Term project
TTK4705 Medical cybernetics

Estimating Blood Glucose Level Based on Subcutaneous Sensors and Analysis of Blood Samples

Martha Halvorsen

Supervisors:
Professor Anders Lyngvi Fougner
PhD candidate Hasti Khoshamadi
PhD candidate Karim Davari Benam

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
DEPARTMENT OF ENGINEERING CYBERNETICS
Trondheim, December 23rd, 2021

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

Faculty of Information Technology
and Electrical Engineering

Department of Engineering
Cybernetics

PROBLEM DESCRIPTION FOR TERM PROJECT

Candidate's name:	Martha Halvorsen
Course:	TTK4550 Teknisk Kybernetikk, Prosjektoppgave
Title (Norwegian):	.
Title (English):	**Estimating blood glucose level based on subcutaneous sensors and analysis of blood samples.**

Description:

In diabetic patients, pancreas is not able to produce insulin and glucagon. Therefore,patients need to check their blood glucose level regularly to inject insulin or glucagon if necessary. Nowadays, blood glucose level is measured by subcutaneous sensors which measures the concentration of the glucose in interstitial fluid instead of directly in blood plasma. There are some other ways of checking blood glucose level such as "fingerpricking" glucose meters, but since they require the users to prick their finger to extract a drop of blood they cannot be used very frequenctly.

In this project, the student will work on using subcutaneous glucose measurements for estimating blood glucose level and other states within the body. This also involves modeling the sensor dynamics, the pharmacodynamics between blood plasma and subcutaneous tissue.

Startup date:	23 August 2021
Submission deadline:	3 January 2022

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

Trondheim, August 2021

Anders Lyngvi Fougner

Contents

List of Figures

List of Tables

Abbriviations

AP	Artifical pancreas
APT	Artificial Pancreas Trondheim
BG	Blood Glucose
BGA	Blood gas analyzer
CGM	Continous glucos monitoring
IG	Interstitium glucose
ISF	Interstitial fluid
KF	Kalman filter
MAE	Mean absolute error
MAPE	Mean absolute percentage error
MSE	Mean squared error
RMSE	Root mean squared error
SMBG	Self monitoring blood glucose
T1D	Type 1 diabetes
T2D	Type 2 diabetes
TF	Transfer function
UIKF	Unknown input Kalman filter

Abstract

Continuous glucose monitor is a popular way of measuring glucose level in people with diabetes. In an artificial pancreas regulation loop the CGM system will inform an insulin pump about the needing amount of insulin. The CGM sensor sits in subcutaneous tissue, hence measures the subcutaneous glucose level, not the actual blood glucose level. The subcutaneous measurements will therefor be affected by the diffusion process in the body between the blood and the subcutaneous tissue which introduces a noticeably delay in glucose level in the subcutaneous tissue. This study aims to reconstruct the blood glucose level, using the subcutaneous glucose measurement, by utilizing different estimation approaches. A standard Kalman filter, as well as an unknown input Kalman filter was researched and implemented in order to achieve this. An inverse transfer function of the diffusion process was already a tested estimation method, and became the competing method to be outperformed. A performance scoring system was also developed, using statistical accuracy measures. The standard Kalman filter shows the most promise in reconstructing the blood glucose level, however like the inverse transfer function, it is highly sensitive to faulty CGM measurement samples, and is tuning wise very aggressive. The unknown input Kalman filter was not successfully implemented, but shows theoretically good capability in estimating the blood glucose. The concluding remarks are that the standard Kalman filter needs adjustments in order to robustly estimate blood glucose, but shows a great deal of promise, while the unknown input filter must be successfully implemented and observed in order to evaluate the actual performance.

1 Introduction

1.1 Diabetes

Diabetes mellitus is a metabolic disorder, that affect millions of people. It is characterized by chronic hyperglycemia in response to ingestion of carbohydrates, fat, and protein, resulting from defects of insulin secretion, insulin action, or both [4]. The disorder can be divided into two conditions: Type 1 Diabetes is caused by an autoimmune reaction resulting in destruction of insulin producing beta-cells, which leads to insulin shortage in the body. T1D require consistent measuring of the BG, and injection of exogenous insulin. Type 2 Diabetes is a result of inefficient use of insulin in the body, due to reduced insulin sensitivity. Adopting a healthier lifestyle is often the first course of action in regards to treating T2D.

1.2 Artificial Pancreas

Artificial pancreas is a type of system which aims to automate insulin delivery in people with diabetes. The system generally consists of a continuous glucose monitor sensor, a control algorithm and an insulin pump. Using closed-loop control the algorithm will, based on a glucose level measurement, calculate how much insulin is needed at the relevant time instant, and communicate this to the insulin pump, which will apply the requested amount. This requires no action from the user. However, the commercially available AP systems are not perfect. In addition to the complicated task of compensating for all meals and activities throughout a day, the general AP must also compensate for the CGM sensor. The CGM sensor is placed in the subcutaneous tissue of a patient, which results in a measure of subcutaneous glucose level. Although the dynamics between the blood glucose level and subcutaneous glucose level are similar, there are some dynamical differences which could lead to non-optimal control of the closed loop control system. In addition, the CGM sensor, like any other sensor, is prone to sensor errors and sensor bias, which also must be compensated for.

1.3 Outline

This report is organized as follows: Section 2 describes relevant background theory about mathematical modeling, control system theory, relevant filters and statistical methods. In section 3 the aim of the study is presented. A description of the data sets used in this project is given in section 4. Section 5 describes the different stages of the project's work, as well as details of the implementation process. Results and additional observations are presented in section 6. A discussion based on the theory and the results will take place in section 7, while a conclusion can be found in section 8. There are also suggestions for future work presented in section 9.

1.4 Artificial Pancreas Trondheim

This thesis is written in collaboration with Artificial Pancreas Trondheim. The artificial pancreas Trondheim research group was established in 2013 at The Norwegian University of Science and Technology, in Trondheim. APT is a cross-disciplinary group of researchers with high competence in the fields of control engineering, biomedical engineering, biosensors, applied clinical research, endocrinology, anesthesia and intensive care medicine, pharmacology, biotechnology, mathematical modelling, biochemistry and chemometrics, as well as collaboration with relevant biosensor industry [5]. The long term goal by the APT group is to develop a robust closed-loop glucose control systems for patients with T1D and T2D.

2 Theory

In this section theory relevant for this project is presented. It will start with a brief presentation of the different sensor systems involved in data sets used in this project's simulations. Next mathematical models of the glucose dynamics in the body, as well as a brief introduction to the glucose regulatory system, will be described. A varied collection of statistical methods for evaluating estimation performance are then introduced. Lastly the selection of methods applied throughout this project are presented, as well as theory relating to discretization and observability.

2.1 Sensor systems

For individuals with diabetes, there are two primary methods for measuring the BG levels; capillary BG measurement, also known as self-monitoring of blood glucose, SMBG, and continuous glucose monitoring, CGM. Both the SMBG and the CGM methods are what is called minimal invasive sensor techniques. Blood gas analyzing of BG is an invasive measurement technique, mostly used in a hospital setting. This paper will focus on the CGM and BGA sensor systems.

2.1.1 Amperometric Sensor

The amperometric glucose sensing technique is based on a chemically-based method of glucose sensing [6]. The sensor estimates glucose by measuring an electrical current generated by the reaction of glucose, either with oxygen or with an immobilized redox mediator [7]. The sensor is in need of calibration, often using capillary blood glucose for this purpose.

2.1.2 Continous Glucose Monitoring

The CGM system consists of a minimally invasive sensor implantable in subcutaneous tissue, a wireless transmitter and a receiver with a display. The sensor part of the CGM system is an amperometric sensor. CGM technique is based on the assumption that a correlation exists between the subcutaneous glucose level and the plasma glucose level. The sensor measures the glucose level in the interstitial fluid that surrounds all cells under the skin, to reflect the plasma glucose level [6].

The assumption of correlation between subcutaneous glucose level and the plasma glucose level is only valid to a certain degree. Whenever the blood glucose concentration is rising or falling, the concentrations will differ with a delay of an average of 10 [min] [6].

Despite being widely used by T1D patients, the sensor is prone to substantial inaccuracies. Factors that affect the sensor accuracy include calibration error, sensor bias, sensor delay and sensor drift [7].

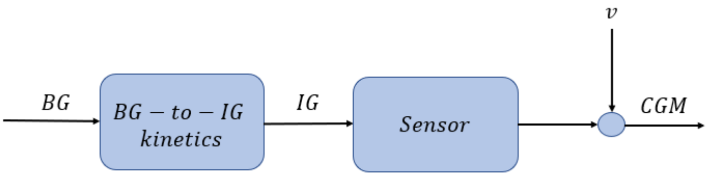


Figure 1: Scheme describing how the CGM data stream is modeled [1].

The diagram in fig. 1 describes how a CGM measurement is obtained. First the BG signal diffuses into the ISF, and is transformed into IG. Then the IG signal is measured by a sensor. The sensed signal is then affected by noise, and the result is the CGM signal. In addition to noise the CGM will also be affected by sensor bias. This bias will be added in the *Sensor*-block in fig. 1. In [8] the bias is modeled as a zero process noise parameter, meaning that it is an unknown constant. This leads to the following sensor equation:

$$y_{cgm,k} = G_{isf} + b_{cgm} + v_{cgm,k}, \tag{1}$$

where b_{cgm} is the bias and $v_{cgm,k}$ is a Gaussian white noise process regarding the non-constant errors of the CGM measurements.

2.1.3 Bloood Gas Analyzer

Blood gas analyzers are used to measure blood gas, pH, electrolytes and a selection of metabolites in whole blood specimens [9]. Among the selection of metabolites it can measure the level of glucose in the blood, hence the analyzer can help determine abnormal glucose levels. In order to do so, the analyzer uses the amperometric measuring technique [10].

The blood used in the analysis is most commonly drawn from the radial artery

[11]. This makes the measuring technique an invasive measuring method, which gives the advantage of enhanced accuracy of the glucose level in the blood [6].

2.2 Mathematical Modeling

2.2.1 Glucose - Insulin regulatory system

In a healthy individual, the blood glucose concentration is narrowly controlled to be between 80 and 90 mg/100 ml during a fasting period. This concentration increases to 120 to 140 mg/100 ml during the first hour or so after a meal, but the feedback systems for control of BG levels restores the glucose concentration back to the fasting period values, usually within two hours after the last absorption of carbohydrates. [2]

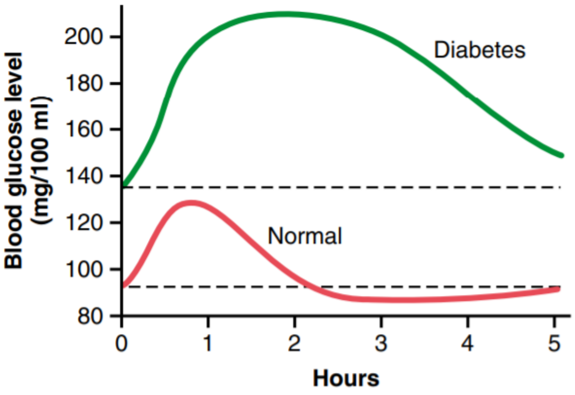


Figure 2: The BG dynamics after a meal for individuals with and without diabetes [2].

There are two main components in this regulatory loop. Namely the hormones glucagon and insulin. Both of which are created in the pancreas. When the glucose concentration rises too high, increased insulin secretion forces the BG concentration to be restored back to a normal value. As the BG levels are decreasing, there will be a rise in glucagon secretion. This will cause an opposite effect, making the BG level rise. Both hormones work together to make the BG level stay around a healthy interval of level variations, in a healthy individual.

2.2.2 Modeling the plasma glucose concentration

The simplest dynamic model that can represent the glucose dynamics is the Rate-only model [8]:

$$\frac{dG_p}{dt} = G_p, \tag{2}$$

where G_p is the plasma glucose concentration.

The model in eq. (2) can be expanded, by dividing the glucose rate into two new states, C_r and C_c. C_r describes a remote compartment, while C_c describes a central compartment. Insulin going into the system will first affect the central compartment. Then, by a first order delay, the affect diffuses over to the remote compartment where it takes effect on the plasma glucose. The state transition equations are [8]:

$$\begin{aligned} \frac{dG_p}{dt}(t) &= C_r(t) \\ \frac{dC_c}{dt}(t) &= -\frac{1}{T_d}C_r(t) \\ \frac{dC_r}{dt}(t) &= \frac{1}{T_d}(C_c(t) - C_r(t)), \end{aligned} \tag{3}$$

where T_d is a time constant, describing the rapidness of flow between the central and remote compartment. T_d also affects how the variance develops. The lower this constant is, the faster the process noise relating to C_c propagates to C_r [8].

Another interpretation of the compartmental states is the description of meal effects when positive, and insulin effects when negative. These two effects are not the only effects that influences the change in BG level. Other processes that may cause increasing or decreasing phenomena in the BG level are also included in the central and remote compartments.

2.2.3 Relationship Between Plasma Glucose and Interstitial Fluid Glucose

In [12] a compartmental approach is used to model the plasma-interstitial fluid glucose dynamics. The dynamic is modeled as a first-order lag, and the mathematical model is:

$$\frac{dC_2}{dt}(t) = -(k_{02} + k_{12})C_2(t) + k_{21}\frac{V_1}{V_2}C_1(t), \tag{4}$$

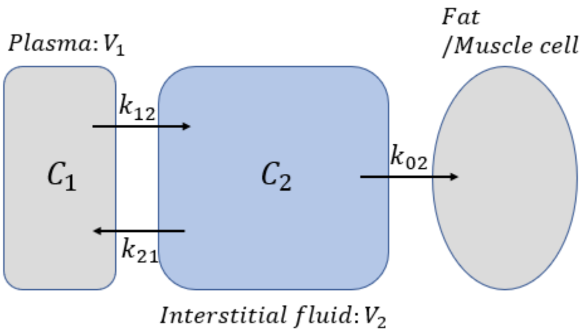


Figure 3: Compartmental representation of capillary plasma and interstitial fluid glucose dynamics.

where k_{02} represents a rate constant for the uptake of glucose in ISF in the subcutaneous tissue, k_{12} and k_{21} are rate constants for diffusion between the plasma and the ISF compartments, V_1 and V_2 are the volumes of the plasma and ISF glucose compartments, respectively. C_2 is the glucose kinetics in the ISF, also known as the second compartment, while C_1 is the glucose kinetics in the plasma, the first compartment. The relationship described by eq. (4) is also visually described by fig. 3. In the reminder of this report $C_2 = G_{isf}$ and $C_1 = G_p$, in order to avoid confusion.

The relationship between the plasma glucose concentration and the ISF glucose concentration can then be transformed into:

$$\frac{dG_{isf}}{dt}(t) = -\frac{1}{T_{isf}}G_{isf}(t) + \frac{g}{T_{isf}}G_p(t), \tag{5}$$

where $g = \left(k_{21}\frac{V_1}{V_2}\right)T_{isf}$ and $T_{isf} = \frac{1}{k_{02}+k_{12}}$. T_{isf} is then the diffusion process time constant, and can be thought of as a constant lag between the two compartments. g functions as the *steady state gain* of the BG-to-IG kinetics. When the system in eq. (5) is at steady state, g equals the ratio $\frac{G_{isf}}{G_p}$ of the steady state concentrations. Physiologically this means that one can expect $g = 1$ [13]. Hence the expression from eq. (5) can be simplified:

$$\frac{dG_{isf}}{dt}(t) = -\frac{1}{T_{isf}}G_{isf}(t) + \frac{1}{T_{isf}}G_p(t). \tag{6}$$

In order to describe the BG-to-IG kinetics, as well as the central and remote compartments of the BG, eq. (3) together with eq. (6) gives the complete mathematical model:

$$\begin{aligned}\frac{dG_p}{dt}(t) &= C_r \\ \frac{dC_c}{dt}(t) &= -\frac{1}{T_d}C_c(t) \\ \frac{dC_r}{dt}(t) &= \frac{1}{T_d}(C_c(t) - C_r(t)) \\ \frac{dG_{isf}}{dt}(t) &= \frac{1}{T_{isf}}(-G_{isf}(t) + G_p(t)),\end{aligned} \tag{7}$$

which can be rewritten into the following system of equations:

$$\begin{bmatrix} \dot{G_p} \\ \dot{C_c} \\ \dot{C_r} \\ \dot{G_{isf}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -\frac{1}{T_d} & 0 & 0 \\ 0 & \frac{1}{T_d} & -\frac{1}{T_d} & 0 \\ \frac{1}{T_{isf}} & 0 & 0 & -\frac{1}{Tisf} \end{bmatrix} \begin{bmatrix} G_p \\ C_c \\ C_r \\ G_{isf} \end{bmatrix}. \tag{8}$$

2.3 Statistical Analysis

When a model or a method is specified, the corresponding performance characteristics should to be verified or validated by comparison with historical data from the process it was designed to predict.

Accuracy is the most important criterion when evaluating a model in an estimation related setting. Hence the goal of the relevant model is to minimize the estimation error between the estimate and the actual real value. The error is defined as follows

$$e_i = y_i - \hat{y}_i, \tag{9}$$

where y_i is the actual real value at iteration number i, and $\hat{y}_i$ is the estimate at time iteration number i.

All presented methods for evaluating accuracy, and their explanations are taken from [14].

Mean Absolute Error

Mean absolute error measures an average magnitude of all errors in a set of forecasts, ignoring the direction of the errors. This can be seen as all errors are assigned equal weights. The closer the MAE is to zero, the better the method evaluated fits the past time series. If MAE is shown to have a larger value, it would indicate that the method evaluated is a poor fit for the past time series.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |e_i|, \tag{10}$$

where n is the number of samples used, and e_i is the error for sample number i. This will be the same for all following methods described in this section.

Mean squared error

The mean squared error is very similar to the MAE, but in this measure the error is *squared*. This results in that larger errors are given more weight, meaning that larger errors are more penalized than smaller errors. The MSE is evaluated the same way as the MAE. The MSE is given in the following equation.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} e_i^2, \tag{11}$$

Root mean squared error

An alternative to the MSE is the root mean squared error. The RMSE takes the square root of the sum of MSE. The seriousness of the estimation error is then denoted in the same dimension as the actual value and the estimate themselves. The change of dimension does not change how the metric is evaluated, hence RMSE is evaluated the same way as one evaluates both MAE and MSE. The RMSE is given in the following equation:

$$\text{RMSE} = \sqrt{\text{MSE}}. \tag{12}$$

Mean absolute percentage error

The mean absolute percentage error corresponds relatively to the MAE, but gives the accuracy a percentage and expresses the relative error. If the MAPE value is less than 10% the estimate is considered to be very accurate. As the percentage increases, the accuracy worsens. A MAPE value between $20-50\%$ is considered acceptable, while when the MAPE $> 50\%$ the estimate is considered inaccurate. The following described the computation of MAPE:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|e_i|}{y_i}100\%. \tag{13}$$

2.4 Discretization

A computer cannot process continuous time, hence discretization is a necessary step for computer simulation. A linear time invariant model consists of continuous differential equations, which when to be used in numerical computing must be transformed into their discrete difference equation counterparts.

Consider the linear time invariant system:

$$\begin{aligned} \dot{x} &= Ax(t) + Bu(t) \\ y(t) &= Hx(t) + Du(t), \end{aligned} \tag{14}$$

where x is the state vector, y is the system output, u is the system input and A, B, C and D are the system matrices.

Under the assumption that the system input is constant during a given sampling time and that A is invertible, the above system will then have the *exactly discretized model* as follows:

$$\begin{aligned} x_{k+1} &= A_d x_k + B_d u_k \\ y_k &= H_d x_k + D_d u_k, \end{aligned} \tag{15}$$

where

$$\begin{aligned}
A_d &= e^{A\Delta t} \\
B_d &= \left(\int_{k\Delta t}^{(k+1)\Delta t} e^{A[(k+1)\Delta t - \tau]} d\tau \right) B \\
H_d &= H \\
D_d &= D
\end{aligned} \tag{16}$$

and Δt is the sampling time [15].

2.5 Observability

Observability is a way of determine how well the internal states of a system can be inferred from knowledge of the external outputs. In an estimation problem, observability is an absolute demand.

The following system:

$$\begin{aligned}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t) + Du(t),
\end{aligned} \tag{17}$$

is said to be completely observable if for any unknown initial state $x(t_0) = x_0$ there exists a finite time $t_1 > t_0$ such that the input $u(t)$ and output $y(t)$ over $t_0 \leq t \leq t_1$ are sufficient information to uniquely determine the initial condition x_0 [16].

Assuming $u(t_0) = 0$ the following expression for $y_0(t)$ is obtained:

$$y_0(t) = Ce^{At}x_0. \tag{18}$$

The n dimensional linear time invariant system (A, C) is observable if and only if the observability matrix

$$O = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}, \tag{19}$$

has full rank:

$$rank(O) = n. \tag{20}$$

2.6 Standard Kalman Filter

The Kalman Filter is an optimal estimator that provides a recursive computational methodology for estimating the state of a discrete-data controlled process from measurements, while also providing an estimate of the uncertainty of the estimates [17]. A figure showing the filter connected to a system can be found in fig. 4.

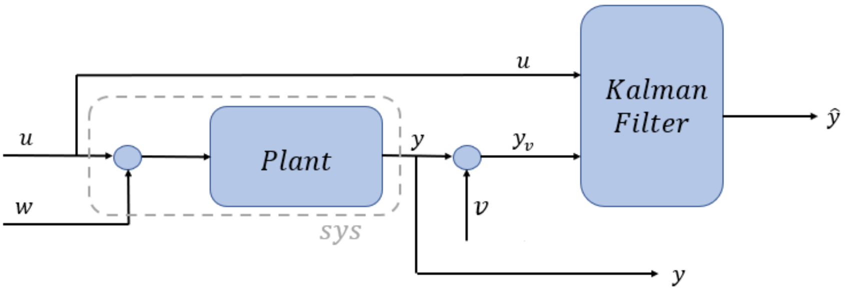


Figure 4: System block diagram with KF.

Given the linear discrete-time system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Hx_k + v_k \end{aligned} \tag{21}$$

where x_k are the states at time k, A is the state transition matrix, u_k is the input at time k, B is the input transition matrix, y_k is the measurement vector and H is the measurement matrix.

The noise processes w_k and v_k are considered white, zero-mean, uncorrelated and have known covariance matrices Q_k and R_k, respectively:

$$\begin{aligned} w_k &\sim (0, Q_k) \\ v_k &\sim (0, R_k) \\ E[w_k w_j^T] &= Q_k \delta_{k-j} \\ E[v_k v_j^T] &= R_k \delta_{k-j} \\ E[v_k w_j^T] &= 0 \end{aligned} \tag{22}$$

where δ_{k-j} is the Kronecker delta function. The Kronecker delta function gives $\delta_{k-j} = 1$ if $k = j$, and $\delta_{k-j} = 0$ if $k \neq j$ [18]. Q_k is the process noise covariance at time k, and R_k is the measurement noise covariance at time k.

The goal of the standard Kalman filter is then to estimate the state x_k based on the knowledge of the system dynamics and the availability of measurements, y_k.

The Kalman filter manages this by combining *a posteriori* and *a priori* estimates of the state x_k. The *a posteriori* estimate is the expected value of x_k conditioned on all of the measurements up to and including time k.

$$\hat{x}_k = E[x_k | y_1, y_2, ..., y_k] = \textit{a posteriori} \text{ estimate.} \tag{23}$$

The *a priori* estimate on the other hand is the expected value of x_k conditioned on all of the measurements prior to time k.

$$\bar{x}_k = E[x_k | y_1, y_2, ..., y_{k-1}] = \textit{a priori} \text{ estimate.} \tag{24}$$

The estimation error covariance matrix of the *a posteriori* estimate at time k is denoted by P_k, while the *a priori* estimation error covaraince matrix at time k is denoted by $\bar{P}_k$.

$$\begin{aligned} P_k &= E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] \\ \bar{P}_k &= E[(x_k - \bar{x}_k)(x_k - \bar{x}_k)^T]. \end{aligned} \tag{25}$$

Assuming a prior estimate $\bar{x}$, the aim is to use the measurement y_k to improve the posterior estimate:

$$\hat{x}_k = \bar{x}_k + K_k(y_k - H\bar{x}_k), \tag{26}$$

where K_k is the Kalman gain. In order to determine K_k the minimum mean-square error is used as the performance criterion [19]. Consider the covariance matrix of the estimate, which now can be rewritten using eq. (21) and eq. (26):

$$\begin{aligned} P_k &= E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] \\ P_k &= E[((x_k - \bar{x}_k) - K_k(Hx_k + v_k - H\bar{x}_k)) \\ &\quad ((x_k - \bar{x}_k) - K_k(Hx_k + v_k - H\bar{x}_k))^T]. \end{aligned} \tag{27}$$

Recognize that $(x_k - \bar{x}_k)$ is the *a priori* estimation error, and that it is uncorrelated with the measurement error v_k. Performing the indicated expectation then yields:

$$P_k = (I - K_k H)\bar{P}_k (I - K_k H)^T + K_k R_k K_k^T. \tag{28}$$

Now this expression is expanded into:

$$P_k = \bar{P}_k - K_k H \bar{P}_k - \bar{P}_k H^T K_k^T + K_k (H \bar{P}_k H^T + R_k) K_k^T, \tag{29}$$

The goal is then to minimize the trace of P, since it is the sum of the MSE in the estimates of all the elements of the state vector [19]. This is done by differentiating the trace of P_k with respect to K_k. Note that the trace of $\bar{P}_k H_k^T K_k^T$ is equal to the trace of it's transpose $K_k H \bar{P}_k$. This gives:

$$\frac{d(\text{trace} P_k)}{dK_k} = -2(H\bar{P}_k)^T + 2K_k(H\bar{P}_k H^T + R_k). \tag{30}$$

Setting the derivative equal to zero and solving for the optimal gain gives:

$$K_k = \bar{P}_k H^T (H \bar{P}_k H^T + R_k)^{-1}. \tag{31}$$

By routine substitution of the optimal gain in eq. (31) into eq. (29) the following result is obtained:

$$P_k = (I - K_k H)\bar{P}_k. \tag{32}$$

As the first measurement is taken at time $k = 1$, the initial value of the *a posteriori* estimate, $\hat{x}_0$, is set equal to the expected value of the initial state x_0.

$$\hat{x}_0 = E[x_0]. \tag{33}$$

Now, given all of the above, the algorithm of the standard KF can be formulated.

The dynamical system is described by the following equations [18]:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Hx_{k+1} + v_k \\ E[w_k w_j^T] &= Q_k \delta_{k-j} \\ E[v_k v_j^T] &= R_k \delta_{k-j} \\ E[v_k w_j^T] &= 0 \end{aligned} \tag{34}$$

The KF is initialized as follows:

$$\begin{aligned} \hat{x_0} &= E[x_0] \\ P_0 &= E[(x_0 - \hat{x_0})(x_0 - \hat{x_0})^T] \end{aligned} \tag{35}$$

For time step $k = 1, 2, ...$ the standard KF is given by the following equations:

$$\begin{aligned} &\text{Prediction:} \\ &\qquad \bar{x}_k = A\hat{x}_{k-1} + Bu_{k-1} \\ &\qquad \bar{P}_k = AP_{k-1}A^T + Q_k \\ &\text{Correction:} \\ &\qquad K_k = \bar{P}_k H^T (H\bar{P}_k H^T + R_k)^{-1} \\ &\qquad \hat{x}_k = \bar{x}_k + K_k(y_k - H\bar{x}_k) \\ &\qquad P_{k+1} = (I - K_k H)\bar{P}_k. \end{aligned} \tag{36}$$

2.7 Unkown Input Klaman Filter

When estimating the state of a dynamical system using a Kalman Filter it is generally assumed that all system parameters, noise covariances and inputs are known. However, in practice there exists many systems in which these assumptions does not hold. Under those circumstances the Kalman Filter technique may not find the optimal estimate, due to the uncertainties in the assumed known parameters.

A solution to this is presented in [20], where an Unknown Input Kalman Filter is derived. The problem is formulated as state estimation of stochastic singular systems with no assumptions about the unknown inputs. The rest of this sub section is devoted to describing the suggested method. A figure describing the connection between the UIKF and a system can be found in fig. 5

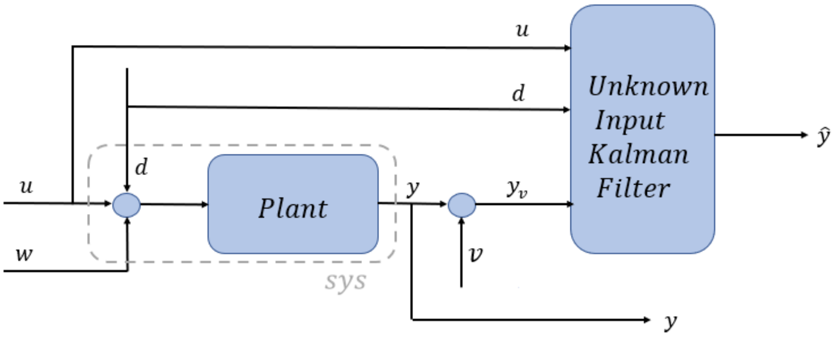


Figure 5: System block diagram with UIKF.

Assume the following linear discrete-time system:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Fd_k + w_k, \\ y_k &= Hx_k + v_k, \end{aligned} \tag{37}$$

where x_k are the states at time k where $x_k \in \mathbb{R}^n$, A is the transition matrix, u_k are the known input at time k, B is the input transition matrix, d_k is the unknown input at time k where $d_k \in \mathbb{R}^q$ and F is the unknown input transition matrix. y_k is the measurement of time k where $y_k \in \mathbb{R}^p$, and H is the measurement matrix.

The noise processes w_k and v_k follow the same assumptions as for the standard Kalman filter (see section 2.6), hence

$$\begin{aligned} w_k &\sim (0, Q_k) \\ v_k &\sim (0, R_k) \\ E[w_k w_j^T] &= Q_k \delta_{k-j} \\ E[v_k v_j^T] &= R_k \delta_{k-j} \\ E[v_k w_j^T] &= 0, \end{aligned} \tag{38}$$

where Q_k is the process noise covariance at time k, and R_k is the measurement noise covariance at time k.

The unknown input Kalman filter aims to estimate the state x_k and the unknown input d_k, assuming no knowledge about d_k. This is only possible under the following assumptions:

- Assumption 1 - $\text{rank}(H) = p$
- Assumption 2 - $\text{rank}(F) = q$
- Assumption 3 - $q \leq p$
- Assumption 4 - $\text{rank}(HF) = q$

The system described in eq. (37) can be formulated as the singular system given in eq. (39). A system is called singular if it is described on the matrix form $E\dot{x} = Ax + Bu$, and E is a singular matrix [21].

$$\begin{aligned} \boldsymbol{E}\boldsymbol{X}_{k+1} &= \boldsymbol{A}\boldsymbol{X}_k + Bu_k + w_k \\ y_k &= \boldsymbol{H}\boldsymbol{X}_k + v_k, \end{aligned} \tag{39}$$

where $\boldsymbol{X}_k = \begin{bmatrix} x_k \\ d_{k-1} \end{bmatrix}$, $\boldsymbol{E} = \begin{bmatrix} I & -F \end{bmatrix}$, $\boldsymbol{A} = \begin{bmatrix} A & 0 \end{bmatrix}$ and $\boldsymbol{H} = \begin{bmatrix} H & 0 \end{bmatrix}$.

The original state and unknown input estimation problem is now reduced to estimation of the semi-states of the singular system given in eq. (39).

From the singular system the following result can be obtained

$$\begin{aligned} \text{rank}\left(\begin{bmatrix} \boldsymbol{E} \\ \boldsymbol{H} \end{bmatrix}\right) &= \text{rank}\left(\begin{bmatrix} I & -F \\ H & 0 \end{bmatrix}\right) \\ &= \text{rank}\left(\begin{bmatrix} I & 0 \\ H & I \end{bmatrix}\begin{bmatrix} I & -F \\ H & 0 \end{bmatrix}\right) \\ &= \text{rank}\left(\begin{bmatrix} I & -F \\ 0 & HF \end{bmatrix}\right) \\ &= n + \text{rank(HF)} \\ &= n + q. \end{aligned}$$

Now consider the following stochastic singular system

$$\begin{aligned} Ex_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Hx_k + v_k, \end{aligned} \tag{40}$$

where $x_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^r$, $y_k \in \mathbb{R}^m$ and $E \in \mathbb{R}^{(p,n)}$, and assume that rank $\left(\begin{bmatrix} E \\ H \end{bmatrix}\right)$ $= n$.

Under all the above assumptions, the optimal state estimator for the singular system in eq. (40) is given by the following recursion:

$$\hat{x}_{k+1} = P_{k+1}(E^T(Q + AP_kA^T)^{-1}(A\hat{x}_k) + H^TR^{-1}y_{k+1}) \tag{41}$$

where $\hat{x}_k$ is the estimate of x_k based on the measurement at time instant k, and

$$P_k = E[(\hat{x}_k - x_k)(\hat{x}_k - x_k)^T] \tag{42}$$

is the estimation error covariance matrix, subject to the following *Generalized Riccati Difference Equation*:

$$P_{k+1} = (E^T(Q + AP_kA^T)^{-1}E + H^TR^{-1}H)^{-1}. \tag{43}$$

The estimate for both the states and the unkown input, based on the measurement, up to time k is given by $\hat{\boldsymbol{X}}_k = \begin{bmatrix} \hat{x}_k \\ \hat{d}_k \end{bmatrix}$, and the estimation covariance matrix is given by $\boldsymbol{P}_k = E[(\hat{\boldsymbol{X}}_k - \boldsymbol{X}_k)(\hat{\boldsymbol{X}}_k - \boldsymbol{X}_k)^T]$ which is partitioned as follows

$$\boldsymbol{P}_k = \begin{bmatrix} P_k^x & P_k^{xd} \\ P_k^{dx} & P_k^d \end{bmatrix}, \tag{44}$$

where

$$P_k^x = E[(\hat{x}_k - x_k)(\hat{x}_k - x_k)^T] \tag{45}$$

is the state estimation error covariance matrix,

$$P_k^d = E[(\hat{d}_k - d_{k-1})(\hat{d}_k - d_{k-1})^T] \tag{46}$$

is the unknown input estimation error covariance matrix,

$$P_k^{xd} = E[(\hat{x}_k - x_k)(\hat{d}_k - d_{k-1})^T] \tag{47}$$

is the cross state and unknown input estimation errors covariance matrix, where $P_k^{xd} = (P_k^{dx})^T$.

From eq. (41) and eq. (43), as well as the definition of $\boldsymbol{E}$, $\boldsymbol{A}$ and $\boldsymbol{H}$ the following is obtained

$$\hat{\boldsymbol{X}}_{k+1} = \boldsymbol{P}_{k+1} \begin{bmatrix} I \\ -F^T \end{bmatrix} \bar{P}_k^{-1} \bar{x}_k + \boldsymbol{P}_{k+1} \begin{bmatrix} H^T R^{-1} \\ 0 \end{bmatrix} y_{k+1} \tag{48}$$

where

$$\begin{aligned} \boldsymbol{P}_{k+1}^{-1} &= \begin{bmatrix} \bar{P}_k^{-1} + H^T R^{-1} H & -\bar{P}_k^{-1} F \\ -F^T \bar{P}_k^{-1} & F^T \bar{P}_k^{-1} F \end{bmatrix} \\ \bar{x}_k &= A \hat{x}_k + B u_k \\ \bar{P}_k &= A P_k^x A^T + Q. \end{aligned} \tag{49}$$

Equation eq. (48) yields

$$\begin{aligned} \hat{x}_{k+1} &= (P_{k+1}^x - P_{k+1}^{xd} F^T) \bar{P}_k^{-1} \bar{x}_k + P_{k+1}^x H^T R^{-1} y_{k+1} \\ \hat{d}_{k+1} &= (P_{k+1}^{dx} - P_{k+1}^d F^T) \bar{P}_k^{-1} \bar{x}_k + P_{k+1}^{dx} H^T R^{-1} y_{k+1}. \end{aligned} \tag{50}$$

Now using the inverse of the partitioned matrix $\boldsymbol{P}_{k+1}$ the following set of difference equations i obtained

$$\begin{aligned} P_{k+1}^x &= (\bar{P}_k^{-1} + H^T R^{-1} H - \bar{P}_k^{-1} F (F^T \bar{P}_k^{-1} F)^{-1} F^T \bar{P}_k^{-1})^{-1} \\ P_{k+1}^d &= (F^T H^T (V + H \bar{P}_k H^T)^{-1} H F)^{-1} \\ P_{k+1}^{xd} &= P_{k+1}^x \bar{P}_k^{-1} F (F^T \bar{P}_k^{-1} F)^{-1} \\ P_{k+1}^{dx} &= P_{k+1}^d F^T \bar{P}_k^{-1} (\bar{P}_k^{-1} + H^T R^{-1} H)^{-1}. \end{aligned} \tag{51}$$

The above equations gives the following:

$$\begin{aligned} (P^x_{k+1} - P^{xd}_{k+1}F^T)\bar{P}_k^{-1} &= I - P^x_{k+1}H^T R^{-1} H \\ (P^{dx}_{k+1} - P^{d}_{k+1}F^T)\bar{P}_k^{-1} &= -P^{dx}_{k+1}H^T R^{-1} H \\ P^x_{k+1} &= (\bar{P}_k^{-1} + H^T R^{-1} H)^{-1} + P^{xd}_{k+1}(P^{dx}_{k+1})^{-1}P^{dx}_{k+1}. \end{aligned} \tag{52}$$

By then inserting eq. (52) in eq. (50) the following is obtained:

$$\begin{aligned} \hat{x}_{k+1} &= \bar{x}_k + P^x_{k+1}H^T R^{-1}(y_{k+1} - H\bar{x}_k) \\ \hat{d}_{k+1} &= P^{dx}_{k+1}H^T R^{-1}(y_{k+1} - H\bar{x}_k). \end{aligned} \tag{53}$$

Then by substituting for P^x_{k+1} in eq. (53), and completing resulting calculations one obtains the optimal state and unknown input estimates:

$$\begin{aligned} \bar{x}_k &= A\hat{x}_k + Bu_k \\ \hat{x}_{k+1} &= \bar{x}_k + F\hat{d}_{k+1} + K^x_{k+1}(y_{k+1} - H(\bar{x}_k + F\hat{d}_{k+1})) \\ \hat{d}_{k+1} &= K^d_{k+1}(y_{k+1} - H\bar{x}_k), \end{aligned} \tag{54}$$

where

$$\begin{aligned} K^x_{k+1} &= (\bar{P}_k^{-1} + H^T R^{-1} H)^{-1} H^T R^{-1} \\ K^d_{k+1} &= P^{dx}_{k+1} H^T R^{-1} \end{aligned} \tag{55}$$

and where $\bar{P}_k^{-1}$ and P^{dx}_{k+1} are given by eq. (49) and eq. (51).

2.8 Inverse Transfer Function

A transfer function is a theoretical function representing the relationship between input signals and output signals of a system. Consider the state space form:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx, \end{aligned} \tag{56}$$

where x is the states of the system, u is the input and y is the output. A, B and C are the system matrices.

The transfer function in the *Laplace* domain from input u to output y is then given by [22]:

$$\frac{y(s)}{u(s)} = C(sI - A)^{-1}B = G(s), \tag{57}$$

This system is presented visually in fig. 6.

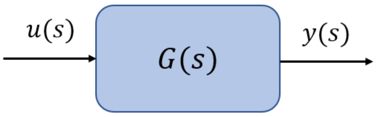


Figure 6: TF from $u(s)$ to $y(s)$.

If the reveres relationship is to be obtained, it will have the following form:

$$\frac{u(s)}{y(s)} = \frac{1}{G(s)}. \tag{58}$$

See fig. 7 for a visual representation.

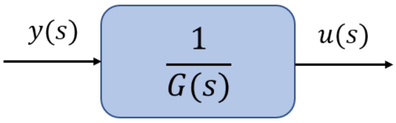


Figure 7: TF from $y(s)$ to $u(s)$.

The inverse of the TF from eq. (58) is only possible if the system has more poles than finite zeros. In other words $\frac{1}{G(s)}$ must be causal. A workaround in the situation that $\frac{1}{G(s)}$ is in fact not causal is to approximate it to some causal system.

One approximation of $\frac{1}{G(s)}$ is:

$$\frac{1}{G(s)} \approx \frac{\alpha G(s)}{1 + \alpha G(s)^2}, \tag{59}$$

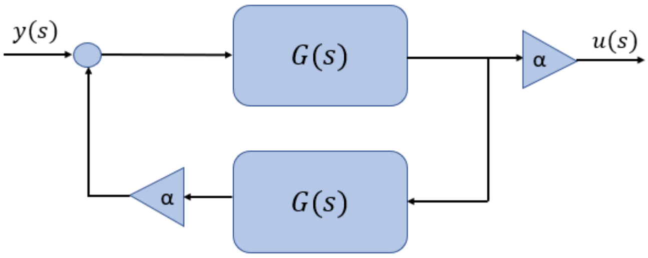


Figure 8: Approximated TF from $u(s)$ to $y(s)$.

where α is considered the close loop gain and $\alpha G(s)^2 >> 1$.

This gives the following transfer function:

$$\frac{u(s)}{y(s)} = \frac{\alpha G(s)}{1 + \alpha G(s)^2} \approx \frac{1}{G(s)}, \tag{60}$$

which is visualized in fig. 8.

3 Aim of the project

The aim of this project is to use subcutaneous glucose measurements for estimating blood glucose. This implies modeling of the sensor dynamics, the pharmacodynamics between blood plasma and subcutaneous tissue. The estimation method has to be able to be applicable in a real-time setting.

An already established method have been tested during previous work, relating to blood glucose reconstruction using subcutaneous glucose measurements. One of the goals of the methods developed trough this project is therefor to outperform this already established method.

The aim of the project can be divided into the following elements:

- Finding a potential model sufficiently describing the relevant physiological processes, through a literature search.
- Finding potential method candidates for blood glucose estimation, through a literature search.
- Implementing the potential model and method candidates.
- Evaluate performance of method candidates.
- Discuss and compare the different method candidates, as well as the model.
- Review the possibility of blood glucose reconstruction in a real-time setting.

4 Data aquisition

The data used for the simulations in this project is collected trough animal experiments performed between 2018-2021, by APT research group. The animals participating in the experiments were pigs.

The subcutaneous sensor used was the Medtronic Enlite sensor, which is assumed to be related to the ISO 15197:2015 standard. This sensor was paired with an Inreada transmitter, which had a sampling time of 1.2 [sec], in order to make a CGM system. The CGMs system measures the IG level of the pigs in the experiments. Two CGM sensors were attached to the pigs during each experiment. The reason for this was redundancy. The BG was measured using a ABL800 FLEX analyzer, which is a BGA system.

A number of three data sets were chosen at random to be used in this project's work. The chosen data sets where from the experiments performed on the 20th Nov 2020, the 1st Mar 2021, and the 5th Mar 2021. On Nov 20th 2020 the duration of the experiment was approximately 12 hours, on Mar 1st and 5th 2021 the duration was approximately 24 hours.

5 Method description and implementation

This section will describe the process of researching and developing a model of the BG-to-IG kinetics as well as relevant methods for BG reconstruction based on IG levels. The code for running the simulations will also also be explained. It will be sectioned according to different stages of the work-process of this project.

A code file was given at the start of the project. This included the process of reading in and structuring the data from the data sets, as well as an already tried approach. This will not be elaborated on in detail. The function `convertToRelativeTime.m` used in this project is borrowed from an open Git repository, and will not be explained in great detail. This includes all the code up until code line 69 in the `Matlab` file `main.m`, as well as `invTF.m` and `convertToRelativeTime.m`. A finial version of the code can be found in Appendix A.

The block diagrams found in this report is made with the program PowerPoint, which is part of the Microsoft Office software package. This includes fig. 3, fig. 6, fig. 7, fig. 8, fig. 4 and fig. 5.

5.1 Initial problem set up

A hand-out code was provided by one of the projects supervisors at the start of the project. The programming language in this code was `Matlab`, hence this is the programming language used in the developed project code. The hand-out code provided a solid foundation, where reading in data, structuring the data, and plotting mechanisms were already in place.

To understand the problem, the BGA measurement array was plotted against the CGM measurement array. The BGA was in the inital hand-out code plotted using dots, this was changed to be a continuous line. A comparison between the two measurements were made, and a problem to be solved was formulated.

An already tried approach for reconstructing BG was also included in the hand-out code. Namely the inverse TF method. This method uses the BG-to-IG kinetics, see eq. (6), and reverses it, in the *Laplace domain*, in order to obtain a BG estimate, where the IG is then the system input.

The transfer function from G_p to G_{isf} can by assuming that the kinetics are linear, and that parameters are time-invariant, be modeled as a first order transfer function [1]:

$$\frac{G_p(s)}{G_{isf}(s)} = \frac{b}{\frac{1}{T_{isf}}s+1} = \frac{bT_{isf}}{s+T_{isf}} = \frac{B}{s+a}, \tag{61}$$

where $a = T_{isf}$ is the pole of the system, T_{isf} is the diffusion process time constant and $B = bT_{isf}$ where b is considered a gain originating from the calibration errors of the CGM sensor. Inverting this system will make $\frac{1}{G(s)}$ a non causal system, hence the approximation introduced in section 2.8 is used:

$$\frac{G_{isf}(s)}{G_p(s)} = \frac{\alpha B(s+a)}{(s+a)^2+\alpha B} = \frac{\alpha Bs+\alpha Ba}{s^2+2as+(a^2+\alpha B^2)}, \tag{62}$$

where α is the closed loop gain of the inverse transfer function for the BG-to-IG kinetics.

In order to understand the workings of this method, it was tuned, and the response was observed. This method can be found in Appendix A, under the function name `invTF(...)`.

5.2 Literature search

The search engine used for finding literature was Google Scholar. Searching for relevant literature was structured according to the problems to be solved, namely:

- Finding an appropriate mathematical model for describing the physiological process
- Finding a suitable method for estimation
- Finding a way of evaluating the fit of the method/model

Finding the appropriate model was the starting point. The typical search words used for this part of the literature search was ”diffusion”, ”plasma and subcutaneous tissue glucose dynamics” and ”mathematical modeling glucose”.

Once a model was determined, the search proceeded into finding a suitable method. Kalman filters was a natural first step in investigating appropriate estimation methods. Very early on it was also suggested that an unknown input Kalman filter might be a suitable choice, hence this was the predominately used search word. Other search words used was ”Kalman filter blood glucose”, ”blood glucose estimation”, ”blood glucose reconstruction using subcutaneous sensor”.

Having found literature on suitable methods, a way of evaluating them was the next step in the literature search. This part of the search consisted of looking into various statistical methods, and finding the most suitable for evaluating accuracy.

5.3 Implementation

5.3.1 Mathematical Implementation

Based on the literature search, the glucose-interstitial fluid dynamic model of choice was the compartmental model presented in 2.2.3, and derived in eq. (7). The model needed to be rewritten into a state space form, in order to be used by Kalman filters, see eq. (21). This gives the state vector $x = \begin{bmatrix} G_p & C_c & C_r & G_{isf} \end{bmatrix}^T$, and the system matrix A derived in eq. (8). No known input was set to affect the system, hence B is not present in the model. Since only the CGM measurement would be available to use in the simulations the measurement matrix H, had to be in accordance with this knowledge. This gives the following result:

$$\begin{aligned} \dot{x} &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -\frac{1}{T_d} & 0 & 0 \\ 0 & \frac{1}{T_d} & -\frac{1}{T_d} & 0 \\ \frac{1}{T_{isf}} & 0 & 0 & -\frac{1}{Tisf} \end{bmatrix} \begin{bmatrix} G_p \\ C_c \\ C_r \\ G_{isf} \end{bmatrix} \\ y &= \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} G_p \\ C_c \\ C_r \\ G_{isf} \end{bmatrix} . \end{aligned} \tag{63}$$

eq. (63) was an appropriate inputless model when using the standard Kalman filter. When using the unkown input Kalman filter, a modification had to be made, namely the inclusion of F and d from eq. (54). The unknown input was set to affect all the states. This results in the following system:

$$
\begin{aligned}
\dot{x} &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -\frac{1}{T_d} & 0 & 0 \\ 0 & \frac{1}{T_d} & -\frac{1}{T_d} & 0 \\ \frac{1}{T_{isf}} & 0 & 0 & -\frac{1}{Tisf} \end{bmatrix} \begin{bmatrix} G_p \\ C_c \\ C_r \\ G_{isf} \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 1 \end{bmatrix} d \\
y &= \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} G_p \\ C_c \\ C_r \\ G_{isf} \end{bmatrix}.
\end{aligned}
\tag{64}
$$

For both filters the noise covariance matrices Q_k and R_k was assumed constant throughout the entire simulation time. Hence $Q_k = Q$ and $R_k = R$. The time constants from eq. (7) needed to be determined before implementing the model in `Matlab`. T_d was set to be 10 [min]. T_{isf} was set to 12 [min]. The sensor bias, relating to eq. (1) was set to be 0.01 [mmol/L].

5.3.2 Code Implementation

A great deal of inspiration for the code is taken from [23], which is the open Git Repository for the code developed in [8]. The function named `convertToRelativeTime(...)` is borrowed from this Git repository. This function was deemed necessary due to the difference in sampling time between the BGA measurement vector and the CGM measurement time vector. Converting to relative time was an easy workaround when t_{BGA} and t_{CGM} was of different lengths and using `date time` as their data types.

The model was built as an object, with the system matrices as properties. This was done for each Kalman filter individually, and the functions are named `setDynModel(...)`. As the model in eq. (7) is defined in continuous time, it was necessary to discretize it. This was done in accordance with the discretization process described in section 2.4, and using the `Matlab` function `expm(...)`. Notice that A is not invertible, however this is handled in the discretization process by `expm(...)` which uses the Taylor series approximation in order to find the matrix exponential. The discretized system transition matrix is called `Phi` in the model. CGM would be the only available measurement to use in the simulations, hence an observability check, see section 2.5 is also done in the same function. For this the `Matlab` inbuilt function `obsv(...)` was used.

When building the standard Kalman filter it was decided to do so in a function. The function is named `standardKF(...)`, and can be found in Appendix A. The first thing done in the function is to build the model, as described above. All system matrices, and their values can be found in table 2, and the connected model parameters can be found in table 1. After building the model the time t is converted to relative time, so the filter output time can be used in the plots. The filter equations, see eq. (36), are then put in a for-loop, which iterates trough time vector t. At $t = 1$ the inital value of the estimates for G_p and G_{isf} is set to $y_{cgm} - b$, in order to eliminate the sensor bias, while the initial value of C_c and C_r is set to 0. Once the for-loop is finished an output object is built, where the estimated states are set as the object's properties, in addition to the G_p covariance matrix, P_{G_p} and the relative time vector.

The unknown input Kalman filter function is built much the same way as `standardKF(...)`, but has some modifications. The function is named `unknownInputKF(...)`, and can be found in Appendix A. Since the UIKF deals with an unknown input, the unknown input system matrix had to be added to the model. This matrix is named F, see table 2. Once the model has been set, the assumptions presented in section 2.7 are checked, using the function `checkAssumptions(...)`. The filter equations, see eq. (54), eq. (49) and eq. (51), are then set in a for-loop which iterates through t. The reminder of the filter function is performed equal to that of the standard Kalman filter.

Next step was to develop some form of performance score. This was done in the `computeError(...)` function. BGA was to be used as the ”real” value when calculating the error. Due to the difference in t_{BGA} and t_{CGM} a method for finding corresponding points in time had to be developed. This was done by finding the minimum value of the absolute value between all of the estimate's time and the real time, at index i. In mathematical terms this looks like $j = \min[\text{abs}(t_2 - t_1(i))]$, where t_2 is the estimate's time vector, while t_1 is be the real value time vector. Note that both time vectors had to be converted to relative time in order for this to work. $\hat{y}(j)$ is then the closest point of $\hat{y}$ in time to $y(i)$. After determining the correct index of $\hat{y}$ MAE is computed according to eq. (10), MSE is computed according to eq. (11), RMSE is computed according to eq. (12) and MAPE is computed according to eq. (13). All accuracy measures are then set as properties for the function output object, `error`.

In `main.m` the first thing done is retrieving the data from the data sets. In the handed out code, both sensors from each data sets were used, however this was changed to just include one sensor in this project's code. For all filters and all data sets the chosen sensor was always the first sensor. The rest of the processing of data from the data sets remained the same as the handed out code. First the BGA time is converted from date time to relative time, so the the BGA measurements can be plotted with the estimates. `main.m` then has a *Choose method* section, where each method has a `do_methodX` variable which is set to 1 if the method is to be performed, and 0 if the method is to be skipped in a code run through. For each method the function containing the estimation calculation is first called, then the statistical score function is called, and lastly plots are produced. The plots all have glucose level [mmol/L] on the y-axis, and time [min] on the x-axis. For the KF and UIKF there is an additional figure, in order to plot C_c and C_r.

The inverse TF method code was slightly changed so it would fit in to the same output format as the KF and the UIKF output objects. Throughout the development of the code, some inbuilt `Matlab` functions has been used. This will not be elaborated on, but they can all be found, in addition to a description of their return value, in table 3. Only the inbuilt functions used by the author is included, hence not the ones used in the handed-out code, nor the borrowed function.

Model parameters	Value
T_{isf}	12 [min]
T_d	10 [min]
$Bias$	0.01 [mmol/L]

Table 1: Model parameters for the compartmental model presented in eq. (7).

System matrices	KF	UIKF
A	$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -\frac{1}{T_d} & 0 & 0 \\ 0 & \frac{1}{T_d} & -\frac{1}{T_d} & 0 \\ \frac{1}{T_{isf}} & 0 & 0 & -\frac{1}{Tisf} \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -\frac{1}{T_d} & 0 & 0 \\ 0 & \frac{1}{T_d} & -\frac{1}{T_d} & 0 \\ \frac{1}{T_{isf}} & 0 & 0 & -\frac{1}{Tisf} \end{bmatrix}$
B	Not in model	Not in model
H	$\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$
D	Not in model	Not in model
F	Not in model	$\begin{bmatrix} 0.1 & 0.1 & 0.1 & 1 \end{bmatrix}^T$

Table 2: System matrices, continuous and discrete, for KF and UIKF, based on eq. (36) and eq. (54).

Function name	Return value
`size(...)`	Returns the array size.
`rank(...)`	Returns the rank of a vector/matrix.
`obsv(...)`	Returns the observability matrix.
`nan(...)`	Returns array of scalar representations of ”not a number”, or NaN.
`isnan(...)`	Returns logical array corresponding to if value is NaN or not.
`length(...)`	Returns length of an array.
`inv(...)`	Returns the inverse square matrix of a value/matrix.
`diag(...)`	Returns diagonal matrix with specified values at the diagonal.
`eye(...)`	Returns the identity matrix.
`min(...)`	Returns the minimum of an array.
`abs(...)`	Returns the absolute value of each element of an array.
`sum(...)`	Returns the sum of an array.
`sqrt(...)`	Returns the square root of each element of an array.
`expm(...)`	Returns the matrix exponential of the input.

Table 3: Used `Matlab` inbuilt functions when developing the code for this project. Descriptions retrieved from [3]

.

6 Results and observations

This section will present the results, and elaborate on some observations based on the results. The section is divided into five parts, where each part, except for the first one, is devoted to one method. The first part is devoted to illustrate the problem at hand, comparing CGM measurements to BGA measurements. The results will, for all tried methods, be presented by a plot with the BGA measurement and CGM measurement for comparison, together with a table showing the initial values, as well as the tuning variables. A statistical analysis of all methods, where the BGA measurement is considered as the real value can be found in table 4. All methods were tested on three different data sets, and the data sets were chosen at random. All methods were tuned using the Mar 5th 2021 data set.

Date	Statistical Method	Inverse TF	KF	UIKF
Mar 1st 2021	MAE	1.13	0.66	-
	MSE	4.70	1.082	-
	RMSE	2.022	1.040	-
	MAPE	20.023%	11.34%	-
Mar 5th 2021	MAE	0.81	0.63	-
	MSE	1.18	1.20	-
	RMSE	1.084	1.096	-
	MAPE	12.66%	10.016%	-
Nov 20th 2020	MAE	0.88	0.62	-
	MSE	5.70	4.80	-
	RMSE	2.39	2.19	-
	MAPE	16.57%	11.65%	-

Table 4: Performance score for all methods.

6.1 Measurement

Observe the lag between the two measurement slopes in fig. 9. This illustrates the difference of measuring glucose in plasma versus interstitial fluid. One can clearly see the lagging characteristics between the BGA BG measurement and CGM IG measurement. However, notice that this lagging effect is not constant throughout the time window. Notice also that on several instances y_{CGM} overshoots y_{BGA}. Also notice the drop in the CGM towards the end of the time window, and how it will effect the following methods.

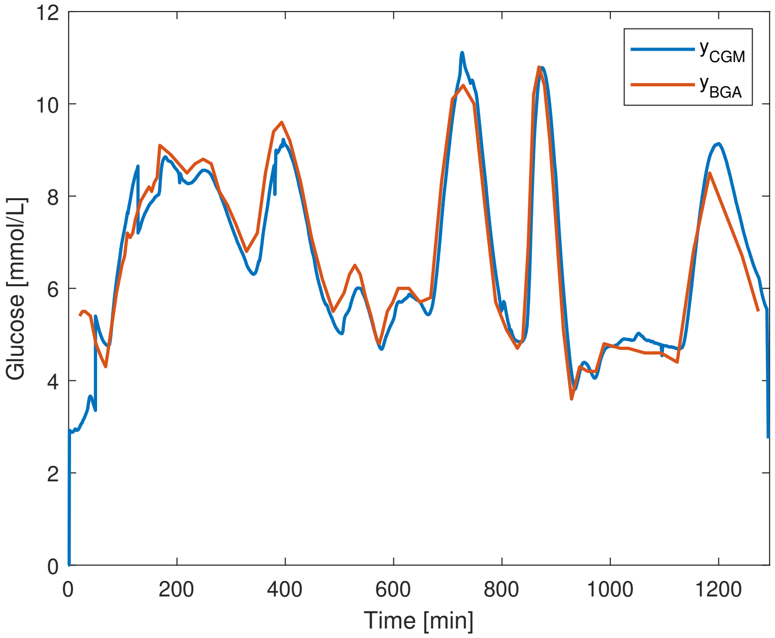


Figure 9: CGM and BGA measurement, Mar 5th

6.2 The Inverse Transfer Function

Initial conditions	Value
x_1	$\begin{bmatrix}0.05 & 0.05\end{bmatrix}^T$
Tuning variables	Value
Pole (a) Gain (B) Closed-loop gain (α)	0.2 $a1.05$ 35

Table 5: Invers TF initial and tuning variables value.

System matrices	Value
A B H	$\begin{bmatrix}-0.40 & -1.58\\ 1 & 0\end{bmatrix}$ $\begin{bmatrix}1 & 0\end{bmatrix}^T$ $\begin{bmatrix}7.35 & 1.47\end{bmatrix}$

Table 6: Reveres BG to IG system matrices.

Immediately one can observe that the inverse TF method is very sensitive to sudden changes in the CGM measurements, see fig. 10, fig. 12 and fig. 13. Whenever there is a sudden drop, or increase in the CGM measurement, the inverse TF estimate starts to oscillate for some time, before it stabilizes again.

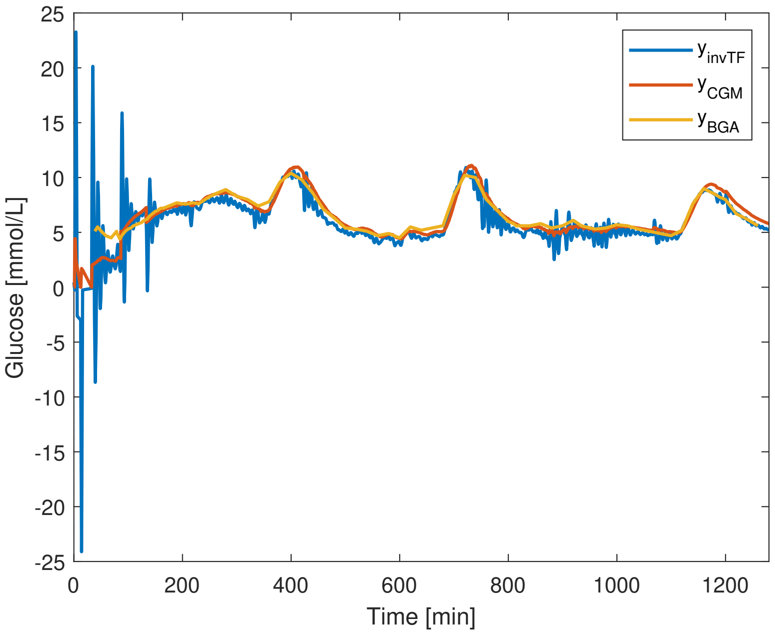


Figure 10: Inverse TF reconstruction of BG on Mar 1st data set

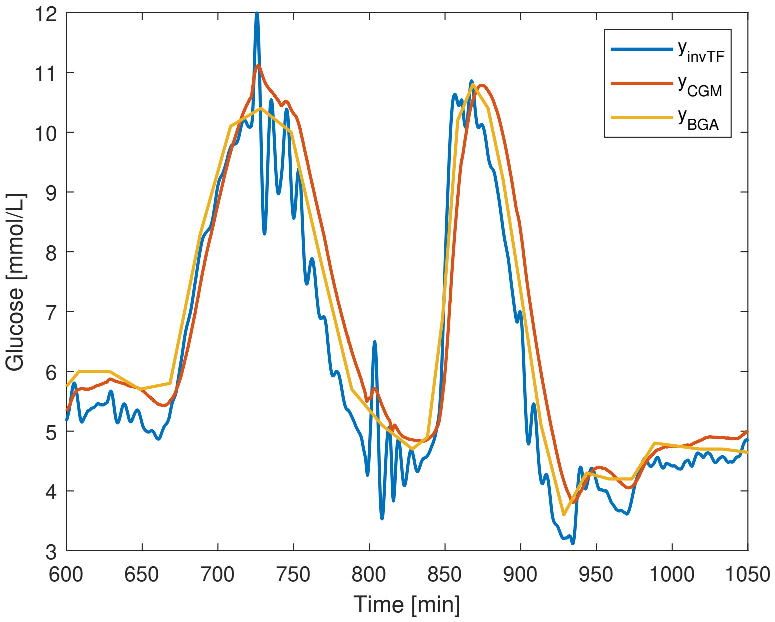


Figure 11: Inverse TF reconstruction of BG on Mar 5th data set, zoomed in.

In the zoomed in figure, see fig. 11, the inverse TF function does seem to be capturing the BGA dynamics, despite the oscillations. The statistical scores are given in table 4, and the initial and tuning variables are given in table 5.

Based on the tuning variables in table 5, the reversed BG to IG system matrices

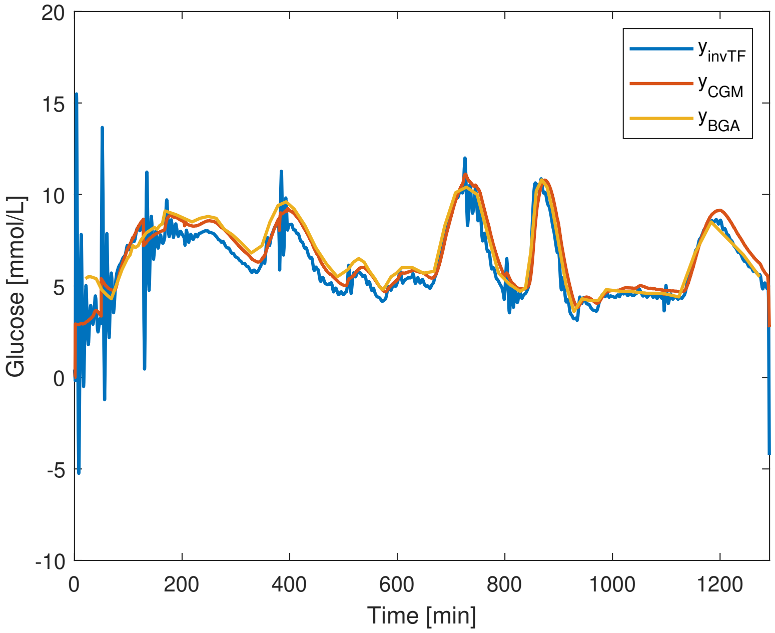


Figure 12: Inverse TF reconstruction of BG on Mar 5th data set

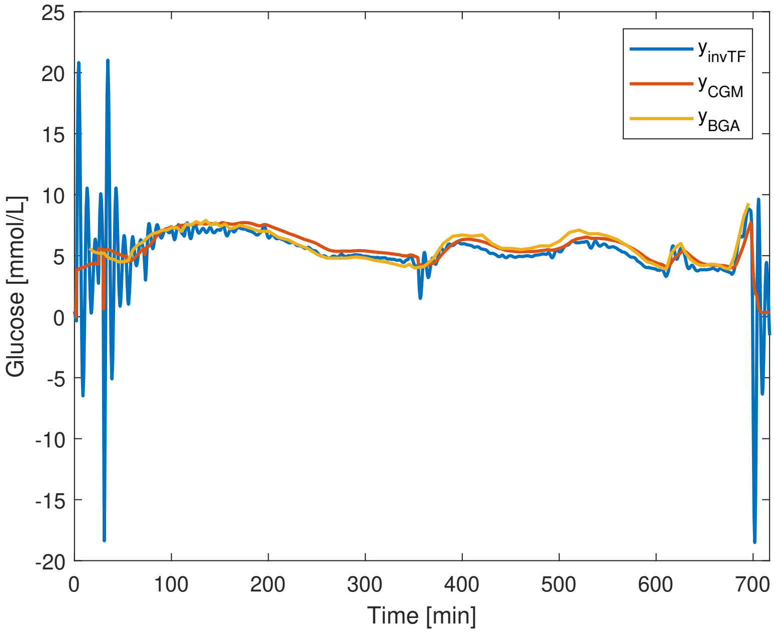


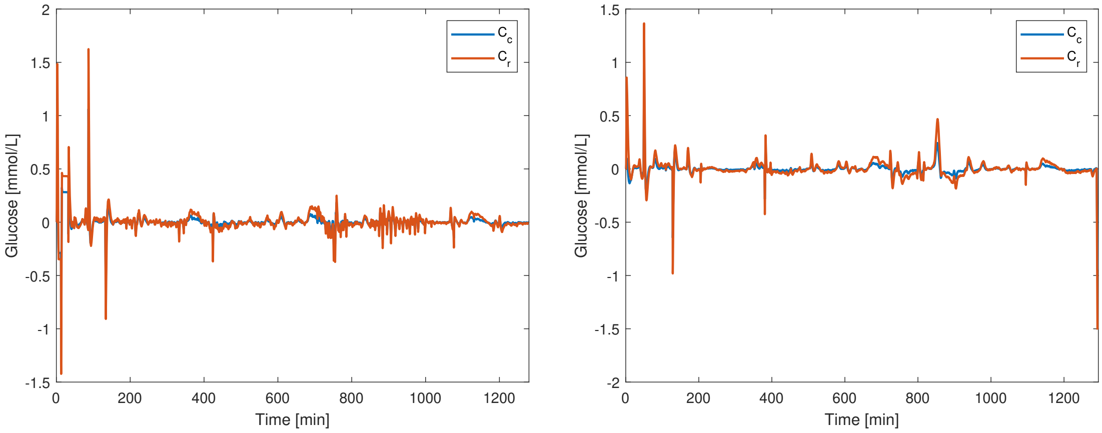
Figure 13: Inverse TF reconstruction of BG on Nov 20th data set

was obtained, where G_{isf} is considered the input u and G_p is considered the output y, see table 6.

6.3 Standard Kalman Filter

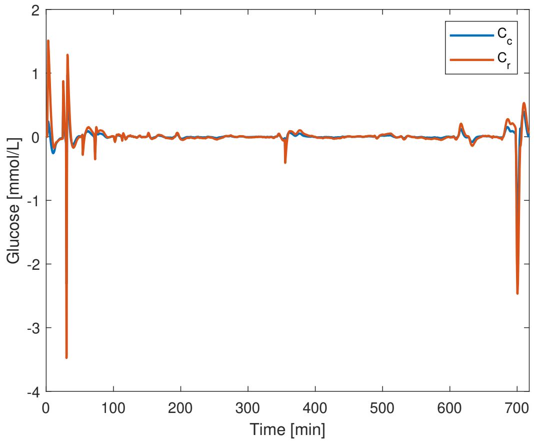
Initial conditions	Value
x_1	$\begin{bmatrix} y_1 - bias & 0 & 0 & y_1 - bias \end{bmatrix}^T$
P_1	$diag(\begin{bmatrix} 0.25 & 1 & 1 & 0.25 \end{bmatrix})$
Tuning variables	Value
Q	$diag(\begin{bmatrix} 0.01 & 0.03 & 0.01 & 0.01 \end{bmatrix})$
R	2

Table 7: KF initial and tuning variables value.



(a) Mar 1st 2021.

(b) Mar 5th 2021



(c) Nov 20th 2020.

Figure 14: KF estimate of C_c and C_r.

Observe the significant spikes especially at the beginning of the x-axis in all plots relating to the KF. Looking at fig. 17, one can observe that the KF barely manages to eliminate the lag at the start of both peaks, however as the BGA rises the estimate improves. In fig. 15 one can see the estimate being very reactant,

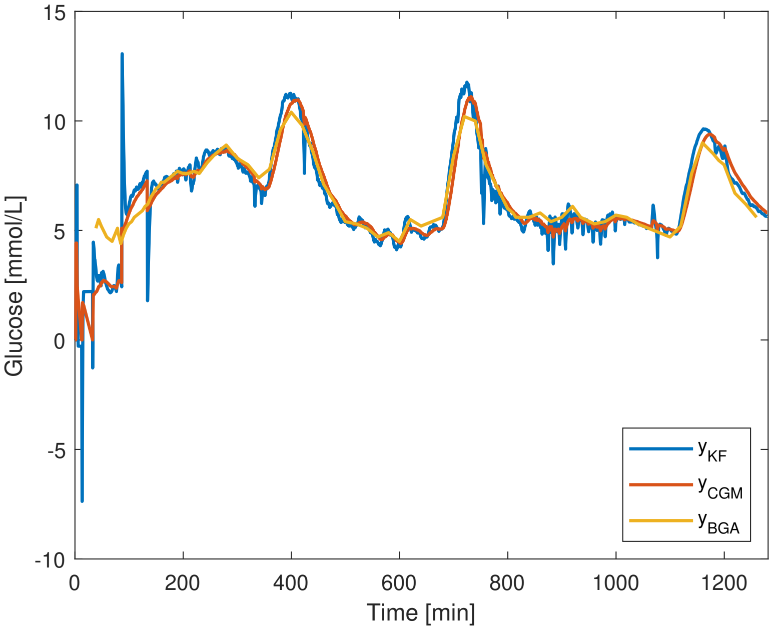


Figure 15: KF reconstruction of BG on Mar 1st data set.

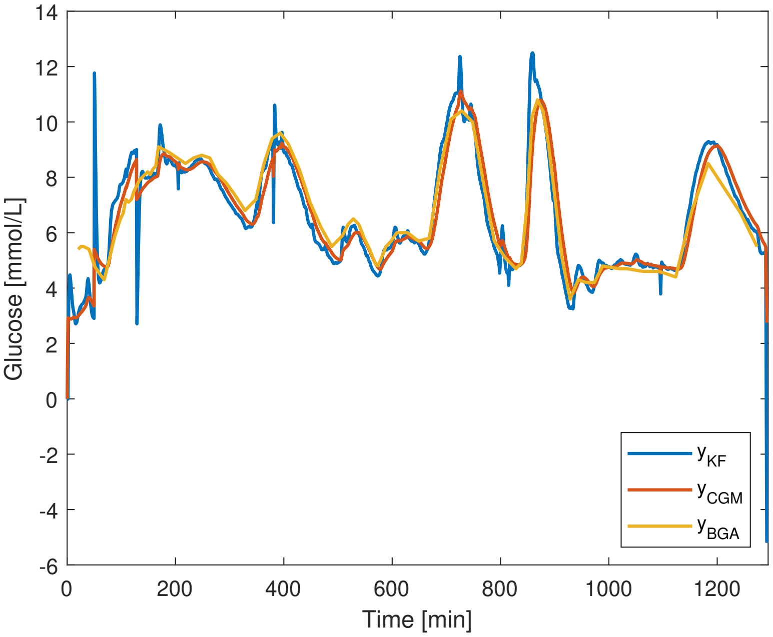


Figure 16: KF reconstruction of BG on Mar 5th data set.

due to noisy measurements. The estimate in fig. 18 is very accurate right from the beginning, but is struggling more in the mid section of the time window. The initial values, as well as the final tuning variables belonging to the KF, can be found in table 7, while the performance score can be found in table 4.

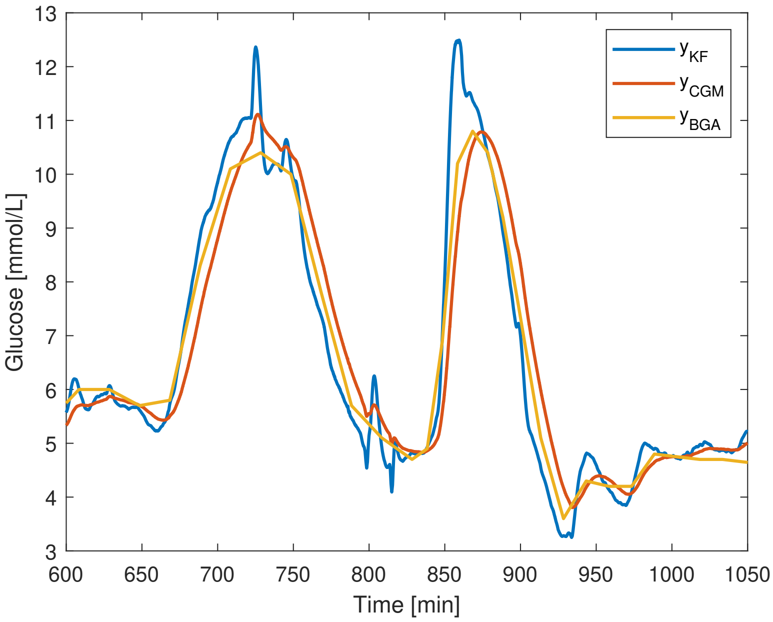


Figure 17: KF reconstruction of BG on Mar 5th data set, zoomed in.

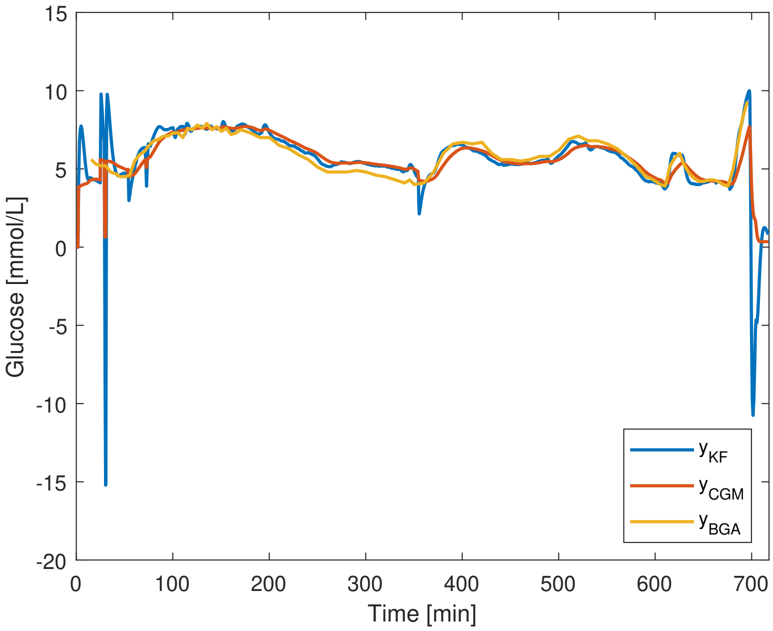


Figure 18: KF reconstruction of BG on Mar 20th data set.

The central and remote compartment estimates can be found in fig. 14. Observe that the value of the states are centered around zero.

The G_{isf} estimate is equal to the CGM measurement for the final tuning values, hence not depicted here.

6.4 Unknown Input Kalman Filter

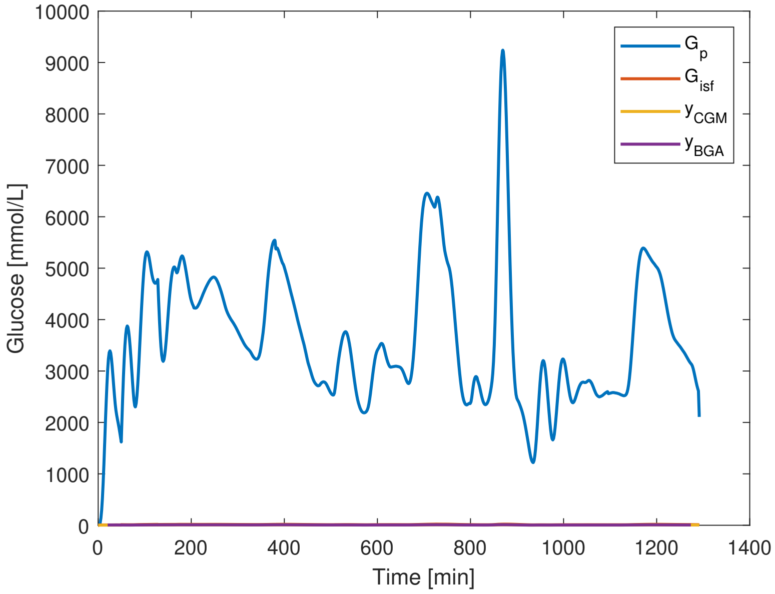


Figure 19: UIKF reconstruction of BG on Mar 5th data set.

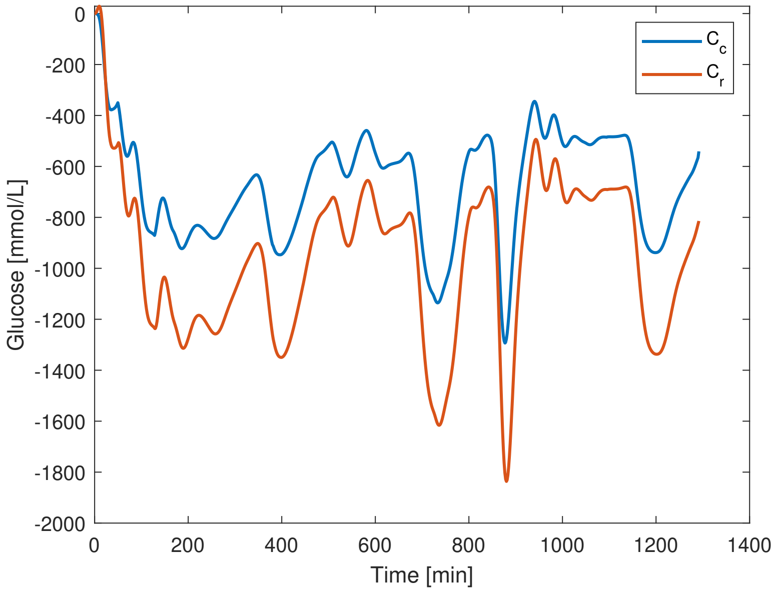


Figure 20: UIKF estimate of C_c and C_r on Mar 5th data set.

Notice the timing of the peaks in fig. 19. Also observe the difference in dynamics of C_c and C_r between fig. 20 and fig. 14b. Final tuning values for the UIKF can be found in table 8.

Initial conditions	Value
x_1	$\begin{bmatrix} y_1 - bias & 0 & 0 & y_1 - bias \end{bmatrix}^T$
P_1	$diag(\begin{bmatrix} 0.25 & 1 & 1 & 0.25 \end{bmatrix})$
d_1	0.5
Tuning variables	**Value**
Q	$diag(\begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix})$
R	2

Table 8: UIKF initial and tuning variables value.

7 Discussion

This section will discuss the results from section 6 as well as the observations made from the results. First however, some discussion based on the code implementation will be presented. This will not elaborate on the specific methods, seen as this will be done separately for each method, but function as an overall code evaluation. Then some points will be discussed regarding both the BGA and the CGM measurements. Then the three tested methods will be evaluated and discussed, both the results and the theory relating to them. The tuning process will be explained, as well as interesting observations made during this process. Potential pros and cons will be discussed, and comparison between the methods will be made. Lastly some thoughts on the performance scoring system will be presented.

7.1 The code

The code up until code line 69, is code from previous work. Hence the discussion regarding the code will be from code line 69 and onward. When choosing which method to run, one has to go into the code and switch the value of certain variables from 0 to 1. This is perhaps a bit cumbersome, but seen as this project is only considered a first step in the process of developing a method for BG estimation, it was seen as acceptable. Time was not spent on optimizing the code, but rather improving the estimates of the different methods.

There are in all four different sections to run in the code; the measurement comparison, the standard KF, the UIKF and the inverse TF method. The code for the inverse TF method was from previous work, while the remaining code was developed in this project. The measurement comparison was fairly simple, and only consists of a time conversion and a plot. The time conversion is a borrowed function from [23]. This is an open Git repository. The KF and UIKF code were more comprehensive, UIKF being the most complex implementation of the two.

In hindsight it is observed that the build of each section could be simplified. A function could have done all of what is performed for each section in `main.m`. This would have made the main file cleaner. However it might have been at the cost of understanding what was going on in the code. The advantage of having all operations done separately is that it is easy to see what is being done for each method, by only looking at the `main.m` file. In order to understand the code, one does not have to look into each function, which might be a positive contribution.

7.2 Comparison to BGA samples

In this report it is chosen to compare the estimates to the BGA measurements. This is because BGA measures the glucose level directly from the blood, hence it is as close one can get to the true BG value. Despite this, there are some problems with using especially this BGA measurement as the ”true” value.

The first issue is that the BGA measurement was sampled at a different rate compared to the CGM. New measurements arrive every 1.2 [sec] in the CGM system, while a varying amount of minutes passes between every new measurement in the BGA. This is the reason why the BGA measurement looks so rigid compared to the CGM slope in fig. 9. Looking at the figure one can see that instead of following a natural curve the BGA cuts to a new values at certain points in time. Due to the long sampling time one cannot completely capture the true BG dynamics with the BGA measurement. It could therefor be that the BG estimates actually show a more true dynamic of the real BG level at certain times in the plots.

Visualizing BGA with a continuous graph probably enhances this strange dynamic, however in order to visually compare the estimates of BG it was deemed necessary. The issue is also present in the statistical analysis, and must therefor also be pointed out. The statistical methods uses BGA as the true value, and again due to the sample rate they cannot be viewed as a perfect analysis. This will be more explored in the discussion of each method separately. Comparing the difference in score with the different plots for each method, one can argue that despite a somewhat flawed true value in the computations, the scores do tell the truth about the most accurate method. Although not without flaws, it was for this term-project considered good enough. In further development however, the statistical analysis should be improved, see section 9.

7.3 CGM Measurement

The CGM measurements are considered the base of all BG estimates, and is therefor an important part of the simulations. Only one stream of CGM measurements was chosen per data set in this project. Although the used measurements streams was of good quality, it was observed that not all sensors provided the same quality. Fault detection was not developed in this project, hence it is logical to think that the estimates would show much worse performance, had more faulty sensor data been used.

Comparing CGM and BGA one can clearly see the T_{isf} effect, see fig. 9. The diffusion from blood to ISF causes a delayed effect in the sensing of glucose with the CGM sensor. However there is a noticeably bigger deviation in the first third of the time window. The CGM seems to drop and increase quite rapidly. This is because the CGM sensor spends some time calibrating. There was not implemented any sort of pre-processing of the measurement, in order to avoid the effects of calibration. The estimates are therefor also affected by this.

A possible solution would be to low-pass filter the CGM, or smooth it in any other way. However, any type of smoothing will introduce additional lag. As the goal of this project is to estimate the BG from CGM, introducing additional lag was viewed a bit counterproductive. However, this additional filter lag could perhaps be eliminated by the filters by altering the model. It is therefor considered a future possibility.

It is also possible to spot some sudden changes in the CGM later in the time window. These are most likely due to faulty measurement samples. A contributing factor to these suggestions is that so sudden changes are not typical characteristics of glucose level change. It is believed that sensor fault detection could have eliminated these samples. There is also a long spike pointing towards zero at the end of the time window in fig. 9. This could be a result of dismounting the sensor, or simply a consequences of stopping the sensor recording. As this anomaly will affect the estimates, it is not considered a counting factor in evaluating the estimates.

7.4 The Inverse Transfer Function

The inverse TF method had already been tested, hence not much time was spent tuning this method to perfection. However, some time was spent on just observing the filter responses based on different tuning variables. The variables in table 5 was chosen as they seemed to provide the best estimate.

Setting a quite low caused the estimate to over estimate some of the peaks in the time window. The oscillations also worsened at certain times, while improving at other times. The estimate could be described as looking somewhat more aggressive. The opposite was observed when a was set high. Now very little oscillations were present, except at the start of the time window, however the estimate was not able to capture the BG dynamics. In other words the estimate became too conservative, only following the CGM slope. Seen as a is the pole of the system, this is very much in line with control system theory.

When slightly adjusting b the estimate was observed to have a seemingly fixed offset value, as seen from the CGM slope, while maintaining the same dynamic.

Increasing b too much, caused the response to became unstable. When setting α too low, the oscillating effects was reduced, but the estimate also worsened, not being able to capture the BG dynamics. Increasing α resulted in more oscillations, and bigger amplitudes. Both b and α are considered gains in the system at play here, hence the estimate's response is not surprising. A collection of plots of the different tuning responses can be found in Appendix B.

This method for estimating BG level can be considered suitable, since it is a fast method, hence applicable in a real-time setting. An obvious drawback is of course the oscillations. It looks like the inverse TF is very sensitive to changes in the CGM, since whenever there is a sudden change in the CGM a oscillating response is starting in the estimate. This is especially noticeable in fig. 10, which is the data set containing the noisiest CGM measurements. Some mechanic to reduce the oscillations should be in place, if the method is to be used in an real-time setting, where precision is important. Time was not spent on perfecting this method, hence no such mechanism was developed in this project. It can also be noted that it isn't until the latter three peaks in fig. 12 that the estimate seems to be accurately capturing the true BG dynamics. This is slower than the competitive KF method for the same data set.

In fig. 13 the opposite response is present. Here the estimate seems to fit better at the first half of the time window, compared to the reminder. In this data set the glucose seems to be rising and falling at a slower rate compared to the other two data sets. The rate of change in glucose could therefor be a counting factor in the inverse TF's performance. Since the oscillating effects are only present in the beginning and towards the end in fig. 13, this theory becomes more believable.

However, one can observe that the inverse TF seems to be estimating the peak value better than the KF, avoiding over estimating th BG level, which is definitely a factor in favour of this method. Compared to the KF this method seems a bit less aggressive. A possible drawback worth mentioning, is that the only output of the filter as of now, is the estimate itself. In some scenarios one could benefit from having more estimated states, as well as an estimate of the uncertainty of the estimated states at one's disposal, which makes this method, less beneficial.

The statistical scores depicted in table 4 show that the quality of the inverse TF estimate is somewhat inconsistent. Especially regarding the MSE and MAPE values. A reason for this inconsistency may be due to the oscillations occurring whenever there is a sudden change in the CGM. y_{BGA} on average consists of only about 100 samples, while the estimate on average consists of over 70 000 samples, for the 24 [h] experiments. If then by chance the estimate is oscillating at the exact same time as the comparing y_{BGA} sample is timed, it would greatly impact the performance score. Between the data sets it could then be by sheer chance that the estimates are not better, or worse. The scores in table 4 may then not be accurately showing the true inverse TF method's performance. Since the estimate shown in fig. 13 is affected by the least amount of oscillations, it gives reason to believe that the scores for the Nov 20th 2020 data set are the most true scores for the inverse TF method, when BGA measurements is set to be the truth.

7.5 The Model

The choice of model for the Kalman filters is motivated by primarily two things. Firstly the model has to be observable when only the CGM is available. Secondly the model should have no known external input. Having a zero known input system makes the model simpler, more generalized and applicable. Hence the model in eq. (7) was chosen. The compartmental model was also used in [8], hence it was already tested and documented to work well in an offline setting. There was no indication that the model would not also work in a real-time application using other variations of Kalman filters, hence the same model was used in this project. The inverse TF method on the other hand, inverts the simpler kinetics from eq. (6).

Non of the implemented KF's were capable of handling non-linear systems, hence T_{isf} needed to be determined before constructing the model. As this project is concerned about the Kalman filter's capability of reconstructing BG level, time was not spent on estimating T_{isf}. Determining T_{isf} could then probably be viewed as a tuning variable, seen as it was adjusted slightly to perfect the estimate. Due to this, T_{isf} was also assumed to be constant. Different values for T_{isf} were tested, and the responses were as one would expect. If T_{isf} was set too low, the estimate would lag behind the BGA together with the CGM, hence the performance suffered. If too high it did not affect the estimate greatly.

Looking at fig. 9 one can see that the assumption of constant lag between the BG and the CGM measurement is not entirely true. The CGM lags behind the BGA by a varying amount, contributing to the belief that T_{isf} is in fact varying through time. This could be a result of the calibration of the CGM sensor, or be due a physiological response. Also it can be mentioned that, despite $T_{isf} = 12$, seemingly fitting all the chosen data sets well, it might not be representative for all individuals. Individual differences should be accounted for, and could, when T_{isf} is set to a constant value, greatly effect the estimates performance. This is an interesting observation, and makes the argument for estimating T_{isf} together with G_p all the more stronger. This will however be a main focus in future work, see section 9.

The second time constant to be determined was T_d. This constant describes the rapidness of flow between the central and remote compartments in eq. (7). Given the information about how T_d affects variance in C_c and C_r, see section 2.2.2, it should be tuned together with the process noise covariance matrix Q. 600 [s] = 10 [min] was the value that was used for T_d in [8]. The model is adapted from this article, hence the same value was tried in this project. Changing T_d did not seem to affect the estimate greatly, hence it was also set to 10 [min] in this work. $\sigma^2_{C_c}$ and $\sigma^2_{C_r}$ was then tuned to fit $T_d = 10$ [min].

The resulting KF estimate's behaviour based on different values of T_{isf} and T_d can be found in Appendix B.

When implementing the model in `Matlab` code, it was done in two separate functions relating to the KF and the UIKF. Both functions are in essence the same, with the addition of the unknown input transition vector for the UIKF. Seen as this was the only difference, a more elegant way of implementing the model would be to have a base function, and then augment the system with the necessary, depending on the method.

The model also needed to be discretized from eq. (7), hence this was done according to section 2.4. Since the model has no known inputs, only the A matrix needed to be discretized. In order to do this a sample time Δt had to be set. New CGM measurements arrived every 1.2 [sec], hence Δt was set to 1.2 [sec]. Not all CGM sensors have the same sampling rate, hence if a different sensor, or data-set with different sampling time is to be used, Δt needs to be adjusted thereafter. The location of this parameter could then perhaps be more reachable, since one would have to go into the function of constructing the model in order to find it. This will be taken into consideration in future work, and may be optimized. F in the UIKF was assumed constant, hence there was no need for discretization of this matrix.

Relating to eq. (1), the bias had to be determined. This value is used when initializing the state vector. Since only CGM measurements were to be used, it proved to be very difficult estimating a possible bias. Since the implemented filters would have to work in a real-time setting, finding the bias by comparing the CGM and BGA measurements was not an option. The bias was therefor set to various constant values, and the response of the G_p estimate was observed in order to find the correct one. Slightly adjusting the bias did not seem to affect the estimate in a great manner. Seen as the bias can vary from sensor to sensor, it would be preferable to construct a method for getting a suitable value for this constant in future work.

7.6 Standard Kalman Filter

The goal of a Kalman filter is to estimate the states of a system, based on outputs of that system. Hence the standard linear Kalman filter was a natural first choice of method to try in regards to this problem. Early on in the project lifetime the standard Kalman Filter was however not considered a good fit. The G_p estimate was not able to eliminate any lag, and only followed the CGM slope. Tuning the filter did not improve the performance. It was considered very odd, and illogical when looking at the model. One would think that the filter, based on the model in eq. (7), would at least make an attempt of trying to mimic some other dynamic than the CGM dynamic. After some time a bug in the `Matlab` code, relating to sampling time, was discovered. This bug was the cause of the ineffective KF. Fixing this bug resulted in the Kalman filter actually producing an estimate that made theoretical sense. The bug will be discussed in further detail in section 7.7.

Once the filter was running as it should in `Matlab`, it was tuned in order to enhance the performance. To evaluate the tuning, comparison with BGA was done visually by looking at plots, as well as observing the statistical measures, and trying to lower them. Extensive tuning was done, in order to observe the response and understand the filter better.

Increasing R caused the estimate to be smoother, but at the cost of being accurate. No change was immediately visible while plotting the estimate together with the BGA measurement, but the statistical scores slightly increased. However it must be noted that R had to be increased quite extensively in order to provoke a significantly worse response. Decreasing R showed minimal effect. Since R describes the measurement covariance, this response is not surprising. Increasing R means that the measurements are more penalized, which again tells the system to trust the measurements less. The ISO standard was not taken into consideration when tuning R, however in further development of the code it will be fairly easy to implement, and will be a prioritized task. It can be considered an overlook in this version of the code. However seen seen as adjusting R did not seem to massively change the outcome, until an extreme adjustment, the overlook is not critical.

Since the system is a four state system, Q is a four by four matrix. Each of the values located on the diagonal of Q describes a variance that corresponds to one of the states in the system. Each of these values were tuned individually and the response was observed. Q is the process covariance matrix of the system, so just like for R, increasing a value of Q penalizes the model of that state more, hence trusting the model less.

Increasing the variance of blood glucose, $\sigma^2_{G_p}$, caused the spikes of the estimate, see fig. 16, to get bigger, hence the filter became more aggressive. Decreasing the variance seemed to have no effect on the estimate. The variance of the remote compartment, $\sigma^2_{C_c}$, did not seem to affect the G_p estimate when adjusted neither down nor up. The response of C_c and C_r became overall bigger when $\sigma^2_{C_c}$ was increased, compared to the response seen in fig. 14b. Decreasing $\sigma^2_{C_c}$ caused both C_c and C_r to be very damped. Increasing $\sigma^2_{C_r}$ caused the same effects for C_r, but not for C_c. This is as one would expect, since C_r is not set to affect C_c in the model, only the other way around. The spikes in G_p was also enhanced, which is also in agreement with the model, see eq. (7). Decreasing $\sigma^2_{C_r}$ made C_r less reactant, and more similar to C_c. The G_p showed no visible response. The last variance is $\sigma^2_{G_{isf}}$. When this was increased the estimate only followed the CGM, as well as the estimate for G_p became much worse, lagging behind the CGM. When decreased, G_{isf} still followed the CGM slope, but showed a slight more dramatic response at the peaks. The G_p estimate was not greatly affected by this, only showing a very small increase in the statistical scores. A collection of plots of the different tuning responses can be found in Appendix B.

The filter seems very sensitive to sudden changes in the CGM, see fig. 16 and fig. 15. One can see the same characteristics in fig. 14b, where the spikes occur at about the exact same time as in the plots for the G_p estimate. It was tried to reduce the spikes of the estimate by tuning the filter, however it was not accomplished without going at the cost of the G_p estimate accuracy. However, despite the noise, the estimate does seem to be capturing the BG dynamic quite well. Just like for the inverse TF method the filter shows the most dramatic response at the start and at the end of the time window. This is most likely due to the CGM calibrating, hence giving non optimal measurement samples. Despite this the KF still outperforms the inverse TF method, see table 4.

There is however one consistent characteristic for the KF that makes it less favorable. The filter seem to consistently over-estimate BG level at the peaks, see fig. 17 and fig. 15. It was tried to reduce this characteristic, but that proved difficult without it going at the cost of the estimate accuracy elsewhere in the time window. For almost all the over-estimated peaks this behaviour can be explained by the filters sensitivity to changes in CGM. Whenever CGM rises slightly, the filter does so as well, only in a more dramatic manner. This gives indication that the filter is too aggressive. However, in fig. 18 the same aggressive behaviour is not present. The reason for this could be that the glucose dynamics seems to be slower in this data set, hence the filter becomes less reactant.

One peak however cannot be explained by this in fig. 17. The right most peak shows very different dynamics compered both to the CGM and the BGA slopes. The true BG dynamic at this exact point in time is however unknown, as BGA can be seen ”cutting” to a new value after the estimate’s odd peak. Therefor one cannot know if the filter is describing the actual hidden BG dynamic at this point, or simply estimating wrongly. It is an interesting behaviour nonetheless, as it is not seen elsewhere in fig. 16, nor in fig. 15 or fig. 18.

In the KF model it is assumed that the noise relating to the measurement equation, eq. (1), is a Guassian white noise process. This assumption can be debated, and might be the cause of some of the estimate's flaws. Some researches claims that the CGM measurement noise is in fact non-Gaussian, and that other probability distributions are more appropriate. Some also argue that the sensor noise is confounded by BG-to-IG kinetics model inaccuracies, and calibration errors [8]. There are also a wide variety of CGM systems, and their sensing technique, although very similar, are not equal.

Meal effects and insulin are also modeled as noise, and much of the other reactions causing BG level to rise or fall is lumped together in the compartmental states. The implications this causes for the system is not known, seen as models with inputs were not tested using the KF. It can however be considered that input-less models, although very general and applicable, might be simplifying the system too much. More models should therefor be tested using the KF, in order to find the best model, or simply confirm that the input-less model is a sufficient model.

The KF seems reasonably consistent regardless of the above, based on the scores in table 4, more so than the inverse TF. This is a positive sign, and gives credibility that the KF will work on any arbitrary data set. The same effect mentioned in section 7.4, about the sample time of y_{BGA} relating to the performance scores, will have an effect also for the KF. However, seen as the KF is very consistent in the scores, there is a notably less effect of this phenomenon. It should however still be taken into consideration when evaluating the values in table 4.

One can see that the Nov 20th 2020 data set has the worst performance, and looking at fig. 18 it is not surprising. The deviation between the estimat and the BGA value is very large at certain times, and looking at table 4 the prominent measure is MSE. Recall that larger errors are given more weight in this accuracy measure.

Despite th KF not performing perfectly now, it is a suitable method considering a real-time application. Looking at eq. (36), one can see that the filter only uses the current measurement, as well as the previous states, as well as the previous covariance matrix. The filter can therefor be called causal. Seen as this is an absolute demand for BG estimation in this project, this counts in favour of the KF.

Another positive side of the KF is that one can easily obtain the covariance matrix of the estimated states. This could prove very useful in a number of applications. The diagonal of the covariance matrix is made up of the variances of every estimated state in the system. In sensor fusion applications, knowing the variance of the estimates using a specific sensors output is a key factor. It could for example tell the algorithm which sensor to trust more at specific times, by looking at how much the estimate varies. This could prove useful, and would elevate the accuracy of the estimate.

7.7 Unknown Input Kalman Filter

The UIKF is an extension of the standard KF, hence the arguments in favour of KF is automatically applied also for the UIKF. What distinguishes the UIKF is the ability to estimate an input, for which nothing is known. The idea of a filter where one could simultaneously estimate the states and an unknown input, was an attractive idea in regards to this project.

Recall section 2.1, where there are mentions of problems relating to the CGM sensor, as well as the concern of assumed noise processes in section 7.6. The model in eq. (7) is also an input-less model, where reacting hormones to glucose, such as glucagon and insulin, are modeled as noise. It would seem that the assumption of all white, zero-mean Gaussian noise processes, has less credibility the more processes it is said to model. If then, instead of white noise, the processes can be modeled as unknown inputs, it would theoretically perhaps be a more accurate model. In addition there are many processes happening simultaneously in the body, which may or may not affect the glucose concentration in the blood. All these process will in the KF model also be assumed as white-noise processes, or lumped into the central and remote states. It is in this project not thoroughly investigated what some of these processes may be, and this should be prioritized in future work. Regardless, it will in many cases not be processes happening in complete randomness, hence the assumption of white noise may not hold. Hence the need of unknown input estimation is only amplified.

An UIKF was then implemented in `Matlab` according to the theory of the filter, presented in section 2.7. One modification was made. In [20] the filter equations uses the next measurement y_{k+1} in order to compute the estimate. A demand for this project is that the BG estimation must be performed by a causal system. In order for the UIKF to be a qualified candidate, this was changed to y_k. It did not seem to affect the response of the filter, neither concerning the visuals nor the performance scores. The transition vector F was also set to affect all the states. The reason for this is simply that all states were assumed to have some sort of unknown input affecting their response, that could not be modeled as a white-noise process.

For a period this filter was the primary focus of this project. However, later in the project lifetime it was discovered that there was a bug relating to the sampling time of the model, as briefly mentioned in section 7.6. Δt was set to be 10 [sec], while the actual sampling time in the data sets used for the simulation is 1.2 [sec]. Since the UIKF produced a somewhat sensible output despite this wrongful sampling time, it was not discovered earlier. In hindsight however, one could notice signs that the filter was not working as it should, despite the output. Tuning the R did not produce the expected response, just shifted the value of estimate slope on the y-axis. Adjusting Q also produced some odd behaviour, however not as noticeable as for R.

Setting Δt to the correct value resulted in massive changes, both for the KF and the UIKF. Not only did it make the standard KF actually produce a very precise estimate, it also made the UIKF estimate very reactant and odd. The UIKF showed a very illogical response, and despite spending time trying to tune this odd response away, it was not accomplished. It was decided, based on these responses, that tuning the KF should be prioritized over investigating the odd behaviour of the UIKF.

The filter was therefor tuned into giving the best possible response. In fig. 19 one can clearly see that the estimates are extensively gained. The peaks can be seen climbing at ridiculously high values of BG, well above the 10^3 range. Even so, if one compares fig. 19 with fig. 16, one sees that the BG dynamic is similar to that of the KF estimates. The peaks occur at about the same times as the peaks in the KF estimate. This proves that there is potential in the UIKF. The tuning of R and Q might be the issue at play here, however the possibility of a bug relating to the matrix computations, resulting in a gaining effect, is also considered.

Looking at the C_c and C_r estimates, peculiar behaviour can also be spotted. In section 2.2.2 these compartments is said to be swinging around zero, showing positive values when describing meal effects and negative values when showing insulin effects. This is not in agreement with the response in fig. 20. As we know from looking at fig. 14b, as well as the information about insulin in the data sets, insulin is not continuously supplied to the subject, and definitely not in the quantity as fig. 20 would suggest, the filter response is considered faulty. Seen as G_p is set to be affected by C_r, it leads to the belief that the tuning of these two states might be the problem. However, non of the attempted tuning value combinations of R and Q were able to change the faulty C_c and C_r dynamics, hence neither G_p. It was also attempted to reduce the impact of the unknown input on these states, by reducing their corresponding value in F. This attempt did however also fail, and the compartmental states were still observed to be extensively low in value, and behaving illogical, according to theory.

Some tuning results can be found in Appendix B. There are fewer plots than for the KF and inverse TF, however the ones that are included are from the tuning variable combinations that provided a solid difference in the estimate response. The ones that did not provoke any significant difference are omitted.

Although not visible in fig. 19, G_{isf} was seen to be quite elevated from the CGM measurements, while maintaining the same dynamic. This is not surprising as G_{isf} is affected by G_p, see eq. (6). The reason why it is not visible, is because the elevation is of such small contrast to the gained G_p estimate, that the plot is not able to show both graphs simultaneously.

The potential of the UIKF, despite the odd responses, is not forgotten. Seen as the G_p estimate is not of a completely wrong dynamic, only gained quite extensively, it is assumed that a desired response is within reach. This will require matrix computation debugging of the code, as well as more finely tuning of the filter. Debugging and tuning are time consuming tasks, hence no further effort was given in this project for finding out what causes this gaining effect in G_p. However, in future work this would be of top priority, seen as according to theory of the UIKF, in addition to the G_p estimate's dynamic in fig. 19, it would be very interesting to see if the UIKF does outperform the standard KF. In addition, it must be mentioned that the UIKF has, like the standard KF, the capability to operate in real-time. This makes the UIKF a valid method to use for estimation of BG in this project. No performance score is given to the UIKF in table 4, since it is known that the current resulting response is not the correct response. Having scores for UIKF would then give a wrongful impression of the filter's capability. The formal evaluation of the filter is therefor saved for future work.

7.8 The Statistical Scores

Since the problems related to the performance score calculations in the code has already been elaborated on, this subsection will rather focus on the chosen statistical methods, and their use.

The statistical methods was chosen based on the work done in [14]. This article presents a smoothing filter algorithm, and thereafter evaluates the filters performance. It is stated in the article that: ”The performance should be verified or validated by comparison of its forecast with historical data for the process it was designed to forecast” [14]. Seen as the goal of this project is very similar to that of this article, the same ways of evaluating the performance was used in this work. This included MAE, MSE, RMSE and MAPE.

There is no consensus among researches as to what the best measure to determine a filters performance is. Hence it seemed reasonable to have a collection of measures, which together could paint an overall picture of the performance. In addition, accuracy is thought of as the most important concern regarding the filter's quality. The more accurate the filter, or the estimate, the smaller deviation from the actual value there is. All the statistical measures used in this work are methods for evaluating accuracy. The collection of methods are also easily interpreted, which helps simplify the overall evaluation. The smaller the value they produce, the better the estimate.

There are more elegant and complex ways of estimating a filter's performance. However, this is saved for future work. It was not a main focus of this project to discover an optimal way of measuring the BG estimates performance, hence these simple methods proved good enough. However, it was discovered that their resulting values were beneficial in more ways, than just evaluating the final performance. While tuning the filters, they proved very useful, seen as not all changes were visible to the eye in the plots when only slightly adjusting the tuning variables. The changes was however very visible in the scores, which made it easier to find the best combination of tuning variables.

One can see in table 4 that all the calculated MAPE scores are well within the accurate and acceptable range, however non of them are within the very accurate range, $< 10\%$. This could, as previously mentioned, be the result of the oscillations, or the spares BGA measurements to use for comparison. However for the Mar 5th data set, the standard KF comes very close. All the remaining values show very promising results.

8 Conclusion

In a real-time setting, both the KF and the inverse TF could be considered suitable methods. However, neither prove to be able to produce a robust estimate of the BG based on subcutaneous sensor measurements. Some form of noise reduction mechanism must be in place in order to make use of the estimate. Seen as the KF outperforms the inverse TF it is concluded in this report that KF is the preferable method to use for BG reconstruction out of the two. The KF also estimates more states than only the blood glucose, and provides an estimated uncertainty of all the estimated states. All of which counts in the KF's favour. The KF remains a good possibility of a real-time reconstruction of BG, and should be further refined in order to improve the estimate.

In regards to the UIKF, very little can be concluded, seen as the filter output is not tuned properly, and some errors are at play. The filter should however still be considered as a candidate, as based on the theory it could be able to eliminate the factors contaminating the measurement samples, as well as handle other unknown system inputs. It is therefor in this report concluded that the UIKF is still a possibility, and should be investigated further.

The comparison with BGA is viewed as not optimal, due to the slow sampling rate of the BGA measurement. This, together with the use of very simple performance measures, causes randomness in the performance score, seen as both the KF and the inverse TF shows oscillating or spiking characteristics. It is however a interesting proof of concept and should in future work be built upon, and further refined, in order to improve the accuracy of the performance scoring.

9 Suggestions for future work

This section will list suggestions for future work relating to this project. Those will be based on the experiences made throughout this project, as well as tasks that were down-prioritized in this project.

Topic 1 Implement measurement covariance matrix in accordance with the relevant ISO standard.

Topic 2 Tune and/or debug the UIKF function in order to determine, and resolve, the gaining effect of the BG estimate.

Topic 3 Having the UIKF working correctly, the response based on the various tuning variables should be observed and reflected upon, just like it has been done for the standard KF in this report. The performance scores should be calculated, and comparison with the inverse TF and KF method should be performed.

Topic 4 Develop a method for estimating the bias and the BG-to-IG diffusion process time constant, T_{isf}.

Topic 5 Extending the linear Kalman filters into handling nonlinear systems. Estimating T_{isf} will make the current model non-linear, hence if this is accomplished, a need for non-linear filtering techniques will be required. It is therefor listed as a topic for future work. There are two options that should be immediately considered: the extended Kalman filter and the unscented Kalman filter. Both of these build upon the standard Kalman filter, hence a way of applying their way of handling non-linear system should be incorporated in the UIKF.

Topic 6 Investigate how to accomplish robust estimation of blood glucose, based on CGM measurements. Robust estimation could help reducing the spiking and oscillating effects, as well as the filters sensitivity to faulty measurement samples. A Robust Kalman filter is a potential candidate that should be investigated.

Topic 7 Improve the performance scoring system developed in this project. It is mentioned in this report that the BGA is not the perfect comparing candidate, suffering from slow sampling rate. Hence a more refined way of computing and evaluating the estimate performance should be in place.

10 Bibliography

[1] Andrea Facchinetti, Simone Del Favero, Giovanni Sparacino, Jessica R. Castle, W. Kenneth Ward, and Claudio Cobelli. Modeling the glucose sensor error. *IEEE Transactions on Biomedical Engineering*, 61(3):620–629, 2014.

[2] J. Hall. *Guyton and Hall Textbook of Medical Physiology.* Saunders, 1600 John F. Kennedy Blvd., Ste 1800, Philadelphia, PA 19103-2899, 12 edition, 2011.

[3] The MathWorks Inc. Functions. `https://se.mathworks.com/help/referencelist.html?type=function` (accessed: 12.12.2021), 2021.

[4] K. Turksoy A. Cinar. *Advances in Artificial Pancreas Systems, Adaptive and Multivariable Predictive Control.* Springer International Publishing AG, Gewerbestrasse 11, 6330 Cham, Switzerland, 2018.

[5] Artifical Pancreas Trondheim. `https://www.apt-norway.com/` (accessed: 18.11.2021).

[6] T. Fernando F. Chee. *Closed-Loop Control of Blood Glucose.* Springer-Verlag Berlin Heidelberg, 2007.

[7] W. K. Ward J. R. Castle. Amperometric glucose sensors: Sources of error and potential benefit of redundancy. *Journal of Diabetes Sicence and Techonolgy*, 4(1):221–225, 2010.

[8] Odd Martin Staal, Steinar Sælid, Anders Fougner, and Øyvind Stavdahl. Kalman smoothing for objective and automatic preprocessing of glucose data. *IEEE Journal of Biomedical and Health Informatics*, 23(1):218–226, 2019.

[9] World Health Organization. Blood gas/ph/chemistry point of care analyzer. `https://www.who.int/medical_devices/innovation/blood_gas_analyzer.pdf` (accessed: 11.12.2021), 2011.

[10] Anders Lyngvi Fougner. TTK26 lysark diabetes glukoseregulering måling. `https://ntnu.blackboard.com` (accessed: 4.12.2021), 2021.

[11] Wikipedia the free encyclopedia. Arterial blood gas test. `https://en.wikipedia.org/wiki/Arterial_blood_gas_test` (accessed: 11.12.2021), 2021.

[12] B.W. Bequette. Optimal estimation applications to continuous glucose monitoring. In *Proceedings of the 2004 American Control Conference*, volume 1, pages 958–962, 2004.

[13] Claudio Cobelli Andrea Facchinetti, Giovanni Sparacino. Reconstruction of glucose in plasma from interstitial fluid continuous glucose monitoring data: Role of sensor calibration. *IEEE Transactions on Biomedical Engineering*, 1(3):671–623, 2007.

[14] O. Ostertag E. Ostertagovà. Forcasting using simple exponential smoothing method. *Acta Electrotechnica et Informatica*, 12(3):62–66, 2012.

[15] Morten D. Pedersen. TTK4115 lecture 3/4. `https://ntnu.blackboard.com` (accessed: 12.12.2021), 2019.

[16] Inna N. Smith. Controllability, observability and realizability. *Electronic Theses and Dissertations*, 2005.

[17] William J.Emery Richard E.Thomson. *Data Analysis Methods in Physical Oceanography.* Elsevier Science Publishing Co. Inc., 3 edition, 2014.

[18] D. Simon. *Optimal Estimation.* John Wiley Sons, Inc., Hoboken, 2006.

[19] Patrick Y. C. Hwang Rober Grover Brown. *Introduction to Random Signals and Applied Kalman Filtering.* John Wiley Sons Inc., 4 edition, 2012.

[20] A. B. Onana S. Nowakowski M. Darouach, M. Zasadzinski. Kalman filtering with unknown inputs via optimal state estimation of singular systems. *International Journal of Systems Science, Taylor Francis*, 26(10):2015–2028, 1995.

[21] J.B. Burl. Singular system - a tutorial. In *Nineteeth Asilomar Conference on Circuits, Systems and Computers, 1985.*, pages 529–533, 1985.

[22] B. A. Foss J. G. Balchen, T. Andresen. *Reguleringsteknikk.* Institutt for teknisk kybernetikk, Odd Bargstads plass 2D, 7491 Trondheim, 6 edition, 2016.

[23] O. M. Staal. kalman-smoothing glucose. `https://github.com/omstaal/kalman-smoothing-glucose` (accessed: 7.10.2021), 2017.

[24] G. Voskanyan G. M. Steil D. B. Keenan, J. J. Mastrototaro. Delays in minimally invasive continuous glucose monitoring devices: A review of current technology. *Journal of Diabetes Science and Technology*, 3(5):1207–1214, 2009.

Appendix A Source code from Matlab

The code `Matlab` code files can be found in attachments, in a zip-file. Here is a list of what is included in the zip-file, as well as a short description:

- `main.m`, the main file of the code. Reads in the data, runs the estimation functions, runs the performance score function and plots the estimate together with the measurements.

- `standardKF.m`, computes the KF estimate, as well as builds the KF model of the system. The model is built by `setDynModel(...)` which is a function included in the file.

- `unknownInputKF.m`, computes the UIKF estimate, as well as builds the UIKF model of the system. The model is built by `setDynModel(...)` which is a function included in the file. The file also contains a function for checking the UIKF assumptions, `checkAssumptions(...)`

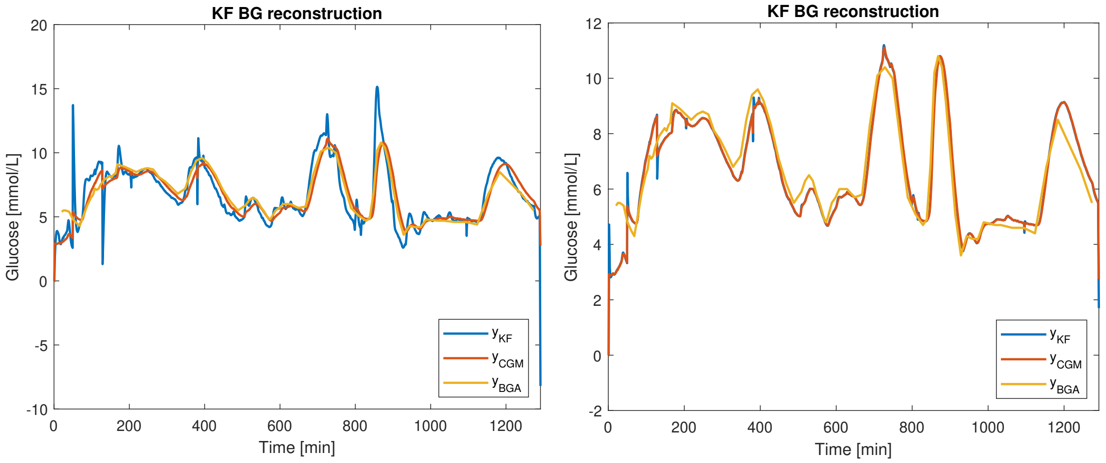
- `invTF.m`, computes the inverted TF estimate of the diffusion model. Note: Not developed in this project, but adjusted to fit in this project.

- `computeError.m`, calculates MAE, MSE, RMSE, MAPE of an estimate.

- `convertToRelativeTime.m`, converts date time data type to relative time data type. Borrowed from the open Git repository in [3].

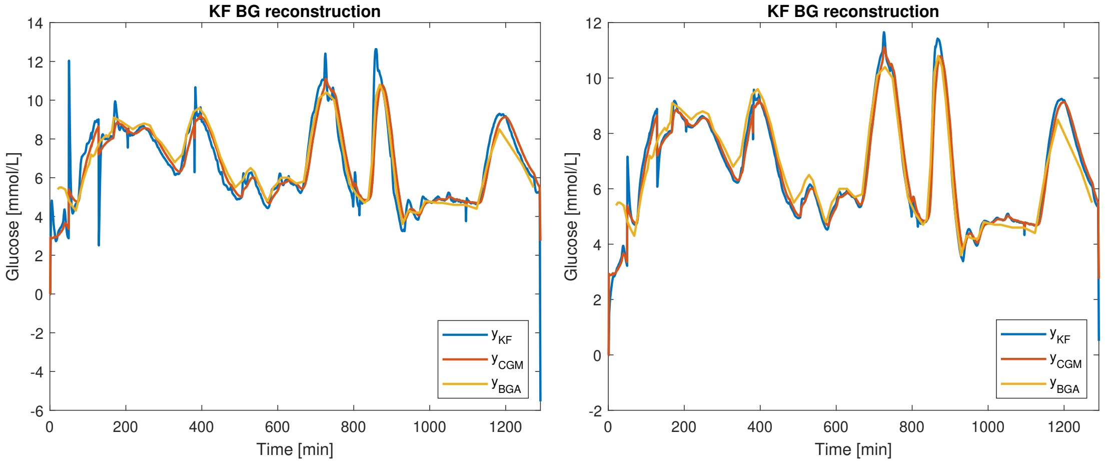
Appendix B Tuning Results

All following plots are of the Mar 5th 2021 data set.



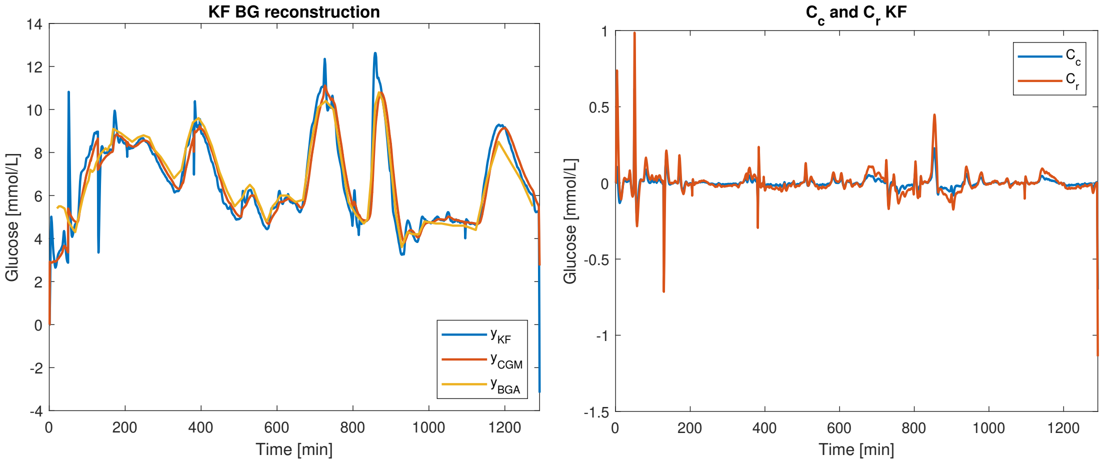
(a) $T_{isf} = 20$ [min] **(b)** $T_{isf} = 1$ [min]

Figure 21



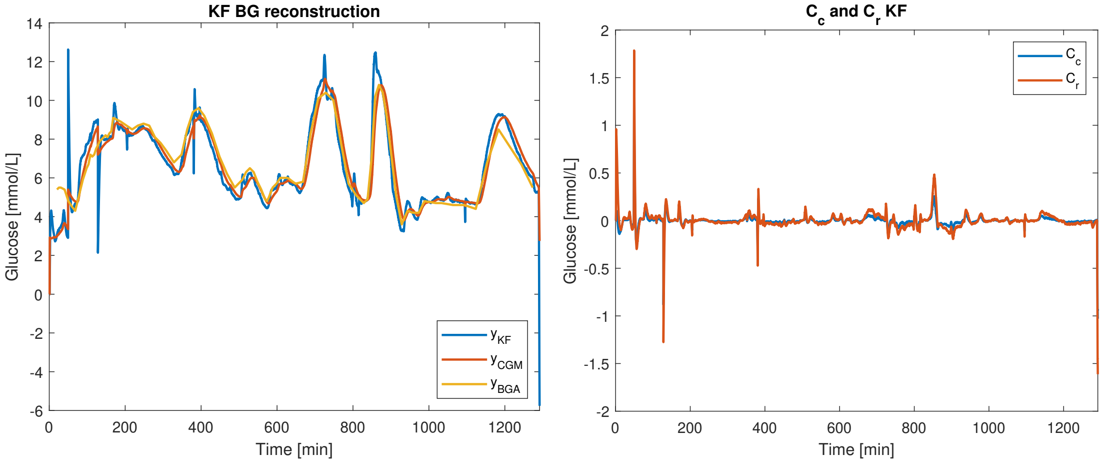
(a) $T_d = 20$ [min] **(b)** $T_d = 1$ [min]

Figure 22



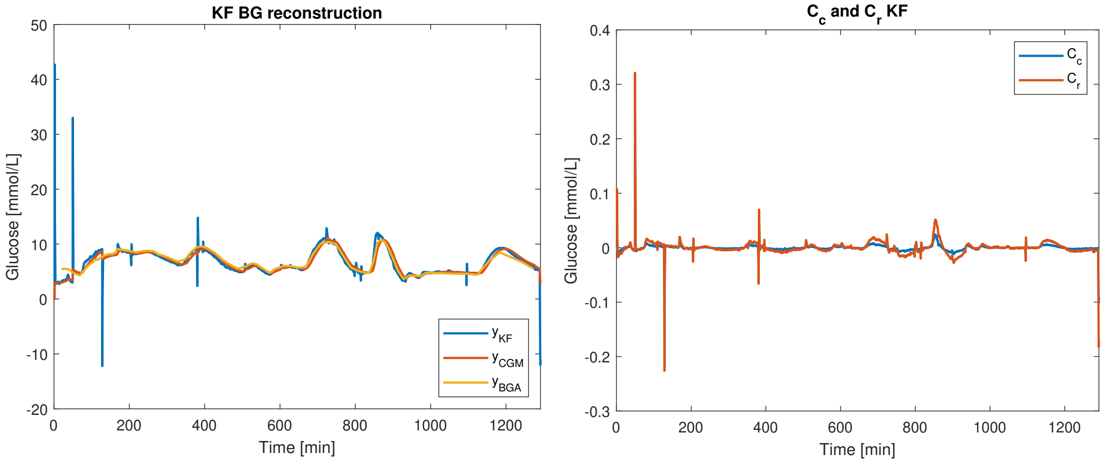
(a) G_p estimate. **(b)** C_c and C_r estimate.

Figure 23: $R = 20$.



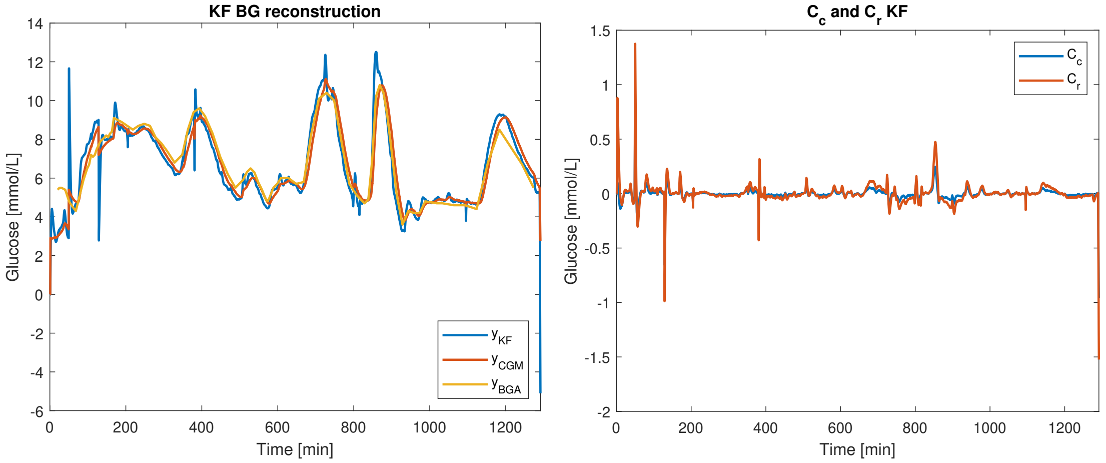
(a) G_p estimate. **(b)** C_c and C_r estimate.

Figure 24: $R = 0.0001$.



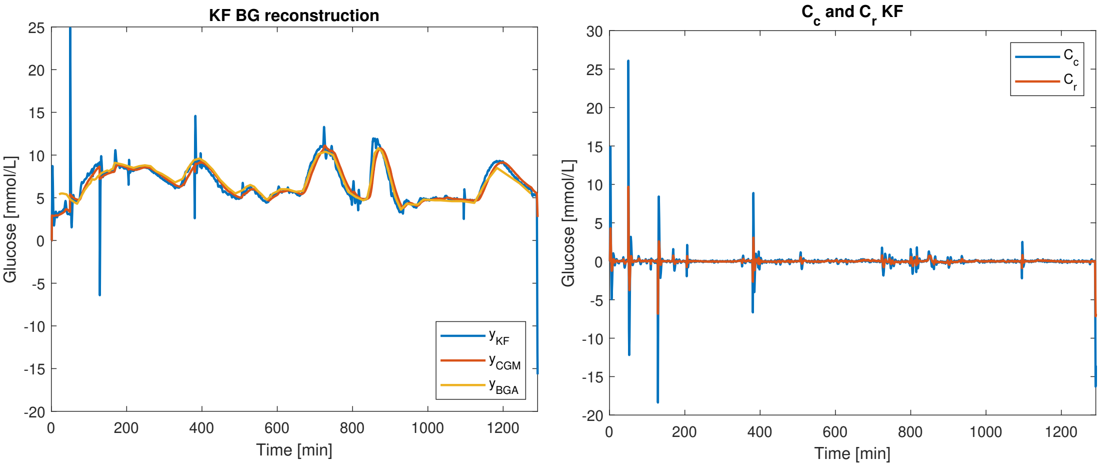
(a) G_p estimate. **(b)** C_c and C_r estimate.

Figure 25: $\sigma^2_{G_p} = 10$.



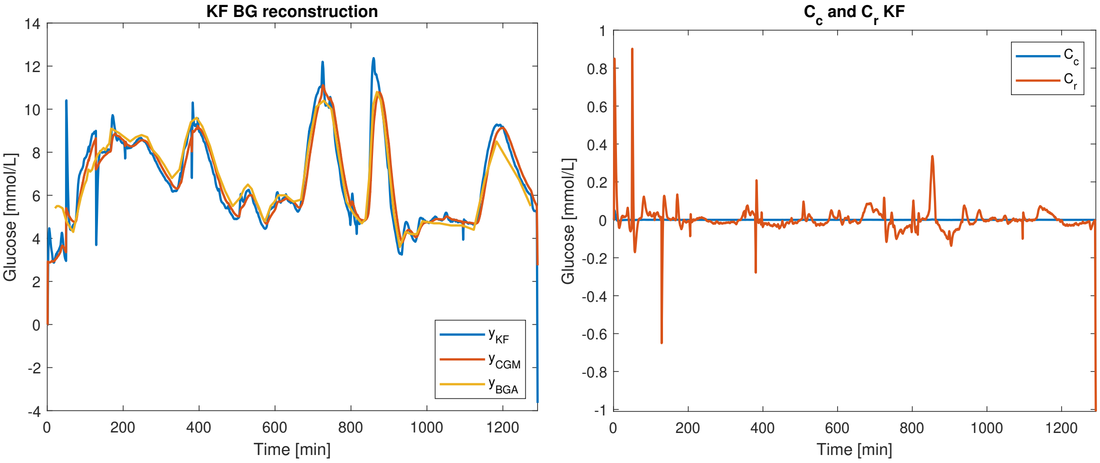
(a) G_p estimate. **(b)** C_c and C_r estimate.

Figure 26: $\sigma^2_{G_p} = 0.0001$.



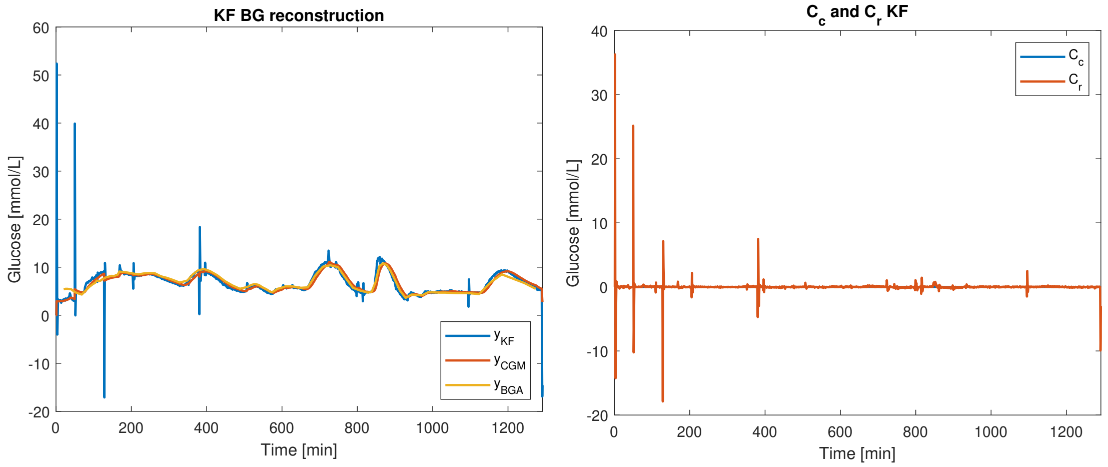
(a) G_p estimate. **(b)** C_c and C_r estimate.

Figure 27: $\sigma^2_{C_c} = 10$.



(a) G_p estimate. **(b)** C_c and C_r estimate.

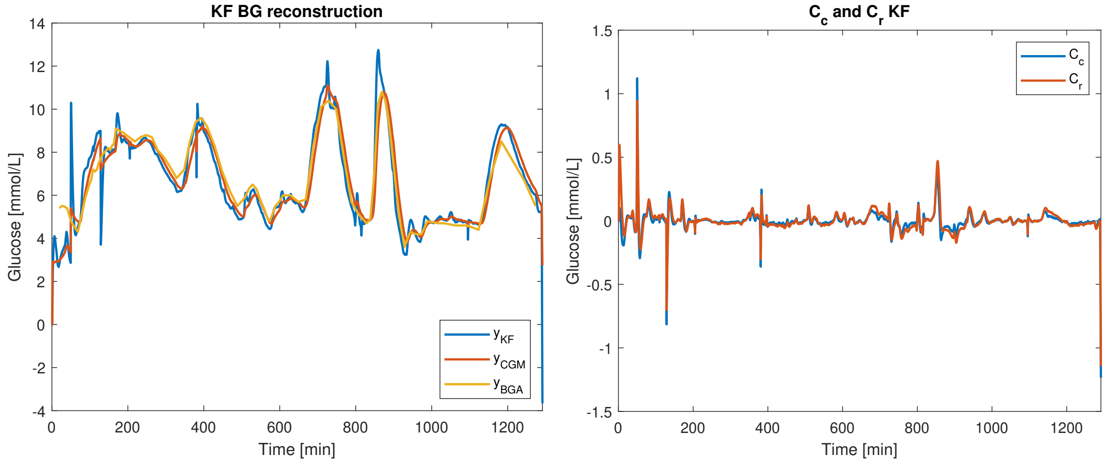
Figure 28: $\sigma^2_{C_c} = 0.0001$.



(a) G_p estimate.

(b) C_c and C_r estimate.

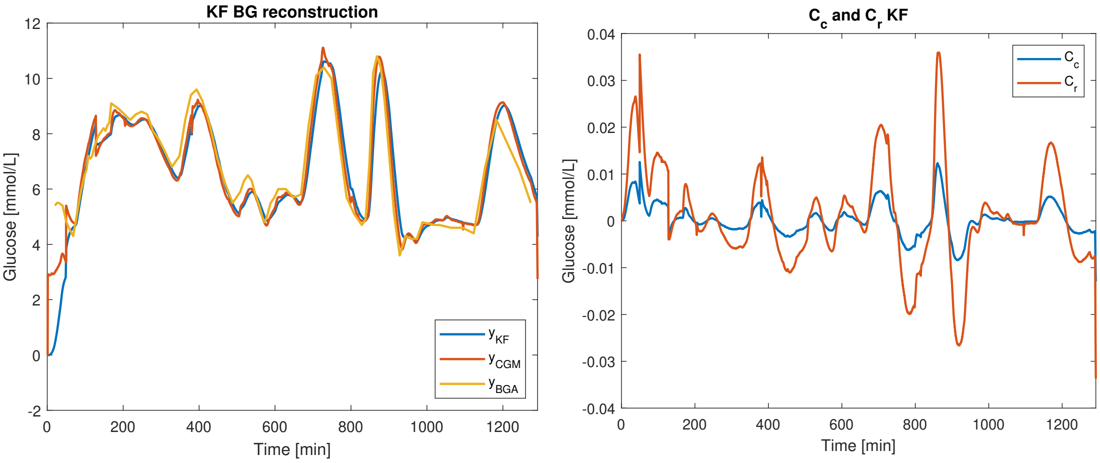
Figure 29: $\sigma^2_{C_r} = 10$.



(a) G_p estimate.

(b) C_c and C_r estimate.

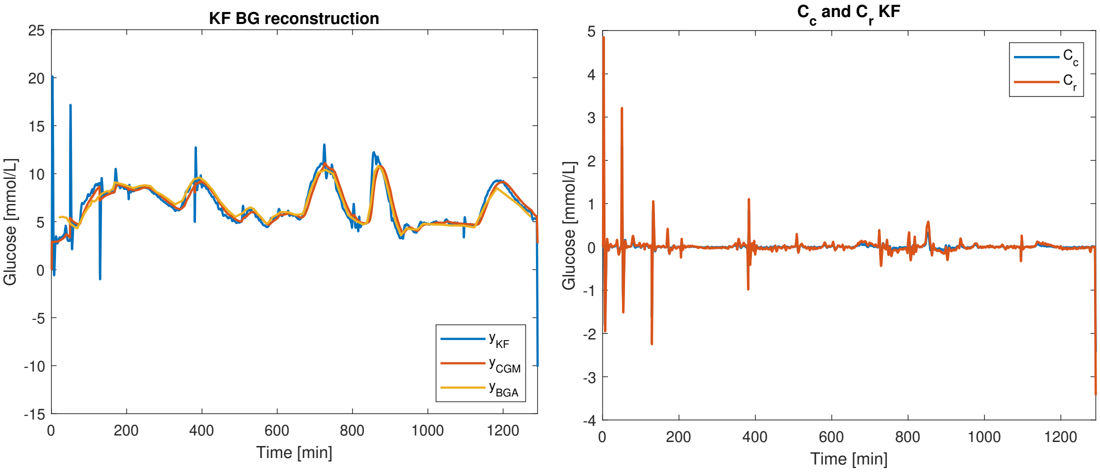
Figure 30: $\sigma^2_{C_r} = 0.0001$.



(a) G_p estimate.

(b) C_c and C_r estimate.

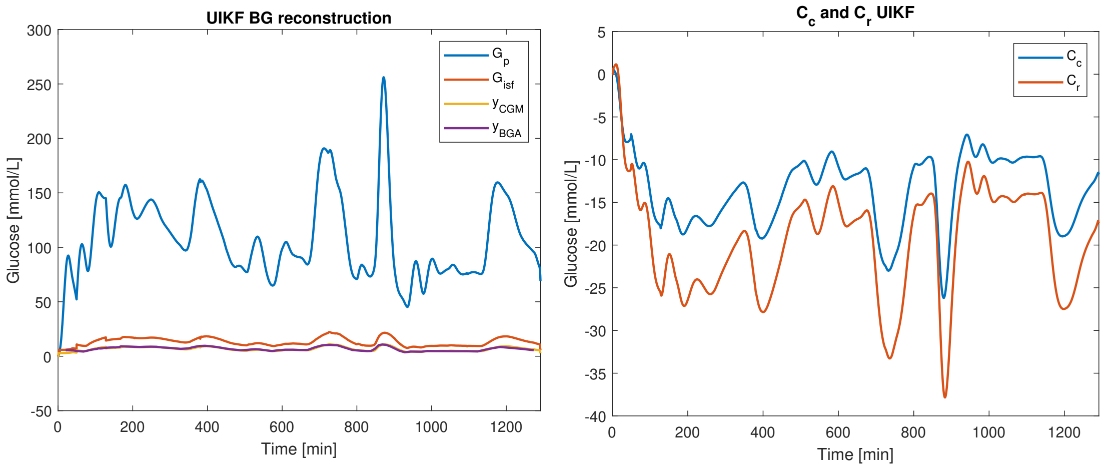
Figure 31: $\sigma^2_{G_{isf}} = 10$.



(a) G_p estimate.

(b) C_c and C_r estimate.

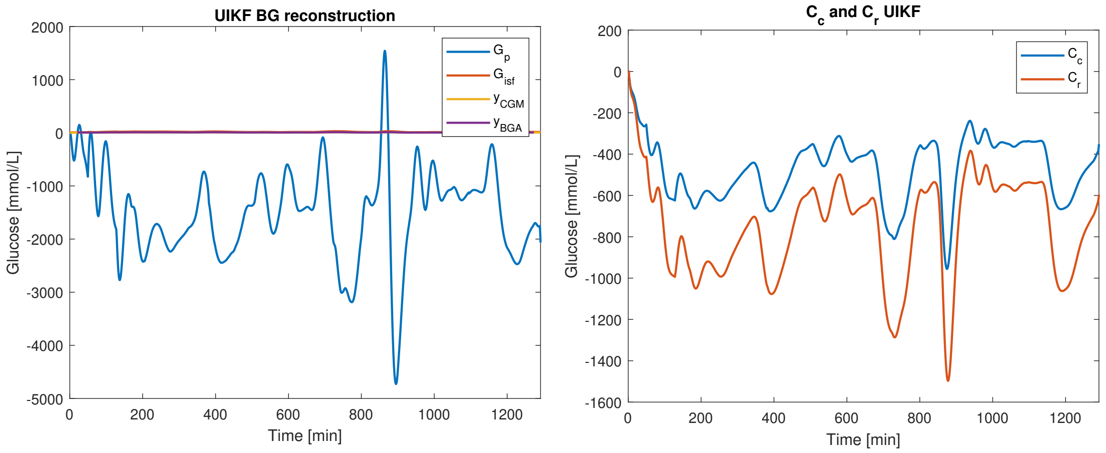
Figure 32: $\sigma^2_{G_{isf}} = 0.0001$.



(a) G_p estimate.

(b) C_c and C_r estimate.

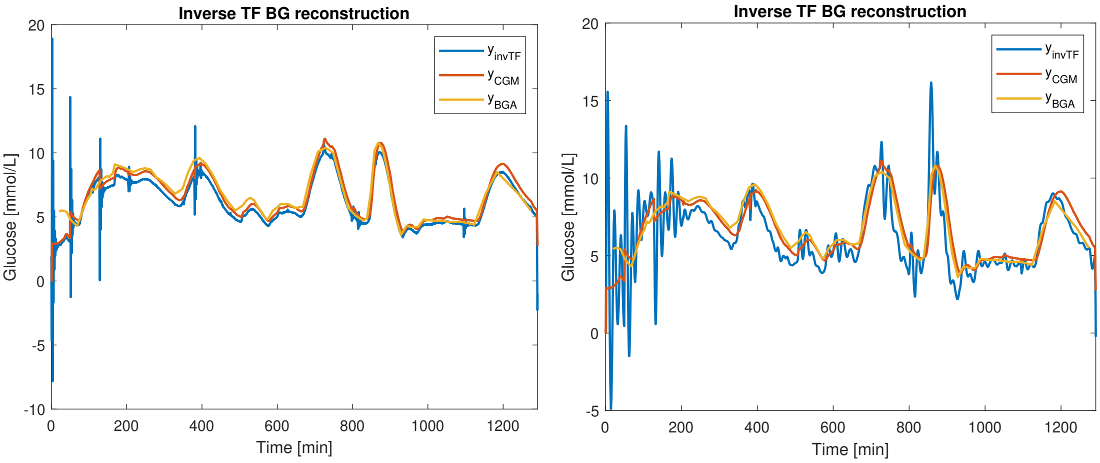
Figure 33: $R = 100$.



(a) G_p estimate.

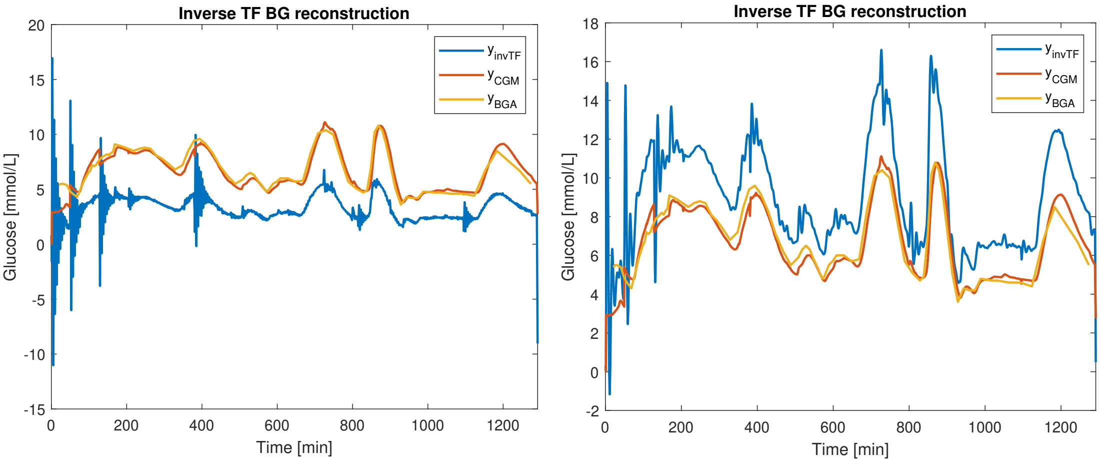
(b) C_c and C_r estimate.

Figure 34: $\sigma^2_{G_{isf}} = 10$.



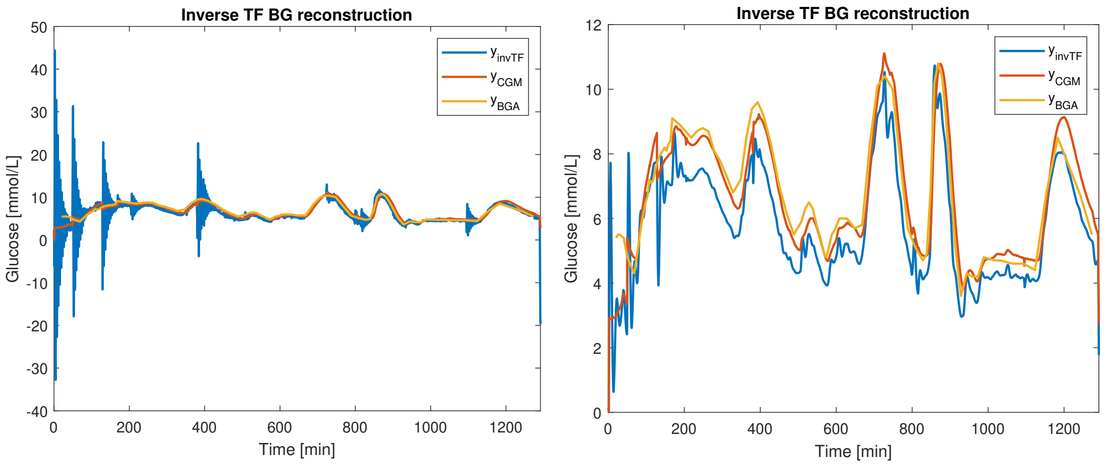
(a) $a = 0.7$. **(b)** $a = 0.1$

Figure 35



(a) $b = 2$. **(b)** $b = 0.8$

Figure 36



(a) $\alpha = 200$.

(b) $\alpha = 10$

Figure 37