## ABSTRACT

ZHANG, MENG. Blood Glucose Forecasting: Data-driven Model, Physiological Model, and Hybrid Approaches. (Under the direction of Kevin B. Flores and Hien T. Tran.)

Type 1 diabetes is a chronic disease due to the patient's pancreas producing no or little insulin. Unfortunately, there is no cure for type 1 diabetes. Controlling blood glucose in the euglycemic range is the main goal of developing an artificial pancreas for type 1 diabetes patients. The computer algorithm in the artificial pancreas can determine the time and amount of insulin delivery through the pump therapy. A computationally efficient and accurate model that can capture the physiological nonlinear dynamics is critical for developing an artificial pancreas.

In this thesis, we find a few approaches to achieve the goal. Four data-driven models, including deep learning architecture and regression models, were employed for the blood glucose prediction using the real-world data obtained for 12 data contributors. A neural network model using an encoder-decoder architecture has the most stable performance and is able to recover missing dynamics in short time intervals. Regression models performed better at long-time prediction horizons (i.e., 60 minutes) and with lower computational costs.

Next, we construct a complex physiological model to gain some interpretability for blood glucose dynamics. The physiological model contains six subsystems where the blood glucose transformation can be observable by solving the inverse problem and obtained the model parameters.

Finally, we combine the data-driven model and physiological model for a hybrid model, which replaced one of the variables in the physiological model by reservoir computing, one of the regression models introduced in data-driven models. In this way, the hybrid model has the interpretability compared to data-driven models and is observed to improve the prediction accuracy compared to the physiological model.

Blood Glucose Forecasting: Data-driven Model, Physiological Model, and Hybrid Approaches

by
Meng Zhang

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Applied Mathematics

Raleigh, North Carolina

2021

APPROVED BY:

| | |
|---|---|
| Alun Lloyd | Arvind K. Saibaba |
| Kevin B. Flores<br>Co-chair of Advisory Committee | Hien T. Tran<br>Co-chair of Advisory Committee |

## DEDICATION

To my future kids, we haven't met yet, but I have been picturing you. When you grow up, please join with me to build a better world.

## BIOGRAPHY

Meng Zhang was born on 7 June 1994 in Shandong, China. She went to Shandong Normal University for the first two years and went abroad for an exchange program at East Tennessee State University. She graduated with a 4.00 GPA (out of 4.00) with a B.S. in Computational Applied Mathematics from East Tennessee State University, and obtained a B.S. in Information Technology and Science from Shandong Normal University in 2016. She started her journey with mathematics with the undergraduate advisor's suggestion for pursuing a Ph.D. degree in Applied Mathematics at North Carolina State University. During her graduate studies, she found research interests with applications of mathematical models in the healthcare field. She had one internship with SAS Institute in Summer 2019 as an econometrics fellow and another internship with UnitedHealth Group in Summer 2020 as a Ph.D. data scientist. The experience of working in industrial companies grew her passion for making contributions to society using the knowledge of mathematics.

She decided to continue working with UnitedHealth Group after completing her Ph.D. degree in 2021, and hopefully could transition her research to a product for the Diabetes community in the near future.

## ACKNOWLEDGEMENTS

This thesis cannot be completed without the help from two important people: Dr. Hien T. Tran, my PhD advisor, and Dr. Chi Zhang, my husband.

I am blessed to meet my Ph.D. advisor, Hien Tran, he is the guiding light of my research journey. He put faith in me while I failed to do so. He is always gentle, energetic, and helpful every time we meet and discuss research. I would also like to thank my husband, Chi Zhang, for being thoughtful while I get upset about the progress of research, and for the kisses and hugs which encouraged me to go through the very difficult moments in the graduate program.

I want to express my thanks to my co-advisor, Kevin Flores, for all the great ideas and inspiration for my research, and for the patience anytime I had questions or concerns. I am also grateful to have Arvind Saibaba on my committee. When I met him at the first lecture in the graduate program, he showed me his enthusiasm and preciseness for treating research and teaching.

I want to express my thanks to my undergraduate advisor, Michele Joyner. She persuaded me to keep studying mathematics and cheered me up during my Ph.D. study. I would also like to thank my friends and colleagues I met in the workshops and at North Carolina State University, especially John Lagergren, Yutian Liu, Juliana Lazarette Pin, Rafael Luiz da Silva, Ruyu Tan, Ling Xiao, and Li Zhu. They offered me a lot of help and company. They made me feel like I had never been alone.

I want to thank my father, Xinqi Zhang, and mother, Bian Jie, for their endless love and support, and for taking the time to listen to my complaints. My mom never had a chance to go to college, but she is always willing to learn and gain knowledge with me.

Finally, I want to thank my little friend, Ahvi. She stayed with me day and night when I was reading literature, designing algorithms, coding, and writing this thesis. She is my best baby cat.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

CHAPTER

# 1

# INTRODUCTION

Diabetes is a common chronic disease in which the blood glucose (BG) levels, also called blood sugar levels, are too high. By the American Diabetes Association, 10.5% of the U.S. population had diabetes in the year 2018. From [Sae19], the estimation of global prevalence in 2019 is 9.3%, with one in two people going undiagnosed for a long period.

We obtain glucose from three resources: the intestinal absorption after meal intake, the decomposition of glycogen in liver and muscles, and other noncarbohydrate compounds (i.e. gluconeogenesis) transformation in the liver and kidney. The plasma-glucose concentration should be in the range of 70 - 140 mg/dl with an empty stomach, and fall below 200 mg/dl after 2 hours digesting a meal [Geo18]. Insulin is the hormone produced by the pancreas that helps glucose go into cells to transfer energy or store the extra glucose (i.e. glycogenesis and lipogenesis). The causes of common type diabetes, type 1 diabetes (T1D) and type 2 diabetes (T2D), are due to lack of production in insulin. Without the regulation of BG levels by insulin, the increasing or decreasing BG levels may cause hyperglycemia or hypoglycemia, which leads to irreversible body damage gradually.

In this thesis, we focus on T1D, in which the patient's body can produce little or no insulin due to insulin-producing pancreatic $\beta$ cell destruction. T1D can be diagnosed at any age, but most patients are diagnosed with T1D before 40. The overall male/female ratio is 1.8 according to the study of [Ost08]. At present, the sensor-augmented pump therapy utilized insulin pumps and continuous glucose monitoring (CGM) [Atk14] to deliver insulin for T1D treatment.

Currently, researchers are trying to combine CGM and insulin pumps with a computer algorithm to compose an artificial pancreas (Figure 1.1). Several papers introducing machine learning or deep learning algorithms [Oor16; Zhu18; Mid18; Mar18; Che18; MO20] while others focus on physiological models [Mant ; Mann ; Sch18; Vic97] to build the connection between devices and predictive

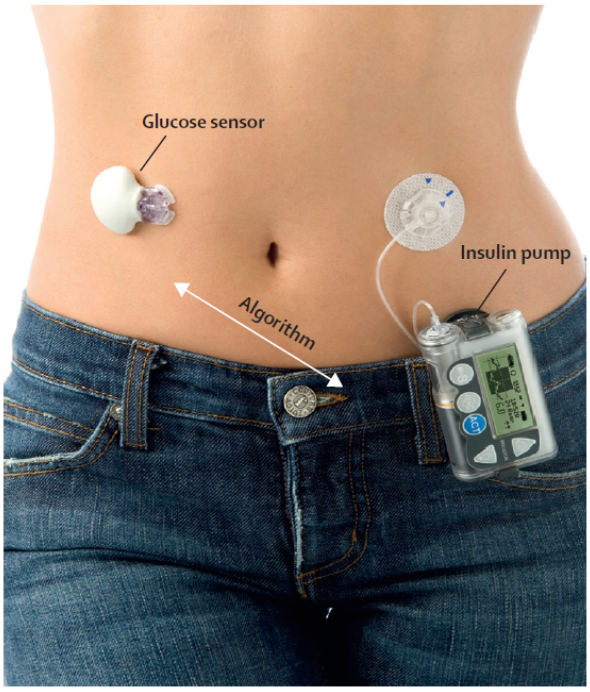


**Figure 1.1** The representation of an insulin pump and CGM device from [Atk14].

algorithms. However, due to the limitation of the current knowledge and other machine errors such as sensor noise, calibration of CGM, unannounced events, etc., the results are not satisfying. Hence, we wish to investigate the care of T1D and develop an efficient way to manage BG levels for artificial pancreases.

## 1.1 Thesis Outline

The remainder of this chapter is to introduce the motivation of the research project and describe the data resources we obtained from the Ohio hospital. In Chapter 2, we present 4 data-driven models and test the performance for BG level predictions, and discuss future improvements. Chapter 3 is dedicated to constructing and testing the physiological system, where we can describe the biological mechanisms of the BG transfer and interaction with organs and tissues. In Chapter 4, we propose a hybrid approach to replace a compartment in the physiological model by a data-driven architecture, and test how much it could improve the prediction results. Finally, conclusions and future works are given in Chapter 5.

## 1.2 Problem Statement

Since T1D is the most severe kind of diabetes and it can not be prevented or cured, monitoring the BG levels for T1D is significantly important. However, we can effectively treat patients with insulin

and manage their BG levels in order to maintain normal body functions. Patients should avoid the situations for either high BG levels (hyperglycemia), or low BG levels (hypoglycemia), where both symptoms can lead to life-threatening phenomena in the long term due to the permanent damage to blood vessels, nerves, and organs. Here, hyperglycemia is defined as having a fasting BG value greater than 140 mg/dl or greater than 180 mg/dl if measured postprandial, within two hours after a meal. Hypoglycemia is defined by low levels of BG when it falls below 60 mg/dl.

Maintaining good BG control for the T1D patient is difficult. Due to the calibration of continuous monitoring devices and measurement noises, identifying the abnormal CGMs and providing the proper treatment are challenging in the T1D community. An accurate prediction of BG could mitigate the events of hyper- or hypo-glycemia. Not only it prevents the impending problems brought from hyper- or hypo-glycemia, but also provides more possibilities for curing more patients by establishing and enhancing individual BG prediction profiles.

Keeping those ideas in mind, we hope to improve the ability of artificial pancreas by developing an efficient algorithm connecting the CGM signals and insulin delivery based on the daily routine of a specific patient. We believe that the artificial pancreas with an accurate predictive model can prevent the BG levels from going out of control. By learning from the performance of data-driven models, i.e. machine learning and physiological models, we are going to propose a hybrid model that utilizes the advantages from both methods. With the accuracy from the data-driven model and interpretability from the physiological model, the hybrid model can be robust in predicting BG levels and providing other biological variables (meal intake, insulin doses, heart rate, etc.). Hopefully, our research project can benefit the T1D patients by managing their BG levels in the normal range and reducing the damage of diabetes to the organs and tissues.

## 1.3 Data Description

We obtain the OhioT1DM Dataset [Marp ] for evaluating the BG levels prediction task. The data contains 12 anonymous T1D data contributors with continuous eight weeks' monitoring, including CGM, insulin device, physiological data, and self-report life events. Those data contributors are referred by the randomly selected ID numbers. Table 1.1 shows the contributor ID, gender, age, device for the insulin pump therapy for each contributor.

There are 5 females and 7 males among the 12 data contributors, 3 of them with age range in 20-40 years, 8 in the age range of 40-60 years, and the rest in the age range of 60-80 years. Only 3 of the contributors use the Medtronic 630G device, which provides better accuracy on CGM reading compared to 530G. The usage of the fitness band is split evenly between Empatica Embrace and Basis Peak. The difference between the two products is the category it collects. The format of data is one XML file for each contributor with training and testing separated approximately at 80% of total data.

The complete categories include: CGM of BG levels every 5 minutes, calibrated BG levels measured by finger sticks periodically, insulin doses (bolus and basal), self-reported events such as meal

**Table 1.1** Detailed information of data contributors with ID, gender, age, devices.

| ID | Gender | Age | Insulin Pump Device | Fitness Band |
|---|---|---|---|---|
| 540 | male | 20-40 | Medtronic 630G | Empatica Embrace |
| 544 | male | 40-60 | Medtronic 530G | Empatica Embrace |
| 552 | male | 20-40 | Medtronic 630G | Empatica Embrace |
| 559 | female | 40-60 | Medtronic 530G | Basis Peak |
| 563 | male | 40-60 | Medtronic 530G | Basis Peak |
| 567 | female | 20-40 | Medtronic 630G | Empatica Embrace |
| 570 | male | 40-60 | Medtronic 530G | Basis Peak |
| 575 | female | 40-60 | Medtronic 530G | Basis Peak |
| 584 | male | 40-60 | Medtronic 530G | Empatica Embrace |
| 588 | female | 40-60 | Medtronic 530G | Basis Peak |
| 591 | female | 40-60 | Medtronic 530G | Basis Peak |
| 596 | male | 60-80 | Medtronic 530G | Empatica Embrace |

intake, exercise, sleep, work, stress, and illness, and the physiological data in fitness bands. The categories from Basis Peak band include 5-minute aggregations of galvanic skin response (GSR), skin temperature, air temperature, heart rate, and step count. The other fitness band Empatica Embrace includes 1-minute aggregations of skin temperature, GSR, and acceleration. However, there exist missing categories in some contributors as shown in Table 1.2.

Each feature is recorded with time index and quantity such as quality level, size, duration, etc. The time is randomly shifted to a future date and month for confidential purposes. We now introduce each feature in the data fields below:

- **Acceleration**: the magnitude of the acceleration of movement that was recorded in Empatica Embrace sensor band for body movement, measured at every one minute.
- **Air temperature**: room temperature measured in Fahrenheit for every 5 minutes for people who wore the Basis Peak sensor band.
- **Band sleep**: sleep time was recorded for people wearing Empatica Embrace devices, and the numeric estimation of sleep quality only for people wearing Basis Peak devices.
- **Basal**: the continuous basal insulin injection rate, the basal rate begins at a specified time point and continues until the next value is set, unless the temporary basal supersedes the normal basal rate advised by a doctor, and then the size of temporary basal will replace the normal basal rate for the duration it is given.
- **Bolus**: fast-acting insulin delivered before a meal or when hyperglycemia occurs, recorded time and quantity of insulin dose.
- **Bolus meal**: if the bolus injection is associated with a meal event, then the carbohydrate estimation is recorded.

**Table 1.2** List of features for each contributor in the original XML file, × represents missing records for a certain feature, ✓ represents available feature.

| Features | 540 | 544 | 552 | 559 | 563 | 567 | 570 | 575 | 584 | 588 | 591 | 596 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Acceleration | ✓ | ✓ | ✓ | × | × | ✓ | × | × | ✓ | × | × | ✓ |
| Air temperature | × | × | × | ✓ | ✓ | × | ✓ | ✓ | × | ✓ | ✓ | × |
| Band sleep | × | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Basal | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Bolus | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Bolus meal | × | × | × | ✓ | ✓ | × | ✓ | ✓ | × | ✓ | ✓ | × |
| CGM | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Exercise | × | ✓ | ✓ | ✓ | ✓ | × | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Finger stick | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| GSR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Heart rate | × | × | × | ✓ | ✓ | × | ✓ | ✓ | × | ✓ | ✓ | × |
| Hypo event | × | ✓ | ✓ | ✓ | ✓ | × | ✓ | ✓ | × | ✓ | ✓ | ✓ |
| Illness | × | × | ✓ | ✓ | ✓ | ✓ | × | ✓ | × | ✓ | ✓ | ✓ |
| Self-report meal | × | × | × | ✓ | ✓ | × | ✓ | ✓ | × | ✓ | ✓ | × |
| Self-report sleep | × | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Skin temperature | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Step count | × | × | × | ✓ | ✓ | × | ✓ | ✓ | × | ✓ | ✓ | × |
| Stress | × | ✓ | × | ✓ | × | × | × | × | × | ✓ | × | ✓ |
| Work | × | ✓ | ✓ | ✓ | ✓ | × | ✓ | ✓ | ✓ | ✓ | × | × |

- **CGM**: monitored BG levels for every 5 minutes, may be shifted forward or backward due to the calibration.
- **Exercise**: self-reported exercises with the starting time and duration, the intensity indicates the contributor's subjective assessment of exercising exertion. On a scale of 1 to 10, 10 is the most intense.
- **Finger Stick**: BG levels obtained through contributor's drop of blood from a finger, usually for calibration of CGM monitoring device.
- **GSR**: also refers to skin conduce or electrodermal activity, measured at every 5 minutes for Basis Peak band and every 1 minute for Empatica Embrace band. The GSR represents the sweat gland activity, the magnitude will increase after exercising.
- **Heart rate**: heartbeats measured every 5 minutes for those people who wore Basis Peak band.
- **Hypo event**: the time associated with a self-reported hypoglycemia event.
- **Illness**: starting time for self-reported illness event.
- **Self-reported meal**: includes the time and estimation size of carbohydrate intake.

- **Self-reported sleep**: time and quality of self-reported sleep. Quality is represented by a number: 1 for poor, 2 for fair, and 3 for good.
- **Skin temperature**: body temperature measured in Fahrenheit for every 5 minutes for people who wore the Basis Peak sensor band, and for the Empatica Embrace band is every 1 minute.
- **Step count**: only available for people who wore the Basis Peak, updates the magnitude of steps a contributor moves every 5 minutes.
- **Stress**: time of self-reported stress event.
- **Work**: self-reported time duration and self-assessing intensity of working. On a scale of 1 to 10, 10 is the most intense.

CHAPTER

# 2

# DATA-DRIVEN MODELS

## 2.1 Introduction

T1D is a chronic disease with an incidence of 15 per 100,000 people worldwide between 1980 and 2019 [Mob20]. Due to the destruction of insulin-producing pancreatic $\beta$ cells in T1D patients, the pancreas can produce little or no insulin. Disruption of insulin regulation can lead to a diseased state since this hormone helps regulate the storage and utilization of glucose obtained from carbohydrate intake. Lack of insulin will lead to a failure to control regular glucose-insulin transformation and can cause long-term complications including permanent damage to organs and nerves. There is no cure for T1D, patients with this disease need lifetime exogenous insulin injections to manage BG levels in the euglycemic range, i.e., 70-140 mg/dl with an empty stomach and below 200 mg/dl after 2 hours of digestion [Geo18].

In recent years, insulin pumps and CGM are two technologies that have been used to develop sensor-augmented pump therapy for T1D treatment [Atk14]. Researchers try to combine CGM and insulin pumps with a computer algorithm to compose a closed-loop system, i.e, artificial pancreas. If the CGM goes above a certain threshold then insulin starts being delivered through the pump and stops if BG levels are sufficiently low. A key challenge to developing this system is designing an accurate control algorithm that predicts the trend of glucose levels and manages the insulin amount to infuse. For this reason, the development of models that can accurately predict BG levels is significant to the effectiveness of the artificial pancreas.

Previously, three types of models have been developed for blood glucose prediction: physiological models, data-driven models, and hybrid models. Physiological models are mathematical models that apply the biological understanding of glucose transformation to simulate the dynamics of the insulin-

glucose regulation system. Data-driven models include statistical models (e.g., linear regression) and artificial neural networks (e.g., deep learning models). Finally, hybrid models combine physiological models with data-driven models.

In the framework of physiological models, the minimal model by Bergman et al. [Ber81] describes the metabolism of glucose and insulin regulation. Man et al. [Mant ] proposed a model that consists of glucose-insulin systems with the effect of glucose absorption in gastro-intestinal, endogenous glucose production, glucose utilization, and renal excretion. Later, Man et al. extended this physiological model to include effects from physical activity [Mann ].

With regards to the data-driven models, a complete review can be found in [Wol19]. In recent years, models attempting to predict BG levels have included Auto-Regression (AR) models [Xie18; Nem20; Kha20; Ma20; Ant20; RR19], XG-Boost [Mid18], Grammatical Evolution (GE) [Con18], Genetic Programming (GP) [Joe20], Neural Basis Expansion for Interpretable Time-Series Forecasting (N-BEATS) [RF20], latent-variable (LV) based models [Sun20], Generative Adversarial Networks (GANs) [Zhu20], shadow network and error imputation [Pav20], Convolutional Neural Networks (CNNs) [Zhu18], and Recurrent Neural Networks (RNNs) [Mar18; Che18; Cap20; May20; Ham20; Bev20; Dan20; Bhi20; Fre20; Yan20; Far19; Aie20; Fox18].

Hybrid models are usually designed to replace or add features simulated by the physiological model in data-driven algorithms. In [Ber18], Bertachi et al. used a mathematical model to construct additional features (insulin on board, carbohydrate on board and activity on board) which are then used as inputs to artificial neural networks. Zecchin et al. [Zec14] considered the carbohydrates absorption model as an input feature in the jump neural network to improve predictive accuracy.

The main goal of this study is to design data-driven personalized BG predictive models for T1D that are computationally efficient, suitable for wearable devices, and perform well on common problems that arise in the analysis of real-world data (missing data, uncalibrated data) for insulin therapy. Here, four different types of models were developed, including deep learning algorithms and regression models, and their performance in predicting BG levels was compared within 30 or 60 minute forecast horizons using data from 12 individual T1D patients. The two deep learning models are Dilated Convolutional Neural Networks (DCNN) model and a Sequence to Sequence Long Short Term Memory (Seq-to-Seq LSTM) model. The two regression models are a Multiple Linear Regression (MLR) model and a Bidirectional Reservoir Computing (BRC) model.

This study has two main contributions: i) Models are trained and tested on a large-scale, multiple features dataset consisting of 12 T1D data contributors. A detailed discussion of 30 and 60 minute time horizon prediction results is provided. ii) To the best of our knowledge, the work presented here is the first to show that a model based on linear regression can outperform state of the art deep learning models. To this end, a novel method of constructing a multi-step linear regression prediction model is developed.

The remainder of the chapter is organized as follows. Section 2.2 presents the current state-of-the-art of data-driven methods specific for the BG level prediction task. Section 2.3 describes the dataset used to evaluate the models. In Section 2.4, the four data-driven models are introduced.

Section 2.5 presents the results followed by a discussion of the model behavior in Section 2.6. Section 2.7 provides conclusion remarks and suggests future research directions.

## 2.2 Related Works

Many data-driven models have been tested for near-time BG level forecasts in the diabetes research community. In the category of regression models, the AR model proposed in [Rei07] provides BG predictions as a linear combination of previously observed signals. A first-order AR model was proposed in [Spa07] using time-varying parameters and was evaluated in 30 and 45 minute prediction horizons.

Several researchers compared the AR model with other predictive models. The recursive AR with exogenous input (ARX) model [Xie18] outperformed a vanilla LSTM, temporal convolution network, and other regression models for the prediction of BG levels from CGM data. However, in the ARIMA model [RR19], characterizing the movement of time series using an AR integrated (ARI) and moving average (MA) strategy resulted in worse performance than a random forest model. In addition, the linear model in [Bev20] resulted in poorer performance in accuracy metrics compared to LSTM networks.

Recent work also included the combination of regression models and neural networks. An AR Time Delayed (ARTiDe) jumps neural network model [Ant20] added the feedback connections from the output layer to the hidden layer and time delays to each connection on an AR model. In [Nem20], the stacked regression contained a Multilayer Perceptron (MLP), an LSTM, and a Partial Least Squares Regression model (PLSR), and used predictions from three models to train a second-level model. A similar stack model [Kha20] was tested on a multi-lag system. Finally, a Residual Compensation Network (RCN) combined with an online ARMA model (RCN-ARMA) [Ma20] was used for simulating the nonlinear trends in BG dynamics.

However, the AR models typically use only CGM data to compute the prediction recursively, and the errors may accumulate in the prediction phase. The utilization of heterogeneous features should be considered in the regression models [Geo12]. In addition, the regression models sometimes do not outperform the deep learning algorithms in predicting chaotic time series tasks.

Developing models that can utilize and accurately predict heterogeneous sources of time series data is still an active area of research. In the framework of deep learning approaches, there are a few related studies that address the use of LSTM layers for BG predictions to avoid the gradient vanishing problem of vanilla RNNs [Hoc97]. In [Mar18], a BG prediction model was trained with only CGM data using multiple LSTM layers and a fully connected output layer. In [Far19], three LSTM networks trained with a transfer learning strategy were evaluated on the Clark error grid [Cla87]. In [Aie20], the past observations and estimated future values provided inputs for two separate LSTM networks, which were then combined using a fully connected layer. The multiple lag strategy was applied to multi-scale LSTM [Yan20] so the network can more effectively model the dependence between features and CGM data. A four-layer bidirectional LSTM was introduced in [Cap20] which contained

a look back period of time. To improve prediction accuracy, the CGM was transferred to a one-hot encoder vector in the LSTM classification model in [May20]. Additionally, the encoder and decoder strategy was first tested in [Fox18] for BG predictions, where Fox et al. developed multi-output deep learning architectures by representing the inputs to a shared layer and then learning through the use of output networks. Several sequence to sequence neural networks were tested in [Bhi20] with LSTM or convolutional layers.

Selecting an artificial neural network and determining the model hyperparameters are crucial steps that can affect the evaluation performance, but can also be time-consuming. For a large neural network model, it can result in a slow training speed and a large combination of hyperparameters to tune. This study aims to develop computational efficient models that minimize the tuning effort and training time.

## 2.3 Data Processing

**Available data.** The OhioT1DM Dataset [Marp ] was initially provided to test the accuracy of models for the BG prediction task in the Blood Glucose Level Prediction (BGLP) Challenge in the third and fifth International Workshop on Knowledge Discovery in Healthcare Data. Each challenge released information from 6 data contributors, after the competition these data were publicly available. All 12 anonymous data contributors (labeled with numbers such as 540, 544, 563, etc.) have a total of around eight weeks of measurements from CGM, insulin therapy, physiological sensors, self-reported events for smartphone applications, with approximately 10 days of data that were held out for out-of-sample testing purposes; the date, month, and year were shifted randomly into the future time to anonymize the data.

The data includes 7 males and 5 females between ages 20 and 60. The constantly worn insulin pump (Medtronic 530G or 630G) measured the data contributors' BG levels every 5 minutes and the bolus and basal insulin dose injection amount; a smartphone recorded self-reported events including carbohydrate consumption, sleep quality, illness level, stress level, hypo-event, work intensity, and exercise duration; and fitness bands (Basis Peak or Empatica Embrace) collected heart rate, room temperature, skin temperature, moving acceleration, step count, and galvanic skin response (GSR). The CGM calibration is carried out every 6-8 hours per day by self-measured fingerstick values.

**Data time grid.** The time grid is uniformly arranged with 5 minute intervals between each time point for all features since the CGM data are collected every 5 minutes. The other features including carbohydrate intake, skin temperature, acceleration, GSR, fingerstick values, and bolus insulin dose are moved to the nearest time grid point when there exist misaligned measurements with the CGM signals. For categorical variables such as illness and hypo-events, event occurrences are represented by an integer value equal to 1 after moving to the corresponding time grid. The interval variables for which features are collected in a continuous period, such as basal insulin dose, exercise duration, and work intensity values, are filled in for the time range where data contributors had the activities.

**Data imputation.** Linear interpolation for continuous variables is used to connect the endpoints of intervals with missing values in the training data. In the test set, the CGM measurements with large missing intervals (over 6 hours) are excluded, and forward filling is used to impute the rest of the missing data, i.e., by carrying forward the last observation with an existing value. For the categorical variables and interval variables, the missing values are replaced with 0 for both training and testing data.

**Features selection.** The features with small incidence (the ratio of number of events to the total length of the dataset is $< 1\%$) were removed for the data contributor when training the predictive models. If a categorical feature is selected for training purposes, but completely missing in the testing set, then the feature will be added in the testing by setting all elements equal to 0. For example, if illnesses are reported in the training set but not in the testing set, we assume the contributor does not get ill in the testing period and set the corresponding categorical variable equal to 0. The four data-driven models are personalized for each individual, thus each data contributor may have a different number of features, the list of specific features used in each data contributor for all data-driven models is in Table 2.1. For a data contributor, the same features were applied to every model. The common features are CGM, fingerstick values, basal and bolus insulin dose, carbohydrate consumption, GSR, and skin temperature.

**Data scaling.** The range of values varies extensively for each feature. Data were normalized by subtracting the mean and dividing by the standard deviation.

**Data splitting.** To observe the behavior and save the best performing model, validation data is necessary. The training set is further separated by using the last 20% of time points for validation. Thus, for the average training duration of 40 days, the last 8 days are used for validation. If a participant has a narrower range of observations, the length of training and validation sets are proportionally decreased. Table 2.2 shows the average length of time-series data and the missing percentage of continuous variables for training, validation and testing sets over the 12 data contributors.

**Data expansion.** To increase the number of training samples, the multi-horizon method is introduced. Previous work [Tai16] suggested that the multi-horizon strategy is accurate for short time series forecasts. It can also control overfitting and is able to recover potential covariate relationships. The training data of each contributor is truncated in a sliding window, moving 1 time-step forward with a fixed window size. In this study, the fixed window size is determined by the input time length of the bidirectional reservoir computing and deep learning models. For the multiple linear regression model, the training data is combined with validation data in the original size.

## 2.4 Methods

The four data-driven models are presented, including two regression models and two deep neural network models.

**Table 2.1** Training features used by the 12 data contributors (dc) for the proposed models. The first column denotes the number of data contributors for which models were developed using the features in the third column. The second column denotes the number of features in the third column. The table is presented from the most features to the least features.

| #dc | #features | Feature list |
|---|---|---|
| 1 | 14 | CGM, fingerstick, basal and bolus insulin dose, carbohydrate consumption, GSR, skin temperature, sleep quality, illness, work intensity, exercise duration, heart rate, room temperature, steps |
| 3 | 13 | CGM, fingerstick, basal and bolus insulin dose, carbohydrate consumption, GSR, skin temperature, sleep quality, work intensity, exercise duration, heart rate, room temperature, steps |
| 1 | 13 | CGM, fingerstick, basal and bolus insulin dose, carbohydrate consumption, GSR, skin temperature, sleep quality, work intensity, hypo-event, heart rate, room temperature, steps |
| 1 | 13 | CGM, fingerstick, basal and bolus insulin dose, carbohydrate consumption, GSR, skin temperature, sleep quality, illness, exercise duration, heart rate, room temperature, steps |
| 2 | 9 | CGM, fingerstick, basal and bolus insulin dose, carbohydrate consumption, GSR, skin temperature, acceleration |
| 4 | 8 | CGM, fingerstick, basal and bolus insulin dose, carbohydrate consumption, GSR, skin temperature |

### 2.4.1 Multiple Linear Regression Model

Due to the structure of linear regression, fitting three-dimensional data adds difficulty in solving the inverse problem. The data expansion is omitted, and the validation set is unnecessary since there are no hyperparameters to tune. To train a model offline and compare it with the other three models described in the following section, we used the combination of the multi-step-ahead strategy to generate multi-horizon predictions as shown in Figure 2.1.

The Multiple Linear Regression (MLR) model is constructed by combining multiple linear regression models. In this approach, $k$ separate linear regression models are used for $k$ time step ahead predictions. Each linear regression model is trained to find the relationship between training series $X_{train}(q_1(t),\ldots,q_m(t))$ and target vector $\mathbf{y}_{train}(t+i)$, where $i=1,2,\ldots,k$, $X(\cdot)$ is the matrix containing $m$ number of feature values $q(t)$ at time $t$, where for a specific data contributor, the value $m$ can be found in Table 2.1, and $\mathbf{y}_{train}(t)$ represents the target CGM levels at time $t$.

For a linear model $L_i$, where $i=1,2,\ldots,6$ for the 30 minutes prediction task, and $i=1,2,\ldots,12$ for the 60 minutes prediction task, the relationship between coefficients $\beta^i$ and $X^i_{train}$, $\mathbf{y}^i_{train}$ for the i-th linear model can be expressed as follows:

$$\begin{bmatrix} x_1(q_1) & x_1(q_2) & \ldots & x_1(q_m) \\ x_2(q_1) & x_2(q_2) & \ldots & x_2(q_m) \\ \vdots & \vdots & \ddots & \vdots \\ x_n(q_1) & x_n(q_2) & \ldots & x_n(q_m) \end{bmatrix} \begin{bmatrix} \beta^i_1 \\ \beta^i_2 \\ \vdots \\ \beta^i_m \end{bmatrix} + \beta^i_0 = \begin{bmatrix} y_{i+1} \\ y_{i+2} \\ \vdots \\ y_{i+n} \end{bmatrix}, \tag{2.1}$$

**Table 2.2** Average length and average percentage of the missing data for continuous variables for training, validation and testing sets over 12 data contributors when the features are available. CGM represents the continuous glucose monitoring, HR stands for heart rate, GSR is galvanic skin response, and $T_{skin}$, $T_{room}$ stand for the skin temperature and room temperature, respectively. For continuous variables in the training and validation sets, the imputation method is linear interpolation. In the testing set, the forward filling method is used for imputing missing data and the large missing intervals of CGM are removed.

| | Average Length | Average percentage of missing data | | | | |
|---|---|---|---|---|---|---|
| | | CGM | Acceleration | HR | GSR, $T_{skin}$ | $T_{room}$ |
| Train | 10061.83 | 12.13% | 36.48% | 4.17% | 20.51% | 4.53% |
| Validation | 2515.17 | 11.67% | 44.85% | 7.70% | 26.47% | 8.08% |
| Test | 2896.83 | 8.63% | 30.78% | 6.75% | 18.94% | 7.09% |

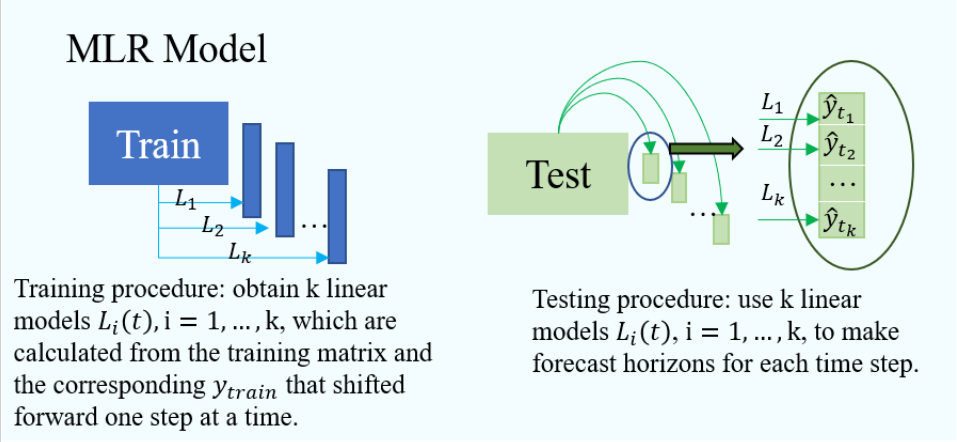


**Figure 2.1** Schematic diagram of the MLR model using multi-horizon prediction, where the $k = 6, 12$ represents the 30 or 60 minutes, respectively.

where the Equation (2.1) shows the expression of $X^i_{train}\beta^i = \mathbf{y}^i_{train}$, for a training set with total length of $n$, with $X^i_{train} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^T$, the training matrix will stay the same for all linear models' calculations. We have $\beta^i = [\beta^i_0, \beta^i_1, \ldots, \beta^i_m]^T$, and $\mathbf{y}^i_{train} = [y_{i+1}, y_{i+2}, \ldots, y_{i+n}]^T$.

After training and obtaining the coefficients and intercepts for $k$ linear regression models $(L_1, L_2, \ldots, L_k)$, predictions are computed by using $k$ linear models in the testing data. To obtain the first prediction vector with length $k$, the following system of equations (2.2) is used.

$$\begin{aligned}
\hat{y}_1 &= \beta^1_0 + x'_0(q_1)\beta^1_1 + x'_0(q_2)\beta^1_2 + \cdots + x'_0(q_m)\beta^1_m, \\
\hat{y}_2 &= \beta^2_0 + x'_0(q_1)\beta^2_1 + x'_0(q_2)\beta^2_2 + \cdots + x'_0(q_m)\beta^2_m, \\
&\ldots \\
\hat{y}_k &= \beta^k_0 + x'_0(q_1)\beta^k_1 + x'_0(q_2)\beta^k_2 + \cdots + x'_0(q_m)\beta^k_m,
\end{aligned} \tag{2.2}$$

where the prediction vector is $[\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_k]$, and $x'$ represents the testing time series, $\hat{y}$ represents the testing target values, $\beta^i$ is the coefficient vector obtained from linear model $L_i$. Additionally, $\mathbf{x}'_0$ is the 12th element in the original testing series, so after excluding the first hour data in testing, it allows to use the complete training set and the 12th row of testing matrix $X_{test}(q_1(t_0), q_2(t_0), \ldots, q_m(t_0))$ to

generate the prediction horizons for the first 30 or 60 minutes predictions.

After obtained the first prediction vector, the next row of the testing matrix ($\mathbf{x}_1'$) is used to calculate the next prediction vector from $\hat{y}_2$ to $\hat{y}_{k+1}$.

$$\begin{aligned}
\hat{y}_2 &= \beta_0^1 + x_1'(q_1)\beta_1^1 + x_1'(q_2)\beta_2^1 + \cdots + x_1'(q_m)\beta_m^1, \\
\hat{y}_3 &= \beta_0^2 + x_1'(q_1)\beta_1^2 + x_1'(q_2)\beta_2^2 + \cdots + x_1'(q_m)\beta_m^2, \\
&\cdots \\
\hat{y}_{k+1} &= \beta_0^k + x_1'(q_1)\beta_1^k + x_1'(q_2)\beta_2^k + \cdots + x_1'(q_m)\beta_m^k,
\end{aligned} \tag{2.3}$$

This method continues to the end of the testing matrix until all future predictions in $k$ time step horizons are obtained. Then the performance is evaluated by comparing each prediction vector with the corresponding labels in that time range. Thus, the performance was obtained by evaluating the error in each prediction vector.

Next, for plotting the predicted time-series, the averaging method of the prediction at a time point $t$ is presented. For the 30 minutes prediction task, except the first 5 testing labels, i.e., $\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4, \hat{y}_5$, have a smaller number of predictions by MLR model, the rest testing labels should have 6 predictions obtained for the 6 linear models. For instance, $\hat{y}_6$ is first obtained from $x_0'$ using the 6th linear model ($L_6$), then obtained from $x_1'$ using the 5th linear model ($L_5$), and the last prediction of $\hat{y}_6$ is obtained from $x_5'$ using the 1st linear model ($L_1$). A similar procedure can describe 60 minutes prediction horizons.

### 2.4.2 Bidirectional Reservoir Computing model

Reservoir computing has much success in accurately predicting multivariate time-series data [Bia18]. The core of reservoir computing is to build a network to process the measurements in a system-independent fashion. The Bidirectional Reservoir Computing (BRC) model is constructed from [Bia18], which introduces the encoder-decoder framework that transforms the inputs by tuning model hyperparameters. The BRC model is constructed from three main components: a linear input layer, a bidirectional recurrent nonlinear reservoir network, and a linear output layer, as shown in Figure 2.2.

In the encoder part of the model, the inputs contain the list of features in Table 2.1 with a fixed window size for a data contributor and expand the dimension by bidirectional reservoir networks, then reduce the dimension though Principal Component Analysis (PCA) and train through a ridge regression model.

Assuming the input time-series was truncated in 3D shape with size $[m, n_{tps}, n_f]$, where $m$ is the length of the input time series after truncation, $n_{tps}$ is the number of time points for prediction (6 or 12), and $n_f$ represents the number of features. Depending on the number of internal units $n_{unit}$ (fixed at 100 in the study), the internal weight matrix $W_{res}$ with shape $[n_{unit}, n_{unit}]$ is randomly initialized from a sparse matrix with uniformly distributed nonzero entries in [-0.5,0.5] with a predefined density parameter. The input weight matrix $W_{in}$ with shape $[n_{unit}, n_f]$ is initialized from

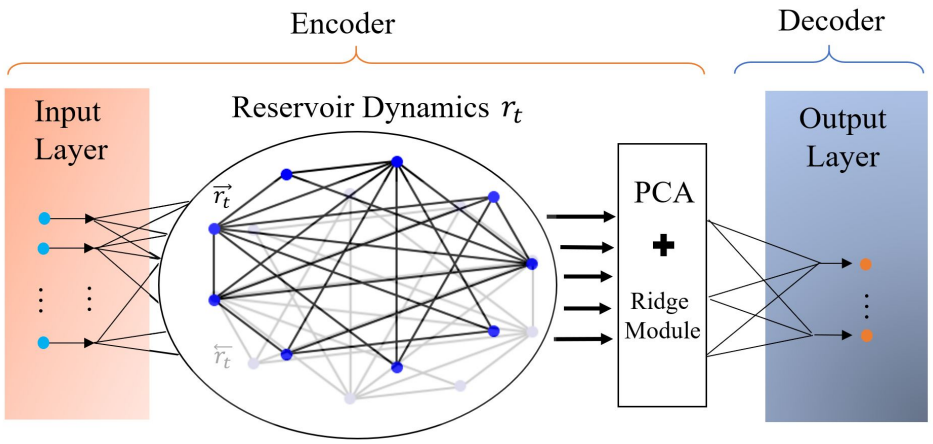


**Figure 2.2** Schematic diagram of the BRC model, $r_t$ is the reservoir state at time $t$, where $r_t$=[$\overleftarrow{r_t}$, $\overrightarrow{r_t}$].

a binomial distribution with the probability for success is 0.5 and the number of events is 1.

Next, we update the weight matrices $W_{res}$ and $W_{in}$ as defined below.

$$\begin{aligned} W^*_{res} &= \frac{W_{res}}{\frac{max(|\mathbf{e}|)}{\phi}}, \\ W^*_{in} &= s_{input}(2W_{in}-1), \end{aligned} \tag{2.4}$$

where **e** is the vector of eigenvalues of the previous $W_{res}$, $\phi$ represents the spectral radius. By setting the $\phi$ in the first equation close to 1, which is equivalent to the normalization of $W_{res}$ obtained from $W^*_{res} \approx \frac{W_{res}}{||W_{res}||}$. For the second equation in (2.4), $s_{input}$ is the input scale, then the $W^*_{in}$ would contain only two values: $s_{input}$ or $-s_{input}$.

The state matrix $r_t$ at time $t$ is combined with the two directional matrices $\overrightarrow{r_t}$, $\overleftarrow{r_t}$, where the forward direction $\overrightarrow{r_t}$ with shape $[m, n_{tps}, n_{unit}]$ is computed by completing the loop of $i$ from the first element ($i=1$) in the second dimension of $\overrightarrow{r}_t$ to the end ($i=n_{tps}$).

$$\begin{aligned} p_t &= W^*_{res} \cdot r_p^T + W^*_{in} \cdot X_t(:,i,:)^T + c_{noise} B_{res}, \\ r_p &= (1-\gamma) r_p + \tanh(p_t), \\ \overrightarrow{r_t}(:,i,:) &= r_p, \end{aligned} \tag{2.5}$$

where $p_t$ stands for the state taking the input information $X_t$ at time $t$, $c_{noise}$ is the noise level and $B_{res}$ represents the noise matrix. The product $c_{noise} B_{res}$ is similar to the bias in other deep learning frameworks. In the second equation, $\gamma$ is a scalar that determines what proportion we want to update by the previous state $r_p$, where $r_p$ was initialized as a zero matrix at the begining of the loop.

The backward direction state matrix $\overleftarrow{r_t}$ was computed using the same method as described in Equation (2.5) with the $X_t$ flipped in the second dimension. Then the dimension of state matrix $r_t = [\overrightarrow{r_t}, \overleftarrow{r_t}]$ reshaped to 2D with size $[m \times n_{tps}, 2n_{unit}]$ and reduced by PCA. PCA utilized the

advantage of Singular Value Decomposition (SVD) and output the key components by selecting the number of singular values in the SVD to achieve dimension reduction. The number of PCA components $n_{PC}$ is selected for choosing the number of large singular values of the centered matrix $\bar{r}_t$ (obtained by subtracting the mean value of each column of the reshaped matrix $r_t$) based on the requirement of large or small size reservoir dynamics. The reduced state matrix $r_t^*$ can be expressed as follows:

$$\begin{aligned} r_t^* &= U\Sigma V^T \\ &= \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n_{PC}} \end{bmatrix} \begin{bmatrix} s_1 & 0 & \dots & 0 \\ 0 & s_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & s_{n_{PC}} \end{bmatrix} \begin{bmatrix} v_1^T & v_2^T & \dots & v_{n_{PC}}^T \end{bmatrix}, \end{aligned} \tag{2.6}$$

where $u_i$ and $v_i$ represent the row vector for the singular matrix $U$ and $V$, respectively, and $s_i$ represents the singular values in the diagonal matrix $\Sigma$ with $s_1 > s_2 > \cdots > s_{n_{PC}} > 0$. After reshaping the reduced state matrix back to 3D, the size of the state matrix is reduced from $[m, n_{tps}, 2n_{unit}]$ to $[m, n_{tps}, n_{PC}]$. Next, the ridge module used the ridge regression and compute the intercepts and coefficients between $r_t^*(j, 1 : n_{tps} - 1, :)$ and $r_t^*(j, 2 : n_{tps}, :)$ for each $j = 1, 2, 3, \dots, m$, and append to the input representation matrix $R_{repr}$ as a row vector.

For the decoder part of the model, ridge regression reads the input representation matrix from the encoder and transfers the information back to obtain multiple prediction horizons. By setting a large ridge regression parameter $\alpha$ in Equation (2.7), the weights are balanced to avoid overfitting.

Ridge regression computes the weights $w$ by finding the minimum solution for the input $X$ and output $\mathbf{y}$ using the following cost function:

$$\min_w \left( ||\mathbf{y} - X\mathbf{w}||_2^2 + \alpha ||\mathbf{w}||_2^2 \right), \tag{2.7}$$

where $X$ and $\mathbf{y}$ correspond with the reshaped matrices after the PCA dimensionality reduction in the representation module. For ridge regression in the decoder, $X$ is the output from encoder $R_{repr}$, and $\mathbf{y}$ represents CGM signals shifted 30 or 60 minutes into the future.

The computational training and testing times depend on the size of the reservoir and input time steps. The validation data is used to tune the number of input time steps and the hyperparameters (i.e., $\gamma$, spectral radius, input scaling) for the reservoir dynamics to obtain both the efficiency and accuracy. The details of fixed model parameters are listed in Table 2.3.

### 2.4.3 Dilated Convolutional Neural Network Model

WaveNet [Oor16], a deep neural network that has been used to accurately generate audio waveforms, relies on a type of neural network layer called a dilated convolutional layer. This dilated convolution layer is used here to build a dilated convolutional neural network (DCNN) model. The model size is largely reduced and there are fewer hyperparameters to tune. The DCNN model is built from

**Table 2.3** The fixed parameters in reservoir computing for individual data contributors.

| | |
|---|---|
| Input time steps | 29 |
| Number of PCA components | 50 |
| Number of internal units in the reservoir | 100 |
| Noise level | 0.01 |
| Output time steps | 6 or 12 |

multiple DCNN layers and two time-distributed dense layers.

**DCNN layer.** The DCNN model uses the strategy of combining dilated convolutions and causal padding, where the dilated convolutions [Yu16] let the dilation rate increase multiplicatively at each layer to achieve an exponential relationship between receptive field size and the layer number.

Let $F : \mathbb{Z}^2 \to \mathbb{R}$ be a discrete function. Let $\Omega_r = [-r, r]^2 \cap \mathbb{Z}^2$ and let $K : \Omega_r \to \mathbb{R}$ be a discrete filter size $(2r+1)^2$. The discrete convolution operator $*$ is given below,

$$(F * K)(p) = \sum_{s+t=p} F(s)K(t). \tag{2.8}$$

Assume $d$ is a dilation factor, the dilated convolutions $*_d$ can be expressed below,

$$(F *_d K)(p) = \sum_{s+d\cdot t=p} F(s)K(t), \tag{2.9}$$

where the area of the receptive field $p$ is expanded by increasing the dilation factor $d$ and fixing $s$ and $t$.

In the Figure 2.3, an example is presented to visualize the advantage of the dilated convolution. In the normal convolution (or when $d = 1$ in the dilate convolution), when a $3 \times 3$ discrete filter $K$ is chosen for a receptive field, each element of the output (top red blocks) was mapped from the receptive field with the same size of the filter. For the dilated convolution ($d = 2$), each element of the output was mapped from a larger receptive field (size $5 \times 5$), while every second column and row (green squares) are ignored.

Then the casual padding is used to avoid using future values, where $c-1$ number of 0 are padding in front of the first neuron in the hidden layer for a $c$ kernel size. In the Figure 2.4, it shows an example of causal padding with kernel size $c = 3$, where the empty neuron (represents the value 0) is padding in front of the first blue neuron. Then the first orange neuron (labeled with 2) in the second hidden layer takes the kernel size of values from the previous hidden layer (2 empty neurons and a blue neuron labeled 1), and the usage of future values (second, third and fourth blue neurons) can be avoided.

Combining those strategies, the DCNN layers can quickly expand the receptive field while avoiding using the future knowledge. As it is shown in Figure 2.5, the information contained in blue neurons in the first hidden layers can be received by the first orange neuron fourth layer by

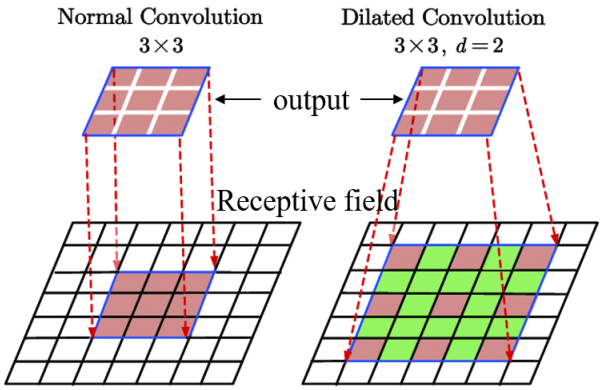


**Figure 2.3** Schematic of the normal convolution mapping and dilated convolution mapping with dilation factor $d = 2$ and filter size $3 \times 3$ in [Du20].

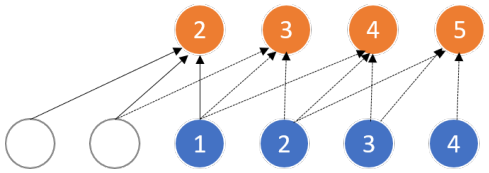


**Figure 2.4** Schematic of the causal padding with kernel size $c = 3$ and number of neurons is 4, where the bottom blue neurons represent the first hidden layer, and the orange neurons are in the second hidden layer. The line arrow presents an example of the information passes to the first neuron in the second hidden layer.

expanding the dilated factor in each layer.

**Time-distributed layer.** A dense layer calculates the output, $o$, using an activation function $f$ involving weights $W$ and biases $\mathbf{b}$, denoted by $o = f(WX + \mathbf{b})$, where $X$ is the input. Since separating each time step for time series is necessary, a time-distributed layer is applied to each dense layer for each time step.

As depicted in Figure 2.6, the multiple DCNN layers (kernel size = 2) with causal padding are used and $L_2$ regularization is applied to train the weights in each layer. In the first time-distributed layer, the ReLu activation function is used with $L_2$ regularization. Then, a dropout layer is used to scale down the output, which is then input to the next time distributed layer. The second time distributed

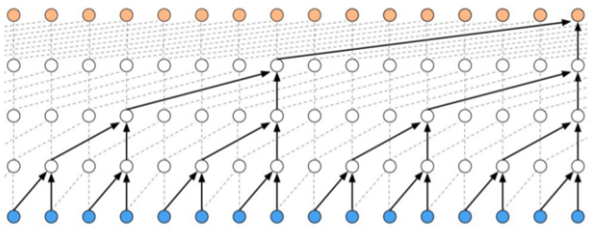

**Figure 2.5** Schematic of a 4 layers DCNN model with dilated factor $d = [1,2,4,8]$ for each layer from [Oor16].

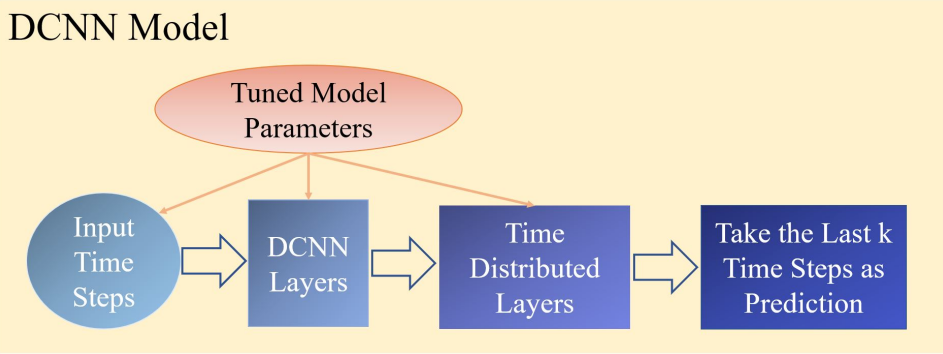


**Figure 2.6** Schematic diagram of the DCNN model.

layer is used to compute the final output using a linear function with $L_2$ regularization. The last 6 or 12 time steps are selected for computing predictions over 30 and 60 minute time horizons, respectively. The input time series for one data contributor has the features presented in Table 2.1, the corresponding target time series are the CGM signals shifted either 30 or 60 minutes into the future.

The model hyperparameters including input time steps (selected from [72, 144, 288, 576]), the number of DCNN layers (selected from [2, 3, 4]), and the number of neurons in the hidden layers (selected from [16, 32]) are tuned by using grid search to determine the best combination of parameters of each data contributor. The hyperparameters are selected with the best RMSE score on the validation set and applied into the training process with the training dataset. During training, the Adam optimizer with a 0.01 learning rate is used based on the best performance in the grid search procedure for the selected combination of hyperparameters. The loss function used in both training and validation is defined as follows.

**Loss function.** The Weighted Mean Squared Error (WMSE) is calculated by the following formula,

$$WMSE = \sum_{i=1}^{N} \Big( (\hat{y}_i - y_i)^2 \times \frac{y_i}{M} \Big), \tag{2.10}$$

where $y_i$ presents the true value and $\hat{y}_i$ stands for the predicted value at time $i \in [0, N]$. $M$ is a fixed scalar, usually selected slightly larger than the maximum of the data values. Compared to the mean squared error loss function, WMSE focuses on amplifying the error in mispredicting hyperglycemia, which is usually underestimated in the training phase.

### 2.4.4 Sequence to Sequence Long Short Term Memory Model

Sequence to sequence (Seq-to-Seq) models [Sut14] use an encoder-decoder framework that can map sequences of different lengths from input to output.

An LSTM layer is a recurrent neural network layer that contains a chain of repeating modules (Figure 2.7) , each module contains three interacting gates: a forget gate $f_t$, input gate $i_t$, and output

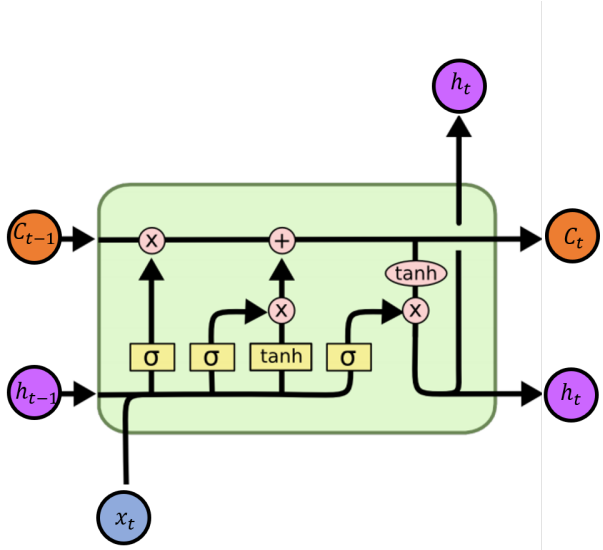


**Figure 2.7** The schematic diagram of a single LSTM module from [Hoc97].

gate $o_t$. The network in the LSTM layer can be explained as follows. The information from the previous state is first computed by the sigmoid function (denoted by $\sigma$) to determine which part to forget (forget state), then a second sigmoid function determines the information to carry on (input gate), and lastly, the output information obtained by the output gate from the third sigmoid function. The internal memory $C_t$ combines the input state and drops the forgotten information from the old cell $\tilde{C}_t$, and the hidden state $h_t$ is updated through an hyperbolic tangent activation function that is applied to a combined vector consisting of the internal memory and the information from the output gate. The mathematical expressions for these terms can be found in the following equations,

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \\ \tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C), \\ C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\ h_t &= o_t \odot \tanh(C_t), \end{aligned} \tag{2.11}$$

where $\cdot$ represents matrix multiplication and $\odot$ stands for Hadamard product (element-wise product).

The Seq-to-Seq LSTM model applies the encoder-decoder strategy using LSTM layers. In the encoder model, the encoder LSTM takes the input source (training time-series) and outputs the hidden state $h_t$ and internal memory $C_t$ at a time $t$ as an encoded vector. Then the encoded vector becomes initial states in the decoder LSTM. For the decoder model, the decoder LSTM also takes a mock target as the decoder input, and produces the output and state vector (hidden state and

internal memory). Last, the linear dense layer is applied to the decoder outputs for completing the decoder model. The schematic diagram is shown in Figure 2.8.

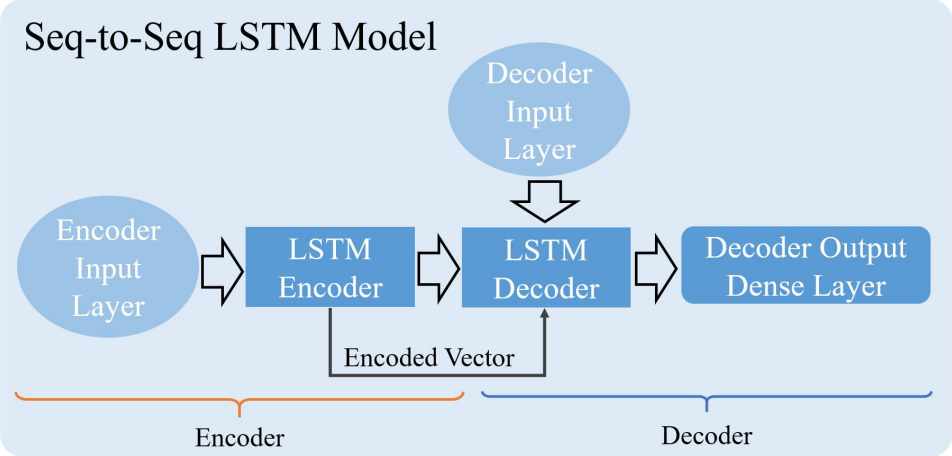


**Figure 2.8** Schematic diagram of the Seq-to-Seq LSTM model.

A window of 6 hours with the training features listed in Table 2.1 is provided for the input series, and the decoder input series is the CGM signals shifted forward by one time step with zero padding used in the first time step. For time $t$, the decoder input series which works as a mock target contains $[0, y(t), y(t+1), \ldots, y(t+k-1)]$, and the target decoder output is $[y(t), y(t+1), y(t+2), \ldots, y(t+k)]$, for the time-series $y$ representing the future 30 or 60 minutes of CGM signals at time $t$ from the training and validation. In the predicting phase, the testing time series is taken by the encoder model with one time step at a time. Then the encoder model outputs the state vector which becomes the input state for the decoder model. The decoder model iteratively computes each element in the prediction vector by updating from the previous predicted value and output states. The model uses an LSTM layer with 128 recurrent units for both the encoder and decoder and summarizes the output through a dense layer. The behavior of the validation set is monitored by the mean squared error and the training is stopped if the loss function is stable after 8 epochs. The best performing model is applied to the time series data in the test set to obtain predictions.

### 2.4.5 Evaluation Metrics

The predictive accuracy for BG levels in training, validation, and test data are evaluated by RMSE and MAE defined below. During the evaluation, the imputed value of CGM in training, validation and test data are excluded. In the test data, the first hour is excluded for utilizing the full length of

training and validation data in the training procedure.

$$RMSE = \sqrt{1/N \sum_{i=1}^{N} (\hat{y}_i - y_i)^2}, \tag{2.12}$$

$$MAE = 1/N \sum_{i=1}^{N} |\hat{y}_i - y_i|, \tag{2.13}$$

where the $\hat{y}_i$, $y_i$ are the predicted value and true value for time index $i$, respectively. $N$ is the total number of data points evaluated. We note that the RMSE is more sensitive to outliers than the MAE.

## 2.5 Results

The evaluation tables for 30 and 60 minute forecast horizons are shown in Tables 2.4 and 2.5 for four data-driven models. All results are averaged over all data contributors. We observe that the Seq-to-Seq LSTM model performed the best in 30 minute ahead predictions and the MLR model performed the best in 60 minute ahead predictions. An example of a prediction plot for each model, using the testing set from one data contributor, is depicted in Figures 2.9 and 2.10.

**Table 2.4** Table of average RMSE and MAE evaluations within a 30 minute forecast horizon for 4 proposed data-driven models. The best testing performance is labeled in bold.

| | RMSE (mg/dl) | | | MAE (mg/dl) | | |
|---|---|---|---|---|---|---|
| Model | Train | Validation | Test | Train | Validation | Test |
| MLR | 14.16 | - | 18.39 | 9.87 | - | 11.76 |
| BRC | 10.82 | 19.29 | 24.49 | 7.74 | 13.50 | 17.43 |
| DCNN | 21.41 | 23.55 | 26.96 | 15.60 | 16.76 | 18.25 |
| Seq-to-Seq LSTM | 10.02 | 13.83 | **17.52** | 7.11 | 9.26 | **10.91** |

**Table 2.5** Table of average RMSE and MAE evaluations within a 60 minute forecast horizon for 4 proposed data-driven models. The best testing performance is labeled in bold.

| | RMSE (mg/dl) | | | MAE (mg/dl) | | |
|---|---|---|---|---|---|---|
| Model | Train | Validation | Test | Train | Validation | Test |
| MLR | 21.39 | - | **24.58** | 15.56 | - | **17.42** |
| BRC | 13.95 | 24.93 | 28.79 | 10.12 | 18.10 | 20.75 |
| DCNN | 35.84 | 36.98 | 38.97 | 27.37 | 27.96 | 29.38 |
| Seq-to-Seq LSTM | 18.22 | 22.40 | 26.44 | 13.96 | 16.54 | 18.43 |

The two evaluation metrics (RMSE and MAE) show a large difference among the four data-driven

models and the shift operator, likely reflecting the fact that CGM signals have large time-varying fluctuations. We found that the four data-driven models had difficulty capturing the extreme values in the training, validation, and testing sets. Additionally, we observed that the 60 minute time horizon predictions were less likely to capture the extreme values compared to the 30 minute predictions, especially when using the deep learning models. The DCNN model is trained on a weighted loss function, which may lead to more fluctuations in the predictions when the CGM signal is smooth.

Two encoder-decoder models were trained, namely the BRC and Seq-to-Seq LSTM models. Compared with other models in the same category (regression or deep learning), the model with encoder-decoder framework has more variance than the input CGM features, while the MLR and DCNN mainly follow the original CGM inputs with variations due to the model structure. With regards to the ability to accurately predict missing data, we found that the encoder-decoder models produced predictions that were similar to the fluctuating trends around the missing regions, whereas the other two models stayed flat especially in 60 minutes prediction (see Figures 2.9 and 2.10 with x-axis labeled with the red triangles around day Jan. 21).

Comparing computational efficiency, the DCNN, Seq-to-Seq LSTM, BRC, and MLR models took an average of 30 minutes, 10 minutes, 20 seconds, and 3 seconds for training each data contributor, respectively. These results strongly suggest that the MLR model is significantly faster to train than deep learning models for the complex multivariate time series data considered in this work.

The differences in performance in the training, validation, and testing sets for the four data-driven models may have resulted from overfitting. In practice, the training performance should always be better than performance on the validation or testing sets. Due to the different imputation methods in the validation and testing sets, the evaluation results indicate that the imputation method may affect the prediction performance according to the model type. For the BRC model, the training error is significantly lower than validation and testing error. Even with setting the ridge regulation parameter $\alpha$ in Equation (2.7) to be a large number (e.g., $10^2$), the results are not optimal. However, the training performance of the BRC model is similar to the Seq-to-Seq LSTM model for 30 minute predictions and outperforms all other models in 60 minute predictions, which demonstrates the potential of the BRC model in time-series forecasting tasks.

## 2.6 Discussion

We compared the accuracy of our models with previous models that were developed for the BGLP challenge on the Ohio T1DM dataset. Table 2.6 shows results comparing our models to models in the first BGLP challenge from the Third International Workshop on Knowledge Discovery in Healthcare, which used data from the first six data contributors in the OhioT1DM dataset and report only RMSEs for 30 minute prediction horizons. Table 2.7 shows results compared to models in the second BGLP challenge from the Fifth International Workshop on Knowledge Discovery in Healthcare, which used data from the remaining six data contributors in the OhioT1DM dataset and report the RMSEs and MAEs for 30 and 60 minute prediction horizons. Due to the limited knowledge of the imputation

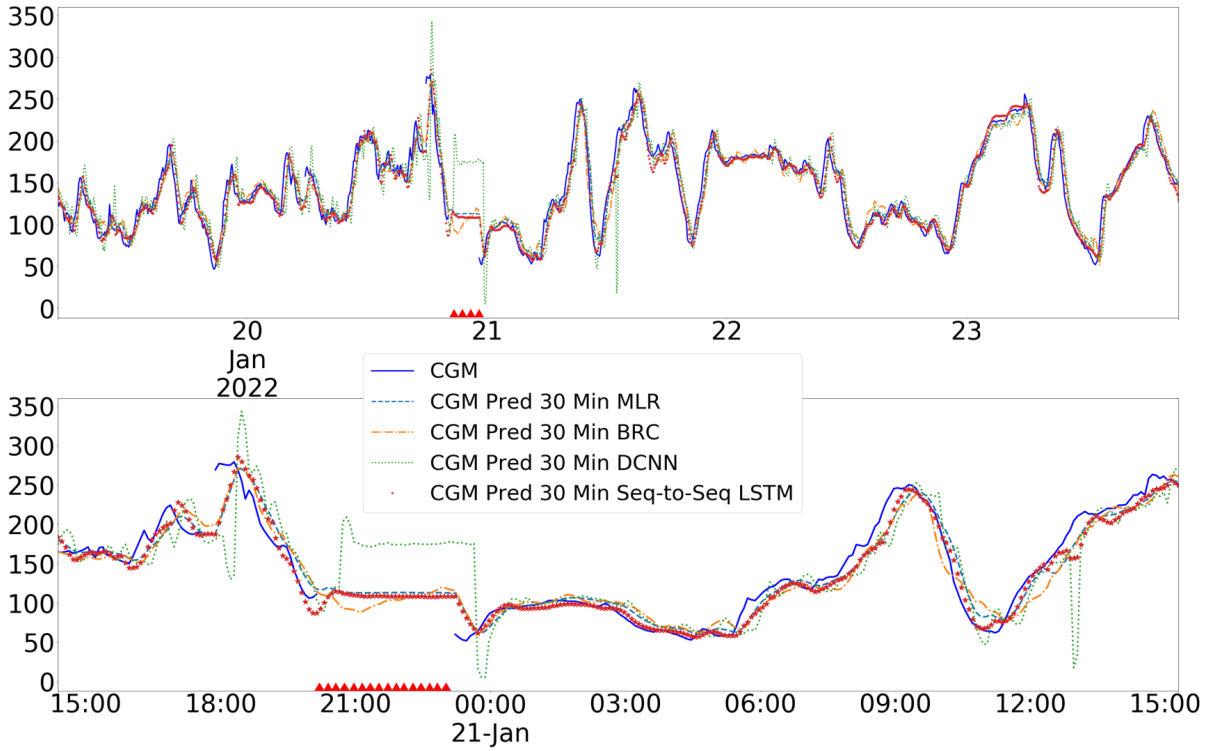


**Figure 2.9** Comparison of forecasting plots for the four proposed models using the testing set of data contributor 591 in 30 minutes multi-horizon predictions. The blue curve represents CGM signals, cyan dashed line represents CGM prediction for MLR model, orange dash-dot line stands for BRC model, green dotted line is for DCNN model, and red star is the Seq-to-Seq LSTM. The red triangles in the x-axis denote the date around Jan. 21 where data for CGM is missing. The bottom subplot is a zoom that displays the time around the missing data interval from Jan. 21.

methods from the literature, the mean value of the corresponding data contributors are compared only. Overall, the results in Tables 2.6 and 2.7 indicate that both our models and previously developed models show large differences in the RMSE and MAE metrics.

The third column in Table 2.6 indicates that our MLR and Seq-to-Seq LSTM models outperformed previously published models, and the fourth column shows that three of our models were more accurate at 60 minute predictions when the dataset does not have large missing intervals, as we found only two contributors had large missing intervals in the testing data among the 6 evaluated contributors. In Table 2.7, our MLR and Seq-to-Seq LSTM models outperformed previously published models for 60 minute predictions using the RMSE evaluation metric, and obtained lower MAEs compare to most models except NN-EIM [Pav20]. Among the 6 data contributors, five data contributors had large missing intervals in the testing set. The failure of obtaining better results in 30 minute predictions may be due to the data processing method we used for the testing data.

In the testing set, we skipped the missing intervals that are above 6 continuous hours of missing CGM signals, and imputed the rest of data by the forward filling. However, the imputation method will largely affect prediction performance. For example, in Figure 2.11, an interesting behavior was observed by the BRC model for one data contributor where the values around June 6th are

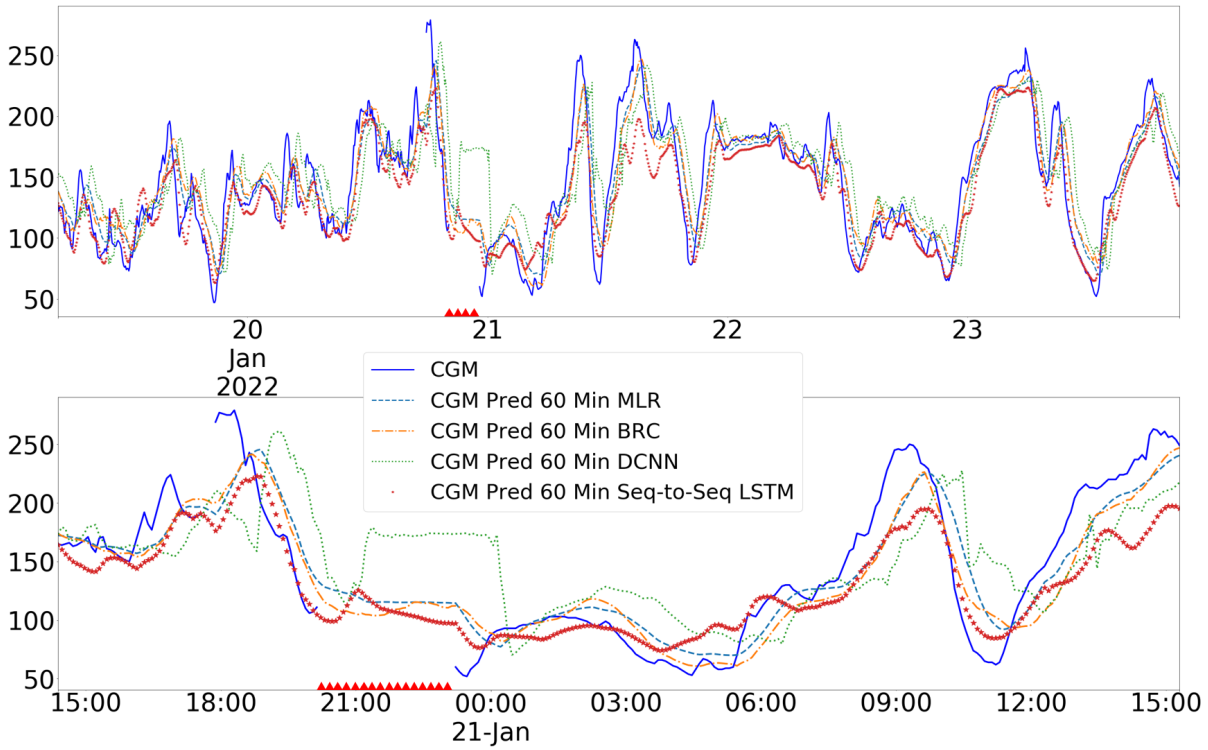


**Figure 2.10** Comparison of forecasting plots for the four proposed models using the testing set of data contributor 591 in 60 minutes multi-horizon predictions. The blue curve represents CGM signals, cyan dashed line represents CGM prediction for MLR model, orange dash-dot line stands for BRC model, green dotted line is for DCNN model, and red star is the Seq-to-Seq LSTM. The red triangles in the x-axis denote the date around Jan. 21 where data for CGM is missing. The bottom subplot is a zoom that displays the time around the missing data interval from Jan. 21.

largely missing. After the missing data, the BRC model starts to predict CGM levels incorrectly. After approximately one day, e.g., at approximately June 7th, the BRC model begins producing more accurate predictions.

The example in Figure 2.11 reveals two issues: first, we should not remove the missing interval. Second, the evaluation by RMSE, MAE, or other metrics may not detect the poor prediction behavior that occurs just after the missing data intervals. Additionally, the low RMSE may be due to the shifted behavior of CGM signals when fewer data points with fluctuations are presented [Zhu18]. Together, these results suggest that the evaluation metrics should always be combined with the visualization of prediction plots for accurate interpretation.

Among the 12 individual data contributors, we found that the performance was highly variable. The data contributors with lower fluctuations in CGM levels and fewer missing values tended to result in better performance in evaluation metrics for all models. For data contributors with a highly fluctuating pattern in CGM levels (perhaps due to an unstable lifestyle) that contained a large amount of missing values, the MLR model outperforms other models. Thus, for large-scale time-series data over a population of individuals, these results suggest it is necessary to test multiple models and that performance of a specific type of model may depend on data heterogeneities in

**Table 2.6** Comparison of results from the Third International Workshop on Knowledge Discovery in Healthcare Data. RMSE (mg/dl) is evaluated within 30 and 60 minute prediction horizons. If a study did not conduct the experiment for 60 minutes, then it is labeled with -. The methods and corresponding authors are presented in the order from the worst to best RMSEs for 30 minutes. We list the models considered in this work as "This study" for comparison. Bold font indicates instances in which our models showed a lower RMSE than previous models.

| Study | Method | RMSE 30 min | RMSE 60 min |
|---|---|---|---|
| Zhu et al. [Zhu18] | Causal CNN | 27.73 | - |
| Contreras et al. [Con18] | GE | 21.19 | 31.34 |
| Midroni et al. [Mid18] | XGBoost | 20.38 | - |
| Martinsson et al. [Mar18] | LSTM | 20.1 | 33.2 |
| Xie, J. and Wang, Q. [Xie18] | ARX | 19.59 | - |
| Bertachi et al. [Ber18] | Hybrid model | 19.33 | 31.72 |
| Chen et al. [Che18] | DRNN | 19.04 | - |
| This study | MLR | **15.46** | **21.59** |
| | BRC | 20.92 | **24.67** |
| | DCNN | 24.98 | 36.39 |
| | Seq-to-Seq LSTM | **14.87** | **23.45** |

data omission and levels of fluctuation.

Notably, we found that better prediction performance was achieved with regression models for short time series forecasts. Overall, our novel MLR model achieves a good performance in predicting a 30 and 60 minutes horizon from a single time point. The model utilizes multiple linear regressions to obtain predictions $k$ time steps in the future and works better than AR models in multi-horizon forecasts, since the latter may accumulate errors during calculations. The BRC model could more accurately capture the possible trend of future CGM levels during the forward filling intervals, and the prediction difference between 30 and 60 minutes is relatively small. Lu et al. [Bia18] note that the performance of reservoir computing models may depend on initial parameters and can sometimes be very effective.

### 2.6.1 Further Investigation

We further tested on a different imputation method, where we applied the polynomial extrapolation for all missing intervals in testing data. The polynomial extrapolation method imputes the missing intervals as follows. First, we find the index of the first missing value and use all former data points to fit coefficients of the polynomial. We then use the predicted value to impute the first missing point. Next, we find the next missing value and construct a new polynomial using all former data points, predicting and filling the missing value from the new polynomial. We repeat this process until all missing data are imputed. Since some of the missing data are over longer time intervals, the polynomial order is usually selected from 3 to 5.

Now comparing with the same data contributor as in Figure 2.11, we observe after the polynomial

**Table 2.7** Comparison of results from the Fifth International Workshop on Knowledge Discovery in Healthcare Data. RMSE (mg/dl) and MAE (mg/dl) are evaluated within 30 and 60 minute prediction horizons. The methods and corresponding authors are presented in the order from the worst to best RMSEs for 30 minutes. We list the models considered in this work as "This study" for comparison. Bold font indicates instances in which our models showed a lower RMSE than previous models.

| Study | Method | RMSE 30 min | MAE 30 min | RMSE 60 min | MAE 60 min |
|---|---|---|---|---|---|
| Bhimireddy et al. [Bhi20] | Seq-to-Seq BiLSTM | 21.8 | 15.0 | 35.0 | 25.0 |
| Cappon et al. [Cap20] | BLSTM | 20.2 | 14.74 | 34.19 | 25.98 |
| Joedicke et al.[Joe20] | GP | 20.13 | 14.05 | 32.18 | 23.58 |
| Ma et al. [Ma20] | RCN-ARMA | 20.03 | 14.52 | 34.89 | 26.41 |
| Daniels et al. [Dan20] | MTCRNN | 19.79 | 13.62 | 33.73 | 24.54 |
| Mayo et al. [May20] | LSTM-NN | 19.4 | 13.9 | 33.4 | 25.0 |
| Sun et al. [Sun20] | LV-based model | 19.37 | 13.76 | 32.59 | 24.64 |
| Hameed et al. [Ham20] | Vanilla RNN | 19.21 | 13.07 | 31.77 | 23.09 |
| Yang et al. [Yan20] | MS-LSTM | 19.05 | 13.50 | 32.03 | 23.83 |
| Khadem et al. [Kha20] | Multi-lag stacking | 19.01 | 13.73 | 33.37 | 24.98 |
| Nemat et al. [Nem20] | Stacked regression | 18.99 | 13.73 | 33.39 | 25.04 |
| Pavan et al. [Pav20] | NN-EIM | 18.63 | 10.08 | 32.27 | 17.69 |
| Zhu et. al. [Zhu20] | GAN | 18.34 | 13.37 | 32.21 | 24.20 |
| Bevan, R. and Coenen, F. [Bev20] | LSTM | 18.23 | 14.37 | 31.1 | 25.75 |
| Rubin-Falcone et al. [RF20] | N-BEATS | 18.22 | 12.83 | 31.66 | 23.60 |
| Freiburghaus et al. [Fre20] | CRNN | 17.45 | 11.22 | 33.67 | 23.25 |
| This study | MLR | 21.32 | 13.13 | **27.56** | 19.10 |
| | BRC | 28.06 | 19.15 | 32.92 | 23.80 |
| | DCNN | 28.93 | 19.41 | 41.24 | 31.28 |
| | Seq-to-Seq LSTM | 20.17 | 11.79 | **29.44** | 19.79 |

imputation, the prediction of the BRC model is consistent and obtained a better evaluation metric scores as shown in Figure 2.12.

The advantage of the BRC model is more obvious on large missing intervals. In Figure 2.13, we plot for the data contributor 544, the prediction comparison of one day BG level dynamics. Observing from the red dash line which represents the imputation of BG levels, we found there exists almost a whole day of missing data on June 28th. The polynomial extrapolation has limitations on imputing for a wide interval resulting in almost a straight line. However, the reservoir computing was able to recover the missing BG level dynamics, and all 4 prediction curves reflect the similar dynamics.

As the importance of regression models was addressed, we investigated the testing performance of regression models by RMSE for 90 and 120 minute predictions based on the improved behavior on the revised imputation method. For MLR, the average RMSEs are 32.85 mg/dl and 39.17 mg/dl for 90 and 120 minutes, respectively. For BRC, it outperformed MLR with average RMSEs of 30.93 mg/dl and 34.17 mg/dl, respectively. The BRC has the potential of predicting in longer horizons and can generate the CGM dynamics by learning the behavior from training in the large missing intervals.

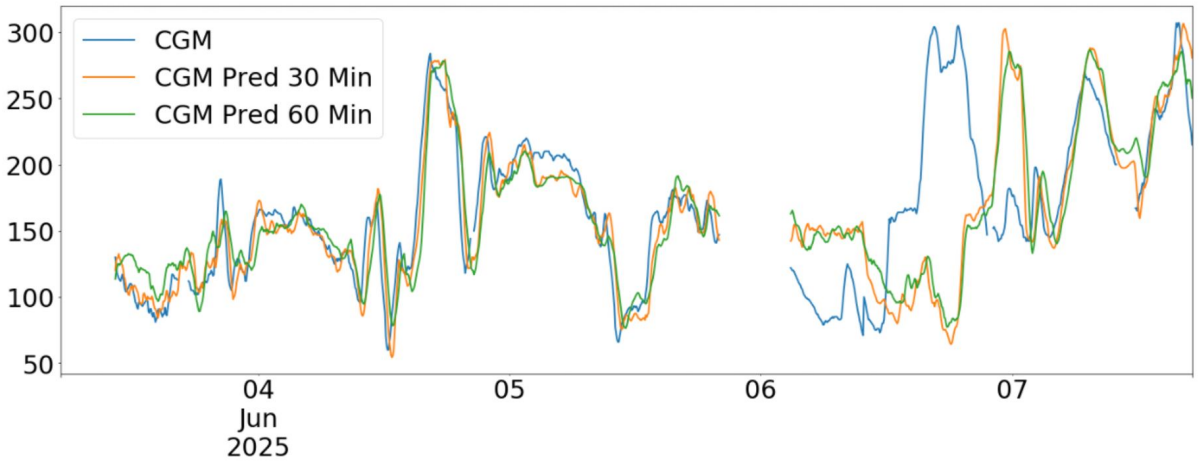


**Figure 2.11** Forecast plots for the BRC model using the testing set of data contributor 552. The blue curve represents CGM signals, orange curve is BG level prediction for a 30 minute time horizon, and green curve is the prediction for a 60 minute time horizon. Note that the date and time were randomly shifted to future date and time in the original OhioT1DM dataset. The missing interval around June 6th is removed when the prediction is calculated.

## 2.7 Contributions

In this work, four data-driven models using regression or deep learning were introduced to obtain multi-horizon predictions in patients with type 1 diabetes. Among these models, the Seq-to-Seq LSTM model performs the best at predicting BG levels within a 30 minute horizon and the MLR model is the best at predicting BG levels within a 60 minute horizon. These models could be used to develop an artificial pancreas that helps control BG concentrations in a closed-loop system. We presented an analysis based on the visualization of BG prediction plots, and found that the Seq-to-Seq LSTM and regression models have the potential to make highly accurate predictions. Further, the experiments were tested without removing the large missing intervals, the BRC model is significantly better than other models discussed in this study for the computational efficiency and prediction accuracy.

In future research, the classification problem of predicting hypo/hyper-glycemia could also be considered and compared with the performance of regression models [Wol19]. The proposed data-driven models can also be explored within the framework of hybrid models, i.e., hybridizing physiological models [Ber81; Mant ; Mann ] with the best performing data-driven models. The BG level forecasts produced by the computationally efficient models considered in this work may be added in a visualization tool for a closed-loop insulin delivery system for T1D patients [Des18], thereby improving the system interface and to encourage healthy activity and learning.

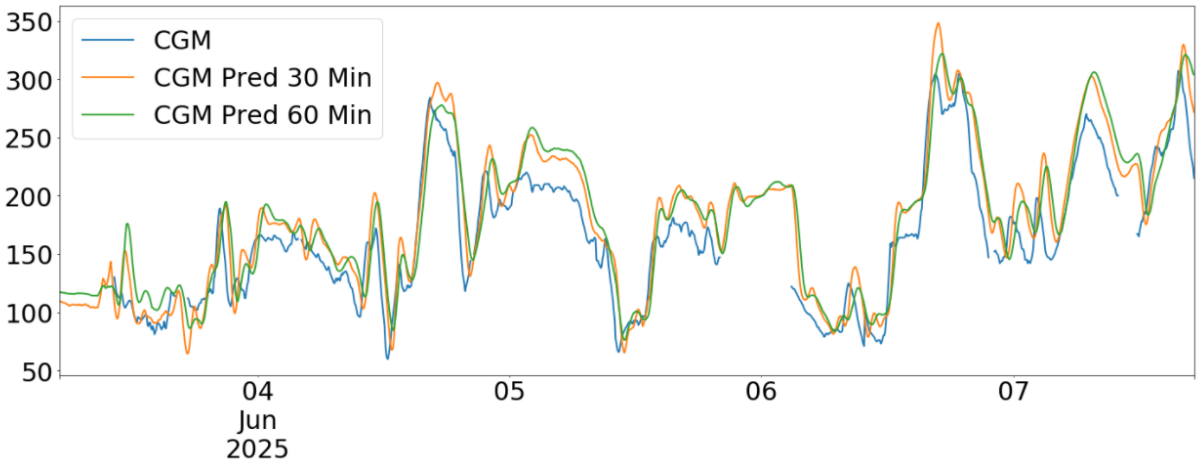


**Figure 2.12** Forecast plots for the BRC model using the testing set of data contributor 552. The blue curve represents CGM signals, orange curve is BG level prediction for a 30 minute time horizon, and green curve is the prediction for a 60 minute time horizon. Note that the date and time were randomly shifted to future date and time in the original OhioT1DM dataset. There is no missing interval removed when training the model.

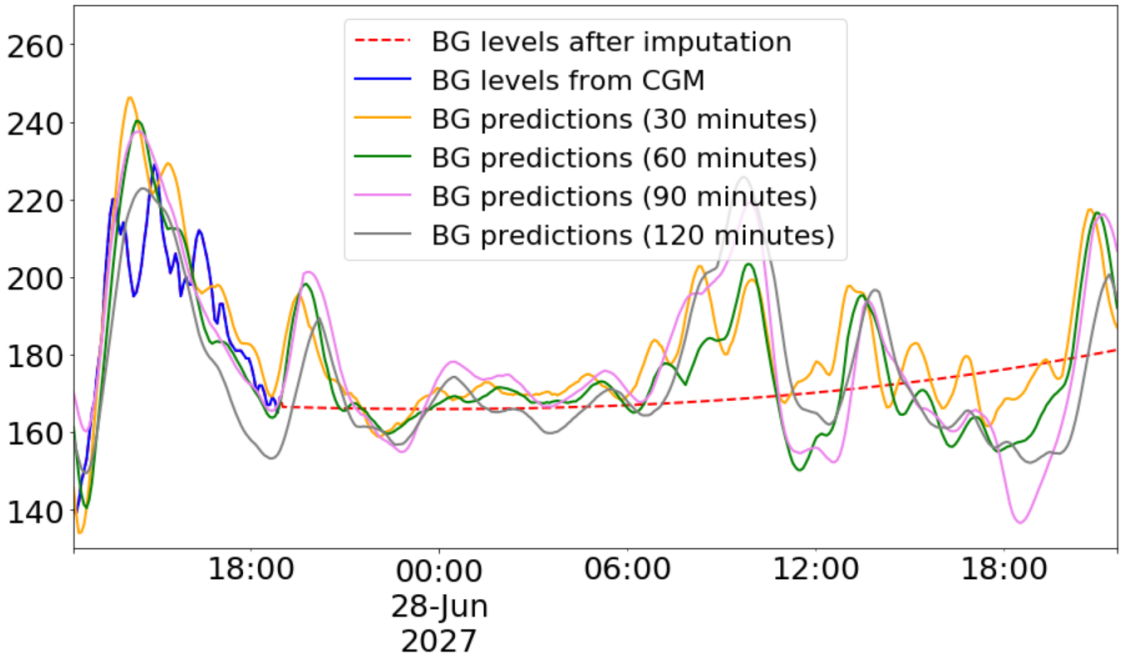


**Figure 2.13** Comparison of forecasting plots of the BRC model using the testing set of data contributor 540, presenting BG levels from CGM (blue), BG levels imputation (red dash line), predictions of 30 minutes (orange), predictions of 60 minutes (green), predictions of 90 minutes (violet), and predictions of 120 minutes (gray). Note that the date and time were randomly shifted to future date and time in the original OhioT1DM dataset. There is no missing interval removed when training the model.

CHAPTER

# 3

# PHYSIOLOGICAL MODELS

## 3.1 Introduction

Mathematical models that describe glucose-insulin transformation are necessary to understand the changes in BG dynamics based on daily activities to build an efficient algorithm for artificial pancreas. For T1D patients, the BG should be kept within the euglycemic range for the organs and tissues to maintain regular functionality [Geo18]. A low BG concentration would rise to obfuscation and coma, while excessive BG levels may cause microvascular damages and nerve damages which can be permanent.

Modeling the glucose-insulin control system for regular individuals can be described as follows. The increase of BG levels is most likely caused by consuming carbohydrates, and insulin will help lower the BG levels in the bloodstream by letting cells absorb the glucose and transfer the remaining glucose to glycogen for storage. Lastly, the renal will excrete the glucose if the level is still above a threshold. When BG levels are low, which could possibly be caused by exercising, sleeping, or a long time with an empty stomach, insulin will transfer the glycogen to glucose in the bloodstream, and a signal will also be released to remind the person to eat.

In the previous chapter, some data-driven models are discussed [Xie18; Nem20; Kha20; Ma20; Ant20; RR19; Mar18; Che18; Cap20; May20; Ham20; Bev20; Dan20; Bhi20; Fre20; Yan20; Far19; Aie20; Fox18]. Despite the high accuracy, it is hard to explain biological mechanisms governing the BG dynamics by just observing the changing in training parameters, i.e., weights and biases.

In contrast, physiological models are developed based on biological understanding and simulate the physiological processes of BG metabolism. Those large-scale simulation models usually include the components for external insulin therapy, carbohydrate intake, and physical activity for T1D.

Their complexities are affected by the postulated structure of the corresponding physiological model, which provides the output simulation for BG dynamics as well as other subsystems changes with respect to time.

### 3.1.1 Literature Review

Modeling a glucose-stimulated pancreatic insulin secretion is challenging, as many works have been studied in the last decades. A complete review of the mathematical model regarding the glucose–insulin system can be found in [Pal13]. The most well-known glucose-insulin systems are proposed by Berger [Ber89], Tarin [Tarc ], and Man [Mant ]. Berger's model [Ber89] provides the relationship among insulin absorption, insulin elimination, glucose utilization of insulin-dependent tissues, and glucose utilization of plasma glucose concentration. Tarin's model [Tarc ] extends the model presented by Mosekilde [Mos89], where the latter one illustrates the transactions among insulin in hexameric form, dimeric form, and bound insulin. Tarin introduces the three-compartment model that represents the insulin masses transfer in the plasma, the liver, and the interstitial tissue. Man's simulation model [Mant ] introduces the relationship between the glucose-insulin system with the glucose fluxes and insulin fluxes, which enables this complex model to observe the influence of other compartments.

The meal absorption systems are proposed by Lehmann [Leh92], Hovorka [Hovly], and Man [Mant ]. Lehmann's model [Leh92] constructs a linear gastric empty function with trapezoidal shape and computes the rate of appearance of ingested glucose in plasma from first-order kinetics of intestinal glucose absorption. Hovorka [Hovly] measures the glucose rate of appearance by an exponential function with the digestion time and carbohydrate bioavailability. Man [Mant ] assumes a linear gastric emptying rate and separates the types of glucose into solid and liquid form.

Meanwhile, some physical exercise dynamic models are proposed by Breton [Bren ] and Man [Mann ]. Breton provides a modified version of Bergman's minimal model [Ber81] where the minimal model links the glucose and insulin subsystem by a negative feedback loop. Breton [Bren ] quantifies the effect of heart rate by the decrements of BG levels in plasma. Man [Mann ] improves the model from Breton by incorporating the effect of mild exercise and moderate exercise depending on how fast the heart rate goes back to steady-state.

However, the majority of models were tested using simulation data and worked for the small-scaled dataset. In this chapter, we proposed complex physiological models for BG dynamics that includes important components such as meal absorption and physical exercise. The models are then validated using the OhioT1DM dataset.

### 3.1.2 Data Preparation

The inputs for a physiological model are insulin, meal intake, heart rate, and CGM signals. For all 12 anonymous data contributors, there are 6 with measurements of these categories. In [Acc18], it is recommended to calibrate CGM signals every 12 hours at least. From the CGM device instruction,

fingerstick measurements are used to calibrate CGM 3-4 times per day to improve the sensor performance. Therefore, to select relatively accurate CGM signals, the difference between the measured fingerstick and the corresponding CGM at the same time points are compared in Table 3.1. The data contributor labeled 570 has the smallest absolute and relative difference between fingerstick measurements and CGM.

**Table 3.1** Calibration comparison table of 6 data contributors, where $N_{fs}$ represents the number of fingerstick measurements for a data contributor, $\sum$|CGM-fs| stands for the summation of the absolute difference between CGM and fingerstick measured at the same time.

| Data contributor | 559 | 563 | 570 | 575 | 588 | 596 |
|---|---|---|---|---|---|---|
| $N_{fs}$ | 139 | 570 | 269 | 214 | 614 | 397 |
| $\sum\|CGM-fs\|$ | 5591 | 9975 | 2998 | 3384 | 12600 | 9336 |
| $\frac{\sum\|CGM-fs\|}{N_{fs}}$ | 40.22 | 17.5 | 11.14 | 15.81 | 20.52 | 23.52 |

We use data contributor 570 as an example for training in the physiological model and report the parameter values and performance for testing. The procedure for the other data contributors follows the same steps as described in this chapter.

The time grid was arranged every 1 minute for all features due to all physiological models from the literature being constructed with the temporal unit of minute. The values of basal and bolus insulin dose, and heart rates were filled in the time grid at the corresponding time point. The missing values of basal insulin dose were imputed by forward filling as described in the dataset [Marp ], while the missing values of bolus insulin dose were filled with 0. The 3rd order polynomial interpolation was used to fill the missing values of heart rate measurements for training data, and we applied forward filling for missing data in testing data to avoid using future values. The image of the imputation in training for heart rate can be illustrated in Figure 3.1. Finally, for carbohydrate intake, we recorded the time index and meal size when the contributor self-reported the events in the smartphone.

The CGM signals were used in the inverse problem, where the loss function compared the differences with the simulation of CGM from physiological model and CGM signals in training proportion at the same time when true CGMs are available. The total data points used in the training phase is 11554, and the additional 2745 data points were held out for testing. Each data point was approximately collected every 5 minutes.

### 3.1.3 Methods for Parameter Estimation

The estimation of model parameters directly determines the overall accuracy of the physiological model. Some of the values can be found in the literature, and the rest can be computed by the parameter estimation problem. However, providing an accurate estimation of all parameters may be impossible for a complex model. Hence, we can select a subset of parameters and fix the rest

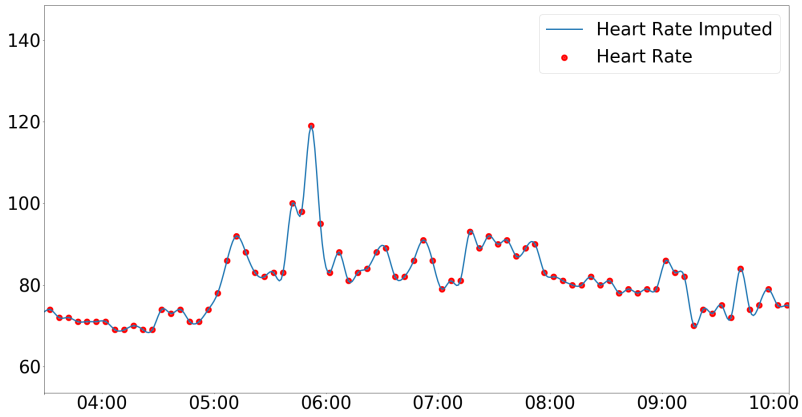


**Figure 3.1** Example of 3rd order polynomial interpolation for heart rate (HR) during 6 hours, the red dots represents the raw HR signals, the cyan curve represents the imputation results for every minute.

instead. The selection procedure is done by the sensitivity and identifiability analysis [Att17].

#### 3.1.3.1 Sensitivity Analysis

Learning how the uncertainty of the input can affect the output is crucial to determine the most influential input variables in a physiological model. The sensitivity analysis is a tool for quantifying the most influential to non-influential variables. In the review chapter from [Mor91], the authors introduce two equivalent global sensitivity analysis methods, screening method and variance decomposition. The screening method, Morris method, is varying each variable while fixing the others and observing the changes in the output. On the other hand, the variance decomposition method requires computing Sobol's indices, where the number of indices growths reflect model sensitivity.

The method of the sensitivity analysis we introduce in this section is the Morris method, which has a complete sensitivity analysis with the possibility of high computation costs. The method classifies the parameters into three groups: negligible effect parameters, large linear effects without interactions, and nonlinear effects with interaction effects. The methodology will be introduced with details in Section 3.1.3.1.3.

First, we investigate how much a parameter can affect the output function $z(t, y; q)$ where $y$ and $q$ represent the variables and parameters of the model, respectively. Here, for our model, $z(t, y; q) = G(t)$, where the glucose concentration $G(t)$ is defined in the third equation of (3.42). The partial derivative of $z$ with respect to the parameter $q$, i.e., $\frac{\partial z(t,y;q)}{\partial q}$, can be computed by automatic differentiation, complex step method, and finite difference approximation. For automatic differentiation (AD), it can numerically compute differentiation using the chain rule repeatedly and obtain the total derivative of the function [Gri89]. Complex step method [Mar00] utilized the property of complex functions and did not introduce the subtractive cancellation error, which limited the range of step size to choose. Finite difference approximation includes forward difference,

backward difference and central difference. From the definition of the derivative, the forward difference approximated $\frac{\partial z(q)}{\partial q} \approx \frac{(z(q_i+h)-z(q_i))}{h}$ for step size $h$, the backward difference gave $\frac{\partial z(q)}{\partial q} \approx \frac{z(q_i)-z(q_i-h)}{h}$, and central difference formula given in (3.4) is the average of forward difference and backward difference.

Regarding the truncation error, forward and backward difference introduced $\mathcal{O}(h)$ truncation error, central difference and complex step method obtained $\mathcal{O}(h^2)$ error, and truncation error of AD depending on the machine precision. Even though the AD is exact up to machine epsilon, the complex step method is faster computationally. Due to the complexity of our physiological model, the complex step method is the first choice and the central difference approximation is the second choice for sensitivity analysis.

#### 3.1.3.1.1 Complex Step Method

The complex step method [AM10] uses the complex functions to find the first derivatives. Thus, the step size $h$ can be small since there is no subtractive cancellation error.

Assume $z(q+ih)$ is an analytic function, where $q$ is the real part and $ih$ is the imaginary part. The Taylor expansion around $q$ is

$$z(q+ih) = z(q) + ih\frac{dz(q)}{dq} - \frac{h^2}{2}\frac{d^2z(q)}{dq^2} + \mathcal{O}(h^3). \tag{3.1}$$

In addition, assume that the function $z$ maps from real line to real line and $q, h$ are real. Then the real part and imaginary parts in Equation (3.1) would result

$$\begin{aligned} Re(z(q+ih)) &= z(q) - \frac{h^2}{2}\frac{d^2z(q)}{dq^2} + \mathcal{O}(h^4) = z(q) + \mathcal{O}(h^2), \\ Im(z(q+ih)) &= h\frac{dz(q)}{dq} + \mathcal{O}(h^3). \end{aligned} \tag{3.2}$$

Rearranging the Equation (3.2), the complex method is obtained with below formulas,

$$\begin{aligned} z(q) &\approx Re(z(q+ih)), \\ \frac{dz}{dq} &\approx \frac{Im(z(q+ih))}{h}, \end{aligned} \tag{3.3}$$

where the truncation error is $\mathcal{O}(h^2)$. The step size $h$ is typically set as machine epsilon ($10^{-16}$) to achieve high precision approximation.

Compared with other derivative-based approaches, the complex step method does not introduce the subtractive cancellation error, allowing a small enough step size $h$ to be selected. In addition, it reduces computational costs compared to AD, especially for complex models.

#### 3.1.3.1.2 Central Difference Approximation

In some cases that the complex step method does not apply, i.e. the parameter is in the index position of the differential system, the direct approach is introduced to estimate the derivatives. The central difference approximation in Equation (3.4) is the average of the forward-difference formula and backward-difference formula.

$$\frac{dz(q)}{dq_i} \approx \frac{z(q_i+h)-z(q_i-h)}{2h}, \tag{3.4}$$

with the truncation error $\mathcal{O}(h^2)$. A good candidate for the step size for the central difference is $h =$(machine epsilon)$^{\frac{1}{3}} \times q_i$ from [Den83]. Therefore, an accurate approximation for the function $z$ needs to be evaluated at high precision.

#### 3.1.3.1.3 Global Sensitivity Ranking

The $\frac{dz(t,y;q)}{dq}$ is obtained with a relatively small step size $h$ using the two methods described above, then the next step is to rank the sensitivities from high to low to identify which ones are more sensitive compared with others. Since the parameters have range from $[10^{-2}, 10^2]$, the logarithm transformation is used to scale the sensitivity rankings, i.e.,

$$\frac{d\log_{10}(z(t,y;q))}{d\log_{10}(q_j)} = \frac{q_j}{z(t,y;q)}\frac{dz(t,y;q)}{dq_j}. \tag{3.5}$$

More practically, the sensitivities of a model parameter can be relatively compared with others [Art17]. The $L_2$ norm is used for scaling the sensitivity coefficients for each parameter as presented in the following equation.

$$S_j(q) = \left\| \frac{\partial z(t,y;q)}{\partial q_j} \right\|_2 = \left( \frac{1}{t_{end}-t_0} \int_{t_0}^{t_{end}} \left( \frac{\partial z(t,y;q)}{\partial q_j} \frac{q_j}{max(z(t,y;q))} \right)^2 \partial t \right)^{\frac{1}{2}}, \tag{3.6}$$

where $q$ represents the set of unknown parameters.

A global sensitivity analysis may be required if the uncertainty of a parameter is high. If the parameters are followed some probability distribution $D(q)$, then the expected values of local sensitivity coefficients can be computed below:

$$E_q(S_j) = \int_Q S_j(q) dD(q), \tag{3.7}$$

where $Q$ is the high dimensional space of parameters. The Monte Carlo approximation can be used to make the local sensitivities pseudo-global. A set of independent parameter values $\{q_m\}_{m=1}^M$ from distribution $D(q)$ is generated and the expected value can be computed with a sample mean:

$$\bar{S}_j = \frac{1}{M}\sum_{m=1}^{M} |S_j(q_m)|. \tag{3.8}$$

The global sensitivity for each parameter was then ranked from high to low, which the highest corresponding to the most sensitive parameter in this setup. The threshold to determine from sensitive to insensitive is computed by the mean of the expected values of all parameters, i.e.,

$$\mu = \frac{1}{L}\sum_{j=1}^{L}\bar{S}_j, \tag{3.9}$$

where $L$ is the total number of parameters. The expected values $\bar{S}_j$ in (3.8) represents the individual effect from the $j$-th parameter to the output, if it is larger than the threshold $\mu$, then it consider sensitive to the model output. The variance of a parameter is obtained by:

$$\sigma_j^2 = \frac{1}{M}\sum_{m=1}^{M}\big(\bar{S}_j - S_j(q_m)\big)^2. \tag{3.10}$$

The standard deviation $\sigma_j$ in (3.10) examines the nonlinearities and/or interaction with other parameters of the $j$-th parameter. With a small $\sigma$, the changes in the parameter results in low variation of the output, which indicates the linear relationship between the parameters and model output. In contrast, a larger $\sigma$ would consider the nonlinear relationship or has interaction with one or more other parameters. In this case, a parameter with a small standard deviation and large mean value should be considered as the sensitive parameter.

#### 3.1.3.2 Identifiability Analysis

After selecting a subset of sensitive parameters, the next step is to determine which parameters can be uniquely identified from the inverse problem. Consider a simple differential equation that $\frac{dz}{dt} = a\,b\,t$, it follows that parameters $a$ and $b$ are unidentifiable since there are infinitely possible combinations of $a$ and $b$.

If we assume that $q^*$ is the minimum of the mean squared error cost function $J(q)$,

$$q^* = \underset{q}{\operatorname{argmin}}\Big(\frac{1}{N}\sum_{i=1}^{N}\big(z_i^d - z(t_i, y; q)\big)^2\Big) = \underset{q}{\operatorname{argmin}}\, J(q), \tag{3.11}$$

where $z_i^d$ represents the corresponding data value at time $t = i$. Then, using the first two terms in the Taylor series expansion around $q^*$ to approximate $z$, we obtain:

$$z(t_i; q) \approx z(t_i, y; q^*) + \frac{dz(t_i, y; q^*)}{dq}(q - q^*). \tag{3.12}$$

Since $q^*$ is the minimizer, the residuals of $z^d - z(t, y; q^*)$ are small and negligible. Substituting

into the cost function (3.11), we get

$$\begin{aligned} J(q) &\approx \frac{1}{N}\sum_{i=1}^{N}\left(z_i^d - z(t_i,y;q^*) - \frac{dz(t_i,y;q^*)}{dq}(q-q^*)\right)^2. \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\left(\frac{dz(t_i,y;q^*)}{dq}(q-q^*)\right)^2, \end{aligned} \tag{3.13}$$

The approximation of the cost function (3.13) can also be expressed as

$$J(q) \approx \frac{1}{N}(S\Delta q)^T S\Delta q, \tag{3.14}$$

where S is the sensitivity matrix contains $S_{ij} = \frac{dz}{dq_j}(t_i, y; q_j)$. Suppose that $\Delta q$ is an eigenvector of $S^T S$ with $S^T S\Delta q = \lambda\Delta q$, then the cost function (3.14) can be rewritten as

$$\begin{aligned} J(q) &\approx \frac{1}{N}\Delta q^T S^T S\Delta q \\ &= \frac{1}{N}\Delta q^T(\lambda\Delta q) \\ &= \frac{1}{N}\lambda\|\Delta q\|_2^2. \end{aligned} \tag{3.15}$$

From the result in (3.15), if the eigenvalue $\lambda$ is close to 0, then the cost function to the second-order approximation is negligible for an arbitrary parameter $q = q^* + \Delta q$, the cost function does not change when moving from $q^*$ to $q^* + \Delta q$ with an arbitrary $\Delta q$. Thus the corresponding parameter associate is unidentifiable. We use the threshold as $10^{-4}$ and find the eigenvalues which are above the threshold and order them in a decreasing order $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_k| > 10^{-4}$. We claim those parameters corresponding to $\lambda_1, \lambda_2, \ldots, \lambda_k$ are identifiable.

After the procedure of the sensitivity and identifiability analysis, a subset of sensitive and identifiable model parameters are selected for the inverse problem. The remaining parameters are fixed with the values according to the literature or from the estimation in the biological range.

#### 3.1.3.3 Optimization Techniques

In a complex physiological model, there are plenty of unknown parameters that need to be determined based on the patient-specific situation.The sensitivity and identifiability analysis are introduced to reduce the number of estimated parameters. Then the model parameters are approximated from the inverse problem. In this instance, the loss function in the inverse problem is minimized using the optimization techniques.

Given the loss function $J : \Omega \subset \mathbb{R}^n \to \mathbb{R}$, we wish to find the local minimizer $p^* \in \Omega$ which satisfies $p^* = \text{argmin}_p J(p)$.

In general, there are four parts of a optimization algorithm:

- **Convergence test**. If the conditions are satisfied for convergence, then output the solution $p^*$

and stop the algorithm.

- **Find the search direction**. To move from the current point $p_k$ to $p_{k+1}$, which makes $J(p_{k+1}) < J(p_k)$, the optimal search direction $d_k$ is required.
- **Find the step size**. Step size $h_k$ is needed for the update of the parameter $p_k$.
- **Update the parameter**. Finally, we set $p_{k+1} = p_k + h_k d_k$ and $k = k+1$, and repeat the process.

Next, a few optimization methods are introduced in this section for updating the estimated parameter $p$ in the loss function until convergence.

#### 3.1.3.3.1 Newton's Method

The Taylor series expansion for the loss function $J(p)$ around the initial point, $p_0$, is defined below

$$J(p) \approx J(p_0) + \nabla J(p_0)(p - p_0) + \frac{1}{2}(p - p_0)^T H(p_0)(p - p_0), \tag{3.16}$$

where $H$ is the Hessian matrix contains the second order derivatives such as $(H)_{i,j} = \dfrac{\partial^2 J}{\partial p_i \partial p_j}$, for the indices $i$ and $j$ in the Hessian matrix. Assume a function $W(p)$ is equal to the right hand side of Equation (3.16), where

$$W(p) = J(p_0) + \nabla J(p_0)(p - p_0) + \frac{1}{2}(p - p_0)^T H(p_0)(p - p_0), \tag{3.17}$$

then the local minimum can be found at the critical values such as $\frac{\partial W}{\partial p} = 0$ for the quadratic function $W(p)$. Thus, the minimum is found when

$$\frac{\partial W}{\partial p} = 0 = \nabla J(p_0) + H(p_0)(p - p_0) \tag{3.18}$$

$$0 = \nabla J(p_0) + H(p_0)p - H(p_0)p_0 \tag{3.19}$$

$$H(p_0)p = H(p_0)p_0 - \nabla J(p_0) \tag{3.20}$$

$$p = p_0 - H^{-1}(p_0)\nabla J(p_0). \tag{3.21}$$

Since $w(p)$ is the approximation of the loss function $J(p)$, by repeating the above procedure iteratively, then the parameter $p$ can be updated by

$$p_{k+1} = p_k - \delta_k H^{-1}(p_k)\nabla J(p_k), \tag{3.22}$$

where $\delta_k$ is a scalar representing the learning rate at time $t = k$. The $\nabla J(p_k)$ can be computed by back-propagation in the neural networks. However, the Hessian matrix and Hessian inverse are difficult to compute and computationally expensive. In practice, an approximation is used

for Hessian or its inverse by using first order derivatives of $J$, which is named the Quasi-Newton method.

#### 3.1.3.3.2 Quasi-Newton method

Assume a matrix $B_{k+1}$ is the approximation to the inverse of the Hessian matrix of $J$ at $p_k$. Rewriting the Equation (3.22) can define the Quasi-Newton [Sha70] update of $d_k$ and $p_{k+1}$ as follows:

$$d_k = -B_k \nabla J(p_k), \tag{3.23}$$

$$p_{k+1} = p_k + h_k d_k, \tag{3.24}$$

with $B_{k+1}$ being positive definite if $B_k$ is positive definite. The update of $B$ can start with $B_0 = I$, the identity matrix. To update $B_k$ at every iteration, we can approximate from the first order derivatives of $J$.

By applying the Taylor expansion at $\nabla J$, the gradient and the Hessian matrix is given:

$$\nabla J(p_{k+1}) = \nabla J(p_k) + \nabla^2 J(p_k)(p_{k+1} - p_k) + \epsilon(p_{k+1} - p_k), \tag{3.25}$$

where $\epsilon$ represents some error terms. If we assume the error is small, then it leads to the Quasi-Newton relation:

$$p_{k+1} - p_k = B_{k+1}(\nabla J(p_{k+1}) - \nabla J(p_k)), \tag{3.26}$$

which is equivalent to

$$H(p_k)(p_{k+1} - p_k) = (\nabla J(p_{k+1}) - \nabla J(p_k)). \tag{3.27}$$

The simplest method for estimate Hessian matrix is secant approximation, which for a function $J : \mathbb{R} \to \mathbb{R}$, the Hessian approximation is

$$H(p_k) = \frac{J'(p_k + 1) - J'(p_k)}{p_{k+1} - p_k}. \tag{3.28}$$

Next, the Broyden formula with rank 1 is introduced to approximate $B_k$. We can write

$$B_{k+1} = B_k + \mathbf{v}\mathbf{v}^T. \tag{3.29}$$

From the Quasi-Newton relation $B_{k+1} y_k = s_k$, where $y_k = \nabla J(p_{k+1}) - \nabla J(p_k)$ and $s_k = p_{k+1} - p_k$, it follows that

$$\begin{aligned} s_k &= B_{k+1} y_k \\ &= (B_k + \mathbf{v}\mathbf{v}^T) y_k \\ &= B_k y_k + \mathbf{v}\mathbf{v}^T y_k, \end{aligned} \tag{3.30}$$

by rearranging the above equation we obtain $\mathbf{v}(\mathbf{v}^T y_k) = s_k - B_k y_k$. Assume $z_k = s_k - B_k y_k$, we obtained

$\mathbf{v} = c\,z_k$, which represents vector $\mathbf{v}$ is in the same direction of vector $z_k$ with a constant $c$. Substitute $z_k$ in the rearrange equation, we obtain

$$\mathbf{v}\mathbf{v}^T y_k = s_k - B_k y_k \tag{3.31}$$

$$(c\,z_k)(c\,z_k)^T y_k = z_k \tag{3.32}$$

$$c^2 z_k z_k^T y_k = z_k \tag{3.33}$$

$$c^2 z_k^T y_k = 1 \tag{3.34}$$

$$c = \frac{1}{\sqrt{z_k^T y_k}}. \tag{3.35}$$

Now, the $\mathbf{v}$ can be expressed as

$$\mathbf{v} = c\,z_k = \frac{s_k - B_k y_k}{\sqrt{z_k^T y_k}}. \tag{3.36}$$

Then substitute the Equation (3.36) in (3.29) and obtain

$$\begin{aligned} B_{k+1} &= B_k + \mathbf{v}\mathbf{v}^T \\ &= B_k + \frac{(s_k - B_k y_k)(s_k - B_k y_k)^T}{z_k^T y_k} \\ &= B_k + \frac{(s_k - B_k y_k)(s_k - B_k y_k)^T}{(s_k - B_k y_k)^T y_k}. \end{aligned} \tag{3.37}$$

This method preserves the symmetry of $B_k$ (where $B_{k+1}^T = B_k^T + (\mathbf{v}\mathbf{v}^T) = B_k + \mathbf{v}\mathbf{v}^T$) but does not guarantee positive definite. Another formula for updating $B_{k+1}$ is the rank 2 corrections as defined below:

$$B_{k+1} = B_k + \alpha\mathbf{u}\mathbf{u}^T + \beta\mathbf{v}\mathbf{v}^T, \tag{3.38}$$

where $\alpha, \beta$ are constant and vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$. Similar to the procedure described in rank 1 corrections above, the formula for rank 2 update is

$$B_{k+1} = B_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{B_k y_k y_k^T B_k}{y_k^T B_k y_k}, \tag{3.39}$$

which guarantee $B_k$ positive definite, Equation (3.39) is also denoted by Davidon-Fletcher-Powell (DFP) formula [Dav59; Fle63].

Finally, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method [Fle87] is derived from the DFP formula with swapping the roles of $s_k$ and $y_k$. The formula approximate the Hessian matrix $H_k$ is defined:

$$H_{k+1} = H_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{H_k s_k s_k^T H_k^T}{s_k^T H_k s_k}, \tag{3.40}$$

where $y_k = H_k s_k$, and $H_k$ is positive definite.

### 3.1.4 Evaluation Method

To evaluate the performance of the physiological model for a data contributor, we compare the RMSE and correlation coefficient in the training and testing data.

The RMSE is introduced as one of the evaluation metrics in Chapter 2 Equation (2.12). Since we would like to compare our physiological model with data-driven models, RMSE will be used in Chapter 3 as well. The RMSEs are only reported when the CGM signals are available for both training and testing.

The correlation coefficient introduces the linear dependence of two variables, where we assume each variable has the same number of elements and same dimensions. For $N$ element vectors $x_A$, $x_B$, the correlation coefficient is given by:

$$\rho(x_A, x_B) = \frac{1}{N-1} \sum_{i=1}^{N} \Big( \frac{x_A(i) - \mu_A}{\sigma_A} \Big) \Big( \frac{x_B(i) - \mu_B}{\sigma_B} \Big), \tag{3.41}$$

where $\mu_A$, $\mu_B$ are the mean values of $x_A$ and $x_B$, respectively, and $\sigma_A$, $\sigma_B$ represent the standard deviations for $x_A$ and $x_B$.

If the correlation coefficient equals 1 or -1, then the tested variables have a linear relationship. When the correlation coefficient equals 0, the variables are linearly independent.

## 3.2 Model

In this section, a physiological model that describes the glucose and insulin transformation in organs and tissues is presented.

Insulin is a hormone that can help with storage and utilization of the glucose obtained from the carbohydrates intake. For a healthy person, insulin is produced from the beta cells in the pancreas to release and store insulin in the bloodstream.

Insulin and glucose work together to control regular body function: if a person eats a meal or snack, the food is digested in the gastrointestinal tract and turns into small sugar units, i.e., glucose. With the help of insulin, the glucose is absorbed first by blood vessels and transferred to different cells, i.e., brain or muscle tissue. Insulin also helps in transferring extra glucose to store in the liver, fat and muscle cells as glycogen. Finally, if the BG level is still high, the extra glucose will be excreted outside the body. On the other hand, if the BG level is low, the pancreas will stop releasing insulin and start producing glucagon, which transfers glycogen to glucose and releases into the bloodstream. Thus, the body maintains homeostasis.

For T1D patients, the pancreas does not produce enough insulin due to no functioning beta cells. The glucose has difficulties transferring from blood to cells, organs, or tissue, and can not store in the liver, fat, and muscle cells - it remains in the bloodstream instead.

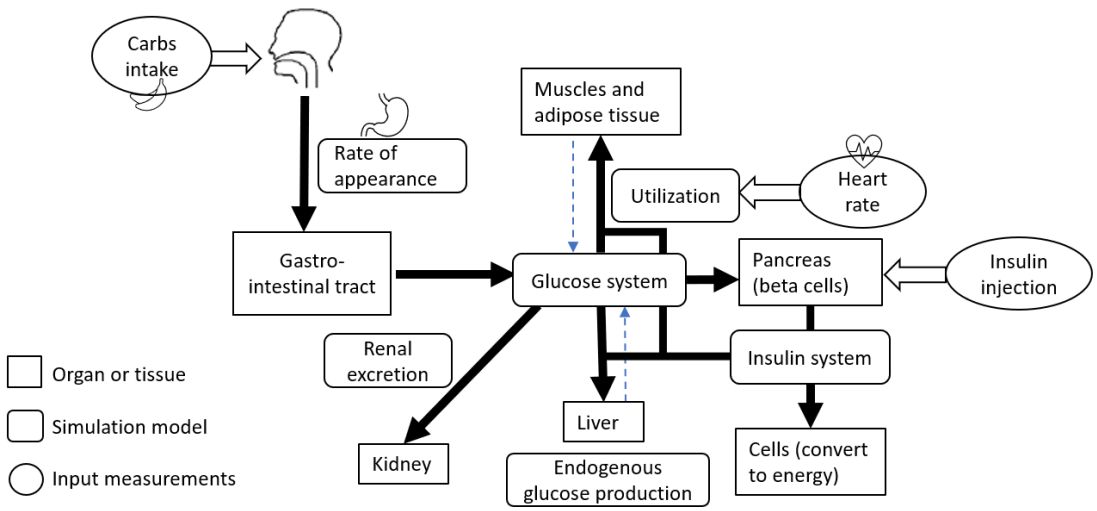


**Figure 3.2** Glucose production and transformation scheme.

The most common treatment is insulin injection, which helps regulate the body's BG levels. The injection requires BG monitoring and estimated meal time. The basal insulin is injected 24 hours through a pump device and the bolus insulin delivers only before or after a meal.

The physiological model is developed based on the above understanding and used ideas from the previous developed models, i.e., glucose-insulin system from Man [Mant ], model for the meal function from Lehmann [Leh92] and Hovorka [Hovly], and the physical activity function [Mann ] modified from Breton [Bren ]. Complex physiological models are constructed and the model parameter values are adjusted according to the dataset. The interaction between glucose and other five model components for physiological models can be visualized in Figure 3.2.

The black arrow represents how the BG produced and transformed between each model component, and the blue dash arrow represents the effect of insulin signal on BG concentration. The rectangular box represents human organ or tissue, the rounded rectangular box represents the simulation relevant to the organ or tissue, and the oval represents the measurement obtained from the data set as the real-time input to simulate the compartment model.

Next, each subsystem is introduced and the corresponding mathematical model is described.

### 3.2.1 Glucose Subsystem

The glucose kinetics [Vic97] in Figure 3.3 describes the behavior of the two-compartment model. The differential equations (3.42) provide the relationship between glucose and other subsystems which will be introduced later in the chapter. The related parameter descriptions and values are

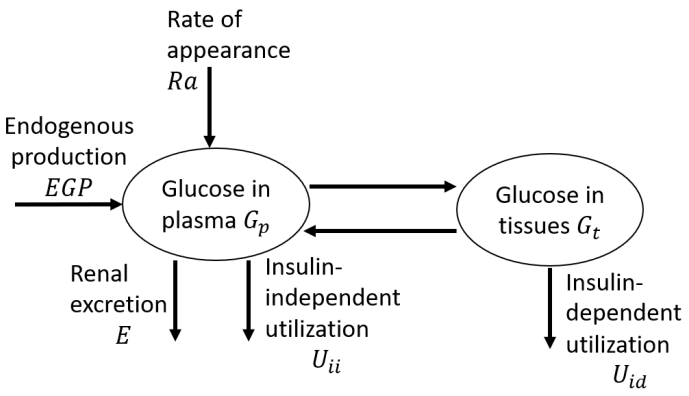


**Figure 3.3** The schematic of BG transformation with 5 subsystems.

listed in Table 3.2.

$$\begin{aligned}
&\dot{G}_p(t) = EGP(t) + Ra(t) - U_{ii}(t) - E(t) - k_1 G_p(t) + k_2 G_t(t), \quad G_p(0) = G_{pb}, \\
&\dot{G}_t(t) = -U_{id}(t) + k_1 G_p(t) - k_2 G_t(t), \quad G_t(0) = G_{tb}, \\
&G(t) = G_p(t)/V_G.
\end{aligned} \tag{3.42}$$

**Table 3.2** Table of the variables in the glucose subsystem, the values of some parameters are from [Mant ]. In the third column, if variables are changing with respect to time, then we denote them as simulations. If variables are classified as input sources, then we denote them as measurements. If a variable represents the parameters in some functions, then the value is presented.

| Variables | Unit | Source/Values | Explanation |
|---|---|---|---|
| $G_p$ | mg/kg | simulation | glucose masses in plasma |
| $G_t$ | mg/kg | simulation | glucose masses in rapidly and slowly equilibrating tissues |
| $G$ | mg/dl | measurements | plasma glucose concentration |
| $EGP$ | mg/kg/min | simulation | endogenous glucose production |
| $U_{ii}$ | mg/kg/min | simulation | insulin-independent glucose utilization |
| $U_{id}$ | mg/kg/min | simulation | insulin-dependent glucose utilization |
| $E$ | mg/kg/min | simulation | renal excretion |
| $Ra$ | mg/kg/min | simulation | glucose rate of appearance in plasma |
| $V_G$ | dl/kg | 1.88 | distribtion volume of glucose |
| $k_1$ | $min^{-1}$ | 0.065 | rate parameter of glucose kinetics |
| $k_2$ | $min^{-1}$ | 0.079 | rate parameter of glucose kinetics |
| $G_{pb}$ | mg/kg | measurement | glucose masses in plasma initial value |
| $G_{tb}$ | mg/kg | simulation | glucose masses in tissues initial value |

From the glucose transformation in Figure 3.2, the increase of BG levels is mainly due to meal intake (represented by the rate of appearance function $Ra$) and transferring from glycogen (represented by endogenous glucose production function $EGP$). From the first differential equation, the

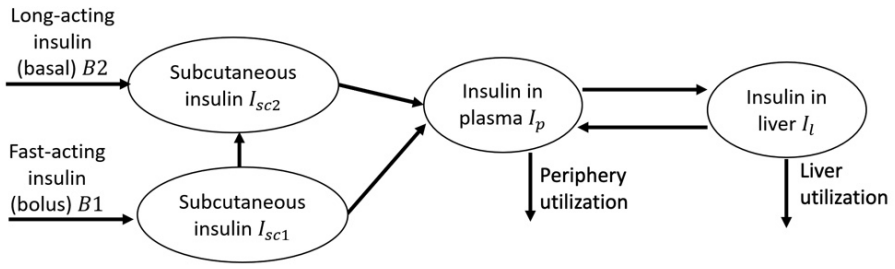


**Figure 3.4** The schematic of insulin transformation.

$EGP$, $Ra$, glucose in tissue ($G_t$) contributes positively to the rate of $G_p$. The glucose is then utilized by insulin-independent utilization ($U_{ii}$), renal excretion ($E$), and transferred a proportion to $G_t$. The initial condition for $G_p$ is also defined as the basal state denoted by $G_{pb}$. Similarly, observed from second differential equation, the rate of $G_t$ contains a positive contribution from $G_p$, and transferred insulin-dependent utilization ($U_{id}$) and a loss term $G_t$, the initial condition is also defined as the basal state $G_{tb}$. In the third equation in (3.42), the BG levels $G$ (also called CGM readings) are defined as plasma glucose $G_p$ divided by a positive constant $V_G$.

### 3.2.2 Insulin Subsystem

The insulin kinetics described in Figure 3.4 is first proposed by [Fer87], and later extended to include the interactions between the fast-acting insulin, long-acting insulin and insulin in plasma [Sch18]. Equations (3.43) provide the detailed relationship between each compartment. The insulin secretion produced by the liver is removed due to T1D can produce little or no insulin. The long-acting insulin contains the basal insulin injection, where the fast-acting insulin includes bolus insulin injection and transfer to subcutaneous insulin and long-acting insulin systems. The detailed description of parameters can be found in Table 3.3.

$$
\begin{aligned}
I(t) &= I_p(t)/V_I, \\
\dot{I}_p(t) &= -(m_2+m_4)I_p(t) + m_1 I_l(t) + B(t), \quad I_p(0) = I_{pb}, \\
\dot{I}_l(t) &= -(m_1+m_3)I_l(t) + m_2 I_p(t), \quad I_l(0) = I_{lb}, \\
B(t) &= k_{a1} I_{sc1}(t) + k_{a2} I_{sc2}(t), \\
\dot{I}_{sc1}(t) &= -(k_{a1}+k_d)I_{sc1}(t) + b_{c1} B_1(t), \quad I_{sc1}(0) = I_{sc1b}, \\
\dot{I}_{sc2}(t) &= -k_{a2} I_{sc2}(t) + k_d I_{sc1} + b_{c2} B_2(t), \quad I_{sc2}(0) = I_{sc2b}.
\end{aligned}
\tag{3.43}
$$

Insulin concentration $I$ is defined as $I_p$ divided by a positive constant $V_I$, where insulin in plasma $I_p$ interacts with insulin in liver $I_l$ with a constant rate $m_2$ and subcutaneous insulin $B$. In the plasma, subcutaneous insulin $B$ moves directly to $I_p$. In addition, insulin in liver $I_l$ contributes to $I_p$ with a rate of $m_1$, and gets back from $I_p$ with rate constant $m_2$. Plasma insulin $I_p$ also transfers

**Table 3.3** Table of the variables in the insulin subsystem, the values of some parameters are from [Mant ; Sch18]. If a parameter is dimensionless, the unit is blank. In the third column, if variables are changing with respect to time, then we denote them as simulations. If variables are classified as input sources, then we denote them as measurements. If a variable represents the parameters in some functions, then the value is presented or we denote them as estimations for unknown parameters.

| Variables | Unit | Source/Values | Explanation |
|---|---|---|---|
| $I_p$ | pmol/kg | simulation | insulin mass in plasma |
| $I_l$ | pmol/kg | simulation | insulin masses in liver |
| $I$ | pmol/l | simulation | plasma insulin concentration |
| $B$ | pmol/kg/min | simulation | rate of appearance of insulin in plasma per minute |
| $B_1$ | pmol/kg/min | measurements | the amount bolus insulin delivered occasionally |
| $B_2$ | pmol/kg | measurements | the amount of basal insulin is continuously infused |
| $V_I$ | l/kg | 0.05 | distribution volume of insulin |
| $m_1$ | $min^{-1}$ | 0.190 | rate parameter of insulin kinetics |
| $m_2$ | $min^{-1}$ | 0.484 | rate parameter of insulin kinetics |
| $m_3$ | $min^{-1}$ | 0.285 | rate parameter of insulin kinetics |
| $m_4$ | $min^{-1}$ | 0.194 | rate parameter of insulin kinetics |
| $k_{a1}$ | $min^{-1}$ | 0.0034 | rate constant of partially absorption |
| $k_{a2}$ | $min^{-1}$ | 0.014 | rate constant of final absorption |
| $k_d$ | $min^{-1}$ | 0.028 | rate constant of remaining diffuses |
| $b_{c1}$ | | estimation | rate parameter of exogenous bolus absorption |
| $b_{c2}$ | $min^{-1}$ | estimation | rate parameter of exogenous basal insulin absorption |
| $I_{lb}$ | pmol/kg | estimation | insulin masses in liver initial value |
| $I_{pb}$ | pmol/kg | estimation | insulin masses in plasma initial value |
| $I_{sc1b}$ | pmol/kg | estimation | insulin masses in subcutaneous compartment initial value |
| $I_{sc2b}$ | pmol/kg | estimation | insulin masses in fast-acting compartment initial value |

a proportion for periphery utilization. Similarly, the $I_l$ contributes to liver utilization at a rate constant $m_3$. The subcutaneous insulin $B$ consists of subcutaneous insulin systems $I_{sc1}$ and $I_{sc2}$. The subcutaneous insulin system $I_{sc1}$ receives the fast-acting insulin $B_1$ with rate constant $b_{c1}$, then transfers to subcutaneous insulin system $I_{sc2}$ with a rate constant $k_d$, and contributes to subcutaneous insulin $B$ at the rate $k_{a1}$. The long-acting insulin $B_2$ acts on subcutaneous insulin system $I_{sc2}$ with a transfer rate $b_{c2}$. Then $I_{sc2}$ contributes to $B$ with rate constant $k_{a2}$. The initial conditions for subcutaneous insulin system $I_{sc1}$, $I_{sc2}$, insulin in plasma $I_p$ and insulin in liver $I_l$ are denoted by $I_{sc1b}, I_{sc2b}, I_{pb}, I_{lb}$, respectively.

### 3.2.3 Endogenous Glucose Production Subsystem

The endogenous glucose production $EGP$ [Man06] describes the interactions between glucose signal and insulin signals in the liver. There are two types of insulin signals, delayed insulin signal $I_d$ and anticipated insulin signal $I_1$ (also named insulin action, which composes the chain reaction of insulin transformation). As shown in Figure 3.5, the plasma insulin flows into $I_1$ with a constant rate $k_i$, and left from $I_1$ to $I_d$ with the same rate constant. When the insulin delayed signal function $I_d$ receives the signal from $I_1$, it transfers to other cells for further usage. The initial condition for both functions is starting from the basal state, namely, $I_b$. Finally, the non-negative $EGP$ forcing

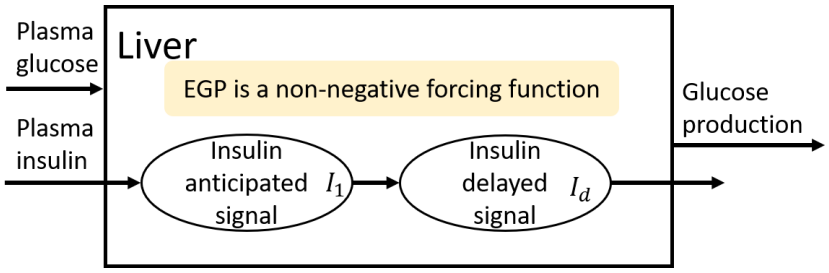


**Figure 3.5** The schematic of glucose production from liver.

function computes by subtracting the proportional of insulin delayed signals $I_d$ and the proportional of glucose concentrations $G_p$ from an initialized constant concentration $k_{p1}$. The details of the parameter are listed in Table 3.4.

$$\begin{aligned} EGP(t) &= k_{p1} - k_{p2}G_p(t) - k_{p3}I_d(t), \\ \dot{I}_1(t) &= -k_i(I_1(t) - I(t)), \quad I_1(0) = I_b, \\ \dot{I}_d(t) &= -k_i(I_d(t) - I_1(t)), \quad I_d(0) = I_b. \end{aligned} \tag{3.44}$$

**Table 3.4** Table of the variables in the EGP subsystem, the values of some parameters are from [Mant ]. In the third column, if variables are changing with respect to time, then we denote them as simulations. If a variable represents the parameters in some functions, then the value is presented or we denote them as estimations for unknown parameters.

| Variables | Unit | Source/Values | Explanation |
|---|---|---|---|
| $EGP$ | mg/kg/min | simulation | endogenous glucose production |
| $I_1$ | pmol/l | simulation | insulin anticipate signal |
| $I_d$ | pmol/l | simulation | delayed insulin signal |
| $k_{p1}$ | mg/kg/min | 2.7 | extrapolated at zero glucose and insulin |
| $k_{p2}$ | $min^{-1}$ | 0.0021 | liver glucose effectiveness |
| $k_{p3}$ | mg/kg/min per pmol/l | 0.009 | parameter governing amplitude of insulin action |
| $k_i$ | $min^{-1}$ | 0.0079 | rate between insulin signal and action |
| $I_b$ | mg/dl | estimation | plasma insulin concentration initial value |

### 3.2.4 Rate of Appearance Subsystem

In this section, we introduce two glucose absorption models, the Hovorka [Hovly] meal model and modified Lehmann [Leh92] meal model, which can be combined with our differential system described above. Both models are tested with the same dataset, and the performance of each model is compared in the Result section 3.3 later.

**Hovorka Model.** The two-compartment model is introduced to simulate the intestinal glucose absorption. The rate of appearance of ingested glucose in plasma $Ra$ is described by Equation

(3.45). The meal absorption rate is determined by the carbohydrate amount $D$, bioavailability $C_{bio}$ which describes the transferring rate from carbohydrate to glucose, and the time to reach the maximum appearance rate of glucose ($T_{max}$). The model is controlled by a transfer rate $1/T_{max}$ in the exponential function. The details of the model parameters are listed in Table 3.5.

$$Ra(t) = \frac{D(t)C_{bio}\,t\,e^{-t/T_{max}}}{T_{max}^2}. \tag{3.45}$$

**Table 3.5** Table of the variables in the first Ra subsystem, the values of some parameters are from [Hovly]. For the dimensionless unit, we leave the unit as blank. In the third column, if variables are changing with respect to time, then we denote them as simulations. If variables are classified as input sources, then we denote them as measurements. If a variable represents the parameters in some functions, then we denote them as estimations for unknown parameters.

| Variables | Unit | Source | Explanation |
|---|---|---|---|
| $Ra$ | mg/kg/min | simulation | glucose rate of appearance in plasma |
| $D$ | mg | measurements | meal size |
| $C_{bio}$ | | estimation | bioavailability absorption rate |
| $T_{max}$ | min | estimation | time to the peak of Ra |

**Modified Lehmann Model.** The second model is defined in terms of the gastric empty function $G_{empty}$, which is a piecewise rectangular shape function controlled by meal size. For a regular size meal, the constant parameter $T_m$ controls the rate of the gastric empty function; and for a snack size meal, the time parameter $T_s$ manages the digestion time. The value of $T_m$ is always larger than $T_s$. This function is modified from the original Lehmann meal model [Leh92] with setting the increasing and decreasing period to be zero and thus reducing two parameters. The function is then changed from the trapezoidal shape to rectangular shape. A visualization of the function $G_{empty}$ is shown in Figure 3.6. The general function of $G_{empty}$ and the $Ra$ are defined in Equation (3.46) and model parameters are given in Table 3.6.

$$\begin{aligned} \dot{Q}_{gut} &= -k_{gabs}Q_{gut}(t) + G_{empty}, \quad Q(0) = 0 \\ G_{empty}(t,D) &= \begin{cases} \frac{C_{bio}D(t)}{T_m}, & D(t) \geq 10 \\ \frac{2C_{bio}D(t)}{T_s}, & D(t) < 10 \end{cases}. \\ Ra(t) &= k_{gabs}Q_{gut}(t). \end{aligned} \tag{3.46}$$

The rate of appearance $Ra$ is assumed to be proportional to the intestine glucose concentration $Q_{gut}$, where the rate of $Q_{gut}$ is controlled by the gastric empty function $G_{empty}$ and a loss term proportional to $Q_{gut}$. The initial condition of $Q_{gut}$ is at the basal state, where we assume no meal consumed after the fuel digestion from the last meal.

If we consider the case when the snack and meal events are close to each other, the $G_{empty}$ is

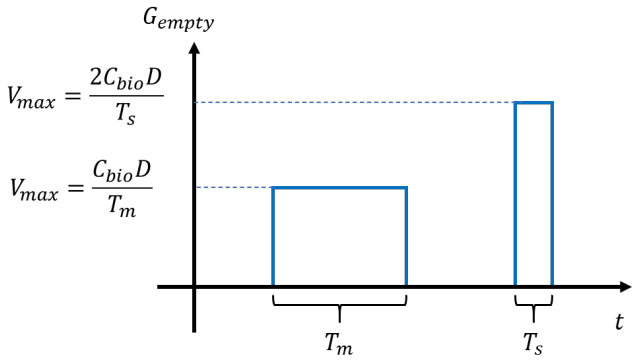


**Figure 3.6** Example of the gastric empty function plots of two meal events (left is a regular size meal, right is a snack size meal).

**Table 3.6** Table of the variables in the second Ra subsystem, the values of some parameters are from [Leh92]. For the dimensionless unit, we leave the unit as blank. In the third column, if variables are changing with respect to time, then we denote them as simulations. If variables are classified as input sources, then we denote them as measurements. If a variable represents the parameters in some functions, then the value is presented or we denote them as estimations for unknown parameters.

| Variables | Unit | Source/Values | Explanation |
|---|---|---|---|
| $Ra$ | mg/kg/min | simulation | glucose rate of appearance in plasma |
| $Q_{gut}$ | mg/kg | simulation | The amount of glucose in the intestine |
| $G_{empty}$ | mg/kg/min | simulation | the rate of gastric emptying |
| $D$ | mg | measurements | meal size |
| $k_{gabs}$ | $min^{-1}$ | 1/60 | rate of glucose absorption from the gut |
| $V_{max}$ | mg/kg/min | 0.36 | maximal rate of gastric emptying |
| $C_{bio}$ | | estimation | bioavailability absorption rate |
| $T_m$ | min | estimation | the time staying at the maximal ($V_{max}$) for a regular meal size |
| $T_s$ | min | estimation | the time staying at the maximal ($V_{max}$) for a snack size |

capable of considering the overlapping effect that the function is aggregated from the level when the second event happened. The 6 cases, including meal overlaps with another meal, meal overlaps with a snack, meal event without overlapping, snack overlaps with a meal, snack overlaps with another snack, snack event without overlapping, aggregation schematics are shown in Figure 3.7.

We notice that if we fix $T_s = 10$ min, $T_m = 20$ min for the piecewise function $G_{empty}$ and $T_{max} = 40$ min in the Hovorka model, then two rates of appearance function are almost identical to each other with a slight difference in speed of decay as it is depicted in Figure 3.8, the Lehmann meal model is faster decay than Hovorka.

### 3.2.5 Glucose Utilization Subsystem

The utilization of glucose has two types, insulin-independent ($U_{ii}$) [Mant ] and insulin-dependent ($U_{id}$) [Bren ; Mann ]. We introduce the function from [Mann ] that improved Breton's activity model [Bren ] by excluding the rapid on-and-off changes.

Insulin-independent utilization $U_{ii}$ represents the usage of glucose from the brain and erythro-

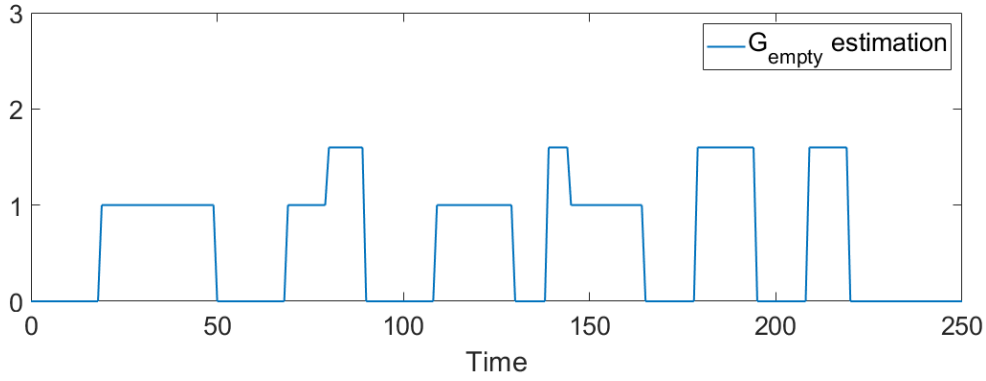


**Figure 3.7** The 6 cases of meal and snack events with or without overlapping.

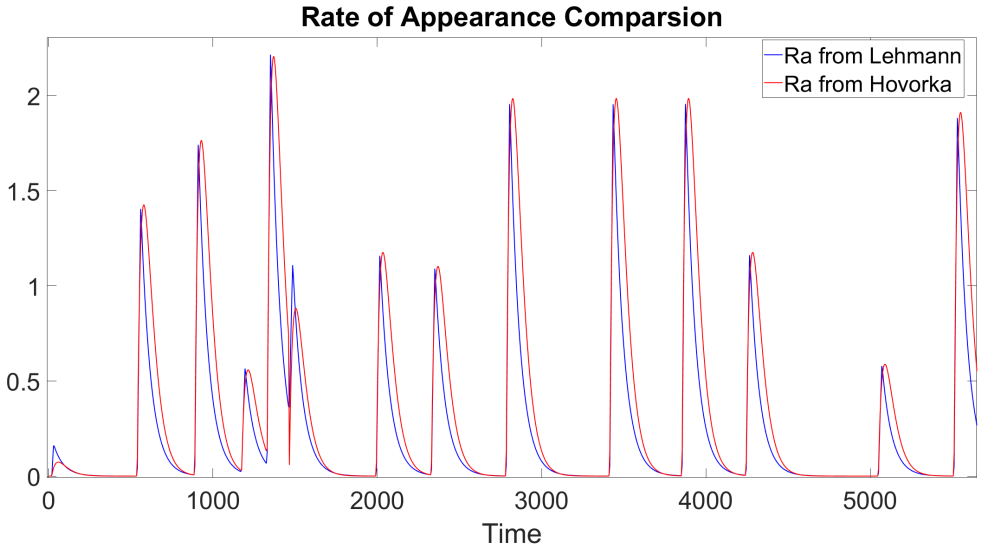


**Figure 3.8** Rate of appearance comparison plot for 2 meal models in a certain setup.

cytes and is modeled by a constant denoted by $F_{cns}$. Insulin-dependent utilization $U_{id}$ describes the glucose behavior during physical activities measured by the heart rate. From the model of [Bren ], $Y$, $f(Y)$, and $Z$ monitor the levels of insulin usage by the changes in the heart rate. For instance, if the heart rate is increasing to a high level, which is normally relevant to an exercise, the function $Y$ will reach a high value, resulting in $f(Y)$ reaches 1 rapidly. Then the value of function $Z$ is slowly increasing, leading to an increasing numerator and decreasing denominator in $U_{id}$, which represents an increment in utilization of glucose. On the other hand, if the heart rate is close to the basal heart rate, then it would result in the decay of $Y$, where $f(Y)$ becomes negligible, and finally $Z$ will fall back slowly to the basal level. Since we assume the initial conditions start at basal state, there is no physical activity involved at the beginning of observation. The complete system is given by Equations (3.47) and parameters are listed in Table 3.7.

$$\begin{aligned}
U_{ii}(t) &= F_{cns},\\
U_{id}(t) &= \frac{V_{m0}+V_{mx}(1+\alpha Z(t))(X(t)+I_b)-V_{mx}I_b}{K_m\big(1-\gamma Z(t)(X(t)+I_b)\big)+G_t(t)}G_t(t),\\
\dot{X}(t) &= -p_{2U}X(t)+p_{2U}(I_p(t)-I_b), \quad X(0)=0,\\
\dot{Y}(t) &= -\frac{1}{T_{HR}}(Y(t)-(HR(t)-HR_b)), \quad Y(0)=0,\\
\dot{Z}(t) &= -\Big(\frac{f(Y(t))}{T_{in}}+\frac{1}{T_{ex}}\Big)Z(t)+f(Y(t)), \quad Z(0)=0,\\
f(Y) &= \frac{(\frac{Y(t)}{aHR_b})^n}{1+(\frac{Y(t)}{aHR_b})^n}.
\end{aligned} \tag{3.47}$$

### 3.2.6 Glucose Renal Excretion Subsystem

Glucose excretion happens when the glucose levels exceed some threshold, and the kidney will excrete the extra glucose. If the glucose in plasma increases to the threshold $k_{e2}$, the extra glucose is released outside the body by the rate of $k_{e1}$. Otherwise, the renal excretion function remains silent in the glucose-insulin control system. The linear relationship between plasma glucose and renal glucose is described by Equation (3.48). The model parameters are in Table 3.8.

$$E(t) = \begin{cases} k_{e1}(G_p(t)-k_{e2}), & G_p(t) > k_{e2},\\ 0, & G_p(t) \le k_{e2}. \end{cases} \tag{3.48}$$

## 3.3 Results

Since the input measurements are given every one minute, a small enough fixed step size can be chosen so that the ODE will be evaluated at the time points where the data is collected. Thus, the fixed step size for Runge-Kutta 4th order method [Asc98] is used as the ODE solver approximating

**Table 3.7** Table of the variables in the glucose utilization subsystem, the values of some parameters are from [Mann ; Bren ]. For the dimensionless unit, we leave the unit as blank. In the third column, if variables are changing with respect to time, then we denote them as simulations. If variables are classified as input sources, then we denote them as measurements. If a variable represents the parameters in some functions, then the value is presented or we denote them as estimations for unknown parameters.

| Variables | Unit | Source/Values | Explanation |
|---|---|---|---|
| $U_{ii}$ | mg/kg/min | simulation | insulin-independent glucose utilization |
| $U_{id}$ | mg/kg/min | simulation | insulin-dependent glucose utilization |
| $X$ | pmol/l | simulation | insulin in the interstitial fluid |
| $Y$ | beats/$min^2$ | simulation | delayed version of the over-basal HR signal |
| $Z$ | | simulation | nonlinear differential equation |
| $f$ | | simulation | function of heart rate |
| $HR$ | beats/min | measurements | heart beats in one minute |
| $F_{cns}$ | mg/kg/min | 1 | glucose uptake by the brain and erythrocytes |
| $V_{m0}$ | mg/kg/min | 2.5 | parameter of glucose utilization at zero insulin action |
| $V_{mx}$ | mg/kg/min per pmol/l | 0.047 | disposal of insulin sensitivity |
| $p_{2U}$ | $min^{-1}$ | 0.0331 | rate constant of insulin action |
| $\gamma$ | | $10^{-7}$ | rate parameter of physical activities |
| $\alpha$ | | $3\times10^{-4}$ | rate parameter of physical activities |
| $a$ | | 0.1 | rate parameter of physical activities |
| n | | 4 | function f parameter |
| $T_{HR}$ | min | 5 | heart rate reaction to exercise |
| $T_{in}$ | min | 1 | quasi-exponential decay |
| $T_{ex}$ | min | 600 | quasi-exponential increase |
| $HR_b$ | beats/min | 60 | heart beats steady-state situation |
| $K_m$ | mg/kg | estimation | parameter of glucose utilization at zero insulin action |

**Table 3.8** Table of the variables in the renal subsystem, the values of some parameters are from [Mant ]. In the third column, if variables are changing with respect to time, then we denote them as simulations. If a variable represents the parameters in some functions, then the value is presented.

| Variables | Unit | Source/Values | Explanation |
|---|---|---|---|
| $E$ | mg/kg/min | simulation | renal excretion |
| $k_{e1}$ | $min^{-1}$ | 0.0005 | glomerular filtration rate |
| $k_{e2}$ | mg/kg | 339 | renal threshold of glucose |

the solutions of the physiological model defined in Section 3.2.

The procedure for a 4th order Runge-Kutta is defined as follows. For an initial value problem

$$\frac{dz}{dt} = f(t,z), \quad z(t_0) = z_0, \tag{3.49}$$

where $z$ is the unknown function of time $t$ that we wish to approximate using the Runge-Kutta method, the rate of changes is provided as $f(t,z)$ and the initial condition is given at $t_0$. A step size

$h > 0$ is picked and the $z_{n+1}$ can be calculated from Equations (3.50).

$$\begin{aligned} z_{n+1} &= z_n + \frac{1}{6}h(k_1 + k_2 + k_3 + k_4), \\ k_1 &= f(t_n, z_n), \\ k_2 &= f(t_n + \frac{h}{2}, z_n + h\frac{k_1}{2}), \\ k_3 &= f(t_n + \frac{h}{2}, z_n + h\frac{k_2}{2}), \\ k_4 &= f(t_n + h, z_n + hk_3). \end{aligned} \tag{3.50}$$

Two physiological models were tested where we change the meal functions from Hovorka to Lehmann meal function. The sensitivity and identifiability analysis are presented for both models. The comparison is given in the following section together with the analysis of model behaviors.

The loss function during the optimization is defined as

$$p^* = \underset{p}{\operatorname{argmin}}\, J(t,p) = \underset{p}{\operatorname{argmin}} \left\{ \frac{1}{N}\sum_{i=1}^{N}\big(f(t_i;p) - y(t_i)\big)^2 + \kappa e^{-\frac{\xi \sum_{i=1}^{N}\left(|f(t_i;p)-A|\right)}{N}} \right\}, \tag{3.51}$$

where $p \in \mathbb{R}^m$ is the vector of $m$ estimated parameters, $f(t_i;p)$ is the time derivative solution at time $t_i$, and $y(t_i)$ is the measured data. The loss function $J(t,p)$ consists of two parts, the mean squared error and a penalty term. The scalar value $\kappa$, $\xi$ is to amplify the penalty term when the output function $f(t_i;q)$ is close to its mean value $A$. When the model output $f(t_i;p)$ is not flat or close to the mean value $A$, then the second component is negligible. Thus, the sensitivity and identifiability analysis also apply to the loss function (3.51).

The quasi-newton method is used during the the optimization, where we first apply the implicit filtering [Kel99] with providing the biological range for each parameter and then update the parameters by minimization of unconstrained multivariable function (in MATLAB it is under package *fminunc*).

### 3.3.1 Results of Physiological Model with Hovorka Meal Function

There are 11 parameters that need to be estimated in Table 3.9, others are taking values from literature as described in Chapter 3.2. Next, we select a subset of sensitive and identifiable parameters for the inverse problem. The procedure of the sensitivity analysis is described in Section 3.1.3.1. For global sensitivity evaluation, we use normal distribution and randomly select 100 parameter values for local sensitivity analysis to make it pseudo-global. The mean of the normal distribution is taken from literature or the biological range observed from the physiological model. The average and standard deviation of global sensitivity rankings for each parameter are presented in Figure 3.9, with the red line representing the threshold to determine the parameters' sensitivities.

From the global sensitivity ranking, we obtain that the top 4 parameters are the most sensitive compared with others, which are $\{C_{bio}, K_m, T_{max}, b_{c2}\}$. We also plot the mean ($\mu$) global sensitivi-

**Table 3.9** Table of unknown parameters for physiological model with Hovorka meal function. For the dimensionless unit, we leave the unit as blank.

| parameter | unit | biological range | relevant subsystem |
|---|---|---|---|
| $G_{tb}$ | mg/kg | [100,300] | glucose subsystem |
| $b_{c1}$ | | [0,4] | insulin subsystem |
| $b_{c2}$ | $min^{-1}$ | [0,4] | insulin subsystem |
| $I_b$ | mg/dl | [0,10] | insulin subsystem |
| $I_{lb}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{pb}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{sc1b}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{sc2b}$ | pmol/kg | [0,10] | insulin subsystem |
| $C_{bio}$ | | [4,10] | rate of appearance subsystem |
| $T_{max}$ | min | [20,120] | rate of appearance subsystem |
| $K_m$ | mg/kg | [100,300] | insulin-dependent utilization subsystem |

ties vs standard deviation ($\sigma$) in Figure 3.10. The results indicate $\{C_{bio}, K_m, T_{max}, b_{c2}\}$ satisfy the condition of $\sigma << \mu$, thus there is no interaction between the selected parameters, and the model output function is linearly depends on the selected subset of parameters following the analysis from [Mor91].

Next, we follow the identifiability analysis in Chapter 3.1.3.2 for computing the eigenvalues of the sensitivity matrix containing the sensitive parameters $\{C_{bio}, K_m, T_{max}, b_{c2}\}$. The eigenvalues of 4 parameters are all above the threshold with the most to least identifiable ranking as $\{C_{bio}, K_m, b_{c2}, T_{max}\}$ which confirms with the analysis above.

As described earlier, the loss function (3.51) is composed of RMSE with a penalty term to avoid the flat and centered behavior of the estimation of CGM. We set the coefficients $\kappa = 100$, $\xi = 10$, and $A$ is the mean value in the time derivative solution $f(t,p)$ for the best convergence speed and prediction accuracy.

The inverse problem is solved by using the implicit filtering method first and fine tuning by the Quasi-Newton method until convergence. For the parameters are not sensitive and identifiable, except for $b_{c1}$, they are all initial values of differential equations. The initial values can be tested with different combinations of random guessing and observe if the first estimation point of BG level is matched with the first CGM signal. For the values of $b_{c1}$, it is estimated along with the 4 selected parameters and fixed its value after several iterations. The parameter estimation results are presented in Table 3.10.

**Table 3.10** Parameter values of 11 unknown parameters with Hovorka meal function. The estimation results for $\{C_{bio}, K_m, b_{c2}, T_{max}\}$ are found using optimization methods, the rest parameters are fixed during the optimization procedure.

| | $C_{bio}$ | $K_m$ | $b_{c2}$ | $T_{max}$ | $b_{c1}$ | $I_b$ | $I_{pb}$ | $I_{lb}$ | $I_{sc1b}$ | $I_{sc2b}$ | $G_{tb}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| values | 9.63 | 151.53 | 0.30 | 504.63 | 2 | 10 | 3 | 3 | 3 | 3 | 100 |

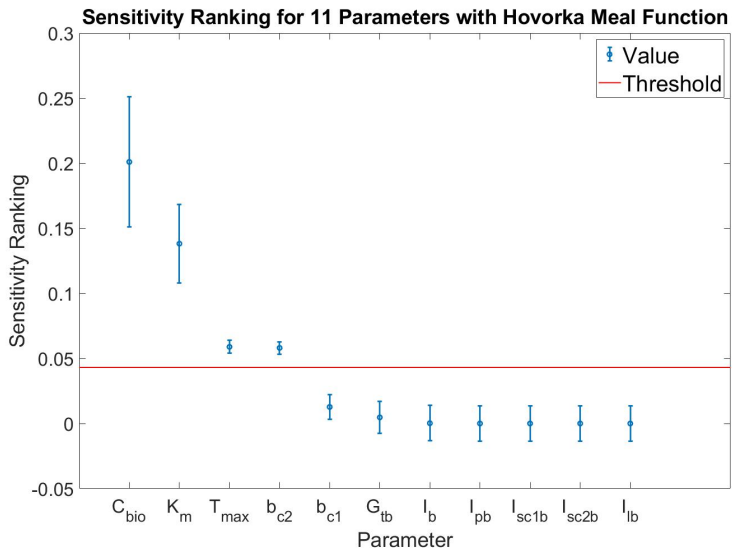


**Figure 3.9** Sensitivity coefficient ranking of physiological model with Hovorka meal function. The threshold is found by Equation (3.9), and the error bar for each parameter contains the expected values found by (3.8) with the bar as the standard deviation with other parameters.

The RMSE and correlation coefficients are used for evaluation of the performance of the physiological model in training and testing dataset. The evaluation metric only compares the difference at the time points that CGMs are taken, and parameters used to generate the testing behavior are estimated from training data only. The training RMSE is 60.52 with a high correlation 0.27 , and testing RMSE is 72.59 with correlation of prediction and true signals as -0.04. The prediction and CGM signal comparison plot for 570 is in Figure 3.11, the testing predictions are much worse than training, where a similar behavior was also observed in data-driven models. The bar graph of evaluation performance among 6 data contributors are shown in Figure 3.13, the mean values are plotted with the standard deviation (error bar).

The prediction results obtained from the physiological model are not satisfying, there are some predicted values close to CGM signals, but most values are far away from CGM signals. Compared with the data-driven model, the disadvantage in the prediction performance of the physiological model is obvious. That may be due to the limitation of the physiological model where the pattern is repeated based on the fixed parameters in each subsystem throughout the time. However, in the data-driven model, the patterns can vary from time to time by adjusting the weights and biases. For the physiological model, the advantages are that the prediction is stable once the parameters are obtained, and parameters have biological meanings for analyzing the characteristics of a T1D patient.

In Figure 3.12, the training dynamics of endogenous glucose production $EGP$, glucose utilization $U$, rate of appearance $Ra$, and plasma insulin $I$ are presented for visualization purposes. Each plot contains the system dynamics from the 0 minutes to 10000 minutes (around 7 days). From the experiments of 204 normal subjects in Man et. al. [Mant ], the biological range of $EGP$ is $[0, 2.5]$

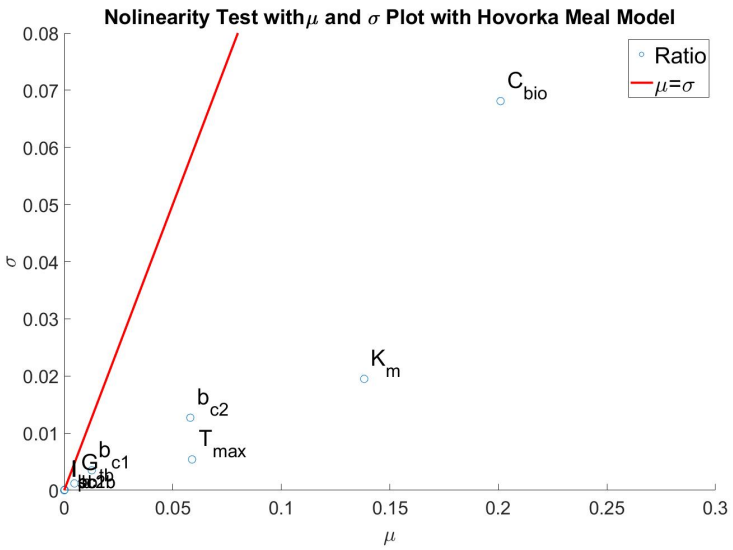


**Figure 3.10** Morris screening using mean and standard deviation of sensitivity coefficients, where $\sigma$ and $\mu$ represents the standard deviation and mean value computed from Equations (3.10) and (3.8), respectively. The red reference line is when $\mu = \sigma$.

*mg/kg/min* for a normal person, then $U$ should lie in range $[0,12]$ *mg/kg/min*, $Ra$ can start from 0 *mg/kg/min* and increase upto 14 *mg/kg/min* after meal consumption, lastly, the plasma insulin level has the range of $[0,800]$ *pmol/l*.

Compared with the estimation of each subsystem with the literature, we found that except the endogenous glucose production $EGP$ is in the biological range, the rate of appearance $Ra$, plasma insulin $I$, and glucose utilization $U$ is significantly lower in the biological range. Thus, on the prediction plot Figure 3.11, the prediction is less fluctuant with the CGM signal, where the extreme values cannot be captured. In the meanwhile, the estimation of $T_{max}$, a parameter measuring the time to peak of the rate of appearance, is out of biological range when the local minimum is detected. If the value of $T_{max}$ is large, then the rate of appearance will be smaller in magnitude. Observing this behavior may indicate the Hovorka meal function is not suitable for this particular data contributor.

### 3.3.2 Results of Physiological Model with Modified Lehmann Meal Function

Similarly, the table of estimated parameters and the biological range suggested from literature are reported in Table 3.11. We applied the same procedure as described above for the second model. We obtain the global sensitivity ranking plot in Figure 3.14 and the Morris screening in Figure 3.15.

We obtain the sensitive parameters that are above the threshold to be $\{C_{bio}, b_{c2}, K_m, T_m\}$. From Morris screening, it first selects the parameters with large effects $\{C_{bio}, b_{c2}, K_m, T_m, b_{c1}\}$, and all listed parameters satisfy $\sigma << \mu$, which indicates the linear relationship with output and no parameter interactions. Since the parameter $b_{c1}$ is close to the threshold in Figure 3.14, we decided to follow the Morris screening result.

The identifiability analysis indicates all 5 sensitive parameters are identifiable with the most to

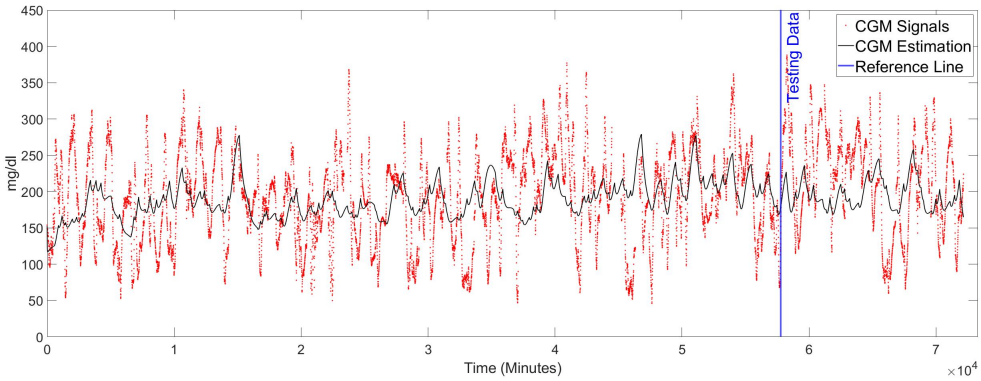


**Figure 3.11** Training and Test simulation of physiological model with Hovorka meal function for 570, the left side of the reference line represents the training data and the left is testing data.

**Table 3.11** Table of unknown parameters for physiological model with Lehmann meal function. For the dimensionless unit, we leave the unit as blank.

| parameter | unit | biological range | relevant subsystem |
|---|---|---|---|
| $G_{tb}$ | mg/kg | [100,300] | glucose subsystem |
| $b_{c1}$ | | [0,4] | insulin subsystem |
| $b_{c2}$ | $min^{-1}$ | [0,4] | insulin subsystem |
| $I_b$ | mg/dl | [0,10] | insulin subsystem |
| $I_{lb}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{pb}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{sc1b}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{sc2b}$ | pmol/kg | [0,10] | insulin subsystem |
| $C_{bio}$ | | [4,8] | rate of appearance subsystem |
| $T_s$ | min | [10,20] | rate of appearance subsystem |
| $T_m$ | min | [20,100] | rate of appearance subsystem |
| $K_m$ | mg/kg | [100,300] | insulin-dependent utilization subsystem |

the least ranking is $\{b_{c2}, C_{bio}, K_m, T_m, b_{c1}\}$. The parameter estimation results are in Table 3.12.

We obtain the training RMSE to be 83.53 with correlation coefficient -0.12, testing RMSE to be 100.54 with correlation coefficient -0.48 for the data contributor 570, and the prediction plots are shown in Figure 3.16 with the subsystems plots depicted in Figure 3.17. Additionally, the bar graph of RMSE and correlation coefficients for training and testing data among 6 data contributors are shown in Figure 3.18.

In the prediction plots comparing the difference between the prediction and CGM signals, we found the Lehmann meal model brings more dynamics to CGM predictions, which in the subplot of $Ra$ function also reflects a large magnitude compared to the Hovorka meal function. The simulation of $Ra$ falls in the biological range for a normal person, and may indicate the physiological model using the Lehmann meal function works better. However, the RMSE and the correlation plot presents an opposite result, the RMSE is larger than the previous model and the correlation coefficients in training and testing indicate the predictions are not reflecting the correct CGM signals. The plasma

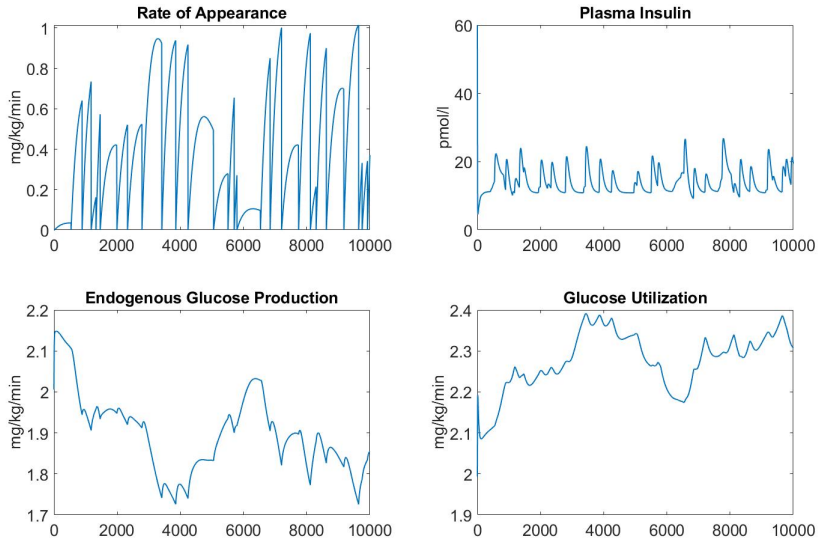


**Figure 3.12** Subsystem dynamics in the physiological model with Hovorka meal function for 570, using the first 10000 training points (around 7 days).

**Table 3.12** Parameter values of 11 unknown parameters with Lehmann meal function. The estimation results for $\{C_{bio}, K_m, b_{c2}, T_m, b_{c1}\}$ are estimated from optimization methods, the rest parameters are fixed during the optimization.

| | $C_{bio}$ | $K_m$ | $b_{c2}$ | $T_m$ | $b_{c1}$ | $T_s$ | $I_b$ | $I_{pb}$ | $I_{lb}$ | $I_{sc1b}$ | $I_{sc2b}$ | $G_{tb}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| values | 4 | 284.67 | 1.63 | 99.88 | 4 | 19.94 | 3 | 3 | 3 | 3 | 3 | 100 |

insulin $I$ average levels are higher than the physiological model with Hovorka function, thus, the predictions of CGM as shown in Figure 3.16 has a large magnitude and obtains the fluctuant patterns that may similar to the biological range of a T1D people. The glucose utilization $U$ and endogenous glucose production $EGP$ are similar with the previous model and may indicate the stability of physiological models with changing of model components.

However, The evaluation metrics show that the physiological model with Lehmann meal function is worse than the Hovorka function, the negative correlation results reflect that the physiological model is incompatible with the data contributor. It may result from the incorrect meal function that generates the wrong patterns for predictions, especially for the testing set.

### 3.3.3 Comparison Between Two Models

In Figure 3.19, we plot simulated CGMs with true CGM signals for both models. The physiological model with Hovorka function results less fluctuant in the simulation but with higher accuracy and higher correlations. Referring to the subsystem plots 3.12 and 3.17, except for rate of appearance function, other subsystems performed a similar behavior for both models with different estimated values of model parameters, which indicates the meal function is crucial for estimating the correct

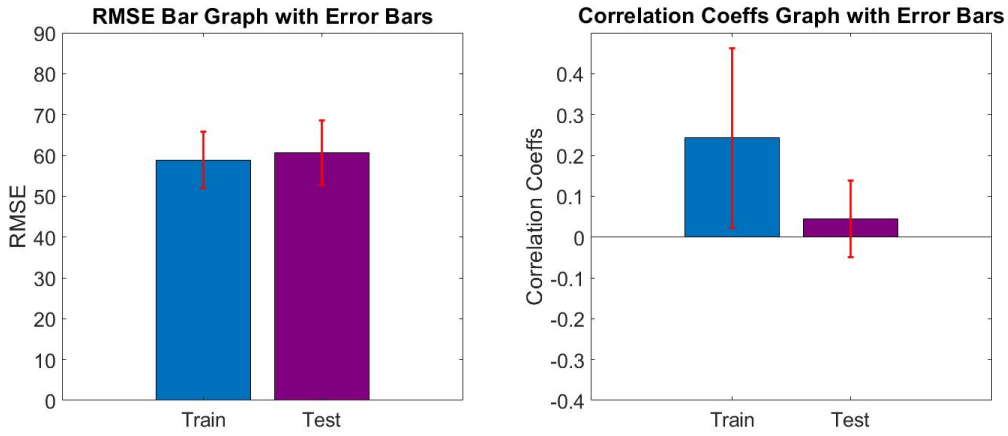


**Figure 3.13** The bar graph for RMSE (left subplot) and correlation coefficients (right subplot) between prediction and the true CGM signals for physiological model with Hovorka meal function among 6 data contributors, the mean values are reported along with the standard deviation in the red error bar.

trend of CGM signals. The bar plot 3.20 of evaluation metrics shows Hovorka meal function introduced high correlation coefficients while a small testing error is associated with modified Lehmann meal function.

## 3.4 Discussion

We construct the state of art physiological model from the ideas of [Mant ; Hovly; Bren ; Mann ], and add modifications based on our input measurements. With this differential system, we are able to understand the transformation of BG between organs and tissues for the T1D data contributors by observing the dynamics in each subsystem. We hope to obtain the simulation of BG levels with a high accuracy (low RMSE) so the physiological model can be better applied to interpret the activities of T1D and provide the guidance for preventing hyperglycemia or hypoglycemia.

Based on the experiments of the physiological model on one specific data contributor, we found that the model has limitations regarding the simulation and prediction, and requires the assumptions in initial conditions. In a complex physiological model we define in this chapter, the number of parameters may not be enough to estimate the multiple patterns throughout the day. The predictions in both experiments (Hovorka and Lehmann) show the physiological model presenting a similar pattern while the input sources slightly vary over time. Due to the input source not containing CGM, this adds difficulty to learn the patterns directly when compared to a data-driven model. The limit model parameters and optimization tools restrict the accuracy and convergence speed. Thus, obtaining an accurate result similar to data-driven models is almost impossible by using only physiological models.

In [Sak17], the authors also obtain a similar behavior for failing to capture the short-term spikes and falls, and in [Tur16] indicates the parameters in the physiological model may be time-varying. A future improvement of the physiological model can be changing the estimated parameters to be

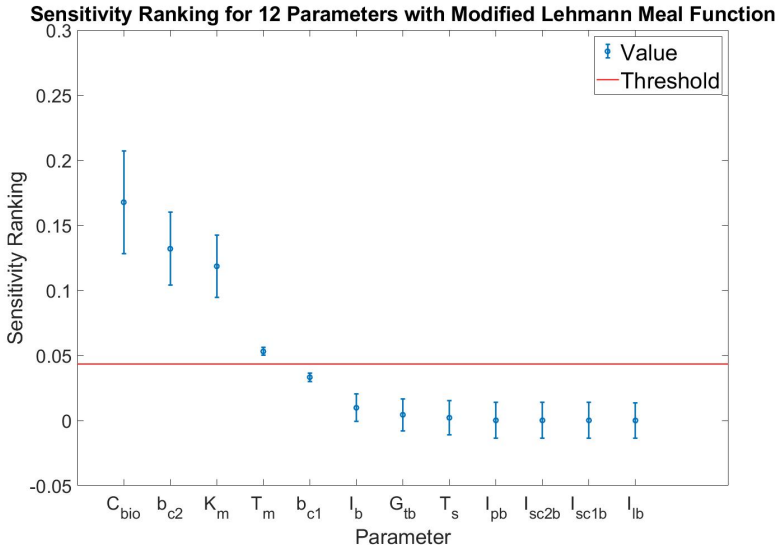


**Figure 3.14** Sensitivity coefficient ranking of physiological model with modified Lehmann meal function. The threshold is found by Equation (3.9), and the error bar for each parameter contains the expected values found by (3.8) with the bar as the standard deviation with other parameters.

adaptive to time with more specific restrictions, i.e., distinguishing the parameter values for day and night.

Another method that could improve the physiological model's result is introducing a hybrid model, in which one or more components of the physiological model are replaced with data-driven models to identify the underlying function for a specific patient. In this way, the physiological model may maintain interpretability while improving the performance.

## 3.5 Contributions

From the experiment of applying physiological models, we learn how the subsystem such as insulin, meal intake, physical exercise affects the glucose dynamics. It also provides the possibility of simulating BG signals without any knowledge of CGM (as it is shown in the testing data predictions). The physiological model took 180 seconds to process and is faster than most deep learning algorithms, but it took efforts to find the perfect model and to improve the estimation accuracy of the model parameters.

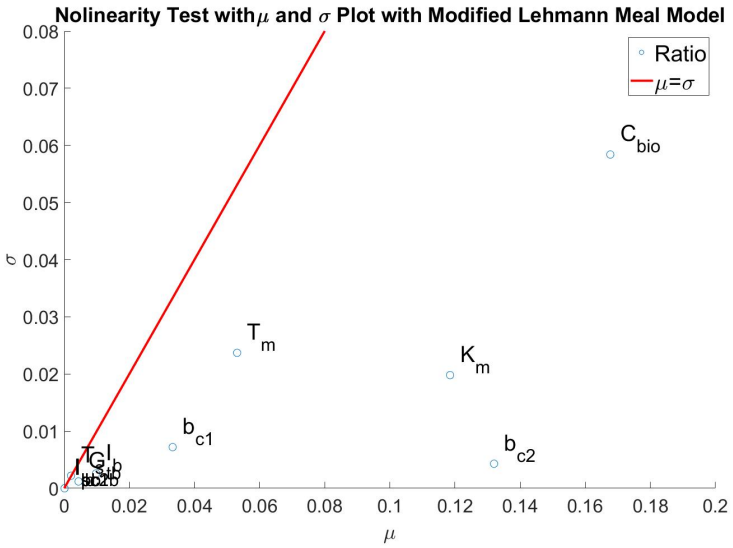


**Figure 3.15** Morris screening using mean and standard deviation of sensitivity coefficients, where $\sigma$ and $\mu$ represents the standard deviation and mean value computed from Equations (3.10) and (3.8), respectively. The red reference line is when $\mu = \sigma$.

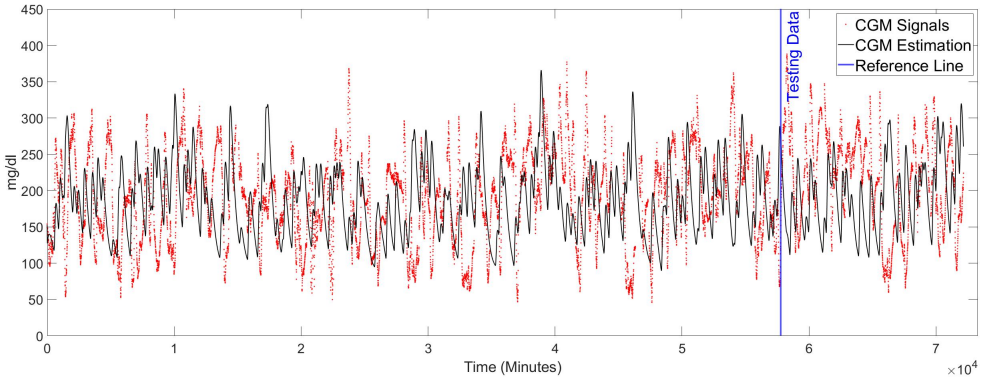


**Figure 3.16** Training and Testing simulation plot of physiological model with modified Lehmann meal function and CGM signals for 570, the left side of the reference line represents the training data and the left is testing data.

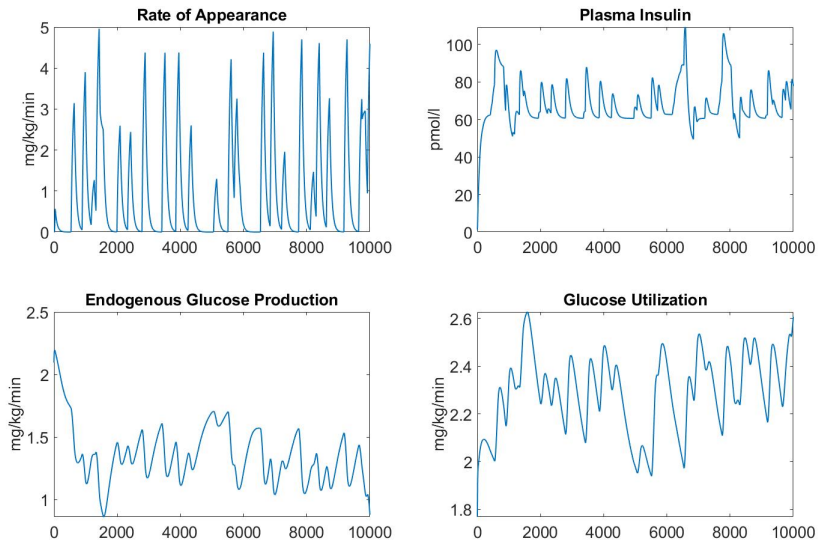


**Figure 3.17** Subsystem dynamics in physiological model with modified Lehmann meal function for 570, using the first 10000 training points (around 7 days).

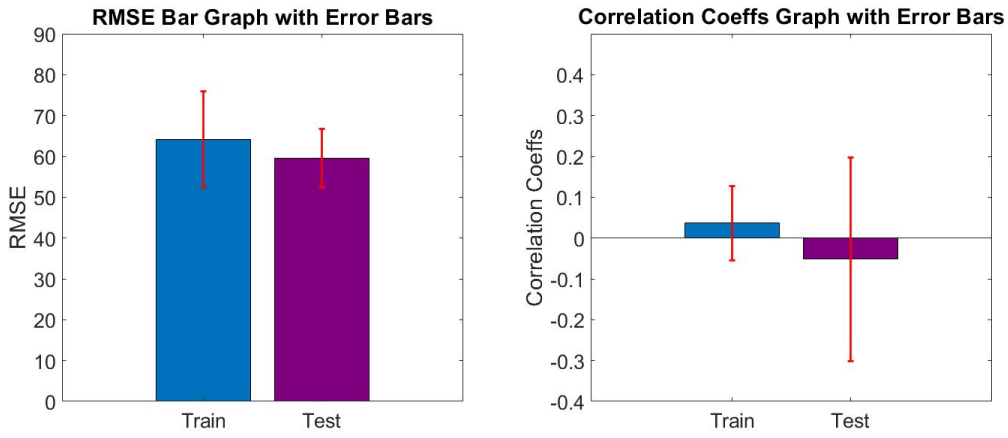


**Figure 3.18** The bar graph for RMSE (left subplot) and correlation coefficients (right subplot) between prediction and the true CGM signals for the physiological model with modified Lehmann meal function among 6 data contributors, the mean values are reported along with the standard deviation in the red error bar.

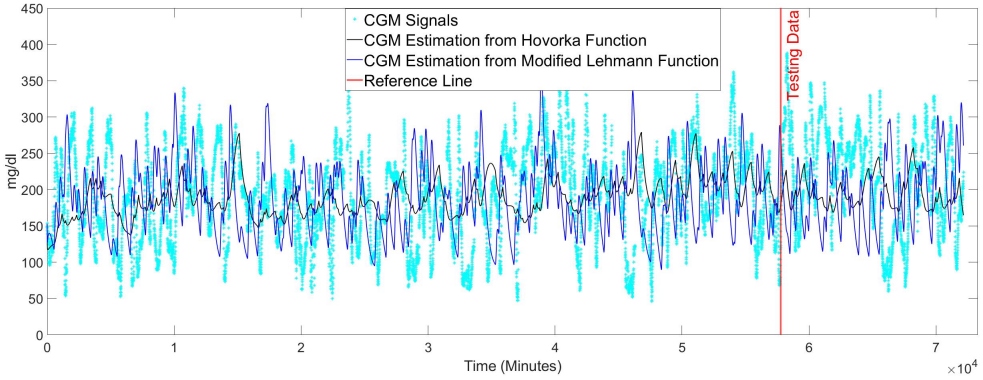


**Figure 3.19** Comparison plot of 2 physiological models with CGM signals for data contributor 570.

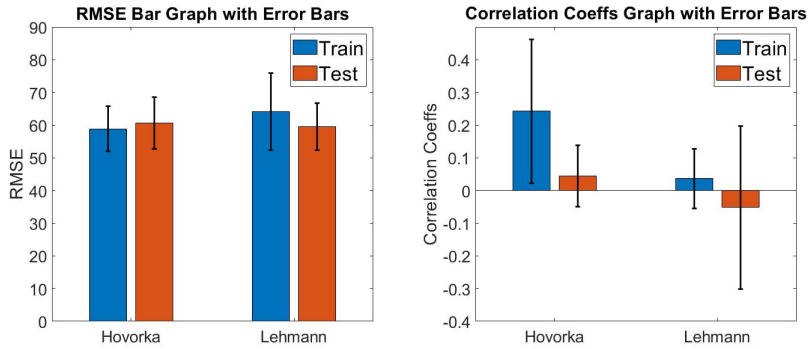


**Figure 3.20** The bar graph evaluates 2 physiological model performance among 6 data contributors, the mean values are reported with black error bar represents the standard deviation.

CHAPTER

# 4

# HYBRID APPROACHES

## 4.1 Motivation

T1D is a chronic disease that requires BG monitoring to prevent hyperglycemia or hypoglycemia. While the meal events usually being the main cause of BG increasing, CGM can also be affected by physical activities, insulin level, and other factors such as stress, illness, and sleeping quality. All those factors may lead to a complex physiological system to model the BG dynamics for a real-world dataset.

Managing the complex system of the physiological model is challenging as it was presented in Chapter 3 and data-driven models are lack of interpretability. A hybrid approach can take the advantage of the interpretability of the physiological model and the accuracy from the data-driven model. A few approaches have been tested on spatio-temporal data, namely, equation learning, inverse-design, and physics-informed neural networks.

Recent publications on equation learning methods include employing the Relevance Vector Machines (RVMs) and Convolutional Neural Networks (CNN) to parameterize ocean mesoscale eddies [Zan20], which brought physical interpretation and lowered the computational cost of implementation. In [Lag20], the learning of partial differential equation models reduced the noise from spatiotemporal data and approximate partial derivatives. Regarding the sparse and noisy data, the equation learning [Nar20] predicted the unobserved dynamics and accurately infer the underlying equation. The universal differential equations [Rac20] used the machine-learnable structures to replace the unknown governing equations and generated predictions from the combined model.

The inverse-design method describes a procedure for finding the underlying equation from the model components and other potiential equations. In [Nar20], the Fisher-KPP was found by

using sparse linear regression to parameterize a list of potential right-hand side terms. The universal differential equations [Rac20] applied the same strategy that the missing functions were selected from a list of polynomials, their products, or trigonometric functions. When the minimum value of the loss function was obtained, the missing function was selected.

Lastly, the physics-informed neural networks method, where [Ray21] used physical equations in the loss function for partial differential equations which helped the neural network perform a better result. The biologically-informed neural networks [Lag00], as an extension of the physics-informed neural networks, used multilayer perceptrons in predicting vitro cell biology assay experiments while managing the convergence by the governing partial differential equations.

However, there are disadvantages for all three methods. The equation learning method is usually applied with the assumption that the input data is accurate and all model parameter is given, when the model parameters are not provided, the prediction of the neural networks along with estimated parameters may not have a biological meaning, i.e., the parameters are possibly outside the biological range to minimize the loss function. For the inverse-design algorithms, the underlying functions can only choose from a finite number of polynomials or other functions, but in the most complex physiological model, one function may include a lot of components. Finally, the physics-informed neural networks mainly focus on a single partial differential equation and the model parameters are always assumed to be reliable.

### 4.1.1 Efforts of Improving Meal Function

Many researchers focused on improving the accuracy of meal function in the physiological model. When collecting the real-world data, contributors with T1D are required to report meal sizes and meal time. However, there could be the case that the contributor forgot to record a meal, recorded in a different time, or meal size measurements were not accurate.

There are various methods for meal detection or correction. In general, all methods can be described in two categories, supervised learning and mathematical model. Supervised learning approach requires the training label of mealtime, where [Zhe20] trained the extended isolation forest where the simulated data are used for CGMs, meal, and insulin rate. Then the identified meal events are set to be training labels. For mathematical models, the mealtime can be identified by detecting the increase of the meal subfunction built from minimal model [Tur16; Zhe19] and stochastic differential equations [Mah17], or by CGM trajectory [Sam18] and glucose rate of change [Das08; Har14]. The meal size can be determined by moving horizon estimation [Mah18; Che19], fuzzy system [Sam17], and variable state dimension approach [Xie17].

However, there exist disadvantages to building an algorithm for correcting meal events. We have limited knowledge of patient-specific situations for building mathematical models, e.g. the duration between a person consumes a meal and BG level starts rising, and the relationship between the rate of change in CGM and the corresponding meal size while a person conducts other daily activities. In the real-world data, we usually do not have a target variable (training labels) for training a supervised learning model.

### 4.1.2 Limitation of Current Meal Functions

David Jenkins [Jen81] first created the concept of the Glycemic Index (GI) in the early 1980s, which classifies carb-containing foods into 3 categories: low GI, medium GI, and high GI. It is a ranking system that describes foods by the effect on BG levels. In the experiment of [O'C08], with insulin bolus controlled, the results indicate that the mean postprandial glycemia of T1D, who consume low GI test meals, was much lower compared to high GI meals for 3 hours observations.

However, the consumed carbohydrates in our OhioT1DM dataset do not indicate their GI classes. Use of a single meal function to model postprandial glycemia behavior may affect the overall accuracy of the physiological model. Thus, we seek for the approaches that reflect the relationship between meal consumption and CGM while independent from the physiological model.

### 4.1.3 Reconstruct Meal Functions Using Machine Learnable Approaches

The idea of Universal Differential Equation (UDE) [Rac20] utilizes the scientific models and machine learning structures for discovering the unknown function in a system. It assumes the part of the physiological model is missing and can be substituted by neural networks or other learnable architectures. With the knowledge of the rest of the physiological model parameters, the UDE is then trained with some optimizers that calculate the differences between estimations and real data.

In the next section, we will introduce an approach that was inspired from UDE, with the underlying meal function replaced by the reservoir dynamics, and provided an estimation of model parameters and biological interpretation as well. In addition, we introduce the meal correction algorithm which detects the dislocated meal events and moves to the correct positions. In section 4.3, the results are evaluated with the replacement of the meal events from the physiological model and the meal function from the hybrid model. In the discussion section 4.4, the evaluation performance is compared with previous models and proposes future improvement. Lastly, a contribution section 4.5 summarizes the major findings.

## 4.2 Methods

To construct a hybrid model, we assume the current meal activity is not correctly reflected in the CGM signals, which may be due to the inaccurate behavior of Rate of Appearance ($Ra$) for the reported meal events. We need to treat the Ra function as an unknown function that we can replace by a learnable architecture.

If an underlying function in the physiological model can be composed from other model variables, the following approach may be applied. The input of learnable architecture is a column vector that contains all the initial conditions of the physiological model, the output is a single value representing the value of the unknown function at the current time step. Then we use this value in the physiological model to compute the ODE solution for the next time step and feed the solution vector in the learnable architecture for computing the next value of the unknown function. By

repeating this process we can get the entire ODE solutions from the starting time point to the end, then we compare the computed solution with some data that we have, to update model parameters both in learnable architecture and the physiological model. This completed procedure represents one iteration.

For a complex system in which the underlying function may not be connected by other model variables, the learnable architecture may be trained by using other input sources, and the parameters in the learnable architecture along with physiological parameters are updated by an inverse problem.

We repeat the process until the algorithm converges, which means after changing the parameter values, the decrements of the loss function are lower than some threshold. Finally, we report the model parameters and evaluate the performance. We can also plot the unknown function to discover the dynamic and compare it with the previously assumed functions. We use one simple example of the SIR model which describes the spread of diseases in section 4.2.1, and then move on to the BG model.

The benefits of this method are it preserves the interpretability of the physiological model and the unknown meal function can also be recovered from the neural network. However, it lost the efficiency of supervised learning since the learnable architectures are treated as mathematical models.

### 4.2.1 An Implementation of SIR Model

The SIR model is a differential system that describes the behavior of a disease spread in susceptible individuals ($S$), infected individuals ($I$), and recovered individuals ($R$) without births, deaths, and immigration population. The full model is given by the Equations (4.1), where $\beta$ represents the infection rate from inflection individuals to susceptible individuals, $\gamma$ is the recovery rate represents the percentage of infected individuals become recovered individuals for any given day, and $N$ is the total number of population.

$$\begin{aligned}
\dot{S}(t) &= -\frac{\beta I(t)S(t)}{N}, \quad S(0) = S_0, \\
\dot{I}(t) &= \frac{\beta I(t)S(t)}{N} - \gamma I(t), \quad I(0) = I_0, \\
\dot{R}(t) &= \gamma I(t), \quad R(0) = R_0, \\
N &= S_0 + I_0 + R_0.
\end{aligned} \tag{4.1}$$

If we are uncertain how a certain disease infects the population, and replace it with a neural network function $f(S, I, R, w)$, the neural network will require some knowledge of the amount of all three classes, and the model parameters (weights and biases) that are denoted by $w$. Now the

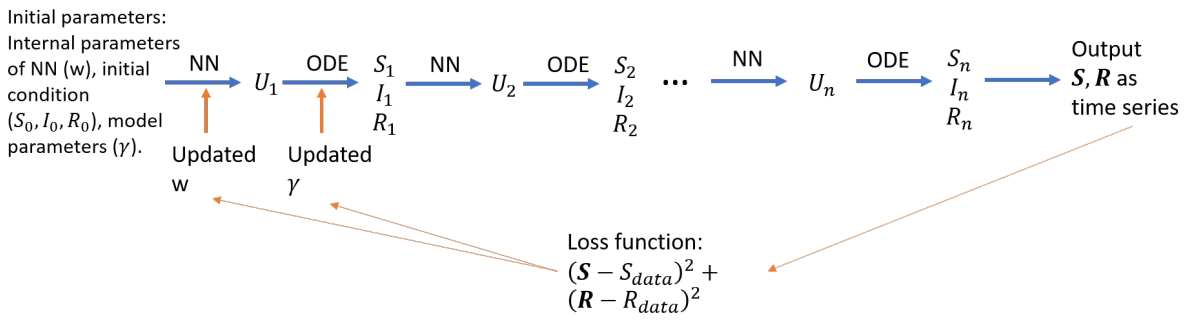


**Figure 4.1** The diagram of hybrid model of SIR problem, where the NN represents the neural network and $w$ is the weight matrix in the NN.

differential system with the unknown function can be described in (4.2).

$$\begin{aligned} \dot{S}(t) &= -f(S,I,R,w), \quad S(0) = S_0, \\ \dot{I}(t) &= f(S,I,R,w) - \gamma I(t), \quad I(0) = I_0, \\ \dot{R}(t) &= \gamma I(t), \quad R(0) = R_0, \\ N &= S_0 + I_0 + R_0. \end{aligned} \tag{4.2}$$

In this implementation, we also assume that we have the data for susceptible and recovered individuals measured throughout the observation, and the recovery rate $\gamma$ is unknown in the SIR model.

In Figure 4.1, it shows the methodology of the hybrid model, where we use $U_i$ to represents the unknown function $f(S,I,R,w)$ at a given time $t_i$, and $\mathbf{S}, \mathbf{R}$ represents the ODE solution time series that composed by $[S_0, S_1, S_2, \ldots, S_n]$ and $[R_0, R_1, R_2, \ldots, R_n]$ for a total length $n+1$ time series. We first proceed with the blue arrow, the first step is to obtain the next time step of the unknown function $U_1$ from the initial condition $[S_0, I_0, R_0]$, and next we compute the ODE solution of the current step as a column vector $[S_1, I_1, R_1]^T$ using the value of $U_1$. By repeating this procedure, we complete the ODE solution series with the time from $t_0$ to $t_n$. Then, we continue with the orange arrow where the loss function is defined as Mean Squared Error (MSE) between ODE solution and real data. We search the parameter values by lowering the loss function until it is stable. Finally, we update the parameter values into our hybrid model. We treat the completion of blue and orange arrows as one iteration. The algorithm stops when the loss function decreases to a certain threshold, or changing parameter values will not improve the performance.

The algorithm of the hybrid model is described in Algorithm 1, besides the initial parameters, we also require the initial guesses for $U_0$, where the value of $U_0$ can determine the convergent rate of the algorithm with different initial guesses. Similarly for initial conditions $[S_0, I_0, R_0]$, the closer to the true value, the faster it will let the hybrid model converge.

We show an experimental result for the SIR hybrid model, where we assume the observation time is 50 hours. We record the disease spread every 6 minutes, so we will have a time series with a

**Algorithm 1** Hybrid Model for SIR

**Output:** Loss
**Input:** $U_0$, $[S_0, I_0, R_0]$, $w$ (weight matrix),$\gamma$ (model parameter), tspan, $S_{data}$, $R_{data}$
**for** t in tspan **do**
    $U_1$=Neural Network($[S_0, I_0, R_0]^T, w$)
    $U_f$=linear interpolate $[U_0, U_1]$ in [tspan(t), tspan(t+1)]
    y=ODE fixed step solver($U_f$,$\gamma$)
    $U_0 = U_1$
    $Y_{pred}$.append($[S_t, I_t, R_t]$)
**end for**
Loss=$\sum_{i=0}^{n}((Y_{pred}(1,:)-S_{data})^2+(Y_{pred}(3,:)-R_{data})^2)$.

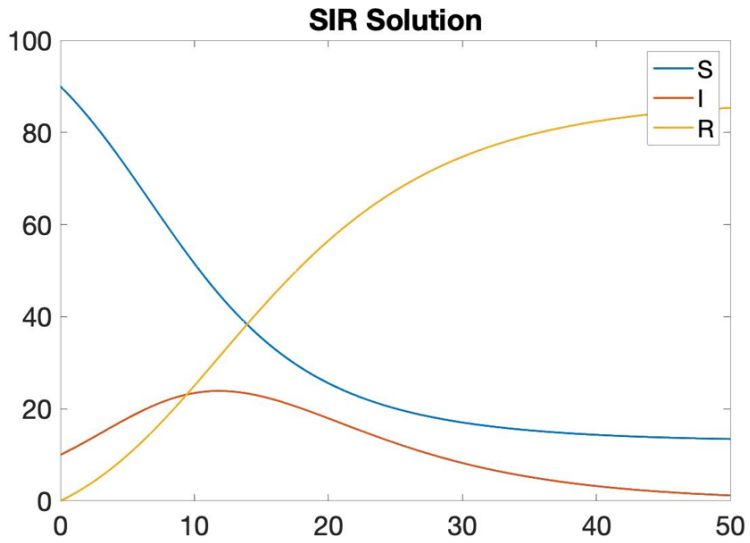


**Figure 4.2** The solution of the SIR model is solved by the fourth order fixed step size ODE solver, the initial values are $[S_0, I_0, R_0] = [90, 10, 0]$.

length of 501 (including one starting point at the beginning). The infection rate $\beta = 1/3$ and recovery rate $\gamma = 1/7$. We assume there are 100 individuals and 90 of them are susceptible, 10 are infected, and 0 is recovered. The behavior of the three classes is presented in Figure 4.2.

Then we generate the noisy time-series of susceptible and recovered individuals for testing the performance of our hybrid model. We add randomized noise of level 5 of each class as shown in Figure 4.3.

We can now test the hybrid model, first we test on a small neural network with 1 hidden layer, there are 5 neurons in the hidden layer with ReLU activation function and output the prediction by a linear output layer. Then we test on a larger neural network with 2 hidden layers by adding a 5 neurons hidden layer with ReLu activation function. Both models use the same initial $U_0 = 3$ and initial conditions $[S_0, I_0, R_0] = [90, 10, 0]$. The performance of the two models are presented in Figure 4.4 and 4.5, where the 3 layers neural network model obtained a better result with a longer running

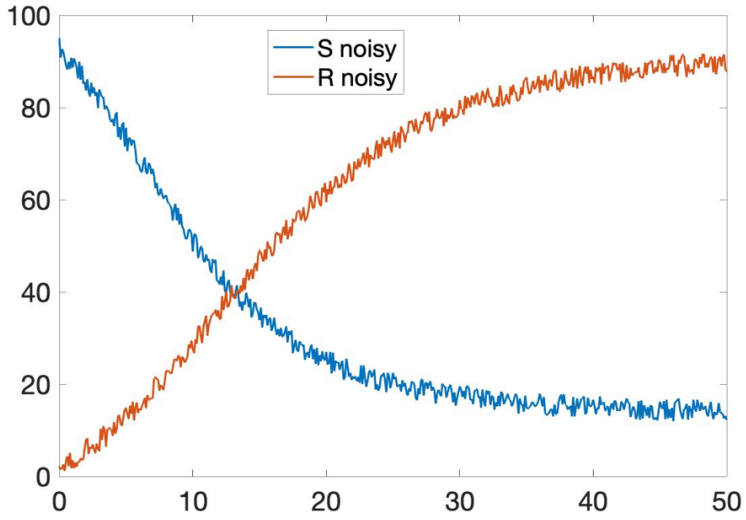


**Figure 4.3** The class of susceptible $S$ and recovered $R$ individuals with noise in the experiment.

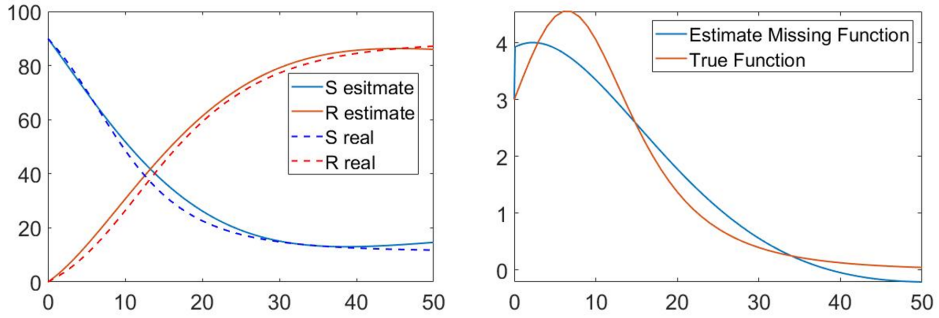


**Figure 4.4** The estimation of SIR hybrid model using 2 layers neural networks, where the estimated recovery rate $\gamma = 0.1982$ and the output from the loss function is 5301.

time.

### 4.2.2 Meal Correction Algorithm

As the results presented in Chapter 3, a few opposed trends of the predictions compared to CGMs were made due to the possibility of mislocation of the meal event. In this section, a meal correction algorithm is introduced to detect the potential meal events and move the misaligned meal events to the right location. In the CGM dynamic plot of data contributor 570, we assume that the large increment of the CGM in the daytime is due to the meal intake.

The OhioT1DM dataset contains two meal sources, self-reported meal from smart phone and bolus meal when the bolus was delivered, which can be misleading. Figure 4.6 shows the plot of 2 meal sources for the entire dataset, the blue + symbol represents the self-reported meal by the data contributor, and the red cycle represents the meal measurements with the bolus injection recorded

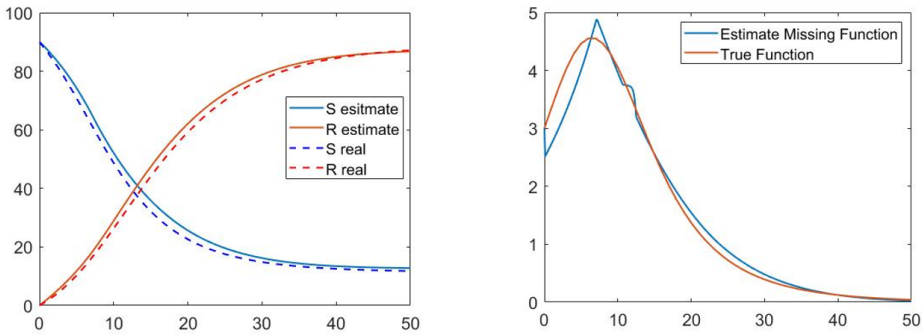


**Figure 4.5** The estimation of SIR hybrid model using 3 layers neural networks, where the estimated recovery rate $\gamma = 0.2023$ and the output from the loss function is 3828.

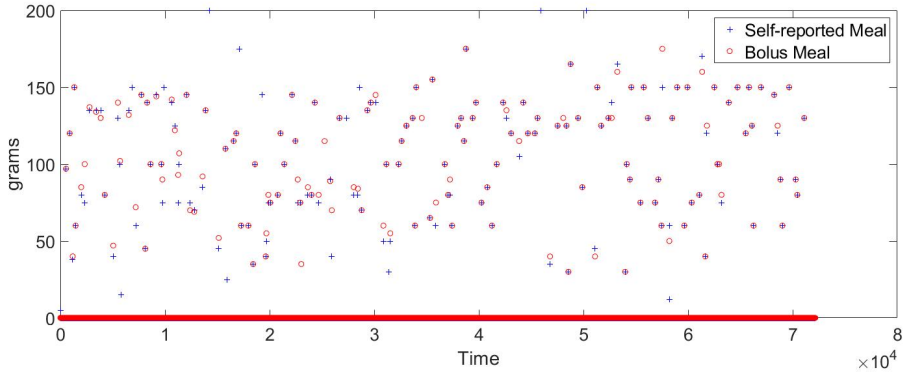


**Figure 4.6** The comparison plot from 2 meal sources for the entire dataset of data contributor 570. If at a time, a meal event did not report in the dataset, we label the meal event as 0.

by the nurse. From the figure, we observe that some of the events are describing the same meal event with the time and meal size the same, but there are other meal events that are recorded in a different time or different size where the two symbols in the figure do not match. We cannot directly combine these meal sources since a lot of duplicate meal events would exist. Thus, a meal correction algorithm is necessary to combine two meal sources and correct the mislocated meal events.

#### 4.2.2.1 Filtering Method

First, the missing values in the training proportion of CGMs are imputed using 3rd order polynomial interpolation for every minute (in Figure 4.7 shows an imputation example). If there are negative values generated by cubic interpolation, then the linear interpolation is used instead.

Then the filtering method was applied to CGMs to lower down the sensor noise and make the meal events more observable. We introduce the low pass Butterworth filter [But30] which can provide the maximally flat response, and the meal detection is examined on filtered CGM signals.

In the filter design, the ideal low pass filter preserves the low-frequency signals within the pass band and filters out the high frequency above the threshold. However, in practice, if the continuous

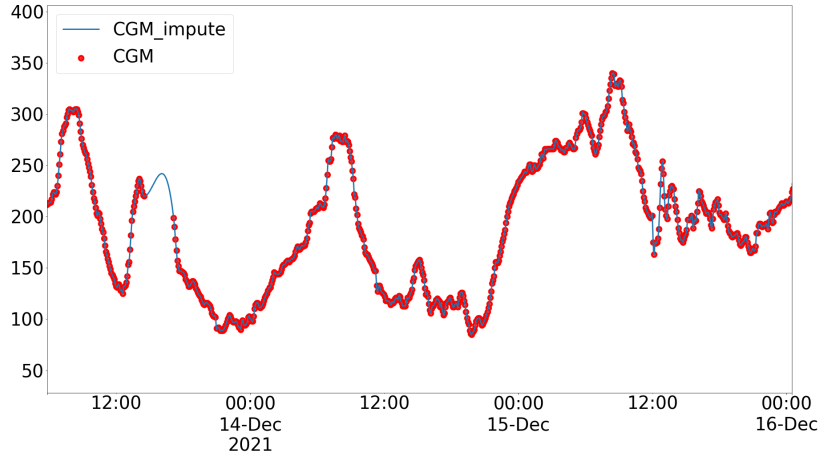


**Figure 4.7** Example of 3rd order polynomial interpolation for continuous blood glucose monitoring (CGM) around 2 days, the red dots represent the raw CGM signals collected every 5 minutes, the cyan curve represents the imputation results for every minute.

signals reach the threshold of the pass band it will get attenuation as it is shown in Figure 4.8.

Butterworth filters [But30], also named maximally flat filters, have a flat frequency response in the passband without peaking [Ell12]. The formula for Butterworth filters depends on the order of the filter is odd (Equation 4.3) or even (Equation 4.4), where Π represents a series of products, M is the order of the filter, $\omega_N$ is the normalized cutoff frequency. The effects of frequency response for different orders of Butterworth filters are in Figure 4.9.

$$T(s)=\left(\frac{\omega_N}{s+\omega_N}\right)\prod_{i=1}^{(M-1)/2}\left(\frac{\omega_N^2}{s^2+2s\cos(\theta_i)\omega_N+\omega_N^2}\right),\quad \theta_i=i\times 180/M. \tag{4.3}$$

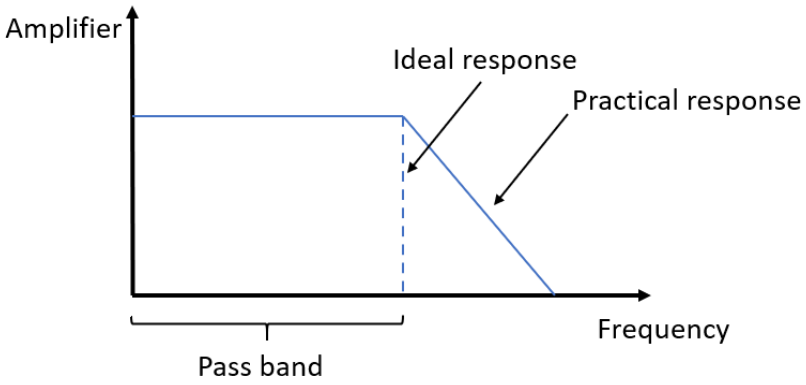


**Figure 4.8** The schematic of ideal and practical low pass filters.The ideal response is presented in the dash line after passed the band width, the practical response is in blue line which decay to 0 with a linear rate.

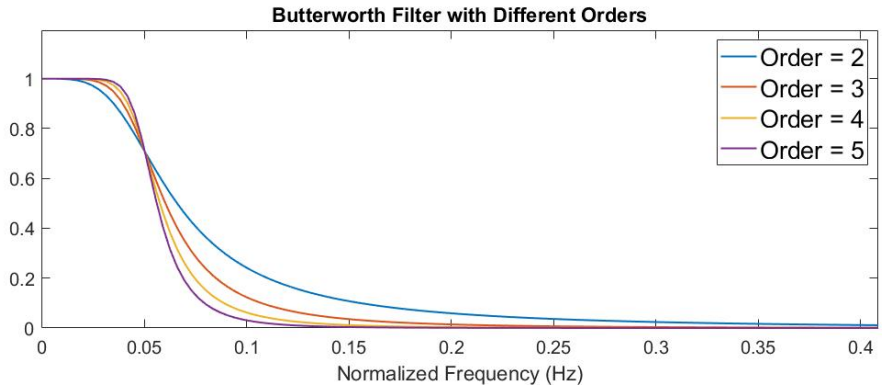


**Figure 4.9** Comparison with different orders of Butterworth filter.

$$T(s)=\prod_{i=1}^{M/2}\left(\frac{\omega_N^2}{s^2+2s\cos(\theta_i)\omega_N+\omega_N^2}\right),\quad \theta_i=(i-0.5)\times 180/M. \tag{4.4}$$

We decided to continue using 570 as an example to test the performance of the meal correction algorithm. After the third order polynomial interpolation of CGM signals, we apply the Butterworth filtering method for dampening the high frequency signals. We choose the order 5 to reduce the number of frequent peaks and falls due to noises. The first row in Figure 4.10 shows the magnitude of the Fourier transform of imputed CGM signals. The high frequency signals are squeezed in the first 2% of the total time length. Next, we plot the normalized frequency and only focus on the first 2% of the discrete Fourier transform to determine the cutoff frequency location. From the second row in Figure 4.10, we obtain approximately at 0.05% the separations of the high frequency and low frequency signals. Thus, we apply the low pass Butterworth filter with order 5 and low passband frequency as 0.05%. The final filtered CGM is shown in Figure 4.11, compared with the imputation results, the filtered CGMs (red curves) are much smoother than the original. The filtered CGMs are then used for the meal correction algorithm later.

#### 4.2.2.2 Confirmation of Existing Meal Events

The meal correction algorithm has two steps, confirmation of the existing meal events from self-reported and bolus measurements, and detection of unannounced meal events based on the observation from filtered CGM signals.

In the confirmation part, we first gather all known meal intake measurements from self-reported and bolus events (Figure 4.12). If the same meal event is recorded by both devices at the same time, we keep the measurements from the self-reported method. Next, we move the time index of meal events to the correct location (Figure 4.13). We decide to move the recorded event to the nearest valley (or the bottom of the "U" shape) in the filtered CGM signals by checking distances of the front and back of the recorded meal events. The pseudo code for the algorithm can be found in the Appendix A at Algorithm 2. Last, we remove the duplicate events due to relocating, where we

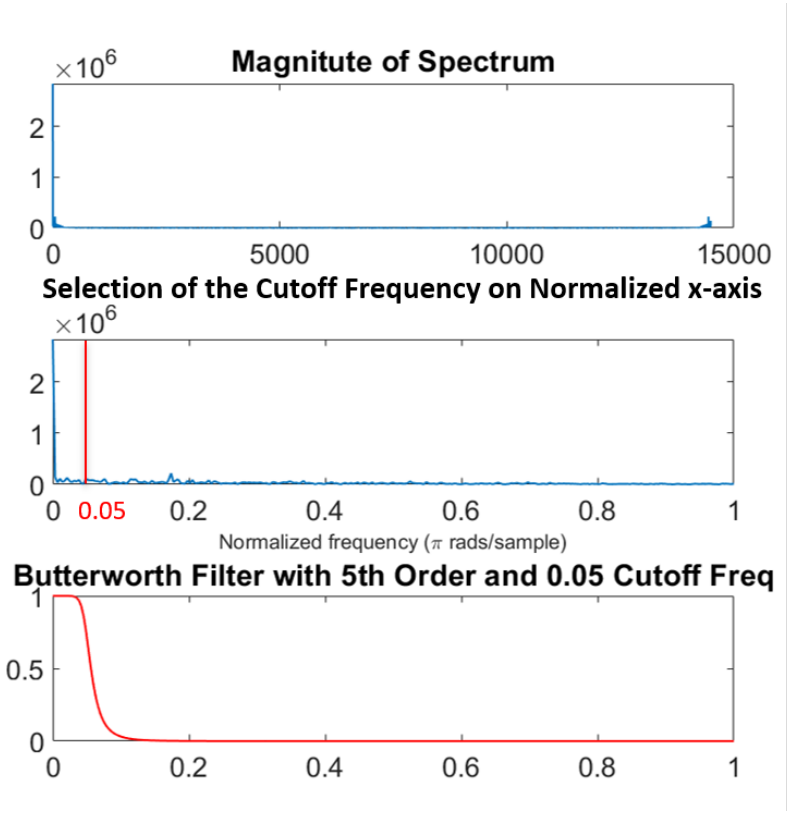


**Figure 4.10** The selection of a low pass 5th order Butterworth filter with the cutoff frequency is at 0.05% of the signals.

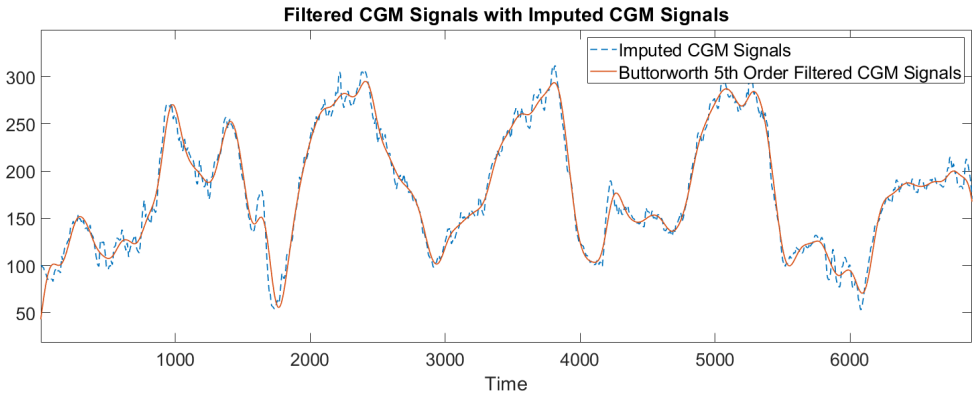


**Figure 4.11** Comparison of filtered signals and original signals in the first 6000 data points.

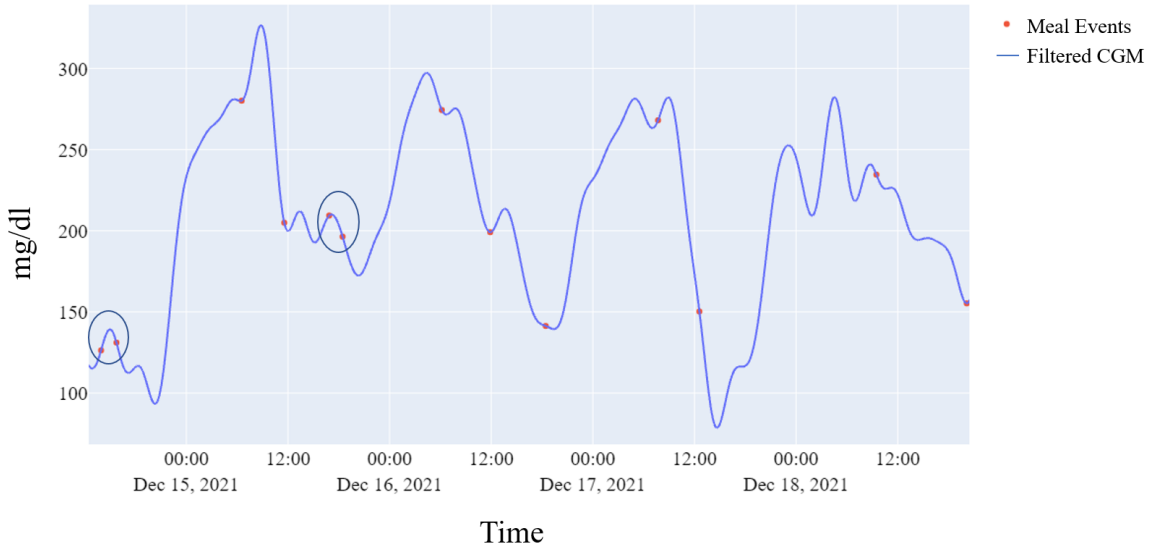


**Figure 4.12** Meal events location plot including the self-reported meal events and bolus meal events.The red dot represents the time meal was taken. The black oval cycle represents the cases where both sources measured the same meal event.

compare the relocation distances from the original time index and discard the one with further distance.

#### 4.2.2.3 Detection of Unannounced Meal Events

In the second part of the meal detection algorithm, we wish to detect unannounced meals. We assume during the daytime, there may exist additional meal events that individual data contributors forget to report. Therefore, we propose an algorithm that deals with the situation where there is a fast increase in CGM without any meal event associated with it.

In Algorithm 4 at Appendix A, our criterion for detecting new meals only focuses on the regular size meal instead of a snack. We only look at the in a day the data contributor reports less than 4 meal intakes during the daytime and separate each meal by 4 hours. In addition, the detected mealtime should correspond with CGM less than 300 mg/dl, because the meal consumption when BG levels are high would lead to hyperglycemia.

We also propose other indicators for meal detection, such as $\tilde{g}$ and $\Delta g$. Since CGM is monitored every 5 minutes, the $\tilde{g}$ is defined as $\tilde{g} = CGM(t+5) - CGM(t)$, representing the change of CGM from the current time point to the next time point. Similarly, the $\Delta g$ is defined as $\Delta g = \frac{CGM(t+30) - CGM(t)}{30}$, measures the rate of change for 30 minutes.

Similarly to Algorithm 2, we wish to detect the meal located at valleys in the CGM curves. In addition, we add the indicator for $\Delta g > 0$ and $\tilde{g} \geq 0$, so that the algorithm selects the location at the falls and only focuses on large peaks and falls. The results of detection of meal events are presented in Figure 4.14. Thus, the meal correction algorithm is complete with 2 parts, the green squares

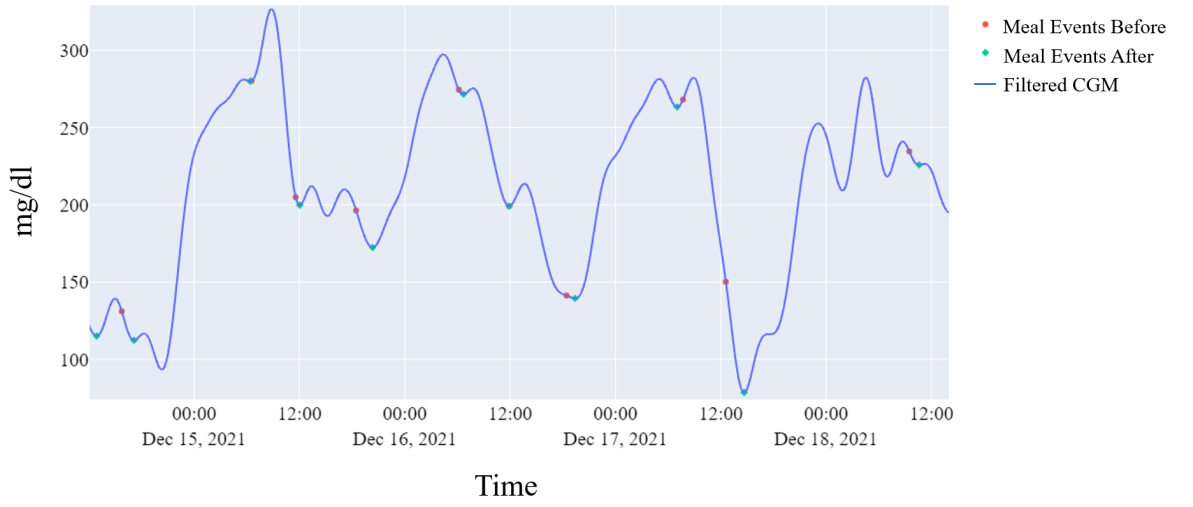


**Figure 4.13** Comparison of relocating meal events and original meal events using the Algorithm 2. The red cycles are the combined meal events from bolus and self-reported with the duplication removed. The green diamonds are the identified new location on the CGMs based on the red cycles.

represent the meal events in the relocation algorithm, where the carbohydrate size corresponding to each event is also recorded. The other part is unannounced meal events, represented by red dots, which does not contain the specific value of carbohydrate grams. We impute all those unannounced meal events with the carbohydrate size as 50 grams as the value of a medium size meal.

### 4.2.3 Reservoir Dynamics of Meal Function

In the data driven model, we introduced the bidirectional reservoir computing (BRC) model. The prominent performance of the model in predicting the BG levels has the potential to combine with the physiological model for both accuracy and interpretability. The BRC can be used to replace the meal function, which would be one of the most effective terms in BG prediction of a physiological model.

The full architecture of the BRC model is described in 2.4.2, in the hybrid approach, we used the key component of the complete BRC model to obtain the estimation of the meal function. Similar to the SIR implementation described in 4.2.1, the hybrid model we proposed in this work is to replace the underlying meal function with reservoir dynamics and feed it in the physiological model.

#### 4.2.3.1 Reservoir Dynamics Description

The detailed setup for reservoir dynamics is introduced in Chapter 2.4.2. Instead of using the ridge module as representation, we use the last module where the last slice of the second dimension in the state matrix after PCA reduction is the representation of reservoir dynamics, denoted by $R_{repr}$. The decoder part is ignored since the reservoir networks are considered as a simulation model for the underlying function.

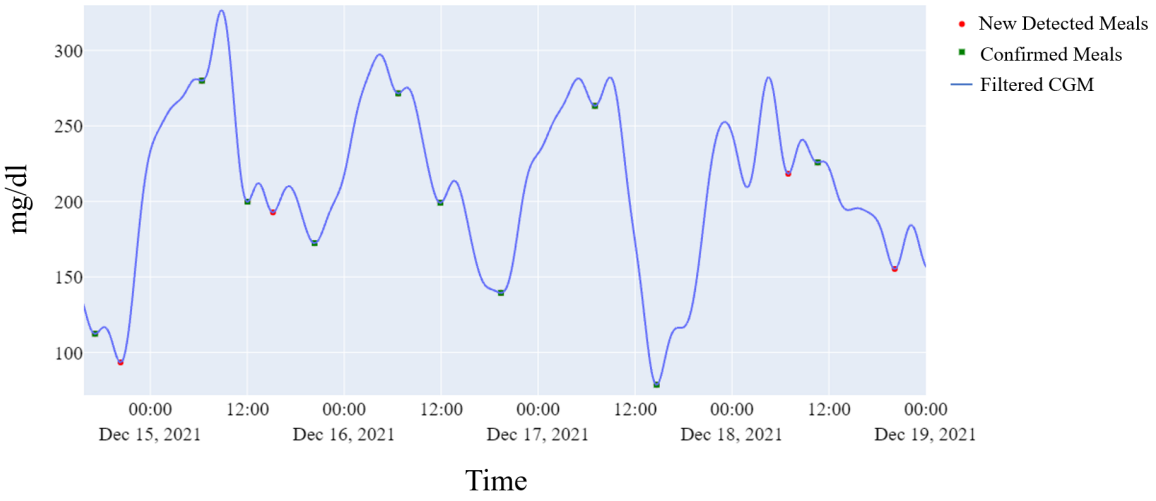


**Figure 4.14** Meal events distribution (red circles) after the detection Algorithm 4, green squares are from the meal confirmation algorithm, the complete meal events after the meal correction algorithm contains both red and green markers.

Here, we define a weight matrix $W_p$ that contained the estimated parameter in the reservoir dynamics with shape $[n_{PC}, n_{tps}]$, then simulated the model function $Ra$ by $Ra = R_{repr} W_p$. Due to the meal function representing the meal appearance rate which should be non-negative, the final outputs are the absolute values of $Ra$. Using the last module in the representation stage would largely reduce the estimated parameters compared to the ridge module.

Under this framework, the total estimated parameters are $n_{PC} \times n_{tps}$, which is the product of the number of principal components in PCA and the time steps.

#### 4.2.3.2 Improvement on Reservoir Dynamics

We add some options for the reservoir dynamics to make it suitable for meal function simulation.

**Meal Duration.** Due to the input sources of reservoir dynamics being the same as meal function in the physiological model, two more parameters can be added to simulate the rate of appearance function, namely $T_{snack}, T_{meal}$. The input meal event series is a sparse time-series data where except the meal value recorded at a mealtime (one single time point), all other elements are 0. Therefore, the time-series of meal events may bring difficulty to the reservoir simulation since the sparse output would result in small fluctuations in CGM dynamics only when meal events occurred.

The parameter $T_{snack}$ described the time duration of a snack-size carbohydrate digestion time, which could range from 0 minutes to 20 minutes. Similarly, $T_{meal}$ represents the regular meal size of carbohydrate digestion time and could range from 20 to 100 minutes. Thus, the input time-series representing the meal information becomes a rectangular shape piecewise function that contains the meal size and meal duration and is distinguished by regular size meals or snack size meals.

**Smooth Filter**. After the random initialization of weight matrices, the simulation of the meal

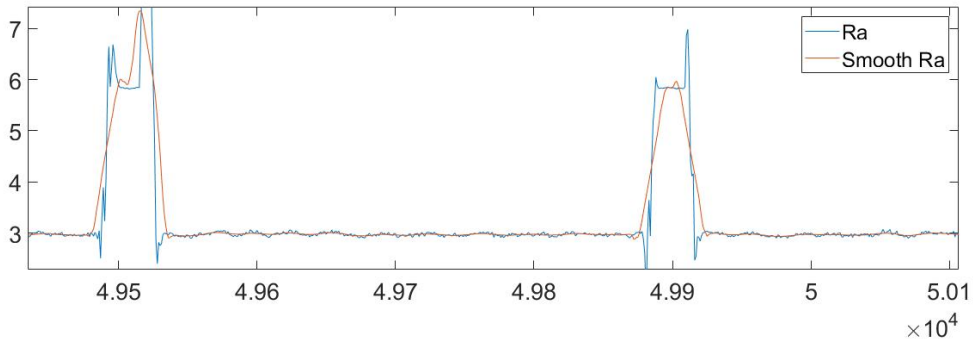


**Figure 4.15** A zoomed in plot for comparing the noisy simulation from reservoir dynamics (cyan curve) with smoothed meal function $Ra$ when the span of moving average is 20 (orange curve).

function in the first few iterations may be very noisy. If the function later feeded into ODE as an input, then the high level noise would increase the solving time and the number of iterations for convergence. Thus, a moving average filter is introduced to smooth the meal function simulated from reservoir dynamics by providing the span of the moving average $l$. The first few elements of the smoothed function $y_{out}$ is defined as follows, where we assume the $y_{in}$ represents the original noisy function.

$$\begin{aligned} y_{out}(1) &= y_{in}(1), \\ y_{out}(2) &= \frac{y_{in}(1)+y_{in}(2)+y_{in}(3)}{3}, \\ y_{out}(3) &= \frac{y_{in}(1)+y_{in}(2)+y_{in}(3)+y_{in}(4)+y_{in}(5)}{5}, \\ &\cdots \\ y_{out}(j) &= \frac{1}{l}\sum_{i=j-\lfloor\frac{l}{2}\rfloor}^{j+\lfloor\frac{l}{2}\rfloor} y_{in}(i), \end{aligned} \tag{4.5}$$

where $j$ is the j-th element in the output $y_{out}$. If we have an even number of span $l$, then the number will be reduced by 1.

By adding the option of a smooth filter, the shape of the simulated meal function $Ra$ may also be changed by tuning the number of spans. Figure 4.15 shows an example of the $Ra$ function with the span in the smooth filter $l = 20$, where the irregular shape of $Ra$ when a meal event happens is smoothed to the shape similar to triangles.

**Shift Operator**. A shift operator is added for estimating the meal function due to the duration after meal consumption and BG levels start to rise. According to [Zur19], the duration between meal consumption and BG level rising will depend on the GI and glycemic load (GL), where GL is the production of the quantity of carbohydrate contained in the consumption and GI. The duration of BG rising can be from 0 to 30 minutes for a person without diabetes, and the increment of BG levels stops after the maximum 6 hours.

Thus, the shift operator works as follows. After we obtain the simulation of $Ra$ in the reservoir

dynamics, the $Ra$ is padded with 0 at the beginning with the length $t_{shift}$ determined by the parameter estimation. The $Ra$ maintains the same length as the time grid by removing the last $t_{shift}$ elements in $Ra$. The range of the selection for $t_{shift}$ is from 0 minute to 50 minutes which allows for the incorrect meal index during a meal event.

### 4.2.4 Methodology for Hybrid Approach

The complete physiological model with six subsystems is presented in Chapter 3. We remove the meal function from either Lehmann [Leh92] or Hovorka [Hovly] and replace it with reservoir dynamics that follows the description in the section 4.2.3.

We test 2 reservoir dynamics, one small reservoir network with $n_{PC} = 3$, while the other hybrid model we use $n_{PC} = 10$.

As described in the previous section, the shift operator and smooth filter is tested on both models, the normalization is also considered as an option for regulating the input meal events in the reservoir dynamic. We present below in Table 4.1 the total estimated parameters in the different settings, where the $n_{phy}$ represents the sensitive and identifiable parameters in the physiological model without parameters from the meal function.

**Table 4.1** Table of the possibility options that affects the number of estimated parameters in the inverse problem. $n_{PC}$ represents the number of principal components, $n_{tps}$ stands for the value of time step, $n_{phy}$ is the number of selected parameters in the physiological model, $n_{meal}$ presents the parameters in meal duration option. We use Res. to abbreviate the reservoir dynamics. Shift and Smooth are two options, and 0 represents the option not chosen.

| Size | $n_{PC}$ | $n_{tps}$ | $n_{phy}$ | $n_{meal}$ | Shift | Smooth | Total parameters |
|---|---|---|---|---|---|---|---|
| Small Res. | 3 | 3 | 3 | 2 | 0 | 0 | 14 |
| Small Res. | 3 | 3 | 3 | 2 | 1 | 0 | 15 |
| Small Res. | 3 | 3 | 3 | 2 | 0 | 1 | 15 |
| Small Res. | 3 | 3 | 3 | 2 | 0 | 1 | 16 |
| Large Res. | 3 | 10 | 3 | 2 | 0 | 0 | 35 |
| Large Res. | 3 | 10 | 3 | 2 | 1 | 0 | 36 |
| Large Res. | 3 | 10 | 3 | 2 | 0 | 1 | 36 |
| Large Res. | 3 | 10 | 3 | 2 | 1 | 1 | 37 |

At present, we have 3 meal sources, which are meal events from self-reported smartphones, bolus meal events, and meal events after the meal correction algorithm. We select the training proportion of the meal events and use a vector to represent each source in the one minute time grid. If a meal event is associated with a certain time, then we fill in the meal value, otherwise, we use 0 to represent no meal consumption at that time.

We can alter the number of input sources as an additional option in the reservoir dynamics, and obtain the simulation of $Ra$ by providing the number of principal components, number of time steps, option of smooth filter, option of shift operator, option of normalization of input sources, and

initial weight matrices. The next step is to feed $Ra$ in the physiological model as an input function, and solve it with other subsystems. The loss function is the same as defined in Chapter 3, where the amplifier $\kappa$ of the penalty term in Equation (3.51) can be adjusted depending on the performance of the hybrid model. The same optimization techniques introduced in Chapter 3.1.3.3 are used in the hybrid model: Quasi-Newton method with BFGS and implicit filtering. The model was first optimized using implicit filtering with given boundaries and then updating the parameters in the Quasi-Newton method with BFGS. The evaluations metric RMSE (2.12) and correlation coefficient (3.41) are obtained by the difference between simulation and true CGM signals for training and testing.

## 4.3 Results

In this section, we first present the results by using the updated meal events from the meal correction algorithm in the physiological models, then discuss the performance of the hybrid model by replacing the meal function in the physiological model with reservoir computing.

### 4.3.1 Results of Updated Meal Events in Physiological Model

The meal correction algorithm is only applied for detecting the incorrect and unannounced meal events in training CGMs. For the testing set, we keep the meal event measurements as the self-reported meal events.

The evaluation results were performed on physiological models introduced in Chapter 3 with the input meal source replaced by the output of the meal correction algorithm in the training process. The loss function and optimization techniques for parameter estimation follow the same procedure as described in Chapter 3, where we use the implicit filtering method with the same boundary conditions and then use the Quasi-Newton BFGS algorithm for convergence. The estimated values for parameters for each physiological model are shown in Table 4.2. After the parameter values are obtained, the values are then used for predicting CGMs for testing series. In the updated performance using Hovorka function, the RMSE for training and testing are 61.96 and 70.58, and correlation coefficients for training and testing are 0.17 and 0.02, respectively. Regarding the performance of modified Lehmann function, the training and testing RMSE are 76.69 and 102.54, where the correlation coefficients are -0.02 and -0.41, respectively.

**Table 4.2** Parameter values of unknown parameters in both models after meal correction algorithm obtained in the training phase, where Lehmann M. is the abbreviation of physiological model with lehmann meal function, and Hovorka M. stands for physiological model with Hovorka meal function.

| Model | $C_{bio}$ | $K_m$ | $b_{c2}$ | $T_m$ | $b_{c1}$ | $T_s$ | $I_b$ | $I_{pb}$ | $I_{lb}$ | $I_{sc1b}$ | $I_{sc2b}$ | $G_{tb}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lehmann M. | 4 | 300 | 1.84 | 98.91 | 4 | 10 | 3 | 3 | 3 | 3 | 3 | 100 |
| Model | $C_{bio}$ | $K_m$ | $b_{c2}$ | $T_{max}$ | $b_{c1}$ | $I_b$ | $I_{pb}$ | $I_{lb}$ | $I_{sc1b}$ | $I_{sc2b}$ | $G_{tb}$ | |
| Hovorka M. | 8.48 | 202.18 | 0.51 | 475.23 | 2 | 3 | 3 | 3 | 3 | 3 | 100 | |

In Figure 4.16, we plot the comparison of the physiological model with Hovorka function regarding of the changing of meal events. From the plot, we observe the prediction results are both small in magnitude and stayed in the region of [150,250] mg/dl mostly. It might not be a good simulation model for T1D people, where their BG levels vary largely every day affected by different activities. Compared the evaluation metrics with result in Chapter 3.3.1, the training performance is better with the previous meal events, but the meal correction algorithm improved the testing performance. This may due to the parameters trained by the update meal events has a better generalization ability. Lastly, the subsystem behaviors are plotted to compare the model performance. In Figure 4.17, the plasma insulin, endogenous glucose production, and glucose utilization are shifted up or down due to the parameter values being slightly different. Then the rate of appearance model showed the major difference, where there are less meals identified by the meal correction algorithm and occurred at different locations. Regarding the parameter values in meal function, the $T_{max}$ is out of the biological range (similar behavior was observed in Chapter 3). Thus, the physiological model with Hovorka function may have limitations on predicting the real-world T1D dataset. The bar graph 4.18 of evaluation metrics among 6 data contributors shows the meal correction algorithm can slightly improve the physiological model performance on testing dataset.

Next, Figure 4.19 depicts the comparison plot for the physiological model with modified Lehmann function. The physiological model using update meal events has a smaller magnitude compared to the one using previous meal events, and sometimes can predict extreme values of CGM signals correctly. Compared to the results obtained in Chapter 3.3.2, we found the updated meal events largely improved the model behavior in the training phase. The RMSE is slightly increased in testing but the correlation coefficient is improved. In the subsystem plot 4.20, we observed a similar behavior as shown in the Hovorka model, where the number of updated meal events are less than previous, and the other subsystems slightly vary due to the changing parameter values. Finally, the comparison plot of evaluation metrics among 6 data contributors is shown in Figure 4.21, where we observed the mean of training error is decreased and the standard deviations are larger in the updated meal events. This may due to the meal correction algorithm can only improve the performance when the previous evaluation results are poor, and may dampen the good performance when the meal events can be correctly reflected by CGM patterns.

Overall, the meal correction algorithm provides the possibility of identifying the meal events and slightly improves the model performance. The results shown in both physiological models may indicate the real-world meal data is not always accurate. There are limitations for this meal correction algorithm such as the meal size cannot be identified and the algorithm cannot be applied to testing series. However, we observe the importance of the meal function in the physiological model after conducting this experiment.

### 4.3.2 Results of Hybrid Model

We construct a hybrid model which replaces the meal function previously defined as modified Lehmann or Hovorka function with reservoir dynamics, where characteristics of unknown parame-

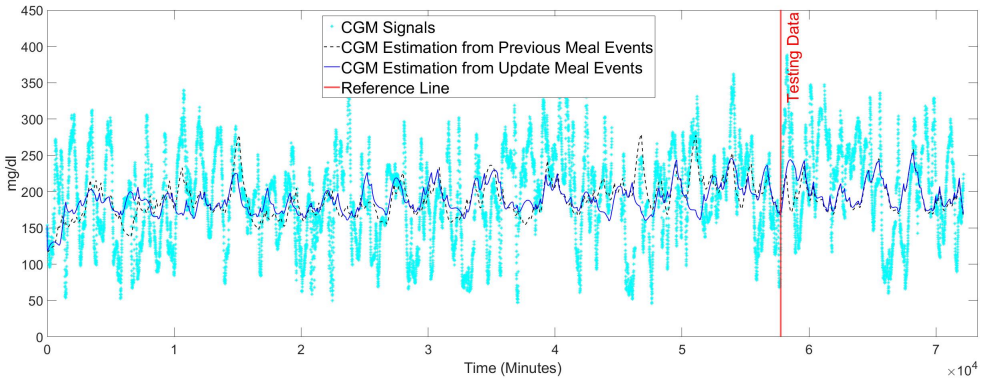


**Figure 4.16** Comparison plot of model performance with CGM signals with/without the meal correction algorithm applied physiological model with Hovorka meal function for 570. The left side of the reference line is training, and the right side is testing. The meal correction algorithm does not apply to testing CGMs.

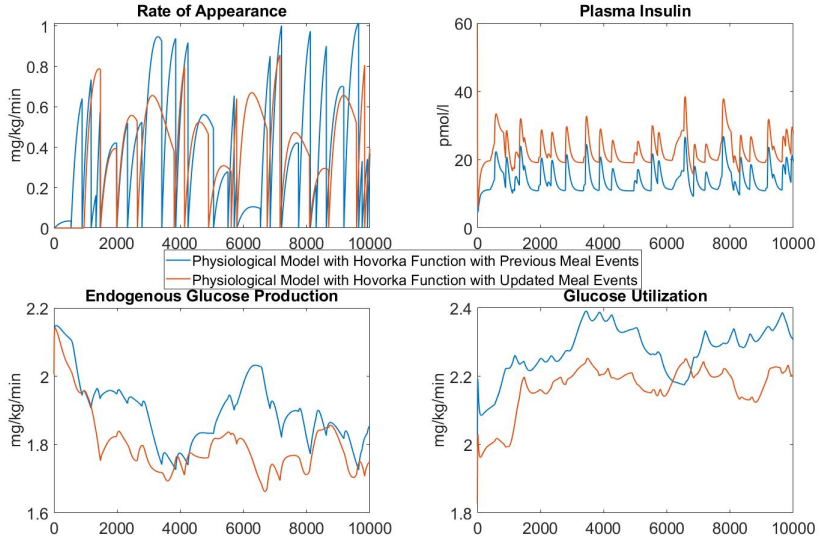


**Figure 4.17** The subsystem dynamics of both physiological models after the meal correction algorithm for 570 using Hovorka meal function. The first 10000 data points (around 7 days) dynamics are presented.

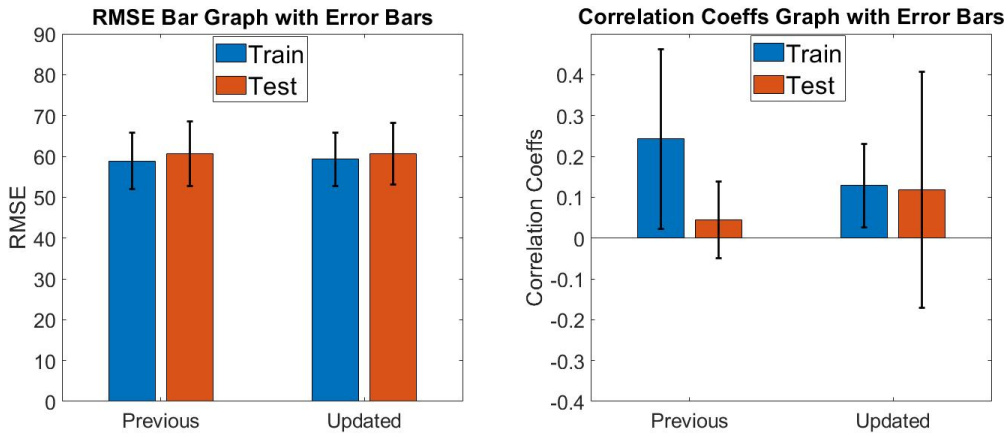


**Figure 4.18** Bar graph of model performance in training and testing among 6 data contributors using Hovorka meal function. First presented the evaluation results for the physiological model with the raw meal data, then presented the evaluation results with the updated meal events with meal correction algorithm.

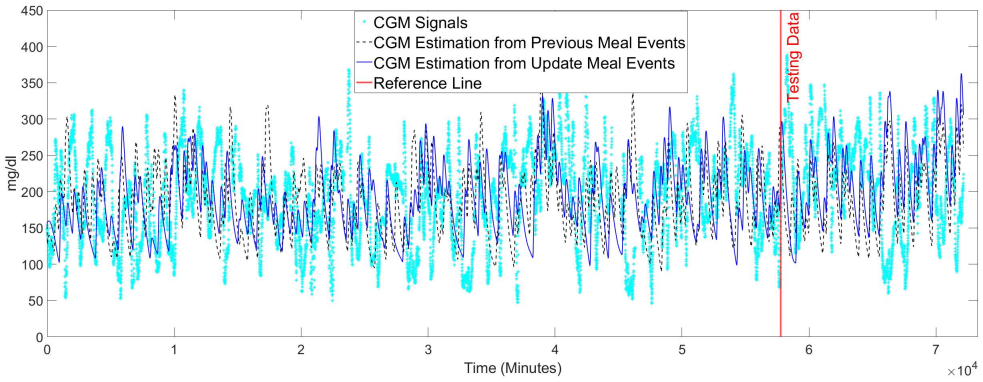


**Figure 4.19** Comparison plot of model performance with CGM signals with/without the meal correction algorithm applied physiological model with modified Lehmann meal function for 570. The left side of the reference line is training, and the right side is testing. The meal correction algorithm does not apply to testing CGMs.

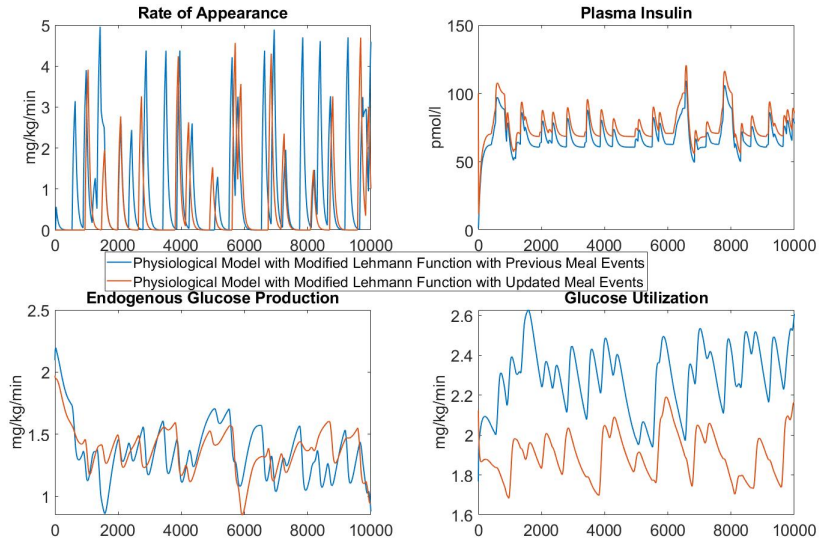


**Figure 4.20** The subsystem dynamics of both physiological models after the meal correction algorithm for 570 using modified Lehmann meal function. The first 10000 data points (around 7 days) dynamics are presented.

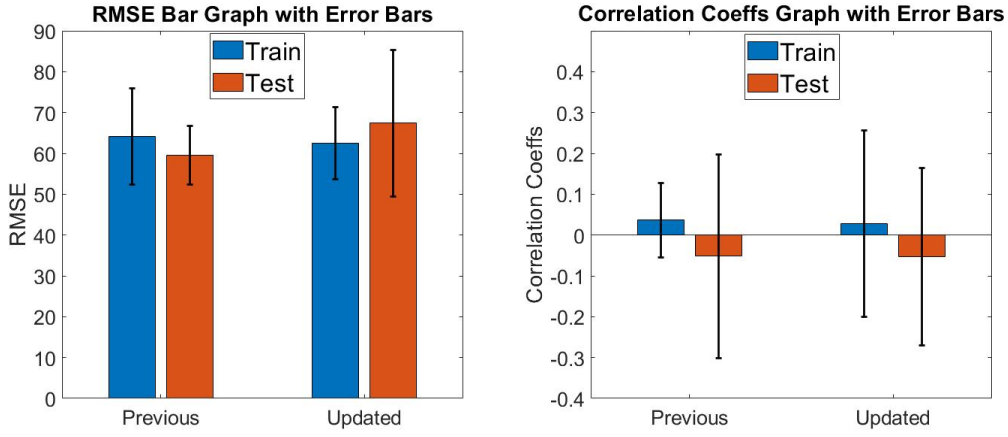


**Figure 4.21** Bar graph of model performance in training and testing among 6 data contributors using modified Lehmann meal function. First presented the evaluation results for the physiological model with the raw meal data, then presented the evaluation results with the updated meal events with meal correction algorithm.

ters are listed in Table 4.3.

**Table 4.3** Table of unknown parameters for hybrid model. The unit is not presented if a parameter is dimensionless. If a parameter is from the physiological model, then the relevant subsystem is provided in the last column, else the description of the parameter is presented.

| Parameter | Unit | Biological Range | Relevant Subsystem/Description |
|---|---|---|---|
| $G_{tb}$ | mg/kg | [100,300] | glucose subsystem |
| $b_{c1}$ | hour/min | [0,4] | insulin subsystem |
| $b_{c2}$ | $min^{-1}$ | [0,4] | insulin subsystem |
| $I_b$ | mg/dl | [0,10] | insulin subsystem |
| $I_{lb}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{pb}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{sc1b}$ | pmol/kg | [0,10] | insulin subsystem |
| $I_{sc2b}$ | pmol/kg | [0,10] | insulin subsystem |
| $K_m$ | mg/kg | [100,300] | insulin-dependent utilization subsystem |
| $T_{snack}$ | min | [10,20] | continuous snack size meal duration |
| $T_{meal}$ | min | [20,100] | continuous snack size meal duration |
| $w$ | | [-2,2] | the elements in the finial weight matrix |
| $t_{shift}$ | min | [0,50] | shift value for rate of appearance function |
| $l$ | | [0,200] | span value of moving average smoothing |

The number of parameter values may be different depending on the size of the reservoir dynamics, as shown in Table 4.1. In the sensitivity and identifiability analysis we introduced in Chapter 3, there is one parameter in the meal function as the most sensitive and identifiable parameter ($C_{bio}$) for both physiological models, and the meal function is considered as a sensitive subsystem in the physiological system. Therefore, we decide to consider all parameters in the reservoir dynamics as important parameters and estimate those in the inverse problem along with the selected parameters in the physiological systems, namely, $K_m$, $b_{c1}$, $b_{c2}$.

By evaluating the different combinations of the reservoir options, we plot the most satisfying results in Figure 4.22 and compare them with physiological models and data-driven models. The best performance hybrid model uses a small size reservoir ($n_{tps} = 3, n_{PC} = 10$) and all 3 meal sources from bolus meal event, self-reported meal event and the output from meal correction algorithm. After normalization, the shift and smooth operator (Shift=1, Smooth=1) are added in the reservoir network. The estimated parameter values are reported in Table 4.4, where the initial values in the physiological subsystems are fixed with the values presented in Table 4.2. The RMSEs for training and testing are 65.43 and 90.36, respectively. The training and testing correlation coefficients are 0.19 and 0.26, respectively. Compared to the best performance for the physiological model, the testing correlation coefficient is improved largely.

In Figure 4.22, we plot the prediction with the CGM signals for data contributor 570, notice that the training part of CGM estimation performed much better than testing. The meal correction algorithm was not applied to the testing data, thus it may lead to the worse behavior presented in

**Table 4.4** Major estimated parameter values in the hybrid model are reported in the table, the other estimated parameters are the weights in the reservoir dynamics and in the range of [-2,2].

| Model | $K_m$ | $b_{c2}$ | $b_{c1}$ | $T_{meal}$ | $T_{snack}$ | $l$ | $t_{shift}$ |
|---|---|---|---|---|---|---|---|
| Hybrid | 100.25 | 3.77 | 0.65 | 118.74 | 10.39 | 44 | 50 |

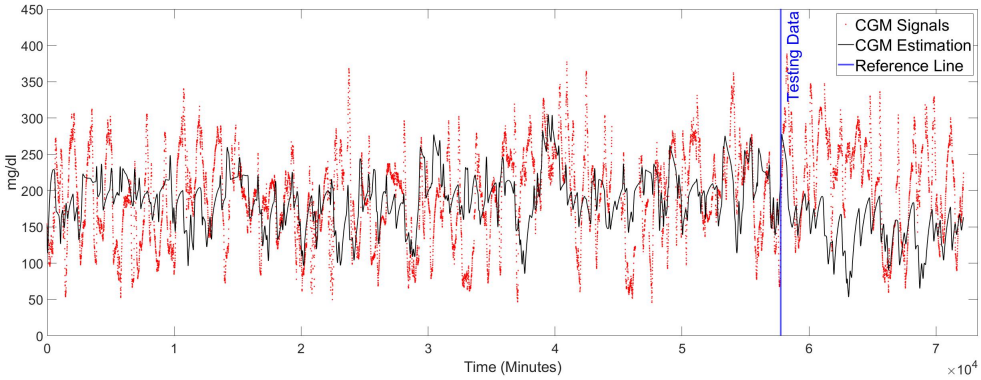


**Figure 4.22** Comparison plot of hybrid model prediction plot with CGM signals for 570. The left side of the reference line is training, and the right side is testing. The meal correction algorithm does not apply to testing CGMs.

the testing phase. In addition, the large fluctuation in the predicted CGMs presents various patterns throughout the time. By looking at the subsystem plot in Figure 4.23, the rate of appearance function generated by reservoir dynamics learned different patterns compared to the Hovorka or Lehmann model described in Chapter 3.

The downward patterns in the rate of appearance function may indicate the mistakes in meal events where the corresponding tangent line of CGM is negative. The average rate of appearance function is above 0 which may represent the meal consumption has a continuous effect for T1D individuals. The upward patterns are similar to the modified Lehmann function in magnitude.

The evaluation metrics among 6 data contributors are shown in the bar plot 4.24, where the variation in training performance is smaller compared to both physiological models, and improving in the testing correlation coefficients in general. The hybrid model for each contributor was able to generate large fluctuation patterns while lowering the errors. The hybrid model has the potential to improve further by carefully tuning the hyperparameters or testing on different sizes of reservoir dynamics.

## 4.4 Discussion

In this section, the hybrid model is presented by replacing the meal function with a data-driven model. The hybrid model using a multi-perceptron network has been successfully applied to the SIR disease model with a simulation dataset. However, for a complex physiological model and real-world data, it is more challenging.

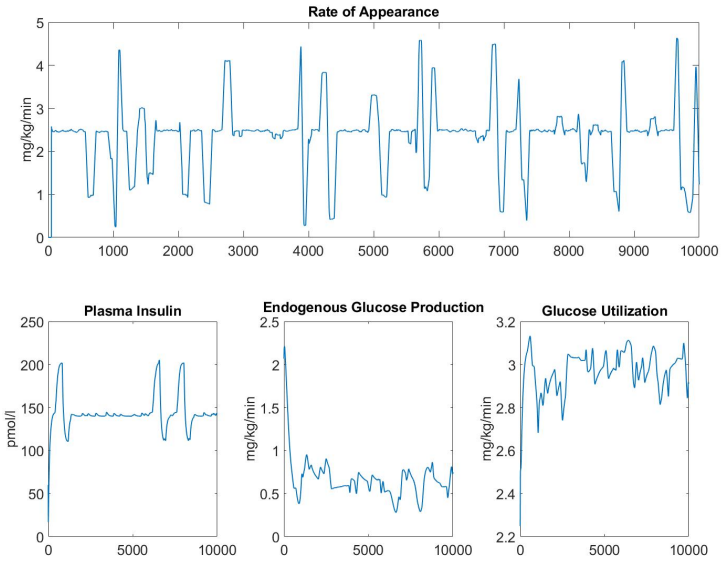


**Figure 4.23** The subsystem dynamics of the hybrid model. The first 10000 data points (around 7 days) dynamics are presented. The starting position of the rate of appearance function is due to a shift operator.

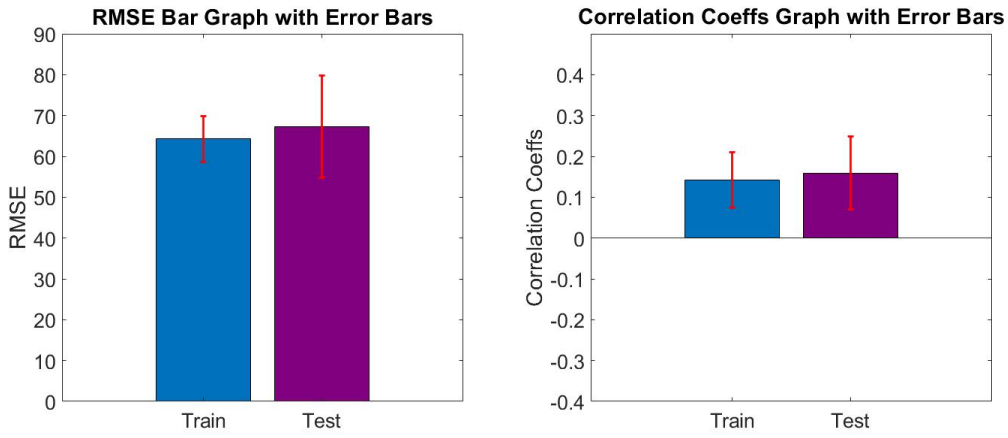


**Figure 4.24** Bar graph of model performance among 6 data contributors in training and testing for hybrid model, the mean values are reported with the red error bar represent the stand deviation.

We also proposed a meal correction algorithm, which identifies the incorrect meal event and unannounced meal events by observing the patterns in CGM signals. It can be applied to detect the meal event in the CGM for all diabetes, and future research is needed for determining the meal sizes. The updated meal events along with other meal events show promising results in the hybrid model, which improves the correlation coefficient and reduces the RMSE for the physiological model with modified Lehmann function.

In future iterations, more sizes of the reservoir computing should be considered, and the hyper-parameters can also be adjusted to obtain better results. We should also investigate the overfitting in training and testing as appears in all models presented in this work.

## 4.5 Contributions

We present a hybrid model that combines the data-driven model and physiological model for BG prediction tasks. The reservoir dynamics, which is the most promising model in data-driven models, was used to replace the underlying meal function as a subsystem in the complex physiological model. Furthermore, a meal correction algorithm is proposed for moving the meal events to the correct location and detecting the unannounced meals. The performance is largely improved using the hybrid approaches compared to the physiological models. The interpretability is obtained compared with data-driven models. The prediction results may be improved further by tuning the reservoir networks and replacing other variables in the physiological model.

CHAPTER

# 5

# CONCLUSION

In this work, we introduced the methodologies for blood glucose prediction in three types: data-driven model, physiological model, and hybrid approaches that combine the data-driven and physiological models.

The data-driven model had the advantage of accuracy, but choosing the suitable model and tuning the model hyperparameters could be challenging for real-world data, such as blood glucose time series for type 1 diabetes. In addition, the data-driven models are considered as black-box algorithms which output the prediction without biological interpretability. The reason data-driven models can provide a better prediction performance for time-series data may be due to the target variable (e.g., blood glucose levels) providing to the data-driven algorithm as an input source with time-shifted backward. Thus, the algorithm can learn the pattern of the signals and provides an accurate prediction. In the studies [Mar18; RR19], those models using only blood glucose levels are also able to provide good performance in predictions. If the target variable is removed from the input source, then most algorithms would have difficulty predicting the correct patterns.

Compared to the data-driven model, the physiological model does not use the blood glucose levels directly as input, but is introduced in the last phase when estimating the model parameters. The physiological model was built from literature where all model compartments and model parameters have biological meaning. The changes in each component in the physiological model will affect the blood glucose at the same time. By studying the physiological model in the blood glucose prediction, we can understand the dynamics changing in tissues or organs for type 1 diabetes, and provide better guidance for instructing the patient's daily activities. However, by the fixed number of parameters, it is difficult to find a suitable model for every patient, and the model parameters may change with respect to time. In addition, the blood glucose level collected from continuous glucose

monitoring contains noise and uncalibrated data which increases the challenge of obtaining a high fidelity physiological model.

The hybrid approaches utilized the most promising model in data-driven algorithms, reservoir computing. It replaced the most difficult simulation function as we found during the physiological experiments. It was found that the physiological model with hybrid components, model performance is improved while maintaining interpretability.

## 5.1 Contributions

Providing an efficient computer algorithm in the artificial pancreas would benefit diabetes patients for controlling the blood glucose level in the euglycemic range and provide warning ahead of time. In addition, we wish to provide the dynamics in the body organs and tissues along with the blood glucose monitor to guide the daily activities, i.e., suggesting duration and intensity of exercise, providing advice on the GI index and carbohydrate size of meal consumption, guiding the insulin injection, etc. This can be done by introducing an efficient hybrid model which contains accuracy and interpretability.

In the data-driven chapter, we found the regression models to be fast and accurate for various pattern dataset, where the reservoir computing can recover the missing dynamics when the blood glucose monitor is disconnected and the CGM is missing. In the physiological model, we construct the complex model with 6 subsystems and 12 variables. Each model component and parameter have biological meaning so we understand more about a specific patient by observing the dynamics in the subsystems. Finally, reservoir computing replaced the meal function in the physiological model, which is the most crucial but challenging function in the complex physiological model. By solving the inverse problem and obtaining the estimation of parameters in the hybrid model, the model performance is improved while maintaining interpretability.

The performance of the hybrid model can be improved as the training data size grows. More parameters can be introduced in the hybrid model to improve the accuracy, and the patient-specific parameters can reflect the daily routine of a type 1 diabetes person after training on a large dataset. After obtaining an accurate hybrid model, by providing the future activities (meal intake, exercise level, insulin bolus injection), it is possible to predict the future CGM dynamics. Then, the treatment such as basal insulin injection can be adjusted qualitatively based on the CGM control.

This work can also benefit type 2 diabetes as the physiological system of blood glucose transformation is similar to type 1 diabetes. With small modifications of the insulin subsystem based on the pathophysiology, the model can be extended to type 2 diabetes individuals.

## 5.2 Future Work

There are limitations of the hybrid approach we proposed in this work by learning from data-driven and physiological perspectives. For example, there may be other functions (i.e., other than meal

function) that the physiological model cannot simulate the dynamics well. Thus, identifying the inaccuracy subsystem in the physiological model is a potential research direction in the future. Currently, we only replaced the most crucial component with data-driven by our analysis and observation, but there could be more model variables that cannot reflect the true dynamics of type 1 diabetes patients. If we can replace those model components with the reservoir dynamics or other learnable architectures such as Seq-to-Seq LSTM, the model performance may be improved.

In addition, there is a limited selection of optimization tools for this hybrid approach, thus solving the inverse problem for a complex ODE with a large number of estimated parameters is difficult and time-consuming. The Hessian matrix of the complex ODE system is hard to compute as well as the gradient of the loss function, this limits the speed of convergence and finding an accurate local minimum. If the optimization tool is further improved for solving this complex inverse problem, then it could largely improve the performance of hybrid models.

## BIBLIOGRAPHY


[Acc18] Acciaroli, G. et al. "Reduction of Blood Glucose Measurements to Calibrate Subcutaneous Glucose Sensors: A Bayesian Multiday Framework". *IEEE Transactions on Biomedical Engineering* **65** (2018), pp. 587–596.

[Aie20] Aiello, E. M. et al. "Therapy-driven Deep Glucose Forecasting". *Engineering Applications of Artificial Intelligence* **87** (2020), p. 103255.

[AM10] Al-Mohy, A. H. & Higham, N. J. "The Complex Step Approximation to the Fréchet Derivative of a Matrix Function". *Numer. Algorithms* **53** (2010), pp. 133–148.

[Ant20] Antoni, F. D. et al. "Auto-Regressive Time Delayed jump neural network for blood glucose levels forecasting". *Knowledge-Based Systems* **203** (2020), p. 106134.

[Art17] Arthur, J. G. et al. "Feasibility of Parameter Estimation in Hepatitis C Viral Dynamics Models". *Journal of Inverse and Ill-Posed Problems* **25** (1 2017), pp. 69–80.

[Asc98] Ascher, U. M. & Petzold, L. R. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Philadelphia: Society for Industrial and Applied Mathematics, 1998.

[Atk14] Atkinson, M. et al. "Type 1 diabetes". *The Lancet* **383**.9911 (2014), pp. 69–82.

[Att17] Attarian, A. & Tran, H. "An Optimal Control Approach to Structured Treatment Interruptions for HIV Patients: A Personalized Medicine Perspective." *Applied Mathematics* **8** (2017), pp. 934–955.

[Ber89] Berger, M. & Rodbard, D. "Computer Simulation of Plasma Insulin and Glucose Dynamics After Subcutaneous Insulin Injection". *Diabetes Care* **12** (1989), pp. 725–736.

[Ber81] Bergman, R. et al. "Physiologic evaluation of factors controlling glucose tolerance in man: measurement of insulin sensitivity and beta-cell glucose sensitivity from the response to intravenous glucose". *J Clin Invest* **68**.6 (1981), pp. 1456–1467.

[Ber18] Bertachi, A. et al. "Prediction of Blood Glucose Levels And Nocturnal Hypoglycemia Using Physiological Models and Artificial Neural Networks". *CEUR Workshop Proceedings* **2148** (2018), pp. 85–90.

[Bev20] Bevan, R. & Coenen, F. "Experiments in non-personalized future blood glucose level prediction". *CEUR Workshop Proceedings* **2675** (2020), pp. 100–104.

[Bhi20] Bhimireddy, A. et al. "Blood Glucose Level Prediction as Time-Series Modeling using Sequence-to-Sequence Neural Networks". *CEUR Workshop Proceedings* **2675** (2020), pp. 125–130.

[Bia18] Bianchi, F. M. et al. "Reservoir computing approaches for representation and classification of multivariate time series". *arXiv preprint arXiv:1803.07870* (2018).

[Bia18] Bianchi, F. M. et al. “Reservoir computing approaches for representation and classification of multivariate time series”. *arXiv e-prints*, arXiv:1803.07870 (2018), arXiv:1803.07870. arXiv: 1803.07870 [cs.NE].

[Bren ] Breton, M. “Physical Activity—The Major Unaccounted Impediment to Closed Loop Control”. *Journal of Diabetes Science and Technology* **2** (Jan. 2008).

[But30] Butterworth, S. “On the Theory of Filter Amplifiers”. *Experimental Wireless and the Wireless Engineer* **7** (1930).

[Cap20] Cappon, G. et al. “A Personalized and Interpretable Deep Learning Based Approach to Predict Blood Glucose Concentration in Type 1 Diabetes”. *CEUR Workshop Proceedings* **2675** (2020), pp. 75–79.

[Che19] Chen, H. et al. “Committed Moving Horizon Estimation for Meal Detection and Estimation in Type 1 Diabetes”. *2019 American Control Conference (ACC)*. 2019, pp. 4765–4772.

[Che18] Chen, J. et al. “Dilated Recurrent Neural Network for Short-time Prediction of Glucose Concentration”. *CEUR Workshop Proceedings* **2148** (2018), pp. 69–73.

[Cla87] Clarke, W. L. et al. “Evaluating Clinical Accuracy of Systems for Self-Monitoring of Blood Glucose”. *Diabetes Care* **10**.5 (1987), pp. 622–628.

[Con18] Contreras, I. et al. “Using Grammatical Evolution to Generate Short-Term Blood Glucose Prediction Models”. *CEUR Workshop Proceedings* **2148** (2018), pp. 91–96.

[Dan20] Daniels, J. et al. “Personalised Glucose Prediction via Deep Multitask Networks”. *CEUR Workshop Proceedings* **2675** (2020), pp. 110–114.

[Das08] Dassau, E. et al. “Detection of a meal using continuous glucose monitoring: implications for an artificial beta-cell”. *Diabetes Care* **31**.2 (2008), pp. 295–300.

[Dav59] Davidon, W. C. “VARIABLE METRIC METHOD FOR MINIMIZATION” (1959).

[Den83] Dennis, J. E. & Schnabel, R. B. *NumericalMethods for Unconstrained Optimization and Nonlinear Equations*. SIAM, 1983.

[Des18] Desai, P. M. et al. “Pictures Worth a Thousand Words: Reflections on Visualizing Personal Blood Glucose Forecasts for Individuals with Type 2 Diabetes”. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* **CHI 18** (2018), pp. 1–13.

[Du20] Du, J. et al. “Brain MRI Super-Resolution Using 3D Dilated Convolutional Encoder–Decoder Network”. *IEEE Access* **PP** (2020), pp. 1–1.

[Ell12] Ellis, G. “Chapter 9 - Filters in Control Systems”. *Control System Design Guide (Fourth Edition)*. Ed. by Ellis, G. Fourth Edition. Boston: Butterworth-Heinemann, 2012, pp. 165 –183.

[Far19] Faruqui, S. et al. “Development of a Deep Learning Model for Dynamic Forecasting of Blood Glucose Level for Type 2 Diabetes Mellitus: Secondary Analysis of a Randomized Controlled Trial”. *JMIR Mhealth Uhealth* **7**.11 (2019), e14452.

[Fer87] Ferrannini, E. & Cobelli, C. “The kinetics of insulin in man. I. General aspects.” *Diabetes Metab. Rev* **3** (1987), pp. 335–363.

[Fle87] Fletcher, R. *Practical Methods of Optimization*. New York: John Wiley & Sons, 1987.

[Fle63] Fletcher, R. & Powell, M. J. D. “A Rapidly Convergent Descent Method for Minimization”. *The Computer Journal* **6**.2 (1963), pp. 163–168. eprint: `https://academic.oup.com/comjnl/article-pdf/6/2/163/1041527/6-2-163.pdf`.

[Fox18] Fox, I. et al. “Deep multi-output forecasting: Learning to accurately predict blood glucose trajectories”. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2018).

[Fre20] Freiburghaus, J. et al. “A Deep Learning Approach for Blood Glucose Prediction of Type 1 Diabetes”. *CEUR Workshop Proceedings* **2675** (2020), pp. 131–135.

[Geo12] Georga, E. et al. “A predictive model of subcutaneous glucose concentration in type 1 diabetes based on Random Forests”. *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2012, pp. 2889–2892.

[Geo18] Georga, E. et al. *Personalized Predictive Modeling in Type 1 Diabetes*. Ioannina, Greece: Academic Press, 2018, pp. 15 –35.

[Gri89] Griewank, A. “On Automatic Differentiation”. *IN MATHEMATICAL PROGRAMMING: RECENT DEVELOPMENTS AND APPLICATIONS*. Kluwer Academic Publishers, 1989, pp. 83–108.

[Ham20] Hameed, H. & Kleinberg, S. “Investigating potentials and pitfalls of knowledge distillation across datasets for blood glucose forecasting”. *CEUR Workshop Proceedings* **2675** (2020), pp. 85–89.

[Har14] Harvey, R. et al. “Design of the glucose rate increase detector a meal detection module for the health monitoring system”. *J Diabetes Sci Technol.* **8**.2 (2014), pp. 307–320.

[Hoc97] Hochreiter, S & Schmidhuber, J. “Long short-term memory”. *Neural Computation* **9**.8 (1997), pp. 1735–1780.

[Hovly] Hovorka, R. et al. “Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes”. *Physiological Measurement* **25**.4 (July 2004).

[Jen81] Jenkins, D. J. et al. “Glycemic index of foods: a physiological basis for carbohydrate exchange”. *The American Journal of Clinical Nutrition* **34**.3 (1981), pp. 362–366. eprint: `https://academic.oup.com/ajcn/article-pdf/34/3/362/24156452/362.pdf`.

[Joe20] Joedicke, D. et al. “Analysis of the performance of Genetic Programming on the Blood Glucose Level Prediction Challenge 2020”. *CEUR Workshop Proceedings* **2675** (2020), pp. 141–145.

[Kel99] Kelley, C. *Iterative Methods for Optimization*. Raleigh, NC, USA: Society for Industrial and Applied Mathematics, 1999.

[Kha20] Khadem, H. et al. “Multi-lag Stacking for Blood Glucose Level Prediction”. *CEUR Workshop Proceedings* **2675** (2020), pp. 146–150.

[Lag20] Lagergren, J. H. et al. “Learning partial differential equations for biological transport models from noisy spatio-temporal data”. *Proc. R. Soc. A.* **467** (2234 2020).

[Lag00] Lagergren, J. H. et al. “Biologically-informed neural networks guide mechanistic modeling from sparse experimental data”. *Plos Computational Biology* **16** (12 2000).

[Leh92] Lehmann, E. D. & Deutsch, T. “A physiological model of glucose insulin interaction in type 1 diabetes”. *Journal of Biomedical Engineering* **14**.3 (1992).

[Ma20] Ma, N. et al. “Online Blood Glucose Prediction Using Autoregressive Moving Average Model with Residual Compensation Network”. *CEUR Workshop Proceedings* **2675** (2020), pp. 151–155.

[Mah17] Mahmoudi, Z. et al. “Fault and meal detection by redundant continuous glucose monitors and the unscented Kalman filter”. *Biomedical Signal Processing and Control* **38** (2017), pp. 86 –99.

[Mah18] Mahmoudi, Z. et al. “Meal Detection for Type 1 Diabetes Using Moving Horizon Estimation”. *2018 IEEE Conference on Control Technology and Applications (CCTA)*. 2018, pp. 1674–1679.

[Man06] Man, C. D. et al. “A model of glucose production during a meal”. *Conf Proc IEEE Eng Med Biol Soc.* **2006** (2006), pp. 5647–5650.

[Mann ] Man, C. et al. “Physical Activity into the Meal Glucose-Insulin Model of Type 1 Diabetes: In Silico Studies”. *Journal of Diabetes Science and Technology* **3**.1 (Jan. 2009).

[Mant ] Man, C. et al. “Meal Simulation Model of the Glucose-Insulin System”. *IEEE Transactions on Biomedical Engineering* **54**.10 (Oct. 2007).

[Marp ] Marling, C. & Bunescu, R. “The OhioT1DM Dataset for Blood Glucose Level Prediction: Update 2020”. *CEUR Workshop Proc.* **2675** (Sep 2020), pp. 71–74.

[Mar00] Martins, J. R. R. A. et al. “An automated method for sensitivity analysis using complex variables”. *American Institute of Aeronautics and Astronautics* (2000).

[Mar18] Martinsson, J. et al. “Automatic Blood Glucose Prediction with Confidence Using Recurrent Neural Networks”. *CEUR Workshop Proceedings* **2148** (2018), pp. 64–68.

[May20] Mayo, M. & Koutny, T. “Neural Multi-class Classification Approach to Blood Glucose Level Forecasting with Prediction Uncertainty Visualisation”. *CEUR Workshop Proceedings* **2675** (2020), pp. 80–84.

[Mid18] Midroni, C. et al. “Predicting Glycemia in Type 1 Diabetes Patients: Experiments with XG-Boost”. *CEUR Workshop Proceedings* **2148** (2018), pp. 79–84.

[Mob20] Mobasseri, M. et al. “Prevalence and incidence of type 1 diabetes in the world: a systematic review and meta-analysis”. *Health promotion perspectives* **10**.2 (2020), pp. 98–115.

[Mor91] Morris, M. “Factorial sampling plans for preliminary computational experiments”. *Technometrics* **33** (1991), pp. 161–174.

[Mos89] Mosekilde, E. et al. “Modeling absorption kinetics of subcutaneous injected soluble insulin.” *Journal of Pharmacokinetics and Biopharmaceutics* **17** (1989), pp. 67–87.

[MO20] Munoz-Organero, M. “Deep Physiological Model for Blood Glucose Prediction in T1DM Patients”. *Sensors* **20**.14 (2020), p. 3896.

[Nar20] Nardini, J. et al. “Learning Equations from Biological Data with Limited Time Samples”. *Bull Math Biol* **82** (119 2020).

[Nem20] Nemat, H. et al. “Data Fusion of Activity and CGM for Predicting Blood Glucose Levels”. *CEUR Workshop Proceedings* **2675** (2020), pp. 120–124.

[O’C08] O’Connell Michele, A. et al. “Optimizing Postprandial Glycemia in Pediatric Patients With Type 1 Diabetes Using Insulin Pump Therapy: Impact of glycemic index and prandial bolus type”. English. *Diabetes care* **31**.8 (2008). Copyright - Copyright American Diabetes Association Aug 2008; Document feature - Illustrations; Tables; Graphs; ; Last updated - 2019-09-06; CODEN - DICAD2, pp. 1491–5.

[Oor16] Oord, A. et al. “WaveNet: A Generative Model for Raw Audio”. *arXiv e-prints* (2016). arXiv: 1609.03499 [cs.SD].

[Ost08] Ostman, J. et al. “Gender differences and temporal variation in the incidence of type 1 diabetes: results of 8012 cases in the nationwide Diabetes Incidence Study in Sweden 1983-2002.” *J Intern Med.* **263**.4 (2008), pp. 386–94.

[Pal13] Palumbo, P. et al. “Mathematical modeling of the glucose–insulin system: A review”. *Mathematical Biosciences* **244**.2 (2013), pp. 69–81.

[Pav20] Pavan, J. et al. “Personalized Machine Learning Algorithm based on Shallow Network and Error Imputation Module for an Improved Blood Glucose Prediction”. *CEUR Workshop Proceedings* **2675** (2020), pp. 95–99.

[Rac20] Rackauckas, C. et al. “Universal Differential Equations for Scientific Machine Learning”. *CoRR* **abs/2001.04385** (2020). arXiv: 2001.04385.

[Ray21] Raymond, S. J. & Camarillo, D. B. *Applying physics-based loss functions to neural networks for improved generalizability in mechanics problems*. 2021. arXiv: 2105.00075 [physics.comp-ph].

[Rei07] Reifman, J. et al. "Predictive Monitoring for Improved Management of Glucose Levels". *Journal of Diabetes Science and Technology* **1**.4 (2007). PMID: 19885110, pp. 478–486.

[RR19] Rodriguez-Rodriguez, I. et al. "Utility of Big Data in Predicting Short-Term Blood Glucose Levels in Type 1 Diabetes Mellitus Through Machine Learning Techniques". *Sensors (Basel)* **19**.20 (2019).

[RF20] Rubin-Falcone, H. et al. "Deep Residual Time-Series Forecasting: Application to Blood Glucose Prediction". *CEUR Workshop Proceedings* **2675** (2020), pp. 105–109.

[Sae19] Saeedi, P. et al. "Global and regional diabetes prevalence estimates for 2019 and projections for 2030 and 2045: Results from the International Diabetes Federation Diabetes Atlas, 9th edition." *Diabetes research and clinical practice* **157**.107843 (2019).

[Sak17] Sakulrang, S. et al. "A fractional differential equation model for continuous glucose monitoring data". *Advances in Difference Equations* **150** (2017), pp. 1–12.

[Sam17] Samadi, S. et al. "Meal Detection and Carbohydrate Estimation Using Continuous Glucose Sensor Data". *IEEE Journal of Biomedical and Health Informatics* **21**.3 (2017), pp. 619–627.

[Sam18] Samadi, S. et al. "Automatic Detection and Estimation of Unannounced Meals for Multivariable Artificial Pancreas System". *Diabetes Technology Therapeutics* **20**.3 (2018), pp. 235–246.

[Sch18] Schiavon, M. et al. "Modeling Subcutaneous Absorption of Fast-Acting Insulin in Type 1 Diabetes". *IEEE Transactions on Biomedical Engineering* **65**.9 (2018), pp. 2079–2086.

[Sha70] Shanno, D. "Conditioning of Quasi-Newton Methods for Function Minimization". *Mathematics of Computing* **24** (1970), pp. 647–656.

[Spa07] Sparacino, G. et al. "Glucose Concentration can be Predicted Ahead in Time From Continuous Glucose Monitoring Sensor Time-Series". *IEEE Transactions on Biomedical Engineering* **54**.5 (2007), pp. 931–937.

[Sun20] Sun, X. et al. "Prediction of Blood Glucose Levels for People with Type 1 Diabetes using Latent-Variable-based Model". *CEUR Workshop Proceedings* **2675** (2020), pp. 115–119.

[Sut14] Sutskever, I. et al. *Sequence to Sequence Learning with Neural Networks*. 2014. arXiv: 1409.3215 [cs.CL].

[Tai16] Taieb, S. & Atiya, A. "A Bias and Variance Analysis for Multistep-Ahead Time Series Forecasting". *IEEE Transactions on Neural Networks and Learning Systems* **27** (2016), pp. 62–76.

[Tarc ] Tarin, C. et al. “Comprehensive pharmacokinetic model of insulin Glargine and other insulin formulations”. *IEEE Trans Biomed Eng 2005* **52**.12 (Dec. 2005).

[Tur16] Turksoy, K. et al. “Meal Detection in Patients With Type 1 Diabetes: A New Module for the Multivariable Adaptive Artificial Pancreas Control System”. *IEEE Journal of Biomedical and Health Informatics* **20**.1 (2016), pp. 47–54.

[Vic97] Vicini, P. et al. “The hot IVGTT two-compartment minimal model: indexes of glucose effectiveness and insulin sensitivity.” *American Journal of Physiology* **273(5)** (1997 Nov), E1024–E1032.

[Wol19] Woldaregay, A. Z. et al. “Data-Driven Blood Glucose Pattern Classification and Anomalies Detection: Machine-Learning Applications in Type 1 Diabetes”. *J Med Internet Res* **21**.5 (2019), e11030.

[Xie17] Xie, J. & Wang, Q. “A Variable State Dimension Approach to Meal Detection and Meal Size Estimation: In Silico Evaluation Through Basal-Bolus Insulin Therapy for Type 1 Diabetes”. *IEEE transaction on biomedical engineering* **64**.6 (2017), pp. 1249–1260.

[Xie18] Xie, J. & Wang, Q. “Benchmark Machine Learning Approaches with Classical Time Series Approaches on the Blood Glucose Level Prediction Challenge”. *CEUR Workshop Proceedings* **2148** (2018), pp. 97–102.

[Yan20] Yang, T. et al. “Multi-Scale Long Short-Term Memory Network with Multi-Lag Structure for Blood Glucose Prediction”. *CEUR Workshop Proceedings* **2675** (2020), pp. 136–140.

[Yu16] Yu, F. & Koltun, V. *Multi-Scale Context Aggregation by Dilated Convolutions*. 2016. arXiv: `1511.07122 [cs.CV]`.

[Zan20] Zanna, L. & Bolton, T. “Data-Driven Equation Discovery of Ocean Mesoscale Closures”. *Geophysical Research Letters* **47** (17 2020).

[Zec14] Zecchin, C. et al. “Jump neural network for online short-time prediction of blood glucose from continuous monitoring sensors and meal information”. *Computer Methods and Programs in Biomedicine* **113**.1 (2014), pp. 144–152.

[Zhe20] Zheng, F. et al. “Unannounced Meal Detection for Artificial Pancreas Systems Using Extended Isolation Forest”. *Annu Int Conf IEEE Eng Med Biol Soc.* **2020** (2020), pp. 5892–5895.

[Zhe19] Zheng, M. et al. “Automated meal detection from continuous glucose monitor data through simulation and explanation”. *Journal of the American Medical Informatics Association* **26**.12 (2019), pp. 1592–1599.

[Zhu18] Zhu, T. et al. “A Deep Learning Algorithm For Personalized Blood Glucose Prediction”. *CEUR Workshop Proceedings* **2148** (2018), pp. 74–78.

[Zhu20] Zhu, T. et al. “Blood Glucose Prediction for Type 1 Diabetes Using Generative Adversarial Networks”. *CEUR Workshop Proceedings* **2675** (2020), pp. 90–94.

[Zur19] Zurbau, A. et al. “Acute effect of equicaloric meals varying in glycemic index and glycemic load on arterial stiffness and glycemia in healthy adults: a randomized crossover trial”. English. *European journal of clinical nutrition* **73**.1 (2019), pp. 79–85.

# APPENDIX

APPENDIX

# A

# MEAL CORRECTION ALGORITHM PSEUDOCODE

**Algorithm 2** Relocation Algorithm

```
1: Output: Meal_index, Meal_size, Dis_back, Dis_front
2: Input: df.CGM (dataframe of CGM), meal_p (2 columns matrix contain meal time t and meal
   size m)
3: for n in [0, length(meal_p)) do
4:   if n!=0 and n!= length(meal_p)-1 then
5:     diff_back=difference(df.CGM[Meal_index[n-1]:meal_p[n]])
6:     diff_front=difference(df.CGM[meal_p[n]:meal_p[n+1]])
7:     if diff_back is non-positive and diff_front is non-negative then
8:       Meal_index.append(meal_p[t(n)])
9:       Meal_size.append(meal_p[m(n)])
10:      Dis_back.append(0)
11:      Dis_front.append(0)
12:      if sign of every element in diff_back is the same then
13:        index_back=meal_index[n-1]
14:        Dis_back.append(length(diff_back))
15:      else
16:        index_back=meal_p[t(n)]-nearest location of the valley (see Algorithm 3)
17:        Dis_back.append(nearest location)
18:      end if
19:      if sign of every element in diff_front is the same then
20:        index_front=Meal_index[n-1]
21:        Dis_front.append(length(diff_front))
22:      else
23:        index_front=meal_p[t(n)]+nearest location of the valley (see Algorithm 3)
24:        Dis_front.append(nearest location)
25:      end if
26:      compare index_front and index_back which closer to meal_p[t(n)],
         and store it in Meal_index, and Meal_size.append(meal_p[m(n)])
27:    else
28:      if n=0 then
29:        diff_back=difference(df.CGM[0:meal_p[0]])
30:        repeat the Operation 6-26
31:      else
32:        diff_front=difference(df.CGM[meal_p[n]:end])
33:        repeat the Operation 5, 7-26
34:      end if
35:    end if
36:  end if
37: end for
```

**Algorithm 3** Find Valley Algorithm

```
Output: loc
Input: array1, front_inc
array2 = find where array1 !=0
while n in array2 do
    if n!=0 then
        if array2[n-1]<array2[n] and array2[n-1]×array2[n]==1 then
            loc_list.append(n)
        end if
    end if
    if length(loc_list)=0 then
        loc_list.append(0)
    end if
    if front_inc=True then
        loc=loc_list[0]
    else
        loc=loc_list[-1]
    end if
end while
```

**Algorithm 4** Detect Unannouncement Meal Algorithm

```
Output: Meal_time
Input: df (contain df.CGM and df.time), meal_c (Meal_index from Algorithm 2)
df.d=extract the day from df.time
for d in [0,end] do
    meal_count=count of meal_c in day d
    df_sub=df in day d from 5:00 a.m. to 10 p.m.
    loc_list=the valley position of the CGM in the day d (see Algorithm 3)
    while t in loc_list do
        if t in meal_c then
            Meal_time.append(meal_c[t])
        else
            if meal_count<=4 then
                if df.CGM[t]<=300 then
                    diff_CGM=df_sub.CGM(t+5)-df_sub.CGM(t)
                    delta_CGM=(df_sub.CGM(t+30)-df_sub.CGM(t))/30
                    if diff_CGM>=0 and delta_CGM>0 then
                        Meal_time.append(meal_c[t])
                        meal_count++
                    end if
                end if
            end if
        end if
    end while
end for
```