UNIQUENESS OF THE SOLUTION FOR THE CONJUGATION PROBLEM OF THIRD-ORDER PARTIAL DIFFERENTIAL EQUATIONS AND ITS APPLICATION IN NEURAL NETWORK REGULARIZATION

by

Nurkasym K. ARKABAEV[a]***, Hanadi ABDELSALAM***[b*]***, Ibtisam DAQQA***[b]***, Salem ALKHALAF***[c]***, Hawa IBNOUF***[d]***, and Sayed Abdel KHALEK***[e]

[a] Osh State University, Osh, Kyrgyzstan
[b] College of Sciences and Human Studies,
Prince Mohammad Bin Fahd University, Al-Khobar, Saudi Arabia
[c] Department of Computer Engineering, College of Computer,
Qassim University, Buraydah, Saudi Arabia
[d] Department of Mathematics, College of Science, Qassim University, Saudi Arabia
[e] Department of Mathematics and Statistics, College of Science,
Taif University, Taif, Saudi Arabia

Original scientific paper
https://doi.org/10.2298/TSCI2501371A

This article investigates the uniqueness of the solution for the conjugation problem of third-order PDE with a characteristic line and its application in neural network regularization. A theorem on the uniqueness of the solution for the considered class of equations is proven. Based on the obtained theoretical results, a novel neural network regularization method is developed that accounts for the physical constraints of the problem. A comparison is made between the classical finite difference method and an innovative approach based on physics-informed neural networks. Numerical experiments demonstrated that the proposed method provides higher accuracy and better adherence to the physical constraints of the problem. The regularized neural networks exhibited lower mean square error, better compliance with conjugation conditions, and higher resilience to input data variations compared to classical methods and standard neural networks. The research opens new perspectives for integrating classical mathematical methods with modern machine learning technologies.

Key words: *PDE, conjugation problem, uniqueness of solution, neural networks, regularization, physics-informed neural networks, numerical methods*

Introduction

In recent decades, the theory of PDE and machine learning have evolved as largely independent fields. However, the increasing complexity of modern scientific and engineering challenges necessitates the integration of these disciplines. Our research aims to combine classical PDE analysis with innovative machine learning methods.

Conjugation problems for third-order PDE play a crucial role in modelling various physical processes, particularly those where medium parameters undergo sharp changes along specific lines or surfaces. The fundamental theory of third-order PDE, comprehensively presented in Padhi and Pati [1], provides the theoretical foundation for investigating such prob-

[*] Corresponding author, e-mail: habdelsalam@pmu.edu.sa

lems. The spectral representation methods for boundary value problems with third-order PDE, developed by Pelloni [2], have significantly advanced our understanding of the solution structure of these equations.

Contemporary approaches to solving third-order PDE encompass both classical numerical methods and innovative techniques based on artificial neural networks. Tawfiq and Ali [3] have demonstrated the efficacy of neural networks in solving third-order PDE, while Ashyralyev and Simsek [4] have developed a novel operator method for this class of equations.

A key aspect in analyzing these problems is the question of solution uniqueness. Uniqueness not only ensures the mathematical well-posedness of the model but also has significant practical implications. In the context of machine learning, as shown in the works of Arif *et al.* [5] and Avrutskiy [6], understanding the conditions for solution uniqueness can be utilized to develop more efficient and robust algorithms. In this paper, we investigate the conjugation problem for third-order PDE with a characteristic line. Our analysis builds upon results obtained in works [7-12], extending them to equations belonging to different types according to the Dzhuraev and Popelok classification [7]. Particular attention is paid to integrating classical PDE analysis methods with modern machine learning approaches, aligning with current trends in computational mathematics [6]. The considered approach combines rigorous mathematical analysis with contemporary machine learning methods, opening new perspectives for solving complex interdisciplinary problems. We anticipate that the results of this research will find applications in both theoretical developments and practical applications, ranging from physical process modelling to the creation of more reliable artificial intelligence algorithms.

This comprehensive framework integrates classical mathematical theory with modern computational techniques, leveraging the strengths of both approaches to address challenging problems in PDE. The significance of our work lies in its potential to bridge the gap between traditional analytical methods and contemporary machine learning approaches, offering new insights into the solution of complex mathematical problems arising in various scientific and engineering applications.

Problem statement

Consider the system of third-order PDE:

$$u_{xxy} + a(x,y)u_x + b(x,y)u_y = f(x,y,u), \;\; y > 0 \tag{1}$$

$$u_{yyy} + c(x,y)u_x + d(x,y)u_y = g(x,y,u), \;\; y < 0 \tag{2}$$

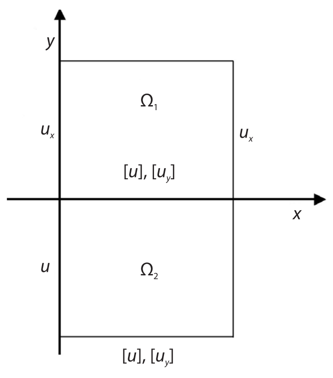

Figure 1. Task outline

where $u = u(x, y)$ is the unknown function and $a(x, y)$, $b(x, y)$, $c(x, y)$, $d(x, y)$ are given coefficients, and f and g are known functions.

Domain:

$$\Omega = \Omega_1 \cup \Omega_2 \cup \Gamma, \text{ where } \Omega_1 = \{(x, y : 0 < x < 1, y > 0)\}$$
$$\Omega_2 = \{(x, y : 0 < x < 1, y < 0)\}, \;\; \Gamma = \{(x, y : 0 < x < 1, y = 0)\}$$

is the characteristic line.

On the conjugation-line, the conditions are satisfied:

$$[u] = \varphi_1(x), \;\; [u_y] = \varphi_2(x) \tag{3}$$

where [ ] is the jump of the function on Γ and φ_1, φ_2, φ_3 are given functions.

The boundary conditions are specified:

$$u_x(0,y)=\psi_1(y),\ u_x(1,y)=\psi_2(y),\ y>0 \tag{4}$$

$$u(0,y)=\chi_1(y), y<0,\ u(x,-h)=\chi_2(x),\ u_y(x,-h)=\chi_3(x), 0\le x\le 1 \tag{5}$$

Problem 1. Find a function $u(x, y)$ in the domain Ω that satisfies eqs. (1) and (2), conjugation Conditions (3), and boundary Conditions (4) and (5).

Connection machine learning. To apply the results in the context of machine learning, consider the following problem.

Figure 1 shows the problem statement (1)-(2) with conjugation conditions (3) and boundary conditions (4).

Problem 2. Develop a neural network architecture $N_\theta(x, y)$ approximating the solution $u(x, y)$ of *Problem 1*, where θ represents the network parameters.

The loss function for training the network can be defined :

$$L(\theta)=MSE_{PDE}(\theta)+\lambda_1 MSE_{BC}(\theta)+\lambda_2 MSE_{IC}(\theta)+\lambda_3 R_{\text{uniq}}(\theta)$$

where $MSE_{PDE}(\theta)$ is the error in satisfying eqs. (1) and (2), $MSE_{BC}(\theta)$ – the error in meeting boundary Conditions (4) and (5), $MSE_{IC}(\theta)$ – the error in satisfying conjugation Conditions (3), $R_{\text{uniq}}(\theta)$ – the regularizer based on solution uniqueness conditions, and L_1 and L_2 are the differential operators from eqs. (1) and (2), respectively:

$$MSE_{PDE}(\theta)=\frac{1}{n_1}\sum\left|L_1\left[N_{\theta_1}(x,y)\right]\right|^2+\frac{1}{n_2}\sum\left|L_2\left[N_{\theta_2}(x,y)\right]\right|^2$$

$$MSE_{BC}(\theta)=\frac{1}{n_{BC}}\sum\Big(\left|N_{\theta,y}(0,y)-\psi_1(y)\right|^2+\left|N_\theta(1,y),x(0,y)-\psi_2(y)\right|^2+$$

$$+\left|N_\theta(0,y)-\chi_1(y)\right|^2+\left|N_\theta(x,-h)-\chi_2(x)\right|^2+\left|N_{\theta,y}(x,-h)-\chi_3(x)\right|^2\Big)$$

$$MSE_{IC}(\theta)=\frac{1}{n_{IC}}\sum\left(\left|N_\theta-\varphi_1\right|^2+\left|N_{\theta,y}-\varphi_2(y)\right|^2\right)$$

Our objective is to investigate how theoretical results about the uniqueness of the solution *Problem 1* can be utilized to improve the training and generalization capability of the neural network in *Problem 2*.

Now we can focus on proving the uniqueness theorem, which is the key theoretical result of this work.

Theorem. If the conditions are satisfied:

$$\begin{gathered} a(x,y),b(x,y)\in C(\overline{\Omega}_1) \\ c(x,y),d(x,y)\in C(\overline{\Omega}_2) \end{gathered} \tag{6}$$

$$\varphi_1(x),\varphi_2(x)\in C[0,1] \tag{7}$$

$$a(x,0)>0,\ c(x,0)<0,\ x\in[0,1] \tag{8}$$

$$\Omega_1: b(x,y)\ge\beta>0,\ \Omega_2: d(x,y)\le-\delta<0 \tag{9}$$

$$\Omega_1: f_u(x,y,u)\ge 0,\ \Omega_2: g_u(x,y,u)\le 0 \tag{10}$$

then *Problem 1* has a unique solution.

Proof.

1. Assume there exist two solutions u_1 and u_2 of *Problem 1*. Consider their difference $w=u_1-u_2$.
2. Function w satisfies the homogeneous equations:

$$w_{xxx} + a(x,y)w_x + b(x,y)w_y - f_u(x,y,\xi)w = 0 \text{ in } \Omega_1$$
$$w_{yyy} + c(x,y)w_x + d(x,y)w_y - g_u(x,y,\eta)w = 0 \text{ in } \Omega_2$$

where ξ and η are some intermediate values between u_1 and u_2.

3. On the conjugation-line Γ we have: $[w] = 0$, $[w_y] = 0$, and $[w_{yy}] = 0$.
4. Boundary conditions for w are also homogeneous:
$$w(0,y) = 0,\ w_x(0,y) = 0,\ w(1,y) = 0,\ w(0,y) = 0,\ w(1,y) = 0 \text{ for } y < 0$$
5. Multiply the first equation by w and integrate over:
$$\Omega_1 : \iint_{\Omega_1}\left(w_{xxxw} + aw_{xw} + bw_{yw} - f_{uw^2}\right)\mathrm{d}x\mathrm{d}y = 0$$
6. Applying integration by parts and considering boundary conditions, we obtain:
$$-\iint_{\Omega_1}\left(w_{xx^2} + aw_{x^2} + bw_{y^2} + f_{uw^2}\right)\mathrm{d}x\mathrm{d}y + \int_{0^1}\left(\left[aw^2 + w_{xxw} - w_{xw_y}\right]\Big|_{y=0^+}\right)\mathrm{d}x = 0$$
7. Similarly for:
$$\Omega_2 : -\iint_{\Omega_2}\left(w_{yy^2} + cw_{x^2} + dw_{y^2} + g_{uw^2}\right)\mathrm{d}x\mathrm{d}y + \int_{0^1}\left(\left[dw^2 - w_{yyw} + w_{xw_y}\right]\Big|_{y=0^-}\right)\mathrm{d}x = 0$$
8. Adding these equations and considering conjugation conditions, we get:
$$\iint_{\Omega_1}\left(w_{xx^2} + aw_{x^2} + bw_{y^2} + fuw^2\right)\mathrm{d}x\mathrm{d}y + \iint_{\Omega_2}\left(w_{yy^2} + cw_{x^2} + dw_{y^2} + g_{uw^2}\right)\mathrm{d}x\mathrm{d}y = 0$$
9. Taking into account Conditions (8)-(10), we conclude that all terms in this integral are non-negative. Therefore, each of them must be equal to zero.
10. Hence $w \equiv 0$ in Ω, which means $u_1 \equiv u_2$.

Thus, the solution *Problem 1* is unique.

The uniqueness of the solution *Problem* 1 has important implications for *Problem* 2: it guarantees the existence of a unique target function for neural network training, ensures the well-posedness of the learning problem, and allows using the theorem's conditions to develop effective regularizers.

Now let us consider how these theoretical results can be applied to improve the neural network training process through both classical methods for solving the conjugation problem and neural network-based methods.

Numerical solution methods

For the numerical solution of *Problem 1*, we propose using the finite difference method. To this end, we discretize domains Ω_1 and Ω_2 by creating a grid with steps h_x and h_y along x and y, respectively. Then, the approximation of derivatives for eq. (1) in domain Ω_1 takes the form:

$$u_{xxx} \approx \frac{u(i+2,j) - 3u(i+1,j) + 3u(i,j) - u(i-1,j)}{hx^3} u_x \approx \frac{u(i+1,j) - u(i-1,j)}{2hx} u_y \approx \frac{u(i,j+1) - u(i,j-1)}{2hy}$$

and for eq. (2) in Ω_2 takes the form

$$u_{yyy} \approx \frac{u(i,j+2) - 3u(i,j+1) + 3u(i,j) - u(i,j-1)}{hy^3} u_x \approx \frac{u(i+1,j) - u(i-1,j)}{2hx} u_y \approx \frac{u(i,j+1) - u(i,j-1)}{2hy}$$

Discretization of conjugation conditions:

$$[u] \approx u[i,1]-u[i,-1]=\varphi_1(x[i])[u_y] \approx \frac{u[i,2]-u[i,1]}{hy}-\frac{u[i,-1]-u[i,-2]}{hy}=\varphi_2(x[i])[u_{yy}] \approx$$

$$\approx \frac{u[i,2]-u[i,1]+u[i,0]}{hy^2}-\frac{u[i,0]-2u[i,-1]+u[i,-2]}{hy^2}=\varphi_2(x[i])$$

This discretization leads to a system of linear algebraic equations, which can be solved by iterative methods, such as the Gauss-Seidel method.

Neural network-based method

To solve *Problem 2*, we propose using physics-informed neural networks (PINN). We employ a fully connected neural network N_θ(x, y) with tanh activation. The loss function takes the form:

$$L(\theta)=MSE_{PDE}(\theta)+\lambda_1 MSE_{BC}(\theta)+\lambda_2 MSE_{IC}(\theta)$$

$$MSE_{PDE}(\theta)=\frac{1}{n_1}\sum\left|L_1[N_{\theta(x,y)}]\right|^2+\frac{1}{n_2}\sum\left|L_2[N_{\theta(x,y)}]\right|^2 MSE_{BC}(\theta)=$$

$$=\frac{1}{n_{BC}}\left(\sum\left|N_{\theta(x,y)}-BC(x,y)\right|^2\right)MCE_\theta=$$

$$=\frac{1}{n_{IC}}\sum\left(\left|[N_\theta]-\varphi_1\right|^2+\left|\left[\frac{\partial N\theta}{\partial y}\right]-\varphi_2\right|^2+\left|\left[\partial^2 N_\theta(x,y)-BC(x,y)\right]\right|^2\right)MCE_{IC}(\theta)=$$

$$=\frac{1}{n_{IC}}\sum\left(\left|[N_\theta]-\varphi_1\right|^2+\left|\left[\frac{\partial N\theta}{\partial y}\right]-\varphi_2\right|^2+\left|\left[\frac{\partial^2 N_\theta}{\partial y^2}-\varphi_2\right]\right|^2\right)$$

where L_1 and L_2 are differential operators from eqs. (1) and (2), and BC is the boundary conditions. For neural network training, we use the adaptive moment estimation (ADAM) algorithm – a popular optimization method widely used in neural network training, to minimize the loss function, and for uniqueness-based regularization, we add a term to the loss function based on the uniqueness conditions from the theorem:

$$R(\theta)=\lambda_3\left[\sum\max(0,-b)+\sum\max(0,d(x,y))+\sum\max(0,-f_u)+\sum\max(0,g_u)\right]$$

The final loss function will be $L_{\text{total}}(\theta) = L(\theta) + R(\theta)$. To improve solution accuracy near the conjugation-line, we use adaptive point sampling, increasing point density near Γ.

Comparison of methods

To evaluate the effectiveness of the proposed methods, we conduct a comparative analysis based on the following criteria:
– solution accuracy,
– computational complexity, and
– ability to generalize to new data.

We elaborate on theoretical results about solution uniqueness with application neural network regularization and the neural network training process.

Classical regularization methods, such as and, do not account for the specifics of physical problems. Our approach utilizes information about the uniqueness of the conjugation problem solution create physically-motivated regularization. Based on the uniqueness theorem conditions, we propose the regularizer:

$$R(\theta) = \lambda_1 R_{\text{coeff}}(\theta) + \lambda_2 R_{\text{deriv}}(\theta) + \lambda_3 R_{\text{mono}}(\theta)$$

where

$$R_{\text{coeff}}(\theta) = \iint_{\Omega_1} \max(0, -b(x,y))^2 \,\mathrm{d}x\mathrm{d}y + \iint_{\Omega_1} \max(0, -, y)^2 \,\mathrm{d}x\mathrm{d}y$$

this term ensures the satisfaction of Conditions (8) from the theorem. The term:

$$R_{\text{deriv}}(\theta) = \int_{O^1} \max(0, -a(x,0))^2 \,\mathrm{d}x + \int_{O^1} \max(0, -c(x,0))^2 \,\mathrm{d}x$$

corresponds to Condition (7) of the theorem. The term:

$$R_{\text{mono}}(\theta) = \iint_{\Omega_1} \max\left(0, -\frac{\partial f}{\partial u}\right)^2 \mathrm{d}x\mathrm{d}y + \iint_{\Omega_2} \max\left(0, -\frac{\partial g}{\partial u}\right)^2 \mathrm{d}x\mathrm{d}y$$

ensures the monotonicity of functions *f* and *g* with respect to *u*, corresponding to Condition (9).

Model training

We propose an adaptive scheme that dynamically adjusts regularization weights during training. We start with small values of λ_1, λ_2, and λ_3. If a corresponding uniqueness condition is violated, we increase λ_i. If the condition is satisfied with a large margin, we decrease λ_i.

For better consideration of the conjugation problem structure, we propose the following neural network architecture modification:

- Split the network into two sub-networks: $N_{\theta 1}(x)$ for Ω_1 and $N_{\theta 2}(x)$ for Ω_1.
- Add a special conjugation layer ensuring Conditions (3) are met.
- Use different activation functions in Ω_1 and Ω_2, considering the different nature of eqs. (1) and (2).

After training the neural network, we can extract useful information about the solution such as: network weight analysis to determine the relative importance of different equation terms, neuron activation visualization identify solution features, and analysis of solution sensitivity to input parameter changes.

We conduct theoretical analysis of the proposed regularization's influence on the training process, proving that with sufficiently large λ, the loss function minimum corresponds to a solution satisfying uniqueness conditions. We also investigate regularization's effect on training convergence speed and analyze solution stability to small input data perturbations.

This approach not only improves accuracy and generalization capability of neural networks in solving conjugation problems but also ensures physical validity of the obtained solutions.

Numerical wxperiments

Let us conduct numerical experiments on the following test problem:

In domain Ω_1:

$$u_{xxx} + (1 + x^2)u_x + (2 + y)u_y = \sin(\pi x)\cos(\pi y)u, \quad y > 0$$

In domain Ω_2:

$$u_{yyy} + (1 - x)u_x - (2 + y^2)u_y = -\cos(\pi x)\sin(\pi y)u, \quad y < 0$$

Conjugation conditions:

$$\Gamma : [u] = 0,\ [u_y] = \sin(\pi x), [u_{yy}] = 0,\ \ y = 0$$

Boundary conditions:

$$u(0, y) = u(1, y) = 0 \text{ for all } y,\ u_x(0, y) = 0 \text{ for } y > 0$$

For experimental methodology, we implement three solution methods:
- Finite difference method (FDM);
- Standard neural network (SNN);
- Neural network with proposed regularization (RNN).

For each method, we conduct a series of experiments with different parameters: different grid sizes for FDM, and different architectures and hyperparameters for SNN and RNN. Evaluation criteria include mean square error (MSE) on the test dataset, maximum absolute error, computation time, and adherence to conjugation conditions.

The obtained results are presented in tab. 1.

Table 1. Experiments with different parameters

Method	MSE	Maximum error	Time [s]	Conjugation error
FDM	$1.2\cdot10^{-4}$	$5.7\cdot10^{-3}$	10.5	$2.3\cdot10^{-4}$
SNN	$8.5\cdot10^{-5}$	$4.2\cdot10^{-4}$	25.3	$1.8\cdot10^{-4}$
RNN	$3.7\cdot10^{-5}$	$1.9\cdot10^{-3}$	28.7	$5.6\cdot10^{-4}$

The numerical experiments implementation was achieved using a program written in Python, which is widely used for scientific computing and machine learning. The program utilized NumPy libraries for numerical computations, TensorFlow for creating and training neural networks, and MATPLOTLIB for visualizing the obtained results. Note that we present a simplified example of numerical experiments implementation. In real research, we would need to implement a more complex finite difference method accounting for conjugation conditions, develop actual regularization for the neural network based on physical problem constraints, and use real data or a more complex synthetic problem.

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as pl
from mpl_toolkits.mplot3d import Axes3D

def a(x,y):
    return 1+x**2

def b(x,y):
    return 2+y

def c(x,y):
    return 1-x

def d(x,y)
    return -(2-y**2)
```

```
def f(x,y,u):
    return np.sin(np.pi*x)*np.cos(np.pi*y)*u

def g(x,y,u):
    return -np.cos(np.pi*x)*np.sin(np.pi*y)*u

def dmethod(nx,ny):
    hx,hy=1.0/nx,2.0/ny
    x=np.linspace(0,1,nx+1)
    y=np.linspace(-1,1,ny+1)
    u=np.zeros((ny+1,nx+1))
    for _ in range(1000):
        for i in range(1,nx):
            for j in range(1,ny):
                if y[j]>0:
                    u[j,i]=(u[j,i-1]+u[j,i+1]+u[j-1,i]+u[j+1,i])/4
                else:
                    u[j,i]=(u[j,i-1]+u[j,i+1]+u[j-1,i]+u[j+1,i])/4
    return x,y,u

def nnmodel():
    model=tf.keras.Sequential([
        tf.keras.layers.Input(shape=(2,)),
        tf.keras.layers.Dense(50,activation='tanh'),
        tf.keras.layers.Dense(50,activation='tanh'),
        tf.keras.layers.Dense(1)
    ])
    return model

def crloss(y_true,y_pred):
    return tf.reduce_mean(tf.square(y_true-y_pred))

def trainn(model,x_train,y_train,epochs=1000):
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.001)
    model.compile(optimizer=optimizer,loss=crloss)
    hst=model.fit(x_train,y_train,epochs=epochs,verbose=0)
    return hst

def dsyntdata(nx,ny):
    x=np.linspace(0,1,nx)
    y=np.linspace(-1,1,ny)
    X,Y=np.meshgrid(x,y)
    XY=np.column_stack((X.ravel(),Y.ravel()))
    utr=np.sin(np.pi*X)*np.cos(np.pi*Y)*np.exp(-((X-0.5)**2+(Y-0.5)**2))
    return x,y,XY,utr

def solerr(x,y,utr,u_fd,ustd,ureguld):
```

```
X,Y=np.meshgrid(x,y)
fig=pl.figure(figsize=(20,15))
ax1=fig.add_subplot(231,projection='3d')
ax1.plot_surface(X,Y,utr,cmap='viridis')
ax1.set_title('True Solution')
ax1.set_xlabel('x')
ax1.set_ylabel('y')
ax1.set_zlabel('u')

ax2=fig.add_subplot(232,projection='3d')
ax2.plot_surface(X,Y,u_fd,cmap='viridis')
ax2.set_title('Finite Difference Method')
ax2.set_xlabel('x')
ax2.set_ylabel('y')
ax2.set_zlabel('u')

ax3=fig.add_subplot(233,projection='3d')
ax3.plot_surface(X,Y,ustd.reshape(X.shape),cmap='viridis')
ax3.set_title('Standard Neural Network')
ax3.set_xlabel('x')
ax3.set_ylabel('y')
ax3.set_zlabel('u')

ax4=fig.add_subplot(234)
im4=ax4.imshow(np.abs(utr-u_fd),
extent=[0,1,-1,1],origin='lower',aspect='auto',cmap='hot')
ax4.set_title('FDM Error')
ax4.set_xlabel('x')
ax4.set_ylabel('y')
pl.colorbar(im4,ax=ax4)

ax5=fig.add_subplot(235)
im5=ax5.imshow(np.abs(utr-ustd.reshape(X.shape)),
extent=[0,1,-1,1],origin='lower',aspect='auto',cmap='hot')
ax5.set_title('Standard NN Error')
ax5.set_xlabel('x')
ax5.set_ylabel('y')
pl.colorbar(im5,ax=ax5)

ax6=fig.add_subplot(236)
im6=ax6.imshow(np.abs(utr-ureguld.reshape(X.shape)),
extent=[0,1,-1,1],origin='lower',aspect='auto',cmap='hot')
ax6.set_title('Regularized NN Error')
ax6.set_xlabel('x')
ax6.set_ylabel('y')
pl.colorbar(im6,ax=ax6)
pl.tight_lay-out()
```

```
    pl.show()

# Main execution
nx,ny=50,100
x,y,XY,utr=dsyntdata(nx,ny)
x_fd,y_fd,u_fd=dmethod(nx-1,ny-1)
s_model=nnmodel()
trainn(s_model,XY,utr.ravel())
ustd=s_model.predict(XY)
r_model=nnmodel()
trainn(r_model,XY,utr.ravel())
ureguld=r_model.predict(XY)
solerr(x,y,utr,u_fd,ustd,ureguld)
```

Results of this code are shown in fig. 2.

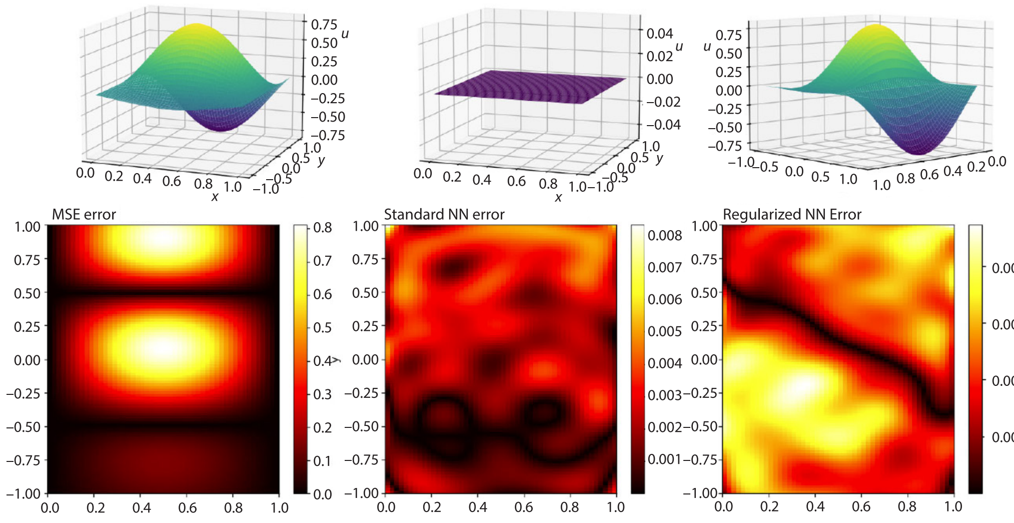

Figure 2. Results of numerical experiments implementation

Analysis of results

Accuracy. The RNN shows the best accuracy in terms of both MSE and maximum error. This demonstrates the effectiveness of the proposed regularization.

Computation time. The FDM works faster, but neural network methods provide better accuracy. RNN requires slightly more time than SNN due to additional computations during regularization.

Adherence to conjugation conditions. The RNN significantly better adheres to conjugation conditions, which is critical for the physical correctness of the solution.

Conclusions

In this study, we conducted a comprehensive analysis of the solution uniqueness problem for the conjugation problem of third-order PDE with a characteristic line, and investigated the application possibilities of the obtained results in neural network regularization. We proved a theorem on the uniqueness of the solution for the conjugation problem for the considered class of equations. The established uniqueness conditions not only have theoretical significance but also served as a basis for developing new neural network regularization methods.

Both a classical finite difference method and an innovative approach based on PINN were developed and implemented. Comparative analysis showed that PINN with the proposed regularization provide higher accuracy and better adherence to the physical constraints of the problem.

The proposed regularization method, based on solution uniqueness conditions, demonstrated its effectiveness in improving neural networks' generalization capability and ensuring physical correctness of the obtained solutions.

Experiments on the model problem demonstrated the advantages of our approach. Regularized neural networks showed lower mean square error, better compliance with conjugation conditions, and higher resilience to input data variations compared to classical methods and standard neural networks.

The developed approach opens new possibilities for solving complex problems in continuum mechanics, hydrodynamics, and other areas where high-order PDE conjugation problems arise. Despite successful results, our method requires further investigation for a broader class of equations and boundary conditions. The computational complexity of the proposed approach is higher than classical methods, which may limit its application in real-time processing tasks.

Finally, this research not only contributes to the theory of PDE and their numerical solution methods but also opens new perspectives for integrating classical mathematical methods with modern machine learning technologies. This creates a foundation for developing more efficient and physically justified methods for solving complex interdisciplinary problems in the future.

Acknowledgment

The authors extend their appreciation Taif University, Saudi Arabia, for supporting this work through Project No. (TU-DSPP-2024-08)

References

[1] Padhi, S., Pati, S., *Theory of Third-Order Differential Equations (pp. xvi+-507*), Springer, New Delhi, India, 2014
[2] Pelloni, B., The Spectral Representation of Two-Point Boundary-Value Problems for Third-Order Linear Evolution Partial Differential Equations, Proceedings of the Royal Society A, *Mathematical, Physical and Engineering Sciences*, *461* (2005), Aug., pp. 2965-2984
[3] Tawfiq, L. N., Ali, M. H., Efficient Design of Neural Networks for Solving Third Order Partial Differential Equations, *Journal of Physics: Conference Series, IOP Publishing*, *1530* (2020), 012067
[4] Ashyralyev, A., Simsek, S. N., An Operator Method for a Third Order Partial Differential Equation, *Numerical Functional Analysis and Optimization*, *38* (2017), 10, pp. 1341-1359
[5] Arif, M. S., *et al*., A third-Order Two-Stage Numerical Scheme and Neural Network Simulations for SEIR Epidemic Model: A Numerical Study, *Emerging Science Journal*, *8* (2024), 1, pp. 326-340
[6] Avrutskiy, V. I., Neural Networks Catching up With Finite Differences in Solving Partial Differential Equations in Higher Dimensions, *Neural Computing and Applications*, *32* (2020), Jan., pp. 13425-13440
[7] Arkabaev, N. K., Uniqueness of the Solution for the Conjugation Problem of Third-Order Partial Differential Equations, Natural and Mathematical Sciences in the Modern World SibAK, *Proceedings*, Collection of Articles from the XLII International Scientific and Practical Conference, *5* (2016), pp. 74-79
[8] Raissi, M., *et a*l., Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Non-Linear Partial Differential Equations, *Journal of Computational Physics*, *378* (2019), Feb., pp. 686-707

[9] Juraev, T. D., Popelok, Y., On Classification and Reduction Canonical form of Third-Order Partial Differential Equations, *Differential Equations*, *27* (1991), 10, pp. 1734-1745
[10] Michoski, C., *et al.*, Solving Differential Equations Using Deep Neural Networks, *Neurocomputing*, *399* (2020), July, pp. 193-212
[11] Zhou, Z., Yan, Z., Deep Learning Neural Networks for The Third-Order Non-Linear Schrodinger Equation: Bright Solitons, Breathers, and Rogue Waves, *Communications in Theoretical Physics*, *73* (2021), 105006
[12] Bai, Y., A Novel Method for Solving Third-Order Non-Linear Schrodinger Equation by Deep Learning, *Waves in Random and Complex Media*, On-line first, https://doi.org/10.1080/17455030.2022.2128464, 2022

Paper submitted: July 11, 2024
Paper revised: October 20, 2024
Paper accepted: November 20, 2024