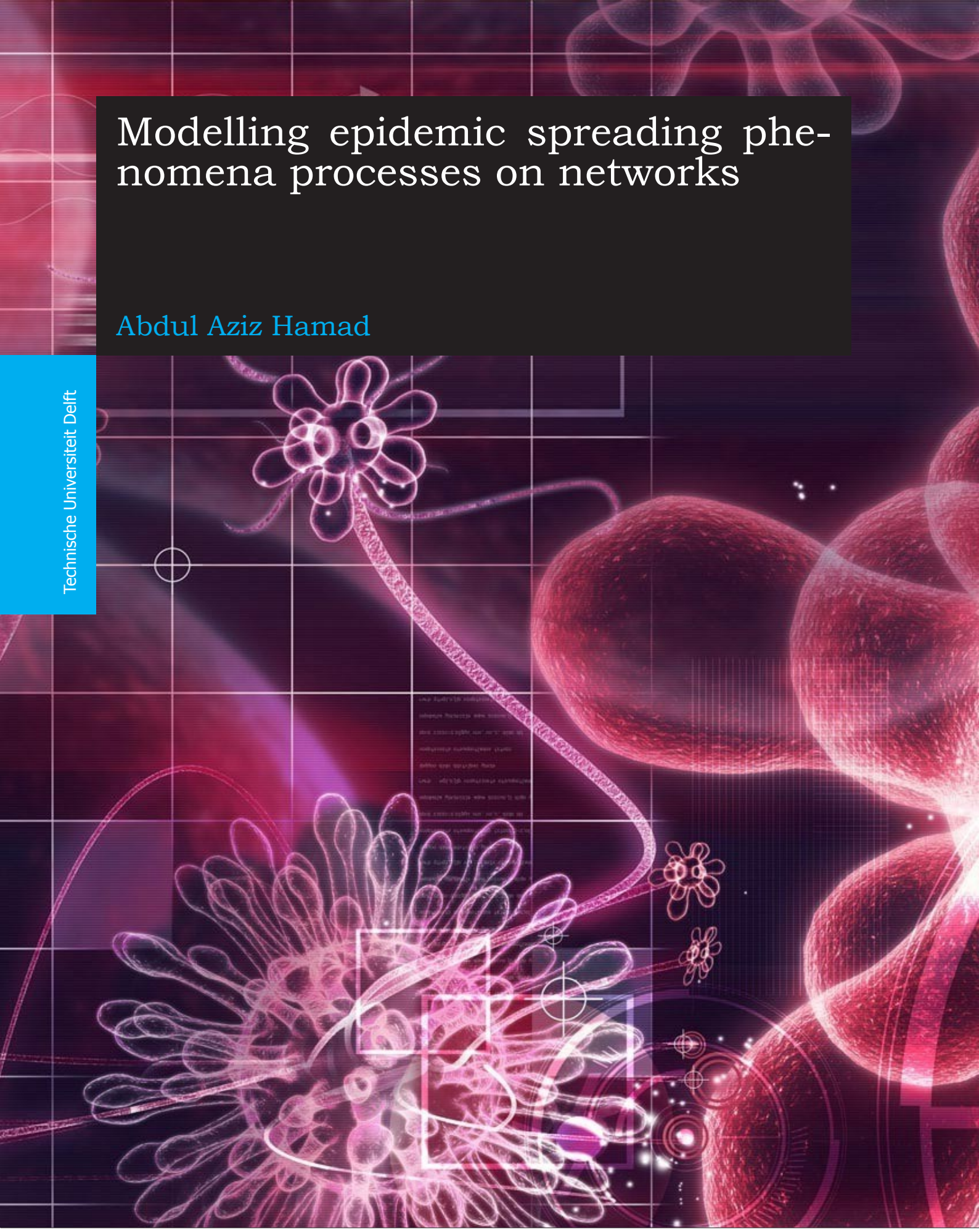
Modelling epidemic spreading phenomena processes on networks

Abdul Aziz Hamad

Technische Universiteit Delft



TU Delft Delft University of Technology

NAS

Challenge the future

PVM 2018-100

Modelling epidemic spreading phenomena processes on networks

by

Abdul Aziz Hamad

in partial fulfillment of the requirements for the degree of

Master of Science
in Electrical Engineering

at the Delft University of Technology,
to be defended publicly on Thursday December 20, 2018 at 10:00 AM.

Daily Supervisor:	Ir. Qiang Liu,	TU Delft
	Ir. Long Ma,	TU Delft
Supervisor:	Prof. dr. ir. Piet Van Mieghem,	TU Delft
Thesis committee:	Prof. dr. ir. Piet Van Mieghem,	TU Delft
	Dr. Jaron Sanders,	TU Delft
	Dr. ir. Jos Weber,	TU Delft

This thesis is confidential and cannot be made public until December 20, 2018.

TU Delft Delft University of Technology

NAS

Preface

This master thesis project is part of the study plan of a Master of Science (MSc) in Electrical Engineering, "Telecommunications and Sensing Systems" track, at the EEMCS faculty of the Delft University of Technology (TU Delft). As this thesis was carried out at the Network Architectures and Services Group (NAS) in TU Delft, I would like at the beginning, to thank professor Piet Van Mieghem for giving me the chance to explore in the field. My time at NAS was very friendly, enjoyable and provided me with many nice memories and experiences that I will keep for life.

I guess I was very lucky having two daily supervisors Qiang Liu and Long Ma. I would like to thank them a lot, it was a pleasure working with them. Here, I am showing my appreciation for Qiang guidance, valuable advice, and for his patience answering and explaining my questions. The quality of the work would not be definitely in this level without his continuous constructive feedback. I am also grateful to Long, for his great ideas and help, especially at the beginning of this thesis, as he gave me all the background, basics information and materials needed in order to execute and start working on the project. Moreover, I would like to thank Jaron Sanders and Jos Weber for being part of my thesis committee.

Finally, I would like to thank my family and all of my friends that supported me during my master...

Abdul Aziz Hamad
Delft, December 2018

Contents

List of Symbols

Graph

A	Adjacency matrix
a_{ij}	Element in A in the i-th row and the j-th column
d_i	Degree of node i in a graph G
G	Graph also denoted as $G(N, L)$
L	Number of links
N	Number of nodes
λ_1	The largest eigenvalue of the network's adjacency matrix A

Epidemics

I_i	The probability of node i being infected
R_i	The probability of node i being recovered
S_i	The probability of node i being susceptible
β	Infection rate
δ	Recovery rate
λ	Immunity rate
u	All one vector

Kalman filter

$f(x)$, F	State transition function/matrix
$h(x)$, H	Measurement function/matrix
J	Number of ensembles
K_k	Kalaman gain at time k
P_k	System state covariance matrix at time k
Q_k	Process noise covariance matrix at time k
R_k	Measurement noise covariance matrix at time k
x_k	System state at time k
$\hat{x}$	Estimated of state x
$\tilde{x}$	Ensemble of state x
$\tilde{x}^{(j)}$	The sample j of state x

Gradient descent (GD)

b	Batch, which is the number of examples used in the GD in a single iteration
$c(x)$	Cost function
$x^{(n)}$	The step n of the state x
$\nabla c(x)$	Gradient of the function $c(x)$
η	Step size

Others

$\mathcal{N}(0, \sigma_n^2)$	Additive White Gaussian Noise (AWGN) with mean 0 and standard deviation (SD) σ_n
σ_n	Noise standard deviation
e_I	The error in fitting the Infection I
e_{fc}	The error in forecasting the Infection I
O	Observations

List of Figures

Abstract

Spreading processes are ubiquitous in nature and societies, e.g. spreading of diseases and computer virus, propagation of messages, and activation of neurons. Computer viruses cause an enormous economic loss. Moreover, many illnesses/diseases still causing a serious threat to public health. For example, the outbreaks of circulating influenza strains cause millions of illness and deaths worldwide every year. Pronounced outbreaks of flu usually occur during winter. This recognized timing allows public health agencies to organize their flu-related mitigation and response activities to prepare for the winter flu season. Although the general wintertime peak of influenza incidence in temperate regions can be easily forecast, the specific intensity, duration, and time of individual local outbreaks are quite changeable. Even after an outbreak has begun, it is still difficult to predict the future characteristics of the epidemic curve. If the diseases/viruses outbreak characteristics could be reliably predicted, the public health response will be better coordinated.

The goal is to develop a fast and accurate epidemic model to estimate, fit and forecast the spreading of an epidemic on a defined network. The aim is to conduct a study over viruses spreading phenomena both theoretically and numerically, then create a general model/algorithm that can be easily applied to different diseases and computer viruses. In this master thesis, we propose a new approach which can be used on real illness/viruses data (such as influenza) to estimate and forecast the epidemic more accurately. The approach is to use a model-inference system combining the network science, susceptible-infected-recovered-susceptible (SIRS) model, statistical filtering techniques and gradient descent. We are able to fit and estimate with a relatively low error compared to other algorithms. Moreover, we forecast the out-breaker with a high accuracy, four weeks before the true out-breaker on synthetic epidemic data. The model is evaluated on a regular graph, Erdös-Rényi graph, Watts-Strogatz small-world graph, & Barabási-Albert graph. Furthermore, the model is carried out on real-world epidemic data (influenza data) for four countries (the Netherlands, Germany, Belgium and the United Kingdom), from the years 2012 to 2017.

1

Introduction

We are part of a network and networks are everywhere around us. Therefore, network science plays, which focuses on complex networks, the main role in different disciplines ranging from man-made infrastructures such as data communications (the internet) and energy networks (smart grid) to biological, neural, social and financial networks. Network science helps to understand the world around us [3]. Modern network science studies dynamic processes on a complex network, and how the network properties influence these dynamic processes. One of the dynamic processes on complex networks is the spreading processes, such as computer viruses propagating on the Internet, rumors spreading on the social network, and epidemic illness/diseases spreading among populations. The exact details of the spreading process are hard to be traced because of the non-linearity of dynamics, network complexity, and large data size. However, the approximation of spreading process can be found and studied using epidemic models [4]. The epidemic model comprises of state variables (e.g. normalized number of infected persons I and normalized number of susceptible persons S) and state parameters. These state variables follow the process spread phenomena on a network. The state parameters (e.g. infectious rate β) represent biological properties of a certain virus strain and host population, which also can vary from region to region and season to season. The dynamic epidemic model accuracy in prediction depends on the appropriate estimation of model state variable and parameters, which represent the forecast initial conditions. In this thesis, we estimate the state variable and parameters by employing ensemble statistical filtering in conjunction with an epidemic model on networks [5–8].

Statistical filtering methods use the observations to recursively inform and train the model [9–11] so that current conditions are better deducted (depicted) and evolving outbreak characteristics (the trajectory of the epidemic infection curve) are better matched. For higher accuracy, the estimation error is corrected with Gradient descent (GD), which gives faster convergence toward the desired values. The epidemic model with the updated state variables and inferred parameters, is propagated forward into the future to generate an accurate and reliable forecast. Skvortsov *et al.* [12] monitor and predict a synthetic epidemic outbreak using particle filter and Susceptible-Infected-Recovered (SIR) epidemic model. Shaman *et al.* [13, 14] monitor and forecast the influenza seasonal outbreaks using ensemble adjustment Kalman filter (EAKF) and Susceptible-Infected-Recovered-Susceptible (SIRS) epidemic model and considering the absolute humidity [15]. In this thesis, we revise the mentioned work considering the epidemic process is on a network rather than one compartment (without network). Then the performance/accuracy is enhanced by using GD and the multi-dimensional graph effect method.

1.1. Research goals and motivation

We study the epidemic spreading phenomena on networks. The goal is to develop a generic fast and accurate algorithm to estimate, fit and forecast epidemic spreading on networks. Our method can be applied to many real epidemic data and can be effectively applied with a small amount of data.

1.2. Thesis organization and outline

The remaining chapters of the thesis contain the following:

Chapter 2
Gives an overview of the background knowledge of complex networks, the epidemic process on networks, Kalman filters and gradient descent.

Chapter 3
Discusses in details the fitting, estimation and forecasting algorithm. The chapter shows also the implementation when the algorithm is validated with synthetic epidemic data and when its test with real-world epidemic data.

Chapter 4
Presents and analyzes the results and the algorithm performance. The chapter evaluates the fitting, estimation and forecasting algorithm validation with synthetic epidemic data spreading on the network. We validate multiple cases of synthetic data, with and without noise, spreading on a regular graph, ER, WS and BA networks. Furthermore, we compare our algorithm with others. Then, the testing results on real-world epidemic data are presented and argued. The real world epidemic data is the World Health Organization (WHO) influenza data for the Netherlands, Germany, Belgium, and the UK from the year 2012 till 2017.

2

Theory and Background

An ensemble filter is employed to estimate the model state variables and parameters of the epidemic process on a network. Then those estimations are corrected using gradient descent for higher accuracy. In this chapter, we introduce the theoretical backgrounds and the concepts used throughout the thesis, such as network models, epidemics on networks, ensemble filters and gradient descent.

2.1. Networks

Network science studies the networks from simple to complex, among diverse fields such as telecommunication networks, computer networks, biological networks, cognitive and semantic networks, and social networks. "Network science consists of the study of network representations of physical, biological, and social phenomena leading to predictive models of these phenomena" [16]. Graph theory is the main building block of network science. Basically, networks can be represented by a graph or multiple graphs, which form a certain topology to perform a specific service. Graphs consist of two simple component, nodes and links. Each node i represents an element in the network (e.g. a router in a computer network, or an individual in a social network, etc.). Each link (ij) is the connection between two nodes i and j. A graph is described by the notation $G(N, L)$, where N is the number of nodes in the network, and L is the number of links. Euler [17] is considered the founder of the Graph theory by using it to solve the Königsberg seven bridges problem.

2.1.1. Algebraic graph theory

In graph $G(N, L)$, links can be weighted, unweighted, directed and undirected. The overall structure is defined by how the nodes are linked to each other to form several types of network topologies, such as regular, complete, ring, star, 2D lattice and tree graphs. Moreover, the graph can be a null Graph, connected graph or non-connected graph. In this master thesis, we only focus on connected undirected unweighted graphs.

The adjacency matrix A is a square N x N symmetric matrix $A^T = A$. The elements $a_{ij} = a_{ji}$ of the matrix indicate whether node i and node j are linked or not in the graph. In undirected unweighted graphs a_{ij} has two values, $a_{ij} = 1$ if node i and j are connected and $a_{ij} = 0$ if not connected.

The degree d_i is the number of neighbors of node i, given by $d_i = \sum_{j=1}^{N} a_{ij}$. Then, the total number of links is $L = \frac{1}{2}\sum_{i=1}^{N} d_i$. The average degree is $\overline{d} = \frac{1}{N}\sum_{i=1}^{N} d_i = \frac{2L}{N}$.

The adjacency matrix A has N eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$, where λ_1 is non-negative and the corresponding eigenvector x_1 has positive elements [18].

2.1.2. Network models

The complete graph K_N is the mother of all graphs and any graph is a subgraph of K_N. In the complete graph, each node is connected to all other nodes and the number of links $L = \frac{N(N-1)}{2}$. The complete graph is also a special case of the regular graph. In the regular graph each node has the same degree $d_i = \overline{d} = k$ for $i = 1, 2, ..., N$.

Erdös-Rényi Random Graph (ER)

Erdős and Rényi [19] introduced the model to generate random graphs. The Erdös-Rényi (ER) random graph is denoted by $G_p(N)$, where N is the total number of nodes and each node pair is connected independently with probability p. The element a_{ij} of the adjacency matrix A (with i≠j) is a Bernoulli random variable. The average number of links ER graph is $E[L] = \frac{N(N-1)}{2}p$. The average degree is $\overline{d} = (N-1)p$. The degree distribution is a binomial distribution $\Pr[d_i = k] = \binom{N-1}{k}p^k(1-p)^{N-1-k}$

Watts-Strogatz (WS) small-world graph

Watts and Strogatz [1] introduced a random graph generation model that produces graphs with small-world properties. The graph has a small average shortest path but a large clustering coefficient. Small world means the average distance between two nodes is proportional to log$N$. Figure 2.1 [1] shows the small-world network.

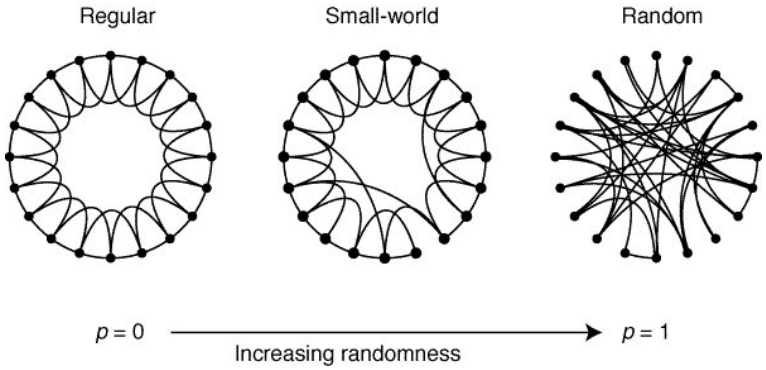


Figure 2.1: Watts-Strogatz small-world model [1]

The model starts with a regular graph, where each node is connected with k nearest nodes. Then with probability p, each link is rewired to a random node.

Barabási-Albert (BA) scale-free graph

The degree distribution of the scale-free network follows a power law distribution $P(k) \sim k^{-\gamma}$, which means few nodes have a very large degree acting as hubs and most nodes have a small degree. The node-degree measurements of the Internet topology indicate that the internet follows a power-law degree distribution $\Pr[d = k] = ck^{-\tau}$. The simplest family of power-law graphs is Barabási-Albert (BA) model [20]. To generate a BA scale-free graph, the model starts with few initial nodes n and follows the degree-biased rule during each step. In each step, a new node will be added and attached to existing nodes with m links to a node with the probability proportional to its degree.

2.2. Epidemics on networks

The epidemic process can model the virus spread in a biological population, the spread of information, computer viruses [6], the propagation of faults or failures in communications and online social networks. The epidemic process described by differential equations involving time derivatives [21]. Analytical or numerical resolution of the system equations facilitates the prediction of the future behavior of the epidemic spread in networks [22]. Epidemic spreading can be a non-Markov process [23, 24] or Markov process [8, 25].

The classical epidemic models assume generally that the population consists of different compartments [26, 27] such as Susceptible (S), Infected (I), and Recovered (R) states. Susceptible state means healthy but susceptible to the disease. Infected state means ill and contagious. Recovered or removed state means removed from the propagation process for a limited time (they can be susceptible again) or unlimited time. Multiple combinations of those stats form many epidemic models, such as SIS, SIR, SIRS, SIRI [28], SIRSI [29], etc. All models start with at least one infected individual that can infect its healthy neighbors, which are susceptible to the disease. The underlining structure of the network has a big influence on the spreading. For example, the SIS epidemic threshold is determined by the largest eigenvalue λ_1 of the adjacency matrix [7]. The autocorrelation of the infection state in the SIS model does not depend on the curing rate if the network is regular [30]

2.2.1. Susceptible-Infected-Susceptible-Recovered (SIRS)

The SIRS model has three states (S,I,R) and three possible transitions, S –> I, I –> R and R –> S. The R state occurs when an infectious individual recovers from the disease and is assumed to have a temporary immunity. The node i can be in one of these states: susceptible $X_i = 0$, infected $X_i = 1$, or recovered $X_i = 2$, for $i = 1, 2, ..., N$. The probability of node i being infected is $I_i \approx \Pr[X_i = 1]$. The probability of node i being susceptible is $S_i \approx \Pr[X_i = 0]$. The probability of node i being recovered is $R_i \approx \Pr[X_i = 2]$. A susceptible node can be infected by infected direct neighbors with rate β. The infected node can be recovered with rate δ. The recovered node can be susceptible again with rate λ. Figure 2.2a shows the SIRS models with the infection rate, the recovery rate and immunity rate.

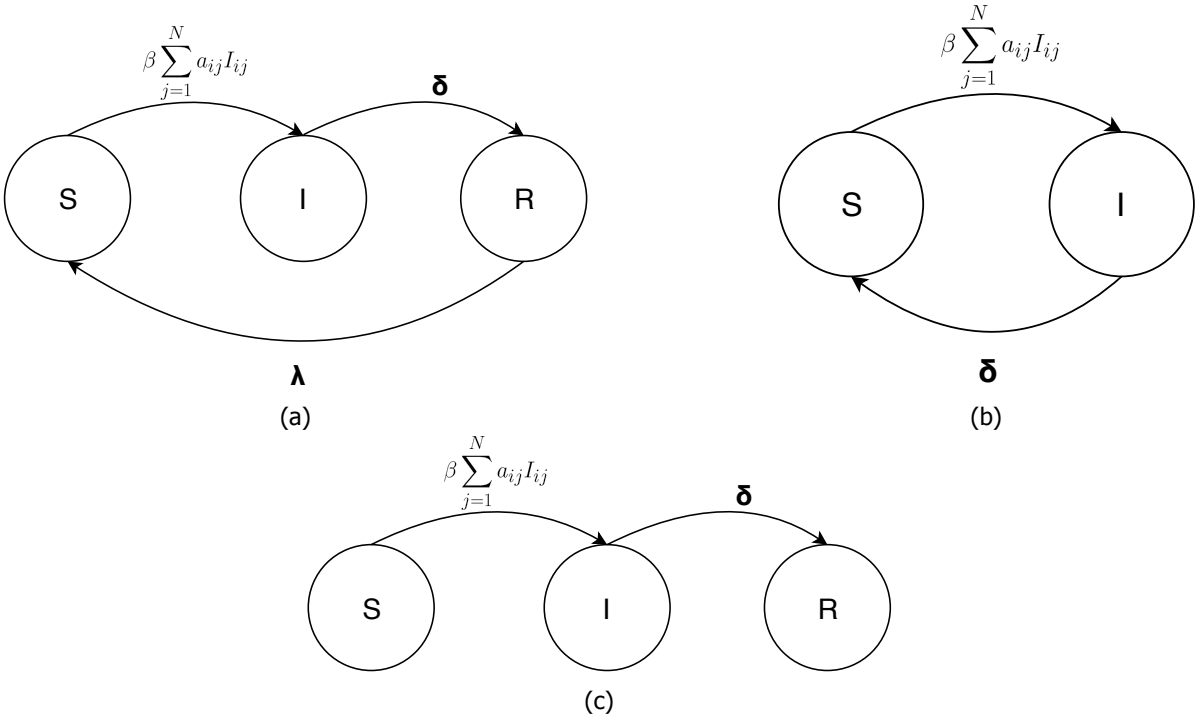


Figure 2.2: Epidemic process: the state transition on node i (a) SIRS, (b) SIS, (c) SIR.

The model is governed by the equations (2.1):

$$\begin{aligned} \frac{dI}{dt} &= \beta \text{diag}(S)AI - \delta I \\ \frac{dS}{dt} &= -\beta \text{diag}(S)AI + \lambda R \\ \frac{dR}{dt} &= \delta I - \lambda R \\ I + S + R &= u \end{aligned} \tag{2.1}$$

where u is the all-one vector, $I = [I_1(t), I_2(t), ..., I_N(t)]^T$ and $S = [S_1(t), S_2(t), ..., S_N(t)]^T$ and diag(S) is a diagonal matrix with elements of S. Figure 2.3 shows the solution of equation (2.1) for different value of λ. If $\lambda = 0$, then the SIRS becomes SIR as shown in figure 2.3c, and if $\lambda >> \delta$, then the SIRS becomes SIS as shown in figure 2.3d. In consequence, the SIS and SIR epidemic models are special cases of the general form SIRS.

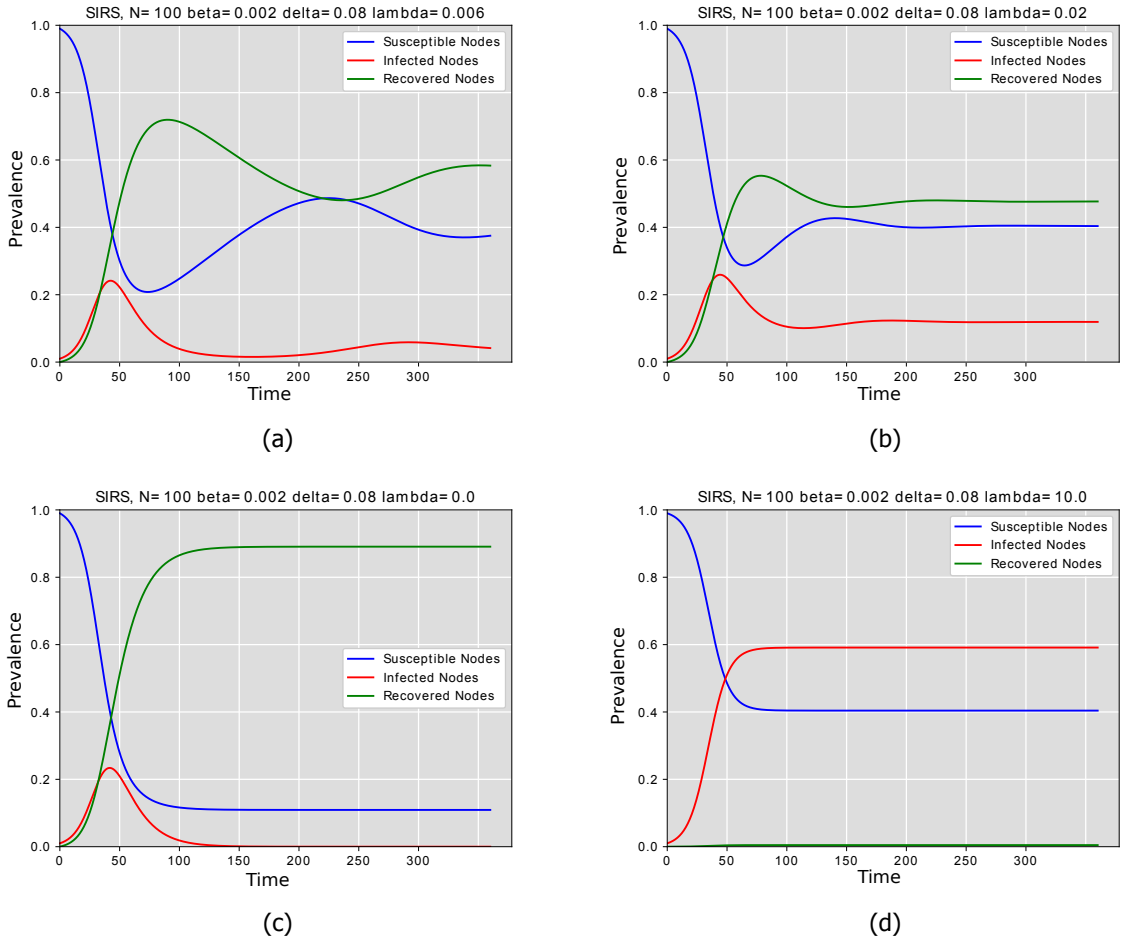


Figure 2.3: SIRS epidemic process on K_{100} network.

2.3. Kalman filter

Kalman filtering is a recursive algorithm which is used to estimate time-dependent physical parameters of a system in the presence of statistical noise and other system inaccuracies. Kalman filter operates in two stages. First is the prediction stage, where the current values of the physical parameters of the system are estimated from the previous measurement. Once the next measurement (corrupted with noise and errors) is observed, the estimated values are updated using a weighted average [31, 32]. As the filters only need the previous state of the system, so it is light on memory and fast, making them suitable for real-time problems.

The Kalman filter addresses the problem of trying to estimate the state x at time k that is governed by the equation:

$$x_k = F_k x_{k-1} + B_k u_{k-1} + w_k \tag{2.2}$$

where F_k is the state transition matrix, B_k is the control input matrix, u_k is control input vector, and the vector $w_k \sim \mathcal{N}(0, Q_k)$ is the process noise vector with the process noise covariance matrix Q_k associated with noisy control inputs.
The observation z at time k of the state x_k is given by:

$$z_k = H_k x_k + v_k \tag{2.3}$$

where v_k represent measurement noise vector at time t, which is assumed to be zero mean Gaussian white noise with covariance matrix R_k (i.e. measurements uncertainty) $v_k \sim \mathcal{N}(0, R_k)$, and H_k is the transformation matrix that maps the state parameters into the measurement domain.

To illustrate the Kalman filter, we present an example system with the state x_k, which is a vector containing two interdependent values a and b:

$$x_k = (a, b) \tag{2.4}$$

We assume the variables a and b follow a Gaussian distribution. Each variable has a mean μ and variance σ^2. The estimate of the system state at time k can be written as:

$$\hat{x}_k = \begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix} \tag{2.5}$$

and the covariance matrix of the system state at time k is given by:

$$P_k = \begin{bmatrix} \Sigma_{\hat{a}\hat{a}} & \Sigma_{\hat{a}\hat{b}} \\ \Sigma_{\hat{b}\hat{a}} & \Sigma_{\hat{b}\hat{b}} \end{bmatrix} \tag{2.6}$$

The state vector $\hat{x}_k$ in equation (2.5) can be estimated using the state transition matrix F_k and the state vector $\hat{x}_{k-1}$ at a previous time $k-1$, in other word by multiplying the state transition matrix with the state vector at time $k-1$, the state vector at time k can be estimated. The state vector estimate at time k can be written as:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} \tag{2.7}$$

The state transition matrix can also be used to calculate the covariance matrix P_k at time k from the one at time $k-1$ [32]:

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T \tag{2.8}$$

Equations (2.7) and (2.8) make the prediction stage of the Kalman filter. The system may experience some external input, if we know this external input, we can modify the filter to account for the external effects. The control input vector u_k contains any external inputs and B_k is the control input matrix which applies the effect of the control input on the state vector. Equations (2.7) can be modified to account for the known external effects:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \tag{2.9}$$

Moreover, some uncertainty can be introduced in the estimated state vector. In other words, we treat the unknown influences as noise with uncertainty covariance matrix Q_k. Taking into account the known and unknown effects, we get the complete expression for the prediction.

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \tag{2.10}$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \tag{2.11}$$

In other words, the new estimate is a prediction made from the previous estimate, plus a correction for known external influences. This is called the predicting stage.

After that, the update stage starts when we obtain an observation vector z_k associated with noise v_k which assumed to be Gaussian with covariance matrix R_k. The covariance matrix R_k is the observation uncertainty in the measurements (for example, due to sensor noise).

The Kalman gain K_k is given by:

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{2.12}$$

where $S_k = R_k + H_k P_{k|k-1} H_k^T$. The updated state estimate $\hat{x}_{k|k}$ is given by:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k y_k \tag{2.13}$$

where $y_k = z_k - H_k \hat{x}_{k|k-1}$. The updated estimate covariance $P_{k|k}$ is given by:

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} (I - K_k H_k)^T + K_k R_k K_k^T \tag{2.14}$$

Equations (2.12), (2.13) and (2.14) give the complete update stage, with $\hat{x}_k$ being the new estimate which will then be used in the next cycle of predict and update. Figure 2.4 shows a graphical representation of the operation of a Kalman filter.

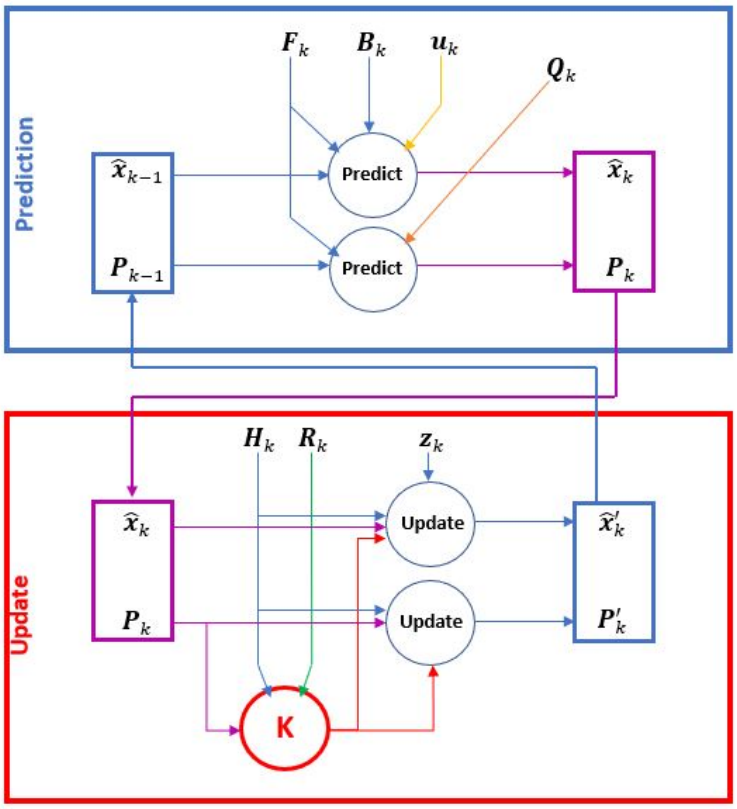


Figure 2.4: Kalman filter

2.3.1. Ensemble Kalman filter (EnKF)

Kalman filter is designed to work with linear systems. Therefore, other flavors of Kalman filter introduce for non-linear systems such as extended Kalman filter (EKF), unscented Kalman filter (UKF), ensemble Kalman filter (EnKF) and ensemble adjustment Kalman filter (EAKF). In Ensemble Kalman filter, an ensemble of J possible state variables is randomly generated using a Monte Carlo method. We consider a nonlinear system with dynamics [33, 34]:

$$x_k = f(x_{k-1}, u_{k-1}) + w_k \tag{2.15}$$

The function $f()$ is the state transition function, and $f(x_{k-1}, u_{k-1})$ projects the state x_{k-1} into the next time period returning x_k with noise $w_k \sim \mathcal{N}(0, Q_k)$.

The observation z_k of the state is given by:

$$z_k = h(x_k) + v_k \tag{2.16}$$

The function $h()$ is the measurement function, which converts or map the state x_k into a measurement domain with noise $v_k \sim \mathcal{N}(0, R_k)$.

The objective is to find the estimation $\hat{x}_k$ of the state x_k given the observations $z_{1:k}$. The filter ensemble is initialized by $\tilde{x} \sim \mathcal{N}(\hat{x}_0, P_0)$, where $\tilde{x} = [\tilde{x}^{(1)}, \tilde{x}^{(2)}, ..., \tilde{x}^{(J)}]$. The EnKF predicting stage is given by:

$$\begin{aligned} \tilde{x}_k^{(j)} &= f(\tilde{x}_{k-1}^{(j)}, u_{k-1}) + w_k \\ \bar{\tilde{x}}_k &= \frac{1}{J}\sum_{j=1}^{J} \tilde{x}_k^{(j)} \\ \tilde{P}_k &= \frac{1}{J-1} e_k e_k^T \end{aligned} \tag{2.17}$$

where e_k is the ensemble error matrix, and it is defined by:

$$e_k = [\tilde{x}_k^{(1)} - \bar{\tilde{x}}_k, ..., \tilde{x}_k^{(J)} - \bar{\tilde{x}}_k] \tag{2.18}$$

The EnKF update stage is given by [35]:

$$\begin{aligned} \tilde{z}_k^{(j)} &= h(\tilde{x}_k^{(j)}) + v_k \\ \bar{\tilde{z}}_k &= \frac{1}{J}\sum_{j=1}^{J} \tilde{z}_k^{(j)} \\ \hat{x}_k^{(j)} &= \tilde{x}_k^{(j)} + K_k(z_k - \tilde{z}_k^{(j)}) \\ \hat{x}_k &= \frac{1}{J}\sum_{j=1}^{J} \hat{x}_k^{(j)} \\ \hat{P}_k &= \tilde{P}_k - K_k P_k^{(zz)} K_k^T \end{aligned} \tag{2.19}$$

where the Kalman gian K_k is given by:

$$K_k = P_k^{(xz)}(P_k^{(zz)})^{-1} \tag{2.20}$$

And $P_k^{(zz)}$ and $P_k^{(xz)}$ are defined by:

$$\begin{aligned} P_k^{(zz)} &= \frac{1}{J-1} l_k l_k^T \\ P_k^{(xz)} &= \frac{1}{J-1} e_k l_k^T \end{aligned} \tag{2.21}$$

where l_k is the ensemble of output error, and it is defined by:

$$l_k = [\tilde{z}_z^{(1)} - \bar{\tilde{z}}_k, ..., \tilde{z}_k^{(J)} - \bar{\tilde{z}}_k] \tag{2.22}$$

Equations (2.19) give the complete update stage of EnKF with J ensemble giving the new estimate $\hat{x}_k$.

2.4. Gradient descent (GD)

Gradient descent (GD) is a generic iterative first-order derivative optimization algorithm, which is capable of finding optimal solutions to a wide variety of problems. The GD minimizes the cost function by tweaking parameters iteratively [2]. The idea is to take steps with an appropriate size to decrease the cost function until the algorithm converges to the minimum of this cost function. The GD is used widely in many application including linear regression and training neural networks.

We assume $c(x)$ is the cost function. The GD algorithm will start with a random initialization $x^{(0)}$ then in each iteration the vector x will be changed by a step [36] in the direction of the negative gradient (figure 2.5). The next values of the vector $x^{(n)}$ is given by the previous vector $x^{(n-1)}$ as following:

$$x^{(n)} = x^{(n-1)} - \eta \nabla_x c(x) \tag{2.23}$$

Equation (2.23) shows that the new $x^{(n)}$ is calculated by subtract $\eta \nabla_x c(x)$ from the $x^{(n-1)}$ to move against the gradient, toward the minimum of the cost function. Here, the step size η comes into play, by multiplying the gradient vector $\nabla_x c(x)$ by η determines how fast the GD algorithm will move. As a result, the final value $x^{(n)}$ may ascertain min($c(x)$).

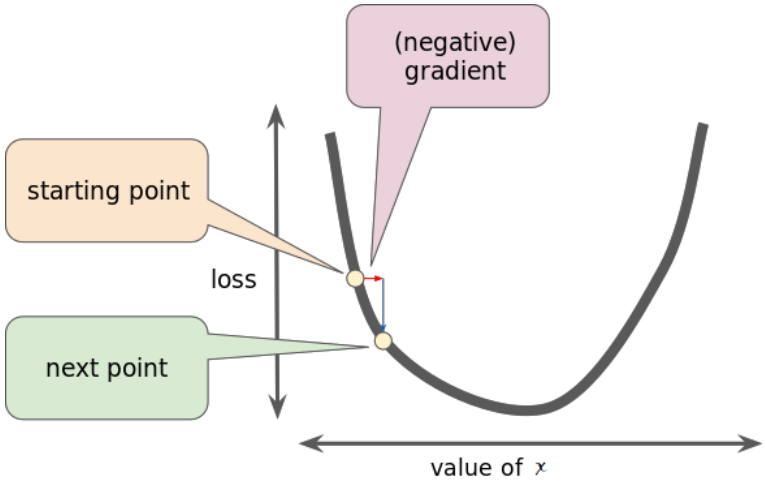
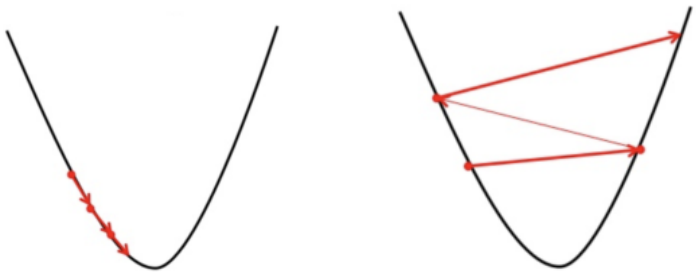


Figure 2.5: Finding the minimum of the cost function

The step size η determines the convergence rate. If the step size is too small as shown in figure 2.6a, then the GD needs many iterations to find the optimal value that minimizes the cost function, which will take a long time. In contrast, if η is too big as shown in figure 2.6b, the algorithm might jump over that optimal value. In worst cases, this might make the algorithm bounces back and forth and diverges very far from the desired value, failing to find a good solution [37].



(a) (b)

Figure 2.6: Learning rate η, (a) Small learning rate, (b) Big learning rate

If the cost function is defined by mean squared error (MSE):

$$c(x) = \frac{1}{b}\sum_{j}(x_j - O_j)^2 \tag{2.24}$$

where x is the estimated values, O is the observations, b is the batch which is the number of examples used in the GD in a single iteration. Then we can define three types of gradient descent. In data science, the types differ from each other by the used amount of data from the dataset [38, 39].

2.4.1. Batch gradient descent

The batch gradient descent (BGD) In each iteration will run over the full training dataset. Therefore, it uses the whole dataset $b = k$ at every step, where k is the number of observations O, making it extremely slow for large datasets.

2.4.2. Stochastic gradient descent

The stochastic gradient descent (SGD) uses a single random example x_j in each iteration, and here $b = 1$. The SGD algorithm much faster than BGD, since it has only one data to manipulate at every

step. SGD is a good choice to train on the huge training dataset. However, it is less regular and returns a noisy gradient compared to BGD. Because of stochastic nature, instead of smooth decreasing until it reaches the minimum, the cost function will bounce around it, decreasing only on average. Over time it will end up very close to the sub-optimal value.

2.4.3. Mini batch gradient descent

Mini-batch gradient descent (MBGD) runs on small random set from the whole dataset to compute the gradients. It is a solution between the BGD and the SGD. MBGD divides the training dataset into mini (small) batches ranging from 10 to 1,000 instances, chosen randomly. Figure 2.7 compares, the paths taken during training, between the three types (BGD, SGD, & MBGD) in feature space. It shows that all of them will end up near the minimum, but BGD stops at the optimal value, while both SGD and MBGD keep bouncing around. However, the computational complexity of BGD is high in each step comparing to SGD.

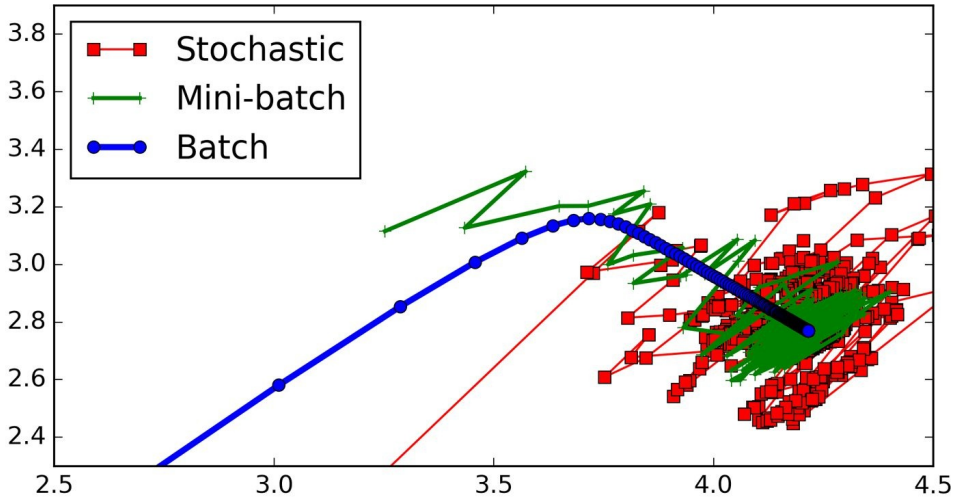


Figure 2.7: Gradient descent types in the parameter vector space [2]

3

Methodology and Implementation

In this chapter, the methodology and implementation will be illustrated and discussed. The first part deals with the framework building and validating with synthetic epidemic data. In this part, we will describe the developed fitting and forecasting epidemic model and algorithm. In the second part, we extend and test the algorithm based on real-world epidemic data.

3.1. Synthetic epidemic data (scenario 1)

The objective is to develop an algorithm to fit and forecast virus spread. Synthetic epidemic spread process on multiple types of networks is used to assist and assess the designed algorithm. The algorithm is developed to fit data and estimate system state variables and parameters. Then, based on the fit and the estimation, the epidemic can be forecasted. The forecast gives a viewpoint of the spreading, in the coming time intervals (sec, min, hours or weeks, depending on the time interval unit). The fitting/estimating means to approximately find the original (true) curves of the state variables (S and I) and model parameters (β, δ and λ) of the epidemic process.

A synthetic epidemic process is generated using a SIRS epidemic model on predetermined network/graph $G(N, L)$, where N nodes are connected by L links, specified by an adjacency matrix A. The parameters β, δ & λ are chosen in the range of the ones in real data, based on Shaman *et al.* [13] that shows the parameters' ranges of influenza in the continental United States. The probability of node i being infected at time t is represented by $I_i(t)$. Additive White Gaussian Noise (AWGN) $n \sim \mathcal{N}(0, \sigma_n^2)$ is added to give randomness $O_i(t) = |I_i(t) + n|$, where $O_i(t)$ is the probability of node i being infected at time t. The synthetic data is used to assist and evaluate the algorithm.

The developed algorithm will process and analyze the observations $O_i(t)$ for the purpose of fitting and forecasting. Mainly, three methods were combined: the SIRS model, the ensemble Kalman filter (EnKF) and the gradient descent (GD). Below, it will be illustrated and described in details.

3.1.1. The fitting and the forecasting

The epidemic fitting algorithm is built based on three main ingredients: SIRS model, EnKF and GD. The fitting algorithm is defined as a problem of estimating the system state variable and parameters at a given discrete time k conditionally on the observations $O[1:k]$, where $O \triangleq \{O_i | i = 1, 2, ..., N\}$. In other words, to find the estimation $\hat{x}[k] = (\hat{I}[k], \hat{S}[k], \hat{\delta}[k], \hat{\lambda}[k])$ of the true probabilities that node i being infected $I_i[k]$, susceptible $S_i[k]$,$\delta[k]$ and $\lambda[k]$, given the observations values $O[1:k]$ for all nodes in the graph, where $\hat{x} \triangleq \{\hat{x}_i | i = 1, 2, ..., N\}$, $\hat{I} \triangleq \{\hat{I}_i | i = 1, 2, ..., N\}$, $\hat{S} \triangleq \{\hat{S}_i | i = 1, 2, ..., N\}$. From Bayes' theorem and Arulampalam *et al.* [10], eq.67 work on particle filters for Online Nonlinear/Non-Gaussian Bayesian tracking we can write:

$$\Pr[\hat{x}[k]|O[1:k]] \propto \Pr[O[k]|\hat{x}[k]] \Pr[\hat{x}[k]|O[1:k-1]] \tag{3.1}$$

Here, the first term on the right-hand side is the observations likelihood of all nodes at time k given the states $\hat{x}$, and the second term is the prior distribution of the system state (EnKF predict stage). The updated distribution (the left-hand side of eq. (3.1)) is called the posterior (EnKF update stage).

In EnKF predict stage, the SIRS model is used as state transition function, to project each ensemble member forward in time to the point at which new observations become available in the update stage. The estimated values then are corrected by the GD algorithm. After the last observation updates the fitting algorithm, the forecast starts. The forecasting is done by solving the SIRS model forward with the last estimated state variables and parameters for the rest of the remaining time intervals. Finally, the results of the fitting/estimating and forecasting is validated with the recorded true values (synthetic epidemic).

3.1.2. Ensemble Kalman filter (EnKF)

The EnKF uses the observations O of the non-linear epidemic process on networks to estimate the SIRS model state variables and parameters. We assume here (scenario 1) the values of A and β are available and constant, which means the network and the type of the spreading virus (e.g. influenza) are known. The state variables and parameters $x = (I, S, \delta, \lambda)$ will be estimated $\hat{x} = (\hat{I}, \hat{S}, \hat{\delta}, \hat{\lambda})$ at each time k. The filter will recursively estimate the system state variables and inference the model parameters. The EnKF will give the estimated updated values $\hat{x} = (\hat{I}, \hat{S}, \hat{\delta}, \hat{\lambda})$ for the SIRS model state variables (I and S) and parameters (δ and λ).

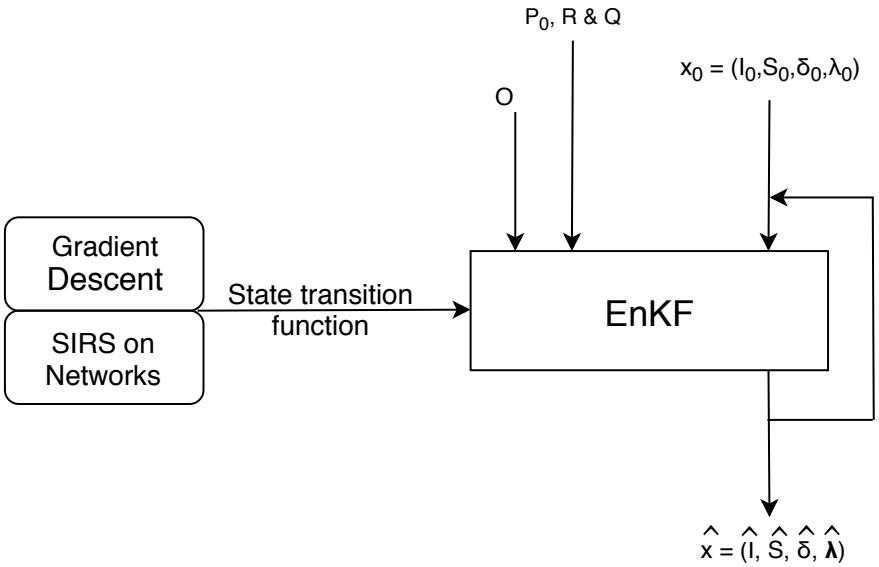


Figure 3.1: Ensemble Kalman filter (EnKF), the developed filter design

Shaman *et al.* [14] apply the ensemble adjustment Kalman filter (EAKF) to entrain weekly observations estimates into a simple humidity-forced SIRS model of influenza. Anderson [40] claims that the EAKF filter performs significantly better than the traditional EnKF. However, Shaman *et al.* [41] compare the performance of six different filters used to model and forecast influenza activity. The results suggest that the ensemble filters are more capable of faithfully recreating the historical influenza observations combined with SIRS model. In addition, Shaman *et al.* [41] claim that EnKF is more accurate than EAKF in the context of SIRS modeling. In this master thesis, we implicate the EnKF with a network SIRS model. Then the error is further corrected and the accuracy is enhanced by GD (more details in 3.1.4).

Figure 3.1 illustrates the filter design, where I_0 and S_0 are the initial state variable, δ_0 and λ_0 are the initial parameters. The filter is driven by the state transition function, which is a nonlinear function due to the non-linearity of the SIRS model. The state transition function projects the current state into the next time period and returns the projected state. The solution of the SIRS equations and the Gradient Descent algorithm is passed to the filter through the state transition function.

3.1.3. SIRS epidemic model on networks

The developed algorithm uses the network SIRS epidemic model to fit/estimate (with EnKF and GD) and forecast the system state $I_i(t)$. This epidemic model is chosen because the SIS and SIR epidemic

models are special cases of general SIRS model as explained in the section 2.2.1.

In the fitting/estimating, The SIRS model plays a role as a state transition function of the non-linear filter (EnKF predict stage). SIRS differential equations (2.1) is solved to project the ensembles $\tilde{x} = (\tilde{I}, \tilde{S}, \tilde{\delta}, \tilde{\lambda})$, into the next time period as following:

$$\tilde{x}^{(j)}_{k|k-1} = f(\tilde{x}^{(j)}_{k-1}) + w_k \tag{3.2}$$

where function $f()$ represents the solved SIRS model. To compensate the computational complexity of the SIRS numerical solving and for higher accuracy we apply one step (one iteration) of GD (section 3.1.4) and the ensemble $\tilde{\delta}$ and $\tilde{\lambda}$ are corrected in eq. (3.7). In the forecasting, the SIRS model is solved using the last estimated parameters for the latest observations values. This operation will show the epidemic spread prediction in future time intervals.

3.1.4. Gradient Descent

The target of using GD here is to correct the small general EnKF error and reach the desired target with higher accuracy and in less EnKF iterations. The aim also to give higher accuracy for the state variables parameters estimation. To implement this our goal is to find the Gradient $\nabla_{\tilde{\delta},\tilde{\lambda}} c(\tilde{I}^{(j)}[k])$, where $c()$ is the cost function and $\tilde{I}^{(j)}[k] = [\tilde{I}^{(j)}_1[k], \tilde{I}^{(j)}_2[k], ..., \tilde{I}^{(j)}_N[k]]$. Here, $\tilde{I}^{(j)}_i[k]$ is the j^{th} value in the ensemble. Also $\tilde{\delta} = [\tilde{\delta}^{(1)}, ..., \tilde{\delta}^{(J)}]$ and $\tilde{\lambda} = [\tilde{\lambda}^{(1)}, ..., \tilde{\lambda}^{(J)}]$ are the ensemble possible value of the state parameter δ and λ respectively, where J is the number of ensemble values. Employing GD requires first the defining of a cost function for each sample $\tilde{I}^{(j)}$ of the ensamble. It is defended based on MSE as following:

$$c(\tilde{I}^{(j)}[k]) = \frac{1}{N} \sum_{i=1}^{N} (\tilde{I}^{(j)}_i[k] - O_i[k])^2 \tag{3.3}$$

We can derive $\nabla c(\tilde{I}^{(j)}[k])$ analytically in respect to the state parameters $\tilde{\delta}$ and $\tilde{\lambda}$ as following:

$$\nabla_{\tilde{\delta},\tilde{\lambda}} c(\tilde{I}^{(j)}[k]) = \begin{pmatrix} \frac{\partial c(\tilde{I}^{(j)}[k])}{\partial \tilde{\delta}} \\ \\ \frac{\partial c(\tilde{I}^{(j)}[k])}{\partial \tilde{\lambda}} \end{pmatrix} = \begin{pmatrix} \frac{\partial c(\tilde{I}^{(j)}[k])}{\partial \tilde{I}^{(j)}[k]} \frac{\partial \tilde{I}^{(j)}[k]}{\partial \tilde{\delta}} \\ \\ \frac{\partial c(\tilde{I}^{(j)}[k])}{\partial \tilde{I}^{(j)}[k]} \frac{\partial \tilde{I}^{(j)}[k]}{\partial \tilde{\lambda}} \end{pmatrix} \tag{3.4}$$

$$= \frac{2}{N} \begin{pmatrix} \sum_{i=1}^{N} (\tilde{I}^{(j)}[k] - O_i[k]) \frac{\partial \tilde{I}^{(j)}_i[k]}{\partial \tilde{\delta}} \\ \\ \sum_{i=1}^{N} (\tilde{I}^{(j)}_i[k] - O_i[k]) \frac{\partial \tilde{I}^{(j)}[k]}{\partial \tilde{\lambda}} \end{pmatrix} \tag{3.5}$$

Theoretically $\frac{\partial \tilde{I}^{(j)}[k]}{\partial \tilde{\delta}}$ and $\frac{\partial \tilde{I}^{(j)}[k]}{\partial \tilde{\lambda}}$ can be found from solving the SIRS model equation (2.1). However, this equation is a non-linear with no analytically explicit solution. Therefore, our own version of the Gradient Descent algorithm is developed. The methodology concept is to find the Gradient Descent of the cost function $\nabla_{\tilde{\delta},\tilde{\lambda}} c(\tilde{I}^{(j)}[k])$ numerically.

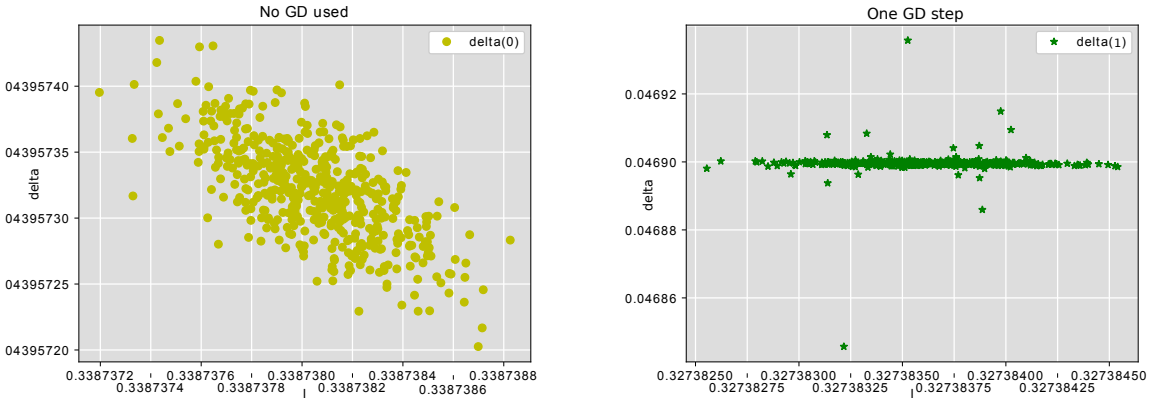
We can write the numerical solution of $\nabla_{\tilde{\delta},\tilde{\lambda}} c(\tilde{I}^{(j)}[k])$ at each EnKF iteration $j = 2, 3, ..., J$ over the J ensembles points $\tilde{I}_i[k]$ for node i at time n, as following:

$$\nabla_{\tilde{\delta},\tilde{\lambda}} c(\tilde{I}^{(j)}[k]) = \begin{pmatrix} \frac{c(\tilde{I}^{(j)}[k]) - c(\tilde{I}^{(j-1)}[k])}{\tilde{\delta}^{(j)}[k] - \tilde{\delta}^{(j-1)}[k]} \\ \\ \frac{c(\tilde{I}^{(j)}[k]) - c(\tilde{I}^{(j-1)}[k])}{\tilde{\lambda}^{(j)}[k] - \tilde{\lambda}^{(j-1)}[k]} \end{pmatrix} \tag{3.6}$$

Then one step update is performed to find new values $\tilde{\delta}^{(j)(1)}[k]$ and $\tilde{\lambda}^{(j)(1)}[k]$ from $\tilde{\delta}^{(j)(0)}[k]$ and $\tilde{\lambda}^{(j)(0)}[k]$, as following:

$$\begin{aligned} \tilde{\delta}^{(j)(1)}[k] &= \tilde{\delta}^{(j)(0)}[k] - \eta \nabla_{\tilde{\delta}} c(\tilde{I}^{(j)}[k]) \\ \tilde{\lambda}^{(j)(1)}[k] &= \tilde{\lambda}^{(j)(0)}[k] - \eta \nabla_{\tilde{\lambda}} c(\tilde{I}^{(j)}[k]) \end{aligned} \tag{3.7}$$

where the notation $^{(1)}$ denote the GD one step. Inspired by the GD we adjust the estimated parameter of each sample in the ensemble. Using GD separately might end up in the local minimum. Therefore, the GD is passed to EnKF through the state transition function. In this way, if the correct initializing values are used, finding the global minimum is possible. Figure 3.2 shows the difference in the state parameter $\tilde{\delta}[k] = [\tilde{\delta}^{(1)}[k], \tilde{\delta}^{(2)}[k], \ldots, \tilde{\delta}^{(J)}[k]]$, where when GD is not used $\tilde{\delta}[k]$ is sparse as shown in 3.2a, while when one GD step is used $\tilde{\delta}[k]$ is concentrated as shown in 3.2b.



(a) (b)

Figure 3.2: The state parameter $\tilde{\delta}[k]$ ensemble values at time $k = 17$, (a) GD is not used, (b) One GD step is used

3.2. Real-world epidemic data

In this section, we focus on fitting and forecasting real virus spread process. We highlight the differences between fitting real-world epidemic data and fitting the synthetic epidemic data in scenario 1. The observation O here represents the total number of persons being infected within an area. The most challenging difficulty is the type of noise associated with the readings (observations) is totally obscure. In real-world epidemic data, just an overall time series reading is provided without having the underlining graph.

The fitting/forecasting algorithm is extended in order to meet the new purpose, especially the EnKF. Data preparation is needed. Moreover, the framework used the developed algorithm "Multidimensional graph effect algorithm" to overcome the problem (See section 3.2.4). In fitting, it works exactly as previously explained in 3.1. Except in synthetic epidemic data scenario, we have N observations curves (one each node) and N fitting curves. While in the real data scenario we have one observation curve and N fitting curves.

The forecast is done by solving the SIRS model on the network forward with the last estimated state variable and parameters for the rest of the remaining time intervals. Since we had N fitting line, N forecasting curves are constructed. In this master thesis, we show all forecasting curves.

3.2.1. Validation with synthetic epidemic data (Scenario 2)

The real-world epidemic data fitting, estimating and forecasting algorithm is validated first with synthetic epidemic data. In scenario 1, we consider that β, A and O_i of each node are known. In this scenario we consider that only $\overline{O} = \frac{\sum_{i=1}^{N} O_i}{N}$ is known without any other information (exactly as real-world epidemic data).

3.2.2. Data preparation (pre-processing)

The influenza Laboratory Surveillance data is collected from the World Health Organization (WHO) [42]. The raw data includes information about the number (observations O) of persons infected with influenza A and B viruses per week for the Netherlands, Germany, Belgium and the UK from 2012 till 2017. The consistency of these raw data in the system is checked first. Normally, Data inconsistencies occur because of observations missing, user entry errors, by corruption in the transmission or storage,

and other reasons. Normally, the dataset of each country has 52 or 53 observations per year (one observation $O[k]$ per week). The concerned country data in each year is compared with the surrounding countries data. Then, the number of observations per year for the studied country is corrected to be identical to the surrounded country. For example, in the WHO influenza data for the year 2012, there are 52 readings per year in the UK, Belgium and Germany, which allows us to assume that the Netherlands also will follow the same number of observations for the same year 2012, as the neighboring countries. Therefore, if the Netherlands had 53 observations this year (2012), this number will be corrected.

After the data consistency is checked, we do the data cleaning process. We considered here, in our raw data the two main issues: Missing data (empty record) problem and zero-value data (while the previous and following observations are high integer values). In case of having only one missing or zero-value observation, it is filled by the average of previous and next values $O[k] = \frac{O[k+1]+O[k-1]}{2}$. In case of having many missing observations in a row, we tag them and take them out totally from the model.

Then we split the observations (the available data of each year) into two parts. The first is upward data $O^{'}$ meaning that the values are increasing. The second is the downward data $O^{''}$ meaning that the values are decreasing. The splitting point is the maximum value. We are splitting the observations to decrease the computation complexity and for the normalization later. As a result, upward data will always occur, while downward data will not occur all the time. Because the infection starts with low value that increases to a maximum point (out-breaker peak) then decreases again.

Since our fitting model includes the SIRS equations, which takes the values of I, R and S in range of [0, 1], the influenza data should be scaled (normalized). We start normalizing the split real data (integer number of infected nodes/people) with an initializing scaling value (the total country/area population S_0). However, the scaling result tends to be small, so we scale it up to the level of the infection rate β. The initials scaled split observations $O_0^{'}$ and $O_0^{''}$ are given by:

$$\begin{aligned} O_0^{'} &= \frac{O^{'}}{S_0}\beta a \\ O_0^{''} &= \frac{O^{''}}{S_0}\beta a \end{aligned} \tag{3.8}$$

where a is a constant depends on the value of β (chosen to make the value of $a\beta > 1$). The scaling method then works depending on the split data and k the number of available observations. Therefore, if there were only upward data, these would be scaled differently than the case of having upwards and downwards at the same time.

$O_1^{'} = O_0^{'}$
$O_1^{''} = O_0^{''}$
if *only upward data* **then**
  while *True* **do**
    if $O_1^{'}$[*last element*] <= $k * \beta * a$ **then**
      $O_1^{'} = O_1^{'} * \beta * a$
    else
      break
    end
  end
else
  while *True* **do**
    if $O_1^{'}$[*last element*] <= $(\beta * a)^2$ **then**
      $O_1^{'} = O_1^{'} * \beta * a$
      $O_1^{''} = O_1^{''} * \beta * a$
    else
      break
    end
  end
end

It's worth to mention that the situation of having both (upwards and downwards data in the same time), is much simpler to fit and forecast than having only upward data.

3.2.3. Noise estimation (EnKF R matrix estimation)

The main problem of dealing with real-world data is that we do not know for sure the type and standard deviation of the noise. The noise, associated with the real data, makes the whole process very difficult. To simplify the problem we decide to deal with a single noise standard deviation value. We assume that S and I should have the same type of noise with the same standard deviation. It is also known that the noise in S and I will impact the estimated value of δ and λ with the same noise type. Therefore, we set a single value for the noise standard deviation based on noticing what kind of information is known about this noise then make the best guess (choosing a small value). This single value then is extended to a $2N + 1$ state uncertainty square matrix R, where N is the total number of nodes, then the matrix R is passed to the EnKF.

3.2.4. Multi-dimensional graph effect

The algorithm is made in order to solve the problems:

- Having time series reading "observation" without having the underlining graph (as in the WHO data).
- Having unknown type and standard deviation of the noise, as previously mentioned.

The scaled observations value at each time interval is copied to all nodes in a graph K_N. In this way, we move from 1-dimensional data to N-dimensional data as shown in figure 3.3. Therefore, if an epidemic initiated to spread, all nodes will be affected at the same time. Then, the spreading process on K_N is perceived to construct later N fitting curve, instead of one fitting curve in the normal method. Each node intends to give a fit line, from its perspective for the observations. Furthermore, the noise effect is overcome by the nodes interaction of the epidemic spread on the network as will be shown in the results. In other words, the graph type and size and the number of links are chosen to reduce the noise to the minimum and support the EnKF to give a better estimation for the model state variable and parameters. This method is compared against not using it (having no network).

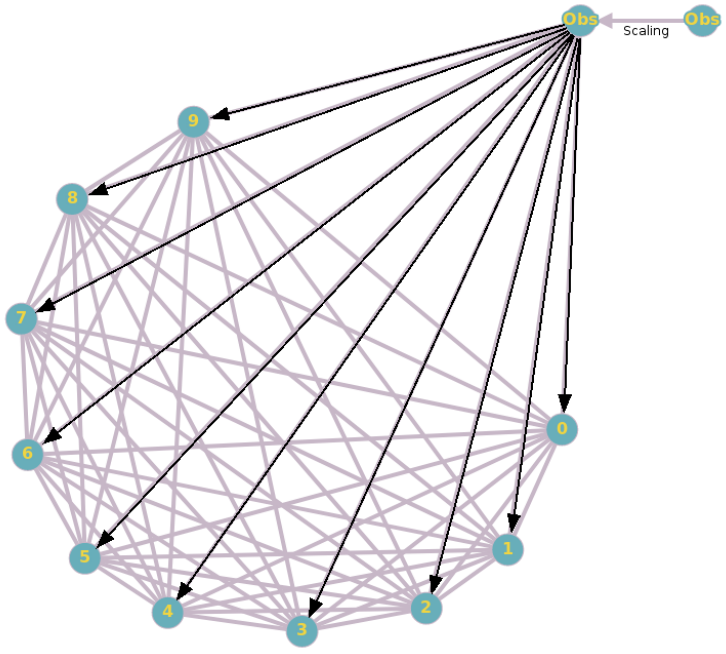


Figure 3.3: Multi-dimensional graph effect

Conclusion

This chapter introduced the methodology in the thesis. Our fitting/forecasting algorithm accuracy and performance is evaluated with synthetic epidemic data spread on a different type of networks. The real world epidemic data fitting/forecasting algorithm is an extended form of the one in the synthetic scenario. In addition to, developing new methods for noise estimation and applying the Multidimensional graph effect algorithm.

4

Results and Analysis

This chapter, we show, analyze and discuss the obtained results. First, the result of fitting, estimating and forecasting on synthetic epidemic data is shown. Then we present the results on real-world epidemic data.

4.1. Synthetic epidemic data (Scenario 1)

In the synthetic epidemic data spread process (Scenario 1), we consider the following values are known:

- The observations O_i, $i = 1, 2, ..., N$ of each node.
- The infection rate β
- The underlining graph $G(N, L)$.

The data (observations O) is generated using the network SIRS model (eq. 2.1) with AWGN stander deviation $\sigma_n = 0$, $\sigma_n = 0.01$ and $\sigma_n = 0.02$, the parameters $\beta = \frac{0.2}{\lambda_1}$, $\delta = 0.05$ and $\lambda = 0.002$, as shown in table below:

Graph model		β	δ	λ
Regular graph (RG) , d=3	G(10,15)	0.06667	0.05	0.002
	G(100,150)	0.06667	0.05	0.002
ER, p=0.4	G(10,12)	0.06668	0.05	0.002
	G(100,1929)	0.0051	0.05	0.002
	G(1000,200183)	0.0005	0.05	0.002
WS, p=0.4	G(10,20)	0.04861	0.05	0.002
	G(100,200)	0.04559	0.05	0.002
BA	G(10,24)	0.0359	0.05	0.002
	G(100,564)	0.01279	0.05	0.002

Table 4.1: Scenario 1, Observations generation parameters

4.1.1. Fitting and estimating

The fitting/estimating goal is to validate our algorithm by accurately estimate the true values of the state variable I, and the state parameters δ and λ. Here, we introduce the the fitting error $e_I = \frac{1}{qN}\sum_{k=1}^{q}\sum_{i=1}^{N}|I_i[k] - \hat{I}_i[k]|$, where $q = 52$ is the last time interval and $I_i[k]$ is the true value of node i is infected at time k.

Figures 4.2, 4.5 and 4.8 show the results of our algorithm on ER, WS, BA graphs respectively, in case the there is no associated noise with the observations. The infection curve fitting error e_I is 0.0006, 0.0005 and 0.00097 respectively. Moreover, the estimate values of δ and λ are very accurate $\hat{\delta} = 0.005$

$\hat{\lambda} = 0.002$, which are the exact true values. Figures 4.3, 4.6 and 4.9 show the same case. However, with noise noise standard deviation (ST) $\sigma_n = 0.01$. The I error fitting is less than 0.003. Furthermore, when the noise is higher $\sigma_n = 0.02$ as in figures 4.4, 4.7 and 4.10, the error also higher $e_I = 0.005$.

Figures 4.11, 4.12 and 4.13 show the results of the same scenario, but without using gradient descent (GD). The error of fitting I is $e_I = 0.004402$, $\hat{\delta} = 0.0479$ $\hat{\lambda} = 0.0021$. Figure A.5 shows the results of the same scenario, but without using GD. The I Error fitting is 0.0065, $\hat{\delta} = 0.0479$ $\hat{\lambda} = 0.0022$.

Figure (4.1) shows all studied graphs/cases fitting error against different AWGN's stander deviation. The comparison between using and not using GD (figure 4.1, table 4.2) shows that using GD has big impact on the fit/estimation accuracy, even with small step size $\eta = 1e^{-6}$.

Graph type			Noise SD σ_n		
			0	**0.01**	**0.02**
RG G(10,15), d=3	GD	e_I	0.000662	0.00247	0.00434
		$\hat{\delta}$	0.05	0.04901	0.04927
		$\hat{\lambda}$	0.002	0.00209	0.00205
	no GD	e_I	0.004402	0.00550	0.00881
		$\hat{\delta}$	0.0479	0.04784	0.04623
		$\hat{\lambda}$	0.0021	0.00211	0.00223
ER G(10,12), p=0.4	GD	e_I	0.000690	0.00307	0.00520
		$\hat{\delta}$	0.05	0.04896	0.0471
		$\hat{\lambda}$	0.002	0.00211	0.002228
	no GD	e_I	0.006262	0.00895	0.01201
		$\hat{\delta}$	0.04728	0.04609	0.04575
		$\hat{\lambda}$	0.00216	0.00224	0.002325
WS G(10,20), p=0.4	GD	e_I	0.000545	0.00240	0.00388
		$\hat{\delta}$	0.05	0.04892	0.04875
		$\hat{\lambda}$	0.002	0.00209	0.00210
	no GD	e_I	0.00461	0.00812	0.00743
		$\hat{\delta}$	0.0478	0.04633	0.04686
		$\hat{\lambda}$	0.00211	0.00219	0.00220
BA G(100,564)	GD	e_I	0.00097	0.00237	0.00583
		$\hat{\delta}$	0.05	0.04925	0.04786
		$\hat{\lambda}$	0.002	0.00205	0.00215
	no GD	e_I	0.00650	0.00703	0.00843
		$\hat{\delta}$	0.04797	0.04102	0.03972
		$\hat{\lambda}$	0.00216	0.00095	0.00098

Table 4.2: Fit/estimate results, synthetic epidemic data (scenario 1)

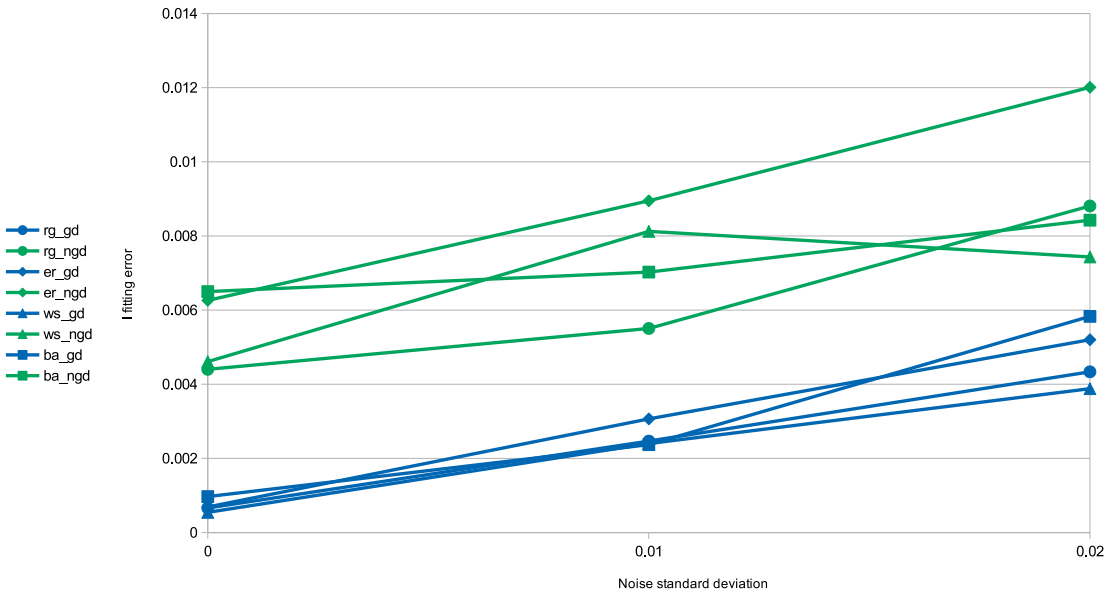
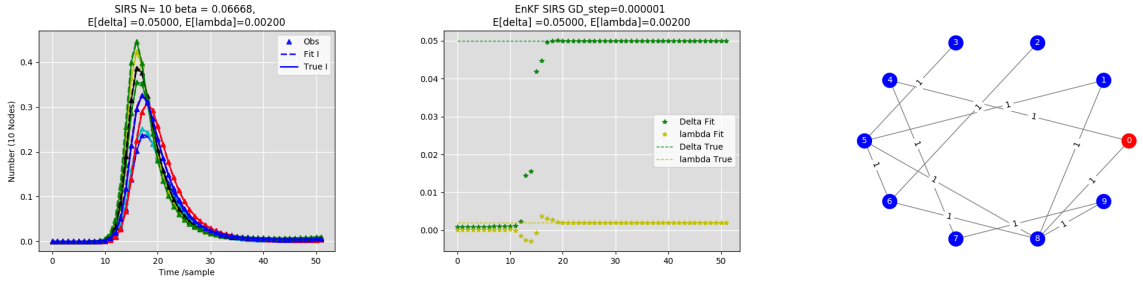


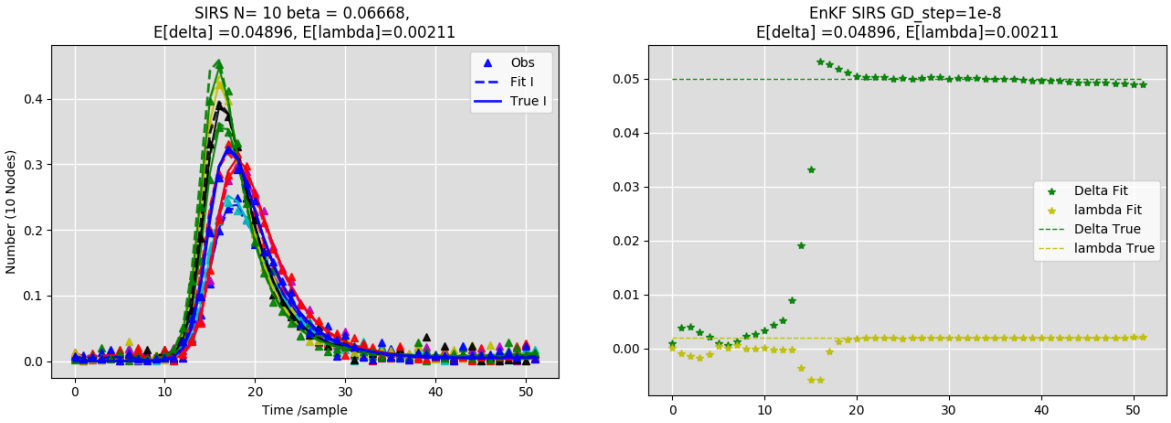
Figure 4.1: The I fitting error e_I for all studied graphs when using and not using GD, synthetic epidemic data (scenario 1)

ER graph



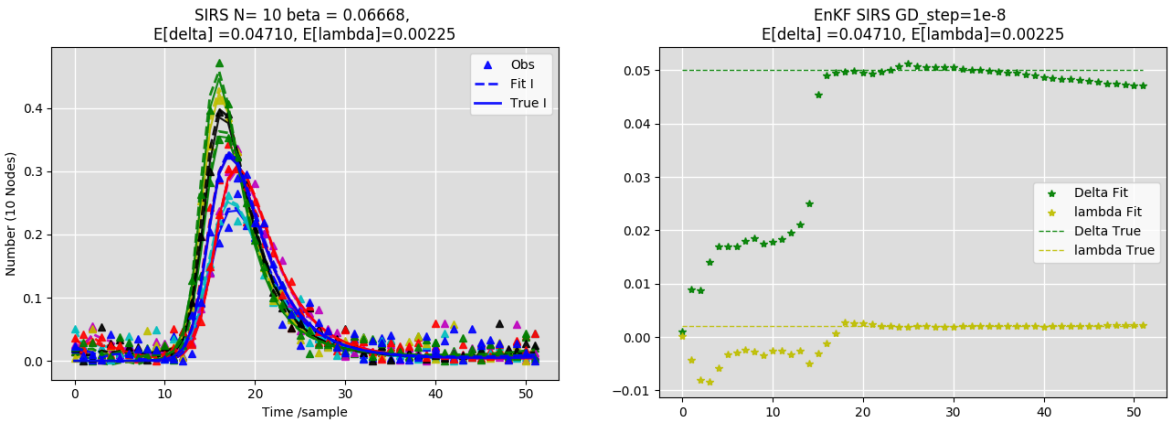
(a) (b) (c)

Figure 4.2: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0$, (a) Infection fitting, $e_I = 0.00006904$, (b) β and λ estimation (c) The ER graph $G(10, 12), p = 0.4$



(a) (b)

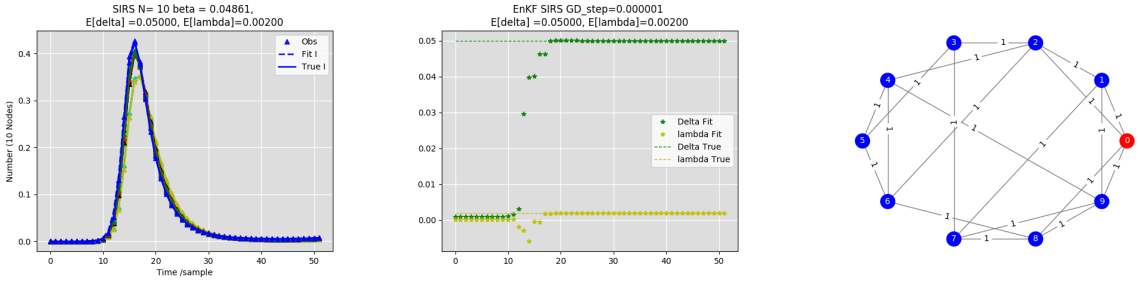
Figure 4.3: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0.01$ associated with O, (a) Infection fitting, $e_I = 0.003065$ (b) β and λ estimation



(a) (b)

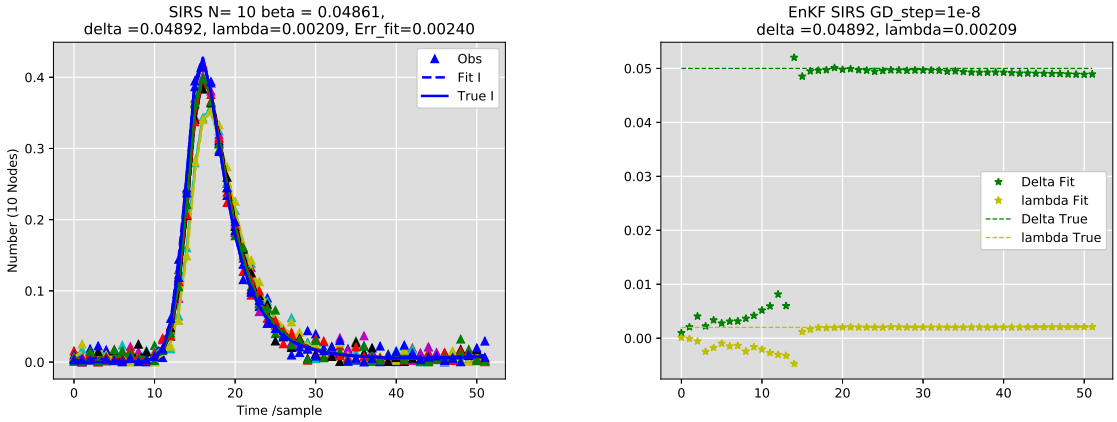
Figure 4.4: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0.02$ associated with O, (a) Infection fitting, $e_I = 0.005201$ (b) β and λ estimation

WS graph



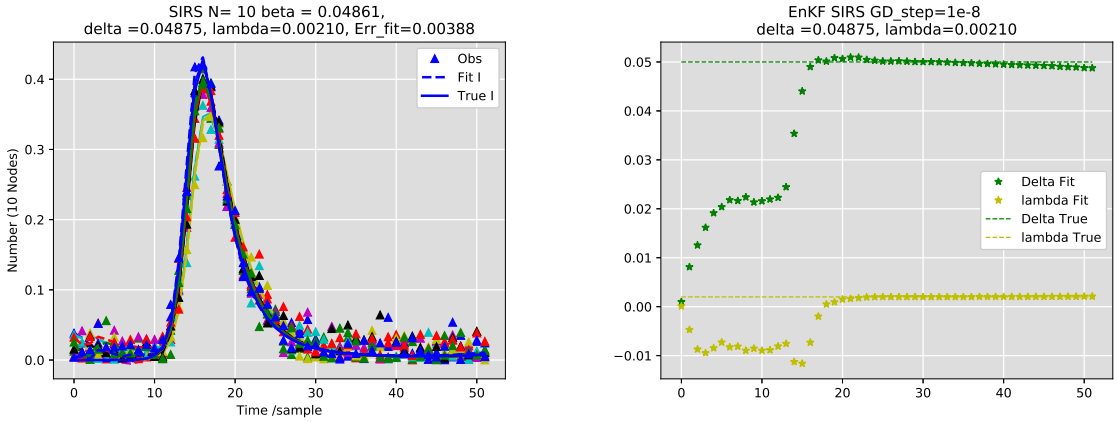
(a) (b) (c)

Figure 4.5: Fitting synthetic epidemic spread on WS graph $G(10, 20)$, when $\sigma_n = 0$, (a) Infection fitting, $e_I = 0.0005446$, (b) β and λ estimation (c) The WS graph $G(10, 24), p = 0.4$



(a) (b)

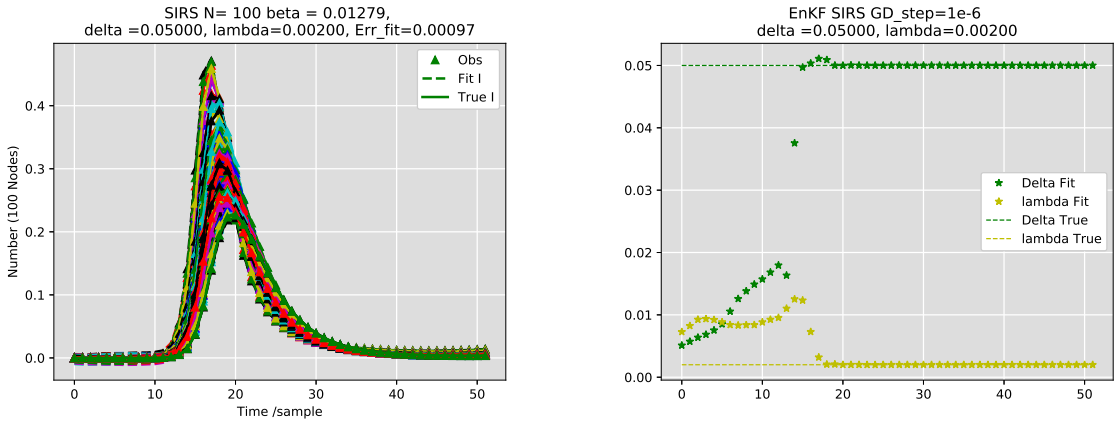
Figure 4.6: Fitting synthetic epidemic spread on WS graph $G(10, 20), p = 0.4$, when $\sigma_n = 0.01$ associated with O, (a) Infection fitting, (b) β and λ estimation



(a) (b)

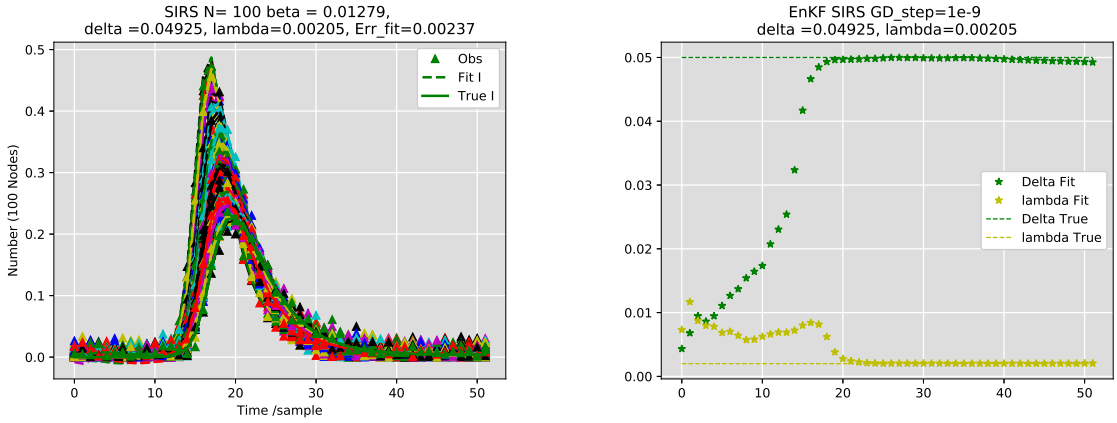
Figure 4.7: Fitting synthetic epidemic spread on WS graph $G(10, 20), p = 0.4$, when $\sigma_n = 0.02$ associated with O, (a) Infection fitting, (b) β and λ estimation

BA graph



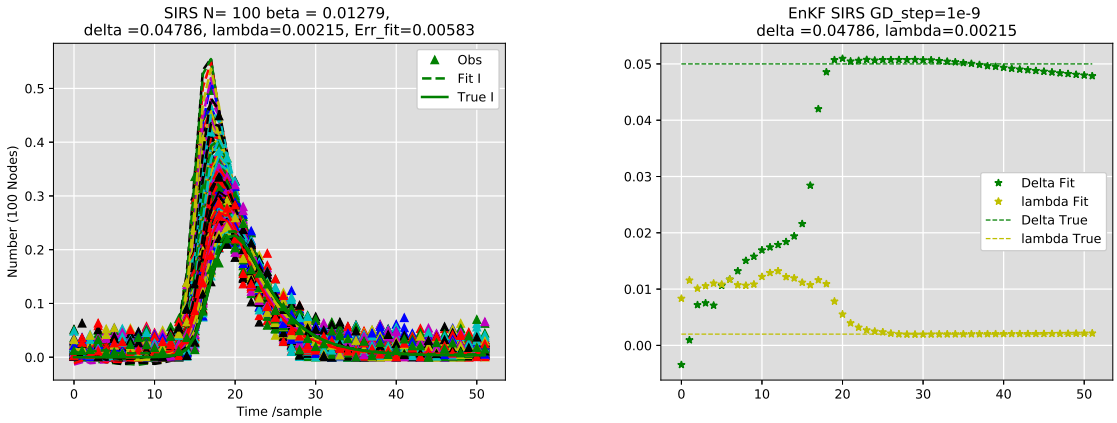
(a) (b)

Figure 4.8: Fitting synthetic epidemic spread on BA graph $G(100, 564)$, when $\sigma_n = 0$ associated with O, (a) Infection fitting, (b) β and λ estimation



(a) (b)

Figure 4.9: Fitting synthetic epidemic spread on BA graph $G(100, 564)$, when $\sigma_n = 0.01$ associated with O, (a) Infection fitting, (b) β and λ estimation



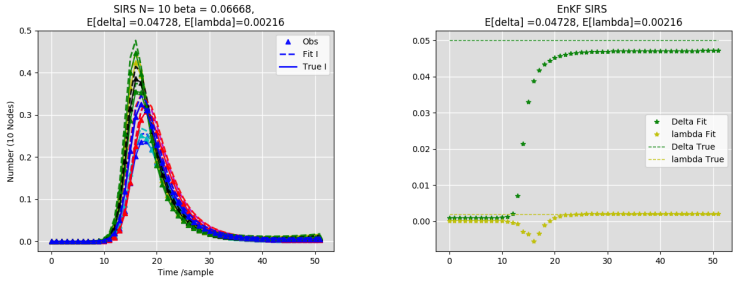
(a) (b)

Figure 4.10: Fitting synthetic epidemic spread on BA graph $G(100, 564)$, when $\sigma_n = 0.02$ associated with O, (a) Infection fitting, (b) β and λ estimation

Fitting without using GD

In this section we show the fitting/estimation when GD is not used. The figures 4.11, 4.12, 4.13, A.4, A.5 and A.6 show that there is always a difference between the true values of δ and λ and the estimated ones. Our interpretation that if we set $R = 0$ in EnKF maybe this gap will be illuminated. However, this cannot happen because if we do then $P_k^{(zz)}$ (Kalman gain eq. (2.20)) cannot be inverted.

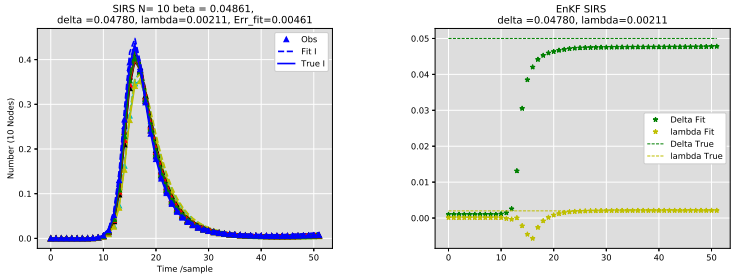
ER graph without using GD



(a) (b)

Figure 4.11: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0$, GD was not used, (a) Infection fitting, $e_I = 0.0062621$, (b) β and λ estimation

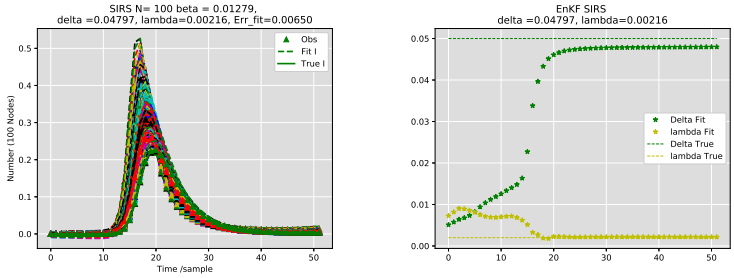
WS graph without using GD



(a) (b)

Figure 4.12: Fitting synthetic epidemic spread on WS graph $G(10,20), p = 0.4$, when $\sigma_n = 0$, GD was not used, (a) Infection fitting, (b) β and λ estimation

BA graph without using GD

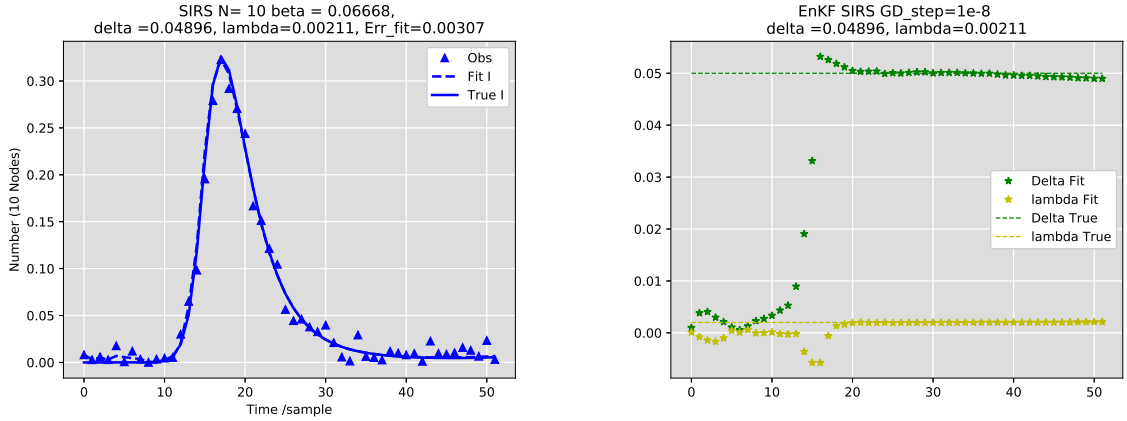


(a) (b)

Figure 4.13: Fitting synthetic epidemic spread on BA graph $G(100,564)$, when $\sigma_n = 0$, GD was not used, (a) Infection fitting, (b) β and λ estimation

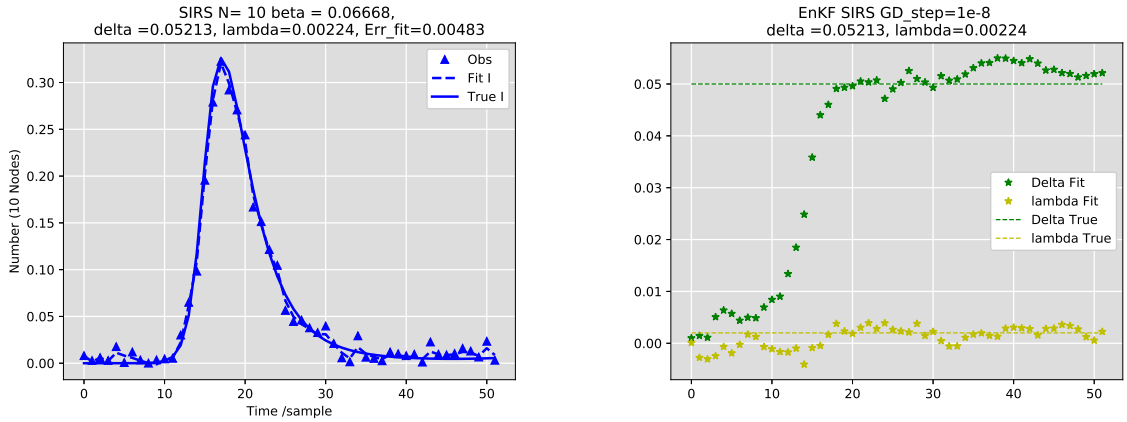
4.1.2. EnKF Q matrix value

In this section, we illustrate the effect of different values of the process noise covariance matrix Q compared to the measurement uncertainty covariance matrix R. Figure 4.14 shows the case when $Q << R$, which means we rely on our SIRS model more than the observations, and this is the chosen case in the thesis. Figure 4.15 shows the case when $Q \approx R$. Figure 4.16 shows the case when $Q >> R$ which here the fit rely more on the observations, and shows that the fit line is just connecting the observations.



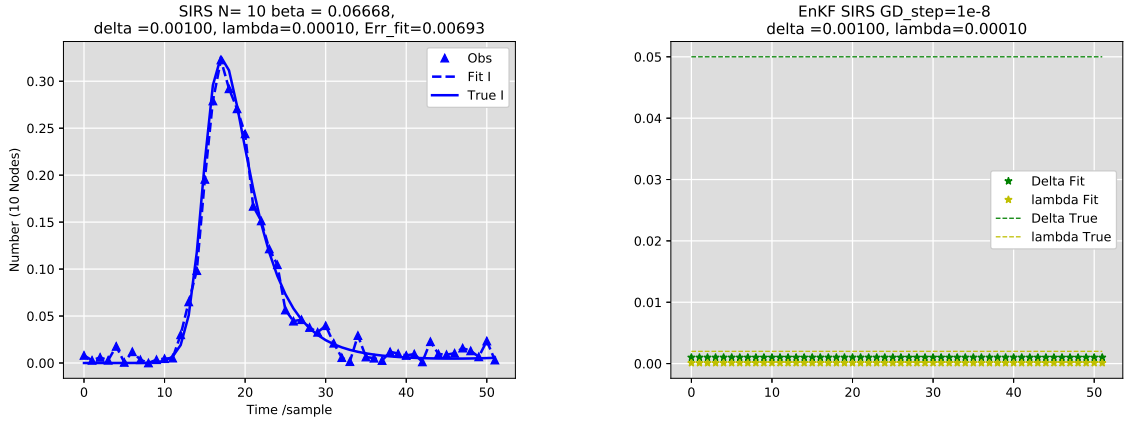
(a) (b)

Figure 4.14: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0.01$ associated with O, $Q << R$ (a) Infection fitting of node $i = 0$, (b) β and λ estimation



(a) (b)

Figure 4.15: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0.01$ associated with O, $Q \approx R$ (a) Infection fitting of node $i = 0$, (b) β and λ estimation



(a) (b)

Figure 4.16: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0.01$ associated with O, $Q >> R$ (a) Infection fitting of node $i = 0$, (b) β and λ estimation

4.1.3. Forecasting

In this section, we show the forecast result from 0-4 weeks before the out-breaker peek. The forecast is done in two cases, with and without Additive white Gaussian noise (AWGN). Figure 4.17 shows the forecast in case the observations have no noise associated with it. Figure 4.18 shows the forecast in case the observations have noise with $\sigma_n = 0.01$ associated with it. Figure 4.19 shows the forecast in case the observation have noise with $\sigma_n = 0.02$ associated with it. The Figures contain the following:

- The observations O, represented by triangles each color for a specific node in the network.
- The true infection curve, represented by a blue line.
- The fitting/estimating infection curve, represented by blue dash-line.
- The forecasting infection curves, represented by yellow stars.

All figures show that we are able to forecast the out-breaker time (week 16 in figure 4.17 and 4.19. Week 14 in figure 4.18) correctly, 3 weeks (4 weeks in the noisy cases) before the true out-breaker. Also it shows the error in forecasting e_{fc}.

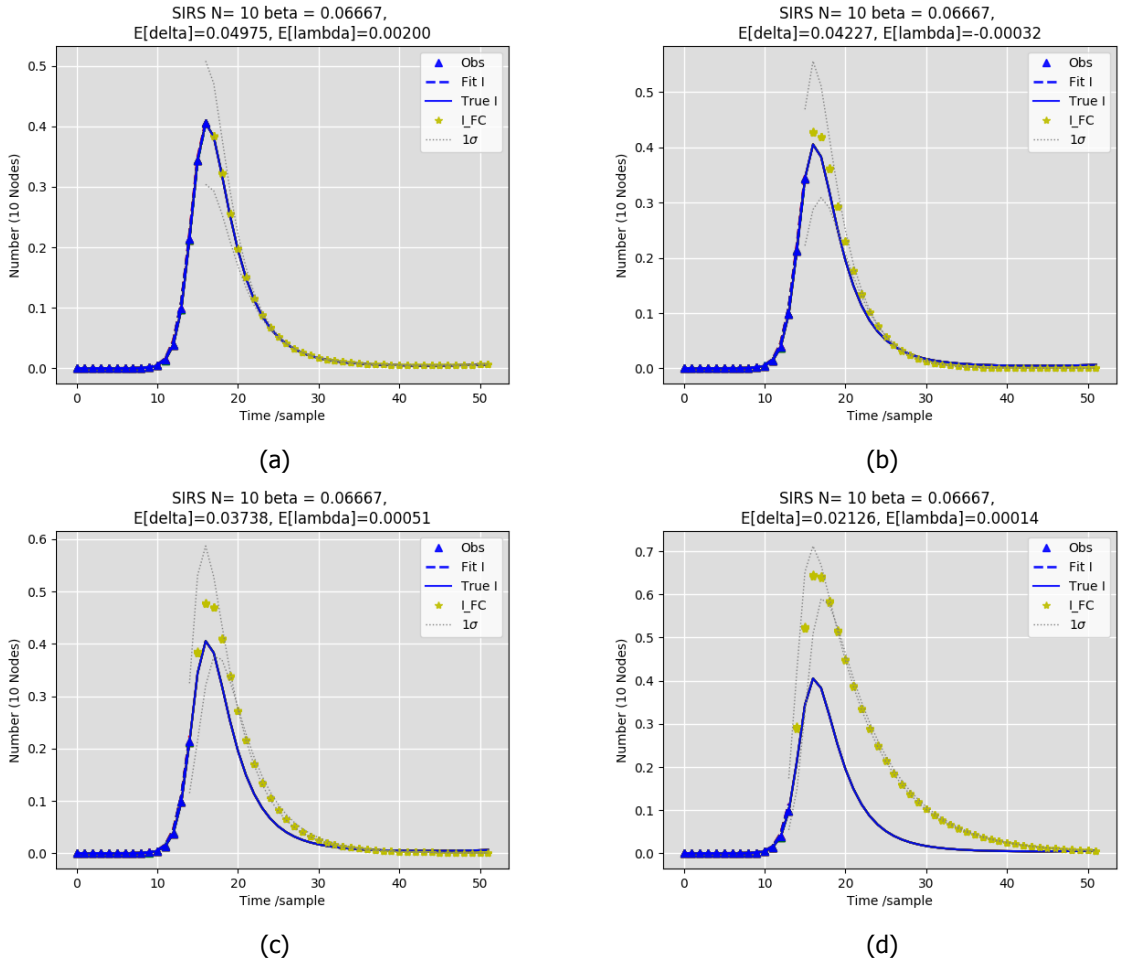
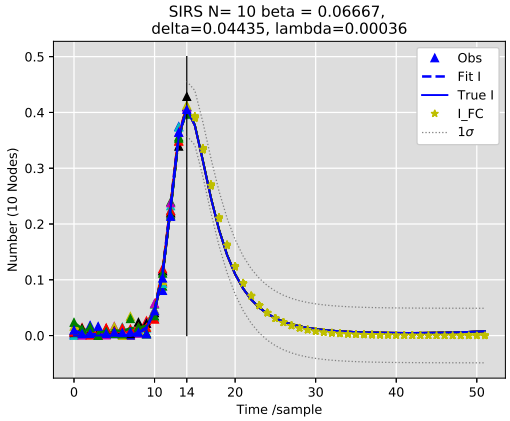
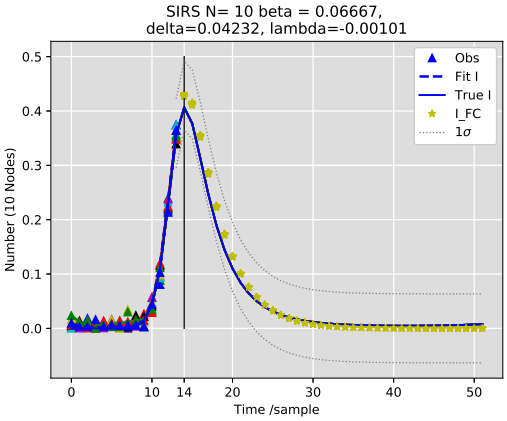


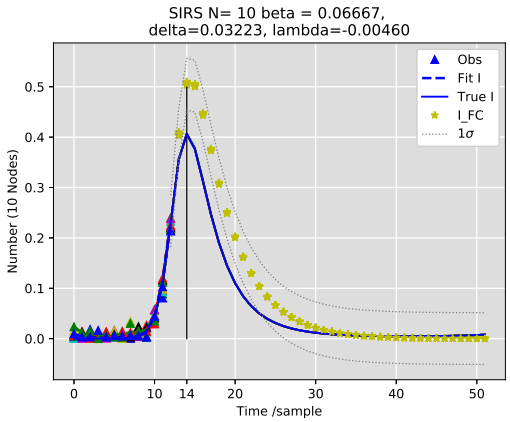
Figure 4.17: Forecasting synthetic epidemic spread on regular graph $G(10, 15)$ $d = 3$, when $\sigma_n = 0$, $\eta = 1e^{-6}$, (a) after the out breaker, $e_{fc} = 0.000534$ (b) 1 weeks before the out breaker, $e_{fc} = 0.009992$ (c) 2 weeks before the out breaker, $e_{fc} = 0.0220$ (d) 3 weeks before the out breaker, $e_{fc} = 0.0926$



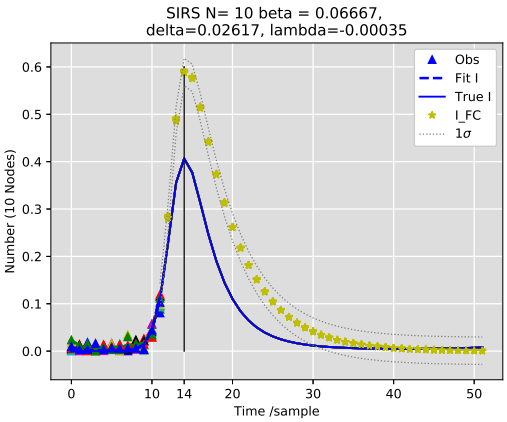
(a)



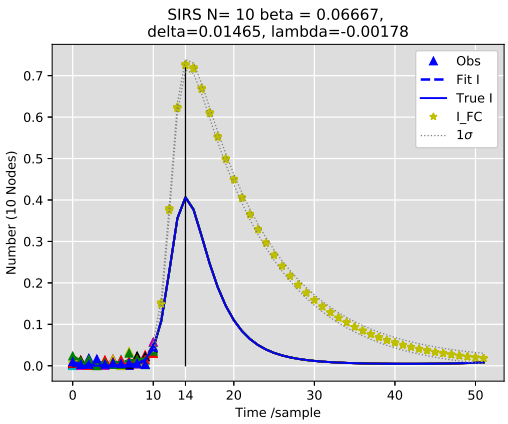
(b)



(c)



(d)



(e)

Figure 4.18: Forecasting synthetic epidemic spread on regular graph $G(10, 15)$ $d = 3$, when $\sigma_n = 0.01$, $\eta = 1e^{-6}$, (a) after the out breaker, $e_{fc} = 0.00652576637513$, (b) 1 weeks before the out breaker, $e_{fc} = 0.00955553359883$, (c) 2 weeks before the out breaker, $e_{fc} = 0.0324632301396$, (d) 3 weeks before the out breaker, $e_{fc} = 0.0572394176196$, (e) 4 weeks before the out breaker, $e_{fc} = 0.139732852203$.

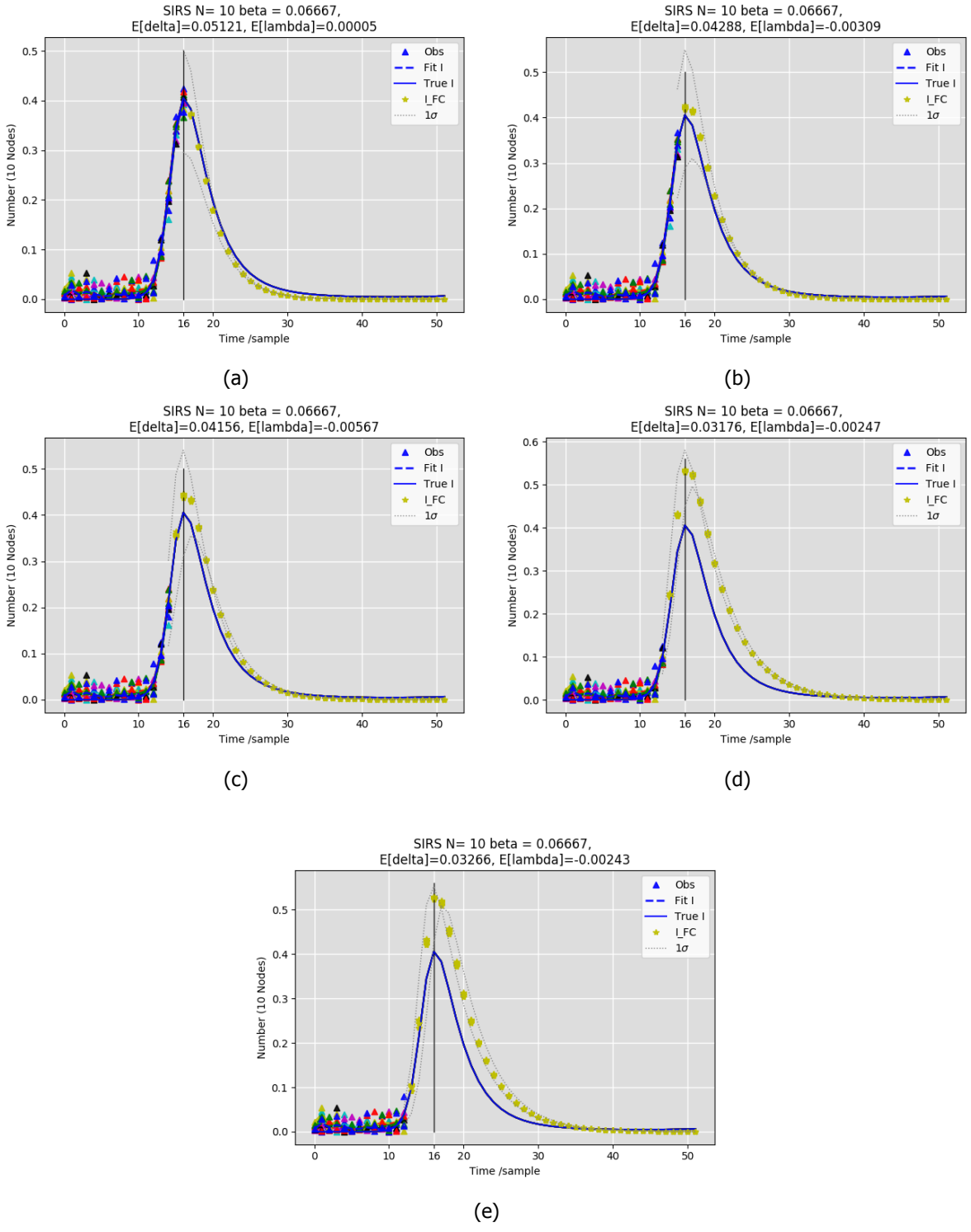


Figure 4.19: Forecasting synthetic epidemic spread on regular graph $G(10, 15)$ $d = 3$, when $\sigma_n = 0.02$, $\eta = 1e^{-6}$, (a) after the out breaker, $e_{fc} = 0.00907431982373$, (b) 1 weeks before the out breaker, $e_{fc} = 0.00917064201825$, (c) 2 weeks before the out breaker, $e_{fc} = 0.0121379037079$, (d) 3 weeks before the out breaker, $e_{fc} = 0.0366554606209$, (e) 4 weeks before the out breaker, $e_{fc} = 0.0332912332529$.

4.2. Synthetic epidemic data (Scenario 2)

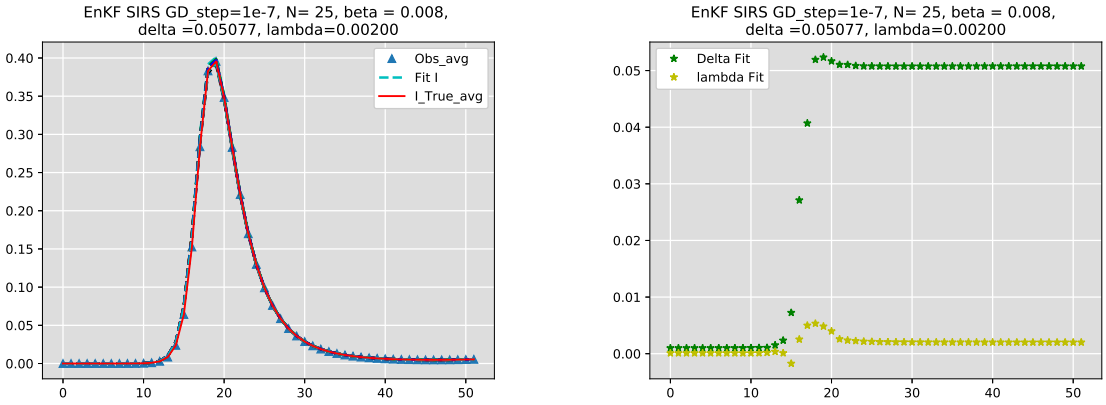
This section shows the Scenario 2 of Synthetic epidemic data fitting and forecasting. The data (average observations) is generated in the same way as 4.1. We present our algorithm results after processing averaged observations. In this data, only the average observations (probability number of nodes is infected per week) is provided. In the process we consider the Multi-dimensional graph effect algorithm has a complete graph K_{25} and $\beta = 0.008$.

4.2.1. Fitting and estimating

In the fitting in the scenario 2, we consider the following values is known:

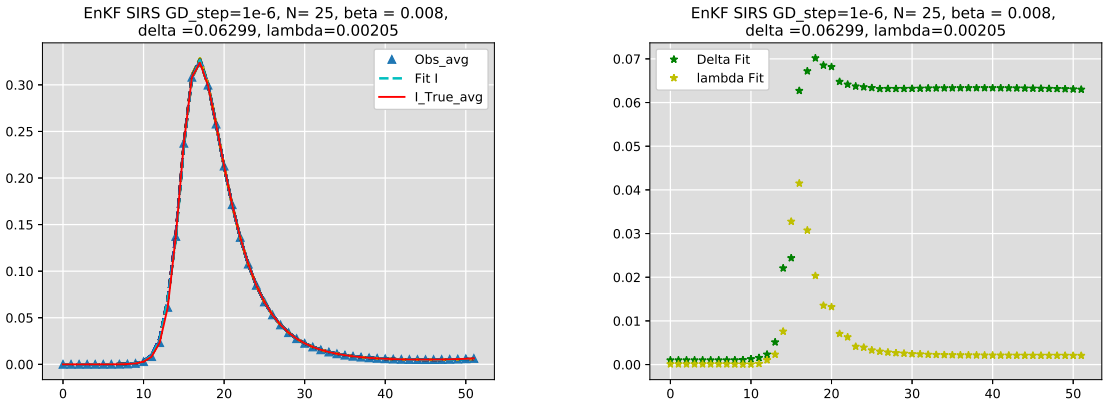
- The average observations $\overline{O} = \frac{\sum_{i=0}^{N-1} O_i}{N}$.

The goal is to validate our algorithm by accurately estimate the true value of the state variable I. All figures show that we are able to fit I and estimate δ and λ.



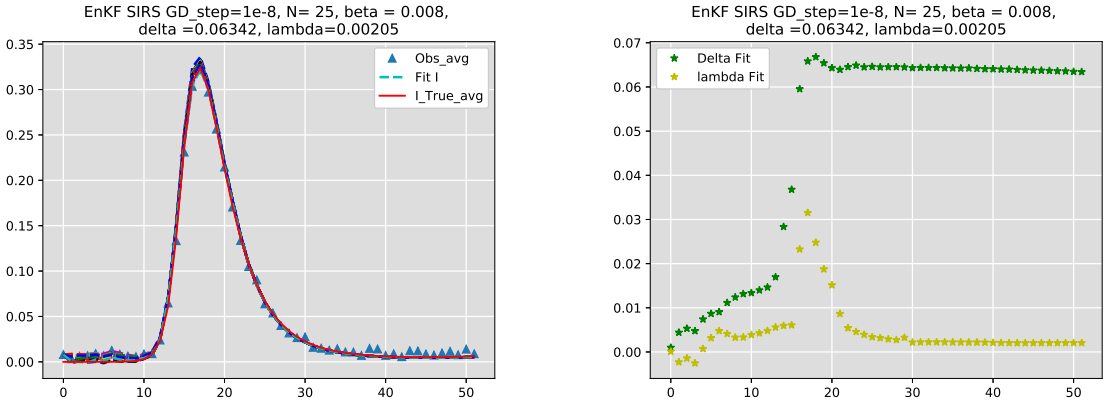
(a) (b)

Figure 4.20: Fitting synthetic epidemic spread on ER graph $G(100, 1929), p = 0.4$, when $\sigma_n = 0$ associated with O, (a) Infection fitting, $e_I = 0.00096804$ (b) β and λ estimation



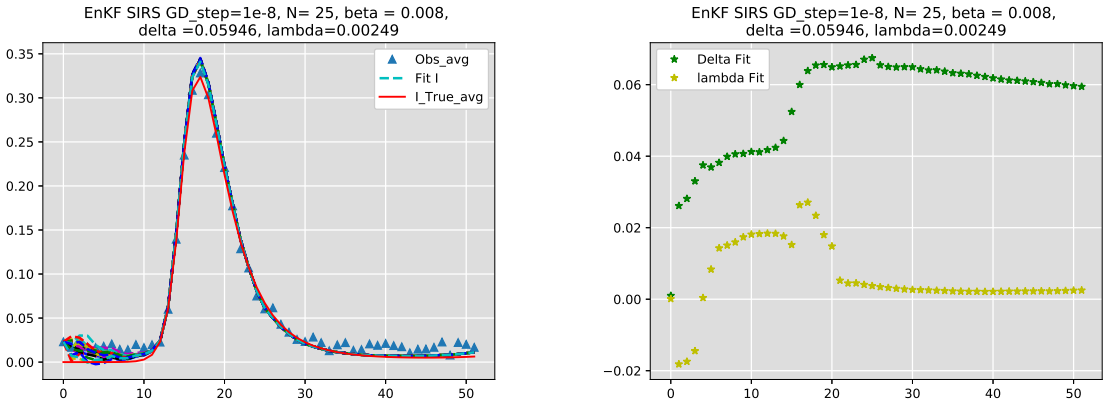
(a) (b)

Figure 4.21: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0$ associated with O, (a) Infection fitting, $e_I = 0.0005383$ (b) β and λ estimation



(a) (b)

Figure 4.22: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0.01$ associated with O, (a) Infection fitting, $e_I = 0.0027714$ (b) β and λ estimation



(a) (b)

Figure 4.23: Fitting synthetic epidemic spread on ER graph $G(10, 12), p = 0.4$, when $\sigma_n = 0.02$ associated with O, (a) Infection fitting, $e_I = 0.00560172$ (b) β and λ estimation

4.2.2. Forecasting

In this section, we show the forecast result from 0-4 weeks before the out-breaker. The forecast is done in two cases, with and without Additive white Gaussian noise (AWGN). The Figures contain the following:

- The average observations $\overline{O}$, represented by triangles.
- The average true infection curve, represented by a red line.
- The fitting/estimating infection curve, represented by N dash-line.
- The forecasting infection curve, represented by yellow stars.

All figures show that we are able to forecast the out-breaker time (week 17) correctly, 4 weeks before the true out-breaker.

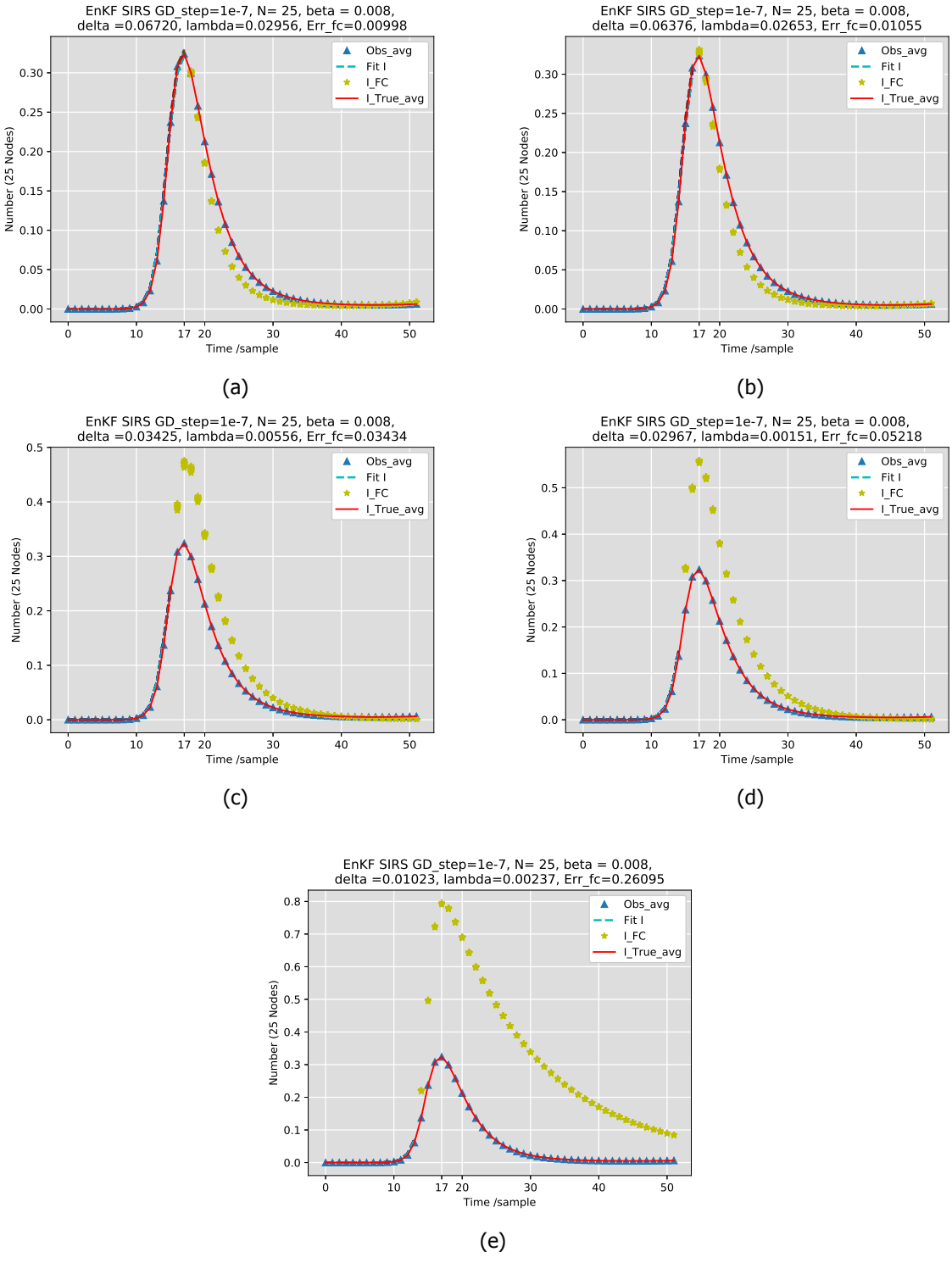


Figure 4.24: Forecasting synthetic epidemic spread on ER $G(10, 12)$ $p = 0.4$, when $\sigma_n = 0$, $\eta = 1e^{-7}$, (a) after the out breaker, (b) 1 weeks before the out breaker, (c) 2 weeks before the out breaker, (d) 3 weeks before the out breaker, (e) 4 weeks before the out breaker.

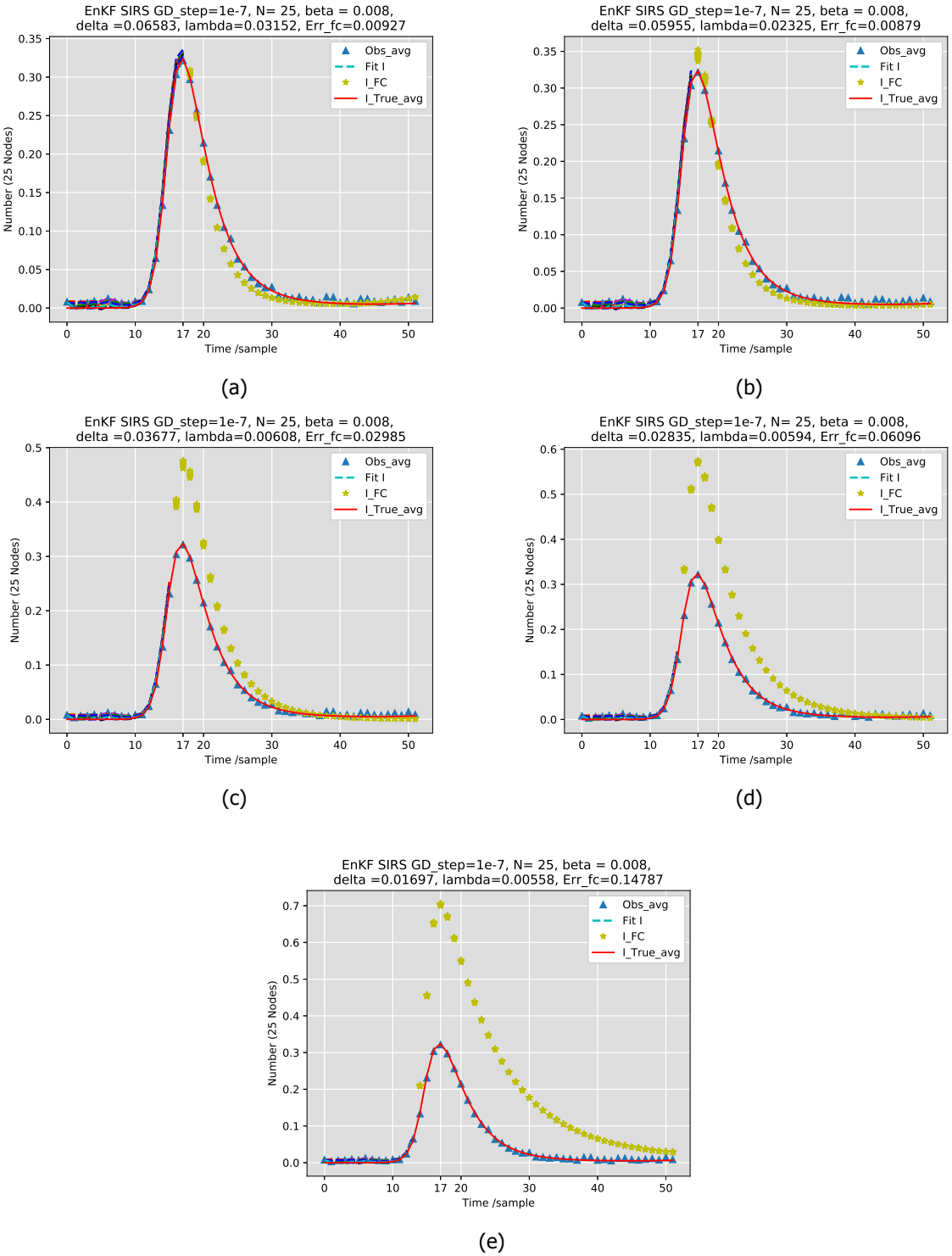
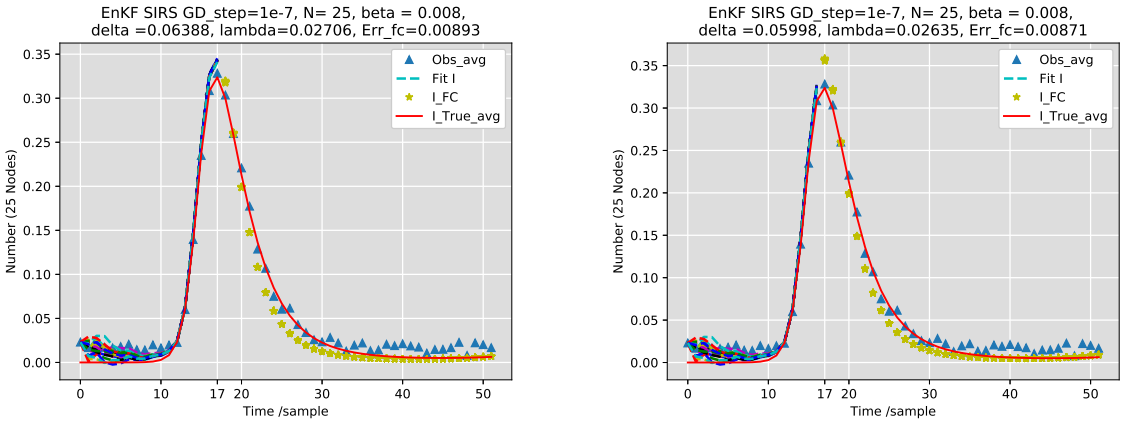
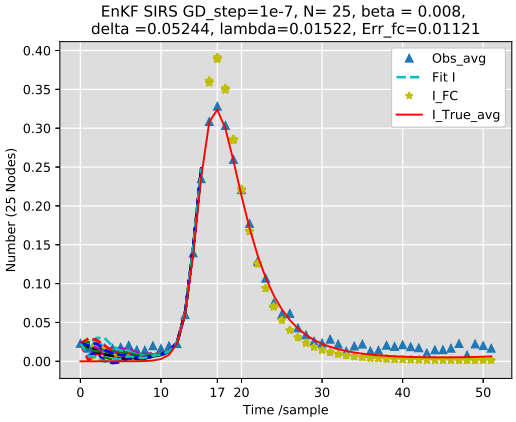


Figure 4.25: Forecasting synthetic epidemic spread on ER $G(10, 12)$ $p = 0.4$, when $\sigma_n = 0.01$, $\eta = 1e^{-7}$, (a) after the out breaker, (b) 1 weeks before the out breaker, (c) 2 weeks before the out breaker, (d) 3 weeks before the out breaker, (e) 4 weeks before the out breaker.

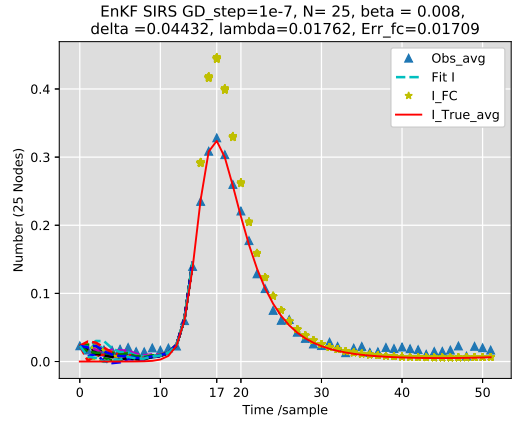


(a)

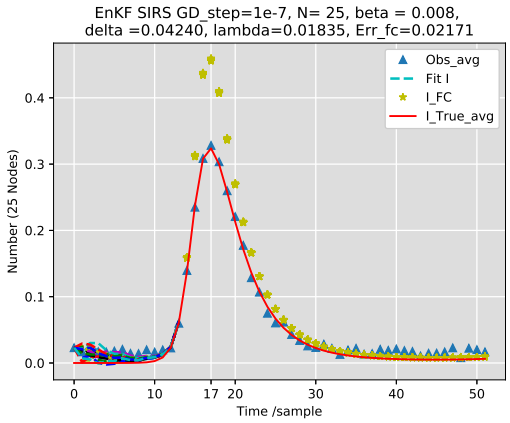
(b)



(c)



(d)



(e)

Figure 4.26: Forecasting synthetic epidemic spread on ER $G(10, 12)$ $p = 0.4$, when $\sigma_n = 0.02$, $\eta = 1e^{-7}$, (a) after the out breaker, (b) 1 weeks before the out breaker, (c) 2 weeks before the out breaker, (d) 3 weeks before the out breaker, (e) 4 weeks before the out breaker.

4.3. Real-world epidemic data

This section shows the real-world epidemic data fitting and forecasting. We present our algorithm results after processing the influenza data. The raw data is collected from the WHO database [42] for the years 2012-2017 for the Netherlands, the UK, Belgium and Germany. In this data, only the observations (number of persons infected by influenza per week) is provided. In the process we consider the Multi-dimensional graph effect algorithm has a complete graph K_{25} and $\beta = 0.004$.

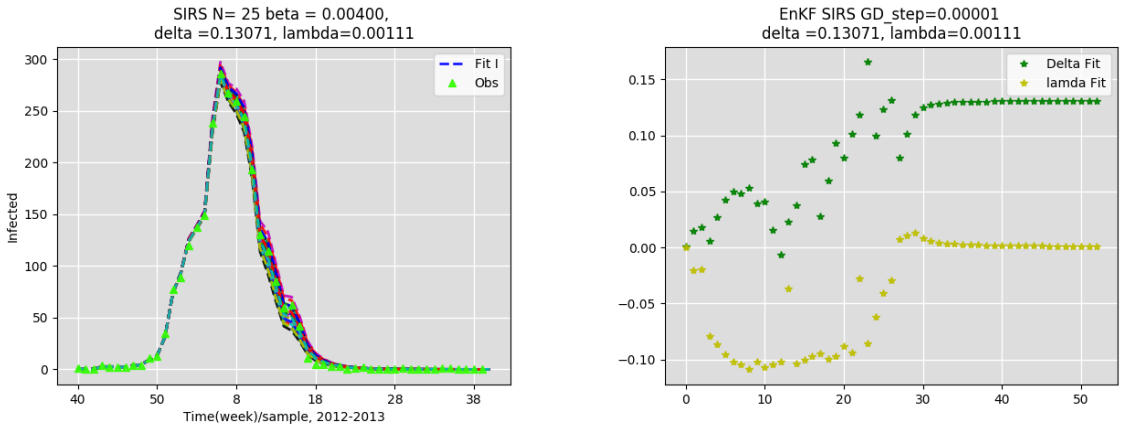
4.3.1. Fitting and estimating

The Figures contain the following:

- The number of infected people (observations O), represented by triangles.
- The fitting infection curve, represented by N dash-line.
- The estimation of δ and λ, represented by green and yellow stars respectively.

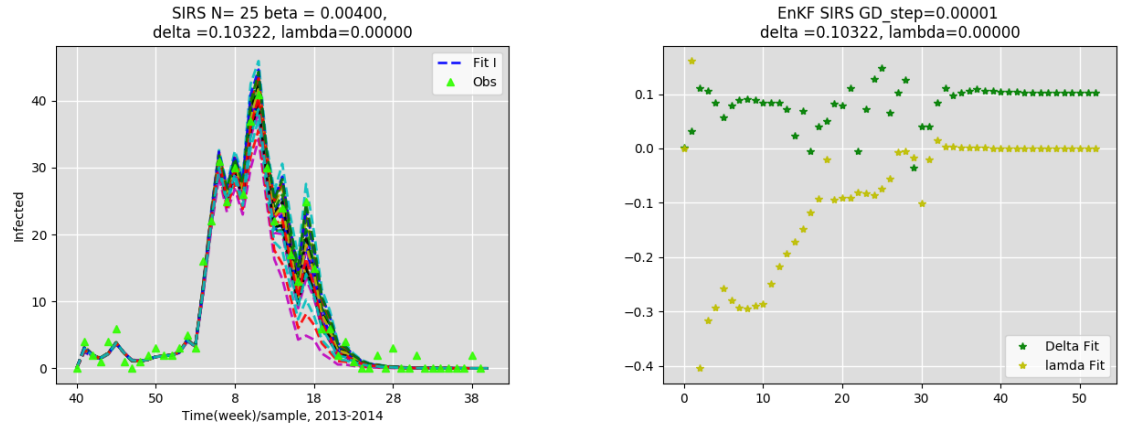
Figures 4.27 - 4.46 show the fitting of O in part (a), also show the δ and λ estimation in part (b). The part (b) of the figures 4.27 - 4.46 shows δ and λ conversion to its desired values. The Figures show that most of the time λ is conversing to zero or very small value which means that the spreading process of each year is following SIR model.

The Netherlands (NL) influenza



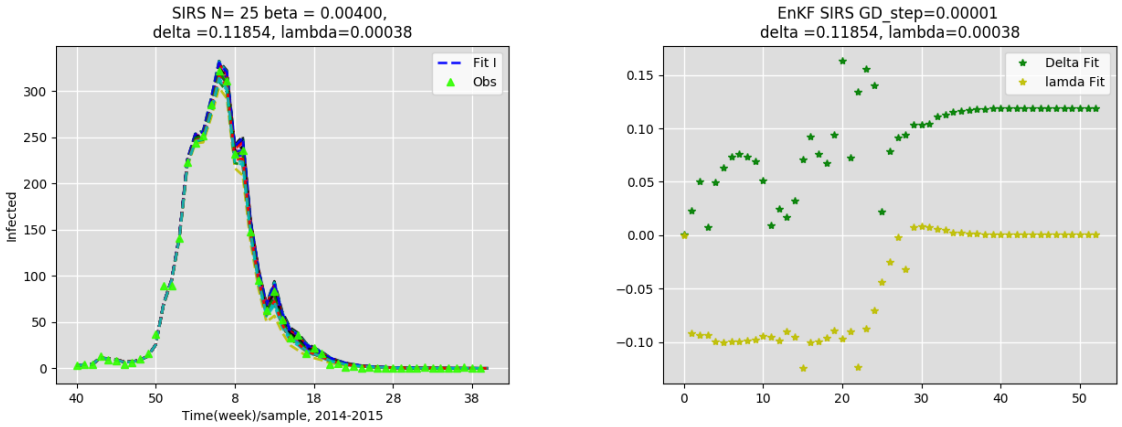
(a) (b)

Figure 4.27: Fitting, NL influenza data 2012-2013 (a) Infection fitting, (b) β and λ estimation



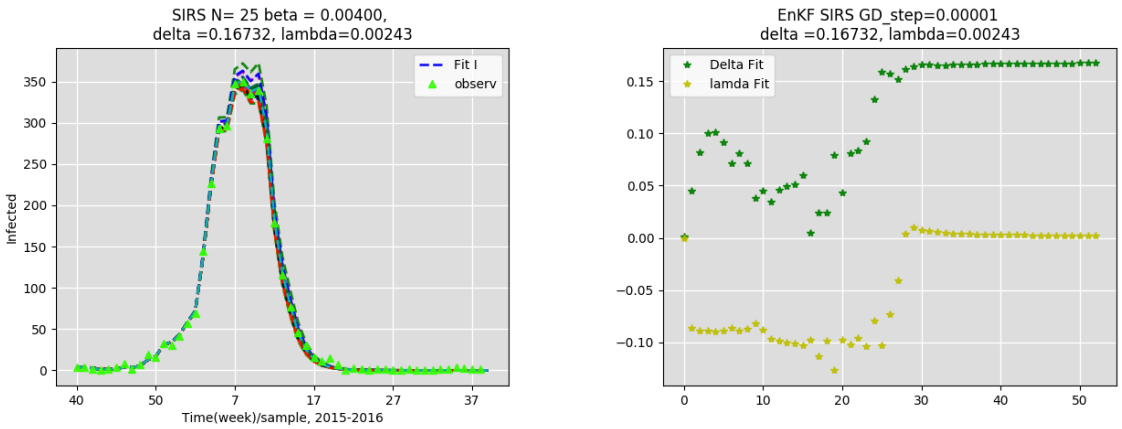
(a) (b)

Figure 4.28: Fitting, NL influenza data 2013-2014 (a) Infection fitting, (b) β and λ estimation



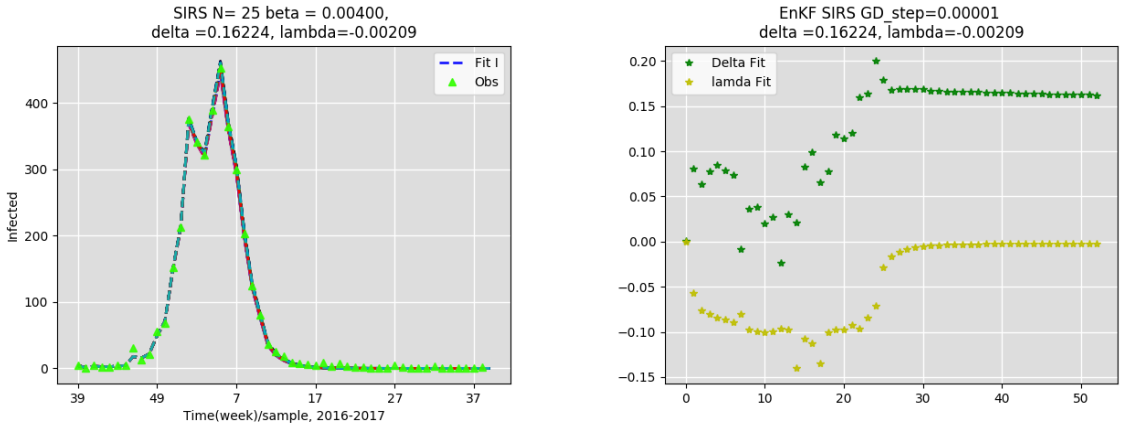
(a) (b)

Figure 4.29: Fitting, NL influenza data 2014-2015 (a) Infection fitting, (b) β and λ estimation



(a) (b)

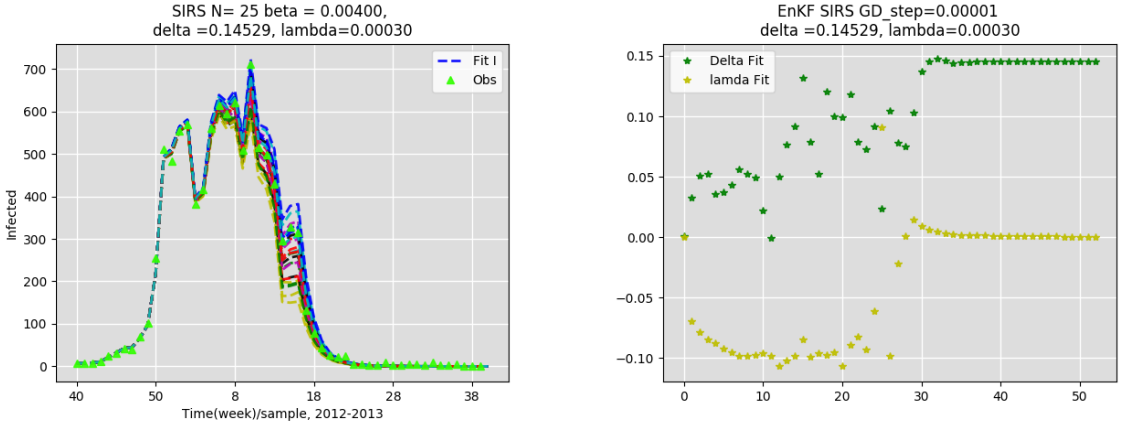
Figure 4.30: Fitting, NL influenza data 2015-2016 (a) Infection fitting, (b) β and λ estimation



(a) (b)

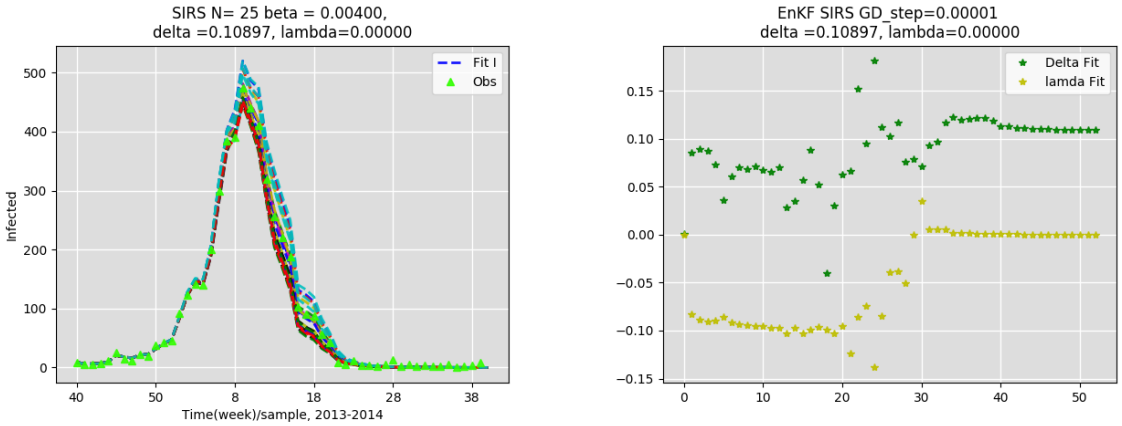
Figure 4.31: Fitting, NL influenza data 2016-2017 (a) Infection fitting, (b) β and λ estimation

The United Kingdom (UK) influenza data



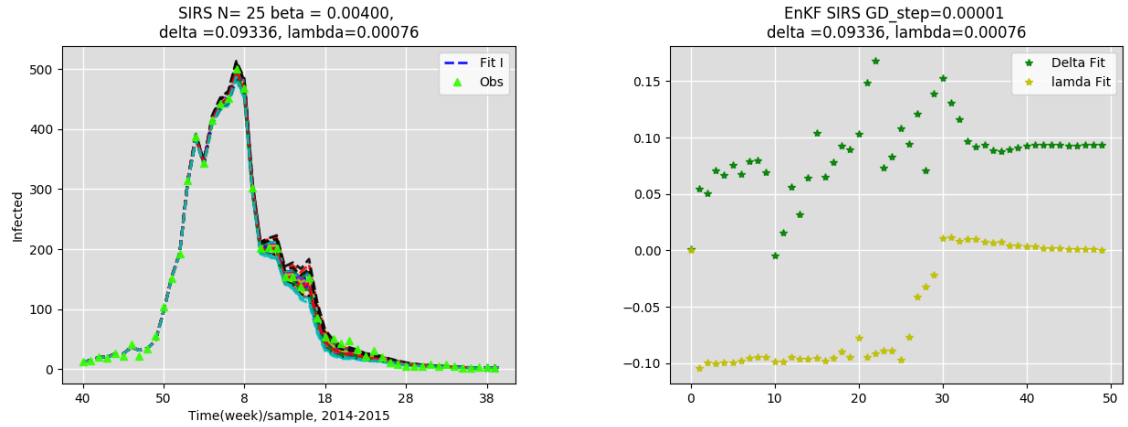
(a) (b)

Figure 4.32: Fitting, the UK influenza data 2012-2013 (a) Infection fitting, (b) β and λ estimation



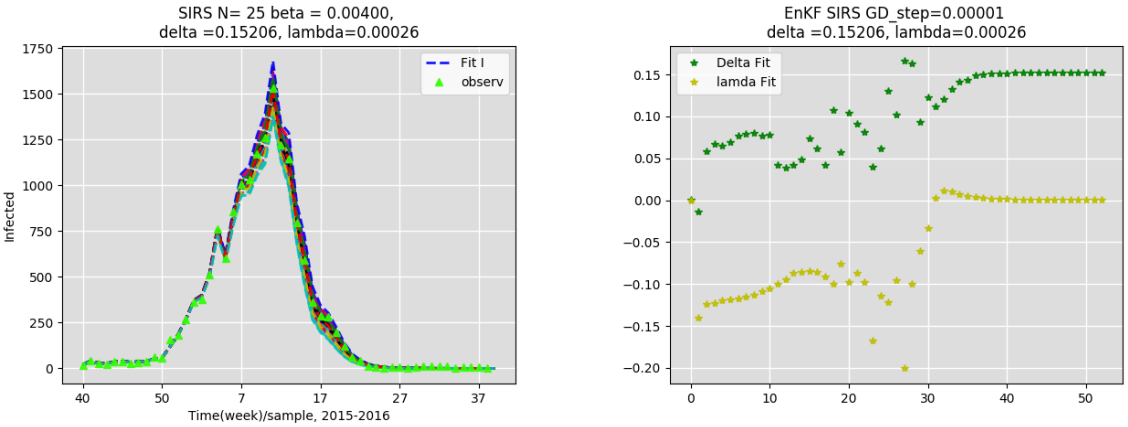
(a) (b)

Figure 4.33: Fitting, the UK influenza data 2013-2014 (a) Infection fitting, (b) β and λ estimation



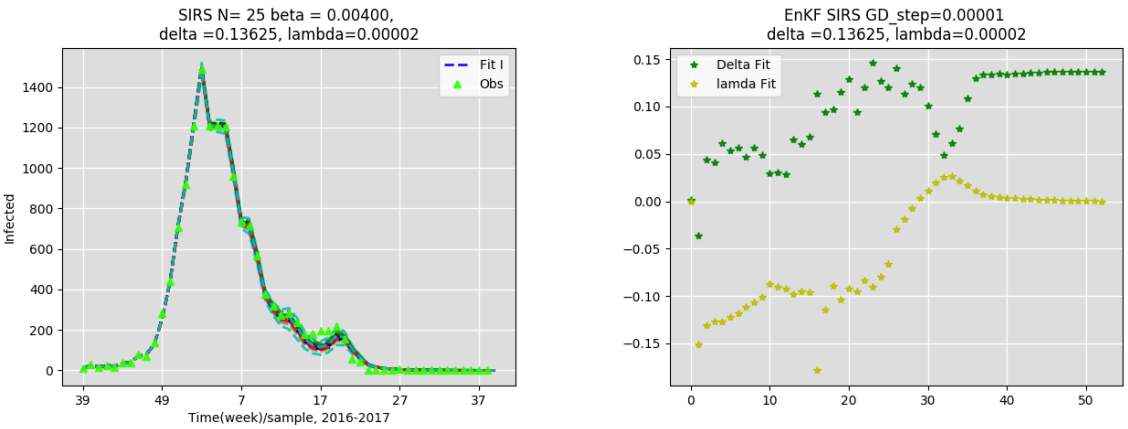
(a) (b)

Figure 4.34: Fitting, the UK influenza data 2014-2015 (a) Infection fitting, (b) β and λ estimation



(a) (b)

Figure 4.35: Fitting, the UK influenza data 2015-2016 (a) Infection fitting, (b) β and λ estimation



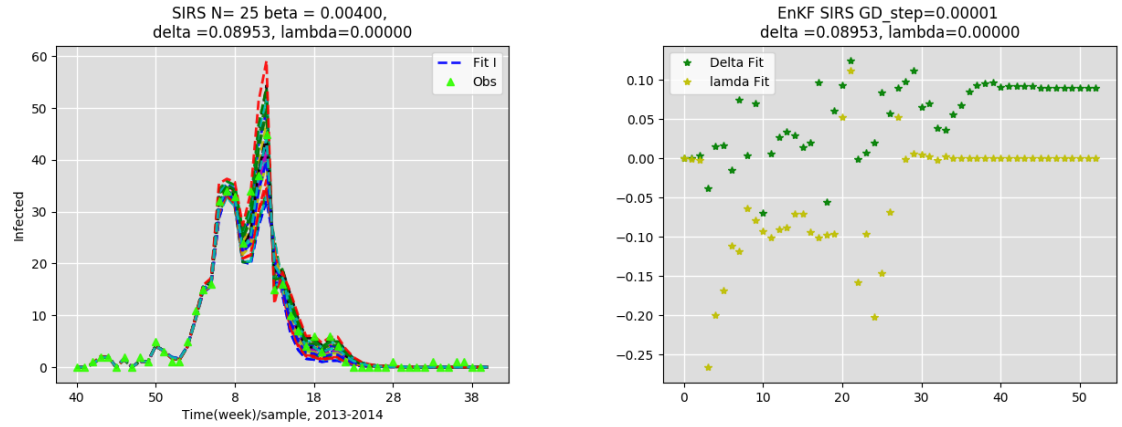
(a) (b)

Figure 4.36: Fitting, the UK influenza data 2016-2017 (a) Infection fitting, (b) β and λ estimation

Germany influenza data

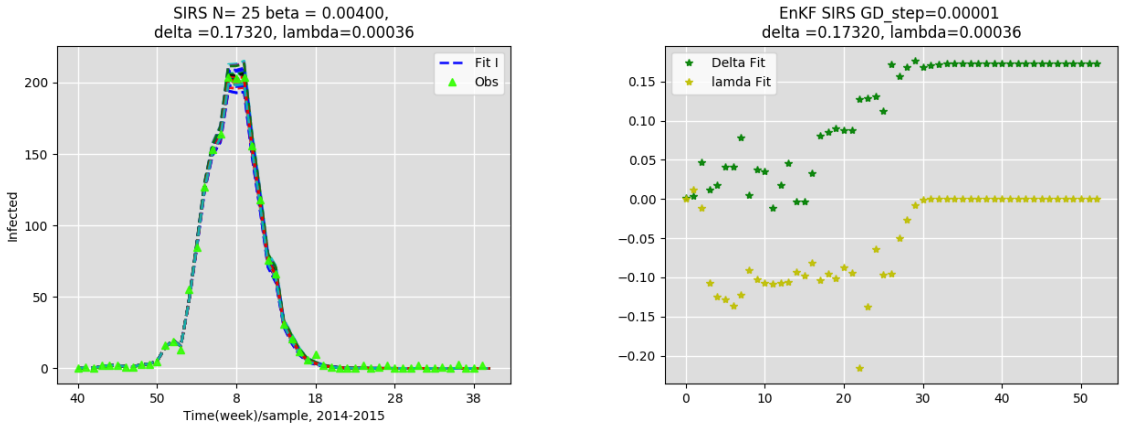
(a)

Figure 4.37: Fitting, Germany influenza data 2012-2013 (a) Infection fitting, (b) β and λ estimation



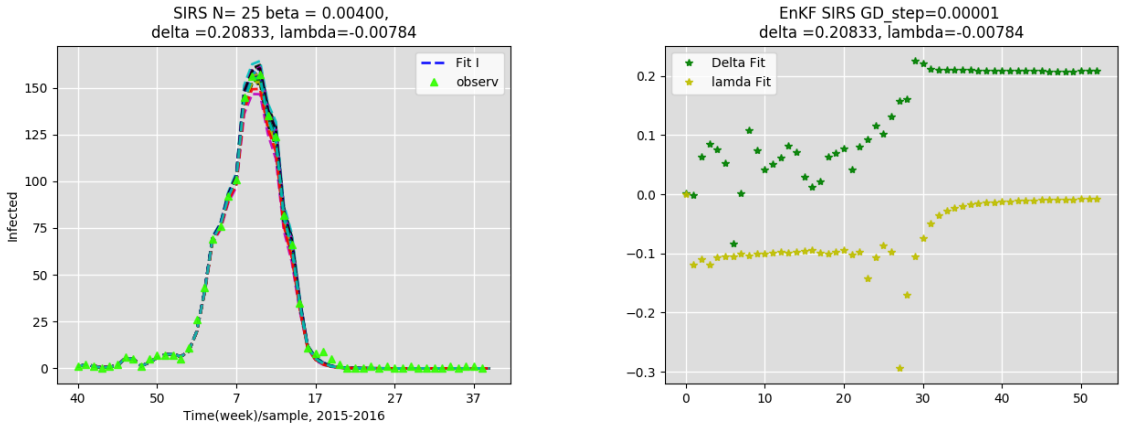
(a)

Figure 4.38: Fitting, Germany influenza data 2013-2014 (a) Infection fitting, (b) β and λ estimation



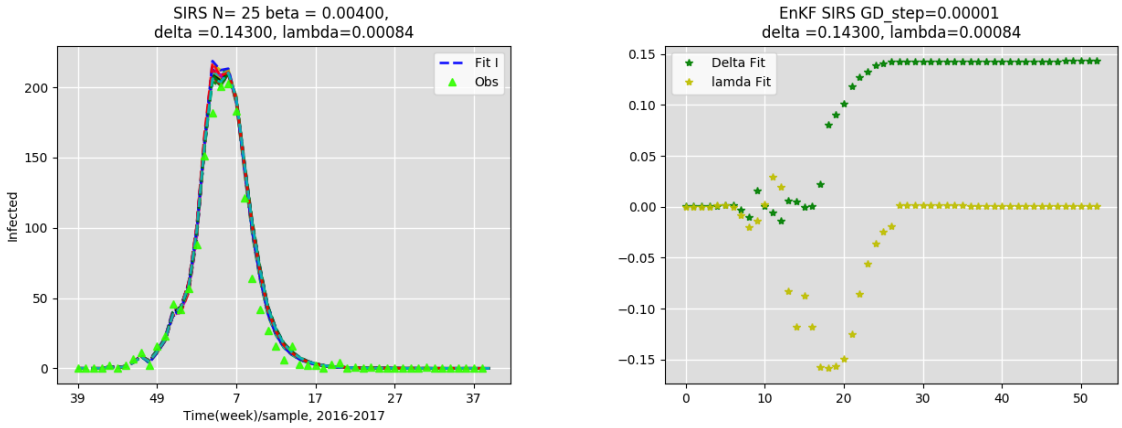
(a)

Figure 4.39: Fitting, Germany influenza data 2014-2015 (a) Infection fitting, (b) β and λ estimation



(a)

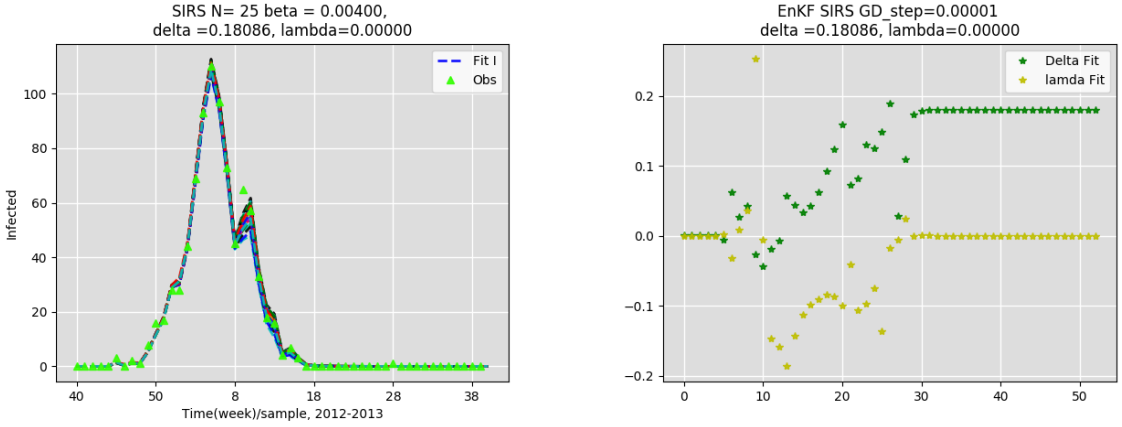
Figure 4.40: Fitting, Germany influenza data 2015-2016 (a) Infection fitting, (b) β and λ estimation



(a)

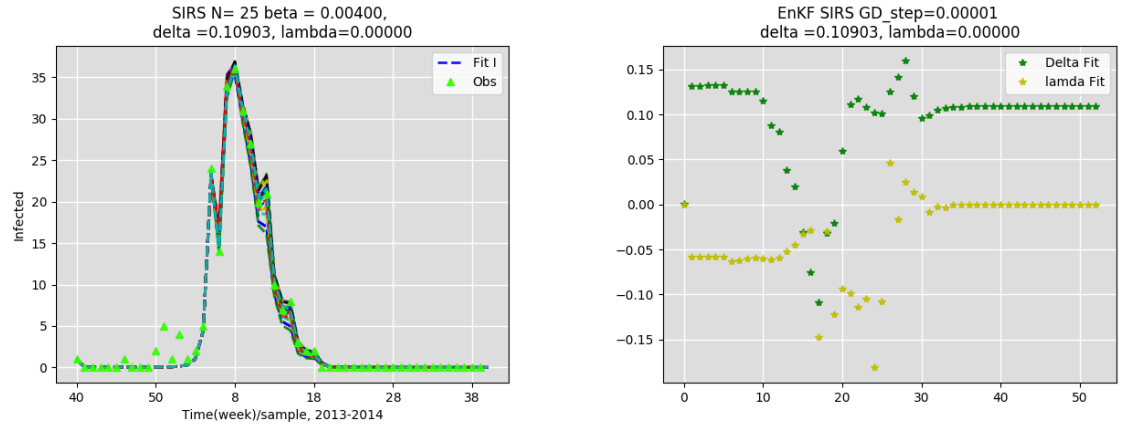
Figure 4.41: Fitting, Germany influenza data 2016-2017 (a) Infection fitting, (b) β and λ estimation

Belgium influenza data



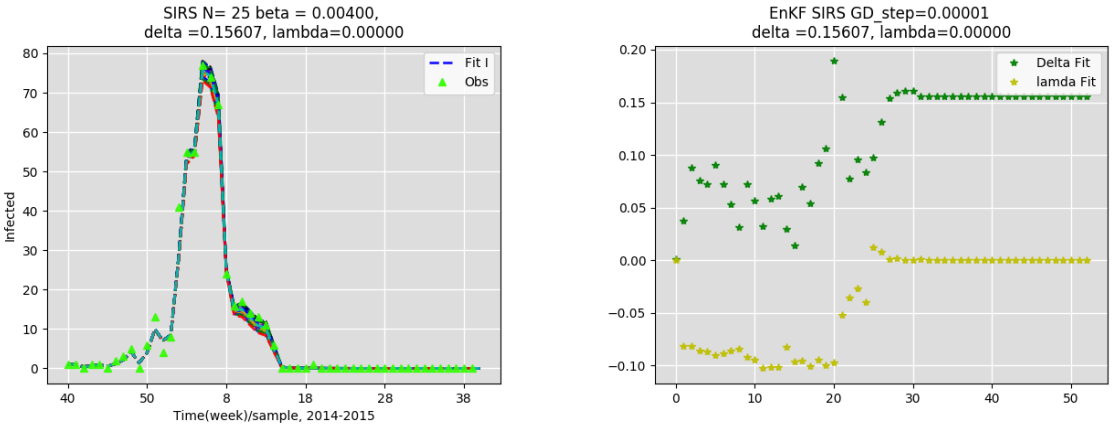
(a) (b)

Figure 4.42: Fitting, Belgium influenza data 2012-2013 (a) Infection fitting, (b) β and λ estimation



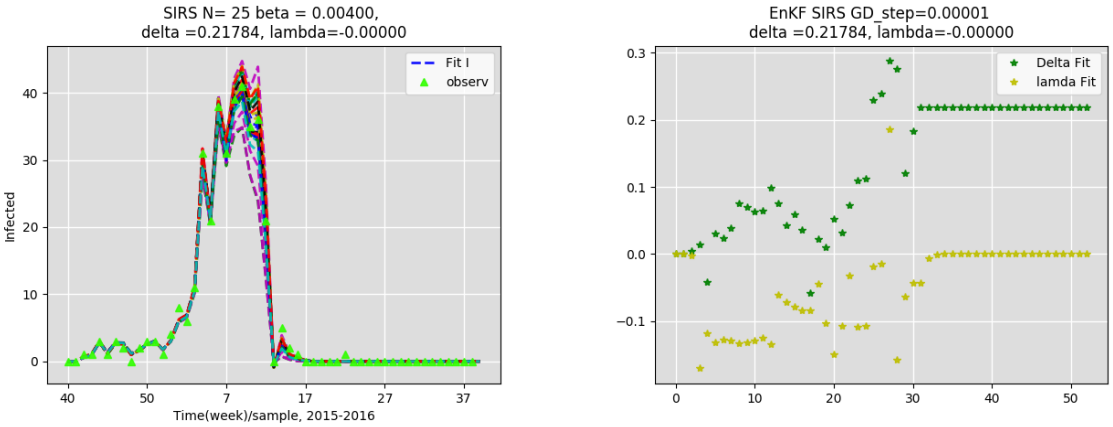
(a) (b)

Figure 4.43: Fitting, Belgium influenza data 2013-2014 (a) Infection fitting, (b) β and λ estimation



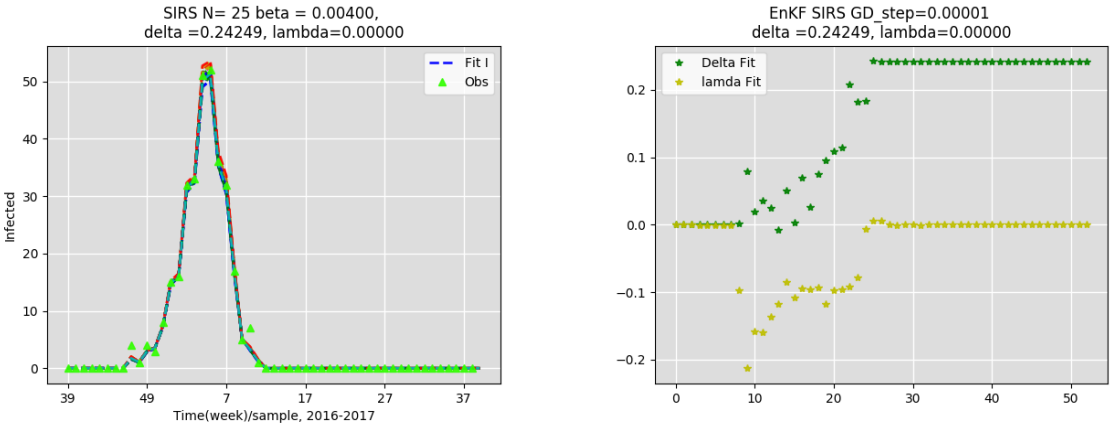
(a) (b)

Figure 4.44: Fitting, Belgium influenza data 2014-2015 (a) Infection fitting, (b) β and λ estimation



(a) (b)

Figure 4.45: Fitting, Belgium influenza data 2015-2016 (a) Infection fitting, (b) β and λ estimation



(a) (b)

Figure 4.46: Fitting, Belgium influenza data 2016-2017 (a) Infection fitting, (b) β and λ estimation

4.3.2. Forecasting

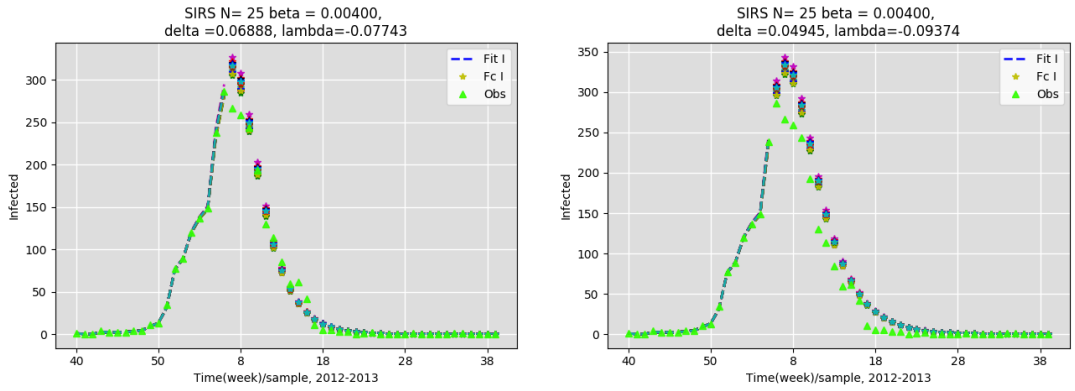
For forecasting, we use the same data used in the fitting (section 4.3.1). The forecasting is done taking 0-4 weeks before the out-breaker. The Figures contain the following:

- The number of infected people (observations O), represented by triangles.
- The fitting/estimating infection curve, represented by N dash-line.
- The forecasting infection curve, represented by stars showing N constricted forecasting curves. In this master thesis, we intended to show all forecasting curves.

Figures 4.47 - 4.66 show the forecasting resulting comparing with real observations values. In general, the hard and aggressive change in the observation slope affects the forecast process and result. As in Figure 4.47d the forecast curve and out-breaker is fine. However, the next observation comes (fig. 4.47c) changing the curve slope which makes the forecasting algorithm deviate from the correct values. Then the next observation comes (fig. 4.47b) aggressively changing the slope making the forecasting algorithm predict higher values. Multiple changing in in the observations up and down (e.g. figures 4.48 and 4.51) effect the forecasting performance especially when two or more observations come with negative slope (fig. 4.51c), then the next observation come with positive slope (fig. 4.51b). Figure 4.51d shows that the algorithm is able to detect one observation with negative slope and it does not affect the output tremendously. However, when two observations in a raw come with negative slope (fig. 4.51c), the algorithm will response and change its slope to follow the observations.

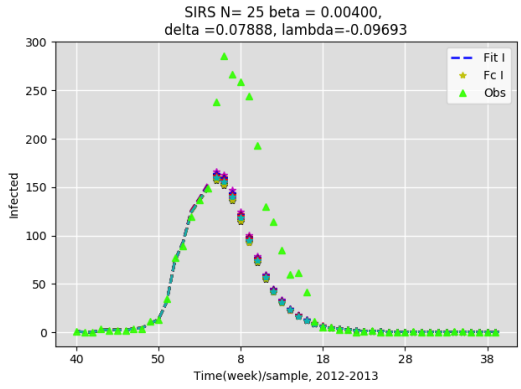
The figures of forecasting real world epidemic data

The Netherlands influenza data 2012-2013

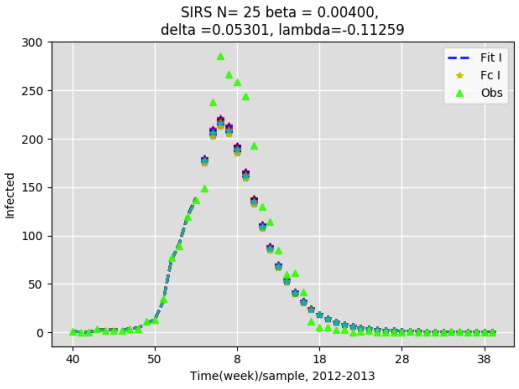


(a)

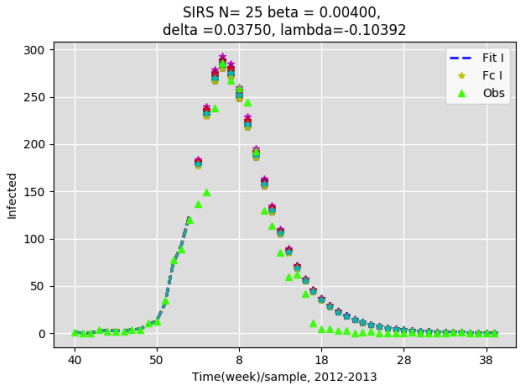
(b)



(c)



(d)



(e)

Figure 4.47: Forecasting, the Netherlands influenza data 2012-2013, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The Netherlands influenza data 2013-2014

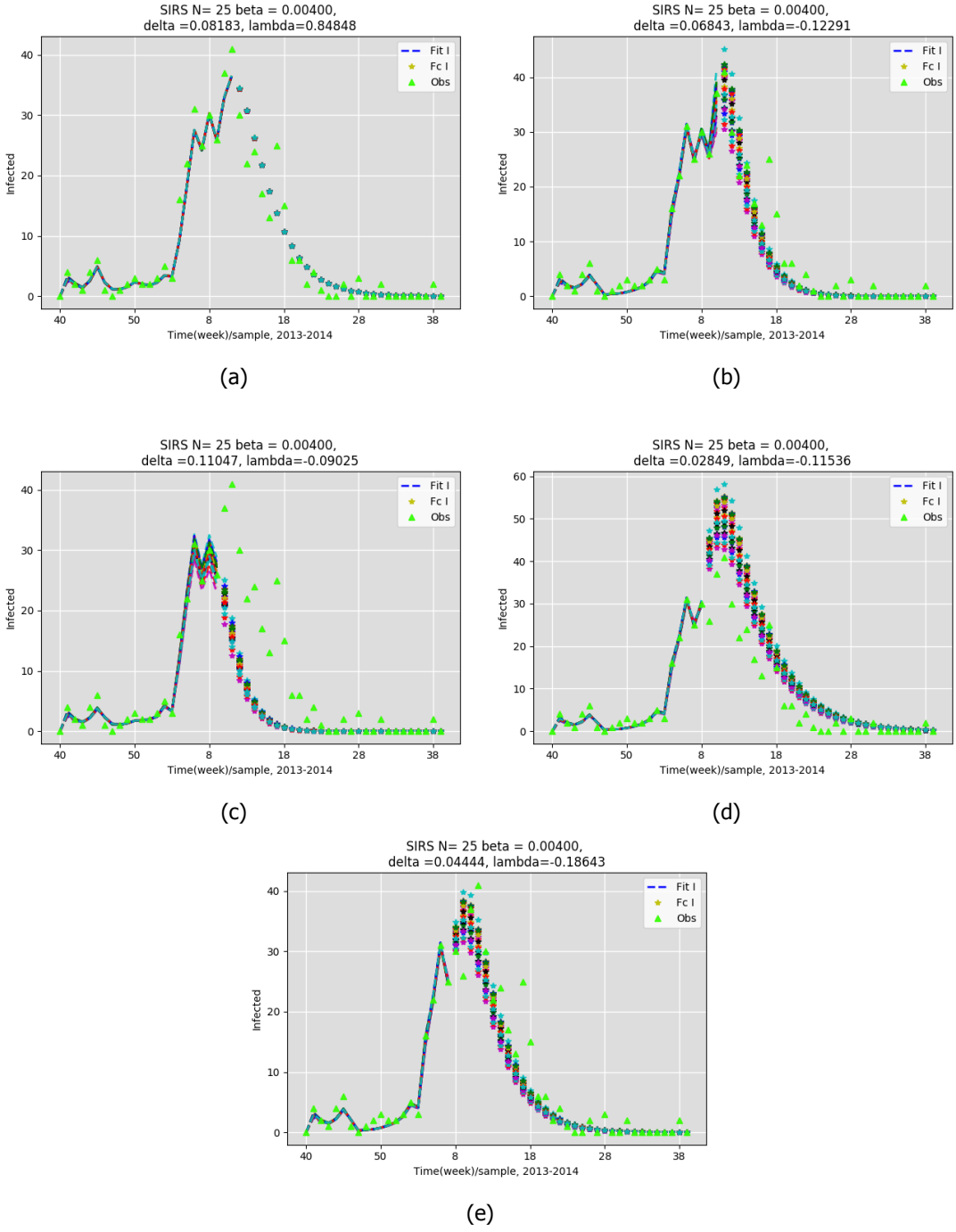


Figure 4.48: Forecasting, the Netherlands influenza data 2013-2014, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The Netherlands influenza data 2014-2015

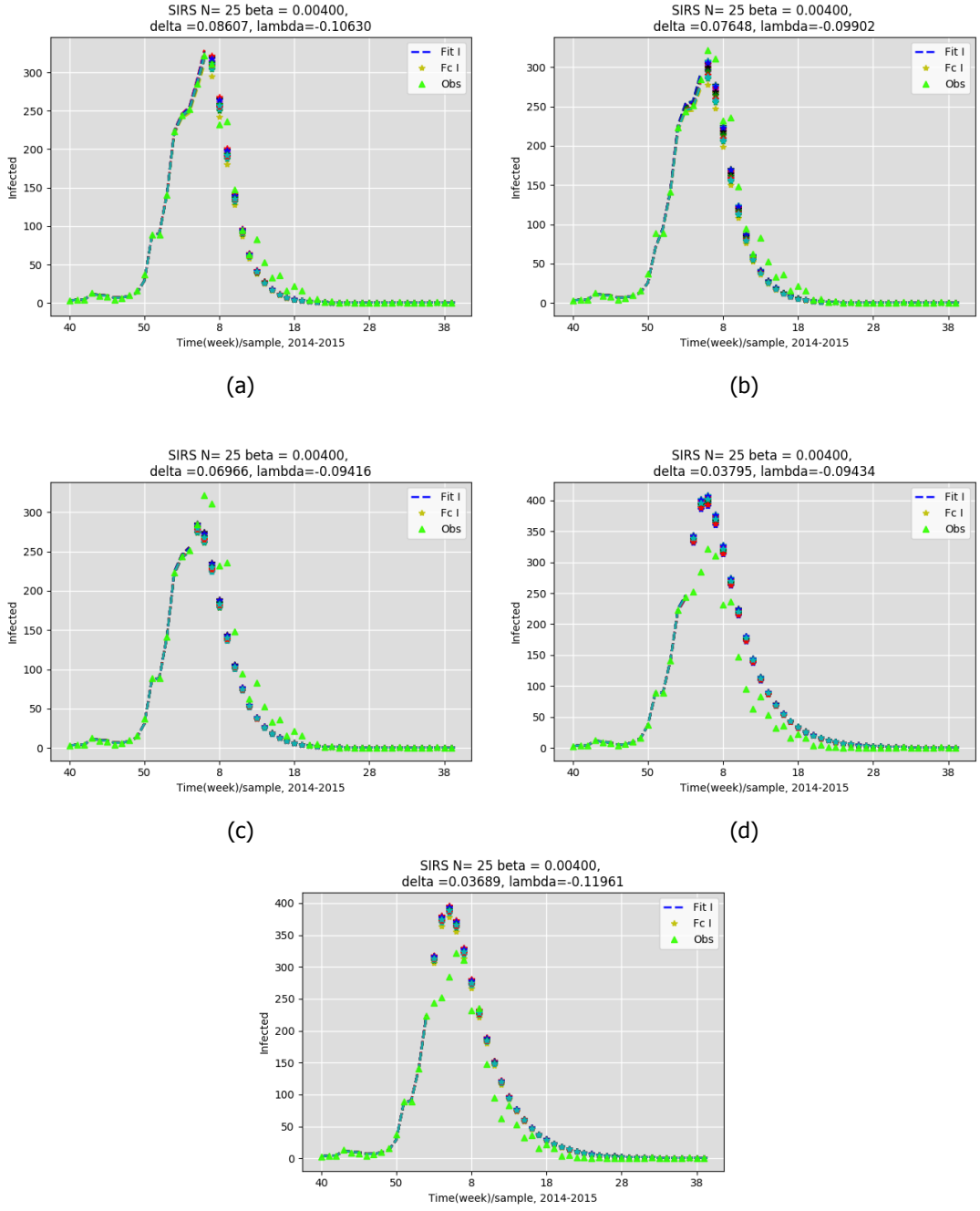


Figure 4.49: Forecasting, the Netherlands influenza data 2014-2015, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The Netherlands influenza data 2015-2016

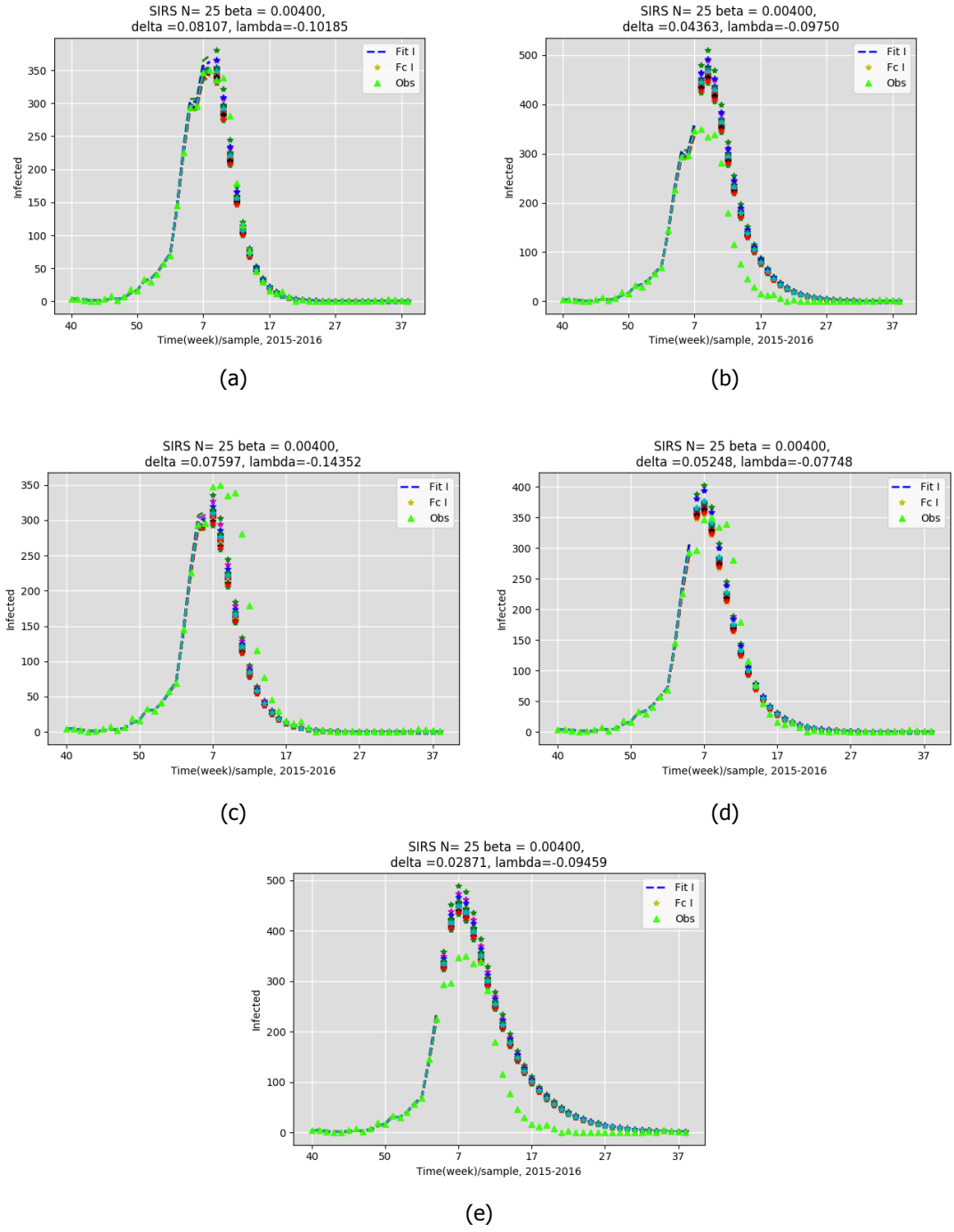


Figure 4.50: Forecasting, the Netherlands influenza data 2015-2016, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The Netherlands influenza data 2016-2017

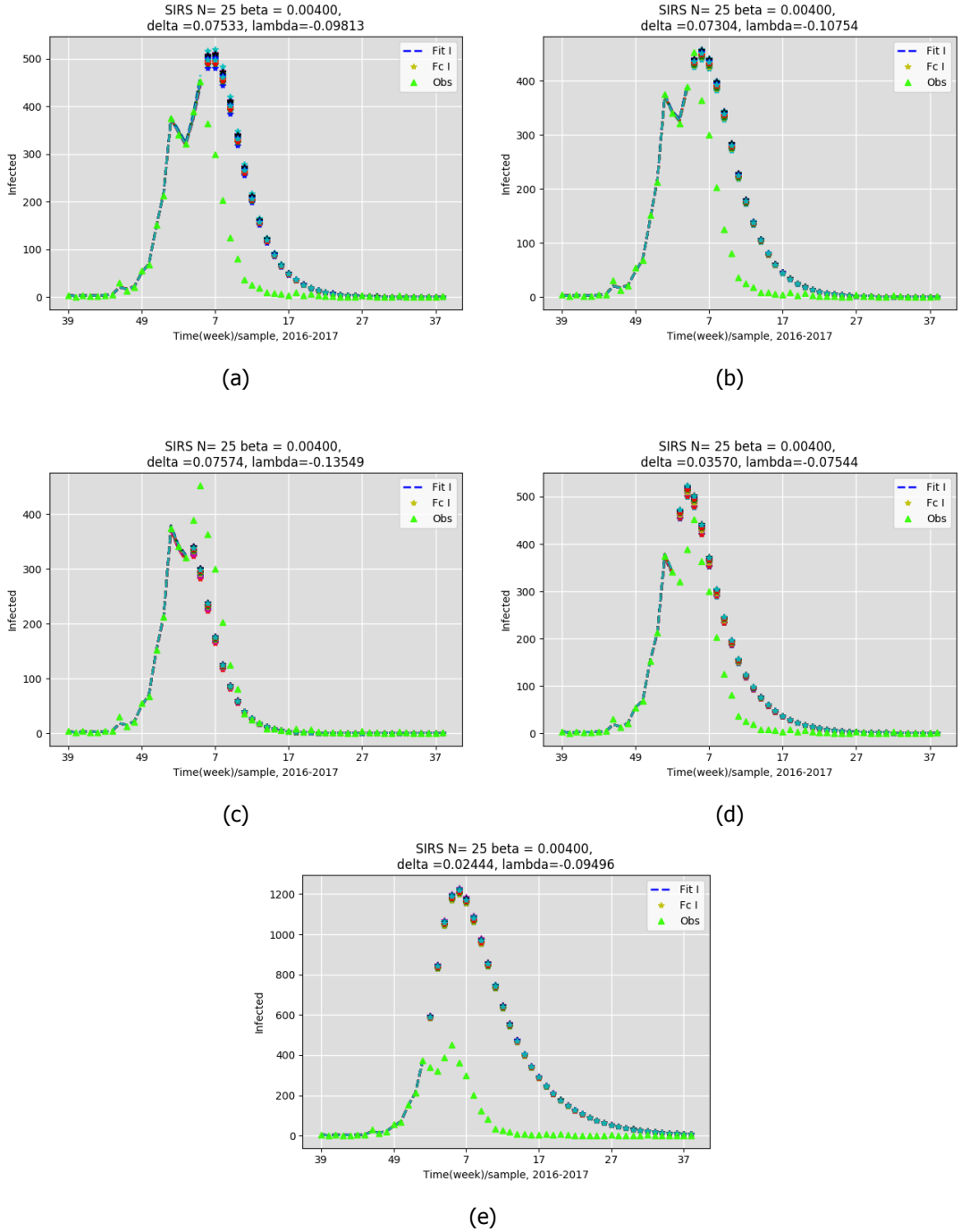


Figure 4.51: Forecasting, the Netherlands influenza data 2016-2017, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The UK's influenza data 2012-2013

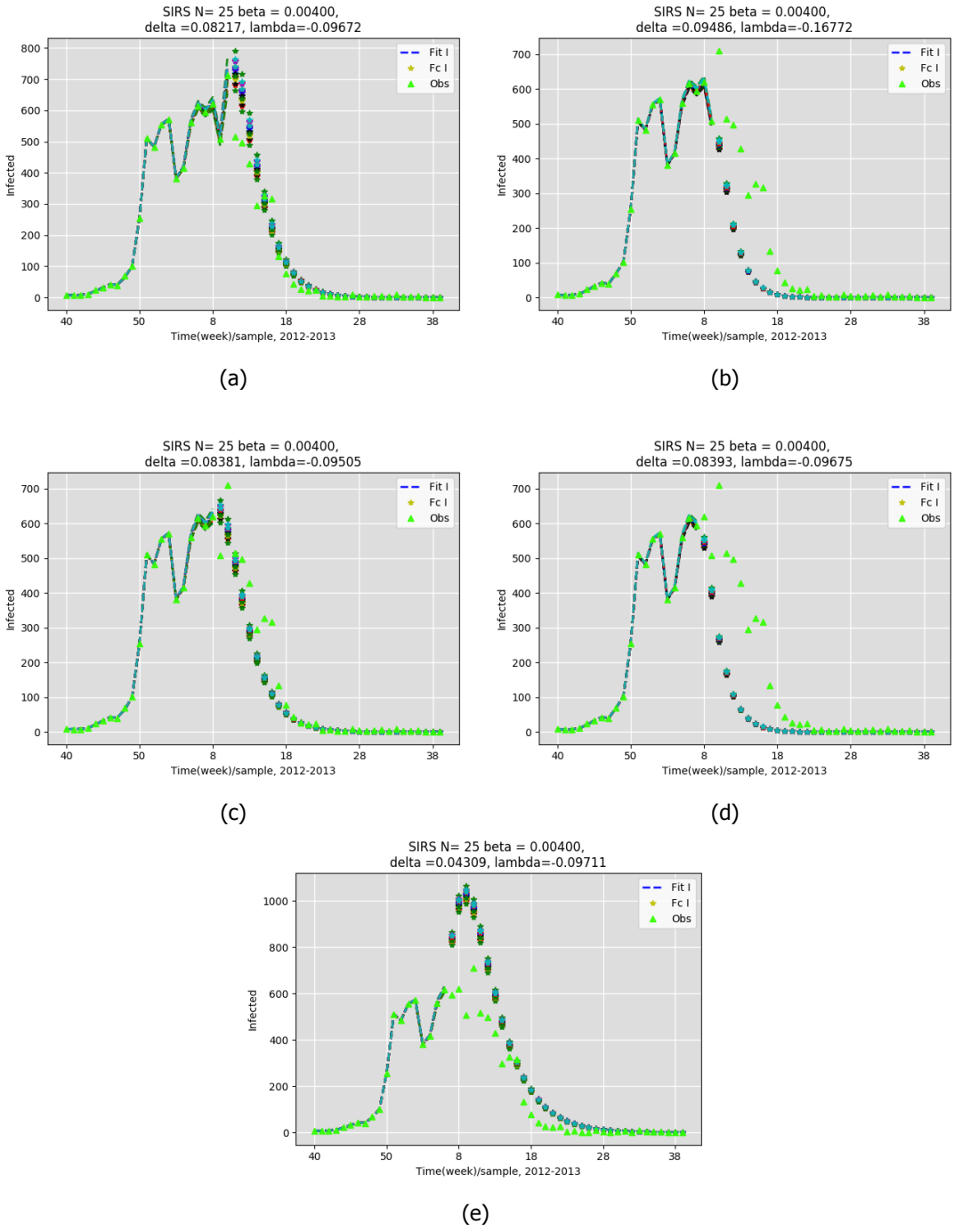


Figure 4.52: Forecasting, The UK's influenza data 2012-2013, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The UK's influenza data 2013-2014

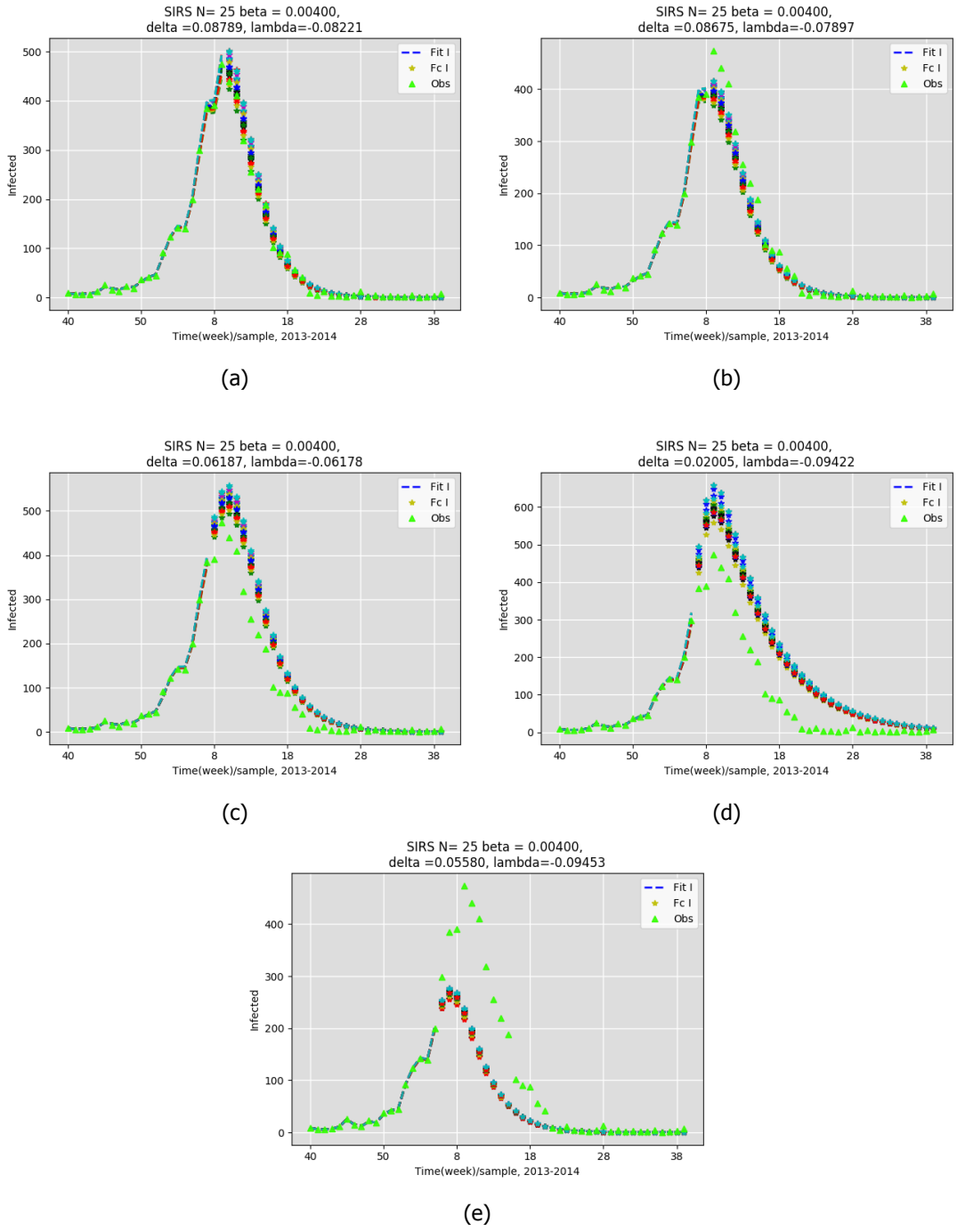


Figure 4.53: Forecasting, the UK's influenza data 2013-2014, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The UK's influenza data 2014-2015

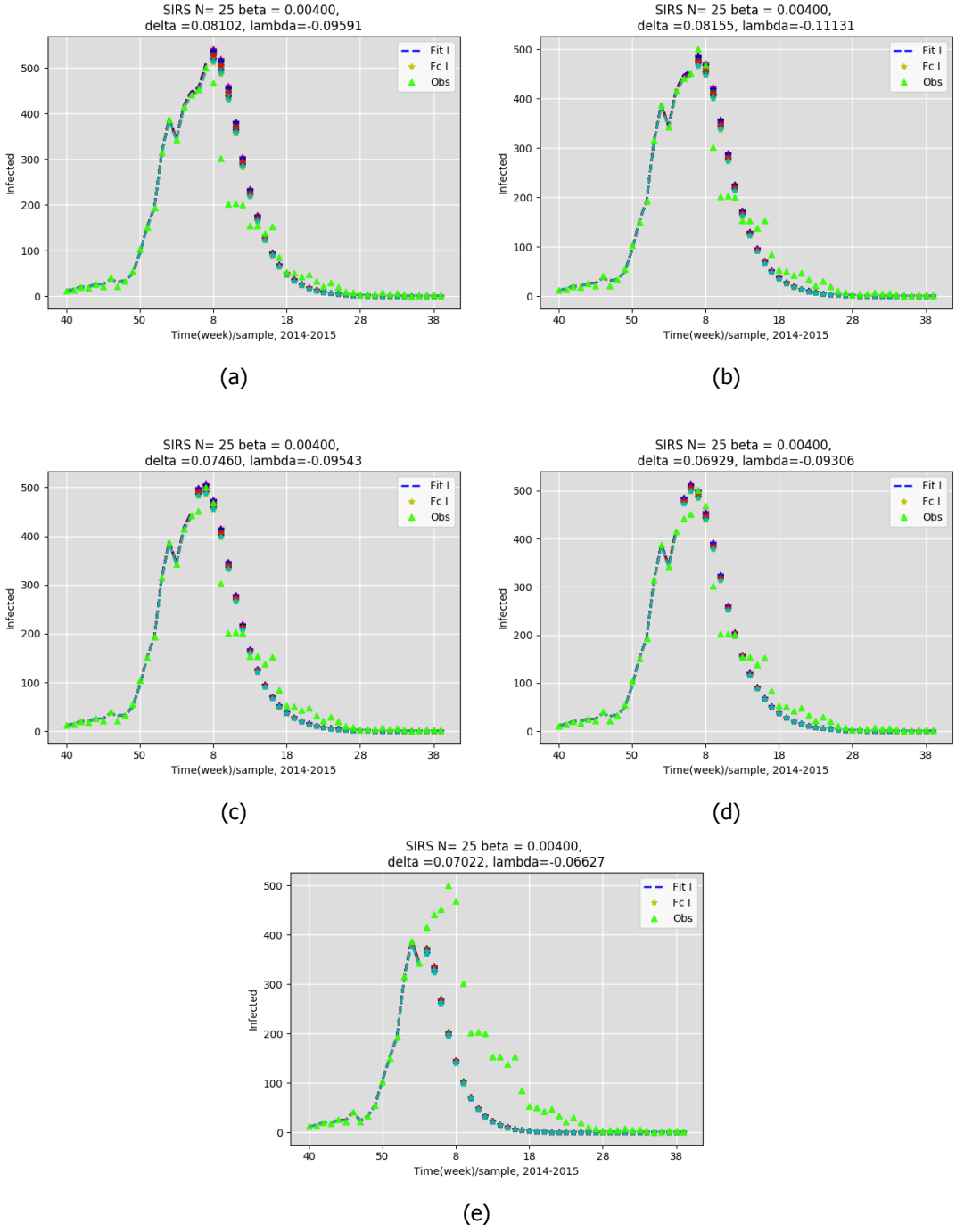


Figure 4.54: Forecasting, the UK's influenza data 2014-2015, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The UK's influenza data 2015-2016

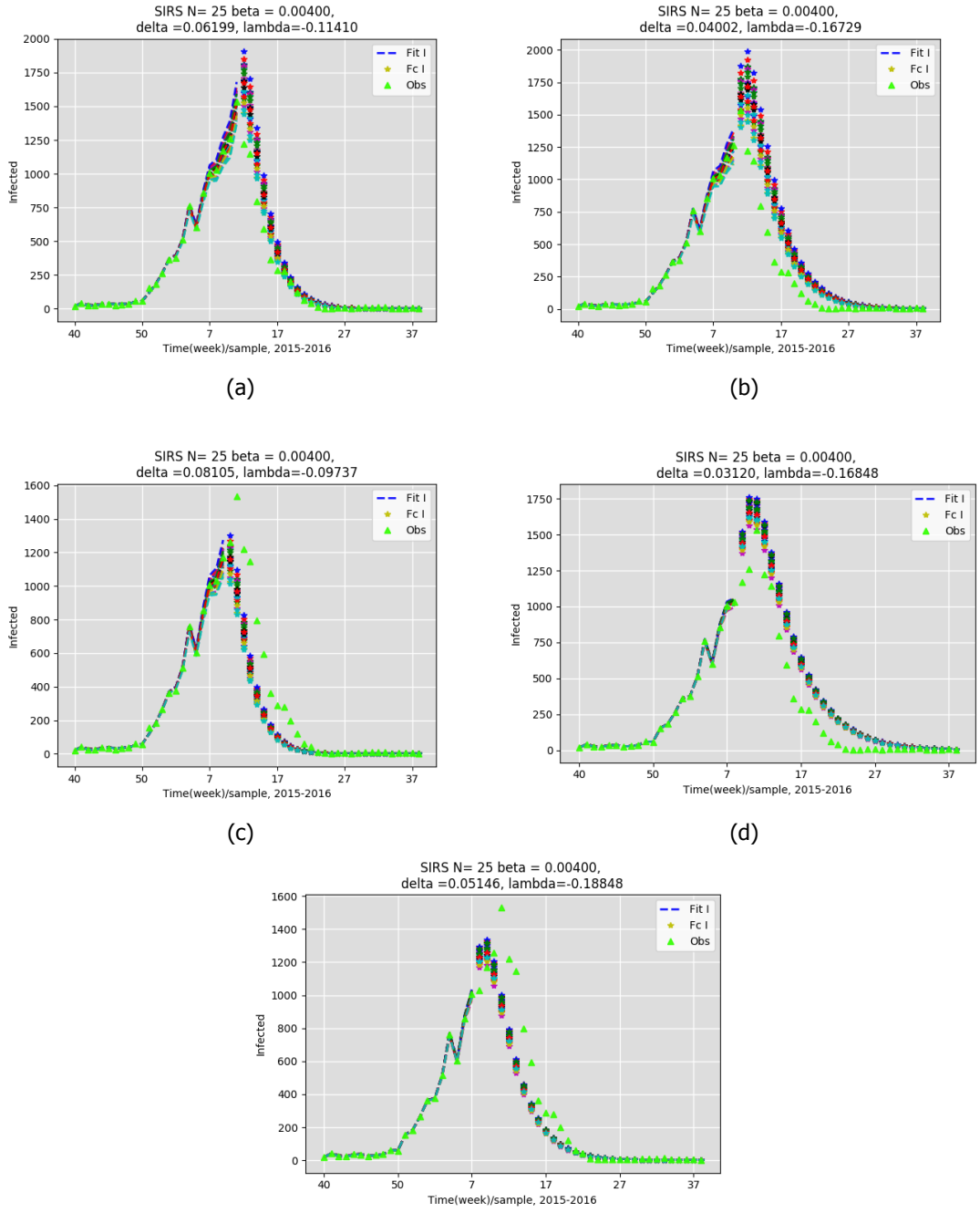


Figure 4.55: Forecasting, the UK's influenza data 2015-2016, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

The UK's influenza data 2016-2017

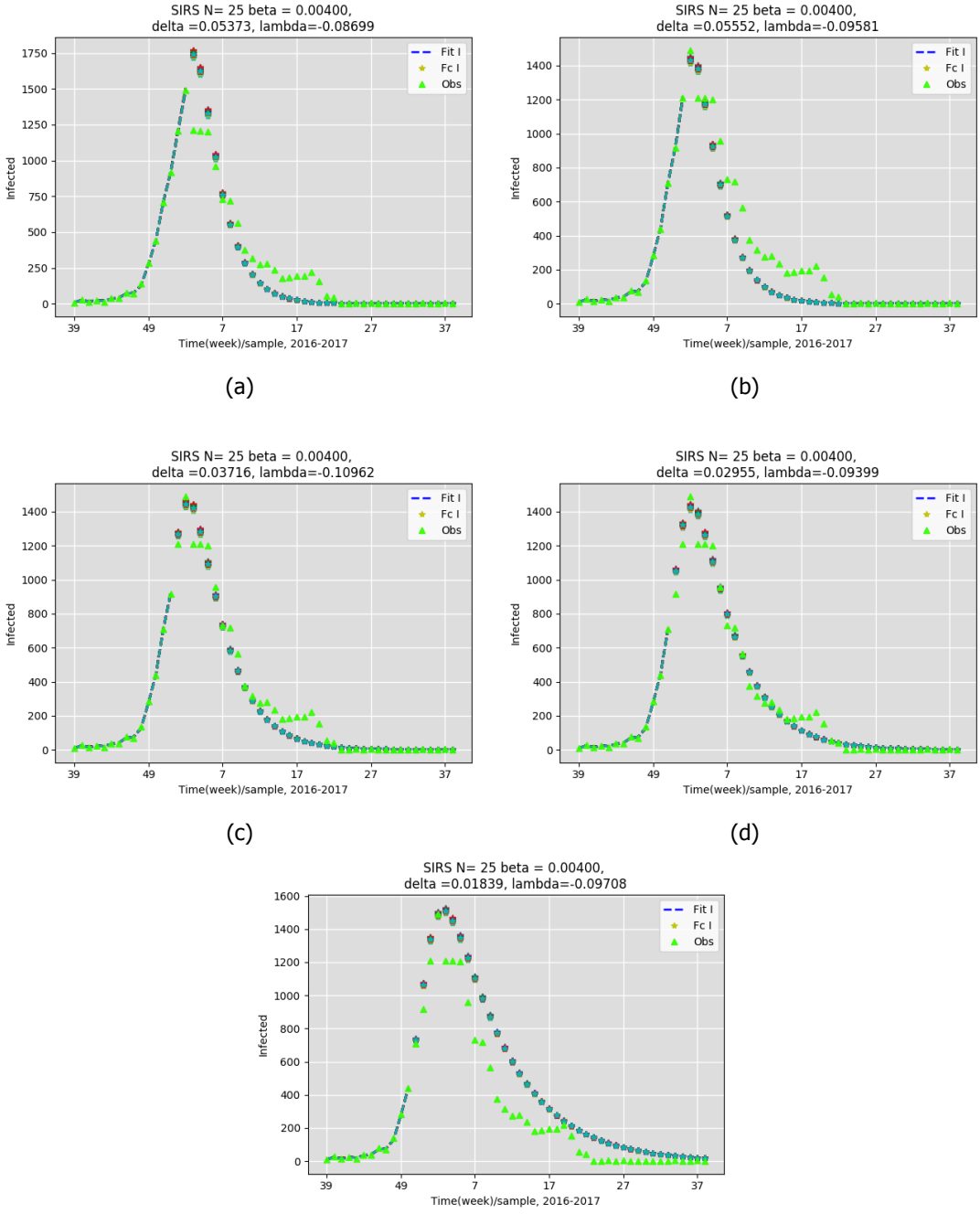


Figure 4.56: Forecasting, the UK's influenza data 2016-2017, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

Germany's influenza data 2012-2013

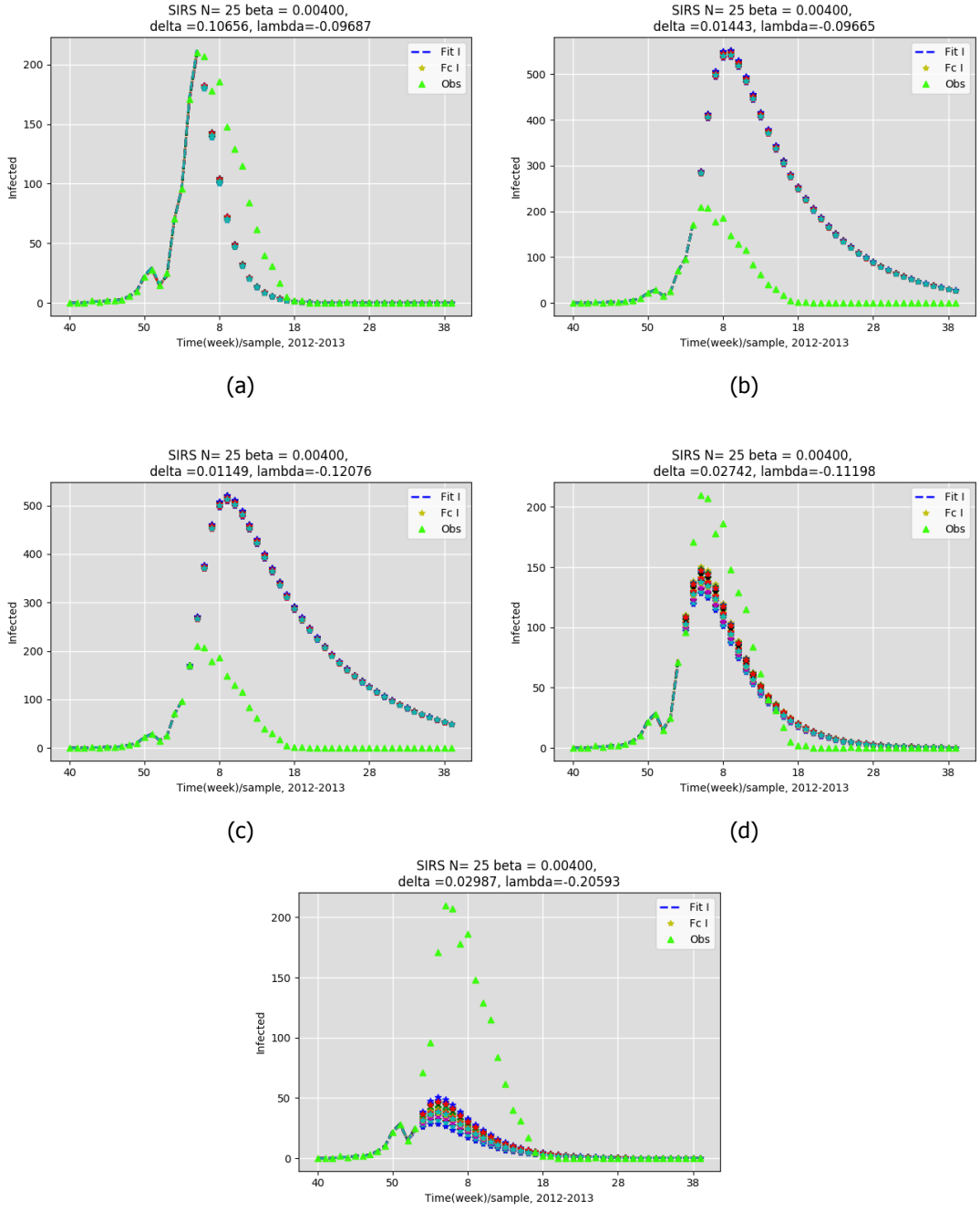


Figure 4.57: Forecasting, Germany's influenza data 2012-2013, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

Germany's influenza data 2013-2014

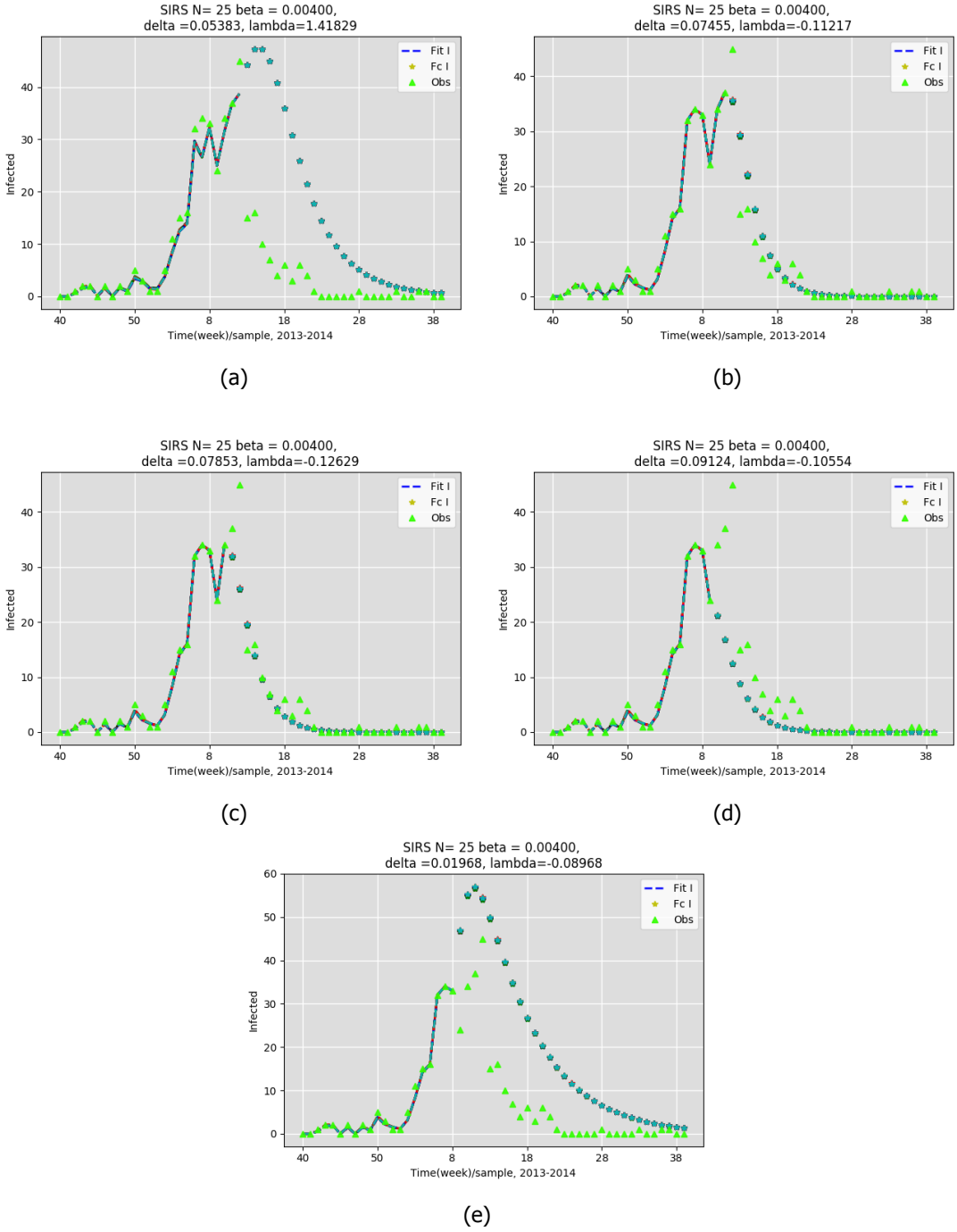


Figure 4.58: Forecasting, Germany's influenza data 2013-2014, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

Germany's influenza data 2014-2015

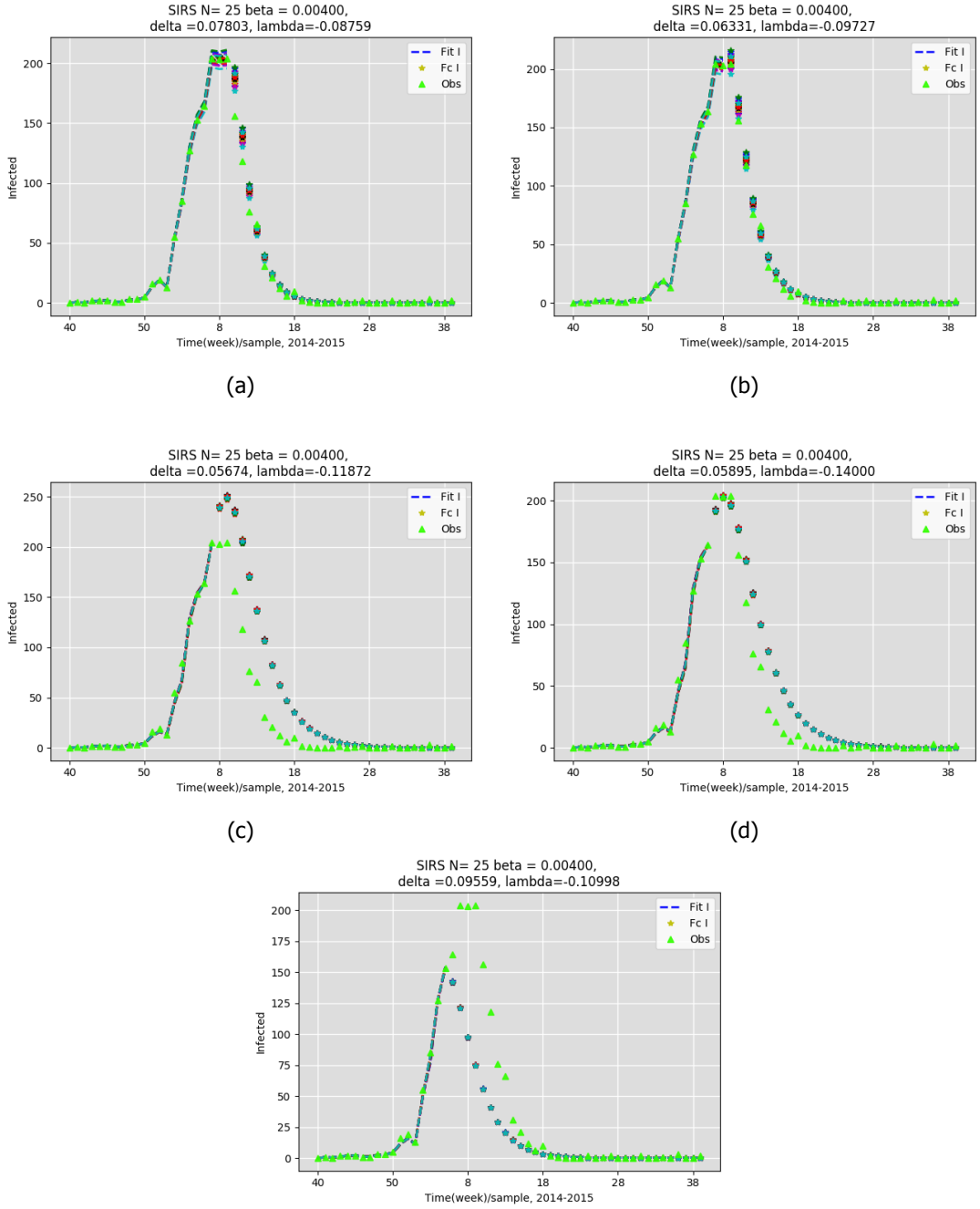


Figure 4.59: Forecasting, Germany's influenza data 2014-2015, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

Germany's influenza data 2015-2016

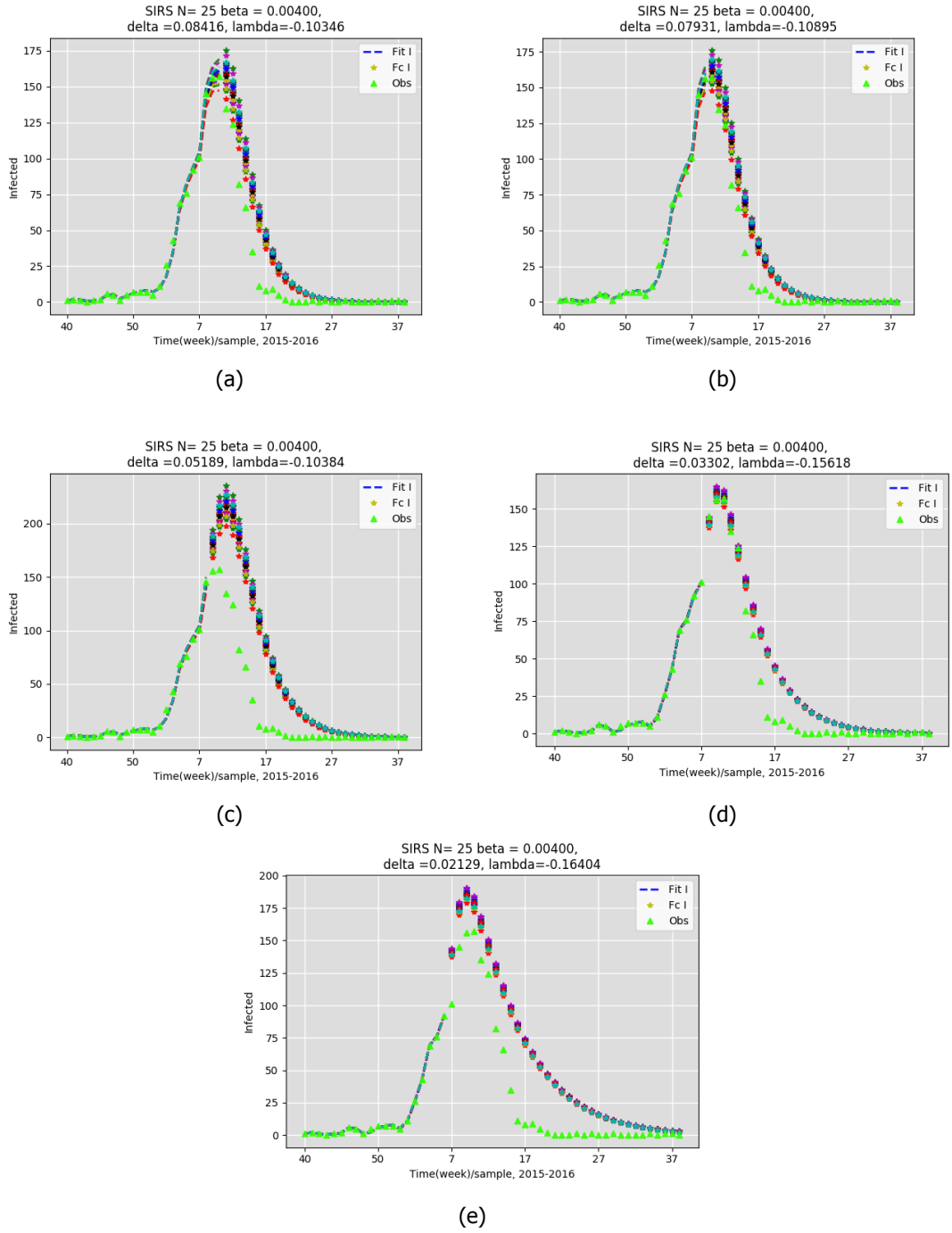


Figure 4.60: Forecasting, Germany's influenza data 2015-2016, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

Germany's influenza data 2016-2017

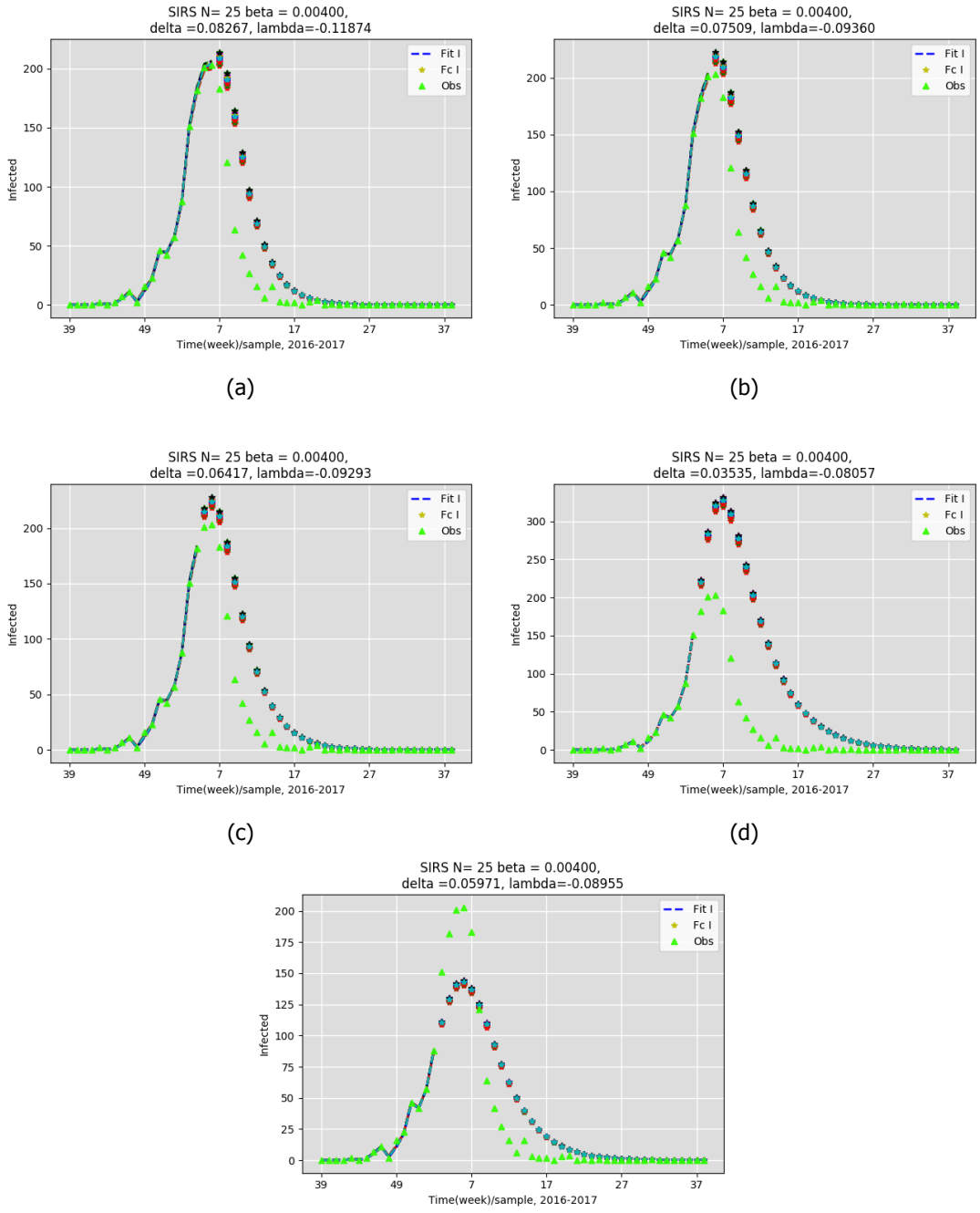


Figure 4.61: Forecasting, Germany's influenza data 2016-2017, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

Belgium's influenza data 2012-2013

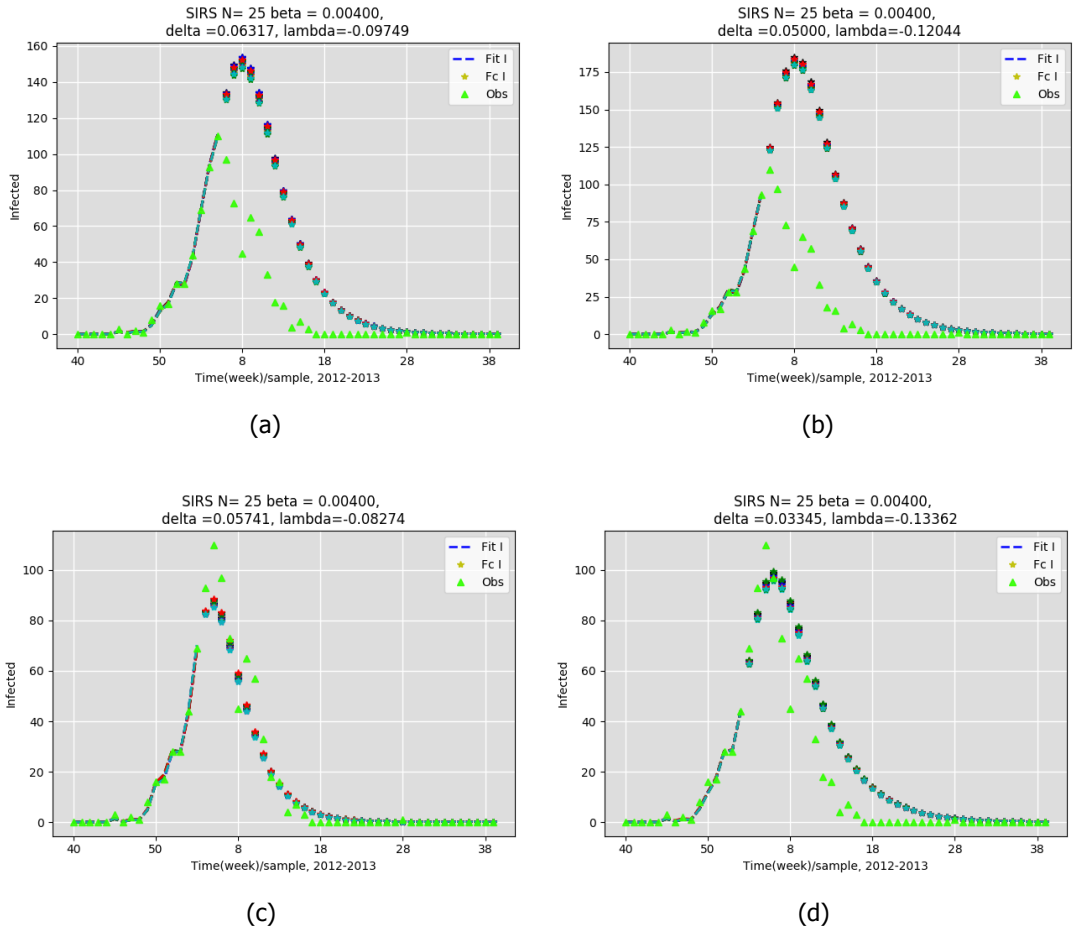


Figure 4.62: Forecasting, Belgium's influenza data 2012-2013, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker,

Belgium's influenza data 2013-2014

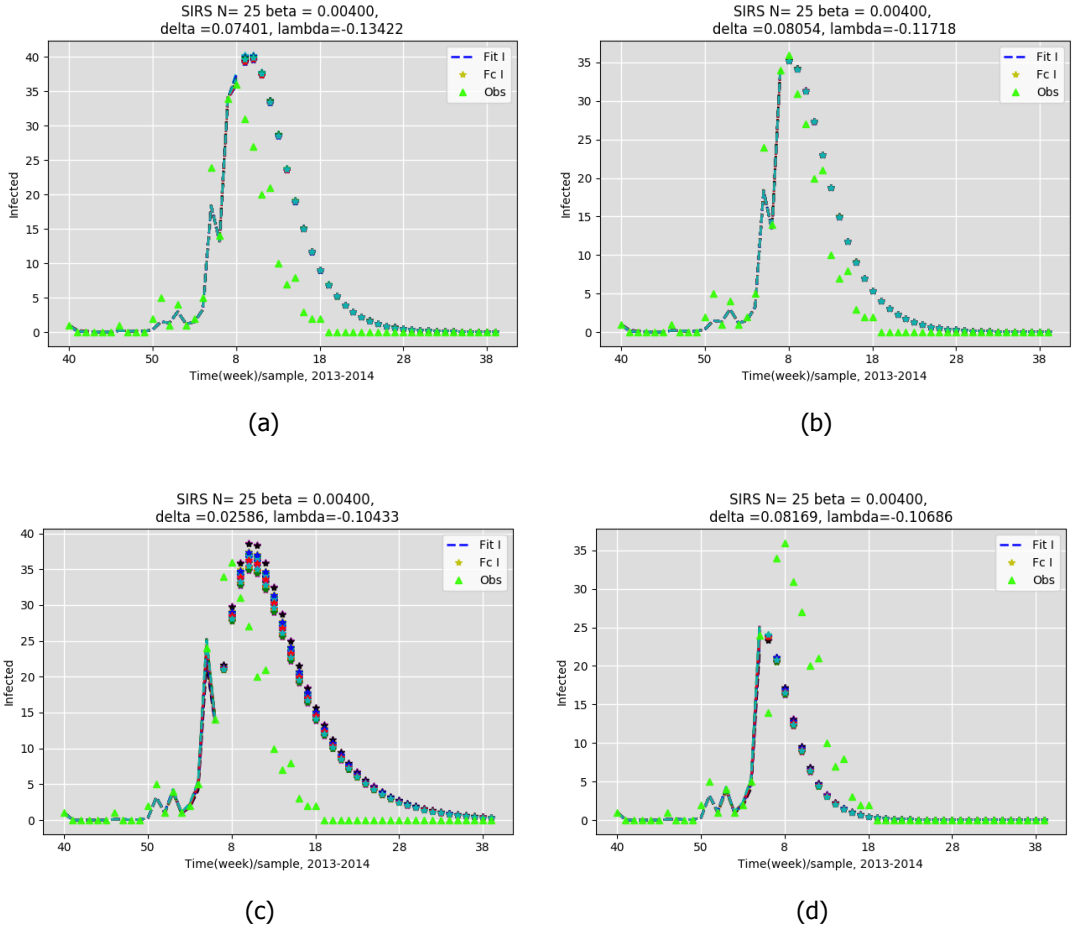


Figure 4.63: Forecasting, Belgium's influenza data 2013-2014, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker,

Belgium's influenza data 2014-2015

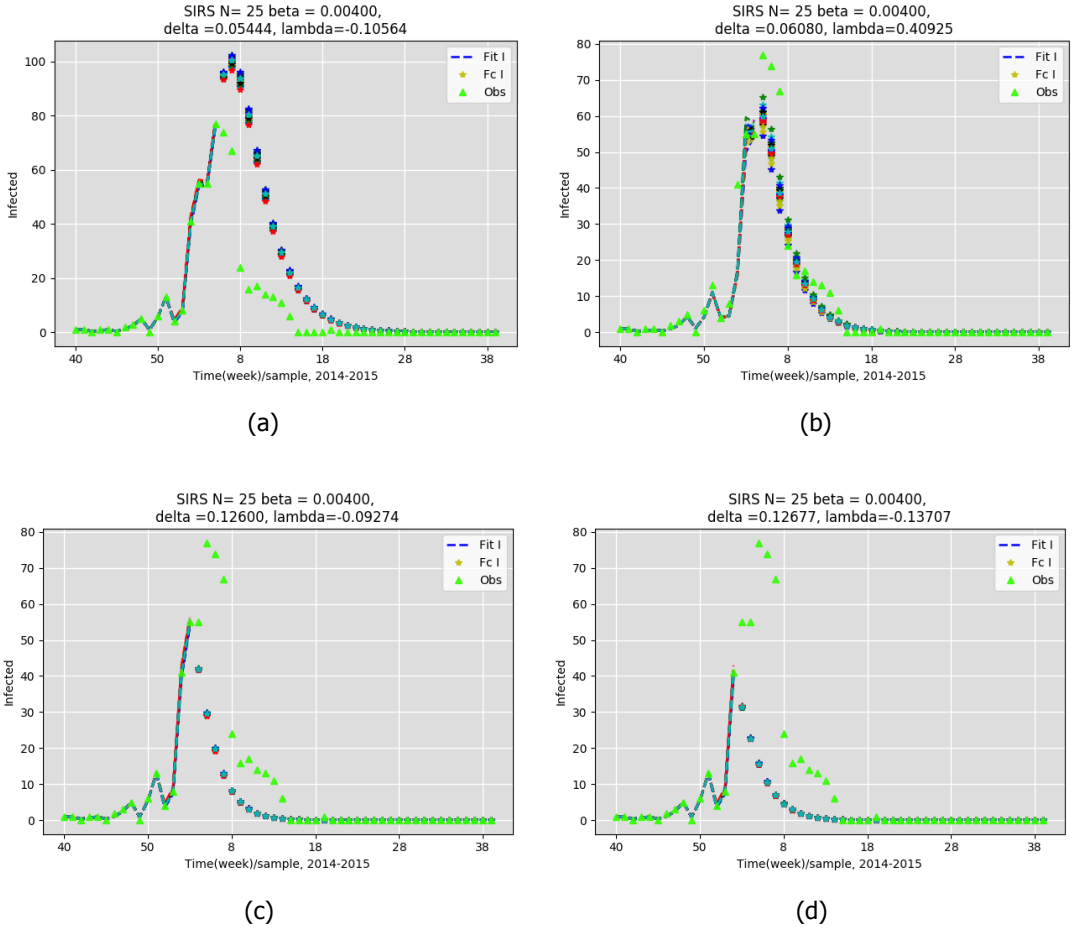
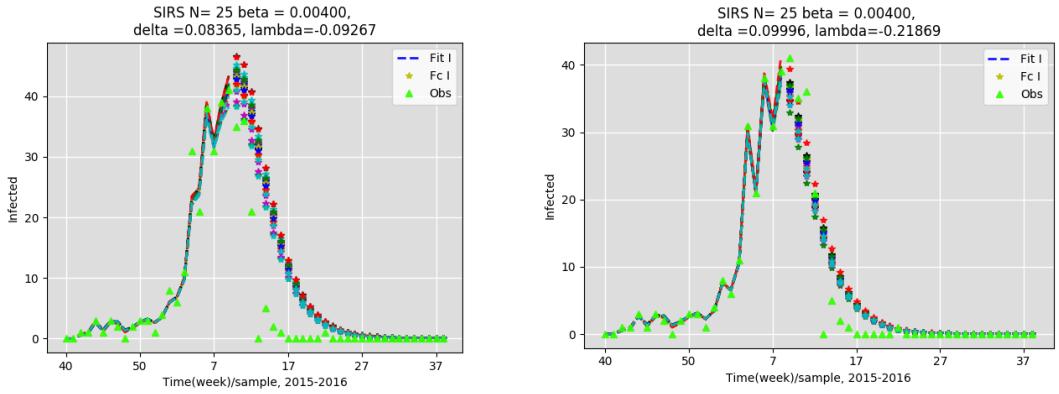


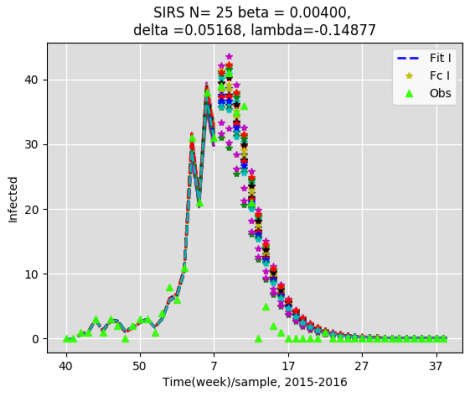
Figure 4.64: Forecasting, Belgium's influenza data 2014-2015, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker,

Belgium's influenza data 2015-2016

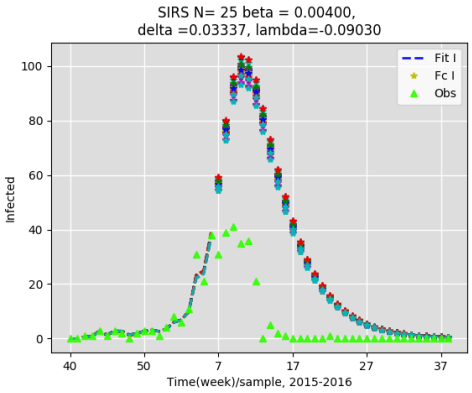


(a)

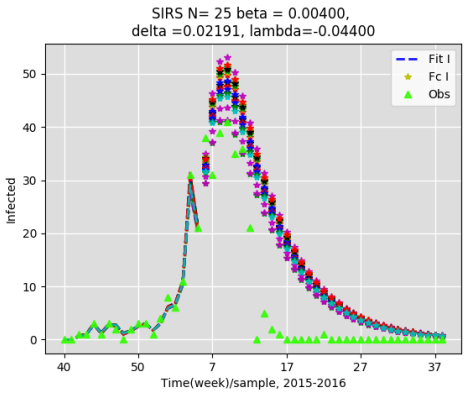
(b)



(c)



(d)



(e)

Figure 4.65: Forecasting, Belgium's influenza data 2015-2016, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker, (e) 4 weeks before the out-breaker,

Belgium's influenza data 2016-2017

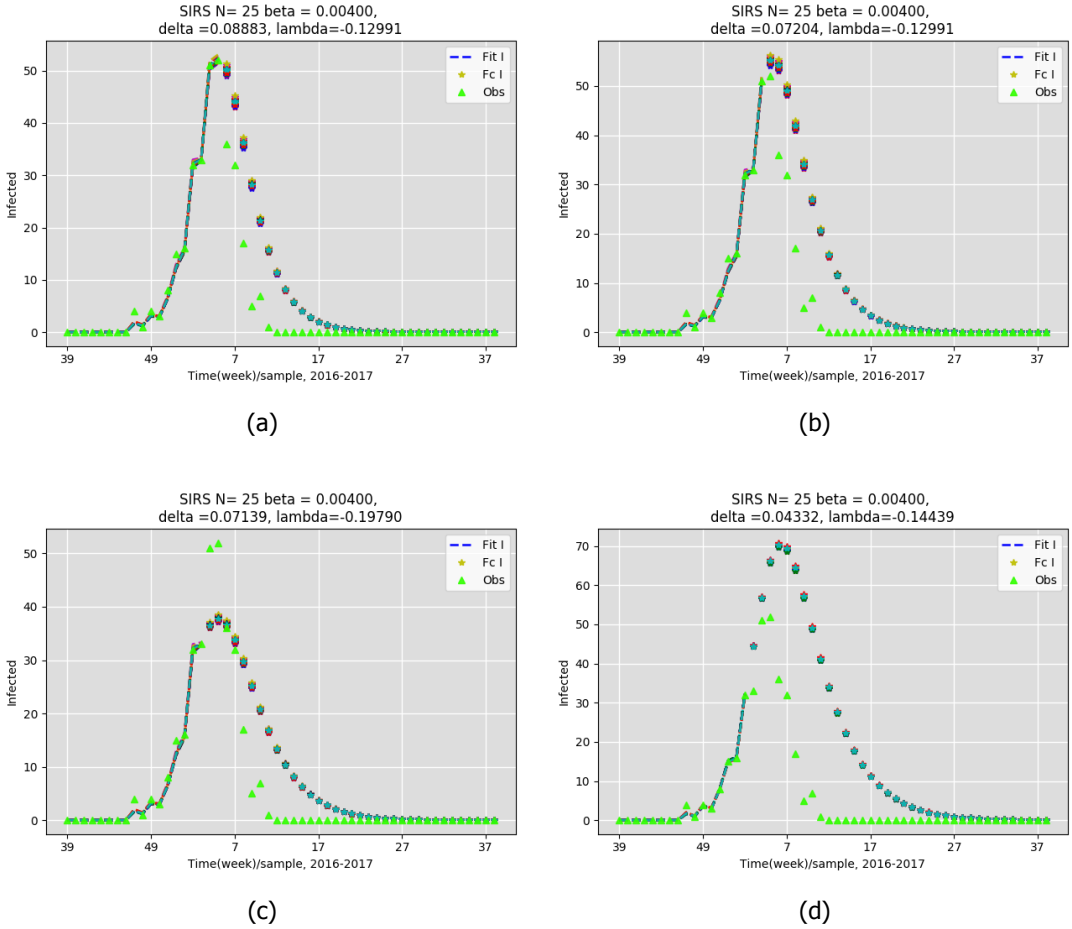


Figure 4.66: Forecasting, Belgium's influenza data 2016-2017, (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker,

4.4. Forecasting real-world epidemic data without the network

This section shows the case of not using the multi-dimensional graph effect (MDGE). Figures 4.67 and 4.68 show the forecasting will always point down compared to figures 4.56 and 4.61. Which does not give a good forecasting for the out-breaker peak compared to using MDGE method.

The UK's influenza data 2016-2017

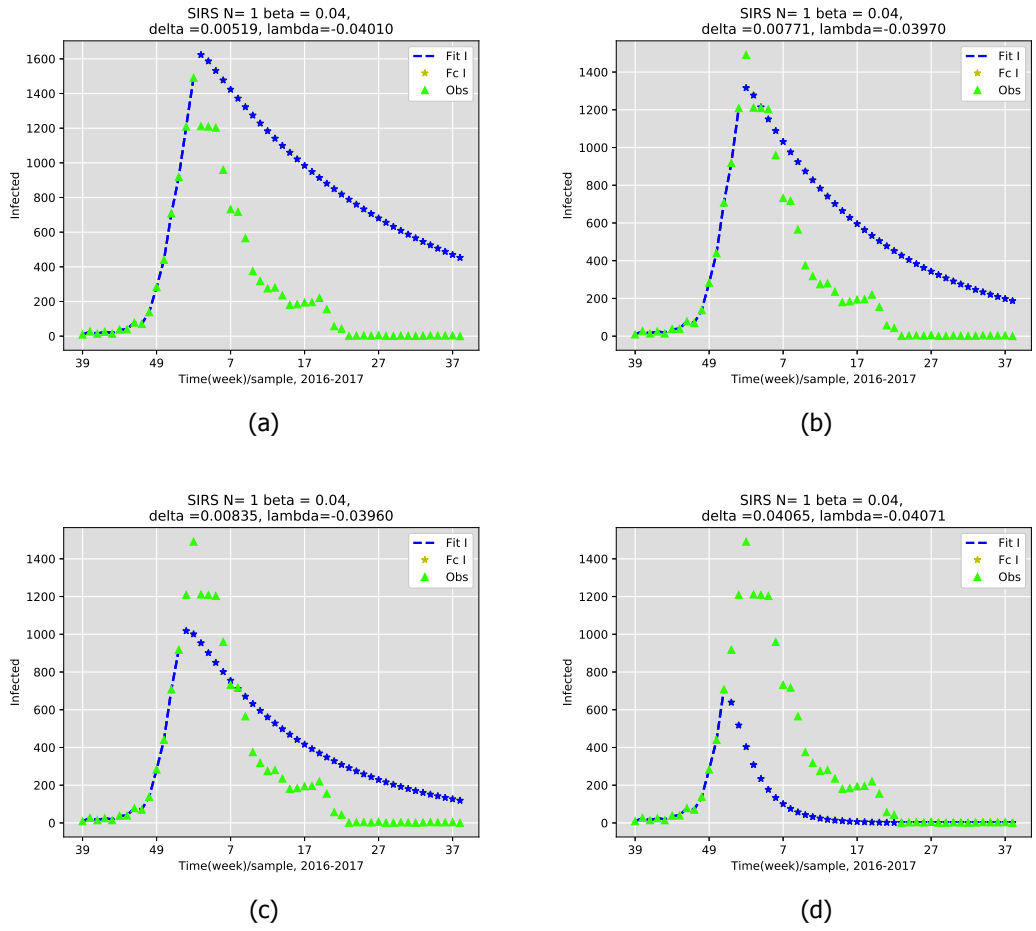


Figure 4.67: Forecasting, the UK's influenza data 2016-2017, without using the MDGE (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker,

Germany's influenza data 2016-2017

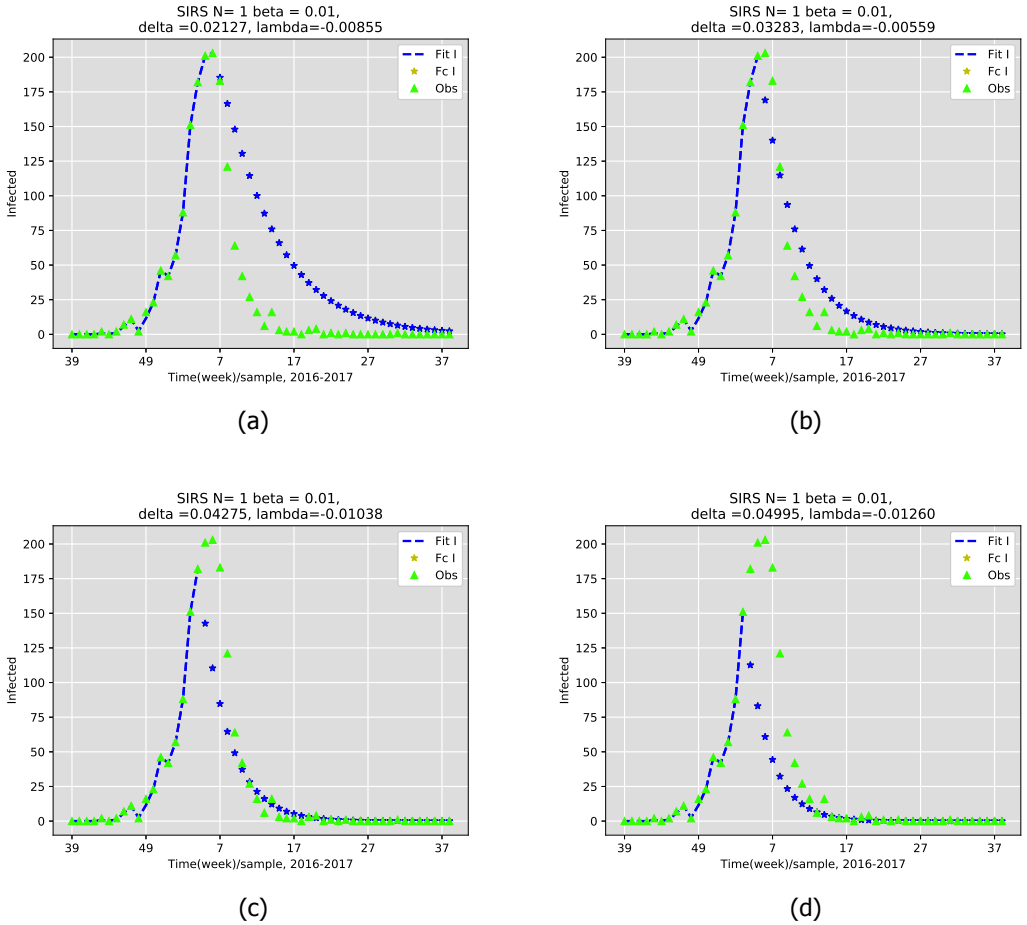


Figure 4.68: Forecasting, Germany's influenza data 2016-2017, without using the MDGE (a) after the out-breaker, (b) 1 weeks before the out-breaker, (c) 2 weeks before the out-breaker, (d) 3 weeks before the out-breaker,

Conclusion

In this chapter, we illustrate and discuss the thesis results of the synthetic epidemic (scenario 1 and 2) and real-world data. In scenario 1, we assume that the network and the epidemic infection rate β are known. Furthermore, we have the observations of each node $O_i(t)$. We fit/estimate I, δ and λ, then I is forecasted. The algorithm is validated and evaluated on synthetic epidemic spreading on four different graph model (regular, ER, WS, BA) and our algorithm is compared to other used algorithm. In scenario 2, we assume that we only have the average observations $\overline{O}(t)$. Therefore, we use the MDGE method to fit/estimate I, δ and λ, and then to forecast I. The algorithm in this scenario is also validated and evaluated on synthetic epidemic data. Finally, our fitting/estimating and forecasting algorithm is tested on influenza data on four countries (the Netherlands, the UK, Belgium and Germany) for the years 2012-2017.

5

Conclusion

In this thesis, we develop a fast and accurate algorithm to fit/estimate and forecast the epidemic process on networks. We employ an ensemble Kalman filter (EnKF) to estimate the epidemic model's (SIRS) state variables and parameters. Then the estimated state's error is corrected using Gradient descent for higher accuracy. Furthermore, we show that it works effectively on several types of graphs. This new approach is validated and evaluated with two synthetic epidemic scenarios and tested on real-world epidemic data (influenza). On synthetic epidemic we fit/estimate the model states with very low error, then forecast the out-breaker peak time correctly four weeks before the true one. Furthermore, we use the Multi-dimensional graph effect (MGDE) method to boost the filter function and overcome the noise.

Future work

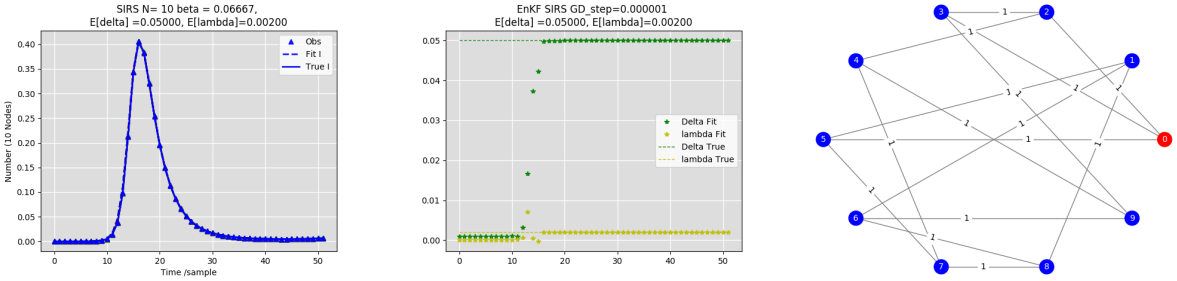
In this thesis, is a proof of concept that we can estimate the states of a certain epidemic model using our specific algorithm. However, a real-life epidemic may follow a more complex model. Therefore, for future work, we would like to test and analyze a more complex epidemic model. Furthermore, other estimation and forecasting algorithms can be developed and compared. Finally, other real-world epidemics spreading on real networks can experiment, and an optimization algorithm can be added to process very large scale networks.

A

Appendix

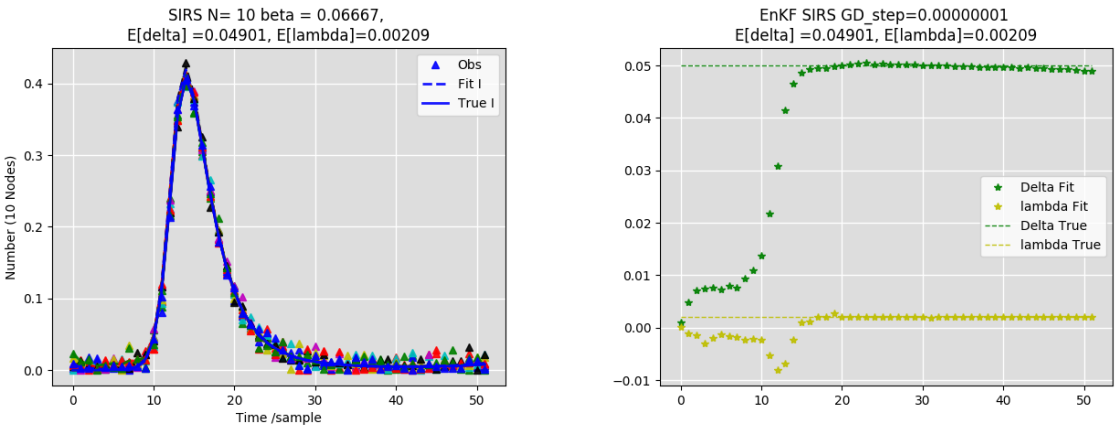
A.1. More results

Regular graph



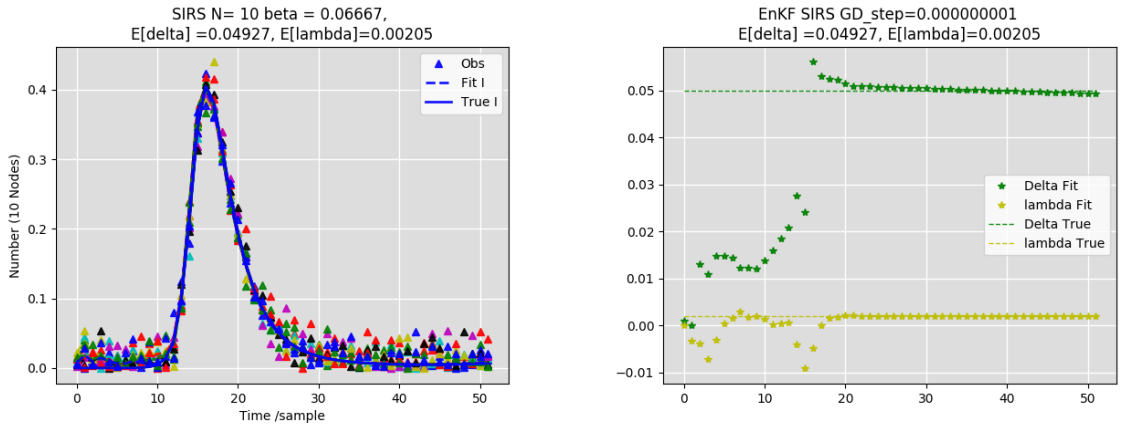
(a) (b) (c)

Figure A.1: Fitting synthetic epidemic spread on regular graph $G(10, 15), d = 3$, when $STD_{noise} = 0$, (a) Infection fitting, $Err_I = 0.000662$ (b) β and λ estimation, (c) The regular graph $G(10, 15), d = 3$



(a) (b)

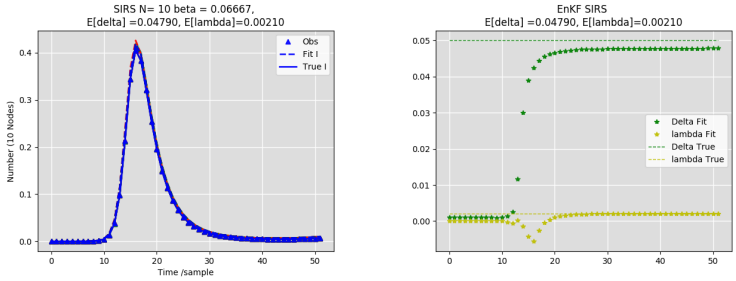
Figure A.2: Fitting synthetic epidemic spread on regular graph $G(10, 15), d = 3$, when $STD_{noise} = 0.01$ associated with Obs, (a) Infection fitting, $Err_I = 0.0024672$ (b) β and λ estimation



(a) (b)

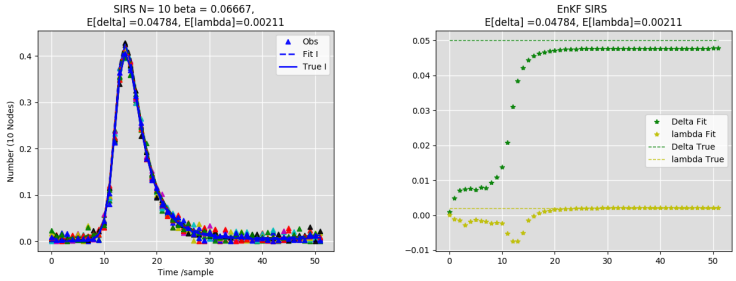
Figure A.3: Fitting synthetic epidemic spread on regular graph $G(10, 15), d = 3$, when $STD_{noise} = 0.02$ associated with Obs, (a) Infection fitting, $Err_I = 0.004335$ (b) β and λ estimation

Regular graph without GD



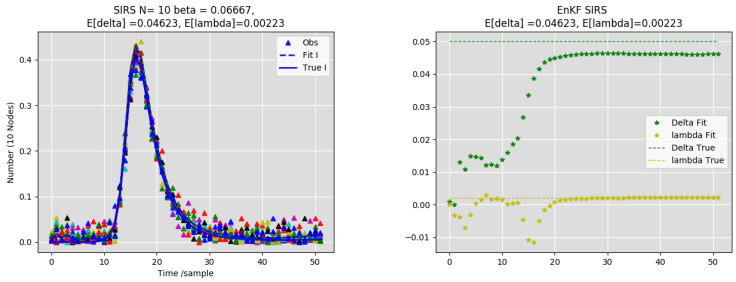
(a) (b)

Figure A.4: Fitting synthetic epidemic spread on regular graph $G(10, 15), d = 3$, when $STD_{noise} = 0$, GD was not used, (a) Infection fitting, $Err_I = 0.004402$, (b) β and λ estimation



(a) (b)

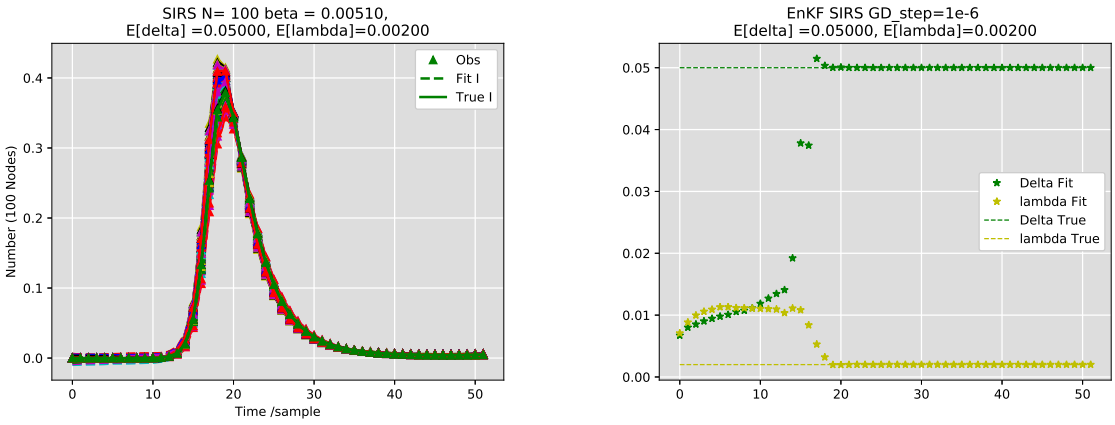
Figure A.5: Fitting synthetic epidemic spread on regular graph $G(10, 15), d = 3$, when $STD_{noise} = 0.01$ associated with Obs, GD was not used, (a) Infection fitting, $Err_I = 0.005504$, (b) β and λ estimation



(a) (b)

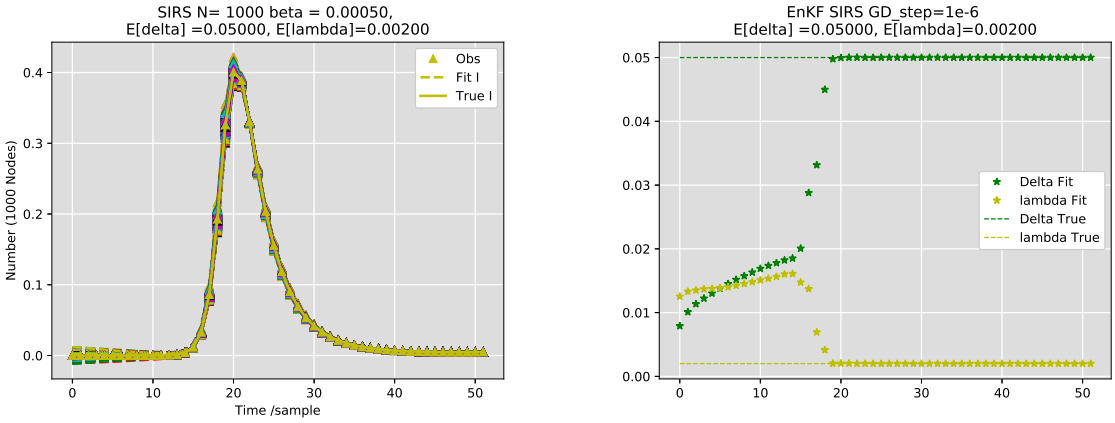
Figure A.6: Fitting synthetic epidemic spread on regular graph $G(10, 15), d = 3$, when $STD_{noise} = 0.02$ associated with Obs, GD was not used, (a) Infection fitting, $Err_I = 0.0088087$, (b) β and λ estimation

ER graph



(a) (b)

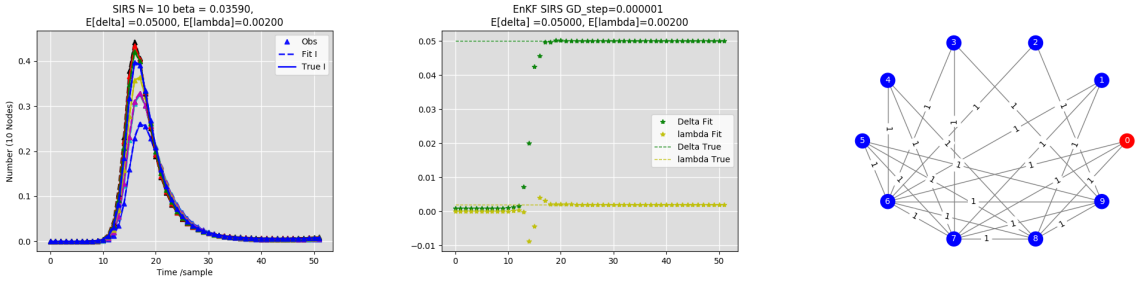
Figure A.7: Fitting synthetic epidemic spread on on ER graph $G(100, 1929), p = 0.4$, when $STD_{noise} = 0$ associated with Obs, (a) Infection fitting, $Err_I = 0.0008948$ (b) β and λ estimation



(a) (b)

Figure A.8: Fitting synthetic epidemic spread on on ER graph $G(1000, 200183), p = 0.4$, when $STD_{noise} = 0$ associated with Obs, (a) Infection fitting, $Err_I = 0.00069011$ (b) β and λ estimation

BA graph



(a) (b) (c)

Figure A.9: Fitting synthetic epidemic spread on BA graph $G(10, 24)$, when $STD_{noise} = 0$, (a) Infection fitting, $Err_I = 0.0006392$, (b) β and λ estimation, (c) The BA graph $G(10, 24)$

A.2. Susceptible-Infected-Susceptible (SIS)

The SIS epidemic process on networks is one of the simplest basic models for spreading on networks. The model has two stats (S,I) and two possible transitions, S –> I and I –> S. The node i can be in one of two states: susceptible or infected. The ratio $\tau = \frac{\beta}{\delta}$ is the effective infection rate. Figure 2.2b shows the SIS models with the infection rate and the curing rate. The SIS process has a transition phase that takes place at the epidemic threshold ($\tau > \tau_c$), $\tau_c = \frac{1}{\lambda_1}$, where λ_1 is the largest eigenvalue of the adjacency matrix A [8]. Therefore, if $\tau > \tau_c$, then the infection survives in the network and becomes epidemic. The epidemic stabilize at the equilibrium $q \approx 1 - \frac{\tau_c}{\tau}$, in case the spread on K_N, as shown in figures A.10a and A.10b. However, if $\tau < \tau_c$ the infection will die out and all nodes become healthy eventually (enters the all-healthy state). The infection stabilize at the equilibrium $q = 0$, as shown in figure A.10c

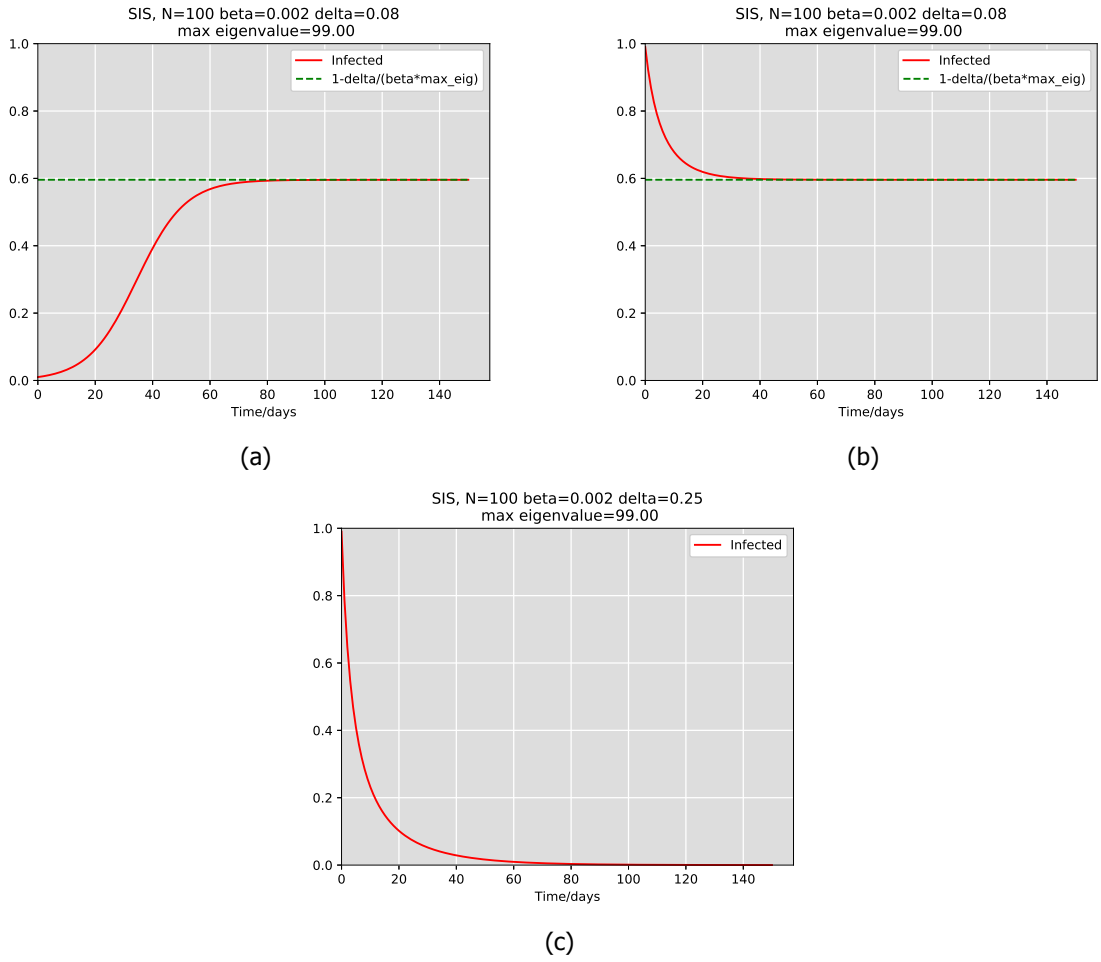


Figure A.10: SIS epidemic process on K_{100} network. (a) $\tau > \tau_c$ and $I_0 < q$, (b) $\tau > \tau_c$ and $I_0 > q$, (c) $\tau < \tau_c$ then $q = 0$

A.3. Susceptible-Infected-Recovered (SIR)

The SIR epidemic model introduce a permanent immunity to the system. The model has three stats (S,I,R) and two possible transitions, S –> I and I –> R. The R state occurs when an infectious individual recovers from the disease and is assumed to have acquired a permanent immunity or is removed (e.g., has died). Figure 2.2c shows the SIR models with the infection rate and the recovery rate. Figure A.11 shows that in SIR the disease will always die over time.

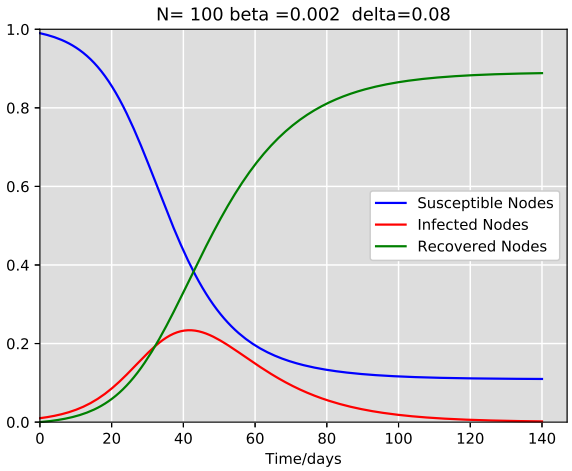


Figure A.11: SIR epidemic process on K_{100} network

A.4. Data preparation and processing

Data processing (DP) concerns over collecting, cleaning, clustering, normalizing and manipulating the items of the raw data primarily for use in analysis to producing a meaningful information [43].

Data manipulation may involve various processes, including:

- Data exploring, using visual exploration and tools to understand the characteristics of the dataset.
- Data validation, ensuring that supplied data is correct and relevant.
- Data sorting, arranging data items in some sequence and/or in different sets.
- Data summarization, reducing detail data to its main points.
- Data aggregation, combining multiple pieces/items of data.
- Data splitting, dividing the data into smaller chunks by using a certain method in order to handle and manipulate the data easier later on, especially when scaling and normalizing.
- Data analysis, collecting, organizing, analyzing, interpreting and presenting the data.
- Data reporting, giving detailed or summarized report of the computed information.
- Data classification, separation of data into various categories.

Data preparation is the pre-processing of the raw data and reshapes it making the data ready to be an input for the model. Data preparation varies depending on the dataset (No two datasets are the same). Therefore, the preparation cannot be carried out without exploring the raw data first [44]. The procedure is not fully automated yet and there is no single guide could cover everything. As a result, a lot of time needed to be spent on this step and systematic approach is required. Data cleaning (cleansing) is one of the most common tasks in data preparation. It is the process of detecting and correcting corrupted or inaccurate records from the recorded observations. The proper and careful data cleaning can make or break the model [45]. In other words, using properly cleaned dataset, makes even simple algorithms/models give an impressive and accurate output.

Removing unwanted observations

The cleaning starts mostly with removing unwanted observations, which includes duplicate or irrelevant observations. The duplicate observations present mostly while data collection, due to combining or receive datasets from multiple sources. The irrelevant observations are the one that exists in the dataset but they do not belong to it. Handling the irrelevant observations before engineering features can prevent from struggling with bad output at the end.

Figure A.12: The irrelevant observations

Fix Structural Errors

Structural errors arise during measurement, data transfer, or other types of ”poor housekeeping.”, also can be caused by typos and mislabeling classes

Filter Unwanted Outliers

Outliers are the extremely big or small values that are located very far away outside the other observations. However, removing an outlier observation without a legitimate reason may badly impact the model's performance. Meaning, the outliers must not be removed because they are just big numbers, for instance, Those big/small numbers (outliers) might be very informative for the model. finally, to drop an outlier a clear good reason is needed, such as if we know, from experience, that the number is too big to be true (e.g. speed of a car is 1500 km/h or maybe negative while all the numbers should be positive, ... etc).

Handle Missing Data

Missing data is a deceptively tricky issue and missing values cannot simply be ignored. The most two commonly recommended ways to deal with the problem are:

- Dropping observations that have missing values, which may lead to drop information.
- Imputing the missing values based on other observations. it is the process of replacing missing data with substituted values.

The fact is both ways can cause problems later on. The best practice is to tag missing data. if the missing data is categorical, you can add a new class for the feature "Miss". This will inform the algorithm that the value is missed. If missing data is numeric, the strategy is flagging and filling the values. Using this technique allows the algorithm to estimate the optimal constant for missingness, instead of just filling it in with the mean.

Bibliography

[1] D. J. Watts and S. H. Strogatz, *Collective dynamics of 'small-world'networks,* nature **393**, 440 (1998).

[2] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems* (" O'Reilly Media, Inc.", 2017).

[3] M. E. J. Newman, *Networks: an introduction* (Oxford University Press, Oxford, 2010).

[4] L. J. Allen, F. Brauer, P. Van den Driessche, and J. Wu, *Mathematical epidemiology*, Vol. 1945 (Springer, 2008).

[5] P. Van Mieghem, *Graph spectra for complex networks* (Cambridge University Press, 2010).

[6] P. Van Mieghem, *Performance analysis of complex networks and systems* (Cambridge University Press, 2014).

[7] P. Van Mieghem, J. Omic, and R. Kooij, *Virus spread in networks,* IEEE/ACM Transactions on Networking (TON) **17**, 1 (2009).

[8] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani, *Epidemic processes in complex networks,* Reviews of modern physics **87**, 925 (2015).

[9] A. Lahrouz, L. Omari, and D. Kiouach, *Global analysis of a deterministic and stochastic nonlinear sirs epidemic model,* Nonlinear Analysis: Modelling and Control **16**, 59 (2011).

[10] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, *A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking,* IEEE Transactions on signal processing **50**, 174 (2002).

[11] P. J. Van Leeuwen, *Particle filtering in geophysical systems,* Monthly Weather Review **137**, 4089 (2009).

[12] A. Skvortsov and B. Ristic, *Monitoring and prediction of an epidemic outbreak using syndromic observations,* Mathematical biosciences **240**, 12 (2012).

[13] J. Shaman, V. E. Pitzer, C. Viboud, B. T. Grenfell, and M. Lipsitch, *Absolute humidity and the seasonal onset of influenza in the continental united states,* PLoS biology **8**, e1000316 (2010).

[14] J. Shaman and A. Karspeck, *Forecasting seasonal outbreaks of influenza,* Proceedings of the National Academy of Sciences **109**, 20425 (2012).

[15] J. Shaman, S. Kandula, W. Yang, and A. Karspeck, *The use of ambient humidity conditions to improve influenza forecast,* PLoS computational biology **13**, e1005844 (2017).

[16] N. R. Council *et al.*, *Network Science Committee on Network Science for Future Army Applications* (The National Academies Press, Washington, DC, 2005).

[17] L. Euler, *Solutio problematis ad geometriam situs pertinentis,* Opera Omnia **7**, 128 (1936).

[18] K.-C. Chang, K. Pearson, and T. Zhang, *Perron-frobenius theorem for nonnegative tensors,* Communications in Mathematical Sciences **6**, 507 (2008).

[19] E. Paul and R. Alfréd, *On random graphs i,* Publicationes Mathematicae (Debrecen) **6**, 290 (1959).

[20] A.-L. Barabási and R. Albert, *Emergence of scaling in random networks,* science **286**, 509 (1999).

[21] R. Klages, *Introduction to dynamical systems*, Vol. 24 (2008).

[22] A. Bagdasaryan, *Systems theoretic techniques for modeling, control and decision support in complex dynamic systems,* Artificial Intelligence Resources in Control and Automation Engineering , 15 (2012).

[23] P. Van Mieghem and R. Van de Bovenkamp, *Non-markovian infection spread dramatically alters the susceptible-infected-susceptible epidemic threshold in networks,* Physical review letters **110**, 108701 (2013).

[24] Q. Liu and P. Van Mieghem, *Burst of virus infection and a possibly largest epidemic threshold of non-markovian susceptible-infected-susceptible processes on networks,* Physical Review E **97**, 022309 (2018).

[25] F. D. Sahneh, C. Scoglio, and P. Van Mieghem, *Generalized epidemic mean-field model for spreading processes over multilayer complex networks,* IEEE/ACM Transactions on Networking (TON) **21**, 1609 (2013).

[26] R. M. Anderson and R. M. May, *Infectious diseases of humans: dynamics and control* (Oxford university press, 1992).

[27] O. Diekmann and J. A. P. Heesterbeek, *Mathematical epidemiology of infectious diseases: model building, analysis and interpretation*, Vol. 5 (John Wiley & Sons, 2000).

[28] P. Guo, X. Yang, and Z. Yang, *Dynamical behaviors of an siri epidemic model with nonlinear incidence and latent period,* Advances in Difference Equations **2014**, 164 (2014).

[29] M. L. G. Kuniyoshi and F. L. P. dos Santos, *Mathematical modelling of vector-borne diseases and insecticide resistance evolution,* Journal of Venomous Animals and Toxins including Tropical Diseases **23**, 34 (2017).

[30] Q. Liu and P. Van Mieghem, *Autocorrelation of the susceptible-infected-susceptible process on networks,* Physical Review E **97**, 062309 (2018).

[31] G. Bishop, G. Welch, *et al.*, *An introduction to the kalman filter,* Proc of SIGGRAPH, Course **8**, 59 (2001).

[32] R. Faragher *et al.*, *Understanding the basis of the kalman filter via a simple and intuitive derivation,* IEEE Signal processing magazine **29**, 128 (2012).

[33] A. Hommels, A. Murakami, and S.-I. Nishimura, *A comparison of the ensemble kalman filter with the unscented kalman filter: application to the construction of a road embankment,* Geotechniek **13**, 52 (2009).

[34] S. Gillijns, O. B. Mendoza, J. Chandrasekar, B. De Moor, D. Bernstein, and A. Ridley, *What is the ensemble kalman filter and how well does it work,* in *American control conference*, Vol. 6 (IEEE, 2006).

[35] R. Labbe, *Kalman and bayesian filters in python* (Github. com, 2014).

[36] *educing loss: Gradient descent,* https://developers.google.com/machine-learning/crash-course/reducing-loss/gradient-descent (2018), accessed: 2018-10-02.

[37] J. Barzilai and J. M. Borwein, *Two-point step size gradient methods,* IMA journal of numerical analysis **8**, 141 (1988).

[38] *Gradient descent,* https://hackernoon.com/gradient-descent-aynk-7cbe95a778da (2018), accessed: 2018-09-15.

[39] *Gradient descent,* https://towardsdatascience.com/gradient-descent-in-a-nutshell-eaf8c (2018), accessed: 2018-09-15.

[40] J. L. Anderson, *An ensemble adjustment kalman filter for data assimilation,* Monthly weather review **129**, 2884 (2001).

[41] W. Yang, A. Karspeck, and J. Shaman, *Comparison of filtering methods for the modeling and retrospective forecasting of influenza epidemics,* PLoS computational biology **10**, e1003583 (2014).

[42] *World health organization (who) influenza dataset,* http://apps.who.int/flumart/Default?ReportNo=12 (2018), accessed: 2018-05-15.

[43] C. French, *Data Processing and Information Technology* (Cengage Learning EMEA, 1996).

[44] D. Pyle, *Data preparation for data mining* (morgan kaufmann, 1999).

[45] *data-cleaning,* https://elitedatascience.com/data-cleaning (2018), accessed: 2018-10-12.