SPECTRAL LEARNING FOR EXPRESSIVE INTERACTIVE ENSEMBLE MUSIC PERFORMANCE

Guangyu Xia **Yun Wang** **Roger Dannenberg** **Geoffrey Gordon**
School of Computer Science, Carnegie Mellon University, USA
{gxia,yunwang,rbd,ggordon}@cs.cmu.edu

ABSTRACT

We apply machine learning to a database of recorded ensemble performances to build an artificial performer that can perform music expressively in concert with human musicians. We consider the piano duet scenario and focus on the interaction of expressive timing and dynamics. We model different performers' musical expression as co-evolving time series and learn their interactive relationship from multiple rehearsals. In particular, we use a spectral method, which is able to learn the correspondence not only between different performers but also between the performance past and future by reduced-rank partial regressions. We describe our model that captures the intrinsic interactive relationship between different performers, present the spectral learning procedure, and show that the spectral learning algorithm is able to generate a more human-like interaction.

1. INTRODUCTION

Ensemble musicians achieve shared musical interpretations when performing together. Each musician performs expressively, deviating from a mechanical rendition of the music notation along the dimensions of pitch, duration, tempo, onset times, and others. While creating this musical interpretation, musicians in an ensemble must listen to other interpretations and work to achieve an organic, coordinated whole. For example, expressive timing deviations by each member of the ensemble are constrained by the overall necessity of ensemble synchronization. In practice, it is almost impossible to achieve satisfactory interpretations on the first performance. Therefore, musicians spend time in rehearsal to become familiar with the interpretation of each other while setting the "communication protocols" of musical expression. For example, when should each musician play rubato, and when should each keep a steady beat? What is the desired trend and balance of dynamics? It is important to notice that these protocols are usually complex and implicit in the sense that they are hard to express via explicit rules. (Musicians in a large ensemble even need a conductor to help set the protocols.) However, musicians are able to learn these protocols very effectively. After a few rehearsals, they are prepared to handle new situations that do not even occur in rehearsals, which indicates that the learning procedure goes beyond mere memorization.

Although many studies have been done on musical expression in solo pieces, the analysis of interactive ensemble music performance is relatively new and has mainly focused on mechanisms used for synchronization, including gesture. Ensemble human-computer interaction is still out of the scope of most *expressive performance* studies, and the interaction between synchronization and individual expressivity is poorly understood. From the synthesis perspective, though *score following and automatic accompaniment* have been practiced for decades, many researchers still refer to this as the "score following" problem, as if all timing and performance information derives from the (human) soloist and there is no performance problem. Even the term "automatic accompaniment" diminishes the complex collaborative role of performers playing together by suggesting that the (human) soloist is primary and the (computer) accompanist is secondary. In professional settings, even piano accompaniment is usually referred to as "collaborative piano" to highlight its importance. To successfully synthesize interactive music performance, all performers should be equal with respect to musical expression, including the artificial performers.

Thus, there is a large gap between music practice and computer music research on the topic of expressive interactive ensemble music performance. We aim to address this gap by mimicking human rehearsals, i.e., learn the communication protocols of musical expression from rehearsal data. For this paper, we consider the piano duet scenario and focus on the interaction of expressive timing and dynamics. In other words, our goal is to build an artificial pianist that can interact with a human pianist expressively, and is capable of responding to the musical nuance of the human pianist.

To build the artificial pianist, we first model different performers' musical expression as co-evolving time series and design a function approximation to reveal the interactive relationship between the two pianists. In particular, we assume musical expression is related to hidden mental states and characterize the piano duet performance as a *linear dynamic system* (LDS). Second, we learn the parameters of the LDS from multiple rehearsals using a spectral method. Third, given the learned parameters, the artificial pianist can generate an expressive performance by interacting with a human pianist. Finally, we conduct evaluation by comparing the computer-generated performances with human performances. At the same time, we

inspect how training set size and the performer's style affect the results.

The next section presents related work. Section 3 describes the model. Section 4 describes a spectral learning procedure. Section 5 shows the experimental results.

2. RELATED WORK

The related work comes from three different research fields: *Expressive Performance*, where we see the same focus of musical expression; *Automatic Accompaniment*, where we see the same application of human-computer interactive performance; and *Music Psychology*, where we see musicology insights and use them to help design better computational models. For detailed historical reviews of expressive performance and automatic accompaniment, we point the readers to [14] and [27], respectively. Here, we only review recent work that has strong connections to probabilistic modeling.

2.1 Expressive Performance

Expressive performance studies how to automatically render a musical performance based on a static score. To achieve this goal, probabilistic approaches learn the conditional distribution of the performance given the score, and then generate new performances by sampling from the learned models. Grindlay and Helmbold [9] use *hidden Markov models* (HMM) and learn the parameters by a modified version of the Expectation-Maximization algorithm. Kim et al. [13] use a *conditional random field* (CRF) and learn the parameters by stochastic gradient descent. Most recently, Flossmann et al. [7] use a very straightforward linear Gaussian model to generate the musical expression of every note independently, and then use a modification of the Viterbi algorithm to achieve a smoother global performance.

All these studies successfully incorporate musical expression with time-series models, which serve as good bases for our work. Notice that our work considers not only the relationship between score and performance but also the interaction between different performers. From an optimization point of view, these works aim to optimize a performance given a score, while our work aims to solve this optimization problem under the constraints created by the performance of other musicians. Also, we are dealing with a real-time scenario that does not allow any backward smoothing.

2.2 Automatic Accompaniment

Given a pre-defined score, automatic accompaniment systems follow human performance in real time and output the accompaniment by strictly following human's tempo. Among them, Raphael's Music Plus One [19] and IRCAM's AnteScofo system [5] are very relevant to our work in the sense that they both use computational models to characterize the expressive timing of human musicians. However, the goal is still limited to temporal synchronization; the computer's musical expression in interactive performance is not yet considered.

2.3 Music Psychology

Most related work in Music Psychology, referred to as sensorimotor synchronization (SMS) and entrainment, studies adaptive timing behavior. Generally, these works try to discover common performance patterns and high-level descriptive models that could be connected with underlying brain mechanisms. (See Keller's book chapter [11] for a comprehensive overview.) Though the discovered statistics and models are not "generative" and hence cannot be directly adopted to synthesize artificial performances, we can gain much musicology insight from their discoveries to design our computational models.

SMS studies how musicians tap or play the piano by following machine generated beats [15-18, 21, 25]. In most cases, the tempo curve of the machine is pre-defined and the focus is on how humans keep track of different tempo changes. Among them, Repp, Keller [21] and Mates [18] argue that adaptive timing requires error correction processes and use a "phase/period correction" model to fit the timing error. The experiments show that the error correction process can be decoupled into period correction (larger scale tempo change) and phase correction (local timing adjustment). This discovery suggests that it is possible to predict timing errors based on timing features on different scales.

Compared to SMS, entrainment studies consider more realistic and difficult two-way interactive rhythmic processes [1, 8, 10-11, 20, 22, 26]. Among them, Goebl [8] investigated the influences of audio feedback in a piano duet setting and claims that there exist bidirectional adjustments during full feedback despite the leader/follower instruction. Repp [20] does further analysis and discovers that the timing errors are auto-correlated and that how much musicians adapt to each other depends on the music context, such as melody and rhythm. Keller [11] claims that entrainment not only results in coordination of sounds and movements, but also of mental states. These arguments suggest that it is possible to predict the timing errors (and other musical expressions) by regressions based on different music contexts, and that hidden variables can be introduced to represent mental states.

3. MODEL SPECIFICATION

3.1 Linear Dynamic System (LDS)

We use a linear dynamic system (LDS), as shown in Figure 1, to characterize the interactive relationship between the two performers in the expressive piano duet. Here, $Y = [y_1, y_2, \dots, y_T]$ denotes the 2nd piano's musical expression, $U = [u_1, u_2, \dots, u_T]$ denotes a combination of the 1st piano's musical expression and score information, and $Z = [z_1, z_2, \dots, z_T]$ denotes the *hidden* mental states of the 2nd pianist that influence the performance. The key

idea is to reveal that the 2nd piano's musical expression is not static. It is not only influenced by the 1st piano's performance but also keeps its own character and continuity over time.

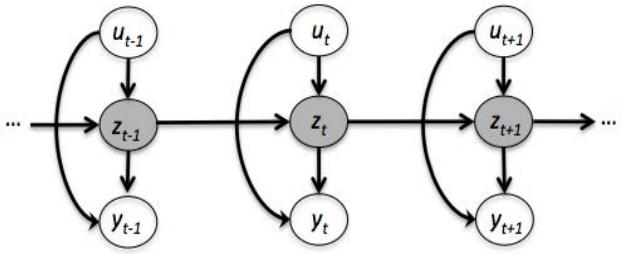

Figure 1. The graphical representation of the LDS, in which grey nodes represent hidden variables.

Formally, the evolution of the LDS is described by the following linear equations:

$$z_t = Az_{t-1} + Bu_t + w_t \quad w_t \sim \mathcal{N}(0, Q) \tag{1}$$

$$y_t = Cz_t + Du_t + v_t \quad v_t \sim \mathcal{N}(0, R) \tag{2}$$

Here, $y_t \in \mathbb{R}^2$ and its two dimensions correspond to expressive timing and dynamics, respectively, $u_t \in \mathbb{R}^l$, which is a much higher dimensional vector (we describe the design of u_t in detail in Section 3.3), and $z_t \in \mathbb{R}^n$, which is a relatively lower dimensional vector. *A, B, C,* and *D* are the main parameters of the LDS. Once they are learned, we can predict the performance of the 2nd piano based on the performance of the 1st piano.

3.2 Performance Sampling

Notice that the LDS is indexed by the discrete variable *t*. One question arises: should *t* represent note index or score time? Inspired by Todd's work [23], we assume that musical expression evolves with score time rather than note indices, and therefore define *t* as score time. Since music notes have different durations, we "sample" the performed notes (of both the 1st piano and the 2nd piano) at the resolution of a half beat, as shown in Figure 2.

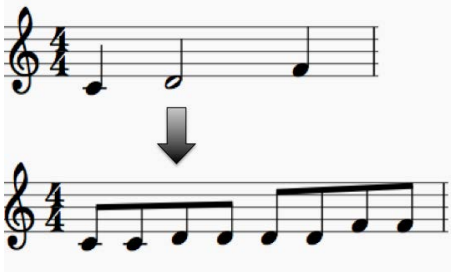

Figure 2. An illustration of performance sampling.

To be more specific, if a note's starting time aligns with a half beat and its *inter-onset-interval* (IOI) is equal to or greater than one beat, we replace the note by a series of eighth notes, each having the same pitch, dynamic, and duration-to-IOI ratio as the original note. Note that we still play the notes as originally written; the sampled representation is only for learning and prediction.

3.3 Input Features Design

To show the design of u_t, we introduce an auxiliary notation $X = [x_1, x_2, \ldots, x_T]$ to denote the raw score information and musical expression of the 1st piano and describe the mapping from X to each component of u_t in rest of this section. Note that u_t is based on sampled score and performance.

3.3.1 Score Features

High Pitch Contour: For the chords within a certain time window up to and including *t*, extract the highest-pitch notes and fit the pitches by a quadratic curve. Then, *high pitch contour* for *t* is defined as the coefficients of the curve. Formally:

$$\widehat{\beta}_t^{high} \stackrel{\text{def}}{=} \underset{\beta}{\operatorname{argmin}} \sum_{i=0}^{p} \left(x_{t-p+i}^{highpitch} - quad_\beta(t-p+i) \right)^2$$

where p is a context length parameter and $quad_\beta$ is the quadratic function parameterized by β.

Low Pitch Contour: Similar to *high pitch contour*, we compute $\widehat{\beta}_t^{low}$ for *low pitch contour*.

Beat Phase: The relative location of *t* within a measure. Formally:

$$\text{BeatPhase}_t \stackrel{\text{def}}{=} (t \bmod \text{MeasureLen})/\text{MeasureLen}$$

3.3.2 The 1st Piano Performance Features

Tempo Context: Tempi of the *p* closest notes directly before *t*. This is a timing feature on a relatively large time scale. Formally:

$$\text{TempoContext}_t \stackrel{\text{def}}{=} \left[x_{t-p}^{\text{Tempo}}, x_{t-p+1}^{\text{Tempo}}, \cdots, x_{t-1}^{\text{Tempo}} \right]^T$$

Here, the tempo of a note is defined as the slope of the least-squares linear regression between the performance onsets and the score onsets of *q* preceding notes.

Onsets Deviation Context: A description of how much the *p* closest notes' onsets deviate from their tempo curves. Compared to the tempo context, this is a timing feature on a relatively small scale. Formally:

$$\text{OnsetsDeviationContext}_t \stackrel{\text{def}}{=} \left[x_{t-p}^{\text{OnsetsDeviation}}, x_{t-p+1}^{\text{OnsetsDeviation}}, \cdots, x_{t-1}^{\text{OnsetsDeviation}} \right]^T$$

Duration Context: Durations of the *p* closest notes directly before *t*. Formally:

$$\text{DurationContext}_t \stackrel{\text{def}}{=} \left[x_{t-p}^{\text{Dur}}, x_{t-p+1}^{\text{Dur}}, \cdots, x_{t-1}^{\text{Dur}} \right]^T$$

Dynamic Context: MIDI velocities of the *p* closest notes directly before *t*. Formally:

$$\text{DynamicContext}_t \stackrel{\text{def}}{=} \left[x_{t-p}^{\text{Vel}}, x_{t-p+1}^{\text{Vel}}, \cdots, x_{t-1}^{\text{Vel}} \right]^T$$

The input feature, u_t, is a concatenation of the above features. We have also tried other features and mappings (e.g., rhythm context, phrase location, and down beat),

and finally picked the ones above through experimentation.

4. SPECTRAL LEARNING PROCEDURE

To learn the model, we use a spectral method, which is rooted in control theory [24] and then further developed in the machine learning field [2]. Spectral methods have proved to be both fast and effective in many applications [3][4]. Generally speaking, a spectral method learns hidden states by predicting the performance future from features of the past, but forcing this prediction to go through a low-rank bottleneck. In this section, we present the main learning procedure with some underlying intuitions, using the notation of Section 3.1.

Step 0: **Construction of Hankel matrices**

We learn the model in parallel for fast computation. In order to describe the learning procedure more concisely, we need some auxiliary notations. For any time series $S = [s_1, s_2, \dots, s_T]$, the "history" and "future" Hankel matrices are defined as follows:

$$S_H \stackrel{\text{def}}{=} \begin{pmatrix} s_1 & \cdots & s_{T-d} \\ \vdots & \ddots & \vdots \\ s_{\frac{d}{2}} & \cdots & s_{T-\frac{d}{2}-1} \end{pmatrix}, S_F \stackrel{\text{def}}{=} \begin{pmatrix} s_{\frac{d}{2}+1} & \cdots & s_{T-\frac{d}{2}} \\ \vdots & \ddots & \vdots \\ s_d & \cdots & s_{T-1} \end{pmatrix}$$

Also, the "one-step-extended future" and "one-step-shifted future" Hankel matrices are defined as follows:

$$S_F^+ \stackrel{\text{def}}{=} \begin{pmatrix} s_{\frac{d}{2}+1} & \cdots & s_{T-\frac{d}{2}} \\ \vdots & \ddots & \vdots \\ s_{d+1} & \cdots & s_T \end{pmatrix}, S_F^S \stackrel{\text{def}}{=} \begin{pmatrix} s_{\frac{d}{2}+2} & \cdots & s_{T-\frac{d}{2}+1} \\ \vdots & \ddots & \vdots \\ s_{d+1} & \cdots & s_T \end{pmatrix}$$

Here, d is an even integer indicating the size of a sliding window. Note that corresponding columns of S_H and S_F are "history-future" pairs within sliding windows of size d; compared with S_F^+, S_F^S is just missing the first row. We will use the Hankel matrices of both U and Y in the following steps.

Step 1: **Oblique projections**

If the true model is LDS, i.e., everything is linear Gaussian, the expected future observations can be expressed linearly by history observations, history inputs, and future inputs. Formally:

$$\mathbb{E}(Y_F | Y_H, U_H, U_F) = [\beta_{Y_H}\ \beta_{U_H}\ \beta_{U_F}] \begin{bmatrix} Y_H \\ U_H \\ U_F \end{bmatrix} \quad (3)$$

Here, $\beta = [\beta_{Y_H}\ \beta_{U_H}\ \beta_{U_F}]$ is the linear coefficient that could be solved by:

$$\hat{\beta} = [\hat{\beta}_{Y_H}\ \hat{\beta}_{U_H}\ \hat{\beta}_{U_F}] = Y_F \begin{bmatrix} Y_H \\ U_H \\ U_F \end{bmatrix}^{\dagger} \quad (4)$$

where † denotes the Moore-Penrose pseudo-inverse. However, since in a real-time scenario the future input, U_F, is unknown, we can only partially explain future observations based on the history. In other words, we care about the best estimation of future observations but just based on the history observations and inputs. Formally:

$$\hat{O}_F \stackrel{\text{def}}{=} \hat{\beta}_H \begin{bmatrix} Y_H \\ U_H \\ 0 \end{bmatrix} = [\hat{\beta}_{Y_H}\ \hat{\beta}_{U_H}\ 0] \begin{bmatrix} Y_H \\ U_H \\ 0 \end{bmatrix} \quad (5)$$

where $\hat{O}_F$ is referred to as the oblique projection of Y_F "along" U_F and "onto" $\begin{bmatrix} Y_H \\ U_H \end{bmatrix}$. In this step, we also use the same technique to compute $\hat{O}_F^+$ and just throw out its first row to obtain $\hat{O}_F^S$.

Step 2: State estimation by singular value decomposition (SVD)

If we knew the true parameters of the LDS, the oblique projections and the hidden states would have the following relationship:

$$\hat{O}_F = \Gamma_f Z_f \stackrel{\text{def}}{=} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{\frac{d}{2}-1} \end{bmatrix} \left[z_{\frac{d}{2}+1}, z_{\frac{d}{2}+2}, \dots, z_{T-\frac{d}{2}} \right] \quad (6)$$

$$\hat{O}_F^S = \Gamma_f Z_f^S \stackrel{\text{def}}{=} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{\frac{d}{2}-1} \end{bmatrix} \left[z_{\frac{d}{2}+2}, z_{\frac{d}{2}+3}, \dots, z_{T-\frac{d}{2}+1} \right] \quad (7)$$

Intuitively, the information from the history observations and inputs "concentrate" on the nearest future hidden state and then spread out onto future observations. Therefore, if we perform SVD on the oblique projections and throw out small singular values, we essentially enforce a bottleneck on the graphical model representation, learning compact, low-dimensional states. Formally, let

$$\hat{O}_F = \mathcal{U} \Lambda \mathcal{V}^T \quad (8)$$

and delete small numbers in Λ and corresponding columns in $\mathcal{U}$ and $\mathcal{V}$. Since LDS is defined up to a linear transformation, we could estimate the hidden states by:

$$\Gamma_f = \mathcal{U} \Lambda^{\frac{1}{2}} \quad (9)$$

$$\hat{Z}_f = \Gamma_f^{\dagger} \hat{O}_F \quad (10)$$

$$\hat{Z}_f^S = \Gamma_f^{\dagger} \hat{O}_F^S \quad (11)$$

Step 3: Parameter estimation

Once we have estimated the hidden states, the parameters can be estimated from the following two equations:

$$\hat{Z}_f^S = A \hat{Z}_f + B U_f^S + e_w \quad (12)$$

$$Y_f = C \hat{Z}_f + D U_f + e_v \quad (13)$$

Here, Y_f and U_f are the 1st rows of Y_F and U_F, i.e., $Y_f = \left[y_{\frac{d}{2}+1}, y_{\frac{d}{2}+2}, \dots, y_{T-\frac{d}{2}} \right]$, $U_f = \left[u_{\frac{d}{2}+1}, u_{\frac{d}{2}+2}, \dots, u_{T-\frac{d}{2}} \right]$. Similarly, U_f^S is the 1st row of U_F^S, i.e., $U_f^S = \left[u_{\frac{d}{2}+2}, u_{\frac{d}{2}+3}, \dots, u_{T-\frac{d}{2}+1} \right]$.

In summary, the spectral method does three regressions. The first two estimate the hidden states by oblique projections and SVD. The third one estimates the parame-

ters. The oblique projections can be seen as de-noising the latent states by using past observations, while the SVD adds low-rank constraints. As opposed to maximum likelihood estimation (MLE), the spectral method is a method-of-moments estimator that does not need any random initialization or iterations. Also note that we are making a number of arbitrary choices here (e.g., using equal window sizes for history and future), not attempting to give a full description of how to use spectral methods. (See Van Overschee & De Moor's book [24] for the details and variations of the learning methods.)

5. EXPERIMENTS

5.1 Dataset

We created a dataset [27] that contains three piano duets: *Danny Boy*, *Serenade* (by Schubert), and *Ashokan Farewell*. All pieces are in MIDI format and contain two parts: a monophonic 1st piano part and a polyphonic 2nd piano part. Each piece is performed 35 to 42 times in different musical interpretations by 5 to 6 pairs of musicians. (Each pair performs each piece of music 7 times.)

5.2 Methods for Comparison

We use three methods for comparison: linear regression, neural network, and the timing estimation often used in automatic accompaniment systems [6]. The first two methods use the same set of features as in the spectral methods, while the 3rd method does not contain any learning procedure and is considered as the baseline.

Linear regression: Referring to the notation in Section 3, the linear regression method simply solves the following equation:

$$Y = \beta U \tag{14}$$

Like the LDS, this method uses the performance of 1st piano to estimate that of the 2nd piano, but it does not use any hidden states or attempt to enforce self-consistency in the musical expression of the 2nd pianist's performance.

Neural network: We use a simple neural network with a single hidden layer. The hidden layer consists of 10 neurons and uses rectified linear units (ReLUs) to produce non-linearity; the single output neuron is linear. Denoting the activation of the hidden units by Z, the neural network represents the following relationship between U and Y:

$$Z = f(W_1 U + b_1) \tag{15}$$

$$Y = W_2 Z + b_2 \tag{16}$$

where

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{17}$$

The neural network is trained by the minibatch stochastic gradient descent (SGD) algorithm, using the mean absolute error as the cost function. The parameters of the neural network (W_1, b_1, W_2, b_2) are initialized randomly, after which they are tuned with 30 epochs of SGD. Each minibatch consists of one rehearsal. The learning rate decays from 0.1 to 0.05 in an exponential fashion during the training. We report the average absolute and relative errors across five runs with different random initializations on the test set.

This method can be seen as an attempt to improve the linear regression method using non-linear function approximation, but it also doesn't consider the self-consistency in the musical expression of the 2nd pianist's performance.

Baseline: The baseline method assumes that local tempo and dynamics are stable. For timing, it estimates a linear mapping between real time and score time by fitting a straight line to 4 recently performed note onsets of the 1st piano. This mapping is then used to estimate the timing of the next note of the 2nd piano. For dynamics, it uses the dynamics of the last performed note of the 1st piano as the estimator.

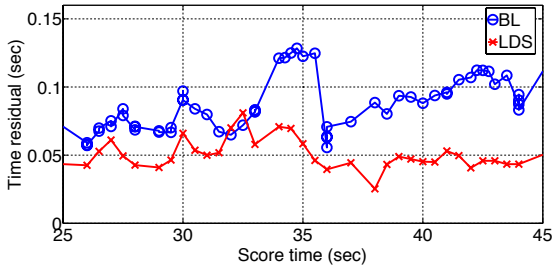

Figure 3. A local view of the absolute timing residuals of the LDS approach.

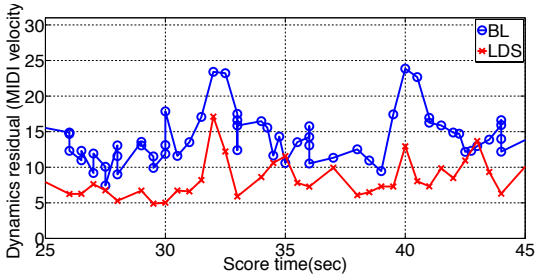

Figure 4. A local view of the absolute dynamics residuals of the LDS approach.

5.3 A Local View of the LDS Method

Figure 3 and Figure 4 show a local view of the expressive timing and dynamics cross-validation result, respectively, for *Danny Boy*. (To have a clear view, we just compare LDS with the baseline here. We show the results of all the methods on all the pieces later.) For both figures, the *x*-axis represents score time and the *y*-axis represents absolute residual between the prediction and human performance. Therefore, small numbers mean better results. The curve with circle markers represents the baseline approach, while the curve with "x" markers represents the LDS approach trained with only 4 randomly selected rehearsals of the *same* piece performed by *other* performers. We can see that the LDS approach performs much

better than the baseline approach with only 4 training rehearsals, which indicates that the algorithm is both accurate and robust.

5.4 A Global View of All Methods

The curves in the previous two figures are a measurement over different performances. If we average the absolute residual across an entire piece of music, we get a single number that describes a method's performance for that piece. I.e., how much on average is the prediction of a method different from the human performance for each note? Figure 5 and Figure 6 show this average absolute residual for timing and dynamics, respectively, for all the methods and pieces combinations with different training set sizes.

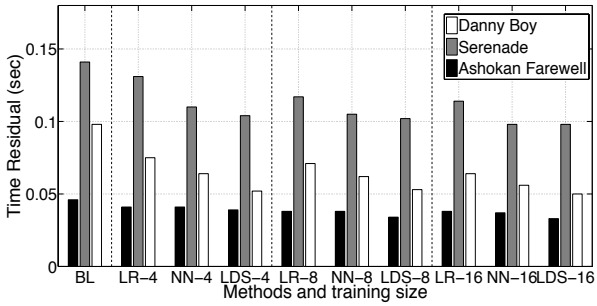

Figure 5. A global view of absolute timing residuals for all pieces and methods. (Smaller is better.)

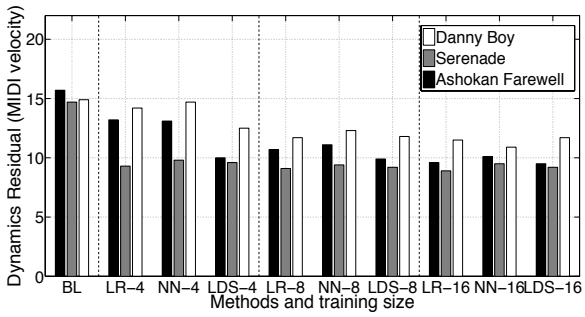

Figure 6. A global view of absolute dynamics residuals for all pieces and methods. (Smaller is better.)

In both figures, the *x*-axis represents different methods with different training set sizes, the *y*-axis represents the average absolute residual, and different colors represent different pieces. For example, the grey bar above the label "NN-4" in Figure 5 is the average absolute timing residual for *Serenade* by using the neural network approach with 4 training rehearsals.

We see that for expressive timing, both neural network and LDS outperform simple linear regression, and the LDS performs the best regardless of the music piece or training set size. This indicates that the constraint of preceding notes (self-consistency) captured by LDS is playing an important role in timing prediction. For expressive dynamics, the difference between different methods is less significant. We see no benefit by using a neural network. But when the training set size is small, LDS still outperforms linear regression. (Which is quite interesting because LDS learns more parameters than linear regression.)

5.5 Performer's Effect

Finally, we inspect whether there is any gain by training a performer-specific model. In other words, we only learn from the rehearsals performed by the *same* pair of musicians. Since each pair of musicians only performs 7 times for each piece, we randomly choose 4 from the 7 performances to make a fair comparison against the results in Figure 5 and Figure 6.

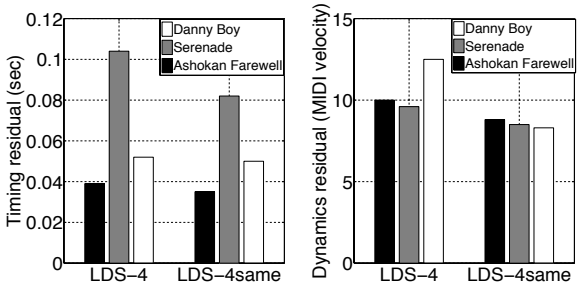

Figure 7. A global view of the performer-specific model.

Figure 7 shows a comparison between performer-specific model and different-performer model. In both sub-graphs, the bars above "LDS-4same" are the results for performer-specific model, while the bars above "LDS-4" are the same as in Figure 5 and Figure 6. Note that they are both cross-validation results and the only difference is the training set. We see that the performer-specific model achieves better results, especially when the different-performer model is not doing a good job.

6. CONCLUSIONS AND FUTURE WORK

In conclusion, we have applied a spectral method to learn the interactive relationship in expressive piano duet performances from multiple rehearsals. Compared to other methods, we have made better predictions based on only 4 rehearsals, and we have been able to further improve the results using a performer-specific model. Our best model is able to shrink the timing residual by nearly 60 milliseconds and shrink the dynamic residual by about 8 MIDI velocity units compared to the baseline algorithm, especially when the baseline algorithm behaves poorly.

In the future, we would like to incorporate some non-linear function approximations with the current graphical representation of the model. An ideal case would be to combine the dynamical system with a neural network, which calls for new spectral learning algorithms. Also, we would like to be more thorough in the evaluations. Rather than just inspecting the absolute difference between computer-generated performance and human performances, we plan to also compare computed-generated results with typical variation in human performances and use subjective evaluation.

7. REFERENCES

[1] C. Bartlette, D. Headlam, M. Bocko, and G. Velikic, "Effect of Network Latency on Interactive Musical Performance," *Music Perception*, pp. 49–62, 2006.

[2] B. Boots, *Spectral Approaches to Learning Predictive Representations* (No. CMU-ML-12-108). Carnegie Mellon Univ., School of Computer Science, 2012.

[3] B. Boots and G. Gordon, "An Online Spectral Learning Algorithm for Partially Observable Nonlinear Dynamical Systems," *Proceedings of the National Conference on Artificial Intelligence*, 2011.

[4] B. Boots, S. Siddiqi, and G. Gordon, "Closing the Learning-planning Loop with Predictive State Representations," *The International Journal of Robotics Research*, pp. 954-966, 2011.

[5] A. Cont, "ANTESCOFO: Anticipatory Synchronization and Control of Interactive Parameters In Computer Music," *Proceedings of International Computer Music Conference*, pp. 33-40, 2011.

[6] R. Dannenberg, "An Online Algorithm for Real-Time Accompaniment," *Proceedings of the International Computer Music Conference*, pp. 193-198, 1984.

[7] S. Flossmann, M. Grachten, and G. Widmer, "Expressive Performance Rendering with Probabilistic Models," *Guide to Computing for Expressive Music Performance*, Springer, pp. 75–98, 2013.

[8] W. Goebl and C. Palmer, "Synchronization of Timing and Motion Among Performing Musicians," *Music Perception*, pp. 427– 438, 2009.

[9] G. Grindlay and D. Helmbold, "Modeling, Analyzing, and Synthesizing Expressive Piano Performance with Graphical Models," *Machine Learning*, pp. 361-387, 2006.

[10] M. Hove, M. Spivey, and L. Krumhansl, "Compatibility of Motion Facilitates Visuomotor Synchronization," *Journal of Experimental Psychology: Human Perception and Performance*, pp. 1525-1534, 2010.

[11] P. Keller, "Joint Action in Music Performances," *Enacting Intersubjectivity*: *A Cognitive and Social Perspective to the Study of Interactions* Amsterdan, The Netherlands: IOS Press, pp. 205-221, 2008.

[12] P. Keller, G. Knoblich, and B. Repp, "Pianists Duet Better When They Play with Themselves: On the Possible Role of Action Simulation in Synchronization," *Consciousness and Cognition*, pp. 102–111, 2007.

[13] T. Kim, F. Satoru, N. Takuya, and S. Shigeki, "Polyhymnia: An Automatic Piano Performance System with Statistical Modeling of Polyphonic Expression and Musical Symbol Interpretation," *Proceedings of the International Conference on New Interfaces for Musical Expression*, pp. 96-99, 2011.

[14] A. Kirke and E. R. Miranda, "A Survey of Computer Systems for Expressive Music Performance," *ACM Surveys* 42(1): Article 3, 2009.

[15] E. Large and J. Kolen, "Resonance and the Perception of Musical Meter". *Connection Science*, pp. 177–208, 1994.

[16] E. Large and C. Palmer, "Perceiving Temporal Regularity in Music," *Cognitive Science*, pp. 1–37, 2002.

[17] E. Large and C. Palmer, "Temporal Coordination and Adaptation to Rate Change in Music Performance," *Journal of Experimental Psychology: Human Perception and Performance*, pp. 1292-1309, 2011.

[18] J. Mates, "A Model of Synchronization of Motor Acts to a Stimulus Sequence: Timing and Error Correction," *Biological Cybernetics*, pp. 463– 473, 1994.

[19] C. Raphae, "Music Plus One and Machine Learning," *Proceedings of International Conference on Machine Learning*, pp. 21-28, 2010.

[20] B. Repp and P. Keller, "Sensorimotor Synchronization with Adaptively Timed Sequences," *Human Movement Science*, pp. 423-456, 2008.

[21] B. Repp and P. Keller, "Adaptation to Tempo Changes in Sensorimotor Synchronization: Effects of Intention, Attention, and Awareness," *Quarterly Journal of Experimental Psychology*, pp. 499-521, 2004.

[22] G. Schöner, "Timing, Clocks, and Dynamical Systems," *Brain and Cognition*, pp. 31-51, 2002.

[23] P. Todd, "A Connectionist Approach to Algorithmic Composition," *Computer Music Journal,* pp. 27-43, 1989.

[24] P. Van Overschee and B. De Moor, *Subspace Identification for Linear Systems: Theory, Implementation*, *applications*. Kluwer Academic Publishers, 1996.

[25] D. Vorberg and H. Schulze, "A Two-level Timing Model for Synchronization," *Journal of Mathematical Psychology*, pp. 56–87, 2002.

[26] A. Wing, "Voluntary Timing and Brain Function: an Information Processing Approach," *Brain and Cognition*, pp. 7-30, 2002.

[27] G. Xia and R. Dannenberg, "Duet Interaction: Learning Musicianship for Automatic Accompaniment," *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2015.