

Learning finite difference methods for reaction-diffusion type equations with FCNN

Yongho Kim^a, Yongho Choi^{b,*}

^a*Faculty of Mathematics, Otto-von-Guericke-Universität Magdeburg, Universitätsplatz 2, 39106, Magdeburg, Germany*

^b*Department of Computer & Information Engineering, Daegu University, Gyeongsan-si, Gyeongsangbuk-do 38453, Republic of Korea*

Abstract

In recent years, Physics-informed neural networks (PINNs) have been widely used to solve partial differential equations alongside numerical methods because PINNs can be trained without observations and deal with continuous-time problems directly. In contrast, optimizing the parameters of such models is difficult, and individual training sessions must be performed to predict the evolutions of each different initial condition. To alleviate the first problem, observed data can be injected directly into the loss function part. To solve the second problem, a network architecture can be built as a framework to learn a finite difference method. In view of the two motivations, we propose Five-point stencil CNNs (FCNNs) containing a five-point stencil kernel and a trainable approximation function for reaction-diffusion type equations including the heat, Fisher's, Allen-Cahn, and other reaction-diffusion equations with trigonometric function terms. We show that FCNNs can learn finite difference schemes using few data and achieve the low relative errors of diverse reaction-diffusion evolutions with unseen initial conditions. Furthermore, we demonstrate that FCNNs can still be trained well even with using noisy data.

Keywords: convolutional neural networks, reaction-diffusion type equations,

*Corresponding author

Email addresses: ykim@mpi-magdeburg.mpg.de (Yongho Kim),

yongho_choi@daegu.ac.kr (Yongho Choi)

URL: <https://sites.google.com/view/yh-choi> (Yongho Choi)

1. Introduction

To express diverse natural phenomena such as sound, heat, electrostatics, elasticity, thermodynamics, fluid dynamics, and quantum mechanics mathematically, various partial differential equations (PDEs) have been derived and numerical methods can be applied to solve these PDEs. Representative numerical methods for solving PDEs include the finite difference method, finite element method, finite volume method, spectral method, and so on. In this study, we focus on the finite difference method (FDM) which divides a given domain into finite grids and finds an approximate solution using derivatives with finite differences [1, 2]. This method uses each points and its neighbors to predict the corresponding point at the next time step. Likewise, in convolutional neural networks (CNNs) [21], convolution operators extract each pixel of outputs using the corresponding pixel and its neighbor pixels of an input. Also, the convolution operators are generally immutable. Hence, well-structured CNNs have the potential to solve partial differential equations numerically [20]. Among various PDEs representing natural phenomena, we consider reaction-diffusion type equations. The reaction-diffusion model has been applied and used in various fields such as biology [4, 5, 6], chemistry [7, 8, 9], image segmentation [10, 11, 12], image inpainting [13, 14, 15], medicine [16, 17, 18]. In this paper, we focus on second order reaction-diffusion type equations, including the heat, Fisher's, Allen-Cahn (AC) equation, and reaction-diffusion equations with trigonometric function terms.

In recent years, neural networks have been widely applied to solve PDEs. As a popular framework, Physics-informed neural networks (PINNs) [23] based on multi-layer perceptrons (MLPs) approximate PDE solutions by the optimization of a loss function including given laws of physics. The primary advantage of PINNs is that PDE solutions can be inferred without any iterative process such

as a recurrence equation with respect to time. Furthermore, it is used for diverse applications such as Hidden Fluid Mechanics (HFMs) [26] that extract hidden variables of a given equation using a PINN and observations. However, training models depending on intricate PDEs and their coefficients is classically difficult. Also, the inference of a PINN depends on a given initial condition, hence an individual training session is required whenever the initial condition changes. To improve the training ability, PINNs have been combined with numerical methods as well as other neural networks such as CNNs have been selected. M. Raissi et al. [23] added Runge-Kutta methods to a PINN model to solve the AC equation. Aditi et al. [24] proposed transfer learning and curriculum regularization which start training PINNs on a specific safe domain and then transfer their learning to a target domain. Hao Ma et al. [25] proposed a U-shaped CNN called U-net [28] and showed that the usage of target data in the loss function part significantly improves the model optimization. Elie Bretin et al. [27] used convolutional neural networks derived from a semi-implicit approach to learn phase field mean curvature flows of the AC equation. We propose Five-point stencil CNNs (FCNNs) containing a five-point stencil kernel and a trainable approximation function to obtain the numerical solutions of second order reaction-diffusion type equations. Our contributions are as follows:

1. We propose a five-point stencil convolution operator to solve reaction-diffusion type equations.
2. We show that finite difference methods can be reconstructed by FCNNs with two consecutive snapshots and that FCNNs achieve low relative errors for diverse evolutions.
3. We demonstrate that the robustness of FCNNs using five reaction-diffusion type equations and noisy data.

The remainder of this paper is organized as follows. In Section 2, we present how to create training data using explicit FDMs and explain the concept of FCNNs, training process, and numerical solutions. In Section 3.1 and 3.2, we measure the relative errors between FCNN and FDM solutions as well as we

show the robustness of FCNNs using the diverse initial conditions and noisy data. In Section 3.3, we compare FCNN to PINN. Finally, summarizes our results and concludes the work in Section 4.

2. Methods and numerical solutions

The FDM divides a given domain into finite grids and find an approximate solution using derivatives with finite differences [1, 2]. To create training data for each equation, we apply an explicit FDM to create training data with a random initial condition. A computational domain is defined using a uniform grid of size $h = 1/N_x = 1/N_y$ and $\Omega_h = \{(x_i, y_j) = (a + (i - 1.5)h, c + (j - 1.5)h)\}$ for $1 \leq i \leq N_x + 2$ and $1 \leq j \leq N_y + 2$. Here, N_x and N_y are mesh sizes on the computational domain $(a, b) \times (c, d)$. Let ϕ_{ij}^n be approximations of $\phi(x_i, y_j, n\Delta t)$ and Δt be temporal step size. The boundary condition is a zero Neumann boundary condition. The Laplacian of a function ϕ is calculated using a five-point stencil method, the Laplacian $\Delta\phi$ can be approximated as follows:

$$\Delta_h\phi \approx \frac{\phi(x+h, y) + \phi(x-h, y) - 4\phi(x, y) + \phi(x, y+h) + \phi(x, y-h)}{h^2}. \quad (1)$$

In this way, the first and second derivatives of ϕ at each point (x_i, y_j) (e.g., $\phi_x, \phi_y, \phi_{xx}$ and ϕ_{yy}) can be approximated within the 3×3 local area centered (x_i, y_j) . This concept can be equivalent to 3×3 convolution kernels. The 3×3 kernels K following properties:

1. $k_1 \oplus k_2 \in K$ for any $k_1, k_2 \in K$ (element-wise summation).
2. $k_1 \odot k_2 \in K$ for any $k_1, k_2 \in K$ (element-wise multiplication).
3. $k^{-1} \in K$ for any $k \in K$ (element-wise division).
4. $ak \in K$ for any $k \in K$ and any real numbers a .

Therefore, second-order PDEs can be solved numerically by the properties of 3×3 kernels [20].

To solve second order reaction-diffusion type equations

$$\phi_t = \alpha \Delta \phi + \beta f(\phi), \quad (2)$$

where α is a diffusion coefficient, β is a reaction coefficient, and f is a smooth function to present reaction effect. We propose FCNN as a recurrence relation:

$$\phi^{n+1} = \phi^n + \Delta t \alpha \Delta_h \phi^n + \Delta t \beta f(\phi^n). \quad (3)$$

As a CNN, $F(\phi^n)$ containing a 5-point stencil filter and a pad satisfying given boundary conditions solves $\Delta t \alpha \Delta_h \phi^n$. To approximate $\phi^n + \Delta t \beta f(\phi^n)$ terms, we define a trainable polynomial function $\epsilon(\phi^n)$ as follows:

$$\epsilon(\phi^n) = a_0 + \sum_{k=1}^N a_k (\phi^n - b)^k, \quad (4)$$

with model parameters a_i for any $i \in \{0, 1, \dots, N\}$ and a real value b . Let $M(\phi^n) = F(\phi^n) + \epsilon(\phi^n)$ be an FCNN. Then, the inference is performed as follows:

$$\phi^{n+1} = M_\theta(\phi^n), \quad (5)$$

where θ is a set of model parameters.

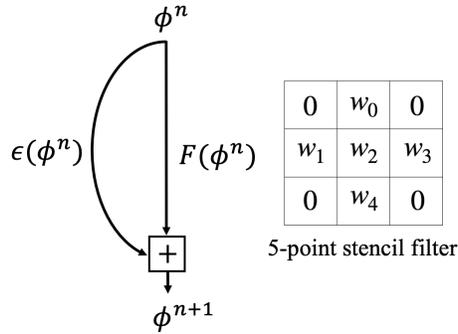


Figure 1: Computational graph of FCNN for second order reaction-diffusion type equations

Figure 1 shows the computational graph of our explicit model FCNN containing model parameters w_i for any $i \in \{0, 1, 2, 3, 4\}$ in a filter. Furthermore, F represents the diffusion term on the uniform grid of x and y axes, so we set up $w_1 = w_3$ and $w_0 = w_4$ to reduce training time. When the five-point stencil filter is used and ϵ is a p -th order polynomial function, the number of model parameters is only $p + 4$. Thus, the set-up enables to learn physical patterns

from few data. In Algorithm 1, an initial image $\phi^0(= u^0)$ and the prediction ϕ^1 at the next time Δt are used with training data u^0 and u^1 to train a model M_θ . The objective function $L(\phi^1, u^1)$ is the mean square error function without physics-informed loss as follows:

$$L(\phi^1, u^1) = \frac{1}{N} \sum_{i=1}^N (\phi_i^1 - u_i^1)^2, \quad (6)$$

where N , ϕ^1 and u^1 are the number of pixels in an output image, a prediction and its target respectively.

Algorithm 1 Training Procedure

Set an initial value $\phi^0 = u^0$, a small constant $\delta > 0$
Initialize $M_\theta(\phi^0) = F(\phi^0) + \epsilon(\phi^0)$ with model parameters θ
while $\ell > \delta$ **do**
 $\phi^1 \leftarrow M_\theta(\phi^0)$
 Compute loss $\ell = L(\phi^1, u^1)$
 Update θ
end while

2.1. Reaction-diffusion type equations

To demonstrate the robustness of FCNNs, we consider reaction-diffusion type equations including the heat, Fisher’s, AC equation, reaction-diffusion equations with trigonometric function terms. The reaction and diffusion coefficients used in each formula are shown in Table 1. For the AC equation, $\beta = 1/\rho^2$

Table 1: Diffusion (α) and reaction (β) coefficients for the simulations.

	Heat	Fisher’s	AC	Sine	Tanh
α	1	1	1	0.1	0.5
β	0	20	6944	40	10

where ρ is the thickness of the transition layer and $\rho_5 \approx 0.012$ [20]. For the other

equations, we select arbitrary reaction coefficients. The zero Neumann boundary condition is used. For all the following equations, the continuous equations and the discretized equations are described in turn.

- Heat equation:

$$\phi_t = \alpha \Delta \phi. \quad (7)$$

- Fisher's equation:

$$\phi_t = \alpha \Delta \phi + \beta(\phi - \phi^2). \quad (8)$$

- AC equation:

$$\phi_t = \alpha \Delta \phi + \beta(\phi - \phi^3). \quad (9)$$

- Reaction-diffusion equation with trigonometric function(sin):

$$\phi_t = \alpha \Delta \phi + \beta \sin(\pi\phi). \quad (10)$$

- Reaction-diffusion equation with trigonometric function(tanh):

$$\phi_t = \alpha \Delta \phi + \beta \tanh(\phi). \quad (11)$$

When $\alpha = 1$ and $\beta = 1$, all the equations show similar evolutions, so we use different reaction coefficient β much larger than diffusion coefficient α as shown in Table 1.

3. Simulation results

3.1. FCNNs

Assume that we observe a reaction-diffusion pattern and investigate the pattern rule under the constraint meaning that the observations and predictions follow the same PDE. Our proposed FCNN is trained using only two consecutive snapshots including the initial and next time step results for each equation. Then, we evaluate the model using diverse unseen initial values.

In the simulations, we use data with random initial values with a 100×100 mesh so that the size of the input data is 102×102 containing a padding as a boundary condition. Also, $N = 3$ (heat, Fisher’s, AC) or 9 (sine, tanh) for $\epsilon(\phi^n)$ is fixed depending on given equations and a 3×3 convolutional filter is used with the stride of 1 in Eq. (4). Hence, the filter has 10,000 ($= ((100 + 2 - 3)/1 + 1)^2$) chances to learn the evolution of results images, so training a model using only two consecutive images suffices to optimize nine or thirteen model parameters $(w_0, \dots, w_4, a_0, \dots, a_3)$.

As an optimizer, we use ADAM [22] with a learning rate of 0.01 and without any regularization. We apply early stopping [29] based on validation data to avoid overfitting. To demonstrate the approximation $\epsilon(\phi^n)$ for non-polynomial functions $f(\phi^n)$, we additionally consider sine and tanh functions in addition to heat, Fisher’s, and AC equations.

For the evaluation, we implement FCNN and FDM respectively and then measure the averaged relative L_2 error with 95% confidence interval over 100 novel random initial values as shown in Table 2.

Table 2: Relative L_2 error between FCNN and FDM. The \pm shows 95% confidence intervals over 100 different random initial values.

Equations	Relative L_2 error
Heat	$8.4 \times 10^{-5} \pm 4 \times 10^{-6}$
Fisher’s	$4.0 \times 10^{-5} \pm 2 \times 10^{-6}$
AC	$1.3 \times 10^{-6} \pm 8 \times 10^{-7}$
Sine	$7.0 \times 10^{-5} \pm 5 \times 10^{-6}$
Tanh	$1.9 \times 10^{-4} \pm 4 \times 10^{-6}$

Furthermore, we validate the errors using different types of initial values for each equation as shown in Table 3. The initial conditions are described in the Appendix Section.

As shown in Fig. 2, the function $c(\phi_{ij})$ calculates the ϕ_{ij} term in the explicit method. Also, the neighboring coefficient of the five-stencil kernel of the explicit

Table 3: Relative L_2 error between FCNN and FDM with diverse initial values.

Initial shapes:	circle	star	circles	torus	maze
Heat	3.4×10^{-5}	4.4×10^{-5}	1.1×10^{-5}	1.1×10^{-4}	4.9×10^{-5}
Fisher's	8.7×10^{-4}	7.2×10^{-4}	1.9×10^{-4}	1.3×10^{-4}	3.7×10^{-5}
AC	2.6×10^{-7}	2.7×10^{-7}	2.3×10^{-7}	2.0×10^{-7}	1.9×10^{-7}
Sine	3.7×10^{-4}	2.2×10^{-4}	9.5×10^{-5}	7.5×10^{-5}	4.1×10^{-5}
Tanh	1.8×10^{-3}	1.4×10^{-3}	6.3×10^{-4}	2.5×10^{-5}	2.9×10^{-5}

method is $\Delta t\alpha/h^2$ and the averaged absolute error between the coefficients w_0, w_1, w_3, w_4 and $\Delta t\alpha/h^2$ is 1.6×10^{-5} . For instance, the explicit method of the AC equation can be expressed as

$$\phi_{i,j}^{n+1} = \frac{\Delta t\alpha}{h^2}(\phi_{i,j+1}^n + \phi_{i,j-1}^n + \phi_{i-1,j}^n + \phi_{i+1,j}^n) + c(\phi_{i,j}^n), \quad (12)$$

where $c(\phi_{i,j}^n) = -4\frac{\Delta t\alpha}{h^2}\phi_{i,j}^n + \beta\Delta t\phi_{i,j}^n - \beta\Delta t(\phi_{i,j}^n)^3$. Finally, it is shown that each numerical scheme can be reconstructed by the proposed FCNN with given u_0 and u_1 .

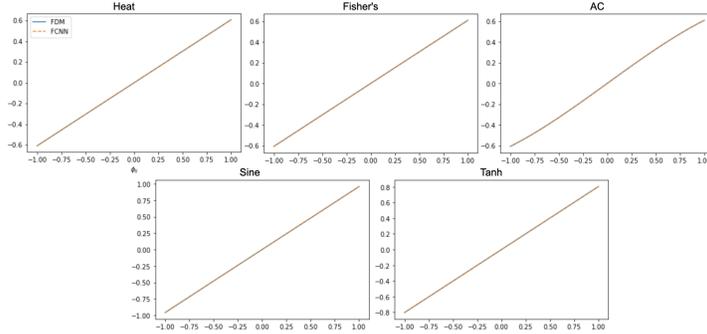


Figure 2: Center function $c(\phi_{ij})$: (blue line) FDMs, (dashed orange line) FCNNs

Figures 3-7 show the time evolution results from unseen initial shapes (circle, star, three circles, torus, and maze) using the pre-trained FCNN of each equation to compare them to the FDM results.

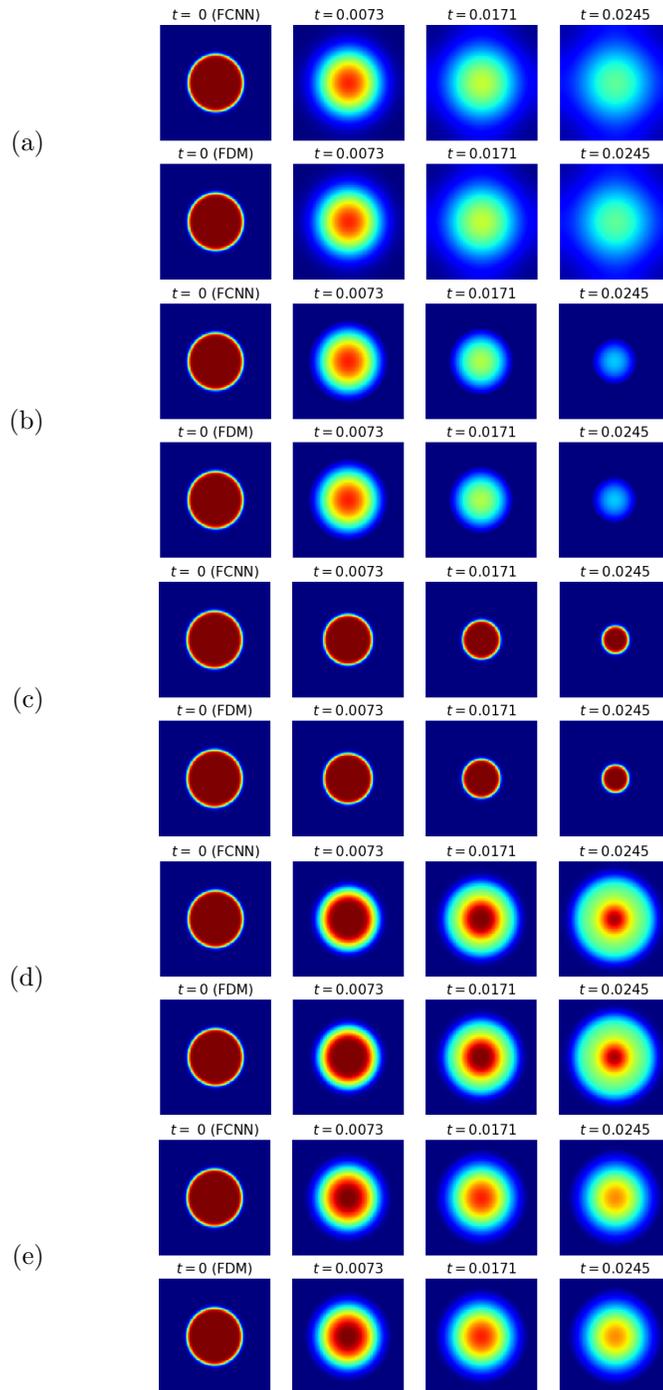


Figure 3: Time evolution of a circle shape of (a) Heat, (b) Fisher's, (c) AC, (d) Sine, and (e) Tanh equations.

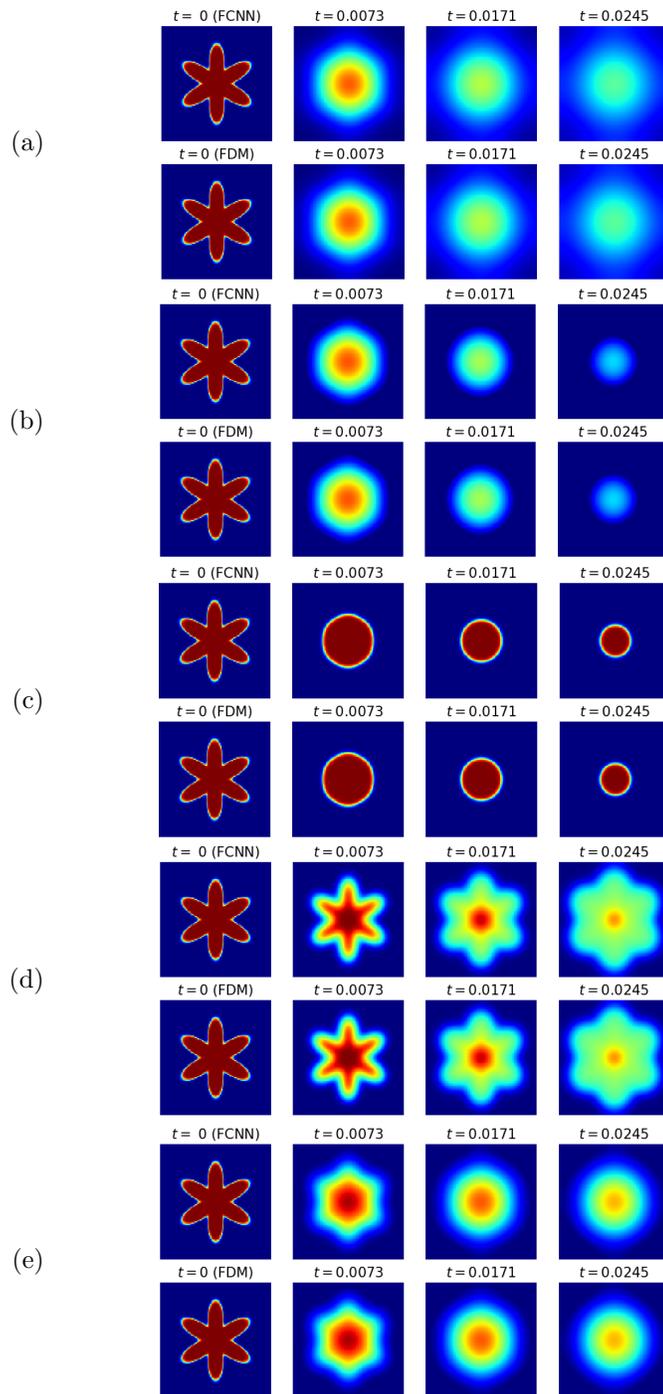


Figure 4: Time evolution of a star shape of (a) Heat, (b) Fisher's, (c) AC, (d) Sine, and (e) Tanh equations.

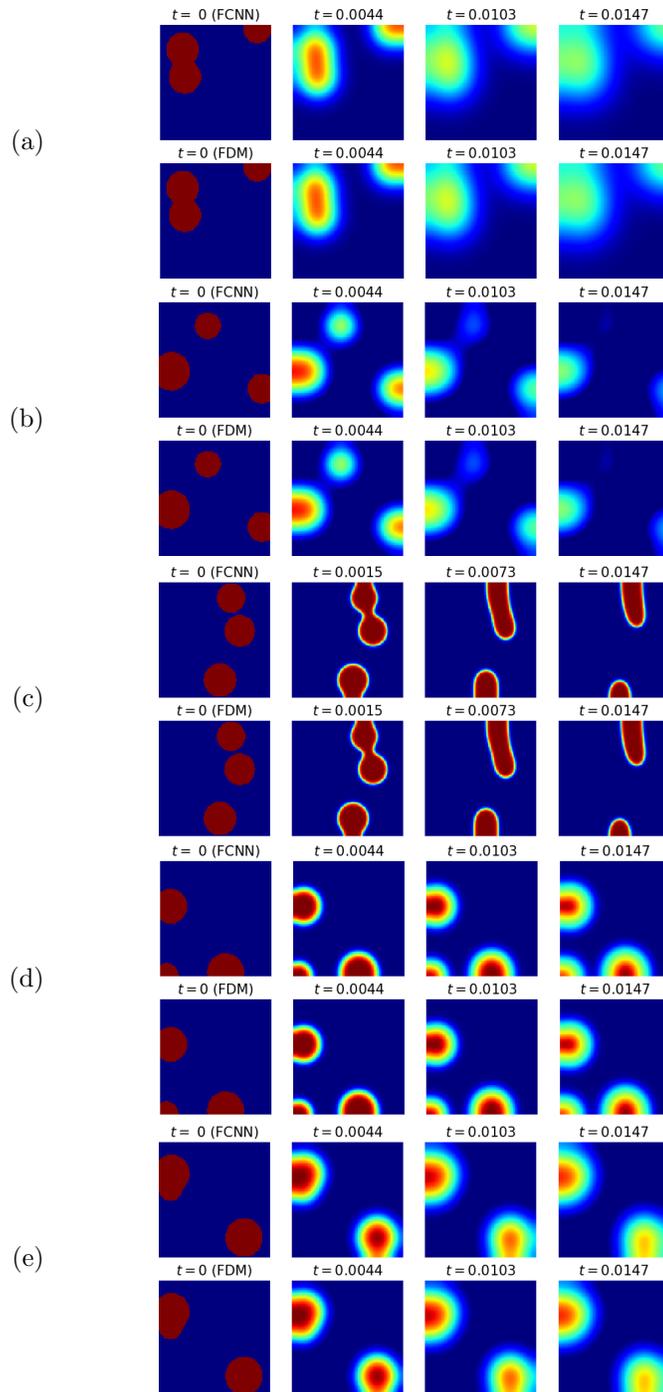


Figure 5: Time evolution of a three circles shape of (a) Heat, (b) Fisher's, (c) AC, (d) Sine, and (e) Tanh equations.

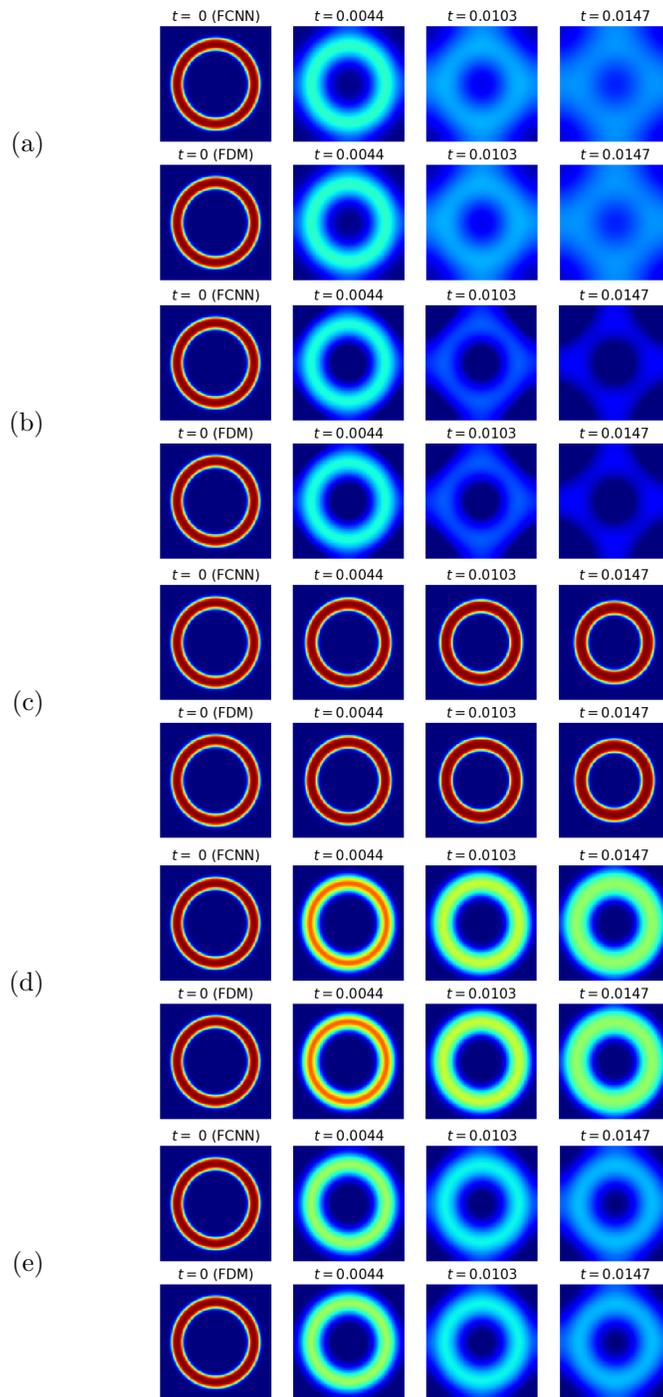


Figure 6: Time evolution of a torus shape of (a) Heat, (b) Fisher's, (c) AC, (d) Sine, and (e) Tanh equations.

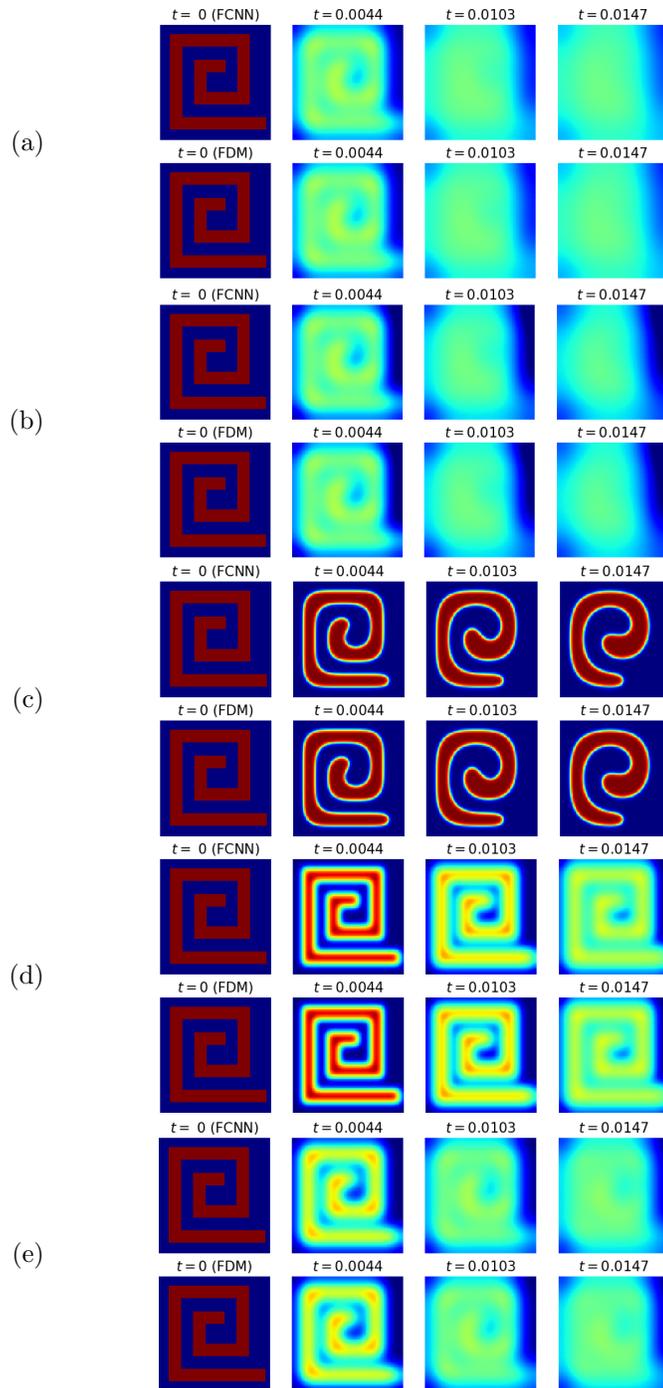


Figure 7: Time evolution of a maze shape of (a) Heat, (b) Fisher's, (c) AC, (d) Sine, and (e) Tanh equations.

3.2. FCNNs with noisy data

Data-driven models are sensitive to data noise. To investigate the effects of noise on the our proposed model, we inject Gaussian random noise $\eta \sim N(0, \sigma^2)$ to u^1 and then the model is trained using u_0 and $u_1 + \eta$ for the AC equation. Table 4 shows that the model could be trained under the noise condition.

Table 4: Relative L_2 error with noise. The \pm shows 95% confidence intervals over 100 different random initial values.

σ	Relative L_2 Error	σ	Relative L_2 Error
0	$1.3 \times 10^{-6} \pm 8 \times 10^{-7}$	10^{-4}	$3.3 \times 10^{-4} \pm 2 \times 10^{-4}$
10^{-6}	$9.1 \times 10^{-5} \pm 6 \times 10^{-4}$	10^{-2}	$1.4 \times 10^{-1} \pm 3 \times 10^{-2}$

Figure 8 shows the results of the inference using contaminated models.

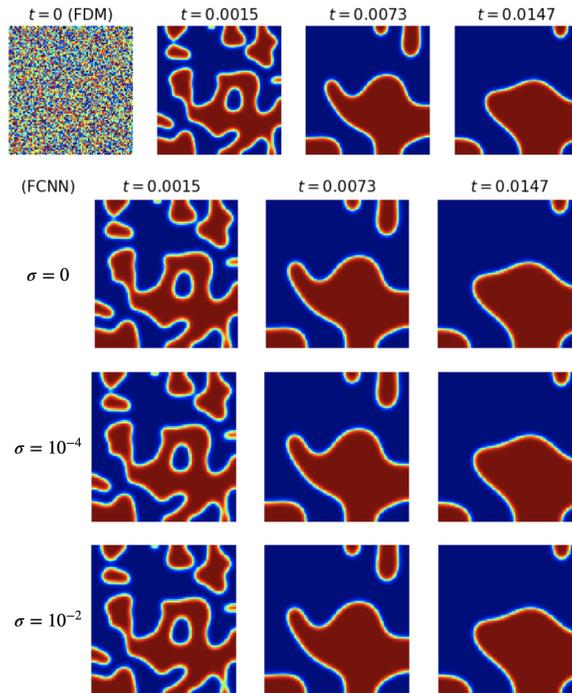


Figure 8: Inference using a contaminated model with the noise impact σ

3.3. Comparison between PINNs and FCNNs

Table 5: PINNs vs. FCNNs in second order reaction-diffusion type equations

	PINNs	FCNNs
model type	continuous	discretized
domain	mesh-free	mesh-dependency
observation	not required	only u_1
optimization	hard	easy
training	u_0 -dependency	u_0 -free

As shown in Table 5, a mesh-free and continuous-time PINN has no constraint on domain structures. Hence, it has the potential to solve PDE solutions without any discretization. Also, observation data are not required to train a model. Nevertheless, several problems remain, such as the untractable PINN optimization and the long training runtime as well as PINNs with each different initial condition u_0 should be trained respectively, despite considering the same PDE. On the contrary, FCNNs can solve the PDE solutions for any initial conditions using a pretrained model that learned evolution patterns from two consecutive snapshots.

We consider the heat equation (7) to compare between the pretrained FCNN and the standard PINN. For the implementation of a PINN, we use 50,000 ($20 \times 50 \times 50$) collocation data, the circle initial condition u_0 (14), with a zero Neumann boundary condition. The baseline network is a MLP consisting of an input layer, three hidden layers, and an output layer with an activation function \tanh in each hidden layer. Each hidden layer has 50 nodes. In the training session, we use the ADAM with a learning rate of 10^{-4} . The loss function is defined as

$$L = L_c + L_b + \lambda L_{ini} \tag{13}$$

where $L_c = \sum_{i=1}^{N_c} f(t_i, x_i, y_i)^2$ is the physics-informed loss, $L_b = \sum_{i=1}^{N_b} \hat{u}(t_i, x_i^b, y_i^b)^2$ is the boundary condition loss with the coordinates (t, x^b, y^b) on the boundary,

$L_{ini} = \sum_{x,y \in \Omega_h} (\hat{u}(0, x, y) - u_0(x, y))^2$ is the initial condition loss, and λ is a positive weight of the initial condition loss. When $\lambda = 1$, it is observed that the L_c and the L_b converge much faster than the L_{ini} causing the training to be biased towards L_c and L_b . Thus, $\lambda = 50$ is selected to alleviate the optimization issue. We perform the simulation on the following specifications: Intel (R) Core (TM) i9-10900K CPU @3.70 GHz, 128 GB RAM/NVIDIA GeForce RTX 3090.

Table 6: FCNN vs. PINN: training runtime and relative L_2 error

	FCNN	PINN
Relative L_2 error	6.03×10^{-5}	1.65×10^{-1}
Training runtime (hours)	0	7

Table 6 shows that training the PINN requires a considerable runtime, and that it is hard to optimize the model despite the expensive training cost. In contrast, the FCNN can predict the evolution without an additional training session by using the pretrained model in Section 3.1. It takes 2 minutes to obtain the pretrained model. The predictions of each method are shown in Fig. 9.

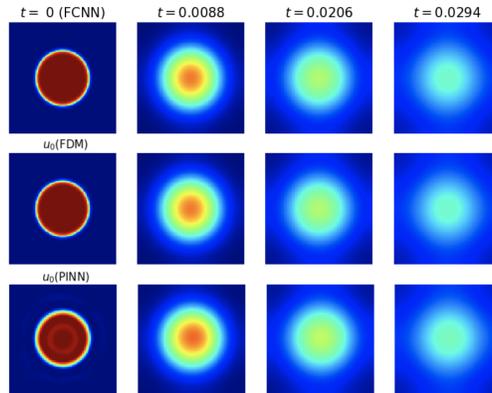


Figure 9: Time evolution of the circle shape of FCNN, FDM, and PINN. PINNs even learn the initial condition from L_{ini} so the trained initial values could not be exactly the same as the given values.

4. Conclusions

In this paper, we proposed Five-point stencil CNNs (FCNNs) containing a five-point stencil kernel and a trainable approximation function. We considered reaction-diffusion type equations including the heat, Fisher’s, Allen–Cahn equations, and reaction-diffusion equations with trigonometric function terms. We demonstrated that our proposed FCNN can be trained using only two consecutive snapshots and can then predict reaction-diffusion evolutions with unseen initial conditions. Also, the robustness of FCNNs was shown by the noise tests and diverse initial conditions. In future works, the characteristics of PINNs are intriguing, although the optimization of PINNs is an intractable problem. We expect that it would be feasible to train FCNNs without u_1 by a physics-informed loss.

Acknowledgments

The corresponding author Y. Choi was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (NRF-2020R1C1C1A0101153713) and supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2022R1I1A307282411). The authors appreciate the reviewers for their constructive comments, which have improved the quality of this paper.

Appendix

In this appendix session, we describe the initial conditions used in the simulation results session 3. A detailed description of these initial conditions can be found in our previous research paper [20].

(1) The initial condition of a circle shape

$$\phi(x, y, 0) = \tanh\left(\frac{R_0 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2}}{\sqrt{2}\epsilon}\right), \quad (14)$$

where R_0 is the initial radius of a circle.

(2) The initial condition of a star shape

$$\phi(x, y, 0) = \tanh \left(\frac{0.25 + 0.1 \cos(6\theta) - \sqrt{(x - 0.5)^2 + (y - 0.5)^2}}{\sqrt{2}\epsilon} \right), \quad (15)$$

where

$$\theta = \begin{cases} \tan^{-1} \left(\frac{y-0.5}{x-0.5} \right), & \text{if } (x > 0.5) \\ \pi + \tan^{-1} \left(\frac{y-0.5}{x-0.5} \right), & \text{otherwise.} \end{cases}$$

(3) The initial condition of a torus shape

$$\phi(x, y, 0) = -1 + \tanh \left(\frac{R_1 - \sqrt{XY}}{\sqrt{2}\epsilon} \right) - \tanh \left(\frac{R_2 - \sqrt{XY}}{\sqrt{2}\epsilon} \right), \quad (16)$$

where R_1 and R_2 are the radius of major (outside) and minor (inside) circles, respectively. And, for simplicity of expression, $XY = (x - 0.5)^2 + (y - 0.5)^2$.

(4) The initial condition of a maze shape

The initial condition of a maze shape is complicated to describe its equation, so refer to the codes which are available from the first author's GitHub web page (<https://github.com/kimy-de/fcnn>) and the corresponding author's web page (<https://sites.google.com/view/yh-choi/code>).

(5) The initial condition of a random shape

$$\phi(x, y, 0) = 0.1\text{rand}(x, y), \quad (17)$$

here the function $\text{rand}(x, y)$ has a random value between -1 and 1 .

References

- [1] P. Zhou, Numerical analysis of electromagnetic fields. Springer Science & Business Media, 2012.

- [2] J. Kim, D. Jeong, SD Yang, Y. Choi, "A finite difference method for a conservative Allen–Cahn equation on non-flat surfaces." *Journal of Computational Physics* 334 (2017): 170-181.
- [3] S Kondo, T Miura, "Reaction-diffusion model as a framework for understanding biological pattern formation." *science* 329.5999 (2010): 1616-1620.
- [4] NF Britton, *Reaction-diffusion equations and their applications to biology.* Academic Press, 1986.
- [5] P Broadbridge, BH Bradshaw-Hajek, "Exact solutions for logistic reaction–diffusion equations in biology." *Zeitschrift für angewandte Mathematik und Physik* 67.4 (2016): 1-13.
- [6] D Jeong, Y Li, Y Choi, M Yoo, D Kang, J Park, J Choi, J Kim "Numerical simulation of the zebra pattern formation on a three-dimensional model." *Physica A: Statistical Mechanics and its Applications* 475 (2017): 106-116.
- [7] BA Grzybowski, *Chemistry in motion: reaction-diffusion systems for micro-and nanotechnology.* John Wiley & Sons, 2009.
- [8] I Sgura, B Bozzini, D Lacitignola, "Numerical approximation of oscillating Turing patterns in a reaction-diffusion model for electrochemical material growth." *AIP Conference Proceedings*. Vol. 1493. No. 1. American Institute of Physics, 2012.
- [9] G Hariharan, R Rajaraman, "A new coupled wavelet-based method applied to the nonlinear reaction–diffusion equation arising in mathematical chemistry." *Journal of Mathematical Chemistry* 51.9 (2013): 2386-2400.
- [10] H Tek, BB Kimia, "Image segmentation by reaction-diffusion bubbles." *Proceedings of IEEE International Conference on Computer Vision*. IEEE, 1995.

- [11] S Esedog, YHR Tsai, "Threshold dynamics for the piecewise constant Mumford–Shah functional." *Journal of Computational Physics* 211.1 (2006): 367-384.
- [12] Z Zhang, YM Xie, Q Li, S Zhou, "A reaction–diffusion based level set method for image segmentation in three dimensions." *Engineering Applications of Artificial Intelligence* 96 (2020): 103998.
- [13] M Bertalmio et al. "Image inpainting." *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. 2000.
- [14] Y Li, D Jeong, J Choi, S Lee, J Kim, "Fast local image inpainting based on the Allen–Cahn model." *Digital Signal Processing* 37 (2015): 65-74.
- [15] J Yu, J Ye, S Zhou, "Reaction-diffusion system with additional source term applied to image restoration." *International Journal of Computer Applications* 975 (2016): 8887.
- [16] E Özüğurlu, "A note on the numerical approach for the reaction–diffusion problem to model the density of the tumor growth dynamics." *Computers & Mathematics with Applications* 69.12 (2015): 1504-1517.
- [17] HG Lee, Y Kim, J Kim, "Mathematical model and its fast numerical method for the tumor growth." *Mathematical Biosciences & Engineering* 12.6 (2015): 1173.
- [18] M Yousefnezhad, CY Kao, SA Mohammadi, "Optimal Chemotherapy for Brain Tumor Growth in a Reaction-Diffusion Model." *SIAM Journal on Applied Mathematics* 81.3 (2021): 1077-1097.
- [19] S.M. Allen, J.W. Cahn, "A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening." *Acta metalurgica* 27.6 (1979): 1085–1095.
- [20] Yongho Kim, Gilnam Ryu, and Yongho Choi (2021) Fast and Accurate Numerical Solution of Allen-Cahn Equation, *Mathematical Prob-*

lems in Engineering, vol. 2021, Article ID 5263989, 12 pages, 2021.
<https://doi.org/10.1155/2021/5263989>

- [21] LeCun, Yann and Boser, Bernhard and Denker, John and Henderson, Donnie and Howard, R. and Hubbard, Wayne and Jackel, Lawrence, (1990), Handwritten Digit Recognition with a Back-Propagation Network, Advances in Neural Information Processing Systems, Vol 2.
- [22] Diederik P. Kingma and Jimmy Ba, (2017). Adam: A Method for Stochastic Optimization, arXiv:1412.6980.
- [23] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis, (2018). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics 378 (2019): 686-707.
- [24] Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney, (2021). Characterizing Possible Failure Modes in Physics-Informed Neural Networks, Neural Information Processing Systems (NeurIPS) 2021, arXiv:2109.01050.
- [25] Hao Ma, Yuxuan Zhang, Nils Thuerey, Xiangyu Hu, Oskar J. Haidn, (2021). Physics-driven Learning of the Steady Navier-Stokes Equations using Deep Convolutional Neural Networks, arXiv:2106.09301.
- [26] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. (2020) Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations, Science 367.6481 (2020): 1026-1030.
- [27] Elie Bretin, Roland Denis, Simon Masnou, and Garry Terii, (2021). Learning Phase Field Mean Curvature Flows With Neural Networks, arXiv:2112.07343.
- [28] Olaf Ronneberger, Philipp Fischer and Thomas Brox, (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation, Medical Image

Computing and Computer-Assisted Intervention (MICCAI) 2015, Lecture Notes in Computer Science, vol 9351. Springer, Cham.

- [29] Lutz Prechelt, (1998). Early Stopping - but when?, In: Orr G.B., Müller KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 1524.