NON-NEGATIVE MATRIX FACTORIZATION FOR IRREGULARLY-SPACED TRANSFORMS

Paris Smaragdis

University of Illinois at Urbana-Champaign,
Adobe Systems, Inc.

Minje Kim

Department of Computer Science
University of Illinois at Urbana-Champaign

ABSTRACT

Non-negative factorizations of spectra have been a very popular tool for various audio tasks recently. A long-standing problem with these methods methods is that they cannot be easily applied on other kinds of spectral decompositions such as sinusoidal models, constant-Q transforms, wavelets and reassigned spectra. This is because with these transforms the frequency and/or time values are real-valued and not sampled on a regular grid. We therefore cannot represent them as a matrix that we can later factorize. In this paper we present a formulation of non-negative matrix factorization that can be applied on data with real-valued indices, thereby making the application of this family of methods feasible on a broader family of time/frequency transforms.

Index Terms— Non-negative Matrix Factorization, Reassignment Method

1. INTRODUCTION

Latent component models on non-negative data have for a while been a very active area of research and have found numerous applications in a wide range of domains, from text analysis [1][2] and recommendation systems [3] to visual scene analysis [4] and music transcription [5]. Because many of these techniques trace their origins back to matrix decompositions, there is often the underlying assumption that the dimension axes of the input data are indexed using integers. Such an integer index is usually used to identify a word, a document, a pixel location, a Fourier frequency bin, etc., all of these quantities being discrete and countable. In other words, the inputs are designed so that they can be represented by a regular grid, most often represented by a matrix. Although this is a natural representation for many problems, e.g. a TF-IDF matrix, a spectrogram, or a digitized image, it is not a very flexible format for many continuous signal representations where the sampling or the representation can be irregular and/or parametric.

In this paper we examine an approach that can analyze such inputs while maintaining the structure of typical latent variable models, i.e. Non-negative Matrix Factorization (NMF). In particular we will focus on representations which are parametric, that is for each available data point we will have a real-valued number denoting its index in every dimension. We will be constraining our analysis to two-dimensional data, thereby directly extending techniques that operate on matrices (or two-dimensional distributions), but it is simple to extend this approach to arbitrary dimensions, any of which can be either discrete or real-valued.

In the remainder of this paper we will introduce the basic model, a model that corresponds to NMF [6, 7]. We will show how to estimate such a model's parameters using data with real-valued dimensions and we will discuss the extra complications and options that arise. We will apply this technique to the analysis of time series and we will show that using such parametric-data approaches we can discover signal structure that would be otherwise invisible to traditional latent variable approaches.

2. NMF FOR IRREGULARLY-SAMPLED DATA

2.1. Non-negative matrix factorization

A regular factorization of a time/frequency matrix is defined as:

$$\mathbf{X} \approx \mathbf{W} \cdot \mathbf{H} \tag{1}$$

where $\mathbf{X} \in \mathbb{R}_+^{M \times N}$ is a matrix containing time/frequency energies, and $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_Z] \in \mathbb{R}_+^{M \times Z}$ and $\mathbf{H} = [\mathbf{h}_1^\top, \mathbf{h}_2^\top, \cdots, \mathbf{h}_Z^\top]^\top \in \mathbb{R}_+^{Z \times N}$ represent Z frequency and time factors, respectively. NMF is a simple and useful factorization that estimates the two factors using the following iterative process:

$$\begin{aligned} \mathbf{P}_z &= \frac{\mathbf{w}_z \cdot \mathbf{h}_z}{\mathbf{W} \cdot \mathbf{H}}, \\ \mathbf{w}_z &= (\mathbf{X} \odot \mathbf{P}_z) \cdot \mathbf{1}^{N \times 1}, \\ \mathbf{h}_z &= \mathbf{1}^{1 \times M} \cdot (\mathbf{X} \odot \mathbf{P}_z), \end{aligned} \tag{2}$$

where $\mathbf{1}^{m \times n}$ is an $m \times n$ matrix of ones, $\odot$ and $\frac{[\cdots]}{[\cdots]}$ stand for element-wise multiplication and division, respectively. We normalize $\mathbf{w}_z$ by the sum of $\mathbf{h}_z$ at the end of every iteration in order to get a spectrum estimate that is unbiased by how much it appears over time. This also sets the magnitude of $\mathbf{W}$ so that we do not have multiple solutions that transfer energy between the two factors.

The downside of this formulation is that the frequency and time axes need to be sampled uniformly, meaning that at each time point we need to have an energy reading for all the frequency values, and vice versa. Unfortunately for certain types of time/frequency transforms, such as constant-Q transforms, wavelets and reassigned spectrograms, this assumptions do not hold and the resulting time/frequency energies cannot be represented using a finite-sized matrix. For such representations we use a different format attaching to each energy value its exact frequency and time location. In order to factorize such transforms we need to redefine the factorization process to accept this new format.

2.2. Reformulation of NMF into a vectorized form

In this section we assume that the transforms that we use are regularly sampled as above, but we will use a different representation to allow us to extend this formulation to non-regularly sampled transforms later. Instead of using a matrix $\mathbf{X}$ to represent the time/frequency energies we will use three vectors, $\mathbf{f} \in \mathbb{Z}^{MN \times 1}$, $\mathbf{t} \in \mathbb{Z}^{MN \times 1}$, and $\text{vec}(\mathbf{X}) = \mathbf{x} \in \mathbb{R}_+^{MN \times 1}$, which will respectively hold the frequency coordinate, the time coordinate, and the energy

value of each time/frequency point[1]. The elements of those vectors, $\mathbf{f}(i)$, $\mathbf{t}(i)$, and $\mathbf{x}(i)$, are indexed by $i = \{1, 2, \cdots, MN\}$.

Using the newly introduced formulation we can rewrite the factorization process as follows:

$$\mathbf{x} = \sum_{z=1}^{Z} \mathbf{v}_z \odot \mathbf{g}_z, \tag{3}$$

where now the pair of vectors $\mathbf{v}_z \in \mathbb{R}_+^{MN\times 1}$ and $\mathbf{g}_z \in \mathbb{R}_+^{MN\times 1}$ correspond to the values of the factors $\mathbf{W}$ and $\mathbf{H}$ as they are evaluated at the frequencies and times denoted by $\mathbf{f}$ and $\mathbf{t}$. With this, the iterative multiplicative update rules turn into the following form:

$$\begin{aligned}
\mathbf{p}_z &= \frac{\mathbf{v}_z \odot \mathbf{g}_z}{\sum_{z'=1}^{Z} \mathbf{v}_{z'} \odot \mathbf{g}_{z'}} \\
\mathbf{v}_z(i) &= \sum_{\forall j : \boldsymbol{f}(j) = \boldsymbol{f}(i)} \mathbf{x}(j)\mathbf{p}_z(j) \\
\mathbf{g}_z(i) &= \sum_{\forall j : \boldsymbol{t}(j) = \boldsymbol{t}(i)} \mathbf{x}(j)\mathbf{p}_z(j)
\end{aligned} \tag{4}$$

It is easy to show that if the frequency/time indices lie on a regular integer grids, i.e. $\mathbf{f}(i) \in \{1, 2, \cdots, M\}$ and $\mathbf{t}(i) \in \{1, 2, \cdots, N\}$, respectively, we will be performing the same operations as in (2). We can furthermore rewrite (4) to process all components simultaneously as:

$$\begin{aligned}
\mathbf{P} &= \frac{\mathbf{V} \odot \mathbf{G}}{(\mathbf{V} \odot \mathbf{G}) \cdot \mathbf{1}^{K\times K}} \\
\mathbf{V} &= \mathbf{D}_f \cdot (\mathbf{P} \odot \mathbf{X}) \\
\mathbf{G} &= \mathbf{D}_t \cdot (\mathbf{P} \odot \mathbf{X})
\end{aligned} \tag{5}$$

where the matrices, $\mathbf{P}$, $\mathbf{V}$, and $\mathbf{G}$, contain Z concatenated column vectors, each of which is for a latent variable z, e.g. $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_Z]$. Additionally, $\mathbf{D}_f, \mathbf{D}_t \in \{0, 1\}^{MN\times MN}$ denote two matrices defined as:

$$\begin{aligned}
\mathbf{D}_f(i,j) &= \begin{cases} 1, & \mathbf{f}(i) = \mathbf{f}(j) \\ 0, & \mathbf{f}(i) \neq \mathbf{f}(j) \end{cases} \\
\mathbf{D}_t(i,j) &= \begin{cases} 1, & \mathbf{t}(i) = \mathbf{t}(j) \\ 0, & \mathbf{t}(i) \neq \mathbf{t}(j) \end{cases}
\end{aligned} \tag{6}$$

Multiplying with these matrices results in summing over all the elements that have the same frequency or time value respectively. The only difference between the formulation in this section and in (2) is that we will obtain the two factors in a different format so that:

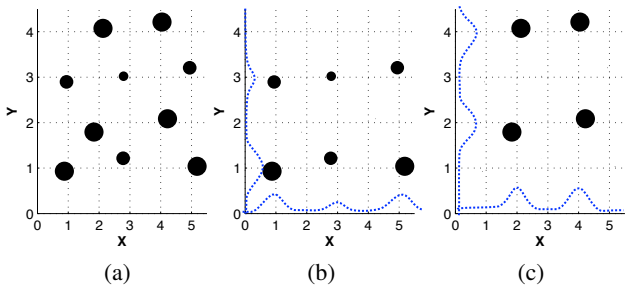
$$\begin{aligned}
\mathbf{w}_z(m) &= \mathbf{v}_z(i), \quad \forall i : \mathbf{f}(i) = m \\
\mathbf{h}_z(n) &= \mathbf{g}_z(i), \quad \forall i : \mathbf{t}(i) = n \\
\mathrm{vec}(\mathbf{w}_z \cdot \mathbf{h}_z) &= \mathbf{v}_z \odot \mathbf{g}_z
\end{aligned} \tag{7}$$

where m and n are uniform indices defined in the ranges, $\{1, 2, \cdots, M\}$ and $\{1, 2, \cdots, N\}$, respectively.

2.3. Non-negative non-regular matrix factorization

The more interesting case is the one where the frequency and time vectors are real-valued and potentially comprised of unique elements. In this case the summations in (4) become meaningless since they will only sum over single points and will never capture the correlations that form as multiple frequencies get excited at roughly the same time.

[1]The vec$(\cdot)$ operator concatenates all the columns of its input matrix to a single column vector.



(a) (b) (c)

Figure 1: Example of a real-valued-index data set. In (a) we see a set of data that is not sampled on a grid, as is evident by the unaligned positioning of the data points. The size of the points indicates the magnitude of their assigned value $\mathbf{x}(i)$. In (b) and (c) we see two of the implied components that make up the data in (a), and their smoothed projections on both axes.

To illustrate such a case let us consider the simple example as shown in Figure 1 (a), where we have $\mathbf{f} \in \mathbb{R}^{MN\times 1}$ and $\mathbf{t} \in \mathbb{R}^{MN\times 1}$, i.e. real-valued frequency/time indices. In this case we need to slightly amend the learning procedure. Previously we used co-activation information to update the learned components. So, for example, if for two points $\mathbf{x}(i)$ and $\mathbf{x}(j)$ we had that $\mathbf{f}(i) = \mathbf{f}(j) = m$ and subsequently $\mathbf{D}_f(i,j) = 1$, we would perform a sum over them when we estimated $\mathbf{v}$. In the case above since all the frequencies are real-valued and potentially unique, this summation would never happen and instead the learned factors $\mathbf{v}$ and $\mathbf{g}$ would be uninformative. In order to alleviate that we redefine the two summing matrices such that $\mathbf{D}_f, \mathbf{D}_t \in \mathbb{R}_+^{MN\times MN}$ and:

$$\mathbf{D}_f(i,j) = e^{\frac{-|\mathbf{f}(i)-\mathbf{f}(j)|^2}{\sigma_f^2}}, \quad \mathbf{D}_t(i,j) = e^{\frac{-|\mathbf{t}(i)-\mathbf{t}(j)|^2}{\sigma_t^2}} \tag{8}$$

This means that we still maintain that $\mathbf{D}_f(i,j) = 1, \forall i,j : \mathbf{f}(i) = \mathbf{f}(j)$ and $\mathbf{D}_t(i,j) = 1, \forall i,j : \mathbf{t}(i) = \mathbf{t}(j)$, but if we have the case where two frequency or time labels are close but not exactly the same we would still sum them, albeit using a lower weight. For distant points the corresponding values in these matrices will be very close zero, so no significant summation would take place.

Using this proposed approach, we obtain the results in Figure 1 (b) and (c). The discovered factorization successfully decomposes the non-uniformly spaced input samples into two intuitively correct latent components. This kind of input cannot be represented using matrix forms as the data indices are not integer-valued. Therefore, it is impossible to otherwise resolve this problem with any latent variable methods such as Probabilistic Latent Semantic Indexing (PLSI) [1], PLCA [8], Latent Dirichlet Allocation (LDA) [2], or even matrix factorization methods such as Non-negative Matrix Factorization (NMF) [6][7] and the Singular Value Decomposition (SVD).

3. EXPERIMENTAL RESULTS

This section highlights the benefits of the proposed model by using some audio examples with parametric representations that are not amenable to analysis using matrix-based methods.

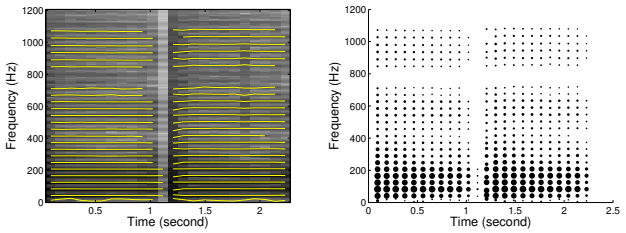


Figure 2: Sinusoidal tracking example. (a) Zoomed-in STFT of a musical sound and estimated sinusoid components (yellow lines). (b) the frequency and intensity of sinusoids are represented with dots, where the size of the dot represents the intensity. Note that the frequency position of the dots is real-valued, but the time is sampled on a regular grid therefore is integer-valued.

3.1. An example with non-regular input along one-dimension

Suppose that we observe a non-regular input stream with a regular time interval. For instance, Figure 2 (a) is the result of a sinusoidal component estimation at every time frame of a series of short-time Fourier transforms (STFT) of a sound. We cannot represent that data using a matrix representation since each sinusoid is positioned on the vertical axis using a real-valued frequency estimate that will not necessarily line up with the integer-valued Fourier bins. In addition to that we have a different number of the sinusoids at different time frames which also makes it hard to force them into a matrix representation.

The sound that is being analyzed consists of two successive bass guitar notes at a low frequency range (around 41Hz), with a very small frequency difference between them (about 0.17 of a semitone). As is well known in the area of music analysis, if we decompose the STFT data of such a sound using an algorithm like PLSI, PLCA or NMF and request two components, we should see that each component will correspond to one of the notes played [9]. As we will see however, this particular sound is problematic with known techniques. Because of the low frequencies involved, we have to use a large Fourier analysis window (8192pt = 0.186 sec in this case) to obtain a high frequency resolution so that the two notes do not have an identical looking representation. Using a hop size of 50% and a Hann window we applied NMF with two components on the magnitude STFT of this sound and we decomposed it to two elements as shown in Figure 3. Upon examination we see that both components average out similar characteristics from both notes and fail to properly segment the input. This is because even with such a long analysis window the magnitude spectra of the two notes are not sufficiently different to be recognized as two components.

We now repeat this experiment, but instead of using the magnitude STFT data we use the sinusoidal analysis data from Figure 2 (b), which is real-valued on the frequency axis. This will provide the extra frequency resolution we need, but will necessitate that we use the proposed algorithm to deal with the non-regular nature of our data. The decomposition results for two components are shown in Figure 3 (c) and (d), where bigger dots indicate more energy. We can see that this algorithm provides the desired decomposition, with each note being a discovered component. We note here that the better results are not a side effect of the algorithm, but rather of a better data representation that suits this problem. This algorithm only becomes necessary because this representation is not analyzable by other known methods.

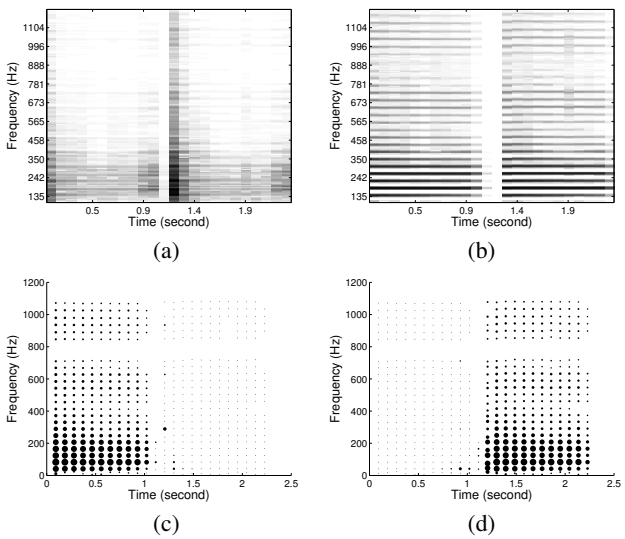


Figure 3: Separation results of regular and non-regular NMF. (a) First component estimate from regular NMF. (b) Second component estimate from regular NMF. (c) First component using non-regular NMF. (d) Second component using non-regular NMF.

3.2. Reassigned spectrogram: non-regular along both axes

In this section we will show an experiment where both axes of our input are real-valued. We will do so by making use of reassigned spectrograms. The reassignment method [10] provides an alternative representation of magnitude spectrograms by estimating a more accurate position of each spectrogram value and *reassigning* that value to a new more accurate time/frequency position. It basically breaks the grid structure by nudging each time/frequency bin out of an integer-valued location. Because of that nudging, the resulting spectrogram can exhibit infinite resolution in both frequency and time domains. This results a much more accurate time/frequency representation of a time series, but the data is now in a form that is very hard to decompose using traditional techniques.

To motivate using this representation, we use the first few seconds of the recording "Donna Lee" by Jaco Pastorius, which is a fast-paced bass solo with some percussion in the background. The played notes are $\{G_3, A_3, G_3, E_3, D_3, D_3^\flat\}$ and there are two different conga hits, one simultaneously with the third note and one with the fifth. Because of the low bass notes we would require high frequency resolution to be able to tell the notes part, but the fast note successions and percussion necessitate high temporal resolution. If we analyze this data using a traditional STFT we obtain the two representations shown at the top of Figure 4. We can see that for a short enough FFT size that provides good temporal resolution, the spectra of the bass notes are virtually indistinguishable, whereas for a large enough window where the note spectra become distinct the timing information is severely smeared. For any combination of STFT parameters it is impossible to obtain an NMF-style factorization that discovers the bass notes and the percussion hits. Alternatively we can use a reassigned spectrum as shown at the bottom of in Figure 4. In that representation it is easier to see the bass notes, as well as the two percussion hits. In Figure 5 we show the reassigned spectrogram, with each point having been labeled according to which component is used to reconstruct it. As we would expect from an NMF-style analysis, the unique spectra of the different

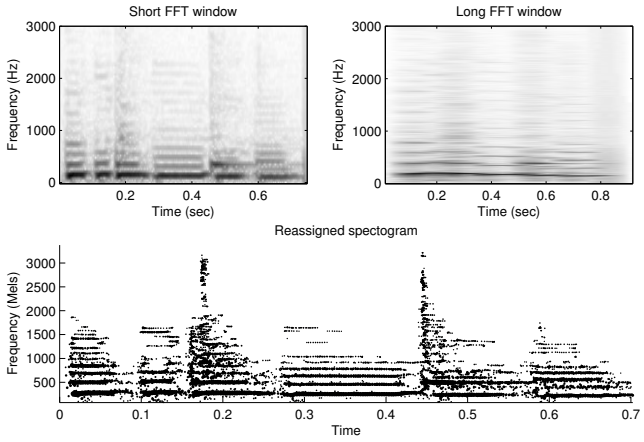


Figure 4: Comparison of a short-window STFT, a long-window STFT, and a reassigned spectrum. For the latter, the size of the points represents the amount of energy. For legibility we stretched the frequency axis to align with the Mel scale. Unlike the traditional spectrograms, this stretching is easy to do without any loss of information because of the parametric format of the reassigned spectra.

notes and the two percussion sounds should emerge as components. Although this is impossible to achieve using a standard STFT and NMF analysis due to time/frequency tradeoff constraints, using the proposed approach we successfully discover all the expected elements, despite their very close overlap in time and frequency.

3.3. Implementation Notes

There are a couple of practical issues that we address in this section regarding the use of kernels. As should be evident variance of the Gaussian kernels $\mathbf{D}_f$ and $\mathbf{D}_t$ that we use can have a dramatic effect on the results. A very small variance will not fill the space enough to learn any structure, whereas too large a variance will blur the results. In the above cases we have a clear sense of the approximate spacing between our data so that we can make a good guess of the proper values, this might not always be the case though. An additional problem is that of computational complexity. Employing the two kernel matrices can be very cumbersome when the number of data samples is in the tens of thousands. To alleviate that we clip small values of $\mathbf{D}_f$ and $\mathbf{D}_t$ to zero. By doing so we can use sparse matrix routines which accelerate computation significantly and also reduce memory footprint.

4. CONCLUSIONS

In this paper we presented a latent component model that operates on inputs that do not lie on a regular grid. We formulated this as a vectorized form of matrix decomposition problem and derived a multiplicative update rules that are analogous to those of NMF. By running experiments on audio data representations that are parametric, we have shown that this algorithm performs as expected and is able to correctly analyze such irregular inputs that gridded-data techniques are not able to.

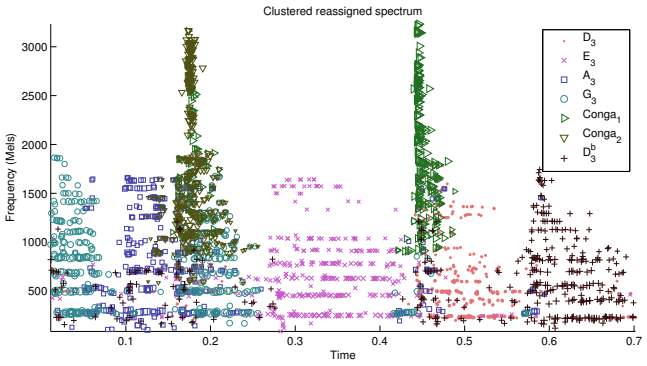


Figure 5: The reassigned spectrogram in Figure 4, with each point labelled by its component association, as denoted by both shape and color. In order to improve legibility not all input points are plotted. One can clearly see that the input is properly segmented according to the notes and the percussion hits.

5. REFERENCES

[1] T. Hofmann, "Probablistic latent semantic indexing," in *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 1999.

[2] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[3] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence, "Probabilistic models for unified collaborative and content-based recommendation in sparse-data environment," in *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI)*, 2001.

[4] L. Cao and L. Fei-Fei, "Spatially coherent latent topic model for concurrent object segmentation and classification," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2007.

[5] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural Computation*, vol. 21, no. 3, pp. 793–830, 2009.

[6] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, 1999.

[7] ——, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 13. MIT Press, 2001.

[8] P. Smaragdis, B. Raj, and M. Shashanka, "A probabilistic latent variable model for acoustic modeling," in *Neural Information Processing Systems Workshop on Advances in Models for Acoustic Processing*, 2006.

[9] P. Smaragdis and J. C. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 2003, pp. 177–180.

[10] F. Auger and P. Flandrin, "Improving the readability of time-frequency and time-scale representations by the reassignment method," *IEEE Transactions on Signal Processing*, vol. 43, no. 5, pp. 1068–1089, May 1995.