Implementation Report: Fractional SEIRD PINN

Antonio Jimenez Ashton Cole

October 27, 2025

This report outlines our implementation and evaluation of a physics-informed neural network (PINN) designed for a fractional-order Susceptible-Exposed-Infected-Recovered-Deceased (SEIRD) epidemic model, based on the framework presented by Zinihi [1]. The referenced paper details the mathematical foundations of fractional epidemic modeling and demonstrates the capability of PINNs to estimate epidemiological parameters, including the fractional order α. As the paper does not provide their implementaion or network hyperparameters, this report focuses on our interpretation, implementation choices, and results obtained from simulating experiments analogous to those in the paper.

1 Fractional SEIRD Model

The SEIRD model is a system of nonlinear first-order ODEs which divides the population into categories of susceptible (S), exposed (E), infected (I), recovered (R), and dead (D), all as functions of time. The original model uses first derivatives, but in this paper they define a similar system using instead fractional derivatives of order $\alpha \in (0, 1]$ to account for so-called memory effects. We see that when $\alpha = 1$, the caputo derivative reduces to the standard first derivative. The population-normalized equations are presented here [1, p. 6]

$$\begin{aligned}
{}^{C}\mathcal{D}_t^{\alpha} s(t) &= -\beta \frac{s(t)i(t)}{1-d(t)} \\
{}^{C}\mathcal{D}_t^{\alpha} e(t) &= \beta \frac{s(t)i(t)}{1-d(t)} - \sigma e(t) \\
{}^{C}\mathcal{D}_t^{\alpha} i(t) &= \sigma e(t) - (\gamma + \mu) i(t) \\
{}^{C}\mathcal{D}_t^{\alpha} r(t) &= \gamma i(t) \\
{}^{C}\mathcal{D}_t^{\alpha} d(t) &= \mu i(t)
\end{aligned}$$

2 PINN Structure

The implemented PINN is a multilayer perceptron (MLP) neural network. It takes time $t \in [0, T]$ as input and outputs approximations of the normalized state variables $(\hat{s}, \hat{e}, \hat{\imath}, \hat{r}, \hat{d})$ [1, p. 6-7]:

$$\mathcal{NN}^{\theta,\Theta} : t \mapsto (\hat{s}(t;\theta), \hat{e}(t;\theta), \hat{\imath}(t;\theta), \hat{r}(t;\theta), \hat{d}(t;\theta)) \tag{1}$$

where θ represents the network's trainable weights and biases, and $\Theta = \{\beta, \sigma, \gamma, \mu, \alpha\}$ are the epidemiological parameters. The network is trained by minimizing a composite loss function $\mathcal{L}(\theta, \Theta)$, which balances multiple objectives [1, p. 8-9]:

$$\mathcal{L}(\theta, \Theta) = \lambda_{\text{data}}\mathcal{L}_{\text{data}} + \lambda_{\text{phys}}\mathcal{L}_{\text{phys}} + \lambda_{\text{IC}}\mathcal{L}_{\text{IC}} + \lambda_{\text{cons}}\mathcal{L}_{\text{cons}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}$$

The loss function is composed of the data loss, physics loss, initial condition loss, conservation loss, and regularization loss. Data loss penalizes the discrepancy between observed and predicted $\hat{s}, \hat{e}, \hat{\imath}, \hat{r}, \hat{d}$ variables. Physics loss enforces the fractional SEIRD dynamics at the collocation points, where the collocation points share the same domain as the synthethic data, but are sampled at a larger frequency ensuring that the dynamics are enforced throughout the entire domain. The initial condition loss enforces the s,e,i,r,d variables match the initial values. Conservation loss enforces population balance. Regularization loss penalizes network weights in order to avoid overfitting.

3 PINN Network Architecture

The MLP developed for this work consists of 3 hidden layers, each with 64 neurons, utilizing the hyperbolic tangent (tanh) activation function. Through preliminary experimentation, this architecture demonstrated sufficient capacity to approximate the complexities of the SEIRD functions.

The paper suggests maintaining non-negativity and approximate population conservation via the output layer, mentioning both 'softplus' and 'softmax' activation functions as possibilities [1, p. 7]. We initially opted to utilize a 'softmax' activation function in the final layer. This simplifies the loss function implementation as 'softmax' naturally transforms the output vector into positive components that sum to one, thereby explicitly enforcing both non-negativity and population conservation. We also explored using a 'softplus' activation (ensuring only positivity) combined with an explicit $\mathcal{L}_{\text{cons}}$ term, but did not observe a significant improvement in parameter estimation performance in our tests that would outweigh the additional model complexity.

Following Algorithm 1 in Zinihi [1, p. 10], the network weights were initialized using Xavier initialization, we opted for Xavier Uniform method provided by PyTorch. We followed the optimizer strategy outlined in Algorithm 1 involves a hybrid approach, starting with Adam for initial exploration followed by L-BFGS for fine-tuning. Lastly, in order to restrict the model learning alpha within a realistic range the model learns the parameter z_alpha which is then passed through the following equation where $min\_alpha = 0.9$ [1, p. 8]:

$$\alpha = min\_alpha + (1.0 - min\_alpha) * sigmoid(z\_alpha) \tag{2}$$

4 Model Training

The training procedure follows the two-stage optimization strategy described in Algorithm 1 of the paper.

1. **Stage 1 (Pre-training):** The fractional order α is fixed near 1 and only the network weights θ are trained by minimizing a reduced loss function, often $\mathcal{L}_{\text{data}} + \mathcal{L}_{\text{IC}}$
2. **Stage 2 (Joint Training):** The constraint on α is released, allowing it to become a trainable parameter along with $\beta, \sigma, \gamma, \mu$. Both the network weights θ and the epidemiological parameters Θ are updated simultaneously by minimizing the full composite loss function, which includes $\mathcal{L}_{\text{phys}}$.

In addition to this two-stage process, we implemented an early stopping mechanism based on the total loss value and a patience counter to help prevent overfitting, which was not explicitly mentioned in the paper's Algorithm 1.

5 Experimentation and Results

We began by exploring our fractional derivative implementation to visualize the impact of varying the fractional order α, as discussed in the paper [1]. We generated solutions to the normalized SEIRD system using our '*caputo_euler*' solver for $\alpha \in \{1, 0.95, 0.9\}$, keeping other parameters constant (based on Mpox values: $\beta = 0.25, \sigma = 0.13, \gamma = 0.052, \mu = 0.005$ [1]).

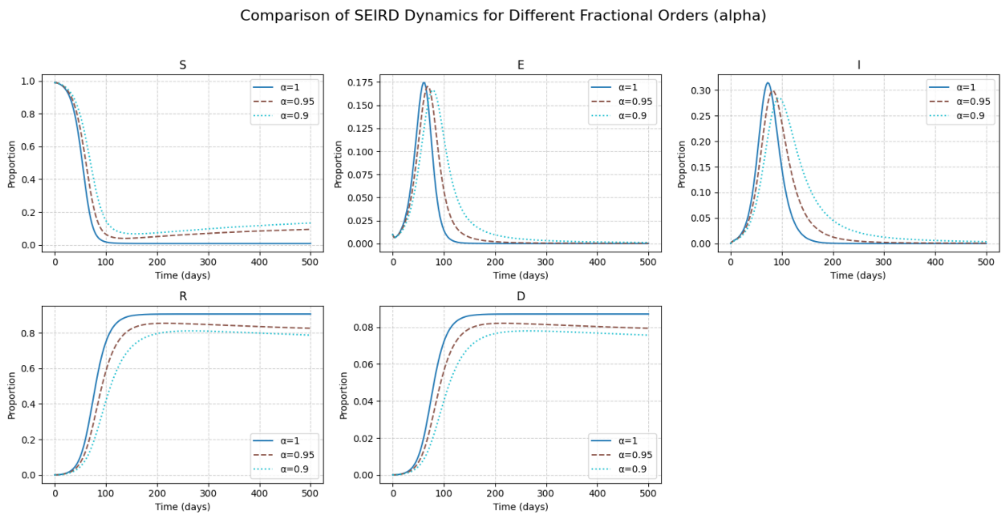

Figure 1: Comparison of SEIRD dynamics generated using the Caputo Euler solver for fractional orders $\alpha = 1$ (solid), $\alpha = 0.95$ (dashed), and $\alpha = 0.9$ (dotted). Shows the delay and flattening effect for $\alpha < 1$.

Figure 1 shows similar patterns to those observed in Figure 2 of Zinihi [1]. As α decreases from 1, the peak of the infected curve (i) is delayed and flattened, and the recovered curve (r) converges to a slightly lower value, illustrating the "memory effect" introduced by the fractional derivative [1, p. 11-12].

Similarly to the paper, we used synthetic data generated via the '*caputo_euler*' solver (with known ground-truth parameters) to verify our PINN implementation [1, p. 11-13]. The true parameters for data generation were set to mimic the Mpox case in the Expirement 1 section: $\alpha_{\text{true}} = 0.95, \beta_{\text{true}} = 0.25, \sigma_{\text{true}} = 0.13, \gamma_{\text{true}} = 0.052, \mu_{\text{true}} = 0.005$ [1, p. 11]. Since we could not locate the exact synthetic data used in the paper, we added Gaussian noise in Experiment 2 to simulate real world data. Additionally, the ground truth epidemiological parameters were randomly selected from a uniform distribution for a specified domain in Experiment 2.

5.1 Experiment 1: Parameter Estimation with Exact Data

This experiment uses the synthetic data generated with $\alpha_{\text{true}} = 0.95$ and no added noise. Stage 1 used Adam with a learning rate of 1e-3 and weight decay of 1e-5 for 5000 epochs, minimizing only data and IC loss with equal weightt. Stage 2 used Adam with lr=1e-3 and weight decay=1e-4 for 30000 epochs where the physics loss and initial condition loss weights are set to 1e-6.

Figures 2 and 3 display the PINN's approximation of the SEIRD states after each training stage, compared to the ground-truth data.

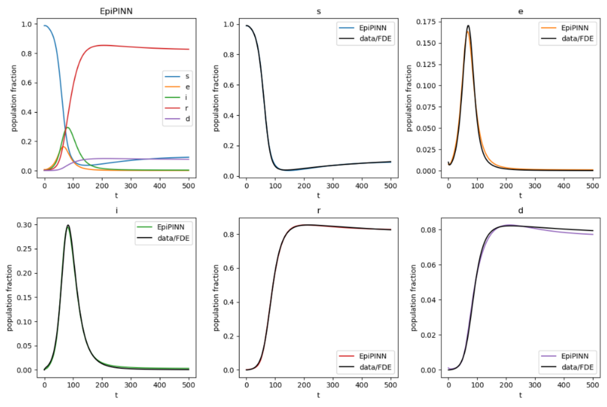

Figure 2: Experiment 1 (No Noise): PINN output vs. True Data after Stage 1 Training.

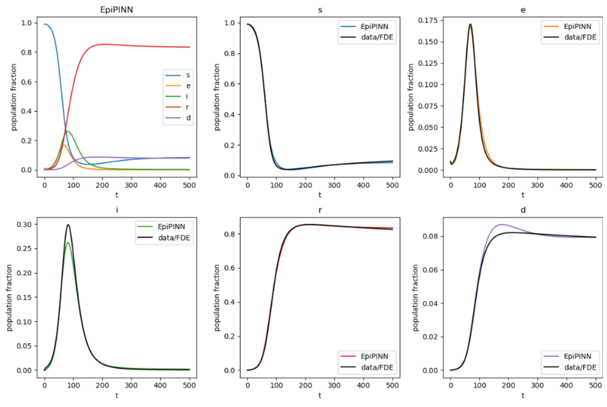

Figure 3: Experiment 1 (No Noise): PINN output vs. True Data after Stage 2 Training.

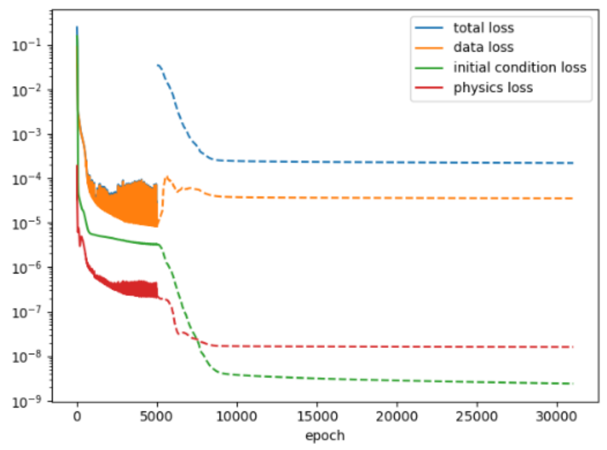

Figure 4: Experiment 1 (No Noise): Loss components during training (Stages 1 & 2).

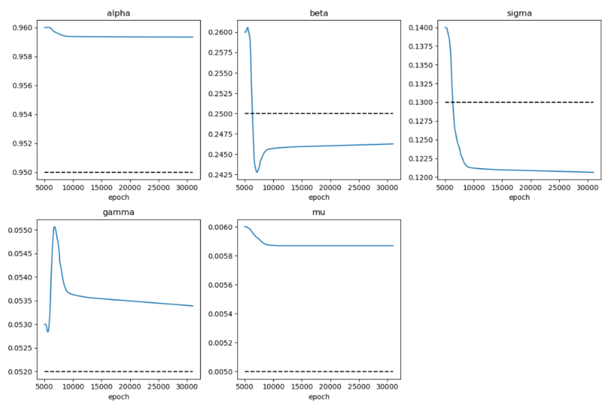

Figure 5: Experiment 1 (No Noise): Convergence of epidemiological parameters during Stage 2 Training. Dashed lines indicate true values ($\alpha = 0.95, \beta = 0.25, \sigma = 0.13, \gamma = 0.052, \mu = 0.005$).

As seen in Figure 2, after Stage 1, the PINN fits the training data very well, as expected since it prioritizes data and IC loss. Figure 3 shows that during Stage 2, where the physics loss is introduced and parameters are learned, the fit to the data slightly degrades as the model attempts to satisfy the fractional ODEs. The loss plot (Figure 4) confirms the decrease in physics and IC loss during Stage 2. However, Figure 5 reveals that while the parameters appear to converge, they do not converge to the correct ground-truth values ($\alpha = 0.95, \beta = 0.25, \sigma = 0.13, \gamma = 0.052, \mu = 0.005$).

5.2 Experiment 2: Parameter Estimation with Noisy Data

Experiment 2 uses the same setup as Experiment 1, but adds Gaussian noise with a relative standard deviation of 20% to the synthetic training data.

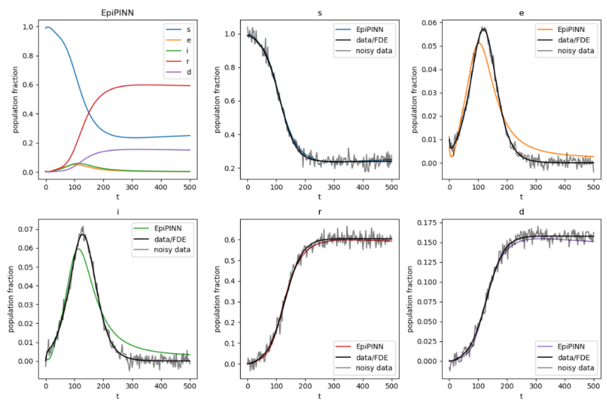
Figure 6: Experiment 2 (20% Noise): PINN output vs. Noisy Data after Stage 1 Training.

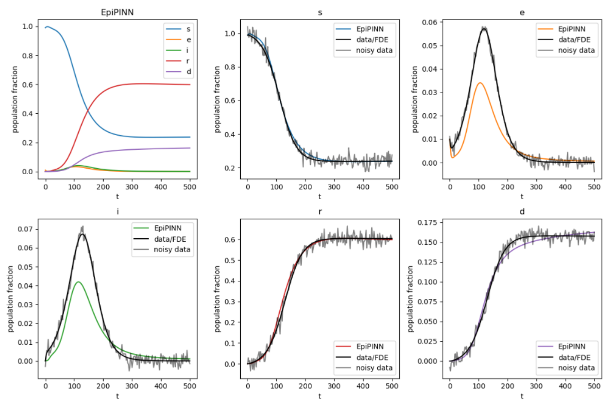
Figure 7: Experiment 2 (20% Noise): PINN output vs. Noisy Data after Stage 2 Training.

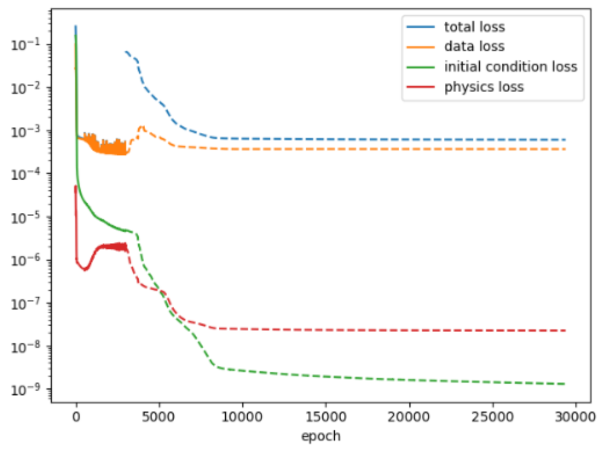

Figure 8: Experiment 2 (20% Noise): Loss components during training (Stages 1 & 2).

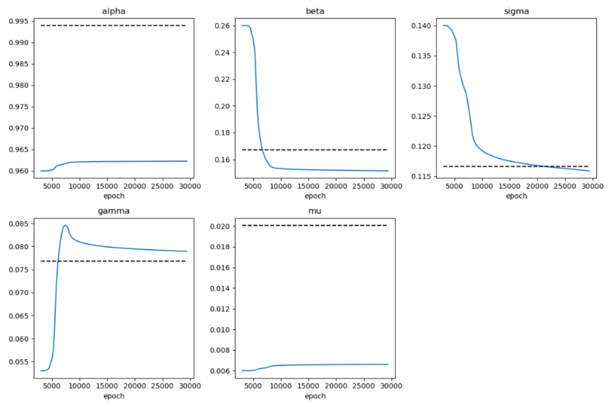

Figure 9: Experiment 2 (20% Noise): Convergence of epidemiological parameters during Stage 2 Training. Dashed lines indicate true values.

With noisy data, the PINN still captures the overall shape of the functions (Figures 6, 7), but exhibits a larger discrepancy compared to the noiseless case, particularly after Stage 2. Similar to Experiment 1, the physics and IC losses decrease during Stage 2 (Figure 8), indicating the model is attempting to adhere to the governing equations. However, the parameter convergence (Figure 9) remains problematic, with final values deviating significantly from the true parameters. Some parameters, like β and σ which appear more prominently in the equations involving larger populations, show slightly better convergence trends than others like μ or α.

The difficulty in learning α might stem from its indirect influence on the loss function, primarily through the Caputo derivative term in $\mathcal{L}_{\text{phys}}$. Normalizing the individual residual components within $\mathcal{L}_{\text{phys}}$ based on the scale of each variable might improve balance, especially since parameters like μ typically have much smaller values than β or σ. Furthermore, the paper's results section, while showing successful trajectory reconstruction (Figure 2) [1, p. 12], presents parameter estimates in tables (Table 2) [1, p. 13] without accompanying convergence plots. The parameter estimates shown in the paper for fixed α values also exhibit noticeable deviation from the ground truth (e.g., learned $\beta \approx 0.14$ vs true $\beta = 0.25$ for $\alpha = 0.95$ case [1]), making it difficult to fully assess the success of their implementation compared to ours based solely on the provided figures.

6 Conclusion

We successfully implemented a PINN framework based on the structure described by Zinihi [1] for a fractional-order SEIRD model. Our implementation included the MLP architecture, the two-stage training process, Xavier initialization, and the composite loss function incorporating the L1 scheme for the Caputo derivative. We visually confirmed our fractional derivative solver reproduces the expected delay effects for $\alpha < 1$.

Through synthetic data experiments (both clean and noisy), we demonstrated the PINN's ability to learn the overall dynamics and fit the provided data reasonably well. However, our implementation struggled to accurately converge to the true underlying epidemiological parameters even with clean data. The lack of detailed convergence plots or explicit code in the reference paper makes a direct comparison of parameter estimation performance challenging. Future work could involve better tuning of hyper parameters and balancing of epidemiological parameters residuals.

References

[1] A. Zinihi, *Identifying Memory Effects in Epidemics via a Fractional SEIRD Model and Physics-Informed Neural Networks*, arXiv:2509.22760v1 [stat.ML], 2025.