SAPIENZA
UNIVERSITÀ DI ROMA

Continuous gravitational-wave signal analysis with General Purpose computing on Graphic Processing Units

Facoltà di Scienze matematiche, fisiche e naturali
Corso di laurea magistrale in Fisica

Candidato
Iuri La Rosa
n° matricola 1672124

Relatori
Cristiano Palomba
Pia Astone

A/A 2016/2017

Abstract

In this thesis I present a new approach to the search data analysis for continuous gravitational waves from asymmetrical rotating neutron stars, using the high parallel computing efficiency and computational power of the modern GPUs. Porting the Hough transform (core of the continuous waves analysis) on the high-level TensorFlow framework, I tested the new code on real data from the O2 run of the LIGO gravitational interferometers. It has been carried out an analysis directed on the Galactic Center region, where a large population of unseen neutron stars is expected. I performed the same procedures of an all-sky search with similar parameters and thresholds, and using a single GPU I obtained results with a factor 20 speed-up with respect to an analogue CPU multicore system. This demonstrate that the GPU programming with a general purpose high-level framework grants a significant improvement of the performance of the analysis done, opening new perspective on the search sensitivity on future analysis with a wider parameter space.

Contents

List of Figures

Introduction

In September 2015 the first gravitational-wave signal has been detected by the two LIGO gravitational interferometers. It was a transient signal from a binary black-hole coalescence, and since then other signals of the same class have been detected, with the birth of Multi-messenger astrophysics.

Another important class of signals are the continuous waves from compact stars which rotate asymmetrically with respect to the rotation axis. The search of such signals is even more challenging mainly because they are much fainter than the coalescences. Moreover, if the sky position of a sources is unknown, one has to face that the continuous nature of the signal means that the characteristic frequency is modulated by the Doppler effect due to the motion of Earth. Indeed, while the number of neutron stars observed by electromagnetic emission is lesser than three thousands, in our galaxy it is expected a population of about 10^9 unseen objects. Since, in principle, a fraction of them could emit gravitational waves within the sensitivity band of the detectors, the search for continuous waves is split in two main classes (plus some intermediate cases): coherent searches on known neutron stars and "blind" searches over the whole sky, where the parameters of the sources are unknown.

The main challenge in the all-sky search for continuous gravitational waves is the need, trying to extract a signal from the data, to face the extremely high computational cost necessary to inspect the source parameters. For this reason hierarchical analysis pipelines are developed, with the intent to reduce the computational cost trying to contain the search sensitivity loss. Nevertheless, the analysis still are very challenging from a computational point of view, and for this reason the fast evolution of the parallel computing on GPU appears promising, with devices more and more powerful and frameworks which show an increasing efficiency in the use of such devices.

The work I present in this Thesis is then organized as follows:

In chapter 1 the fundamentals of the theory of gravitational waves are summarized, explaining how the non-linear Einstein field equation is approximated using a perturbative approach: a variation of the field is seen as a little perturbation of the background metric and a tensorial linear field equation is extracted for the propagating wave. We'll see then how a gravitational wave induces a geodesic deviation, which results in a variation of the coordinates separation between two points in the manifold according to the wave polarization, suggesting how to detect a gravitational signal. Finally some classes of sources are treated, presenting the respective characteristic waveform and the amplitude we expect.

In chapter 2 we'll discuss the detection of gravitational waves. At first are presented how a gravitational interferometer works, referring to the current Earth based detectors, explaining how the sensitivity curves are formed. Subsequently we'll see the data analysis procedures used to see faint signals in a large noise. A particular focus is dedicated to the analysis pipeline used to the continuous waves search.

In chapter 3 I show the focus of my Thesis work, describing firstly the fundamentals of the GPU computing and presenting a very efficient high level framework, usable for a wide variety of purposes with the GPUs: TensorFlow. Afterward I explain in details the Frequency-Hough transform, which represents the heavier computational part of the continuous waves pipeline analysis. Indeed, i have used TensorFlow to port the Frequency-Hough transform on GPU. A full comparison between the original and the GPU implementation of the software has been done, showing a significant speed-up of this part of the analysis, around a factor of 20. To probe the actual parallelism efficiency of the GPU code i performed a series of benchmark varying the search parameters. Finally, to apply the developed code, I used the data from the O2 run of the LIGO gravitational interferometers to perform an analysis over an interesting region of the sky around the Galactic Center, where a large population of unseen neutron star is expected to exist. This is the first Galactic Center search done with LIGO O2 data and the resulting upper limits are significantly better then the past searches. With the reduced computational cost obtained, it is possible to expand the parameter space of the analysis, in order to increase the probability of a detection.

Chapter 1

Theory of gravitational waves

Gravitational waves are a phenomenon described by Albert Einstein within General Relativity theory (Einstein, 1916, Einstein, 1918) as perturbations of space-time generated by a non stationary gravitational field. According to Einstein's equations, the tensor that describes the gravitational field is the metric, then gravitational waves are metric waves: they propagate geometry variations in spacetime, and consequently the proper distance between spacetime points changes in time.

A way to describe the propagation of gravitational waves is the perturbative approach, which starts from a known exact solution of Einstein's equations. For this reason it is necessary a brief exposition of some useful elements in the theory of General Relativity (Ferrari and Gualtieri, 2014).

1.1 Riemann tensor

The Riemann tensor, called also Curvature tensor, describes the geometry of a curved manifold. A gravitational field is a metric that changes the *flat* geometry of the spacetime, expressed by the Minkowky metric, into a curved one. The Riemann tensor is then an useful tool to describe that curved geometry. To define it we need to know the equations of motion of a free particle in a gravitational field and how the concept of parallel transport changes in a curved geometry.

1.1.1 Geodesic equations

Geodesic equations describe the motion of a particle under the exclusive action of a gravitational field, when it is observed in an arbitrary reference frame.

For the sake of simplicity, we can write it as a consequence of the *equivalence principle* $m_I \propto m_G$. An equivalent statement of the principle is that in any point of the spacetime, subject to an arbitrary gravitational field, we can always define, at least locally, an inertial frame where physical laws take the same form prescribed by Special Relativity, i.e. the form in absence of gravity. This frame is called *locally inertial frame*, where the distance between two points must coincide with the distance in the Minkowky's metric:

$$ds^2 = \eta_{\mu\nu} dx^\mu_M dx^\nu_M; \quad \eta_{\mu\nu} = diag(-1, 1, 1, 1)$$

In a locally inertial frame the equations of motion of the particle, subject only to gravity, has to be $d^2x^\mu_M/d\tau^2 = 0$, with τ as the particle proper time.

Now let's move in a frame which coordinates are expressed by an arbitrary coordinate transformation in the Minkowsky metric: $x^\mu = x^\mu(x^\mu_M)$. In this frame the new distance can be expressed as

$$ds^2 = \eta_{\alpha\beta}\frac{\partial x^\alpha_M}{\partial x^\mu}dx^\mu\frac{\partial x^\beta_M}{\partial x^\nu}dx^\nu = g_{\mu\nu}dx^\mu dx^\nu,$$

being $g_{\mu\nu} = \partial_\mu x^\alpha_M \partial_\nu x^\beta_M \eta_{\alpha\beta}$ the new metric tensor, and the equations of motion of the particle, or *geodesic equations* become

$$\frac{d^2x^\alpha}{d\tau^2} + \Gamma^\alpha{}_{\mu\nu}\frac{dx^\mu}{d\tau}\frac{dx^\nu}{d\tau} = 0, \tag{1.1}$$

where

$$\Gamma^\alpha{}_{\mu\nu} = \frac{\partial x^\alpha}{\partial x^\lambda_M}\frac{\partial^2 x^\lambda_M}{\partial x^\mu \partial x^\nu}$$

are called *affine connections*.

The additional term in 1.1, proportional to $\Gamma^\alpha_{\mu\nu}$, expresses how the gravitational field acts on the particle motion, as seen by a frame different from the locally inertial one.

1.1.2 Covariant derivatives

Let us consider a vector $\mathbf{V} = V^\mu \mathbf{e}_{(\alpha)}$. Its derivative with respect to the coordinate x^ν is

$$\frac{\partial \mathbf{V}}{\partial x^\nu} = \frac{\partial V^\mu}{\partial x^\nu}\mathbf{e}_{(\mu)} + V^\mu\frac{\partial \mathbf{e}_{(\mu)}}{\partial x^\nu}.$$

In a Minkowky spacetime we can impose that each basis vector in a point is equal to the same basis vector in any other point, then it will be $\partial_\nu \mathbf{e}_{(\mu)} = 0$.

In a general spacetime, for the equivalence principle we can choose in any point of the manifold a locally inertial frame, where the laws of physics are those of Special Relativity, like in a Minkowky spacetime. Then the basis vectors in that frame are constant.

Consider a certain point p of the manifold. When we make a coordinate transformation Λ in p, from a general reference frame to the locally inertial one, the old general basis vectors $\mathbf{e}_\alpha$ can be expressed in terms of the new Minkowskyan basis vectors $\mathbf{e}^M_{\alpha'}$, with the equation $\mathbf{e}_{(\alpha)} = \Lambda^{\alpha'}_\alpha \mathbf{e}^M_{(\alpha')}$. Since $\mathbf{e}^M_{(\alpha')}$ are constant, then $\partial_\beta \mathbf{e}_{(\alpha)} = (\partial_\beta \Lambda^{\alpha'}_\alpha)\mathbf{e}^M_{(\alpha')}$; since $\partial_\beta \mathbf{e}_{(\alpha)}$ is a vector, for any β, it can be shown that we can express it as a linear combination of the basis vectors with the affine connections as coefficients: $\partial_\beta \mathbf{e}_{(\alpha)} = \Gamma^\mu{}_{\alpha\beta}\mathbf{e}_{(\mu)}$. So $\Gamma^\mu{}_{\alpha\beta}$ expresses how the basis vector α changes, with respect to the change of the coordinate β in a general non-Minkowskian spacetime.

Then we can define the covariant derivatives of the components of the vector V:

$$\begin{aligned}\partial_\beta \mathbf{V} &= \partial_\beta V^\alpha \mathbf{e}_{(\alpha)} + V^\alpha \Gamma^\mu{}_{\alpha\beta}\mathbf{e}_{(\mu)} = \\ &= \left(\partial_\beta V^\alpha + V^\sigma \Gamma^\alpha{}_{\sigma\beta}\right)\mathbf{e}_{(\alpha)} = \\ &= D_\beta V^\alpha \cdot \mathbf{e}_{(\alpha)}\end{aligned} \tag{1.2}$$

with $D_\beta V^\alpha \coloneqq \partial_\beta V^\alpha + V^\sigma \Gamma^\alpha_{\sigma\beta}$ the covariant derivative of V^α, which are then the components of the vector $\partial_\beta \mathbf{V}$. Note that, since $\partial_\mu \mathbf{e}_M = 0$, if we are in a Minkowsky spacetime or in an inertial frame of a general spacetime, it will be $D_\beta V^\alpha \equiv \partial_\beta V^\alpha$ and then $\Gamma^\alpha_{\mu\nu} \equiv 0$.

If we have, for example, a (0 2) tensor $T_{\mu\nu}$, its covariant derivative will be $D_\alpha T_{\mu\nu} = \partial_\alpha T_{\mu\nu} - T_{\sigma\nu}\Gamma^\sigma{}_{\alpha\mu} - T_{\mu\sigma}\Gamma^\sigma{}_{\alpha\nu}$. It is useful to note that, since for the principle of equivalence it is always possible to chose a coordinate system where $g_{\mu\nu} \equiv \eta_{\mu\nu}$, then

$$D_\alpha g_{\mu\nu} = D_\alpha \eta_{\mu\nu} = \partial_\alpha \eta_{\mu\nu} - \eta_{\sigma\nu}\Gamma^\sigma{}_{\alpha\mu} - \eta_{\mu\sigma}\Gamma^\sigma{}_{\alpha\nu}.$$

But affine connections and the derivative of the metric tensor are always zero in a flat metric, so we have $D_\alpha g_{\mu\nu} = 0$. We can solve this equation with respect to $\Gamma^\alpha{}_{\mu\nu}$ to explicit the relation between affine connections and metric tensor:

$$\Gamma^\alpha{}_{\mu\nu} = \frac{g^{\sigma\alpha}}{2}\left(\partial_\nu g_{\sigma\mu} + \partial_\mu g_{\nu\sigma} - \partial_\sigma g_{\mu\nu}\right) \tag{1.3}$$

1.1.3 Parallel transport

Parallel transport is defined as a translation where, for each infinitesimal displacement, the displaced vector must be parallel to the original one, and must have the same length.

In a flat space the displaced vector keeps the same direction, any path chosen. Conversely in a curved geometry, e.g. on a sphere, when we parallel transport the vector, it must always be tangent to the sphere. This means that in a closed path a vector can change direction. Then on a curved manifold the parallel transport of a vector depends on the path along which is transported and it is impossible to define a globally parallel vector field.

To formalize this concept let us consider, in a locally inertial frame, a curve with a function of the proper time parameter τ, describing for the sake of simplicity the wordline of a massive particle. Let $\mathbf{V}$ be a vector field defined at every point of the curve and $\mathbf{U}$ the vector tangent to the curve with components $dx^\mu/d\tau$. If we translate $\mathbf{V}$ parallel to itself its components do not change ($dV^\mu/d\tau = 0$), but

$$\frac{dV^\mu}{d\tau} = \frac{\partial V^\mu}{\partial x^\nu_M}\frac{\partial x^\nu_M}{\partial \tau} = U^\nu \partial_\nu V^\mu$$

and since we are in a inertial frame it holds $U^\nu \partial_\nu V^\mu = U^\nu D_\nu V^\mu$ and we have the covariant equation $U^\nu D_\nu V^\mu = 0$. Covariant means that it holds in any reference frame and we can explicit the invariant expression of the parallel transport of a vector:

$$U^\beta D_\beta V^\alpha = \frac{dx^\beta}{d\tau}\left(\partial_\beta V^\alpha + \Gamma^\alpha{}_{\beta\nu V^\nu}\right) = \frac{dV^\alpha}{d\tau} + \Gamma^\alpha{}_{\beta\nu} V^\nu U^\beta = 0. \tag{1.4}$$

We can note that when we parallely transport V along a curve in flat space we have $dV^\mu/d\tau = 0$; instead in a curved space the components of the vector change, and the change is given by

$$\frac{dV^\mu}{d\tau} = -\Gamma^\mu{}_{\nu\alpha} V^\alpha U^\nu$$

Now let us express the geodesic equations using covariant derivatives. Consider a particle, subject at most to gravitational forces, with worldline $x^\mu(\tau)$ and four-velocity (i.e. tangent vector to the worldline) $U^\mu = dx^\mu/d\tau$. Then geodesic equations become

$$U^\alpha D_\alpha U^\mu = 0$$

In this form geodesic equation is the equation of the parallel transport of the tangent vector **U** along the geodesic. This means that geodesics are those curves which parallely transport their own tangent vectors.

1.1.4 Curvature tensor

For the equivalence principle we can always find a reference frame where at least locally $g_{\mu\nu} \to \eta_{\mu\nu}$ and $\Gamma^\alpha{}_{\mu\beta} = 0$. Consequently, since $\Gamma^\alpha{}_{\mu\beta}$ contains the first derivative of the metric tensor (Equation 1.3), the latter will vanish. In order to know if we are in the presence of a gravitational field, we need the second derivative $\partial_\alpha \partial_\beta g_{\mu\nu}$.

The Curvature tensor, or Riemann tensor, is defined by the formula

$$R^\mu{}_{\nu\alpha\beta} = \partial_\alpha \Gamma^\mu{}_{\nu\beta} - \partial_\beta \Gamma^\mu{}_{\nu\alpha} - \Gamma^\mu{}_{\rho\beta}\Gamma^\rho{}_{\nu\alpha} + \Gamma^\mu{}_{\rho\alpha}\Gamma^\rho{}_{\nu\beta} \tag{1.5}$$

Its definition comes from the idea that the information about the curvature of the space must be contained in the second derivatives of the metric, and therefore in the first derivative of the affine connections $\Gamma^\alpha{}_{\mu\beta}$.

Using the concept of parallel transport, R tells us how does a vector change due to the curvature of the spacetime, when it is parallely transported along a loop. Consider an infinitesimal rectangular closed loop between the four points

$$(x^1, x^2) = \begin{cases} (a^1, a^2) \\ (a^1 + \delta a^1, a^2) \\ (a^1, a^2 + \delta a^2) \\ (a^1 + \delta a^1, a^2 + \delta a^2) \end{cases}$$

with $\delta a^1, \delta a^2 \to 0$, and a generic vector **V**. When we transport it along the loop, variation of its components δV^μ will be

$$\delta V^\mu = \delta a^1 \delta a^2 \left(\partial_2 \Gamma^\mu{}_{\nu 1} - \partial_1 \Gamma^\mu{}_{\nu 2} - \Gamma^\mu{}_{\rho 1}\Gamma^\rho{}_{\nu 2} + \Gamma^\mu{}_{\rho 2}\Gamma^\rho{}_{\nu 1}\right) V^\nu.$$

In the above formula, δa^1 and δa^2 are respectively the non vanishing components of the displacement vectors $\delta\vec{x}_{(1)}$, $\delta\vec{x}_{(2)}$ along the direction of the basis vectors $\vec{e}_{(1)}$, $\vec{e}_{(2)}$. If we chose an orthonormal basis ($\vec{e}_{(1,2)} = \delta^\mu_{1,2}$) we can write the components $\delta x^\mu_{1,2} = \delta a_{1,2}\delta^\mu_{1,2}$ (δ^μ_ν is the Kronecker delta). More generally, the equations of δV^μ are:

$$\begin{aligned} \delta V^\mu &= \delta x^\beta_{(1)} \delta x^\alpha_{(2)} \left(\partial_\alpha \Gamma^\mu{}_{\nu\beta} - \partial_\beta \Gamma^\mu{}_{\nu\alpha} - \Gamma^\mu{}_{\beta\rho}\Gamma^\rho{}_{\nu\alpha} + \Gamma^\mu{}_{\alpha\rho}\Gamma^\rho{}_{\nu\beta}\right) V^\nu = \\ &= \delta x^\beta_{(1)} \delta x^\alpha_{(2)} R^\mu{}_{\nu\alpha\beta} V^\nu \end{aligned}$$

We can recap now some properties of the Curvature tensor useful to understand its meaning in the following sections of this chapter:

- it gives information on the curvature of the spacetime, telling us how the curvature affect the parallel transport of a four-vector;
- in flat spacetime $\delta V^\mu = 0$ along any closed loop (parallel transport doesn't change vectors in flat geometry), so $R^\mu{}_{\nu\alpha\beta} = 0$ in any reference frame;
- as we said in section 1.1.2, in locally inertial frame the affine connections will vanish, and if we use equation 1.3, Riemann tensor components will become

$$R^\mu{}_{\nu\alpha\beta} = \frac{1}{2} g^{\mu\rho} \left(\partial_\nu \partial_\alpha g_{\rho\beta} - \partial_\nu \partial_\beta g_{\rho\alpha} + \partial_\rho \partial_\beta g_{\nu\alpha} - \partial_\rho \partial_\alpha g_{\nu\beta} \right).$$

  Lowering the index μ:

$$R_{\mu\nu\alpha\beta} = g_{\mu\rho} R^\rho{}_{\nu\alpha\beta} = \partial_\nu \partial_\alpha g_{\mu\beta} - \partial_\nu \partial_\beta g_{\mu\alpha} + \partial_\mu \partial_\beta g_{\nu\alpha} - \partial_\mu \partial_\alpha g_{\nu\beta}.$$

  That means in a locally inertial frame the non linear part of the Riemann tensor will vanish;
- $R^\mu{}_{\nu\alpha\beta}$ is antisymmetric with respect to the exchange of β and α indices. In fact interchanging $\delta x_{(1)}$ and $\delta x_{(2)}$, δV^μ changes sign, because we change the direction of the loop, e.g. from "clockwise" to "counterclockwise". This means that the sign of Riemann tensor can be chosen arbitrarily. Similarly since $R_{\mu\nu\alpha\beta} = R_{\alpha\beta\mu\nu}$ then $R_{\mu\nu\alpha\beta} = -R_{\nu\mu\alpha\beta}$. Moreover, by symmetry properties, it can be shown that $R_{\mu\nu\alpha\beta} + R_{\mu\beta\nu\alpha} + R_{\mu\alpha\beta\nu} = 0$;
- Differentiating R in a locally inertial frame with respect to x^ρ one can obtain the tensor equation

$$D_\rho R_{\mu\nu\alpha\beta} + D_\beta R_{\mu\nu\rho\alpha} + D_\alpha R_{\mu\nu\beta\rho} = 0. \tag{1.6}$$

1.1.5 Geodesic deviation

By the Principle of equivalence, we can always choose a locally inertial frame where affine connections $\Gamma^\alpha{}_{\mu\nu}$ vanish and the metric is flat. Conversely, if the spacetime is flat we can always define a coordinate system which reproduces in a local frame the same effects of an arbitrary gravitational field. We can study the motion of a single particle and gather information about the fictitious gravitational field, but these measurements can't allow to know whether the force is real or not. This can be understood only by comparing the motion of close particles, or more precisely comparing the behavior of close geodesics.

Let us consider two particles moving freely in a gravitational field respectively along the geodesics $x^\mu(\tau)$ and $x^\mu(\tau) + \delta x^\mu(\tau)$. If we define a set of geodesics $x^\mu(\tau, p)$, with τ proper time or an affine parameter, and p the label of different geodesics, we can write $\delta x^\mu = \partial x^\mu / \partial p$. Defining the tangent vector to the geodesic $u^\mu = \partial x^\mu / \partial \tau$, it follows $\partial u^\mu / \partial p = \partial \delta x^\mu / \partial \tau$.

Using these relations, we can compute the covariant derivatives of $\mathbf{u}$ along the curve $\tau = const$, whose tangent vector is $\delta\mathbf{x}$:

$$\delta x^\mu D_\mu u^\alpha = \frac{\partial u^\alpha}{\partial p} + \Gamma^\alpha{}_{\mu\nu} u^\nu \delta x^\mu;$$

similarly, the covariant derivatives of $\delta\mathbf{x}$ along the curve $p = const$ (i.e. along the geodesic) are

$$u^\mu D_\mu \delta x^\alpha = \frac{\partial \delta x^\alpha}{\partial \tau} + \Gamma^\alpha{}_{\mu\nu} \delta x^\nu u^\mu.$$

If we use the second covariant derivatives of the vector $\delta\mathbf{x}$ along the curve $p = const$, writing its first derivatives as $D_\tau \delta x^\alpha = u^\mu D_\mu \delta x^\alpha$, we can obtain the equations of the *geodesic deviation*:

$$D_\tau^2 \delta x^\alpha = R^\alpha{}_{\beta\mu\nu} u^\beta u^\mu \delta x^\nu. \tag{1.7}$$

It shows that the relative acceleration of nearby particles moving along geodesics depends on the curvature tensor. Since $R^\alpha{}_{\beta\mu\nu}$ vanishes only when the gravitational field is zero or uniform *and* stationary, the geodesic deviation contains the information on the gravitational field in a given spacetime.

1.2 Einstein's field equations

The equations of gravitational field are more complicated than the equations of the electromagnetic field. The reason is that electromagnetic interaction is linear, gravitation is not. For example, electromagnetic waves comes from the motion of charged particles but the four-momentum isn't source for the field.

With gravitational force, since $E \propto m$, mass and energy can transform one into another, so they are manifestation of the same physical quantity. Therefore if the mass is source of gravitational field, so must be the energy: for example a mass distribution in motion can irradiate gravitational waves, which carry energy which is in turn source of gravitational field . This means that energy must appear in the field equations as a source too. The field equations then can't be linear.

However, for weak gravitational field and non-relativistic particles we shall require that the new equations must reduce to the Poisson equation $\nabla^2\Phi = 4\pi G\rho$, with ρ the mass-density distribution and Φ the Newtonian potential. Let's start then analyzing equations of the motion in the weak field limit.

1.2.1 Geodesics in weak field limit

Let us consider a non-relativistic particle in motion in a weak and stationary (but not uniform) gravitational field. Be τ/c the proper time. Since $v^i = d_t x^i \ll c$, then $d_\tau x^i \ll c d_\tau t = d_\tau x^0$. Then we can write geodesic equations for Γ^μ_{00} as

$$\frac{d^2 x^\mu}{d\tau^2} + \Gamma^\mu{}_{00} \left(\frac{dx^0}{d\tau}\right)^2 = 0.$$

The field is stationary, so $\partial_0 g_{0\nu} = 0$ and Γ^μ_{00} become

$$\Gamma^\mu_{00} = \frac{g^{\mu\nu}}{2}\left(2\partial_0 g_{0\nu} - \partial_\nu g_{00}\right) = -\frac{g^{\mu\nu}}{2}\partial_\nu g_{00}$$

We have assumed a weak field, then we can choose a coordinate system such that we can write the metric tensor as a small perturbation of the flat metric:

$g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}$, ($|h_{\mu\nu}| << 1$)[1]. Retaining only the terms up to first order in $h_{\mu\nu}$

$$\Gamma^{\mu}_{00} \sim -\frac{\eta^{\mu\nu}}{2}\partial_\nu h_{00}$$

and the geodesic equations take the form

$$\frac{d^2x^\mu}{d\tau^2} - \frac{\eta^{\mu\nu}}{2}\partial_\nu h_{00}\left(\frac{dx^0}{d\tau}\right)^2 = 0.$$

Since the field is stationary, then $\partial_0 h_{00} = 0$ and the temporal component of the geodesic equation become $d^2x^0/d\tau^2 = 0$[2]. Regarding the spatial part, we can rescale coordinates such that $cdt = d\tau \Rightarrow \tau = ct$ and the equations become $d^2\vec{x}/d\tau^2 = \nabla h_{00}/2$, where $\nabla = (\partial_1, \partial_2, \partial_3)$.

The corresponding Newtonian equation is $d^2\vec{x}/dt^2 = \nabla\Phi$. Then we have $h_{00} = -2\Phi/c^2$ and for a spherical field:

- $h_{00} = -2GM/c^2r \Rightarrow g_{00} = -(1 + 2\Phi/c^2)$
- being $T_{\mu\nu}$ the stress energy tensor, within the non-relativistic approximation it becomes $T_{00} \sim \rho c^2$, reproducing the Laplace equation and we can write $\nabla^2 g_{00} = -8\pi G/c^4 \cdot T_{00}$

The latter equation is a useful tool to build a Lorentz-invariant equation of the field: it shows the form that the equation should have and gives a relation that the field equation must satisfy in the weak field limit.

To build a covariant tensorial equation, we need then to construct a tensor $G_{\mu\nu}$ for arbitrary (non-stationary) energy-matter distribution described by $T_{\mu\nu}$, using only $g_{\mu\nu}$ and its derivatives, such that the field equations takes the form

$$G_{\mu\nu} = \frac{8\pi G}{c^4}T_{\mu\nu}. \tag{1.8}$$

1.2.2 The field equation

In order to explicit the expression of the field equation, $G_{\mu\nu}$ must satisfy a series of requisites:

- like $T_{\mu\nu}$ it must be a symmetric 2nd degree tensor and must satisfy a covariant conservation law $D_\mu G^{\mu\nu} = 0$;
- must contain only second order derivatives of the metric tensor, such that the scale invariance of the Laplace equation is preserved, so must be linear in second derivatives of g and must contain products of first derivatives up to second order;
- must reduce to $G_{00} \sim -\nabla^2 g_{00}$ in weak field limit.

[1] In this approximation we consider only first order terms, so we can raise or lower indices using the flat metric: e.g. $h^\mu{}_\nu = g^{\mu\rho}h_{\rho\nu} \sim \eta^{\mu\rho}h_{\rho\nu} + O(h^2_{\mu\nu})$.

[2] Remember that choosing $\nu = 0$, the only non-zero component of the flat metric is η^{00}.

We have already seen a tensor that gives information about the gravitational field and with the same linearity features with respect to $g_{\mu\nu}$, the Riemann tensor. But $R^{\alpha}{}_{\beta\mu\nu}$ is a (1 3) order tensor and doesn't satisfy the covariant conservation law. In order to obtain a 2nd order tensor from $R^{\mu}{}_{\nu\alpha\beta}$ we can contract it with the metric, defining the Ricci tensor

$$R_{\mu\nu} \coloneqq g^{\rho\sigma} R_{\rho\mu\sigma\nu} = R^{\rho}{}_{\mu\rho\nu}; \tag{1.9}$$

contracting again we can obtain a scalar, the *scalar curvature* $R = R^{\rho}{}_{\rho}$.

G is therefore defined as $G_{\mu\nu} \coloneqq C_1 R_{\mu\nu} + C_2 g_{\mu\nu} R$, which $C_{1,2}$ coefficients have to be determined. We have already satisfied linearity and symmetry conditions.

To satisfy the conservation condition we put $D_\mu G^{\mu\nu} = C_1 D_R^{\mu\nu} + C_2 g_{\mu\nu} D_\mu R = 0$[3] and using Bianchi identities we obtain that the condition is satisfied if $C_2/C_1 = -1/2$, so that $G_{\mu\nu}$ becomes

$$G_{\mu\nu} = C(R_{\mu\nu} - \frac{g_{\mu\nu}}{2} R)$$

We can find C using the weak field limit: since $g \simeq \eta$, we have $R_{ii} \simeq R/2$ and $G_{00} \simeq C 2 R_{00}$. Computing R_{00}, it must be $C = 1$ in order that G_{00} reduces to $-\nabla^2 g_{00}$. Therefore the Einstein field equations are those in Equation 1.8, with $G_{\mu\nu}$ called the Einstein tensor:

$$G_{\mu\nu} = R_{\mu\nu} - \frac{g_{\mu\nu}}{2} R \tag{1.10}$$

If we explicit $G_{\mu\nu}$ using only the Ricci tensor, Einstein's equations become

$$R_{\mu\nu} = \frac{8\pi G}{c^4} \left(T_{\mu\nu} - \frac{1}{2} g_{\mu\nu} T^{\rho}_{\rho} \right). \tag{1.11}$$

In vacuum $T_{\mu\nu} = 0 \Rightarrow R_{\mu\nu} = 0$[4].

1.2.3 Gauge

Einstein tensor has 10 independent components, then Einstein's equations are 10, one for each component. But these equations are not independent, because the Bianchi identities imply $D_\mu G^{\mu\nu} = 0$, which in turn implies 4 relations that G must satisfy. So we have only 6 independent equations with 10 unknown functions.

This means that we have 4 degrees of freedom, which derive from the freedom in the choice the coordinate system. This means that the solution $g_{\mu\nu}$ of the Einstein's equation isn't uniquely determined, but only up to an arbitrary coordinate transformation $x^{\mu'} = x^{\mu'}(x^{\mu})$. When we choose a gauge, the four degrees of freedom are bounded and the symmetry disappears. While in the electromagnetic interaction we have only one gauge degree of freedom, in gravity instead we can have 4 independent gauge choices.

The *harmonic gauge* is an useful example: $\Gamma^{\rho} = g^{\mu\nu} \Gamma^{\rho}{}_{\mu\nu} = 0$.

[3] Remember that $D_\mu g^{\mu\nu} = 0$, always.

[4] Note that while Ricci tensor vanishes in vacuum, Riemann doesn't, unless gravitational field is null, or constant and uniform.

1.3 Perturbative approach to gravitational waves

Be $\overline{g}_{\mu\nu}$ a (constant) known exact solution of Einstein's equation. Let us assume that at least in a certain reference frame exists a small perturbation $h_{\mu\nu}$ such that $|h_{\mu\nu}| << |\overline{g}_{\mu\nu}|$ for each $\mu,\ \nu$. So we have a metric tensor $g_{\mu\nu} = \overline{g}_{\mu\nu} + h_{\mu\nu}$. With this metric, the stress energy tensor too will be the sum of an unperturbed term and a perturbation: $T_{\mu\nu} = \overline{T}_{\mu\nu} + T^{pert}_{\mu\nu}$.

Computing $\Gamma^{\alpha}{}_{\mu\nu}$ for the perturbed metric, with terms up to order $O(h^2)$ we have

$$\Gamma^{\alpha}{}_{\mu\nu}(g) = \Gamma^{\alpha}{}_{\mu\nu}(\overline{g}) + \Gamma^{\alpha}{}_{\mu\nu}(h) + O(h^2),$$

with

$$\Gamma^{\alpha}{}_{\mu\nu} = \frac{1}{2}\overline{g}^{\sigma\alpha}\left(\partial_\nu g_{\sigma\mu} + \partial_\mu g_{\nu\sigma} - \partial_\sigma g_{\mu\nu}\right)$$

and, keeping only the terms at first order in h,

$$\begin{aligned}\Gamma^{\alpha}{}_{\mu\nu}(h) = &\frac{1}{2}\overline{g}^{\sigma\alpha}\left(\partial_\nu h_{\sigma\mu} + \partial_\mu h_{\nu\sigma} - \partial_\sigma h_{\mu\nu}\right) + \\ &+ \frac{1}{2}h^{\sigma\alpha}\left(\partial_\nu \overline{g}_{\sigma\mu} + \partial_\mu \overline{g}_{\nu\sigma} - \partial_\sigma \overline{g}_{\mu\nu}\right) + O(h^2)\end{aligned}$$

We assumed $\overline{g}$ to be an exact solution of Einstein's equation in vacuum: $R_{\mu\nu}(\overline{g}) = 8\pi G/c^4\left(\overline{T}_{\mu\nu} - \overline{g}_{\mu\nu}\overline{T}/2\right)$. Therefore when we substitute $\Gamma^{\alpha}{}_{\mu\nu}(g) = \Gamma^{\alpha}{}_{\mu\nu}(\overline{g}) + \Gamma^{\alpha}{}_{\mu\nu}(h)$ described above we have

$$R_{\mu\nu}(g) = R_{\mu\nu}(\overline{g}) + R_{\mu\nu}(h),$$

where $R_{\mu\nu}(\overline{g})$ is known and where the Ricci tensor for the perturbation, $R_{\mu\nu}(h)$, retaining only first order terms, reduces to

$$R_{\mu\nu}(h) = \frac{8\pi G}{c^4}(T^{pert}_{\mu\nu} + \frac{1}{2}g_{\mu\nu}T^{pert}). \tag{1.12}$$

These are the Einstein's equations for gravitational waves, described as small perturbations of the background metric. Note that they are linear in $h_{\mu\nu}$.

This approach works well in many situations, since gravitational waves are often very faint. Nevertheless, it can't fit in strong gravity conditions, when gravitational waves can't be treated as a small perturbation.

1.3.1 Propagation of metric as a wave

Let us consider a perturbed metric $g_{\mu\nu}$. Defining the unperturbed one $\overline{g}_{\mu\nu} \equiv \eta_{\mu\nu}$ and the small perturbation $h_{\mu\nu}$, we can write $g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}$. Since $\Gamma^{\alpha}{}_{\mu\nu}(\overline{g}_{\mu\nu} = \eta_{\mu\nu}) = 0$, the affine connections are

$$\Gamma^{\alpha}_{\mu\nu} = \frac{1}{2}\eta^{\alpha\rho}(\partial_\mu h_{\rho\nu} + \partial_\nu h_{\mu\rho} - \partial_\rho h_{\mu\nu}) + O(h^2).$$

The wave equations will be then:

$$\begin{aligned}R_{\mu\nu}(g) = R_{\mu\nu}(h) &= \partial_\alpha\Gamma^{\alpha}_{\mu\nu} - \partial_\nu\Gamma^{\alpha}_{\mu\alpha} = \\ &= -\frac{1}{2}\left[\Box_{flat}h_{\mu\nu} - \left(\partial_\lambda\partial_\mu h^{\lambda}_{\nu} + \partial_\nu\partial_\lambda h^{\lambda}_{\mu} - \partial_\mu\partial_\nu h^{\lambda}_{\lambda}\right)\right] = \\ &= \frac{16\pi G}{c^4}\left(T^{pert}_{\mu\nu} - \frac{\eta_{\mu\nu}}{2}T^{pert\,\lambda}_{\lambda}\right),\end{aligned} \tag{1.13}$$

where $\Box_{flat} = \eta^{\mu\nu}\partial_\mu\partial_\nu$ is the D'Alambertian in flat spacetime.

The solutions of this equations are not uniquely determined. By the gauge symmetry any coordinate transformation on a metric tensor, solution of the equations above, is still a solution. But since we are in the weak field limit (i.e. small perturbations of the flat metric), we can make only transformations which preserve the condition $|h'_{\mu\nu}| \sim |h_{\mu\nu}| \ll 1$.

It's convenient then to choose a coordinate system such that the harmonic gauge condition is satisfied ($g^{\mu\nu}\Gamma^{\rho}{}_{\mu\nu} = 0$). In this way the Einstein's equation for the gravitational waves can be simplified: the harmonic gauge condition is equivalent to the expression $\partial_\mu h^{\mu}{}_{\nu} = \partial_\nu h^{\mu}{}_{\mu}/2$, and if we impose it in this form in Equation 1.13, we'll get a relation in the form of a wave equation (to hereafter $T = T^{pert}$):

$$\Box_{flat} h_{\mu\nu} = -\frac{16\pi G}{c^4}(T_{\mu\nu} - \frac{\eta_{\mu\nu}}{2}T^{\lambda}_{\lambda}),$$

If we define the tensor $\tilde{h}_{\mu\nu} := h_{\mu\nu} - \eta_{\mu\nu}h^{\lambda}_{\lambda}$ the two conditions will become

$$\begin{cases} \Box_{flat}\tilde{h}_{\mu\nu} = -\frac{16\pi G}{c^4}T_{\mu\nu} \\ \partial_\mu \tilde{h}^{\mu}_{\nu} = 0 \end{cases} \tag{1.14}$$

and as outside the source (i.e. in vacuum) $T_{\mu\nu} = 0$, then $\Box_{flat}\tilde{h}_{\mu\nu} = 0$.

Since propagation of gravitational waves in vacuum is linear, it can be expressed as superposition of plane waves. The simplest solution is a monochromatic plane wave $\tilde{h}_{\mu\nu} = \Re[A_{\mu\nu}e^{ikx}]$, with the polarization tensor $A_{\mu\nu}$ and the wave vector $k = (\omega/c, \vec{k})$. In this form we can see that the harmonic gauge condition implies a transverse wave. In fact if we substitute this solution in the gauge condition we have

$$\eta^{\mu\alpha}\partial_\mu A_{\alpha\nu}e^{ikx} = 0 \Rightarrow \eta^{\mu\alpha}A_{\alpha\nu}k_\mu = 0 \Rightarrow k_\mu A^{\mu}_{\nu} = 0.$$

This means that the wave vector is orthogonal to the polarization tensor, then the wave is transverse.

If we instead substitute the solution in the wave equation we get that the wave vector k is a null vector

$$k_\mu k^\mu = 0 \Rightarrow \omega = ck_0 = c\sqrt{k_i k_i}.$$

Thus a perturbation of a flat spacetime propagate as a transverse wave travelling at speed of light. We have seen how Einstein's theory of gravity predicts the existence of gravitational waves.

1.3.2 TT-gauge

Consider a wave travelling on the x direction. Since gravitational waves are transverse, equation 1.14 will be $(\partial_t^2/c + \partial_x^2)\tilde{h}^{\mu}_{\nu} = 0$. This implies that, respectively for forward/backward waves, $\tilde{h}^{\mu}_{\nu} = \tilde{h}^{\mu}_{\nu}(\chi = t \mp x/c)$ and $\partial_\mu \tilde{h}^{\mu}_{\nu}(\chi) = 0$. If we explicit and compute the derivatives with respect to t and x we'll obtain $\tilde{h}^{t}_{\nu} = \mp\tilde{h}^{x}_{\nu}$.

Harmonic gauge condition is not sufficient to determinate the gauge uniquely. It can be shown that it's always possible to find an infinitesimal vector ϵ_μ, solution

of the wave equation, whose four components can be used to set to zero four components of $\tilde{h}_{\mu\nu}$:

$$\begin{cases} \tilde{h}^t_i = 0 \ \ (i = x, y, z) \\ \tilde{h}^y_y + \tilde{h}^z_z = 0. \end{cases}$$

But we said that components along propagation direction are, in absolute value, equal to the time components, so if the temporal components are 0 then the x components will be zero too. The surviving components will be then $\tilde{h}^y_y = -\tilde{h}^z_z$, and $\tilde{h}^y_z = \tilde{h}^z_y$ because the symmetry of the metric tensor. Moreover, in this gauge $\tilde{h}_{\mu\nu} = h_{\mu\nu}$, so we can write the metric tensor:

$$h_{\mu\nu} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & h_+ & h_\times \\ 0 & 0 & h_\times & -h_+ \end{pmatrix}$$

If we write $h_{\mu\nu}$ as a superposition of two plane waves, we'll get the two polarization tensors, which describe the two polarization states, called + and ×:

$$A^+_{\mu\nu} \propto \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & +1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \quad A^\times_{\mu\nu} \propto \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & +1 \\ 0 & 0 & +1 & 0 \end{pmatrix}$$

In conclusion, a gravitational wave has only two degrees of freedom, which correspond to the two possible polarization states. The gauge in which this is clearly visible is called *TT-gauge*, where TT refers to the fact that in this gauge the metric tensor is traceless and has non-zero components only in the transverse directions with respect the propagation direction.

1.3.3 Effects of GW on motion

For the equivalence principle, it's not possible to study the effect of a gravitational wave on the motion of a single particle (i.e. in a local frame attached to the particle).

Let's study then the relative motion of two particles induced by a gravitational wave. Initially they are at rest in neighbour positions with coordinates respectively x^μ_A and x^μ_B; let's assume that at time $t = 0$ they are reached by a plane gravitational wave in TT-gauge, propagating along the x axis (i.e. with non-zero components only on the y-z plane). The metric is

$$g_{\mu\nu}(t - x/c) = \eta_{\mu\nu} + h_{\mu\nu}(t - x/c)$$

Since $h^{TT}_{00} = 0$, for all t and x, then $g_{00} = \eta_{00} = -1$ and we can assume that both particles have proper time $\tau = x^0 = ct$. When the wave arrives, the metric changes, from the unperturbed η, with a perturbation h, and the proper distance $ds^2(t - x/c) = g_{\mu\nu}(t - x/c)dx^\mu dx^\nu$ will change as well.

The coordinate separation four-vector $\delta x^\mu = x^\mu_B - x^\mu_A$ satisfies the geodesic deviation equation

$$D_\tau^2 \delta x^\alpha = R^\alpha{}_{\beta\mu\nu} u^\beta u^\mu \delta x^\nu,$$

with $u_\mu = \partial x\mu/\partial\tau$ the tangent vector to the geodesic for a proper time τ.

Now, let us consider a local inertial frame with coordinates $\{x^{\mu'}\}$ centered on the geodesic of the particle A (i.e. $x^0 = \tau = ct_A$, $x^{i'}_A = 0$). In the neighbourhood of A the metric is $ds^2 = \eta_{\alpha'\beta'} dx^{\alpha'} dx^{\beta'} + O(|\delta x|^2)$, so the interval differs from an interval with Minkowky's metric by terms of order $|\delta x|^2$ [5]. So the spatial coordinates of B will be the spatial components of δx: $x^{i'}_B \approx \delta x^{i'}$. From now on, we'll stay in the local inertial frame attached to A, but we're dropping the colon $'$ symbol.

We can now compute the geodesic deviation at time $t = 0$: $d_t^2 \delta x^i = R^i_{00j} \delta x^j$. We're considering the gravitational wave as a small perturbation of the flat metric, then in the Riemann tensor every term with $\Gamma^\alpha{}_{\mu\beta}$ will vanish:

$$\begin{aligned} R^i_{00k} &= (\partial_0^2 h^i_k + \partial^i \partial_k h_{00} - \partial_0 \partial_k h^i_0 - \partial^i \partial_0 h_{0k})/2 = \\ &= \partial_0^2 h^i_k/2 = \eta^{ji} \partial_0^2 h_{jk}/2 \end{aligned}$$

At $t \leq 0$ the two particles are at rest with separation vector $\delta x^j = \overline{\delta x}^j = const$. When the wave arrives, the relative position will change to $\delta x^j(t) = \overline{\delta x}^j + \delta x^j_{t>0}(t)$, with $\delta x^j_{t>0}(t) << 1$ since h is a small perturbation.

The equation of the deviation will be then

$$d_t^2 \delta x^j_{t>0} = \frac{\eta^{ji}}{2} \partial_t^2 h^{TT}_{jk} \delta \overline{x}^k + O(h^2)$$

and integrating, it will become

$$\delta x^j \left(t - \frac{x}{c}\right) = \delta x^j_0 + \frac{1}{2} \eta^{ji} h^{TT}_{ik} \left(t - \frac{x}{c}\right) \delta \overline{x}^k \tag{1.15}$$

The wave travels along the x direction, then in TT-gauge we have $\delta x^1 = \overline{\delta x}^1 = const$, while the two other components will have a deviation

$$\begin{cases} \Delta \delta x^2 = (h_+ \overline{\delta x}^2 + h_\times \overline{\delta x}^3) \\ \Delta \delta x^3 = (-h_+ \overline{\delta x}^3 + h_\times \overline{\delta x}^2) \end{cases}$$

I.e. particles will be accelerate *respectively* only on the plane orthogonal to the propagation direction, as we already said.

If we impose that h is linearly polarized along $+$ or $\times$ polarization direction, then

$$h_{\binom{+}{\times}} = 2\Re \left[A_{\binom{+}{\times}} e^{i\omega\left(t - \frac{x}{c}\right)} \right] = 2A_{\binom{+}{\times}} \cdot cos\left(\omega \left(t - \frac{x}{c}\right)\right)$$

(assuming the amplitudes A as real functions). Then if we imagine a ring in vacuum (Figure 1.1), it will be distorted to form an ellipse, with major semiaxes along the axes or rotated of 45°, depending on if we have a $+$ or $\times$ polarized wave respectively.

[5] This means that in the region neighbor A, it will be $\left(\Gamma^{\alpha'}_{\mu'\nu'}\right)_A \approx 0$.

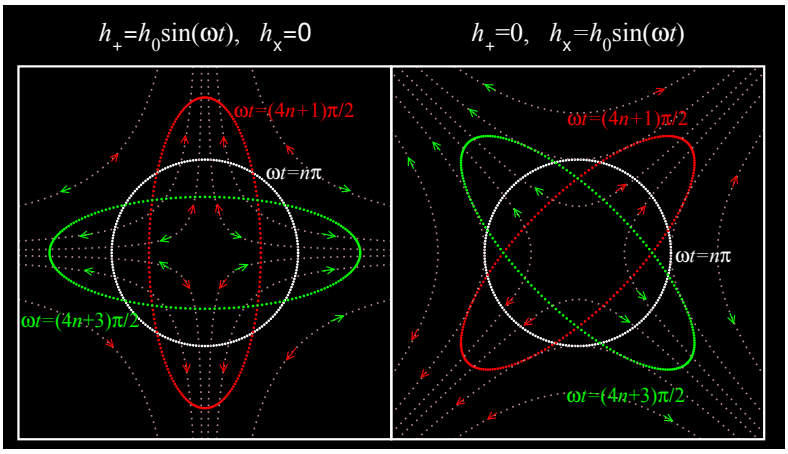

Figure 1.1: The effect of a linearly polarized TT-gauge gravitational waves, travelling along the x direction, on a ring in vacuum, as snapshots at four quarters of a period of the incoming wave. It can be seen the reason of the name of the two polarizations.

1.4 Quadrupole formalism

The quadrupole formalism is an useful approach to compute the waveform emitted by an evolving physical system, described by the stress-energy tensor.

We'll solve the equation of the wave with the harmonic gauge condition satisfied, under the slow-motion approximation[6]: the source is confined in a region much smaller than the wavelength of the emitted radiation. This means that $T_{\mu\nu} \neq 0$ in a region with spatial coordinates $|x^i| < \epsilon$, with ϵ such that $\lambda = 2\pi c/\omega \gg \epsilon$. This implies that $\epsilon\omega \ll c$, so we are considering processes with typical velocities much smaller than c.

The Fourier transforms of metric $\tilde{h}_{\mu\nu}$ and $T_{\mu\nu}$ tensor have both the form $f_{\mu\nu}(t,x) = \int_{-\infty}^{+\infty} f(\omega, x^i)_{\mu\nu} e^{-i\omega t} d\omega$. Therefore we can write equation 1.14 in frequency domain as

$$\left(\nabla^2 + \frac{\omega^2}{c^2}\right) \tilde{h}_{\mu\nu}(\omega, x^i) = -\frac{16\pi G}{c^4} T_{\mu\nu}(\omega, x^i) \tag{1.16}$$

Now we need to solve this equation inside and outside the source, imposing that the two solutions have to be the same at the boundary of the source.

Outer solution In vacuum, equation 1.16 is

$$\left(\nabla^2 + \frac{\omega^2}{c^2}\right) \tilde{h}_{\mu\nu}(\omega, x^i) = 0$$

and the simplest solution is a spherical wave with ingoing and outgoing terms:

$$\tilde{h}_{\mu\nu}(\omega, r) = \frac{1}{r} A_{\mu\nu}(\omega) e^{i\frac{\omega}{c}r} + \frac{1}{r} Z_{\mu\nu}(\omega) e^{-i\frac{\omega}{c}r}.$$

We are interested only in the wave emitted by the source, so we'll hold only the outgoing term in the equation and set to zero the other. We need to find the amplitude $A_{\mu\nu}$ solving the equation inside the source.

[6] Remember that we're still using the wave equation with the weak field approximation.

Inner solution Here $T_{\mu\nu} \neq 0$ and we'll consider the full wave equation 1.16. Integrating both members over the source volume, using the divergence theorem on $\nabla^2 h_{\mu\nu} = div(grad(h_{\mu\nu}))$, at first order approximation the equation gives the amplitude $A_{\mu\nu}$ as integral of $T_{\mu\nu}$ over the source volume:

$$A_{\mu\nu}(\omega) = \frac{4G}{c^4} \int_V T_{\mu\nu}(\omega, x^i) d^3x$$

If we join the two solutions we'll have respectively in frequency and time domain:

$$\begin{cases} \tilde{h}_{\mu\nu}(\omega, r) = \frac{4G}{c^4} \cdot \frac{1}{r} e^{i\frac{\omega}{c}r} \int_V T_{\mu\nu}(\omega, x^i) d^3x \\ \tilde{h}_{\mu\nu}(t, r) = \frac{4G}{c^4} \cdot \frac{1}{r} \int_V T_{\mu\nu}(t - \frac{r}{c}, x^i) d^3x \end{cases} \tag{1.17}$$

Note that the solution in time domain automatically satisfies the harmonic gauge condition, but we still have to project $\tilde{h}$ on the TT-gauge.

1.4.1 TT-gauge choice

First of all we notice that by conservation of four-momentum $\partial_0 \int_V T^{\mu 0} = 0 \Rightarrow P^\mu = \int_V T^{\mu 0} d^3x = const$. This means that, in the equation 1.17, $\tilde{h}^{\mu 0} = const$, and since we're interested in time variation of the field, we can impose that these components are $\tilde{h}^{\mu 0} = 0$[7].

Moreover, if we use the Virial Theorem, we can write the spatial components of the stress-energy tensor as

$$2 \int_V T^{ij} d^3x = \frac{1}{c^2} \partial_t^2 \int_V T^{00} x^i x^j d^3x,$$

where the right-hand side is the second time derivative of the quadrupole moment tensor of the system:

$$\int_V T^{ij}(t, x^k) d^3x = \frac{1}{2} d_t^2 q^{ij}(t)$$

To see the physical degrees of freedom, we can project the solution $\tilde{h}_{ij}$ ($\tilde{h}_{0\mu} = 0$) and the quadrupole moment q_{ij} to the TT-gauge with respect an observer, using the proper projector

$$\mathcal{P}_{ijkm} = P_{ik} P_{jm} - \frac{1}{2} P_{ij} P_{km},$$

where $P_{ij} = \delta_{ij} - n_i n_j$ with $\vec{n}$ the unit vector orthogonal to the wavefront. We can define then $h_{ij}^{TT} := \mathcal{P}_{ijkm} h_{km} = \mathcal{P}_{ijkm} \tilde{h}_{km}$ and $Q_{ij}^{TT} := \mathcal{P}_{ijkm} q_{km}$. The solution of the wave equation in TT-gauge, expressed with the quadrupole moment is:

$$\begin{cases} h_{\mu 0}^{TT} = 0 \\ h_{ij}^{TT}(t, r) = \frac{2G}{c^4 r} d_t^2 Q_{ij}^{TT}(t - r/c) \end{cases}$$

Indeed the gravitational radiation has a quadrupolar nature, and not dipolar. In fact an isolated system of masses, with a gravitational dipole momentum $\vec{d}_G =$

[7] Note that we obtained the same result of the TT-gauge for the temporal components, but from conservation principles prior to the gauge choice.

$\sum_i m_i \vec{r}_i$ must satisfy the law of conservation of the total momentum, so $d_t \vec{d}_G = \sum_i m_i d_t \vec{r}_i = const$: the second derivative will be 0 and the gravitational flux too.

The quadrupolar momentum instead can have non-zero time second derivatives. For a stationary and spherical symmetric distribution of energy-matter, quadrupolar momentum is constant, even if the distribution rotate around an axis (here cylindrical symmetry is enough). Therefore the time derivative will be zero with no gravitational flux. To produce gravitational wave we need then a certain amount of asymmetry, in fact we can consider the quantity $\ddot{Q}$ as a measure of the asymmetry of the source.

1.4.2 Gravitational flux

Let us consider the spatial components of the solution of the wave equation

$$h_{jk}^{TT} = \frac{2G}{c^4 r}\left[\ddot{Q}_{ij}^{TT}(t - r/c)\right].$$

Flux is defined as energy emitted for unit time and surface: $dL_{GW} = \frac{dE_{GW}}{dtdS}$.

We can compute it from the difference between the stress-energy tensor in a generic frame, $T^{\mu\nu}$, and the same tensor expressed in a local inertial frame. In the latter every first derivatives of the metric vanish, so it is possible to write it as a four-position derivative of a quantity $\upsilon^{\mu\nu\alpha}$, antisymmetric in μ and α: $T_{lif}^{\mu\nu} = \partial_\alpha \upsilon^{\mu\nu\alpha}$. We can define this difference as $\delta T^{\mu\nu} \coloneqq \partial_\alpha \upsilon^{\mu\nu\alpha} - T^{\mu\nu}$.

This difference is obviously zero in an inertial frame, moreover we can use the symmetry property of υ to demonstrate that the quantity $\partial_\mu\left(\delta T^{\mu\nu} + T^{\mu\nu}\right) = \partial_\mu\left(\partial_\alpha \upsilon^{\mu\nu\alpha}\right) = 0$. We have then a conservation law analogue to the four-momentum conservation. We can interpret then $\delta T^{\mu\nu}$ as the entity that contains information on the energy and momentum carried by the gravitational field.

It's possible to explicit the expression of $\delta T^{\mu\nu}$ and then to compute the gravitational luminosity emitted by a source integrating the flux over the whole solid angle:

$$\begin{aligned} L_{GW} &= \int \frac{dE_{GW}}{dtdS} dS = \\ &= \int c\delta T^{0k} r^2 d\Omega = \\ &= \frac{G}{2c^5}\frac{1}{4\pi}\int d\Omega \sum_{ij}\left(\dddot{Q}_{ij}^{TT}\right)^2. \end{aligned}$$

Solving the integral, the sum and the projectors will elide and we can get the expression

$$L_{GW} = \frac{G}{5c^5}\dddot{Q}_{ij}(t - r/c)\dddot{Q}_{ij}(t - r/c) \tag{1.18}$$

where $Q_{ij} \coloneqq q_{ij} - 1/3 \cdot \delta_{ij} q_k^k$ is the *reduced* quadrupole momentum, which is traceless by definition, then in TT-gauge $Q_{ij}^{TT} = \mathcal{P}_{ijkm} q_{km} = \mathcal{P}_{ijkm} Q_{km}$.

If we consider, as an example, an asymmetric rotating object, for the sake of simplicity an ellipsoid with major axis a (much greater than the two others axes),

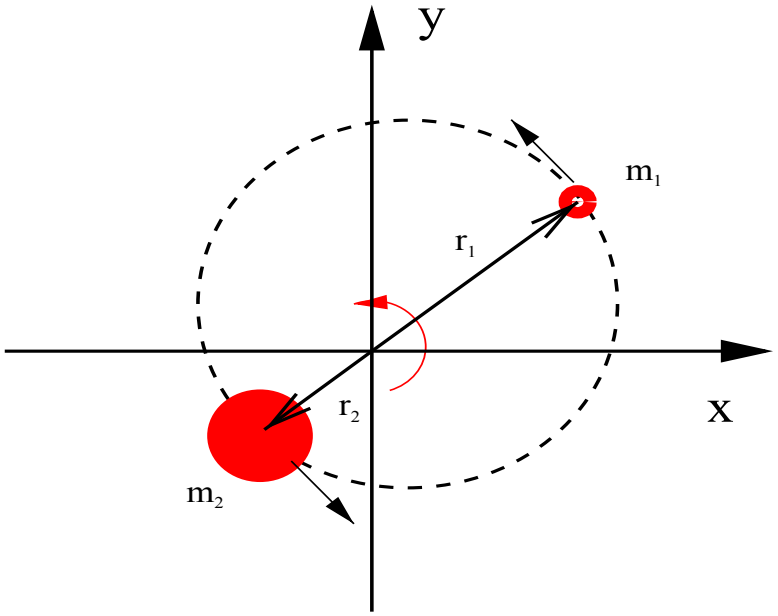

Figure 1.2: Scheme of two orbiting masses.

mass M rotating on the plane normal to a with pulsation ω, the gravitational luminosity will be

$$L_{GW} \approx \frac{G}{c^5} M^2 a^4 \omega^6.$$

Since the constant $G/c^5 \sim 10^{-53}$ is very low, it is immediately clear that an observable signal requires very high asymmetries, mass or frequencies. Moreover, only a small fraction of a signal can be seen by a detector, due to very low typical cross sections.

1.5 Sources

As we said before, to produce gravitational waves we need systems with big masses (on the order of some solar masses at least) rotating asymmetrically. Such systems are mainly binaries or non-axisymmetric rotating stars. Other sources can be asymmetrical supernovae core collapse, or phenomena occurred in the early stages of the Universe, just after the Big Bang. Despite this Thesis focuses on research of continuous signals from rotating stars, one can not avoid to mention the only kind of gravitational wave signals detected so far: the coalescence of black holes or neutron stars binaries.

1.5.1 Binary sistems

Let us consider a binary system composed of two stars, with mass respectively m_1 and m_2 and same radius R_*, orbiting together around the center of mass of the system. We can consider circular orbits with radius respectively r_1, r_2 and the two stars as point masses (it's a good approximation when $r \gg R_*$). Let us define:

- total mass $M = m_1 + m_2$ and reduced mass $\mu = m_1 m_2 / M$;
- proper length of orbital separation $l_0 = r_1 + r_2$ (then $r_{1,2} = m_{1,2} l_0 / M$);
- orbital pulsation (from Kepler's law) $\omega = \omega_K = \sqrt{GM/l^3}$;
- coordinates of the mass $m_{1,2}$ $(x_{1,2}, y_{1,2})$ (i.e. the motion is on the x-y plane) with $x_{1,2} = m_{1,2} l_0 / M \cdot cos(\omega t)$ and $y_{1,2} = m_{1,2} l_0 / M \cdot sin(\omega t)$);
- the 00-component of the stress-energy tensor in slow motion approximation

$$T^{00} = \sum_{n=1}^{2} m_n c^2 \delta(x - x_n) \delta(y - y_n) \delta z$$

The amplitude tensor then will be the rotation matrix in the x-y plane. In fact, if we compute the quadrupole moment tensor q_{ij}, the time varying part of the non vanishing components will be

$$\begin{cases} q_{xx} = -q_{yy} = \frac{\mu l_0^2}{2} cos(2\omega t) \\ q_{xy} = q_{yx} = \frac{\mu l_0^2}{2} sin(2\omega t) \end{cases} .$$

Note that the time variation of the quadrupole moment, then also of the gravitational-wave, is periodic with frequency twice the orbital frequency.

If we define the matrix

$$A_{ij} = \begin{pmatrix} cos(2\omega t) & sin(2\omega t) & 0 \\ sin(2\omega t) & -cos(2\omega t) & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

we can write $q_{ij} = \mu l_0^2 / 2 \cdot A_{ij}$ and, using $\omega = \omega_K$, in TT-gauge the gravitational wave will be

$$h_{ij}^{TT} = h_0(-A_{ij}^{TT}),$$

with $h_0 = \frac{4\mu M G^2}{r l_0 c^4}$ and A_{ij}^{TT} to be determined using the projectors $\mathcal{P}_{ijkm}$, depending on the inclination between the line of sight with unit vector $\hat{n}$ and the orbital plane.

For example, with a wave seen face-on, the two polarizations have the same squared module components, so we see a circularly polarized wave; conversely, a wave seen face off is linearly polarized transversely to the propagation direction. In both cases, as we can see in the arguments of the cosine and sine functions, the wave emitted has a frequency doubled with respect the orbital one.

The flux emitted is

$$L = \frac{32}{5} \frac{G}{c^5} \mu l_0^4 \omega^6$$

Substituting μ and ω we obtain $L \propto M^3 / l_0^5$. That is, more massive and closer systems will emit stronger signals.

Orbital evolution and coalescence

Even if the system doesn't radiate electro-magnetic waves, it loses energy by gravitational waves emission, so the orbit will become closer with time. In adiabatic approximation, that is with orbital parameters varying slowly enough, the orbital energy lost by the system will be $\dot{E}_{orb} = -L_{GW}$. With keplerian orbits $\omega = \omega_K$, $E_{orb} = K + U$ is

$$E_{orb} = \frac{1}{2}\frac{G\mu M}{l_0} - \frac{G\mu M}{l_0} = -\frac{1}{2}\frac{G\mu M}{l_0(t)}$$

$$\Rightarrow \dot{E}_{orb} = -E_{orb}\frac{d_t l_0}{l_0}$$

Since $\omega_K = \sqrt{GM/l_0^3} = 2\pi/P_K$ we have that the orbital period evolution, due to gravitational wave emission, is

$$\dot{P} = \frac{3}{2}\frac{P}{E_{orb}}L_{GW}$$

where P and E_{orb} are period and orbital energy measured.

From this equation we can obtain the time evolution of l_0 with the time derivative

$$\dot{l_0} = \frac{l_0}{E_{orb}}L_{GW};$$

integrating this equation over times, with $l_0(t=0) = \bar{l}_0$ we have

$$l_0(t) = \bar{l}_0^4\left(1 - \frac{t}{t_{coal}}\right), \tag{1.19}$$

with the coalescence time $t_{coal} = \frac{5}{256}\frac{c^5}{G^3}\frac{\bar{l}_0^4}{\mu M^2}$[8]

Waveform

We supposed an adiabatic regime; this means that, for both objects of the binary system, we can approximate every pseudo-orbit of the inspiral motion as a stationary circular orbit with $\omega = \omega_K$. Consider the binary system at a time $\bar{t}$ with $l_0 = \bar{l}_0$ and $\omega = \overline{\omega} = \omega_K(\overline{l_0})$. Using equation 1.19, the time evolution of ω_K, at a time t sufficiently small to preserve adiabatic approximation, is

$$\omega_K(t) = \sqrt{\frac{GM}{l_0^3(t)}} = \overline{\omega}\left(1 - \frac{t}{t_{coal}}\right)^{-\frac{3}{8}}. \tag{1.20}$$

Then, since the frequency of the emitted gravitational-wave is twice the orbital frequency, it will follow a time evolution $\nu_{GW}(t) = \omega_K(t)/\pi$.

[8] Remember that we have considered point masses. For real objects coalescence will start before t_{coal}. Moreover, approaching to the coalescence, slow motion and weak field approximations will fail.

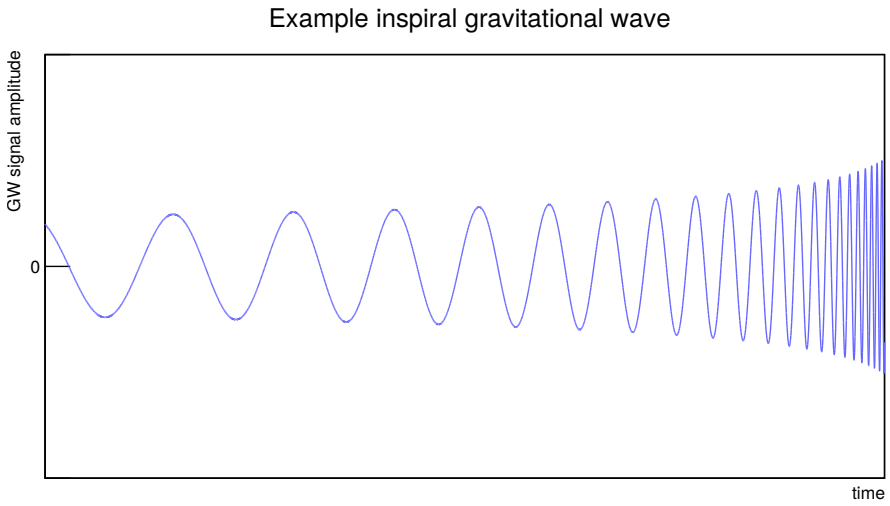

Figure 1.3: Example waveform of a gravitational wave signal emitted by a binary system.

Knowing the frequency temporal-function of the emitted signal, we can write its amplitude time evolution as

$$h_0(t) = \frac{4\mu MG^2}{rc^4}\frac{1}{l_0(t)} = \frac{4\pi\mu MG^2}{rc^4}\left(\frac{\pi\nu_{GW}^2(t)}{GM}\right)^{\frac{1}{3}} \tag{1.21}$$

Therefore the amplitude grows with ν_{GW} in time. Because this peculiar behavior, this waveform is called *chirp*. Rearranging the formula we can write

$$h_0(t) = \frac{4\pi^{2/3}G^{5/3}\mathcal{M}^{5/3}}{c^4 r}\nu_{GW}^{2/3}(t)$$

where $\mathcal{M} = \mu^{3/5}M^{2/5}$ is called *chirp mass*. It is an important parameter because it's the only direct measure of the mass of the system achievable using only the gravitational-wave signal detected.

1.5.2 Asymmetrical rotating star

As we said in chapter 1, emission of gravitational-waves from an object is intrinsically linked to its symmetry: the quadrupole moment, so the amplitude of the emitted wave, is much higher as much the object is far from symmetry in its time evolution. To describe how a single rotating object can emit gravitational waves, we can consider an ellipsoid with the following characteristics:

- equation
$$\left(\frac{x_1}{a}\right)^2 + \left(\frac{x_2}{b}\right)^2 + \left(\frac{x_3}{c}\right)^2 = 1$$
with semiaxes a, b, c;
- uniform density ρ and total mass M;
- inertia tensor
$$I_{ij} = \int_V \rho\left(r^2\delta_{ij} - x_i x_j\right)$$

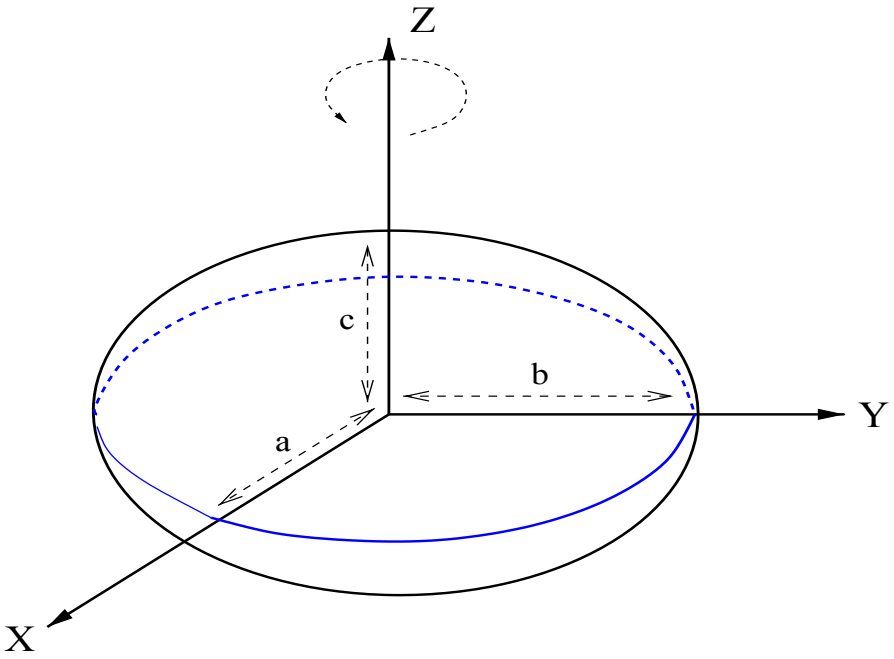

Figure 1.4: Scheme of a rotating ellipsoid.

- quadrupole moment

$$q_{ij} = \int_V \rho x_i x_j dx^3 = -I_{ij} + \delta_{ij} q \Rightarrow$$

$$Q_{ij} = q_{ij} - \frac{1}{3}\delta_{ij} q = -\left(I_{ij} - \frac{1}{3}\delta_{ij} I\right) \tag{1.22}$$

where $q = q^i_i$ and $I = I^i_i$

In the co-rotating frame

$$I'_{ij} = \frac{M}{5}\begin{pmatrix} b^2+c^2 & 0 & 0 \\ 0 & c^2+a^2 & 0 \\ 0 & 0 & a^2+b^2 \end{pmatrix} = \begin{pmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{pmatrix},$$

with $I_{1,2,3}$ the principal moments of inertia;

If we see the ellipsoid in an inertial frame where it rotates around one of its principal moments of inertia, e.g. I_3, with angular velocity $\vec{\omega} = (0, 0, \omega)$, its rotation matrix will be

$$R_{ij} = \begin{pmatrix} cos(\omega t) & -sin(\omega t) & 0 \\ sin(\omega t) & cos(\omega t) & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Then the I_{ij} tensor in the inertial frame is $I_{ij} = (RI'R^T)_{ij}$. Computing it and substituting its component in the equation 1.22 we can write the varying part of the quadrupole moment as

$$Q_{ij} = \frac{I_2 - I_1}{2} \begin{pmatrix} cos(2\omega t) & sin(2\omega t) & 0 \\ sin(2\omega t) & -cos(2\omega t) & 0 \\ 0 & 0 & 0 \end{pmatrix} = \frac{I_2 - I_1}{2} A_{ij}.$$

Note that again the wave frequency will be twice the rotational frequency of the star; moreover, if $I_1 = I_2$, i.e. $a = b$, the ellipsoid will preserve its axisymmetry around its rotation axis and won't emit any gravitational wave.

It's useful to express the quadrupole moment using the ellipticity

$$\epsilon := 2\frac{a-b}{a+b} \approx \frac{I_2 - I_1}{I_3}$$

$$\Rightarrow Q_{ij} = \frac{1}{2}\epsilon I_3 A_{ij};$$

Once again, if we project the wave in TT-gauge we'll have $h^{TT}(t,r)_{ij} = h_0(-A^{TT})_{ij}(t - r/c)$, where the result depends on the direction of the observer with respect to the rotating axis. The coefficient h_0 is

$$h_0 = \frac{4G\omega^2}{c^4 r} I_3 \epsilon \tag{1.23}$$

and the gravitational wave flux will be

$$L_{GW} = \frac{32G\omega^6}{5c^5}\epsilon^2 I^2,$$

If we suppose $a \sim b$, then $I \sim MR_*^2$ and $\omega \sim \omega_K$. Therefore $L \propto (M/R_*)^5\epsilon^2$. Thus we can say then that more massive, compact and *oblate* stars will emit sensibly stronger signals.

Waveform

In the non relativistic approximation the rotational energy of a rotating object is $E_{rot} = I\omega^2/2$ and its time derivative $\dot{E}_{rot} = I\omega\dot{\omega}$. Now let us suppose a stationary rotating compact star, which loses energy mainly via gravitational waves, then $L_{GW} \approx -\dot{E}_{rot}$ and the frequency will decrease in time. But if the frequency decreases, rotational energy will do too and with it the emitted gravitational flux. So the spin-down $\dot{\nu}$ of the star will fade in time.

If we explicit both members of the equation $L_{GW} \approx -\dot{E}_{rot}$ we get

$$\frac{32G}{5c^5}(2\pi\nu)^6\epsilon^2 I^2 \approx I 4\pi^2 \nu|\dot{\nu}|$$

$$\Rightarrow \epsilon \approx \left(\frac{5c^5}{512\pi^4 G}\frac{|\dot{\nu}|}{\nu^5 I}\right)^{\frac{1}{2}},$$

and the corresponding amplitude, using the above formula in equation 1.23, is

$$h_0(\nu) = \left(\frac{5G}{2c^3 r^2} I \frac{|\dot{\nu}|}{\nu}\right)^{\frac{1}{2}}. \tag{1.24}$$

We have then a relation between the dynamics parameters of a gravitational waves emitting system and its ellipticity [9].

For the well known pulsars like those in Crab and Vela nebulae we have upper limits on the amplitude $h_0 \approx 10^{-25} - 10^{-24}$ and on the ellipticity $\epsilon \approx 10^{-5} - 10^{-4}$ (Aasi et al., 2015b), and the last all-sky searches using the data from the O1 LIGO run gave similar limits (B. P. Abbott et al., 2017a).

Wobbling star

In principle it can be possible that the asymmetry of the rotating star don't lays on the plane orthogonal to the rotation axis. In this case we can define a *wobble* angle θ between the rotation axis and I_3. If we put, for the sake of simplicity, $I_1 = I_2$ and $\theta \ll 1$, the rotation matrix from the co-rotating frame to the inertial frame is

$$R_{ij} = \begin{pmatrix} cos(\omega t) & -sin(\omega t) & -\theta sin(\omega t) \\ sin(\omega t) & cos(\omega t) & \theta cos(\omega t) \\ 0 & -\theta & 1 \end{pmatrix}$$

and the quadrupole moment

$$Q_{ij} = (I_1 - I_3)\theta \begin{pmatrix} 0 & 0 & -sin(\omega t) \\ 0 & 0 & cos(\omega t) \\ -sin(\omega t) & cos(\omega t) & 0 \end{pmatrix}.$$

The wave amplitude will be

$$h_0 = \frac{2G\omega^2}{c^4 r}(I_1 - I_3)\theta$$

Note that a wobbling star emits gravitational wave at the rotation frequency (not doubled as the non-wobbling case).

1.5.3 Other sources

Another couple of classes of sources of gravitational waves have quite opposite characteristics: very fast bursts from a core collapse supernova and the continuous and stochastic cosmological background from the early stages of the Universe.

Bursts

In the definition of "burst" fall a high number of different phenomena that, in principle, can emit gravitational waves as long as they happen asymmetrically. An important class of sources of this kind are Type II supernovae, for which exists a large literature about modeling the core collapse and the subsequent explosion.

[9] If the compact star loses energy from any other sources, e.g. electromagnetic waves emitted by pulsating neutron stars, the ϵ estimation is an upper limit: in realistic cases the gravitational contribution at the spin-down is only a fraction of the total energy loss, the real ϵ will be lower than the value measured from the observed spin-down.

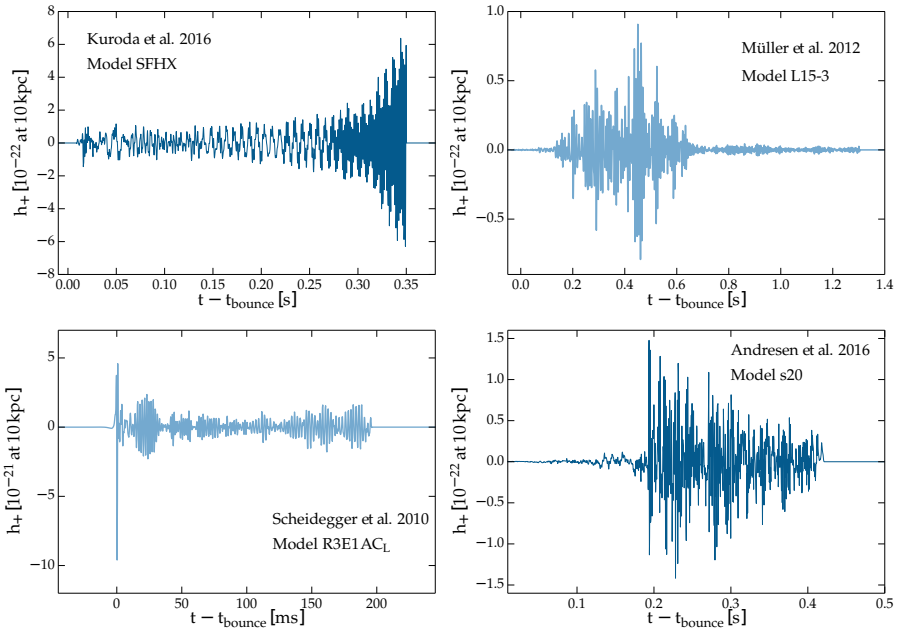

Figure 1.5: Some simulated waveform using four models of core collapse supernova, with a $10kpc$ far $15M_\odot$ progenitor star ($20M_\odot$ for the bottom right plot) (Powell, Szczepanczyk, and Heng, 2017).

Briefly, when the iron core of an old star, where no thermonuclear process can happen anymore, reaches a mass above the Chandrasekhar limit $1.4M_\odot$, the electron degeneracy pressure is no more sufficient to support the core against the gravitational attraction of the core itself. At this point it rapidly collapses until the density is enough to start a massive electron-capture process that turns the most part of the matter in the core in neutrons, which degeneracy pressure can sustain more mass than the electron one, unleashing a huge amount of neutrinos. Neutrinos interact with the outer layers of the core causing a big explosion that involves the rest of the dying star. At the center of the supernova, depending of the dynamics and masses involved, a neutron star or a black hole may form.

If the collapse is not spherically symmetric, the bounce subsequent to the collapse, due to the release of neutrinos, will have a high quadrupole moment, causing a burst of gravitational wave emission.

In figure 1.5 some example waveforms for some of the many models existing are shown, as summarized in Powell, Szczepanczyk, and Heng, 2017.

Cosmic background

A very interesting kind of sources has a cosmological nature. Similarly to the Cosmic Microwave Background, it is a incoherent superposition of waves from phenomena acting in the early moments of Universe. The detection of the cosmic gravitational stochastic background would have great consequences in our knowledge of the early Universe and its evolution.

The main reason is that particles (e.g. photons, gravitons) that decouple from the primordial plasma at a time t_{dec}, when the Universe had a temperature T_{dec}, give a snapshot of it at that time. The weaker is the fundamental interaction of the

particle, the higher was the energy scale when they lost the thermal equilibrium with the plasma, then earlier in the Universe life they decouple from the plasma. Since the gravitational interaction is far weaker than the others, detecting gravitational waves can probe very deeply in the primordial Universe: the graviton decouple before the Planck energy scale $E_P \approx 10^{19} GeV$, giving then informations on the physics at very high energies. Gravitational waves produced at that time will preserve typical frequencies, spectrum and intensity, giving informations on the production conditions.

The form of the spectrum of the cosmic gravitational-wave background is strongly model dependent, involving quantum gravity and beyond standard model physics, but from general considerations it can be modeled as a power-law $\propto \nu^{\alpha}$, with α between 0 and 3 and a cutoff at frequencies of the GHz order (Maggiore, 2000).

Chapter 2

Detecting gravitational waves

The hunt for gravitational waves detection was finally accomplished in September 2015, when for the first time the LIGO gravitational interferometers detected the waves generated by a binary black holes system coalescence (B. P. Abbott et al., 2016b). In the following months, other black hole coalescences were detected(B. P. Abbott et al., 2016a, B. P. Abbott et al., 2017b). In August 2017 the Virgo and LIGO interferometers had their first triple detection (B. P. Abbott et al., 2017c), bringing an impressive contribution to parameters and source sky position identification.

Just three days later (B. P. Abbott et al., 2017c) Virgo and LIGO detected for the first time a neutron stars coalescence, identified through the characteristic *chirp* signal in the gravitational interferometer. Around 1.7 seconds later a large number of telescopes and observatory covering the whole electromagnetic spectrum detected a gamma-ray burst and on a longer time scale the electromagnetic afterglow in the same region of the sky. Being the first GW observation which has been confirmed by non-gravitational means, it gave birth to the multi-messenger astrophysics.

Current gravitational waves detectors use interferometry to measure the geodesic deviation caused by a gravitational wave. For this reason we'll explain the fundamentals of how a gravitational interferometer works.

In Chapter 1 we saw the theoretical mechanisms that bring to the production of gravitational waves from some kind of sources. Now we're going to show the instrumental and analysis fundamentals of the gravitational wave research. When we'll have a clear picture of the problems we face, we can explain why is still such hard to find continuous waves, and how people search for a signal trying to overcome the higher difficulties of this kind of signals.

2.1 Interferometric GW detector

Since the fifties of the 20th century it was proposed and developed the use of light signals between two free masses in a gravitational field, able to characterize the metric and eventually its variation (Pirani, 1956). This idea has various applications such as pulsar timing, planetary ranging, spacecraft Doppler tracking and gravitational interferometers (Ricci, 2017).

The the first idea of interferometric detection of gravitational waves was due

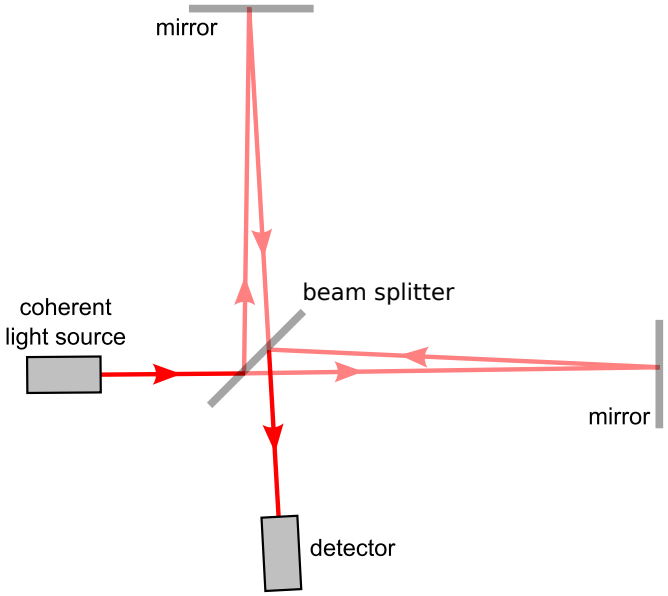

Figure 2.1: Scheme of a Michelson interferometer.

to the independent works of Weber, Forward and Gertenshtein, Pustovoit in the '60s (Wolf, 1998, Saulson, 2017,). In the '70s, Weiss (Weiss, 1972) treated the technical problems of the realization and operation of a gravitational interferometer; in the meantime, Forward was working independently to the development the first operating prototype (Forward, 1978), based on the Michelson configuration.

The gravitational interferometers Virgo and LIGO share many key features. Starting from the basic Michelson setup we'll show briefly how the modern gravitational interferometers work. For the specifications of the two detectors we'll refer to the respective literature (B. P. Abbott et al., 2009, Aasi et al., 2015a, Caron et al., 1997, Acernese et al., 2014).

2.1.1 Michelson interferometer

Let us consider a collimated, coherent, monochromatic electromagnetic source: a laser. In the Michelson configuration of an interferometer, the light beam is split by a semi-transparent mirror (the *beam splitter*) between two orthogonal arms, with mirrors at each end. The reflected light will come back to beam splitter, such that the two beams can join and interfere together, e.g. constructively or destructively. The final beam hits a detector, usually a photodiode, where the light power is measured (fig. 2.1).

The mirrors of the interferometer are the *test-masses*. Let us suppose that they are in *free fall*, i.e. the only force that acts on them is gravity and their motion is on geodesics. As we said in Section 1.1.5 we can't have informations on the variation of the metric using, for example, an accelerometer on a single mirror. Instead, if we use photons traveling between mirrors we can have informations on how the optical path changes by geodesic deviation: being the speed of light a constant in every frame, the time of flight will change if a gravitational wave crosses our interferometer.

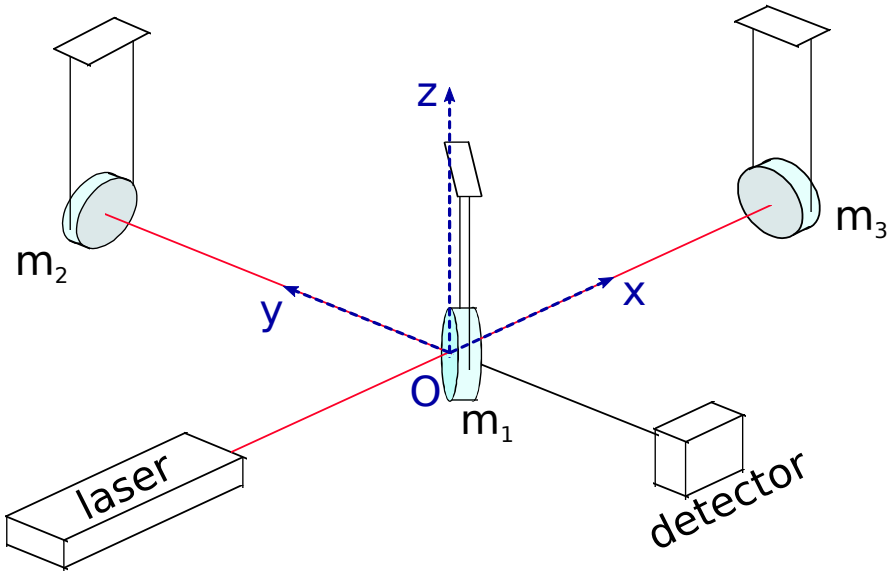

Figure 2.2: Earth based gravitational Michelson interferometer (Pitkin et al., 2011).

Let us consider a plane, linearly + polarized gravitational wave in TT-gauge[1] propagating along the z direction, and the arms of the interferometer with same length respectively on the x and y direction. The interval will be

$$ds^2 = (\eta_{\mu\nu} + h_{\mu\nu}(t))dx^\mu dx^\nu = c^2dt^2 - (1 + h(t))dx^2 - (1 - h(t))dy^2.$$

In a real interferometer on Earth the test-masses (our beam splitter and the two mirrors) can't be in free fall and they are hung to form pendulums. Using the geodesic deviation, the action of a gravitational wave on a test-mass n, as seen by an inertial frame, can be expressed as an acceleration field with respect to the center of mass of the system:

$$a_j^{(n)}(t, \mathbf{r}^{(n)}) = \frac{1}{2c^2}\partial_t^2 h_{jk} r_k^{(n)},$$

where $\mathbf{r}^{(n)} = (x^{(n)} - x_{CM}, y^{(n)} - y_{CM}, z^{(n)} - z_{CM})$ is the position of the test-mass n. If we set the reference frame like in Figure 2.2, with the origin on the beam splitter, we can write the equation of geodesic deviation separately for the three masses, using the relaxing time t_r and characteristic pulsation ω_p of the pendulums (we are supposing they have the same mass and same dissipation)

$$\begin{cases} \ddot{x}^{(n)} + \frac{\dot{x}^{(n)}}{t_r} + \omega_p^2 x^{(n)} = \frac{1}{2c^2}\left[\ddot{h}_{11}(x^{(n)} - x_{CM}) + \ddot{h}_{12}(y^{(n)} - y_{CM})\right] \\ \ddot{y}^{(n)} + \frac{\dot{y}^{(n)}}{t_r} + \omega_p^2 y^{(n)} = \frac{1}{2c^2}\left[\ddot{h}_{21}(x^{(n)} - x_{CM}) + \ddot{h}_{22}(y^{(n)} - y_{CM})\right] \end{cases}.$$

Supposing that the interferometer has both arms with length L, the length difference caused by a gravitational wave is the solution of the differential equation

$$d_t^2\Delta L + \frac{d_t\Delta L}{t_r} + \omega_p\Delta L = \frac{1}{c^2}d_t^2 h_{11} L \tag{2.1}$$

[1] In this section we'll set every time in TT-gauge, so the superscript "TT" will be omitted.

and depends on the functional form of the second time derivative of $h_{11}(t)$.

The phase variation due to a length difference of the arms ΔL is $\Delta\phi = \omega_\gamma \Delta L/c$, with $\omega_\gamma/2\pi$ the frequency of the light emitted by the laser. For a sinusoidal wave $h(t) = h_0 cos(\omega_G t)$ with amplitude h_0 and pulsation ω_G[2] the phase variation acquired by a photon, in a $2L$ long round trip from the beam splitter to the mirror and back, will be

$$\Delta\phi \approx \frac{2\omega_\gamma L}{c} h_0 cos(\omega_G t). \tag{2.2}$$

When a signal crosser the interferometer, the photodiode will see an oscillation of the light power at the same frequency of the gravitational wave, with an amplitude $\propto h_0$.

This is indeed the case of the Virgo and LIGO interferometers, with the prescription that between the laser and the beam splitter there is an *Electro-Optical Modulator*, which modulate the frequency of the light with a $\sim MHz$ carrier, such that the sidebands generated by the modulation couple with the gravitational signal crossing the interferometer. In this way the signal will stay, in the light spectrum of the beam, in two symmetrical modulation components aside from the carrier, and the latter is insensitive to the action of a gravitational wave.

With this configuration the output power is

$$W_{out} = W_{in} cos^2(\Delta\phi), \tag{2.3}$$

then the output of the detector is a direct measure of $h(t)$.

From equation 2.3 we can also see that the amount of power variation $dW_{out}/d\Delta L$ is function of $\Delta\phi$ and he maximum is reached when $W_{out} = W_{in}/2$.

This means that we have a maximum power variation when light interference is halfway from both constructive and destructive. This is called *half fringe* condition. However, we neglected phase differences caused by the noise: in real interferometers noises play an important role, and the half fringe is not the best condition, because despite it gives the highest signal output, it doesn't return the best signal to noise ratio.

As we said, when we modulate the light beam, only the two sidebands contain informations on the signal. Choosing an operating point such that, in absence of gravitational signal, the rejoining beams interfere perfectly destructively (*dark fringe*), we can exclude the carrier component and allow to pass one of the sidebands, excluding in this way a big part of the noise sources, except to the ones which couple with the allowed band similarly to the gravitational signal. For example, in addition to the laser frequency noise removal, we are less vulnerable to its power fluctuations, since in dark fringe the operating point neglects the power of the light source in no-signal conditions (Saulson, 2017).

We can interpret qualitatively the choice to work with a *null* instrument with the fact that it's easier to work with null measurements setting the no-signal state

[2]Note that are valid both the approximation of geometrical optics and small antenna limit. The first because the frequency of the laser is much higher than the gravitational wave frequency. The second is valid for the present day interferometers, for which the time of a complete travel of a photon in a $\sim km$ arm length is much smaller of the period of a gravitational wave. Both approximations simplify the response equation of the interferometer.

to $W = 0$ and waiting for a gravitational signal, instead to search a very small power variation in a large noisy optical power.

However, until now we considered ideal mirrors with a perfect reflectivity $R_{x,y}$. In real interferometers the mirrors have a contrast $C = 2R_xR_y/(R_x^2 + R_y^2) < 1$ and the best signal to noise ratio is achieved imposing in equation 2.3 $cos\phi \approx -1 + \sqrt{2(1-C)}$. Anyway, the mirrors have a high reflectivity, then C is near 1 and the interference is very close to the dark fringe.

To set an operating point it is sufficient to introduce a length difference in the arms of the interferometer, prior to the one induced by the gravitational wave, to obtain a $\Delta\phi_{op}$ such that W_{out} with no signal has the value we want.

2.1.2 Fabry-Perot Cavity

As we showed in Equation 2.2, the sensitivity of an interferometric detector depends on the length of the optical path. It's possible to increase this length putting a semi-transparent mirror between the terminal mirror and the beam splitter, such that the light beam travels many times along the arm, then is extracted and directed to the splitter where interfere with the light from the other arm. In this way we form a *Fabry-Perot* resonant cavity. In resonant conditions, the light inside a Fabry-Perot cavity with length L has n normal modes at frequencies $\nu_n = (n + 1/2)c/L$, and the difference between two consecutive resonance frequencies is defined *free spectral range* $\Delta\nu_{FSR} = c/2L$.

A parameter that strongly characterize a Fabry-Perot cavity is it's *finesse* F, defined as:

$$F = \pi \frac{\sqrt{R_1 R_2}}{(1 - R_1 R_2)}$$

($R_{1,2}$ are respectively the reflectivity of the entrance and terminal mirror of a single arm). It is a "quality" factor that says the precision to resolve his resonance lines. In fact, while with an input light at $\nu = \nu_n$ the electromagnetic field inside the cavity has certainly a maximum gain, the resonance peak can't be a *delta-function* because the imperfections of the mirrors. The width of the resonance peak is inverse-proportional to the finesse of the cavity. The full width at half maximum of the resonance peak is therefore defined as $\Delta\nu_{FWHM} = \Delta\nu_{FSR}/F$

An imperfection on the reflectivity of the mirrors means that there are losses which set a limit on maximum light power achievable in the cavity. From the particle point of view, it means that finesse defines how many times the cavity is able to bounce a photon between the two mirrors, before absorbing it in the system losses. The effective number of bounces will be $N_{eff} = 2F/\pi$ and the relative effective arm length $N_{eff}L$.

Knowing the effective path length of a photon in the cavity, we can define the storage time $\tau_s = N_{eff}L/c = 2FL/\pi c$. The inverse of the storage time is the frequency cut-off of the cavity $2\pi\omega_c = 1/\tau_s$. The expression of the effective cavity optical length is then function of the frequency of the incoming gravitational wave:

$$L_{opt} = \frac{2FL}{\pi} \frac{1}{\pi\sqrt{1 + (\omega_G/\omega_c)^2}}. \tag{2.4}$$

this means that with a Michelson-Fabry-Perot interferometer the phase variation depends on the signal frequency.

2.1.3 Noise

Since the amplitude of gravitational wave signals is very small, knowing and managing the instrumental noise is the key to detect them. We can classify the total noise of the interferometer in various components.

Quantum noise

The quantum noise is a source of error in the measured output optical power, which will give an uncertainty on the positions of the mirrors, and then on the gravitational waves amplitudes. It is generated by vacuum fluctuations of the electromagnetic field entering in the interferometer. These fluctuations interfere with the light circulating in the arm cavities and produce random phase differences on the photons: This generates the shot and radiation pressure noise. A complete quantum discussion is outside the purpose of this work, but we can expose an effective semi-classical approach which can describe correctly the expected quantum noise power spectrum of an interferometer.

Let us firstly consider the random fluctuation on the photon counts in the detector. It follows a Poisson statistics, with an average count n and the standard deviation $\sigma_n = \sqrt{n}$. It depends directly on the power of the light beam: the greater is the number of photons n, the smaller will be $\sqrt{n}/n = 1/\sqrt{n}$ and the signal to noise ratio will be better.

Let us suppose that the measure is done in a sampling time Δt , the output power is W_{out} and the laser pulsation ω_γ. The energy carried by a single photon is $\hbar\omega_\gamma$, then the energy associated to n photons in the detector will be $n\hbar\omega = W_{out}$, with a power standard deviation $\sigma_{W_{out}} = \sqrt{n}\hbar\omega$. The output counts standard deviation will be then

$$\sigma_n = \sqrt{n} = \sqrt{\frac{W_{out}\Delta t}{\hbar\omega_\gamma}}$$

From this relation we can derive the error σ_L on the length variation. From statistics, we know that $\sigma_{W_{out}} = dW_{out}/dL \cdot \sigma_L$, since we related $\sigma_{W_{out}}$ with σ_n, we have

$$\sigma_L = \frac{1}{\sqrt{n}} \cdot \frac{1}{W_{out}} \frac{dW_{out}}{dL}.$$

The error on W_{out} will give then an error on the length variation: the fluctuation on the number count of photons will produce a fluctuation on the measured position of the mirrors and therefore on the measured wave amplitude $\sigma_h = \sigma_L/L$. This is the so-called *shot noise*.

Let us suppose to be in half fringe, then $W_{out} = W_{in}/2$ and the shot noise amplitude spectral density will be

$$\sqrt{S_{h_{shot}}(\omega_G)} = \frac{c}{\omega_\gamma L}\sqrt{\frac{\hbar\omega_\gamma}{W_{in}}}, \tag{2.5}$$

with $S_{h_{shot}}$ the power spectral density. It is useful to note that, in a Michelson interferometer, the shot noise spectrum doesn't depend on the frequency.

As we said in section 2.1.2, with Fabry-Perot cavities in the arms, the equivalent strain spectrum given by the shot noise will be no longer a white noise, but it is function of ω_G; in particular the sensitivity falls down for $\omega_G \geq \omega_c$.

When the interferometer is in the dark fringe condition and the light is modulated in the way described in Section 2.1.1, the carrier reaches a minimum at the interferometer output and the light power reflected back toward the laser reaches a maximum. Inserting a mirror between the laser and the beam splitter, we can send back in the cavities the reflected light, building another Fabry-Perot cavity formed by the power recycling mirror and the rest of the interferometer. In this way we can amplify the light power circulating inside the interferometer, depending on how many times in average a photon can be recycled before it will be absorbed, and the shot noise contribution will be smaller. Another optimization is to add another recycling mirror between the beam splitter and the photodiode. This is called *signal recycling* and it's purpose is to amplify the signal sidebands produced at the detector's output by the gravitational waves.

Nevertheless, adding light power in the interferometer has the drawback of an increase of the radiation pressure on the mirrors. In the semi-classical approach, we can describe the effect of the momentum transfer of the photons using the statistical fluctuation of the number of photons hitting the mirror in a certain time Δt.

When a photon bumps on the surface of the mirror, transferring a part of its momentum, induces oscillations on the pendulum. More power of the beam means more photons that will exert a higher radiation pressure, adding a fluctuation to the position of the mirrors. Considering, for the sake of simplicity, the standard Michelson configuration with perfect mirrors, the force due to a radiation pressure with power W is W/c. The fluctuation on the power will produce then a fluctuation of the force on the mirror, with standard deviation

$$\sigma_F = \frac{\sigma_W}{c} = \hbar\omega_\gamma \frac{\sigma_n}{\Delta t} = \frac{1}{c}\sqrt{\frac{\hbar\omega_\gamma W_{in}}{2\Delta t}}$$

and amplitude spectrum $\sqrt{S_F(\omega_G)} = \sqrt{\hbar\omega_\gamma W_{in}}/c$.

The mirror will be displaced by the radiation pressure force by an amount

$$x(\omega_G) = \frac{\sqrt{S_F(\omega_G)}}{M\omega_G^2} = \frac{1}{M\omega_G^2} \cdot \frac{1}{c}\sqrt{\hbar\omega_\gamma W_{in}}.$$

The respective fluctuation on the two arms are anti-correlated: if an arm has an additional photon, in the other arm there will be a missing photon. For this reason the length difference between the two arms will be twice $x(\omega_G)$. The radiation pressure noise spectrum is then

$$\sqrt{S_{h_{rad}}} = \frac{2x(\omega_G)}{L} = \frac{2}{MLc}\sqrt{\hbar\omega_\gamma W_{in}}\frac{1}{\omega_G^2} \tag{2.6}$$

A way to reduce radiation pressure noise without an increase of shot noise is surely to use mirrors with higher masses. Things are not so easy if we want to use

a light beam with less power: comparing equations 2.6 and 2.5 we can easily see that the power dependence of the two noises is exactly inverse.

Both sources are related to the quantum nature of electromagnetic waves, and the maximum precision achievable measuring the output light beam is bounded by the uncertainty principle. For this reason the two quantum noise sources are treated together as a common source of optical readout noise $S_{qn}(\omega_G) = S_{shot}(\omega_G) + S_{rad}(\omega_G)$.

For each frequency value, exists an optimal power such that $S_{shot} = S_{rad}$ and $\sqrt{S_{qn}}$ has a minimum. This is called *standard quantum limit*:

$$\sqrt{S_{QL}(\omega_G)} = \frac{2}{\omega_G L}\sqrt{\frac{\hbar}{M}}. \tag{2.7}$$

It is important to remind that, although the above results are correct, there are fundamental conceptual differences between the semi-classical approach and the formal quantum behavior.

For example, we considered a fluctuation on the number of photons in the entrance of the interferometer, and we assumed that the contributions of the noise from the two arms anti-correlated, obtaining a maximum in the quantum noise. In a complete quantum mechanical discussion, the vacuum fluctuations as already mentioned generate the shot and radiation pressure noise.

We can demonstrate that the only significant contribution to the quantum noise comes from the vacuum fluctuations at the interferometer output, which generate photons that enter in the arms through the beam splitter and interfere with the circulating beams, in fact the fluctuations at the interferometer input are correlated in the two arms and they cancel each other.

Furthermore, it has been shown there are no contribution from the quantization of the test-masses, or from the uncertainty principle associated with the test-mass state (Braginsky et al., 2003). This means that the quantum noise is associated only to the light and the uncertainty principle regards the amplitude-phase errors on the outcoming beam. Indeed, still in Braginsky et al., 2003 is demonstrated that as a complementary fact the output photon numbers detected by the photodiode can be thought as classical quantities, and then these outputs measurements can be returned back at the input expressing them as anti-correlated length variations of the interferometers arm. This is why the semi-classical approach is correct.

External sources

Basically, the main noise source outside the experimental apparatus is the seismic activity of Earth's crust.

The absolutely most significant source is the noise from the continuous micro-seismic activity. In a reasonable quiet site on Earth it follows a spectrum $\approx 10^{-7} - 10^{-6} \cdot \nu^{-2}\ (m/3000m) \cdot 1/\sqrt{Hz}$ in all spatial dimensions (Pitkin et al., 2011). This means that at $\sim 100Hz$ where LIGO and Virgo detectors have a quantum noise around $10^{-23}Hz$, the seismic noise is twelve orders of magnitude higher, tearing down any probability to detect gravitational waves. It is mandatory then to isolate the mirrors both from the horizontal and the vertical components of the seismic vibrations.

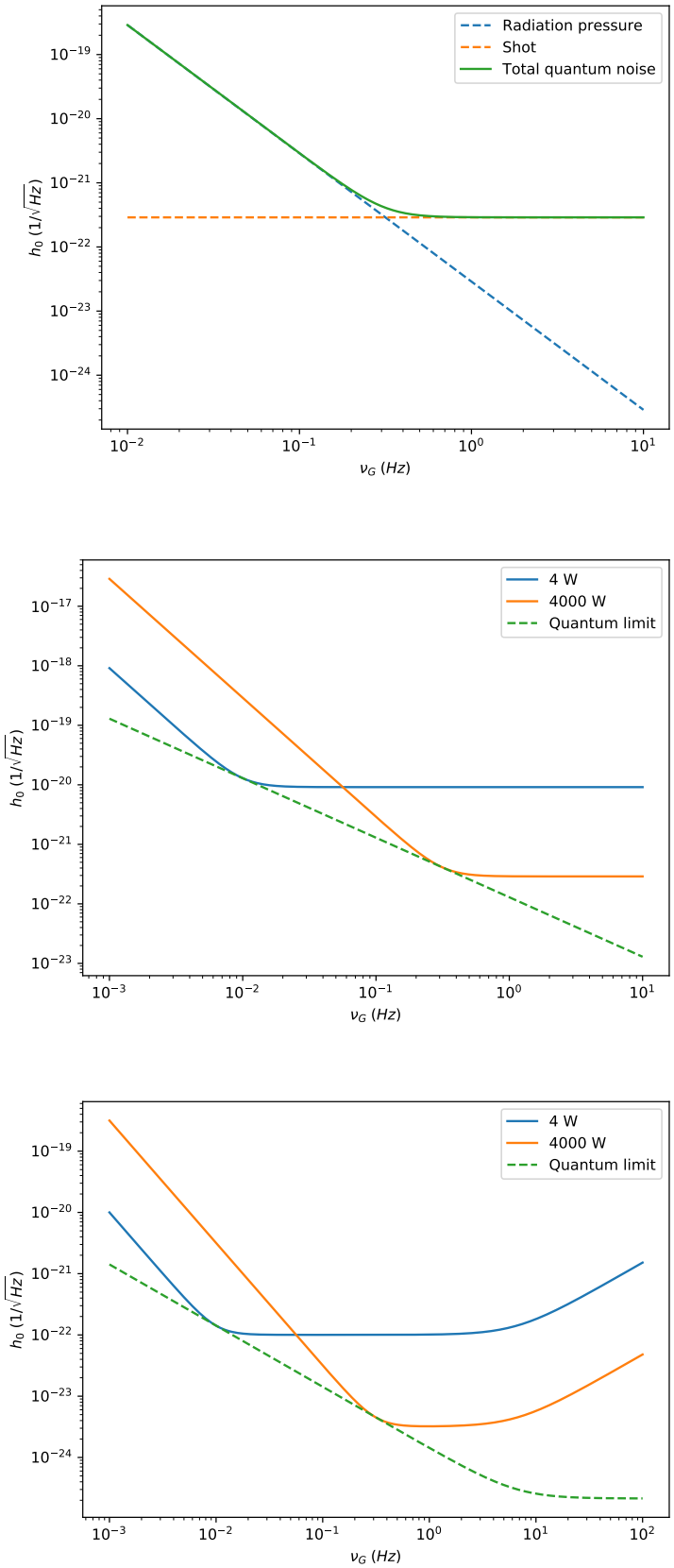

Figure 2.3: Plot of the quantum noise with various configurations of a Michelson interferometer with a $\lambda_\gamma = 1064\,nm$ laser and 40 Kg mirrors. Top: $L = 4000\,m$, $W = 4000\,W$; mid: $L = 4000\,m$, $W = 4,\ 4000\,W$; bottom: equipped with Fabry-Perot cavities with $F = 450$, the frequency cut on L_{opt} (Equation 2.4) increases the noise at higher frequencies.

The way to solve the problem is to build a chain of pendulums where the lower mass is connected to the higher mass with one or more thin strings attached to a spring. In this way each pendulum attenuate the horizontal vibration by a factor (ω_p/ω_G) and each spring attenuate the vertical vibration by a factor (ω_s/ω_G), where ω_p and ω_s are respectively the characteristic pulsation of the pendulum and the spring.

Another source of noise is the presence of seismic waves near the surface of the Earth, which cause density fluctuations of the Earth surface, producing gravity gradients changing in time: when a wave of gravity gradient passes nearby the test-masses of the interferometer, they experience a variation of gravitational attraction. It is then closely related to the seismic noise, but it is far more difficult to reduce, because it acts directly on the mirrors.

There are two ways to manage this noise: the first is to use an array of seismometers distributed around each test-mass, to monitor density variations expected to be coupled to local gravity fluctuations; the second is to build the detector underground, where the seismic surface waves cannot exist.

Thermal noise

Another important noise class is the one caused by the thermal fluctuations of the test-masses. Supposing that the masses and the last stage suspension pendulums has the same temperature T at thermal equilibrium, to each vibration mode of the mirror or oscillation mode of the pendulum is associated an energy fluctuation k_BT, with k_B the Boltzmann constant. Each of these modes can be seen as an armonic oscillator that causes an oscillation on the position on the mirror.

Considering the vibration mode i, being t_{r_i} its relaxing time, the power spectral density of the stochastic force that causes the fluctuation is $S_{F_i} = 2k_BTM_i/t_{r_i}$, where M_i is the effective mass of the harmonic oscillator that describes the mode. In turn, the spectral density of the displacement of the mirror will be the spectral density of the force, multiplied the squared absolute value of the associated harmonic oscillator transfer function:

$$S_x(\omega_G) = \frac{2k_BT}{t_{r_i}} \frac{1}{M_i} \frac{1}{\left(\omega_G^2 - \omega_i^2\right)^2 + \left(\omega_G/t_{r_i}\right)^2},$$

with ω_i the ith mode characteristic pulsation.

Considering the thermal noise of the pendulum, for a typical sensitive band of a ground based gravitational interferometer $\omega_G \gtrsim \omega_i$, with $\omega_i = \omega_p$, $t_{r_i} = t_r$ from Equation 2.1 and $M_i = M$ of the mirror. Then the associated amplitude spectrum has an upper limit

$$\sqrt{S_{h_{th}}^{pend}(\omega_G)} > \frac{1}{\omega_G^2}\sqrt{\frac{2k_BT}{M}\sum_i \frac{1}{t_r}} \tag{2.8}$$

and will affect mainly the low frequency part of the detector band.

The thermal vibration modes of the mirror instead have $\omega_G \lesssim \omega_i$ and we have

$$\sqrt{S_{h_{th}}^{mirr}(\omega_G)} = \frac{1}{L}\sqrt{2k_BT\sum_i \frac{1}{M_iQ_i\omega_i^3}} \tag{2.9}$$

where Q_i are quality factors that express the energy dissipation associated to the mode i, similarly to the relaxing time of a damped oscillator.

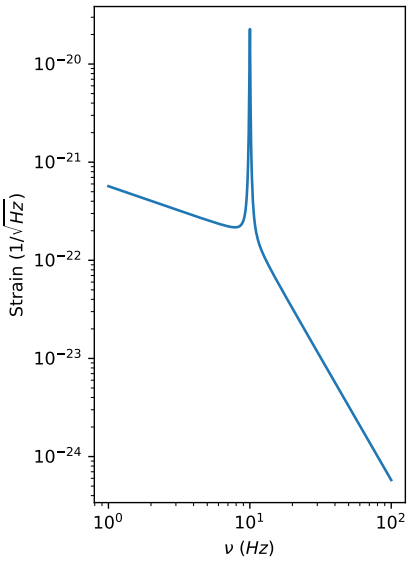

Figure 2.4: Plot of a thermal noise resonance with $\nu_i = 10\ Hz$, for a $M = 40\,Kg$ terminal mirror in an interferometer arm long $4km$, showing the different power-law dependence of the left and right tails.

From this formulae we can see that, to reduce the thermal noise, one can use more massive mirrors improving their inertia, cooling systems to reduce T, improving the quality factors of the test-mass system. The main reason of the latter is that, while we have assumed in Equation 2.9 that Q_i does not depend on frequencies, this is true only near the resonance frequencies: far from there $Q_i \propto \omega_G$, then the spectral amplitude of the mirror mode i will be $\sqrt{S_{h_i}} \propto \omega_G^{-1}$ for $\omega_G << \omega_i$, and for $\omega_G >> \omega_i$ we have $Q_i \propto \omega_G^5 \Rightarrow \sqrt{S_{h_i}} \propto \omega_G^{-5}$ (see Figure 2.4).

Looking left plot on Figure 2.5, it can be seen that at low frequency the most significant source of thermal noise are the suspensions of the mirrors. The thermal noise from the mirrors has a resonance above the operating range, so the left power-law tail affect the sensitivity of the detector in the most sensitive frequency band.

To achieve a noise thermal spectrum $\approx 10^{-23}\ 1/\sqrt{Hz}$ with a test-mass system at room temperature it should be $Q_i \approx 10^6$, and this requires a lot of effort. In particular, the thermal noise associated with the mirror coatings, required to have high reflectivity, limits the sensitivity of the present-day second generation detectors in the most sensitive frequency band. For this reason it's very promising the development of cryogenic techniques for the upcoming gravitational detectors (e.g. KAGRA, Somiya, 2012).

Setup imperfections

As for any real experiment, there are a number of uncertainties when we setup and use the instruments. There are three main sources of noise caused by these imperfections:

- the laser power fluctuations, as we already mentioned, give an additional term to the quantum noise, since the latter depends on the light power of the beam. As we said, to reduce the readout fluctuations the interferometer uses dark fringe and phase modulated beams ;

- misalignment, error on lengths of the cavities, differences in the geometry of the spherical mirrors (e.g. curvature radius), differences in the optical characteristics of the cavities (reflectivity, losses) will cause imperfection in the setup of the operating point, causing an imperfect best SNR and an amplification of fluctuations on the laser frequency;

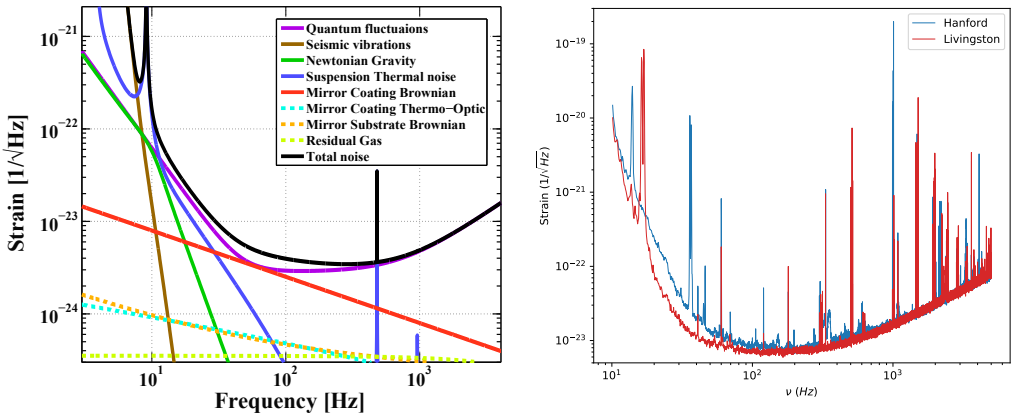

Figure 2.5: Left: recap of all the main noise sources (Aasi et al., 2015a). Right: the sensitivity curves of the LIGO detectors.

- to avoid light diffusion, every part of the interferometer is in high vacuum with $P \sim 10^{-7} - 10^{-9} Pa$. Nevertheless, the residual gas pressure P cause fluctuations due to scattering of beam photons which power spectrum can be approximated as

$$S_{h_{vac}} \gtrsim 7 \cdot 10^{-25} \left(\frac{P}{10^{-7} Pa} \right)^{-\frac{1}{2}} Hz^{\frac{1}{2}}$$

2.1.4 Antenna pattern

As we said in Section 1.4, a TT-gauge gravitational wave is seen in different ways depending on the wave polarization and direction between the observer and the source rotational axis. Since an interferometer doesn't have a spherical geometry, similarly the geodesic deviation induced by a gravitational signal to the arms of the detector will cause different response, depending both on the polarization of the wave and the propagating direction of the incoming wave, with respect the plane where the arms lay. This means that the detector has different sensitivity to signals coming from different directions: an interferometer in a certain position on Earth "watches" different sky positions more or less well at a time t, as can be seen in figure 2.6.

2.2 Searching signals in the data

Up to this point we depicted two sides of the big challenge for the detection of gravitational waves: which form we expect from the signal for various source types and how the detectors work. Now we need to join these two different but complementary aspects, trying to explain how we can extract a possible signal from very noisy data.

First of all let's consider how we manage the data. The raw output of the photodiode of an interferometer is a sampled measure of light power, this is called

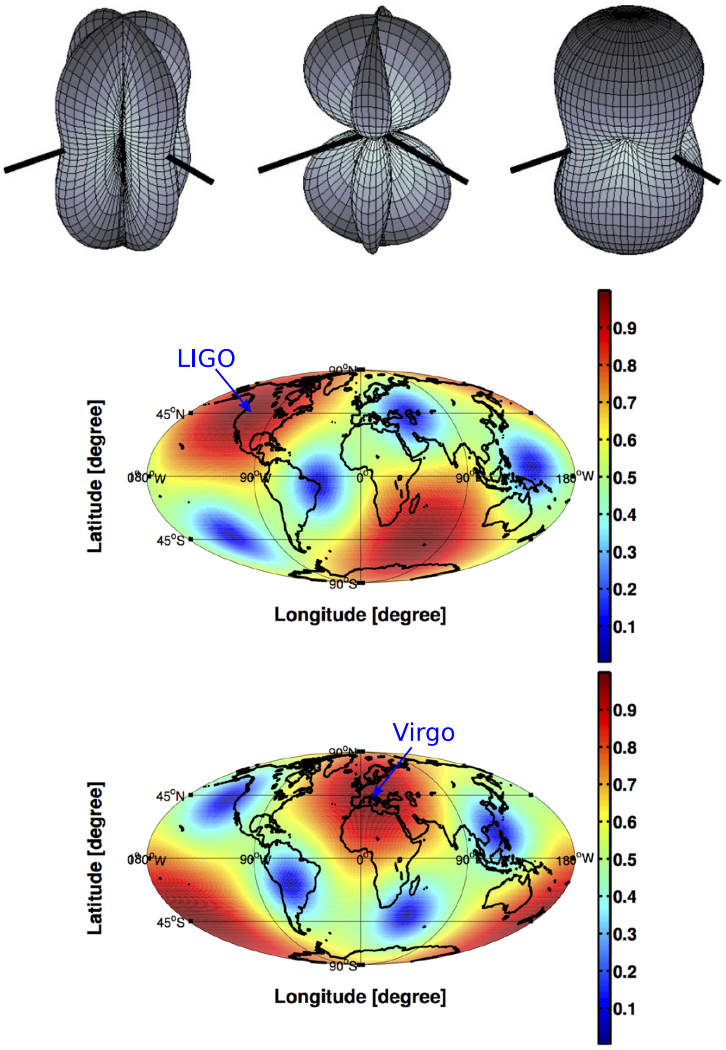

Figure 2.6: Top: antenna pattern for a Michelson interferometer, in the small antenna approximation, for $+$, $\times$ and averaged polarization. The beam splitter is at the origin of the pattern and the arms orientation are indicated by the solid black lines. Bottom: antenna patterns represented as colorbar plots on a Mollweide projection of the Earth, for LIGO and Virgo interferometers. The plot shows at a fixed time how at different position on Earth a detector can respond to a gravitational wave signal, in function of the incoming direction in the sky.

time series. A time series from our detectors is then basically a long stream of digital random voltage measurements, with discrete times ($t_{i+1} - t_i$ = sampling time), where a subset of the series with duration T will be different to the next one. Conversely, the expected signal are deterministic well defined functions of time (stochastic background excepted).

A useful way to study fluctuations and oscillations is to work in frequency domain via the Fourier transform. Being $s(t)$ the series or the signal function, we'll express its Fourier transform as $S(\nu)$. Since the time dependence is discrete, the Fourier transform is computed via *Fast Fourier Transform* (FFT) algorithms.

2.2.1 Data characterization

With a set of data in a certain time or frequency interval, key tools for data analysis are operations that relate pairs of function. Considering for example two functions of time $s_1(t)$ and $s_2(t)$. We can define then[3]

Cross-correlation $(s_1 \star s_2)(t) \coloneqq \int_{-\infty}^{+\infty} s_1(\tau)s_2(t+\tau)d\tau$. This is a way to express the degree or relatedness of s_1 and s_2 with a certain offset t. An immediate example of use is the cross-correlation between the time series from two interferometers, with a offset $t = D/c$, where D is the 3D Euclid distance between the two detectors.

Auto-correlation $(s\star s)(t) \coloneqq \int_{-\infty}^{+\infty} s(\tau)s(t+\tau)d\tau$, setting $s_1 = s_2 = s$ in the above definition. Chosing various time lags t_i, this is a measure of the relatedness of a function with itself at different offsets. Obviously for $t = 0$ (and $t = kT$ for periodic functions with period T), the autocorrelation has a maximum. It could be a good tool when we see a "bump" in a time series and we want to know if and how often this bump occurs again. The zero-lag autocorrelation is quite useful because $(s \star s)(0) \equiv \langle s^2 \rangle$, that is the squared mean over the sample. If the function has zero time average, then $\langle s^2 \rangle = \sigma^2$ and $(s \star s)(0)$ is the variance of the function, giving us the magnitude of the noise. So if $s(t)$ is a random time series, the power spectrum of s is the Fourier transform of its autocorrelation function

$$P_s(\nu > 0) \coloneqq \frac{2}{\sqrt{2\pi}} \int_0^{+\infty} (s \star s)(t) e^{-i2\pi ft} dt,$$

where there is a factor 2 because we don't define power spectral density at negative frequencies. The amplitude spectral density is in turn defined as $s(\nu) \coloneqq \sqrt{P_s}$.

Convolution $(s_1 * s_2)(t) \coloneqq \int_{-\infty}^{+\infty} s_1(\tau)s_2(\tau - t)d\tau$. The different sign of the time offset in the definition changes the meaning of the integration, since while the correlation is a measure of similarity of a function with respect to another function, the convolution is an operator that modifies a function using a defined template. It is well suited, for example, for filters: a function that firstly matches the the end of the filter with the time series start, then slides on it changing its shape. Defining the impulse response $f(t)$ of the filter as the output due to an impulsive input applied at $t = 0$, the output $v(t)$ for a generic input $s(t)$ will be the convolution $v(t) = \int_{-\infty}^{t} s(\tau)f(\tau - t)d\tau$. By the convolution theorem, the transfer function $F(\nu)$ of the filter, defined as the Fourier transform of f, is simply the ratio of the Fourier transforms of output and input : $F(\nu) = V(\nu)/S(\nu)$.

[3]To avoid annoying repetitiveness we assume only continuous functions, so the definitions are in integral form. Nevertheless, when we manage data, they are usually discrete time series. The conversion to continuous to discrete definitions is straightforward.

2.2.2 Data analysis with a single detector

As we saw, the cross-correlation gives an estimate of the similarity of two functions. If we consider a signal template $h(t)$ and the output time series from our detector $o(t)$, the cross-correlation between them will give us informations about the presence of the signal h in the data. Let us suppose, for the sake of simplicity, that the noise follows a Gaussian distribution uniform in frequencies. Then at each t the cross-correlation $(o \star h)(t)$, with no signal in the data, has a Gaussian probability distribution too, with a certain average that we can set 0 and a variance σ^2 that, in the chosen hypothesis of white noise, will give our noise amplitude.

If at the time t there is a signal sufficiently strong, then we expect $(o \star h)(t)$ returns an unlikely high value with respect to the variance of the noise σ_n^2. Defining a threshold on the cross-correlation, in terms of signal to noise ratio SNR, higher values at certain times will label signal candidates with a certain confidence: in hypothesis of Gaussian noise, a certain SNR value n, will correspond to a confidence of $n\sigma_n$. So we can interpret the signal to noise ratio as an estimation of the probability of the no-signal hypothesis. We can use the idea of a match between data and a template to filter out the noise and improve SNR. This is the basic concept of the so-called *matched filter*.

Optimum filter

Let us consider the convolution between $o(t)$ and the chosen signal template $h(t)$. It can be shown that in the white noise hypothesis this is the filter that maximizes the signal to noise ratio. When the noise isn't white the filtering is a bit more complicated, because we need firstly to *whiten* the noise.

It is convenient to work in frequency domain. Being $f(t) = h(-t)$ our impulse response in white noise hypothesis, and $o(t)$ the non-white noisy output of the detector, therefore the Fourier transforms will be $F(\nu) = e^{-i2\pi\nu t_0} H^*(\nu)$ and $O(\nu)$. To whiten the noise is sufficient to apply another filter before the matched one with transfer function $W(\nu) = 1/N(\nu)^2$, where N is the noise power spectrum. Once the data are whitened, by linearity we can apply F. The optimum filter for non-white noise will be then

$$M(\nu) \propto (F \circ W)(\nu) = \frac{e^{-i2\pi\nu t_0} H^*(\nu)}{N(\nu)^2}.^{4} \tag{2.10}$$

As we can easily imagine, a precise definition of the template of the searched signal is crucial. For this reason, it is mandatory to model the correct functional form and, since we can't know a priori the function parameter values, to define a bench of templates of the same function, varying the parameters within a likely parameter space.

[4] Keep in mind that the proportionality holds because the filters are defined up to a constant in the templates.

2.2.3 Coincidences between multiple detectors

In the search for gravitational wave signals, in many cases it is fundamental the presence of at least two independent detectors. In fact, in principle one could use only one detector and look for "excess" power in the spectrum, as said before, but we do not know well enough the noise to exclude that a certain candidate signal is an artifact due to a disturbance in the noise.

A fundamental way to exclude that case is to use a detector network: different detectors can exclude internal sources of noise. Moreover if they are in different places we can exclude external sources[5]. In this way, fixed a false alarm probability on every detector, a network candidate signal will be far more unlikely a fake.

There are basically two ways to manage the data coming from two gravitational detectors. The first is to see coindicences: if we have a candidate signal on a detector, for example an online matched filter algorithm returns a signal to noise ratio above a certain threshold for a binary coalescence, then the same template is searched in the other detectors within a coincidence window depending on the respective distance. Another way to exclude from data any local features is to cross-correlate two detectors, using the time series or their Fourier transform and taking into account the relative distance and orientation. This approach is crucial, for example, for the search of the stochastic gravitational wave background, since it is indistinguishable from the noise of a detector (Romano and Cornish, 2017, B. P. Abbott et al., 2017e).

2.3 Continuous waves search

The search for continuous gravitational waves, both targeted from known pulsars or *blindly* over the whole sky, is far more challenging with respect to the case of transient signals from coalescing binaries. The first difference is the strain amplitude of the signal: directed searches on known neutron stars gave upper limits on the ellipticity such that a possible gravitational signal is below $\approx 10^{-24}$-10^{-25} (Aasi et al., 2015b), 4-5 orders of magnitude lower than detectable binary coalescences.

If we don't know the sky position of the source, as happens with blind searces, detecting continuous waves is more difficult also because the dimension of the parameter-space. In fact, despite we know ~ 3000 neutron stars from their electromagnetic emission, though there may be around a billion of undetected neutron stars in our galaxy. Since we don't know anything of such objects, apart for the many unknown intrinsic parameters of the systems, they can stay at every direction on the sky. The Doppler modulation due the Earth rotation and orbital motion with respect to the source will spread the monochromatic signal from a neutron star in many small peaks in the power spectra. Unable to properly correct the Doppler effect, in blind searches we use appropriate precautions, preventing the Doppler spread at the cost of a limited time integration in the Fourier transform of the time series. Doing this, we can use the characteristic path in the frequency/time plane of the Doppler modulation to recognize candidate sources.

[5]Up to disturbance of Earth size scale, like fluctuations of the Earth magnetic field (Izabela et al., 2017).

If we had an isolated detector freely traveling in space, in principle one could simply think to gather a time series with length T and search for a sinusoidal signal from a rotating system in an extremely high (the signal to noise ratio will be $\propto \sqrt{T}$) and narrow peak of the power spectrum. With this idea one could want to integrate over the longest possible time series.

However, our detectors are on Earth and the Doppler modulation is unavoidable, causing the fact that a sinusoidal peak is spanned with time on various frequencies, with a significant loss of signal significance. In the spectrum there will be indeed, instead of the single peak of a monochromatic signal, the characteristic carrier plus the sidebands corresponding to the modulation. For a modulation with frequency ν_m, the number of sidebands at a frequency ν is $\delta(\nu) \approx \nu/\nu_m$ and each sidebands peaks have an amplitude $\approx 1/\sqrt{\delta}$ (Saulson, 2017).

So we have a situation where the spectrum peak should grow as the square root of the integration time, but the frequency resolution of the Fourier transform goes as $1/T$, letting soon the sidebands to be resolved. With a sufficiently short time series, the Fourier transform will not suffer the modulation because the spread will be less wide than the resolution in frequencies, and every sideband will fall in the same frequency bin.

For the daily modulation ($\nu_m = 1.2 \cdot 10^{-5} Hz$) there are ≈ 10 sidebands, causing a factor 3 amplitude loss; with the annual modulation ($\nu_m = 3.2 \cdot 10^{-8} Hz$) there are $\approx 10^6$ sidebands and the spectral peak will be reduced by a factor of 10^3.

To solve the problem, we must Doppler correct the detector data: if we know the sky position of a candidate source we can easily recover the original peak removing the sidebands. The precision on the sky position needs to be quite fine anyway, since with a better resolution in frequency domain we need as much a precise correction of the Doppler effect. In all-sky searches this become unmanageable, since we have to correct the data for any sky position.

These considerations are in the hypothesis of a stationary detector noise and a perfectly coherent signal from the source. In reality we have disturbances and glitches both in the instrumental noise and in the signal from the source, adding more difficulties to the described picture.

For these reasons research groups in the Virgo and LIGO collaborations have developed algorithm and hierarchical procedures to try to extract the signal from the data of the gravitational interferometers (B. P. Abbott et al., 2017a), trying to reduce the computational cost with small sensitivity loss. In the following we'll summarize the *Frequency-Hough* pipeline analysis (Astone et al., 2014), which core is an implementation of the Hough transform (Hough, 1962). In the next chapter we'll describe in details how the Hough transform works and why has a central role in the analysis method.

2.3.1 Analysis pipeline

In this section we will focus on the data analysis of a wide-band all-sky search for gravitational waves emitted by asymmetrical rotating neutron stars, using the data from the LIGO and Virgo detectors. It is structured in hierarchical procedures, developed to allow large computational cost reduction trying to minimize the loss in sensitivity with respect to a directed or targeted search.

The procedure uses data time series from the detectors to compute a database of short fast-Fourier transforms (FFT) at various frequency bands with length, depending on the chosen band, from ~ 15 to ~ 120 minutes. Every FFT is whitened and from each one the most significant peaks are selected, building a frequency/time map of the selected peaks and correcting the frequencies removing the daily and annual Doppler effect. The heaviest part is then the computation of the Hough transform, looking for a straight pattern of peaks that could suggest a sinusoidal signal with a certain frequency and spin-down. For every possible pattern found by the Hough, signal candidates are selected and matched with the candidates selected by the same procedure in the data from the other detectors. The *coincident* candidates are then subject to various verification steps and to a follow-up analysis in order to confirm or reject them. Let's give a few more details on every step.

Short Fourier transforms

The Fourier transforms are obtained with FFT algorithms, using detector calibrated data divided into chunks interlaced each other by half. The time series are cleaned, removing identified disturbances before the FFT computation and putting to zero every data collected during *non-science* times.

It has been chosen to use different data chunk durations T_{FFT} to compute the Fourier transforms in different frequency bands, following a criterion according which a possible signal is spread by the Doppler effect lesser than a frequency bin. Since the resolution of the Fourier transform goes as the inverse of the integration time, the frequency bin is $1/T_{FFT}$. Choosing T_{FFT} of the order of minutes or hours we are sensible to the daily modulation, then the maximum FFT duration and the maximum frequency of the Fourier transform are related by the equation $T_{FFT_{max}} \sim \nu_m/\sqrt{\nu_{max}}\ s$, where $\nu_m = 1.2 \cdot 10^5 Hz$ is the daily modulation frequency and ν_{max} is the chosen maximum frequency of the FFT.

In order to satisfy in a practical way this criterion is enough to choose four different T_{FFT} corresponding to four frequency bands:

$\nu(Hz)$	$T_{FFT}(s)$
10-128	8192
128-512	4096
512-1024	2048
1024-2048	1024

Given N_t chunks of the selected time series, the unit time is the T_{FFT} of the frequency band chosen, and the single FFT is identified by a time $kT_{FFT}/2$, with $k = 0, ..., N_t$ (the factor $1/2$ is due to the interlace of the FFTs). In a FFT, the frequencies are identified by a frequency j/T_{FFT}, where $1/T_{FFT}$ is the width of the jth bin.

Peakmap

Once we have the full FFT database in the chosen time and frequency intervals, a good way to have informations on the ith FFT $S_i(j)$ over the frequencies $\nu_j =$

j/T_{FFT}, is to compute the ratio between the periodogram $S_{P_i}(j) = |S_i(j)|$ and the auto-regressive average spectrum S_{AR_i}:

$$R_{ij} = \frac{S_{P_i}(j)}{S_{AR_i}(j)}.$$

In the R_{ij} matrix we select elements, in the time-frequency plane, for which both the conditions are valid:

- R_{ij} is higher than a threshold θ,
- it is a local maximum,

and they are called peaks. The full set of peaks forms a *peakmap*, which is then a binary image on the time-frequency plane.

The double conditions choice lowers significantly the number of peaks selected, reducing the computational cost, and most importantly makes the pipeline less vulnerable to spectral disturbances, increasing the robustness of the research at a price of a relatively low sensitivity loss (Astone et al., 2014).

It's important to estimate how many signal peaks can be missed and how many noise disturbances are instead selected in the peakmap, because both affect the next step of the analysis. Assuming gaussian noise, given the size of R_{ij} as $N = N_t N_\nu$, the probability to select n peaks by random fluctuations of the noise follows a binomial distribution, with expectation value Np_0 and variance $Np_0(1-p_0)$. The probability p_0 to have a local maximum above the threshold θ due to the noise is

$$p_0 = e^{-\theta} - e^{-2\theta} + \frac{1}{3}e^{-3\theta}. \tag{2.11}$$

If we have a signal with spectral amplitude λ, the probability to select a signal peak is instead

$$p_\lambda \approx p_0 + \frac{\lambda}{2}\theta(e^{-\theta} - e^{-2\theta} + e^{-3\theta}). \tag{2.12}$$

Then we have that both false dismissal probability $(1 - p_\lambda)$ and false alarm probability p_0 depend on the threshold θ.

Doppler correction and Hough map

The Hough transform will be treated in details in the next chapter (Section 3.2) but, shortly, it is a method to recognize patterns in images (Hough, 1962). In its simplest implementation, it identifies straight lines in the image, returning their parameters as coordinates in the parameters space.

Besides we didn't correct the Doppler effect from the data so far, it is still necessary to remove it, because with the Hough transform it is simpler to search for a straight line than a double modulated signal by daily and annual frequencies. In order to do this, remember that we chose the FFT length such that the signal power by the Doppler effect remains in the same frequency bin. For this reason it is just necessary to shift the frequency values in the peakmap for each FFT, for every sky position we want to analyze, according to the velocity vector of the detector during the FFT sum time. In this way a double sinusoidal pattern in the peakmap,

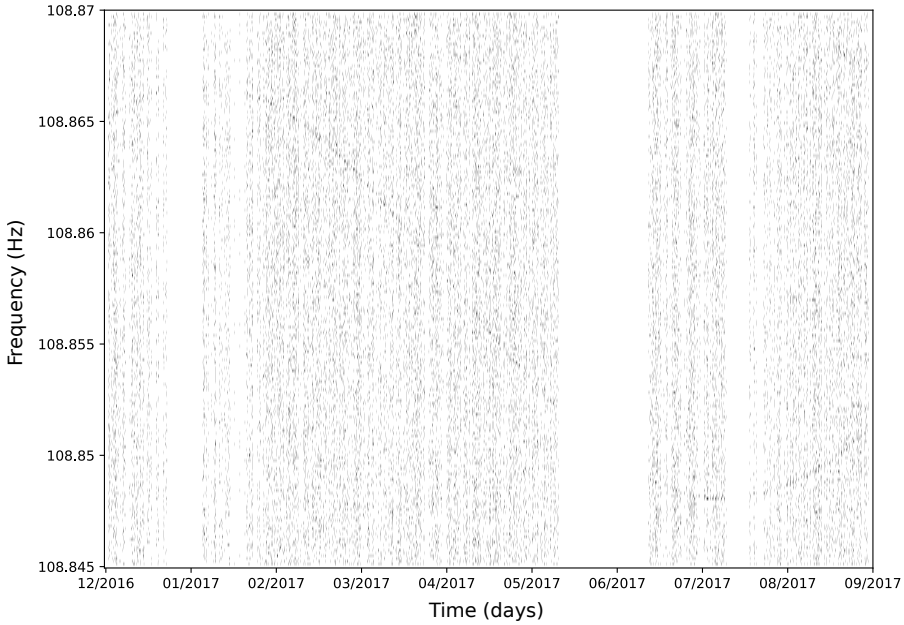

Figure 2.7: An example of a peakmap before the Doppler correction. It can bee seen the hardware injection of a pulsar modulated by the annual Doppler effect.

due to the superposition of orbital and rotational Doppler-shifts, becomes a straight line. A good correction is crucial, since with the Hough transform we look exactly for a straight line, and a deviation from the equation 2.13 causes a heavy loss in sensitivity.

After the Doppler correction we look in the peakmaps for characteristic patterns of emitting neutron stars. We expect a frequency that varies over time according to the spin-down parameter $\dot{\nu}_0$, assuming the second order spin-down contribution can be neglected:

$$\nu(t) = \nu_0 + \dot{\nu}_0(t - t_0) \tag{2.13}$$

where ν_0 is the source reference frequency at time t_0. So the patterns we look for in the peakmap are straight lines over the whole observation time, with a reference frequency ν_0 and the spin-down $\dot{\nu}$ as slope.

The Frequency-Hough transform is an implementation of the Hough transform from the observed frequency/time plane in the source reference frequency/spin-down plane: every point $(t - t_0, \nu)$ in the peakmap is transformed in a straight line with slope $-1/(t - t_0)$. In this way we obtain that every collinear peak of a signal will form straight lines with different slopes in the Hough map, generating a family of lines intersecting in the same point of the parameters space. The coordinates of the incidence point give the parameters of the signal straight line in the peakmap: ν_0 and $\dot{\nu}_0$.

This is the core of the analysis, since it is the part where actually the data are inspected to search signals, and represent also the heaviest computational stage of the pipeline.

Candidate selection and coincidences

When we have computed, for the observation time and the frequency band chosen, the full set of Hough maps for every sky position of interest, a certain number of candidates N are selected from each Hough map. There are several possible selection criteria, but the one used in the Frequency-Hough pipeline aims for robustness using a maximum selection, in order to avoid to be overwhelmed by disturbances in a similar way to the peak selection criteria. The selection indeed is done splitting it in N frequency belts and searching in each belt for the pixel with the highest amplitude n. For every candidate selected, its parameter-space coordinates and *critical ratio* are stored. The latter is defined as

$$CR = \frac{n - \langle n \rangle}{\sigma_n} \tag{2.14}$$

with $\langle n \rangle$ the average amplitude in the belt and σ_n its standard deviation. The second highest candidate in the belt is chosen if it is above the stripe average amplitude and far enough from the primary one (Astone et al., 2014).

Actually, the presence of disturbances in the data, both in time and frequency domain cause the Gaussian noise hypothesis fall. In this situation the average amplitude suffers the presence of a long tail in the distribution, and a median results to be more robust (Astone et al., 2014). Using the median it is possible to define an estimator of the dispersion parameter, to use instead the standard deviation. Calling $m(x)$ the median of a random variable x, the dispersion parameter is defined as

$$s(x) = \frac{m(|x| - m(x))}{C},$$

where $C = 0.6745$ is a normalization factor such that $s(x) = \sigma_x$ is x follows a Gaussian distribution. The CR definition become then straightforwardly

$$CR = \frac{n - m(n)}{s(n)}.$$

Once we have selected N candidates for every sky position, using the data from one detector, they are stored all together. We expect that most of the selected candidates faked a signal, then we have to do a deeper analysis in order to reduce the false alarm rate.

The first and most important step is the search for coincidences comparing the candidates gathered from two detectors. Since many candidates can be sons of the same signal or disturbance, before the coincidence step it is useful to cluster similar candidates.

Given two datasets, each one with $N_{1,2}$ candidates, the discrete parameters space coordinates of two candidates chosen from the respective dataset are $\mathbf{c}_{1,2} = (\lambda_{1,2}, \beta_{1,2}, \nu_{1,2}, \dot{\nu}_{1,2})$, with λ, β sky position in ecliptic coordinates. The distance d is defined as the Euclid norm between them, and the coincidence of a couple of candidates from the two datasets is established if they have a distance in the parameter space below a certain value, e.g a few bins for every dimension. When this condition is satisfied, the coincidence is set and the candidates are stored for further analysis.

Both choices on the number of candidates for each dataset and the number coincidences to select N_{coin}, depend on a compromise between computational cost and search sensitivity.

Verification and follow-up

To further reduce the number of false candidates is useful to do some verification on the coincident candidates, in order to have a further reduction on the computational cost of the last step of the analysis: the *follow-up*.

Firstly we remove any candidate associated to known calibration or resonance line, then we check whether the candidate couple has compatible amplitudes, since we expect that the signal amplitude estimation from the Hough map should be the same in different dataset.

The last step of the selection is to impose a critical ratio threshold on the coincident candidates. This is a delicate step, because both the computational power and the search sensitivity depends on the CR_{thr} chosen.

Surviving candidates are subject to a follow-up analysis. It consists in a deeper analysis of a small portion of the parameter space of each candidate, using new longer FFTs with a longer integration time and a new Hough transform stage.

This last step do not change the research sensitivity, which depends on the thresholds and selections explained above. In other words, a signal discarded before can't be recovered. However we still need to reduce the false alarm probability to increase the detection confidence.

2.3.2 Sensitivity estimation

The amplitude of a pixel in the Hough map, in the hypothesis of only noise data follows a binomial probability distribution, with average number count $\langle n \rangle = Np_0$ and standard deviation $\sqrt{Np_0(1-p_0)}$, with N the number of FFT of the peakmap and $p_0(\theta)$ the probability in equation 2.11. The critical ratio of a pixel with count n will be then

$$CR = \frac{n - Np_0}{\sqrt{Np_0(1-p_0)}}. \tag{2.15}$$

Both θ and the threshold on CR influences the sensitivity of the research. In the following we'll assume the noise fluctuation follow a random Gaussian probability distribution. The sensitivity obtained will be a good optimistic approximation of the true research sensitivity.

Choice of θ

A criterion for the choice of the threshold θ in the peakmap construction should reduce considerably the computational power with a low sensitivity loss.

We can define the spectral amplitude of a signal with the square module of its Fourier transform $H(\nu)$, in units of equalized noise expressed in terms of the detector noise spectral density $S_n(f)$:

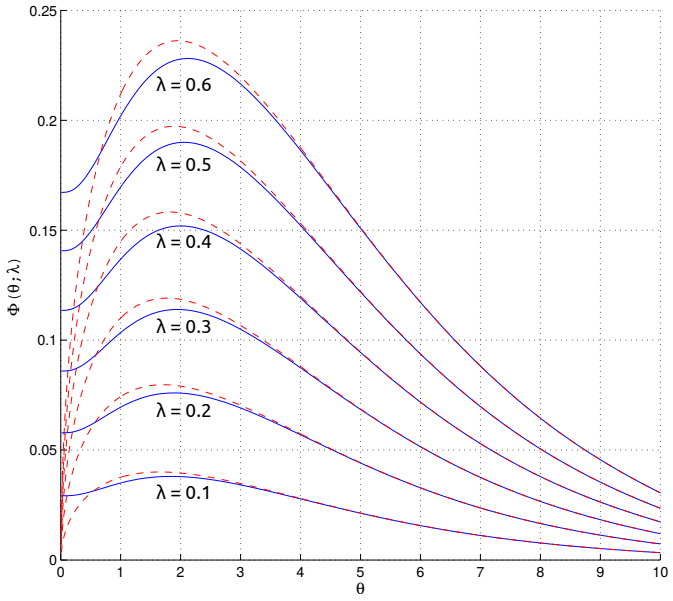

Figure 2.8: Plot of the function Φ for various signal amplitude values. Blue continue curves correspond to local maxima above threshold peaks, red dashed to only above threshold peaks condition.

$$\lambda = \frac{4|H(\nu)|^2}{T_{FFT} S_n(f)}.$$

A good criterion to the choice of θ can be then the maximization of the expectation value of CR when a certain signal with spectral amplitude λ is present:

$$\mu_{CR}(\theta, \lambda) = \frac{N(p_\lambda - p_0)}{\sqrt{N p_0 (1 - p_0)}} = \sqrt{N}\Phi(\theta, \lambda)$$

As can bee seen plotting Φ (Figure 2.8), with a low signal amplitude the value of the function varies weakly with $\theta \in [1, 3]$. For this reason, since we expect faint signals, a good choice can be $\theta = 2.5$ for any λ. Fixing this threshold the sensitivity loss is negligible ($\sim 1\%$) with respect to a threshold set at the maximum of the function Φ , with the advantage of a heavy peaks number reduction in the peakmap, thus reducing a lot the computational cost of the Hough transform.

Moreover, the gain in sensitivity we could have selecting every peak above threshold in a FFT, instead the local maxima, carries a sensitivity improvement of $\sim 5\%$ with the drawback of $\sim 9\%$ computational load increase (Astone et al., 2014). Most importantly, the selection of only the maxima gives more robustness to the analysis. In fact, often a disturbance in the data affects a group of neighbor frequencies, generating a high number of peaks in the FFT spectrum. If we select only the maximum one, the risk to be blinded by a disturbance is far lower.

Choice of CR_{thr}

Let us suppose to have a set of candidates selected from the Hough and that, in the frequency band chosen for the analysis, T_{FFT} is constant and that the number

of selected candidates isn't frequency dependent.

Typically the amplitude of a pixel in the Hough map in a long time analysis is quite high ($\gtrsim 10^2$), then we can use the Gaussian approximation to the binomial distribution to give the probability of n counts:

$$P_{\theta,\lambda}(n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(n-\mu)^2}{2\sigma^2}},$$

with $\mu = Np$ and $\sigma^2 = Np(1-p)$ where p is p_0 in the only noise hypothesis or p_λ in presence of a signal.

If we define $CR_{thr} = CR(n_{thr})$ with fixed N and p_0, using the above approximation the probability to select candidate in the no-signal hypothesis (i.e. the candidate is due only to a fluctuation of the noise) can be computed easily with the complementary error function

$$\begin{aligned} P_{fa}(n_{thr}) &= \int_{n_{thr}}^{\infty} P_{\theta,0}(n)dn = \\ &= \frac{1}{2}\,\mathrm{erfc}\left(\frac{n_{thr} - Np_0}{\sqrt{2Np_0(1-p_0)}}\right) \\ &= \frac{1}{2}\,\mathrm{erfc}\left(\frac{CR_{thr}}{\sqrt{2}}\right) \end{aligned} \tag{2.16}$$

Since we selected candidates with the highest CR, in the no-signal hypothesis we have that $P_{fa} = N_{cand}/N_{tot}$, where N_{cand} is the number of fake candidates and N_{tot} is the total number of points in the parameter space of our analysis. Then the threshold on CR can be written as

$$CR_{thr} = \sqrt{2}\,\mathrm{erfc}^{-1}(2P_{fa}) = \sqrt{2}\,\mathrm{erfc}^{-1}\left(2\frac{N_{cand}}{N_{tot}}\right) \tag{2.17}$$

We can choose a CR_{thr} such that the false alarm probability is below a certain value, remembering however that in this way we can exclude a true signal candidate. Indeed, the threshold on CR is an important parameter of the sensitivity of the research, as much as of its computational cost.

2.3.3 Search sensitivity

The sensitivity with a confidence level Γ is defined as the minimum signal amplitude which would produce a candidate in a fraction $\geq \Gamma$ of a statistical *ensemble* of the experiment. It does not depend on the result of the analysis, i.e. the actual candidates chosen and verified.

The probability to select a candidate with a count above a chosen threshold, for a given signal spectral amplitude λ, is

$$P_{n>n_{thr}}(\lambda) = \int_{n_{thr}}^{\infty} P_{\theta,\lambda}(n)dn.$$

We can obtain the sensitivity value imposing $P_{n>n_{thr}}(\lambda) = \Gamma$, and we'll get

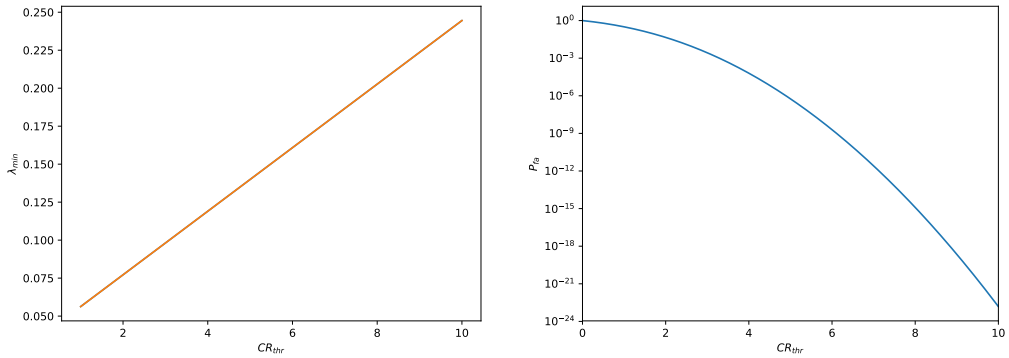

Figure 2.9: Left: plot of λ_{min} vs CR_{thr} with $\Gamma = 95\%$, $\theta = 2.5$ and $N_{FFT} \approx 3400$. Right: dependence of P_{fa} with CR_{thr}.

$$\text{erfc}\left(\frac{n_{thr} - Np_\lambda}{\sqrt{2Np_\lambda(1-p_\lambda)}}\right) = 2\Gamma.$$

Solving the equation in the small signal approximation, we have the minimum detectable spectral amplitude, function of θ, CR_{thr} and Γ

$$\lambda_{min} \approx \frac{2}{\theta}\sqrt{\frac{p_0(1-p_0)}{Np_1^2}}\left(CR_{thr} - \sqrt{2}\,\text{erfc}^{-1}(2\Gamma)\right), \tag{2.18}$$

with $p_1 = e^{-\theta} - 2e^{-2\theta} + e^{-3\theta}$. The corresponding minimum detectable strain amplitude, given the detector spectral power $S_n(f)$, is

$$h_{0_{min}} \approx \frac{4.02}{N^{1/4}\theta^{1/2}}\sqrt{\frac{S_n(f)}{T_{FFT}}}\left(\frac{p_0(1-p_0)}{p_1^2}\right)^{\frac{1}{4}}\sqrt{CR_{thr} - \sqrt{2}\,\text{erfc}^{-1}(2\Gamma)} \tag{2.19}$$

It's important to point out that λ_{min} is linearly dependent on the critical ratio threshold, while the sensitivity loss dependence on θ showed in figure 2.8 is always weaker in the small signal hypothesis. However, the false alarm probability relation with CR_{thr} is much stronger since is driven by the complementary error function: if we choose a lower CR_{thr} the computational load of the follow-up will raise much more than the sensitivity improvement (see Figure 2.9).

Chapter 3

Continuous waves with the GPU Hough transform

In this chapter we'll focus on the programming and computational part of the search for continuous waves.

One of the main objectives of this Thesis work is to study and demonstrate the feasibility of a complete continuous waves analysis with *GPGPU* (General-Purpose computing on Graphics Processing Units), using a high level programming code. From the many existing libraries and frameworks, I chose the younger one developed by Google and released with an open source license: TensorFlow (2017).

TensorFlow works with CUDA Toolkit libraries from NVIDIA and it is based on Python, with a symbolic programming paradigm and a syntax similar to high level scientific programs/languages such MATLAB, or numeric Python libraries like Numpy. Despite it has been originally developed for machine learning and neural networks, TensorFlow fits well for a wide variety of purposes and, specifically, for scientific data analysis. The main reasons of this choice are:

- high GPU efficiency;
- high level programming, which implies fast development of new codes and relatively low steep learning curve;
- linked to the above point, but very important to deserve a dedicated one, high level programming implies high portability: a developed code can run on every CUDA GPU, without the need to force the user to heavy customization, memory management, cores management.

Once we have summarized the main characteristics of GPGPU, in the rest of this chapter it will be explained how the Hough transform works and why is important a GPU version with efficient parallelization. At last, the new algorithm will be applied in a semi-directed analysis of the sky region around the galactic center, with a setting similar to the all-sky searches (Aasi et al., 2016, B. P. Abbott et al., 2017a), in order to extract a set of continuous signal candidates to be further analyzed successively. As I'll show, this first version of the GPU algorithm goes 20 times faster, on a single Tesla K20, than the original MATLAB code on a Xeon CPU with twelve 2.40GHz cores. An all-sky search works on a huge parameter space,

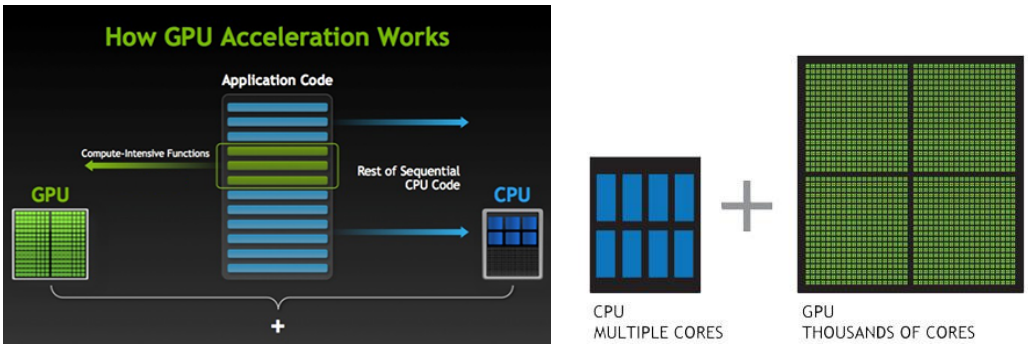

Figure 3.1: Scheme of joined CPU-GPU computation: the sequential part of a program is more efficiently managed by the CPU, while the GPU manages the massive parallel computations which on CPU would cause a much higher total computation time (image from nvidia.com).

requiring millions of core hours, a GPU parallelism could bring to a significantly higher computation efficiency, allowing to run much deeper analysis.

3.1 GPGPU

In the last years the need for higher and higher computational power collided with the increasing physical difficulty and cost to obtain faster CPUs with higher clock rate. With the end of the so-called Moore's law, it has become crucial to apply more and more parallelization on the computing systems. While the first obvious practice is to parallelize CPUs, the cost of a CPU cluster is still quite high and performances difficult to manage.

Since the '70 of the last century, the videogame development needed a high performance (for the time) dedicated computing hardware, to compute at a high rate images describing 2D scenes. In the '90s, with the birth of 3D graphics, it became mandatory to use dedicated devices to render 3D scenes, and the 2D rendering was brought on CPU, since new technologies allowed to manage the latter directly in the central processor of the host system.

These devices are the Graphing Processing Units, structured as integrated circuit on a board with various components: mainly interfaces, memory and graphic processors. From the beginning the GPUs are designed to split the task of draw a scene on different core types: one manages the *shaders*, i.e. small programs which describe the physical aspect of an object with respect to light, shadows, colors, able to simulate a 3D object; the second type manages texture mapping, that means that deforms a 2D image and applies it on the 3D model; the last type collects the informations coming from the other processors to render the image we see on the screen. Despite the first GPUs had substantially the same number of cores for each type, the shaders processors are the most important because carry the heaviest part of the work: they basically build the 3D scene.

The typical problem a GPU solves is to compute which different pixels of a monitor should show something, taking into account the user interaction, the neighbor pixels and the physical dimensions of the scene. Everything has to be done

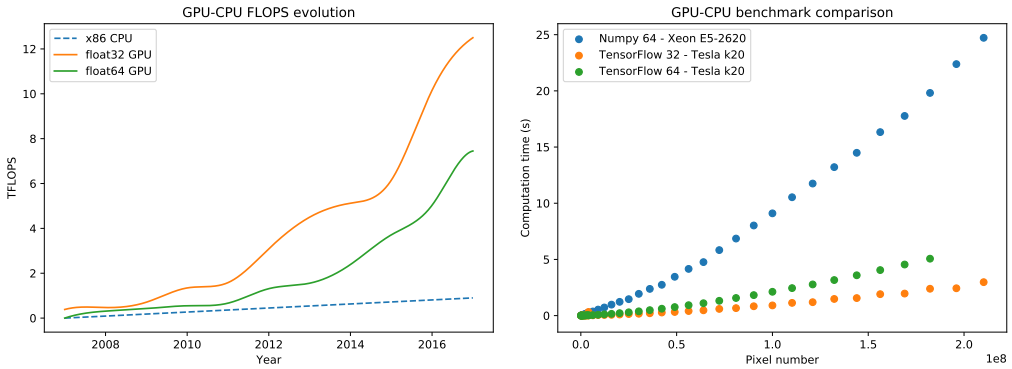

Figure 3.2: Left: trend of GPU and CPU FLOPS capability in the last ten years. Right: a simple benchmark on the device used for the Frequency-Hough analysis, based on the generation of two random arrays and a matrix multiplication on them. Looking the green dots, it can be seen how the performances both in computational time and memory usage are worst using 64 bit data: at $\sim 2 \cdot 10^8$ pixels the GPU went out of memory. The devices used for the test are a NVIDIA Tesla K20m, with ~ 2500 cores at 706 MHz and 5 GB memory, in an host system with 60 GB RAM and an Intel Xeon CPU E5-2620 v3, with 12 2.40 GHz cores, 2 threads each one.

at a high rate (around the refresh rate of the screen, 60 Hz usually). To manage such computations, it is useful to have a high number of processors, which take care of different tasks and image pieces, communicating each other in a very efficient way with shared memories. Indeed this is the basic description of the structure of a GPU: a device thought and developed in time to do, with the best possible efficiency, computations with an extremely high parallelism.

The challenge between the ever-increasing demand of realistic graphics and the very fast technology evolution, brought to have at present days GPUs with hundreds and even thousands of cores and various GB of dedicated RAM. In Figure 3.3 are shown three images which can give an idea of the difference in computational power needed to reproduce an image from 2D to 3D videogames, and from the first 3D to present days. Nowadays, a 3D scene is a real detailed physical simulation with a high level of complexity and likelihood.

Despite the single core of a typical GPU has a lower clock rate than a common CPU core, it is far more efficient to work in parallel with the others, while CPUs are still the best option to do sequential calculations. A good approach is then to demand serialized operations to the CPU and reserve the GPU only or well parallelizable problems, where it gives far more computational power than a single multi-core CPU of similar cost (see Figure 3.1 and right panel in Figure 3.2).

Thanks to the GPUs technology, the processing power for floating point calculations exploded in the last ten years (Figure 3.2, left), and the use of GPUs in many different fields, from scientific research to economics and so on, gave birth to the general purpose GPU computing. The most fascinating thing of this evolution is that, at a low cost, most people have a multi-core system for highly efficient parallel computing in their own personal computers.

In this contex, with the creation of CUDA (2007) and OpenCL (2009) libraries,

Figure 3.3: An example on how the video rendering evolved from the '80s to present days in videogames industry. It's easy to imagine how much more computational power it has been necessary to develop, in order to pass from 2D to 3D rendering, and at present days, to add physical simulations at displayed scenes and interactions of the user.

it has been established the development and production of GPUs dedicated to scientific massive calculations.

The two main competitors AMD and NVIDIA can be considered, with exchanging fortunes in time, at the same level from the point of view of reliability and computational power of their respective products. However, the latter has expanded in the GPGPU market thanks to well documented APIs together with a high degree of optimization on the relative architecture in its GPUs.

Since a GPGPU software developer in principle wants to optimize his/her codes to run them as fast as possible on a device, the most natural programming languages to work with CUDA are of *low-level* type (C, C++, Fortran). This made the approach to GPU programming very steep, because of the need to manage at low level every piece of the data and instruction flow in the various part of the device. For this reason in 2015 Google developers released, under Apache 2.0 open source license, a new *high-level* framework for GPGPU programming, based on Python. It uses the full set of CUDA libraries and it allows to work with a general compilation of the package for Python, or compile it specifically for a given GPU, to obtain a higher level of optimization.

3.1.1 Fundamentals of GPU computing

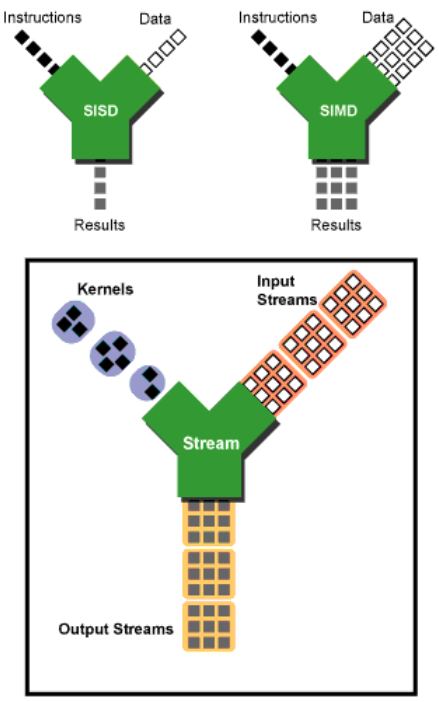

Figure 3.4: Schemes of three different programming paradigms (image from arstechnica.com).

The difference between serial and parallel computation consists in the application of instructions to a set of data together (Single Instruction on Multiple Data), instead of applying them once at a time (Single Instruction on Single Datum). GPU parallelization as a slight different paradigm called *stream processing*, well suited for *big data*.

In the stream processing paradigm the data are organized in blocks, often called *batches*; in a similar way the instructions are grouped together in *kernels*. Under the hypothesis that the instructions are completely non-sequential, the kernels are roughly inner loops where a set of operations are applied to all the data in a batch. Each invocation of a kernel is independent, allowing for the parallel execution on N different threads uniformly. Kernels and data of a single batch are then distributed over the core/memory topology of the GPU, where each core works with SIMD paradigm.

When the data of a batch passed on, the next one will be loaded on the GPU and instructions in kernels are ran on the new data. As one can easily understand, this approach mixes sequential and parallel computing. It is necessary since very often the data to manage from the storages are orders of magnitude greater than the memory capability of the used devices. In this context

it's important to manage in an intelligent way the input/output pipeline and the memory allocation.

Both in OpenCL and CUDA the kernels have to be defined separately and successively called in the code, with a proper assignment of memory and cores for the data computational needs, depending on the device used. High level libraries have pre-compiled kernels for their functions. For example, the application of a TensorFlow function on a set of data means that a kernel is invoked and ran on the data, with automatic assignment of the computations on the topology of the specific GPU(s) used.

In GPU programming, both high and low level, it's important to keep in mind that:

1. Every operation has to be thought intrinsically non-sequential. The GPU hardware is optimized to work with uniform streaming, introducing sequential instructions will be strongly sub-optimal;

2. The memory has to be managed wisely. If the code doesn't exploit well the GPU topology, every data transfer could hide bottlenecks in the bandwidth , with the outcome that only a little portion of the cores of the GPU are used together, while the big part of the device awaits new data.

From the big-data point of view, a good memory management is of primary importance. A GPU has several kinds of memory, but for numerical work the most useful ones are the *shared* and *global* memories. The GPU cores are organized in blocks each one with a shared memory, and every block communicates with the global memory (a common dynamic RAM module). The shared memory is 100 times faster in data transfer than the global memory but it is very small (only 48 kB) and it's used for tiny and very fast operations.

This is only for what concern the GPU memories, but we often need to load data from the storage or the host system memory to the GPU and vice versa. Every data transfer of this kind is a big bottleneck of the code execution, because of a bandwidth usually far smaller than the inner bandwidth of the GPU. Moreover it involves the CPU and slows the graphic units work in the continuous data synchronization.

Within the memory capability of the GPU system, the data should be loaded in the global memory, every needed operation has to be performed and the results stored, only once per batch. With multi-GPU systems the above prescriptions are quite similar; in addition one has to consider the communication between different GPUs, relaxing some optimization conditions if they are directly linked with a common memory.

At the end of this section it should be clear why the availability of a high level programming language that works on GPU is a great advantage. A single device has three different kinds of cores (shaders, textures, rendering), each one linked in different ways by two different kinds of memory (e.g. global, shared) with different size and, as we said before, different bandwidth. A code optimized for a device could be totally a mess with another device, and if one has various systems with different models, it would be necessary to manage the optimizations differently from board to board.

```
1  import tensorflow as tf
2
3  N = 100
4
5  # Build a graph with:
6  # two matrices with random gaussian values;
7  matrix1 = tf.random_normal((N,N))
8  matrix2 = tf.random_normal((N,N))
9
10 # a matricial product between them.
11 product = tf.matmul(matr1, matr2)
12
13 # Launch the graph in a session,
14 sess = tf.Session()
15
16 # evaluate the tensor 'product'
17 sess.run(product)
```

Listing 1: Example of a TensorFlow code.

3.1.2 TensorFlow

TensorFlow, developed originally for machine learning, has become a complete framework for numerical computation with its release, and now it is quite general to be used in various areas.

As previously mentioned, the main characteristic of TensorFlow is the high-level structure. The way it works in the execution of codes follows the natural operation of the GPUs with the stream processing mentioned above, and takes advantage of it with a full symbolic programming frame. It uses the dataflow paradigm, which is an equivalent definition of stream processing, where a program is modeled as a graph of operations, where the data *flow* through.

The central unit of data in TensorFlow are N-dimensional arrays called *tensors*. Operations on the tensors are represented by nodes in the graph (displayable by the TensorBoard visualizer, see figure 3.9), while the edges are the input/output tensors which link the operations of the flow. The graph is then a series of operations interchanging tensors: every instruction of the code is a symbol and acts logically as a function. It is not runned until it is specifically called by the interpreter.

A simple example can be the code used for the benchmark in Figure 3.2 (see Listing 1). When one wants to execute a code, it is necessary first to open a TensorFlow session (row 14) which will build the graph defined by the instructions before. The above code is represented with the graph showed in Figure 3.5

Subsequently it is possible to call the instruction that returns the values interested (row 17). When `product` is called, `matrix1` and `matrix2` are generated and the data are used by the next command: the two 100×100 edges go from the `matrix1` and `matrix2` nodes to `product`. On the evaluation the interpreter run in series every command in the graph which serves to evaluate the chosen node, like a chain of functions, using the GPU kernels already included in the built-in TensorFlow functions definitions.

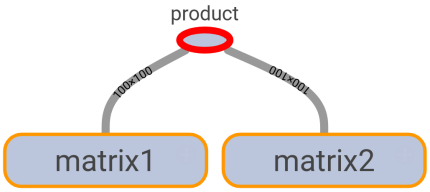

Figure 3.5: TensorFlow graph of the example code.

This is useful because in this way one can define and create tensors directly on GPU, or run instructions using only the GPU memory once the data are loaded there, involving the least possible the CPU, with the least possible data transfer between the GPU and the host system memory. If a TensorFlow code is well written, it will manage with high efficiency memory and core topology, with no need by the user to manage every low-level aspect of the computation. Moreover, the same Tensorflow code ran on a single GPU, with a few more code lines and little extra effort, can be extended on multiple GPUs very naturally. In fact TensorFlow is developed to work efficiently with multiple GPUs, thanks to the big-data and neural-network original purpose of the framework.

It's important to remark that in TensorFlow there are two kinds of tensors: *constant* and *variable*. The one used so far is the constant kind. When it is defined it becomes an uneditable node in the graph, which remains and occupies memory until the session is closed. Moreover, every operation on a constant is a new node. For example, an instruction on a loop cycle will open a new node at every iteration, filling the GPU memory. For this reason, when it is unavoidable a certain serialization there exists the variable tensors. A variable is still a node in the graph, but can be modified and updated. While a constant is thought for a direct single calculation which occupies at most the whole GPU memory at once, variables can be used for more complex tasks.

Despite it is surely young and sometimes unripe (version 1.0 was released in February 2017), it is experiencing a very fast development with exciting improvements and extensions, thanks also to the community contribution granted by the open source approach.

3.2 Hough transform

The so-called Hough transform is a method patented by Hough, 1962, for patterns recognition in pictures. It was conceived for the study of subatomic particle tracks in bubble chambers. At the time, to see the tracks of charged particles in the bubble chamber medium, they were photographed at each event, and a human trained observer used to spend hours analyzing each picture to distinguish the various tracks of an event. When the event rate became too high, it was necessary an automation system for the tracks recognition. The one proposed by Hough was to split the image of an event in sufficiently small sectors such that a curved track is divided with good approximation in line segments. For each sector he built an electronic device able to detect and store parameters of the straight lines.

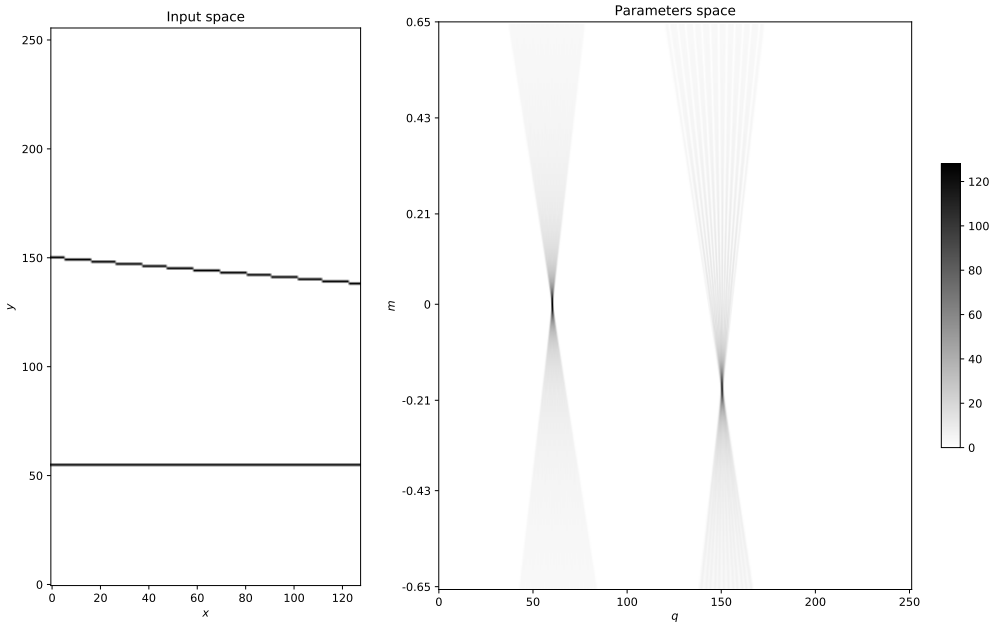

Figure 3.6: A Hough transform of two straight lines with different slope.

Considering an image where the line is formed by a series of colinear black points with a pure white background, the *detection* of a track using the Hough transform consists in the transformation of the bubble track picture in another picture (we can call it the Hough map), where single black points with coordinates (x, y) are converted in straight lines with slope and intercept given by the points coordinates in the input picture. If we take a sequence of points along a straight track with equation $y = mx + q$, they will be represented in the Hough map by straight lines changing slopes and intercepts and, by simple Euclid geometry consideration, they will form an intersecting family of lines. The coordinates of the incidence point, in the Hough space, will be the parameters values (m, q) of the line in the coordinate space (see Figure 3.6 for an example).

We can identify then the Hough transform space as the parameter space, according to the parameters of the pattern we are looking for in the input image. A problem working in the continuum is that, if we have a noisy and disturbed image, we have to inspect in the Hough map every couple of intersecting lines, to distinguish the intersections due to noise points from the actual signal colinear points in the input image. The computational effort will be unmanageable since it goes as n^2, with n the number of points inspected in the image. It is then necessary to impose a certain discretization in the parameter space (Duda and Hart, 1972).

In practice we always work with digitized images, then the parameter space is already discretized: using images with many non-zero valued pixels, the transform will return a Hough map with a certain resolution. The Hough map will be then a 2D histogram, where many lines may be binned in the same pixel and the pixel will have a count depending on how many lines entered in that parameter space region. Then a given pixel in the map records the number of lines passing through it. The count amplitude on each pixel will give informations on the significance of a point

in the parameter space, within the error given by the image resolution. In this way instead of inspecting every line intersection, we simply count the amplitude of the Hough pixel, with a much lower computation effort.

The Hough transform works also with input images which pixels have not only binary values but, e.g, a gray scale of values between 0 and 1. In this case a point in the image is transformed to the Hough map with a weight.

The resolution of the Hough map can be enhanced with respect to the input image, in order to give a better precision on the parameters estimation. Or, conversely, if the computation is too demanding, it is possible to reduce the map resolution to simplify the transformation (at the price of a greater uncertainty on the parameters estimation).

Moreover, it's possible to generalize the Hough transform beyond the straight line pattern recognition: in general we can use an N-dimensional manifold as input of the transform and an M-dimensional parameter space as output, searching for other curves than a straight line (Duda and Hart, 1972, Ballard, 1981). This characteristic makes the Hough transform suitable for many different applications and is one of the reasons of the success and diffusion of the method.

3.2.1 Frequency-Hough transform

The current implementation of the Frequency-Hough is a MATLAB code which runs on CPU. Here I will explain how this algorithm works and its main features, taking into account an optimization valid only for computing on CPU. Further i will present the GPGPU implementation, specifying when a choice made for the original code could be changed to obtain a better performance on GPU.

As we said in section 2.3.1, the Frequency-Hough transform starts with an input peakmap where the Doppler effect due to the Earth's motion is already corrected.

The transform is from the peakmap frequency/time plane to the gravitational waves frequency/spin-down parameters plane. With *intrinsic frequency* ν_0 and spin-down $d = \dot{\nu}$ as parameters of a given neutron star waveform, the expected path in the peakmap at first order is

$$\nu = \nu_0 + d(t - t_0), \tag{3.1}$$

where t_0 is an arbitrary reference time. Similar to the transform proposed by Hough, each point in the peakmap with coordinates $(t - t_0, \nu)$ is transformed into a straight line in the Hough (ν_0, d) parameters plane (Antonucci et al., 2008):

$$d = -\frac{1}{t - t_0}(-\nu_0 + \nu)$$

Since the slope of these lines depends on the reference times, usually it's convenient to choose t_0 as the half length of the observation time. In this way peaks on a same row will be transformed on the Hough matrix in lines which span symmetrically with respect to the reference frequency ν_0 column, reducing the uncertainty of the parameters .

Keeping in mind that the frequency bins in the peakmap have a certain width $\Delta\nu = 1/T_{FFT}$, a peakmap point is actually transformed into a stripe between two parallel straight lines:

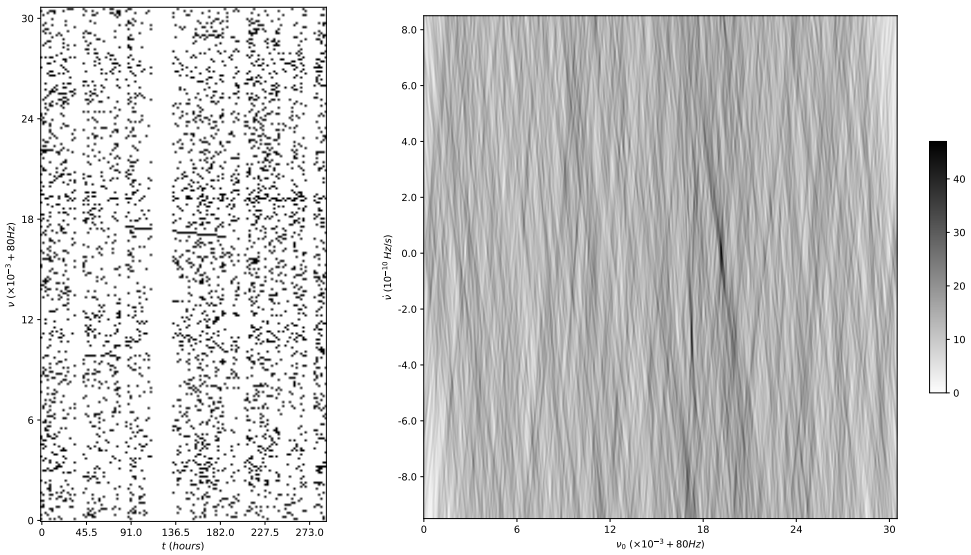

Figure 3.7: An example Frequency-Hough transform of a peakmap with an injected signal from (Muciaccia, 2017) Thesis. The peakmap is computed with data extracted from the Virgo-LIGO $O2$ run, with $T_{FFT} = 8192\,s$. The signal is a simple long transient from $t \sim 86\,h$ and $t \sim 189\,h$. Looking the $\dot{\nu} = 0\ Hz/s$ row of the map, it can be seen how the transform could be vulnerable to disturbances or fluctuations of the noise that create peak patterns. This is the reason because verifications and follow-up on candidates are necessary.

$$d_{\pm} = \frac{1}{t - t_0}(-\nu_0 + \nu \pm \Delta\nu/2)$$

Using this fact the Frequency-Hough is implemented with a differential method, more efficient when the frequency resolution in the parameter space is enhanced with respect to the input space. Which can be summarized as follow:

- the map is initialized as a matrix with as much frequency bins as the peakmap, and with the number of spin-down bins we want to inspect;
- in a loop, the values of the matrix elements corresponding to the relation $\nu_0 = \nu - \Delta\nu/2 - (t - t_0)d$ are incremented by 1;
- once the elements returned by the above relation are found and incremented, it is straightforward and computationally efficient, in high-level frameworks, to use array slicing and decrement by 1 the elements corresponding to the relation $\nu_0 = \nu + \Delta\nu/2 - (t - t_0)d$;
- once the *differential* map is computed and populated, every row is cumulatively summed along the frequencies.

An example of the explained Frequency-Hough transform computation is shown in Figure 3.7.

If we want more precision in the parameters, fixed the maximum and minimum values of d and ν_0, it's possible to increase the resolution on both parameters, at

the price of an increment of the Hough map size. A common choice is to define a new frequency resolution $\Delta\nu_0 = \Delta\nu/10$, which grants 8% higher pixel amplitude (Antonucci et al., 2008), thanks to the reduction of the digitization effect, with a minor computational increase since the greater part of the load is the creation of the differential map. Regarding the spin-downs, the most natural choice for the bin width is $\Delta\dot{\nu} = \Delta\nu/T_{obs}$. Enhancing this resolution will bring a very little improvement in the amplitude of the pixels, since the map goes to only 3.6% discretization amplitude loss with the natural spin-down binning, to $\sim 2\%$ with an enhancement factor 2, but at the cost of a big increase in the computations necessary to populate the differential map (Antonucci et al., 2008).

It's important to notice that the spin-down of a neutron star could depend on many phenomena, and the function $\dot{\nu}(t)$ could have in principle a complicated form, even neglecting discontinuities associated to neutron star *glitches*. Since the typical values of a neutron star spin-down are quite small, however, a Taylor expansion stopped at first order is often a good approximation, but higher order spin-down terms can be necessary for specific cases, e.g. with very young objects. This, of course, comes at the price of an expansion of the parameter space, with a corresponding increase of the computational cost.

3.2.2 The GPGPU algorithm for the Frequency-Hough transform

The development of the new Frequency-Hough algorithm started from the MATLAB code within the SNAG Toolbox (Frasca and D'Antonio, 2017). The first step of the GPU porting was to write a first fully vectorized version with Scipy and Numpy libraries (Scipy, 2017). Numpy has a sintax similar to MATLAB, and TensorFlow takes many aspect of the Numpy sintax, then it was a useful approach to add progressively more complexity to the development.

Writing a code in a high level programming language, vectorization is a vital step. Using a library with functions well developed and compiled in a low level language will be always faster than a custom function with similar instructions. More importantly, a good numeric library acts in a similar way on scalars and on arrays, applying efficiently a single instruction on a big amount of data in a way extremely more efficient than, for example, a Python for-loop. The first goal of the development work I completed was indeed the removal of any kind of loop.

The successive step was the migration to TensorFlow. Here, the memory management was crucial. The original code uses data from `.mat` files with the peakmaps in a `float64` sparse format: since many elements of a peakmap are zero, usually it is useful to store only the coordinates of the peaks instead the whole matrix. In the case of a peakmap with N_{peaks} non-zero elements and size $N_{row} \cdot N_{col}$, this happens if $64^2 N_{peaks} < N_{row} N_{col}$, relation always true with the threshold chosen to select the peaks and the typical time-frequency ranges of a continuous waves search. Since I worked on only one GPU with $\sim 5\,GB$ memory, it was necessary to preserve the sparse structure of the input matrices. This could create problems if time or frequency arrays values exceed the precision of `float32` data type: only workstation GPUs have `float64` precision, and it is still very suboptimal with

Table 3.1: Time computation comparison between the MATLAB Frequency-Hough code running on CPU and the TensorFlow code on GPU, using the peakmaps from the input files used for the analysis, generated with data from the 9 months long $O2$ LIGO run: for the $T_{FFT} = 8192\,s$ frequency band we have a peakmap for each $1\,Hz$ between 10 and $128\,Hz$; for the $T_{FFT} = 4096\,s$ peakmaps are large $5\,Hz$ from 128 to $512\,Hz$. Each peakmap in both frequency bands covers the whole run duration. The spin-down range was the same for both frequency bands ($[-10^{-9}, 10^{-10}\,Hz/s$), bringing respectively to 210 and 105 bins. The devices used are a Intel Xeon E5-2620 CPU with twelve $2.40GHz$ cores and a NVIDIA Tesla K20m with ~ 2500 cores at $706\,MHz$.

	$T_{FFT} = 8192\,s$	$T_{FFT} = 4096\,s$
MATLAB with Xeon E5-2620	14.2 s	32.5 s
TensorFlow with Tesla k20 (32 bit)	0.72	1.73 s

respect to the 32bit optimization of the hardware. Moreover, 64bit data takes more space in memory, limiting the parallelization capability of the code.

To preserve the higher computation power of the GPU, all the data were loaded from a `.mat` file with the Scipy function `scipy.loadmat` in the host RAM as Numpy (`float64` by default) arrays, and then passed to TensorFlow with tensors definition where the `float32` data type was imposed[1]. A better input pipeline can be defined, using out of memory data load with appropriate file format and libraries. This is deferred to future work.

The highest challenge in the GPU vectorization of the code has been the parallelization on spin-down. To limit the memory usage on the single GPU, since every spin-down row in the Hough map is created independently, I solved the problem using the `tf.map_fn` function: it applies a function along every element of a (possibly nested) tensor, parallelizing the instruction over the GPU. The integration with `tf.cumsum`, instead, is intrinsically sequential so it becomes rapidly the most inefficient part of the code when the frequency resolution is enhanced: if with `enhancement = 1` it represent $\lesssim 10\%$ of the total computation time, with `enhancement = 10` it raises to the 33%.

Despite some inefficiencies, the GPU Frequency-Hough code runs on a Tesla K20 very fast, with a factor of ~ 20 speed-up with respect to the MATLAB code, as shown in Table 3.1. In Figure 3.8 can be seen a more accurate benchmark. In Listing 2, I show the commented code of the TensorFlow Frequency-Hough function.

[1]Since the data type management is subtle with GPU programming, TensorFlow needs type consistency in the data used for each function, and allows to specify the type output for almost all functions and tensors definitions.

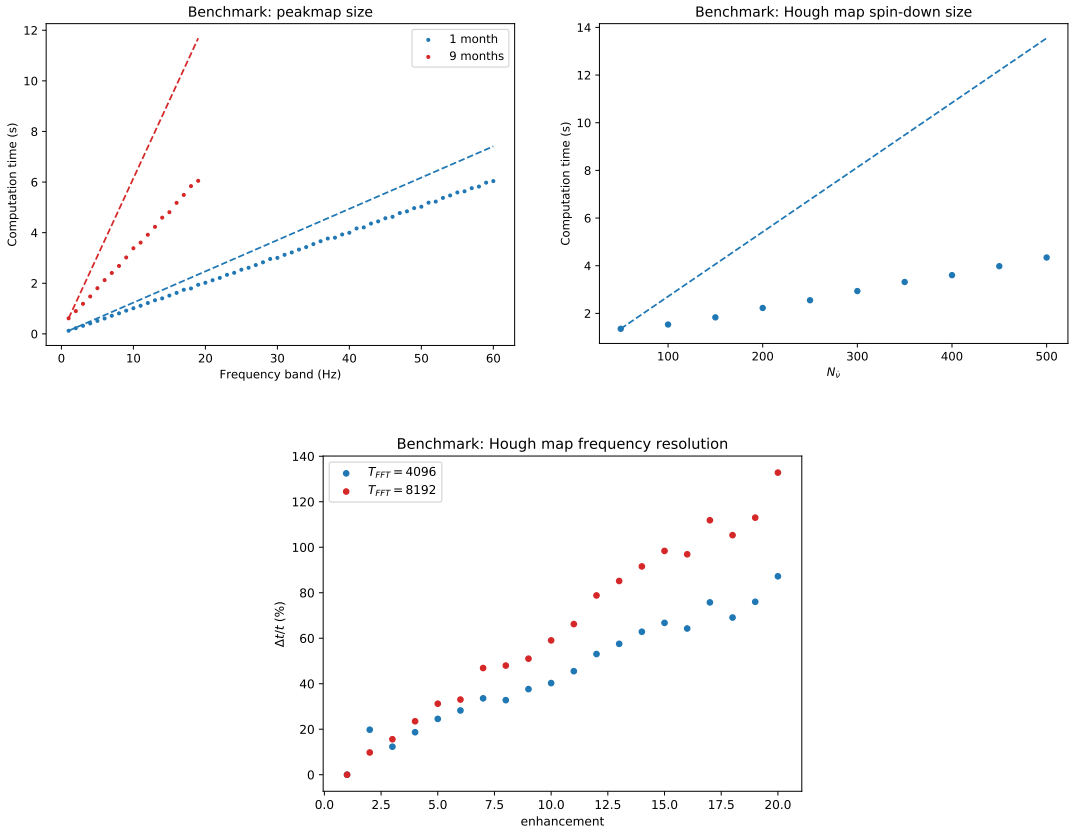

Figure 3.8: GPU benchmarks on some key parameters of the Frequency-Hough algorithm with $T_{FFT} = 4096\,s$. Top left: computation time in function of the frequency band covered by the peakmap. The actual computation times (the dots) are compared to the one I'd obtain serializing the GPU Hough over $1\,Hz$ peakmaps (dashed lines). The red plots show the halving serial to parallel computation time with a 9 months long peakmap, but with a faster filling GPU memory. The blue plot comes from a 1 month long peakmap, with a 20% time gain. Top right: computation time in function of the number of spin-down step of the Hough, with $T_{obs} = 9$ months and $\Delta\nu = 5\,Hz$. Dots and dashed lines have the same meaning explained above, then the plot shows a 68% time reduction with respect the serialized case, proving that the vectorization with the `tf.map_fn` function is successful. Bottom: relative time increase of the Hough map computation, increasing the frequency resolution enhancement factor. Unlike the CPU algorithm, the additional computational load is no longer negligible. The reason is that the integration step is intrinsically sequential and can't be efficient on GPU.

```
def frequencyHough(nu,t, nuDot, w, numRows, numColumns):
  """
  Returns the Frequency-Hough transform of a sparse peakmap.
  Parameters:
  nu, t : 1D tensors
    The coordinates of the peakmap in sparse format.
  w :  1D tensor
    The values of the peaks.
  nuDot : 1D tensor
    The spin-downs values over which calculate the Hough transform.
  numRows, numColumns : int scalars
    Size of the Hough map
  Returns:
  houghMap : 2D tensor
    The Hough transform matrix
  """
  # The frequency resolution enhancement in the map.
  enhancement = 10
  # A certain number of columns is added to avoid
  # frequencies to be transformed outside the image;
  # they will be removed once the Hough map is computed.
  securbelt = 4000
  numColumns = numColumns + securbelt

  # This function computes the transform histogram for a given spin-down
  def rowTransform(ithSD):
     sdTimed = tf.multiply(nuDot[ithSD], t)
     transform = tf.round(nu-sdTimed+securbelt/2)
     transform = tf.cast(transform, dtype=tf.int32)
     # The rounding operation brings a certain number of peaks in the same
  ↪  frequency/spin-down bin in the Hough map; the left edge is then
  ↪  computed binning that peaks properly;
     # this is the core of the code, with the most computational effort
     values = tf.unsorted_segment_sum(w, transform, numColumns)
     return values
  # To keep under control the memory usage, the map function is a
  # very useful tool to apply the same function over a vector
  # in this way the vectorization is preserved.
  houghLeft = tf.map_fn(rowTransform, tf.range(0, numRows),
                        dtype=tf.float32, parallel_iterations=10)
  # Let's superimpose the right edge on the image
  houghRight = tf.subtract(houghLeft[:,enhancement:numColumns],
                           houghLeft[:,0:numColumns - enhancement])
  houghDiff = tf.concat([houghLeft[:,0:enhancement],houghRight],1)
  # Finally, the Hough map is computed integrating along the frequencies
  houghMap = tf.cumsum(houghDiff, axis = 1)
  return houghMap
```

Listing 2: Source code of the GPU Frequency-Hough algorithm written with TensorFlow.

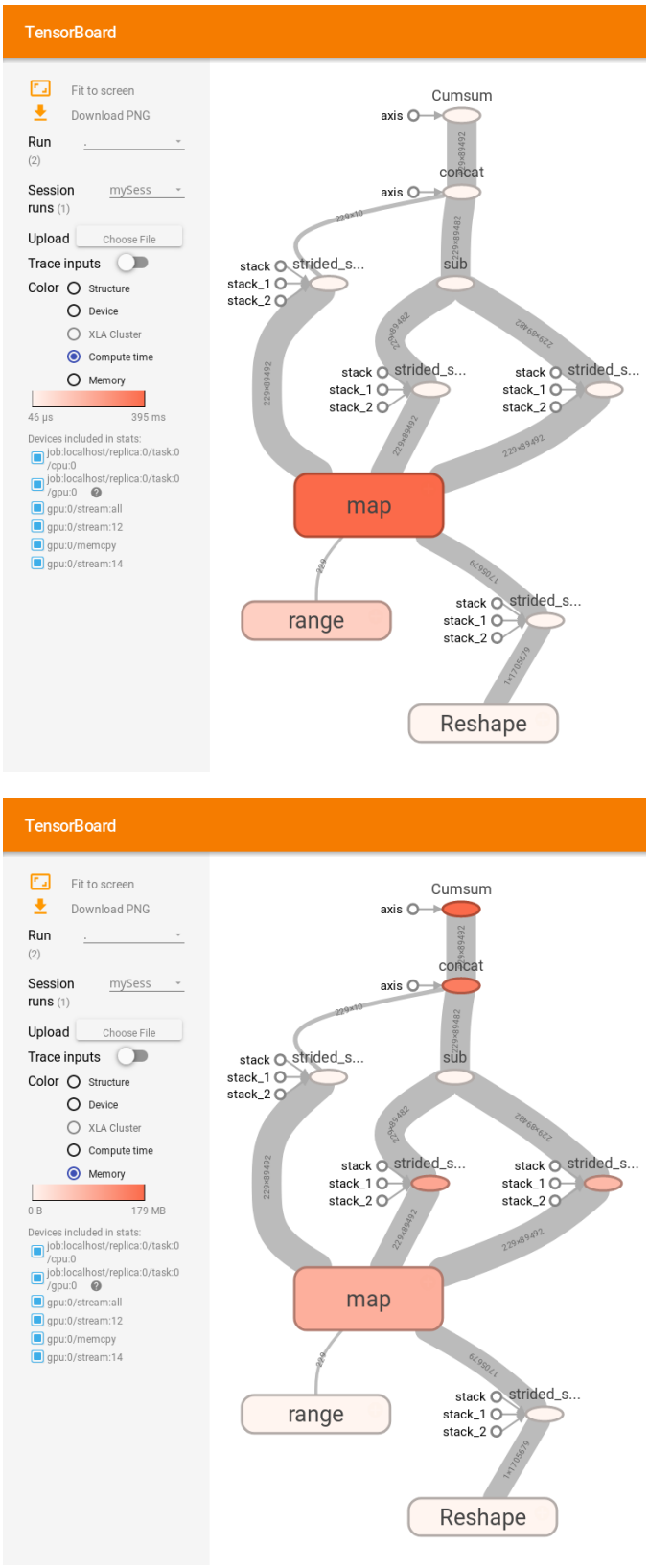

Figure 3.9: The TensorFlow graph of the Frequency-Hough code. It is an useful tool to profile memory and processors usage and check if part of the code goes to CPU because doesn't have a GPU kernel.

3.3 Analysis

Once the GPU algorithm for the Frequency-Hough transform was ready, it has been used for an analysis on real data, with the purpose to test it with a simplified version of the all-sky search. The data used are those from the nine months long $O2$ run of the two LIGO detectors. I chose a delimited portion of the sky corresponding to the Galactic Center region and ran, for each selected sky point, Doppler correction on the peakmaps, Hough transform, candidate selection and candidate coincidences. Both Doppler correction and candidate selection have been carried out with a simple `numpy` porting of the SNAG functions `hfdf_patch` and `hfdf_peaks`.

The data are organized in *input files*, MATLAB structures in `.mat` format, containing many informations, including the non-corrected peakmap and the velocity vectors of the detector for every FFT used to create the peakmap, essential to the Doppler correction. For convenience the peakmaps are computed for the whole observation time in various frequency bands and stored in different input files. We have an input file for every 1 Hz below 128 Hz (i.e. for $T_{FFT} = 8192\ s$), and every 5 Hz above. This file multiplicity inevitably slows the analysis, because it forced me to serialize over the different files and compute the analysis every 1-5 Hz. A way to improve the efficiency could be to use bigger peakmaps and, with limited GPU memory availability, an out-of-memory progressive input-output pipeline.

Once the candidates of the whole frequency band are selected from the data, they are stored in a different `.mat` file for each detector. Finally, an estimate of the search sensitivity has done using coincidences between the candidates from the two detectors.

3.3.1 Parameter space choice

To give an exhaustive test bench for the GPU Frequency-Hough algorithm, I used the same choices typically done in an all-sky search. Regarding the parameter space, however, it was necessary to narrow the domain to specific intervals in the respective physical dimensions, in order to limit the total computing time.

Sky position

Recent studies show that an interesting region of the sky for a direct search of continuous waves search could be the Galactic Center, where many observations suggest that it could be present a large population of neutron stars, mainly in the millisecond pulsar class (Bartels, Krishnamurthy, and Weniger, 2016, Rajwade, Lorimer, and Anderson, 2017). Following Rajwade, Lorimer, and Anderson, 2017, a promising area appears to be the square delimited by $\pm 10°$ in galactic longitude and latitude. This area is enough large to give also an idea on the difficulties that an all-sky search with GPU must take into account.

In fact, looking toward different sky positions means different Doppler correction, Frequency-Hough and candidate selection, for every point identified by the chosen resolution on ecliptic coordinates. A well defined sky grid is fundamental to make sure that the uncertainty of the Doppler correction frequency shift is smaller than

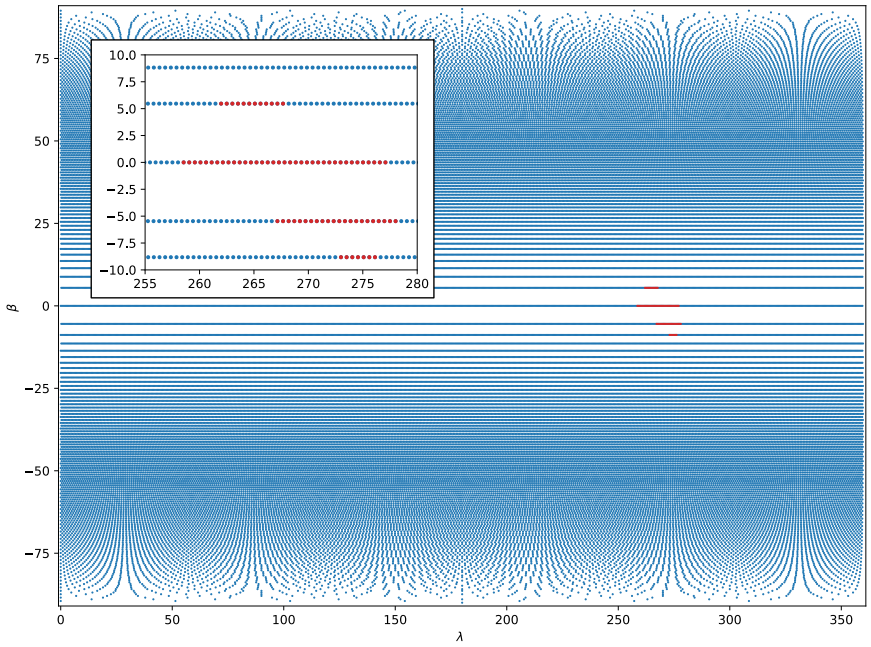

Figure 3.10: A scatterplot of the sky grid in ecliptic coordinates computed for $T_{FFT} = 8192\ s$. In red is drawn the chosen region for the analysis, corresponding to the $\pm 10°$ square in galactic coordinates, around the galactic center.

the peakmap frequency bin. If that weren't the case, then the Doppler correction error could lowers significantly the amplitude of a signal in the Hough map. This brings to a high number of sky position to analyze, providing both the main computational and memory load if one want to parallelize this parameter space dimension, and forcing to the best compromise from parallel and serial computation.

Using the criterion explained in (Antonucci et al., 2008) and (Astone et al., 2014) the resolution in ecliptic coordinates (λ, β) is such that two nearby sources with same emitting frequency ν_0 are distinguished, if the different Doppler correction between the two position would produce a frequency difference greater than the frequency resolution: $\Delta\nu_D > \delta\nu = 1/T_{FFT}$. The dimension of a sky patch will be determined then by the two equations

$$\begin{cases} \delta\lambda = \frac{1}{N_D cos\beta} \\ \delta\beta = \frac{1}{N_D sin\beta} \end{cases} \tag{3.2}$$

The *Doppler number* N_D is defined as

$$N_D(\nu_0) = \nu_0 T_{FFT} \omega_{orb} R_{orb}/c \sim 10^{-4} \nu_0 T_{FFT}, \tag{3.3}$$

where ω_{orb} and R_{orb} are the average orbital parameters of Earth. Note that given β, in equation 3.2 the width in ecliptic longitude is uniform. Conversely, the width in latitude depends on the angular distance from the ecliptic.

To simplify the analysis I chose to impose ν_0 in equation 3.3 as the maximum frequency of each used T_{FFT} band (8192 s and 4096 s as we'll see in the next section). In this way I worked with only one sky grid for every frequency band chosen, obtaining an effective enhanced resolution with a higher number of points

in the sky. With the enhanced sky resolution the small discretization reduction in sensitivity loss (Antonucci et al., 2008) comes with a strong increase of the computation load. A plot of the sky region inspected can be seen in Figure 3.10.

Frequency

Although it has been hypothesized that the Galactic Center neutron star population consists mainly in millisecond pulsars, I chose to run the analysis only over the first two frequency bands, with $T_{FFT} = 8192\,s$, $4096\,s$, respectively corresponding to $\nu_{8192} \in [10; 128]\,Hz$ and $\nu_{4096} \in [128; 512]\,Hz$, for two reasons. The first is that the best sensitivity region of the two detectors is between the end of the first band and the begin of the second; secondly, clues on the presence of a large population of unseen millisecond pulsars comes with a prevision of a smaller number of slower pulsars ((Bartels, Krishnamurthy, and Weniger, 2016)). Finally, as saw in Equation 3.2, the appropriate sky resolution for frequencies above $512\,Hz$ raises very much the number of points for which the Hough transform has to be calculated. However, a rough estimate of the computation time needed to process the input files from $512\,Hz$ to $2048\,Hz$ is around 170 hours with the single Tesla K20, then the analysis in this frequency band is scheduled in the near future.

As already said, the resolution in frequency increases as the duration of the time series used. A frequency bin in the peakmap is of $\delta\nu = 1/T_{FFT}$, while in the Hough map the resolution enhancement by a factor 10 has been maintained in this analysis, despite the significant computation efficiency loss (Figure 3.8): $\delta\nu_H = \delta\nu/10$.

Since the size of the peakmap scales quite well with the GPU parallelism, the GPU Frequency-Hough could support easily higher coherence times, losing efficiency only in the integration step and in memory usage (keep in mind that the more memory is occupied by a single Hough map operation, the lesser we can parallelize over the time-frequency space)

Spin-down

The chosen spin-down range goes from a reasonable maximum value for typical isolated neutron stars, $\dot{\nu}_{min} \sim -10^{-9}\,Hz/s$ to a small spin-up around $\dot{\nu}_{max} \sim 10^{-10}\,Hz/s$, taking into account that an isolated star could accrete a small quantity of angular momentum from the surrounding environment.

As previously said, the natural choice for the spin-down binning is a width which variation corresponds to a variation of only one bin in frequency during the whole observation time: $\delta\dot{\nu} = \delta\nu/T_{obs}$. This choice corresponds to 210 spin-down bins with $T_{obs} = 9$ months and $T_{FFT} = 8192\,s$, 105 bins with $T_{FFT} = 4096\,s$. A higher resolution in spin-down can be allowed, with small computation time increase, but raising significantly the memory load of the Hough map, giving a small benefit from the reduction of the digitization effect. More rewarding from the point of view of the analysis results could be to expand the parameter space with a larger spin-down range, with similar computation load effects.

3.3.2 Procedure

The logical scheme of the analysis procedure carried out is:

1. generation of the sky grid and selection of an interesting region;
2. core analysis, consisting in Doppler correction of the peakmaps using the input files, Frequency-Hough, candidate selection;
3. coincidences of candidates from the two detectors;

In the following, these three steps are described in detail.

Sky region selection

The sky grid has been generated using the SNAG function `pss_optmap`. The output of the function is an array with five columns: (λ, β, $\delta\lambda$, $\delta\beta^-$, $\delta\beta^+$). The $\pm 10°$ region in both galactic coordinates has been selected with a MATLAB script using the SNAG function `astro_coord_fix` to convert the whole coordinate grid in galactic coordinates, leaving untouched the widths (the last three columns). The array rows such that $|\lambda_{gal}| < 10$, $|\beta_{gal}| < 10$ are picked with the MATLAB function `find` and finally the coordinate columns are converted back in ecliptic coordinates. The sky patches for $T_{FFT} = 8192\ s$ and $4096\ s$ are computed only once and stored in a `.mat` file, in order to load them with the core analysis code in Python. The selection resulted in 78 sky positions for $T_{FFT} = 8192\ s$ and 254 for $T_{FFT} = 4096\ s$.

Core analysis

Actually, the first aim was to integrate the Hough TensorFlow code in the existing MATLAB pipeline, but this is at the moment infeasible due the extremely low integration between MATLAB and Python. It has been necessary then to develop, in addition to the Frequency-Hough code, Pythons function also for the Doppler correction and candidate selection.

Since a GPU implementation of the whole pipeline was beyond the purpose of this Thesis, I simply translated the SNAG functions `hfdf_patch` and `hfdf_peak` to Python using Numpy, with no optimization and parallelism. Unfortunately the Doppler correction is strongly interconnected with the Frequency-Hough transform, so I was forced to work serially over sky positions.

The Doppler correction and Frequency-Hough transform has been tested on three hardware injection of signals in the data (see Figure 3.11), with parameters as shown in Table 3.2.

Table 3.2: Hardware injections parameters.

	pulsar_3	pulsar_5	pulsar_8
$\nu_0\ (Hz)$	108.89	52.81	194.31
$\dot{\nu}(Hz/s)$	$-1.46 \cdot 10^{-17}$	$-4.03 \cdot 10^{-18}$	$-8.65 \cdot 10^{-9}$
$\lambda\ ()$	193	277	338
$\beta\ ()$	-31	-61	-27

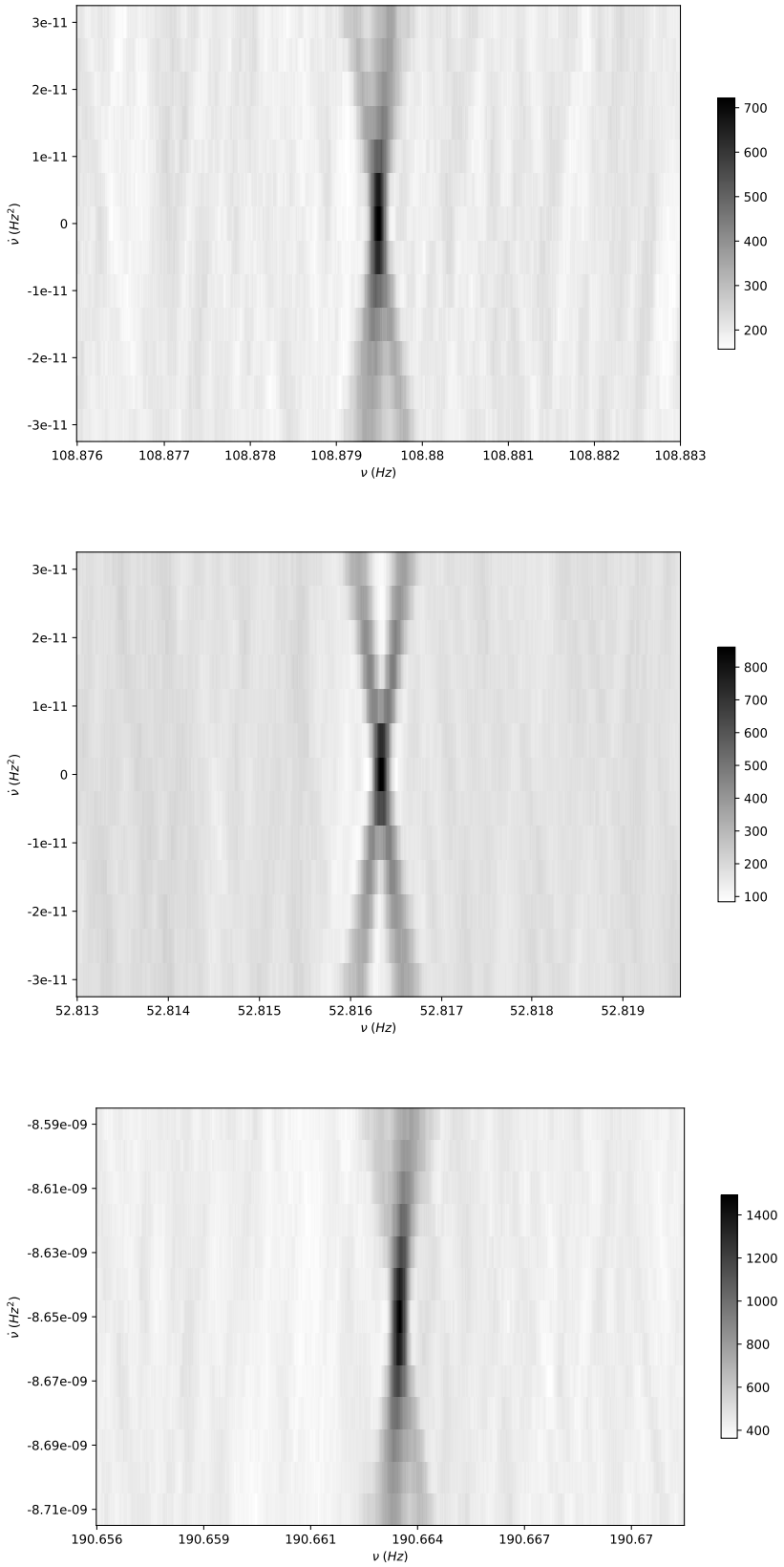

Figure 3.11: The GPU Frequency-Hough algorithm applied on three hardware injections, with parameters as showed in Table 3.2. Top: pulsar_3. Mid: pulsar_5. Bottom: pulsar_8.

The typical all-sky search choice in the candidate selection is to split the Hough map in stripes large $0.1\,Hz$ for every FFT length class, and in those stripes the pixel with maximum amplitude is searched. With the purpose to lighten up a future follow-up analysis, I chose to enlarge this width to $0.5\,Hz$ for $T_{FFT} = 4096\,s$, because the larger frequency range and the higher number of sky positions. This has the effect to systematically raise the CR of the candidates: wider stripes means that the maximum is searched on a larger number of pixel, and the selected one is the greatest of the five maxima I could find with a stripe width of $0.1\,Hz$. Remembering figure 2.9, higher CR means a small decrease of the false alarm probability at the cost of a small decrease of the search sensitivity.

The core analysis can be summarized in this simplified version of the used code:

```
...
# First of all i define some job parameters:FFT length, frequency step;
#  spin-down range, step, array values.
...
# Loading data for doppler correction,
square = scipy.io.loadmat('PATH/TO/square')['square']
numPoints = square.shape[0]
# loading list of input files (with the peakmaps).
fileslist = numpy.array(glob.glob('PATH/TO/DATAFOLDER')).sort
numFiles = fileslist.size
# Initialization of candidates array.
numCands = #the value chosen
candidates = numpy.zeros((numFiles,numPoints,numCands*2,9))

# Loop on the files of the FFT frequency band chosen.
for ithFile in numpy.arange(numFiles):
        # Loading the ith .mat input file and the useful data,
        nu, t, v = load_data(fileslist[ithFile])

        # loop into every sky positions in the area chosen.
        for point in numpy.arange(0,numPoints):
                # Doppler correction;
                nuCorr = dopp_corr(point)
                # now let's use TensorFlow: defining tf constants,
                nuHM, tHM, nudotH;  = tf_const_definitions(nuCorr,t,nudot)
                # calculating the Frequency-Hough transofrm,
                houghmap = frequencyHough(point,nuHM, nudotHM, tHM)
                session = tf.Session()
                # running a TensorFlow graph returns a Numpy NDarray.
                hough = session.run(houghmap)
                tf.reset_default_graph()
                session.close()
                # Candidates selection.
                candidates[ithFile,point] = candSel(numCands, nuCorr,
 ↪  hough, patch[point])
# Finally i save the whole candidates set to a .mat file.
```

Some notes on the above code:

- the analysis is ran once for $T_{FFT} = 8192\ s$ and once for $T_{FFT} = 4096\ s$ for each LIGO detectors. In the first frequency band with $\nu \in [10, 128]\ Hz$ it took almost 4 hours, while in the second ($\nu \in [128, 512]\ Hz$) around 20 hours due to the higher sky resolution and bigger frequency band;

- the input files are in a file format which can't support a complete out-of-memory input pipeline, so I was forced to cycle over them. A way to improve efficiency reducing the files serialization could be simply to build input files with bigger peakmaps and, better but more difficult, implement a proper input-output pipeline with a different file format;

- TensorFlow doesn't overwrite a node of the graph if an operation is in a cycle. Every iteration with constant definitions and operations on them creates new nodes, occupying memory for every returned value of the iterated instructions. For this reason I reset the graph and close the session at the end of the cycle, unfortunately lowering efficiency because TensorFlow has to open a new session at each iteration, taking some time. To avoid this, if a certain sequencing is unavoidable, the code should use only variables which values are updated in the iterations;

- every single value returned by the new codes has been compared with those returned by the original SNAG codes, in order to have a total exact match both in Doppler corrected frequencies, in Hough maps and in selected candidates;

- the times array with $T_{obs} = 9$ months exceeded the 32 bit floating point precision capability, causing a difference of $\sim 0.4\%$ in the amplitude of the Hough map with respect the original one computed with 64 bit sparse peakmaps. In order to satisfy the previous point for the Hough map, despite a little difference in the actual amplitude values, I chose to switch the time and spin-downs tensors to `float64`. The only numerical operation which ran with `float64` data was the simple $\dot{\nu}_i \times \vec{t}$ product (row 27, Listing 2): since values of the spin-down array have a floating point precision well below the 32 *bit* threshold, I could cast the product returned values to `float32`. Consequently, the computation time of `frequencyHough` on the used device increased by a factor about 10-20%. Remember that only workstation GPUs can work with `float64`, still with a much lower FLOPS capability with respect to 32 *bit* data (Figure 3.2), then the `float32` data type is mandatory with GPU computation. A different dimension of the peakmap (shorter along times, longer along frequencies) or a progressive data input and map computation can help. Anyway, in further tests it should be studied better how the times are truncated passing from 64 to 32 bit precision with detectors runs long about one year, and how candidates selection can be altered.

Coincidences

Once every input files of a selected frequency band is analyzed for every sky points in the chosen region, the candidates selected from a detector are saved all

together in memory storage as `.mat` files, to limit as far as possible the read/write operations since they hold only a few MB and are easy to manage. From this point on, every operation on the candidates can be done within MATLAB.

For the coincidences step I used the SNAG function `coin_candidates_direc`, storing the coincident candidates in files ready to future verifications and follow-up analysis. In both frequency bands the coincident candidates are $\sim 2\%$ of the candidates selected using the data from only one detector.

3.3.3 Results

Without any deeper analysis on the coincident candidates, I computed the search sensitivity and the false alarm probability, using the noise spectral amplitude of the both detector for the O2 run and two different threshold choices on CR:

1. a threshold, which value corresponds to have, in average and in a given FFT frequency band, only one candidate due to noise fluctuations. This hypothesis gives a low false alarm probability with a higher CR_{thr}:

$$P_{fa} = \frac{N_{cand}}{N_{tot}} = \frac{1}{N_\nu N_{\dot\nu} N_{sky}}$$

The corresponding CR_{thr} for a given P_{fa} comes from Equation 2.17:

$$CR_{thr} = \sqrt{2}\,\mathrm{erfc}^{-1}(2P_{fa}) = \sqrt{2}\,\mathrm{erfc}^{-1}(\frac{2}{N_\nu N_{\dot\nu} N_{sky}});$$

2. another hypothesis is instead that we have only faint signals with low CR and then we set a low CR_{thr} in order to keep them, while the highest CR candidates may come from disturbances in the data. We can select then the minimum CR from the candidates in a given frequency range. Since the sensitivity curves I used have a resolution of 0.125 Hz, I chose this range. This implies, of course, a better sensitivity at the cost of a larger number of surviving candidates which must be followed-up

In Figure 3.13 are shown the differences on the number of candidates to follow-up and respective false alarm probability, while in Figure 3.14 it can bee seen how changes the resulting search sensitivity. Remember that the estimation of search sensitivities following Section 2.3.3 is based on the hypothesis such that the noise follows a Gaussian probability distribution. This is not generally true, so the $h_{0_{min}}(\nu)$ showed are optimistic values, while the real search sensitivity can be computed using software injection and generally are expected to be slightly worse than the ones in Figure 3.14.

Using Equation 1.23, we can express the minimum detectable wave amplitude, in terms of minimum ellipticity of a source:

$$\epsilon = \frac{c^4 r}{4G\omega^2}\frac{1}{I_3}h_0$$

A parameter in the formula is the moment of inertia of the neutron star along the rotational plane I_3, which is obviously related to the density of the object. Under

certain hypothesis on the nature of the matter inside the star, various equations of state have been proposed, with slight differences on the density that the degeneracy pressure of the matter can sustain. This brings to a range on the moments of inertia allowed by these models, which can be from $\sim 10^{38}\, kg\, m^2$ to $\sim 5 \cdot 10^{38}\, kg\, m^2$ with more exotic equation of state (Johnson-McDaniel, 2013), for a mass range $1 - 2.5 M_\odot$. Assuming moments of inertia within this range, the minimum detectable ellipticity values are shown in Figure 3.12.

It is interesting to compare these values to the maximum values of ellipticity predicted by various equations of state (Johnson-McDaniel and Owen, 2013). It results that, excluding frequencies below $20\, Hz$, the sensitivity of this search might allow to detect signals from neutron stars with a wide range of maximum permissible ellipticity for various masses and equations of state. For instance, more standard equations of state admits ellipticity up to $\sim 10^{-7}$-10^{-5}, with masses between 1.2 and 2 $M_\odot$, while more exotic models with a hybrid hadron-quark matter can reach $\epsilon \sim 10^{-3}$.

Since the plot in Figure 3.12 shows the minimum detectable ellipticity, the search sensitivities computed allow to detect sources at higher frequencies with a wide variety of equation of states, including the more standard ones. At lower frequencies, instead, only exotic matter neutron stars could be detected.

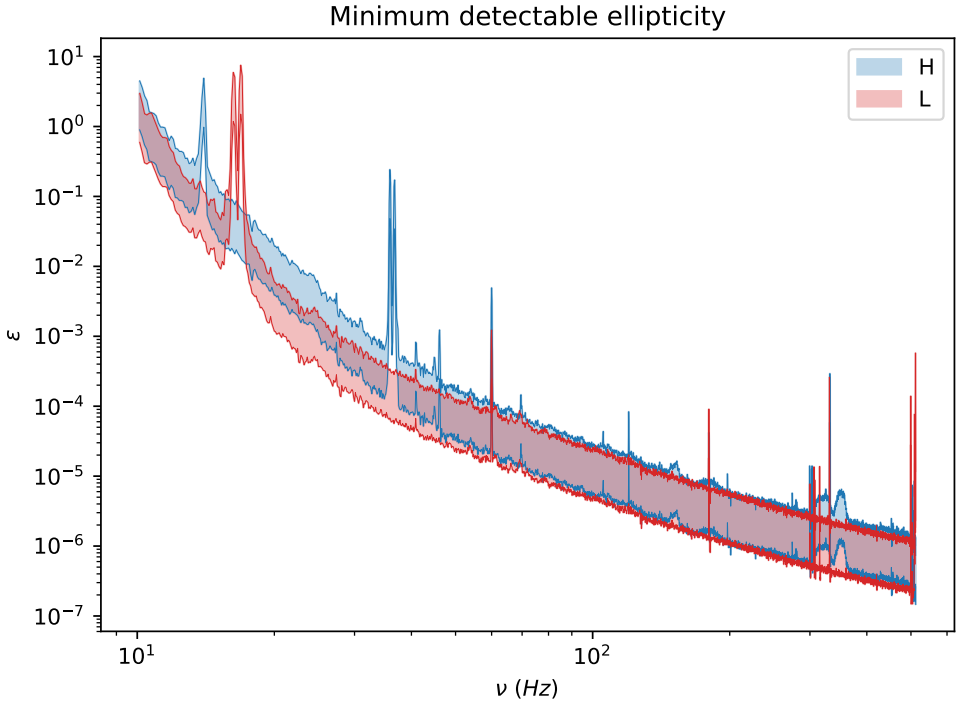

Figure 3.12: Minimum detectable ellipticity using the sensitivity of this research, using Equation 1.23. The belts are for moments of inertia in the range $[1,\ 5] \cdot 10^{38}\, kg\, m^2$, identified by Johnson-McDaniel, 2013, as allowed values using various kinds of equations of state.

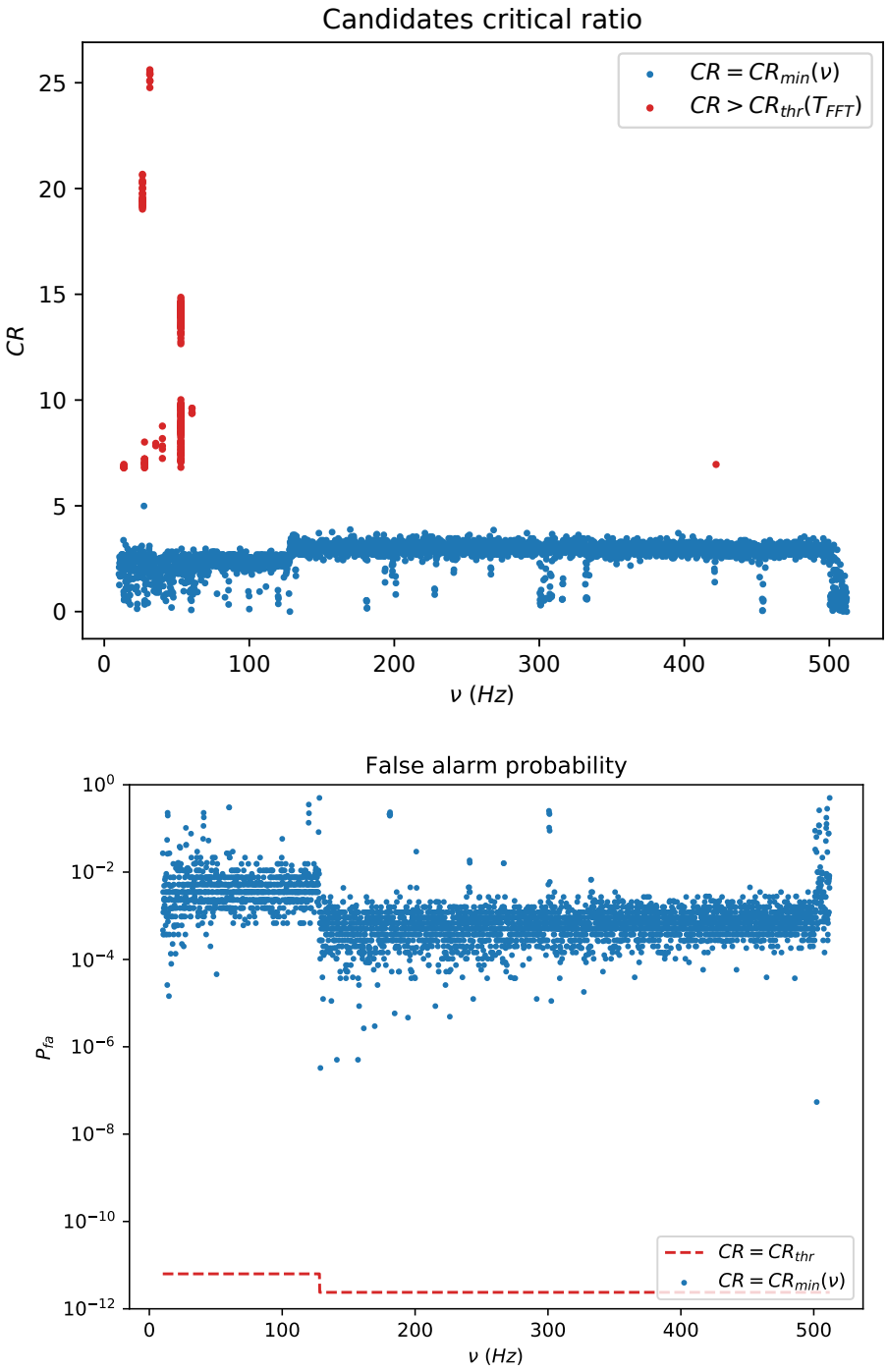

Figure 3.13: Comparison between the two threshold choices on CR. Left: plot of the candidates CR from both detectors, in function of the frequency, for the two hypothesis on the threshold. Right: the false alarm probability computed for both detectors and hypothesis.

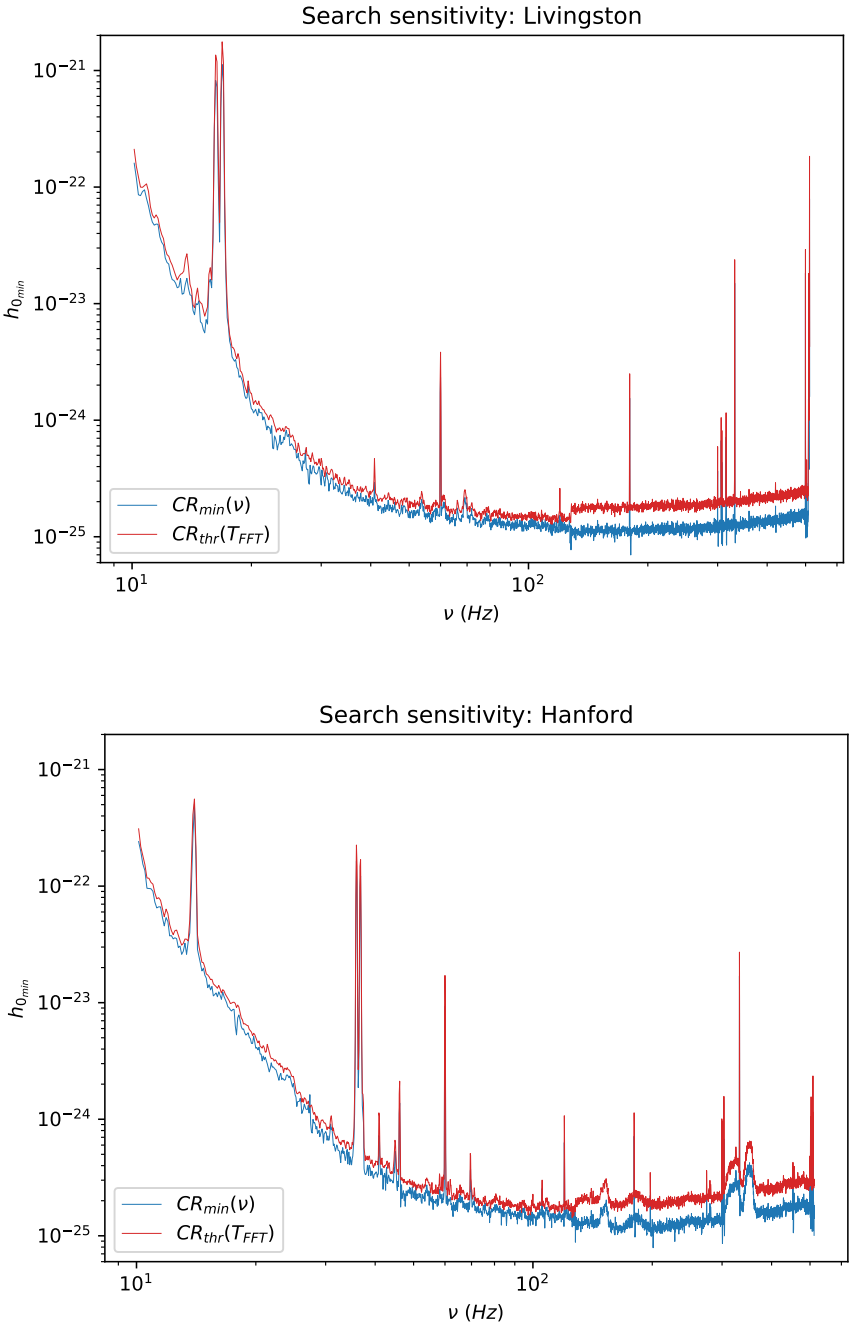

Figure 3.14: The sensitivity curves of the search computed using Equation 2.19, for LIGO Livingston (left) and Hanford (right). The blue curves are computed with the threshold defined by Hypothesis 2, the red curves with CR_{thr} from the Hypothesis 1.

Conclusions

In this Thesis I showed how can be useful in the search for continuous gravitational waves the use of GPGPU to develop analysis algorithm. Parallelizing the Frequency-Hough transform computation on GPU, and serializing over sky positions and over the input files, I obtained a speed-up of the analysis of a factor 20 in comparison to the original code. I showed also that improving the degree of parallelism, i.e. analyzing the more data possible at once, the efficiency of the analysis can be still increased.

A faster analysis may allow to expand the parameters space of the search, increasing its sensitivity. In fact, despite the blind continuous wave search have to face a very large volume of the parameters space, managing a big amount of data, the definitely biggest part of the computations to produce a candidate on a certain point of the parameter space, depends mainly on the specific point. The whole analysis pipeline can be then easily vectorized and parallelized in an efficient way on GPUs, thanks also to the high level structure of TensorFlow.

With this objective in mind a few arrangements and improvement of the code could be:

- extend the code to multi-GPU systems, since TensorFlow is very efficient working with many GPUs
- developing a scalable big-data input/output pipeline which can work out-of-memory with different devices, using appropriate modern file formats and libraries, trying to balance serialization and parallelism;
- work with bigger peakmaps, potentially covering the whole analyzed frequency/time plane;
- study the feasibility of matrices in dense format instead sparse: in this way a parallel out-of-memory input could be easier to manage;
- develop GPU codes for Doppler correction and candidate selection, which could work together with the Hough transform parallelizing over the sky position;
- since a certain degree of serialization is unavoidable, it's important to remove every iterated constant tensor or operation, since as we said every new node in the TensorFlow graph will not deleted until the session is closed, wasting a big amount of memory;

- remove any `float64` data, e.g. with data chunks covering a time range large enough to the 32bit floating point precision.

With a more complete and performing analysis code, runnable by a cluster of GPU, it would be possible to further deepen the analysis presented in this Thesis, trying to extend the sky region selected, to cover the whole sky, and the frequency range, to cover the whole sensitivity band of the detector. Moreover, more features can be added to the procedure. For instance one could:

- use adaptive Frequency-Hough transform, for which each pixel is filled no more with a boolean number but with values between 0 and 1, to take into account the effect of the detector antenna pattern to the expected response with respect to a given sky position and noise non-stationarity;

- add the second order spin-down parameter to the Hough transform in order to detect young neutron stars with high spin-down values;

- since the candidate follow-up analysis uses the Frequency-Hough transform, it could be useful to develop a code which could use the GPU code of the transform: with a faster follow-up one could select a bigger set of candidates from the Hough maps, increasing the search sensitivity.

The field of big-data computation with GPU is in fast evolution, with new hardware architectures which raise more and more the computational power available and an increasing number of frameworks which allow to develop codes with an high varieties of purposes. For this reason an increasing part of scientific analysis in the next years will use these devices and frameworks.

Developing GPGPU codes for gravitational waves search, and specifically for one of the most challenging task such the continuous waves search, is of a primary importance: one of the research field most computationally demanding, can't avoid to use the power granted by the modern graphic devices, especially if, as demonstrated by this Thesis, the analysis pipeline responds so well if converted in vectorized GPU codes. At this point, to have a cluster of GPUs could grant a great ratio between scientific result benefits and financial costs.

Bibliography

Aasi, J. et al. (2015a). "Advanced ligo". In: *Classical and quantum gravity* 32.7, p. 074001.

— (2015b). "Narrow-band search of continuous gravitational-wave signals from Crab and Vela pulsars in Virgo VSR4 data". In: *Physical Review D* 91.2, p. 022004.

— (2016). "First low frequency all-sky search for continuous gravitational wave signals". In: *Physical Review D* 93.4, p. 042007.

Abbott, B. P. et al. (2009). "LIGO: the laser interferometer gravitational-wave observatory". In: *Reports on Progress in Physics* 72.7, p. 076901.

— (2016a). "GW151226: Observation of gravitational waves from a 22-solar-mass binary black hole coalescence". In: *Physical Review Letters* 116.24, p. 241103.

— (2016b). "Observation of gravitational waves from a binary black hole merger". In: *Physical review letters* 116.6, p. 061102.

— (2017a). "All-sky search for periodic gravitational waves in the O1 LIGO data". In: *Physical Review D* 96.6, p. 062002.

— (2017b). "GW170104: Observation of a 50-Solar-Mass Binary Black Hole Coalescence at Redshift 0.2". In: *Physical Review Letters* 118.22, p. 221101.

— (2017c). "GW170814: A three-detector observation of gravitational waves from a binary black hole coalescence". In: *Physical Review Letters* 119.14, p. 141101.

— (2017d). "GW170817: observation of gravitational waves from a binary neutron star inspiral". In: *Physical Review Letters* 119.16, p. 161101.

— (2017e). "Upper limits on the stochastic gravitational-wave background from Advanced LIGO's first observing run". In: *Physical review letters* 118.12, p. 121101.

Acernese, F. et al. (2014). "Advanced Virgo: a second-generation interferometric gravitational wave detector". In: *Classical and Quantum Gravity* 32.2, p. 024001.

Antonucci, F. et al. (2008). "Detection of periodic gravitational wave sources by Hough transform in the f versus plane". In: *Classical and Quantum Gravity* 25.18, p. 184015.

Astone, P. et al. (2014). "Method for all-sky searches of continuous gravitational wave signals using the frequency-Hough transform". In: *Physical Review D* 90.4, p. 042002.

Ballard, D. H. (1981). "Generalizing the Hough transform to detect arbitrary shapes". In: *Pattern recognition* 13.2, pp. 111–122.

Bartels, R., S. Krishnamurthy, and C. Weniger (2016). "Strong support for the millisecond pulsar origin of the Galactic center GeV excess". In: *Physical review letters* 116.5, p. 051102.

Braginsky, V. B. et al. (2003). "Noise in gravitational-wave detectors and other classical-force measurements is not influenced by test-mass quantization". In: *Physical Review D* 67.8, p. 082001.

Caron, B. et al. (1997). "The Virgo interferometer". In: *Classical and Quantum Gravity* 14.6, p. 1461.

Duda, R. O. and P. E. Hart (1972). "Use of the Hough transformation to detect lines and curves in pictures". In: *Communications of the ACM* 15.1, pp. 11–15.

Einstein, A. (1916). "Näherungsweise Integration der Feldgleichungen der Gravitation". In: *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften*, pp. 688–696.

— (1918). "Über Gravitationswellen". In: *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften*, pp. 154–167.

Ferrari, V. and L. Gualtieri (2014). "General Relativity". In: *Lectures of the courses "General Relativity" and "GW, NS and BH"*.

Forward, R. L. (1978). "Wideband laser-interferometer graviational-radiation experiment". In: *Physical Review D* 17.2, p. 379.

Frasca, S., S. D'Antonio, et al. (2017). *SNAG*. URL: grwavsf.roma1.infn.it/snag.

Hough, P. V. C. (1962). *Method and means for recognizing complex patterns*. Tech. rep.

Izabela, K. L. et al. (2017). "Globally coherent short duration magnetic field transients and their effect on ground based gravitational-wave detectors". In: *Classical and Quantum Gravity* 34.7, p. 074002.

Johnson-McDaniel, N. K. (2013). "Gravitational wave constraints on the shape of neutron stars". In: *Physical Review D* 88.4, p. 044016.

Johnson-McDaniel, N. K. and B. J. Owen (2013). "Maximum elastic deformations of relativistic stars". In: *Physical Review D* 88.4, p. 044004.

Khronos, Group (2009). *OpenCL*. URL: www.khronos.org/opencl.

Maggiore, M. (2000). "Gravitational wave experiments and early universe cosmology". In: *Physics Reports* 331, pp. 283–367.

Muciaccia, F. (2017). "Using Deep Convolutional Neural Networks to build a low-latency classifier to trigger the detection of long gravitational-wave transients". Master degree thesis. Sapienza Università di Roma.

NVIDIA, Corporation (2007). *CUDA*. URL: developer.nvidia.com/cuda-toolkit.

Pirani, F. A. E. (1956). "On the Physical significance of the Riemann tensor". In: *Acta Phys. Polon.* 15, pp. 389–405.

Pitkin, M. et al. (2011). "Gravitational wave detection by interferometry (ground and space)". In: *Living Reviews in Relativity* 14.1, p. 5.

Powell, J., M. Szczepanczyk, and I. S. Heng (2017). "Inferring the core-collapse supernova explosion mechanism with three-dimensional gravitational-wave simulations". In: *arXiv preprint arXiv:1709.00955*.

Rajwade, K. M., D. R. Lorimer, and L. D. Anderson (2017). "Detecting pulsars in the Galactic Centre". In: *Monthly Notices of the Royal Astronomical Society* 471.1, pp. 730–739.

Ricci, F. (2017). "Gravitazione Sperimentale". In: *Lectures of the course "Experimental Gravitation"*.

Romano, J. D. and N. J. Cornish (2017). "Detection methods for stochastic gravitational-wave backgrounds: a unified treatment". In: *Living Reviews in Relativity* 20.1, p. 2.

Saulson, P. R. (2017). *Fundamentals of interferometric gravitational wave detectors*. World scientific.

Scipy, developers (2017). *Scipy*. URL: www.scipy.org.

Somiya, K. (2012). "Detector configuration of KAGRA–the Japanese cryogenic gravitational-wave detector". In: *Classical and Quantum Gravity* 29.12, p. 124007.

TensorFlow, The Authors (2017). *TensorFlow*. URL: www.tensorflow.org.

Weiss, R. (1972). *Electromagnetically coupled broadband gravitational antenna*.

Wolf, E. (1998). *Progress in Optics*. Progress in Optics v. 38. Elsevier Science, p. 87.