Real-Time Emulation of a Marshall JCM 800 Guitar Tube Amplifier, Audio FX Pedals, in a Virtual Pedal Board

Michel Buffa, Jerome Lebrun

Real-Time Emulation of a Marshall JCM 800 Guitar Tube Amplifier, Audio FX Pedals, in a Virtual Pedal Board

Michel Buffa
Université Côte d'Azur
CNRS, INRIA
buffa@i3s.unice.fr

Jerome Lebrun
Université Côte d'Azur
CNRS
lebrun@i3s.unice.fr

ABSTRACT

The ANR project WASABI [12] will last 42 months and consists in developing a 2 million songs database with interactive WebAudio enhanced client applications [1]. Client applications target composers, music schools, sound engineering schools, musicologists, music streaming services and journalists. In this paper, we present a virtual pedal board (a set of chainable audio effects on the form of "pedals"), and a guitar tube amplifier simulation for guitarists, that will be associated with songs from the WASABI database. Music schools and music engineering schools are interested in such tools that can be run in a Web page, without the need to install any further software. Take a classic rock song: isolate the guitar solo, study it, then mute it and play guitar real-time along the other tracks using an online guitar amplifier that reproduces the real guitar amp model used in the song, with its signature sound, proper dynamic and frequency response. Add some audio effects such as a reverberation, a delay, a flanger, etc. in order to reproduce Pink Floyd's guitar sound or Eddie Van Halen famous "Brown Sound". Learn interactively, guitar in hands, how to fine tune a compressor effect, or how to shape the sound of a tube guitar amp, how to get a "modern metal" or a "Jimi Hendrix" sound, using only your Web browser.

CCS CONCEPTS

• **Software and its engineering → Software organization and properties → Software system structures →** Abstraction, modeling and modularity.

KEYWORDS

WebAudio, Tube Guitar Amp Simulation, Audio Effects, Plugin Architecture, Web Standards

1 INTRODUCTION

The W3C WebAudio API is now a recommendation candidate, and proposes a set of nodes for building an "audio processing graph". The implementation of these nodes in the browser allows for developing a whole range of new applications. This API comes with a large set of predefined nodes (gain, filters, wave-shapers, delay, stereo panner, etc.) that can be assembled into an "audio graph". Some developers have managed to write some impressive web applications (Digital Audio Workstations, real time audio effects, tube guitar amp simulation, synthesizers, organs, pianos, real time 3D sound spatialization etc.), but there was so far no good way to make low level processing until the recent addition of the AudioWorklet node, that is the last inclusion in the WebAudio API version 1 (the now obsolete ScriptProcessor node was initially designed for this purpose but had many flaws).

For the WASABI project, we developed the first online digital emulation of a real tube guitar amplifier: the Marshall JCM 800, a popular amp used by many classic rock artists (AC/DC, Led Zeppelin, Guns N'Roses etc.). Section 2 will detail this work.

In general, guitarists use additional audio effects in the form of audio effect pedals, plugged between the electric guitar and the amplifier, or to the FX loop of the amplifier. Very often we call this set of FX pedals a "pedal board". For WASABI, we wrote a "virtual pedal board" web application, and we created, using WebAudio, some of the most common audio effect pedals. We also started to design a "WebAudio plugin architecture", collaborating with other groups of researchers sharing the same interests. This work is detailed in section 3.

Finally, in section 4, we will conclude with some perspectives and detail the demonstration setup we propose.

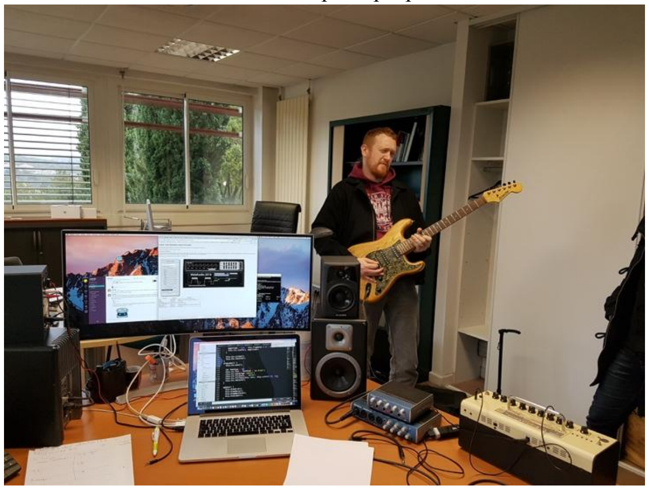


Figure 1: Typical demo setup: a guitar, a low latency sound card, speakers, and a Web browser with the apps running.

[1] The wasabi database is at https://wasabi.i3s.unice.fr/

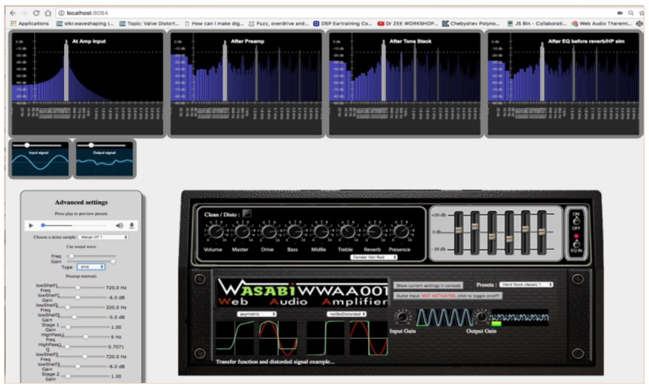
2 TUBE GUITAR AMPLIFIER SIMULATION

Guitar amplifier digital models became popular with devices such as the pod series by Line 6 in the early 2000s. More recently, Fractal Audio Systems introduced the Axe FX-II amp modeler, a comprehensive preamp/effects digital processor containing a vast virtual inventory of hundreds of vintage and modern guitar amps. In 2002, Amplitube, an audio plugin commercialized by IK Multimedia, was the first successful software amp simulation on the market, followed soon by Guitar Rig from Native Instruments. Today, we can find hundreds of native plugins (commercial or freeware, only a few are open source) for digital audio workstations that simulate existing guitar amps, or original designs by their authors. Software amp simulations are popular on-the-run solutions, or when the production budget for recording is low, as they are cheaper and more flexible than their digital hardware equivalents, whereas some may claim they'll never be the same as the real thing.

These last two years, we have been developing a tube guitar amplifier simulation using the WebAudio API with the aim to faithfully reproduce the classic rock Marshall JCM 800 tube amplifier used by artists such as AC/DC, Guns and Roses, ZZ-Top, etc. Each stage of the real amp has been recreated from the electronic schematic (tube preamp, tone stack, reverb, tube power amp and speaker simulation). We've also added an extra multiband EQ. This "classic rock" amp simulation has been used in real gigs and can be favorably compared with some native amp simulation both in terms of latency, sound quality, dynamics and comfort of the guitar play (see [7] for details). The amp is open source[2] and can be tested online[3], even without a real guitar plugged-in. It comes with an audio player, dry guitar samples and a wave generator that can be used as inputs. Figure 2 shows the current GUI, with some optional frequency analyzers and oscilloscopes to probe the signal at different stages of the simulation. One purpose was to evaluate the limits of the WebAudio API and to see if it was possible to design a web-based guitar amp simulator that could compete with native simulations. Blind tests with real guitarists, including professional ones have been conducted and the WebAudio amp came second compared to a set of four different commercial, native, guitar amp simulators[4].

Many papers have been written about vacuum-tube guitar amplifiers modeling [1][6], and about the particularities of linear and non-linear distortion effects suited for guitar [2][3][4][5]. More generally, works such as James J. Clark's "Advanced programming techniques for modular synthesizers" book, are not focused on guitar but discuss thoroughly the different approaches for achieving a distortion effect.

There are two main approaches for simulating the different parts of a guitar amplifier: one called the technique of virtual analog (or physical modeling) that consists in faithfully simulating the electronic schematic with tools like the industry standard SPICE analog circuit simulator to translate the circuit into equations to be solved. These general equations are typically nonlinear differential algebraic equations that may be solved using integration methods, roots solver algorithms, and sparse matrix techniques. SPICE can produce C++ code ready to be executed. However, it is often necessary to make simplifications and optimizations to achieve solutions suited for real-time processing. This is critical for the modeling of the vacuum tubes used in guitar amplifiers with their typical interactions with other parts of the circuitry (see [1] for a review of common techniques). Also, the physical approach usually assumes perfect behavior of the electronic components used (resistors, capacitors, tubes, etc.) making it difficult to render the typical empirical design / fine tuning, ala "musical instrument making", of the tube-based guitar amps in which much of the coloring of the sound comes indeed from the limitations and non-linearities of the electronic components used.



Choose a demo sample: Metal riff 1
Use sound wave
Freq
Gain
Type: sine
Preamp internals
lowShelf1 Freq: 720.0 Hz
lowShelf1 Gain: -6.0 dB
lowShelf2 Freq: 320.0 Hz
lowShelf2 Gain: -5.0 dB
Stage 1 Gain: 1.00
HighPass1 freq: 6 Hz
HighPass1 Q: 0.7071
lowShelf3 Freq: 720.0 Hz
lowShelf3 Gain: -6.0 dB
Stage 2 Gain: 1.00
Fender Hot Rod
asymetric
notSoDistorded
Show current settings in console
Presets : Hard Rock classic 1
Guitar input: NOT ACTIVATED, click to toggle on/off!

Figure 2: Full GUI of the tube amp guitar emulation, with debug tools displayed (freq. analyzers, advances settings etc.).

An alternative technique consists in higher-level emulations: each "logical" being identified (filters, tubes, etc.) analyzed and individually emulated using separate models with now the aim to perceptually approximate these parts. This may be electronically less accurate as some effects and interactions such as the current feedback effect of overloaded tubes or the action of the speaker impedance on the sound tone may not be fully reproduced. However, the results obtained are usually more consistent and perceptually faithful with the sounds / distortions obtained with real tube guitar amps. Furthermore, this approach is much simpler and more adapted to the WebAudio framework with its current limitations (custom processing on audio samples with the script processor node is not usable without introducing latency or glitches, for example). Also, WebAudio proposes nodes (such as the wave shaper node, the biquad filter node) that can be used for modeling tubes and filters, and it has been shown that when properly used, wave shaping techniques associated with appropriate filtering give good results. The famous pod XT by Line 6 effect processor uses such techniques [1].

[2] https://github.com/micbuffa/WebAudio-Guitar-Amplifier-Simulator-3
[3] https://wasabi.i3s.unice.fr/AmpSim3 and a version with measure tools activated: https://wasabi.i3s.unice.fr/AmpSimFA
[4] Videos of tests with real guitarists: https://wasabi.i3s.unice.fr/AmpSim3/userEvaluation.html

As far as we know there is no previous work where authors have tried to simulate a complete tube guitar amp using WebAudio. The most advanced work we have found is a Google Chrome application named GuitarFX[5] that proposes simple amp models with a set of audio effect pedals, but does not fully recreate each stages of a real amp (only one wave shaper per amp, for example), or the guitar amps proposed by the commercial services soundtrap.io and bandlab.com that are based on original designs, and do not aim to reproduce a real, existing, multi-stage amp. Our proposed demonstration is also a good test bench to assess the current limitations of Web Audio in terms of latency (driver, audio buffer size, sample rate frequency, etc.). Now, with respect to the version previously demonstrated [8], we have fully redesigned our approach to the power stage in the tube guitar amp to include a proper emulation of the push-pull stage with output transformer, including the "Negative Feedback (NFB)" and a realistic "Presence" control. Many important new features have also been added like the possibility to emulate the control of biasing points of the tubes or to alter the profile of the NFB. See [13] for details about the modelization of a tube guitar power amp.

3 VIRTUAL PEDAL BOARD AND FX PEDALS

With the WebAudio API standard nodes, it is possible to create more complex audio effects such as a simple delay (delay node, gain node, filter node, feedback loop), a phaser (used a lot in reggae music, made of cascaded allpass filter nodes whose parameters are modulated by an oscillator node), a chorus (the guitar sound in Roxanne by The Police, delay, oscillator, filters), distortion effects (fuzz, metal, classic, crunchy, creamy -there are lots of distortion models- that involves wave shaper nodes and filters), etc.

Sometimes, you need to chain some of these complex effects: add a stereo delay, a fuzz, a reverb, plug this into the amplifier presented in section 2 and you can get close to David Gilmour's Pink Floyd sound. In the native audio application world, such high-level effects are called "plugins" and can be easily shared among hosts, generally Digital Audio Workstations such as GarageBand, Cubase, Ableton Live, etc. There are some competing standards for these plugins and hosts: VSTs and VSTis by Steinberg, Audio Units (Apple), RTAS (Avid), LV2 (Unix), etc. -all describe in detail the host-plugin interfacing to ease integration of any plugin into any host, and how plugins can work together (being connected, expose their parameters, number of inputs/outputs, etc.). Another standard, JUCE, aims to be "above the others" and is a way to "write once, then generate the code for any standard".

Such a high-level "audio plugin" standard does not exist yet for WebAudio, but is targeted for the v2 of the API. For the WASABI project, we've been developing a "plugin host": a "virtual pedal board" that is a recreation of a physical pedal board. This pedal board can be used to assemble WebAudio "plugins", the input is either the signal coming from the guitar or from a sound file, the output is the speakers (Figures 3 and 4). In the WASABI pedal board, you can drag and drop effects, connect them, change the settings using knobs or sliders, control them using a MIDI control device (if your browser supports the WebMidi API), and you can also plug the tube guitar amp simulator presented in section 2 -we made it a plugin too. Each "pedal" or "amplifier" is a WebComponent, and implements our draft version of a "WebAudio" plugin API[6].

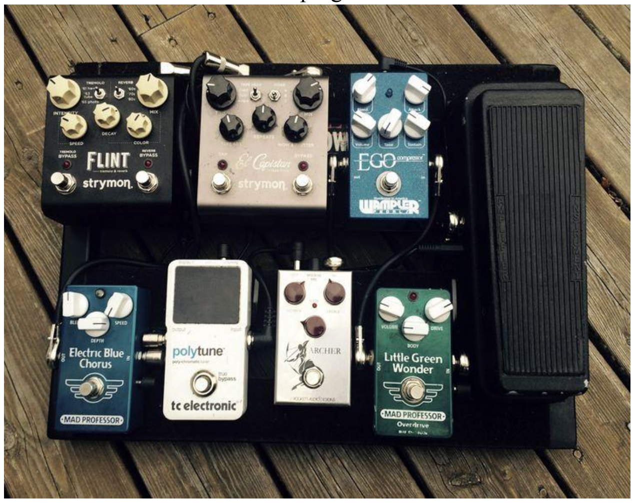


Figure 3: A real pedal board used by guitarists. Audio FX pedals are chained together before going to the amplifier.

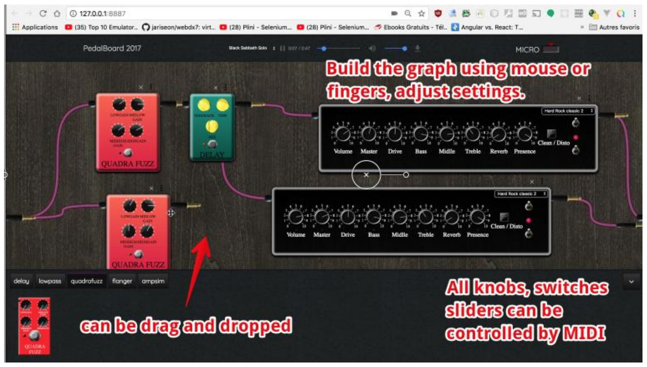


Figure 4: Virtual pedal board. On the right two instances of the guitar amp presented in section 1, here as an audio plugin with a minimal GUI.

In 2012, Google Chrome was the first to propose low-latency access to live audio from a microphone or other audio input on MacOS, followed by a Windows implementation (with a longer latency). Soon Opera, Firefox and more recently Microsoft Edge also implemented this features that relies on the Media Capture and Streams API from W3C. Chris Wilson's "Input Effects"

[5] https://tinyurl.com/ljdhuqh

[6] API specification and the source code of the pedal board, effects, etc. can be found at: https://github.com/guizmo2000/TER-Pedalboard-creation

demo[7] was one of the first to show real time sound processing effects written with Web Audio. He proposed implementations of some famous effects such as delay, distortion, wah, etc. His demo did not allow to chain effects but proved that low latency processing could be achieved. Pedals.io is a JavaScript recreation of some classic audio effect pedals for guitarists (delay, chorus, overdrive, etc.), and we find nearly the same implementations of these effects in many Web Audio JavaScript libraries such as toneJS, tunaJS, pizzicatoJS [8], etc. Two commercial services, soundtrap.io and bandlab.com propose also a set of audio effects.

Our approach is different as we would like to go towards an open WebAudio plugin standard, and for this we need 1) to design an API, 2) to develop a prototype (host and effects) and 3) to work with other groups with the same goal and share plugins with them. Since 2013, we developed our own set of effects in plain JavaScript/WebAudio [11]. Other researchers tried to attract native audio application developers who code in C++ or with the popular FAUST language. We are collaborating with J. Kleimola, and O. Larkin authors of the webaudiomodules.org initiative [10], that proposes both a host and a standard for porting native plugins to WebAudio. The webaudiomodules toolchain compiles C++ code to WebAssembly and wraps the result into an AudioWorklet node, it then becomes possible to port a native VST or JUCE plugin into a traditional WebAudio node with a minimum of efforts. The FAUST language compiler [9], also targets WebAssembly now, and produces WebAssembly code that runs in an AudioWorklet node too [14]. This is interesting as there are many existing plugins/modules/pieces of code written in FAUST or as native plugin for other popular standards (VSTs etc.). Other initiatives such as the one by Nicholas Jillings et al. [9] proposes in addition to consider the concept of "session" in a plugin host.

The consensus for now is that an audio plugin should behave like a standard low level WebAudio node, exposing its I/Os, parameters, name, version, etc. The way to distribute and reuse WebAudio plugins is still under consideration: the standard APIs behind the WebComponents (HTML templates, shadow DOM, custom elements and HTML imports), is a good candidate, and webaudiomodules.org and our approach use them so far. However some browser vendors are still not supporting them as the HTML imports API is already competing with the JavaScript modules (ES6 "imports") feature and with npm, the popular package manager for the JavaScript programming language (it is the default package manager for the JavaScript runtime environment Node.js) that can also be considered for publishing/importing plugins. Other problems include the GUI of these plugins: how to deal with the different libraries and JS frameworks developers are using? In the current version of our work, we chose to rely on W3C standards exclusively. Webaudiomodules and FAUST components are compatible with our host with little adjustments and we are collaborating closely with the creators of these proposals to converge toward a unified API that could become a basis for a WebAudio plugin standard in the WebAudio API version 2.

[7] https://webaudiodemos.appspot.com/input/index.html

[8] , https://github.com/Tonejs/Tone.js, https://github.com/Theodeus/tuna, https://alemangui.github.io/pizzicato

[9] http://faust.grame.fr

4 SETTINGS FOR THE DEMO

We propose to demonstrate the first emulation of a Marshall JCM 800 guitar tube amp running in a browser, with a virtual pedal board to chain audio FX pedals. These web tools can be played real-time with a guitar, and controlled using any MIDI controller. We recommend for the best experience to use MacOS and a low latency sound card. We propose to compare our WebAudio tube amp simulation and effects with native simulations such as Guitar Rig by Native Instruments (used by many musicians and guitarists).

A typical demo is: look for a song in the WASABI database, switch to the embedded multitrack player, mute the guitar track and open the pedal board, build your own virtual guitar rig, plug your guitar, adjust the sound and play along with the other tracks, with no noticeable latency.

ACKNOWLEDGMENTS

This work was supported by the French Research National Agency (ANR) and the WASABI team (contract ANR-16-CE23-0017-01). ElMahdi Korfed and Guillaume Etevenard who helped developing these tools.

REFERENCES

[1] J. Pakarinen and D.T. Yeh. 2009. A review of digital techniques for modeling vacuum-tube guitar amplifiers. *Computer Music Journal* 33, 2 (2009), 85-100.
[2] B. Holmes and M. van Walstijn. 2015. Improving the robustness of the iterative solver in state-space modeling of guitar distortion circuitry. In *Proc. 18th Int. Conference on Digital Audio Effects* (DAFx-15). Trondheim, Norway.
[3] C.H. Chang. 2011. *DESC9115: Digital Audio Systems-Final Project Overdrive/Distortion*. In *Repository of Tech. Reports*, University of Sydney, Australia. http://hdl.handle.net/2123/7608
[4] J. Macak and J. Schimmel. 2010. Real-time guitar tube amplifier simulation using an approximation of differential equations. In *Proc. 13th Int. Conference on Digital Audio Effects* (DAFx-10). Graz, Austria.
[5] D.T. Yeh, J.S. Abel, A. Vladimirescu, and J.O. Smith. 2008. Numerical methods for simulation of guitar distortion circuits. *Computer Music Journal*, 32, 2 (2008), 23-42.
[6] D.T. Yeh and J.O. Smith. 2006. Discretization of the '59 Fender Bassman tone stack. In *Proc. 9th Int. Conference on Digital Audio Effects* (DAFx-06). Montreal, Canada.
[7] M. Buffa and J. Lebrun. 2017. Real time tube guitar amplifier simulation using WebAudio. In *Proc. 3rd Web Audio Conference* (WAC 2017). London, UK.
[8] M. Buffa and J. Lebrun. 2017. Web Audio Guitar Tube Amplifier vs Native Simulations. In *Proc. 3rd Web Audio Conference* (WAC 2017). London, UK.
[9] N. Jillings and al. 2017. Intelligent audio plugin framework for the Web Audio API. In *Proc. 3rd Web Audio Conference* (WAC 2017). London, UK.
[10] J. Kleimola and O. Larkin. 2015. Web audio modules. In *Proc. 12th Sound and Music Computing Conference* (SMC15). Maynooth, Ireland.
[11] M. Buffa, M. Demetrio, and N. Azria. 2016. Guitar pedal board using WebAudio. In *Proc. 2th Web Audio Conference* (WAC 2016). Atlanta, USA.
[12] M. Buffa and al. 2017. WASABI: a Two Million Song Database Project with Audio and Cultural Metadata plus WebAudio enhanced Client Applications. In *Proc. 3rd Web Audio Conference* (WAC 2017). London, UK.
[13] I. Cohen and T. Helie. 2010. Real-Time Simulation of a Guitar Power Amplifier. In *Proc. of the 13th Int. Conference on Digital Audio Effects* (DAFx-10). Graz, Austria.
[14] S. Letz, Y. Orlarey, and D. Fober. 2017. Compiling Faust Audio DSP Code to WebAssembly. In *Proc. 3rd Web Audio Conference* (WAC 2017). London, UK.