Energy-based learning

SCO colloquium, 13 May 2025

Henk van Waarde

Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence

Jan C. Willems Center for Systems and Control

University of Groningen

Based on joint work with: Anne-Men Huijzer, Tom Chaffey and Bart Besselink

Problem: digital computers consume enormous amounts of energy

- ChatGPT: ± 3 Wh per query, ± 3 GWh per day

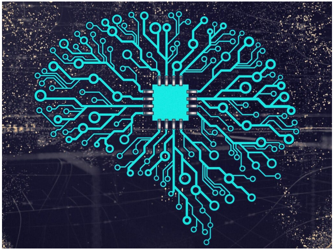
Alternative: analog computing

- discrete values $\rightarrow$ analog signals (voltages/currents)

Prominent example: neuromorphic computing

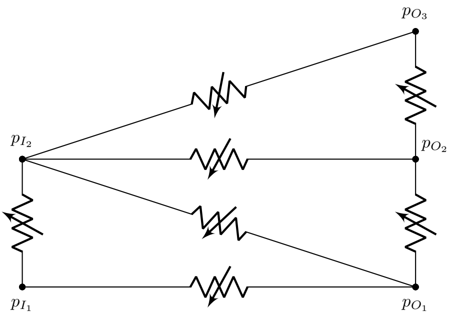
- aim: create circuit elements that behave like biological neurons

"the brain is a factor of 1 billion more efficient than our present digital technology"
–Carver Mead[1]



[1] C. Mead, *Neuromorphic electronic systems*, Proc. IEEE, 78(10): pp 1629–1636, 1990.

This talk: learning from input-output data in resistive electrical circuits



The data: a pair of voltage potentials (p_I, p_O^D)

Goal: Adjust the conductances of the resistors so that the circuit maps p_I to p_O^D

- **Tool**: **Energy-based learning algorithms**

Problem formulation

Graph theory:

- $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ connected undirected graph
- $\mathcal{V} = \{1, 2, \ldots, N\}$ set of nodes
- $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ set of B branches
- $D \in \mathbb{R}^{N \times B}$ incidence matrix

Circuit theory:

- $p \in \mathbb{R}^N$ vector of voltage potentials at nodes
- $j \in \mathbb{R}^N$ nodal currents entering each node
- $v \in \mathbb{R}^B$ voltages across the branches
- $g \in \mathbb{R}^B$ vector of positive conductances
- $G := \mathrm{diag}(g) \in \mathbb{R}^{B \times B}$ diagonal matrix of conductances

---

Using the laws of Kirchhoff and Ohm, we get:

$$j = DGD^\top p$$

where $DGD^\top$ is the Laplacian matrix of $\mathcal{G}$.

Partitioned matrices:

- Input nodes $\mathcal{V}_I$ and outputs $\mathcal{V}_O$ such that $\mathcal{V} = \mathcal{V}_I \cup \mathcal{V}_O$ and $\mathcal{V}_I \cap \mathcal{V}_O = \varnothing$
- Define $N_I := |\mathcal{V}_I|$ and $N_O := |\mathcal{V}_O|$
- Partition:

$$p = \begin{bmatrix} p_I \\ p_O \end{bmatrix}, \; j = \begin{bmatrix} j_I \\ j_O \end{bmatrix}, \text{ and } D = \begin{bmatrix} D_I \\ D_O \end{bmatrix},$$

with $p_I, j_I \in \mathbb{R}^{N_I}$, $p_O, j_O \in \mathbb{R}^{N_O}$, $D_I \in \mathbb{R}^{N_I \times B}$ and $D_O \in \mathbb{R}^{N_O \times B}$.

Assumptions on currents:

- Sources at input nodes, leading to j_I. Output currents: $j_O = 0$.

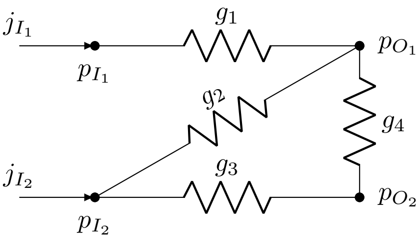
Thus,

$$\begin{bmatrix} j_I \\ 0 \end{bmatrix} = \begin{bmatrix} D_I G D_I^\top & D_I G D_O^\top \\ D_O G D_I^\top & D_O G D_O^\top \end{bmatrix} \begin{bmatrix} p_I \\ p_O \end{bmatrix},$$

leading to: $p_O = -(D_O G D_O^\top)^{-1} D_O G D_I^\top p_I$.

Note: $D_O G D_O^\top$ is invertible because $\mathcal{G}$ is connected (Laplacian has kernel $\operatorname{im} \mathbf{1}$).

Example:



$$D = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ \hdashline -1 & -1 & 0 & 1 \\ 0 & 0 & -1 & -1 \end{array}\right] = \left[\begin{array}{c} D_I \\ \hdashline D_O \end{array}\right]$$

In this case, $\begin{bmatrix} p_{O1} \\ p_{O2} \end{bmatrix} = \begin{bmatrix} g_1+g_2+g_4 & -g_4 \\ -g_4 & g_3+g_4 \end{bmatrix}^{-1} \begin{bmatrix} g_1 & g_2 \\ 0 & g_3 \end{bmatrix} \begin{bmatrix} p_{I1} \\ p_{I2} \end{bmatrix}$

Total power in the network:

$$v^\top G v = p^\top D G D^\top p = (D_I^\top p_I + D_O^\top p_O)^\top G (D_I^\top p_I + D_O^\top p_O).$$

Fact: Given p_I, the vector p_O is the one minimizing the total power:

$$p_O = \arg\min_{x \in \mathbb{R}^{N_O}} (D_I^\top p_I + D_O^\top x)^\top G (D_I^\top p_I + D_O^\top x).$$

For $\epsilon > 0$, define the set

$$\mathcal{C}_\epsilon := \{x \in \mathbb{R}^B \,|\, x_k \geqslant \epsilon,\, k = 1, 2, \ldots, B\}.$$

We will assume that $g \in \mathcal{C}_\epsilon$ can be **adjusted**

To emphasize dependence on g, we write $p_O(g) = -(D_O G D_O^\top)^{-1} D_O G D_I^\top p_I$

By Kirchhoff's voltage law:

$$v(g) = D^\top \begin{bmatrix} p_I \\ p_O(g) \end{bmatrix} = D^\top \begin{bmatrix} I \\ -(D_O G D_O^\top)^{-1} D_O G D_I^\top \end{bmatrix} p_I$$

Problem: Given p_I and desired output potentials p_O^D, find a sequence $\left(g^t\right)_{t=0}^\infty$ in $\mathcal{C}_\epsilon$, where each g_k^{t+1} is determined locally (using g_k^t and $v_k(g^t)$), such that

$$g^t \to g^* \;\text{ as }\; t \to \infty$$

for some $g^* \in \mathcal{C}_\epsilon$ satisfying $p_O(g^*) = p_O^D$.

Assumption: Throughout the talk we assume that such g^* exists.

First proposed by Hopfield[2] and Hinton et al.[3]

Ingredients of the energy-based learning model:

1. a parameter vector θ
2. a vector of input variables x
3. a vector of hidden variables h
4. a vector of output variables o
5. an energy function E, so that $E : (\theta, x, h, o) \mapsto e \in \mathbb{R}$

Given θ and x, the **hidden** and **output variable** are defined as:

$$(h_*, o_*) := \arg\min_{(h,o)} E(\theta, x, h, o).$$

Training goal: Given input-output data (x, y), find θ so that $o_* = y$.

[2] J.J. Hopfield, *Neurons with graded response have collective computational properties like those of two-state neurons*, Proc. of the national academy of sciences, 81(10):3088-3092, 1984.

[3] G.E. Hinton et al., *Boltzmann machines: constraint satisfaction networks that learn*, technical report, Carnegie-Mellon University, Department of Computer Science Pittsburgh, PA, 1984.

Different energy-based learning algorithms:

- Contrastive learning[4]
- Equilibrium propagation[5]
- Coupled learning[6]

Leitmotif: contrast two states of the model and update θ iteratively

---

We focus on **contrastive learning**. Given θ, consider two states:

1. Free state: Fix x. This yields hidden and output variables (h_*, o_*)
2. Clamped state: Fix both input x and output y. Yields hidden variable

$$h_*^{\mathsf{CL}} := \arg\min_h E(\theta, x, h, y).$$

Parameter update: For $\gamma > 0$, the learning rule for the parameters is:

$$\theta^{\mathsf{new}} = \theta - \gamma \left(\frac{\partial E}{\partial \theta}(\theta, x, h_*^{\mathsf{CL}}, y) - \frac{\partial E}{\partial \theta}(\theta, x, h_*, o_*) \right)$$

[4] J.R. Movellan, *Contrastive Hebbian learning in the continuous Hopfield model*, Connectionist models, pp.10-17, Elsevier, 1991.

[5] B. Scellier and Y. Bengio, *Equilibrium propagation: bridging the gap between energy-based models and backpropagation*, Frontiers in computational neuroscience, 11:24,2017.

[6] Stern et al., *Supervised learning in physical networks: from machine learning to learning machines*, Physical Review X, 11(2):021045, 2021.

Parameter update: Let $\gamma > 0$. The learning rule for the parameters is:

$$\theta^{\text{new}} = \theta - \gamma \left(\frac{\partial E}{\partial \theta}(\theta, x, h_*^{\text{CL}}, y) - \frac{\partial E}{\partial \theta}(\theta, x, h_*, o_*) \right)$$

Interpretation: $\frac{dE}{d\theta} = \frac{\partial E}{\partial \theta} + \left(\frac{\partial h}{\partial \theta}\right)^\top \frac{\partial E}{\partial h} + \left(\frac{\partial o}{\partial \theta}\right)^\top \frac{\partial E}{\partial o}$.
Thus, by definition of h_* and o_*,

$$\frac{\partial E}{\partial h}(\theta, x, h_*, o_*) = 0 \quad \text{and} \quad \frac{\partial E}{\partial o}(\theta, x, h_*, o_*) = 0.$$

$\Longrightarrow \frac{dE}{d\theta}(\theta, x, h_*, o_*) = \frac{\partial E}{\partial \theta}(\theta, x, h_*, o_*)$. Also, $\frac{dE}{d\theta}(\theta, x, h_*^{\text{CL}}, y) = \frac{\partial E}{\partial \theta}(\theta, x, h_*^{\text{CL}}, y)$.

The point: Define the contrastive function

$$Q(\theta, x, y) := E(\theta, x, h_*^{\text{CL}}, y) - E(\theta, x, h_*, o_*).$$

Then the learning rule is:

$$\theta^{\text{new}} = \theta - \gamma \frac{dQ}{d\theta}(\theta, x, y).$$

So contrastive learning **performs gradient descent** on Q!

Inspired by the energy-based learning paradigm, we define the contrastive function:

$$Q(g) := (v^D)^\top G v^D - v(g)^\top G v(g)$$

where $v^D := D^\top \begin{bmatrix} p_I \\ p_O^D \end{bmatrix}$ and $v(g) = D^\top \begin{bmatrix} p_I \\ p_O(g) \end{bmatrix}$.

Using the fact that $\frac{dQ}{dg}(g) = \frac{\partial Q}{\partial g}(g)$ we obtain:

$$\frac{dQ}{dg}(g) = (v^D)^2 - v(g)^2,$$

where, for $x \in \mathbb{R}^B$, we define

$$x^2 := \begin{bmatrix} x_1^2 \\ x_2^2 \\ \vdots \\ x_B^2 \end{bmatrix}.$$

The contrastive learning rule is thus:

$$g^0 \in \mathcal{C}_\epsilon, \text{ and } g^{t+1} = g^t - \gamma((v^D)^2 - v(g^t)^2) \text{ for } t = 0, 1, 2, \ldots$$

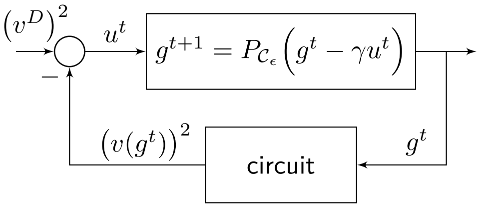
However, this does not ensure that $g^{t+1} \in \mathcal{C}_\epsilon$...

Definition: Let $\mathcal{C} \subseteq \mathbb{R}^n$ be nonempty, closed and convex. The projection $P_{\mathcal{C}} : \mathbb{R}^n \to \mathcal{C}$ is defined as

$$P_{\mathcal{C}}(x) := \arg\min_{\hat{x} \in \mathcal{C}} \|\hat{x} - x\|.$$

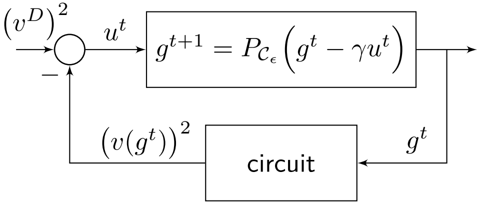
Projected gradient descent algorithm: Let $\gamma > 0$ and $g^0 \in \mathcal{C}_\epsilon$. Define

$$g^{t+1} = P_{\mathcal{C}_\epsilon}\left(g^t - \gamma((v^D)^2 - v(g^t)^2)\right) \quad \text{for } t = 0, 1, 2, \ldots$$



Projected gradient descent (PGD) algorithm: Let $\gamma > 0$ and $g^0 \in \mathcal{C}_\epsilon$. Define

$$g^{t+1} = P_{\mathcal{C}_\epsilon}\left(g^t - \gamma((v^D)^2 - v(g^t)^2)\right) \quad \text{for} \quad t = 0, 1, 2, \ldots$$



Comments:

1. Distributed algorithm using local update rules because:

$$g_k^{t+1} = \max\{\epsilon, g_k^t - \gamma((v_k^D)^2 - v_k(g^t)^2)\}$$

   for $k = 1, 2, \ldots, B$.

2. Same circuit is used for training (i.e., updating g^t) and inference

---

Main open question: does this algorithm **converge** (to something meaningful)**?**

Let $\mathcal{C} \subseteq \mathbb{R}^n$ be nonempty, closed and convex. We view our PGD algorithm as a fixed-point iteration:

$$x^0 \in \mathcal{C}, \quad x^{t+1} = f(x^t),$$

where $f : \mathcal{C} \to \mathcal{C}$ and $x^t \in \mathcal{C}$ for $t = 0, 1, \ldots$.

Definition: The set of fixed points of f is defined as

$$\operatorname{Fix} f := \{x \in \mathcal{C} \,|\, x = f(x)\}.$$

Definition: The function f is called **Lipschitz continuous** if there exists $L \geqslant 0$ such that

$$||f(x) - f(y)|| \leqslant L||x - y||$$

for all $x, y \in \mathcal{C}$. It is **nonexpansive** if it is Lipschitz continuous with $L = 1$.

Definition: We say f is **averaged** if there exists a nonexpansive $\bar{f} : \mathcal{C} \to \mathcal{C}$ and an $\alpha \in (0, 1)$ such that $f(x) = \alpha x + (1 - \alpha)\bar{f}(x)$ for all $x \in \mathcal{C}$.

Theorem (Krasnosel'skiĭ-Mann): Assume that f is averaged and $\operatorname{Fix} f \neq \varnothing$. Then, as $t \to \infty$, $x^t \to x^*$ for some $x^* \in \operatorname{Fix} f$.

Idea: Now apply this with $\mathcal{C} = \mathcal{C}_\epsilon$ and $f : \mathcal{C}_\epsilon \to \mathcal{C}_\epsilon$ defined by

$$f(g) = P_{\mathcal{C}_\epsilon}\left(g - \gamma \frac{dQ}{dg}(g)\right).$$

Note: This f is the composition of $P_{\mathcal{C}_\epsilon}$ and $g \mapsto g - \gamma \frac{dQ}{dg}(g)$.

Fact: Let $f_1, f_2 : \mathcal{C} \to \mathcal{C}$ be averaged. Then $f_1 \circ f_2$ is averaged.

Fact[7]: $P_{\mathcal{C}}$ is averaged for any nonempty closed convex set $\mathcal{C}$.

Fact[8]: If Q is convex and $\frac{dQ}{dg}$ is Lipschitz continuous with constant L, then the function $g \mapsto g - \gamma \frac{dQ}{dg}(g)$ is averaged for any $\gamma \in (0, \frac{2}{L})$.

Question: How to show convexity and Lipschitz continuity?

[7] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, CMS Books in Mathematics, New York, NY: Springer New York, 2011.

[8] E. K. Ryu and W. Yin, *Large-Scale Convex Optimization: Algorithms and Analyses via Monotone Operators*, Cambridge University Press, 2022.

Lemma[9]: The function $Q : \mathcal{C}_\epsilon \to \mathbb{R}$ is convex.

Idea of the proof: The Hessian matrix

$$H = \begin{bmatrix} \frac{\partial^2 Q}{\partial g_1^2} & \frac{\partial^2 Q}{\partial g_1 \partial g_2} & \cdots & \frac{\partial^2 Q}{\partial g_1 \partial g_B} \\ \frac{\partial^2 Q}{\partial g_2 \partial g_1} & \frac{\partial^2 Q}{\partial g_2^2} & \cdots & \frac{\partial^2 Q}{\partial g_2 \partial g_B} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 Q}{\partial g_B \partial g_1} & \frac{\partial^2 Q}{\partial g_B \partial g_2} & \cdots & \frac{\partial^2 Q}{\partial g_B^2} \end{bmatrix}$$

has the nice formula $H(g) = 2\,\mathrm{diag}\big(v(g)\big) D_O^\top (D_O G D_O^\top)^{-1} D_O\,\mathrm{diag}\big(v(g)\big)$.

This implies $H(g) \geqslant 0$ for all $g \in \mathcal{C}_\epsilon$, thus Q is convex!

Lemma: The function $\frac{dQ}{dg}$ is Lipschitz continuous on $\mathcal{C}_\epsilon$ with

$$L := \frac{2}{\epsilon} \left(\|D_I\| + \sqrt{N_I N_O} \|D_O\| \right)^2 \|p_I\|^2.$$

[9] M.A. Huijzer, T. Chaffey, B. Besselink, and H.J. van Waarde, *Convergence of energy-based learning in linear resistive networks*, https://arxiv.org/abs/2503.00349, 2025.

To summarize:

- The function $Q : \mathcal{C}_\epsilon \to \mathbb{R}$ is convex
- The function $\frac{dQ}{dg}$ is Lipschitz continuous on $\mathcal{C}_\epsilon$ with

$$L := \frac{2}{\epsilon}\left(\|D_I\| + \sqrt{N_I N_O}\|D_O\|\right)^2 \|p_I\|^2.$$

Corollary: The function $g \mapsto g - \gamma \frac{dQ}{dg}(g)$ is averaged for any $\gamma \in (0, \frac{2}{L})$.

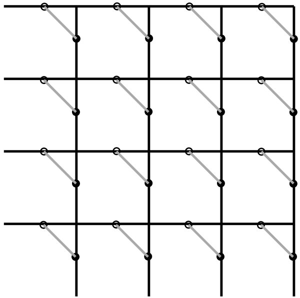
Theorem[10]: Let $\gamma \in (0, \frac{2}{L})$ and $g^0 \in \mathcal{C}_\epsilon$. Define

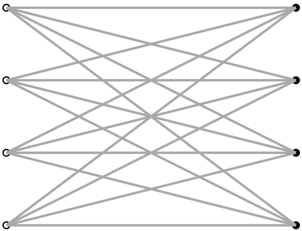
$$g^{t+1} = P_{\mathcal{C}_\epsilon}\left(g^t - \gamma((v^D)^2 - v(g^t)^2)\right) \quad \text{for} \quad t = 0, 1, 2, \dots$$

As $t \to \infty$, $g^t \to g^*$ where $g^* \in \mathcal{C}_\epsilon$ is such that $p_O(g^*) = p_O^D$.

So the **contrastive learning algorithm** solves our problem!

[10] M.A. Huijzer, T. Chaffey, B. Besselink, and H.J. van Waarde, *Convergence of energy-based learning in linear resistive networks*, https://arxiv.org/abs/2503.00349, 2025.



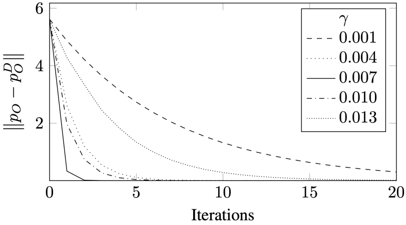


Crossbar array: $\mathcal{G}$ is a complete bipartite graph

Often used for **matrix-vector multiplication**

We consider $N_I = 40$, $N_O = 30$, $\epsilon = 0.1$

$$p_I = \begin{bmatrix} 1 & 2 & \cdots & 40 \end{bmatrix}^\top$$



Summary:

- Applied **energy-based learning** to a linear resistive circuit
- Proved convergence
  1. Contrastive function Q is convex
  2. Gradient $\frac{dQ}{dg}$ is Lipschitz continuous
- Stochastic projected gradient descent in case of multiple samples:

$$(p_{I,k}, p^D_{O,k}) \text{ for } k = 1, 2, \ldots, n.$$

Ongoing and future work:

- Nonlinear resistors
- Dynamics (capacitors/inductors)
- Hidden layers

Thank you!

Convergence of energy-based learning in linear resistive networks

Anne-Men Huijzer, Thomas Chaffey, Bart Besselink and Henk J. van Waarde

***Abstract*— Energy-based learning algorithms are alternatives to backpropagation and are well-suited to distributed implementations in analog electronic devices. However, a rigorous theory of convergence is lacking. We make a first**

in analog electronics was first investigated in the 1980s [14]–[19], and has seen a recent resurgence. This is, in part, due to the ability of analog circuits to perform inference many times faster than conventional neural networks [20]–[22].