AEGD – adaptive gradient descent with energy

Hailiang Liu

Iowa State University
Department of Mathematics

Mathematics and Deep Learning Collective
Iowa State University, October 16, 2020.
Deep learning

Al aims to create intelligent machines that work and react like humans.

- Data \((X, \hat{Y}) + \cdots\)
- Data propagation [Neural network architecture]

\[
T : X \rightarrow Y(X, \theta)
\]

- Learning by minimizing the network loss [Training]

\[
\theta = \text{argmin} f(\theta), \quad f(\theta) = L(Y(X, \theta), \hat{Y}).
\]

- Applications [Generalize well]
Outline

- Selection dynamics for deep neural networks (w/ Peter Markowich)
  - motivated by evolutionary Biology
- AEGD – adaptive gradient descent with energy (w/ Xuping Tian)
  - motivated by classical Mechanics
Selection dynamics

Collaborator: Peter Markowich (KAUST)


- Derive a new PDE framework via the ‘thermodynamic’ limit
- Rigorous analysis of the forward problem
- Derive three optimality conditions via optimal control
- Propose two training algorithms
A neural network can be seen as a recursively defined function $\Phi$ on $\mathbb{R}^d$ into $\mathbb{R}^{N_L}$:

$$\Phi = L_L \circ F_{L-1} \cdots L_2 \circ F_1 \circ L_1.$$ 

Here $L_k$ is an affine linear map from $\mathbb{R}^{N_{k-1}}$ to $\mathbb{R}^{N_k}$ ($N_0 = d$):

$$L_k(x) = a_k - B_k x$$

where $B_k$ are $N_k \times N_{k-1}$ matrices (network weights) and $a_k$ are $N_k$-vectors (network biases). $N_0 = d$, and $F_k$ is a nonlinear mapping from $\mathbb{R}^{N_k}$ into itself.

Multilayer feedforward networks are known to be a class of universal approximators. Cybernko (1988), Kurt Hornik (1989)
For the residual network [He et al 16], the simplest form for one layer is given by

\[ z^{l+1} = z^l + \tau \sigma(a_l - B_l z_l) \]

which can be seen as an explicit Euler step of the ODE system

\[ \dot{z}(t) = \sigma(a(t) - B(t)z(t)) \]

where \( t > 0 \) corresponds to the depth of the network. By now this is a fairly common procedure in DNNs [Weinan E 17]. In practical applications the dimensions \( N_0 \cdots N_l \) vary significantly from one network layer to the next. The approach we take is to pose the ODE system on an infinitely dimensional space of continuously defined functions.
The ‘thermodynamic’ limit

The forward problem amounts to modeling and simulating the propagation of data.

- $t$: depth of the network; $y$: neuron identifier
- $\sigma$: nonlinear activation function
- $b = b(y, z, t)$: the selection weight function, $a = a(y, t)$: the bias function

$$\partial_t f(y, t) = \sigma \left( a(y, t) - \int_{z \in Y} b(y, z, t)f(z, t)dz \right)$$

The input learning data set $f(y, t = 0) = f_i(y)$ then serves as the initial data.
Analysis of forward problems

- A general existence result
- Characterization of steady states
- Stability of steady states
- Large time asymptotics
- On solutions for the ReLu activation

Selection dynamics in data is induced by the interaction of neurons!
Optimal control problem

- The learning problem is to train network parameters \((a, b)\):

\[
\min J(f(\cdot, T), \tilde{f}(\cdot))
\]

such that

\[
\partial_t f(y, t) = \sigma \left( a(y, t) - \int_Y b(y, z, t)f(z, t)dz \right), \quad t \in (0, T],
\]

where \(J\) is a suitable choice of objective/loss function.

- This problem is challenging for several reasons:
  - It is a high dimensional and non-convex optimization
  - Computational costs per example are high
  - Very deep architectures are prone to various problems in the training process.
Here is how we compute the gradient:

(a) solve
\[ f_t = \sigma(a - B_b f), \quad 0 < t \leq T, \quad f(t = 0) = f_0 \]
for \( f = f_{a,b} = f_{a,b}(y, t), \ u_{a,b} := a - B_b f_{a,b}; \)

(b) solve
\[ r_s = B_b^\top (\sigma'(u_{a,b}(s)) r(s)), \quad 0 \leq s < T, \]
\[ r(s = T) = f_{a,b}(T) - \tilde{f} \]
for \( r = r_{a,b} = r_{a,b}(y, s). \) Then

**Proposition 0.1**

(i) \( D_a J(a, b)(y, s) = \sigma'(u_{a,b}(y, s)) r_{a,b}(y, s), \)

(ii) \( D_b J(a, b)(y, z, s) = -f_{a,b}(z, s) \sigma'(u_{a,b}(y, s)) r_{a,b}(y, s). \)
Pontryagin Maximum Principle

Define
\[ I(a, b) = -\frac{1}{2} \int_Y (f_{a,b}(y, T) - \tilde{f}(y))^2 \, dy. \]

Look for
\[ \max_{(a,b) \in A} I(a, b) = I(a^*, b^*). \]

Also we define the Hamiltonian
\[ H(f, r, a, b) := \int_Y \sigma(a - B_b f) r \, dy, \]
where \( r \) is the co-state variable. Then \((a^*, b^*)\) satisfies the maximum-principle.

\[ H(f^*, r^*, a^*, b^*) = \max_{(a,b) \in A} H(f^*, r^*, a, b) = \max_{(a,b) \in A} \int_Y \sigma(a - B_b f^*) r^* \, dy. \]
We now present an alternative approach based on the dynamic programming principle. Consider

$$
\partial_s f(y, s) = \sigma(a(y, s) - (B_b f)(y, s)), \quad t < s \leq T,
$$

$$
f(y, t) = v(y)
$$

for general $\nu(\cdot) \in L^2(Y)$. Let a general cost functional be

$$
J_{\nu, t}(a, b) = \int_t^T \int_Y L(f(y, s), a, b) dy ds + \frac{1}{2} \int_Y (f(y, T) - \tilde{f})^2 dy,
$$

and a value functional be

$$
F(\nu, t) = \inf_{(a, b) \in A} J_{\nu, t}(a, b) = J_{\nu, t}(a^*, b^*).$$
Then

**Theorem 0.2**

Assume the value functional $F$ is smooth in its arguments $(v, t)$. Then $F(v, t)$ solves the functional Hamilton-Jacobi-Bellman (HJB) equation

$$
\partial_t F(v, t) + \min_{(a, b) \in A} \left\{ \int_Y D_v F(v, t) \sigma(a - B_b v) dy + \int_Y L(v, a, b) dy \right\} = 0
$$

with the terminal condition

$$
F(v, T) = \frac{1}{2} \int_Y (v(y) - \tilde{f}(y))^2 dy.
$$
Consider the following finite-sum optimization problem

\[
\min_\theta \left\{ f(\theta) = \frac{1}{N} \sum_{i=1}^{N} f_i(\theta), \; \theta \in \mathbb{R}^n \right\},
\]

where \( f_i(\theta) := L(F(x_i; \theta); y_i) \) is the loss of a given Machine Learning (ML) model \( F(\cdot; \theta) \) on the training data \( \{x_i, y_i\} \), parametrized by \( \theta \).

- Propose a novel optimization algorithm using energy
- Develop a convergence theory
- Experimental results
Gradient Descent (GD)

- Explicit form (Cauchy 1847)
  \[ \theta^{k+1} = \theta^k - \eta \nabla_{\theta} f(\theta^k), \]
  
  easy implementation; convergence issues

- Implicit form
  \[ \theta^{k+1} = \theta^k - \eta \nabla_{\theta} f(\theta^{k+1}). \]
  
  This is equivalent to the proximal point algorithm (PPA)\(^1\):
  \[ \theta^{k+1} = \text{argmin}_{\theta} \left\{ f(\theta) + \frac{1}{2\eta} \| \theta - \theta^k \|^2 \right\}. \]
  
  monotonically decreasing for any \( \eta > 0 \); implicit nonlinearity

Can we adapt the standard GD (explicit form) and improve convergence and stability with arbitrary step size as in PPA but easy to implement?

\(^1\)R. Rockafellar, 1976
In the face of many significant developments, the age-old stochastic gradient descent (SGD) algorithm (1951) remains one of the most popular method for training DNNs. Finding new and simple methods that boost the performance of state of the art algorithms remains one of the most pressing problems in machine learning.

- IEQ for 4th order nonlinear PDEs [Yang 2016, Liu-Yin 2020]
- SGD with Momentum: [Polyak64]
- SGD with adaptive size [Tieleman-Hinton12, Duchi 11, Zeiler12, Ruder16]
- ADAM (Kingma-Ba15) is SGD with both momentum and adaptive size
- Nesterov accelerated gradient (NAG) (Nesterov, 1983)
- Variance reduction methods [Defazio et al 2014 ..]
Recall: momentum helps to speed up convergence and reduce oscillation, say by the HB\(^2\) scheme of form:

\[ \theta^{k+1} = \theta^k - \eta \nabla_\theta f(\theta^k) + \mu (\theta^k - \theta^{k-1}), \]

where \(0 < \mu < 1\) is a constant. This upon passing limit \(\eta \to 0\) and \(\mu \to 1\) with \(\alpha = \frac{\eta}{(1-\mu)^2}\) kept fixed leads to the second order ODE of form

\[ \alpha \ddot{\theta} + \dot{\theta} = -\nabla f, \]

The faster convergence and reduced oscillation with momentum may be viewed as driven by the energy dissipation law \(\dot{E} = -|\dot{\theta}|^2 \leq 0\) with the energy \(E(t) = f(\theta(t)) + \frac{\alpha}{2} |\dot{\theta}|^2\).

\(^2\)B. Polyak, 1964
Adaptive Gradient Descent with Energy (AEGD)

Define $g(\theta) = \sqrt{f(\theta)} + c$ with $c$ chosen so that $f(\theta) + c > 0$. Introduce $r = g(\theta)$, then $r^2$ plays the role of energy and

$$\nabla f = 2r \nabla g, \quad r = g(\theta).$$

Based on this we introduce the following update rule:

$$\theta^{k+1} = \theta^k - 2\eta r^{k+1} \nabla g(\theta^k), \quad k = 0, 1, 2, \cdots$$

$$r^{k+1} - r^k = \nabla g(\theta^k) \cdot (\theta^{k+1} - \theta^k).$$

The most striking feature of AEGD is that it is unconditionally energy stable in the sense that $r^k$ is strictly decreasing in $k$. 
AEGD and element-wise AEGD

This is actually a linear algorithm and easy to implement by

\[ r^{k+1} = \frac{r^k}{1 + 2\eta|\nabla g(\theta^k)|^2}, \quad r^0 = \sqrt{f(\theta^0) + c} \]

\[ \theta^{k+1} = \theta^k - 2\eta r^{k+1} \nabla g(\theta^k), \quad k = 0, 1, 2, \ldots \]

In order to allow the use of different learning rate for each direction, we also consider an element-wise AEGD, set forth as follows:

\[ r_{i}^{k+1} = \frac{r_{i}^k}{1 + 2\eta (\partial_i g(\theta^k))^2}, \quad i \in [n], r_{i}^0 = \sqrt{f(\theta^0) + c} \]

\[ \theta_{i}^{k+1} = \theta_{i}^k - 2\eta r_{i}^{k+1} \partial_i g(\theta^k), \quad k = 0, 1, 2, \ldots \]
Consider \( \min \{ f(\theta), \theta \in \mathbb{R}^n \} \), where \( f(\theta) \) is differentiable and bounded from below so that \( f(\theta) + c > 0 \) for some \( c > 0 \). Then

(i) AEGD is unconditionally energy stable in the sense that for any step size \( \eta > 0 \),

\[
(r^{k+1})^2 = (r^k)^2 - (r^{k+1} - r^k)^2 - \eta^{-1}\|\theta^{k+1} - \theta^k\|^2,
\]

\( r^k \) is strictly decreasing and convergent with \( r^k \to r^* \) as \( k \to \infty \), and also

\[
\lim_{k \to \infty} \|\theta^{k+1} - \theta^k\| = 0, \quad \sum_{j=0}^{\infty} \|\theta^{j+1} - \theta^j\|^2 \leq \frac{1}{\eta}((r^0)^2 - (r^*)^2).
\]
Theorem 1

(ii) Element-wise AEGD is unconditionally energy stable in the sense that for any step size $\eta > 0$,

$$(r_{i}^{k+1})^2 = (r_{i}^{k})^2 - (r_{i}^{k+1} - r_{i}^{k})^2 - \eta^{-1}(\theta_{i}^{k+1} - \theta_{i}^{k})^2, \quad i \in [n],$$

$r_{i}^{k}$ is strictly decreasing and convergent with $r_{i}^{k} \to r_{i}^{*}$ as $k \to \infty$, and also

$$\lim_{k \to \infty} \|\theta^{k+1} - \theta^{k}\| = 0, \quad \sum_{j=0}^{\infty} \|\theta^{j+1} - \theta^{j}\|^2 \leq \eta \sum_{i=1}^{n} ((r_{i}^{0})^2 - (r_{i}^{*})^2).$$
A closer look at convergence

AEGD can be reformulated as GD with a variable step size

$$\theta^{k+1} = \theta^k - \eta_k \nabla f(\theta^k), \quad \eta_k := \eta \frac{r^{k+1}}{g(\theta^k)}.$$  

This with $L$-smoothness of $f$ yields

$$f(\theta^{k+1}) \leq f(\theta^k) - \left( \frac{1}{\eta_k} - \frac{L}{2} \right) \|\theta^{k+1} - \theta^k\|^2 < f(\theta^k),$$

when $\eta_k$ is small enough so that $\eta_k < 2/L$. Therefore, AEGD of the above form is expected to enjoy the same tight convergence rates as GD does.
After finite number of iterations $\eta_j$ can be ensured (by choosing $\eta$ to be suitably small if necessary) to fall below a threshold under which $f(\theta^j)$ turns into a strictly decreasing sequence, hence convergent.

**Figure:** Effective learning rate of the 2D-Rosenbrock function
Suppose $f$ is differentiable and bounded from below. Let $\theta^k$ be the $k$-th iterate generated by the AEGD (with $\eta_j = \eta \frac{r^{j+1}}{g(\theta^j)}$). Set

$$\bar{\eta}_{k-m} := \frac{1}{k-m} \sum_{i=m}^{k-1} \eta_i,$$

then we have convergence rates in three distinct cases:

(i) If $f$ is non-convex and differentiable, then

$$\min_{0 \leq j < k} \|\nabla f(\theta^j)\| \leq \frac{\sqrt{\eta((r^0)^2 - (r^*)^2)}}{\sqrt{k \bar{\eta}_k}}.$$
Theorem 2

(ii) $f$ is convex and $L$-smooth with the global minimizer $\theta^*$. If $\max_{k_0 \leq j < k} \eta_j \leq 1/L$ for some $k_0 \geq 0$, then

$$f(\theta^k) - f(\theta^*) \leq \frac{||\theta^{k_0} - \theta^*||^2}{2(k - k_0)\bar{\eta}_{k-k_0}}.$$ 

(iii) $f$ is $\alpha$-strongly convex and $L$-smooth with the global minimizer $\theta^*$. If $\max_{k_0 \leq j < k} \eta_j \leq \frac{2}{\alpha + L}$ for some $k_0 \geq 0$, then

$$||\theta^k - \theta^*|| \leq e^{-\alpha(k-k_0)\bar{\eta}_{k-k_0}} ||\theta^{k_0} - \theta^*||.$$ 

Convergence rates are also obtained for the element-wise AEGD, with more subtle analysis.
Algorithm 1 AEGD. Good default setting for parameters are $c = 1$ and $\eta = 0.1$ for Deep Learning problems

**Require:** \( \{f_j(\theta)\}_{j=1}^{N} \), \( \eta \): the step size, \( \theta^0 \): initial guess of \( \theta \), and \( T \): the total number of iterations.

**Require:** \( c \): a parameter such that \( f(\theta) + c > 0 \) for all \( \theta \in \mathbb{R}^n \), \( r_i^0 = \sqrt{f(\theta^0)} + c \) for \( i \in [n] \): initial energy, \( r^0 = (r_1^0, r_2^0, \ldots, r_n^0) \)

1. **for** \( k = 0 \) to \( T - 1 \) **do**
2. \( \nabla f_{jk} = \nabla_\theta f_{jk}(\theta^k) \) (\( j_k \) is a random sample from \([N]\)).
3. \( \nabla g_{jk} = \nabla f_{jk} / (2\sqrt{f_{jk}(\theta^k)} + c) \) (get gradients of \( g_{jk}(\theta) \) at step \( k \))
4. \( r^{k+1} = r^k / (1 + 2\eta(\nabla g_{jk})^2) \) (update energy)
5. \( \theta^{k+1} = \theta^k - 2\eta r^{k+1} \odot \nabla g_{jk} \) (update parameters)
6. **return** \( \theta^T \)
Performance testing problems

Consider searching the minima $x^*$ of the following quadratic function

$$f(x_1, x_2, \ldots, x_{100}) = \sum_{i=1}^{50} x_{2i-1}^2 + \sum_{i=1}^{50} \frac{x_{2i}^2}{10^2},$$

and the 2D Rosenbrock function

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2.$$
GDM tends to overshoot and detour to the local minima. While adaptive optimization algorithms like Adam and AEGD go along a more direct path to the minima.

Figure: Trajectories of different algorithms on the 2D Rosenbrock function
K-means clustering

We consider $k$-means clustering problem for a set of data points $\{p_i\}_{i=1}^N$ in $\mathbb{R}^d$ with $K$ centroids $\{x_j\}_{j=1}^K$. Denote $x = [x_1, \cdots, x_K] \in \mathbb{R}^{Kd}$, we seek to minimize the quantization error:

$$\min_{x \in \mathbb{R}^{Kd}} \left\{ f(x) := \frac{1}{2N} \sum_{i=1}^N \min_{1 \leq j \leq K} \|x_j - p_i\|^2 \right\}. $$

**Figure:** The histogram of the quantization error of $k$-means on Iris trained by EM, GD and AEGD over 100 independent experiments.
Adam(W)$^3$ appear to perform better than the other algorithms early. But by epoch 150 when the learning rates are decayed, AEGD(W) begin to outperform other methods in test accuracy.

$^3$X(W): X with decoupled weight decay (I. Loshchilov & F. Hutter, 2019)
In the above two experiments, AEGD(W) even surpasses SGDM by 1% in test accuracy. However, the generalization capability of AEGD (or even other optimization algorithms) still needs further exploration.
Main contributions with AEGD

- It is shown to be unconditionally energy stable, irrespective of the step size.
- It feathers all the expected convergence rates for first order gradient-based methods, yet with only mild conditions on the initial step size.
- It allows a larger initial step size than (S)GD.
- It converges fast even for objective functions that have a large condition number numerically.
- It is robust with respect to initial data.
- It has visible advantages over the (S)GD algorithm with momentum.
- It shows potential for training neural networks with better generalization.
Optimal control on COVID-19

We introduce a dynamic control model for monitoring the virus propagation.

The goal is to minimize the number of infectious and death agents of the population or the error to the reported cases for the training of the model.
The SEIRD model [SIR model by Kermack-McKendrick, 1927] with time-varying epidemic parameters:

\[
\begin{align*}
\frac{dS}{dt} &= A - \beta SI / N - dS \\
\frac{dE}{dt} &= \beta SI / N - \epsilon E - dE \\
\frac{dI}{dt} &= \epsilon E - (\gamma + \mu)I - dI \\
\frac{dR}{dt} &= \gamma I - dR \\
\frac{dD}{dt} &= \mu I
\end{align*}
\]

where the whole population is divided into mutually exclusive groups: susceptible (\(S\)), exposed (\(E\)), infected (\(I\)), recovered (\(R\)) and deceased (\(D\)). \(A\) denotes the recruitment rate of the population due to birth and immigration, \(d\) is the natural death rate, \(\beta\) is the infection rate, \(\mu\) is the rate for virus-related death, \(\gamma\) is the rate for recovery and \(\epsilon\) is the rate at which the exposed individuals become infective.
To control the epidemic propagation we define the loss function as

\[ J = \sum_{i=1}^{n-1} L(U(t_i)) + g(U(T)) \]

where \( U = [S, E, I, R, D]^T \),

\[ L(U(t_i)) = \lambda_1 |I(t_i) - I_c(t_i)|^2 + \lambda_2 |D(t_i) - D_c(t_i)|^2, \quad 1 \leq i \leq n - 1 \]

\[ g(U(T)) = \lambda_1 |I(T) - I_d(T)|^2 + \lambda_2 |D(T) - D_d(T)|^2, \]

and \( 0 = t_0 < t_1 < t_2 < \ldots < t_n = T \), \( \lambda_i \) are user-defined normalization factors. The loss function is composed of two parts:

- \( L \) measures the error between the candidate solution \((I, D)\) to the SEIRD system and the reported data \((I_c, D_c)\) at intermediate times,
- \( g \) measures the error between the candidate solution \((I, D)\) and the desired data \((I_d, D_d)\) at the end time.
Data-driven optimal control

Denote $\theta = [\beta, \epsilon, \gamma, \mu]^T$, we aim to

- find optimal parameters $\theta(t)$ for $0 \leq t \leq t_{n-1}$ such that the solution to the SEIRD system fits the reported data at the grid points $\{t_i\}_{i=1}^{n-1}$ as close as possible, and
- find desired parameters $\theta(t)$ for $t_{n-1} \leq t \leq T$ that is able to control the epidemic spreading at time $T$ at desired values.

This allows us to reformulate the task as an discrete-time optimal control problem

$$
\min \quad J = \sum_{i=1}^{n-1} L(U(t_i)) + g(U(T)) \\
\text{s.t.} \quad \dot{U} = F(U, \theta) \quad t \in (0, T], \quad U(0) = U_0, \quad \theta(t) \in \Theta
$$
Data fitting results

Figure: (a) Infectious and death cases in USA (b) SEIRD parameters. The whole time is divided at 0, 30, 50, 70, 100, 130, 160, 190, 220 for training.

The most recent basic reproduction number $R_0 = \frac{\beta}{\mu + \gamma}$ is about 1.34.
Prediction

Use the parameters at the 220th day to make prediction for the next 7 days.

Figure: Prediction for the next 7 days by SEIRD model
Scheduled control

Figure: Scheduled control for the next 30 days by SEIRD model
Final remarks

- The self-contained selection PDE model is able to express selection, propagation, and control of network parameters.
- AEGD shows potential for training neural networks with better generalization.
- Data-driven SEIRD informed control appears to be a promising approach.

THANK YOU ALL