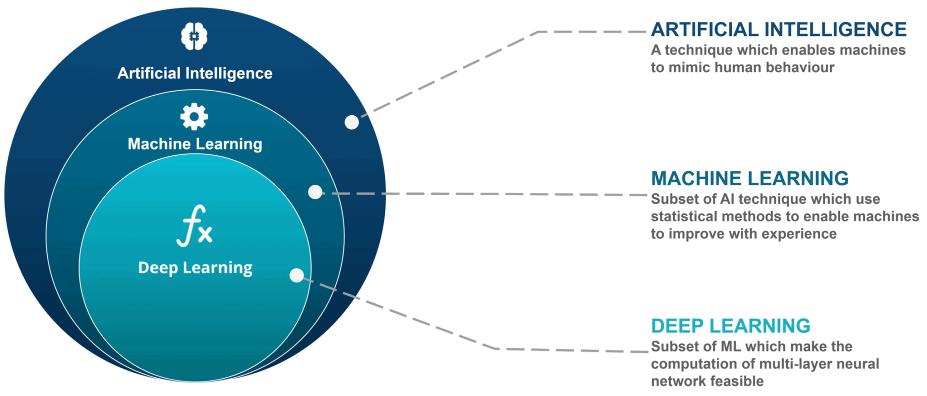
Artificial Intelligence
Machine Learning
f_x
Deep Learning
ARTIFICIAL INTELLIGENCE
A technique which enables machines to mimic human behaviour
MACHINE LEARNING
Subset of AI technique which use statistical methods to enable machines to improve with experience
DEEP LEARNING
Subset of ML which make the computation of multi-layer neural network feasible

FIGURE 4.1: Illustration of the difference between AI, machine learning and deep learning. Source: [93].

Within machine learning, there are two basic types of approaches: supervised learning and unsupervised learning. In the former case, the goal is to predict the outcomes of new data. During the training, the predicted outcomes are compared to the expected outputs, known as labels, and the result is used by the network to learn over time. Classification and regression tasks are examples of problems where labeled data are particularly helpful. In contrast, unsupervised learning models work on their own to discover the inherent structure of data. The purpose is to discover patterns and trends in the data without the help of labeled outputs. These algorithms are generally used in clustering or dimensionality reduction problems, where a-priori knowledge of the data is often missing.

The first neural network has been proposed in 1958 by Frank Rosenblatt [94] and is known as the Multi-Layer Perceptron (MLP). It consists of nodes arranged in multiple layers, and non-linearly activated via an activation function. Each node includes a simple linear model of the form $y = Ax + b$, representing an artificial neuron. The activation functions are essential to introduce the non-linearities needed to learn complex relationships between the input data and the desired outputs[1]. They play the role of the axons in biological neurons, allowing them to transfer or discard the information to other neurons. Fig. 4.2 illustrates the analogy with the model of a neuron. In an MLP, the nodes from one layer are connected to all the nodes in the next layer so that the information can flow through the whole network. Such a layer is denoted as a fully-connected layer.

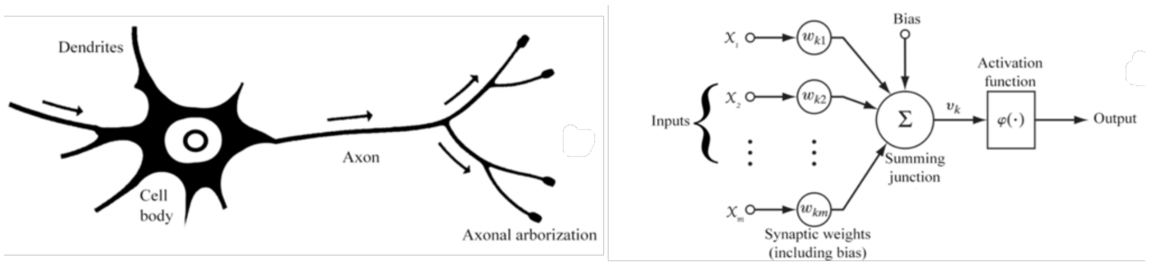

FIGURE 4.2: Similarity between a biological (left) and an artificial (right) neuron. The neuron receives the signals from the other neurons via the dendrites, concatenates the information and propagates it through the axonal connections. The artificial neurons sum the weighted inputs, apply an activation function to the sum and generate an output signal. Adapted from [95] and [96].

All neural networks share a common goal: minimizing a loss function. A loss function is a mathematical expression used to assess how close are the predictions of the network from the expected outputs. By minimizing the value of the loss, the predictions progressively approach the desired outputs and the network consequently learns how to perform its task. This optimization phase is known as the learning or training phase.

4.1.2 How do neural networks learn?

Gradient descent

The learning phase of a neural network is an iterative process in which the network modifies its inner parameters, or weights, to minimize a loss function. The idea is to find the best set of parameters via a technique called gradient descent. Gradient descent is a method to update the weights of a network by evaluating the derivatives of the loss function with respect to these weights. To illustrate how it works, let us consider a loss function $\mathcal{L}$ defined

[1]Linear algebra shows that any composition of linear functions can be reduced to a single linear function.

over some parameters θ, as seen in Fig. 4.3. Let us try to find the minimum of the loss with an iterative procedure, starting from an initial guess θ_0. For a small perturbation ϵ of this starting point, the loss can be written:

$$\hat{\mathcal{L}}(\epsilon;\theta_0) = \mathcal{L}(\theta_0) + \epsilon^T \nabla_\theta \mathcal{L}(\theta_0) + \frac{1}{2\gamma}||\epsilon||^2 \tag{4.1}$$

using a second-order Taylor expansion with γ being a constant. To minimize the loss, we need to solve:

$$\begin{aligned} \nabla_\epsilon \hat{\mathcal{L}}(\epsilon;\theta_0) &= 0 \\ \nabla_\theta \mathcal{L}(\theta_0) + \frac{1}{\gamma}\epsilon &= 0 \end{aligned} \tag{4.2}$$

which happens when

$$\epsilon = -\gamma \nabla_\theta \mathcal{L}(\theta_0). \tag{4.3}$$

The optimal step to update the parameters is therefore proportional to the gradient of the loss. By repeating this procedure, we find a general rule to update the network's parameters:

$$\theta_{t+1} = \theta_t - \gamma \nabla_\theta \mathcal{L}(\theta_t) \tag{4.4}$$

where γ is known as the learning rate.

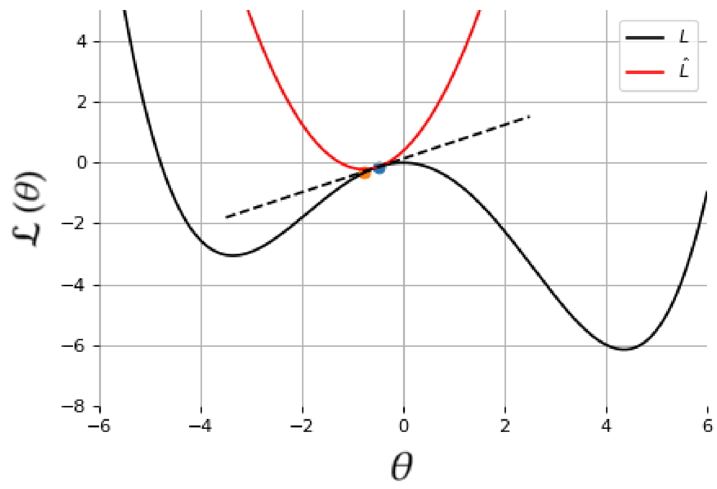

FIGURE 4.3: Illustration of the gradient descent algorithm. The black curve shows the loss function while the red curve represents its approximation via a Taylor expansion. The dotted line indicates the local gradient that is used to update the initial parameters from the blue dot to the orange dot. The graph is adapted from [97].

The iterative procedure described above consists in approximating the loss $\mathcal{L}$ with a second-order Taylor expansion around a local point and then evaluating the local gradient. Finally, the parameters are updated thanks to expression 4.4.

It is important to consider that the procedure of finding the minimum of the loss function is dependent on the data. The gradient $\nabla_\theta \mathcal{L}(\theta_t)$, and therefore the approximation of the loss, will depend on the data x that have been used to evaluate the loss, i.e. $\nabla_\theta \mathcal{L}(x, \theta_t)$. The best approximation could be achieved by evaluating the loss over all the data available at every step of the gradient descent algorithm. This method is known as batch gradient descent. However, since thousands or even millions of samples are often required to train a neural network, this would lead to a memory overload or would take too much time to complete. Instead one can decide to evaluate the loss over only 1 sample at a time. This procedure is known as stochastic gradient descent. Unfortunately, although being computationally cheap, the approximation of the total loss is imprecise and leads to a very long training time to reach the minimum of the loss. As a compromise, a subset of the data, known as a mini-batch, is used to improve the accuracy of the gradient and accelerate the training phase. The size of the mini-batch subset is designated as (mini-)batch size. Fig. 4.4 illustrates the gradient descent algorithm in the 3 different cases mentioned above. In the parameter space, batch gradient descent allows to get the most accurate feedback and converge rapidly towards the minimum. Stochastic gradient descent is inaccurate and takes time to converge while mini-batch gradient descent is a good compromise between speed and accuracy.

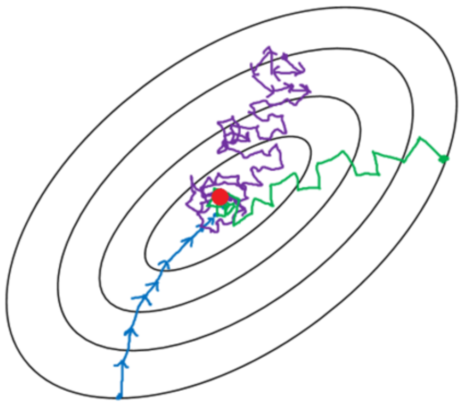

FIGURE 4.4: Illustration of the different gradient descent procedures in the parameter space. Batch (blue), stochastic (purple) and mini-batch (green) gradient descents are shown. Batch gradient descent provides ideal feedback but is impractical. Stochastic gradient descent is computationally cheap but imprecise. Source: [98].

Backpropagation

At this stage, we have a rule to update the parameters of the neural network that is based on the gradient of the loss function, the latter being evaluated over a mini-batch of data. The gradient of a function depending on K parameters is a vector of all the partial derivatives:

$$\nabla \mathcal{L} = \left[\frac{\partial \mathcal{L}}{\partial \theta_0}, \frac{\partial \mathcal{L}}{\partial \theta_1}, ..., \frac{\partial \mathcal{L}}{\partial \theta_{K-1}} \right]. \tag{4.5}$$

Computing the gradient requires to evaluate all the partial derivatives with respect to the network's parameters. Let us consider a very simple network in order to demonstrate how it is done in practice. As shown in Fig. 4.5, we build a multi-layer perceptron with 3 inputs, 2 intermediate (or hidden) layers comprising 2 nodes, and 1 output. The 3 inputs, noted x, are first passed through a linear model $Wx + b$ and then through a non-linear activation function f. The output of the linear model is denoted h while the output of the activation function is designated as a. The intermediate values are fed to a second hidden layer, producing the final output S.

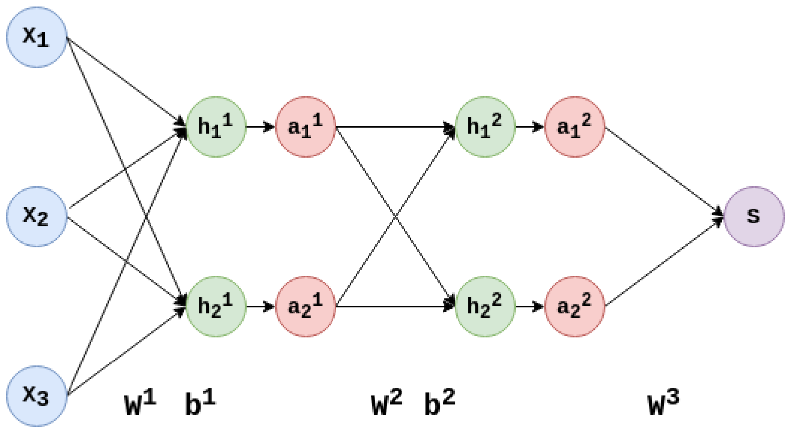

FIGURE 4.5: Example of a multi-layer perceptron. The inputs x are fed to two hidden layers which produce the output S. The outputs of the linear models before and after the activation functions are noted h and a respectively. The weights corresponding to the first and second hidden layers are indicated as $[W^1, b^1]$ and $[W^2, b^2]$ respectively while W^3 stands for the weights of the last layer.

The forward pass of the inputs through the network can be summarized into a set of 5 expressions:

$$h^1 = W^1\,x + b^1 \quad \text{with} \quad h^1 = \begin{bmatrix} h^1_1 \\ h^1_2 \end{bmatrix}, \quad W^1 = \begin{bmatrix} w^1_{11} & w^1_{12} & w^1_{13} \\ w^1_{21} & w^1_{22} & w^1_{23} \end{bmatrix}, \quad b^1 = \begin{bmatrix} b^1_1 \\ b^1_2 \end{bmatrix} \tag{4.6}$$

$$a^1 = f(h^1) \quad \text{with} \quad a^1 = \begin{bmatrix} a^1_1 \\ a^1_2 \end{bmatrix} \tag{4.7}$$

$$h^2 = W^2\,a^1 + b^2 \quad \text{with} \quad h^2 = \begin{bmatrix} h^2_1 \\ h^2_2 \end{bmatrix}, \quad W^2 = \begin{bmatrix} w^2_{11} & w^2_{12} \\ w^2_{21} & w^2_{22} \end{bmatrix}, \quad b^2 = \begin{bmatrix} b^2_1 \\ b^2_2 \end{bmatrix} \tag{4.8}$$

$$a^2 = f(h^2) \quad \text{with} \quad a^2 = \begin{bmatrix} a^2_1 \\ a^2_2 \end{bmatrix} \tag{4.9}$$

$$S = W^3\,a^2 \quad \text{with} \quad W^3 = \begin{bmatrix} w^3_1 & w^3_2 \end{bmatrix} \tag{4.10}$$

where the superscripts refer to the layers of the network. Note that the bias b of the last layer has been set to zero for simplicity.

Let us compute the partial derivative of the loss with respect to w^2_{22}. Using the forward pass equations (4.6) to (4.10) and the chain rule of partial derivatives, it leads to:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w^2_{22}} &= \frac{\partial \mathcal{L}}{\partial h^2_2} \quad \frac{\partial h^2_2}{\partial w^2_{22}} \\ &= \frac{\partial \mathcal{L}}{\partial a^2_2} \quad \frac{\partial a^2_2}{\partial h^2_2} \quad \frac{\partial h^2_2}{\partial w^2_{22}} \\ &= \frac{\partial \mathcal{L}}{\partial S} \quad \frac{\partial S}{\partial a^2_2} \quad \frac{\partial a^2_2}{\partial h^2_2} \quad \frac{\partial h^2_2}{\partial w^2_{22}} \\ &= \frac{\partial \mathcal{L}}{\partial S} \quad w^3_2 \quad f'(h^2_2) \quad a^1_2 \end{aligned} \tag{4.11}$$

The second and fourth terms are obtained during the forward pass, which does not require additional computations. The first and third terms depend on the form of the loss

$\mathcal{L}$ and the activation function f. By choosing adequately these two functions, the evaluation of the partial derivative can be computationally cheap. The four terms obtained can be assigned to different arrows in the graph of the network. As seen in Fig. 4.6, each term allows to go back to the node indicated in the denominator, forming a chain of derivatives that propagated backward. This chain is accordingly known as backpropagation.

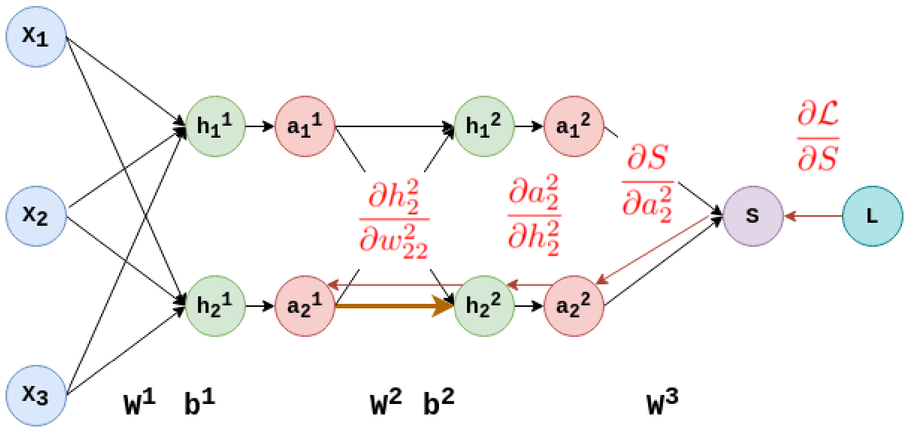
x_1
x_2
x_3
$h_1{}^1$
$a_1{}^1$
$h_2{}^1$
$a_2{}^1$
$h_1{}^2$
$a_1{}^2$
$h_2{}^2$
$a_2{}^2$
S
L
$\frac{\partial \mathcal{L}}{\partial S}$
$\frac{\partial S}{\partial a_2^2}$
$\frac{\partial a_2^2}{\partial h_2^2}$
$\frac{\partial h_2^2}{\partial w_{22}^2}$
W^1 b^1
W^2 b^2
W^3

FIGURE 4.6: Backpropagation procedure applied to a multi-layer perceptron. The weight of interest is indicated by a gold arrow. The loss value L has been added to the graph since it is a function of the predictions S and the expected output y, namely $\mathcal{L}(S, y)$.

A further remark can be raised concerning the values of the weights. As gradient descent is an iterative procedure, it requires an initial guess of the optimal parameters of the network. However, in view of expression (4.11), these values cannot be set to zero. If it was the case, the gradients would be null and the weight values would remain zero.

4.1.3 Activation and loss functions

Activation functions

The role of activation functions is multiple. They introduce the requested non-linearities while constraining the outputs of neurons, which in turn influence the learning phase. In view of (4.11), it is clear that the activation functions must be differentiable. Another fundamental condition concerns computational performance. As state-of-the-art neural networks can include up to billions of parameters, the forward pass has to be fast if we want to complete the training in a decent period of time. Therefore, activation functions must show a very simple analytical expression. Some of the most widely used activation functions are shown in Fig. 4.7.

The ReLU [99] and its variants, Leaky ReLU [100] and ELU [101], are used in most modern networks. Note that the ReLU and the LeakyReLU are not differentiable in zero. In practice, this has no consequence on the evaluation of the gradients (4.11) since the values that flow in the networks are never exactly zero. The hyperbolic tangent and the Sigmoid are particularly useful to constrain the values of the nodes in the intervals $[-1, 1]$ and $[0, 1]$ respectively. They are mostly added at the last layer of the network in order to produce probabilities or confidence scores.

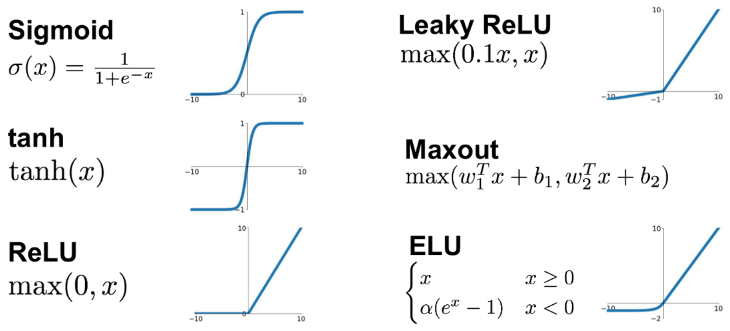

FIGURE 4.7: Some of the activation functions used in current neural networks.

The loss function

The choice of the loss is conditioned by the objective of the training. For what concerns neural networks, the training procedure is almost exclusively supervised, i.e. the goal is classification or regression. Although recent developments have been proposed to build unsupervised learning strategies based on neural networks, their use is still limited [102]. We will therefore focus on losses adapted to classification and regression tasks.

When the goal is to achieve the prediction of some continuous values such as the price of houses or temperatures, regression losses are needed to provide accurate feedback to the network. The Mean Squared Error (MSE) and the Mean Absolute Error (MAE) are among the most frequently used losses. Their analytical expressions are given by:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_{i,true} - y_{i,predicted}\right)^2 \quad \text{and} \quad MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_{i,true} - y_{i,predicted}\right|. \tag{4.12}$$

where $y_{i,true}$ and $y_{i,predicted}$ stand for the labels and the network's predictions respectively, and the sum is performed over a number n of outputs. The MSE has the advantage of penalizing large errors while showing no local minima, which almost guarantees the training to converge to the optimal parameters. However, if the data present some outliers, they can impact the training with undesired large errors. This does not arise with the MAE which applies the absolute value. The MAE is nonetheless more computationally expensive and the absence of local minima cannot be guaranteed. A compromise of both losses is found through the Huber loss [103]:

$$L_\delta = \begin{cases} \frac{1}{2}\left(y_{true} - y_{predicted}\right)^2, & \text{if}\left|y_{true} - y_{predicted}\right| \leq \delta \\ \delta\left|y_{true} - y_{predicted}\right| - \frac{1}{2}\delta^2, & \text{otherwise} \end{cases} \tag{4.13}$$

which behaves like the MSE for small errors and is similar to the MAE for large errors. The only drawback of the Huber loss is that the threshold δ that distinguishes small errors from large ones needs to be tuned, which highly depends on the problem considered.

Classification can be viewed as the prediction of discrete outputs corresponding to the different classes expected. When only two classes are considered, the Binary Cross Entropy (BCE) is used:

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}\left(y_{i,true}\, log(y_{i,predicted}) + (1 - y_{i,true})\, log(1 - y_{i,predicted})\right) \tag{4.14}$$

Note that the use of the logarithm implies the predictions $y_{i,predicted}$ to be bounded in the interval $[0,1]$, which can be satisfied with the use of the Sigmoid activation. More generally, for a number C of classes, the above expression becomes:

$$CE = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{C}\left(y_{ij,true}\, log(y_{ij,predicted})\right) \tag{4.15}$$

which is known as the Cross-Entropy (CE). The latter requires to add the Softmax activation at the last layer of the network:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \quad for\ i = 1, 2, \ldots, C \tag{4.16}$$

where the sum is performed over C classes. The Softmax normalizes the output to a probability distribution over the expected classes, enforcing the network to output a vector of values for which the sum equals 1.

4.1.4 Optimizers, initialization and normalization

Optimizers

Training a neural network consists in computing the gradients via backpropagation and using the update rule to modify its weights. With this method, the only free parameter is the learning rate γ. However, the update rule (4.4) does not converge well in practice. This happens because of the loss expression, the architecture of the network and the complexity of the data. Altogether, they can lead to a very complex problem for which it is not easy to find the optimal parameters, i.e. the minimum of the loss curve. As neural networks have up to billions of parameters, a direct representation of this curve, as it is done for a single parameter in Fig 4.3, is therefore impossible. Li et al. [104] have thus developed a method that allows to visualize the loss curve by projecting this high-dimensional surface onto a 3D space. An example of the loss surface for a deep neural network is shown in the left panel of Fig. 4.8. With this representation, it is possible to understand why our update rule is not performing well.

Consider that we want to find the minimum of the loss surface in Fig. 4.8 starting from the green dot. At that point the loss is maximum and the gradient is zero in a small neighbourhood. Let us consider that we manage to find a way to move from that point and we start to follow the gradient, being the slope of the surface. As far as the slope is negative, we keep moving down till we land on a flat area (yellow dot). There, as the gradient is close to zero, the update rule $\theta_{t+1} = \theta_t - \gamma\nabla_\theta\mathcal{L}(\theta_t)$ will not help us to move away from that point. It is even worse if we account for the slight positive slope next to the canyon shown in dark blue. Considering only the local gradient to decide where to move in the parameter space is generally not a good idea. Instead, the inertia acquired during the descent could help to move in the direction of the canyon, where the minimum probably lies. Momentum-based techniques have been proposed to add inertia in the choice of the step direction according to:

$$\begin{aligned} \theta_{t+1} &= \theta_t + u_t \\ u_t &= \alpha u_{t-1} - \gamma\nabla_\theta\mathcal{L}(\theta_t) \end{aligned} \tag{4.17}$$

where u is called the velocity and α is a constant parameter to weigh the contribution from the previous step. A graphical representation is shown in the right panel of Fig. 4.8.

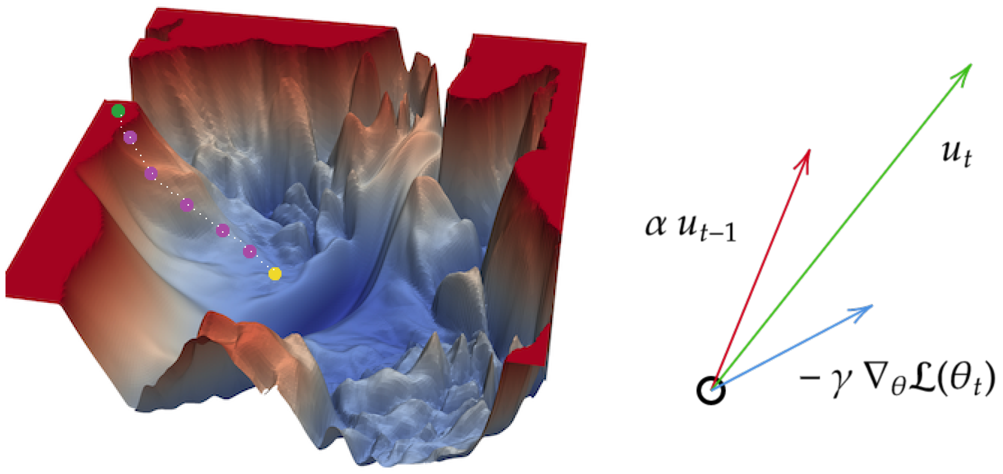

FIGURE 4.8: Left: Loss surface of a deep neural network according to the projection defined in [104]. The height of the surface indicates the value of the loss in this projected space. Adapted from [104]. Right: Representation of the principle behind momentum-based techniques. The final step u_t is a combination of the local gradient $\nabla_\theta \mathcal{L}(\theta_t)$ and the previous step u_{t-1}.

The algorithms that compute the gradients, apply the backpropagation procedure and implement momentum-based techniques are called optimizers. All in all, they are responsible for the convergence of the training to the optimal network's parameters. In practice, two optimizers implementing momentum-like methods are widely used: ADAM [105] and RMSprop [106]. Both involve additional steps compared to (4.17) but the purpose is identical.

Initialization

In the previous part, we tried to find the minimum of a loss by following the slope of the loss surface. Our set of initial parameters, namely the green dot, was a poor choice. Indeed, we end up in a flat region where the loss is at its maximum. If we had to follow the update rule $\theta_{t+1} = \theta_t - \gamma \nabla_\theta \mathcal{L}(\theta_t)$, we would be stuck there. Here, the momentum cannot be used since there is no previous step. This inconvenient position can be solved by randomly choosing another point on the surface, with no guarantee to end up in a different situation. Instead, some strategies have been defined in order to discard poor initial guesses. The most effective schemes are the Xavier [107] and He [108] initializations[2]:

$$\text{Xavier :} \quad w_{ij}^l \sim \mathcal{U}\left[-\sqrt{\frac{6}{q_l + q_{l-1}}}, \sqrt{\frac{6}{q_l + q_{l-1}}}\right] \tag{4.18}$$

$$\text{He:} \quad w_{ij}^l \sim \mathcal{U}\left[-\sqrt{\frac{6}{q_{l-1}}}, \sqrt{\frac{6}{q_{l-1}}}\right] \tag{4.19}$$

where w_{ij}^l is a weight of the layer l, q_l is the number of neurons in the same layer and $\mathcal{U}[-a, a]$ denotes a uniform distribution in the interval $[-a, a]$. They both rely on the conservation of the variance of the weights across the layers. Preserving the variance of the

[2]Note that these expressions are the uniform version of the Xavier and He distributions. A Gaussian variant has also been proposed for both methods.

weights allows the information to flow in the network without reducing or magnifying its amplitude, which is a major concern for deep neural networks.

Normalization

In both Xavier and He initializations, the values of the weights in a layer depend on the number of neurons in the preceding layer. Their result can be obtained by enforcing the variance of the values stored in the neurons to be identical for all layers, i.e. $\mathcal{V}(h^l) = \mathcal{V}(h^{l-1})$. Consequently, this condition also holds for the very first layer $\mathcal{V}(h^1) = \mathcal{V}(h^0)$, where $h^0 = x$ are the input nodes (see Fig. 4.6). This further implies that the inputs should follow a certain distribution. In practice, this constraint is rarely satisfied since the input data can have very different natures (price, temperature, air humidity, etc.) which all have different scales and units. However, we can enforce this condition by normalizing the data to a Gaussian distribution via

$$\begin{gathered} x' = \frac{x - \hat{\mu}}{\hat{\sigma}} \quad \text{with} \\ \hat{\mu} = \frac{1}{N} \sum_{x=0}^{N-1} x \quad \text{and} \quad \hat{\sigma}^2 = \frac{1}{N} \sum_{x=0}^{N-1} (x - \hat{\mu})^2 \end{gathered} \tag{4.20}$$

where N is the total number of inputs.

In some cases, it does not really make sense to transform your data, in particular when you are interested in their distribution. That is why another method has been proposed to normalize the data: batch normalization [109]. It consists in performing the normalization in between the layers of the network, by normalizing over the mini-batch. The normalization therefore becomes a special kind of layer that can be added while defining the network. Its expression is almost identical to (4.20) except that the mean and the standard deviation are estimated over a subset of the data. Batch normalization has shown to be very effective, reducing the training time by up to a factor of 10 for some network architectures [109].

4.1.5 Methodology to train neural networks

The training of a neural network is nothing else but an optimization problem with thousands or millions of parameters. The goal of the training is to minimize a loss function that corresponds to the problem we want to solve, i.e. classification or regression. However, in the case of a small dataset or if the neural network has too many parameters, the loss can decrease while the network does not learn how to solve the problem. Instead, the network memorizes the inputs and assigns them the optimal output in order to reduce the loss, which is the only criterion to minimize. If we show new data to the network after the training, it will not know how to perform its task and it is likely to give a random output. This problem is known as overfitting. A way to solve overfitting is to separate the data into two sets: a training set and a validation set. The former is used to train the network, meaning computing the gradients and performing the backpropagation to update the weights. The latter controls the training by taking care that the network performs well on a new unseen dataset. The loss is thus also evaluated on the validation set and its value indicates when to stop the training in case of overfitting. When the training loss decreases but the validation loss increases, this reveals that the network starts to overfit the training data. The optimal solution therefore lies slightly before the overfitting region, as seen in the left panel of Fig. 4.9. Usually, the validation set comprises from 5 to 20% of the total data available, leaving a significant portion to train the network.

Once the network has been designed and the weights have been initialized properly, we still have to choose the value of the learning rate and the batch size. Both are related since the batch size affects the evaluation of the loss over the data and the learning rate influences the contribution of the gradients in the weight updates. Therefore, they have to be adjusted together. It is however recommended to set the batch size to a power of 2 to increase the efficiency of the matrix operations performed on the GPU. For what concerns the learning rate, its value can have a substantial impact on the loss curve, as shown in the right panel of Fig. 4.9. A high learning rate induces large value updates even in the case of small gradients, which can prevent the optimizer to find the optimal set of parameters. Contrarily, a low learning rate tends to attenuate the feedback from the gradients and extends the time needed to reach the optimal solution.

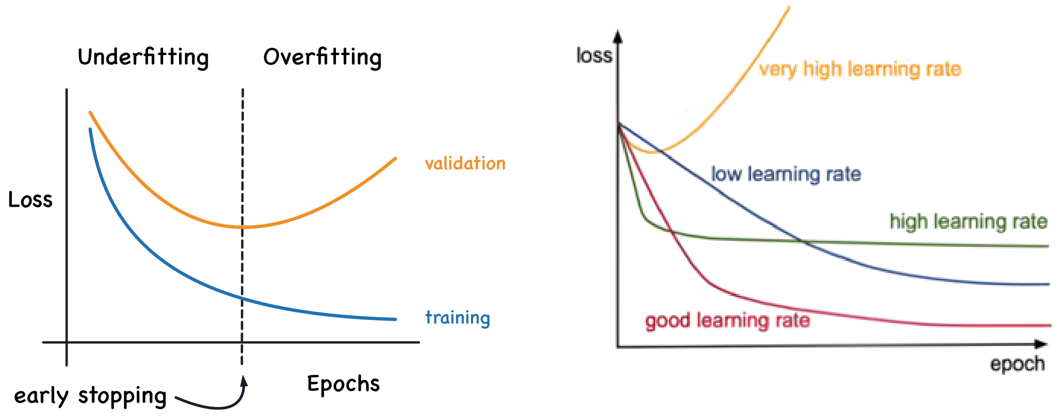

FIGURE 4.9: Left: Representation of the overfitting problem in neural network training. A validation dataset is used to control and prevent overfitting. The optimal training is achieved when the validation loss starts to rise, as depicted by the dashed line. Right: Effect of the learning rate on the evolution of the loss. A low learning rate induces a long training time while a high learning rate does not yield the optimal solution. The plot on the right is taken from [110].

4.2 Convolutional neural networks

4.2.1 From MLPs to convolutions

The first ever neural network, the multi-layer perceptron (MLP), processes the input data as one-dimensional arrays. In lots of applications, it is required to analyze images or videos in order to detect specific shapes or unusual events. To use this very first architecture, we might need to flatten our 2D or 3D arrays into 1D inputs. If we consider images of shape 1000x1000, the total number of input nodes will be 1 million. Typical MLPs usually show at least 3 layers for which the number of neurons decreases with depth, starting from 100 or 1000. In the former case, the first layer of our MLP would contain 100 million parameters. Such a large number of trainable parameters is intractable for most researchers and companies.

A way to reduce this number consists in rethinking how the inputs are combined inside the network. In MLPs, every neuron is a combination of all the neurons in the previous layer. That is the brute-force method which might be useful when we have no prior knowledge of the data. However, in images and videos, there is no need to combine all the pixels together to highlight a specific event such as a tennis player hitting a ball. The local pixels around the player are enough to distinguish what is happening. With this idea in mind, researchers have proposed to introduce convolutions in neural networks [111]. Convolutions make use of kernels that slide over the data as seen in Fig. 4.10. The output of a convolution is a smaller array whose values are a linear combination of the kernel values with the initial array. In computer vision, kernels are widely used to highlight lines or curves in images. The choice of the values inside the kernel conditioned the output. In neural networks, the kernel values are left as free parameters so that the network can learn which filters lead to the best result.

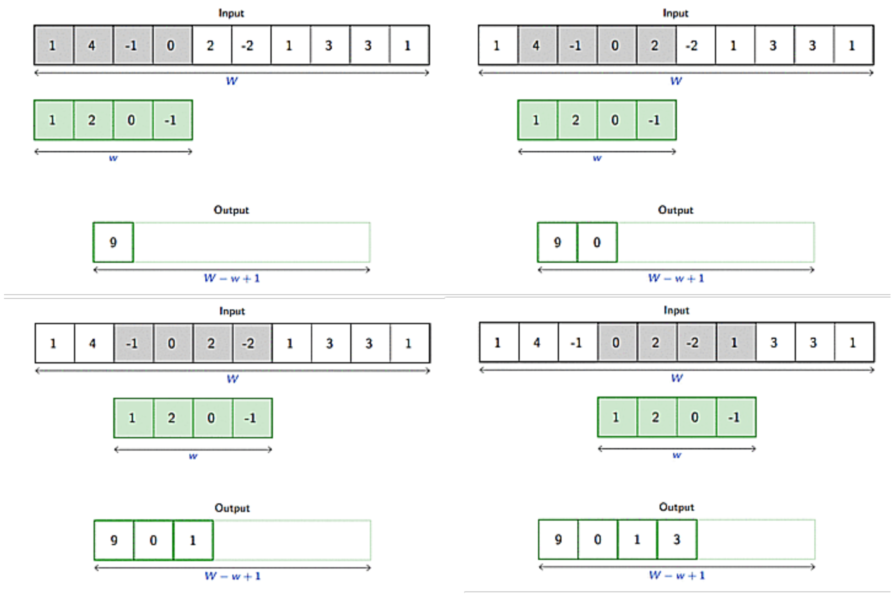

FIGURE 4.10: Illustration of how a convolution kernel is applied to a 1D array. From top left to bottom right, the kernel slides over the array, combining only the local information to produce an output value.

An interesting property of convolutions is their ability to be stackable. As the output of a convolution is a smaller array, we can feed this array to another convolution which will further combine these first-stage features. This property, known as feature hierarchy, is the fundamental principle in convolutional neural networks (CNNs). A second characteristic concerns their invariance to translation. Since the kernel slides over the array, it applies the same filter all over the data, allowing to highlight features even if they are translated within the data[3].

The basic principle behind CNNs is to apply successive convolutions to the input data to allow the network to learn features at different levels. The first layers detect low-level features such as lines or colors while the deeper layers combine them into more complex objects like a wheel or a tree. This procedure can be applied in any dimension, making CNNs particularly suitable to analyze images and videos.

[3]Note that convolutions are not invariant to rotations or other transformations like squeezing or stretching.

4.2.2 Parameters of a convolution

The convolution operation can be adapted via four main parameters: the kernel size, the stride, the padding and the dilation. As in many edge detection algorithms, the kernel size can be tuned. A larger kernel has little effect on the shape of the output array while it increases the number of parameters to train. As seen in Fig. 4.11, a single convolution with a 5x5 kernel is equivalent to two stacked 3x3 convolutions while having more parameters ($5^2 = 25$ versus $2 * 3^2 = 18$). The general rule is to build deep networks with small kernels, which benefits memory usage and training time. Also, odd-sized kernels are preferred over even-sized kernels because of symmetry. For odd-sized kernels, the parameters of the kernel are symmetrically distributed around the output pixel, which is not the case for even-sized kernels. As a consequence, some distortions can happen in the output array. Small odd kernels are therefore the common choice in CNN architectures.

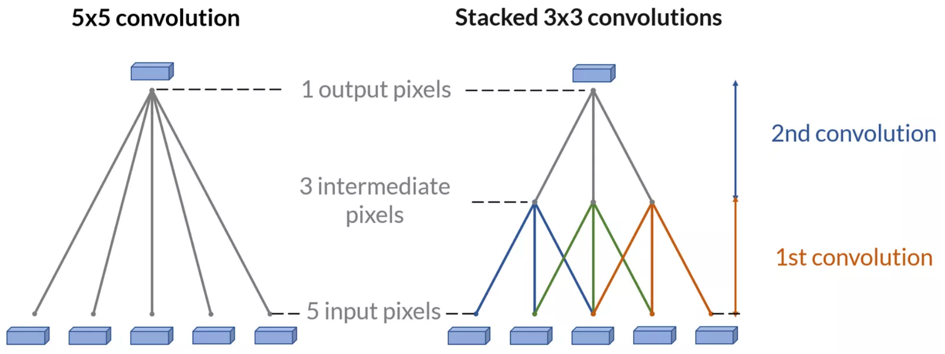

FIGURE 4.11: Comparison of a 5x5 convolution and 2 stacked 3x3 convolutions applied over a vector of 5 input values. Stacked convolutions with small kernels lead to fewer parameters than convolutions with large kernels, reducing the cost in memory and the training time. Adapted from [112].

The stride is the second free parameter of a convolution. It represents the step in between two consecutive applications of the kernel over the input pixels. A classical convolutional that has a step of 1 pixel therefore shows a stride of 1. Strides of 2 or 3 are widely used since they reduce the size of the input array by a similar factor. An example of the effect of the stride on a convolution is seen in Fig 4.12.

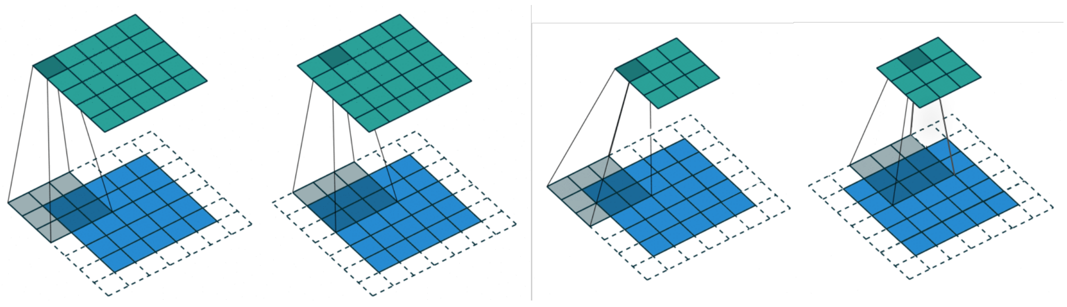

FIGURE 4.12: Representation of the convolution operation with stride $s = 1$ (left) and stride $s = 2$ (right). The size of the output array is highly dependent on the value of the stride.

The third free parameter is known as the padding. It consists in adding values around the input array to control the size of the output, as depicted in Fig 4.13. In general, the arrays are padded with zeroes so that they do not affect the linear combination of the input values. There exist other padding modes that reflect or copy the boundary values. Padding can also help reduce the loss of information at the borders. As the values at the border appear only once in the convolution process, their information could vanish in deep networks. Padding the input can eliminate this effect.

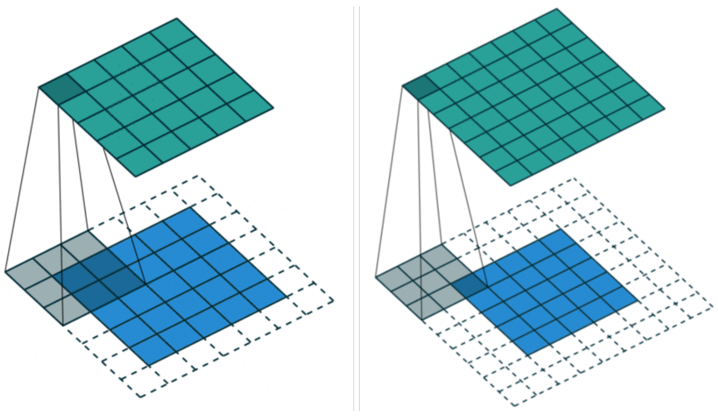

FIGURE 4.13: Illustration of a convolution operation applied on zero-padded input arrays. A single (left) or double (right) padding is shown. The output of the two operations only differs by the width of the padding.

The dilation is the last adjustable parameter of a convolution. Dilation refers to the spacing left between the input values when applying the kernel on the input array. A dilation factor of 2 indicates that 1 pixel out of 2 will be combined through the kernel. By default, there is no space left between the input pixels and the dilation factor is equal to 1. Dilated convolutions are useful to increase the field of view of a single convolution without increasing the number of parameters. A 3x3 dilated convolution sees as far as a classical 5x5 convolution but with only 9 parameters to train. Fig. 4.14 shows how dilation is used in convolution operations.

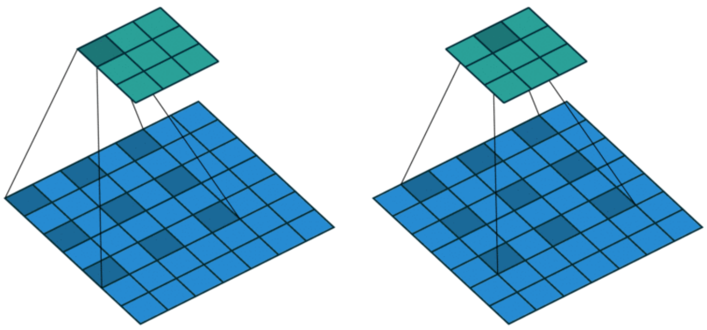

FIGURE 4.14: Illustration of two successive steps of a convolution operation with a dilation factor of 2.

A convolution operation with kernel size k, stride s, padding p and dilation d transforms an input array according to:

$$W_{out} = \left\lfloor \frac{W_{in} - 1 + 2\,p - d\,(k-1)}{s} + 1 \right\rfloor \tag{4.21}$$

where $\lfloor . \rfloor$ specifies the nearest integer down and W_{in} and W_{out} stand for the input and output sizes respectively. In the case of 2D or 3D arrays, this formula holds in any dimension.

Kernel size, stride, padding and dilation are set prior to the training of the network and are thus not trainable. We distinguish them from the trainable parameters by calling them hyperparameters.

4.2.3 Ingredients of a CNN

The main idea behind convolution networks is to exploit a feature hierarchy. Low-level features are detected by the first layers and combined into more complex features as they go deeper in the network. However, the convolution operation is still a linear operation and some non-linearities are needed. That is why activation functions are inserted in between each convolution operation. Each value of the output array will then be passed to an activation function such as ReLU or ELU.

The objective of a CNN is often to classify images or to give them a score based on a specific feature. For this, the network has to produce a vector of classes or a single value accordingly. In the case of large images, reducing their size down to a single value can be difficult and requires to apply a lot of stacked convolution layers. This further increases the number of weights in the network, extending the training time. Instead of using dozens of convolution layers, we can use a mix of convolution and fully-connected layers. The set of convolution layers then serves as a feature extractor, which combined them through several fully-connected layers. The latter are easier to tune since we precisely set the number of neurons in the output layers, which is very practical in the case of a classification task. In general, up to 3 fully-connected layers are used after the convolution part. A lot of state-of-the-art CNNs have adopted this mixed architecture [113, 114, 115].

Training a CNN consists in discovering and transferring features that can help perform the desired task. However, a convolution layer is defined by a unique kernel and cannot learn many features at once. If we want to transfer different features, we need to apply different kernels at the same stage in the network architecture. In practice, every convolution layer applies N_l kernels to the input array, leading to N_l outputs, called feature maps. The next layer then applies its first kernel to each feature map and sums them to form the first feature map. This procedure is repeated over the N_{l+1} kernels of that layer. Although there is no golden rule in the number of kernels to apply at the different layers, it is recommended to increase it by a factor of 2 from one layer to the next one. This allows to combine the low-level features into even more high-level features that will be used to produce the final output. The number of feature maps at each convolution layer is therefore another hyperparameter that needs to be chosen before the training phase.

Strided convolutions are not the only layers to reduce the size of the input arrays. In a lot of applications, pooling layers are used. They consist in parameter-free operations, like averaging or maxing out, that reduce the size of the input array by an integer value as seen in Fig. 4.15. The pooling operation can be viewed as a convolution with a stride equal to the kernel size, also known as pooling factor. In this way, they synthesize the information contained in each block of the input array, leading to an array that has been reduced by a factor equal to the pooling factor. Contrary to strided convolutions, these layers do not have trainable parameters. In applications where large images are involved, they can alleviate the memory requirements and therefore accelerate the training time.

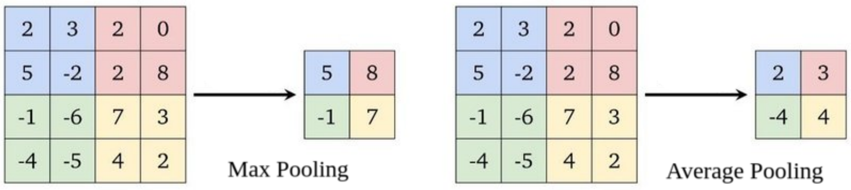

FIGURE 4.15: Illustration of the maximum (left) and average (right) pooling operations on a 4x4 input array. In both cases the pooling factor is equal to 2.

A typical convolutional neural network is therefore composed of convolution and fully-connected layers, activation functions, pooling layers and possibly batch normalization layers. Fig. 4.16 shows how these layers interact in a typical architecture used for classification tasks. As the input data are images, the output features need to be flattened before passing them to the fully-connected layers. Note that there is no need to add an activation function after a pooling operation since it does not contain any trainable parameter.

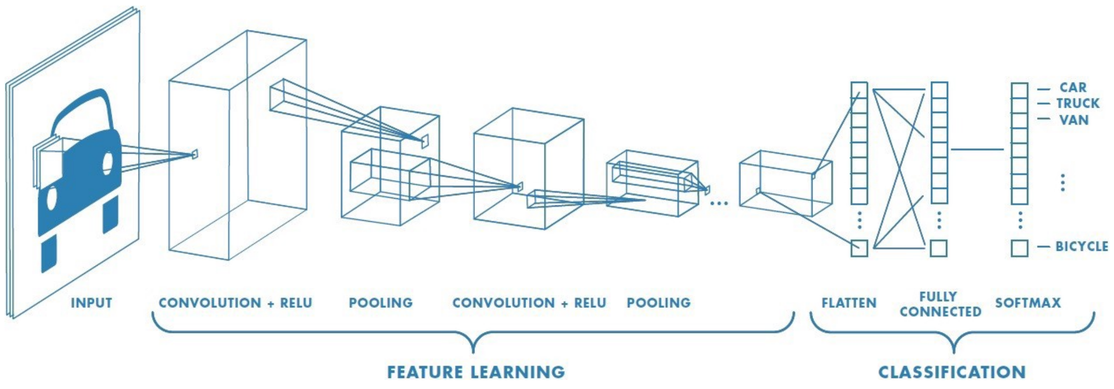

FIGURE 4.16: Common CNN architecture used for classification. The network is made of two parts: a set of convolution layers and 2 fully-connected layers. The former serves as a feature learning algorithm while the latter takes the final decision by combining the high-level features. The last layer returns a vector quantifying the probabilities for the image to belong to each class. Taken from [116].

In applications like image segmentation or super-resolution, the goal is not to produce a class vector but rather an image. However, all the precedent layers only reduce the image size or leave it unchanged[4]. That is why a reverse convolution operation has been implemented, known as transposed convolution. Fig. 4.17 shows how they work. Transposed convolutions can be tuned in the same way as classical convolutions, i.e. with the same hyperparameters.

Transposed convolutions are used in numerous neural network architectures to upscale the information [117, 118]. In general, these networks comprise two parts. The first part is a series of convolutions that learn the relevant features in the input and the second is a set of transposed convolutions that upscale these features up to the original image size. This is particularly well suited for image segmentation tasks where we aim at locating objects and their boundaries in the input image. Transposed convolutions therefore

[4]Padding can be used to slightly increase the image size up to a certain limit that is the size of the kernel. Beyond that, this is equivalent to adding redundant or useless information in the network.

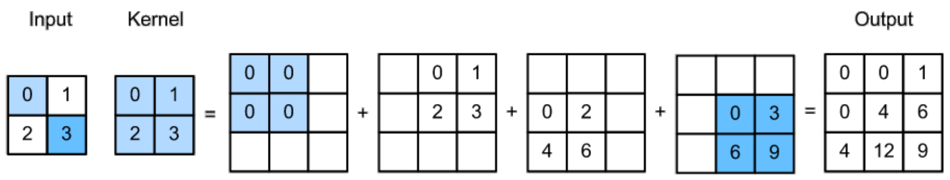

FIGURE 4.17: Illustration of a transposed convolution applied to a 2x2 input array. The kernel is multiplied by each value in the input and the intermediate results are stored adequately in larger arrays. The final step consists in summing all the intermediate arrays into the final output.

act as methods to upscale precisely the learned features. However, transposed convolutions might suffer from undesired artifacts in the image produced [114, 119]. This happens when the kernel size is not a multiple of the stride [120] as seen in Fig. 4.18. In particular, it appears with the ideal kernel size of 3 and the minimal stride of 2. It consists in input values that are overused with respect to their immediate neighbors, creating a checkerboard pattern in the output. To circumvent this problem, Odena et al. [120] have proposed to use parameter-free upsampling methods followed by a convolution. The suggested upsampling strategies are either the nearest neighbor algorithm or the bilinear interpolation. They both consist in interpolating the values of the input array to enlarge the latter by an integer factor.

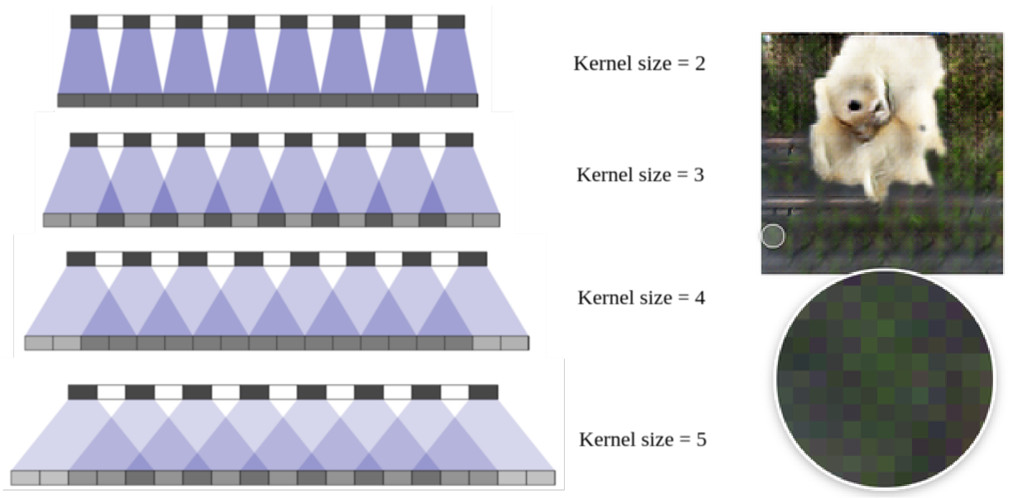

FIGURE 4.18: Checkerboard artifacts introduced by transposed convolutions in one-dimensional arrays (left) and images (right). The stride used in the left image is 2. The repeated pattern is the consequence of the kernel size not being a multiple of the stride. Both images are adapted from [120].

In the following chapter, we will design a search strategy based on convolutional neural networks. As the essence of burst searches is to identify clusters of pixels in time-frequency images, CNNs are the natural choice to reach that objective. For this, we will build our network with some of the ingredients defined above. Table 4.1 summarizes all the layers and presents their hyperparameters, trainable weights and possible varieties.

	Hyperparameters	N° **of weights**	**Varieties**
Convolution	Kernel size k, stride s, padding p, dilation d, N° of feature maps N_f	$N_f * k^2$	/
Fully-connected	N° of neurons in the layer N_l	$N_l * N_{l-1}$	/
Pooling	Pooling factor	/	Maximum or average pooling
Transposed convolution	Kernel size k, stride s, padding p, dilation d, N° of feature maps N_f	$N_f * k^2$	/
Upsampling layers	/	/	Nearest neighbor or bilinear interpolation
Batch normalization	Batch size	/	/
Activation function	Negative slope of the Leaky ReLU or negative slope of the ELU or None	/	ReLU, Leaky ReLU ELU, Sigmoid, Tanh Softmax, etc.

TABLE 4.1: Summary of the layers used to build typical convolutional neural networks. The table shows the adaptable hyperparameters, the number of weights per layer as well the available varieties within that layer.

Chapter 5

Anomaly detection for Long duration BUrst Searches

5.1 Limitations from burst searches

5.1.1 Waveform models

The techniques used in burst searches aim at being model-agnostic. The various detection algorithms proposed in the different pipelines should therefore not be biased towards a particular waveform model. In the long duration regime, we can further assume that the signals are well-behaved in the time-frequency plane. This means that the signals should be spread over contiguous TF pixels forming thin solid curves. In terms of machine learning, the search for minute-long bursts is equivalent to finding curves in images. This is a task that suits particularly well to convolutional neural networks. However, if we train a CNN to recognize the handful of simulated models, it could lead to a bias in our search algorithm. CNNs perform well in identifying shapes and patterns that they have seen during the training phase but achieve poor results on new signals. If we want to build a true model-agnostic algorithm based on neural networks, we therefore need to find a way to mimic the burst models and broaden the parameter space that they cover in the TF plane.

Frequency-swept cosines are particularly suited to mimic the behavior of long duration burst models. They are defined through the expression:

$$y(t) = cos(\phi(t)) \quad \text{with} \quad \phi(t) = \int_0^t 2\pi\, f(t)\, dt \tag{5.1}$$

where $f(t)$ can be any function describing the frequency evolution over time. The *Scipy* library [121] allows to generate frequency-swept cosines following 4 different frequency evolutions: linear, quadratic, hyperbolic and logarithmic. Considering a signal starting at $(0, f_0)$ and evolving toward (t_1, f_1), the instantaneous frequency of the signal for each case is indicated in Table 5.1. For a quadratic evolution, note that the expression of $f(t)$ depends on where the vertex of the parabola is attached. This further determines if the parabola will be oriented upwards or downwards. In the case of a logarithmic frequency evolution, the requirement that f_0 and f_1 have the same sign is trivially ensured since we work with positive frequencies.

All these expressions are implemented within *Scipy* [121] and we can use them to generate long burst-like signals. We can even go further by mimicking the energy evolution displayed in some waveform models. In some emission mechanisms, the GW energy is not evenly released over the duration of the phenomenon. As an example, eccentric CBC events show more energy at the end of the signal since it corresponds to the merger phase.

Freq. evolution	**f(t)**	**Conditions**
Linear	$f(t) = f_0 + (f_1 - f_0)\frac{t}{t_1}$	/
Quadratic	$f(t) = f_0 + (f_1 - f_0)\left(\frac{t}{t_1}\right)^2$ $f(t) = f_1 - (f_1 - f_0)\left(\frac{t_1 - t}{t_1}\right)^2$	$\Rightarrow$ vertex at $(0, f_0)$ $\Rightarrow$ vertex at (t_1, f_1)
Hyperbolic	$f(t) = f_0 f_1 t_1 \frac{1}{(f_0 - f_1)t + f_1 t_1}$	f_0 and f_1 must be nonzero
Logarithmic	$f(t) = f_0 \left(\frac{f_1}{f_0}\right)^{t/t_1}$	f_0 and f_1 must be nonzero and have the same sign

TABLE 5.1: Mathematical expressions of some frequency evolution models over time. The mandatory conditions on f_0 and f_1 are shown in the right column.

For frequency-swept cosines, the energy distribution can be adjusted via a Kaiser filter [122] as shown in Fig. 5.1. For this, a Kaiser window twice as long as the signal is generated and the signal is multiplied by either the first or second half of the window. When the first half is implied, the final chirp shows an increasing energy as the signal evolves in time and the reverse holds when the second half is concerned. A shape parameter β is used to control the width of the Kaiser window. A larger beta parameter implies a narrower Kaiser window leading to a more uneven energy distribution across the signal duration.

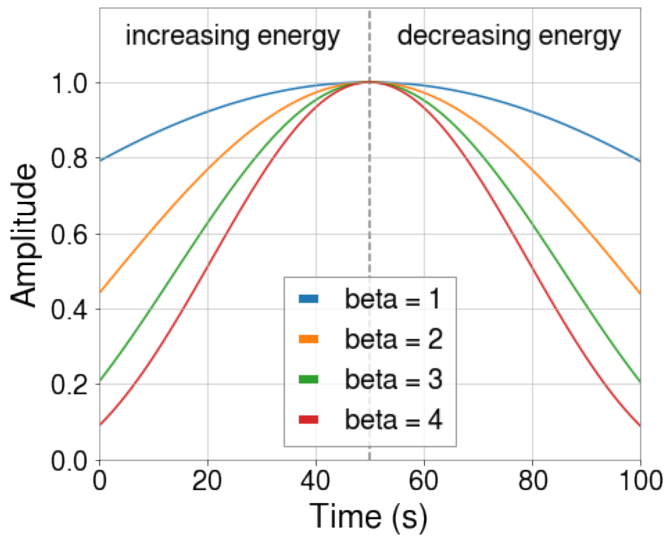

FIGURE 5.1: Examples of Kaiser windows with different values for the β parameter.

Ultimately, the starting and ending frequencies, the duration as well as the frequency and energy evolutions can be freely adapted to mimic long duration burst models. Fig. 5.2 illustrates some examples of frequency-swept cosines, also known as chirps, generated with the *Scipy* library. These chirps are almost identical to the long-duration burst models selected for O3 (see Fig. 3.3) while covering the full time-frequency plane. However, the harmonics that appear in GW emission models like *ISCOchirp* [79] or *ECBC* [83] are not reproduced. These harmonics come from emission mechanisms such as multiple mass moments in the torus around black holes [123] or eccentricity oscillations in eccentric compact binary coalescences [83]. They usually show less power than the main component of the gravitational wave and show up exclusively at high amplitudes. For now, we leave harmonics aside for the sake of simplicity. We will thus generate chirp signals to build our training dataset.

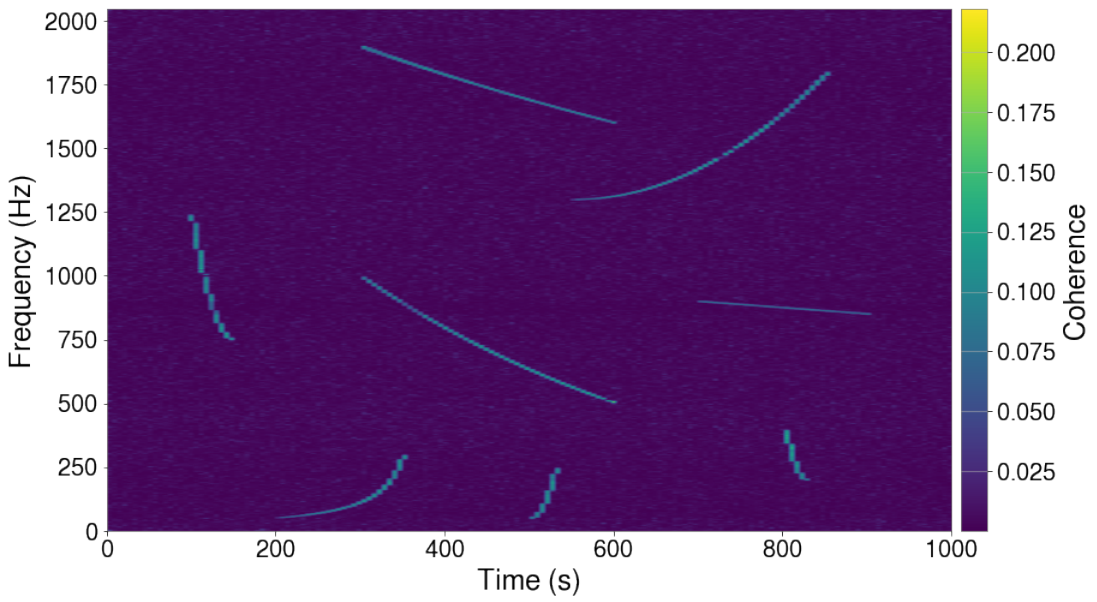

FIGURE 5.2: Examples of frequency-swept cosines injected into a coherence spectrogram. The frequency resolution is 2 Hz with time bins of 6 s.

5.1.2 Injection procedure

In burst searches, we measure the intensity of injected signals via the root sum squared value of the strain, defined as:

$$h_{rss} = \sqrt{\int \left(h_+^2(t) + h_\times^2(t)\right) dt} \tag{5.2}$$

However, this expression only depends on the injected signal itself and could vary with the local noise level. Fig. 5.3 shows a waveform model injected with a h_{rss} value of 10^{-21} in two different spectrograms. Although the model is injected with the same h_{rss} value in the two TF maps, the signal is buried in the background noise in the right panel while it is easily depicted in the left panel. This is the consequence of the variability of the noise spectrum over time. Such TF maps can fool a neural network during supervised learning and eventually cause the training to be badly conditioned [124]. It is critical to assign accurate labels to the training images to end up with a powerful neural network. A new injection criterion is therefore required to form a healthy dataset.

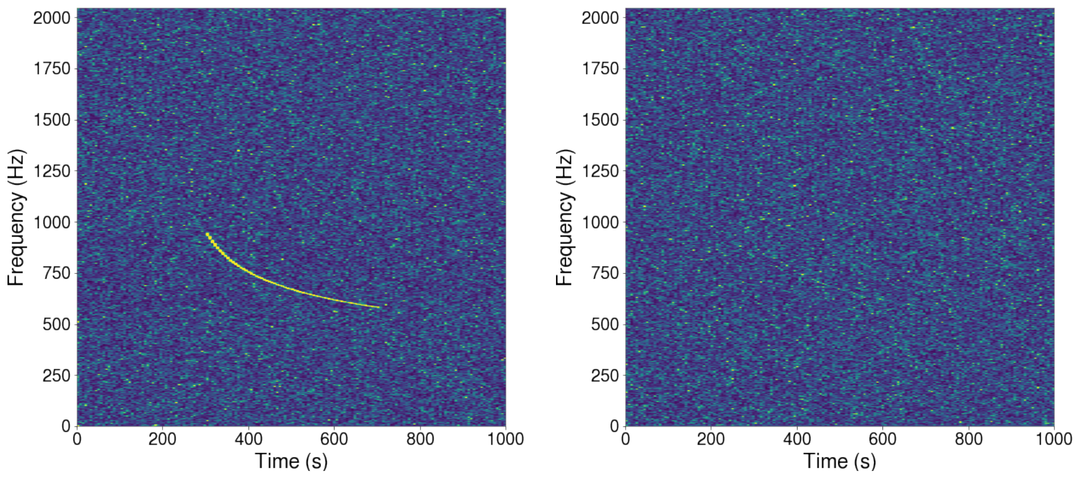

FIGURE 5.3: Injection of a *Magnetar-F* model [74] at 300 s with an h_{rss} value of 10^{-21} in two different O3a background coherence spectrograms. In the left image, the GW signal is easily recognized while it is buried in the background noise in the right image. The GPS time at the start of the H1 and L1 data in the left panel is respectively 1248306006 and 1248273184, while it is 1246174396 and 1246108524 for the right panel. The frequency resolution is 2 Hz with time bins of 6 s.

To be visible in time-frequency representations, an injected signal has to stand above the local noise level. Defining a noise-only coherence spectrogram N and the same spectrogram to which a signal has been injected by S, we propose a new injection criterion:

$$CA = \sum_{i,j} \left(S_{ij} - N_{ij}\right) \tag{5.3}$$

where the sum is carried over all the pixels (i, j) of the injected model. We call this new criterion the Coherent Amplitude (CA). The pixel-to-pixel difference allows to fine-tune how much a signal emerges from the local noise level and form a dataset with different levels of intensity. This is particularly useful when training procedures like curriculum learning [125] are used. With this new criterion, it is now possible to adapt the strength of signals to the noise level in TF images. Fig. 5.4 shows the same background TF maps, as in Fig. 5.3, in which a *Magnetar* model has been injected with an identical coherent amplitude. Contrarily to Fig. 5.3, both signals are seen equally well, leaving a clear footprint in the TF map.

As the coherent amplitude is defined on the final coherent spectrogram, we need to iterate over h_{rss} values to obtain the desired amplitude. The procedure consists in adapting the h_{rss} value to produce a TF map in which the signal shows the target coherent amplitude. For this, we stop the iteration loop when the current amplitude CA_{cur} is within 10% of the target amplitude CA_{tar}. The full injection algorithm can be seen in Alg. 5.1. The core algorithm is a basic dichotomous search among the stored h_{rss} values. Even if the algorithm is demonstrated on a burst model, the procedure can also be applied to chirp signals, which will allow us to build a faithful dataset to train our neural network.

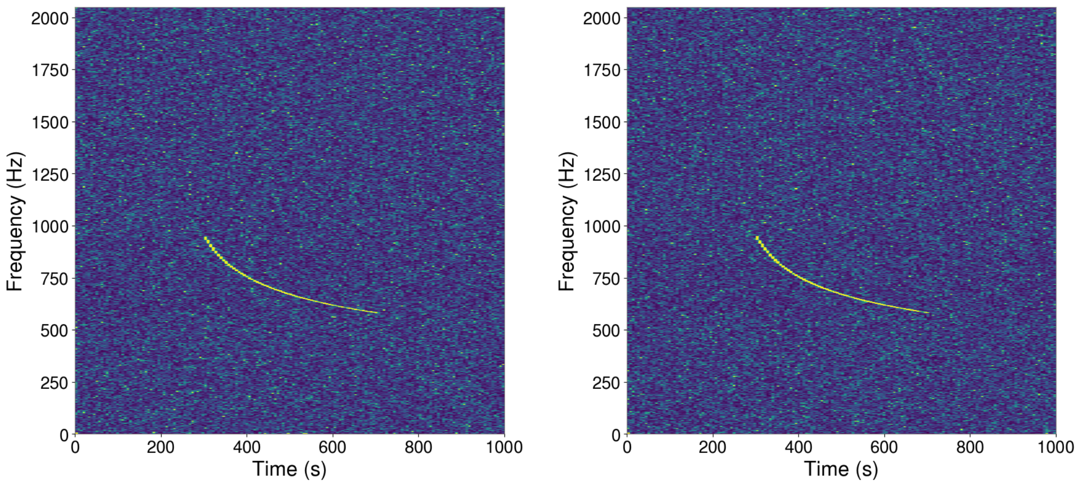

FIGURE 5.4: Injection of a *Magnetar-F* model [74] at 300 s with a coherent amplitude of 10 in two different O3a background coherence spectrograms. The equivalent h_{rss} value is respectively 10^{-21} and $2.7\,10^{-21}$ from left to right. Both GW signals appear with a similar intensity despite the very different background noise. The GPS time at the start of the H1 and L1 data in the left panel is respectively 1248306006 and 1248273184, while it is 1246174396 and 1246108524 for the right panel. The frequency resolution is 2 Hz with time bins of 6 s.

Algorithm 5.1 Injection procedure based on the coherent amplitude

```
1:  ◇ Fetch data
2:  ◇ Generate noise-only coherent spectrogram $N$
3:  ◇ Inject signal at default $h_{rss}$
4:  ◇ Generate coherent spectrogram $S$
5:  $\Rightarrow CA_{cur} \leftarrow \sum_{i,j} \left(S_{ij} - N_{ij}\right)$
6:
7:  while $CA_{cur} \notin [0.9 * CA_{tar}, 1.1 * CA_{tar}]$ do
8:
9:      ◇ Find closest $CA$ from above and below $CA_{tar}$
10:     ◇ Find corresponding $h_{rss}$ from above $h^{+}_{rss}$ and below $h^{-}_{rss}$
11:
12:     if $\nexists\, h^{+}_{rss}$ then
13:         $h_{rss} \leftarrow h^{-}_{rss} * 2$
14:     else if $\nexists\, h^{-}_{rss}$ then
15:         $h_{rss} \leftarrow h^{+}_{rss}/2$
16:     else
17:         $h_{rss} \leftarrow (h^{+}_{rss} + h^{-}_{rss})/2$
18:     end if
19:
20:     ◇ Inject signal at new $h_{rss}$
21:     ◇ Generate coherent spectrogram $S$
22:     $\Rightarrow CA_{cur} \leftarrow \sum_{i,j} \left(S_{ij} - N_{ij}\right)$
23:     ◇ Store new $h_{rss}$ and $CA_{cur}$
24:
25: end while
```

The relationship between the h_{rss} and the coherent amplitude depends on the local noise around the signal and is therefore not linear. Nonetheless, it is valuable to show how a change in h_{rss} affects the coherent amplitude and vice-versa. For this, let us evaluate the coherent amplitude on different noise backgrounds, as a function of the h_{rss}. The left panel of Fig. 5.5 illustrates how the relationship varies with the noise level. The change in coherent amplitude for a fixed h_{rss} can be substantial, reaching almost a factor of 7 at $h_{rss} = 10^{-21}$. This confirms why a h_{rss} is a poor choice to inject signals in images processed with neural networks. The right panel of Fig. 5.5 shows the dependency of the relationship to the waveform model used for a fixed background noise. As these models have different morphologies, they are all affected in a singular way by a change in h_{rss}. Note that the curves in Fig. 5.5 highly depend on the time and frequency resolutions. Indeed, a larger frequency bin implies more GW energy captured in a single bin, which increases the local coherence and in turn the coherent amplitude. However, a larger frequency bin also reduces the number of pixels in the signal footprint, reducing the coherent amplitude. Both effects act antagonistically on the shape of the curves shown below.

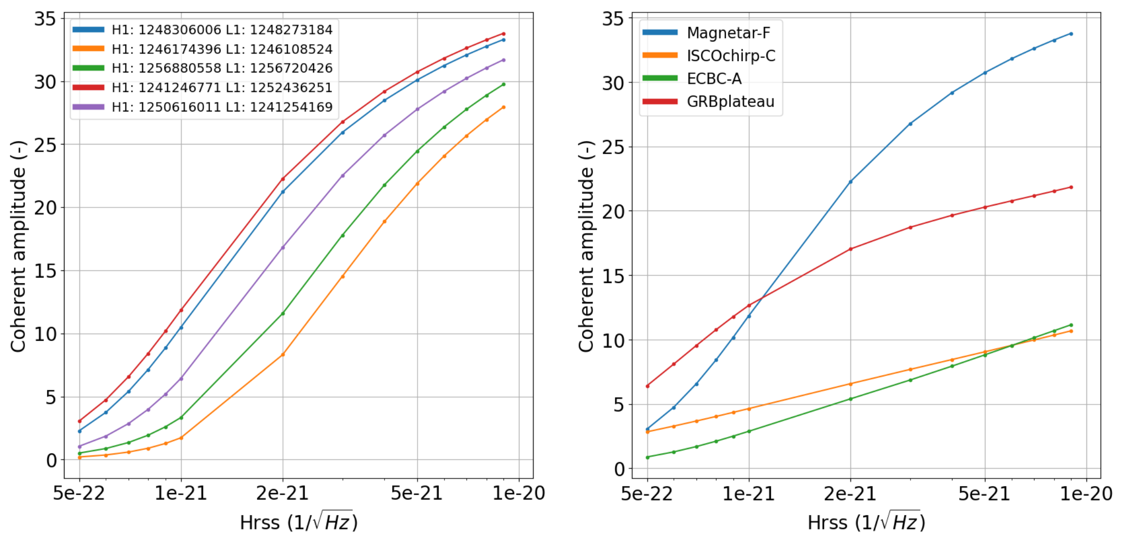

FIGURE 5.5: Coherent amplitude as a function of the h_{rss}. Left: A *Magnetar-F* model [74] is injected at 400 s in 5 different O3a noise backgrounds. The GPS time at the start of the H1 and L1 data is indicated in the legend of the plot. Right: 4 long-duration models have been injected into the same O3a noise background for comparison. The GPS time at the start of the H1 and L1 data is 1241246771 and 1252436251 respectively. The plots have been generated with frequency resolution of 2 Hz and time bins of 6 s.

5.1.3 Size of the time-frequency arrays

The time and frequency resolutions of the generated spectrograms have an impact on the sensitivity of the search. The longer the time segments, the more GW energy will be accumulated in a single pixel, leading to a higher coherence. As the noise appears to be coherent on very small timescales ($\ll 1$ second) [36], increasing the length of the time segments allows to reduce the coherence of the noise versus the coherence of hypothetical signals, yielding an increased signal-to-noise ratio. However, longer time bins will cause short signals ($\sim$ 10 seconds) to fall into very few pixels, making them harder to detect. An identical reasoning can be made for the frequency resolution. In this work, we use a 6-s time resolution combined with a 2-Hz frequency bin as a good compromise. Taking a 1000-s data stream from the Advanced LIGO interferometers spanning frequencies up to 2048 Hz results in coherence spectrograms with dimensions of *166x1025* pixels.

The time-frequency arrays are heavy to store in the default 32-bit *Python* precision because of their large amount ($> 10^5$) of pixels. Each TF map weighs about 13 megabytes.

As the RAM of the GPU is limited and should host both the network architecture and the current mini-batch, there exists a maximum batch size determined by the weight of the training images. Heavy images lead to small batch sizes, which eventually increase the training time and downgrade the performance [126]. It is recommended to use batch sizes above 10 to avoid a highly noisy gradient descent, and to take advantage of the speed-up of matrix-matrix products over matrix-vector products [126]. Noisy gradients are obtained when the gradients evaluated at each training iteration poorly generalize to all the samples in the dataset, which extends the time needed for the network to converge.

To allow larger batch sizes, we save the spectrograms as RGB images (8-bit integers). A further argument in favour of using RGB images is the wide use of this format in deep learning applications, e.g. some neural networks even require a 3-channel image as input [113, 127]. This transformation leads to a gain in memory of roughly 26 compared to the 32-bit *NumPy* arrays [128]. The reduction in memory requirements nonetheless comes with a small loss of precision in the values of the array. As RGB images save 256 (2^8) integer values, floating point values are rounded to 2^{-8} after rescaling. The maximum loss of precision for values falling right in between two integer levels is therefore $2^{-8}/2$, which is less than 0.2%. To evaluate the impact of such loss, we add a Gaussian noise with a standard deviation of 0.2% (0.002) to a TF map in which a glitch and a signal are present. Fig. 5.6 compares the TF map before and after the noise addition, respectively in the left and right panels. The TF images do not show any difference and both the signal and the glitch appear identical to the human eye.

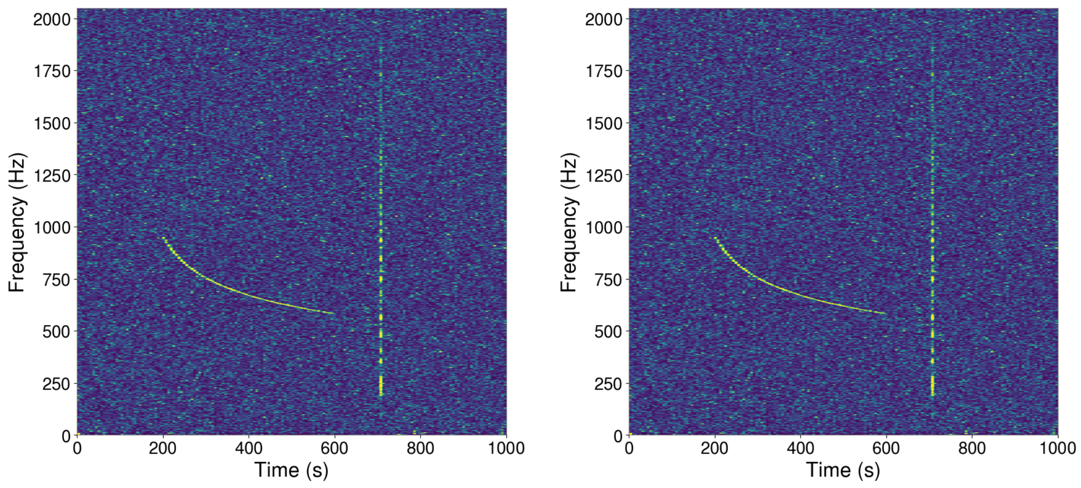

FIGURE 5.6: Comparison of TF maps with (right) and without (left) the addition of random Gaussian noise with a standard deviation of 0.002. The effect of the noise is barely visible and does not prevent a clear detection. A *Magnetar-F* model [74] has been injected at 200 s. The GPS time at the start of the H1 and L1 data is respectively 1264183464 and 1264172464. The frequency resolution is 2 Hz with time bins of 6 s.

Saving the coherent TF maps into RGB images leads to spread the information in 3 channels, whose balance depends on the color map used to draw the initial array. As these channels do not have any physical meaning, we choose to display evenly the information in all of them. For this, we use the *cubehelix* color map from the *Matplotlib Python* library [129]. As seen in Fig. 5.7, the GW signal is clearly visible in all channels and none seems to contain more information than the others. We therefore select this color map to build our training dataset.

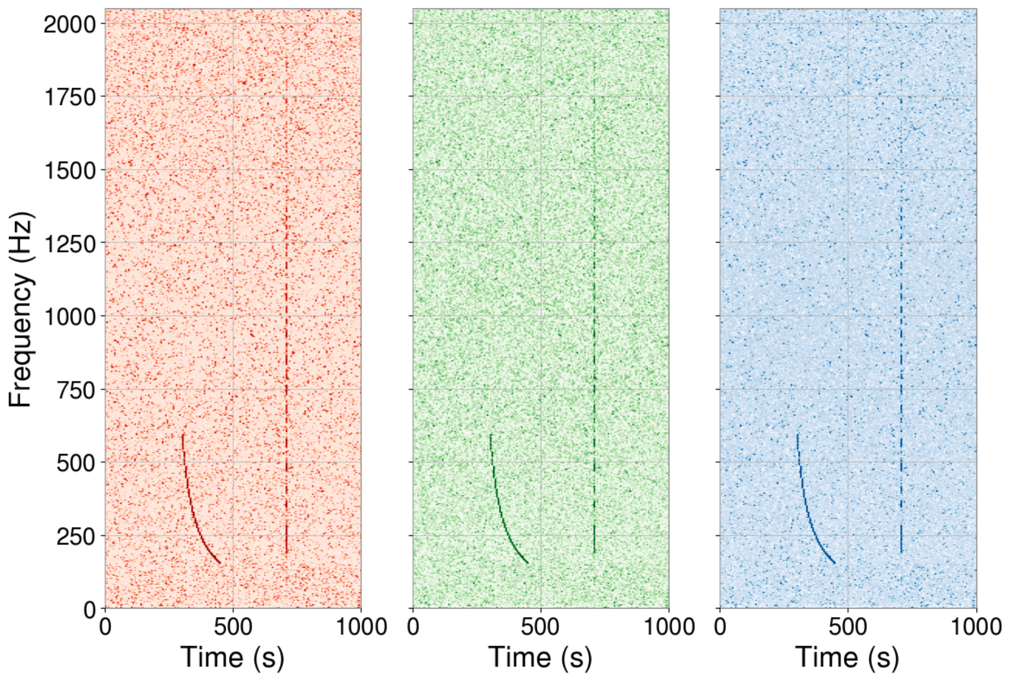

FIGURE 5.7: RGB representation of a TF map in which a chirp signal has been injected. The GW signal and the glitch are seen equally well in the 3 channels of the image. The GPS time at the start of the H1 and L1 data is respectively 1264183464 and 1264172464. The frequency resolution is 2 Hz with time bins of 6 s.

5.2 Method

5.2.1 Network architecture

Neural networks and machine learning techniques have been applied to many fields in gravitational wave physics [130]. Especially, convolutional neural networks have shown successful applications in the detection of binary black hole [131] and binary neutron star coalescences [132, 133] and estimation of their parameters [134], as well as in the classification of detector glitches [58]. CNNs have also been used in burst-related applications such as the detection or generation of short bursts [54, 135] and the identification of the GW counterpart from supernovae [136, 137].

Most of the applications of CNNs in GW physics consist in classifying data into several classes. In the search for GW signals, these classes reduce to either noise or signal. However, it is sometimes risky to rely on a unique number to state if there is a signal in the data. Given the gradual sensitivity improvement of the LIGO and Virgo detectors, it seems clear that the first burst event will hardly stand out of the noise in the TF images. In such a case, the class probability is likely to be close to 0.5 and we run the risk of missing it.

In this thesis, I propose to use a Convolutional Neural Network as a non-linear noise-removal filter in order to highlight pixels of interest. I described the full methodology in my first paper [138]. The objective is to produce a map where only the pixels that could belong to chirp signals remain. For this, we use an architecture inspired by [139]. The network is made up of two parts, a downscaling part that keeps the useful information through its different layers, and an upscaling part that aims at localizing precisely this information in a map with the same dimensions as the input of the network. The connections between the downscaling and upscaling parts are known as skipped connections, which are nothing else than concatenation operations. They help the network to learn how to precisely position the signals when upscaling the intermediate feature maps. Fig. 5.8 shows the architecture of the network, called ALBUS (Anomaly detection for Long duration BUrst Searches).

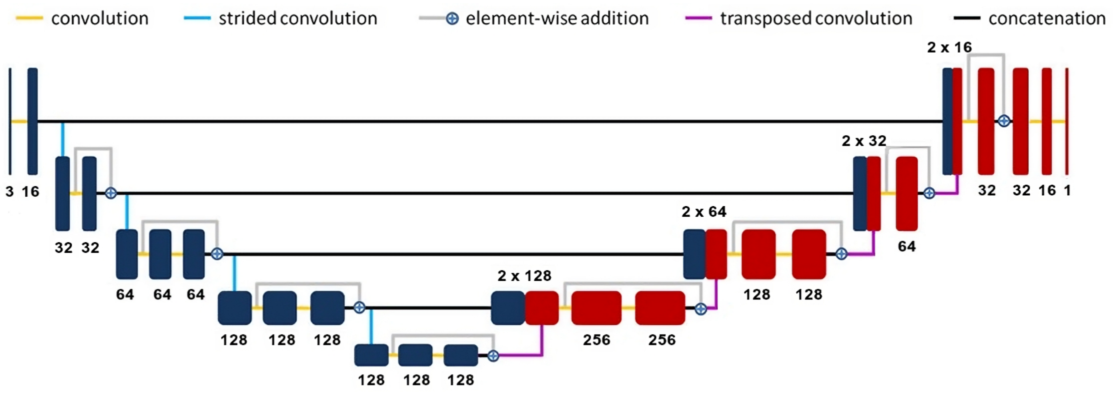

FIGURE 5.8: Architecture of ALBUS, modified from [139]. The downscaling and upscaling parts are represented in blue and red respectively. These two parts are coupled thanks to skipped connections, represented as concatenation lines. The numbers in black indicate the number of feature maps used at each stage of the network.

We exclusively use the ELU [101] as the activation function through the network and we add batch normalization after every convolution layer except the last one. Even though we aim at producing values in the interval $[0, 1]$, we do not constrain the output of the last convolution with a Sigmoid function. Once again, an ELU activation is used so that the network is free to learn the correct range of values. We also add dropout layers [140] after the convolution layers (yellow) showing 128 feature maps at the bottom of the downscaling network (blue). Dropout consists in randomly discarding some weights with a probability set by the user. It allows to prevent the network from being dependent on some features and therefore reduces the risk of overfitting. It is important to note that dropout layers are activated only during the training phase. For the layers of concern, we use a dropout probability of 50%. The full architecture of ALBUS can be seen in detail in Appendix A.

5.2.2 Dataset generation

To efficiently train our network, we need to feed it with noise-only TF maps as well as some containing chirp signals. For this, we use time-slides [76] applied on LIGO Hanford (H1) and LIGO Livingston (L1) data from O3a to generate 9000 coherence spectrograms (equivalent to 104 days of data). In half of them, we inject chirps at 9 different coherent amplitudes[1] (500 samples for each intensity). Note that some chirps drawn randomly might not contain enough pixels to reach coherent amplitudes above 20, as it is the case for *ISCOchirp* and *ECBC* models in Fig. 5.5. In such a case, the maximum amplitude obtainable is imposed by our algorithm. All the injection parameters and their range are summarized in Table 5.2. The delay refers to the time from the start of the spectrogram where signals are injected. We set a low-frequency threshold at 30 Hz because of the high noise level of the Advanced LIGO detectors at lower frequencies [36].

[1]Note that the amplitude of chirp signals are first scaled down to 10^{-21} prior to being injected.

	Range of values
Duration	10 - 500 s
Delay	0 - 500 s
Frequency range	30 - 2000 Hz
Frequency evolution	linear, quadratic, logarithmic or hyperbolic
β parameter - Kaiser	1 - 4
Coherent amplitudes	2, 3, 4, 6, 8, 10, 15, 20, 25, 30

TABLE 5.2: Parameters used to inject chirp signals in the TF maps. All the parameters are uniformly drawn from their range of values.

Once trained, the network produces a map that indicates which pixels are susceptible to being part of a burst event. To reach that goal, we need a way to assess how good is the pixel selection during the training phase. A simple way of evaluating the progress made by the network is to compare its output with a target map. The target map can be easily built since we know where we inject the signals in the input TF image. Our target map is defined by first applying a threshold on the spectrogram pixels corresponding to the 99^{th} percentile of the values. Only the top 1% pixels are kept, the others are set to zero. Then, we select the pixels that belong to the footprint (or mask) of the signal in the input TF map. Finally, we normalize the target map so that the maximum value is 1. This procedure leads to a target map that follows the intensity evolution of the signals through the input map. An example of a spectrogram containing a chirp signal and its corresponding target map can be seen in Fig. 5.9.

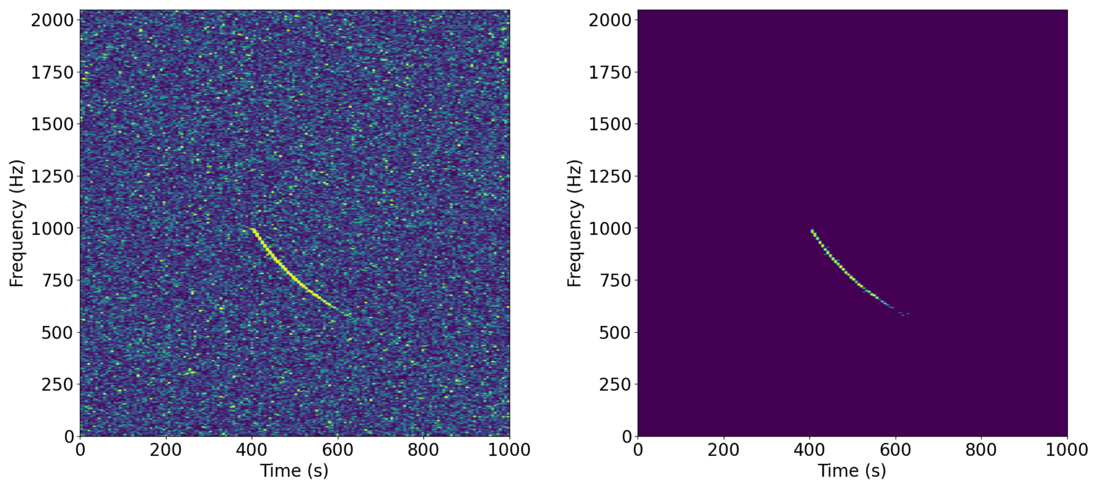

FIGURE 5.9: Background spectrogram in which a chirp signal is injected at 300 s (left) and its associated target map used for the training phase (right). The energy distribution of the chirp signal can also be seen in the target map, providing adequate samples for the training phase.

Note that we could have selected the pixels in the mask of the injected signal and set them all to 1. However, this would force the network to set output values to either 0 or 1, with no intermediate values. It can be risky to enforce extreme values since it would prevent us from ranking the detections based on their amplitudes. Moreover, it becomes difficult to discard false detections through simple postprocessing methods such as thresholding. We rather need a definition that can output both high and low values depending on the intensity of the signal found in the TF image.

5.2.3 Training procedure

Training a network means minimizing a loss function adapted to our objectives. Here, we want the output of ALBUS to approach as close as possible to a target. The loss should therefore account for the comparison between these two maps while producing a single value. The Mean Squared Error (MSE) loss is well suited to our problem. Considering the output of ALBUS, noted O, and a target map T associated with a particular input TF map, the MSE loss is defined as:

$$MSE = \frac{1}{2} \sum_{i,j} \left(T_{ij} - O_{ij}\right)^2 \tag{5.4}$$

where the sum is carried over the pixels (i, j) of the maps. As the MSE does not show any local minima, we are guaranteed to end up with well-behaved training. In the case of background TF maps, we simply set all the pixels in the target map to zero, i.e. T_{ij} is zero everywhere.

The training algorithms have all been coded with *Pytorch* [141]. The ADAM optimizer [105] has been chosen with a learning rate of 10^{-4}. We set the batch size to 20 where one half is taken from the background images and the other half from the injection images. The validation set is made of 10% of both datasets. Fig. 5.10 shows the training and validation losses as a function of the number of epochs. An epoch refers to the time needed by the network to see once each single image in the training set. The training loss decreases monotonically which suggests that the learning progresses evenly. The validation loss remains in close vicinity of the training loss ruling out any overfitting on the training data. We decided to stop the training after 30 epochs since both losses started to reach a plateau, indicating that no major improvements were made by the network.

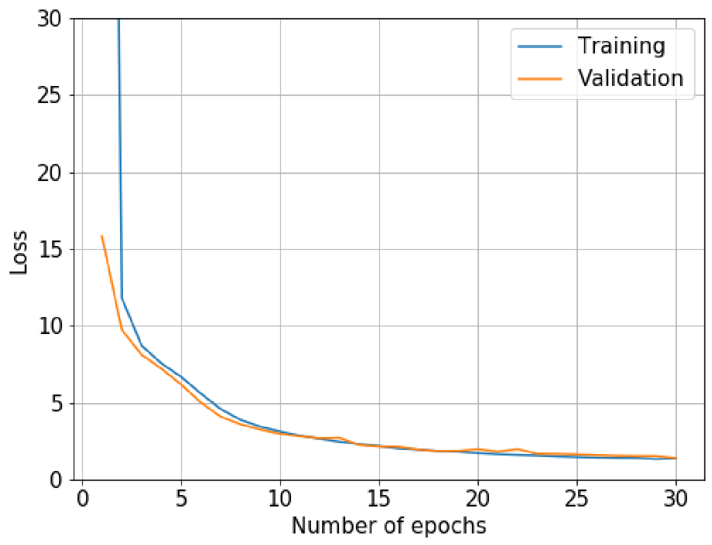

FIGURE 5.10: Training and validation losses for a 30-epoch training of ALBUS. The loss slowly reaches a plateau, indicating no further improvement of the network after 30 epochs.

5.2.4 Detection performance

Burst signals

Figure 5.11 shows the output of ALBUS for 4 different waveforms from the selected models in [55]. The simulated signals are well recognized and the variation of intensity in the input map is also seen in the localization map. An additional remark can be made concerning the lower right panel of Fig. 5.11, where a few pixels above the curve (from 950 to 1200 Hz at 500 seconds) are highlighted in the output map. Indeed, our network is not only looking at the pixels having a high value but also at the connectivity between these pixels. It then naturally prolongs the main structure to catch pixels following the general trend of the signal. This property has nonetheless some limits, as depicted in the lower left panel. Some pixels are indeed missed at the end of the *GRBplateau* waveform, probably because of a high density of high-value pixels in that region of the image.

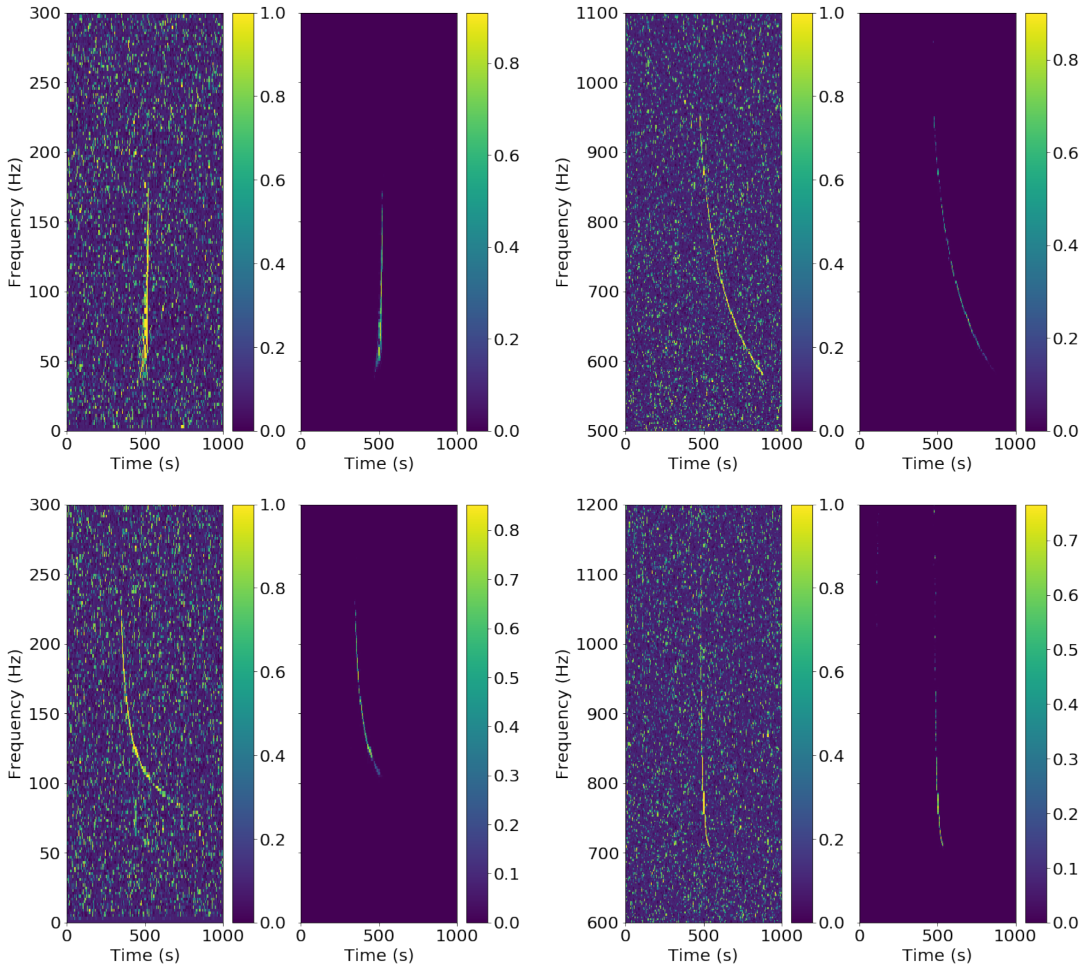

FIGURE 5.11: Detection performance on long-duration burst waveforms (top left: *ECBC-C* [83], top right: *Magnetar-F* [74], bottom left: *GRBplateau* [82] and bottom right: *ISCOchirp-B* [79]). The left image of each panel is the red channel of the input TF image and the right panel shows the output of ALBUS. Most of the pixels from the GW signals are detected when they stand above the noise level, with the exception of the harmonics of *ECBC-C* and the tail of *GRBplateau*.

Another point concerns the harmonics of some burst models. It is reasonable to foresee that ALBUS will not detect them as they have not been implemented in the training dataset. The upper left panel of Figure 5.11 displays this phenomenon to some extent. The harmonics appearing to the left of the rising chirp in the input TF image are not found in the output map. This effect is certainly due to the extrapolation capability mentioned above, for which ALBUS is looking for smoothly connected signals rather than thick signal footprints. The choice of not incorporating harmonics of chirp signals in the data has here minimal consequences since the main signal showing these harmonics is still detected.

Background

Even if our algorithm is good at recognizing chirps and burst waveforms, it will not be accepted as an official search engine if the false-alarm rate is large. It is therefore important to evaluate how ALBUS processes background TF maps. Figure 5.12 shows the output of ALBUS for 4 different background images. The output map shows values of the order of 0.01 except for the lower right panel. This trend is observed for all the processed spectrograms, with the exception of some isolated hot pixels as seen in the lower left panel. In any case, the highlighted pixels appear sparse and unconnected, confirming that our network is searching for connectivity among high-value pixels.

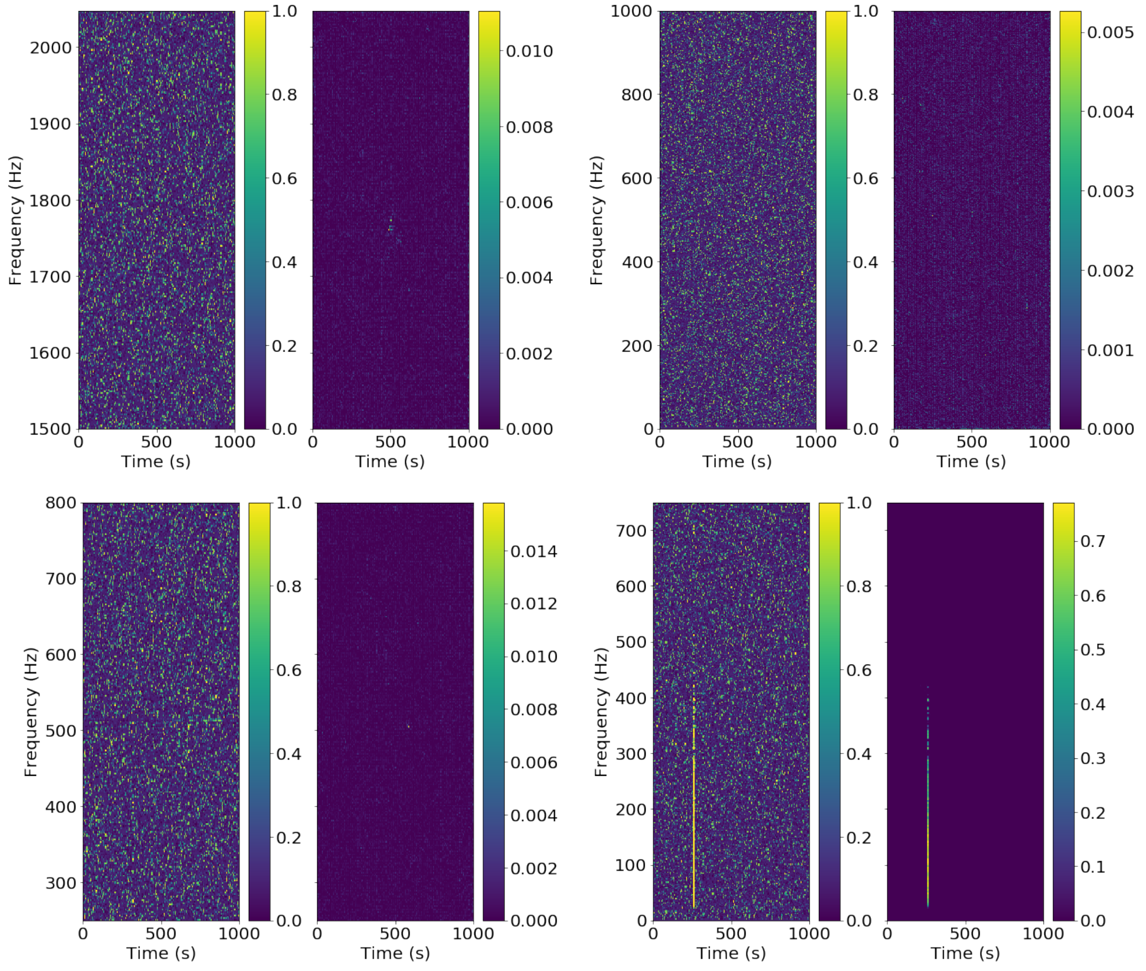

FIGURE 5.12: Detection performance on background spectrograms. The left image of each panel is the red channel of the input TF image and the right panel shows the output of ALBUS. ALBUS cleans out most of the background noise and only a couple of hot pixels appear in the output image. The lower right panel shows the sensitivity of our network to glitches. Although it has not been trained on such noise artifacts, ALBUS recognizes them precisely.

A further remark concerns transient noises or glitches. As most of the glitches last less than 6 seconds [36], they show up in our TF maps as straight vertical lines as seen in the lower right panel of Fig. 5.12. Our search in coincidence reduces their impact since a glitch from L1 data needs to fall in the same overlapping time bin as another glitch from H1 while showing a sufficiently high signal-to-noise ratio (SNR) and sharing some frequency bandwidth to show up in the coherence spectrogram. As we used random time slides to produce our background, this condition is rarely fulfilled, leaving only a couple of glitches over 4500 background TF maps. This small amount of glitches explains why ALBUS does not consider them as part of the background noise and actually detects them.

5.3 Going further with ALBUS

5.3.1 Time and frequency resolutions

As a tentative to improve ALBUS, we decided to switch from a frequency resolution of 2 Hz to 4 Hz. This is done in order to reduce the size of the images by a factor of 2, allowing us to set larger batch sizes while training our CNN. This choice has minor effects on the

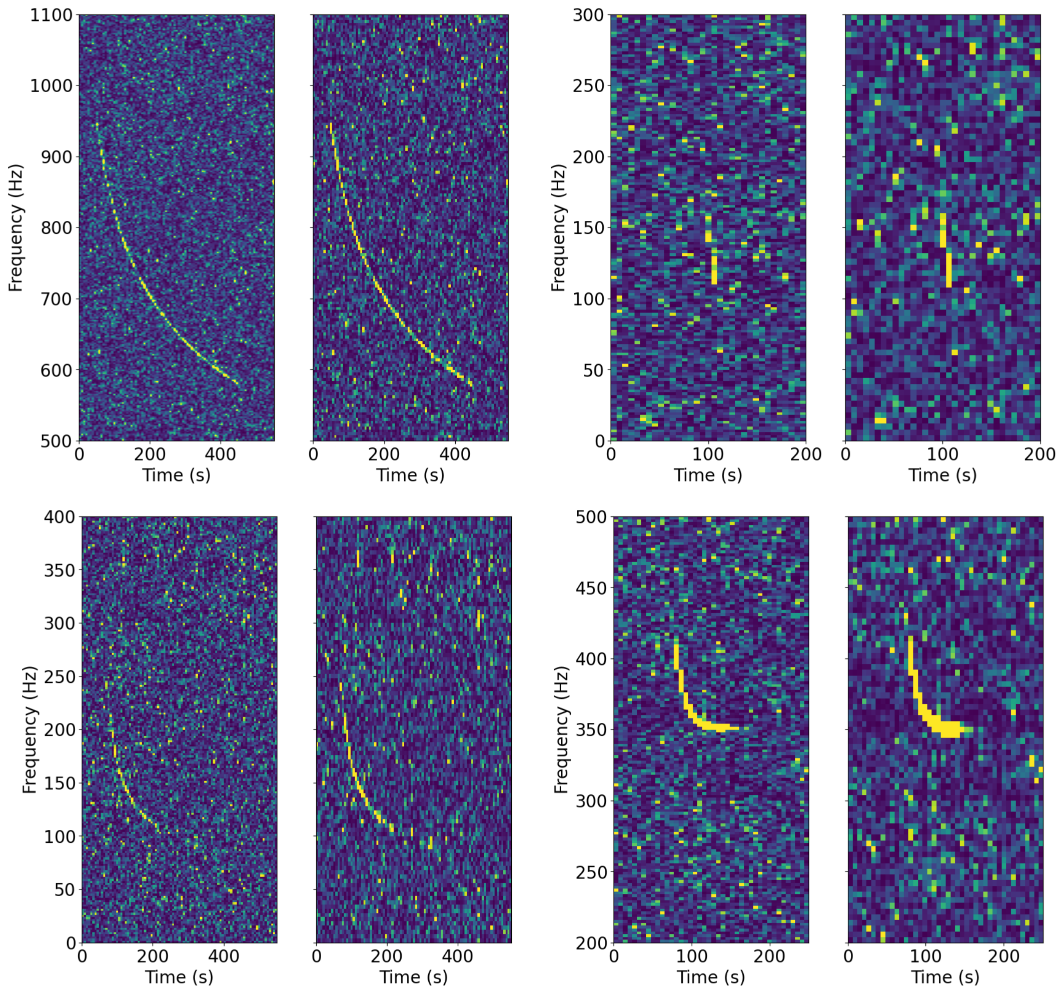

FIGURE 5.13: Coherence spectrograms produced with a frequency resolution of 2 Hz (left panel) and 4 Hz (right panel). Some long-duration burst models have been injected for comparison (top left: *Magnetar-F* [74], top right: *ADI-B* [80], bottom left: *GRBplateau* [82] and bottom right: *ISCOchirp-B* [79]).

signals injected in TF maps, as seen in Fig. 5.13. As a consequence of a smaller number of pixels to form their TF morphology, burst models show more angular and straight footprints. However, larger pixels cause an increase in the coherence of burst signals versus the coherence of the noise, leading to an improved SNR. The GW signals are therefore more easily visible in the TF maps for the same value of the h_{rss}.

The coherent amplitude of the signals is nonetheless barely affected by the change in frequency resolution. As seen in Fig. 5.14, the coherent amplitude roughly reaches the same values for a given h_{rss}, compared to the former resolution (see right panel of Fig. 5.5). We therefore keep the same range of values for any future chirp dataset.

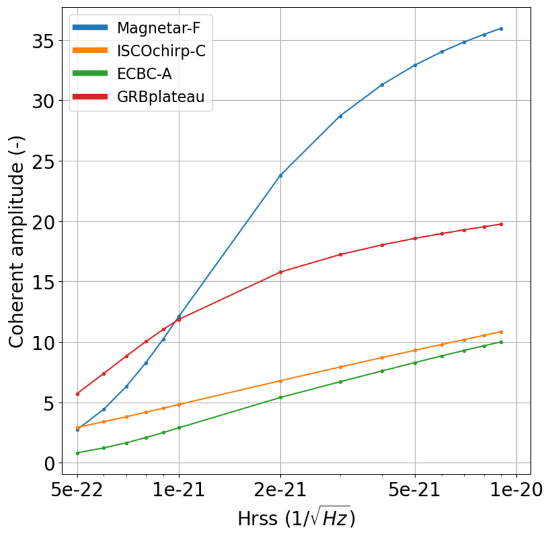

FIGURE 5.14: Coherent amplitude as a function of the h_{rss} for a frequency resolution of 4 Hz. As a comparison, 4 long-duration models have been injected into the same O3a background noise. The GPS time at the start of the H1 and L1 data is 1241246771 and 1252436251 respectively.

5.3.2 Discriminating glitches from bursts

In the last section, we trained ALBUS with chirp signals having various morphologies in the TF plane. We show that this methodology is adapted to detect long-duration burst models. However, ALBUS can also recover glitches fairly and a human inspection is needed to discriminate them from burst signals. In coincidence searches, this becomes rapidly intractable in view of the thousands of glitches found out of millions of background maps. We could design a strategy that consists in eliminating any vertical cluster of pixels in the output map. Nonetheless, some burst waveforms like the *ISCOchirp* [79] show very peaking footprints and we run the risk of discarding some parts of these signals.

Rather we can train ALBUS to discriminate glitches from burst signals. This can be achieved by adapting ALBUS to produce 2 output images: one that will contain the pixels that belong to burst signals and another which gathers the pixels from glitches. In the following, we will refer to these two outputs as Anomaly and Glitch maps respectively. To train ALBUS on glitches, we need to build a third dataset containing several thousands of glitches. As we barely get a few tens of them by using random time slides, we would need to generate millions of background images and discard most of them. Rather, we could inject fake realistic glitches with tools such as *gengli* [56, 60]. However, *gengli* can only generate *blip* glitches for the moment. Blips are one of the 23 classes that have been characterized by Gravity Spy [58, 142]. As they have a limited frequency bandwidth ($\leq$ 256 Hz) [36, 143], they would limit the bandwidth sensitivity of ALBUS if used exclusively in our dataset.

We thus have to rely on the glitches detected so far to constitute the training set. In O3a, a lot of them have been recorded by Gravity Spy [144]. As we know where to find them in the data, we can force two glitches to fall in the same overlapping time bin. For this, we load the data containing the glitches in H1 and L1 and shift them so that the latter fall into the same time bin. In this way, we maximize the probability of finding glitches in the coherence maps. To account for variability in the footprints of the correlated glitches, we select 17 glitch classes with SNR ranging from 20 to 10000 in both H1 and L1 data. These pre-selected glitches will be used to build our glitch dataset. Table 5.3 summarizes the details behind the glitch selection. As in the case of chirp injection, the delay refers to the time from the start of the spectrogram where the glitches lie.

Glitch classes	Blip, Low Frequency Burst, Scattered Light, Tomte, Whistle, Extremely Loud, Koi Fish, Power Line, Violin Mode, Air Compressor, Repeating Blips, 1400Ripples, 1080Line, Helix, Paired Doves, Scratchy, None of the Above
SNR ranges	20-30, 30-40, 40-50, 50-100, 100-150, 150-200, 200-300, 300-500, 500-10000
N° **per SNR range**	≤ 30
Delay	[6, 994] s
Total	**H1:** 1320 **L1:** 1560

TABLE 5.3: Summary of the attributes of the glitches selected from H1 and L1.

The procedure for building a glitch dataset is therefore straightforward. First, we randomly choose a glitch among the pre-selected glitches both in H1 and L1. Then, we produce the coherence spectrogram and process it with the former version of ALBUS. As it is sensitive to glitches, the output map contains the footprint of the glitches. Therefore, we only have to check the output of ALBUS to detect the presence of a glitch in the initial TF map. To quantify the strength of the anomalies found in the original spectrogram, we introduce the anomaly score (AS), defined as:

$$AS = \sum_{i,j} O_{i,j} \ \text{if} \, O_{i,j} > 0.5 \max(O) \tag{5.5}$$

where O is the output map of ALBUS and i and j indicate the time and frequency dimensions. The anomaly score can be thought of as the sum over the pixels remaining after thresholding the values in the output map. This threshold has been chosen to exclude all the values close to zero, as they are quite numerous given the size of the TF maps and can have an impact on the final score. The anomaly score can also be used to rank the detected signals as seen in Figure 5.15 where an extended glitch shows a higher score compared to a glitch that spans a smaller frequency bandwidth.

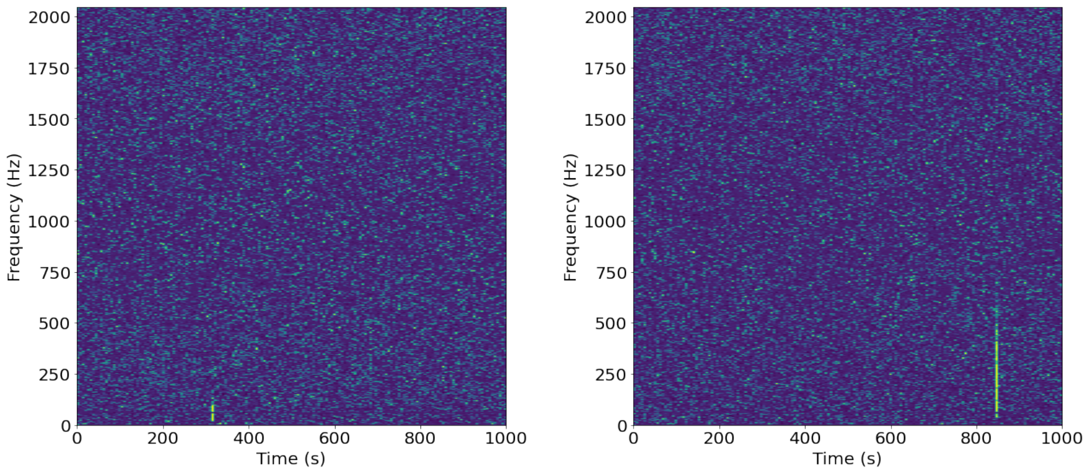

FIGURE 5.15: Examples of correlated glitches with different anomaly scores. The left panel shows a glitch for which the AS is 7 while the glitch in the right panel has an AS of 43. The frequency resolution of the spectrograms is 4 Hz with time bins of 6 s.

To determine the minimum AS that effectively corresponds to the presence of a glitch in the input TF image, we processed 10000 spectrograms. A histogram of the resulting anomaly scores can be seen in Fig. 5.16. Almost all the background maps seem to show an anomaly score below 2, with only 17 of them above that value. After visual inspection, all the background images with scores above 4 show a correlated glitch. We thus set the threshold to confirm the presence of a correlated glitch to 4. Every background TF map showing an AS above 4 will therefore be selected as a sample of the glitch dataset.

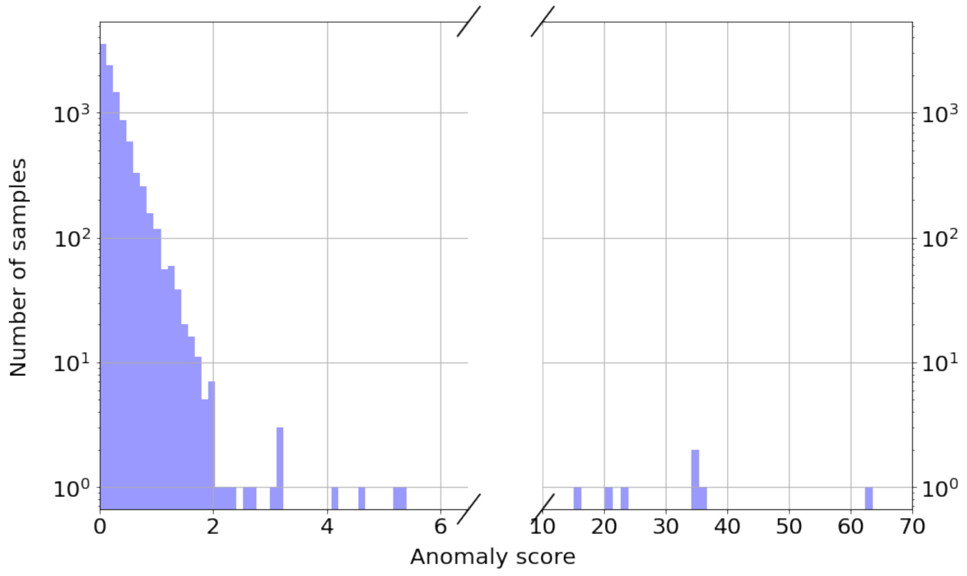

FIGURE 5.16: Histogram of anomaly scores for 10000 background TF maps. The majority of scores lie below 2, confirming that ALBUS has not detected any significant pattern. All the TF images showing an Anomaly score above 4 contain a glitch.

5.3.3 Simulating harmonics

In order to further enhance the capabilities of ALBUS, we implement a method to simulate waveform harmonics found in burst models like *ISCOchirp* and *ECBC*. From a former analysis, we established that our network can miss loud signals when they show some harmonics. This is probably due to their TF footprints which appear thicker in the input spectrograms. As we exclusively inject frequency-swept cosines in TF images, our initial dataset did not contain any thick signal. In turn, as it has never seen such TF structures, ALBUS does not highlight the relevant pixels.

To overcome this limitation, we propose a method to simulate harmonics in chirp signals. Starting from an initial main chirp, the method consists in successively injecting the main signal with slightly modified parameters and reduced amplitude. In this way, the harmonics share the common structure of the main signal but remain dimmed with respect to the latter. As an example, let us consider Fig. 5.17. As the signal is chirping up, we could simply adapt the starting frequency of the harmonics to produce footprints similar to *ECBC* signals (see Fig. 3.3). Then, the frequency spacing between the tails of the harmonics becomes a free parameter. In the same way, the number of harmonics found above and below the main signal can be tuned to generate chirps with very different TF footprints. Note that, for the sake of simplicity, we will consider the same number of harmonics above and below the injected chirp. When referring to a number of harmonics of 2, we therefore signify that 2 of them are found on either side of the main signal.

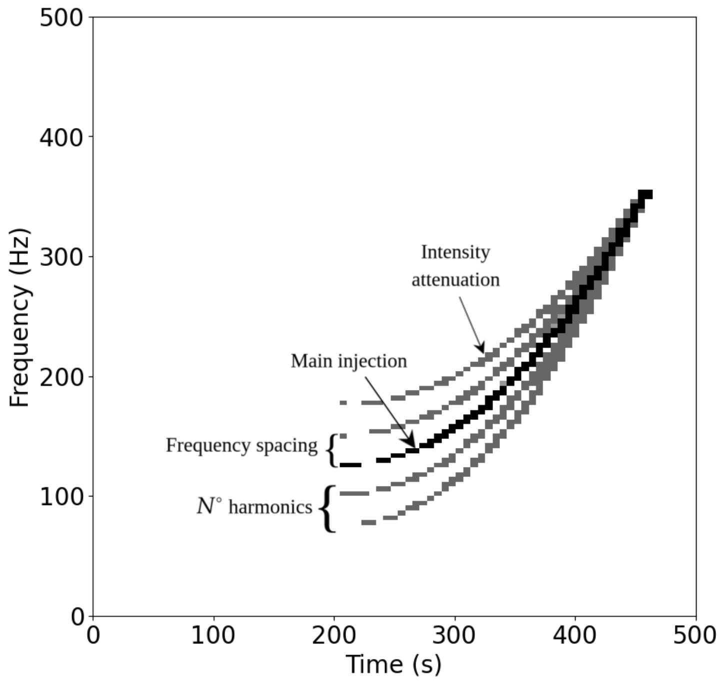

FIGURE 5.17: Illustration of the free parameters for a chirp signal including harmonics. The number of harmonics as well as their frequency spacing and their intensity attenuation with respect to the main signal can be adapted.

As the chirp injection procedure still relies on the h_{rss} value[2], one could simply dim the harmonics by using an attenuation coefficient. For example, we can inject the main chirp with $h_{rss} = 10^{-21}$ while the harmonics' amplitude amounts to $7\,10^{-22}$. In this particular

[2]Even if the decision criterion is based on the coherent amplitude, chirps are injected with the h_{rss}.

case, the attenuation coefficient is 0.7.

Both the frequency spacing, the number of harmonics and the attenuation coefficient are 3 new free parameters in our chirp injection procedure. Fig. 5.17 illustrates how they affect the generated harmonics. The frequency spacing is chosen to be constant and in the range $[10, 30]$ Hz. The number of harmonics on either side of the main signal is either chosen to be 1, 2 or 3 to mimic the behavior of burst models like *ISCOchirp*. After visual inspection, it seems that attenuation coefficients from 0.5 to 0.8 are adequate to produce chirp signals whose harmonics appear at high amplitudes.

5.3.4 Generation of the new training dataset

In order to improve ALBUS by adding glitch discrimination and harmonics identification capabilities, we need to build a new training dataset. In addition to the background and chirp data, we build a third dataset containing exclusively correlated glitches. As nothing prevents a glitch to appear in the same TF map as a burst signal, we also need to consider the case where a glitch and a chirp are both present in the spectrograms. We then end up with 4 different sets of TF images, referred to as background, chirp, glitch and combined datasets.

All the input TF maps now have 2 target maps: one for the chirps and the other for the glitches. The glitch target map consists of the output of ALBUS prior to the injection of any chirp signal. Therefore, TF images from the background and chirp datasets show a null glitch target map. Both the chirp and glitch target maps are shown in Fig. 5.18 in the case where both can be seen in the input spectrogram. ALBUS will be trained to highlight excess of power and discriminate whether it corresponds to a chirp or a glitch.

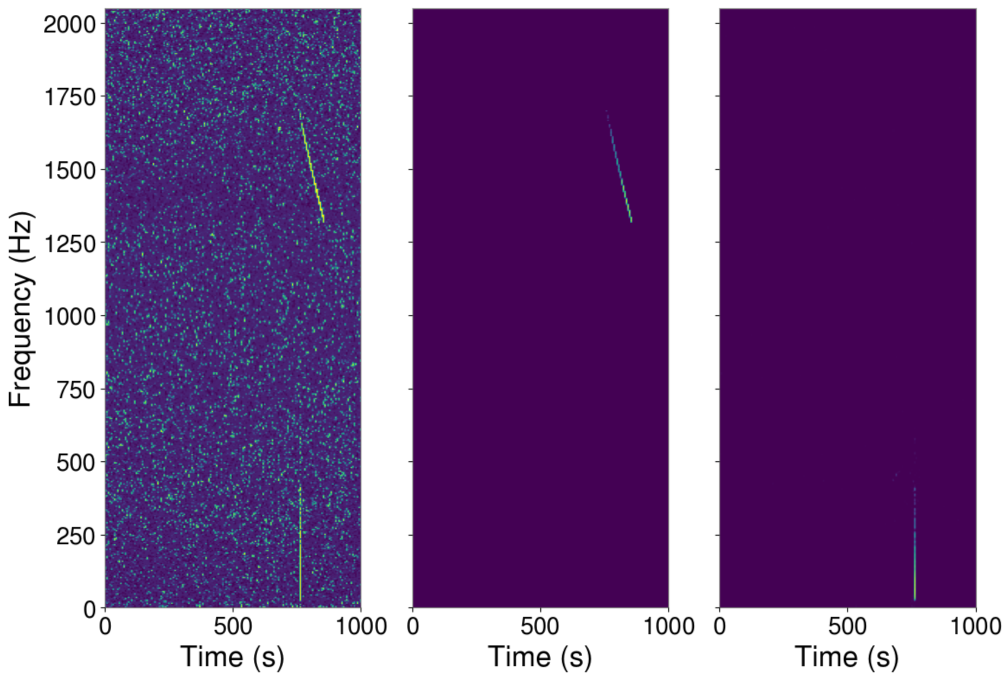

FIGURE 5.18: Coherence spectrogram containing both a chirp signal and a correlated glitch. The middle and right panels show the corresponding chirp and glitch target maps, used in the training phase.

In a former coherent search, we analyzed 10 years of background (≈ 300000 TF maps) with ALBUS. The output maps were then processed with Pyxel which clusters the remaining pixels to form triggers. More details about the processing steps will be given in Chapter 6. As a sanity check, we overlay the centroid of every trigger in a single TF map. Fig. 5.19 illustrates the results. Most of the triggers are found in the first half of the map, between 0 and 600 seconds. As a consequence of setting the injection delays in the interval $[0, 500]$ s, ALBUS ends up being more sensitive on that side of the map. This is highly problematic as it drops the overall sensitivity of our network. Even if this choice was made so that any signal is entirely contained in the map, we adapt the delays to the interval $[0, 950]$ s. We therefore implement a method to cut off the chirp signals that exceed the limits of the TF maps.

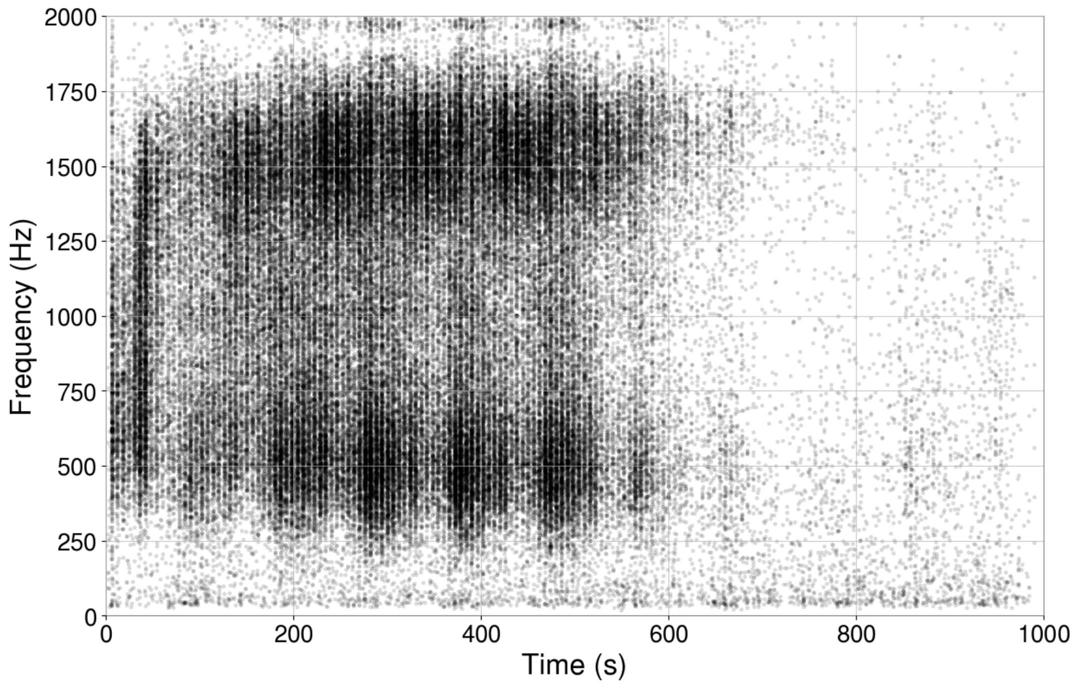

FIGURE 5.19: Trigger distribution for 10 years of background. The trigger identification is conducted on the output Anomaly map of ALBUS. The remaining high-value pixels are first clustered via Pyxel, which estimates their bandwidth and duration. Their centroid is finally derived and plotted as a black dot in the above graph. As a consequence of the injection delays being below 500 s, ALBUS ends up being more sensitive on the left-hand side of the input TF image.

We further adjust the coherent amplitudes to add more variability to the chirp dataset. Chirps are ultimately injected with 15 different intensities in coherence spectrograms. Another change concerns the bandwidth of these chirps, which is now limited to 1000 Hz. This is done because very extended chirps require only a slight excess of power to give a low coherent amplitude (≤ 4). They might still be mostly hidden by the local noise along their TF footprint. Limiting the bandwidth of chirp signals helps to solve this problem. It is important to note that this will not necessarily be harmful to the performances of ALBUS. In its first version, ALBUS has shown good extrapolation capabilities with the ability to detect glitches while he was not trained on them. We therefore expect ALBUS to be sensitive to very extended signals even if they do not show up in the training dataset. Table 5.4 summarizes the latest injection parameters.

	Range of values
Duration	10 - 500 s
Delay	0 - 950 s
Frequency range	30 - 2000 Hz
Max. Bandwidth	1000 Hz
Frequency evolution	linear, quadratic, logarithmic or hyperbolic
β **parameter - Kaiser**	1 - 4
Coherent amplitudes	2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 18, 20, 25, 30
N° **harmonics**	1, 2, 3
Frequency spacing	10, 15, 20, 25, 30 Hz
Attenuation coeff.	0.5, 0.6, 0.7, 0.8

TABLE 5.4: Final chirp injection parameters used to optimize ALBUS. All the parameters are uniformly drawn from their range of values.

5.3.5 Results

Using time slides, we generate 15000 background images (corresponding to 6 months of data) as well as 15000 spectrograms in which we inject a chirp signal (1000 per coherent amplitude). Among these chirps, roughly 3300 show harmonics while the rest consists of simple chirps. We also produce 1 million background spectrograms with adapted time slides to select glitches with the first version of ALBUS. Out of this total, we end up with 27994 TF maps containing glitches. In comparison with random time-slides where we found 11 glitches out of 10000 maps, our method yields roughly 28 times more glitches per generated spectrogram. We further split this amount into two subsets; one part constitutes our glitch dataset (15000 maps) while the other part is used to build the combined dataset (12994 maps). In total, we generate 15000 images including both a glitch and a chirp signal. In this mixed dataset, about 3800 maps include chirp signals with harmonics.

We split our dataset into training, validation and testing sets with proportions $60\% - 6.66\% - 33.33\%$ (i.e. $9000 - 1000 - 5000$ images from each dataset). The testing set will be used to examine the performances of ALBUS on a collection of TF images from each dataset once the training is completed. We keep the ADAM optimizer with a learning rate of $5\,10^{-5}$ and set the batch size to 32. Note that each batch is a collection of TF images chosen randomly over our 4 datasets. Fig. 5.20 illustrates the behavior of both the training and validation losses. They jointly decrease until progressively reaching a plateau. We stop the training after 20 epochs since no further improvement is observed.

In terms of performance, ALBUS is now able to discriminate glitches from chirps even when the latter have a short duration, as seen in Fig. 5.21. Both the chirp and glitch footprints are well-highlighted with respect to the surrounding background. It is also important to note that both chirps and glitches are recognized even if they show discontinuous parts. This property allows us to select only the pixels that stand out of the noise or to discard glitches where they actually contaminate the input TF map.

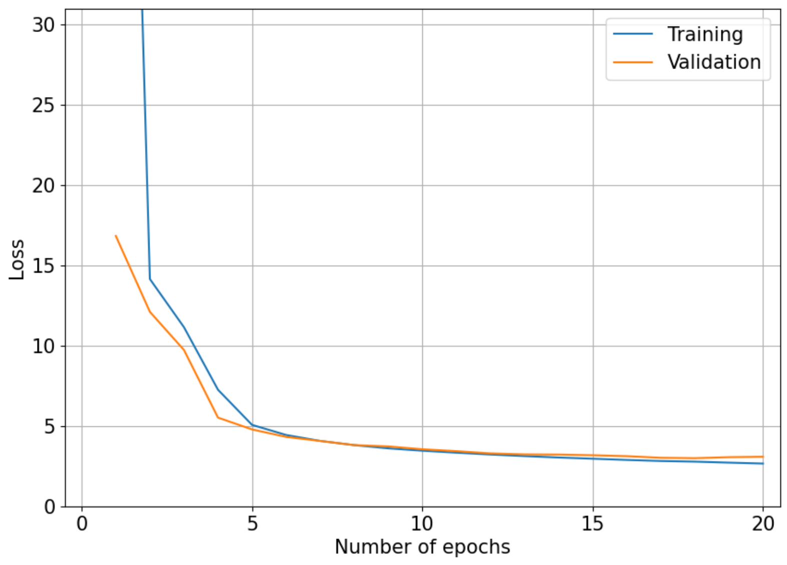

FIGURE 5.20: Training and validation losses for a 20-epoch training of ALBUS on the new dataset. As the validation loss starts to reach a plateau, we stop the training after only 20 epochs.

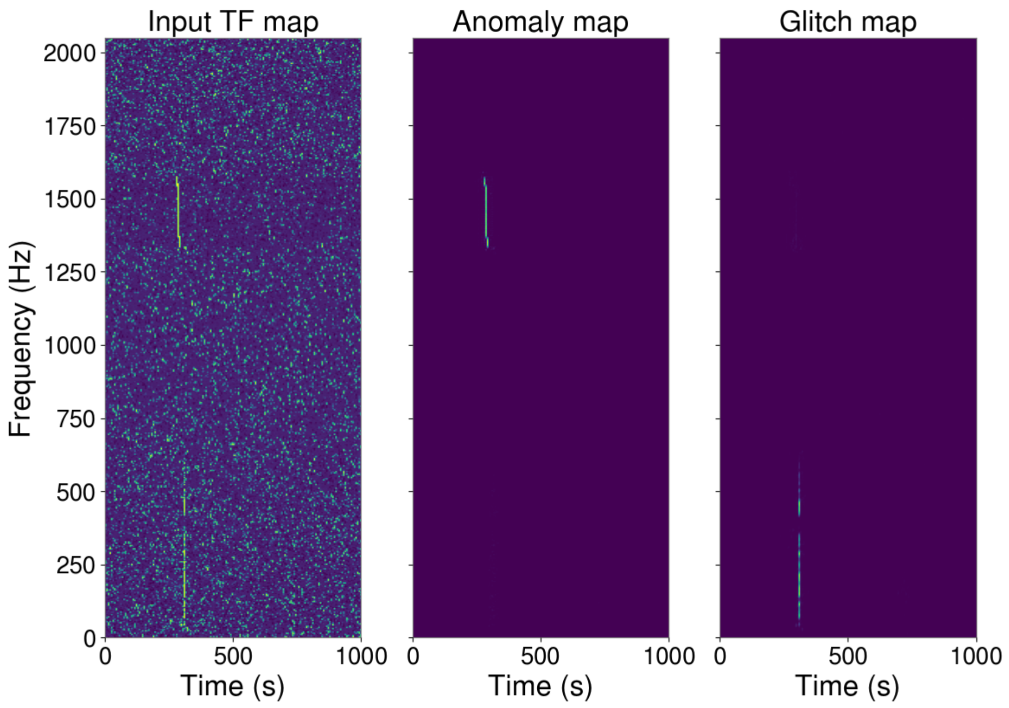

FIGURE 5.21: Coherence spectrogram containing both a chirp signal and a correlated glitch. The middle and right panels depict the corresponding Anomaly and Glitch maps, produced by ALBUS. Note how ALBUS classifies the chirp and the glitch in the correct output image.

A further remark can be raised regarding overlapping signals. Indeed, glitches can happen at the same time as a GW signal, as it was the case for GW170817 [26]. It is therefore fundamental for ALBUS to discern both types of events when their TF windows overlap. Fig. 5.22 shows that ALBUS can isolate the contribution from both the chirp and the glitch, facilitating the trigger identification.

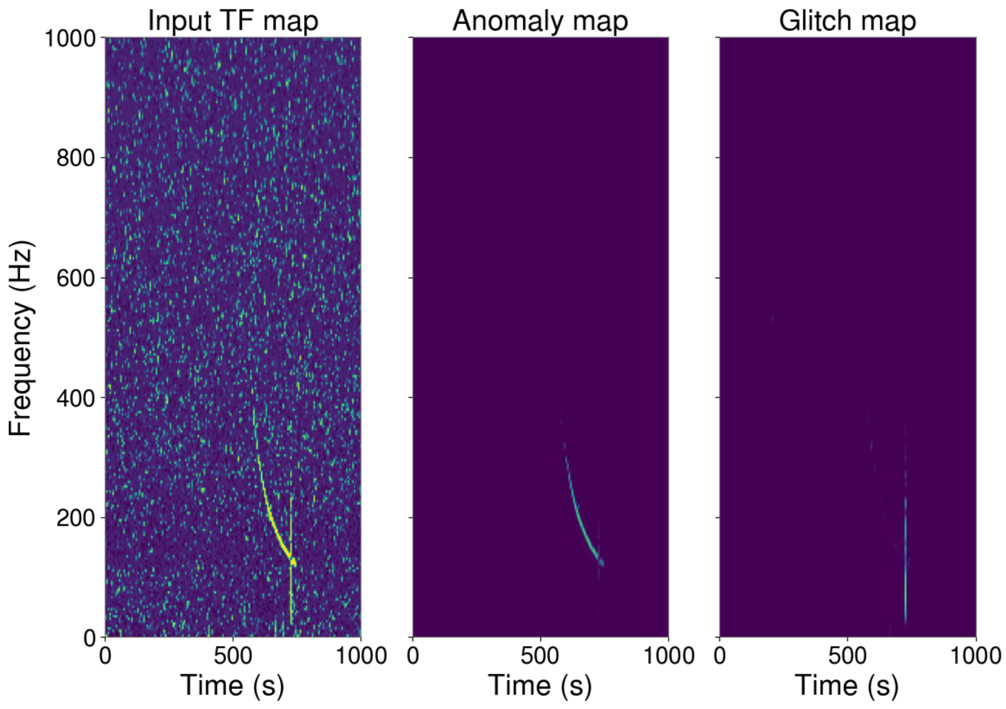

FIGURE 5.22: Coherence spectrogram where a chirp signal overlaps with a correlated glitch. The middle and right panels show the outputs produced by ALBUS, namely the Anomaly and Glitch maps. ALBUS is able to discriminate chirps from glitches even when they overlap in the TF plane.

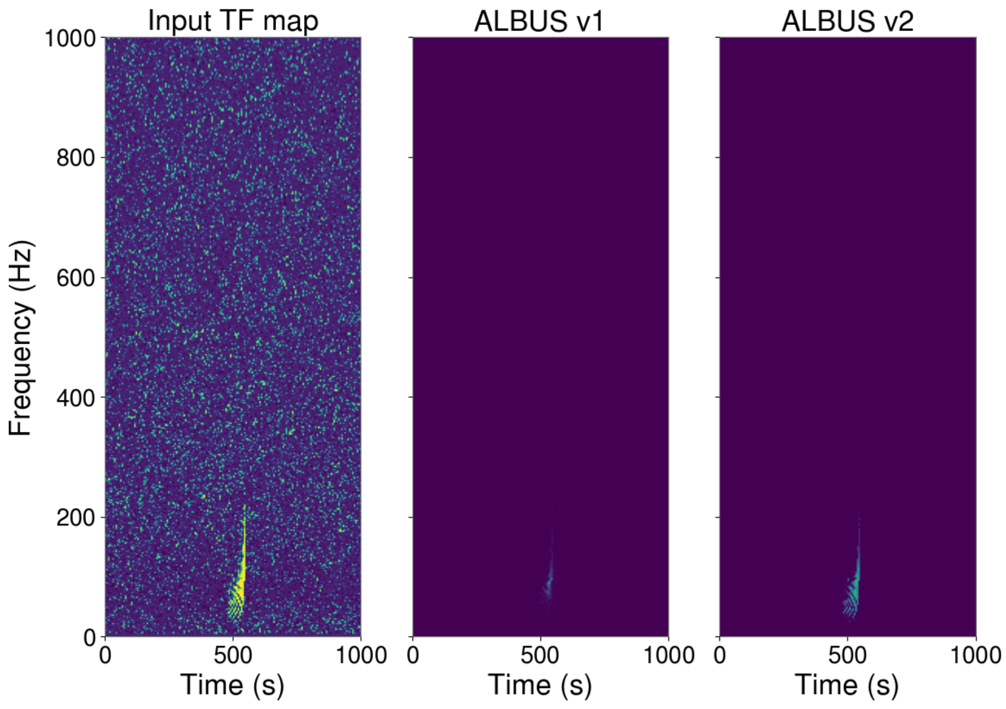

FIGURE 5.23: Coherence spectrogram in which a *ECBC-C* model has been injected. For comparison, the Anomaly map of both the first and second versions of ALBUS are shown in the middle and right panels respectively. The old version of ALBUS is not able to detect signals with a thicker footprint while this is now possible with its newly-trained version.

Another improvement concerns the harmonics of burst signals. As we have introduced a strategy to simulate harmonics in the training dataset, ALBUS is now able to recognize them in burst models. Fig. 5.23 illustrates the Anomaly maps of both the old and the new version of ALBUS. The former version struggles to identify multiple curves in a narrow TF space. This ends up reducing the number of detected pixels and can lead to a drop in the sensitivity to burst signals like *ISCOchirp* and *ECBC*. The new version of ALBUS accurately highlights the full signal, enabling better detection performance across a wide range of waveforms.

In order to analyze the performances of ALBUS on the 4 different datasets, we need a statistic that reflects what has been highlighted in both the Anomaly and Glitch maps. For this, we can use the definition of the anomaly score and apply it likewise to the Glitch map. We end up with two scores, defined by:

$$AS = \sum_{i,j} A_{i,j} \text{ if } A_{i,j} > 0.5 \max(A) \tag{5.6}$$

$$GS = \sum_{i,j} G_{i,j} \text{ if } G_{i,j} > 0.5 \max(G) \tag{5.7}$$

where the sum is performed both on the time and frequency dimensions and A and G stand for the Anomaly and Glitch maps respectively. Note that this time, the newly trained

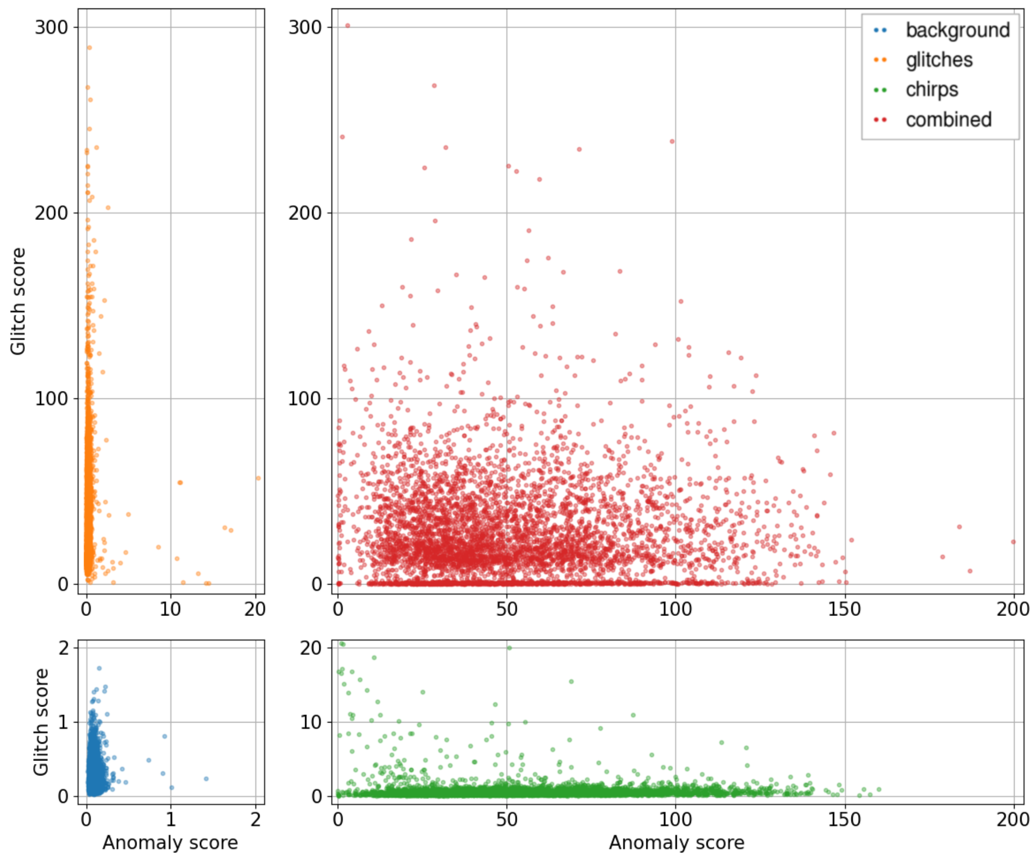

FIGURE 5.24: Anomaly and glitch scores obtained for the testing set. 5000 TF images from each of the 4 datasets have been processed (top left: glitches, top right: combined, bottom left: background, bottom right: chirps).

ALBUS is used to evaluate both scores from the TF maps. Fig. 5.24 shows the scores obtained for each of the 20000 TF maps (4x5000) in the testing set. Although the scores do not correspond to the actual triggers found in the map, they give a good indication of what ALBUS detects for each dataset. Note that the scale of both the Anomaly and Glitch scores is different for the background images since they show very low scores. In the same way, the extent of the chirp and glitch plots have been adapted for illustration purposes.

For the background images, both the Anomaly and Glitch scores are lower than 2, leaving no doubt that nothing is found in those maps. For the chirp dataset, the distribution of the TF maps is very peaked at low glitch scores. Nonetheless, some chirp TF maps can show glitch scores up to 30. This primarily happens when the chirp signals have a short duration (<15 seconds) and broad frequency bandwidth. Fig 5.25 shows the distribution of anomaly and glitch scores as a function of the chirp duration. As it can be seen, most of the chirps featuring a glitch score greater than 5 are short-duration chirps. Since they span only 2 or 3 time bins, a part of them can then be mistaken as a glitch, as illustrated in Fig. 5.26. Due to random background noise, the signal is divided into three parts for which the longer vertical footprint is considered as a glitch by ALBUS. Such misidentifications happen only at low coherent amplitudes and barely impact the sensitivity of ALBUS since a part of the signal is still found in the Anomaly map.

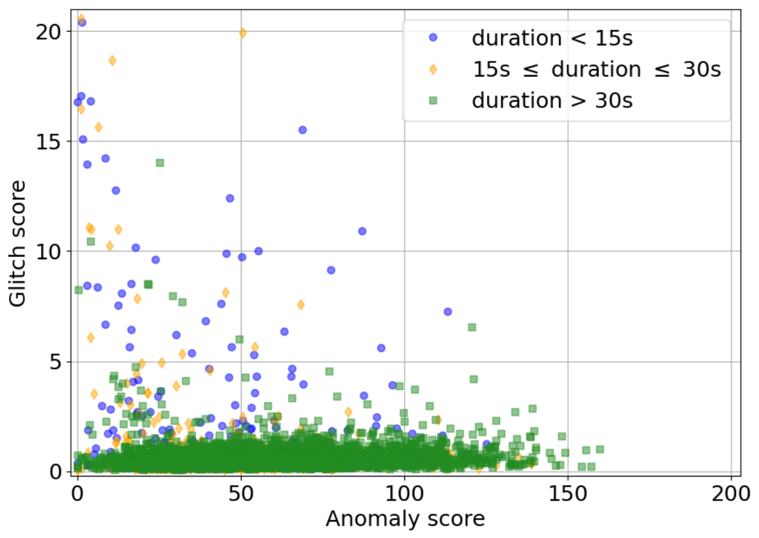

FIGURE 5.25: Anomaly and glitch scores obtained for the 5000 images in the chirp testing set as a function of the duration of the chirps. Most of the chirps showing a glitch score over 5 last less than 30 seconds.

For what concerns glitches, they show a wide range of glitch scores associated with very low anomaly scores. However, exceptional glitches can introduce noise artifacts that mimic the behavior of burst signals. As most of the glitches last less than 6 seconds, they fall into a single time bin in our TF maps, giving them a sharp vertical footprint. ALBUS has probably also learned this peculiar feature during the training. However, rare longer glitches show a thicker footprint that may therefore not be fully recognized as glitches, as in Fig. 5.27. A couple of these glitches were found in the testing set, leading to anomaly scores between 5 and 20. The last dataset, for which a chirp and a glitch are found in the TF maps, shows a wide range of anomaly and glitch scores, reflecting the variability introduced in the dataset generation.

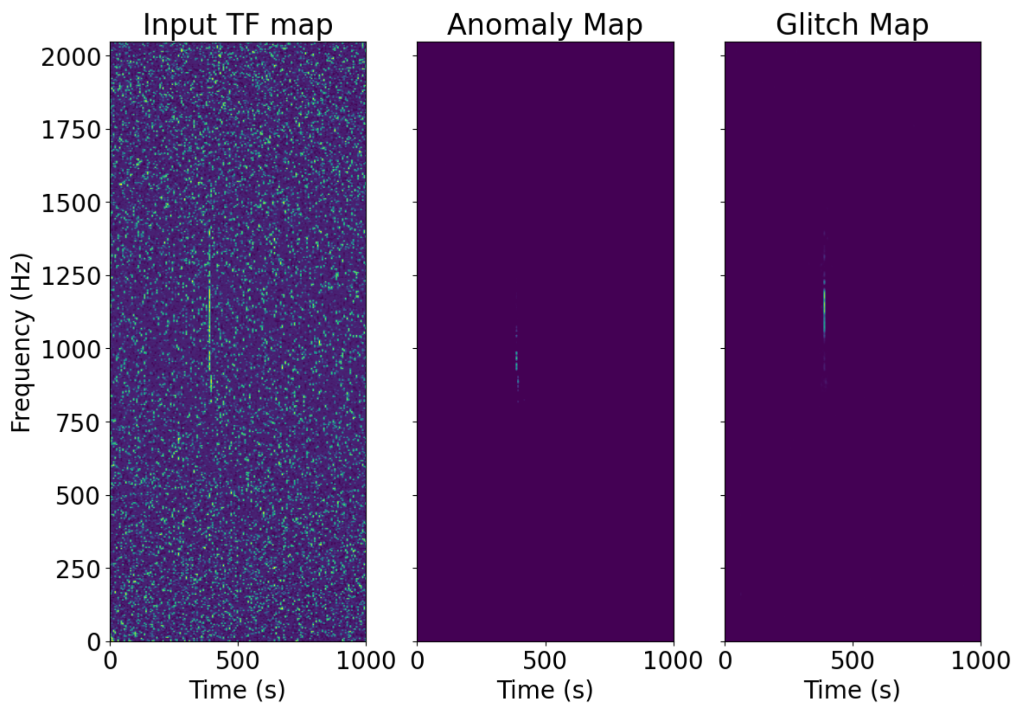

FIGURE 5.26: Coherence spectrogram from the chirp testing set. The middle and right panels show the outputs produced by ALBUS, namely the Anomaly and Glitch maps. As the chirp signal is divided into three parts, ALBUS identifies the vertical track as being a distinct glitch while it correctly classifies the two other parts.

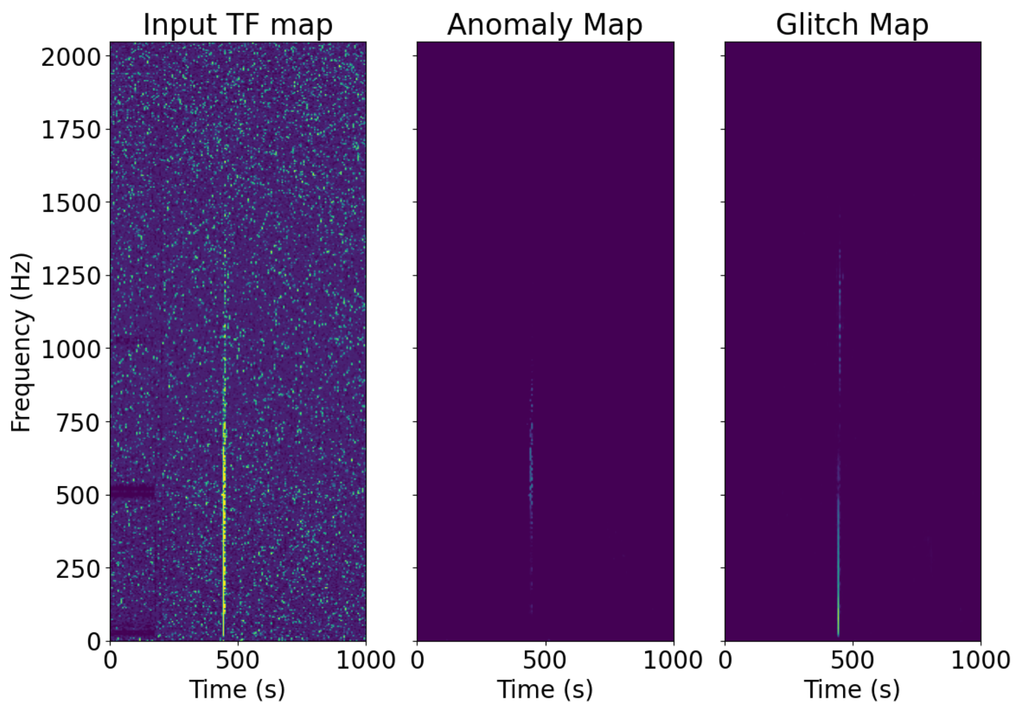

FIGURE 5.27: Coherence spectrogram from the glitch testing set. The middle and right panels show the outputs produced by ALBUS, namely the Anomaly and Glitch maps. A part of the glitch footprint is classified as a chirp, probably because it spreads over more than one time bin.

Now that we have trained ALBUS and improved its detection capabilities, we need to run an extended search and compare our performances to other pipelines. For this, we should first include ALBUS into Pyxel and determine which detection statistic achieves optimal results. The methods developed to achieve these objectives are described and analyzed in Chapter 6.

Chapter 6

Towards a new burst pipeline

6.1 Integrating ALBUS into Pyxel

6.1.1 Pipeline structure and methods

The methods developed in Chapter 5 built up into a new machine-learning-based approach to the search for long duration bursts. However, as with all detection engines, ALBUS has to be included in a pipeline to be used effectively during a search. To this aim, I built ALBUS so that it matches the existing workflow of Pyxel. Previously, Pyxel processed TF maps by using thresholding, morphological operations and clustering methods. Since ALBUS produces outputs with the same size as the input TF map, these techniques can still be applied with minimal adjustments. In this context, ALBUS acts as a noise rejection engine that provides clean TF maps to Pyxel.

The procedure to run ALBUS on data from H1 and L1 is as follows. First, the data from both detectors are whitened via their respective PSDs. Then, Pyxel generates a coherence spectrogram of the data via a GWpy [145] routine and normalizes it across all frequencies for the entire duration of the TF map. ALBUS then processes the resulting TF map to produce two outputs, the Anomaly and Glitch maps. The last steps consist in clustering the pixels remaining in the Anomaly map to form triggers and compute relevant statistics.

The clustering method consists in first applying a Yen's threshold [146] on the Anomaly map. Yen's threshold is an adaptive local method that determines the best threshold according to the noise in the map. It therefore guarantees to select adequate pixels without setting a global threshold for all the processed TF maps. As seen in the previous chapter, most of the noise is cleared out by ALBUS which makes Yen's threshold a very effective method to select the pixels with the highest values. Then, the Euclidean Distance Transform (EDT) is performed on the resulting image. This transform replaces each background pixel by its shortest distance to the closest trigger highlighted via Yen's threshold. Applying a threshold on the Euclidean distance allows the clustering of unconnected pixels belonging to the same event. In this way, GW events can be recovered via a unique trigger showing a higher detection statistic than multiple shorter triggers. In this work, we consider that pixels showing an EDT smaller than 5 pixels are part of the same triggers, which allows to connect individual triggers 10 pixels away. We empirically notice that a lower threshold does not connect parts of triggers together while a higher threshold clusters noise speckles. We end up with N masks revealing the N triggers found in the Anomaly map. Fig. 6.1 illustrates the output of the different processing steps from the input TF map to the trigger detection. The final triggers (in pink) are superimposed on the input image.

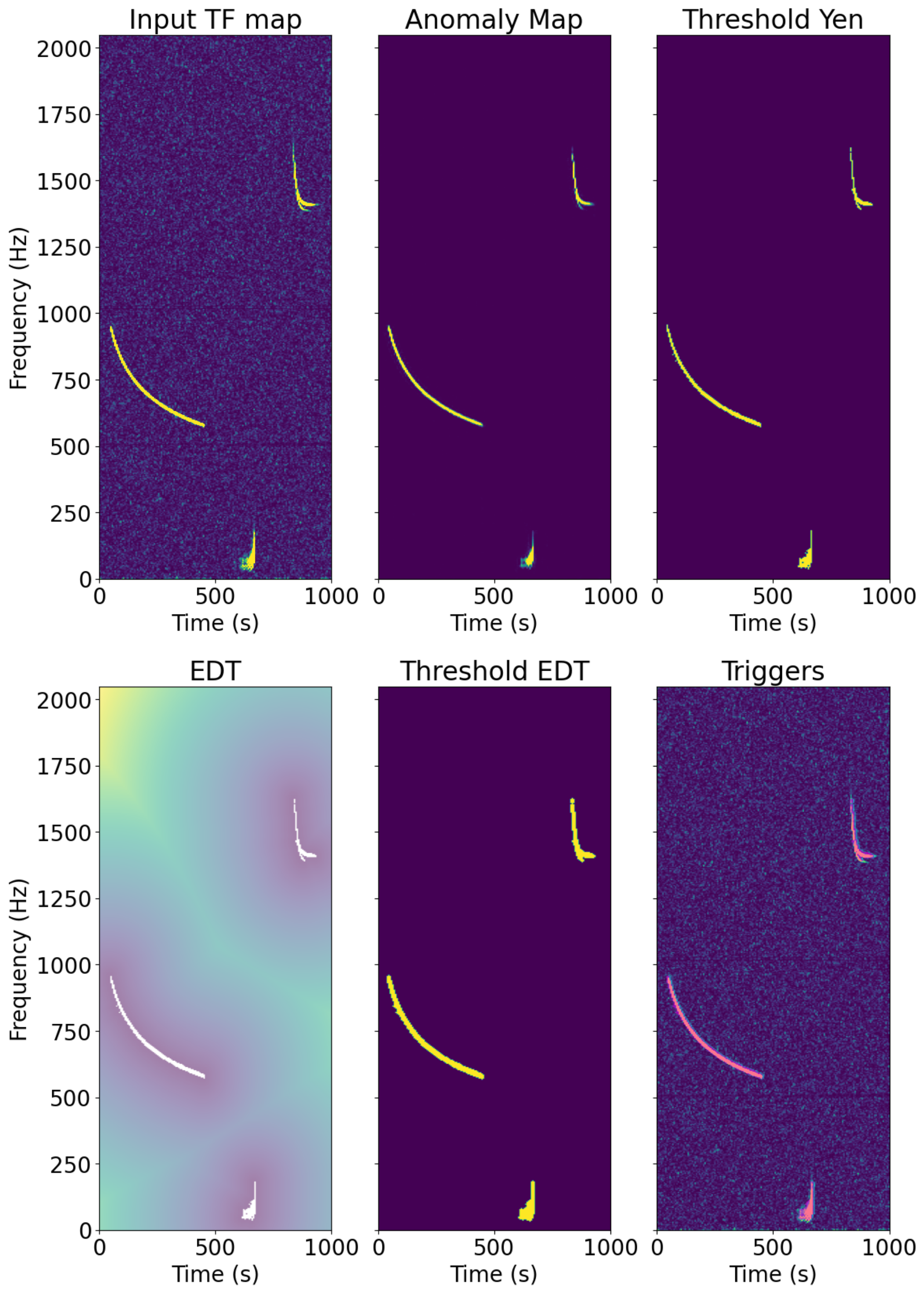

FIGURE 6.1: Illustration of the successive steps to achieve trigger identification. Each panel is the output of the method referred to as its subtitle. From top left to bottom right: input TF map, Anomaly map, Yen's threshold, Euclidean Distance Transform, threshold on the Euclidean distance, triggers superimposed on the input TF map.

6.1.2 Detection statistics

In order to build a new search pipeline targeting minute-long transients, we need one (or several) detection statistics to summarize what has been found in the TF arrays. These statistics ultimately quantify the loudness of the detected triggers over the background noise. As we now have 2 output maps, the Anomaly and Glitch maps, we can use both contents to define several useful statistics.

We have shown in Chapter 5 that the anomaly and glitch scores allow to rank the detections by summing over the pixels respectively in the Anomaly map and the Glitch map. We can therefore apply the same methodology for each individual trigger. We thus define the anomaly score for a trigger t as:

$$AS_t = \sum_{\text{mask}} A \tag{6.1}$$

where A stands for the Anomaly map and the sum is carried over the trigger mask. In the same way, we can assign a glitch score to each identified trigger with:

$$GS_t = \sum_{\text{mask}} G \tag{6.2}$$

where G is the Glitch map. We can combine these two scores into a unique statistic via:

$$\kappa = \frac{AS_t}{AS_t + GS_t} \tag{6.3}$$

With this definition, κ should tend to 1 in the presence of a coherent burst signal and take values $\ll 1$ in case of glitches. For convenience, we define:

$$p_\kappa = -\,log(|1-\kappa|), \tag{6.4}$$

such that the detection statistic increases with the significance of the trigger. Note that, in the case of triggers containing few pixels with low values, AS_t and GS_t could be of the same order, leading to high values for p_κ. To discard such triggers, we can impose a minimum value on AS_t to be considered a valuable trigger. A background analysis is necessary to choose this threshold appropriately.

6.1.3 Trigger features

Aside from the former statistics, triggers also have a morphology that could help us to classify them as a GW signal or as noise. Short signals ($\leq$ 6 seconds) showing a large bandwidth are likely to be glitches. Even though ALBUS has been trained to discriminate glitches from potential GW signals, exceptionally loud glitches (or at least a part of them) could still leak into the Anomaly map. In this context, we record the bandwidth and duration of the triggers which could be useful in future postprocessing cuts. Additional features are also provided via the Scikit-image package [147]. The package proposes various measures of labeled regions such as their area, their mean intensity, their orientation, etc. Some of these measures could be useful in discriminating background triggers from GW candidates. In total, more than 30 geometric features are reported. The exhaustive list of features and their definition can be found in Appendix B. Figure 6.2 illustrates the complete workflow of our pipeline (Pyxel + ALBUS) from the pre-processing steps to the evaluation of the anomaly score and the p_κ statistic.

The anomaly score and the p_κ statistic might not be the adequate detection statistic when considering millions of background TF maps. Their definitions arise from the analysis of tens of thousands of images, which only represent a subset of the required millions of background maps to be analyzed for testing new pipelines. It is therefore necessary to assess which statistics or features are the most helpful to classify triggers into background noise and potential GW events. For this, we will use XGBoost [148], a decision tree machine learning algorithm. Decision trees are trees of if-else statements that aim at predicting the label of the input data while minimizing the number of tree branches. XGBoost produces multiple trees at once to test which features and which trees best classify the data. Once trained, XGBoost provides an importance score for each feature in the training dataset. The importance score quantifies how valuable was an attribute in the construction of the decision trees. The more a feature is used to build key decision trees, the higher the relative importance. In turn, we can use these importance scores to select the most relevant features and combine them into a unique statistic or use them as post-processing cuts.

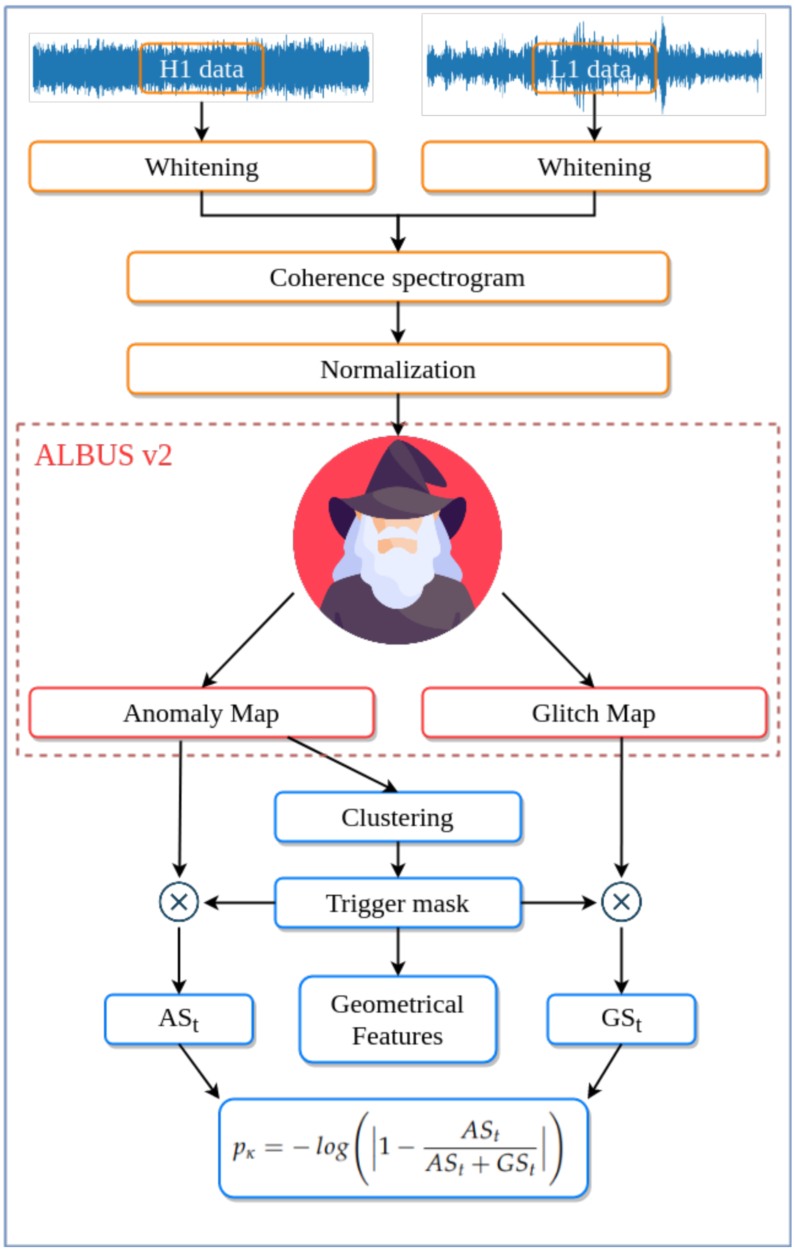

FIGURE 6.2: Diagram illustrating the workflow of Pyxel with ALBUS as the main detection engine.

6.1.4 Preparation for a new run

As the detectors are improved during upgrade phases, the background noise spectrum changes and new glitch classes can appear [58]. Even if ALBUS has learned to highlight high-value pixels in various TF maps, its performances are driven by the noise seen during the training. It is thus preferable to train ALBUS on the noise of the current run for optimal results. The strategy to form the 4 required datasets is described in Chapter 5 and illustrated in Fig. 6.3. Note that the glitch selection process is the main time-consuming step in order to prepare ALBUS for the current observing run.

As an alternative, one can also train ALBUS from its latest training state obtained on data from the previous run. The strategy would consist in training ALBUS on data from the new run with a low learning rate ($10^{-5} - 10^{-6}$) so that its weights are only slightly modified. In this way, the glitch selection procedure and the training set generation can be drastically shortened. As an example, we could only 10000 TF maps rather than the actual 40000 to train ALBUS, speeding up the data generation by a factor of 4. Note that the training would also be faster with fewer TF images although it only amounts to a couple of hours.

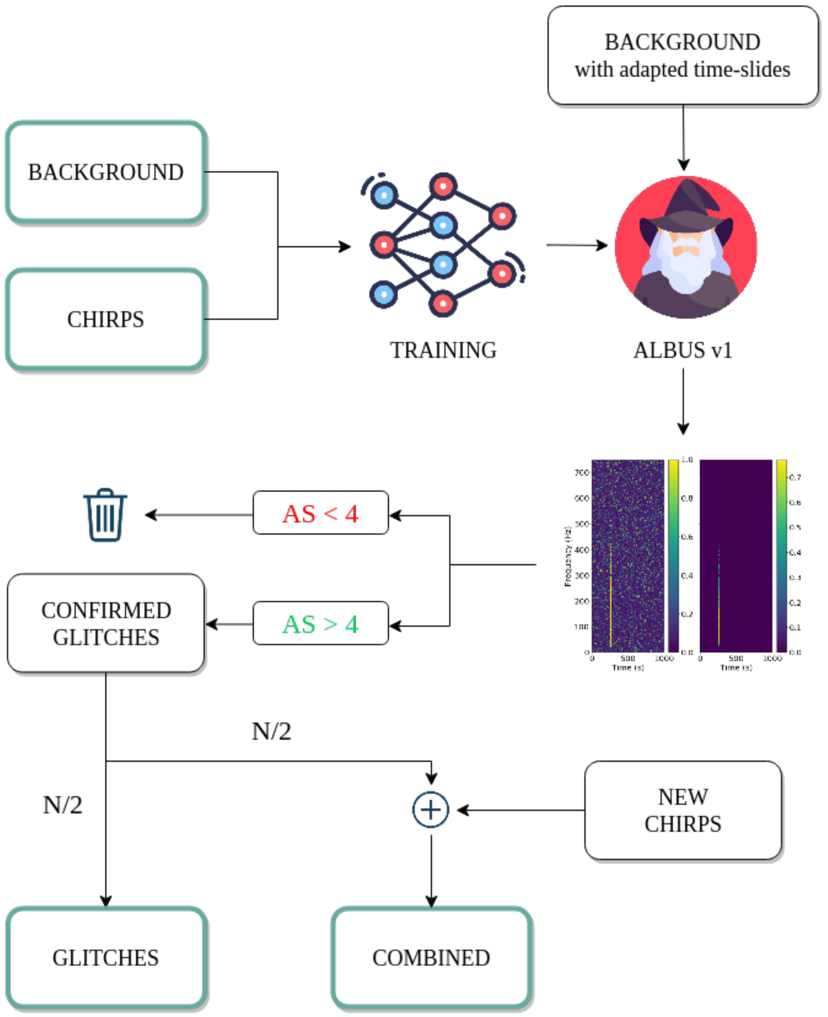

FIGURE 6.3: Diagram illustrating how the training datasets of ALBUS are formed. The boxes enclosed in green contours are then used to train the final version of ALBUS.

6.2 Burst all-sky pipeline benchmark

6.2.1 Description of the project

The LIGO-Virgo-KAGRA Burst group has planned to benchmark all burst pipelines as a preparation for the O4 run. The project consists of a Mock-Data Challenge (MDC) using O3 data from the LIGO and Virgo detectors in which waveforms are injected. Specifically, the datasets cover 40 days of O3 from GPS time 1262304000 (January 5 2020 23:59:42 UTC) to 1265760000 (February 14 2020 23:59:42 UTC). The waveform models are injected in 10 different datasets, provided as multiple strain channels.

The primary goal of the benchmark project is to offer a fair comparison between online and offline pipelines. For this, one measures the fraction of GW events that are detected above threshold as a function of the amplitude (efficiency curve) and the number of false detections (false alarm). The results shall be provided as a list of triggers with the following information:

- peak GPS time
- start GPS time
- end GPS time
- characteristic frequency (Hz)
- lower frequency (Hz)
- upper frequency (Hz)
- ranking statistic value
- False-alarm rate (Hz)

Most of these features are directly obtained from the triggers after the clustering step except the False-Alarm Rate (FAR). In order to derive its value for every single trigger found in the TF map, we have to compare them with background triggers. The background analysis therefore consists in processing a substantial amount of TF maps to produce relevant FAR thresholds, such as 1/10 years or 1/100 years.

6.2.2 Background analysis

In order to produce background TF maps, we first select the time segments where both H1 and L1 were simultaneously observing from January 5 to February 14, 2020. This is done so that we can compare the background maps obtained via time slides and the zero-lag TF maps, that we will call foreground in the rest of this work. The analysis of the background and the foreground should give statistically equivalent results. In total, we analyzed 100 years of background, corresponding to more than 3 million TF maps spanning 1000 seconds of data. Out of these images, we recorded more than 16 million triggers. In the foreground case, H1 and L1 share only 2034 1000-second-long segments, adding up to roughly 24 days. The Anomaly and Glitch scores of the triggers found in both analyses are shown in Fig. 6.4. The distributions of the foreground triggers match the distributions of the background both for the Anomaly and Glitch scores. More precisely, their distributions appear to be similar within a factor 10^3, which is consistent with the number of TF maps processed in both cases.

Two populations of triggers exist in the background. The first population shows low Anomaly and Glitch scores from 10^{-6} up to 10^{-2}. The triggers belonging to this family of background events are single pixels located in the lower left corner of the TF maps. They arise as a consequence of our clustering algorithm. When the Anomaly map contains almost exclusively low values (meaning that nothing relevant has been highlighted in the input TF map), Yen's threshold return a default value that points toward the first pixel of

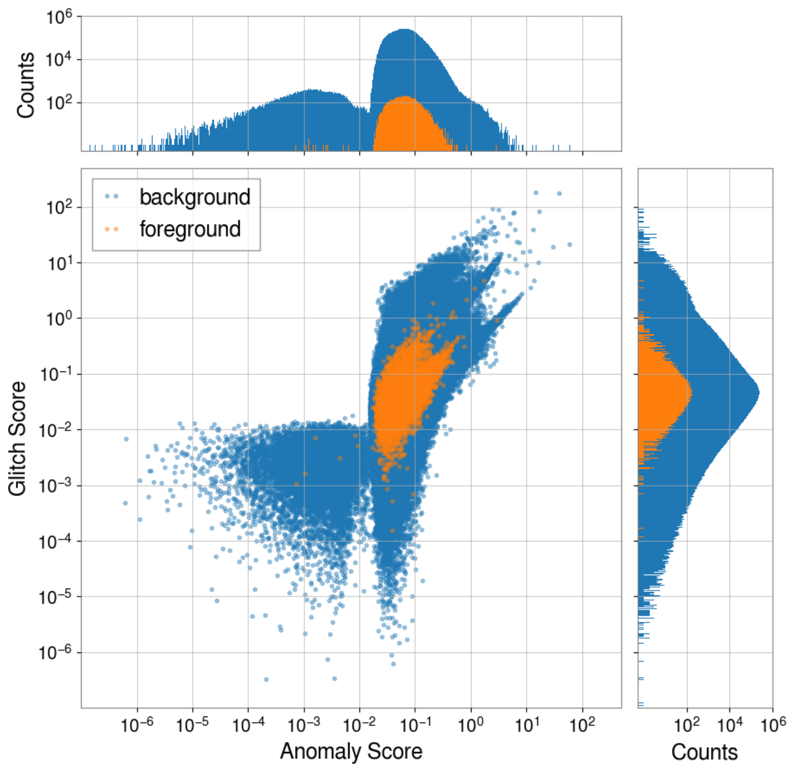

FIGURE 6.4: Anomaly scores and Glitch scores for 100 years of background triggers. For comparison, the triggers found in the foreground are superimposed over the background triggers.

the array. That pixel then goes through the other steps and is finally recorded as a trigger. As the subsequent triggers show low scores, they have low impact on our analyses. This particularity has been removed from the most recent version of Pyxel.

The second population consists of most of the triggers found during the analysis. The Glitch score of the relevant triggers seems independent of the Anomaly score except at high values. This effect is the consequence of the use of the trigger mask. As the mask is defined through the Anomaly map, higher Anomaly scores originate from wider trigger masks, which in turn leads to higher Glitch scores when applied over the Glitch map. This trend is nonetheless not observed for all combinations of scores. Some spikes can be depicted in Fig. 6.4 in both distributions, revealing triggers of different origins. Fig. 6.5 illustrates an example of a trigger belonging to the spike with the lowest Glitch scores. Even if the intensity of the pixels is low, Yen's threshold reveals a vertical line at the border of the image, around 1700 Hz. Similar lines also appear at both edges of the map around 200 Hz for triggers that are part of the lower and central spike. They are shown in Appendix C. The cause of these spikes might have been edge effects caused by down times of H1 and/or L1 right before the start or shortly after the end of the map. However, it would explain less than 5% of all the concerned triggers ($\approx$ 2600). In view of the symmetrical locations of the triggers and the checkerboard patterns appearing in the Anomaly map of Fig. 6.5, these triggers likely come from our neural network. As ALBUS does not detect any clear cluster of high-value pixels, both the Anomaly and Glitch maps show low values that reveal its inner convolution patterns. The checkerboard pattern appears clearly when feeding a null TF map to ALBUS, as can be seen in Fig. 6.6. However, the triggers related to these artifacts are not harmful to our analysis since they show pixel intensities at least one order of magnitude lower than the loudest background triggers.

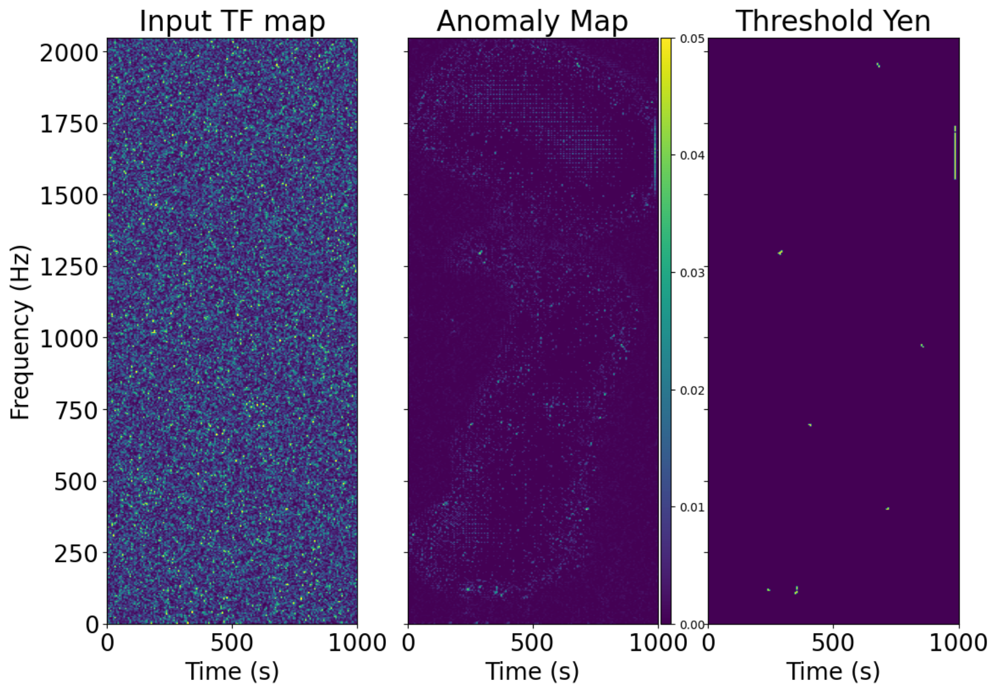

FIGURE 6.5: Left: Input background TF map starting at GPS times 1263922618 and 1263897618 respectively for H1 and L1. Centre: Anomaly map, output of ALBUS. Right: Result of the Yen's threshold applied to the Anomaly map. ALBUS does not find any evidence for correlated pixels in the input TF map, revealing artifacts at low values (10^{-3}) which are further selected with Yen's threshold.

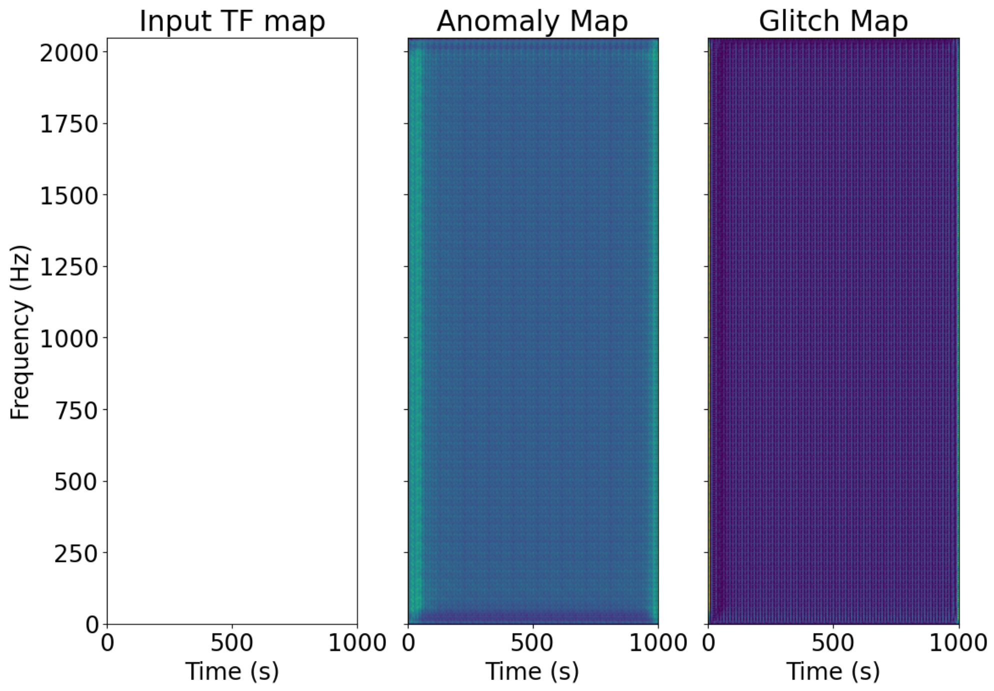

FIGURE 6.6: Left: Input TF map filled with zero values. Centre: Anomaly map, output of ALBUS. Right: Glitch map, output of ALBUS. As the input pixels are all set to zero, the inner convolution operations of ALBUS are revealed in the Anomaly and Glitch maps.

The third spike in the background distribution shows large values of Glitch scores while spread over a larger area. It corresponds to glitches that are sufficiently loud to leak into the Anomaly map, as shown in Fig. 6.7. As both maps are produced by ALBUS, the same neurons are responsible for the detection and classification of pixels in those maps. However, their values are different and the triggers show a Glitch score higher than their Anomaly score. Such triggers should therefore not be a major concern when considering their p_κ value.

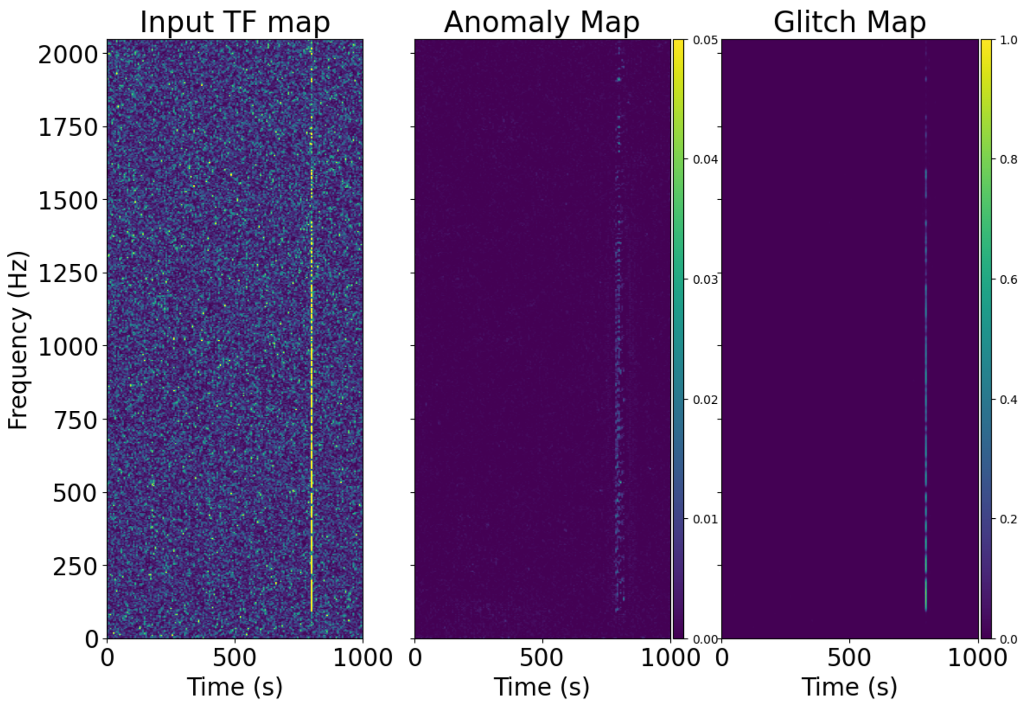

FIGURE 6.7: Left: Input background TF map starting at GPS times 1262688618 and 1262677618 respectively for H1 and L1. Centre: Anomaly map, output of ALBUS. Right: Glitch map, output of ALBUS. The Anomaly and Glitch scores associated with the loudest trigger are 1.43 and 25.86 respectively. The glitch appearing in the LIGO data leaks into the Anomaly map although being well classified in the Glitch map.

It is useful to show the distribution of triggers in frequency with respect to the date to check if ALBUS has effectively covered the 40 days allocated for the Burst MDC. This is done in Fig. 6.8, where only the triggers showing an Anomaly score above 0.5 are shown. As we can see, some dates do not show any trigger, leaving blank spaces in the figure. These times correspond to periods where H1 was not observing in coincidence with L1. The most remarkable missing segment happened around the 18th of January 2020. At that date, a series of microseisms caused the Livingston detector to lose the lock for more than 2 days, before getting back to normal operations 3 days later. This explains the blank space left in Fig. 6.8. Another remark concerns the presence of triggers forming horizontal lines around 1500 Hz, probably coming from the violin modes of the silica fibers holding the mirrors.

Another way of representing the distribution of triggers consists in showing their centroid over a single TF map. Fig. 6.9 illustrates how the loudest triggers are spread in time and frequency. With the exception of the triggers coming from our network's artifacts, most triggers are found in the center of the map with frequencies from 100 Hz to 500 Hz. This observation is the consequence of two effects. Firstly, as we show the centroid of the triggers, long triggers will barely appear at the start or end of the TF map, concentrating

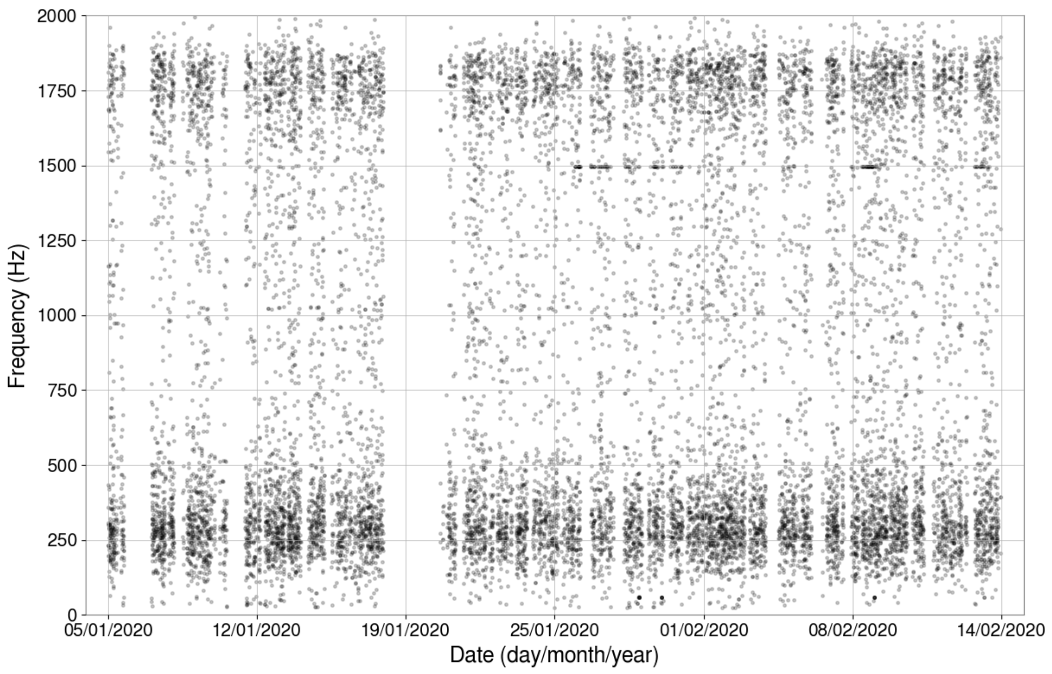

FIGURE 6.8: Trigger distribution as a function of the date at the H1 detector for 100 years of background. For the sake of readability, only the triggers showing an AS above 0.5 are shown.

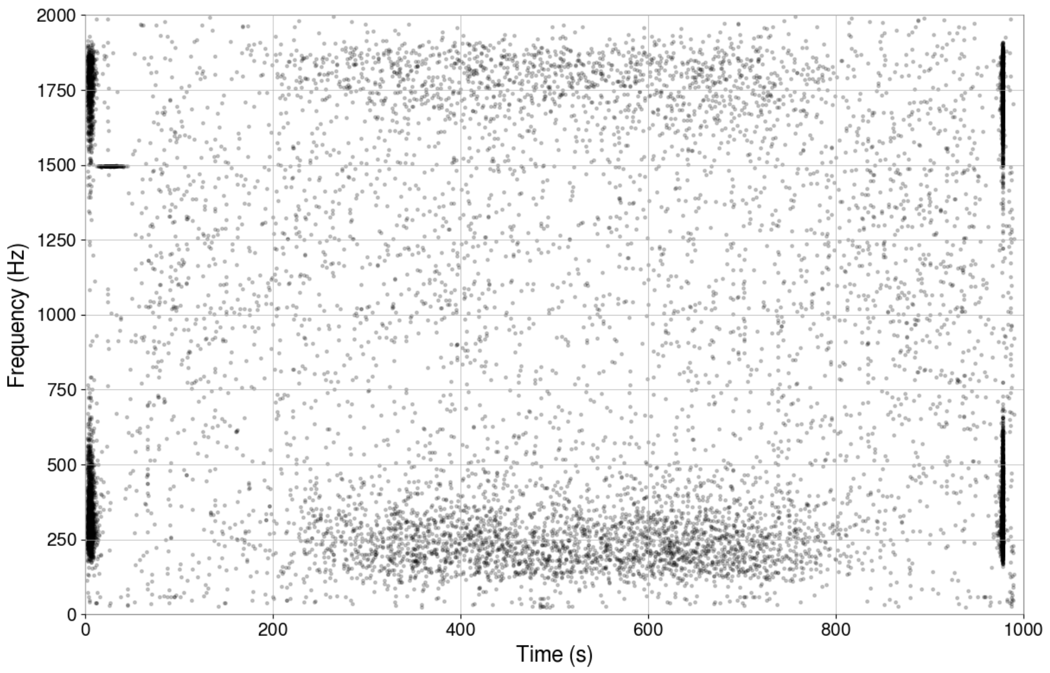

FIGURE 6.9: Trigger centroid distribution for 100 years of background. For the sake of readability, only the triggers showing an AS above 0.5 are shown. With the exception of the edge artifacts, most of the triggers are found below 500 Hz.

the triggers in the central regions of the image. Secondly, most of the loudest background triggers are likely to come from glitches arising in the data of H1 and/or L1. From detector characterization studies [36, 58], we know that most of their power appears at low frequencies, typically below 1000 Hz. It is therefore legitimate to find the centroid of the majority of background triggers below 500 Hz. A further remark concerns the triggers appearing along the violin mode of the silica fibers at 1500 Hz in the first 50 seconds of the map. Their origin can be related to the down times of detectors caused by earthquakes. Indeed, the violin resonances are excited by earthquakes, which impacts the TF maps [36]. Fig. 6.10 shows this effect, particularly strong for the 500 Hz violin mode, when the H1 detector was down twice on the 24th of January 2020. The last comment refers to the slight excess of triggers at high frequencies. Since ALBUS has been trained with random chirps covering the time-frequency plane, no bias was introduced during the training. As a verification, we show the distribution of triggers for roughly 40000 injections in Fig. 6.11. It can be seen that our network performs equally well whatever the central frequency of the injected models. The slight excess of triggers at high frequencies could then possibly comes from the noise itself although no definite source has been identified.

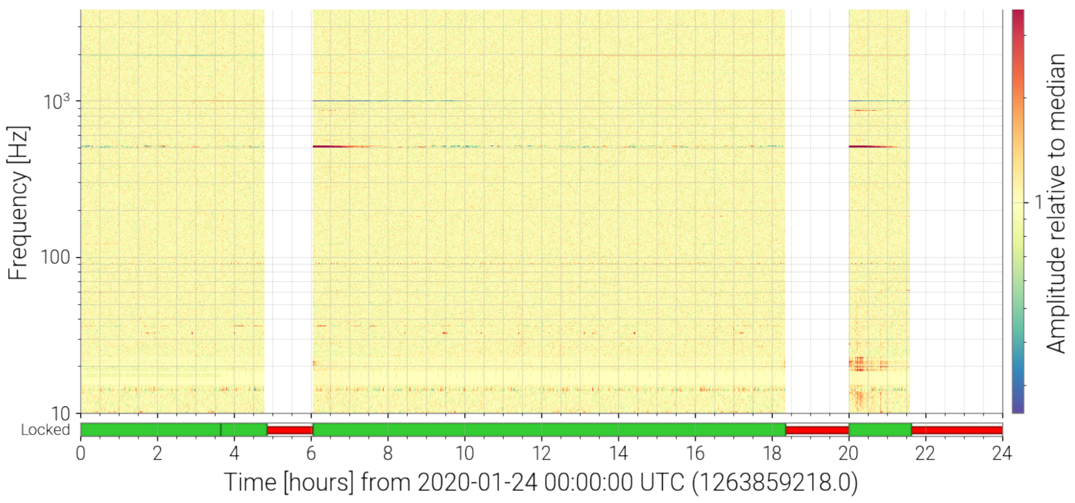

FIGURE 6.10: Time-frequency map of the H1 strain on 24th January 2020. The bottom line shows the periods where the detector was locked (green) and not locked (red). When the detector is not locked, the strain is considered unfaithful and no data are available, leaving blank spaces. As the earthquake excites the violin modes of the silica fibers, their amplitude rises for a couple of hours before returning to a normal state. Credits: Detector characterization group.

In order to build the optimal detection statistic based on the features we have recorded, we use XGBoost to rank the features based on their importance. For this, we perform multiple injections of burst waveforms and process them with ALBUS. The exhaustive list of models includes *msmagnetar-A* [149], *ISCOchirp-A* and -C [79], *PT-A* [78], *ADI-B* and -D [80], *NCSACAM-C* [83], *CM09-long*[82]. The waveforms are injected over a range of h_{rss} values from $1e^{-22}$ to $9e^{-20}$. Then, we group all the triggers and their features into a single text file. Note that some features are discarded since they are not pertinent to discriminate whether the trigger comes from the noise or from an actual GW signal. For example, the GPS times as well as the centroid and the bounding box are not pertinent. In the same way, we gather the background triggers showing an Anomaly score above 0.5. In total, 13850 background triggers and 17860 injection triggers are found in the dataset. Once XGBoost is trained to find the best decision trees, it indicates the relative importance of each of the features. We report the results in Fig. 6.12.

As it can be seen, the mean intensity of the pixels is by far the most decisive criterion when classifying triggers. However, relying on this single feature to detect GW signals is not optimal since the loudest background triggers show similar pixel intensities. Rather,

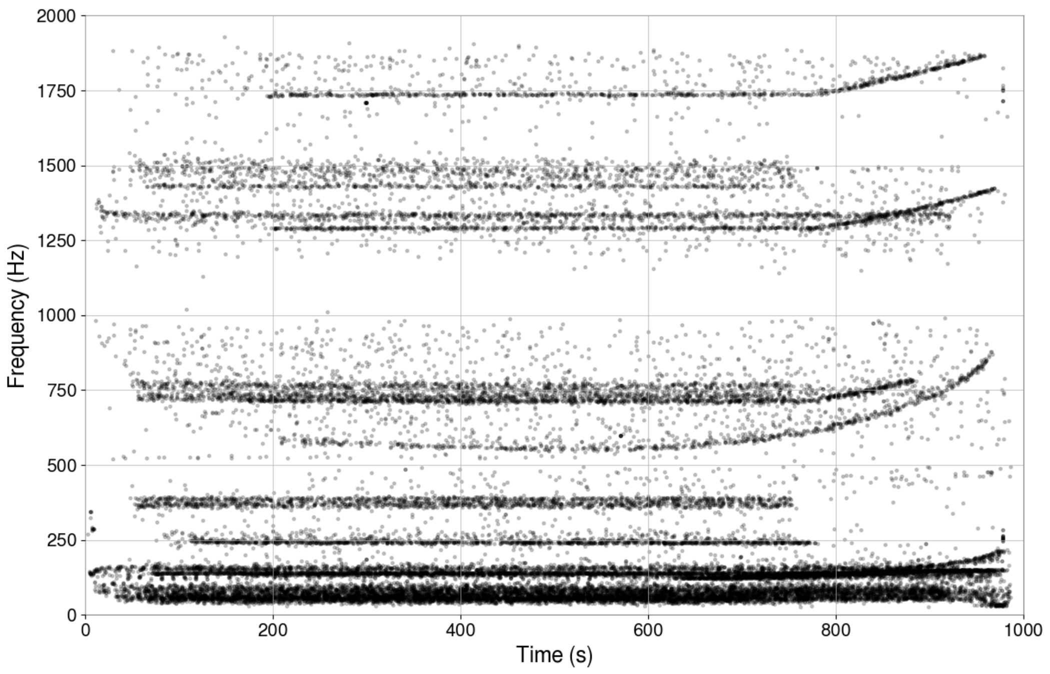

FIGURE 6.11: Trigger centroid distribution for 39600 injections. For the sake of readability, only the triggers showing an AS above 0.5 are shown. The triggers appear to be evenly distributed both in time and in frequency. The ascending curves arise because long chirping down signals are cut when they exceed the extent of the map.

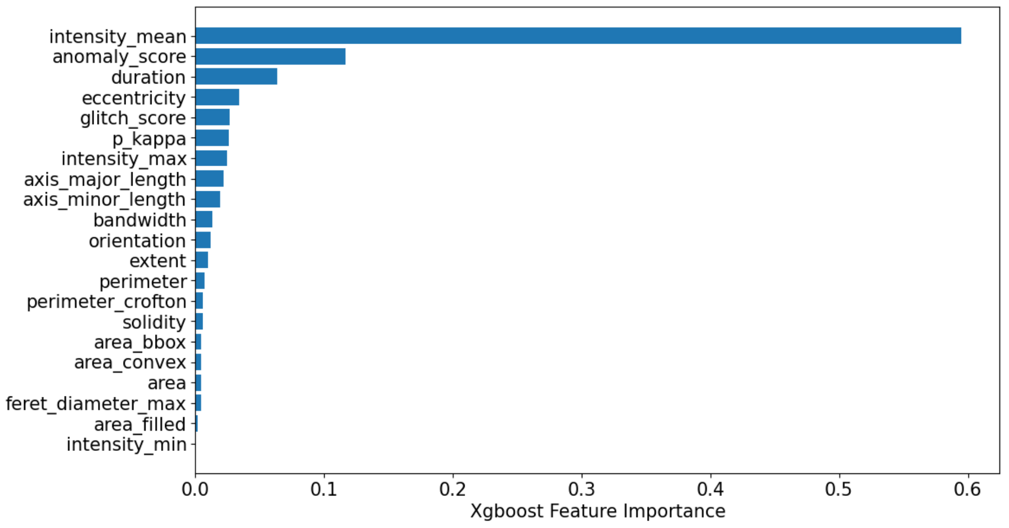

FIGURE 6.12: Ranking of the trigger features based on the importance estimated via XGBoost.

it is preferable to use a combination of the top features to build a unique statistic. Among the 6 most important features, 4 of them are correlated with the strength of GW signals: the mean intensity, the anomaly score, p_κ and the duration. For the first 3 features, it is obvious that the louder the GW signal, the larger their values. For what concerns the duration, it is not related to the amplitude of the GW signals in the first place. However, longer signals both cover more pixels and are less likely of coming from a glitch in the data. The fourth feature in XGBoost ranking, namely the eccentricity of the trigger, can also help differentiate glitches from GW signals. It is defined as the eccentricity of the ellipse that has the same second moments as the trigger. This feature can be viewed as a measure of the curvature of the trigger shape. If the trigger is a vertical or horizontal line, the eccentricity therefore approaches zero. The eccentricity is then useful in discarding triggers coming from glitches or violin modes as well as power lines. In view of the above conclusions, we propose the following detection statistic:

$$D_s = I_m^3 \, AS \, d \, e \, p_\kappa \tag{6.5}$$

where I_m is the mean intensity of the pixels belonging to the trigger, AS is the Anomaly score, d is the duration, e is the eccentricity and p_κ is defined in expression (6.4). The power 3 in the above expression is used to weigh the importance of the mean intensity with respect to the other features.

Note that we did not include the Glitch score in expression (6.5). As the Glitch score quantifies the proportion of the trigger that has been classified in the Glitch map, it increases for triggers originating from glitches but leaking in the Anomaly map. It would therefore have appeared in the denominator of (6.5). However, moderate values of glitch scores (< 5), which can appear for loud GW injections, would have highly penalized our detection statistic. Rather, it appears in the expression of our statistic through p_κ, which still accounts for the Glitch score.

We can now examine the loudest background triggers with respect to our statistic D_s and eventually determine thresholds on other features (i.e. cuts) to lower the FAR of GW triggers. In Table 6.1, we show the main characteristics of the 10 highest triggers while their TF footprints are overlaid in a single TF map in Fig. 6.13. Note that the orientation refers to the angle (in radians) the trigger makes with respect to the vertical. As we can see, the first 2 triggers stand above the others in terms of D_s values due to their high Anomaly score. Besides, 4 of the 5 top triggers show a relatively high Glitch score correlated with a very steep footprint, and therefore a low orientation. These triggers are confirmed to be loud glitches (SNR $>$ 100) after inspection of the daily summary pages provided by the detector

D$_s$	**I**$_m$	**AS**	**GS**	**p**$_\kappa$	**e**	**dur.** (s)	**bw** (Hz)	**orientation**
3.7327	0.1572	59.56	20.96	1.35	1.00	12	452	-0.0036
2.5469	0.1706	30.20	19.12	0.95	1.00	18	192	0.0004
0.1161	0.0846	16.42	9.97	0.97	1.00	12	248	-0.0093
0.0936	0.1146	3.90	3.05	0.82	0.81	24	36	0.2489
0.0889	0.0882	9.80	8.91	0.74	0.99	18	120	0.0141
0.0627	0.0857	2.57	0.15	2.88	0.75	18	32	0.2701
0.0617	0.0533	2.82	0.18	2.82	0.95	54	60	0.5309
0.0612	0.0530	3.60	0.70	1.81	0.88	72	32	-1.3579
0.0352	0.0639	8.75	15.69	0.44	0.97	36	104	-0.0210
0.0342	0.0677	2.85	0.56	1.80	0.90	24	44	0.2789

TABLE 6.1: Main features of the 10 loudest background triggers ranked as a function of their D_s value.

characterization team. Some GW models like *ISCOchirp* or *ADI* might also show a very steep behavior but they either have a longer duration or a chirp-down behavior, which is not observed in the case of the background triggers. After visual inspection, it appears that ruling out triggers showing an absolute orientation lower than 0.015 (equivalent to a tilt of 0.86°) eliminates half of the 100 loudest triggers.

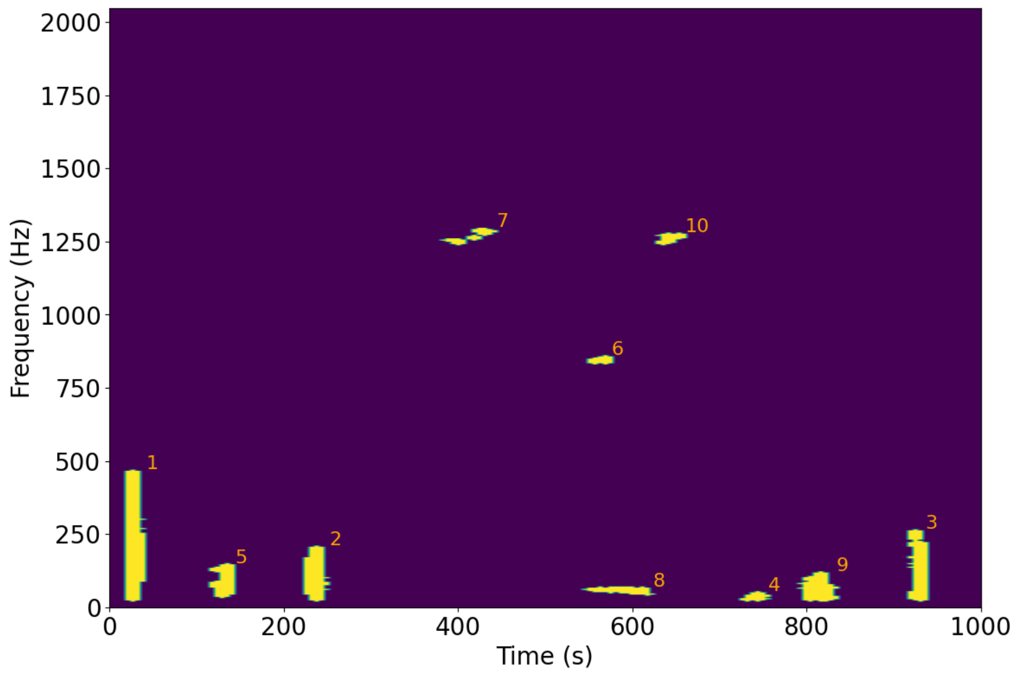

FIGURE 6.13: Footprints of the 10 loudest background triggers shown in a unique TF map. The number next to the trigger indicates its rank among the 10 loudest events.

It is important to associate a FAR to any trigger when running a search. As we have found the optimal statistic regarding our recorded features and an adequate cut on the orientation, we can draw the cumulative inverse histogram of D_s to determine the values associated with FARs such as 1 per 10 years or 1 per 100 years. This is done in Fig. 6.14. The threshold on the orientation allows the elimination of the tail of the histogram so that it falls off sharply. Such behavior is expected for well-conditioned pipelines and confirms that our detection statistic is appropriately defined. We can now assign a FAR to any trigger coming from the foreground or from injected models and evaluate their likelihood to be GW signals.

6.2.3 Foreground analysis

As we compared the background and foreground distribution in terms of Anomaly and Glitch scores, it is important to verify if any foreground trigger stands above the others. It would then potentially constitute a follow-up candidate. For this, we rank the top 10 foreground triggers as a function of their D_s statistic in Table 6.2. Note that we apply the cut on the orientation beforehand. The second column shows the FAR associated with their D_s value based on the background histogram. The loudest trigger has a FAR equal to 7.54e-07, equivalent to an event happening twice per month of observing data. As we analyzed roughly 25 days of coincident data, it is expected to find such triggers. We can therefore confidently conclude that no GW event has been found by our pipeline in the foreground data.

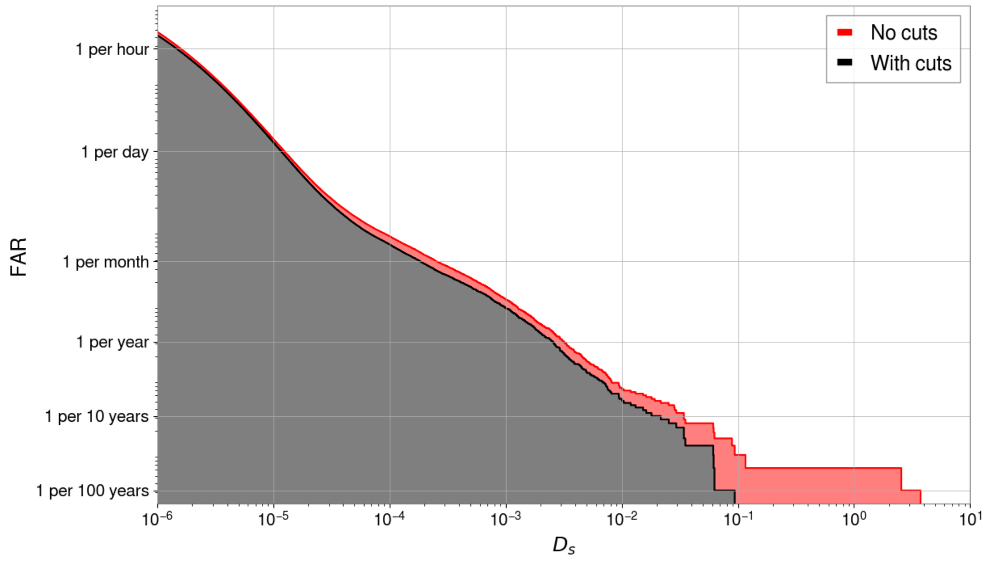

FIGURE 6.14: Histogram of the D_s values for 100 years of background triggers. The red area indicates the distribution of all triggers while the grey area refers to triggers passing the orientation cut.

$\mathbf{D}_s$	**FAR** (Hz)	$\mathbf{I}_m$	**AS**	**GS**	$\mathbf{p}_\kappa$	**e**	**duration**	**bandwidth**
0.000083	7.5454e-07	0.0187	0.47	0.16	1.37	0.83	24	28
0.000053	1.1536e-06	0.0167	0.43	0.18	1.23	0.90	24	32
0.000024	3.3597e-06	0.0113	0.35	0.17	1.14	0.85	48	24
0.000021	4.1552e-06	0.0089	0.36	0.14	1.31	0.96	66	44
0.000020	4.5498e-06	0.0068	0.72	0.46	0.94	0.88	108	48
0.000019	5.0490e-06	0.0174	0.24	0.09	1.29	0.63	18	20
0.000019	5.0656e-06	0.0124	0.38	0.16	1.21	0.70	30	32
0.000018	5.2370e-06	0.0146	0.28	0.09	1.43	0.61	24	20
0.000017	6.0064e-06	0.0126	0.29	0.10	1.37	0.88	24	32
0.000017	6.0255e-06	0.0192	0.19	0.04	1.66	0.62	12	16

TABLE 6.2: Main features of the 10 loudest foreground triggers ranked as a function of their D_s value.

6.2.4 Performance on injected waveforms

A set of waveform models has been selected and injected into O3b data, distributed over 10 different strain channels. Specifically, more than 30 waveforms are included, for which most of them are short duration models. Only a couple of waveforms show a duration larger than our time resolution, namely:

- ADI-B (9.4 s) and ADI-D (142 s) [80]
- BAR-5 (102.5 s) [82, 150]
- CM09-long (2500 s) [82]
- ISCOchirp-A (238 s) [79]

- msmagnetar-A (4000 s) [149]
- NCSACAM-C (65.5 s) [83]
- PT-A (25 s) [78]

We run the analysis of the 10 different channels and report the h_{rss} value at which half of the injections, showing a FAR lower than 1 per 10 years, from each waveform are recovered. This value is known as the 50% h_{rss}. We then compare our results with the best and worst values obtained by various pipelines participating in the MDC in Fig. 6.15. For the sake of confidentiality, the efficiency curves cannot be shown in this work. A dedicated LVK paper is in preparation. Instead, we mark the best and worst 50% h_{rss}, which constitute a range of expected sensitivities, shown as a red box in Fig. 6.15. Note that *CM09-long* and *msmagnetar-A* are very long waveforms, lasting more than 1000 seconds, which cannot be processed by all pipelines. This explains why their respective red boxes collapse into a single horizontal mark, meaning that only 1 other pipeline produced results for these models.

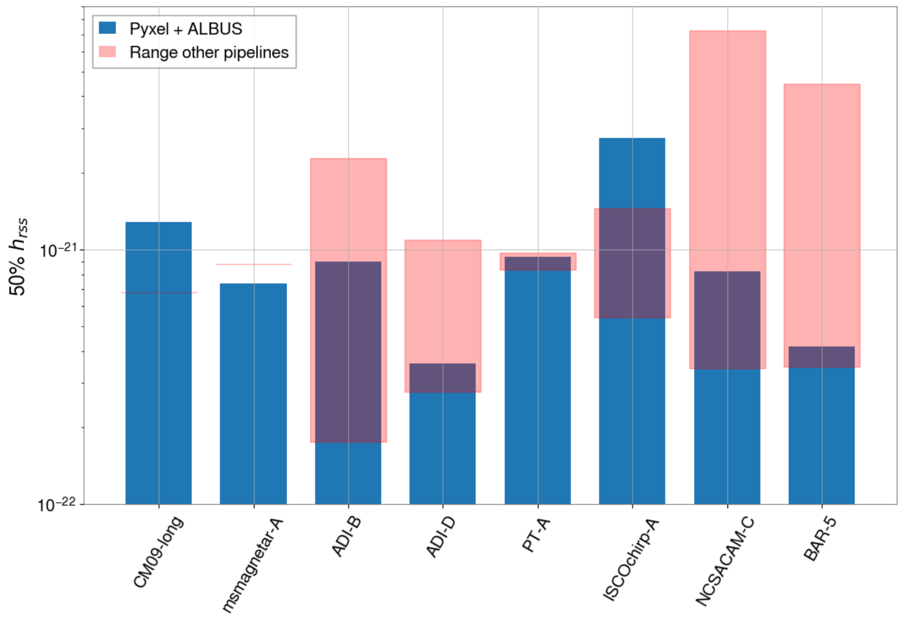

FIGURE 6.15: Comparison of the 50% h_{rss} values achieved by Pyxel (with ALBUS) with the results obtained by several pipelines participating in the Burst MDC. A red box is drawn over each GW model to indicate the best and worst results among the other pipelines. Note that the results are shown for a FAR of 1 per 10 years.

We observe that our results lie in the range of sensitivity of the other pipelines for almost all the waveform models. In particular, our results are close to the best 50% h_{rss} for 3 of the 8 models tested. We also deliver the best results for the *msmagnetar A* model.

Despite these observations, we must recognize that the conditions of the MDC are not optimal for ALBUS. As limited time segments are allocated for the MDC, many injections are found close to each other. As we process 1000-second segments, a lot of TF maps show multiple signals with different amplitudes. Our normalization procedure can thus hinder some injections if a loud signal is present in the map. Fig. 6.16 shows the effect of a loud injection on the other signals. By contrast, Fig. 6.17 illustrates the results when the loud injection is not present in the data. As we can see, the *NCSACAM-C* model is fully recovered in the second case while most of its footprint is missed in the former case. The D_s statistic of all triggers is impacted by the presence of a loud injection, even if they do not

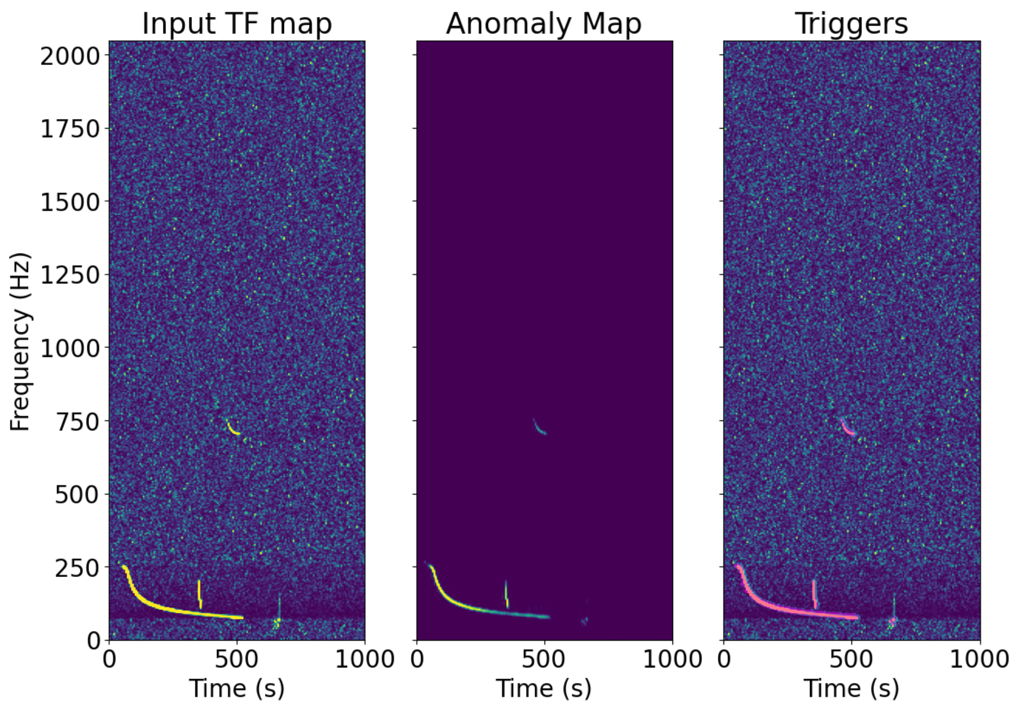

FIGURE 6.16: Left: Input foreground TF map starting at GPS time 1263082149 where 4 injections are performed (from left to right: *GRBplateau* [82], *ADI-B* [80], *ISCOchirp-C* [79], *NCSACAM-C* [83]). Centre: Anomaly map, output of ALBUS. Right: Triggers (pink) superimposed on the input TF map. The presence of the loud *GRBplateau* prevents a clear detection of the *NCSACAM-C* injection.

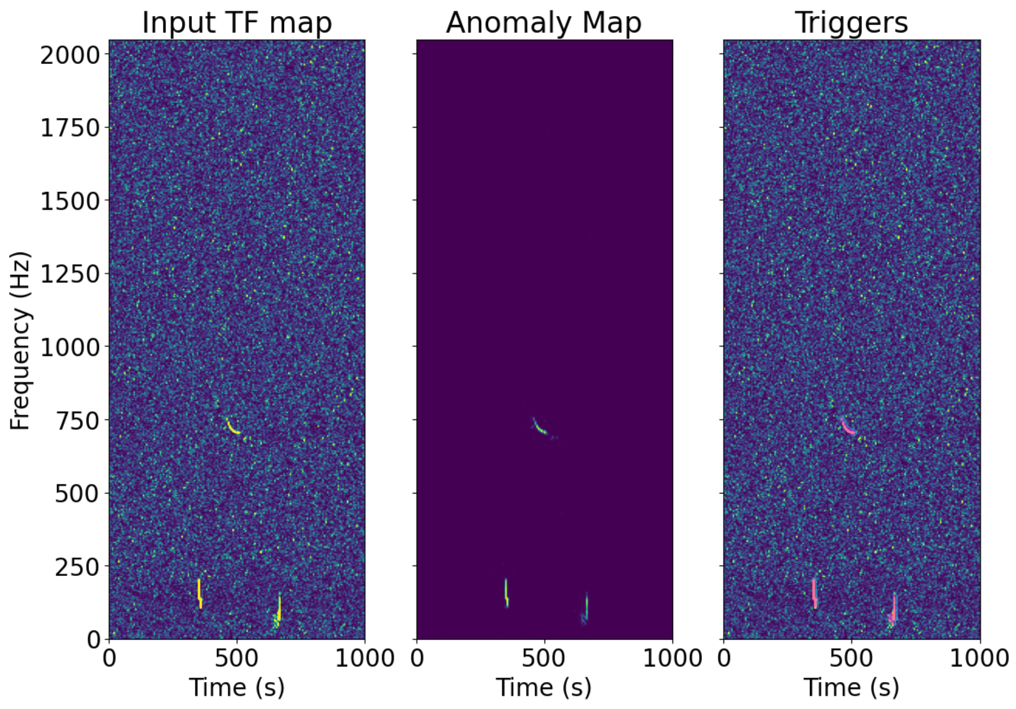

FIGURE 6.17: Left: Input foreground TF map starting at GPS time 1263082149 where 3 injections are performed (from left to right: *ADI-B* [80], *ISCOchirp-C* [79], *NCSACAM-C* [83]). Centre: Anomaly map, output of ALBUS. Right: Triggers (pink) superimposed on the input TF map. Since no loud signal is present in the data, all the injections are recovered properly.

share some frequency bandwidth. For example, the trigger associated with the *ISCOchirp-C* model has a D_s value varying from 3.43 to 0.20 while it lies well above the loud injection. It is even worse for the *NCSACAM-C* trigger, which goes from $D_s = 0.1939$ to $D_s = 0.0043$, corresponding to a FAR of 1 per 25 years. Besides the normalization procedure, the training of ALBUS also affects the results of the MDC. Our network has been trained with TF maps built from the 4 different combinations of 1 glitch and 1 chirp. There is therefore a maximum of 2 "signals" per map and we cannot expect ALBUS to perform as well with more signals. Moreover, since no minute-long GW burst has been observed so far, it is very likely that the first detection will be unique in a 1000-second data segment.

Another remark concerns the results obtained for the *ISCOchirp-A* waveform, a 200-second long GW model. It is surprising for us to obtain the highest 50% h_{rss} among all the other pipelines, including some pipelines tuned to search for short GW bursts. We therefore examined the missed injections to understand the problem. Fig. 6.18 shows a TF map evaluated on a data stream of the first strain channel and the Anomaly and Glitch maps produced by ALBUS. In total, 4 patterns are depicted by the network, for which 3 of them are classified as glitches. However, the long vertical track at around 300 seconds is labeled as a *ISCOchirp-A* model in the MDC injection files. Moreover, it corresponds to a loud injection, with an h_{rss} value equal to 1e-21. From the O3 long duration paper [55] (see Fig. 3.3), this model shows a frequency band above 1400 Hz, which does not match with the footprint observed in Fig. 6.18. To extend the investigation further, we select a very loud missed injection and analyze it with ALBUS in Fig. 6.19. As we can see, two *ISCOchirp-A* models are found in the input TF image. The two signals appear together with a vertical line spanning the whole map. This is likely coming from spectral leakage

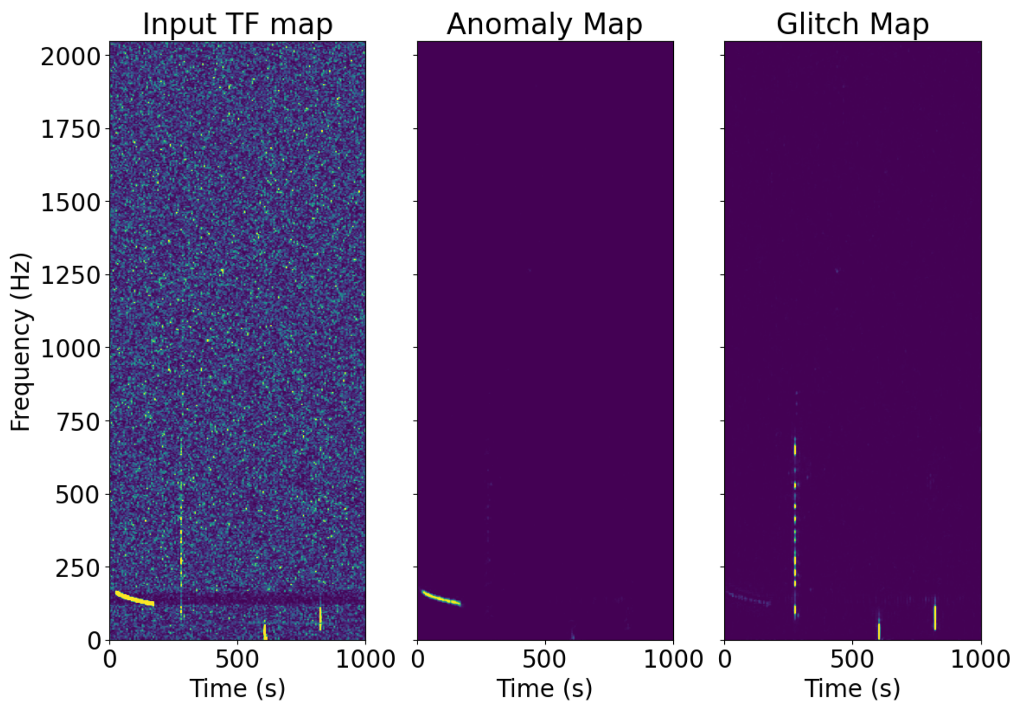

FIGURE 6.18: Left: Input TF map built from the MDC data (channel 1) starting at GPS time 1262553296. Centre: Anomaly map, output of ALBUS. Right: Glitch map, output of ALBUS. The *ISCOchirp-A* waveform, appearing as a vertical dashed line in the input TF image, is classified as a glitch by our network.

from higher frequencies. These artifacts can arise when discrete signals are Fourier transformed. The discontinuities at the borders can cause some of the power to leak into other frequencies, although it does not belong to the real signal bandwidth. These artifacts can be minimized by tapering the waveform signal before adding it to the background noise. In view of Fig. 6.18 and Fig. 6.19, it is likely that this step has not been conducted properly when generating the waveform file. Even if ALBUS is able to discriminate the spectral leakage from the main waveform at high amplitudes, it still impacts the identification of the true waveform footprint by adding spurious high-value pixels all over the Anomaly map. At low amplitudes, the signal is then mainly a vertical artifact classified as a glitch which explains the unexpectedly high 50% h_{rss} for the *ISCOchirp-A* waveform.

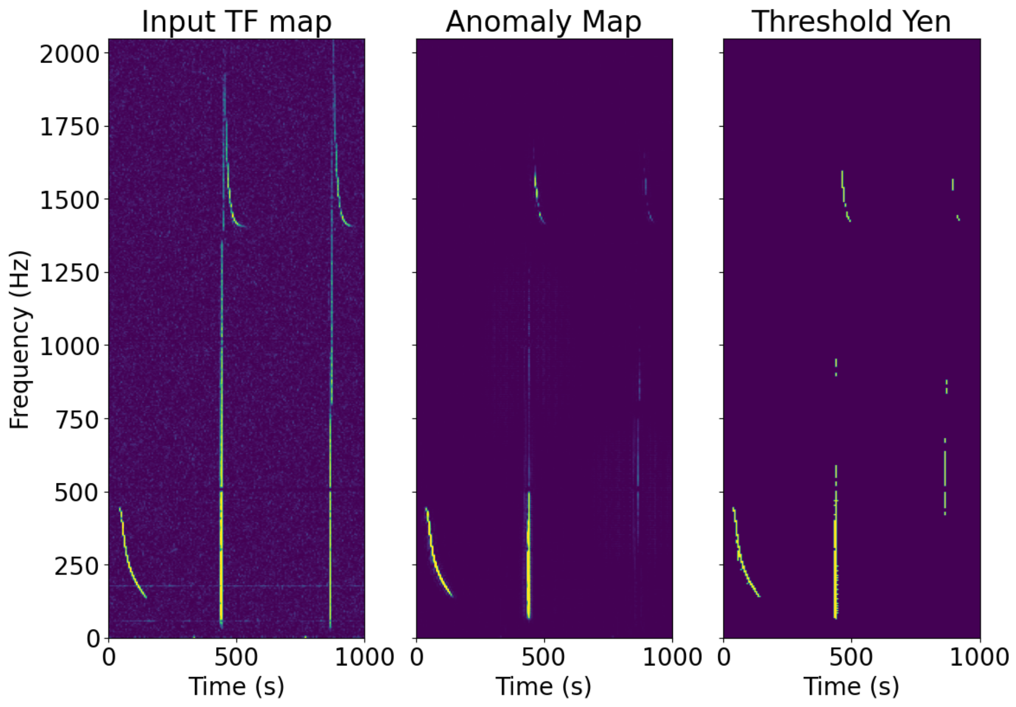

FIGURE 6.19: Left: Input TF map built from the MDC data (channel 1) starting at GPS time 1262304167. Centre: Anomaly map, output of ALBUS. Right: Yen's threshold applied on the Anomaly map. The *ISCOchirp-A* models show unusual vertical lines that affect the output of ALBUS. The second *ISCOchirp-A* injection is not recovered as a potential GW signal since it is split into two isolated triggers.

We must also note that ALBUS has been trained on TF maps built from O3a noise while the MDC is based on O3b. This should have limited impact on the final results in view of the evolution of the LIGO and Virgo BNS ranges over the third observing run, represented in Fig. 6.20. The BNS range gives an indication of the level of noise in the detectors at a precise moment in time. Although it is only slightly different from O3a and O3b, it could be desirable to train ALBUS on noise showing a spectrum similar to the data processed.

The last remark concerns the results obtained for signals shorter or slightly longer our time resolution (6 seconds). In Fig. 6.18, we can see that two short signals (at around 600 and 800 seconds) are reported in the Glitch map. These signals are part of the short duration waveforms injected to benchmark pipelines targeting short bursts. With our resolution, it is basically impossible to distinguish them from glitches. This is also likely happening for the *ADI-B* model. As this waveform is only 9 seconds long, it covers 2 or 3 time bins. At low amplitudes, the signal can therefore be very similar to glitches and

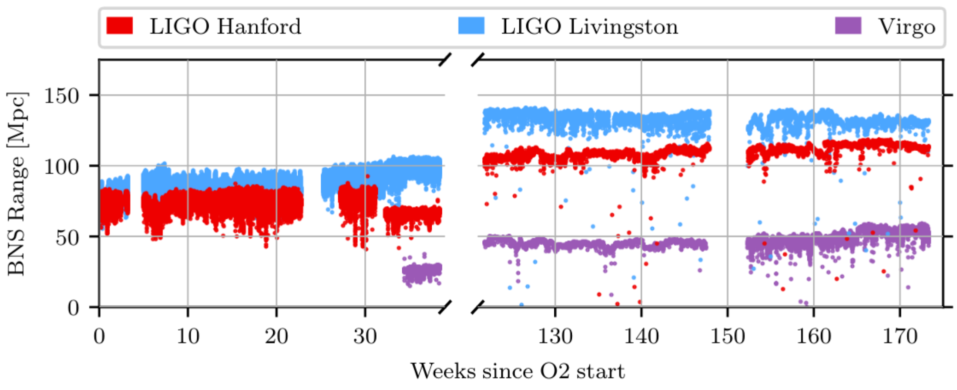

FIGURE 6.20: BNS range of the LIGO and Virgo detectors for the O2 and O3 runs. The gap between weeks 40 and 120 corresponds to the upgrade phase taking place after the O2 run and before the O3 run, which is itself divided into O3a and O3b around weeks 150. Source: [36].

classified as such by ALBUS. Although being competitive with the other pipelines, this explains why our results are closer to the maximum 50% h_{rss} value than the minimum.

6.2.5 Comparison with results from O3 pipelines

We previously concluded that the conditions of the Burst MDC are not favourable for our neural network, which has not been used to deal with a lot of signals in a single image. An alternative would be to build additional TF maps including several chirps and train ALBUS on this new dataset, closer to the conditions of the MDC. However, this would require to conduct again the background analysis, which takes at least an additional month to complete. Instead, we can run an independent analysis with only a single GW signal per TF map and compare our results to the pipelines involved in the O3 long duration paper. We therefore perform injections for 11 GW models showing a duration larger than our time resolution. We select logarithmic-spaced h_{rss} amplitudes varying from 1e-22 to 9e-21, with 100 injections for every value. For the sake of comparison, we inject the waveforms at GPS times corresponding to the MDC challenge. As the pipelines involved in the O3 burst analysis have used a FAR threshold of 1 per 50 years, we stick to this threshold to recover the injections in the TF maps. A trigger is considered to match an injection when it shares at least 5 pixels with the injection footprint. Note that our recovery criterion is slightly more stringent than what is done for the MDC, where a time-frequency bounding box is used. We finally draw the comparison with the best result obtained among all the other pipelines in Fig. 6.21.

Our results show that we reach the lowest 50% h_{rss} for 7 of the 11 waveform models. In particular, our results on the *ISCOchirp-A* model are improved since it does not show any spectral leakage. We also find confirmation that our time resolution is not suitable for short waveforms like *ADI-B*, probably because they are mistaken for glitches. For what concerns the *NCSACAM* waveforms, ALBUS seems to lie slightly behind the best results from the O3 long duration paper. The addition of simulated harmonics in the chirp training dataset has not enabled us to perform better than the best pipelines but it shows encouraging results. ALBUS is definitely better at recognizing their TF patterns with the presence of harmonics in the training datasets.

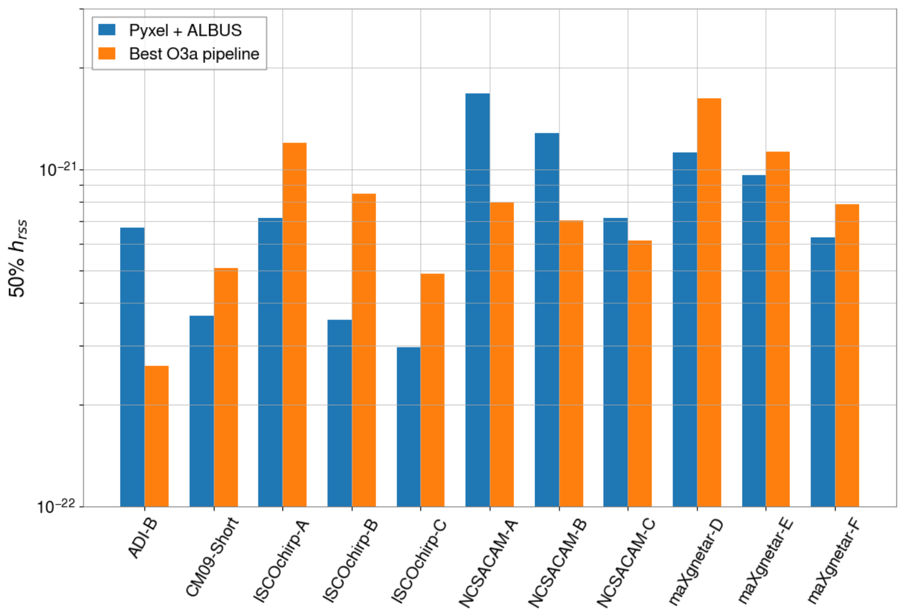

FIGURE 6.21: Comparison of the 50% h_{rss} values achieved by Pyxel (with ALBUS) with the best results obtained by pipelines taking part in the O3 long duration GW paper. Note that the results are shown for a FAR of 1 per 50 years. Source: [55]

It is important to note that the results of the other pipelines date back to 2 years ago. The majority of algorithms behind these pipelines have certainly been improved or complemented with new post-processing techniques such as XGBoost. Despite that consideration, our pipeline is very likely to yield the best results on some waveform models.

If we directly compare the results obtained on the MDC datasets with our local analysis, we can point out an improvement in the results over almost all waveforms. For this, we have added some of the GW models present in the MDC data to our analysis. Then, we select the triggers that have a FAR lower than 1 over 10 years. Fig. 6.22 illustrates the results in terms of 50% h_{rss} values. The unique difference between the two analyses lies in the number of signals seen in every TF map. Note the exception of the very long *msmagnetar-A* and *CM09-long*, which occupy almost the whole TF map and are therefore left as the unique signal in the TF image both in our local analysis and in the MDC datasets. The results associated with these waveforms are therefore expected to depend on the injection recovery procedure. For the others, a clear improvement is observed from 12% to 75% for the *ISCOchirp-A* waveform, to which we have already brought an explanation.

6.3 Future work and prospects

Taking advantage of the speed of neural networks, ALBUS can process roughly 150000 images per day on a 64-core machine. This capability can be exploited to process day-by-day data within less than 3 minutes, which is currently inaccessible for the other long duration burst pipelines. ALBUS can therefore serve as an early tool to highlight significant triggers within the foreground data, which could save precious time for members of the other pipelines. In this way, any significant long duration trigger could be reported within several minutes, enabling astronomers to find potential electromagnetic counterparts.

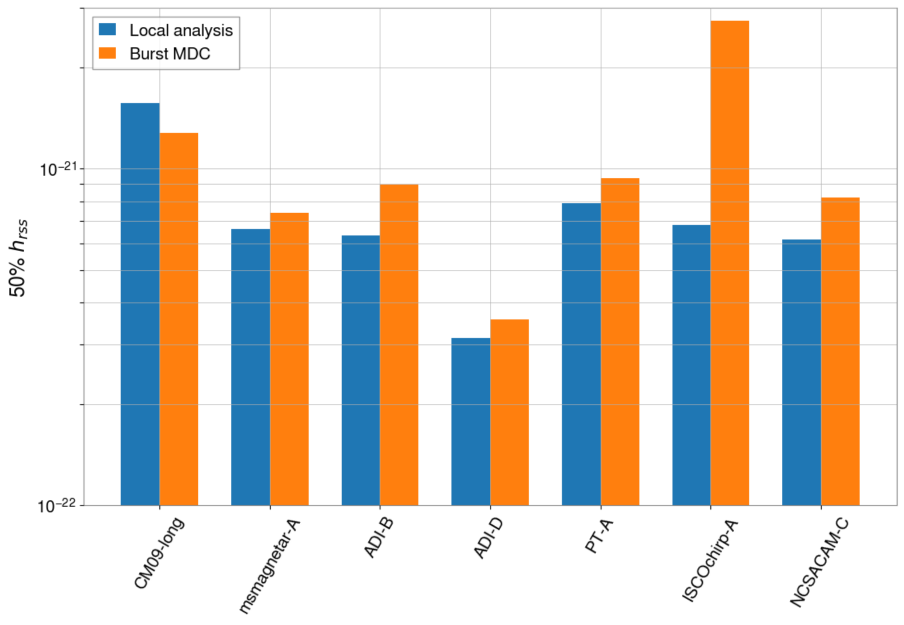

FIGURE 6.22: Comparison of the results obtained by Pyxel (with ALBUS) in the burst MDC with the results of an independent analysis. Note that the results are shown for a FAR of 1 per 10 years.

In future work, we would like to consider the use of neural networks to achieve clustering in the Anomaly map. This would reduce the number of postprocessing steps and prevent them from affecting the number or the final shape of the triggers. For example, Yen's adaptive threshold yields an average of 5 to 6 low-intensity triggers per background map, which greatly increase the number of triggers to process. In the presence of a lot of high-value pixels, Yen's threshold can also eliminate a part of the GW signals, compromising our D_s statistic. We can achieve the clustering operation with a simple convolutional network composed of a few layers. By designing an adequate loss function and adding it to the current loss used to train ALBUS, both pixel identification and clustering could be achieved in a single training procedure.

It could also be valuable to process multiple TF maps with different resolutions. Although ALBUS has been trained on chirps with different lengths and frequency bandwidths, its sensitivity to short waveforms like *ADI-B* is limited by the time resolution of the TF images. Processing multiple images at different resolutions would enable ALBUS to perform well on both long and short burst signals. To this aim, the different TF images could be passed to ALBUS as it is currently the case with the RGB channels.

In the same way, we can remove the normalization procedure from our pre-processing steps. As a consequence, the presence of loud signals would have limited impact on the surrounding faint signals. Without this step, the violin modes of the silica fibers as well as the power line will appear in the input TF map. Their presence is not necessarily harmful to ALBUS since they will also appear in the background images. Throughout the training phase, ALBUS would therefore learn that these lines are part of the background spectrum.

Given the capabilities of ALBUS to distinguish signals of different shapes, it could naturally be adapted to other searches based on TF images. The most straightforward adaption would consist in increasing the time and frequency resolutions to search for short GW bursts, such as GW counterparts from supernovae. Although the correlated noise has a very different signature on scales of tenths of seconds, the methodology described in Chapter 5 could comply with this new paradigm. At this scale, most of the glitches appear with their specific shapes as described by Gravity Spy. Data quality tools like Omicron [151] or Gravity Spy itself can then be used to reduce the rate of false alarms in the data. In the same way, ALBUS can be tuned to search for CBC signatures in the LIGO and Virgo data, usually limited by the presence of blip glitches. A convolutional network like ALBUS, providing both classification and detection at the same stage, seems well indicated to address this problem.

6.4 Conclusion

Throughout this thesis, we have built the first machine learning algorithm dedicated to the search for long duration bursts. As burst searches are not naturally adapted to the use of deep learning methods, we have defined new criteria and techniques to overcome the intrinsic limitations. We employed frequency-swept cosines to mimic long duration waveform models so that our network is not biased toward the signals present in the training dataset. Then, we developed a new amplitude measure based on pixel-to-pixel difference to ensure that chirp signals stand above the background noise. This step was of primordial importance in order to prevent ALBUS not to be fooled during the training phase, which could lead to disastrous results as well as a lot of false detections. Once we generated our dataset, we decided to save the TF maps in RGB format to allow larger batch sizes, which is critical for the convergence of neural networks. From a preliminary analysis, we pointed out the impact of glitches on the results of ALBUS. We thus adapt it to distinguish glitch patterns from potential burst signals.

While integrating ALBUS into Pyxel, we first relied on the Anomaly score to characterize the background and rank the detected signals. As we record the triggers together with several geometric features, we decided to use XGBoost to find the most significant characteristics and use the latter to build an improved detection statistic. This new statistic helps to reduce the number of significant background triggers, which in turn enhances the detection of GW waveforms. The results obtained on the Burst MDC showed that our new pipeline is competitive with the current burst pipelines, although the conditions were not optimal for us. An additional analysis revealed that ALBUS performs better on TF maps showing fewer signals per map. It could however be easily adjusted by adding more simulated chirp signals in the training dataset.

We have shown that neural networks can be applied to the search for long duration transients, despite the numerous limitations inherent to burst searches. Given the expected growth and variety of deep learning models in the future, we hope the GW community will continue to include them in their analyses. Both their speed and accuracy are valuable assets that should prevail in the quest for the extremely faint echoes from our universe that are gravitational waves.

Appendix A

Architecture of ALBUS

Layer	Conv. parameters	N° feature maps	Dropout	Batch norm.	Activation
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	16	No	Yes	ELU
Strided Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	32	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	32	No	Yes	ELU
Addition	/	/	/	/	ELU
Strided Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	64	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	64	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	64	No	Yes	ELU
Addition	/	/	/	/	ELU
Strided Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	128	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	128	Yes (p=0.5)	Yes	ELU

Layer	**Conv. parameters**	**N° feature maps**	**Dropout**	**Batch norm.**	**Activation**
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	128	Yes (p=0.5)	Yes	ELU
Addition	/	/	/	/	ELU
Strided Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	128	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	128	Yes (p=0.5)	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	128	Yes (p=0.5)	Yes	ELU
Addition	/	/	/	/	ELU
Transposed Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	128	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	256	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	256	No	Yes	ELU
Addition	/	/	/	/	ELU
Transposed Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	64	No	Yes	ELU
Zero Padding	padding=(0,0,1,0)	/	/	/	/
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	128	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	128	No	Yes	ELU

Layer	**Conv. parameters**	**$N°$ feature maps**	**Dropout**	**Batch norm.**	**Activation**
Addition	/	/	/	/	ELU
Transposed Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	32	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	64	No	Yes	ELU
Addition	/	/	/	/	ELU
Transposed Conv.	kernel=(3,3) stride=(2,2) padding=(1,1,1,1)	16	No	Yes	ELU
Zero Padding	padding=(0,0,1,0)	/	/	/	/
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	32	No	Yes	ELU
Addition	/	/	/	/	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	16	No	Yes	ELU
Conv.	kernel=(3,3) stride=(1,1) padding=(1,1,1,1)	1	No	No	ELU

Appendix B

List of recorded triggers features

Feature	Definition
Area bbox	Area of the bounding box including the trigger.
Area convex	Area of the convex hull image, which is the smallest convex polygon that encloses the trigger.
Area filled	Area of the trigger with all the holes filled in.
Axis major length	The length of the major axis of the ellipse that has the same normalized second central moments as the trigger.
Axis minor length	The length of the minor axis of the ellipse that has the same normalized second central moments as the trigger.
Bandwidth	Extent of the trigger in frequency (Hz).
Delay	Time from the start of the spectrogram where the trigger is found (s).
Duration	Duration of the trigger (s).
Eccentricity	Eccentricity of the ellipse that has the same second-moments as the trigger.
Equivalent diameter area	Area of the circle that has the same area as the trigger.
Euler number	Euler number of the trigger computed as the number of connected components subtracted by the number of holes.
Extent	Ratio of pixels in the trigger to pixels in the bounding box.
Freq end	Maximum frequency of the trigger (Hz)
Freq peak	Frequency at the weighted centroid. The weighted centroid is evaluated as the centroid of the trigger pixels, weighted by their intensity.
Freq start	Minimum frequency of the trigger (Hz)
GPS start H1	GPS time at the start of the trigger in the H1 stream.
GPS start L1	GPS time at the start of the trigger in the L1 stream.
GPS end H1	GPS time at the end of the trigger in the H1 stream.
GPS end L1	GPS time at the end of the trigger in the L1 stream.

GPS peak H1	GPS time at the weighted centroid in the H1 stream.
GPS peak L1	GPS time at the weighted centroid in the L1 stream.
Intensity max	Maximum intensity of pixels in the trigger.
Intensity mean	Mean intensity of pixels in the trigger.
Intensity min	Minimum intensity of pixels in the trigger.
Num pixels	Number of pixels in the trigger
Orientation	Angle between the vertical and the major axis of the ellipse that has the same second moments as the trigger.
Perimeter	Perimeter of the trigger computed using a 4-connectivity.
Perimeter crofton	Perimeter of the trigger approximated with the Crofton formula.
Solidity	Ratio of pixels in the trigger to pixels of the convex hull image.

TABLE B.1: List of features recorded for each trigger detected in the time-frequency spectrograms. The features are listed in alphabetic order.

Appendix C

Examples of background triggers showing edge artefacts

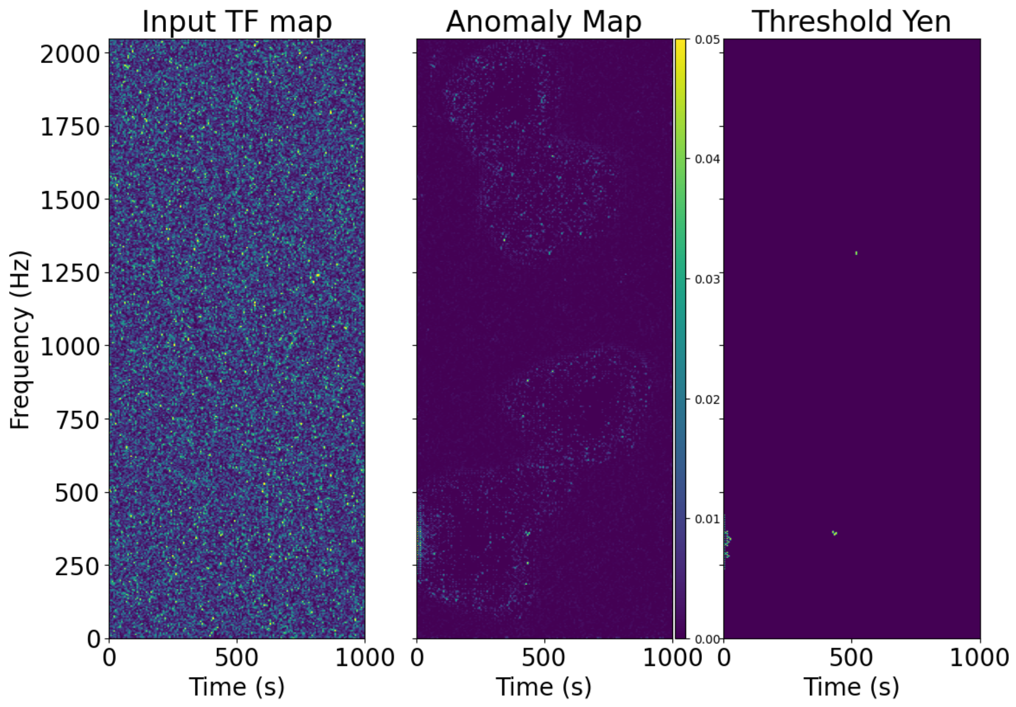

FIGURE C.1: Left: Input background TF map starting at GPS times 1262549618 and 1262488618 respectively for H1 and L1. Centre: Anomaly map, output of ALBUS. Right: Result of the Yen's threshold applied to the Anomaly map. ALBUS reveals a vertical artifact at the lower left edge of the TF map, further selected as a trigger with Yen's threshold.

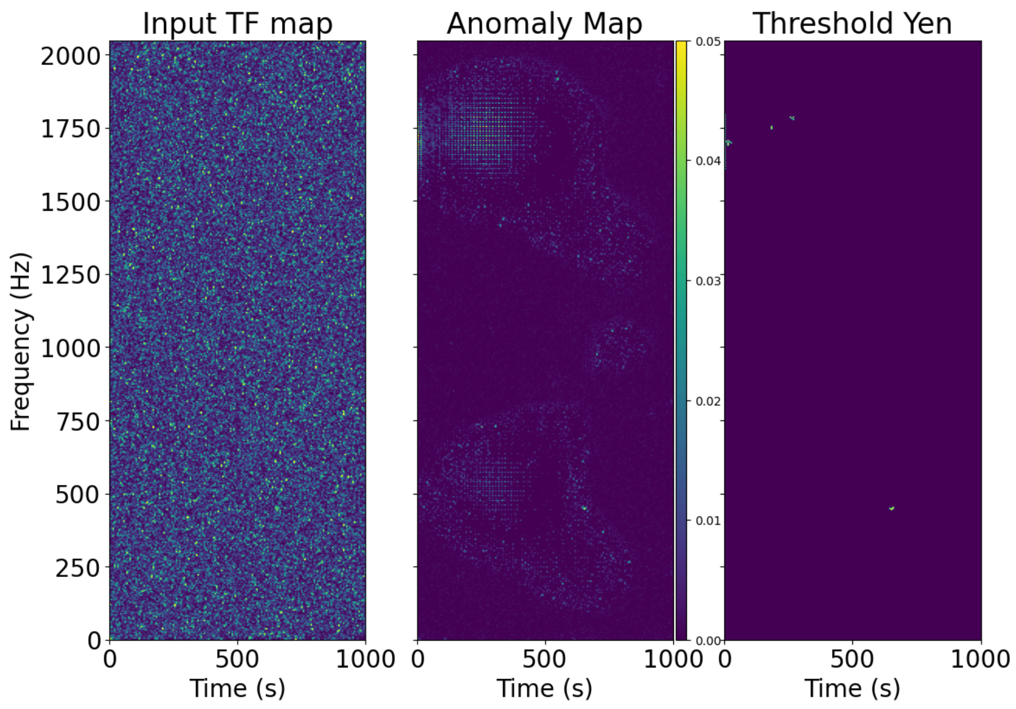

FIGURE C.2: Left: Input background TF map starting at GPS times 1263089618 and 1263001618 respectively for H1 and L1. Centre: Anomaly map, output of ALBUS. Right: Result of the Yen's threshold applied to the Anomaly map. ALBUS reveals a vertical artifact at the upper left edge of the TF map, further selected as a trigger with Yen's threshold.

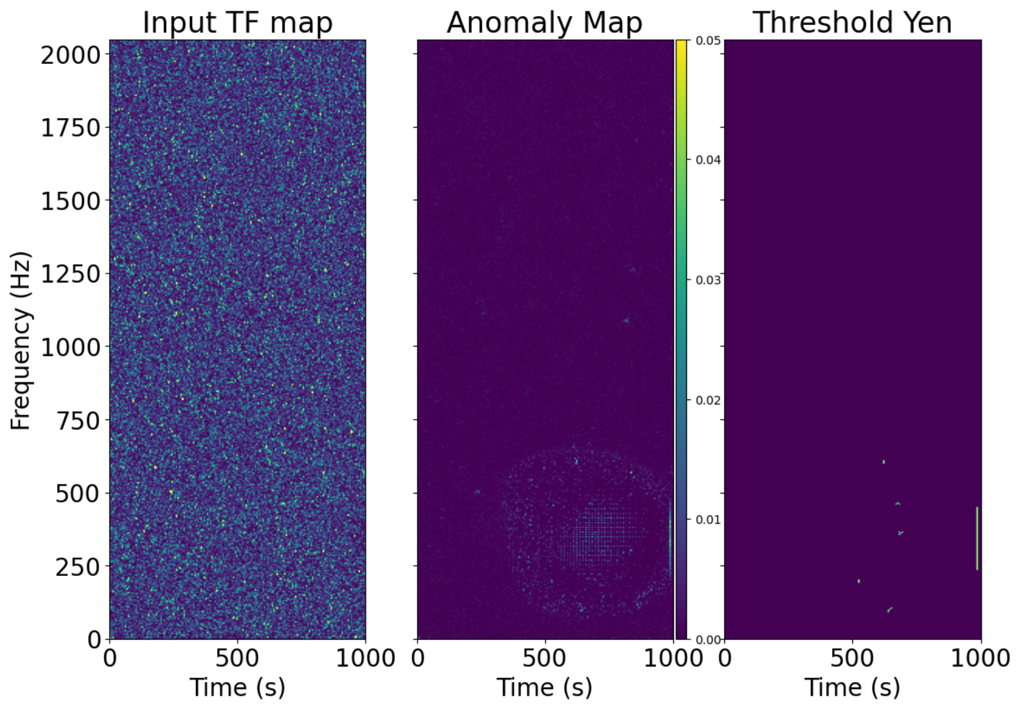

FIGURE C.3: Left: Input background TF map starting at GPS times 1264627618 and 1264448618 respectively for H1 and L1. Centre: Anomaly map, output of ALBUS. Right: Result of the Yen's threshold applied to the Anomaly map. ALBUS reveals a vertical artifact at the lower right edge of the TF map, further selected as a trigger with Yen's threshold.

Bibliography

[1] S. Weinberg. *Gravitation and cosmology*. John Wiley and Sons, New York, 1972.

[2] S. Carroll. *Spacetime and Geometry: An Introduction to General Relativity*. Benjamin Cummings, 2003.

[3] M. Maggiore. *Gravitational Waves. Vol. 1: Theory and Experiments*. Oxford University Press, 2007.

[4] A. Einstein. Über Gravitationswellen. *Sitzungsberichte der K&ouml;niglich Preussischen Akademie der Wissenschaften*, pages 154–167, January 1918.

[5] K. Glampedakis. Seminar on Gravitational Waves. Fisica del cosmos curso, Departamento de Fisica, Universidad de Murcia. 2014.

[6] M. Sieniawska and M. Bejger. Continuous Gravitational Waves from Neutron Stars: Current Status and Prospects. *Universe*, 5(11), 2019.

[7] G. Woan. Searches for continuous gravitational waves in the advanced detector era. Talk at CERN. September 2016.

[8] J. Aasi et al. Searching for stochastic gravitational waves using data from the two colocated LIGO Hanford detectors. *Phys. Rev. D*, 91:022003, January 2015.

[9] B. P. Abbott et al. Search for the isotropic stochastic background using data from Advanced LIGO's second observing run. *Phys. Rev. D*, 100(6):061101, September 2019.

[10] C. Caprini and D. G. Figueroa. Cosmological Backgrounds of Gravitational Waves. *Class. Quant. Grav.*, 35(16):163001, 2018.

[11] B. Allen. The Stochastic gravity wave background: Sources and detection. In *Les Houches School of Physics: Astrophysical Sources of Gravitational Radiation*, pages 373–417, April 1996.

[12] C. R. Nave. HyperPhysics: Light and Vision. Department of Physics and Astronomy, Georgia State University. http://hyperphysics.phy-astr.gsu.edu/hbase/phyopt/michel.html.

[13] B. Swinkels. Experimental detection of gravitational waves. Gravitational Waves course, Nikhef. April 2018.

[14] S. Hild. Large Interferometers for small displacements: A technological view of Gravitational Wave detection. OFS-20, Edinburgh, October 2009.

[15] S. Miyoki. Various Noise sources in GWDS. University of Toyama, September 2018.

[16] C. Bond, D. Brown, A. Freise, and K. Strain. Interferometer techniques for gravitational-wave detection. *Living Reviews in Relativity*, 19, February 2017.

[17] J. Ramirez. Optical Cavity Inference Techniques for Low Noise Interferometry. Final reports from LIGO SURF program, November 2019.

[18] C. Cahillane. Fabry-Perot Optical Cavity. https://ccahilla.github.io/fabryperot.html. April 2022.

[19] S. Rowan and J. Hough. The detection of gravitational waves. In *1998 European School of High-Energy Physics*, pages 301–311, 1998.

[20] L. Cominsky. LIGO: Detecting Gravitational Waves. Sonoma State University course. https://universe.sonoma.edu/moodle/course/.

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

[22] L. S. Finn and D. F. Chernoff. Observing binary inspiral in gravitational radiation: One interferometer. *Phys. Rev. D*, 47:2198–2219, March 1993.

[23] B. P. Abbott 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. *Phys. Rev. X*, 9:031040, September 2019.

[24] R. Abbott et al. GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo during the First Half of the Third Observing Run. *Phys. Rev. X*, 11(2), June 2021.

[25] R. Abbott et al. GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo During the Second Part of the Third Observing Run. November 2021.

[26] B. P. Abbott et al. GW170817: Observation of Gravitational Waves from a Binary Neutron Star Inspiral. *Phys. Rev. Lett.*, 119(16):161101, October 2017.

[27] B. P. Abbott et al. GW190425: Observation of a Compact Binary Coalescence with Total Mass $\sim 3.4\ M_\odot$. *Astrophys. J. Lett.*, 892(1):L3, March 2020.

[28] R. Abbott et al. Observation of Gravitational Waves from Two Neutron Star–Black Hole Coalescences. *Astrophys. J. Lett.*, 915(1):L5, 2021.

[29] J. Aasi et al. Advanced LIGO. *Class. Quant. Grav.*, 32(7):074001, March 2015.

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

[31] H. Miao. Quantum Techniques in Laser Interferometric Gravitational-wave Detectors. QSFP, Oxford University, 2018.

[32] D. Ganapathy. A Brief Introduction to Squeezing in GW detectors. Introductory squeezing talk at Christopher Newport University, 2022.

[33] J. Aasi et al. Enhanced sensitivity of the LIGO gravitational wave detector by using squeezed states of light. *Nature Photonics*, 7(8):613–619, July 2013.

[34] F. Acernese et al. Increasing the Astrophysical Reach of the Advanced Virgo Detector via the Application of Squeezed Vacuum States of Light. *Phys. Rev. Lett.*, 123:231108, December 2019.

[35] K. Craig. Coating Thermal Noise Research for Gravitational Wave Detectors. ICRR, University of Tokyo, April 2013.

[36] D. Davis et al. LIGO detector characterization in the second and third observing runs. *Class. Quant. Grav.*, 38(13):135014, June 2021.

[37] 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. *Phys. Rev. D*, 97:082002, April 2018.

[38] A. Staley, D. Martynov, R. Abbott, R. Adhikari, K. Arai, S. Ballmer, L. Barsotti, A. Brooks, R. DeRosa, S. Dwyer, A. Effler, M. Evans, P. Fritschel, V. Frolov, C. Gray, C. Guido, R. Gustafson, M. Heintze, D. Hoak, and C. Wipf. Achieving resonance in the Advanced LIGO gravitational-wave interferometer. *Class. Quant. Grav.*, 31, December 2014.

[39] A. Staley. Locking the Advanced LIGO Gravitational Wave Detector: with a focus on the Arm Length Stabilization Technique. *Columbia University thesis*, 2015.

[40] R. Abbott et al. Open data from the first and second observing runs of Advanced LIGO and Advanced Virgo. *SoftwareX*, 13:100658, 2021.

[41] S. Biscans, J. Warner, R. Mittleman, C. Buchanan, M. Coughlin, M. Evans, H. Gabbard, J. Harms, B. Lantz, N. Mukund Menon, A. Pele, C. Pezerat, P. Picart, H. Radkins, and T. Shaffer. Control strategy to limit duty cycle impact of earthquakes on the LIGO gravitational-wave detectors. *Class. Quant. Grav.*, 35, March 2018.

[42] N. Mukund Menon, B. O'Reilly, S. N. Somala, and S. Mitra. Effect of induced seismicity on advanced gravitational wave interferometers. *Class. Quant. Grav.*, 36, May 2019.

[43] B. P. Abbott et al. Prospects for observing and localizing gravitational-wave transients with Advanced LIGO, Advanced Virgo and KAGRA. *Living Rev. Relativ.*, 23(1), September 2020.

[44] C. Buchanan. The Advanced LIGO Detector and Seismic Noise. Presentation to Masters Committee, Louisiana State University, July 26, 2017.

[45] A. Biswas, J. McIver, and A. Mahabal. New methods to assess and improve LIGO detector duty cycle. *Class. Quant. Grav.*, 37(17):175008, August 2020.

[46] B P Abbott et al. A guide to LIGO–virgo detector noise and extraction of transient gravitational-wave signals. *Class. Quant. Grav.*, 37(5):055002, feb 2020.

[47] Craig Cahillane, Joe Betzwieser, Duncan A. Brown, Evan Goetz, Evan D. Hall, Kiwamu Izumi, Shivaraj Kandhasamy, Sudarshan Karki, Jeff S. Kissel, Greg Mendell, Richard L. Savage, Darkhan Tuyenbayev, Alex Urban, Aaron Viets, Madeline Wade, and Alan J. Weinstein. Calibration uncertainty for advanced LIGO's first and second observing runs. *Phys. Rev. D*, 96(10), November 2017.

[48] S. Karki, D. Tuyenbayev, S. Kandhasamy, B. P. Abbott, T. D. Abbott, E. H. Anders, J. Berliner, J. Betzwieser, C. Cahillane, L. Canete, C. Conley, H. P. Daveloza, N. De Lillo, J. R. Gleason, E. Goetz, K. Izumi, J. S. Kissel, G. Mendell, V. Quetschke, M. Rodruck, S. Sachdev, T. Sadecki, P. B. Schwinberg, A. Sottile, M. Wade, A. J. Weinstein, M. West, and R. L. Savage. The advanced LIGO photon calibrators. *Review of Scientific Instruments*, 87(11):114503, November 2016.

[49] J. C. Driggers et al. Improving astrophysical parameter estimation via offline noise subtraction for Advanced LIGO. *Phys. Rev. D*, 99:042001, February 2019.

[50] F Acernese et al. Calibration of advanced Virgo and reconstruction of the gravitational wave signal $h(t)$ during the observing run O2. *Class. Quant. Grav.*, 35(20):205004, September 2018.

[51] B. P. Abbott et al. LIGO: the Laser Interferometer Gravitational-Wave Observatory. *Reports on Progress in Physics*, 72(7):076901, June 2009.

[52] D. G. Keppel. Signatures and Dynamics of Compact Binary Coalescences and a Search in LIGO's S5 Data. *Dissertation (Ph.D.), California Institute of Technology*, 2009.

[53] G. A. Prodi. Gravitational Wave Transients from Compact Binary Coalescences. *Vulcano Workshop May 2018, Vulcano Island, Sicily, Italy*, 2018.

[54] V. Skliris, M. Norman, and P. Sutton. Real-Time Detection of Unmodeled Gravitational-Wave Transients Using Convolutional Neural Networks. 2020. arXiv:2009.14611.

[55] R. Abbott and others. All-sky search for long-duration gravitational-wave bursts in the third Advanced LIGO and Advanced Virgo run. *Phys. Rev. D*, 2021.

[56] M. Lopez, V. Boudart, K. Buijsman, A. Reza, and S. Caudill. Simulating transient noise bursts in LIGO with generative adversarial networks. *Phys. Rev. D*, 106:023027, July 2022.

[57] J. Glanzer, S. Banagiri, S. B. Coughlin, S Soni, M. Zevin, C. P. L. Berry, O. Patane, S. Bahaadini, N. Rohani, K. Crowston, V. Kalogera, C. Østerlund, L. Trouille, and A. Katsaggelos. Data quality up to the third observing run of advanced LIGO: Gravity Spy glitch classifications. *Class. Quant. Grav.*, 40(6):065004, February 2023.

[58] M. Zevin et al. Gravity Spy: integrating advanced LIGO detector characterization, machine learning, and citizen science. *Class. Quant. Grav.*, 34(6):064003, February 2017.

[59] N. J. Cornish and T. B. Littenberg. Bayeswave: Bayesian inference for gravitational wave bursts and instrument glitches. *Classical Quantum Gravity*, 32(13):135012, 2015.

[60] M. Lopez, V. Boudart, S. Schmidt, and S. Caudill. Simulating Transient Noise Bursts in LIGO with gengli, 2022.

[61] M. A. Bizouard. Gravitational wave burst searches. *Gravitational waves, Ecole de Physique des Houches, Session CX*, July 2018.

[62] E. E. Flanagan and S. A. Hughes. Measuring gravitational waves from binary black hole coalescences. II. The waves' information and its extraction, with and without templates. *Phys. Rev. D*, 57:4566–4587, April 1998.

[63] J. Neyman and E. S. Pearson. On the Problem of the Most Efficient Tests of Statistical Hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231:289–337, 1933.

[64] P. J. Sutton. Gravitational-Wave Burst Detection: Sources & Methods. *International School of Physics "Enrico Fermi", Varenna*, July 2017.

[65] C. Cutler and E. E Flanagan. Gravitational waves from merging compact binaries: How accurately can one extract the binary's parameters from the inspiral waveform? *Phys. Rev. D*, 49(6):2658, 1994.

[66] S. Klimenko, G. Vedovato, M. Drago, F. Salemi, V. Tiwari, G. A. Prodi, C. Lazzaro, K. Ackley, S. Tiwari, C. F. Da Silva, and G. Mitselmakher. Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors. *Phys. Rev. D*, 93:042004, February 2016.

[67] S. Klimenko, S. Mohanty, M. Rakhmanov, and G. Mitselmakher. Constraint likelihood analysis for a network of gravitational wave detectors. *Phys. Rev. D*, 72(12), December 2005.

[68] W. G. Anderson, P. R. Brady, J. D. E. Creighton, and E. A. Flanagan. Excess power statistic for detection of burst sources of gravitational radiation. *Phys. Rev. D*, 63(4), January 2001.

[69] V. Necula, S Klimenko, and G. Mitselmakher. Transient analysis with fast Wilson-Daubechies time-frequency transform. *Journal of Physics: Conference Series*, 363(1):012032, June 2012.

[70] A. Macquet, M. A. Bizouard, N. Christensen, and M. Coughlin. Long-duration transient gravitational-wave search pipeline. *Phys. Rev. D*, 104(10):102005, November 2021.

[71] M. Fays. Exploring Long Duration Gravitational-Wave Transients. *Cardiff University Thesis*, 2017. http://orca.cardiff.ac.uk/id/eprint/110245.

[72] L. Wen and B. F. Schutz. Coherent network detection of gravitational waves: the redundancy veto. *Class. Quant. Grav.*, 22(18):S1321–S1335, September 2005.

[73] E. Cuoco, G. Calamai, L. Fabbroni, G. Losurdo, M. Mazzoni, R. Stanga, and F. Vetrano. On-line power spectra identification and whitening for the noise in interferometric gravitational wave detectors. *Class. Quant. Grav.*, 18(9):1727–1751, April 2001.

[74] S. Dall'Osso, B. Giacomazzo, R. Perna, and L. Stella. Gravitational Waves from Massive Magnetars Formed in Binary Neutron Star Mergers. *Astrophys. J.*, 798(1):25, January 2015.

[75] P. Welch. The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, 15(2):70–73, 1967.

[76] M. Was, M. A. Bizouard, V. Brisson, F. Cavalier, M. Davier, P. Hello, N. Leroy, F. Robinet, and M. Vavoulidis. On the background estimation by time slides in a network of gravitational wave detectors. *Class. Quant. Grav.*, 27(1):015005, 2010.

[77] S. Klimenko, G. Vedovato, M. Drago, G. Mazzolo, G. Mitselmakher, C. Pankow, G. Prodi, V. Re, F. Salemi, and I. Yakushin. Localization of gravitational wave sources with networks of advanced detectors. *Phys. Rev. D*, 83(10), May 2011.

[78] A. Piro and E. Thrane. Gravitational waves from fallback accretion onto neutron stars. *Astrophys. J.*, 761:63, 2012.

[79] M. H. P. M. Van Putten. Directed searches for broadband extended gravitational-wave emission in nearby energetic core-collapse supernovae. *Astrophys. J.*, 819(2):169, Mar 2016.

[80] M. H. P. M. van Putten. Proposed source of gravitational radiation from a torus around a black hole. *Phys. Rev. Lett.*, 87:091101, 2001.

[81] M. H. P. M. Van Putten, A. Levinson, H. Lee, T. Regimbau, M. Punturo, and G. Harry. Gravitational radiation from gamma-ray bursts as observational opportunities for ligo and virgo. *Phys. Rev. D*, 69, 08 2003.

[82] A. Corsi and P. Meszaros. Gamma-ray burst afterglow plateaus and gravitational waves: multi-messenger signature of a millisecond magnetar? *Astrophys. J.*, 702:1171–1178, August 2009.

[83] E. A. Huerta, P. Kumar, B. Agarwal, D. George, H. Schive, H. P. Pfeiffer, R. Haas, W. Ren, T. Chu, M. Boyle, D. A. Hemberger, L. E. Kidder, M. A. Scheel, and B. Szilagyi. Complete waveform model for compact binaries on eccentric orbits. *Phys. Rev. D*, 95:024038, Jan 2017.

[84] P. J. Sutton. A Rule of Thumb for the Detectability of Gravitational-Wave Bursts, 2013.

[85] B. P. Abbott et al. Search for gravitational-wave bursts in the first year of the fifth LIGO science run. *Phys. Rev. D*, 80(10), November 2009.

[86] J. Abadie et al. All-sky search for gravitational-wave bursts in the first joint LIGO-GEO-Virgo run. *Phys. Rev. D*, 81(10), May 2010.

[87] A. Corsi and B. J. Owen. Maximum gravitational-wave energy emissible in magnetar flares. *Phys. Rev. D*, 83(10), May 2011.

[88] A. L. Piro and E. Pfahl. Fragmentation of Collapsar Disks and the Production of Gravitational Waves. *Astrophys. J.*, 658(2):1173, April 2007.

[89] P. J. Sutton. Gravitational-Wave Burst Detection: Sources. *Banach Center School of Gravitational Waves, Warsaw*, July 2013.

[90] B. P. Abbott et al. All-sky search for long-duration gravitational-wave transients in the second Advanced LIGO observing run. *Phys. Rev. D*, 99(10), May 2019.

[91] E. Thrane and M. Coughlin. Detecting gravitational-wave transients at 5σ: A hierarchical approach. *Phys. Rev. Lett.*, 115:181102, October 2015.

[92] T. Prestegard. Unmodeled searches for long-lasting gravitational-wave signals with ligo and studies of underground seismic noise for future gravitational-wave detectors. *University of Minnesota thesis*, 2016. https://hdl.handle.net/11299/182183.

[93] N.S. Saravana. How Deep Learning is Different from Machine Learning? November 2019. https://datawider.com/how-deep-learning-is-different-from-machine-learning/.

[94] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958.

[95] M. Arbib. *The Handbook of Brain Theory and Neural Network*, volume 26. January 2003.

[96] S. S. Haykin. *Neural networks and learning machines*. Pearson Education, Upper Saddle River, NJ, third edition, 2009.

[97] G. Louppe. Deep learning. *Faculty of Applied Sciences, University of Liège*, 2023.

[98] S. Shaw. Batch, Mini Batch and Stochastic gradient descent. *Medium*, August 2020. https://sweta-nit.medium.com/batch-mini-batch-and-stochastic-gradient-descent-e9bc4cacd461.

[99] K. Fukushima. Cognitron: A Self-Organizing Multilayered Neural Network. *Biol. Cybern.*, 20(3–4):121–136, sep 1975.

[100] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, 2013.

[101] D. Clevert, Unterthiner T., and S. Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[102] H. U. Dike, Y. Zhou, K. K. Deveerasetty, and Q. Wu. Unsupervised Learning Based On Artificial Neural Network: A Review. In *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, pages 322–327, 2018.

[103] P. J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73 – 101, 1964.

[104] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. Visualizing the Loss Landscape of Neural Nets. In *Neural Information Processing Systems*, 2018.

[105] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[106] G. Hinton, N. Srivastava, and K. Swersky. Coursera Neural Networks for Machine Learning. Lecture 6. 2018. http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

[107] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, 2010.

[108] K. He, X. Zhang, S. Ren, and J. Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[109] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

[110] M. Torres-Velázquez, W. Chen, X. Li, and A. McMillan. Application and Construction of Deep Learning Networks in Medical Imaging. *IEEE Transactions on Radiation and Plasma Medical Sciences*, pages 1–1, October 2020.

[111] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[112] A. Chazareix. How to optimize Convolutional Layer with Convolution Kernel. May 2023. https://www.sicara.fr/blog-technique/2019-10-31-convolutional-layer-convolution-kernel.

[113] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, Los Alamitos, CA, USA, June 2016. IEEE Computer Society.

[114] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

[115] C. Szegedy, L. Wei, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[116] S. Saha. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. December 2018. https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/.

[117] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[118] I. J. Goodfellow et al. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

[119] J. Pons, S. Pascual, G. Cengarle, and J. Serrà. Upsampling Artifacts in Neural Audio Synthesis. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3005–3009, 2021.

[120] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and Checkerboard Artifacts. *Distill*, 2016.

[121] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, 17(3):261–272, February 2020.

[122] J. F. Kaiser. Digital Filters. In F.F. Kuo and J.F. Kaiser, editors, *Systems Analysis by Digital Computer*, pages 218–285. John Wiley and Sons, New York, 1966.

[123] M. H. P. M. Van Putten. Ligo/virgo searches for gravitational radiation in hypernovae. *Astrophys. J.*, 575(2):L71–L74, August 2002.

[124] X. Zhu and X. Wu. Class Noise vs. Attribute Noise: A Quantitative Study. *Artificial Intelligence Review*, 22:177–210, 2004.

[125] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum Learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA, 2009. Association for Computing Machinery.

[126] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*. Springer, 2013.

[127] M. Tan and Q. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, June 2019.

[128] C. R. Harris et al. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[129] P. Barrett, J. Hunter, J. T. Miller, J. C. Hsu, and P. Greenfield. matplotlib – A Portable Python Plotting Package. In P. Shopbell, M. Britton, and R. Ebert, editors, *Astronomical Data Analysis Software and Systems XIV*, volume 347 of *Astronomical Society of the Pacific Conference Series*, page 91, December 2005.

[130] E. Cuoco et al. Enhancing gravitational-wave science with machine learning. *Machine Learning: Science and Technology*, 2(1):011002, December 2020.

[131] A. Menendez-Vazquez, M. Kolstein, M. Martinez, and L. M. Mir. Searches for compact binary coalescence events using neural networks in the LIGO/Virgo second observation period. *Phys. Rev. D*, 103(6):062004, March 2021.

[132] G Baltus, J. Janquart, M. Lopez, A. Reza, S. Caudill, and J. R. Cudell. Convolutional neural networks for the detection of the early inspiral of a gravitational-wave signal. *Phys. Rev. D*, 103(10):102003, 2021.

[133] P. Krastev. Real-time detection of gravitational waves from binary neutron stars using artificial neural networks. *Phys. Lett. B*, 803:135330, 2020.

[134] P. Krastev et al. Detection and parameter estimation of gravitational waves from binary neutron-star mergers in real LIGO data using deep learning. *Physics Letters B*, 815:136161, April 2021.

[135] J. McGinn et al. Generalised gravitational wave burst generation with generative adversarial networks. *Class. Quant. Grav.*, 38(15):155005, 2021.

[136] M. Lopez, I. Di Palma, M. Drago, P Cerda-Duran, and F. Ricci. Deep learning for core-collapse supernova detection. *Phys. Rev. D*, 103:063011, March 2021.

[137] A. Iess, E. Cuoco, F. Morawski, and J. Powell. Core-Collapse supernova gravitational-wave search and deep learning classification. *Machine Learning: Science and Technology*, 1(2):025014, June 2020.

[138] V. Boudart and M. Fays. Machine learning algorithm for minute-long burst searches. *Phys. Rev. D*, 105(8):083007, April 2022.

[139] F. Xing, Y. Xie, X. Shi, P. Chen, Z. Zhang, and L. Yang. Towards pixel-to-pixel deep nucleus detection in microscopy images. *BMC Bioinformatics*, 20:472, 2019.

[140] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[141] A. Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., Red Hook, NY, USA, 2019.

[142] S. Soni, C. P. L. Berry, S. B. Coughlin, M. Harandi, C. B. Jackson, K. Crowston, C. Østerlund, O. Patane, A. K. Katsaggelos, L. Trouille, V. G. Baranowski, W. F. Domainko, K. Kaminski, M. A. L. Rodriguez, U. Marciniak, P. Nauta, G. Niklasch, R. R. Rote, B. Téglás, C. Unsworth, and C. Zhang. Discovering features in gravitational-wave data through detector characterization, citizen science and machine learning. *Class. Quant. Grav.*, 38(19):195016, September 2021.

[143] B. P. Abbott et al. Effects of data quality vetoes on a search for compact binary coalescences in advanced LIGO's first observing run. *Class. Quant. Grav.*, 35(6):065010, February 2018.

[144] J. Glanzer, S. Banagiri, S. B. Coughlin, S. Soni, M. Zevin, C. P. L. Berry, O. Patane, S. Bahaadini, N. Rohani, K. Crowston, and C. Østerlund. Data quality up to the third observing run of Advanced LIGO: Gravity Spy glitch classifications, 2022.

[145] D. M. Macleod, J. S. Areeda, S. B. Coughlin, T. J. Massinger, and A. L. Urban. GWpy: A Python package for gravitational-wave astrophysics. *SoftwareX*, 13:100657, 2021.

[146] J.-C. Yen, F.-J. Chang, and S. Chang. A new criterion for automatic multilevel thresholding. *IEEE Transactions on Image Processing*, 4(3):370–378, 1995.

[147] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu. scikit-image: image processing in Python. *PeerJ*, 2:e453, June 2014.

[148] T. Chen and C. Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery.

[149] P. D. Lasky, C. Leris, A. Rowlinson, and K. Glampedakis. The Braking Index of Millisecond Magnetars. *Astrophys. J.*, 843(1):L1, June 2017.

[150] R. Coyne, A. Corsi, and B. J. Owen. Cross-correlation method for intermediate-duration gravitational wave searches associated with gamma-ray bursts. *Phys. Rev. D*, 93(10):104059, May 2016.

[151] F. Robinet, N. Arnaud, N. Leroy, A. Lundgren, D. Macleod, and J. McIver. Omicron: A tool to characterize transient noise in gravitational-wave detectors. *SoftwareX*, 12:100620, 2020.