Sound Matching an Analogue Levelling Amplifier Using the Newton-Raphson Method

Chin-Yun Yu[1] György Fazekas[1]

[1]Centre for Digital Music
Queen Mary University of London

AES International Conference on Artificial Intelligence and Machine Learning for Audio

Table of Content

Table of Content

Virtual Analogue (VA) of Dynamic Range Controllers (DRC)

- DRCs: **Compressors**, limiters, expanders, and gates.
- Time-varying non-linear systems.
- VA: creating a digital signal processer that emulates the eletrical behaviour of a reference unit [Im24].

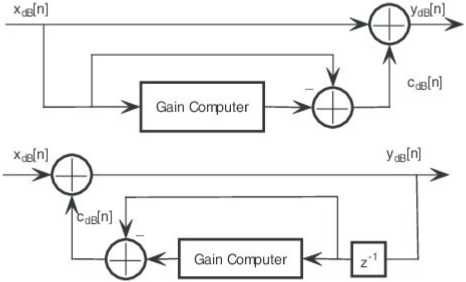

Figure: Feed-forward (upper) and feed-back (lower) DRCs. Source: [GMR12].

Motivations

- Previous grey-box methods [WV22; Yu+24] use gradient descent (GD) to optimise feed-forward compressor parameters.
- Newton-Raphson (NR) converge faster and more robustly than GD (under certain conditions).
- Feed-forward compressors has few parameters ($M < 10$) $\rightarrow$ quadratic storage cost of NR is negligible.

$\rightarrow$ Explore the NR method for *grey-box* modelling of analogue compressors[1].

[1] It has been used in white-box VA modelling [BTC15]

Contributions

1. Verify the feasibility of using the NR method for modelling the LA-2A compressor.
2. Accelerate the backpropagation method from [Yu+24] on modern GPUs using parallel algorithms
3. Present the mapping function $\mathbb{R} \to \mathbb{R}^M$ from the circuit's peak reduction to the parameters, providing an interpretable and intuitive way to control the compressor.
4. Open-sourced the resulting model, 4A-2A, as a VST plugin.

Table of Content

Target Unit: LA-2A

- An optical levelling amplifier (LA) from Teletronix, designed in the 1960s.
- Known for its smooth and musical compression.
- 2 control parameters: Peak Reduction and Gain.

Figure: The LA-2A hardware unit. Source: Wiki.

The Chosen Function f

- The differentiable feed-forward compressor from [Yu+24].
- Five parameters ($M = 5$): Threshold (CT), Ratio (R), Attack time (t_{at}), Release time (t_{rt}), Make-up gain (γ).

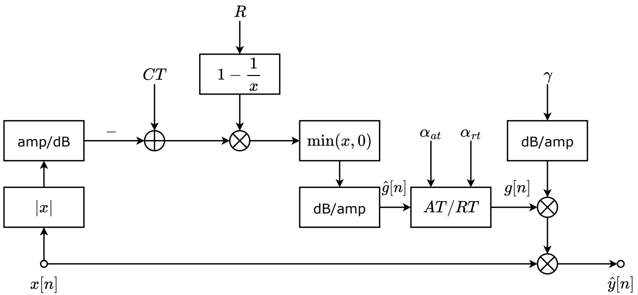

Figure: Block diagram of the feed-forward compressor.

Problem Formulation

Discrete input signal $\mathrm{x} \in \mathbb{R}^N$, target signal (sampled) $\mathrm{y} \in \mathbb{R}^N$, a chosen discrete function $f : \mathbb{R}^N \times \mathbb{R}^M \to \mathbb{R}^N$, parameters $\theta \in \mathbb{R}^M$, and a distance function $\mathcal{D} : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}_+$.

For simplicity, we denote $f_{\mathrm{x}}(\theta) := f(\mathrm{x}, \theta)$.

Objective

$$\theta^* = \arg\min_{\theta} \mathcal{D}(f_{\mathrm{x}}(\theta), \mathrm{y})$$

It is similar to *sound matching*.

The Newton-Raphson (NR) Method

- Iterative minimum-finding algorithm[2] for $\mathcal{L} : \theta \mapsto \mathcal{D}(f_x(\theta), y)$.
- Start with an initial guess θ_0.
- Update rule:
$$\theta \mapsto \theta - [\nabla^2 \mathcal{L}(\theta)]^{-1} \nabla \mathcal{L}(\theta)$$
where $\nabla \mathcal{L}(\theta) \in \mathbb{R}^M$ and $\nabla^2 \mathcal{L}(\theta) \in \mathbb{R}^{M \times M}$ are the gradient and Hessian of $\mathcal{L}(\theta)$.
- Converges quadratically if
  1. $\mathcal{L}$ is twice differentiable: ✓
  2. $\mathcal{L}$ is convex around the minimum: empirically ✓(?)

[2] In white-box VA, NR is used to solve differential equations [BTC15].

Table of Content

Jacobian Computation

$$\mathrm{y} = g(\mathrm{x}), \quad \nabla g(\mathrm{x}) = \frac{\partial \mathrm{y}}{\partial \mathrm{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_N}{\partial x_1} & \cdots & \frac{\partial y_N}{\partial x_N} \end{bmatrix}$$

- Reverse-mode Jacobian uses Vector-Jacobian product (VJP, a.k.a. backpropagation):

$$\mathrm{vjp} : \mathrm{v} \mapsto \mathrm{v}^\top \nabla \mathcal{L}(\theta), \quad \mathrm{v} \in \mathbb{R}^M$$

- Forward-mode Jacobian uses Jacobian-Vector product (JVP):

$$\mathrm{jvp} : \mathrm{v} \mapsto \nabla \mathcal{L}(\theta)\mathrm{v}$$

- When the Jacobian is wide, reverse-mode is more efficient, and vice versa.

Hessian Computation

- Hessian is the Jacobian of the gradient:

$$\nabla^2 \mathcal{L}(\theta) = \nabla(\nabla \mathcal{L}(\theta)) = \nabla h(\theta), \quad h : \mathbb{R}^M \to \mathbb{R}^M$$

- Four ways to compute the Hessian:
  1. reverse-over-reverse (VJP of VJP)
  2. forward-over-reverse (JVP of VJP, generally the best)
  3. reverse-over-forward (VJP of JVP, generally the worst)
  4. forward-over-forward (JVP of JVP)
- 1 and 2 should be equally fast (theoretically) for square Hessians.
- `torchcomp` [Yu+24] only implements VJP.

Gradient of the Ballistics

- The gain reduction $g[n] \in \mathbb{R}_+$ is computed using an time-varying one-pole filter (ballistics):

$$\underbrace{g[n]}_{\text{output}} = \begin{cases} \alpha_{at}\hat{g}[n] + (1-\alpha_{at})g[n-1] & \text{if } \hat{g}[n] < g[n-1] \\ \alpha_{rt}\hat{g}[n] + (1-\alpha_{rt})g[n-1] & \text{otherwise} \end{cases}$$

$$= \underbrace{\tilde{g}[n]}_{\text{input}} + \underbrace{\beta[n]}_{\text{multiplier}} \underbrace{g[n-1]}_{\text{previous output}}$$

where $\beta[n] = 1 - \alpha_{at}$ or $1 - \alpha_{rt}$, and $\tilde{g}[n] = (1-\beta[n])\hat{g}[n]$.

- Its backpropagation is also an one-pole [Yu+24]

$$\nabla_{\tilde{g}[n]}\mathcal{L}(\theta) = \nabla_{g[n]}\mathcal{L}(\theta) + \beta[n+1]\nabla_{\tilde{g}[n+1]}\mathcal{L}(\theta).$$

which is used for the inner VJP of $\nabla\mathcal{L}(\theta)$.

Gradient of the Ballistics (Cont.)

- The JVP of the VJP of the ballistics is also an one-pole:

$$\nabla_{g[n]}\nabla\mathcal{L}(\theta) = \nabla_{\tilde{g}[n]}\nabla\mathcal{L}(\theta) + \beta[n]\nabla_{g[n-1]}\nabla\mathcal{L}(\theta)$$

- The inner JVP and outer VJP can be expressed as one-poles

$$\nabla_\theta g[n] = [\nabla_\theta \tilde{g}[n] + \nabla_\theta \beta[n]\,(g[n-1] - \hat{g}[n])] + \beta[n]\nabla_\theta g[n-1]$$
$$\nabla_\theta \nabla_{\tilde{g}[n]}\mathcal{L}(\theta) = [\nabla_\theta \nabla_{g[n]}\mathcal{L}(\theta) + \nabla_\theta \beta[n]\nabla_{\tilde{g}[n+1]}\mathcal{L}(\theta)] + \beta[n+1]\nabla_\theta \nabla_{\tilde{g}[n+1]}\mathcal{L}(\theta)$$

Accelerating One-poles Using Parallel Associative Scan

- One-poles are inherently sequential.

$$y[n] = x[n] + \beta[n]\left(x[n-1] + \beta[n-1]\left(x[n-2] + \cdots\right)\right)$$

- Re-expressed into parallelisable form [Ble90]:

$$(*, y[n]) = \cdots \oplus (\beta[n-1], x[n-1]) \oplus (\beta[n], x[n])$$
$$(\alpha, x) \oplus (\beta, z) = (\alpha\beta, z + \alpha x)$$

- Implementation in CUDA is available [MC18].

Table of Content

Experiment Setup

- SignalTrain [CH20], paired recordings sampled at 44.1 kHz from LA-2A.
  - Peak reduction $[0, 5, 10, \ldots, 100]$, fixed gain.
  - Identical 20 min audio stimulus consisting of both musical and synthetic sounds.
- Split the audio into 12 s chunks with 1 s overlap.
- Apply a pre-emphasis filter[3] $\frac{1-z^{-1}}{1-0.995z^{-1}}$ to reduce low-frequency noise [WV22].

[3] Accelerated using parallel scan.

Analysis on Computational Cost

Table: The memory cost and runtime of different hessian computation strategies in PyTorch.

	`torch.autograd.functional.hessian`	`torch.func`			
	rev-rev	rev-rev	fwd-rev	rev-fwd	fwd-fwd
Memory (MB)	1066	2358	3534	6306	5364
Time (ms)	26.5	28.1	28.1	64.7	38.9

- RTX 3060 12GB, batch size 16.
- rev-rev and fwd-rev are equally fast, but rev-rev uses less memory.
- `torch.func` has higher overhead $\rightarrow$ `torch.autograd` is used for experiments.

Optimisation Strategy

- Damped NR method with backtracking line search [MS82].
- Initialisation: $CT = -36$ dB, $\gamma = 0$ dB, $R = 4$, $t_{at} = 1$ ms, and $t_{rt} = 200$ ms.
- Peak reductions $100 \rightarrow 95 \rightarrow 90 \rightarrow \cdots$ to ensure a good initial guess.
- training on the entire dataset (520 min audio) takes less than 20 min.

For ≥ 40 peak reductions converges to the same point in different initialisations $\rightarrow \mathcal{L}(\theta)$ is mostly convex around the optimal solution.

...The average compression ratio is always set at roughly 3:1, while the average Attack time is 10 milliseconds, and the Release time is about 60 milliseconds for 50% of the release, and anywhere from 1 to 15 seconds for the rest. [4]

[4] www.uaudio.com/blog/la-2a-collection-tips-tricks

Results: Parameter Curves

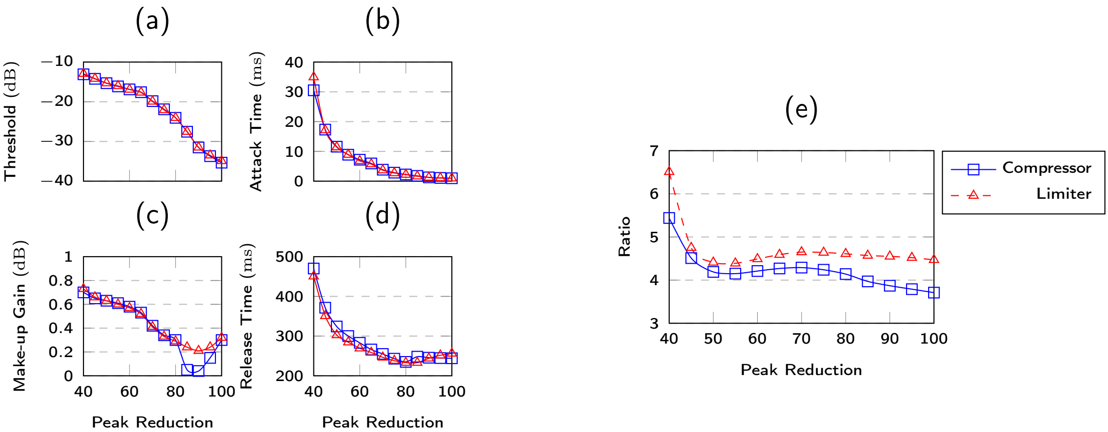

Figure: The mapping from LA-2A peak reduction to the compressor's parameters.

Evaluation Metrics

- Error-to-Signal Ratio (ESR):

$$\mathrm{ESR}(\hat{\mathrm{y}}, \mathrm{y}) = \frac{\sum_n (y[n] - \hat{y}[n])^2}{\sum_n y[n]^2}$$

- Loudness Dynamic Range [NML22] difference (ΔLDR):

$$\mathrm{LDR}(\mathrm{y}) = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1} L_{\mathrm{y}}[n]^2}, \quad L_{\mathrm{y}}[n] = 10\log_{10}\frac{\mathrm{RMS}_{\mathrm{short}}(y[n])}{\mathrm{RMS}_{\mathrm{long}}(y[n])}$$

$$\Delta\mathrm{LDR}(\hat{\mathrm{y}}, \mathrm{y}) = \mathrm{LDR}(\hat{\mathrm{y}}) - \mathrm{LDR}(\mathrm{y})$$

- $\Delta\mathrm{LDR} > 0$, it means $\hat{\mathrm{y}}$ is less compressed than y and vice versa.

Baselines

- **UAD**: The official UAD LA-2A plugin [5].
- **CLA-2A**: The CLA-2A plugin from Waves [6].
- **CA-2A**: The CA-2A plugin from Cakewalk [7].
- **4A-2A-G**: 4A-2A with a GRU as make-up gain [WV22].

[5] www.uaudio.com/uad-plugins/compressors-limiters/teletronix-la-2a-tube-compressor
[6] www.waves.com/plugins/cla-2a-compressor-limiter
[7] legacy.cakewalk.com/Products/CA-2A

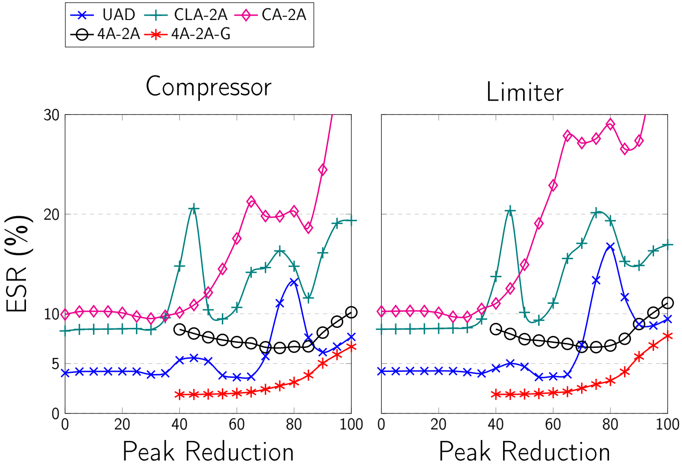

Figure: ESR of all the evaluated models under different peak reduction and mode.

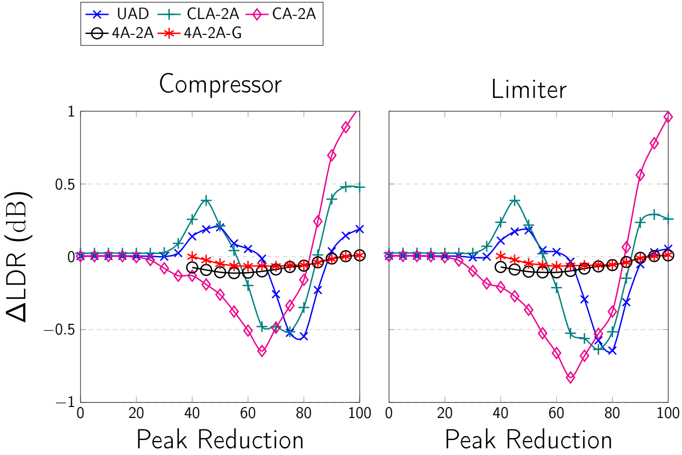

Figure: ΔLDR of all the evaluated models under different peak reduction and mode.

Table of Content

Conclusion and Future Work

- NR is a promising approach for the grey-box modelling of audio compressors, when efficient Hessian computation is available.
- 4A-2A objectively performs comparably to the best commercial LA-2A plugins while providing interpretable control.

Future directions:

- Subjective evaluation.
- Explore feed-back digital compressors as f.
- Apply to other analogue compressors (etc. 1176).
- Model LA-2A's two-stage release behaviour.

Links

Figure: Source code, experiments, and VST plugin are open-sourced under the MPL-2.0 license at github.com/aim-qmul/4a2a.

References

[Ble90] Guy E. Blelloch. *Prefix Sums and Their Applications*. Tech. rep. CMU-CS-90-190. School of Computer Science, Carnegie Mellon University, Nov. 1990.

[BTC15] Stefan Bilbao, Alberto Torin, and Vasileieos Chatziioannou. "Numerical Modeling of Collisions in Musical Instruments". English. In: *Acta Acustica united with Acustica* 101.1 (Jan. 2015), pp. 155–173.

[CH20] Benjamin Colburn and Scott Hawley. *SignalTrain LA2A Dataset*. Version 1.1. May 2020. DOI: 10.5281/zenodo.3824876. URL: https://doi.org/10.5281/zenodo.3824876.

[GMR12] Dimitrios Giannoulis, Michael Massberg, and Joshua D Reiss. "Digital dynamic range compressor design—A tutorial and analysis". In: *Journal of the Audio Engineering Society* 60.6 (2012), pp. 399–408.

[Im24] Sohyun Im. *Introductory Guide to Virtual Analog Modelling: Intersection of Analog and Digital Audio Processing*. Audio Developer Conference. 2024. URL: https://youtu.be/ds0DnRkhMNE.

[MC18] Eric Martin and Chris Cundy. "Parallelizing Linear Recurrent Neural Nets Over Sequence Length". In: *ICLR*. 2018.

[MS82] Jorge J Moré and Danny C Sorensen. *Newton's method*. Tech. rep. Argonne National Lab.(ANL), Argonne, IL (United States), 1982.

[NML22] Shahan Nercessian, Russell McClellan, and Alexey Lukin. "A direct microdynamics adjusting processor with matching paradigm and differentiable implementation". en. In: *Proc. DAFx*. 2022.

[WV22] Alec Wright and Vesa Välimäki. "Grey-box modelling of dynamic range compression". In: *Proc. DAFx*. 2022, pp. 304–311.

[Yu+24] Chin-Yun Yu et al. "Differentiable All-pole Filters for Time-varying Audio Systems". In: *Proc. DAFx*. 2024, pp. 345–352.