SPORCH: An Algorithm for Orchestration Based on Spectral Analyses of Recorded Sounds

David Psenicka

School of Music, University of Illinois
email: dpsenick@uiuc.edu

Abstract

SPORCH (SPectral ORCHestrater) is a Lisp based computer program that provides orchestrations for any ensemble of acoustic instruments based on any arbitrary sound file input. The result, when played, approximates the sound source in timbre and sound quality. The amount of approximation depends on the nature of the source material, the instruments specified, and other controlling parameters. SPORCH is a compositional tool for composers who wish to work directly with complex timbres or sonorities when composing for acoustic instruments. Since it is able to detect the presence of specific instruments and pitches within a complex chordal structure, it also has potential as an analytical tool.

1 Introduction

Composers have recently created new approaches to orchestration by considering acoustic phenomena as a source of inspiration. This way of thinking is not only concerned with technique but is also the background for an aesthetic that emphasizes the evolution of sound material as opposed to the relationships of inner sections, motives, themes, or other such elements. (Moscovich 1997) Pioneers in this approach/aesthetic include the composers associated with the Spectral Movement that began in France during the 1970's, Gerard Grisey and Tristan Murail being the most well known among them. (Rose 1996) A common approach is to base choice of pitch material and instrumentation on information derived from spectrum analyses of various sound sources, or to derive material based on acoustic properties such as combination and difference tones or FM synthesis. This enables the composer to have control over the gradual and perceptual evolution of sound in ways analogous to various sound phenomena known in the field of acoustics.

Fundamental to the creation and development of spectral music is the introduction of the digital computer and with it the ability to analyze the frequency content of recorded sound material. With the computational resources available today, the ideas mentioned above have become more realizable for a large number of composers and current interest in these approaches is growing. It would be useful to devise a tool capable of enabling a composer to influence the timbral and textural qualities of orchestration and harmony in a more direct way, or even to actually specify a certain desired timbre or possibly the fusion of two or more timbres with a certain degree of accuracy. If a direct translation could be made between acoustic sounds or processes and the realization of them with acoustic instruments, the entire realm of timbre and sonority would be much more accessible for composers of acoustic music. This would allow new techniques in harmony and orchestration and also affect the possible ways in which the electro-acoustic medium can be combined with the acoustic domain.

2 Description of the Algorithm

Following is a description of SPORCH, a program designed to analyze a recorded sound and output a list of instruments, pitches and dynamic levels that when played together create a sonority whose timbre and quality approximate that of the analyzed sound. The interface is in the form of a Common Lisp function, which takes as some of its input arguments a source sound file and a specification for a group of instruments. The program itself is written in both Common Lisp/CLOS and C++ (the Lisp language must have a suitable foreign functions interface for it to be fully functional), and requires Common Music (Taube 1997) to run. Any instruments may be specified for use by the algorithm, as long as relevant acoustic data (explained below) and recordings of each of the pitches over its entire range are available. The program draws information from its own database, which must be built by the user for it to make its decisions.

The creation of SPORCH was motivated by a desire to experiment with new methods of determining harmony and instrumentation. It was initially inspired by the techniques briefly mentioned above and a desire to use these ideas in my own music. Many Lisp programming environments now exist which give composers tools for analyzing and representing sound data. OpenMusic (Assayag, *et al.* 1999), CLM (Schottstaedt 1994), and Common

Music are several examples. SPORCH may easily be incorporated into these environments, provided links are made between its internal data structures and whatever interfaces the user wishes to use.

Following is a description of how SPORCH works.

2.1 The Instrument Database

Before the program can be run, a database containing technical information for all instruments used must be built. An instrument may have several techniques associated with it (for example, a string instrument may have separate data for both arco and pizzicato). The database contains the following information for each instrument and technique:

- The pitch range (if applicable).
- A reduced pitch range (if applicable), specifying the pitches that are generally easier to play. This is used by the orchestration algorithm to produce parts that tend to stay within a comfortable range for the instrument. The parts are then more playable than they would be otherwise.
- An approximation of the loudest and softest dynamic levels, expressed as two envelopes over the entire range of the instrument. The envelopes specify the upper and lower bounds in terms of SPL values.
- A collection of the most prominent peak values taken from a spectrum analysis of each pitch at various dynamic levels. If information for some pitches is missing, data from the nearest neighboring pitch is transposed and used in place of the missing data. The same is also true with respect to dynamic levels—missing data is transposed from neighboring existing data. The harmonic spectrum as a function of pitch and dynamic is then available over the entire pitch and dynamic range of the instrument.
- Notation information (clef, transposition, etc.). This information is used to store results into a file suitable for viewing or importing into a notation program.
- References to the original source sound files used to build the database. This is used for resynthesis of the program's output.

The database exists mostly in the form of a Lisp list structure, stored in a single file. The peak data mentioned above is stored in a large collection of files, each one also containing a list structure of data. Each combination of instrument, technique, pitch and dynamic level is represented. The spectrum analyses used to produce this data are FFT analyses of either the steady state portions of the tones or the attack portions if this is more appropriate (as in staccato articulations). A set of functions exists for the purpose of building this database automatically from a given set of source sound files, so that any instrument may be inserted as long as recordings of it are obtained and all the relevant acoustic information is known. As mentioned above, if only some pitches are available (for example, recordings exist of pitches only at whole-step intervals from each other), the orchestration algorithm automatically transposes data from the nearest available pitch when it is needed.

When the database is built, a pitch detection function decides which peaks are important in a sound file's FFT data to include in its collection of prominent peaks. This information is extracted so that noisy data in the FFT analysis is eliminated. The function is designed to find peaks in both harmonic and inharmonic spectra and to ignore frequencies that contribute only as noise. The procedure for this is as follows. Points of local maximum amplitude with respect to frequency (peaks) are found. These amplitude values are analyzed with respect to their relation to a certain range of neighboring amplitude values. This range is determined by searching for nearby maximum amplitude points with nearly equal or higher amplitude levels than the currently analyzed peak. Within this range, the lowest maximum amplitude values on either side of the analyzed peak are then found (in other words, the smallest peaks in close proximity on either side). Peaks that have relatively high amplitudes (i.e. past a certain threshold) in relation to these lowest points are regarded as significant and selected for storage in the database. Certain threshold values are set for this decision to be made. The result is that only peak amplitude values that are significantly higher than the amplitudes of the spectrum at nearby frequencies are selected for storage. All other peaks are discarded.

The reason for using such an algorithm to extract this data is that a general method exists for analyzing any sound with strong frequency components. Sounds for any instrument or even sounds not associated with any typical instrument may then be easily inserted into the database. The same algorithm is used to analyze the source sound files given as input to the orchestration function described below.

2.2 The Orchestration Function

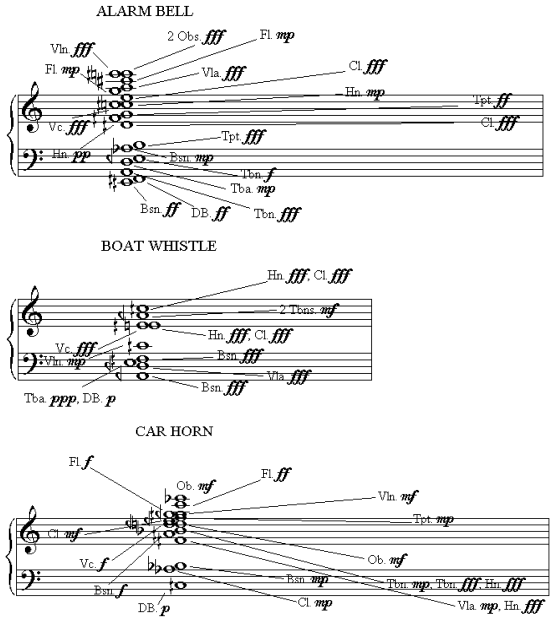
Once the database is built and the appropriate acoustic data is provided, any sound recording may then be analyzed and orchestrated by calling the orchestration function. The function accepts the following arguments: the filename to the source file, a list structure specifying which instruments to use (and how many of each, including whether or not certain substitution may be made), and values specifying the dynamic and pitch ranges allowed (i.e. full or reduced). When finished, the program then

outputs an object containing raw analysis data, which may then be passed to other functions for refinement or output to a notation program. The orchestration algorithm works as follows:

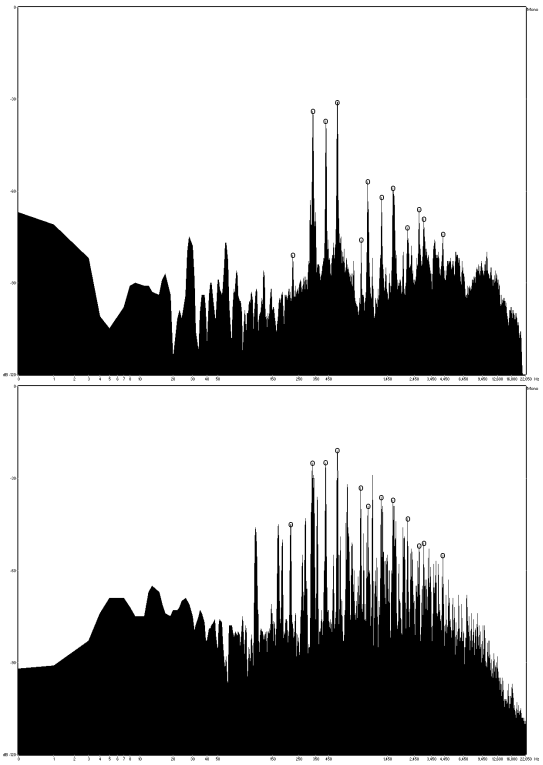
1. The sound source file is opened and analyzed for prominent peaks in its frequency spectrum (via the procedures described above).
2. The program iterates through every possible instrument, technique, pitch, and dynamic level and evaluates these to determine which one is the "best fit" for the given sound source. The best fit is determined using a procedure that compares the source peak data to the instrument peak data and rates their similarity with a numerical value. More specifically, the rating is determined by how much the source spectrum can be decreased by "subtracting" the instrument peaks from the source peaks. The following steps describe this in more detail:
   a. The program iterates through every selected prominent peak for the instrument/technique/pitch/dynamic level currently being evaluated.
   b. An attempt is made to match the peak with a corresponding peak in the source sound. A match is made if the two peaks occur within a certain specified frequency range.
   c. If the two peaks match, the amplitude value of the instrument peak is subtracted from the amplitude of the matching source peak. (Amplitude values may become negative.)
   d. If there isn't a match, a new "peak" is created in the source sound data with the same frequency as the instrument peak. The amplitude of the new peak is the negative value of the instrument peak's amplitude. (In other words, a new "negative" peak is created.)
   e. After iterating through each instrument peak, the sum of the squares of each peak in the source is calculated to determine a value loosely analogous to the average power of the source's spectrum. Squaring the peaks gives more importance to higher values. Both positive and negative peak amplitude values contribute to this sum.
3. A lower value resulting from this procedure indicates a better fit. If the two spectra are nearly equivalent, the result is close to 0. If they are completely different, the result is a large number. This number might be even greater than the result if nothing at all were subtracted from the source data, in which case the particular instrument, technique, pitch and dynamic level combination is completely discarded as a candidate for the best fit and the program continues to search through other possibilities.
4. If no instruments are able to decrease the spectrum by any amount, the program ends its search and returns the results of its analysis.
5. If the above is not the case, the program adds the best fit to its list and replaces the source peak analysis with the "subtracted" one used in the above evaluation. All subsequent evaluations then use this data to determine the next best fit. The iteration continues until either no instrument is found that decreases the source data or all of the specified instruments have been used up.

In essence, the algorithm described above finds a collection of instruments, pitches, and dynamic levels whose frequency contents add together to form some crude resemblance to the frequency content of the source. Although the resulting orchestration produces a sound completely different from the original, recognizable timbral and pitch characteristics of the original sound are present to some degree in the orchestrated version. The amount of similarity varies depending on the source used and the instrumentation specified. In general, sound sources with a strong pitch element produce orchestrations whose results sound relatively closer with respect to pitch and timbre. Sound sources that contain noise, however, are also useful—the algorithm simply attempts to approximate the noise by selecting a somewhat random but biased collection of instruments and pitches.

The following illustrations are sample outputs from the program. Three sound sources were given as input along with the specification for a small chamber orchestra. The output specifies quarter tones and dynamic levels ranging from ppp to fff.



The following graphs show the spectrum of the original boat whistle sound source and the spectrum of its orchestration. The circles indicate common points between the two graphs:



(Above) Spectrum of Original Score, (Below) Spectrum of Orchestration

Although the spectrum of the orchestration contains many more prominent frequency components than that of the original, most of the original partials are present. The energy of the sound in both examples is also concentrated within the same frequency range. When comparing the two aurally, the timbre of the orchestrated sound is very similar to the timbre of the original, given the fact that the texture of the two sounds are completely different.

3 Conclusion

SPORCH is a compositional tool that gives the composer a great deal of control over aspects of harmony, orchestration and dynamics. It can be used as a source of harmonic material, to create gradual shifts from one sonority to another, combine or interpolate between different timbres, create groups of related sonorities, integrate instrumental sounds with recorded sounds, simulate electronic synthesized sounds, and so on. I have used SPORCH to generate an entire orchestra piece from by analyzing a six minute long audio file of music concrete source material at half second increments. The result is *Image,* a piece for chamber orchestra. Output from the program was stored in a data structure on which I did further processing and manipulation (distributing material among players, adding quarter-tone inflections, etc.). I have also used the program in other works in a less direct way to inform me with harmonic and instrumentation data which are then integrated with other compositional techniques.

Although it has the potential to be used as a tool for extracting pitch or instrument information from a musical recording, tests have shown that SPORCH tends to become confused when asked to recognize more than one or two pitches. The orchestrate function will often accurately assess the pitch content of a chord, although it will usually add extra pitches at lower dynamic levels in an attempt to orchestrate peaks that didn't match perfectly during the search process. It will also often correctly guess the instrument when given a recording of one playing a single note or several notes, but again begins to add incorrect information as the source input becomes more complex. Accuracy is heavily dependent on input constraints, including the range of dynamic levels allowed and the amount of amplification or attenuation applied to the source.

Future versions to this program will improve its ability to extract such information. Refinements and heuristics will also be developed to enhance the underlying search algorithm. A graphical user interface will also be added. SPORCH will soon be available as a free downloadable open source package that can be run on Macintosh or Windows computers or compiled in any ANSI-standard Common Lisp/CLOS programming environment.

4 Acknowledgments

Thanks to Heinrich Taube for his support in the development of this software.

References

Assayag, Gérard, Camilo Rueda, Mikael Laurson, Carlos Agon, and Olivier Delerue. 1999. "Computer-assisted composition at IRCAM: From PatchWork to OpenMusic." *Computer Music Journal* 23(3): 59-72.

Moscovich, Viviana. 1997. "French spectral music: An introduction." *A Tempo* 200 (April): 21-27.

Rose, François. 1996. "Introduction to the Pitch Organization of French Spectral Music." *Perspectives of New Music* 34(2): 6-39.

Schottstaedt, William. 1994. "Machine tongues. XVII: CLM--Music V meets Common Lisp." *Computer Music Journal* 18(2): 30-37.

Taube, Heinrich. 1997 "An Introduction to Common Music." *Computer Music Journal* 21(1): 29-34.