Multivariate Sound Info Retrievable from MIDI Music, Digital Photography, and Satellite Imagery: SAS to the Rescue

Cecil Hallum
Department of Mathematics and Statistics
Sam Houston State University
Huntsville, Texas

INTRODUCTION

Since 1982, a common source of digital music information has been the MIDI (Musical Instrumentation Digital Interface) music that is readily available from the Web. Sometime ago, myself and a graduate student (Cannon and Hallum (2004)) developed a SAS program that converts a MIDI music file from its raw "object code form" into a multivariate data set (i.e., vector-valued data set) comprised of instrument-specific sound frequencies (in hertz) for further manipulation by SAS or similar software; in doing so, a number of sound-related research questions could be pursued by taking advantage of the resulting linear algebraic framework. Interesting questions which surfaced beginning back in 2004 included the following:

1. "Is it possible to 'map' one MIDI tune 'effectively' to another; e.g., can a Bach tune be 'mapped' to a Beethoven tune, can a Hank Williams tune be 'mapped' to the National Anthem, etc.? Obviously such files possess a considerable amount of "interesting and worthwhile" sound information, and as a result, what are some key characteristics, if any, that might be retrievable based on a choice for a transform (e.g., simple matrix multiplication comes to mind immediately) to operate on such a file; plus, what can be learned and benefits gained, if any, from such questioning that could lend insight to a worthwhile spin-off or two"?

2. "Is it possible to identify 'equivalence classes' of tunes that could then be connected (algebraically at least), in some recognizable manner, back to a certain cadre of artists, to a certain music genre, to the Billboard top 20 over some window of time, etc.?"

3. Plus several others to be discussed at conference time.

Considerable discussion was given to address these questions in the Cannon and Hallum (2004) SCSUG (South Central SAS Users Group) paper with emphasis on using frequencies (in hertz) of play of the various instruments as the key initial MIDI data source. Further discussion is provided herein to re-visit these questions, to provide further insight and some further generalization to the transformation approach (at the expense of a little redundancy here and there); even more, a primary emphasis is given to identifying and demonstrating spin-offs from these efforts resulting largely from the insight gained from thinking from the perspective of multivariate MIDI algebraic mindset attained from these initial endeavors; specifically, such insight has lent itself to:

1. structuring a capability that will, eventually, provide the means to effectively assist a seeing-impaired individual "listen" to digital photography, "listen to the landscape", "listen" to the petals of a flower, "listen" to a sunrise, etc., and

2. identifying and demonstrating other usages of SAS to "control and/or manipulate sound" while resorting to its GUI (Graphical User Interface) capabilities to permit an ease of use for a general user.

As an example of the latter, a SAS GUI capability is discussed and demonstrated herein for the handling of interfaces between synthesizers, digital audio workstation software, and the storage and retrieval of music files from data bases containing hundreds of songs which the author uses as backup rhythm (and lead as well) in music practice sessions. All sound results will be amped at conference time to permit ease of listening. SAS/Basics, SAS/Statistics, SAS/IML, and SAS/Applications Development are utilized as the key SAS tools in the developments herein and in this presentation itself as well.

The key layout of a MIDI file after getting "unpackaged" from its initial "object code" form by SAS is simply that of a matrix whereby each row represents instrument-specific frequencies (in hertz) played by no more than a maximum of 16 instruments – so each matrix will have no more than 16 columns; its rows, however, will typically number in the thousands (one for each clock beat in time). So the first entry in a particular row might be the frequency (in hertz) of play of an acoustic guitar; the second entry might be the frequency of play of a piano, etc. Consequently, all of an artist's instrumental work (i.e., after being put into matrix form from the initial raw MIDI form) can be written as a collection of matrices Ω. The same can be said about any musician's instrumental work, any genre of instrumental music, etc., whereby the MIDI music is in digital (i.e., frequency) form after the aforementioned conversion. So Figure 1 represents the makeup of a given row (written here in column form) of the MIDI matrix after conversion to frequencies; of course, the instruments utilized will differ across differing tunes and across different rows as well.

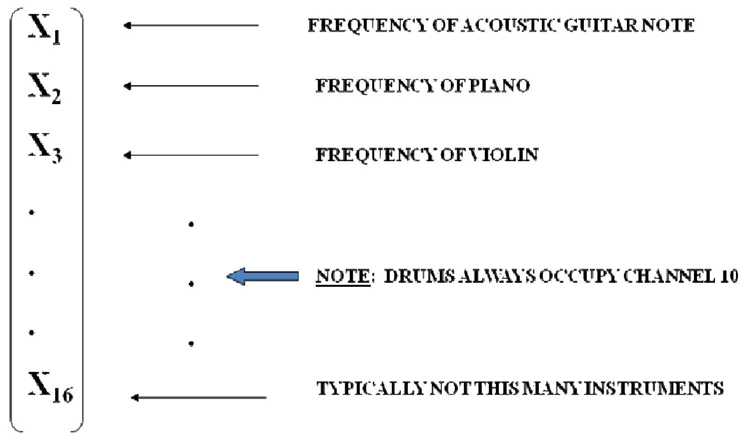

X_1
FREQUENCY OF ACOUSTIC GUITAR NOTE
X_2
FREQUENCY OF PIANO
X_3
FREQUENCY OF VIOLIN
NOTE: DRUMS ALWAYS OCCUPY CHANNEL 10
X_{16}
TYPICALLY NOT THIS MANY INSTRUMENTS

Figure 1: The Vector Layout of the MIDI Instrument Frequency Data

OBJECTIVES

A first objective is to re-visit the Cannon and Hallum (2004) results regarding the transformation derivation to clarify, generalize (a little) and re-visit some of the developmental details and how SAS came to the rescue here. A second objective is to discuss details of several additional "sound-based" spin-off applications that have resulted from what has been learned from the MIDI endeavors. Since the developed SAS AF frame modules provide a GUI approach to convert raw MIDI files to a format readable by SAS, its underlying structure will be reviewed herein in a bit more detail for clarification. Specifics of the SAS code that convert the frequencies (after transformations, manipulations, etc.) back into a raw MIDI file to hear the auditory information played back through a music synthesizer will be clarified for better understanding of the "reversal" logistics. Finally, based on insights gained from the MIDI experience, this second objective will include details regarding spin-off applications including:

- an approach, and a demonstration of its usage, to assist a seeing-impaired person "hear" digital information (e.g., from digital photography), and
- a SAS GUI approach for improving practice music sessions and handling large data bases of music files (including MIDI, MP3, WAVE, WMA, and other file types) and to include a choice, via the GUI, for the software module to "play" a selected file.

These items will be demonstrated in real-time at conference time. Of course, the following objectives are always at the forefront as well:

- identification of additional questions for future research by undergraduate and/or graduate students as part of their research papers, practicums, and theses;
- the opportunity to discuss such specifics with faculty colleagues (e.g., John, Marty, Clay, etc.....they know who they are!!) having an interest in music and related topics is an additional "music to my ears" objective!

SOME RELATED RESULTS FROM THE LITERATURE

The reader is advised to re-visit the references in the Cannon and Hallum (2004) paper once again since they remain pertinent to this discussion as well. The next several paragraphs comprise a literature look into the contributions to-date in the area of primarily <u>multivariate type</u> investigations of MIDI data related to the type of research interests discussed herein.

The Das, Howard, and Smith paper (1999) postulated that the analysis of multivariate MIDI data allows for the statistical analysis of motion in music. Their paper dealt with analysing the kinematic motion components within music, specifically music velocity (i.e. tempo) and music acceleration/deceleration (i.e. tempo change). They utilized differing multivariate variables (i.e., not the multivariate sound frequencies) than those investigated herein.

J. H. Jensen, Christensen, and S. H. Jensen (2004) tested the sensitivity of similarity measures to transpositions and concluded that it would also be relevant to measure the dependency on tempo, combinations of instruments, bandwidth and audio compression to get insightful supplement to genre classification. Continued research will be done to see ways and means of bringing such information in with the frequencies data retrieved and utilized herein.

Much work has already been done on music similarity and on the related task of genre classification, along with many other such concerns (check the additional references included herein). In particular, genre classification is often used to evaluate music similarity measures since it simplifies evaluation compared to the numerous user evaluations that are otherwise needed.

It should be noted that considerable additional data can be accessed after completion of the SAS MIDI "decoupling" process (i.e., more than just instrument frequencies can be retrieved) and which can be made available to support other research efforts. For example, the tempo, the volume with which each string, piano key, etc. is stroked (sometimes called the "attack"), the degree of vibrato, and other such information is also retrievable from the MIDI data as well. These can obviously support research efforts as well and several have been addressed already in various ways in a number of the papers listed in the reference section herein.

SAS GUI TO CONVERT A MIDI TO A MATRIX OF FREQUENCIES

The developed SAS GUI module that converts MIDI files into SAS readable form has its initial screen given in Figure 2. This AF screen for selecting a MIDI tune has been considerably enhanced since the 2004 SCSUG; the explanation of its usage is included here for completeness. To select a MIDI tune, the user must first select a category. Here "BLUES" has been selected, which automatically populated the entries in the "MIDI FILES" list box to only those tunes that are "blues" tunes. As one can see, the tune "MISSISSIPPIBLUES" is the MIDI selected from this menu. Indexing the "APPLY" pushbutton transforms the MIDI file into a text file. At this point the user may choose to view the MIDI text file in the newly transformed text form by clicking on "VIEW MIDI", which brings up the AF frame screen shown in Figure 3. The file that is executed by the SAS GUI at this point is the one that converts the MIDI from its object code form into a text file, plus it then separates out all text parts from all numerical information. To see SAS code, click on the following address:

http://www.shsu.edu/~mth_crh/SCSUG/experiment_try.sas.

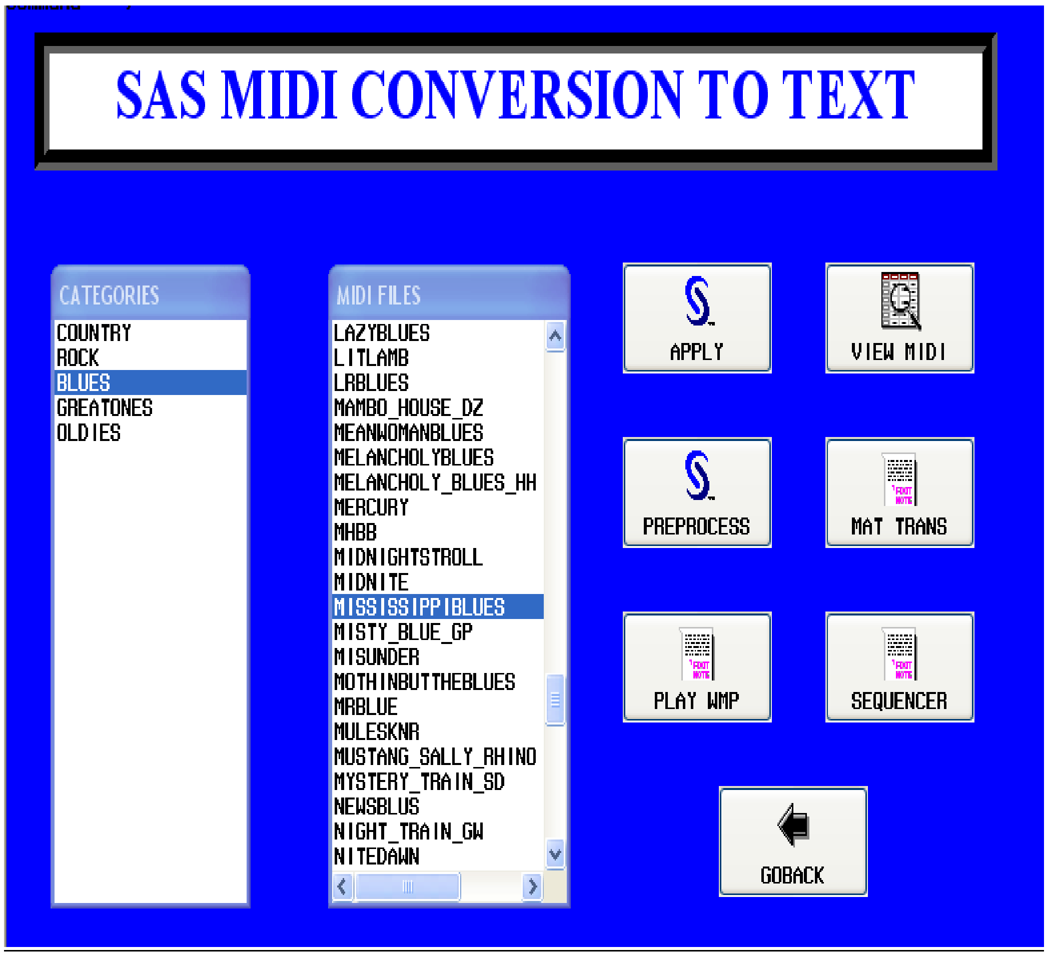
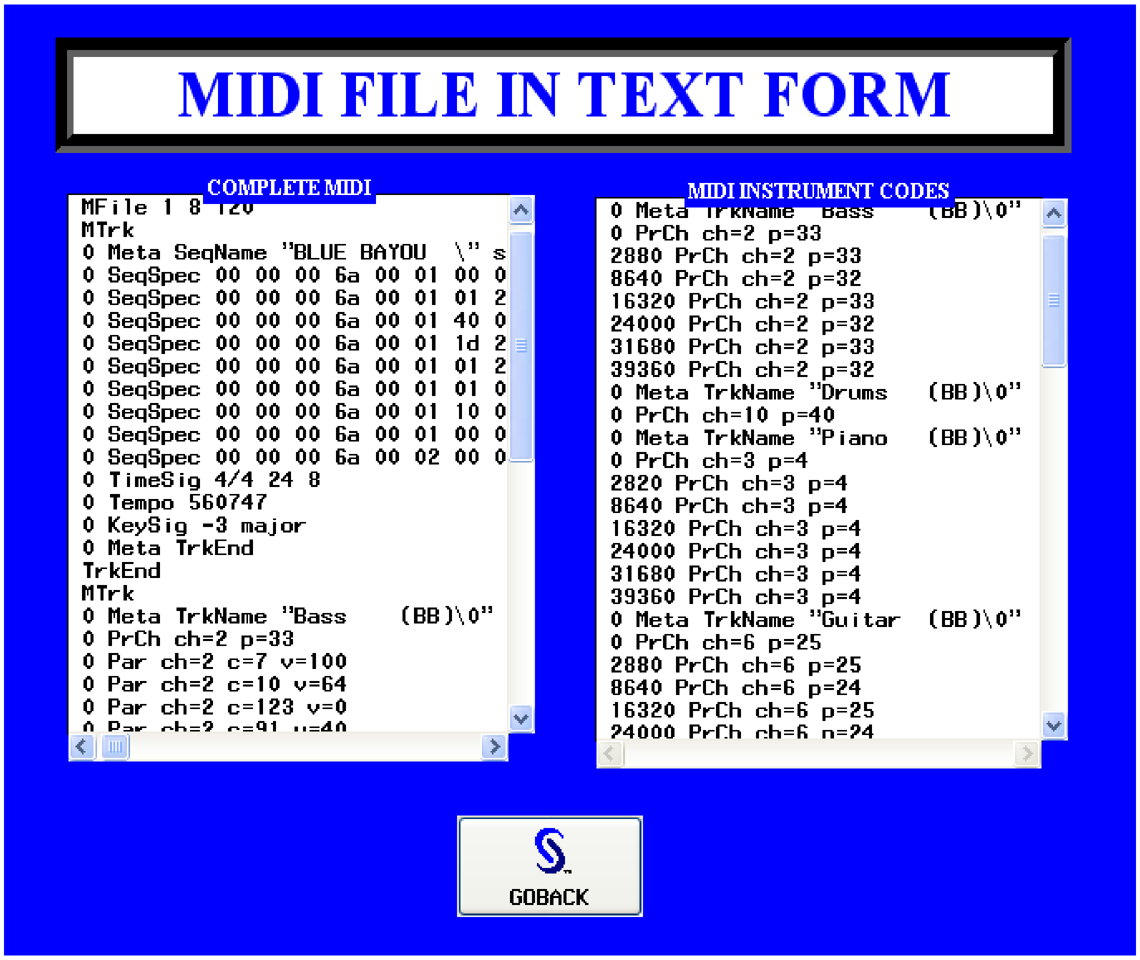


Figure 2: Initial GUI Screen for Converting MIDI Data to SAS Format.

The user may also choose to listen to the MIDI file by selecting “PLAY WMP,” which automatically pulls up the Windows Media Player, or a music synthesizer “SEQUENCER” which pulls up the PowerTracks Pro (a product of ©PGMusic Inc.) synthesizer. In order to complete the key second step for conversion to a matrix, click on the “PREPROCESS” icon. This will strip off a key part of the text from the MIDI text file and prepare it for entry into the SAS “PROC IML” module which permits the matrix-level manipulations that is the fundamental level of preparation prior to putting into matrix form. Once the “PREPROCESS” icon is indexed then the “GOBACK” pushbutton is indexed to return to the SAS editor. Once back in the SAS editor, a particular SAS program can be run to strip the text information from the MIDI files and reorganize it into a matrix format such that the information for each instrument is restricted to an assigned column with all of the clock beats aligned to match the timing of all other instruments present in a tune. This program (named “MAT_TRANSFORM”) goes through each element of the MIDI file organizing the information carefully into columns of instruments and rows corresponding to clock beats, creating a matrix that can be manipulated by PROC IML in SAS. This PROC IML program is a basic program that creates the actual MIDI matrix by converting the organized data from MIDI pitch code (0 to 127, i.e., 7 bit data) into standard sound frequencies and then inputs the information into a matrix which can easily be manipulated. The SAS code underlying this part of the SAS GUI is available at http://www.shsu.edu/~mth_crh/SCSUG/mat_transform.sas.



```
MFile 1 8 120
MTrk
0 Meta SeqName "BLUE BAYOU   \" s
0 SeqSpec 00 00 00 6a 00 01 00 0
0 SeqSpec 00 00 00 6a 00 01 01 2
0 SeqSpec 00 00 00 6a 00 01 40 0
0 SeqSpec 00 00 00 6a 00 01 1d 2
0 SeqSpec 00 00 00 6a 00 01 01 2
0 SeqSpec 00 00 00 6a 00 01 01 0
0 SeqSpec 00 00 00 6a 00 01 10 0
0 SeqSpec 00 00 00 6a 00 01 00 0
0 SeqSpec 00 00 00 6a 00 02 00 0
0 TimeSig 4/4 24 8
0 Tempo 560747
0 KeySig -3 major
0 Meta TrkEnd
TrkEnd
MTrk
0 Meta TrkName "Bass      (BB)\0"
0 PrCh ch=2 p=33
0 Par ch=2 c=7 v=100
0 Par ch=2 c=10 v=64
0 Par ch=2 c=123 v=0
0 Par ch=2 c=91 v=40
```

```
0 Meta TrkName "Bass      (BB)\0"
0 PrCh ch=2 p=33
2880 PrCh ch=2 p=33
8640 PrCh ch=2 p=32
16320 PrCh ch=2 p=33
24000 PrCh ch=2 p=32
31680 PrCh ch=2 p=33
39360 PrCh ch=2 p=32
0 Meta TrkName "Drums     (BB)\0"
0 PrCh ch=10 p=40
0 Meta TrkName "Piano     (BB)\0"
0 PrCh ch=3 p=4
2820 PrCh ch=3 p=4
8640 PrCh ch=3 p=4
16320 PrCh ch=3 p=4
24000 PrCh ch=3 p=4
31680 PrCh ch=3 p=4
39360 PrCh ch=3 p=4
0 Meta TrkName "Guitar    (BB)\0"
0 PrCh ch=6 p=25
2880 PrCh ch=6 p=25
8640 PrCh ch=6 p=24
16320 PrCh ch=6 p=25
24000 PrCh ch=6 p=24
```

Figure 3: MIDI File in Text Form for Viewing (Results from Indexing "View MIDI.")

Once the matrix transformation program is run, another SAS program (named "GO_BACK.SAS") can be executed to transform back to the MIDI format for playback purposes. After execution of the SAS program that transforms the matrix of frequencies back to MIDI object code form, the Windows Media player automatically opens, allowing the user to listen to the results. Again after the matrix of frequencies are manipulated in whatever way one wishes, additional SAS Screen Control Language (SCL) code allows playback of "transformed MIDI" so the user can discern to what degree the manipulations altered (if at all) the auditory properties of the original tune. The SAS code that is the basis for this part of the SAS GUI is retrievable at the following address:

http://www.shsu.edu/~mth_crh/SCSUG/go_back.sas

More detailed specifics of the "MIDI to text" SAS GUI capability is described below:

1. The process starts with a selected MIDI file; initially, this file is in "object code" form (e.g., in hexadecimal, packed binary, or some other unreadable (by a standard text editor) form); this file is called the "Raw MIDI File", i.e., the RMF. From Figure 2, the names of all the available MIDI files populate the listbox on the right after selection of the CATEGORY in the listbox on the left. So categorical (or genre) selections can first (e.g., COUNTRY, BLUES, etc.) be accomplished prior to selection of a specific MIDI tune from that category.

2. A free downloadable program (along with "readme" directions to see exactly how it operates) referred to by the file name MF2T.EXE out on the Web (just Google it) is run on the selected RMF to do the following:
    a. It changes the RMF file to a text MIDI file (i.e., a TMF file, which is readable by any standard text editor) and stores it in the same folder with the "txt" filetype and with the same first part name.
    b. It results in all music notes being 7 bit data (i.e., ranging from 0 to 127).
    c. Notes for a particular instrument, say instrument *i*, are referenced by the terminology "channel *i* data".
    d. A corresponding file, named T2MF.EXE, will transform a text MIDI file back to a RMF for play back through a sequencer (T2MF.EXE is available at the same Web site).

3. The SAS SCL (Screen Control Language) software that is key to the AF Frame part of the SAS GUI capability has the following characteristics:
    a. Upfront, it numbers every line of the original TMF so that the "original sequence" of <u>everything</u> is retained.
    b. It converts the 7 bit note data (i.e., that ranging from 0 to 127) to standard hertz frequencies (f): using the conversion formula (here $0 \leq n \leq 127$):

    $$f = 440 \times 2^{(n - 69)/12}$$

    Note that the key of A in the fourth octave (denoted by A4) is equal to 440 hertz, which occurs when $n = 69$ in the above. Obviously, solving the same equation for n results in the backward conversion formula in going from frequencies back to standard MIDI instrument codes (done after all experimental/other manipulations are finished and, at which time, the user is ready to hear the "impact" (e.g., after a transformation) in playback mode, i.e., back in standard MIDI "object code" mode).
    c. The text controlling part of the MIDI file is stored separately from the frequencies for later use when one wishes to transform back to an RMF for playback through a sequencer.
    d. Finally, the numerical frequencies, after separation from the text part of the MIDI, are output to an m by p matrix M where m is the number of clockbeats and p is the number of instruments. Thus, the m_{ij} component of

M is the frequency (in hertz) of the jth instrument at the ith clockbeat in time.

e. In this form, M is now readily available for algebraic manipulations inside SAS (e.g., using IML --- or for passing to any other similar software).

GENERAL ALGEBRAIC MODEL UTILIZED

Throughout this section, the assumption is that one has two MIDI matrices created using the IML program discussed above (i.e., both are comprised of the frequencies in hertz), i.e., M_0 is an $m \times r$ matrix comprised of m clock beats and r instrument frequencies throughout, and M_1 is a $k \times p$ matrix comprised of k clock beats and p instrument frequencies throughout. The assumed matrix model relationship between MIDI tunes M_0 and M_1 is expressed by

$$M_0 = LM_1R + E,$$

where L is a $m \times k$ matrix, R is a $p \times r$ matrix, and E is the error matrix (this model is a slightly more generalized model than that in the Cannon and Hallum (2004) paper as a result of a default approach for selecting R and the manner of incorporating it into the derivation for obtaining the matrix L (e.g., default values for R may be resorted to that eliminate the problem of a possible discrepancy between dimensions on either side of the above transformation equation)). Based on the manner in which matrices M_0 and M_1 are formed (i.e., with rows representing the clock beats in time and the columns representing the instruments), *the transformation L might appropriately be referred to as the "within instruments" transformation while the transformation R might be referred to as the "between instruments" transformation*. Such labels appear appropriate when one considers the way matrix multiplication is carried out on the rows and columns of the matrix M_1. To lessen the break in continuity here, the derivations are confined to Appendix 1 along with the logistics for the default selection of the matrix R (as a choice for, again, solving the discrepancy of matrix dimensions in the defining equation). So from the Appendix 1 results, one obtains:

<u>Result 1</u>: Assuming the linear relationship between two MIDI tunes is given by $M_0 = LM_1R + E$, where R is a given fixed matrix, then the transformation L that minimizes $Tr(E'E)$ is given by

$$L = M_0R'M_1'(M_1RR'M_1')^+ + Z\left(I - M_1RR'M_1'(M_1RR'M_1')^+\right)$$

where Z is an arbitrary matrix conformable for multiplication and the estimator of E is $\hat{E}$ where $\hat{E} = M_0 - LM_1R = M_0[I - (M_1R)^+(M_1R)]$ and $LM_1R = M_0(M_1R)^+M_1R$. Consequently, one can see that if the column space of the MIDI tune M_0 is contained in the column space of M_1R then $\hat{E} = 0$.

Some Key Observations/Conclusions

The principal components of a MIDI matrix permit ranking the sources of variation attributable to the instruments that comprise the matrix columns. Running a principal component analysis on the covariance matrix of the MIDI tune frequencies yields all the eigenvalues and associated eigenvectors for the tune. SAS ranks the eigenvalues and, thus, ranks the influential components in regard to how much of the variability in the tune is explained by each particular component. The eigenvectors can be used further to determine how influential each instrument is by investigating the eigenvector components. The instrument corresponding to the eigenvector component having the largest absolute value (associated with the largest eigenvalue) is the most influential. In all cases investigated to-date, an interesting (but not all that surprising) observation is that the first principal component or two typically result in pinpointing which instrument(s) is (are) the melody instrument(s).

A number of conclusions worthy of note are listed as follows:

1. Further evidence now exists to support the conclusion (initially reported by Cannon and Hallum (2004)) that the first principal component (of the covariance matrix computed from the rows of M_0 (which, again, are the across instrument frequencies in hertz)) typically identifies the most influential instrument(s) (at times, it may be more than one principal component that does so). Interpreting the information based on the relative magnitude of the coordinates of the orthonormalized eigenvectors utilized in creating the principal components is, at times, an effective way to do this (this is, in fact, done and discussed in item 3 below).

2. Since a number of MIDI tunes are not necessarily completely identifiable by a single instrument melody, additional instruments' contributions to the MIDI (typically identifiable from investigating one or more additional principal components, in the order of decreasing eigenvalue magnitudes) are interesting to investigate in regard to insights into the characteristics and unique features of the piece of music. Again, these results were initially highlighted by Cannon and Hallum (2004) and have now been further re-enforced.

3. Moreover, as an add-on to Conclusion 2, investigating those principal components that explain at least 90% to 95% of a tune's variability (as measured by the cumulative proportion of eigenvalues) often suffice to identify which instruments most uniquely characterize a MIDI tune. The following are the principal component analysis results for a National Anthem MIDI that utilizes 9 instruments (so it's a 9 column matrix in raw vector-value form):

PRINCIPAL COMPONENTS OF THE NATIONAL ANTHEM MIDI

PC 1	PC 2	PC 3	PC 4	PC 5	PC 6	PC 7	PC 8	PC 9
0.046353	0.029518	0.052600	0.651548	0.753598	-.038773	0.000336	-.013192	0.010531,
0.362822	0.857738	0.357391	-.065598	-.024029	0.004087	0.000634	-.002518	0.003643,
0.006507	-.003629	0.011974	0.052495	0.006123	0.730550	0.063865	0.468737	.489399,
0.428838	-.498772	0.751078	-.053457	-.013731	-.012309	-.001704	-.001643	0.001699,
-.001084	-.001167	0.001001	0.001679	-.000234	-.000556	0.525589	0.579421	0.622913,
0.061422	0.008761	0.011051	0.749229	-.655655	-.066505	-.003158	0.014273	-.012832,
0.008627	-.007909	0.001697	0.053374	-.027577	0.677627	-.108431	-.482550	0.540796,
0.823659	.120356	-.552368	-.036770	0.024441	-.001040	0.004254	0.002607	-.003810,
-.002061	-.001042	0.002750	0.004524	-.006767	0.031966	0.841361	-.459683	-.282315

PC	EIGVAL		% EACH	% CUMULATIVE	
1	457331.360	84651.910	0.4803	0.4803	
2	372679.450	298240.229	0.3914	0.8717	
3	74439.221	37729.504	0.0782	0.9499	←1st 3 Principal
4	36709.717	27734.064	0.0386	0.9885	Components
5	8975.653	7770.862	0.0094	0.9979	explain 95%
6	1204.791	796.726	0.0013	0.9992	of the
7	408.065	208.876	0.0004	0.9996	variation.
8	199.188	6.658	0.0002	0.9998	
9	192.530		0.0002	1.0000	

Table 1: Principal Component Results on a National Anthem MIDI

Based on the principal components of a National Anthem MIDI in Table 1, the first three account for 95% of the variation across the instruments; based on the position of the highlighted weights, the 2nd, 4th, and 8th instruments comprise the majority of information in this particular rendition of the national anthem. Consideration for transformation of this MIDI file based on various principal components amounts to particular choices made for the "across instruments" transformation *R* as the only transformation. To illustrate interesting choices here for this rendition of the National Anthem, the following results were generated, using the specifics discussed herein, "for you listening pleasure":

1. The original National Anthem MIDI can be listened to at:

   http://www.shsu.edu/~mth_crh/SCSUG/natanthem_2.mid

2. From the above, since the first three principal components accounted for about 95% of the total variation, upon checking the coordinates of the first 3 principal components, instruments 2, 4, and 8 were the most important; to listen to just those three instruments (i.e., excluding the other 6 instruments), click on the following:

   http://www.shsu.edu/~mth_crh/SCSUG/natanthem_3_pca_selected_instruments.mid

3. Another interesting MIDI is the transformation of the National Anthem MIDI using the full 9 by 9 principal component matrix (specifically, this is the matrix listed at the top of Table 1); consequently, the original MIDI file was converted to

its frequencies in hertz using the SAS approach discussed previously, then matrix multiplication was performed using the principal component matrix as the value of the matrix *R*, and, finally, the MIDI was converted back for playback. There were no particular expectations regarding the audio on playback; at this time, playback was done for curiosity purposes. To listen to the "principal component matrix transformed results", click on the following address:

http://www.shsu.edu/~mth_crh/SCSUG/natanthem_2_trans_by_9_by_9_pcm.mid

4. The MIDI resulting from multiplying by the weights of the top three principal components (i.e., those three having the largest 3 eigenvalues), with the other columns of *R* being zeroed out, can be heard by clicking on this address:

http://www.shsu.edu/~mth_crh/SCSUG/natanthem_2_by_top_3_pcs.mid

Other results concerning other specific choices for R will be presented at conference time.

Note that if R is an orthogonal matrix, then $R^{'} = R^{-1}$. Then LM_1R in Result 1 reduces to $LM_1R = M_0R^{-1}\left(M_1^{'}M_1\right)^{+}\left(R^{'}\right)^{-1}R^{'}M_1^{'}M_1R$ after distributing the pseudoinverse (see Schott (1997)) across the parentheses, which becomes $LM_1R = M_0R^{-1}\left(M_1^{'}M_1\right)^{+}M_1^{'}M_1R$. For this to reduce further, consider that $\left(M_1^{'}M_1\right)^{+} = \left(M_1^{'}M_1\right)^{-1}$, which is possible only if M_1 is of full column rank, in which case $LM_1R = M_0R^{-1}\left(M_1^{'}M_1\right)^{-1}M_1^{'}M_1R = M_0R^{-1}R = M_0$. This implies that if M_1 is of full column rank and R is orthogonal, then the transformation of M_1R (i.e., by multiplying by L on the left) into the MIDI matrix M_0 results in no error. Note that in order to satisfy the condition that $R^{'} = R^{-1}$, it suffices for R to be a permutation matrix. Also note that the transformation of M_1 into M_0 will result with no error if the MIDI matrix M_1 is of full column rank. This yields Conclusion 4, which is a special case of Result 1.

4. The transformation whereby matrix M_1 is multiplied by a matrix R on the right, i.e., $M_0 = LM_1R + E$, does not affect the relationship between M_0 and M_1 provided R is an orthogonal matrix. (Note: In the special case whereby R is a permutation matrix, the above multiplication by R amounts to a switching of instruments in regard to the frequencies played). This observation was made in the Cannon and Hallum (2004) paper.

5. Since increasing the frequency of a note by a semitone, or half-step, is accomplished by multiplication by the factor $\sqrt[12]{2}$, which is approximately 1.059, this can be accomplished by simply multiplying the frequency matrix by a diagonal matrix with the constant on the diagonal (which, of course, is equivalent to multiplying by this constant as simply a scalar).

Some additional "listening" examples regarding the general transformation results obtained from the use of the conclusions of Result 1 are listed as follows:

- The results of transforming the tune Ashogan's Farewell (a Civil War tune) to the National Anthem:
  - First, below is the address of the Ashogan's Farewell MIDI tune itself:

    http://www.shsu.edu/~mth_crh/SCSUG/Ashogans_Farewell.mid

    NOTE: Here, upon checking, the Ashogan's Farewell MIDI matrix is of rank 2 while the National Anthem MIDI matrix is of rank 9; obviously, this type of relationship between two MIDI tunes impacts the auditory playback sound of the National Anthem in the range and null spaces of Ashogan's Farewell. Speaking algebraically, the more of the National Anthem matrix that lies in the range space of the Ashogan's Farewell matrix, the more distinguishable the playback expectedly would be. A similar conclusion can be made regarding the part of the National Anthem that lies in the null space of Ashogan's Farewell. Similar conclusions should hold for the other mapping attempts discussed below as well. At this point, these observations are made for "aesthetic pleasure" only; specific utilities (or the lack thereof) for such information will be forthcoming with future research.

  - To playback the part of the National Anthem that lies in the range space of Ashogan's Farewell, click on the following:

    http://www.shsu.edu/~mth_crh/SCSUG/Ash_NatAnth_Range.mid

    Here M_1 is the Ashogan's Farewell MIDI matrix of frequencies and M_0 is the National Anthem MIDI matrix (and $M_0(M_1R)^+M_1R$ is the mapping of the National Anthem to the range space of Ashogan's Farewell; it is the MIDI file at the above Web address).

  - To playback the part of the National Anthem that lies in the null space of Ashogan's Farewell (i.e., this is $\hat{E}$ in Result 1), click on the following:

    http://www.shsu.edu/~mth_crh/SCSUG/Ash_NatAnth_Null.mid

- The results of transforming Eidelweiss (an integral part of the movie *The Sound of Music*) to the Wild Side of Life (WSL, a Hank Williams tune) are discussed below:

- First, below are the addresses of Eidelweiss and the Wild Side of Life MIDI files utilized herein (just to have a "feel" for their sound):

  Eidelweiss: http://www.shsu.edu/~mth_crh/SCSUG/Eidelweiss.mid

  WSL: http://www.shsu.edu/~mth_crh/SCSUG/WSL.mid

  NOTE: Both matrices have 8 columns, but the Eidelweiss MIDI matrix is of rank 2, while the Wild Side of Life MIDI matrix is of rank 8. The reason the Eidelweiss matrix is of rank 2 is because it is played with 2 instruments, but both instruments were duplicated three times each, for a total of 8 columns (again, with 6 being duplicates of the original two).

- To playback the part of the Wild Side of Life that lies in the range space of Eidelweiss (i.e., $M_0(M_1R)^+M_1R$; see Result 1), click on the following:

  http://www.shsu.edu/~mth_crh/SCSUG/Eid_WSL_Range.mid

  Note here that M_1 is the Eidelweiss MIDI matrix of frequencies and M_0 is the Wild Side of Life MIDI matrix of frequencies.

- To playback the part of the Wild Side of Life that lies in the null space of Eidelweiss (i.e., again this is $\hat{E}$ in Result 1), click on the following:

  http://www.shsu.edu/~mth_crh/SCSUG/Eid_WSL_Null.mid

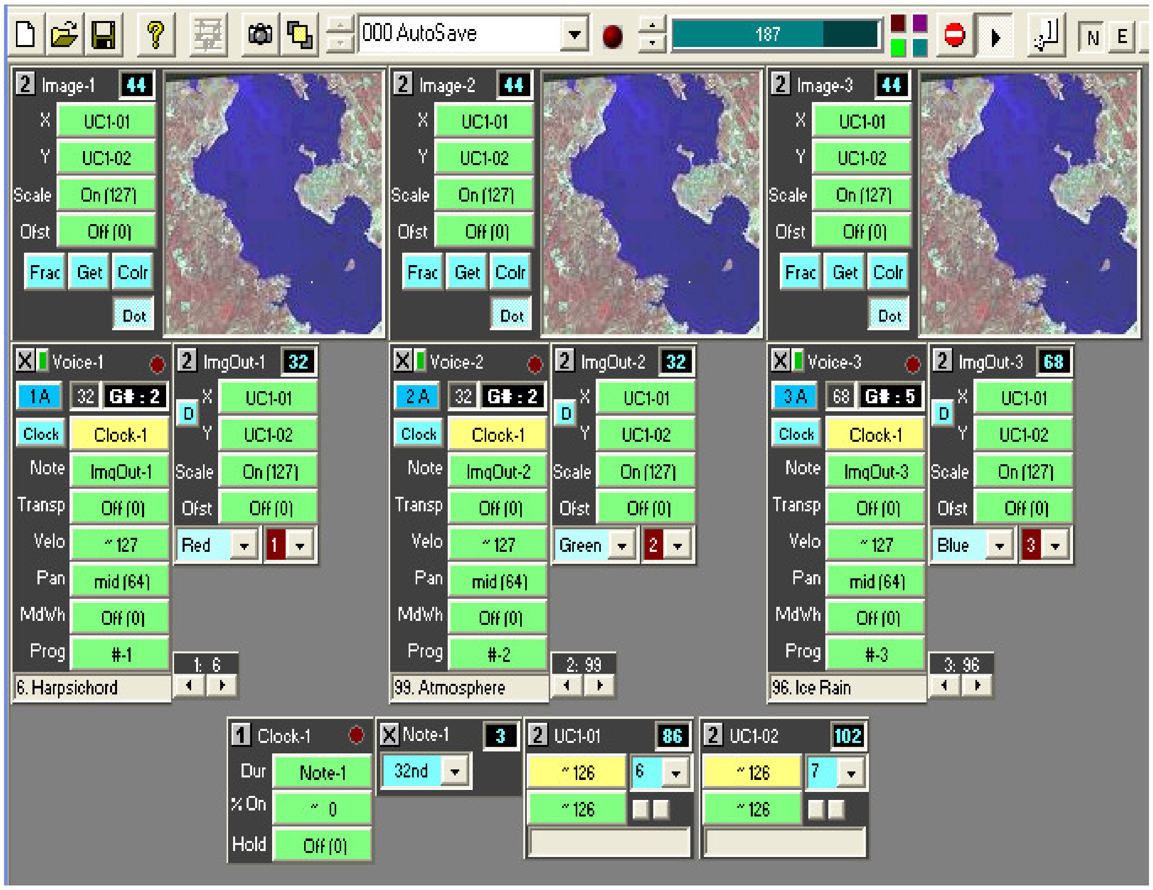
  Note that the file played back here is the file: $\hat{E} = M_0[I - (M_1R)^+(M_1R)]$. This is that part of the Wild Side of Life matrix that lies in the null space of the Eidelweiss matrix.

- Other "mapping" results will be available for demonstration at conference time.

"LISTENING" TO VECTOR-VALUED, DIGITAL DATA – A MIDI SPIN-OFF

At the outset of this study, it was anticipated (i.e., hoped) that considerable insight would be gained from the study of MIDI data to assist a seeing-impaired person in regard to "hearing objects" via the sound sensor, i.e., the ear. This is demonstrated with the aid of the ©SoftStep Version 3.2 software (at www.algoart.com ; I believe SoftStep has recently been updated and is probably now referred to by the name "Artwonk") which, with a little programming, has the ability to simultaneously sound out the red, green and blue (RGB) values at the pixel-level from digitized photographs; consequently, success was had in getting SoftStep to playback sounds for each pixel's color in a digitized photo, which is specifically the task of sounding out vectors of length 3 with coordinates R, G, and B based on the location of the cursor.

Through the use of this software, one can listen to 7 bit data (i.e., integer values ranging from 0 through 127) whereby the larger the numerical value, (i.e., the closer it is to 127), the higher the sound pitch. But this is precisely the type of approach utilized in "hearing" MIDI music which is multivariate in structure as well. The tricky part here was that of "fooling" the ©SoftStep software into playing vector-valued data (it is designed to sound out univariate values). Below in Figure 4 is a screen shot of the object-filled screen that permits listening to the red, green, and blue (RGB) values at the pixel-level of a Landsat satellite image of Lake Livingston. The effort of getting SoftStep to sound out the red, green, and blue pixel values simultaneously required dropping a triple of each scene on the screen and having three identical screen control language modules running simultaneously. This effort was totally successful and will be demonstrated at conference time; note in Figure 4 that there are three identical scenes of Lake Livingston on the screen, one is used for sounding out the redness in the pixel, one for sounding out the greenness in the pixel, and the third for sounding out the blueness in the pixel. After some preliminary screen control language programming in SoftStep software, upon moving the cursor over the Lake Livingston image, the user hears the results of a sounding out of the RGB values for each pixel the cursor moves across. This leads to a number of possibilities for assisting a seeing-impaired person to "hear" digital information. At conference time, this capability will be demonstrated on a digital photo of a sunset, a digital photo of the petals of a flower, and on an additional "very memorable photo" as well (to be revealed at conference time).


000 AutoSave 187

2 Image-1 44 — X UC1-01, Y UC1-02, Scale On (127), Ofst Off (0); Frac Get Colr Dot
2 Image-2 44 — X UC1-01, Y UC1-02, Scale On (127), Ofst Off (0); Frac Get Colr Dot
2 Image-3 44 — X UC1-01, Y UC1-02, Scale On (127), Ofst Off (0); Frac Get Colr Dot

Voice-1 — 1A 32 G#:2; Clock Clock-1; Note ImgOut-1; Transp Off (0); Velo ~127; Pan mid (64); MdWh Off (0); Prog #-1; 6. Harpsichord; 1: 6
2 ImgOut-1 32 — D; X UC1-01; Y UC1-02; Scale On (127); Ofst Off (0); Red 1

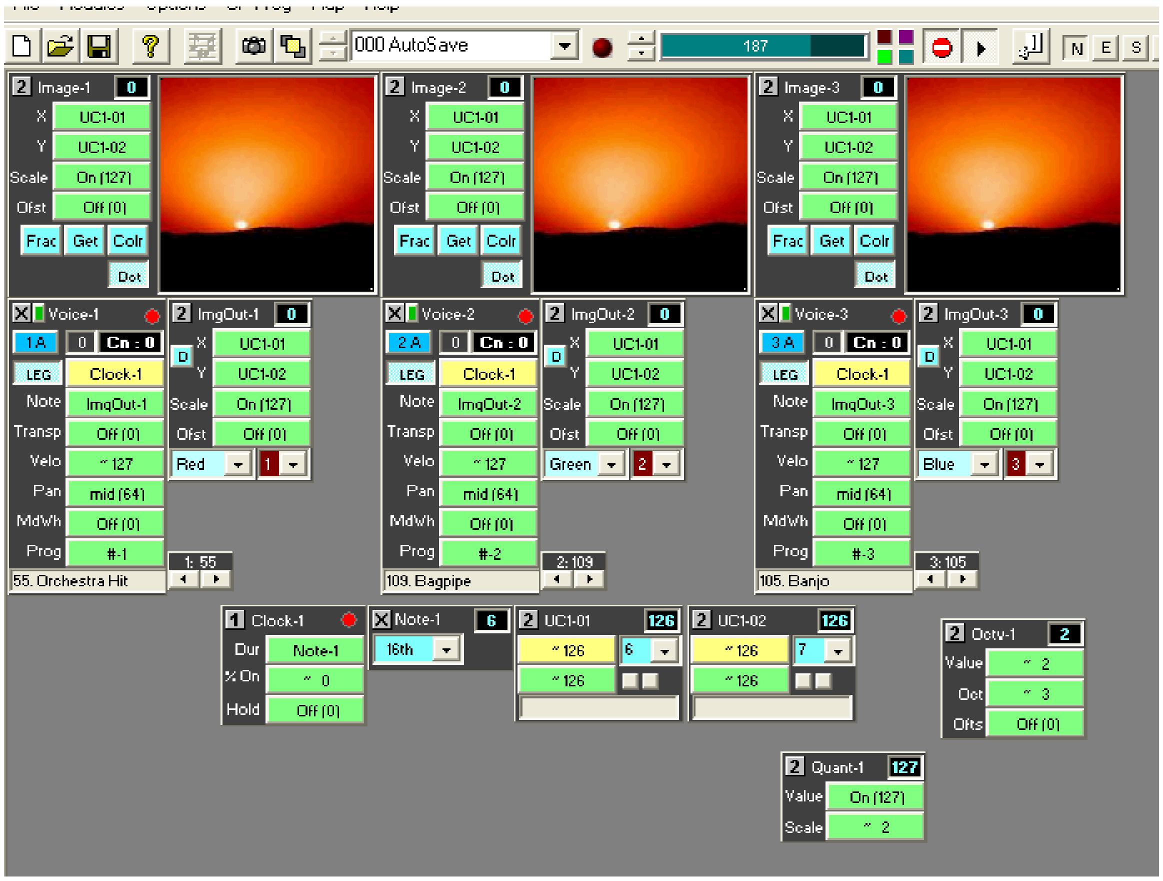
Voice-2 — 2A 32 G#:2; Clock Clock-1; Note ImgOut-2; Transp Off (0); Velo ~127; Pan mid (64); MdWh Off (0); Prog #-2; 99. Atmosphere; 2: 99
2 ImgOut-2 32 — D; X UC1-01; Y UC1-02; Scale On (127); Ofst Off (0); Green 2

Voice-3 — 3A 68 G#:5; Clock Clock-1; Note ImgOut-3; Transp Off (0); Velo ~127; Pan mid (64); MdWh Off (0); Prog #-3; 96. Ice Rain; 3: 96
2 ImgOut-3 68 — D; X UC1-01; Y UC1-02; Scale On (127); Ofst Off (0); Blue 3

1 Clock-1 — Dur Note-1; %On ~0; Hold Off (0)
Note-1 3 — 32nd
2 UC1-01 86 — ~126 6; ~126
2 UC1-02 102 — ~126 7; ~126

Figure 4: GUI Screen for "Listening" to a Satellite Image of Lake Livingston

A number of thoughts come to mind in regard to further worthwhile research along these lines: one is that of investigating patterns of multiple cursors that a seeing-impaired person might have access to in regard to “choices for listening configurations”. As an example, consider the lap board in Figure 6 for possible usage by a seeing-impaired person. Assuming that the indexing of each button on this board results in a predetermined choice for the number of cursors (i.e., not necessarily just a single cursor) along with a chosen pattern with which these are dispersed over the digital scene based possibly on a selected beginning point (e.g., a dispersing of 6 cursors outward in a semicircular array form from the sun’s center in Figure 5); such considerations could be a big benefit for the seeing-impaired and they could easily be taught to use this setup for enhancing listening options. Other possibilities will be discussed at conference time.



000 AutoSave
187

2 Image-1 0
X UC1-01
Y UC1-02
Scale On (127)
Ofst Off (0)

2 Image-2 0
X UC1-01
Y UC1-02
Scale On (127)
Ofst Off (0)

2 Image-3 0
X UC1-01
Y UC1-02
Scale On (127)
Ofst Off (0)

Voice-1
1A 0 Cn : 0
LEG Clock-1
Note ImgOut-1
Transp Off (0)
Velo ~ 127
Pan mid (64)
MdWh Off (0)
Prog #-1
55. Orchestra Hit
1: 55

2 ImgOut-1 0
D
X UC1-01
Y UC1-02
Scale On (127)
Ofst Off (0)
Red 1

Voice-2
2 A 0 Cn : 0
LEG Clock-1
Note ImgOut-2
Transp Off (0)
Velo ~ 127
Pan mid (64)
MdWh Off (0)
Prog #-2
109. Bagpipe
2: 109

2 ImgOut-2 0
D
X UC1-01
Y UC1-02
Scale On (127)
Ofst Off (0)
Green 2

Voice-3
3 A 0 Cn : 0
LEG Clock-1
Note ImgOut-3
Transp Off (0)
Velo ~ 127
Pan mid (64)
MdWh Off (0)
Prog #-3
105. Banjo
3: 105

2 ImgOut-3 0
D
X UC1-01
Y UC1-02
Scale On (127)
Ofst Off (0)
Blue 3

1 Clock-1
Dur Note-1
% On ~ 0
Hold Off (0)

Note-1 6
16th

2 UC1-01 126
~ 126 6
~ 126

2 UC1-02 126
~ 126 7
~ 126

2 Octv-1 2
Value ~ 2
Oct ~ 3
Ofts Off (0)

2 Quant-1 127
Value On (127)
Scale ~ 2

Figure 5: GUI Screen for the Capability to “Listen” to a Digital Photo of a Sunset

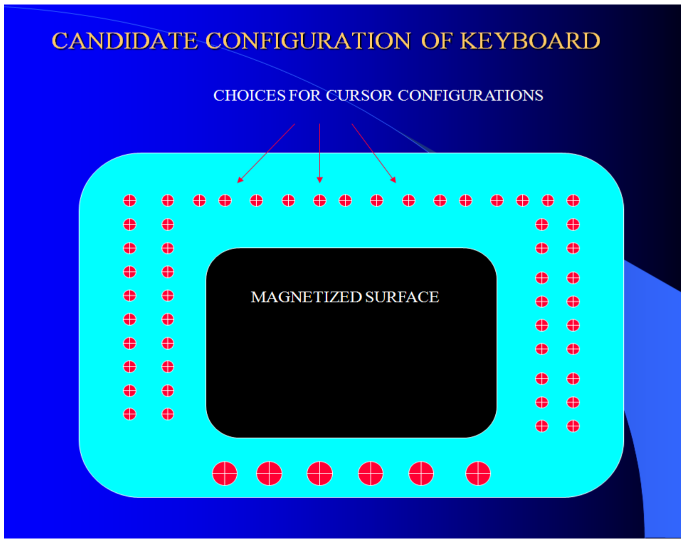

CANDIDATE CONFIGURATION OF KEYBOARD
CHOICES FOR CURSOR CONFIGURATIONS
MAGNETIZED SURFACE

Figure 6: Lap Board for a Seeing-Impaired Person's Configuration Choices

A PRACTICE BACKUP RHYTHM BAND PLUS GUI INTERFACE TO MUSIC

As a wrap-up to this discussion regarding a GUI capability for handling music entities, last, but far from least, is my favorite: a SAS GUI capability to expeditiously select from a large library of music existing in numerous formats with each requiring differing software modules for playback. The author extensively uses this capability in SAS for practice sessions in preparation for gigs, for fun in relaxed music sessions with music friends, and for just general listening. The screen below (see Figure 7) is essentially self-explanatory, but this capability will be further demonstrated and explained at conference time. This SAS GUI is set up to interface with several other music software modules including Band-in-a-Box, PowerTracks, RealBand, Sonar 8.5, ©Karafun (a karaoke player) and Windows Media Player; the first three are products of ©PGMusic, Sonar 8.5 is a product of ©Creative, and, of course, Windows Media Player is a product of ©MicroSoft. You may wish to Google each for further information.

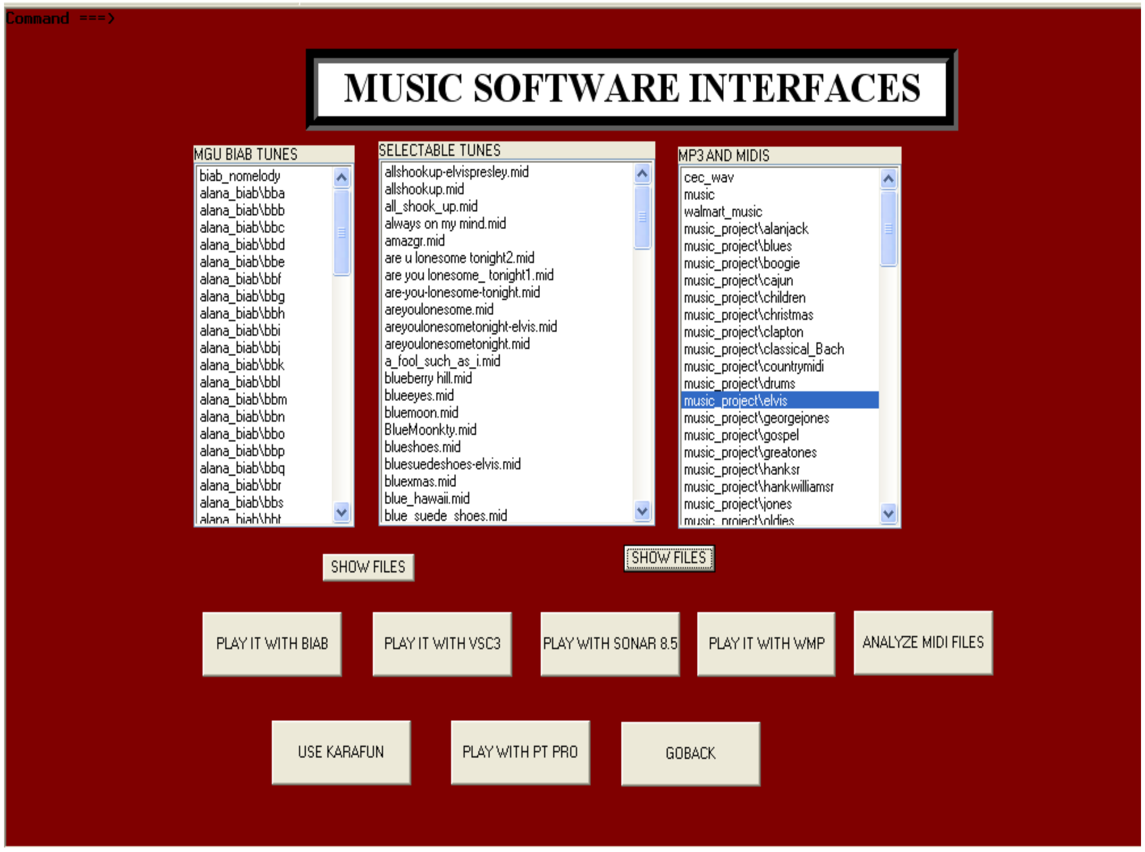


Figure 7: A GUI Interface to Band-in-a-Box, to Sonar 8.5, to RealBand, to Windows Media Player, to PowerTracks PRO, and to Karafun Karaoke Player

Incidentally, all GUI screens herein are contained in the file "transmusic.dat" available at http://www.shsu.edu/~mth_crh/SCSUG/transmusic.dat. This file was produced using PROC CPORT. To bring these screens along with their SCL (screen control language) and put them in place on another computer (on which SAS is implemented), the assumption is that a SAS library exists with the name "MUSIC" and, by running the PROC CIMPORT with the catalog name MUSIC.MIDI as the output catalog, all screens should load without a problem. Feel free to contact the author if assistance is needed in doing this.

ADDITIONAL RESEARCH QUESTIONS

Future research will include, as time permits, comparisons of numerous additional known pieces of music (some similar, some quite different; e.g., it would be particularly interesting to discover ways and means for effectively determining similarities and dissimilarities within and across waltzes, rock tunes, blues tunes, various artists, key top Billboard tunes, etc.).

1. How might one extensively compare (where such requires a mathematical and/or statistical assessment and/or modeling effort to permit doing so) the "similarities" and "dissimilarities" of a given tune with:

   a. A given collection of tunes that one has specific interest in?
   b. A given genre of tunes?
   c. The current Billboard top 20?
   d. Those of a particular artist?
   e. A specific category (e.g., waltzes)?
   f. Etc.

2. In number 1, what "similarities" and/or "dissimilarities" are sufficient for distinguishing across a collection of tunes; in what manner might these be arrived at mathematically and/or statistically?

3. Again, in number 1 and 2, how might one arrive at a minimal set of discriminant features that are quantifiable.

4. Given a collection of tunes (possibly quite large), it appears worthwhile to determine ways and means of identifying subgroups that "make sense" in a mathematical and/or statistical manner. For example, one way might be to find a way to have the resulting subgroups have "maximum similarity" within each subgroup and, at the same time, "maximal dissimilarity" between groups (such as optimal clustering approaches might permit).

5. To what extent might the positive features of one tune be effectively transformed to another via mathematical/statistical transformations (e.g., what transformations might be worthy of consideration and to what extent might it be (or not be) successful). What transformations might improve a tune's "listening aesthetics", e.g., improve vibrato, improve the balance of volumes across instruments, ways and means to achieve optimal mixing, etc.?

6. Given that MP3 and WMA files are sampled WAVE files, how might this sampling be done differently to permit other advantages?

7. All the above can be considered separately for:

   a. MIDI tunes (probably the easiest).
   b. MP3 and WMA files.
   c. WAVE and other CD audio files.

8. What can be learned from any of the above to help in other areas? For example, how might digital pictures (which can be digitized into RGB values) be transformed and converted to sound in such a way to permit a seeing-impaired person to "listen" to digital photography in a meaningful way that might be even more enlightening (i.e., beyond the "somewhat simple" ways discussed herein).

9. How about a design for bar coding so that aesthetically pleasing sounds are heard (maybe even discernible music) when an item is scanned in the super market or wherever?

REFERENCES

Cannon, Lisa and Hallum, Cecil (2004), "SAS and Multivariate Music: A GUI for Multivariate Analysis of Musical Instrumentation Digital Interface (MIDI) Data", SAS Conference Proceedings: South-Central SAS User Group 2004, Austin, Texas.

Chew, E., and Chen, Y. (2003), "Mapping MIDI to the Spiral Array: Disambiguating Pitch Spellings," Proceedings of the International Multimedia Conference, Berkeley, CA.

Das, M., Howard, D., and Smith, S. (1999), "Motion Curves in Music: the Statistical Analysis of MIDI Data," Proceedings of the 25th Euromicro Conference, Milan.

Dubnov, S., Assayag, G., Lartillot, O., and Bejerano, G. (2003), "Using Machine-Learning Methods for Musical Style Modeling," *Computer*, 36, 73-80.

Foote, J., Cooper, M., and Nam, U. (2002), "Audio Retrieval by Rhythmic Similarity," Proceedings of the International Conference of Music Information Retrieval, Paris.

Jensen, J.H., Christensen, M.G. and Jensen, S. H. "A Framework for Analysis of Music Similarity Measures," ©2004 EURASIP available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.138.105&rep=rep1&type=pdf.

Jensen, J.H., Christensen, M.G., Ellis, Daniel P.W., and Jensen, S.H., (2008), "A Tempo-insensitive Distance Measure for Cover Song Identification based on Chroma Features", *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 2209–2212.

Johnson, R., and Wichern, D. (2002), *Applied Multivariate Statistical Analysis* (5th ed.), Upper Saddle River, NJ: Prentice Hall.

Krygier, J. (1994), "Sound and Geographic Visualization," *Visualization in Modern Cartography*, New York: Pergamon, 149-166.

Lartillot, O. (2003), "Perception-Based Musical Pattern Discovery," Proceedings of the International Computer Music Conference, Singapore.

Lodha, S., Beachan, J., Heppe, T., Joseph, A., and Zane-Ulman, B. (1997), "MUSE: A Musical Data Sonification Toolkit," Proceedings of the International Conference on Auditory Display, Palo Alto, CA.

Martins, A., Portela, L., Rangayyan, R., Amaro, E., and Ruschioni, R. (1996), "Auditory Display and Sonification of Textured Image," Proceedings of the International Conference on Auditory Display, Palo Alto, CA.

Moon, T., and Wynn, S. (1999), *Mathematical Methods and Algorithms for Signal Processing*. Upper Saddle River, NJ: Prentice Hall.

Myers, R., and Milton, J. (1998), *A First Course in the Theory of Statistical Models* (2nd ed.), New York: McGraw-Hill.

Pauletto, S., and Hunt, A. (2004), "Interactive Sonification in Two Domains: Helicopter Flight and Physiotherapy Movement Analysis," Proceedings of the International Workshop on Interactive Sonficiation, Bielefeld, Germany.

Petersen and Pedersen (2008), The Matrix Cookbook, available at http://matrixcookbook.com/.

Pickens, J. (2000), "A Survey of Feature Selection Techniques for Music Information Retrieval," Proceedings of the International Symposium on Music Information Retrieval, Plymouth, MA.

Schott, J. (1997), *Matrix Analysis for Statistics*, New York: John Wiley and Sons.

Serra, X. (1997), "Musical Sounds Modeling with Sinusoids Plus Noise," *Musical Signal Processing*, eds. C. Roads, S. Pope, A. Picialli, and G. De Poli, Barcelona: Swets and Zeitlinger Publishers.

Tzanetakis, G., Ermolinskyi, A., and Cook, P. (2002), "Pitch Histograms in Audio and Symbolic Music Information Retrieval," Proceedings of the International Computer Music Conference, Gothenburg, Sweden.

Yeung, E. (1980), "Pattern Recognition by Audio Representation of Multivariate Analytical Data," *Analytic Chemistry*, 52,1120-1123.

APPENDIX 1: DERIVATION OF RESULT 1

The key derivation question is: "What are the desired characteristics of the transformations L and R that result from 'mapping' one MIDI tune to another?" The distance utilized herein between two matrices (i.e., M_0 and LM_1R in this particular case) is the Frobenius norm (see Schott (1997)) defined (using the trace operator "Tr") as $d(M_0, LM_1R)$ where

$$d(M_0, LM_1R) = \sqrt{Tr\left((M_0 - LM_1R)'(M_0 - LM_1R)\right)}.$$

In this expression, the prime in the exponent position is notation for the matrix transpose operator. Notice that $M_0 - LM_1R$ is simply the error associated with the above transformation of M_1 into M_0. Emphasis is now given to finding the transformation L. The matrix R is a known matrix (with a default value to be discussed shortly) carefully preselected to ensure that the matrix dimensions are conformal between the right and left sides of the equation $M_0 = LM_1R + E$. In order to find an appropriate transformation, L of M_1, the transformation obtained is the one that minimizes the square of the Frobenius norm, $Tr(E'E) = Tr[(M_0 - LM_iR)'(M_0 - LM_iR)]$, i.e., this is the "error minimization criterion" utilized herein (See Schott (1997)). Two identities of matrix differentiation utilized herein to minimize the Frobenius norm are (here A, B, C and X are matrices and differentiation is the usual (see the Matrix Cookbook (2008)): in particular, if $f(X)$ is a scalar function of the matrix X then the derivative of $f(X)$ with respect to the matrix X is defined by

$$\frac{\partial f(X)}{\partial X} = \left(\frac{\partial f(X)}{\partial x_{i,j}}\right)$$

which is the standard component-wise differentiation. From this, we have

$$\frac{\partial}{\partial X} Tr[B'X'CXB] = C'XBB' + CXBB' \quad \text{and} \quad \frac{\partial}{\partial X} Tr[AXB] = A'B'.$$

where, of course, "Tr" is the trace operator.

Consequently, upon taking the derivative of $Tr(E'E)$ with respect to the matrix L, i.e.,

$$\frac{\partial}{\partial L} Tr(E'E) = \frac{\partial}{\partial L} Tr[R'M_1{}' L'LM_1R - R'M_1{}' L'M_0 - M_0{}' LM_1R + M_0{}' M_0]$$

and equating the result to zero, the equation $LM_1RR'M_1{}' = M_0R'M_1{}'$ results; solving this for L yields

$$L = M_0R'M_1{}'(M_1RR'M_1{}')^+ + Z\left(I - M_1RR'M_1{}'(M_1RR'M_1{}')^+\right)$$

where the superscript "+" denotes the matrix pseudoinverse (see Schott(1997)). Multiplying by M_1R on the right in the above expression results in

$$LM_1R = M_0(M_1R)'[(M_1R)(M_1R)']^+ + Z(M_1R - M_1R) = M_0(M_1R)^+ M_iR.$$

Note that if M_iR is of full column rank, then

$$(M_1R)^+ = \left((M_1R)'(M_1R)\right)^{-1}(M_1R)',$$

in which case $LM_1R = M_0$, or $\hat{E} = 0$. Consequently, one obtains Result 1.

A Default Choice for R: The "Between Instruments" Transformation

Since MIDI tune M_0 is $m \times r$ and M_1 is $k \times p$, for the matrix LM_1R to be conformable for addition, the matrix R will be selected to be of dimension $p \times r$. In particular, if M_0 has fewer columns than M_1, i.e., $r < p$, then the MIDI tune M_1 utilizes $p - r$ more instruments than M_0. In this situation, the default value for R is that of an $r \times r$ identity matrix concatenated below (i.e., vertically) by a $(p-r) \times r$ matrix of zeros. If $r > p$, then the default value for R is the $p \times p$ identity matrix concatenated on the right by a $p \times (r-p)$ matrix of zeros. To complete the default assignment, if $r = p$**,** then R is selected to be the $p \times p$ identity matrix. Finally, for investigative and/or experimental purposes, many other choices (e.g., a few columns of the principal components matrix computed from the row vectors of M_1) can be made for R. Choices and properties of the matrix R are given in the following discussion. It should be noted that this default choice for R is done, typically, only for the purpose of attaining conformability for multiplication, i.e., there is nothing "magic" about this choice. In fact, note that in the case where M_1 utilizes more instruments (i.e., this is the case $r < p$) than M_0, then the default choice for R actually results in simply muting the instruments that play the last $p-r$ columns of M_1.