BỘ GIÁO DỤC VÀ ĐÀO TẠO

VIỆN HÀN LÂM KHOA HỌC VÀ CÔNG NGHỆ VIỆT NAM

HỌC VIỆN KHOA HỌC VÀ CÔNG NGHỆ

VĂN BÁ CÔNG

GIẢI BÀI TOÁN CAUCHY CHO MỘT SỐ PHƯƠNG TRÌNH ĐẠO HÀM RIÊNG BẰNG MẠNG NEURAL NHÂN TẠO

LUẬN VĂN THẠC SĨ TOÁN ỨNG DỤNG

HÀ NỘI - 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO

VIỆN HÀN LÂM KHOA HỌC VÀ CÔNG NGHỆ VIỆT NAM

HỌC VIỆN KHOA HỌC VÀ CÔNG NGHỆ

VĂN BÁ CÔNG

GIẢI BÀI TOÁN CAUCHY CHO MỘT SỐ PHƯƠNG TRÌNH ĐẠO HÀM RIÊNG BẰNG MẠNG NEURAL NHÂN TẠO

LUẬN VĂN THẠC SĨ TOÁN ỨNG DỤNG

Mã số: 8460112

NGƯỜI HƯỚNG DẪN KHOA HỌC: GS.TSKH. ĐINH NHO HÀO

HÀ NỘI - 2023

LỜI CAM ĐOAN

Tôi cam đoan rằng các kết quả được trình bày trong luận văn này là công trình nghiên cứu tổng quan do riêng tôi thực hiện và hoàn thành dưới sự hướng dẫn của GS.TSKH Đinh Nho Hào. Các khái niệm và số liệu trong luận văn được tổng hợp từ các tài liệu khoa học đáng tin cậy và được trích dẫn rõ ràng nguồn gốc. Đóng góp của tôi là tổng hợp tài liệu và trình bày thêm kết quả số minh họa cho phương pháp tôi đề xuất. Tôi chịu trách nhiệm hoàn toàn với lời cam đoan này.

Hà Nội, tháng 10 năm 2023

Học viên

Văn Bá Công

LỜI CẢM ƠN

Luận văn thạc sĩ này được hoàn thành tại Học viện Khoa học và Công nghệ - Viện Hàn lâm Khoa học và Công nghệ Việt Nam. Sau một thời gian học tập và nghiên cứu, dưới sự chỉ bảo tận tình của thầy giáo hướng dẫn, đến nay luận văn của tôi đã hoàn thành.

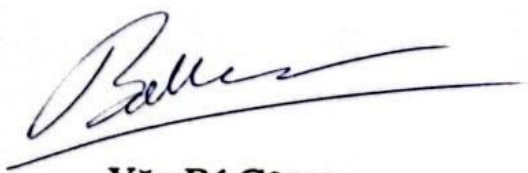
Lời đầu tiên, tôi xin bày tỏ lòng biết ơn sâu sắc đến thầy giáo hướng dẫn GS.TSKH Đinh Nho Hào, người đã tận tình hướng dẫn, tạo điều kiện và giúp đỡ tôi trong suốt quá trình học tập và nghiên cứu. Đồng thời tôi cũng xin bày tỏ lòng biết ơn chân thành đến PGS.TS Phạm Qúy Mười người đã truyền đạt những kiến thức nền tảng đầu tiên, hỗ trợ và động viên tôi vượt qua mọi khó khăn.

Tôi cũng xin gửi lời cảm ơn chân thành nhất đến các thầy cô tại Viện Toán học, Học viện Khoa học và Công nghệ đã giúp đỡ, tạo điều kiện thuận lợi để tôi có được môi trường học tập và nghiên cứu trong suốt thời gian qua. Cảm ơn Quỹ Đổi mới sáng tạo Vingroup (VinIF) đã tài trợ học bổng thạc sĩ trong hai năm để tôi yên tâm học tập và nghiên cứu tại Hà Nội (Mã số: VINIF.2021.ThS.VTH.03; VINIF.2022.ThS.VTH.02).

Cuối cùng tôi xin chân thành cảm ơn gia đình, bạn bè và các anh chị trong nhóm seminar đã giúp đỡ, động viên, chia sẻ kiến thức, tạo điều kiện cho tôi hoàn thành tốt luận văn thạc sĩ này.

Tôi xin chân thành cảm ơn!

Học viên



Văn Bá Công

MỤC LỤC

KÍ HIỆU DÙNG TRONG LUẬN VĂN

$\mathbb{R}$	Tập hợp các số thực;
$\mathbb{R}^n$	Không gian Euclid n chiều;
Ω	Tập mở trong $\mathbb{R}^n$;
Γ	Tập con của $\partial\Omega$;
$C^2(\Omega)$	Không gian các hàm có đạo hàm liên tục cấp hai trên Ω;
X, Y	Không gian định chuẩn (hoặc Hilbert) X, Y;
$R(A)$	Miền giá trị của toán tử A;
$\|A\|$	Chuẩn của toán tử A;
A^*	Toán tử liên hợp của toán tử A;
$\mathcal{L}$	Toán tử tuyến tính;
I	Toán tử đơn vị;
R_α	Toán tử chỉnh hóa;
α	Tham số chỉnh hóa;
ANN	Mạng neuron nhân tạo (Artificial Neural Network);
w	Trọng số (weights);
b	Độ lệch (bias);
σ	Hàm kích hoạt (hàm sigmoid);
$\left\|u\right\|$	Chuẩn cho trường hợp ANN.

DANH SÁCH BẢNG

DANH SÁCH HÌNH VẼ

MỞ ĐẦU

Trong những năm gần đây, mạng neuron nhân tạo (Artificial Neural Network = ANN) đã trở thành một công cụ phổ biến được sử dụng để giải các bài toán có số chiều lớn hoặc phi tuyến. Tính hiệu quả và tính tổng quát của ANN đã thúc đẩy việc áp dụng kỹ thuật này vào các bài toán cho phương trình vi phân, phương trình đạo hàm riêng khác nhau. Trong số đó, bài toán Cauchy là một lớp bài toán đặc thù của phương trình đạo hàm riêng. Ứng dụng và vai trò của bài toán Cauchy rất quan trọng trong nhiều lĩnh vực như khoa học, công nghệ, địa vật lý, y học, vật lý plasma, thủy động học,... Cụ thể, trong việc nghiên cứu trường trọng lực (trường điện và từ trường), chúng ta cần xác định thế vị của trường bên ngoài một vật thể dựa trên giá trị thế vị trong một phần của miền đó. Trong việc xác định hoạt động não điện hoặc hoạt động tim, người ta sử dụng các đo đạc điện thế và cường độ dòng điện tại hộp sọ hoặc ngực, sau đó giải quyết một bài toán Cauchy cho phương trình elliptic tương ứng ([1], [2], [3], [4]).

Trong nghiên cứu này, cho $\Omega \subset \mathbb{R}^d$ là một miền có biên liên tục $\partial\Omega$, với d là số chiều không gian, và $\Gamma \subset \partial\Omega$. Chúng tôi xét bài toán Cauchy cho phương trình elliptic

$$\begin{cases} \mathcal{L}u(\mathbf{x}) = 0, & \mathbf{x} \text{ trong } \Omega, \\ u(\mathbf{x}) = f, & \mathbf{x} \text{ trên } \Gamma, \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = g, & \mathbf{x} \text{ trên } \Gamma, \end{cases} \tag{0.1}$$

và bài toán Cauchy cho phương trình parabolic

$$\begin{cases} \frac{\partial u(\mathbf{x},t)}{\partial t} + \mathcal{L}u(\mathbf{x},t) = 0, & (\mathbf{x},t) \text{ trong } \Omega \times \mathcal{T}, \\ u(\mathbf{x},t) = f, & (\mathbf{x},t) \text{ trên } \Gamma \times \mathcal{T}, \\ \frac{\partial u(\mathbf{x},t)}{\partial \mathbf{n}} = g, & (\mathbf{x},t) \text{ trên } \Gamma \times \mathcal{T}, \\ u(\mathbf{x},0) = h, & \mathbf{x} \text{ trong } \Omega. \end{cases} \tag{0.2}$$

Trong đó, $\mathbf{n}$ là vector pháp tuyến đơn vị hướng ra ngoài, $\mathcal{L}$ là một toán tử elliptic

đều, và $\mathcal{T} = [0, T]$ là khoảng thời gian. Mục tiêu của bài toán ngược Cauchy là tìm một nghiệm u trong miền Ω và trên phần còn lại của biên $\partial\Omega/\Gamma$ sao cho nó thỏa mãn phương trình trạng thái của bài toán Cauchy, cùng với các điều kiện ban đầu và điều kiện biên thích hợp h, f, g.

Bài toán Cauchy cho phương trình elliptic là một bài toán đặc biệt trong tập hợp các bài toán Cauchy, nó là *bài toán đặt không chỉnh theo nghĩa Hadamard.* Một bài toán được gọi là *đặt chỉnh* nếu thỏa mãn các tính chất sau:

1. Tồn tại một nghiệm của bài toán (tính tồn tại).

2. Tồn tại không quá một nghiệm (tính duy nhất).

3. Nghiệm phụ thuộc liên tục vào dữ kiện của bài toán (tính ổn định).

Nếu ít nhất một trong ba điều kiện này không thoả mãn, chúng ta nói rằng *bài toán đặt không chỉnh.* Đối với bài toán Cauchy của phương trình elliptic, không phải với tất cả các dữ kiện Cauchy bài toán đều có nghiệm, nghiệm chỉ tồn tại khi và chỉ khi có độ trơn và tính tương thích giữa các dữ kiện Cauchy. Hadamard đã đưa ra các điều kiện về tính giải được của bài toán Cauchy cho phương trình Laplace vào năm 1902. Ông cũng đã đưa ra một ví dụ nổi tiếng về sự phụ thuộc không liên tục vào các dữ kiện Cauchy của bài toán Cauchy cho phương trình Laplace. Cụ thể, Hadamard đã xem xét bài toán Cauchy cho phương trình Laplace 2 chiều.

$$u_{xx} + u_{yy} = 0, \quad a < x < b, \quad 0 < y < Y, \tag{0.3}$$

$$u(x, 0) = \varphi(x), \quad u_y(x, 0) = \psi(x), \quad a < x < b, \quad 0 < y < Y. \tag{0.4}$$

Vào năm 1902, Hadamard [5] đã chỉ ra rằng, nếu φ và ψ là các hàm liên tục, thì bài toán sẽ có nghiệm khi và chỉ khi hàm số

$$\frac{1}{2}\varphi(x) - \frac{1}{2\pi}\int_a^b \psi(\xi)\log|x - \xi|d\xi$$

giải tích trên khoảng $a < x < b$. Điều này cho thấy rằng, ở đây, điều kiện Cauchy không nhất thiết phải là các hàm giải tích.

Sau đó, vào năm 1917, Hadamard [6] đã đưa ra ví dụ: Với $\varphi = \varphi_n(x) = 0$ và $\psi = \psi_n(x) = ne^{-\sqrt{n}}\sin(nx)$, bài toán (0.3)- (0.4) có nghiệm là

$$u_n(x, y) = e^{-\sqrt{n}}\sin(nx)\sinh(ny).$$

Chúng ta có thể thấy rằng $|\varphi_n|, |\psi_n|$ giới nội, nhưng $|u_n(x, y)|$ không giới nội khi $x > 0$ và $n \to \infty$. Do đó, nghiệm của bài toán (0.3)- (0.4) không phụ thuộc một cách liên tục vào điều kiện Cauchy.

Đối với bài toán Cauchy cho phương trình parabolic, tính đặt không chỉnh được Ginsberg [7] đưa ra bằng ví dụ xét phương trình nhiệt $u_{xx} = u_t$, cho $0 < x < 1, 0 < t \leq T$, với các điều kiện Cauchy: $u(0,t) = \exp(ikt)$ và $u_x(0,t) = 0$. Ta có:

$$u(x,t) = \cosh(x\sqrt{ik})\exp(ikt).$$

Chúng ta thấy rằng, nếu $k \to \infty$, thì $|u(x,t)| \sim \exp(x\sqrt{k/2}) \to \infty$, nhưng $|u(0,t)| = 1$ bị chặn. Nghĩa là, dù giá trị k có thay đổi đến đâu, giá trị tại điểm $x = 0$ vẫn giữ nguyên là 1 và không phụ thuộc vào k. Điều này cho thấy rằng nghiệm của bài toán không phụ thuộc liên tục vào điều kiện Cauchy tại $x = 0$.

Trong nghiên cứu của mình, Hadamard đã chỉ ra các ví dụ chứng minh rằng không phải với dữ kiện Cauchy nào bài toán cũng có nghiệm và nghiệm của bài toán Cauchy cho phương trình Laplace (trường hợp đặc biệt nhất của phương trình elliptic) nói chung không phụ thuộc liên tục vào dữ kiện Cauchy. Do đó, Hadamard cũng cho rằng các bài toán đặt không chỉnh không có ý nghĩa vật lý vì nghiệm không phụ thuộc liên tục vào điều kiện Cauchy của bài toán. Tuy nhiên, trong thực tế nhiều bài toán trong các lĩnh vực như khoa học và công nghệ, như đã được đề cập ở trên, dẫn đến bài toán Cauchy và bài toán đặt không chỉnh nói chung. Vì những lý do này, từ đầu thập kỷ 50 của thế kỷ trước, nhiều nghiên cứu đã tập trung vào bài toán đặt không chỉnh. Các nhà toán học như A. N. Tikhonov, M. M. Lavrent'ev, F. John, C. Pucci, V. K. Ivanov đã tiên phong trong lĩnh vực này ([8], [9], [10], [11], [12], [13]). Vào năm 1963, Tikhonov [14] đã đưa ra phương pháp hiệu chỉnh nổi tiếng, từ đó lý thuyết về bài toán đặt không chỉnh đã được phát triển mạnh mẽ và áp dụng rộng rãi trong hầu hết các bài toán thực tế. Điều này đã làm cho bài toán đặt không chỉnh và bài toán ngược trở thành các lĩnh vực độc lập quan trọng trong toán học và tính toán. Bài toán Cauchy cũng không nằm ngoài xu hướng này.

Trong luận văn này, chúng tôi tập trung vào giải số bài toán Cauchy (0.1) và (0.2) một cách ổn định dựa trên ANN. Đầu tiên, chúng tôi giải thích sự khó khăn khi giải số các bài toán này. Giả sử chúng ta có một phương trình $Au = b$ cần giải, trong đó A là một toán tử tuyến tính hoặc phi tuyến từ không gian hàm X sang không gian hàm Y, và b là một điều kiện đã cho thuộc không gian Y. Khi bài toán là bài toán đặt không chỉnh, không phải với mọi dữ kiện b bài toán đều có nghiệm và thậm chí khi có nghiệm, nghiệm đó không phụ thuộc liên tục vào dữ kiện b hoặc (tồn tại theo một nghĩa nào đó). Tính không ổn định này của bài toán khiến việc giải số trở nên khó khăn. Một sai số nhỏ trong điều kiện của bài toán có thể dẫn đến một sai số lớn trong nghiệm. Do đó, việc giải bài toán số trở nên khó khăn vì không thể loại

bỏ hoàn toàn sai số trong điều kiện. Bên cạnh sai số thường gặp trong quá trình đo đạc, chúng ta còn phải đối mặt với sai số không thể bỏ qua do quá trình rời rạc hóa và sự làm tròn của máy tính. Mục tiêu của lý thuyết bài toán đặt không chỉnh là cung cấp các phương pháp số hiệu quả để giải quyết các bài toán này một cách ổn định. Để đạt được mục tiêu đó, trước tiên cần nghiên cứu tính *ổn định có điều kiện* của bài toán, đã được nghiên cứu trong các tài liệu ([12], [13], [15], [16]). Năm 1943, Tikhonov đã đưa ra nhận xét ban đầu trong bài viết [8], sau này được gọi là *ổn định theo nghĩa Tikhonov* ([9], [15], [17]). Tiếp sau đó, M. M. Lavrent'ev ([10], [11]) đã đưa ra các *đánh giá ổn định dạng Holder* cho bài toán Cauchy của phương trình Laplace.

Phương pháp số để giải bài toán Cauchy cho phương trình elliptic và parabolic đang nhận được sự quan tâm ngày càng lớn do nhu cầu thực tiễn. Việc tìm kiếm các thuật toán ổn định và hiệu quả cho bài toán này là một thách thức khó khăn nhưng rất cần thiết. Trong thời gian gần đây, phương pháp sử dụng mạng neuron nhân tạo giải bài toán ngược, phương trình đạo hàm riêng đã thu hút được nhiều sự quan tâm của nhiều nhà khoa học trên thế giới và đã có nhiều công trình nghiên cứu nổi bật như: [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30],... Mặc dù lý thuyết toán học cho phương pháp này chưa được phát triển một cách chi tiết, nhưng sự hiệu quả và tính ứng dụng rộng rãi của nó, đặc biệt là trong các bài toán có số chiều lớn, bài toán phi tuyến hoặc có tính kỳ dị, đã khuyến khích nhiều nhà nghiên cứu quốc tế áp dụng kỹ thuật này.

Công trình đầu tiên [31] giới thiệu việc sử dụng mạng neuron là của George E. Hinton và đồng nghiệp vào năm 1986. Tác giả đã giới thiệu một kiến trúc mạng neuron nhân tạo mới, được gọi là "mạng neuron tích chập" (convolutional neuron network - CNN). Bài báo này tập trung vào việc sử dụng mạng neuron tích chập để giải quyết bài toán phân loại ảnh. Đặc biệt, họ áp dụng thuật toán lan truyền ngược (backpropagation) để tự động học các trọng số của mạng neuron dựa trên dữ liệu huấn luyện. Mặc dù nghiên cứu không trực tiếp sử dụng mạng neuron để giải phương trình đạo hàm riêng, nhưng lại đánh dấu sự xuất hiện ban đầu của các mô hình mạng neuron và thuật toán backpropagation, đóng góp quan trọng cho sự phát triển mạnh mẽ của deep learning và ứng dụng rộng rãi của mạng neuron trong các lĩnh vực khác nhau. Năm 1998, [20] Lagaris, Likas và Fotiadis đã ứng dụng mạng neuron nhân tạo để giải các phương trình vi phân thường. Ý tưởng này được mở rộng cho phương trình vi phân bậc cao [21], [22] và phương trình đạo hàm riêng [23]. Các kết quả nghiên cứu đã tạo tạo ra một cơ sở lý thuyết cho việc sử dụng mạng neuron giải các bài toán phương

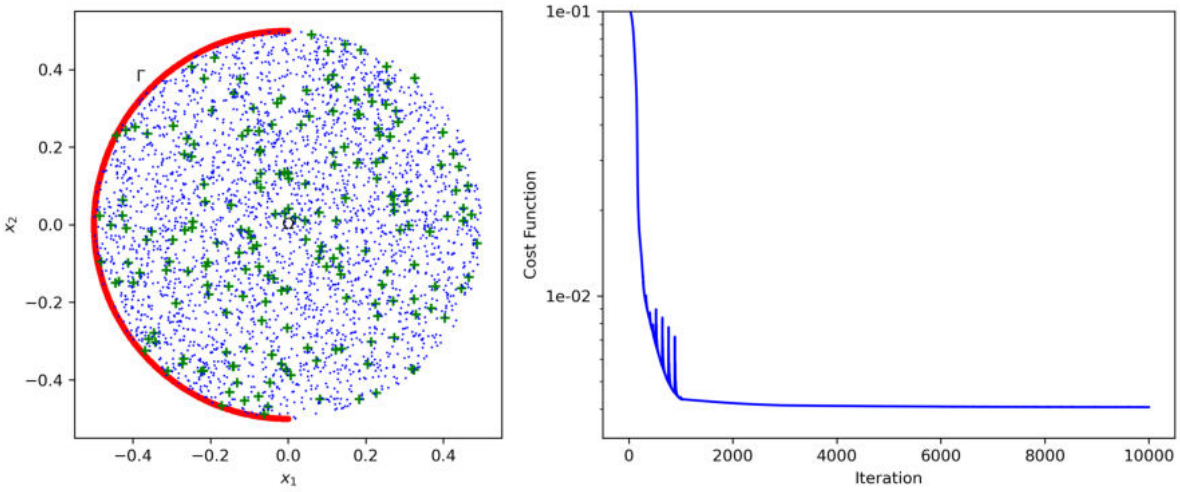
trình đạo hàm riêng và có ảnh hưởng sâu sắc đến các nghiên cứu sau này.

Năm 2018, [24] tác giả Sirignano và Spiliopoulos giới thiệu thuật toán DGM (Deep Galerkin Method), một thuật toán tiên tiến kết hợp giữa deep learning và phương pháp Galerkin truyền thống để tạo ra cách tiếp cận mới giải các bài toán PDEs. Phương pháp này đã có một số tính năng ưu việt hơn so với các phương pháp truyền thống. Năm 2019, [18] Maziar Raissi và cộng sự đã trình bày việc sử dụng mạng neuron học sâu, kết hợp với kiến thức vật lý, để giải các bài toán thuận và ngược của phương trình đạo hàm riêng phi tuyến. Phương pháp được chứng minh là có hiệu quả đối với nhiều bài toán khác nhau, bao gồm phương trình vi phân đạo hàm bậc phân số [25], bài toán thuận và ngược ngẫu nhiên [26].

Năm 2022, [30] Yixin Li và Xianliang Hu đã sử dụng mạng neuron nhân tạo để xấp xỉ bài toán Cauchy cho phương trình đạo hàm riêng tuyến tính. Tác giả khẳng định phương pháp cho kết quả ổn định nhưng khi chúng tôi chạy thử nghiệm trên máy tính bằng các code do chính tác giả viết thì kết quả không đúng như mô tả. Cụ thể, xét bài toán Cauchy sau

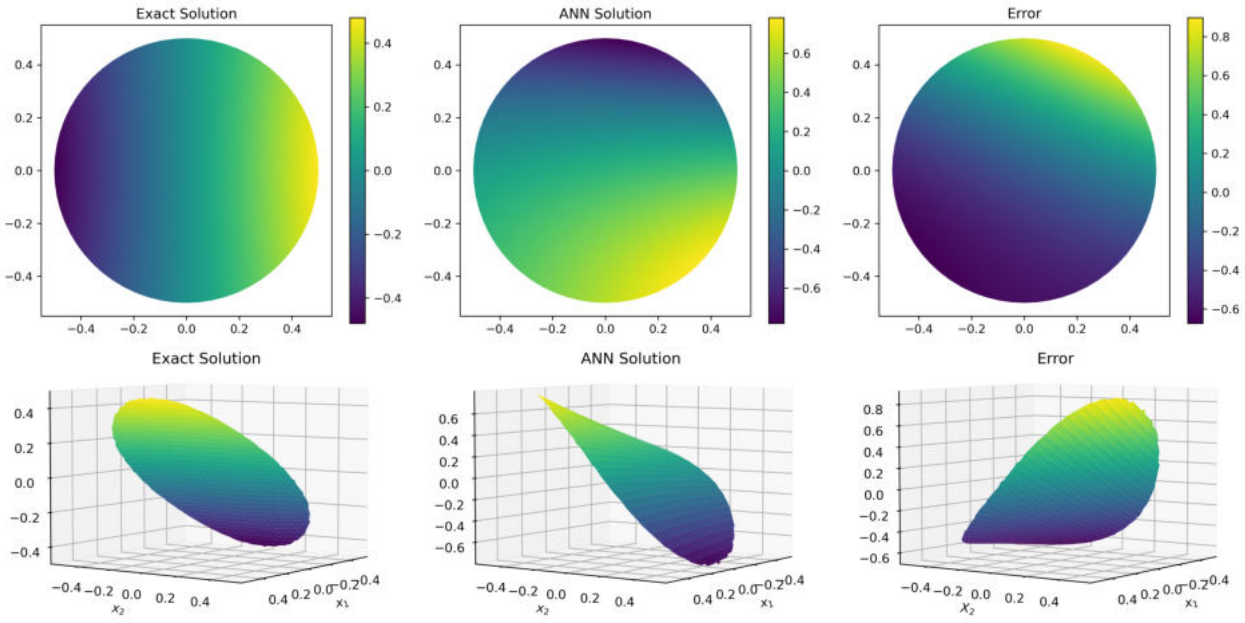
$$\begin{cases} \frac{\partial u(\mathbf{x};t)}{\partial t} + \Delta u(\mathbf{x};t) = 0, & \mathbf{x}, t \text{ in } \Omega, \mathcal{T} \\ u(\mathbf{x};t) = \sin(x_1)\cos(x_2)e^{2t}, & \mathbf{x}, t \text{ on } \Gamma, \mathcal{T} \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = \left[\cos(x_1)\cos(x_2)e^{2t}, -\sin(x_1)\sin(x_2)e^{2t}\right] * \mathbf{n}, & \mathbf{x}, t \text{ on } \Gamma, \mathcal{T} \\ u(\mathbf{x};0) = \sin(x_1)\cos(x_2), & \mathbf{x} \text{ in } \Omega \end{cases}$$

trong đó, Ω là một miền có biên liên tục $\partial\Omega$, Γ là 1/2 biên đường tròn và $\mathcal{T} = \left[0, \frac{\pi}{2}\right]$.



Hình 1: Bên trái là miền dữ liệu Ω, biên Γ(màu đỏ) được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước dữ liệu đào tạo $N_r = 10000$ điểm trên miền Ω và $N_b = 2500$ điểm trên biên Γ. Bên phải là biểu đồ hội tụ của thuật toán ADAM.

Theo biểu đồ sự hội tụ, chúng ta có thể thấy rằng khoảng 1000 lần lặp đầu tiên giá trị của hàm mất mát giảm nhanh nhưng không ổn định, sau đó giá trị của hàm có xu hướng theo một đường thẳng.



Hình 2: Nghiệm chính xác, nghiệm số ANN và sai số với $t = \frac{\pi}{5}$ trong trường hợp nhiễu 1%.

Dựa vào hình minh họa, ta thấy sai số giữa nghiệm chính xác và nghiệm số ANN là lớn. Điều này cho thấy thuật toán được áp dụng cho bài toán chưa đủ hiệu quả và không ổn định. Theo chúng tôi, nguyên nhân không có được sự ổn định của lời giải số là do tác giả đã không chỉnh hóa bài toán. Với suy luận như vậy chúng tôi đã đi đến ý tưởng viết lại bài toán Cauchy dưới dạng bài toán biến phân kết hợp với chỉnh hóa Tikhonov, sau đó dùng mạng neuron nhân tạo để giải bài toán đã chỉnh hóa này. Từ những lý do trên nên tôi chọn đề tài: **"Giải bài toán Cauchy cho một số phương trình đạo hàm riêng bằng mạng neural nhân tạo"**, với mong muốn sử dụng mạng neuron nhân tạo kết hợp với chỉnh hóa Tikhonov để giải quyết bài toán này.

Nội dung của luận văn được trình bày trong 3 chương. Ngoài ra, luận văn còn có phần mở đầu, kết luận, tài liệu tham khảo và phụ lục.

Chương 1: Cơ sở lý thuyết. Trong chương này, chúng tôi sẽ trình bày những kiến thức cơ bản về định nghĩa không gian hàm, bài toán Cauchy cho phương trình elliptic và parabolic, bài toán đặc chỉnh, lý thuyết về chỉnh hóa Tikhonov, và trình bày tổng quan về mạng neuron.

Chương 2: Giải bài toán Cauchy cho phương trình đạo hàm riêng bằng chỉnh hóa Tikhonov với mạng neuron nhân tạo. Trong chương này, chúng tôi

trình bày một phương pháp giải bài toán Cauchy bằng cách kết hợp phương pháp chỉnh hóa Tikhonov với mạng neuron nhân tạo để giải bài toán bài toán Cauchy đặt không chỉnh cho phương trình elliptic và parabolic.

Chương 3: Kết quả mô phỏng. Trong chương này, chúng tôi sẽ thử nghiệm mạng neuron nhân tạo (ANN) kết hợp với phương pháp chỉnh hóa Tikhonov thông qua các ví dụ cụ thể trong không gian hai chiều (2D), ba chiều (3D) cho hai trường hợp tuyến tính và phi tuyến. Những ví dụ này sẽ giúp chúng ta hiểu rõ hơn về việc kết hợp ANN với chỉnh hóa Tikhonov cũng như tính hiệu quả và tiềm năng của phương trong việc giải quyết các bài toán Cauchy cho phương trình đạo hàm riêng.

Chương 1
CƠ SỞ LÝ THUYẾT

Trong chương này, chúng tôi nhắc lại định nghĩa một số không gian hàm cơ bản, bài toán Cauchy cho phương trình elliptic và parabolic, bài toán đặt chỉnh, lý thuyết chỉnh hóa Tikhonov, và trình bày tổng quan về mạng neuron. Những kết quả này là cần thiết và phục vụ cho việc nghiên cứu ứng dụng ANN kết hợp phương pháp chỉnh hóa Tikhonov để giải bài toán Cauchy trong chương 2 và chương 3. Hầu hết các kết quả đều không chứng minh mà tham khảo ở các tài liệu [9], [16], [32], [33], [34], [35].

1.1. Định nghĩa các không gian hàm

Nội dung phần này được trích dẫn dựa trên tài liệu tham khảo [32] trang 702.

(i) $C(\Omega) = \{u : \Omega \to \mathbb{R} \mid u \text{ liên tục}\}$.

(ii) $C(\bar{\Omega}) = \{u \in C(\Omega) \mid u \text{ liên tục đều}\}$.

(iii) $C^k(\Omega) = \{u : \Omega \to \mathbb{R} \mid u \text{ là khả vi liên tục } k \text{ lần}\}$.

(iv) $L^p(\Omega) = \{u : \Omega \to \mathbb{R} \mid u \text{ là đo được Lebesgue, } \|u\|_{L^p(\Omega)} < \infty\}$, trong đó

$$\|u\|_{L^p(\Omega)} = \left(\int_\Omega |u|^p dx\right)^{1/p} \quad (1 \le p < \infty).$$

(v) $H^1(\Omega) = \{u \in L^2(\Omega) : \nabla u \in L^2(\Omega)\}$, trong đó ∇u là gradient của u. Chuẩn $\|u\|_{H^1(\Omega)}$ được định nghĩa như sau:

$$\|u\|_{H^1(\Omega)} = \left(\int_\Omega |u|^2 \, dx + \int_\Omega |\nabla u|^2 \, dx\right)^{1/2}$$

(vi) $H^2(\Omega) = \{u \in L^2(\Omega) : D^\alpha u \in L^2(\Omega), |\alpha| \le 2\}$.

1.2. Bài toán Cauchy cho phương trình loại elliptic

Nội dung phần này được trích dẫn dựa trên tài liệu tham khảo [36], [37].

Giả sử Ω là một miền (có thể không giới nội) trong không gian $\mathbb{R}^n, n \ge 2$, với biên đủ

trơn. Xét phương trình elliptic trong miền Ω

$$-\sum_{i,j=1}^{n}\frac{\partial}{\partial x_i}\left(a_{ij}(x)\frac{\partial u}{\partial x_j}\right)+\sum_{i=1}^{n}\frac{\partial}{\partial x_i}\left(b_i(x)u\right)+\sum_{i=1}^{n}c_i(x)\frac{\partial}{\partial x_i}(u)+d(x)u=f \tag{1.1}$$

với các hệ số a_{ij}, b_i, c_i, d trong $L_p(\Omega)$ nào đó. Hơn thế nữa a_{ij} thoả mãn điều kiện elliptic đều

$$\sum_{i,j=1}^{n}a_{ij}\xi_i\xi_j\geq a_0\sum_{i=1}^{n}\xi_i^2 \quad \text{với } a_0>0, \quad \xi=(\xi_1,\ldots,\xi_n)\in\mathbb{R}^n.$$

Giả sử Γ_0 là một phần của biên $\partial\Omega$. Trên Γ_0 cho hai điều kiện Cauchy

$$u|_{\Gamma_0}=\varphi \tag{1.2}$$

$$\left.\frac{\partial u}{\partial N}\right|_{\Gamma_0}=\psi, \tag{1.3}$$

ở đây, $\frac{\partial u}{\partial N}(x)=\sum_{i,j=1}^{n}a_{ij}u_{x_i}\cos(\nu,x_i)-\sum_{i=1}^{n}c_iu\cos(\nu,x_i)$, còn (ν,x_i) là góc giữa pháp tuyến ngoài đối với $\partial\Omega$ và trục x_i.

Bài toán tìm u thoả mãn (1.1) - (1.3) được gọi là bài toán Cauchy cho phương trình (1.1). Đây là một trong những bài toán Cauchy đơn giản nhất cho phương trình elliptic. Ngoài bài toán này, ta còn có các bài toán dạng Cauchy khác, ví dụ như ngoài điều kiện Cauchy (1.2), (1.3) tại một số phần của biên ta có thêm một điều kiện biên. Sự tồn tại và tính ổn định của bài toán có thể tham khảo ở các tài liệu sau [12], [37], [39]. Tính đặt không chỉnh của bài toán Cauchy cho phương trình loại elliptic có thể tham khảo ví dụ của Hadamard [5], [6] ở phần mở đầu của luận văn.

1.3. Bài toán Cauchy cho phương trình loại parabolic

Nội dung phần này được trích dẫn dựa trên tài liệu tham khảo [36], [37].

Giả sử Ω là một miền (có thể không giới nội) trong không gian $\mathbb{R}^n, n\geq 2$, với biên đủ trơn. Xét phương trình parabolic trong miền $\Omega\times(0,T)$.

$$u_t-\sum_{i,j=1}^{n}\frac{\partial}{\partial x_i}\left(a_{ij}(x,t)u_{x_j}+a_i(x,t)u\right)+\sum_{i=1}^{n}b_i(x,t)u_{x_i}+a(x,t)u=f(x,t)+\frac{\partial f_i(x,t)}{\partial x_i}, \tag{1.4}$$

với các hệ số a_{ij}, a_i, b_i, a trong $L_p(\Omega)$ nào đó. Giả sử Γ_0 là một phần của biên $\partial\Omega$, kí hiệu $S_T=\Gamma_0\times(0,T)$, bài toán cho điều kiện Cauchy sau

$$u|_{S_T}=\varphi \tag{1.5}$$

$$\left.\frac{\partial u}{\partial N}\right|_{S_T}=\psi \tag{1.6}$$

$$u|_{t=0,x\in\Omega}=\phi, \tag{1.7}$$

ở đây, $\frac{\partial u}{\partial N}(x) = \sum_{i,j=1}^{n} a_{ij} u_{x_j} \cos(n, x_i)$, còn (n, x_i) là góc giữa pháp tuyến ngoài đối với $\partial\Omega \times (0, T)$ và trục x_i.

Bài toán tìm u thoả mãn (1.4) - (1.7) được gọi là bài toán Cauchy cho phương trình (1.4). Đây là một trong những bài toán Cauchy đơn giản nhất cho phương trình parabolic. Ngoài bài toán này, ta còn có các bài toán dạng Cauchy khác, ví dụ như ngoài điều kiện (1.5), (1.6), (1.7) tại một số phần của biên ta có thêm một điều kiện biên. Sự tồn tại và tính ổn định của bài toán có thể tham khảo ở các tài liệu sau [37], [38], [40]. Tính đặt không chỉnh của bài toán Cauchy cho phương trình loại Parabolic có thể tham khảo ví dụ của Ginsberg [7] ở phần mở đầu của luận văn hoặc tham khảo trong luận án tiến sĩ khoa học [38] của GS. TSKH Đinh Nho Hào.

1.4. Bài toán đặt không chỉnh và lý thuyết chỉnh hóa

Nội dung kiến thức phần này được trích dẫn từ tài liệu [33].

Định nghĩa 1.1. (Tính đặt chỉnh và đặt không chỉnh theo nghĩa Hadamard [33]): Giả sử U, V là các không gian định chuẩn và một ánh xạ $A : U \longrightarrow V$ (tuyến tính hoặc phi tuyến). Bài toán $Au = b$ gọi là đặt chỉnh, nếu thỏa các tính chất sau

1. Tính tồn tại (existence): Với mọi $b \in V$ tồn tại $u \in U$ sao cho $Au = b$,

2. Tính duy nhất (uniqueness): Với mọi $b \in V$ có không quá một $u \in U$ sao cho $Au = b$,

3. Tính ổn định (stability): Nghiệm u phụ thuộc liên tục vào b, nghĩa là với mọi dãy $\{u_n\} \subset U$ và $Au_n \longrightarrow b$ thì $u_n \longrightarrow u (n \longrightarrow \infty)$.

Nếu bài toán không thỏa ít nhất một trong ba tính chất trên thì bài toán đó được gọi là đặt không chỉnh (ill-posed).

Sự chỉnh hóa, nghĩa là ta xét sự xấp xỉ giữa nghiệm chính xác (ứng với dữ liệu chính xác) và nghiệm chỉnh hóa (ứng với dữ liệu nhiễu). Một sự chỉnh hóa được gọi là tốt nếu sai số xấp xỉ càng nhỏ.

Định lý 1.1. *Cho X, Y là hai không gian định chuẩn và U là tập mở trong X. Nếu $A : U \longrightarrow Y$ compact và X vô hạn chiều thì A^{-1} không liên tục; nghĩa là, phương trình $Af = g$ đặt không chỉnh.*

Định nghĩa 1.2. Một phương pháp chỉnh hóa là một họ các toán tử tuyến tính, bị chặn $R_\alpha : Y \longrightarrow X, \alpha > 0$ sao cho

$$\lim_{\alpha \to 0} R_\alpha A x = x, \forall x \in X.$$

Định nghĩa 1.3. Cho toán tử tuyến tính, bị chặn $A : X \longrightarrow Y$ và $y \in Y$. Khi đó

$$J_\alpha(x) = \|Ax - y\|^2 + \alpha\|x\|^2, x \in X,$$

được gọi là phiếm hàm Tikhonov.

Định lý 1.2. *Cho X, Y là hai không gian Hilbert. $A : X \longrightarrow Y$ là toán tử tuyến tính compact, bị chặn và $\alpha > 0$. Khi đó phiếm hàm Tikhonov J_α có một cực tiểu là $x^\alpha \in X$. Cực tiểu x^α này là nghiệm duy nhất của phương trình*

$$\alpha x^\alpha + A^*Ax^\alpha = A^*y.$$

Tốc độ hội tụ của phương pháp chỉnh hóa Tikhonov trong trường hợp dữ liệu bị nhiễu được đưa ra trong các định lí sau:

Định lý 1.3. *Nếu $x^* = A^*z \in A^*(Y)$ với $\|z\| \leq E$ thì khi chọn $\alpha(\delta) = \frac{c\delta}{E}(c > 0)$, ta có:*

$$\left\|x_\alpha^\delta - x^*\right\| \leq \frac{1}{2}\left(\sqrt{c} + \frac{1}{\sqrt{c}}\right)\sqrt{\delta E}.$$

Định lý 1.4. *Nếu $x^* = A^*Az$ với $\|z\| \leq E$, thì với cách chọn:*

$$\alpha(\delta) = c\left(\frac{\delta}{E}\right)^{\frac{2}{3}}, c > 0,$$

ta có:

$$\left\|x_\alpha^\delta - x^*\right\| \leq \left(c + \frac{1}{2\sqrt{c}}\right)E^{\frac{1}{3}}\delta^{\frac{2}{3}}.$$

Bậc hội tụ của chỉnh hoá Tikhonov:

Mệnh đề 1.1. *Cho $A : X \to Y$ là toán tử tuyến tính, compact, đơn ánh sao cho $R(A)$ là vô hạn chiều. Xét $x \in X$, nếu tồn tại một hàm liên tục:*

$$\alpha : [0, +\infty) \to [0, +\infty)$$

với $\alpha(0) = 0$ sao cho:

$$\lim_{\delta \to 0}\left\|x_{\alpha(\delta)}^\delta - x\right\|\delta^{-\frac{2}{3}} = 0$$

với mọi $y^\delta \in Y, \left\|y^\delta - Ax\right\| \leq \delta$, thì $x = 0$.

1.5. Chọn tham số chỉnh hóa

Trong phần này, chúng tôi trình bày hai phương pháp để chọn tham số chỉnh hóa, bao gồm phương pháp hậu nghiệm (posterior regularization) và phương pháp L-curve (xem [9], [16], [41]). Cả hai phương pháp này thường được sử dụng để giải quyết vấn đề chọn tham số chỉnh hóa trong phương trình $Au = b$.

1.5.1. Phương pháp hậu nghiệm

Phương pháp hậu nghiệm (posterior regularization [9], [16]) là một phương pháp được sử dụng hiệu quả trong việc giải quyết các bài toán ước lượng tham số khi có sẵn thông tin tiên nghiệm (prior information). Trong phương pháp này, ta xem xét bài toán ước lượng tham số dưới dạng một bài toán tối ưu, trong đó mục tiêu là tìm một giá trị tham số phù hợp để giảm thiểu sai số giữa dữ liệu quan sát và dữ liệu dự đoán. Cho A là một toán tử tuyến tính ánh xạ từ không gian tham số u sang không gian quan sát b, u_α^δ là nghiệm chỉnh hóa được ước lượng, b^δ là dữ liệu quan sát được, ta xem xét bài toán ước lượng tham số dưới dạng:

$$\left\| Au_\alpha^\delta - b^\delta \right\| = \tau\delta,$$

trong đó, δ độ nhiễu; τ là một hằng số, xác định mức độ sai số được chấp nhận giữa dữ liệu quan sát và dữ liệu dự đoán và thường $\tau > 1$. Để tìm giá trị tham số chỉnh hóa α, ta có thể sử dụng thuật toán chia đôi (bisection algorithm) như sau:

Thuật toán 1 : Giải phương trình chỉnh hóa hậu nghiệm bằng phương pháp chia đôi

procedure

Bước 1: Xác định khoảng giá trị ban đầu cho α, ta có thể chọn khoảng $[\alpha_{\min}, \alpha_{\max}]$. Đây là khoảng chứa các giá trị của α gần với giá trị chính xác.

Bước 2: Thực hiện thuật toán tìm kiếm nhị phân để xác định giá trị của α.

Bước 2.1: Đặt $\alpha = (\alpha_{\min} + \alpha_{\max})$ / 2 là giá trị ở giữa của khoảng ban đầu.

Bước 2.2: Tính $\|Au_\alpha^\delta - b^\delta\|$ bằng cách sử dụng giá trị α hiện tại và giá trị đã biết của b.

Bước 2.3: So sánh giá trị tính toán được với giá trị mục tiêu $\tau\delta$.

if $\|Au_\alpha^\delta - b^\delta\| < \tau\delta$ **then**

    Tăng giá trị của α. Đặt $\alpha_{\min} = \alpha$ và quay lại **Bước 2.1**.

else if $\|Au_\alpha^\delta - b^\delta\| > \tau\delta$ **then**

    Giảm giá trị của α. Đặt $\alpha_{\max} = \alpha$ và quay lại **Bước 2.1**.

else

    $\|Au_\alpha^\delta - b^\delta\| = \tau\delta$. Ta đã tìm thấy giá trị α cần tìm. Kết thúc thuật toán.

end if

Bước 3: Lặp lại **Bước 2** cho đến khi tìm được giá trị α cần tìm hoặc đạt được độ chính xác yêu cầu.

end procedure

Trong phương pháp này, mức độ nhiễu (noise level) trong dữ liệu là một yếu tố quan trọng. Phương pháp hậu nghiệm sử dụng thông tin về mức độ nhiễu để xác định một giá trị thích hợp cho α. Phương pháp này thường cho kết quả tốt hơn khi mức độ nhiễu được xác định chính xác và đáng tin cậy.

1.5.2. Phương pháp L-curve

Phương pháp L-curve [41] là một phương pháp đồ thị được sử dụng để xác định giá trị tối ưu của tham số chỉnh hóa α. Cách tiếp cận này không phụ thuộc vào mức độ nhiễu (noise level) trong dữ liệu. Ý tưởng của L-curve là vẽ đường cong trong $\mathbb{R}^2$ $(\log \|Au-b\|, \log \|u\|)$, trong đó $\|Au-b\|$ là sai số dữ liệu và $\|u\|$ là sai số nghiệm. Điểm góc trên đường cong L-curve thường tương ứng với sự cân bằng tốt giữa sai số dữ liệu và sai số nghiệm của mô hình, giá trị α tương ứng với điểm này được chọn là giá trị tối ưu. Để tìm hệ số chỉnh hóa α, ta có thể thực hiện các bước sau để xây dựng đường cong L-curve:

Thuật toán 2 : L-curve

1: **procedure**
2: Xây dựng một tập các giá trị α.
3: Khởi tạo một tập để lưu trữ giá trị $\log \|Au - b\|$ và $\log \|u\|$.
4: **for** mỗi giá trị α trong tập đã xác định **do**
5: Giải PT chỉnh hóa Tikhonov để tính toán giá trị $\|Au - b\|$ và $\|u\|$ tương ứng.
6: Thêm giá trị $\log \|Au - b\|$ và $\log \|u\|$ vào tập đã khởi tạo.
7: **end for**
8: Vẽ đường cong L-curve trong $\mathbb{R}^2$ $(\log \|Au - b\|, \log \|u\|)$ sử dụng các điểm từ tập đã xác định.
9: Xác định điểm góc của đường cong L-curve (điểm có độ cong lớn nhất).
10: Chọn giá trị α tương ứng với điểm góc là giá trị chỉnh hóa tối ưu.
11: **end procedure**

Thuật toán bắt đầu bằng việc xây dựng một tập giá trị α. Sau đó, với mỗi giá trị α trong tập đã xác định, thuật toán giải phương trình chỉnh hóa Tikhonov để tính toán các giá trị $\|Au-b\|$ và $\|u\|$. Các giá trị logarithmic tương ứng được thêm vào các tập tương ứng. Sau khi thu thập được các giá trị logarithmic, thuật toán vẽ đường cong L-curve trong $\mathbb{R}^2$ $(\log \|Au - b\|, \log \|u\|)$. Điểm góc của đường cong, có độ cong lớn nhất, được xác định và giá trị α tương ứng với điểm này được chọn là giá trị chỉnh hóa tối ưu.

1.6. Tổng quan về mạng neuron nhân tạo

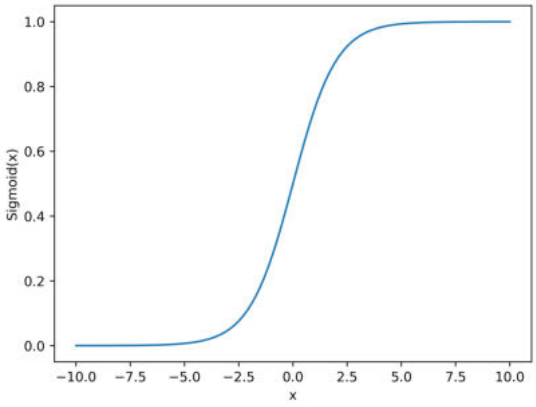
Mạng thần kinh nhân tạo (Artificial Neural Networks - ANN) là một mô hình lập trình được tạo ra dựa trên cấu trúc của mạng thần kinh trong hệ thống não của con người. Nhờ sự kết hợp với kỹ thuật học sâu (Deep Learning - DL), ANN đã phát triển thành một công cụ mạnh mẽ trong việc giải quyết các vấn đề phức tạp như nhận dạng hình ảnh, giọng nói, xử lý ngôn ngữ tự nhiên và cả các vấn đề liên quan đến lĩnh vực vật lý. Trong phần này, chúng tôi sẽ trình bày tổng quan và chi tiết về ANN, dựa trên nội dung tài liệu tham khảo [35].

1.6.1. Sigmoid Neurons

Chúng ta sẽ xây dựng mạng neuron dựa trên hàm sigmoid, được định nghĩa như sau:

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{1.8}$$

trên khoảng $-10 \leq x \leq 10$.



Hình 1.1: Đồ thị hàm Sigmoid.

Để thuận lợi cho việc tính toán, ta cần hiểu hàm sigmoid dưới khái niệm vector hóa. Giả sử có một vectơ $z \in \mathbb{R}^m$ và hàm $\sigma : \mathbb{R}^m \to \mathbb{R}^m$, ta định nghĩa σ là một phép toán áp dụng hàm sigmoid cho từng thành phần của vectơ z độc lập, tức là giá trị của $\sigma(z)$ tại thành phần thứ i là $\sigma(z_i)$. Điều này được biểu diễn như sau:

$$(\sigma(z))_i = \sigma(z_i) = \frac{1}{1 + e^{-z_i}}.$$

Giả sử rằng các số thực tạo ra bởi các neuron trên một lớp được tổng hợp thành một vector a, ta có thể biểu diễn vector đầu ra từ lớp tiếp theo bằng công thức sau:

$$\sigma(Wa + b) \tag{1.9}$$

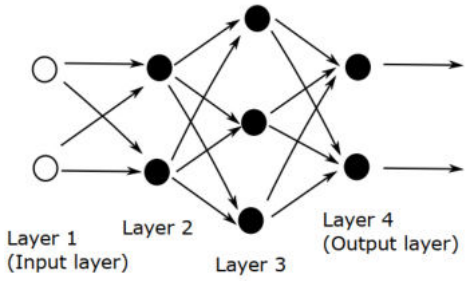
trong đó, ma trận W chứa các trọng số (weights) và vector b chứa các giá trị độ lệch (bias). Số cột của ma trận W tương ứng với số neuron trên lớp trước, và số hàng của W tương ứng với số neuron trên lớp hiện tại. Số thành phần trong vector b cũng tương ứng với số neuron trên lớp hiện tại. Để nhấn mạnh vai trò của neuron thứ i trong công thức trên (1.9), chúng ta có thể biểu diễn thành phần thứ i của nó theo công thức sau:

$$\sigma\left(\sum_j w_{ij}a_j + b_i\right).$$

Ở đây, tổng được tính qua tất cả các thành phần trong vector a.

1.6.2. Mạng neuron nhân tạo

Trong phần này, sẽ giới thiệu một tập ký hiệu đầy đủ, cho phép xác định một mạng neuron tổng quát. Trước khi đi vào chi tiết, chúng ta xét một ví dụ về mạng neuron nhân tạo bốn lớp cụ thể như hình 1.2



Hình 1.2: Mạng neuron nhân tạo bốn lớp.

Dữ liệu biểu diễn dưới dạng $x \in \mathbb{R}^2$. Do đó, các trọng số và độ lệch cho lớp thứ 2 được biểu diễn bằng một ma trận $W^{[2]} \in \mathbb{R}^{2\times 2}$ và một vector $b^{[2]} \in \mathbb{R}^2$ tương ứng. Đầu ra lớp thứ 2 được tính bằng công thức:

$$\sigma\left(W^{[2]}x + b^{[2]}\right) \in \mathbb{R}^2$$

Lớp thứ 3 có ba neuron, mỗi neuron nhận đầu vào từ không gian $\mathbb{R}^2$. Do đó, các trọng số và độ lệch cho lớp thứ 3 được biểu diễn bằng một ma trận $W^{[3]} \in \mathbb{R}^{3\times 2}$ và một vector $b^{[3]} \in \mathbb{R}^3$ tương ứng. Đầu ra từ lớp thứ 3 được tính bằng công thức:

$$\sigma\left(W^{[3]}\sigma\left(W^{[2]}x + b^{[2]}\right) + b^{[3]}\right) \in \mathbb{R}^3$$

Lớp thứ 4, cũng là lớp đầu ra, có hai neuron. Mỗi neuron nhận đầu vào từ không gian $\mathbb{R}^3$. Do đó, các trọng số và độ lệch cho lớp này được biểu diễn bằng một ma trận

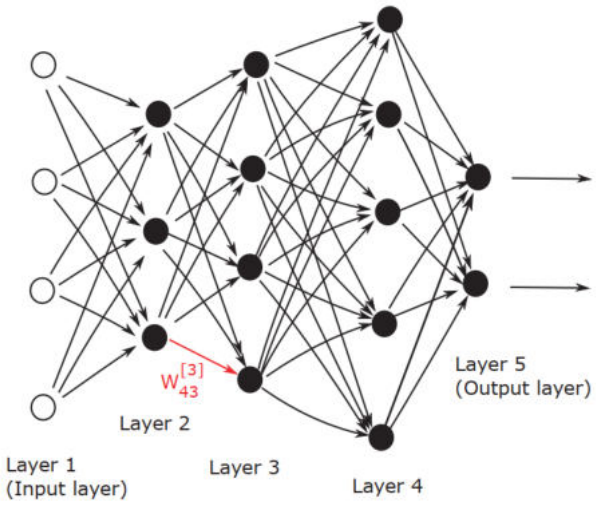
$W^{[4]} \in \mathbb{R}^{2\times 3}$ và một vector $b^{[4]} \in \mathbb{R}^2$ tương ứng. Đầu ra lớp thứ 4 là tính trên toàn bộ mạng neuron bằng công thức sau:

$$F(x) = \sigma\left(W^{[4]}\sigma\left(W^{[3]}\sigma\left(W^{[2]}x + b^{[2]}\right) + b^{[3]}\right) + b^{[4]}\right) \in \mathbb{R}^2. \tag{1.10}$$

Giả sử có $N = 10$ điểm dữ liệu (hoặc điểm huấn luyện) trong $\mathbb{R}^{n_1}$, mỗi điểm huấn luyện $\left\{x^{\{i\}}\right\}_{i=1}^{10}$ ta có đầu ra tương ứng $\left\{y\left(x^{\{i\}}\right)\right\}_{i=1}^{10}$ trong $\mathbb{R}^{n_L}$. Hàm chi phí có dạng

$$\text{Cost}\left(W^{[2]}, W^{[3]}, W^{[4]}, b^{[2]}, b^{[3]}, b^{[4]}\right) = \frac{1}{10}\sum_{i=1}^{10}\frac{1}{2}\left\|y\left(x^{\{i\}}\right) - F\left(x^{\{i\}}\right)\right\|_2^2 \tag{1.11}$$

Trường hợp tổng quát, giả sử một mạng neuron có L lớp, trong đó lớp 1 và lớp L là đầu vào và đầu ra. Mỗi lớp thứ l, với $l = 1, 2, 3, \ldots, L$, chứa n_l neuron. Số chiều của dữ liệu đầu vào là n_1, mạng neuron từ không gian $\mathbb{R}^{n_1}$ đến không gian $\mathbb{R}^{n_L}$. Chúng ta sử dụng ma trận trọng số $W^{[l]} \in \mathbb{R}^{n_l \times n_{l-1}}$ để biểu diễn trọng số tại lớp l. Cụ thể, $w_{jk}^{[l]}$ là trọng số áp dụng từ neuron k tại lớp $l-1$ đến neuron j tại lớp l. Tương tự, vector $b^{[l]} \in \mathbb{R}^{n_l}$ là độ lệch của lớp l, nghĩa là neuron j tại lớp l sử dụng độ lệch $b_j^{[l]}$.



Hình 1.3: Cấu trúc của ANN với L lớp ẩn.

Hình 1.3, xét mạng neuron gồm $L = 5$ lớp. Trong ví dụ này, $n_1 = 4, n_2 = 3, n_3 = 4, n_4 = 5$, $n_5 = 2$. Vì vậy, các ma trận trọng số là $W^{[2]} \in \mathbb{R}^{3\times 4}$, $W^{[3]} \in \mathbb{R}^{4\times 3}$, $W^{[4]} \in \mathbb{R}^{5\times 4}$, $W^{[5]} \in \mathbb{R}^{2\times 5}$, và các vector độ lệch là $b^{[2]} \in \mathbb{R}^3$, $b^{[3]} \in \mathbb{R}^4$, $b^{[4]} \in \mathbb{R}^5$, $b^{[5]} \in \mathbb{R}^2$. Cho một đầu vào $x \in \mathbb{R}^{n_1}$, chúng ta mô tả hoạt động của mạng bằng cách ký hiệu $a_j^{[l]}$ là đầu ra (hoặc *kích hoạt*) của neuron thứ j tại lớp thứ l. Khi đó, ta có các phương trình sau:

$$a^{[1]} = x \in \mathbb{R}^{n_1}, \tag{1.12}$$

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right) \in \mathbb{R}^{n_l} \quad \text{cho } l = 2, 3, \ldots, L. \tag{1.13}$$

Rõ ràng, phương trình (1.12) và (1.13) tạo thành một thuật toán để truyền dữ liệu đầu vào qua mạng để tạo một đầu ra là $a^{[L]} \in \mathbb{R}^{n_L}$. Thuật toán này được mô tả ở mục 5 của tài liệu [35] như một phương pháp để huấn luyện mạng. Giả sử chúng ta có N điểm dữ liệu huấn luyện $\left\{x^{(i)}\right\}_{i=1}^{N}$ trong không gian $\mathbb{R}^{n_1}$, và mỗi điểm huấn luyện đi kèm với đầu ra tương ứng $\left\{y\left(x^{(i)}\right)\right\}_{i=1}^{N}$ trong không gian $\mathbb{R}^{n_L}$. Tổng quát hóa công thức (1.11), hàm chi phí bậc hai mà chúng ta cần tối thiểu hóa có dạng:

$$\text{Cost} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\left\|y\left(x^{(i)}\right) - a^{[L]}\left(x^{(i)}\right)\right\|_2^2, \tag{1.14}$$

1.6.3. Phương pháp Gradient ngẫu nhiên

Để tối thiểu hóa hàm chi phí (cost function):

$$\min \text{Cost}(p) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\left\|y\left(x^{\{i\}}\right) - a^{[L]}\left(x^{\{i\}}\right)\right\|_2^2, p \in \mathbb{R}^s.$$

Trong đó,

$$C_{x^{\{i\}}}(p) = \frac{1}{2}\left\|y\left(x^{\{i\}}\right) - a^{[L]}\left(x^{\{i\}}\right)\right\|_2^2.$$

$$\nabla \text{Cost}(p) = \frac{1}{N}\sum_{i=1}^{N}\nabla C_{x^{\{i\}}}(p).$$

Thuật toán Gradient Descent (GD) và Stochastic Gradient Descent (SGD) là hai phương pháp quan trọng được sử dụng trong quá trình huấn luyện mạng neuron nhân tạo (ANN).

Thuật toán 3 : Giảm độ dốc (Gradient Descent).

1: Chọn $p^0 \in \mathbb{R}^s$ ban đầu.

2: $p^{k+1} = p^k - \eta\nabla \text{Cost}(p^k)$.

3: Khi k tiến đến vô cùng, $p^* = \lim_{k\to\infty} p^k$.

Thuật toán GD tính toán gradient của hàm mất mát trên toàn bộ tập dữ liệu đào tạo để cập nhật các trọng số của mạng neuron. Vì vậy thuật toán GD có thể cập nhật trọng số chậm hơn, đặc biệt khi dữ liệu đào tạo lớn.

Thuật toán 4 : Gradient ngẫu nhiên (Stochastic Gradient Descent).

1: Chọn $p^0 \in \mathbb{R}^s$ ban đầu.

2: Chọn ngẫu nhiên i từ tập $\{1, 2, \ldots, N\}$.

3: $p^{k+1} = p^k - \eta\nabla C_{x^{\{i\}}}(p^k)$.

4: Khi k tiến đến vô cùng, $p^* = \lim_{k\to\infty} p^k$.

Thuật toán SGD mỗi lần cập nhật, nó sử dụng chỉ một mẫu dữ liệu ngẫu nhiên từ tập dữ liệu đào tạo để tính gradient và cập nhật trọng số. Do đó, SGD thường cập nhật trọng số nhanh và hội tụ nhanh hơn GD. Ngoài ra, viêc sử dụng mẫu ngẫu nhiên có thể giúp tránh rơi vào điểm cực tiểu địa phương.

Lựa chọn giữa GD và SGD phụ thuộc vào bài toán cụ thể và kích thước dữ liệu đào tạo. SGD thường được ưa chuộng trong các tình huống khi tập dữ liệu lớn và khi muốn đạt được tốc độ học nhanh hơn.

1.6.4. Lan truyền ngược

Trong phần này, giới thiệu thuật toán lan truyền ngược (back propagation) để tính đạo hàm các hệ số trong mạng neuron nhân tạo.

Chúng ta viết lại

$$C_{x^{\{i\}}}(p) = \frac{1}{2}\left\|y\left(x^{\{i\}}\right) - a^{[L]}\left(x^{\{i\}}\right)\right\|_2^2,$$

để đơn giản ta viết lại như sau:

$$C = \frac{1}{2}\left\|y - a^{[L]}\right\|_2^2.$$

Nhắc lại $a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right) \in \mathbb{R}^{n_l}$ và đặt $z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]} \in \mathbb{R}^{n_l}$,

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right) = \sigma\left(z^{[l]}\right), \quad l = 1, 2, \ldots, L.$$

Định nghĩa

$$\delta^{[l]} = \left(\frac{\partial C}{\partial z_j^{[l]}}\right)^T_{1\le j\le n_l} = \left(\delta_j^{[l]}\right)^T_{1\le j\le n_l} \in \mathbb{R}^{n_l}, \quad l = 1, 2, \ldots, L$$

$z_j^{[l]}$ là đầu vào có trọng số cho neuron thứ j tại lớp l, $\delta_j^{[l]}$ được gọi là sai số trong neuron thứ j tại lớp l.

Bây giờ, chúng ta cũng cần định nghĩa phép nhân Hadamard. Cho $x = (x_1, x_2, \ldots, x_n), y = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^n$, tích Hadamard

$$x \circ y = (x_1y_1, x_2y_2, \ldots, x_ny_n)^T \in \mathbb{R}^n.$$

Với ký hiệu này, chúng ta có thể trình bày các kết quả sau đây, mà đều là hệ quả của đạo hàm hợp (chain rule).

Bổ đề 1.1. *Ta có:*

$$\delta^{[L]} = \sigma'\left(z^{[L]}\right) \circ \left(a^L - y\right), \tag{1.15}$$

$$\delta^{[l]} = \sigma'\left(z^{[l]}\right) \circ \left(W^{[l+1]}\right)^T \delta^{[l+1]} \quad cho\ 2 \leq l \leq L-1, \tag{1.16}$$

$$\partial b_j^{[l]} = \delta_j^{[l]} \quad cho\ 2 \leq l \leq L, \tag{1.17}$$

$$\frac{\partial C}{\partial w_{jk}^{[l]}} = \delta_j^{[l]} a_k^{[l-1]} \quad cho\ 2 \leq l \leq L. \tag{1.18}$$

Thuật toán 5 : Lan truyền ngược (Backpropagation).

procedure
  Cho số bước lặp từ 1 đến N_{iter}.
  Chọn một số nguyên k một cách ngẫu nhiên từ $\{1, 2, 3, \ldots, N\}$. $x^{\{k\}}$ là điểm dữ liệu huấn luyện hiện tại.
  $a^{[1]} = x^{\{k\}}$
  for $l = 2$ đến L **do**
    $z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]}$
    $a^{[l]} = \sigma(z^{[l]})$
    $D^{[l]} = \mathrm{diag}(\sigma'(z^{[l]}))$
    $\delta^{[L]} = D^{[L]}(a^{[L]} - y(x^{\{k\}}))$
  end for
  for $l = L - 1$ đến 2 **do**
    $\delta^{[l]} = D^{[l]}(W^{[l+1]})^T \delta^{[l+1]}$
  end for
  for $l = L$ đến 2 **do**
    $W^{[l]} \to W^{[l]} - \eta \delta^{[l]} {a^{[l-1]}}^T$
    $b^{[l]} \to b^{[l]} - \eta \delta^{[l]}$
  end for
end procedure

Trong thuật toán trên, N_{iter} là số lần lặp, L là số lớp ẩn trong mạng neuron, $W^{[l]}$ và $b^{[l]}$ là trọng số và độ lệch của lớp thứ l, σ là hàm kích hoạt, σ' là đạo hàm của hàm kích hoạt, $y(x^{\{k\}})$ là đầu ra mong muốn tương ứng với điểm dữ liệu huấn luyện $x^{\{k\}}$, η là tốc độ học (learning rate). Biến $\delta^{[l]}$ là sai số trong nueron tại lớp thứ l, $a^{[l]}$ đầu ra lớp thứ l, $z^{[l]}$ tổng trọng số đầu vào của lớp thứ l và $D^{[l]}$ là ma trận đường chéo chứa đạo hàm của hàm kích hoạt tại lớp thứ l.

1.6.5. Định lý xấp xỉ phổ quát

Định lý 1.5. *[27] Cho $\varphi : \mathbb{R} \to \mathbb{R}$ là một hàm số liên tục, bị chặn và không đồng nhất bằng hằng số. Ký hiệu I_m là hình vuông đơn vị trong $\mathbb{R}^m$, tức tập $[0,1]^m$. Ký hiệu $C(I_m)$ là không gian các hàm số liên tục trong I_m. Khi đó, với mọi $\varepsilon > 0$ và $f \in C(I_m)$, tồn tại số nguyên dương N, các số thực v_i, b_i và các vector $w_i \in \mathbb{R}^m, i = 1, 2, \ldots, N$ sao cho*

$$F(x) = \sum_{i=1}^{N} v_i \varphi\left(w_i^T x + b_i\right)$$

thỏa mãn

$$|F(x) - f(x)| < \varepsilon$$

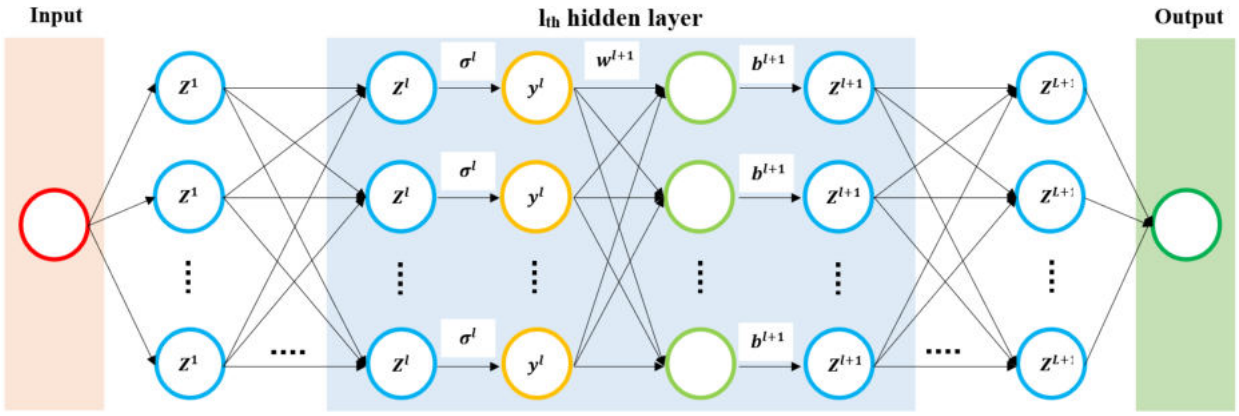
với mọi $x \in I_m$. Hay hàm số có dạng $F(x)$ trù mật trong $C(I_m)$. Khẳng định vẫn đúng khi thay I_m bằng tập compact bất kỳ.

Định lý 1.5 khẳng định về việc mạng neuron có thể xấp xỉ mọi hàm số liên tục trên tập compact. Tuy nhiên, phiên bản năm 1991 chỉ áp dụng cho mạng neuron một tầng ẩn với số neuron trong tầng đó không giới hạn, chứ không áp dụng cho mạng nhiều tầng ẩn. Nếu chỉ sử dụng một tầng ẩn, số lượng neuron trong đó có thể sẽ phải tăng lên tới hàng triệu khiến việc huấn luyện thực tế là bất khả thi. Cho tới 2017, Zhou Lu [42] và Hanin [43] đã chứng minh các phiên bản của định lý dành cho mạng có chiều sâu thay vì chiều rộng không giới hạn. Đây đã là cơ sở của rất nhiều hướng tiếp cận sử dụng mạng neuron trong các bài toán khác nhau và trong luận văn là sử dụng mạng neuron để xấp xỉ nghiệm của phương trình đạo hàm riêng.

Trên đây là toàn bộ lý thuyết tổng quan về cấu trúc của một ANN. Bây giờ, chúng tôi xét cấu trúc chi tiết của mạng neuron áp dụng cho Chương 2 và Chương 3. Giả sử mạng gồm L lớp ẩn, trong đó lớp đầu vào và đầu ra được ký hiệu là lớp 0 và $L+1$. Mỗi lớp của mạng sử dụng một hàm kích hoạt phi tuyến σ. Về mặt toán học mạng ANN được xem là một ánh xạ từ không gian $\mathbb{R}^N$ đến không gian $\mathbb{R}$, có cấu trúc như Hình 1.4

Dựa trên cấu trúc của ANN, ta có thể hiểu rằng z_l là đầu vào, σ_l là hàm kích hoạt, w^l là trọng số và b^l là độ lệch của lớp l với $l = 0, 1, ..., L$. Công thức biểu diễn mạng neuron giữa các lớp liền kề như sau:

$$\mathbf{z}^{l+1} = \mathbf{w}^{l+1}\mathbf{y}^l + \mathbf{b}^{l+1},$$
$$\mathbf{y}^{l+1} = \sigma_{l+1}\left(\mathbf{z}^{l+1}\right)$$



Hình 1.4: Cấu trúc của ANN với L lớp ẩn

Ở đây, $\mathbf{y}^{l+1}$ đại diện cho cả đầu vào và đầu ra của lớp ẩn thứ $l+1$ và lớp thứ l. Ký hiệu $\mathbf{w} = \{\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^{L+1}\}$, $\mathbf{b} = \{\mathbf{b}^1, \mathbf{b}^2, \ldots, \mathbf{b}^{L+1}\}$ và đầu vào được ký hiệu là x. Để đơn giản, chúng tôi định nghĩa đầu ra như sau:

$$\mathbf{y}^{L+1} := NET(\mathbf{x}; \mathbf{w}, \mathbf{b}),$$

điều này cho thấy rằng ANN nhận đầu vào $\mathrm{x} \in \mathbb{R}^N$ và được tham số hóa bởi $\mathbf{w}$ và $\mathbf{b}$.

Chương 2

GIẢI BÀI TOÁN CAUCHY CHO PHƯƠNG TRÌNH ĐẠO HÀM RIÊNG BẰNG CHỈNH HÓA TIKHONOV VỚI MẠNG NEURON NHÂN TẠO

Trong phần này, chúng tôi sẽ giới thiệu phương pháp sử dụng mạng neuron nhân tạo kết hợp với chỉnh hóa Tikhonov để xấp xỉ bài toán Cauchy cho phương trình đạo hàm riêng. Trong [30], các tác giả đã sử dụng mạng neuron để giải bài toán Cauchy, tuy nhiên, chưa áp dụng phương pháp chỉnh hóa Tikhonov cho bài toán. Do đó, kết quả thu được không ổn định trong nhiều trường hợp bài toán phức tạp, có số chiều lớn hoặc bài toán có độ phi tuyến cao. Vấn đề quan trọng trong bài toán Cauchy là làm thế nào để xử lý tính đặt không chỉnh trong quá trình xấp xỉ, và đây vẫn là một thách thức lớn trong lĩnh vực nghiên cứu này.

Trong chương này, đầu tiên chúng tôi trình bày mô hình mạng neuron nhân tạo kết hợp với phương pháp chỉnh hóa Tikhonov cho bài toán Cauchy cho phương trình elliptic (0.1) trong mục 2.1 và parabolic (0.2) trong mục 2.2. Sau đó, chúng tôi sẽ trình bày thuật toán để huấn luyện mạng trong mục 2.3. Cuối cùng, chúng tôi trình bày về sự hội tụ của xấp xỉ mạng neuron trong mục 2.4.

Để áp dụng chỉnh hóa Tikhonov cho bài toán (0.1), chúng ta cần chuyển bài toán về dạng phương trình toán tử sau

$$Au = b, \text{ trong đó } Au = \begin{bmatrix} \mathcal{L}u \\ u|_\Gamma \\ \frac{\partial u}{\partial \mathbf{n}}|_\Gamma \end{bmatrix}, b = \begin{bmatrix} 0 \\ f \\ g \end{bmatrix}. \tag{2.1}$$

Bài toán (0.2) cũng viết tương tự. Ở đây, A là toán tử tuyến tính và đơn ánh đi từ không gian $\mathbb{R}^n$ vào không gian $\mathbb{R}^m$ trên trường $\mathbb{R}$. Hơn nữa, dữ liệu vế phải b không

được biết chính xác mà chỉ có dữ liệu xấp xỉ b^δ của b với $\delta > 0, b^\delta \in \mathbb{R}$ thỏa mãn:

$$\left\|b - b^\delta\right\| \leq \delta. \tag{2.2}$$

Chúng ta luôn giả sử rằng phương trình không bị nhiễu (2.1) tồn tại một nghiệm u. Nói cách khác, chúng ta giả sử rằng $b \in R(A)$. Với A là hàm đơn ánh thì u là nghiệm duy nhất.

Thông thường, để tìm nghiệm xấp xỉ cho bài toán (2.1), người ta nghĩ đến việc tìm u^δ, nghiệm của phương trình:

$$Au^\delta = b^\delta,$$

và xem u^δ là giá trị xấp xỉ của nghiệm chính xác u. Chúng ta biết rằng điều này chỉ đúng khi bài toán (2.1) là đặt chỉnh và $b^\delta \in R(A)$. Chú ý, phương trình trên có thể không giải được vì chúng ta không thể đảm bảo dữ liệu được đo b^δ nằm trong miền giá trị $R(A)$. Hơn nữa, ngay cả khi phương trình giải được, vì bài toán (2.1) là đặt không chỉnh nên u^δ không phải là nghiệm xấp xỉ của nghiệm chính xác u.

Như đã được đề cập trước đó, bài toán Cauchy là một bài toán đặt không chỉnh. Vì vậy, trong phần này, chúng ta sẽ kết hợp mạng neuron nhân tạo (ANN) với phương pháp chỉnh hóa Tikhonov để giải bài toán Cauchy cho phương trình elliptic và parabolic.

2.1. Áp dụng chỉnh hóa Tikhonov cho bài toán Cauchy trong phương trình elliptic

Theo cách tiếp cận của Li và Hu [30] là tìm nghiệm $\bar{u}$ cho bài toán (0.1) dưới dạng đầu ra của mạng neuron $NET(\mathbf{x}; \mathbf{w}, \mathbf{b})$. Định nghĩa hàm mục tiêu

$$J(\bar{u}) = \|\mathcal{L}\bar{u}\|^2_{L_2(\Omega)} + \|\bar{u} - f\|^2_{L_2(\Gamma)} + \left\|\frac{\partial \bar{u}}{\partial \mathbf{n}} - g\right\|^2_{L_2(\Gamma)}. \tag{2.3}$$

Khi đó, phương pháp ANN cho bài toán (0.1) được viết như sau:

$$\begin{aligned} &\min_{\mathbf{w},\mathbf{b}} J(\bar{u}) \\ &\text{sao cho } \bar{u} = NET(\mathbf{x}; \mathbf{w}, \mathbf{b}), \end{aligned} \tag{2.4}$$

ở đây, $NET : \Omega \to \mathbb{R}$ là một ánh xạ của mạng neuron, $\mathbf{w}$ và $\mathbf{b}$ là các tham số trong mạng.

Trong phần này, chúng ta nghiên cứu phương pháp chỉnh hóa Tikhonov cho bài toán (0.1). Do đó, họ toán tử chỉnh hóa được định nghĩa như sau:

$$\begin{aligned} &R_\alpha b := \min_{\mathbf{w},\mathbf{b}} J_\alpha(\bar{u}), \alpha > 0 \\ &\text{sao cho } \bar{u} = NET(\mathbf{x}; \mathbf{w}, \mathbf{b}) \end{aligned} \tag{2.5}$$

trong đó, $J_\alpha(\bar{u})$ là phiếm hàm Tikhonov xác định bởi:

$$J_\alpha(\bar{u}) = \|\mathcal{L}\bar{u}\|^2_{L_2(\Omega)} + \|\bar{u} - f\|^2_{L_2(\Gamma)} + \left\|\frac{\partial \bar{u}}{\partial n} - g\right\|^2_{L_2(\Gamma)} + \alpha\Phi(u) \tag{2.6}$$

ở đây, α là hệ số chỉnh hóa, $\Phi(u)$ là hàm chỉnh hóa có thể là các hàm sau:

$$\left|\bar{u}\right|^2_{L_2(\Gamma 2)} = \sum_{i=1}^{N_{\Gamma 2}} (\bar{u}_i)^2, \quad \left|\bar{u}\right|^2_{L_2(\Omega)} = \sum_{i=1}^{N_o} (\bar{u}_i)^2, \quad \left|\frac{\partial \bar{u}}{\partial n}\right|^2_{L_2(\Gamma 2)} = \sum_{i=1}^{N_{\Gamma 2}} \left(\frac{\partial \bar{u}_i}{\partial \mathbf{n}}\right)^2,$$

$$\begin{aligned}\left|\bar{u}\right|^2_{H_1(\Omega)} &= \left|\bar{u}\right|^2_{L_2(\Omega)} + \left|\nabla \bar{u}\right|^2_{L_2(\Omega)} \\ &= \sum_{i=1}^{N_o} (\bar{u}_i)^2 + \sum_{i=1}^{N_o} \left(\left(\frac{\partial \bar{u}_i}{\partial x_1}\right)^2 + \left(\frac{\partial \bar{u}_i}{\partial x_2}\right)^2 + ... + \left(\frac{\partial \bar{u}_i}{\partial x_d}\right)^2\right).\end{aligned}$$

Để giải bài toán (2.6), chúng ta lấy mẫu ngẫu nhiên một số điểm $\mathbf{x}_{in} = [x_1, x_2, \ldots, x_N]$ trong Ω, với N_o, N_d, N_n, N_k là số điểm thuộc Ω, Γ (điều kiện biên Dirichlet), Γ (điều kiện biên Neumann), N_k (số điểm chỉnh hóa), tương ứng sao cho $N_o+N_d+N_n+N_k = N$. Để đánh giá tính ổn định của xấp xỉ, chúng ta thêm thủ công nhiễu thống kê vào dữ liệu nhãn f, g như sau:

$$\left\|f^\delta - f\right\|_\Gamma \leq \delta, \quad \left\|g^\delta - g\right\|_\Gamma \leq \delta,$$

trong đó δ là mức độ nhiễu thống kê. Để thuận tiện trong tính toán giải số, hàm mục tiêu (2.6) có thể được biểu diễn dưới dạng rời rạc như sau:

$$\begin{aligned}J_\alpha(\bar{u}) &= J_o(\bar{u}) + J_d(\bar{u}) + J_n(\bar{u}) + \alpha\Phi(u) \\ &= \sum_{i=1}^{N_o} (\mathcal{L}\bar{u}_i)^2 + \sum_{i=1}^{N_d} \left(\bar{u}_i - f_i^\delta\right)^2 + \sum_{i=1}^{N_n} \left(\frac{\partial \bar{u}_i}{\partial \mathbf{n}} - g_i^\delta\right)^2 + \alpha\Phi(u)\end{aligned} \tag{2.7}$$

trong đó, $\bar{u}_i = NET(\mathbf{x}_i; \mathbf{w}, \mathbf{b})$, $f_i^\delta = f^\delta(\mathbf{x}_i)$, $g_i^\delta = g^\delta(\mathbf{x}_i)$.

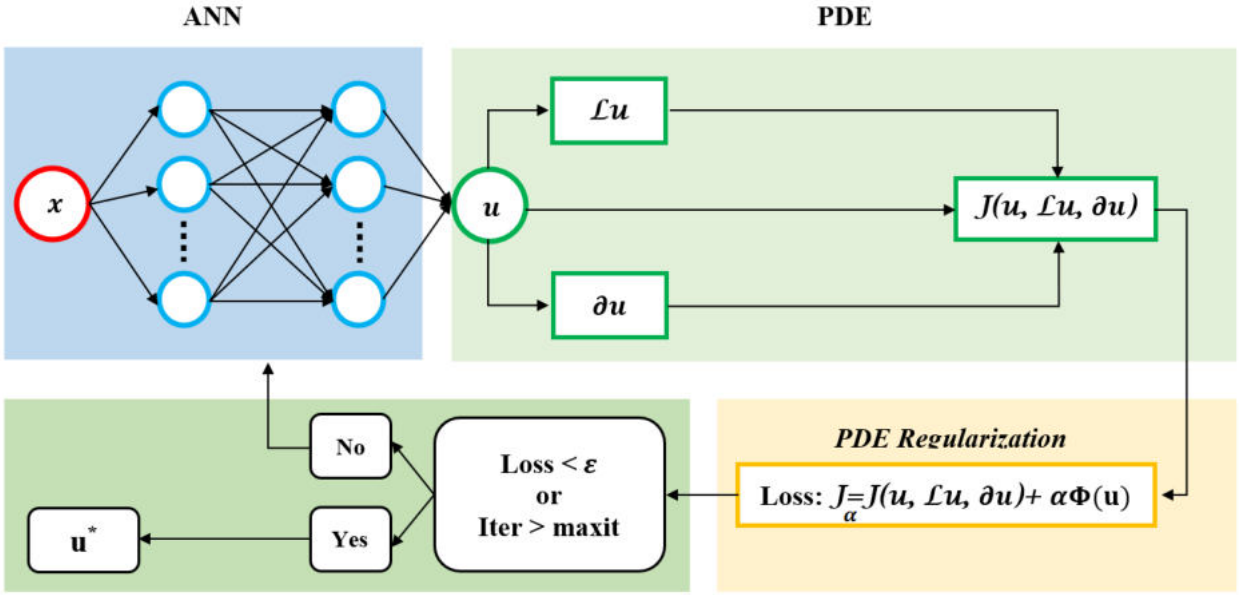
Áp dụng thuật toán lan truyền ngược để tính các đạo hàm sau

$$\begin{aligned}\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{w}} &= \frac{\partial J_o(\bar{u})}{\partial \mathbf{w}} + \frac{\partial J_d(\bar{u})}{\partial \mathbf{w}} + \frac{\partial J_n(\bar{u})}{\partial \mathbf{w}} + \alpha\frac{\partial \Phi(u)}{\partial \mathbf{w}} \\ &= 2\left(\sum_{i=1}^{N_o} \mathcal{L}\bar{u}_i \frac{\partial \mathcal{L}\bar{u}_i}{\partial \mathbf{w}} + \sum_{i=1}^{N_d} \left(\bar{u}_i - f_i^\delta\right)\frac{\partial \bar{u}_i}{\partial \mathbf{w}} + \sum_{i=1}^{N_n} \left(\frac{\partial \bar{u}_i}{\partial \mathbf{n}} - g_i^\delta\right)\frac{\partial^2 \bar{u}_i}{\partial \mathbf{n}\partial \mathbf{w}}\right) + \alpha\frac{\partial \Phi(u)}{\partial \mathbf{w}}.\end{aligned} \tag{2.8}$$

Tương tự,

$$\begin{aligned}\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{b}} &= \frac{\partial J_o(\bar{u})}{\partial \mathbf{b}} + \frac{\partial J_d(\bar{u})}{\partial \mathbf{b}} + \frac{\partial J_n(\bar{u})}{\partial \mathbf{b}} + \alpha\frac{\partial \Phi(u)}{\partial \mathbf{b}} \\ &= 2\left(\sum_{i=1}^{N_o} \mathcal{L}\bar{u}_i \frac{\partial \mathcal{L}\bar{u}_i}{\partial \mathbf{b}} + \sum_{i=1}^{N_d} \left(\bar{u}_i - f_i^\delta\right)\frac{\partial \bar{u}_i}{\partial \mathbf{b}} + \sum_{i=1}^{N_n} \left(\frac{\partial \bar{u}_i}{\partial \mathbf{n}} - g_i^\delta\right)\frac{\partial^2 \bar{u}_i}{\partial \mathbf{n}\partial \mathbf{b}}\right) + \alpha\frac{\partial \Phi(u)}{\partial \mathbf{b}}.\end{aligned} \tag{2.9}$$

Cách tính $\mathcal{L}\bar{u}$, $\frac{\partial \bar{u}}{\partial \mathbf{n}}$, và $\bar{u}$ cùng với quá trình lan truyền ngược tương ứng sẽ được trình bày chi tiết trong phần phụ lục của luận văn. Ta có thể hiểu phương pháp ANN khi áp dụng chỉnh hóa Tikhonov cho bài toán (0.1) bằng sơ đồ trong Hình 2.1 dưới đây



Hình 2.1: Sơ đồ ANN để giải bài toán Cauchy trong trường áp dụng chỉnh hóa Tikhonov.

Bằng cách thêm hàm chỉnh hóa vào hàm mục tiêu $J(\bar{u})$, chúng ta thu được một hàm mục tiêu mới $J_\alpha(\bar{u})$ như trong công thức (2.7). Quá trình tối ưu hiện tại là tìm giá trị của biến $\bar{u}$ để hàm mục tiêu mới này đạt giá trị nhỏ nhất.

2.2. Áp dụng chỉnh hóa Tikhonov cho bài toán Cauchy trong phương trình parabolic

Tương tự, trong [30] tác giả đã định nghĩa hàm mục tiêu cho bài toán (0.2) như sau:

$$\begin{aligned} J(\bar{u}) = &\left\| \frac{\partial \bar{u}}{\partial t} + \mathcal{L}\bar{u} \right\|^2_{L_2(\Omega\times\mathcal{T})} \\ &+ \|\bar{u} - f\|^2_{L_2(\Gamma\times\mathcal{T})} + \left\| \frac{\partial \bar{u}}{\partial \mathbf{n}} - g \right\|^2_{L_2(\Gamma\times\mathcal{T})} + \|\bar{u} - h\|^2_{L_2(\Omega)}, \end{aligned} \tag{2.10}$$

bài toán (0.2) có thể được viết lại là

$$\begin{aligned} &\min_{\mathbf{w},\mathbf{b}} J(\bar{u}) \\ &\bar{u} = NET(\mathbf{x}, t; \mathbf{w}, \mathbf{b}) \end{aligned} \tag{2.11}$$

trong đó, $\mathbf{w}$ và $\mathbf{b}$ là các tham số trong mạng.

Trong phần này, chúng ta nghiên cứu phương pháp chỉnh hóa Tikhonov cho bài toán (0.2). Do đó, họ toán tử chỉnh hóa được định nghĩa như sau:

$$\begin{aligned} &R_\alpha b := \min_{\mathbf{w},\mathbf{b}} J_\alpha(\bar{u}), \alpha > 0 \\ &\text{sao cho } \bar{u} = NET(\mathbf{x}; \mathbf{w}, \mathbf{b}) \end{aligned} \tag{2.12}$$

trong đó, $J_\alpha(\bar{u})$ là phiếm hàm Tikhonov xác định bởi:

$$\begin{aligned} J_\alpha(\bar{u}) = &\left\| \frac{\partial \bar{u}}{\partial t} + \mathcal{L}\bar{u} \right\|^2_{L_2(\Omega\times\mathcal{T})} \\ &+ \|\bar{u} - f\|^2_{L_2(\Gamma\times\mathcal{T})} + \left\| \frac{\partial \bar{u}}{\partial \mathbf{n}} - g \right\|^2_{L_2(\Gamma\times\mathcal{T})} + \|\bar{u} - h\|^2_{L_2(\Omega)} + \alpha\Phi(u) \end{aligned} \tag{2.13}$$

ở đây, α là hệ số chỉnh hóa, $\Phi(u)$ là hàm chỉnh hóa có thể là một trong các hàm sau: $\left|\bar{u}\right|^2_{L_2(\Gamma 2)}$, $\left|\bar{u}\right|^2_{L_2(\Omega)}$, $\left|\frac{\partial \bar{u}}{\partial n}\right|^2_{L_2(\Gamma 2)}$, $\left|\bar{u}\right|^2_{H_1(\Omega)}$.

Để giải bài toán (2.13), chúng ta lấy ngẫu nhiên một số điểm $\mathbf{x}_{in} = [x_1, x_2, \ldots, x_N]$ trong không gian $\Omega \times \mathcal{T}$, với N_o, N_d, N_n, N_t, N_k là số điểm lấy mẫu thuộc $\Omega \times \mathcal{T}$, $\Gamma \times \mathcal{T}$ (điều kiện biên Dirichlet), $\Gamma \times \mathcal{T}$ (điều kiện biên Neumann), N_k (số điểm chỉnh hóa), tương ứng sao cho $N_o + N_d + N_n + N_t + N_k = N$. Để đánh giá tính ổn định của xấp xỉ, chúng ta thêm thủ công nhiễu thống kê vào dữ liệu nhãn f, g, h, sao cho

$$\left\|f^\delta - f\right\|_\Gamma \leq \delta, \quad \left\|g^\delta - g\right\|_\Gamma \leq \delta, \quad \left\|h^\delta - h\right\|_\Omega \leq \delta,$$

trong đó δ là mức độ nhiễu thống kê. Để thuận tiện trong tính toán giải số, hàm mục tiêu (2.13) được viết dưới dạng rời rạc như

$$\begin{aligned} J_\alpha(u) &= J_o(u) + J_d(u) + J_n(u) + J_t(u) + \alpha\Phi(u) \\ &= \sum_{i=1}^{N_o} \left(\frac{\partial \bar{u}_i}{\partial t} + \mathcal{L}\bar{u}_i \right)^2 \\ &+ \sum_{i=1}^{N_d} \left(\bar{u}_i - f_i^\delta\right)^2 + \sum_{i=1}^{N_n} \left(\frac{\partial \bar{u}_i}{\partial \mathbf{n}} - g_i^\delta \right)^2 + \sum_{i=1}^{N_t} \left(\bar{u}_i - h_i^\delta\right)^2 + \alpha\Phi(u), \end{aligned} \tag{2.14}$$

trong đó, $\bar{u}_i = NET\left(\mathbf{x}_i, t_i; \mathbf{w}, \mathbf{b}\right), f_i^\delta = f^\delta\left(\mathbf{x}_i\right), g_i^\delta = g^\delta\left(\mathbf{x}_i\right), h_i^\delta = h^\delta\left(\mathbf{x}_i\right)$.

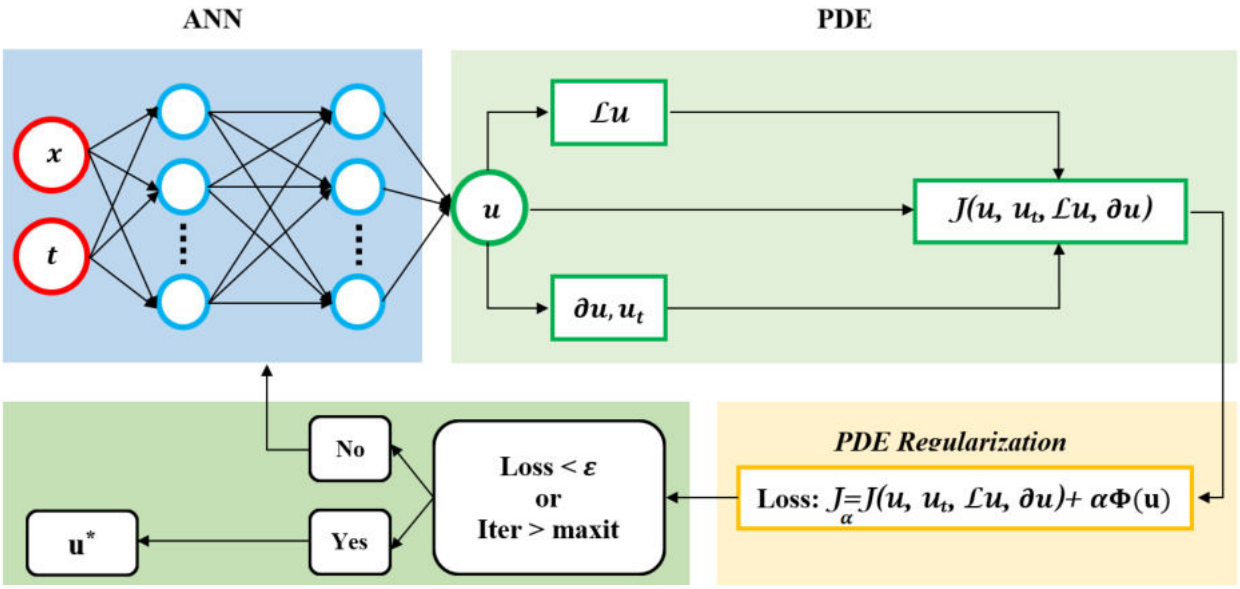
Áp dụng thuật toán lan truyền ngược để tính các đạo hàm sau

$$\begin{aligned} \frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{w}} &= \frac{\partial J_o(\bar{u})}{\partial \mathbf{w}} + \frac{\partial J_d(\bar{u})}{\partial \mathbf{w}} + \frac{\partial J_n(\bar{u})}{\partial \mathbf{w}} + \frac{\partial J_t(\bar{u})}{\partial \mathbf{w}} + \alpha\frac{\partial \Phi(u)}{\partial \mathbf{w}} \\ &= 2\left[\sum_{i=1}^{N_o} \left(\frac{\partial \bar{u}_i}{\partial t} + \mathcal{L}\bar{u}_i \right) \left(\frac{\partial^2 \bar{u}_i}{\partial t \partial \mathbf{w}} + \frac{\partial \mathcal{L}\bar{u}_i}{\partial \mathbf{w}} \right) \right. \\ &+ \sum_{i=1}^{N_n} \left(\frac{\partial \bar{u}_i}{\partial \mathbf{n}} - g_i^\delta \right) \frac{\partial^2 \bar{u}_i}{\partial \mathbf{n} \partial \mathbf{w}} + \sum_{i=1}^{N_d} \left(\bar{u}_i - f_i^\delta\right) \frac{\partial \bar{u}_i}{\partial \mathbf{w}} \\ &\left. + \sum_{i=1}^{N_t} \left(\bar{u}_i - h_i^\delta\right) \frac{\partial \bar{u}_i}{\partial \mathbf{w}} \right] + \alpha\frac{\partial \Phi(u)}{\partial \mathbf{w}}. \end{aligned} \tag{2.15}$$

Tương tự,

$$\begin{aligned}\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{b}} &= \frac{\partial J_o(\bar{u})}{\partial \mathbf{b}} + \frac{\partial J_d(\bar{u})}{\partial \mathbf{b}} + \frac{\partial J_n(\bar{u})}{\partial \mathbf{b}} + \frac{\partial J_t(\bar{u})}{\partial \mathbf{b}} + \alpha \frac{\partial \Phi(u)}{\partial \mathbf{b}}, \\ =2 &\left[\sum_{i=1}^{N_o} \left(\frac{\partial \bar{u}_i}{\partial t} + \mathcal{L}\bar{u}_i \right) \left(\frac{\partial^2 \bar{u}_i}{\partial t \partial \mathbf{b}} + \frac{\partial \mathcal{L}\bar{u}_i}{\partial \mathbf{b}} \right) \right. \\ &+ \sum_{i=1}^{N_n} \left(\frac{\partial \bar{u}_i}{\partial \mathbf{n}} - g_i^\delta \right) \frac{\partial^2 \bar{u}_i}{\partial \mathbf{n} \partial \mathbf{b}} + \sum_{i=1}^{N_d} \left(\bar{u}_i - f_i^\delta \right) \frac{\partial \bar{u}_i}{\partial \mathbf{b}} \\ &\left. + \sum_{i=1}^{N_t} \left(\bar{u}_i - h_i^\delta \right) \frac{\partial \bar{u}_i}{\partial \mathbf{b}} \right] + \alpha \frac{\partial \Phi(u)}{\partial \mathbf{b}}.\end{aligned} \tag{2.16}$$

Cách tính $\frac{\partial \bar{u}}{\partial t}, \mathcal{L}\bar{u}, \frac{\partial \bar{u}}{\partial \mathbf{n}}, \bar{u}$ và lan truyền ngược tương ứng được trình bày chi tiết ở phần phụ lục của luận văn. Ta có thể hiểu phương pháp ANN khi áp dụng chỉnh hóa Tikhonov cho bài toán (0.2) bằng sơ đồ trong Hình 2.2 dưới đây



Hình 2.2: Sơ đồ ANN để giải bài toán Cauchy trong trường áp dụng chỉnh hóa Tikhonov.

Bằng cách thêm hàm chỉnh hóa vào hàm mục tiêu $J(\bar{u})$, chúng ta thu được một hàm mục tiêu mới $J_\alpha(\bar{u})$ như trong công thức (2.14). Quá trình tối ưu hiện tại là tìm giá trị của biến $\bar{u}$ để hàm mục tiêu mới này đạt giá trị nhỏ nhất.

2.3. Thuật toán huấn luyện mạng

Thuật toán ban đầu cho các mạng neuron là phương pháp giảm độ dốc (GD). Nó được thực hiện như sau:

$$\mathbf{w}^{n+1} = \mathbf{w}^n - \Delta t \frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{w}}, \qquad \mathbf{b}^{n+1} = \mathbf{b}^n - \Delta t \frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{b}}$$

trong đó, n là số lần lặp và Δt là bước thời gian. Tuy nhiên, trong nghiên cứu của chúng tôi, thuật toán Gradient Descent (GD) không đạt được kết quả số tốt như chúng tôi mong đợi. Trong [30], các tác giả đã sử dụng thuật toán ADAM. Chúng ta đều biết rằng thuật toán ADAM là một thuật toán tối ưu ngẫu nhiên ổn định và nhanh. Tuy nhiên, dường như thuật toán ADAM không hiệu quả cho các bài toán Cauchy phi tuyến 2D và 3D. Vì vậy, để cải thiện quá trình tối ưu hóa mô hình, chúng tôi đã phải kết hợp hai thuật toán tối ưu là ADAM và L-BFGS. Chúng tôi kỳ vọng rằng bằng cách sử dụng cả hai thuật toán này, sẽ tăng cường hiệu suất của quá trình tối ưu hóa và đạt được kết quả tốt hơn trong việc huấn luyện và ước lượng các tham số của mô hình.

Công thức chính của ADAM cho việc cập nhật trọng số $\mathbf{w}$ được biểu diễn như sau:

$$\begin{cases} \mathbf{w}^{n+1} = \mathbf{w}^n - \Delta t \frac{v_w^{n+1}}{\sqrt{s_w^{n+1}}+\epsilon} \frac{\partial J_\alpha(\bar{u})}{\partial w^n} \\ v_w^{n+1} = \frac{\beta_1 v_w^n + (1-\beta_1)\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{w}^n}}{1-\beta_1^n} \\ s_w^{n+1} = \frac{\beta_2 s_w^n + (1-\beta_2)\left(\frac{\partial J_\alpha(\bar{u})}{\partial w^n}\right)^2}{1-\beta_2^n} \end{cases} \quad \begin{cases} \mathbf{b}^{n+1} = \mathbf{b}^n - \Delta t \frac{v_b^{n+1}}{\sqrt{s_b^{n+1}}+\epsilon} \frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{b}^n} \\ v_b^{n+1} = \frac{\beta_1 v_b^n + (1-\beta_1)\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{b}^n}}{1-\beta_1^n} \\ s_b^{n+1} = \frac{\beta_2 s_b^n + (1-\beta_2)\left(\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{b}^n}\right)^2}{1-\beta_2^n} \end{cases} \tag{2.17}$$

Trong cả hai công thức trên, $\mathbf{w}^n$ và $\mathbf{b}^n$ là các trọng số và bias tại vòng lặp thứ n; Δt đại diện cho bước thời gian; v_w^n, s_w^n, v_b^n, s_b^n là các ma trận tham số tại vòng lặp thứ n; β_1 và β_2 là các hằng số trong khoảng từ 0 đến 1, đại diện cho tỉ lệ giảm gradient và tỉ lệ giảm gradient bình phương, tương ứng; ϵ là một hằng số nhỏ được thêm vào trong mẫu số để tránh việc chia cho 0.

Thuật toán 6 : Giải bài toán Cauchy bằng thuật toán ADAM

procedure
  Đầu vào: $\mathbf{x}^i \in \Omega$ hoặc $\Omega \times \mathcal{T}$; Dữ liệu $f^\delta, g^\delta, h^\delta$; Số lớp ẩn L; Số nơron N;
  Khởi tạo cấu trúc mạng neuarl; Trọng số $\mathbf{w}^l$, bias $\mathbf{b}^l$ và các tham số khác;
  while Iter $\leq np$ **do**
    Tính kết quả đầu ra $\mathbf{y}^{L+1}$ với mạng;
    Tính hàm mục tiêu J_α cho bởi (2.7) và (2.14);
    Tính gradient theo $\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{w}}$ và $\frac{\partial J_\alpha(\bar{u})}{\partial \mathbf{b}}$;
    Cập nhật $\mathbf{w}^l$ và $\mathbf{b}^l$ bằng thuật toán (2.17);
  end while
  Đầu ra: Quá trình của hàm mục tiêu J_α;
  Đầu ra: Nghệm $\bar{u}$ của PDEs;
end procedure

L-BFGS (Limited-memory Broyden–Fletcher–Goldfarb–Shanno) [44] là một thuật toán tối ưu hóa phổ biến được sử dụng trong việc tối ưu hóa tham số cho mạng neuron. Thuật toán này thuộc loại các thuật toán quasi-Newton, và nổi tiếng vì khả năng hiệu quả và linh hoạt trong việc giải quyết các bài toán tối ưu hóa không ràng buộc. Với cấu trúc lưu trữ hạn chế, L-BFGS rất phù hợp với việc tối ưu hóa mô hình mạng neuron, đặc biệt là khi mô hình có số lượng tham số lớn. Thay vì tính toán ma trận Hessian đầy đủ như các phương pháp Newton truyền thống, L-BFGS chỉ cần lưu trữ một số xấp xỉ của ma trận Hessian để giảm chi phí tính toán. Công thức chính của thuật toán L-BFGS cho việc cập nhật trọng số $\mathbf{w}$ và bias $\mathbf{b}$ là như sau:

$$\begin{cases} \mathbf{w}^{n+1} = \mathbf{w}^n - \beta \mathbf{H}_n^{-1} \nabla J_\alpha(\mathbf{w}^n), \\ \mathbf{b}^{n+1} = \mathbf{b}^n - \beta \mathbf{H}_n^{-1} \nabla J_\alpha(\mathbf{b}^n). \end{cases} \tag{2.18}$$

Trong đó, $\mathbf{w}^n$ và $\mathbf{b}^n$ là trọng số và bias tại vòng lặp thứ n; $\nabla J_\alpha(\mathbf{w}^n)$ và $\nabla J_\alpha(\mathbf{b}^n)$ là gradient của hàm mục tiêu J_α tại $\mathbf{w}^n$ và $\mathbf{b}^n$; $\mathbf{H}_n^{-1}$ là ma trận xác định dương nghịch đảo BFGS được xấp xỉ từ lịch sử của các bước trước đó và được cập nhật theo quy tắc BFGS trong quá trình lặp lại của thuật toán và được biểu diễn như sau:

$$\mathbf{H}_n^{-1} = \left(\mathbf{I} - \rho_n \mathbf{s}_n \mathbf{y}_n^T\right) \mathbf{H}_{n-1}^{-1} \left(\mathbf{I} - \rho_n \mathbf{y}_n \mathbf{s}_n^T\right) + \rho_n \mathbf{s}_n \mathbf{s}_n^T, \tag{2.19}$$

với $\mathbf{H}_{n-1}^{-1}$ là ma trận xác định dương nghịch đảo BFGS của vòng lặp trước, $\mathbf{I}$ là ma trận đơn vị, ρ_n là hệ số cập nhật được tính toán dựa trên các vector $\mathbf{s}_n$ và $\mathbf{y}_n$:

$$\rho_n = \frac{1}{\mathbf{y}_n^T \mathbf{s}_n}, \tag{2.20}$$

$\mathbf{s}_n = \mathbf{w}^{n+1} - \mathbf{w}^n$ là vector chênh lệch giữa các trọng số tại hai vòng lặp liên tiếp, $\mathbf{y}_n = \nabla J_\alpha(\mathbf{w}^{n+1}) - \nabla J_\alpha(\mathbf{w}^n)$ là vector chênh lệch giữa các gradient tại hai vòng lặp liên tiếp. Công thức mô tả quá trình cập nhật ma trận xác định dương nghịch đảo BFGS $\mathbf{H}_n^{-1}$ dựa trên lịch sử của các vector $\mathbf{s}_n$ và $\mathbf{y}_n$. Quá trình này giúp xấp xỉ ma trận Hessian của hàm mục tiêu và đảm bảo tính chất xác định dương của ma trận xác định dương nghịch đảo $\mathbf{H}_n^{-1}$.

Trong công thức trên, β là một hệ số học được chọn sao cho thuật toán hội tụ tốt. Các giá trị khác nhau của β có thể được thử nghiệm để tìm ra giá trị tối ưu cho bài toán cụ thể. Việc cập nhật trọng số $\mathbf{w}$ và bias $\mathbf{b}$ dựa trên công thức trên giúp tối ưu hóa hàm mục tiêu và điều chỉnh các tham số của mạng neuron để đạt được kết quả tốt nhất khi giải bài toán Cauchy.

Thuật toán 7 : Giải bài toán Cauchy bằng thuật toán L-BFGS

procedure
  Đầu vào: $\mathbf{x}^i \in \Omega$ hoặc $\Omega \times \mathcal{T}$; Dữ liệu $f^\delta, g^\delta, h^\delta$;
  Đầu vào:Số lớp ẩn L; Số nơron N; Hệ số học (learning rate) β;
  Khởi tạo cấu trúc mạng neuarl;
  Khởi tạo trọng số (weights) $\mathbf{w}^l$, bias $\mathbf{b}^l$ và các tham số khác;
  Khởi tạo các ma trận nghịch đảo $\mathbf{H}_n^{-1}$ của ma trận Hessian xấp xỉ;
  while Iter $\leq np$ **do**
    Tính kết quả đầu ra $\mathbf{y}^{L+1}$ với mạng;
    Tính hàm mục tiêu J_α cho bởi (2.7) và (2.14);
    Tính gradient $\nabla J_\alpha(\mathbf{w}^n)$ và $\nabla J_\alpha(\mathbf{b}^n)$;
    Cập nhật ma trận xác định dương nghịch đảo $\mathbf{H}_k^{-1}$;
    Cập nhật trọng số $\mathbf{w}^l$ và bias $\mathbf{b}^l$ theo (2.18);
  end while
  Đầu ra: Quá trình của hàm mục tiêu J_α;
  Đầu ra: Nghệm $\bar{u}$ của PDEs;
end procedure

Thuật toán này sử dụng phương pháp L-BFGS để tìm giá trị cực tiểu của hàm mục tiêu và cập nhật trọng số và độ lệch của mạng neuron. Phương pháp L-BFGS là một thuật toán tối ưu không ràng buộc dựa trên gradient, nó sử dụng một ma trận xác định dương nghịch đảo để xấp xỉ ma trận Hessian của hàm mục tiêu. Quá trình lặp lại này được thực hiện cho đến khi số lần lặp vượt qua giới hạn np. Cuối cùng, thuật toán trả về quá trình của hàm mục tiêu J_α và nghiệm $\bar{u}$ của bài toán ngược Cauchy là kết quả cuối cùng.

2.4. Phân tích hội tụ cho xấp xỉ ANN

Trong phần này, chúng tôi sẽ trình bày về sự tương đương giữa bài toán (0.2) và bài toán tối ưu (2.11). Để chứng minh điều này, chúng ta cần định nghĩa khái niệm *trù mật (dense)* và *m-trù mật (m-dense)* trong miền $\Omega \times \mathcal{T}$.

2.4.1. Trù mật và m-trù mật của ANN

Theo tài liệu [28], các định nghĩa về *trù mật (denseness)* và *m-trù mật (m-denseness)* được định nghĩa như sau:

Định nghĩa 2.1. Một mạng $Net(\mathbf{x}, \mathbf{t}; \mathbf{w}, \mathbf{b})$ được gọi là *trù mật* nếu thỏa mãn điều

kiện sau:

$$\|Net(\mathbf{x},\mathbf{t};\mathbf{w},\mathbf{b}) - f(\mathbf{x},\mathbf{t})\|_1 \leq \epsilon, \quad \forall f \in C(\bar{\Omega}), \tag{2.21}$$

ở đây, $||\cdot||_1$ là chuẩn L_1.

Định nghĩa 2.2. Một mạng $Net(\mathbf{x},\mathbf{t};\mathbf{w},\mathbf{b})$ được gọi là *m-trù mật* nếu thỏa mãn điều kiện sau:

$$\max_{|\alpha|\leq m} \|\nabla^\alpha Net(\mathbf{x},\mathbf{t};\mathbf{w},\mathbf{b}) - \nabla^\alpha f(\mathbf{x},\mathbf{t})\|_1 \leq \epsilon. \tag{2.22}$$

ở đây, $||\cdot||_1$ là chuẩn L_1.

Giả sử mạng có l lớp ẩn có thể được xem như một ánh xạ $\mathcal{A}_n^l(\mathbf{x},\mathbf{t})$, trong đó $\mathbf{x}$ và $\mathbf{t}$ là đầu vào và đầu ra của mạng. Ánh xạ này được biểu diễn bằng công thức:

$$\mathcal{A}_n^l(\mathbf{x},\mathbf{t}) = \left\{ \xi(\mathbf{x};t) : \mathbb{R}^d \to \mathbb{R} \mid \xi(\mathbf{x};t) = \sum_{i=1}^{n} \xi_i \sigma\left(\mathbf{w}_i^l z^l(\mathbf{x};t) + b_i^l\right) \right\},$$

trong đó d là kích thước của đầu vào và n là số lượng neuron trong lớp thứ l. Có thể định nghĩa ngắn gọn là $\mathcal{A}^l(\mathbf{x},\mathbf{t}) = \cup_{n=1}^{\infty} \mathcal{A}_n^l(\mathbf{x},\mathbf{t})$.

2.4.2. Tính trù mật của $\mathcal{A}^l(\sigma)$

Xét tập Ω bị chặn trong không gian $\mathbb{R}^d$ với biên $\partial\Omega$. Trong [28], Kurt Hornik đã chứng minh tính *trù mật* và tính *m-trù mật* của mạng một lớp ẩn, và điều này sẽ được mở rộng sang mạng nhiều lớp ẩn trong định lý 2.1. Để thuận tiện cho chứng minh, ta xét một bổ đề quan trọng sau:

Bổ đề 2.1. *Định nghĩa hàm mạng neuron lớp ẩn thứ l_{th} $(l = 1, 2, \ldots, L+1)$ như sau:*

$$\mathbf{z}^l(\mathbf{x};t) = \mathbf{w}^l \sigma\left(\mathbf{z}^{l-1}(\mathbf{x};t)\right) + \mathbf{b}^l, \quad \mathbf{z}^l \in \mathbb{R}^{n_l}$$

với mọi $\epsilon > 0$, tồn tại $\mathbf{A}^l = \left[a_1^l, a_2^l, \ldots, a_d^l\right]^T \in \mathbb{R}^{d\times n_l}$ sao cho

$$\left\|\mathbf{A}^l \sigma\left(\mathbf{z}^l(\mathbf{x};t)\right) - (\mathbf{x},\mathbf{t})\right\|_{\bar{\Omega}\times\mathcal{T}} := \sup_{i|(\mathbf{x}_i,\mathbf{t}_i)\in\bar{\Omega}\times\mathcal{T}} \left|a_i^l \sigma\left(\mathbf{z}^l(\mathbf{x};t)\right) - (\mathbf{x}_i,\mathbf{t}_i)\right| < \epsilon. \tag{2.23}$$

Chứng minh. Sử dụng phương pháp quy nạp.

I. Kiểm tra phương trình (2.23) đúng khi $l = 1$.

Theo định lý 2 trong tài liệu [28], rõ ràng là với bất kỳ $\epsilon > 0$ và $x_i \in \Omega$, tồn tại $a_i^1 \in \mathbb{R}^{n_1}$ sao cho

$$\left\|\mathbf{A}^1 \sigma\left(\mathbf{z}^1(\mathbf{x};t)\right) - \mathbf{x}\right\|_{\bar{\Omega}} = \left|a_i^1 \sigma\left(\mathbf{z}^1(\mathbf{x};t)\right) - x_i\right| < \epsilon,$$

II. Giả sử phương trình (2.23) đúng với $l = k$, ta chứng minh nó đúng với $l = k + 1$.

Cố định $\mathbf{A}^{k+1}$, vì hàm sigmoid σ thỏa mãn tính liên tục Lipschitz nên ta có

$$\begin{aligned}
\sup_i \left| a_i^{k+1} \sigma\left(\mathbf{z}^{k+1}(\mathbf{x};t)\right) - x_i \right| &= \sup_i \left| a_i^{k+1} \sigma\left(\mathbf{w}^{k+1} \sigma\left(\mathbf{z}^{k}(\mathbf{x};t)\right) + \mathbf{b}^k\right) - x_i \right| \\
&= \sup_i \left| \sum_j a_{ij}^{k+1} \sigma\left(\mathbf{w}_j^{k+1} \sigma\left(\mathbf{z}^{k}(\mathbf{x};t)\right) + \mathbf{b}_j^k\right) - x_i \right| \\
&\leq \sup_i \left| \sum_j a_{ij}^{k+1} \left(\sigma\left(\mathbf{w}_j^{k+1} \sigma\left(\mathbf{z}^{k}(\mathbf{x};t)\right) + \mathbf{b}_j^k\right) - x_i\right) \right| \\
&+ \sup_i \left| \sum_j a_{ij}^{k+1} x_i - x_i \right| \\
&\leq \sup_i \left| \sum_j a_{ij}^{k+1} \epsilon \right| + \sup_i \left| \sum_j a_{ij}^{k+1} x_i - x_i \right| \\
&\leq \epsilon, \quad \text{với } \sum_j a_{ij}^{k+1} = 1,
\end{aligned}$$

hoàn thành chứng minh bổ đề 2.1. ■

Với bổ đề trên, Li và Hu [30] đã mở rộng định lý trong [45] thành mạng neuron nhiều lớp ẩn, xem trong định lý sau:

Định lý 2.1. *Cho σ là hàm kích hoạt sigmoid, mạng $\mathcal{A}^l(\sigma)$ là trù mật trong $C(\bar{\Omega} \times \mathcal{T})$.*

Chứng minh. Theo định lý 1 trong tài liệu [45], ta có:

$$\left\| A_1 \sigma\left(\mathbf{w}^1 \mathbf{x} + b^1\right) - f(\mathbf{x}) \right\| \leq \epsilon, \quad \forall f \in C(\bar{\Omega} \times \mathcal{T}). \tag{2.24}$$

Dễ thấy rằng:

$$\begin{aligned}
\|\mathcal{A}_d(\sigma) - f(\mathbf{x})\| &= \left\| A_{L+1} \sigma\left(\mathbf{z}^{L+1}(\mathbf{x};t)\right) - f(\mathbf{x}) \right\| \\
&\leq \left\| A_1 \sigma\left(\mathbf{w}^1 \mathbf{x} + \mathbf{b}^1\right) - f(\mathbf{x}) \right\| \\
&+ \left\| A_{L+1} \sigma\left(\mathbf{z}^{L+1}(\mathbf{x};t)\right) - A_1 \sigma\left(\mathbf{w}^1 \mathbf{x} + \mathbf{b}^1\right) \right\| \\
&\leq 2\epsilon, \quad \forall f \in C(\bar{\Omega} \times \mathcal{T})
\end{aligned}$$

hoàn thành chứng minh định lý 2.1. ■

2.4.3. Tính m - trù mật của $\mathcal{A}^l(\sigma)$

Đầu tiên, ta xem xét một bổ đề quan trọng sau:

Bổ đề 2.2. *Định nghĩa hàm mạng neuron lớp ẩn thứ l_{th} như sau:*

$$\mathbf{z}^l(\mathbf{x};t) = \mathbf{w}^l\sigma\left(\mathbf{z}^{l-1}(\mathbf{x};t)\right) + \mathbf{b}^l, \quad \mathbf{z}^l \in \mathbb{R}^{n_l}$$

với mọi $\epsilon > 0$, tồn tại ma trận $\mathbf{A}^l = \left[a_1^l, a_2^l, \ldots, a_d^l\right]^T \in \mathbb{R}^{d\times n^l}$, sao cho:

$$\max_{|\alpha|\le m} \sup_{\mathbf{x},\mathbf{t}\in\bar{\Omega}\times\mathcal{T}} \left|\nabla^\alpha \mathbf{A}^l\sigma\left(\mathbf{z}^l(\mathbf{x};t)\right) - \nabla^\alpha \mathbf{A}^1\sigma\left(\mathbf{w}^1(\mathbf{x},\mathbf{t}) + \mathbf{b}^1\right)\right| < \epsilon. \tag{2.25}$$

Chứng minh. Sử dụng phương pháp quy nạp để chứng minh bổ đề này.

I. Kiểm tra phương trình (2.25) đúng khi $l = 1$. Rõ ràng rằng với bất kỳ $\epsilon > 0$ và $x_i \in \Omega$, ta có:

$$\max_{|\alpha|\le m} \sup_{\mathbf{x}\in\bar{\Omega}} \left|\nabla^\alpha \mathbf{A}^l\sigma\left(\mathbf{z}^l(\mathbf{x};t)\right) - \nabla^\alpha \mathbf{A}^1\sigma\left(\mathbf{w}^1\mathbf{x} + \mathbf{b}^1\right)\right| = 0 < \epsilon, \tag{2.26}$$

và phương trình (2.25) được kiểm tra.

II. Giả sử phương trình (2.25) đúng với $l = k$, ta chứng minh nó đúng với $l = k + 1$.

Cố định $\mathbf{A}^{k+1}$, vì hàm sigmoid σ thỏa mãn tính chất liên tục Lipschitz và bị chặn, nên ta có:

$$\begin{aligned}
&\max_{|\alpha|\le m} \sup_{\mathbf{x}\in\bar{\Omega}} \left|\nabla^\alpha \mathbf{A}^{k+1}\sigma\left(\mathbf{z}^{k+1}(\mathbf{x};t)\right) - \nabla^\alpha \mathbf{A}^1\sigma\left(\mathbf{w}^1\mathbf{x} + \mathbf{b}^1\right)\right| \\
=&\max_{|\alpha|\le m} \sup_{\mathbf{x}\in\bar{\Omega}} \sup_i \left|\nabla^\alpha a_i^{k+1}\sigma\left(\mathbf{z}^l(\mathbf{x};t)\right) - \nabla^\alpha a_i^1\sigma\left(\mathbf{w}^1\mathbf{x} + \mathbf{b}^1\right)\right| \\
=&\max_{|\alpha|\le m} \sup_{\mathbf{x}\in\bar{\Omega}} \sup_i \left|\nabla^\alpha \left(\sum_j a_{ij}^{k+1}\sigma\left(\mathbf{w}_j^{k+1}\sigma\left(\mathbf{z}^k(\mathbf{x};t)\right) + \mathbf{b}_j^k\right) - \sum_j a_{ij}^1\sigma\left(\mathbf{w}_j^1\mathbf{x} + \mathbf{b}_j^1\right)\right)\right| \\
\le&\max_{|\alpha|\le m} \sup_{\mathbf{x}\in\bar{\Omega}} \sup_i \nabla^\alpha \left(\left|\sum_j a_{ij}^{k+1}\left(\sigma\left(\mathbf{w}_j^{k+1}\sigma\left(\mathbf{z}^k(\mathbf{x};t)\right) + \mathbf{b}_j^k\right) - \sigma\left(\mathbf{w}_j^1\mathbf{x} + \mathbf{b}_j^1\right)\right)\right|\right. \\
&\left. + \left|\sum_j \left(a_{ij}^{k+1} - a_{ij}^1\right)\sigma\left(\mathbf{w}_j^1\mathbf{x} + \mathbf{b}_j^1\right)\right|\right) \\
\le& L\epsilon \quad \text{với } \sum_j a_{ij}^{k+1} = \sum_j a_{ij}^1 = 1,
\end{aligned}$$

hoàn thành chứng minh của bổ đề 2.2. ■

Với bổ đề trên, Li và Hu [30] đã mở rộng định lý 3 trong tài liệu [45] cho mạng neuron đa lớp ẩn, từ đó thu được định lý sau:

Định lý 2.2. *Cho hàm kích hoạt sigmoid $\sigma \in C^m(\bar{\Omega}\times\mathcal{T})$, ta có $\mathcal{A}^l(\sigma)$ m-trù mật đều trên $C^m(\bar{\Omega}\times\mathcal{T})$.*

Chứng minh. Theo Định lý 3 trong tài liệu [45], ta có

$$\max_{|\alpha|\leq m} \left\| \nabla^\alpha A_1 \sigma \left(\mathbf{w}^1 \mathbf{x} + b^1 \right) - \nabla^\alpha f(\mathbf{x}) \right\| \leq \epsilon, \forall f \in C^m(\bar{\Omega}). \tag{2.27}$$

Rõ ràng là

$$\begin{aligned} &\max_{|\alpha|\leq m} \left\| \nabla^\alpha A_{L+1} \sigma \left(\mathbf{z}^{L+1}(\mathbf{x}; t) \right) - \nabla^\alpha f(\mathbf{x}) \right\| \\ &\leq \max_{|\alpha|\leq m} \left(\| \nabla^\alpha A_{L+1} \sigma \left(\mathbf{z}^{L+1}(\mathbf{x}; t) \right) - \nabla^\alpha A_1 \sigma \left(\mathbf{w}^1 \mathbf{x} + b^1 \right) \right) \| \\ &+ \left\| \nabla^\alpha A_1 \sigma \left(\mathbf{w}^1 \mathbf{x} + b^1 \right) - \nabla^\alpha f(\mathbf{x}) \right\| \Big) \\ &\leq (L+1)\epsilon, \forall f \in C^m(\bar{\Omega} \times \mathcal{T}). \end{aligned} \tag{2.28}$$

hoàn thành chứng minh định lý 2.2. ■

2.4.4. Sự tương đương giữa bài toán (0.2) và (2.11)

Điều kiện 1. Giả sử ban đầu bài toán (0.2) có các điều kiện sau đây:

a) Tồn tại một nghiệm duy nhất u^p trong (0.2), và $u^p \in H^2(\Omega \times \mathcal{T})$.

b) $\mathcal{L}$ liên tục Lipschitz trên $\bar{\Omega} \times \mathcal{T}$.

c) Hàm h thuộc lớp $C^2(\bar{\Omega} \times \mathcal{T})$ và có đạo hàm cấp một bị chặn trên $\bar{\Omega} \times \mathcal{T}$.

d) Biên $\partial\Omega \in C^2$.

Với những điều kiện này, sự tương đương được thiết lập trên các định lý sau:

Định lý 2.3. *Với mọi $\epsilon > 0$, tồn tại một chuỗi mạng neuron $\psi^n(\mathbf{x}; t; \mathbf{w}, \mathbf{b})$ sao cho $J(\psi^n) < K\epsilon$, trong đó $K = \max(|\Omega \times \mathcal{T}|, |\Gamma \times \mathcal{T}|, |\Omega|)$ và $J(\psi^n)$ là phương trình (2.10).*

Chứng minh. Bây giờ, chúng ta định nghĩa $\psi \in \mathcal{A}_d(\sigma)$ là một mạng neuron xấp xỉ. Giả sử rằng toán tử $\mathcal{L} \in C^m\left(\mathbb{R}^d\right)$ và hàm sigmoid $\sigma_l \in C^m\left(\mathbb{R}^d \times \mathcal{T}\right)$ không phải là hằng số và bị chặn trong khoảng $[0, 1]$. Rõ ràng rằng $\Omega \times \mathcal{T}$ là một tập con compact trong $\mathbb{R}^d$. Theo định lý 2.2, ta có

$$\max_{|\alpha\leq m|} \sup_{\mathbf{x}\in\Omega} |\nabla^\alpha u(\mathbf{x}; t) - \nabla^\alpha \psi| < \frac{\epsilon}{4}, \text{ với mọi } u \in C^m\left(\mathbb{R}^d \times \mathcal{T}\right), \tag{2.29}$$

do đó

$$\begin{aligned} &\sup_{\mathbf{x}\in\Gamma, t\in\mathcal{T}} |u - \psi| + \sup_{\mathbf{x}\in\Gamma, t\in\mathcal{T}} \left| \frac{\partial u}{\partial \mathbf{n}} - \frac{\partial \psi}{\partial \mathbf{n}} \right| \\ &+ \sup_{\mathbf{x}\in\Omega, t\in\mathcal{T}} \left| \frac{\partial u}{\partial t} + \mathcal{L}u - (\frac{\partial \psi}{\partial t} + \mathcal{L}\psi) \right| + \sup_{\mathbf{x}\in\Omega, t=0} |u - \psi| \\ &< 4 \sum_{\alpha=0}^{m} \sup_{\mathbf{x}\in\bar{\Omega}} |\nabla^\alpha u(\mathbf{x}; t) - \nabla^\alpha \psi| < \epsilon \end{aligned} \tag{2.30}$$

Gọi $u^p(\mathbf{x})$ là nghiệm của bài toán (0.2) và sử dụng kết luận từ phương trình (2.30) và bất đẳng thức Hölder, ta có

$$\begin{aligned}
J(\psi) &= \left\| \frac{\partial \psi}{\partial t} + \mathcal{L}\psi - (\frac{\partial u^p}{\partial t} + \mathcal{L}u^p) \right\|^2_{L^2(\Omega\times\mathcal{T})} \\
&+ \left\| \frac{\partial \psi}{\partial \mathbf{n}} - \frac{\partial u^p}{\partial \mathbf{n}} \right\|^2_{L^2(\Gamma\times\mathcal{T})} + \|\psi - u^p\|^2_{L^2(\Gamma\times\mathcal{T})} + \|\psi - u^p\|^2_{L^2(\Omega\times(t=0))} \\
&\leq |\Gamma\times\mathcal{T}| \sup_{\mathbf{x}\in\Gamma, t\in\mathcal{T}} |u^p - \psi|^2 + |\Omega| \sup_{\mathbf{x}\in\Omega, t=0} |u^p - \psi|^2 \\
&+ |\Gamma\times\mathcal{T}| \sup_{\mathbf{x}\in\Gamma, t\in\mathcal{T}} \left| \frac{\partial u^p}{\partial \mathbf{n}} - \frac{\partial \psi}{\partial \mathbf{n}} \right|^2 \\
&+ |\Omega\times\mathcal{T}| \sup_{\mathbf{x}\in\Omega, t\in\mathcal{T}} \left| \frac{\partial \psi}{\partial t} + \mathcal{L}\psi - (\frac{\partial u^p}{\partial t} + \mathcal{L}u^p) \right|^2 \\
&< K\epsilon^2 < K\epsilon, \text{ khi } \epsilon \to 0,
\end{aligned} \tag{2.31}$$

hoàn thành chứng minh định lý 2.3. ■

Định lý 2.4. *Chuỗi mạng neuron xấp xỉ $\{\psi^n\}$ hội tụ đến nghiệm u^* của (0.2) khi $n \to \infty$.*

Chứng minh. Vì ψ^n là một nghiệm của (2.11), rõ ràng rằng $J(\psi^n) \leq J(\psi)$ với mọi $\psi \in \mathcal{A}_d(\sigma)$. Đặt $0 \leq J(\psi^n) \leq J(\psi) \leq K\epsilon$. Khi $\epsilon \to 0$, tương đương với

$$\begin{aligned}
\left(\frac{\partial}{\partial t} + \mathcal{L}\right)\psi^n &= g^n \quad \text{trong } \Omega\times\mathcal{T}, \\
\psi^n - f &= g^n_f \quad \text{trên } \Gamma\times\mathcal{T}, \\
\frac{\partial \psi^n}{\partial \mathbf{n}} - g &= g^n_g \quad \text{trên } \Gamma\times\mathcal{T} \\
\psi^n(x;0) - h &= g^n_h \quad \text{trong } \Omega,
\end{aligned}$$

cho g^n, g^n_f, g^n_g, g^n_h thỏa mãn

$$\|g^n\|^2_{L^2(\Omega\times\mathcal{T})} + \left\|g^n_f\right\|^2_{L^2(\Gamma\times\mathcal{T})} + \left\|g^n_g\right\|^2_{L^2(\Gamma\times\mathcal{T})} + \|g^n_h\|^2_{L^2(\Omega\times(t=0))} \to 0, \text{ khi } n \to \infty \tag{2.32}$$

Giả sử điều kiện 1 thiết lập và $\psi^n \in L^2(\Omega\times\mathcal{T})$. Rõ ràng là $\{\psi^n\}$ bị chặn đều đối với n trong $L^\infty\left(\mathcal{T}, L^2(\Omega)\right)$, nghĩa là tồn tại một dãy con, được ký hiệu $\left\{\hat{\psi}^n\right\}$, hội tụ tới u theo nghĩa yếu trong $L^\infty\left(\mathcal{T}, L^2(\Omega)\ \right)$.

Tiếp theo theo điều kiện 1 và định lý 7.3 trong [24], rõ ràng là tồn tại hằng số $C < \infty$ sao cho

$$\int_{\bar{\Omega}\times\mathcal{T}} \left(\frac{\partial}{\partial t}\mathcal{L}\right)\hat{\psi}^n d\mathbf{x}dt < C,$$

dẫn đến $\{\bar{\psi}^n\}$ hội tụ hầu khắp nơi về u trong $\bar{\Omega} \times \mathcal{T}$. Sau đó, có thể chứng minh rằng ψ^n là nghiệm của bài toán (0.2) khi $n \to \infty$, hoàn thành chứng minh của định lý 2.4.

Cuối cùng, chúng ta đã chứng minh bài toán (0.2) tương đương với bài toán (2.11)(cũng như bài toán (0.1) và (2.4) đều thỏa mãn điều kiện 1. ■

2.5. Nhận xét

Trong chương này, chúng tôi đã trình bày các kết quả cụ thể như sau:

1. Trình bày mô hình mạng neuron nhân tạo (ANN) áp dụng chỉnh hóa Tikhonov cho cả bài toán Cauchy phụ thuộc thời gian và không phụ thuộc thời gian.

2. Áp dụng hiệu quả thuật toán ADAM kết hợp với thuật toán L-BFGS để đào tạo mạng neuron nhân tạo (ANN) giải bài toán Cauchy đặt không chỉnh cho phương trình elliptic và parabolic.

3. Trình bày sự hội tụ xấp xỉ mạng neuron.

Để minh chứng cho tính hiệu quả khi áp dụng phương pháp chỉnh hóa Tikhonov kết hợp với ANN trong giải bài toán Cauchy cho phương trình đạo hàm riêng, chúng tôi trình bày phần ví dụ mô phỏng ở chương 3.

Chương 3
KẾT QUẢ MÔ PHỎNG

Trong chương này, trình bày các ví dụ số mô phỏng quá trình huấn luyện mạng neuron nhân tạo (ANN) để giải bài toán Cauchy, bao gồm cả trường hợp tuyến tính và phi tuyến. Để tối ưu hóa các tham số của mạng, chúng tôi sử dụng kết hợp giữa thuật toán ADAM và thuật toán L-BFGS. Hàm kích hoạt sigmoid được chọn cho mỗi lớp ẩn trong ANN. Tham số chỉnh hóa α được chọn theo phương pháp L-curve và phương pháp hậu nghiệm. Các ví dụ được viết trên ngôn ngữ lập trình Python.

Các ví dụ trong luận văn này sử dụng dữ liệu số hóa và chúng tôi đã biết được giá trị chính xác của nghiệm, từ đó có thể tính được sai số L_2 để đánh giá hiệu suất của mạng ANN. Gọi u^* là giá trị chính xác của nghiệm và $\bar{u}$ là giá trị xấp xỉ của nghiệm, N là số lượng điểm dữ liệu. Độ chính xác của $\bar{u}$ được tính bằng công thức sau:

$$L_2 = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\bar{u}\left(x_i\right) - u^*\left(x_i\right)\right)^2}$$

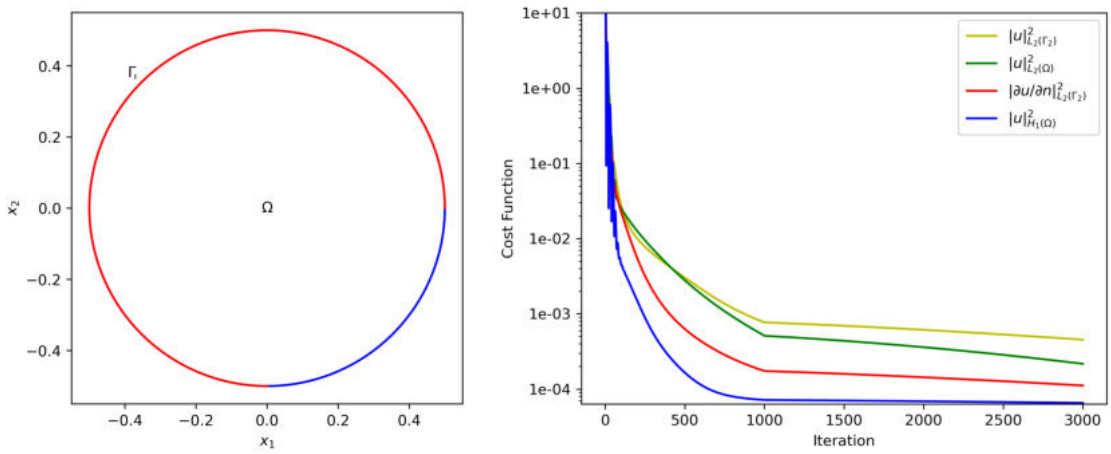
trong đó, $\{x_i\}_{i=1}^N$ là tập hợp các điểm trên miền Ω. Chúng tôi sử dụng một tập hợp các điểm được chọn ngẫu nhiên từ trong và trên Ω để tính sai số L_2.

3.1. Ví dụ cho bài toán tuyến tính 2D

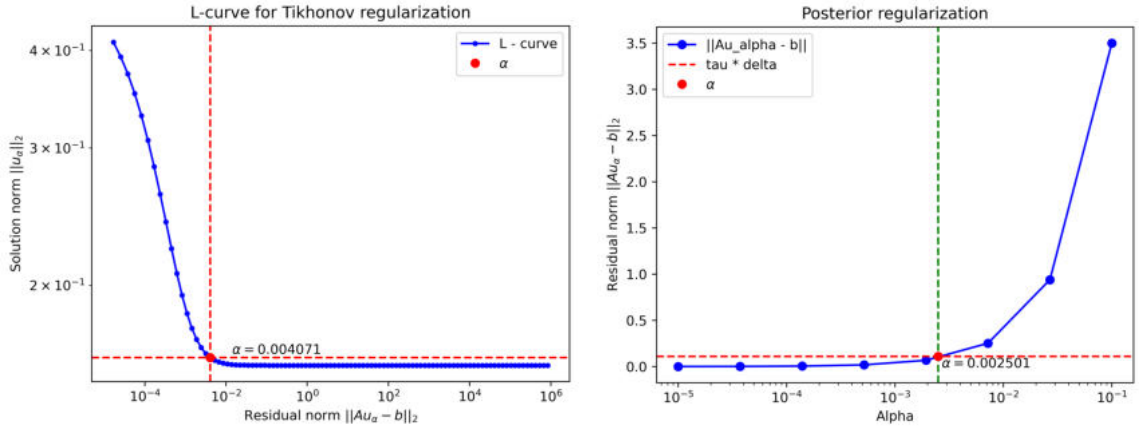
Ví dụ 3.1. Xét bài toán Cauchy cho phương trình phụ thuộc thời gian:

$$\begin{cases} \frac{\partial u(\mathbf{x};t)}{\partial t} + \Delta u(\mathbf{x};t) = 0 & \mathbf{x}, t \text{ trong } \Omega, \mathcal{T} \\ u(\mathbf{x};t) = \sin\left(x_1\right)\cos(x_2)e^{2t} & \mathbf{x}, t \text{ trên } \Gamma_1, \mathcal{T} \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = \left[\cos(x_1)\cos(x_2)e^{2t}, -\sin(x_1)\sin(x_2)e^{2t}\right] * \mathbf{n} & \mathbf{x}, t \text{ trên } \Gamma_1, \mathcal{T} \\ u(\mathbf{x};0) = \sin(x_1)\cos(x_2) & \mathbf{x} \text{ trong } \Omega \end{cases}$$

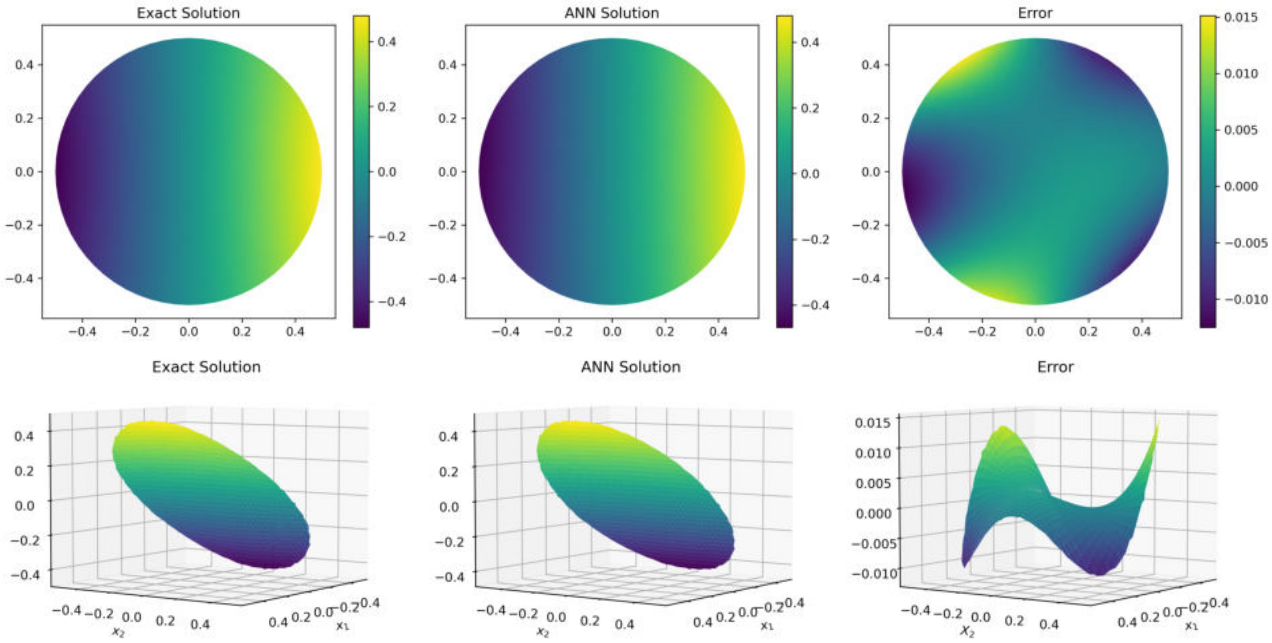
trong đó, Ω là một miền có biên liên tục $\partial\Omega$, Γ_1 là 3/4 biên đường tròn và $\mathcal{T} = \left[0, \frac{\pi}{2}\right]$. Các điểm dữ liệu trên miền Ω và biên Γ_1 được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3000$ điểm trên miền Ω và $N_b = 300$ điểm trên biên Γ_1.



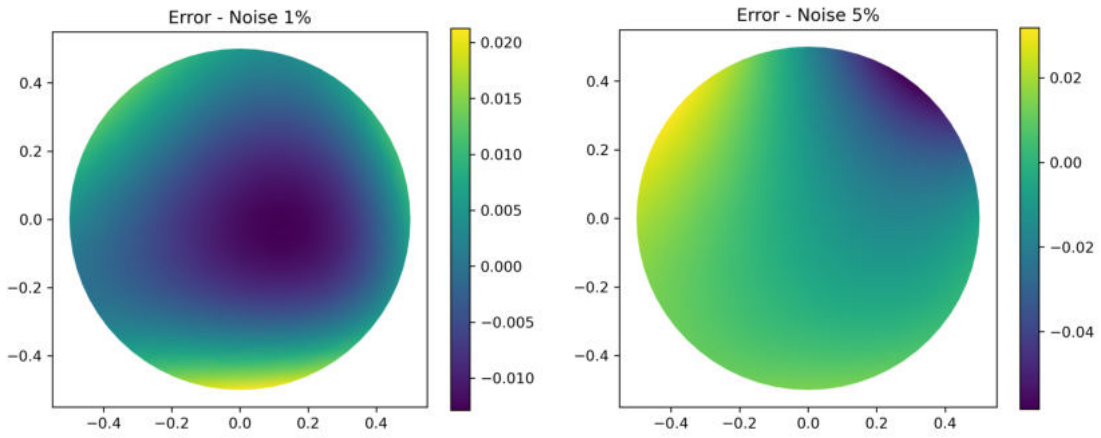
Hình 3.1: Miền dữ liệu Ω, Γ_1 và lịch sử hội tụ qua mỗi bước lặp của thuật toán ADAM.



Hình 3.2: Đồ thị minh họa phương pháp L-curve và phương hậu nghiệm.



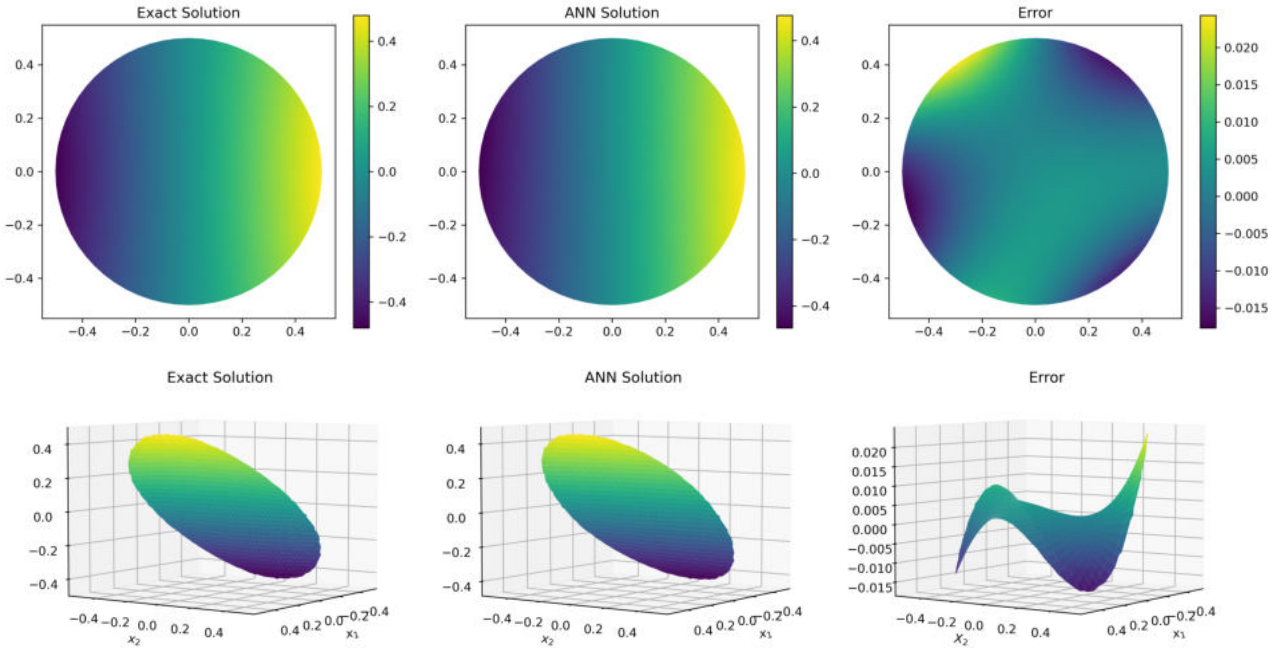
Hình 3.3: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số với $t = \frac{\pi}{5}$ trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.004071$ chọn theo phương pháp L-curve.



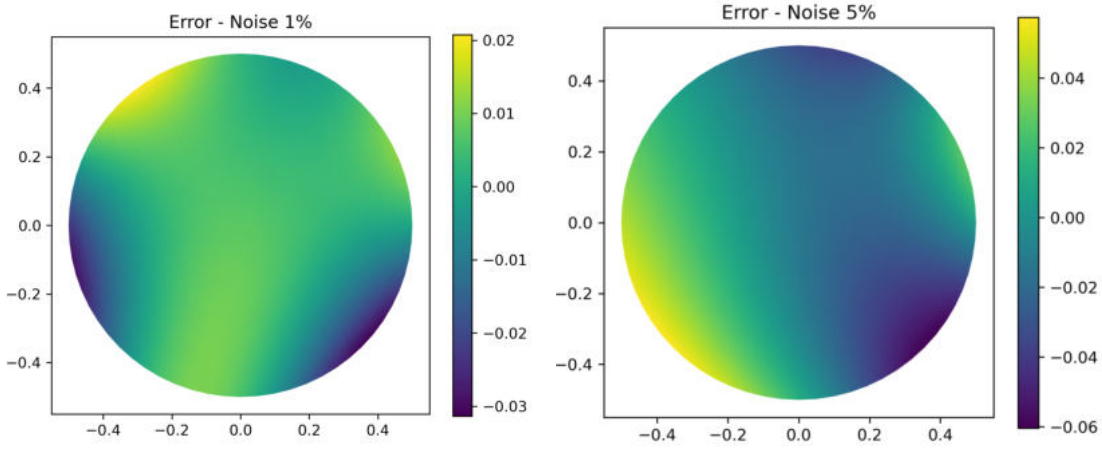
Hình 3.4: Sai số giữa nghiệm chính xác và nghiệm số với $t = \frac{\pi}{5}$ trong các trường hợp nhiễu $1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.004071$ chọn theo phương pháp L-curve.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	5.215928e-03	8.335032e-03	2.789440e-02

Bảng 3.1: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$ với $t = \frac{\pi}{5}$. Tham số chỉnh hóa $\alpha = 0.004071$ chọn theo phương pháp L-curve.



Hình 3.5: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số với $t = \frac{\pi}{5}$ trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.002501$ chọn theo phương pháp hậu nghiệm.



Hình 3.6: Sai số giữa nghiệm chính xác và nghiệm số với $t = \frac{\pi}{5}$ trong các trường hợp nhiễu $1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.002501	0.006152	0.013729
Sai số - L^2	6.173957e-03	9.525805e-03	3.563391e-02

Bảng 3.2: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$ với $t = \frac{\pi}{5}$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

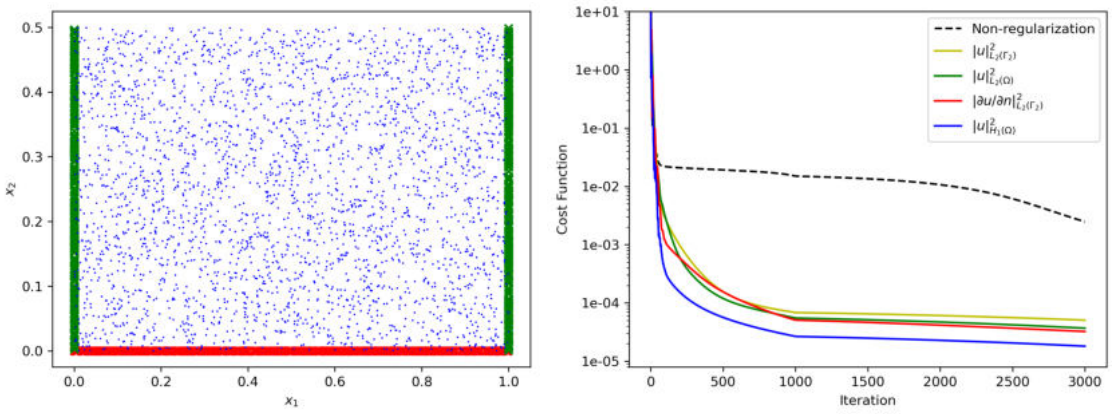
Ví dụ 3.2. Cho $\Omega \subset \mathbb{R}^2$ là hình chữ nhật mở $(0,1) \times (0,0.5)$ và định nghĩa các tập con sau của $\partial\Omega$:

$$\Gamma_1 := \{(x_1, 0); x_1 \in (0,1)\}, \qquad \Gamma_2 := \{(x_1, 0.5); x_1 \in (0,1)\},$$
$$\Gamma_3 := \{(0, x_2); x_2 \in (0,0.5)\}, \quad \Gamma_4 := \{(0.5, x_2); x_2 \in (0,0.5)\}.$$

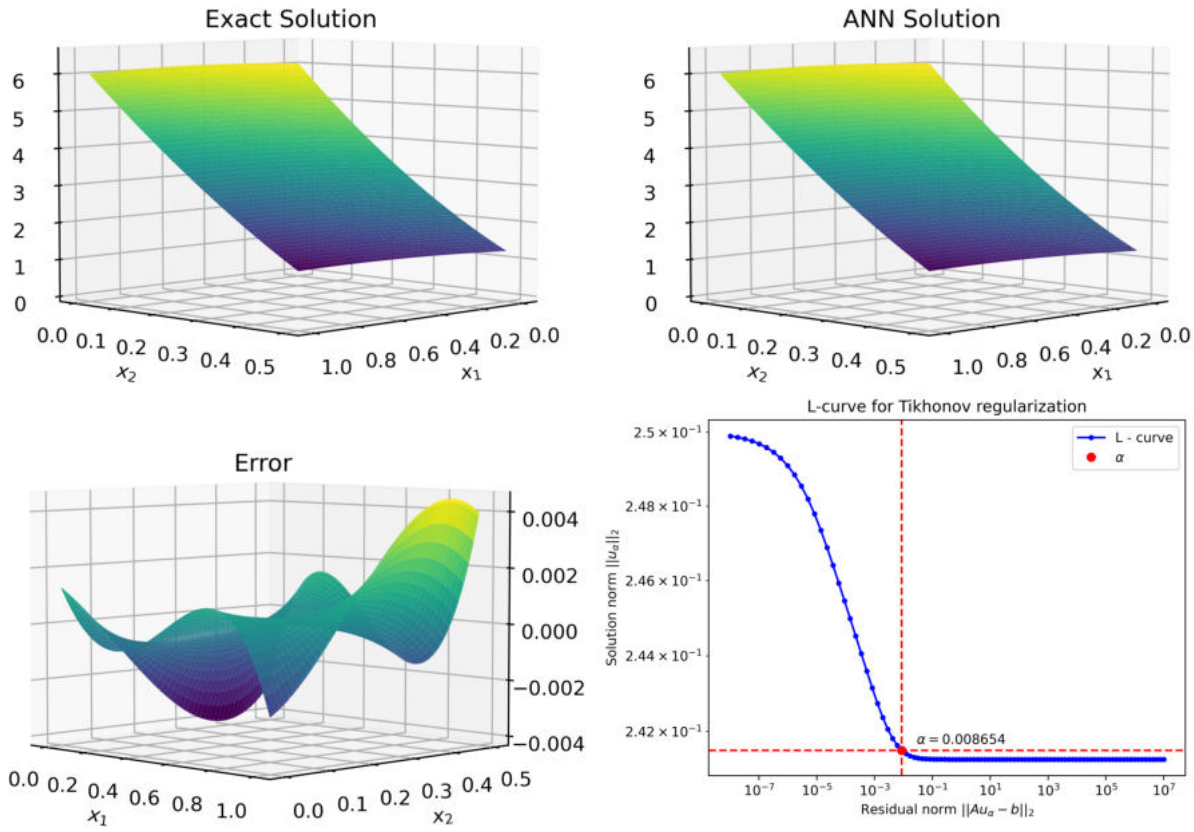
Xét bài toán Cauchy sau

$$\begin{cases} \Delta u(\mathbf{x}) = 0 & \mathbf{x} \text{ trong } \Omega \\ u(\mathbf{x}) = x_1^2 + 5x_1 & \mathbf{x} \text{ trên } \Gamma_1 \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = \nabla(x_1^2 - x_2^2 + 5x_1 + 2x_2 - 3x_1x_2 + e^{x_1}\sin x_2) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1 \\ u(\mathbf{x}) = x_1^2 - x_2^2 + 5x_1 + 2x_2 - 3x_1x_2 + e^{x_1}\sin x_2 & \mathbf{x} \text{ trên } \Gamma_3 \cup \Gamma_4 \end{cases}$$

Trong ví dụ này các điểm dữ liệu trên miền Ω và biên $\Gamma_1, \Gamma_3 \cup \Gamma_4$ được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3500$ điểm trên miền Ω và $N_b = 350$ điểm trên biên $\Gamma_1, \Gamma_3 \cup \Gamma_4$.



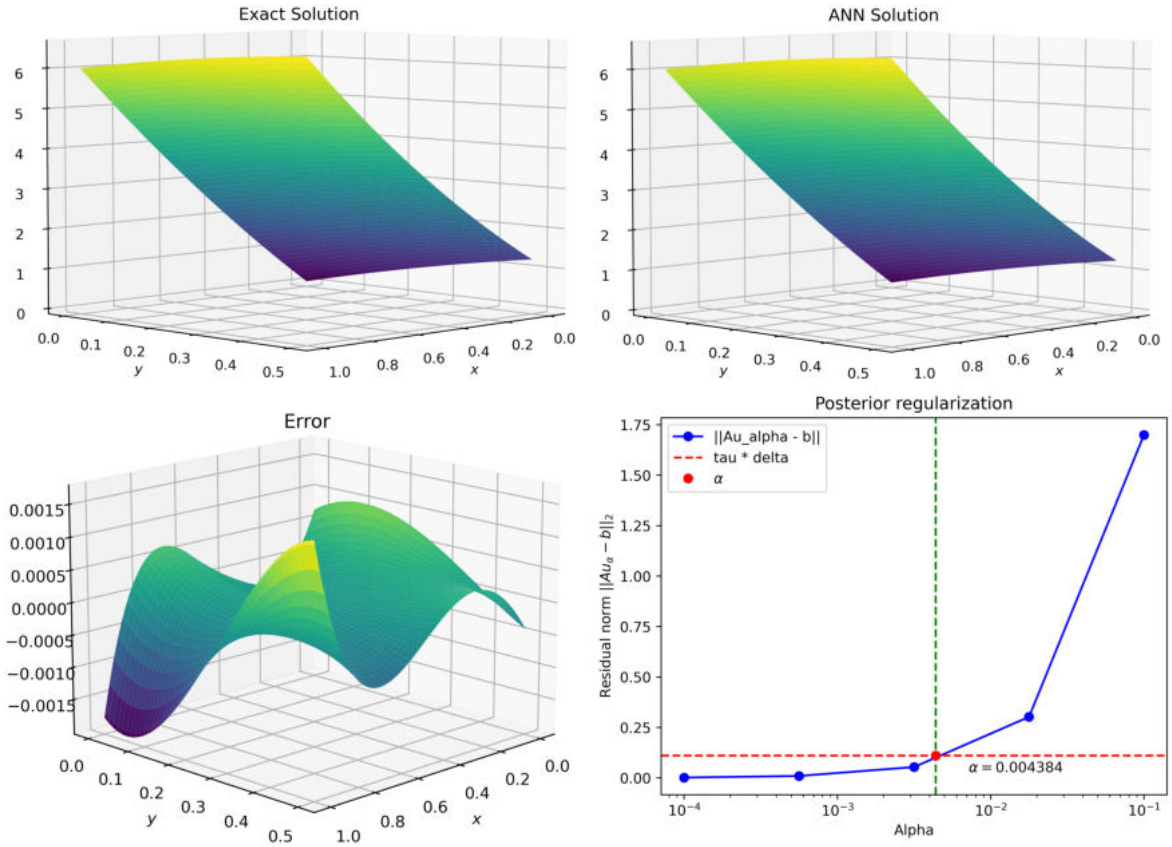
Hình 3.7: Miền dữ liệu $\Omega, \Gamma_1, \Gamma_3 \cup \Gamma_4$ và lịch sử hội tụ của thuật toán ADAM.



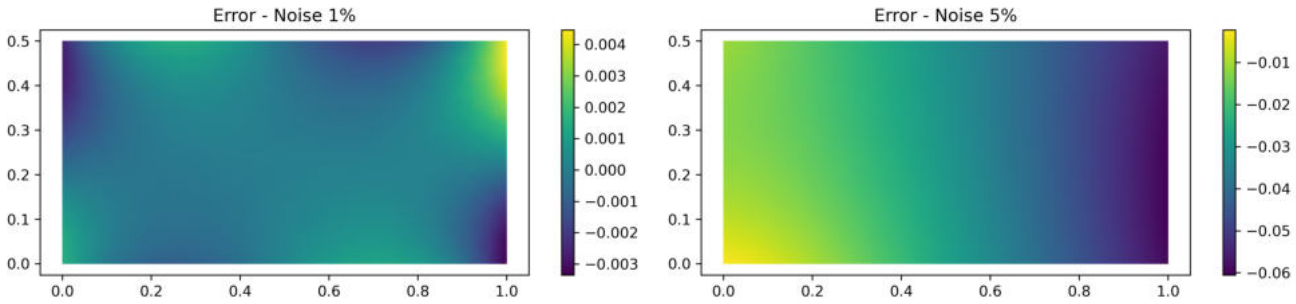
Hình 3.8: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.008654$ chọn theo phương pháp L-curve.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	1.117616e-03	2.015243e-03	2.823571e-02

Bảng 3.3: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.008654$ chọn theo phương pháp L-curve.



Hình 3.9: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.004384$ chọn theo phương pháp hậu nghiệm.



Hình 3.10: Nghiệm số và sai số trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.0043841	0.007323	0.0132282
Sai số - L^2	4.413840e-04	1.005104e-03	3.471376e-02

Bảng 3.4: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

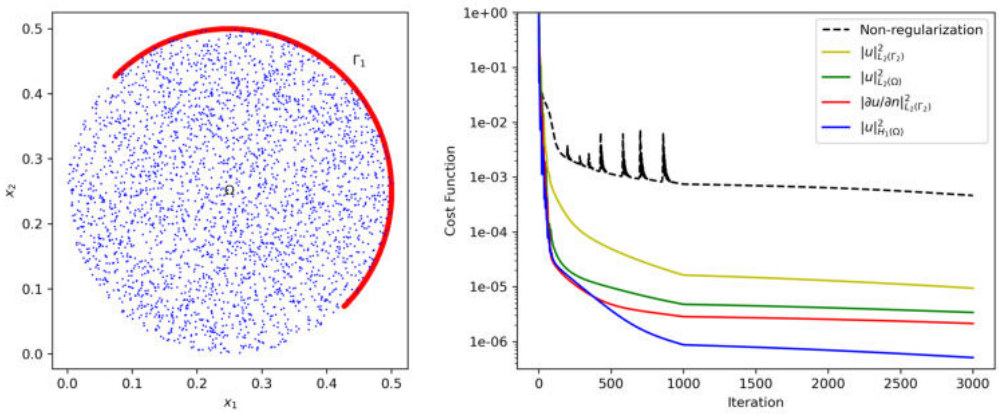
ADAM	Không chỉnh	$\left\|\bar{u}\right\|^2_{L_2(\Gamma 2)}$	$\left\|\bar{u}\right\|^2_{L_2(\Omega)}$	$\left\|\frac{\partial \bar{u}}{\partial n}\right\|^2_{L_2(\Gamma 2)}$	$\left\|\bar{u}\right\|^2_{H_1(\Omega)}$
L	3	3	3	3	3
N	7	7	7	7	7
Nhiễu (%)	0.1%	0.1%	0.1%	0.1%	0.1%
α	0	0.008654	0.008654	0.008654	0.008654
Bước lặp	3000	3000	3000	3000	3000
Thời gian/Bước lặp	0.00613s	0.00641s	0.00740s	0.00620s	0.00831s
Hàm mất mát (Loss)	2.48e-03	5.06e-05	3.68e-05	3.22e-05	1.81e-05

Bảng 3.5: Kết quả số cho bài toán bằng thuật toán ADAM.

Ví dụ 3.3. Xét bài toán Cauchy cho phương trình không phụ thuộc thời gian:

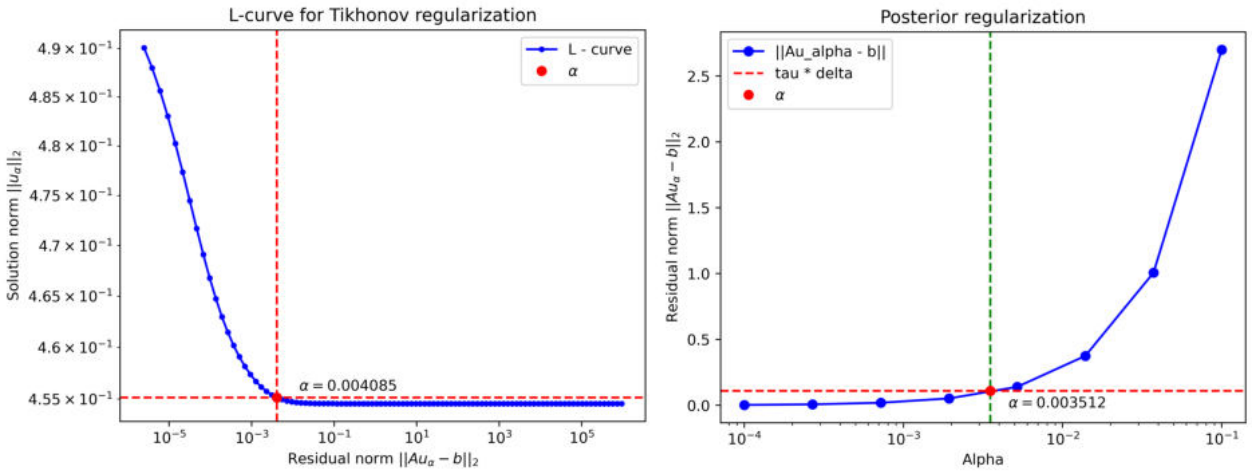
$$\begin{cases} \Delta u(\mathbf{x}) = 0 & \mathbf{x} \text{ trong } \Omega, \\ u(\mathbf{x}) = x_1 e^{-x_1}\sin(x_2) - x_2 e^{-x_1}\cos(x_2) & \mathbf{x} \text{ trên } \Gamma_1, \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = \nabla(x_1 e^{-x_1}\sin(x_2) - x_2 e^{-x_1}\cos(x_2)) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1, \end{cases}$$

trong đó Ω là miền và Γ_1 là 1/2 biên của đường tròn như Hình 1.

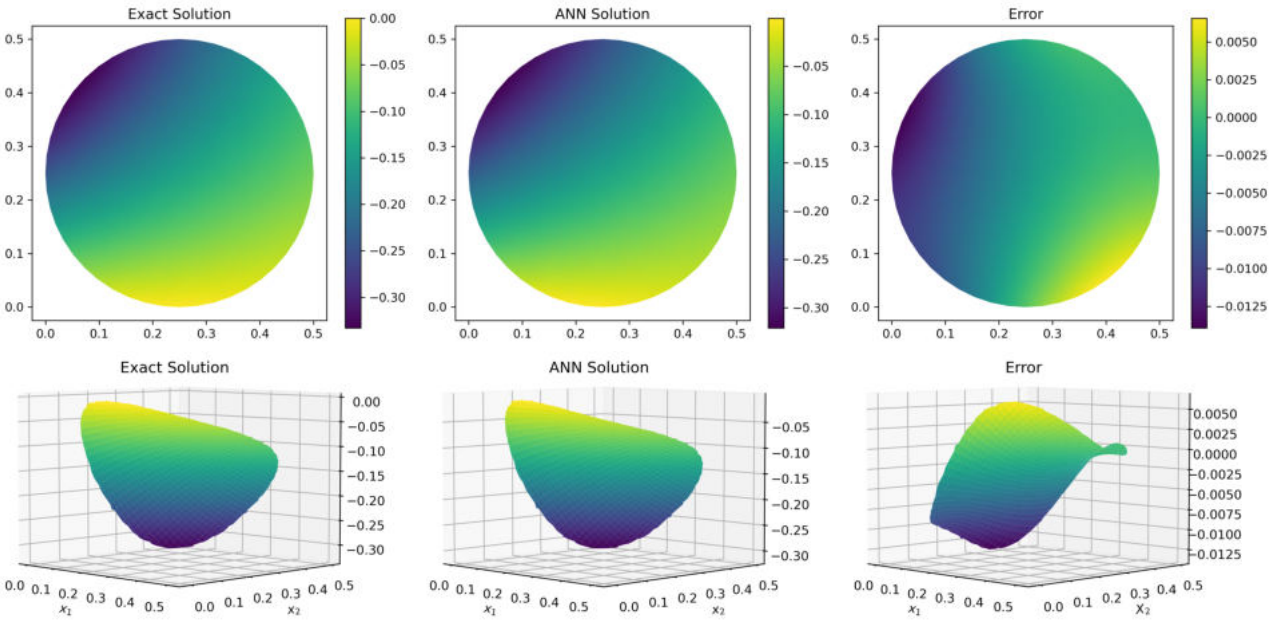


Hình 3.11: Miền dữ liệu Ω, Γ_1 và lịch sử hội tụ qua mỗi bước lặp của thuật toán ADAM.

Trong ví dụ này các điểm dữ liệu trên miền Ω và biên Γ_1 được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3000$ điểm trên miền Ω và $N_b = 300$ điểm trên biên Γ_1.



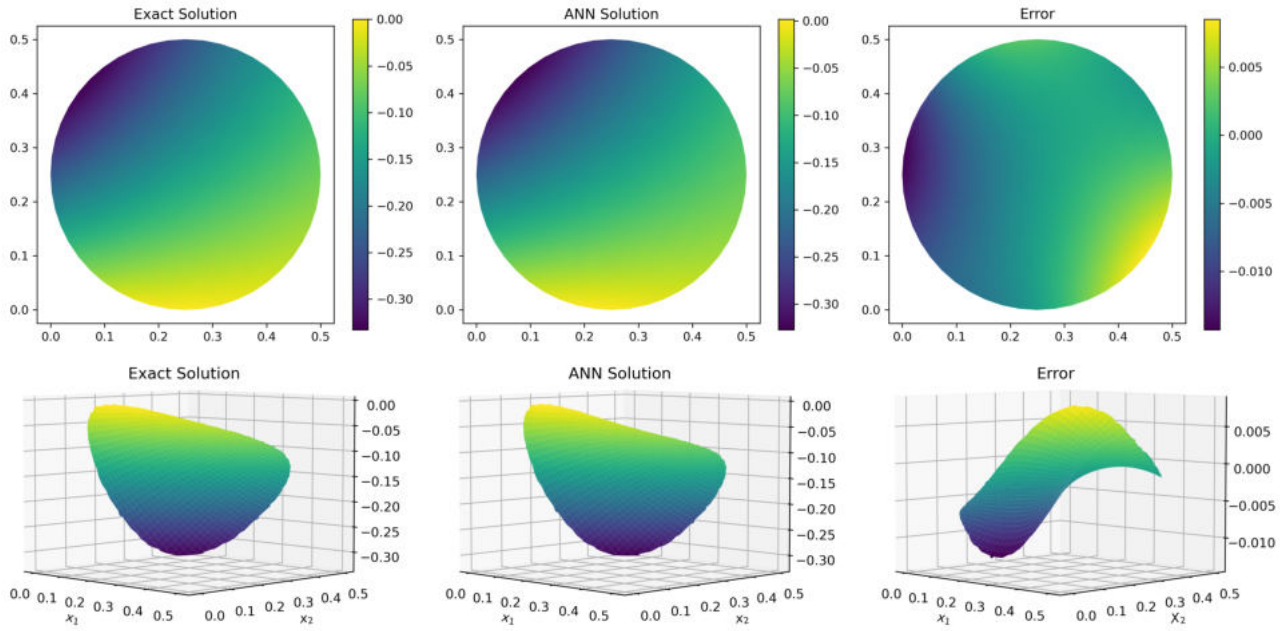
Hình 3.12: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



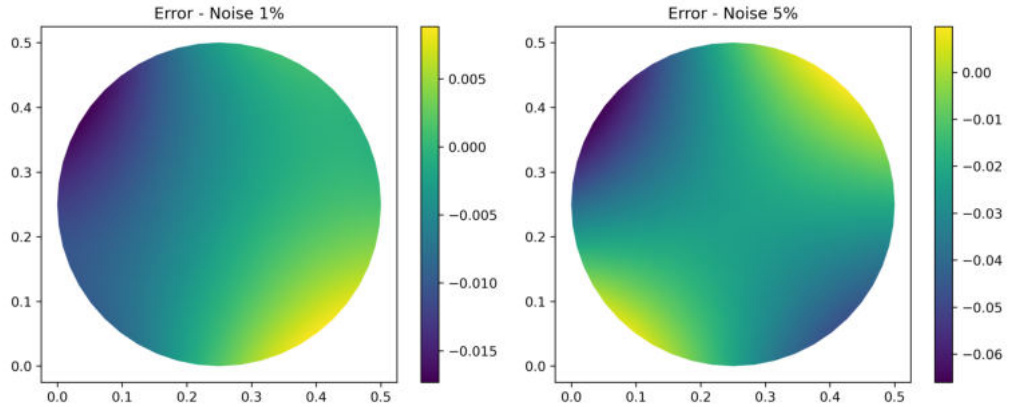
Hình 3.13: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.004085$ chọn theo phương pháp L-curve.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	4.750695e-03	8.671825e-03	2.823641e-02

Bảng 3.6: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.004085$ chọn theo phương pháp L-curve.



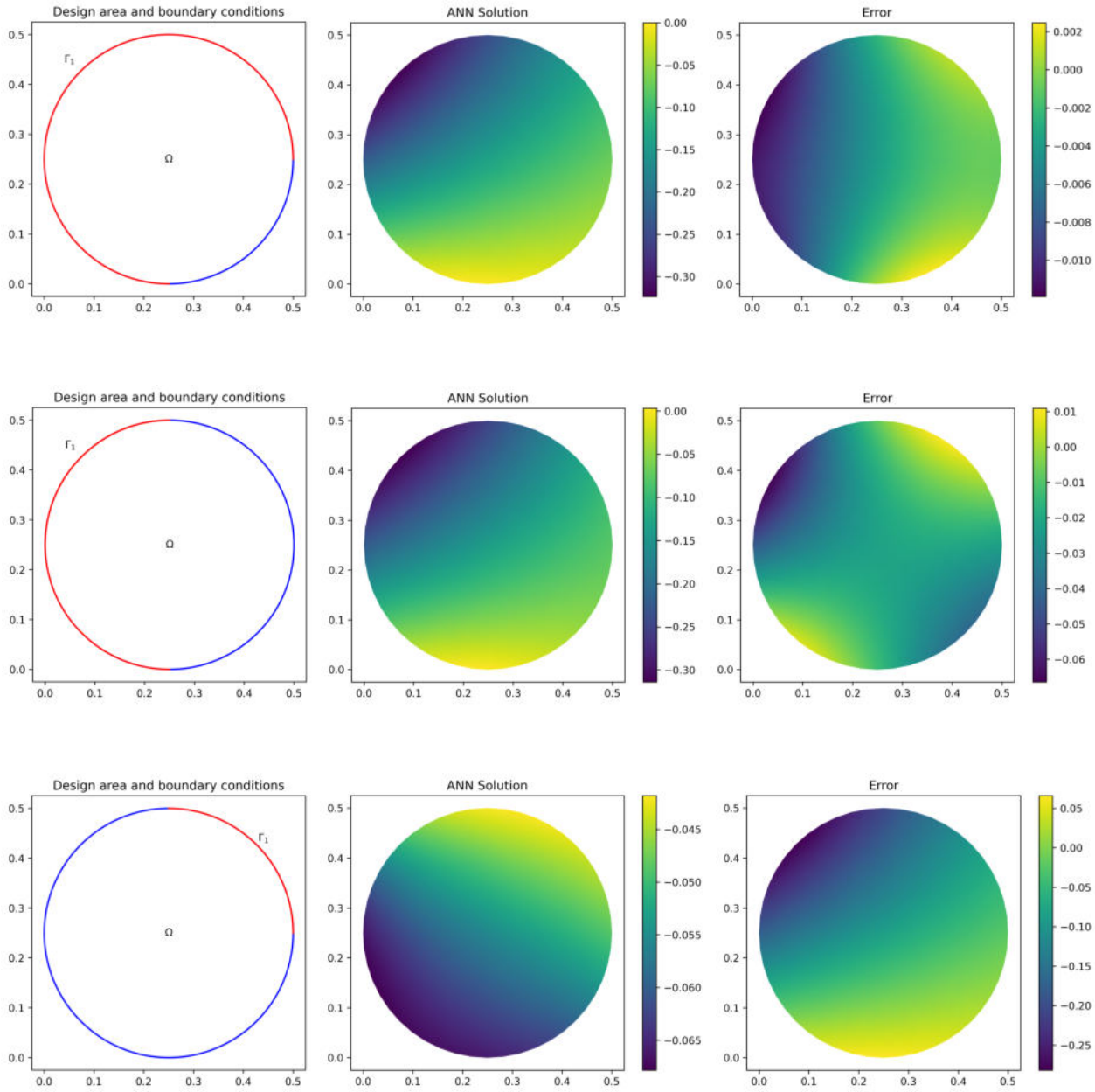
Hình 3.14: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.003512$ chọn theo phương pháp hậu nghiệm.



Hình 3.15: Sai số giữa nghiệm chính xác và nghiệm số trong các trường hợp nhiễu $1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.0035121	0.0057246	0.0212954
Sai số - L^2	5.163842e-03	6.849026e-03	2.853535e-02

Bảng 3.7: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.



Hình 3.16: So sánh các kết quả khi ta thay đổi dữ liệu Cauchy trên biên Γ_1 lần lược 3/4, 1/2, 1/4 cho bài toán. Tham số chỉnh hóa $\alpha = 0.004085$ chọn theo phương pháp L-curve.

Trường hợp	Γ_1 là 3/4 biên	Γ_1 là 1/2 biên	Γ_1 là 1/4 biên
Sai số - L^2	5.518368e-03	2.684806e-02	1.225842e-01

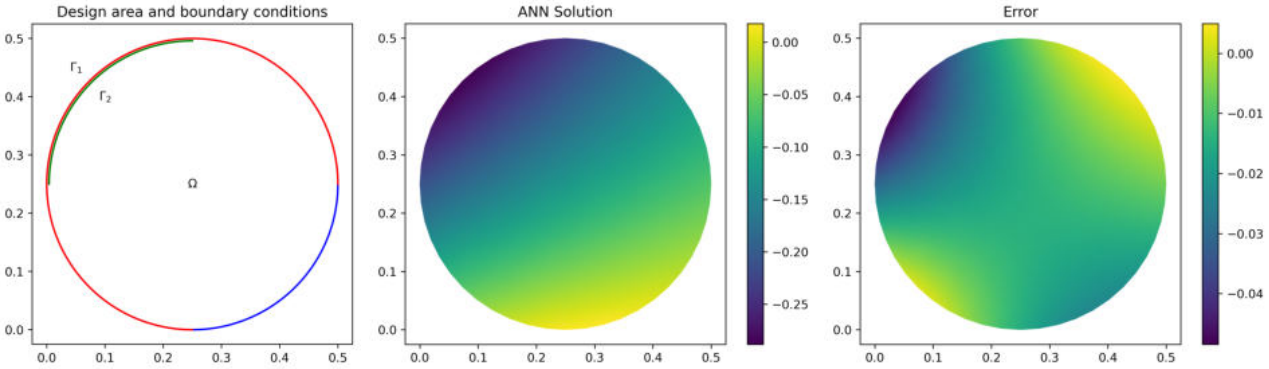
Bảng 3.8: Sai số L^2 trong các trường hợp thay đổi các điều kiện Cauchy với nhiễu 1%. Tham số chỉnh hóa $\alpha = 0.004085$ chọn theo phương pháp L-curve.

Bảng số liệu trên thể hiện kết quả sai số L^2 trong các trường hợp thay đổi các điều kiện Cauchy với mức nhiễu (noise) bằng 1%. Ở trường hợp Γ_1 là 3/4 biên, sai số L^2

là 5.518368e-03. Trong khi đó, sai số tăng lên rất nhiều lần ở trường hợp Γ_1 là 1/4 biên với giá trị là 1.225842e-01. Điều này cho thấy độ nhạy cảm của phương pháp khi thay đổi các điều kiện Cauchy và khi giảm xuống 1/4 trên biên thì bài toán không thể giải được.

Bây giờ, ta điều chỉnh điều kiện Dirichlet(Γ_1) là 3/4 biên, điều kiện Neumann(Γ_2) là 1/4 của biên và $\Gamma_2 \subset \Gamma_1$. Cụ thể bài toán được viết lại như sau

$$\begin{cases} \Delta u(\mathbf{x}) = 0 & \mathbf{x} \text{ trong } \Omega, \\ u(\mathbf{x}) = x_1 e^{-x_1}\sin(x_2) - x_2 e^{-x_1}\cos(x_2) & \mathbf{x} \text{ trên } \Gamma_1, \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = \nabla(x_1 e^{-x_1}\sin(x_2) - x_2 e^{-x_1}\cos(x_2)) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_2, \end{cases}$$



Hình 3.17: Kết quả khi ta thay đổi dữ liệu Cauchy cho bài toán.

Trường hợp	Nhiễu	Sai số - L^2
Điều kiện biên Dirichlet (Γ_1) là 3/4, Điều kiện biên Neumann (Γ_2) là 1/4 và $\Gamma_2 \subset \Gamma_1$	1%	1.865966e-02

Bảng 3.9: Sai số L^2 trong các trường hợp thay đổi điều kiện Cauchy. Tham số chỉnh hóa $\alpha = 0.004085$ chọn theo phương pháp L-curve.

3.2. Ví dụ cho bài toán tuyến tính 3D

Ví dụ 3.4. Cho $\Omega \subset \mathbb{R}^3$ là hình khối 3D gồm tất cả các điểm (x_1, x_2, x_3) trong đó $0 \le x_1 \le 1$, $0 \le x_2 \le 1$, $0 \le x_3 \le 1$ và định nghĩa các tập con của $\partial\Omega$:

$$\Gamma_1 := \{x | 0 \le x_1 \le 1, 0 \le x_2 \le 1, x_3 = 0\}, \quad \Gamma_2 := \{x | 0 \le x_1 \le 1, 0 \le x_2 \le 1, x_3 = 1\},$$

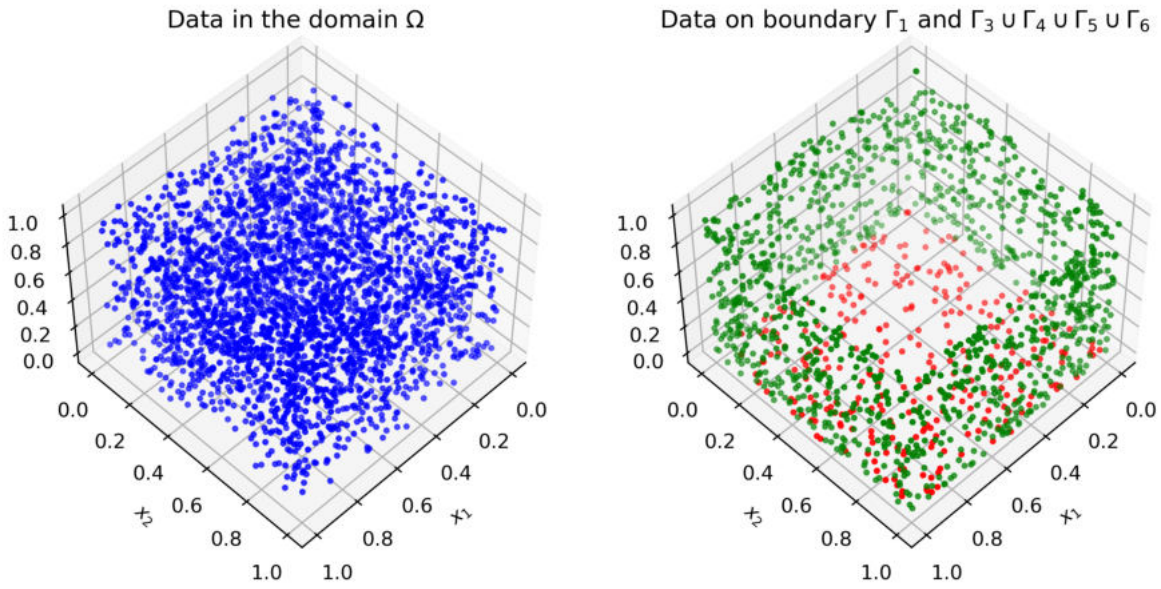
$$\Gamma_3 := \{x | x_1 = 0, 0 \le x_2 \le 1, 0 \le x_3 \le 1\}, \quad \Gamma_4 := \{x | x_1 = 1, 0 \le x_2 \le 1, 0 \le x_3 \le 1\},$$

$$\Gamma_5 := \{x | 0 \le x_1 \le 1, x_2 = 0, 0 \le x_3 \le 1\}, \quad \Gamma_6 := \{x | 0 \le x_1 \le 1, x_2 = 0, 0 \le x_3 \le 1\}.$$

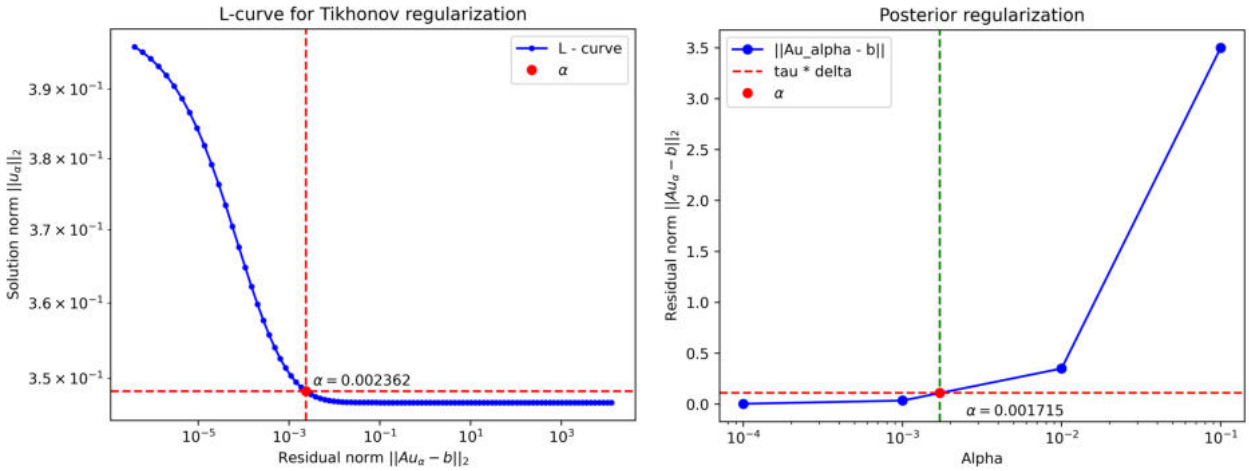
Xét bài toán Cauchy sau

$$\begin{cases} \Delta u(\mathbf{x}) = 0 & \mathbf{x} \text{ trong } \Omega \\ u(\mathbf{x}) = 0 & \mathbf{x} \text{ trên } \Gamma_1 \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = \nabla(\sinh(\sqrt{2}x_1)\sin x_2 \sin x_3) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1 \\ u(\mathbf{x}) = \sinh(\sqrt{2}x_1)\sin x_2 \sin x_3 & \mathbf{x} \text{ trên } \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6 \end{cases}$$

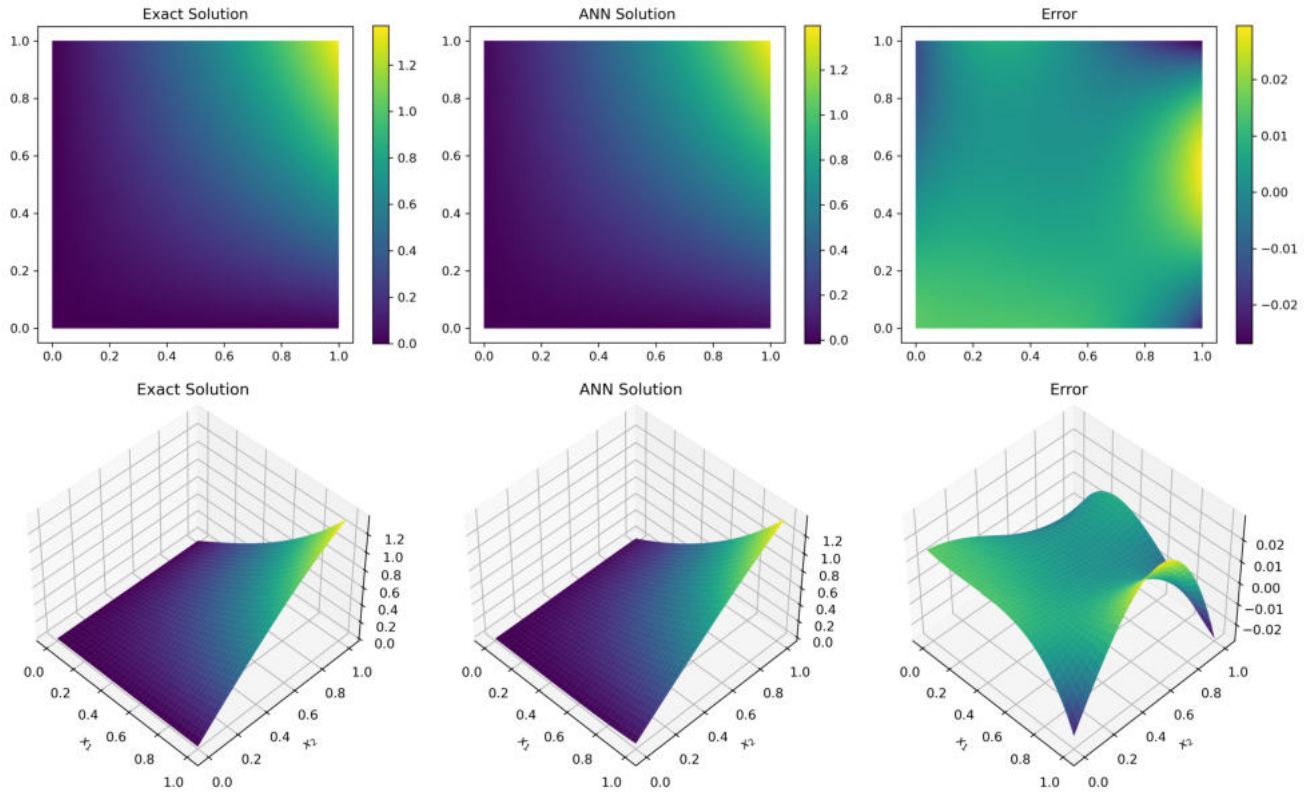
Trong ví dụ này các điểm dữ liệu trên miền Ω và biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$ được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3500$ điểm trên miền Ω và $N_b = 1000$ điểm trên biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$. Chạy bằng thuật toán L-BFGS.



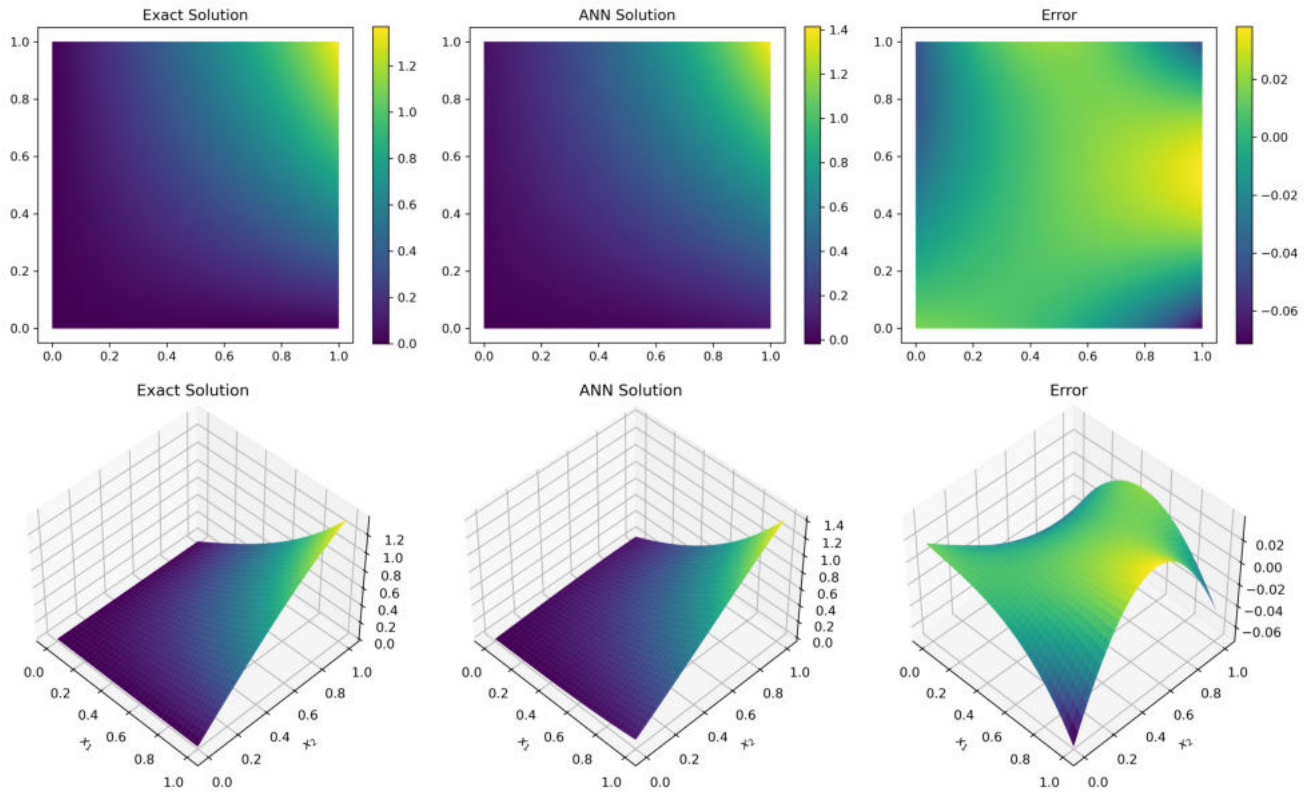
Hình 3.18: Miền dữ liệu Ω(màu xanh), Γ_1(màu đỏ) và $\Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$(màu xanh lá).



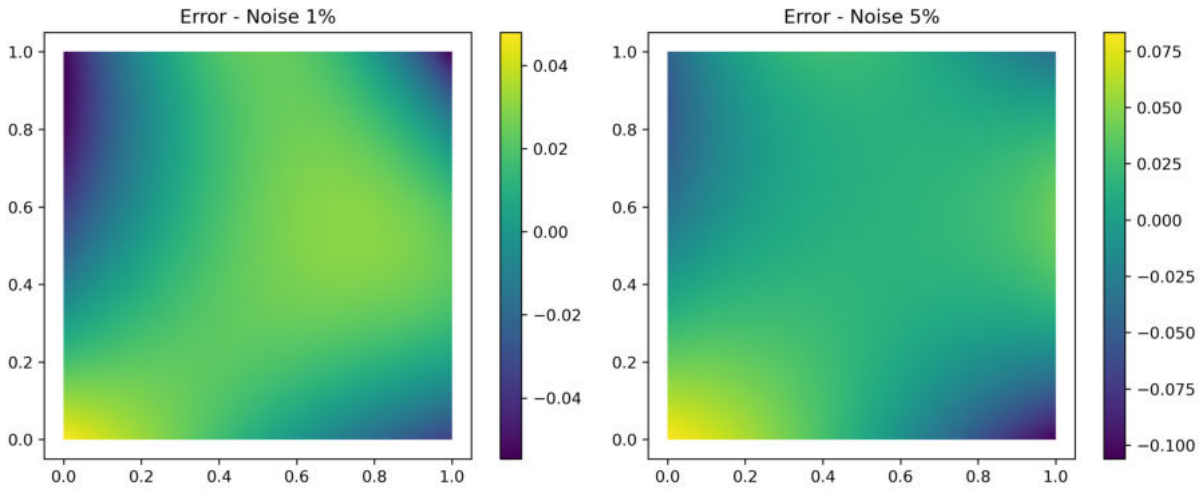
Hình 3.19: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



Hình 3.20: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.002362$ chọn theo phương pháp L-curve.



Hình 3.21: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.001715$ chọn theo phương pháp hậu nghiệm.



Hình 3.22: Sai số giữa nghiệm chính xác và nghiệm số của phương pháp ANN trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.001715	0.006371	0.023163
Sai số - L^2	9.345354e-03	2.199138e-02	3.014483e-02

Bảng 3.10: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	8.186427e-03	2.414273e-03	3.725438e-02

Bảng 3.11: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.002362$ chọn theo phương pháp L-curve.

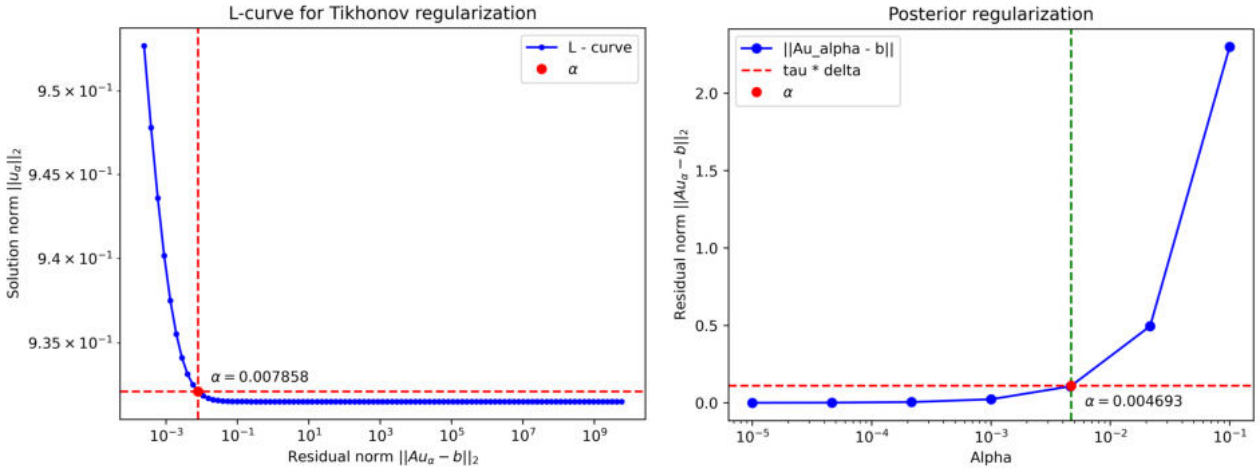
Ví dụ 3.5. Cho $\Omega \subset \mathbb{R}^3$ là hình khối 3D gồm tất cả các điểm (x_1, x_2, x_3) trong đó $0 \leq x_1 \leq 0.5$, $0 \leq x_2 \leq 1$, $0 \leq x_3 \leq 1$ và định nghĩa các tập con của $\partial\Omega$:

$$\Gamma_1 := \{x | 0 \leq x_1 \leq 0.5, 0 \leq x_2 \leq 1, x_3 = 0\}, \quad \Gamma_2 := \{x | 0 \leq x_1 \leq 0.5, 0 \leq x_2 \leq 1, x_3 = 1\},$$
$$\Gamma_3 := \{x | x_1 = 0, 0 \leq x_2 \leq 1, 0 \leq x_3 \leq 1\}, \quad \Gamma_4 := \{x | x_1 = 0.5, 0 \leq x_2 \leq 1, 0 \leq x_3 \leq 1\},$$
$$\Gamma_5 := \{x | 0 \leq x_1 \leq 0.5, x_2 = 0, 0 \leq x_3 \leq 1\}, \quad \Gamma_6 := \{x | 0 \leq x_1 \leq 0.5, x_2 = 1, 0 \leq x_3 \leq 1\}.$$

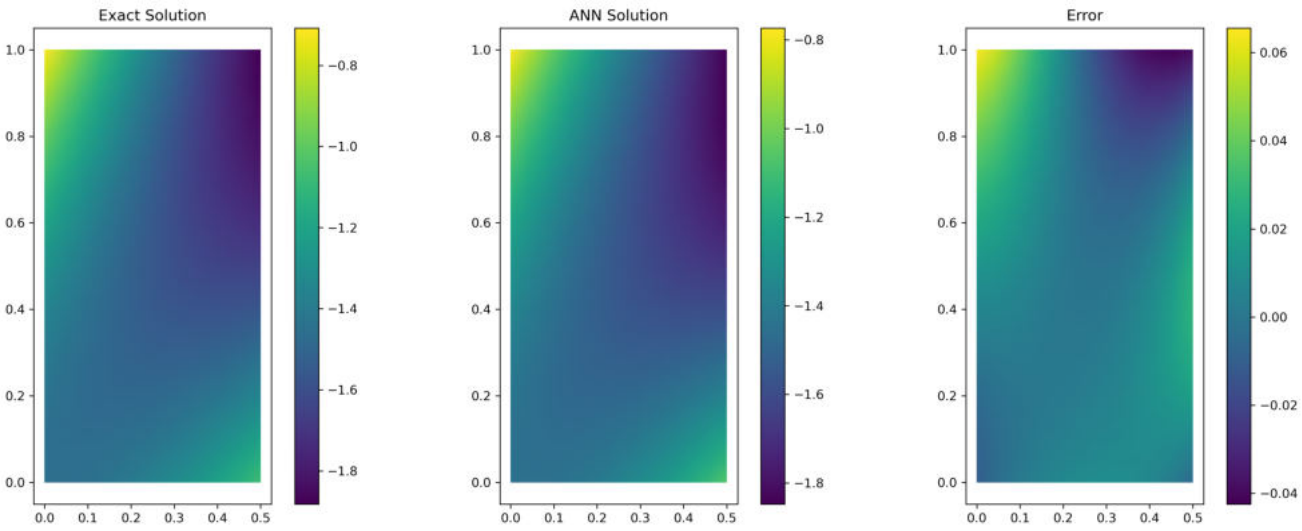
Xét bài toán Cauchy sau

$$\begin{cases} \Delta u(\mathbf{x}) = 0 & \mathbf{x} \text{ trong } \Omega \\ u(\mathbf{x}) = x_1^2 + x_2^2 + \cosh(\sqrt{2}x_1)\cos x_2 & \mathbf{x} \text{ trên } \Gamma_1 \\ \frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} = \nabla(x_1^2 + x_2^2 - 2x_3^2 - 3x_1x_2x_3 + \cosh(\sqrt{2}x_1)\cos x_2 \cos x_3) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1 \\ u(\mathbf{x}) = x_1^2 + x_2^2 - 2x_3^2 - 3x_1x_2x_3 + \cosh(\sqrt{2}x_1)\cos x_2 \cos x_3 & \mathbf{x} \text{ trên } \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6 \end{cases}$$

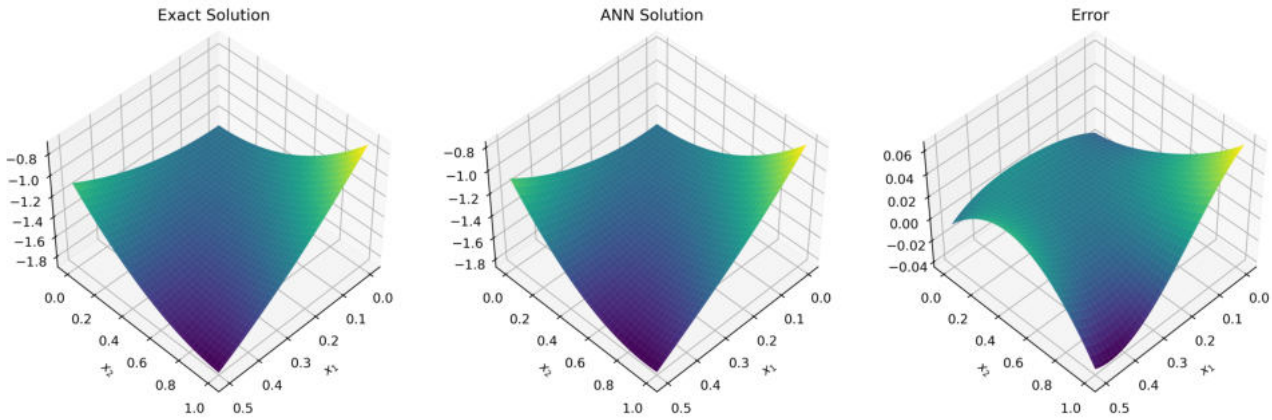
Trong ví dụ này các điểm dữ liệu trên miền Ω và biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$ được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3500$ điểm trên miền Ω và $N_b = 1000$ điểm trên biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$. Chạy bằng thuật toán L-BFGS.



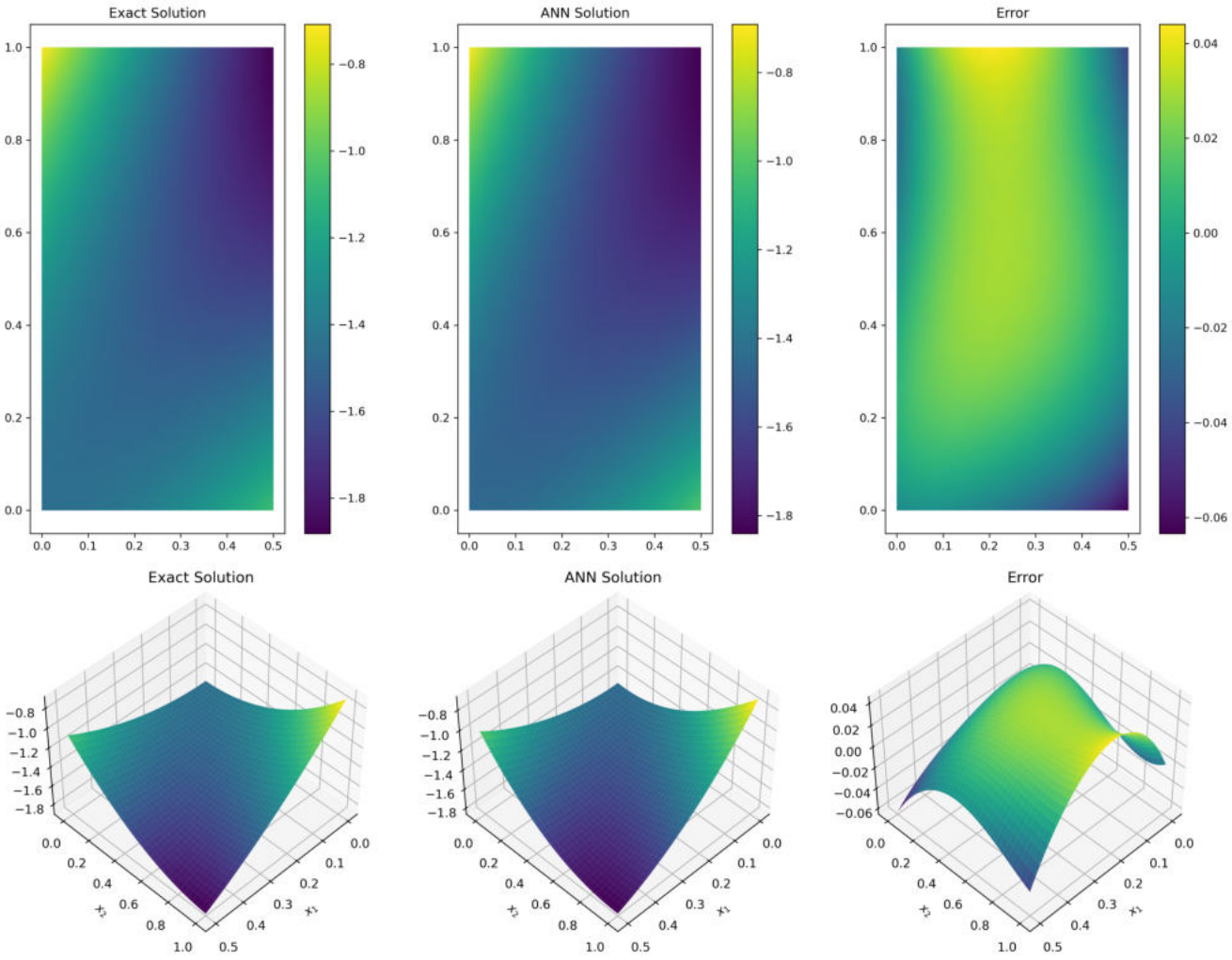
Hình 3.23: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



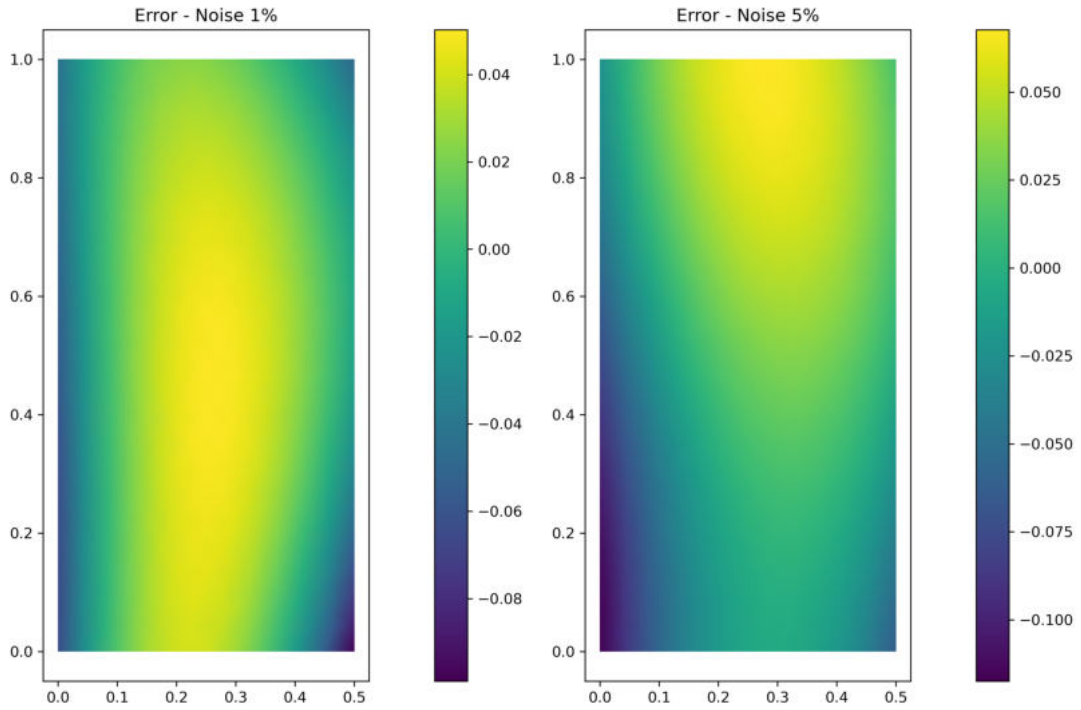
Hình 3.24: Hình minh họa 2D nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.007858$ chọn theo phương pháp L-curve.



Hình 3.25: Hình minh họa 3D nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.007858$ chọn theo phương pháp L-curve.



Hình 3.26: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.004693$ chọn theo phương pháp hậu nghiệm.



Hình 3.27: Sai số giữa nghiệm chính xác và nghiệm số của phương pháp ANN trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.004693	0.009182	0.047321
Sai số - L^2	1.703517e-02	3.408868e-02	4.205427e-02

Bảng 3.12: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	1.832647e-02	3.725541e-02	4.634183e-02

Bảng 3.13: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.007858$ chọn theo phương pháp L-curve.

3.3. Ví dụ cho bài toán phi tuyến 2D

Cho $\Omega \subset \mathbb{R}^n$ là một tập mở bị chặn và $\Gamma \subset \partial\Omega$ là một đa tạp nào đó. Cho hàm số $q : \mathbb{R} \to \mathbb{R}+$, ký hiệu P là một toán tử elliptic bậc hai có dạng

$$P(u) = -\nabla \cdot (q(u)\nabla u)$$

được định nghĩa trên Ω. Xét bài toán *Cauchy phi tuyến elliptic* như sau (không phụ thuộc vào thời gian):

$$\begin{cases} Pu = h, \text{ trong } \Omega \\ u = f, \text{ tại } \Gamma \\ q(u)u_v = g, \text{ tại } \Gamma \end{cases}$$

trong đó, các hàm số $f, g : \Gamma \to \mathbb{R}$ đã cho được gọi là dữ liệu Cauchy, phía bên phải của phương trình là một hàm số $h : \Omega \to \mathbb{R}$ được cho bởi $h(x, y) = -2\bar{u}(x_1, x_2)|\nabla \bar{u}(x_1, x_2)|^2$.

Ví dụ 3.6. Cho $\Omega \subset \mathbb{R}^2$ là hình tròn mở có tâm $(0.25, 0.25)$, $R = 0.25$, $\partial\Omega$ là biên và Γ_1 là tập con của $\partial\Omega$ (1/2 của biên). Lấy $q(t) = 1 + t^2$ và $\bar{u} : \bar{\Omega} \to \mathbb{R}$ là hàm điều hòa

$$\bar{u}(x, y) := e^{x_2} \sin(x_1).$$

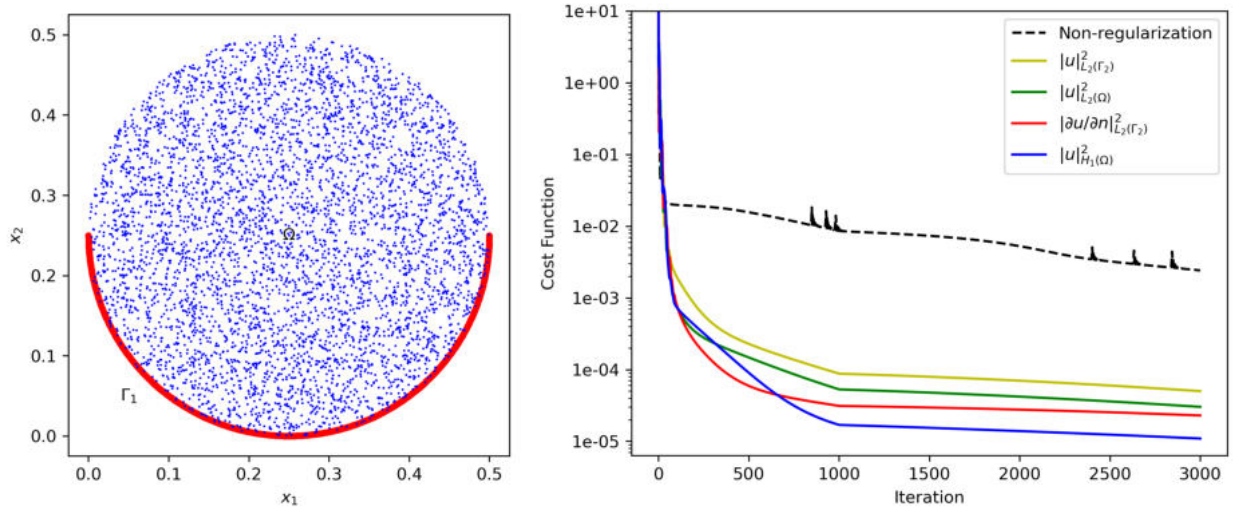
Xét bài toán Cauchy sau

$$\begin{cases} -\nabla \cdot (q(u)\nabla u) = h, \text{ trong } \Omega \\ u = f, \text{ trên } \Gamma_1 \\ \dfrac{\partial u(\mathbf{x}, \mathbf{y})}{\partial \mathbf{n}} = g, \text{ trên } \Gamma_1 \end{cases}$$

$$f(x) = e^{x_2} \sin(x_1), \quad g(x) = [e^{x_2} \cos(x_1), e^{x_2} \sin(x_1)] * \mathbf{n}$$

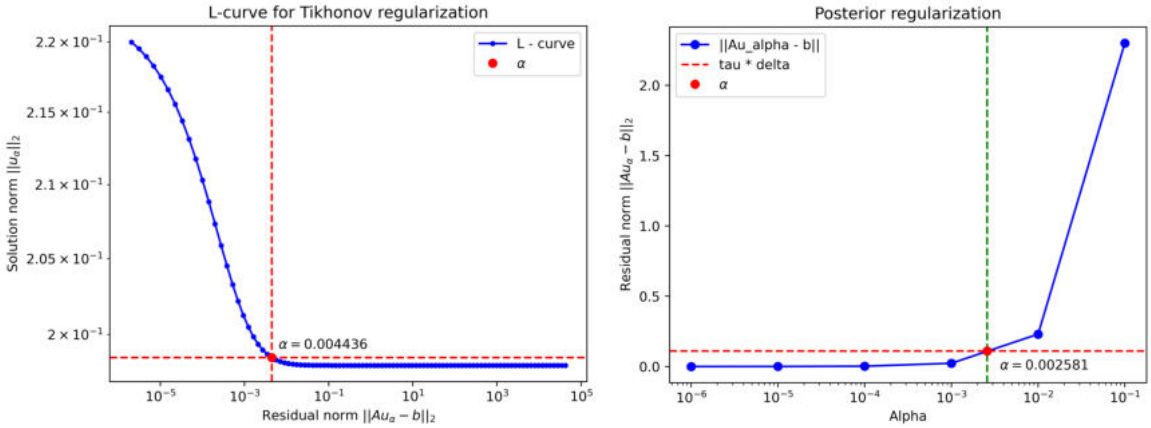
được cho tại Γ_1 và vế phải $h : \Omega \to \mathbb{R}$ được cho bởi

$$h(x, y) = -2\bar{u}(x_1, x_2)|\nabla \bar{u}(x_1, x_2)|^2 = -(e^{x_2} \sin(x_1))\sqrt{(e^{x_2} \cos(x_1))^2 + (e^{x_2} \sin(x_1))^2}$$

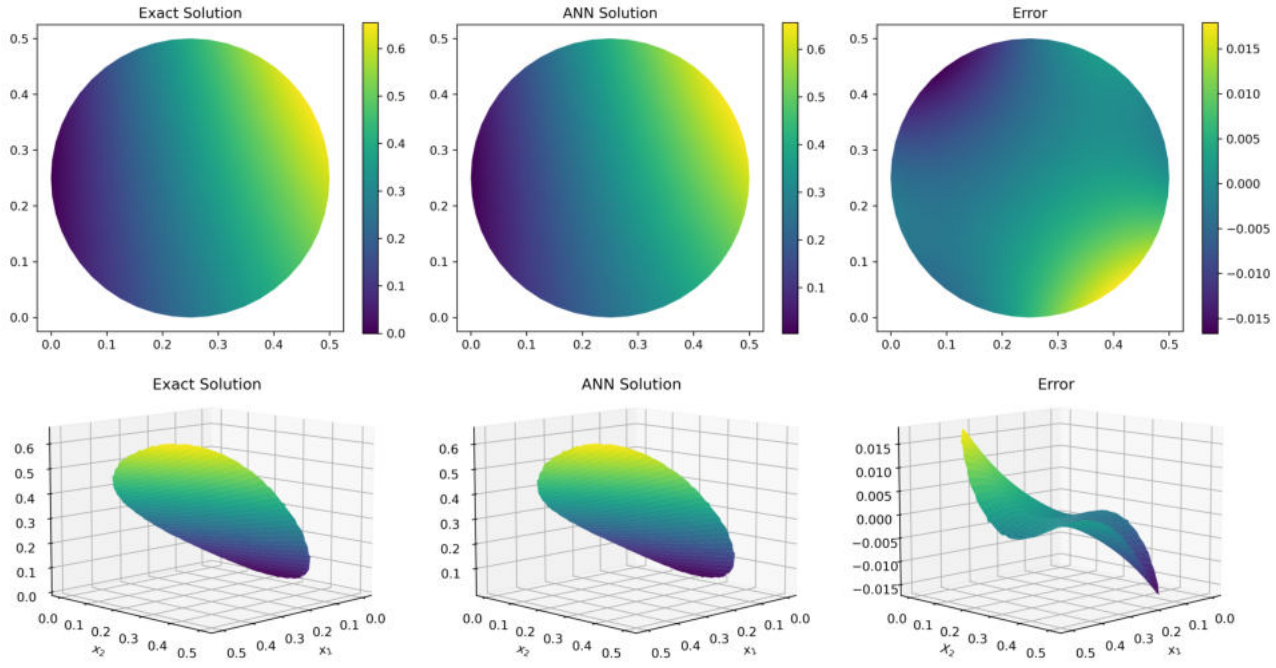


Hình 3.28: Miền dữ liệu Ω, Γ_1 và lịch sử hội tụ của thuật toán ADAM.

Trong ví dụ này các điểm dữ liệu trên miền Ω và biên Γ_1 được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3500$ điểm trên miền Ω và $N_b = 350$ điểm trên biên Γ_1.



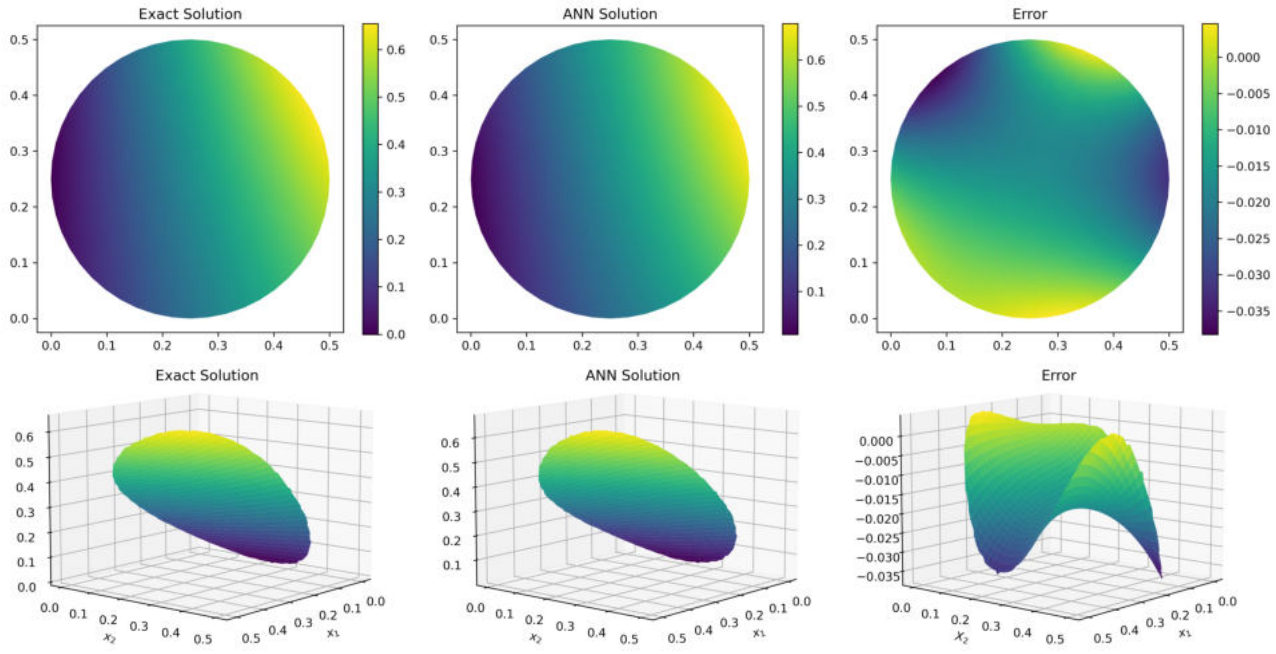
Hình 3.29: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



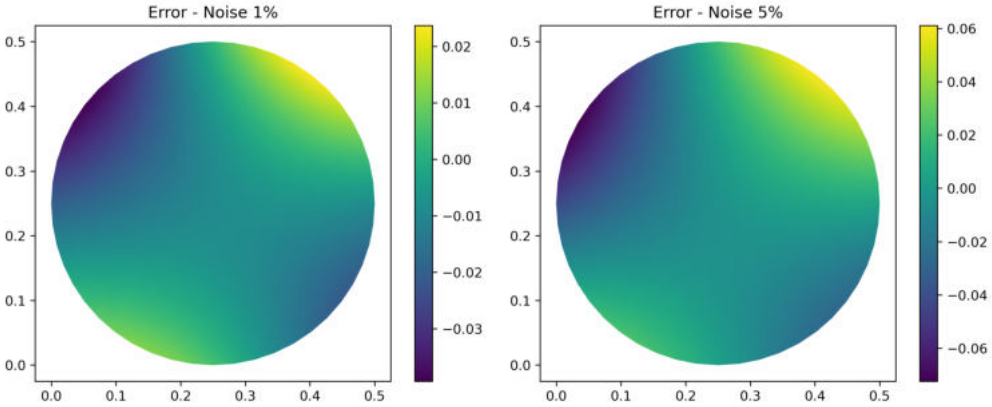
Hình 3.30: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.004436$ chọn theo phương pháp L-curve.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	8.173542e-03	1.854531e-02	2.823537e-02

Bảng 3.14: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.004436$ chọn theo phương pháp L-curve.



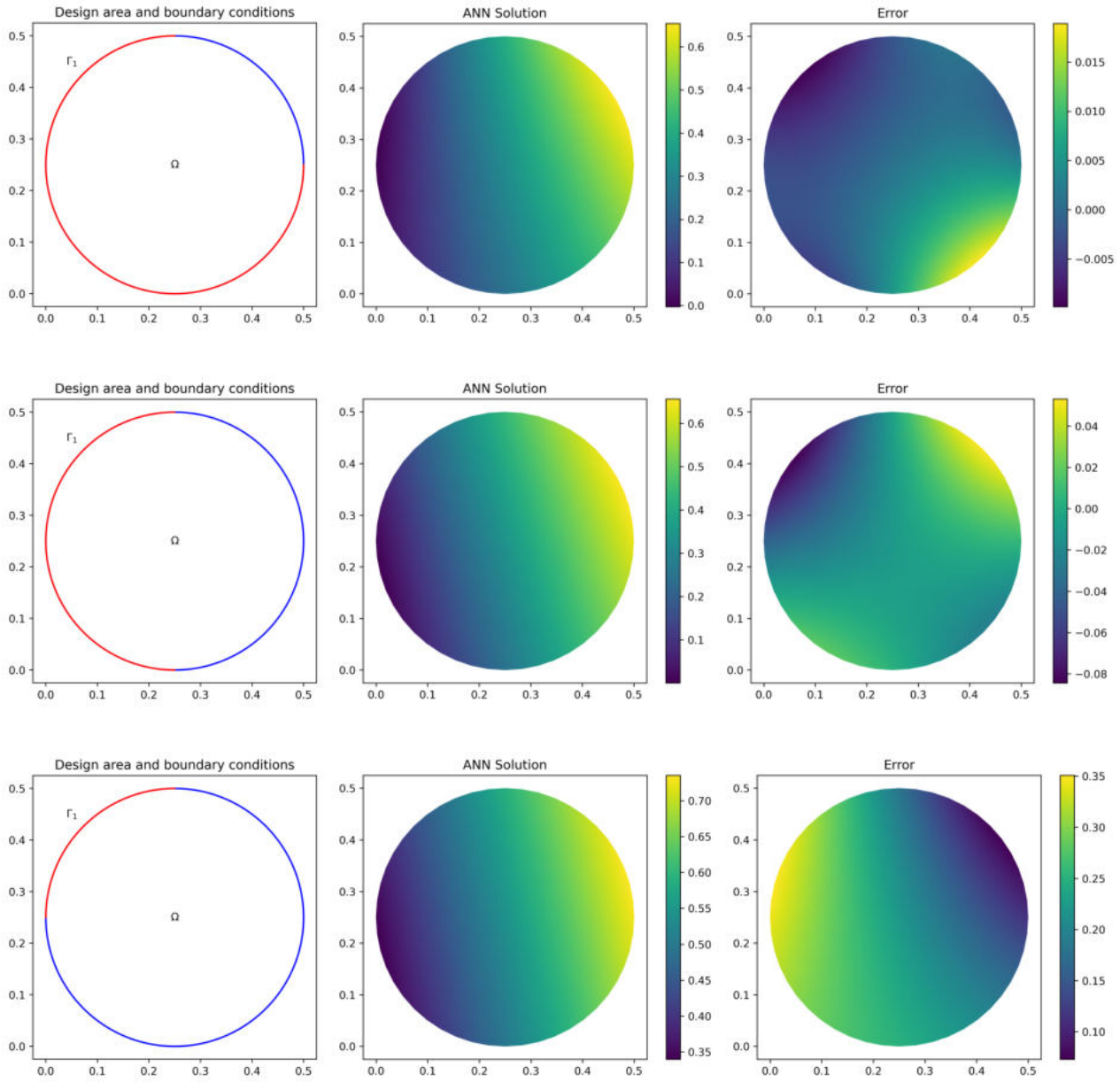
Hình 3.31: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.002581$ chọn theo phương pháp hậu nghiệm.



Hình 3.32: Sai số giữa nghiệm chính xác và nghiệm số ANN trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.002581	0.00814	0.023148
Sai số - L^2	6.586173e-03	1.428083e-02	2.566206e-02

Bảng 3.15: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.



Hình 3.33: So sánh các kết quả khi ta thay đổi dữ liệu Cauchy trên biên Γ_1 lần lược 3/4, 1/2, 1/4 cho bài toán. Tham số chỉnh hóa $\alpha = 0.004436$ chọn theo phương pháp L-curve.

Trường hợp	Γ_1 là 3/4 biên	Γ_1 là 1/2 biên	Γ_1 là 1/4 biên
Sai số - L^2	5.860621e-03	2.814800e-02	2.334579e-01

Bảng 3.16: Sai số L^2 trong các trường hợp thay đổi các điều kiện Cauchy với nhiễu 1%. Tham số chỉnh hóa $\alpha = 0.004436$ chọn theo phương pháp L-curve.

Ví dụ 3.7. Cho $\Omega \subset \mathbb{R}^2$ là hình vuông mở $(0, 0.5) \times (0, 0.5)$ và định nghĩa các tập con sau của $\partial\Omega$:

$$\Gamma_1 := \{(x, 0); x \in (0, 0.5)\}, \quad \Gamma_2 := \{(x, 0.5); x \in (0, 0.5)\},$$

$$\Gamma_3 := \{(0, y); y \in (0, 0.5)\}, \quad \Gamma_4 := \{(0.5, y); y \in (0, 0.5)\}.$$

Lấy $q(t) = 1 + t^2$ và $\bar{u} : \bar{\Omega} \to \mathbb{R}$ là hàm điều hòa

$$\bar{u}(x, y) := x^2 - y^2 - 2x + xy.$$

Xét bài toán Cauchy sau

$$\begin{cases} -\nabla \cdot (q(u)\nabla u) = h, \text{ trong } \Omega \\ u = f, \text{ trên } \Gamma_1 \\ q(u)u_v = g, \text{ trên } \Gamma_1 \\ u = \bar{u}, \text{ trên } \Gamma_3 \cup \Gamma_4 \end{cases}$$

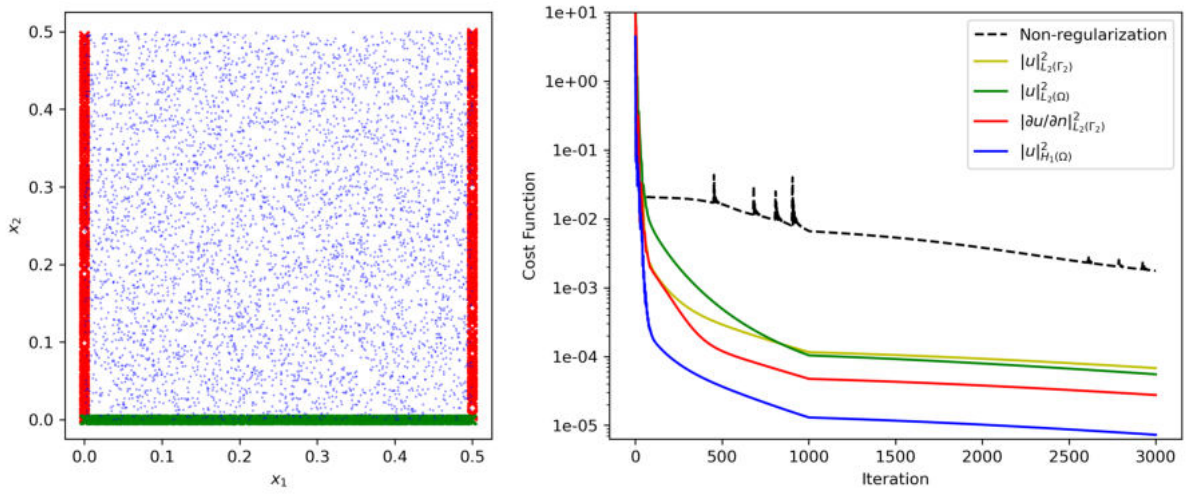
trong đó dữ liệu Cauchy

$$f(x) = x^2 - 2x, \quad g(x) = \left[1 + \left(x^2 - 2x\right)^2\right](-x)$$

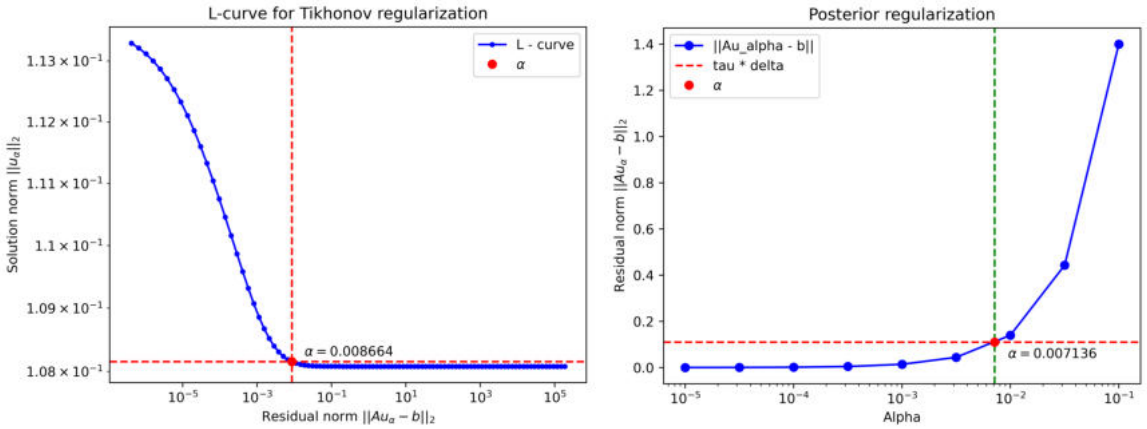
được cho tại Γ_1 và vế phải $h : \Omega \to \mathbb{R}$ được cho bởi

$$\begin{aligned} h(x, y) &= -2\bar{u}(x, y)|\nabla \bar{u}(x, y)|^2 \\ &= -2(x^2 - y^2 - 2x + xy)\sqrt{(2x - 2 - y)^2 + (-2y + x)^2} \end{aligned}$$

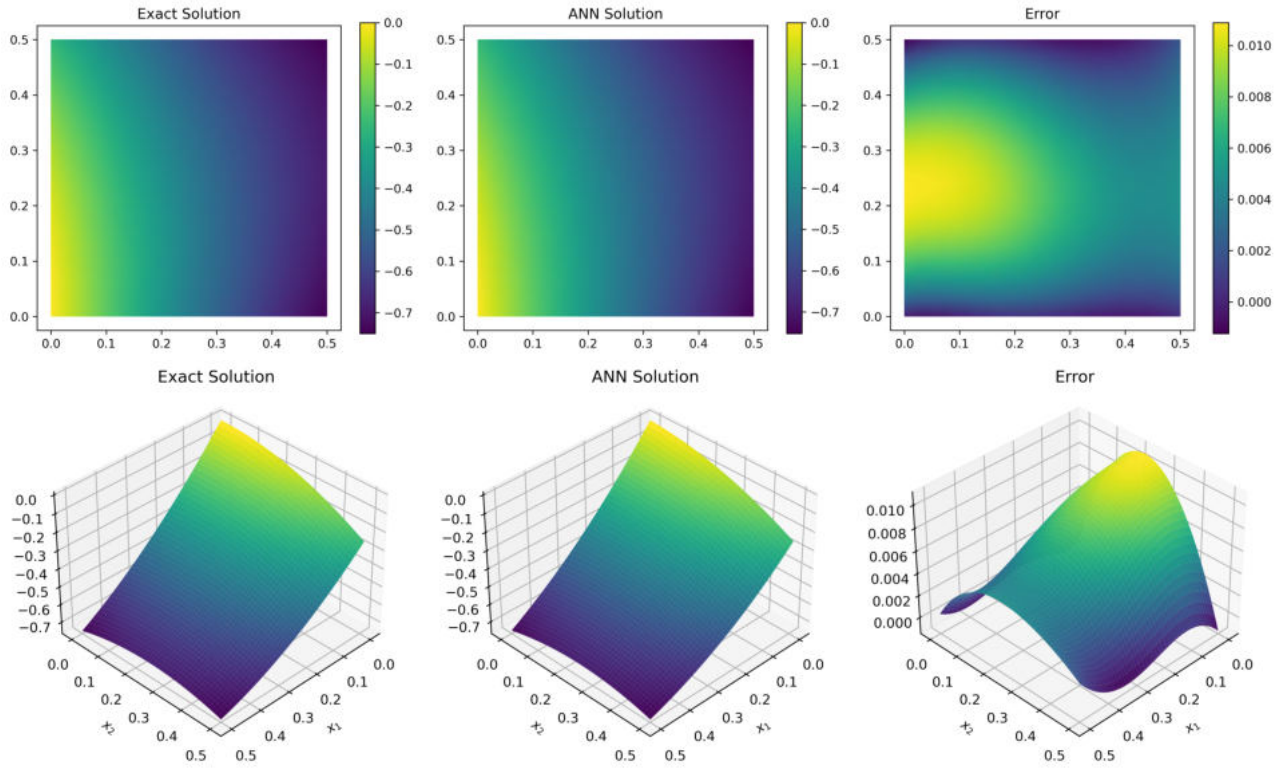
Trong ví dụ này các điểm dữ liệu trên miền Ω và biên $\Gamma_1, \Gamma_3 \cup \Gamma_4$ được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3000$ điểm trên miền Ω và $N_b = 300$ điểm trên biên $\Gamma_1, \Gamma_3 \cup \Gamma_4$.



Hình 3.34: Miền dữ liệu $\Omega, \Gamma_1, \Gamma_3 \cup \Gamma_4$ và lịch sử hội tụ của thuật toán ADAM.



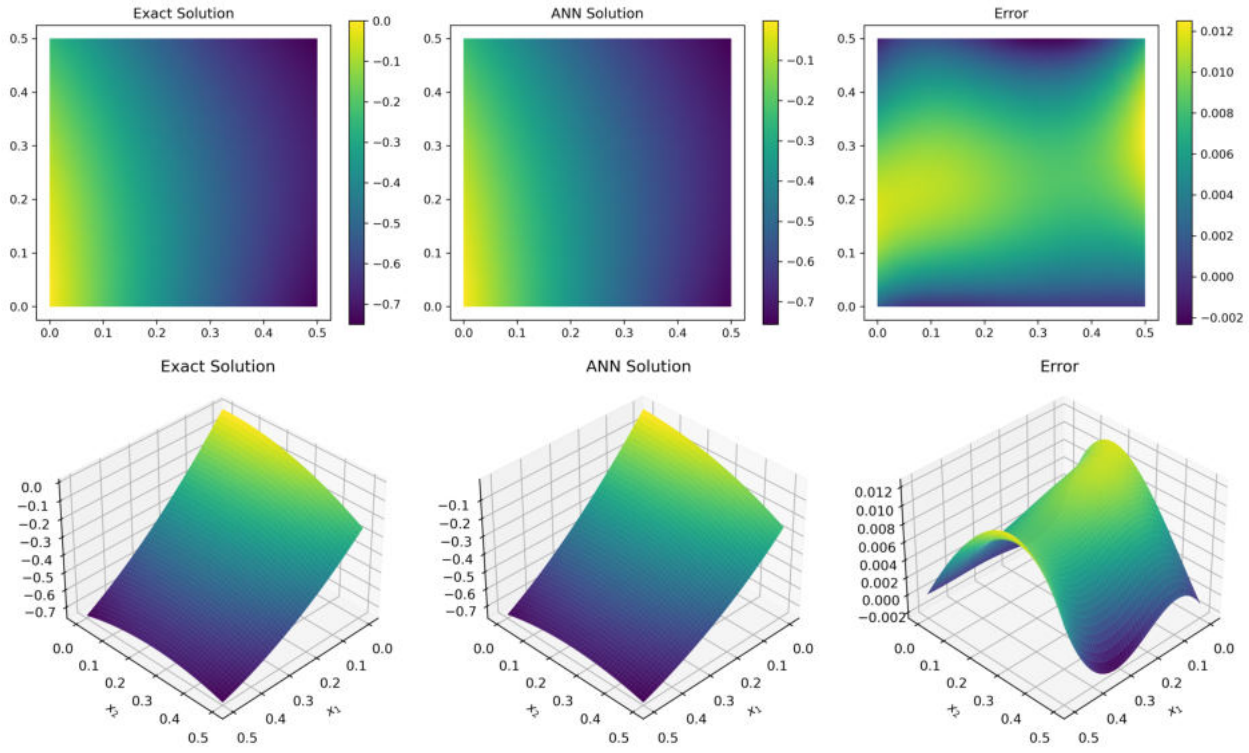
Hình 3.35: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



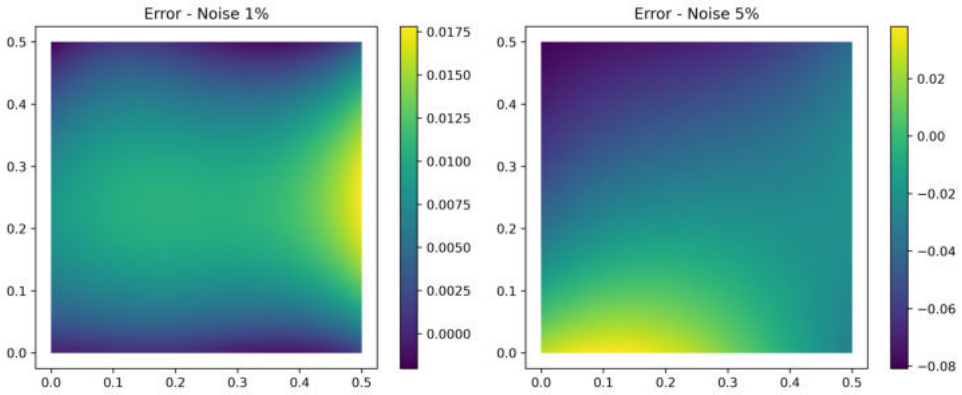
Hình 3.36: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.008664$ chọn theo phương pháp L-curve.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	5.702368e-03	7.164284e-03	3.225173e-02

Bảng 3.17: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.008664$ chọn theo phương pháp L-curve.



Hình 3.37: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.007136$ chọn theo phương pháp hậu nghiệm.



Hình 3.38: Sai số giữa nghiệm chính xác và nghiệm số ANN trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.007136	0.009521	0.049123
Sai số - L^2	7.162943e-03	8.205305e-03	3.797375e-02

Bảng 3.18: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Ví dụ 3.8. Cho $\Omega \subset \mathbb{R}^2$ là hình vuông mở $(0, 0.25) \times (0, 0.25)$ và định nghĩa các tập con sau của $\partial\Omega$:

$$\Gamma_1 := \{(x, 0); x \in (0, 0.25)\}, \quad \Gamma_2 := \{(x, 0.25); x \in (0, 0.25)\},$$

$$\Gamma_3 := \{(0, y); y \in (0, 0.25)\}, \quad \Gamma_4 := \{(0.25, y); y \in (0, 0.25)\}.$$

Lấy $q(t) = 1 + t^2$ và $\bar{u} : \bar{\Omega} \to \mathbb{R}$ là hàm điều hòa

$$\bar{u}(x, y) := x^2 - y^2 - 3xy + e^x \cos y$$

Xét bài toán Cauchy sau

$$\begin{cases} -\nabla \cdot (q(u)\nabla u) = h, \text{ trong } \Omega \\ u = f, \text{ trên } \Gamma_1 \\ q(u)u_v = g, \text{ trên } \Gamma_1 \\ u = \bar{u}, \text{ trên } \Gamma_3 \cup \Gamma_4 \end{cases}$$

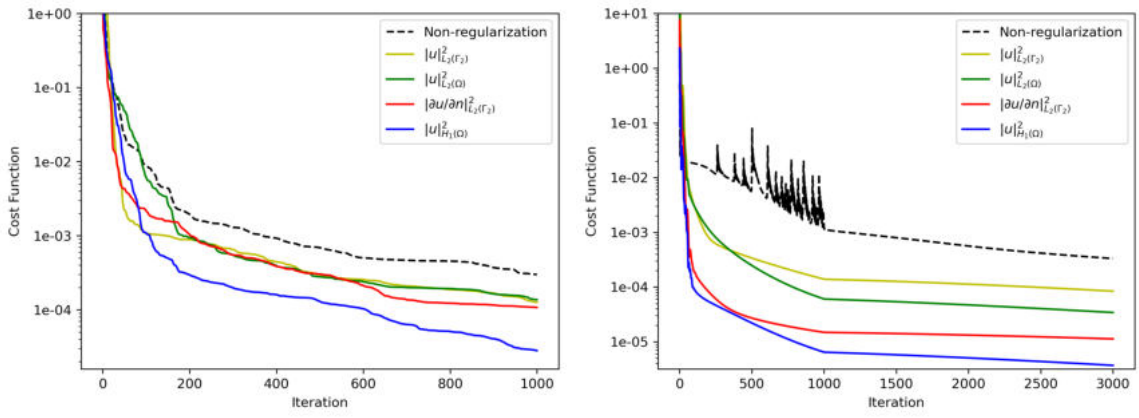
trong đó dữ liệu Cauchy

$$f(x) = x^2 + e^x, \quad g(x) = \left[1 + \left(x^2 + e^x\right)^2\right](3x)$$

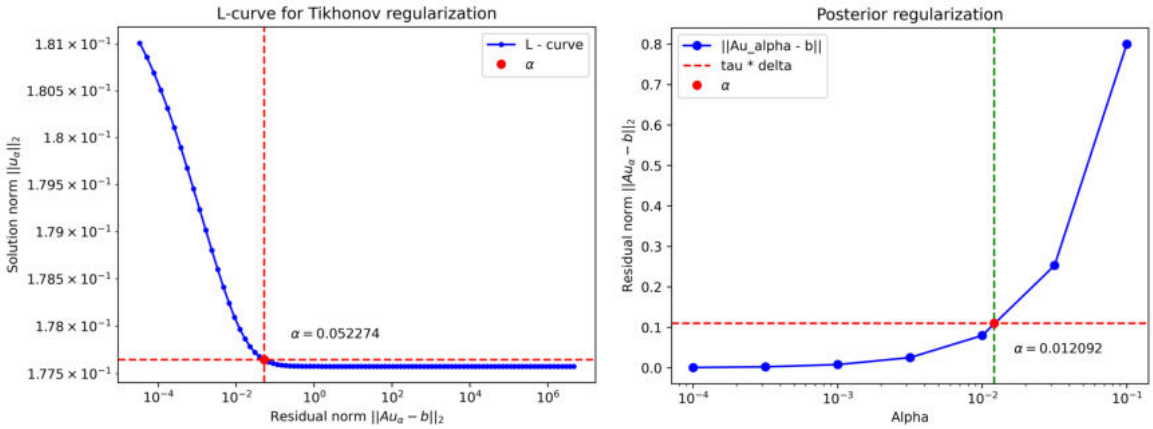
được cho tại Γ_1 và vế phải $h : \Omega \to \mathbb{R}$ được cho bởi

$$\begin{aligned} h(x, y) &= -2\bar{u}(x, y)|\nabla \bar{u}(x, y)|^2 \\ &= -2(x^2 - y^2 - 3xy + e^x \cos y)\sqrt{(2x - 3y + e^x \cos y)^2 + (-2y - 3x - e^x \sin y)^2} \end{aligned}$$

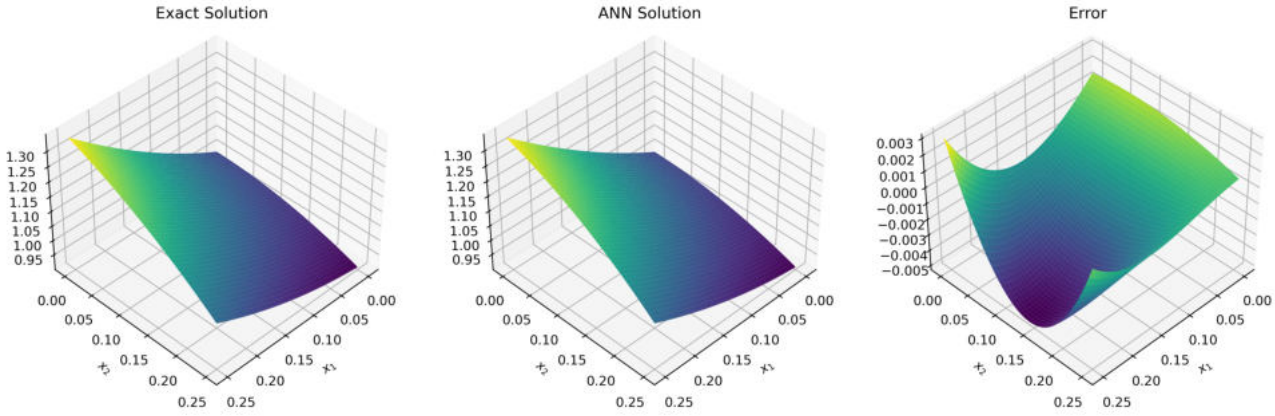
Trong ví dụ này các điểm dữ liệu trên miền Ω và biên $\Gamma_1, \Gamma_3 \cup \Gamma_4$ được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3000$ điểm trên miền Ω và $N_b = 300$ điểm trên biên $\Gamma_1, \Gamma_3 \cup \Gamma_4$.



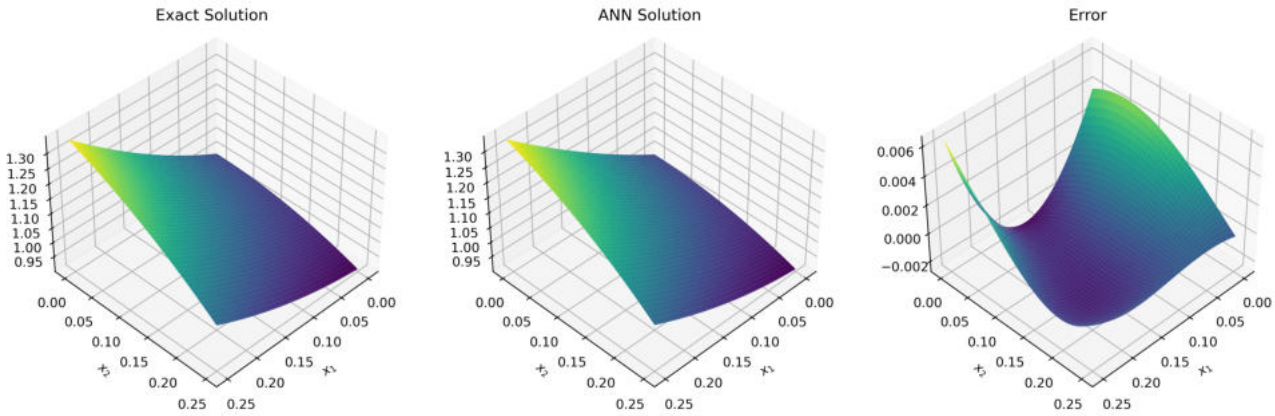
Hình 3.39: Lịch sử hội tụ qua mỗi bước lặp, bên trái là thuật toán L-BFGS, bên phải là ADAM.



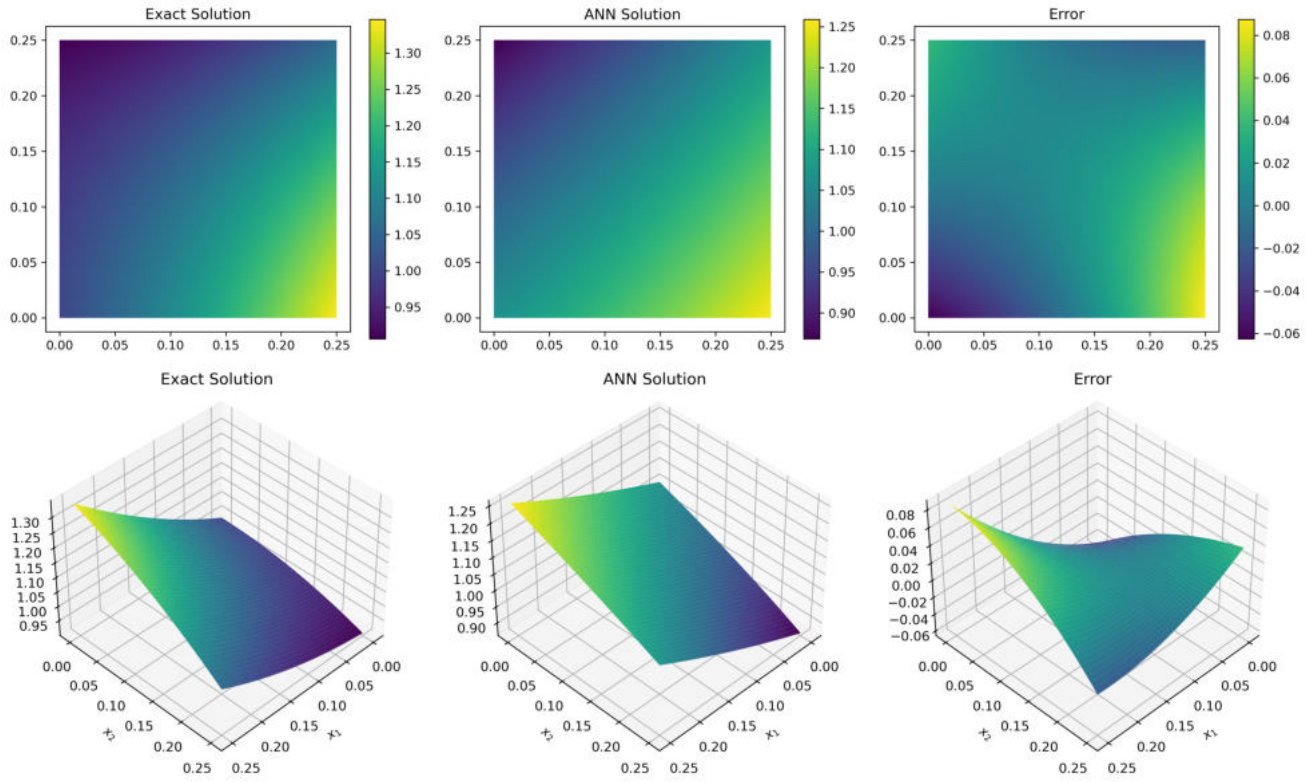
Hình 3.40: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



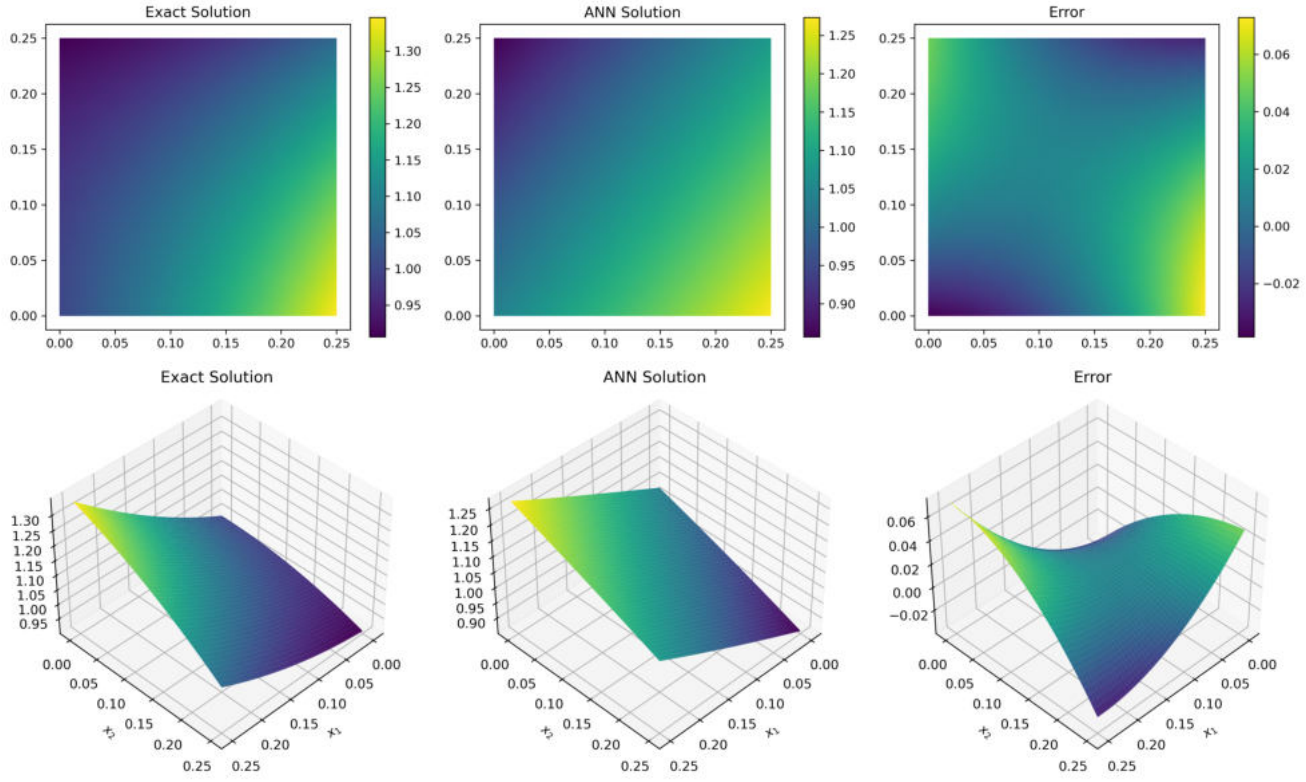
Hình 3.41: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1% bằng thuật toán L-BFGS. Tham số chỉnh hóa $\alpha = 0.052274$ chọn theo phương pháp L-curve.



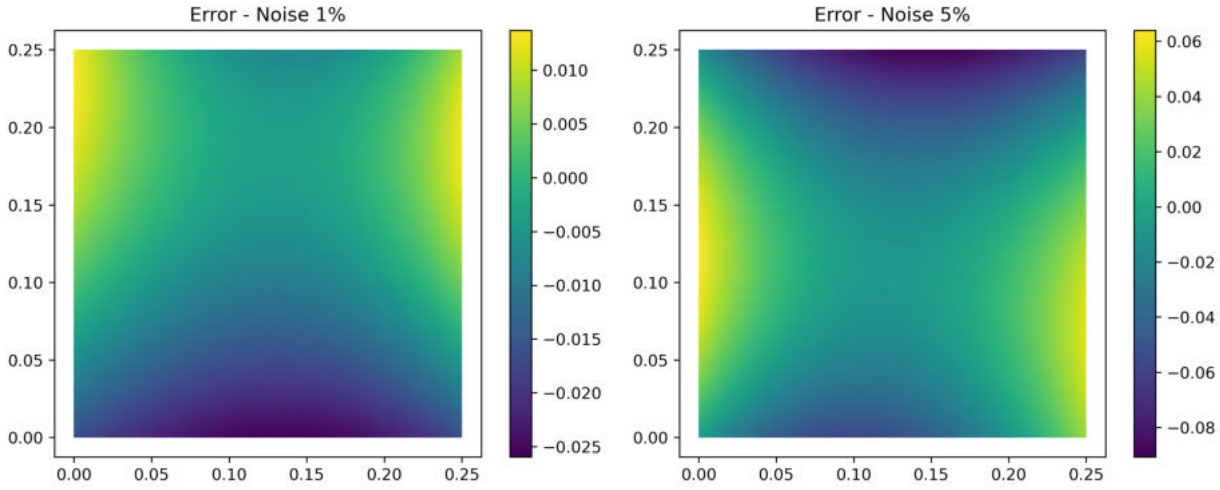
Hình 3.42: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1% bằng thuật toán L-BFGS. Tham số chỉnh hóa $\alpha = 0.012092$ chọn theo phương pháp hậu nghiệm.



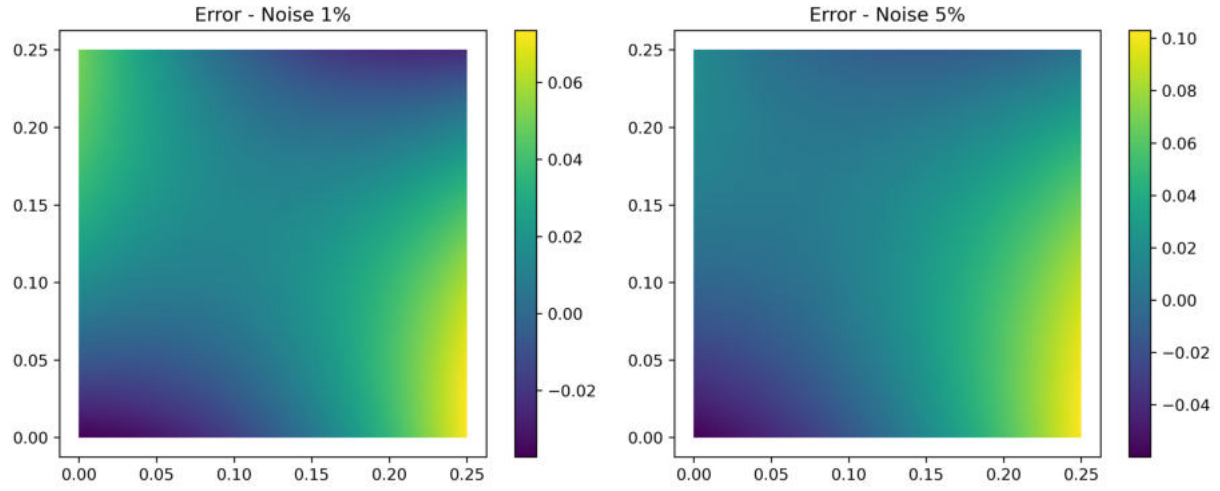
Hình 3.43: Nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1% bằng thuật toán ADAM. Tham số chỉnh hóa $\alpha = 0.004436$ chọn theo phương pháp L-curve.



Hình 3.44: Nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1% bằng thuật toán ADAM. Tham số chỉnh hóa $\alpha = 0.012092$ chọn theo phương pháp hậu nghiệm.



Hình 3.45: Sai số giữa nghiệm chính xác và nghiệm số trong trường hợp nhiễu 1%, 5% - thuật toán L-BFGS. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.



Hình 3.46: Sai số giữa nghiệm chính xác và nghiệm số trong trường hợp nhiễu 1%, 5% - thuật toán Adam. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.012092	0.059738	0.081362
Sai số L^2 - Thuật toán L-BFGS	2.345575e-03	8.085642e-03	2.452185e-02
Sai số L^2 - Thuật toán ADAM	1.353665e-02	2.672507e-02	4.154810e-02

Bảng 3.19: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
Sai số L^2 - Thuật toán L-BFGS	1.723624e-03	5.153463e-03	2.237712e-02
Sai số L^2 - Thuật toán ADAM	2.016246e-02	3.615211e-02	4.162531e-02

Bảng 3.20: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.052274$ chọn theo phương pháp L-curve.

L-BFGS	Không chỉnh	$\left\|\bar{u}\right\|^2_{L_2(\Gamma 2)}$	$\left\|\bar{u}\right\|^2_{L_2(\Omega)}$	$\left\|\frac{\partial \bar{u}}{\partial n}\right\|^2_{L_2(\Gamma 2)}$	$\left\|\bar{u}\right\|^2_{H_1(\Omega)}$
L - N	3 - 7	3 - 7	3 - 7	3 - 7	3 - 7
Nhiễu (%)	0.1%	0.1%	0.1%	0.1%	0.1%
α	0	0.012092	0.012092	0.012092	0.012092
Bước lặp	1000	1000	1000	1000	1000
Thời gian/Bước lặp	0.0731s	0.0846s	0.0725s	0.0871s	0.0931s
Hàm mất mát (Loss)	2.99e-04	1.26e-04	1.36e-04	1.07e-04	2.81e-05
ADAM	Không chỉnh	$\left\|\bar{u}\right\|^2_{L_2(\Gamma 2)}$	$\left\|\bar{u}\right\|^2_{L_2(\Omega)}$	$\left\|\frac{\partial \bar{u}}{\partial n}\right\|^2_{L_2(\Gamma 2)}$	$\left\|\bar{u}\right\|^2_{H_1(\Omega)}$
L - N	3 - 7	3 - 7	3 - 7	3 - 7	3 - 7
Nhiễu (%)	0.1%	0.1%	0.1%	0.1%	0.1%
α	0	0.012092	0.012092	0.012092	0.012092
Bước lặp	3000	3000	3000	3000	3000
Thời gian/Bước lặp	0.00647s	0.00731s	0.00815s	0.00708s	0.00831s
Hàm mất mát (Loss)	3.31e-04	8.39e-05	3.41e-05	1.12e-05	3.69e-06

Bảng 3.21: Kết quả số cho bài toán bằng thuật toán L-BFGS và thuật toán ADAM

Từ bảng số liệu trên, ta có thể thấy rằng thuật toán ADAM hoạt động nhanh hơn so với thuật toán L-BFGS, tuy nhiên kết quả của ADAM khá chênh lệch so với kết

quả L-BFGS. Số liệu trong bảng này cho chúng ta thấy rằng khi sử dụng thuật toán ADAM, số lần lặp cần thiết để đạt được kết quả tốt là 3000, trong khi đó khi sử dụng thuật toán L-BFGS thì chỉ cần 1000 lần lặp để đạt được kết quả tương tự. Tuy nhiên, thời gian tính toán mỗi lần lặp lại của thuật toán L-BFGS lớn hơn đáng kể so với thuật toán ADAM.

Từ kết quả so sánh phía trên cùng với các kết quả [44] nghiên cứu của các tác giả đi trước, ta thấy rằng L-BFGS có thể tìm ra giải pháp chấp nhận được nhanh hơn ADAM và sử dụng ít vòng lặp hơn. Điều này là rõ ràng bởi vì thuật toán tối ưu hóa ADAM chỉ dựa vào đạo hàm bậc nhất trong khi đó L-BFGS sử dụng đạo hàm bậc hai của hàm tổn thất. Tuy nhiên, theo như kinh nghiệm khi chạy số, trong nhiều trường hợp khi sử dụng L-BFGS sẽ bị mắc kẹt ở nghiệm cục bộ. Do đó, khi chạy số ta nên kết hợp cả hai thuật toán ADAM và L-BFGS để có thể cho ra một kết quả tốt nhất có thể.

3.4. Ví dụ cho bài toán phi tuyến 3D

Cho $\Omega \subset \mathbb{R}^3$ là hình khối 3D, được định nghĩa $\Omega := \{(x, y, z) \in (0, a) \times (0, b) \times (0, c)\}$. Xét bài toán Cauchy cho phương trình phi tuyến trong miền Ω

$$u_{xx}(x, y, z) + u_{yy}(x, y, z) + u_{zz}(x, y, z) + \mathcal{L}[u(x, y, z)] = S(x, y, z) + \mathcal{N}[u(x, y, z)], \quad (3.1)$$

và định nghĩa các tập con của $\partial\Omega$:

$$\Gamma_1 := \{x | 0 \le x_1 \le a, 0 \le x_2 \le b, x_3 = 0\}, \quad \Gamma_2 := \{x | 0 \le x_1 \le a, 0 \le x_2 \le b, x_3 = c\},$$
$$\Gamma_3 := \{x | x_1 = 0, 0 \le x_2 \le b, 0 \le x_3 \le c\}, \quad \Gamma_4 := \{x | x_1 = a, 0 \le x_2 \le b, 0 \le x_3 \le c\},$$
$$\Gamma_5 := \{x | 0 \le x_1 \le a, x_2 = 0, 0 \le x_3 \le c\}, \quad \Gamma_6 := \{x | 0 \le x_1 \le a, x_2 = b, 0 \le x_3 \le c\}.$$

trong đó, $\mathcal{L}$ là một toán tử tuyến tính bậc nhất, $\mathcal{N}$ là một toán tử phi tuyến bậc nhất.

Dữ liệu Cauchy được cho như sau

$$\begin{cases} u(x_1, x_2, 0) = f & \mathbf{x} \text{ trên } \Gamma_1 \\ \frac{\partial u(x_1, x_2, x_3)}{\partial \mathbf{n}} = \nabla(u) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1 \\ u(x_1, x_2, x_3) = \bar{u} & \mathbf{x} \text{ trên } \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6 \end{cases} \quad (3.2)$$

Bài toán tìm u thỏa mãn (3.1)-(3.2) được là bài toán Cauchy cho phương trình phi tuyến (3.1). Dữ liệu $u(x, y, c)$ không được xác định cụ thể. Tuy nhiên, chúng ta đã xác định dữ liệu trên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$ để khôi phục dữ liệu trên biên còn lại của bài toán, từ đó hoàn tất dữ liệu trên toàn biên. Đây là một dạng bài toán Cauchy cho hình hộp.

Ví dụ 3.9. Cho $\Omega \subset \mathbb{R}^3$ là hình khối 3D gồm tất cả các điểm (x_1, x_2, x_3) trong đó $0 \le x_1 \le 0.5$, $0 \le x_2 \le 1$, $0 \le x_3 \le 1$ và định nghĩa các tập con của $\partial\Omega$:

$$\Gamma_1 := \{x | 0 \le x_1 \le 0.5, 0 \le x_2 \le 1, x_3 = 0\}, \quad \Gamma_2 := \{x | 0 \le x_1 \le 0.5, 0 \le x_2 \le 1, x_3 = 1\},$$
$$\Gamma_3 := \{x | x_1 = 0, 0 \le x_2 \le 1, 0 \le x_3 \le 1\}, \quad \Gamma_4 := \{x | x_1 = 0.5, 0 \le x_2 \le 1, 0 \le x_3 \le 1\},$$
$$\Gamma_5 := \{x | 0 \le x_1 \le 0.5, x_2 = 0, 0 \le x_3 \le 1\}, \quad \Gamma_6 := \{x | 0 \le x_1 \le 0.5, x_2 = 1, 0 \le x_3 \le 1\}.$$

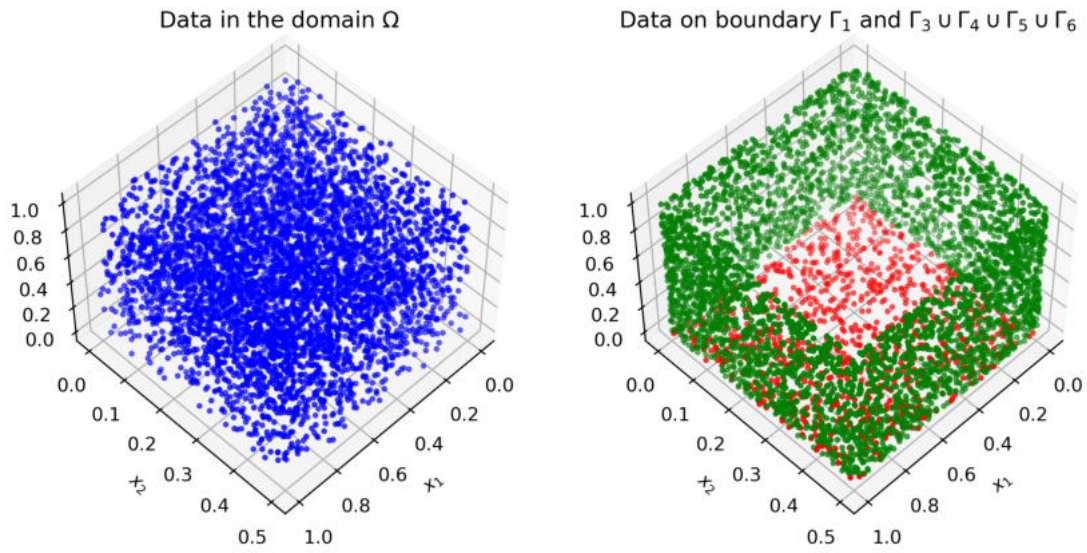
Xét bài toán Cauchy sau

$$u_{x_1x_1} + u_{x_2x_2} + u_{x_3x_3} + \mathcal{L}[u(x_1, x_2, x_3)] = S(x_1, x_2, x_3) + \mathcal{N}[u(x_1, x_2, x_3)], \quad \mathbf{x} \text{ trong } \Omega, \tag{3.3}$$

dữ liệu Cauchy được cho như sau

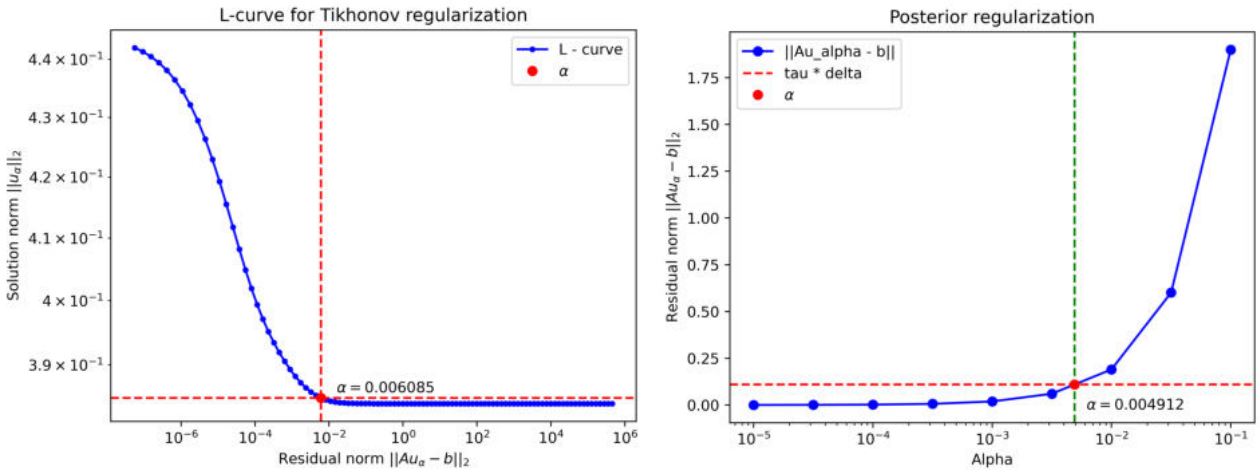
$$\begin{cases} u(x_1, x_2, 0) = xye^x + xy^2 & \mathbf{x} \text{ trên } \Gamma_1 \\ \frac{\partial u(x_1,x_2,x_3)}{\partial \mathbf{n}} = \nabla(xye^x + xy^2 + x^2z + xz^2) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1 \\ u(x_1, x_2, x_3) = xye^x + xy^2 + x^2z + xz^2 & \mathbf{x} \text{ trên } \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6 \end{cases} \tag{3.4}$$

Nghiệm chính xác của bài toán là $u(x_1, x_2, x_3) = xye^x + xy^2 + x^2z + xz^2$, $\mathcal{L} = -2u$, $\mathcal{N} = e^u + u^2$ và $S(x_1, x_2, x_3)$ dễ dàng suy ra khi ta thế nghiệm chính xác vào phương trình (3.3).

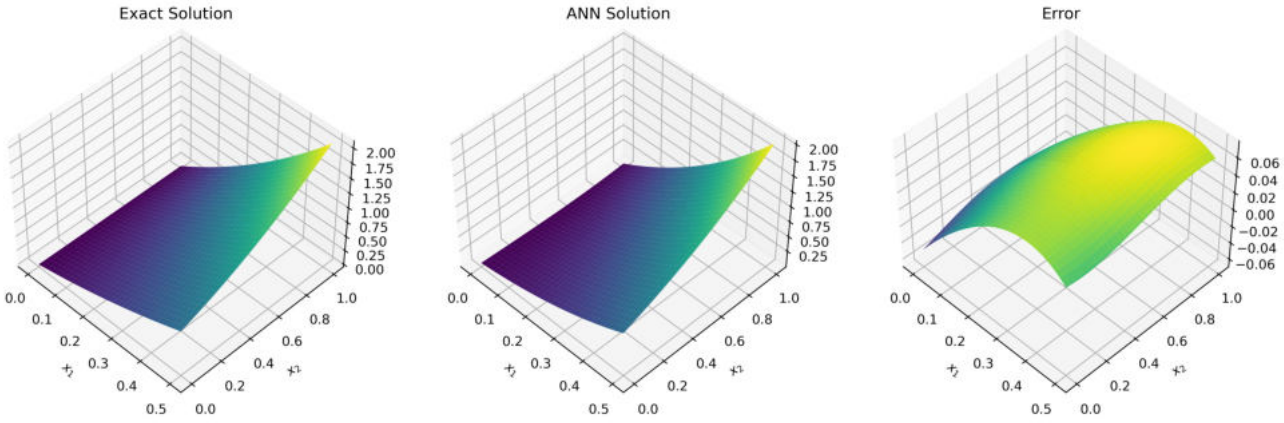


Hình 3.47: Miền dữ liệu Ω(màu xanh), Γ_1(màu đỏ) và $\Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$(màu xanh lá).

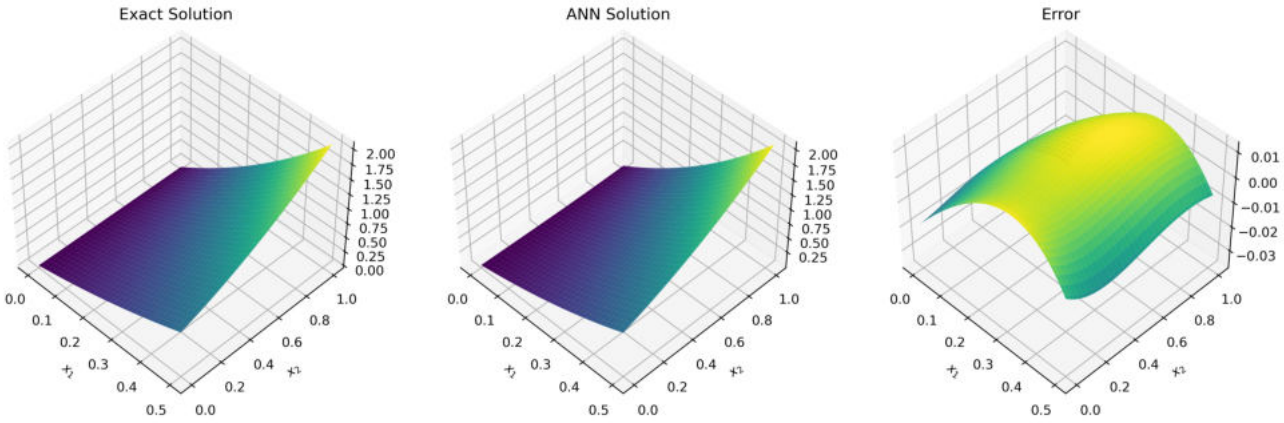
Trong ví dụ này các điểm dữ liệu trên miền Ω và biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$ được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3500$ điểm trên miền Ω và $N_b = 1000$ điểm trên biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$. Chạy bằng thuật toán L-BFGS.



Hình 3.48: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



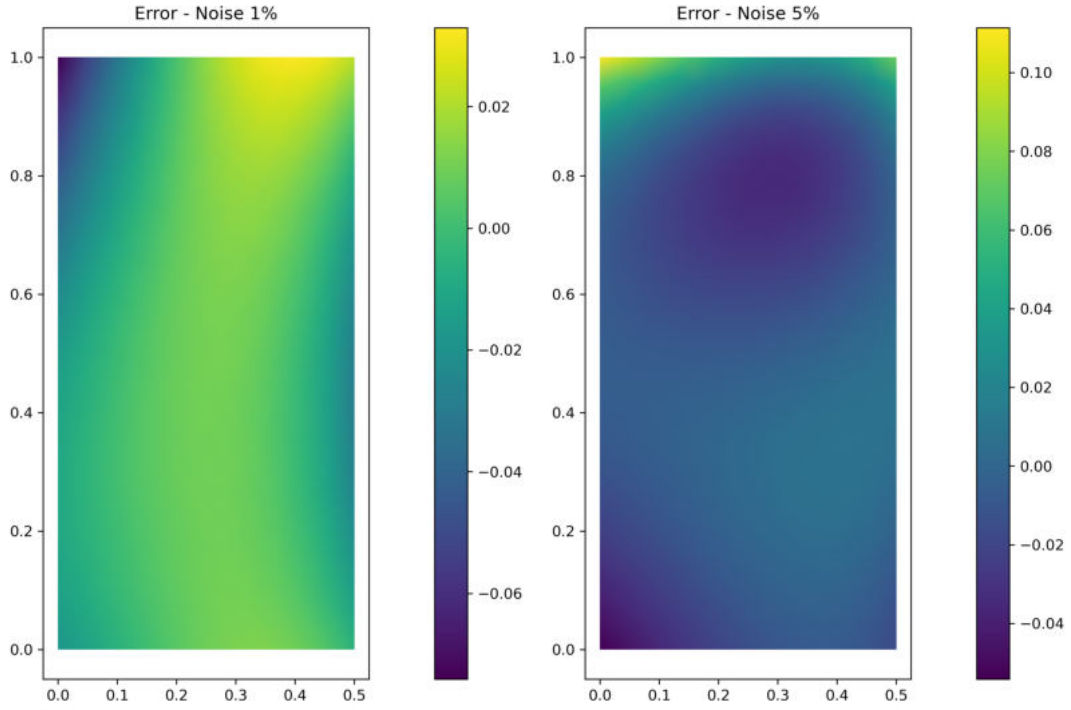
Hình 3.49: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.006085$ chọn theo phương pháp L-curve.



Hình 3.50: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.004912$ chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	1.253721e-02	2.254321e-02	4.745323e-02

Bảng 3.22: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.004436$ chọn theo phương pháp L-curve.



Hình 3.51: Sai số giữa nghiệm chính xác và nghiệm số của phương pháp ANN trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.004912	0.007938	0.018333
Sai số - L^2	9.598386e-03	1.626991e-02	4.174019e-02

Bảng 3.23: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Ví dụ 3.10. Cho $\Omega \subset \mathbb{R}^3$ là hình khối 3D gồm tất cả các điểm (x_1, x_2, x_3) trong đó $0 \leq x_1 \leq 1$, $0 \leq x_2 \leq 1$, $0 \leq x_3 \leq 1$ và định nghĩa các tập con của $\partial\Omega$:

$$\Gamma_1 := \{x|0 \le x_1 \le 1, 0 \le x_2 \le 1, x_3 = 0\}, \quad \Gamma_2 := \{x|0 \le x_1 \le 1, 0 \le x_2 \le 1, x_3 = 1\},$$
$$\Gamma_3 := \{x|x_1 = 0, 0 \le x_2 \le 1, 0 \le x_3 \le 1\}, \quad \Gamma_4 := \{x|x_1 = 1, 0 \le x_2 \le 1, 0 \le x_3 \le 1\},$$
$$\Gamma_5 := \{x|0 \le x_1 \le 1, x_2 = 0, 0 \le x_3 \le 1\}, \quad \Gamma_6 := \{x|0 \le x_1 \le 1, x_2 = 1, 0 \le x_3 \le 1\}.$$

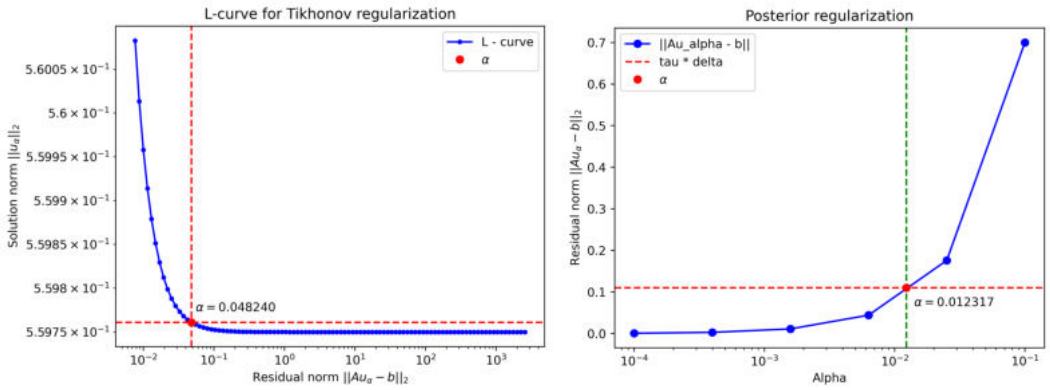
Xét bài toán Cauchy sau

$$u_{x_1x_1} + u_{x_2x_2} + u_{x_3x_3} + \mathcal{L}[u(x_1, x_2, x_3)] = S(x_1, x_2, x_3) + \mathcal{N}[u(x_1, x_2, x_3)], \quad \mathbf{x} \text{ trong } \Omega, \tag{3.5}$$

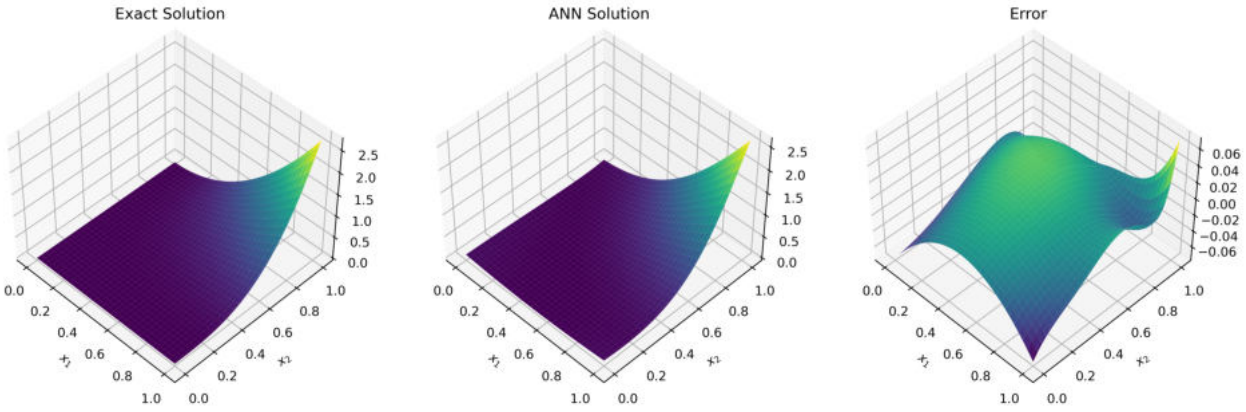
dữ liệu Cauchy được cho như sau

$$\begin{cases} u(x_1, x_2, 0) = x^2y^3 & \mathbf{x} \text{ trên } \Gamma_1 \\ \frac{\partial u(x_1,x_2,x_3)}{\partial \mathbf{n}} = \nabla(x^2y^3e^z) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1 \\ u(x_1, x_2, x_3) = x^2y^3e^z & \mathbf{x} \text{ trên } \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6 \end{cases} \tag{3.6}$$

Với nghiệm chính xác của bài toán là $u(x_1, x_2, x_3) = x^2y^3e^z$, $\mathcal{L} = 2u$, $\mathcal{N} = u_{x_1}^2 + u_{x_2}^2 + u^4 - 2uu_{x_1}$ và $S(x_1, x_2, x_3)$ dễ dàng suy ra khi ta thế nghiệm chính xác vào phương trình (3.5).



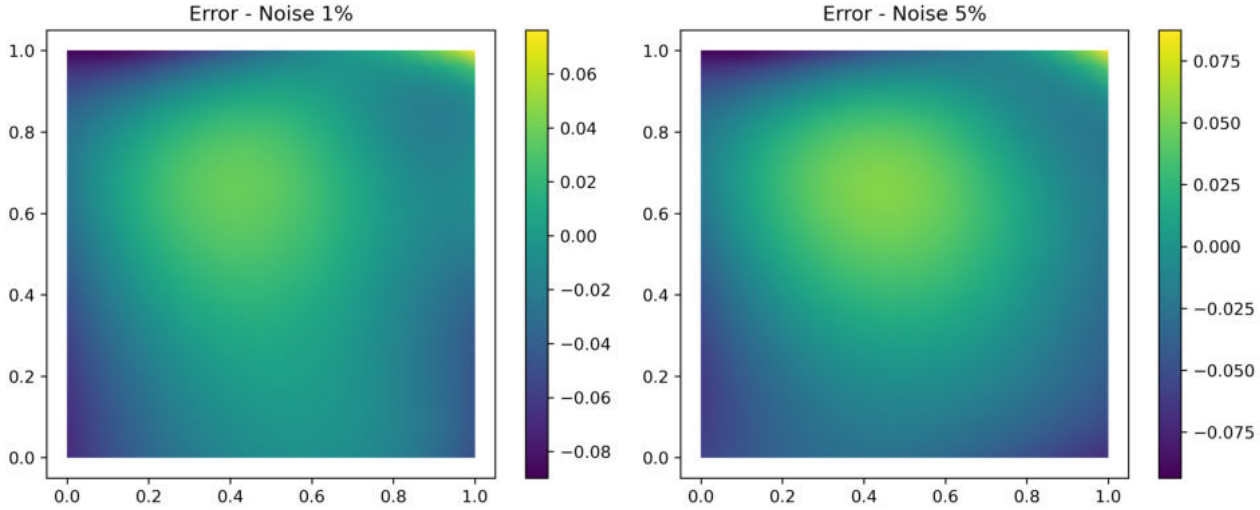
Hình 3.52: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



Hình 3.53: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.048240$ chọn theo phương pháp L-curve.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	2.2634271e-02	3.1524983e-02	4.373218e-02

Bảng 3.24: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.048240$ chọn theo phương pháp L-curve.



Hình 3.54: Sai số giữa nghiệm chính xác và nghiệm số ANN trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.012317	0.062317	0.089273
Sai số - L^2	1.352378e-02	2.509405e-02	3.091071e-02

Bảng 3.25: Tham số chỉnh hóa α và sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Ví dụ 3.11. Cho $\Omega \subset \mathbb{R}^3$ là hình khối 3D gồm tất cả các điểm (x_1, x_2, x_3) trong đó $0 \leq x_1 \leq 0.5$, $0 \leq x_2 \leq 0.5$, $0 \leq x_3 \leq 1$ và định nghĩa các tập con của $\partial\Omega$:

$$\Gamma_1 := \{x | 0 \leq x_1 \leq 0.5, 0 \leq x_2 \leq 0.5, x_3 = 0\}, \quad \Gamma_2 := \{x | 0 \leq x_1 \leq 0.5, 0 \leq x_2 \leq 0.5, x_3 = 1\},$$

$$\Gamma_3 := \{x | x_1 = 0, 0 \leq x_2 \leq 0.5, 0 \leq x_3 \leq 1\}, \quad \Gamma_4 := \{x | x_1 = 0.5, 0 \leq x_2 \leq 0.5, 0 \leq x_3 \leq 1\},$$

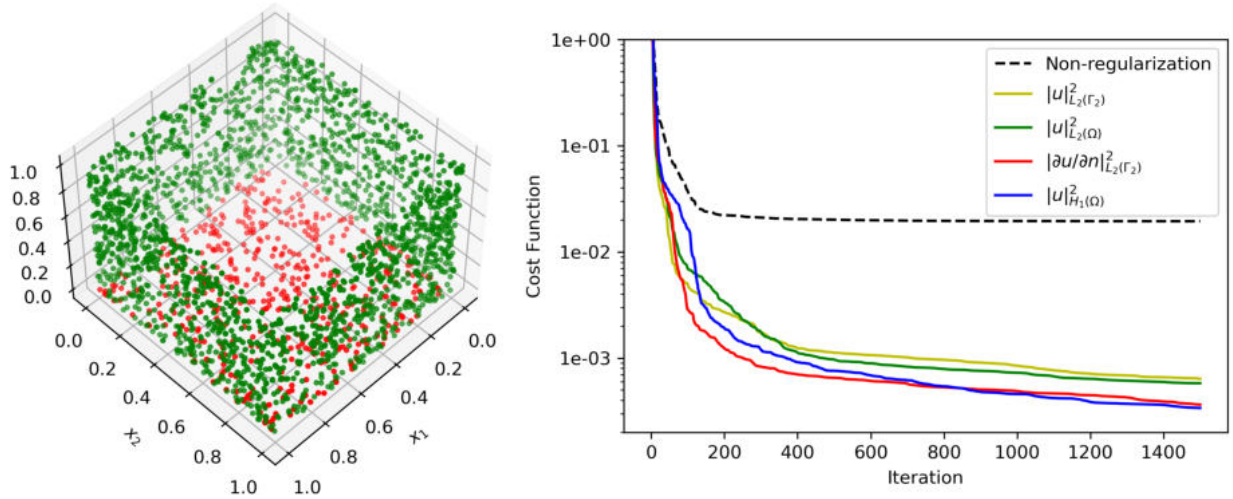
$$\Gamma_5 := \{x | 0 \leq x_1 \leq 0.5, x_2 = 0, 0 \leq x_3 \leq 1\}, \quad \Gamma_6 := \{x | 0 \leq x_1 \leq 0.5, x_2 = 0.5, 0 \leq x_3 \leq 1\}.$$

Xét bài toán Cauchy sau

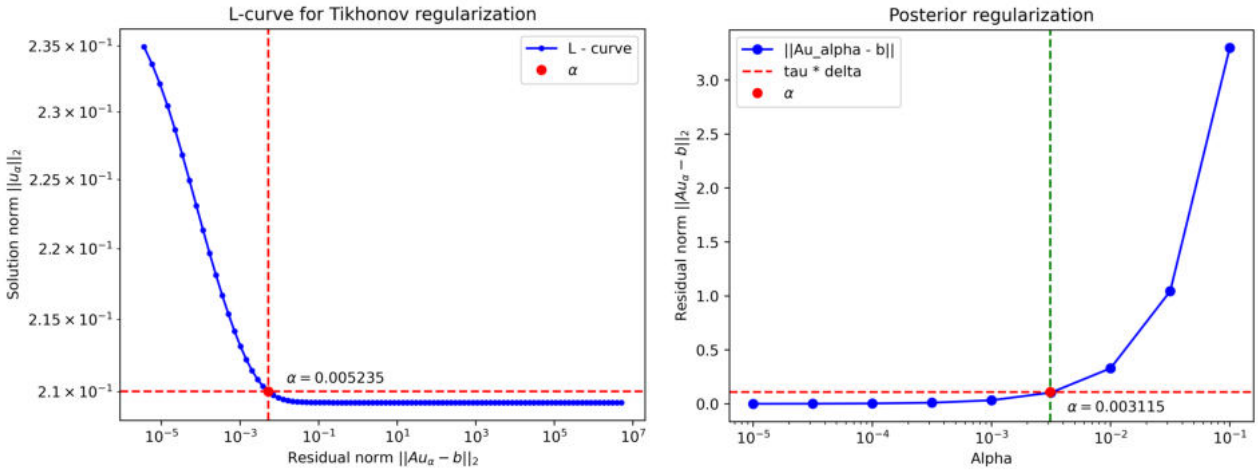
$$\begin{cases} u_{x_1x_1} + u_{x_2x_2} + u_{x_3x_3} + \mathcal{L}[u(x_1,x_2,x_3)] = S(x_1,x_2,x_3) + \mathcal{N}[u(x_1,x_2,x_3)], & \mathbf{x} \text{ trong } \Omega \\ u(x_1,x_2,0) = \frac{1}{x_1+x_2+1} & \mathbf{x} \text{ trên } \Gamma_1 \\ \frac{\partial u(x_1,x_2,x_3)}{\partial \mathbf{n}} = \nabla(\frac{1}{x_1+x_2+x_3+1}) * \mathbf{n} & \mathbf{x} \text{ trên } \Gamma_1 \\ u(x_1,x_2,x_3) = \frac{1}{x_1+x_2+x_3+1} & \mathbf{x} \text{ trên } \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6 \end{cases}$$

Với nghiệm chính xác của bài toán là $u(x_1,x_2,x_3) = \frac{1}{x_1+x_2+x_3+1}$, $\mathcal{L} = 0$, $\mathcal{N} = u^3$ và $S(x_1,x_2,x_3) = 0$.

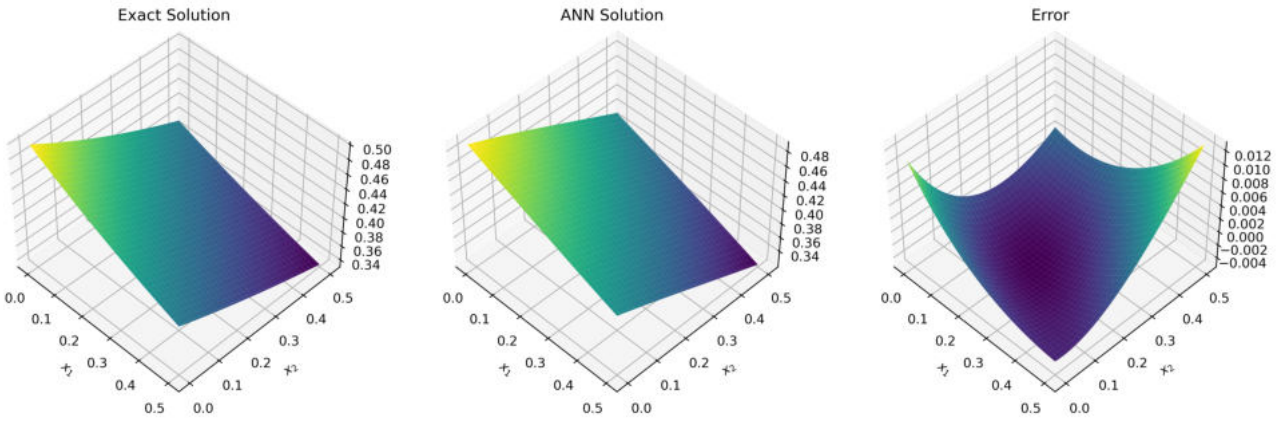
Trong ví dụ này các điểm dữ liệu trên miền Ω và biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$ được tạo ra bằng cách lấy mẫu ngẫu nhiên từ phân phối đều với kích thước kích thước dữ liệu đào tạo $N_r = 3500$ điểm trên miền Ω và $N_b = 1000$ điểm trên biên $\Gamma_1, \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$.



Hình 3.55: Biên Γ_1(màu đỏ), $\Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \cup \Gamma_6$(màu xanh lá). Lịch sử hội tụ của thuật toán L-BFGS qua mỗi bước lặp.



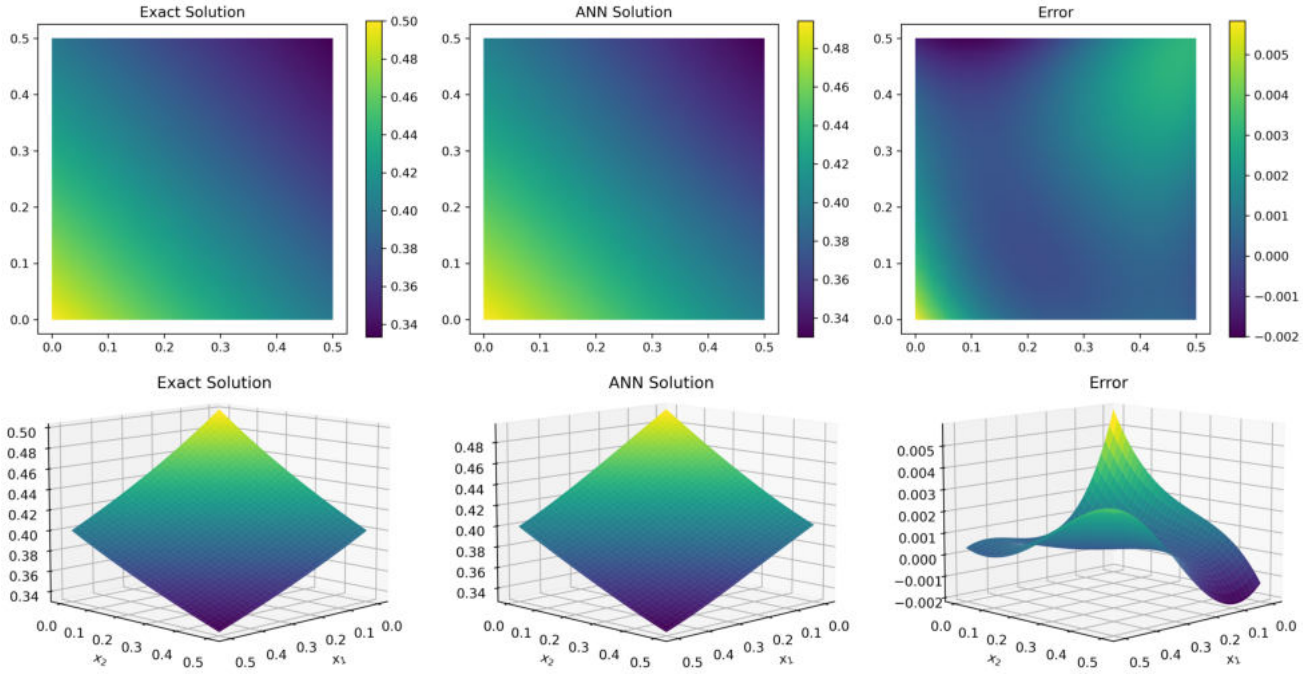
Hình 3.56: Đồ thị minh họa phương pháp L-curve và phương pháp hậu nghiệm.



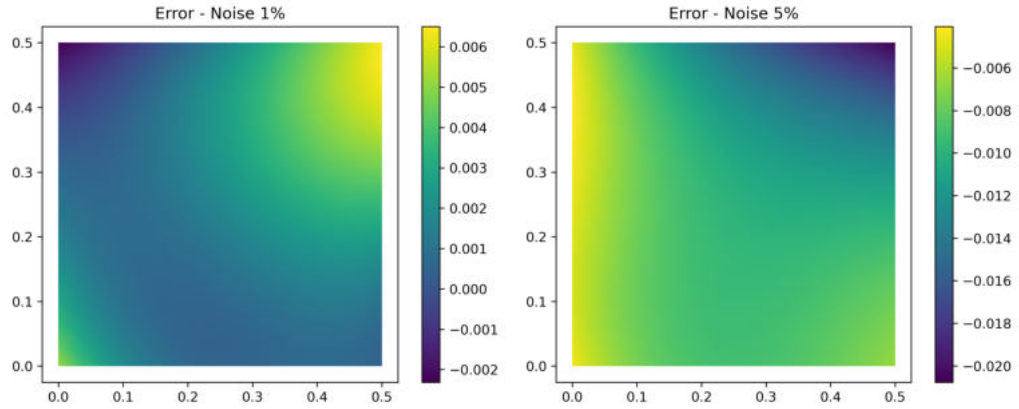
Hình 3.57: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.005235$ chọn theo phương pháp L-curve.

Nhiễu (%)	0.1%	1%	5%
Sai số - L^2	3.687912e-03	7.465824e-02	2.272312e-02

Bảng 3.26: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.005235$ chọn theo phương pháp L-curve.



Hình 3.58: Hình minh họa nghiệm chính xác, nghiệm số ANN và sai số trong trường hợp nhiễu 0.1%. Tham số chỉnh hóa $\alpha = 0.003115$ chọn theo phương pháp hậu nghiệm.



Hình 3.59: Sai số giữa nghiệm chính xác và nghiệm số ANN trong trường hợp nhiễu 1%, 5%. Tham số chỉnh hóa α chọn theo phương pháp hậu nghiệm.

Nhiễu (%)	0.1%	1%	5%
α	0.003115	0.008123	0.028563
Sai số - L^2	1.433903e-03	2.479424e-03	1.016865e-02

Bảng 3.27: Sai số L^2 trong các trường hợp nhiễu $0.1\%, 1\%, 5\%$. Tham số chỉnh hóa $\alpha = 0.003115$ chọn theo phương pháp hậu nghiệm.

L-BFGS	Không chỉnh	$\lvert\bar{u}\rvert^2_{L_2(\Gamma 2)}$	$\lvert\bar{u}\rvert^2_{L_2(\Omega)}$	$\lvert\frac{\partial\bar{u}}{\partial n}\rvert^2_{L_2(\Gamma 2)}$	$\lvert\bar{u}\rvert^2_{H_1(\Omega)}$
L	5	5	5	5	5
N	10	10	10	10	10
Nhiễu (%)	0.1%	0.1%	0.1%	0.1%	0.1%
α	0	0.005235	0.005235	0.005235	0.005235
Bước lặp	1500	1500	1500	1500	1500
Thời gian/Bước lặp	0.0606s	0.0713s	0.0893s	0.0725s	0.0891s
Hàm mất mát (Loss)	1.94e-02	6.38e-04	5.79e-04	3.64e-04	3.38e-04

Bảng 3.28: Kết quả số cho bài bằng toán thuật toán L-BFGS.

3.5. Nhận xét

Trong chương này, chúng tôi đã trình bày các kết quả cụ thể như sau:

1. Trình bày các ví dụ để minh họa cụ thể phương pháp cho bài toán Cauchy tuyến tính, cho trường hợp hai chiều (2D) và ba chiều (3D).

2. Trình bày các ví dụ để minh họa cụ thể phương pháp cho bài toán Cauchy phi tuyến, cho trường hợp hai chiều (2D) và ba chiều (3D).

3. Trình bày ví dụ so sánh hai thuật toán ADAM và L-BFGS (xem ví dụ 3.8).

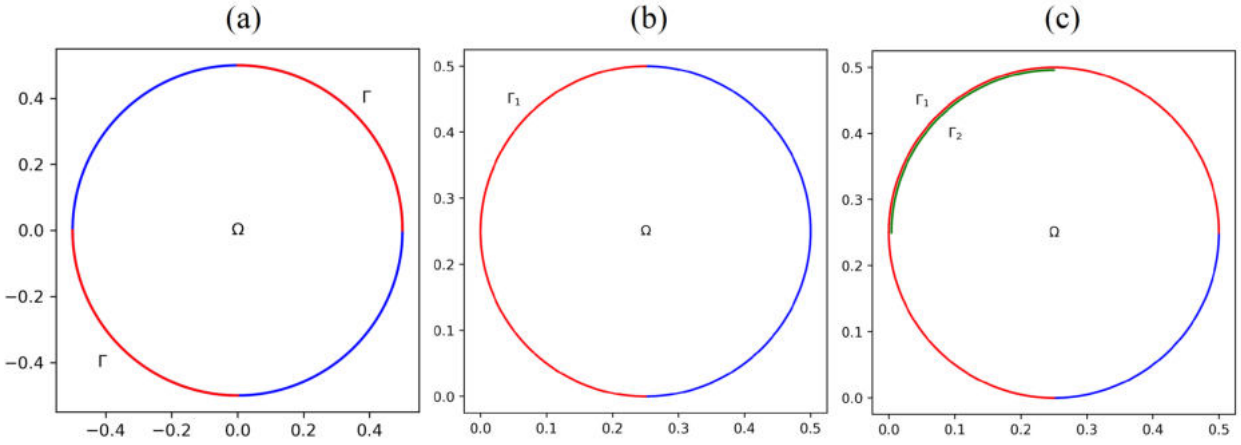
Các ví dụ số thể hiện sự hiệu quả của việc kết hợp phương pháp chỉnh hóa Tikhonov với mạng neuron nhân tạo (ANN) trong việc giải bài toán Cauchy. Kết quả cho thấy tính ổn định của phương pháp khi đối mặt với nhiễu ở mức 0.1%, 1%, 5%, và sai số vẫn duy trì ở khoảng từ 10^{-2} đến 10^{-3} cho tất cả các trường hợp.

Trong toàn bộ các ví dụ số, chúng tôi chọn tham số chỉnh hóa α theo phương pháp L-curve và phương pháp hậu nghiệm. Phương pháp L-curve là một phương pháp đồ thị được sử dụng để xác định giá trị tối ưu của tham số chỉnh hóa α, cách tiếp cận này không phụ thuộc vào mức độ nhiễu (noise level) trong dữ liệu. Trong thực tế, chúng ta thường không biết trước được mức độ nhiễu (noise level), do đó phương pháp L-curve thường được ưa chuộng và áp dụng rộng rãi trong các bài toán thực tế.

Ngược lại, trong phương pháp hậu nghiệm (posterior regularization), mức độ nhiễu (noise level) trong dữ liệu trở thành một yếu tố quan trọng để xác định tham số chỉnh hóa α. Phương pháp này thường cho kết quả tốt hơn khi mức độ nhiễu được xác định chính xác và đáng tin cậy.

Tóm lại, cả hai phương pháp L-curve và phương pháp hậu nghiệm đều có thể được sử dụng để chọn tham số chỉnh hóa trong các bài toán chỉnh hóa. Việc lựa chọn phương pháp tùy thuộc vào bối cảnh và mục tiêu của từng bài toán cụ thể. Tuy nhiên, để đạt được kết quả tính toán tốt, chúng ta nên kết hợp cả hai phương pháp này. Bằng cách sử dụng cả phương pháp L-curve và phương pháp hậu nghiệm, chúng ta có thể tăng khả năng đánh giá và lựa chọn một giá trị α phù hợp cho từng bài toán chỉnh hóa.

Để minh chứng cho tính hiệu quả khi áp dụng phương pháp chỉnh hóa Tikhonov kết hợp với ANN trong giải quyết bài toán Cauchy, chúng tôi có vài nhận xét cho ví dụ 3.1, 3.3, 3.6 như sau



Trong nghiên cứu của Li và Hu [30], không sử dụng chỉnh hóa, họ đã áp dụng điều kiện biên đối xứng như được biểu diễn trong hình (a) cho ví dụ 1(xem [30]). Việc này giúp cho bài toán trở nên dễ dàng hơn trong quá trình tính toán. Tuy nhiên, nếu không sử dụng chỉnh hóa và chọn điều kiện biên như hình (b) và (c), phương pháp của Li và Hu [30] không thể đạt kết quả tốt. Trong nghiên cứu của chúng tôi, nhờ áp dụng phương pháp chỉnh hóa Tikhonov kết hợp với mạng ANN, chúng tôi có khả năng giải quyết bài toán một cách dễ dàng và đạt được kết quả tốt với thời gian tính toán hiệu quả và nhanh chóng (xem ví dụ 3.1, 3.3, 3.6).

KẾT LUẬN VÀ KIẾN NGHỊ

Luận văn đã tập trung nghiên cứu về đề tài: **"Giải bài toán Cauchy cho một số phương trình đạo hàm riêng bằng mạng neural nhân tạo"** đã trình bày được các kết quả sau:

1. Giới thiệu tổng quan về cấu trúc của mạng neuron nhân tạo (ANN) và ứng dụng ANN để giải các bài toán liên quan đến phương trình đạo hàm riêng.

2. Trình bày một phương pháp mới giải bài toán Cauchy đặt không chỉnh cho phương trình elliptic và parabolic thông qua việc kết hợp phương pháp chỉnh hóa Tikhonov với mạng neuron nhân tạo. Để đạt được hiệu suất tốt, chúng tôi sử dụng thuật toán ADAM kết hợp với L-BFGS, một công cụ mạnh mẽ, để huấn luyện mạng neuron. Tham số chỉnh hóa α được chọn theo phương pháp L-curve và phương pháp hậu nghiệm. Cuối cùng, trình bày về tính hội tụ của phương pháp này.

3. Trình bày ví dụ hai chiều (2D), ba chiều (3D) cho cả hai trường hợp tuyến tính và phi tuyến. Kết quả số thu được từ những ví dụ cho thấy phương pháp mới trong luận văn là hiệu quả. Điều này tạo cơ sở cho nghiên cứu và ứng dụng mạng neuron nhân tạo kết hợp với chỉnh hóa trong việc giải các bài toán ngược cho phương trình đạo hàm riêng.

Các kết quả trong luận văn là mới và đang tiếp tục hoàn thiện để viết thành một bài báo khoa học.

Hướng nghiên cứu tiếp theo của luận văn sẽ bao gồm các khía cạnh sau:

1. Chúng tôi sẽ tiếp tục nghiên cứu và hoàn thiện lý thuyết kết hợp phương pháp chỉnh hóa Tikhonov và ANN để nâng cao khả năng giải bài toán Cauchy và các bài toán phương trình đạo hàm riêng khác.

2. Nghiên cứu và cải thiện lý thuyết liên quan đến tính ổn định và hội tụ của phương pháp giải bài toán Cauchy bằng ANN, để tăng độ tin cậy của mô hình học máy.

3. Nghiên cứu các phương pháp chỉnh hóa khác kết hợp với ANN để giải bài toán

Cauchy. Điều này có thể tìm ra phương pháp tiếp cận mới hiệu quả hơn.

Hướng nghiên cứu mở rộng về việc áp dụng mạng neuron nhân tạo để giải phương trình đạo hàm riêng:

1. Phát triển lý thuyết chỉnh hóa áp dụng cho ANN vì chỉnh hóa giúp tăng độ tin cậy của mô hình học máy. Nó có thể giúp kiểm soát hiện tượng quá khớp (overfitting) bằng cách giảm độ phức tạp của mô hình, đảm bảo rằng nó hoạt động tốt trên dữ liệu mới.

2. Nghiên cứu tính ổn định, tính hội tụ và độ tin cậy của phương pháp giải phương trình đạo hàm riêng bằng ANN trong các bài toán thực tế.

3. Có thể tập trung nghiên cứu ứng dụng ANN vào giải nhiều loại phương trình đạo hàm riêng khác nhau như: Phương trình đạo hàm riêng phi tuyến, phương trình dạng Hamilton-Jacobi, hoặc các phương trình trong lĩnh vực vật lý học.

4. Nghiên cứu cách tích hợp ANN vào các phương pháp truyền thống hoặc kết hợp với xác suất, thống kê để tạo ra các kết quả chính xác và hiệu quả hơn. Điều này có thể áp dụng trong các bài toán với điều kiện biên phức tạp hoặc cần độ chính xác cao.

5. Nghiên cứu tối ưu hóa cấu trúc ANN để giảm thiểu số lượng tham số và tăng khả năng tự học của mạng. Điều này giúp cải thiện hiệu suất và giảm thời gian đào tạo.

6. Nghiên cứu áp dụng ANN để giải các bài toán cụ thể trong các lĩnh vực như hình ảnh y học, dự đoán thời tiết, mô phỏng không gian và thời gian, hoặc tối ưu hóa quy trình sản xuất.

TÀI LIỆU THAM KHẢO

[1] Colli Franzone P., Magenes E., 1979, On the inverse potential problem of electrocardiology, *Calcolo*, 16, pp. 459-538.

[2] Colli Franzone P., Guerri L., Tentoni S., Viganotti C., Baruffi S., Spaggiari S., Taccardi B., 1985, A mathematical procedure for solving the inverse potential problem of electrocardiography, Analysis of the time-space accuracy from in vitro experimental data, *Math. Biosci*, 77, no. 1-2, 353-396.

[3] Colli Franzone P., Guerri L., Taccardi B., Viganotti C., 1985, Finite element approximation of regularized solutions of the inverse potential problem of electrocardiography and applications to experimental data., *Calcolo*, 22, no. 1, 91-186.

[4] Johnson C. R., 1997, Computational and numerical methods for bioelectric field problems, *Critical Reviews in Biomedical Engineering*, 25, no. 1 , pp. 1-81.

[5] Hadamard J., 1902, Sur les problèmes aux derivèes partielles et leur signification physique, *Bull. Univ. Princeton*, 13, pp. 49-32.

[6] Hadamard J., 1923, Lectures on Cauchy's Problem in Linear Partial Differential Equations, Yale Univ. Press.

[7] F. Ginsberg, 1963, On the Cauchy problem for the one-dimensional heat equation,*Math. Comp.*, 17, pp. 257-269.

[8] Tikhonov A. N., 1943, On the stability of inverse problems, *Dokl. Akad. Nauk SSSR*, 39, No. 5, pp. 195-198.

[9] Tikhonov A. N. and Arsenin V. Y., 1977, Solution of Ill-posed Problems, Wiley, New York (*in Russian*).

[10] Lavrent'ev M. M., 1955, On Cauchy's problem for Laplace's equation, *Dokl. Akad. Nauk SSSR* (N.S.) 102, pp. 205-206 (*in Russian*).

[11] Lavrent'ev M. M., 1956, On the Cauchy problem for Laplace equation, *Izv. Akad. Nauk SSSR. Ser. Mat.*, 120, 819-842 (*in Russian*).

[12] John F., 1955, A note on improper problems in partial differential equations, *Comm. Pure Appl. Math.*, 8, pp. 494-495.

[13] John F., 1960, Continuous dependence on the data for solutions of partial differential equations with a predescribed bound, *Comm. Pure Appl.*, 13, pp. 551-586.

[14] Tikhonov A. N., 1963, On the solution of ill-posed problems and the method of regularization, *Dokl. Akad. Nauk SSSR*, 151, pp. 501-504 (*in Russian*).

[15] Isakov V., 1998, Inverse Problems for Partial Differential Equations, Springer-Verlag, New York.

[16] Baumeister J., 1987. Stable solution of inverse problem. Vieweg & Sohn, Braunschweig.

[17] Lavrent'ev M. M., Romanov V. G. and Shishat-skii S. P., 1986. Ill-posed Problems of Mathematical Physics and Analysis. Transl. Math. Monographs, Amer. Math. Soc. Providence, R. I..

[18] Maziar Raissi, Paris Perdikaris, George Em Karniadakis, 2019, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, *Journal of Computational Physics*, 378, pp. 686-707, *DOI: 10.1016/j.jcp.2018.10.045*.

[19] Hao Xu, Dongxiao Zhang, Nanzhe Wang, 2021, Deep-learning based discovery of partial differential equations in integral form from sparse and noisy data. *Journal of Computational Physics*, 445, pp. 110592.

[20] I.E. Lagaris, A. Likas, D.I. Fotiadis, 1998, Artificial neural networks for solving ordinary and partial differential equations, *IEEE Trans. Neural Netw*, 9 (5), 987–1000.

[21] I.E. Lagaris, A.C. Likas, D.G. Papageorgiou, 2000, Neural-network methods for boundary value problems with irregular boundaries, *IEEE Trans. Neural Netw*, 11 (5), 1041–1049.

[22] A. Malek, R.S. Beidokhti, 2006, Numerical solution for high order differential equations using a hybrid neural network-optimization method, *Appl. Math. Comput*, 183 (1), 260–271.

[23] L.P. Aarts, P. van der Veer, 2001, Neural network method for solving partial differential equations, *Neural Process. Lett*, 14 (3), 261–271.

[24] J. Sirignano, K. Spiliopoulos, 2018, DGM: a deep learning algorithm for solving partial differential equations, *J. Comput. Phys*, 375, 1339–1364.

[25] G. Pang, L. Lu, G.E. Karniadakis, 2019, fPINNs: fractional physics-informed neural networks, *SIAM J. Sci. Comput*, 41 (4), A2603–A2626, doi:10.1137/18M1229845.

[26] D. Zhang, L. Lu, G.E. Karniadakis, 2019, Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems, *J. Comput. Phys*, 399, 108925.

[27] K. Hornik, 1989, Multilayer feedforward networks are universal approximators, *Neural Netw*, 2, 359–366.

[28] K. Hornik, 1991, Approximation capabilities of multilayer feedforward networks, *Neural Netw*, 4, 251–257.

[29] Maziar Raissi, Paris Perdikaris, George Em Karniadakis, 2017, Physics Informed Deep Learning (Part I-II): Data-driven Solutions of Nonlinear Partial Differential Equations, *arXiv 1711.10561*.

[30] Yixin Li, Xianliang Hu, 2022, Artificial neural network approximations of Cauchy inverse problem for linear PDEs, *Applied Mathematics and Computation*, Volume 414, 1 February, 126678.

[31] Hinton, G.E., Rumelhart, D.E., & Williams, R.J., 1986, Learning representations by back-propagating errors, *Nature*, 323, 533-536.

[32] Evans, L. C., 2010, Partial Differential Equations, *American Mathematical Society.*

[33] A. Kirsch, 2011, An Introduction to the Mathematical Theory of Inverse Problems, 2nd edn, Applied Mathematical Sciences, Vol. 120. Springer, Berlin .

[34] Knabner, Vessella, 1988, The optimal stability estimate for some ill-posed Cauchy problems for a parabolic equation, *Math. Methods in the App. Sci*, 10, pp. 575-583.

[35] Catherine F. Higham and Desmond J. Higham, 2019, Deep Learning: An Introduction for Applied Mathematicians, *Society for Industrial and Applied Mathematics (SIAM)*, 61(4), pp. 860-891.

[36] O.A. Ladyzhenskaya, 1985, The Boundary Value Problems of Mathematical Physics, *Applied Mathematical Sciences.*

[37] S. P. Shishatskii M. M. Lavrentev, V. G. Romanov, 1986 Ill-Posed Problems of Mathematical Physics and Analysis, *American Mathematical Society.*

[38] Dinh Nho Hao, 1995, Methods for inverse heat conduction problems, *Habilitation Thesis.*

[39] Dinh Nho Hào và Pham Minh Hien, 2003, Stability results for the Cauchy problem for the Laplace equation in a strip, Inverse Problems 19, pp. 833-844.

[40] D.A. amd P. Besala., 1966, Uniqueness of solutions of the Cauchy problem for parabolic equations, *J. Math. Anal. Appl.,* 13, pp. 516-526.

[41] Hansen, P. C., 1992, The L-curve and its use in the numerical treatment of inverse problems, *The Royal Society.* doi:10.1098/rspa.1992.0073.

[42] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, Liwei Wang, 2017, The Expressive Power of Neural Networks: A View from the Width, *Advances in Neural Information Processing Systems (NIPS)*, 30.

[43] Boris Hanin, 2019, Universal Function Approximation by Deep Neural Nets with Bounded Width and ReLU Activations, *Mathematics*, 7(10), 992.

[44] Dong C. Liu, Jorge Nocedal, 1989, On the limited memory BFGS method for large scale optimization, *Mathematical Programming*, volume 45, 503–528.

[45] K. Hornik, 1991, Approximation capabilities of multilayer feedforward networks, *Neural Networks*, 4:251-257.

PHỤ LỤC

Trong phần phụ lục này, chúng tôi trình bày một số tính toán về đạo hàm và lan truyền ngược dựa trên quy tắc đạo hàm hợp (chain rule).

3.6. Lan truyền ngược với ANN

Lan truyền ngược là quá trình tính đạo hàm trong mạng neuron nhân tạo (ANN). Để tính lan truyền ngược với đạo hàm $\left(\frac{\partial y^{L+1}}{\partial w^l}, \frac{\partial y^{L+1}}{\partial b^l}\right)$, chúng ta có thể tính đạo hàm của $\mathbf{z}^l$ và y^l. Theo quy tắc đạo hàm hợp, ta có:

$$\begin{aligned}\frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{w}^l} &= \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{z}^l} * \frac{\partial \mathbf{z}^l}{\partial \mathbf{w}^l} = \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{z}^l} * \mathbf{y}^{l-1},\\ \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{b}^l} &= \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{z}^l} * \frac{\partial \mathbf{z}^l}{\partial \mathbf{b}^l} = \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{z}^l}.\end{aligned} \tag{3.7}$$

Chúng ta định nghĩa $\delta^l := \frac{\partial y^{L+1}}{\partial \mathbf{z}^l}, l = 1, 2, \ldots, L+1$. Từ phương trình (3.7), ta có:

$$\begin{aligned}&\delta^{L+1} = \sigma'_{L+1}\left(\mathbf{z}^{L+1}\right),\\ &\delta^l = \left(\mathbf{w}^{l+1}\right)^T * \delta^{l+1} \odot \sigma'_l\left(\mathbf{z}^l\right), \quad l = 1, 2, \ldots, L;\end{aligned}$$

ở đây, kí hiệu $\odot$ biểu thị phép nhân các phần tử tương ứng của hai vectơ.

3.7. Lan truyền ngược đạo hàm cấp 1 với ANN

Giả sử rằng đầu vào $\mathbf{x} \in \Omega \times \mathcal{T} \subset \mathbb{R}^d \times \mathbb{R}^1$, lan truyền ngược với đạo hàm $\left(\frac{\partial^2 y^{L+1}}{\partial w^1 \partial x_i}, \frac{\partial^2 y^{L+1}}{\partial b^l \partial x_i}\right)$ như sau:

$$\frac{\partial^2 \mathbf{y}^{L+1}}{\partial \mathbf{x}_i \partial \mathbf{w}^l} = \frac{\partial \delta^l}{\partial \mathbf{x}_i} * \left(\mathbf{y}^{l-1}\right)^T + \delta^l * \frac{\partial \left(\mathbf{y}^{l-1}\right)^T}{\partial \mathbf{x}_i}, \tag{3.8}$$

$$\frac{\partial^2 \mathbf{y}^{L+1}}{\partial \mathbf{x}_i \partial \mathbf{b}^l} = \frac{\partial \delta^l}{\partial \mathbf{x}_i}, \tag{3.9}$$

trong đó $\frac{\partial \delta^l}{\partial \mathbf{x}_i}$ được tính như sau:

$$\begin{aligned}\frac{\partial \delta^{L+1}}{\partial \mathbf{x}_i} &= \sigma''_{L+1}\left(\mathbf{z}^{L+1}\right) \odot \frac{\partial \mathbf{z}^{L+1}}{\partial x_i}\\ \frac{\partial \delta^l}{\partial \mathbf{x}_i} &= \left(\mathbf{w}^{l+1}\right)^T * \frac{\partial \delta^{l+1}}{\partial x_i} \odot \sigma'_l\left(\mathbf{z}^l\right) + \left(\mathbf{w}^{l+1}\right)^T * \delta^{l+1} \odot \sigma''_l\left(\mathbf{z}^l\right) \odot \frac{\partial \mathbf{z}^l}{\partial x_i}.\end{aligned}$$

Công thức $\frac{\partial y^l}{\partial x_i}$ và $\frac{\partial z^l}{\partial x_i}$ trong các phương trình trên có thể biểu diễn cho từng lớp như sau:

$$\begin{aligned}
&\frac{\partial \mathbf{z}^1}{\partial x_i} = \mathbf{w}^1 \\
&\frac{\partial \mathbf{z}^{l+1}}{\partial x_i} = \mathbf{w}^{l+1} * \frac{\partial \mathbf{z}^l}{\partial x_i} \odot \sigma_l' \left(\mathbf{z}^l\right) \\
&\frac{\partial \mathbf{y}^l}{\partial x_i} = \frac{\partial \mathbf{z}^l}{\partial x_i} \odot \sigma_l' \left(\mathbf{z}^l\right).
\end{aligned}$$

Khi $\Omega \subset \mathbb{R}^d, d > 3$, để thuận tiện trong tính toán ma trận, vectơ $\mathbf{f} = [f_1, f_2, \ldots, f_m]$, ma trận Jacobi của $\mathbf{f}$ được định nghĩa là:

$$G(\mathbf{f}) = \begin{bmatrix}
\frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_d} & \frac{\partial f_1}{\partial t} \\
\frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_d} & \frac{\partial f_2}{\partial t} \\
\vdots & \vdots & \ddots & \vdots & \\
\frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_d} & \frac{\partial f_m}{\partial t}
\end{bmatrix},$$

và đối với một vectơ $\mathbf{u} = [u_1, u_2, \ldots, u_n]$, các ma trận đường chéo được định nghĩa như sau:

$$\operatorname{diag}(\mathbf{u}) = \begin{bmatrix}
u_1 & 0 & \ldots & 0 \\
0 & u_2 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & u_n
\end{bmatrix}.$$

Với định nghĩa trên, ta có các công thức trong dạng ma trận như sau:

$$\begin{aligned}
&G\left(\delta^{L+1}\right) = \operatorname{diag}\left(\sigma_{L+1}''\left(\mathbf{z}^{L+1}\right)\right) * G\left(\mathbf{z}^{L+1}\right), \\
&G\left(\delta^l\right) = \operatorname{diag}\left(\sigma_l'\left(\mathbf{z}^l\right)\right) * \left(\mathbf{w}^{l+1}\right)^T * G\left(\delta^{l+1}\right) \\
&\qquad\qquad + \operatorname{diag}\left(\sigma_l''\left(\mathbf{z}^l\right)\right) * \left(\mathbf{w}^{l+1}\right)^T * \delta^{l+1} \odot G\left(\mathbf{z}^l\right), \\
&\frac{\partial \delta^l}{\partial \mathbf{x}_i} = G\left(\delta^l\right)[i],
\end{aligned} \tag{3.10}$$

trong đó:

$$
\begin{aligned}
&G\left(\mathbf{z}^1\right) = \mathbf{w}^1 * I_{d+1},\\
&G\left(\mathbf{z}^l\right) = \mathbf{w}^l * \operatorname{diag}\left(\sigma_{l-1}\left(\mathbf{z}^{l-1}\right)\right) * G\left(\mathbf{z}^{l-1}\right), \quad l = 2, 3, \ldots, L+1,\\
&G\left(\mathbf{y}^l\right) = \operatorname{diag}\left(\sigma_l'\left(\mathbf{z}^l\right)\right) * G\left(\mathbf{z}^l\right), \quad l = 1, 2, \ldots, L+1,\\
&I_d \text{ là ma trận đơn vị } d \times d.
\end{aligned}
$$

3.8. Lan truyền ngược đạo hàm cấp 2 với ANN

Giả sử đầu vào $\mathbf{x} \in \Omega \times \mathcal{T} \subset \mathbb{R}^d \times \mathbb{R}^1$ và lan truyền ngược với đạo hàm $\left(\frac{\partial^3 y^{L+1}}{\partial \mathbf{w}^1 \partial^2 x_i}, \frac{\partial^3 y^{L+1}}{\partial b^l \partial^2 x_i}\right)$ được tính như sau:

$$
\begin{aligned}
&\frac{\partial^3 \mathbf{y}^{L+1}}{\partial \mathbf{w}^1 \partial^2 x_i} = \frac{\partial^2 \delta^l}{\partial x_i^2} * \left(\mathbf{y}^{l-1}\right)^T + 2\frac{\partial \delta^l}{\partial x_i} * \frac{\partial \left(\mathbf{y}^{l-1}\right)^T}{\partial x_i} + \delta^l * \frac{\partial^2 \left(\mathbf{y}^{l-1}\right)^T}{\partial x_i^2},\\
&\frac{\partial^3 \mathbf{y}^{L+1}}{\partial \mathbf{b}^l \partial^2 x_i} = \frac{\partial^2 \delta^l}{\partial x_i^2},
\end{aligned}
\tag{3.11}
$$

trong đó $\frac{\partial^2 \delta^l}{\partial x_i^2}$ được tính bằng công thức:

$$
\begin{aligned}
\frac{\partial^2 \delta^{L+1}}{\partial x_i^2} &= \sigma_{L+1}'''\left(\mathbf{z}^{L+1}\right) \odot \left(\frac{\partial \mathbf{z}^{L+1}}{\partial x_i}\right)^2,\\
\frac{\partial^2 \delta^l}{\partial x_i^2} &= \left(\mathbf{w}^{l+1}\right)^T * \left(\frac{\partial^2 \delta^{l+1}}{\partial x_i^2} \odot \sigma_l'\left(\mathbf{z}^l\right) + 2\frac{\partial \delta^{l+1}}{\partial x_i} \odot \sigma_l''\left(\mathbf{z}^l\right) \odot \frac{\partial \mathbf{z}^l}{\partial x_i}\right)\\
&\quad + \left(\mathbf{w}^{l+1}\right)^T * \delta^l \odot \left(\sigma_l'''\left(\mathbf{z}^l\right) \odot \left(\frac{\partial \mathbf{z}^l}{\partial x_i}\right) + \sigma_l''\left(\mathbf{z}^l\right) \odot \frac{\partial^2 \mathbf{z}^l}{\partial x_i^2}\right).
\end{aligned}
$$

Công thức phân tích của $\frac{\partial^2 y^l}{\partial x_i^2}$ và $\frac{\partial^2 z^l}{\partial x_i^2}$ trong các phương trình trên có thể biểu diễn cho từng lớp như sau:

$$
\begin{aligned}
&\frac{\partial^2 \mathbf{z}^1}{\partial x_i^2} = 0,\\
&\frac{\partial^2 \mathbf{z}^{l+1}}{\partial x_i^2} = \mathbf{w}^{l+1} * \left(\frac{\partial^2 \mathbf{z}^l}{\partial x_i^2} \odot \sigma_l'\left(\mathbf{z}^l\right) + \left(\frac{\partial \mathbf{z}^l}{\partial x_i}\right)^2 \odot \sigma_l''\left(\mathbf{z}^l\right)\right),\\
&\frac{\partial^2 \mathbf{y}^l}{\partial x_i^2} = \frac{\partial^2 \mathbf{z}^l}{\partial x_i^2} \odot \sigma_l'\left(\mathbf{z}^l\right) + \left(\frac{\partial \mathbf{z}^l}{\partial x_i}\right)^2 \odot \sigma_l''\left(\mathbf{z}^l\right).
\end{aligned}
$$

Khi $\Omega \subset \mathbb{R}^d, d > 3$, để thuận tiện khi tính toán ở dạng ma trận, vectơ $\mathbf{u}, G^2(\mathbf{u})$ được định nghĩa là:

$$G^2(\mathbf{u}) = \begin{bmatrix} \frac{\partial^2 u_1}{\partial x_1^2} & \frac{\partial^2 u_1}{\partial x_2^2} & \cdots, & \frac{\partial^2 u_1}{\partial x_d^2} \\ \frac{\partial^2 u_2}{\partial x_1^2} & \frac{\partial^2 u_2}{\partial x_2^2} & \cdots, & \frac{\partial^2 u_2}{\partial x_d^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 u_n}{\partial x_1^2} & \frac{\partial^2 u_n}{\partial x_2^2} & \cdots, & \frac{\partial^2 u_n}{\partial x_d^2} \end{bmatrix}.$$

Ma trận của đạo hàm cấp 2 được viết như sau:

$$\begin{aligned} G^2\left(\delta^{L+1}\right) =& \operatorname{diag}\left(\sigma_{L+1}^{(3)}\right) * \left(\nabla \mathbf{z}^{L+1} \odot \nabla \mathbf{z}^{L+1}\right) \\ &+ \operatorname{diag}\left(\sigma_{L+1}''\left(\mathbf{z}^{L+1}\right)\right) * G^2\left(\mathbf{z}^{L+1}\right), \end{aligned} \tag{3.12}$$

$$\begin{aligned} G^2\left(\delta^l\right) =& \operatorname{diag}\left(\sigma_l'\left(\mathbf{z}^l\right)\right) * \left(\mathbf{w}^{l+1}\right)^T * G^2\left(\delta^{l+1}\right) \\ &+ \operatorname{diag}\left(\sigma_l''\left(\mathbf{z}^l\right)\right) * \left(\mathbf{w}^{l+1}\right)^T * \left(2\nabla \delta^{l+1}\right) \odot \nabla \mathbf{z}^l \\ &+ \operatorname{diag}\left(\sigma_l^{(3)}\left(\mathbf{z}^l\right)\right) * \left(\mathbf{w}^{l+1}\right)^T * \delta^{l+1} \odot \left(\nabla \mathbf{z}^l \odot \nabla \mathbf{z}^l\right) \\ &+ \operatorname{diag}\left(\sigma_l''\left(\mathbf{z}^l\right)\right) * \left(\mathbf{w}^{l+1}\right)^T * \delta^{l+1} \odot J\left(\mathbf{z}^l\right), l = 1, 2, \ldots, L \end{aligned} \tag{3.13}$$

$$\frac{\partial^2 \delta^l}{\partial x_i^2} = G^2\left(\delta^l\right)[i]. \tag{3.14}$$

Ta thiết lập phương trình cho $G^2\left(\mathbf{z}^l\right)$ và $G^2\left(\mathbf{z}^l\right)$:

$$\begin{aligned} G^2\left(\mathbf{z}^1\right) =& \mathbf{w}^1 * O_{d+1}, \\ G^2\left(\mathbf{z}^l\right) =& \mathbf{w}^l * \left(\operatorname{diag}\left(\sigma_{l-1}''\left(\mathbf{z}^{l-1}\right)\right) * \left(\nabla \mathbf{z}^{l-1} \odot \nabla \mathbf{z}^{l-1}\right)\right. \\ &\left. + \operatorname{diag}\left(\sigma_{l-1}'\left(\mathbf{z}^{l-1}\right)\right) * G^2\left(\mathbf{z}^{l-1}\right)\right), \\ G^2\left(\mathbf{y}^l\right) =& \operatorname{diag}\left(\sigma_l''\left(\mathbf{z}^l\right)\right) * \left(\nabla \mathbf{z}^l \odot \nabla \mathbf{z}^l\right) + \operatorname{diag}\left(\sigma_l'\left(\mathbf{z}^l\right)\right) * G^2\left(\mathbf{z}^{L+1}\right)), \end{aligned}$$

trong đó O_d là ma trận không có cỡ $d \times d$.

Đầu tiên, chúng tôi đưa ra công thức lan truyền ngược cho bài toán (0.2), và bài toán (0.1) cũng tương tự.

Lan truyền ngược cho điều kiện biên Dirichlet và điều kiện ban đầu có thể dễ dàng thu được trong phương trình (3.7). Điều kiện biên Neumann và phương trình trạng thái được xét với bài toán (0.2).

3.9. Lan truyền ngược cho điều kiện biên Neumann

Trong bài toán phụ thuộc thời gian, chúng ta có:

$$G\left(\mathbf{y}^{L+1}\right) = \left[\begin{array}{cccc} \frac{\partial \mathbf{y}^{L+1}}{\partial x_1} & \frac{\partial \mathbf{y}^{L+1}}{\partial x_2} & \cdots & \frac{\partial \mathbf{y}^{L+1}}{\partial x_d} \quad \frac{\partial \mathbf{y}^{L+1}}{\partial t} \end{array}\right]. \tag{3.15}$$

Điều kiện biên Neumann được viết như sau:

$$\frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{n}} = \nabla \mathbf{y}^{L+1} * D^T(\mathbf{x}) = G\left(\mathbf{y}^{L+1}\right)[1:d] * D^T(\mathbf{x}), \tag{3.16}$$

trong đó $D(\mathbf{x}) = (\beta_1, \beta_2, \ldots, \beta_d)$ biểu thị hướng tại các điểm $\mathbf{x}$. Tương tự, lan truyền ngược của điều kiện biên Neumann được viết như sau:

$$\frac{\partial^2 \mathbf{y}^{L+1}}{\partial \mathbf{n} \partial \mathbf{w}^l} = \frac{\partial \nabla \mathbf{y}^{L+1}}{\partial \mathbf{w}^l} * D^T(\mathbf{x}) = \nabla \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{w}^l} * D^T(\mathbf{x}), \tag{3.17}$$

$$\frac{\partial^2 \mathbf{y}^{L+1}}{\partial \mathbf{n} \partial \mathbf{b}^l} = \frac{\partial \nabla \mathbf{y}^{L+1}}{\partial \mathbf{b}^l} * D^T(\mathbf{x}) = \nabla \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{b}^l} * D^T(\mathbf{x}). \tag{3.18}$$

Đặt $\text{sum}(A)$ là vectơ có phần tử tương ứng là tổng của các cột của A, được viết như sau:

$$\begin{aligned} \nabla \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{w}^l} * D^T(\mathbf{x}) = &\text{sum}\left(\text{diag}(D(\mathbf{x})) * G\left(\delta^l\right)[1:d]\right) * \left(\mathbf{y}^{l-1}\right)^T \\ &+ \delta^l * \text{sum}\left(\text{diag}(D(\mathbf{x})) * G\left(\mathbf{y}^{l-1}\right)^T [1:d]\right), \end{aligned} \tag{3.19}$$

$$\nabla \frac{\partial \mathbf{y}^{L+1}}{\partial \mathbf{b}^l} * D^T(\mathbf{x}) = \text{sum}\left(\text{diag}(D(\mathbf{x})) * G\left(\delta^l\right)[1:d]\right). \tag{3.20}$$

Vì vậy, chúng ta có thể tính toán lan truyền ngược cho điều kiện biên Neumann với các phương trình (3.16) đến (3.20).

3.10. Lan truyền ngược cho phương trình trạng thái

Rõ ràng $\left(\frac{\partial y^{L+1}}{\partial t}\right)$ trong bài toán (0.2) có thể được tính trực tiếp bằng phương trình tham chiếu (3.8) và (3.9). Điều quan trọng hơn để tính toán là lan truyền ngược cho toán tử $\mathcal{L}$. Có rất nhiều sự lựa chọn cho $\mathcal{L}$ và trong bài này, chúng tôi chỉ trình bày toán tử gradient ∇ và toán tử Laplace Δ. Nhiều toán tử bậc cao cũng tính tương tự.

Chúng ta biết rằng $\nabla \mathbf{y}^{L+1} = G\left(\mathbf{y}^{L+1}\right)[1:d]$, do đó lan truyền ngược có thể được tính trực tiếp theo phương trình (3.8). Sau đó, hãy xem xét toán tử Laplace. Chúng ta biết rằng

$$\Delta \mathbf{y}^{L+1} = \sum_{i=1}^{d} \frac{\partial^2 \mathbf{y}^{L+1}}{\partial x_i^2} \tag{3.21}$$

Theo phương trình (3.12), đặt $\Delta\mathbf{u} = \text{sum}\left(G^2(\mathbf{u})[1:d]\right)$. Định nghĩa $G_d(\mathbf{u}) := G^2(\mathbf{u})[1:d]$, ta có lan truyền ngược như sau

$$\begin{aligned}
\frac{\partial \Delta\mathbf{y}^{L+1}}{\partial \mathbf{w}^l} &= \sum_{i=1}^{d} \frac{\partial^3 \mathbf{y}^{L+1}}{\partial x_i^2 \partial \mathbf{w}^l} \\
&= \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2} \left(\delta^l * \left(\mathbf{y}^{l-1}\right)^T\right) \\
&= \left(\sum_{i=1}^{d} \frac{\partial^2 \delta^l}{\partial x_i^2}\right) * \left(\mathbf{y}^{l-1}\right)^T + 2\sum_{i=1}^{d} \frac{\partial \delta^l}{\partial x_i} * \frac{\partial \left(\mathbf{y}^{l-1}\right)^T}{\partial x_i} \\
&+ \delta^l * \left(\sum_{i=1}^{d} \frac{\partial^2 \left(\mathbf{y}^{l-1}\right)^T}{\partial x_i^2}\right) \\
&= \text{sum}\left(G_d\left(\delta^l\right)\right) * \left(\mathbf{y}^{l-1}\right)^T + 2\nabla\delta^l * \nabla\left(\mathbf{y}^{l-1}\right)^T \\
&+ \delta^l * \text{sum}\left(G_d^T\left(\mathbf{y}^{l-1}\right)\right)\Big)
\end{aligned} \tag{3.22}$$

$$\frac{\partial \Delta\mathbf{y}^{L+1}}{\partial \mathbf{b}^l} = \sum_{i=1}^{d} \frac{\partial^3 \mathbf{y}^{L+1}}{\partial x_i^2 \partial \mathbf{b}^l} = \text{sum}\left(G_d\left(\delta^l\right)\right). \tag{3.23}$$

Với tất cả các phương trình từ (3.21) đến (3.23), ta có thể tính toán lan truyền ngược cho phương trình trạng thái của nhiều toán tử $\mathcal{L}$.