Exploring the use of Random Projections for Gravitational Wave Data Analysis

A Thesis

submitted to
Indian Institute of Science Education and Research Pune
in partial fulfillment of the requirements for the
BS-MS Dual Degree Programme

by

Sumeet Kulkarni
20121054

Indian Institute of Science Education and Research Pune
Dr. Homi Bhabha Road,
Pashan, Pune 411008, INDIA.

Submitted on March 30, 2017

Supervisor: **Dr. Sukanta Bose**

Certificate

This is to certify that this dissertation entitled **Exploring the use of Random Projections for Gravitational Wave Data Analysis** towards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents work carried out by **Sumeet Kulkarni** at the Inter-University Centre for Astronomy and Astrophysics (IUCAA), Pune under the supervision of **Dr. Sukanta Bose**, Professor, Inter-University Centre for Astronomy and Astrophysics (IUCAA), Pune , during the academic year 2016-2017.

Dr. Sukanta Bose

Committee:

Dr. Sukanta Bose

Dr. Prasad Subramanian

This thesis is dedicated to my Grandfather, late Prof. V.R. Kulkarni

Declaration

I hereby declare that the matter embodied in the report entitled **Exploring the use of Random Projections for Gravitational Wave Data Analysis** are the results of the work carried out by me at the Inter-University Centre for Astronomy and Astrophysics (IUCAA), Pune, under the supervision of **Dr. Sukanta Bose** and the same has not been submitted elsewhere for any other degree.

Sumeet Kulkarni

Acknowledgments

I am indebted to several people for helping me toward the fruition of this project over the past year. I am grateful to Prof. Sukanta Bose for giving me this opportunity and being an excellent mentor and teacher throughout the course of this project. I am thankful to Dr. Dilip Krisnaswamy from IBM Research, Bangalore for providing his expertise on Random Projection through our detailed discussions and email exchanges. I also thank Dr. Prasad Subramanian for guiding me as my TAC member.

I thank the Inter-University Centre for Astronomy and Astrophysics (IUCAA), Pune for taking me under its shade and providing an incredible academic atmosphere conducive to my research. I am heavily indebted to Khun Sang Phukon (PhD student, IIT Kanpur and visiting scholar at IUCAA) for patiently producing sets of templates for me to work with, teaching me how to run jobs on the cluster and providing valuable inputs through discussions on my work. I am thankful to all members of the Gravitational Waves group at IUCAA, in particular my former guide Dr. Varun Bhalerao, my batchmate Varun Srivastav, graduate students Bhooshan Gadre, Javed Rana, Chaitanya Afle, Anirban Ain, and postdoctoral scholar Anuradha Gupta for patiently answering all my queries and expanding my knowledge of this field over dozens of group meetings and Tea sessions. I thank Dr. Jayanti Prasad for letting me use the resources of Sarathi, the IUCAA LDAS cluster for my computational work.

I am really thankful to my parents for their loving support. I thank all my friends at IISER, especially my office-mates in the physics room for drowning all the highs and lows of the past year together with freshly brewed coffee. I thank the Institute for providing me with invaluable opportunities throughout to participate in research projects, including this one as a part of my undergraduate study and the Department of Science and Technology (DST) for funding it.

Abstract

The first observation run of advanced LIGO returned a surprising 3 detections of coalescing Binary Black Hole (BBH) systems. Having shown much promise as strong Gravitational Wave candidates lying within the aLIGO detector sensitivity, efforts are now being directed at fine-tuning our searches to detect more Compact Binary Coalescence (CBC) sources, especially ones involving Neutron Stars. A combination of algorithmic and software speed-up strategies have been explored for achieving a low-latency detection of signals from these systems, to generate timely alerts for Electromagnetic Follow-up observations. In this study, we investigate another mathematical technique, called Random Projection,which guarantees the preservation of information in high-dimensional data structures under projection to a lower dimension following the Johnson-Lindenstrauss lemma. We explore the applicability of Random Projections for reducing Gravitational Wave templates in order to speed up the computation of matched filtering in the time domain.

Contents

Preliminaries

Definition 0.0.1. *The Euclidean Distance between two d-dimensional vectors $u, v \in R^d$, also known as the $l2$-norm, is defined as:*

$$\| u - v \|_2^2 = \sqrt{\sum_{i=0}^{d} (u_i^2 - v_i^2)}$$

Definition 0.0.2. *The Inner Product between two d-dimensional vectors $u, v \in R^d$ is defined as:*

$$u \cdot v = \sum_{i=0}^{d} (u_i v_i)$$

Chapter 1

Introduction

September 14, 2015. 09:50:45 UTC. The twin LIGO (Laser Interderometer Gravitational-Wave Observatory) detectors at Livingston, LA and Hanford, WA were still in their sixth engineering run when they picked up an identical loud signal, 7 milliseconds apart. Months of intensive data screening, testing and analysis finally confirmed the two chirping sinuosoidal signals (depicted in the adjacent figure[3]) to have come from an astrophysical source: the merger of two black holes, $36M_\odot$ and $29M_\odot$ in mass into a single black hole having a mass of $62M_\odot$. The deficit of almost $3M_\odot$ was released as energy in the form of Gravitational Waves. This first confirmed detection of Gravitational Waves was christened GW150914, marking the date and indicating that many more were to follow. The era of Gravitational Wave astronomy had begun. In this study, we shall investigate mathematical methods of data analysis to try speeding up our searches for Gravitational Wave signals extracted from the tonnes of noisy data collected by LIGO.

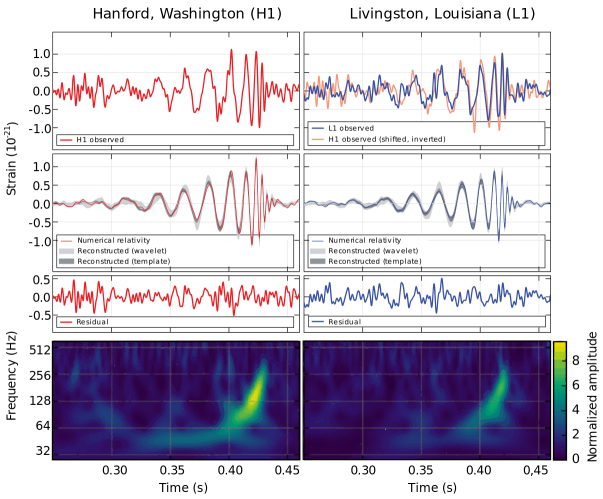


Figure 1.1: GW150914: Signals from the first ever Gravitational Wave Detection

1.1 Gravitational Wave Data Analysis

Compact Binary Coalescences (CBCs) involving pairs of Neutron Stars or Black Holes are the most promising sources of Gravitational waves detectable within the sensitivities of advanced LIGO. The most popular technique for finding CBC Gravitational Wave signals in LIGO data is *matched filtering*. This involves a noise-weighted correlation of the detector data with a set of theoretically modelled *templates*, which characterise the signal from a binary system in terms of its strain amplitude $\mathcal{A}(t)$ and phase $\phi(t)$ evolution.

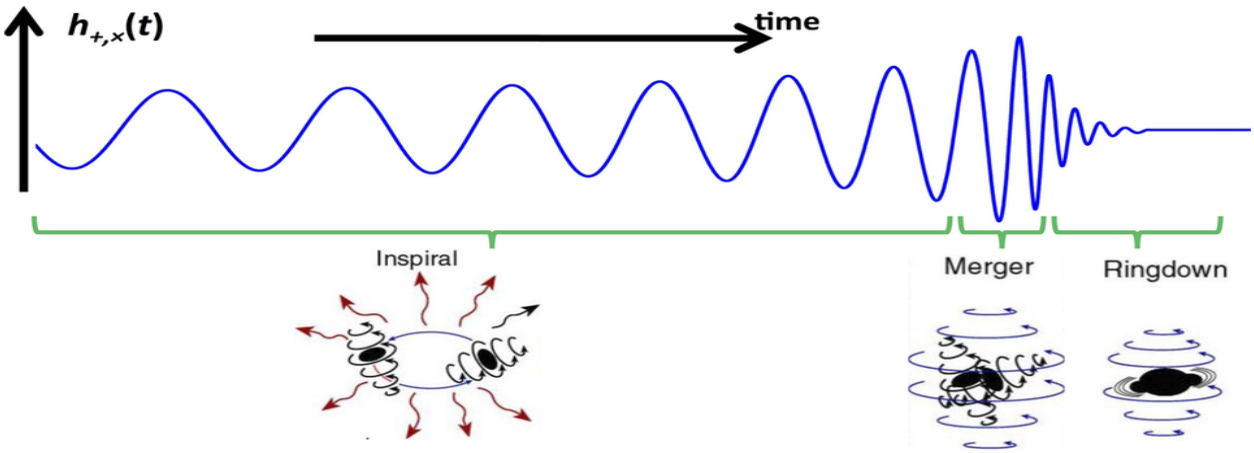


Figure 1.2: A Gravitational Wave template [2]

As illustrated in Figure 1.2, the Gravitational Wave templates of CBCs are *chirping sinusoids*. The strain $h(t)$ frequency and amplitude increases in frequency as the binary components revolve around each other faster in successively shorter orbits, radiating energy in the form of Gravitational Waves. This is known as the *inspiral* phase. As we shall see in chapter 4, this portion is modelled using the Post-Newtonian expansion of the perturbative Einstein field equations to find an approximate solution. The waveform frequency reaches its maximum close to the *merger* phase, when the components collide. This portion is harder to model and requires the use of Numerical Relativity. Post-merger, the system rests down in the *ringdown* phase, which can be modelled using the Quasi-Normal modes of black holes.

Gravitational Wave signals come in two independent polarisations: the *plus* polarisation (h_+) and the *cross* polarisation ($h_\times$), inclined at an angle of $\pi/4$ with respect to each other. The waveform observed at a detector can be expressed as[14]:

$$h(t) = F_+(\theta, \phi, \psi)h_+(t;\xi) + F_\times(\theta, \phi, \psi)h_\times(t;\xi) \tag{1.1}$$

Where F_+ and $F_\times$ are the *detector response functions*:

$$F_+(\theta,\phi,\psi) = -\frac{1}{2}(1+\cos^2\theta)\cos 2\phi\cos 2\psi - \cos\theta\sin 2\phi\sin 2\psi \tag{1.2}$$

$$F_\times(\theta,\phi,\psi) = \frac{1}{2}(1+\cos^2\theta)\cos 2\phi\sin 2\psi - \cos\theta\sin 2\phi\cos 2\psi \tag{1.3}$$

These functions depend on the sky coordinates of the binary system (θ,ϕ) and the polarisation (ψ). They characterise the strength of an incoming signal relative to the detector orientation. The LIGO interferometers have the maximum sensitivity to detect signals that arrive perpendicular to their face, and specifically for polarisations along the lengths of the arms. These 3 parameters are *extrinsic* to the modelling of the templates, being independent of the binary system in question. Two additional extrinsic parameters are the *time of arrival*, (t_a) and the *initial phase* (ϕ_0) of the binary. The functional form of the two polarisation phases is in Eqn(1.1) is given by:[6]

$$h_+(t) = -\frac{(1+\cos^2\iota)}{2}\left(\frac{G\mathcal{M}}{c^2 D}\right)\left(\frac{t_a - t}{5G\mathcal{M}/c^3}\right)^{-1/4}\cos\left[2\phi_0 + 2\phi(t-t_a;M,\mu)\right] \tag{1.4}$$

$$h_\times(t) = (-\cos\iota)\left(\frac{G\mathcal{M}}{c^2 D}\right)\left(\frac{t_a - t}{5G\mathcal{M}/c^3}\right)^{-1/4}\sin\left[2\phi_0 + 2\phi(t-t_a;M,\mu)\right] \tag{1.5}$$

Where G is the Gravitational constant and c is the velocity of light. D, $\mathcal{M}$, and μ constitute the *intrinsic* parameters of the binary. D represents the distance to the source. If $m1$ and $m2$ are the two component masses, the following parameters are determined from them: The *total mass* $(M) = m1 + m2$, the *reduced mass* $\mu = (m1m2)/M$, the *symmetric mass ratio* $\eta = \mu/M$ and the *chirp mass* $\mathcal{M} = \mu^{3/5}M^{2/5} = \eta^{3/5}M$.

The antenna response functions from Eqn.(1.2)(1.3) can be treated as constant over the time from which the signal starts arriving until the coalescence time.[6]. The incoming

inspiral strain can thus be written as:

$$h(t) = -\left(\frac{G\mathcal{M}}{c^2 D_{eff}}\right)\left(\frac{t_a - t}{5G\mathcal{M}/c^3}\right)^{1/4} \cos\left[2\phi_0 + 2\phi(t - t_a; M, \mu)\right] \tag{1.6}$$

We can represent the detector data stream when a signal $(s(t))$, assumed to be similar to a template, $(h(t))$ is present as:

$$x(t) = s(t) + n(t)$$

where $n(t)$ represents the noise. The data in the Fourier domain is given by:

$$\tilde{x}(f) = \int_{-\infty}^{\infty} e^{i2\pi ft} x(t) dt \tag{1.7}$$

We use **Matched Filtering** to extract the signal $s(t)$ from the data $x(t)$. The match filter output is defined as:

$$y = \langle x|h\rangle = \langle s|h\rangle + \langle n|h\rangle \tag{1.8}$$

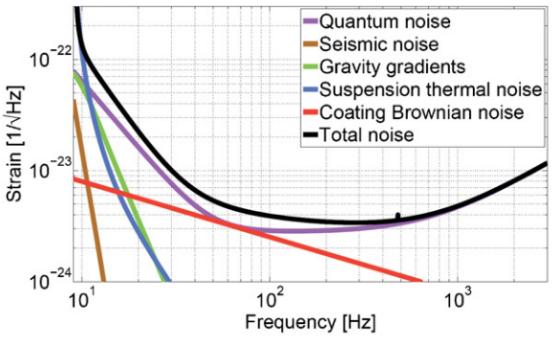
Here, the inner product is defined as:

$$\langle a|b\rangle = 4\mathrm{Re}\int_0^{\infty} \frac{\tilde{a}^*(f)\tilde{b}(f)}{S_h(f)} \tag{1.9}$$

$S_h(f)$ represents the one-sided noise power spectral density (PSD), defined as:

$$E[\tilde{n}(f)\tilde{n}^*(f)] = \frac{1}{2}\delta(f - f')S_h(f) \tag{1.10}$$

The adjacent figure [17] shows the aLIGO noise PSD along with the principle sources of noise.



Noise in the detector stream is assumed to be stationary gaussian noise which randomly fluctuates about a mean value of zero. In this case, the noise component of the matched filter, $\langle n|h\rangle$ would be negligible and the signal component, $\langle s|h\rangle$ would dominate IF the template closely matches the signal.

The Signal to Noise Ratio (SNR) of the matched filtering calculation between the data and a template is defined as:[21]

$$\rho \equiv \frac{\langle s|h\rangle}{\sqrt{E[\langle n|h\rangle^2]}} \tag{1.11}$$

Assuming we have *normalised* templates, the noise output of the filter turns out to be a gaussian random variable with zero mean and unit variance. This makes the denominator in Eqn.(1.11) unity, and hence

$$\rho = \langle s|h\rangle = 4\mathrm{Re}\int_0^\infty \frac{\tilde{s}^*(f)\tilde{h}(f)}{S_h(f)} \tag{1.12}$$

Time Domain Filtering

The matched filtering operation can be carried out in the time domain. Here, we depict the discretely sampled data as $x[t]$ and the template we match it with, also sampled at the same rate, as $h[t]$. We weigh these time series with the inverse of the noise PSD, a process known as *whitening*. This suppresses those frequencies at which the interferometer sensitivity is low. Matched filtering in the time domain involves convolving a whitened data stream with a whitened template:

$$\rho[\tau] = \sum_{t=0}^{N-1} x\left[\tau - t\right] h\left[t\right] \tag{1.13}$$

This gives us a time series of SNR, $\rho[\tau]$ which peaks at the time corresponding to the maximum overlap between the signal and the template. This operation can also be thought of as a series of time-shifted *inner products* of the whitened data and the whitened template.

1.2 EM Follow-up and Low-latency Searches

One of the next big achievements in this field will be the first successful Electromagnetic follow-up observation of a Gravitational Wave source, especially a compact binary containing at least one Neutron Star. Such mergers have been modelled [23] to emit afterglow emissions coming from various astrophysical processes. Binary Neutron Star (BNS) mergers are believed to be the progenitors of Short Duration Gamma Ray Bursts (SGRBs). SGRB as well as X-ray emission from a BNS system are modelled to take place a mere seconds after the merger. Transient signals in the optical spectrum (kilonovae) coming from the tidal tail of the surrounding gas are expected to start coming a few hours post merger and last for an order of a few days. Radio afterglows from the ejecta are longer lasting transients which may last for weeks post merger. Observing an event in both the EM and GW spectra would have many invaluable advantages. It is thus imperative to detect Gravitational Wave signals as soon as possible, so as to provide timely alerts to observatories for follow-up. This has led to the development of *low-latency* online search pipelines, like GSTLAL[22]. Figure 1.4 summarises the current time delays involved in various stages of the GW search pipeline.

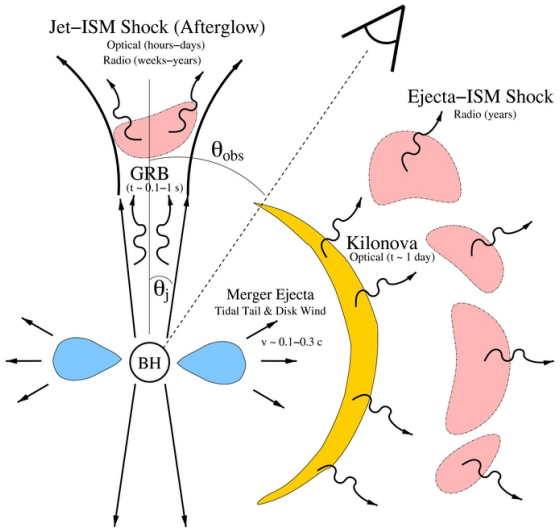


Figure 1.3: Electromagnetic emission from a Binary Neutron Star merger

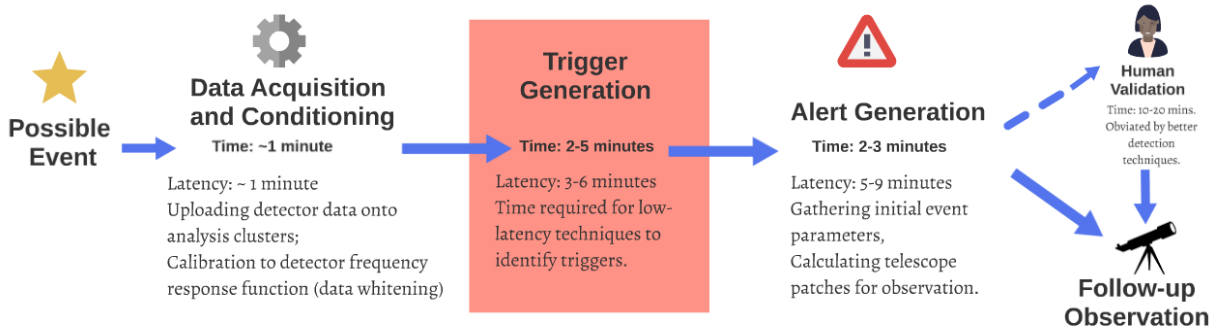


Figure 1.4: The buildup of latency: From signal to sky-search

1.2.1 Fast Time Domain searches: A problem in data reduction

Let us look at the numbers involved in carrying out Time-Domain (TD) matched filtering searches using Gravitational Wave templates. The templates are generated using various combinations of intrinsic binary parameters like masses and spins. To ensure that we conver the parameter space efficiently, the number of templates we have to produce ranges from tens of thousands to a few million. In addition, the template lengths vary from a few seconds for binary black holes to over 20 minutes for binary neutron stars. If sampled at a rate of 4096 Hz., the templates have a number of time points of the order of a few million. The TD convolutions involved in the matched filter calculations now become computationally very expensive. Scanning a stream of data with a huge template bank, with all templates within it sampled at millions of time points makes this a high-dimensional problem. We imagine each sampling point in the template time series to be a dimension of a vector space. Each template thus represents a *data vector* in this space, and we have many such templates. This is where *data reduction* techniques, which we shall read about in the next chapter, are useful. They can either help us reduce the number of templates we need to compare the data with, or the effective number of time points used in the computations.

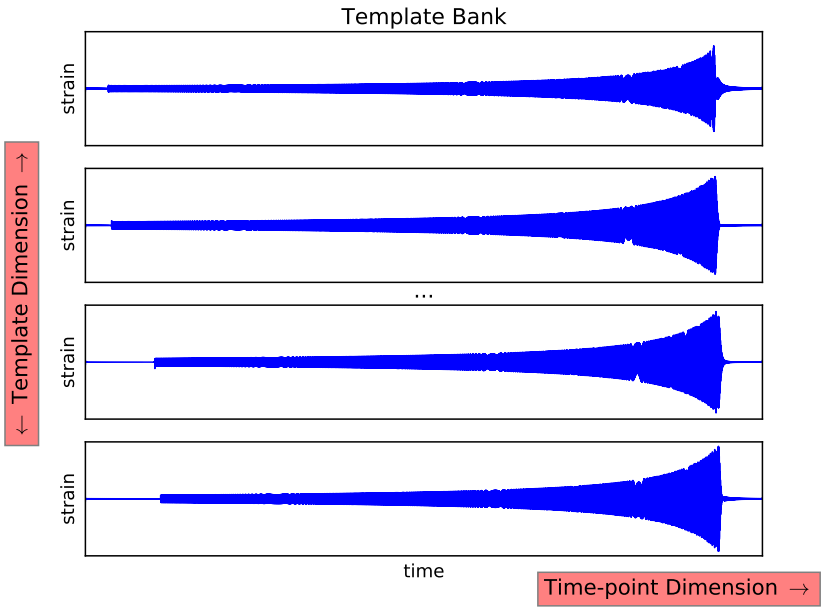


Figure 1.5: A problem in dimensionality reduction...

Chapter 2

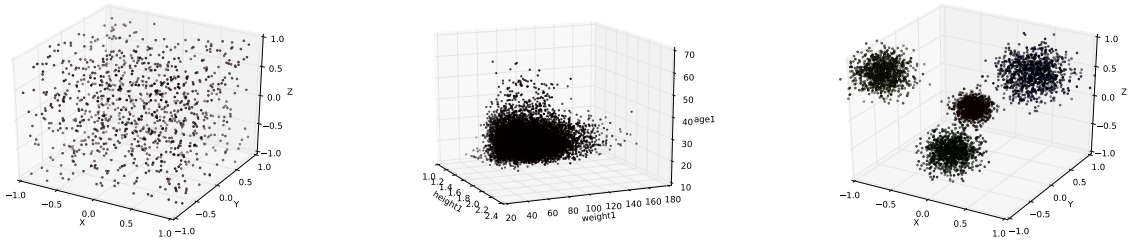
Data Reduction and Fast Gravitational Wave Searches

Big Data has revolutionised the information age we live in. Every passing day, we leave a digital footprint in terms of the web pages we visit, the things we buy online, the songs we listen to, and so on. Beyond personal information, the immense digitisation of the material world, like e-print newspapers, books and journals, photographs and forensic data have led to a surge in the development of data analytic techniques for efficient ways of storing, handling and querying large amounts of data.

Different kinds of digital data structures are visualised as vector spaces in high dimensions, populated with data-vectors representing real-world objects. A popular example is the 'bag of words' representation of textual data in the form of books or documents. Here, we imagine every word in the English language as being one dimension along a vector space of rank equal to the size of the entire English vocabulary. We can populate this vector space with any form of textual documents as data vectors. Each vector (document) has an integral magnitude of components along a particular word-axis equivalent to the number of times that word is used inside it. For example, if a document contained just the first paragraph of this chapter, it would have a magnitude of: 7 along the 'the' axis, 4 along the 'and' axis, 4 along the 'data' axis and 1 along the 'forensic' axis. At the same time, you will appreciate that a huge number of vector components corresponding to words which do not belong to the text, like 'panda', 'iridescent' and 'relativity' are zeros. Regardless of how verbose a

book or a document is, the proportion of words it contains would be negligible to the total size of the vocabulary. This is an example of a *sparse* data structure.

If we populate the bag-of-words vector space with books from a library, we would find that data vectors corresponding to books that are of a particular subject, say Physics, tend to cluster together. This is because they would have stronger components along scientific word-coordinates like 'magnetism' and 'momentum'. One would thus expect to find clusters of books corresponding to different subjects distributed along the vector space. Figure 2.1 summarises the different kinds of data structures based on their distributions in a given high dimensional space.



(a) Random, Uniform data **(b)** Structured data **(c)** Clustered data

Figure 2.1: Types of Data distributions in 3 Dimensions

As we increase the number of dimensions in our representative data space, we run into several problems. Richard E. Bellman coined the term '*The Curse of Dimensionality*' to describe the various counter-intuitive phenomena that occur as dimensionality increases far and beyond the 3 dimensions of the physical world we live in. Examples include the exponential sparsification of data due to the n-fold increase in volume corresponding to a linear increase in the dimensionality, n. This leads to an overwhelming percentage of relatively orthogonal vectors in high dimensional spaces (more on this in chapter 3). High-dimensional data is very hard to store, handle and make computations with. Even polynomial time algorithms are largely inefficient when the dimensionality exceeds a few orders of magnitude.

Data Reduction involves converting or projecting high dimensional data into lower dimensions, without affecting the inherent structure or relationships between the data vectors. This carries an assumption that a given high dimensional data distribution actually has a lower intrinsic dimensionality. It is an effort to transform the data into another space of a lower rank, consisting of a fewer number of basis vectors. There are a number of ways to do

this, the most common being Singular Value Decomposition.

2.1 Singular Value Decomposition

The Singular Value Decomposition (SVD)[27] theorem states that any rectangular matrix can be decomposed in the following way:

$$A_{mn} = U_{mm}\Sigma_{mn}V_{nn}^{T} \tag{2.1}$$

Such that:

- The columns of U are the orthonormal eigenvectors of AA^T,
- The rows of V^T are the orthonormal eigenvectors of A^TA,
- Σ is a diagonal matrix containing the square roots of the eigenvalues of U, which are the same as the eigenvalues of V, arranged in descending order:
$$\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \sigma_4 \geq ... \geq \sigma_n \geq 0$$

Singular Value Decomposition **picks out** the directions, or eigenvectors along which the data distribution (or variability) is maximum. The singular values (σ_s) quantise the magnitudes of this variability. They are the eigenvalues corresponding to each basis vector pointing in a certain eigendirection.

Truncated SVD

The data reduction part comes in the Truncated SVD representation, which is identical to eqn.(2.1), except that it includes only the first r out of n **significant** singular values and their corresponding basis vectors. In other words, it is a reconstruction of the data using a basis of vectors which captures the inherent data distribution. We assume that the $(n - r)$ basis eigenvectors which are left out do not contain significant information about the data.

We shall look at how SVD works with an example of a dataset in 3 dimensions, which helps us visualise the procedure.

SVD in 3 Dimensions

Figure 2.1b in the preceding section is actually the 3-dimensional representation of individual athletes who took part in the Rio 2016 Olympic games. The athletes are scattered along 3 coordinates corresponding to their height, weight and age. This dataset comprising data of 11,538 athletes from 205 countries was taken from the Rio 2016 Official Website.[1]

The dataset has the following inherent structure: we expect taller athletes to weigh more, and hence their height vs weight scatter should follow a somewhat linear trend (Figure 2.2). Though I did not qualify for Rio, the red dot representing my own components along the height and weight axes places me as a perfectly average olympic athlete. I would be cast as an outlier from this distribution only if a third dimension signifying the number of push-ups a person can do were added. The third axis which we do add, thankfully, is that of age. We expect the linear Height-Weight trend to be followed for athletes across all ages, since age does not introduce characteristic biases in either height or weight. Hence, in the 3 dimensions of height, weight and age, we expect this data set to lie along a plane: the linear height-weight trendline extended along the age axis. A truncated SVD does precisely this: it picks out this plane and projects the data along its principal axes.

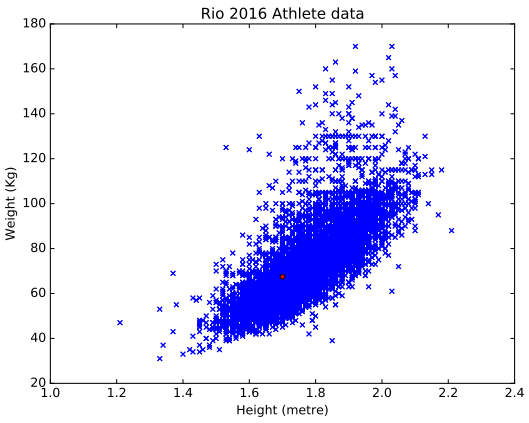
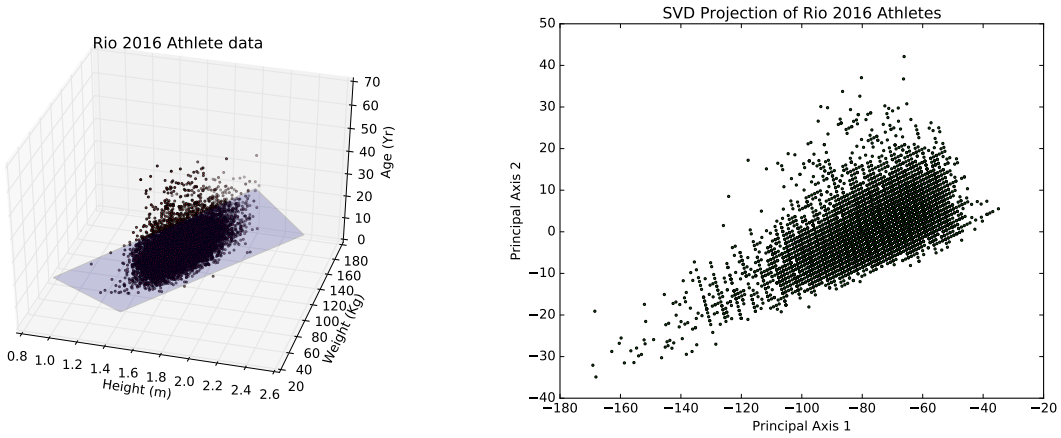


Figure 2.2: Rio 2016 Athletes: Weight vs Height

Figure 2.3a helps visualise the SVD plane for the Rio 2016 data scatter. The diagonal matrix of eigenvalues for this dataset is:



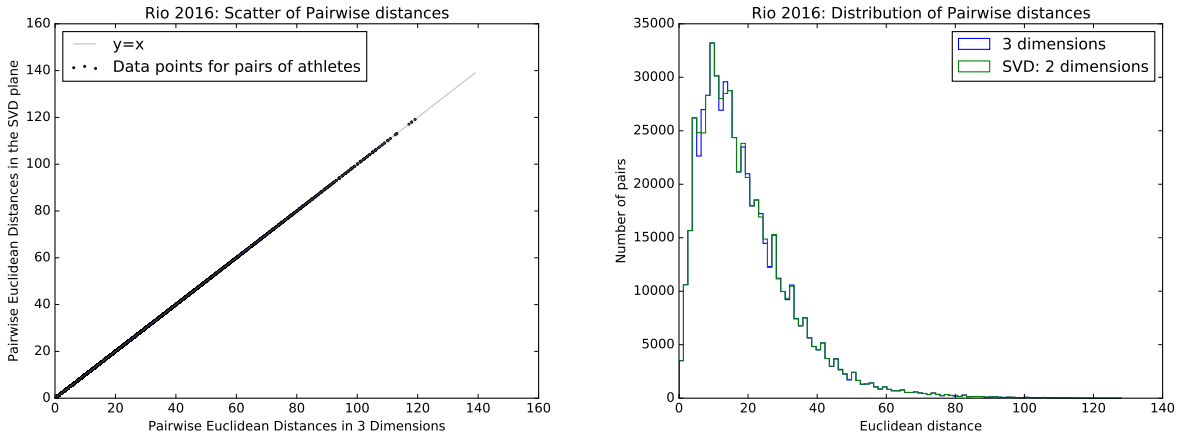
(a) Actual Data **(b)** Projected Data

Figure 2.3: Visualising Rio 2016 Athletes in different bases

$$\Sigma = \begin{bmatrix} 8175.32 & 0 & 0 \\ 0 & 739.14 & 0 \\ 0 & 0 & 23.85 \end{bmatrix}$$

The third principal component, (23.86) is over two and one orders of magnitude lower as compared to the first two components respectively. Hence, we can use the truncated SVD including only the first two basis vectors to reconstruct the entire dataset in 2 dimensions. The 2-D scatter of the athletes along this SVD plane is depicted in Figure 2.3b We observe a larger distribution, capturing greater variation in the data scatter along this plane. Contrast this with Figure 2.2, which is essentially a direct projection of the dataset onto the Height-Weight plane by suppressing the Age axis.

The truncated SVD thus helps us perform computations on this dataset in only 2 Dimensions, which is faster. How to characterise the conservation of the properties of our dataset? One such metric is the Euclidean Distance. It tells us how close two data points (athletes) are to each other. Athletes of a similar age, height and weight are separated by a smaller distance. A weightlifter and a gymnast will have a greater distance separating their corresponding data points. Figure 2.4a shows the scatter of pairwise distances in 3 Dimensions compared with the same in the 2D SVD plane. It shows a perfectly linear trend. What this means is that two athletes placed a certain distance apart in the 3D scatter have the same separation in the Principal plane as well. The distribution plots of pairwise distances in 3D



(a) Comparison of Pairwise Distances

(b) Conservation of Pairwise distances

Figure 2.4: Conservation of data structure under SVD

and reduced 2D also match perfectly as seen in Figure 2.4b.

Summarising, we can compare relationships between a dataset faster using reconstructed vectors in a lower dimension. While this was just a toy example to help us visualise the technique, one can really speed up calculations in applications involving larger dimensions, just like the problem posed in section 1.2.1.

2.2 Dimensionality reduction of Gravitational Wave Templates

The problem of a Time-Domain (TD) matched filtering based Compact Binary search involves convolving a large number of Gravitational Wave templates $h_\alpha[t]$ with the incoming signal $s[t]$. The number of templates taken from the parameter space is of the order of a few tens of thousand to a million $(10^5 - 10^6)$, while the number of time samples is of the order of a few million (10^6). GSTLAL, a TD search pipeline for aLIGO, uses the LLOID [10] algorithm to reduce the dimensionality in both the time-sample as well as template space. For reducing the number of templates, they apply Singular Value Decomposition. For reducing the time-point dimension, they use frequency based subsampling.

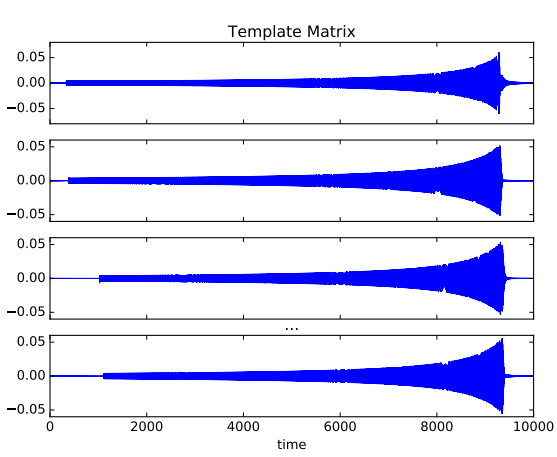
2.2.1 Reduction of Template space using SVD

In order to lower the rate of false positives, a high SNR threshold is used for characterising a detection. As a consequence, the template bank constructed is densely populated such that adjacent templates have a *minimal match* of 0.97. This ensures a small maximum loss of SNR with respect to a given template. This arrangement also has a flip side: the more we populate our parameter space with templates, the greater number of similar templates we generate. This introduces degeneracies within a set of the templatebank. These are exploited using SVD, which eliminates the degeneracies by producing a set of orthogonal basis vectors which are used to reconstruct the templates.

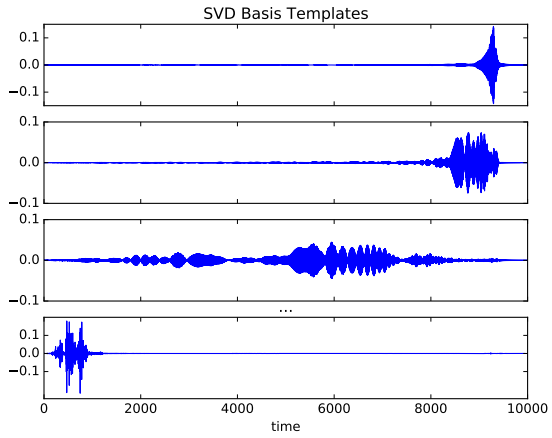
Consider a template bank containing N templates sampled at M time points. The $N \times M$ template data matrix is represented as H_{NM}. The α^{th} template can be reconstructed using Singular Value Decomposition of the template data matrix as[11]:

$$h_\alpha[t] = \sum_{\nu=1}^{N} v_{\alpha\nu}\sigma_\nu u_\nu[t] \tag{2.2}$$

Where V is the matrix of N row vectors forming an orthogonal basis along the Principal Components of H, and σ is the diagonal matrix of eigenvalues in decreasing order.



(a) Example Templates



(b) Examples of the Principal basis vectors: the most significant at the top, the least at the bottom

Figure 2.5: SVD of a Gravitational Wave templatebank

The time domain expression of the SNR (1.13) can be thus written as:

$$\rho_\alpha[\tau] = \sum_{t=0}^{j-1} x[\tau - t] \sum_{\nu=1}^{N} v_{\mu\nu}\sigma_\nu u_\nu[t] \tag{2.3}$$

Owing to redundancies in the template waveforms, assume that we have $k < N$ significant components. The templates can now be reconstructed using only the basis vectors corresponding to the first k significant eigenvalues:

$$h'_\alpha[t] = \sum_{\nu=1}^{k} v_{\alpha\nu}\sigma_\nu u_\nu[t] \tag{2.4}$$

The SNR from the reconstructed templates is given by:

$$\rho'_\alpha[\tau] = \sum_{t=0}^{j-1} x[\tau - t] \sum_{\nu=1}^{k} v_{\mu\nu}\sigma_\nu u_\nu[t] \tag{2.5}$$

Since the truncated SVD reconstruction is not perfect, we get errors in the value of the reconstructed SNR. These are characterised by the Fractional SNR loss, given by:

$$\frac{\delta\rho_\alpha}{\rho_\alpha} = 1 - \frac{|\rho'_\alpha|}{|\rho_\alpha|} \tag{2.6}$$

The average Fractional SNR Loss turns out to be equal to the summation of all the discarded Singular Values.

$$\langle\frac{\delta\rho}{\rho}\rangle = \frac{1}{2N} \sum_{\alpha=N'+1}^{N} \sigma_\alpha^2 \tag{2.7}$$

This means that the accuracy of reconstruction can be tweaked by adjusting the number of SVD basis vectors used in reconstruction. Cannon et al. [11] found that we can use an order of magnitude lower number of basis vectors compared to the number of templates whilst keeping the average fractional loss in SNR of the order of 0.001.

2.2.2 Reduction of Time-sample space

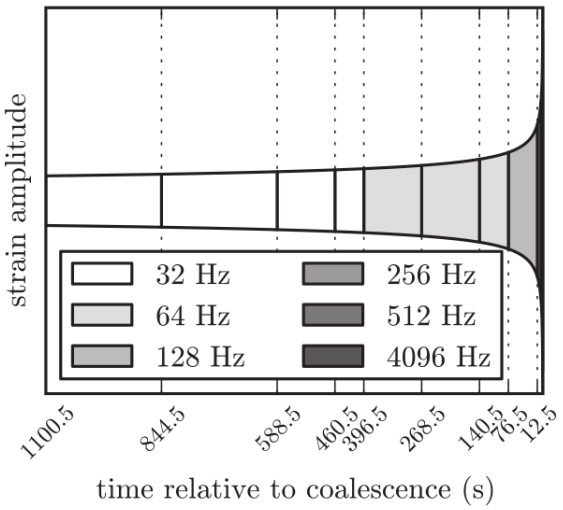


Figure 2.6: Frequency-based subsampling

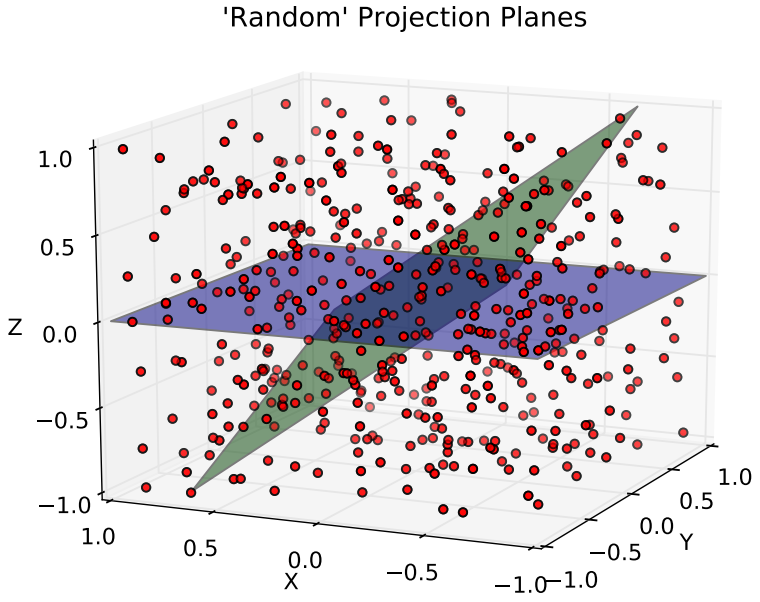
The LLOID algorithm uses **frequency-based subsampling** to reduce the number of time points used in the match calculation. As given in equation 1.6, The frequency of the Gravitational Wave chirps increases as the signal evolves with time. The aLIGO sensitivity ranges from 10 Hz. to about 10 KHz. Hence, an incoming inspiral signal has a frequency of 10 Hz., which increases by over two orders of magnitude by the time it reaches the merger phase.

According to the **Nyquist frequency criterion**, the minimum frequency at which a signal can be sampled without loss of information is twice the maximum frequency component present in the signal. This criterion is exploited by slicing a Gravitational Wave template into different parts based on frequency. The inspiral section, which evolves slowly with a lower frequency, can be sampled at a lower rate, reducing the number of time points. The subsequent slices are sampled at successively higher frequencies (Figure 2.6). This obviates the need of sampling the entire signal at the maximum frequency (4096 Hz.) that captures the merger part.

It is worthwile noting that if a typical binary Neutron Star-Black Hole signal, lasting for about 5 minutes is sampled at the second lowest frequency i.e. 64 Hz, the number of time points will be reduced from 1,228,800 to 19,200 time points, but with a significant loss of information.

An Alternative Dimensionality Reduction Technique

A disadvantage of the SVD approach is the heavy computation involved in finding the reduced set of basis vectors. In some scenarios, this may negate the computational efficiency gained in working with this lower dimensional basis. We pose the following question: Is there a faster way of projecting data into lower dimensions, while conserving its structure the way SVD does? SVD is indeed the optimal way of finding an appropriate basis to project data into. What if we do not take the effort to find such a basis altogether? What if we project our data onto random subspaces? Will we still preserve how it is distributed? Sounds like an idea too good to be true. But that is what they say about all good ideas...



Chapter 3

Random Projections

> In mathematics you don't understand things. You just get used to them.
>
> John von Neumann

Random Projection is a powerful mathematical technique for dimensionality reduction. It was conceived following the seminal result by William B. Johnson and Joram Lindenstrauss [19] in the early 1980s. The fundamental idea is that linearly projecting data in high dimensions to a random subspace preserves the inherent geometry of its distribution. In other words, the common metrics of closeness between pairs of data vectors, like the Euclidean distance and Inner Product are conserved under such a projection with a high probability, within a given error tolerance. Further, we can reduce the dimensionality of our system logarithmically with respect to the number of data vectors, irrespective of the original number of ambient dimensions. In this chapter, we will understand how this technique works, how it is applied and what advantages it offers over other data reduction techniques.

Random Projections have found applications in a variety of fields. Bingham et al. have used it to classify images and textual data, and compared its performance against DFT, DCT and SVD [7]. In a similar application, Papadimitriou et al. use Random Projection as a precursor to Latent Semantic Indexing [26]. It has been used for developing face recognition software by Goal et al[16]. S. Dasgupta has used this technique to classify and characterise Gaussian clusters [12]. The immense capability Random Projection offers in going down to low dimensions makes it valuable to design training sets of feature vectors for Machine Learning algorithms, as demonstrated by Fradkin et al [15]. Random Projection has also found scientific applications, having been used for MRI image analysis and compressive signal processing.[9]

3.1 The Technique

Consider a high-dimensional space with d-dimensions, populated with n vectors. A vector in this space is represented by v^d. Random Projection, as the name suggests, is a projection of this high d-dimensional vector space into a lower dimensional subspace involving $k << d$ coordinates, using a projection matrix having k *random vectors* in d dimensions. The projection of a single vector can be represented as :

$$\begin{bmatrix} \\ k \\ \\ \end{bmatrix} = \frac{1}{\sqrt{k}} \begin{bmatrix} \\ k \times d \\ \\ \end{bmatrix} \begin{bmatrix} \\ d \\ \\ \\ \end{bmatrix}$$

$$v^k = \frac{1}{\sqrt{k}} \mathbf{R}_{k \times d} u^d \tag{3.1}$$

The entire dataset can be represented by a data matrix of n column vectors in d-

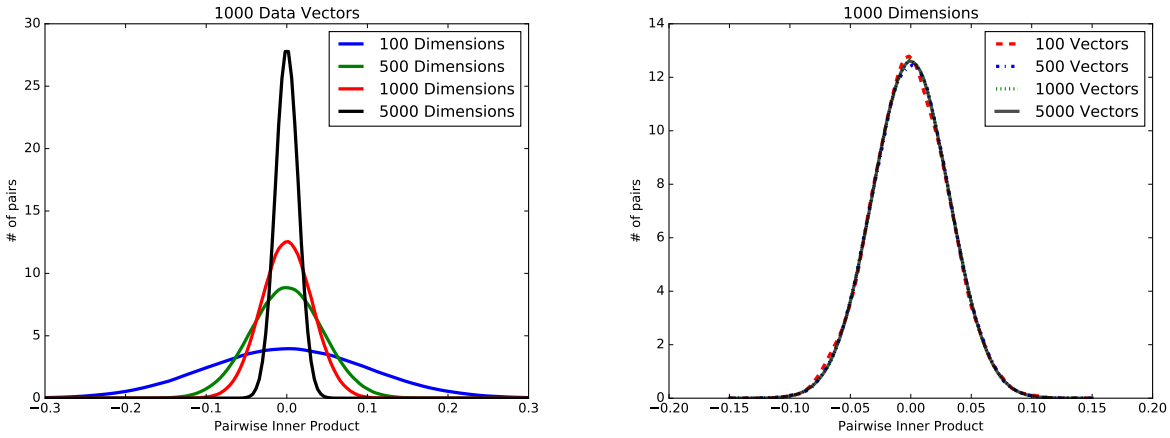
dimensions: V_{dn}. The Random Projection of all data vectors together to a subspace forms a data matrix of n projected column vectors in k-dimensions: V'_{kn}. This transformation can be represented by:

$$V'_{kn} = \frac{1}{\sqrt{k}} \mathcal{R}_{kd} V_{dn}$$

How does one determine the Random Projection matrix, $\mathcal{R}$? This matrix can have its entries picked randomly from different types of distributions. The selection of $\mathcal{R}$ has no bearings on the nature of our dataset. The simplest kind of distribution is a standard normal (Gaussian) distribution. It turns out that any distribution having the property of zero mean and unit variance can serve the purpose of Random Projection. This fact can be exploited to make computations involving these matrices faster and more efficient. We shall discuss Random Projection matrices generated from alternative distributions in section 3.2.2.

Take, for now, the Gaussian case, with $\mathcal{R}_{ij} \in \mathcal{N}(0, 1)$. We can imagine the $k \times d$ matrix as a set of k randomly generated basis (row) vectors in a d-dimensional space. Mathematicians would be quick to point out that if we want to call this process a projection onto a k-dimensional subspace, the basis of the projection matrix R needs to be orthogonal. It seems implausible that our randomly generated vectors can guarantee this, since orthonormalisation of a set of vectors involves an application of the elaborate Gram-Schmidt process.

This requirement is not perfectly satisfied, but adequately taken care of by exploiting an aspect of the curse of dimensionality. As pointed out in the previous chapter, the exponential sparsification of randomly distributed data in high dimensions leads to an overwhelming percentage of pairs of vectors being nearly orthogonal. The higher the dimension, the more orthogonal pairs of vectors become. This is illustrated in Figure 3.1a. Surprisingly, this degree of orthogonality is dictated by the number of dimensions alone, regardless of how many vectors populate the space (Figure 3.1b). As table 3.1 shows, it follows a neat $\frac{\sqrt{N}}{N}$ relation for an N-dimensional vector space. As we shall see later, Gravitatioinal Wave templates have an ambient dimension of N over a million dimensions, so we are safe on the orthogonality front.



(a) Constant Number of Vectors **(b)** Constant Number of Dimensions

Figure 3.1: Orthogonality of Vectors in Higher Dimensions: The percentage of orthogonal pairs depends only on the ambient dimension (a) and not on the population (b)

Table 3.1: Deviation from orthogonality for pairs of vectors in successively higher dimensions

N	Standard Deviation From Distribution	$\frac{\sqrt{N}}{N}$
100	0.098	0.100
500	0.044	0.044
1000	0.031	0.031
5000	0.014	0.014

Normalisation is trivial once we have a quasi-orthogonal basis. The factor $\frac{1}{\sqrt{k}}$ in equation 3.1 is not for normalisation, but for scaling the norm of vectors in the dataset, a property crucial to make Random Projections work. The row vectors of a Random Projection matrix are thus almost orthogonal to each other. As an example, we use a $[\mathcal{R} : 2000 \times 5000]$ projection matrix in section 5.3. The mean square deviation of the entries of $\mathcal{R}\mathcal{R}^T$ from the entries of the corresponding rank-2000 Identity matrix, ($\mathcal{I}_{2000}$) is 0.008, less than a percent.

Hence, we now have a fast and computationally cheap technique of projecting data onto a *nearly* orthogonal subspace. We now claim that such a projection guarantees a preservation of distance metrics within the dataset, even when the projection scales the dimensionality down to an order logarithmically lower in the number of data points we have. Let us investigate how this is achieved theoretically.

3.2 Why Random Projections work: A Theoretical Background

Consider a set of m vectors populating a high dimensional space having d components.

Lemma 3.2.1 (Johnson, Lindenstrauss[19]). *For any $0 < \epsilon < 1$ and any integer m, there exists a positive integer k such that*

$$k = \frac{24 \log n}{2\epsilon^2 + 3\epsilon^3} \tag{3.2}$$

Then for any set V of m points in R^d, there exists a mapping $f : R^d \to R^k$ such that $\forall x_i, x_j \epsilon$ V,

$$(1-\epsilon)\|x_i - x_j\|^2 \leq \|f(x_i) - f(x_j)\|^2 \leq (1+\epsilon)\|x_i - x_j\|^2 \tag{3.3}$$

With a high probability. Further, this mapping can be found out in linearised polynomial time.

For its application to Random Projection, this lemma is often restated in its distributional form:

Lemma 3.2.2 (Distributional Johnson-Lindenstrauss).

$$Pr[(1-\epsilon)\|x_i - x_j\|^2 \leq \|f(x_i) - f(x_j)\|^2 \leq (1+\epsilon)\|x_i - x_j\|^2] \geq 1 - \frac{1}{n^2} \tag{3.4}$$

Proof. [13]

Let $x \in R^d$ be a vector in the original d-dimensional space.

$$\mathbf{x} = [x_1, x_2, x_3, ..., x_d]$$

Take the mapping $f : R^d \to R^k$ to be the linear transformation using a $k \times d$ matrix R, having its entries taken from a Gaussian distribution with mean 0 and a standard deviation of 1: $R_{ij} \in \mathcal{N}(0, 1)$

Under the projection

$$(\mathbf{R}x)_i = \frac{1}{\sqrt{k}} \sum_{j=1}^{d} R_{ij} x_j$$

x will be transformed to the k-dimensional vector $\mathcal{R}x$

$$\mathbf{Rx} = [(\mathbf{R}x)_1, (\mathbf{R}x)_2, (\mathbf{R}x)_3, ..., (\mathbf{R}x)_k]$$

Let $\|x\|^2$ represent the Euclidean Norm i.e. the squared Euclidean distance from the Origin.

We begin by proving **norm conservation**.

Proposition 3.2.1. *The linear mapping $R : x^d \rightarrow (\mathbf{R}x)^k$ preserves the norm of all data vectors upto a multiplicative factor of $(1 \pm \epsilon)$ with a high probability.*

$$Pr\{(1-\epsilon)\|x\|^2 \leq \|\mathbf{R}x\|^2 \leq (1+\epsilon)\|x\|^2\} \geq 1 - \frac{1}{n^2} \tag{3.5}$$

The squared norm of the projected vector is:

$$\|\mathbf{R}x\|^2 = \sum_{i=1}^{k} (\mathbf{R}x)_i^2 \tag{3.6}$$

We start out by finding the expected value of this projected norm:

$$\begin{aligned}
E\left(\|\mathbf{R}x\|^2\right) &= E\left(\sum_{i=1}^{k}(\mathbf{R}x)_i\right)^2 \\
&= \sum_{i=1}^{k}\left(E(\mathbf{R}x)_i^2\right) \\
&= \sum_{i=1}^{k}\frac{1}{k}E\left(\sum_{j=1}^{d}(R_{ij}x_j)\right)^2 \\
&= \sum_{i=1}^{k}\frac{1}{k}\sum_{1\le j,k\le d}x_jx_kE(R_{ij}R_{ik}) \\
&= \sum_{i=1}^{k}\frac{1}{k}\sum_{1\le j,k\le d}x_jx_k\delta_{jk} \\
&= \sum_{i=1}^{k}\frac{1}{k}\sum_{j=1}^{d}x_i^2 \\
&= \sum_{i=1}^{k}\frac{1}{k}\|x\|^2
\end{aligned}$$

Hence,

$$E\left(\|\mathbf{R}x\|^2\right) = \|x\|^2 \tag{3.7}$$

The expected value of the norm of the transformed vector in the k-dim space is equal to its original norm in the d-dim space.

This does not prove norm preservation, yet. We need to prove that the distribution of the norm value under projection is tightly concentrated around its expected value. For this, we propose the following non-constructive proof using the probabilistic method:

We aim to find $P\left[\|\mathbf{R}x\|^2 \ge (1+\epsilon)\|x\|^2\right]$, $P\left[\|\mathbf{R}x\|^2 \le (1-\epsilon)\|x\|^2\right]$ and show that these probabilities are tiny.

Define a new vector $\mathbf{Z}$:

$$\mathbf{Z} = \frac{\sqrt{k}}{\|x\|}(\mathbf{R}x) \tag{3.8}$$

The components of which are

$$z_i = \frac{1}{\|x\|}R^T x \tag{3.9}$$

Therefore,

$$\|\mathbf{Z}\|^2 = \sum_{i=1}^{k} \frac{1}{\|x\|^2}(R^T x)^2 = k\frac{\|\mathbf{R}x\|^2}{\|x\|^2} \tag{3.10}$$

Let $y = \|\mathbf{Z}\|^2$

The Probability we are looking for is:

$$\begin{aligned} P\left[\|\mathbf{R}x\|^2 \geq (1+\epsilon)\|x\|^2\right] &= P\left[y \geq (1+\epsilon)k\right] \\ &= P\left[e^{\lambda y} \geq e^{(1+\epsilon)k}\right] .....(\forall \lambda \geq 0) \end{aligned}$$

By Markov's Inequality [24]:

$$P\left[y \geq a\right] \leq E\frac{[y]}{a}$$

Therefore,

$$\begin{aligned} P\left[e^{\lambda y} \geq e^{(1+\epsilon)k}\right] &\leq \frac{E[e^{\lambda y}]}{e^{(1+\epsilon)k}} \\ &\leq \prod_{i=1}^{k} \frac{E[e^{\lambda y_i^2}]}{e^{(1+\epsilon)k}} \\ &\leq \left(\frac{E[e^{\lambda y_i^2}]}{e^{(1+\epsilon)k}}\right)^k \\ &\leq \left(\frac{1}{\sqrt{1-2\lambda}\cdot e^{(1+\epsilon)}}\right)^k \end{aligned}$$

Setting $\lambda = \frac{\epsilon}{2(1+\epsilon)}$:

$$\begin{aligned} P\left[e^{\lambda y} \geq e^{(1+\epsilon)k}\right] &\leq \left[(1+\epsilon)e^{-\epsilon}\right]^{\frac{k}{2}} \\ &\leq e^{(-\epsilon^2/2-\epsilon^3/3)k/2} \ldots \left(\text{using } \log(1+x) \leq x - \frac{x^2}{2} - \frac{x^3}{3}\right) \end{aligned}$$

The corresponding mirrored probability, $P\left[\|\mathbf{R}x\|^2 \leq (1-\epsilon)\|x\|^2\right] \leq e^{(-\epsilon^2/2-\epsilon^3/3)k/2}$ can be proved in a similar manner.

How to ensure these tail bounds fall steeply? Equation 3.2 suggests taking the following value of k:

$$k = \frac{4\log n}{\frac{\epsilon^2}{2} + \frac{\epsilon^3}{3}}$$

This implies,

$$\begin{aligned} P\left[\|\mathbf{R}x\|^2 \geq (1+\epsilon)\|x\|^2\right] &= P\left[\|\mathbf{R}x\|^2 \leq (1-\epsilon)\|x\|^2\right] \\ &\leq \exp\left(\frac{-(\epsilon^2/2+\epsilon^3/3)}{2}k\right) \\ &\leq \exp\left(\frac{-(\epsilon^2/2+\epsilon^3/3)}{2}\frac{4\log n}{(\epsilon^2/2+\epsilon^3/3)}\right) \\ &\leq e^{-2\log n} \end{aligned}$$

Hence,

$$Pr\{\|\mathbf{R}x\|^2 \notin (1 \pm \epsilon)\|x\|^2\} \leq \frac{2}{n^2} \tag{3.11}$$

The norm of a single projected vector is tightly concentrated around that of the original vector in a high dimensional space.

Since x is an arbitrary vector in the d-dimensional space, this property will hold true for any *distance* vector dx in the space, where $dx = x_i - x_j$ and x_i, x_j are data vectors.

Therefore, we can extend norm preservation to the conservation of all pairwise distances, by union bonding over all such distance vectors. If the total number of data vectors is n, we have a total of $\binom{n}{2} = \frac{n(n-1)}{2}$ pairs.

This gives the following probability for all pairwise distances to lie outside the $\epsilon-$embedding:

$$\begin{aligned} P\left[\cup x_i\right] &\leq \sum_{i=1}^{n} P\left[x_i\right] \\ &\leq \frac{n(n-1)}{2}\frac{2}{n^2} \\ Pr\{\|\mathbf{R}x_i - \mathbf{R}x_j\|^2 \notin (1 \pm \epsilon)\|x_i - x_j\|^2\} &\leq 1 - \frac{1}{n} \end{aligned}$$

Therefore,

$$Pr\{(1-\epsilon)\|x_i - x_j\|^2 \leq \|\mathbf{R}x_i - \mathbf{R}x_j\|^2 \leq (1+\epsilon)\|x_i - x_j\|^2\} \geq 1 - \frac{1}{n} \tag{3.12}$$

which is very high for large n, thereby completing our proof.

3.2.1 Conservation of Inner Products

Here we prove the above JL bounds for another metric for closeness between data points, namely, the Inner Product. The inner product is a far more important metric for our bigger picture, since the matched filtering output of a Gravitational Wave search is the Inner Product between the data and a template at the point of their maximum overlap. 1.1

The inner product between to vectors x and y can be written as $\langle x \cdot y \rangle = x^T y$. We can re-write this in terms of the Euclidean distance between x and y as:

$$x^T y = \frac{1}{4}\left(\|x+y\|^2 - \|x-y\|^2\right) \tag{3.13}$$

Now, if we project the two vectors under Random Projections to k-dimensions:

$$x \to \mathbf{R}x$$

$$y \to \mathbf{R}y$$

The Inner Product between the projected vectors can be written as:

$$(\mathbf{R}x)^T\mathbf{R}y = \frac{1}{4}\left(\|\mathbf{R}x + \mathbf{R}y\|^2 - \|\mathbf{R}x - \mathbf{R}y\|^2\right) \tag{3.14}$$

Applying the distributional JL lemma on both the distance terms separately followed by a union bound achieves the following bound on the conservation of Inner Products:

$$Pr[(1-\epsilon)x^Ty \leq (\mathbf{R}x)^T\mathbf{R}y \leq (1+\epsilon)x^Ty] \leq e^{-\left(\frac{\epsilon^2}{8}\right)} \tag{3.15}$$

The JL Lemma thus guarantees the preservation of the Inner Product metric as well, in a Randomly Projected subspace. This preservation however strongly depends on the value of the original inner product, as we shall see in the following corollary[20]:

Corollary 3.2.2. *(Relative distortion bounds) If θ is the angle between two d-dimensional vectors x and y, We obtain the following relative distortion bound for the conservation of their inner product:*

$$Pr[(1-\epsilon)x^Ty \leq (\mathbf{R}x)^T\mathbf{R}y \leq (1+\epsilon)x^Ty] \leq \exp\left(-\frac{\epsilon^2}{8}\cos^2\theta\right) \tag{3.16}$$

Proof. The angle θ is related to the inner product by $\cos\theta = \frac{\|x\|\cdot\|y\|}{x^Ty}$

If x and y are both normalised,

$$\cos\theta = \frac{1}{x^Ty}$$

Dividing both sides of 3.16 by x^Ty, we get:

$$Pr\left[1 - \frac{\epsilon}{\cos\theta} \leq \frac{(\mathbf{R}x)^T\mathbf{R}y}{x^Ty} \leq 1 + \frac{\epsilon}{\cos\theta}\right] \leq e^{\left(-\frac{\epsilon^2}{8}\right)} \tag{3.17}$$

Setting $\eta = \frac{\epsilon}{\cos\theta}$ and solving the JL bound for η recovers eqn. (3.14) to prove the corollary.

3.2.2 Alternate Distributions

Looking back at our proof for the Johnson Lindenstrauss lemma, we notice a surprising fact. Though we have used a projection matrix having i.i.d. standard gaussian entries, the only property of the Normal distribution we have utilised is the fact that its first two moments are zero and one respectively. What this means, is that *any* distribution having 0 mean and unit variance can be used to construct a Random Projection matrix! This fact can be exploited to construct matrices which are easy to construct and fast to project with. Achlioptas [4] gives 2 such distributions:

1. **Sparse Distribution:**
   This Random Projection matrix is *sparse*, with the fact that $2/3^{rd}$ of its entries are zeros:

$$r_{ij} = \begin{cases} +\sqrt{3}, & \text{with probability} \frac{1}{6} \\ 0, & \text{with probability} \frac{2}{3} \\ -\sqrt{3} & \text{with probability} \frac{1}{6} \end{cases} \tag{3.18}$$

2. **Binary Distribution:**
   This Random Projection matrix has its entries taken from a binary distribution:

$$r_{ij} = \begin{cases} +1, & \text{with probability} \frac{1}{2} \\ -1, & \text{with probability} \frac{1}{2} \end{cases} \tag{3.19}$$

3.3 Experiments with Random Projection

Having seen the theoretical guarantees Random Projections offer, let us put them into practice. We start off by analysing a set of *normalised* d-dimensional vectors randomly distributed in the $[-1, 1]^d$ hypercube. Following is the list of parameters of our dataset:

- m : Number of data vectors = 1000
- d : Original dimension = 20,000

- k_i : Values of projected dimension, $\in [15000, 10000, 5000, 2000]$

The size of the Random Projection matrix projecting from $d = 20000$ to a lower dimension of k is $20000 \times k$. I have projected the entire dataset to 4 different values of reduced dimensionality. Let us check out various properties of this dataset under Random Projection.

Norm Preservation

Norm preservation was the premise in proving the Johnson Lindenstrauss lemma. Since we started with a set of normalised vectors, it would be interesting to see how the norms of the projected vectors are distributed around 1.0

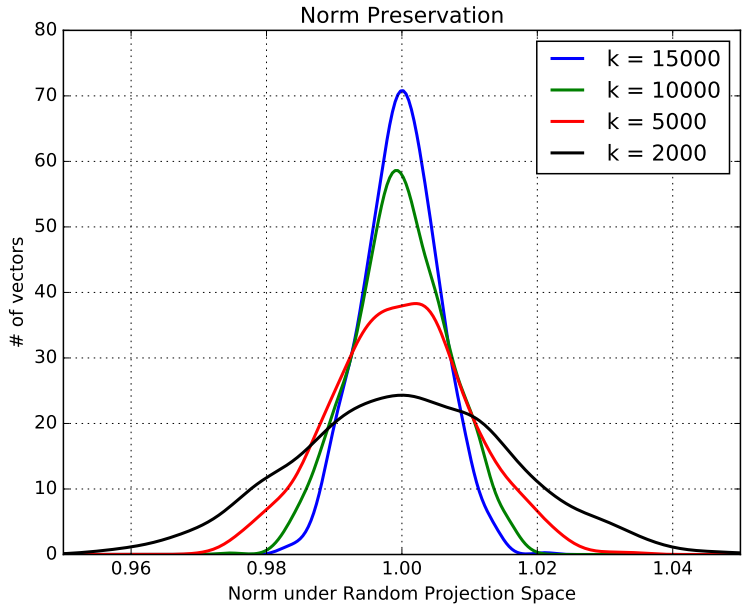
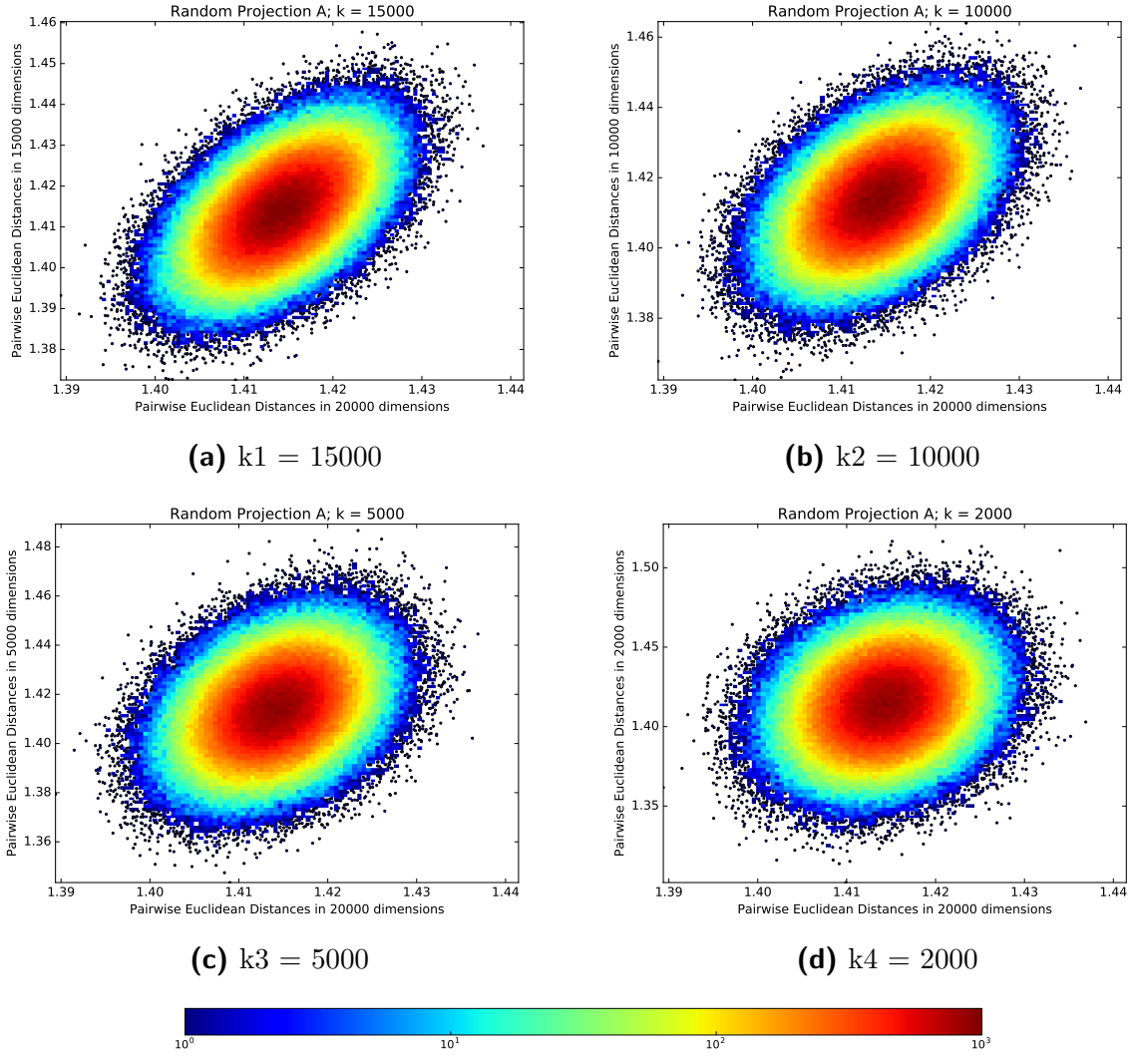


Figure 3.2: Conservation of Norm

The projected norm distribution for all reduced dimensions is peaked around 1.0, with a standard deviation given in table 3.2. The skewed trend for the first three projections towards mean values less than 1.0 is an artifact of the randomness; it changes if we use a different projection matrix.

Distance Preservation

The Union bound guarantees a better conservation bound for preservation of pairwise distances. Figure 3.3 displays the scatter of the projected pairwise distances to the dimensions (a) : $k1 = 15,000$, (a) : $k2 = 10,000$, (a) : $k3 = 5000$ and (a) : $k4 = 2000$ resepectively. We observe scatter trends that look like ellipses with their major axes oriented aling the 45° straight line. The ellipses get fatter, that is their scatter increases in variance as we project the data onto lower dimensions.



(a) k1 = 15000

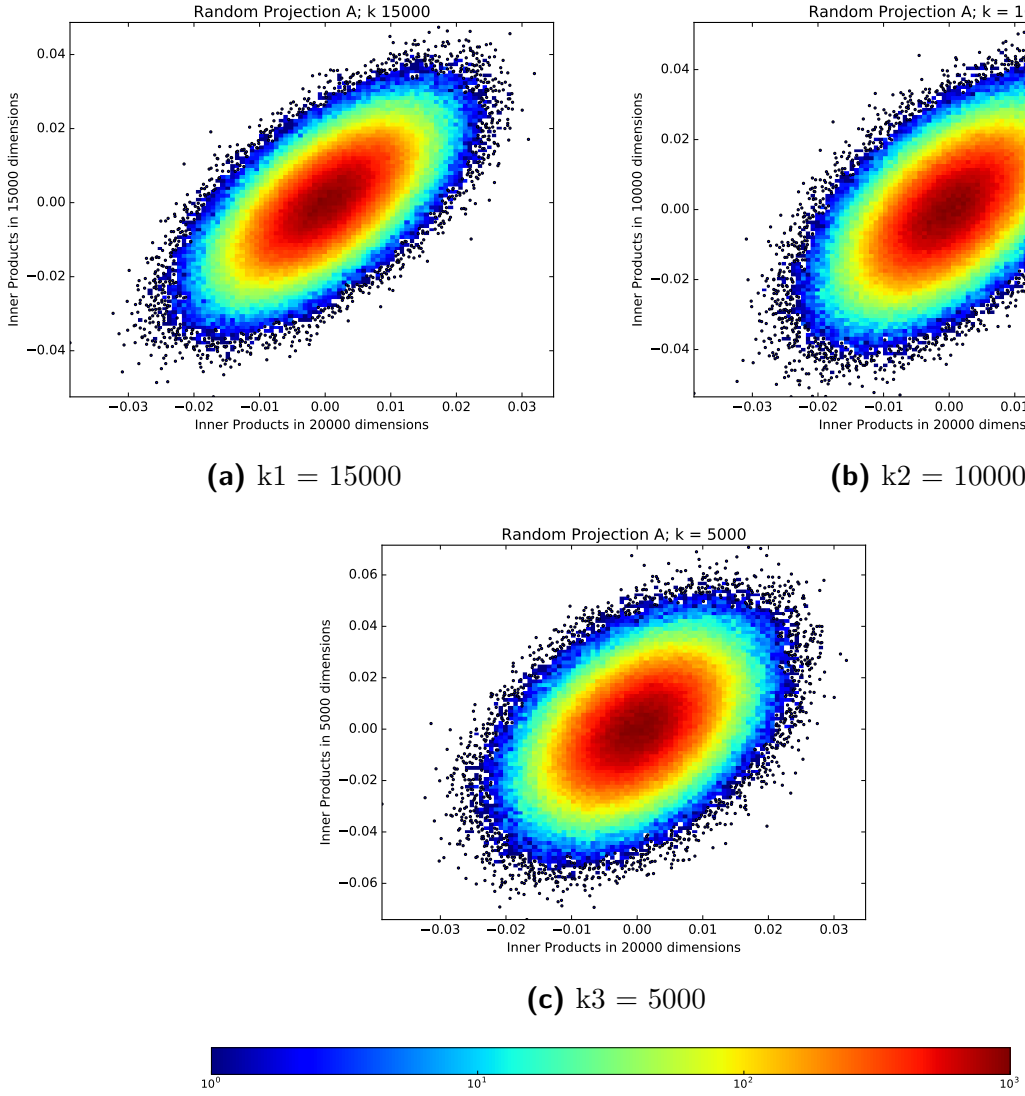
(b) k2 = 10000

(c) k3 = 5000

(d) k4 = 2000

Figure 3.3: Distance Preservation under Random Projection

Inner Product Preservation



(a) k1 = 15000

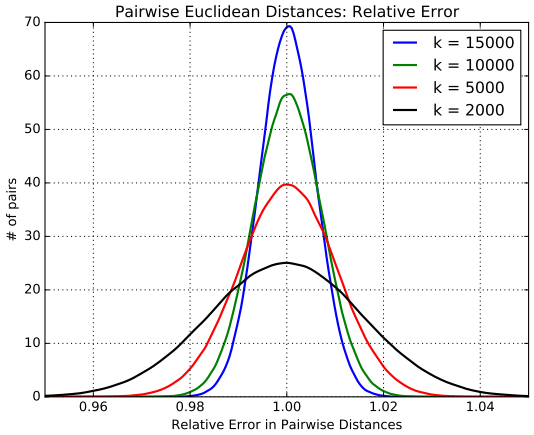
(b) k2 = 10000

(c) k3 = 5000

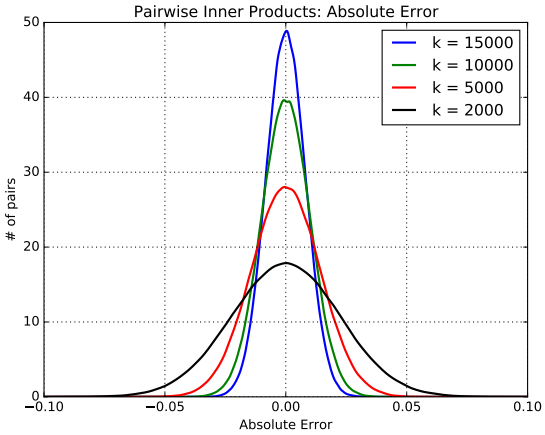
Figure 3.4: Inner Product Preservation under Random Projection

The distortion of Inner Products closely matches that of the Euclidean distances for the dataset above. It should be noted though that this dataset is uniformly distributed in the d-hypercube and normalised, and hence the distribution of Inner Products in the original space is strongly centred around 0.0. This means most pairs of vectors are nearly orthogonal. As we shall see later, the scatter plots look different if the vectors lie along the same or opposite directions.

Relative Errors in conservation



(a) Distance conservation: Relative Error



(b) Inner Product conservation: Error

Figure 3.5a shows the following relative error in the distribution of pairwise Euclidean distances:

$$\Delta = \frac{\|x_i - x_j\|^2 - \|\mathbf{R}x_i - \mathbf{R}x_j\|^2}{\|x_i - x_j\|^2}$$

Not surprisingly, the variances of the Errors vary inversely with the degree of dimensionality reduction. The same trend holds for Errors in the conservation of Inner Products Figure 3.5b. Here, I have plotted the absolute differences between pairwise inner products in the full space and under projection.

In Appendix A, we have repeated the experiments with Random Projection from this section using Achlioptas' **Sparse RP matrix** (Equation 3.18). The scatters and the distributions we obtain are identical to the Gaussian case. Table 3.2 summarises the Errors in conservation of different metrics under both types of projections. Given that the JL lemma guarantees an $\epsilon = \mathcal{O}\left(\sqrt{\frac{\log n}{k}}\right)$ embedding for a reduced dimension of k, it is interesting to see whether the relative errors scale accordingly for our set of reduced dimensions. The Sparse Random Projection thus gives us a computationally faster way to apply the RP technique.

Table 3.2: Comparison of Errors: Gaussian vs Sparse Projections

	$\frac{1}{3}\sqrt{\frac{\log n}{k}}$	Norm		Euclidean Distance		Inner Product	
	Estimated	Gaussian	Sparse	Gaussian	Sparse	Gaussian	Sparse
k1	0.007	0.005	0.005	0.006	0.005	0.008	0.008
k2	0.009	0.007	0.006	0.007	0.007	0.010	0.009
k3	0.012	0.010	0.009	0.010	0.010	0.014	0.013
k4	0.020	0.015	0.015	0.015	0.015	0.022	0.022

Relative Distortion of Inner Products

Corollary 3.2.2 gives a relative distortion bound for Inner Products under Random Projections. The bound is strong for strongly correlated or anti-correlated (opposite) pairs of data vectors, and weak for uncorrelated or orthogonal pairs. This fact has important implications for conserving match values between pairs of Gravitational Wave templates, which we shall see in chapter 4. In this section, we attempt to verify and reproduce the relative distortion bounds under the framework for Random Projections being developed.

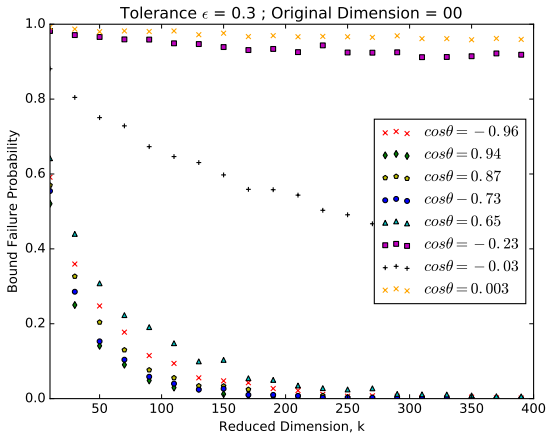
We experiment with a set of 8 normalised vectors in 300 dimensions, distributed such that the following 8 pairs amongst them display a range of Inner Product values from -1.0 to 1.0. We shall check how strongly the bounds depend on the relative Inner Product, and how the Inner Products are conserved as we reduce the dimension from 300 to 10. The way we calculate the associated probabilities, is given below:

1. Take any one pair of vectors x and y in 300 dimensions. Their dot product here is P.
2. Choose a reduced dimension of k = 100.
3. Generate 2000 independent Random Projection matrices going from N=300 to k=100.
4. Project x and y into each of these 2000 100-dim subspaces and calculate the dot product in that space (Call this Q)
5. For each of the 2000 instances, determine the ratio of the dot product in reduced space to that of the original space, i.e. Q/P. Pick a tolerence value of ϵ and count it if Q/P lies within $(1 \pm \epsilon)$.

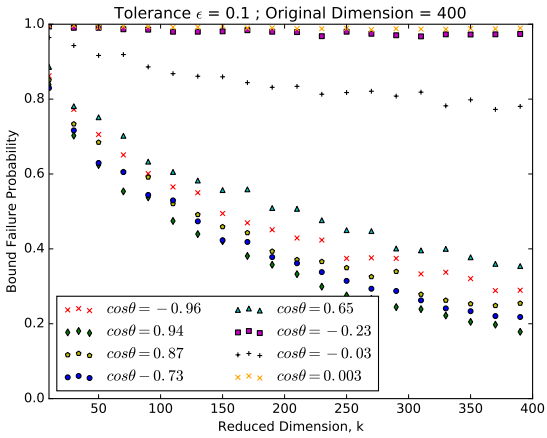
6. The empirical probability for the conservation of the dot product when the space is reduced to 100 dimensions is then given by the number of times Q/P is within $(1 \pm \epsilon)$, divided by 2000.

7. The **bound failure probability** is simply 1 subtracted by the probability calculated above.

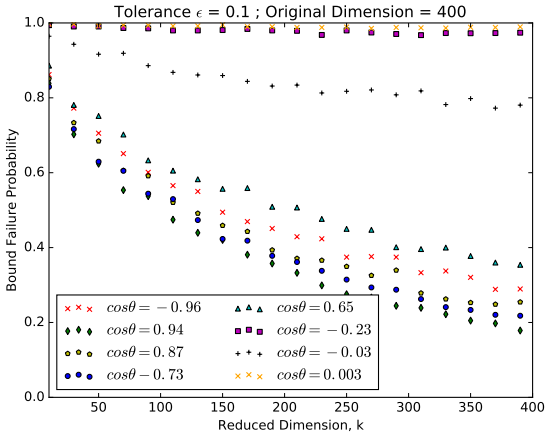
The above set of calculations are repeated for all 8 vector pairs, and all values of reduced dimensions ranging from 10 to 300 in an increment of 10.



(a) $\epsilon = 0.03$



(b) $\epsilon = 0.01$



(c) $\epsilon = 0.005$

Figure 3.6: Probabilities for Inner Product Conservation for different tolerances. We observe that orthogonal pairs are conserved with a far less probability compared to adjacent/anti-adjacent pairs

Chapter 4

Random Projections of Gravitational Wave Templates

We have seen that Random Projections in accordance with the theory for datasets uniformly spread in a high dimensional space. We now turn our attention towards applying them to reduce the time sample space of Gravitational Wave templates. In this chapter, we try to address the following questions: Do Random Projections work at all for datasets containing periodic time series? If yes, what is the parameter space to choose our set of templates from? How closely should they populate the space? What number of templates should we pick, and what are the associated Johnson Lindenstrauss conservation bounds? What degree of dimensionality reduction could we achieve? Finally, we shall see results of how good a job Random Projections do in preserving template information in a lower dimension, and discuss the time of operation for projecting with different RP matrices.

4.1 Templates

Before we dig deep into their dimensions, let us look at what kind of Gravitational Wave templates we are working with. We use inspiral-only binary waveforms, modelled up to the Innermost Stable Circular Orbit (ISCO). The lower cutoff frequency is set at aLIGOs sensitivity threshold, 10 Hz. The generated waveforms evolve up to their ISCO frequencies or the upper cutoff frequency of 2048 Hz. whichever is higher. This also sets our sampling

rate at twice this value, 4096 Hz., based on the Nyquist criterion.

The waveforms are generated using the Post-Newtonian (PN) approximation [8], which computes the evolution of the orbital phase $\phi(t)$ of the binary by perturbative expansion of its characteristic velocity $v = (\pi MF)^{1/3}$ (M = Total Mass of the binary system, F = Frequency of the Gravitational Wave). This evolution is characterised by the following pair of differential equations:

$$\phi(t) = \frac{v^3}{M} \tag{4.1}$$

$$\dot{v} = \frac{-\mathcal{F}(v)}{E'(v)} \tag{4.2}$$

Where $\mathcal{F}(v)$ is the Gravitational Wave luminosity and $E'(v)$ is the derivative of the binding energy with respect to v. Different models of PN waveforms arise owing to different ways of expanding the ratio $\frac{-\mathcal{F}(v)}{E'(v)}$. The waveform model we use is TaylorT4, in which the rational polynomial $\frac{-\mathcal{F}(v)}{E'(v)}$ is expanded in v to a consistent PN order.

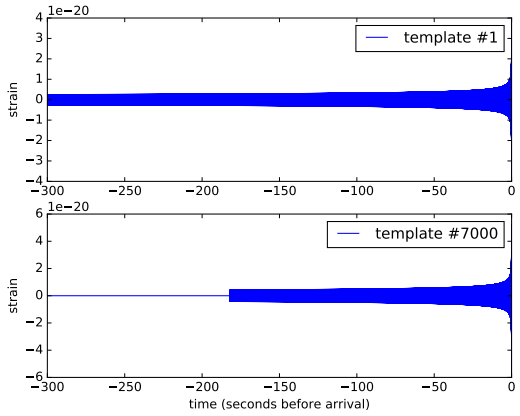
Next, we determine the intrinsic parameter ranges of the binary to choose our set of templates from. How to determine the individual mass parameters? For this, let us go back to our motivation: We wish to develop a fast and computationally cheap search technique by reducing the time-sample dimension. One of its important outcomes would be to generate quick alerts for possible follow-up observations, as discussed in chapter 1. For this, at least one of the binary components must be a Neutron Star (NS). This requirement has an added advantage. The chirp time of a binary is inversely proportional to its total mass. This is because more massive systems tend to lose their energy through GW emission faster. Hence, Black Hole-Black Hole (BH-BH) signals last for a very short time, of the order of seconds, and have a relatively lower number of time samples in their corresponding templates. At the other end, some NS-NS signals may last for over 20 minutes. Random Projections are expected to give more elegant results if the templates have a high dimensionality in the first place, and hence having a Neutron Star in our binary systems is useful.

We now look at NS-NS and NS-BH systems. An important factor to consider is how closely the templates populate the parameter space. One metric used to determine this is the match. It tells us how alike two templates are, and what is the degree of overlap between them. If can defined as the maximum inner product (overlap) value obtained as we slide two templates relatively over the difference in their chirp times. In order to ensure that we do

not miss out on signals or gain false positives, the template space is populated such that the match between any two adjacent templates is 0.97 i.e. they have a 97% overlap[25]. This enables us to cover the parameter space as effectively as possible.

The minimal match requirement leads to an inverse relation of the density of template population at a given mass range with the total mass in that range. This means that the lighter NS-NS regime has a greater number density of templates as compared to NS-BH or BH-BH. In other words, if we want to include a fixed number of templates in our templatebank, we would be able to cover a larger portion of the parameter space for NS-BH binaries as compared to the NS-NS ones.

Is there such a limit to the number of templates that we can include in our template bank? Yes. It arises because of a limitation in using one Random Projection matrix across an entire bank. Refer to the adjacent figure which shows the shortest and the longest waveforms from a particular bank of 7000 waveforms. The difference in their chirp times is almost 2 minutes. If we use Random Projections to go down to a particular dimension k, the shape of the projection matrix would be the length of the longest template times k. In this case, the longest waveform would be transformed effectively. However, the shorter one would waste almost half of the projection components operating upon the zero-padded section in its initial portion. This might result in significant loss of information.

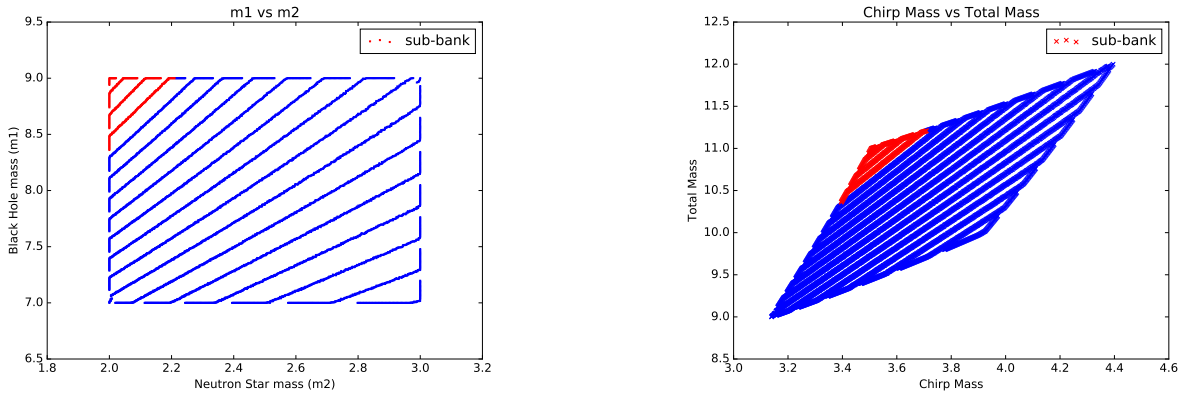


Hence, there is a need to divide the entire templatebank into sub-banks, such that there is little difference in chirp-times of the shortest and longest waveforms within a sub-bank. Given that we had to start with a limited number of templates, and wanted to cover a sizeable chunk of the parameter space, we fixed a Neutron Star-Black Hole (NS-BH) templatebank having the following mass parameters:

- **Neutron Star mass range:** $(m1) = 2M_{\odot} - 3M_{\odot}$
- **Black Hole mass range:** $(m2) = 7M_{\odot} - 9M_{\odot}$

We define the following additional mass parameters for a binary:

- **Total Mass:** $M = m1 + m2$
- **Symmetric Mass Ratio:** $\eta = \frac{m1 \cdot m2}{M^2}$
- **Chirp Mass:** $\mathcal{M} = \eta^{3/5} M$
  This governs the rate of frequency sweep at PN-order.



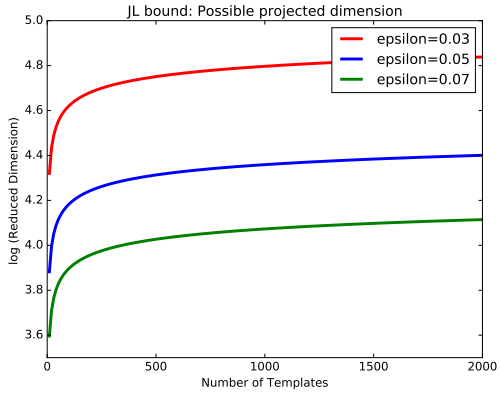
(a) Distribution of Templates in the component mass space

(b) Distribution of Templates in the Total Mass-Chirp Mass space

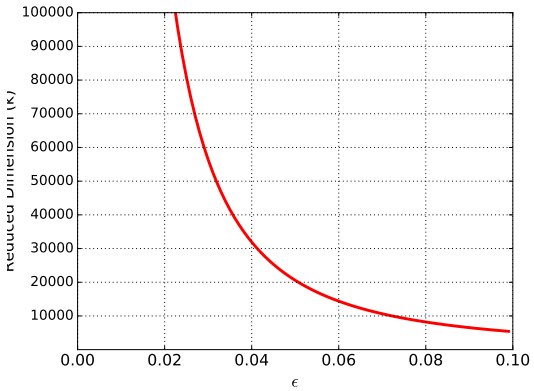
Figure 4.1: Our chosen NS-BH Template Bank. The sub-bank for one Random Projection is highlighted in red.

Figure 4.1 displays the distribution of templates in the template bank. It has a total of around 8000 templates, generated using PYCBC[28]. From this, we choose a sub-bank having mass ranges $m1 \in [8.3, 9.0], m2 \in [2.0, 2.2]$ to test Random Projections. This bank was chosen such that the largest chirp time difference between any pair of templates within it was only 20 seconds. The sub-bank is highlighted red in the figure. The reason we chose 500 as the number of templates in the sub-bank, is elucidated in the next section.

4.2 Targeted Reduced Dimension



(a) JL bounds for the reduced dimension corresponding to different numbers of data vectors in the vector space



(b) JL bounds for the reduced dimension for projecting 500 vectors, corresponding to different error tolerances

Figure 4.2: Determining the degree of Dimensionality Reduction

We shall now proceed to determine what degree of dimensionality reduction we can achieve for the templates in our sub-bank. The ambient dimensionality i.e. the number of time samples in each template is 1,270,805 or ~ 1.2 million. However, this fact is irrelevant. Recall that the JL estimate for reduced dimension (Equation 3.2) depends only on the number of data vectors or templates (n) occupying the data space:

$$k = \frac{24 \log n}{2\epsilon^2 + 3\epsilon^3} \tag{4.3}$$

In Figure 4.2a, we plot curves for the targeted reduced dimension based on the number of templates in a sub-bank, corresponding to different values of expected distortion (error limit). We observe that these curves begin to saturate for $n \geq 500$. This means that even if we populate the template space beyond this, we should not expect to achieve a greater reduction in dimensionality for a given error tolerance.

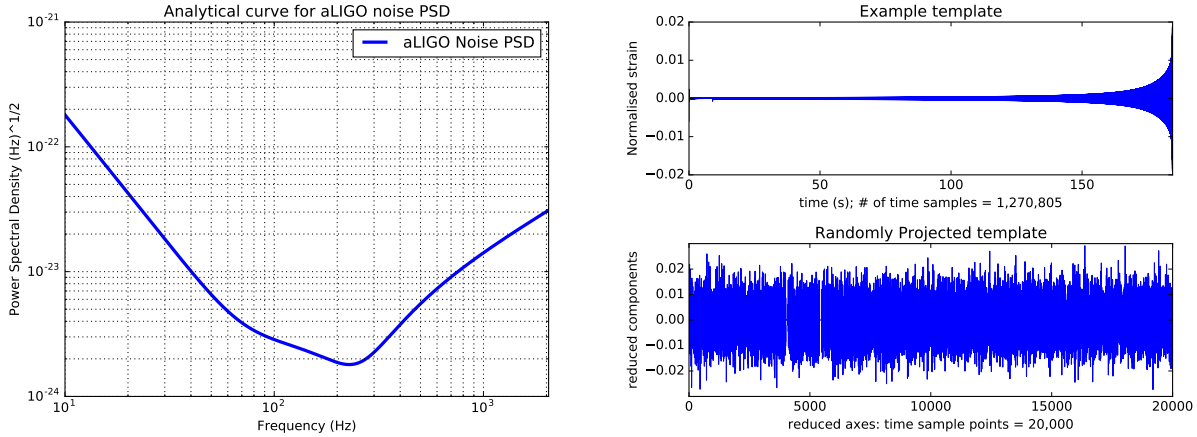
That is why we chose to start with 500 NS-BH templates. Next, we needed to set a value for the error tolerance, ϵ to determine the reduced dimension(k). Figure 4.2b displays the relation between ϵ and k for 500 data points. We chose a value of $\epsilon = 5$ and the corresponding reduced dimension of $k = 20000$, a 60-fold reduction from the original template dimension.

4.3 Whitening, Projection and Match Calculation

Before projecting the templates, we whitened them with the following analytical form of the advanced LIGO noise power spectral density[5], plotted in Figure 4.3a.

$$S_h(f(x)) = \frac{10^{-49}}{Hz.}\left[x^{-4.14} - 5^{-x^2} + 111\left(\frac{1 - x^2 - x^4/2}{1 + x^2/2}\right)\right] \tag{4.4}$$

Here $x = f/f_0$; $f_0 = 215.0$ Hz. being the frequency of maximum sensitivity for aLIGO.



(a) aLIGO: Analytical PSD curve

(b) A template and its projected vector

Figure 4.3

We use a [1270805 × 20000] Gaussian matrix for our Random Projection. Figure 4.3b shows an example whitened template, and how its randomly projected data vector looks if plotted as a time series. Doing the latter is downright wrong since the reduced templates no longer constitute time series in the conventional sense. However, take a moment to realise that the seemingly noisy object in the second plot might effectively preserve all the information present in the well-defined template in the first plot. We are claiming that this randomly projected construct is in every bit as good as a painstackingly modelled template.

The match value for a pair of templates is calculated by determining the maximum overlap between the two templates as the shorter one is slid over the longer one over the difference in their chirp times. At this point, we project both templates to the RP subspace, and compute their direct inner product. This is the value of their match in the RP subspace.

4.4 Results

We shall now look at the results of projecting our 500 template sub-bank to a reduced subspace of 20000 dimensions. In order to verify whether the geometry of this high-dimensional space is preserved under RP, we shall look at pairwise match values as a metric to determine the relative closeness of pairs of templates. If the match values are preserved in the reduced subspace within the given tolerance, we can conclude that the templates are structured in a manner similar to their distribution in the full space.

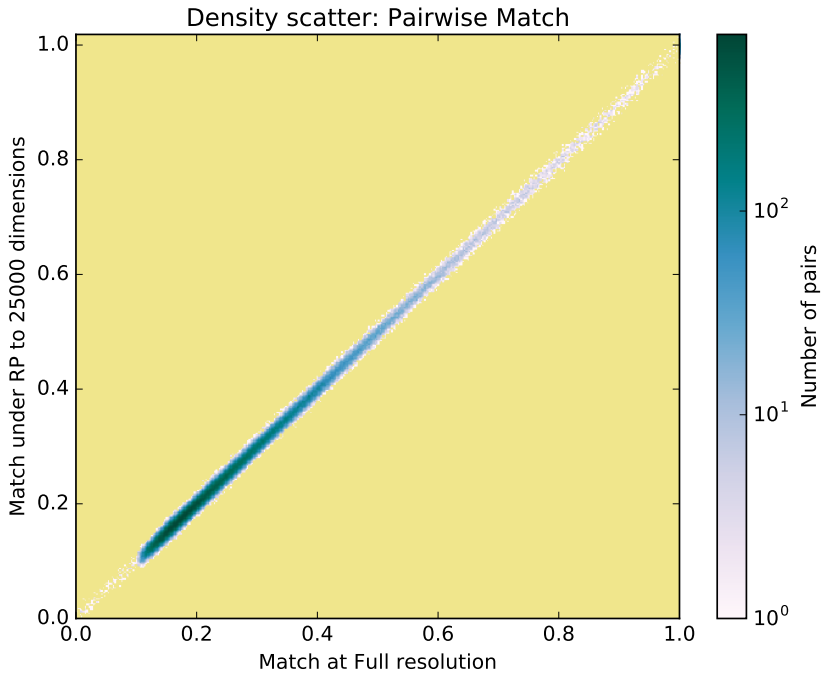


Figure 4.4: The density scatter of pairwise match values under Random Projection shows a near-perfect linear trend

Figure 4.4 is a density scatter plot of match values in the original dimension against those in the reduced subspace. We see that they match very closely, lying along the 45° line. We observe that Random Projections do a much better job of conserving the inner product relations in this dataset compared to the uniform dataset in lower dimensions we experimented with in chapter 3. The high ambient dimensionality of the data enables us to better replicate the JL bounds in reducing dimensionality. It is interesting to see that the scatter is not too distorted for inner products greater than 0.5, as predicted by the weaker relative distortion bound.

In Figure 4.5, we see the distribution of pairwise match values across the dataset. Both distributions match very closely, indicating a strong preservation of template space structure. This means that templates previously far apart do not have greater overlaps in the reduced space, thereby obviating the worry of an increased rate of false positives.

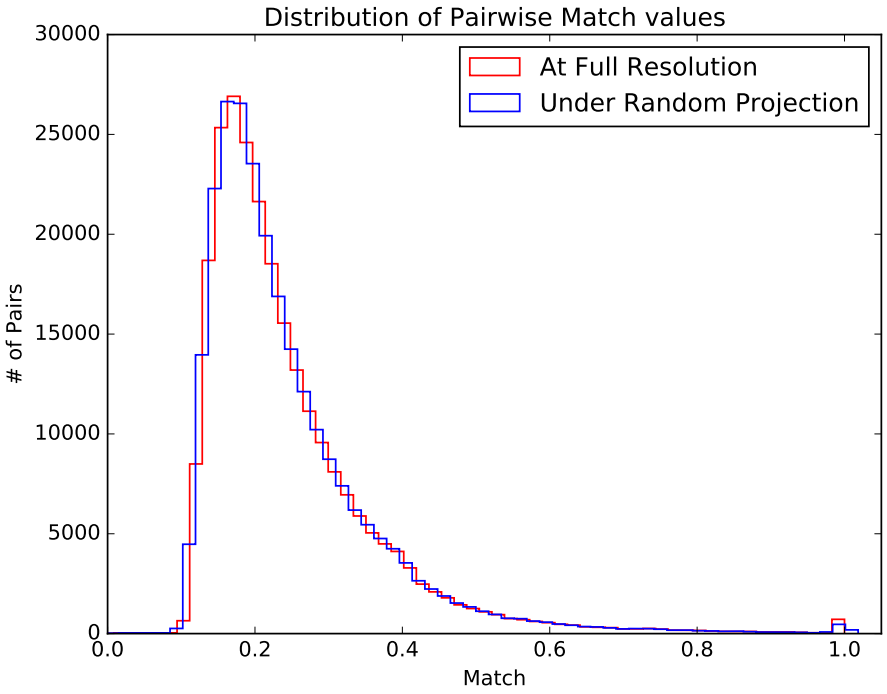


Figure 4.5: A comparison of the pairwise Match distribution at full resolution and under Random Projection

Another way of verifying the conservation of template properties is to look at the autocorrelation function of a single template around its time of arrival (peak value). The adjacent plot shows a perfect match between the autocorrelation of a template and its projected vector.

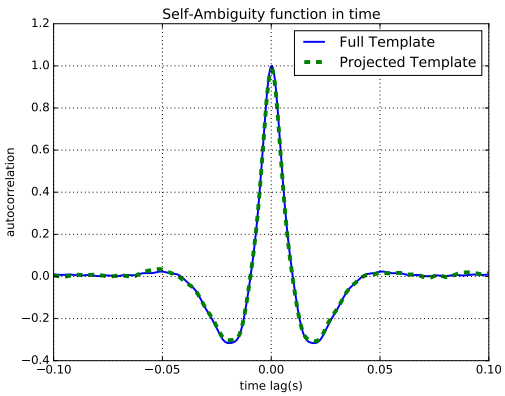


Figure 4.6: Autocorrelation function for a template and its projected vector

4.4.1 Signal Detection in the presence of Noise

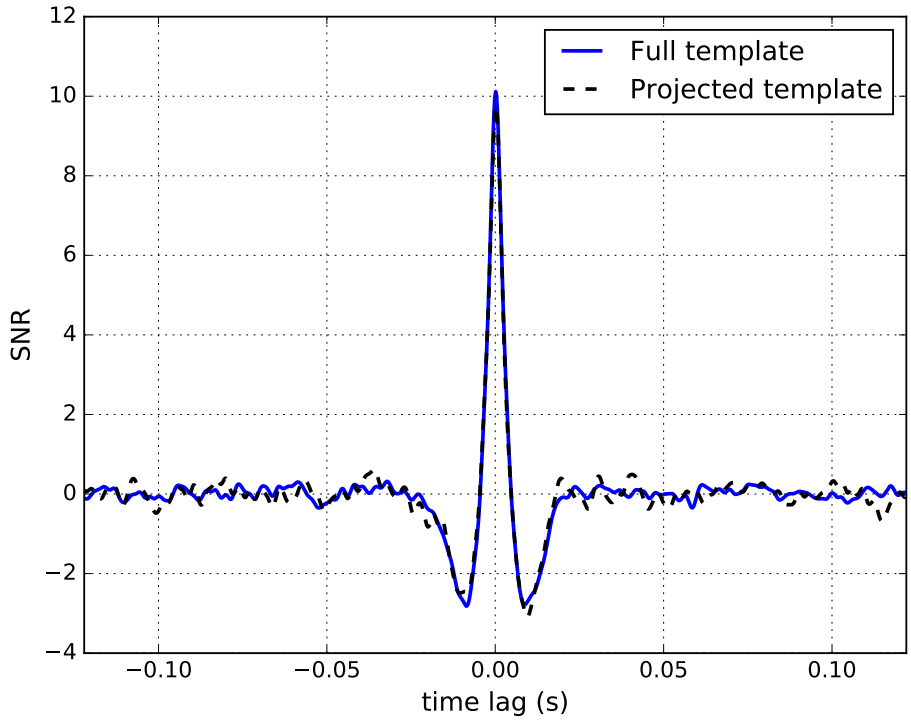


Figure 4.7: Cross-correlation of a template with a simulated noisy data generated from it.

We took template # 203 from our sub-bank and simulated data by adding Gaussian colored noise of magnitude 100 to it. The chunk of data was whitened using the aLIGO PSD. The data was generated such that a cross-correlation with the same template (# 203) would give an SNR of 10. The SNR time series for the full resolution correlation is given by the blue line in Figure 4.7. We correlated the same chunk of data under Random Projection and compared it to the RP-reduced template # 203. We observe that the black dashed SNR time series corresponding to it exactly matches the original time series at the SNR peak at zero time shift. For other time lags where the value of the SNR is close to zero, the Randomly Projected matches fluctuate differently, with a slightly greater distortion. This is expected and explained by the relative distortion bounds on orthogonal Inner Products 3.2.1.

4.4.2 Errors in SNR conservation

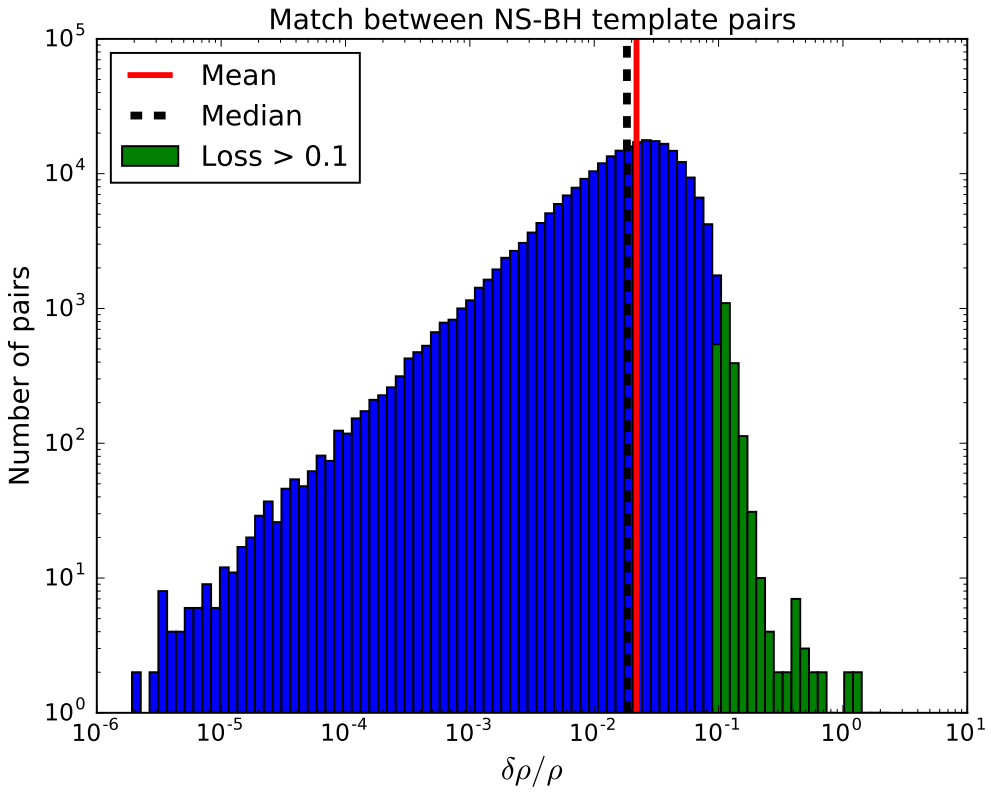


Figure 4.8: Relative loss of SNR under Random Projection

Figure 4.8 shows the relative loss of match values under Random Projection. The mean of the distribution is -0.0051, with a standard deviation of 0.039 (just under 4%). The distortion is hence less than the 0.05 limit we set using the JL bound. The degree of dimensionality reduction we achieve is impressive, with over a 50-fold decrease. Contrast this to using subsampling (2.2.2) at its coarsest to reduce the number of time points to around 20000. Doing so would mean sampling the latter stages of the waveform far below the Nyquist rate and losing information. With Random Projection, we achieve the same without losing information. The 4% loss in accuracy means that we might miss out on that proportion of candidate events in a single detector. This value is skewed due to some pairs showing deviations larger than 0.1 (highlighted in green). Let us look at which pairs show such deviations.

A word on the outliers

Figure 4.8 shows a number of pairs which show a relative pairwise SNR loss of greater than 0.1. In fact, there are 24302 such pairs. Losing 10% of SNR or more is detrimental. What causes these pairs to deviate in the projection space? The reason this happens, is because all these pairs of vectors are relatively close to being orthogonal in the original space. As we saw in section 3.2.1, the JL lemma does not guarantee a strong inner product conservation for values that are originally close to being orthogonal. Hence, we observe these outliers. Figure 4.9 shows the distribution of their match values, which has a mean value of 0.01 with a deviation of 0.005. On the brighter side, it is unlikely that these outliers would affect a search over the templatebank. This is because even if the signal mimicked one of the templates in these pairs, a 10% deviation in the SNR would likely not show up as a false positive, since they being nearly orthogonal ensures that the SNR is obtained low in the first place.

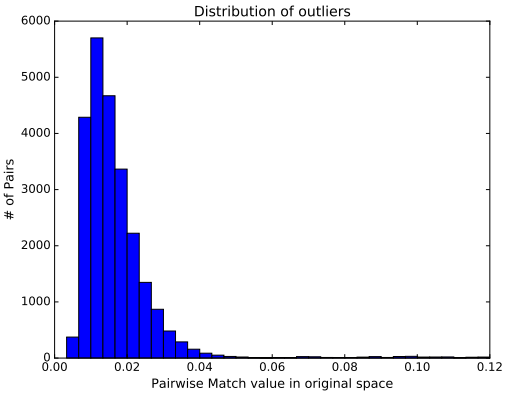
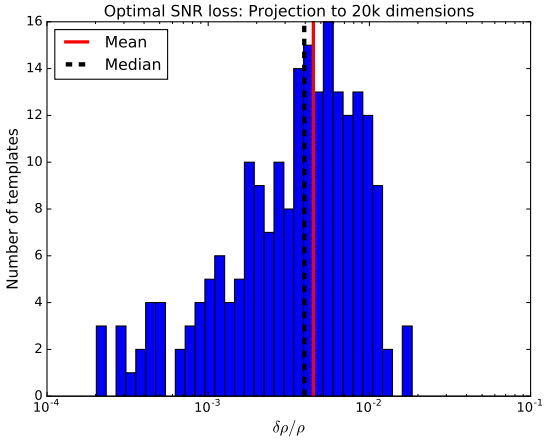


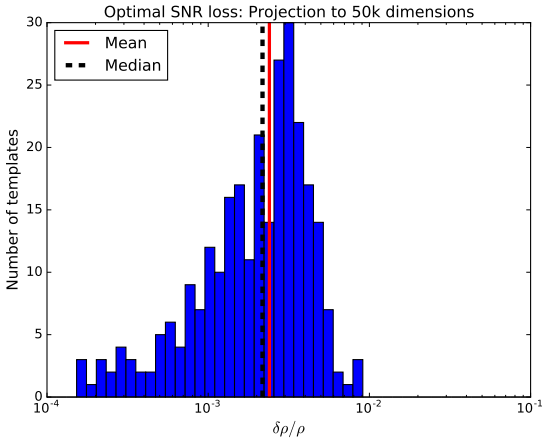
Figure 4.9

4.4.3 Relative loss of Optimal SNR

The most optimal SNR is obtained through a cross-correlation of a template with a signal comprised of the same template. Without added noise, if the template is normalised, the value of the optimal SNR is 1.0, the norm of the template. We now investigate how the optimal SNR is affected under Random Projection. This reduces to checking the norm conservation. Figures 4.10a and 4.10b represent this conservation for a Random Projection to 20000 and 50000 dimensions respectively. Ideally, we would like to have all values of the relative loss to be under 0.01. This is easily satisfied for a projection to 50000 dimensions (Mean deviation 0.004). For 20000 dimensions (Mean deviation = 0.002), we observe that the SNR loss values just spill over this threshold for a handful of templates.



(a) Relative SNR loss with a reduced dimension of 20000



(b) Relative SNR loss with a reduced dimension of 50000

Figure 4.10

What is a good trade-off between setting a particular threshold for limiting the SNR loss, and reducing the computational complexity by projecting lower in dimension? This question needs to be addressed while building a search pipeline using Random Projections. As a first estimate, an maximum error of less than 0.01 is good for a low-latency offline search where a timely trigger generation is of prime importance.

Chapter 5

Applying Random Projections to Data streams

Having tested the immense power of Random Projections to preserve the information contained within a Gravitational wave time domain template in far fewer dimensions, we turn to the problem of applying these *reduced* templates in time domain searches. We first discuss the challenges involved in projecting data streams using Random Projections, and then present an algorithm to do this efficiently. We analyse for what regimes of a Gravitational Wave search can our algorithm be applied to: Can we use it in an online search? What plan of action can we implement for targeted searches in an offline mode? Finally, we investigate an attractive scheme to combine Random Projections and SVD in order to carry out searches over entire template banks.

The Challenge

This problem of projecting data streams turns out to be extremely non-trivial, primarily because of the fact that *we cannot carry out time domain convolutions using RP-reduced templates*. The reason for this limitation is clear: a convolution of two time series involves a series of *sliding inner products* of one template over the other. We have seen that Random Projections conserve the direct inner product between two templates very well. However, if we shift one of the templates by a time step in a convolution, it no longer represents the same

template-vector in our time-sample space. By time-shifting, we are essentially performing a rotation of the template vector, and hence changing its orientation completely. This of course changes its inner product with respect to the second template which does not undergo the same rotation.

5.1 Data Streams and Convolutions

Fig [5.1] illustrates the operation of matched filtering using a single template$h(t)$. Let the length of the whitened template be d time samples, also known as its chirp time. We have a data stream $x(t)$of length $l > d$ time samples. This whitened data stream consists of a stationary gaussian noise series $n(t)$ with the signal (approximated as the template with added noise, whitened using 4.4) present somewhere inside it: $x(t) = h(t) + n(t)$. The data can be visualised using a series blocks, each block representing one time sample.

The time domain correlation, represented by Equation 1.13:

$$\rho_\alpha[\tau] = \sum_{t=0}^{l-1} x[\tau - t]h[t] \tag{5.1}$$

is a sliding inner product operation over successive time steps τ. Figure 5.1 depicts this operation at full resolution. We have a sliding window which holds the template which we want to match the data with. Convolution involves sliding this window over the entire data stream, and calculating the inner product at each instance. When the sliding window passes exactly across the window where the signal is present in the data, the template and the signal completely overlap, giving the maximum match value. This time instant represents the time of arrival $\tau_{arrival}$ of the signal.

5.2 Sketches using Random Projections

We can implement Random Projections to speed up the above operation. How do we project the data stream into the Random Projection subspace? The idea is to form *sketches* of the data stream: a sketch being the Random Projection vector corresponding to a sliding

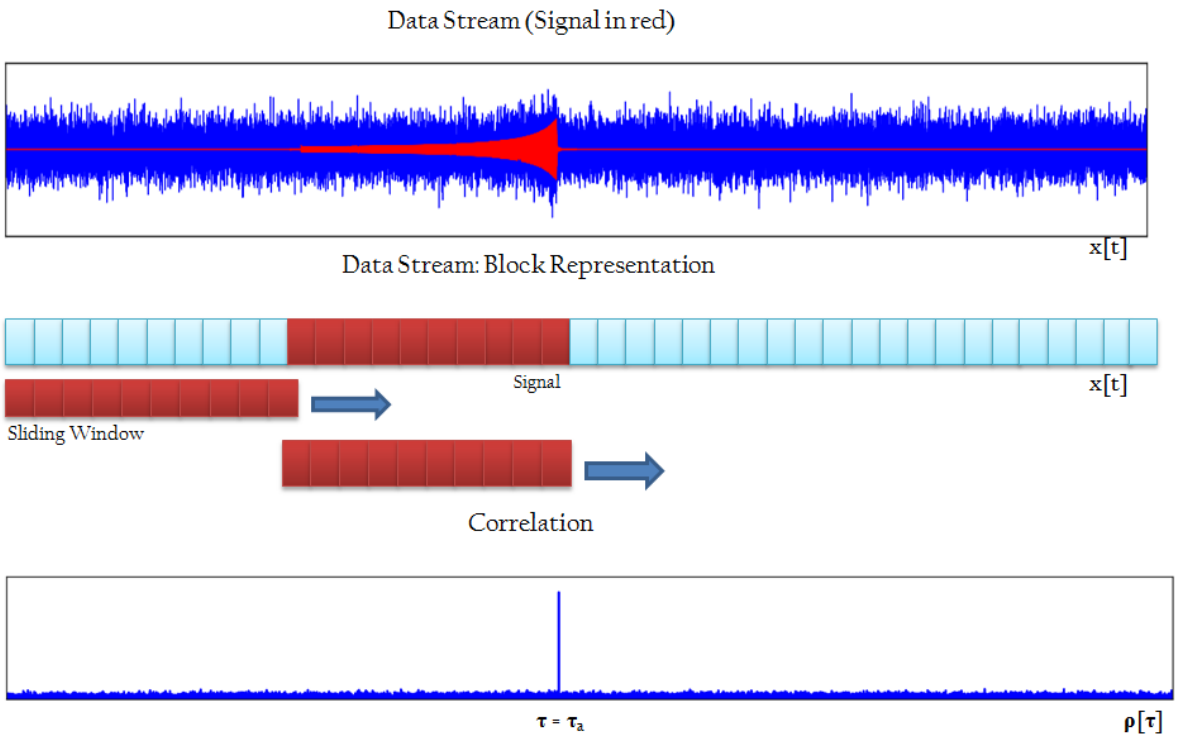


Figure 5.1: Time Domain Correlation

window in the convolution. Figure 5.2 illustrates the procedure. We'll begin by elucidating thte different periods of dimension involved in the algorithm:

- **l** : The number of time samples or blocks in the data stream.
- **d** : The length of a sliding window, equal to the length of templates in the sub-bank.
- **k** : The length of a sketch, i.e. the projected dimension of a template/sliding window. This is of the $\mathcal{O}(\log d)$.
- **n** : The number of time steps a sliding window slides over during a convolution. This is equal to $(l + d - 1)$. This is also the number of sketch vectors generated.

The time domain convolution requires $n = l + d - 1$ sliding window inner product operations. The Sketching approach involves projecting all these sliding windows using a Random

Projection matrix, to k dimensions. These sketches will then be compared with the RP-reduced template. The convolution will now involve n inner products of the n sketches with the reduced template, where each inner product will have k components. Hence, the complexity of the time domain matched filtering reduces from $\mathcal{O}(ld)$ to $\mathcal{O}(lk) = \mathcal{O}(l \log d)$.

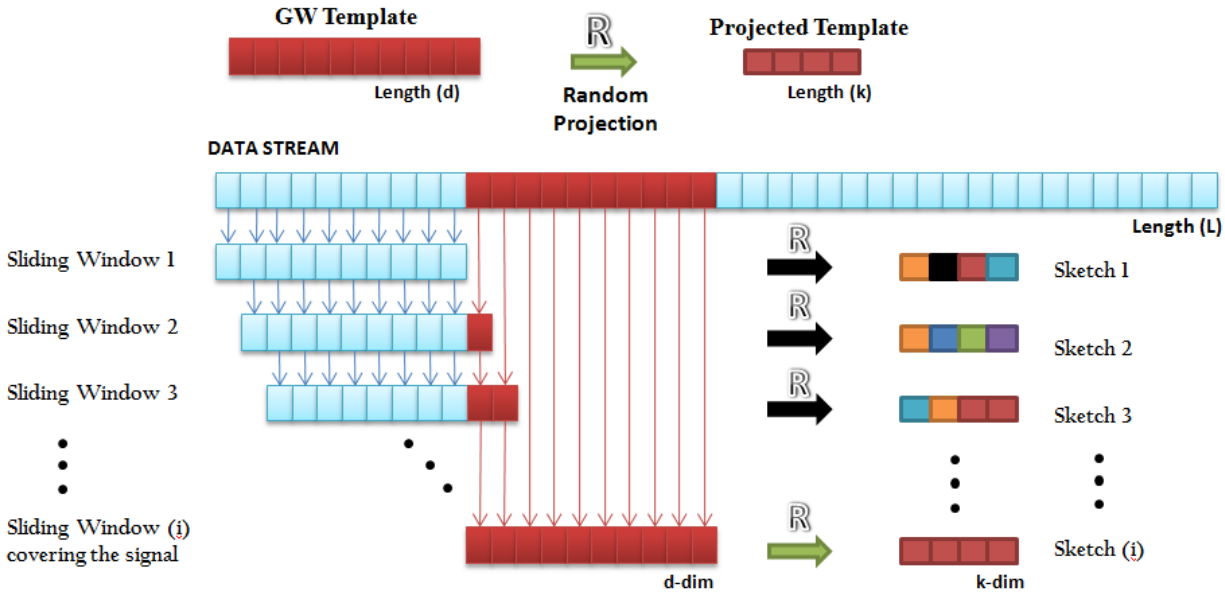


Figure 5.2: Random Sketching

This makes it faster than the corresponding operation done at full resolution. But there's a caveat: we have to include the computational cost of calculating these sketch vectors. The Random Projection to form the sketch of one sliding window involves a matrix multiplication of the $k \times d$ projection matrix and the n dimensional window. This process has a complexity of $\mathcal{O}(kd)$, and hence an $\mathcal{O}(lkd)$ for l such sketches. This eventually ends up being costlier than a time domain convolution involving complete templates.

Fast Sketch Computations

P. Indyk et al. [18] suggest using the Cooley-Tukey radix-2 algorithm[29] for a fast computation of the sketches. Popularly known as the Fast Fourier Transform (FFT) algorithm, it can be used to reduce the complexity involved in the matrix multiplication step of the Random Projection. It exploits the fact that the d-dimensional sliding windows can be represented as a single-shift circulant matrix, which is diagonalised by multiplication with the Fourier Vandermonde matrix in $\mathcal{O}(\log d)$ steps. One such operation gives us the first of the k components of the projected vector in one go for all the d sketches. Repeating this d times gives us all the sketches in $\mathcal{O}(d \log d)$ time, as opposed to $\mathcal{O}(d^2)$ for a naive projection. Once the sketches are ready, the match calculation requires a $\mathcal{O}(dk) = \mathcal{O}(d \log d)$ time. This approach is also discussed in [30].

Another way to speed up this process is to use sparse Random Projections, introduced in section 3.2.2. These matrices only require k additions per projection, and hence all the sketches can be computed in $\mathcal{O}(kl)$ time. Table 5.1 summarises the theoretical complexity of all the aforementioned processes. The sparse projection is expected to give the fastest results in principle, and we are working towards developing a pipeline to integrate it into a search.

Table 5.1

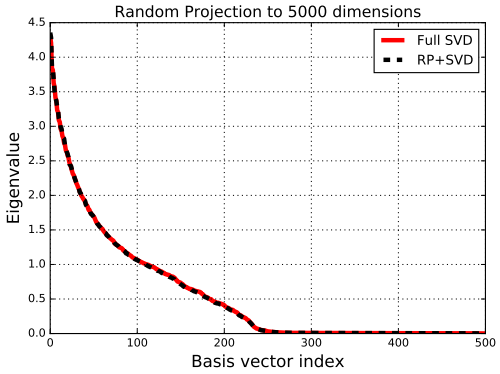
Method	Complexity
Full Time Domain	$\mathcal{O}(ld)$
Naive Random Sketches	$\mathcal{O}(ld \log d)$
Cooley-Tukey Random Sketches	$\mathcal{O}(l \log^2 d)$
Sparse Random Sketches	$\mathcal{O}(l \log d)$

5.3 Combining Random Projections with SVD

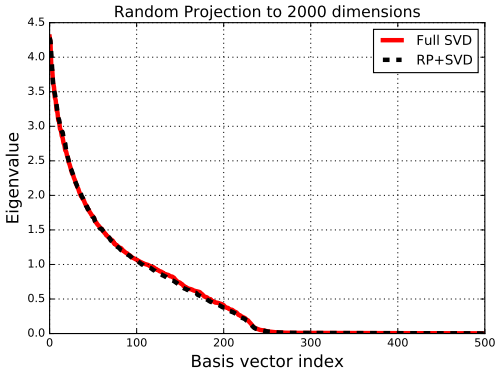
The Random Projection technique offers a promising way of reducing the time-dimension. It achieves a far greater decrease in complexity compared to the existing technique of frequency-based subsampling. While this project started with the aim of finding a faster alternative to the existing SVD-based LLOID algorithm, we found that both approaches are fundamentally different. SVD reconstructs the SNR by using a truncated basis of principal vectors of the

templates. Random Projections projects all the templates onto a subspace of the ambient time-sample dimension. We now wonder, can we tame the beast we set out to conquer? Can Random Projections be used in conjunction with SVD to create an algorithm which reduces dimensionality optimally in *both* the template and the time dimensions?

To investigate this, we perform experiments using the same set of whitened templates from the NS-BH sub-bank, but resampled to 10,000 time points. We call this the full resolution template bank. Next, we project the entire bank using Gaussian Random Projection to two dimensions: k1 = 5000 and k2 = 2000. We call these two reduced template banks RPbankA, having 5000 dimensions and RPbankB, having 2000 dimensions. We test whether we can perform a Singular Value Decomposition of these reduced banks which gives results identical to an SVD of the original template bank.



(a) Eigenvalue comparison with 5000-dim projected vectors



(b) Eigenvalue comparison with 2000-dim projected vectors

Figure 5.3

We compare the distribution of Singular Values obtained for the full template bank against those for each of the reduced banks in Figure 5.3b. The distributions have a very precise overlap for both reduced banks. This is quite a significant and a rather non-intuitive result. The SVD on a template bank, as explained in chapter 2, works to exploit the degeneracies specific to the phase evolution of adjacent templates. It is indeed interesting to note that this technique works even for the Randomly Projected template vectors, which are no longer time series but mathematical constructs which preserve distance relations within pairs of templates. In a broad sense, this can be explained by the fact that Random Projections also conserve how the data is geometrically distributed in a reduced dimension, and hence do not change in what direction the principal components lie.

RP + SVD for signal detection

Given that the principal components of the RP-reduced templates are identical to those of the original templates, can we apply SVD to reconstruct the projected templates and use them to detect signals? Here we present a primitive analysis of how effectively Random Projections can combine with SVD to reduce the dimensionality both in the number of templates and time-points, and whether this gives results compatible with their counterparts at full resolution. We have generated sample data by adding noise to one of the templates (template # 315) as shown in the adjacent figure.

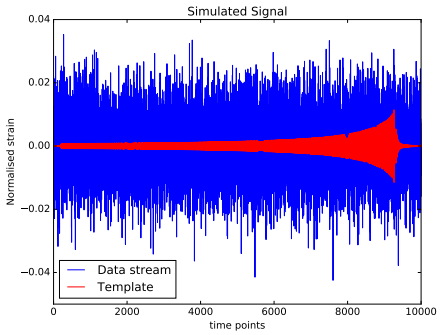
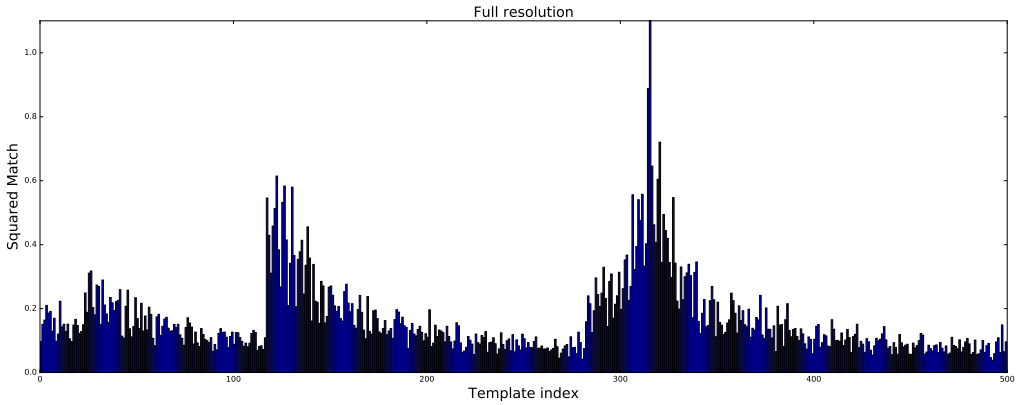


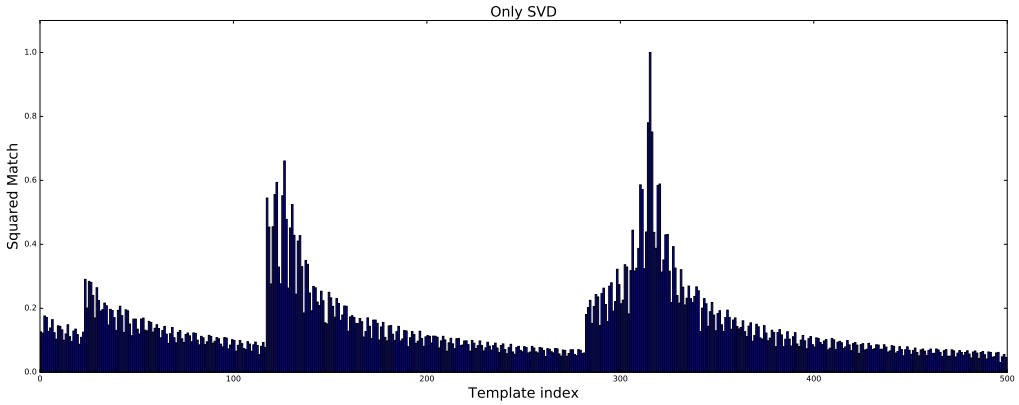
Table 5.2: Comparison times for SVD computation using Full vs. Reduced Templates

Template bank	Time taken to compute SVD
Full Templates (dimension: 10000)	13.22 seconds
RP-Templates, dimension = 5000	4.6 seconds
RP-Templates, dimension = 2000	517 milliseconds

Testing RP+SVD for a toy search model: In Figures 5.4 and 5.5, we plot the Maximum correlation values between the data, simulated by adding noise to template # 315, with the entire sub-bank of 500 templates. Each subplot displays these search plots for different kinds of reduced templates: 5.4a Original templates at full resolution, 5.4b Using only SVD on them, 5.5a Randomly Projected templates, and 5.5b SVD of Randomly Projected templates. It is evident that all methods produce comparable results, with the latter two being significantly faster to compute. Using RP in conjunction with SVD certainly holds good promise for future work.

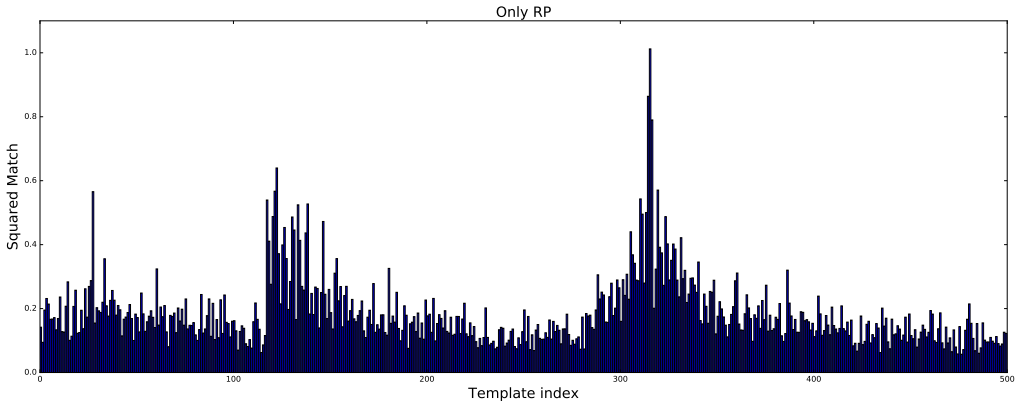


(a) Original Templates at Full resolution. No Dimensionality reduction along the template/time-point dimension

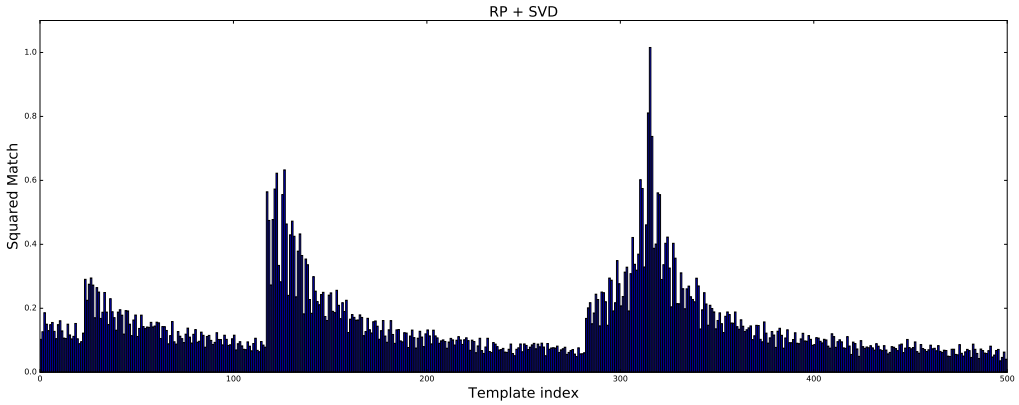


(b) Original templates reconstructed using 250 SVD basis vectors. Dimensionality reduction only along the template-dimension

Figure 5.4: Search at Full resolution, without and with using SVD



(a) Randomly Projected templates to 5000 time sample reduced-dimension. Dimensionality reduction only along the time-point dimension



(b) Randomly Projected templates to 5000 time samples, reconstructed using 250 SVD basis vectors. Dimensionality reduction along both the template and time-point dimensions

Figure 5.5: Search in the Random Projection subspace, without and with using SVD

Conclusions and Future Work

We have investigated how Random Projections can be implemented for reducing the dimensionality of CBC Gravitational Wave templates in terms of their time sampling points. We have found that we can reduce this dimensionality by over 60 times the original lengths of the templates, whilst preserving the information present in the time series. We tested the conservation of different metrics defining the closeness and the distribution of templates in the high-dimensional space. The accuracies obtained for this conservation under a high degree of reduction are within an error of 4%. These can be improved by projecting to higher dimensions following the bounds derived from the Johnson-Lindenstrauss Lemma.

The results obtained are very promising and this opens several avenues in furthering this approach to find applications in real time compact binary searches. Two such options have been explored in this work. The first one was to sketch an algorithm for applying Random Projection to scan data streams and carry out its complexity analysis. The second was to explore the possibility of combining Random Projection with Singular Value Decomposition of templates to obtain a two-fold dimensionality reduction.

In the future, we plan to implement the Random sketching algorithm and test it for searching signals in real data. This would give us a better idea of the errors involved and the rate of false positives in order to determine the efficacy of using RP-reduced templates in a search pipeline. It would also be interesting to see whether templates generated by including component spins from the binary system as additional intrinsic parameters can be reduced effectively using a combination of Random Projection and SVD.

Appendix A

Experiments with Sparse Random Projections

Here we present the results of the experiments with Random Projections from section 3.2.2, performed using Achlioptas' **Sparse Random Projection** matrix. Recall that this matrix has $2/3^{rd}$ of its entries zeros, and the rest partitioned equally into $\pm\sqrt{3}$. This greatly speedens up calculating the projection. Instead of performing a matrix-vector multiplication to project the vector, Sparse Projection only requires a couple of additions to compute each projected component. In this section, we verify the conservation of various data metrics from section3.3, using Sparse Random Projections.

1. **Norm Preservation**

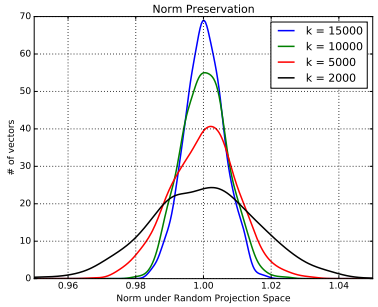
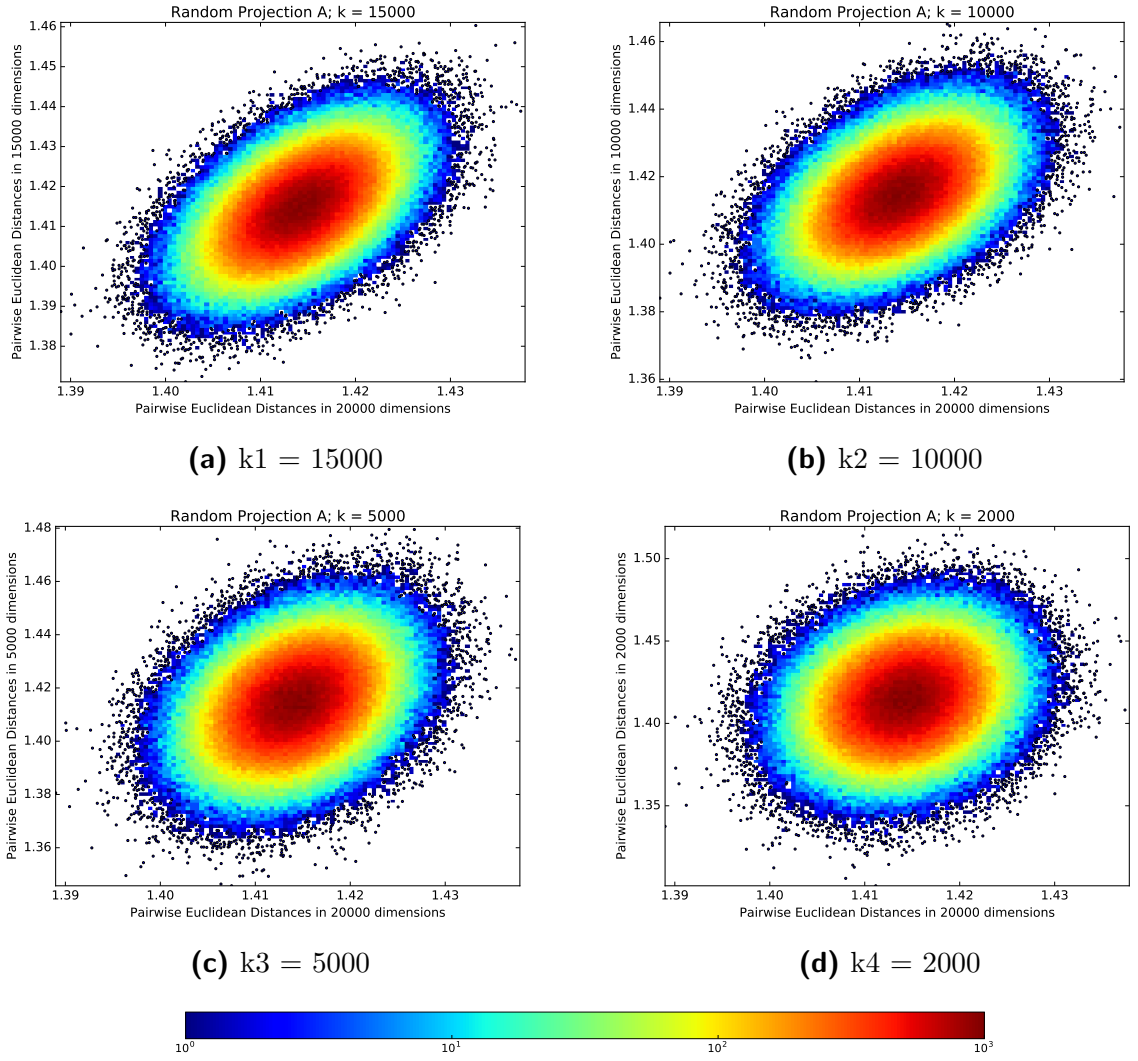


Figure A.1: Conservation of Norm

In addition to speeding up our calculations, Sparse projection matrices are also easier to store. Owing to their structure, we do not need to store every float value, which is either 0, $+\sqrt{3}$ or $-\sqrt{3}$. It thus suffices to store the indices corresponding to the non-zero entries and use them as masks while projection. The sparse nature of these matrices occupy much less storage memory compared to their Gaussian counterparts.

2. **Distance Preservation**

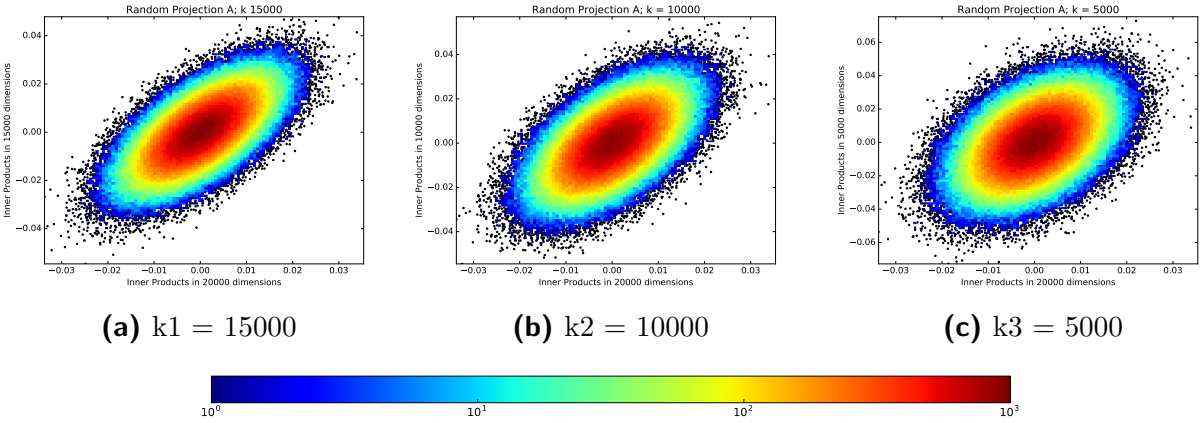


(a) k1 = 15000 **(b)** k2 = 10000

(c) k3 = 5000 **(d)** k4 = 2000

Figure A.2: Distance Preservation under Random Projection

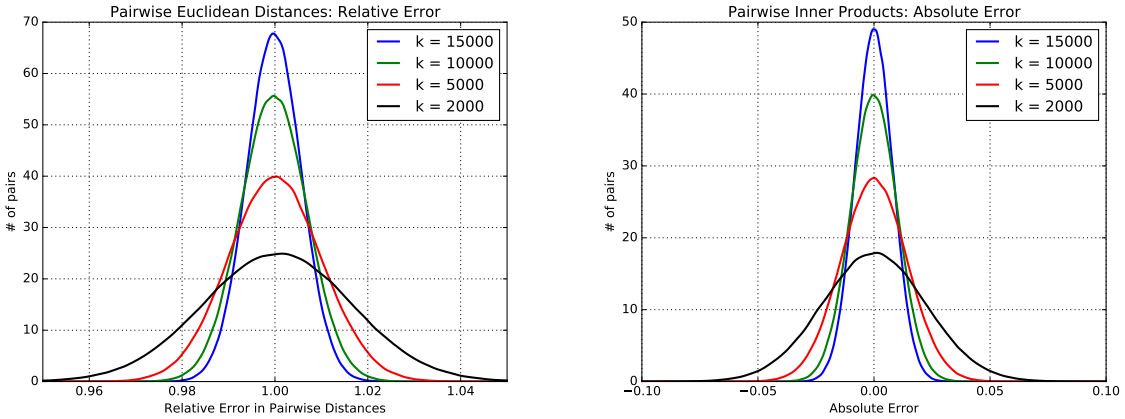
3. **Inner Product Preservation**



(a) k1 = 15000 **(b)** k2 = 10000 **(c)** k3 = 5000

Figure A.3: Inner Product Preservation under Random Projection

4. **Relative Errors**



(a) k1 = 15000 **(b)** k2 = 10000

We hence obersve the Sparse Random Projections give results identical to Gaussian Random Projections. This property will prove to be vital in speeding things up for an application of Random Projections in an online Gravitational Wave search pipeline.

Bibliography

[1] Rio 2016 official website. `https://www.rio2016.com/`.

[2] Website: Sounds of spacetime. `https://www.soundsofspacetime.org/`.

[3] Abbott, B. P., Abbott, R., Abbott, T., Abernathy, M., Acernese, F., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R., et al. (2016). Observation of gravitational waves from a binary black hole merger. *Physical review letters*, 116(6):061102.

[4] Achlioptas, D. (2001). Database-friendly random projections. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 274–281. ACM.

[5] Ajith, P. and Bose, S. (2009). Estimating the parameters of nonspinning binary black holes using ground-based gravitational-wave detectors: Statistical errors. *Phys. Rev. D*, 79:084032.

[6] Allen, B., Anderson, W. G., Brady, P. R., Brown, D. A., and Creighton, J. D. (2012). Findchirp: an algorithm for detection of gravitational waves from inspiraling compact binaries. *Physical Review D*, 85(12):122006.

[7] Bingham, E. and Mannila, H. (2001). Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 245–250. ACM.

[8] Buonanno, A., Iyer, B. R., Ochsner, E., Pan, Y., and Sathyaprakash, B. (2009). Comparison of post-newtonian templates for compact binary inspiral signals in gravitational-wave detectors. *Physical Review D*, 80(8):084043.

[9] Candes, E. J. and Romberg, J. K. (2005). Signal recovery from random projections. In *Electronic Imaging 2005*, pages 76–86. International Society for Optics and Photonics.

[10] Cannon, K., Cariou, R., Chapman, A., Crispin-Ortuzar, M., Fotopoulos, N., Frei, M., Hanna, C., Kara, E., Keppel, D., Liao, L., et al. (2012). Toward early-warning detection of gravitational waves from compact binary coalescence. *The Astrophysical Journal*, 748(2):136.

[11] Cannon, K., Chapman, A., Hanna, C., Keppel, D., Searle, A. C., and Weinstein, A. J. (2010). Singular value decomposition applied to compact binary coalescence gravitational-wave signals. *Physical Review D*, 82(4):044025.

[12] Dasgupta, S. (2000). Experiments with random projection. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 143–151. Morgan Kaufmann Publishers Inc.

[13] Dasgupta, S. and Gupta, A. (1999). An elementary proof of the johnson-lindenstrauss lemma. *International Computer Science Institute, Technical Report*, pages 99–006.

[14] Fairhurst, S. (2009). Triangulation of gravitational wave sources with a network of detectors. *New Journal of Physics*, 11(12):123006.

[15] Fradkin, D. and Madigan, D. (2003). Experiments with random projections for machine learning. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 517–522. ACM.

[16] Goal, N., Bebis, G., and Nefian, A. (2005). Face recognition experiments with random projection. In *Proceedings SPIE Vol*, volume 5779, pages 426–437.

[17] Harry, G. M., Collaboration, L. S., et al. (2010). Advanced ligo: the next generation of gravitational wave detectors. *Classical and Quantum Gravity*, 27(8):084006.

[18] Indyk, P., Koudas, N., and Muthukrishnan, S. Identifying representative trends in massive time series data sets using sketches.

[19] Johnson, W. B. and Lindenstrauss, J. (1984). Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1.

[20] Kabán, A. (2015). Improved bounds on the dot product under random projection and random sign projection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 487–496. ACM.

[21] Luan, J., Hooper, S., Wen, L., and Chen, Y. (2012). Towards low-latency real-time detection of gravitational waves from compact binary coalescences in the era of advanced detectors. *Physical Review D*, 85(10):102002.

[22] Messick, C., Blackburn, K., Brady, P., Brockill, P., Cannon, K., Cariou, R., Caudill, S., Chamberlin, S. J., Creighton, J. D. E., Everett, R., Hanna, C., Keppel, D., Lang, R. N., Li, T. G. F., Meacher, D., Nielsen, A., Pankow, C., Privitera, S., Qi, H., Sachdev, S., Sadeghian, L., Singer, L., Thomas, E. G., Wade, L., Wade, M., Weinstein, A., and Wiesner, K. (2017). Analysis framework for the prompt discovery of compact binary mergers in gravitational-wave data. *Phys. Rev. D*, 95:042001.

[23] Metzger, B. D. and Berger, E. (2012). What is the most promising electromagnetic counterpart of a neutron star binary merger? *The Astrophysical Journal*, 746(1):48.

[24] Mitzenmacher, M. and Upfal, E. (2005). *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge university press.

[25] Owen, B. J. and Sathyaprakash, B. S. (1999). Matched filtering of gravitational waves from inspiraling compact binaries: Computational cost and template placement. *Phys. Rev. D*, 60:022002.

[26] Papadimitriou, C. H., Tamaki, H., Raghavan, P., and Vempala, S. (1998). Latent semantic indexing: A probabilistic analysis. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 159–168. ACM.

[27] Strang, G. (2006). *Linear Algebra and Its Applications*. Thomson, Brooks/Cole.

[28] Usman, S. A. et al. (2016). The PyCBC search for gravitational waves from compact binary coalescence. *Class. Quant. Grav.*, 33(21):215004.

[29] Vazirani, U., Papadimitriou, C., and Dasgupta, S. (2006). *Algorithms*. McGraw-Hill Education.

[30] Zhao, X. (2006). *High performance algorithms for multiple streaming time series*. PhD thesis, New York University.