Convolutional Denosing AutoEncoders - Feature Learning for LISA

Abstract.

Several uniquely challenging problems will need to be overcome in order to make data analysis for the Laser Interferometer Space Antenna (LISA) possible. First and foremost is the identification of individual sources from a data stream containing an unknown high number of overlapping signals. Source separation is still an unsolved problem in signal processing, and in recent years machine-learning techniques have increasingly been proposed as solutions. In this work we explore the ways in which a feature learning approach could help tackling LISA's challenges. We develop a machine-learning algorithm using Convolutional Denosing AutoEncoders (CDAEs) and apply it to the separation of overlapping sources within mixed spectrograms. Each CDAE is trained to separate one source while treating the other sources in the mixture as background noise. When tested on unseen data the model achieves excellent results, demonstrating the ability to learn ordered spatial information and successfully and reliably separating the sources present by type. Moving forward, the CDAE model shows the potential to be a solution to the LISA data analysis problems.

1. Introduction

The Laser Interferometer Space Antenna (LISA) [8] has recently been accepted as the third large-class mission in ESA's Science programme to address the theme of *The Gravitational Universe* [7]. Currently scheduled for launch in 2034 [35], LISA will be the first space-based gravitational waves (GWs) observatory. Its present design consists of three identical spacecrafts in an equilateral triangle configuration of side 2.5×10^9 m, orbiting the Sun in a stable Earth-trailing orbit.

The length of LISA's arms, as well as its position in space away from the contamination of the Earth gravity field, will allow sensitivity in the low frequency 0.1 - 100 mHz GW window, enabling us to detect a rich variety of sources with high mass and redshift. This will complement the existing and planned ground-based GW observatories, which have access to the high frequency 10 - 1000 Hz window.

LISA measurements will be carried out via laser interferometry between free flying test masses inside the drag-free spacecrafts. Laser beams will be exchanged between the spacecrafts, constantly monitoring their separation. Following the same principle underlining ground-based GW detectors, any incoming GWs will cause variations in the optical path-length between the detector's components, which will be recorded as the difference in frequency between the locally-generated and received laser signals. These variations are on the order of picometers and therefore require outstanding target sensitivity.

The LISA mission concept has existed since the late 1990s, and already prior to ESA's full endorsement, many of its theoretical and technical challenges have been subject of ongoing international efforts by the LISA Consortium, such as the successful test mission LISA Pathfinder. Following approval, as the mission now moves to its planning stages, several studies are being conducted to build the tools that will be used in the development of a data-processing pipeline.

In the work that follows some of the unique data analysis issues that will affect the mission are highlighted, and, after an introduction on the relevant background information, an approach based on *autoencoders* – a class of machine learning algorithms, is presented as potential solution.

2. LISA data analysis - two key problems

In recent years the successful hunt for GW signals in data from ground-based LIGO detectors validated decades of efforts to overcome significant mathematical, statistical and technical challenges. Although LISA data analysis shares considerable heritage with ground-based analysis, particularly in regard to constructing the template waveforms for binary black holes and binary stars, the actual search methods and noise handling techniques are very different, and include several tricky problems.

2.1. Laser noise & TDI

The largest contribution to the detector's noise is due to the frequency stability of the reference lasers, which is $\sim 10^4$ times higher than its target sensitivity. This is an intrinsic property of the lasers, and should therefore also affect ground-based GW detectors. The fundamental difference between the two cases is in the length of the interferometer arms: in equal-arm ground-based detectors the noise in the laser light is common to both arms, experiencing exactly the same delay, and thus cancels when it interferes at the beam-splitter before the photo detector. The situation for LISA is more complicated. The constellation's arm lengths will not be perfectly equal, making it not possible for the laser noise to be cancelled in the same way.

This poses a significant challenge which seems, at first, to make GWs observations with LISA impossible. The currently accepted approach to tackle this is Time Delay Interferometry, a post-processing technique for space-based GW detectors introduced by Armstrong, Estabrook and Tinto [1][27]. Its aim is to calculate TDI variables: time-shifted linear combinations of the six individual data streams that achieve cancellation of the common frequency noise.

The technique's mathematical formulation is based on the theory of rings and modules and computational commutative algebra. It shows that all possible interferometric combinations cancelling the laser noise are included in an algebraic module, called the module of syzygies. It can be shown that this module is generated by only four generators, which can be linearly combined to form all possible noise-free data combinations [29].

Time Delay Interferometry has been subject of careful study in the past decade (see [29] and references therein). Early approximate results (first generation TDI) which only included the effects of rotation but not arms flexing or other motions, have now been extended to a second generation, which incorporate all the motions of LISA.

Despite the complexity of the problem, TDI showed that LISA is indeed capable of delivering frequency-noise-free data. The technique is currently the ESA endorsed method to achieve LISA target sensitivity [29], but, when it comes to its practical implementation, it is not free from problems.

The primary unsolved issue with the TDI technique is the accurate (in the order of ns) determination of the time delays needed to cancel the laser noise, in the continuously evolving LISA configuration. Different solutions have been put forward, including real-time, onboard knowledge of the light-travel times via a triggering approach [28], and computational post-processing methods [26], but no definite answer has been found.

Although the laser noise problem wasn't the key focus of our investigation, the approach which was developed could prove to be useful in tackling this challenge also; more details will be discussed in section 8.

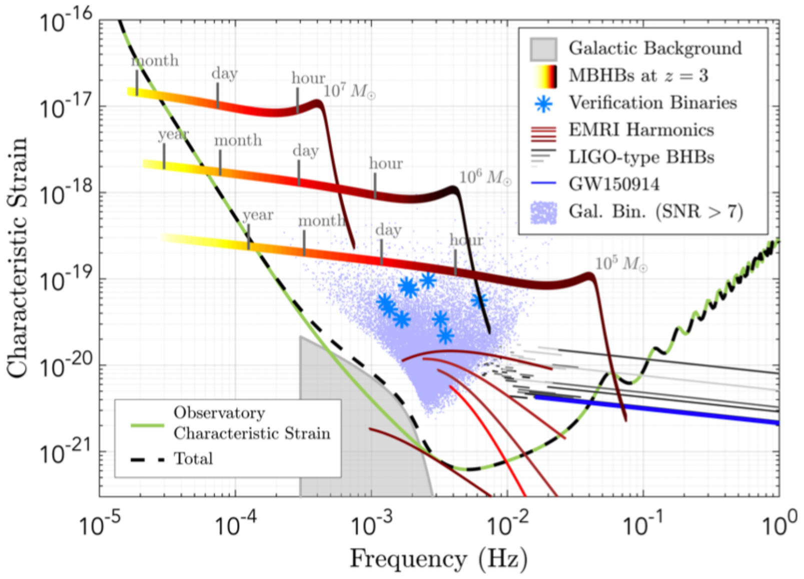


Figure 1: Examples of GW sources in the frequency range of LISA, compared with its sensitivity for a 3-arm configuration. Figure from [8].

2.2. Source separation

The detector's sensitivity to lower GW frequencies compared to LIGO, changes significantly the landscape of detectable sources, strongly influencing the science objectives of the mission, and introducing new challenges to the field. Figure 1 shows the expected sensitivity and some potential signals.

The most numerous sources in the low-frequency GW sky will be compact binaries sources of galactic origin, indicated by purple dots in the diagram. Galactic Binaries (GBs) systems include mainly white dwarfs binaries, but also various combinations neutron stars and stellar origin black holes. Higher frequency systems will be resolved individually as nearly monochromatic GW signals [22], with a characteristic frequency and amplitude modulation imparted by the detector orbital motion. A subset of known sources, indicated by the blue asterisks, will serve as verification signals. In addition, the high number of lower frequency unresolvable systems will form a stochastic 'confusion signal', represented in Figure 1 by the gray shaded area. The current best population synthesis models [30], paired with the reference detector sensitivity, then suggest that it should be possible to detect and resolve $\sim 25,000$ individual GBs over a mission lifetime of 4 years.

The upper shaded tracks in Figure 1 represent three examples of expected Massive Black Hole Binaries (MBHB) systems. These are events believed to occur when, following a galactic merger, the central massive black holes (MBHs) spiral inwards and eventually merge, producing large quantities of gravitational radiation. The origin and properties of these systems are closely connected with the clustering of cosmic structures [24], which make MBHB a target of high interest for LISA. Their inspiral, merger and ringdown is expected to cross the entire LISA frequency spectrum, in a transient GW signal lasting from months to days down to hours. Much is still not understood about how these sources form, but within conservative population models [20], LISA is expected to detect a few tens of MBHB mergers per year.

Another expected GW signal is from Extreme Mass Ratio Inspirals (EMRIs) - the long-lasting inspiral and plunge of compact objects such as white dwarfs, neutron stars and stellar origin

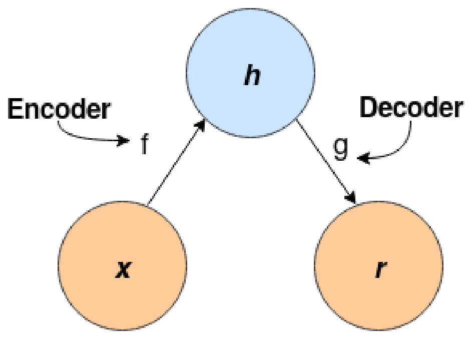


Figure 2: The general structure of an autoencoder, mapping an input **x** to an output (called reconstruction) **r** through an internal representation or code **h**. The autoencoder has two components: the encoder **f** (mapping **x** to **h**) and the decoder **g** (mapping **h** to **r**) [12].

black holes into MBHs in the centre of galaxies. The GW signal from these sources is expected to be split into frequency harmonics (red lower curves in Figure 1), due to their complicated orbital dynamics. EMRIs, if detected, have the potential to directly map the spacetime around MBHs. Within the limits of the large uncertainty in the astrophysics of EMRIs the rate of observed systems could range from tens up to thousands objects per year [11].

In addition to the described sources, LISA will be able to detect the most massive stellar origin black holes binaries in the nearby Universe (grey and blue tracks in Figure 1), earlier in their inspiral, before they cross into the LIGO band, at a rate of a few tens per year [25]. Finally LISA will be also searching for both a stochastic GW background and unmodelled exotic sources, which may probe new physics in the early Universe [8].

By considering the several observable sources and their expected detection rates, as well as the all-sky sensitivity of the detector, a central data analysis problem starts to come into focus: as soon as the LISA detector will be switched on, it will be able to observe several thousands sources simultaneously. The overlapping of the sources severely limits how well we can extract information about the individual contributing objects [5], and can be thought as a task analogous to determining what every guest of a busy party is saying, something referred to as the "Cocktail Party Problem". In order to then perform accurate parameters estimations it becomes fundamental to find a way to resolve the individual contributions to the data stream.

Source separation for LISA formed the primary objective of our investigation.

3. Autoencoders and feature learning

Although source separation formed the primary objective of our work, in looking for a solution, we found it useful to place both discussed problems in a wider signal processing context. Then, what do the solutions to the laser noise and source separation problem have in common?

They both can be thought as a pre-processing procedure involving *feature* or *representation learning*, i.e. the extraction of a (often reduced) set of features or representations from the initial dataset, which are informative and non-redundant (e.g. TDI noise-free linear combinations or overlapping signal components), and which enable or facilitate subsequent interpretation.

Following this approach, and drawing inspiration from other fields such as image and audio processing, we centred our investigation around a machine-learning technique called *autoecoders*.

Autoencoders [3][15][21] are a type of neural networks dedicated to feature learning; they learn efficient data codings by simply being trained to match their output to their input.

The general structure of an autoencoder is presented in Figure 2. An encoder function takes the input **x**, and produces a hidden representation, **h**:

$$\mathbf{h} = f(W\mathbf{x}),$$

where W is a matrix of trainable parameters or weights. A decoder function then produces a reconstruction, $\mathbf{r}$, of the input:

$$\mathbf{r} = g(V\mathbf{h}),$$

where V is also a matrix of weights.

It is straightforward to see that the easiest way to achieve optimal reconstruction would be for the network to trivially copy the input to the output by setting W and V to be identity matrices. This would, of course, not be useful, and autoencoders are instead designed to be unable to learn to copy perfectly. In their most common application this is achieved via *dimensionality reduction*: by setting the dimension of the hidden layer $\mathbf{h}$ to be smaller than the input $\mathbf{x}$, the network is forced to extract the most important features of the input data. The autoencoder is then called *undercomplete*.

As most types of neural networks, autoencoders undergo *training* before being used in the *testing* stage. During training the model essentially performs an optimisation task. For any input the network will calculate a loss function L, such, for example, the mean square error (MSE), $L \propto | \mathbf{r} - \mathbf{h} |^2$, which relates the input to the output and quantifies the quality of any particular set of weights W and V. The learning process can then be thought as a way to gradually find the optimal set of values for W and V which minimises the loss function L, and equivalently maximise the accuracy of the reconstruction. There are several different optimisation methods [2][19] used to reach the minimum of the loss function, a common one is gradient descent via backpropagation: a procedure which computes the gradient of L, and, backpropagating the information through the network, updates the values of W and V until a minimum is reached. Backpropagation is done over multiple iterations called epochs.

It can be shown that an autoencoder with linear encoder and decoder functions and a mean square error loss will learn to perform Principal Component Analysis (PCA) [12], a well-known statistical procedure for dimensionality reduction [17]. For this reason, when extended to the non-linear case, autoencoders are often thought as a powerful non-linear generalisation of PCA.

Once a network has been trained it is ready to be tested on unseen data, and incorporated into the wider task at hand.

4. Convolutional Denoising AutoEncoder - CDAE

A key issue for LISA, source separation is a long-standing usolved problem in signal processing, which affects really any dataset composed of a mixture of signals. It is an active area of research, especially in the study of audio signals, where machine-learning approaches have become increasingly popular in recent years (see e.g. [32], section 3).

After a review of the literature on this subject, we decided to center our investigation around a model called *Convolutional Denoising AutoEncoder* (CDAE) [34] [10], and took inspiration from [13], where CDAEs are applied to single channel audio source separation .

The first step in the design of any neural network, and indeed of a CDAE model, is to carefully consider the characteristics of the dataset one wishes to work with. In our case we decided to work with *spectrograms*, i.e. visual representation of the frequency evolution of the signal over time. This choice was influenced by the fact that LISA target sources are easily characterised in frequency vs time plots, and it meant that the model was designed with a 2D input in mind.

Figure 3 shows a schematic overview of the model. One can see the characteristic alternated layers structure of neural networks, as well as the elements discussed in the previous section: the encoder and decoder sections (in this case symmetrical), and a central unit of reduced

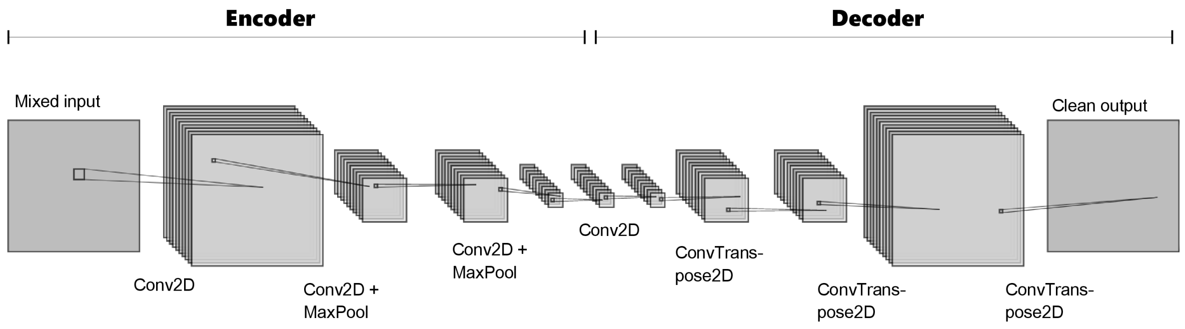


Figure 3: Schematic overview of the CDAE model. “Conv2D” denotes a convolutional layer with a Tanh activation function, “MaxPool” denotes a max-pooling layer, and “ConvTranspose2D” denotes a transpose convolutional layer with a Tanh activation function.

dimensions. The CDAE network builds on the fundamental elements of a vanilla autoencoder by adding two key ingredients.

The term *“convolutional”* refers to the particular type of operation performed on the input during encoding and decoding. In so called convolutional layers, indicated by “Conv2D” in Figure 3, a 2D filter (of dimensions ranging from 15×15 to 5×5 pixels) is progressively slid across the input image with a 1 pixel stride, and for every position the convolution between the filter and the corresponding part of the image is calculated. This operation is simultaneously done with multiple filters, each having their own set of trainable parameters. The strength of this method lies in the ability of convolutional layers to extract ordered spatial structures present in the input image, where multiple filters can be dedicated to different sets of features (e.g. edges or specific patterns), and the result is a set of “feature maps”, each containing some information about the original image. In our model each convolutional layer was followed by a Tanh activation function, and a max-pooling layer, indicated by “Max-Pool” in Figure 3, where only the largest element from a specified window was carried forward, so to obtain a down-sampling of the image.

In the decoder part of the network, transpose convolutions (“ConvTranspose2D” in Figure 3) with Tanh activation functions were used in a similar way to achieve up-sampling. The final output image was therefore of the same size of the input.

The term *“denoising”* is a reference to denoising anutoencoders (DAEs) [33] [12] a specific class of autoencoders used to learn noise robust low-dimensional features even when the inputs are perturbed with some noise. In our case this refers to the strategy used during training to achieve source separation: one CDAE was trained to extract one target source from the mixture by treating all the other sources as background noise to be suppressed. This means that as many CDAEs as the number of sources in the mixture are needed to resolve the input signal completely.

In the following section, a detailed description of how the CDAE model was implemented and applied to our specific investigation is presented, followed by an account of the model’s performance and results.

5. Method & Results

5.1. Training datasets

The first step in training the CDAEs was to create suitable training datasets. We decided to work with data in the time-frequency domain, and began our investigation by creating spectrograms of “toy” GW signals, i.e. analytical signals which resemble the behaviour of the typical LISA sources.

Four toy analytical signal were selected:

(i) a simple sine wave, to represent unmodulated GBs sources:

$$x(t) = A\sin(2\pi f t),$$

where A was amplitude, f frequency, and t time;

(ii) a sinusoidal exponential chirp,

$$x(t) = A\sin\left[b\pi f_0\left(\frac{k^{(t)}-1}{\ln(k)}\right)\right],$$

with

$$k = \left(\frac{f_1}{f_0}\right)^{\frac{1}{t_1}},$$

where A was amplitude, f_0 and f_1 the initial and final frequencies, and t_1 the chirping time;

(iii) and a truncated version of the previous exponential chirp, to represent MBHB inspirals and mergers;

(iv) a frequency modulated sine wave, to represent GBs modulated by the detector orbital motion:

$$x(t) = A\sin\left(2\pi f_c t + B\sin\left(2\pi f_m t + \phi\right)\right),$$

where A was amplitude, f_c and f_m the carrier and modulating frequencies, B the modulation index and ϕ the phase.

Each signal was sampled every 0.1 s over a period of 10 minutes, resulting in 6000 samples in the time domain. The spectrograms were then calculated using fast Fourier transforms over segments of length 200 s, with a 70% overlap. A blackman window function was used over each segment. This resulted in spectrogram images of dimensions 100×100 pixels. For each signal a total of 1000 unique spectrograms were created. The signals' parameters were either fixed or allowed to take a uniformly random value within a range, as summarised in Table 1.

Signal	**Parameter range**				
Sine waves	$A = 1.0$	$f = [1.0, 4.0]$ Hz	-	-	-
Chirps	$A = 1.0$	$t_1 = 350.0$ s	$f_0 = [0.5, 1.5]$ Hz	$f_1 = [1.5, 2.5]$ Hz	$b = 2$
Truncated chirps	$A = 1.0$	$t_1 = 250.0$ s	$f_0 = [0.5, 1.5]$ Hz	$f_1 = [1.5, 2.5]$ Hz	$b = 2$
Frequency modulated sine waves	$A = 1.0$	$f_m = 20.0$ Hz	$f_c = [1.0, 4.0]$ Hz	$B = [50.0, 200.0]$	$\phi = [0.0, 2\pi]$

Table 1: Summary of training signals parameters.

The analytical signals were then mixed in two ways: sines + chirps, and truncated chirps + frequency modulated sines. These and their corresponding clean/target components were used to train four separate CDAEs, one for the extraction of each source. Furthermore the two training runs for the sines + chirps mixture were repeated with the addition of white Gaussian noise.

In a subsequent part of the investigation we moved from purely analytical signals to more realistic LISA data, by creating mixtures containing simulated MBHB chirps.

The MBHB waveforms were created using the inspiral-merger-ringdown phenomenological model (IMRPhenomD) [16] [18], with the addition of the LISA antenna response in the frequency domain. The signals spanned a period of 15 days, sampled with a cadence of 20 s (total number of samples: 64800). The fast Fourier transform of these signals was calculated over segments of

MBHBs Parameter	
Ecliptic Latitude (rad)	0.5
Ecliptic Longitude (rad)	1.2
Chirp mass	[3000.0, 10000.0]
Mass ratio	1.5
Magnitude spin 1	0.91
Magnitude spin 2	0.9
Coalescence time (days)	[10, 20]
Redshift z	2.5
Distance (Mpc)	20941.5

Table 2: Summary of MBHB waveforms parameters.

Layer (type)	Number of filters	Filters size	Output Shape
Input	-	-	(1, 100, 100)
Conv2D + Tanh	10	(15, 15)	(10, 100, 100)
MaxPool	10	(2, 2)	(10, 50, 50)
Conv2D + Tanh	20	(5, 5)	(20, 50, 50)
MaxPool	20	(5, 5)	(20, 10, 10)
Conv2D + Tanh	30	(5, 5)	(30, 10, 10)
Conv2D + Tanh	40	(5, 5)	(40, 10, 10)
ConvTranspose2D + Tanh	30	(5, 5)	(30, 50, 50)
ConvTranspose2D + Tanh	20	(2, 2)	(20, 100, 100)
ConvTranspose2D + Tanh	10	(5, 5)	(10, 100, 100)
ConvTranspose2D + Tanh	1	(5, 5)	(1, 100, 100)
Output	-	-	(1, 100, 100)
Total number of trainable parameters = 89,900			

Table 3: Detailed structure of each CDAE.

length 1000 s, with an 80% overlap. As before a blackman window function was used over each segment. Lastly the spectrograms were downsampled to match the 100×100 pixels size. 1000 unique spectrograms were created, where all but two of the MBHBs parameters were kept fixed, as summarised in Table 2.

For the final training runs, a mixture of these MBHB signals and the frequency modulated sine waves was created, to be used to train two additional CDAEs.

Eight CDAEs networks were trained in total.

5.2. CDAEs properties and training technique

A key part of the model design was the choice of a set of *"hyperparameters"*, i.e. any parameter whose value is set before the learning process begins. These can be either structural, such as the number of layers, the size of the filters, the number of filters in each layer, the max-pooling and up-sampling ratios, etc.; or specific to the training setup, such as the input characteristics,

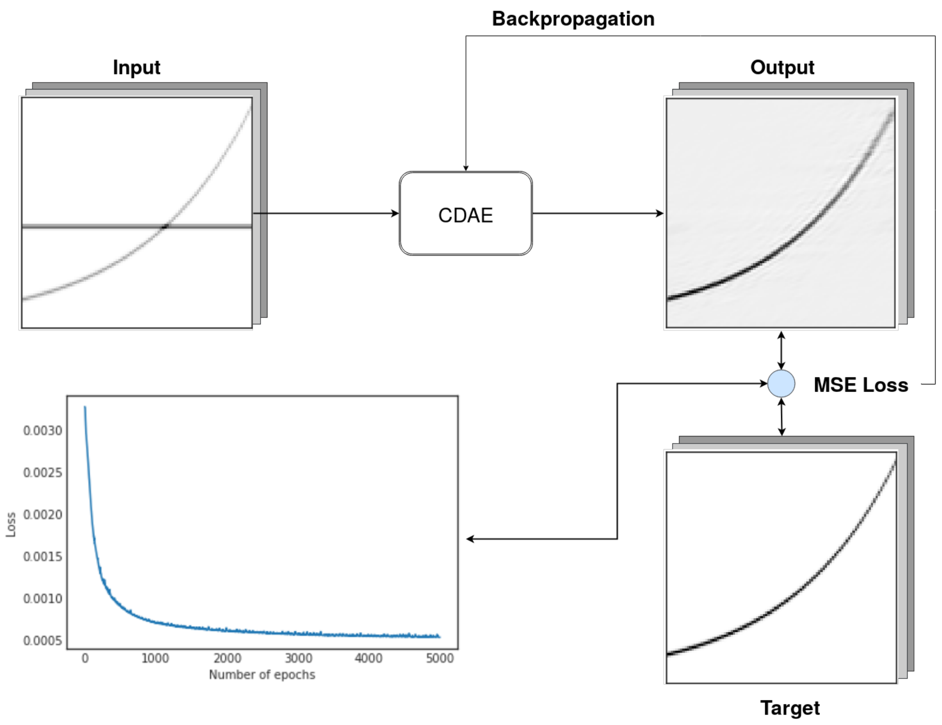


Figure 4: Schematic representation of the CDAE training process. Bottom left curve shows the value of the loss function over the training run.

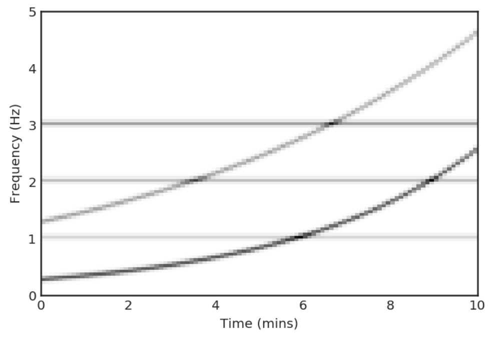
the training batches size, the number of epochs, etc.

Both sets of hyperparameters were determined empirically, partly based on the design adopted from [13], partly influenced by the properties of the signals' spectrograms. A detailed summary of the structure of each CDAE used is presented in Table 3 (please refer to Appendix 1 for the corresponding python script).

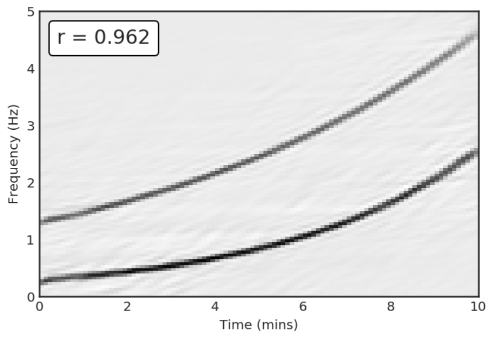
When it comes to training, it is of course desirable for the network to learn as many realisations of the input as possible, over wide parameter ranges. In our case the selected setup constituted the middle-way between a manageable training time and the ability of the trained networks to generalise and interpolate over the signals' parameter space.

Figure 4 shows a schematic representation of the training process. An instance of a mixture spectrogram is at the top left; during training the network tries to achieve the optimal reconstruction of the clean target (bottom right) spectrogram, by iteratively calculating the MSE loss between the the output and the target. The information is then backpropagated through the network causing the trainable weights to be updated. In this way the network learns to suppress all signals, but the target, as noise. The bottom left curve shows an example of how the value of the loss function decreases to a minima over the the training run.

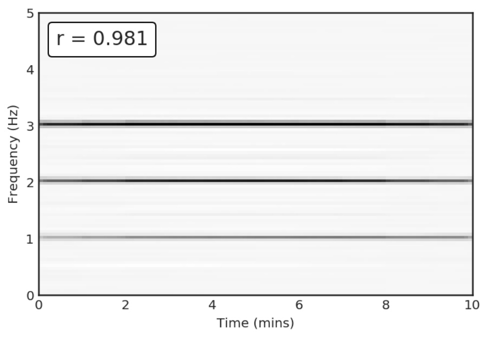
The parameters for all the networks were initiated randomly. All the networks were trained using backpropagation with an Adam optimiser [19], and a learning rate of 0.001, for a maximum of 5000 epochs. We implemented our proposed algorithm using Pytorch [36], a Python-based deep learning package, and the open-source Google platform Colaboratory [37]. For an overview of the training script used please refer to Appendix 2.



(a)



(b)



(c)

Figure 5: Performance of trained CDAEs on a chirps and sine waves mixture with no noise (a). Chirp-net output, (b). Sine-net output, (c). Top left r numbers in (b) and (c) are the correlation coefficients between the outputs and the corresponding targets.

5.3. Results

Following training the networks are ready to be applied to the separation of unseen data. For testing, data consisting of arbitrary mixtures of the four analytical signals and the MBHB signals, with various parameters, was generated. To quantitatively test performance, we calculated the Pearson correlation coefficient, r, between the network's outputs and the input's target components, i.e.

$$r = \frac{\sum_{i,j}^{n}(x_{i,j} - \bar{x})(y_{i,j} - \bar{y})}{\sqrt{\sum_{i,j}^{n}(x_{i,j} - \bar{x})^2}\sqrt{\sum_{i,j}^{n}(y_{i,j} - \bar{y})^2}},$$

where x is the input mixture spectrogram, $\bar{x}$ is its the mean value, y is the target spectrogram, and $\bar{y}$ is its mean value. The correlation coefficient ranges between 0 - no correlation, and 1 - maximum correlation, which means identical images.

Figure 5 shows an example of the performance of two trained networks (one for each type of source present) on a mixture of multiple sine waves and chirps with no noise. An important feature the model exhibits is the capability of the trained networks to individuate and separate the sources by type, even though they have never been exposed to mixtures containing multiple sine waves or multiple chirps. Furthermore the components of (a) in Figure 5 present variations of certain parameters which were kept fixed in the training data, such as varying amplitudes, and varying b for the chirps, which were successfully rendered in the separated outputs. The remarkable accuracy of the trained networks is evident in the r scores (given in the legends of subfigures (b) and (c)) achieved between the outputs and the targets.

In subsequent experiments CDAEs were trained with spectrograms corrupted by random Gaussian noise of fixed standard deviation, σ, and clean targets; and were tested with signals of various amplitudes. Figure 6 shows the networks performance at different levels of signal-to-noise ratio (SNR), defined here simply as:

$$\frac{A_i}{\sigma_i},$$

where A_i is the amplitude of the signals and σ_i the standard deviation of the noise in the i input spectrogram. In the first column we can see how, when given a purely noisy input, the two networks return an equally random output, while still searching for the spatial patterns they have learned to recognise. As soon as the signals start to emerge from the noise in the next two columns, they are successfully separated.

The correlation coefficients for these experiment were calculated between the outputs and noise-free targets, and are shown in Figure 7. As expected the networks performance improves with increasing SNR, reaching a stable r value of around 0.9 within a SNR of 2.

Figure 6 and 7 show the additional useful capability of the network to remove noise from the input as well as separating the sources, as a result of the adopted training strategy.

The next testing runs involved the CDAEs trained with mixtures of truncated chirps and frequency modulated sine waves. As previously mentioned, these two analytical signals were selected for their affinity to expected GW signals: MBHBs in the last stages of their inspiral are known to trace out a characteristic frequency chirp before merging, while monochromatic signals from GBs will be sine waves modulated by the detector's orbital motion (the modulation was heavily accentuated in our experiments).

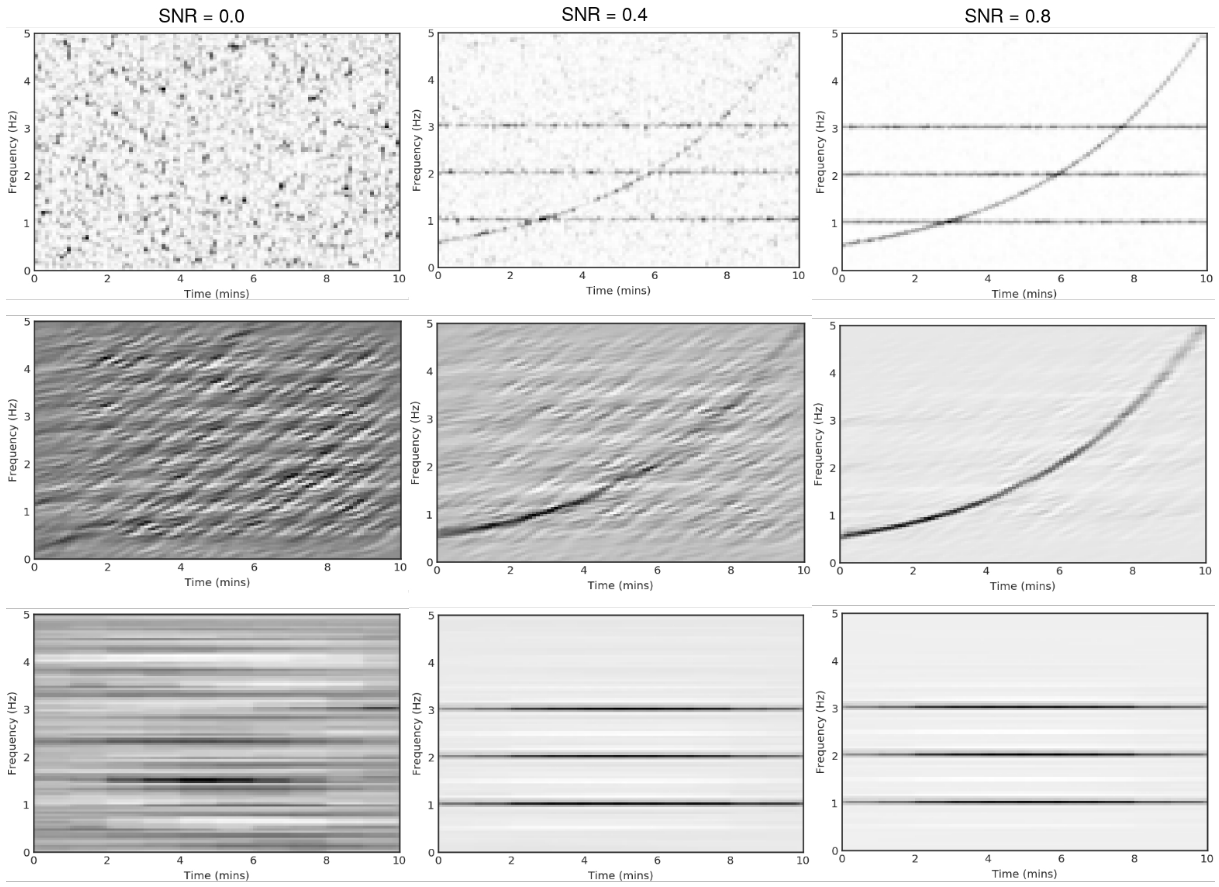
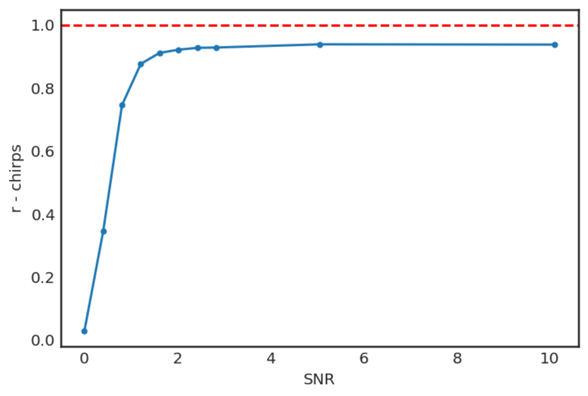
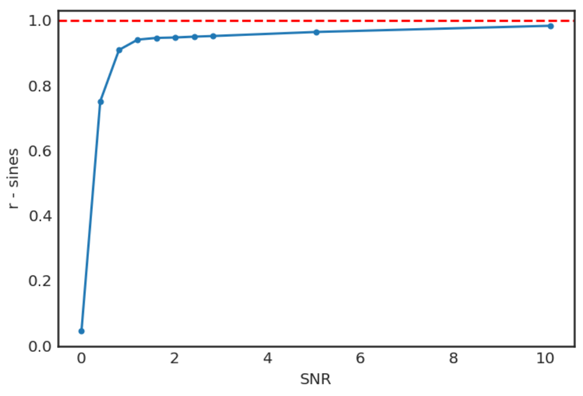


Figure 6: Performance of the model at different SNR levels. Top row: mixed input at SNR 0.0, 0.4 and 0.8. Middle row: chirp output. Bottom row: sines output.

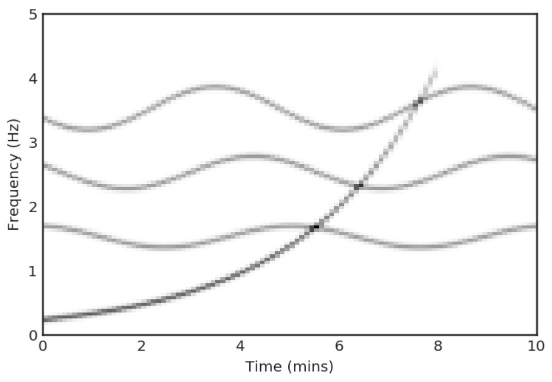


(a)

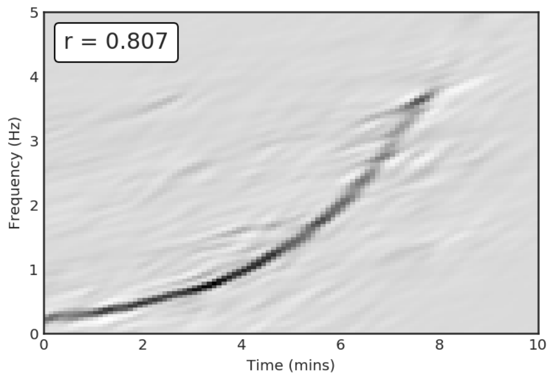


(b)

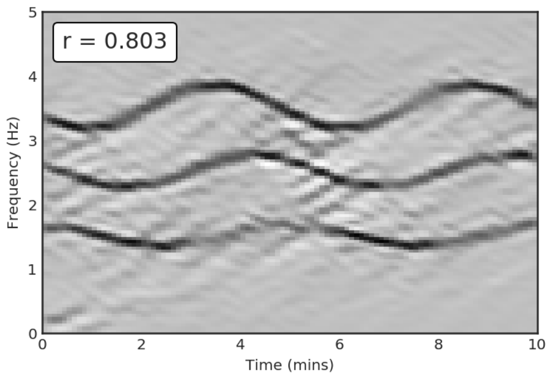
Figure 7: Correlation coefficient scores versus SNRs for chirps (a), and sine waves (b).



(a)



(b)

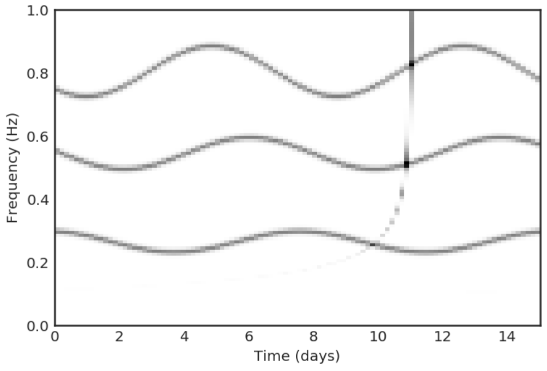


(c)

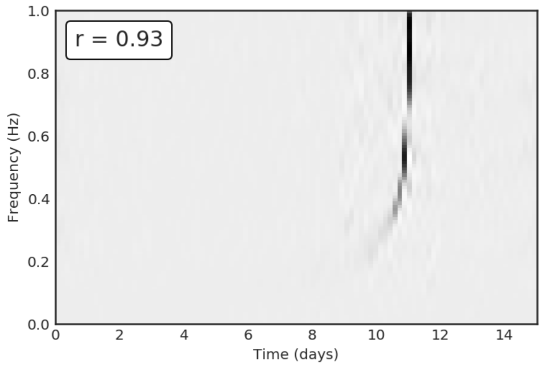
Figure 8: Performance of trained CDAEs on a truncated chirps (b) and frequency modulated sine waves (c) mixture with no noise. Top left r numbers in (b) and (c) are the correlation coefficients between the outputs and the corresponding targets.

Figure 8 shows the results for a mixture of multiple such signals. It is evident how more crowed and complicated signals are more challenging for the CDAEs to separate, with more noise and artefacts appearing in the outputs. This is also highlighted by the lower r scores. But nonetheless the mixture's components are reliably separated by the networks.

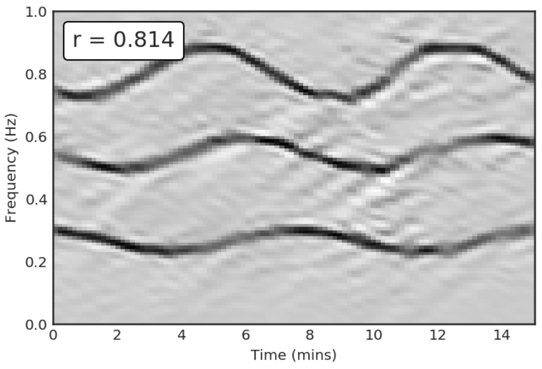
Finally, Figure 9 shows the model's performance with a mixture containing a simulated



(a)



(b)



(c)

Figure 9: Performance of trained CDAEs on a MBHB chirp (b) and frequency modulated sine waves (c) mixture with no noise. Top left r numbers in (b) and (c) are the correlation coefficients between the outputs and the corresponding targets.

MBHB merger track. Even though the realistic signal is arguably more complicated to generate than any of the analytic signals, in this example we see that the trained network is capable of achieving an overlap score for the separated MBHB signal higher than 0.9, which is a very promising result.

6. Discussion

The results presented in the previous section constitute a proof of principle of the model potential to achieve source separation in mixture spectrograms. In placing these results in a wider context, we wish to bridge the gap between the toy data the model was tested on, and realistic astrophysical data. In doing so, in this section, we highlight the model's strengths and weaknesses, focussing on the way in which the model's structure and usage could be improved in future experiments.

6.1. Network structure - hyperparameter tuning

As mentioned in section 5.1. the choice of the model's hyperparameters was a largely empirical process. The final configuration used is shown in Table 3, and it was loosely based on the one used in [13].

Although somewhat arbitrary, many of the implemented CDAE architecture choices were educated guesses driven by the physical characteristics of the GW spectrograms used. An example is the adoption of Tanh rather than rectified linear units (ReLUs), as activation function;

with a (-1, 1) range (rather than ReLU's (0, 1)), Tanh was better suited for inputs with large areas of near zero value, i.e. anywhere where the signals tracks were not present (this was especially true for the noise-free cases), and helped speed up convergence by avoiding substantial parts of the network to be passive, a problem encountered when using ReLUs.

Similarly the size of the filter used in convolutional layers was decided by thinking of the smallest spatial variation exhibited by the signals' spectrograms, to make sure the network was capable of learning the input's characteristic features. Ultimately the input spectrograms were structurally quite simple, which justified the choice of a relatively shallow 8 layers network, and a hidden encoded dimension of only 10×10 pixels.

What is important to stress is that, as of now, there is no known systematic way to make completely informed hyperparameters choices, i.e. how choose the network characteristics which will maximise the quality of the output. Usually the the parameters of a general model, such as the CDAE, are *tuned* to achieve optimal results to the problem at hand, meaning that the networks become highly specialised algorithms.

Hyperparameter tuning commonly consists of a grid or random search over the hyperparameter space to find the optimal setup. Examples of model's characteristics that can be found in this way are: the number of layers, the size of the filters, the number of filters in each layer, the max-pooling and up-sampling ratios, etc.

Thorough hyperparameter tuning can be especially helpful in meeting the requirements of datasets of higher complexity.

6.2. Input characteristics

Bringing this model forward as a potential source separation algorithm will require careful consideration of the characteristics of the data one wishes to work with. In many ways the network's performance is most strongly determined by the training dataset it learns from.

When it comes to LISA, this is tied to our ability to realistically simulate the possible GW sources. The ideal training dataset would include all possible realisations of all detectable sources, and would therefore span an intractably large parameter space. This is of course computationally prohibitive. The real potential of the proposed method lies in its flexibility and ability to interpolate between samples of the training dataset, and it was for example exhibited in Figure 5-8 where the networks successfully coped with features (e.g. varying signal amplitudes, multiple sources of the same type) they were never exposed to.

Ultimately this means that a subset of all the possible signals realisations might be enough to approximate the wider parameter space (akin to the concept of a template bank, used for matched filtering techniques). The task could be further simplified by training multiple CDAEs, each dedicated to learn different sections of the set.

In a practical scenario, training would constitute the vast majority of the computational effort, and is a procedure that would need to be computed only once. Once the networks have converged to an optimal solution, they would be ready to be used with real unseen data at a vastly reduced cost in comparison to the training stage.

The main limitation of the model arises when large parts of the input signals directly overlap or are spatially really close, e.g. sine waves with very similar frequencies. When this happens, the networks can no longer distinguish two separate signals, and will return a separated superposition of the two. This can be a major issue if the number and overlap of sources present exceeds the ability of the network to disentangle the mixture.

Nonetheless it is worth stressing that source separation is still a largely unsolved issue in signal processing, and it is of fundamental importance for the success of the LISA mission to find a solution. The results presented in this work highlight how a machine learning approach involving feature extraction has the potential and flexibility to tackle this problem in a constructive way.

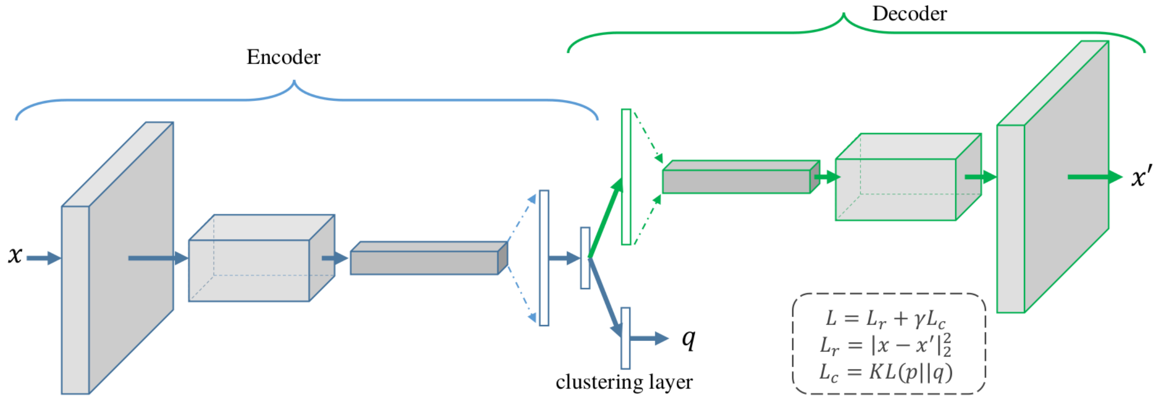


Figure 10: The structure of deep convolutional embedded clustering (DCEC), composed of a convolutional autoencoder with a clustering layer connected its central hidden layer. Reproduced from [14].

7. Future work

The potential of the CDAE idea for feature extraction is further considered in this section, where two of the key ways in which the model can be modified and adapted to more in-depth LISA data analysis tasks are described and discussed.

7.1. Global fit

A major goal for LISA data analysis will be not only the identification of individual sources from a data stream containing an unknown number of overlapping signals, but also the parameter estimation of all of these sources.

This would involve the simultaneous search over the complete parameter space spanned by the sources, of dimensions upwards of 50,000, a goal sometimes referred to as *"global fit"*. Although a few initial solutions have been proposed, such as Markov Chain Monte Carlo (MCMC) [4], and genetic algorithms [6], it is still unclear whether it will be possible to perform such an extensive parameter search.

Although, one of the instinctive ways one might think to apply the discussed CDAE model is a sort of "divide and conquer" approach, where the the data stream is first separated into its components via the trained networks and then a parameter search over the smaller spaces of each component is performed, it is important to realise that this would not be achievable or sensible to do in practice, where a global fit would always be the most accurate procedure.

The feature learning neural network approach does however still hold significant potential to tackle the global fit challenge. By modifying the network to include a probabilistic classification procedure, embedded in the CDAE architecture, the model could be trained to deliver point estimates of the separated sources parameters.

A analogous procedure was developed in [14], where the authors built a network called "deep convolutional embedded clustering (DCEC)", capable of simultaneously learning reduced representations of the data with a encoder-decoder approach, while at the same time performing a clustering task. The structure of the DCEC model is shown in Figure 10.

In a similar way a classification layer could be embedded in the CDAE architecture. Then, by paring each target spectrogram in the training dataset to a vector containing values of the source parameters (i.e. a set of physical parameters which correspond to the waveform like the ones shown in Table 2 for MBHBs), during training the networks would not only learn to separate the sources, but also how sets of parameters are related to specific waveforms in the frequency domain. During testing a network trained in this way would then return the separated sources paired with a set of point estimates for their parameters.

If this is achieved, it would constitute an invaluable resource to perform a global fit of the data. When performing a Bayesian parameter estimation, the returned point estimates could be used as starting points of a MCMC search over the sources multi-dimensional parameter space and, if sufficiently close, could make convergence to the true values computationally feasible.

7.2. Working in the time domain - seq2seq autoecoders

When working with spectrograms, an inevitable loss of information occurs: the spectrograms do not retain information about the phase of the signal they represent. This presents a great limitation to the analysis of GW signals, because phase encodes much of the information about the source parameters. For this reason time-domain data might be preferred over spectrograms.

To be able to work in the time-domain an autoencoder model must be able to extract structured temporal information from one-dimensional time vectors, in the same way in which convolutions are used to learn spatial features in images. To this aim the CDAE model could be modified by substituting convolutions with sequential operations, capable of learning the complex dynamics within the temporal ordering of input sequences as well as use an internal memory to remember or use information across long input sequences. These type of operations are used within the broad class of Recurrent Neural Networks (RNNs), such as Long Short-Term Memory models, and are widely used for text processing applications.

The new AE network would then be a “sequence to sequence” autoencoder (seq2seq AE), capable of taking a time series input and creating a reconstruction of it as output, while learning a useful reduced feature representation in its hidden layer.

After study of the connection between autoencoder and PCA, mentioned in section 4, and in turn the relationship between PCA and TDI explored in [23], we believe that sequence to sequence autoencoders could constitute a new way to obtain laser-noise-free linear combinations of the LISA data stream, making them a target for future investigations.

8. Conclusions

A comprehensive study of the potential of a feature learning approach to LISA data analysis was presented. After highlighting some of the unique challenges data analysis for this mission will present, we delved into an exploration of how autoencoders - a class of machine learning algorithms, could be used to tackle these issues. We worked with a model called Convolutional Denoising AutoEncoder, and demonstrated how it could be trained to perform the separation of sources within mixed spectrogram images. After creating a varied training dataset containing mixtures of four analytical sources and a simulated MBHB source, we trained one CDAE network for the extraction of each source. We achieved excellent results, with the model capable of successfully separating the sources present in unseen mixtures, with correlation coefficient scores with the target components as close as 0.9. Additionally the model demonstrates a good flexibility when dealing with unseen characteristics and noise, and, with future design studies, the potential to be applied to realistic LISA data. Finally we discussed some of the ways in which the model could be expanded and adapted to meet wider requirements of data analysis for LISA, such as the achievement of a “global fit” of all the sources which will be present in the detector’s output.

Acknowledgements

I would like to express my sincere gratitude to my project supervisors, Dr Natalia Korsakova and Prof Graham Woan, for their invaluable support and guidance during the duration of the project. Thank you!

References

[1] Armstrong, J. W., Estabrook, F. B. and Tinto, M., *Time-Delay Interferometry for Space-Based Gravitational Wave Searches*, Astrophys. J., vol. 527, p. 814-826 (1999)

[2] Bottou L., Curtis F. E., Nocedal J., *Optimization Methods for Large-Scale Machine Learning* eprint arXiv:1606.04838 (2016)

[3] Bourlard H., Kamp Y. *Auto-association by multilayer perceptrons and singular value decomposition*, Biological Cybernetics, vol. 59, p. 291-294 (1988)

[4] Cornish N. J., Crowder J., *LISA data analysis using MCMC methods*, Phys. Rev. D, vol. 72, p. 043005 (2005)

[5] Crowder J., Cornish N. J., *LISA source confusion* Phys. Rev. D, vol. 70, p. 082004 (2004)

[6] Crowder J., Cornish N. J., Lucas Reddinger J. *LISA data analysis using genetic algorithms*, Phys. Rev. D, vol. 73, p. 10110 (2006)

[7] Danzmann K. and the LISA Consortium, *The Gravitational Universe*, https://arxiv.org/abs/1305.5720 (2013)

[8] Danzmann et al, *Laser Interferometer Space Antenna*, Submitted to ESA on January 13th in response to the call for missions for the L3 slot in the Cosmic Vision Programme (2017)

[9] Dhurandhar S. V., Nayak K. R., Vinet J. -Y., *Algebraic approach to time-delay data analysis for LISA*, Phys. Rev. D, vol. 65, p. 102002 (2002)

[10] Du B., Xiong W., Wu J., Zhang L., Zhang L., Tao D., *Stacked convolutional denoising auto-encoders for feature representation*, IEEE Trans. on Cybernetics, vol. 47, no. 4, pp. 10171027 (2017)

[11] Gair J., Babak S., Sesana A., Amaro-Seoane P., Barausse E., Berry C. P. L., Berti E., Sopuerta C., *Prospects for observing extreme-mass-ratio inspirals with LISA*, J. Phys. Conf. Ser., vol. 840, no.1, p. 012021 (2017)

[12] Goodfellow I., Bengio Y., Courville A., *Deep Learning*, MIT Press, ch. 14 (2016)

[13] Grais E. M., Plumbley M. D., *Single channel source separation using convolutional denoising autoencoders* CoRR, abs/1703.08019 (2017)

[14] Guo X., Liu X., Zhu E., Yin J., *Deep Clustering with Convolutional Autoencoders*, Neural Information Processing. ICONIP 2017. Lecture Notes in Computer Science, vol. 10635. Springer, Cham (2017)

[15] Hinton G. E., Zemel R. S., *Autoencoders, minimum description length, and Helmholtz free energy*, In NIPS1993 (1994)

[16] Husa S., Khan S., Hannam M., Pürrer M., Ohme F., Forteza X. J., Bohé A., *Frequency-domain gravitational waves from nonprecessing black-hole binaries. I. New numerical waveforms and anatomy of the signal*, Phys. Rev. D, vol. 93, no. 4, p. 044006 (2016)

[17] Jolliffe I. T., *Principal Component Analysis*, Springer (2002)

[18] Khan S., Husa S., Hannam M., Ohme F., Pürrer M., Forteza X. J., Bohé A., *Frequency-domain gravitational waves from nonprecessing black-hole binaries. II. A phenomenological model for the advanced detector era*, Phys. Rev. D, vol. 93, no. 4, p. 044007 (2016)

[19] Kingma D. P., Ba J., *Adam: A Method for Stochastic Optimization* eprint arXiv:1412.6980 (2014)

[20] Klein A., Barausse E., Sesana A., Petiteau A., Berti E., Babak S., Gair J., Aoudia S., Hinder I., Ohme F., Wardell B., *Science with the space-based interferometer eLISA: Supermassive black hole binaries*, Phys. Rev. D, vol. 93, p. 024003 (2016)

[21] LeCun Y., *Modles connexionistes de lapprentissage*, Ph.D. thesis, Universit de Paris VI (1987)

[22] Nelemans G., Yungelson L. R., Portegies Zwart S. F., *The gravitational wave signal from the Galactic disk population of binaries containing two compact objects*, Astron. Astrophys., vol. 375, p. 890-898 (2001)

[23] Romano J. D., Woan G., *A principal component analysis for LISA – the TDI connection*, Phys. Rev. D, vol. 73, p. 102001 (2006)

[24] Sesana A., Haardt F., Madau P., Volonteri M., *Low-frequency gravitational radiation from coalescing massive black hole binaries in hierarchical cosmologies*, Astrophys. J., vol. 611, no. 2, p. 623 (2004)

[25] Sesana A., *Prospects for Multiband Gravitational-Wave Astronomy after GW150914* Physical Review Letters, vol. 116, p. 231102 (2016)

[26] Shaddock D. A., Ware B., Spero R. E., Vallisneri M., *Post-processed time-delay interferometry for LISA*, Phys.Rev. D, vol. 70, p. 081101 (2004)

[27] Tinto, M. and Armstrong, J. W., *Cancellation of laser noise in an unequal-arm interferometer detector of gravitational radiation*, Phys. Rev. D, vol. 59, p. 102003 (1999)

[28] Tinto M., Shaddock D. A., Sylvestre J., Armstrong J. W., *Implementation of time-delay interferometry for LISA*, Phys. Rev. D, vol. 67, p. 122003 (2003)

[29] Tinto M., Dhurandhar S. V., *Time-Delay Interferometry*, Living Reviews in Relativity, 17, no. 1 (2014)

[30] Toonen S., Nelemans G., Portegies Zwart S., *Supernova Type Ia progenitors from merging double white dwarfs: Using a new population synthesis model*, Astron. Astrophys., vol. 546, p. A70 (2012)

[31] Vallisneri M., *Geometric time delay interferometry*, Phys. Rev. D, vol. 72, p. 042003 (2005)

[32] Various authors, *Audio Source Separation*, Edited by Shoji Makino, Springer (2018)

[33] Vincent P., Larochelle H., Bengio Y., Manzagol P. A., *Extracting and composing robust features with denoising*

autoencoders, in Proc. ICML, p. 10961103 (2008)
[34] Zhao M., Wang D., Zhang Z., Zhang X., *Music removal by convolutional denoising autoencoder in speech recognition*, In proc. APSIPA (2016)
[35] http://sci.esa.int/lisa
[36] https://pytorch.org
[37] https://colab.research.google.com

APPENDIX 1 - CDAE python script

```python
import numpy as np
import torch

class CDAE(nn.Module):

    def __init__(self):
        super(CDAE, self).__init__()

        self.encoder = nn.Sequential(
            nn.Conv2d(1, 10, kernel_size=15, stride=1, padding=7, bias=False),
            nn.Tanh(),
            nn.MaxPool2d(kernel_size=(2,2), stride=(2,2)),
            nn.Conv2d(10, 20, kernel_size=5, stride=1, padding=2, bias=False),
            nn.Tanh(),
            nn.MaxPool2d(kernel_size=(5,5), stride=(5,5)),
            nn.Conv2d(20, 30, kernel_size=5, stride=1, padding=2, bias=False),
            nn.Tanh()
            nn.Conv2d(30, 40, kernel_size=5, stride=1, padding=2, bias=False),
            nn.Tanh()
        )

        self.decoder = nn.Sequential(
            nn.ConvTranspose2d(40, 30, kernel_size=5, stride=5, bias=False),
            nn.Tanh(),
            nn.ConvTranspose2d(30, 20, kernel_size=2, stride=2, bias=False),
            nn.Tanh(),
            nn.ConvTranspose2d(20, 10, kernel_size=5, stride=1, padding=2, bias=
    False),
            nn.Tanh(),
            nn.ConvTranspose2d(10, 1, kernel_size=5, stride=1, padding=2, bias=
    False),
            nn.Tanh()
        )

    def forward(self, x):

        x = self.encoder(x)
        x = self.decoder(x)

        return x
```

APPENDIX 2 - Training python script

```python
import numpy as np
from matplotlib import pyplot as plt
plt.style.use('seaborn-white')

import torch
from torch import nn, optim
import torch.nn.functional as F
from torch.autograd import Variable

from CDAE_model import *

import copy
import random
from datetime import datetime

class CDAE_trainer:
    def __init__(self,
                 model,
                 path,
                 filename,
                 criterion = nn.MSELoss(),
                 optimizer=optim.Adam,
                 lr=0.001,
                 weight_decay=1e-5):
        self.model = model
        self.lr = lr
        self.weight_decay = weight_decay
        self.criterion = criterion
        self.optimizer_type = optimizer
        self.optimizer = self.optimizer_type(params=self.model.parameters(), lr=
    self.lr, weight_decay=self.weight_decay)
        self.path = path
        self.filename=filename

    def train(self,
              mixed_input, # mixed sources spectrograms
              target,      # clean target source spectrograms
              n,           # datasets size (tot number of spectrograms)
              device,      # GPU vs CPU
              batch_size,
              epochs,
              H,           # height spectrograms
              W):          # width spectrograms

        self.model.train()

        inputs = Variable(torch.from_numpy(mixed_input)).to(device)
        inputs = inputs.reshape(n,1,H,W)
        inputs = inputs.type(torch.cuda.FloatTensor)

        target = Variable(torch.from_numpy(target)).to(device)
        target = target.reshape(n,1,H,W)
        target = target.type(torch.cuda.FloatTensor)

        loss_save = []
        current_epoch = 0

        for current_epoch in range(epochs):

            idx  = random.sample(list(np.arange(n)),batch_size)

            outputs = self.model(inputs[idx,:,:,:]) # apply network

            loss = self.criterion(outputs, target[idx,:,:,:]) # calculate loss
            self.optimizer.zero_grad() #reset optimiser
            loss.backward() # backpropagation step
            self.optimizer.step() # optimiser step

            loss_save.append(loss.item())

        # monitor progress
            if current_epoch%1000==0:
                print('epoch {}, loss {}'.format(current_epoch, loss.item()))

                plt.imshow(target[0,:,:].reshape(H,W).cpu().numpy(),aspect='auto', origin='lower', extent=[0,30,0,1.0])
                plt.xlabel('Time (days)')
                plt.ylabel('Frequency (Hz)')
                plt.title('Target')
                plt.show()

                plt.imshow(outputs[0,:,:,:].reshape(H,W).detach().cpu().numpy(), aspect='auto', origin='lower', extent=[0,30,0,1.0])
                plt.xlabel('Time (days)')
                plt.ylabel('Frequency (Hz)')
                plt.title('Separated output')
                plt.show()

        plt.plot(loss_save)
        plt.xlabel('Number of epochs')
        plt.ylabel('Loss')
        plt.show()

        print('Saving trained model...')

        now = datetime.now()
        time_string = now.strftime("%d-%m-%Y_%H:%M:%S")
        torch.save(self.model, self.path + '/' + self.filename + '_' + time_string + '.pt')

        print('Model '+ self.filename + '_' + time_string + '.pt' + ' saved.')
```