Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Eira Mørch-Thoresen

Review of the Effect of Meals on the Glucose-Insulin Regulatory System

Master's thesis in Cybernetics and Robotics
Supervisor: Anders Lyngvi Fougner, Hasti Khoshamadi
May 2021

NTNU
Kunnskap for ei betre verd

Eira Mørch-Thoresen

Review of the Effect of Meals on the Glucose-Insulin Regulatory System

Master's thesis in Cybernetics and Robotics
Supervisor: Anders Lyngvi Fougner, Hasti Khoshamadi
May 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Faculty of Information Technology
and Electrical Engineering
Department of Engineering
Cybernetics

PROBLEM DESCRIPTION FOR MASTER THESIS

Candidate's name: Eira
Course: TTK4900 Teknisk Kybernetikk, Masteroppgave
Title (Norwegian): Litteraturstudium av hvordan måltidene påvirker glukosereguleringen i kroppen.
Title (English): **Review of the effect of meals on the glucose-insulin regulatory system.**
Description:

This project is affiliated with Artificial Pancreas Trondheim (APT) at NTNU. In APT, we aim at developing a system for robust closed-loop glucose control in diabetes mellitus type 1. In such a system, a mathematical model is exploited in the controller (model predictive control), as well as in the simulator the controller is tested on. This master thesis relates to this mathematical model.

For people with type 1 diabetes mellitus, the pancreatic β-cells do not secrete endogenous insulin, which is essential for glycemic control. Treatment with exogenous insulin is needed to maintain their blood glucose concentration within narrow bounds in order to avoid hypo- and hyperglycemia (low and high glucose level, respectively).

The artificial pancreas is a system combining a glucose sensor, a control algorithm, and an insulin infusion device which can help optimize glycemic control in diabetes. The typical disturbances considered for controller design are meals, which increase the glucose level and physical activity, which has both acute and delayed effect on glucose metabolism including the increase of insulin sensitivity, insulin-dependent and -independent glucose uptake and the endogenous glucose production (i.e. glucose released from the energy storage in the liver).

The student has previously worked on the "Sørensen model" and extending this to include physical activity. In this master thesis, the aim is to review models that describe the effect of meals on the glucose-insulin regulatory system and add one or several of these to the existing model. Specifically, the student will take on the following tasks;

1. Review the literature on the digestive system and how meals affect glucose level, as well as mathematical models describing this.
2. Select one or several model structures.
3. Find suitable data sets or collect necessary data to validate and evaluate the model(s).

Startup date: 4 January 2021
Submission deadline: 30 May 2021

Supervisor(s): Associate Professor Anders Lyngvi Fougner
PhD candidate Hasti Khoshamadi

Trondheim, 8 December 2020

Anders Lyngvi Fougner

Abstract

Diabetes mellitus is a disease that is characterized by the lack of controlling blood glucose levels due to no insulin production or reduced insulin sensitivity. Treatment of diabetes therefore include injections of exogenous insulin. The artificial pancreas is a device that automates the delivery of exogenous insulin and thus enables automatic blood glucose regulation. One of the main disturbances to the artificial pancreas are meals which increases the blood glucose. It is therefore important to have knowledge about how different types of meals affect the glucose levels. An important tool here are mathematical glucose-insulin meal models that can be used in simulations to compute what the postprandial glucose response will be for different types of meals. This thesis aims to investigate these meal models. That includes a literature review where 11 different meal models are compared. Three of these meal models were then coupled with a whole-body glucose model (Sorensen model) and the model parameters were then estimated using a dataset containing glucose and meal data from six patients. The results showed that two of the models were able to satisfactorily fit the experimental data in identification, whereas one model did not. The identification results were then tested on a test dataset. The test results were varying and indicated that more data should be used in identification to reduce overfitting. Still, some of this variation between training and testing was within what could be expected due to normal differences between meals.

Sammendrag

Diabetes mellitus er en sykdom som kjennetegnes ved mangelen på å kontrollere blodsukkernivået grunnet ingen produksjon av insulin eller redusert insulinsensitivitet. Behandling av diabetes inkluderer derfor injeksjoner av eksogent insulin. En kunstig bukspyttkjertel er en innretning som automatiserer injeksjonene av eksogent insulin og muliggjør derfor automatisk blodsukkerregulering. En av hovedforstyrrelsene til den kunstige bukspyttkjertelen er måltider, noe som øker blodsukkeret. Det er derfor viktig å ha kunnskap om hvordan ulike måltider påvirker blodsukkeret. Et viktig hjelpemiddel her er matematiske glukose-insulin-måltidsmodeller som kan brukes i simuleringer til å predikere hva postprandiale glukoseverdier vil være. Denne masteroppgaven har som mål å undersøke slike måltidsmodeller. Det inkluderer et litteratursøk der 11 ulike måltidsmodeller ble sammenlignet. Tre av disse modellene ble så koblet sammen med en fullkroppsglukosemodell (Sorensen-modellen) og modellparameterne ble så estimert ved hjelp av et datasett som inneholder glukosemålinger og måltidsdata fra seks pasienter. Resultatene viste at to av de tre modellene var i stand til å etterligne den eksperimentelle dataen i tilfredsstillende grad, mens én modell ikke var det. De estimerte parameterne ble så testet på et testdatasett. Resultatet av testingen var varierende og indikerte at mer data skulle ha blitt brukt i parameteridentifikasjonen for å redusere overtilpasning. Likevel er variasjon mellom ulike måltid forventet og kan forklare mye av forskjellen i resultat mellom trening og test.

Preface

This thesis is written as a master thesis at the Cybernetics and Robotics study program at Norwegian University of Science and Technology (NTNU) and in partnership with the research group Artificial Pancreas Trondheim (APT). The work was performed during the spring of 2021, between January and the end of May. This work is intended to contribute to the research field of diabetes and artificial pancreas, and more specifically how meals affect blood glucose and how this can be mathematically modeled. My supervisors were Anders Lyngvi Fougner and Hasti Khoshamadi.

Tools that were utilized in this thesis were MATLAB in which all simulations and data preparation was done, and Excel to store data files. A medical textbook[1] was provided by my supervisors such that I could understand the physiology behind diabetes and the gastrointestinal system. For finding relevant literature, Google Scholar was used together with the NTNU provided access to scientific publications online. A publicly available experimental dataset[2] was also used in parameter identification.

I would like to thank my supervisors for their guidance throughout the semester. Despite of not being able to meet in person because of covid-19, we still carried out supervision meetings every week online.

List of Figures

List of Tables

Abbreviations

AP	artificial pancreas
APT	Artificial Pancreas Trondheim
ATP	adenosine triphosphate
BG	blood glucose
CGM	continuous glucose measurement
CHO	carbohydrates
FFA	free fatty acid
GI	gastrointestinal
MSE	mean squared error
ODE	ordinary differential equation
OGTT	oral glucose tolerance test
T1D	Type 1 Diabetes
T2D	Type 2 Diabetes

Contents

1 Introduction

This section will give a brief description of some background theory to motivate the aim of this thesis.

1.1 Diabetes mellitus

Diabetes mellitus is a disease that affects blood glucose (BG) regulation. Diabetes can be divided into two groups, Type 1 Diabetes (T1D) and Type 2 Diabetes (T2D). T1D is caused by auto-immune destruction of the beta cells in the pancreas, causing a lack of insulin production. This lack of insulin production causes hyperglycemia (high BG). Thus, a T1D patient has to be treated with exogenous insulin. T2D is a lifestyle related disease. It is not caused by destruction of the beta cells, but rather reduced sensitivity of the insulin receptors. This causes the cells to be unable to utilize the glucose in the blood, yielding hyperglycemia. Treatment of T2D includes taking on a healthy lifestyle through healthy eating and exercise. In T1D, the individual has to measure their BG levels and inject exogenous insulin with a pump several times a day. A device that does this automatically is called an artificial pancreas, and would greatly improve the life of a type 1 diabetic patient. Parts of this section is cited from the term project by Mørch-Thoresen, 2020[3].

1.2 Artificial pancreas

An artificial pancreas (AP) is a system combining a glucose sensor, a control algorithm, and an insulin pump which can help optimize glycemic control in T1D (fig. 1.1). The AP computes the amount of insulin needed based on BG measurements, and then injects the insulin through a pump. A disturbance to the artificial pancreas is meals. After a meal, the carbohydrates in the food causes the BG level to increase and insulin needs to be injected. An AP that takes ingested food into account is important to stay within the euglycemic range. However, different meals will have different effects on the glucose levels. Meals that are rich in fat may slow down absorption of glucose, whereas glucose from foods such as sugar, rice and bread is absorbed faster[1]. A diabetic patient needs to count the carbohydrates in the meal in order to inject the correct amount of insulin. This can be challenging, and a system such as the AP that does this automatically would be of great help.

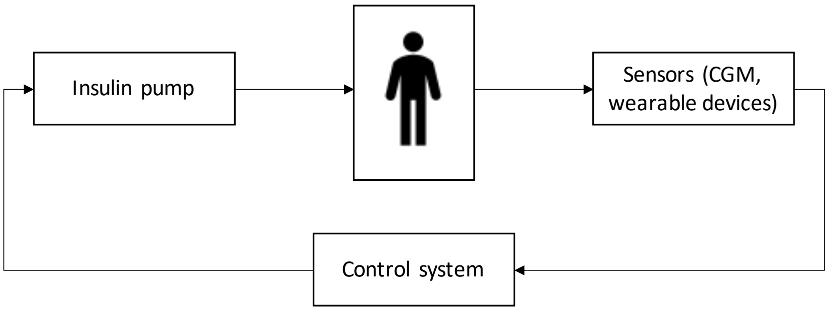


Figure 1.1: Artificial pancreas block diagram

1.3 Aim of study

The aim of this master thesis is to

- Investigate how a meal affects the blood glucose levels through reading relevant literature about physiology
- Perform a literature review of mathematical models describing a meal into the glucose-insulin dynamics
- Select one or several meal models and incorporate them into the Sorensen model
- Perform a parameter identification on the meal model(s) using a suitable data set

1.4 Outline

This thesis is organized as follows: section 2 describes relevant background theory about physiology, section 3 discusses how the glucose-insulin dynamics and meals can be mathematically described, section 4 goes through theory on how parameters can be identified and section 5 presents the meal models that were found in the literature review. Identification of meal model parameters is discussed in section 6 and section 7, section 8 contains discussion and a conclusion is presented in section 9. Lastly, the appendix contains additional relevant information such as equations and code.

1.5 Artificial Pancreas Trondheim (APT)

This thesis is written in partnership with Artificial Pancreas Trondheim (APT). APT is a research group established in 2013 at The Norwegian University of Science and Technology (NTNU) in Trondheim and consists of researchers in the fields of control engineering, biomedical engineering, biosensors, applied clinical research, endocrinology, anesthesia and intensive care medicine, pharmacology, biotechnology, mathematical modeling, biochemistry and chemometrics[4]. APT has a long-term goal of developing a robust artificial pancreas for patients with T1D and T2D. This section is cited from the term project report by Mørch-Thoresen, 2020[3].

2 Physiology

This section will present relevant theory about physiology for this thesis. That includes diabetes, the gastrointestinal system and how meals affect glucose levels.

2.1 Glucose-insulin regulatory system and diabetes

During fasting, the BG levels in a healthy person are tightly controlled to be between 80 and 90 mg/100 ml. After a meal containing carbohydrates, the BG may rise up to 120-140 mg/100 ml, but is rapidly returned to the euglycemic range due to the release of insulin[5]. This control of blood glucose is facilitated through the release of the two hormones insulin and glucagon.

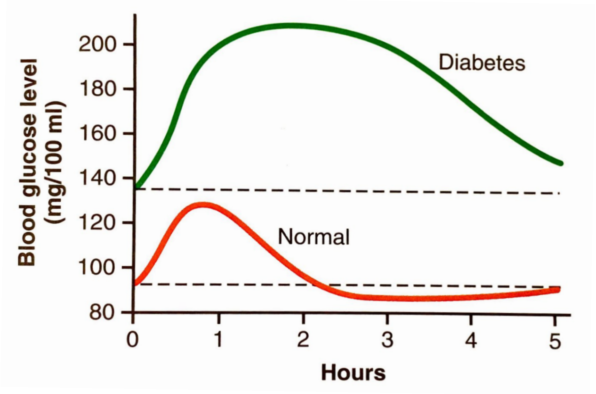


Figure 2.1: Effects on BG after a meal for people with and without T1D[5]

Insulin is a hormone that decreases BG levels in three ways:

1. By turning glucose into ATP through a process which is called glycolysis. ATP is energy that is to be used immediately by the cells anywhere in the body.
2. By turning glucose into glycogen. This process is called glycogenesis. This is short-term storage of energy mainly in the liver and muscles.
3. By turning glucose into fatty acids (lipids). This process is called lipogenesis. The fatty acids are stored in adipose tissue and is a long-term storage of energy.

When BG levels are low, the hormone glucagon is released. Glucagon increases BG through

1. Breaking down liver and muscle glycogen into glucose. This is called glycogenolysis.
2. Turning amino acids into glucose through a process called gluconeogenesis.

In addition, glucagon also turns fatty acids in adipose tissue into ketone bodies through ketogenesis. Parts of this section is cited from the term project report by Mørch-Thoresen, 2020[3].

2.2 Secretion of insulin and glucagon

Insulin and glucagon are produced in the endocrine pancreas. The endocrine pancreas consist of a type of tissue called islets of Langerhans. An islet consists of four types of cells; alpha, beta, delta and F cells[6]. The beta cells secrete insulin and the alpha cells secrete glucagon. In an individual with T1D, the pancreas does not secrete insulin due to auto-immune destruction of the beta cells. This means that the individual has no ability to lower BG levels. This causes a constant state of hyperglycemia if not treated with exogenous insulin. Parts of this section is cited from the term project report by Mørch-Thoresen, 2020[3].

2.3 Hyper- and hypoglycemia

Hyperglycemia is the state where the BG levels are above the normal range. Untreated high blood glucose concentration over time can lead to heart attack, stroke or blindness due to abnormal function of the blood vessels in multiple tissues[5]. Hypoglycemia, on the other hand, occurs when the BG levels are too low. This is dangerous because glucose is the only nutrient used by the brain, and a lack of glucose can cause dizziness, seizures and coma. Hypoglycemia occurs when too much insulin is injected, and can also happen due to prolonged vigorous exercise. Correct injections of insulin is therefore necessary to stay healthy for a person with T1D. Parts of this section is cited from the term project report by Mørch-Thoresen, 2020[3].

2.4 Gastrointestinal system

The gastrointestinal (GI) system consists of the gastrointestinal tract and its associated glandular structures. This includes the oral cavity, esophagus, stomach, small intestine, large intestine, rectum, and the glandular structures such as salivary glands, liver, exocrine pancreas and intestinal glands[1] (ref fig. 2.2). The major function of the GI system is to provide nutrition through ingestion, with the main principles being secretion, digestion, absorption and motility. Secretions from exocrine glands facilitate digestion and promote absorption of nutrients. Digestion refers to the process where foodstuff are broken down into smaller particles, whereas absorption refers to the process where the products of digestion are transported from the lumen of the GI tract into the blood. Motility refers to the movement of the GI tract (such as gastric emptying) due to the presence of smooth muscles.

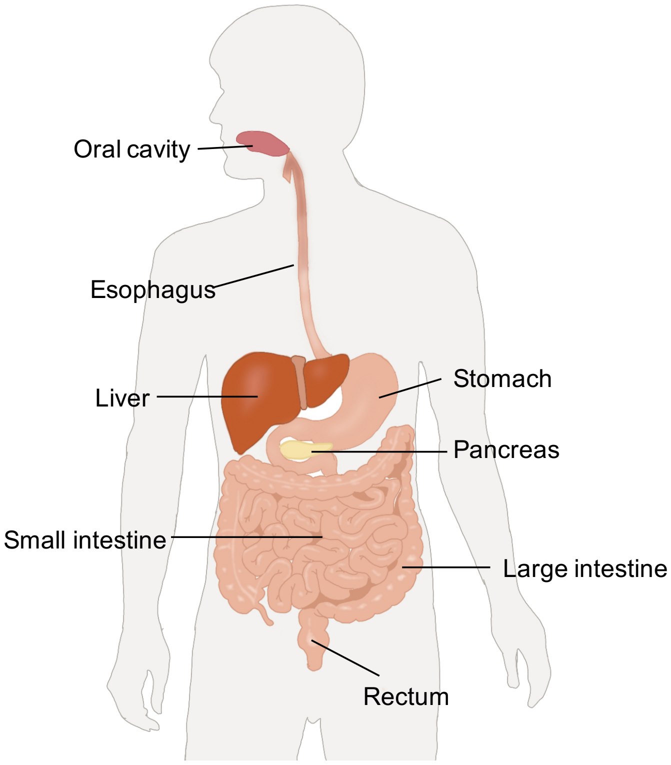


Figure 2.2: Gastrointestinal system

Food enters through the oral cavity where it is chewed and mixed with saliva. The esophagus transports the food into the stomach through peristaltic contractions. In the stomach, the food is transformed into a thick semi-fluid mass called chyme. This is done through churning and mixing the food with gastric juices as shown in fig. 2.3. The chyme is then released into the small intestine at a controlled rate.

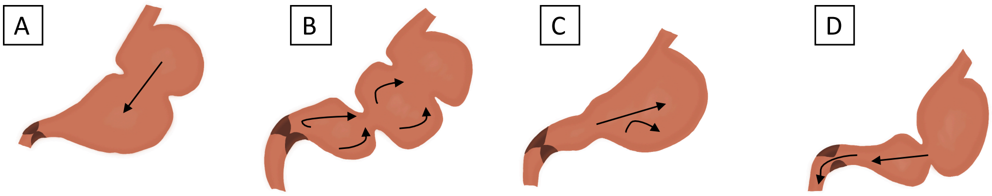


Figure 2.3: Gastric motility: the pyloric sphincter is closed from A to C while the food is mixed and grinded. A - peristaltic contraction, B - antral contraction, C - retropulsion. In D, the sphincter opens to let chyme into the duodenum. Adapted from G. K. Pal[1]

The small intestine is where the major part of digestion takes place, and can be further divided into three parts; duodenum, jejunum and ileum. The inside of the small intestine has finger-like projections that are called villi. They increase the inner surface area up to 300 m^2 to increase absorption[1]. Once all the nutrients are digested and absorbed in the small intestine, the remaining contents goes to the large intestine where water and electrolytes are absorbed. Finally, the waste products enter the rectum where it is stored until disposed.

2.5 Gastric emptying

The term gastric emptying refers to the emptying of the gastric contents into the duodenum (small intestine). Gastric emptying occurs when the pyloric sphincter opens to let chyme leave the stomach and enter the duodenum. The gastric emptying is precisely controlled due to inhibitory and excitatory vagal motor neurons[7]. There are several factors that affect the rate of gastric emptying. Meals rich in fat slow down gastric emptying. This is because fat forms an oily layer on top of the other gastric contents. This, in addition to the weak contractions of the stomach, slows down emptying. Another factor that affects emptying is the consistency of the food. Liquid food is emptied faster than solid food[1]. Gastric emptying can also work differently in T1D than in healthy people. A study from 1995[8] showed that individuals with T1D had delayed solid and liquid gastric emptying compared to healthy subjects. The rate of gastric emptying also varies with the blood glucose levels. Gastric emptying is slower during hyperglycemia when compared with euglycemia and accelerated during hypoglycemia[9].

2.6 Digestion and absorption

The process of breaking down food into smaller pieces is called digestion. This happens partially in the stomach and partially in the small intestine. In the stomach, the food is mixed with gastric juices to make chyme. When the chyme enters the small intestine, pancreatic and biliary enzymes are released. They help to break down the particles into the final form for absorption. The main nutrients from food that are absorbed in the small instestine are carbohydrates, proteins and fat.

- Carbohydrates are broken down into monosaccharides
- Proteins are broken down into amino acids and peptides
- Fat is broken down into fatty acids and monoglycerides

The nutrients are then absorbed through the wall of the small intestine and into the blood. Carbohydrates end up as blood glucose (energy), amino acids from proteins are used in formation and maintenance of tissues, and fatty acids from fat end up as energy or stored in adipose tissue[10].

2.7 Carbohydrates and blood glucose levels

Carbohydrates is the only nutrient that affect BG levels directly and are molecules made up of carbon, hydrogen and oxygen[11]. The most simple type of carbohydrates are the monosaccharides. They are building blocks for the more complex di- and polysaccharides. The different types of carbohydrates and their food sources are shown in table 2.1.

Carbohydrate type	Source	Functions
Monosaccharides		
Glucose	Sugar, rice, bread, vegetables	Final form for tissue utilization
Fructose	Fruits, honey	Converted to glucose by liver
Galactose	Milk lactose	Converted to glucose by liver
Disaccharides		
Sucrose	Sugar cane, pineapple	Converted to glucose by liver
Lactose	Milk	Converted to glucose by liver
Maltose	Germinating seeds	Converted to glucose by liver
Polysaccharides		
Starch	Plants, rice, potato, wheat, corn	Converted to glucose by liver
Soluble fibers	Fruits, grains, legume	Incr. abs. time of food
Insoluble fibers	Vegetables, wheat bran	Incr. passage of intestinal content

Table 2.1: Carbohydrates and their food sources[10]

Of the absorbed monosaccharides, glucose is in the final form and ready to be used directly, whereas fructose and galactose are transported to the liver where they are converted into glucose. BG levels will then increase. Since the carbohydrates are broken down into monosaccharides before absorption, the more complex carbohydrates are taken up more slowly and gives a more controlled increase in BG than simpler molecules. Something that describes this is the glycemic index. The glycemic index is a relative ranking of how quickly a food increases BG levels. It ranks foods from 0 to 100, where high glycemic index means fast absorption and low index means slower absorption. Foods with high glycemic index are for example sugar/glucose (100), white wheat bread (75) and boiled potato (78). Low index foods include kidney beans (24), skim milk (37) and soya beans (16)[12].

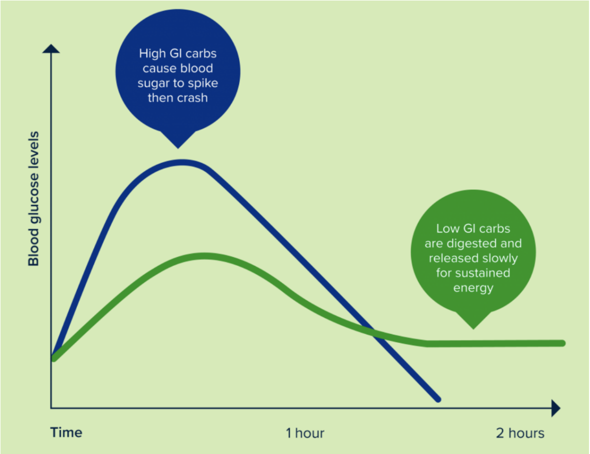


Figure 2.4: Glycemic index. Figure borrowed from Glycemic Index Foundation[13]

Nutritionally, it makes sense to divide carbohydrates into digestible and non-digestible carbohydrates. The non-digestible carbohydrates come from dietary fibers that cannot be broken down by human digestive enzymes. However, they are still important as they keep the gut healthy, slow down absorption of glucose, flatten postprandial BG response and are useful in weight management[11] [14]. The digestible carbohydrates are broken down into monosaccharides and then absorbed in the small intestine as discussed previously.

2.8 Diabetes and meals

Since ingested carbohydrates increase BG levels, it is important that this is accounted for through insulin injections in T1D to stay within the euglycemic range for as much time as possible. This is done by estimating the amount of carbohydrates in a meal and self-administer an insulin bolus[15]. It is also important to remember when dosing a bolus, that it is not only the amount of carbohydrate that affects the BG, but also the type of carbohydrate (glycemic index, fibers). However, it is important to note that postprandial BG responses are highly individual and that the glycemic index is not always accurate. Zeevi et al[16] showed that people responded very differently to the same standardized meals. One cause of this is the gut microbiota composition. Microbiota are important for utilization of some of the non-digestible carbohydrates that human enzymes cannot break down. The study showed that when providing info of the microbiota composition along with information about the ingested food to a machine learning algorithm, the algorithm was able to predict individual BG responses. A third thing to consider is that meals rich in fat and protein slow down absorption, and that this can cause prolonged hyperglycemia[15]. All in all, the diabetic patient has to estimate the carbohydrate content of each meal and then inject an insulin bolus. However, individual factors affects the BG response, and the individual must therefore take this into account.

2.9 Summary

This section has presented relevant theory about physiology for this thesis. That includes info about the glucose-insulin system, gastrointestinal system, digestion and absorption, carbohydrates and postprandial blood glucose.

3 Mathematical modeling

This section will discuss how the glucose-insulin system can be described mathematically, and then present some of the most known models. A discussion about how meal and gastrointestinal dynamics can be incorporated into these models follows afterwards.

3.1 Compartmental analysis

When developing a mathematical model of the glucose-insulin system, it is very common to use compartmental analysis. That means to assign tissues or organs to different compartments that represent the quantity of some material inside them. A differential equation is then developed by writing reaction kinetics and mass balances between inflows and outflows to the compartment. Figure 3.1 shows two compartments.

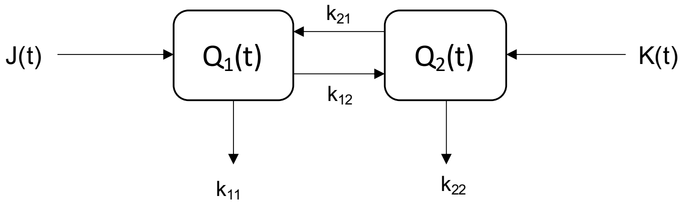


Figure 3.1: Mathematical modeling with compartments. Adapted from Chee and Fernando[17]

Q_1 and Q_2 represents the quantity of materials in compartments 1 and 2 respectively, whereas the different k_{ij}s are flow rates to and from compartments. $J(t)$ and $K(t)$ are exogenous flows of material. This results in the following mass balance equations:

$$\frac{dQ_1(t)}{dt} = -k_{11}Q_1(t) - k_{12}Q_1(t) + k_{21}Q_2(t) + J(t) \tag{3.1}$$

$$\frac{dQ_2(t)}{dt} = k_{12}Q_1(t) - k_{21}Q_2(t) - k_{22}Q_2(t) + K(t) \tag{3.2}$$

In a glucose-insulin model, the same approach is used, but the compartments are now representing the amount of insulin and glucose in different tissues in the body. One example of such a compartment can be "plasma glucose". The amount of glucose in the plasma, i.e. blood, is then given by the difference between what flows in and what flows out. Inflows can be hepatic glucose production or glucose from food. Outflows can be utilization of glucose by the muscles.

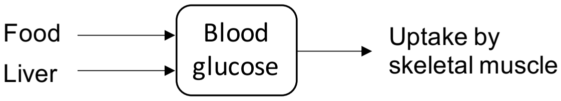


Figure 3.2: Example of a compartment

Parts of this section is cited from the term project report by Mørch-Thoresen, 2020[3].

3.2 Bergman model

The Bergman model[18], also called the minimal model because of its simplicity, was developed in the 1980's. It consists of three ODEs that represent plasma glucose $G(t)$, plasma insulin $I(t)$ and insulin action $X(t)$. The equations for a person in a diabetic state are given by

$$\frac{dG(t)}{dt} = -p_1(G(t) - G_b) - X(t)G(t) + p(t) \tag{3.3}$$

$$\frac{dX(t)}{dt} = p_3(I(t) - I_b) - p_2X(t) \tag{3.4}$$

$$\frac{dI(t)}{dt} = -n(I(t) - I_b) + u(t) \tag{3.5}$$

The terms $p(t)$ and $u(t)$ represents exogenous infusions of glucose and insulin respectively. I_b and G_b are basal levels of insulin and glucose. Refer to fig. 3.3 below for a visualization of how the three states interact with each other.

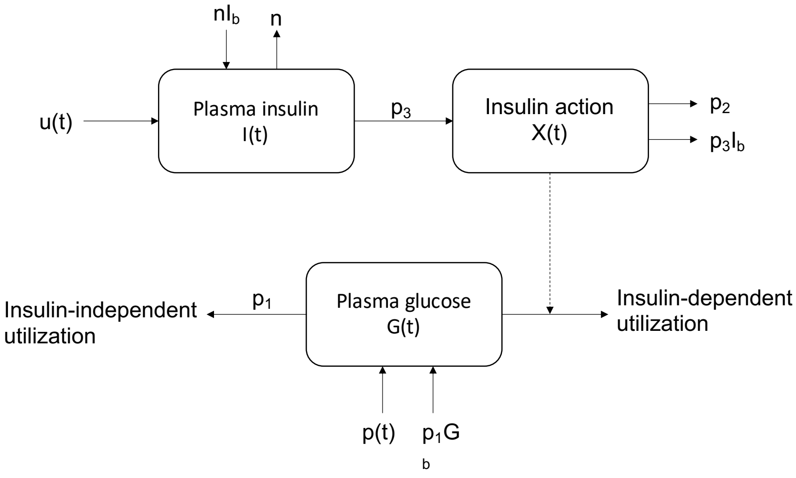


Figure 3.3: Scheme of Bergman model.

The amount of plasma insulin is affected by exogenous insulin infusion, and is again affecting insulin action. The insulin action state describes how insulin is needed to decrease BG, i.e. insulin-dependent utilization of glucose. Glucose can also decrease through insulin-independent utilization, such as usage by the brain. Glucose levels increases through exogenous infusion. The simplicity of this model is both its strength and its limitation. It is easy to comprehend while it also is somewhat accurate in modeling glucose and insulin levels. However, it only uses lumped compartments, and a physiologically accurate distinction between organs or tissues is not modeled.

3.3 Sorensen model

The Sorensen model is an extensive mathematical model that describes glucose and insulin in the human body by 19 ODEs. It was developed in 1985 as a part of a PhD thesis. It divides he body into six compartments, namely brain, heart/lungs, liver, gut, kidney and periphery as shown in fig. 3.4.

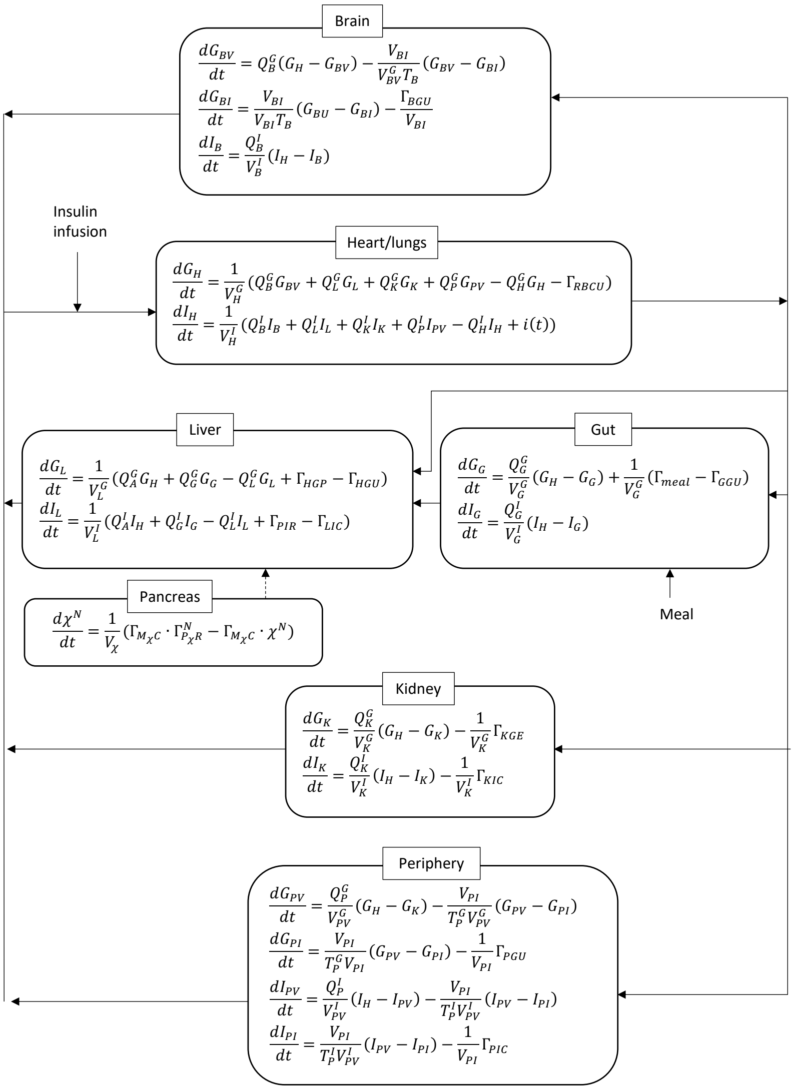

Brain
$\frac{dG_{BV}}{dt} = Q_B^G(G_H - G_{BV}) - \frac{V_{BI}}{V_{BV}^G T_B}(G_{BV} - G_{BI})$
$\frac{dG_{BI}}{dt} = \frac{V_{BI}}{V_{BI} T_B}(G_{BU} - G_{BI}) - \frac{\Gamma_{BGU}}{V_{BI}}$
$\frac{dI_B}{dt} = \frac{Q_B^I}{V_B^I}(I_H - I_B)$
Insulin infusion
Heart/lungs
$\frac{dG_H}{dt} = \frac{1}{V_H^G}(Q_B^G G_{BV} + Q_L^G G_L + Q_K^G G_K + Q_P^G G_{PV} - Q_H^G G_H - \Gamma_{RBCU})$
$\frac{dI_H}{dt} = \frac{1}{V_H^I}(Q_B^I I_B + Q_L^I I_L + Q_K^I I_K + Q_P^I I_{PV} - Q_H^I I_H + i(t))$
Liver
$\frac{dG_L}{dt} = \frac{1}{V_L^G}(Q_A^G G_H + Q_G^G G_G - Q_L^G G_L + \Gamma_{HGP} - \Gamma_{HGU})$
$\frac{dI_L}{dt} = \frac{1}{V_L^I}(Q_A^I I_H + Q_G^I I_G - Q_L^I I_L + \Gamma_{PIR} - \Gamma_{LIC})$
Gut
$\frac{dG_G}{dt} = \frac{Q_G^G}{V_G^G}(G_H - G_G) + \frac{1}{V_G^G}(\Gamma_{meal} - \Gamma_{GGU})$
$\frac{dI_G}{dt} = \frac{Q_G^I}{V_G^I}(I_H - I_G)$
Pancreas
$\frac{d\chi^N}{dt} = \frac{1}{V_\chi}(\Gamma_{M_\chi C} \cdot \Gamma_{P_\chi R}^N - \Gamma_{M_\chi C} \cdot \chi^N)$
Meal
Kidney
$\frac{dG_K}{dt} = \frac{Q_K^G}{V_K^G}(G_H - G_K) - \frac{1}{V_K^G}\Gamma_{KGE}$
$\frac{dI_K}{dt} = \frac{Q_K^I}{V_K^I}(I_H - I_K) - \frac{1}{V_K^I}\Gamma_{KIC}$
Periphery
$\frac{dG_{PV}}{dt} = \frac{Q_P^G}{V_{PV}^G}(G_H - G_K) - \frac{V_{PI}}{T_P^G V_{PV}^G}(G_{PV} - G_{PI})$
$\frac{dG_{PI}}{dt} = \frac{V_{PI}}{T_P^G V_{PI}}(G_{PV} - G_{PI}) - \frac{1}{V_{PI}}\Gamma_{PGU}$
$\frac{dI_{PV}}{dt} = \frac{Q_P^I}{V_{PV}^I}(I_H - I_{PV}) - \frac{V_{PI}}{T_P^I V_{PV}^I}(I_{PV} - I_{PI})$
$\frac{dI_{PI}}{dt} = \frac{V_{PI}}{T_P^I V_{PV}^I}(I_{PV} - I_{PI}) - \frac{1}{V_{PI}}\Gamma_{PIC}$

Figure 3.4: Scheme of Sorensen model. Solid arrows represent blood flow between compartments. The pancreas (glucagon) model is only assumed to have effect on endogenous glucose production Γ_{HGP}.

The types of states in the model are glucose concentration G, insulin concentration I and glucagon concentration χ, where the subscripts describes which compartment it is modeling (B - brain, H - heart/lungs, L - liver, G - gut, K - kidney and P - periphery). If a second subscript is included, that indicates if it is interstitial fluid space (I) or vascular blood water space (V). The parameters V and Q represent volume and vascular blood water flow rate. Another type of important parameter/variable are the metabolic sources or sinks. They are indicated by the symbol Γ with different subscripts. For example, Γ_{HGP} means hepatic glucose production. The mass balance equations are written on the form

$$V_J^X \frac{dC_J(t)}{dt} = Q_J^X(C_{Ji}(t) - C_J(t)) - \Gamma_{JU} \tag{3.6}$$

where subscript J represents the compartment and superscript X the type of solute (insulin, glucose)[19]. C can either be insulin, glucose or glucagon. All equations and parameters of the Sorensen model are listed in appendix A. A strength of the Sorensen model is that it is physiologically accurate as it is modeling glucose and insulin in different tissues, and not only in lumped compartments. It is thus able to capture more of what is happening in the body.

3.4 Hovorka model

The model by Hovorka[20] was developed in 2004 and consists of a glucose subsystem (two states), insulin subsystem (one state) and insulin action subsystem (three states). The equations model are given by

$$\frac{dQ_1(t)}{dt} = -(\frac{F_{01}^C}{V_G G(t)} + x_1(t))Q_1(t) + k_{12}Q_2(t) - F_R + U_G(t) + EGP_0(1 - x_3(t)) \tag{3.7}$$

$$\frac{dQ_2(t)}{dt} = x_1(t)Q_1(t) - (k_{12} + x_2(t))Q_2(t)y(t)G(t) = \frac{Q_1}{V_G} \tag{3.8}$$

$$\frac{dI(t)}{dt} = \frac{U_I(t)}{V_I} - k_e I(t) \tag{3.9}$$

$$\frac{dx_1(t)}{dt} = -k_{a1}x_1(t) + k_{b1}I(t) \tag{3.10}$$

$$\frac{dx_2(t)}{dt} = -k_{a2}x_2(t) + k_{b2}I(t) \tag{3.11}$$

$$\frac{dx_3(t)}{dt} = -k_{a3}x_3(t) + k_{b3}I(t) \tag{3.12}$$

where Q_1 and Q_2 represent glucose in accessible and non-accessible compartments. I is the amount of insulin and x_1, x_2, x_3 are insulin actions on transfer from Q_1 to Q_2, utilization of glucose and EGP (endogenous glucose production). The equations are also shown in fig. 3.5.

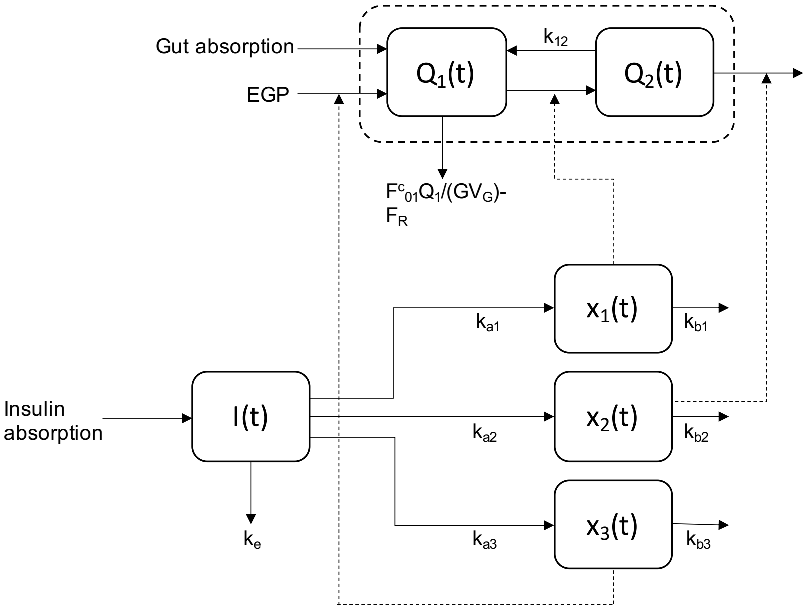


Figure 3.5: Scheme of Hovorka model

3.5 UVA/Padova model

The UVA/Padova model[21] is an FDA approved diabetes simulator, meaning that it can be used as a substitute for preclinical trials for certain insulin treatments, including closed-loop algorithms for AP. The first version was released in 2008, and it was updated in both 2013 and 2017. It consists of compartments describing glucose, liver, GI tract, muscle and adipose tissue, insulin, insulin delivery, glucagon, alpha cells and glucagon delivery. Figure 3.6 shows how these compartments interact with each other. This model consists of many equations that will not be listed here. The reader is referred to check the appendix of Dalla Man et al, 2017[21] where all equations are listed.

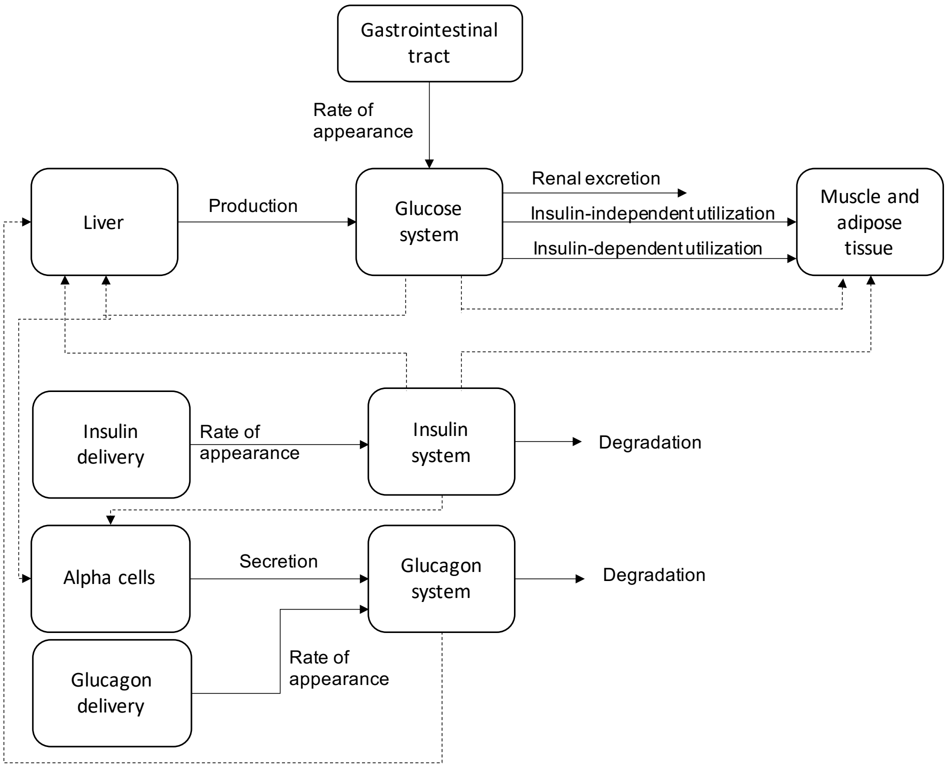


Figure 3.6: Scheme of UVA/Padova 2017

3.6 Modeling meal dynamics

The previous subsections have presented some of the most known glucose models. In order for them to represent glucose dynamics accurately, the effects of a meal should be included. That is mainly to model how BG increases after eating carbohydrates. However, there are many other important factors that affect BG as well. Based on the discussion about the GI system in section 2, there are a number of factors that should or could be included in a meal model. They are

Amount of ingested carbohydrates or glucose
Digestible carbohydrates increase BG levels. A large amount of carbohydrates will increase BG more than a small amount. It is therefore important to include the amount of CHO in the meal model.

Type of carbohydrate
As discussed previously, digestible carbohydrates from different foods will have different effects on BG. Sugar is absorbed much faster than for instance starch. A way of including this in a model could be by using the glycemic index.

Mixed meals and other macronutrients
A typical meal does not only consist of carbohydrates. Nutrients such as fat, protein and dietary fibers are important parts of a meal. These nutrients typically slow down absorption of glucose into the blood. Including these dynamics would be beneficial in order to be even

more accurate.

Stomach and gastric emptying
After a meal, food is first stored in the stomach where it is grinded and turned into chyme before it is sent to the small intestine. Including a stomach compartment in a meal model would increase the physiological accuracy. It is also important to be able to fit the model better to the individual since people with T1D are known to have malfunctioning gastric emptying[9]. Thus, including some adjustable parameters here could help fitting the model to the individual.

Small intestine and absorption rate
The small intestine is the last stop before nutrients are absorbed and is therefore an important part of the process of digestion. Absorption of glucose depends on the amount of food in the small intestine, so adding a compartment describing the gut would be helpful.

Physiological accuracy
When developing a model, there are two approaches one can take. One is to create a model that is motivated by knowledge about the GI system. The other one is to make a more empirically based model, meaning that it is based more on empirical data rather than theoretical knowledge. Both approaches can give good models, but the ones based on theoretical knowledge can sometimes provide better understanding.

Order of ingestion
Dietary fibers slow down absorption. That means that if fibers are ingested before fast sugars, one can have improved BG response. This is something that can be valuable to include in a meal model.

Personalized factors
Another thing that can be included in a meal model is that people react differently to same foods. One cause of this is the microbiota composition. However, this is a very detailed factor to include, and is probably not included in many models.

3.7 Summary

This section has presented theory about how mathematical model describing the glucose-insulin system are developed. The most known glucose-insulin models were then presented. Lastly, important factors to include in a meal model were discussed.

4 Parameter identification and evaluation

This section will go through theory about parameter identification and evaluation of dynamical systems.

4.1 Cost function optimization

The goal of parameter identification is to find the parameter values of a model so that the output of the model is as close to measured empirical data as possible. This is done by minimizing a cost function (also known as error function). Given a system described by the model

$$y_{sim} = f(x;\, p) \tag{4.1}$$

where x is the input and p are parameters, and empirical measured data described as y_{meas}, the cost function C can be defined as an error or distance measure between the two

$$C = d(y_{sim}, y_{meas}) \tag{4.2}$$

which in this thesis will be defined as the mean squared error (MSE). The optimization problem can then be formulated as

$$\min_{p} d(y_{sim}, y_{meas}) \quad s.t. \quad \text{some constraints} \tag{4.3}$$

where the constraints can for instance be lower and upper bounds, linear and nonlinear equality constraints. Parameter values p that minimize the cost function are found by a optimization algorithm. Refer to fig. 4.1 for a graphical description of the process. In the case of this thesis, the model with parameters to estimate is a glucose-insulin meal model (described in section 5) and the empirical data comes from real life measurements of blood glucose, insulin infusion and meal data (dataset described in section 6). Inputs to the model are meals and insulin injections (basal and bolus), while the output is glucose level.

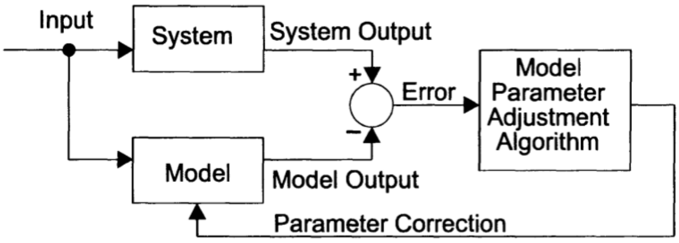
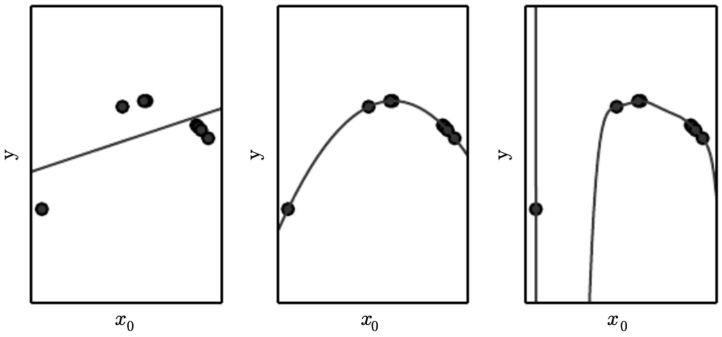


Figure 4.1: Parameter identification. Figure borrowed from Moeller[22]

Regarding the parameter adjustment algorithm, there are many different algorithms depending on the complexity of the problem. A glucose-insulin model will typically fall under the category of nonlinear programming and will therefore need an algorithm that is suited for solving such a task. This was later in this thesis implemented in MATLAB through the function `fmincon`.

4.2 Training, validation, test and overfitting

The empirical measured data described in the section above is called training data. The purpose of the training data is to fit the model. However, it is desired that the model is able to predict the glucose response for any type of meal, and the identified parameters are therefore tested on a new dataset (test set) to see if the model is able to produce good results for data that has not been used in fitting. If the model performs well on both training and test data, it is said to be able to generalize. An obstacle to generalizability is overfitting. Overfitting occurs when the model is able to fit to the training data, but not the test data (the gap between training error and test error is too large)[23], and typically happens when the training dataset is small or when the model has too many parameters. A solution to this is to increase the size of the training dataset. On the other hand, there is underfitting which occurs when the training error is too large. The goal is therefore to find an optimal solution where the model has both low training error and a small gap between training and test error. Figure 4.2 shows the differences in performance for a model that is underfitting or overfitting vs. an optimal result.



(a) Underfitting **(b)** Optimal fit **(c)** Overfitting

Figure 4.2: The model that is underfitting is not able to follow the curve of the data, whereas the model that is overfitting is following the data “too well”. In the middle, the optimal model is capturing the curve of the data points. Figure borrowed from Goodfellow[23].

Another thing that might improve the results is to take use of a validation dataset to tune hyperparameters. Hyperparameters are settings to the optimization algorithm such as step size, number of iterations or other factors that affects the training result. The validation dataset is therefore constructed to estimate the generalization error after training and update hyperparameters accordingly[23]. In this thesis, only a training and test dataset will be used, whereas a validation set is not utilized.

4.3 Summary

This section has gone through theory about how minimizing a cost function can be used to obtain a set of optimal parameters. This technique will later be used in this thesis to fit meal models to a dataset containing glucose measurements.

5 Literature review of meal models

This section will present the meal models that were found in the literature review that was performed.

5.1 Method

The literature review was performed in January and February 2021. The goal of the review was to find mathematical models describing how meals affect blood glucose levels. Google Scholar was the search engine that was used, and words that were searched for included “meal” or “food” in combination with for instance glucose model, glucose simulator, diabetes, postprandial, mathematical model, Bergman, Sorensen, UVA/Padova etc. The models that were considered relevant were organized into a table with descriptions about title, author(s), year, short description and which main model they were used in. After the review was finished, the models were analyzed and compared. Figures showing compartments and their relations were made such that the models could be compared more easily. Findings that have been included in this review, i.e. considered relevant, are meal models that are described by algebraic or differential equations. Machine learning algorithms have not been included. The results from the literature review are presented in the next section.

5.2 Results

Lehmann and Deutsch, 1992[24]
This meal model consists of one compartment describing glucose in the gut. Ingested carbohydrates (glucose) enters the gut through a gastric emptying function G_{empt} that depends on the amount of ingested carbohydrates. Glucose leaves the gut compartment as absorption into the blood. The model equations are given by

$$\dot{G}_{gut}(t) = G_{empt}(t) - k_{abs} \cdot G_{gut}(t) \tag{5.1}$$

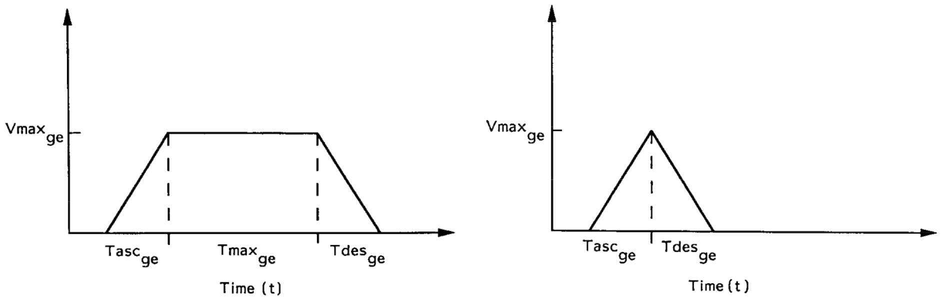
where G_{empt} is a function whose shape is either trapezoidal or triangular depending on the amount of carbohydrates in the meal (refer to fig. 5.1). The equation for G_{empt} is given by

$$G_{empt} = \begin{cases} (V_{max}/T_{asc})t & \text{if } t < T_{asc} \\ V_{max} & \text{if } T_{asc} \leq t < T_{asc} + T_{max} \\ V_{max} - (V_{max}/T_{des})(t - T_{asc} - T_{max}) & \text{if } T_{asc} + T_{max} \leq t < T_{asc} + T_{max} + T_{des} \\ 0 & \text{elsewhere} \end{cases} \tag{5.2}$$

where T_{max} describes the duration for which the gastric emptying is at its maximum (V_{max}), while T_{asc} and T_{des} describe the increase and decrease time for the gastric emptying to reach its maximum. V_{max} is given by

$$V_{max} = \frac{a \cdot CHO}{T_{asc} + 2 \cdot T_{max} + T_{des}} \tag{5.3}$$

with $a = 2$. The variable CHO describes the amount of ingested carbohydrates in grams. Default values of T_{asc} and T_{des} were set to 30 min. However, for small meals (CHO<10g), T_{max} is set to zero, so that G_{empt} becomes triangular. Graphically, G_{empt} looks like this:



(a) G_{empt} for CHO>10g

(b) G_{empt} for CHO<10g

Figure 5.1: Gastric emptying curves for different amounts of ingested carbohydrates. Figures taken from Lehmann and Deutsch[24]

The negative term in eq. (5.1) describes the absorption of glucose from the gut and into the blood. A graphical representation of the model can be seen in fig. 5.2 below.

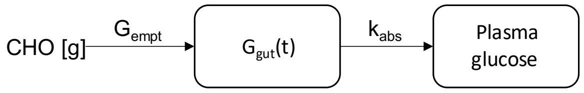


Figure 5.2: Scheme of meal model by Lehmann and Deutsch

The meal model by Lehmann and Deutsch was added to the Bergman model by Lynch and Bequette[25] in 2002. They also added a state describing subcutaneous glucose.

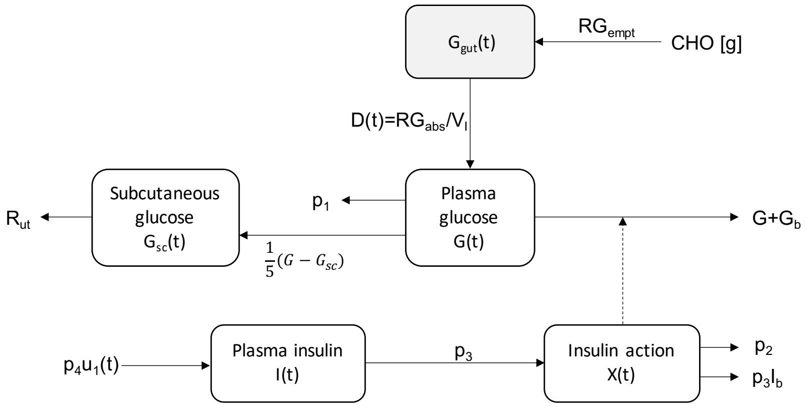


Figure 5.3: Scheme of model by Lynch and Bequette where the Lehmann & Deutsch meal model was coupled with the Bergman model. The gray compartment represents the meal model by Lehmann and Deutsch.

Roy and Parker, 2006[26]

This meal model is based on the structure of the Lehmann and Deutsch model, but extended to include protein and fat. The meal model was paired with a Bergman model that was augmented with free fatty acid (FFA) dynamics. The motivation for including FFA dynamics was that during rest, FFA stands for about 90% of the energy source for the muscles. The amounts of the three nutrients glucose, protein and fat in the gut are given by

$$\dot{N}_G(t) = x_G \cdot G_{empt}(t) - k_G \cdot N_G(t) \tag{5.4}$$

$$\dot{N}_P(t) = x_P \cdot G_{empt}(t) - k_P \cdot N_P(t) \tag{5.5}$$

$$\dot{N}_F(t) = x_F \cdot G_{empt}(t) - k_F \cdot N_F(t) \tag{5.6}$$

where G_{empt} is the same trapezoidal function as in Lehmann and Deutsch[24] (equation 5.2). The value for V_{max} (highest value of gastric emptying) depends on the amount of ingested nutrients in grams:

$$V_{max} = 2N_{tot}/(T_{asc} + 2 \cdot T_{max} + T_{des}) \tag{5.7}$$

Glucose and FFA then appears in the blood through the two equations

$$u_2(t) = k_G \cdot N_G(t) + 0.6k_P \cdot N_P(t) \tag{5.8}$$

$$u_3(t) = k_F \cdot N_F(t) \tag{5.9}$$

where it is assumed that 60% of proteins are turned into glucose. The absorption rates $u_2(t)$ and $u_3(t)$ are simply added as a disturbance to the differential equations describing $\dot{G}(t)$ and $\dot{F}(t)$.

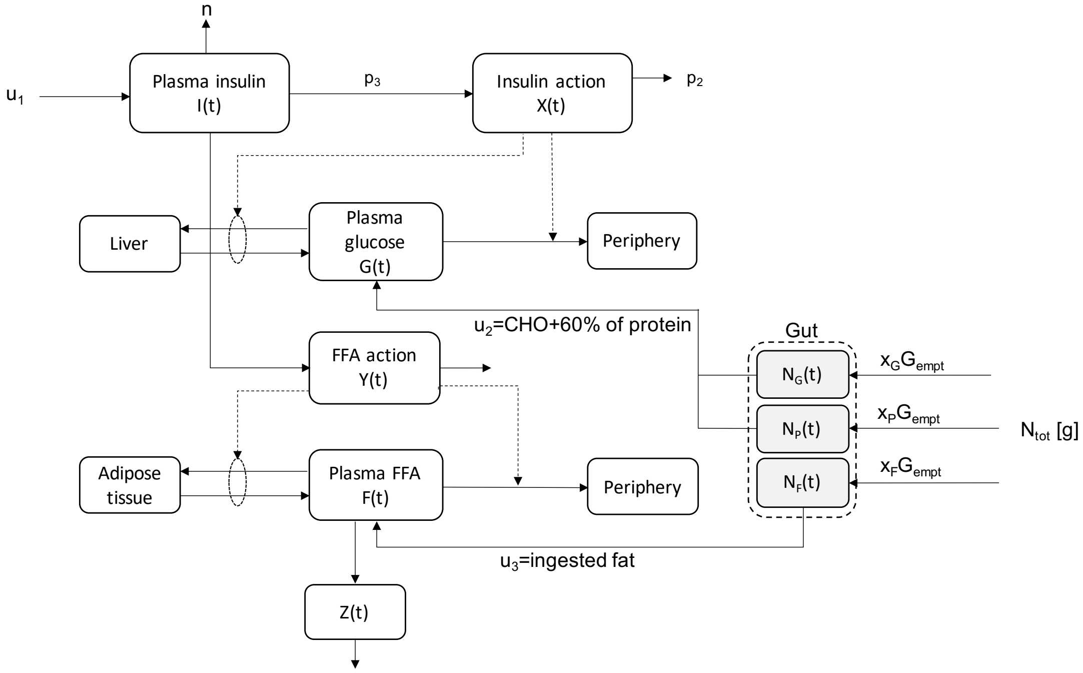


Figure 5.4: Scheme of extended Bergman model by Roy and Parker. The gray compartments represents the meal part of the model.

Data from a mixed meal tolerance test (CHO = 70 g, protein = 18 g, FFA = 20 g) was used in simulations. The value for V_{max} was 2.2 g/min from calculations of eq. (5.7) based on the meal composition. Other parameter values were $T_{max} = 35$ min, $k_G = 0.022$ min^{-1}, $k_P = 0.0097$ min^{-1}, $k_F = 0.015$ min^{-1} and $T_{asc} = T_{des} = 10$ min. The model successfully captured the dynamics of plasma glucose concentration.

Natalucci et al, 2003[27]

Natalucci presents in his paper a Bergman minimal model coupled with the meal dynamics by Lehmann and Deutsch[24]. However, the term describing gastric emptying has been modified and is no longer represented by a trapezoidal function as in the original model. The gut compartment is described by the equation

$$\dot{G}_{gut}(t) = G_{empt}(t) - k_{abs} \cdot G_{gut}(t) \tag{5.10}$$

i.e. the same as in the Lehmann and Deutsch meal model, but G_{empt} has been replaced by a power exponential function

$$G_{empt}(t) = Dk\beta \cdot \exp{(-kt)^{\beta}} \tag{5.11}$$

where D is the amount of ingested glucose, $\beta = 1.23$ is a shape factor and $k = 0.014$ is a velocity constant. The whole model can be visualized in fig. 5.5 below.

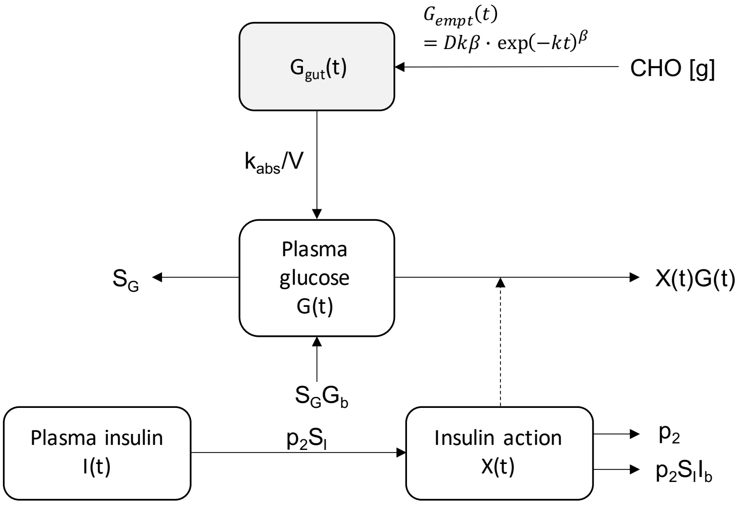


Figure 5.5: Scheme of extended Bergman model by Natalucci. The gray compartment represents the meal part of the model.

The reason for modeling the equation for G_{empt} as a power exponential was that such a curve had previously been used to fit a gastric emptying curve during an OGTT[28]. Data from nine non-diabetic subjects was used to obtain estimates of k_{abs} and the insulin sensitivity S_I.

UVA/Padova, 2017[21]

The UVA/Padova simulator has a subsystem describing the gastrointestinal tract (refer to fig. 3.6). This part couples ingested carbohydrates to glucose rate of appearance into the blood. The GI subsystem consists of a three compartments in a chain. The first two compartments describes the amount of glucose in the stomach in the solid and then liquid state. This corresponds to the grinding of the stomach to make chyme. The third compartment represents the gut. The equations are given by

$$Q_{sto}(t) = Q_{sto1}(t) + Q_{sto2}(t) \tag{5.12}$$

$$\dot{Q}_{sto1}(t) = -k_{gri} \cdot Q_{sto1} + D \cdot \delta(t) \tag{5.13}$$

$$\dot{Q}_{sto2}(t) = -k_{empt}(Q_{sto}) \cdot Q_{sto2}(t) + k_{gri} \cdot Q_{sto1}(t) \tag{5.14}$$

$$\dot{Q}_{gut}(t) = -k_{abs} \cdot Q_{gut}(t) + k_{empt}(Q_{sto}) \cdot Q_{sto2}(t) \tag{5.15}$$

$$Ra(t) = \frac{f \cdot k_{abs} \cdot Q_{gut}(t)}{BW} \tag{5.16}$$

where Q_{sto1} is amount of glucose in the stomach in the solid phase, Q_{sto2} in the liquid phase, Q_{gut} is amount of glucose in the intestine, D is the amount of ingested glucose, $\delta(t)$ is an impulse function, BW body weight and Ra rate of appearance. The rate k_{gri} is constant, whereas the gastric emptying rate $k_{empt}(Q_{sto})$ is a nonlinear function that depends on the amount of glucose in the stomach. The equation is given by

$$k_{empt}(Q_{sto}) = k_{min} + \frac{k_{max} - k_{min}}{2} \cdot [\tanh \alpha(Q_{sto} - b \cdot D) - \tanh \beta(Q_{sto} - c \cdot D) + 2] \tag{5.17}$$

where k_{min} and k_{max} refers to the minimum and maximum rate of gastric emptying. The constants α and β are computed as

$$\alpha = \frac{5}{2 \cdot D \cdot (1 - b)} \tag{5.18}$$

$$\beta = \frac{5}{2 \cdot D \cdot c} \tag{5.19}$$

A qualitative plot (fig. 5.6) may help to understand what this function looks like.

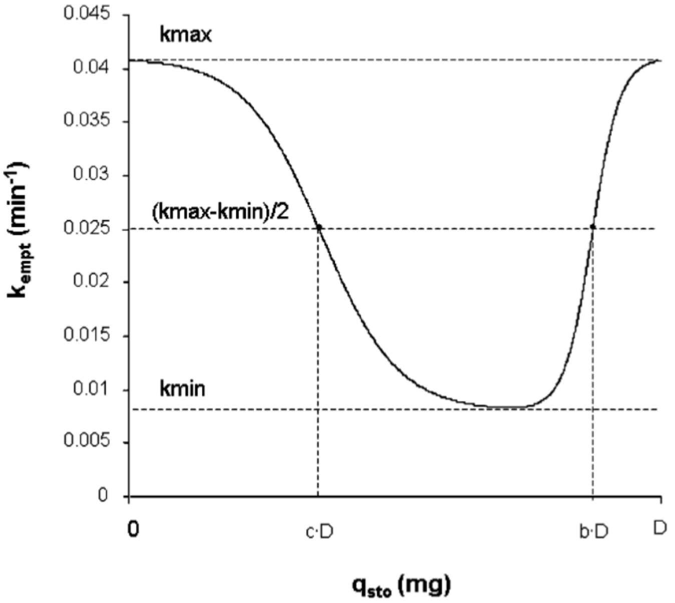


Figure 5.6: Gastric emptying in UVA/Padova meal model. Figure borrowed from Dalla Man et al.[29]

When the stomach contains the full amount of ingested glucose from a meal ($Q_{sto} = D$), the rate of gastric emptying will be maximum (k_{max}). As the amount of glucose in the stomach descreases, the rate of gastric emptying will also decrease until it reaches a minimum k_{min}. When the stomach is empty ($Q_{sto} = 0$), the transport of stomach contents to the gut will again be zero because the expression $k_{empt}(Q_{sto}) \cdot Q_{sto2}$ will be zero. Table 5.1 below summarizes the parameters of the UVA/Padova meal model.

Parameter	Meaning	Unit
k_{min}, k_{max}	Min. and max. rate of gastric emptying	min^{-1}
k_{abs}	Rate of absorption from the gut	min^{-1}
k_{gri}	Rate of grinding (from solid to liquid)	min^{-1}
D	Amount of ingested carbohydrates	mg
f	Fraction of intestinal absorption which appears in plasma	dimensionless
BW	Body weight	kg
α, β	Rate of decrease and increase of gastric emptying	
b, c	Constants	dimensionless

Table 5.1: Parameters in UVA/Padova meal model.

A visual representation of the meal model equations are shown below in fig. 5.7.

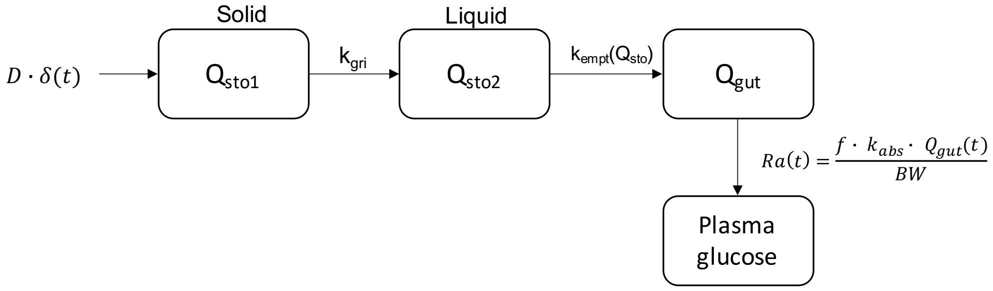


Figure 5.7: Meal subsystem in UVA/Padova

This meal model was first developed in Dalla Man et al., 2006[29] where four different meal models were validated using gold standard data on *Ra* obtained by a triple-tracer technique. The meal model that is now included in the UVA/Padova simulator were the one that performed best and fitted data from both an OGTT and a mixed meal in average and on an individual level. Another feature of this meal model is that it is able to distinguish between different *Ra* patterns, such as for instance diabetic versus non-diabetic subjects[29].

Hovorka, 2004[20]

The meal model by Hovorka is represented by a chain of two compartments. The equations are

$$\dot{D}_1(t) = A_G D_G - \frac{D_1(t)}{t_{max,G}} \tag{5.20}$$

$$\dot{D}_2(t) = \frac{D_1(t)}{t_{max,G}} - \frac{D_2(t)}{t_{max,G}} \tag{5.21}$$

where D_G is the amount of ingested carbohydrates and A_G is carbohydrate bioavailability (constant given by 0.8). Absorption into the accessible compartment Q_1 (where measurements of glucose are made) can be written as

$$U_G(t) = \frac{D_G A_G t e^{-t/t_{max,G}}}{t_{max,G}^2} \tag{5.22}$$

where $t_{max,G}$ is the time constant of the absorption, from the beginning of the meal consumption, for the absorption rate to reach its maximum. The whole Hovorka model with the meal subsystem indicated by grey boxes can be seen in fig. 5.8.

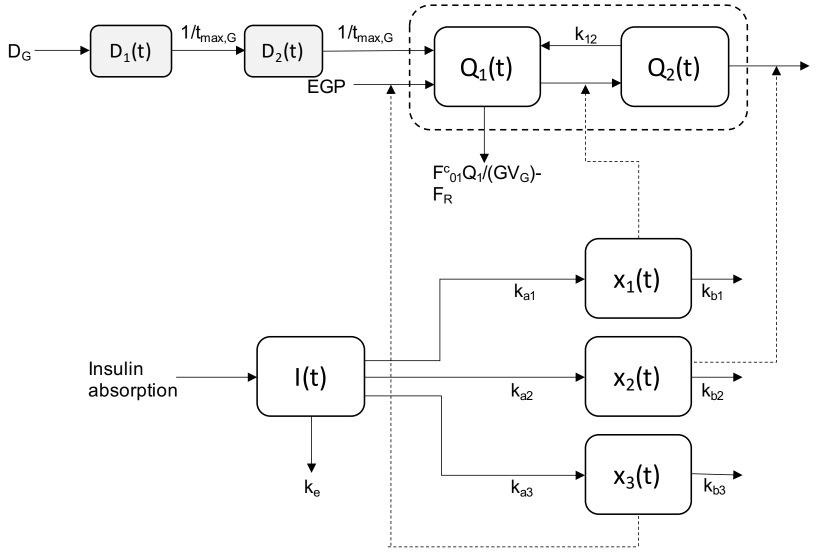


Figure 5.8: Scheme of Hovorka. Gray compartments are meal subsystem.

The paper that presented this meal model has not justified the model by any sort of physiological arguments. Neither are the meal compartments named after the stomach or the gut. It therefore seems like this meal structure is simply just a two-compartmental delay chain with no specific motivation behind other than the delay between ingestion of carbohydrates and the increase of BG levels.

Rashid et al, 2019[30]

This paper from 2019 presents a simulation software for testing control algorithms. The patient model is given by the Hovorka model but have been extended it to include exercise effects. The meal model is very similar to Hovorka, 2004[20], but have added a distinction between carbohydrates from a regular meal and rescue carbohydrates that act twice as fast.

The equations are

$$\dot{D}_1(t) = A_G \cdot D(t) - \frac{D_1}{t_{max,G}} \tag{5.23}$$

$$\dot{D}_2(t) = \frac{D_1}{t_{max,G}} - \frac{D_2}{t_{max,G}} \tag{5.24}$$

$$\dot{DH}_1(t) = A_G \cdot DH(t) - \frac{DH_1}{t_{max,g}} \tag{5.25}$$

$$\dot{DH}_2(t) = \frac{DH_1}{t_{max,g}} - \frac{DH_2}{t_{max,g}} \tag{5.26}$$

$$U_G(t) = \frac{D_2}{t_{max,G}} + \frac{DH_2}{t_{max,g}} \tag{5.27}$$

$$t_{max,g} = \frac{t_{max,G}}{2} \tag{5.28}$$

$D(t)$ and $DH(t)$ is the amount of carbohydrate intake for a regular meal and rescue carbohydrates respectively. The distinction between “regular” and fast-acting carbs allows for a little more accurate reproduction of a meal. A visual representation of the entire model can be seen in fig. 5.9 below.

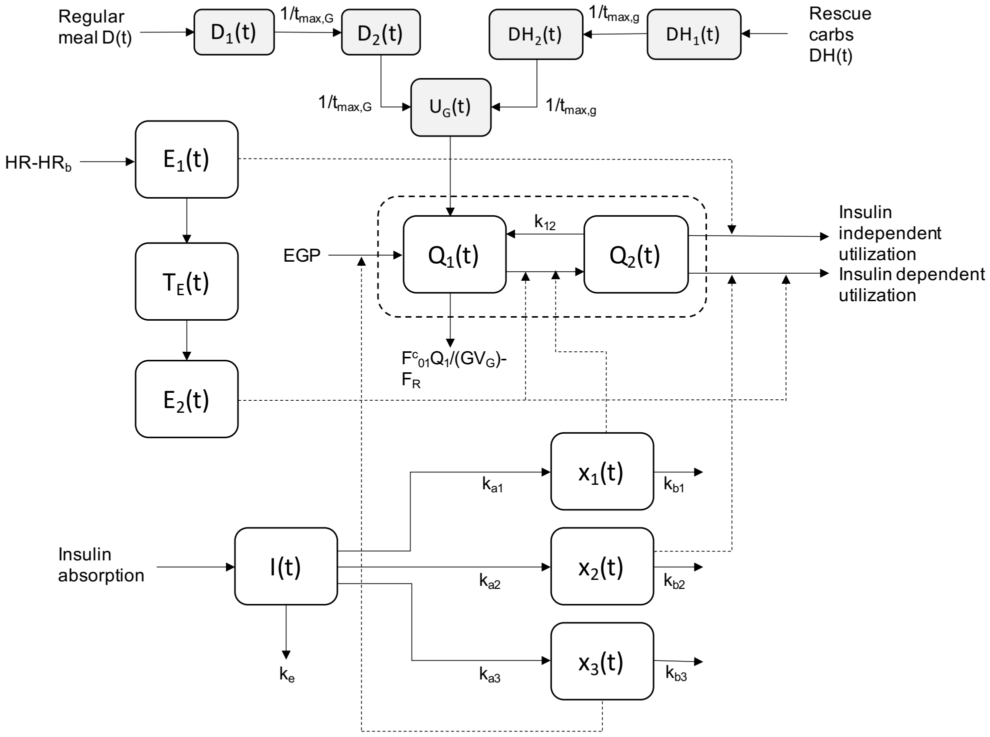


Figure 5.9: Scheme of model by Rashid et al. The gray compartments represent the meal subsystem. The compartments E_1, T_E and E_2 are exercise compartments whereas the rest of the comparments originates from the Hovorka model.

Fisher, 1991[31]

The meal model by Fisher is probably the most simple of the models presented in this literature review. It consists of one equation that describes the absorption rate of glucose into the blood directly. That is

$$P(t) = B \cdot \exp(-kt) \tag{5.29}$$

with $B = 0.5$ and $k = 0.05$. The model assumes that the meal occurs at $t = 0$. The reasoning behind this model is that in OGTTs with normal subjects, the plasma levels increase quite rapidly within 30 minutes and then fall to the base level within 2-3 hours. The absorption rate $P(t)$ was thus made to match that assumption. Figure 5.10 shows what eq. (5.29) looks like. At time $t = 0$ a meal is ingested, such that the rate of appearance is at 0.5. After 120 min, the absorption of glucose has decreased to 0.

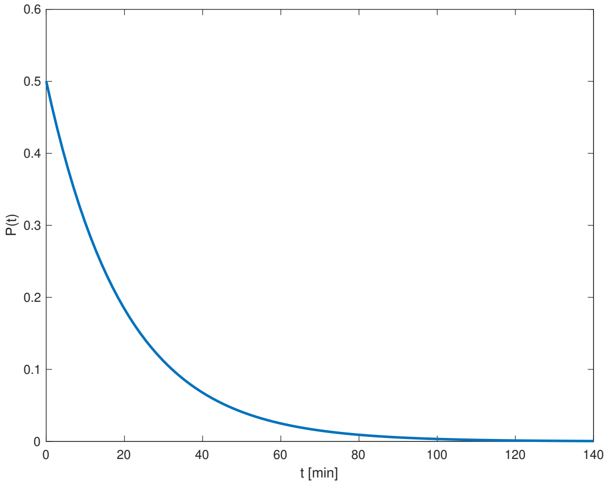


Figure 5.10: Rate of appearance $P(t)$ from Fisher model

The meal model was paired with the Bergman model and then used in simulations for glucose control where three different insulin infusion programs were tested. A lack of this model is that it does not include the amount of carbohydrates or any states describing the stomach or intestines. However, it is very simple and is easy to couple with a glucose model.

Farmer et al, 2009[32]

The meal model by Fisher[31] presented in the section above was modified slightly by Farmer and used in simulations of the Sorensen and Bergman models. The new model is given by the equation

$$D(t) = \begin{cases} 0 & \text{if } t < t_{meal} \\ A \cdot \exp(-b(t - t_{meal})) & \text{if } t \geq t_{meal} \end{cases} \tag{5.30}$$

where A represents the size of the meal and is given by

$$A = M_{meal}b \tag{5.31}$$

where M_{meal} is the amount of carbohydrate content and $b = 0.05$ is the absorption rate. Meals can now occur at any time (t_{meal}) and a plot of the absorption of glucose after a meal is shown in fig. 5.11.

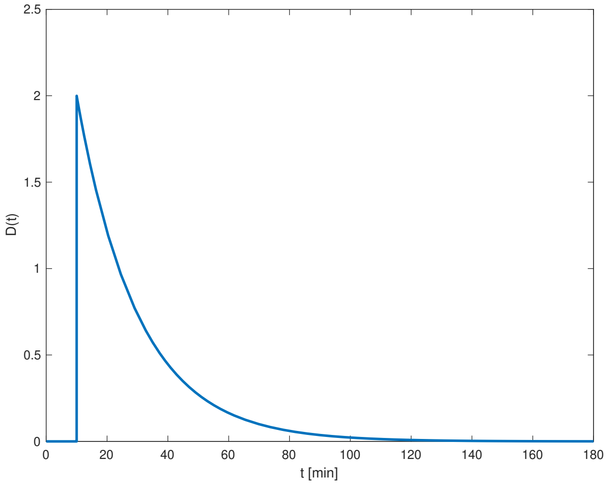


Figure 5.11: Plot showing $D(t)$ from the Farmer meal model. A meal of 40g carbohydrates ingested at $t_{meal} = 10$min. After 120 minutes from the meal started, the absorption has returned to zero.

In the paper by Farmer, the meal model was then used in simulations of a 50g carbohydrate meal with both the Bergman model and the Sorensen model. It was also used in control schemes. This model is still very simple, but allows for differences in the amplitude (determined by A) and decay (determined by b).

Rozendaal et al, 2015[33]

This model, also called the Eindhoven Diabetes Education Simulator (E-DES), is a physiological model consisting of four compartments. That is gut, plasma, interstitial fluid and subcutaneous tissue. The meal subsystem is included via the gut compartment with the equation

$$\frac{dM_G^{gut}(t)}{dt} = \sigma k_1^\sigma t^{\sigma-1} \cdot e^{-(k_1 t)^\sigma} \cdot D_{meal} - k_2 \cdot M_G^{gut}(t) \tag{5.32}$$

that represents the amount of glucose in the gut. The parameters are

Parameter	Meaning	Value	Unit
k_1	Rate of glucose appearance in gut	0.0145	min^{-1}
k_2	Rate of gut emptying	0.276	min^{-1}
σ	Shape factor	1.34	dimensionless
D_{meal}	Amount of ingested carbohydrates	-	g

Table 5.2: Parameters in E-DES meal model.

Glucose from the gut then appears in the glucose plasma equation as a disturbance. This is represented in eq. (5.33) by the first term on the right hand side of the equation sign.

$$\frac{dG^{pl}(t)}{dt} = k_2 \frac{f}{V_G M^b} M_G^{gut}(t) + g^{liv}(t) - g^{non-it}(t) - g^{it}(t) - g^{ren}(t) \tag{5.33}$$

The other terms represent inflows and outflows to the glucose plasma compartment, such as endogenous glucose production or renal excretion. The entire glucose-insulin model by Rozendaal et al is shown in fig. 5.12 below.

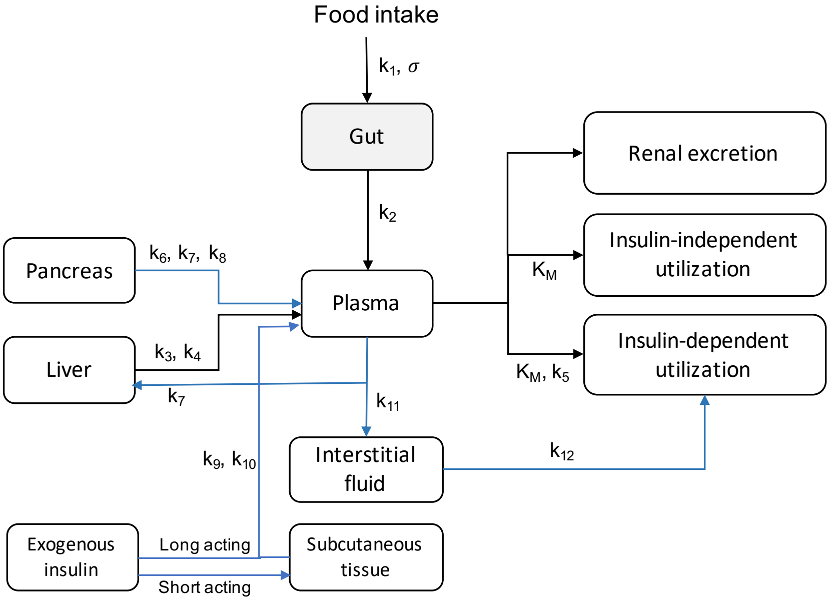


Figure 5.12: Rozendaal scheme. Blue arrows represent insulin fluxes, black arrows are glucose fluxes. The amount of glucose and/or insulin is only computed in gut, plasma, interstitial fluid and subcutaneous tissue compartments. The meal part is represented by the gray compartment.

The parameters of the model were found through nonlinear least squares optimization on data from healthy subjects. This model has later been used in a study on the effect of different types of food[34].

Fabietti et al, 2006[35]

The model by Fabietti et al. is a control-oriented model of glucose and insulin dynamics. It is developed from the Bergman minimal model, but has been extended to include more compartments. The meal part of this model is included via the exogenous glucose compartment (fig. 5.14). The intake of carbohydrates is split into three terms depending on which type of carbohydrate is ingested. The meal model equations are given by

$$E_g = A_g(s) + A_s(s) + A_m(s) \tag{5.34}$$

where $A_g(s)$ is absorption rate of sugar, $A_s(s)$ fast absorption rate of starch, and $A_m(s)$ slow absorption of starch from a mixed meal. The equations for the absorption rates are

$$A_g(s) = (1 - F_s)\frac{16.6}{(s + 1.44)(s + 135)}R_i(s) \tag{5.35}$$

$$A_s(s) = F_s(1 - F_m)\frac{467}{(s + 1.61)(s + 7.20)(s + 7.18)}R_i(s) \tag{5.36}$$

$$A_m(s) = F_sF_m\frac{75.1}{(s + 0.466)(s + 5.54)(s + 5.86)(s + 6.43)}R_i(s) \tag{5.37}$$

where F_s is the fraction of starch in the meal and $(1 - F_s)$ the fraction of sugar. F_m is the fraction of a mixed meal in the total amount of starch. An OGTT is thus represented by setting $F_s = 0$. $R_i(s)$ is the rate of ingestion.

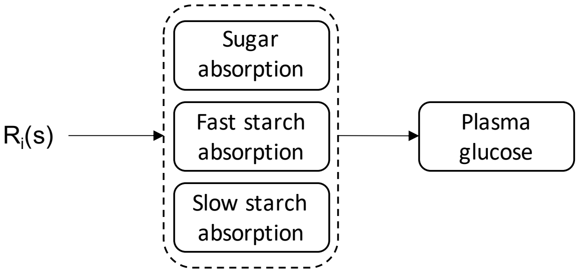


Figure 5.13: Scheme of meal model by Fabietti et al. Absorption dynamics of sugar, fast starch and slow starch are represented by 2nd, 3rd and 4th order transfer functions respectively.

A figure of the whole model is shown below.

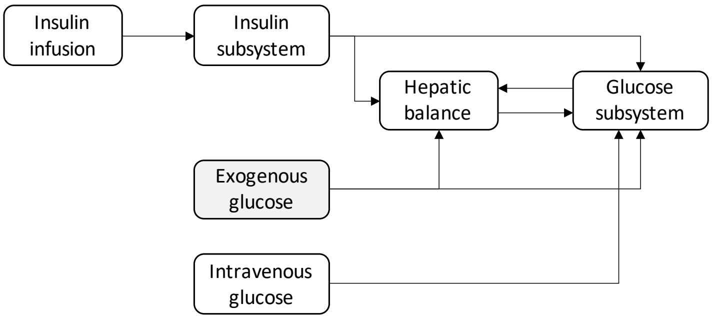


Figure 5.14: Scheme of model by Fabietti et al. Meals are included in the exogenous glucose-compartment (gray box). Insulin subsystem consists of 3 compartments, glucose subsystem consists of 2 compartments.

An advantage of this model is that it includes different absorption rates. As discussed in section 2, carbohydrates need to be broken into monosaccharides before absorption, and this causes more complex carbohydrates to be absorbed more slowly than for instance sugar. It is therefore more physiologically accurate to include these different types of absorption rates.

Lema-Perez et al, 2018[36]

This model aims to be very physiologically accurate. Instead of using lumped compartments, it models real process systems. However, this is not an entire meal model, but only a stomach submodel. The stomach is divided into three compartments, stomach blood, stomach wall and stomach contents. The stomach contents compartment models the amount of gastric mass in the stomach, such as carbohydrates, fat, proteins, fiber, water etc. The stomach wall produces gastric juices that are released into the stomach contents. The last compartment represents the capillaries around the stomach. The model is shown in fig. 5.15.

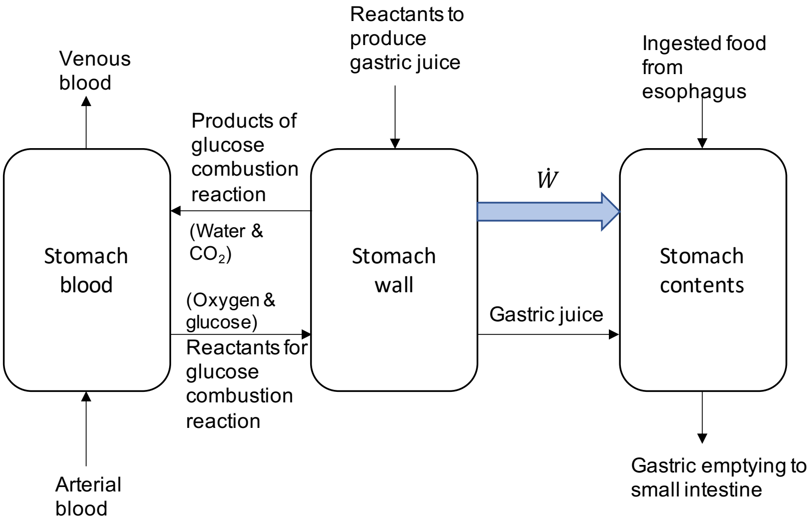

Venous blood
Reactants to produce gastric juice
Ingested food from esophagus
Products of glucose combustion reaction
$\dot{W}$
(Water & CO_2)
Stomach blood
Stomach wall
Stomach contents
(Oxygen & glucose)
Reactants for glucose combustion reaction
Gastric juice
Arterial blood
Gastric emptying to small intestine

Figure 5.15: Stomach model by Lema-Perez et al.

This model consists of many equations and variables that will not be listed here. Since the model is so extensive, it is maybe not the best one suited for control schemes, but can nevertheless provide useful information. The stomach model is planned to be part of a whole-body model developed by the same research group.

5.3 Discussion

All the models presented in the previous section describe how the glucose dynamics changes after a meal. Even though they all aim to solve the same problem, there are a lot of different solutions. However, there are some things that are similar for many of the models. Such a thing is the model structure. From the literature review, it has become clear that there are two model structures that are common. That is to either model the GI system as a two-compartment chain that typically represent the stomach and the gut, or to model it through one compartment where the stomach and gut are lumped together. The models that use one compartment are Rozendaal et al[33], Natalucci et al[27], Roy and Parker[26], Lehmann and Deutsch[24]. This type of structure is shown in fig. 5.16.

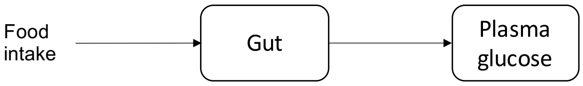


Figure 5.16: Meal models with one compartment

The meal models that model the stomach and gut as separate compartments include Hovorka[20], Rashid et al[30] and UVA/Padova[21]. The UVA/Padova meal model actually further divide the stomach into two compartments describing solid and liquid food (i.e. the process of making chyme). Figure 5.17 shows what this structure looks like.

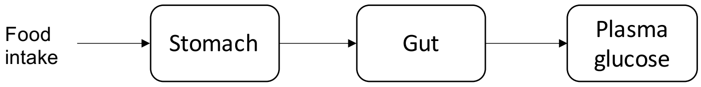


Figure 5.17: Meal models with two compartments

Using two compartments definitely increases the physiological accuracy, but is also increasing the number of states and parameters. If the goal is the use a minimal model, it is maybe sufficient with one compartment as in fig. 5.16.

Gastric emptying is another factor that is important to capture. Gastric emptying refers to the emptying of the stomach contents into the small intestine. Some models were too simple to include gastric emptying (Fisher[31], Farmer[32], Fabietti[35]), while other models includes it. The models by Lehmann & Deutch[24] and Roy & Parker[26] models gastric emptying as a trapezoidal curve, while Natalucci[27] and Rozendaal[33] uses a power exponential. The UVA/Padova meal model[21] uses a nonlinear rate between the stomach and gut compartment.

Another important factor to model is the rate of appearance (Ra) of glucose from food into the blood. In other words, at which rate and amount ingested carbohydrates (glucose) appear in the blood. Three models that just modeled this directly without any states describing the stomach or gut, were the models by Fisher[31], Farmer[32] and Fabietti[35]. Other models typically modeled Ra through first order linear kinetics dependent on the amount of glucose in the gut-compartment. The equation for rate of appearance would then look something like $Ra(t) = k \cdot G_{gut}(t)$ where k is a constant. This approach is seen in UVA/Padova[21], Rozendaal[33], Natalucci[27], Roy and Parker[26], and Lehmann and Deutsch[24]. The models by Hovorka[20] and Rashid[30] also use this approach even though they don't specify it as Ra.

It is also important for a meal model to be accurate that it includes more info about the ingested food. Either by giving nuances to the type of ingested carbohydrate, or to include the effect of ingested fat or protein. There were actually only two models that included different types of carbohydrates. The model by Rashid et al[30] used "regular meal" and "rescue carbohydrates" that was absorbed twice as fast. Fabietti[35] had three types of carbohydrates with different absorption rates; sugar, fast starch and slow starch. There were also only two models including other macronutrients. The meal model by Roy and Parker[26] considered a meal of carbohydrates, fat and protein. The extensive stomach model by Lema-Perez[36] also modeled reaction kinetics and dynamics of many substrates, including carbohydrates, fat and proteins.

There are in total a lot of factors to think about when modeling the GI system into a glucose-insulin model. The models from the literature review have been compared on the basis of some of these factors. That is summarized in table 5.3.

5.4 Summary

A total of 11 meal models have been presented. Their similarities and differences have been discussed, and it was seen that two model structures were common. The models were also compared on the basis of how they model gastric emptying, rate of appearance and which type of meal input they take. These factors are summarized in table 5.3.

Model	**Amount of CHO**	**Type of CHO**	**Mixed meal Fat, protein**	**Stomach, gastric empt.**	**Gut, rate of appearance**	**Number of compartments**
Lehmann & Deutsch, 1992	✓[g]			✓	✓	1
Roy & Parker, 2006	✓[g]		✓	✓	✓	1
Natalucci, 2003	✓[g]			✓	✓	1
UVA/Padova, 2017	✓[mg]			✓	✓	3
Hovorka, 2004	✓[g]			✓	✓	2
Rashid, 2019	✓[g]	✓		✓	✓	4
Fisher, 1991					✓	0
Farmer, 2009	✓[mg]				✓	0
Rozendaal, 2015	✓[mg]			✓	✓	1
Fabietti, 2006	✓	✓			✓	1
Lema-Perez, 2018	✓[mol]		✓	✓	✓	3

Table 5.3: Summary of meal models

6 Meal model identification setup

This section will go through the identification that was performed on three of the meal models from the literature review in section 5. The purpose of this was to assess how the models performed compared to real glucose data, i.e. to see if the simulated postprandial glucose response resembled this measured data. A publicly available data set was used to perform the parameter identification. The data had to be prepared to be used for identification, that is, converted to a format that could be used as an input to the differential equations. The meal models were coupled with the Sorensen model. A subcutaneous insulin model was also included.

6.1 Data set

The data that was used in identification was the Ohio T1DM data set[2] (2018 version), which consists of data for six people with T1D. Four of them were female and two were male. The subjects wore a Medtronic 530G insulin pump and a Medtronic Enlite CGM sensor together with a Basis Peak fitness band. The subjects also reported life-event data such as stress, sleep, meals and work. The data is stored in an XML-file for each patient with the following entries; patient info, glucose level, finger stick, basal insulin, temp basal insulin, insulin bolus, meal, sleep, work, stressors, hypo event, illness, exercise, heart rate, galvanic skin response, skin temperature, air temperature, steps and sleep from Basis band. The data also contains a viewer which enables data visualization.

6.2 Data preparation

The XML-files containing the Ohio data set were imported into MATLAB and then made into a table for each data entry (such as glucose level or meals) with timestamps in one column and the measured value in the other. These tables were also stored as excel files. However, since the data was intended to be used in the Sorensen simulator, the data had to be interpolated to match the simulator's timestep of 0.1 minute. Glucose data that was measured every 5th minute was interpolated to every 0.1 minute using a Kalman filter developed for automatic preprosessing of glucose data[37] as shown in fig. 6.1. This also provided estimates of the glucose level at times where the CGM measurements were missing.

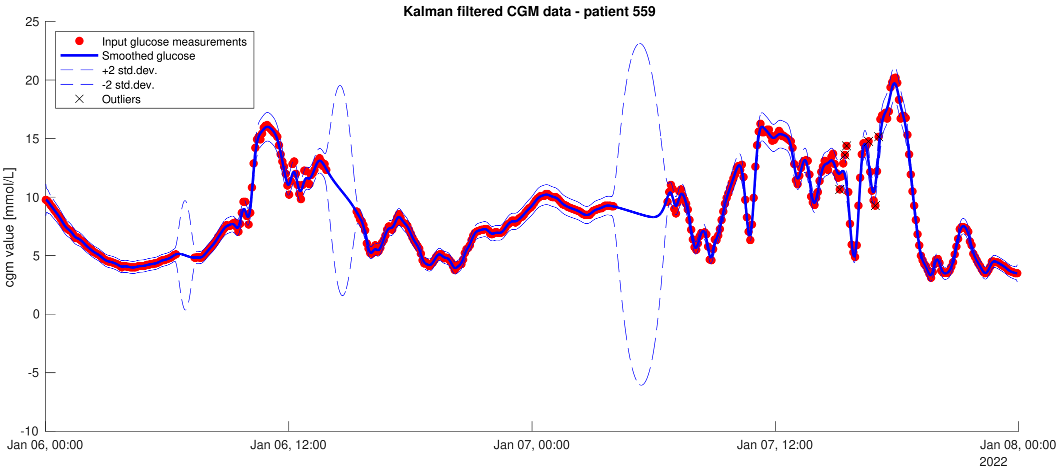
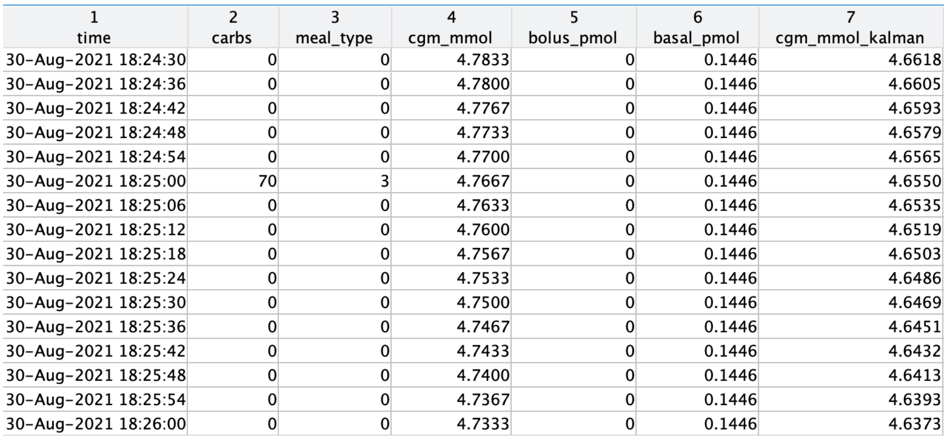


Figure 6.1: Kalman filtered CGM data of two days. The three spots where CGM data is missing is filled with estimated glucose values which is represented by the blue line. The dashed lines represent the estimated value ± 2 standard deviations.

Meal data from the Ohio data set were filled with zeros for the timestamps where food was not ingested. The type of meal was also included as a number from 1 to 5, describing breakfast, lunch, dinner, snack and hypo correction respectively. Bolus insulin data were filled with zeros at the timestamps where not infused. Basal and temp basal data had to be combined since the temp basal insulin supersedes the basal rate when active. Lastly, the interpolated glucose, meal and insulin data was combined in one large table for each patient. The units in the Ohio T1DM dataset was mg/dL for glucose measurements, U (units) and U/hour for bolus and basal insulin, and grams for carbohydrates. The glucose and insulin data had to be converted to the same units as the Sorensen simulator uses, i.e. mmol/L (divide by 18.018) and pmol/L (multiply by 6.94) respectively. The table was then stored as an excel file, and the data was now ready to be used as input to the Sorensen simulator. The structure of the data was then looking like the table shown in fig. 6.2.



1	2	3	4	5	6	7
time	carbs	meal_type	cgm_mmol	bolus_pmol	basal_pmol	cgm_mmol_kalman
30-Aug-2021 18:24:30	0	0	4.7833	0	0.1446	4.6618
30-Aug-2021 18:24:36	0	0	4.7800	0	0.1446	4.6605
30-Aug-2021 18:24:42	0	0	4.7767	0	0.1446	4.6593
30-Aug-2021 18:24:48	0	0	4.7733	0	0.1446	4.6579
30-Aug-2021 18:24:54	0	0	4.7700	0	0.1446	4.6565
30-Aug-2021 18:25:00	70	3	4.7667	0	0.1446	4.6550
30-Aug-2021 18:25:06	0	0	4.7633	0	0.1446	4.6535
30-Aug-2021 18:25:12	0	0	4.7600	0	0.1446	4.6519
30-Aug-2021 18:25:18	0	0	4.7567	0	0.1446	4.6503
30-Aug-2021 18:25:24	0	0	4.7533	0	0.1446	4.6486
30-Aug-2021 18:25:30	0	0	4.7500	0	0.1446	4.6469
30-Aug-2021 18:25:36	0	0	4.7467	0	0.1446	4.6451
30-Aug-2021 18:25:42	0	0	4.7433	0	0.1446	4.6432
30-Aug-2021 18:25:48	0	0	4.7400	0	0.1446	4.6413
30-Aug-2021 18:25:54	0	0	4.7367	0	0.1446	4.6393
30-Aug-2021 18:26:00	0	0	4.7333	0	0.1446	4.6373

Figure 6.2: Table with formatted data. This is from patient 588.

The dates of the Ohio dataset have been shifted by a random amount into the future in order to de-identify the dataset. However, days of the week and times of the day have been kept the same.

6.3 Sorensen simulator

The glucose simulator based on the Sorensen model uses the revised equations from Panunzi et al.[38] where some equations and parameter values of the original Sorensen model have been updated. It is implemented in MATLAB and uses a 4th order Runge-Kutta method to solve the differential equations. The code is available at Github at `https://github.com/iasi-cnr/A-Revised-Sorensen-Model`. The simulator had to be modified to include subcutaneous insulin infusion so that the insulin data from the Ohio T1DM dataset could be used. The insulin model that was implemented was the Wilinska 2005[39] (model 1) (also in Hovorka 2004[20]) model of subcutaneous insulin, which is a two-compartment model. The equations are given by

$$\dot{S}_1(t) = u(t) - k_a \cdot S_1(t) \tag{6.1}$$

$$\dot{S}_2(t) = k_a \cdot S_1(t) - k_a \cdot S_2(t) \tag{6.2}$$

$$\Gamma_{ISC} = k_a \cdot S_2(t) \tag{6.3}$$

with appearance in the heart and lungs compartment as

$$\frac{dI_H(t)}{dt} = \frac{1}{V_{IH}}(\Gamma_{ISC} + Q_B^I I_B + Q_L^I I_L + Q_K^I I_K + Q_P^I I_{PV} - Q_H^I I_H) \tag{6.4}$$

The parameter k_a represents the time constant for transfer between the subcutaneous insulin compartments ($k_a = \frac{1}{T_{ins}}$) and was set to 0.0144 as in Rashid 2019[30]. Figure 6.3 shows how the subcutaneous insulin was added. Meal models were coupled with the gut compartment through the rate Γ_{meal} in the equation

$$\frac{dG_G(t)}{dt} = \frac{Q_G^G}{V_G^G}(G_H(t) - G_G(t)) + \frac{1}{V_G^G}(\Gamma_{meal} - \Gamma_{GGU}) \tag{6.5}$$

as shown in fig. 6.3. For the meal models that had their own gut compartment (Lehmann & Deutsch and UVA/Padova), this compartment was dropped and Sorensen's own gut compartment was used instead. How this was done is shown in detail in the appendix.

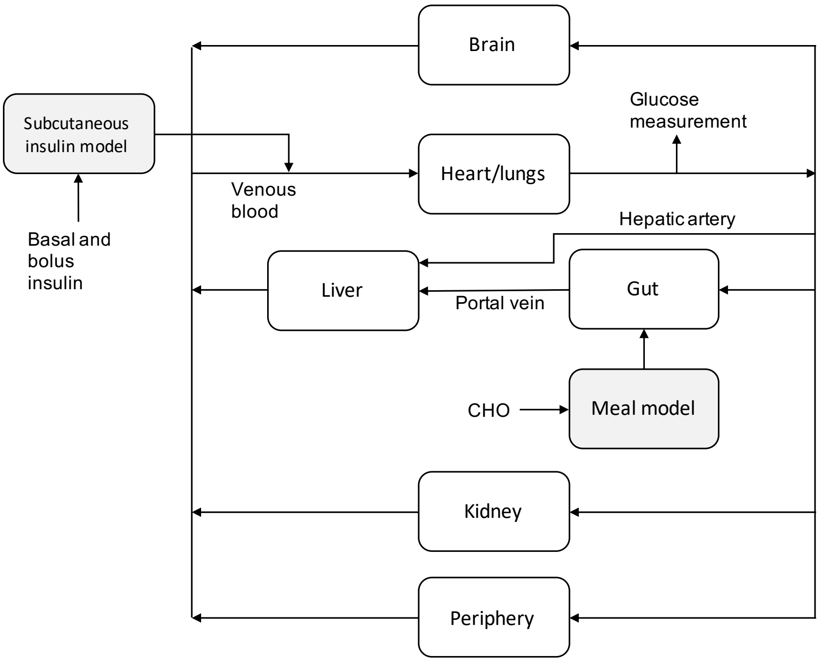


Figure 6.3: Adding meal and subcutaneous insulin models to the Sorensen model.

6.4 Meal models

Meal models from the literature review to identify were chosen by their compatibility with both the Sorensen model and the Ohio T1DM dataset. Only models that were compatible with both could be used. For example, the model by Roy & Parker could not be used since it needs data on ingested fat and protein as well as carbohydrate, which is something the Ohio dataset does not contain. The model also needs compartments modeling FFA dynamics, which is something the Sorensen model does not have. Similarly, other meal models were ruled out, leaving us with six possible models (Lehmann & Deutsch, Natalucci, UVA/Padova, Hovorka, Farmer, Rozendaal). Table 6.1 below shows which models were compatible and not.

Meal model	**Compatible with Sorensen**	**Can use Ohio dataset**
Lehmann & Deutsch	Yes	Yes. Only need CHO data
Roy & Parker	No. Need to model FFA dynamics	No. Need FFA and protein data
Natalucci	Yes	Yes. Only need CHO data
UVA/Padova	Yes	Yes. Only need CHO data
Hovorka	Yes	Yes. Only need CHO data
Rashid	Yes	No. Need type of CHO
Fisher	Yes	No. No CHO input
Farmer	Yes	Yes. Only need CHO data
Rozendaal	Yes	Yes. Only need CHO data
Fabietti	No. Control oriented	No. Need type of CHO
Lema-Perez	No. Too extensive	No. Need data on many substrates

Table 6.1: Compatibility of meal models with Sorensen simulator and Ohio T1DM dataset.

After this, three models of different complexity levels were picked out to be identified. These were Farmer, UVA/Padova and Lehmann & Deutsch. The models were then implemented in the Sorensen simulator by modifying the state vector and adding the necessary algebraic and differential equations.

6.5 Identification and training data

Identification of parameters was implemented through the MATLAB function `fmincon`. The cost function to minimize was defined as the MSE between glucose output from simulating the Sorensen model (heart and lungs compartment) and glucose data from the Ohio dataset. The goal was then to find the optimal choice of meal model parameters, meaning the parameters that yields a simulation output that is as close to the empirical dataset as possible such that the cost function is minimized.

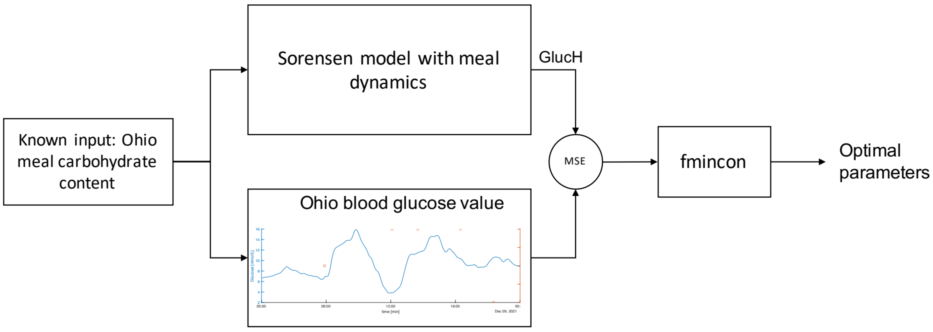
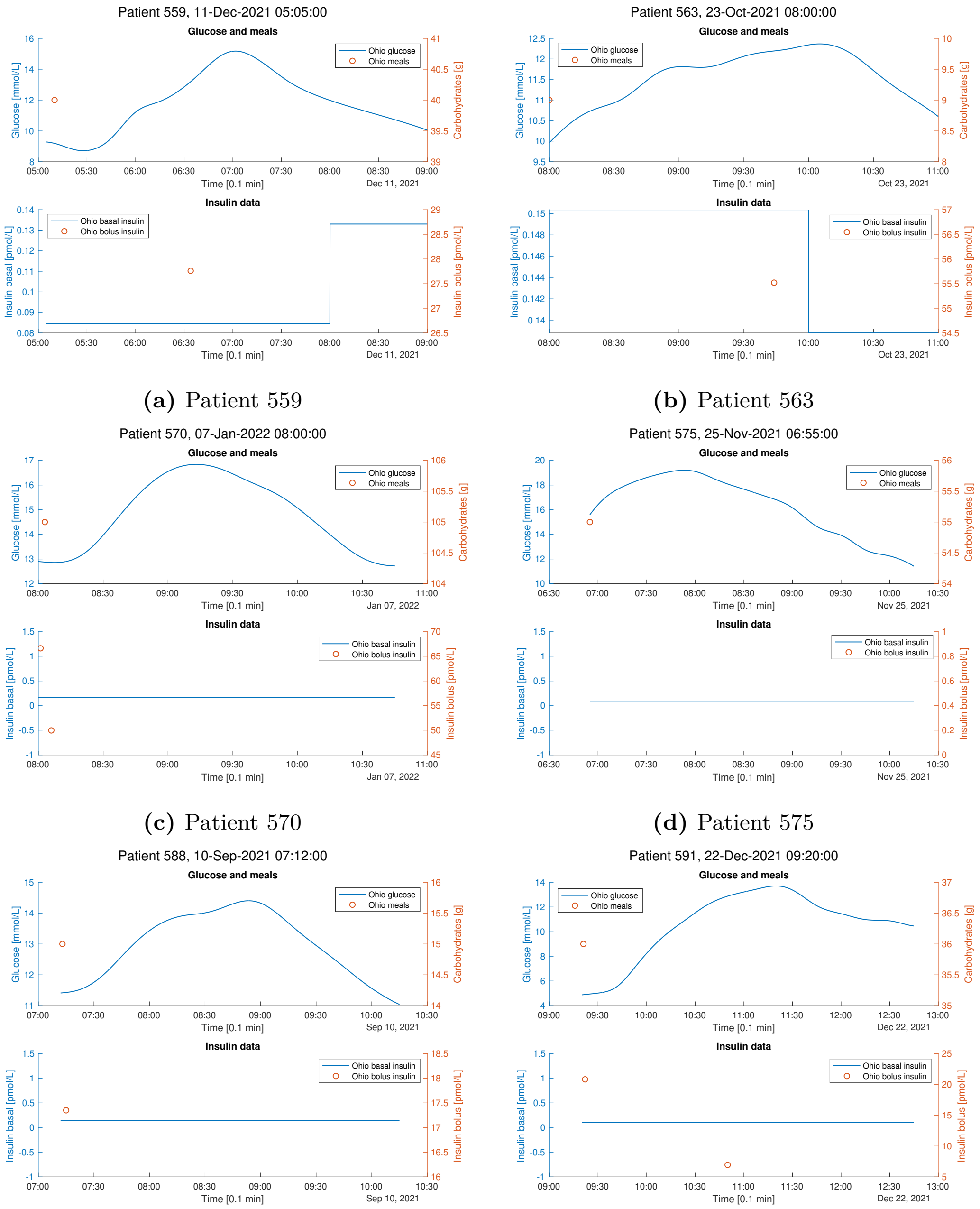

Known input: Ohio meal carbohydrate content
Sorensen model with meal dynamics
GlucH
Ohio blood glucose value
MSE
fmincon
Optimal parameters

Figure 6.4: Parameter identification of meal models was performed through minimizing the MSE between the measured glucose data and the simulated glucose data. The MSE was then used as input to the `fmincon` function.

The parameters in the three different meal models were identified on one meal (breakfast) from each of the 6 patients. Plots of these meals are shown below in fig. 6.5. They range from a small carbohydrate content (9g) to a large carbohydrate content (105g). The meals were chosen such that they had a couple of hours before next meal was ingested, so that the postprandial glucose response was as clear as possible and not disturbed by a next meal. They were also chosen based on the quality of the CGM measurement, meaning that meals where a lot of data was missing was not included. Before optimizing, some time was spent to find a good initial point for the parameters before optimization. This is because `fmincon` is a local optimizer, and the optimized minimum relies heavily on the initial point. This included some manual tuning by trial and error which took approximately 10 minutes for each patient. Once a satisfactory initial point was found, the optimization algorithm was run.



(a) Patient 559

(b) Patient 563

(c) Patient 570

(d) Patient 575

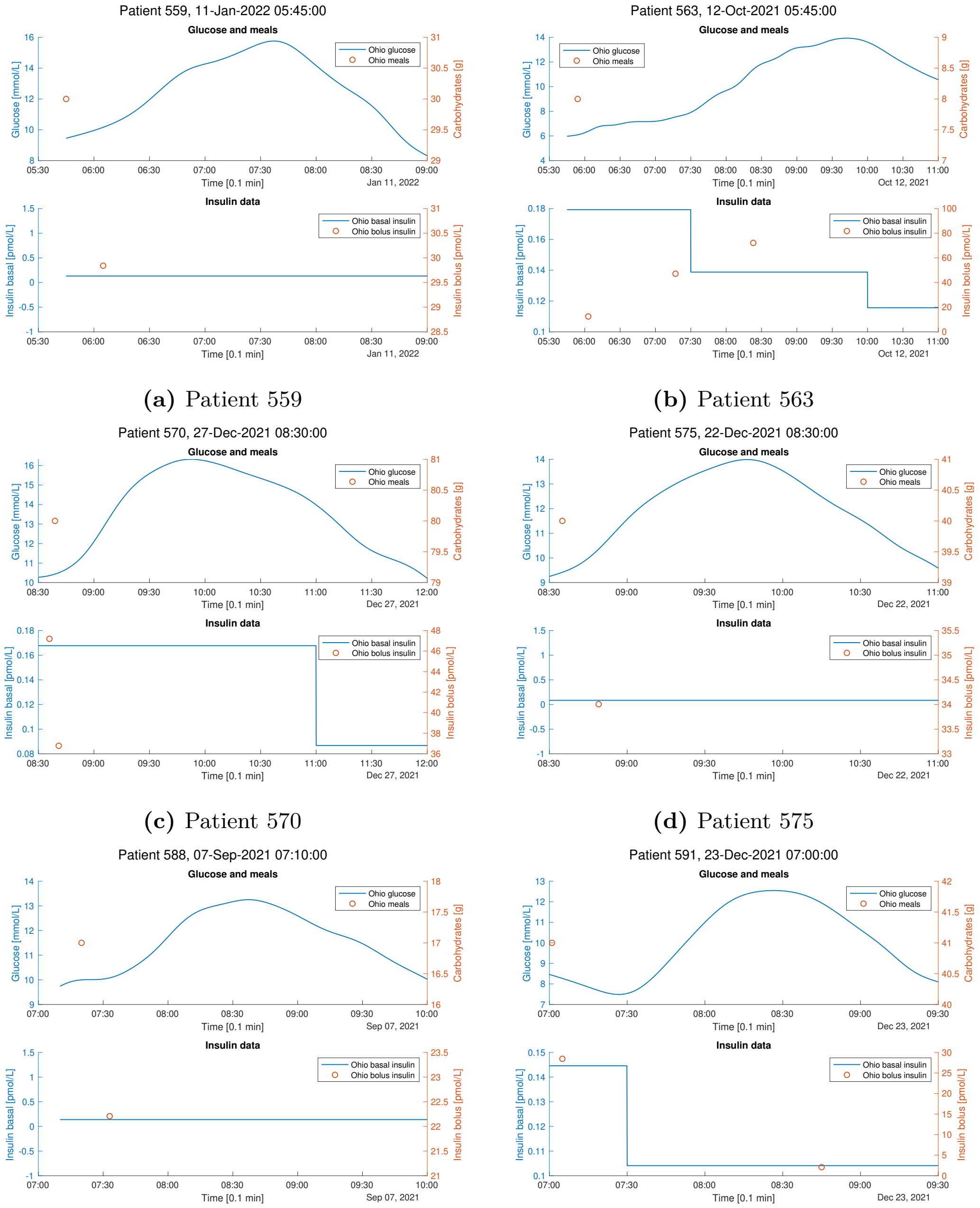
(e) Patient 588

(f) Patient 591

Figure 6.5: Training data. Glucose, meal and insulin data from the 6 meals in the Ohio T1DM dataset that was used in parameter identification.

6.6 Evaluation and test data

After the parameters have been identified on the training dataset presented in fig. 6.5, the results will be tested on data that are previously unknown to the optimizer (test dataset). The test data was picked out to be meals of the same type (breakfast) and approximately same carbohydrate content as the training data. For example, for patient 559, the training meal size was 30g and validation meal was 40g. For patient 563, this was 9g and 8g respectively. Figure 6.6 shows the glucose and insulin for the test dataset.



(a) Patient 559

(b) Patient 563

(c) Patient 570

(d) Patient 575

(e) Patient 588

(f) Patient 591

Figure 6.6: Test data. Glucose, meal and insulin data from the 6 meals in the Ohio T1DM dataset that was used in evaluation.

6.7 Summary

This section has presented the setup of the parameter identification of three meal models. The dataset that was used have been presented, including how it was formatted. How the three meal models have been coupled with the Sorensen glucose simulator has also been discussed. The results of the identification and testing are given in the next section.

7 Results of meal model identification and evaluation

This section will go through the results from the identification and evaluation of the meal model parameters.

7.1 Farmer

7.1.1 Identification

The only parameter that were to be fitted was the parameter b in eq. (5.30) with the original value of 0.05. This parameter controls both the amplitude and decay of the exponential function. However, simulations showed that using an amplitude of $CHO \cdot 0.05$ was way too small (for example, a meal of 40g carbohydrate would only increase glucose levels by 2 mmol/L). An additional constant a was therefore included instead to control the amplitude such that the equation now became

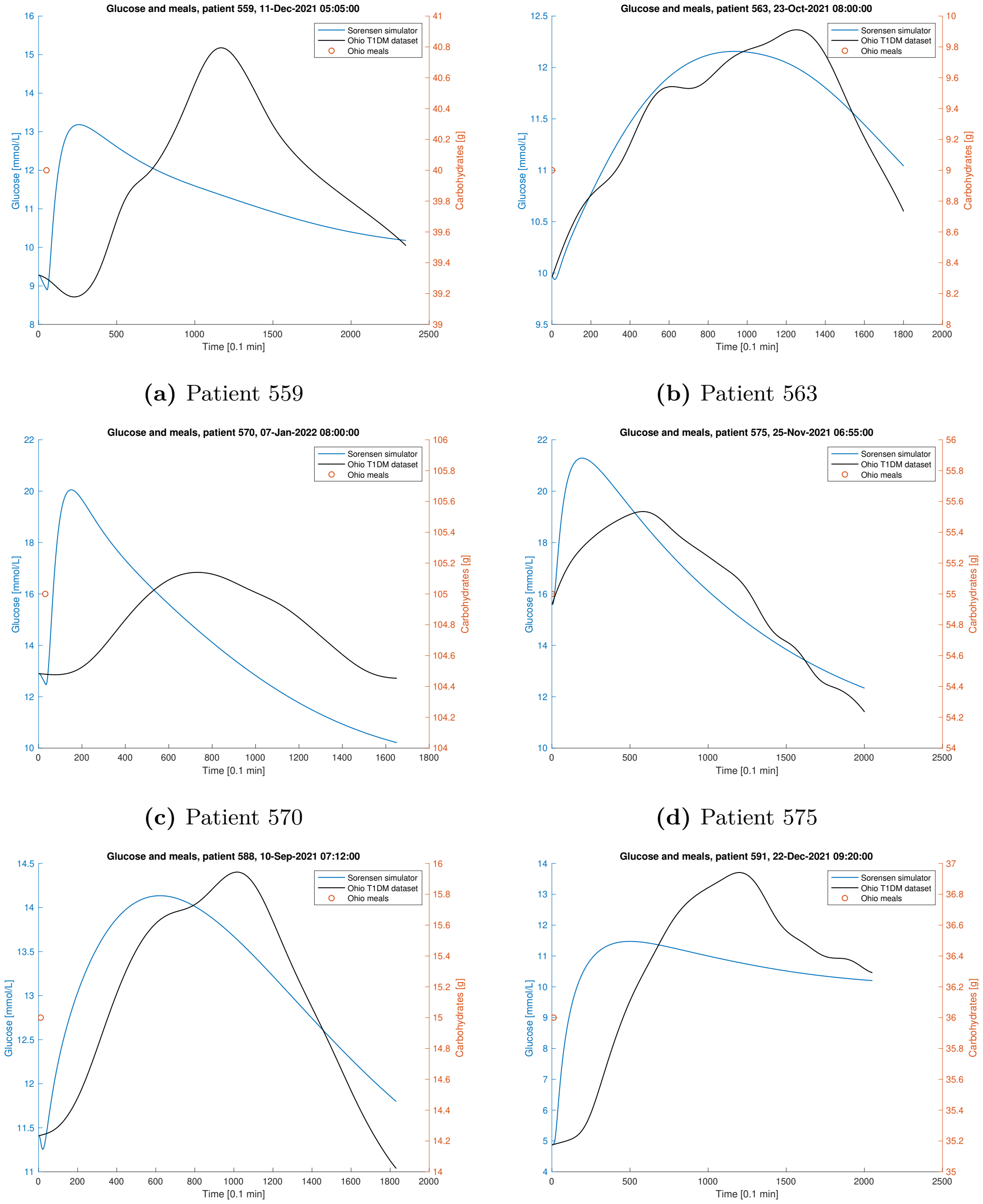
$$\Gamma_{meal} = CHO \cdot a \cdot e^{-b(t-t_{meal})} \tag{7.1}$$

with $a = 0.2$. The results of the parameter identification are shown in table 7.1 below.

Patient	Date	CHO	Init. param.	Est. param.	MSE
559	11-Dec-2021 05:05-09:00	40g	0.05	0.0846	(7.8358) 5.4757
563	23-Oct-2021 08:00-11:00	9g	0.05	0.0175	(2.0603) 0.0438
570	07-Jan-2022 08:00-10:45	105g	0.05	0.1637	(127.1929) 11.0815
575	25-Nov-2021 06:55-10:15	55g	0.05	0.0795	(11.2153) 2.1390
588	10-Sep-2021 07:12-10:15	15g	0.05	0.0225	(3.9346) 0.3897
591	22-Dec-2021 09:20-12:45	36g	0.05	0.0543	(5.6083) 5.5556

Table 7.1: Parameter identification on six individual meals for the Farmer meal model. The parameter to identify was b. The last column describes MSE for the initial parameters (in parenthesis) and after optimization.

Figure 7.1 below shows a comparison between the experimental data and the simulated blood glucose value. It is clear that this meal model was not able to fit to the experimental data very well. An average MSE of 4.1142 after parameter identification strengthens this statement.



(a) Patient 559

(b) Patient 563

(c) Patient 570

(d) Patient 575

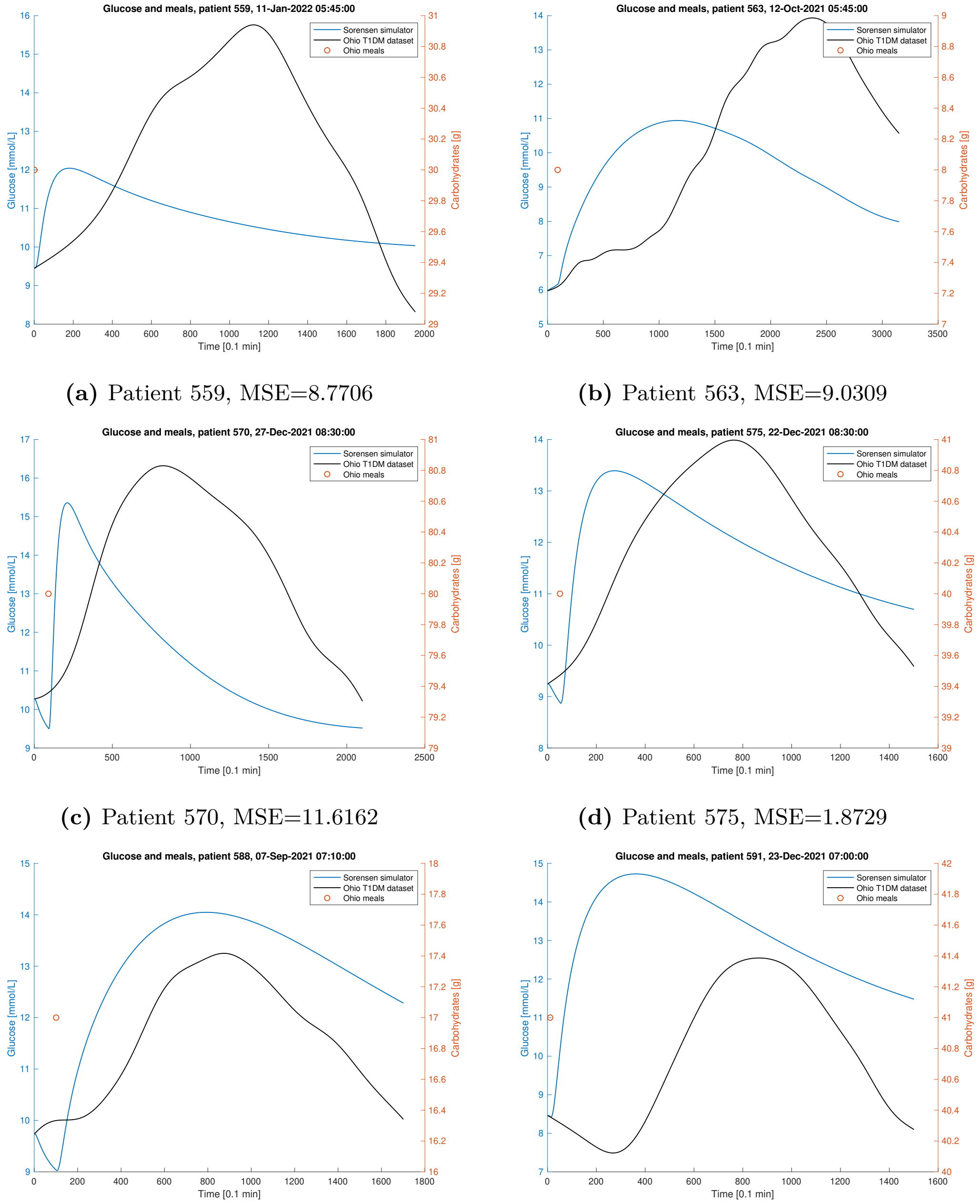
(e) Patient 588

(f) Patient 591

Figure 7.1: Results of Farmer meal model parameter identification. The plots show the simulated glucose in the heart and lungs compartment vs. the experimental data. The orange circles represent when a meal was ingested.

7.1.2 Evaluation

Since this model already had bad results in identification, there were no expectations of a good test data fit. The results are shown below in fig. 7.2. Test gave an average MSE of 7.9232.



(a) Patient 559, MSE=8.7706

(b) Patient 563, MSE=9.0309

(c) Patient 570, MSE=11.6162

(d) Patient 575, MSE=1.8729

(e) Patient 588, MSE=2.0384

(f) Patient 591, MSE=14.2099

Figure 7.2: Evaluation of Farmer model on test data

7.2 Lehmann & Deutsch

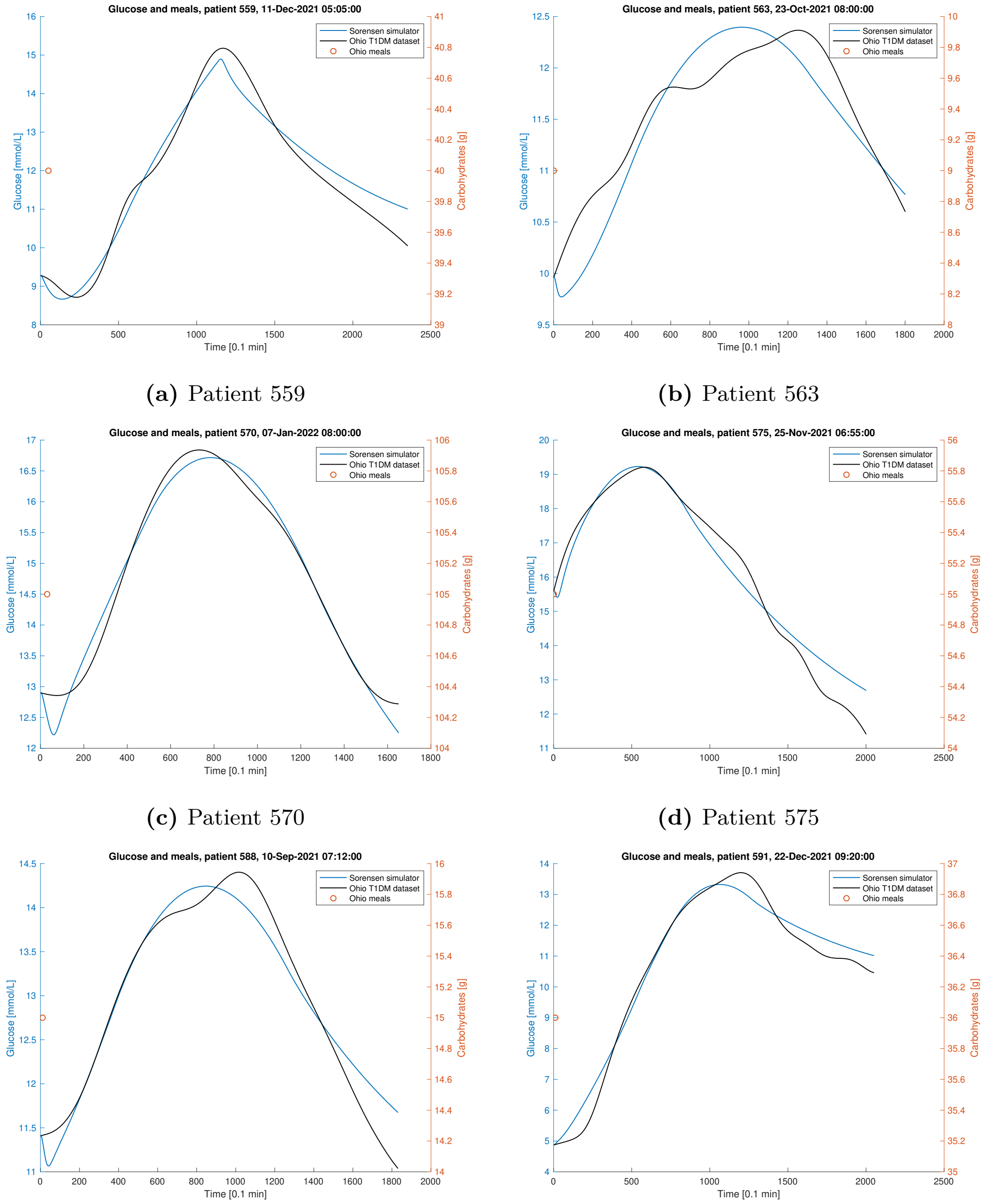
7.2.1 Identification

The parameters that were to be fitted in this meal model were T_{asc}, T_{max}, T_{des} and a from eq. (5.2) and eq. (5.3). The parameter T_{asc} determines how fast the glucose level is increasing, T_{max} determines the amount of minutes for which the glucose level is at its highest, and T_{des} determines how fast the glucose level is decreasing afterwards. a contributes to determining the maximum glucose value. The results are shown in table 7.2 and fig. 7.3.

Patient	**Date**	**CHO**	**Init. params.**	**Est. params.**	**MSE**
559	11-Dec-2021 05:05-09:00	40g	[47, 5, 56, 6]	[49.1638, 2.0219, 59.2644, 6.1919]	(0.1973) 0.1628
563	23-Oct-2021 08:00-11:00	9g	[2, 80, 50, 18]	[2.000, 89.9997, 37.7809, 19.9999]	(0.1819) 0.0944
570	07-Jan-2022 08:00-10:45	105g	[10, 60, 42.9997, 3.2]	[3.1499, 76.3339, 42.6501, 3.4029]	(0.1624) 0.0397
575	25-Nov-2021 06:55-10:15	55g	[5, 70, 5, 5]	[2.0000, 79.1594, 2.0005, 5.2752]	(0.3116) 0.2680
588	10-Sep-2021 07:12-10:15	15g	[5, 80, 30, 15]	[2.0002, 87.4488, 34.337, 16.6438]	(0.1342) 0.0618
591	22-Dec-2021 09:20-12:45	36g	[50, 45, 20, 6]	[51.8804, 49.2228, 22.3572, 7.2913]	(0.2531) 0.1633

Table 7.2: Parameter identification on six individual meals for the Lehmann & Deutsch meal model. The parameter vector corresponds to [T_{asc}, T_{des} T_{max}, a]. The last column describes MSE for the initial parameters (in parenthesis) and after optimization.

The model was able to replicate the glucose dynamics after a meal from the experimental data. For all patients, the shape of the curve is following the Ohio data. The average MSE for the six simulations was 0.1317.



(a) Patient 559

(b) Patient 563

(c) Patient 570

(d) Patient 575

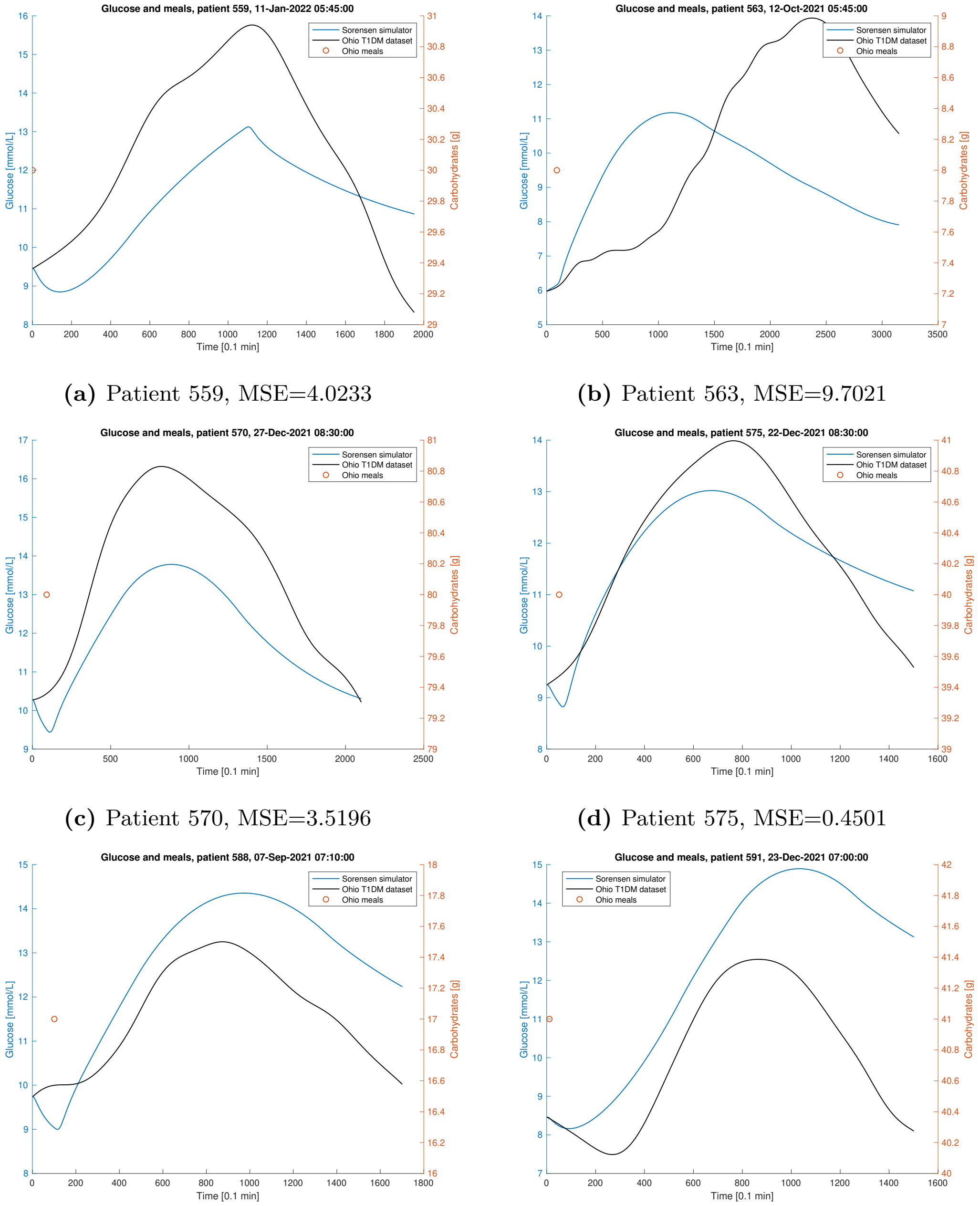
(e) Patient 588

(f) Patient 591

Figure 7.3: Results of Lehmann & Deutsch meal model parameter identification. The plots show the simulated glucose in the heart and lungs compartment vs. the experimental data. The orange circles represent when a meal was ingested.

7.2.2 Evaluation

Evaluation of parameters on the test dataset was performed. The results are shown in fig. 7.4. The resulting average MSE was 4.4120, which is substantially higher than on the training data. But, when looking at the plots, the test results are tolerable, especially in patients 570, 575 and 588.



(a) Patient 559, MSE=4.0233

(b) Patient 563, MSE=9.7021

(c) Patient 570, MSE=3.5196

(d) Patient 575, MSE=0.4501

(e) Patient 588, MSE=1.8259

(f) Patient 591, MSE=6.9508

Figure 7.4: Evaluation of Lehmann & Deutsch meal model on test data.

7.3 UVA/Padova

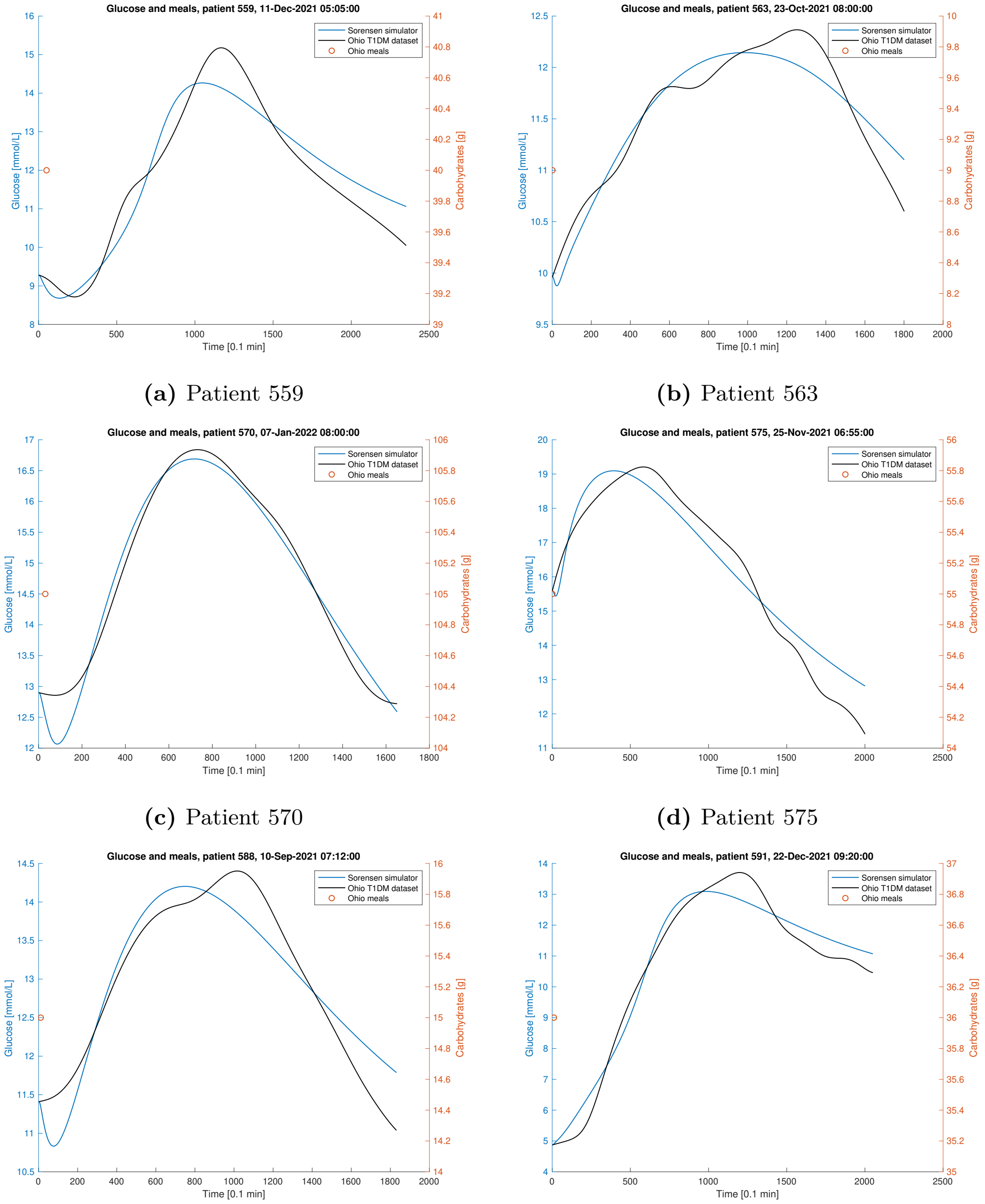
7.3.1 Identification

The parameters that were to be fitted in this model were b, c k_{min}, k_{max} and k_{gri} from eq. (5.12) and eq. (5.17). An additional parameter f was added to control the amplitude as is also done in the original model in the equation describing $Ra(t)$, where the amount of glucose in the gut is multiplied by $\frac{k_{abs} \cdot f}{BW}$. However, since the gut equation was skipped when coupling the meal model with the Sorensen model (stomach equations were coupled with Sorensen's own gut compartment), this amplitude control had to be implemented otherwise. It was instead multiplied with Γ_{meal} in the gut compartment. The weight of the patients was not provided in the dataset, and therefore f had to capture the effects of BW as well. A linear inequality constraint was also included in parameter identification of this meal model to ensure that $k_{min} < k_{max}$. The results of the parameter identification are presented in table 7.3 below.

Patient	**Date**	**CHO**	**Initial params.**	**Optimized params.**	**MSE**
559	11-Dec-2021 05:05-09:00	40g	[0.69, 0.17, 0.006, 0.03, 0.03, 0.25]	[0.7542, 0.0100, 0.0040, 0.0499, 0.0606, 0.1854]	(1.3538) 0.3120
563	23-Oct-2021 08:00-11:00	9g	[0.69, 0.17, 0.006, 0.045, 0.045, 0.6]	[0.0145, 0.2973, 0.0101, 0.9962, 0.0163, 0.6929]	(0.1437) 0.0406
570	07-Jan-2022 08:00-10:45	105g	[0.69, 0.17, 0.006, 0.04, 0.04, 0.1]	[0.4956, 0.2550, 0.0171, 00.0387, 0.0386, 0.1061]	(0.1682) 0.0616
575	25-Nov-2021 06:55-10:15	55g	[0.69, 0.17, 0.006, 0.06, 0.06, 0.17]	[0.5020, 0.2550, 0.0471, 0.4740, 0.0350, 0.1616]	(0.5553) 0.4011
588	10-Sep-2021 07:12-10:15	15g	[0.69, 0.17, 0.006, 0.05, 0.04, 0.48]	[0.7135, 0.1178, 0.0004, 00.0418, 0.0416, 0.5052]	(0.1752) 0.1396
591	22-Dec-2021 09:20-12:45	36g	[0.69, 0.17, 0.006, 0.03, 0.03, 0.25]	[0.7317, 0.0116, 0.0000, 0.0488, 0.0520, 0.2174]	(0.6628) 0.2272

Table 7.3: Parameter identification on six individual meals for the UVA/Padova meal model. The parameter vector corresponds to [b, c k_{min}, k_{max}, k_{gri}, f]. The last column describes MSE for the initial parameters (in parenthesis) and after optimization.

The model was able to capture the dynamics of the experimental data quite well in all patients. The average MSE after parameter identification was performed was 0.1970. Figure 7.5 shows a comparison of simulations with optimized parameters and the experimental data. The simulations are able to follow the Ohio data satisfactory.



(a) Patient 559

(b) Patient 563

(c) Patient 570

(d) Patient 575

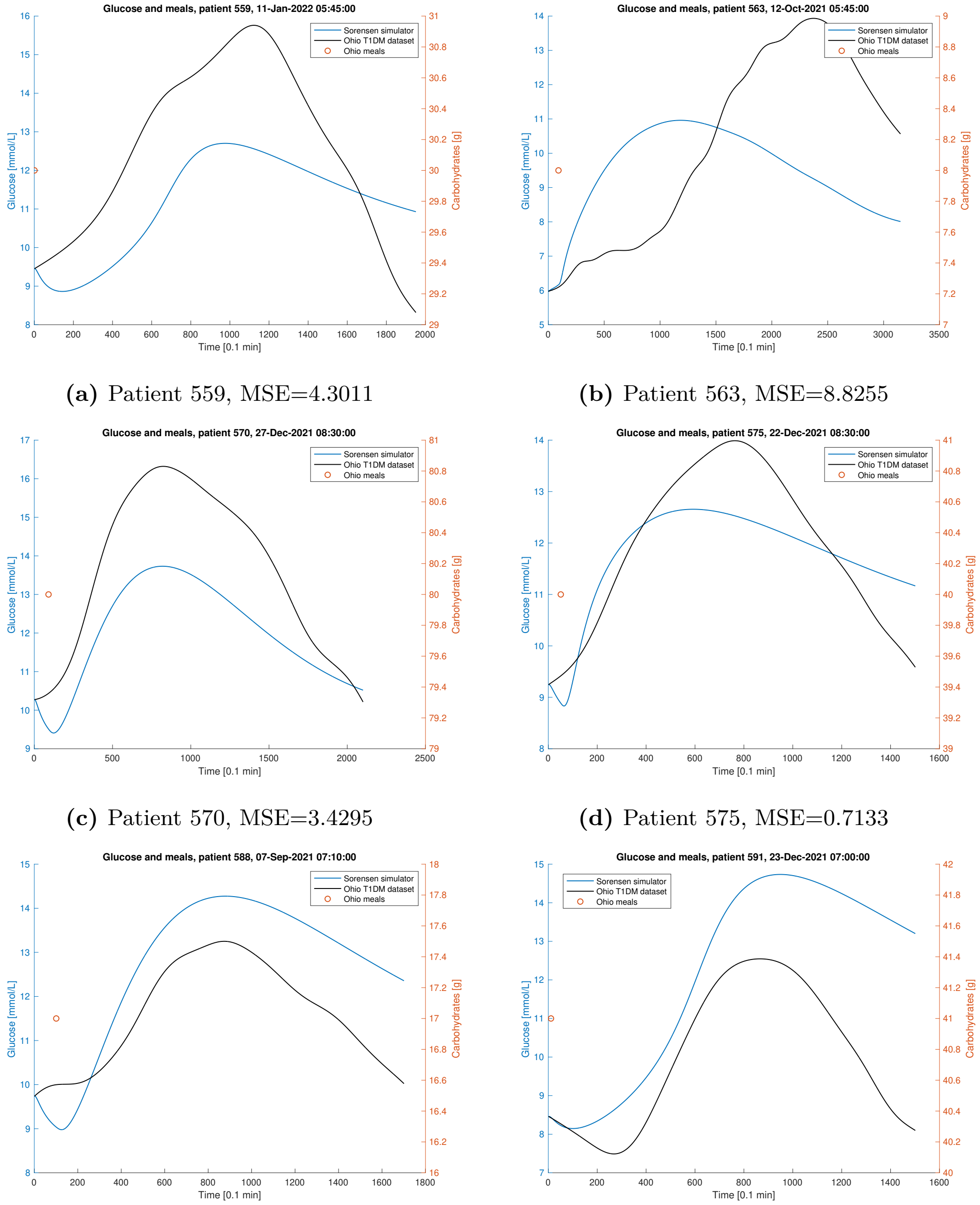
(e) Patient 588

(f) Patient 591

Figure 7.5: Results of UVA/Padova meal model parameter identification. The plots show the simulated glucose in the heart and lungs compartment vs. the experimental data. The orange circles represent when a meal was ingested.

7.3.2 Evaluation

The identified parameters from table 7.3 were evaluated on the test dataset. The result is shown in fig. 7.6 below. Testing gave a mean MSE of 4.2735, which is much higher than the results on the training data which was 0.1970. For all patients except 563, the problem in the test results lie more within the amplitude than in shape. The amplitude is affected by parameter f. For some patients, the glucose curve is increasing either too early or too late. This is especially easy to see in patient 559 where the glucose is increasing too late. The parameters that affect this behavior are k_{min}, k_{max} and k_{gri}. A high value speeds up the increase, while a small value slows it down. However, some of the gap in MSE between the training and test meal is expected due to natural variation between meals.



(a) Patient 559, MSE=4.3011

(b) Patient 563, MSE=8.8255

(c) Patient 570, MSE=3.4295

(d) Patient 575, MSE=0.7133

(e) Patient 588, MSE=1.8260

(f) Patient 591, MSE=6.5456

Figure 7.6: Evaluation of UVA/Padova meal model on test data.

7.4 Summary

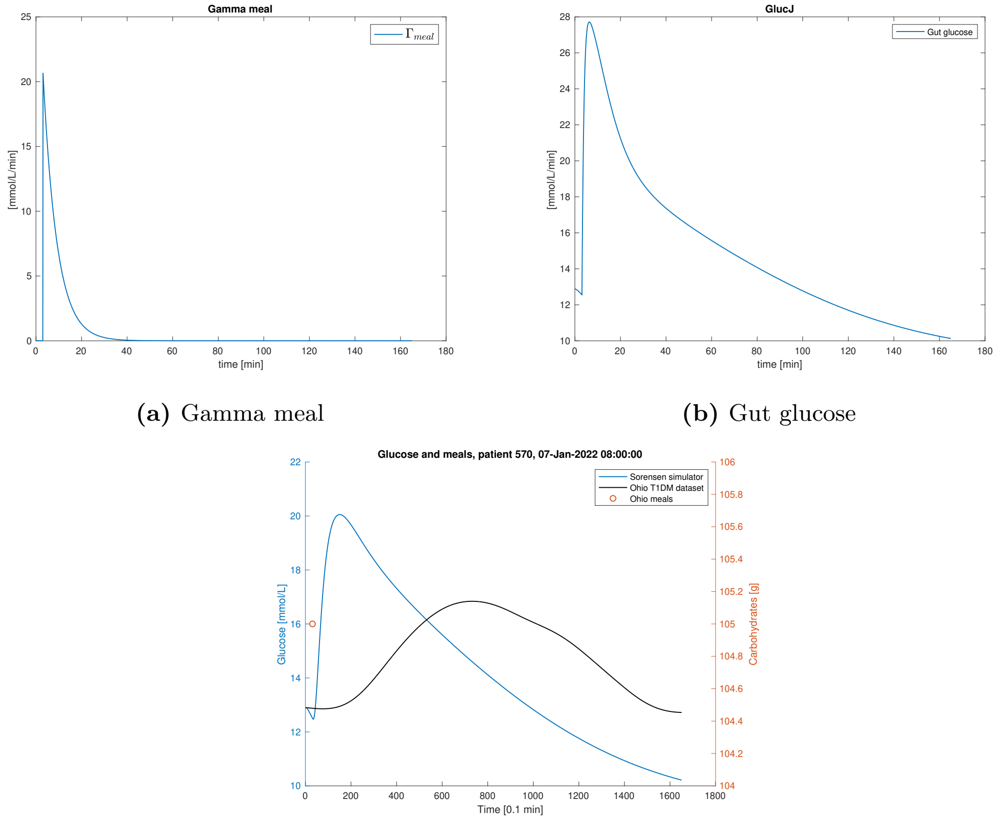
This section has presented the results of the parameter identification and evaluation.

8 Discussion

This thesis has aimed to investigate how the impact of a meal on the blood glucose levels can be mathematically modeled. The first part of this work was to map out existing meal models by performing a literature review. 11 meal models were found and they were compared on the basis of their complexity, structure and physiological accuracy. The details of this discussion and comparison were presented in section 5.3 and will not be repeated here. The second part of this thesis revolved around parameter identification of 3 meal models. The models were assessed to see if they could capture the postprandial glucose dynamics of six patients from an experimental dataset[2]. Following is a discussion of how well the three models performed.

8.1 Farmer

The Farmer model was the most simple meal model and only consisted of an exponential function. The parameter to be fitted was the absorption constant b which determined the decay of the exponential function. This model was not able to fit to the experimental data very well. It had no delay after the carbohydrates were ingested, resulting in a huge increase in blood glucose immediately after. This behavior is depicted in fig. 8.1. This did not resemble the experimental data which had a more delayed type of increase. The decay part of the glucose curve might have been able to fit to the experimental data, but some additional dynamics would have been needed to be added for the whole model to fit well. However, it is important to keep in mind that this model was developed from the Fisher model[31], which again was based on an OGTT rather than a meal. In an OGTT, glucose is ingested in a liquid form and absorption happens much faster since there is no need to digest the liquid. Citing Fisher[31], “the aim is for the model to produce the desired effect of the plasma glucose levels rising quite rapidly to a maximum in less than 30 min and then falling to the base level after about 2-3 hours”. This means that the model will not be able to reproduce what is happening in a mixed meal where absorption can take much longer time than in an OGTT. However, it is a very simple model that is easy to interpret, so if the goal is simplicity and a qualitative representation of a meal, this model might be satisfactory. Nonetheless, in this thesis it fell short in fitting to the six experimental meals. Neither did it give good results in evaluation on the test data, which was expected.



(a) Gamma meal

(b) Gut glucose

(c) Blood glucose

Figure 8.1: Farmer gamma meal, gut and blood glucose. Blood glucose levels rise too rapidly in this model, causing it to be unable to fit to the experimental data.

8.2 Lehmann & Deutsch

The model by Lehmann & Deutsch performed significantly better than the Farmer model. As seen in fig. 7.3, it was able to replicate the curve of the experimental data to a satisfactory extent. For all patients, the simulated meal response was very similar to the experimental data. On average, it gave a MSE of 0.1317 which is much better than the Farmer model. However, the testing gave an average MSE of 4.4120. This higher test result was expected since only one meal was used in parameter identification for each patient. Using a small amount of data allows for overfitting and thus low generalizability. To be able to lower the test error, one would need to train on many more meals. However, due to a limited amount of time, that was not possible for this thesis. Nevertheless, this meal model was able to capture postprandial glucose dynamics and it is likely that with a larger training dataset, it would be able to generalize better as well. It is also important to remember that there are a lot of differences between meals and that this can explain the test error. Even though the MSE from testing was higher than in training, the test results still show that the model is following the shape of the glucose curve after a meal. The error lies mostly within the discrepancy in amplitude, or sometimes how quickly the curve is increasing or decreasing. As opposed to the Farmer model, this meal model was able to recreate the delay between ingestion of food and blood glucose increase. That was captured through the parameter T_{asc} which determines the amount of time it takes for the gastric emptying to reach its maximum. The identified parameters varied somewhat between patients, and it was difficult to find a pattern. However, some similarities were found in parameters between patients. T_{asc} were in 4 of 6 patients very small (around 2-3 minutes), T_{des} was in 5 of 6 patients between 50 and 90 minutes, whereas T_{max} varied between 20 and 60 minutes for all patients except one. These trends resulted in a gastric emptying curve that had a fast increase, but a much slower decrease. Figure 8.2 depicts this. The amplitude a was usually around 5, but had in patients 563 and 588 a value of 16 and 19.

Patient	T_{asc}	T_{des}	T_{max}	a
559	49.1638	2.0219	59.2644	6.1919
563	2.0001	89.9997	37.7809	19.9999
570	3.1499	76.3339	42.6501	3.4029
575	2.0000	79.1594	2.0005	5.2752
588	2.0002	87.4488	34.3377	16.6438
591	51.8804	49.2228	22.3572	7.2913

Table 8.1: Identified parameters, Lehmann & Deutsch

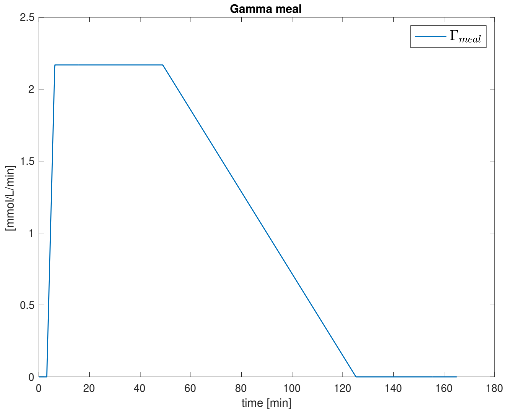
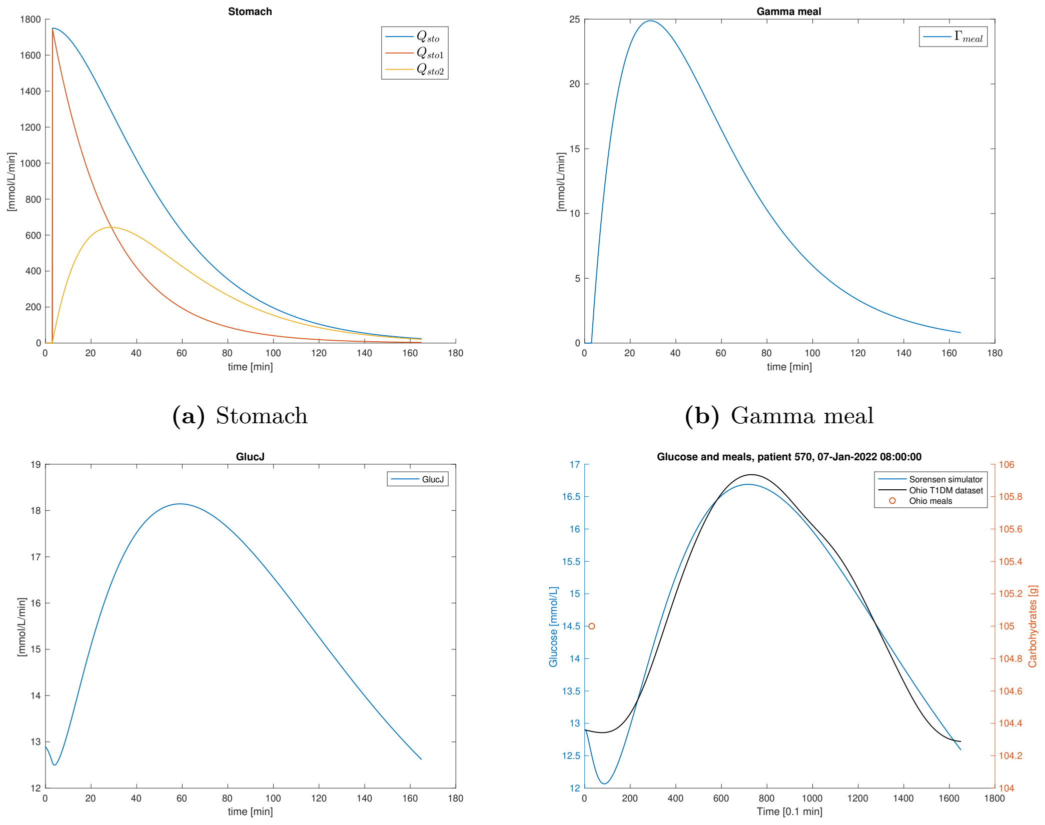


Figure 8.2: Gamma meal, patient 570. Fast increase, slow decrease.

8.3 UVA/Padova

Lastly, the meal model from the UVA/Padova simulator was identified and evaluated. Average training MSE was 0.1970, while average test MSE was 4.2735. The model was able to follow the experimental data after identification which is reflected in the low training error. However, overfitting is a problem in this model as well due to the small training dataset, and the test error is therefore higher than the training error. Nevertheless, the training results were very good and test results were tolerable, so it can be expected that with more data, the model will be able to generalize better. One of the factors that made this model fit so well to the experimental data was probably its two stomach compartments that causes a delay between ingestion of carbohydrates and the increased BG level. The model is therefore able to capture the time it takes for the stomach to grind and digest the food before absorption. A plot of these two stomach compartments are shown in fig. 8.3a. A meal is represented by a delta function which increases the amount of glucose in Q_{sto1} immediately, whereas the glucose in the second compartment Q_{sto2} increases more slowly. Absorption by the gut is given by the rate Γ_{meal} which is shown in fig. 8.3b. Lastly, glucose reaches the heart and lungs compartment.



(a) Stomach

(b) Gamma meal

(c) Gut glucose

(d) Blood glucose

Figure 8.3: UVA/Padova meal model stomach, gut and blood glucose for patient 570.

The parameter values of the UVA/Padova meal model did not vary very much from the original values in Dalla Man, 2006[29]. The original value of b was 0.69, and was identified to be between 0.49 and 0.75 in five of six patients. Parameter c with the original value of 0.17 was identified between 0.1 and 0.3, except in two patients. k_{max} and k_{gri} were around 0.04, which is similar to the original value of 0.05. These two parameters controls the rate at which glucose moves between compartments, and higher values means a faster increase in BG. The only parameter that was different from the original values was k_{min}, which was often estimated to something between 0.01 and 0.04, i.e. a faster rate than the original value of 0.006. The parameter f was created when coupling the meal model with the Sorensen model, and did therefore not exist in the original model. f was in most cases estimated to something between 0.1 and 0.2, but was in patients 563 and 588 much higher (0.5 and 0.69 respectively). This was also the case in the Lehmann & Deutsch model, where the amplitude a was much higher for the same two patients. This suggests that these two meals were somewhat different than the others. They have in common that they are the two smallest meals in terms of carbohydrate (9g and 15g), and maybe therefore needed an extra amplification since the measured BG levels did still increase a lot.

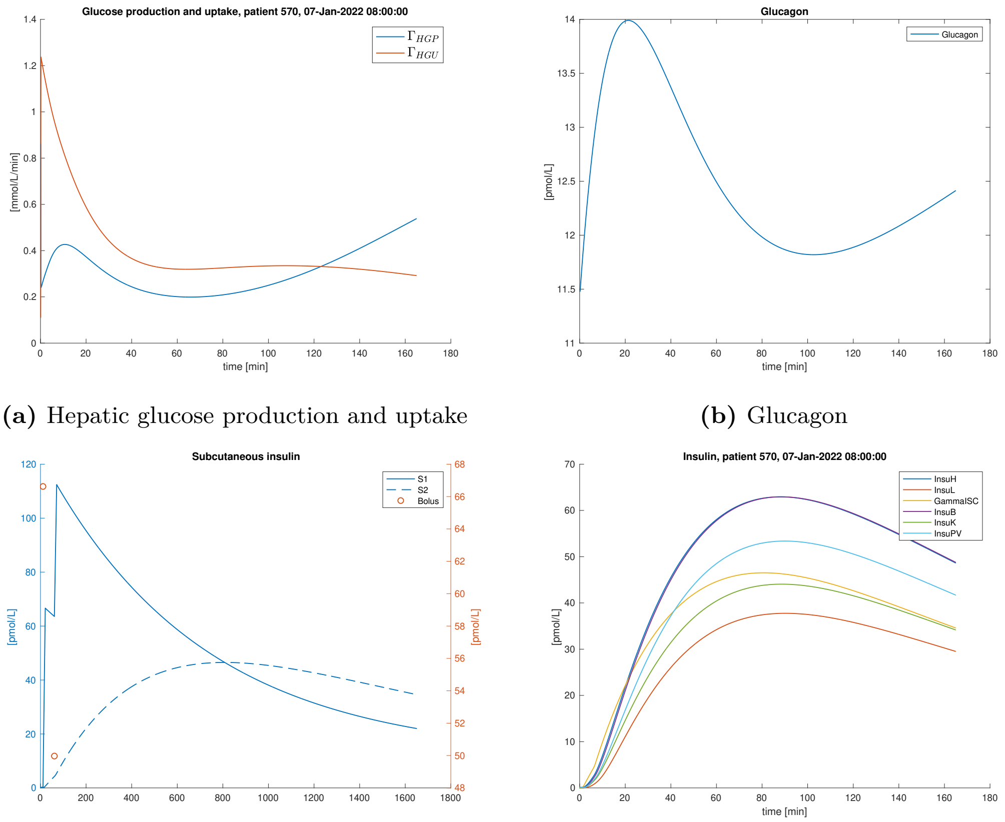
Patient	$\boldsymbol{b}$	$\boldsymbol{c}$	$\boldsymbol{k_{min}}$	$\boldsymbol{k_{max}}$	$\boldsymbol{k_{gri}}$	$\boldsymbol{f}$
559	0.7542	0.0100	0.0400	0.0499	0.0606	0.1854
563	0.0145	0.2973	0.0101	0.9962	0.0163	0.6929
570	0.4956	0.2550	0.0171	0.0387	0.0386	0.1061
575	0.5020	0.2550	0.0471	0.4740	0.0350	0.1616
588	0.7135	0.1178	0.0004	0.0418	0.0416	0.5052
591	0.7317	0.0116	0.0000	0.0488	0.0520	0.2174

Table 8.2: Identified parameters, UVA/Padova

In the original UVA/Padova meal model, the rate of appearance is divided by the body weight BW of the patient. The weight of the patients was not provided by the Ohio T1DM dataset, and could therefore not be included in the simulations. As a result, the parameter f also had to capture this. Differences in body weight might therefore explain some of the differences in the estimated value of f.

8.4 Sorensen simulator

Even though the most important thing to consider when including a meal model to the whole-body model of Sorensen is the glucose level in the heart and lungs compartment (as done in parameter identification), it is important to remember that the Sorensen simulator is complex, and that adding a meal model to it will affect what is happening in all other compartments. A subcutaneous insulin model was also added, which made the insulin dynamics change. Some plots of other relevant compartments and rates are therefore added to increase understanding of what is going on during the simulations. These plots are depicted in fig. 8.4. These figures are from the simulation of a meal from patient 570 (fig. 7.5c) where the peak glucose level occurs at 80 minutes. The production of glucose (Γ_{HGP}) and the glucagon concentration are both decreasing until the glucose level reaches its peak at 80 min, and then slowly increasing again. This makes sense with how the body wants to maintain a stable glucose level. Figure 8.4c shows the subcutaneous insulin model that was added. Two boluses are given in a short amount of time, which makes the first insulin compartment $S1$ increase. There is a delay for when the levels are increasing in the second compartment $S2$. This corresponds with how subcutaneous insulin injections has a delay before in appearing in plasma. The last plot (fig. 8.4d) shows the level of insulin in various compartments. Since the simulated patient is a type 1 diabetic, no insulin are produced endogenously, but depends only on the injected bolus and basal rate.



(a) Hepatic glucose production and uptake

(b) Glucagon

(c) Subcutaneous insulin

(d) Insulin

Figure 8.4: Inside Sorensen simulator during a meal. This is with the UVA/Padova meal model and the meal and insulin inputs from patient 570.

8.5 Challenges

Even though the models were able to fit to the experimental data in most cases, there are still a lot of challenges in identifying the parameters. Most of these challenges originate from the dataset itself and its trustworthiness. All information about meals from this dataset are self-reported by the patient and is therefore subject to errors. The amount of carbohydrate might not be correct, neither the time of the meal. When going through the data, it was seen that sometimes meals were not reported at all (but glucose levels increased, so the meal must have happened), or that meals were reported too early or too late according to the glucose curve (ref. fig. 8.5). When using this type of data in parameter identification, one might end up with parameters trying to compensate for this misreported time of a meal. The parameters will then not represent the true postprandial glucose response. Even though the meals that were chosen for the parameter identification did not seem to suffer from misreported time of announcement, one can still not be 100 % sure. The data is also recorded during "everyday life", meaning that the patients are working, exercising, sleeping etc. during the data collection. In other words, there are factors that might have influenced the data that are not considered by the simulator and can potentially cause errors.

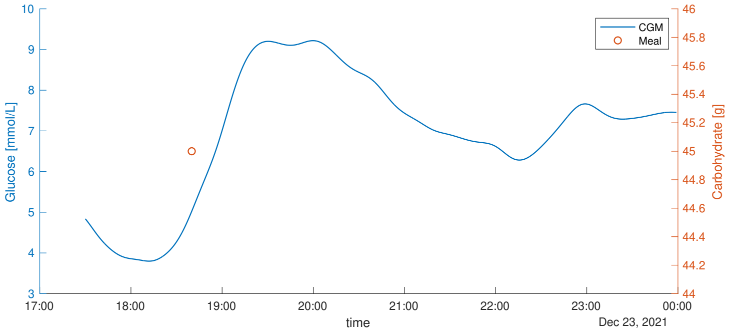


Figure 8.5: Misreported meal from Ohio T1DM dataset. One can clearly see that the curve has been going up for a while before the meal is reported.

Another challenge is that the optimizer is comparing simulated blood glucose values with CGM measurements. A CGM sensor measures the glucose in the interstitial fluid rather than in the blood, and will typically have a delay of 5 minutes. This is because the glucose moves from the blood vessels and capillaries first and then into the interstitial fluid[40]. Ideally, one should have used blood glucose data instead of CGM, or implemented some CGM dynamics to the Sorensen model.

A third challenge is the validity of the subcutaneous insulin model that was added to the Sorensen simulator. It was a simple model with the same rate parameter for both compartments, with a value of $k_a = 0.0144$ resulting in a time-constant of 69.444 minutes. There was no way of identifying this parameter or to check its validity since the Ohio dataset did not contain blood insulin measurements (only insulin infusion). An erroneous insulin rate could potentially cause simulated blood glucose values to decrease too fast or too slow.

The optimizer itself and the choice of cost function is also something that affects the results of the parameter identification. As stated earlier, the optimizer `fmincon` only provides local results and is therefore dependent on the initial parameter values. Time was spent in order to find a good initial point, but there might exist another initial point that would provide a better result. The cost function was implemented as the MSE between the simulated glucose in heart and lungs compartment and the measured glucose data from the dataset. No other type of cost function was tested out.

It is also important to keep in mind that the Sorensen simulator is a “general” glucose simulator and that parameters that affect for instance insulin sensitivity or glucagon response have not been adapted to fit the individual patient. Since intra-patient variability is not captured by the Sorensen simulator, postprandial glucose responses are not completely accurate.

Lastly, the only type of meal considered in identification of parameters in this thesis were breakfasts. Breakfasts are subject to the dawn phenomenon (an increase in BG between 02:00 and 08:00 in the morning), and glucose values might be affected from that. Future work can therefore include identifying parameters on other types of meals (lunch, dinner, snack etc.).

After the parameters were identified, the models were evaluated on a test dataset. This dataset has not been seen by the optimizer, and is therefore completely new data. Since the parameters were identified to fit the training data which only consisted of one meal from one patient, the test error ended up being larger than the training error. This is what is known as overfitting. This happens because there are many differences between individual meals, so that fitting parameters to one meal might not be suitable for another meal. Even though the carbohydrate content was approximately the same in both training and test meals, there are many other differences to consider. Both the content of other macronutrients and the time of ingestion vary between the two meals. For instance, the test meal for patient 563 (fig. 6.6b) is slower than the others (slow increase in glucose), which might be caused by a high fat content. This is not similar to the training meal for patient 563, which has a much faster increase in blood glucose. It is therefore important to give the optimizer several meals such that the parameters capture these differences. Ideally, more data should have been used in identification to obtain more credible and general results. This could have been implemented by a cost function that simulated several meals and then adding up the MSE from each meal, such that one would find one set of optimal parameters common for all these meals. However due to a limited amount of time and the fact that optimizing on just one meal took around 60 minutes, this was never tested out in this thesis.

8.6 Summary

This section has discussed the performance of the three meal models. The Farmer model failed in both identification and evaluation, mostly due to its structure that only allowed for an immediate increase in BG after ingestion which did not fit the experimental data. The Lehmann & Deutsch model and the UVA/Padova model gave much better results and were able to mimic the postprandial glucose response of the experimental data in identification. However, on the test data, the results were not as good. This was caused by the small amount of training data, and the models are believed to perform better with a larger training dataset. Still, it is important to keep in mind that all meals are different and that some variation in test results are to be expected. Even though the test results were not completely satisfactory, this work has nonetheless provided great insights in how a meal model and its structure affects glucose levels.

9 Conclusion

This thesis has investigated the effect meals have on the blood glucose levels and how that can be mathematically modeled. This includes theory about the physiology of the gastrointestinal system and diabetes. A literature review was performed, and 11 meal models were found. The literature review showed that

- The most common way to quantify a meal was through the amount of carbohydrates (although some models included more information such as amount of fat, proteins or type of carbohydrate)
- The two most common structures were to use one gut compartment, or one stomach and one gut compartment
- More complex information such as order of ingestion or other personalized factors were not included in any of the models

Three of the meal models from the literature review were then used in parameter identification and evaluation. Results showed that

- The UVA/Padova and Lehmann & Deutsch models performed much better than the Farmer model
- Models with a delayed increase in glucose fit better to the experimental data. This was probably due to the fact that the stomach needs time to grind stomach contents before absorption is possible
- Due to a small training dataset, overfitting was an issue

Future work could be to fix some of the challenges that was mentioned in the discussion in section 8. Most important would be to use more data in parameter identification in order to improve test results. Another improvement could be to use a “better” dataset that contained blood glucose measurements instead of CGM, blood insulin measurements and more trustworthy meal information regarding carbohydrate content and time of ingestion. Lastly, more time can be spent on tuning initial values before running the optimizer in order to find a better local minimum.

A Sorensen model

The following section is adapted from the revised Sorensen model[38].

Parameters and variables
G – glucose concentration [mmol/L]
I – insulin concentration [pmol/L]
χ – glucagon concentration [pmol/L]
V – volume [L]
Q – vascular blood water flow rate [L/min]
Γ – metabolic source or sink rate [mmol/min or pmol/min]
T – trans-capillary diffusion rate [min]
t – time [min]

Subscripts
First subscript: which compartment (B – brain, H – heart/lungs, G – gut, L – liver, K – kidney, P – periphery, A - artery)
Second subscript: interstitial fluid space (I) or vascular blood water space (V)

Superscripts
G – glucose
I - insulin
χ - glucagon
B - basal value
N - normalized value (divided by basal value)

Metabolic sources or sinks
Γ_{PGU} - peripheral glucose uptake, eq. (A.17)
Γ_{RBCU} - red blood cell glucose uptake [10 mg/min]
Γ_{PIR} - peripheral insulin release [0 mg/min] for a type 1 diabetic
Γ_{LIC} - liver insulin clearance, eq. (A.33)
Γ_{KIC} - kidney insulin clearance, eq. (A.35)
Γ_{PIC} - peripheral insulin clearance, eq. (A.36)
Γ_{KGE} - kidney glucose excretion, eq. (A.31)
Γ_{HGU} - hepatic glucose uptake, eq. (A.27)
Γ_{HGP} - hepatic glucose production, eq. (A.20)
Γ_{GGU} - gut glucose utilization [20 mg/min]
Γ_{BGU} - brain glucose uptake [70 mg/min]
$\Gamma_{P_\chi C}$ - plasma glucagon clearance, eq. (A.37)
$\Gamma_{M_\chi C}$ - metabolic glucagon clearance [0.91 L/min]
$\Gamma_{P_\chi R}$ - pancreatic glucagon release, eq. (A.38)

Equations

Brain compartment

$$\frac{dG_{BV}(t)}{dt} = \frac{Q_B^G}{V_{BV}^G}(G_H(t) - G_{BV}(t)) - \frac{V_{BI}}{V_{BV}^G T_B^G}(G_{BV}(t) - G_{BI}(t)) \tag{A.1}$$

$$\frac{dG_{BI}(t)}{dt} = \frac{1}{T_B^G}(G_{BV}(t) - G_{BI}(t)) - \frac{\Gamma_{BGU}}{V_{BI}} \tag{A.2}$$

$$\frac{dI_B(t)}{dt} = \frac{Q_B^I}{V_B^I}(I_H(t) - I_B(t)) \tag{A.3}$$

Heart/lungs compartment

$$\frac{dG_H(t)}{dt} = \frac{1}{V_H^G}(Q_B^G G_{BV}(t) + Q_L^G G_L(t) + Q_K^G G_K(t) + Q_P^G G_{PV}(t) - Q_H^G G_H(t) - \Gamma_{RBCU}) \tag{A.4}$$

$$\frac{dI_H(t)}{dt} = \frac{1}{V_H^I}(Q_B^I I_B(t) + Q_L^I I_L(t) + Q_K^I I_K(t) + Q_P^I I_{PV}(t) - Q_H^I I_H(t) + i(t)) \tag{A.5}$$

Gut compartment

$$\frac{dG_G(t)}{dt} = \frac{Q_G^G}{V_G^G}(G_H(t) - G_G(t)) + \frac{1}{V_G^G}(\Gamma_{meal} - \Gamma_{GGU}) \tag{A.6}$$

$$\frac{dI_G(t)}{dt} = \frac{Q_G^I}{V_G^I}(I_H(t) - I_G(t)) \tag{A.7}$$

Liver compartment

$$\frac{dG_L(t)}{dt} = \frac{1}{V_L^G}(Q_A^G G_H(t) + Q_G^G G_G(t) - Q_L^G G_L(t) + \Gamma_{HGP} - \Gamma_{HGU}) \tag{A.8}$$

$$\frac{dI_L(t)}{dt} = \frac{1}{V_L^I}(Q_A^I I_H(t) + Q_G^I I_G(t) - Q_L^I I_L(t) + \Gamma_{PIR} - \Gamma_{LIC}) \tag{A.9}$$

Kidney compartment

$$\frac{dG_K(t)}{dt} = \frac{Q_K^G}{V_K^G}(G_H(t) - G_K(t)) - \frac{1}{V_K^G}\Gamma_{KGE} \tag{A.10}$$

$$\frac{dI_K(t)}{dt} = \frac{Q_K^I}{V_K^I}(I_H(t) - I_K(t)) - \frac{1}{V_K^I}\Gamma_{KIC} \tag{A.11}$$

Periphery compartment

$$\frac{dG_{PV}(t)}{dt} = \frac{Q_P^G}{V_{PV}^G}(G_H(t) - G_K(t)) - \frac{V_{PI}}{V_{PV}^G T_P^G}(G_{PV}(t) - G_{PI}(t)) \tag{A.12}$$

$$\frac{dG_{PI}(t)}{dt} = \frac{1}{T_P^G}(G_{PV}(t) - G_{PI}(t)) - \frac{\Gamma_{PGU}}{V_{PI}^G} \tag{A.13}$$

$$\frac{dI_{PV}(t)}{dt} = \frac{Q_P^I}{V_{PV}^I}(I_H(t) - I_{PV}(t)) - \frac{V_{PI}}{V_{PV}^I T_P^I}(I_{PV}(t) - I_{PI}(t)) \tag{A.14}$$

$$\frac{dI_{PI}(t)}{dt} = \frac{1}{T_P^I}(I_{PV}(t) - I_{PI}(t)) - \frac{\Gamma_{PIC}}{V_{PI}^I} \tag{A.15}$$

Glucagon system

$$\frac{d\chi^N(t)}{dt} = \frac{1}{V^\chi}(\Gamma_{M_\chi C} \cdot \Gamma_{P_\chi R}^N - \Gamma_{M_\chi C} \cdot \chi^N(t)) \tag{A.16}$$

Metabolic sinks and sources equations

$$\Gamma_{PGU} = M^I_{PGU} \cdot M^G_{PGU} \cdot 35 \tag{A.17}$$

$$M^I_{PGU} = 7.03 + 6.52 \cdot \tanh\left[0.338(I^N_{PI}(t) - 5.82)\right] \tag{A.18}$$

$$M^G_{PGU} = G^N_{PI}(t) \tag{A.19}$$

$$\Gamma_{HGP} = M^I_{HGP} \cdot M^{\chi}_{HGP} \cdot M^G_{HGP} \cdot 155 \tag{A.20}$$

$$\frac{dM^I_{HGP}}{dt} = \frac{1}{\tau_1}(M^{I\infty}_{HGP} - M^I_{HGP}) \tag{A.21}$$

$$M^{I\infty}_{HGP} = 1.2793 - 1.0647 \cdot \tanh\left[1.733(I^N_L(t) - 0.849)\right] \tag{A.22}$$

$$M^{\chi}_{HGP} = M^{\chi 0}_{HGP} - f_2 \tag{A.23}$$

$$M^{\chi 0}_{HGP} = 2.7 \cdot \tanh\left[0.39\chi^N(t)\right] \tag{A.24}$$

$$\frac{df_2}{dt} = \frac{1}{\tau_{\chi}}(\frac{M^{\chi 0}_{HGP} - 1}{2} - f_2) \tag{A.25}$$

$$M^G_{HGP} = 1.42 - 1.41 \cdot \tanh\left[0.62(G^N_L(t) - 0.497)\right] \tag{A.26}$$

$$\Gamma_{HGU} = M^I_{HGU} \cdot M^G_{HGU} \cdot 20 \tag{A.27}$$

$$\frac{dM^I_{HGU}}{dt} = \frac{1}{\tau_1}(M^{I\infty}_{HGU} - M^I_{HGU}) \tag{A.28}$$

$$M^{I\infty}_{HGU} = 2.0 \cdot \tanh\left[0.55 I^N_L(t)\right] \tag{A.29}$$

$$M^G_{HGU} = 5.66 + 5.66 \cdot \tanh\left[2.44(G^N_L(t) - 1.48)\right] \tag{A.30}$$

$$\Gamma_{KGE} = 71 + 71 \cdot \tanh\left[0.11 G_K(t) - 460\right] \quad 0 < G_K(t) < 460 mg/dL \tag{A.31}$$

$$\Gamma_{KGE} = -330 + 0.872 G_K(t) \quad G_K(t) \geq 460 \tag{A.32}$$

$$\Gamma_{LIC} = f_{LIC} \cdot (Q^I_H I_H(t) + Q^I_G I_G(t) + \Gamma_{PIR}) \tag{A.33}$$

$$\Gamma_{PIR} = 0 \quad \text{(for T1D)} \tag{A.34}$$

$$\Gamma_{KIC} = f_{KIC} \cdot Q^I_K I_H(t) \tag{A.35}$$

$$\Gamma_{PIC} = \frac{I_{PI}(t)}{(\frac{1-f_{PIC}}{f_{PIC}})(\frac{1}{Q^I_P}) - \frac{T^I_P}{V_{PI}}} \tag{A.36}$$

$$\Gamma_{P_{\chi}C}(t) = \Gamma_{M_{\chi}C}(t) \cdot \chi^N(t) \tag{A.37}$$

$$\Gamma^N_{P_{\chi}R} = M^I_{P_{\chi}R} \cdot M^G_{P_{\chi}R} \tag{A.38}$$

$$M^G_{P_{\chi}R} = 2.93 - 2.10 \cdot \tanh\left[4.18(G^N_H(t) - 0.61)\right] \tag{A.39}$$

$$M^I_{P_{\chi}R} = 1.31 - 0.61 \cdot \tanh\left[1.06(I^N_H(t) - 0.47\right] \tag{A.40}$$

Parameter values and units

Parameter	**Value**	**Unit**
τ_1	25	min
τ_χ	65	min
V_{BV}^G	0.35	L
V_{BI}^G	0.45	L
V_H^G	1.38	L
V_L^G	2.51	L
V_G^G	1.12	L
V_K^G	0.66	L
V_{PV}^G	1.04	L
V_{PI}^G	6.74	L
Q_B^G	0.59	L/min
Q_H^G	4.37	L/min
Q_A^G	0.25	L/min
Q_L^G	1.26	L/min
Q_G^G	1.01	L/min
Q_K^G	1.01	L/min
Q_P^G	1.51	L/min
T_B^G	2.1	min
T_P^G	5.0	min
f_{LIC}	0.4	dimensionless
f_{KIC}	0.3	dimensionless
f_{PIC}	0.15	dimensionless
V_B^I	0.26	L
V_H^I	0.99	L
V_G^I	0.94	L
V_L^I	1.14	L
V_K^I	0.51	L
V_{PV}^I	0.74	L
V_{PI}^I	6.74	L
Q_B^I	0.45	L/min
Q_H^I	3.12	L/min
Q_A^I	0.18	L/min
Q_K^I	0.72	L/min
Q_P^I	1.05	L/min
Q_G^I	0.72	L/min
Q_L^I	0.9	L/min
T_P^I	20.0	min
V^χ	11.31	L

Table A.1: Parameter values Sorensen model.

B Meal models added to Sorensen model

All 3 meal models were coupled with the Sorensen model through the Γ_{meal} rate in the gut compartment.

Farmer model

$$\Gamma_{meal} = CHO \cdot a \cdot \exp\left(-b \cdot (t - t_{meal})\right) \tag{B.1}$$

$$\frac{dG_G(t)}{dt} = \frac{Q_G^G}{V_G^G}(G_H(t) - G_G(t)) + \frac{1}{V_G^G}(\Gamma_{meal} - \Gamma_{GGU}) \tag{B.2}$$

Lehmann & Deutsch model

$$G_{empt} = \begin{cases} (V_{max}/T_{asc})t & \text{if } t < T_{asc} \\ V_{max} & \text{if } T_{asc} \le t < T_{asc} + T_{max} \\ V_{max} - (V_{max}/T_{des})(t - T_{asc} - T_{max}) & \text{if } T_{asc} + T_{max} \le t < T_{asc} + T_{max} + T_{des} \\ 0 & \text{elsewhere} \end{cases} \tag{B.3}$$

$$V_{max} = \frac{a \cdot CHO}{T_{asc} + 2 \cdot T_{max} + T_{des}} \tag{B.4}$$

$$\Gamma_{meal} = G_{empt}(t) \tag{B.5}$$

$$\frac{dG_G(t)}{dt} = \frac{Q_G^G}{V_G^G}(G_H(t) - G_G(t)) + \frac{1}{V_G^G}(\Gamma_{meal} - \Gamma_{GGU}) \tag{B.6}$$

UVA/Padova meal model

$$Q_{sto}(t) = Q_{sto1}(t) + Q_{sto2}(t) \tag{B.7}$$

$$\dot{Q}_{sto1}(t) = -k_{gri} \cdot Q_{sto1} + D \cdot \delta(t) \tag{B.8}$$

$$\dot{Q}_{sto2}(t) = -k_{empt}(Q_{sto}) \cdot Q_{sto2}(t) + k_{gri} \cdot Q_{sto1}(t) \tag{B.9}$$

$$\Gamma_{meal} = f \cdot k_{empt}(Q_{sto}(t)) \cdot Q_{sto2} \tag{B.10}$$

$$\frac{dG_G(t)}{dt} = \frac{Q_G^G}{V_G^G}(G_H(t) - G_G(t)) + \frac{1}{V_G^G}(\Gamma_{meal} - \Gamma_{GGU}) \tag{B.11}$$

C Code - simulator and identification

This section presents the MATLAB code for the Sorensen simulator which is coupled with the UVA/Padova meal model. The simulator is the revised Sorensen simulator[38], but has been modified to include the equations for the meal models and inputs from the Ohio T1DM dataset. The code for the cost function and parameter identification is included as well. Some files from the Sorensen simulator is not included here due to its size, but the most relevant files in order to understand the setup are included. Code for the two other meal models (Lehmann & Deutsch and Farmer) is not included since it is very similar to the UVA/Padova code. The only changes are the meal equations and parameters, but they are added to the simulator in the exact same way.

CreateInputArraysOhio.m

```
global date_ patient_nr data_one_day time_one_day ...
    meal_one_day cgm_one_day bolus_one_day basal_one_day CHO_0

patient_nr = '570';

% Read all data files to workspace
% data_559 = readtable('559/interpolated/interpolated_data_with_kalman_559.xlsx');
% data_563 = readtable('563/interpolated/interpolated_data_with_kalman_563.xlsx');
% data_588 = readtable('588/interpolated/interpolated_data_with_kalman_588.xlsx');
% data_591 = readtable('591/interpolated/interpolated_data_with_kalman_591.xlsx');
% data_570 = readtable('570/interpolated/interpolated_data_with_kalman_570.xlsx');
% data_575 = readtable('575/interpolated/interpolated_data_with_kalman_575.xlsx');

switch patient_nr
    case '559'
        data = evalin('base','data_559');
        data = removevars(data,{'total'});
        t_start = {'11-Dec-2021 05:05:00'};
        t_end = {'11-Dec-2021 09:00:00'};
        CHO_0 = 40000; % mg carbohydrate
    case '563'
        data = evalin('base','data_563');
        t_start = {'23-Oct-2021 08:00:00'};
        t_end = {'23-Oct-2021 11:00:00'};
        CHO_0 = 9000;
    case '570'
        data = evalin('base','data_570');
        t_start = {'07-Jan-2022 08:00:00'};
        t_end = {'07-Jan-2022 10:45:00'};
        CHO_0 = 105000;
    case '575'
```

```
        data = evalin('base','data_575');
        t_start = {'25-Nov-2021 06:55:00'};
        t_end = {'25-Nov-2021 10:15:00'};
        CHO_0 = 55000;
    case '588'
        data = evalin('base','data_588');
        t_start = {'10-Sep-2021 07:12:00'};
        t_end = {'10-Sep-2021 10:15:00'};
        CHO_0 = 15000;
    case '591'
        data = evalin('base','data_588');
        t_start = {'22-Dec-2021 09:20:00'};
        t_end = {'22-Dec-2021 12:45:00'};
        CHO_0 = 36000;
end

% Extract data from chosen time period and patient
date_ = datestr(t_start);
i1 = find(data.time == t_start);
i2 = find(data.time == t_end);

data_one_day = data(i1:i2, [1,2,4,5,6,7]);
time_one_day = data_one_day.time;
meal_one_day = data_one_day.carbs;
cgm_one_day = data_one_day.cgm_mmol_kalman;
bolus_one_day = data_one_day.bolus_pmol;
basal_one_day = data_one_day.basal_pmol;

% Find times where meal is ingested, bolus is injected and basal change happens
global meal_index_times bolus_index_times basal_index_times

meal_index_times = find(data_one_day.carbs ~= 0);
bolus_index_times = find(data_one_day.bolus_pmol ~= 0);

basal_index_times = [];
prev_val = data_one_day.basal_pmol(1);

for i=1:length(data_one_day.basal_pmol)
    basal_val = data_one_day.basal_pmol(i);
    if basal_val ~= prev_val
        basal_index_times = [basal_index_times i];
        prev_val = basal_val;
    end
end

basal_index_times = transpose(basal_index_times);
```

CostFunction.m

```
function error = CostFunction(params)

global b_p0 c_p0 k_min_p0 k_max_p0 k_gri_p0 ...
    Tzero_ Tend_ Insu_init Gluc_init f
tic
% Parameters to identify
b_p0 = params(1);
c_p0 = params(2);
k_min_p0 = params(3);
k_max_p0 = params(4);
k_gri_p0 = params(5);
f = params(6);

% Make arrays for insulin and meal input
createInputArraysOhio;

% Set start and end time according to Ohio data time
Tzero_ = 0;
Tend_ = minutes(data_one_day.time(end) - data_one_day.time(1));

% Initial values for glucose and insulin
Insu_init = basal_one_day(1);
Gluc_init = cgm_one_day(1);

% Simulate Sorensen model
SorensenAutoTester;

% Extract glucose data after simulation
glucose_sim = STATEVARS(:,3);
glucose_ohio = data_one_day.cgm_mmol_kalman;
minLength = min(length(glucose_sim), length(glucose_ohio));
glucose_sim = glucose_sim(1:minLength);
glucose_ohio = glucose_ohio(1:minLength);

% Calculate MSE
error = immse(glucose_ohio, glucose_sim);
toc
end
```

optimizeParameters.m

```
% Initial point
p0 = [0.69, 0.17, 0.006, 0.03, 0.03, 0.25];

% Lower and upper bounds
lb = [0.01, 0.01,  0,  0,  0, 0.01];
ub = [0.99,  0.5,  0.1,  1,  1, 1];

% Inequality constraint to ensure k_min < k_max
A = [0, 0, 1, -1, 0, 0];
b = 0;

N = length(p0);
options = optimoptions('fmincon','Display','iter','MaxFunctionEvaluations', ...
      1000*N,'MaxIterations',150);
params_opt = fmincon(@CostFunction,p0,A,b,[],[],lb,ub,[],options);
```

updateParvalsFromOhio.m

```
% Change values for meals and insulin infusion in Sorensen simulator at correct time

% update meal data
for i=1:length(meal_index_times)
    if Time == floor(meal_index_times(i)/10)
       Time_meal = floor(meal_index_times(i)/10);
       % multiply by 1000 to get mg instead of g
       CHO = data_one_day.carbs(meal_index_times(i))*1000;
    end
end

% update bolus data
for i=1:length(bolus_index_times)
    % insulin infusion = bolus + basal
    if Time == floor(bolus_index_times(i)/10)
        insulin_inf = data_one_day.bolus_pmol(bolus_index_times(i)) + ...
            data_one_day.basal_pmol(bolus_index_times(i));
    end
    % must be set to basal rate again at next step since a bolus is
    % delivered all at once
    if Time == floor(bolus_index_times(i)/10)+1
        insulin_inf = data_one_day.basal_pmol(bolus_index_times(i)+1);
    end
end

% update basal data
for i=1:length(basal_index_times)
    if Time == floor(basal_index_times(i)/10)
        insulin_inf = data_one_day.basal_pmol(basal_index_times(i));
    end
end
```

SorensenAutoTester.m

```
%==========================================================================
% SCRIPT SorensenAutoTester.m: automatic pure-Matlab (no C++) Tester
% Sorensen V01.01.41 20190724 (Gemini 13.01.06, BMLib 10.0.2
% Autocoder 02.11.09, coded 24-Jul-2019 16:06:54)
%==========================================================================

% calls DetermineParameters and produces the initialization values for
% the variables (parameters initialized BEFORE variables)
SorensenInitializeParvals;
SorensenInitializeStateVars;
TIME = Tzero:Tdelta:Tend;
nTimes = length(TIME);
STATEVARS = zeros(nTimes,nDepVars);
STATEVARS(1,:) = CurrentY;

% 20190709 Andrea De Gaetano, Marcello Pompa: Runge Kutta 4
% NOTE: this MATLAB implementation of RK4 uses vector assignments and therefore
% does NOT replicate exactly the corresponding C++ RK routine. By vector-assigning
% the increments to the (differentially-defined) Ys, the intermediate and final
% computation of the derivatives of one Y depend on the intermediate and final
% values of the other Ys. Conversely, when using loops, the intermediate and
% final computation of the derivatives of one Y depend on the interval INITIAL
% values of the other Ys, same as in C++. This last arrangement is consistent with
% the fact that the algebraic variables too are updated only after the computation
% of all the differential variables and that therefore the algebraic variable values
% used during all of the intermediate computations are the interval initial values.
% For smooth functions the differences in the implementations ought to be minimal.
% In case of differences, the C++ version is the logically consistent one.
% In any case, for non-smooth functions a variable-step integrator such as
% Runge-Kutta-Fehlberg ought to be used.

fprintf(1,’\ n.’);
Time = TIME(1);
for (tk=1:(nTimes-1))
   CurrentY0 = CurrentY;

   % update meal and insulin inputs
   updateParvalsFromOhio;

   SorensenCurrentY2NamedVars;
   SorensenComputeDerivatives;
   dY1 = out1; % RK constant 1
   Time = TIME(tk) + Tdelta/2;
   CurrentY = CurrentY0 + Tdelta/2 * dY1;
   SorensenCurrentY2NamedVars;
   SorensenComputeDerivatives;
   dY2 = out1; % RK constant 2

   CurrentY = CurrentY0 + Tdelta/2 * dY2;
   SorensenCurrentY2NamedVars;
   SorensenComputeDerivatives;
   dY3 = out1; % RK constant 3

   Time = TIME(tk+1);
   CurrentY = CurrentY0 + Tdelta * dY3;
   SorensenCurrentY2NamedVars;
   SorensenComputeDerivatives;
   dY4 = out1; % RK constant 4

   CurrentY = CurrentY0 + Tdelta * (dY1/6 + dY2/3 + dY3/3 + dY4/6);
   SorensenForceVars;
   SorensenCurrentY2NamedVars; % needed to do the next computations
   SorensenComputeDiracs;
   SorensenNamedVars2CurrentY; % rebuild CurrentY
   SorensenForceVars;
   SorensenCurrentY2NamedVars; % needed to do the next computations
   SorensenComputeAlgebraic;
   SorensenNamedVars2CurrentY; % rebuild CurrentY
   SorensenForceVars;
   STATEVARS(tk+1,:) = CurrentY; % add state values for each timestep to STATEVARS matrix
   %fprintf(1,’.’); if (mod(tk,50)==0) fprintf(1,’\ n’); end  % show progress
end

SorensenLoadNames;
```

SorensenInitializeParvals.m

```
% ====================================================================================
% SCRIPT SorensenInitializeParvals.m: assignment of initial values to named parameters
% Sorensen V01.01.41 20190724 (Gemini 13.01.06, BMLib 10.0.2
Autocoder 02.11.09, coded 24-Jul-2019 16:06:54)
% ====================================================================================

global PARMIN PARMAX PARDETM;

nPars = 152;
```

```
PARMIN = zeros(152,1); % max and min value of a certain parameter
PARMAX = zeros(152,1);
PARDETM = zeros(152,1); % PARDETM = 0 if not determined in DetermineParams.m
PARMIN(001) = -30; PARMAX(001) = 1440; PARDETM(001)=0;
PARMIN(002) = 0; PARMAX(002) = 10080; PARDETM(002)=0;
PARMIN(003) = 0; PARMAX(003) = 600; PARDETM(003)=0;
PARMIN(004) = 0; PARMAX(004) = 10; PARDETM(004)=0;
PARMIN(005) = 0; PARMAX(005) = 10; PARDETM(005)=0;
PARMIN(006) = 0; PARMAX(006) = 10; PARDETM(006)=0;
PARMIN(007) = 0.01; PARMAX(007) = 50; PARDETM(007)=0;
PARMIN(008) = 0.1; PARMAX(008) = 20; PARDETM(008)=0;
PARMIN(009) = 0; PARMAX(009) = 10; PARDETM(009)=0;
PARMIN(010) = 0; PARMAX(010) = 10; PARDETM(010)=0;
PARMIN(011) = 0; PARMAX(011) = 10; PARDETM(011)=0;
PARMIN(012) = 0; PARMAX(012) = 10; PARDETM(012)=0;
PARMIN(013) = 0; PARMAX(013) = 10; PARDETM(013)=0;
PARMIN(014) = 0; PARMAX(014) = 10; PARDETM(014)=0;
PARMIN(015) = 0; PARMAX(015) = 10; PARDETM(015)=0;
PARMIN(016) = 0; PARMAX(016) = 10; PARDETM(016)=0;
PARMIN(017) = 0; PARMAX(017) = 10; PARDETM(017)=0;
PARMIN(018) = 0; PARMAX(018) = 10; PARDETM(018)=0;
PARMIN(019) = 0; PARMAX(019) = 10; PARDETM(019)=0;
PARMIN(020) = 0; PARMAX(020) = 10; PARDETM(020)=0;
PARMIN(021) = 0; PARMAX(021) = 10; PARDETM(021)=0;
PARMIN(022) = 0; PARMAX(022) = 10; PARDETM(022)=0;
PARMIN(023) = 0; PARMAX(023) = 10; PARDETM(023)=0;
PARMIN(024) = 0.01; PARMAX(024) = 50; PARDETM(024)=0;
PARMIN(025) = 0; PARMAX(025) = 10; PARDETM(025)=0;
PARMIN(026) = 0; PARMAX(026) = 100; PARDETM(026)=0;
PARMIN(027) = 0; PARMAX(027) = 100; PARDETM(027)=0;
PARMIN(028) = 0; PARMAX(028) = 100; PARDETM(028)=0;
PARMIN(029) = 0; PARMAX(029) = 100; PARDETM(029)=0;
PARMIN(030) = 0; PARMAX(030) = 100; PARDETM(030)=0;
PARMIN(031) = 0; PARMAX(031) = 100; PARDETM(031)=0;
PARMIN(032) = 0; PARMAX(032) = 100; PARDETM(032)=0;
PARMIN(033) = 0; PARMAX(033) = 100; PARDETM(033)=0;
PARMIN(034) = 0; PARMAX(034) = 100; PARDETM(034)=0;
PARMIN(035) = 0; PARMAX(035) = 100; PARDETM(035)=0;
PARMIN(036) = 0; PARMAX(036) = 100; PARDETM(036)=0;
PARMIN(037) = 0; PARMAX(037) = 100; PARDETM(037)=0;
PARMIN(038) = 0; PARMAX(038) = 100; PARDETM(038)=0;
PARMIN(039) = 0; PARMAX(039) = 100; PARDETM(039)=0;
PARMIN(040) = 0; PARMAX(040) = 100; PARDETM(040)=0;
PARMIN(041) = 0; PARMAX(041) = 100; PARDETM(041)=0;
PARMIN(042) = 0; PARMAX(042) = 20; PARDETM(042)=0;
PARMIN(043) = 0; PARMAX(043) = 100; PARDETM(043)=0;
PARMIN(044) = 0; PARMAX(044) = 100; PARDETM(044)=0;
PARMIN(045) = 0; PARMAX(045) = 100; PARDETM(045)=0;
PARMIN(046) = 0; PARMAX(046) = 100; PARDETM(046)=0;
PARMIN(047) = 0; PARMAX(047) = 100; PARDETM(047)=0;
PARMIN(048) = 0; PARMAX(048) = 100; PARDETM(048)=0;
PARMIN(049) = 0; PARMAX(049) = 20; PARDETM(049)=0;
PARMIN(050) = 0; PARMAX(050) = 100; PARDETM(050)=0;
PARMIN(051) = 0; PARMAX(051) = 100; PARDETM(051)=0;
PARMIN(052) = 0; PARMAX(052) = 100; PARDETM(052)=0;
PARMIN(053) = 0; PARMAX(053) = 100; PARDETM(053)=0;
PARMIN(054) = 0; PARMAX(054) = 100; PARDETM(054)=0;
PARMIN(055) = 0; PARMAX(055) = 100; PARDETM(055)=0;
PARMIN(056) = 0; PARMAX(056) = 10; PARDETM(056)=0;
PARMIN(057) = 0; PARMAX(057) = 10; PARDETM(057)=0;
PARMIN(058) = 0; PARMAX(058) = 10; PARDETM(058)=0;
PARMIN(059) = 0; PARMAX(059) = 10; PARDETM(059)=0;
PARMIN(060) = 0; PARMAX(060) = 10; PARDETM(060)=0;
PARMIN(061) = 0; PARMAX(061) = 10; PARDETM(061)=0;
PARMIN(062) = 0; PARMAX(062) = 10; PARDETM(062)=0;
PARMIN(063) = 0; PARMAX(063) = 10; PARDETM(063)=0;
PARMIN(064) = 0; PARMAX(064) = 10; PARDETM(064)=0;
PARMIN(065) = 0; PARMAX(065) = 10; PARDETM(065)=0;
PARMIN(066) = 0; PARMAX(066) = 10; PARDETM(066)=0;
PARMIN(067) = 0; PARMAX(067) = 10; PARDETM(067)=0;
PARMIN(068) = 0; PARMAX(068) = 10; PARDETM(068)=0;
PARMIN(069) = 0; PARMAX(069) = 10; PARDETM(069)=0;
PARMIN(070) = 0; PARMAX(070) = 10; PARDETM(070)=0;
PARMIN(071) = 0; PARMAX(071) = 50; PARDETM(071)=0;
PARMIN(072) = 0; PARMAX(072) = 10; PARDETM(072)=0;
PARMIN(073) = 0; PARMAX(073) = 10; PARDETM(073)=0;
PARMIN(074) = 0; PARMAX(074) = 10; PARDETM(074)=0;
PARMIN(075) = 0; PARMAX(075) = 10; PARDETM(075)=0;
PARMIN(076) = 0; PARMAX(076) = 10; PARDETM(076)=0;
PARMIN(077) = 0; PARMAX(077) = 10; PARDETM(077)=0;
PARMIN(078) = 0; PARMAX(078) = 1; PARDETM(078)=0;
PARMIN(079) = 0; PARMAX(079) = 100000; PARDETM(079)=0;
PARMIN(080) = 0; PARMAX(080) = 1e+06; PARDETM(080)=0;
PARMIN(081) = 0; PARMAX(081) = 1; PARDETM(081)=0;
PARMIN(082) = 0; PARMAX(082) = 10; PARDETM(082)=0;
PARMIN(083) = 0; PARMAX(083) = 1; PARDETM(083)=0;
PARMIN(084) = 0; PARMAX(084) = 1; PARDETM(084)=0;
PARMIN(085) = 0.1; PARMAX(085) = 1000; PARDETM(085)=0;
PARMIN(086) = 0.1; PARMAX(086) = 1000; PARDETM(086)=0;
PARMIN(087) = 0; PARMAX(087) = 10; PARDETM(087)=0;
PARMIN(088) = 0.1; PARMAX(088) = 1000; PARDETM(088)=0;
PARMIN(089) = 0; PARMAX(089) = 100; PARDETM(089)=0;
PARMIN(090) = 0; PARMAX(090) = 100; PARDETM(090)=0;
PARMIN(091) = 0; PARMAX(091) = 100; PARDETM(091)=0;
PARMIN(092) = 0; PARMAX(092) = 100; PARDETM(092)=0;
PARMIN(093) = 0; PARMAX(093) = 100; PARDETM(093)=0;
PARMIN(094) = 0; PARMAX(094) = 100; PARDETM(094)=0;
PARMIN(095) = 0; PARMAX(095) = 100; PARDETM(095)=0;
PARMIN(096) = 0; PARMAX(096) = 100; PARDETM(096)=0;
PARMIN(097) = 0; PARMAX(097) = 100; PARDETM(097)=0;
PARMIN(098) = 0; PARMAX(098) = 150000; PARDETM(098)=0;
PARMIN(099) = 0; PARMAX(099) = 150000; PARDETM(099)=0;
PARMIN(100) = -10; PARMAX(100) = 100; PARDETM(100)=0;
PARMIN(101) = -10; PARMAX(101) = 100; PARDETM(101)=0;
PARMIN(102) = 0; PARMAX(102) = 20; PARDETM(102)=0;
PARMIN(103) = 0; PARMAX(103) = 80000; PARDETM(103)=0;
PARMIN(104) = -3; PARMAX(104) = 1440; PARDETM(104)=0;
PARMIN(105) = -3; PARMAX(105) = 1440; PARDETM(105)=0;
PARMIN(106) = 0; PARMAX(106) = 1000; PARDETM(106)=1;
PARMIN(107) = 0; PARMAX(107) = 1000; PARDETM(107)=1;
PARMIN(108) = 0; PARMAX(108) = 1000; PARDETM(108)=1;
PARMIN(109) = 0; PARMAX(109) = 1000; PARDETM(109)=1;
PARMIN(110) = 0; PARMAX(110) = 1000; PARDETM(110)=1;
PARMIN(111) = 0; PARMAX(111) = 1000; PARDETM(111)=1;
PARMIN(112) = 0; PARMAX(112) = 1000; PARDETM(112)=1;
PARMIN(113) = 0; PARMAX(113) = 1000; PARDETM(113)=1;
PARMIN(114) = 0; PARMAX(114) = 1; PARDETM(114)=1;
PARMIN(115) = 0; PARMAX(115) = 1; PARDETM(115)=1;
PARMIN(116) = 0; PARMAX(116) = 1; PARDETM(116)=1;
PARMIN(117) = 0; PARMAX(117) = 1; PARDETM(117)=1;
PARMIN(118) = 0; PARMAX(118) = 200100; PARDETM(118)=1;
PARMIN(119) = 0; PARMAX(119) = 10000; PARDETM(119)=1;
PARMIN(120) = 0.1; PARMAX(120) = 50; PARDETM(120)=1;
PARMIN(121) = 0.1; PARMAX(121) = 50; PARDETM(121)=1;
PARMIN(122) = 0.1; PARMAX(122) = 50; PARDETM(122)=1;
PARMIN(123) = 0.1; PARMAX(123) = 50; PARDETM(123)=1;
PARMIN(124) = 0.1; PARMAX(124) = 50; PARDETM(124)=1;
PARMIN(125) = 0.1; PARMAX(125) = 50; PARDETM(125)=1;
PARMIN(126) = 0.1; PARMAX(126) = 50; PARDETM(126)=1;
PARMIN(127) = 0; PARMAX(127) = 20; PARDETM(127)=1;
PARMIN(128) = 0; PARMAX(128) = 10; PARDETM(128)=1;
PARMIN(129) = 0; PARMAX(129) = 10; PARDETM(129)=1;
PARMIN(130) = 0; PARMAX(130) = 10; PARDETM(130)=1;
PARMIN(131) = 0; PARMAX(131) = 10; PARDETM(131)=1;
PARMIN(132) = 0; PARMAX(132) = 100; PARDETM(132)=1;
PARMIN(133) = 0; PARMAX(133) = 10; PARDETM(133)=1;
PARMIN(134) = 0; PARMAX(134) = 10; PARDETM(134)=1;
PARMIN(135) = 0; PARMAX(135) = 20; PARDETM(135)=1;
PARMIN(136) = 0; PARMAX(136) = 20; PARDETM(136)=1;
PARMIN(137) = 0; PARMAX(137) = 1000; PARDETM(137)=1;
PARMIN(138) = 0; PARMAX(138) = 1000; PARDETM(138)=1;
PARMIN(139) = 0; PARMAX(139) = 100; PARDETM(139)=1;
PARMIN(140) = 0; PARMAX(140) = 10; PARDETM(140)=1;
PARMIN(141) = 0; PARMAX(141) = 100; PARDETM(141)=1;
PARMIN(142) = 0; PARMAX(142) = 100; PARDETM(142)=1;
PARMIN(143) = 0; PARMAX(143) = 100; PARDETM(143)=0; % ka
PARMIN(144) = 0; PARMAX(144) = 200; PARDETM(144)=0; % insulin_inf
PARMIN(145) = 0; PARMAX(145) = 2000; PARDETM(145)=1;% alpha
PARMIN(146) = 0; PARMAX(146) = 2000; PARDETM(146)=1; % beta
PARMIN(147) = 0; PARMAX(147) = 200000; PARDETM(147)=0; % CHO [mg]
PARMIN(148) = 0; PARMAX(148) = 200; PARDETM(148)=0; % b
PARMIN(149) = 0; PARMAX(149) = 200; PARDETM(149)=0; % c
PARMIN(150) = 0; PARMAX(150) = 200; PARDETM(150)=0; % k_min
PARMIN(151) = 0; PARMAX(151) = 200; PARDETM(151)=0; % k_max
PARMIN(152) = 0; PARMAX(152) = 200; PARDETM(152)=0; % k_gri
PARMIN(153) = 0; PARMAX(153) = 200000; PARDETM(153)=0; % Time_meal

Tzero                    = Tzero_; % (bigtheta 001);
Tend                     = Tend_; % (bigtheta 002);
Tdelta                   = 0.1; % (bigtheta 003);
QfloGB                   = 0.59; % (bigtheta 004);
VolGBV                   = 0.35; % (bigtheta 005);
VolBI                    = 0.45; % (bigtheta 006);
TdifB                    = 2.1; % (bigtheta 007);
GlucH0                   = 5.07333; % (bigtheta 008);
GammaBGU                 = 0.388889; % (bigtheta 009);
QfloGL                   = 1.26; % (bigtheta 010);
QfloGK                   = 1.01; % (bigtheta 011);
QfloGP                   = 1.51; % (bigtheta 012);
QfloGH                   = 4.37; % (bigtheta 013);
GammaRBCU                = 0.0555556; % (bigtheta 014);
VolGH                    = 1.38; % (bigtheta 015);
QfloGJ                   = 1.01; % (bigtheta 016);
VolGJ                    = 1.12; % (bigtheta 017);
GammaJGU                 = 0.111111; % (bigtheta 018);
QfloGA                   = 0.25; % (bigtheta 019);
VolGL                    = 2.51; % (bigtheta 020);
VolGK                    = 0.66; % (bigtheta 021);
VolGPV                   = 1.04; % (bigtheta 022);
VolPI                    = 6.74; % (bigtheta 023);
TdifGP                   = 5; % (bigtheta 024);
GammaBPGU                = 0.194444; % (bigtheta 025);
beta0PGU                 = 7.03; % (bigtheta 026);
beta1PGU                 = 6.52; % (bigtheta 027);
beta2PGU                 = 0.338; % (bigtheta 028);
beta3PGU                 = 5.82; % (bigtheta 029);
beta0HGP                 = 2.7; % (bigtheta 030);
```

```
beta1HGP             = 0.388852; % (bigtheta 031);
tauCgon              = 65; % (bigtheta 032);
beta2HGP             = 1.21; % (bigtheta 033);
beta3HGP             = 1.14; % (bigtheta 034);
beta4HGP             = 1.66; % (bigtheta 035);
beta5HGP             = 0.887748; % (bigtheta 036);
tauInsu              = 25; % (bigtheta 037);
beta6HGP             = 1.42; % (bigtheta 038);
beta7HGP             = 1.41; % (bigtheta 039);
beta8HGP             = 0.62; % (bigtheta 040);
beta9HGP             = 0.504543; % (bigtheta 041);
GammaHGP0            = 0.861111; % (bigtheta 042);
beta0HGU             = 2; % (bigtheta 043);
beta1HGU             = 0.549306; % (bigtheta 044);
beta2HGU             = 5.66; % (bigtheta 045);
beta3HGU             = 5.66; % (bigtheta 046);
beta4HGU             = 2.44; % (bigtheta 047);
beta5HGU             = 1.4783; % (bigtheta 048);
GammaHGU0            = 0.111111; % (bigtheta 049);
beta0KGE             = 0.394444; % (bigtheta 050);
beta1KGE             = 0.394444; % (bigtheta 051);
beta2KGE             = 0.198; % (bigtheta 052);
beta3KGE             = 25.5556; % (bigtheta 053);
beta4KGE             = 1.834; % (bigtheta 054);
beta5KGE             = 0.0872; % (bigtheta 055);
QfloIB               = 0.45; % (bigtheta 056);
VolIB                = 0.26; % (bigtheta 057);
VolIH                = 0.99; % (bigtheta 058);
QfloIL               = 0.9; % (bigtheta 059);
QfloIK               = 0.72; % (bigtheta 060);
QfloIP               = 1.05; % (bigtheta 061);
QfloIH               = 3.12; % (bigtheta 062);
VolIJ                = 0.94; % (bigtheta 063);
QfloIJ               = 0.72; % (bigtheta 064);
VolIL                = 1.14; % (bigtheta 065);
QfloIA               = 0.18; % (bigtheta 066);
FracLIC              = 0.4; % (bigtheta 067);
FracKIC              = 0.3; % (bigtheta 068);
VolIK                = 0.51; % (bigtheta 069);
VolIPV               = 0.74; % (bigtheta 070);
TdifIP               = 20; % (bigtheta 071);
FracPIC              = 0.15; % (bigtheta 072);
beta1PIR             = 3.27; % (bigtheta 073);
beta2PIR             = 7.33333; % (bigtheta 074);
beta3PIR             = 2.879; % (bigtheta 075);
beta4PIR             = 3.02; % (bigtheta 076);
beta5PIR             = 1.11; % (bigtheta 077);
```

```
KappaRinsu           = 0.00794; % (bigtheta 078);
Rinsu0               = 44310; % (bigtheta 079);
KappaRinsuPotn       = 4025; % (bigtheta 080);
KappaPotnPtgt        = 0.0482; % (bigtheta 081);
KappaPinhPrp         = 0.931; % (bigtheta 082);
EMME1                = 0.00747; % (bigtheta 083);
EMME2                = 0.0958; % (bigtheta 084);
InsuPV0              = 91; % (bigtheta 085);
Cgon0                = 11.48; % (bigtheta 086);
GammaMCC             = 0.91; % (bigtheta 087);
VolC                 = 11.31; % (bigtheta 088);
beta0PCR             = 2.93; % (bigtheta 089);
beta1PCR             = 2.1; % (bigtheta 090);
beta2PCR             = 4.18; % (bigtheta 091);
beta3PCR             = 0.621325; % (bigtheta 092);
beta4PCR             = 1.31; % (bigtheta 093);
beta5PCR             = 0.61; % (bigtheta 094);
beta6PCR             = 1.06; % (bigtheta 095);
beta7PCR             = 0.471419; % (bigtheta 096);
Func20               = 0; % (bigtheta 097);
GammaIVG0            = 0; % (bigtheta 098);
GammaIVGin           = 0; %64.81; % (bigtheta 099); changed
TimeIVG              = -3; % (bigtheta 100);
TimeIVGend           = 0; % (bigtheta 101);
GammaIVI0            = 0; % (bigtheta 102);
GammaIVIin           = 0; % (bigtheta 103);
TimeIVI              = 0; % (bigtheta 104);
TimeIVIend           = 0; % (bigtheta 105);
InsuH0               = 107.059; % (bigtheta 106);
InsuK0               = 74.9412; % (bigtheta 107);
InsuB0               = 107.059; % (bigtheta 108);
InsuJ0               = 107.059; % (bigtheta 109);
InsuPI0              = 40.9651; % (bigtheta 110);
InsuL0               = 151.488; % (bigtheta 111);
GammaBPIR            = 0; %130.879; % (bigtheta 112); zero for T1D
GammaPIC0            = 16.8618; % (bigtheta 113);
Pprp0                = 0.19032; % (bigtheta 114);
Ptgt0                = 0.158572; % (bigtheta 115);
Pinh0                = 0.19032; % (bigtheta 116);
Potn0                = 0.158572; % (bigtheta 117);
InitialRinsu0        = 108507; % (bigtheta 118);
Secr0                = 128.53; % (bigtheta 119);
GlucPV0              = 4.94456; % (bigtheta 120);
GlucK0               = 5.07333; % (bigtheta 121);
GlucBV0              = 4.4142; % (bigtheta 122);
GlucJ0               = 4.96332; % (bigtheta 123);
GlucL0               = 5.58039; % (bigtheta 124);
```

```
GlucBI0              = 2.59938; % (bigtheta 125);
GlucPI0              = 4.80032; % (bigtheta 126);
MIPGU0               = 0.992859; % (bigtheta 127);
MCHGP0               = 1; % (bigtheta 128);
MCOHGP0              = 1; % (bigtheta 129);
MIHGP0               = 1; % (bigtheta 130);
MIHGPinf0            = 1; % (bigtheta 131);
MGHGP0               = 1; % (bigtheta 132);
MIHGU0               = 1; % (bigtheta 133);
MIHGUinf0            = 1; % (bigtheta 134);
MGHGU0               = 1; % (bigtheta 135);
GammaKGE0            = 0.000236777; % (bigtheta 136);
GammaLIC0            = 90.8929; % (bigtheta 137);
GammaKIC0            = 23.1247; % (bigtheta 138);
MGPCR0               = 1; % (bigtheta 139);
MIPCR0               = 1; % (bigtheta 140);
GammaPCC0            = 10.4468; % (bigtheta 141);
GammaBPCR            = 10.4468; % (bigtheta 142);
ka                   = 0.0144; % (bigtheta 143); absorption constant for subcutaneous insu
insulin_inf          = Insu_init; % (bigtheta 144); infusion rate of subcutaneous insulin
alpha                = 0; % (bigtheta 145);
beta                 = 0; % (bigtheta 146);
CHO                  = CHO_0; % (bigtheta 147);  mg of carbohydrate
b                    = b_p0; % (bigtheta 148);
c                    = c_p0; % (bigtheta 149);
k_min                = k_min_p0; % (bigtheta 150);
k_max                = k_max_p0; % (bigtheta 151);
k_gri                = k_gri_p0; % (bigtheta 152);
Time_meal            = 10000; % (bigtheta 153);

% run DetermineParameters on the named parameters before assigning all values to bigtheta
SorensenDetermineParameters;

% build bigtheta
bigtheta = zeros(nPars,1);
SorensenParvals2Bigtheta;
```

SorensenInitializeStateVars.m

```
% ==================================================================================
% SCRIPT SorensenInitializeStateVars.m: assign the starting values to ALL
% dependent variables
% Sorensen V01.01.41 20190724 (Gemini 13.01.06, BMLib 10.0.2, Autocoder 02.11.09
% coded 24-Jul-2019 16:06:54)
% ==================================================================================

global VARMIN VARMAX;

nDepVars = 62;

VARMIN = zeros(63,1);
VARMAX = zeros(63,1);
VARMIN(001) = 0; VARMAX(001) = 1440;
VARMIN(002) = 0; VARMAX(002) = 200;
VARMIN(003) = 0; VARMAX(003) = 200;
VARMIN(004) = 0; VARMAX(004) = 200;
VARMIN(005) = 0; VARMAX(005) = 200;
VARMIN(006) = 0; VARMAX(006) = 200;
VARMIN(007) = 0; VARMAX(007) = 200;
VARMIN(008) = 0; VARMAX(008) = 100;
VARMIN(009) = 0; VARMAX(009) = 200;
VARMIN(010) = 0; VARMAX(010) = 200;
VARMIN(011) = 0; VARMAX(011) = 200;
VARMIN(012) = 0; VARMAX(012) = 200;
VARMIN(013) = 0; VARMAX(013) = 200;
VARMIN(014) = 0; VARMAX(014) = 999;
VARMIN(015) = 0; VARMAX(015) = 999;
VARMIN(016) = 0; VARMAX(016) = 999;
VARMIN(017) = 0; VARMAX(017) = 999;
VARMIN(018) = 0; VARMAX(018) = 999;
VARMIN(019) = 0; VARMAX(019) = 999;
VARMIN(020) = -20; VARMAX(020) = 999;
VARMIN(021) = 0; VARMAX(021) = 999;
VARMIN(022) = 0; VARMAX(022) = 999;
VARMIN(023) = 0; VARMAX(023) = 999;
VARMIN(024) = 0; VARMAX(024) = 999;
VARMIN(025) = 0; VARMAX(025) = 999;
VARMIN(026) = 0; VARMAX(026) = 9999;
VARMIN(027) = 0; VARMAX(027) = 10000;
VARMIN(028) = 0; VARMAX(028) = 9999;
VARMIN(029) = 0; VARMAX(029) = 9999;
VARMIN(030) = 0; VARMAX(030) = 9999;
VARMIN(031) = 0; VARMAX(031) = 99999;
VARMIN(032) = 0; VARMAX(032) = 9999;
VARMIN(033) = 0; VARMAX(033) = 9999;
VARMIN(034) = 0; VARMAX(034) = 9999;
VARMIN(035) = 0; VARMAX(035) = 200;
VARMIN(036) = 0; VARMAX(036) = 999;
VARMIN(037) = 0; VARMAX(037) = 99999;
VARMIN(038) = 0; VARMAX(038) = 999;
VARMIN(039) = 0; VARMAX(039) = 999;
VARMIN(040) = 0; VARMAX(040) = 90000;
VARMIN(041) = 0; VARMAX(041) = 1000;
VARMIN(042) = 0; VARMAX(042) = 1000;
VARMIN(043) = 0; VARMAX(043) = 1e+07;
VARMIN(044) = 0; VARMAX(044) = 99999;
VARMIN(045) = 0; VARMAX(045) = 1000;
VARMIN(046) = 0; VARMAX(046) = 1000;
VARMIN(047) = 0; VARMAX(047) = 1000;
VARMIN(048) = 0; VARMAX(048) = 999;
VARMIN(049) = 0; VARMAX(049) = 200;
VARMIN(050) = 0; VARMAX(050) = 999;
VARMIN(051) = 0; VARMAX(051) = 999;
VARMIN(052) = 0; VARMAX(052) = 10;
VARMIN(053) = 0; VARMAX(053) = 10;
VARMIN(054) = 0; VARMAX(054) = 1000;
VARMIN(055) = 0; VARMAX(055) = 1000;
VARMIN(056) = 0; VARMAX(056) = 1000;
VARMIN(057) = 0; VARMAX(057) = 1000;
VARMIN(058) = 0; VARMAX(058) = 10000;
VARMIN(059) = 0; VARMAX(059) = 10000;
VARMIN(060) = 0; VARMAX(060) = 10000;
VARMIN(061) = 0; VARMAX(061) = 10000;
VARMIN(062) = 0; VARMAX(062) = 10000;
VARMIN(063) = 0; VARMAX(063) = 10000;

GlucBV = Gluc_init; % GlucBV0; % CurrentY(001)
GlucBI = Gluc_init; %GlucBI0; % CurrentY(002)
GlucH = Gluc_init; %GlucH0; % CurrentY(003)
GlucNH = 1.000000000000; % CurrentY(004)
GlucJ = Gluc_init; %GlucJ0; % CurrentY(005)
GlucL = Gluc_init; %GlucL0; % CurrentY(006)
GlucNL = 1.000000000000; % CurrentY(007)
GlucK = Gluc_init; %GlucK0; % CurrentY(008)
GlucPV = Gluc_init; %GlucPV0; % CurrentY(009)
GlucPI = Gluc_init; %GlucPI0; % CurrentY(010)
GlucNPI = 1.000000000000; % CurrentY(011)
GammaPGU = GammaBPGU; % CurrentY(012)
MIPGU = MIPGU0; % CurrentY(013)
GammaHGP = GammaHGP0; % CurrentY(014)
MIHGP = MIHGP0; % CurrentY(015)
MIHGPinf = MIHGPinf0; % CurrentY(016)
MCHGP = MCHGP0; %   CurrentY(017)
MCOHGP = MCOHGP0; %   CurrentY(018)
Fun2 = Func20; % CurrentY(019)
MGHGP = MGHGP0; % CurrentY(020)
GammaHGU = GammaHGU0; % CurrentY(021)
MIHGU = MIHGU0; % CurrentY(022)
```

```
MIHGUinf = MIHGUinf0; % CurrentY(023)
MGHGU = MGHGU0; % CurrentY(024)
GammaKGE = GammaKGE0; % CurrentY(025)
InsuB = Insu_init; %InsuB0; % CurrentY(026)
InsuH = Insu_init; %InsuH0; % CurrentY(027)
InsuNH = 1.000000000000; % CurrentY(028)
InsuJ = Insu_init; %InsuJ0; % CurrentY(029)
InsuL = Insu_init; %InsuL0; % CurrentY(030)
InsuK = Insu_init; %InsuK0; % CurrentY(031)
InsuPV = Insu_init; %InsuPV0; % CurrentY(032)
InsuPI = Insu_init; %InsuPI0; % CurrentY(033)
InsuNPI = 1.000000000000; % CurrentY(034)
InsuNL = 1.000000000000; % CurrentY(035)
GammaLIC = GammaLIC0; % CurrentY(036)
GammaKIC = GammaKIC0; % CurrentY(037)
GammaPIC = GammaPIC0; % CurrentY(038)
GammaPIR = 0; %GammaBPIR; % CurrentY(039) zero for type 1 diabetes
Potn = Potn0; % CurrentY(040)
Pinh = Pinh0; % CurrentY(041)
Rinsu = InitialRinsu0; % CurrentY(042)
Secr = Secr0; % CurrentY(043)
SecrN = 1.000000000000; % CurrentY(044)
Pprp = Pprp0; % CurrentY(045)
Ptgt = Ptgt0; % CurrentY(046)
Cgon = Cgon0; % CurrentY(047)
CgonN = 1.000000000000; % CurrentY(048)
GammaPCC = GammaPCC0; % CurrentY(049)
GammaPCR = GammaBPCR; % CurrentY(050)
MGPCR = MGPCR0; % CurrentY(051)
MIPCR = MIPCR0; % CurrentY(052)
GammaIVG = GammaIVG0; % CurrentY(053)
GammaIVI = GammaIVI0; % CurrentY(054)
Gamma_meal = 0; % CurrentY(055)
GammaISC = Insu_init; % CurrentY(056)
S1 = Insu_init; % CurrentY(057)
S2 = Insu_init; % CurrentY(058)
Q_sto = 0; % CurrentY(059)
Q_sto1 = 0; % CurrentY(060)
Q_sto2 = 0; % CurrentY(061)
k_empt = 0; % CurrentY(062)

CurrentY = zeros(62,1);

SorensenNamedVars2CurrentY;
SorensenForceVars;
```

DetermineParameters.m

```
% ================================================================================
% SCRIPT SorensenDetermineParameters.m: compute the values of determined from free
% parameters. Sorensen V01.01.41 20190724 (Gemini 13.01.06, BMLib 10.0.2
% Autocoder 02.11.09, coded 24-Jul-2019 16:06:54)
% ===============================================================================

InsuH0 = InsuPV0/(1-FracPIC);    % (bigtheta 106)
InsuK0 = InsuH0*(1-FracKIC);    % (bigtheta 107)
InsuB0 = InsuH0;    % (bigtheta 108)
InsuJ0 = InsuH0;    % (bigtheta 109)
InsuPI0 = InsuPV0-((QfloIP*TdifIP/VolPI)*(InsuH0-InsuPV0));    % (bigtheta 110)
InsuL0 = 1/QfloIL*(QfloIH*InsuH0-QfloIB*InsuB0
    -QfloIK*InsuK0-QfloIP*InsuPV0);    % (bigtheta 111)
GammaBPIR = 0; %QfloIL/(1-FracLIC)*InsuL0 - QfloIJ*InsuJ0-QfloIA*InsuH0; % (bigtheta 112)
GammaPIC0 = InsuPI0/(((1-FracPIC)/FracPIC)*(1/QfloIP)-TdifIP/VolPI); % (bigtheta 113)
Pprp0 = pow((GlucH0),beta1PIR) /( pow((beta2PIR),beta1PIR)+
    beta3PIR*pow((GlucH0),beta4PIR) ); % (bigtheta 114)
Ptgt0 = pow(Pprp0,beta5PIR);    % (bigtheta 115)
Pinh0 = Pprp0;    % (bigtheta 116)
Potn0 = Ptgt0;    % (bigtheta 117)
InitialRinsu0 = ((KappaRinsu*Rinsu0)+ KappaRinsuPotn * Potn0)
    /(KappaRinsu+EMME1* Potn0); % (bigtheta 118)
Secr0 = EMME1*Ptgt0*InitialRinsu0;    % (bigtheta 119)
GlucPV0 = GlucH0 - GammaBPGU/QfloGP;    % (bigtheta 120)
GlucK0 = GlucH0;    % (bigtheta 121)
GlucBV0 = GlucH0 - GammaBGU/QfloGB;    % (bigtheta 122)
GlucJ0 = GlucH0-GammaJGU/QfloGJ;    % (bigtheta 123)
GlucL0 = (QfloGA*GlucH0+QfloGJ*GlucJ0+GammaHGP0-GammaHGU0)/QfloGL; % (bigtheta 124)
GlucBI0 = GlucBV0-(GammaBGU*TdifB)/VolBI;    % (bigtheta 125)
GlucPI0 = GlucPV0-GammaBPGU*TdifGP/VolPI;    % (bigtheta 126)
MIPGU0 = beta0PGU+beta1PGU*tanh(beta2PGU*(1-beta3PGU));    % (bigtheta 127)
MCHGP0 = beta0HGP   * tanh(beta1HGP * 1) - Func20;    % (bigtheta 128)
MC0HGP0 = beta0HGP   * tanh(beta1HGP * 1);    % (bigtheta 129)
MIHGP0 = beta2HGP - beta3HGP * tanh(beta4HGP * (1-beta5HGP));    % (bigtheta 130)
MIHGPinf0 = MIHGP0;    % (bigtheta 131)
MGHGP0 = beta6HGP-beta7HGP*tanh(beta8HGP*(1-beta9HGP));    % (bigtheta 132)
MIHGU0 = beta0HGU * tanh(beta1HGU);    % (bigtheta 133)
MIHGUinf0 = MIHGU0;    % (bigtheta 134)
MGHGU0 = beta2HGU+beta3HGU*tanh(beta4HGU*(1-beta5HGU));    % (bigtheta 135)
GammaKGE0 = (GlucK0<beta3KGE) * (beta0KGE+beta1KGE*tanh(beta2KGE*
    (GlucK0-beta3KGE))) + (GlucK0 >= beta3KGE)
    * (-beta4KGE+beta5KGE*GlucK0); % (bigtheta 136)
GammaLIC0 = FracLIC*(QfloIA*InsuH0+QfloIJ*InsuJ0+GammaBPIR);    % (bigtheta 137)
GammaKIC0 = FracKIC*(QfloIK*InsuH0);    % (bigtheta 138)
```

```
MGPCR0 = beta0PCR - beta1PCR * tanh(beta2PCR * (1-beta3PCR));   % (bigtheta 139)
MIPCR0 = beta4PCR - beta5PCR * tanh(beta6PCR * (1-beta7PCR));   % (bigtheta 140)
GammaPCC0 = Cgon0*GammaMCC;    % (bigtheta 141)
GammaBPCR = GammaPCC0;         % (bigtheta 142)
alpha = 5 / (2*CHO*(1-b)); % (bigtheta 145)
beta = 5 / (2*CHO*c);      % (bigtheta 146)
```

SorensenComputeDerivatives.m

```
% =====================================================================================
% SCRIPT SorensenComputeDerivatives.m: compute the derivatives of the differential
% variables. Sorensen V01.01.41 20190724 (Gemini 13.01.06, BMLib 10.0.2
% Autocoder 02.11.09, coded 24-Jul-2019 16:06:54)
% =====================================================================================

% Builds contextually the   out1   column (!) vector used by Matlab's Odefile
out1 = [];
dGlucBVdt = (GlucH - GlucBV) * QfloGB / VolGBV - VolBI / (TdifB * VolGBV) *
    (GlucBV - GlucBI);
out1 = [out1; dGlucBVdt];
dGlucBIdt = 1 / TdifB * (GlucBV - GlucBI) - GammaBGU / VolBI;
out1 = [out1; dGlucBIdt];
dGlucHdt = (QfloGB * GlucBV + QfloGL * GlucL +  QfloGK * GlucK + QfloGP * GlucPV -
    QfloGH * GlucH - GammaRBCU + GammaIVG) / VolGH;
out1 = [out1; dGlucHdt];
% Variable GlucNH (CurrentY 004) is not differentially expressed
out1 = [out1; 0];
dGlucJdt = (GlucH - GlucJ) * QfloGJ / VolGJ + (f*Gamma_meal - GammaJGU) / VolGJ;
out1 = [out1; dGlucJdt];
dGlucLdt = (QfloGA * GlucH + QfloGJ * GlucJ - QfloGL * GlucL + GammaHGP
    - GammaHGU) / VolGL;
out1 = [out1; dGlucLdt];
%  Variable GlucNL (CurrentY 007) is not differentially expressed
out1 = [out1; 0];
dGlucKdt = (GlucH - GlucK) * QfloGK / VolGK - GammaKGE / VolGK;
out1 = [out1; dGlucKdt];
dGlucPVdt = QfloGP / VolGPV * (GlucH - GlucPV) - VolPI / (TdifGP * VolGPV)
    * (GlucPV - GlucPI);
out1 = [out1; dGlucPVdt];
dGlucPIdt = (GlucPV - GlucPI) / TdifGP - GammaPGU / VolPI;
out1 = [out1; dGlucPIdt];
% Variable GlucNPI (CurrentY 011) is not differentially expressed
out1 = [out1; 0];
% Variable GammaPGU (CurrentY 012) is not differentially expressed
out1 = [out1; 0];
```

```
% Variable MIPGU (CurrentY 013) is not differentially expressed
out1 = [out1; 0];
% Variable GammaHGP (CurrentY 014) is not differentially expressed
out1 = [out1; 0];
dMIHGPdt = (MIHGPinf - MIHGP) / tauInsu;
out1 = [out1; dMIHGPdt];
% Variable MIHGPinf (CurrentY 016) is not differentially expressed
out1 = [out1; 0];
% Variable MCHGP (CurrentY 017) is not differentially expressed
out1 = [out1; 0];
% Variable MCOHGP (CurrentY 018) is not differentially expressed
out1 = [out1; 0];
dFun2dt = ((MCOHGP - 1.0) / 2.0 - Fun2) / tauCgon;
out1 = [out1; dFun2dt];
% Variable MGHGP (CurrentY 020) is not differentially expressed
out1 = [out1; 0];
% Variable GammaHGU (CurrentY 021) is not differentially expressed
out1 = [out1; 0];
dMIHGUdt = (MIHGUinf - MIHGU) / tauInsu;
out1 = [out1; dMIHGUdt];
% Variable MIHGUinf (CurrentY 023) is not differentially expressed
out1 = [out1; 0];
% Variable MGHGU (CurrentY 024) is not differentially expressed
out1 = [out1; 0];
% Variable GammaKGE (CurrentY 025) is not differentially expressed
out1 = [out1; 0];
dInsuBdt = QfloIB / VolIB * (InsuH - InsuB);
out1 = [out1; dInsuBdt];
dInsuHdt = (QfloIB * InsuB + QfloIL * InsuL + QfloIK * InsuK + QfloIP * InsuPV
    - QfloIH * InsuH + GammaISC) / VolIH;
out1 = [out1; dInsuHdt];
% Variable InsuNH (CurrentY 028) is not differentially expressed
out1 = [out1; 0];
dInsuJdt = QfloIJ / VolIJ * (InsuH - InsuJ);
out1 = [out1; dInsuJdt];
dInsuLdt = (QfloIA * InsuH + QfloIJ * InsuJ - QfloIL * InsuL + GammaPIR - GammaLIC)
    / VolIL;
out1 = [out1; dInsuLdt];
dInsuKdt = (QfloIK / VolIK) * (InsuH - InsuK) - GammaKIC / VolIK;
out1 = [out1; dInsuKdt];
dInsuPVdt = (QfloIP/VolIPV) * (InsuH - InsuPV) - VolPI / (VolIPV * TdifIP) * (InsuPV - Ins
out1 = [out1; dInsuPVdt];
dInsuPIdt = (1 / TdifIP)
    * (InsuPV - InsuPI) - GammaPIC / VolPI;
out1 = [out1; dInsuPIdt];
% Variable InsuNPI (CurrentY 034) is not differentially expressed
out1 = [out1; 0];
```

```
% Variable InsuNL (CurrentY 035) is not differentially expressed
out1 = [out1; 0];
% Variable GammaLIC (CurrentY 036) is not differentially expressed
out1 = [out1; 0];
% Variable GammaKIC (CurrentY 037) is not differentially expressed
out1 = [out1; 0];
% Variable GammaPIC (CurrentY 038) is not differentially expressed
out1 = [out1; 0];
% Variable GammaPIR (CurrentY 039) is not differentially expressed
out1 = [out1; 0];
dPotndt = KappaPotnPtgt * (Ptgt - Potn);
out1 = [out1; dPotndt];
dPinhdt = KappaPinhPrp * (Pprp - Pinh);
out1 = [out1; dPinhdt];
dRinsudt = KappaRinsu * (Rinsu0 - Rinsu) + KappaRinsuPotn * Potn - Secr;
out1 = [out1; dRinsudt];
% Variable Secr (CurrentY 043) is not differentially expressed
out1 = [out1; 0];
% Variable SecrN (CurrentY 044) is not differentially expressed
out1 = [out1; 0];
% Variable Pprp (CurrentY 045) is not differentially expressed
out1 = [out1; 0];
% Variable Ptgt (CurrentY 046) is not differentially expressed
out1 = [out1; 0];
dCgondt = (GammaPCR - GammaPCC) / VolC;
out1 = [out1; dCgondt];
% Variable CgonN (CurrentY 048) is not differentially expressed
out1 = [out1; 0];
% Variable GammaPCC (CurrentY 049) is not differentially expressed
out1 = [out1; 0];
% Variable GammaPCR (CurrentY 050) is not differentially expressed
out1 = [out1; 0];
% Variable MGPCR (CurrentY 051) is not differentially expressed
out1 = [out1; 0];
% Variable MIPCR (CurrentY 052) is not differentially expressed
out1 = [out1; 0];
% Variable GammaIVG (CurrentY 053) is not differentially expressed
out1 = [out1; 0];
% Variable GammaIVI (CurrentY 054) is not differentially expressed
out1 = [out1; 0];
% Variable Gamma_meal (CurrentY 055) is not differentially expressed
out1 = [out1; 0];
% Variable GammaISC (CurrentY 056) is not differentially expressed
out1 = [out1; 0];
dS1dt = insulin_inf - ka*S1;
out1 = [out1; dS1dt];
dS2dt = ka*S1 - ka*S2;
```

```
out1 = [out1; dS2dt];
% Variable Q_sto (CurrentY 059) is not differentially expressed
out1 = [out1; 0];
dQ_sto1dt = CHO*kronDel(Time, Time_meal) - k_gri*Q_sto1;
out1 = [out1; dQ_sto1dt];
dQ_sto2dt = k_gri*Q_sto1 - k_empt*Q_sto2;
out1 = [out1; dQ_sto2dt];
% Variable k_empt (CurrentY 062) is not differentially expressed
out1 = [out1; 0];
```

SorensenComputeAlgebraic.m

```
% ==================================================================================
% SCRIPT SorensenComputeAlgebraic.m: compute the values of algebraic variables
% Sorensen V01.01.41 20190724 (Gemini 13.01.06, BMLib 10.0.2
% Autocoder 02.11.09, coded 24-Jul-2019 16:06:54)
% ==================================================================================


% Variable GlucBV (CurrentY 001) is not determined
% Variable GlucBI (CurrentY 002) is not determined
% Variable GlucH (CurrentY 003) is not determined
GlucNH = GlucH / GlucH0;
% Variable GlucJ (CurrentY 005) is not determined
% Variable GlucL (CurrentY 006) is not determined
GlucNL = GlucL / GlucL0;
% Variable GlucK (CurrentY 008) is not determined
% Variable GlucPV (CurrentY 009) is not determined
% Variable GlucPI (CurrentY 010) is not determined
GlucNPI = GlucPI / GlucPI0;
GammaPGU = GammaBPGU * GlucNPI * MIPGU;
MIPGU = beta0PGU + beta1PGU * tanh(beta2PGU * (InsuNPI - beta3PGU));
GammaHGP = GammaHGP0 * MIHGP * MCHGP * MGHGP;
% Variable MIHGP (CurrentY 015) is not determined
MIHGPinf = beta2HGP - beta3HGP * tanh(  beta4HGP*(InsuNL - beta5HGP));
MCHGP = MC0HGP - Fun2;
MC0HGP = beta0HGP * tanh(beta1HGP * CgonN);
% Variable Fun2 (CurrentY 019) is not determined
MGHGP = (beta6HGP - beta7HGP * tanh(beta8HGP * (GlucNL - beta9HGP)));
GammaHGU = GammaHGU0 * MIHGU * MGHGU;
% Variable MIHGU (CurrentY 022) is not determined
MIHGUinf = beta0HGU * tanh(beta1HGU *  InsuNL);
MGHGU = beta2HGU + beta3HGU * tanh(beta4HGU * (GlucNL -  beta5HGU));
GammaKGE = (GlucK < beta3KGE) * (beta0KGE + beta1KGE * tanh(beta2KGE*
    (GlucK - beta3KGE ))) + (GlucK >= beta3KGE) * (- beta4KGE + beta5KGE * GlucK);
```

```
% Variable InsuB (CurrentY 026) is not determined
% Variable InsuH (CurrentY 027) is not determined
InsuNH = InsuH / InsuH0;
% Variable InsuJ (CurrentY 029) is not determined
% Variable InsuL (CurrentY 030) is not determined
% Variable InsuK (CurrentY 031) is not determined
% Variable InsuPV CurrentY 032) is not determined
% Variable InsuPI (CurrentY 033) is not determined
InsuNPI = InsuPI / InsuPI0;
InsuNL = InsuL / InsuL0;
GammaLIC = FracLIC * (QfloIA * InsuH + QfloIJ * InsuJ + GammaPIR);
GammaKIC = FracKIC * (QfloIK * InsuH);
GammaPIC = InsuPI / (((1.0 - FracPIC  ) / FracPIC) * (1 / QfloIP) - (TdifIP / VolPI));
GammaPIR = SecrN * GammaBPIR;
% Variable Potn (CurrentY 040) is not determined
% Variable Pinh (CurrentY 041) is not determined
% Variable Rinsu (CurrentY 042) is not determined
Secr = (Pprp > Pinh)*((EMME1 * Ptgt + EMME2 * (Pprp - Pinh)) * Rinsu)
    + (Pprp <= Pinh) * (EMME1 * Ptgt * Rinsu);
SecrN = Secr / Secr0;
Pprp = pow(GlucH,beta1PIR)/(pow(beta2PIR,beta1PIR)+beta3PIR*pow(GlucH,beta4PIR));
Ptgt = pow(Pprp,beta5PIR);
% Variable Cgon (CurrentY 047) is not determined
CgonN = Cgon / Cgon0;
GammaPCC = GammaMCC * Cgon;
GammaPCR = GammaBPCR * MGPCR * MIPCR;
MGPCR = beta0PCR - beta1PCR * tanh(beta2PCR * (GlucNH - beta3PCR));
MIPCR = beta4PCR - beta5PCR * tanh(beta6PCR * (InsuNH - beta7PCR));
GammaIVG = GammaIVG0+(GammaIVGin)*(Time>=TimeIVG)*(Time<=TimeIVGend);
GammaIVI = GammaIVI0+(GammaIVIin)*(Time>=TimeIVI)*(Time<=TimeIVIend);
Gamma_meal = k_empt*Q_sto2;
GammaISC = S2;
% Variable S1 (CurrentY 057) is not determined
% Variable S2 (CurrentY 058) is not determined
Q_sto = Q_sto1 + Q_sto2;
% Variable Q_sto1 (CurrentY 060) is not determined
% Variable Q_sto2 (CurrentY 061) is not determined
k_empt = k_min + (k_max - k_min)/2 * (tanh(alpha*(Q_sto-b*CHO))
    - tanh(beta*(Q_sto-c*CHO)) + 2);
```

D Code - data preparation

This section presents the code from the preparation of the Ohio T1DM dataset. That includes converting the XML-files into xlsx-files, extracting the necessary data (glucose, insulin, meals), convert units, increase sample time through interpolation and Kalman filtering and lastly saving the prepared data for each patient into a large xlsx-file.

Convert from XML to xlsx (main.m)

This code file was given to me by my supervisor Hasti Khoshamadi.

```
clc;
clear all;
sampleXMLfile = ’591-ws-training.xml’;
mlStruct = parseXML(sampleXMLfile);
clear Data
for i= 1:18
    clear V
    clear t
    clear t2
    clear t3
    t1 = {};
    t2 = {};
    n=floor(length(mlStruct.Children(2*i).Children)/2);

    for j=1:n
        if i==5 % bolus
            V(j,1)=str2double(mlStruct.Children(2*i).Children(2*j).Attributes(1).Value);
            V(j,2)=str2double(mlStruct.Children(2*i).Children(2*j).Attributes(2).Value);
            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(3).Value;
            t(j,1) = datetime(DateString,’InputFormat’,’dd-MM-yyyy HH:mm:ss’);
            t2j,1=datestr(t(j,1),’dd-mmm-yyyy HH:MM:SS’);
            t3(j,1)=juliandate(t(j,1));
        elseif i==6 % meal
            V(j,1)=str2double(mlStruct.Children(2*i).Children(2*j).Attributes(1).Value);
            Food_Type=mlStruct.Children(2*i).Children(2*j).Attributes(3).Value;

            V(j,2)= strcmp(Food_Type, ’Breakfast’)+...
                2*strcmp(Food_Type, ’Lunch’)+...
                3*strcmp(Food_Type, ’Dinner’)+...
                4*strcmp(Food_Type, ’Snack’)+...
                5*strcmp(Food_Type, ’HypoCorrection’);


            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(2).Value;
            t(j,1) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');
            t2j,1=datestr(t(j,1),'dd-mmm-yyyy HH:MM:SS');
            t3(j,1)=juliandate(t(j,1));
        elseif i==7 || i==8 || i==18
            V(j,1)=str2double(mlStruct.Children(2*i).Children(2*j).Attributes(1).Value);
            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(2).Value;
            t(j,1) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');
            t2j,1=datestr(t(j,1),'dd-mmm-yyyy HH:MM:SS');
            t3(j,1)=juliandate(t(j,1));
            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(3).Value;
            t(j,2) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');%ts_end
            t2j,2=datestr(t(j,2),'dd-mmm-yyyy HH:MM:SS');
            t3(j,2)=juliandate(t(j,2));
        elseif i==9 ||  i==11
            V(j,1)=0;
            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(2).Value;
            t(j,1) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');
            t2j,1=datestr(t(j,1),'dd-mmm-yyyy HH:MM:SS');
            t3(j,1)=juliandate(t(j,1));
        elseif i==10
            V(j,1)=0;
            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(1).Value;
            t(j,1) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');
            t2j,1=datestr(t(j,1),'dd-mmm-yyyy HH:MM:SS');
            t3(j,1)=juliandate(t(j,1));
        elseif i==12
            V(j,1)=str2double(mlStruct.Children(2*i).Children(2*j).Attributes(2).Value);
            V(j,2)=str2double(mlStruct.Children(2*i).Children(2*j).Attributes(3).Value);
            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(4).Value;
            t(j,1) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');
            t2j,1=datestr(t(j,1),'dd-mmm-yyyy HH:MM:SS');
            t3(j,1)=juliandate(t(j,1));
        else

            V(j,1)=str2double(mlStruct.Children(2*i).Children(2*j).Attributes(end).Value);
            DateString = mlStruct.Children(2*i).Children(2*j).Attributes(1).Value;
            t(j,1) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');
            t2j,1=datestr(t(j,1),'dd-mmm-yyyy HH:MM:SS');
            t3(j,1)=juliandate(t(j,1));
            if i==4%temp_ba and sleep
                DateString = mlStruct.Children(2*i).Children(2*j).Attributes(2).Value;
                t(j,2) = datetime(DateString,'InputFormat','dd-MM-yyyy HH:mm:ss');
                t2j,2=datestr(t(j,2),'dd-mmm-yyyy HH:MM:SS');
                t3(j,2)=juliandate(t(j,2));
            end
        end
    end
    Data{i}.Name=mlStruct.Children(2*i).Name;
    Data{i}.Value=V;
    Data{i}.Time=t;
    Data{i}.Time2=t2;
    Data{i}.Time3=t3;
end
```

Save converted files (save_results_in_folder.m)

This code file was given to me by my supervisor Hasti Khoshamadi.

```
% specify folder to save files in
folder = sampleXMLfile(1:3);

% save files
glucose_level=table(Data{1}.Time, Data{1}.Value);
path = strcat(folder, '/glucose_level.xlsx');
writetable(glucose_level, path);

basal=table(Data{3}.Time, Data{3}.Value);
path = strcat(folder, '/basal.xlsx');
writetable(basal, path, 'Sheet', 1);

temp_basal=table(Data{4}.Time, Data{4}.Value);
path = strcat(folder, '/temp_basal.xlsx');
writetable(temp_basal, path, 'Sheet', 1);

bolus=table(Data{5}.Time,Data{5}.Value);
path = strcat(folder, '/bolus.xlsx');
writetable(bolus, path, 'Sheet', 1);

meal=table(Data{6}.Time,Data{6}.Value);
path = strcat(folder, '/meal.xlsx');
writetable(meal, path, 'Sheet', 1);
```

interpolate.m

```
% read data from file
patient_nr = sampleXMLfile(1:3);
cgm_table = readtable(strcat(patient_nr, '/glucose_level.xlsx'));
meal_table = readtable(strcat(patient_nr, '/meal.xlsx'));
basal_table = readtable(strcat(patient_nr, '/basal.xlsx'));
temp_basal_table = readtable(strcat(patient_nr, '/temp_basal.xlsx'));
bolus_table = readtable(strcat(patient_nr, '/bolus.xlsx'));
```

```
% interpolate cgm data
cgm_time = cgm_table.Var1;
cgm_data = cgm_table.Var2;
time_array = cgm_time(1):seconds(6):cgm_time(end);
y = interp1(cgm_time, cgm_data, time_array, 'linear');

% make table to store data
time_array = transpose(time_array);
interpolated_data = array2table(time_array, 'VariableNames',{'time'});
% fill with zeros when not eating
interpolated_data.carbs = zeros(length(interpolated_data.time), 1);
interpolated_data.meal_type = zeros(length(interpolated_data.time), 1);

% add carbohydrate amount and meal type on correct time slot
for i=1:length(meal_table.Var1)

    ts = meal_table.Var1(i);
    carb = meal_table.Var2_1(i);
    meal = meal_table.Var2_2(i);

    % insert carb amount at where interpolated_data.time = ts
    j = find(interpolated_data.time == ts);
    interpolated_data.carbs(j) = carb;
    interpolated_data.meal_type(j) = meal;

end

% add a column with cgm data
y = transpose(y);
interpolated_data.cgm = y;

% insulin data
% bolus
interpolated_data.bolus = zeros(length(interpolated_data.time), 1);

for i=1:length(bolus_table.Var1)

    ts = bolus_table.Var1(i);
    ts = dateshift(ts, 'start', 'minute', 'nearest');
    bolus_dose = bolus_table.Var2_2(i);

    % insert bolus dose at where insulin_interp.time = ts
    j = find(interpolated_data.time == ts);
    interpolated_data.bolus(j) = bolus_dose;

end
```

```
% basal
interpolated_data.basal = NaN(length(interpolated_data.time), 1);

for i=1:length(basal_table.Var2)

    ts = basal_table.Var1(i);
    ts = dateshift(ts, 'start', 'minute', 'nearest');
    basal_rate = basal_table.Var2(i);

    % insert basal rate at where insulin_interp.time = ts
    j = find(interpolated_data.time == ts);
    interpolated_data.basal(j) = basal_rate;

end

interpolated_data.basal = fillmissing(interpolated_data.basal, 'previous');
first_basal_value = basal_table.Var2(1);
interpolated_data.basal(isnan(interpolated_data.basal)) = first_basal_value;

% temp_basal supersedes basal between ts_1 and ts_2
interpolated_data.temp_basal_and_basal = interpolated_data.basal;

for i=1:length(temp_basal_table.Var2)

    ts_1 = dateshift(temp_basal_table.Var1_1(i), 'start', 'minute', 'nearest');
    ts_2 = dateshift(temp_basal_table.Var1_2(i), 'start', 'minute', 'nearest');

    temp_basal_value = temp_basal_table.Var2(i);

    j1 = find(interpolated_data.time == ts_1);
    j2 = find(interpolated_data.time == ts_2);

    for k=j1:j2
        interpolated_data.temp_basal_and_basal(k) = temp_basal_value;
    end

end

% Convert units of glucose and insulin data
interpolated_data.cgm_mmol = mgdl_to_mmol(interpolated_data.cgm);
interpolated_data.bolus_pmol = units_to_pmol(interpolated_data.bolus);
interpolated_data.basal_pmol = units_to_pmol(interpolated_data.temp_basal_and_basal)/60;
% divide by 60 to get per minute

% Remove unnecessary columns
interpolated_data = removevars(interpolated_data, {'cgm', 'bolus', ...
    'basal', 'temp_basal_and_basal'});

% Save as excel file
writetable(interpolated_data, strcat(patient_nr, '/interpolated/interpolated_data_', ...
    patient_nr, '.xlsx'));
```

Unit conversion (units_to_pmol.m and mgdl_to_mmol.m)

```
function f = units_to_pmol(n)
    f = n*6.94;
end

function f = mgdl_to_mmol(n)
    f = n/18;
end
```

Kalman smoothing (test.m)

This code file uses the Kalman filter for glucose data[37] by Staal.

```
patient_nr = '575';
opts = detectImportOptions(strcat(patient_nr, '/glucose_level.xlsx'))
opts = setvartype(opts,{'Var2'},'double');
table = readtable(strcat(patient_nr, '/glucose_level.xlsx'),opts);

% Smooth 2 and 2 days at a time
indices_559 = [0,516,1092,1640,1971,2527,3074,3488,3940,4509,5040, ...
    5527,6103,6647,7165,7712,8230,8676,9213,9774,10279,10796];
indices_563 = [0,421,997,1573,2113,2684,3260,3785,4356,4929,5217, ...
    5757,6297,6744,7320,7896,8431,8872,9448,10024,10452,11025,11549,12124];
indices_588 = [0,434,1586,2094,2670,3246,3781,4357,4933,5435,6011, ...
    6430,7006,7582,8116,8692,9268,9797,10373,10949,11488,12064,12640];
indices_591 = [0,371,933,1504,2042,2530,3089,3654,4191,4764,5340, ...
    5876,6437,7013,7081,7486,8012,8224,8623,9195,9698,10271,10847];
indices_570 = [0,376,952,1528,2073,2649,3138,3694,4258,4754,5330, ...
    5900,6445,7021,7587,8133,8643,9067,9643,10178,10694,10982];
indices_575 = [0,420,1357,1929,2412,2964,3506,4040,4507,5077,5643, ...
    6114,6678,7226,7695,8248,8679,9237,9772,10239,10791,11346,11866];

switch patient_nr
    case '559'
        indices = indices_559;
    case '563'
        indices = indices_563;
    case '588'
        indices = indices_588;
    case '591'
        indices = indices_591;
    case '570'
        indices = indices_570;
    case '575'
        indices = indices_575;
end

time = [];
cgm_mmol = [];

for i=1:length(indices)-1
    i1 = indices(i)+1;
    i2 = indices(i+1);
    tablenew = table(i1:i2,:);
    y = convertTo_mmol_L(tablenew.Var2); % glucose measurements
    t = tablenew.Var1; % time
    timevector = t(1):seconds(6):t(end);

    smoother_result = SmoothSMBGData(t,y,'outlierRemoval',1, ...
        'dynamicModel',2, 'tout', timevector);

    % Append interpolated data to arrays
    time = [time, timevector];
    cgm_mmol = [cgm_mmol, smoother_result.y_smoothed_at_tout'];

    date = t(1);
    date.Format = 'dd-MMM-yyyy';
    date = datestr(date);

    figure, hold on
    plot(t,y,'r.','MarkerSize',20)
    plot(smoother_result.t_i,smoother_result.y_smoothed,'b-','LineWidth',2);
    plot(smoother_result.t_i,smoother_result.y_smoothed+ ...
        2*smoother_result.y_smoothed_sd,'b--');
    plot(smoother_result.t_i,smoother_result.y_smoothed- ...
        2*smoother_result.y_smoothed_sd,'b--');
    ol = smoother_result.outliers==1;
    plot(t(ol),y(ol),'kx','MarkerSize',10)
    legend('Input glucose measurements','Smoothed glucose','+2 std.dev.', ...
        '-2 std.dev.','Outliers','location','NorthWest')
    title(strcat('Kalman filtered CGM data - patient', patient_nr))
    ylabel('cgm value [mmol/L]')
    x0=100;
    y0=200;
    width=1000;
    height=400;
    set(gcf,'position',[x0,y0,width,height])
    saveas(gcf, strcat(patient_nr, '/kalman_filter_plot_', patient_nr, ...
        '_', date, '.eps'), 'epsc')

end

time = transpose(time);
cgm_mmol = transpose(cgm_mmol);
interp_table = array2table(time, 'VariableNames',{'time'});
interp_table.cgm_mmol_kalman = cgm_mmol;
writetable(interp_table, strcat(patient_nr,'/', patient_nr, ...
    '_cgm_kalman_interpolated.xlsx'));
```

combine_kalman_cgm_and_other_data.m

```
kalman_table = readtable('575/interpolated/575_cgm_kalman_interpolated.xlsx');
table_old = readtable('575/interpolated/interpolated_data_575.xlsx');

% This is extremely slow, takes approx 2 hours
% Add kalman data at the indices where we have timestamps
kalman_cgm = kalman_table.cgm_mmol_kalman;
kalman_time = kalman_table.time;

% Make column to store Kalman data
table_old.cgm_mmol_kalman = NaN(length(table_old.time), 1);

for i=1:length(kalman_cgm)

    ts = kalman_time(i);
    cgm = kalman_cgm(i);
    j = find(table_old.time == ts);
    table_old.cgm_mmol_kalman(j) = cgm;

end

writetable(table_old, '575/interpolated/interpolated_data_with_kalman_575.xlsx');
```

References

[1] G. K. Pal. Gastrointestinal System. In *Comprehensive Textbook of Medical Physiology*, chapter 36-51, pages 327–427. Jaypee Brothers Medical Publishers, 2017.

[2] C. Marling and R. Bunescu. The Ohio T1DM dataset for blood glucose level prediction. *In The 3rd International Workshop on Knowledge Discovery in Healthcare Data, Stockholm, Sweden*, 2018.

[3] E. Mørch-Thoresen. Review of the effect of physical activity on the glucose-insulin regulatory system. *Term project, Norwegian University of Science and Technology (NTNU)*, 2020.

[4] Artificial Pancreas Trondheim. `https://www.apt-norway.com`. [Accessed 25.1.21].

[5] J. Hall. *Guyton and Hall Textbook of Medical Physiology*. Saunders, 12th edition, 2011.

[6] G. K. Pal. Endocrine Pancreas. In *Comprehensive Textbook of Medical Physiology*, chapter 60, pages 527–546. Jaypee Brothers Medical Publishers, 2017.

[7] R. K. Goyal et al. Advances in the physiology of gastric emptying. *Neurogastroenterology and Motility*, 2018.

[8] K. L. Jones et al. Relationships between gastric emptying, intragastric meal distribution and blood glucose concentration in diabetes mellitus. *Journal of Nuclear Medicine*, 1995.

[9] M. Horowitz et al. Gastric emptying in diabetes: clinical significance and treatment. *Diabetic Medicine*, 2002.

[10] G. K. Pal. Physiology of Nutrition. In *Comprehensive Textbook of Medical Physiology*, chapter 161, pages 1282–1294. Jaypee Brothers Medical Publishers, 2017.

[11] Store Norske Leksikon. Carbohydrates. `https://sml.snl.no/karbohydrater_-_ern\T1\aering`, 2020. [Accessed 26.1.21].

[12] Harvard Health Publishing. Glycemic index. `https://www.health.harvard.edu/diseases-and-conditions/glycemic-index-and-glycemic-load-for-100-foods`, 2020. [Accessed 26.1.21].

[13] Glycemic Index Foundation. Low gi explained. `https://www.gisymbol.com/low-gi-explained/`. [Accessed 12.5.21].

[14] A. Cinar and K. Turksoy. *Advances in Artificial Pancreas Systems*. Springer, 2018.

[15] A. El Fathi et al. An Overview of Postprandial Glucose Regulation in Type 1 Diabetes. *IEEE control systems magazine*, 2018.

[16] D. Zeevi et al. Personalized nutrition by prediction of glycemic responses. *Cell Press*, 2015.

[17] F. Chee and T. Fernando. *Closed-Loop Control of Blood Glucose*. Springer, 2007.

[18] R. Bergman. Toward physiological understanding of glucose tolerance: minimal-model approach. *American Diabetes Association*, 1989.

[19] A. Roy. Dynamic modeling of free fatty acid, glucose, and insulin during rest and exercise in insulin dependent diabetes mellitus patients. *University of Pittsburgh, PhD thesis*, 2008.

[20] R. Hovorka et al. Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes. *Physiol. Meas.*, 2004.

[21] C. Dalla Man et al. The UVA/Padova Type 1 Diabetes Simulator Goes From Single Meal to Single Day. *Journal of Diabetes Science and Technology*, 2017.

[22] D. P. F. Moeller. *Mathematical and Computational Modeling and Simulation.* Springer, 2004.

[23] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT press, 2017.

[24] E. D. Lehmann and T. Deutsch. A physiological model of glucose-insulin interaction in type 1 diabetes mellitus. *J. Biomed. Eng.*, 1992.

[25] S. M. Lynch and B. W. Bequette. Model predictive control of blood glucose in type i diabetics using subcutaneous glucose measurements. *Proceedings of the American Control Conference*, 2002.

[26] A. Roy and R. S. Parker. Mixed meal modeling and disturbance rejection in type i diabetic patients. *Proceedings of the 28th IEEE EMBS Annual International Conference*, 2006.

[27] S. Natalucci et al. Glucose absorption and insulin sensitivity from oral glucose tolerance test. *Proceedings of the 25th Annual International Conference of the IEEE EMBS*, 2003.

[28] J. Schirra et al. Gastric emptying and release of incretin hormones after glucose ingestion in humans. *J. Clin. Invest.*, 1996.

[29] C. Dalla Man et al. A system model of oral glucose absorption: Validation on gold standard data. *IEEE Transactions on Biomedical Engineering*, 2006.

[30] M. Rashid et al. Simulation software for assessment of nonlinear and apative multivariable control algorithms: Glucose-insulin dynamics in type 1 diabetes. *Computers and Chemical Engineering*, 2019.

[31] M. E. Fisher. A semiclosed-loop algorithm for the control of blood glucose levels in diabetics. *IEEE Transactions on Biomedical Engineering*, 1991.

[32] T. G. Farmer et al. Effectiveness of intravenous infusion algorithms for glucose control in diabetic patients using different simulation models. *Ind. Eng. Chem. Res.*, 2009.

[33] A. H. Haas and Y. J. W. Rozendaal et al. A Physiology-Based Model Describing Heterogeneity in Glucose Metabolism: The Core of the Eindhoven Diabetes Education Simulator (E-DES). *Journal of Diabetes Science and Technology*, 2015.

[34] Y. J. W. Rozendaal et al. Model-based analysis of postprandial glycemic response dynamics for different types of food. *Clinical Nutrition Experimental*, 2018.

[35] P. G. Fabietti et al. Control oriented model of insulin and glucose dynamics in type 1 diabetics. *Med Biol Eng Comput*, 2006.

[36] L. Lema-Perez et al. Phenomenological-based model of human stomach and its role in glucose metabolism. *Journal of Theoretical Biology*, 2018.

[37] O. M. Staal et al. Kalman Smoothing for Objective and Automatic Preprocessing of Glucose Data. *IEEE Journal of Biomedical and Health Informatics*, 2018.

[38] S. Panunzi, M. Pompa, and A. De Gaetano. A revised Sorensen model: Simulating glycemic and insulinemic response to oral and intra-venous glucose load. *PLOS ONE*, 2020.

[39] M. Wilinska et al. Insulin Kinetics in Type-1 Diabetes: Continuous and Bolus Delivery of Rapid Acting Insulin. *IEEE Transactions on Biomedical Engineering*, 2005.

[40] Medtronic. Why sensor glucose does not equal blood glucose. `https://www.medtronicdiabetes.com/customer-support/sensors-and-transmitters-support/why-sensor-glucose-does-not-equal-blood-glucose`, 2020. [Accessed 6.5.21].

Eira Mørch-Thoresen

Review of the Effect of Meals on the Glucose-Insulin Regulatory System

NTNU

Kunnskap for ei betre verd