JPRODUCTIONCRITIC: AN EDUCATIONAL TOOL FOR DETECTING TECHNICAL ERRORS IN AUDIO MIXES

Cory McKay
Marianopolis College and CIRMMT
cory.mckay@mail.mcgill.ca

ABSTRACT

jProductionCritic is an open-source educational framework for automatically detecting technical recording, editing and mixing problems in audio files. It is intended to be used as a learning and proofreading tool by students and amateur producers, and can also assist teachers as a timesaving tool when grading recordings.

A number of novel error detection algorithms are implemented by jProductionCritic. Problems detected include edit errors, clipping, noise infiltration, poor use of dynamics, poor track balancing, and many others.

The error detection algorithms are highly configurable, in order to meet the varying aesthetics of different musical genres (e.g. Baroque vs. noise music). Effective general-purpose default settings were developed based on experiments with a variety of student pieces, and these settings were then validated using a reserved set of student pieces.

jProductionCritic is also designed to serve as an extensible framework to which new detection modules can be easily plugged in. It is hoped that this will help to galvanize MIR research relating to audio production, an area that is currently underrepresented in the MIR literature, and that this work will also help to address the current general lack of educational production software.

1. INTRODUCTION

Audio production is a broad field that essentially involves recording and creating music. Important aspects include:

- *Recording:* configuring an acoustic environment, microphone selection, microphone placement, etc.
- *Editing:* shifting segments of audio in time within a track, or moving them between tracks.
- *Mixing:* combining multiple tracks with appropriate gains, panning and EQ settings, applying effects, etc.
- *Synthesis:* artificially generating audio.
- *Sampling:* incorporating pre-existing audio.
- *Mastering:* preparing a mix for final distribution via specific audio formats.

DAW (Digital Audio Workstation) software has come to play a central role in production. Such software ranges from recording-oriented tools like Avid Pro Tools, to live performance-oriented software such as Ableton Live, to free tools like Audacity.

Improved functionality, better user interfaces and decreasing costs have made audio production more and more accessible in recent years. This has helped to cause an explosion of content created in home studios, ranging from amateur mashups to recordings by professional musicians. While this has certainly resulted in a great deal of interesting music, it has also led to error-prone technical work on the part of overconfident amateur producers who lack the professional training that was previously necessary to be involved in production at all.

This problem is part of the motivation behind the jProductionCritic software, which automatically detects technical production errors, especially those relating to editing and mixing. It can help students and amateur producers check their work for unnoticed errors, much as one might use a grammar checker when writing prose. This is beneficial from an educational perspective, as it teaches users to notice problems that they might not otherwise have known to look for. This in effect trains them to improve their listening skills, which are arguably a producer's greatest asset, and pushes them to learn how to avoid or correct the detected problems, thus improving not only their current work, but also the skills that they will be able to apply in the future.

Such error checking software is also useful to those teaching audio production, as it can greatly facilitate grading. While it would certainly be ill-advised to rely exclusively on automated marking, as expert humans are needed to fully evaluate the difficult-to-quantify aesthetics involved in the art of production, simply automating the painstaking task of enumerating and time-stamping basic technical errors can be a great time saver.

Finally, error checking software could even be of some use to professional audio engineers as a final verification tool, just as professional writers make use of spellcheckers. It is not unheard of to find technical errors in professional work, often due to rushed production schedules and the high costs of studio time.

Aside from such practical benefits, developing algorithms for detecting production errors can also have important research value. As discussed in Section 2, not only is there currently a surprising dearth of production-oriented research in the MIR community specifically, but those tools that do exist in the general audio world tend to be closed-source black boxes or based on disappointingly naïve algorithms. This presents an exciting research opportunity, particularly considering the importance of production from both commercial and artistic perspectives. jProductionCritic has therefore been designed not only as a ready-to-use application, but also as a modular framework for developing and deploying new error detection and analysis algorithms in the future.

2. RELATED RESEARCH

Commercial DAW software like Pro Tools tends to offer some basic error detection functionality, and extensive additional functionality can be added via plug-ins. Unfortunately, with certain important exceptions, such functionality tends to be relatively simplistic in implementation and based on proprietary closed-source code, making it expensive to use and difficult to extend. Also, such software tends to emphasize correcting problems rather than detecting where they occur (the latter is not always necessary for the former), something that is of limited educational value. Finally, DAW applications and plug-ins tend to address specific problems independently, with limited functionality for presenting errors to users via integrated interfaces.

Moving outside the domain of DAW software, there are a few commercial integrated error detection systems, such as Fraunhofer IDMT's A/V Analyzing Toolbox [4] and Quadriga Audiofile-Inspector [12]. Unfortunately, as with commercial DAWs, such software is closed-source and thus difficult for independent researchers to extend. This software is also limited in the range of errors detected and in the sophistication of its processing.

In terms of open-source integrated systems, the very basic Digital Audio Error Detection [13] is the only one available. A few open-source error detection DAW plug-ins can also be found, but they are isolated algorithms that each only look for individual errors, and have no integration with one another. Furthermore, they are largely intended for professional use, and typically require a significant amount of knowledge to use, limiting their usefulness in educational contexts.

With respect to research in the MIR community, surprisingly little work has been done relating directly to audio production, even though many of the audio features and metrics used in MIR research are highly relevant to this domain. Scott and Kim [9] and Montecchio and Cont [6] provide good examples of the kind of production-oriented MIR research has been done, but even this high-quality work focuses on automating production tasks rather than finding errors. Such automation is certainly very useful in practice, but it does not address the educational needs emphasized by jPoductionCritic.

There has been a substantial amount of research done outside the MIR community on detecting errors in audio signals. However, this focuses mainly on techniques associated with specific problems rather than general integrated systems. Furthermore, much of this research relates to domains such as broadcasting and audio compression, with less focus on production-oriented problems, and with almost no attention paid to addressing the issue from an educational perspective. Having noted this, there are many technical papers than can each be very useful in detecting specific production problems, particularly in AES (Audio Engineering Society) and IEEE (Institute of Electrical and Electronics Engineers) publications. There are also a number of important general references on audio production, including books by Barlett and Barlett [1], Vaseghi [10], Owsinki [7] and Huber and Runstein [3], the first of which includes a particularly useful chapter on the kinds of defects that one can encounter in an improperly prepared mix.

3. DESIGN AND FUNCTIONALITY

The first main design objective of jProductionCritic is that it be useful and accessible to music students, amateur producers and teachers, all of whom may have little or no experience with software development. To this end, jProductionCritic is distributed with a detailed manual in order to make it as easy to learn as possible. Its basic interface is also designed to be minimalistic and direct so that users can avoid being distracted by anything superfluous. Users simply need to specify an audio file or batch of files to check and where reports are to be saved, and the software automatically takes care of the rest.

Of course, it is also important that the software be highly configurable for those who desire flexibility. There is therefore a separate extensive configuration file that advanced users can modify in order to control which errors are checked for, what error thresholds are used for each error, and so on. It is thus possible to customize jProductionCritic for certain styles of music (e.g. metal vs. jazz), or to simply use the provided general-purpose default settings without worrying about the details.

jProductionCritic processes final mixes in the form of single mono or stereo files rather than DAW projects with tracks still separated out. Although this does make certain errors much harder to detect, it also ensures that no new unchecked errors are incorporated during final mixing and mastering. This also makes it possible to process any standard audio recording with jProductionCritic, and avoids tying it to any particular DAW framework.

Three types of error reports can be generated by jProductionCritic for each audio file. The first is simply an enumeration of the errors that were detected, annotated with time stamps indicating either an instantaneous time or time range, as appropriate. Errors are also marked as being mild, moderate or severe.

The second type of report consists of a series of Audacity Label Tracks, one for each type of time-specific error. These are metadata tracks that Audacity displays alongside audio and MIDI tracks. This can be very useful for visually demonstrating to users where errors occur in a waveform or spectrogram. Audacity was chosen in particular because it is free and thus accessible to all users, whether or not they used it to prepare the audio being checked for errors.

The third type of error report consists of error annotations in Weka ARFF [11] or ACE XML [5], two file formats related to machine learning that are used by the MIR community. Although not directly applicable to the educational context targeted by jProductionCritic, these reports could be helpful to MIR researchers who might want to use the output of jProductionCritic in research involving machine learning. These formats can also be useful when performing experimental validations.

jProductionCritic is implemented in Java, in order to help make it as cross-platform and accessible as possible. This avoids forcing users to buy proprietary environments such as Matlab, and avoids the installation and linking problems that one can encounter with languages like C++.

The second main design objective of jProductionCritic is that it serve as a framework under which new error detection algorithms can be developed and deployed, and not only as a ready-to-use application. This is an important priority in encouraging future MIR research and development focusing on audio production. A strong emphasis was therefore put on designing jProductionCritic using a modular and highly extensible architecture to which new error checking algorithms can be easily added as plug-ins, with virtually no changes needed to the overall jProductionCritic processing infrastructure. Special attention was also paid to extensively documenting the code.

jProductionCritic is distributed as an integrated part of the jMIR [5] suite of MIR research software. This allows researchers to easily combine jProductionCritic's functionality with other jMIR components, such as the jAudio feature extractor or the ACE meta-learning system.

As with all jMIR components, jProductionCritic is free and open-source.

4. TECHNICAL ERRORS ASSESSED

Due to limited space, only broad overviews of jProductionCritic's main error detection algorithms are provided in the sub-sections below. Those wishing to read more details on any particular algorithm are encouraged to view the jProductionCritic manual or the Java class associated with the error type, both of which are available at http://jmir.sourceforge.net.

It is important to emphasize here that there are many important subtle subjective and artistic qualities that must be considered if one is to truly evaluate the production quality of a mix. Performing such an evaluation is well beyond the current technological capabilities of any automated system, and is best left to human experts, such as professional producers and instructors.

jProductionCritic therefore only attempts to detect clear objective technical errors, which many students and amateurs can still unfortunately produce many of. jProductionCritic is intended as a supplement and aid to human experts, not as a replacement for them.

Of course, even with this policy there can still be ambiguity with respect to certain error types. What might be considered unwanted noise in a classical flute recording, for example, might be part of a desirable production aesthetic in a flute sample used in an electronic dance track. Fortunately, jProductionCritic's diverse range of configuration settings makes it possible to easily modify the detection thresholds of given error types, or to disable them entirely, in order to match the various production aesthetics of different musical styles.

4.1 Digital Clipping

Digital clipping occurs when a signal exceeds the representational limits of its bit depth. Clipped signals are characterized by flat peaks and troughs, as samples are rounded to maximum and minimum values. Digitally clipped signals sound rough and distorted, and are almost never aesthetically desirable. Analog clipping, in contrast, can be desirable in certain styles of music, and is characterized by more curved peaks and troughs.

Digital clipping tends to occur in two main ways in student work: either the gain is set too high during recording or synthesis of an individual track, or the gains on individual tracks mixed onto the same channel are too high, such that the combined signals clip, even if none of the source signals are themselves clipped individually.

Although clipping detection is a common software feature, the popular implementation of simply flagging any samples at the representational limits is surprisingly naïve. This approach has two major problems. Firstly, a sample that actually should have a value at the representational limit is not in fact clipped, and such samples are to be expected in normalized signals. Secondly, students may attempt to hide clipping by reducing the master gain in the final mix, such that sample values fall below representational limits (and are thus not flagged) but the signal distortion caused by the clipping remains.

The approach used by jProductionCritic can overcome these two problems: if a number of adjacent samples beyond a threshold have an identical signal value (whether or not it is at the representational limit), then report clipping. The number of such consecutive samples gives an indication of clipping severity.

Despite its simplicity and effectiveness, other uses of this technique were not found in the literature, although related counting techniques *at* the representational limit have been used. It should be noted that the literature also includes spectral approaches for detecting clipping, but these can be too sensitive for styles of music where analog clipping (or its digital simulation) is desirable.

4.2 Edit Clicks

An edit click occurs when an improperly treated edit is made, and can result in a discontinuity in the waveform that typically sounds like a click. This can happen when two signals are spliced together, or at the beginnings and ends of tracks (due to a sudden jump in the signal from or to silence). Although there are a number of techniques that can be used to avoid edit clicks, students and amateurs often neglect to use them.

Although the literature includes a substantial number of techniques for detecting instantaneous noise like clicks in general, it largely neglects edit clicks in particular. This is problematic from an educational perspective, as it is useful for students to know where imperfections in their work come from.

jProductionCritic uses a simple technique to detect edit clicks based on windows of only four samples: report an edit click if a signal jumps in value beyond a threshold from samples 2 to 3, but does not change in value beyond another threshold when progressing from samples 1 to 2 or 3 to 4. This approach is sensitive to improperly executed edits, is relatively impervious to false positives, and can also provide a severity measurement. This technique is also surprisingly absent from the literature, although related techniques considering much broader spreads than four samples are used for detecting instantaneous noise in general. Clicks at the beginnings and ends of tracks are simply found by looking for first and last samples far from zero, respectively.

It should be noted that this algorithm focuses only on a particular kind of edit error. It does not detect edit errors in general, of which there are many other types (e.g. a splice involving two segments of audio recorded under very different reverberant conditions).

4.3 Other Clicks, Pops and Instantaneous Noise

There are also many other types of undesirable instantaneous noise. Plosive pops due to the improper micing of a singer or noise when a needle jumps on a record are just two examples amongst many.

Although, there are a number of established techniques for detecting such problems, many of them tend to produce false positives. jProductionCritic's approach, which also produces some false positives but was still found to be the most effective during comparative experiments, is to high-pass filter the signal (due to the common assumption that unwanted noise will stand out most clearly against the musical signal in the high frequency range) and look for sudden and unusual peaks in the filtered signal's spectral flux.

4.4 Hums and Other Background Noise

Tracks can also be infiltrated by various types of sustained noise (as opposed to the more sudden and short-lived types of noise discussed above). Ventilation systems in recording environments and faulty cable shielding are two of the many possible sources. Detecting such noise in general can be particularly difficult, as it can be hard to distinguish from the musical signal. Although the literature does include certain sophisticated techniques, including approaches based on Hidden Markov models [8], these tend to be too limited in the styles of music to which they can be applied, so it was decided to use a simpler and more general technique11.

jProductionCritic's basic approach is to calculate the power spectrum of the audio and look for sustained peaks in particular frequency regions that are present in all or most of the audio. Extra weighting is applied if these peaks are still present in otherwise quiet parts of the signal. This approach tends to work reasonably well for detecting loud noise, but can miss quieter noise, and can result in false positives for those styles of music that feature sustained drones.

jProductionCritic also has specialized detectors that look for electrical noise (e.g. ground loops), a common problem in imperfectly configured or used studios. Such noise consists of a hum at the AC frequency of the power supply (and its integer multiples), which is generally either 50 Hz or 60 Hz, depending on where one is.

4.5 Phasing

Phasing is a problem that occurs when a signal is mixed with another signal that includes a phase delayed version of itself. This can occur, for example, when two omnidirectional microphones mapped to the same channel are too close to each other, or a single microphone is too close to an acoustically reflective surface. This results in cancellation or reinforcement of various frequencies, depending on the phase offset, which can result in a muddy tone.

Although the literature specifies several effective ways to detect phasing before mixing is carried out, it is much more difficult to automatically detect afterwards, and is easily confused with sometimes desirable comb filter effects like flanging. jProductionCritic's (admittedly limited) approach is to look for consistent troughs in the power spectrum of a track.

4.6 Dynamic Range

A common mistake made by students is to keep gains excessively low due to fear of clipping. Students then sometimes exacerbate this by forgetting to normalize their work during mastering (which can be desirable in order to achieve relatively consistent volumes). Another potential problem is that some tracks are insufficiently dynamically compressed (a desirable "hot" aesthetic in pop styles) or, conversely, do not have enough dynamic range (a problem for styles such as classical music).

To address the first issue, jProductionCritic reports an error if the maximum absolute sample value is too far below the representational maximum. To address the other two problems, optional style-specific configuration settings can be specified to generate errors if the standard deviation of the windowed RMS across a track is too high or too low, respectively.

4.7 Stereo Balance and Channel Similarity

Some students do not include enough channel separation in their recordings to create a sufficient sense of stereo space, or even forget to specify panning settings at all. Additionally, students sometimes fail to properly balance the stereo channels, with the result that one stereo channel is consistently louder than the other.

jProductionCritic compares the left and right stereo channels and generates an error if the signal correlation is too high. It was found that this works better in general than spectral approaches. An error is also generated if the RMS of one channel as a whole is too high relative to the RMS of the other channel as a whole.

4.8 Other Errors Assessed

There are several additional errors that can be reported by jProductionCritic if desired. These include, among others:

- Too much silence (either absolute or at the noise floor) at beginnings and ends of tracks.
- Audio dropout.
- DC signal offset.
- Poor encoding parameters (e.g. low sampling rate or bit depth, lossy compression, etc.) in cases where high-quality masters should be used.

jProductionCritic also reports basic summary metadata (e.g. track length, audio encoding parameters, etc.).

5. VALIDATION EXPERIMENTS

Much of the error detection processing described above is based on thresholds, which the user has the option of specifying via configuration settings. However, it is important that it also be possible to apply jProductionCritic easily and effectively to arbitrary types of music without any user tweaking. To this end, experiments were performed to first arrive at good default configuration settings, and to then validate these settings' effectiveness.

In order to do this, music technology assignments were collected from multiple sections of three different courses over four semesters at Marianopolis College. Most but not all of the students involved were enrolled in the music program. Some of these assignments required students to make classical or jazz recordings using Pro Tools (in studio and live), and others required students to make mashups in any musical style using Audacity. The instructor's original (and later re-verified) corrections to the assignments served as the ground truth. In all, 110 assignments were collected.

Forty-four of these assignments were randomly selected and used to experimentally choose the error detection algorithms and tune their configuration settings. Once this was done, the remaining 66 assignments were then processed by jProductionCritic in order to verify that the configuration settings had not been overfitted to the tuning set. The results of this validation experiment are shown in Table 1:

	True Positives	False Positives	False Negatives
Human	499	0	8
jPC	452	38	55

Table 1. Results of the validation experiment comparing jProductionCritic's performance with expert human correction. Values indicate the total number of errors detected combined across all 66 validation assignments.

It is interesting to note that 8 true technical errors were detected by jProductionCritic that were wrongly missed during original human correction (they were found to be true errors upon manual secondary verification). It was also found upon secondary verification that, unlike jProductionCritic, the original corrector did not wrongly indicate any false errors.

Overall, it can be seen that jProductionCritic performed quite well. It found 89% of the true errors, compared with 98% found by the expert course instructor. Furthermore, 92% of the errors detected by jProductionCritic were in fact true errors. This is impressive when one recalls that the assignments were in a variety of musical styles, and were all processed using the same default configuration settings. It should be noted, however, that these results would be even more meaningful if students at different institutions with different instructors had been involved in the study.

With respect to the relative performance of the different error types, the algorithms for detecting phasing, background noise and, to a lesser degree, instantaneous noise (other than edit clicks) were by far the worst performers. It was difficult during the tuning stage to find configuration settings for them that would minimize false positives while also maximizing true positives, and this was reflected in the validation stage, where these three types of error detectors were responsible for 73% of all jProductionCritic's false positives and false negatives. The other algorithms performed relatively similarly (and successfully).

While jProductionCritic is still not as good as a human expert, it did perform well enough to be at least comparable, and it certainly caught many errors missed by the students. It is sufficiently good to serve as a timesaving and verification tool for teachers, and can effectively provide students and amateur producers with valuable feedback for improving their work.

6. CONCLUSIONS AND FUTURE RESEARCH

It is hoped that jProductionCritic will help to address several underserved needs: the absence of integrated open-source production error checking software in general; the absence of software intended to meet the educational needs of audio production students in particular; and the relatively limited attention given to both production and education software in the MIR community to date.

From a research perspective, jProductionCritic has the advantages of including a number of original error detection algorithms and of being fully open-source. Unlike almost all other integrated production error detection software, its algorithms are not proprietary black boxes. Furthermore, jProductionCritic has a modular and easily extensible design that is intended to encourage its use as a framework for future MIR research on developing additional error checking algorithms.

From an applied perspective, jProductionCritic is the only known production-oriented software that is intended to meet the specific needs of education, and looks for many more errors than any other known general integrated system. Moreover, the validation experiments found that the software performs more than well enough to be used successfully in practice.

The first priority for future work is to port jProductionCritic to a standard DAW plug-in format, such as VST or Nyquist. This will greatly increase its accessibility to students. Another priority is to implement it as a Vamp plug-in so that it can be used with Sonic Visualiser [2], which would increase the scope and clarity of information that could be shown to users by supplementing the Audacity Label Tracks currently used.

It would also be useful to implement an interface with which teachers could specify a grading scheme, so that assignments could be marked more easily, and students could have an idea what grades they will receive before submitting. Of course, it is important to reserve room for the instructor's subjective judgment when doing this.

There are also many other useful error detection algorithms that remain to be implemented, including detection of poor EQ, too much or too little reverberation, excessive performance artifacts, etc. There is also still plenty of potential to refine and improve the existing algorithms, especially those that performed poorly in the validation tests, perhaps with the ultimate goal of making jProductionCritic more useful in professional contexts.

jProductionCritic, its code and documentation can all be downloaded for free from: http://jmir.sourceforge.net.

7. REFERENCES

[1] Barlett, B. and J. Barlett. 2009. *Practical recording techniques: The step-by-step approach to professional audio recording.* Burlington, MA: Focal Press.

[2] Cannam, C., C. Landone, M. Sandler, and J. P. Bello. 2006. The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. *Proceedings of the International Conference on Music Information Retrieval*. 324–7.

[3] Huber, D. M., and R. E. Runstein. 2009. *Modern recording techniques.* Burlington, MA: Focal Press.

[4] Kühhirt, U. 2008. *A/V Analyzing Toolbox.* Retrieved 8 May 2013, from http://www.idmt.fraunhofer.de/en/Service_Offerings/technologies/a_d/av_analyzing_toolbox.html.

[5] McKay, C. 2010. Automatic music classification with jMIR. *Ph.D. Dissertation*. McGill University, Canada.

[6] Montecchio, N., and A Cont. 2011. Accelerating the mixing phase in studio recording productions by automatic audio alignment. *Proceedings of the International Society for Music Information Retrieval Conference.* 627–32.

[7] Owsinski, B. 2013. *The mixing engineer's handbook.* Independence, KY: Course Technology PTR.

[8] Sabri, M., J. Alirezaie, and S. Krishnan. 2003. Audio noise detection using hidden Markov model. *Proceedings of the IEEE Workshop on Statistical Signal Processing*. 637–40.

[9] Scott, J., and Y. E. Kim. 2011. Analysis of acoustic features for automated multi-track mixing. *Proceedings of the International Society for Music Information Retrieval Conference.* 621–6.

[10] Vaseghi, S. V. 2009. *Advanced digital signal processing and noise reduction.* Chichester, West Sussex: Wiley.

[11] Witten, I. H., E. Frank, and M. A. Hall. 2011. *Data mining: Practical machine learning tools and techniques.* New York: Morgan Kaufman.

[12] *Audiofile-Inspector & Digital Error Checker*. Retrieved 8 May 2013, from http://www.cube-tec.com/products/quadriga/quadriga-features-system-integration/quadriga-features-audiofile-inspector-and-digital-error-checker.

[13] *Digital Audio Error Detection.* Retrieved 8 May 2013, from http://digauderrodetec.sourceforge.net.