HW2 Report

Kangqi Yu 121090735

October 24, 2023

1 Introduction

I will explore several different specifications of infectious disease models (SIS, SIR, SEIR and SIRS) on an infective cases dataset. The main result is the SEIR model fits the data best.

2 Data

The data we use in this report with title of *The data of infected cases*, from CDC. The following is the scatter plot of this data: From the plot we can see it has a peek and the number of cases go to 0 at the end.

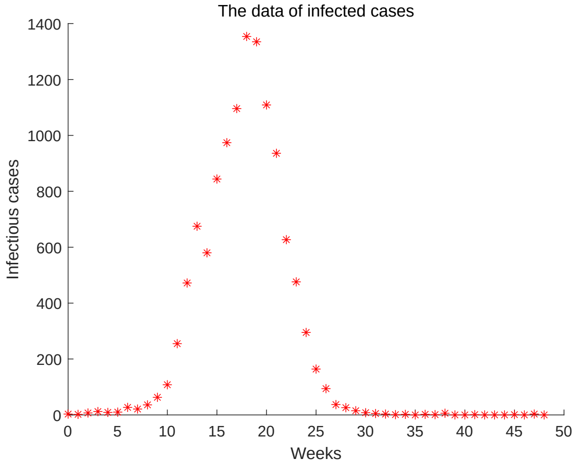

Figure 1: Plot of data

3 Model

3.1 SIS model

3.1.1 Parameters description

- Infective $I(t)$: the number of infective people in the population.
- Susceptible $S(t)$: healthy people but can become infective by contact with an infective.
- Infection rate β: on average, the probability that a random infective person infects a random susceptible persons in time Δt is $\beta \Delta t$.
- Recovery rate γ: the probability that an infective recovers during time Δt is $\gamma \Delta t$.

3.1.2 Equilibrium points and stability

The SIS model is given by

$$\begin{cases} \frac{dS}{dt} = -\beta SI + \gamma I \\ \frac{dI}{dt} = \beta SI - \gamma I \\ N = S + I \end{cases} \tag{1}$$

To find the equilibrium points, we set $\frac{dS}{dt} = \frac{dI}{dt} = 0$, then we have $I = 0, S = N$; $I = N - \frac{\gamma}{\beta}, S = \frac{\gamma}{\beta}$. The Jacobian matrix is given by

$$J = \begin{bmatrix} -\beta I & -\beta S + \gamma \\ \beta I & \beta S - \gamma \end{bmatrix} \tag{2}$$

It has two eigenvalues are 0 and $-\gamma + \beta S - \beta I$. For $I = 0, S = N$, the eigenvalues are 0 and $\beta N - \gamma$. When $N < \gamma/\beta$, the ODE system is stable; when $N > \gamma/\beta$, the ODE system is unstable. For $I = N - \frac{\gamma}{\beta}, S = \frac{\gamma}{\beta}$, the eigenvalues are $\gamma - \beta N$ and 0. When $N < \gamma/\beta$, the ODE system is unstable; when $N > \gamma/\beta$, the ODE system is stable. When $t \to \infty$ and $N > \gamma/\beta$, the system will converge to $I = N - \frac{\gamma}{\beta}, S = \frac{\gamma}{\beta}$, which has an epidemic; when $t \to \infty$ and $N < \gamma/\beta$, the system will converge to $I = 0, S = N$, which has no epidemic.

3.1.3 Basic Reproductive Ratio

The basic reproductive ratio R_0 is the expected number of secondary infective produced by a single primary infective in a completely susceptible population. Let's assume that the the probability of an individual infected at time 0 is still infective at time t as $l(t)$. So we have:

$$l(t+\Delta t) = l(t)(1-\gamma\Delta t) \implies l'(t) = -\gamma l(t) \implies l(t) = e^{-\gamma t}$$

.

$$R_0 = \int_0^\infty \beta N l(t) dt = \frac{\beta N}{\gamma}$$

3.2 SIR model

3.2.1 Parameters description

- Infective $I(t)$: the number of infective people in the population.
- Susceptible $S(t)$: healthy people but can become infective by contact with an infective.
- Removed $R(t)$: people who are immuned or died.
- Infection rate β: on average, the probability that a random infective person infects a random susceptible persons in time Δt is $\beta\Delta t$.
- Recovery rate γ: the probability that an infective recovers during time Δt is $\gamma\Delta t$.

3.2.2 Equilibrium points and stability

The SIR model is given by

$$\begin{cases} \frac{dS}{dt} = -\beta SI \\ \frac{dI}{dt} = \beta SI - \gamma I \\ \frac{dR}{dt} = \gamma I \\ N = S + I + R \end{cases} \tag{3}$$

To find the equilibrium points, we set $\frac{dS}{dt} = \frac{dI}{dt} = \frac{dR}{dt} = 0$, then we have $I = 0; S + R = N$.

The Jacobian matrix is given by

$$J = \begin{bmatrix} -\beta I & -\beta S & 0 \\ \beta I & \beta S - \gamma & 0 \\ 0 & \gamma & 0 \end{bmatrix} \tag{4}$$

Once $I = 0$, the Jacobian matrix will be

$$J = \begin{bmatrix} 0 & -\beta S & 0 \\ 0 & \beta S - \gamma & 0 \\ 0 & \gamma & 0 \end{bmatrix} \tag{5}$$

Then the eigenvalues of this Jacobian matrix are 0 and $\beta S - \gamma$. At the initial stage, $S \approx N$, which means the eigenvalues are 0 and $\beta N - \gamma$. When $N < \gamma/\beta$, the ODE system is stable; when $N > \gamma/\beta$, the ODE system is unstable. When $t \to \infty$ and $N > \gamma/\beta$, the system is unstable at $I = 0, S = N$, which has an epidemic; when $t \to \infty$ and $N < \gamma/\beta$, the system will converge to $I = 0, S = N$, which has no epidemic.

3.2.3 Basic Reproductive Ratio

The basic reproductive ratio R_0 is the expected number of secondary infective produced by a single primary infective in a completely susceptible population. Let's assume that the the probability of an individual infected at time 0 is still infective at time t as $l(t)$. So we have:

$$l(t + \Delta t) = l(t)(1 - \gamma \Delta t) \implies l'(t) = -\gamma l(t) \implies l(t) = e^{-\gamma t}$$

.

$$R_0 = \int_0^\infty \beta N l(t) dt = \frac{\beta N}{\gamma}$$

3.3 SEIR model

3.3.1 Parameters description

- Infective $I(t)$: the number of infective people in the population.
- Susceptible $S(t)$: healthy people but can become infective by contact with an infective.
- Removed $R(t)$: people who are immuned or died.
- Exposed $E(t)$: people who are exposed to the virus but not yet infective.
- Infection rate β: on average, the probability that a random infective person infects a random susceptible persons in time Δt is $\beta \Delta t$.
- Recovery rate γ: the probability that an infective recovers during time Δt is $\gamma \Delta t$.

- Incubation rate a: the probability that an exposed person becomes infective during time.
- transition rate μ: the probability that a person transfer from non-susceptible to susceptible.

3.3.2 Equilibrium points and stability

The SEIR model is given by

$$\begin{cases} \frac{dS}{dt} = \mu N - \mu S - \beta IS \\ \frac{dE}{dt} = \beta IS - (\mu + a)E \\ \frac{dI}{dt} = aE - (\gamma + \mu)I \\ \frac{dR}{dt} = \gamma I - \mu R \\ N = S + E + I + R \end{cases} \tag{6}$$

To find the equilibrium points, we set $\frac{dS}{dt} = \frac{dE}{dt} = \frac{dI}{dt} = \frac{dR}{dt} = 0$. We have a trivial solution, $I = 0; R = 0; E = 0; S = N$. And a non-trivial solution, $S = \frac{(\mu+\gamma)(\mu+a)}{a\beta}; R = \frac{\gamma}{\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1); E = \frac{\mu(\mu+\gamma)}{a\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1); I = \frac{\mu}{\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1)$

The Jacobian matrix is given by

$$J = \begin{bmatrix} -\beta I - \mu & 0 & -\beta S & 0 \\ \beta I & -(\mu + a) & \beta S & 0 \\ 0 & a & -(\gamma + \mu) & 0 \\ 0 & 0 & \gamma & -\mu \end{bmatrix} \tag{7}$$

Put $I = 0; R = 0; E = 0$ into the Jacobian matrix, we have

$$J = \begin{bmatrix} -\mu & 0 & -\beta N & 0 \\ 0 & -(\mu + a) & \beta N & 0 \\ 0 & a & -(\gamma + \mu) & 0 \\ 0 & 0 & \gamma & -\mu \end{bmatrix} \tag{8}$$

The eigenvalues of this Jacobian matrix are $-\mu$, $-\mu$ and two roots of $\lambda^2 + (\gamma + 2\mu + a)\lambda + (r + \mu)(\mu + a) - a\beta N = 0$. For this equation, we have already known $\gamma > 0, \mu > 0, a > 0$, the axis of symmetry intersects with the negative part of x axis. So, whether these two roots have negative real parts depending on the intersection of parabolic curve and the y axis. When $(\gamma + \mu)(\mu + a) - a\beta N \geq 0$, the real parts of two roots are negative (or only one root is 0) and the equilibrium is stable. When $(\gamma + \mu)(\mu + a) - a\beta N < 0$, at least one root have positive real part, the equilibrium is unstable.

If $S = \frac{(\mu+\gamma)(\mu+a)}{a\beta}; R = \frac{\gamma}{\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1); E = \frac{\mu(\mu+\gamma)}{a\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1); I = \frac{\mu}{\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1)$ exists, $R \geq 0$, $E \geq 0$ and $I \geq 0$ for sure. So $(\gamma + \mu)(\mu + a) - a\beta N > 0$. We can check that in this condition, the eigenvalues of Jacobian matrix all have negative real parts. So this equilibrium is stable.

When $t \to \infty$ and $(\gamma + \mu)(\mu + a) - a\beta N < 0$, the system will converge to $S = \frac{(\mu+\gamma)(\mu+a)}{a\beta}; R = \frac{\gamma}{\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1); E = \frac{\mu(\mu+\gamma)}{a\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1); I = \frac{\mu}{\beta}(\frac{a\beta N}{\mu^2+a\mu+\gamma\mu+a\gamma} - 1)$; when $t \to \infty$ and $(\gamma + \mu)(\mu + a) - a\beta N \geq 0$, the system will converge to $I = 0; R = 0; E = 0; S = N$.

3.3.3 Basic Reproductive Ratio

As $R_0 \geq 1$ means there is a epidemic, we can get $R_0 = \frac{a\beta N}{(\gamma+\mu)(\mu+a)}$

3.4 SIRS model

3.4.1 Parameters description

- Infective $I(t)$: the number of infective people in the population.
- Susceptible $S(t)$: healthy people but can become infective by contact with an infective.
- Removed $R(t)$: people who are immuned or died.
- Infection rate β: on average, the probability that a random infective person infects a random susceptible persons in time Δt is $\beta\Delta t$.
- Recovery rate γ: the probability that an infective recovers during time Δt is $\gamma\Delta t$.
- Immunity loss rate ξ: the probability that an immuned person loses immunity during time

3.4.2 Equilibrium points and stability

The SIRS model is given by

$$\begin{cases} \frac{dS}{dt} = -\beta SI + \xi R \\ \frac{dI}{dt} = \beta SI - \gamma I \\ \frac{dR}{dt} = \gamma I - \xi R \\ N = S + I + R \end{cases} \tag{9}$$

To find the equilibrium points of this system, I set $\frac{dS}{dt} = \frac{dI}{dt} = \frac{dR}{dt} = 0$. Then we have $S = N; I = 0; R = 0$ or $S = \gamma/\beta; I = \frac{N-\gamma/\beta}{1+\gamma/\xi}; R = \frac{N-\gamma/\beta}{\xi/\gamma+1}$.

The Jacobian matrix is given by

$$J = \begin{bmatrix} -\beta I & -\beta S & \xi \\ \beta I & \beta S - \gamma & 0 \\ 0 & \gamma & -\xi \end{bmatrix} \tag{10}$$

Let $det(J - \lambda I) = 0$, we have one eigenvalue is 0 and the other two are the roots of $\lambda^2 + (\beta I + \gamma + \xi - \beta S)\lambda + \gamma\beta I + r\xi + \beta I\xi - \xi\beta S = 0$

Given $S = N; I = 0; R = 0$, eigenvalues are 0, $-\xi$ and $\beta N - \gamma$. Generally, $\xi > 0$. When $N < \gamma/\beta$, the ODE system is stable. Given $S = \gamma/\beta; I = \frac{N-\gamma/\beta}{1+\gamma/\xi}; R = \frac{N-\gamma/\beta}{\xi/\gamma+1}$, this solution set implies $N > \lambda/\beta$. At this equilibrium point, we have 0 as an eigenvalue and two negative real part roots in $\lambda^2 + (\beta I + \xi)\lambda + \gamma\beta I + \beta I\xi = 0$ (for $\beta I + \xi > 0$ and $\gamma\beta I + \beta I\xi > 0$, the axis of symmetry intersects with the negative part of x axis and intersects with the y axis at a positive point, so the roots are negative). So this equilibrium point is stable. When $t \to \infty$ and $N < \gamma/\beta$, the system will converge to $S = N; I = 0; R = 0$; When $t \to \infty$, $N > \gamma/\beta$ and $(\beta I - \xi)^2 \geq 4\gamma\beta I$ (from $\Delta \geq 0$ of the equation), the system will converge to $S = \gamma/\beta; I = \frac{N-\gamma/\beta}{1+\gamma/\xi}; R = \frac{N-\gamma/\beta}{\xi/\gamma+1}$; When $t \to \infty$, $N > \gamma/\beta$ and $(\beta I - \xi)^2 < 4\gamma\beta I$ (from $\Delta < 0$ of the equation), the system is periodic.

3.4.3 Basic Reproductive Ratio

The basic reproductive ratio R_0 is the expected number of secondary infective produced by a single primary infective in a completely susceptible population. Let's assume that the the probability of an individual infected at time 0 is still infective at time t as $l(t)$. So we have:

$$l(t+\Delta t) = l(t)(1-\gamma\Delta t) \implies l'(t) = -\gamma l(t) \implies l(t) = e^{-\gamma t}$$

.

$$R_0 = \int_0^\infty \beta N l(t) dt = \frac{\beta N}{\gamma}$$

4 Empirical results

4.1 Method

To estimate parameters in the models, I use *lsqcurvefit* function in MATLAB. The function is used to solve nonlinear least-squares curve-fitting problems.

4.2 Result

4.2.1 Fit the data to SIS model

The code to fit the data is in the Appendix. The result is shown in the following figure:

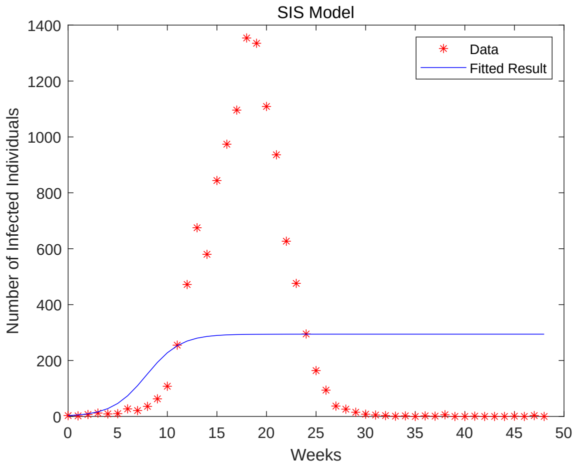

Figure 2: SIS Model Fitted Result

The details of it are

- Fitted β: 0.001971
- Fitted γ: 310.435862

- R_0: 1.001870
- RMSE (Root Mean Squared Error): 380.283851
- MeAE (Mean Absolute Error): 291.220724
- MaAE (Maximum Absolute Error): 1060.444669

4.2.2 Fit the separate data to SIS model

I sparate the dataset to training data and test data according to the odevity of the week number (the even part is the training dataset and the odd part is the test dataset). The code to fit the data is in the Appendix. The result is shown in the following figure:

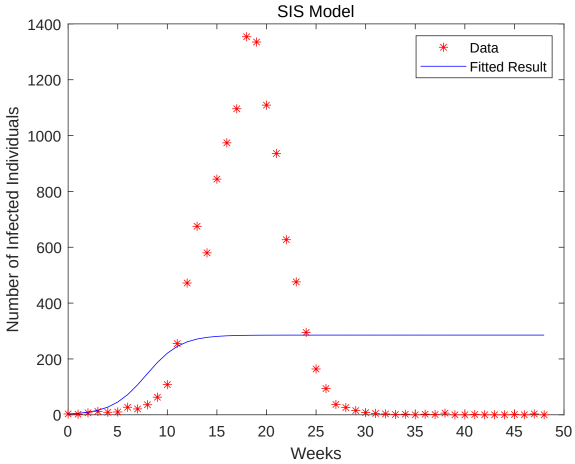

Figure 3: SIS Model Separate Data Fitted Result

The details of it are

- Fitted β: 0.002021
- Fitted γ: 318.242604
- R_0: 1.001812
- Training RMSE (Root Mean Squared Error): 373.478914
- Training MeAE (Mean Absolute Error): 280.936431
- Training MaAE (Maximum Absolute Error): 1069.422554

- Test RMSE (Root Mean Squared Error): 387.413773
- Test MeAE (Mean Absolute Error): 297.124192
- Test MaAE (Maximum Absolute Error): 1050.030656

4.2.3 Fit the data to SIR model

The code to fit the data is in the Appendix. The result is shown in the following figure:

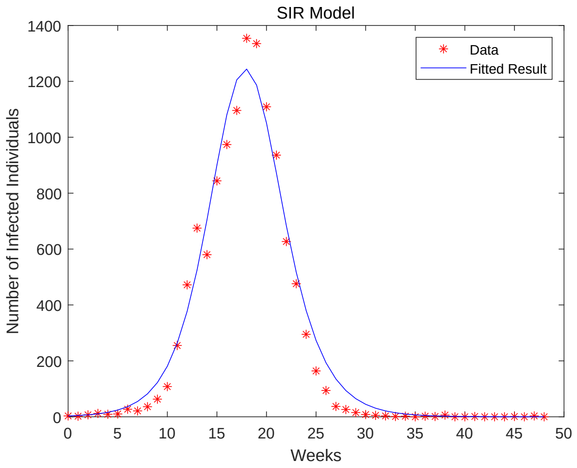

Figure 4: SIR Model Fitted Result

The details of it are

- Fitted β: 0.000022
- Fitted γ: 2.982421
- R_0: 1.139924
- RMSE (Root Mean Squared Error): 60.723076
- MeAE (Mean Absolute Error): 40.975050
- MaAE (Maximum Absolute Error): 149.888745

4.2.4 Fit the separate data to SIR model

I sparate the dataset to training data and test data according to the odevity of the week number (the even part is the training dataset and the odd part is the test dataset). The code to fit the data is in the Appendix. The result is shown in the following figure:

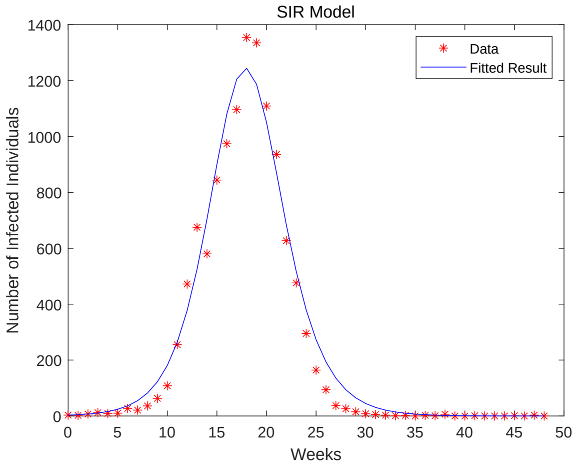

Figure 5: SIR Model Separate Data Fitted Result

The details of it are

- Fitted β: 0.000022
- Fitted γ: 2.982421
- R_0: 1.139924
- Training RMSE (Root Mean Squared Error): 58.468210
- Training MeAE (Mean Absolute Error): 40.331389
- Training MaAE (Maximum Absolute Error): 124.586564
- Test RMSE (Root Mean Squared Error): 62.986116
- Test MeAE (Mean Absolute Error): 41.645532
- Test MaAE (Maximum Absolute Error): 149.888731

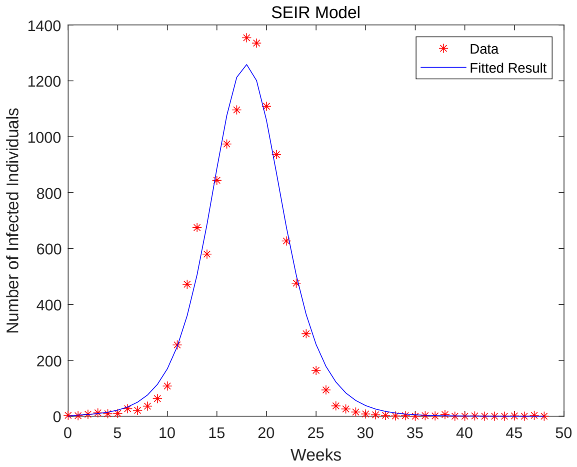

Figure 6: SEIR Model Fitted Result

4.2.5 Fit the data to SEIR model

The code to fit the data is in the Appendix. The result is shown in the following figure:

The details of it are

- Fitted β: 0.000101
- Fitted γ: 13.761815
- Fitted μ: 0.000000
- Fitted a: 3.379388
- R_0: 1.159444
- RMSE (Root Mean Squared Error): 57.019285
- MeAE (Mean Absolute Error): 37.225383
- MaAE (Maximum Absolute Error): 169.058115

4.2.6 Fit the separate data to SEIR model

I sparate the dataset to training data and test data according to the odevity of the week number (the even part is the training dataset and the odd part is the test dataset). The code to fit the data is in the Appendix. The result is shown in the following figure:

The details of it are

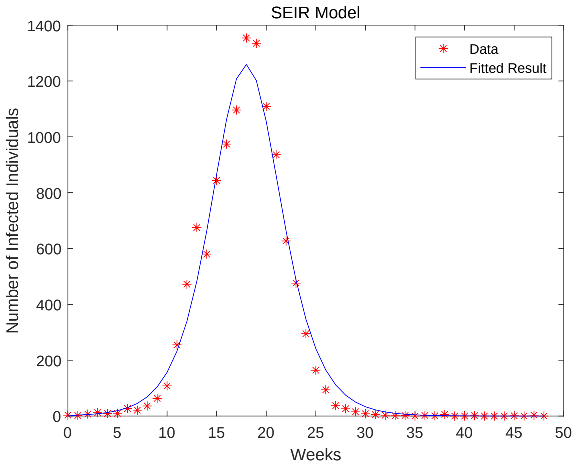

Figure 7: SEIR Model Separate Data Fitted Result

- Fitted β: 0.000061
- Fitted γ: 8.159985
- Fitted μ: 0.000083
- Fitted a: 3.725080
- R_0: 1.174294
- Training RMSE (Root Mean Squared Error): 48.956673
- Training MeAE (Mean Absolute Error): 32.021108
- Training MaAE (Maximum Absolute Error): 132.786756
- Test RMSE (Root Mean Squared Error): 61.041520
- Test MeAE (Mean Absolute Error): 36.294226
- Test MaAE (Maximum Absolute Error): 192.317321

4.2.7 Fit the data to SIRS model

The code to fit the data is in the Appendix. The result is shown in the following figure:

The details of it are

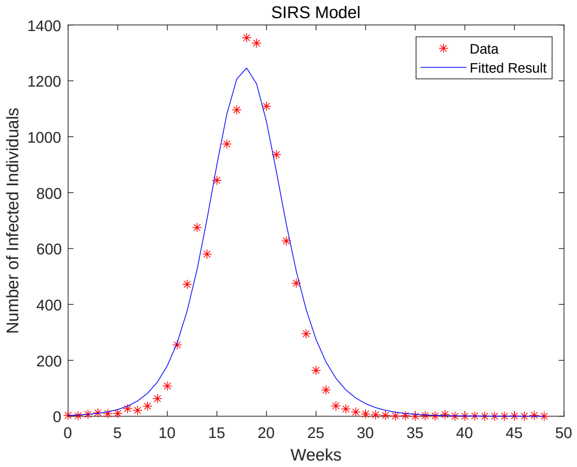

Figure 8: SIRS Model Fitted Result

- Fitted β: 0.000022
- Fitted γ: 2.979318
- Fitted ξ: 0.000000
- R_0: 1.140034
- RMSE (Root Mean Squared Error): 60.731237
- MeAE (Mean Absolute Error): 41.017104
- MaAE (Maximum Absolute Error): 150.259531

4.2.8 Fit the separate data to SIRS model

I sparate the dataset to training data and test data according to the odevity of the week number (the even part is the training dataset and the odd part is the test dataset). The code to fit the data is in the Appendix. The result is shown in the following figure:
The details of it are

- Fitted β: 0.000022
- Fitted γ: 2.990896
- Fitted ξ: 0.000001

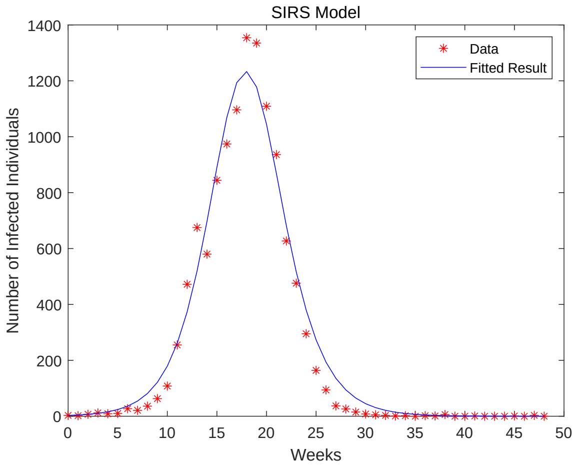

Figure 9: SIRS Model Separate Data Fitted Result

- R_0: 1.139235
- Training RMSE (Root Mean Squared Error): 58.284472
- Training MeAE (Mean Absolute Error): 40.273312
- Training MaAE (Maximum Absolute Error): 120.868918
- Test RMSE (Root Mean Squared Error): 63.406632
- Test MeAE (Mean Absolute Error): 41.346049
- Test MaAE (Maximum Absolute Error): 156.902558

5 Discussions

From the previous results, we can see that the SEIR model fits the data best. The reason may be that the SEIR model has more parameters than other models (μ and a), which help it to capture the transition and exposition processes, so it can fit the data better.

6 Appendix

6.1 Data plot

```
weeks = 0:48;
cases = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 8
scatter(weeks, cases, 'r*');
title('The␣data␣of␣infected␣cases');
xlabel('Weeks');
ylabel('Infectious␣cases');
```

6.2 Code to fit the data to SIS model

```
weeks = 0:48;
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
N = 157759;

guess = [1, 1];
options=optimoptions('lsqcurvefit');
options = optimoptions(options,'StepTolerance',1e-10,'FunctionTolerance
upper_bound =[inf,inf];
lower_bound =[0,0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I), guess
beta_fitted = params_fitted(1)/N;
gamma_fitted = params_fitted(2);

I0 = I(1);
S0 = N - I0;
[t, Y] = ode45(@(t,y) SIS_model(t, y, beta_fitted*N, gamma_fitted, N),
figure;
plot(weeks, I, 'r*', t, Y(:,2), 'b-');
xlabel('Weeks');
ylabel('Number␣of␣Infected␣Individuals');
legend('Data', 'Fitted␣Result');
title('SIS␣Model');
```

```
fprintf('Fitted beta: %.6f\n', beta_fitted);
fprintf('Fitted gamma: %.6f\n', gamma_fitted);
R_0 = beta_fitted*N/gamma_fitted;
fprintf('R_0: %.6f\n', R_0);

rmse = sqrt(mean((I' - Y(:,2)).^2));
meae = mean(abs(I' - Y(:,2)));
maae = max(abs(I' - Y(:,2)));

fprintf('RMSE (Root Mean Squared Error): %.6f\n', rmse);
fprintf('MeAE (Mean Absolute Error): %.6f\n', meae);
fprintf('MaAE (Maximum Absolute Error): %.6f\n', maae);

function dy = SIS_model(t, y, beta, gamma, N)
S = y(1);
I = y(2);
dy = zeros(2,1);
dy(1) = -beta * S * I/N + gamma * I;
dy(2) = beta * S * I/N - gamma * I;
end

function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
N = 157759;
I0 = I(1);
S0 = N - I0;
[T, Y] = ode45(@(t,y) SIS_model(t, y, beta, gamma, N), t, [S0; I0]);
I_predicted = interp1(T, Y(:,2), t, 'linear', 'extrap');
end
```

6.3 Code to fit the data to SIS model separately

```
weeks = 0:48;
weeks_train = weeks(1:2:end);
weeks_test = weeks(2:2:end);
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
I_train = I(1:2:end);
I_test = I(2:2:end);
N = 157759;

guess = [0.01, 0.1];
options=optimoptions(’lsqcurvefit’);
options = optimoptions(options,’StepTolerance’,1e−10,’FunctionTolerance
upper_bound =[inf,inf];
lower_bound =[0,0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I_train),
beta_fitted = params_fitted(1)/N;
gamma_fitted = params_fitted(2);

I0 = I(1);
S0 = N − I0;
[t, Y] = ode45(@(t,y) SIS_model(t, y, beta_fitted∗N, gamma_fitted, N),
figure;
plot(weeks, I, ’r∗’, t, Y(:,2), ’b−’);
xlabel(’Weeks’);
ylabel(’Number␣of␣Infected␣Individuals’);
legend(’Data’, ’Fitted␣Result’);
title(’SIS␣Model’);

fprintf(’Fitted␣beta:␣%.6f\n’, beta_fitted);
fprintf(’Fitted␣gamma:␣%.6f\n’, gamma_fitted);
```

```
R_0 = beta_fitted*N/gamma_fitted;
fprintf('R_0: %.6f\n', R_0);

rmse_train = sqrt(mean((I_train' - Y(1:2:end,2)).^2));
meae_train = mean(abs(I_train' - Y(1:2:end,2)));
maae_train = max(abs(I_train' - Y(1:2:end,2)));

rmse_test = sqrt(mean((I_test' - Y(2:2:end,2)).^2));
meae_test = mean(abs(I_test' - Y(2:2:end,2)));
maae_test = max(abs(I_test' - Y(2:2:end,2)));

fprintf('Training RMSE (Root Mean Squared Error): %.6f\n', rmse_train);
fprintf('Training MeAE (Mean Absolute Error): %.6f\n', meae_train);
fprintf('Training MaAE (Maximum Absolute Error): %.6f\n', maae_train);
fprintf('Test RMSE (Root Mean Squared Error): %.6f\n', rmse_test);
fprintf('Test MeAE (Mean Absolute Error): %.6f\n', meae_test);
fprintf('Test MaAE (Maximum Absolute Error): %.6f\n', maae_test);

function dy = SIS_model(t, y, beta, gamma, N)
S = y(1);
I = y(2);
dy = zeros(2,1);
dy(1) = -beta/N * S * I + gamma * I;
dy(2) = beta/N * S * I - gamma * I;
end

function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
N = 157759;
I0 = I(1);
```

```
S0 = N - I0;
[T, Y] = ode45(@(t,y) SIS_model(t, y, beta, gamma, N), t, [S0; I0]);
I_predicted = interp1(T, Y(:,2), t, 'linear', 'extrap');
end
```

6.4 Code to fit the data to SIR model

```
weeks = 0:48;
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
N = 157759;

guess = [0.1, 0.1];
options=optimoptions('lsqcurvefit');
options = optimoptions(options,'StepTolerance',1e-10,'FunctionTolerance
upper_bound =[inf,inf];
lower_bound =[0,0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I), guess
beta_fitted = params_fitted(1)/N;
gamma_fitted = params_fitted(2);

I0 = I(1);
S0 = N - I0;
R0 = 0;
[t, Y] = ode45(@(t,y) SIR_model(t, y, beta_fitted*N, gamma_fitted, N),
figure;
plot(weeks, I, 'r*', t, Y(:,2), 'b-');
xlabel('Weeks');
ylabel('Number of Infected Individuals');
legend('Data', 'Fitted Result');
title('SIR Model');
```

```
fprintf('Fitted beta: %.6f\n', beta_fitted);
fprintf('Fitted gamma: %.6f\n', gamma_fitted);
R_0 = beta_fitted*N/gamma_fitted;
fprintf('R_0: %.6f\n', R_0);

rmse = sqrt(mean((I' - Y(:,2)).^2));
meae = mean(abs(I' - Y(:,2)));
maae = max(abs(I' - Y(:,2)));

fprintf('RMSE (Root Mean Squared Error): %.6f\n', rmse);
fprintf('MeAE (Mean Absolute Error): %.6f\n', meae);
fprintf('MaAE (Maximum Absolute Error): %.6f\n', maae);

function dy = SIR_model(t, y, beta, gamma, N)
S = y(1);
I = y(2);
R = y(3);
dy = zeros(3,1);
dy(1) = -beta/N*S*I;
dy(2) = beta/N*S*I - gamma*I;
dy(3) = gamma*I;
end
% Define the objective function for least squares fitting
function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
N = 157759;
I0 = I(1);
S0 = N - I0;
R0 = 0;
[T, Y] = ode45(@(t,y) SIR_model(t, y, beta, gamma, N), t, [S0; I0; R0])
```

```
I_predicted = interp1(T, Y(:,2), t, 'linear', 'extrap');
end
```

6.5 Code to fit the data to SIR model separately

```
weeks = 0:48;
weeks_train = weeks(1:2:end);
weeks_test = weeks(2:2:end);
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
I_train = I(1:2:end);
I_test = I(2:2:end);
N = 157759;

guess = [0.00001, 1];
options=optimoptions('lsqcurvefit');
options = optimoptions(options,'StepTolerance',1e−10,'FunctionTolerance
upper_bound =[inf,inf];
lower_bound =[0,0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I), guess
beta_fitted = params_fitted(1)/N;
gamma_fitted = params_fitted(2);

I0 = I(1);
S0 = N − I0;
R0 = 0;
[t, Y] = ode45(@(t,y) SIR_model(t, y, beta_fitted*N, gamma_fitted, N),
figure;
plot(weeks, I, 'r*', t, Y(:,2), 'b−');
xlabel('Weeks');
ylabel('Number of Infected Individuals');
legend('Data', 'Fitted Result');
```

```
title('SIR Model');

fprintf('Fitted beta: %.6f\n', beta_fitted);
fprintf('Fitted gamma: %.6f\n', gamma_fitted);
R_0 = beta_fitted*N/gamma_fitted;
fprintf('R_0: %.6f\n', R_0);

rmse_train = sqrt(mean((I_train' - Y(1:2:end,2)).^2));
meae_train = mean(abs(I_train' - Y(1:2:end,2)));
maae_train = max(abs(I_train' - Y(1:2:end,2)));

rmse_test = sqrt(mean((I_test' - Y(2:2:end,2)).^2));
meae_test = mean(abs(I_test' - Y(2:2:end,2)));
maae_test = max(abs(I_test' - Y(2:2:end,2)));

fprintf('Training RMSE (Root Mean Squared Error): %.6f\n', rmse_train);
fprintf('Training MeAE (Mean Absolute Error): %.6f\n', meae_train);
fprintf('Training MaAE (Maximum Absolute Error): %.6f\n', maae_train);
fprintf('Test RMSE (Root Mean Squared Error): %.6f\n', rmse_test);
fprintf('Test MeAE (Mean Absolute Error): %.6f\n', meae_test);
fprintf('Test MaAE (Maximum Absolute Error): %.6f\n', maae_test);

function dy = SIR_model(t, y, beta, gamma, N)
S = y(1);
I = y(2);
R = y(3);
dy = zeros(3,1);
dy(1) = -beta/N*S*I;
dy(2) = beta/N*S*I - gamma*I;
dy(3) = gamma*I;
end
```

```
% Define the objective function for least squares fitting
function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
N = 157759;
I0 = I(1);
S0 = N − I0;
R0 = 0;
[T, Y] = ode45(@(t,y) SIR_model(t, y, beta, gamma, N), t, [S0; I0; R0])
I_predicted = interp1(T, Y(:,2), t, ’linear’, ’extrap’);
end
```

6.6 Code to fit the data to SEIR model

```
weeks = 0:48;
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
N = 157759;

guess = [0.001, 0.1, 0.00001, 0.1];
options=optimoptions(’lsqcurvefit’);
options = optimoptions(options,’StepTolerance’,1e−10,’FunctionTolerance
upper_bound =[inf,inf, inf, inf];
lower_bound =[0, 0, 0, 0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I), guess
beta_fitted = params_fitted(1);
gamma_fitted = params_fitted(2);
mu_fitted = params_fitted(3);
a_fitted = params_fitted(4);

I0 = I(1);
E0 = 0;
```

```
S0 = N - I0;
R0 = 0;
[t, Y] = ode45(@(t,y) SEIR_model(t, y, beta_fitted, gamma_fitted, mu_f
figure;
plot(weeks, I, 'r*', t, Y(:,2), 'b-');
xlabel('Weeks');
ylabel('Number of Infected Individuals');
legend('Data', 'Fitted Result');
title('SEIR Model');

fprintf('Fitted beta: %.6f\n', beta_fitted);
fprintf('Fitted gamma: %.6f\n', gamma_fitted);
fprintf('Fitted mu: %.6f\n', mu_fitted);
fprintf('Fitted a: %.6f\n', a_fitted);
R_0 = a_fitted*beta_fitted*N/((mu_fitted + a_fitted)*(mu_fitted + gamm
fprintf('R_0: %.6f\n', R_0);

rmse = sqrt(mean((I' - Y(:,2)).^2));
meae = mean(abs(I' - Y(:,2)));
maae = max(abs(I' - Y(:,2)));

fprintf('RMSE (Root Mean Squared Error): %.6f\n', rmse);
fprintf('MeAE (Mean Absolute Error): %.6f\n', meae);
fprintf('MaAE (Maximum Absolute Error): %.6f\n', maae);

function dy = SEIR_model(t, y, beta, gamma, mu, a, N)
S = y(1);
E = y(2);
I = y(3);
R = y(4);
dy = zeros(4,1);
```

```
dy(1) = mu * N - mu * S - beta * S * I;
dy(2) = beta * S * I - (mu + a) * E;
dy(3) = a * E - (gamma + mu) * I;
dy(4) = gamma * I - mu * R;
end
% Define the objective function for least squares fitting
function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
mu = params(3);
a = params(4);
N = 157759;
I0 = I(1);
E0 = 0;
S0 = N - I0;
R0 = 0;
[T, Y] = ode45(@(t,y) SEIR_model(t, y, beta, gamma, mu, a, N), t, [S0;
I_predicted = interp1(T, Y(:,2), t, ’linear’, ’extrap’);
end
```

6.7 Code to fit the data to SEIR model separately

```
weeks = 0:48;
weeks_train = weeks(1:2:end);
weeks_test = weeks(2:2:end);
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
I_train = I(1:2:end);
I_test = I(2:2:end);
N = 157759;

guess = [0.0001, 1, 0.001, 0.1];
```

```
options=optimoptions(’lsqcurvefit’);
options = optimoptions(options,’StepTolerance’,1e-10,’FunctionTolerance
upper_bound =[inf,inf, inf, inf];
lower_bound =[0, 0, 0, 0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I), guess
beta_fitted = params_fitted(1);
gamma_fitted = params_fitted(2);
mu_fitted = params_fitted(3);
a_fitted = params_fitted(4);

I0 = I(1);
E0 = 0;
S0 = N - I0;
R0 = 0;
[t, Y] = ode45(@(t,y) SEIR_model(t, y, beta_fitted, gamma_fitted, mu_f
figure;
plot(weeks, I, ’r*’, t, Y(:,2), ’b-’);
xlabel(’Weeks’);
ylabel(’Number□of□Infected□Individuals’);
legend(’Data’, ’Fitted□Result’);
title(’SEIR□Model’);

fprintf(’Fitted□beta:□%.6f\n’, beta_fitted);
fprintf(’Fitted□gamma:□%.6f\n’, gamma_fitted);
fprintf(’Fitted□mu:□%.6f\n’, mu_fitted);
fprintf(’Fitted□a:□%.6f\n’, a_fitted);
R_0 = a_fitted*beta_fitted*N/((mu_fitted + a_fitted)*(mu_fitted + gamm
fprintf(’R_0:□%.6f\n’, R_0);

rmse_train = sqrt(mean((I_train’ - Y(1:2:end,2)).^2));
meae_train = mean(abs(I_train’ - Y(1:2:end,2)));
```

```
maae_train = max(abs(I_train' - Y(1:2:end,2)));

rmse_test = sqrt(mean((I_test' - Y(2:2:end,2)).^2));
meae_test = mean(abs(I_test' - Y(2:2:end,2)));
maae_test = max(abs(I_test' - Y(2:2:end,2)));

fprintf('Training RMSE (Root Mean Squared Error): %.6f\n', rmse_train);
fprintf('Training MeAE (Mean Absolute Error): %.6f\n', meae_train);
fprintf('Training MaAE (Maximum Absolute Error): %.6f\n', maae_train);
fprintf('Test RMSE (Root Mean Squared Error): %.6f\n', rmse_test);
fprintf('Test MeAE (Mean Absolute Error): %.6f\n', meae_test);
fprintf('Test MaAE (Maximum Absolute Error): %.6f\n', maae_test);

function dy = SEIR_model(t, y, beta, gamma, mu, a, N)
S = y(1);
E = y(2);
I = y(3);
R = y(4);
dy = zeros(4,1);
dy(1) = mu * N - mu * S - beta * S * I;
dy(2) = beta * S * I - (mu + a) * E;
dy(3) = a * E - (gamma + mu) * I;
dy(4) = gamma * I - mu * R;
end
% Define the objective function for least squares fitting
function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
mu = params(3);
a = params(4);
N = 157759;
```

```
I0 = I(1);
E0 = 0;
S0 = N - I0;
R0 = 0;
[T, Y] = ode45(@(t,y) SEIR_model(t, y, beta, gamma, mu, a, N), t, [S0;
I_predicted = interp1(T, Y(:,2), t, ’linear’, ’extrap’);
end
```

6.8 Code to fit the data to SIRS model

```
weeks = 0:48;
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
N = 157759;

guess = [2e−05, 2, 0.06];
options=optimoptions(’lsqcurvefit’);
options = optimoptions(options, ’Display’, ’iter’, ’StepTolerance’,1e−1
upper_bound =[inf, inf, inf];
lower_bound =[0, 0, 0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I), guess
beta_fitted = params_fitted(1);
gamma_fitted = params_fitted(2);
xi_fitted = params_fitted(3);

I0 = I(1);
S0 = N - I0;
R0 = 0;
[t, Y] = ode45(@(t,y) SIRS_model(t, y, beta_fitted, gamma_fitted, xi_fi
figure;
plot(weeks, I, ’r*’, t, Y(:,2), ’b−’);
xlabel(’Weeks’);
```

```
ylabel('Number of Infected Individuals');
legend('Data', 'Fitted Result');
title('SIRS Model');

fprintf('Fitted beta: %.6f\n', beta_fitted);
fprintf('Fitted gamma: %.6f\n', gamma_fitted);
fprintf('Fitted xi: %.6f\n', xi_fitted);
R_0 = beta_fitted*N/gamma_fitted;
fprintf('R_0: %.6f\n', R_0);

rmse = sqrt(mean((I' - Y(:,2)).^2));
meae = mean(abs(I' - Y(:,2)));
maae = max(abs(I' - Y(:,2)));

fprintf('RMSE (Root Mean Squared Error): %.6f\n', rmse);
fprintf('MeAE (Mean Absolute Error): %.6f\n', meae);
fprintf('MaAE (Maximum Absolute Error): %.6f\n', maae);

function dy = SIRS_model(t, y, beta, gamma, xi)
S = y(1);
I = y(2);
R = y(3);
dy = zeros(3,1);
dy(1) = -beta*S*I + xi*R;
dy(2) = beta*S*I - gamma*I;
dy(3) = gamma*I - xi*R;
end
% Define the objective function for least squares fitting
function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
```

```
xi = params(3);
N = 157759;
I0 = I(1);
S0 = N − I0;
R0 = 0;
[T, Y] = ode45(@(t,y) SIRS_model(t, y, beta, gamma, xi), t, [S0; I0; R0
I_predicted = interp1(T, Y(:,2), t, ’linear’, ’extrap’);
end
```

6.9 Fit the separate data to SIRS model separately

```
weeks = 0:48;
weeks_train = weeks(1:2:end);
weeks_test = weeks(2:2:end);
I = [3, 2, 7, 12, 9, 10, 27, 21, 36, 63, 108, 255, 472, 675, 580, 844,
I_train = I(1:2:end);
I_test = I(2:2:end);
N = 157759;

guess = [2e−05, 2, 0.1];
options=optimoptions(’lsqcurvefit’);
options = optimoptions(options, ’Display’, ’iter’, ’StepTolerance’,1e−1
upper_bound =[inf, inf, inf];
lower_bound =[0, 0, 0];
params_fitted = lsqcurvefit(@(params, t) objective(params, t, I), guess
beta_fitted = params_fitted(1);
gamma_fitted = params_fitted(2);
xi_fitted = params_fitted(3);

I0 = I(1);
S0 = N − I0;
```

```
R0 = 0;
[t, Y] = ode45(@(t,y) SIRS_model(t, y, beta_fitted, gamma_fitted, xi_fi
figure;
plot(weeks, I, 'r*', t, Y(:,2), 'b-');
xlabel('Weeks');
ylabel('Number of Infected Individuals');
legend('Data', 'Fitted Result');
title('SIRS Model');

fprintf('Fitted beta: %.6f\n', beta_fitted);
fprintf('Fitted gamma: %.6f\n', gamma_fitted);
fprintf('Fitted xi: %.6f\n', xi_fitted);
R_0 = beta_fitted*N/gamma_fitted;
fprintf('R_0: %.6f\n', R_0);

rmse_train = sqrt(mean((I_train' - Y(1:2:end,2)).^2));
meae_train = mean(abs(I_train' - Y(1:2:end,2)));
maae_train = max(abs(I_train' - Y(1:2:end,2)));

rmse_test = sqrt(mean((I_test' - Y(2:2:end,2)).^2));
meae_test = mean(abs(I_test' - Y(2:2:end,2)));
maae_test = max(abs(I_test' - Y(2:2:end,2)));

fprintf('Training RMSE (Root Mean Squared Error): %.6f\n', rmse_train);
fprintf('Training MeAE (Mean Absolute Error): %.6f\n', meae_train);
fprintf('Training MaAE (Maximum Absolute Error): %.6f\n', maae_train);
fprintf('Test RMSE (Root Mean Squared Error): %.6f\n', rmse_test);
fprintf('Test MeAE (Mean Absolute Error): %.6f\n', meae_test);
fprintf('Test MaAE (Maximum Absolute Error): %.6f\n', maae_test);
```

```
function dy = SIRS_model(t, y, beta, gamma, xi)
S = y(1);
I = y(2);
R = y(3);
dy = zeros(3,1);
dy(1) = -beta*S*I + xi*R;
dy(2) = beta*S*I - gamma*I;
dy(3) = gamma*I - xi*R;
end
% Define the objective function for least squares fitting
function I_predicted = objective(params, t, I)
beta = params(1);
gamma = params(2);
xi = params(3);
N = 157759;
I0 = I(1);
S0 = N - I0;
R0 = 0;
[T, Y] = ode45(@(t,y) SIRS_model(t, y, beta, gamma, xi), t, [S0; I0; R0
I_predicted = interp1(T, Y(:,2), t, ’linear’, ’extrap’);
end
```