Online algorithms for Nonnegative Matrix Factorization with the Itakura-Saito divergence

Augustin Lefèvre, Francis Bach, Cédric Févotte

Online algorithms for Nonnegative Matrix Factorization with the Itakura-Saito divergence

Augustin Lefèvre AUGUSTIN.LEFEVRE@INRIA.FR
INRIA/ENS Sierra project
23, avenue d'Italie, 75013 Paris

Francis Bach FRANCIS.BACH@ENS.FR
INRIA/ENS Sierra project
23, avenue d'Italie, 75013 Paris

Cédric Févotte FEVOTTE@TELECOM-PARISTECH.FR
CNRS LTCI; Télécom ParisTech
37-39, rue Dareau, 75014 Paris

Abstract

Nonnegative matrix factorization (NMF) is now a common tool for audio source separation. When learning NMF on large audio databases, one major drawback is that the complexity in time is $O(FKN)$ when updating the dictionary (where (F, N) is the dimension of the input power spectrograms, and K the number of basis spectra), thus forbidding its application on signals longer than an hour. We provide an online algorithm with a complexity of $O(FK)$ in time and memory for updates in the dictionary. We show on audio simulations that the online approach is faster for short audio signals and allows to analyze audio signals of several hours.

1 Introduction

In audio source separation, nonnegative matrix factorization (NMF) is a popular technique for building a high-level representation of complex audio signals with a small number of basis spectra, forming together a dictionary (Smaragdis et al., 2007; Févotte et al., 2009; Virtanen et al., 2008). Using the Itakura-Saito divergence as a measure of fit of the dictionary to the training set allows to capture fine structure in the power spectrum of audio signals as shown in (Févotte et al., 2009).

However, estimating the dictionary can be quite slow for long audio signals, and indeed intractable for training sets of more than a few hours. We propose an algorithm to estimate Itakura-Saito NMF (IS-NMF) on audio signals of possibly infinite duration with tractable memory and time complexity. This article is organized as follows : in Section 2, we summarize Itakura-Saito NMF, propose an algorithm for online NMF, and discuss implementation details. In Section 3, we experiment our algorithms on real audio signals of short, medium and long durations. We show that our approach outperforms regular batch NMF in terms of computer time.

2 Online Dictionary Learning for the Itakura-Saito divergence

Various methods were recently proposed for online dictionary learning (Mairal et al., 2010; Hoffman et al., 2010; Bucak and Gunsel, 2009). However, to the best of our knowledge, no algorithm exists for online dictionary learning with the Itakura-Saito divergence. In this section we summarize IS-NMF, then introduce our algorithm for online NMF and explain briefly the mathematical framework.

2.1 Itakura-Saito NMF

Define the Itakura-Saito divergence as $d_{IS}(y,x) = \sum_i (\frac{y_i}{x_i} - \log \frac{y_i}{x_i} - 1)$. Given a data set $V = (v_1, \dots, v_N) \in \mathbb{R}_+^{F \times N}$, Itakura-Saito NMF consists in finding $W \in \mathbb{R}_+^{F \times K}$, $H = (h_1, \dots, h_N) \in \mathbb{R}_+^{K \times N}$ that minimize the following objective function :

$$\mathcal{L}_H(W) = \frac{1}{N} \sum_{n=1}^{N} d_{IS}(v_n, W h_n)\,, \tag{1}$$

The standard approach to solving IS-NMF is to optimize alternately in W and H and use majorization-minimization (Févotte and Idier, in press). At each step, the objective function is replaced by an auxiliary function of the form $\mathcal{L}_H(W, \underline{W})$ such that $\mathcal{L}_H(W) \leq \mathcal{L}_H(W, \underline{W})$ with equality if $W = \underline{W}$:

$$\mathcal{L}_H(W, \underline{W}) = \sum_{fk} A_{fk} \frac{1}{W_{fk}} + B_{fk} W_{fk} + c\,. \tag{2}$$

where $A, B \in \mathbb{R}_+^{F \times K}$ and $c \in \mathbb{R}$ are given by:

$$\begin{aligned} A_{fk} &= \textstyle\sum_{n=1}^{N} H_{kn} V_{fn} (\underline{W} H)_{fn}^{-2} \underline{W}_{fk}^2\,, \\ B_{fk} &= \textstyle\sum_{n=1}^{N} H_{kn} (\underline{W} H)_{fn}^{-1}\,, \\ c &= \textstyle\sum_{f=1}^{F} \sum_{n=1}^{N} \log \frac{V_{fn}}{(\underline{W} H)_{fn}} - F\,. \end{aligned} \tag{3}$$

Thus, updating W by $W_{fk} = \sqrt{A_{fk}/B_{fk}}$ yields a descent algorithm. Similar updates can be found for h_n so that the whole process defines a descent algorithm in (W, H) (for more details see, e.g., (Févotte and Idier, in press)). In a nutshell, batch IS-NMF works in cycles: at each cycle, all sample points are visited, the whole matrix H is updated, the auxiliary function in Eq. (2) is re-computed, and W is then updated. We now turn to the description of online NMF.

2.2 An online algorithm for online NMF

When N is large, multiplicative updates algorithms for IS-NMF become expensive because at the dictionary update step, they involve large matrix multiplications with time complexity in $O(FKN)$ (computation of matrices A and B). We present here an online version of the classical

multiplicative updates algorithm, in the sense that only a subset of the training data is used at each step of the algorithm.

Suppose that at each iteration of the algorithm we are provided a new data point v_t, and we are able to find h_t that minimizes $d_{IS}(v_t, W^{(t)}h_t)$. Let us rewrite the updates in Eq. (3). Initialize $A^{(0)}, B^{(0)}, W^{(0)}$ and at each step compute :

$$\begin{aligned} A^{(t)} &= A^{(t-1)} + (\tfrac{v_t}{(W^{(t-1)}h_t)^2} h_t^\top) \cdot (W^{(t-1)})^2 \,, \\ B^{(t)} &= B^{(t-1)} + \tfrac{1}{W^{(t-1)}h_t} h_t^\top \,, \\ W^{(t)} &= \sqrt{\tfrac{A^{(t)}}{B^{(t)}}} \,. \end{aligned} \tag{4}$$

Now we may update W each time a new data point v_t is visited, instead of visiting the whole data set. This differs from batch NMF in the following sense : suppose we replace the objective function in Eq. (1) by

$$L_T(W) = \frac{1}{T} \sum_{t=1}^{T} d_{IS}(v_t, Wh_t) \,, \tag{5}$$

where $(v_1, v_2, \dots, v_t, \dots)$ is an infinite sequence of data points, and the sequence $(h_1, \dots, h_t, \dots)$ is such that h_t minimizes $d_{IS}(v_t, W^{(t)}h)$. Then we may show that the modified sequence of updates corresponds to minimizing the following auxiliary function :

$$\hat{L}_T(W) = \sum_k \sum_f \left(A_{fk}^{(T)} \frac{1}{W_{fk}} + B_{fk}^{(T)} W_{fk} \right) + c \,. \tag{6}$$

If T is fixed, this problem is exactly equivalent to IS-NMF on a finite training set. Whereas in the batch algorithm described in Section 2.1, all H is updated once and then all W, in online NMF, each new h_t is estimated exactly and then W is updated once. Another way to see it is that in standard NMF, the auxiliary function is updated at each pass through the whole dataset from the most recent updates in H, whereas in online NMF, the auxiliary function takes into account all updates starting from the first one.

Extensions Prior information on H or W is often useful for imposing structure in the factorization (Lefèvre et al., 2011; Virtanen, 2007; Smaragdis et al., 2007). Our framework for online NMF easily accomodates penalties such as :

- Penalties depending on the dictionary W only.
- Penalties on H that are decomposable and expressed in terms of a concave increasing function ψ (Lefèvre et al., 2011): $\Psi(H) = \sum_{n=1}^{N} \psi(\sum_k H_{kn})$.

2.3 Practical online NMF

We provided a description of a pure version of online NMF, we now discuss several extensions that are commonly used in online algorithms and allow for considerable gains in speed.

Algorithm 1 Online Algorithm for IS-NMF

Input training set, $W^{(0)}$, $A^{(0)}$, $B^{(0)}$, ρ, β, η, ε.
$t \leftarrow 0$
repeat
  $t \leftarrow t+1$
  draw v_t from the training set.
  $h_t \leftarrow \arg\min_h d_{IS}(\varepsilon + v_t, \varepsilon + Wh)$
  $a^{(t)} \leftarrow (\frac{\varepsilon+v_t}{(\varepsilon+Wh_t)^2} h_t^\top) \cdot W^2$
  $b^{(t)} \leftarrow \frac{1}{\varepsilon+Wh_t} h_t^\top$
  if $t \equiv 0\ [\beta]$
    $A^{(t)} \leftarrow A^{(t-\beta)} + \rho \sum_{s=t-\beta+1}^{t} a^{(s)}$
    $B^{(t)} \leftarrow B^{(t-\beta)} + \rho \sum_{s=t-\beta+1}^{t} b^{(s)}$
    $W^{(t)} \leftarrow \sqrt{\frac{A^{(t)}}{B^{(t)}}}$
    for $k = 1 \dots K$
      $s \leftarrow \sum_f W_{fk}\,, \quad W_{fk} \leftarrow W_{fk}/s$
      $A_{fk} \leftarrow A_{fk}/s\,, \quad B_{fk} \leftarrow B_{fk} \times s$
    end for
  end if
until $\|W^{(t)} - W^{(t-1)}\|_F < \eta$

Finite data sets. When working on finite training sets, we cycle over the training set several times, and randomly permute the samples at each cycle.

Sampling method for infinite data sets. When dealing with large (or infinite) training sets, samples may be drawn in batches and then permuted at random to avoid local correlations of the input.

Fresh or warm restarts. Minimizing $d_{IS}(v_t, Wh_t)$ is an inner loop in our algorithm. Finding an exact solution h_t for each new sample may be costly (a rule of thumb is 100 iterations from a random point). A shortcut is to stop the inner loop before convergence. This amounts to compute only an upper-bound of $d_{IS}(v_t, Wh_t)$. Another shortcut is to warm restart the inner loop, at the cost of keeping all the most recent regression weights $H = (h_1, \dots, h_N)$ in memory. For small data sets, this allows to run online NMF very similarly to batch NMF : each time a sample is visited h_t is updated only once, and then W is updated. When using warm restarts, the time complexity of the algorithm is not changed, but the memory requirements become $O((F+N)K)$.

Mini-batch. Updating W every time a sample is drawn costs $O(FK)$: as shown in simulations, we may save some time by updating W only every β samples i.e., draw samples in batches and then update W. This is also meant to stabilize the updates.

Scaling past data. In order to speed up the online algorithm it is possible to scale past information so that newer information is given more importance :

$$\begin{aligned} A^{(t+\beta)} &= A^{(t)} + \rho \sum_{s=t+1}^{t+\beta} a^{(s)} \,, \\ B^{(t+\beta)} &= B(t) + \rho \sum_{s=t+1}^{t+\beta} b^{(s)} \,, \end{aligned} \tag{7}$$

where we choose $\rho = r^{\beta/N}$. We choose this particular form so that when $N \to +\infty$, $\rho = 1$. Moreover, ρ is taken to the power β so that we can compare performance for several batch sizes and the same parameter r. In principle this rescaling of past information amounts to discount each new sample at rate ρ, thus replacing the objective function in Eq. (5) by :

$$\frac{1}{\sum_{t=1}^{T} r^t} \sum_{t=1}^{T} r^{T+1-t} l(v_t, W) \,, \tag{8}$$

Rescaling W. In order to avoid the scaling ambiguity, each time W is updated, we rescale $W^{(t)}$ so that its columns have unit norm. $A^{(t)}$, $B^{(t)}$ must be rescaled accordingly (as well as H when using warm restarts). This does not change the result and avoids numerical instabilities when computing the product WH.

Dealing with small amplitude values. The Itakura-Saito divergence $d_{IS}(y, x)$ is badly behaved when either $y = 0$ or $x = 0$. As a remedy we replace it in our algorithm by $d_{IS}(\varepsilon+y, \varepsilon+x)$. The updates were modified consequently in Algorithm 1.

Overview. Algorithm 1 summarizes our procedure. The two parameters of interest are the mini-batch size β and the forgetting factor r. Note that when $\beta = N$, and $r = 0$, the online algorithm is equivalent to the batch algorithm.

3 Experimental study

In this section we validate the online algorithm and compare it with its batch counterpart. A natural criterion is to train both on the same data with the same initial parameters $W^{(0)}$ (and $H^{(0)}$ when applicable) and compare their respective fit to a held-out test set, as a function of the computer time available for learning. The input data are power spectrogram extracted from single-channel audio tracks, with analysis windows of 512 samples and 256 samples overlap. All silent frames were discarded.

We make the comparison for small, medium, and large audio tracks (resp. 10^3, $10^4, 10^5$ time windows). W is initialized with random samples from the train set. For each process, several seeds were tried, the best seed (in terms of objective function value) is shown for each process. Finally, we use $\varepsilon = 10^{-12}$ which is well below the hearing threshold.

Small data set (30 seconds). We ran online NMF with warm restarts and one update of h every sample. From Figure 1, we can see that there is a restriction on the values of (β, r) that we can use : if $r < 1$ then β should be chosen larger than 1. On the other hand, as long as $r > 0.5$, the stability of the algorithm is not affected by the value of β. In terms of speed, clearly setting

$r < 1$ is crucial for the online algorithm to compete with its batch counterpart. Then there is a tradeoff to make in β : it should picked larger than 1 to avoid instabilities, and smaller than the size of the train set for faster learning (this was also shown in (Mairal et al., 2010) for the square loss).

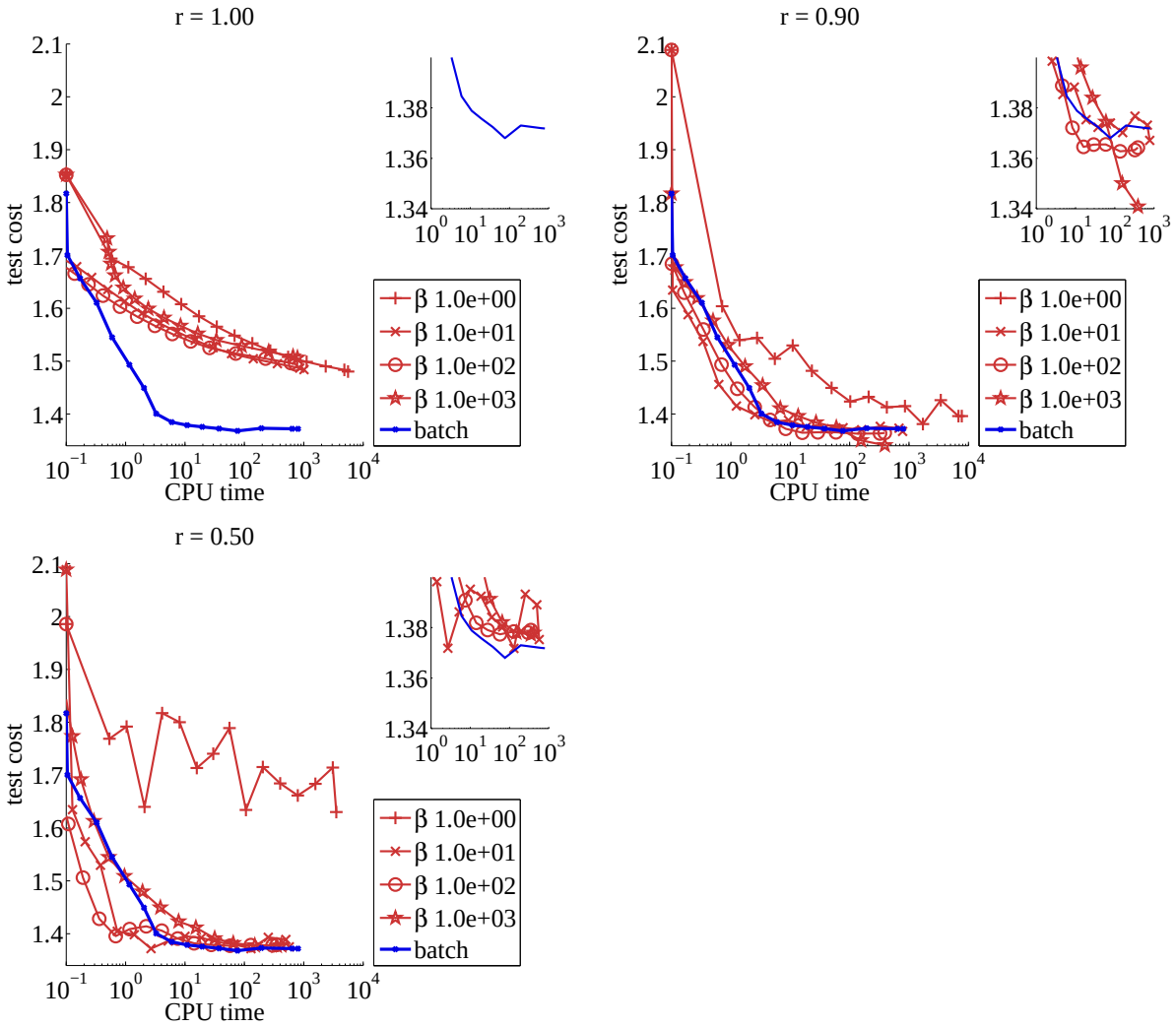

Figure 1: Comparison of online and batch algorithm on a thirty-seconds long audio track.

Medium data set (4 minutes). We ran online NMF with warm restarts and one update of h every sample. The same remarks apply as before, moreover we can see on Figure 2 that the online algorithm outperforms its batch counterpart by several orders of magnitude in terms of computer time for a wide range of parameter values.

Large data set (1 hour 20 minutes). For the large data set, we use fresh restarts and 100 updates of h for every sample. Since batch NMF does not fit into memory any more, we compare

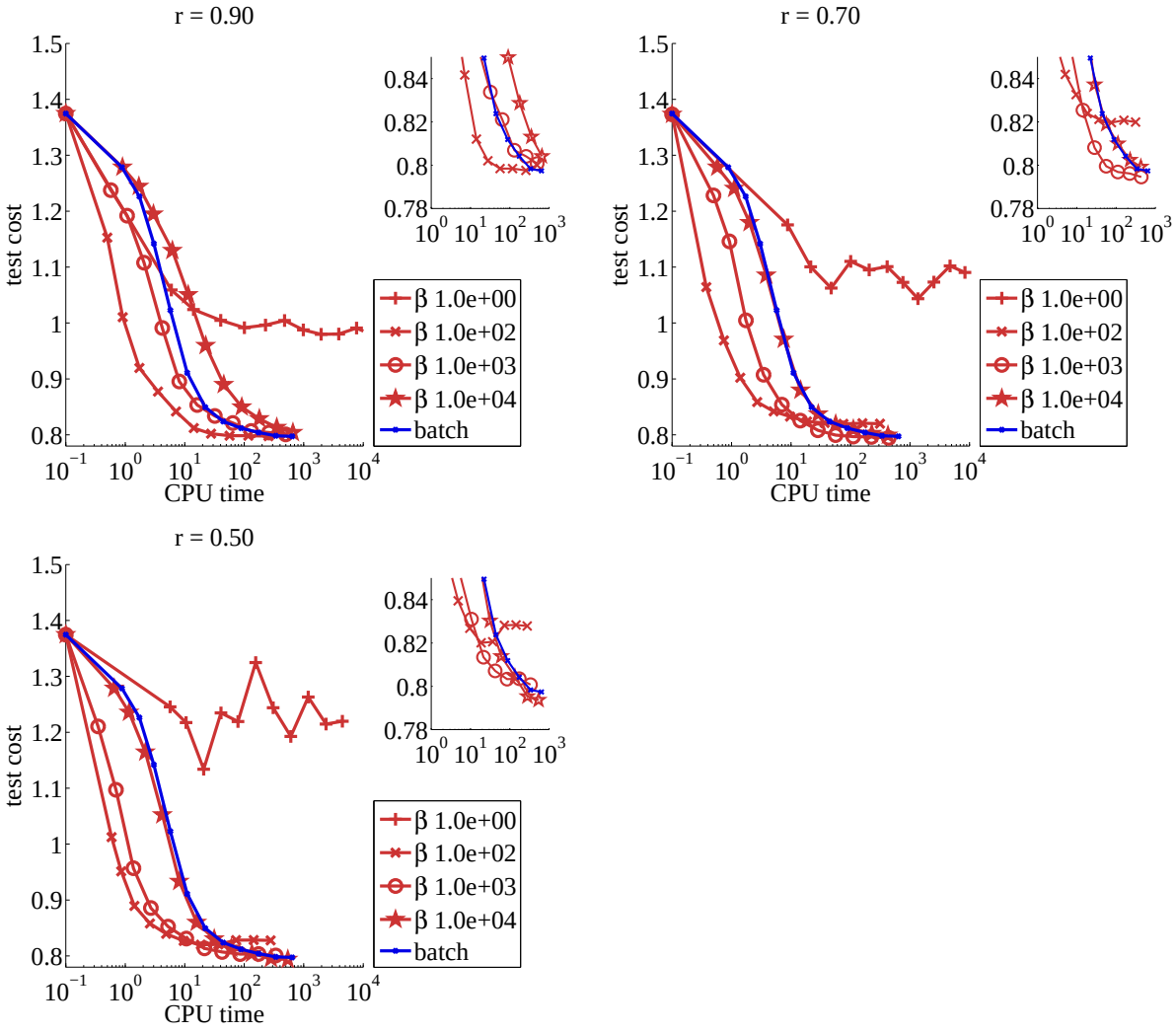

Figure 2: Comparison of online and batch algorithm on a three-minutes long audio track.

online NMF with batch NMF learnt on a subset of the training set. In Figure 3, we see that running online NMF on the whole training set yields a more accurate dictionary in a fraction of the time that batch NMF takes to run on a subset of the training set. We stress the fact that we used fresh restarts so that there is no need to store H offline.

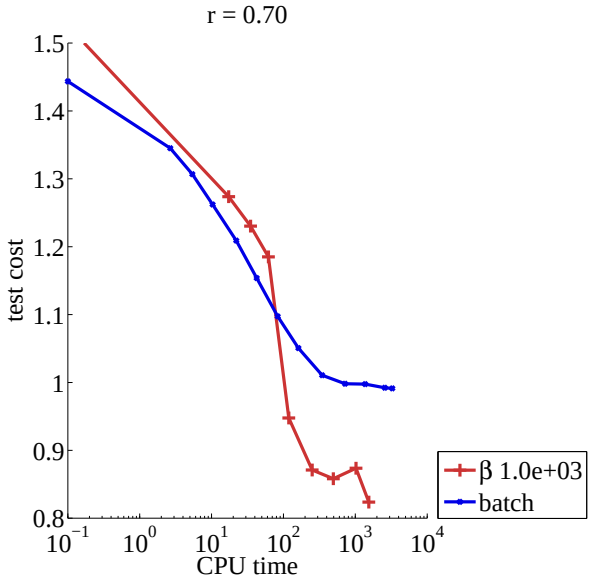

Figure 3: Comparison of online and batch algorithm on an album of Django Reinhardt (1 hour 20 minutes).

Summary. The online algorithm we proposed is stable provided minimal restrictions on the values of the parameters (r, β) : if $r = 1$, then any value of β is stable. If $r < 1$ then β should be chosen large enough. Clearly there is a tradeoff in choosing the mini-batch size β, which is explained by the way it works : when β is small, frequent updates of W are an additional cost as compared with batch NMF. On the other hand, when β is small enough we take advantage of the redundancy in the training set. From our experiments we find that choosing $r = 0.7$ and $\beta = 10^3$ yields satisfactory performance.

4 Conclusion

In this paper we make several contributions : we provide an algorithm for online IS-NMF with a complexity of $O(FK)$ in time and memory for updates in the dictionary. We propose several extensions that allow to speedup online NMF and summarize them in a concise algorithm (code will be made available soon). We show that online NMF competes with its batch counterpart on small data sets, while on large data sets it outperforms it by several orders of magnitude. In a pure online setting, data samples are processed only once, with constant time and memory cost. Thus, online NMF algorithms may be run on data sets of potentially infinite size which opens up many possibilities for audio source separation.

References

S. Bucak and B. Gunsel. Incremental subspace learning via non-negative matrix factorization. *Pattern Recognition*, 42(5):788–797, 2009.

C. Févotte and J. Idier. Algorithms for nonnegative matrix factorization with the beta-divergence. *Neural Computation*, in press.

C. Févotte, N. Bertin, and J.-L. Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis. *Neural Comput.*, 21(3):793–830, 2009.

Matthew D. Hoffman, David M. Blei, and Francis Bach. Online learning for latent dirichlet allocation. In *Neural Information Processing Systems*, 2010.

A. Lefèvre, F. Bach, and C. Févotte. Itakura-Saito nonnegative matrix factorization with group sparsity. *In Proc. Int. Conf. on Acous., Speech, and Sig. Proc. (ICASSP)*, 2011.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

P. Smaragdis, B. Raj, and M.V. Shashanka. Supervised and semi-supervised separation of sounds from single-channel mixtures. *In Proc. Int. Conf. on ICA and Signal Separation. London, UK*, September 2007.

T. O. Virtanen, A. T. Cemgil, and S. J. Godsill. Bayesian extensions to nonnegative matrix factorisation for audio signal modelling. In *Proc. of IEEE ICASSP*, Las Vegas, 2008. IEEE.

T.O. Virtanen. Monaural sound source separation by non-negative matrix factorization with temporal continuity and sparseness criteria. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 15(3), March 2007.