Multi-pitch Streaming of Harmonic Sound Mixtures

Zhiyao Duan, *Member, IEEE,* Jinyu Han, and Bryan Pardo, *Member, IEEE*

***Abstract*—Multi-pitch analysis of concurrent sound sources is an important but challenging problem. It requires estimating pitch values of all harmonic sources in individual frames *and* streaming the pitch estimates into trajectories, each of which corresponds to a source. We address the streaming problem for monophonic sound sources. We take the original audio, plus frame-level pitch estimates from any multi-pitch estimation algorithm as inputs, and output a pitch trajectory for each source. Our approach does not require pre-training of source models from isolated recordings. Instead, it casts the problem as a constrained clustering problem, where each cluster corresponds to a source. The clustering objective is to minimize the timbre inconsistency within each cluster. We explore different timbre features for music and speech. For music, harmonic structure and a newly proposed feature called uniform discrete cepstrum (UDC) are found effective; while for speech, MFCC and UDC works well. We also show that timbre-consistency is insufficient for effective streaming. Constraints are imposed on pairs of pitch estimates according to their time-frequency relationships. We propose a new constrained clustering algorithm that satisfies as many constraints as possible while optimizing the clustering objective. We compare the proposed approach with other state-of-the-art supervised and unsupervised multi-pitch streaming approaches that are specifically designed for music or speech. Better or comparable results are shown.**

***Index Terms*—Multi-pitch analysis, pitch streaming, timbre tracking, cochannel speech, constrained clustering.**

I. Introduction

MULTI-PITCH (fundamental frequency) analysis of harmonic sound mixtures is a fundamental problem in audio signal processing. In music information retrieval, it is of great interest to researchers working in automatic music transcription [1], source separation [2], melody extraction [3], etc. In speech processing, it is helpful for multi-talker speech recognition [4] and prosody analysis [5]. It is also a step towards solving the cocktail party problem [6].

According to MIREX[1], multi-pitch analysis can be addressed at three levels. The first (and easiest) level is to collectively estimate pitch values of all concurrent sources at each individual time frame, without determining their sources. This is known as *multi-pitch estimation (MPE)*. Most work in multi-pitch analysis performs at this level and a number of methods have been proposed. For music, time domain methods [7]–[9] and frequency domain methods [10]–[17] have been proposed. For speech, several methods [18]–[21] estimate pitches of two concurrent speakers, but no existing work addresses three or more concurrent speakers.

The second level is called *note tracking* in music information retrieval. The task is to estimate continuous segments that typically correspond to individual notes or syllables. This is often achieved by assuming the continuity and smoothness of the pitch contours, connecting pitch estimates that are close in both time and frequency. Note that each pitch contour comes from one source but each source can have many contours (e.g. one contour per musical note or spoken word). Several methods have been proposed to perform at this level, for music [22]–[25] or speech [26].

The third (and most difficult) level is to stream pitch estimates into a single pitch trajectory over an entire conversation or music performance for each of the concurrent sources. The trajectory is much longer than those estimated at the second level, and contains multiple discontinuities that are caused by silence, non-pitched sounds and abrupt frequency changes. Therefore, techniques used at the second level to connect close pitch estimates are not enough to connect short pitch segments into streams. We argue timbre information is needed to connect discontinuous pitch segments of a single sound source. We call the third level *multi-pitch streaming*[2].

In this paper, we address the third level multi-pitch streaming problem. Our approach requires three inputs: the original audio mixture, the estimated pitches at every time frame from an existing MPE algorithm, and the number of sources. Our approach assumes monophonic and harmonic sound sources and streams pitch estimates into multiple pitch trajectories, each of which corresponds to an underlying source.

We formulate this problem as a constrained clustering problem, where the clustering objective is to maintain timbre consistency and the constraints are based on the relationships between pitch estimates in time and frequency. Compared with existing methods, our approach has the following advances:

- *Unsupervised.* It does not require training source models using isolated recordings of the underlying sources.
- *General.* It can deal with both music and speech, whereas existing approaches deal with either music or speech.
- *Compatible.* It can work with any MPE algorithm.

In this work, we also introduce a new cepstrum feature that is more suitable for representing timbre in multi-source mixtures than the standard approach and a new constrained clustering algorithm to handle the issues that arise from error-prone input pitches and large numbers of constraints.

A preliminary version of the proposed approach was published in [27] for music data. The current article generalizes

Z. Duan is with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY. E-mail: zhiyao.duan@rochester.edu. B. Pardo is with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL. E-mail: pardo@cs.northwestern.edu. J. Han is with Gracenote. Email: jhan@gracenote.com.

[1]The Music Information Retrieval Evaluation eXchange (MIREX) is an annual evaluation campaign for Music Information Retrieval (MIR) algorithms. Multiple Fundamental Frequency Estimation & Tracking is one of its tasks.

[2]This is also sometimes called *multi-pitch tracking*, however multi-pitch tracking also refers to the first or second level in the literature. Therefore, we use streaming to refer the third level in this paper.

the work to speech, introduces the new cepstral representation previously mentioned, adds computational complexity analysis, has comprehensive experiments on both music and speech data and compares to several state-of-the-art methods. The sum of these things makes this article a significant advance over our preliminary work.

The rest of the paper is organized as follows. We first describe related work in automated streaming in Section II. We then formulate the problem in Section III, then describe the algorithm to solve the problem in Section IV. In Section V we describe our timbre representations. In Section VI and VII we present experiments on music and speech, respectively. Finally we conclude the paper in Section VIII.

II. Related Work in Automated Streaming

Few perform multi-pitch analysis at the streaming level. Kashino and Murase [28] proposed a Bayesian network approach to integrate musicological and timbre information to stream pitches of multiple concurrent monophonic musical instruments. However, this method requires ground-truth notes (with both pitch and time information) as inputs. It has not been tested in more realistic scenarios where the inputs are estimated pitches at the frame level.

Vincent [29] proposed a three-layer (state, source, and mixture) Bayesian network to estimate the pitches and separate the signals of musical instruments in a stereo recording. The approximate azimuths of the instruments are required as input. The parameters of the network need to be pre-learned from solo or mixture recordings of these instruments.

Bay *et al.* [30] proposed to estimate and stream pitches of polyphonic music using a probabilistic latent component analysis framework. This method also needs to pre-learn a spectral dictionary for each pitch of each instrument present in the mixture, from their isolated recordings.

Wohlmayr *et al.* [31] proposed a factorial hidden Markov model to estimate and stream pitches of two simultaneous talkers. The model parameters need to be trained for the talkers present in the mixture using their isolated recordings. These supervised methods [29]–[31] prevent their usage in scenarios when prior training on specific sources is unavailable.

Recently, Hu and Wang [32] proposed an unsupervised approach to estimate and stream pitches, and separate their signals of two simultaneous talkers. However, this approach was proposed only for speech and has not been tested for other kinds of audio data such as music.

In psychoacoustics, *sequential grouping* refers to the human auditory scene analysis process that streams auditory scene segments into meaningful auditory events [33]. Multi-pitch streaming can be viewed as a kind of sequential grouping process. A related concept is *simultaneous grouping*, which refers to grouping simultaneous time-frequency elements into meaningful auditory events. MPE can be viewed as a kind of simultaneous grouping process.

Our approach (MPE + streaming) uses a feed-forward approach to address the multi-pitch streaming problem. It does not use information from the streaming level to inform an existing MPE module used as input. This may not be optimal, since errors generated in the MPE stage may cause additional errors in the streaming stage. However, the simplicity and clarity of our modular design let us build on existing work in MPE and independently optimize different levels of the system, whereas jointly determining pitch candidates and their streams may require a very complicated model and can be computationally intractable.

An alternate way to combine sequential and simultaneous grouping is to first do sequential grouping (partial tracking) then do simultaneous grouping (grouping partials into sources). In the literature, partial tracking is addressed by assuming the value continuity [34] or the slope continuity [35] of the frequencies and amplitudes of partials. Therefore, a tracked partial would not be longer than a note or a syllable, and the "birth" and "death" of partials need to be addressed. In [36], a clustering approach based on frequency and amplitude continuity is proposed to track partials and group them into sources simultaneously, however, it still cannot group non-continuous partials since no timbre information is used.

III. Streaming as Constrained Clustering

We formulate the streaming problem as a constrained clustering problem, where the system takes three inputs: the original audio mixture, the set of instantaneous pitch estimates provided at each time frame by an existing multi-pitch estimation system, and the number of sources. The clustering objective is to maintain timbre consistency, based on the assumption that sound objects coming from the same source have similar timbre. Must-link constraints are imposed between pitches that are close in both time and frequency, to encourage them to be clustered into the same trajectory. We impose cannot-link constraints between pitches at the same time frame, to prevent them being assigned to the same source. We propose a novel algorithm to solve this constrained clustering problem.

A. Streaming Pitches by Clustering

We assume an audio mixture containing K monophonic sound sources. For each time frame we assume we have the output of a multi-pitch estimator that provides at most K concurrent pitch estimates. Some frames may contain less than K or even no pitches. We associate the ith pitch estimate with a timbre represented as an n-dimensional vector $\mathbf{t}_i$.

We view multi-pitch streaming as a pitch clustering problem, where each cluster is a pitch stream corresponding to a source. The clustering objective is to minimize the total within-stream distance of the timbres of the pitch estimates:

$$f(\Pi) = \sum_{k=1}^{K} \sum_{\mathbf{t}_i \in S_k} \|\mathbf{t}_i - \mathbf{c}_k\|^2. \tag{1}$$

Here, Π is a partition of the pitch estimates into K streams; $\mathbf{t}_i$ is the timbre feature vector of pitch i; $\mathbf{c}_k$ is the centroid of timbres in stream S_k; and $\|\cdot\|$ denotes the Euclidean norm. This is the same as the K-means clustering objective.

To justify the clustering objective function, we note that humans use timbre to discriminate and track sound sources [33]. Given an appropriate timbre feature, we expect that a note (vowel) is more similar in timbre to another note (vowel) produced by the same instrument (talker), than to a note (vowel) produced by a different instrument (talker). Different choices of timbre vectors can be found in Section V.

B. Adding Locality Constraints

The K-means algorithm can be used to minimize the clustering objective Eq. (1). However, it is not enough to provide satisfying pitch streaming results, as shown in Figure 1.

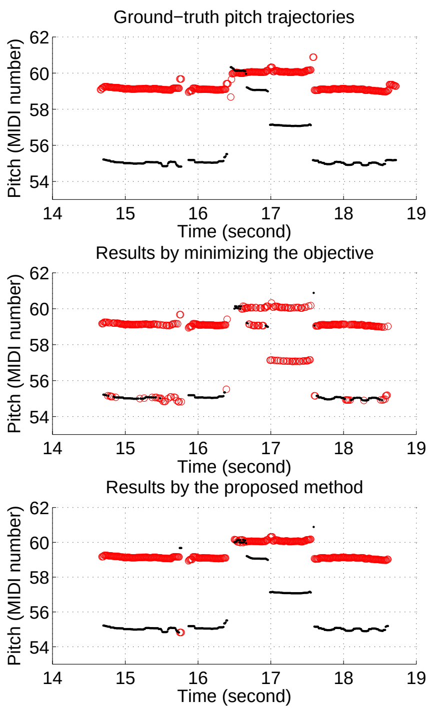


Fig. 1. Comparison of the ground-truth pitch streams, K-means clustering ($K = 2$) results (i.e. only minimizing the objective function), and the proposed method's results (i.e. considering both objective and constraints). Both the K-means and the proposed method take the ground-truth pitches as inputs, use 50-d harmonic structure from Section V as the timbre feature, and randomly initialize their clusterings. Each point in these figures is a pitch. Different instruments are marked with different markers (circles for saxophone and dots for bassoon).

In the middle panel of Figure 1, a number of pitches are clustered into the wrong trajectory. For example, the pitches around MIDI number 55 from 14.8 sec to 15.8 sec form a continuous contour and are all played by the bassoon. However, in the clustering, some of them are assigned to saxophone. In another example, from 16.8 sec to 17.6 sec, the K-means clustering puts two simultaneous pitches into the saxophone stream. This is not reasonable, since saxophone is a monophonic instrument.

If we know that different sources do not often perform the same pitch at the same time and all sources are monophonic, we can impose two kinds of constraints on some pairs of the pitches to improve clustering: A *must-link* constraint is imposed between two pitches that differ less than Δ_t in time and Δ_f in frequency. It specifies that two pitches close in both time and frequency should be assigned to the same cluster. A *cannot-link* constraint is imposed between two pitches in the same frame. It specifies that two simultaneous pitches should be assigned to different clusters. These must-links and cannot-links form the set of all constraints C. The bottom panel of Figure 1 shows the result obtained from our proposed algorithm, considering both the objective and constraints.

C. Constrained Clustering and Its Properties

Given the clustering objective and constraints, the multi-pitch streaming problem becomes a constrained clustering problem with binary constraints. In seeking a good clustering, the objective function (within-stream timbre inconsistency) should be minimized while the constraints (assumptions about pitch relationship) should be satisfied.

There exist a number of constrained clustering algorithms [37]–[39] that deal with binary constraints, however, they cannot be applied due to this problem's unique properties:

- *Inconsistent Constraints:* Constraints are imposed on pitch estimates which contain errors, hence the constraints themselves also contain errors. Also, the assumptions that the constraints are based on are not always correct. Two sources may occasionally perform the same pitch, and two pitches produced by the same monophonic source may be concurrent due to room reverberation. Therefore, the constraints may not be consistent with each other.
- *Heavily Constrained:* Since the pitch of each source often evolves smoothly over short periods (several frames), almost every pitch estimate is involved in some must-links. Also, since most of the time there are multiple sound sources playing simultaneously, almost every pitch estimate is involved in some cannot-links. This makes the clustering problem heavily constrained.

Because of the "Inconsistent Constraints" property, there may not exist any clustering satisfying all the constraints. This makes existing algorithms [37], [38] inapplicable, since they attempt to find a clustering minimizing the objective while satisfying all the constraints. Even if we assume all constraints are consistent, [39] proved that finding a feasible solution, i.e. a label assignment without violating any constraint, of a clustering problem containing cannot-links is NP-complete.

Therefore, we should not try to satisfy all the constraints. Instead, we seek an algorithm that minimizes the objective while satisfying as many constraints as possible. An *Incremental Constrained Clustering* algorithm [39] fits this purpose. However, we will show that [39] is inapplicable to our problem in Section IV-A. Thus, we need to design a new incremental constrained clustering algorithm for our problem.

IV. ALGORITHM

In our work, a point p is a pitch estimate with an associated fundamental frequency, time, and timbre. A partition Π is an assignment of each pitch estimate to exactly one of K streams (clusters). This is also referred to as a clustering. The objective function $f(\Pi)$ returns the total within-stream timbre inconsistency, as described in Eq. (1).

In this section we describe a novel incremental constrained clustering algorithm. It starts from an initial partition Π_0 that satisfies a subset of all the constraints $C_0 \subset C$. Then it iteratively minimizes the objective function while incrementally satisfying more constraints. Note that, although we apply it to the streaming problem, the algorithm is more general than that and may be applied to any problem of set partitioning under constraints with an objective function.

A. Forming the Initial Partition

For a general incremental constrained clustering problem, the initial partition Π_0 can be simply set by a random label assignment of all the instances. For our multi-pitch streaming problem, we can have a more meaningful initialization: We set Π_0 by sorting pitches in each frame from high to low and assigning labels from 1 to K. This is possible because, if there are K monophonic sound sources, there are at most K pitches in each frame. We call this *pitch-order initialization.*

For many audio mixtures, including much polyphonic music and two-talker speech of different genders, pitch-order initialization is better than random initialization. This is because pitch streams may not often interweave in these cases. Nevertheless, pitch-order initialization does not solve the streaming problem even in these cases. This is because the pitch estimates from MPE systems typically contain many pitch errors, causing stream assignment based on pitch order to fail. In the experiments, we compare the effects of different initializations.

For pitch-order initialization Π_0, its satisfied constraints C_0 contains all cannot-links in C. This is because cannot-links are only imposed on concurrent pitches, which are assigned to different clusters (streams) in Π_0.

Given Π_0 and C_0, we want to minimize the objective function while incrementally adding constraints. Davidson *et al.* [39] showed that incrementally adding new constraints is NP-hard in general, but they identified several conditions under which the clustering could be efficiently updated to satisfy the new and old constraints. The conditions require either 1) at least one point involved in the new constraint is not currently involved in any old constraint or 2) the new constraint is a cannot-link.

For our problem, however, from the initial constraints C_0, neither of the two conditions can be met. This is because: 1) Due to the "Heavily Constrained" property, almost every pitch estimate has already been constrained by some cannot-links, so Condition 1 is not met. 2) Since all the cannot-links are already in C_0, any new constraint will be a must-link, so Condition 2 is not met. Therefore, the algorithm in [39] will do nothing beyond the pitch-order initialization.

B. A Novel Incremental Constrained Clustering Algorithm

Here we describe a new incremental constrained clustering algorithm (Algorithm 1) that alternately updates the partition and set of satisfied constraints, starting from initial partition Π_0 and satisfied constraints C_0.

Suppose we are in the t-th iteration, where the previous partition is Π_{t-1} and the set of constraints that it satisfies is C_{t-1}. We first update Π_{t-1} to a new partition Π_t which strictly decreases the objective function *and* also satisfies C_{t-1} (Line 4). We then find which (if any) constraints that Π_t satisfies, which were not satisfied by Π_{t-1}. We add those constraints to the set of satisfied constraints, giving us C_t (Line 5). So we have $f(\Pi_{t-1}) > f(\Pi_t)$ and $C_{t-1} \subseteq C_t$. Although in some iterations Π_t does not satisfy more constraints than Π_{t-1} and $C_{t-1} = C_t$, in general the set of satisfied constraints will expand. The key of this algorithm is Line 4, and will be explained in Section IV-C and Algorithm 2. If no new partition is returned in Line 4, Algorithm 1 will terminate. We will show that it always terminates in Section IV-F.

Algorithm 1: IncrementalClustering

Input : N points to be partitioned into K clusters; f: the objective function to be minimized; C: the set of all constraints; Π_0: initial partition; $C_0 \subseteq C$: constraints satisfied by Π_0.

Output: A partition Π_t and constraints it satisfies, C_t.

1 $t \leftarrow 0$;
2 **do**
3     $t \leftarrow t + 1$;
4   Π_t = `FindNewPartition(`Π_{t-1},C_{t-1},f`)`;
5   C_t = The set of constraints satisfied by Π_t;
6 **while** $\Pi_t \neq \Pi_{t-1}$;
7 **return** Π_t and C_t;

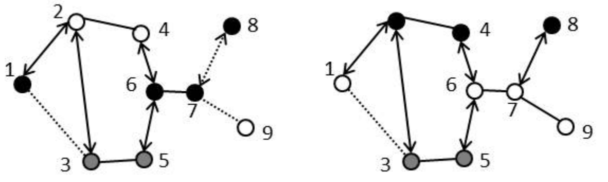
C. Find A New Partition by Swapping Labels

In Line 4 of Algorithm 1, we want to update Π_{t-1} to a new partition Π_t that strictly decreases the objective function and also satisfies the constraints in C_{t-1}. We do this by moving at least one point between streams in Π_{t-1}. However, if we move some point p (recall points are pitch estimates) from cluster S_k to cluster S_l (recall clusters are streams), all the points that have a must-link to p according to C_{t-1} should be moved from S_k to S_l, because we want C_{t-1} to be satisfied by the new partition as well. Then all the points in S_l that have cannot-links to any of the above-mentioned points need also be moved out of S_l. If they are moved to another stream S_m, then the points in S_m that have cannot-links with the above-mentioned points in S_l according to C_{t-1} need to be moved, and this will cause a chain reaction.

We deal with this issue by defining the *swap set* of points that may be affected by changing the stream of p from S_k to S_l. Then we will define the *swap* operation to change the cluster label for all points in the swap set without breaking any currently-satisfied constraints in C_{t-1}.

Given a node p and two streams S_k and S_l, the swap set is the set of points from these clusters that have a path to p through the currently-satisfied constraints in C_{t-1}, subject to the condition that the path only involve points from streams S_k and S_l. Note that a currently-satisfied constraint involving points from other streams is not an edge here. In other words, the swap set is the maximally connected subgraph containing p between streams S_k and S_l.

Consider the left panel of Figure 2. Suppose we want to move point 6 in the left panel from black to white. The swap set for point 6 in a black-white swap is the set of points 1, 2, 4, 6 and 7. They form the maximally connected graph containing the point 6 between the two clusters, using the currently satisfied constraints as edges. The swap set for point 6 in a black-gray swap is points 3, 5, 6, 7.



(a) before swap (b) after swap

Fig. 2. An illustration of the swap operation. Here we have 9 points from 3 streams (white, gray and black). Must-links are depicted as lines without arrows, and cannot-links are lines with arrows. Constraints satisfied by the current partition are in solid lines, and those not satisfied are in dotted lines.

The *swap* operation involves flipping the cluster for all points in the swap set. Those formerly in S_l move to S_k. Those formerly in S_k move to S_l. Figure 2 illustrates a white-black swap. Here, we swap these five points and get a new partition shown in the right panel. The new partition satisfies all the constraints that were satisfied before, but it also satisfies two more constraints in this example, i.e. the cannot-link between point 7 and 8, and the must-link between point 7 and 9.

D. Proof Constraints are Preserved by a Swap

The swap operation is guaranteed to preserve all currently-satisfied constraints. Proof:

Split the constraints satisfied prior to swap into those between points within the swap set, and those involving points outside the swap set. First consider the within-swap-set constraints. All satisfied must-links between points in the swap-set remain satisfied after a swap. This is true because all points in the swap set that share a cluster prior to the swap will share a cluster after the swap. Similarly, all cannot-links between points in the swap-set remain satisfied, since all points which are not in the same cluster are still not in the same cluster after the swap.

Now we address currently-satisfied constraints involving points outside the swap set. Any of these constraints must be a cannot-link, and the outside point involved in this constraint must be in a third stream different from the streams that define the swap set. This is because otherwise the outside point would be in the swap set, according to the swap set definition. Since the swap operation never assigns the cluster label of the third stream to any point in the swap set, this cannot-link remains satisfied. Consider point 3 in Figure 2 as an illustrative example.

E. Finding a New Partition

The swap operation assures the set of satisfied constraints is expanded (or remained the same), but it does not say anything about the objective function. It is possible that the objective function is not decreased after the swap.

To make sure the objective function is also strictly decreased, we only do a swap operation that does strictly decrease the objective function. To find such a swap operation, we randomly traverse all the points and try all their swap operations (i.e. try changing streams for each pitch estimate). We stop the traversal when we find any swap operation that decreases the objective function and return the new partition after the swap. If we cannot find such a swap operation after traversing all the points, then there is no new partition that strictly decreases the objective function and also satisfies the currently satisfied constraints. In this case, we return the current partition and Algorithm 1 terminates. This subroutine is described in Algorithm 2.

Algorithm 2: FindNewPartition

Input : Π_{t-1}: a K-partition of N points; C_{t-1}: constraints satisfied by Π_{t-1}; f: objective function to be minimized.

Output: Π_t: A new K-partition that also satisfies C_{t-1} and with $f(\Pi_t) \leq f(\Pi_{t-1})$.

$f_{best} \leftarrow f(\Pi_{t-1})$;
$\Pi_t \leftarrow \Pi_{t-1}$;
while $f_{best} == f(\Pi_{t-1})$ *&& not all the points* $p_1, \cdots, p_N$ *are traversed* **do**
  Pick p_n at random, without replacement. Suppose p_n is in stream S_k.;
  for $l \leftarrow 1, \cdots, K; l \neq k$ **do**
    Find the swap set of p_n between S_k and S_l in Π_{t-1} according to C_{t-1}; Do *swap* to get a new clustering Π_s and its centroids.;
    if $f(\Pi_s) < f_{best}$ **then**
      $f_{best} \leftarrow f(\Pi_s)$;
      $\Pi_t \leftarrow \Pi_s$;
    end
  end
end
return Π_t;

F. Algorithm Analysis

Algorithm 1 always terminates, possibly to some local optimum, because the space of feasible partitions is finite and in every iteration the new partition found by "FindNewPartition" strictly decreases the objective function.

In each iteration of our algorithm, the space of feasible partitions, given the satisfied constraints, is shrunk. Take the multi-pitch streaming problem as an example. Suppose there are K monophonic sources, and T time frames. In the worst case, the total number of pitches N equals to KT, then the size of the solution space without any constraints is K^{KT}. After imposing the initial constraints (all cannot-links) C_0, the space is shrunk to about $(K!)^T$. This is because, each time frame has $K!$ distinct assignments of K pitch estimates to K streams.

After imposing all the constraints C (assuming they are consistent), suppose the typical number of pitch estimates in a must-link group (a group of pitches connected by must-links) is M, then there are in total about KT/M must-link groups. Suppose also that each must-link group is involved in a K-clique with cannot-link edges (each note is overlapped by $K-1$ other notes, which can be common). Then the solution space is further reduced to $(K!)^{\frac{KT}{MK}} = (K!)^{T/M}$. A typical value of M is 20 (i.e. a must-link group spans 20 frames). With the constraints expanded, not only more domain knowledge is incorporated to refine the clustering, the shrunk space also eliminates a lot of local minima of the objective function, where Algorithm 1 can be trapped.

The worst case running time of each iteration of Algorithm 1 is $O(KN^2)$, in terms of the number of all points N and the number of clusters K. This is because in Algorithm 2, there are at most NK nested loops from Line 6 to Line 11. Line 6, 7 and 9 all cost $O(N)$ operations in the worst case (when the size of the swap set is $O(N)$). In most cases, however, the swap set is much smaller than N. Taking the multi-pitch streaming problem as an example, the size of a swap set typically does not increase with the length of the music and the number of sources. This is because breaks between notes (or words) naturally bound the number of pitch estimates that must be considered in a swap set to a constant. In this case, each iteration of Algorithm 1 costs $O(KN)$.

How long then, does Algorithm 1 take in practice? In our experiments, a typical four-part Bach chorale (25 seconds long) from the Bach10 dataset in Section VI has about 9,000 pitch estimates and 15,000 constraints. The algorithm takes about 300 iterations to terminate from pitch-order initialization. This requires about 11 minutes on one core of a 4-core 2.67GHz CPU). Assuming random initialization of the partition, the algorithm requires 2,800 iterations to terminate (43 minutes on the same computer). In practice, one can terminate the algorithm earlier, if the partition is already good enough.

V. Timbre Features

The constrained clustering approach described in this work depends on a clustering objective function which, in turn, depends on a timbre representation for the pitch estimates. While there are a number of approaches to representing timbre [40], [41], our problem formulation requires a simple approach that can be calculated from a multi-source mixture for pitch estimate in a single time frame, where time frames are on the order of 50 milliseconds in length. Here, we describe two previously-used timbre representations: harmonic structure and mel-frequency cepstral coefficients (MFCCs). We then propose a new representation: the uniform discrete cepstrum (UDC).

A. Harmonic Structure

This approach was previously used with success in [2]. It is defined as a vector of relative logarithmic amplitudes of the harmonics of a pitch estimate. The harmonics are at integer multiples of the pitch. We use the first 50 harmonics to create the timbre vector $\mathbf{t}_i$. We choose this dimensionality because most instruments have less than 50 prominent harmonics. For each harmonic, we use the peak-finder from [2] to see if there is a significant peak within a musical quarter-tone. If no peak is associated, the magnitude of the harmonic is set to 0dB, else it is set to the value of the nearest peak. Then, the representation is normalized. This is a simple, clear representation. Note that the assumptions here are that it will not be overly impacted by overlapping harmonics from different sources, and that the within-source variation in harmonic structure will be less than the between-source difference.

B. Mel-frequency Cepstral Coefficients (MFCC)

MFCCs have been widely used to represent the timbre of speech signals in many problems, including speech recognition, speaker identification, etc. To calculate an MFCC feature vector for an audio frame, the magnitude spectrum of the frame is first mapped onto the Mel-frequency scale to better approximate the frequency resolution of the human ear:

$$\mathbf{mel}(f) = \begin{cases} 3f/200 & \text{if } f \leq 1000\text{Hz} \\ 15 + \ln(f/1000)/0.0688 & \text{if } f > 1000\text{Hz} \end{cases} \quad (2)$$

Then, the typical steps used in creating an ordinary cepstrum (see Section V-C) are applied. In this work, we use Dan Ellis's implementation [42], with a 40-band Mel filter bank.

To calculate the MFCC feature for an individual pitch estimate, we first need to separate its magnitude spectrum from the mixture. We do so using a simple harmonic masking approach [43]. Assume each pitch estimate corresponds to a single source. If there are K pitch estimates in the current time-frame, then each frequency bin in the spectrum is a harmonic of between 0 and K pitch estimates. For nonharmonic bins ($hc = 0$) the mixture energy is evenly distributed to all concurrent pitches. For a bin that is the harmonic of a single pitch estimate, the mixture energy in that bin is assigned to that single source. For a bin that is the harmonic of several pitch estimates, the mixture energy is distributed among the corresponding sources.

Here, the proportion of energy assigned to a pitch estimate decreases as the harmonic index increases. If the bin is the 10th harmonic of pitch p and the 2nd of pitch q, q will receive more energy. This distribution is in inverse proportion to the square of harmonic indices. It is equivalent to assuming that harmonic sources concentrate their energy in the lower partials, a reasonable assumption for many sources. The width of a harmonic is set to 40Hz for a 46ms-long hamming window (for music) and 60Hz for a 32ms-long hamming window (for speech), which approximately corresponds to the range where the power spectrum of the hamming window decreases for 6dB from the top.

C. Uniform Discrete Cepstrum

We now describe an alternate approach to calculating a cepstral representation only from a set of sparse and possibly non-uniform points in the mixture spectrum that are likely to come from a single source, without the requirement of separation. We name this representation as uniform discrete cepstrum (UDC).

Let $\mathbf{f} = [f_1, \cdots, f_N]^T$ and $\mathbf{a} = [a_1, \cdots, a_N]^T$ be the full set of frequencies and log-amplitudes of the mixture spectrum of discrete Fourier transform (DFT). Suppose $\hat{\mathbf{f}} = [\hat{f}_1, \cdots, \hat{f}_L]^T$ and $\hat{\mathbf{a}} = [\hat{a}_1, \cdots, \hat{a}_L]^T$ are the subset of the spectral points that are likely to solely belong to the source we want to model[3]. We call these points the *observable spectral points* for the source. These points, for a harmonic source in an

[3] In fact, $\hat{\mathbf{f}}$ need not to be a subset of frequency bins in Fourier analysis. They can be frequencies in between the bins, and $\hat{\mathbf{a}}$ can be the corresponding interpolated values. In this case, the first equality of Eq. (5) will be an approximation.

audio mixture, usually correspond to its harmonics. Therefore, we detect the first 50 harmonics of the source from its given pitch using the way described in Section V-A, and use them as the set of points. Then the UDC is calculated by

$$\mathbf{c}_{\text{udc}} = \hat{M}^T \hat{\mathbf{a}}, \tag{3}$$

where p is the order of the cepstral representation, i.e. the number of coefficients;

$$\hat{M} = \begin{pmatrix} 1 & \sqrt{2}\cos(2\pi 1 \hat{f}_1) & \cdots & \sqrt{2}\cos(2\pi(p-1)\hat{f}_1) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \sqrt{2}\cos(2\pi 1 \hat{f}_L) & \cdots & \sqrt{2}\cos(2\pi(p-1)\hat{f}_L) \end{pmatrix}. \tag{4}$$

Since the sparse set of observable spectral points $\hat{\mathbf{f}}$ and $\hat{\mathbf{a}}$ form a subset of the full spectrum $\mathbf{f}$ and $\mathbf{a}$, we can rewrite Eq. (3) as

$$\mathbf{c}_{\text{udc}} = M^T \tilde{\mathbf{a}} = (M^T M)^{-1} M^T \tilde{\mathbf{a}}, \tag{5}$$

where $\tilde{\mathbf{a}}$ is a sparse log-amplitude spectrum of the same dimensionality with the full mixture spectrum $\mathbf{a}$. It takes values of $\mathbf{a}$ at the sparse observable spectral points, and zeros everywhere else. M contains the first p columns of a discrete-cosine transform (DCT) matrix:

$$M = \begin{pmatrix} 1 & \sqrt{2}\cos(2\pi 1 f_1) & \cdots & \sqrt{2}\cos(2\pi(p-1)f_1) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \sqrt{2}\cos(2\pi 1 f_N) & \cdots & \sqrt{2}\cos(2\pi(p-1)f_N) \end{pmatrix}. \tag{6}$$

$\hat{M}$ in Eq. (4) is a sub-matrix (a subset of rows) of M in Eq. (6) at the L observable frequency bins. The second equality of Eq. (5) comes from the fact that the columns of M are orthogonal and $M^T M$ is an identity matrix.

Eq. (3) tells us how to calculate UDC and Eq. (5) tells us the calculation is equivalent to taking the DCT of the sparse log-amplitude spectrum $\tilde{\mathbf{a}}$. This seems like a usual cepstrum calculation, but note that $\tilde{\mathbf{a}}$ is not a separated spectrum, unlike what is used in MFCC calculation in Section V-B.

The development of UDC was inspired by the discrete cepstrum (DC)[4] proposed by Galas and Rodet in [44], which is calculated by

$$\mathbf{c}_{\text{dc}} = (\hat{M}^T \hat{M})^{-1} \hat{M}^T \hat{\mathbf{a}}, \tag{7}$$

UDC and DC share the virtue that both can be calculated from a sparse set of frequencies of the spectrum, hence can be used to calculate a cepstral representation of a source from the mixture spectrum without the need of source separation. However, they have some fundamental differences in terms of their physical meanings. We need to describe the basic concept of cepstral representations to explain this.

The concept of a cepstrum is to approximate (up to a scale) a log-amplitude spectrum $a(f)$ by a weighted sum of p sinusoids

$$a(f) \approx c_0 + \sqrt{2}\sum_{i=1}^{p-1} c_i \cos(2\pi i f), \tag{8}$$

[4]Its name is confusing since, "discrete" often refers to the implementation in the digital world, such as discrete cosine transform. Here, however, "discrete" refers to the fact that a discrete cepstrum can be calculated from a number of isolated analysis frequencies in a spectrum.

where the weights $\mathbf{c} = [c_0, c_1, \cdots, c_{p-1}]^T$ form a cepstrum of order p; f is the normalized frequency (Hz divided by the sampling rate). A common approximation criterion is to minimize the Euclidean distance between both sides of Eq. (8), which leads to the least square solution of the weights.

Another concept needed to differentiate DC and UDC is the difference between a spectral envelope and a smoothed spectrum. A *spectral envelope* is a smoothed curve that wraps tightly around the magnitude spectrum, linking some peaks [45]. Its calculation only uses some spectral peaks while ignoring the other parts of the spectrum. A *smoothed spectrum* is a smoother version of the magnitude spectrum, where some spectral details (e.g. small peaks and valleys) are smoothed out. Its calculation (e.g. taking a moving average of the spectrum) usually uses the whole spectrum.

From Eq. (7) we can see that DC is the least square solution of Eq. (8) only considering the observable frequencies. In other words, DC represents (and can reconstruct) a smooth curve that approximately goes through the observable spectral points. When these points are harmonics of a source used in this paper, this curve is a spectral envelope of the source spectrum.

From Eq. (3) we can see that UDC is *not* the least square solution of Eq. (8) only considering the observable frequencies. However, from Eq. (5) we can see UDC is the least square solution of Eq. (8) considering all frequencies and using the sparse spectrum $\tilde{\mathbf{a}}$. In other words, UDC represents (and can reconstruct) a smooth curve that approximates $\tilde{\mathbf{a}}$, i.e. a smoothed spectrum of the source.

Both of a spectral envelope and a smoothed spectrum can serve as a timbre representation. In our experiments we found that UDC serves as a good timbre feature for comparing one with another. However, DC's calculated from different spectra of the same source were found not similar to each other. This prevents it being used as a timbre feature of sound sources for statistical comparison purposes. In fact, DC was only used for the purpose of spectral envelope reconstruction when it was proposed in [44] and improved in [46]. It was not proposed or tested as a timbre feature for statistical comparisons. While the full analysis and experiments about why UDC is better than DC as a timbre feature for statistical comparisons exceeds the scope of this paper, in the experiments (Figure 3 and 6) we show this is the case in the multi-pitch streaming application.

VI. Experiments on Polyphonic Music

In this section, we test the proposed multi-pitch streaming algorithm on polyphonic music recordings. Through the experiments, we want to answer the following questions: 1) Which timbre representation is best for streaming? 2) How does the proposed streaming algorithm perform on music recordings with different polyphony? 3) What is the effect of different input MPE systems on streaming performance? 4) Which components (e.g. initialization, timbre objective, locality constraints) of the proposed algorithm significantly affect the streaming results? The code and datasets can be downloaded at http://www.ece.rochester.edu/~zduan/.

A. Experimental Setup

1) Dataset: We use the Bach10 dataset. This dataset consists of real musical instrumental performances of ten pieces of J.S. Bach four-part chorales. Each piece is about thirty seconds long and was performed by a quartet of instruments: violin (Track 1), clarinet (Track 2), tenor saxophone (Track 3) and bassoon (Track 4). Each musician's part was recorded in isolation while the musician listened to the others through headphones. The sampling rate was 44.1kHz. The ground-truth pitch trajectories were created using the robust single pitch detection algorithm YIN [47] on the isolated instrument recordings, followed by manual corrections where necessary.

For each of the ten pieces, we created single-channel recordings of six duets, four trios and one quartet, by mixing the individual tracks with different combinations. This provided us in total 110 pieces of music with different polyphony.

2) Input Multi-pitch Estimates: As stated before, the proposed multi-pitch streaming algorithm can take frame-level pitch estimates from any MPE algorithm as inputs. Here we test it using three MPE algorithms. We provide the number of instruments in the mixture to these MPE algorithms and let them estimate the instantaneous polyphony in each frame.

The first one is our previous work [17], denoted by "Duan10". It is a general MPE algorithm based on probabilistic modeling of spectral peaks and non-peak regions of the amplitude spectrum.

The second one is [14], denoted by "Klapuri06". We use Klapuri's original implementation and suggested parameters. This is an iterative spectral subtraction approach. At each iteration, a pitch is estimated according to a salience function and its harmonics are subtracted from the mixture spectrum.

The third one is [13], denoted by "Pertusa08". We use Pertusa's original implementation and suggested parameters. This is a rule-based algorithm. In each time frame, it first selects a set of pitch candidates from spectral peaks, then all their possible combinations are generated. The best combination is chosen by applying a set of rules, taking into account its harmonic amplitudes and spectral smoothness.

Since pitch estimates of MPE algorithms contain errors and these errors will be propagated to the streaming results, we also use ground-truth pitches as inputs and let the proposed approach cluster these error-free pitches into trajectories.

3) Parameter Settings: For all the MPE algorithms, the audio mixture is segmented into frames with 46ms-long frames with 10ms hop size. The pitch range of Duan10 and Klapuri08 is set to C2-B6 (65Hz-1976Hz). The pitch range of Pertusa08 is set as-is.

In imposing the must-links, we set the time and frequency difference thresholds Δ_t and Δ_f to 10ms and 0.3 semitones, respectively. 10ms is the time difference between adjacent frames, and 0.3 semitones correspond to the range that the pitch often fluctuates within a note. These thresholds are quite conservative to assure that most must-links are correct.

After clustering, we perform an additional postprocessing step. We merge two adjacent must-link groups of the same instrument if their time gap (the time interval between the offset of the previous group and the onset of the latter group) is less than 100ms. We also remove must-link groups that are shorter than 100ms. We choose this threshold because 100ms is the length of a 32nd note in a piece of music with a moderate tempo of 75 beats per minute. This step does not affect the objective streaming accuracy, but improves the perceptual quality significantly.

4) Evaluation Measure: Given a polyphonic music with K monophonic instruments, the proposed multi-pitch streaming algorithm streams pitch estimates in individual frames into K pitch trajectories, each of which corresponds to an instrument. To evaluate the streaming results, we first find the bijection between the K ground-truth pitch trajectories and the K estimated trajectories. In the experiment, we choose the bijection that gives us the best overall multi-pitch streaming accuracy. This accuracy is defined as follows. For each estimated pitch trajectory, we call a pitch estimate in a frame correct if it deviates less than 3% in Hz (a quarter-tone) from the pitch in the same frame and in the *matched* ground-truth pitch trajectory. This threshold is in accordance with the standard tolerance used in measuring correctness of pitch estimation for music [14]. Then the overall multi-pitch estimation and streaming accuracy for one piece of music is defined as:

$$\mathrm{Acc} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}, \tag{9}$$

where TP (true positives) is the number of correctly estimated and streamed pitches, FP (false positives) is the number of pitches that are present in some estimated trajectory but do not belong to its matched ground-truth trajectory, and FN (false negatives) is the number of pitches that belong to some ground-truth trajectory but are not present in its matched estimated trajectory.

B. Comparison of Timbre Features

To investigate the effects of timbre features on the multi-pitch streaming performance, we ran the proposed approach on the ground-truth pitch inputs, comparing system performance using four timbre representations: 50-d harmonic structure, 21-d MFCC, 21-d DC, and 21-d UDC. The harmonic structure, DC and UDC are all calculated from the mixture spectrum directly; while the MFCC is calculated from separated signal of each pitch estimate using harmonic masking, as described in Section V-B. To remove the effect of pitch height arrangement of different tracks, we initialize all partitions randomly. Figure 3 shows the results.

In this and all the following box plots figures, each box represents the accuracy distribution of a number of music pieces. The lower and upper lines of each box show 25th and 75th percentiles of the sample. The line in the middle of each box is the sample median. The lines extending above and below each box show the extent of the rest of the samples, excluding outliers. Outliers are defined as points over 1.5 times the interquartile range from the sample median and are shown as crosses.

For all the polyphonies, harmonic structure and UDC work well, and outperform MFCC and DC significantly. The validity of harmonic structure for musical instruments has been validated in our previous work [2], [27]. When polyphony is two or three, UDC works slightly better than harmonic structure, and when polyphony is four they are comparable.

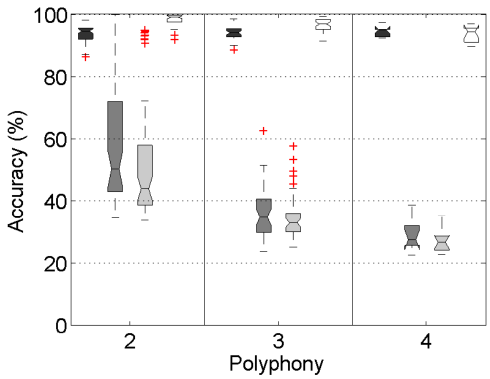


Fig. 3. Comparison of multi-pitch streaming accuracy of the proposed approach using four kinds of timbre features: 50-d harmonic structure (black), 21-d MFCC (dark gray), 21-d DC (light gray) and 21-d UDC (white). Input pitches are ground-truth pitches without track information. Clusterings are randomly initialized to remove the pitch order information. Each box of polyphony 2, 3 and 4 represents 60, 40 and 10 data points, respectively.

On the other hand, MFCC and DC achieve much worse results. Two reasons may be credited for the bad performance of MFCC. First, the source separation step in the MFCC calculation is unreliable. Second, besides the harmonic part, MFCC also encodes the inharmonic part of the separated spectrum, which is more error-prone.

C. The Effect of the Input Multi-pitch Estimation

Figure 3 shows that 21-d UDC achieves comparable or slightly better results than 50-d harmonic structure when the input pitches are ground-truth pitches. However, we found when the input pitches are pitch estimates of musical instruments provided by MPE approaches, harmonic structure outperforms UDC, especially when polyphony is four. Therefore, in the following experiments, we use 50-d harmonic structure to test the performance of our system in combination with several existing MPE approaches in Figure 4.

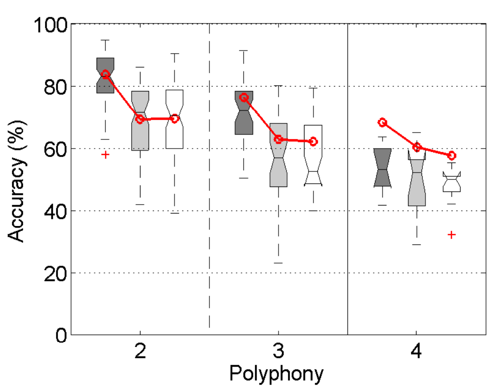


Fig. 4. Boxplots of overall multi-pitch streaming accuracies achieved by the proposed method on the Bach chorale music pieces, taking input pitch estimates provided by three MPE algorithms: Duan10 (dark gray), Klapuri06 (light gray) and Pertusa08 (white). Each box of polyphony 2, 3 and 4 represents 60, 40 and 10 data points, respectively. The lines with circles show the average accuracy of the three input MPE algorithms.

Note MPE accuracy is defined as the overall multi-pitch streaming accuracy except that a pitch estimate is called correct only according to the time and frequency criteria, ignoring the trajectory information. Therefore, the average overall multi-pitch streaming accuracy cannot be higher than the average MPE accuracy.

Comparing the accuracies achieved with the three MPE inputs, we see that the one taking Duan10 as inputs are much better than those taking Klapuri06 and Pertusa08 inputs. This is in accordance with their average input MPE accuracies. More accurate MPE inputs lead to more accurate multi-pitch streaming results. The median accuracy achieved by the best multi-pitch streaming configuration (using Duan10 as input) is about 83% for duets, 72% for trios and 53% for quartets. This is promising, considering the difficulty of the task. The only information provided to the MPE algorithm and the proposed streaming algorithm about these music recordings is the number of instruments in the mixture.

D. Individual Analysis of System Components

As described in Section III, the proposed approach utilizes two kinds of information to cluster pitch estimates. Timbre is utilized through the objective function; while pitch locality information is utilized through the constraints. We claimed that both are essential to achieve good results. In addition, we claimed that the pitch-order initialization is more informative than a random initialization in Section IV-A.

In this experiment, we analyze the effect caused by each individual aspect and their combinations. More specifically, we run the clustering algorithm in the following configurations, with the 50-d harmonic structure as the timbre feature:

1) *Timbre*: from random initialization, run the algorithm to only optimize the timbre objective function; equivalent to K-means algorithm.
2) *Locality*: from random initialization, run the algorithm to only satisfy more locality constraints.
3) *T+L*: from random initialization, run the full version of the proposed algorithm to optimize the timbre objective as well as satisfy more locality constraints.
4) *Order*: clustering by only pitch-order initialization.
5) *O+T*: Configuration 1 with pitch-order initialization.
6) *O+L*: Configuration 2 with pitch-order initialization.
7) *O+T+L*: Configuration 3 with pitch-order initialization.

Figure 5 shows box plots of the multi-pitch streaming accuracy of these configurations on the ten quartets. It can be seen that the pitch-order initialization itself (Order) does not provide a satisfying clustering, even though the pitch trajectories of the Bach chorales rarely interweave. This is due to the polyphony estimation and pitch estimation errors. Only using the locality constraints information, no matter what initialization (Locality and O+L), achieves the worst clustering. Only using the timbre information (Timbre and O+T) achieves better clustering but still non-satisfying. Utilizing both timbre and locality information (T+L and O+T+L) achieves significantly better clustering than only using either one of them. This supports our claim that both timbre and locality are essential for good clustering. In this case, the pitch-order initialization does not help the clustering much, as a nonparametric paired

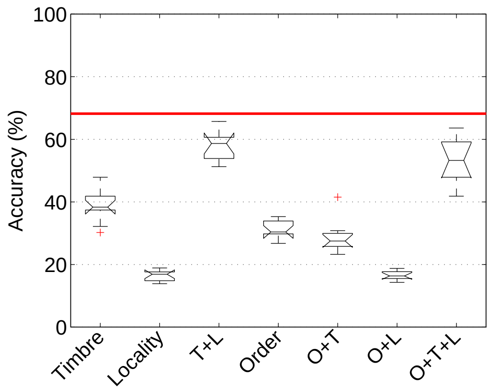


Fig. 5. Box plots of multi-pitch streaming accuracies of the proposed approach with different system configurations, taking the same input pitch estimates from Duan10. Each box contains 10 data points corresponding to the 10 quartets. The horizontal line is the average input MPE accuracy.

sign test favors the null hypothesis that the median difference between T+L and O+T+L is 0 ($p = 0.11$). However, the pitch-order initialization does make the algorithm converge faster, because the final clustering is "closer" (requires less swaps) from the pitch-order initialization than from a random initialization, since the pitch trajectories of the music pieces do not often interweave. For example, the number of iterations for Algorithm 1 to terminate on the first quartet is reduced from 2781 to 313.

VII. Experiments on Multi-talker Speech

We also tested the proposed multi-pitch streaming algorithm on multi-talker speech. Similar to the music experiments, we want to answer the questions proposed in the beginning of Section VI, but in the speech context.

A. Experimental setup

1) Dataset: The dataset we use is the Pitch-Tracking Database from Graz University of Technology (PTDB-TUG) [48]. This database consists of recordings of twenty English native speakers (ten male and ten female) from different home countries (USA, Canada, England, Ireland and South Africa), reading phonetically rich sentences from the TIMIT corpus [49]. The TIMIT corpus consists of 450 phonetically-compact sentences and 1890 phonetically-diverse sentences. Each sentence was read by one female and one male subject. In total there are 4680 recorded utterances, 900 of which are of phonetically-compact sentences and 3780 are of phonetically-diverse sentences. Each utterance has about four seconds of voice and a couple of seconds of silence before and after. All the recordings were recorded in 48kHz.

Among the 3780 phonetically-diverse utterances from all twenty subjects, we selected five male and five female subjects to form the test set. This accounts for 1890 utterances. We randomly mixed these utterances with equal RMS levels to generate each multi-talker speech mixture. We considered four conditions according to the number of talkers and their gender relations: two-talker different gender (DG), two-talker same gender (SG), three-talker DG and three talker SG. We generate 100 mixtures for each condition, totalling 400 test mixtures. Among the 100 three-talker DG mixtures, 47 are mixtures of one male and two females. Among the 100 three-talker SG mixtures, 52 are female mixtures.

Since we found the ground-truth pitch tracks provided with the data set contained some errors, we generate our own ground-truth pitch tracks with Praat [50] on the utterances, using a frame length of 32 ms and a hop size of 10 ms. The pitch range of the utterances is between 65Hz and 370Hz.

2) Input Multi-pitch Estimates: Similar to the music experiments, we ran the proposed streaming approach using input pitch estimates from different MPE algorithms. Again, we provide the maximum number of talkers in the mixture to these MPE algorithms and let them estimate the instantaneous polyphony in each frame. The first one is our algorithm [17], denoted "Duan10". We trained this system with 500 multi-talker mixtures using phonetically-compact utterances of the other five male and five female subjects.

The second one is [18], denoted as "Wu03". We use their original implementation and suggested parameters. This algorithm uses a hidden Markov model (HMM) to model both the change in instantaneous polyphony and pitch values. It can estimate pitches of up to two simultaneous talkers,

The third one is [21], denoted as "Jin11". We use their original implementation and suggested parameters. This algorithm extends [18] to reverberant environments, and can also estimate pitches of up to two simultaneous talkers.

Similar to the music experiments, we also use ground-truth pitches as inputs and let the proposed approach to cluster these error-free pitches into trajectories.

3) Parameter Settings: The audio mixtures are segmented into frames with length of 32ms and hop size of 10ms. The pitch range is set to 65Hz-370Hz for all algorithms. In imposing the must-link constraints, we set the time and frequency difference thresholds Δ_t and Δ_f to 10ms and 1 semitone, respectively. The frequency threshold is larger than that used for music, since speech utterances often have fast gliding pitch contours.

4) Evaluation Measure: As in Section VI-A4, we use the multi-pitch estimation and streaming accuracy in Eq. (9) to measure the performance of the proposed approach. Differently, the frequency difference threshold to judge if a pitch estimate is matched with a ground-truth pitch is set to 10% of the ground-truth pitch frequency in Hz. This is larger than what is used for music, but is commonly used in existing multi-pitch analysis methods [18], [21], [31] for speech.

5) Comparison Method: We compare the proposed approach with two state-of-the-art multi-pitch estimation and streaming systems. The first one is a supervised method based on a factorial HMM [31], denoted by "Wohlmayr11". One HMM is used for each talker to estimate and stream the talker's pitches. The HMM parameters are trained on isolated training utterances. In our comparison, we use their source code and provided gender-dependent models, which give the most supervision information that we can use. The gender information gives it a small information advantage over our proposed method and the other comparison method.

The other method is an unsupervised method designed for cochannel speech separation [32], denoted by "Hu12". We use their source code and suggested parameters. This method estimates a pitch trajectory for each talker to construct a binary time-frequency mask to separate the mixture spectrogram. This method is built on the tandem algorithm [51]. Similar to the proposed approach, [32] also views the multi-pitch streaming problem as a constrained clustering problem, although the formulations are different. Note that [32] is only designed and tested for two-talker speech mixtures.

B. Comparison of Timbre Features

We ran the proposed approach with four kinds of features on the ground-truth pitch inputs: 50-d harmonic structure, 21-d MFCC, 21-d DC, and 21-d UDC.

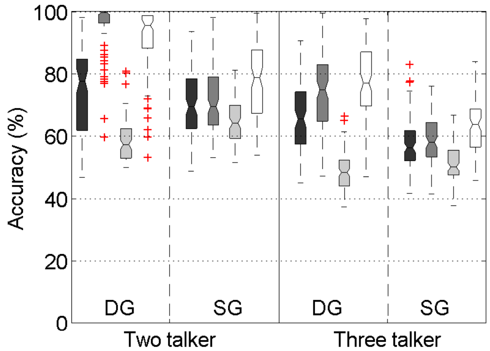


Fig. 6. Comparison of multi-pitch streaming accuracies of the proposed approach using three kinds of timbre features: 50-d harmonic structure (black), 21-d MFCC (dark gray), 21-d DC (light gray) and 21-d UDC (white). Input pitches are ground-truth pitches without track information.

The results are shown as boxplots in Figure 6. Specifications of all boxplots in this paper are described in Section VI-B. In the figure, we can see the DC feature achieves the worst results in all conditions. Second, in three out of four conditions, MFCC and UDC both significantly outperform harmonic structure, supported by a paired sign test at the 5% significance level. Third, in the two-talker DG (different gender) condition, both MFCC and UDC achieve very good streaming accuracy where MFCC achieves almost perfect results. However, when the conditions become harder, especially in the SG (same gender) conditions, UDC significantly outperforms MFCC. This is because the calculation of MFCC requires source separation, which becomes less reliable when there is more overlap between concurrent sources. In contrast, the calculation of UDC is performed directly from the mixture spectrum.

C. Overall Results

Figure 7 shows the overall comparison between Wohlmayr11, Hu12 and the proposed approach with input from three MPE algorithms, using the 21-d UDC timbre feature. It can be seen that the unsupervised methods (Hu12 and the proposed method with different inputs) significantly outperform Wohlmayr11, which uses the gender information in the mixture. We note that Wohlmayr11 is a supervised approach and its full strength can only be shown when a model is trained for each talker in the mixture. The good results obtained by the proposed method illustrate both its compatibility with different MPE algorithms and its effectiveness in performing streaming.

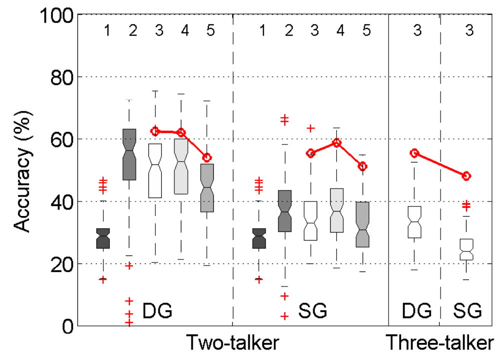


Fig. 7. Comparison of multi-pitch streaming accuracies of 1) Wohlmayr11, 2) Hu12, and the proposed approach taking inputs from 3) Duan10, 4) Wu03 and 5) Jin11. Each box has 100 data points. The circled red lines above the boxes show the average accuracy of input pitch estimates, prior to streaming.

The proposed system achieves statistically indistinguishable results from the state-of-the-art method Hu12. In the two-talker different-gender condition, the proposed system (taking either Duan10 or Wu03 as input), showed performance indistinguishable at the 5% significance level from Hu12 using a nonparametric paired sign test. Similarly, in the two-talker single-gender condition, Hu12 and Proposed (taking Wu03 as input) show results that are statistically indistingishable. However, the proposed multi-pitch streaming approach handles general harmonic sounds (as shown in Section VI) and speech mixtures with over two simultaneous talkers (as shown in the three-talker condition in Figure 7).

The errors caused by the proposed streaming approach (instead of the MPE algorithms) can be read from the gap between the box medians and the average accuracy of input pitch estimates. In the two-talker DG condition, this gap is fairly small, indicating that the proposed streaming algorithm works well. In the two-talker SG condition, this gap is significantly enlarged. This is because the pitch trajectories interweave with each other, making many must-link constraints imposed in the streaming process incorrect. The gap is further enlarged in the three-talker SG condition. One interesting thing to notice is that there is no significant difference of the performance between two-talker SG and three-talker DG conditions. This means that adding a talker with a different gender to an existing two-talker SG mixture does not influence the pitch estimation and streaming result much. The errors made by the proposed streaming approach can also be seen in Figure 6, which compares clustering using different timbre features based on ground-truth pitch estimates.

D. Individual Analysis of System Components

We analyze the effectiveness of different system components, similar to Section VI-D. Figure 8 shows box plots of

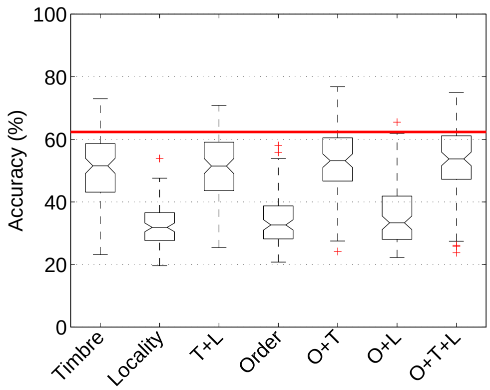


Fig. 8. Box plots of multi-pitch streaming accuracies of the proposed approach with different system configurations, taking the same input pitch estimates from Duan10. Each box contains 100 data points corresponding to the 100 two-talker DG excerpts. The horizontal line is the average input MPE accuracy, which sets an upper bound of the average streaming accuracy.

the multi-pitch streaming accuracies of these configurations on the 100 two-talker DG excerpts using the 21-d UDC feature. It can be seen that the pitch order information (Order) does not provide results as good as in the music dataset. This is expected, as the pitch activity of the two talkers often do not overlap in time and the pitch order initialization would label almost all the pitches incorrectly to the first cluster. Only using the locality information (Locality) or combining it with the pitch order information (O+L) also does not achieve good results, which is also expected.

What we did not expect is the good performance of only using the timbre information (Timbre) or its combination with pitch order (O+T). They achieve comparable results to T+L and O+T+L. This indicates that the UDC timbre feature is good to discriminate the two talkers, while the locality information does not help much.

VIII. Conclusions

In this paper we proposed a constrained-clustering approach for multi-pitch streaming of harmonic sound sources. Given pitch estimates in individual time frames provided by some multi-pitch estimation (MPE) algorithm, the proposed approach streams pitch estimates of the same source into a long and discontinuous pitch trajectory. This approach is unsupervised, i.e. it does not require pre-training source models on isolated recordings. It is general and can be applied to different kinds of harmonic sounds (e.g. musical instruments, speech, etc.). It is also highly compatible and can take the outputs of any MPE methods as inputs. Experiments show our approach achieves good performance on both speech and music.

We also proposed a new variant of cepstrum called uniform discrete cepstrum (UDC) to represent the timbre of sound sources. UDC can be calculated from the mixture spectrum directly. Experiments show that UDC is better suited to streaming than ordinary cepstrum features, such as MFCC, which requires source separation before feature calculation.

For future work, we would like to improve the problem formulation. Currently the constraints are binary. It may be beneficial to design soft constraints so that many existing nonlinear optimization algorithms can be used. In addition, we would like to incorporate higher-level domain knowledge such as musicological information into the objective and constraints. We also would like to design new features and apply the proposed algorithm on more kinds of harmonic sounds and explore its broader applications. Finally, designing a method that can jointly estimate the pitches and the streams that they belong to would be an important direction to pursue to solve the multi-pitch analysis problem.

Acknowledgment

We thank Ke Hu, Mingyang Wu, Zhaozhang Jin, DeLiang Wang and Michael Wohlmayr for providing their source code, and Anssi Klapuri and Antonio Pertusa for sharing their executable programs. We also thank the three anonymous reviewers for their thorough comments. This research was supported by National Science Foundation grants IIS-0643752 and IIS-0812314.

References

[1] A. Klapuri and M. Davy, eds., *Signal Processing Methods for Music Transcription*. Springer, 2006.

[2] Z. Duan, Y. Zhang, C. Zhang, and Z. Shi, "Unsupervised single-channel music source separation by average harmonic structure modeling," *IEEE Trans. Audio Speech Language Processing*, vol. 16, no. 4, pp. 766–778, 2008.

[3] J. Han and C.-W. Chen, "Improving melody extraction using probabilistic latent component analysis," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 33–36, 2011.

[4] M. Cooke, J. R. Hershey, and S. Rennie, "Monaural speech separation and recognition challenge," *Computer Speech and Language*, vol. 24, pp. 1–15, 2010.

[5] D.-n. Jiang, W. Zhang, L.-q. Shen, and L.-h. Cai, "Prosody analysis and modeling for emotional speech synthesis," in *Proc. IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, pp. 281–284, 2005.

[6] E. C. Cherry, "Some experiments on the recognition of speech, with one and two ears," *Journal of the Acoustic Society of America*, vol. 25, pp. 975–979, 1953.

[7] T. Tolonen and M. Karjalainen, "A computationally efficient multipitch analysis model," *IEEE Trans. on Speech and Audio Processing*, vol. 8, no. 6, pp. 708–716, 2000.

[8] A. de Cheveigné and H. Kawahara, "Multiple period estimation and pitch perception model," *Speech Commun.*, vol. 27, pp. 175–185, 1999.

[9] M. Davy, S. J. Godsill, and J. Idier, "Bayesian analysis of polyphonic Western tonal music," *Journal of the Acoustical Society of America*, vol. 119, pp. 2498–2517, 2006.

[10] R. C. Maher and J. W. Beauchamp, "Fundamental frequency estimation of musical signals using a two-way mismatch procedure," *Journal of the Acoustical Society of America*, vol. 95, no. 4, pp. 2254–2263, 1994.

[11] M. Goto, "A real-time music-scene-description system: predominant-f0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Communication*, vol. 43, no. 4, pp. 311–329, 2004.

[12] C. Yeh, A. Röbel, and X. Rodet, "Multiple fundamental frequency estimation of polyphonic music signals," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 225–228, 2005.

[13] A. Pertusa and J. M. Inesta, "Multiple fundamental frequency estimation using Gaussian smoothness," in *Proc. IEEE International Conference on Acoustics, Speech Signal Processing (ICASSP)*, pp. 105–108, 2008.

[14] A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," in *Proc. ISMIR*, pp. 216–221, 2006.

[15] V. Emiya, R. Badeau, and B. David, "Multipitch estimation of quasi-harmonic sounds in colored noise," in *Proc. International Conference on Digital Audio Effects (DAFx)*, 2007.

[16] S. Saito, H. Kameoka, K. Takahashi, T. Nishimoto, and S. Sagayama, "Specmurt analysis of polyphonic music signals," *IEEE Trans. Speech Audio Processing*, vol. 16, no. 3, pp. 639–650, 2008.

[17] Z. Duan, B. Pardo, and C. Zhang, "Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions," *IEEE Trans. Audio Speech Language Processing*, vol. 18, no. 8, pp. 2121–2133, 2010.

[18] M. Wu, D. Wang, and G. J. Brown, "A multipitch tracking algorithm for noisy speech," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 3, pp. 229–241, 2003.

[19] F. Sha and L. Saul, "Real-time pitch determination of one or more voices by nonnegative matrix factorization," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, pp. 1233–1240, 2005.

[20] F. Bach and M. Jordan, "Discriminative training of hidden Markov models for multiple pitch tracking," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 489–492, 2005.

[21] Z. Jin and D. Wang, "HMM-based multipitch tracking for noisy and reverberant speech," *IEEE Trans. Audio Speech Language Processing*, vol. 19, no. 5, pp. 1091–1102, 2011.

[22] M. Ryynanen and A. Klapuri, "Polyphonic music transcription using note event modeling," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 319–322, 2005.

[23] G. E. Poliner and D. P. W. Ellis, "A discriminative model for polyphonic piano transcription," in *EURASIP Journal on Advances in Signal Processing*, 2007.

[24] H. Kameoka, T. Nishimoto, and S. Sagayama, "A multipitch analyzer based on harmonic temporal structured clustering," *IEEE Trans. on Audio Speech and Language Processing*, vol. 15, no. 3, pp. 982–994, 2007.

[25] W.-C. Chang, A. W. Y. Su, C. Yeh, A. Robel, and X. Rodet, "Multiple-f0 tracking based on a high-order HMM model," in *Proc. International Conference on Digital Audio Effects (DAFx)*, 2008.

[26] J. Le Roux, H. Kameoka, N. Ono, A. de Cheveigne, and S. Sagayama, "Single and multiple f0 contour estimation through parametric spectrogram modeling of speech in noisy environments," *IEEE Trans. Audio Speech Language Processing*, vol. 15, no. 4, pp. 1135–1145, 2007.

[27] Z. Duan, J. Han, and B. Pardo, "Song-level multi-pitch tracking by heavily constrained clustering," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 57–60, 2010.

[28] K. Kashino and H. Murase, "A sound source identification system for ensemble music based on template adaptation and music stream extraction," *Speech Communication*, pp. 337–349, 1999.

[29] E. Vincent, "Musical source separation using time-frequency source priors," *IEEE Trans. Audio Speech Language Processing*, vol. 14, no. 1, pp. 91–98, 2006.

[30] M. Bay, A. F. Ehmann, J. W. Beauchamp, P. Smaragdis, and J. S. Downie, "Second fiddle is important too: pitch tracking individual voices in polyphonic music," in *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pp. 319–324, 2012.

[31] M. Wohlmayr, M. Stark, and F. Pernkopf, "A probabilistic interaction model for multipitch tracking with factorial hidden Markov models," *IEEE Trans. Audio Speech Language Processing*, vol. 19, no. 4, pp. 799–810, 2011.

[32] K. Hu and D. Wang, "An unsupervised approach to cochannel speech separation," *IEEE Trans. Audio Speech Language Processing*, vol. 21, no. 1, pp. 122–131, 2013.

[33] A. Bregman, *Auditory Scene Analysis: The Perceptual Organization of Sound*. Cambridge, Massachusetts: The MIT Press, 1990.

[34] R. J. McAulay and T. F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 34, no. 4, pp. 744–754, 1986.

[35] P. Depalle, G. Garcia, and X. Rodet, "Tracking of partials for additive sound synthesis using hidden markov models," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 225–228, 1993.

[36] M. Lagrange and G. Tzanetakis, "Sound source tracking and formation using normalized cuts," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 61–64, 2007.

[37] K. Wagstaff and C. Cardie, "Clustering with instance-level constraints," in *Proc. International Conference on Machine Learning (ICML)*, pp. 1103–1110, 2000.

[38] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl, "Constrained k-means clustering with background knowledge," in *Proc. International Conference on Machine Learning (ICML)*, pp. 577–584, 2001.

[39] I. Davidson, S. Ravi, and M. Ester, "Efficient incremental constrained clustering," in *Proc. ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 240–249, 2007.

[40] A. Klapuri, "Analysis of musical instrument sounds by source-filter-decay model," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 53–56, 2007.

[41] J. J. Burred, A. Röbel, and T. Sikora, "Dynamic spectral envelope modeling for timbre analysis of musical instrument sounds," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 663–674, 2010.

[42] D. P. W. Ellis, "PLP and RASTA (and MFCC, and inversion) in Matlab," 2005. online web resource.

[43] Z. Duan and B. Pardo, "Soundprism: an online system for score-informed source separation of music audio," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1205–1215, 2011.

[44] T. Galas and X. Rodet, "An improved cepstral method for deconvolution of source-filter systems with discrete spectra: Application to musical sounds," in *Proc. of International Computer Music Conference (ICMC)*, pp. 82–84, 1990.

[45] D. Schwarz, "Spectral envelopes in sound analysis and synthesis," Master's thesis, Universität Stuttgart Fakultät Informatik, 1998.

[46] O. Cappé, J. Laroche, and E. Moulines, "Regularized estimation of cepstrum envelope from discrete frequency points," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 1995.

[47] A. de Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *Journal of the Acoustical Society of America*, vol. 111, pp. 1917–1930, 2002.

[48] G. Pirker, M. Wohlmayr, S. Petrik, and F. Pernkopf, "A pitch tracking corpus with evaluation on multipitch tracking scenario," in *Proc. Interspeech*, pp. 1509–1512, 2011.

[49] J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, and N. Dahlgren, "The DARPA TIMIT acoustic-phonetic continuous speech corpus cdrom." NTIS, order number PB01-100354, 1993. now available from LDC.

[50] P. Boersma, "Praat, a system for doing phonetics by computer," *Glot International*, vol. 5, no. 9/10, pp. 341–345, 2001.

[51] G. Hu and D. Wang, "A tandem algorithm for pitch estimation and voiced speech segregation," *IEEE Trans. Audio Speech Language Processing*, vol. 18, no. 8, pp. 2067–2079, 2010.