

# Thuật toán tự thích nghi cho lập trình tự động và ứng dụng trong học máy

Trần Ngọc Thăng<sup>1</sup>    Trịnh Ngọc Hải<sup>1</sup>

<sup>1</sup>Khoa Toán - Tin  
Đại học Bách Khoa Hà Nội, Hai Bà Trưng, Hà Nội, Việt Nam

## **Nhóm thuyết trình:** Nhóm 1

### **Các thành viên:**

|                      |           |
|----------------------|-----------|
| Nguyễn Ngọc Tuấn Anh | 202400029 |
| Trần Tuấn Anh        | 202416124 |
| Nguyễn Hữu Chính     | 202416143 |
| Trần Mạnh Cường      | 202416147 |
| Đỗ Hải Đăng          | 202400035 |
| Bùi Tuấn Đạt         | 202400096 |
| Nguyễn Hồng Đạt      | 202416153 |
| Phạm Đình Minh Đức   | 202400038 |
| Bùi Tiến Dũng        | 202416167 |
| Trần Huy Dương       | 202300025 |
| Nguyễn Khắc Duy      | 202400100 |
| Nguyễn Hoàng Gia     | 202400040 |

## 1 Giới thiệu chung

- Đặt vấn đề
- Phương pháp đề xuất

## 2 Kiến thức nền tảng

- Thiết lập bài toán
- Các tính chất quan trọng

## 3 Các kết quả chính

- Thuật toán Gradient Descent (GD)
- Thuật toán Gradient Descent Adaptive (GDA)
- Chứng minh sự hội tụ
- Biến thể stochastic (SGDA)

## 4 Kết quả thực nghiệm

- Thực nghiệm toán học
- Ứng dụng trong học máy

- **Vai trò của GD:** Là công cụ phổ biến giải quyết các bài toán quy hoạch từ lỗi đến không lỗi, với nhiều ứng dụng thực tế.
- **Thực trạng:** Các bài toán học máy thường có số chiều lớn và hàm mục tiêu **không lỗi**.
- **Hạn chế của một số phương pháp hiện tại:**
  - Line-search (Tìm kiếm theo tia): Chi phí tính toán quá lớn cho mỗi lần lặp, đặc biệt khi tính giá trị hàm phức tạp.
  - Sử dụng hằng số Lipschitz (biết trước): Dẫn đến hội tụ chậm do chỉ dùng một phần ước lượng không chính xác, hoặc thông tin này không có sẵn.
  - Độ dài bước giảm dần (Diminishing step-size): Gây ra tốc độ hội tụ chậm (ví dụ: phương pháp của Kiwiel cho quy hoạch tựa lỗi).

# Phương pháp đề xuất

- **Mục tiêu:** Đề xuất thuật toán độ dài bước tự thích ứng mới.
- **Phạm vi áp dụng:**
  - Hàm mục tiêu: Không lỗi và trơn.
  - Tập ràng buộc: Không bị chặn, đóng và lỗi.
- **Cơ chế hoạt động:** Giảm dần độ dài bước một cách ổn định cho đến khi thỏa mãn một điều kiện xác định, không cần biết trước hằng số Lipschitz.
- **Ứng dụng:** Hiệu quả cho các bài toán quy mô lớn như hồi quy logistic đa biến và mạng nơ-ron.

## 1 Giới thiệu chung

- Đặt vấn đề
- Phương pháp đề xuất

## 2 Kiến thức nền tảng

- Thiết lập bài toán
- Các tính chất quan trọng

## 3 Các kết quả chính

- Thuật toán Gradient Descent (GD)
- Thuật toán Gradient Descent Adaptive (GDA)
- Chứng minh sự hội tụ
- Biến thể stochastic (SGDA)

## 4 Kết quả thực nghiệm

- Thực nghiệm toán học
- Ứng dụng trong học máy

## Bài toán tối ưu hóa tổng quát (OP)

Chúng ta xét bài toán tối ưu hóa tổng quát ( $OP(f, C)$ ):

$$\min_{x \in C} f(x)$$

### Các giả thiết cơ bản:

- $C \subset \mathbb{R}^m$ : Là tập **lồi**, **đóng** và khác rỗng (có thể không bị chặn).
- $f : \mathbb{R}^m \rightarrow \mathbb{R}$ : Là hàm khả vi trên tập mở chứa  $C$ .
- $\nabla f$ : Có tính chất  **$L$ -Lipschitz liên tục** trên  $C$ , tức là tồn tại  $L > 0$  sao cho:

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \forall x, y \in C$$

- Giả sử tập nghiệm của ( $OP(f, C)$ ) là khác rỗng.

# Phép chiếu lên tập lồi

Thuật toán sử dụng phép chiếu  $P_C(x)$  để đảm bảo nghiệm luôn nằm trong vùng chấp nhận.

## Định nghĩa 1

$$P_C(x) := \operatorname{argmin}\{\|z - x\| : z \in C\}$$

## Mệnh đề 1 (Bauschke và Combettes 2011)

Với mọi  $x, y \in \mathbb{R}^m$  và  $z \in C$ :

- ①  $\|P_C(x) - P_C(y)\| \leq \|x - y\|$  (Tính không giãn).
- ②  $\langle z - P_C(x), x - P_C(x) \rangle \leq 0$  (Tính chất biến phân).



# Định nghĩa các dạng hàm lồi

Theo Mangasarian (1965) và các tài liệu cơ sở :

## Định nghĩa 2

Hàm  $f$  được gọi là: Với mọi  $x, y \in C$  và  $\lambda \in [0, 1]$ .

- **Lồi (Convex):**  $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$ .
- **Tựa lồi (Quasiconvex):**  $f(\lambda x + (1 - \lambda)y) \leq \max\{f(x); f(y)\}$ .
- **Giả lồi (Pseudoconvex):**  $\langle \nabla f(x), y - x \rangle \geq 0 \Rightarrow f(y) \geq f(x)$

Mối quan hệ bao hàm:

$$\text{Lồi} \implies \text{Giả lồi} \implies \text{Tựa lồi}$$

# Điều kiện của Hàm tựa lồi

Để chứng minh sự hội tụ của thuật toán trên hàm tựa lồi, ta cần tính chất sau:

## Mệnh đề 2 (Dennis và Schnabel 1983)

Hàm khả vi  $f$  là **tựa lồi** trên  $C$  khi và chỉ khi:

$$f(y) \leq f(x) \implies \langle \nabla f(x), y - x \rangle \leq 0$$

*Ý nghĩa:* Nếu giá trị hàm tại  $y$  nhỏ hơn tại  $x$ , thì hướng từ  $x$  đến  $y$  phải tạo góc tù với gradient tại  $x$ .

## Mệnh đề 3 (Dennis và Schnabel 1983)

Giả sử  $\nabla f$  là  $L$ -Lipschitz liên tục trên  $C$ . Với mọi  $x, y \in C$ , ta có:

$$|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leq \frac{L}{2} \|y - x\|^2$$

*Ý nghĩa:* Bất đẳng thức này chặn trên sai số giữa hàm số  $f(y)$  và xấp xỉ tuyến tính của nó tại  $x$ .

# Bổ đề chứng minh hội tụ

Để chứng minh sự hội tụ của dãy lặp  $\{x^k\}$ , ta sử dụng bổ đề sau:

## Bổ đề 1 (Xu 2002)

Cho các dãy số thực  $\{a_k\}, \{b_k\} \subset (0, \infty)$  thỏa mãn điều kiện:

$$a_{k+1} \leq a_k + b_k, \quad \forall k \geq 0$$

và tổng chuỗi hội tụ:  $\sum_{k=0}^{\infty} b_k < \infty$

Khi đó, tồn tại giới hạn hữu hạn:  $\lim_{k \rightarrow \infty} a_k = c \in \mathbb{R}$

*Vai trò:* Bổ đề này được dùng để xử lý sự nhiễu động trong quá trình lặp, đảm bảo thuật toán vẫn hội tụ dù có sai số nhỏ.

- 1 Giới thiệu chung
  - Đặt vấn đề
  - Phương pháp đề xuất
- 2 Kiến thức nền tảng
  - Thiết lập bài toán
  - Các tính chất quan trọng
- 3 Các kết quả chính
  - Thuật toán Gradient Descent (GD)
  - Thuật toán Gradient Descent Adaptive (GDA)
  - Chứng minh sự hội tụ
  - Biến thể stochastic (SGDA)
- 4 Kết quả thực nghiệm
  - Thực nghiệm toán học
  - Ứng dụng trong học máy

# Thuật toán Gradient Descent

## Thuật toán Gradient Descent (GD)

**Bước 0.** Chọn điểm khởi tạo  $x^0 \in C$ , tham số bước  $\lambda \in (0, 2/L)$ . Đặt  $k = 0$ .

**Bước 1.** Với  $x^k$  đã cho, tính

$$x^{k+1} = P_C(x^k - \lambda \nabla f(x^k)).$$

**Bước 2.**

- Nếu  $x^{k+1} = x^k$  thì **dừng**.
- Ngược lại, đặt  $k := k + 1$ , quay lại **Bước 1**.

# Thuật toán Gradient Descent (GD)

- $x^k$  : nghiệm xấp xỉ tại vòng lặp  $k$ .
- $-\nabla f(x^k)$  : hướng dốc nhất làm giảm hàm mục tiêu.
- $\lambda$  : bước nhảy (step-size), cố định trong khoảng  $(0, 2/L)$ .
- $P_C$  : phép chiếu lên tập ràng buộc  $C$ :
- Nếu  $C = \mathbb{R}^m$  thì  $P_C$  là ánh xạ đồng nhất  $\Rightarrow$  GD không ràng buộc.

**Ý tưởng chính:** đi theo hướng  $-\nabla f(x^k)$ , sau đó kéo nghiệm trở lại miền khả thi  $C$ .

# Thuật toán Gradient Descent Adaptive (GDA)

## Thuật toán GDA

**Bước 0.** Chọn  $x^0 \in C$ ,  $\lambda_0 > 0$ , hệ số  $\sigma, \kappa \in (0, 1)$ . Đặt  $k := 0$ .

**Bước 1.** Với  $x^k, \lambda_k$  đã cho, tính

$$x^{k+1} = P_C(x^k - \lambda_k \nabla f(x^k)).$$

Nếu  $f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle$  thì  $\lambda_{k+1} := \lambda_k$ , ngược lại  $\lambda_{k+1} := \kappa \lambda_k$ .

**Bước 2.**

- Nếu  $x^{k+1} = x^k$  thì **dừng**.
- Ngược lại, đặt  $k := k + 1$ , quay lại **Bước 1**.

- Ý tưởng: giảm dần cỡ bước một cách ổn định cho đến khi thỏa mãn điều kiện, không cần chi phí tính toán lớn cho hằng số Lipschitz hoặc tìm kiếm theo tia.



**Chú ý 1** Nếu thuật toán GDA dừng tại bước  $k$ , thì  $x^k$  là một điểm dừng của bài toán  $OP(f, C)$ . Thật vậy, vì  $x^{k+1} = P_C(x^k - \lambda_k \nabla f(x^k))$ , áp dụng Mệnh đề 1-(2), ta có:

$$\langle z - x^{k+1}, x^k - \lambda_k \nabla f(x^k) - x^{k+1} \rangle \leq 0 \quad \forall z \in C. \quad (1)$$

Nếu  $x^{k+1} = x^k$ , ta thu được:

$$\langle \nabla f(x^k), z - x^k \rangle \geq 0 \quad \forall z \in C, \quad (2)$$

điều này có nghĩa là  $x^k$  là điểm dừng của bài toán. Hơn nữa, nếu  $f$  là hàm tựa lồi, từ (2) suy ra  $f(z) \geq f(x^k)$  với mọi  $z \in C$ , hay  $x^k$  là nghiệm của  $OP(f, C)$ .

## Định lý

Giả sử dãy  $\{x^k\}$  được sinh bởi thuật toán GDA. Khi đó:

- Dãy  $\{f(x^k)\}$  hội tụ và mọi điểm giới hạn (nếu có) của  $\{x^k\}$  là 1 điểm dừng của bài toán.
- Nếu  $f$  **tựa lồi** trên  $C$  thì  $\{x^k\}$  hội tụ tới một điểm dừng của bài toán.
- Nếu  $f$  **giả lồi** trên  $C$  thì  $\{x^k\}$  hội tụ tới 1 nghiệm của bài toán.

# Chứng minh định lý

Dãy  $\{f(x^k)\}$  hội tụ và mọi điểm giới hạn (nếu có) của  $\{x^k\}$  là 1 điểm dừng của bài toán.

Từ mệnh đề 3 và (1), ta có:

$$f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle - \left( \frac{1 - \sigma}{\lambda_k} - \frac{L}{2} \right) \|x^{k+1} - x^k\|^2. \quad (3)$$

Ta chứng minh được rằng dãy  $\{\lambda_k\}$  có giới hạn khác 0, hay  $\lambda_k$  chỉ thay đổi trong một số hữu hạn bước và sẽ ổn định sau đó. Suy ra tồn tại  $k_1 \in \mathbb{N}$  sao cho với mọi  $k \geq k_1$ , ta có  $\lambda_k = \lambda_{k_1}$  và

$$f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle. \quad (4)$$

Nhận thấy rằng  $\langle \nabla f(x^k), x^k - x^{k+1} \rangle \geq 0$ , ta suy ra dãy  $\{f(x^k)\}$  là hội tụ và

$$\sum_{k=0}^{\infty} \langle \nabla f(x^k), x^k - x^{k+1} \rangle < \infty, \quad \sum_{k=0}^{\infty} \|x^{k+1} - x^k\|^2 < \infty. \quad (5)$$

# Chứng minh định lý

Từ (1), với mọi  $z \in C$ , ta có

$$\begin{aligned}\|x^{k+1} - z\|^2 &= \|x^k - z\|^2 - \|x^{k+1} - x^k\|^2 + 2\langle x^{k+1} - x^k, x^{k+1} - z \rangle \\ &\leq \|x^k - z\|^2 - \|x^{k+1} - x^k\|^2 + 2\lambda_k \langle \nabla f(x^k), z - x^{k+1} \rangle.\end{aligned}\quad (6)$$

Giả sử  $\bar{x}$  là một điểm tụ của dãy  $\{x^k\}$ . Tồn tại một dãy con  $\{x^{k_i}\} \subset \{x^k\}$  sao cho  $\lim_{i \rightarrow \infty} x^{k_i} = \bar{x}$ . Với bất đẳng thức trên cho  $k = k_i$  và lấy giới hạn khi  $i \rightarrow \infty$ . Chú ý rằng  $\|x^k - x^{k+1}\| \rightarrow 0$ , và  $\nabla f$  là hàm liên tục, ta nhận được

$$\langle \nabla f(\bar{x}), z - \bar{x} \rangle \geq 0 \quad \forall z \in C,$$

Do đó  $\bar{x}$  là điểm dừng.

# Chứng minh định lý

Nếu  $f$  **tựa lồi** trên  $C$  thì  $\{x^k\}$  hội tụ tới một điểm dừng của bài toán.

Đặt

$$U := \left\{ x \in C : f(x) \leq f(x^k) \quad \forall k \geq 0 \right\}.$$

Vì  $U$  chứa tập nghiệm của  $OP(f, C)$ , và do đó,  $U$  không rỗng. Lấy  $\hat{x} \in U$ .  
Vì  $f(x^k) \geq f(\hat{x})$  với mọi  $k \geq 0$  và  $f$  là hàm tựa lồi nên

$$\langle \nabla f(x^k), \hat{x} - x^k \rangle \leq 0 \quad \forall k \geq 0. \quad (7)$$

Kết hợp (6) và (7), ta có

$$\|x^{k+1} - \hat{x}\|^2 \leq \|x^k - \hat{x}\|^2 - \|x^{k+1} - x^k\|^2 + 2\lambda_k \langle \nabla f(x^k), x^k - x^{k+1} \rangle. \quad (8)$$

Áp dụng Bổ đề 1 với  $a_k = \|x^k - \hat{x}\|^2$ ,  $b_k = 2\lambda_k \langle \nabla f(x^k), x^k - x^{k+1} \rangle$ , ta suy ra rằng dãy  $\{\|x^k - \hat{x}\|\}$  hội tụ với mọi  $\hat{x} \in U$ .

Lại có dãy  $\{x^k\}$  bị chặn, tồn tại một dãy con  $\{x^{k_i}\} \subset \{x^k\}$  sao cho

$$\lim_{i \rightarrow \infty} x^{k_i} = \bar{x} \in C$$

Ta có dãy  $\{f(x^k)\}$  là dãy không tăng và hội tụ. Nên  $\lim_{k \rightarrow \infty} f(x^k) = f(\bar{x})$  và  $f(\bar{x}) \leq f(x^k)$  với mọi  $k \geq 0$ . Do đó  $\bar{x} \in U$  và dãy  $\{\|x^k - \bar{x}\|\}$  hội tụ. Từ đó suy ra,

$$\lim_{k \rightarrow \infty} \|x^k - \bar{x}\| = \lim_{i \rightarrow \infty} \|x^{k_i} - \bar{x}\| = 0.$$

Ta đã có mỗi điểm tụ của  $\{x^k\}$  là một điểm dừng của bài toán. Khi đó, toàn bộ dãy  $\{x^k\}$  hội tụ đến  $\bar{x}$  - một điểm dừng của bài toán.

Trong trường hợp  $f$  là hàm giả lồi, từ **(7)** áp dụng định nghĩa giả lồi cùng với **mệnh đề 2** điểm dừng này trở thành một nghiệm của  $OP(f, C)$ .  $\square$

# Thuật toán Gradient Descent Adaptive (GDA)

## Chú ý 2

Với mọi  $\lambda \in (0, 2/L)$  luôn tồn tại  $\sigma \in (0, 1)$  sao cho  $\lambda \leq 2(1 - \sigma)/L$ . Khi đó chọn  $\lambda_0 \in (0, 2(1 - \sigma)/L)$  thì  $\lambda_k = \lambda_0$  với mọi  $k$ .

- Như vậy GD là 1 trường hợp đặc biệt của GDA.
- Nếu  $L$  biết trước có thể chọn  $\lambda$  cố định thì theo định lý trên thuật toán vẫn hội tụ.

# Stochastic Gradient Descent Adaptive Algorithm-SGDA

## Thuật toán GD Thích nghi Ngẫu nhiên (SGDA)

**Bước 0.** Chọn  $x^0 \in C$ ,  $\lambda_0 \in (0, +\infty)$ ,  $\sigma, \kappa \in (0, 1)$ . Đặt  $k = 0$ .

**Bước 1.** Lấy mẫu  $\xi^k$  và tính  $x^{k+1}$  cùng  $\lambda_{k+1}$ :

$$x^{k+1} = P_C(x^k - \lambda_k \nabla f_{\xi^k}(x^k))$$

**Nếu**  $f_{\xi^k}(x^{k+1}) \leq f_{\xi^k}(x^k) - \sigma \langle \nabla f_{\xi^k}(x^k), x^k - x^{k+1} \rangle$   
**thì**  $\lambda_{k+1} := \lambda_k$ ; **ngược lại**  $\lambda_{k+1} := \kappa \lambda_k$ .

**Bước 2.**

- Nếu  $x^{k+1} = x^k$  thì **dừng**.
- Ngược lại, đặt  $k := k + 1$ , quay lại **Bước 1**.



- 1 Giới thiệu chung
  - Đặt vấn đề
  - Phương pháp đề xuất
- 2 Kiến thức nền tảng
  - Thiết lập bài toán
  - Các tính chất quan trọng
- 3 Các kết quả chính
  - Thuật toán Gradient Descent (GD)
  - Thuật toán Gradient Descent Adaptive (GDA)
  - Chứng minh sự hội tụ
  - Biến thể stochastic (SGDA)
- 4 Kết quả thực nghiệm
  - Thực nghiệm toán học
  - Ứng dụng trong học máy

## Ví dụ 1 (Ví dụ 5.2, Liu et. al., 2022)

Xét bài toán (P):

$$\min f(x) = \frac{x_1^2 + x_2^2 + 3}{1 + 2x_1 + 8x_2}$$

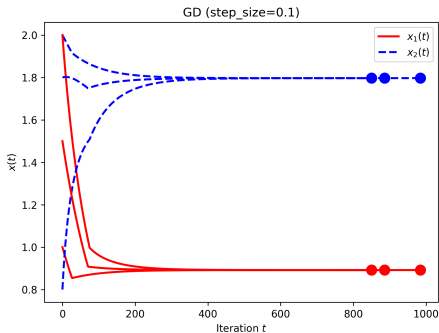
$$\text{v.đ.k } x \in C,$$

với  $C = \{ x = (x_1, x_2)^T \in \mathbb{R}^2 \mid g_1(x) = -x_1^2 - 2x_1x_2 \leq -4, x_1, x_2 \geq 0 \}$ .

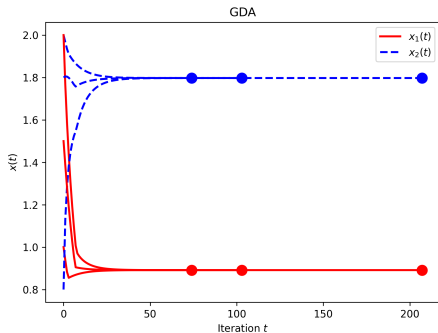
Hàm mục tiêu  $f$  là giả lồi trên tập lồi  $C$ .

| Thuật toán | $f(x^*)$ | Iter |
|------------|----------|------|
| GDA        | 0.40935  | 130  |
| GD         | 0.40935  | 910  |

# Thực nghiệm toán học



(a) GD



(b) GDA

Hình: Kết quả chạy thực nghiệm ví dụ 1

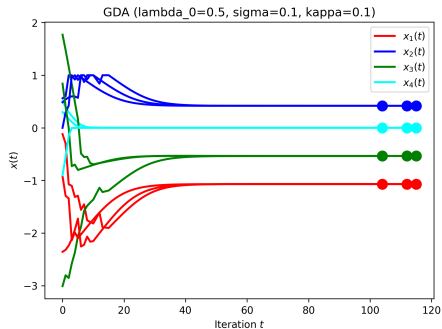
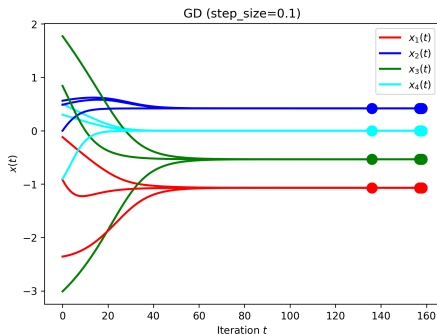
## Ví dụ 2 (Ví dụ 5.1, Liu et. al., 2022)

Xét bài toán tối ưu giả lồi không trơn sau đây:

$$\begin{aligned} \min \quad & f(x) = \frac{e^{|x_2-3|} - 30}{x_1^2 + x_3^2 + 2x_4^2 + 4}, \\ \text{v.đ.k} \quad & g_1(x) = (x_1 + x_3)^3 + 2x_4^2 \leq 10, \\ & g_2(x) = (x_2 - 1)^2 \leq 1, \\ & 2x_1 + 4x_2 + x_3 = -1, \end{aligned}$$

trong đó  $x = (x_1, x_2, x_3, x_4)^\top \in \mathbb{R}^4$ . Hàm mục tiêu  $f(x)$  là hàm giả lồi không trơn trên miền  $C$ .

# Thực nghiệm toán học



Hình: Kết quả chạy thực nghiệm ví dụ 2

| Thuật toán | $f(x^*)$ | Iter |
|------------|----------|------|
| GDA        | -3.0908  | 110  |
| GD         | -3.0908  | 150  |

## Ví dụ 3 (Ví dụ 4.5, Ferreira et. al., 2022)

Cho  $e := (1, \dots, n) \in \mathbb{R}^n$  là một vector,  $\alpha > 0$  và  $\beta > 0$  là các hằng số thỏa mãn  $2\alpha > 3\beta^{3/2}\sqrt{n}$ . Xét bài toán (P) với hàm liên quan:

$$f(x) := a^\top x + \alpha x^\top x + \frac{\beta}{\sqrt{1 + \beta x^\top e}} e^\top x,$$

với  $a \in \mathbb{R}_{++}^n$  và tập ràng buộc không lồi  $C := \{x \in \mathbb{R}_{++}^n : 1 \leq x_1 \cdots x_n\}$

Kết quả chạy thực nghiệm *ví dụ 3* với  $\beta = 0.741271, \alpha = 4\beta^{3/2}\sqrt{n+1}$ .

# Thực nghiệm toán học

| Results with Adaptive Step Size (gda_multiplier=5.0) |                          |           |       |  |                |           |       |
|--|--------------------------|-----------|-------|--|----------------|-----------|-------|
| n  | Algorithm GDA (proposed) |           |       |  | Algorithm GD   |           |       |
|  | f_opt                    | time (ms) | iters |  | f_opt          | time (ms) | iters |
| 10   | 80.080567                | 5.187273  | 18    |  | 80.080567      | 5.340099  | 19    |
| 20   | 219.590296               | 4.446507  | 17    |  | 219.590296     | 4.818201  | 19    |
| 50   | 852.496543               | 5.086899  | 17    |  | 852.496543     | 3.758669  | 19    |
| 100  | 2394.338032              | 2.675772  | 17    |  | 2394.338032    | 9.163141  | 20    |
| 200  | 6732.212169              | 8.964062  | 17    |  | 6732.212169    | 8.924246  | 20    |
| 500  | 26429.055560             | 11.014938 | 17    |  | 26429.055560   | 3.299236  | 20    |
| 1000   | 74531.265821             | 4.306316  | 17    |  | 74531.265821   | 8.611441  | 20    |
| 2000   | 210371.422765            | 6.401777  | 17    |  | 210371.422765  | 7.229328  | 20    |
| 3000   | 386136.150348            | 7.884979  | 17    |  | 386136.150348  | 7.285357  | 20    |
| 10000  | 2345979.128179           | 20.025969 | 17    |  | 2345979.128180 | 19.420147 | 20    |

Hình: GDA với  $\lambda_0 = 5/L$ .

# Thực nghiệm toán học

| Results with Adaptive Step Size (gda_multiplier=2.0) |                          |           |       |                |           |       |  |
|--|--------------------------|-----------|-------|----------------|-----------|-------|--|
| n  | Algorithm GDA (proposed) |           |       | Algorithm GD   |           |       |  |
|  | f_opt                    | time (ms) | iters | f_opt          | time (ms) | iters |  |
| 10   | 80.080567                | 5.170822  | 8     | 80.080567      | 10.428429 | 19    |  |
| 20   | 219.590296               | 3.693819  | 8     | 219.590296     | 5.551577  | 19    |  |
| 50   | 852.496543               | 5.488157  | 8     | 852.496543     | 12.067556 | 19    |  |
| 100  | 2394.338032              | 2.471209  | 8     | 2394.338032    | 9.481430  | 20    |  |
| 200  | 6732.212169              | 0.000000  | 8     | 6732.212169    | 4.953146  | 20    |  |
| 500  | 26429.055560             | 5.886316  | 9     | 26429.055560   | 4.676104  | 20    |  |
| 1000   | 74531.265821             | 2.899885  | 9     | 74531.265821   | 5.962849  | 20    |  |
| 2000   | 210371.422765            | 1.003504  | 9     | 210371.422765  | 0.000000  | 20    |  |
| 3000   | 386136.150348            | 12.046814 | 9     | 386136.150348  | 8.279085  | 20    |  |
| 10000  | 2345979.128179           | 8.660316  | 9     | 2345979.128180 | 19.539595 | 20    |  |

Hình: GDA với  $\lambda_0 = 2/L$ .



# Thực nghiệm toán học

| Results with Fixed Step Size (gda_multiplier=5.0, step_size=0.1) |                          |            |       |                |            |       |  |
|--|--------------------------|------------|-------|----------------|------------|-------|--|
| n  | Algorithm GDA (proposed) |            |       | Algorithm GD   |            |       |  |
|  | f_opt                    | time (ms)  | iters | f_opt          | time (ms)  | iters |  |
| 10   | 80.080567                | 15.069962  | 46    | 80.080567      | 0.000000   | 8     |  |
| 20   | 219.590296               | 17.896414  | 58    | 219.590296     | 2.617359   | 13    |  |
| 50   | 852.496543               | 21.991491  | 90    | 852.496543     | 5.728245   | 21    |  |
| 100  | 2394.338032              | 30.868053  | 111   | 2394.338032    | 8.029699   | 29    |  |
| 200  | 6732.212169              | 27.752638  | 100   | 6732.212169    | 11.726856  | 41    |  |
| 500  | 26429.055560             | 40.580988  | 130   | 26429.055560   | 20.916939  | 66    |  |
| 1000   | 74531.265821             | 44.952154  | 112   | 74531.265821   | 34.481049  | 92    |  |
| 2000   | 210371.422765            | 61.488390  | 138   | 210371.422765  | 57.462692  | 129   |  |
| 3000   | 386136.150348            | 50.976515  | 92    | 386136.150348  | 94.659328  | 161   |  |
| 10000  | 2345979.128179           | 137.495756 | 101   | 2345979.128179 | 426.794291 | 293   |  |

Hình: GDA với  $\lambda_0 = 0.5$ .

# Thực nghiệm toán học

| Results with Fixed Step Size (gda_multiplier=2.0, step_size=0.1) |                          |            |       |                |            |       |  |
|--|--------------------------|------------|-------|----------------|------------|-------|--|
| n  | Algorithm GDA (proposed) |            |       | Algorithm GD   |            |       |  |
|  | f_opt                    | time (ms)  | iters | f_opt          | time (ms)  | iters |  |
| 10   | 80.080567                | 8.711338   | 19    | 80.080567      | 0.000000   | 8     |  |
| 20   | 219.590296               | 8.127928   | 26    | 219.590296     | 3.469229   | 13    |  |
| 50   | 852.496543               | 11.977434  | 41    | 852.496543     | 6.461382   | 21    |  |
| 100  | 2394.338032              | 9.682655   | 55    | 2394.338032    | 11.731386  | 29    |  |
| 200  | 6732.212169              | 19.770622  | 78    | 6732.212169    | 10.179520  | 41    |  |
| 500  | 26429.055560             | 31.902075  | 109   | 26429.055560   | 22.556067  | 66    |  |
| 1000   | 74531.265821             | 32.226086  | 103   | 74531.265821   | 36.172628  | 92    |  |
| 2000   | 210371.422765            | 51.384926  | 114   | 210371.422765  | 65.045834  | 129   |  |
| 3000   | 386136.150348            | 84.401369  | 130   | 386136.150348  | 95.977306  | 161   |  |
| 10000  | 2345979.128179           | 172.491789 | 125   | 2345979.128179 | 400.845051 | 293   |  |

Hình: GDA với  $\lambda_0 = 0.2$ .

# Ứng dụng trong học máy

Phương pháp đề xuất được đánh giá qua 03 bài toán phổ biến trong Machine Learning để chứng minh hiệu quả tính toán và độ chính xác:

- ➊ **Lựa chọn đặc trưng có giám sát (Supervised feature selection):**
- ➋ **Hồi quy Logistic đa biến (Multi-variable logistic regression):**
- ➌ **Phân loại ảnh với Mạng nơ-ron (Neural networks for classification):**

Các bài toán trên được mô hình hóa dưới dạng các bài toán tối ưu.

# Lựa chọn đặc trưng có giám sát

- **Mô hình:** Tối thiểu hóa hàm phân thức giả lỗi trên tập lỗi

$$\text{minimize} \quad \frac{w^T Q w}{\rho^T w}$$

với điều kiện  $e^T w = 1; w \geq 0$ ,

- $w$ : Vector trọng số đặc trưng cần tìm (Feature score).
- $Q$ : Ma trận dư thừa thông tin.
- $\rho$ : Vector mức độ liên quan đến nhãn.
- $e$ : Vector đơn vị  $(1, \dots, 1)^T$ , ràng buộc tổng trọng số = 1.
- **Dữ liệu:** Tập dữ liệu Parkinsons (23 đặc trưng, 195 mẫu).
- **So sánh:** Thuật toán GDA đề xuất với Thuật toán mạng nơ-ron động (RNN).

# Lựa chọn đặc trưng có giám sát

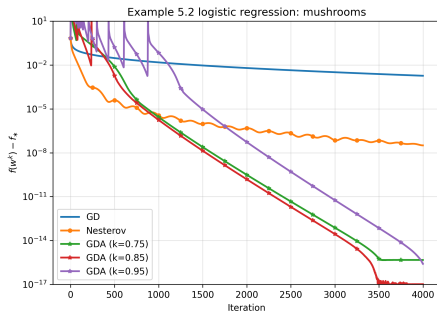
| Comparing Scipy vs GDA |                          |           |       |          |           |       |  |
|------------------------|--------------------------|-----------|-------|----------|-----------|-------|--|
| seed                   | Algorithm GDA (proposed) |           |       | Scipy    |           |       |  |
|                        | f_opt                    | time (ms) | iters | f_opt    | time (ms) | iters |  |
| 1                      | 0.152478                 | 0.018614  | 32    | 0.152478 | 0.006164  | 19    |  |
| 11                     | 0.153480                 | 0.096673  | 51    | 0.153480 | 0.030229  | 15    |  |
| 21                     | 0.153834                 | 0.015980  | 31    | 0.153834 | 0.006214  | 18    |  |
| 31                     | 0.153501                 | 0.029897  | 55    | 0.153501 | 0.005530  | 16    |  |
| 41                     | 0.154281                 | 0.026025  | 50    | 0.154281 | 0.004012  | 13    |  |
| 51                     | 0.154185                 | 0.031173  | 60    | 0.154185 | 0.005908  | 15    |  |
| 61                     | 0.154878                 | 0.034290  | 57    | 0.154878 | 0.006208  | 16    |  |
| 71                     | 0.153465                 | 0.021693  | 33    | 0.153465 | 0.005638  | 16    |  |
| 81                     | 0.153517                 | 0.031792  | 59    | 0.153517 | 0.003538  | 13    |  |
| 91                     | 0.153269                 | 0.030206  | 52    | 0.153269 | 0.005027  | 15    |  |

Hình: Time qua các vòng lặp.

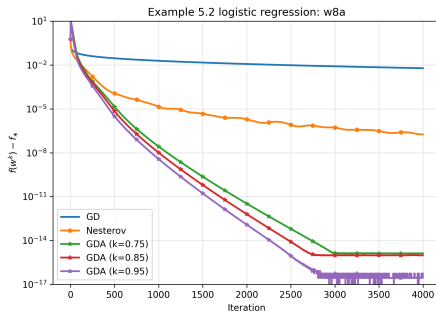
# Hồi quy Logistic đa biến

- **Bài toán:** Tối ưu hàm mất mát Cross-entropy có điều chuẩn  $L_2$  (Convex programming).
- **Dữ liệu:** Mushrooms và W8a.
- **So sánh:**
  - Gradient Descent (GD) truyền thống.
  - Phương pháp tăng tốc Nesterov.
- **Nhận xét:**
  - GDA cho giá trị hàm mục tiêu tốt hơn qua các vòng lặp so với GD và Nesterov trên cả 2 tập dữ liệu.
  - Cỡ bước tự thích nghi giảm dần theo thời gian giúp hội tụ ổn định.

# Hồi quy Logistic đa biến



(a) Mushrooms



(b) W8a

Hình: Kết quả chạy thực nghiệm Hồi quy Logistic đa biến.

# Phân loại ảnh với Mạng nơ-ron

## Thiết lập thực nghiệm

- **Mô hình:** Kiến trúc ResNet-18.
- **Dữ liệu:** CIFAR-10 (Dữ liệu ảnh).
- **Thuật toán:** Sử dụng biến thể ngẫu nhiên **SGDA** (do hàm mục tiêu không lồi/không giả lồi).

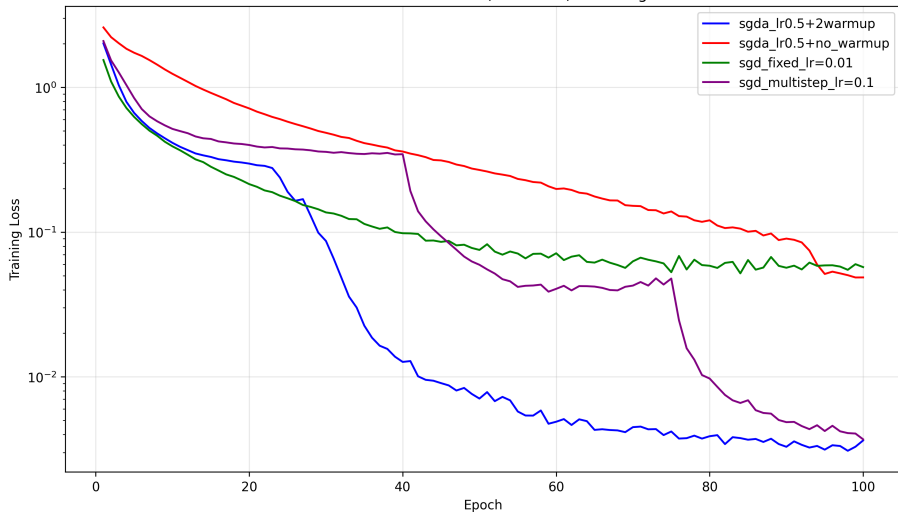
## Nhận xét khi so sánh với SGD

- **Test Accuracy:** SGDA đạt độ chính xác kiểm tra cao hơn qua các vòng lặp.
- **Training Loss:** SGDA giảm hàm mất mát huấn luyện tốt hơn so với SGD.
- *Lưu ý:* Dù không đảm bảo hội tụ lý thuyết mạnh như trường hợp lồi, SGDA vẫn hoạt động hiệu quả như một kỹ thuật heuristic.



# Phân loại ảnh với mạng nơ-ron

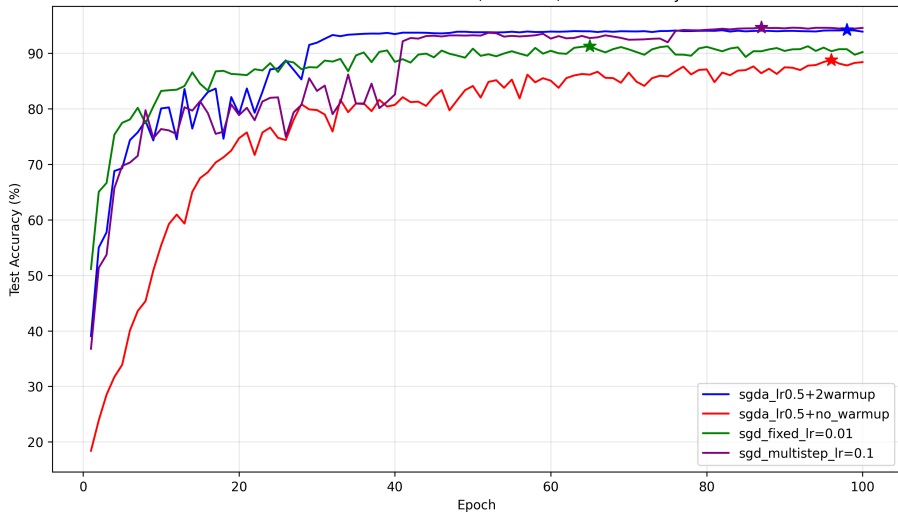
SGD vs SGDA on CIFAR-10 (ResNet18) - Training Loss



Hình: Training loss qua các vòng lặp.

# Phân loại ảnh với mạng nơ-ron

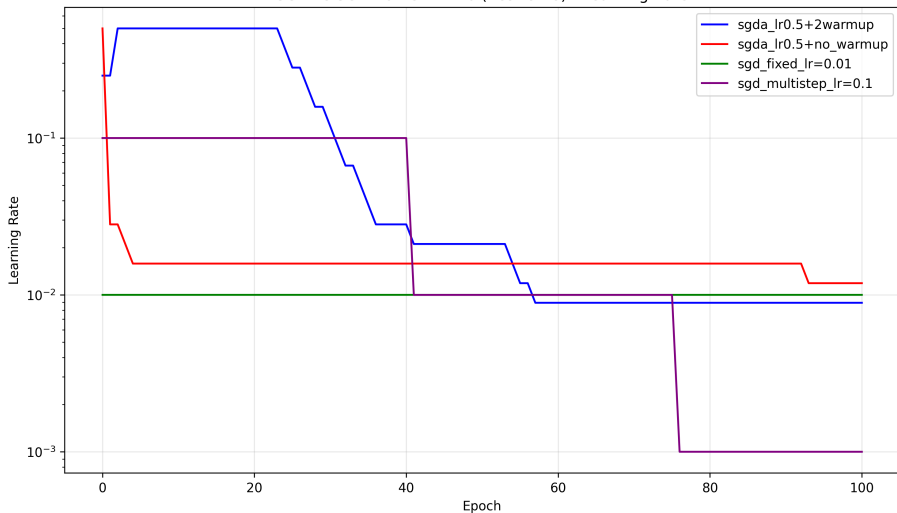
SGD vs SGDA on CIFAR-10 (ResNet18) - Test Accuracy



Hình: Test Accuracy qua các vòng lặp.

# Phân loại ảnh với mạng nơ-ron

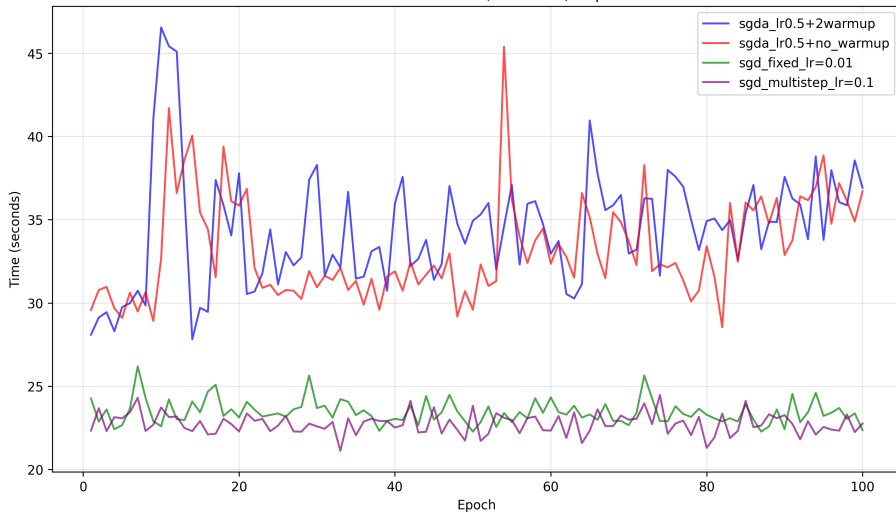
SGD vs SGDA on CIFAR-10 (ResNet18) - Learning Rate



Hình: Learning rate qua các vòng lặp.

# Phân loại ảnh với mạng nơ-ron

SGD vs SGDA on CIFAR-10 (ResNet18) - Epoch Time



Hình: Time qua các vòng lặp.

**Nhóm trưởng:** Bùi Tuấn Đạt

- **Email:** Dat.BT2400096@sis.hust.edu.vn
- **Github:** <https://github.com/tuasananh/NMPPTU-Team1-SAAQP>

*Xin chân thành cảm ơn mọi người đã lắng nghe bài thuyết trình của nhóm!*