<table>
<thead>
<tr>
<th>Title</th>
<th>A Probabilistic Framework Based on Mathematical Models with Application to Medical Data Streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>Enright, Catherine</td>
</tr>
<tr>
<td>Publication Date</td>
<td>2012-08-01</td>
</tr>
<tr>
<td>Item record</td>
<td><a href="http://hdl.handle.net/10379/3572">http://hdl.handle.net/10379/3572</a></td>
</tr>
</tbody>
</table>

Some rights reserved. For more information, please see the item record link above.
A Probabilistic Framework
Based on Mathematical Models
with
Application to Medical Data Streams

Catherine Enright, B.Eng.
College of Engineering and Informatics
National University of Ireland, Galway.

A thesis submitted for the degree of
Doctor of Philosophy
August, 2012

Ph.D. Supervisor: Dr. Michael Madden
## Contents

Abstract ................................................................................................................................. IV
Acknowledgements ................................................................................................................. V
Abbreviations ......................................................................................................................... VI

1. Introduction ......................................................................................................................... 1
   1.1 Motivation ....................................................................................................................... 2
   1.2 Proposed Solution ......................................................................................................... 4
   1.3 Thesis Structure ........................................................................................................... 5
   1.4 Summary of Contributions ........................................................................................ 7
   1.5 Publications from this Thesis ..................................................................................... 8

2. Background ......................................................................................................................... 10
   2.1 Bayesian Networks ...................................................................................................... 10
   2.2 Dynamic Bayesian Networks ..................................................................................... 12
   2.3 Particle Filtering .......................................................................................................... 17
   2.4 Glycaemia in Critical Care ........................................................................................ 19
   2.5 Conclusions .................................................................................................................. 24

3. Related Research ............................................................................................................... 25
   3.1 Artificial Intelligence in Critical Care Medicine ......................................................... 25
   3.2 BNs and DBNs in Critical Care Medicine ................................................................. 28
   3.3 DBNs Incorporating ODEs and SDEs ........................................................................ 29
   3.4 Handling the Inefficiency of Fixed Time Step Inference. ........................................ 30
   3.5 Modelling Glycaemia ................................................................................................ 32
   3.6 Conclusions .................................................................................................................. 36
4. Mapping ODEs to DBNs ................................................................. 37
   4.1 Methodology for Constructing DBNs from ODEs ....................... 38
   4.2 Expanding the DBN to Represent Measurement Uncertainty .......... 40
   4.3 Expanding the DBN to Re-estimate Parameters ...................... 40
   4.4 Selection of DBN Parameters .................................................. 41
   4.5 Evaluation on an Abstract Model ........................................ 41
   4.6 Conclusions ........................................................................ 52
5. DBNs Incorporating an RK2 Solver .................................................. 54
   5.1 Methodology for Constructing RK2 DBN ................................ 54
   5.2 Evaluation ........................................................................... 56
   5.3 Conclusions ........................................................................ 63
6. Adaptive-Time Particle Filtering ........................................................ 65
   6.1 Adaptive-Time Particle Filtering ........................................... 65
   6.2 Choosing the Step Size .......................................................... 69
   6.3 Evaluation ........................................................................... 72
   6.4 Full Steps and Sub-Steps ....................................................... 88
   6.5 Conclusions ........................................................................ 89
7. A DBN to Model Glycaemia ............................................................. 91
   7.1 The System of ODEs ............................................................. 91
   7.2 The DBN Derived from the ODEs ......................................... 93
   7.3 Analysis of Number of Particles Required ............................ 96
   7.4 Evaluation ........................................................................... 97
   7.5 Limitations of the ICU-MM .................................................. 108
   7.6 The ICING DBN .................................................................. 111
Abstract

This thesis presents a new probability-based framework which exploits existing domain knowledge in the form of mathematical models and uses real-time data streams to individualise model parameters.

We apply this new framework to personalising drug dosage regimes in critically ill patients, using routinely collected bedside and lab data. However, it can equally be applied to engineering, medical and scientific problems where noisy temporal data must be analysed and domain knowledge in the form of ordinary differential equation (ODE) models is available.

The first contribution of this thesis is a methodology for incorporating ODEs in a Dynamic Bayesian Network (DBN) framework. By doing this, we can handle data and model uncertainty in a principled manner, perform temporal data mining with noisy and missing data, and can individualise model parameters to the patient automatically and continuously, using data streams.

For this first contribution a first-order Euler solver is encapsulated in the DBN framework. ODEs may be approximated more efficiently using higher-order solvers. A second contribution is therefore a methodology for encapsulating a higher-order solver and an assessment of when this is preferable.

The third contribution is a new Adaptive-Time Particle Filtering algorithm for performing efficient inference on our DBN framework. In standard DBN particle filtering, time steps are of fixed length. Our new algorithm allows the time step length to adapt automatically in each step, greatly improving the efficiency of inference.

The final contribution is to apply the above techniques to the problem of regulating glycaemia in intensive care patients. Hyperglycaemia is a common phenomenon in critical care and is associated with increased morbidity and mortality. However, critically ill patients have highly variable responses to the insulin that is administered to regulate glycaemia. A solution such as the one described here, that can individualise the dosage, could therefore be of great benefit.
Acknowledgements

This research project would not have been possible without the support of many people. Firstly, I would like to express my gratitude to my Ph.D supervisor, Dr. Michael Madden. He always found the time and energy to guide and challenge me. He continually encouraged me to delve deeper and this search for clarity has immensely contributed to this research. His support is greatly appreciated.

For this project, I had the pleasure of collaborating with Dr. Niall Madden, Dr. Petri Piiroinen, Liam O’Callaghan and Nhan Anh Thai from the School of Mathematics, Statistics & Applied Mathematics, National University of Ireland, Galway. I am grateful to them all for their insight, ideas and curiosity. I am particularly grateful to Dr. Niall Madden for his patience in explaining mathematical concepts to me and also for the ideas he contributed to the overall project.

Our collaborators from University Hospital Galway, Prof. John Laffey, Dr. Brian Harte and Ms. Anne Mulvey were incredibly generous with their time and knowledge. Their enthusiasm and drive to explore all avenues in an effort to improve the care of their patients is inspiring. I would also like to thank Conor Lane for spending a summer carefully collecting data for this project and the Galway Research Ethics committee for granting us permission to do so.

I would like to thank the members of my Graduate Research Committee, Prof. Gerard Lyons and Dr. Jim Duggan and the members of the Information Technology Discipline administrative and technical staff, Phil Keys, Tina Earls, Mary Hardiman, Joe O’Connell and Peter O’Kane.

I wish to thank Prof. Stuart Russell for permitting us to use the AIMA code, developed by his research group from the University of California, Berkeley, as a basis for this project and his input into our initial findings.

Finally, this project could not have been undertaken without the financial support of Science Foundation Ireland. The material in this thesis is based upon works supported by the Science Foundation Ireland under Grant No. 08/RFP/CMS1254.
# Abbreviations

The following is a list of the key abbreviations used in this thesis:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>BN</td>
<td>Bayesian Network</td>
</tr>
<tr>
<td>DBN</td>
<td>Dynamic Bayesian Network</td>
</tr>
<tr>
<td>ICU</td>
<td>Intensive Care Unit</td>
</tr>
<tr>
<td>ODE</td>
<td>Ordinary Differential Equation</td>
</tr>
<tr>
<td>ABG</td>
<td>Arterial Blood Gas</td>
</tr>
<tr>
<td>ICU-MM</td>
<td>Intensive Care Unit – Minimal Model</td>
</tr>
<tr>
<td>ICING</td>
<td>Intensive Control Insulin-Nutrition-Glucose Model</td>
</tr>
<tr>
<td>UHG</td>
<td>University Hospitals Galway</td>
</tr>
<tr>
<td>I.V.</td>
<td>Intravenous</td>
</tr>
<tr>
<td>IVP</td>
<td>Initial Value Problem</td>
</tr>
</tbody>
</table>
1. Introduction

This thesis presents a new probability-based framework which exploits existing domain knowledge in the form of mathematical models and uses real-time data streams to individualise model parameters.

We assert that mathematical models can be considered sufficient statistics of prior experimentation in a domain and that when used in a probabilistic framework, they provide a sound foundation for data mining, inference and decision making under uncertainty.

The first key contribution of this thesis is a methodology for encapsulating mathematical models in the form of ordinary differential equations (ODEs) in a Dynamic Bayesian Network (DBN) framework. (DBNs are explained in detail in Section 2.2.) This contribution is important because the proposed DBN framework can

- handle both data and model uncertainty in a principled manner,
- be used for temporal data mining with noisy and missing data, and
- be used to re-estimate model parameters automatically and continuously using data streams.

The knowledge engineering effort involved in building DBNs can be significantly reduced by basing them on readily available ODE models.

A second contribution is a methodology for encapsulating a second-order solver in the DBN framework and an assessment of when this is preferable. The first contribution incorporates a first-order Euler solver. However, some ODEs can be approximated more efficiently using higher-order solvers. It is therefore important to
establish how a higher-order solver can be incorporated and when it is appropriate to do so.

A third key contribution of this thesis is a new Adaptive-Time Particle Filtering algorithm for performing inference on our DBN framework. The importance of this contribution lies in the improved efficiency of the inference task, especially where the dynamics of the system are not uniform over time or the underlying ODEs are stiff.

The fourth and final contribution is an application of the above techniques to the problem of regulating glycaemia in intensive care patients. As will be explained in Section 2.4, controlling glycaemia in critically ill patients is of great importance to clinicians as both hyperglycaemia and hypoglycaemia have negative consequences for patients. Critically ill patients have highly variable responses to the insulin that is administered to regulate glycaemia. Techniques, such as the ones described in this thesis, that can personalise the insulin dosage regimes, using routinely collected bedside and lab data, could therefore be of great benefit.

While we apply our techniques to personalising drug dosages they can equally be applied to engineering, medical and scientific problems where noisy temporal data must be analysed and domain knowledge in the form of ordinary differential equation (ODE) models is available.

1.1 Motivation

Work on this thesis was carried out as part of a wider project, funded by Science Foundation Ireland under Grant No. 08/RFP/CMS1254. The overall project goal was to develop techniques to individualise drug dosage using the relevant data readily available in an Intensive Care Unit (ICU). It was an interdisciplinary project. Liam O’Callaghan and Dr. Petri Piiroinen, worked on developing new mathematical models to describe the glucoregulatory system in critically ill patients. Nhan Anh Thai and Dr. Niall Madden examined suitable numerical methods to efficiently solve
the systems of ODEs. John Laffey, Professor of Anaesthesia and Intensive Care, was our clinical collaborator. Under the supervision of Dr. Michael Madden, my role was to develop and implement the DBN framework.

Pharmacokinetics is the study of the movement of a drug through the body as it is absorbed, distributed, metabolised and excreted. Pharmacodynamics is the study of how a drug affects the body. Mathematical models already exist to describe the pharmacokinetics and pharmacodynamics of all drugs; the textbook by Shargel et al. [1] for example, provides a good introduction to the basic models. They are developed during clinical trials, published for all approved drugs and used to determine safe dosage regimes for the average patient. However the critically ill patient’s unique parameters can vary considerably from those of the average patient.

Many drugs have a very narrow therapeutic range, i.e. too high a dose can cause toxicity, whereas too low a dose will not have the desired therapeutic effect. However, because the ICU is a highly controlled environment, it is also data-rich. This project aims to exploit the potential of this data for individualising model parameters and therefore determine suitable dosage regimes.

In collaboration with clinicians in University Hospital Galway, insulin was identified as a drug for which an individualised dosage regime would be useful. Insulin is used to regulate glycaemia. Stress-induced hyperglycaemia is a common phenomenon in critical care [2] and is associated with increased morbidity and mortality [3][4][5]. Proper control of glycaemia in critically ill patients in the ICU is therefore a subject that is of great importance to clinicians. As we will see in Section 2.4.2.1, there have been a number of studies carried out to determine the glucose range that should be targeted by clinicians; however these have produced conflicting results [6]. The optimal target range for blood glucose and the optimal approach to controlling blood glucose levels in critically ill patients is still unclear.

Current practice makes use of a sliding-scale based on the patient’s plasma glucose levels [7]. This method does not consider the individual’s response to earlier insulin
infusions nor does it consider other factors that influence glycaemia levels, such as the patient’s weight, severity of illness or other prescribed medication. More importantly, this method does not take into account the inter-individual and intra-individual variability in the patient response to administered insulin.

Building a framework capable of reasoning in the context of large uncertainty is the central objective of this project. However, as well as dealing with uncertainty, the temporal nature of the problem must also be addressed. A patient’s glucose level depends not only on the current intravenous (I.V.) infusion rates but also past infusion rates and past glucose levels. DBNs are an effective tool for modelling uncertainty in a temporal environment. Both Bayesian Networks (BNs) and DBNs have previously been used in the medical setting. However most of the applications we reviewed only use discrete variables (for example, [8][9]). Most DBNs are built manually [10] and traditionally knowledge elicitation is viewed as a bottleneck [11].

1.2 Proposed Solution

In many domains, for example, engineering and medicine, expert knowledge in the form of mathematical models is readily available [12][13]. We assert that knowledge in such a form can be considered sufficient statistics of all prior experimentation in the domain, embodying generic or abstract knowledge of it. When used in a probabilistic framework, we believe that such models provide a sound foundation for data mining, inference, and decision making under uncertainty.

Ordinary Differential Equation (ODE) models are generally available in mathematical [14], engineering [13] biological [15][16][17] and biomedical textbooks [18] [12] and research publications [19][20], and typically describe general population-level behaviours. In order to describe individuals, model parameters must be re-calibrated using observations of the individual. However, in most real-life situations, these observations contain uncertainty. They may be noisy as they can be subject to measurement error and/or simple transcription errors. Data may be missing.
Also the frequency of the observations may be sparse relative to the dynamics of the underlying system thus making it difficult to individualise the parameters.

In this thesis, we propose a DBN framework that can handle both data and model uncertainty in a principled manner, can be used for temporal data mining with noisy and missing data, and can be used to re-estimate model parameters automatically using data streams. We can significantly reduce the knowledge engineering effort involved in building DBNs by basing them on readily available ODE models and a systematic approach for doing so is presented.

A standard assumption when performing inference in DBNs is that time steps are fixed. Generally, the time step chosen is small enough to capture the dynamics of the most rapidly changing variable. This can result in DBNs having a natural time step that is very short, leading to inefficient inference; this is particularly an issue for DBNs derived from ODEs and for systems where the dynamics are not uniform over time.

Therefore, we propose a new algorithm, called Adaptive-Time Particle Filtering, as an alternative to the standard fixed time step particle filtering. In Adaptive-Time Particle Filtering, the DBN automatically adapts the time step lengths to suit the dynamics in each step. The resulting system efficiently infers probable values of hidden variables using multiple time series of evidence, some of which may be sparse, noisy or incomplete.

This new framework along with the new particle-filtering algorithm is applied to the problem of modelling glycaemia in ICU patients and validated on real patient data.

1.3 Thesis Structure

In Chapter 2, a brief background is presented on DBNs and particle filtering, as they form the basis of the proposed solution. The problem to which we apply our techniques is that of modelling glycaemia in the ICU. This is also introduced in Chapter 2.
Chapter 3, on related research is subdivided to address four different questions:

1. Have others applied DBNs or indeed BNs in critical care medicine?
2. Have others incorporated ODEs in DBNs?
3. How have others dealt with the inefficiency of fixed time step inference?
4. How have other researchers attempted to model Glycaemia in the ICU?

In Chapter 4, a direct automatic mapping from ODEs to a DBN by incorporating a first order Euler approximation is presented.

In Chapter 5, we demonstrate how higher order solvers can be used and if and when this may be more appropriate is discussed. Both approaches are validated on abstract models for which we have the true solution.

The novel Adaptive-Time Particle Filtering algorithm is presented in Chapter 6. This algorithm is evaluated on a DBN based on a variant of the van der Pol oscillator with parameters chosen to create a stiff problem. We show how the DBN framework can be used to track the dynamic system and re-estimate model parameters using observations from the true solution. We demonstrate an example where this Adaptive-Time Particle Filtering algorithm gives more accurate results than the standard fixed time step approach, but using only one fifth of the number of time steps (see Section 0).

In Chapter 7, our approach is applied to the motivating problem from critical care medicine. An existing system of ODEs is incorporated into a DBN in order to build a system to assist in prescribing the correct insulin dosage. Using only the sparse noisy data streams available at the bedside, the DBN framework efficiently predicts glucose levels, and the predictions are more accurate than an existing methodology.

Finally in Chapter 8, we discuss the overall benefits of the new methodologies introduced in this thesis, their effectiveness and indeed limitations when applied to the problem of modelling glycaemia in critically ill patients.
1.4 **Summary of Contributions**

To summarise this chapter, the key contributions of this work are as follows:

1) A new methodology for mapping systems of ordinary differential equations into a Dynamic Bayesian Network. The proposed framework encapsulates a first-order Euler solver, can account for data and model uncertainty and can be used to individualise model parameters over time using temporal data streams.

2) A methodology for encapsulating a second-order ODE solver in a DBN framework. This framework can be of use when the underlying ODE model is stiff. Like the first-order framework, data and model uncertainty can be handled in a principled manner and model parameters can be individualised using real-time observations.

3) A new Adaptive-Time Particle Filtering algorithm. This new algorithm offers significant efficiency improvements when compared to traditional fixed-time step particle filtering. These efficiency improvements are of most benefit when performing inference on fast-slow systems or where the underlying ODE model is stiff.

4) A new framework for modelling glycaemia as it changes in response to insulin and glucose infusions in critically ill patients. This new framework can be used by clinicians to individualise dosage regimes and monitor patient behaviour.
### 1.5 Publications from this Thesis

Key aspects of this thesis have already been published at peer-reviewed conferences or are under review in a peer-reviewed journal.


   Parts of this publication appear in Chapters 5, 6 and 7 of this thesis.


   Parts of this publication appear in Chapter 4 and Chapter 7 of this thesis.


   Parts of this publication appear in Chapter 7 of this thesis.


   Parts of this publication appear in Chapter 7 of this thesis.

Aspects have also been published in non-peer reviewed conferences:

2. **Background**

The purpose of this chapter is to provide the necessary context and background for the topics discussed in subsequent chapters of this thesis.

The chapter is divided into four sections. Firstly Bayesian Networks are introduced as they form the basis of Dynamic Bayesian Networks which are then introduced in Section 2.2. For this thesis, inference is carried out using Particle Filtering; the algorithm used for this work is described in Section 2.3. Finally the problem domain, modelling glycaemia in critically ill patients, to which the techniques developed for this thesis are applied, is explained in Section 2.4.

2.1 **Bayesian Networks**

Pearl [21] succinctly describes Bayesian Networks as “directed acyclic graphs in which the nodes represent variables, the arcs signify the existence of direct causal influences between the linked variables, and the strengths of these influences are expressed by forward conditional probabilities”.

In Bayesian Networks each variable to be considered is represented by a node. Bayesian Networks can be considered a representation of independence relationships between the nodes. If an arc exists from node $X$ to node $Y$, $X$ is said to be a parent of $Y$. The absence of an arc between two nodes indicates they do not influence each other directly and are conditionally independent. In a Bayesian Network, each node is conditionally independent of its non-descendants in the graph given the values of all its parents.

Many BN structures can represent the same independence relationships. However, it is recommended [22] [23] that the structure reflects causal order so that causes are parents of effects. Non-causal ordering can require more arcs and result in a less compact topology [22] [23].
Each node $X_j$ has an associated conditional probability distribution $P(X_j|\text{Parents}(X_j))$ to quantify the effect of the parents on the node. These distributions are referred to as the conditional probability tables (CPTs).

The network structure and the conditional probability tables can be viewed as a representation of the full joint probability distribution for each variable. Each entry in the joint distribution is represented by the product of the appropriate elements of the conditional probability tables. The joint probability of a set of variables can be determined by

$$P(x_1, \ldots, x_n) = \prod_{j=1}^{n} P(x_j|\text{parents}(X_j))$$  

where $x_1$ denotes a particular assignment to the variable $X_j$ and $\text{parents}(X_j)$ denotes the specific values of the variables in $\text{Parents}(X_j)$.

Nodes in a Bayesian Network may be continuous or discrete. Nodes can be either observed or hidden. One purpose of a BN can be to infer probable values for the hidden nodes. Observed nodes are nodes for which we have an observation of its true state, e.g. a measurement for a patient’s glucose level at a particular time. Hidden nodes cannot be directly observed, e.g. the patient’s sensitivity to insulin. We can, however, infer a probable value for the patient’s insulin sensitivity using the measured glucose levels.

The graphical representation used in this thesis is shown in Figure 1. In this representation observed nodes are shaded black and hidden nodes are shaded grey.
Figure 1: Basic concepts and representation of a Bayesian Network. Hidden nodes are shown here in grey. Observed nodes are shown in black.

The frequently used Naïve Bayes model (see for example [22]) is a particular class of Bayesian Network used where a single class node is a parent of a number of evidence nodes. All evidence nodes are assumed to be conditionally independent of each other, given the class node. A Tree-augmented Naïve Bayes (TAN) Classifier [24] is a classifier without the conditional independence restriction between the evidence nodes. Arcs are allowed between the evidence nodes. However evidence nodes can have the class node and at most one other node as parents. In this way, evidence nodes form a tree structure. Both Naïve Bayes and TAN are popular in Machine Learning for classification applications. The paper by Madden [25] offers a good comparison between Naïve Bayes, TAN and General Bayesian Networks.

2.2 Dynamic Bayesian Networks

DBNs, which are the subject of this thesis, are Bayesian Networks that represent temporal systems, i.e. where the past state influences the current state.

DBNs were originally introduced by Dean and Kanazawa [26]. The purpose of a DBN is to infer probable values for the hidden variables as they evolve over time. DBNs discretise time and we denote each time step with the index $i$. If we un-roll the
DBN it yields a BN structure repeated over time, the joint distribution therefore follows from Equation (1) and over $T$ time-slices can be given by:

$$P(Z_{0:T}) = \prod_{i=0}^{T} \prod_{j=1}^{N} P(Z_i^j | \text{parents}(Z_i^j))$$

We partition the nodes into input nodes (denoted by $I_i$), hidden nodes (denoted by $X_i$), and observation nodes (denoted by $Y_i$). Input nodes are observed and are typically parents of the hidden $X_i$ nodes. The hidden nodes are in turn parents of the observed $Y_i$ nodes. In specifying a DBN three sets of probability distributions are required:

1. The initial state distribution $P(X_0)$,
2. The sensor/observation models $P(X_i | I_i)$ and $P(Y_i | X_i)$.
3. The transition model $P(X_{i+1} | X_i)$.

DBNs are first-order Markov in the sense that the state at time step $i + 1$ is assumed to depend only on the state at time step $i$. DBNs are generally assumed to have a fixed time step. The modeller chooses a natural fixed time step size in order to specify the model structure and the transitional probability tables.

In this thesis, DBNs are assumed to be time invariant, i.e., their structure does not change over time, as opposed to non-stationary DBNs proposed by Robinson and Hartemink [27]. Detailed discussions on DBNs can be found in [28],[23] and [22].

### 2.2.1 Graphical Representation

In subsequent chapters, DBNs are represented using a compact format shown on the left in Figure 2. Hidden nodes are shown in grey here; different shading for different node types is used later. Observed nodes are shown in black. Dependencies within a time-slice are indicated with a solid arc. Inter-slice dependencies are indicated by a dotted line.
On the right-hand side the “un-rolled” DBN is presented. From this version, it is clear that a DBN is a Bayesian Network with a repeating structure for each time slice, with arcs from one time slice to the next.

![Diagram of DBN](image)

Figure 2: Two different representations of a simple DBN. The compact representation on the left hand side is used in this thesis. Hidden nodes are shown here in grey. Observed nodes are shown in black. Dependencies within a time-slice are indicated with a solid arc. Inter-slice dependencies are indicated by a dotted line. The “un-rolled” representation on the right is also commonly used.

### 2.2.2 Inference

For the purpose of this thesis, we use the definition provided by Russell and Norvig [23] for inference, which is “the computation from observed evidence of posterior probabilities for query propositions”. There are four basic inference tasks that can be solved in a DBN [23]:

- **Filtering**: Keeping track of the current state. This is the task of computing the posterior distribution given all the evidence to date, i.e. $P(X_i | y_{1:i})$.
- **Prediction**: Computing the posterior distribution over future states, i.e. $P(X_{i+k} | y_{1:i})$ for some $k > 0$.
- **Smoothing**: Computing the posterior distribution over past states, i.e. $P(X_{k} | y_{1:i})$. for some $k > 0$ such that $0 \leq k < i$. The smoothed estimate will be more accurate as it incorporates more evidence.
• Most Likely Explanation: Determining the most likely sequence of states to have
generated a sequence of observations, i.e. $\text{argmax}_{x_{1:t}} P(x_{1:t} | y_{1:t})$.

For the purposes of this thesis we are interested in 1 and 2 above, Filtering and
Prediction. Exact inference is only possible in simple DBNs. For more complex
DBNs approximate filtering and prediction can be performed using a fixed time step
particle filtering algorithm [29], as explained in Section 2.3.

2.2.3 DBN Creation
Both DBN structure and parameters can be learned from data [28][30][31]. However
most DBNs are constructed by hand, using knowledge elicited from domain experts
[10]. This knowledge elicitation process is difficult and time consuming [32] [33]. In
this thesis, we propose that the knowledge elicitation bottleneck can be bypassed by
using existing summarised domain knowledge in the form of ordinary differential
equations.

2.2.4 Special Cases of DBNs
A Hidden Markov Model (HMM) is a special class of DBN with only one discrete
state variable and one observation variable. Every HMM can be represented as a
DBN and every discrete DBN can be reformulated as a HMM.

A Kalman Filter is a DBN particularly suited to modelling dynamic linear systems
with Gaussian noise. It assumes continuous variables and linear transition model
(DBNs on the other hand can have both discrete and continuous variables). Every
Kalman filter model is a DBN, but few DBNs are Kalman Filters. Extensions of
Kalman Filters have been developed to allow non-linear transition models however
these are outside the scope of this thesis.
2.2.5 Other BN Variants for Handling Temporal Relationships

2.2.5.1 Temporal Bayesian Networks of Events

Arroyo-Figueroa and Sucar introduced an alternative representation called Temporal Bayesian Network of Events (TBNE) [34]. In a TBNE each node represents an event or state change of a variable and an arc corresponds to a causal temporal relationship. A temporal node represents the time that a variable changes state, including an option of no-change. The main difference between a TBNE and a DBN is that the TBNE representation is based on state changes at different times, represented by temporal nodes, whereas the DBN representation is based on state values at different times, represented by state variables.

Figure 3 shows a simple example adapted from [34], whereby the level of a patient’s internal bleeding can be inferred from the time interval, following a collision, when vital signs become unstable. Severe internal bleeding will cause the vital signs to become unstable in 10-30 minutes whereas slight internal bleeding will cause the vital signs to become unstable in 30-60 minutes; these are modelled as separate states in the VS node.

![Diagram of TBNE](image)

Figure 3: The states in a sample TBNE adapted from [34]
2.2.5.2 Continuous Time Bayesian Networks

Similarly to TBNEs, Continuous Time Bayesian Networks (CTBNs) [35] also handle the timing of state changes.

In CTBNs, time is explicitly represented in intensity matrixes and not discretised as in DBNs. They are designed for querying the network for the distribution over the time when particular events of interest occur.

2.3 Particle Filtering

Particle Filtering is the sequential Monte Carlo method by which we infer the most probable states of the DBN nodes. In this work, a standard particle filtering algorithm developed for control theory by Gordon et al. [29] is used.

A particle contains a value for every node in the network in both the current and previous time slice. Thus, each particle has values instantiating all nodes in the network, where each node is consistent with all other nodes and with the evidence (nodes for which actual values are observed). Taken together, the particles can be used to assess most likely values of nodes and their distributions.

Where an observation exists at the current time slice, its value is used, otherwise a random value is chosen from the node’s distribution. Each particle is weighted to determine the probability of this set of node values given the observations. For each time slice the particles are summarized to return the weighted mean and weighted standard deviation for each node. Prior to processing the next time slice, the set of particles are “re-sampled”, i.e. individual particles are either multiplied or suppressed. The probability that a sample is selected is proportional to its weight. Higher-weight samples will spawn multiple copies, whereas lower-weight samples will die-off. The purpose of the re-sampling step is to focus the set of samples on the high-probability regions of the state space.
An alternative to re-sampling is evidence reversal [36]. Using evidence reversal, the time slice is restructured so that evidence nodes for a time slice become ancestors of state variables.

The implementation of the fixed-time-step particle filtering algorithm is part of the AIMA framework code being used for this project which was originally developed by Professor Stuart Russell and his research group from the University of California, Berkeley, and to which we added the new functionality detailed later in Chapter 6.

2.3.1 Description of the Particle-Filtering Algorithm

**Step 1:** A set of $S$ particles is created. Each particle is a vector with a slot per node for each of 2 time slices. (As the value of any particular node may depend on values in the previous time slice, we must keep track of the previous time slice).

**Step 2:** Each particle is populated with a set of probable values:

- Nodes are sampled one at a time in hierarchical order, i.e. parent nodes are sampled before child nodes (since children have conditional probability distributions that are conditioned on their parents’ values).
- If an observation for a node exists, this value is used; otherwise the node is sampled based on its parent’s values.

**Step 3:** Each particle is weighted based on the observations:

- The weight of a particle is the product of each of the conditional probabilities of the evidence values given the parent node values in the particle.

$$weight = \prod_j P(Evidence_j | Parents_j) \quad \text{over all nodes } j \text{ with evidence}$$

The weights are normalised.
Step 4: As there are $|S|$ particles, there are $|S|$ proposed values for each sampled node in a time slice. For each node the weighted mean and weighted standard deviation of the possible values are calculated.

Step 5: Prior to moving to the next time slice the set of particles is re-sampled:

- A set of new particles is selected from the current population of particles. The probability that a sample is selected is proportional to its weight.
- A particle contains values for all nodes for each of 2 time slices, $i - 1$ and $i$. At this point the values for time slice $i - 1$ are discarded.
- The process is repeated from Step 2. On entering Step 2 each particle vector only contains the values for time slice $i$. By sampling from the transition model, values time slice $i + 1$ are selected. Steps 2-5 are repeated for each time slice.

2.4 Glycaemia in Critical Care

This section commences with a brief background of glucoregulation in healthy humans and then proceeds to describe the problem of hyperglycaemia in ICU patients. Next, the difficulties in selecting and achieving an appropriate plasma glucose level are explained.

2.4.1 Insulin/Glucose Dynamics in Healthy Humans

Glucose can be considered the body’s fuel. Muscle and fat tissue cells require sufficient glucose to function. In a healthy human the pancreas produces two hormones, glucagon and insulin, to regulate blood sugar levels. Figure 4 is our simplified representation of the interaction between the insulin hormone and glucose in healthy humans.
Figure 4: Our simplified representation of the glucoregulatory system in a healthy human.

Following a meal, plasma glucose levels typically increase. When this increase is above a certain threshold, insulin is released by the pancreas. This insulin has the effect of lowering plasma glucose levels. It does this in two ways. Firstly, it stimulates glucose uptake by muscles and fat tissue – insulin is often viewed as the key that unlocks the tissue cells allowing glucose to enter the tissue. Secondly, insulin inhibits glucose release from the liver and stimulates glucose storage in the liver as glycogen.

When plasma glucose levels are too low, the pancreas activates the release of the hormone glucagon. Glucagon stimulates the liver to convert its stored glycogen to glucose and release this glucose to the blood stream.

It should be noted that glucose absorption in the brain is not insulin dependent.
Hyperglycaemia occurs when too much glucose is in the blood stream; hypoglycaemia occurs when too little glucose is in the blood stream. Both have negative consequences.

In the case of a Type I diabetic patient, either no insulin or insufficient insulin is produced by the pancreas.

2.4.2 **Hyperglycaemia in the ICU**

This thesis is concerned with critically ill patients as opposed to diabetic patients. In an ICU, patients often experience stress-induced hyperglycaemia [2]. Kavanagh and McCowen [37] provide an excellent summary of the clinical problem. Stress-induced hyperglycaemia in ICU patients is caused by increased concentrations of stress hormones (adrenaline, growth hormone, glucocorticoid and glucagon) [12], the use of medications such as exogenous glucocorticoids and catecholamines and the administration of intravenous dextrose, in parenteral nutrition and antibiotic solutions [2]. The consequences of elevated glucose levels may be manifested at the molecular or cellular level, combining to cause tissue abnormalities that include sepsis, impaired wound healing, and neuromyopathy (disease of both muscles and nerves) [37]. It has been shown that hyperglycaemia is associated with increased mortality and increased morbidity in critically ill patients [4].

To prevent hyperglycaemia in the ICU, exogenous insulin is administered. Due to its quick action and short half life, intravenous (IV) insulin is the preferred choice for rapid correction of hyperglycaemia [38]. Determining the optimal target range for blood glucose and the optimal approach to controlling blood glucose levels in critically ill patients is however still a matter for debate, as is discussed in the next section.

2.4.2.1 **What is the Optimal Glucose Range?**

Tight control of plasma glucose levels (80-110 mg/dl) has previously been demonstrated to improve outcome in a predominantly surgical population of
critically ill patients [4]. In contrast, the more recent NICE-SUGAR study found a blood glucose target of less than 180 mg/dl resulted in lower mortality than did a tight target of 81 - 108 mg/dl [39]. Kansagara et al. [6] recently conducted a review of 21 trials, including the two mentioned above, to evaluate the benefits and harms of intensive insulin therapy (IIT) in hospitalized patients. They conclude “No consistent evidence demonstrates that IIT targeted to strict glycaemic control compared with less strict glycaemic control improves health outcomes in hospitalized patients. Furthermore, IIT is associated with an increased risk for severe hypoglycaemia”.

Even one episode of hypoglycaemia has been shown to increase mortality in critically ill patients [40]. Therefore despite several trials considerable uncertainty still exists regarding the optimal glucose range and how best to achieve it. Even when a target range is selected, achieving this target range, while preventing both hypoglycaemia and hyperglycaemia, is difficult.

2.4.3 Targeting a Glucose Range

As mentioned, exogenous insulin is administered intravenously to critically ill patients to regulate blood glucose levels. This insulin dosage must be balanced with the glucose that is administered both enterally (i.e. via the gastrointestinal tract) and parenterally (i.e. intravenously) as part of nutritional feeds. Blood glucose levels are measured regularly and depending on the measured glucose levels the insulin dosages are adjusted.

Typically, in a busy ICU, there a number of ways in which glucose is measured. All require a sample of blood to be drawn from the patient which can be uncomfortable for the patient and is labour intensive for the nursing staff. The most accurate glucose measurements are obtained by sending arterial blood samples to the laboratory for testing [41]. However there can be a time delay of many hours in getting the results. A patient’s condition can change dramatically in this time period. Point of Care (POC) methods are more common. Very accurate results are obtained using Arterial Blood Gas Analysers (ABG) located in the ICU [42] when the sample is analysed
within 10 minutes of the blood draw. In some ICUs finger-stick glucose measurements are used, despite the fact that they have been shown to be inaccurate for critically ill patients [43]. Not only is the technique prone to large measurement error but poor peripheral perfusion (e.g., circulatory shock) can result in a lower glucose value in capillary than venous blood [44].

We use both lab glucose measurements and ABG glucose measurements for the analysis in this thesis. The glucose level reported by the lab techniques and the ABG machines is plasma glucose. Plasma glucose is measured in mg/dl or mmol/l.

The frequency at which plasma glucose is measured is a key factor in regulating glucose [41]. Frequent, accurate, and timely glucose measurements are required for appropriate infusion dosing [41][45]. An unstable patient may be monitored hourly, but a stable patient glucose may only be measured every 4 hours. Insulin is a very fast-acting hormone; its half-life in the blood stream is only a few minutes. Even hourly measurements could be considered infrequent when compared with the dynamics of insulin and glucose. However, the need for regular measurements must be balanced with the patients comfort and the work-load of the staff. Although continuous blood monitors are available they are expensive and not as accurate as ABG or Lab results. They are not widely used in the ICU setting [41].

2.4.4 Considerations when Modelling Glycaemia

Because of the importance of avoiding hyperglycaemia and hypoglycaemia and the difficulty in getting frequent accurate glucose measurements, a model that tracks and predicts a patient’s glucose levels in real-time is needed. There are a number of factors that must be considered in designing such a model, as described next.

Inter-Patient Variability: Substantial variability has been noted in the responses of different patients to insulin and glucose infusions [46][20]. This is due to a variety of reasons. These include the reason for which the patient was admitted to the ICU; for example, a patient with sepsis is more likely to have hyperglycaemia than a patient who was admitted following cardiac surgery [41]. Severity of illness is known to
Background

affect glucose metabolism [47]. There may also be interactions with other medications being taken by the individual; for example, steroids can sometimes reduce insulin sensitivity [41]. Pre-existing conditions such as diabetes also affect the patient’s response to insulin and glucose. For such reasons, substantial variability is seen in the responses of different patients to insulin infusions. It is therefore necessary to develop a model that can be calibrated to a wide variety of patients.

**Intra-Patient Instability:** Patients in an ICU tend to be unstable: their individual insulin sensitivity can fluctuate as their condition changes. In fact, glucose variability has been shown to be a significant independent predictor of ICU and hospital mortality [48]. Patient parameters must therefore be continually re-estimated in real-time to account for both sudden and slow changes.

**Inaccurate and Incomplete Data:** As mentioned plasma glucose measurements are subject to instrumentation error [44][43]. There may also be inaccuracies in the recording of data. Typically results are manually entered into clinical databases. Times and quantities may be approximated or misreported. Data may be missing, e.g., glucose from medications administered in a glucose solution may not be recorded.

We address these considerations when designing our own framework for modelling glycaemia in Chapter 7.

2.5 Conclusions

In this chapter, we provided some context and background for the topics discussed in subsequent chapters of this thesis.

Firstly, we introduced Bayesian Networks and Dynamic Bayesian Networks. Then we presented the standard particle filtering algorithm that we use to perform fixed-time step inference in DBNs. Finally the problem domain, modelling glycaemia in critically ill patients, to which the techniques developed for this thesis are applied, was introduced. In the next chapter, we look at research relating to this thesis.
3. Related Research

This thesis proposes using DBNs to model glycaemia in ICU patients. Before looking specifically at DBNs, a very brief overview of how Artificial Intelligence (AI) has been applied in Critical Care medicine and the challenges of the environment is presented. Then in Section 3.2, we look at existing examples of how DBNs are used in critical care medicine and how they are typically constructed. Two issues arise when using DBNs. Firstly, most DBNs are constructed manually using knowledge elicited from domain experts. This is time consuming and some refer to it as the knowledge elicitation bottleneck. We propose an alternative by using mathematical models as a source of expert knowledge. In Section 3.3 we examine other research that has used a similar approach. The second issue is the inefficiency of fixed time step inference. In Section 3.4 we look at existing approaches that have been taken to address this. Finally in Section 3.5, an examination of the approaches taken by other researchers to modelling glycaemia in the ICU is presented.

3.1 Artificial Intelligence in Critical Care Medicine

Artificial Intelligence techniques have been applied in medical settings for many years. The ICU is particularly suited to the use of AI tools due to the wealth of available data and the opportunities for increased efficiencies [49]. The purpose of this section is not to present a complete history or ontology of AI applications in the ICU setting but rather to give a flavour of the variety of applications that have been proposed. The 2001 review by Hanson et al. [49] of artificial intelligence tools in the ICU provides a good overview. They conclude that neural networks and fuzzy systems are particularly useful for waveform analysis; fuzzy controllers can be integrated into bedside devices such as fluid and medication infusion devices, mechanical ventilators, and dialysis machines; Bayesian networks and neural networks can be used in the development of smart alarms; case-based reasoning,
machine learning algorithms, and visualization tools can be used to analyse information from data warehouses describing the characteristic of an individual ICU.

Tools for predicting morbidity, mortality and length of stay have also been proposed. Barbini et al. [50] and Cevenini et al. [51] compared different models for predicting ICU morbidity following cardiac surgery. They found Bayesian classifiers and logistic regression models to be superior to an artificial neural network and the k-nearest neighbour classifier in terms of generalisation and calibration for this particular task. Ramon et al. [52] consider four different data mining algorithms for predicting 14 different tasks including probability of survival, length of stay in the ICU, probability of developing inflammation and the probability of developing kidney dysfunction. Their results for predicting probability of survival were better than the results obtained using the standard APACHE II¹ score method [53] used by most ICUs. Using the APACHE II score the area under the ROC-curve (AUC) was 75%. The best results were obtained using a Naive Bayes Classifier (AUC= 88%). The AUC for Tree-Augmented Naïve Bayesian networks was 86% and for First Order Random Forests was 82%. (First Order Random Forests are Random Forests in which the tests are first order logic queries [54].) Decision Trees provided the least promising results with AUC = 79%. It is interesting to note that no one technique proved superior for all 14 tasks.

Other research includes Cismondi et al.’s fuzzy system for predicting the outcome of lab results [55], and the INTCARE system [56][57] which combines data mining techniques and decision support systems to predict organ failure and suggest therapeutic treatment. The INTCARE research also addresses the need to distribute the application so it is available to doctors via mobile devices as well as in the ICU.

¹ APACHE II stands for Acute Physiology and Chronic Health Evaluation II. It is a severity of disease classification system developed by Knaus et al. [53]. On admission to an ICU patients are assigned a score of between 0 and 71 based on routine physiologic measurements, age and previous health status. Higher scores indicate more severe illness and an increased risk of mortality.
MIMIC II is a project undertaken by MIT, Philips Medical Systems and the Beth Israel Deaconess Medical Centre to develop and evaluate advanced ICU patient monitoring systems that will improve the efficiency, accuracy and timeliness of clinical decision making in intensive care. They aim to develop a research database from more than 30000 ICU patients. Their research to-date includes applying a Kalman Filter to estimate blood pressure and heart rate derived from the ABP waveform in the presence of high levels of persistent noise and artefact [58], using a Bayesian Network to estimate fluid requirements in the ICU [59], eliminating false alarms using classification trees and neural networks [60]. Zhang’s paper [61] describing the use of decision trees to predict hypoglycaemia in the intensive care patients is of particular relevance to this project and is discussed in Section 3.5.2 below.

A recently-published review by Bright et al. [62] shows the widespread interest across all medical fields in clinical decision-support systems. They examined 128 trials. From these they conclude that “clinical decision support had a favourable effect on prescribing treatments, facilitating preventive care services, and ordering clinical studies across diverse venues and systems”. They also stress the importance of delivering the right information to the right person in a timely manner.

3.1.1 Challenges of the ICU Environment

Much research has been carried out in applying AI techniques to the unique ICU environment, but there are challenges. Heldt et al. [63] conclude that the enormous amount of ICU data and its poor organization makes integration and interpretation time-consuming and inefficient and that “there is a critical need to integrate the disparate clinical information into a single, rational framework and to provide the clinician with hypothesis-driven displays that succinctly summarize a patient’s trajectory over time”. Ramon et al. [52] also discuss the difficulties faced in the ICU environment and list the vast amount of data, the individual nature of patient characteristics and noisy data (much data is manually entered and can be difficult to
Related Research

interpret, e.g. different people may uses different names for the same product, sensor/measurement noise is also an issue) as major challenges.

It is in this context that we address the drug delivery problem. Given the vast array of patient data at our disposal and the inter-patient variability how can computational intelligence best be used to integrate the relevant data and propose an individualized drug dose thus reducing adverse risks to the patients?

3.2 BNs and DBNs in Critical Care Medicine

BNs and DBNs have previously been used in the medical setting. An excellent introduction to the field is provided by Lucas et al. [33].

Aleks et al. [64] describe an application of DBNs to analysing ICU data. They demonstrated accurate detection and removal of artefacts in the arterial-line blood pressure sensor data. Charitos et al. [9] developed a DBN to successfully diagnose ventilator-associated pneumonia in ICU Patients. Research from the MIMIC II project mentioned in Section 3.1 includes applying a Kalman Filter to estimate blood pressure and heart rate derived from the ABP waveform in the presence of high levels of persistent noise and artefact [58], using a Bayesian Network to estimate fluid requirements in the ICU [59] and using a Bayesian network to model the cardiovascular system[8]. These BNs and DBNs have mostly only used discrete variables; the work in this thesis contains both discrete and continuous variables.

Techniques have been developed to learn both BN and DBN structure and parameters from data. More recently techniques for combining expert knowledge and automated learning for building BN structure [65][66] have been published. However most models, including the ones noted here, are manually constructed using knowledge elicited from domain experts [10] [33]. This is a time consuming task [33] [65]. Knowledge elicitation therefore remains a bottleneck; this is clear in the paper of van Gerven et al. [32], for example, which provides an excellent demonstration of the steps required to build a DBN model for prognosis of carcinoid patients. In order
to bypass this bottleneck one of the contributions of this thesis is a methodology for automatically constructing DBNs from mathematical models since mathematical models can be considered to embody existing expert knowledge.

3.3 DBNs Incorporating ODEs and SDEs

Work has previously been carried out to use mathematical models in both Bayesian Networks and DBNs. Bellazzi et al. [67] provide a good comparison of some of these methods. While some focus on predicting individual model parameters which are then used off-line [68] others discretise the state-space [69] and so do not explicitly incorporate the model equations in their original form. Voortman et al. [70] propose building causal graphs from time-series data and exploiting the ODEs to impose constraints on the model structure. In the separate, but related, topic of simulating human physiology, Abkai and Hesser [71] recognised the benefits of using both deterministic and probabilistic models. However unlike our approach which is explained in Chapter 4, they separate the ordinary differential equation solvers and the DBN models.

Evers and Lucas [11] recently proposed constructing DBNs for Linear Dynamic Systems. They too recognise that using existing models can significantly reduce the knowledge engineering effort required when building DBNs. Linear Dynamic Systems are amenable to efficient simulation, since the exact solution is available in a closed form. In contrast, this thesis focuses on non-linear systems which typically cannot be solved exactly, and so must be treated using numerical solvers as will be explained in Chapter 4.

The work most similar to ours was carried out by Anderson and Højbjerre [72]. They reworked the Minimal Model of Bergman et al. [73] (see Section 3.5.1) into a DBN model. They transform the Bergman model into SDEs and in a similar manner to this thesis, they then use the graphical model to estimate the model parameters and handle measurement uncertainty. In the SDE formulation, noise appears explicitly as a term in the equation. The solution is understood to be itself a stochastic process. To
simulate such solutions numerically, one may use methods such as the Euler-Maruyama approach (see, e.g., [74]) to generate approximate solutions for a given set of random walks representing the Wiener processes. The approach used in this thesis is more direct: the model is given in the form of deterministic ODEs and mapped directly to the DBN which then incorporates the effects of noise, and generates solutions using a numerical technique such as the (standard) Euler approach. As will be explained, the approach presented in this thesis does not require a transformation of a system of ODEs to SDEs prior to constructing the DBN. This is important because the vast majority of the mathematical models available are structured as ODEs, not SDEs.

3.4 **Handling the Inefficiency of Fixed Time Step Inference.**

An essential characteristic of Dynamic Bayesian Networks (DBNs) is that they have a fixed time step. Selecting the appropriate time step has significant implications for model construction and computation time. However, there are cases where fixed time steps are inappropriate, such as:

1. When the dynamics of the nodes in the DBN evolve at different rates from each other;
2. When events occurring between the chosen time step intervals are important;
3. When the dynamics of the model are not uniform over time, i.e. there are times of rapid change and times of slow change.

While the first two of these can be addressed by choosing the smallest possible time step, this quickly becomes computationally inefficient.

To deal with variables that evolve at different rates, Chatterjee and Russell [10] propose methods to convert DBNs with a naturally small time steps to ones with a larger time step. In the case of Alex et al. [64], a time step that matches the evidence is chosen. However, as a fixed step size is still required these approaches do not address Cases 2 and 3 above.
Performing irregular inference at the times at which evidence occurs has also been researched. Ramati and Shahar [75] propose a new modelling class that they call *Irregular-Time Bayesian Networks*. In their framework, inference is only performed at selected times e.g. when evidence occurs. A varying coefficient model with penalized splines is used to transition from one observation to the next, to find the best fit. However as the authors point out, although the need to choose a specific time granularity is removed, the number of knots must still be specified. This implies an expectation that the time between observations is relatively uniform.

Wilkin and Nicholson [76] take a very different approach and model inference as a diffusion process. They use Fokker-Plank equations to solve transition functions for DBNs with arbitrary length time intervals. This approach and other approaches using SDEs, aim to model process noise. The framework introduced in this thesis explicitly models noise as measurement uncertainty and parameter uncertainty and then reduces the uncertainty over time by individualising model parameters using real-time evidence.

For cases where the model has no natural time step, Nodelman et al. [35] propose *Continuous-time Bayesian Networks* (CTBNs) and avoid time-slicing altogether. These are particularly suited to networks with discrete variables, (although Ng et al. [77] have proposed a limited extension to incorporate continuous variables) and to cases where it is important to know about events at particular times. In this way, Cases 1 and 2 above are effectively handled. In the context of CTBNs, Saria et al. [78] propose methods to allow different clusters in the CTBN evolve at different time granularities and for the network to automatically select the appropriate granularity. This idea of automatically selecting the time step is one we too are interested in however we apply a similar concept to DBNs with continuous nodes (see Chapter 6).
3.5 Modelling Glycaemia

3.5.1 Mathematical Models

Models of the glucose-insulin system have been used since the eighties for diagnosing diabetes. Most are compartmental models (see, for example, [1]) described by differential equations. In compartmental models, tissues are grouped into one or more compartments where substances move to and from a central plasma compartment. Within each compartment the substance is considered to be uniformly distributed.

The Minimal Model by Bergman et al. [73] is the most well known and well understood of all glycaemia models. Although over thirty years old it is still widely used [79]. It was designed for diagnosing diabetes and consists of two equations to model glucose disappearance (2a and 2b) and a third equation to model insulin kinetics (2c).

\[
\frac{dG}{dt}(t) = (p_1 - X(t))G(t) - p_1(G_b) \tag{2a}
\]

\[
\frac{dX}{dt}(t) = -p_2X(t) + p_3I(t) \tag{2b}
\]

\[
\frac{dl}{dt}(t) = y(G(t) - h) - nl(t) \tag{2c}
\]

Here \( t \) is time and \( G(t) \) is the plasma glucose concentration, \( X(t) \) is proportional to insulin in the remote compartment and \( I(t) \) is the plasma insulin concentration at time \( t \).

In recent years, a number of models have been proposed specifically for the critical care patient. Many of these models [19] [80] [81] have their origins in Bergman’s Minimal Model. Van Herpe et al. [19] developed the Intensive Care Unit – Minimal Model (ICU-MM). This model will be presented in detail in Chapter 7, where it is integrated in a DBN. One of the advantages of this model is the fact that it is a minimal model, models with too much complexity can be difficult to use. However
one of the drawbacks of the original ICU-MM as presented by Van Herpe et al. is that it requires continuous glucose measurements. As mentioned earlier, continuous glucose monitors are not commonly used in ICUs.

The recently-published Intensive Control Insulin-Nutrition-Glucose model (ICING) [82] is also based on the structure of Bergman’s model. According to its proposers it encompasses the best attributes of two previous models [80] [81]. One of the motivations behind this model was the need to develop a model that requires the minimal number of parameters to be identified and individualised. The authors conclude that such models are well suited to the clinical setting which is noisy and highly variable. The resulting model makes a number of population assumptions leaving just one time-varying patient specific parameter to be individualised. In their analysis hourly glucose measurements are used to individualise the time-varying parameter. Another advantage of this model is that it can handle both enteral and parenteral nutrition, whereas the ICU-MM is designed for patients in receipt of parenteral nutrition only. Typically patients are moved to enteral feeds as soon as possible.

Hovorka et al. claim that while models based on the Minimal Model are extremely useful for estimating insulin sensitivity, they may oversimplify glucoregulation in the critically ill [20]. They proposed an alternative model which is made up of five sub-models: endogenous insulin secretion; insulin kinetics; enteral glucose absorption; insulin action and glucose kinetics [20]. This model has been shown to be effective with hourly to 4 hourly measurements, which are more realistic in a busy ICU setting. However there are 19 parameters that must be identified. Estimating individualised values for a large number of parameters is difficult with sparse evidence.

As can be seen from the above, a number of metabolic models based on ODEs exist to model the glycaemia in critically ill patients. However it is difficult to compare results to determine which model performs best. All studies use different patient cohorts and different cohorts will produce different results [83]. Le Compte et al. [84]
have produced a benchmark dataset which can be used to compare the predictive accuracy of competing models; however this author has not found any published results using the benchmark dataset.

Van Herpe et al. [85] published a comparison of the predictive performance of the ICU-MM [19] with that of Hovorka et al.’s model [20]. With their particular data set the ICU-MM outperformed Hovorka et al.’s model. For both models, the patient specific parameters were estimated by minimizing the squared error between the predicted and the observed glycaemia levels. The authors suggest that the high complexity of Hovorka et al.’s model, with 19 parameters to be estimated, and the limited data set may have led to over-fitting and thus resulted in the poor performance.

What is clear from reviewing existing mathematical models is that the people who formulated them seek a balance between capturing the complex physiological interactions and the simplifications required to ensure the models are useable in real-time in the critical care environment where data may be sparse and noisy. All models are published along with a methodology for identifying and individualising the patient parameters. Most use an offline methodology for this, either a linear least squared optimisation [19] or an integral fitting method [86] [82] or Bayesian approach [87], all of which require a learning window; for example, Van Herpe et al. use 24 hours of data where measurements are every 3-minutes [19], Chase et al. [88] require measurements every 15 minutes for the first hour. What the DBN framework proposed in this thesis offers is an online methodology that can individualise the model parameters in real-time from the very first glucose measurement. Each new glucose measurement can be incorporated and model parameters adjusted accordingly without the need to re-fit using a number of previous values.

3.5.2 Other Approaches
A 2009 review of computerized decision-support systems for tight glycaemic control by Eslami et al. [89] provides a comprehensive list of the various techniques that
have been tested the hospital setting. Comparing techniques proved difficult as different studies use different success factors. They also found that computerized systems were often introduced along with protocol changes making it difficult to assess which was responsible for any improvement.

Computer-based protocols [90][91][92][93] offer an alternative to the physiological models discussed in Section 3.5.1 above. These techniques involve taking existing rule-based hospital protocols and incorporating them into an application to display protocols or recommend changes to the intravenous insulin dosage in response to glucose measurements or recommend the glucose sampling interval. Take for example the GRIP system [91], which uses the following formula to recommend a change in the insulin infusion:

$$\Delta I = (1 + 0.25 \cdot I_{-4h}) \cdot (0.2 \cdot (G_0 - G_{target}) + 0.3 \cdot \Delta_{-4h}G)$$

In this formula, $I_{-4h}$ is the mean pump rate over the 4 hours preceding the last glucose measurement, $G_0$ is the most recent plasma glucose value, $G_{target}$ is the target glucose value, and $\Delta_{-4h}G$ is the change in glucose level between the last glucose value and the value 4 hours earlier.

This and other techniques have been shown to be effective at stabilising glycaemia within a target range. Some are passive, giving advice on demand; others actively seek input and provide feedback. While these techniques ease the burden on medical staff and are less prone to human error than paper based approaches, they do not allow for patient-specific care, as they cannot be calibrated to the individual patient.

Rather than tracking and predicting glucose levels, Zhang [61] used classification trees to predict hypoglycaemia in intensive care patients. He found that the two most recent glucose measurements and the slope of recent changes in blood glucose concentration with respect to the change in insulin infusion were the most informative features of the feature set they considered, for predicting hypoglycaemia.
While predicting hypoglycaemia is useful, a model that can be used to predict the glucose range in response to both the insulin and glucose infused is more useful to clinicians because it can also be used for inference and scenario modelling.

Another interesting line of research involves using glucose models for model-based fault detection. Lin at al. use changes in glucose response as a predictor of sepsis [94] and Meynaar et al. use blood glucose amplitude variability as a predictor of mortality [95]. Within the research project we are working on, it has been suggested that when patients are not responding as our model predicts they should, it could be used as an alarm to notify doctors that a patient’s condition has changed unexpectedly. To-date we have not tested this.

### 3.6 Conclusions

In this chapter, we presented a very brief overview of how Artificial Intelligence has been applied in Critical Care medicine and the challenges of the environment. We looked at examples of BNs and DBNs applied in medicine and how they are typically constructed. We also examined existing approaches for using mathematical models as a basis for DBNs and existing approaches for dealing with the inefficiency of fixed time step inference. Finally, we discussed the approaches taken by other researchers to modelling glycaemia in the ICU.

In the next chapter, we propose a methodology for using mathematical models as a source of expert knowledge for DBNs.
4. Mapping ODEs to DBNs

In many domains, for example, engineering and medicine, expert knowledge in the form of mathematical models is readily available. We contend that knowledge in such a form can be considered sufficient statistics of all prior experimentation in the domain, embodying generic or abstract knowledge of it. We believe that when used in a probabilistic framework, such models provide a sound foundation for data mining, inference, and decision making under uncertainty.

Chase et al. provide a strong argument for using physiological mathematical models in the critical care environment [96] and give examples where physiological models already form the basis of applications to manage sedation [97], cardiovascular diagnosis and therapy [98], mechanical ventilation [99], and the diagnosis of sepsis [100]. They argue that mathematical models offer significant physiological insight into patient status and behaviour.

Many of these models are in the form of ordinary differential equations. ODEs play a prominent role in medical settings, modelling for example, physiological systems [12][101] and drug dynamics[1].

Typically ODEs describe general population-level behaviours. In order to describe individuals, model parameters must be re-calibrated using observations of the individual. However, in most real-life situations, these observations contain uncertainty. They can be subject to measurement error or simple transcription errors. Data may be missing. Relevant quantities may not be measured or recorded. Observations may be sparse relative to the dynamics of the underlying system thus making it difficult to individualise the parameters.
Our motivation for moving from an ODE formulation to a DBN formulation is that the DBN offers an efficient framework for re-estimating model parameters dynamically over time, based on accumulated evidence.

Also, the knowledge elicitation bottleneck associated with manual DBN construction can be bypassed by basing the DBN on readily available ODE models.

In this chapter, we introduce our methodology for automatically constructing a DBN framework from a given ODE model and evaluate the framework on an abstract model for which we have the true solution.

4.1 Methodology for Constructing DBNs from ODEs

When using ODEs to model any non-trivial real-world situation, it is usually the case that the systems are so complex that the solution is not available in a closed form. Instead, numerical methods that estimate the solution at discrete points in time are employed. The simplest technique for initial value problems (IVPs) is Euler's method, (see for example, [102]).

Consider the following IVP: find \( N(t) \) such that \( N(t_0) \) is given, and

\[
\frac{dN}{dt} = f(N, t) = f(N, t; A, P_1, P_2, ..., P_m), \quad \text{for all } t > t_0,
\]

where \( N \) may be scalar-valued (for a single equation) or vector-valued (for a coupled system). Other terms in \( f \) are a time-varying coefficient \( A \), and model parameters \( P_1, ..., P_m \).

We will find approximations to \( N \) at times \( t_1, t_2, t_3, ... \). Let us denote by \( N_i \) the approximate for \( N(t_i) \). Setting \( h_i = t_{i+1} - t_i \), Euler's method is

\[
N_{i+1} = N_i + h_i f(N_i, t_i) \quad \text{for } i = 0, 1, ..., \quad (3)
\]

Thus the rate of change of \( N \) at step \( i \) is
and we can rewrite (3) as

\[ N_{i+1} = N_i + h_i \Delta N_i \text{ for } i = 0, 1, \ldots \]  

This approximation is first-order accurate in the sense that the error, \(|N(t_i) - N_i|\), is proportional to \(\max|h_i|\).

When we map the ODEs to a DBN, we set the DBN's discrete time steps to be of duration \(h_i\) also.

We encapsulate an Euler approximation by mapping Equations (4) and (5) directly to two deterministic nodes, \(\Delta N\) and \(N\), in the DBN, as illustrated in Figure 5.

As the figure shows, Node \(\Delta N\) is a deterministic node, and its parent nodes are set to be all the terms needed to solve \(f(N_i, t_i)\), in the same time slice of the DBN.

Node \(N\) is also a deterministic node. In each time slice, it evaluates the current value of \(N_{i+1}\) using Equation (5), hence its parents are set to be itself and node \(\Delta N\) from the previous time slice. (Note that inter-time slice arcs are shown in dashed arcs in Figure 5.)

In the DBN, the model parameters, \(P_1, \ldots, P_m\), are represented as continuous nodes.
This procedure may be applied to a system of ODEs, by creating a sub-net for each equation and adding dependencies between them, as dictated by terms appearing in the equations.

4.2 Expanding the DBN to Represent Measurement Uncertainty

The DBN provides a natural framework to reason with noisy data. The observed value for the variable to be approximated is assumed to contain a certain amount of measurement error.

![Diagram](image)

Figure 6: Extra evidence nodes (black) are added to model the relationship between observed and true values.

As can be seen in Figure 6, each observed measurement (for example Observed N) is modelled as a continuous distribution whose mean is its parent node (N), the true variable value and whose standard deviation represents the measurement uncertainty. Similarly, each actual input (A) to a system differs from the intended input (Intended A), which is observed, and so a clear distinction is created in the DBN. The true value (A) is represented as a conditional distribution whose mean is the intended value.

4.3 Expanding the DBN to Re-estimate Parameters

Model parameters $P_1, ..., P_m$ are allowed to vary over time. In Figure 7, they are represented as continuous nodes. Distributions on the initial state model can be
viewed as the distribution of the population values. These population values can be learned from the data or obtained from the published literature. All model parameters are allowed to vary in each time step by including a conditional dependency on its value in the previous time, as shown in Figure 7; they can therefore converge to values appropriate to the individual case over time, based on evidence from the temporal data stream.

![Diagram](image)

Figure 7: Extra inter-slice arcs on nodes $P_1 \ldots P_m$ to allow parameters to be tuned to the evidence over time.

4.4 Selection of DBN Parameters

As noted in Section 4.1 above, model parameters are represented as continuous nodes and the initial state model distribution can be viewed as the distribution of the population values. These population values are often published along with the ODE models and can therefore be taken from the literature. In the case where there are no published values, the parameters can be learned from the data. Using data from a large number of subjects, both the sensor model and the transition model parameters can be learned using an Expectation Maximization (EM) algorithm [31].

4.5 Evaluation on an Abstract Model

We validate this methodology in a setting that is independent of model and data errors, by choosing a system of ODEs for which the exact solution is available:
\[
\frac{dG}{dt}(t) = G(t)X(t) - P_1, \quad \frac{dX}{dt}(t) = P_2X(t) + A(t)G(t),
\]

subject to the initial conditions \(G(0) = 1, X(0) = 0\).

If we take \(P_1 = P_2 = 1\), and the time varying term as \(A(t) = e^t\), then the solution is \(G(t) = e^{-t}, X(t) = e^t - 1\).

### 4.5.1 Basic Euler Simulation

#### 4.5.1.1 Numeric Error

Suppose that we wish to simulate the solution from time \(t = 0\) to \(t = 1\), and the true solution of \(G\) is observed at infrequent time points in that interval. We compute the solution using Euler’s method with 120 steps while updating the value of \(G\) to the correct value, at the given observed steps i.e. 10, 20, 40, 60, 90 and 100. We then calculate the root mean squared error (i.e., the Euclidean norm) between the exact solution and the Euler approximation as follows:

\[
RMSE_G = \sqrt{\frac{\sum_{k=0}^{N} (G(t_k) - G_k)^2}{N}},
\]

where \(N\) is the number of time steps, in this case 120. Using 120 steps, \(RMSE_G = 5.24 \times 10^{-4}\). Using the same approach for the error calculation, \(RMSE_X = 5.73 \times 10^{-3}\). These values are the error introduced by approximating the solution using Euler’s method, i.e. the pure numeric error.

#### 4.5.1.2 Numeric and Data Error

Now computing the solution using Euler’s method, but with the incorrect values of \(P_1 = 1.2\) and \(P_2 = 0.8\), and updating the value of \(G\) to the correct value at the given observed time points, the difference between the true and estimated solution, is found to be \(RMSE_G = 0.0253\) and \(RMSE_X = 0.101\). These errors are comprised of both
data and numeric error. Clearly they are much larger than the pure numeric errors noted above and so we can say the data error dominates. For clarity these numbers are shown in Table 1.

<table>
<thead>
<tr>
<th>Error Type</th>
<th>RMSE&lt;sub&gt;G&lt;/sub&gt;</th>
<th>RMSE&lt;sub&gt;X&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Numeric Error</strong>: Error using Euler’s method with correct parameters</td>
<td>5.24 x 10^-4</td>
<td>5.73 x 10^-3</td>
</tr>
<tr>
<td><strong>Numeric and Data Error</strong>: Error using Euler’s method with incorrect parameters</td>
<td>0.0253</td>
<td>0.101</td>
</tr>
</tbody>
</table>

Table 1: Comparison of errors using Euler’s method, with and without parameter error. The data error dominates in the second case.

4.5.2 Using Our DBN Framework for the Abstract Model

Using the methodology of Section 4.1, a DBN structure is derived from Equation (6), and is expanded as described in Sections 4.2 and 4.3 to produce the structure shown in Figure 8.

![DBN for abstract example](image)

Figure 8: DBN for abstract example (6).

Each of the model variables, G and X, are represented as deterministic nodes. Each has a corresponding delta node, ΔG and ΔX, to capture the changes in these quantities in each time slice. The time step length \( h = \frac{1}{120} \). To allow the model parameters \( P_1 \) and \( P_2 \) vary over time, an inter-slice dependency is introduced for each
and shown as a dashed arc in Figure 8. The conditional probability tables for the continuous nodes are shown in Table 7.

<table>
<thead>
<tr>
<th>Node</th>
<th>Initial State Model</th>
<th>Transition Model</th>
<th>Sensor Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>$N(1.2, 0.3162)$</td>
<td>$N(P_{1,t-1}, 0.03162)$</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>$N(0.8, 0.3162)$</td>
<td>$N(P_{2,t-1}, 0.008162)$</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td>$N(\text{Intended } A, 0.1)$</td>
</tr>
<tr>
<td>Observed G</td>
<td></td>
<td></td>
<td>$N(G, 0.0001)$</td>
</tr>
</tbody>
</table>

Table 2: Condition probability tables for continuous nodes of DBN shown in Figure 8

We repeat the numerical experiment using this DBN, again with inaccurate starting values of $P_1 = 1.2$ and $P_2 = 0.8$, and with correct observations for $G$ at time steps 10, 20, 40, 60, 90 and 100. As shown in Table 3, this yields RMSEs for $G$ and $X$ of $2.7 \times 10^{-3}$ and $5.86 \times 10^{-2}$ respectively, much smaller errors than those obtained using Euler's method above.

<table>
<thead>
<tr>
<th></th>
<th>$RMSE_G$</th>
<th>$RMSE_X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Solution</td>
<td>0.0253</td>
<td>0.101</td>
</tr>
<tr>
<td>DBN Solution</td>
<td>0.0027</td>
<td>0.0586</td>
</tr>
</tbody>
</table>

Table 3: Comparison of error obtained from standard Euler approximation and the DBN framework when compared to the true solution. Both have incorrect starting parameters.

The results show that the DBN-based solution produces a closer approximation of the true solution. This is because in the standard ODE solution, updated values of $G$ (the evidence) are used only to correct $G$ at that point in time; no changes are made to $P_1$ or $P_2$. Conversely, the DBN solution not only corrects $G$, it also seeks to infer from this new evidence what $P_1$ and $P_2$ should be. This can be seen in Figure 9 where the inferred value for $P_1$ adjusts from the initial incorrect setting of 1.2 to the correct value of 1.0 when the first observation at time step 10 is incorporated. At this time its uncertainty collapses (represented as one standard deviation around the inferred mean in the Figure 9). As the time in between observations increases, so too
does the range of possible values for $P_1$. This is why the standard deviations are seen to grow in between observations.

Figure 9: Values inferred by DBN for $P_1$. $P_1$ is initially set to the incorrect starting value of 1.2. Once evidence is incorporated, the inferred value adjusts to the correct value of 1.0 and the uncertainty collapses.

$P_2$ on the other hand does not directly influence $G$ in the DBN. For this reason it requires a number of observations before its mean tends to the correct value. This can be seen in Figure 10.

Figure 10: Values inferred by DBN for $P_2$. $P_2$ is initially set to the incorrect starting value of 0.8.

Figure 11 shows the values inferred by the DBN for both $G$ and $X$. Although difficult to see in the main figure, when observations of the true solution are incorporated the DBN aligns with them. This is easier to see in the enlarged view in Figure 12. Because observations of $G$ are incorporated there is little uncertainty in the inferred
values, hence the standard deviations are small. They collapse whenever a new observation is incorporated and grow in between observations. This can be seen in Figure 12. Conversely there are a range of combinations for $X$ and $P_2$ that can explain the observed values for $G$, hence the standard deviations for these values do not collapse. In fact, in the case of $X$ its standard deviation grows over time.

Figure 11: Values inferred by the DBN for $G$ and $X$.

Figure 12: Enlarged view of the DBN values inferred for $G$ from time step 5 to 40. When observations of the true solution are incorporated the DBN aligns with them. The uncertainty, shown as one standard deviation, grows in-between observations.
4.5.3 **How Many Particles Should be Used?**

An important consideration when running inference with a particle filtering algorithm is determining the appropriate number of particles to use. If using too few particles, the complete search space will not be explored and the DBN may not converge to the true solution. However using a very large number of particles will significantly increase the time and space requirements.

We carried out sensitivity analysis in order to assess the appropriate number of particles to use for the DBN incorporating the abstract model described in Section 4.5.2. Sensitivity analysis involved running the simulation 5 times for each number of particles. For each variable the RMSE is calculated from the inter-run difference. In Figure 13, the average inter-run RMSE for each variable is expressed as a percentage of the variable's maximum value. As the number of particles is increased, the resulting predictions become more consistent. As can be seen, using 10000 particles or fewer, there is considerable variation in the results from the 5 runs. At 50000 particles the runs are reasonably consistent and increasing the number of particles beyond 50000 does not substantially reduce the inter-run variation. For the results presented in this chapter, 50000 particles were therefore used.
4.5.4 What Time Step Length Should be Used?

In Chapter 6, an algorithm that automatically selects the appropriate number of variable length time steps to use will be presented. However, before moving to an adaptive approach it is informative to examine the behaviour when time steps of different fixed lengths are employed.

Using the basic Euler DBN of Figure 8, 30 runs were completed using an increasing number of fixed time steps for the period $0 \leq t \leq 1$. For each step size the standard deviation on the Gaussian transition probabilities is scaled relative to the original time step, which in this case is $\frac{1}{120}$.

$$\sigma = \frac{\sigma_{original}}{\sqrt{\frac{h_{original}}{h_{new}}}}$$
The mean error and corresponding standard deviations are shown in Table 4, and in Figure 14 these mean RMSE values are shown in a log-log plot. As can be seen, increasing the number of time steps, and hence decreasing the time step length $h$, does not yield a more accurate solution. In fact, the results show an increase in the RMSE as $h$ is decreased.

<table>
<thead>
<tr>
<th>Number of Fixed Time Steps</th>
<th>$G$ Mean RMSE</th>
<th>$G$ Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>0.0021</td>
<td>0.000161</td>
</tr>
<tr>
<td>60</td>
<td>0.00271</td>
<td>0.000172</td>
</tr>
<tr>
<td>120</td>
<td>0.00285</td>
<td>0.000113</td>
</tr>
<tr>
<td>480</td>
<td>0.00306</td>
<td>0.000310</td>
</tr>
<tr>
<td>1920</td>
<td>0.00387</td>
<td>0.000905</td>
</tr>
</tbody>
</table>

Table 4: The mean RMSE$_G$ over 30 runs and the corresponding standard deviation.

![Log-log plot](image)

Figure 14: The log-log plot comparing the mean RMSE$_G$ values for different time step sizes. As can be seen a larger number of time steps (corresponding to smaller step lengths) results in larger errors.

We compare each pair of time steps. Taking the null hypothesis that each number of time steps yield an equal RMSE, and a significance level of 0.01, t-tests indicate that
the null hypothesis can be rejected. The p-values for comparisons between the different numbers of time steps are $\leq 0.001$. The differences in $G$’s RMSE error are statistically significant.

But why does the error increase as we increase the number of time steps? As we have seen, in this model an Euler approximation is used. This is first-order accurate in the sense that the numerical error is proportional to $h_i = t_{i+1} - t_i$. One would therefore expect the model to produce more accurate results with smaller time-step lengths, as this reduces the numerical error. However, the DBN model encompasses both numerical and data error. In this example and indeed most practical examples, the data error dominates.

By increasing the number of time steps we increase the numerical accuracy of the solution. However, because the data error dominates, no overall improvement is observed in the error. This can be seen more clearly in Figure 15 where we add the results from the basic Euler approximations to the log-log plot. The corresponding RMSEs are shown in Table 5.
Mapping ODEs to DBNs

Figure 15: The log-log plot comparing the mean RMSE values for different time step sizes.

<table>
<thead>
<tr>
<th>Number of Time Steps</th>
<th>RMSE&lt;sub&gt;G&lt;/sub&gt;</th>
<th>Euler with Correct Parameters</th>
<th>Euler with Incorrect Parameters</th>
<th>DBN with Incorrect Starting Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>0.000233</td>
<td>0.02455</td>
<td>0.002099</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>0.00102</td>
<td>0.02512</td>
<td>0.002707</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>0.000524</td>
<td>0.02533</td>
<td>0.002849</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>0.000134</td>
<td>0.02549</td>
<td>0.003058</td>
<td></td>
</tr>
<tr>
<td>1920</td>
<td>0.0000337</td>
<td>0.02553</td>
<td>0.003874</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Comparison of the RMSEs using different time steps. With the correct parameters the error using the Euler method decreases as the number of steps increases. With the incorrect parameters the error using the Euler method does not change substantially as the number of steps increases. Using the DBN framework with the incorrect starting parameters the error does not decrease as we increase the number of time steps.

Firstly, observe the Euler error with the correct starting parameters in Figure 15, this is pure numeric error. As expected when the number of time steps is increased, resulting in smaller step lengths, the numeric error decreases. However, when data
error is introduced in the model parameters, we can observe the error using the Euler approximation no longer decreases as we increase the number of time steps. As mentioned earlier this is because the data error is dominating. This data error does not change as we vary the number of time steps.

In a similar manner, the error from the DBN with the incorrect starting parameters, does not improve as we increase the number of time steps. It produces a lower error than the Euler approximation with the incorrect starting parameters because it can infer corrected values for the model parameters. However, increasing the number of time steps does not increase the frequency of the evidence from which we infer the correct model parameters. In each case, only 6 observations of the true value for \( G \) are incorporated. Therefore, the corrected parameter values inferred by the DBN do not differ substantially as we vary the number of time steps. It is for this reason we would conclude that the error does not decrease as we increase the number of time steps.

However, not only does it not decrease, in fact it gets slightly worse, a behaviour that is not observed in the standard Euler approximation. It is unclear why this is so. It is the opinion of this author that as the number of steps between observations increases the model drifts further from the true solution.

4.6 Conclusions

In this chapter, a methodology for incorporating systems of ordinary differential equations in a Dynamic Bayesian Network was presented. The proposed methodology handles both data and model uncertainty in a principled manner. Real-time observations from the true solution are used to individualise model parameters and align the solution with reality.

In the case where data error dominates over numeric error, the most accurate results are obtained using a minimal number of time steps. The time step length should be
chosen small enough ensure stability of the underlying ODE model, but as large as possible to ensure a minimal number of steps in between observations.

In the next chapter, we address the question of ‘stiff’ problems where when using the basic Euler method the time step length must be very small to ensure the numeric error does not dominate. An alternative to using very small time steps is to incorporate a second-order or higher-order solver.
5. DBNs Incorporating an RK2 Solver

The Euler approximation used in the previous chapter is first-order accurate in the sense that the numerical error is proportional to \( h_t = t_{i+1} - t_i \). Therefore, decreasing \( h_t \) reduces the numerical error, but of course leads to increased run times. A standard technique for obtaining a more accurate solution while using the same step size is to use a higher order method. Here, we look at how a higher order solver, a Runge-Kutta solver (see, for example [102]) can be incorporated in our DBN framework.

5.1 Methodology for Constructing RK2 DBN

Second-order Runge-Kutta (RK2) methods are of the form:

\[
\begin{align*}
N_{i+1} &= N_i + h_t(ak_1 + bk_2) \text{ for } i = 0, 1, \\
k_1 &= f(t_i, N_i), \\
k_2 &= f(t_i + \alpha h_t, N_i + \beta h_t k_1).
\end{align*}
\]  

(7)

Here \( h_t k_1 \) approximates the change in \( N \) at \( t = t_i \) (as in Euler's method), and \( h_t k_2 \) represents the change at a specific point in \([t_i, t_{i+1}]\) determined by \( \alpha \) and \( \beta \). One takes \( a = 1 - b \), so as to obtain a weighted average between \( k_1 \) and \( k_2 \).

If \( a = 1 \), Equation (7) reduces to the Euler method (since \( b = 1 \)) but any other value of \( a \) gives a method that is second order, in the sense that the error is proportional to \( h_t^2 \). Since \( h_t \ll 1 \), this should be much more accurate than Euler’s method where the error is proportional to \( h_t \).

Examples of RK2 methods include the Modified (Midpoint) Euler method, which has \( \{a, b, \alpha, \beta\} = \{0, 1, \frac{1}{2}, \frac{1}{2}\} \) and the Improved Euler method which has \( \{a, b, \alpha, \beta\} = \{\frac{1}{2}, \frac{1}{2}, 1, 1\} \).
In our implementation, we leave the parameters \(\{a, b, \alpha, \beta\}\) to be set at run-time, so that the DBN in Figure 16 can employ any method formulated in the general RK2 framework. In a similar manner to the Euler DBN construction, the equations in (7) are each mapped directly to deterministic nodes in the DBN.

In this thesis two structures are used. The first, illustrated in Figure 16, uses the values of model parameters from the current slice when evaluating \(k_1\) and \(k_2\).

![First DBN structure incorporating an RK2 solver. Solid arcs connect nodes within a time slice; dashed arcs connect nodes between time slices.](image)

Some models include time-varying coefficients, for example \(A(t)\) in Equation (6), introduced in Chapter 4 but repeated here for convenience.

\[
\frac{dG}{dt}(t) = G(t)X(t) - P_1, \quad \frac{dX}{dt}(t) = P_2X(t) + A(t)G(t),
\]  

(6)

where \(A(t) = e^t\).

In these cases, *Improved Euler* is the most appropriate RK2 formulation, as for other formulations the DBN would have to approximate evidence for the time-varying inputs at some point \(t\) where \(t_i < t < t_{i+1}\). In this second RK2 structure, when evaluating \(k_1\), model parameters and time-varying coefficients from the previous time slice, i.e. \(t_i\), are used; however when evaluating \(k_2\) model parameters and time-varying coefficients from the current time slice, i.e. \(t_{i+1}\), are used. This results in the structure shown in Figure 17.
5.2 Evaluation

5.2.1 The Non-Stiff Abstract Model

The abstract model, Equation (6) is incorporated here in a DBN with an RK2 solver.
Figure 18: DBN incorporating an Improved Euler solver for the abstract model (6).

Setting the run time parameters to \( \{a, b, \alpha, \beta\} = \{\frac{1}{2}, \frac{1}{2}, 1, 1\} \), an Improved Euler solver is implemented. For each method and each number of time steps, 60 runs were completed. The average RMSE relative to the true solution are shown in Table 6 and as a log-log plot in Figure 19.
DBNs Incorporating an RK2 Solver

Figure 19: Comparison of mean RMSEs for the Euler DBN and the RK2 DBN using different numbers of time steps for the same interval.

<table>
<thead>
<tr>
<th>Number of Time Steps</th>
<th>RMSE&lt;sub&gt;G&lt;/sub&gt;</th>
<th>Euler DBN with Incorrect Starting Parameters</th>
<th>RK2 DBN with Incorrect Starting Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>0.0021</td>
<td>0.0019</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>0.0027</td>
<td>0.0025</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>0.0028</td>
<td>0.0028</td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>0.0031</td>
<td>0.0032</td>
<td></td>
</tr>
<tr>
<td>1920</td>
<td>0.0031</td>
<td>0.0037</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Comparison of mean RMSE values for the Euler DBN and the RK2 DBN using different numbers of time steps for the same interval.

We can observe that there is little difference between the Euler and RK2 approximations. This is not really surprising, as noted in Section 4.5.4 data error dominates numeric error in this model. Therefore using a higher order solver to improve the numeric accuracy has little or no effect.
Because the Euler DBN contains fewer nodes than the RK2 DBN it is simpler to implement and understand, it is faster and uses less memory.

To investigate if benefits are to be gained with stiff problems a second abstract model is introduced.

5.2.2 A Stiff Abstract Model

Take

\[
\frac{dG}{dt}(t) = P_1A(t)G(t)X(t) + P_2B(t) \tag{8a}
\]

\[
\frac{dX}{dt}(t) = P_3X(t) + P_4C(t)G(t) + P_5D(t) \tag{8b}
\]

subject to the initial conditions \(G(0) = 0, X(0) = 0\). If we take \(P_1 = -2/5, P_2 = 2, P_3 = -2k, P_4 = -5, P_5 = 5\), and the time varying terms as

\[
A(t) = ke^{2kt},
\]

\[
B(t) = k \tanh(kt),
\]

\[
C(t) = ke^{-2kt},
\]

\[
D(t) = ke^{-2kt},
\]

then the solution is \(G(t) = \tanh(kt)^2, X(t) = 5 \tanh(kt)e^{-2kt}\).

This is once again a non-linear system. By choosing different values for \(k\) the problem can be made “stiffer”. We choose \(k = 7\) for our analyses. Figure 20 shows the DBN.
Figure 20: RK2 DBN for second abstract model (8). Dotted lines indicate inter-slice dependencies; solid lines indicate intra-slice dependencies.

$P_1$ and $P_5$ are initialised with the inaccurate starting values of $P_1 = -0.5$ and $P_5 = 6$ (the correct values are $P_1 = -0.4$ and $P_5 = 5$) and with true observations for $G$ at various time points. The conditional probability tables for the continuous nodes are shown in Table 7.

<table>
<thead>
<tr>
<th>Node</th>
<th>Initial State Model</th>
<th>Transition Model</th>
<th>Sensor Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>$N(-0.5, 0.06)$</td>
<td>$N(P_{1,t-1}, 0.000004)$</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>$N(2, 0.002)$</td>
<td>$N(P_{2,t-1}, 0.00005)$</td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>$N(-14, 0.14)$</td>
<td>$N(P_{3,t-1}, 0.00001)$</td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>$N(-5, 0.005)$</td>
<td>$N(P_{4,t-1}, 0.00005)$</td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>$N(6, 0.6)$</td>
<td>$N(P_{5,t-1}, 0.00005)$</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td>$N(\text{Intended } A, 1 \times 10^{-5})$</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>$N(\text{Intended } B, 1 \times 10^{-5})$</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>$N(\text{Intended } C, 1 \times 10^{-5})$</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>$N(\text{Intended } D, 1 \times 10^{-5})$</td>
<td></td>
</tr>
<tr>
<td>Observed G</td>
<td></td>
<td>$N(G, 0.1)$</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Conditional probability tables for continuous nodes of DBN shown in Figure 20.

The error from the Euler DBN, with the correct starting values for $P_1$ and $P_5$ and using 48 time steps, is 19.60%. The error from the Euler DBN, with the incorrect starting
values for $P_1$ and $P_5$, is 20.11%. The difference between the Euler DBN results with the correct and incorrect parameters is only 0.51%. We therefore say the numeric error dominates the data error when using 48 time steps. In the following section, we can observe that using the RK2 DBN reduces this numeric error.

5.2.2.1 Comparison of Euler DBN and RK2 DBN both with 48 time steps

Firstly it is instructive to look at the graphs for both the Euler DBN and the RK2 DBN. The RK2 DBN is formulated as Improved Euler i.e., $\{a, b, \alpha, \beta\} = \{\frac{1}{2}, \frac{1}{2}, 1, 1\}$. Figure 21 shows the inferred values for $G$ and $X$ obtained using 48 time steps and the Euler DBN. It can be seen that in between the observations the inferred values for $G$ tend to stray. This is due to both numerical and data error. However, as mentioned above, in this case the numeric error dominates.

Figure 21: Inferred $G$ and inferred $X$ from the Euler DBN using 48 time steps.

When we switch to the RK2 DBN the large errors, observed before the evidence is incorporated, are reduced. This can be observed in Figure 22 which shows the output from the RK2 DBN. The RK2 output is an obvious improvement when compared to the Euler DBN output in Figure 21 above.
The purpose of this investigation is to determine if efficiency improvements can be gained by using the higher order solver on this model. To do this, we compare the results from both DBNs using various time steps in the next section.

5.2.2.2 Comparison of Euler and RK2 DBN Results with Various Time Steps

The log-log plot of the RMSE versus the number of time steps is shown in Figure 23. It is based on the data in Table 8. The RMSE is calculated based on the difference between the values inferred by the DBN and the true solution. The results shown are the mean RMSE of 30 runs.

Figure 23 and Table 8 show that the lowest error is produced using an RK2 DBN with 48 time steps. To get an equivalent error using an Euler DBN at least 240 steps would be needed. Hence, it can be concluded that in this case, by using an RK2 DBN a significantly lower number of steps are required and so the RK2 DBN results in efficiency improvements.
Incorporating an RK2 or higher order solver can increase the numerical accuracy, that is, it can reduce the error that is introduced into the simulation because we are using an approximate solution to the differential equation rather than the true solution.

In our stiff example, Equation (8), significant efficiency benefits are gained using the higher-order RK2 solver because larger step sizes can be used. However it must be noted that in this example the model parameters are well-constrained and small.
changes in the model parameters do not significantly affect the problem stiffness. In the next chapter this is not the case, instead small changes significantly affect the problem stiffness. As will be shown, this leads to a situation whereby, when model parameters are allowed to vary, a smaller number of time steps cannot be used by the DBN incorporating a higher order solver. This is discussed further in Section 6.3.3.

In Section 5.2.1 we found that for systems that are not stiff, there are no benefits to be gained by using higher-order schemes. In Section 5.2.2 we showed that in certain stiff problems using a higher-order scheme can reduce the number of steps required. An approach that would lead to efficiency improvements in both stiff and non-stiff problems would be preferable. In Chapter 6, an adaptive time stepping strategy to do just that is outlined.
6. Adaptive-Time Particle Filtering

For the examples in the previous chapters, inference was performed using a standard fixed-time step particle filtering algorithm [29]. Careful consideration was given to step-size selection. The step size must be chosen to be sufficiently small so that numerical error is not significant. However, reducing step sizes results in increased computation, so a balance must be struck between a practical run-time and numerical accuracy. As has been noted earlier in this thesis, for non-stiff systems this was not a concern; however, for stiff problems very small step sizes may be required and inference quickly becomes inefficient. An alternative approach, that involves adapting the step size according to the dynamics at each step, is therefore presented here.

In this chapter, we propose a new algorithm for particle filtering that allows for non-fixed step sizes. We call this algorithm Adaptive-Time Particle Filtering. It is comprised of two parts. In Section 6.1 an approach where each particle is allowed adapt the step size independently according to its own dynamics is presented. Once we have a scheme that allows for variable time steps, we implement a mechanism for automatically choosing those time steps. This is described in Section 6.2. These techniques are evaluated in Section 6.3 by applying them to a model based on the classic van der Pol oscillator.

Finally in Section 6.4, we present an alternative for improving the efficiency of inference that we termed full steps and sub-steps.

6.1 Adaptive-Time Particle Filtering

In the standard fixed-time step DBN, when filtering and prediction are carried out, the results are reported at each step. With our new Adaptive-Time Particle Filtering, the user can specify the intervals at which the results should be reported. In certain situations, the adaptive algorithm may choose very small step sizes to capture rapidly
changing dynamics, for example 0.01 minutes, but the user may only be interested in the predicted values at a larger time interval, for example, every 10 minutes. The user can specify this summary interval or a set of summary times.

Each particle is propagated independently to the next summary time, allowing each particle to adapt the step size for the dynamics that result from its particular set of values.

In fixed-time step particle filtering, if the granularity of the evidence time stamp does not match the fixed-time step, it must be approximated to the nearest step. One advantage of our adaptive approach is that exact time stamps can be used.

For our algorithm, evidence is divided into two types; continuous and instantaneous. Continuous evidence is defined as evidence that remains constant until a new value is reported. On the other hand, instantaneous evidence is defined as evidence with a value at a particular moment in time. Where new instantaneous evidence or changes in continuous evidence occur, a summary step is invoked.

At each summary step, the particles are weighted based on the evidence, and re-sampling ensures that the samples with the higher weights are more likely to be propagated to the next step.

A formal description of the inference algorithm is shown in Algorithm 1. In the algorithm, \( S \) denotes the set of particles. The times at which summaries are reported.
are denoted \( \{R_0, \ldots, R_K\} \). Between summaries, the time steps are \( \{t_0, t_1, \ldots\} \); note that these may be different for different particles.
Algorithm 1: Adaptive-Time Particle Filtering returns $Q$, a vector of summarised particles for each summary step

**Inputs:**

$F$: finish time; $N$: number of particles; $\{R_0, ..., R_K\}$ summary times

**Local Variables:**

$S$: a vector of particles of size $N$; $W$: a vector of weights of size $N$;

$Q$: a vector of summarised particles;

$h$: proposed step size; $\bar{h} = R_1 - R_0$

$t_0 = 0; \ i = 0; \ k = 0$

**while** $R_{k+1} < F$ **do**

**for** $p = 1 \rightarrow |S|$ **do**

$i \leftarrow 0; \ t_i = R_k$

**while** $t_i \leq R_{k+1}$ **do**

**repeat**

$S_p \leftarrow$ sample from $P(x_t | x_{t-1})$ \ {Scaled according to the step size}

$(tolOK, \bar{h}) \leftarrow$ Check Tolerance($\bar{h}$) \ {See Algorithm 2}

**if** $tolOK$ **then**

$t_{i+1} = t_i + \bar{h}$

**else**

$t_i = t_{i-1} + \bar{h}$ \ {Try again with a smaller step}

**end if**

**until** $tolOK$

$i \leftarrow i + 1$

**end while**

**end for**

$W \leftarrow P(y_{t+1} | x_{t+1})$ \ {Weight particles based on evidence}

$Q \leftarrow$ summarise $S$ based on $W$ \ {Store weighted mean and standard deviation of all particles}

$S \leftarrow$ re-sample $S$ based on $W$ \ {Select most likely particles for next iteration}

$k \leftarrow k + 1$

**end while**

**return** $Q$
6.2 Choosing the Step Size

The step size control mechanism used is shown in Algorithm 2. It is based on a procedure outlined by Butcher [86, Section 202] and demonstrated by Nhan [103], that aims to control the numeric error introduced at each time step. To do this, we must estimate the local error. The error is estimated using the delta nodes described in Section 4.1. This estimated error is compared to a prescribed tolerance. If the tolerance is met, the current step is accepted and a new step size is proposed for the next step, which may be bigger. If the tolerance is exceeded, the current step is rejected and a reduced step size proposed.

In order to control the numeric error introduced at each time step, we must estimate the local error. Butcher [86] shows that a good estimate of the local error is

\[
\frac{h_i^2}{2} \varepsilon, \quad \varepsilon = \frac{|f(N_{i+1}, t_{i+1}) - f(N_i, t_i)|}{h_i}
\]

In the notation of Section 4.1

\[
\varepsilon = \frac{|\Delta N_{i+1} - \Delta N_i|}{h_i}.
\]

Note, however, that for a coupled system, \(N\) is a vector valued quantity. Since we wish to control the largest error in any component in the system, we set

\[
\varepsilon = \max \frac{|\Delta N_{i+1} - \Delta N_i|}{h_i}.
\]

We aim to keep the estimated error below a prescribed tolerance, \(\tau\). If the tolerance is met, the current step is accepted and a new step size is proposed for the next step, which may be bigger or smaller.
Adaptive-Time Particle Filtering

The term $\sqrt{2 \tau/\epsilon}$ is an estimate of the maximum step size that would ensure the tolerance is met: any step size less than or equal to this is likely to yield a numerical approximation with the desired accuracy. So that the choice is unlikely to overestimate the step size and hence trigger an unnecessary extra step, we set the new step size to be $q \sqrt{2 \tau/\epsilon}$ where $q \in (0,1)$ is a user-chosen parameter. Values in the range $0.6 - 0.9$ work well. In our examples in this thesis we set $q = 0.9$.

When the solution switches from a region in which its derivative changes rapidly to a region in which its derivative changes slowly, the algorithm will increase the step size in order to maintain efficiency. However, as noted by Butcher [86], if the step size changes rapidly then the error will be adversely affected. Therefore, we control the rate at which the step size increases with the parameter $M_1 > 1$, the maximum factor by which the current step size will be increased. The proposed step size is the smaller of the estimated step size based on achieving the desired error i.e., $q \sqrt{2 \tau/\epsilon}$, and $h_i M_1$. In this thesis we set $M_1 = 1.5$.

If the tolerance is exceeded, the current step is rejected and a reduced step size proposed. Once again the proposed step size is estimated as $q \sqrt{2 \tau/\epsilon}$. To ensure a reduced step size is always recommended we introduce $M_2$ and set $M_2 < 1$. The algorithm recommends the smaller of the estimated step size $q \sqrt{2 \tau/\epsilon}$, and a factor of the current step size $h_i M_2$. In our examples in this thesis we set $M_2 = 0.9$.

In Appendix 1, we present the results of sensitivity tests on $q$, $M_1$ and $M_2$ based on the van der Pol DBN which is introduced in the next section. In summary, they show that the algorithm is reliable for any set of values satisfying $0.6 \leq q \leq 0.9$, $M_1 > 1$, and $M_2 < 1$. 

70
Algorithm 2: Check Tolerance($h_i$) returns Boolean to indicate if tolerance check passes and $\bar{h}$

**Inputs:**
$h_i$: previous step size
$\tau$: Prescribed tolerance
$\Delta N_{i+1}, \Delta N_i$: Vectors of delta nodes.

**Local Variables:**
$M_1$: Maximum factor by which step size is increased
$M_2$: Factor by which step size will decrease if tolerance is not met
$q$: Safety factor so the step size is just less than the maximum recommended.

$$\varepsilon \leftarrow \max |\Delta N_{i+1} - \Delta N_i| / h_i$$

if $(h_i^2 \varepsilon / 2) < \tau$ then

$$\bar{h} \leftarrow \min \left( q \sqrt{2 \tau / \varepsilon}, h_i M_1 \right)$$

$tolOK = true$

else

$$\bar{h} \leftarrow \min \left( q \sqrt{2 \tau / \varepsilon}, h_i M_2 \right)$$

$tolOK = false$

end if

return ($tolOK$, $\bar{h}$)
6.3 Evaluation

In this section, we evaluate our techniques by applying them to a model based on the classic van der Pol oscillator for which we know the true solution.

Using the techniques outlined in this thesis we have three options for performing inference:

**Option 1:** Perform fixed time step inference on the DBN incorporating a first-order Euler solver.

**Option 2:** Perform fixed time step inference on the DBN incorporating an RK2 solver.

**Option 3:** Perform adaptive-time step inference on the DBN incorporating a first-order Euler solver.

In the following subsections, we firstly introduce the DBN encapsulating the van der Pol model, and then we apply each of the options enumerated above to three different scenarios:

**Scenario 1:** The model parameters are known and only numerical error exists.

**Scenario 2:** The model parameters are unknown and evidence from the true solution is used to find the correct parameters.

**Scenario 3:** The model parameters are unknown and the initial conditions are incorrect.

In all cases, we calculate the root mean square error (RMSE) and the mean absolute error (MAE) by comparing the predicted value with the true solution at 0.1 intervals. Both the RMSE and MAE are expressed as a percentage of the range of the true solution. In Scenario 3, we provide the incorrect initial value, therefore we exclude the initial time period from all the error calculations by only calculating the error in the range from $t = 0.5$ to $t = 10$. 
6.3.1 The van der Pol Example

Although the classic van der Pol oscillator was originally proposed for modelling electrical circuits, it has found many applications. Most relevant to this study are the wide variety of applications in biology; many examples can be found in Britton’s textbook [15]. Here we present the van der Pol equation as a system of two initial value problems:

\[
\begin{align*}
\frac{dy_1}{dt}(t) &= \frac{1}{\varepsilon}(y_2 - \frac{y_1^3}{3} + y_1), \quad y_1(0) = 1 \quad (9a) \\
\frac{dy_2}{dt}(t) &= a - y_1, \quad y_2(0) = 1 \quad (9b)
\end{align*}
\]

The dynamics of the system are determined by the values of \(\varepsilon\) and \(a\). For the purposes of these investigations \(\varepsilon\) is chosen to be 0.1 and \(a\) is set to 0.5. The resulting system is stiff: there are regions where the solution varies rapidly and standard numerical schemes often fail to yield a physically meaningful approximation to the solution unless extremely small step sizes are used; Butcher provides a full discussion on stiff systems [Section 112, 86].

Using the methodology for building DBNs from systems of differential equations described in Chapter 4, a DBN, shown in Figure 25, is constructed based on Equation (9). The DBN incorporates an Euler approximation of the system of ODEs. The delta nodes (\(\Delta y_1\) and \(\Delta y_2\)) capture changes in quantities over time, calculated using the differential equations. Each delta node has as its parents the various terms needed to solve the appropriate differential equation.

As noted earlier in this thesis, the DBN provides a natural framework to handle instrument measurement uncertainty. The observed value for the variable to be approximated is assumed to contain a certain amount of measurement error. As can be seen in Figure 25, the observed \(y_1\) measurements are modelled as a continuous distribution where its mean is its parent node, the true variable value.
The model parameters $\varepsilon$ and $\alpha$ are also represented as continuous nodes. Distributions on the initial state model represent the population distribution. Distributions on the transition model allow the model parameters to vary in each time step by including a conditional dependency on its value in the previous time; they can therefore converge to values appropriate to the individual case over time, based on evidence streams. Linear Gaussian distributions are used and in each time step the standard deviation is scaled relative to the natural time step that is defined when the DBN is created, as the inference particles follow a Gaussian random walk:

$$\sigma = \sigma_{\text{natural}} \frac{h_{\text{natural}}}{h_{\text{adapted}}}$$

Figure 25: Model structure for the Euler DBN based on the van der Pol oscillator. Nodes $y_1, y_2, \Delta y_1$ and $\Delta y_2$ are deterministic nodes. Nodes $\varepsilon$ and $\alpha$ are continuous. Solid arcs connect nodes within a time slice; dashed arcs connect nodes between time slices. Evidence nodes (black) model the relationship between observed and true values, and inter-slice arcs on nodes $\varepsilon$ and $\alpha$ allow parameters to be tuned to the evidence over time.

In a similar manner, as outlined in Chapter 5, a DBN for the van der Pol system incorporating an RK2 solver is built. This is shown in Figure 26.
6.3.2 **Scenario 1: No Uncertainty in Model Parameters and No Evidence**

In this scenario, we examine the behaviour when the model parameters are known and do not contain any uncertainty. Although we do not use our DBNs in this way in the real applications, since the main benefit of using a DBN framework is to model uncertainty, it is instructive to examine the behaviour in these conditions.

6.3.2.1 **Option 1: Euler DBN with Fixed Time Steps**

Firstly, let us examine a DBN incorporating a first-order Euler approximation of the van der Pol oscillator, using fixed time steps. The van der Pol oscillator, with the chosen model parameters of $\varepsilon = 0.1$ and $a = 0.5$ is a stiff problem. In order to obtain a solution with reasonable numerical accuracy, very small time steps sizes must be used.

Figure 27 shows the $y_1$ output of the van der Pol DBN using 164 fixed time steps for the period from $t = 0$ to $t = 10$, along with the benchmark solution to which we compare our output. As the van der Pol oscillator does not have a closed-form solution, a benchmark solution was obtained using Matlab’s `ode45` solver [104]. Significant instability can be observed in the Euler approximation, which is due to the problem stiffness. A significant lag is also visible.
When the number of fixed time steps in the DBN is increased, it gives a solution much closer to the benchmark solution, as seen in Figure 28 where 2475 steps are used. This improved accuracy comes at a significant computational cost.

Figure 27: Scenario 1, Option 1: The $y_1$ output for the Euler DBN with 164 fixed time steps. Significant instability and significant lag is visible. This model had the correct starting parameters and no evidence was incorporated.

Figure 28: Scenario 1, Option 1: The Euler DBN using 2475 fixed time steps produces a more stable output. This improvement comes at a computational cost. This model had the correct starting parameters and no evidence was incorporated.
6.3.2.2  **Option 2: RK2 DBN with Fixed Time Steps**

In order to reduce the number of steps required a DBN that incorporates an RK2 solver can be used. As discussed in Chapter 5, run-time parameters allow us employ any method formulated in the general RK2 framework (7). For this scenario, we implement Modified (Midpoint) Euler. Using 164 steps, as was used with the Euler DBN above, the RK2 DBN yields a stable solution with a RMSE of 2.47%. The $y_1$ output is shown in Figure 29. Although not perfect, it is a significant improvement over Option 1 when 164 time steps were used for the same scenario.

![Figure 29: Scenario 1, Option 2: The $y_1$ output for the RK2 (Modified Euler) DBN with 164 fixed time steps. This model had the correct starting parameters and no evidence was incorporated.](image)

6.3.2.3  **Option 3: Euler DBN with Adaptive Time Steps**

Next we examine the behaviour when inference is performed on a DBN incorporating a first-order Euler solver using our Adaptive-Time Particle Filter.

As noted earlier, some additional information must be specified for the adaptive-time particle filter. The interval at which results will be reported, the summary interval, is in this case set to 0.1. Evidence, although not used in this particular scenario, must be specified as either continuous or instantaneous. In the case of the van der Pol model, $y_1^{\text{observed}}$ is the only node for which we have evidence, and this evidence is
instantaneous. The settings used for the van der Pol tolerance checking are: $M_1 = 1.5$; $M_2 = 0.9$; $q = 0.9$.

We select a range of tolerances (0.1, 0.01, 0.001, 0.0001), note the number of time steps executed in each case and compute the error relative to the benchmark solution. These errors are compared to the errors obtained using the same number of fixed time steps in the Euler DBN and the RK2 DBN. The results are shown in Table 9. Figure 30 shows these results as a log-log plot. It is clear that for the van der Pol DBN, Option 2 and Option 3 provide significant efficiency improvements. For a given number of steps, the RK2 DBN (Option 2) yields the lowest error and in this scenario is clearly the best option. However, when uncertainty is introduced, this is not the case, as will be shown in the next section.

<table>
<thead>
<tr>
<th>No. of Time Steps</th>
<th>Option 1: Euler DBN</th>
<th>Option 2: RK2 DBN</th>
<th>Option 3: DBN with Adaptive Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>164</td>
<td>24.38% / 14.28%</td>
<td>2.47% / 1.43%</td>
<td>18.33%/ 9.91% $\tau =0.1$</td>
</tr>
<tr>
<td>319</td>
<td>13.57% / 6.97%</td>
<td>0.62% / 0.29%</td>
<td>1.63% / 0.82% $\tau =0.01$</td>
</tr>
<tr>
<td>817</td>
<td>5.78% / 2.70%</td>
<td>0.10% / 0.04%</td>
<td>1.23% / 0.66% $\tau =0.001$</td>
</tr>
<tr>
<td>2475</td>
<td>1.96% / 0.89%</td>
<td>0.01% / 0.005%</td>
<td>0.44% / 0.24% $\tau =0.0001$</td>
</tr>
</tbody>
</table>

Table 9: Scenario 1: Number of steps executed for each option. The DBN was initialised with the correct starting parameters, which were not allowed to vary. No evidence was incorporated. For each option the error is calculated based on the difference between the DBN model and the benchmark solution at 0.1 intervals.
Figure 30: Scenario 1: A log-log plot of the number of steps executed versus the resulting RMSE for each option.

Figure 31 shows the results obtained by the adaptive particle filtering algorithm with a prescribed tolerance of $\tau=0.01$. This setting results in an average of 319 steps per particle and the RMSE for $y_1$ is 1.63%. This error is comparable with the error achieved using 2475 steps with the fixed time step Euler DBN. For our investigations this is deemed an acceptable numerical error and therefore the tolerance is set to 0.01 for Scenarios 2 and 3.

Figure 31: Scenario 1, Option 3: The $y_1$ output for the Euler DBN using Adaptive-Time Particle Filtering, produced with 319 steps. This model had the correct starting parameters and no evidence was incorporated.
6.3.3 Scenario 2: Incorrect Starting Parameters

An advantage of using a DBN with either fixed or adaptive time steps to model dynamic systems is that Bayesian networks can account for both numeric and data uncertainty by incorporating evidence from the true solution.

Again taking the van der Pol DBN, let us assume the true values for the model parameters are unknown and must be inferred from population values. The population means for $\varepsilon$ and $\alpha$ are set to incorrect values of 0.07 and 0.7 respectively (the true values used to generate evidence from the benchmark solution are $\varepsilon = 0.1$ and $\alpha = 0.5$). The standard deviations on the initial state models are set to 0.01 and 0.08 respectively. From one time step to the next, they are allowed to vary by setting the mean equal to the value on the previous slice and the standard deviation on the transition model to 0.001 and 0.015 respectively.

In order to select the correct model parameters, evidence (in this case, true values taken from the benchmark solution) is incorporated. As in many real-world situations, this evidence is both sparse and infrequent. Unlike the real world, however, in this scenario the evidence does not contain any measurement uncertainty.

The results for each of the options are shown in Table 10. For the fixed time step approaches, the number of time steps was increased by 100 in each iteration until a stable solution was obtained. The results given are the minimum number of steps required to obtain a stable solution. For the Euler DBN with fixed time steps, the number of steps required was 900 and for the RK2 DBN the number of steps required was 1100. For the adaptive approach, the tolerance was set 0.01 and the algorithm executed an average of 348 adaptive steps per particle.
Adaptive-Time Particle Filtering

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Method Description</th>
<th>Number of Time Steps</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option 1: Euler DBN with Fixed Steps</td>
<td>900</td>
<td>2.56%</td>
<td>1.10%</td>
<td></td>
</tr>
<tr>
<td>Option 2: RK2 DBN with Fixed Steps</td>
<td>1100</td>
<td>3.61%</td>
<td>1.78%</td>
<td></td>
</tr>
<tr>
<td>Option 3: Euler DBN with Adaptive Steps</td>
<td>348</td>
<td>2.23%</td>
<td>0.72%</td>
<td></td>
</tr>
</tbody>
</table>

Table 10: Scenario 2: Minimum number of time steps required to obtain a stable solution for each option. The DBN was initialsed with incorrect starting parameters and sparse evidence was incorporated.

It is interesting to note that a much larger number of steps are required by both the Euler DBN and the RK2 DBN when the model terms are allowed to vary when compared to Scenario 1 where model terms do not vary.

This is because when the model terms are allowed to vary, each particle selects a value from the sample space. Certain values, for example, $\varepsilon = 0.01$ result in a significantly stiffer solution. While these particles may be unlikely and will be eliminated once new evidence is incorporated, they must be considered until then. As mentioned earlier, in order to obtain a stable solution for a stiffer problem, smaller time step sizes are needed. Therefore, in order to handle the small number of particles that result in a stiffer problem, very small step sizes must be used by both the Euler and RK2 fixed time step DBNs. If not, the solution for certain particles can cause some spurious spikes, seen in Figure 32 when using 700 steps for Option 1. Increasing the number of steps to 900 eliminates these spikes (Figure 33) and produces an accurate result for the Euler fixed time step DBN. Requiring this large number of steps results in significant computational cost and manually choosing the appropriate number of steps is time consuming and cumbersome. It might not even be possible to know when we have achieved an appropriate number of steps since in general we don’t have a true solution to compare to.
Adaptive-Time Particle Filtering

Figure 32: Scenario 2, Option 1: The $y_1$ results from the Euler DBN using 700 fixed time steps. Parameters are initialised with incorrect values and intermittent evidence is provided. At approximately $t=3.2$ we can see that some particles resulted spurious spikes.

Figure 33: Scenario 2, Option 1: The $y_1$ results from the Euler DBN using 900 fixed time steps. Parameters are initialised with incorrect values and intermittent evidence is provided. By increasing the number of steps a stable solution is obtained.

On the other hand, the adaptive approach allows each particle to choose its own appropriate step size. Therefore, the minority of particles in the region that results in a very stiff solution can adopt a very small step size, whereas the majority of
particles in the region that results in a less stiff solution can adopt a larger step size. Thus from Scenario 1 to Scenario 2, the average number of steps executed per particle increased from 319 to only 348, whereas the number of steps required by the RK2 DBN increased from 164 steps to 1100 steps, as all particles required a smaller step size to accommodate the stiffest trajectory.

Figure 34 shows the output for $y_1$ for Option 3 obtained using only 348 adaptive time steps, along with the evidence that is used by the DBN to align its parameters with reality.

In Figure 35 and Figure 36, we can see that by incorporating evidence from the benchmark solution, the DBN infers corrected values for the model parameters from the starting values. For both $\varepsilon$ and $\alpha$, the inferred values converge towards the true values. The DBN chooses values for the model parameters that result in a solution for $y_1$ that is as close as possible to the observed values for $y_1$, thereby accounting for both the data and numerical error.

![Figure 34: Scenario 2, Option 3: The $y_1$ output for the DBN using Adaptive-Time Particle Filtering, when parameters are initialised with incorrect values and intermittent evidence is provided. Only 345 adaptive steps were needed.](image)

Adaptive-Time Particle Filtering
Figure 35: Scenario 2, Option 3: $\varepsilon$ is initialised with an incorrect value. However, the DBN using Adaptive-Time Particle Filtering and intermittent evidence infers a corrected value, accounting for both data uncertainty and numerical error. The graph shows both the mean and one standard deviation for $\varepsilon$ as inferred by the DBN.

Figure 36: Scenario 2, Option 3: $\alpha$ is initialised with an incorrect value. However, the DBN using Adaptive-Time Particle Filtering and intermittent evidence infers a corrected value, accounting for both data uncertainty and numerical error. The graph shows both the mean and one standard deviation for $\alpha$ as inferred by the DBN.
6.3.4 **Scenario 3: Incorrect Starting Parameters and Incorrect Initial Values**

In many real-world models, not all model variables have good initial values available, so these must be estimated from experience. This can lead to situations where the model must adjust rapidly over the first few time steps. To simulate this situation, we consider once again the van der Pol model (9), but with the initial value of the first component set to $y_1(0) = 4$ and the incorrect starting parameters used in Scenario 2.

As can be seen in Figure 37, the DBN using Adaptive-Time Particle Filtering handles this situation by initially using an increased number of smaller steps. In this figure we can also observe that larger numbers of smaller steps are used at times where dynamics change rapidly.

![Figure 37](image)

Figure 37: Scenario 3, Option 3: The average number of steps per particle used by the adaptive particle filtering algorithm in each summary interval.

Using $\tau = 0.01$ a reasonable output is obtained with an average of only 357 adaptive steps, as shown in Figure 38. The Adaptive-Time Particle Filtering RMSE = 2.60% as shown in Table 11, again the RMSE is calculated at 0.1 intervals.
Figure 38: Scenario 3, Option 3: Here even though $y_1$ at $t = 0$ was set to 4, which is a very poor starting estimate, the DBN using adaptive time particle filtering still produces reasonable results with only 357 steps of varying length.

In contrast, a significant increase in the number of fixed time steps is required for both the basic Euler and the RK2 fixed time step DBNs. The minimum number of steps required for the basic Euler fixed time step DBN is 2100 steps (Figure 39); with fewer steps, it cannot capture the initial period of rapidly changing dynamics. Using the RK2 DBN at least 4100 steps must be used (Figure 40).

This scenario highlights the potential efficiency improvements that can be gained using the adaptive approach.
Figure 39: Scenario 3, Option 1: The output from the Euler DBN with fixed time steps and a poor starting estimate of 4 for $y_1$. This required 2100 fixed time steps.

Figure 40: Scenario 3, Option 2: The output from the RK2 DBN with fixed time steps and a poor starting estimate of 4 for $y_1$. This required 4100 fixed time steps.

<table>
<thead>
<tr>
<th>Option</th>
<th>Number of Time Steps</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option 1: Euler DBN with Fixed Steps</td>
<td>2100</td>
<td>3.42%</td>
<td>1.71%</td>
</tr>
<tr>
<td>Option 2: RK2 DBN with Adaptive Steps</td>
<td>4100</td>
<td>3.97%</td>
<td>1.91%</td>
</tr>
<tr>
<td>Option 3: Euler DBN with Adaptive Steps</td>
<td>357</td>
<td>2.60%</td>
<td>1.04%</td>
</tr>
</tbody>
</table>
Table 11: Scenario 3: Comparison of results from Euler and RK2 DBNs with fixed time steps and a DBN with adaptive time steps where \( y_4 \) at \( t = 0 \) was set to the incorrect value of 4.

6.4 Full Steps and Sub-Steps

We also explored another technique for improving the efficiency of the inference task that we termed full steps and sub-steps. At a specified interval, for example every minute, full steps are run. These are identical to the steps in fixed time steps particle filtering i.e.

1. Particles are generated for each node in the network based on its initial state distribution and conditioned on its parents.
2. Particles are weighted based on the evidence.
3. Particles are summarised and a weighted mean and weighted standard deviation is reported.
4. Particles are re-sampled prior to moving to the next step.

In between each full step, sub-steps are executed. Sub-steps are much more efficient as not all nodes are sampled and not all steps in the particle filtering algorithm are executed. In each sub-step:

1. Particles are generated only for the \( \text{delta} \) and \( \text{variable} \) deterministic nodes (see Section 4.1 for an explanation of these nodes). Model parameter values are not allowed to vary in between full steps.
2. Particles are not weighted as there is never new evidence to be considered in a sub-step.
3. Particles are not summarised as the user is seldom interested in the results at such a fine granularity.
4. Particles are not re-sampled.

The advantage of this approach is that significant efficiency improvements are gained by using sub-steps instead of full steps. The sub-steps can have a very small step size to accommodate very stiff problems. Take for example Scenario 3 above
with the basic Euler DBN. It takes approximately 6786 seconds to execute 2100 full steps from $t = 0$ to $t = 10$ using 100000 samples. In this case, the step length is $\frac{10}{2100}$. Using full steps at 0.1 intervals, and sub-steps of length $\frac{10}{2100}$ the execution time is reduced to 1251 seconds.

However the sub-steps approach shares its drawbacks with the other fixed step approaches in that significant effort is required to select the appropriate full step length and sub-step length. The full step length must still match the granularity of the evidence time stamps. For example, if the time stamps on the evidence are at a 1-minute granularity, full steps must be run every minute, even if the evidence may be an average of 60 minutes apart. Having full steps only every 60 minutes may not align exactly with the evidence and so the evidence would have to be approximated to the nearest 60 minute time step, thus introducing more uncertainty.

6.5 Conclusions

In this chapter, four approaches for re-estimating model parameters and accounting for data error and numerical error were evaluated:

1. Performing fixed time step inference on the DBN incorporating a first-order Euler solver.
2. Performing fixed time step inference on the DBN incorporating an RK2 solver.
4. Performing inference using full steps and sub-steps.

We conclude that the third of these, the adaptive approach, is the easiest to use and the most efficient when data uncertainty exists.

Using the Adaptive-Time Particle Filtering algorithm, the tolerance is set at the beginning and no change or fine-tuning is required to run inference in more
challenging scenarios. On the other hand, using the fixed time step approach, for each scenario a different fixed step size is necessary. In each scenario we had to experiment with changing the step size when using the fixed time step approach, until a stable solution was obtained. This is a cumbersome and time consuming task. In the van der Pol example it is relatively straight-forward to assess when the appropriate step size is reached as the solution can be compared to the benchmark solution. In practical applications, this is not always possible and so it may be difficult to know when the appropriate number of steps has been achieved.

In the next chapter, we use our methodology for encapsulating ODEs in DBNs on a practical medical example and examine the efficiency improvements to be gained by performing inference using our Adaptive-Time Particle Filter.
7. A DBN to Model Glycaemia

As explained in Section 2.4.2, hyperglycaemia is commonly experienced in critically ill patients. It is associated with increased morbidity and mortality. To regulate glycaemia, insulin is administered intravenously. In this chapter, we use our new methodology that was presented in Chapter 4 to construct a DBN based on an existing ODE model. This DBN is used to predict a patient’s glucose level in response to an insulin infusion and intravenous nutrition. The model is validated on data from real patients and the results compared to an existing methodology using the same underlying ODE model.

7.1 The System of ODEs

The starting point for constructing the DBN is the ICU-Minimal Model (ICU-MM) of Van Herpe et al. [19], which was introduced in Section 3.5.1. It is a model for predicting plasma glucose levels in critically ill patients who are in receipt of a glucose and insulin infusion. It is described by a system of four differential equations:

\[
\begin{align*}
\frac{dG}{dt}(t) &= (P_1 - X(t))G(t) - P_1 G_b + \frac{F_G}{V_G} \quad (10a) \\
\frac{dX}{dt}(t) &= P_2 X(t) + P_3 (I_1(t) - I_b) \quad (10b) \\
\frac{dI_1}{dt}(t) &= \alpha \max(0, I_2(t)) - n(I_1(t) - I_b) + \frac{F_I}{V_I} \quad (10c) \\
\frac{dI_2}{dt}(t) &= \beta \gamma (G(t) - h) - n I_2(t) \quad (10d)
\end{align*}
\]

Here, \( G \) is the plasma glucose level, \( X \) is the effect insulin has on the plasma glucose, \( I_1 \) is the plasma insulin level and \( I_2 \) the endogenous insulin produced by the pancreas.
Van Herpe et al. [19], describe the model terms as follows:

\( G \): The glucose concentration in blood plasma.

\( I_1 \): The insulin concentration in blood plasma.

\( X \): The effect of insulin on net glucose disappearance. \( X \) is proportional to the insulin in the remote compartment.

\( I_2 \): The remote insulin. This variable does not have a strictly defined clinical interpretation but can be considered the fraction of insulin concentration derived from the endogenous insulin secretion.

\( G_b \): The basal value of plasma glucose.

\( I_b \): The basal value of plasma insulin.

\( F_I \) and \( F_G \): The intravenous rate of insulin and glucose are the two input variable to the model.

\( V_G \): The glucose distribution volume which is set to \( \text{body mass} \times 1.6 \).

\( V_I \): The insulin distribution volume which is set to \( \text{body mass} \times 120 \).

\( P_1 \): The glucose effectiveness (i.e. the fractional clearance of glucose) when insulin remains at basal level.

\( P_2 \): The fractional rate of net remote insulin disappearance.

\( P_3 \): The fractional rate of insulin-dependent increase.

\( \gamma \): The proportion by which endogenous insulin is released when glycaemia exceeds a threshold.

\( \beta \): \( \beta \) has no physiological significance. It is included to keep units correct and set equal to 1 min.
\( \alpha \): \( \alpha \) has no physiological significance. It is included to keep units correct but does require individualisation.

7.2 The DBN Derived from the ODEs

Using our new procedure that was described in Chapter 4, the DBN structure is derived from the ICU-MM. Each equation is mapped to a subnet in the DBN. The DBN contains both hidden and observed nodes. Hidden (continuous or discrete) random nodes are dark grey, observed nodes are black and deterministic nodes are light grey.

Delta nodes capture changes in quantities over time. These changes are calculated using the differential equations of the ICU-MM (10). Each delta node has, as parent nodes, the various terms needed to solve the appropriate differential equation.

To illustrate this, Figure 41 shows the section of the DBN that is related to Equation (10a) of the ICU-MM. Here, the \( \Delta G \) node determines the per time step change in plasma glucose levels. The current plasma glucose level is determined based on the glucose level and \( \Delta G \) calculated in the previous time slice. Each of the terms in the differential equation for \( G \) is represented as a parent node of \( \Delta G \).

![Figure 41: Subnet of DBN for Equation (10a) of ICU-MM](image-url)
The DBN is expanded as described in Section 4.3. Model parameters are allowed to vary over time. Their initial mean values are based on the literature [59] [89] [106]. Each model parameter is allowed to vary in each $h$ by including a conditional dependency on its value in the previous time. In this way, they can converge to values appropriate to the individual case over time, based on the evidence. The model parameter means are shown, along with the standard deviations for the initial state and transition models, in Table 12.

Model parameters are represented as truncated Gaussian nodes, in order to constrain the DBN to postulate values that are not unrealistic for nodes. For example, the true value for $P_1$ cannot be a negative value, only positive values are possible.

Limits were also placed on some deterministic nodes using min/max functions. For example, it is not possible to have a negative quantity of glucose in plasma, so a limit is placed on node $G$ to reflect this.

The observed value for plasma glucose (Observed $G$ in the DBN) is assumed to contain a certain amount of measurement error. It is therefore modelled with a Gaussian distribution whose mean is its parent node, the actual plasma glucose level, $G$. Likewise, the data from the ICU reflects the prescribed intravenous infusion rates for insulin and glucose; the actual administered rates may be different. Therefore, the actual rates are modelled with Gaussian distributions whose means are the prescribed rates. I.V. status nodes are introduced to indicate if the patient is in receipt of an infusion. When the I.V status is on, the actual rates are modelled with Gaussian distributions whose means are the prescribed rates. In this way, data uncertainty is handled. When the I.V status is off, the actual rates are zero; they do not differ from the prescribed rates. Note that while we use Gaussian distributions in this model, other distributions could be used where suitable.

In a similar manner, a subnet for each of the other equations (10b -10d) is added to the DBN. The full DBN is shown in Figure 42.
Figure 42: The ICU-MM system of differential equations mapped to a DBN.
A DBN to Model Glycaemia

<table>
<thead>
<tr>
<th>Node</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Initial State</td>
<td>Model Transition Model</td>
</tr>
<tr>
<td>$G_b$</td>
<td>$G_0$</td>
<td>40</td>
<td>0.05</td>
</tr>
<tr>
<td>$I_b$</td>
<td>$-13 + 0.22 IBW_R$ (see footnote)</td>
<td>5</td>
<td>0.005</td>
</tr>
<tr>
<td>$P_1$</td>
<td>-0.0131 per min</td>
<td>0.013</td>
<td>0.005</td>
</tr>
<tr>
<td>$P_2$</td>
<td>-0.0135 per min</td>
<td>0.013</td>
<td>0.002</td>
</tr>
<tr>
<td>$P_3$</td>
<td>$2.9E-6 ml/(min^2\mu U)$</td>
<td>$2.0E-6$</td>
<td>$1.0E-6$</td>
</tr>
<tr>
<td>$h$</td>
<td>136 mg/dl</td>
<td>30</td>
<td>0.1</td>
</tr>
<tr>
<td>$n$</td>
<td>0.013 per min</td>
<td>0.1</td>
<td>0.001</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>3.11</td>
<td>0.5</td>
<td>0.01</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>0.00536 per min</td>
<td>$1.0E-5$</td>
<td>$1.0E-5$</td>
</tr>
<tr>
<td>Weight</td>
<td>Observed Weight</td>
<td>1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 12: Conditional probability tables for the model parameters of the ICU-MM

The nodes $V_g$ and $V_i$ are modelled as deterministic nodes. Their values are calculated as $1.6 \times weight$ and $120 \times weight$ respectively.

7.3 Analysis of Number of Particles Required

In order to assess the number of particles that should be used, we carried out a sensitivity analysis. This involved running the DBN simulation 5 times for each number of particles. For each variable the Root Mean Square Error (RMSE) is calculated from the inter-run difference. In Figure 43 below, the average RMSE for

$\text{IBW}_R$ as defined in [106] refers to the body weight relative to the ideal body weight (defined in the Metropolitan Life Insurance Tables) expressed as a percentage.
each variable is expressed as a percentage of the maximum variable value. The graph is based on the results from Patient 23 for the time period 300-500 minutes. It can be observed that as the number of particles is increased, the resulting predictions are more consistent. For the results presented in this chapter, 50000 particles were used as Figure 43 indicates this number of particles produces consistent results.

Figure 43: This graph shows how, as the number of particles is increased, the resulting predictions are more consistent.

7.4 Evaluation

7.4.1 Description of the Data

For comparative evaluation of the methods, data was used from patients in the ICU of University Hospital Galway (UHG).

Two distinct datasets were used. Firstly data from historical patient records were used. The dataset was previously described in a publication co-authored by this author [107]. These patients were selected by Dr. Brian Harte and the data was extracted by Ms. Anne Mulvey. The patients were not on specific insulin therapy trials, so the dataset only contains routine measurements. Accordingly, plasma glucose measurements are infrequent and sporadic. At times, changes in the plasma
glucose cannot be explained with the data available; this may be because either information is incomplete (e.g., if the patient was administered glucose that was not recorded) or measurements are inaccurate (e.g., due to data-entry errors or measurement assay).

However, this dataset provides a realistic sample of the routine data available in a busy ICU where a system, such as the one described here, could eventually be deployed for patient monitoring and simulations of the effects of therapies.

Data from patients with the following characteristics was selected:

- Sepsis as a primary diagnosis
- Non-diabetic
- Not in receipt of steroids
- No major organ failure
- Only in receipt of parenteral nutrition.

The second dataset, a subset of which was used for a publication by our group [108], was actively gathered for this project by a medical student, Mr. Conor Lane, under the supervision of Prof. John Laffey. Once again the patients were not on specific insulin therapy trials. Glucose measurements were taken hourly for a 12 hour period for 9 patients. Care was taken to ensure the correct time stamp was entered on all relevant data records. As the medical student taking the extra measurements was only available for a short period of time, the patient selection criteria were relaxed. The patient characteristics for this group were:

- Sepsis as a primary diagnosis
- Non-diabetic
- No major organ failure
Permission for extracting and gathering this data was given by the Galway Research Ethics Committee, UHG. All records were anonymised and stored on encrypted drives.

7.4.2 DBN Results for a Sample Patient

For the purposes of this discussion, a reasonably stable patient (Patient 30) from the first dataset is selected. In this section, inference is performed using the standard particle filtering algorithm with a 1-minute time step. Later, in Section 7.4.5, we compare the results using fixed-time step inference and adaptive-time step inference.

As can be seen in Figure 44, the observations for plasma glucose are intermittent; the DBN therefore makes internal predictions of plasma glucose levels in between observations. The accuracy of the predictions can be evaluated by comparing the predicted value at the time of a measurement to the actual value, prior to the DBN incorporating the actual value as evidence. It should be noted that the measurement may not be perfectly accurate. For example, at approximately 34 hours the glucose measurement jumps to over 200mg/dl despite the fact that there is no change in the infusions. It is likely that this, and the drop to 130mg/dl at approximately 43 hours, are both incorrect measurements.

In Figure 44, the dark lines are the mean values inferred by the DBN at each minute, and the lighter shaded areas show one standard deviation, to give a sense of the uncertainty associated with each prediction.
Figure 44: The upper graph shows the prescribed insulin and glucose infusion rates. The lower graph shows the measured glucose levels as squares and the predicted mean plasma glucose level in blue along with a shaded area showing the predicted standard deviation.

One can observe that the mean value often jumps when a new observation becomes available. There are factors that are not captured in the model that influence plasma glucose levels. Because of these, mean values predicted by the model can drift from reality in between observations. When a new observation is available, the model tends to realign with it.

It is informative to consider how the standard deviations vary over time. Because the DBN always assumes variability of values over time, and because observations of plasma glucose levels are available only intermittently, as the time since the last
observation increases, the range of possible values increases, so the uncertainty of the predictions also increases. Whenever an observation is provided, the DBN's plasma glucose prediction realigns to a value close to this, and its uncertainty collapses.

It should be noted that glucose measurements (evidence) are very infrequent relative to the time step (1 minute) used in the DBN. This time step is determined by the system’s dynamics.

7.4.3 Comparison to ODEs with Parameter Re-estimation
Van Herpe et al. re-estimate the model parameters by using an unconstrained nonlinear optimization algorithm [109]. Attempts to apply that specific approach on our dataset, which has less frequent evidence, were unsuccessful; the optimisation algorithm frequently failed to converge. Therefore, a variant on the method is used [110] which allows bounds to be placed on the parameters to be re-estimated. As our glucose measurements are less frequent, all measurements within the previous 24 hours are used each time parameters are re-estimated. The dashed line in Figure 45 shows the trajectory for the ODEs with re-estimation.

Figure 45: The glucose predictions from the DBN compared to an Euler approximation with an optimisation algorithm for parameter re-estimation shown as dashed green line.
It can be observed in Figure 45 that the Euler approximation with the optimisation algorithm performs rather poorly for patient 30. The set of parameters selected by the algorithm do not allow the model to respond to the changes in the infusions. At 68 hours, it predicts a glucose level of zero which is not realistic.

7.4.4  **DBN Results for Twelve Patients**

Figure 46 shows a comparison of the average root mean squared error calculated using the difference between the actual glucose measurements and glucose predictions for twelve ICU patients, chosen at random. In 10 out of 12 cases our DBN method outperforms the other method, very substantially so in some cases, such as Patient 101, where the ODE solution gives an error of 48.57% but our DBN method produces a much lower error of 9.28%. In the case of Patient 60, the optimisation algorithm could not find appropriate parameters, whereas the DBN framework was able to make predictions, albeit with a high RMSE of 38.98%. Hence the ODE result for P60 is not plotted in Figure 46.

It should be noted that while we are comparing the RMSE of predictions relative to measured values, the measured values are not always perfectly accurate.
A DBN to Model Glycaemia

<table>
<thead>
<tr>
<th>Patient</th>
<th>RMSE</th>
<th>ODE with Re-estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>P23</td>
<td>11.90%</td>
<td>16.17%</td>
</tr>
<tr>
<td>P91</td>
<td>19.43%</td>
<td>41.18%</td>
</tr>
<tr>
<td>P30</td>
<td>16.45%</td>
<td>33.33%</td>
</tr>
<tr>
<td>P40</td>
<td>34.75%</td>
<td>45.49%</td>
</tr>
<tr>
<td>P61</td>
<td>16.09%</td>
<td>19.90%</td>
</tr>
<tr>
<td>P24</td>
<td>16.32%</td>
<td>29.60%</td>
</tr>
<tr>
<td>P64</td>
<td>7.82%</td>
<td>17.81%</td>
</tr>
<tr>
<td>P09</td>
<td>20.74%</td>
<td>20.40%</td>
</tr>
<tr>
<td>P01</td>
<td>25.17%</td>
<td>21.41%</td>
</tr>
<tr>
<td>P102</td>
<td>13.19%</td>
<td>19.19%</td>
</tr>
<tr>
<td>P101</td>
<td>9.28%</td>
<td>48.57%</td>
</tr>
<tr>
<td>P60</td>
<td>38.98%</td>
<td>Nan</td>
</tr>
<tr>
<td>Average (excluding P60)</td>
<td><strong>17.38%</strong></td>
<td><strong>28.46%</strong></td>
</tr>
</tbody>
</table>

Table 13: Average RMSE (lower is better) for 12 patients, comparing results using our DBN approach with fixed-time steps and an Euler approximation with parameter re-estimation.

To verify these results are significant we carried out a paired t-test. Comparing the null hypotheses, that the DBN RMSE is equal to the ODE RMSE, to the hypotheses that the DBN RMSE is lower that the ODE RMSE, the paired t-test gives a p-value of 0.006. At a significance level of 0.05, we reject the null hypotheses. The results are statistically significant. For our patient cohort, the RMSE using the DBN methodology is lower than the RMSE using an Euler approximation with re-estimation.

These results highlight the advantages of incorporating the ODE system in a DBN framework. The DBN model adjusts parameters as soon as it receives the first piece of evidence, in contrast to methodologies that require a calibration window containing a number of observations. The DBN reacts to both gradual and sudden changes in model parameters as it tracks a range of possible trajectories. The DBN accounts for data uncertainty and model uncertainty in a principled manner.
7.4.5 Results Using Adaptive-Time Particle Filtering

By performing inference with the Adaptive-Time Particle Filter introduced in Chapter 6, these benefits can be gained much more efficiently. In Figure 47, a comparison of the RMSEs of the glycaemia predictions using the fixed-time step inference and the adaptive-time step inference are shown. Overall, there is no significant difference in accuracy. The results for most patients are slightly better with the adaptive method, however when we compare the average over all patients the difference is less than 1%.

![Figure 47: Average RMSE for 12 patients, comparing results using a DBN with fixed time steps and one with the new Adaptive-Time Particle Filtering algorithm proposed in this thesis. Also shown are the results from the Euler approximation with parameter re-estimation.](image)
A DBN to Model Glycaemia

<table>
<thead>
<tr>
<th>Patient</th>
<th>ICU-MM DBN with fixed-time steps</th>
<th>ICU-MM DBN with adaptive-time steps</th>
<th>ODE with Re-estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>P23</td>
<td>11.90%</td>
<td>11.42%</td>
<td>16.17%</td>
</tr>
<tr>
<td>P91</td>
<td>19.43%</td>
<td>20.28%</td>
<td>41.18%</td>
</tr>
<tr>
<td>P30</td>
<td>16.45%</td>
<td>17.00%</td>
<td>33.33%</td>
</tr>
<tr>
<td>P40</td>
<td>34.75%</td>
<td>32.83%</td>
<td>45.49%</td>
</tr>
<tr>
<td>P61</td>
<td>16.09%</td>
<td>15.03%</td>
<td>19.90%</td>
</tr>
<tr>
<td>P24</td>
<td>16.32%</td>
<td>16.45%</td>
<td>29.60%</td>
</tr>
<tr>
<td>P64</td>
<td>7.82%</td>
<td>7.43%</td>
<td>17.81%</td>
</tr>
<tr>
<td>P09</td>
<td>20.74%</td>
<td>17.39%</td>
<td>20.40%</td>
</tr>
<tr>
<td>P01</td>
<td>25.17%</td>
<td>24.54%</td>
<td>21.41%</td>
</tr>
<tr>
<td>P102</td>
<td>13.19%</td>
<td>12.58%</td>
<td>19.19%</td>
</tr>
<tr>
<td>P101</td>
<td>9.28%</td>
<td>12.28%</td>
<td>48.57%</td>
</tr>
<tr>
<td>P60</td>
<td>38.98%</td>
<td>37.05%</td>
<td>Nan</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>19.18%</strong></td>
<td><strong>18.69%</strong></td>
<td><strong>Nan</strong></td>
</tr>
</tbody>
</table>

Table 14: Average RMSE for 12 patients, comparing results using a DBN with fixed time steps and one with the new Adaptive-Time Particle Filtering algorithm proposed in this thesis. Also shown are the results from the Euler approximation with parameter re-estimation.

While 60 1-minute time steps are required per patient per hour using the fixed time steps approach, an average of 12 adaptive time steps per patient per hour are executed by the adaptive algorithm. The average number of steps executed per patient per hour is shown in Table 15. The fixed time step approach required 5 times the number of steps. Using the adaptive time approach inference is performed much more efficiently without compromising the accuracy of the predictions. In general, analysing data from sensor streams in real time imposes computational constraints because data must be processed as they arrive, so time savings like this can be important.
A DBN to Model Glycaemia

<table>
<thead>
<tr>
<th>Fixed Steps</th>
<th>Adaptive-Time Particle Filtering</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Patients</td>
<td>P23</td>
</tr>
<tr>
<td>60</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 15: Average number of adaptive steps per hour for each patient compared to 60 fixed 1-minute step: this shows that the new adaptive time step algorithm yields significant time savings.

7.4.6 **Comparison to a Trivial Model**

As shown in Section 7.4.4, the DBN methodology out-performs the technique used by the ICU-MM authors. However, it is useful to also compare our methodology to a trivial model. In the trivial model, the most recent observation of plasma glucose is used as the predicted value until a new observation is made. Figure 48 and Table 16 show a comparison of the results from our ICU-MM DBN with adaptive steps and the trivial model. Although in 10 out of 12 cases the DBN out-performs the trivial model, a paired t-test indicates that the differences, at a significance level of 0.05, are not statistically significant (p-value = 0.134).

Although we call this model a trivial model, the term may be misleading. This data is from patients who are actively managed in the ICU. Clinicians’ decisions regarding insulin dosage and the frequency of glucose measurements are based on their expertise, experience and clinical guidelines. Clinicians aim to keep glucose levels stable and within a safe range. Essentially, one could take the view that the trivial model is aligned with the decisions made by the clinicians.
Figure 48: Comparison of RMSEs from the ICU-MM DBN and the trivial model. In the trivial model the most recent glucose measurement is used as the glucose prediction until a new measurement is observed.
A DBN to Model Glycaemia

<table>
<thead>
<tr>
<th>Patient</th>
<th>ICU-MM DBN with adaptive-time steps</th>
<th>Trivial Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>P23</td>
<td>11.42%</td>
<td>15.85%</td>
</tr>
<tr>
<td>P91</td>
<td>20.28%</td>
<td>26.15%</td>
</tr>
<tr>
<td>P30</td>
<td>17.00%</td>
<td>19.50%</td>
</tr>
<tr>
<td>P40</td>
<td>32.83%</td>
<td>35.15%</td>
</tr>
<tr>
<td>P61</td>
<td>15.03%</td>
<td>16.31%</td>
</tr>
<tr>
<td>P24</td>
<td>16.45%</td>
<td>25.68%</td>
</tr>
<tr>
<td>P64</td>
<td>7.43%</td>
<td>17.90%</td>
</tr>
<tr>
<td>P09</td>
<td>17.39%</td>
<td>20.35%</td>
</tr>
<tr>
<td>P01</td>
<td>24.54%</td>
<td>9.80%</td>
</tr>
<tr>
<td>P102</td>
<td>12.58%</td>
<td>15.53%</td>
</tr>
<tr>
<td>P101</td>
<td>12.28%</td>
<td>10.78%</td>
</tr>
<tr>
<td>P60</td>
<td>37.05%</td>
<td>48.36%</td>
</tr>
<tr>
<td>Average</td>
<td><strong>18.69%</strong></td>
<td><strong>21.78%</strong></td>
</tr>
</tbody>
</table>

Table 16: Comparison of RMSEs from the ICU-MM DBN with adaptive-time steps and the trivial model. In the trivial model the most recent glucose measurement is used as the glucose prediction until a new measurement is observed.

7.5 Limitations of the ICU-MM

The ICU-MM model has a number of limitations.

Firstly the model is very sensitive to $G_b$, which is basal glucose. In the ICU-MM the glucose level will always tend towards the inferred value for $G_b$. Therefore the model is very sensitive to the initial state and transition distributions on this node. As mentioned, the ICU-MM is based on the Minimal Model by Bergman et al. [73], which is designed for diagnosing diabetes. In the case of Bergman et al.’s Minimal Model, basal glucose is a patient’s glucose level following an overnight fast. For critically ill patients it is rarely possible to measure the basal glucose level, as patients may not be fasting on entry to the ICU and their bodies are already under stress. Van Herpe et al. [90] therefore use the following equation to estimate basal glucose:
We experimented with using this equation but found better results setting the initial value for basal glucose to the first glucose measurement with a large standard deviation. In the context of the ICU-MM it is difficult to know how the basal glucose node should be allowed to vary over time. Does a patient’s basal glucose change as their condition deteriorates or improves? The conditional probability tables for basal glucose chosen for the patient cohort in this study may not be optimal and indeed may not work for other patient cohorts. We attempted to learn the optimal distributions for $G_b$ from the data, but this was unsuccessful perhaps because the dataset is too small.

In our implementation of the ICU-MM, there were 10 model parameters to be individualised to the patient. There are possibly many combinations of values for these parameters that can explain the evidence. In Appendix 2, a full set of graphs for all the model parameters for Patient 30 is shown. It can be observed that some nodes, for example $\alpha$ (shown below in Figure 49) or gamma, do not converge to the patient-specific values.

![Figure 49](image)

Figure 49: Values Inferred by the ICU-MM DBN for the model parameter alpha for Patient 30. Note the evidence has little influence on this node and so it does not converge to a patient-specific value.

There may be a case, in future work, for fixing such values so they cannot vary.

The ICU-MM does not consider enteral feed. Typically patients are moved to enteral feeding as soon as possible. The data that was collected specifically for this project
was mainly comprised of patients in receipt of enteral feeds. This patient data could therefore not be used with the ICU-MM. For this reason we undertook some work incorporating a different ODE model.
7.6 The ICING DBN

The ICING model [82] offers some promise as there is only one time-varying parameter and it can account for both enteral as well as parenteral feeding. Also, it does not require an estimate of basal glucose. The ICING model evolved from 2 previous models [80] [81] and is described as follows:

\[
\frac{dBG}{dt}(t) = -p_GBG(t) - S_1BG(t) \frac{Q_t}{1 + \alpha_GQ(t)} + \frac{P(t) + EGP_b - CNS}{V_G}
\]  
(11a)

\[
\frac{dQ}{dt}(t) = n_i(I(t) - Q(t)) - n_c \frac{Q_t}{1 + \alpha_GQ(t)}
\]  
(11b)

\[
\frac{dl}{dt}(t) = -n_kI(t) - \frac{n_Ll(t)}{1 + \alpha_Ll(t)} - n_i(I(t) - Q(t)) + \frac{u_{ex}(t)}{V_i} + (1 - x_L) \frac{u_{en}}{V_L}
\]  
(11c)

\[
\frac{dP1}{dt}(t) = -d_1P1 + D(t)
\]  
(11d)

\[
\frac{dP2}{dt}(t) = -\min(d_2P2, P_{max}) + d_1P1
\]  
(11e)

\[
P(t) = \min(d_2P2, P_{max}) + PN(t)
\]  
(11f)

\[
u_{en}(t) = k_1e^{-l(t)k_2/r_3}
\]  
(11g)

For a full description of this model please refer to Lin et al. [82]. Here only a brief description of the terms, based on Lin et al. [82], is given.

\(BG(t)\) is the absolute blood glucose level. \(EGP_b\) is the endogenous glucose production rate for a patient receiving no exogenous glucose or insulin, \(CNS\) is the insulin independent central nervous system glucose uptake.

In Equation (11a), insulin independent glucose removal (excluding central nervous system uptake \(CNS\)) and the suppression of endogenous glucose production from \(EGP_b\) with respect to \(BG(t)\) are compounded and represented by \(p_G\).
Insulin mediated glucose removal and the suppression of endogenous glucose production are similarly compounded and represented by $S_I$. $S_I$ represents the whole-body insulin sensitivity. Its variation through time can be significant, particularly for highly dynamic, critically ill patients.

Equations (11b) and (11c) define the insulin pharmacokinetics. $Q(t)$ represents the effect of previously infused insulin being utilised over time in the interstitium (the space between cells in a tissue). The receptor-bound insulin $\frac{Q_t}{1 + \alpha_C Q(t)}$ is the insulin effective for glucose removal to cells.

Equations (11d)–(11f) represent the gastric absorption of glucose, where $D(t)$ (mmol/min) represents dextrose from enteral feeding, $P1$ (mmol) represents the glucose in the stomach and $P2$ (mmol) is for the gut. Transport rates between the compartments are $d_1$ (min⁻¹) and $d_2$ (min⁻¹). Parenteral dextrose is represented by $PN(t)$.

Equation (11g) is a generic representation of endogenous insulin production, with the base rate being $k_1$ (mU/min). This is suppressed with elevated plasma insulin levels. The exponential suppression is described by generic constants $k_2$ and $k_3$.

In our DBN implementation, shown in Figure 50, fixed values are used for all model parameters based on the values given by Lin et al. [82] with the exception of $S_I$, $n_i$, $p_g$ and $EGP_b$. The conditional probability tables for these nodes are shown in Table 17.

$S_I$ is modelled as a truncated Gaussian distribution. The distribution has an upper limit of $6 \times 10^{-4}$ and a lower limit of 0. The value of $S_I$ depends on its value in the previous time slice. This is represented in the transition probability table as a Gaussian distribution whose mean is the node value in the previous slice.

Parameters $n_i$, $p_g$ and $EGP_b$ are modelled as uniform distributions. In the first time slice, a wide range of values are chosen from the initial state distribution. In each
subsequent slice, the value is based on the value previous slice. In this way, as evidence is incorporated, their values converge to patient specific values.

<table>
<thead>
<tr>
<th>Node</th>
<th>Initial State Model</th>
<th>Transition Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{t}$</td>
<td>$N(3 \times 10^{-4}, 2.5 \times 10^{-4})$</td>
<td>$(S_{t+1}, 1 \times 10^{-5})$</td>
</tr>
<tr>
<td>$n_{t}$</td>
<td>$U(1 \times 10^{-4}, 2 \times 10^{-2})$</td>
<td>$n_{t-1}$</td>
</tr>
<tr>
<td>$E GP_{b}$</td>
<td>$U(0, 3.5)$</td>
<td>$E GP_{b,t-1}$</td>
</tr>
<tr>
<td>$p_{b}$</td>
<td>$U(1 \times 10^{-3}, 0.1)$</td>
<td>$p_{b,t-1}$</td>
</tr>
</tbody>
</table>

Table 17: Conditional probability tables for model parameters in the ICING DBN.

Figure 50: DBN for the ICING model (11).

7.6.1 Results Using ICING DBN

Figure 51 shows a comparison of the results obtained using the ICU-MM DBN, the ICING DBN and the trivial model. Inference was preformed using the Adaptive-Time Particle Filtering algorithm and 100000 particles. For the Adaptive-Time Particle Filtering algorithm $\tau = 1$ and the summary interval is 15 minutes. The RMSEs are given in Table 18.
Figure 51: Comparison of the results obtained using the ICU-MM DBN, the ICING DBN and the trivial model.

Table 18: Comparison of RMSEs from each model for each patient

<table>
<thead>
<tr>
<th>Patient</th>
<th>ICU-MM DBN</th>
<th>ICING DBN</th>
<th>Trivial Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>P23</td>
<td>11.42%</td>
<td>10.15%</td>
<td>15.85%</td>
</tr>
<tr>
<td>P91</td>
<td>20.28%</td>
<td>22%</td>
<td>26.15%</td>
</tr>
<tr>
<td>P30</td>
<td>17.00%</td>
<td>16.74%</td>
<td>19.50%</td>
</tr>
<tr>
<td>P40</td>
<td>32.83%</td>
<td>36.54%</td>
<td>35.15%</td>
</tr>
<tr>
<td>P61</td>
<td>15.03%</td>
<td>25.81%</td>
<td>16.31%</td>
</tr>
<tr>
<td>P24</td>
<td>16.45%</td>
<td>19.91%</td>
<td>25.68%</td>
</tr>
<tr>
<td>P64</td>
<td>7.43%</td>
<td>16.56%</td>
<td>17.90%</td>
</tr>
<tr>
<td>P09</td>
<td>17.39%</td>
<td>18.99%</td>
<td>20.35%</td>
</tr>
<tr>
<td>P01</td>
<td>24.54%</td>
<td>9.36%</td>
<td>9.80%</td>
</tr>
<tr>
<td>P102</td>
<td>12.58%</td>
<td>19.11%</td>
<td>15.53%</td>
</tr>
<tr>
<td>P101</td>
<td>12.28%</td>
<td>9.19%</td>
<td>10.78%</td>
</tr>
<tr>
<td>P60</td>
<td>37.05%</td>
<td>44%</td>
<td>48.36%</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>18.69%</strong></td>
<td><strong>20.66%</strong></td>
<td><strong>21.78%</strong></td>
</tr>
</tbody>
</table>
Overall these results show similar prediction errors to those obtained with the DBN based on the ICU-MM. It could be that there is significant measurement error in our dataset, making it difficult to individualise the parameters for any model and difficult to accurately predict future behaviour.

The ICING results could possibly be improved with further work in the area of model refinement, validation and comparison. It would be beneficial implement the integral method for individualising model parameters recommended by the ICING authors and compare these results with the ICING DBN results on our patient data.

We also tested this ICING DBN with the additional patient data from our second dataset that included patients on enteral feeds. This data was more carefully collected and the average RMSE over all patients using the trivial model is lower than the first dataset. The results from the ICING DBN are compared to the trivial model in Figure 52 and Table 19. In 5 out of 7 patients the ICING DBN modestly out-performs the trivial model.

![Figure 52: The RMSE from the ICING DBN and the trivial model for patients on an enteral feed.](image)
Table 19: The RMSE from the ICING DBN and the trivial model for patients on an enteral feed.

### 7.7 Conclusions

In this chapter, we applied our new techniques to the problem of regulating glycaemia in intensive care patients. Critically ill patients have highly variable responses to the insulin that is administered to regulate glycaemia.

Firstly, we incorporated the ICU-MM model in our DBN framework. In Section 7.4.4, we showed that our technique produced better results on our patient data than the methodology used by the ICU-MM authors. However, when compared to the trivial model, in Section 7.4.6 the results were similar.

We then incorporated a second model, the ICING model, in our DBN. The results were not substantially different to the results obtained using the ICU-MM system of ODEs. These results may be limited by the measurement error in the dataset. We also tested the ICING DBN with data from additional patients who were in receipt of enteral feed. The average RMSE using this dataset was much lower than the first dataset.

It should be noted that for the purposes of assessing model performance and comparing it to other techniques, in this chapter we used a RMSE calculation based on the predicted mean plasma glucose level. The DBN, in fact, reports a range of
likely values as a mean and one standard deviation. A fairer metric may be to assess if the measured values lie within the predicted range i.e. one standard deviation. In Appendix 3 and 4, we show the predicted plasma glucose ranges for all patients along with the measured values. It can be seen that most of the time the measured values lie within one predicted standard deviation.

Our techniques have advantages when compared to other methodologies. As mentioned in Section 3.5, other approaches for modelling glycaemia use an offline methodology for individualising model parameters. The DBN framework proposed in this thesis offers a continuous online methodology. The DBN model adjusts parameters as soon as it receives the first piece of evidence, in contrast to methodologies that require a calibration window containing a number of observations. The DBN reacts to both gradual and sudden changes in model parameters as it tracks a range of possible trajectories. This is important in the critical care environment where both gradual and sudden changes occur. Finally, the DBN accounts for data uncertainty in a principled manner.
8. Conclusions

In this chapter, we discuss the benefits of the new methodologies introduced in this thesis, as well as their applicability, effectiveness and indeed limitations. We also discuss the challenges we encountered when applying our methodologies to a critical care problem with limited data and we discuss our framework for modelling glycaemia in critically ill patients.

8.1 Constructing DBNs from ODE Models

In this thesis, a new methodology for constructing DBNs by basing them on systems of ODEs has been presented. As was noted in Chapters 4 and 7, the advantages of encapsulating ODEs in DBNs are:

- The proposed DBN framework can be used for temporal data mining in noisy uncertain environments.
- The proposed framework handles both data uncertainty and model uncertainty in a principled manner.
- Real-time observations can be used to individualise the model parameters and align the DBN with reality.
- Both gradual and sudden changes in model parameters can be tracked.
- The DBN framework adjusts model parameters with the very first observation; a long learning window is not required.
- By automatically constructing the DBN structure from an existing mathematical model, the knowledge elicitation task is simplified.
- The formulation is widely applicable as ODEs are used extensively in the fields of engineering, medicine and science.
8.2 Performing Efficient Inference

For this new DBN framework to be of use in a real-time environment, efficient inference is necessary. This is especially an issue when the underlying ODE is stiff or there are periods of rapidly changing dynamics and periods where the dynamics are stable. In this thesis, different approaches have been explored to address this problem and here we discuss the advantages and disadvantages of each.

8.2.1 Technique 1: A Fixed Time Step DBN Incorporating a First-Order Euler Solver

This method is the simplest to implement. However, it has a number of disadvantages:

1. If the incorporated system is stiff, very small time steps are needed and the inference task can be inefficient.

2. Finding the appropriate time steps to use can be time consuming. In Section 4.5.4, we found that minimizing the number of steps between observations gives the best results. This means we must choose the largest step size possible while still ensuring the numerical solver yields a stable solution. This optimal time step length can be difficult to find, especially in stiff systems where small variations in the model parameters lead to even stiffer problems. In the example in Section 6.3.3, we had to incrementally test increasing numbers of step sizes before finding a stable solution. This is time-consuming and cumbersome.

3. Because fixed-length time steps are used, evidence must be approximated to the nearest time step. This can be an issue in very stable systems where large time steps could be used but these may not align with the evidence.

8.2.2 Technique 2: A Fixed Time Step DBN Incorporating an RK2 Solver

In a limited number of situations a DBN incorporating an RK2 solver can result in more efficient inference. By using an RK2 solver as opposed to an Euler solver, a smaller number of larger steps can sometimes be used, leading to the improved efficiency. This technique only proves effective in stiff problems where the model
parameters are well constrained and varying the parameter values, within the valid range, does not alter the model stiffness. This was shown in Section 5.2.2.2.

As was the case with Technique 1, finding the appropriate time steps to use can be time consuming and evidence has to be approximated to the nearest fixed time step. The other disadvantage is that the DBN structure is more complex with a larger number of nodes. More complex DBNs can be difficult to interpret and also the time required to run inference increases as the number of nodes increases.

8.2.3 **Technique 3: A DBN Incorporating an Euler Solver with Adaptive-Time Particle Filtering.**

As mentioned above, using our new methodology for incorporating an Euler solver in a DBN, results in a simple and straightforward mapping from the ODEs to a DBN framework. Using our new Adaptive-Time Particle Filtering algorithm on this DBN framework is significantly faster than using the standard fixed-time step particle filtering. This was shown in a theoretical example in Section 6.3 and in a medical example in Section 7.4.4. Adaptive-Time Particle Filtering can be used on stiff and non-stiff models, although the efficiency improvements are more obvious in stiff and/or fast-slow systems.

Another significant advantage is that it is easy to use, as the problem becomes more difficult, for example greater data error is introduced or more variability in the model parameters, the algorithm automatically adjusts the step size. This removes the requirement for the modeller to run numerous iterations to find the optimal step size.

Summary steps (see Section 6.1) are automatically aligned with the evidence removing the need to approximate evidence to the time step as with the fixed time step approaches.

8.2.4 **Technique 4: Full Steps and Sub-Steps**

In Section 6.4, we briefly discussed the option of using full steps and sub-steps. As mentioned, the advantage of this approach is that significant efficiency
improvements are gained by using sub-steps instead of full steps. The sub-steps can have a very small step size to accommodate very stiff problems.

However this technique shares its drawbacks with the other fixed step approaches, those being significant effort is required to select the appropriate full step length and sub-step length and the full step length must still match the granularity of the evidence time stamps.

8.3 The Challenges of the Intensive Care Domain

The ICU is a carefully controlled, data-rich stochastic environment making it ideally suited for exploiting data mining techniques. However it is not without its challenges.

For this project, some of the data used was historical data from the ICU in University Hospital Galway. As with any ICU, patient care is the priority, not data collection. Therefore, from a research perspective, data that is routinely collected in a busy ICU suffers for certain drawbacks when compared to data collected during a specific clinical study. Data may not be gathered at an optimal frequency. Data may be missing, for example, whether medication administered was suspended in a glucose solution. Time stamps may be inaccurate. We observed that some timestamps were set to the nearest hour thereby introducing a large degree of temporal uncertainty.

Critically ill patients are highly variable in their response to care and treatment, making learning population distributions difficult, particularly when sample sizes are small.

Another interesting observation made in during discussions with our clinical collaborators (Professor John Laffey, Dr. Brian Harte and Ms. Anne Mulvey) was that because the patient condition is actively managed by medication, subsets of the data may not reflect a patient’s underlying condition. For example, we were looking for further markers to indicate a patient’s condition was worsening and therefore perhaps their insulin resistance changing. An obvious marker is temperature; an increase in temperature above the normal temperature can indicate deterioration in a
patient’s condition. However, the data rarely reflects an increase in temperature because temperature is controlled and managed in an ICU. Medication is administered to reduce temperature, so the temperature reading on its own may be misleading. It is interesting that in a study by Saria et al. [111] they found that physiological data from the first 3 hours of life could successfully predict illness severity or long-term morbidity of pre-term infants. This time frame was chosen as it is less likely to be confounded by medical interventions. So although the ICU provides a data-rich environment, the data may not reflect the true picture. The management and control of patient conditions make using ICU data for predicting future responses non-trivial.

8.4 The DBN Framework for Modelling Glycaemia

In Chapter 7, a DBN for modelling glycaemia in ICU patients was presented. A model such as this could be used in a number of ways in a clinical setting:

1. To track a patient’s glycaemia levels.
2. To simulate an individual patient’s response to different dosage regimes.
3. To recommend a change in insulin dosage.
4. To suggest to staff that a new glucose measurement should be taken as the model uncertainty has grown beyond an acceptable limit.
5. As an alarm to indicate that the patient is not responding as expected and perhaps their condition has changed.

The first DBN in Chapter 7 incorporated the ICU-MM ODE model. Although using the DBN methodology out-performed the technique used by the ICU-MM authors the results are not substantially better than the results obtained from the trivial model. We also incorporated the ICING ODE model in our DBN framework. The advantage of the ICING ODE model is that it includes terms to model enteral feeding and has fewer parameters than the ICU-MM that require individualisation. The results are not substantially different from the results obtained with the ICU-MM.
The advantage of our DBN framework is that it offers an online methodology for continuously re-estimating model parameters. It adjusts parameters as soon as it receives the first piece of evidence and reacts to both gradual and sudden changes in model parameters as it tracks a range of possible trajectories. This is important as critically ill patients have a highly variable response to administered insulin and this response can change over time as a result of changes in the patient’s condition or other administered medication. Finally, the DBN accounts for data uncertainty and model uncertainty in a principled manner.

8.5 Future Work

While the methodology described in this thesis concerns building the DBN structure using an existing body of knowledge in the form of ODEs, future work could exploit the knowledge in the ICU database, to learn the conditional probability tables. By combining the knowledge available in the data with the expert knowledge available in the form of differential equations, we believe we will have a powerful tool for reasoning with uncertain and sparse data.

Further work needs to be done on both the ICU-MM DBN and the ICING DBN. It would be beneficial to expand the DBNs to incorporate other factors that are known to influence glycaemia in the critically ill and so make use of all the available information. Steroids, which are commonly administered to septic patients in the ICU, are known to increase glycaemia [41]. More critically ill patients tend to be more insulin resistant and have more variable glucose dynamics [86] – the APACHE II score and lactate levels [112] are good indicators of severity of illness. The cause of the illness should also be considered, for example patients with sepsis are more likely to experience hyper-glycaemia than patients follow cardiac surgery [41]. The DBNs could be expanded by introducing nodes to represent these factors. These new nodes would influence the initial state distributions for the nodes representing insulin sensitivity.
Substantial effort would be required to put a system, such as the one envisaged in this thesis, in practice by the bedside. This effort would lie in the areas of interworking with existing databases, designing an effective method for displaying predictions and glucose trajectories, reliability testing and clinical trials.

8.6 Conclusions
The motivating question for this thesis is how to efficiently track and predict dynamic variables in a stochastic environment with only sparse real-time evidence. In this thesis, a Dynamic Bayesian Network that incorporates mathematical models is proposed as a solution.

Much knowledge of human physiology is formalised as systems of differential equations. The first contribution of this thesis is a methodology for automatically incorporating systems of ordinary differential equations in Dynamic Bayesian Networks. Expert knowledge in the form of ODEs is widely available in text books and articles. By exploiting this readily available knowledge we can significantly reduce the knowledge engineering effort required to build DBNs. The proposed DBNs provide a natural framework for handling data uncertainty and model uncertainty. Multiple strands of sparse, noisy temporal evidence are used to calibrate model parameters to the individual and track both gradual and sudden changes in the parameters over time.

A second contribution of this thesis is an expansion of this DBN framework to incorporate a second-order solver which can be of benefit when the underlying ODE model is stiff.

A third contribution is a new Adaptive-Time Particle Filtering algorithm that uses time steps that adapt according to the dynamics at each time step. Each particle follows its own adaptive-time scheme leading to advantages in both efficiency and accuracy. In fast-slow systems, there are periods where very small steps must be used to capture rapidly changing dynamics, but large steps are more suitable during
Conclusions

periods of slow change. No one step size is optimal. In examples where the underlying ODE system is stiff and fixed step sizes are being used, they must be very small. This quickly becomes inefficient even when using solvers of orders higher than one. Our Adaptive-Time Particle Filtering algorithm addresses these inefficiencies by allowing the time step size to adapt automatically. Another advantage of the adaptive approach is that time steps are automatically aligned with the exact time stamp of the evidence. In traditional fixed time step approaches, evidence must be approximated to the nearest fixed step, introducing further uncertainty.

In this thesis, both the methodology for incorporating ODEs into DBNs and the efficiency improvements to be gained by using our new Adaptive-Time Particle Filtering approach were examined on both theoretical models and a real-world critical care medical model.

The final contribution of this thesis is the application of the new methodologies to predicting a critically ill patient's plasma glucose levels in response to insulin and glucose infusions. With the data available, which is sporadic and may be inaccurate and incomplete, the DBN approach out-performs a previous ODE-based approach demonstrating that the DBN method is effective at re-estimating model parameters and reasoning with sparse and potentially unreliable data.

This thesis addresses an issue of great importance in drug delivery, that of personalising the drug dosage to the individual patient. While here insulin is investigated, the framework has the potential to be used to individualise any drug dosage or to be incorporated into a control system to automate drug delivery.

Not only is the approach presented to Bayesian time-series data mining applicable to drug delivery, it is also applicable to many engineering, medical and scientific problems, where sparse and noisy temporal data must be analysed and domain knowledge in the form of mathematical models is available.
References


References


References


References


Appendix 1

In Section 6.2, we presented our algorithm for controlling the step size. The values $q$, $M_1$ and $M_2$ affect the step size recommendations. The following tables show the sensitivity of the algorithm to these values. These tests were carried out on Scenario 1 for the van der Pol DBN (Section 6.3.2). For all tests the tolerance, $\tau = 0.01$.

In Table 20, $M_1$ and $M_2$ are fixed, and various values of $q$ are tested. The value $q$ is used to scale our estimate of the appropriate step size, as overestimating the step size would result in the subsequent step being rejected. In Table 20, we can see that when $q = 1$, and therefore our estimate is not scaled, the number of rejections is 115. In these cases, we have overestimated the step size. We avoid this by using $q < 1$. As can be seen for values of $q < 0.5$ we have very few steps rejected, as the recommended step size is much smaller than our estimate. However, at these low values of $q$, the step size is prevented from growing when appropriate and so we lose the benefit of the adaptive approach. Values satisfying $0.6 \leq q \leq 0.9$ work best.

In Table 21, $q$ and $M_2$ are fixed, and various values of $M_1$ are tested. When the solution switches from a region in which its derivative changes rapidly to a region in which its derivative changes slowly, the algorithm will increase the step size. To control the increase the algorithm selects the minimum of the estimated step size and $M_1 h_i$. $M_1$ is therefore the maximum factor by which the step size will be increased. As can be seen in Table 21, for the van der Pol DBN once the $M_1 > 1.5$, the results are similar as in these cases the algorithm selects the lower estimated step size.
<table>
<thead>
<tr>
<th>q</th>
<th>$M_1$</th>
<th>$M_2$</th>
<th>No. Steps Accepted</th>
<th>No. Steps Rejected</th>
<th>Total No. Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>1.5</td>
<td>0.9</td>
<td>2238</td>
<td>1</td>
<td>2239</td>
</tr>
<tr>
<td>0.2</td>
<td>1.5</td>
<td>0.9</td>
<td>1136</td>
<td>1</td>
<td>1137</td>
</tr>
<tr>
<td>0.3</td>
<td>1.5</td>
<td>0.9</td>
<td>767</td>
<td>1</td>
<td>768</td>
</tr>
<tr>
<td>0.4</td>
<td>1.5</td>
<td>0.9</td>
<td>603</td>
<td>1</td>
<td>604</td>
</tr>
<tr>
<td>0.5</td>
<td>1.5</td>
<td>0.9</td>
<td>490</td>
<td>4</td>
<td>494</td>
</tr>
<tr>
<td>0.6</td>
<td>1.5</td>
<td>0.9</td>
<td>408</td>
<td>9</td>
<td>417</td>
</tr>
<tr>
<td>0.7</td>
<td>1.5</td>
<td>0.9</td>
<td>361</td>
<td>12</td>
<td>373</td>
</tr>
<tr>
<td>0.8</td>
<td>1.5</td>
<td>0.9</td>
<td>335</td>
<td>31</td>
<td>366</td>
</tr>
<tr>
<td>0.9</td>
<td>1.5</td>
<td>0.9</td>
<td>319</td>
<td>76</td>
<td>395</td>
</tr>
<tr>
<td>1</td>
<td>1.5</td>
<td>0.9</td>
<td>305</td>
<td>115</td>
<td>420</td>
</tr>
</tbody>
</table>

Table 20: The number of steps executed for various values of $q$.

<table>
<thead>
<tr>
<th>$M_1$</th>
<th>q</th>
<th>$M_2$</th>
<th>No. Steps Accepted</th>
<th>No. Steps Rejected</th>
<th>Total No. Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>0.9</td>
<td>0.9</td>
<td>346</td>
<td>71</td>
<td>417</td>
</tr>
<tr>
<td>1.5</td>
<td>0.9</td>
<td>0.9</td>
<td>319</td>
<td>76</td>
<td>395</td>
</tr>
<tr>
<td>2</td>
<td>0.9</td>
<td>0.9</td>
<td>312</td>
<td>77</td>
<td>389</td>
</tr>
<tr>
<td>2.5</td>
<td>0.9</td>
<td>0.9</td>
<td>314</td>
<td>80</td>
<td>394</td>
</tr>
<tr>
<td>3</td>
<td>0.9</td>
<td>0.9</td>
<td>314</td>
<td>81</td>
<td>395</td>
</tr>
<tr>
<td>3.5</td>
<td>0.9</td>
<td>0.9</td>
<td>314</td>
<td>81</td>
<td>395</td>
</tr>
</tbody>
</table>

Table 21: The number of steps executed for various values of $M_1$. 

135
In Table 22, $q$ and $M_1$ are fixed and various values of $M_2$ are tested. To ensure a reduced step size is always recommended when the tolerance is exceeded, we choose $M_2 < 1$. As can be seen in Table 22, for the van der Pol DBN, the algorithm is reliable for any value of $M_2 < 1$ but $M_2 = 0.9$ works best.

<table>
<thead>
<tr>
<th>$M_2$</th>
<th>$q$</th>
<th>$M_1$</th>
<th>No. Steps Accepted</th>
<th>No. Steps Rejected</th>
<th>Total No. Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.9</td>
<td>1.5</td>
<td>466</td>
<td>46</td>
<td>512</td>
</tr>
<tr>
<td>0.2</td>
<td>0.9</td>
<td>1.5</td>
<td>400</td>
<td>45</td>
<td>445</td>
</tr>
<tr>
<td>0.3</td>
<td>0.9</td>
<td>1.5</td>
<td>370</td>
<td>50</td>
<td>420</td>
</tr>
<tr>
<td>0.4</td>
<td>0.9</td>
<td>1.5</td>
<td>353</td>
<td>59</td>
<td>412</td>
</tr>
<tr>
<td>0.5</td>
<td>0.9</td>
<td>1.5</td>
<td>339</td>
<td>61</td>
<td>400</td>
</tr>
<tr>
<td>0.6</td>
<td>0.9</td>
<td>1.5</td>
<td>334</td>
<td>81</td>
<td>415</td>
</tr>
<tr>
<td>0.7</td>
<td>0.9</td>
<td>1.5</td>
<td>325</td>
<td>77</td>
<td>402</td>
</tr>
<tr>
<td>0.8</td>
<td>0.9</td>
<td>1.5</td>
<td>320</td>
<td>77</td>
<td>397</td>
</tr>
<tr>
<td>0.9</td>
<td>0.9</td>
<td>1.5</td>
<td>319</td>
<td>76</td>
<td>395</td>
</tr>
</tbody>
</table>

Table 22: The number of steps executed for various values of $M_2$. 
Appendix 2

The following graphs show the values inferred by the ICU-MM DBN for all model variables and model parameters as they change over time for Patient 30. These are the results using the Adaptive-Time Particle Filter.
Appendix 2

- $V_C$

- Basal Glucose

- $P_1$

- $P_2$
Appendix 3

Following are the graphs showing the glucose and insulin infusion and the plasma glucose predictions made by the ICU-MM DBN using Adaptive-Time Particle Filtering.

Patient 23
Patient 91

Glucose and Insulin Infusion Rates for Patient 91

Glucose Levels for Patient 91
Patient 40
Patient 61

Glucose and Insulin Infusion Rates for Patient 61

Glucose Levels for Patient 61
Patient 24

Glucose and Insulin Infusion Rates for Patient 24

Glucose Levels for Patient 24
Patient 64
Patient 09

Glucose and Insulin Infusion Rates for Patient 09

Glucose Levels for Patient 09
Patient 01

![Graph showing glucose and insulin infusion rates for Patient 01.](image)

![Graph showing glucose levels for Patient 01.](image)
Patient 102

[Graphs showing glucose and insulin infusion rates and glucose levels over time for Patient 102]
Patient 101

**Glucose and Insulin Infusion Rates for Patient 101**

**Glucose Levels for Patient 101**
Patient 60

**Glucose and Insulin Infusion Rates for Patient 60**

**Glucose Levels for Patient 60**
Appendix 4

The ICU-MM does not account for enteral feeding, therefore for those patients in our second dataset (see Section 7.4.1) the ICING DBN was used to make plasma glucose predictions. Following are the graphs showing the enteral glucose, the glucose and insulin intravenous infusion and the plasma glucose predictions made by the ICING DBN using Adaptive-Time Particle Filtering.

Patient 103
Patient 104
Patient 105
Patient 106
Patient 107

Prescribed Glucose and Insulin for Patient 107

Glucose Levels for Patient 107
Patient 108

Prescribed Glucose and Insulin for Patient 108

Glucose Levels for Patient 108
Patient 109

(mg/hr) Prescribed Glucose and Insulin for Patient 109

(mg/dl) Glucose Levels for Patient 109