National Technical University of Athens

M.Sc. in Applied Mathematical Sciences

M.Sc. Thesis

Analytic Fourier-Domain Models for the Post-Merger Gravitational Wave Signal in Binary Neutron Star Mergers

Author
Anna Marinitsi

Supervisor
Prof. Nikolaos Stergioulas

October 30, 2023

Abstract

Gravitational-wave astronomy is a rapidly emerging new field with great potential to expand our knowledge of the Universe. An anticipated new source of gravitational waves is the post-merger phase of binary neutron star mergers. If detected, it will allow us to constrain the equation of state of high-density matter at finite temperatures. Successful detection requires highly accurate analytical templates that describe the gravitational wave emission of the post-merger remnants. To date, only simple templates exist in the frequency domain. In this thesis, we compute analytic templates in the frequency domain that agree exactly with their corresponding time-domain representations. The most advanced model includes a time-evolving dominant post-merger frequency. Using nonlinear least-squares fits to synthetic data, we show that our new frequency-domain models have advantages over previous, simpler expressions used in the literature. Our models are thus useful for detection and parameter estimation of the post-merger phase in binary neutron star mergers.

Acknowledgements

I am immensely grateful to the people who supported me throughout the entire journey of preparing this thesis. I would like to thank my supervisor, Prof. Nikolaos Stergioulas, for his trust and support, my family and the people close to me that have been a constant source of encouragement and motivation. Finally, I owe a special debt of gratitude to Evi and Labis for their unwavering support and inspiration. You are the best!

Contents

Chapter 1

Introduction

In this Chapter, we will summarize the main results of the General Theory or Relativity, that lead to the description of gravitational waves, and we will give a summary of relevant background knowledge on neutron stars and binary neutron star mergers.

Differential Geometry

For this and the next Section, we will follow mainly the books by B. F. Schutz [21] and R. d' Inverno [13].

Manifolds

The Theory of General Relativity is based on a four-dimensional manifold. In an $n-$ dimensional manifold, each point of the manifold $\mathcal{M}$ is described by n real numbers, $x^1, x^2, ..., x^n$. This allows for a one-to-one correspondence between the manifold and an $n-$ dimensional Euclidean space. There are multiple coordinate systems that cover different regions of $\mathcal{M}$, and when two of these coordinate systems overlap, there is a set of equations that can be used to convert the coordinates of a point from one system to the other. For example, if the coordinates x^μ cover a region U and the coordinates x'^μ cover a region U' and there is a common region of U and U', then the coordinates of a point P in the common region can be expressed as:

$$x^{\mu'} = x^{\mu'}\left(x^1, x^2, ..., x^n\right), \text{ for } \mu' = 1, ..., n, \tag{1.1}$$

and the inverse equations are as follows:

$$x^{\mu} = x^{\mu}\left(x^{1'}, x^{2'}, ..., x^{n'}\right), \text{ for } \mu = 1, ..., n. \tag{1.2}$$

If the equations are differentiable, meaning that the partial derivatives

$$\frac{\partial x^{\mu'}}{\partial x^{\nu}} \text{ and } \frac{\partial x^{\nu}}{\partial x^{\mu'}} \tag{1.3}$$

exist, this indicates that the manifold M is a differentiable manifold so the Jacobian matrix can be calculated. The $n \times n$ matrix $A^{\mu'}_{\nu} = \partial x^{\mu'}/\partial x^{\nu}$ is the Jacobian matrix of the transformation (1.1).

Tensors

A curve that passes through a point A in an $n-$dimensional manifold is determined by n functions and can be expressed as a function of a scalar parameter λ:

$$x^{\mu} = z^{\mu}(\lambda). \tag{1.4}$$

At a close point A', the value of the parameter λ will be $\lambda + d\lambda$. The tangent vector v^{μ} at the point A is given by equation:

$$v^{\mu} = \frac{dx^{\mu}}{d\lambda}. \tag{1.5}$$

This tangent vector v^{μ} is called a contravariant vector.

The transformation of a coordinate system x^{μ} to another $\tilde{x}^{\mu}$, is defined by n equations:

$$\tilde{x}^{\mu} = f^{\mu}(x^{\nu}), \;\; \mu, \nu = 0, 1, ..., n-1. \tag{1.6}$$

The inverse transformation is given by:

$$x^{\mu} = g^{\mu}(\tilde{x}^{\nu}), \;\; \mu, \nu = 0, 1, ..., n-1, \tag{1.7}$$

with $g(x^{\nu}) = f^{-1}(x^{\nu})$. Transformations between two systems are valid only if the Jacobians of the transformations are different than zero,

$$\det\left|\frac{\partial \tilde{x}^{\mu}}{\partial x^{\nu}}\right| \neq 0 \text{ and } \det\left|\frac{\partial x^{\mu}}{\partial \tilde{x}^{\nu}}\right| \neq 0. \tag{1.8}$$

An infinitely small vector dx^{ν} of a system x^{μ} will be transformed into a system $\tilde{x}^{\mu}$ as follows:

$$d\tilde{x}^{\mu} = \sum_{\nu} \frac{\partial \tilde{x}^{\mu}}{\partial x^{\nu}} dx^{\nu} = \sum_{\nu} \frac{\partial f^{\mu}}{\partial x^{\nu}} dx^{\nu} \text{ for } \nu = 0, 1, ..., n-1. \tag{1.9}$$

The components of the contravariant vector v^{μ} in a different coordinate system $\tilde{x}^{\mu}$ can be determined by transformations

$$\tilde{v}^{\mu} = \sum_{\nu} \frac{\partial \tilde{x}^{\mu}}{\partial x^{\nu}} v^{\nu}. \tag{1.10}$$

This is because the scalar parameter λ is invariant under a coordinate transformation.

If $\phi(x)$ is a scalar function in a coordinate system x^{μ}, its partial derivatives $n\ \partial\phi/\partial x^{\mu}$ can be transformed into a different coordinate system $\tilde{x}^{\mu}$ as follows:

$$\frac{\partial \phi}{\partial \tilde{x}^{\mu}} = \sum_{\nu} \frac{\partial \phi}{\partial x^{\nu}} \frac{\partial x^{\nu}}{\partial \tilde{x}^{\mu}}. \tag{1.11}$$

The set of n quantities $b_{\mu} = \{b_0, b_1, ..., b_{n-1}\}$ is transformed in the same way as the partial derivatives of a scalar function and is called a covariant vector. These transformations, based on Ea. (1.11), are expressed as follows:

$$b_{\mu} = \sum_{\nu} \frac{\partial x^{\nu}}{\partial \tilde{x}^{\mu}} \tilde{b}_{\nu}. \tag{1.12}$$

Scalars and vectors (covariant and contravariant) are examples of a more general type of geometric quantity known as tensors. The scalars are zero-rank tensors, and the vectors are first-rank tensors. Here, we define the second-rank contravariant tensor in an $n-$ dimensional space, $T^{\mu\nu}$, which has n^2 components that transform according to the equation:

$$\tilde{T}^{\alpha\beta} = \frac{\partial\tilde{x}^\alpha}{\partial x^\mu}\frac{\partial\tilde{x}^\beta}{\partial x^\nu}T^{\mu\nu}. \tag{1.13}$$

We can define the second-rank covariant tensor $T_{\mu\nu}$ and the mixed tensor $T^\mu{}_\nu$ in a similar way, as follows:

$$\tilde{T}^\alpha{}_\beta = \frac{\partial\tilde{x}^\alpha}{\partial x^\mu}\frac{\partial x^\nu}{\partial\tilde{x}^\beta}T^\mu{}_\nu \text{ and } \tilde{T}_{\alpha\beta} = \frac{\partial x^\mu}{\partial\tilde{x}^\alpha}\frac{\partial x^\nu}{\partial\tilde{x}^\beta}T_{\mu\nu}. \tag{1.14}$$

This process can be repeated indefinitely, allowing us to construct third-rank, fourth-rank, and higher-order tensors.

$$\tilde{T}_{\alpha\beta\ldots}{}^{\gamma\delta\ldots} = \left(\frac{\partial x^\epsilon}{\partial\tilde{x}^\alpha}\right)\left(\frac{\partial x^\zeta}{\partial\tilde{x}^\beta}\right)\cdots\left(\frac{\partial\tilde{x}^\gamma}{\partial x^\eta}\right)\left(\frac{\partial\tilde{x}^\delta}{\partial x^\theta}\right)\cdots T_{\epsilon\zeta\ldots}{}^{\eta\theta\ldots}. \tag{1.15}$$

Covariant Derivative

If we assign a vector to each point in a space, then we have created a vector field. Similarly, if we assign an n -rank tensor to each point, we have created a tensor field. The partial derivative of a scalar field $\phi = \phi(x^\alpha)$ (zero rank tensor) produces a rank one tensor field Eq. (1.11) as shown in the equation:

$$\frac{\partial\phi}{\partial x^\alpha} = \frac{\partial\tilde{x}^\lambda}{\partial x^\alpha}\frac{\partial\phi}{\partial\tilde{x}^\lambda}. \tag{1.16}$$

The partial derivative of a contravariant vector field, $A^\lambda(x^\mu)$, is expressed as:

$$\frac{\partial A^\lambda}{\partial x^\kappa} \equiv \partial_\kappa A^\lambda \equiv A^\lambda{}_{,\kappa}. \tag{1.17}$$

When this is transformed into a new coordinate system $\tilde{x}^\kappa$, the following equation is obtained:

$$\begin{aligned} A^\mu{}_{,\kappa} &= \frac{\partial}{\partial x^\alpha}\left(\frac{\partial x^\mu}{\partial\tilde{x}^\nu\tilde{A}^\nu}\right) = \frac{\partial\tilde{x}^\rho}{\partial x^\alpha}\frac{\partial}{\partial\tilde{x}^\rho}\left(\frac{\partial x^\mu}{\partial\tilde{x}^\nu}\tilde{A}^\nu\right) \\ &= \frac{\partial^2 x^\mu}{\partial\tilde{x}^\nu\partial\tilde{x}^\rho}\frac{\partial\tilde{x}^\rho}{\partial x^\alpha}\tilde{A}^\nu + \frac{\partial x^\mu}{\partial\tilde{x}^\nu}\frac{\partial\tilde{x}^\rho}{\partial x^\alpha}\frac{\partial\tilde{A}^\nu}{\partial\tilde{x}^\rho}. \end{aligned} \tag{1.18}$$

This results in a second-order tensor only in the case of a linear transformation (the first term of the expression is equal to zero). To achieve this, we construct the covariant derivative in such a way that the troublesome terms are eliminated.

$$\nabla_\alpha A^\mu = A^\mu{}_{;\alpha} = A^\mu{}_{,\alpha} + \Gamma^\mu_{\alpha\lambda}A^\lambda = \partial_\alpha A^\mu + \Gamma^\mu_{\alpha\lambda}A^\lambda. \tag{1.19}$$

The quantity $\Gamma^\mu_{\alpha\lambda}$ is known as the connection and is expressed as:

$$\tilde{\Gamma}^\lambda_{\rho\nu} = \frac{\partial^2 x^\mu}{\partial\tilde{x}^\nu\partial\tilde{x}^\rho}\frac{\partial\tilde{x}^\lambda}{\partial x^\mu} + \frac{\partial x^\kappa}{\partial\tilde{x}^\rho}\frac{\partial x^\sigma}{\partial\tilde{x}^\nu}\frac{\partial\tilde{x}^\lambda}{\partial x^\mu}\Gamma^\mu_{\kappa\sigma}. \tag{1.20}$$

This connection is not a tensor itself, but rather a quantity that can be used to convert a non-tensor quantity into a second-order tensor.

Geodesic Curves

A geodesic is a curve in which the velocity of a particle is tangent to the curve and is determined by a parallel transport. The curve $x^a = x^a(s)$, with s being a physical parameter, is a geodesic if the tangent vector meets the requirement that $D(dx^a/ds)/Ds = 0$. This can be expressed by using the definition of the absolute derivative ($\frac{Du^a}{D\lambda} = \frac{du^a}{d\lambda} + \Gamma^a_{cb}\frac{dx^b}{d\lambda}u^c$), which results in equation $\frac{d^2x^a}{ds^2} + \Gamma^a_{cb}\frac{dx^b}{ds}\frac{dx^c}{ds} = 0$.

$$\frac{d^2x^a}{ds^2} + \Gamma^a_{bc}\frac{dx^b}{ds}\frac{dx^c}{ds} = 0. \tag{1.21}$$

This equation is the differential equation of the geodesic curve, which is a generalization of the straight line in spaces with $\Gamma^a_{bc} \neq 0$.

General Relativity

Riemannian Spacetimes

The metric tensor is a fundamental mathematical construct that establishes rules to calculate the distances between two points within a specified spatial context. It plays a central role in precisely characterizing the geometry of the defined space. In a 3D Euclidean space, in Cartesian coordinates, the shortest distance between two nearby points (x, y, z) and $(x + dx, y + dy, z + dz)$ is given by:

$$ds^2 = dx^2 + dy^2 + dz^2. \tag{1.22}$$

In 4-dimensional Minkowski spacetime, used in special relativity, nearby distances are defined by the Minkowski metric, which accounts for both space and time. Given two points with coordinates (t, x, y, z) and $(t + dt, x + dx, y + dy, z + dz)$ the distance (space-time separation) between them is given by:

$$ds^2 = -c^2dt^2 + dx^2 + dy^2 + dz^2, \tag{1.23}$$

where c represents the speed of light. In general, for an $n-$dimensional spacetime, the distance between two close events is:

$$ds^2 = g_{ab}dx^a dx^b = \sum_{a,b=1}^{n} g_{ab}dx^a dx^b, \tag{1.24}$$

where g_{ab} is the metric tensor.

The metric tensor matrix of the 3-dimensional Euclidean space is:

$$g_{\alpha\beta} = \epsilon_{ab} \equiv \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \text{diag}|[1, 1, 1] \tag{1.25}$$

where $\alpha, \beta = 1, 2, 3$ The Minkowskian metric tensor is:

$$g_{\alpha\beta} = \eta_{ab} \equiv \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \text{diag}[-1, 1, 1, 1], \tag{1.26}$$

where $a, b = 0, 1, 2, 3$ In Riemannian geometry, the metric tensor is symmetric, so the form of the metric is a $n \times n$ matrix:

$$g_{ab} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n1} & g_{n2} & \cdots & g_{nn} \end{bmatrix}. \tag{1.27}$$

Matrices are second-order tensors. If $\det[g_{ab}] \neq 0$, then the g_{ab} matrix is invertible. Since g_{ab} is symmetric, the inverse matrix, if it exists, is also a symmetric second-order tensor.

$$g_{ac}g^{cb} = \delta^b_a, \tag{1.28}$$

where δ^b_a is the Kronecker δ.

In the cases where the metric is diagonal, with components g_{ii} for $i = 1, 2, ..., n$, it follows that $g^{ii} = 1/g_{ii}$. This diagonal matrix allows for the conversion of contravariant and covariant indices using the metric tensor. For example, when working with a contravariant vector u^a, one can transform it into a covariant vector u_a by contracting it with the metric tensor: $u_a = g_{ab}u^b$. The scalar quantity u^2, defined as $u^2 \equiv g_{ab}u^a u^b = g^{ab}u_a u_b = u_a u^a$, serves as a measure of the vector's length or magnitude squared. In the context of relativity, vectors are categorized based on the value of u^2: A vector is described as timelike if $u_a u^a < 0$, spacelike if $u_a u^a > 0$ and lightlike (or null) if $u_a u^a = 0$.

Christoffel Symbols

The covariant derivative of a tensor gives rise to the definition of the connection, as seen in Eq. (1.21). Christoffel symbols, which originate from the metric tensor, constitute a set of numerical coefficients that define the connection in a manifold. They play a fundamental role in the context of differentiating and parallel transporting vectors along curves within the manifold. In Riemann spaces, the Christoffel symbols are given by:

$$\Gamma^a_{bc} = \frac{1}{2}g^{ad}\left(\partial_c g_{db} + \partial_b g_{dc} - \partial_d g_{bc}\right). \tag{1.29}$$

Due to the symmetry of the metric tensor, they satisfy:

$$\Gamma^a_{bc} = \Gamma^a_{cb}. \tag{1.30}$$

This implies that in Riemannian spaces, torsion is absent.

Geodesic curves in GR

Within the four-dimensional spacetime of General Relativity, the preferred fundamental measure along a curve is the proper length. However, as an alternative to the proper length, we can introduce another fundamental parameter, denoted as proper time (τ).

$$d\tau^2 = -g_{ab}dx^a dx^b. \tag{1.31}$$

If a curve in spacetime, described by the parametric equations $x^a = x^a(\tau)$, is a geodesic, and the tangent vector ($u_a = dx^a/d\tau$) satisfies the condition $u_a u^a = -1$, then the curve follows a path of extremal length with respect to proper time, so:

$$\frac{d^2x^a}{d\tau^2} + \Gamma^a_{bc}\frac{dx^b}{d\tau}\frac{dx^c}{d\tau} = 0. \tag{1.32}$$

Riemann, Ricci, and Einstein tensors

Tensors are essential for describing the curvature of spacetime and are the basis for formulating Einstein's field equations, which govern the gravitational dynamics of matter and energy. The Riemann (or curvature) tensor is:

$$R^a{}_{bcd} = -\partial_d\Gamma^a_{bc} + \partial_c\Gamma^a_{bd} - \Gamma^e_{bc}\Gamma^a_{ed} + \Gamma^e_{bd}\Gamma^a_{ec}. \tag{1.33}$$

The Ricci tensor is defined by contracting the Riemann tensor and by the further contraction over the Ricci tensor one obtains the Ricci scalar, which provides a measure of the "average" curvature of Riemannian space. So, the Ricci tensor and the Ricci scalar are, respectively:

$$R_{ab} \equiv R^c{}_{acb} \text{ and } R \equiv R_a{}^a = g_{ab}R^{ab}. \tag{1.34}$$

In General Relativity, the Bianchi identity is:

$$\begin{aligned}
\nabla_a R^a{}_{bcd} + \nabla_b R^a{}_{cad} + \nabla_c R^a{}_{abd} &= 0, \\
g^{cd}(-\nabla^a R_{bcad} + \nabla_b R_{cd} - \nabla_c R_{bd}) &= 0, \\
-\nabla^a R_{ba} + \nabla_b R - \nabla^d R_{bd} &= 0, \\
-2\nabla^a \left(R_{ab} - \frac{1}{2} g_{ab} R \right) &= 0.
\end{aligned} \tag{1.35}$$

For the previous steps, contraction and Riemann tensor symmetries were used. For example: $R^a{}_{abd} = -R^a{}_{bad} = R_{bd}$ and $-g^{cd}\nabla^a R_{bcad} = -\nabla^a g^{cd} R_{bcad} = -\nabla^a R_{ba}$. The expression in parentheses represents the Einstein tensor.

$$G_{ab} \equiv R_{ab} - \frac{1}{2} g_{ab} R. \tag{1.36}$$

Field Equations

The equation that extends the Poisson equation for gravity ($\nabla^2\Phi = 4\pi G\rho$) to the framework of General Relativity is expressed as follows:

$$G_{ab} = \frac{8\pi G}{c^4} T_{ab}. \tag{1.37}$$

The left side of the equation is relevant to the geometry of the space and the right side includes any form of energy density in the energy momentum tensor T^{ab}. The above equation can be justified in the following way: One can attempt to generalize Poisson's equation as

$$R_{ab} + \lambda g_{ab} R = \kappa T^{ab}, \tag{1.38}$$

where λ and κ are constants. To ensure the conservation of energy and momentum in the context of general relativity, one can impose the condition $\nabla_b T^{ab} = 0$, which is a generalization of the corresponding law in special relativity. Then, also

$$\nabla_b (R^{ab} + \lambda g^{ab} R = 0). \tag{1.39}$$

But, from the Bianchi identity Eq. (1.35):

$$\nabla_b (R^{ab} + \lambda g^{ab} R) = 0. \tag{1.40}$$

For this to be true, λ should be equal to $\lambda = -\frac{1}{2}$. When gravitational effects are weak and velocities are much lower than the speed of light (the Newtonian limit), the field equations should reduce to the familiar Poisson equation. This happens when the parameter κ tahashe value of $\kappa = \frac{8\pi G}{c^4}$. As a result, Eq. (1.37) is obtained.

The field equations can be further generalized, by taking advantage of the fact that the metric tensor is invariant under covariant transformations ($\nabla^c g_{ab} = 0$) and introducing an additional term that is proportional to the metric tensor:

$$R_{ab} - \frac{1}{2} g_{ab} R + \Lambda g_{ab} = \frac{8\pi G}{c^4} T_{ab} , \tag{1.41}$$

where Λ is called the cosmological constant. If that term is moved to the right side of the equation:

$$R_{ab} - \frac{1}{2} g_{ab} R = 8\pi \left(T_{ab} - \frac{\Lambda}{8\pi} g_{ab} \right), \tag{1.42}$$

where we set $G = c = 1$. The term $-\frac{\Lambda}{8\pi}$ has been interpreted as the energy density of vacuum, although the value calculated from theory does not agree with Λ as obtained from cosmological observations.

Gravitational Waves

In this Section we will be adhering to the same format as Maggiore's book on gravitational waves [18] and Flanagan's introduction to them [15].

General Relativity is a highly nonlinear theory, thus we in general consider wave solutions that are nonlinear. However, it is important to note that linear approximations can be made in specific situations, where the gravitational field is weak and the spacetime curvature is only slightly perturbed from flat (Minkowski) spacetime. These approximations occur by applying perturbation theory to the metric tensor and neglecting second-order terms. This simplification leads to a linear wave equation that accurately describes the propagation of gravitational waves, especially when considering their behavior far from an astrophysical source. Near a source, the linear approximation may not be precise and higher-order corrections or a full nonlinear solution may be required.

The Linearization of General Relativity

Linearized gravity is essentially a perturbation theory applied to Minkowski spacetime. This means that we start with the assumption of a flat Minkowski metric as the background spacetime, and then we introduce small perturbations to describe deviations from this flat spacetime metric. So, the metric becomes

$$g_{ab} = \eta_{ab} + h_{ab} \; , \tag{1.43}$$

where $||h_{\mu\nu}|| << 1$ and $\eta_{\mu\nu} = \mathrm{diag}[-1,1,1,1]$ the Minkowskian metric tenor. For $||h_{\mu\nu}|| << 1$ to occur, the gravitational field must be weak. In linearized gravity, the perturbation is assumed to be small, so only terms that are linear in $\eta_{\mu\nu}$ are kept, while higher-order terms are ignored. We can utilize the metric tensor $\eta_{\mu\nu}$ to raise and lower the indices. Spatial ones can be expressed in either the "up" or "down" position without altering the quantity's value, i.e., $f^x = f_x$. However, when it comes to time indices, the sign changes when raised or lowered: $f^t = -f_t$.

The linearized Christoffel symbols are

$$\delta\Gamma^{\mu}_{ab} = \frac{1}{2}\eta^{\mu\nu}\left(\partial_b h_{\nu a} + \partial_a h_{b\nu} - \partial_\nu h_{ab}\right) \Rightarrow$$

$$\delta\Gamma^{\mu}_{ab} = \frac{1}{2}\left(\partial_b h^{\mu}{}_{a} + \partial_a h^{\mu}{}_{b} - \partial^{\mu} h_{ab}\right) \Rightarrow$$

$$\delta\Gamma^{\mu}_{ab} = \frac{1}{2}\left(h^{\mu}{}_{a,b} + h^{\mu}{}_{b,a} - h_{ab}{}^{,\mu}\right). \tag{1.44}$$

The linearized Riemann tensor (1.33) becomes:

$$\delta R^{a}{}_{bcd} = \frac{1}{2}\partial_c\left(\partial_d h^{a}{}_{b} + \partial_d h^{a}{}_{d} - \partial^{a} h_{bd}\right) - \frac{1}{2}\partial_d\left(\partial_c h^{a}{}_{b} + \partial_b h^{a}{}_{c} - \partial^{a} h_{bc}\right), \Rightarrow$$

$$\delta R^{a}{}_{bcd} = \frac{1}{2}\left(\partial_c\partial_b h^{a}{}_{d} - \partial_c\partial^{a} h_{bd} - \partial_d\partial_b h^{a}{}_{c} + \partial_d\partial^{a} h_{bc}\right) \tag{1.45}$$

and

$$\delta R_{a\mu\nu b} = \frac{1}{2}\left(\partial_b\partial_\mu h_{a\nu} + \partial_\nu\partial_a h_{\mu b} - \partial_b\partial_a h_{\mu\nu} - \partial_\nu\partial_\mu h_{ab}\right) \Rightarrow$$

$$\delta R_{a\mu\nu b} = \frac{1}{2}\left(h_{a\nu,b\mu} + h_{\mu b,\nu a} - h_{\mu\nu,ba} - h_{ab,\nu\mu}\right). \tag{1.46}$$

The linearized Ricci tensor (1.34) is created by contracting the Riemann tensor:

$$\delta R_{\mu\nu} = \eta^{ab} R_{a\mu b\nu} = \frac{1}{2}\eta^{ab}\left(h_{a\nu,b\mu} + h_{\mu b,\nu a} - h_{\mu\nu,ba} - h_{ab,\nu\mu}\right) =$$

$$\frac{1}{2}\left(h^{b}{}_{\nu,\mu b} - h_{,\mu\nu} + h_{\mu}{}^{a}{}_{,a\nu} - h_{\mu\nu,a}{}^{a}\right) \Rightarrow$$

$$\delta R_{\mu\nu} = \frac{1}{2}\left(h^{a}{}_{\nu,\mu a} - h_{,\mu\nu} + h_{\mu}{}^{a}{}_{,a\nu} - h_{\mu\nu,a}{}^{a}\right). \tag{1.47}$$

By contracting again, the linearized Ricci scalar (1.34) is formed:

$$\delta R = \eta^{\mu\nu} R_{\mu\nu} = \frac{1}{2}\eta^{\mu\nu}\left(h^{a}{}_{\nu,\mu a} - h_{,\mu\nu} + h_{\mu}{}^{a}{}_{,a\nu} - h_{\mu\nu,a}{}^{a}\right) \Rightarrow$$

$$\delta R = \frac{1}{2}\left(h^{a\mu}{}_{,a\mu} - \eta^{\mu\nu} h_{,\mu\nu} + h^{\nu a}{}_{,a\nu} - h_{,a}{}^{a}\right) \Rightarrow$$

$$\delta R = h^{a\mu}{}_{,a\mu} - h_{,a}{}^{a}. \tag{1.48}$$

Then, from the Ricci tensor Eq. (1.47) and Ricci scalar Eq. (1.48) in linear gravity, the linearized Einstein tensor is formed:

$$\delta G_{\mu\nu} = \frac{1}{2}\left(h^{a}{}_{\nu,\mu a} - h_{,\mu\nu} + h_{\mu}{}^{a}{}_{,a\nu} - h_{\mu\nu,a}{}^{a}\right) - \frac{1}{2}\eta_{\mu\nu}\left(h^{a\mu}{}_{,a\mu} - h_{,a}{}^{a}\right). \tag{1.49}$$

Finally, the linearized field equations are:

$$h^{a}{}_{\nu,\mu a} - h_{,\mu\nu} + h_{\mu}{}^{a}{}_{,a\nu} - h_{\mu\nu,a}{}^{a} - \eta_{\mu\nu}\left(h^{a\mu}{}_{,a\mu} - h_{,a}{}^{a}\right) = 16\pi\delta T_{\mu\nu}, \tag{1.50}$$

where $G = c = 1$.

The linearized field equations describe the propagation of gravitational waves. This is more apparent when working with a modified metric tensor $\overline{h}_{ab}$ of the metric perturbation tensor h_{ab} introduced earlier. This tensor is known as the trace-reversed perturbation and is defined as: $\overline{h}_{ab} = h_{ab} - \frac{1}{2}\eta_{ab} h \Rightarrow h_{ab} = \overline{h}_{ab} + \frac{1}{2}\eta_{ab} h$, where h is the trace, for which $\overline{h} = -h$. By replacing h_{ab} with $\overline{h}_{ab} + \frac{1}{2}\eta_{ab} h$ one obtains:

$$\overline{h}_{\mu a}{}^{,a}{}_{\nu} + \frac{1}{2}\eta_{\mu a}{}^{,a}{}_{\nu} + \overline{h}_{\nu a}{}^{,a}{}_{\mu} + \frac{1}{2}\eta_{\nu a} h^{,a}{}_{\mu} - \overline{h}_{\mu\nu}{}^{,a}{}_{a} - \frac{1}{2}\eta_{\mu\nu} h^{,a}{}_{a} - h_{,\mu\nu} - \eta_{\mu\nu}\left(\overline{h}_{a\mu}{}^{,a\mu} - \frac{1}{2}\eta_{a\mu} h^{,a\mu} + h_{,a}{}^{a}\right) = 16\pi\delta T_{\mu\nu}\,. \tag{1.51}$$

The Laplace operator in the Minkowski space or else known as the d'Alembertian operator is $\Box = \partial_a \partial^a = \nabla^2 - \partial_t^2$. By using this operator, the previous expression can be written as:

$$-\eta_{\mu\nu}\overline{h}_{a\mu,}{}^{a\mu} + \overline{h}_{\mu a,}{}^{a}{}_{\nu} - \Box\overline{h}_{\mu\nu} + \overline{h}_{\nu a,}{}^{a}{}_{\mu} = 16\pi\delta T_{\mu\nu}\,. \tag{1.52}$$

To simplify this expression, one can select a suitable coordinate system or gauge. Suppose that we perform a general infinitesimal coordinate transformation of the form $x^{a'} = x^{a} + \xi^{a}$, where $\xi^{a}(x^{b})$ an infinitesimal displacement. The transformed metric is:

$$g'_{\mu\nu}(x') = \frac{\partial x^{a}}{\partial x'^{\mu}}\frac{\partial x^{b}}{\partial x'^{\nu}} g_{ab}(x). \tag{1.53}$$

and one obtains

$$h'_{\mu\nu} = h_{\mu\nu} - \tilde{g}_{\nu\rho}\xi^{\rho}{}_{,\mu} - \tilde{g}_{\rho\mu}\xi^{\rho}{}_{,\nu} - \xi^{\rho}\tilde{g}_{\mu\nu,\rho} + O(\xi^{2}), \tag{1.54}$$

The tilde symbol is used to denote the background metric. Also:

$$\begin{aligned}\bar{g}_{\rho\nu}\bar{\nabla}_\mu\xi^\rho + \bar{g}_{\rho\mu}\bar{\nabla}_\nu\xi^\rho &= \bar{g}_{\rho\nu}\xi^\rho_{,\mu} + \bar{g}_{\rho\mu}\xi^\rho_{,\nu} - \frac{1}{2}(\bar{g}_{\nu\mu,\kappa} + \bar{g}_{\nu\kappa,\mu} - \bar{g}_{\mu\kappa,\nu})\xi^\kappa - \frac{1}{2}(\bar{g}_{\mu\nu,\kappa} + \bar{g}_{\mu\kappa,\nu} - \bar{g}_{\nu\kappa,\mu})\xi^\kappa \\ &= \bar{g}_{\rho\nu}\xi^\rho{}_{,\mu} + \bar{g}_{\rho\mu}\xi^\rho{}_{,\nu} - \bar{g}_{\mu\nu,\kappa}\xi^\kappa. \qquad (1.55)\end{aligned}$$

So eventually,

$$h'_{\mu\nu} = h_{\mu\nu} - 2\bar{\nabla}_{(\mu}\xi_{\nu)}. \qquad (1.56)$$

In a flat background spacetime, the metric is

$$h'_{\mu\nu} = h_{\mu\nu} - 2\partial_{(\mu}\xi_{\nu)}, \qquad (1.57)$$

and the trace-reversed metric becomes

$$\bar{h}'_{\mu\nu} = h'_{\mu\nu} - \frac{1}{2}\eta_{\mu\nu}h' = \bar{h}_{\mu\nu} - 2\partial_{(\nu}\xi_{\mu)} + \eta_{\mu\nu}\partial^a\xi_a. \qquad (1.58)$$

A gauge that is commonly used, is the Lorentz gauge $\bar{h}_{ab}{}^{,a} = 0$. Then

$$\bar{h}'_{ab}{}^{,a} = \bar{h}_{ab}{}^{,a} - \xi_{a,b}{}^{,a} - \Box\xi_b + \xi_{a,b}{}^{,a} = \bar{h}_{ab}{}^{,a} - \Box\xi_b. \qquad (1.59)$$

In this case the infinitesimal coordinate transformation must be

$$\Box\xi_b = \bar{h}_{ab}{}^{,a}. \qquad (1.60)$$

But, the initial metric perturbation fulfills the Lorentz gauge, which means that

$$\Box\xi_b = 0. \qquad (1.61)$$

From Eq. (1.52) it is obvious that

$$\delta G_{\mu\nu} = \frac{1}{2}(\bar{h}_{\mu,a\nu}{}^a + \bar{h}_{\nu a,\mu}{}^{,a} - \Box\bar{h}_{\mu\nu} - \eta_{\mu\nu}\bar{h}_{b,a}{}^{a,b}). \qquad (1.62)$$

By applying the Lorentz, one obtains

$$\delta G_{\mu\nu} = -\frac{1}{2}\Box\bar{h}_{\mu\nu}. \qquad (1.63)$$

Therefore, the linearized field equations (1.52) become

$$\Box\bar{h}_{ab} = -16\pi\delta T_{ab}. \qquad (1.64)$$

In vacuum, this becomes

$$\Box\bar{h}_{\mu\nu} = 0. \qquad (1.65)$$

Hence, in vacuum, the metric perturbations travel in the form of gravitational waves.

Vacuum Spacetime

The general homogeneous solution to the wave-like equation, based on the superposition principle for linear systems, can be written as

$$\bar{h}_{\mu\nu}(\mathbf{x}, t) = Re \int d^3k A_{\mu\nu}(\mathbf{k}) e^{i(\mathbf{k}\cdot\mathbf{x}-\omega t)} d^3k, \tag{1.66}$$

where, $\bar{h}_{\mu\nu}$ represents the metric perturbation tensor and describes how spacetime is deformed by the presence of gravitational waves, $\omega = |\mathbf{k}|$ is the angular frequency, $A_{\mu\nu}(\mathbf{k})$ are the complex coefficients that determine the amplitude and phase of the gravitational wave at each wave-vector $\mathbf{k}$.

These solutions are subject to the constraint $k^{\mu}(\omega, \mathbf{k})A_{\mu\nu} = 0$ which arises from the Lorentz gauge condition. The solutions represented by $\bar{h}_{\mu\nu}(\mathbf{x}, t)$ are gravitational waves. However, they also encompass a significant number of non-radiative degrees of freedom. Since, these degrees of freedom do not contribute to the physical effects of the propagating gravitational waves they can be removed. In this direction, it is important to consider some key spacetime conditions. Firstly, the global vacuum, where the energy-momentum tensor $T_{\mu\nu} = 0$ throughout spacetime and secondly the asymptotic flatness of spacetime, where the metric perturbation $h_{\mu\nu}$ approaches zero as r tends towards infinity. Attention falls to a specific subset of solutions, those that belong to the category of homogeneous, asymptotically flat solutions of the linearized Einstein equation (1.64). Within this subset, additional gauge specializations are possible, including the choice of the Lorentz gauge. By employing these gauge choices, the metric perturbation can have an exclusively spatial and traceless character. So,

$$h_{tt} = h_{ti} = 0, \tag{1.67}$$

and

$$h = h_i{}^i = 0. \tag{1.68}$$

For the spatial perturbation, the Lorentz gauge implies that it is transverse, as a result of:

$$h_{ij}{}^{,i} = \partial^i h_{ij} = 0. \tag{1.69}$$

Condition (1.69) is known as the transverse-traceless or (TT) gauge, which results in the determination of the gauge for the metric perturbation with only the essential physical information. Initially, there was a symmetric tensor with 10 degrees of freedom due to its 4-dimensional nature. However, the application of the TT gauge conditions introduced 8 constraints, effectively reducing the degrees of freedom to just 2. Furthermore, within the TT gauge, there exists a close mathematical relationship linking the linearized Riemann tensor (1.45) and the metric perturbation, by:

$$R_{itjt} = -\frac{1}{2}\ddot{h}_{ij}^{TT}. \tag{1.70}$$

The two degrees of freedom left within the metric perturbation give rise to two polarization components.

To illustrate this concept, consider a gravitational wave propagating in the z-direction, expressed as: $h_{ij}^{TT} = h_{ij}^{TT}(t - z)$. The Lorentz gauge condition, $h_{zj,z}^{TT} = \partial_z h_{zj,z}^{TT} = 0$ implies that h_{zj}^{TT} must remain constant. In order to maintain asymptotic flatness of spacetime ($h_{\mu\nu} \to 0$ when $r \to \infty$), this constant must be zero. As a result, only the components $h_{xx}^{TT}, h_{xy}^{TT}, h_{yy}^{TT}, h_{yx}^{TT}$ endure. Due to their inherent symmetry and adherence to the trace-free condition, it becomes apparent that only two of these components are truly independent. The two independent waveforms of the gravitational waves are:

$$h_{xx}^{TT} = -h_{yy}^{TT} \equiv h_+(t-z), \tag{1.71}$$

$$h_{xy}^{TT} = h_{yx}^{TT} \equiv h_\times(t-z). \tag{1.72}$$

In matrix form, the transverse-traceless metric perturbation can be expressed as:

$$h_{ij}{}^{TT} = \begin{pmatrix} h_+ & h_x & 0 \\ h_x & -h_+ & 0 \\ 0 & 0 & 0 \end{pmatrix}_{ij} cos[\omega(t-z/c)],$$

where h_+ and $h_\times$ represent the two distinct polarizations of the plane wave: the plus polarization and the cross polarization. Or, in a more simple way:

$$h_{ab}{}^{TT} = \begin{pmatrix} h_+ & h_x \\ h_x & -h_+ \end{pmatrix}_{ab} cos[\omega(t-z/c)].$$

Consequently, the spacetime interval ds^2 can be described by:

$$ds^2 = -c^2dt^2+dz^2+[1+h_+cos[\omega(t-z/c)]]dx^2+[1+h_-cos[\omega(t-z/c)]]dy^2+2h_\times cos[\omega(t-z/c)]dxdy.$$

Generation of Gravitational Waves

Up to now, the description and effects of the gravitational waves corresponded to the far field regions, which are regions far away from any potential source where the amplitude of the wave is really small. To fully understand how gravitational waves are produced, it is crucial to consider the surrounding area of the source, often referred to as the "near field" or "near zone." In this area, the metric perturbation might be significant, and consequently, the linearized approximation described earlier might lose its revelance. In these cases, gravitational waves can carry important amounts of energy and momentum away from their sources, influencing the motion of these sources in what's termed backreaction. In such situations, a complete description requires the use of full nonlinear general relativity (GR) equations or nonlinear corrections.

The linearized Einstein equations (1.62), always have solutions. This is because the d'Alembertian operator is invertible.

$$\bar{h}_{ab}(x) = 16\pi \int d^4x' G(x-x')T_{ab}. \tag{1.73}$$

Similarly, drawing an analogy to electromagnetism, the suitable solution, given that we are addressing a radiation problem, is the retarded Green's function:

$$G(x-x') = -\frac{1}{4\pi|\vec{x}-\vec{x}'|}\delta(ct_{\text{ret}} - xt'), \tag{1.74}$$

so that

$$\Box_x G(x-x') = \delta^4(x-x'),$$

where

$$t_{\text{ret}} \equiv t - \frac{|\vec{x}-\vec{x}'|}{c}, \tag{1.75}$$

where t_{ret} is the retarded time. By using the above on the solution Eq. (1.74),

$$\bar{h}_{ab} = \frac{4\pi}{c^4}\int d^3x' \frac{1}{|\vec{x}-\vec{x}'|} T_{ab}\left(t - \frac{|\vec{x}-\vec{x}'|}{x}, \vec{x}'\right). \tag{1.76}$$

The spatial projector normal to a unit direction $\hat{n}$ is defined as

$$P_{ij}(\hat{n}) = \delta_{ij} - n_i n_j \ , \tag{1.77}$$

and one can form

$$\Lambda_{ij,kl}(\hat{n}) = P_{ik}P_{jl} - \frac{1}{2}P_{ij}P_{kl}. \tag{1.78}$$

Then,

$$h_{ij}^{TT} = \Lambda_{ij,kl}h_{kl}. \tag{1.79}$$

The solution then is,

$$h_{ij}^{TT} = \frac{4G}{c^4}\Lambda_{ij,kl}\int d^3\vec{x}\frac{1}{|\vec{x}-\vec{x}'|}T_{kl}\left(t - \frac{|\vec{x}-\vec{x}'|}{c}, \vec{x}'\right). \tag{1.80}$$

At significant distances $r \equiv |x|$ from a source that is much larger in size $d \ll r$, $|\vec{x}-\vec{x}'| \simeq r - \vec{x}' \cdot \hat{n} + O\left(\frac{d^2}{r}\right)$ while disregarding higher-order terms of r,

$$h_{ij}^{TT}(t,\vec{x}) \simeq \frac{4G}{c^4}\Lambda_{ij,kl}(\hat{n})\int T_{kl}\left(t - \frac{r}{c} + \frac{\vec{x}'\cdot\hat{n}}{c}, \vec{x}'\right)d^3\vec{x}. \tag{1.81}$$

When dealing with a source in motion with non-relativistic velocities over a spatial scale d and a time scale $1/\omega_s$ then $u \sim \omega_s d \ll c$. The emitted frequency of the gravitational wave is $\omega \sim \omega_s$ and so the wavelength $\lambda/2\pi = c/\omega \sim d(c/\omega) \gg d$.

Looking at the Fourier transform of the stress-energy tensor denoted as T_{kl}:

$$T_{kl}\left(t - \frac{r}{c} - \frac{\vec{x}'\cdot\hat{n}}{c}, \vec{x}'\right) = \int \frac{d^4k}{(2\pi)^4}\tilde{T}_{kl}(\omega,\vec{k})e^{-i\omega(t-r/c+\vec{x}'\cdot\hat{n}/c)+i\vec{k}\cdot\vec{x}'}. \tag{1.82}$$

The exponential term of this expression can be expanded as follows:

$$e^{-i\omega(t-r/c+\vec{x}'\cdot\hat{n}/c)+i\vec{k}\cdot\vec{x}'} = e^{-i\omega(t-r/c)} \times \left[1 - i\frac{\omega}{c}x'^i n^i + \frac{1}{2}\left(-i\frac{\omega}{c}\right)^2 x'^i x'^j n^i n^j + ...\right]. \tag{1.83}$$

This expansion can be equivalently represented as an expansion in the time domain:

$$T_{kl}\left(t - \frac{r}{c} - \frac{\vec{x}'\cdot\hat{n}}{c}, \vec{x}'\right) = T_{kl}\left(t - \frac{r}{c}, \vec{x}'\right) + \frac{x'^i n^i}{c}\partial_0 T_{kl} + \frac{1}{2c^2}x'^i x'^j n^i n^j \partial_0^2 T_{kl} + ... \ . \tag{1.84}$$

In this expansion, all the derivatives are estimated at the point $(t - r/c, \vec{x}')$.

This expansion can also be expressed more concisely by utilizing multipole moments of T^{ij}

$$\begin{aligned} S^{ij}(t) &= \int d^3x T^{ij}(t,\vec{x}) \to \text{monopole}, \\ S^{ij,k}(t) &= \int d^3x T^{ij}(t,\vec{x})x^k \to \text{dipole}, \\ S^{ij,kl}(t) &= \int d^3x T^{ij}(t,\vec{x})x^k x^l \to \text{quadrupole}. \end{aligned} \tag{1.85}$$

In this notation, the comma distinguishes between the spatial indices derived from the stress-energy tensor and those arising from $x^{i1}, ..., x^{iN}$. By inserting this expansion and substituting the multipole moments in (1.81), the following is obtained:

$$h_{ij}^{\mathrm{TT}}(t,\vec{x}) = \frac{4G}{rc^4}\Lambda_{ij,kl}(\hat{n})\left[S^{kl} + \frac{1}{c}n^m\dot{S}^{kl,m} + \frac{1}{2c^2}n^m n^p \ddot{S}^{kl,mp} + \dots\right]_{\mathrm{ret}}. \tag{1.86}$$

Here, each factor of x^k in the spatial integral contributes a factor of order d and every time derivative brings a factor of order ω_s. Therefore, this is effectively an expansion in $\omega_s d/c \sim u/c$.
The significance of the introduced momenta becomes apparent when considering the momenta of the energy density T^{00} and linear momentum T^{0i}. These momenta are defined as:

$$\begin{aligned} M &= \frac{1}{c^2}\int d^3x T^{00}(t,\vec{x}), \\ M^i &= \frac{1}{c^2}\int d^3x T^{00}(t,\vec{x})x^i, \\ M^{ij} &= \frac{1}{c^2}\int d^3x T^{00}(t,\vec{x})x^i x^j, \\ M^{ijk} &= \frac{1}{c^2}\int d^3x T^{00}(t,\vec{x})x^i x^j x^k. \end{aligned} \tag{1.87}$$

Similarly, the momenta of $\frac{T^{0i}}{c}$ are defined as:

$$\begin{aligned} P^i &= \frac{1}{c}\int d^3x T^{0i}(t,\vec{x}), \\ P^{i,j} &= \frac{1}{c}\int d^3x T^{0i}(t,\vec{x})x^j, \\ P^{i,jk} &= \frac{1}{c}\int d^3x T^{0i}(t,\vec{x})x^j x^k. \end{aligned} \tag{1.88}$$

From the conservation law of the stress-energy tensor:

$$\begin{aligned} \dot{M} &= 0, \\ \dot{M}^i &= P^i, \\ \dot{M}^{ij} &= P^{i,j} + P^{j,i}, \\ \dot{M}^{ijk} &= P^{i,jk} + P^{j,ki} + P^{k,ij}. \end{aligned} \tag{1.89}$$

and

$$\begin{aligned} \dot{P}^i &= 0, \\ \dot{P}^{i,j} &= S^{ij}, \\ \dot{P}^{i,jk} &= S^{ij,k}. \end{aligned} \tag{1.90}$$

The equations $\dot{M} = 0$ and $\dot{P}^i = 0$ represent the principles of mass conservation and total momentum conservation for the source, respectively. In addition, the identity $\dot{P}^{i,j} - \dot{P}^{j,i} = S^{ij} - S^{ji} = 0$ reflects the conservation of angular momentum for the source.

The momenta of the multipole expansion can now be expressed by using the momenta of energy-density and linear momentum:

$$S^{ij} = \frac{1}{2}\ddot{M}^{ij}, \tag{1.91}$$

$$\dot{S}^{ij,j} = \frac{1}{6}\dddot{M}^{ijk} + \frac{1}{3}\left(\ddot{P}^{i,jk} + \ddot{P}^{j,ki} + \ddot{P}^{k,ij}\right). \tag{1.92}$$

Similar relationships hold for higher-order momenta.

Using Eq. (1.91) in Eq. (1.86) and retaining the main term gives:

$$\left[h_{ij}^{\mathrm{TT}}(t,\vec{x})\right]_{\text{quad}} = \frac{2G}{rc^4}\Lambda_{ij,kl}(\hat{n})\ddot{M}^{kl}(t-r/c). \tag{1.93}$$

The reduced quadrupole moment tensor is:

$$\begin{aligned} Q^{ij} &\equiv M^{ij} - \delta^{ij}M_{kk}, \\ &= \int \left(x^i x^j - \frac{1}{3}r^2\delta^{ij}\right)\rho(t,\vec{x})d^3x. \end{aligned} \tag{1.94}$$

where $\rho = T^{00}/c^2$, is the mass density. By Substituting in the last expression,

$$\begin{aligned} \left[h_{ij}^{\mathrm{TT}}(t,\vec{x})\right]_{\text{quad}} &= \frac{2G}{rc^4}\Lambda_{ij,kl}(\hat{n})\ddot{Q}_{kl}(t-r/c), \\ &\equiv \frac{2G}{rc^4}\ddot{Q}_{ij}^{\mathrm{TT}}(t-r/c). \end{aligned} \tag{1.95}$$

This equation is known as the quadrupole formula for gravitational wave radiation.

When considering the direction of wave propagation along the z-axis, the expression becomes:

$$\Lambda_{ij,kl}(\hat{\mathrm{z}})\ddot{M}^{kl} = \begin{pmatrix} \frac{1}{2}\left(\ddot{M}_{11} - \ddot{M}_{22}\right) & \ddot{M}_{12} & 0 \\ \ddot{M}_{21} & -\frac{1}{2}\left(\ddot{M}_{11} - \ddot{M}_{22}\right) & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{1.96}$$

The "+"and"×" polarization amplitudes are given by:

$$\begin{aligned} h_+ &= \frac{1}{r}\frac{G}{c^4}(\ddot{M}_{11} - \ddot{M}_{22}) \\ h_\times &= \frac{2}{r}\frac{G}{c^4}\ddot{M}_{12} \end{aligned} \tag{1.97}$$

These quantities are computed in retarded time $t - r/c$.

Interaction of GWs with a Detector

In General Relativity, the idea of a "gravitational force" as described by Newtonian gravity is replaced by the concept that objects that are not being restrained in spacetime follow curved lines, the geodesics. Thus, the motion of objects in general relativity is described by geodesic equations, which dictate how the coordinates of these objects evolve in curved spacetime. The geodesic equations can be expressed in terms of the proper time experienced by the particle τ or the coordinate time t, depending on the context. By substituting the derivatives into the geodesic equations (1.32) with the coordinate time t and using $a = t$ with the spatial equations $a = j$, the coordinate acceleration of a free falling body is

$$\frac{\mathrm{d}^2x^i}{\mathrm{d}t^2} = -\left(\Gamma^i_{tt} + 2\Gamma^i_{tj}v^j + \Gamma^i_{jk}v^jv^k\right) + v^i\left(\Gamma^t_{tt} + 2\Gamma^t_{tj}v^j + \Gamma^t_{jk}v^jv^k\right), \tag{1.98}$$

where $u^i = dx^i/dt$ is the coordinate velocity of the object along the spatial direction j. In linearized gravity, where the non-flat part of the metric is primarily influenced by a gravitational wave in the transverse-traceless (TT) gauge, and under the assumption of non-relativistic motion for a test body ($u^i \ll 1$), we can simplify the equations of motion as follows:

$$\frac{d^2x^i}{dt^2} + \Gamma^i_{tt} = 0. \tag{1.99}$$

Within the TT gauge,

$$\Gamma^i_{tt} = \Gamma_{itt} = \frac{1}{2}(2\partial_t h^{TT}_{jt} - \partial_j h^{TT}_{tt}) = 0. \tag{1.100}$$

Here $h^{TT}_{at} = 0$. Substituting the simplified Christoffel symbols into the equation of motion, $d^2x^i/dt^2 = 0$. This equation suggests that, in the context of linearized gravity and TT gauge, the test body follows Newton's first law of motion, which states that an object that is not being acted upon by an outside force will remain in its current state of rest or motion. In the transverse-traceless gauge of general relativity, the coordinates describing the position of a slowly moving, freely falling body are not affected directly by the gravitational wave. However, even though the objects themselves remain stationary, the coordinates are distorted by the GWs. Therefore, in the context of general relativity, it is essential to emphasize the importance of coordinate-invariant observables, in other words quantities that do not depend on the specific choice of coordinates, and that provide a more reliable way to describe physical phenomena.

For example, consider the case where there are two freely falling particles, at $z = 0$, at a distance L_c between them on the x-axis. If a gravitational wave propagates in the TT gauge along the z-axis, denoted by $h^{TT}_{t,z}$, the proper distance L between these two particles is affected by the spacetime curvature caused by the GW. The proper distance L is given by:

$$\begin{aligned} L &= \int_0^{L_c} \mathrm{d}x\sqrt{g_{xx}} = \int_0^{L_c} \mathrm{d}x\sqrt{1 + h^{\mathrm{TT}}_{xx}(t, z = 0)} \\ &\simeq \int_0^{L_c} \mathrm{d}x\left[1 + \frac{1}{2}h^{\mathrm{TT}}_{xx}(t, z = 0)\right] = L_c\left[1 + \frac{1}{2}h^{\mathrm{TT}}_{xx}(t, z = 0)\right]. \end{aligned} \tag{1.101}$$

In the TT gauge, where the coordinates of each particle remain fixed, it is worth noting that if a different gauge were used where the two particles are allowed to move relative to the coordinates, the limits of integration in Equation (1.101) would need to be adjusted accordingly. The equation of proper distance reveals that the proper separation between the two particles oscillates, leading to a fractional length change $\delta L/L$. This fractional change in proper distance is approximately given by

$$\frac{\delta L}{L} \simeq \frac{1}{2}h^{TT}_{xx}(t, z = 0). \tag{1.102}$$

Here h^{TT}_{xx} is the amplitude of the gravitational wave component in the x direction at $z = 0$. Although the TT gauge is used, it is important to note that the result obtained is not gauge-dependent. The wave strain h, is the amolitude of a gravitational wave.

In laser interferometric observatories designed to detect gravitational waves, the proper distance L is of great significance, because of its direct correlation with the accumulation of phase information. The accumulated phase change, denoted $\delta\phi$, occurs when a photon traverses the arm of a laser interferometer in the presence of a gravitational wave. The difference in phase $\delta\phi$ signifies a shift in the phase of the laser light within the interferometer. This phase change is mathematically expressed as $\delta\phi = 4\pi\delta L/\lambda$, where λ denotes the wavelength of the laser light utilized in the interferometer and δL corresponds to the alteration in the length of one of the interferometer arms caused by the gravitational wave. Gravitational waves propagate, stretch, and squeeze spacetime, resulting in minute alterations in distances, including the length of the interferometer's arms.

The geodesic deviation equations describe how the separation L between geodesics changes over time due to the curvature of spacetime. Thus, these equations explore how nearby particles or objects, that are moving along geodesics, experience changes in their relative positions due to gravitational waves. Taking into account two separate but close paths in spacetime given by $x^a = z^a(\tau)$ with velocity $u^a(\tau) = dz/d\tau$ and $x^a(\tau) = z^a(\tau) + L^a(\tau)$, where $L^a(\tau)$ is a small displacement vector, the displacement vector can be written as $\vec{L} = L^a \partial_a$ and describes the motion from one point on a path to a nearby point on the same path. The behavior of this separation vector $\vec{L}$ is mathematically described by the geodesic deviation equation:

$$u^b \nabla_b \left(u^c \nabla_c L^a\right) = -R^a{}_{bcd}[\vec{z}(\tau)] u^b L^c u^d. \tag{1.103}$$

Here, $R^a{}_{bcd}[\vec{z}(\tau)]$ are the components of the Riemann curvature tensor at the position of the geodesics described by the path $\vec{z}(\tau)$ at a specific proper time τ. The equation essentially describes how the change in the separation vector L^a between two nearby geodesics is influenced by the curvature. This equation is valid up to the linear order in L^a.

In the context of a local proper reference frame, the geodesic deviation equation (1.103) is valid, and its implications are as follows

$$z^i(\tau) = 0, \qquad g_{ab}(\tau, 0) = \eta_{ab}, \qquad \Gamma^a_{bc} = 0. \tag{1.104}$$

This suggests that the metric has the form

$$ds^2 = -dt^2 + dx^2 + O\left(\frac{x^2}{R^2}\right). \tag{1.105}$$

The radius of curvature of spacetime, denoted by R, is given by $R^{-2} \sim ||R_{abcd}||$. The proper distance between two geodesics located at $x^i = 0$ and $x^i = L^i(t)$ is $\mathbf{L} = \sqrt{L_i L_i}$, with fractional corrections of the order L^2/R^2. The curvature of spacetime due to a gravitational wave (GW) of amplitude h and wavelength λ is $R^{-2} \sim h/\lambda^2$, so the fractional errors are $\sim hL^2/\lambda^2$.of the detectors. Since the detectors have a separation much smaller than the wavelength of the gravitational wave, the errors caused by this are much smaller than the fractional distance changes of approximately h due to the GW. Therefore, the separation of the detectors is considered appropriate.

In the coordinates of the local proper reference frame Eq. (1.104), the equation for the geodesic deviation (1.103) can be expressed as

$$\frac{\mathrm{d}^2 L^i(t)}{\mathrm{d}t^2} = -R_{itjt}(t, 0) L^j(t). \tag{1.106}$$

The covariant time-derivative operator $u^a \nabla_a$ is replaced with $\partial/\partial t$, $\vec{u} = \partial_t$, and $L^a = (0, L^i)$ is used. The quantity R_{itjt} is invariant in linearized theory, so any suitable coordinate system can be used to calculate it. Using the expression (1.70) for the Riemann tensor in terms of the metric perturbation h^{TT}_{ij} in the TT gauge, it follows that

$$\frac{\mathrm{d}^2 L^i}{\mathrm{d}t^2} = \frac{1}{2} \frac{\mathrm{d}^2 h^{\mathrm{TT}}_{ij}}{\mathrm{d}t^2} L^j. \tag{1.107}$$

Integrating this equation with $L^i(t) = L^i_0 + \delta L^i(t)$ and $\delta L \ll \vec{L_0}$ gives

$$\delta L^i(t) = \frac{1}{2} h^{TT}_{ij}(t) L^j_0. \tag{1.108}$$

If one chooses Cartesian coordinates such that the two arms of the interferometer lie along the x-axis and y-axis, with the beam splitter at the origin and the propagation of GW along the z -axis, the only nonzero components of the metric perturbations are $h_{xx}^{TT} = -h_{yy}^{TT} = h_+$ and $h_{xy}^{TT} = h_{yx}^{TT} = h_\times$. These two polarization components, $h_+(t-z)$ and $h_\times(t-z)$, were obtained in equations (1.72) and (1.71). Taking the beam splitter at $\vec{x} = 0$ and the end mirror as the two nearby geodesics, Eq. (1.108) can be used to calculate the relative change in the length of the two arms, where $L = |\vec{L}|$,

$$\frac{\delta L_x}{L} = \frac{1}{2} h_+, \qquad \frac{\delta L_y}{L} = -\frac{1}{2} h_+. \tag{1.109}$$

where x and y refer to two distinct arms.

Gravitational waves (GWs) act in a tidal manner, compressing along one axis and stretching along the other. In this setup, the detector is only sensitive to the + polarization of the GW. The $\times$ polarization has a similar effect, except that it compresses and stretches along axes that are rotated 45^o relative to the x and y axes. Figure 1.1 shows the force lines for the two different polarizations.

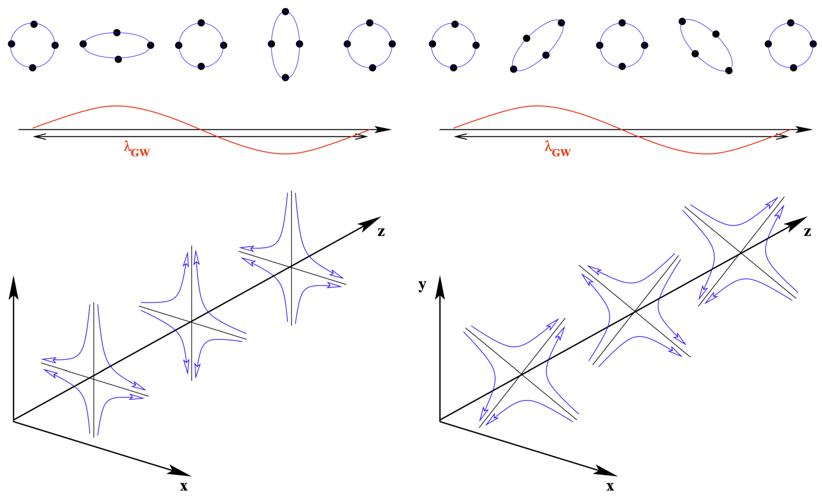

Figure 1.1: Gravitational waves can be characterized by two distinct transverse polarizations: left-handed (plus) and right-handed (cross). This illustration shows a gravitational wave propagating along the z-axis and impacting the (x, y) plane. Figure from [9].

Neutron Stars

Stars have intense gravitational pressure in their cores that allows them to fuse hydrogen into helium through a series of nuclear reactions. For stars more massive than the Sun, as the star ages, it continues to fuse heavier elements in its core, including helium into carbon, carbon into oxygen, and so on. Eventually, when the core of the star becomes mostly composed of iron, iron fusion, instead of releasing energy, consumes it. The iron core becomes unstable as it accumulates more mass and is unable to support itself against the force of gravity. When the core reaches a critical mass, it rapidly collapses under its own weight. This core collapse triggers an extremely violent explosion known as a supernova. The outer layers of the star are blasted into space at tremendous velocities, releasing an immense amount of energy in the process. Depending on the mass of the collapsing core, one of two outcomes can occur. A neutron star or a black hole. If a star has a mass between 1 and 2 solar masses($M_\odot$) then what remains of its collapsed core becomes a neutron star. Neutron stars have a radius of about 12km (with a current uncertainty of about one km), they can spin very fast, and are the densest objects known. Neutron stars provide a unique opportunity to gain knowledge about matter, gravity, and the extreme conditions of the universe. They are a perfect astrophysical laboratory for gaining valuable insights [22, 17].

In the fall of 2015 a groundbreaking discovery occurred, as the first direct observation of gravitational waves was made from the merging of a binary black hole system by the LIGO interferometer [1]. This marked the first-ever direct observation of gravitational waves, confirming a key prediction of Albert Einstein's theory of General Relativity. Two years later, in August 2017, the LIGO and Virgo collaboration successfully detected the first gravitational waves originating from a binary neutron star (NS) merger [3]. The observation of this coalescence has advanced the understanding of the merging process and provided some initial insights into the properties of neutron stars.

The characteristics of extremely dense matter, and in particular the Equation of State (EoS), continue to be a subject of incomplete understanding, especially when considering densities at or beyond nuclear saturation. Nuclear saturation density represents the equilibrium density of the atomic nuclei and serves as a reference point for the study of matter under extreme conditions [22]. This difficulty in understanding is primarily due to the complex behaviors and interactions of numerous nuclear particles, commonly known as the nuclear many-body problem. Furthermore, it is unclear whether extra exotic particles, such as hyperons, exist in higher densities or even if a phase transition to deconfined quark matter occurs in neutron stars. Quarks, as elementary constituents of matter, are known to form protons, neutrons, and various hadrons through a strong nuclear force. However, when temperatures and pressures are extremely high, it is proposed that quarks can be released from their confinement and become free particles, mainly interacting through the strong force [17].

Moreover, in the case of nonrotating neutron stars, the structural properties are uniquely determined by the equation of state (EoS) and are mathematically described by the Tolman-Oppenheimer-Volkoff equations. These equations account for relativistic hydrostatic equilibrium. The relationship between mass and radius of a neutron star, the deformability due to tidal forces, and the upper limit of mass for nonrotating neutron stars still have considerable uncertainties. The goal is to establish a connection between the observable aspects of the gravitational wave signal and the properties of the equations of state, with a specific focus on the post-merger phase. Neutron stars, which are extremely dense and are primarily composed of neutrons, are essential for investigating these challenges [16].

Merger Stages

In chronological order, the three phases of the coalesence are the inspiral, the merger, and the ringdown, which involves the remnant object settling into a stable state. At the beginning of the inspiral phase, the relative velocity and gravitational field are notably low. As they revolve around one another, gravitational waves are released, resulting in a decrease in the orbital energy of the system. This causes the objects to gradually move inwards in a spiral motion. As they get closer to each other, they experience a stronger gravitational attraction, leading to the release of gravitational potential energy [16]. According to the law of energy conservation, this lost potential energy is transformed into kinetic energy, causing the orbits of the compact objects to accelerate, which in turn results in the emission of even more powerful gravitational waves. Consequently, the objects lose more orbital energy, pulling them even closer, and this cycle continues. Ultimately, this gradual loss of energy in the form of gravitational waves leads to the merger of the two celestial objects. The gravitational wave signal is mainly determined by the orbital motion, resulting in a characteristic chirping signal with increasing amplitude and frequency. The duration τ of the inspiral phase is significantly dependent on the masses of the binary components and is highly sensitive to the starting orbital separation, α, where $\tau \propto \alpha^4$. A gravitational wave signal from a binary system of neutron stars will reach a frequency of approximately 10 Hz in the gravitational wave spectrum only a few tens of seconds prior to merging. At this point, the frequency of the emitted gravitational waves falls within the sensitivity window ground-based detectors, which spans a range from about 10 Hz to $O(1\text{kHz})$ for the current generation of detectors [18].

In a compact binary system, the individual masses can be determined with some accuracy, but a certain combination, called the chirp mass, can be obtained very accurately. This is defined as

$$M_{\text{chirp}} = \frac{(M_1 M_2)^{3/5}}{(M_1 + M_2)^{1/5}}, \tag{1.110}$$

where M_1, M_2 are the individual masses. The impact of the mass ratio $q = M_2/M1$ is less prominent in the earlier stages of the inspiral, but becomes increasingly significant as the inspiral phase approaches its conclusion [12]. The chirp mass alone can only provide an estimate of the total mass if the mass ratio is not well defined. The determination of the individual masses of the binary system has good accuracy only for mergers that have a high signal-to-noise ratio (SNR).

When the orbital period reaches 1 millisecond and the merger of the binary approaches, there is a pronounced tidal distortion in the binary system. The level of this distortion depends on the masses of the stars and the equation of state (EoS) of the neutron star matter. Due to the significant orbital angular momentum, the stars merge with a high impact parameter. The final result of this merger is based on the combined mass of the binary and the EoS. For higher masses, the remnant star cannot resist the gravitational forces acting upon it and collapses into a black hole in less than 1 millisecond ("prompt collapse"). In the case of a lower total mass, the remnant of the merger can also form a neutron star. Whether the merger remnant will collapse into a black hole or form a heavier neutron star depends on the matter properties of the NS. In the event of a collapse, some matter may be ejected from the remnant, resulting in the formation of a torus encircling the central black hole [16].

If the merger generates a neutron star, the resulting remnant is initially a rapidly rotating and highly distorted structure that is subject to intense oscillations. Because the remnant is spinning rapidly, it has the potential to resist gravitational collapse, even when its total mass exceeds the maximum mass limit for nonrotating neutron stars. Redistribution of angular momentum, coupled with losses from gravitational waves, mass ejection, and neutrino cooling, may lead to a "delayed collapse" of the remnant on a timescale of tens of hundreds of milliseconds. In the case of very low-mass components and depending on the EoS, even a stable remnant may be formed [16].

The post-merger gravitational wave spectrum

The main target is to create methods for extracting previously unknown characteristics of neutron stars and high-density matter, by analyzing observable data and specifically the gravitational wave signals emitted during neutron star mergers [7]. The concept is that the equation of state has a direct impact on the dynamics of a merger, consequently leaving a mark on the gravitational wave signal during the post-merger phase. A likely result of a binary neutron star merger is a meta-stable remnant consisting of a differentially rotating neutron star. Three different representative post-merger spectra for such a scenario are shown in Fig. (1.2). The lower frequency segment of the spectrum is mainly produced in the inspiral

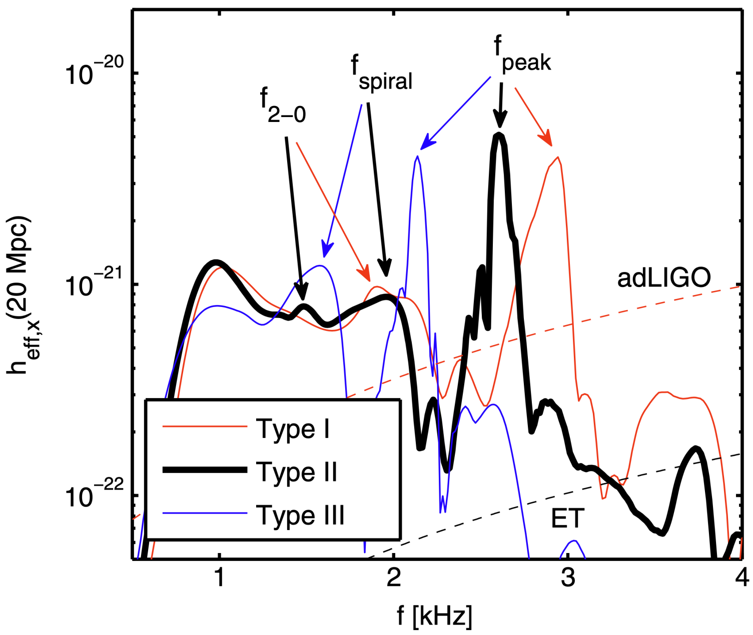

Figure 1.2: Post-merger GW spectra for three different cases, using the DD2 (black), NL3 (blue) and LS220 (red) EOSs (cross polarization along the polar axis at a reference distance of 20 Mpc). The dashed lines show the anticipated sensitivity curves of Advanced LIGO [2] (red) and of the Einstein Telescope [20]. Figure from [6].

phase. As the amplitude grows and the binary system approaches the merger phase, the gravitational wave frequency increases to around 1 kHz. Within the post-merger spectrum, multiple visible peaks appear with O(kHz) frequencies. These peaks are linked to specific oscillation modes and dynamic characteristics of the post-merger remnant [24, 6]. There is a main oscillation frequency f_{peak}, which is consistently present in all merger simulations that do not immediately collapse into a BH. From an observational perspective, this peak is the most significant as it can be detected with the highest SNR [11].

Origin and interpretation of peaks in postmerger gravitational-wave spectra

The post-merger phase of a neutron star merger produces a frequency spectrum with many distinct peaks. To effectively detect and interpret these post-merger GW signals, it is essential to understand the underlying physical processes responsible for generating the different features of the spectrum. It is therefore important to gain a deeper understanding of the origins of the components within the post-merger GW spectrum. Exploring the sources and interconnections among these various components holds significant potential to uncover more information about the incompletely understood equation of state of high-density matter beyond what can be determined from the measurement of the main peak.

The characteristics of the primary peak, f_{peak}, provide insight into the origins of the two most distinct secondary peaks at lower frequencies, referred to as f_{2-0} and f_{spiral} [7]. In observations, for frequencies lower than f_{peak}, only secondary peaks are of importance, as the sensitivity of ground-based gravitational wave detectors drops significantly at higher frequencies. An effective method for examining the oscillation modes in rotating stars involves the extraction of their eigenfunctions through Fourier analysis of simulation data [23]. The eigenfunction extraction process is as follows: The process is initiated by performing a Fourier analysis on the pressure evolution on a grid of fixed points that covers the equatorial plane (a two-dimensional plane perpendicular to the star's axis of rotation, effectively dividing it into two equal halves). From the results of the Fourier spectra, the next step is to determine the dominant frequency f_{peak}. This frequency is found consistently throughout the entire star. The Fourier amplitude is extracted at all points in the equatorial plane, and specifically at the discrete frequency f_{peak}. The outcome of this process is a two-dimensional distribution of the amplitude, which represents the eigenfunction of the oscillation mode under examination. It is important to note that the overall scaling is not significant, since the eigenfunctions are strictly defined as linear perturbations.

As an illustrative example, Figure 1.3 provides a color-coded representation of this eigenfunction. This particular example exhibits a distinct quadrupolar structure (characterized by an azimuthal mode number "$m = 2$") with radial nodal lines [24]. This analysis thus offers evidence that the primary peak in the gravitational wave (GW) spectrum originates from the fundamental quadrupolar fluid mode.

The dynamics of the merger process suggest that the fundamental quasi-radial mode of the remnant is likely to be activated at a specific frequency denoted as f_0. This mode is nearly spherically symmetric and produces only weak gravitational wave emission. Nevertheless, a non-linear interaction between the quasi-radial oscillation and the quadrupolar mode leads to the emission of strong GWs, providing an explanation for some of the secondary peaks. The combination of the two modes results in the emergence of quasi-linear combination frequencies, specifically $f_{2\pm0} = f_{peak} \pm f_0$ [24]. To link specific characteristics in the gravitational wave spectrum with this mechanism, it is important to identify the quasi-radial mode f_0 through the hydrodynamical evolution, utilizing the Fourier technique described above. After determining f_0 and f_{peak}, the secondary peaks in the GW spectrum can be identified, which can be interpreted as quasi-linear combination frequencies $f_{2\pm0}$.

In several cases, it is evident that there is at least one additional secondary peak at frequencies lower than f_{peak}. In [6], it was shown that this secondary peak is created by the orbital motion of two bulges that form immediately after the merger, appearing on the surface of the remnant, see Fig. (1.4).

During the merging process, the stars become strongly distorted due to tidal forces. The material at the outer edges of the stars that have been stretched by the tidal force cannot

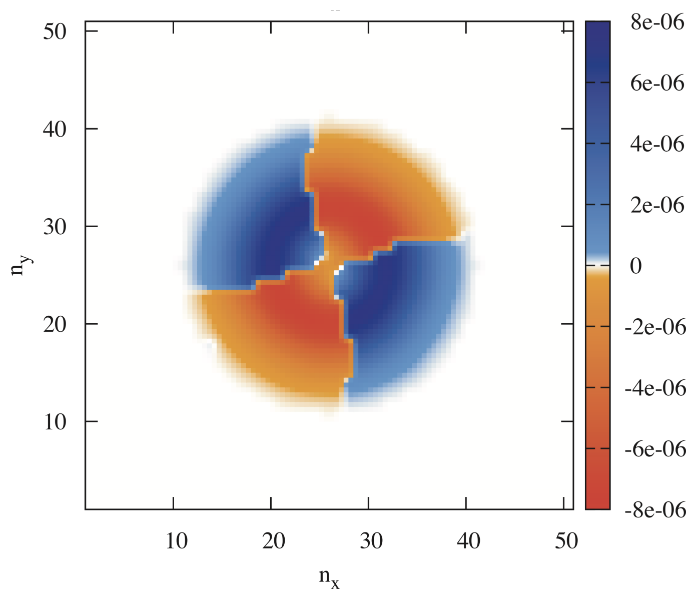

Figure 1.3: Eigenfunction corresponding to the pressure oscillation at the frequency $f = f_{\text{peak}}$ within the equatorial plane. The merger is of a binary neutron star system with masses 1.35-1.35 $M_{\odot}$, using the Shen EoS. Figure from [24].

keep up with the faster rotation of the cores of the original stars that make up the inner part of the remnant. This results in the formation of two bulges at opposite sides of the central remnant, which orbit it at a slower rate. These bulges orbit the central remnant at a reduced orbital frequency and typically disintegrate in a matter of milliseconds. This distinctive feature in the gravitational wave spectrum is referred to as f_{spiral} [6]. Interestingly, it has been observed that the f_{spiral} feature is more intense in mergers with low total binary masses and stiff equations of state (EoSs). This is understandable, since lower total binary masses (M_{tot}) and stiffer equations of state (EoSs) indicate stars that are less tightly bound to each other. This, in turn, promotes the formation of larger tidal bulges. The gap between the frequencies, $f_{\text{peak}} - f_{\text{spiral}}$, corresponds to a frequency that is present in the temporal development of the central lapse function but does not match the frequency of the primary quasi-radial mode. This modulation at low frequencies becomes prominent in the case of low-mass neutron star mergers with rather stiff equations of state (EoS). The cause of this modulation can be attributed to the orientation of the bulges with respect to the pattern of quadrupolar deformation within the core. This orientation influences the overall compactness of the system, which is reflected in the evolution of the lapse function [6].

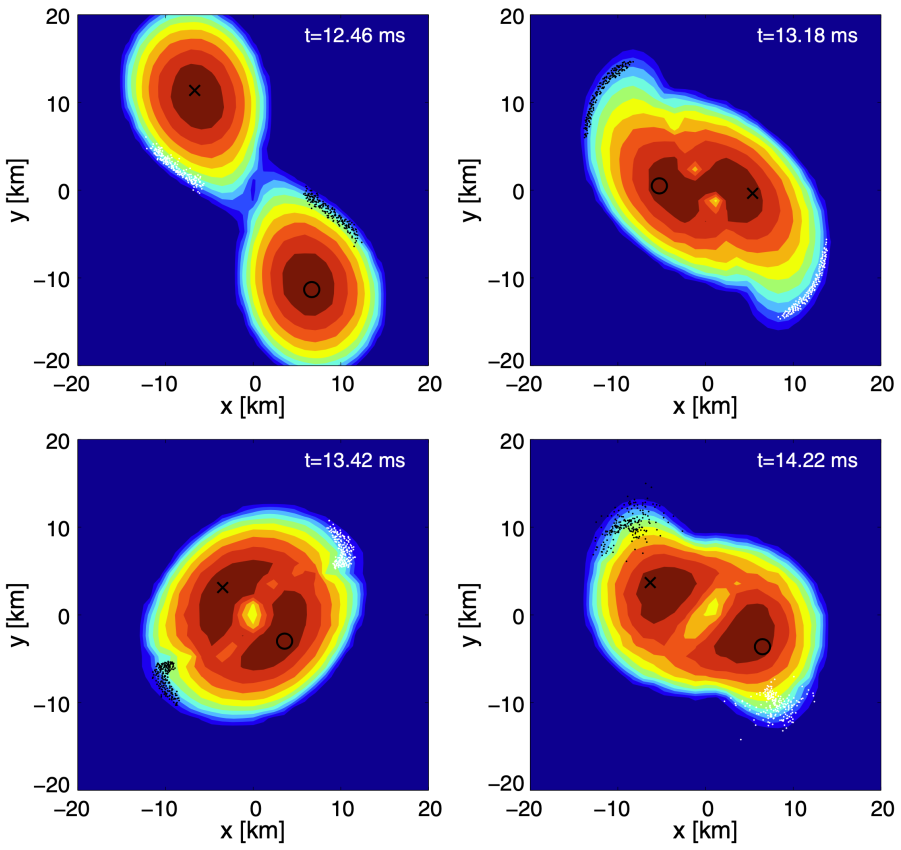

Figure 1.4: The evolution of rest-mass density in the equatorial plane is illustrated here. The black and white dots indicate the positions of chosen fluid elements of the antipodal bulges. The orbital motion of this spiral deformation pattern produces a new peak in the gravitational wave spectrum, f_{spiral}. Figure from [6].

Chapter 2

Post-Merger Models in the Frequency Domain

Waveform Models

Gravitational waves offer a unique view of the Universe. To date, a significant number of gravitational wave events have been observed. Up until the previous (O3) observational run of the LIGO-Virgo-KAGRA (LVK) Collaboration, nearly 90 observations of merging compact objects were made [4]. Periodically, gravitational-wave detectors undergo upgrades aimed at improving their sensitivity and precision. These enhancements are crucial for detecting gravitational waves originating from sources located at greater distances within space. After the latest upgdrades, the O4 science run is currently ongoing. In the coming years, new gravitational wave observatories, including ground-based detectors such as the Einstein Telescope [20] and Cosmic Explorer [14], are planned to enhance our observational capabilities. Additionally, the Laser Interferometer Space Antenna (LISA) [5] is scheduled for launch in the second half of the 2030's, which will further expand our ability to detect gravitational waves from sources emitting at lower frequencies.

Solving Einstein's field equations directly is a complex task due to the highly nonlinear nature of the equations and their dependence on numerous variables. There are ten independent components in Einstein's field equations, making them a set of ten partial differential equations. While it is theoretically possible to solve these equations analytically for specific, simplified scenarios, solutions are extremely challenging for more complex systems, such as binary black hole or binary neutron star mergers. Therefore, waveform models are required to facilitate the detection and sky localization of gravitational waves, estimate source parameters, and validate Einstein's theory of general relativity.

These models are developed using a variety of approaches, including analytical methods, numerical simulations, or hybrid methods that combine multiple techniques. Analytical methods involve applying mathematical principles and solving differential equations to describe the behavior of astrophysical systems. At the other end of the spectrum, Numerical Relativity (NR) directly solves Einstein's equations through high-performance computing, delivering precise predictions for the strong-field dynamics of merging binaries. Some analytical methods [10] are as follows :

- Post-Newtonian (PN) expansions: provide approximate solutions to Einstein's field equations for the metric tensor in the inspiral phase. These expansions use small parameters (such as $1/c^2$) to quantify deviations from Newton's gravity, making them suitable for weak field scenarios. While higher-order terms can enhance accuracy, in strong-field situations numerical terms are typically used.

- Perturbation theory: is a mathematical approach used to study small deviations from known solutions. In this technique, the problem is split into a known part, which could be a non-rotating black hole, and a small perturbation. It finds extensive use during the linearized phase of gravitational wave modeling, particularly in scenarios with weak gravitational fields where deviations from a flat spacetime background are small. Such conditions are often encountered during the early inspiral phase of binary systems, where the gravitational interaction is relatively weak, or when the two involved objects significantly differ in mass, leading to a more predictable perturbative response. It is also used in the ring-down phase of binary black hole mergers, assuming that the emitted gravitational waves are a combination of quasi-normal models of the final black hole.

- Effective One Body (EOB) techniques blend analytical approximations with numerical methods to comprehensively model binary systems, offering insights into inspiral, merger, and ringdown phases. This technique translates the two-body problem into that of a single particle in an effective metric [8].

These models provide valuable insights into properties such as the stars' masses, radii, and equations of state, which are challenging to obtain through alternative methods.

By comparing observed waveforms to those predicted by General Relativity (GR) and identifying deviations, one could probe the fundamental nature of gravity itself. The frequency evolution of GWs allows us to infer the masses of the objects involved, while the amplitude of the signals, combined with knowledge of the source masses, provides an estimate of the distance to the system. Moreover, through careful analysis of the time of arrival, amplitude, and phase of GW signals at multiple detectors, we can constrain the sky location of the source. The modulations of amplitude and phase also offer insights into properties such as spins and eccentricity of the binary system.

Fourier analysis

Continuous Fourier Transform

The Fourier transform is an integral transformation that converts functions from the time or space domains into a form that describes the frequencies present in the original function. The result of the transform is a complex valued function of frequency and is used to analyze and interpret the signal in the frequency domain. In the context of gravitational waves, the Fourier transform plays a crucial role in characterizing and understanding the waveforms emitted by astrophysical sources, enabling us to extract valuable insights from the data. The continuous Fourier transform of a time series $h(t)$ is given by:

$$\tilde{h}(f) = \int_{-\infty}^{\infty} h(t) e^{-i2\pi f t}\, dt, \tag{2.1}$$

where f is frequency.

Discrete Fourier Transform

The transition from the Continuous Fourier Transform (CFT) to the Discrete Fourier Transform (DFT) is a process that involves converting continuous data into a discrete representation that enables the examination of the signal within the discrete domain. This process uses the trapezoidal rule, a numerical integration method that approximates the integral of a function by dividing the area under the curve into a series of trapezoids. The accuracy of this method increases with a greater number of subintervals.

To achieve this transformation, the continuous signal of interest is discretized by selecting $N+1$ specific time points that are uniformly separated by an interval Δt. These distinct time points, t_n, correspond to the values h_n for $n = -\frac{N}{2}$,..., $\frac{N}{2}$. We assume that the signal starts at time 0 and its total duration is T. The integral in the definition of the Fourier transform is then converted to a sum of discrete parts :

$$\begin{aligned}
\tilde{h}(f_k) &= \int_0^T h(t)e^{-i2\pi f_k t}\,dt \\
&\simeq \sum_{n=-N/2}^{N/2-1}\left(h_n e^{-i2\pi f_k t_n} + h_{n+1}e^{-i2\pi f_k t_{n+1}}\right) \\
&= \left(h_{-N/2}e^{-i2\pi f_k t_{-N/2}} + 2h_{-N/2+1}e^{-i2\pi f_k t_{-N/2+1}} + \ldots + h_{N/2-1}e^{-i2\pi f_k t_{N/2-1}} + h_{N/2}e^{-i2\pi f_k t_{N/2}}\right)\frac{\Delta t}{2}.
\end{aligned}$$

We further assume that the signal is periodic (a fundamental assumption in Fourier theory), so that $h_{-N/2} = h_{N/2}$. Then,

$$\begin{aligned}
\tilde{h}(f_k) &\simeq \sum_{n=-N/2}^{N/2-1} h_n e^{-i2\pi f_k t_n}\Delta t \\
&= \Delta t \sum_{n=-N/2}^{N/2-1} h_n e^{-i(2\pi k/N)n} \\
&= \Delta t H_k,
\end{aligned} \tag{2.2}$$

where

$$H_k = \sum_{n=-N/2}^{N/2-1} h_n e^{-i\frac{2k\pi}{N}n}, \tag{2.3}$$

defines the discrete Fourier transform (DFT).

Both the discrete signal and the DFT are periodic, so the samples of the discrete signal and the DFT can be rearranged by moving them forward for half a cycle. This will cause samples to be labeled $n = 0$,..., $N-1$. As a result, the discrete Fourier transform of a function $h(t)$ becomes:

$$H_k = \sum_{n=0}^{N-1} h_n e^{-i\frac{2k\pi}{N}n}, \tag{2.4}$$

which is a more common convention in the literature.

Convolution

The continuous Fourier transform of a product of two functions is the continuous convolution of the two individual Fourier transforms. So, given two functions $h(t)$ and $g(t)$ and their

Fourier transforms $\tilde{h}(f)$, $\tilde{g}(f)$, the convolution of h and g, denoted as $(\tilde{h} * \tilde{g})(f)$, is expressed as:

$$(\tilde{h} * \tilde{g})(f) = \int_{-\infty}^{\infty} \tilde{h}(f')\tilde{g}(f - f')\, df', \tag{2.5}$$

and coincides with the Fourier transfor of the product $h(t)g(t)$.

Analytic Waveform Models in the Time Domain

Most analytical waveform models for the post-merger phase of binary neutron star mergers have been defined in the time domain. This includes models that use physical parameters (amplitude, frequency, frequency drift, damping time and phase), based on the oscillation properties of a perturbed star, as well as models that a purely phenomenological, using a large number of parameters with no immediately physical interpretation. A specific example of a time-domain model with physical parameters, which we are going to use in this work, models the post-merger signal as a sum of damped oscillators. It is written as

$$h(t) = \sum_{i=0}^{N} A_i e^{-(t-t_{\text{merger}})/\tau_i} \sin(2\pi f_i (t - t_{\text{merger}}) + \phi_i), \tag{2.6}$$

where,

- N is the number of dampled oscillators participating in the model,
- t_{merger} is the time at which the amplitude of the gravitational wave becomes maximum, indicating the onset of merger. It is typically expressed in seconds (s).
- $h(t)$ is the strain of the wave at a given time; a dimensionless quantity that measures the relative stretching and squeezing of lengths in a spacetime through which the gravitational wave travels.
- A_i represents the dimensionless amplitude of each damped oscillator.
- τ_i: This parameter characterizes the timescale at which the amplitude of each component decays over time due to damping. It is also measured in seconds.
- f_i: Is the frequency of the $i-$th component and represents the number of oscillations per unit of time. The frequency f is measured in Hertz (Hz) and its inverse, the period T, is measured in seconds (s).
- ϕ_i: is the phase angle of the $i-$th component. This parameter determines the initial phase of each damped oscillator in the waveform model. It is measured in radians.

In parameter estimation computations, using, for example, Bayesian methods, a large number of computations of the waveform model is required. However, the criterion for comparing the model to data is formulated in the frequency domain. Hence, for each individual evaluation, one needs to perform a numerical DFT of the analytic model, which increases the computational cost. It is, therefore, desirable to formulate post-merger waveform models directly in the *frequency domain.* In this work, we are going to transform analytic time-domain models to corresponding frequency-domain models using the continuous Fourier transform.

We used Mathematica to find the continuous Fourier transform of analytic time-domain models. Our results were then directly compared to the numerical DFT of the same models, with the same parameters, to validate the correctness of our results.

Computational Workflow

The computational part of this work was performed in a Linux environment. Continuous transforms were obtained in `Mathematica 11.0` and discrete transforms were obtained in `Python 3.10`. The libraries employed in `Python` were `NumPy 1.24.3`, `SciPy 1.8.1`, `Matplotlib 3.5.3` and `SymPy 1.12`. For DFT, we used the `fftpack` module of `SciPy`. For the more advanced waveform model, we also computed convolutions. Once a waveform model was obtained in `Mathematica`, we first exported the resulting expression to `C` using the `CForm[ ]` command and then transformed it to a `Python` library using `SWIG`, a software development tool that links programs written in `C` and `C++` to a variety of high-level programming languages. Certain special functions (Erfi(z)) were included in the `C` code using the `Faddeeva` package. The resulting workflow is shown schematically in Fig. 2.1.

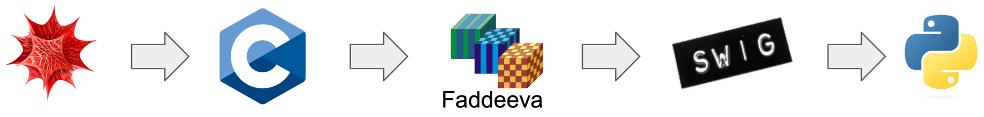

Figure 2.1: Workflow for transforming an analytic expression obtained initially in `Mathematica` to a `Python` library.

Analytic Waveform Models in the Frequency Domain

Below, we discuss several waveform models, starting from the simplest case of a single dampled oscillator, continuing with a sum of damped oscillators and arriving at the more advanced model that includes a time-evolving dominant frequency.

Single damped oscillator without window function

Our simplest example is a single damped oscillator, which is defined as follows in the time domain:

$$h(t) = A_1 e^{-\left(t-t_{\mathrm{merger}}\right)/\tau_1} \sin\left(2\pi f_1 (t - t_{\mathrm{merger}}) + \phi_1\right). \tag{2.7}$$

The continuous Fourier transform in `Mathematica` is found with the following command:

$$\begin{aligned}&\texttt{FourierTransform}[\mathrm{A}_1 * \texttt{Exp}[-(t-\mathrm{t}_{\mathrm{merger}})/\tau_1] * \texttt{Sin}[2 * \texttt{Pi} * \mathrm{f}_1 * (t - \mathrm{t}_{\mathrm{merger}}) + \phi_1],\\ &\qquad t,\ 2 * \texttt{Pi} * \mathrm{f},\ \texttt{FourierParameters}- > \{1,-1\}],\end{aligned} \tag{2.8}$$

with the result

$$\begin{aligned}\tilde{h}(f) \quad = \quad & i\pi A_1 \left[e^{2i\pi t_{\mathrm{merger}} f_1 + t_{\mathrm{merger}}/\tau_1 - i\phi_1} \delta\left(2f\pi + 2\pi f_1 - \frac{i}{\tau_1}\right)\right.\\ & \left. - e^{-2i\pi t_{\mathrm{merger}} f_1 + t_{\mathrm{merger}}/\tau_1 + i\phi_1} \delta\left(-2f\pi + 2\pi f_1 + \frac{i}{\tau_1}\right)\right].\end{aligned} \tag{2.9}$$

The result of this transform contains Dirac Delta functions, $\delta(z)$ with *complex* arguments. Note that `Mathematica` does not produce a valid numerical computation for $\delta(z)$, even in a simple scenario, such as evaluating $\delta(1+i)$.

When computing the distinct Fourier transforms of the functions

$$h_1(t) = A_1 e^{-\left(t-t_{\mathrm{merger}}\right)/\tau_1}, \tag{2.10}$$

and

$$h_2(t) = \sin\left(2\pi f_1\left(t - t_{\text{merger}}\right) + \phi_1\right) \tag{2.11}$$

to perform a convolution, the same problem appears, since the Fourier transform of Eq. (2.10) is

$$\tilde{h}_1(f) \;=\; 2A_1 e^{t_{\text{merger}}/\tau_1}\pi\delta(-2f\pi + i/\tau_1) \tag{2.12}$$

which again contains the Dirac delta function with a complex argument. The problem originates from the fact that the time-domain model of Eq. (2.7) does not represent a finite duration of the signal, i.e. it goes back to an infinite amplitude as $t \to -\infty\}$, which results in the δ functions when it is transformed to the frequency domain. Therefore, although this model can be used in applications in the time domain, it is not appropriate for applications in the frequency domain.

Single damped oscillator with window function

We resolved the problem identified in the previous section, by multiplying the time-domain model with an appropriate window function, to effectively enforce a finite duration D of the signal. To achieve this, we use the *rectangular window* function $\Pi[t/D]$, but with a starting time $t = t_{\text{merger}}$ and end time $t = t_{\text{merger}} + D$, i.e.

$$\Pi[(t - t_{\text{merger}})/D] = \begin{cases} 1 & \text{for } t_{\text{merger}} \le t \le t_{\text{merger}} + D \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

We construct this window function in `Mathematica` as the difference between two *Heaviside step functions.* The latter is defined as

$$\theta(t) = \begin{cases} 1 & \text{for } t \ge 0 \\ 0 & \text{for } t < 0. \end{cases} \tag{2.14}$$

Hence, we write

$$\Pi[(t - t_{\text{merger}})/D] = \theta(t - t_{\text{merger}}) - \theta(t - t_{\text{merger}} - D), \tag{2.15}$$

and multiply the time-domain model of Eq. (2.7) by this window function, resulting in a time-domain model with a finite duration:

$$h_D(t) = \Pi[(t - t_{\text{merger}})/D] \cdot A_1 e^{-\left(t - t_{\text{merger}}\right)/\tau_1} \sin\left(2\pi f_1(t - t_{\text{merger}}) + \phi_1\right). \tag{2.16}$$

In Fig. 2.2 we show a particular example of a single damped oscillator, without the window function (black line) and with the window function (blue line). We obtained the analytic Fourier transform of Eq. (2.16) using `Mathematica`,

$$\begin{aligned}&\texttt{FourierTransform}[\mathrm{A}_1 * \Pi[t\_] * \texttt{Exp}[-(t - \mathrm{t}_{\text{merger}})/\tau_1] * \texttt{Sin}[2 * \texttt{Pi} * \mathrm{f}_1 * (t - \mathrm{t}_{\text{merger}}) + \phi_1],\\ &\quad t,\ 2 * \texttt{Pi} * \mathrm{f},\ \texttt{FourierParameters}- > \{1, -1\}].\end{aligned} \tag{2.17}$$

However, the result still contained δ functions with complex arguments:

$$
\begin{aligned}
\tilde{h}_D(f) \;=\;& \frac{A_1}{4}\left[\frac{2i\tau_1}{i-2\pi(f+f_1)\tau_1}+2\pi\delta\left(2\pi(f+f_1)-i/\tau_1\right)\right] ie^{-i(\phi_1+2f\pi t_{\rm merger})} \\
&+\frac{A_1}{4}e^{-D/\tau_1}\left[\frac{2i\tau_1}{i-2\pi(f-f_1)\tau_1}+2\pi\delta\left(2\pi(f+f_1)-i/\tau_1\right)\right] ie^{i(\phi_1-2\pi D(f-f_1)-2\pi f t_{\rm merger})} \\
&+\frac{A_1}{4}e^{-D/\tau_1}\left[\frac{2i\tau_1}{i-2\pi(f+f_1)\tau_1}+2\pi\delta\left(2\pi(f+f_1)-i/\tau_1\right)\right] ie^{-i(\phi_1+2\pi D(f+f_1)+2f\pi t_{\rm merger})}
\end{aligned}
\tag{2.18}
$$

We managed to obtain a different expression, that does not involve δ functions, deriving the Fourier transform of Eq. (2.16) using convolution of the Fourier transform of its individual terms. Specifically, we split Eq. (2.16) into the product of two terms:

$$
h_D(t) = h_1(t)\cdot h_2(t), \tag{2.19}
$$

where

$$
h_1(t) = \Pi[(t-t_{\rm merger})/D]\cdot A_1 e^{-\left(t-t_{\rm merger}\right)/\tau_1}, \tag{2.20}
$$

and

$$
h_2(t) = \sin\left(2\pi f_1(t-t_{\rm merger})+\phi_1\right). \tag{2.21}
$$

We find the individual analytic Fourier transforms as

$$
\begin{aligned}
\tilde{h}_1(f) \;=\;& \frac{A_1\tau_1(1-i2\pi f\tau_1)}{1+4\pi^2 f^2\tau_1^2}\left(e^{i2\pi D f}-e^{-D/\tau_1}\right)e^{-i2\pi f(D+t_{\rm merger})} \\
&-e^{(1+i2\pi f\tau_1)(D+t_{\rm merger})/\tau_1}\;\theta(-D-t_{\rm merger}) \\
&+e^{(1+i2\pi f\tau_1)(D+t_{\rm merger})/\tau_1}\;\theta(-t_{\rm merger}),
\end{aligned}
\tag{2.22}
$$

and

$$
\tilde{h}_2(f) \;=\; -\frac{1}{2}ie^{-i(2\pi f_1 t_{\rm merger}-\phi_1)}\delta(-f+f_1)+\frac{1}{2}ie^{i(2\pi f_1 t_{\rm merger}-\phi_1)}\delta(f+f_1). \tag{2.23}
$$

Then, the Fourier transform of $h_D(t)$ is the convolution of $\tilde{h}_1(f)$ and $\tilde{h}_2(f)$. That is

$$
\tilde{h}_D(f) = \int_{-\infty}^{+\infty}\tilde{h}_1(f')\tilde{h}_2(f-f')df'. \tag{2.24}
$$

The result is

$$\begin{aligned}
\tilde{h}_D(f) \;=\; & \frac{1}{2} A_1 e^{-i\phi_1} \tau_1 \Big\{ e^{-D[2i\pi(f+f_1)+1/\tau_1]-2if\pi t_{\text{merger}}} \Big[[i - 2\pi(f - f_1)\tau_1] \\
& + e^{2i\phi_1 + D[2i(f+f_1)\pi + 1/\tau_1]} [i - 2\pi(f + f_1)\tau_1] \\
& + e^{D[2i(f+f_1)\pi + 1/\tau_1]} [-i + 2\pi(f - f_1)\tau_1] \\
& + e^{2i(\phi_1 + 2Df_1\pi)} [-i + 2\pi(f + f_1)\tau_1] \Big] \Big\} / \\
& [-i + 2\pi(f - f_1)\tau_1][-i + 2\pi(f + f_1)\tau_1] \\
& \\
& + i \left[\frac{e^{2i\phi_1 - 2if_1\pi t_{\text{merger}} + t_{\text{merger}}/\tau_1}}{1 + 4\pi^2\tau_1^2 (f - f_1)^2} - \frac{e^{(2if_1\pi + 1/\tau_1)t_{\text{merger}}}}{1 + 4\pi^2\tau_1^2 (f + f_1)^2} \right] \theta(-D - t_{\text{merger}}) \\
& + \left[- \frac{ie^{2i\phi_1 - 2if_1\pi t_{\text{merger}} + t_{\text{merger}}/\tau_1}}{1 + 4\pi^2\tau_1^2 (f - f_1)^2}) + \frac{ie^{(2if_1\pi + 1/\tau_1)t_{\text{merger}}}}{1 + 4\pi^2\tau_1^2 (f + f_1)^2} \right] \theta(-t_{\text{merger}}) \qquad (2.25)
\end{aligned}$$

and it is free from complex arguments in δ functions.

Fig. 2.3 shows the continuous Fourier transform of Eq. (2.25) for the representative case of the single damped oscillator with the orthogonal window function in Fig. 2.2. We compare our result to the corresponding numerical DFT of the time series. We find excellent agreement between our analytic Fourier transform and the numerical DFT of the time series. The small oscillations seen in the analytic Fourier transform and due to the phenomenon of *spectral leakage*, caused by the finite duration D of the orthogonal window function. The numerical DFT coincides with the analytic result at the local minima of these oscillations. To verify that the oscillations are entirely due to the spectral leakage, we repeat the above comparison in Fig. 2.4, but this time for a significantly increased duration of $D = 0.125$. For such large duration, the oscillations due to spectral leakage are no longer visible.

The numerical codes for producing the comparisons in Figs. 2.2, 2.3, and 2.4 are presented in Appendix A.

Multiple damped oscillators with window function

As a second example, we consider the sum of four damped oscillators, which is a more realistic description of the post-merger GW emission in binary neutron star mergers. In this case, the analytic model in the time domain is

$$\begin{aligned}
h(t) \;=\; & \Pi[(t - t_{\text{merger}})/D] \cdot \\
& \Big[A_1 e^{-(t - t_{merger})/\tau_1} \sin\left(2\pi f_1 (t - t_{\text{merger}}) + \phi_1\right) \\
& + A_2 e^{-(t - t_{merger})/\tau_2} \sin\left(2\pi f_2 (t - t_{\text{merger}}) + \phi_2\right) \\
& + A_3 e^{-(t - t_{merger})/\tau_3} \sin\left(2\pi f_3 (t - t_{\text{merger}}) + \phi_3\right) \\
& + A_4 e^{-(t - t_{merger})/\tau_4} \sin\left(2\pi f_4 (t - t_{\text{merger}}) + \phi_4\right) \Big] . \qquad (2.26)
\end{aligned}$$

A fundamental property of Fourier analysis is that the Fourier transform of a sum of functions is equivalent to the sum of the Fourier transforms of those functions. Thus, in the case of a four-oscillator sum in Eq. (2.26), we can compute the Fourier transform of each oscillator and then sum them up. As previously demonstrated, trying to compute the Fourier transform for the single damped oscillator of Eq. (2.7) without multiplying with a window function

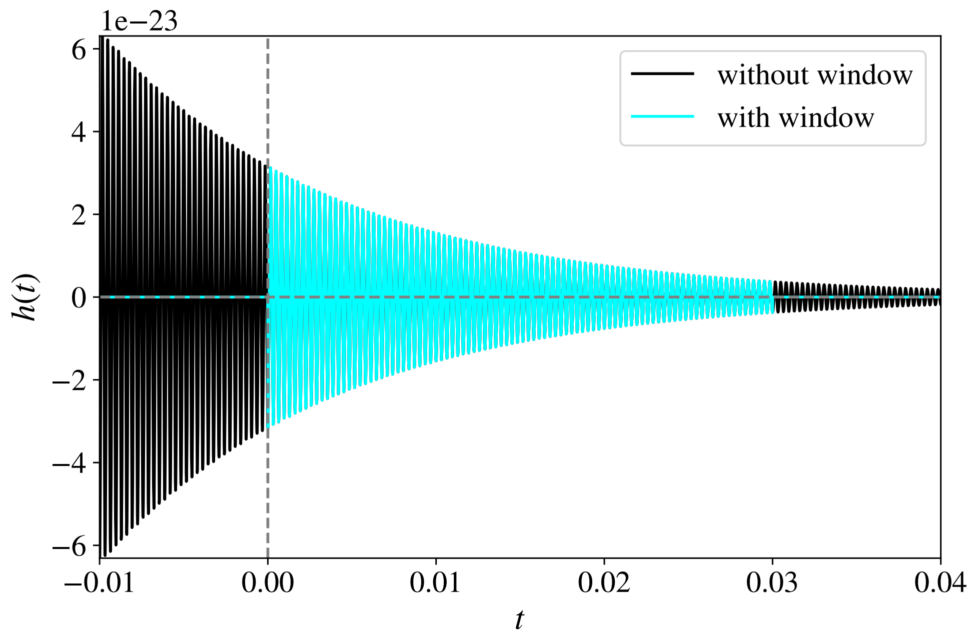

Figure 2.2: Representative example of a single dampled oscillator in the time domain, without a window function (black line) and with the rectangular window function (blue line). The values used are $D = 0.03$, $t_{\text{merger}} = 0$, $A_1 = 10^{-22.5}, \tau_1 = 2.5 \times 10^{-2.25}$, $\phi_1 = -\pi/2$, $f_1 = 3100$ (dimensionless units).

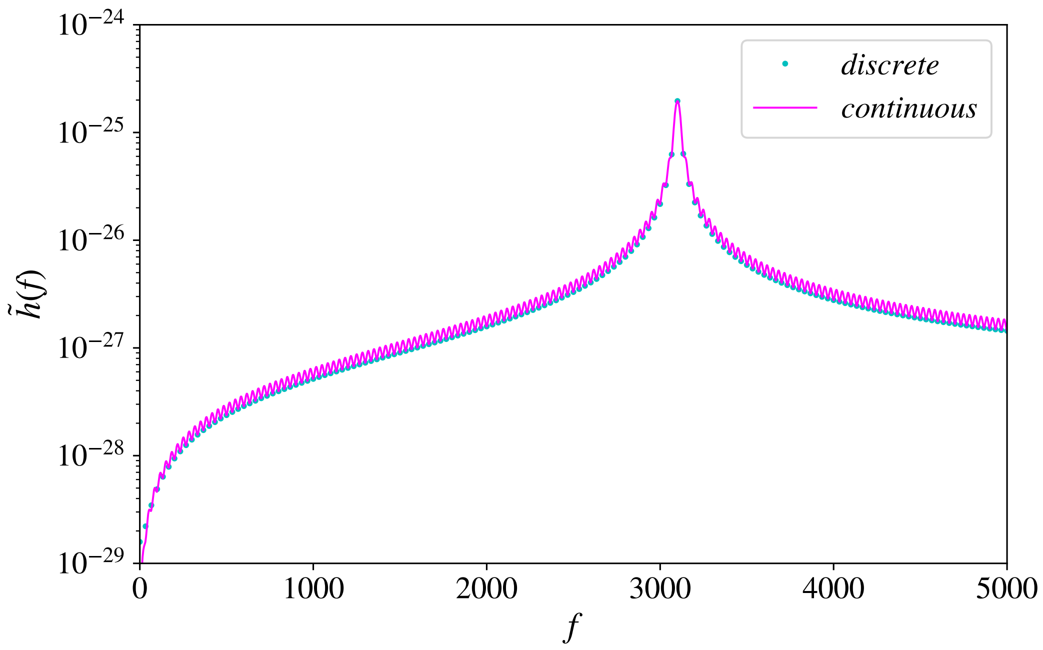

Figure 2.3: Comparison of the analytic continuous Fourier transform to the numerical DFT for the particular model of a single damped oscillator with a rectangular window funtion shown in Fig. 2.2.

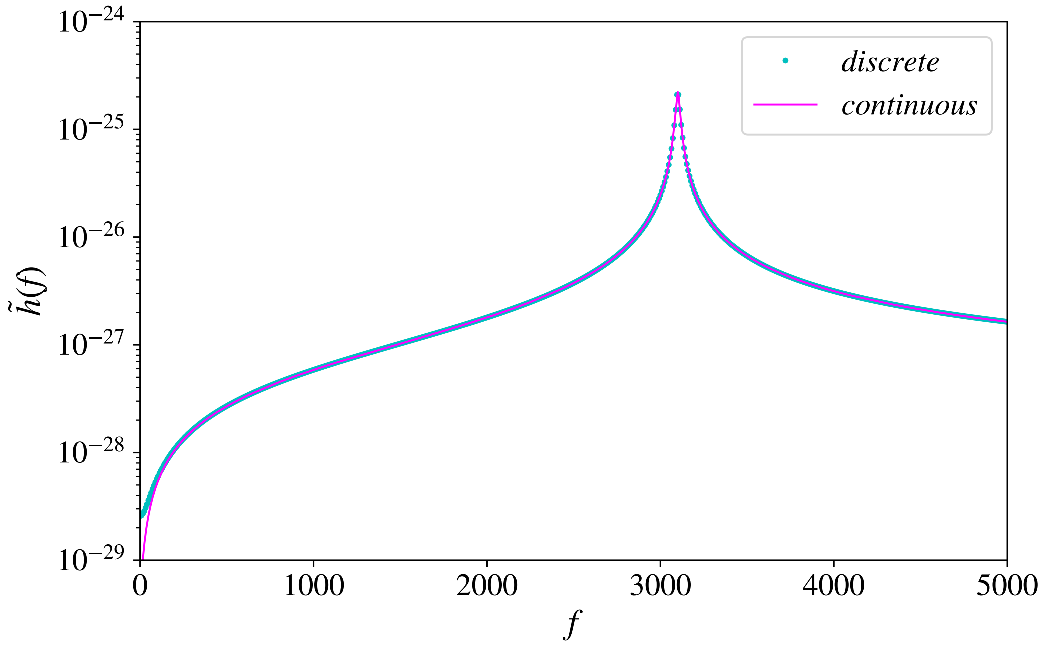

Figure 2.4: Same as Fig. 2.3, but for a significantly longer signal duration.

to enforce a finite duration, becomes impractical due to the presence of δ functions with complex arguments. Therefore, the use of an orthogonal window function is essential and we apply the same one to each of the four oscillators participating in Eq. (2.26).

Following the same process as earlier the sum of the convolution is:

$$
\begin{aligned}
\tilde{h}_D(f) \;=\;& \frac{1}{2}A_1 e^{-i\phi_1}\tau_1 \Big\{ e^{-D[2i\pi(f+f_1)+1/\tau_1]-2if\pi t_{\text{merger}}} \Big[[i-2\pi(f-f_1)\tau_1] \\
&+ e^{2i\phi_1+D[2i(f+f_1)\pi+1/\tau_1]}[i-2\pi(f+f_1)\tau_1] \\
&+ e^{D[2i(f+f_1)\pi+1/\tau_1]}[-i+2\pi(f-f_1)\tau_1] \\
&+ e^{2i(\phi_1+2Df_1\pi)}[-i+2\pi(f+f_1)\tau_1]\Big]\Big\}/ \\
&[-i+2\pi(f-f_1)\tau_1][-i+2\pi(f+f_1)\tau_1] \\
\\
&+ i\left[\frac{e^{2i\phi_1-2if_1\pi t_{\text{merger}}+t_{\text{merger}}/\tau_1}}{1+4\pi^2\tau_1^2(f-f_1)^2} - \frac{e^{(2if_1\pi+1/\tau_1)t_{\text{merger}}}}{1+4\pi^2\tau_1^2(f+f_1)^2}\right]\theta(-D-t_{\text{merger}}) \\
&+ \left[-\frac{ie^{2i\phi_1-2if_1\pi t_{\text{merger}}+t_{\text{merger}}/\tau_1}}{1+4\pi^2\tau_1^2(f-f_1)^2}) + \frac{ie^{(2if_1\pi+1/\tau_1)t_{\text{merger}}}}{1+4\pi^2\tau_1^2(f+f_1)^2}\right]\theta(-t_{\text{merger}}) \\
&+ \frac{1}{2}A_2 e^{-i\phi_2}\tau_2 \Big\{ e^{-D[2i\pi(f+f_2)+1/\tau_2]-2if\pi t_{\text{merger}}} \Big[[i-2\pi(f-f_2)\tau_2] \\
&+ e^{2i\phi_2+D[2i(f+f_2)\pi+1/\tau_2]}[i-2\pi(f+f_2)\tau_2] \\
&+ e^{D[2i(f+f_2)\pi+1/\tau_2]}[-i+2\pi(f-f_2)\tau_2] \\
&+ e^{2i(\phi_2+2Df_2\pi)}[-i+2\pi(f+f_2)\tau_2]\Big]\Big\}/ \\
&[-i+2\pi(f-f_2)\tau_2][-i+2\pi(f+f_2)\tau_2]
\end{aligned}
$$

$$
\begin{aligned}
&+ i\left[\frac{e^{2i\phi_2-2if_2\pi t_{\mathrm{merger}}+t_{\mathrm{merger}}/\tau_2}}{1+4\pi^2\tau_2^2(f-f_2)^2}-\frac{e^{(2if_2\pi+1/\tau_2)t_{\mathrm{merger}}}}{1+4\pi^2\tau_2^2(f+f_2)^2}\right]\theta(-D-t_{\mathrm{merger}})\\
&+\left[-\frac{ie^{2i\phi_2-2if_2\pi t_{\mathrm{merger}}+t_{\mathrm{merger}}/\tau_2}}{1+4\pi^2\tau_2^2(f-f_2)^2}) + \frac{ie^{(2if_2\pi+1/\tau_2)t_{\mathrm{merger}}}}{1+4\pi^2\tau_2^2(f+f_2)^2}\right]\theta(-t_{\mathrm{merger}})\\
&+\frac{1}{2}A_3e^{-i\phi_3}\tau_3\Bigg\{e^{-D[2i\pi(f+f_3)+1/\tau_3]-2if\pi t_{\mathrm{merger}}}\Big[[i-2\pi(f-f_3)\tau_3]\\
&+e^{2i\phi_3+D[2i(f+f_3)\pi+1/\tau_3]}[i-2\pi(f+f_3)\tau_3]\\
&+e^{D[2i(f+f_3)\pi+1/\tau_3]}[-i+2\pi(f-f_3)\tau_3]\\
&+e^{2i(\phi_3+2Df_3\pi)}[-i+2\pi(f+f_3)\tau_3]\Big]\Bigg\}/\\
&[-i+2\pi(f-f_3)\tau_3][-i+2\pi(f+f_3)\tau_3]
\end{aligned}
$$

$$
\begin{aligned}
&+ i\left[\frac{e^{2i\phi_3-2if_3\pi t_{\mathrm{merger}}+t_{\mathrm{merger}}/\tau_3}}{1+4\pi^2\tau_3^2(f-f_3)^2}-\frac{e^{(2if_3\pi+1/\tau_3)t_{\mathrm{merger}}}}{1+4\pi^2\tau_3^2(f+f_3)^2}\right]\theta(-D-t_{\mathrm{merger}})\\
&+\left[-\frac{ie^{2i\phi_3-2if_3\pi t_{\mathrm{merger}}+t_{\mathrm{merger}}/\tau_3}}{1+4\pi^2\tau_3^2(f-f_3)^2}) + \frac{ie^{(2if_3\pi+1/\tau_3)t_{\mathrm{merger}}}}{1+4\pi^2\tau_3^2(f+f_3)^2}\right]\theta(-t_{\mathrm{merger}})\\
&+\frac{1}{2}A_4e^{-i\phi_4}\tau_4\Bigg\{e^{-D[2i\pi(f+f_4)+1/\tau_4]-2if\pi t_{\mathrm{merger}}}\Big[[i-2\pi(f-f_4)\tau_4]\\
&+e^{2i\phi_4+D[2i(f+f_4)\pi+1/\tau_4]}[i-2\pi(f+f_4)\tau_4]\\
&+e^{D[2i(f+f_4)\pi+1/\tau_4]}[-i+2\pi(f-f_4)\tau_4]\\
&+e^{2i(\phi_4+2Df_4\pi)}[-i+2\pi(f+f_4)\tau_4]\Big]\Bigg\}/\\
&[-i+2\pi(f-f_4)\tau_4][-i+2\pi(f+f_4)\tau_4]
\end{aligned}
$$

$$
\begin{aligned}
&+ i\left[\frac{e^{2i\phi_4-2if_4\pi t_{\mathrm{merger}}+t_{\mathrm{merger}}/\tau_4}}{1+4\pi^2\tau_4^2(f-f_4)^2}-\frac{e^{(2if_4\pi+1/\tau_4)t_{\mathrm{merger}}}}{1+4\pi^2\tau_4^2(f+f_4)^2}\right]\theta(-D-t_{\mathrm{merger}})\\
&+\left[-\frac{ie^{2i\phi_4-2if_4\pi t_{\mathrm{merger}}+t_{\mathrm{merger}}/\tau_4}}{1+4\pi^2\tau_4^2(f-f_4)^2}) + \frac{ie^{(2if_4\pi+1/\tau_4)t_{\mathrm{merger}}}}{1+4\pi^2\tau_4^2(f+f_4)^2}\right]\theta(-t_{\mathrm{merger}}) \qquad (2.27)
\end{aligned}
$$

Fig. (2.5) shows a representative example of a waveform in the time domain, described by the sum of four damped oscillators with an orthogonal window function of Eq. (2.26). In Fig. (2.6), we also show the time evolution of the individual oscillators participating in the model, for the same case as in Fig. (2.5). The analytic Fourier transform of this time evolution is displayed in Fig. (2.7) and compared to the corresponding numerical DFT. We find excellent agreement between the analytic and numerical results.

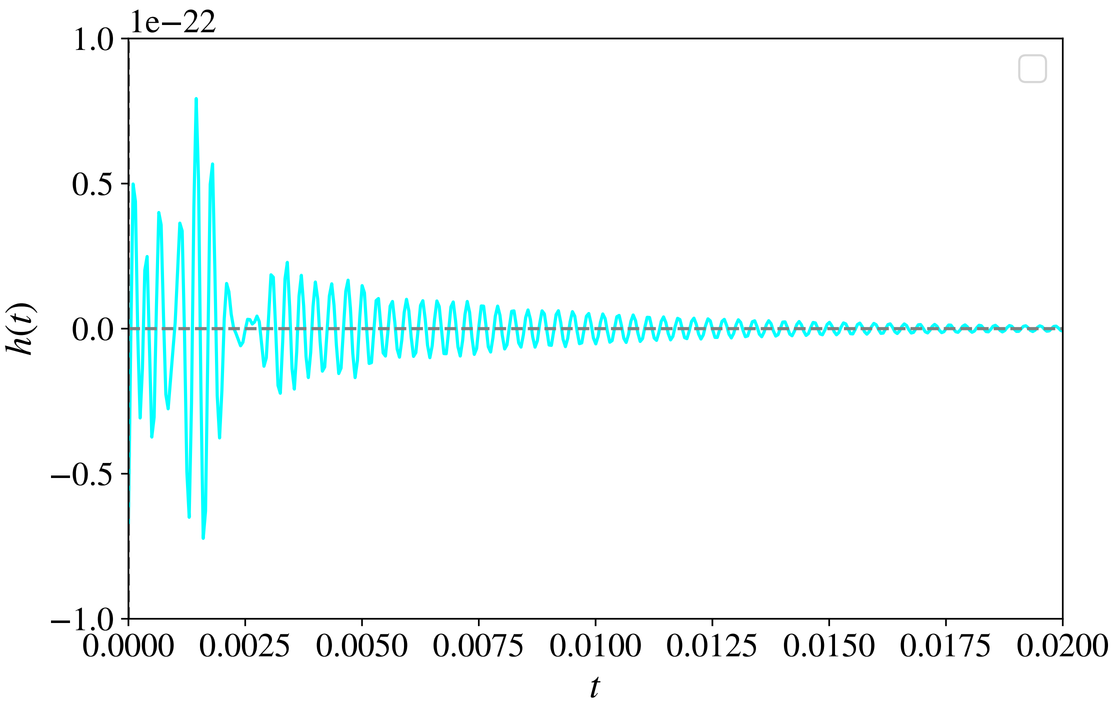

Figure 2.5: A representative case of an analytic post-merger waveform model in the time domain, consisting of the sum of four damped oscillators of Eq. (2.26). The duration is $D = 0.125$, and the merger time is $t_{\rm merger} = 0$. The parameters of the individual oscillators are $A_1 = 10^{-22.5}$, $A_2 = 10^{-22.3}$, $A_3 = 10^{-22.3}$, $A_4 = 10^{-22.3}$, $\tau_1 = 10^{-2.25}$, $\tau_2 = 10^{-2.8}$, $\tau_3 = 10^{-2.8}$, $\tau_4 = 10^{-2.8}$, $\phi_1 = -\pi/2$, $\phi_2 = \pi/2$, $\phi_3 = -\pi/2$, $\phi_4 = -\pi/2$, $f_1 = 3100$, $f_2 = 2750$, $f_3 = 2460$, $f_1 = 3640$, (dimensionless units).

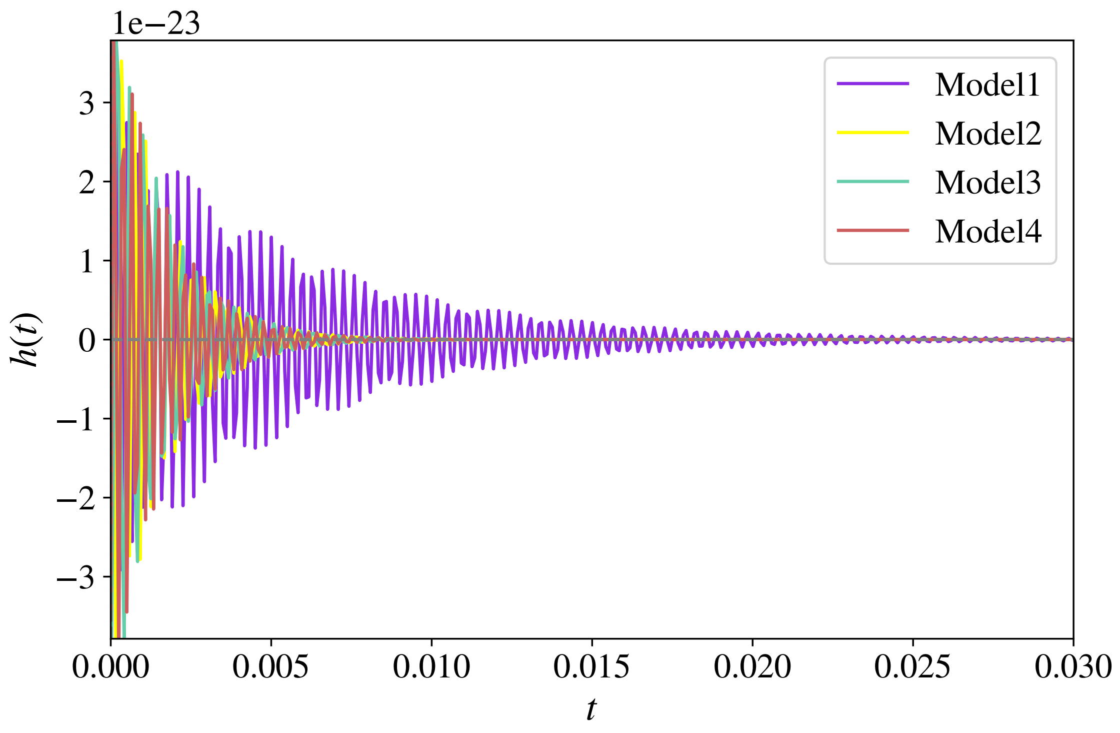

Figure 2.6: Same as Fig. 2.5, but only the individual oscillators (Model1 - Model4) participating in the sum of Eq. (2.26) are shown.

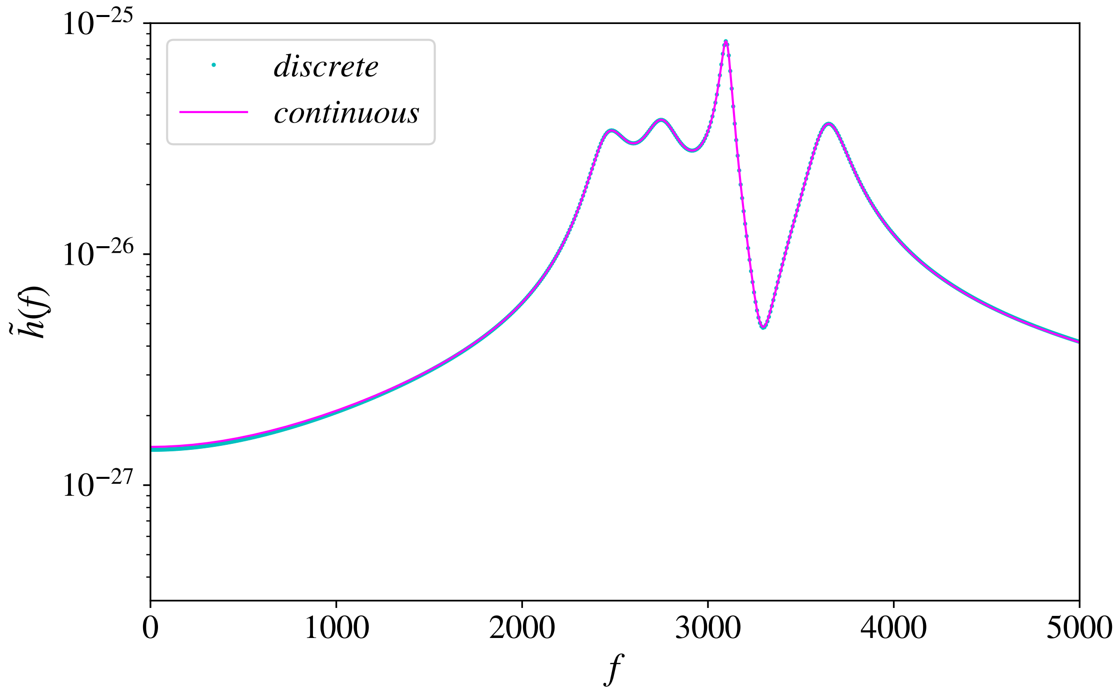

Figure 2.7: Comparison of the analytic continuous Fourier transform and the numerical DFT for the particular model comprising four damped oscillators with a rectangular window function shown in Figs. 2.5 and 2.6. An excellent agreement is observed.

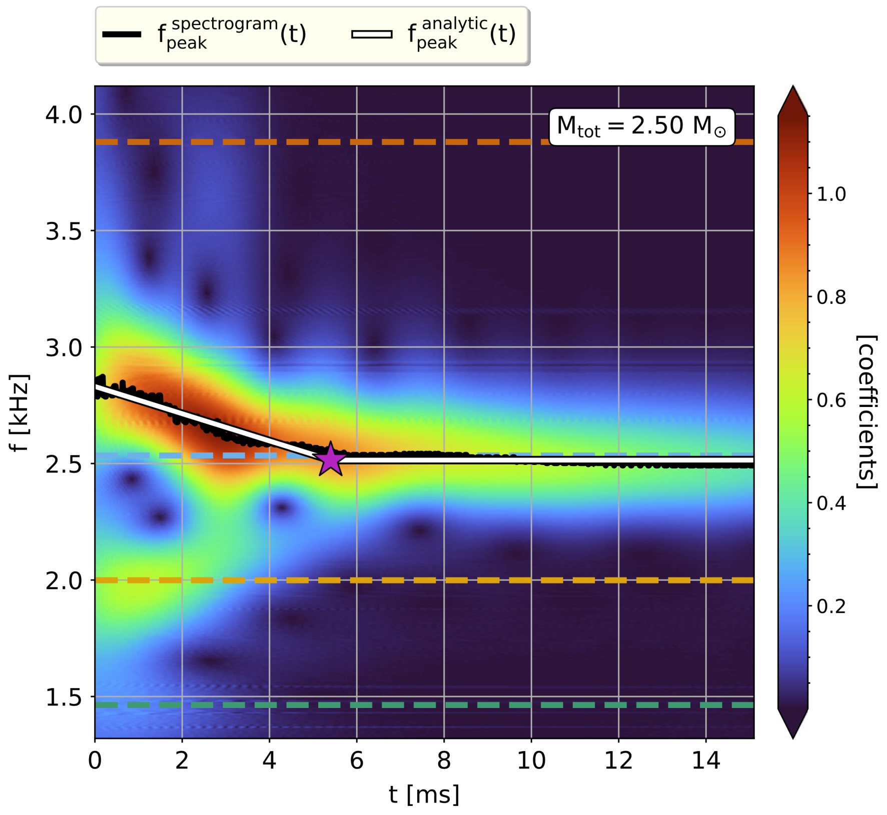

Figure 2.8: A spectrogram of the post-merger gravitational wave emission from a particular binary neutron star merger simulation is shown. The dominant oscillation frequency, f_{peak}, changes over the first 5ms after merger and then remains constant. This time evolution can be modelled with a piecewise analytic fit (as indicated by the lines). The purple star marks the point at which the frequency becomes constant. The dashed lines represent other post-merger frequencies. Figure from [25].

Multiple damped oscillators with time-dependent f_{peak} and window function

In [25] an analytical time domain model was introduced for the GW emission in the post-merger phase of a binary neutron star system that includes four damped oscillators, but with a time-dependent frequency for the dominant peak. The motivation for introducing the time dependence in the dominant oscillator was the observation that in many simulations the dominant frequency f_{peak} was observed to have a strong time dependence for a few milliseconds after merger. A particular example is shown in the spectrogram of Fig. 2.8, which focuses on the time evolution of f_{peak}. In [25], this particular time dependence was modeled as a piecewise function consisting of two linear pieces. The first piece describes a linearly varying frequency from t_{merger} up to a time t_*, whereas the second piece corresponds to a constant frequency.

The model introduced in [25] can be written as:

$$\begin{aligned} h(t) \quad = \quad & A_1 e^{-(t-t_{\mathrm{merger}})/\tau_1} \sin\left[2\pi f_1 (t - t_{\mathrm{merger}}) + \phi_1\right] \\ & + A_2 e^{-(t-t_{\mathrm{merger}})/\tau_2} \sin\left[2\pi f_2 (t - t_{\mathrm{merger}}) + \phi_2\right] \\ & + A_3 e^{-(t-t_{\mathrm{merger}})/\tau_3} \sin\left[2\pi f_3 (t - t_{\mathrm{merger}}) + \phi_3\right] \end{aligned}$$

$$+ A_{\text{peak}}(t), \tag{2.28}$$

where

$$A_{\text{peak}}(t) = \begin{cases} A_{\text{peak}} e^{-(t-t_{\text{merger}})/\tau_{\text{peak}}} \sin\left[2\pi\left(f_{\text{peak},0} + \zeta_{\text{drift}} \frac{(t-t_{\text{merger}})}{2}\right)\right] & \\ \qquad +(t - t_{\text{merger}}) + \phi_{\text{peak},0} & \text{if } t \leq t_* \\ A_{\text{peak}}\ e^{-(t-t_{\text{merger}})/\tau_{\text{peak}}} \sin\left[2\pi f_{\text{peak}}(t_*)(t - t_*)\right] + \phi_{\text{peak}}(t_*) & \text{if } t > t_* \end{cases} \tag{2.29}$$

and

$$f_{\text{peak}}(t_*) = f_{\text{peak},0} + \zeta_{\text{drift}} \frac{(t_* - t_{\text{merger}})}{2},$$

$$\phi_{\text{peak}}(t_*) = 2\pi\left(f_{\text{peak},0} + \zeta_{\text{drift}} \frac{(t_* - t_{\text{merger}})}{2}\right)(t_* - t_{\text{merger}}) + \phi_{\text{peak},0},$$

$t = t_*$ is the time after which the frequency remains constant,

$t = t_{\text{merger}}$ is the time at which $|h(t)| = \sqrt{h_+^2(t) + h_\times^2(t)}$ reaches maximum.

The time domain model can be transformed into a frequency domain model if it is first multiplied with an orthogonal unit window function. In this case, we choose a window function with a total width of T that extends from $-T/2$ to $T/2$ and then obtain the continuous Fourier transform with `Mathematica`, as described in the previous sections. However, this example has three oscillators in the form [2.16] and a fourth one that is time dependent. Once again, the Fourier transform of the sum is the sum of the Fourier transforms. For the first three oscillators, we use a Fourier transform corresponding to Eq. (2.25), but for the new orthogonal window function.

We define the fourth oscillator in `Mathematica` as a piecewise function multiplied by an orthogonal window function

$$\begin{aligned} h_{\text{peak}}(t) \;=\; & (\texttt{HeavisideTheta}\,[t + T/2] - \texttt{HeavisideTheta}\,[t - T/2]) * \\ & \texttt{Piecewise}[\{\{A_{\text{peak}} * \texttt{Exp}\left[-(t - t_{\text{merger}})/\tau_{\text{peak}}\right] * \\ & \texttt{Sin}\Big[2 * \texttt{Pi} * \left(f_{peak,0} + \zeta_{drift} * \left(t - t_{\text{merger}}\right)/2\right) \\ & *(t - t_{\text{merger}}) + \phi_{\text{peak},0}\Big]\ ,\ t < t_*, \\ & \{A_{\text{peak}} * \texttt{Exp}\left[-(t - t_{\text{merger}})/\tau_{\text{peak}}\right] * \\ & \texttt{Sin}\Big[2 * \texttt{Pi} * (f_{\text{peak},0} + \zeta_{\text{drift}} * (t_* - t_{\text{merger}})/2) \\ & *(t - t_{\text{merger}}) + \phi_{\text{peak},0}\Big]\ \ t >= t_*\},\ \ \{0, \texttt{True}\}\}]. \end{aligned} \tag{2.30}$$

The continuous transform of [2.30], $\tilde{h}_{\text{peak}}(f)$, is calculated in `Mathematica`, resulting in a very large expression (even you using the `Simplify` command, and hence we do not display the result here.

After the continuous Fourier transforms for each oscillator were computed individually, we found the transform of the complete model of Eq. (2.28), after it is multiplied with the orthogonal window funtion, by adding the individual contributions:

$$\tilde{h}(f) = \sum_i^3 \tilde{h}_i(f) + \tilde{h}_{peak}(f). \tag{2.31}$$

In Fig. (2.9) we show a particular example of the complete model of Eq. (2.28), with time-evolving $f_{\rm peak}$ in the time domain (we used dimensionless units, with the convention of $1{\rm ms} = 1$). The corresponding continuous Fourier transform that we obtained for this model is shown in Fig. (2.10) and we compare it to the numerical DFT of the same time series. Apart from the expected spectral leakage for such a short-duration signal, we find excellent agreement between the continuous Fourier transform and the numerical DFT, confirming the validity of our obtained analytic expression. To confirm that that the small oscillations present in the continuous transform are only due to spectral leakage, we show a magnification of Fig. (2.10) in the frequency range $0 \le f \le 1$ in Fig. (2.11). The discrete values of the numerical DFT coincide with the minima of the spectral-leakage-induced oscillation in the continuous transform, as expected.

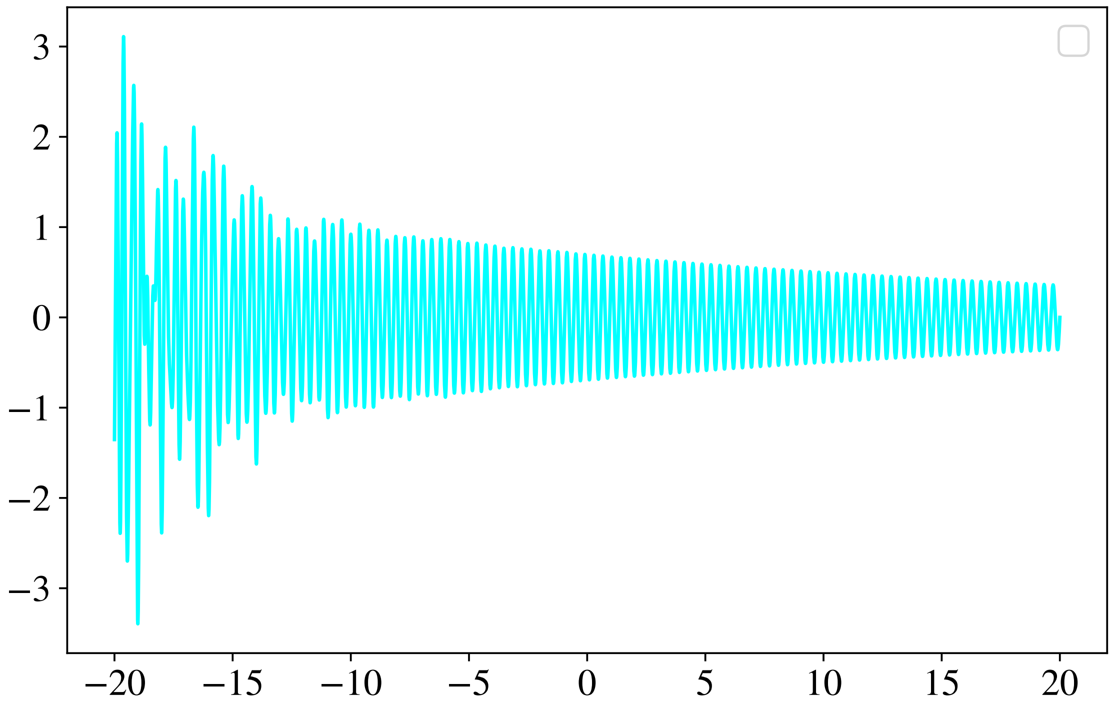

Figure 2.9: Time evolution of model [2.28] for $A_{\rm peak} = 10/\exp(2)$, $t_* = -15, f_{\rm peak,0} = 3$, $t_{\rm merger} = -20$, $\phi_{peak,0} = 0$, $\tau_{\rm peak} = 30$, $\zeta_{\rm drift} = -0.15$, $T = 40$, $f_1 = 2.4$, $f_2 = 2$, $f_3 = 4$, $\tau_1 = 3$, $\tau_2 = 3$, $\tau_3 = 3$, $A_1 = 10/\exp(2)$, $A_2 = 10/\exp(2)$, $A_3 = 10/\exp(2)$, $\phi_1 = \pi/2$, $\phi_2 = -\pi/2$, $\phi_3 = -\pi/2$

Conversion to Python library

In order to make full use of the continuous Fourier transforms we obtained in this worked, we converted their expressions to `Python` libraries. This will enable their usage in different applications, such as Bayesian parameter estimation codes, which are written in this programming language. To achieve this conversion, we had to solve several issues. Below, we give explicit details of how this conversion was achieved.

As a first step, we exported the final `Mathematica` expression to `C` using the built-in `CForm[ ]` command. However, we found that this built-in conversion is not free of issues, which we had to resolve manually. Specifically, we had to replace

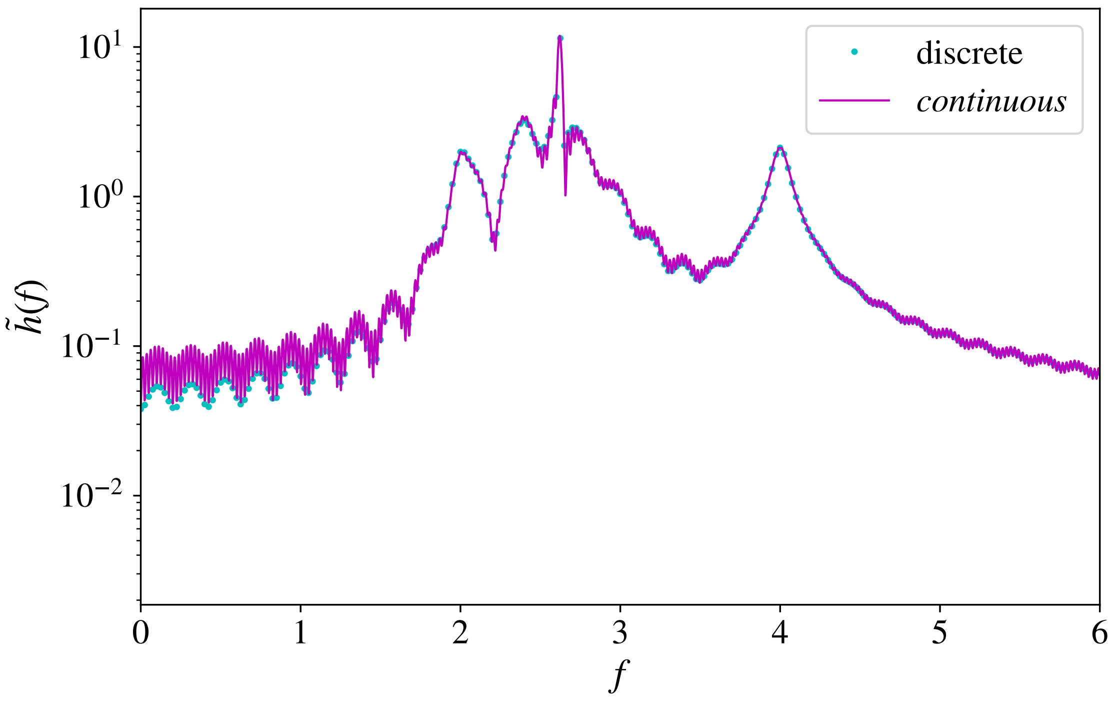

Figure 2.10: Comparison of the analytic continuous Fourier transform and the numerical DFT for the particular model comprising four damped oscillators, with a time-dependent dominant peak and with a rectangular window function shown in Fig. 2.9. An excellent agreement is observed.

- `Power`(`E`, ...) with `cexp`(),
- `Complex`(`0`, −`1`) with −`I`,
- `Complex`(`0`, `2`) with `2I`,
- `Pi` with `M_PI`.

In the C code, it was also essential to manually create and include necessary header files, declare variables with their respective data types, and define functions, for example those that were defined in the original Mathematica code, but were not defined in the C code. For example, the `HeavisideTheta` step function had to be created within the `C` code. The specific `C` code for the single oscillator is shown in Appendix A2.

The next step was to convert the to `C` code `Python`. This involved generating an interface (a text file with an `.i` extension), containing a list of `C/C++` functions, along with directives and declarations that were created to specify how the code should be adapted and made accessible within Python. The extension file includes information about functions, classes, data types, and other elements to be exposed in Python. In this process, `SWIG` (Simplified Wrapper and Interface Generator) read and interpreted the interface file to generate the necessary wrapper code, allowing Python and C to communicate without any issues.

The transition from C to Python also involved the following steps: Initially, the `C` source code was compiled into an object file with the `GCC` compiler. To maximize the performance of the code, we employed the `-O3` compiler flag. This flag activates a high level of optimization. The interface acts as a connection between the `C` and `Python` environments, allowing us to

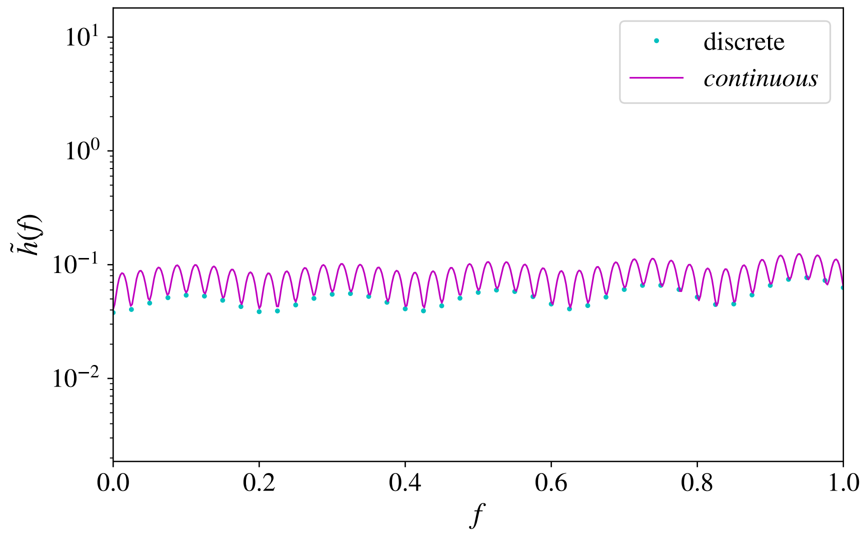

Figure 2.11: A magnification of Fig. (2.10) in the frequency range $0 \leq f \leq 1$, demonstrating that the observed oscillations in the continuous Fourier transform are due to spectral leakage.

define how `C` functions and data types could be accessed and used in Python. Following this, `SWIG` was employed with the `-python` option to generate Python bindings for the functions defined in the interface. This produced a `Python` wrapper file that serves as a connection between `Python` and the `C` code. Both the original `C` source code and the newly created wrapper code were then compiled into object files using `GCC`, using the `-O3` and `-fPIC` flags, which produced a position-independent code. Here we also specified the include directory for `Python` headers. Finally, these object files were linked together to form a shared library, using the `-O3` and `-shared` flag, which creates a shared library with a `.so` extension. The shared library includes the compiled `C` code and `SWIG`-generated wrapper code and enables `Python` to interact with the `C` code.

The discrete Fourier transform was computed directly in `Python`, using the `fftpack` module. The code is given in Appendix C5.

Having completed both transformations, our next step was to import the required libraries, including the `.so` library for the continuous transform into a `Python` code.

In the model [2.28], the steps followed were in general the same. In this case, there is another function within the result called the imaginary error function, written as Erfi. The error function, denoted as Erf, is a well-known mathematical function used to describe the cumulative distribution of a Gaussian or normal distribution. Erfi is a extension of Erf for compex values. For $z \in \mathbb{C}$

$$\mathrm{Erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\, dt, \tag{2.32}$$

and

$$\mathrm{Erfi}(z) = -i\mathrm{Erf}(iz) = \frac{2}{\sqrt{\pi}} = \int_0^z e^{t^2}\, dt. \tag{2.33}$$

Erfi(z) is recognized and readily available in software environments such as `Mathematica` and Python's `SciPy` library. Although Erfi(z) is supported both in `Mathematica` and in `Python`, it is not a built-in function in `C`. Therefore, when working with mathematical algorithms that involve Erfi(z), appropriate implementations or external libraries are used to facilitate the transition from `C` to `Python`. In this work, we have employed the `Faddeeva` package, a free/open-source `C++` code with wrappers for several languages, including `C` and `Python`. The `Faddeeva` package, developed by Steven G. Johnson, provides a `C` code to calculate various error functions of arbitrary complex arguments, including the Erfi(z) function. After downloading the `Faddeeva.c`, `Faddeeva.cc` codes and their corresponding header files, `Faddeeva.h` and `Faddeeva.hh`, we compiled `Faddeeva.c` and our code with the `GCC` compiler and `Faddeeva.cc` with the `g++` compiler. Then, according to the steps described above, we generated a `Python wrapper` with an interface and the `SWIG` command and then we combined all the object files that were produced into a shared library file with a `.so` extension. This shared library could now be used to link and access Python functions. The `Faddeeva.h` header file is also included in the interface and the `C` code, and an Erfi(z) function was constructed in `C`, based on the `Faddeeva_erfi` complex function of `Faddeeva`.

Nonlinear Least-Squares Fit

The `LMFIT` package offers simple Python tools that help create complex fitting models for nonlinear least-squares problems and use these models to analyze actual data. `LMFIT` offers a number of useful features such as: handling parameter objects instead of simple floats as variables, easy changing of fitting algorithms and improved estimation of confidence intervals and also many built-in models for common line shapes that are ready-to-use.

In our case, we used the `ExpressionModel` class of `LMFIT`. It is the primary tool for defining custom models using mathematical expressions and performing curve fitting so that we can create and fit models to our data. In order to do a non-linear least-squares fit of a model to data, the main task is to write an objective function that takes the values of the fitting variables and calculates either values that are to be minimized. Practically, this is achieved by using the `fit()` function on the analytical expression of our oscillators.

We first imported the `ExpressionModel` class from the `LMFIT` library into Python. Then, we inserted our analytic expression, Eq. (2.25), of the continuous Fourier transform for the case of a single damped oscillator with an orthogonal window function. We added some random noise, to simulate real data, and performed a discrete Fourier transform. This represented the target data $y(x)$ (retaining only positive frequencies), which we then tried to fit, using the absolute value of the analytic expression (we were fitting for the magnitude of the Fourier transform). Then we used the `ExpressionModel` class on the absolute value of our analytic expression, converted to string type, as required by `LMFIT`. Lastly, we applied the fit, starting from appropriate initial values.

In Fig. 2.12, we show the fit obtain for our analytic model of Eq. (2.25) to the data generated with the same model, but with added noise. The fit is faithful to the data, achieving a high $R^2 = 0.962$. It also correctly describes the noisy part of the spectrum. In addition, we performed a second fit, but this time using a different analytic expression, that of a Lorentzian, which has been used in other works (see, e.g. [19]). The explicit form of the Lorentzian (which is the Fourier transform of a damped oscillator) is

$$\tilde{h}(f) = \frac{c_0 c_2}{\sqrt{(f-c_1)^2 + c_2^2}} e^{-i \arctan\left(\frac{f-c_1}{c_2}\right)}, \tag{2.34}$$

where c_0 corresponds to the maximum value, c_1 to the main emission frequency, and c_2 to the inverse of the damping time, which sets the Lorentzian's width. With the Lorentzian, we find a smaller $R^2 = 0.913$ and large deviations in the noisy parts of the spectrum.

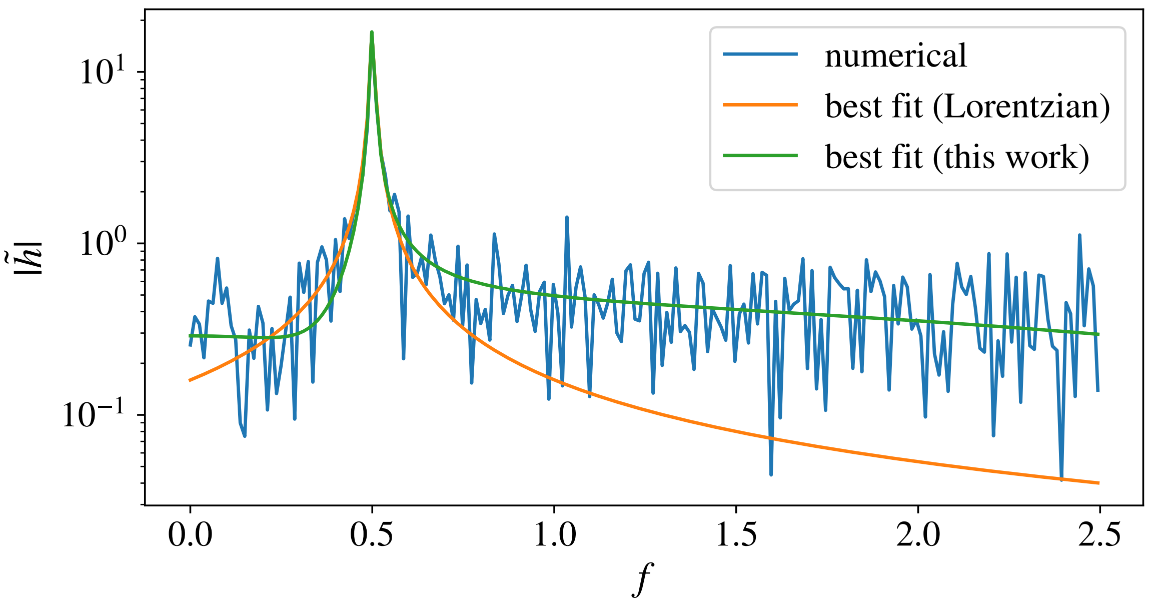

Figure 2.12: A representative case of nonlinear least-squares fitting of our analytical Fourier transform and a simpler Lorentzian expression to data generated for a single damped oscillator with values $A_{\text{noise}} = 0.02$, $t_{\text{merger}} = 0$, $D = 80$, $A = 1$, $f_0 = 0.5$, $\tau = 40$, $\phi_1 = -3\pi/2$ (dimensionless units). The Lorentzian achieves $R^2 = 0.913$, whereas our expression reaches $R^2 = 0.962$.

This comparison demonstrates, that our new, analytic expression of Eq. (2.25) describes the data better than the simpler Lorentzian of Eq. (2.34). This is because the Lorentzian has no information on the actual duration of the signal. In contrast, our new expression of Eq. (2.25) has been constructed explicitly for a signal of specific duration D. Moreover, in the case of multiple oscillators, these can be combined in a linear sum having arbitrary phases. However, in contrast to our Eq. (2.25), the Lorentzian of (2.34) does not model the phase of the signal, which does not allow it to model faithfully multiple oscillators combined with arbitrary phases. Hence, our new expression is more versatile and accurate, and we expect that it will find application in realistic detection strategies and parameter estimation algorithms for the post-merger phase of binary neutron star mergers.

Conclusions

Gravitational-wave astronomy is a rapidly expanding field, that has the potential to significantly broaden our understanding of the cosmos. It is expected that binary neutron star mergers will produce gravitational waves in their post-merger phase. If these waves can be detected, it will give us the opportunity to determine the equation of state of high-density matter at different temperatures. In order to be successful in detecting gravitational waves for such sources, highly precise analytical models must be created that describe the gravitational wave emission of the post-merger remnants. Currently, only basic templates are available in the frequency domain. In this thesis, we calculated analytic templates in the frequency domain that precisely match their time-domain counterparts.

We started deriving the continuous Fourier transform for a single damped oscillator and included an orthogonal time window, to restrict the oscillator to a finite duration in time. This allowed us to obtain an expression that does not include δ functions, as was the case for infinite duration. To arrive at this expression, we utilized convolutions. Next, we generalized this expression to the case of multiple damped oscillators, and, finally, we found the expression for the case when the dominant post-merger frequency is evolving in time (according to a specific piecewise function). In all cases, we demonstrated excellent agreement between the obtained continuous Fourier transforms and the corresponding numerical discrete Fourier transform, confirming the validity of our results.

To test the usefulness of our new expressions, we performed a linear fit of our expression for a single damped oscillator to synthetic data with noise and found that it is more versatile and accurate than the simpler Lorentzian expression used in the literature. We anticipate that our new expressions will prove especially valuable in cases of multiple post-merger peaks that are superimposed with arbitrary phases, as well as in cases where the dominant frequency is evolving in time.

In future work, we plan to use the new analytic expressions in detection and Bayesian parameter estimation algorithms of the post-merger phase of binary neutron star mergers. The higher accuracy of our expressions will increase the accuracy of the estimation of physical parameters of post-merger remnants, allowing for new physics to be probed, such as the influence of finite temperatures on the equation of state, the presence of magnetic fields, or the development of late-time rotational instabilities.

Chapter 3

Appendix

Appendix A: Numerical codes for single damped oscillator

A1: Python - Time Domain for a single oscillator

Below is the Python code for a single oscillator in the time domain, with and without the window function.

```
#Imported libraries
import numpy as np
from numpy import exp ,pi,sin
import sympy as sp
from sympy import symbols, Function
from sympy import *
import matplotlib as mpl
import matplotlib.pyplot as plt

#Plot-related configurations
mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
plt.rcParams['figure.figsize']=(8 , 5); plt.rcParams['legend.fontsize']=16

#Set variables
duration=0.03; tmerger = 0
A1 = 10**(-22.5);  tau1 = 2.5*10**(-2.25);  phi1 = -np.pi/2;  f1 = 3100
num=2500
t_n = np.linspace (0-0.01, duration+0.01, num)
dt = t_n[1] - t_n[0]
values_with_window = np.zeros(num)

#Definition of window function
class rect(Function):
  @classmethod
  def eval(cls, arg):
    return Heaviside(arg -tmerger) - Heaviside(arg -tmerger-duration)

#Calculation of strain h(t)
values_without_window = A1 * np.exp(-(t_n - tmerger) / tau1) *
np.sin(2 * np.pi * f1 * (t_n - tmerger) + phi1)
for i in range(num):
  values_with_window[i]= rect(t_n[i])*(A1 * np.exp(-(t_n[i] - tmerger) /
   ↪  tau1) *
np.sin(2 * np.pi * f1 * (t_n[i] - tmerger) + phi1))

#Plot
plt.plot(t_n, values_without_window,color='black', label=r'without
 ↪  window')
plt.plot(t_n, values_with_window,color='cyan', label=r'with window')
plt.xlabel(r'$t$'); plt.ylabel(r'$h(t)$')
plt.axis([0-0.01 ,duration+0.01 ,-10**(-22.2) , 10**(-22.2)])
plt.axhline(y=0,color='grey',linestyle="--")
plt.axvline(x=0,color='grey',linestyle="--")
plt.legend()
plt.show()
```

A2: C - Fourier Transform implementation from `Mathematica`

When obtaining the continuous Fourier transform representation in `Mathematica`, we proceed to implement it in `C`:

```
 //include header files
 #include <stdio.h>
 #include <math.h>
 #include <complex.h>

 //Define variables and their types
 float  A1, tau1, duration, f1, tmerger, phi1;

// Define a function to calculate the Heaviside step function
float HeavisideTheta (float y) {
      float HeavisideTheta_result;
      HeavisideTheta_result = copysignf(0.5,y) + 0.5;
      return HeavisideTheta_result;
 }

//Calculate Fourier transformation with specified parameters.
   double complex fourier(float f, float A1, float tau1, float duration,
   float f1, float tmerger,  float phi1) {

   // Complex variable to store the result
      double complex result;
      //here we inserted the C code
      result =
  (A1*tau1*((cexp(-(duration*(2*I*f*M_PI + 2*I*f1*M_PI + 1/tau1)) -
  ↪  2*I*f*M_PI*tmerger)*(I - 2*f*M_PI*tau1 + 2*f1*M_PI*tau1 + cexp(2*I*phi1
  ↪  + duration*(2*I*f*M_PI + 2*I*f1*M_PI + 1/tau1))*(I - 2*f*M_PI*tau1 -
  ↪  2*f1*M_PI*tau1) + cexp(duration*(2*I*f*M_PI + 2*I*f1*M_PI + 1/tau1))*(-I
  ↪  + 2*f*M_PI*tau1 - 2*f1*M_PI*tau1) + cexp(2*I*(phi1 +
  ↪  2*duration*f1*M_PI))*(-I + 2*f*M_PI*tau1 + 2*f1*M_PI*tau1)))/((-I +
  ↪  2*f*M_PI*tau1 - 2*f1*M_PI*tau1)*(-I + 2*f*M_PI*tau1 + 2*f1*M_PI*tau1)) +
  ↪  I*(cexp(2*I*phi1 - 2*I*f1*M_PI*tmerger + tmerger/tau1)/(1 +
  ↪  4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) - 8*f*f1*cpow(M_PI,2)*cpow(tau1,2)
  ↪  + 4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)) - cexp((2*I*f1*M_PI +
  ↪  1/tau1)*tmerger)/(1 + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) +
  ↪  8*f*f1*cpow(M_PI,2)*cpow(tau1,2) +
  ↪  4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)))*HeavisideTheta(-duration -
  ↪  tmerger) + ((-I*cexp(2*I*phi1 - 2*I*f1*M_PI*tmerger + tmerger/tau1))/(1
  ↪  + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) -
  ↪  8*f*f1*cpow(M_PI,2)*cpow(tau1,2) +
  ↪  4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)) + (I*cexp((2*I*f1*M_PI +
  ↪  1/tau1)*tmerger))/(1 + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) +
  ↪  8*f*f1*cpow(M_PI,2)*cpow(tau1,2) +
  ↪  4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)))*HeavisideTheta(-tmerger)))/(2.*cexp(I*phi1));
  return result;
```

A3: C - Interface for Python implementation

The interface of the above C code is:

```
//the name of the module we are creating.
%module python_name

//C code that includes the <complex.h> header and declares the variables
%{
#include <complex.h>
extern float A1, tau1, duration, f1, tmerger, phi1;
%}

//This specifies that the variables declared earlier should be treated
//as input parameters.
%apply float& {A1, tau1, duration, f1, tmerger, phi1};

// C/C++ functions are declared to be included in the SWIG interface.
%{
   extern float HeavisideTheta(float y);
   extern double complex fourier(float f, float A1, float tau1, float
↪  duration,
   float f1, float tmerger, float phi1);
 %}

%include <complex.i>
```

```
//These lines specify the parts of the SWIG interface that can be
//called from Python.
extern float HeavisideTheta(float y);
extern double complex fourier(float f, float A1, float tau1, float duration,
float f1, float tmerger, float phi1);
```

[0.5cm]

A4: Terminal Commands

- `swig -python interface_name.i` → this creates an `interface_name_wrapper.c` file
- `gcc -O3 -c -fPIC c_code.c interface_name_wrapper.c -I/path_of_python_header_files`→ this creates an interface`_name_wrapper.o` and a `c_code.o` file.
- `gcc -O3 -shared c_code.o interface_name_wrapper.o -o _python_name.so` → This produces the `.so` extention we can import in Python.

A5: Python - Fourier Transform and Plots

```
#Import the .so file
import _python_name
#Import the libraries
import numpy as np
from numpy import exp ,pi
import matplotlib as mpl
import matplotlib.pyplot as plt
import scipy
from scipy import fftpack
from sympy import *

#Plot-related configurations
mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
plt.rcParams['figure.figsize']=(8.5, 5.5);
↪  plt.rcParams['legend.fontsize']=16

#Set variables
duration=0.03  ;tmerger = 0
A1 = 10**(-22.5); tau1 = 2.5*10**(-2.25) ;phi1 = -np.pi/2 ;f1 = 3100

#points for analytical Fourier
f_min = 0
f_max = 5000
num_points = 800
f_values = np.linspace(f_min, f_max, num_points)
```

```
result_values = np.array([_python_name.fourier(f , A1, tau1, duration, f1,
↪  tmerger, phi1) for f in f_values])
real_values = np.real(result_values)
imag_values = np.imag(result_values)

#points for numerical Fourier
num=40000
t_n = np.linspace (tmerger, tmerger +duration, num)
dt = t_n[1] - t_n[0]
h_n =np.zeros(num)
for i in range (num):
 h_n[i] = A1 * np.exp(-(t_n[i] - tmerger) / tau1) * np.sin(2 * np.pi * f1 *
  ↪  (t_n[i] - tmerger) + phi1)

#numerical Fourier
H_k = fftpack.fft(h_n)
H_k_shift = fftpack.fftshift(H_k)
f_k = fftpack.fftfreq(h_n.size, d = dt)
f_shift = fftpack.fftshift(f_k)
A=np.abs(H_k_shift)

#Plot of numerical Fourier
plt.plot(f_shift, A*dt, 'co', markersize=2, label=r'$discrete$')

#Plot of analytical Fourier
plt.plot(f_values, np.sqrt(real_values**2+imag_values**2), color='magenta',
↪  linewidth=1.0, label=r'$continuous$')

#Plot Parameters
plt.xlabel(r'$f$') ; plt.ylabel(r"$\~h(f)$")
plt.axis([0,5000,1*10**(-29) , 1*10**(-24)])
plt.yscale("log")
plt.legend()
plt.show()
```

Appendix B: Four simple oscillators

B1: Python - Contribution of each oscillator in the time domain

```
import numpy as np
from numpy import exp ,pi,sin
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
```

```
plt.rcParams['figure.figsize']=(8.5,5.5) ;
 ↪  plt.rcParams['legend.fontsize']=16

#Set variables
duration=0.125
tmerger = 0
A1 = 10**(-22.5); tau1 = 10**(-2.25); f1 = 3100; phi1 = -np.pi/2
A2 = 10**(-22.3); tau2 = 10**(-2.8); f2 = 2750; phi2 = np.pi/2
A3 = 10**(-22.3); tau3 = 10**(-2.8); f3 = 2460; phi3 = -np.pi/2
A4 = 10**(-22.3); tau4 = 10**(-2.8); f4 = 3640; phi4 = -np.pi/4

#Time steps
num=1500
t_n = np.linspace (0, duration, num)
dt = t_n[1] - t_n[0]

#Defining each model
Model1 = np.zeros(num)
Model2 = np.zeros(num)
Model3 = np.zeros(num)
Model4 = np.zeros(num)

for i in range(num):
  Model1[i] = (A1 * np.exp(-(t_n[i] - tmerger) / tau1)) * np.sin(2 * np.pi *
   ↪  f1 * (t_n[i] - tmerger) + phi1)
  Model2[i]=(A2 * np.exp(-(t_n[i] - tmerger) / tau2)) * np.sin(2 * np.pi *
   ↪  f2 * (t_n[i] - tmerger) + phi2)
   Model3[i] = A3 * np.exp(-(t_n[i] - tmerger) / tau3) * np.sin(2 * np.pi *
    ↪  f3 * (t_n[i] - tmerger) + phi3)
   Model4[i] = A4 * np.exp(-(t_n[i] - tmerger) / tau4) * np.sin(2 * np.pi *
    ↪  f4 * (t_n[i] - tmerger) + phi4)

#Plot
plt.plot(t_n, Model1,color='blueviolet', label=r'Model1')
plt.plot(t_n, Model2,color= 'yellow', label=r'Model2')
plt.plot(t_n, Model3,color='mediumaquamarine', label=r'Model3')
plt.plot(t_n, Model4,color='indianred', label=r'Model4')
plt.xlabel(r'$t$')
plt.ylabel(r'$h(t)$')
plt.axis([0 ,0.03,-3/5*10**(-22.2) , 3/5*10**(-22.2)])
plt.axhline(y=0,color='grey',linestyle="--")
plt.axvline(x=0,color='grey',linestyle="--")
plt.legend()
plt.show()
```

B2: Python - Time Domain for four oscillators

```
import numpy as np
from numpy import exp ,pi,sin
import matplotlib as mpl
import matplotlib.pyplot as plt
mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
plt.rcParams['figure.figsize']=(8.5,5.5) ;
↪  plt.rcParams['legend.fontsize']=16

#Set variables
duration=0.125
tmerger = 0
A1 = 10**(-22.5); tau1 = 10**(-2.25); f1 = 3100; phi1 = -np.pi/2
A2 = 10**(-22.3); tau2 = 10**(-2.8); f2 = 2750; phi2 = np.pi/2
A3 = 10**(-22.3); tau3 = 10**(-2.8); f3 = 2460; phi3 = -np.pi/2
A4 = 10**(-22.3); tau4 = 10**(-2.8); f4 = 3640; phi4 = -np.pi/4

#Time step
num=2500
t_n = np.linspace (0, duration, num)
dt = t_n[1] - t_n[0]
sum = np.zeros(num)

for i in range(num):
  sum[i]= A1 * np.exp(-(t_n[i] -tmerger) / tau1) * np.sin(2 * np.pi * f1 *
   ↪  (t_n[i] - tmerger) + phi1) +A2 * np.exp(-(t_n[i] - tmerger) / tau2) *
   ↪  np.sin(2*np.pi*f2*(t_n[i] - tmerger) + phi2)+ A3 * np.exp(-(t_n[i] -
   ↪  tmerger) / tau3) * np.sin(2 * np.pi * f3 * (t_n[i] - tmerger) + phi3)
   ↪  + A4 * np.exp(-(t_n[i] - tmerger) / tau4) * np.sin(2 * np.pi * f4 *
   ↪  (t_n[i] - tmerger) + phi4)

#Plot
plt.plot(t_n, sum,color='cyan')
plt.xlabel(r'$t$')
plt.ylabel(r'$h(t)$')
plt.axis([0 ,0.02 ,-10**(-22) ,10**(-22)])
plt.axhline(y=0,color='grey',linestyle="--")
plt.axvline(x=0,color='grey',linestyle="--")
plt.show()
```

B3: C - Fourier Transform implementation from Mathematica

When obtaining the continuous Fourier transform representation in `Mathematica`, we proceed to implement it in `C`:

```
//Add header files
#include <stdio.h>
#include <math.h>
#include <complex.h>

//Define variables and their types
float  A1, phi1, f1, tmerger, tau1, duration, A2, phi2, f2, tau2, A3, phi3,
↪  f3, tau3, A4, phi4, f4, tau4;

// Define a function to calculate the Heaviside step function
float HeavisideTheta (float y) {
        float HeavisideTheta_result;
        HeavisideTheta_result = copysignf(0.5,y) + 0.5;
        return HeavisideTheta_result;
}

//Calculate Fourier transformation with specified parameters
double complex fourier(float f, float A1, float phi1, float f1, float
↪  tmerger, float tau1, float duration, float A2, float phi2, float f2,
↪  float tau2, float A3, float phi3, float f3, float tau3, float A4, float
↪  phi4, float f4, float tau4) {

// Complex variable to store the result (Here insert C code)
double complex result;
```

```
  result =(A1*tau1*((cexp(-(duration*(2*I*f*M_PI + 2*I*f1*M_PI + 1/tau1))
↪  - 2*I*f*M_PI*tmerger)*(I - 2*f*M_PI*tau1 + 2*f1*M_PI*tau1 +
↪  cexp(2*I*phi1 + duration*(2*I*f*M_PI + 2*I*f1*M_PI + 1/tau1))*(I -
↪  2*f*M_PI*tau1 - 2*f1*M_PI*tau1) + cexp(duration*(2*I*f*M_PI +
↪  2*I*f1*M_PI + 1/tau1))*(-I + 2*f*M_PI*tau1 - 2*f1*M_PI*tau1) +
↪  cexp(2*I*(phi1 + 2*duration*f1*M_PI))*(-I + 2*f*M_PI*tau1 +
↪  2*f1*M_PI*tau1)))/((-I + 2*f*M_PI*tau1 - 2*f1*M_PI*tau1)*(-I +
↪  2*f*M_PI*tau1 + 2*f1*M_PI*tau1)) + I*(cexp(2*I*phi1 -
↪  2*I*f1*M_PI*tmerger + tmerger/tau1)/(1 +
↪  4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) - 8*f*f1*cpow(M_PI,2)*cpow(tau1,2)
↪  + 4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)) - cexp((2*I*f1*M_PI +
↪  1/tau1)*tmerger)/(1 + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) +
↪  8*f*f1*cpow(M_PI,2)*cpow(tau1,2) +
↪  4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)))*HeavisideTheta(-duration -
↪  tmerger) + ((-I*cexp(2*I*phi1 - 2*I*f1*M_PI*tmerger + tmerger/tau1))/(1
↪  + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) -
↪  8*f*f1*cpow(M_PI,2)*cpow(tau1,2) +
↪  4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)) + (I*cexp((2*I*f1*M_PI +
↪  1/tau1)*tmerger))/(1 + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau1,2) +
↪  8*f*f1*cpow(M_PI,2)*cpow(tau1,2) +
↪  4*cpow(f1,2)*cpow(M_PI,2)*cpow(tau1,2)))*HeavisideTheta(-tmerger)))/(2.*cexp(I*phi1))
↪  + (A2*tau2*((cexp(-(duration*(2*I*f*M_PI + 2*I*f2*M_PI + 1/tau2)) -
↪  2*I*f*M_PI*tmerger)*(I - 2*f*M_PI*tau2 + 2*f2*M_PI*tau2 + cexp(2*I*phi2
↪  + duration*(2*I*f*M_PI + 2*I*f2*M_PI + 1/tau2))*(I - 2*f*M_PI*tau2 -
↪  2*f2*M_PI*tau2) + cexp(duration*(2*I*f*M_PI + 2*I*f2*M_PI + 1/tau2))*(-I
↪  + 2*f*M_PI*tau2 - 2*f2*M_PI*tau2) + cexp(2*I*(phi2 +
↪  2*duration*f2*M_PI))*(-I + 2*f*M_PI*tau2 + 2*f2*M_PI*tau2)))/((-I +
↪  2*f*M_PI*tau2 - 2*f2*M_PI*tau2)*(-I + 2*f*M_PI*tau2 + 2*f2*M_PI*tau2)) +
↪  I*(cexp(2*I*phi2 - 2*I*f2*M_PI*tmerger + tmerger/tau2)/(1 +
↪  4*cpow(f,2)*cpow(M_PI,2)*cpow(tau2,2) - 8*f*f2*cpow(M_PI,2)*cpow(tau2,2)
↪  + 4*cpow(f2,2)*cpow(M_PI,2)*cpow(tau2,2)) - cexp((2*I*f2*M_PI +
↪  1/tau2)*tmerger)/(1 +     4*cpow(f,2)*cpow(M_PI,2)*cpow(tau2,2) +
↪  8*f*f2*cpow(M_PI,2)*cpow(tau2,2) +
↪  4*cpow(f2,2)*cpow(M_PI,2)*cpow(tau2,2)))*HeavisideTheta(-duration -
↪  tmerger) + ((-I*cexp(2*I*phi2 - 2*I*f2*M_PI*tmerger + tmerger/tau2))/(1
↪  + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau2,2) -
↪  8*f*f2*cpow(M_PI,2)*cpow(tau2,2) +
↪  4*cpow(f2,2)*cpow(M_PI,2)*cpow(tau2,2)) + (I*cexp((2*I*f2*M_PI +
↪  1/tau2)*tmerger))/(1 + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau2,2) +
↪  8*f*f2*cpow(M_PI,2)*cpow(tau2,2) +
↪  4*cpow(f2,2)*cpow(M_PI,2)*cpow(tau2,2)))*HeavisideTheta(-tmerger)))/(2.*cexp(I*phi2))
↪  + (A3*tau3*((cexp(-(duration*(2*I*f*M_PI + 2*I*f3*M_PI + 1/tau3)) -
↪  2*I*f*M_PI*tmerger)*(I - 2*f*M_PI*tau3 + 2*f3*M_PI*tau3 + cexp(2*I*phi3
↪  + duration*(2*I*f*M_PI + 2*I*f3*M_PI + 1/tau3))*(I - 2*f*M_PI*tau3 -
↪  2*f3*M_PI*tau3) + cexp(duration*(2*I*f*M_PI + 2*I*f3*M_PI + 1/tau3))*(-I
↪  + 2*f*M_PI*tau3 - 2*f3*M_PI*tau3) + cexp(2*I*(phi3 +
↪  2*duration*f3*M_PI))*(-I + 2*f*M_PI*tau3 + 2*f3*M_PI*tau3)))/((-I +
↪  2*f*M_PI*tau3 - 2*f3*M_PI*tau3)*(-I + 2*f*M_PI*tau3 + 2*f3*M_PI*tau3)) +
↪  I*(cexp(2*I*phi3 - 2*I*f3*M_PI*tmerger + tmerger/tau3)/(1 +
↪  4*cpow(f,2)*cpow(M_PI,2)*cpow(tau3,2) - 8*f*f3*cpow(M_PI,2)*cpow(tau3,2)
↪  + 4*cpow(f3,2)*cpow(M_PI,2)*cpow(tau3,2)) - cexp((2*I*f3*M_PI +
↪  1/tau3)*tmerger)/(1 + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau3,2) +
↪  8*f*f3*cpow(M_PI,2)*cpow(tau3,2) +
↪  4*cpow(f3,2)*cpow(M_PI,2)*cpow(tau3,2)))*HeavisideTheta(-duration -
↪  tmerger) + ((-I*cexp(2*I*phi3 - 2*I*f3*M_PI*tmerger + tmerger/tau3))/(1
↪  + 4*cpow(f,2)*cpow(M_PI,2)*cpow(tau3,2) -
return result;
}
```

B4: C - Interface for Python implementation

```
//the name of the module we are creating.
%module python_name

//C code that includes the <complex.h> header and declares the variables
%{
#include <complex.h>
extern float A1, phi1, f1, tmerger, tau1, duration, A2, phi2, f2, tau2, A3,
↪  phi3, f3, tau3, A4, phi4, f4, tau4;
%}

//This specifies that the variables declared earlier should be treated
//as input parameters.
%apply float& {A1, phi1, f1, tmerger, tau1, duration, A2, phi2, f2, tau2,
↪  A3, phi3, f3, tau3, A4, phi4, f4, tau4};

// C/C++ functions are declared to be included in the SWIG interface
%{
extern float HeavisideTheta(float y);
extern double complex fourier(float f, float A1, float phi1, float f1, float
↪  tmerger, float tau1, float duration, float A2, float phi2, float f2,
↪  float tau2, float A3, float phi3, float f3, float tau3, float A4, float
↪  phi4, float f4, float tau4);
%}

%include <complex.i>

//These lines specify the parts of the SWIG interface that can be
//called from Python.
extern float HeavisideTheta(float y);
extern double complex fourier(float f, float A1, float phi1, float f1, float
↪  tmerger, float tau1, float duration, float A2, float phi2, float f2,
↪  float tau2, float A3, float phi3, float f3, float tau3, float A4, float
↪  phi4, float f4, float tau4);
```

B5: Terminal Commands

- `swig -python interface_name.i` → this creates an `interface_name_wrapper.c` file
- `gcc -O3 -c -fPIC c_code.c interface_name_wrapper.c -I/path_of_python_header_files` → this creates an `interface_name_wrapper.o` and a `c_code.o` file.
- `gcc -O3 -shared c_code.o interface_name_wrapper.o -o _python_name.so` → This produces the `.so` extention we can import in Python.

B6: Python - Fourier Transform and Plots

```
#Import libraries
import _python_name
import numpy as np
from numpy import exp ,pi
import matplotlib as mpl
import matplotlib.pyplot as plt
import scipy
from scipy import fftpack
mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
plt.rcParams['figure.figsize']=(8.5,5.5) ;
↪  plt.rcParams['legend.fontsize']=16

#Set variables
duration=0.125
tmerger = 0
A1 = 10**(-22.5); tau1 = 10**(-2.25); f1 = 3100; phi1 = -np.pi/2
A2 = 10**(-22.3); tau2 = 10**(-2.8); f2 = 2750; phi2 = np.pi/2
A3 = 10**(-22.3); tau3 = 10**(-2.8); f3 = 2460; phi3 = -np.pi/2
A4 = 10**(-22.3); tau4 = 10**(-2.8); f4 = 3640; phi4 = -np.pi/4

#points for discrete Fourier steps
num=100000
t_n = np.linspace (0, duration, num)
dt = t_n[1] - t_n[0]

#Continuous Fourier
f_min = 0
f_max = 5000
num_points = 2500
f_values = np.linspace(f_min, f_max, num_points)
result_values = np.array([_simple4.fourier(f , A1, phi1, f1, tmerger, tau1,
↪  duration, A2, phi2, f2, tau2 , A3, phi3, f3, tau3, A4, phi4, f4, tau4)
↪  for f in f_values])
real_values = np.real(result_values)
imag_values = np.imag(result_values)

#Discrete Fourier
h_n =np.zeros(num)
for i in range (num):
  h_n[i] = A1 * np.exp(-(t_n[i] - tmerger) / tau1) * np.sin(2 * np.pi * f1 *
   ↪  (t_n[i] - tmerger) + phi1) + A2 * np.exp(-(t_n[i] - tmerger) / tau2) *
   ↪  np.sin(2 * np.pi * f2 * (t_n[i] - tmerger) + phi2) + A3 *
   ↪  np.exp(-(t_n[i] - tmerger) / tau3) * np.sin(2 * np.pi * f3 * (t_n[i] -
   ↪  tmerger) + phi3) + A4 * np.exp(-(t_n[i] - tmerger) / tau4) * np.sin(2
   ↪  * np.pi * f4 * (t_n[i] - tmerger) + phi4)

H_k = fftpack.fft(h_n)
H_k_shift = fftpack.fftshift(H_k)
f_k = fftpack.fftfreq(h_n.size, d = dt)
f_shift = fftpack.fftshift(f_k)
A=np.abs(H_k_shift)

#Plots
plt.plot(f_shift, A*dt, 'co', markersize=1,  label=r'$discrete$')
plt.plot(f_values,np.sqrt(real_values**2+imag_values**2), color='magenta',
 ↪  linewidth=1.0, label=r'$continuous$')
plt.xlabel(r'$f$') ; plt.ylabel(r"$\~h(f)$")
plt.axis([0,5000,1*10**(-27.5) , 1*10**(-25)])
plt.yscale("log")
plt.legend()
plt.show()
```

Appendix C: Four oscillators, of which one with time-dependent frequency

C1: Python - Time domain for four oscillators.

This code presents a time domain model for the gravitational wave strain that features four oscillators, one of which incorporates a time-dependent frequency.

```
import numpy as np
from numpy import exp ,pi
import matplotlib as mpl
import matplotlib.pyplot as plt
from scipy import fftpack
mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
plt.rcParams['figure.figsize']=(8.5,5.5)
plt.rcParams['legend.fontsize']=16

#Set variables
Apeak = 10/exp(2)
tstar = -15
fpeak0 = 3
tmerger = -20
taupeak = 30
```

```
zdrift = -0.15
phipeak0 = 0
T = 40
A1 = 10/exp(2); tau1 = 3; f1 = 2.4; phi1 = pi/2
A2 = 10/exp(2); tau2 = 3; f2 = 2; phi2 = -np.pi/2
A3 = 10/exp(2); tau3 = 3; f3 = 4; phi3 = -np.pi/2
phipeaktstar = 2 * np.pi * (fpeak0 + zdrift * (tstar - tmerger) / 2) *
↪  (tstar - tmerger) + phipeak0
fpeaktstar = fpeak0 + zdrift * (tstar - tmerger) / 2


num=1500
t_n = np.linspace (-20, 20, num)
dt = t_n[1] - t_n[0]

y_n =np.zeros(num)
h_n =np.zeros(num)

#Strain of Apeak
for i in range(num):
    if t_n[i] <= tstar:
        y_n[i] = Apeak * np.exp(-(t_n[i] - tmerger) / taupeak) * np.sin(2 *
        ↪  np.pi * (fpeak0 + zdrift * (t_n[i] - tmerger) / 2) * (t_n[i] -
        ↪  tmerger) + phipeak0)
    else:
        y_n[i] = Apeak * np.exp(-(t_n[i] - tmerger) / taupeak) * np.sin(2 *
        ↪  np.pi * fpeaktstar * (t_n[i] - tstar) + phipeaktstar)

#Total strain
for i in range (num):
    h_n[i] = A1 * np.exp(-(t_n[i] - tmerger) / tau1) * np.sin(2 * np.pi * f1
    ↪  * (t_n[i] - tmerger) + phi1) + A2 * np.exp(-(t_n[i] - tmerger) /
    ↪  tau2) * np.sin(2 * np.pi* f2 * (t_n[i] - tmerger) + phi2) + A3 *
    ↪  np.exp(-(t_n[i] - tmerger) / tau3) * np.sin(2 * np.pi * f3 * (t_n[i]
    ↪  - tmerger) + phi3) + y_n[i]

#Plot
plt.plot(t_n, h_n)
plt.plot(t_n, h_n,color='cyan')
plt.xlabel(r'$t$')
plt.ylabel(r'$h(t)$')
plt.axhline(y=0,color='grey',linestyle="--")
plt.show()
```

C2: C - Fourier Transform implementation from Mathematica

Steps are in general as before, except that the `Erfi` function is defined. The expression is too long in this case, so it will not be added here.

```
//Add header files
#include <stdio.h>
#include "Faddeeva.h"
#include <math.h>
#include <complex.h>

//Define variables and their types
float Apeak, tstar, fpeak0, tmerger, taupeak, zdrift, phipeak0, T, f1, f2,
↪  f3, tau1, tau2, tau3, A1, A2, A3, phi1, phi2, phi3;


// Define a function to calculate the Erfi function
double complex Erfi(double complex z) {
    double complex result = Faddeeva_erfi(z, 1e-12);
    return result;
}

// Define a function to calculate the Heaviside step function
float HeavisideTheta (float y) {
        float HeavisideTheta_result;
        HeavisideTheta_result = copysignf(0.5,y) + 0.5;
        return HeavisideTheta_result;
}

//Calculate Fourier transformation with specified parameters
double complex fourier(float f, float Apeak, float tstar, float fpeak0,
↪  float tmerger, float taupeak, float zdrift, float phipeak0, float T,
↪  float f1, float f2, float f3, float tau1, float tau2, float tau3, float
↪  A1, float A2, float A3, float phi1, float phi2, float phi3) {

 // Complex variable to store the result (here insert C code)
        double complex f_result;
        f_result =

return result;
}
```

C3: C - Interface for Python implementation

```
  //the name of the module we are creating.
  %module python_name

  //Includes the <complex.h> header and declares the variables
  %{
  extern float Apeak, tstar, fpeak0, tmerger, taupeak, zdrift, phipeak0, T,
↪  f1, f2, f3, tau1, tau2, tau3, A1, A2, A3, phi1, phi2, phi3;
  %}
  //This specifies that the variables declared earlier should be treated
  //as input parameters.
  %apply float& { Apeak, tstar, fpeak0, tmerger, taupeak, zdrift, phipeak0,
↪  T, f1, f2, f3, tau1, tau2, tau3, A1, A2, A3, phi1, phi2, phi3 };

  // C/C++ functions are declared to be included in the SWIG interface
  %{
  #include "Faddeeva.h"
  extern double complex Erfi(double complex z);
  extern float HeavisideTheta(float y);
  extern double complex fourier(float f, float Apeak, float tstar, float
↪  fpeak0, float tmerger, float taupeak, float zdrift, float phipeak0,
↪  float T, float f1, float f2, float f3, float tau1, float tau2, float
↪  tau3, float A1, float A2, float A3, float phi1, float phi2, float phi3);
  %}

  %include <complex.i>
  %include "Faddeeva.h"


  //These lines specify the parts of the SWIG interface that can be
//called from Python.

  extern double complex Faddeeva_w(double complex z, double relerr);
  extern double Faddeeva_w_im(double x);

  extern double complex Faddeeva_erfcx(double complex z, double relerr);
  extern double Faddeeva_erfcx_re(double x);

  extern double complex Faddeeva_erf(double complex z, double relerr);
  extern double Faddeeva_erf_re(double x);

  extern double complex Faddeeva_erfi(double complex z, double relerr);
  extern double Faddeeva_erfi_re(double x);

  extern double complex Faddeeva_erfc(double complex z, double relerr);
  extern double Faddeeva_erfc_re(double x);

  extern double complex Faddeeva_Dawson(double complex z, double relerr);
  extern double Faddeeva_Dawson_re(double x);

  extern double complex Erfi(double complex z);
  extern float HeavisideTheta(float y);
  extern double complex fourier(float f, float Apeak, float tstar, float
↪  fpeak0, float tmerger, float taupeak, float zdrift, float phipeak0,
↪  float T, float f1, float f2, float f3, float tau1, float tau2, float
↪  tau3, float A1, float A2, float A3, float phi1, float phi2, float phi3);
```

C4: Terminal Commands

- gcc −std=c99 −c −fPIC Faddeeva.c −o Faddeeva_c.o −lm → creates an object file.
- g++ −c −fPIC Faddeeva.cc −o Faddeeva_cc.o −lm → creates an object file.
- gcc −c −fPIC c_code.c −I/path_of_python_header_files/include/python3.10
- swig −python interface_name.i → this creates an interface_wrapper.c file
- gcc −O3 −c −fPIC c_code.c interface_name_wrapper.c −I/path_of_python_header_files→ this creates an interface_name_wrapper.o and a c_code.o object file.
- gcc −shared c_code.o interface_name_wrapper.o Faddeeva_cc.o Faddeeva_c.o −o _python_name.so −L/usr/lib/$x86_64$−linux−gnu−I/path_of_python_header_files/include/pytho −lstdc++ −lm→ this creates an _python_name.so library file.

Now that we have the continuous Fourier transform as a library with a .so extention, we can import it into Python.

C5: Python - Fourier Transform and Plots

```
#Import libraries and make the plot-related configurations
import _python_name
import numpy as np
from numpy import exp ,pi
import matplotlib as mpl
import matplotlib.pyplot as plt
from scipy import fftpack
mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
plt.rcParams['figure.figsize']=(8.5,5.5) ;
↪  plt.rcParams['legend.fontsize']=16

#Set values
Apeak = 10/exp(2); fpeak0 = 3; taupeak = 30 ;phipeak0 = 0
tstar = -15
tmerger = -20
zdrift = -0.15
T = 40
A1 = 10/exp(2); tau1 = 3; f1 = 2.4; phi1 = pi/2
A2 = 10/exp(2); tau2 = 3; f2 = 2; phi2 = -np.pi/2
A3 = 10/exp(2); tau3 = 3; f3 = 4; phi3 = -np.pi/2
phipeaktstar = 2 * np.pi * (fpeak0 + zdrift * (tstar - tmerger) / 2) *
↪  (tstar - tmerger) + phipeak0
fpeaktstar = fpeak0 + zdrift * (tstar - tmerger) / 2
```

```
f_min = 0
f_max = 6
num_points = 10000
f_values = np.linspace(f_min, f_max, num_points)
result_values = np.array([_test.fourier(f , Apeak, tstar, fpeak0, tmerger,
 ↪  taupeak, zdrift, phipeak0 , T, f1, f2, f3, tau1, tau2, tau3, A1, A2, A3,
 ↪  phi1, phi2, phi3) for f in f_values])
real_values = np.real(result_values)
imag_values = np.imag(result_values)


#Definition of strain and Discrete Fourier
num=10000
t_n = np.linspace (-20, 20, num)
dt = t_n[1] - t_n[0]
y_n =np.zeros(num)
h_n =np.zeros(num)

for i in range(num):
    if t_n[i] <= tstar:
        y_n[i] = Apeak * np.exp(-(t_n[i] - tmerger) / taupeak) * np.sin(2 *
         ↪  np.pi * (fpeak0 + zdrift * (t_n[i] - tmerger) / 2) * (t_n[i] -
         ↪  tmerger) + phipeak0)
    else:
        y_n[i] = Apeak * np.exp(-(t_n[i] - tmerger) / taupeak) * np.sin(2 *
         ↪  np.pi * fpeaktstar * (t_n[i] - tstar) + phipeaktstar)

for i in range (num):
    h_n[i] = A1 * np.exp(-(t_n[i] - tmerger) / tau1) * np.sin(2 * np.pi * f1
     ↪  * (t_n[i] - tmerger) + phi1) + A2 * np.exp(-(t_n[i] - tmerger) /
     ↪  tau2) * np.sin(2 * np.pi* f2 * (t_n[i] - tmerger) + phi2) + A3 *
     ↪  np.exp(-(t_n[i] - tmerger) / tau3) * np.sin(2 * np.pi * f3 * (t_n[i]
     ↪  - tmerger) + phi3) + y_n[i]

#Discrete Fourier Transform
H_k = fftpack.fft(h_n)
H_k_shift = fftpack.fftshift(H_k)
f_k = fftpack.fftfreq(h_n.size, d = dt)
f_shift = fftpack.fftshift(f_k)

A = np.abs(H_k_shift)
plt.plot(f_shift, A*dt, 'co',label=r'$discrete$')
plt.plot(f_values, np.sqrt(real_values**2+imag_values**2), 'm',
 ↪  label=r'$continuous$')
plt.xlabel(r'$f$') ; plt.ylabel(r"$\~h(f)$")
plt.xlim(0,6)
plt.ylim(1*10**(-2),100)
plt.yscale('log')
plt.legend()
plt.show()
```

Appendix D: Python - Nonlinear Least-Squares Fit

Curve fitting code for the analytical and Lorentzian models to data.

```
import numpy as np
from numpy import pi,exp,sin, heaviside
from scipy import fftpack
import random
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline
import sys
import warnings
if not sys.warnoptions: warnings.simplefilter("ignore")

!pip install lmfit

from lmfit.models import ExpressionModel

mpl.rcParams['mathtext.fontset'] = 'stix'
mpl.rcParams['font.family'] = 'STIXGeneral'
mpl.rcParams['font.size'] = 18
mpl.rc('xtick', labelsize=16) ; mpl.rc('ytick', labelsize=16)
plt.rcParams['figure.figsize']=8,4 ; plt.rcParams['legend.fontsize']=16


##Single frequency damped sinusoid with some noise
tmerger=0
tau1 = 40
phi1 = -3*np.pi/2
f1 = 0.5
A1 = 1
Anoise = 0.02
duration=80
num=401
t_n = np.linspace(0,duration, num)       # linearly space time array
dt= t_n[1] - t_n[0] # increment between times in time array
h_n = A1* np.exp(-(t_n - tmerger) / tau1) * np.sin(2*np.pi*f1*(t_n -
 ↪  tmerger) + phi1)

for i in range(0,num):               # add noise
   h_n[i] += Anoise*random.randint(-10,10)

H_k= fftpack.fft(h_n)
H_k_shift = fftpack.fftshift(H_k)
```

```
f_k = fftpack.fftfreq(h_n.size, d = dt)
f_shift= fftpack.fftshift(f_k)

A = np.abs(H_k_shift)


#Create an array y(x) of the numerical Fourier transform for positive
↪  frequencies only.
halflen = int((len(f_shift)-1)/2)
x = f_shift[halflen:]
y = A[halflen:]*dt

#Now use the analytic model to do the fit, starting with the trial values
expr = ("abs(0.5 * A1 * exp(-1j * phi1) * tau1 * ("
    "((exp(-80 * (2j * x* pi + 2j * f1 * pi + 1 / tau1)) - 2j * x* pi * 0) *
    ↪  ("
    "1j - 2 * x* pi * tau1 + 2 * f1 * pi * tau1 + exp(2j * phi1 + 80 * (2j *
    ↪  x* pi + 2j * f1 * pi + 1 / tau1)) * ("
    "1j - 2 * x* pi * tau1 - 2 * f1 * pi * tau1) + exp(80 * (2j * x* pi + 2j
    ↪  * f1 * pi + 1 / tau1)) * (-1j + 2 * x* pi * tau1 - 2 * f1 * pi *
    ↪  tau1) + exp(2j * (phi1 + 2 * 80 * f1 * pi)) * (-1j + 2 * x* pi *
    ↪  tau1 + 2 * f1 * pi * tau1))"
    ") / ("
    "(-1j + 2 * x* pi * tau1 - 2 * f1 * pi * tau1) * (-1j + 2 * x* pi * tau1
    ↪  + 2 * f1 * pi * tau1)"
    ") + 1j * ("
    "exp(2j * phi1 - 2j * f1 * pi * 0 + 0 / tau1) / (1 + 4 * (x* pi)**2 *
    ↪  tau1**2 - 8 * x* f1 * (pi)**2 * tau1**2 + 4 * (f1 * pi)**2 *
    ↪  tau1**2) - exp((2j * f1 * pi + 1 / tau1) * 0) / (1 + 4 * (x* pi)**2
    ↪  * tau1**2 + 8 * x* f1 * (pi)**2 * tau1**2 + 4 * (f1 * pi)**2 *
    ↪  tau1**2)))"
    " * (80 + 0 >= 0) * (0 >= 0))"
)
gmod_analytic = ExpressionModel(expr)
result = gmod_analytic.fit(y, x=x, tau1 = 40, f1 = 0.5, A1 = 1,
↪  phi1=-3*np.pi/2)
print(result.fit_report())

#Now use the Lorentzian model to do the fit, starting with the trial values
gmod_lorentzian = ExpressionModel("abs(c0*c2/sqrt( (x-c1)**2 + c2**2 ) *
↪  exp(-1j*arctan((x-c1)/c2)))")
result_lorentzian = gmod_lorentzian.fit(y, x=x, c0 = 17.5, c1 = 0.5, c2 =
↪  1/200)
print(result_lorentzian.fit_report())

#Plot the best fit against the data
plt.plot(x, y, label="numerical")
plt.plot(x, result_lorentzian.best_fit, '-', label='best fit (Lorentzian)')
plt.plot(x, result.best_fit, '-', label='best fit (this work)')
plt.xlabel(r'$f$')
plt.ylabel(r'$|\tilde h|$')
plt.yscale("log", nonpositive='clip')
plt.legend()
plt.show()
```

Bibliography

[1] Observation of gravitational waves from a binary black hole merger. *Phys. Rev. Lett.*, 116:061102, Feb 2016.

[2] J. Aasi et al. Advanced LIGO. *Class. Quant. Grav.*, 32:074001, 2015.

[3] B.P. Abbott, R. Abbott, T.D. Abbott, F. Acernese, K. Ackley, C. Adams, T. Adams, P. Addesso, R.X. Adhikari, V.B. Adya, and et al. Analytic models of the spectral properties of gravitational waves from neutron star merge remnants. *Physical Review Letters*, 105(16), Oct 2017.

[4] R. Abbott et al. GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo During the Second Part of the Third Observing Run. 11 2021.

[5] Amaro-Seoane P., et al. Laser interferometer space antenna, 2017.

[6] A. Bauswein and N. Stergioulas. Unified picture of the post-merger dynamics and gravitational wave emission in neutron star mergers. *Phys. Rev. D*, 91:124056, Jun 2015.

[7] Andreas Bauswein, Nikolaos Stergioulas, and Hans-Thomas Janka. Exploring properties of high-density matter through remnants of neutron-star mergers. *European Physical Journal A*, 52:56, March 2016.

[8] A. Buonanno and T. Damour. Effective one-body approach to general relativistic two-body dynamics. *Phys. Rev. D*, 59:084006, Mar 1999.

[9] Alessandra Buonanno. Gravitational waves. In *Les Houches Summer School - Session 86: Particle Physics and Cosmology: The Fabric of Spacetime*, 9 2007.

[10] Alessandra Buonanno and B. S. Sathyaprakash. *Sources of Gravitational Waves: Theory and Observations.* 10 2014.

[11] J. A. Clark, A. Bauswein, N. Stergioulas, and D. Shoemaker. Observing gravitational waves from the post-merger phase of binary neutron star coalescence. *Classical and Quantum Gravity*, 33(8):085003, April 2016.

[12] Thibault Damour and Alessandro Nagar. Effective one body description of tidal effects in inspiralling compact binaries. *Physical Review D*, 81(8), Apr 2010.

[13] R. A. D'Inverno. *Introducing Einstein's relativity.* 1992.

[14] Matthew Evans et al. A Horizon Study for Cosmic Explorer: Science, Observatories, and Community. 9 2021.

[15] Éanna É Flanagan and Scott A Hughes. The basics of gravitational wave theory. *New Journal of Physics*, 7:204–204, Sep 2005.

[16] John L. Friedman and Nikolaos Stergioulas. Astrophysical implications of neutron star inspiral and coalescence. *International Journal of Modern Physics D*, 29(11):2041015–632, January 2020.

[17] James M. Lattimer. Neutron stars are gold mines. *International Journal of Modern Physics E*, 26:1740014, January 2017.

[18] Michele Maggiore. *Gravitational Waves. Vol. 1: Theory and Experiments.* Oxford Master Series in Physics. Oxford University Press, 2007.

[19] Anna Puecher, Tim Dietrich, Ka Wa Tsang, Chinmay Kalaghatgi, Soumen Roy, Yoshinta Setyawati, and Chris Van Den Broeck. Unraveling information about supranuclear-dense matter from the complete binary neutron star coalescence process using future gravitational-wave detector networks. *Phys. Rev. D*, 107(12):124009, 2023.

[20] Punturo M., et al. The einstein telescope: a third-generation gravitational wave observatory. *Classical and Quantum Gravity*, 27(19):194002, sep 2010.

[21] Bernard Schutz. *A First Course in General Relativity.* 2009.

[22] Stuart L. Shapiro and Saul A. Teukolsky. *Black Holes, White Dwarfs and Neutron Stars: The Physics of Compact Objects.* 1986.

[23] Nikolaos Stergioulas, Theocharis A. Apostolatos, and José A. Font. Non-linear pulsations in differentially rotating neutron stars: mass-shedding-induced damping and splitting of the fundamental mode. *Monthly Notices of the Royal Astronomical Society*, 352(4):1089–1101, Aug 2004.

[24] Nikolaos Stergioulas, Andreas Bauswein, Kimon Zagkouris, and Hans-Thomas Janka. Gravitational waves and non-axisymmetric oscillation modes in mergers of compact object binaries. *Monthly Notices of the Royal Astronomical Society*, 418(1):427–436, Sep 2011.

[25] Andreas Bauswein Theodoros Soultanis and Nikolaos Stergioulas. Analytic models of the spectral properties of gravitational waves from neutron star merger remnants. *Phys. Rev. D*, 105(4):043020, February 2022.