SPARSE ATOMIC MODELING OF AUDIO: A REVIEW

Corey Kereliuk,
SPCL* & CIRMMT†
McGill University, Montréal, Canada
corey.kereliuk@mail.mcgill.ca

Philippe Depalle,
SPCL* & CIRMMT†
McGill University, Montréal, Canada
depalle@music.mcgill.ca

ABSTRACT

Research into sparse atomic models has recently intensified in the image and audio processing communities. While other reviews exist, we believe this paper provides a good starting point for the uninitiated reader as it concisely summarizes the state-of-the-art, and presents most of the major topics in an accessible manner. We discuss several approaches to the sparse approximation problem including various greedy algorithms, iteratively re-weighted least squares, iterative shrinkage, and Bayesian methods. We provide pseudo-code for several of the algorithms, and have released software which includes fast dictionaries and reference implementations for many of the algorithms. We discuss the relevance of the different approaches for audio applications, and include numerical comparisons. We also illustrate several audio applications of sparse atomic modeling.

1. INTRODUCTION

Many natural signals can be sparsely represented (or sparsely approximated) if an appropriate basis can be found. For example, a short block of samples from a quasi-periodic sound will have a sparse Fourier transform if the block size is a multiple of the pitch period. We often seek sparse representations, or sparse models, because they lead to a clear interpretation. If we compare several models that summarize a data set equally well, we usually prefer the sparser models, since each variable tends to be more meaningful[1]. This is especially true if we are interested in audio effects, since we desire a meaningful mapping between the control parameters and the perceived outcome.

In this paper we limit ourselves to the following model:

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \boldsymbol{\varepsilon} \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^M$ is a sampled sound, $\mathbf{\Phi} \in \mathbb{C}^{M\times N}$ is a dictionary of (possibly) complex atoms, and $\boldsymbol{\varepsilon}$ is additive noise. We call $\text{supp}(\mathbf{x}) = \{i | x_i \neq 0\}$ the support of $\mathbf{x}$, and define the sparsity of $\mathbf{x}$ as $\|\mathbf{x}\|_0 = |\text{supp}(x)|$, which is the cardinality of the support. We say that $\mathbf{y}$ is synthesis-sparse in the dictionary $\mathbf{\Phi}$ if $\|\mathbf{x}\|_0 \ll M$. In this paper we focus primarily on synthesis sparsity, however, it is worth noting that several recent works consider signals which are analysis-sparse: that is, signals for which $\|\Omega\mathbf{x}\|_0 \ll M$ (where Ω is an analysis operator) [1, 2].

Although, real sound signals may not be truly sparse in any basis, they are often *compressible*. We say that $\mathbf{y}$ is compressible in $\mathbf{\Phi}$ if the sorted magnitudes of $\mathbf{x}$ decay according to a power-law. This means that we may discard many of the small coefficients in $\mathbf{x}$ without a huge sacrifice in the perceived quality.

The formulation in (1) is quite common in audio processing since we may consider the wavelet transform, the modified discrete cosine transform (MDCT), and the short time Fourier transform (STFT) as instances of this model (if we choose $\mathbf{\Phi}$ appropriately and set $\boldsymbol{\varepsilon}$ to 0).

In this paper we focus on the case where $N > M$, which means that the dictionary contains a redundant set of waveforms. This situation arises quite naturally in audio processing. For example, the STFT is often oversampled so that *i)* smoother analysis windows may be used, and *ii)* to make the transform more invariant to shifts in the input signal. Both of these considerations lead to a redundant dictionary. Furthermore, it is often useful to build hybrid dictionaries from the union of several different dictionaries. This allows us to match the dictionary waveforms to the type of features we expect to encounter in the signal. As described in §10.1 this fact can be used to build multilayer signal expansions.

In the first part of this paper we examine several state-of-the-art approaches for estimating a sparse $\mathbf{x}$ given a redundant dictionary $\mathbf{\Phi}$. We discuss most of the major approaches and their variants. Along the way we point out which algorithms have the potential to work with the large data sets common in audio applications. In the second part of this paper we perform some numerical comparisons of these algorithms, and review some important audio applications that can benefit from sparse atomic modeling.

2. THE METHOD OF FRAMES

We first consider the case without an explicit noise term, i.e., $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$. There are many possible solutions that satisfy this equation when $N > M$ since $\mathbf{\Phi}$ has a null space (and adding an element from the null space does not change the solution). One possible solution is:

$$\mathbf{x} = \mathbf{\Phi}^H \boldsymbol{S}^{-1}\mathbf{y} \tag{2}$$

where $\boldsymbol{S} = \mathbf{\Phi}\mathbf{\Phi}^H$ is called the frame operator, and $\mathbf{\Phi}^H$ denotes the conjugate transpose. The frame operator is invertible if its minimum eigenvalue is greater than zero and its maximum eigenvalue is finite. In the finite dimensional case (which is the only case we consider in this paper), the latter condition is always satisfied, and the former condition is satisfied whenever $\mathbf{\Phi}$ has rank M (which is to say the columns of $\mathbf{\Phi}$ span M dimensional space). In the literature (2) is known as the method of frames (MOF) [3]. A very comprehensive review on frames can be found in [4].

The MOF solution is unique in the sense that $\mathbf{x}$ is orthogonal to the null space of $\mathbf{\Phi}$ and hence has the minimum 2-norm out of all possible solutions. As such the MOF can be viewed as the

[*] Sound Processing and Control Laboratory
[†] Centre for Interdisciplinary Research in Music Media and Technology
[1] This principle is often referred to as *Occam's razor*.

solution to the following problem:

$$\arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}\|_2^2 \quad \text{subject to} \quad \mathbf{y} = \mathbf{\Phi x} \tag{3}$$

As noted by several authors the MOF solution is usually *not sparse* due to the fact that the 2-norm places a very high penalty on large coefficients (and thus there tend to be many small yet significant coefficients) [5].

3. SPARSE APPROXIMATIONS

As noted in the introduction the ℓ_0 pseudo norm, $\|\mathbf{x}\|_0$, is a direct measure of sparsity. In light of the MOF formulation in (3) this leads us to question if we can solve the following problem:

$$\arg\min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \|\mathbf{y} - \mathbf{\Phi x}\|_2^2 \leq T \tag{4}$$

where T is proportional to the noise variance (or 0 in the noiseless case). Unfortunately, the solution to this problem involves the enumeration of all possible subsets of columns from $\mathbf{\Phi}$, which is a combinatorial problem [6]. This problem is also unstable in the noiseless case, since a small perturbation of $\mathbf{y}$ can dramatically effect $\|\mathbf{x}\|_0$.

There are two main strategies that have been explored in the literature to recover sparse solutions. The first tactic is to use greedy algorithms, which build up a solution by selecting one coefficient in $\mathbf{x}$ per iteration. By stopping a greedy algorithm early, a sparse approximation is guaranteed. The second strategy is to rephrase (4) using a cost function that can be tractably minimized. We refer to this approach as relaxation. Although both of these tactics often lead to suboptimal solutions, there are certain conditions under which the optimal solution may be recovered [7, 8].

In the following sections we explore these two approaches and discuss algorithms for their solutions. We then provide some numerical results to compare the different algorithms, and discuss their suitability for audio applications.

4. GREEDY APPROACHES

4.1. Matching Pursuit

The matching pursuit (MP) algorithm is one of the most well known greedy algorithms for sparse approximation [9]. In MP we start with an empty solution $\mathbf{x}^{(0)} = \mathbf{0}$, and adjust one coefficient in $\mathbf{x}$ at each iteration. This coefficient is chosen so as to minimize the residual error at each iteration. For example, on the n^{th} iteration we define $\mathbf{x}^{(n)} = \mathbf{x}^{(n-1)} + \alpha\boldsymbol{\delta}$, where α is a scalar and $\boldsymbol{\delta}$ is a unit vector with one non-zero component that indicates which element of $\mathbf{x}$ should be updated. The residual can then be written as $\mathbf{r}^{(n)} = \mathbf{y} - \mathbf{\Phi x}^{(n)} = \mathbf{r}^{(n-1)} - \alpha\boldsymbol{\phi}$, where the atom $\boldsymbol{\phi}$ is the column of $\mathbf{\Phi}$ identified by $\boldsymbol{\delta}$. At each iteration we seek an atom $\boldsymbol{\phi}$ and scalar α that minimize the current residual:

$$\arg\min_{\alpha\in\mathbb{C}, \boldsymbol{\phi}\in\mathbf{\Phi}} \frac{1}{2}\|\mathbf{r}^{(n-1)} - \alpha\boldsymbol{\phi}\|_2^2 \tag{5}$$

Solving for α we find

$$\alpha = \frac{\boldsymbol{\phi}^H\mathbf{r}^{(n-1)}}{\boldsymbol{\phi}^H\boldsymbol{\phi}} \tag{6}$$

where $\boldsymbol{\phi}^H\mathbf{r}^{(n-1)} = \sum_k \boldsymbol{\phi}^*[k]\mathbf{r}^{(n-1)}[k]$. We often normalize the dictionary atoms so that $\boldsymbol{\phi}^H\boldsymbol{\phi} = 1$ (we will assume this is the case from here on out). Plugging this value of α back into (5) it is straightforward to show that the atom which decreases the residual error most is given by

$$\arg\max_{\boldsymbol{\phi}\in\mathbf{\Phi}} |\boldsymbol{\phi}^H\mathbf{r}^{(n-1)}| \tag{7}$$

Algorithm 1 summarizes the steps in MP.

Algorithm 1 Matching Pursuit

```
1: init: $n = 0, \mathbf{x}^{(n)} = \mathbf{0}, \mathbf{r}^{(n)} = \mathbf{y}$
2: repeat
3:     $i_n = \arg\max_i |\boldsymbol{\phi}_i^H\mathbf{r}^{(n)}|$
4:     $\alpha_n = \boldsymbol{\phi}_{i_n}^H\mathbf{r}^{(n)}$
5:     $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \alpha_n\boldsymbol{\delta}_{i_n}$
6:     $\mathbf{r}^{(n+1)} = \mathbf{r}^{(n)} - \alpha_n\boldsymbol{\phi}_{i_n}$
7:     $n = n + 1$
8: until stopping condition
```

The stopping condition is usually based on a combination of the desired signal to residual ratio (SRR), and maximum number of iterations allowed.

After k iterations the signal approximation is

$$\hat{\mathbf{y}} = \sum_{n=0}^{k-1}\alpha_n\boldsymbol{\phi}_{i_n} \tag{8}$$

We can avoid explicit computation of the residual in algorithm 1 if we multiply both sides of line 6 by $\boldsymbol{\phi}_j^H$. This gives

$$\boldsymbol{\phi}_j^H\mathbf{r}^{(n+1)} = \boldsymbol{\phi}_j^H\mathbf{r}^{(n)} - \alpha_n\boldsymbol{G}[i_n, j] \tag{9}$$

where $\boldsymbol{G} = \mathbf{\Phi}^H\mathbf{\Phi}$ is the Gram matrix, and $\boldsymbol{\phi}_j^H\mathbf{r}^{(n)}$ was already calculated in the previous iteration. In practice the Gram matrix is often too big to be stored, however, in many cases it will have a sparse structure so an update of this form can still be useful. For example, when local dictionaries are used the majority of entries in the Gram matrix are zero, so many of the inner products do not need to be updated [10].

Equation (9) reveals that the inner products at iteration $n+1$ depend on the atom selected at iteration n (via the Gram matrix). When the atoms are correlated (as they will be in a redundant dictionary) this dependence can lead to the algorithm making suboptimal choices. Nonetheless, the residual is guaranteed to converge to zero in norm as the number of iterations tends to infinity [9].

4.2. Variants and Extensions

It should be noted that in MP we may select atoms at each iteration based on criteria other than minimizing the residual energy. For example, if we have some *a priori* knowledge about the signal we can modify the selection criteria to include this information. To illustrate, in [11] a psychoacoustic weighting was applied before minimizing the residual, and in [12] an MP-variant was introduced that avoids selecting atoms which might lead to pre-echo artifacts. Likewise, we might also restrict the search region for atoms at each iteration. For example, in [13] the search region was restricted so that only overlapping chains of atoms (similar to partials) were extracted by the algorithm. This flexibility in the selection of atoms is a great advantage of MP over some of the other algorithms introduced later.

With MP we can also refine the atom parameters at each iteration using Newton's method [9]. This allows one to find a continuous estimate for the atom parameters even when using a discrete dictionary. Also, in [14] a method known as cyclic matching pursuit was introduced that allows one to find a continuous estimate of the amplitude and frequency parameters when using a dictionary of complex sinusoids.

4.3. Orthogonalization

There are several variants of MP which use orthogonalization to improve the performance (i.e., achieve a higher SRR with fewer atoms). For example, we can update the coefficient vector $\mathbf{x}$ by orthogonal projection every k iterations. This process is known as backprojection. If we let $\Delta_k \triangleq \text{supp}(\mathbf{x}^{(k)})$, and denote $\mathbf{x}_{\Delta_k} \triangleq \{x_i | i \in \Delta_k\}$ and $\mathbf{\Phi}_{\Delta_k} \triangleq \cup_{i\in\Delta_k} \boldsymbol{\phi}_i$, then we can write backprojection as:

$$\hat{\mathbf{x}}_{\Delta_k} = \arg \min_{\text{supp}(\mathbf{x})=\Delta_k} \frac{1}{2}\|y - \mathbf{\Phi}_{\Delta_k}\mathbf{x}\|_2^2 \tag{10}$$

The solution to this equation is given by $\hat{\mathbf{x}}_{\Delta_k} = \mathbf{\Phi}^+_{\Delta_k} y$, where $\mathbf{\Phi}^+_{\Delta_k}$ indicates the pseudo-inverse.

If we carry backprojection to its logical extreme, and update the coefficients after every iteration we arrive at an MP-variant known as orthogonal matching pursuit (OMP) [15]. OMP tends to be very computationally expensive, since it requires computation and inversion of the partial Gram matrix at every iteration. However, since the residual remains orthogonal to the selected atoms at every iteration, OMP will never select the same atom twice (this is not the case for MP), and is guaranteed to converge in M (or fewer) iterations.

There are also several variants of OMP that have been discussed in the literature. For example, in optimized OMP (OOMP) [16] the atom selection metric is adjusted in order to improve the residual decay, and in stagewise OMP (StOMP) [17] several atoms are selected and orthogonalized at each iteration.

In [18] several fast algorithms were introduced which approximate OMP using gradient and conjugate gradient information. Further, in [10] a fast approximate OMP algorithm was proposed for use with local dictionaries. This algorithm exploits the fact that many dictionaries of practical interest are local in the sense that the majority of atoms are orthogonal to one another (since they are supported on disjoints sets). This allows one to work with a much smaller partial Gram matrix, and dramatically speeds up the algorithm in practice. As such these algorithms hold considerable promise for audio applications.

4.4. Conjugate Subspaces

In standard MP we select just one atom at each iteration. When working with complex atoms and a real signal it can be useful to select a conjugate subspace at each iteration. This can be done by replacing α by $[\alpha \quad \alpha^*]^T$ and $\boldsymbol{\phi}$ by $[\boldsymbol{\phi} \quad \boldsymbol{\phi}^*]$ in (5) which leads to the solution outlined in [5].

Using complex atoms with conjugate subspaces leads to two important advantages when working with audio. Firstly, using complex atoms allows one to estimate the phase without explicitly parameterizing this value. Second, when selecting low or high frequency atoms the inner products can be biased by spectral leakage from the negative frequency spectrum. Since this approach selects a subspace consisting of one positive and one negative frequency atom, it is resilient to this possible bias (further details can be found in [19]).

4.5. Weak Matching Pursuit

A modification to MP known as weak matching pursuit (WMP) can be practically useful when dealing with very large dictionaries, where the computation of inner products would ordinarily be prohibitive [9, 20]. At each iteration of WMP we select an atom from a subset of the full dictionary:

$$\mathbf{\Phi}_\Lambda^{(n)} = \left\{ \boldsymbol{\phi}_i \,\middle|\, \left|\boldsymbol{\phi}_i^H \mathbf{r}^{(n)}\right| \geq \beta \max_j \left|\boldsymbol{\phi}_j^H \mathbf{r}^{(n)}\right| \right\} \tag{11}$$

where $\beta \in (0, 1]$ is a relaxation factor. It has been shown that the WMP will converge even if β changes from iteration to iteration [21]. In [22] and [23] this strategy was used to prune the dictionary around local maxima, leading to a significant computational savings.

5. RELAXED APPROACHES

As mentioned in §3 the second major class of algorithms for sparse approximation are based on relaxation. In essence, we relax the hard ℓ_0 pseudo norm problem by replacing it with a cost function that can be tractably minimized.

In order to proceed let us replace the ℓ_0 pseudo norm in (4) by a function $f(\mathbf{x})$ that measures the sparsity of $\mathbf{x}$:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{subject to} \quad \|\mathbf{y} - \mathbf{\Phi x}\|_2^2 \leq T \tag{12}$$

This equation can also be written in an equivalent unconstrained form:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi x}\|_2^2 + \lambda f(\mathbf{x}) \tag{13}$$

We refer to $f(\mathbf{x})$ as a regularization term, and note that the scalar λ controls the degree of regularization (i.e., trades off our desire for a sparse solution with our wish for a low approximation error).

There are many regularizers that promote sparsity. For example, the ℓ_p-norms, $0 \leq p \leq 1$ are well-known to promote sparsity:

$$f(\mathbf{x}) = \|\mathbf{x}\|_p^p = \sum_i |x_i|^p \tag{14}$$

We can begin to see why the ℓ_p-norms, $0 \leq p \leq 1$, promote sparsity by visualizing the shape of ℓ_p-balls in two dimensions. In fig. 1 the feasible set of solutions for a hypothetical problem is indicated by a dashed line. The minimum ℓ_p-norm solution is found by expanding the ℓ_p-ball until it intersects the solution space. As can be seen for $p \leq 1$ the ℓ_p-ball intersects the solution space along one of the coordinate axes, leading to a solution with only one non-zero component. Notice that when $p > 1$ the solution contains two non-zero components. The solution for $p = 2$ corresponds to the minimum energy solution (which is calculated using the method of frames). Geometrically the ℓ_p-balls, $0 \leq p \leq 1$, are sharply pointed and aligned with the coordinate axes, which tends to induce sparse solutions.

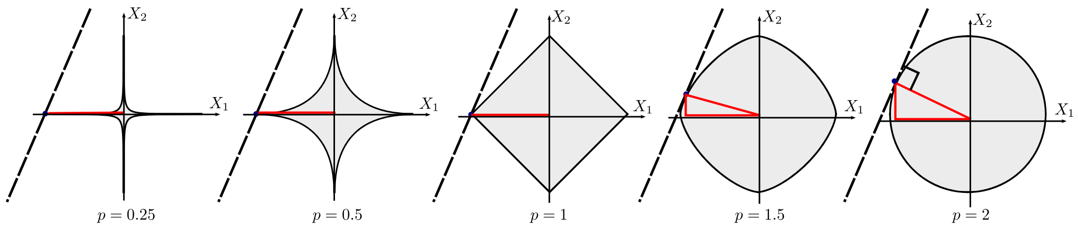

Figure 1: Illustration of the shape of ℓ_p-balls in 2-dimensions. The dashed line represents the set of feasible solutions for a hypothetical problem. Note that for $p > 1$ the minimum ℓ_p-norm solution contains 2 non-zero components, whereas for $p \leq 1$, the solution contains only 1 non-zero component. Also note that $p = 2$ is the minimum energy solution (it is the vector normal to the solution space).

Other sparsity measures are also possible, for example, the Shannon and Rényi entropies are known to act as good sparsity measures[2] [24, 25].

5.1. Basis Pursuit

The fact that ℓ_1 minimization often leads to sparse solutions has been known for sometime [26]. In the signal processing literature minimization of (12) with an ℓ_1-norm regularization term is known as basis pursuit (BP) when $T = 0$ and basis pursuit denoising (BPDN) when $T \neq 0$ [27]. In the statistics literature a very similar formulation was presented under the name least absolute shrinkage and selection operator (LASSO) [28].

Using the ℓ_1-norm is attractive since *i)* it promotes sparsity, and *ii)* it is convex (and thus this problem can be tackled using the large body of techniques developed for convex optimization [29]).

In [27] the BP problem was solved using linear programming (LP), however, as pointed out in [30], LP cannot be used with complex coefficients. In this case a second-order cone program (SOCP) may be used instead.

In [30] it was noted that we can downsample a sparse signal using random projections (using results from compressed sensing (CS) theory [31]). This strategy was used in [30] to downsample the input signal before applying a SOCP. This significantly reduces the problem size and hence the computational cost (which is very interesting to note for audio applications).

In the following sections we discuss algorithms for the solution to the unconstrained problem (13).

5.2. Iteratively Re-weighted Least Squares

Iteratively re-weighted least squares (IRLS) is an algorithm that can be used to solve the sparse approximation problem with both convex and non-convex regularization terms. The premise of IRLS stems from the following fact: if we define a diagonal weight matrix as:

$$W_p = \text{diag}(|x_i|^{p-2}) \tag{15}$$

then we can write the ℓ_p-norm of $\mathbf{x}$ in quadratic form as follows:

$$\|\mathbf{x}\|_p^p = \mathbf{x}^H W_p \mathbf{x} = \sum_i x_i^2 |x_i|^{p-2} = \sum_i |x_i|^p \tag{16}$$

This allows us to write (13) as:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 + \lambda \mathbf{x}^H W_p \mathbf{x} \tag{17}$$

The least squares solution to this equation is:

$$\hat{\mathbf{x}} = (\mathbf{\Phi}^H\mathbf{\Phi} + 2\lambda W_p)^{-1}\mathbf{\Phi}^H\mathbf{y} \tag{18}$$

However, since $W_p = \text{diag}(|x_i|^{p-2})$ is a function of $\mathbf{x}$, we must solve this equation in an iterative fashion. The pseudocode in algorithm 2 demonstrates the basic IRLS algorithm. To avoid division by zero, we initialize $\mathbf{x}$ with all ones. In practice many of the coefficients of $\mathbf{x}$ will shrink, but never reach zero. A variation on this algorithm could include the identification of an *active set* of coefficients from $\mathbf{x}$. Small coefficients from $\mathbf{x}$ (and the associated columns from $\mathbf{\Phi}$) could then be pruned from the active set.

Algorithm 2 IRLS

```
init: $n = 0, \mathbf{x}^{(n)} = \mathbf{1}$
repeat
    $W_p^{(n)} = \text{diag}(|x_i^{(n)}|^{p-2})$
    $\mathbf{x}^{(n+1)} = (\mathbf{\Phi}^H\mathbf{\Phi} + 2\lambda W_p^{(n)})^{-1}\mathbf{\Phi}^H\mathbf{y}$
    $n = n + 1$
until stopping condition
```

In [32] a slightly different procedure is described, whereby a solution is sought in the noiseless case. This algorithm has the same form as algorithm 2, except that at each iteration the updates proceed according to

$$\mathbf{x}^{(n+1)} = (W^{(n)})^{-1}\mathbf{\Phi}^H(\mathbf{\Phi}(W^{(n)})^{-1}\mathbf{\Phi}^H)^{-1}\mathbf{y} \tag{19}$$

This variant of IRLS is referred to as the focal underdetermined system solver (FOCUSS) in the literature [33]. A detailed examination of IRLS and its convergence properties is provided in [34].

There are several points that should be noted regarding the IRLS algorithm. Firstly, the algorithm is sensitive to the initialization point when $p < 1$, which means a local minimum could be returned. In many applications we may be able to find a suitable initialization point using other algorithms (e.g., we could use the pseudo-inverse as an initialization point). Second, the algorithm requires a matrix inversion. It can be practical to perform the matrix inversion using Cholesky or QR factorization for small dictionaries. However, when working with audio, these factorizations are usually not practical due to their computational complexity, and because we often can't explicitly store the dictionary in memory. In this case we can perform the matrix inversion using conjugate gradient descent with appropriate preconditioning as suggested in [35]. As noted in [36] when $\mathbf{\Phi}$ is an orthonormal basis the matrix inverse is trivial. Furthermore, if $\mathbf{\Phi}$ is a union of orthonormal bases, we can invert one basis at a time in an iterative fashion using block coordinate relaxation (BCR) [37].

[2] In fact, the Rényi entropies can be interpreted as logarithmic versions of the ℓ_p-norms.

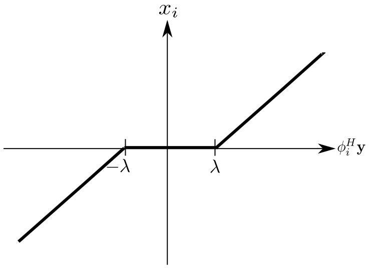

Figure 2: The shrinkage curve $x_i = \psi(\boldsymbol{\phi}_i^H \mathbf{y})$ for the ℓ_1-norm function.

5.3. Iterative Shrinkage

In [38] a method known as shrinkage was introduced to solve (13) for the case when $\mathbf{\Phi}$ is an orthonormal basis. To understand shrinkage, we start by re-writing the objective function from (13) as:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{\Phi}(\mathbf{\Phi}^{-1}\mathbf{y} - \mathbf{x})\|_2^2 + \lambda f(\mathbf{x}) \tag{20}$$

when $\mathbf{\Phi}$ is an orthonormal basis, this simplifies to[3]:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{\Phi}^H\mathbf{y} - \mathbf{x}\|_2^2 + \lambda f(\mathbf{x}) \tag{21}$$

$$= \arg\min_{\mathbf{x}} \sum_i \frac{1}{2}(\boldsymbol{\phi}_i^H\mathbf{y} - x_i)^2 + \lambda f(x_i) \tag{22}$$

In this form the joint optimization problem has been factored into a sum of scalar optimization problems which can be solved individually.

The solution to the i^{th} scalar optimization problem is given by:

$$x_i + \lambda f'(x_i) = \boldsymbol{\phi}_i^H\mathbf{y} \tag{23}$$

Solving for x_i we get:

$$x_i = \psi(\boldsymbol{\phi}_i^H\mathbf{y}) \tag{24}$$

where

$$\psi^{-1}(u) = u + \lambda f'(u) \tag{25}$$

When f is the ℓ_1-norm we find that $\psi^{-1}(u) = u + \lambda\text{sign}(u)$, which leads to:

$$\psi(u) = \text{sgn}(u)\max(0, |u| - \lambda) \tag{26}$$

The curve of the ℓ_1 shrinkage function, ψ, is graphed in figure 2. Filtering the transform coefficients according to this graph is known as shrinkage or soft thresholding. It was shown in [39] that shrinkage can be performed with complex coefficients by simply shrinking the modulus of $\boldsymbol{\phi}_i^H\mathbf{y}$ and leaving the argument unchanged. We also note that different regularization terms will lead to different shrinkage curves[4]. For example, in the limit as $p \to 0$ we obtain a variant known as hard thresholding [40].

We now discuss how to perform shrinkage with an overcomplete dictionary. In this case an iterative shrinkage (IS) algorithm is required, whereby a series of simple shrinkage operations are performed until convergence [41, 42, 43, 44, 40]. Here we outline the approach discussed in [43].

In this approach we again decouple the optimization problem and solve a series of 1-D problems. Assume we are given the transform coefficients at the n^{th} iteration, $\mathbf{x}^{(n)}$. Now assume all entries in $\mathbf{x}^{(n)}$ are fixed, except for the i^{th} entry, which we wish to refine. We can write the objective function as:

$$\arg\min_{w} \frac{1}{2}\|\mathbf{y} - (\mathbf{\Phi}\mathbf{x}^{(n)} - \boldsymbol{\phi}_i x_i^{(n)} + \boldsymbol{\phi}_i w)\|_2^2 + \lambda f(w) \tag{27}$$

In essence this removes the contribution of the i^{th} atom from the model, and allows us to replace it with a new estimate, w. Taking the derivative with respect to w and setting the result to zero we get (assuming unit norm atoms):

$$w + \lambda f'(w) = x_i^{(n)} + \boldsymbol{\phi}_i^H(\mathbf{y} - \mathbf{\Phi}\mathbf{x}^{(n)}) \tag{28}$$

which is in the same form as (23), and so can be solved using a shrinkage operator. The pseudocode for iterative shrinkage (IS) listed in algorithm 3:

Algorithm 3 Iterative Shrinkage (IS)

```
init: $n = 0, \mathbf{x}^{(n)} = \mathbf{0}$
repeat
    $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)}$
    for $i = 0$ to $N - 1$ do
        $x_i^{(n+1)} = \psi(x_i^{(n)} + \boldsymbol{\phi}_i^H(\mathbf{y} - \mathbf{\Phi}\mathbf{x}^{(n+1)}))$
    end for
    $n = n + 1$
until stopping condition
```

In [43], empirical results are presented comparing IS to IRLS. Although IRLS has much faster convergence, it also requires a matrix inversion, which can be prohibitive for large dictionaries. Iterative shrinkage on the other hand, converges more slowly, but it is computationally much simpler (and does not require a matrix inversion), so it can easily be used with large overcomplete dictionaries.

It is important to note that IS tends to underestimate the magnitude of the synthesis coefficients. We can correct for this bias after running IS by taking the orthogonal projection of $\mathbf{y}$ onto the support identified by the algorithm. In practice there may be some coefficients that are very small (but non-zero). These coefficients can be eliminated by hard thresholding prior to the debiasing step.

The particular version of shrinkage in algorithm 3 is somewhat slow because it requires each x_i to be updated sequentially in the inner loop. In [43] a simple modification of this algorithm was introduced that allows all of the coefficients in $\mathbf{x}$ to be updated in parallel. This leads to a significant speed up in the algorithm. We note that the IS algorithm developed in [42] also uses a parallel update.

There have also been several recent papers introducing fast IS techniques [45, 46]. These algorithms use information from the two previous iterates to update the current solution, and can be up to an order of magnitude faster. These fast shrinkage algorithms would likely be quite useful for audio applications.

5.4. Bayesian Methods

In a probabilistic setting we assume the signal is constructed according to:

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \boldsymbol{\varepsilon} \tag{29}$$

[3] by Parseval's theorem $||\mathbf{\Phi}z|| = ||z||$, and the fact that $\mathbf{\Phi}^{-1} = \mathbf{\Phi}^H$

[4] Not all shrinkage functions can be calculated analytically. In such a case a look-up table can be used.

where $\mathbf{x}$ and $\boldsymbol{\varepsilon}$ are random variables representing the signal and noise, respectively[5]. We will assume that $\boldsymbol{\varepsilon}$ is white Gaussian noise with covariance matrix $\Sigma = \sigma^2 I$ in order to simplify calculations (however more general noise models are certainly possible).

The negative log-likelihood of $\mathbf{x}$ is given by

$$\mathcal{L}(\mathbf{x}) = -\log p(\mathbf{y}|\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{\Phi x}\|_2^2 + C \tag{30}$$

where C is a constant independent of $\mathbf{x}$. As discussed in §2 when $\mathbf{\Phi}$ is overcomplete there are multiple values of $\mathbf{x}$ such that $\mathbf{y} = \mathbf{\Phi x}$, which means the maximum likelihood solution is ill-defined.

In order to find sparse solutions we can instead find the maximum *a priori* (MAP) estimate which incorporates a prior on the transform coefficients. The MAP estimate is found by application of Baye's rule:

$$\hat{\mathbf{x}}_{MAP} = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}) = \arg\max_{\mathbf{x}} \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})} \tag{31}$$

It is usually mathematically simpler to minimize the negative log-likelihood which is:

$$\hat{\mathbf{x}}_{MAP} = \arg\min_{\mathbf{x}} -\log p(\mathbf{y}|\mathbf{x}) - \log p(\mathbf{x}) + \log p(\mathbf{y}) \tag{32}$$

Let us denote the negative log-probability of $\mathbf{x}$ as $f(\mathbf{x}) = -\log p(\mathbf{x})$. Then (32) becomes

$$\hat{\mathbf{x}}_{MAP} = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{\Phi x}\|_2^2 + f(\mathbf{x}) + C \tag{33}$$

where C is a constant independent of $\mathbf{x}$. Notice that (33) is essentially the same problem as (13). It is interesting to note that when $p(\mathbf{x})$ is an independent and identically distributed (i.i.d.) Laplacian prior:

$$p(\mathbf{x}) = \prod_i p(x_i) = \prod_i \frac{\lambda}{2}\exp(-\lambda|x_i|) \tag{34}$$

then

$$f(\mathbf{x}) = -\log p(\mathbf{x}) = \lambda\|\mathbf{x}\|_1 + C \tag{35}$$

In other words, MAP estimation with a Laplacian prior is equivalent to ℓ_1 regularization. The Laplacian distribution is sharply peaked at zero with heavy tails as illustrated in fig. 3. This density thus encourages many small coefficients, yet does not place a heavy penalty on large coefficients, which tends to promote sparse solutions. This new viewpoint helps to illustrate why the ℓ_1-norm acts as a good sparsity measure. It should be noted that this is just one possible interpretation, and that other interpretations are certainly possible as suggested in the recent paper [48].

Since we are free to investigate priors other than the Laplacian, and also because we can use alternative noise models, the Bayesian approach is quite flexible. For example, in [49], a piecewise continuous prior was used, which is even more peaked around zero than the Laplacian prior. In [50], a design methodology is discussed outlining some of the necessary conditions for a prior to be sparsity inducing.

It should be noted that in the above formulation the synthesis coefficients were assumed to be i.i.d.. In reality this may not be a good assumption, since we expect some structure in the synthesis coefficients (for example chains of coefficients that form partials). In fact, all of the algorithms discussed up to this point ignore this important factor. We discuss methods for sparse and structured decompositions in §6.

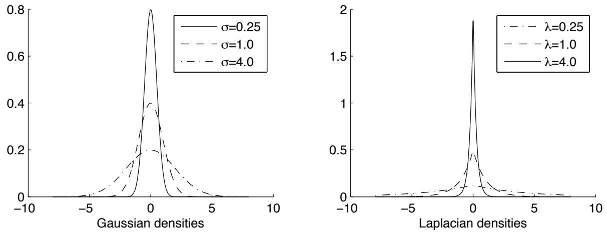

Figure 3: Illustration of Gaussian densities (left) and Laplacian densities (right).

5.4.1. Bayesian Variants and Extensions

There are several variations that can be applied within the Bayesian framework. For example, in [51] [52] [53] each transform coefficient is modeled by a zero-mean Gaussian prior. A hyper-prior is then placed on the variance, in order to express the belief that, for the majority of coefficients, the variance should be near zero [53]. As illustrated in fig. 3, as the variance of a Gaussian tends to zero, the distribution becomes infinitely peaked at zero. An expectation maximization (EM) procedure can be used to find MAP solutions for these models. A conceptually similar approach is discussed in [54].

Another variant known as sparse linear regression was introduced in [55] and [56]. In this formulation the atomic model is augmented to include a binary indicator variable $\gamma_i \in \{0, 1\}$:

$$\mathbf{y} = \sum_{i=0}^{N-1} \gamma_i x_i \boldsymbol{\phi}_i + \boldsymbol{\varepsilon} \tag{36}$$

The vector $\boldsymbol{\gamma} = [\gamma_0, \ldots \gamma_{N-1}]^T$ indicates the presence or absence of each coefficient in the model. If we could somehow determine a sparse indicator vector, then we could find the optimal transform coefficients via orthogonal projection. This problem setup is known in the literature as Bayesian variable selection [57].

Using the indicator variables we can form a mixed prior for each transform coefficient:

$$p(x_i|\gamma_i, \sigma_i) = (1 - \gamma_i)\delta(x_i) + \gamma_i \mathcal{N}(x_i|0, \sigma_i^2) \tag{37}$$

where $\mathcal{N}(\cdot|0, \sigma^2)$ indicates a zero-mean Gaussian density with variance σ^2, and $\delta(\cdot)$ is a dirac distribution. This *spike + slab* distribution enforces sparsity conditionally on γ. We then seek a MAP solution of the form

$$(\hat{\mathbf{x}}, \hat{\boldsymbol{\gamma}}) = \arg\max_{\mathbf{x}, \boldsymbol{\gamma}} p(\mathbf{x}, \boldsymbol{\gamma}|\mathbf{y}) \tag{38}$$

where the density $p(\mathbf{x}, \boldsymbol{\gamma}|\mathbf{y})$ can be found by marginalizing the complete posterior. The type of prior we place on $\boldsymbol{\gamma}$, strongly effects the type of solutions that will be found. For example, an independent Bernoulli prior could be used if we don't expect any dependancies between coefficients. In [58] the indicator variables are given time-frequency dependencies by modeling the joint distribution of $\boldsymbol{\gamma}$ as a Markov chain. In general the density $p(\mathbf{x}, \boldsymbol{\gamma}|\mathbf{y})$ cannot be found analytically. In this case, a solution can be found using Monte Carlo inference (e.g., Gibbs sampling).

6. STRUCTURED APPROXIMATIONS

The majority of techniques discussed to this point simply aim at recovering a sparse solution to (1). In addition to sparsity, we of-

[5] In some cases $\mathbf{\Phi}$ can also be considered a random matrix, for example, if we wish to perform dictionary learning [47].

ten have other *a priori* information regarding the content of our signals. In audio signals we expect strong time-frequency dependencies between different transform coefficients. For example, we expect some time-continuity between active coefficients in tonal sounds, and some frequency-continuity in transient sounds. Furthermore, we expect spatial dependencies in multichannel recordings [59].

There are several ways of structuring the information in a decomposition. We can impose structure directly using the atoms themselves (for example, by using harmonic atoms). Likewise, we can impose structure by preferentially selecting coherent atoms in the decomposition (for example, by only selecting overlapping chains of atoms). Finally, we can impose structure as a post-processing step, by clustering atoms according to their similarity. All of these approaches have been used in the literature. We briefly review the most prominent techniques.

Sparse linear regression, which was addressed in the previous section, uses binary indicator variables to indicate the absence or presence of each transform coefficient in the model. The type of joint prior placed on the set of indicator variables can be used to model dependencies between the transform coefficients, as was done in [55][58].

An algorithm known as molecular matching pursuit (MMP) is described in [60]. MMP is an iterative greedy algorithm that estimates and subtracts a tonal or transient molecule from the residual at each stage. Tonal molecules are defined as overlapping chains of MDCT atoms with the same frequency (± 1 bin). Transient molecules are sets of wavelet coefficients forming a connected tree. A similar algorithm is described in [13] which uses a multi-scale Gabor dictionary. These algorithms segment the signal into transient and tonal objects, which could be useful for selective processing and filtering of audio signals. There are strong similarities between MMP and partial tracking algorithms [61, 62], especially when an overlap-add (OLA) synthesis model is used. However, the advantage of MMP and its variants over traditional partial tracking techniques is that multi-scale features can be captured by the algorithm, since larger, more general dictionaries can be used.

An algorithm known as harmonic matching pursuit (HMP) was suggested in [23]. This algorithm uses a dictionary of harmonic atoms[6]. At each stage in the pursuit the signal is projected onto a harmonic subspace. In order to manage the complexity of the algorithm the search for best harmonic atom is limited to a subset of the dictionary using a weak approximate MP. The same authors also developed stereo atoms for two channel recordings [59].

The authors in [63] also discussed an approach inspired by both harmonic matching pursuit and molecular matching pursuit. Their mid-level representation was shown to be useful for several tasks including solo and polyphonic instrument recognition.

In [64] a post-processing technique known as agglomerative clustering (AC) was introduced to impose structure on the synthesis coefficients. AC works by traversing an adjacency matrix, which measures the similarity between atoms. If two atoms are present in the decomposition, and significantly close in the adjacency matrix, then they are grouped together. This process is then repeated to form large clusters of coherent atoms.

In [65] a technique using iterative shrinkage (see §5.3) was developed to find sparse and structured decompositions when used with time-frequency dictionaries. This technique relies on a mixed-norm regularization term which tends to induce structure in the decomposition (for details see [65] and the references therein).

Package	MP	OMP	IRLS	IS	BP	Languages	URL
SparseLab	✓	✓	✓	✓	✓	Matlab	[66]
Sparsify	✓	✓	-	-	-	Matlab	[67]
MPTK	✓	-	-	-	-	C++[7]	[68]
GabLab	✓	✓	✓	✓	-	Matlab	[69]

Table 1: Software packages for sparse approximation.

7. SOFTWARE TOOLS

Table 1 lists several software packages for sparse approximation that are freely available. This list is by no means exhaustive, but it does include some of the more well-known choices that are available.

In practice not all packages are well-suited for audio analysis. For example, many of the solvers in SparseLab require explicit access to the columns or rows of the dictionary. This is problematic for audio analysis, since we often work with huge amounts of data and dictionaries that aren't explicitly stored.

In practice MPTK is very fast, and contains many optimizations that make it suitable for use with audio and very large shift-invariant dictionaries.

The Gablab software [69] (which has been released in conjunction with this paper) was written with audio analysis in mind. GabLab comes bundled with functions for creating fast Gabor dictionaries, and unions of Gabor frames[8]. The computation of inner products in GabLab is performed using the fast Fourier transform. Furthermore, all of the algorithms in GabLab work with complex atoms.

In the following sections we compare the four main algorithms discussed in this paper (MP, OMP, IS and IRLS) according to several different criteria. We then discuss several audio applications that could benefit from sparse atomic modeling. All of the numerical experiments were performed using GabLab.

8. COMPUTATIONAL COMPLEXITY

A detailed analysis of the computational complexity of MP and OMP for general and fast local dictionaries can be found in [10]. We note that MP and IS are each dominated by the calculation of inner products, and thus have a similar computational complexity. However, when local dictionaries are used the inner product update can be performed faster in the MP algorithm (since fewer inner products need to be calculated for each iteration).

OMP and IRLS are both dominated by the calculation and inversion of the (partial) Gram matrix at each iteration. In GabLab this matrix inversion is performed using conjugate gradient descent.

[6] a harmonic atom is defined as a weighted sum of Gabor atoms with integer frequency ratios.

[7] Matlab bindings are bundled with the MPTK distribution.

[8] A Gabor frame can be viewed as a generalization of an oversampled STFT matrix.

The difference in speed between these algorithms depends to a large degree on the number of iterations required for convergence. In the following section we provide empirical results which illustrate how many iterations of each algorithm are required in specific test cases.

9. COMPARISON

In this section we compare the performance of MP, OMP, IS (p=1), and the IRLS (p=1) algorithm using some simple synthetic test signals. In the following tests we used a 3-scale complex Gabor dictionary constructed from Hann windows of length 2048, 512, and 64 samples with 50% overlap. The sampling rate used was 44.1kHz.

9.1. Example: a compressible signal

For the first test we used a quadratic chirp swept between 100 Hz and 15 kHz. As can be seen from the spectrogram in fig. 4, this signal is compressible in the Fourier domain, i.e., many of the coefficients are small. Furthermore, since the bandwidth of the chirp evolves over time, a multi-scale dictionary (such as the one proposed above), should posses the capability to model both the narrowband and wideband parts of the chirp respectively.

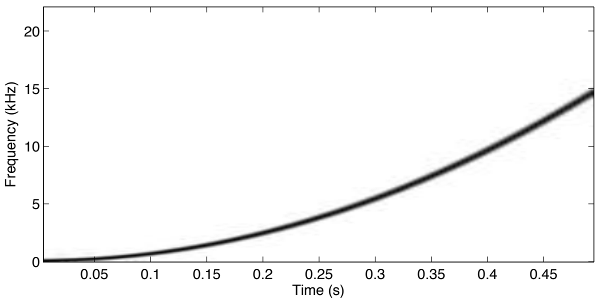

Figure 4: Spectrogram of quadratic chirp test signal. Spectrogram parameters: Hann window of length 512, with 50% overlap.

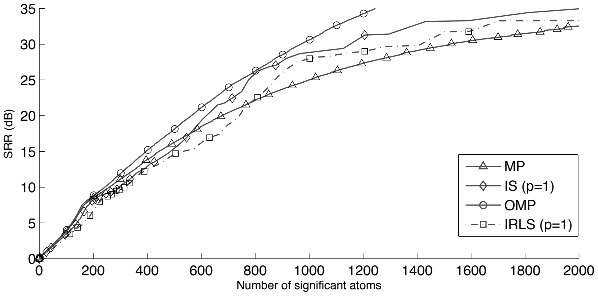

Figure 5: SRR vs. Number of significant atoms for quadratic chirp decomposition.

SRR	10 dB			20 dB			30 dB		
Algorithm	iterations	# of atoms	cpu time (s)	iterations	# of atoms	cpu time (s)	iterations	# of atoms	cpu time (s)
MP	131	260	8.0	354	678	21.4	844	1521	51.1
IS	92	292	**6.6**	151	618	**10.6**	230	1127	**16.5**
OMP	122	**242**	116.0	282	**560**	381.2	484	**960**	1224.6
IRLS	**73**	315	89.0	**69**	733	127.2	**67**	1406	206.5

Table 2: Summary of quadratic chirp decomposition results.

Figure 5 shows the signal to residual ratio (SRR) vs. the number of significant atoms in the decomposition for each algorithm. An atom was deemed significant if its magnitude exceeded 10^{-4}. For IS and IRLS the data points were generated by running the algorithms until convergence for different values of λ. It should be noted that this process can be accelerated significantly using 'warm starts', i.e., initializing the next run of the algorithm using the previous value at convergence. This works in practice because a small change in λ usually causes a small change in the solution. It should also be noted that after IS and IRLS converged the coefficient vector was thresholded (with threshold 10^{-4}) and debiased as explained in section §5.3.

Examining the curves in fig. 5, we see that OMP provides the best SRR vs. number of atoms. MP, IS, and IRLS perform similarly, although beyond 800 atoms, IS and IRLS both maintain a higher SRR than MP. Table 2 summarizes the data in this graph and adds two new pieces of information: i) the number of iterations required for convergence and, ii) the amount of CPU time used by each of the algorithms (for reference, all of the algorithms were run in MATLAB®on the same 2.6 GHz dual core Mac Pro). Of course, the speed of these algorithm is implementation dependent. However, combined with the number of iterations required for convergence these numbers do reveal interesting differences between the various algorithms.

As described in §8, IS and MP have approximately the same complexity per iteration, however, as seen in table 2, IS requires fewer iterations to converge than MP, and hence uses less CPU time. The difference is more dramatic for high SRRs and, although not shown in table 2, for very low SRRs MP is indeed faster.

9.2. Example: a sparse signal plus noise

For this example, we generated a random sparse signal using the 3-scale Gabor dictionary introduced in the previous section. This signal was generated by first drawing 500 indices from a uniform distribution to make up the support vector. The real and imaginary coefficients were then drawn from a normal distribution with unit mean and variance 0.1. Conjugate atoms were also added to make the signal real. The test signal was 0.5s in duration and contained 984 non-zero coefficients[9]. We then added white Gaussian noise to the signal so that the SNR was 5dB and compared the performance of MP, OMP, IS and IRLS at denoising the signal.

Figure 6 displays the output SNR vs. number of atoms for each of the algorithms. Near the true sparsity level (984 atoms), OMP offers the best reconstruction in terms of output SNR. MP has its peak located in a similar location, although the SNR is lower[10]. Both IS and IRLS require more atoms to reach their peak SNR,

[9] There are slightly less than 1000 non-zero coefficients because repeated coefficients were discarded, and some coefficients were DC atoms (so no conjugate was added).

[10] Stopping the algorithm here and running backprojection would probably result in a better performance.

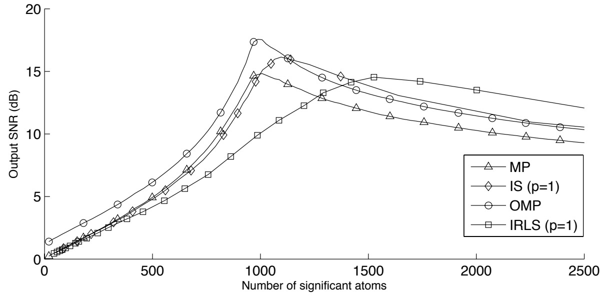

Figure 6: Results for de-noising a random sparse signal corrupted by white Gaussian noise (input SNR = 5dB).

although IS achieves its peak with fewer significant atoms than IRLS.

In order to compare how well the true support was recovered we also measure the Type I and II errors which are defined as follows. If we let Δ represent the true support and $\hat{\Delta}$ represent the estimated support:

$$\text{Type I error} \triangleq 1 - \frac{|\Delta \cap \hat{\Delta}|}{|\Delta|} \tag{39}$$

$$\text{Type II error} \triangleq 1 - \frac{|\Delta \cap \hat{\Delta}|}{|\hat{\Delta}|} \tag{40}$$

Figure 7 illustrates the Type I and Type II errors vs. the number of atoms for each of the algorithms. MP, OMP, and IS all have very low errors near 984 atoms (the true sparsity level), which suggests that these algorithms do a good job recovering the correct support. IRLS on the other hand, has a harder time recovering the true support. It is interesting to note the differences between IS and IRLS since both algorithms attempt to minimize the same cost function. We must remember however, that this cost function is not *strictly* convex, which means there could be multiple solutions with the same cost.

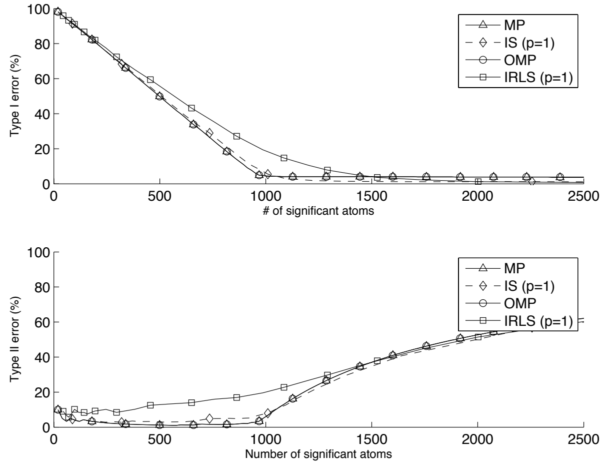

Figure 7: Error in support estimation. Top: Type I error. Bottom: Type II error.

10. AUDIO APPLICATIONS

In this section we highlight some audio applications that can benefit from sparse atomic modeling.

10.1. Multilayer Expansions

Multilayer expansions are often very useful for audio processing and effects. For example, the *tonal* + *transient* + *noise* expansion segments the signal into important perceptual units.

To find a multilayered expansion we start by defining the dictionary as $\mathbf{\Phi} = \cup_{i=1}^{I} \mathbf{\Phi}_i$, where each $\mathbf{\Phi}_i$ is a frame adapted to a certain signal feature. For example, for $I = 2$, we might take $\mathbf{\Phi}_1$ to be a Gabor frame with a short duration window and $\mathbf{\Phi}_2$ to be a Gabor frame with a long duration window. These frames are well suited for the analysis of transient and tonal structures, respectively.

Now, if we solve the system $\mathbf{y} = \mathbf{\Phi x} = [\mathbf{\Phi}_1 \mathbf{\Phi}_2][\mathbf{x}_1 \mathbf{x}_2]^T$ with a sparsity constraint, it follows that the transient components will be encapsulated by $\mathbf{x}_1$ and tonal components will be encapsulated by $\mathbf{x}_2$[11].

For example, fig. 8 shows the multilayer analysis of a 5s long glockenspiel excerpt, which was chosen because it has rather distinct tonal and transient parts. The analysis was performed with a 2-scale Gabor dictionary with Hann windows of length 2048 and 32 samples with 50% overlap. The sampling rate used was 44.1kHz. The particular analysis shown was performed using IS, however, all of the algorithms discussed lead to fairly similar results. The interested reader can listen to the multilayer expansions found using MP, IS, and IRLS on the companion website [69].

10.2. Denoising

As shown in the example presented in §9.2, prior knowledge of sparsity or compressibility is often useful for signal denoising. Further, as discussed in §5.4, regularization with a sparse prior can be interpreted as a MAP estimate in certain situations. An additional denoising example using the glockenspiel excerpt from the previous section can be found on the companion website [69].

10.3. Time-Frequency Modification

In this paper we have primarily focused on the use of Gabor frames and sparsity of the synthesis coefficients. This point-of-view is useful for time-frequency modifications, since the synthesis coefficients of a Gabor frame can be used to control the time-frequency content of the signal. For example, on our companion website [69] we include an example of a major-to-minor transposition of an acoustic guitar chord. This effect was achieved using the following steps:

1. A tonal + transient expansion was performed as described in §10.1.
2. The tonal atoms were then classified based on whether or not they belonged to the major third note in the chord (this requires an multiple f_0 estimation).
3. The major third atoms were then synthesized and flattened by 100 cents to produce a minor third note.
4. The original signal was then re-synthesized without the major third atoms and added to the minor third signal to produce a minor chord.

[11] Provided $\mathbf{\Phi}_1$ and $\mathbf{\Phi}_2$ are incoherent with one another.

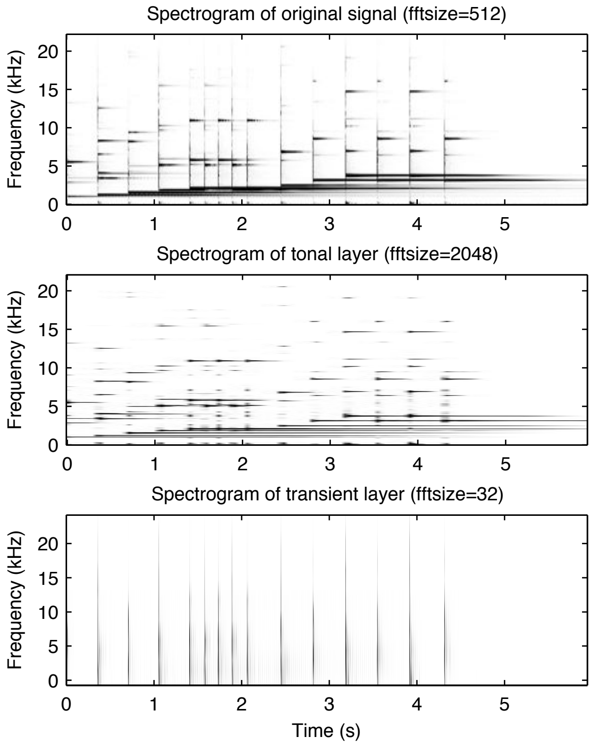

Figure 8: Multilayered expansion of a glockenspiel excerpt. Top: original spectrogram. Middle: tonal layer. Bottom: transient layer. In this example the total SRR was 32 dB and 2% of the synthesis coefficients were non-zero.

As can be heard on the website [69] the result is relatively convincing despite the naïvety of this approach. We also provide a similar example of time-stretching with transient preservation on the website [69].

We can also selectively process atoms by type. For example short duration atoms could be attenuated or amplified to smooth or accentuate an attack.

10.4. Granular Synthesis

As discussed in [70] the sparse synthesis model can be used to achieve many standard granular effects. For example, we can:

1. Apply time and frequency jitter to the atom parameters.
2. Change the atom envelopes.
3. Change the atom durations (*bleed*).
4. Change the density of atoms per unit time.

We provide several audio examples of these effects on the companion website [69].

10.5. Inverse-problems

In some cases we may not be able to directly observe the true signal. For example, we may be forced to work with limited data due to hardware requirements or assumptions regarding stationarity. In other cases we may only have access to a noisy or reverberant signal. Likewise, the signal might be downsampled, have small gaps, or be corrupted with clicks. We can often describe these types of degradations as:

$$\mathbf{z} = \Psi \mathbf{y} + \varepsilon \tag{41}$$

where $\mathbf{z}$ is the observed signal, Ψ is a (known) linear degradation operator, $\mathbf{y}$ is the true signal, and ε is additive noise. If $\mathbf{y}$ has a sparse representation $\mathbf{y} = \mathbf{\Phi x}$ then we can re-write (41) as

$$\mathbf{z} = \mathbf{Dx} + \varepsilon \tag{42}$$

where $\mathbf{D} = \Psi\mathbf{\Phi}$. Armed with the knowledge that $\mathbf{x}$ is sparse, we can attempt to estimate $\hat{\mathbf{x}}$ using the dictionary $\mathbf{D}$ and any of the techniques discussed in this paper. We can then generate an estimate of the true signal as $\hat{\mathbf{y}} = \mathbf{\Phi}\hat{\mathbf{x}}$. Under certain conditions regarding $\mathbf{D}$ and the sparsity of $\mathbf{x}$, it is possible to exactly recover $\mathbf{y}$ [71]. This premise was recently applied in [72] for audio restoration.

It has also been shown that when Ψ is a random matrix, $\mathbf{y}$ can be recovered (with high probability) if the number of rows in Ψ is large enough. This is the basis of compressed sensing (CS) [31].

11. CONCLUSION

In this paper we reviewed sparse atomic models for audio and discussed several algorithms that can be used to estimate the model parameters. This included an exploration of greedy, relaxed, and Bayesian approaches to the sparse approximation problem, as well as a brief look at structured approximations. Further, we provided a few numerical comparisons that serve to illustrate some of the practical differences between the algorithms discussed. Lastly we included a discussion of several interesting audio applications that can benefit from sparse atomic modeling. We remind the reader that many of the examples in this paper along with sound files and MATLAB®code can be found online [69].

We are currently working on MATLAB®implementations of local OMP [10], and fast IS [46], which will be added to a future release of GabLab. We also plan to implement several structured decomposition techniques and to expand upon the comparisons performed in this paper.

12. REFERENCES

[1] M. Elad, P. Milanfar, and R. Rubinstein, “Analysis versus synthesis in signal priors,” *Inverse Problems*, vol. 23, pp. 947–968, 2007.

[2] S. Nam, M. Davies, M. Elad, and R. Gribonval, “Cosparse analysis modeling-uniqueness and algorithms,” *Proc. Int. Conf. Acoust. Speech Sig. Proc.*, 2011.

[3] I. Daubechies, A. Grossmann, and Y. Meyer, “Painless nonorthogonal expansions,” *J. Math. Phys*, vol. 27, no. 5, pp. 1271–1283, 1986.

[4] J. Kovacevic and A. Chebira, “Life Beyond Bases: The Advent of Frames (Part I),” *IEEE Sig. Proc. Mag.*, vol. 24, no. 4, pp. 86–104, July 2007.

[5] M. Goodwin, “Matching pursuit with damped sinusoids,” in *Proc. Int. Conf. Acoust. Speech Signal Process*, 1997, vol. 3, pp. 2037–2040.

[6] G. Davis, S. Mallat, and Z. Zhang, “Adaptive time-frequency decompositions with matching pursuit,” in *Proc. of SPIE*, 1994, vol. 2242, pp. 402–413.

[7] J.A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[8] J.A. Tropp, "Just relax: Convex programming methods for identifying sparse signals in noise," *IEEE Trans. on Inf. Theory*, vol. 52, no. 3, pp. 1030–1051, 2006.

[9] S.G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. on Sig. Proc.*, vol. 41, no. 12, pp. 3397–3415, 1993.

[10] B. Mailhé, R. Gribonval, P. Vandergheynst, and F. Bimbot, "Fast orthogonal sparse approximation algorithms over local dictionaries," *Signal Processing*, 2011.

[11] R. Heusdens, R. Vafin, and W.B. Kleijn, "Sinusoidal modeling using psychoacoustic-adaptive matching pursuits," *Signal Processing Letters, IEEE*, vol. 9, no. 8, pp. 262–265, 2002.

[12] R. Gribonval, E. Bacry, S. Mallat, P. Depalle, and X. Rodet, "Analysis of sound signals with high resolution matching pursuit," in *Proc. of the IEEE Int. Sym. on Time-Freq. and Time-Scale Analysis*, 1996, pp. 125–128.

[13] P. Leveau and L. Daudet, "Multi-resolution partial tracking with modified matching pursuit," in *Proc. Eur. Signal Process. Conf.*, 2006.

[14] M.G. Christensen and S.H. Jensen, "The cyclic matching pursuit and its application to audio modeling and coding," in *Signals, Systems and Computers, 2007. ACSSC 2007. Conference Record of the Forty-First Asilomar Conference on*. IEEE, 2007, pp. 550–554.

[15] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," *Asilomar Conf. on Sig., Sys. and Comp.*, vol. 1, pp. 40–44, 1993.

[16] L. Rebollo-Neira and D. Lowe, "Optimized orthogonal matching pursuit approach," *Signal Processing Letters, IEEE*, vol. 9, no. 4, pp. 137–140, 2002.

[17] David L. Donoho, Yaakov Tsaig, Iddo Drori, and Jean-Luc Starck, "Sparse Solution of Underdetermined Linear Equations by Stagewise Orthogonal Matching Pursuit," Tech. Rep., Stanford, 2006.

[18] T. Blumensath and M.E. Davies, "Gradient pursuits," *IEEE Trans. on Sig. Proc.*, vol. 56, no. 6, pp. 2370–2382, 2008.

[19] M. Goodwin, *Handbook of Speech Processing*, chapter 12, pp. 229–258, Springer-Verlag, Berlin, 2008.

[20] R. Gribonval and M. Nielsen, "Approximate weak greedy algorithms," *Adv. in Comp. Math.*, vol. 14, no. 4, pp. 361–378, 2001.

[21] V.N. Temlyakov, "Weak greedy algorithms," *Adv. in Comp. Math.*, vol. 12, no. 2-3, pp. 213–227, 2000.

[22] F. Bergeaud and S. Mallat, "Matching pursuit: Adaptive representations of images and sounds," *Comp. and Applied Math.*, vol. 15, pp. 97–110, 1996.

[23] R. Gribonval and E. Bacry, "Harmonic decomposition of audio signals with matching pursuit," *IEEE Trans. on Sig. Proc.*, vol. 51, no. 1, pp. 101–111, 2003.

[24] R.R. Coifman and M.V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. on Inf. Theory*, vol. 38, no. 2, pp. 713–718, 1992.

[25] F. Jaillet and B. Torrésani, "Time-Frequency Jigsaw Puzzle: adaptive multiwindow and multilayered Gabor expansions," *Int. J. Wavelets, Multires. and Inf. Proc.*, vol. 5, no. 2, pp. 293–315, 2007.

[26] J.F. Claerbout and F. Muir, "Robust modeling with erratic data," *Geophysics*, vol. 38, pp. 826, 1973.

[27] S.S. Chen, D.L. Donoho, and M.A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[28] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *J. Royal Stat. Soc.*, pp. 267–288, 1996.

[29] S.P. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University Press, 2004.

[30] M.G. Christensen and S.H. Jensen, "On compressed sensing and its application to speech and audio signals," in *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*. IEEE, 2009, pp. 356–360.

[31] D.L. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, 2006.

[32] B.D. Rao and K. Kreutz-Delgado, "An affine scaling methodology for best basis selection," *IEEE Trans. on Sig. Proc.*, vol. 47, no. 1, pp. 187–200, 1999.

[33] I.F. Gorodnitsky and B.D. Rao, "Sparse signal reconstruction from limited data using FOCUSS: A Re-weighted minimum norm algorithm," *IEEE Trans. on Sig. Proc.*, vol. 45, no. 3, pp. 600–616, 1997.

[34] I. Daubechies, R. DeVore, M. Fornasier, and S. Gunturk, "Iteratively re-weighted least squares minimization for sparse recovery," *Commun. Pure Appl. Math*, vol. 63, no. 1, pp. 1–38, 2009.

[35] Z. He, A. Cichocki, R. Zdunek, and S. Xie, "Improved FOCUSS method with conjugate gradient iterations," *IEEE Trans. on Sig. Proc.*, vol. 57, no. 1, pp. 399–404, 2009.

[36] M.E. Davies and L. Daudet, "Sparse audio representations using the MCLT," *Signal processing*, vol. 86, no. 3, pp. 457–470, 2006.

[37] S. Sardy, A.G. Bruce, and P. Tseng, "Block coordinate relaxation methods for nonparametric signal denoising with wavelet dictionaries," *Journal of computational and graphical statistics*, vol. 9, pp. 361–379, 2000.

[38] D.L. Donoho and J.M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[39] M. Kowalski, "Sparse regression using mixed norms," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 303–324, 2009.

[40] T. Blumensath, M. Yaghoobi, and M.E. Davies, "Iterative hard thresholding and l0 regularisation," in *Proc. Int. Conf. Acoust. Speech Sig. Proc.*, 2007.

[41] M.A.T. Figueiredo and R.D. Nowak, "An EM algorithm for wavelet-based image restoration," *IEEE Trans. on Image Proc.*, vol. 12, no. 8, pp. 906–916, 2003.

[42] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math*, vol. 57, no. 11, pp. 1413–1457, 2004.

[43] M. Elad, "Why simple shrinkage is still relevant for redundant representations?," *IEEE Trans. on Inf. Theory*, vol. 52, no. 12, pp. 5559–5569, 2006.

[44] K.K. Herrity, A.C. Gilbert, and J.A. Tropp, "Sparse approximation via iterative thresholding," in *Proc. Int. Conf. Acoust. Speech Signal Proc.*, 2006, vol. 3, pp. 624–627.

[45] J.M. Bioucas-Dias and M.A.T. Figueiredo, "A new twist: two-step iterative shrinkage/thresholding algorithms for image restoration," *Image Processing, IEEE Transactions on*, vol. 16, no. 12, pp. 2992–3004, 2007.

[46] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

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

[48] R. Gribonval, "Should penalized least squares regression be interpreted as Maximum A Posteriori estimation?," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2405–2410, May 2011.

[49] M.D. Plumbley, S.A. Abdallah, T. Blumensath, and M.E. Davies, "Sparse representations of polyphonic music," *Signal Processing*, vol. 86, no. 3, pp. 417–431, 2006.

[50] K. Kreutz-Delgado, J.F. Murray, B.D. Rao, K. Engan, T.W. Lee, and T.J. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Computation*, vol. 15, no. 2, pp. 349–396, 2003.

[51] D.P. Wipf and B.D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Trans. on Sig. Proc.*, vol. 52, no. 8, pp. 2153–2164, 2004.

[52] M.E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, 2001.

[53] M.A.T. Figueiredo, "Adaptive sparseness for supervised learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 9, pp. 1150–1159, 2003.

[54] G.H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed ℓ_0 norm," *IEEE Trans. on Sig. Proc.*, vol. 57, pp. 289–301, 2009.

[55] P.J. Wolfe, S.J. Godsill, and W.J. Ng, "Bayesian variable selection and regularization for time-frequency surface estimation," *J. Royal Stat. Soc.*, vol. 66, no. 3, pp. 575–589, 2004.

[56] C. Fevotte and S.J. Godsill, "Sparse linear regression in unions of bases via Bayesian variable selection," *IEEE Sig. Proc. Let.*, vol. 13, pp. 441–444, 2006.

[57] E.I. George and R.E. McCulloch, "Approaches for Bayesian variable selection," *Statistica Sinica*, vol. 7, pp. 339–374, 1997.

[58] C. Fevotte, B. Torresani, L. Daudet, and S.J. Godsill, "Sparse linear regression with structured priors and application to denoising of musical audio," *IEEE Trans. Audio, Speech, Lang. Proc.*, vol. 16, no. 1, pp. 174–185, 2008.

[59] R. Gribonval, "Sparse decomposition of stereo signals with matching pursuit and application to blind source separation of more than two sources from a stereo mixture," in *Proc. Int. Conf. Acoust. Speech Signal Proc.*, 2002, vol. 3, pp. 3057–3060.

[60] L. Daudet, "Sparse and structured decompositions of signals with the molecular matching pursuit," *IEEE Trans. Audio, Speech, Lang. Proc.*, vol. 14, no. 5, pp. 1808–1816, 2006.

[61] X. Serra and J.O. Smith, "Spectral modeling synthesis: a sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[62] P. Depalle, G. Garcia, and X. Rodet, "Tracking of partials for additive sound synthesis using hidden Markov models," in *Proc. Int. Conf. Acoust. Speech Sig. Proc.*, 1993, vol. 1, pp. 242–245.

[63] P. Leveau, E. Vincent, G. Richard, and L. Daudet, "Instrument-specific harmonic atoms for mid-level music representation," *IEEE Transactions on Audio Speech and Language Processing*, vol. 16, no. 1, pp. 116–128, 2008.

[64] B.L. Sturm, J.J. Shynk, and S. Gauglitz, "Agglomerative clustering in sparse atomic decompositions of audio signals," in *Proc. Int. Conf. Acoust. Speech Signal Proc.*, 2008, pp. 97–100.

[65] M. Kowalski and B. Torrésani, "Sparsity and persistence: mixed norms provide simple signal models with dependent coefficients," *Sig., Image and Video Proc.*, vol. 3, no. 3, pp. 251–264, 2009.

[66] D. Donoho, "Sparselab website," 2011, `http://sparselab.stanford.edu/`.

[67] T. Blumensath, "Sparsify website," 2011, `http://users.fmrib.ox.ac.uk/~tblumens/sparsify/sparsify.html`.

[68] S. Krstulovic and R. Gribonval, "MPTK: Matching Pursuit made tractable," in *Proc. Int. Conf. Acoust. Speech Signal Proc.*, 2006, vol. 3, pp. 496–499.

[69] C. Kereliuk, "Gablab website and supplementary materials," 2011, `http://www.music.mcgill.ca/~corey/dafx2011`.

[70] B.L. Sturm, C. Roads, A. McLeran, and J.J. Shynk, "Analysis, visualization, and transformation of audio signals using dictionary-based methods," *Journal of New Music Research*, vol. 38, no. 4, pp. 325–341, 2009.

[71] S.G. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, Academic Press, Burlington, MA, 3rd edition, 2009.

[72] A. Adler, V. Emiya, G. Jafari, M. Elad, R. Gribonval, and M.D. Plumbley, "Audio Inpainting," Research report, Mar. 2011.