Scientific Research Publishing

Applied Mathematics, 2023, 14, 290-313
https://www.scirp.org/journal/am
ISSN Online: 2152-7393
ISSN Print: 2152-7385

Parameter Identification in Traveling Wave Solutions of a Modified Fisher's Equation

Zhixuan Jia, Ali Nadim

Institute of Mathematical Sciences, Claremont Graduate University, Claremont, USA
Email: zhixuan.jia@cgu.edu, ali.nadim@cgu.edu

How to cite this paper: Jia, Z.X. and Nadim, A. (2023) Parameter Identification in Traveling Wave Solutions of a Modified Fisher's Equation. *Applied Mathematics*, **14**, 290-313.
https://doi.org/10.4236/am.2023.145018

Received: March 25, 2023
Accepted: May 26, 2023
Published: May 29, 2023

Abstract

In this work, we focus on the inverse problem of determining the parameters in a partial differential equation from given numerical solutions. For this purpose, we consider a modified Fisher's equation that includes a relaxation time in relating the flux to the gradient of the density and an added cubic non-linearity. We show that such equations still possess traveling wave solutions by using standard methods for nonlinear dynamical systems in which fixed points in the phase plane are found and their stability characteristics are classified. A heteroclinic orbit in the phase plane connecting a saddle point to a node represents the traveling wave solution. We then design parameter estimation/discovery algorithms for this system including a few based on machine learning methods and compare their performance.

Keywords

PDE, Traveling Wave Solution, Stability Analysis, Machine Learning, Optimization, Embedding

1. Introduction

Many important processes such as fluid flow, heat and mass transport, wave motion, and others involve quantities that vary in space and time and can be modeled by partial differential equations (PDEs). Most PDEs, specially if nonlinear, do not have explicit analytical solutions and numerical methods must be used to solve them. The inverse problem for PDEs/ODEs is one of determining unknown coefficients, parameters, or functions in the equation from a set of observations or measurements. Normally, the inverse problem is more challenging than the forward problem due to its ill-posed nature, in which small errors in the measurements can lead to large errors in the estimated parameters or functions. Inverse problems arise in many scientific and engineering applications, such as

medical imaging, geophysics, non-destructive testing, and material characterization. For example, PDE inverse problems can be used to determine properties of materials from measurements of waves or fields, or to reconstruct images of internal organs from X-rays or CT scans. In this article, we focus on the traveling wave solution of a nonlinear diffusion-reaction equation, namely Fisher's equation but with some modifications. We develop methods for solving the inverse problem, which aims to discover the PDEs that govern a space-time-dependent process and the parameters within those PDEs using various approaches, including machine learning and optimization. Given a numerical solution, the goal is to determine which PDE and what parameters gave rise to that solution.

Fisher's equation was first introduced in 1930 [1] to describe the spread of advantageous genes. Canosa extended it to depict population growth in 1973 [2]. In Murray's book [3], the history and various applications of Fisher's equation are discussed in detail. There has been growing interest in studying the traveling wave solutions of Fisher's equation. Some researchers have found analytic solutions in specific cases. For example, [4] provided an exact solution with wave speed $\pm 5/\sqrt{6}$; [5] provided the solution of quadratic Fisher equation in both one and two dimensions; and [6] provided analytic approximate solutions of some nonlinear parabolic dynamical wave equations (including Fisher's equation) with a modified variational iteration algorithm.

In addition to analytic solutions, there are also many articles that focus on numerical solutions of Fisher's equation. For example, [7] presented the numerical solution of several types of Fisher's reaction-diffusion equation by using cubic trigonometric B-spline functions and the differential quadrature method; [8] used forward-in-time and central-in-space (FTCS) method; and [9] applied Atangana-Baleanu fractional derivative with spectral collocation methods.

In this paper, a relaxation effect is introduced in an equation relating flux to density; the linear part of the operator then becomes hyperbolic and yields a finite speed for the propagation of disturbances. This adds a parameter making the identification problem a bit more challenging. A similar relaxation effect has been introduced into the convection-diffusion equation; for example, [10] solved the unsteady convection-diffusion equation with such a relaxation effect by using third-order accurate implicit-explicit (IMEX) Runge-Kutta method in time and fifth-order finite difference WENO scheme in space. [11] provided an analytic approach with fractional operators, including Liouville-Caputo and Atangana-Baleanu-Caputo operators; [12] used a system similar to the one discussed in the current paper but with different parameters, and they provided large-time solutions with a specific nonlinear function on the right-hand side:
$F(u) = \varepsilon u(1-u)(1+\sigma u)$ taking $0 < \varepsilon < 1$ and $-1 < \sigma$.

Recently, instead of using traditional PDE numerical solvers, researchers have started to solve direct and inverse PDE problems using machine learning algorithms. For example, [13] solved PDEs with deep learning; they provided some examples like the Black-Scholes equation, and the Hamilton-Jacobi-Bellman eq-

uation. [14] introduced a new feed-forward deep network, PDE-Net, to solve PDEs. Additionally, a new deep learning framework called Physics-Informed Neural Network (PINN) has been used to solve PDEs. For this method, [15] provided the general structure and algorithms, and [16] obtained numerical solutions by using a Bayesian PINN (B-PINN) in both forward and inverse nonlinear directions with noisy data.

As to parameter estimation, [17] proposed two methods (parameter cascading and Bayesian approach) to estimate the parameters in PDEs, [18] estimated the parameters of a laminar flow model with the help of machine learning, and [19] designed a data-driven model that can be used to discover the key parameters in PDEs through time series measurements in the spatial domain.

Fisher's equation combines the logistic population growth model, $\dot{u} = ku(1-u/C)$, where k is the growth rate at low densities and C is the carrying capacity, with a diffusion equation in which density $u(x,t)$ is a function of space x and time t and satisfies the PDE $u_t = Du_{xx} + ku(1-u/C)$. Upon scaling u with C, time t with k^{-1} and x with $(D/k)^{1/2}$, the dimensionless form of Fisher's equation becomes $u_t = u_{xx} + u(1-u)$. By introducing the flux $q(x,t)$, we can rewrite this scaled reaction-diffusion equation in the form

$$\begin{cases} u_t + q_x = F(u), \\ u_x = -q, \end{cases} \quad (-\infty < x < \infty, t > 0), \tag{1}$$

where $F(u) = u(1-u)(1-\gamma_1 u)(1-\gamma_2 u)\cdots$. Here, we have modified the logistic expression for the growth rate $F(u)$ by allowing for higher order polynomials to see their effect on the parameter identification problem.

The second equation in the system above relates the flux q, instantaneously, to the negative of the gradient of the density u, which is common in diffusion models (such as Fick's law of diffusion). However, in some cases, it may take a finite time for the flux to adjust to the variations in the gradient. To account for this relaxation process, one can introduce a dimensionless relaxation time ε in the flux relation. The new system then reads:

$$\begin{cases} u_t + q_x = F(u), \\ \varepsilon q_t + u_x = -q. \end{cases} \quad (-\infty < x < \infty, t > 0, \varepsilon \geq 0). \tag{2}$$

By eliminating $q(x,t)$, we can also write this system as a single second-order equation in the form:

$$u_{xx} - \varepsilon u_{tt} = u_t\left[1 - \varepsilon F'(u)\right] - F(u). \tag{3}$$

In the special case $\varepsilon = 0$ and $F(u) = u(1-u)$, the equation becomes the original Fisher's equation which is a parabolic PDE; when $\varepsilon > 0$, it becomes hyperbolic.

In this paper, we focus on traveling wave solutions of this system. To that end, we take both $u(x,t)$ and $q(x,t)$ to depend on the single variable $\zeta = x - Vt$ with velocity $V > 0$ in the system (2) and equation (3). The result is a system of coupled first-order ordinary differential equations:

$$\begin{cases} -Vu' + q' = F(u), \\ u' - V\varepsilon q' = -q, \end{cases} \quad (-\infty < \zeta < \infty, V > 0, \varepsilon \geq 0), \tag{4}$$

and a second order ODE:

$$(\varepsilon V^2 - 1)u'' = [1 - \varepsilon F'(u)]Vu' + F(u), \tag{5}$$

where $F(u) = u(1-u)(1-\gamma_1 u)\cdots$. These equations are our starting points to analyze the properties of the traveling wave solutions.

In this article, we mostly focus on the ODE system (4), and the second-order ODE (5). In our experiments, we find that larger values of ζ are needed to show a significant change in the solution u as the order of nonlinearity increases. For example, if we set γ_1 and γ_2 to 0.6 and 0.8, respectively, the wave front occurs when ζ is around 750. Most of the time, we restrict our attention on the nonlinearity $F(u) = u(1-u)(1-\gamma u)$ with just a single parameter γ. As such, our modified Fisher's equation includes three parameters: ε, V, and γ, which we try to identify from a given numerical solutions of the system (2) or the Equation (5).

The traveling wave fronts that are of interest to us have a u profile that smoothly goes from 1 to 0 (two of the fixed points of the system) as ζ ranges from $-\infty$ to ∞. To justify the existence of such traveling waves, we first carry out a stability analysis of the nonlinear dynamical system for arbitrary values of V, ε, and γ. Then, we consider the inverse problems of finding the parameters given some numerical solution. We consider two different cases of the inverse problem: one in which both profiles $u(\zeta)$ and $q(\zeta)$ are available, and another where only one of these, namely $u(\zeta)$ is known. We refer to these as Type 1 and Type 2 inverse problems, respectively. For Type 1 problems, we can determine the parameters V, ε, and γ through the use of appropriate optimization problems. For Type 2 problems, we recover parameters V, ε, and γ by means of machine learning methods. In this case, we first use an ensemble method to get a more accurate initial value for velocity V, which is crucial for enhancing the performance of the optimization problem to determine the remaining two parameters ε and γ.

2. Methods

In this section, we focus on Equations (4) and (5). We first take $F(u) = u(1-u)(1-\gamma_1 u)(1-\gamma_2 u)\cdots$ where $0 \leq \gamma_i < 1$, and u is a proper density field that lies within the interval [0, 1]. Later, we restrict to the case where $F(u)$ is cubic with only one extra parameter γ. After the traveling-wave transformation and with the help of a stability analysis in the phase plane, we identify the proper range of parameters V and ε, which turn out to be $2 \leq V$ and $0 \leq \varepsilon < 1/V^2$, for which traveling waves are obtained, keeping the density between 0 and 1. We then generate a large set of numerical solutions using these values to train machine learning algorithms. Next, assuming that some numerical traveling wave solution is given (including both u and q profiles or the u pro-

file alone), we design parameter estimation models to estimate the velocity V, relaxation time ε, and parameter γ that gave rise to that numerical solution.

2.1. Stability Analysis

Stability analysis of fixed points in the phase plane is a powerful approach for qualitative understanding of the behavior of a dynamical system. The equilibrium points can typically be classified as linearly stable, unstable, or semi-stable. Although those classifications are local, it is often possible to infer qualitatively the global trajectories in the phase plane through further analysis. For instance a trajectory connecting two separate fixed points can represent a solution that goes from one constant value to another. However, due to the complex nature of dynamics in the presence of multiple fixed points and nonlinearities, while linear stability analysis is a valuable tool, it may not be suitable for all systems or provide a full picture of the dynamics.

Equation (4) can be written in matrix-vector notation:

$$\begin{bmatrix} -V & 1 \\ 1 & -\varepsilon V \end{bmatrix} \begin{bmatrix} u_\zeta \\ q_\zeta \end{bmatrix} \equiv \boldsymbol{A} \begin{bmatrix} u_\zeta \\ q_\zeta \end{bmatrix} = \begin{bmatrix} F(u) \\ -q \end{bmatrix}. \tag{6}$$

Matrix $\boldsymbol{A}$ is invertible provided that εV^2 is not equal to 1. We linearize the system about fixed points $(u,q)=(u_i,0)$ by introducing small perturbations $\xi(\zeta)$ and $\eta(\zeta)$ through $u=u_i+\xi(\zeta)$ and $q=0+\eta(\zeta)$. The linearized system becomes:

$$\begin{aligned} \begin{bmatrix} \xi \\ \eta \end{bmatrix}' &= \begin{bmatrix} -V & 1 \\ 1 & -\varepsilon V \end{bmatrix}^{-1} \begin{bmatrix} F(u_i+\xi) \\ -\eta \end{bmatrix} \\ &= \frac{1}{1-\varepsilon V^2} \begin{bmatrix} \varepsilon F'(u_i) V & -1 \\ F'(u_i) & -V \end{bmatrix} \begin{bmatrix} \xi \\ \eta \end{bmatrix} \\ &\equiv \boldsymbol{C} \begin{bmatrix} \xi \\ \eta \end{bmatrix}. \end{aligned}$$

The eigenvalues of $\boldsymbol{C}$ determine the stability of the fixed points; to simplify the notation, let $\beta = F'(u_i)$; the two eigenvalues at each fixed point are then given by

$$\lambda = \frac{V}{2(1-\varepsilon V^2)} \left[(\varepsilon\beta - 1) \pm \sqrt{(1+\varepsilon\beta)^2 - \frac{4\beta}{V^2}} \right]. \tag{7}$$

The function $F(u)$ has roots at $u=0$ and $u=1$ in addition to those introduced by the additional factors; thus there are always two equilibrium points at 0 and 1. At these points $\beta(0)=1$ and $\beta(1)=-\prod_i(1-\gamma_i)$. We seek traveling wave solutions in which u varies smoothly from 1 to 0 as ζ goes from $-\infty$ to ∞. As such $u=0$ must be a stable node (if it were a spiral, u would oscillate as it approached 0 and become negative at times), while $u=1$ can be an unstable node or saddle. The corresponding eigenvalues for these equilibrium points are given as:

$$\lambda(0)=\frac{V}{2\left(1-\varepsilon V^2\right)}\left[\varepsilon-1\pm\sqrt{\left(1+\varepsilon\right)^2-\frac{4}{V^2}}\right],$$

$$\lambda(1)=\frac{V}{2\left(1-\varepsilon V^2\right)}\left[-\prod_i\left(1-\gamma_i\right)\varepsilon-1\pm\sqrt{\left(1-\varepsilon\prod_i\left(1-\gamma_i\right)\right)^2+\frac{4\prod_i\left(1-\gamma_i\right)}{V^2}}\right].$$

When $\varepsilon=0$, in order for the eigenvalues $\lambda(0)$ not to be complex which would make the fixed point a spiral, V has to be greater than 2. For any V, we must keep ε less than $1/V^2$ in order to avoid a singularity and change of sign in the eigenvalues. In order for u to have a smooth transition between 1 and 0, the two key parameters in our system, velocity V and relaxation time ε, can be restricted to be within the respective ranges: $2\le V$, and $0\le\varepsilon<1/V^2$.

2.2. Parameter Estimation for the Cubic Non-Linearity

Using $F(u)=u(1-u)(1-\gamma u)$, and applying constraints for ε and V, the scaled system (4) reduce to:

$$\begin{cases}-Vu'+q'=u(1-u)(1-\gamma u),\\ u'-\varepsilon Vq'=-q,\end{cases}\quad\left(2\le V,0\le\varepsilon<\frac{1}{V^2},0\le\gamma<1\right).\tag{8}$$

Assuming that a numerical solution is given over some range, we design several approaches to determine the parameters for two types of inverse problems: Type 1 where numerical solutions u and q (density and flux) are both given with u in the range $[0.95,0.05]$; Type 2 where the numerical solution u (density only) is given in the range $[0.95,0.05]$. A good approximation of the first and second derivatives is essential for getting high accuracy in the parameter estimates. We use the following high order central difference formulae for this purpose:

$$\begin{aligned}60hf'(x_j)&=-f_{j-3}+9f_{j-2}-45f_{j-1}+45f_{j+1}-9f_{j+2}+f_{j+3}+O(h^6),\\ 180h^2f''(x_j)&=2f_{j-3}-27f_{j-2}+270f_{j-1}-490f_j+270f_{j+1}\\&\quad-27f_{j+2}+2f_{j+3}+O(h^6).\end{aligned}\tag{9}$$

To characterize the performance in our results, we will use the Root Mean Square Error (RMSE) which is defined as:

$$\mathrm{RMSE}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i-y_i)^2},$$

where y_i is the true value, and $\hat{y}_i$ is the estimated value of y_i.

2.2.1. Type 1 Inverse Problem

Assume that both numerical solutions u and q with u in the range $[0.95,0.05]$ are given. Solving the first equation for V in (8) yields:
$V=-\left[u(1-u)(1-\gamma u)-q'\right]/u'$. Since V is supposed to be a constant (independent of ζ if the estimated $\hat{\gamma}$ is equal or close to the true value γ, there should be almost no variance in $\hat{V}$ calculated at different ζ locations. In

other words, if we calculate the variance in V (thought of as a function of ζ) as $\hat{\gamma}$ is varied, it should be very close to 0 when $\hat{\gamma}$ is close to the true value γ. Therefore, we generate our first model which solves the following optimization problem:

$$\begin{aligned} &\text{minimize}: g(\gamma) = \text{Var}\left(\frac{q' - u(1-u)(1-\gamma u)}{u'}\right), \\ &\text{subject to}: 0 \le \gamma < 1. \end{aligned} \tag{10}$$

It is hard to see the convexity of $g(\gamma)$ directly, but we can transform this problem into the following quadratic form in which $g(\gamma)$ is strongly convex (See the Appendix for details):

$$\begin{aligned} &\text{minimize}: g(\gamma) = \frac{1}{n-1}\left[\gamma^2 c_1 - 2\gamma c_2 + c_3\right], \\ &\text{subject to}: c_1 = \boldsymbol{v}_2^{\mathrm{T}} D \boldsymbol{v}_2,\ c_2 = \boldsymbol{v}_1^{\mathrm{T}} D \boldsymbol{v}_2,\ c_3 = \boldsymbol{v}_1^{\mathrm{T}} D \boldsymbol{v}_1, \end{aligned}$$

where $D = \left(\boldsymbol{I} - \boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}/n\right)^{\mathrm{T}}\left(\boldsymbol{I} - \boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}/n\right)$, $\boldsymbol{v}_1 = \left(q' + u^2 - u\right)/u'$, $\boldsymbol{v}_2 = \left(u^3 - u^2\right)/u'$, and $\boldsymbol{1}$ is a column vector of ones. Here, n is the number of data points in the vector of numerical solution values u.

After obtaining $\hat{\gamma}$, we can estimate ε and V by following equations obtained by averaging the appropriate pointwise values of the parameters along the numerical traveling wave profile:

$$\hat{V} = \frac{1}{n}\sum_{i=1}^{n} \frac{q_i' - u_i(1-u_i)(1-\hat{\gamma}u_i)}{u_i'},$$

$$\hat{\varepsilon} = \frac{1}{n}\sum_{i=1}^{n} \frac{u_i' + q_i'}{V_i q_i'} = \frac{1}{n}\sum_{i=1}^{n} \frac{(u_i' + q_i')u_i'}{\left(q_i' - u_i(1-u_i)(1-\hat{\gamma}u_i)\right)q_i'}.$$

2.2.2. Type 2 Inverse Problem

Assuming that only some numerical solution of u is given in the range $u \in [0.95, 0.05]$ without having the corresponding flux q, we devise an alternative method for estimating the parameters. The solution can be generated by solving the second order ODE (5). Considering the different forms of numerical equations, it can sometimes estimate u' too. Therefore, it is essential to come up with other methods where the only input is a part of the u solution. To do that, we first solve for γ from the equation

$$\left(\varepsilon V^2 - 1\right)u'' = \left[1 - \varepsilon\left(3\gamma u^2 - 2(\gamma+1)u + 1\right)\right]Vu' + u(1-u)(1-\gamma u)$$

treating V and ε as given. This yields:

$$\begin{aligned} \gamma &= \frac{u'' + u - u^2 + Vu' - \varepsilon\left(V^2 u'' + Vu' - 2Vu'u\right)}{u\left(u - u^2\right) + \varepsilon Vu'u(3u-2)} \\ &= \left[u'' + u - u^2 + Vu' + \frac{\left(u - u^2\right)\left(Vu'' + u' - 2u'u\right)}{u'(3u-2)}\right]\frac{1}{u\left(u - u^2\right) + \varepsilon Vu'u(3u-2)} \\ &\quad - \frac{Vu'' + u' - 2u'u}{u'u(3u-2)}. \end{aligned}$$

To simply the notation, we define four vectors as follows: $\boldsymbol{a} = u'' + u - u^2 + Vu' + \left(u - u^2\right)\left(Vu'' + u' - 2u'u\right)/u'\left(3u - 2\right)$, $\boldsymbol{b} = u\left(u - u^2\right)$, $\boldsymbol{c} = Vu'u\left(3u - 2\right)$, and $\boldsymbol{d} = -\left(Vu'' + u' - 2u'u\right)/\left[u'u\left(3u - 2\right)\right]$. Ignoring the vector sign for ease of notation, we can write $\gamma = a/(b + c\varepsilon) + d$. Next, like in the previous case, we regard this as the following two-variable optimization problem:

$$\begin{aligned} &\text{minimize}: f\left(V,\varepsilon\right) = \text{Var}\left(\frac{a}{b + c\varepsilon} + d\right), \\ &\text{subject to}: 2 \le V, 0 \le \varepsilon < 0.25. \end{aligned} \tag{11}$$

It is clear that $f\left(V,\varepsilon\right)$ is not a convex function. However, we found that if V is equal to or around the true velocity, we can transform this equation into the following form where $w\left(\varepsilon\right)$ is a strongly convex function (see the Appendix for details).

$$\begin{aligned} &\text{minimize}: w\left(\varepsilon\right) = \boldsymbol{h}_1^{\mathrm{T}} D\boldsymbol{h}_1\varepsilon^2 + 2\boldsymbol{d}D\boldsymbol{h}_1\varepsilon, \\ &\text{subject to}: 0 \le \varepsilon < \frac{1}{V^2}, \end{aligned}$$

where $D = \left(\boldsymbol{I} - \boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}/n\right)^{\mathrm{T}}\left(\boldsymbol{I} - \boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}/n\right)$, and $\boldsymbol{h}_1 = -\left[a_1c_1/b_1^2, \cdots, a_nc_n/b_n^2\right]^{\mathrm{T}}$.

Therefore, if there is a good initial guess for the velocity, the solution of the optimization problem (11) always exists and is unique. We designed an ensemble model to make such predictions. Also, in addition to solving (11), we use an embedding technique to estimate the parameters.

2.2.3. Variable Selection

To begin the design of the ensemble model, the first step is variable selection. In our model, we generate our data set from the u solutions. Since $u\left(\zeta\right)$ is a traveling front, it can be shifted right or left without changing form, so the origin in ζ is arbitrary. It is therefore better to work with variables that are independent

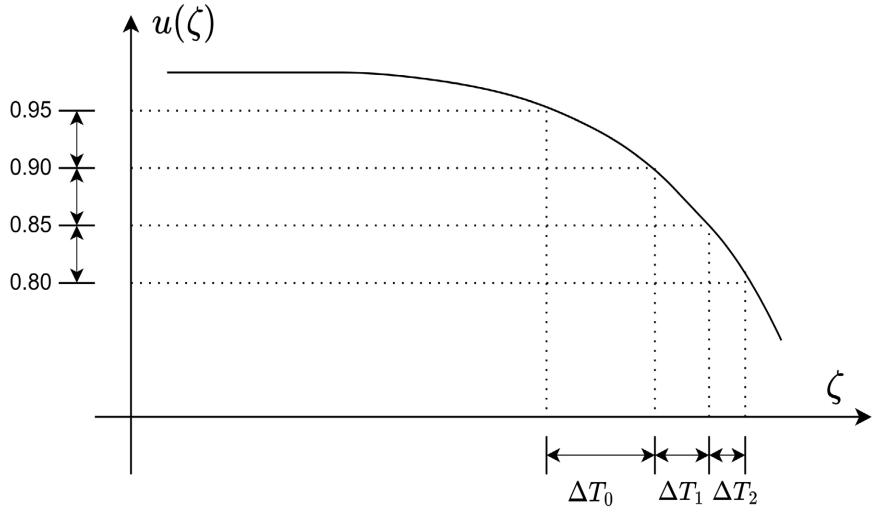

Figure 1. The process of the first three intercept variables that $U = \left\{0.95, 0.85, \cdots, 0.05\right\}$. After estimating all $\hat{\zeta}_i$, we generate the intercept variable set: $T = \left\{\Delta T_0, \Delta T_1, \cdots, \Delta T_{10}\right\}$.

of such shifts. Here is the main process we used to generate our data set from a numerical solution $u(\zeta)$, as shown in Figure 1:

1. Define a set of fixed range intervals along the u-axis, namely $U := \{U_1, U_2, \cdots, U_k\}$; the numerical solution is given by $u = \{u_1, u_2, \cdots, u_n\}$, and its corresponding variables set is $\zeta = \{\zeta_1, \zeta_2, \cdots, \zeta_n\}$.

2. Since most of the time, the numerical solution does not include the exact ζ_i that corresponds to U_i, we estimate it by linear interpolation: assuming the neighbors of U_i are U_i^- and U_i^+, which can be found in the solution set u and their corresponding ζ values: ζ_i^- and ζ_i^+. The linear interpolation estimate of ζ_i is given by $\hat{\zeta}_i = \left(U_i - U_i^-\right)\left(\zeta_i^+ - \zeta_i^-\right) / \left(U_i^+ - U_i^-\right) + \zeta_i^-$.

3. After estimating all $\hat{\zeta}_i$, to avoid that `horizontally shift' issue, design the intercept variables as the increments:
$T = \left\{\hat{\zeta}_1 - \hat{\zeta}_0, \hat{\zeta}_2 - \hat{\zeta}_1, \cdots, \hat{\zeta}_k - \hat{\zeta}_{k-1}\right\} := \left\{\Delta T_0, \Delta T_1, \cdots, \Delta T_{k-1}\right\}$.

4. Repeat steps 1 to 3 until the intercept variables of all numerical solutions are generated.

Table 1 shows sample input data generated from four intercept variables in U.

2.2.4. Ensemble Model with Linear Machine Learning Algorithms

In our approach, prediction of the traveling wave velocity is a regression task; therefore, we propose using existing ensemble techniques along with the following common linear models:

1) **Linear regression:** This is one of the most famous methods, which is easy to calculate and apply. It considers the problem of finding the column vector β in $y = X\beta + \varepsilon$, which can be formulated as:

$$\min_w \left\| Xw - y \right\|_2^2,$$

where X represents the input data, w is the vector of weights, and y is the target variable. If there is a positivity constraint in linear regression, in which all coefficients must be positive except the intercept, we solve the following optimization problem:

$$\min_w \left\| Xw - y \right\|_2^2 \quad \text{subject to } w \geq \mathbf{0}.$$

2) **Ridge regression/Tikhonov regularization:** Due to multicollinearity in

Table 1. One sample variables table contains 4 intercept variables with $U = \{0.05, 0.35, 0.65, 0.95\}$, velocity V from 2 to 10, time relaxation ε from 0 to $0.99/V^2$, and $\gamma = 0$.

ΔT_0	ΔT_1	ΔT_2	ΔT_3	V	ε
3.198128	1.878075	2.082083	4.596184	2.0	0
3.186127	1.884075	2.076083	4.596184	2.0	0.001244
3.198128	1.878075	2.082083	4.596184	2.0	0.002487
...	...	...	...	...	...
19.614785	9.666387	9.720389	19.896796	10.0	0.0099

data, most people consider using ridge regression to avoid this issue by imposing a penalty on the size of the coefficients. One of the reasons it works is that it does not require unbiased estimators. This problem can be written as:

$$\min_w \|Xw - y\|_2^2 + \alpha \|w\|_2^2,$$

where α is a non-negative parameter that controls the degree of shrinkage.

3) **Lasso regression:** "Lasso" stands for Least Absolute Shrinkage and Selection Operator. Similarly to Ridge regression, Lasso regression adds a penalty term to avoid multicollinearity problems, but unlike Ridge regression, it uses the L_1 norm rather than the L_2 norm. We can express this problem as:

$$\min_w \|Xw - y\|_2^2 + \alpha \|w\|_1,$$

where X represents the input data, w is the vector of weights, y represents the target variable, and the non-negative parameter α controls the degree of shrinkage.

4) **Bayesian ridge regression:** This method assumes that the output y comes from a probability distribution, rather than having a single value. We express this as:

$$P(y \mid X, w, \alpha) = N(y \mid Xw, \alpha),$$

where α is a random variable estimated from the data. In practice, people use gamma distributions which are conjugated prior to the precision of the Gaussian distribution (focusing on two parameters: α and λ), rather than the normal distribution directly.

5) **Bayesian Automatic Relevance Determination (ARD) regression:** This method is similar to Bayesian Ridge Regression but with sparse weights, containing a weight matrix without spherical Gaussian distribution. This means that its coefficients w_i can be drawn from a normal distribution with zero mean and precision λ_i:

$$p(w \mid \lambda) = N(w \mid 0, A^{-1}),$$

where A is a positive definite diagonal matrix with $diag(A) = \{\lambda_1, \lambda_2, \cdots, \lambda_n\}$.

6) **Stochastic Gradient Descent (SGD) Regression:** This is another popular gradient descent method. However, compared to regular gradient descent, it induces randomness in the process. Due to this characteristic, SGD can randomly pick one data point from the whole set at each iteration, which reduces the computation time dramatically.

The reason we chose these regression methods is that although linear regression is easy to apply, it is sensitive to outliers and is prone to noise and overfitting. Therefore, to reduce the effects of such overfitting, we introduce other regression methods such as Ridge and Lasso which add a penalty term, Bayesian Ridge which provides a confidence bound on predictions, and ARD regression which provides a sparser solution than Bayesian regression. Additionally, we also applied the SGD regression due to its high speed of execution.

2.2.5. Embedding Method Process

By using the previous ensemble model, the predicted velocity is very close to the true velocity, but its performance is still not good enough for relaxation time ε and parameter γ prediction. Therefore, instead of using one ensemble model to predict all parameters, we applied an embedding method. Roughly, the process can be shown as is shown in Figure 2.

There are several advantages to our approach:

1) **Low cost:** In our model, during the training process, the range of velocities typically falls between 2 and 10, and a good initial guess of the velocity is one with an error less than about 10^{-3} compared to the true velocity. However, designing a data set with velocities that have around 10^{-3} increments, considering the increments of ε and γ, might give rise to more than one hundred thousand different cases which makes the learning process too long to be practical. Our approach reduces the computational time and cost.

2) **Ease of application:** Compared to optimization problem solvers, the ensemble model approach is relatively easy to apply. We have already defined the structure of the ensemble model, so we can simply train another one using the embedded data.

2.2.6. Main Process for Type 2 Inverse Problems

In summary, similar to the previous case, we use an ensemble model to predict a good initial value of the traveling wave velocity. After that, we not only solve the optimization problem (11), but also apply the embedding method to predict the parameters. The whole process can be summarized in Figure 3.

3. Results

In this section, we present some key numerical solutions of the traveling wave

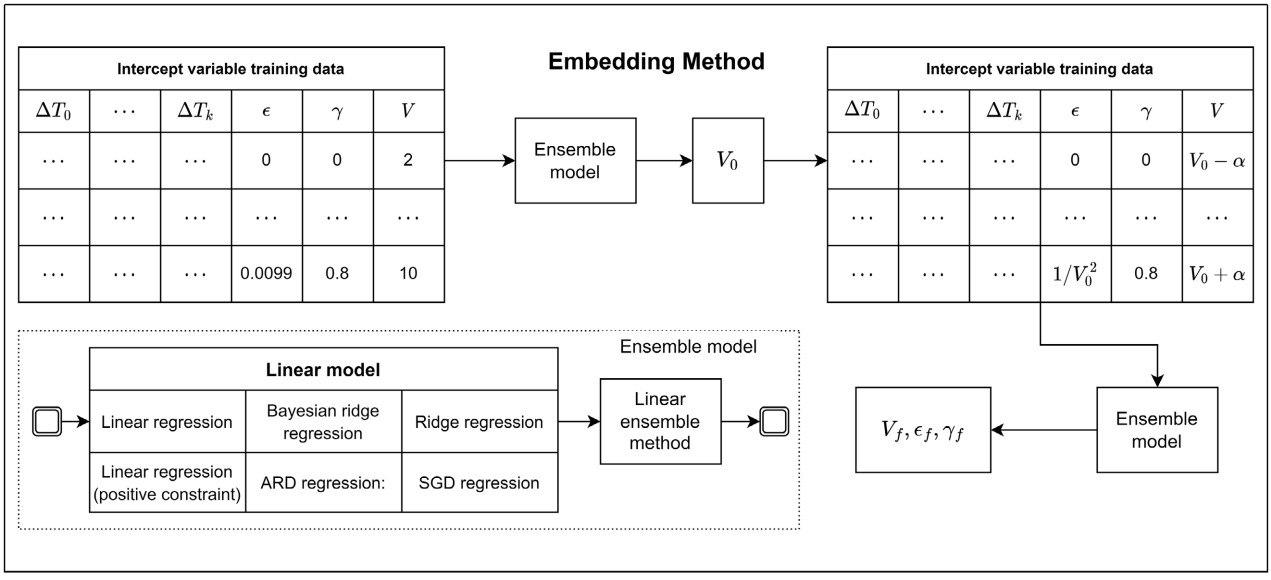

Figure 2. A schematic diagram of the embedding process: the original training set considers the velocity range from 2 to 10. After V_0 prediction, generate new training set with a velocity from $V_0-\alpha$ to $V_0+\alpha$ where α is a small value.

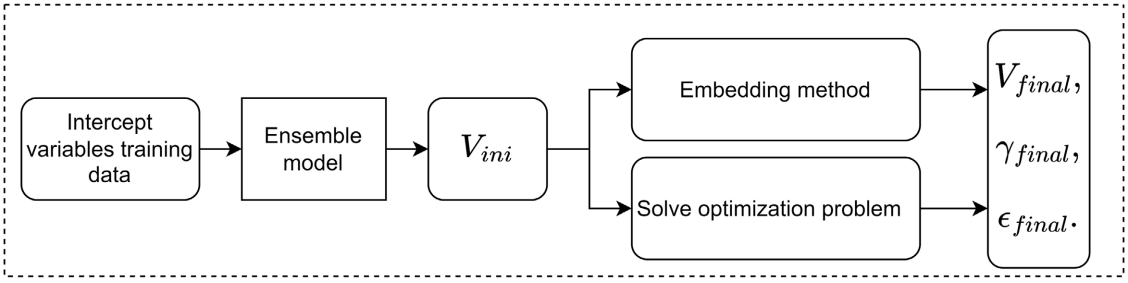

Figure 3. A schematic representation of the type 2 inverse problem.

problem, consider it in the phase plane, and characterize the performance of the parameter estimation methods from a given solution. We include both types of inverse problems where both u and q are available or when only the u profile is given.

3.1. Traveling Wave Solutions

To solve the nonlinear Equation (8), we can use standard ODE solvers such as Python SciPy's "odeint". The pair of coupled equations can be written in terms of u and u' (which constitutes the phase plane) or in terms of u and q:

- In terms of density and its derivative:

$$\begin{cases} u' = w, \\ w' = \dfrac{1}{\varepsilon V^2 - 1}\left[\left(1 - \varepsilon F'(u)\right)Vw + F(u)\right] \end{cases} \quad 2 \le V,\ 0 \le \varepsilon < 1/V^2.$$

- In terms of density and flux when $\varepsilon = 0$:

$$\begin{cases} u' = -q \\ q' = F(u) - Vq \end{cases} \quad 2 \le V,\ 0 \le \gamma < 1.$$

- In terms of density and flux when $\varepsilon \neq 0$:

$$\begin{cases} u' = \dfrac{1}{1 - \varepsilon V^2}\left[\varepsilon V F(u) - q\right] \\ q' = \dfrac{1}{\varepsilon V}\left[\dfrac{1}{1 - \varepsilon V^2}\left[\varepsilon V F(u) - q\right] + q\right] \end{cases} \quad 2 \le V,\ 0 < \varepsilon < 1/V^2.$$

Figure 4 displays some examples of the phase plane and numerical solutions of u with $F(u) = u(1-u)$ and $\varepsilon = 0$. The fixed points are along the horizontal axis in the left panels at $u = 0$ and $u = 1$. They are marked with red dots in the figure. The heteroclinic trajectory connecting them in the phase plane is the travelling wave solution along which u decreases from 1 to 0. It is clear that when $0 < V < 2$, the point $u = 0$ is a stable spiral and $u = 1$ is a saddle point, and when $V > 2$, $u = 0$ become a stable node while $u = 1$ remains a saddle point. In both cases, starting very close to the saddle point along the heteroclinic trajectory and moving toward the attracting fixed point, u starts at 1, but when the attracting fixed point is a spiral, the profile oscillates about 0 before ending up at that fixed point (top right panel), while when the fixed point at zero is a node, the profile decreases monotonically without oscillation (bottom right panel). If u

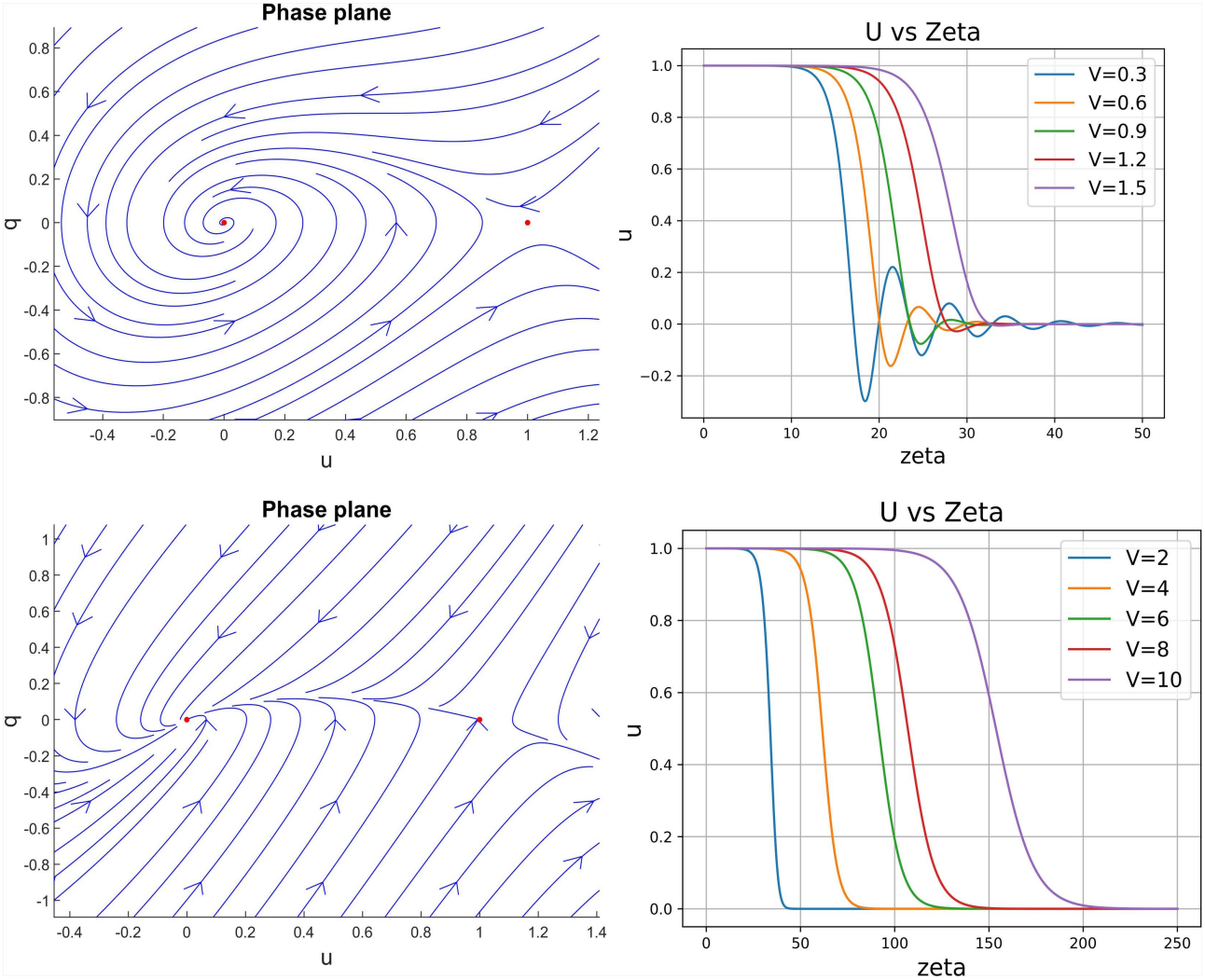

Figure 4. This figure shows two examples of phase plane and numerical solution of system (4) where the non-linear function $F(u)=u(1-u)$ with $\varepsilon=0$ and different velocity V. The first example shows the phase plane with the case $V=0.5$, and the numerical solution with the velocity $V=0.3, 0.6, 0.9, 1.2, 1.5$; The second example shows the phase plane where $V=2$ and its numerical solution with the velocity $V=2, 4, 6, 8, 10$.

represents a density variable that cannot be negative, only the latter case is physical. That is the reason we restrict our attention to the traveling wave velocity range $V>2$.

In **Figure 5** we show another solution using the (u,q) formulation when $V=10$, $\varepsilon=0$ and $\gamma=0.8$. When starting very close to $u=1$, it takes a much longer range of ζ to see the variation of u toward zero in this case.

3.2. Parameter Estimation

3.2.1. Type 1 Inverse Problem

The Type 1 problem is the one where both u and q profiles are given. As explained earlier, we take that portion of the density profile $u(\zeta)$ for which u is from 0.95 to 0.05. In our tests, we focus on the parameter ranges given in **Table 2**.

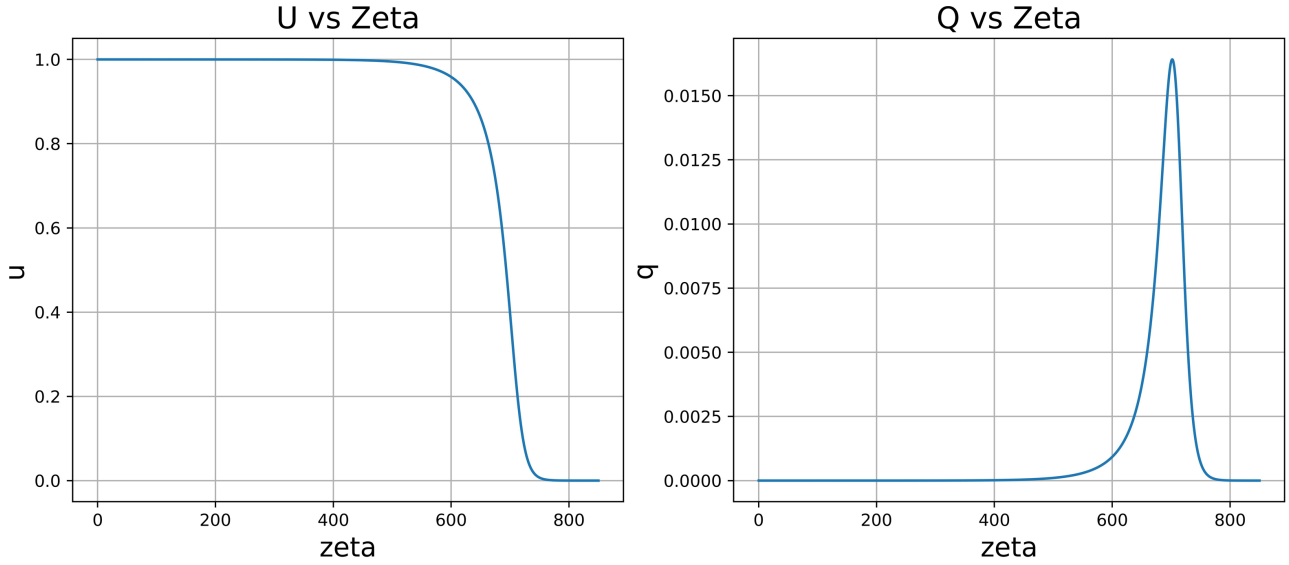

Figure 5. Numerical solution of u and q with $V=10$, $\varepsilon=0$, and $\gamma=0.8$.

Table 2. Type 1 inverse problem parameters test table.

	Range	Number
Velocity V	$2 \le V \le 10$	50
Relaxation time ε	$0 \le \varepsilon < 1/V^2$	5
Parameter γ	$0 \le \gamma \le 0.8$	20

Table 3. Type 1 inverse problem performance table.

	RMSE
Velocity V	4.368×10^{-6}
Relaxation time ε	7.479×10^{-5}
Parameter γ	1.356×10^{-6}

Overall, there are 5000 cases in this experiment, and for each case, there may be a few hundred to a few thousand points for the specific range of u and the corresponding q. Table 3 gives the root mean square error values in the estimates of the three parameters over all the cases. The performance is excellent. However, if the data comes not from a numerical but from a physical experiment in which only the density u and not the flux q can be measured in a traveling wave front, the parameter estimation process is more challenging. We discuss that case as the Type 2 inverse problem later in this section.

To further explore the characteristics of the method, in the following three figures, we show the performance in more detail in cases where two of the three parameters V, ε, and γ are fixed while the third varies.

In Figure 6, ε and γ are fixed while V is allowed to range from 2 to 10. The three panels compare the predicted values of the three parameters to their

true counterparts. The agreement is very good. Note that in the rightmost panel, the vertical axis scale is showing the difference between the true solution of $\gamma = 0.5$ and the predicted solution, with the range being from 0 to 10^{-6}, indicating excellent agreement. This is indicated by the notation above the top left of the plot. Figure 7 and Figure 8 show similar comparisons where ε and γ are respectively varied while the other two parameters are fixed. In the panels with the notation above the top left of the plot, the difference between true and predicted values is being plotted.

3.2.2. Type 2 Inverse Problem

In this part, in addition to characterizing the performance of the method, we also apply an epistemic uncertainty analysis on velocity. To do so, we designed

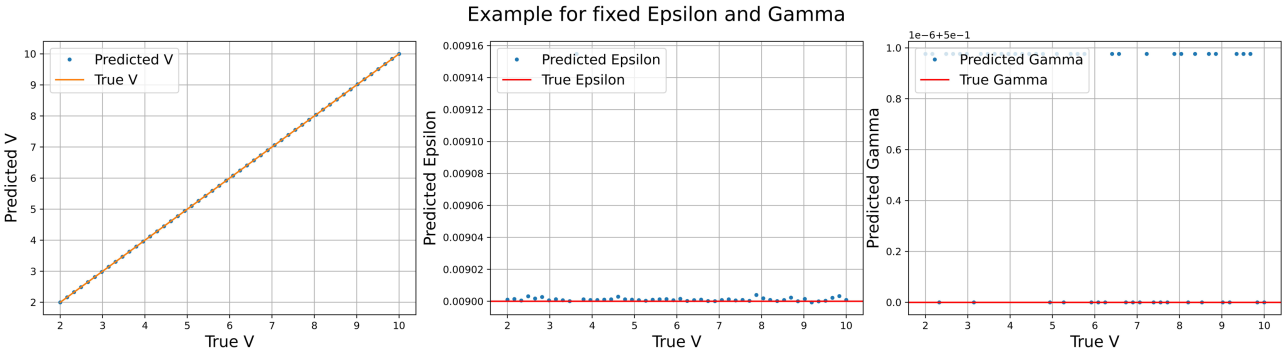

Figure 6. Type 1 inverse problem performance where $2 \le V \le 10$, $\varepsilon = 0.009$, and $\gamma = 0.5$.

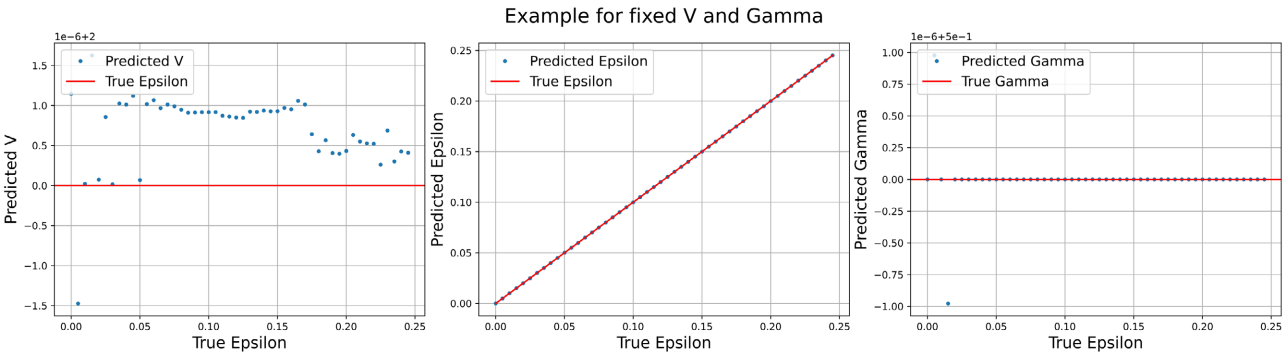

Figure 7. Type 1 inverse problem performance where $V = 2$, $0 \le \varepsilon < 1/2^2$, and $\gamma = 0.5$.

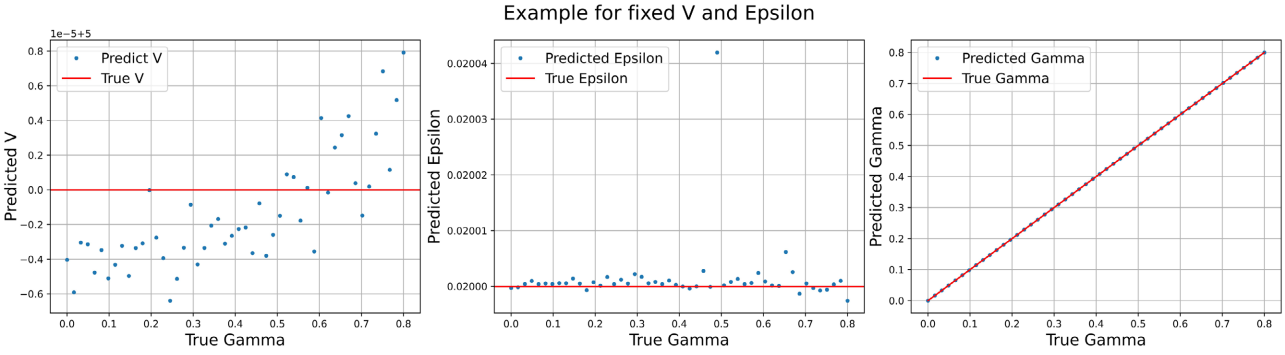

Figure 8. Type 1 inverse problem performance where $V = 5$, $\varepsilon = 0.02$, and $0 \le \gamma \le 0.8$.

two test data sets, in which the velocity of one set is within the same range as the training set but with different values, and for the other test set, the velocities are outside the training range, as listed in Table 4. The goal is to see how well the method performs in predicting the velocity values if they fall outside the range for which the method has been trained.

In this experiment, there are 2500 cases in the training set, and 300 cases inside and outside test data. For each numerical solution, we have a few hundred to a few thousand data points at the chosen range of u depending on the value of $\zeta_{\max}$ needed to obtain the full traveling front profile.

The RMSE errors for both the optimization solver (see Table 5) and the embedding method (see Table 6) are quite small, though the ones for the embedding method are one or two orders of magnitude larger. However, as we mentioned above, the time savings associated with the embedding method may make it worthwhile as a viable alternative to the original optimization approach. Both methods are successful in predicting the velocities outside the range in which they were trained. In Figures 9-11 we provide comparisons of true and predicted values of the parameters, when two of them are fixed while the third varies, as we did with the Type 1 inverse problem. Keep in mind that in the plots where there is a notation above the plot, it is the difference between true and predicted values that is being plotted. For the embedding method, the inside and

Table 4. Parameter ranges for training and test sets.

	Range	Number
Training velocity V	$2 \le V \le 5$	25
Inside test velocity V	$2 \le V \le 5$	10
Outside test velocity V	$5 \le V \le 10$	10
Relaxation time ε	$0 \le \varepsilon < 1/V^2$	5
Parameter γ	$0 \le \gamma \le 0.8$	20

Table 5. Optimization solver inside/outside test performance table.

	Inside test RMSE	Outside test RMSE
Velocity V	4.114×10^{-6}	3.767×10^{-5}
Relaxation time ε	4.255×10^{-5}	5.540×10^{-4}
Parameter γ	2.427×10^{-6}	1.123×10^{-5}

Table 6. Embedding method inside/outside test performance table.

	Inside test RMSE	Outside test RMSE
Velocity V	3.168×10^{-3}	7.461×10^{-4}
Relaxation time ε	5.910×10^{-3}	5.151×10^{-3}
Parameter γ	6.703×10^{-3}	5.318×10^{-3}

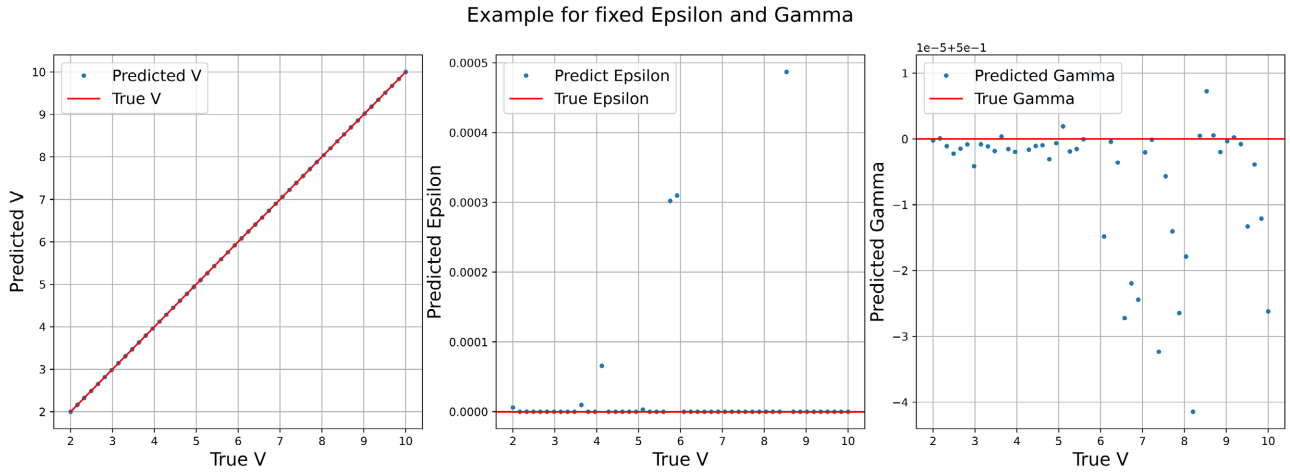

Figure 9. Type 2 inverse problem optimization method where $2 \le V \le 10$, $\varepsilon = 0$, and $\gamma = 0.5$.

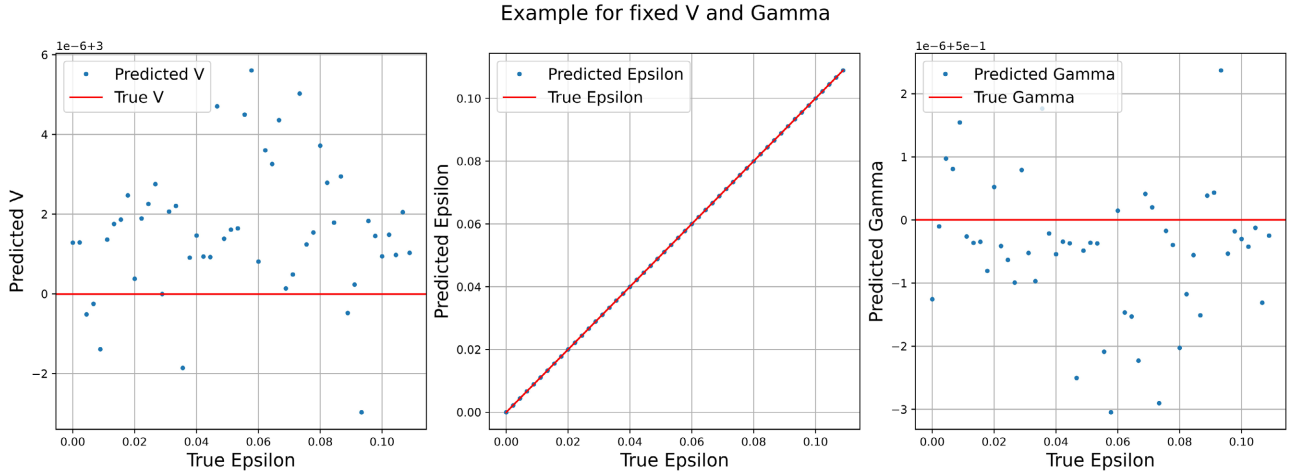

Figure 10. Type 2 inverse problem optimization method where $V = 3$, $0 \le \varepsilon < 1/3^2$, and $\gamma = 0.5$.

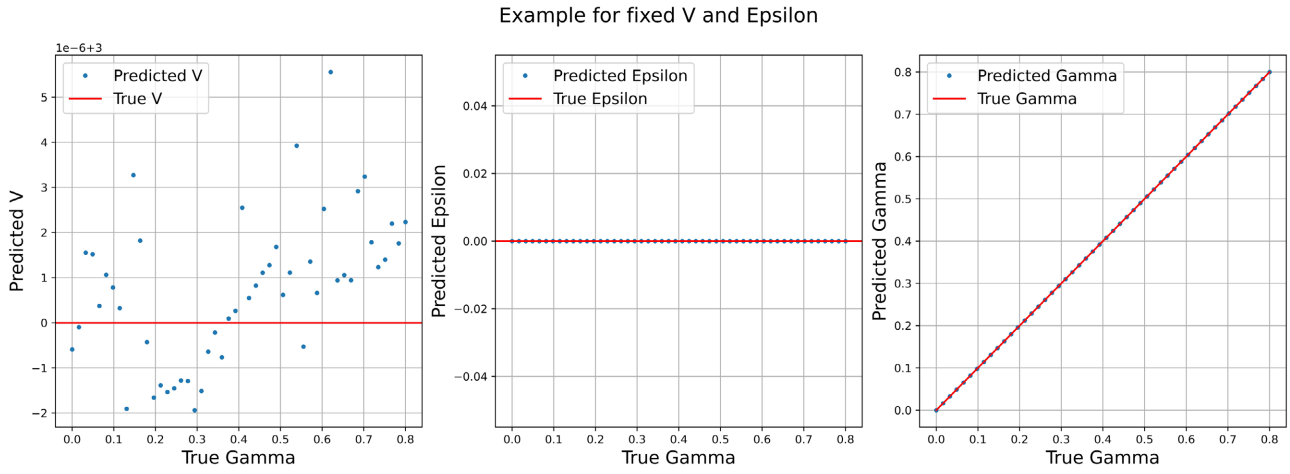

Figure 11. Type 2 inverse problem optimization method where $V = 3$, $\varepsilon = 0$, and $0 \le \gamma \le 0.8$.

outside test performances are illustrated in Figure 12 and Figure 13.

Overall, compared to the optimization problem solver, the performance of the

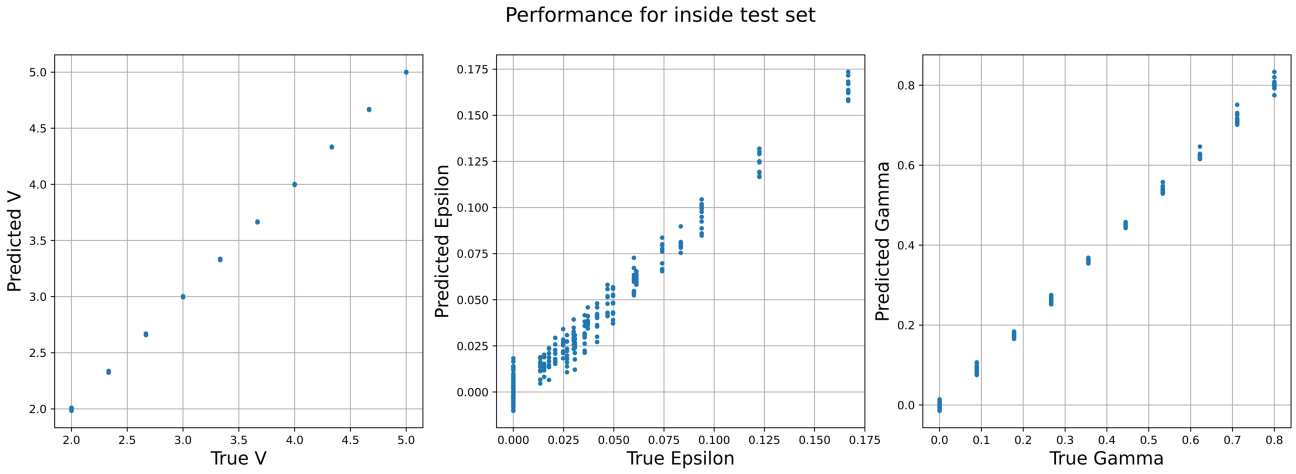

Figure 12. Type 2 inverse problem embedding method of inside test performance.

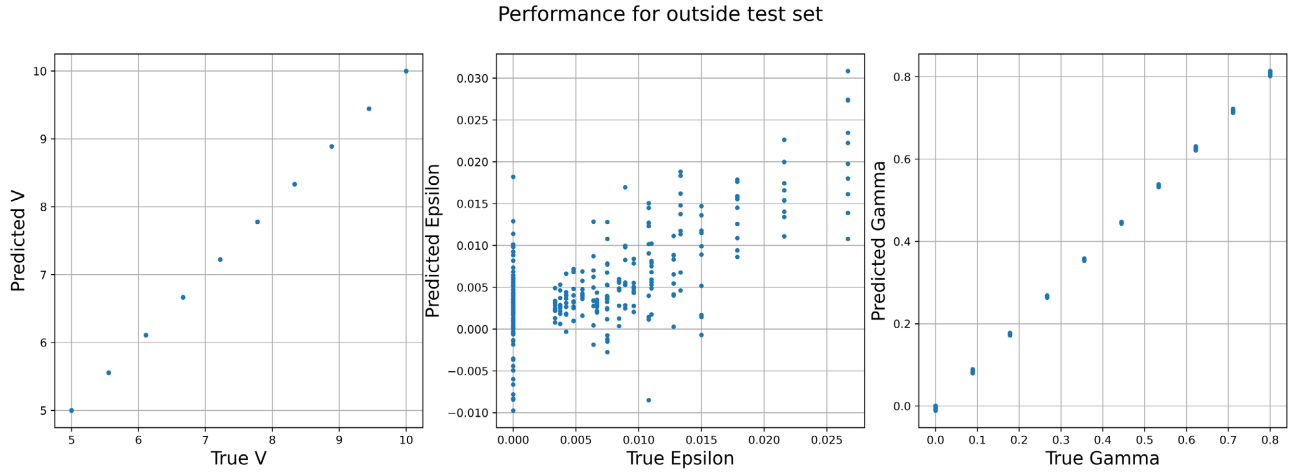

Figure 13. Type 2 inverse problem embedding method of outside uncertain test performance.

embedding method is not as good. However, both methods are acceptable with errors less than 10^{-3} for both inside and outside test cases. The embedding method is considerably faster.

4. Discussion

With a modified Fisher's partial differential equation (PDE) as our test case, we have shown how the three parameters in this PDE can be inferred once a numerical solution of the traveling wave profile is given. Ultimately, the goal is to be able to take experimental data on a traveling front and successfully determine the PDE that gave rise to that profile.

In this paper, as proof-of-principle, we used a numerical solution of the PDE as a surrogate for actual experimental data. The traveling front can be characterized in terms of a density profile that varies smoothly from one to zero across some distance (at a fixed time), or in time (at a fixed spatial location), and also a flux function that appears in the conservation equation for the density field. In most experiments, it is more likely that the density field is the only observable

quantity while the flux is not directly measured. As such, we considered two types of inverse problems, one in which both the density and flux profiles are known, and one where only the density profile is given. In the former case, a fairly robust method based on minimization of the variance of certain functions related to the density and flux could be used to infer the parameters accurately. When only the density profile is available, the inverse problem is more challenging, but using methods from machine learning, we were able to obtain satisfactory estimates of the parameters from the solution. In our experiments, the number of points in the range where significant changes in the solution u occurs plays a crucial role in achieving low error, particularly when solving optimization problems (10) and (11). Similarly, for the embedding method, building the second intercept variable training dataset with a higher density for each parameter leads to higher accuracy, but it also results in a longer training process.

Overall, the performance of parameter estimation method for both cases is acceptable. However, there is still room for improvement in our model. Firstly, we did not introduce any extra noise in our data (there is some noise during the linear interpolation approximation), therefore, to study the effect of noise, future studies can add extra white noise to the solution of u before trying to infer the parameters. Secondly, we did not have access to actual traveling front data from a physical experiment to apply our method. Future work will focus on experimental systems with traveling fronts or waves with the goal of finding the parameters in the PDE that gives rise to that wave, or the form of the PDE itself. Another research field is inverse problems based on the solutions of PDEs directly. In this kind of problem, parameters can be estimated by solving the linear equation $\Phi\zeta = u_t$, where Φ represents the library of terms related to u and its spatial derivatives that may appear on the right-hand side of the time-evolution equation for u, and ζ represents the coefficient vector for the terms in that library. After estimating the coefficients, algorithms used to solve the inverse problem can be used to verify the accuracy of the solution.

Acknowledgements

We are deeply grateful to Professor Marina Chugunova for her helpful comments on this paper.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Fisher, R.A. (1930) The Genetical Theory of Natural Selection. Clarendon, Oxford. https://doi.org/10.5962/bhl.title.27468

[2] Canosa, J. (1973) On a Nonlinear Diffusion Equation Describing Population Growth. *IBM Journal of Research and Development*, **17**, 307-313. https://doi.org/10.1147/rd.174.0307

[3] Murray, J.D. (1989) Mathematical Biology. Springer, New York.

https://doi.org/10.1007/978-3-662-08539-4

[4] Ablowitz, M.J. and Zeppetella, A. (1979) Explicit Solutions of Fisher's Equation for a Special Wave Speed. *Bulletin of Mathematical Biology*, **41**, 835-840. https://doi.org/10.1016/S0092-8240(79)80020-8

[5] Brazhnik, P.K. and Tyson, J.J. (1999) On Traveling Wave Solutions of Fisher's Equation in Two Spatial Dimensions. *SIAM Journal on Applied Mathematics*, **60**, 371-391. https://doi.org/10.1137/S0036139997325497

[6] Ahmad, H., Seadawy, A.R., Khan, T.A. and Thounthong, P. (2020) Analytic Approximate Solutions for Some Nonlinear Parabolic Dynamical Wave Equations. *Journal of Taibah University for Science*, **14**, 346-358. https://doi.org/10.1080/16583655.2020.1741943

[7] Tamsir, M., Dhiman, N. and Srivastava, V. (2017) Cubic Trigonometric b-Spline Differential Quadrature Method for Numerical Treatment of Fisher's Reaction-Diffusion Equations. *Alexandria Engineering Journal*, **57**, 2019-2026. https://doi.org/10.1016/j.aej.2017.05.007

[8] Hasnain, S. and Saqib, M. (2017) Numerical Study of One Dimensional Fishers KPP Equation with Finite Difference Schemes. *American Journal of Computational Mathematics*, **7**, 70-83. https://doi.org/10.4236/ajcm.2017.71006

[9] Saad, K.M., Khader, M.M., Gómez-Aguilar, J.F. and Baleanu, D. (2019) Numerical Solutions of the Fractional Fisher's Type Equations with Atangana-Baleanu Fractional Derivative by Using Spectral Collocation Methods. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **29**, Article ID: 023116. https://doi.org/10.1063/1.5086771

[10] Shen, W.S., Zhang, C.J. and Zhang, J. (2011) Relaxation Method for Unsteady Convection-Diffusion Equations. *Computers & Mathematics with Applications*, **61**, 908-920. https://doi.org/10.1016/j.camwa.2010.12.039

[11] Delgado, V.M., Gómez-Aguilar, J.F. and Taneco-Hernández, M. (2019) Analytical Solution of the Time Fractional Diffusion Equation and Fractional Convection-Diffusion Equation. *Revista Mexicana de Fisica*, **65**, 82-88. https://doi.org/10.31349/RevMexFis.65.82

[12] Leach, J. and Bassom, A. (2021) Large-Time Solutions of a Class of Scalar, Nonlinear Hyperbolic Reaction-Diffusion Equations. *Journal of Engineering Mathematics*, **130**, Article No. 2. https://doi.org/10.1007/s10665-021-10159-7

[13] Han, J.Q., Jentzen, A., *et al.* (2018) Solving High-Dimensional Partial Differential Equations Using Deep Learning. *Proceedings of the National Academy of Sciences*, **115**, 8505-8510. https://doi.org/10.1073/pnas.1718942115

[14] Long, Z.C., Lu, Y.P., Ma, X.Z. and Dong, B. (2018) PDE-net: Learning PDEs from Data. *Proceedings of the* 35*th International Conference on Machine Learning*, **80**, 3208-3216.

[15] Raissi, M., Perdikaris, P. and Karniadakis, G.E. (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**, 686-707. https://doi.org/10.1016/j.jcp.2018.10.045

[16] Yang, L., Meng, X.H. and Karniadakis, G.E. (2021) B-PINNs: Bayesian Physics-Informed Neural Networks for Forward and Inverse PDE Problems with Noisy Data. *Journal of Computational Physics*, **425**, Article ID: 109913. https://doi.org/10.1016/j.jcp.2020.109913

[17] Xun, X.L., Cao, J.G., Mallick, B., Maity, A. and Carroll, R.J. (2013) Parameter Estimation of Partial Differential Equation Models. *Journal of the American Statistical*

Association, **108**, 1009-1020. https://doi.org/10.1080/01621459.2013.794730

[18] Ruan, Y.D., Nadim, A., Duvvoori, L. and Chugunova, M. (2021) Liquid Films Falling down a Vertical Fiber: Modeling, Simulations and Experiments. *Fluids*, **6**, Article No. 281. https://doi.org/10.3390/fluids6080281

[19] Rudy, S.H., Brunton, S.L., Proctor, J.L. and Kutz, J.N. (2017) Data-Driven Discovery of Partial Differential Equations. *Science Advances*, **3**, e1602614. https://doi.org/10.1126/sciadv.1602614

Appendix

A. Convexity analysis of $\mathrm{Var}\left(\left[q'-u(1-u)(1-\gamma u)\right]/u'\right)$

First, transform this into $\mathrm{Var}\left(\left(q'+u^2-u\right)/u'-\left(u^3-u^2\right)/u'\gamma\right)$; then to simplify notation, define $\boldsymbol{v}_1=\left(q'+u^2-u\right)/u'$, $\boldsymbol{v}_2=\left(u^3-u^2\right)/u'$, and $\boldsymbol{c}=\boldsymbol{v}_1-\boldsymbol{v}_2\gamma$. So, during the sample variance $S^2(\boldsymbol{c})$ and sample mean $\bar{c}$ estimation, we have:

$$\text{Sample variance: } S^2(\boldsymbol{c})=\frac{1}{n-1}\sum_{i=1}^{n}\left(c_i-\bar{c}\right)^2=\frac{1}{n-1}\left\|\boldsymbol{c}-\bar{c}\mathbf{1}\right\|^2,$$

$$\text{Sample mean: } \bar{c}=\frac{1}{n}\sum_{i=1}^{n}c_i=\frac{1}{n}\mathbf{1}^{\mathrm{T}}\boldsymbol{c}=\frac{1}{n}\boldsymbol{c}^{\mathrm{T}}\mathbf{1}.$$

where $\|\cdot\|$ is the 2-norm, and $\mathbf{1}$ is the $n\times 1$ column vector all whose elements are 1. With that:

$$\begin{aligned}S^2(\boldsymbol{c})&=\frac{1}{n-1}\left\|\boldsymbol{c}-\frac{1}{n}\boldsymbol{c}^{\mathrm{T}}\mathbf{1}\mathbf{1}\right\|^2\\&=\frac{1}{n-1}\boldsymbol{c}^{\mathrm{T}}\left(\boldsymbol{I}-\frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)^{\mathrm{T}}\left(\boldsymbol{I}-\frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)\boldsymbol{c}.\end{aligned}$$

Next define $D=\left(\boldsymbol{I}-\frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)^{\mathrm{T}}\left(\boldsymbol{I}-\frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)$. It is clear that D is a symmetric matrix. Then, we have:

$$\begin{aligned}S^2(\boldsymbol{v}_1-\boldsymbol{v}_2\gamma)&=\frac{1}{n-1}\boldsymbol{c}^{\mathrm{T}}D\boldsymbol{c}\\&=\frac{1}{n-1}\left[\boldsymbol{v}_1^{\mathrm{T}}D\boldsymbol{v}_1-2\gamma\boldsymbol{v}_2^{\mathrm{T}}D\boldsymbol{v}_1+\gamma^2\boldsymbol{v}_2^{\mathrm{T}}D\boldsymbol{v}_2\right].\end{aligned}$$

To simplify the notation, define constants $c_1=\boldsymbol{v}_2^{\mathrm{T}}D\boldsymbol{v}_2$, $c_2=\boldsymbol{v}_1^{\mathrm{T}}D\boldsymbol{v}_2$, $c_3=\boldsymbol{v}_1^{\mathrm{T}}D\boldsymbol{v}_1$, so the function being minimized can be written as $g(\gamma)=\left[\gamma^2c_1-2\gamma c_2+c_3\right]/(n-1)$, and since $c_1>0$, by the second-order derivative condition for convexity $g(\gamma)$ is a convex function as $0\le\gamma<1$. Next, we would like to check whether $g(\gamma)$ is strongly convex. To do that, we define $x,y\in\{0\le\gamma<1\}$, then:

$$g(y)-g(x)=\frac{1}{n-1}\left[c_1\left(y^2-x^2\right)-2c_2(y-x)\right],$$

$$\begin{aligned}\nabla g(x)(y-x)&=\frac{1}{n-1}(2xc_1-2c_2)(y-x)\\&=\frac{2}{n-1}\left(xyc_1-yc_2-x^2c_1+xc_2\right).\end{aligned}$$

So that:

$$\begin{aligned}&g(y)-g(x)-\nabla g(x)(y-x)\\&=\frac{1}{n-1}\left[c_1\left(y^2-x^2\right)-2c_2(y-x)-2c_1xy+2c_2y+2c_1x^2-2c_2x\right]\\&=\frac{1}{n-1}c_1(y-x)^2.\end{aligned}$$

Therefore, there exists a positive constant m such that:

$$g(y)-g(x)-\nabla g(x)(y-x)=\frac{c_1}{n-1}(y-x)^2\geq\frac{m}{2}(y-x)^2.$$

Then, this function is strongly convex, and there always exists a unique solution for the following optimization problem:

$$\text{minimize}: g(\gamma)=\frac{1}{n-1}\left[\gamma^2 c_1-2\gamma c_2+c_3\right]$$
$$\text{subject to}: c_1=\boldsymbol{v}_2^{\mathrm{T}}D\boldsymbol{v}_2,\quad c_2=\boldsymbol{v}_1^{\mathrm{T}}D\boldsymbol{v}_2,\quad c_3=\boldsymbol{v}_1^{\mathrm{T}}D\boldsymbol{v}_1.$$

B. Convexity analysis of $\mathrm{Var}\left(a/(b+c\varepsilon)+d\right)$ when V is fixed

First, define vector $\boldsymbol{H}(\varepsilon)=\left[H_1,\cdots,H_n\right]^{\mathrm{T}}$ where $H_i=a_i/(b_i+c_i\varepsilon)$, and $\boldsymbol{k}=\boldsymbol{H}+\boldsymbol{d}$. From the previous proof, we have $S^2(\varepsilon)=\boldsymbol{k}^{\mathrm{T}}D\boldsymbol{k}/(n-1)$ and D is symmetric, therefore, for this case, we have the following results:

$$S^2(\varepsilon)=\frac{1}{n-1}\boldsymbol{k}^{\mathrm{T}}D\boldsymbol{k}=\frac{1}{n-1}(\boldsymbol{H}+\boldsymbol{d})^{\mathrm{T}}D(\boldsymbol{H}+\boldsymbol{d})$$
$$=\frac{1}{n-1}\left(\boldsymbol{H}^{\mathrm{T}}D\boldsymbol{H}+2\boldsymbol{d}^{\mathrm{T}}D\boldsymbol{H}+\boldsymbol{d}^{\mathrm{T}}D\boldsymbol{d}\right).$$

Since V is fixed, $\boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}$ would be a fixed constant and $\boldsymbol{d}^{\mathrm{T}}D\boldsymbol{d}$ is non-negative. Therefore to minimize $S^2(\varepsilon)$, instead of solving the original function directly, we consider this minimize function $f(\varepsilon)=\boldsymbol{H}^{\mathrm{T}}D\boldsymbol{H}+2\boldsymbol{d}^{\mathrm{T}}D\boldsymbol{H}$. Next, from our numerical solutions, $\left|c\varepsilon/b\right|<\left|c/(bV^2)\right|<1$ is always true. Therefore we can apply geometric series in our equation, so that $\boldsymbol{H}(\varepsilon)$ can be written as:

$$\boldsymbol{H}(\varepsilon)=\begin{bmatrix}\dfrac{a_1}{b_1}\dfrac{1}{1+\dfrac{c_1}{b_1}\varepsilon}\\ \vdots\\ \dfrac{a_n}{b_n}\dfrac{1}{1+\dfrac{c_n}{b_n}\varepsilon}\end{bmatrix}=\begin{bmatrix}\dfrac{a_1}{b_1}\displaystyle\sum_{m=1}^{\infty}\left(-\frac{c_1}{b_1}\right)^m\varepsilon^m\\ \vdots\\ \dfrac{a_n}{b_n}\displaystyle\sum_{m=1}^{\infty}\left(-\frac{c_n}{b_n}\right)^m\varepsilon^m\end{bmatrix}=\sum_{m=1}^{\infty}\begin{bmatrix}\dfrac{a_1}{b_1}\left(-\dfrac{c_1}{b_1}\right)^m\\ \vdots\\ \dfrac{a_n}{b_n}\left(-\dfrac{c_n}{b_n}\right)^m\end{bmatrix}\varepsilon^m=\sum_{m=1}^{\infty}\boldsymbol{h}_m\varepsilon^m.$$

Therefore, we have the new minimization function $f(\varepsilon)$:

$$f(\varepsilon)=\boldsymbol{H}^{\mathrm{T}}D\boldsymbol{H}+2\boldsymbol{d}^{\mathrm{T}}D\boldsymbol{H}$$
$$=\left(\sum_{m=1}^{\infty}\boldsymbol{h}_m^{\mathrm{T}}\varepsilon^m\right)D\left(\sum_{m=1}^{\infty}\boldsymbol{h}_m\varepsilon^m\right)+2\boldsymbol{d}^{\mathrm{T}}D\left(\sum_{m=1}^{\infty}\boldsymbol{h}_m\varepsilon^m\right)$$
$$=\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\boldsymbol{h}_m^{\mathrm{T}}D\boldsymbol{h}_n\varepsilon^{m+n}+2\sum_{m=1}^{\infty}\boldsymbol{d}D\boldsymbol{h}_m\varepsilon^m$$
$$=\boldsymbol{h}_1^{\mathrm{T}}D\boldsymbol{h}_1\varepsilon^2+2\boldsymbol{d}D\boldsymbol{h}_1\varepsilon+O\left(\varepsilon^3\right).$$

By the second-order derivative condition for convexity $f(\varepsilon)$ is a convex function. Next, let's prove that $f(\varepsilon)$ is strongly convex. Like the previous proof, we define the same two variables $x,y\in\left\{0\leq\varepsilon<1/V^2\right\}$, and there exists a positive constant m such that:

$$f(y)-f(x)-\nabla f(x)(y-x)=\boldsymbol{h}_1^{\mathrm{T}}D\boldsymbol{h}_1(y-x)^2\geq\frac{m}{2}(y-x)^2.$$

Therefore, function $f(\varepsilon)$ is strongly convex, and instead of the original optimization problem, we can solve the following problem which can get the same

results:

$$\text{minimize}: f(\varepsilon) = \boldsymbol{h}_1^{\mathrm{T}} D \boldsymbol{h}_1 \varepsilon^2 + 2\boldsymbol{d} D \boldsymbol{h}_1 \varepsilon$$

$$\text{subject to}: 0 \le \varepsilon < \frac{1}{V^2}.$$