A Novel Method for Removing the Cyclic Frequency Shift due to the Earth's Motion about the Sun from Astronomical Continuous-Wave Data

Curtis Rau, Colin Gordon, David Witalka, and Samuel Akwei-Sekyere
Michigan State University

This senior thesis is presented to Michigan State University for partial fulfillment of the requirements for a baccalaureate degrees in Physics and Neuroscience. In this paper we will introduce the Gustafson Algorithm for removing frequency/phase modulation form a signal. This method is powerful because it works in the frequency rather than time domain (explain why this is an advantage). Here we will focus on its applications to LIGO, but there are many applications. Some other examples are: -Radio wave astronomy -Comunications systems using FM? -What are some other possible applications??

-we are given time series data. -we are looking for phase modulated gravitational waves. -form of phase modulation depends on the nature of the binary orbit. [5] [2] [1] [3] [4]

DIVISION OF LABOR

Curtis Rau

- organized topics and problems of interest, directed group research
- wrote algorithm and discussion of results
- made figures for presentation and paper
- wrote rough draft and template for paper
- Wrote the C++ implementation

Colin Gordon

- researched and solved phase modulation of astronomical signal caused by rotation of earth using circular orbit approximation
- wrote introduction
- reviewed paper and edited

David Witalka

- created and organized presentation slides
- researched and edited sections on sampling theorem, Nyquist Frequency and aliasing, Carson's rule
- reviewed paper and edited

Samuel Akwei-Sekyere

- reviewed and expanded paper
- researched and wrote sections on error and noise analysis

I. INTRODUCTION

Einstein published his theory of general relativity in 1915. A consequence of his theory was that space time could stretch and contract when acted upon by a gravitational source. It didn't take long then to realize that under the right conditions this stretching and contracting could cause waves through the very fabric of spacetime. Go forward nearly 100 years and the very first evidence for these waves existing is discovered by the LIGO detector. They measured the ripples in space due to two massive black holes inspiralling and finally colliding together in an event that consumed 30 solar masses. This confirmed Einstein's theory and now the field of gravitational astronomy has been born. It is believed that gravitional wave sources are actually fairly common in our universe and sources of continuous monochromatic (one frequency) gravitational radiation are predicted to exist; at least sources whose frequency wonders over many periods []. If they do exist they are apparently very low intensity as viewed from earth because none have been observed to date []. There is one phenomena in particular that may be significantly repressing the amplitude of these signals as seen from Earth. This would be frequency modulation. When a wave is frequency modulated, its power is distrbuted amongst an infinite number of sidebands which are the frequencies measured when taking Fourier Transform of the data that comes from the sinusoidal phase shift due to the Doppler effect from Earth's motion relative to the source. To calculate frequency observed for a gravitationl wave we need to think about the source as an object that is Megaparsecs away (potentially in other galaxies). they are so far that their Sphereical coordinate will not change significantly and the object will move relative to the sun with nearly constant velocity over the time we collect data. This means that the sun will see the monochromatic wave produced by the source as monochromatic. Based on this and that the waves travel at the speed of light can write the wave as a lorentz invariant 4 vector.

We introduce some variables: Γ is the modulation index, ω_e is the angular frequency of the Earth's rotation about the Sun, ω_0 is the frequency of the wave to be detected, ϕ_0 is the phase of the wave to be detected at $t = 0$, ϕ_e is the relative phase difference between the Earth's rotation about the Sun and the wave to be detected. in the equation below we use theta as the polar angle from the axis perpendicular to the orbit of Earth, gamma is the lorentz factor from special relativity. beta is the velocity of Earth's orbit divided by the speed of light. **K** is the four vector of the wave's frequency and wave number.

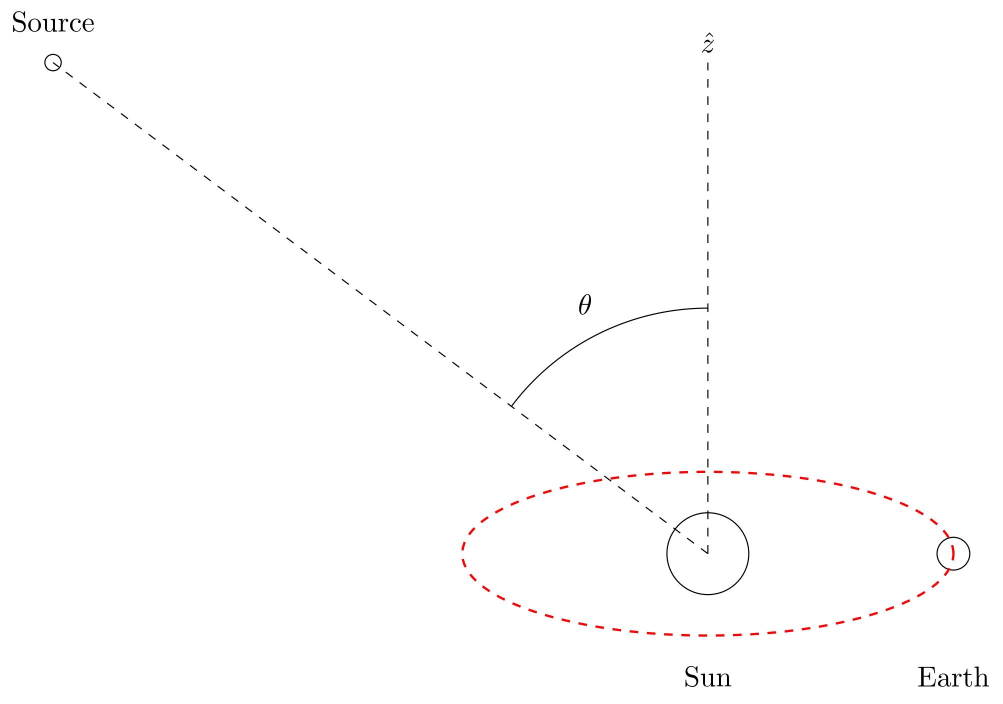


FIG. 1

$$\mathbf{K} = \begin{pmatrix} \gamma & -\gamma\beta & 0 & 0 \\ -\gamma\beta & \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -\sin(\omega_e t + \phi_e) & \cos(\omega_e t + \phi_e) & 0 \\ 0 & -\cos(\omega_e t + \phi_e) & -\sin(\omega_e t + \phi_e) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \omega_0/c \\ -k\sin\theta \\ 0 \\ -k\cos\theta \end{pmatrix} \tag{1}$$

$$= \begin{pmatrix} \gamma\left(\omega_0/c - \beta k \sin\theta \sin(\omega_e t + \phi_e)\right) \\ \gamma\left(-\beta\omega_0/c + k\sin\theta\sin(\omega_e t + \phi_e)\right) \\ \gamma k \sin\theta\cos(\omega_e t + \phi_e) \\ -k\cos\theta \end{pmatrix} \tag{2}$$

So the new instentanious frequency in the earth frame is

$$\omega(t) = \omega_0\gamma\left(1 - \beta\sin\theta\sin(\omega_e t + \phi_e)\right) \tag{3}$$

phase is the time integral of frequency so the expression for the wave we are looking for is the real part of

$$h(t) = h_0 e^{i\left(\omega' t + \Gamma\cos(\omega_e t + \phi_e)\right)} \tag{4}$$

$$\omega' = \gamma\omega_0 \tag{5}$$

$$\Gamma = \frac{\gamma\beta\omega_0\sin\theta}{\omega_e} \tag{6}$$

An important note is that Γ is a function of only the azmuthal angle of the source with respect to the normal of the gallactic plane. This algorithm will also work perfectly well with any frequency modulated wave produced in a matter similiar to the above detections. In this way there could be applications for the Gustafson algorithm in areas outside of gravitational waves in areas such as radio wave astronomy and communications using FM (Frequency Modulated) signals. We attempt to keep the discussion of the algorithm and the results as general as possible for this reason.

II. THE GUSTAFSON ALGORITHM

It would be nice to have an algorithm that does the following:

$$h_0 e^{i(\omega_0 t + \phi_0 + \Gamma\cos(\omega_1 t + \phi_1))} \rightarrow h_0\delta(\omega_0)\delta(\omega_1)\delta(\Gamma) \tag{7}$$

Previous methods have utilized, where f(t) is the data.

$$h_0 e^{i(\omega_0 t + \phi_0 + \Gamma\cos(\omega_1 t + \phi_1))} = f(t) \tag{8}$$

$$h_0 e^{i\phi_0} e^{i\omega_0 t} = f(t) e^{-i\Gamma\cos(\omega_1 t + \phi_1)} \tag{9}$$

$$h_0 e^{i\phi_0}\mathcal{F}_t\left[e^{i\omega_0 t}\right] = \mathcal{F}_t\left[f(t)e^{-i\Gamma\cos(\omega_1 t + \phi_1)}\right] \tag{10}$$

define $F_t[f(t)](\omega) = \hat{f}(\omega)$, and invoking the convolution theorem

$$\frac{h_0 e^{i\phi_0}}{\sqrt{2\pi}}\delta(\omega_c - \omega) = \hat{f}(\omega) \star F_t\left[e^{-i\Gamma\cos(\Omega t + \phi)}\right] \tag{11}$$

invoking the Jacobi-Anger Expansion which has the form []:

$$e^{iz\cos(\theta)} = \sum_{n=-\infty}^{\infty} e^{in\pi/2} J_n(z) e^{in\theta} \tag{12}$$

so equation (2) becomes

$$\frac{h_0 e^{i\phi_0}}{\sqrt{2\pi}}\delta(\omega-\omega_0) = \hat{f}(\omega)\star\mathcal{F}_t\left[\sum_{n=-\infty}^{\infty} e^{in\pi/2}J_n(-\Gamma)e^{in(\omega_1 t+\phi_1)}\right] \tag{13}$$

Bessel Functions have the properties

$$J_n(-z) = (-1)^n J_n(z) \tag{14}$$

so

$$\mathcal{F}_t\left[\sum_{n=-\infty}^{\infty} e^{in\pi/2}J_n(-\Gamma)e^{in(\omega_1 t+\phi_1)}\right] = \sum_{n=-\infty}^{\infty}(-1)^n e^{in\pi/2}e^{in\phi_1}J_n(\Gamma)\mathcal{F}_t\left[e^{in\omega_1 t}\right] \tag{15}$$

$$= \frac{1}{\sqrt{2\pi}}\sum_{n=-\infty}^{\infty} e^{in(\phi_1-\pi/2)}J_n(\Gamma)\delta(\omega-n\omega_1) \tag{16}$$

where we have used $(-1)^n i^n = (-i)^n = e^{-in\pi/2}$, and the linearity of the Fourier Transform $F_t[f+g] = F_t[f]+F_t[g]$. The Fourier transform of this is

$$h_0 e^{i\phi_0}\delta(\omega-\omega_0) = \hat{f}(\omega)\star\left[\sum_{n=-\infty}^{\infty} e^{in(\phi_1-\pi/2)}J_n(\Gamma)\delta(\omega-n\omega_1)\right] \tag{17}$$

so performing the convolution

$$\hat{f}(\omega)\star\delta(\omega-n\omega_1) = \int \hat{f}(\omega-\widetilde{\omega})\delta(\widetilde{\omega}-n\omega_1)d\widetilde{\omega} \tag{18}$$

$$= \hat{f}(\omega-n\omega_1) \tag{19}$$

which brings us to the almost final answer:

$$h_0 e^{i\phi_0}\delta(\omega-\omega_0) = \sum_{n=-\infty}^{\infty} e^{in(\phi_1-\pi/2)}J_n(\Gamma)\hat{f}(\omega-n\omega_1) \tag{20}$$

but for later convenience let's have $n \to -n$. We will use the other property of the Bessel Functions: []

$$J_{-n}(z) = (-1)^n J_n(z) \tag{21}$$

Which brings us to the Gustafson Algorithm (someone should check this out before we submit the paper):

$$h_0 e^{i\phi_0}\delta(\omega-\omega_0) = \sum_{n=-\infty}^{\infty} e^{-in(\phi_1+\pi/2)}J_n(\Gamma)\hat{f}(\omega+n\omega_1) \tag{22}$$

This algorithm has been implemented in the past to demodulate the carrier frequency to recover the true amplitude [**?**].

This algorithm in its current form cannot be used on real data because it assumed a complex waveform. This assumption allowed for a clean demodulation, which is spoiled if one takes the real of this function. This is due to the nonlinearity (is this the correct term) of the real opperator as demonstrated by

$$\Re\{\cdot b\} \neq \Re\{a\}\cdot\Re\{b\} \tag{23}$$

We can use this algorithm as a guess as to find the form that will work with a real valued waveform. Upon comparison of the Fourier transforms of the complex and real waveforms (see appendix A and B) we see they are very similar. The real looks like the complex with an additional sum of mirrored terms (explain what I mean by this).

What would we have to do to get the fourier transform of the real to look exactly like the fourier transform of the complex wave? Unsurprisingly not much because they are such similar functions. We would only have to multiply each term by some phase factor to get the real fourier transform to look like the sum of two complex fourier transforms. After adding in the phase factors we plug the fourier transform of the real wave into the Gustafson Algorithm which will yeald two delta functions now instead of one (because the fourier transform of the real is the sum of two complex wave fourier transforms). One delta function will be at $+\omega_0$, this is the one of interest, and the other will be at $-\omega_0$. (A much more rigerous explanation of this is needed.) If we carry the search out over only positive values of ω_0 then we can throw out all terms in red of the fourier transform of the real wave because they are inconsequential to the algorithm. Finally, the phase terms that were added to the foureier transform of the real to make it look like the complex can be encorperated into the Guatafson Algorithm. The only remaining thing to do is multiply the Gustafson Algorithm by two (explain why).

A. Discussion of The Gustafson Algorithm

It is not explicitly obvious how The Gustafson Algorithm, Eq. 22, can be used as a search algorithm able to determin the values $\omega_0, \omega_1, \phi_0, \phi_1, \Gamma$. In this section we seek to make obvious what Eq. 22 can do.

Now let's assume that the data takes the form

$$f(t) = \Re\left\{h_0 e^{i(\omega_0 t + \phi_0 + \Gamma\cos(\omega_1 t + \phi_1))}\right\} \tag{24}$$

but we do not know the values of $\omega_0, \omega_1, \phi_0, \phi_1, \Gamma$. To determin these values we will try the values $\widetilde{\omega}_0, \widetilde{\omega}_1, \widetilde{\phi}_1, \widetilde{\Gamma}$ in the Gustafson Algorithm

$$? = \sum_{n=-\infty}^{\infty} e^{-in(\widetilde{\phi}_1 + \pi/2)} J_n(\widetilde{\Gamma}) \hat{f}(\omega + n\widetilde{\omega}_1) \tag{25}$$

To see what this will yeald we substitute the Fourier Transform of the real of the waveform equation (B6), into the above equation

$$\begin{aligned} &\frac{h_0}{2} \sum_{m,n=-\infty}^{\infty} e^{i(m\phi_1 - n\widetilde{\phi}_1)} J_n(\widetilde{\Gamma}) J_m(\Gamma) \\ &\qquad \times \left[e^{i(\phi_0 + (m-n)\pi/2)} \delta(\omega - \omega_0 + n\widetilde{\omega}_1 - m\omega_1) + e^{-i(\phi_0 + (m+n)\pi/2)} \delta(\omega + \omega_0 + n\widetilde{\omega}_1 - m\omega_1)\right] \end{aligned} \tag{26}$$

This needs to be treated in cases.

Case 1: Let's consider the most likely case when $\omega \neq \omega_0$ and $\omega_1 \neq \widetilde{\omega}_1$. Let's examin the term $\delta(\omega - \omega_0 + n\widetilde{\omega}_1 - m\omega_1)$ in the above equation in detail. This delta function appears in the double sum and is a function of both m and n, so the $n\widetilde{\omega}_1 - m\omega_1$ is changing over the sums whereas the value $\omega - \omega_0$ is constant (not a function of m or n). This delta is non-zero only if

$$0 = \omega - \omega_0 + n\widetilde{\omega}_1 - m\omega_1 \tag{27}$$

$$\omega_0 - \omega = n\widetilde{\omega}_1 - m\omega_1 \tag{28}$$

This can happen twice or never. By the Euclidean Algorithm, the delta will only be non-zero if

$$\gcd(\widetilde{\omega}_1, \omega_1) = \omega_0 - \omega \tag{29}$$

If this is false then the sum is zero. If it is true then call the values for m and n that make it true $\pm m_o$ and $\mp n_o$. So then the sum is

Placeholder for equation

Case 2: Now consider the case when $\omega = \omega_0$ and $\omega_1 \neq \widetilde{\omega}_1$. The term $\delta(\omega - \omega_0 + n\widetilde{\omega}_1 - m\omega_1)$ will only be non-zero if

$$\widetilde{\omega}_1 = \frac{m}{n}\omega_1 \tag{30}$$

Again this can happen twice or never. If this is false then the sum is zero.

Placeholder for equation

Case 3: Now consider the case when $\omega \neq \omega_0$ and $\omega_1 = \widetilde{\omega}_1$. The term $\delta(\omega - \omega_0 + n\widetilde{\omega}_1 - m\omega_1)$ will only be non-zero if

$$\omega - \omega_0 = (n - m)\widetilde{\omega}_1 \tag{31}$$

Again this can happen twice or never. If this is false then the sum is zero.

Cases 1-3: In all of these cases

Case 4: Now consider the case when $\omega = \omega_0$ and $\omega_1 = \widetilde{\omega}_1$. The term $\delta(\omega - \omega_0 + n\widetilde{\omega}_1 - m\omega_1)$ will only be non-zero if $m = n$ which will happen an infinite number of times. The sum becomes

$$\frac{h_0}{2} e^{i\phi_0} \delta(0) \sum_{n=-\infty}^{\infty} e^{in(\phi_1 - \widetilde{\phi}_1)} J_n(\widetilde{\Gamma}) J_n(\Gamma) \tag{32}$$

Case 4a: Now if we let $\Gamma = \widetilde{\Gamma}$ and $\phi_1 = \widetilde{\phi}_1$ then the sum becomes

$$\frac{h_0}{2} e^{i\phi_0} \delta(0) \sum_{n=-\infty}^{\infty} J_n^2(\Gamma) = \frac{h_0}{2} e^{i\phi_0} \delta(0) \tag{33}$$

So if we guess every search parameter correctly, $\omega = \omega_0$, $\omega_1 = \widetilde{\omega}_1$, $\Gamma = \widetilde{\Gamma}$, and $\phi_1 = \widetilde{\phi}_1$, then we recover the

III. CARSON'S RULE AND THE NYQUIST FREQUENCY

In order too fully reconstruct the carrier frequency we would need to take an infinite sum over the sidebands. As this is impractical and unnecessary as many terms are insignificant, we need to find a way to limit the number of sidebands we are summing over. The first problem is locating the sidebands' characteristic frequency. There are infinite sideband frequencies and eventually they will be higher than the Nyquist frequency which will cause aliasing of the sideband frequency. The Nyquist frequency, ψ_N, is defined as the maximum frequency that can be accurately reconstructed and is equal to half of the sampling frequency, ψ_s.

$$\psi_N = \frac{1}{2}\psi_s \tag{34}$$

Any side band with a frequency of ψ_0 higher than the Nyquist frequency, but less than the sampling frequency, will be reconstructed with a frequency of (ψ_s — ψ_0). That is the frequency will be under-sampled and an alias frequency will be produced. When this occurs the best case scenario is that the alias frequency is unique and does not correspond to the frequency of another lower frequency sideband. In this case we can use anti-aliasing techniques to correct for the aliasing and still obtain information about the amplitude and frequency of the higher side bands. The worst case scenario is that the aliased frequency is too close to another lower frequency sideband and this will cause two problems: a systematic error will be added to the measurement of the lower side band's amplitude, and we will be unable to determine the necessary information on what the amplitude of the higher frequency would be. To minimize the effect this might have we would like any sidebands with a frequency over the Nyquist frequency to have as low an amplitude as possible. For this we can employ Carson's rule. Carson's rule can be understood to say that almost all (roughly 98 percent) of the power for a frequency-modulated sinusoidal signal is contained within a finite bandwidth B_T, defined by:

$$B_T = 2(\Delta\psi + \psi_m) \tag{35}$$

where $\Delta\psi$ is the peak frequency deviation of the instantaneous frequency $\psi(t)$ from the center carrier frequency ψ_c, and ψ_m is the highest frequency in the modulating signal. In our case the highest frequency of the modulating signal is the carrier frequency, ψ_c. From this calculation we can adjust our sampling frequency to ensure it is large enough in order to minimize aliasing. This also puts a limit on the number of sidebands we need to sum over to just those that fall within the bandwidth. The important thing to take away from this is that it takes an infinite bandwidth to prefectly transmit a phase or frequency modulated wave, regardless of how smooth it is, but in practice only a finite bandwidth is needed for an accurate approximation.

A. Consequences of Discretizing

-drichlet kernel

IV. IMPLEMENTING THE GUSTAFSON ALGORITHM IN C++

A. Calculating Bessel Functions

Known problems with the algorithm include calculating large order bessel functions. The bessel functions of natural number order are given by

$$J_n(z) = \sum_{m=0}^{\infty} \frac{(-1)^n}{m!(m+n)!} \left(\frac{z}{2}\right)^{2m+n} \tag{36}$$

so calculating the nth Bessel function involves calculating factorials greater than n!. The highest n for which we can store n! in an unsigned long integer type, the largest positive integer type, is 20:

$$\begin{aligned} 2^{64} - 1 &= 18,446,744,073,709,551,615 \qquad (37) \\ < 21! &= 51,090,942,171,709,440,000 \qquad (38) \end{aligned}$$

if we are willing to sacrafise some precision to truncation we can use a double precision floating point integer which allows values upto $1.797,693,134,862,315,7 \cdot 10^{308}$

$$170! \approx 7.25 \cdot 10^{306} < 1.797,693,134,862,315,7 \cdot 10^{308} < 171! \approx 1.24 \cdot 10^{309} \tag{39}$$

This is not acceptable hovever, because $|J_n(z)| \leq 1$, so when performing the sum we need precision all the way down to at the very least 0.1, which would require a mantissa with roughly 308 digits. This is not only impracticle, but it is by far one of the slowest ways one could calculate values of the Bessel functions. This whole buiseness of wresteling with factorials can be avoided by calculating the Bessel Functions using Bessel's Equation 40 [3].

$$x^2 J_n''(x) + x J_n'(x) + (x^2 - n^2) J_n(x) = 0 \tag{40}$$

The standard perscription for numerically solving differential equations is first to discretize the independent variable $x \to x_i = x_{min} + i \cdot \delta x$. Then use the limit definitions of the derivatives; we will use the three point definitions because their error goes as $O(\delta x^2)$ rather than using the two point definition with an error that goes as $O(\delta x)$ [**?**]. Also, because there are only one order of Bessel Functions in Bessel's Equation, the order is implied by the n that appears, so we will drop the subscript of n, $J_n \to J$, and we will further adopt the notation $J(x_i) = J_i$.

$$J_i' = \frac{J_{i+1} - J_{i-1}}{2\delta x} \tag{41}$$

$$J_i'' = \frac{J_{i-1} - 2J_i + J_{i+1}}{\delta x^2} \tag{42}$$

So the discretized form of Bessel's Equation is

$$x_i^2 \frac{J_{i-1} - 2J_i + J_{i+1}}{\delta x^2} + x \frac{J_{i+1} - J_{i-1}}{2\delta x} + (x^2 - n^2) J_i = 0 \tag{43}$$

$$\left(\frac{x_i^2}{\delta x^2} - \frac{x_i}{2\delta x}\right) J_{i-1} + \left(\left(1 - \frac{2}{\delta x^2}\right) x_i^2 - n^2\right) J_i + \left(\frac{x_i^2}{\delta x^2} + \frac{x_i}{2\delta x}\right) J_{i+1} = 0 \tag{44}$$

Where we have collected like terms of J_i. For convenience we define coefficients to simplify the above equation.

$$a_i = \frac{x_i^2}{\delta x^2} - \frac{x_i}{2\delta x} \tag{45}$$

$$b_i = \left(1 - \frac{2}{\delta x^2}\right) x_i^2 - n^2 \tag{46}$$

$$c_i = \frac{x_i^2}{\delta x^2} + \frac{x_i}{2\delta x} \tag{47}$$

These coefficients can be simplified by using the definition for x_i, and taking $x_{min} = 0$.

$$a_i = i\left(i - \frac{1}{2}\right) \tag{48}$$

$$b_i = i^2\left(\delta x^2 - 2\right) - n^2 \tag{49}$$

$$c_i = i\left(i + \frac{1}{2}\right) \tag{50}$$

So the discretized Bessel equation with simplified coefficients is

$$a_i J_{i-1} + b_i J_i + c_i J_{i+1} = 0 \tag{51}$$

Wrighting out the system of equations explicitly:

$$b_1 J_1 + c_1 J_2 = -a_1 J_0 \tag{52}$$

$$a_2 J_1 + b_2 J_2 + c_2 J_3 = 0 \tag{53}$$

$$a_3 J_2 + b_3 J_3 + c_3 J_4 = 0 \tag{54}$$

$$\vdots \tag{55}$$

$$a_{n-1} J_{n-2} + b_{n-1} J_{n-1} + c_{n-1} J_n = 0 \tag{56}$$

$$a_n J_{n-1} + b_n J_n = -c_n J_{n+1} \tag{57}$$

Which suggests Bessel Equation be written as a matrix equation.

$$\begin{pmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & a_3 & b_3 & c_3 & \\ & & \vdots & \vdots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{pmatrix} \begin{pmatrix} J_1 \\ J_2 \\ J_3 \\ \vdots \\ J_{n-1} \\ J_n \end{pmatrix} = \begin{pmatrix} -a_1 J_0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ -c_n J_{n+1} \end{pmatrix} \tag{58}$$

This is a tridiagonal matrix system that charactorized a one dimentional differential equation. It can be easilly solved using the standard Thomas Algorithm. To program this in C++ we need all the arrays to be indexed from 0, so

$$\begin{pmatrix} B_0 & C_0 & & & \\ A_1 & B_1 & C_1 & & \\ & A_2 & B_2 & C_2 & \\ & & \vdots & \vdots & \\ & & A_{N-4} & B_{N-4} & C_{N-4} \\ & & & A_{N-3} & B_{N-3} \end{pmatrix} \begin{pmatrix} J_1 \\ J_2 \\ J_3 \\ \vdots \\ J_{N-3} \\ J_{N-2} \end{pmatrix} = \begin{pmatrix} -A_0 J_0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ -C_{N-3} J_{N-1} \end{pmatrix} \tag{59}$$

$$A_i = a_{i+1} = (i+1)\left(i+\frac{1}{2}\right) \tag{60}$$

$$B_i = b_{i+1} = (i+1)^2\left(\delta x^2 - 2\right) - n^2 \tag{61}$$

$$C_i = c_{i+1} = (i+1)\left(i+\frac{3}{2}\right) \tag{62}$$

1. The Thomas Algorithm

The Thomas Algorithm is a lightning fast way to solve tridiagonal systems like equation 59. The algorithm consists of two steps. First there is a forward substitution which eliminates the A_i's, and modifies the B_i's and the vector on the right which we will call the source vector S_i; $\quad \forall i \in [0, N-3]$.

$$R_i = R_i - \frac{A_i}{B_{i-1}} R_{i-1}; \qquad \forall i \in [1, N-3] \tag{63}$$

$$B_i \to B_i - \frac{A_i}{B_{i-1}} C_{i-1}; \qquad \forall i \in [1, N-3] \tag{64}$$

$$S_i \to S_i - \frac{A_i}{B_{i-1}} S_{i-1}; \qquad \forall i \in [1, N-3] \tag{65}$$

Then there is a backwards substitution eliminating the C_i's, again modifying the S_i's. Finally we solve for J_i.

$$R_i = R_i - \frac{C_i}{B_{i+1}} R_{i+1}; \qquad \forall i \in [N-4, 0] \tag{66}$$

$$S_i \to S_i - \frac{C_i}{B_{i+1}} S_{i+1}; \qquad \forall i \in [N-4, 0] \tag{67}$$

$$J_i = \frac{S_{i-1}}{B_{i-1}}; \qquad \forall i \in [N-2, 1] \tag{68}$$

The time it takes to perform the Thomas Algorithm goes as N [].

2. Boundary Conditions

To solve this second-order differential equation we will need two boundary conditions. The boundary condition at $x = 0$ is simple.

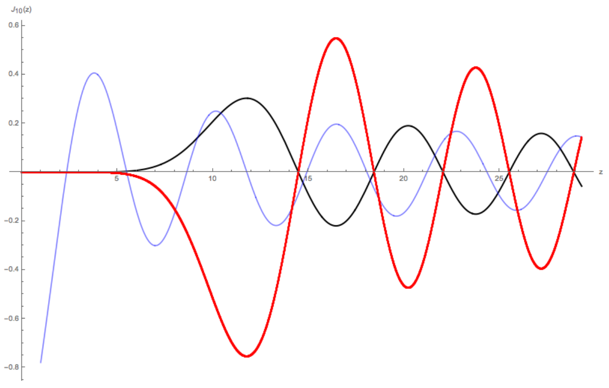
$$J_n(0) = \begin{cases} 1 & : n = 0 \\ 0 & : n \neq 0 \end{cases}$$

For the other boundary condition we *could* use the asymtotic form of the Bessel functions. Theorem 5.1 in [3] states:

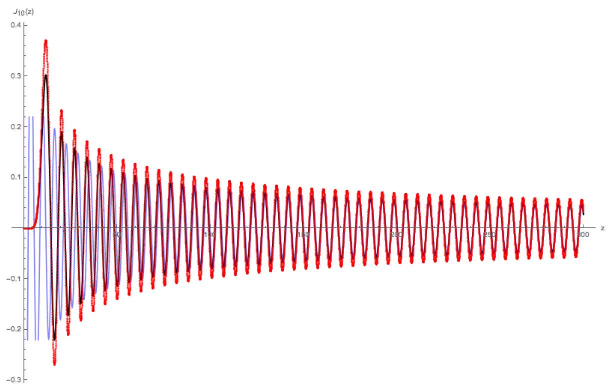
For each $n \in N$ there is a constant $C_n \in R$ such that, if $x \geq 1$, then

$$\left| J_n(x) - \sqrt{\frac{2}{\pi x}} \cos\left(x - \frac{\pi}{4}(2n+1)\right) \right| \leq \frac{C_n}{x^{3/2}} \tag{69}$$

This is an asymptotic expansion, so it is only valid for large values of x. It turns out the accuracy of the solution to bessels equation relies heavilly on the accuracy of the endpoint at $x \neq 0$. Figures 2a and 2b illistrate this point quite well. The conclusion is that the asymtotic form of the bessel functions is not accurate enough for even fairly large valuse of z. We can also do



(a) Notice how the asymtotic form and the one calculated by our differential equation solver match up at the maximum value for z, but do not match up with the true solution. This has far reaching consequences. Whatever multiplicative factor the endpoint is off by seems to carry through for every other point. In this case the multiplicative factor (asymtotic / true) is almost exactly -2.5. Notice how this inverts the solution and makes it 250% larger.



(b) For much larger values of z the asymtotic form is a better approximation to the true value. In this case the multiplicative factor (asymtotic / true evaluated at $z = 300$) is roughly 1.2 which results in the soultion being roughly 20% larger than it should be.

FIG. 2: The blue curve is the asymtotic form of the bessel functions as given in equation 69, the black curve is $J_{10}(x)$ as given by Mathematica, and the red curve is the one calculated by our differential equation solver. Note, both of these solutions were calculated using 10,000 points. Even though the solution in figure 2a has 10 times the point density, it is limited in accuracy by the endpoint!

$$J_n(z) = \frac{1}{\pi}\int_0^\pi \cos\left[z\sin\theta - n\theta\right] d\theta \tag{70}$$

Which is computed using a Riemann Sum

$$J_n(z) \approx \frac{1}{N}\sum_{i=0}^{N+1} \cos\left[z\sin(i\cdot dx) - n\cdot i\cdot dx\right]; \qquad dx = \frac{\pi}{N} \tag{71}$$

The function being integrated becomes more oscillitory as n increases. Numerical analysis shows that n gives the number of zeros the function has on the interval $(0, \pi)$ for $z = 0$. For $n = 0$ the number of zeros is given by $Floor(\frac{2z}{\pi})$. We want a way to ensure the calculation is accurate regardless of how oscillatory it is, but we also want it to be as fast as possible, so we use these facts regarding the zeros to ensure there are roughly the same number of integration points between zeros. By taking the number of (evenly spaced) integration points to be

$$N = 100\cdot(n+1)\cdot Floor\left(\frac{2z}{\pi}+1\right) \tag{72}$$

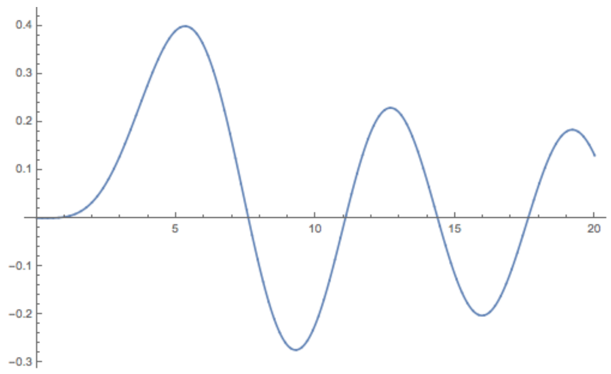
we acheved excellent results (accurate consistantly to 16 decimal places) for a wide range of z and n. The $+1$'s are to ensure accurate results when z or n are zero. The algorithm can have problems when J is extremely small. Then the function can be off by many orders of magnitude, but because it is essentially zero, it does not seem to effect our results when used as a boundary condition for solving Bessel's Differential Equation.

3. *Calculating Bessel Functions: Results*

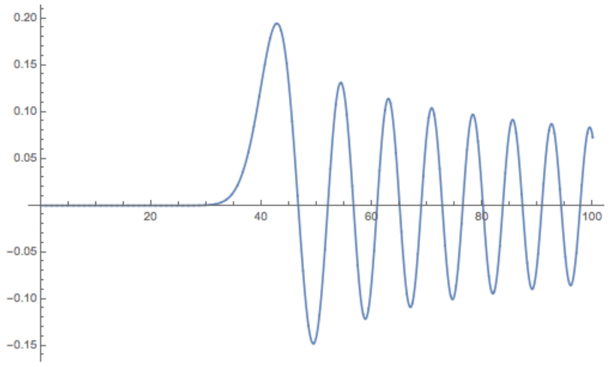
The ultimate benchmark for sucess in calculating Bessel Functions in this manner (using the Thomas Algorithm to solve Bessel's Differential Equation) is to compare it to the stardard way of calculating the value at each point using the Riemann Sum (as defined in equation 71 section IV A 2). So how much faster is it? This depends on the values of n and z because we use a dynamic number of integration points as given by eq. 72. Table I gives the speed increse for various parameters.

Parameters	Point by Point Using Integral	Thomas Algo. Solving Bessel's Equation	Speed Increse
n = 3	0.051786 s	0.00013800 s	375
N = 500			
$z_{max} = 20.0$			
n = 14	113.33 s	0.003937 s	28,785
N = 50,000			
$z_{max} = 100.0$			

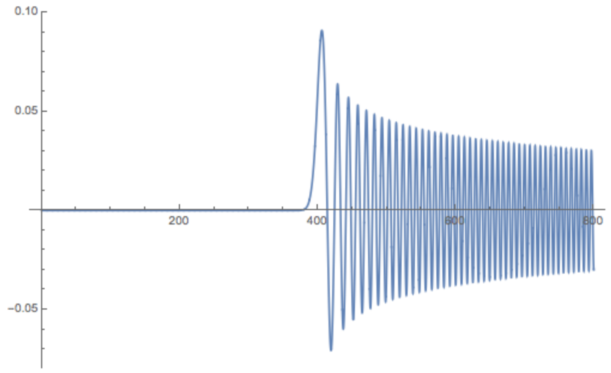
TABLE I: N is the number of points returned, n is the order of the bessel function, and the points calculated were for $z \in [0, z_{max}]$



(a) n=4, numPts=10,000, xMax=20, comptime=0.00082



(b) n=40, numPts=10,000, xMax=100, comptime=0.00092



(c) n=400, numPts=10,000, xMax=800, comptime=0.00200; About half the computation time went to calculating the endpoint!

FIG. 3

Table 3 gives three examples of Bessel Functions that were calculated using the Thomas Algorithm to solve Bessel's Equation.

V. CONCLUSION

Recap the benifits of working in the frequency domain vs the time domain. It is the belief of this author that the methods outlined in this paper are unlikely to be useful for searching for gravitational waves.

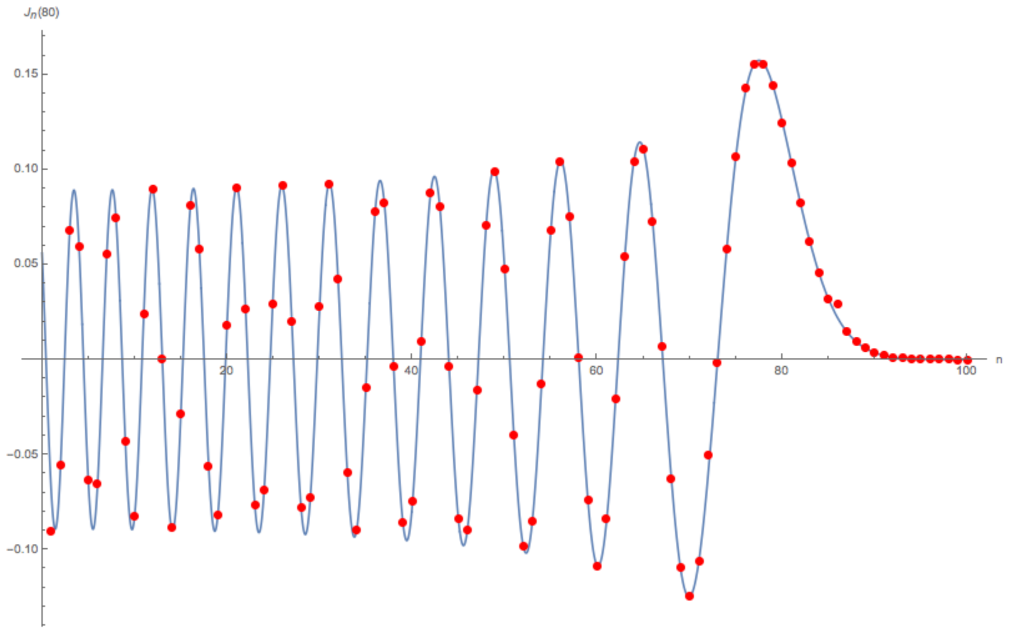


FIG. 4: This plot is a plot of $J_n(80)$ for values of $n \in (0, 100)$. The blue curve was generated by the BesselJ function of Mathematica. The red dots were calculated by my code. The two sets of data are in excelent agreement at except perhaps $n = 0, 85$ in which case there is a slight disagreement. This is a small selection of the $400, 000$ data points my code generated in 0.33 seconds (on 5 year old a loptop). It solved the Bessel differential equation for 4000 steps in $\Gamma \in (0, 100)$, then saves the output, and moved on to solve Bessel's Equation for the next value of n until all values of Γ and n were solved for.

Appendix A: The Fourier Transform of The Complex Function

I BELIEVE I FORGOT SOME FACTORS OF 2PI HERE!! Use the convolution theorem Use the Jacobi-Anger expansion 12 Use the linearity of the Fourier Transform Use the linearity of the Convolution

$$
\begin{aligned}
\mathcal{F}_t\left[h_0 e^{i(\omega_0 t+\phi_0+\Gamma\cos(\omega_1 t+\phi_1))}\right] &= h_0 e^{i\phi_0}\mathcal{F}_t\left[e^{i\omega_0 t}e^{i\Gamma\cos(\omega_1 t+\phi_1)}\right] && \text{(A1)}\\
&= h_0 e^{i\phi_0}\mathcal{F}_t\left[e^{i\omega_0 t}\right]\star\mathcal{F}_t\left[e^{i\Gamma\cos(\omega_1 t+\phi_1)}\right] && \text{(A2)}\\
&= h_0 e^{i\phi_0}\delta(\omega-\omega_0)\star\mathcal{F}_t\left[\sum_{n=-\infty}^{\infty} e^{in\pi/2}J_n(\Gamma)e^{in(\omega_1 t+\phi_1)}\right] && \text{(A3)}\\
&= h_0 e^{i\phi_0}\delta(\omega-\omega_0)\star\sum_{n=-\infty}^{\infty} e^{in\pi/2}e^{in\phi_1}J_n(\Gamma)F_t\left[e^{in\omega_1 t}\right] && \text{(A4)}\\
&= h_0 e^{i\phi_0}\delta(\omega-\omega_0)\star\sum_{n=-\infty}^{\infty} e^{in(\phi_1+\pi/2)}J_n(\Gamma)\delta(\omega-n\omega_1) && \text{(A5)}\\
&= h_0 e^{i\phi_0}\sum_{n=-\infty}^{\infty} e^{in(\phi_1+\pi/2)}J_n(\Gamma)\delta(\omega-\omega_0)\star\delta(\omega-n\omega_1) && \text{(A6)}\\
&= h_0 e^{i\phi_0}\sum_{n=-\infty}^{\infty} e^{in(\phi_1+\pi/2)}J_n(\Gamma)\delta(\omega-\omega_0-n\omega_1) && \text{(A7)}
\end{aligned}
$$

Appendix B: The Fourier Transform of The Real-Valued Function

$$\mathcal{F}_t\left[\Re\left\{h_0 e^{i(\omega_0 t+\phi_0+\Gamma\cos(\omega_1 t+\phi_1))}\right\}\right] = \frac{1}{2}\mathcal{F}_t\left[h_0 e^{i(\omega_0 t+\phi_0+\Gamma\cos(\omega_1 t+\phi_1))} + h_0 e^{-i(\omega_0 t+\phi_0+\Gamma\cos(\omega_1 t+\phi_1))}\right] \tag{B1}$$

$$= \frac{h_0}{2}\left[e^{i\phi_0}\mathcal{F}_t\left[e^{i(\omega_0 t+\Gamma\cos(\omega_1 t+\phi_1))}\right] + e^{-i\phi_0}\mathcal{F}_t\left[e^{i(-\omega_0 t-\Gamma\cos(\omega_1 t+\phi_1))}\right]\right] \tag{B2}$$

From the previous section we see that

$$\mathcal{F}_t\left[e^{i(\omega_0 t+\Gamma\cos(\omega_1 t+\phi_1))}\right] = \sum_{n=-\infty}^{\infty} e^{in(\phi_1+\pi/2)} J_n(\Gamma)\delta(\omega-\omega_0-n\omega_1) \tag{B3}$$

and by replacing ω_0 with $-\omega_0$ and Γ with $-\Gamma$, and using the fact that $J_n(-\Gamma) = (-1)^n J_n(\Gamma)$ we also have

$$\mathcal{F}_t\left[e^{i(-\omega_0 t-\Gamma\cos(\omega_1 t+\phi_1))}\right] = \sum_{n=-\infty}^{\infty} e^{in(\phi_1-\pi/2)} J_n(\Gamma)\delta(\omega+\omega_0-n\omega_1) \tag{B4}$$

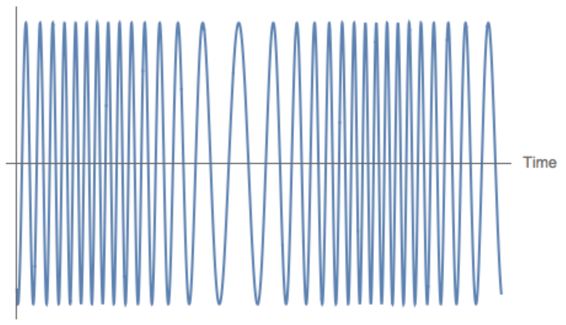
so the Fourier Transform of the real part of our frequency/phase modulated signal is

$$\begin{aligned}\mathcal{F}_t\left[\Re\left\{h_0 e^{i(\omega_0 t+\phi_0+\Gamma\cos(\omega_1 t+\phi_1))}\right\}\right] &= \frac{1}{2}h_0\left[e^{i\phi_0}\sum_{n=-\infty}^{\infty} e^{in(\phi_1+\pi/2)} J_n(\Gamma)\delta(\omega-\omega_0-n\omega_1)\right.\\ &\qquad\left. + e^{-i\phi_0}\sum_{n=-\infty}^{\infty} e^{in(\phi_1-\pi/2)} J_n(\Gamma)\delta(\omega+\omega_0-n\omega_1)\right]\\ &= \frac{1}{2}h_0\sum_{n=-\infty}^{\infty} e^{in\phi_1} J_n(\Gamma)\left[e^{i(\phi_0+n\pi/2)}\delta(\omega-\omega_0-n\omega_1)\right.\\ &\qquad\left. + e^{-i(\phi_0+n\pi/2)}\delta(\omega+\omega_0-n\omega_1)\right]\end{aligned} \tag{B5}$$

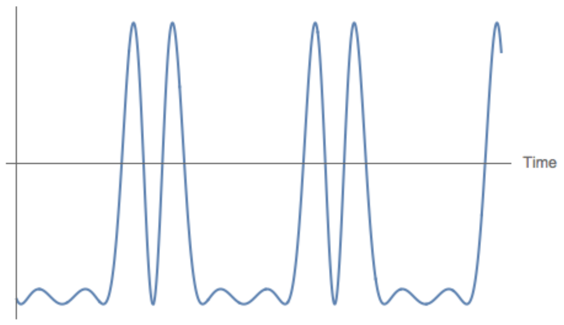
$$\hat{f}(\omega) = \frac{1}{2}h_0\sum_{n=-\infty}^{\infty} e^{in\phi_1} J_n(\Gamma)\left[e^{i(\phi_0+n\pi/2)}\delta(\omega-\omega_0-n\omega_1) + e^{-i(\phi_0+n\pi/2)}\delta(\omega+\omega_0-n\omega_1)\right] \tag{B6}$$

Notice that, by taking the fourier transform of just the real part of our frequency modulated function it has added the terms in red. (Go into more detail why that is) The other important change is that the X_n and Y_n terms were added (Again, go into more detail why that is).

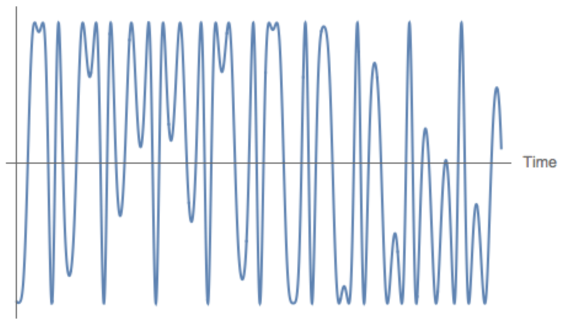
Appendix C: Selected Waveforms



(a) A "nice" signal that is what comes to mind for cyclic frequency modulated signals. $\Gamma = 9.88, \omega_0 = 3.62, \omega_1 = 0.21, \phi_1 = 3.14$



(b) One example of the exotic signals cyclic frequency modulation can produce. $\Gamma = 2.88, \omega_0 = 1.5, \omega_1 = 0.75, \phi_1 = 0$



(c) A particularly nasty waveform that could be mistaken as noise by looking at it. $\Gamma = 3, \omega_0 = 0.42, \omega_1 = 0.46, \phi_1 = 0.1$

[1] Deanna Emery. A coherent three dimensional fft based search scheme for gravitational waves from binary neurton star systems. *LIGO DCC*, (T1500307-v2), September 2015.

[2] B.P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration). Observation of gravitational waves from a binary black hole merger. *Physical Review Letters*, 116(061102):1–16, 12 February 2016.

[3] Gerald B. Folland. *Fourier Analysis and its Applications*. Number ISBN 0-534-17097-3. Brooks and Cole Publishing Company, 1992.

[4] David J. Griffiths. *Introduction to Quantum Mechanics*. Number ISBN 0-13-111892-7. Pearson, second edition edition, 2005.

[5] Peter R. Saulson. *Fundamentals of Interferometric Gravitational Wave Detectors*. World Scientific Publishing Company, March 31, 1994.