UNSUPERVISED LEARNING OF SPARSE AND SHIFT-INVARIANT DECOMPOSITIONS OF POLYPHONIC MUSIC

T. Blumensath, M. Davies

Queen Mary, University of London
Department of Electronic Engineering
Mile End Road, London E1 4NS, UK

ABSTRACT

Many time-series in engineering arise from a sparse mixture of individual components. Sparse coding can be used to decompose such signals into a set of functions. Most sparse coding algorithms divide the signal into blocks. The functions learned from these blocks are, however, not independent of the temporal alignment of the blocks. We present a fast algorithm for sparse coding that does not depend on the block location. To reduce the dimensionality of the problem, a subspace selection step is used during signal decomposition. Due to this reduction an Iterative Reweighted Least Squares method can be used for the constrained optimisation. We demonstrate the algorithm's abilities by learning functions from a polyphonic piano recording. The found functions represent individual notes and a sparse signal decomposition leads to a transcription of the piano signal.

1. INTRODUCTION

Many time-series can be regarded as arising from a sparse linear mixture of individual processes. It is often desirable to express such time-series using a sparse mixture of functions which then reveal the structure of the processes underlying the signal. Olshausen [1] showed that a sparse decomposition of images leads to functions similar to the receptive fields found in the primate visual cortex V1 (i.e. being *localized*, *oriented* and *bandpass*). Lewicki [2] gave a statistical analysis of the problem and developed a slightly improved algorithm. Abdallah [3] used standard ICA techniques to learn functions from audio signals. The functions observed were Gabor like atoms with shorter time support for speech signals and longer time support for musical signals. The algorithms of Lewicki and Olshausen as well as the ICA method used by Abdallah work on fixed blocks and are not shift-invariant and so the learned functions are dependent upon the block positions. In image and time series analysis this fixed block position leads to the learning of filtered features and their translations.

The algorithm developed here uses a subspace selection step to reduce the dimensionality of the problem. This leads to a significant increase in speed of Lewicki's and Olshausen's algorithms which then enables us to extend these algorithms to shift-invariant learning.

2. THEORY

We assume the following linear mixture model:

$$\mathbf{x}_n = \sum_{i\in I, l\in L} \mathbf{a}_{il} s_{iln} + \epsilon = \mathbf{A}\mathbf{s}_n + \epsilon \tag{1}$$

Here $\mathbf{x}_n$ is the n^{th} block of the observed time-series. (We will drop the subscript n from now on.) $\mathbf{A}$ is a matrix with the individual functions $\mathbf{a}_{il}$ as its columns, where $i \in I$ is the label for the function of the set I of all functions and $l \in L$ is the shift of the function with L being the set of all possible shifts. s_{il} are the decomposition coefficients associated with the i^{th} function at the l^{th} shift. ϵ is a column vector of i.i.d zero mean noise.

To include the shifts we have to assume that the matrix $\mathbf{A}$ not only contains the individual functions but also all possible shifts. $\mathbf{A}$ therefore contains truncated functions for some shifts. The coefficients $\mathbf{s}$ then not only select a function but also give its location. Matrix $\mathbf{A}$ is now a matrix specifying $|I|$ linear time-shift-invariant filters. (Here $|\cdot|$ denotes cardinality.) $\{s_{il}\}_{l\in L}$ can then be seen as the input sequences to these filters where $\mathbf{x}$ is the noisy observation of the sum of their outputs.

To estimate the matrix $\mathbf{A}$ we can calculate the maximum likelihood estimate of:

$$P(\mathbf{x}|\mathbf{A}) = \int P(\mathbf{x}|\mathbf{A},\mathbf{s})P(\mathbf{s})\, d\mathbf{s} \tag{2}$$

In the shift-invariant model the maximum likelihood estimate has to be calculated with respect to the individual functions $\mathbf{a}_i$ by taking the structure of $\mathbf{A}$ into account.

To find the signal decomposition coefficients $\mathbf{s}$ we can calculate the MAP estimate of:

$$P(\hat{\mathbf{s}}|\hat{\mathbf{A}},\mathbf{x}) \tag{3}$$

Here $\hat{\mathbf{A}}$ and $\hat{\mathbf{s}}$ are the current approximations of $\mathbf{A}$ and $\mathbf{s}$ respectively. In the following we will drop the hat notation for readability.

We use Bayes theorem to write (3) as:

$$P(\mathbf{s}|\mathbf{A},\mathbf{x}) \propto P(\mathbf{x}|\mathbf{A},\mathbf{s})P(\mathbf{s}) \tag{4}$$

Taking logarithms and inserting the Gaussian distribution into the right hand side of (4) we can find the MAP estimate of $\mathbf{s}$.

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \lambda_1 \sum_m |\epsilon_m|^2 - \log P(\mathbf{s}) \tag{5}$$

ϵ_m is the m_{th} value of ϵ. Here we use $\epsilon = \mathbf{x} - \mathbf{A}\mathbf{s}$ and assume that the $P(\epsilon) \sim \mathcal{N}(0, \sigma\mathbf{I})$ where $\mathbf{I}$ is the identity matrix. Additive constants have also been dropped and the multiplicative constants in the first term have been collected into λ_1.

This work was partly supported by EPSRC Grant GR/R54620.

It is important to note that we make an independence assumption on the individual coefficients such that $P(\mathbf{s}) = \prod P(s_{il})$. To find a sparse signal decomposition a sparse prior distribution $P(s_{il})$ has to be used. Different priors have been proposed in the literature. Probably the most commonly used is the Laplacian distribution. This would lead to a second term in expression (5) of $\lambda_2 \sum_{il} |s_{il}|$ which is a L_1 norm constraint. (We again collect constants this time into λ_2.) Using a hierarchical model for the prior with $P(\mathbf{s}|\tau) \sim \mathcal{N}(0, \tau)$ and a Jeffrey's hyper prior for the variance τ leads to an algorithm similar to the IRLS method proposed here when using a $\sum \log |s_{il}|$ constraint. (See [4] for a derivation.)

For a general constraint $f(\mathbf{s})$ we have:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \sum_m |\epsilon_m|^2 + \lambda f(\mathbf{s}) \tag{6}$$

where we have collected the multiplicative constants into λ. λ is now a function of the variances of the prior and the likelihood. It regulates the trade-off between sparseness and noise variance.

As we have to include all possible shifts of each function in the matrix $\mathbf{A}$ the size of the problem is increased by at least a factor of $2 * N - 1$ so that $\mathbf{A}$ includes at least one complete function. (Here N is the length of the functions.) To be able to use standard optimisation methods a reduction in the size of the problem is required. We propose an efficient subspace selection method below.

In order to estimate $\mathbf{A}$ it can be shown (see [3]) that the likelihood in equation (2) can be written as the expectation of the gradient of $P(\mathbf{x}|\mathbf{A}, \mathbf{s})$ with respect to (3). In the shift-invariant model this gradient is:

$$\frac{\partial}{\partial \mathbf{a}_{il}} P(\mathbf{x}|\mathbf{A}, \mathbf{s})\mathbf{s} = \Sigma(\mathbf{x} - \mathbf{A}\mathbf{s}) \star \{s_{il}^T\}_{l \in L} \tag{7}$$

Here $\mathbf{s}$ refers to the current MAP approximation for a given observation $\mathbf{x}$ and the current approximation of $\mathbf{A}$ and $\star$ is the convolution operator.

The updating of $\mathbf{A}$ leads to a weighted averaging of the shifted signal blocks $\mathbf{x}$. The weights are determined by the value of s_{il} and are also dependent on the particular implementation of the algorithm. (i.e. in the online learning approach used here the previous weights decay with each new update.) To be able to learn the exact function as present in $\mathbf{x}$ it is important that the weights are only non-zero at the exact shift of the function in the signal. If this is not the case, filtering will occur, which is an averaging of weighted shifts. This problem is especially significant when the shift is not taken into account as in the work reported by Olshausen [1], Lewicki [2] and Abdallah [3] . This might be one of the reasons the functions presented in their work were all bandpass.

3. ALGORITHM

The proposed algorithm can be roughly broken into three parts.

1. Selecting a subspace of functions and shifts for each observation vector $\mathbf{x}_n$.
2. Finding the minimum of (6) in this subspace.
3. Updating the functions in matrix $\mathbf{A}$.

These steps are further explained below.

3.1. Subspace selection

Instead of finding $P(\mathbf{s}|\mathbf{A}, \mathbf{x})$ in a high dimensional space we reduce the problem size by considering only a low dimensional subspace. The selection of this subspace can be done by selecting a space spanned by the vectors $\mathbf{a}_{il}$ for which $P(s_{il}|\mathbf{A}, \mathbf{x})$ is high. Note that we here only model the signal $\mathbf{x}$ by one function $\mathbf{a}_{il}$. We also impose a constraint on the possible shifts of a function so that a function cannot overlap with a shifted version of itself by more than a fixed amount. This has to be done as functions shifted only slightly are similar to themselves (i.e. have a high autocorrelation for low lag values). The probability $P(s_{il}|\mathbf{A}, \mathbf{x})$ is assumed to be i.i.d. Gaussian. This assumption does not take account of time dependencies in the residual if just one function is used but is otherwise justified by the central limit theorem as explained below.

The subspace selection is then implemented by calculating the correlation of the signal with all basis functions at all shifts which can be achieved using a fast convolution method. We then iterate through the following two steps. First we select the basis function and associated shift with the highest correlation. We then set this correlation value, as well as correlation values for which shifts are prohibited by our constraint on function overlap, to zero. In this way we select a subspace of fixed size in which the optimisation in (6) becomes feasible.

This procedure can be justified statistically as follows. We can factor the posterior for $\mathbf{s}$, using the index t instead of the indices i and l to denote the function and the associated shift.

$$P(\mathbf{s}|\mathbf{A}, \mathbf{x}) = P(s_{t_1}|\mathbf{A}, \mathbf{x})P(s_{t_2}|s_{t_1}, \mathbf{A}, \mathbf{x}) \dots \tag{8}$$

We only work with the MAP estimates for each distribution and further truncate the right hand side to a few terms. We assume a uniform prior for $P(s_t)$ and also presume that $P(\mathbf{x}|\mathbf{A}, s_t)$ is Gaussian. This assumption can be justified by noting that $\mathbf{x}$ is dominated by the mixture of a number of functions when the noise ϵ has small variance. If we assume that all functions have a similar distribution then near Gaussianity follows from the central limit theorem. We therefore write:

$$P(s_t|\mathbf{A}, \mathbf{x}) \sim \mathcal{N}(\mathbf{a}_t s_t, \mathbf{\Sigma}) \tag{9}$$

We use this expression to calculate:

$$t_1 = \arg\max_t P(s_t|\mathbf{A}, \mathbf{x}) \tag{10}$$

which is the index t, which maximises $\mathbf{x}^T \mathbf{a}_t$. Here we have used $\mathbf{\Sigma} = \delta \mathbf{I}$.

We now have to find an expression for $P(s_{t_m}|s_{t_{1:m-1}}, \mathbf{A}, \mathbf{x})$ where we use the subscript notation $1 : m$ to denote all variables with subscripts between 1 and m. Using Bayes' rule we find:

$$P(s_{t_m}|\mathbf{A}, \mathbf{x}, s_{t_{1:m-1}}) \propto P(\mathbf{x}|\mathbf{A}, s_{t_{1:m}})P(s_{t_m}|s_{t_{1:m-1}}) \tag{11}$$

We can incorporate the constraint on function shifts by using the prior $P(s_{t_m}|s_{t_{1:m-1}}) = P(s_{t_m})U_{t_{1:m-1}}$, where $P(s_{t_m})$ is again a uniform distribution and $U_{t_{1:m-1}}$ is a function which is zero for shifts around $l_{1:m-1}$ but otherwise has a value of ν normalising the distribution. We make the (not necessarily correct) assumption that $P(\mathbf{x}|\mathbf{A}, s_{t_{1:m}}) = P(\mathbf{x}|\mathbf{A}, s_{t_m})$. This gives the major reduction in computational time of the subspace selection method. Note that for a Matching Pursuit algorithm the distribution $P(\mathbf{x}|\mathbf{A}, s_{t_{1:m}})$ is Gaussian with a mean of $\sum a_{t_{1:m}} s_{t_{1:m}}$ whilst we use a mean of $a_{t_m} s_{t_m}$.

Selecting the index t_m can therefore be done in a similar fashion as above. We again need the correlation of all functions at those shifts which do not violate the constraint. These correlations

have been calculated before and do not have to be re-evaluated. In a Matching Pursuit algorithm we would have to recalculate the correlation in each step as it is determined with the residual.

3.2. IRLS

In this low dimensional subspace the optimisation problem in (6) now becomes solvable. We use an Iterative Reweighted Least Squares (IRLS) approach. (For a constraint of the form $\sum \log |s_{il}|$ this leads to the same algorithm proposed by Figueiredo in [4]. It is also similar to the FOCUSS algorithm proposed by Rao et. al. [5] when extended to the noisy mixture case [6].)

We can write (6) as:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \|x - \mathbf{A}\mathbf{s}\|_2^2 + \lambda |\mathbf{s}|^T \mathbf{U}^{-T} \mathbf{U}^{-1} |\mathbf{s}| \qquad (12)$$

Where $\mathbf{U}^{-1}$ is a weighting matrix which converts the right part of the expression (12) into a standard least squares minimisation problem. We choose $\mathbf{U}^{-1}$ so that equation (12) has the same fixed points as equation (6). $\mathbf{U}^{-1}$ is then a function of the unknown fixed points which can be approximated in each step.

Solving expression (12) for $\mathbf{s}$ we get the iterative process:

$$\hat{\mathbf{s}}^{[k+1]} = \mathbf{U}^{[k]} (\mathbf{U}^{[k]} \mathbf{A}\mathbf{A}^T \mathbf{U}^{[k]} + \tilde{\lambda}^{[k]} \mathbf{I})^{-1} \mathbf{U}^{[k]} \mathbf{A}\mathbf{x} \qquad (13)$$

And $\mathbf{U}^{[k]}$ is calculated as:

$$\mathbf{U}^{[k]} = diag \left| \hat{s}_i^{[k]} \right|^{\frac{2-p}{2}} \qquad (14)$$

Where p is the norm of the required solution. For $p = 0$ this formulation leads to the $\sum \log |s_{il}|$ constraint and not to an L_0 norm (see [7]).

Following Figueiredo [8] λ is set to the estimated noise variance when using the $\sum \log |s_{il}|$ constraint which leads to the following expression:

$$\tilde{\lambda}^{[k]} = \frac{\|\mathbf{x}^{[k]} - \mathbf{A}\mathbf{s}^{[k]}\|_2^2}{m} \qquad (15)$$

where m is the dimension of the vector $\mathbf{x}$.

3.3. Learning the parameters

Learning the model parameters, i.e. the matrix $\mathbf{A}$, can be achieved using an approach similar to the gradient algorithm proposed by Olshausen [1]. For the shift-invariant model the update for a function $\mathbf{a}_i$ is:

$$\Delta \mathbf{a}_i = \mu (\mathbf{x} - \mathbf{A}\mathbf{s}) \star \{s_{il}^T\}_{l \in L} \qquad (16)$$

Assuming that $\mathbf{\Sigma} = \sigma \mathbf{I}$ we have included the variance into the learning rate. It would be possible to estimate the expected variance, but in the experiments reported below the parameter μ was kept fixed.

By using the MAP estimate of $\mathbf{s}$ some information is however lost. This is especially critical for those function shifts for which only part of the function contributes to the current observation $\mathbf{x}$. For example the one sample at the beginning or end of the observed block could arise from any of the functions by selecting an appropriate coefficient. This would be reflected in the full distribution of $\mathbf{s}$ by an increase in variance. We therefore only include those shifts for which the entire function contributes to the observation. The coefficients found for truncated functions are less reliable and are therefore not considered for the updating of the functions. (Note that this does not bias the estimate as the data blocks are selected at random locations.)

We therefore write the update step as:

$$\Delta \mathbf{a}_i = \mu (\mathbf{x} - \mathbf{A}\mathbf{s}) \star \{s_{il}^T\}_{l \in \overline{L}} \qquad (17)$$

where $\overline{L}$ is the set of all shifts for which functions are not truncated.

The functions are normalised after each update to deal with the scale ambiguity in the model.

4. RESULTS

To test the algorithm we used a recording of L. von Beethoven's Sonata for Piano No. 12, in A flat, Scherzo (Allegro molto). The original stereo recording was summed to mono and resampled at 8000 Hz. The number of possible functions was set to 50, a function length of 1024 samples was chosen, μ was set to 0.1 and the maximally allowed amount of overlap of one function with a shifted version of itself was set to 50%. The IRLS algorithm used a fixed number of 10 iterations.

After around 100,000 iterations 10 of the functions did not show any harmonic structure and were of a noisy nature. The other 40 functions had a clear harmonic structure. Of those 40 functions 35 had different fundamental frequencies whilst the other 5 functions had a fundamental frequency equal to at least one other function. Analysis of the functions further showed that the fundamental frequencies corresponded to the notes of the western equally tempered 12 tone scale spanning a range from C#2 to A5 with some notes missing. Most functions were harmonic in that their spectrum had a harmonic series of peaks. The amplitude of these peaks varied with one harmonic often having a much higher amplitude than the others. It was notable that the learned functions did not contain much high frequency energy and there were also no harmonic series present with very low fundamental frequencies even though such notes were present in the analysed signal. However, some of the 10 noise like functions were found to have a high concentration of low frequency energy. A typical selection of the learned functions is shown in figure (1) with their magnitude spectrum shown in figure (2).

Calculating the estimate of $\mathbf{s}$ using the learned functions makes a sparse and shift-invariant decomposition of the signal possible. By assigning notes to the individual functions and by assuming that each function represents a piano note played for the length of that function, a transcription of the music was possible. The coefficients could be converted into MIDI notation using the coefficient amplitude as the velocity, the coefficient time location as the note onset and the length of the basis function as the duration of each note. A representation of individual MIDI notes over time is given in figure (3). The melody line is correctly identified and so are some of the chords. Only a few notes seem to be detected incorrectly and a number of notes were not found at all. A similar experiment using a recording of a MIDI controlled acoustic piano playing L. von Beethoven's Bagatelle Nr 1 Opus 33 showed 56% of correctly detected notes, 42% of notes not detected which were in the original recording and 15% of notes detected which were not in the original MIDI file.

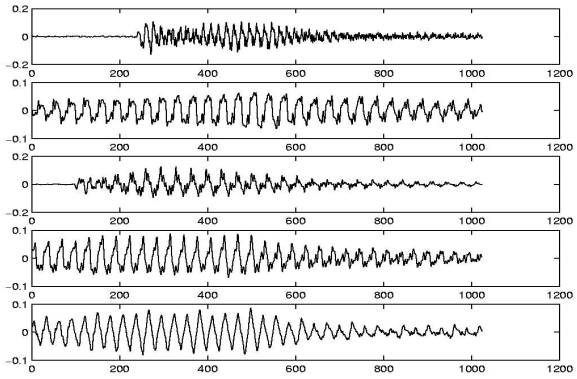

Fig. 1. A selection of 5 harmonic functions of the 50 functions learned.

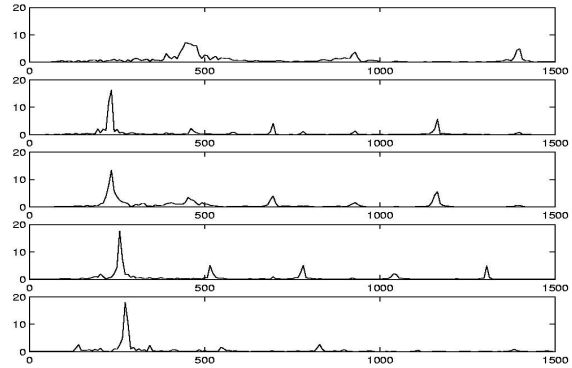

Fig. 2. The magnitude spectrum of the 5 functions of Fig.1 showing frequencies below 1500Hz. Higher frequency content was negligible.

5. CONCLUSION

We have developed an algorithm capable of learning statistically independent functions for a sparse signal representation using a shift-invariant approach. This was made possible by the use of an efficient subspace selection step reducing the computational burden of the required optimisation procedure.

We demonstrated the algorithm by learning functions from polyphonic piano recordings. Most of these functions corresponded to individual notes and the coefficients could be used to obtain approximate transcriptions of the score of the music played. The advantages of the presented approach are that it makes only a few assumptions on the signal. These are the independence of the individual coefficients, the fact that functions are not expected to overlap substantially with shifted versions of themselves (which introduces a conditional dependency on the location of the functions) and the sparsity of the coefficients. No assumptions have been made on the functions themselves (apart from the fixed length of the functions). The ability to learn a sparse and shift-invariant transform should therefore be applicable to other musical mixtures as well as to many other engineering problems.

The main restriction of the current implementation is the subspace selection step, which restricts the selected subspace to those functions with a high energy in the signal. Another shortcoming

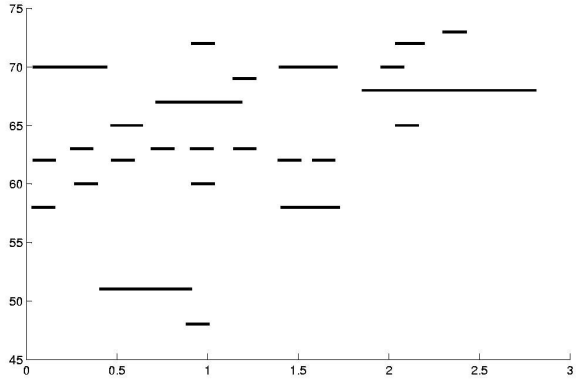

Fig. 3. An extract of the MIDI note representation obtained from the signal decomposition. The y axis shows MIDI note numbers between 45 and 75 and the x axis displays time in seconds.

is the attenuated high frequency content in the learned functions, which seems to be the result of the used model which only models shifts of a full sample, whilst the analysed signal is produced by a process in which functions can occur at arbitrary locations. These inaccuracies in the model as well as inaccuracies in inference of exact locations lead to learned functions which are filtered versions of the underlying features.

6. REFERENCES

[1] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, no. 381, pp. 607–609, 1995.

[2] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Computation*, no. 12, pp. 337–365, 2000.

[3] S. Abdallah, *Towards Music Perception by Redundancy Reduction and Unsuprvised Learning in Probabilistic Models*. PhD thesis, King's College London, February 2003.

[4] M. A. T. Figueiredo and A. K. Jain, "Bayesian learning of sparse classifiers," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - CVPR'2001*, (Hawaii), pp. 35–41 Vol. 1, IEEE, December 2001.

[5] B. D. Rao and I. F. Gorodnitsky, "Sparse signal reconstruction from limited data using focuss: a re-weighted minimum norm algorithm," *IEEE Transactions on Signal Processing*, vol. 45, pp. 600–616, March 1997.

[6] J. F. Murray and K. Kreutz-Delgado, "An improved FOCUSS-based learning algorithm for solving sparse linear inverse problems," *Conference Record of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers.*, 2001.

[7] B. D. Rao and K. Kreutz-Delgado, "An affine scaling methodology for best basis selection," *IEEE Transactions on Signal Processing*, vol. 47, pp. 187–200, January 1999.

[8] M. A. T. Figueiredo, "Adaptive sparseness for supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1150–1159, Sept. 2003.