Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 177 (2020) 354–362

Procedia Computer Science

www.elsevier.com/locate/procedia

The 10th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH 2020)
November 2-5, 2020, Madeira, Portugal

SmartCGMS as a Testbed for a Blood-Glucose Level Prediction and/or Control Challenge with (an FDA-Accepted) Diabetic Patient Simulation

Tomas Koutny[a]*, Martin Ubl[b]

[a]*NTIS – New Technologies for Information Society, University of West Bohemia, Technicka 8, 306 14 Plzen, Czech Republic*
[b]*Department of Computer Science and Engineering, University of West Bohemia, Technicka 8, 306 14 Plzen, Czech Republic*

Abstract

Diabetic patient desires to avoid hypo- and hyperglycemic episodes, which result from insufficient insulin production. As the diabetes disease progresses, it requires an advance control of external insulin administration with an insulin pump. Given the importance of blood-glucose level prediction for the insulin therapy, there is a Blood-Glucose Level Prediction Challenge. This prediction is based on a post-mortem dataset, which include a number of signals related to the daily life of a diabetic patient. We propose replacing these post-mortem signals with an *in-silico* diabetic patient. For this purpose, we can use the SmartCGMS continuous glucose monitoring and controlling framework together with an FDA-accepted diabetic patient simulation. As a result, a competing researcher have the same conditions as a developer of a real-life insulin pump, connected to a real diabetic patient. When using SmartCGMS, simulated, prototyped and real devices can work together. This approach reduces the difference between laboratory and practical results, thus increasing the level of realism for the entire challenge. As a report on the current SmartCGMS state, we describe the previously unpublished features, which enable an improved glucose level prediction and/or control challenge.

Keywords: diabetes; prediction; control; smartcgms

* Corresponding author. Tel.: +420 377 632 437; fax: +420 377 632 402.
E-mail address: txkoutny@kiv.zcu.cz

10.1016/j.procs.2020.10.048

1. Introduction

Diabetes is a silent civilization disease. It manifests with an elevated blood glucose level (BG) [13]. Either absolute or relative insulin insufficiency causes elevated BG. Type 1 diabetic (T1D) patient suffers from absolute insulin insufficiency due to near-to-zero insulin production by the pancreatic cells. Type 2 diabetic (T2D) patient suffers from relative insulin insufficiency, because cells need more insulin to produce the required amount of energy [25].

Elevated BG continuously damages multiple internal organs [11]. It leads to both chronic and acute complications. Insulin lowers BG by promoting its utilization. Therefore, diabetic patients use insulin pump to inject the insulin from an external reservoir. While the insulin pump is common with T1D patients, it may benefit T2D patients as well [4].

As the disease progresses, BG control becomes more complex and demanding. Inefficient BG control deteriorates patient's condition and quality of life. A sophisticated computer algorithm may be a solution, when directly controlling the insulin pump [9]. Such an insulin pump would be augmented with a sensor of continuous glucose monitoring system (CGMS). Possibly, an advanced insulin-pump controller can connect to other sensors as well.

Let us consider an insulin pump, which delivers the insulin at a constant basal rate (IBR). Let this pump be capable of suspending the insulin delivery (LGS aka low-glucose suspend), if BG falls below a given threshold [27]. By predicting BG instead of sensing the current level, we can further reduce the number of hypoglycemic episodes by possibly adjusting IBR [5, 6]. We can predict BG based on past BG levels, and optionally past levels of other signals such as acceleration, body temperature, etc. [8, 12].

There is Blood Glucose Level Prediction Challenge. It focuses directly on BG prediction, using multiple signals. In its present form, a competing researcher obtains OhioT1DM dataset [24]. This is a training dataset, designed for a development of a BG prediction method. Then, organizers of the challenge evaluate the prediction method using validation dataset. Using root mean square error (RMSE), the prediction methods compete in 30- and 60-minutes prediction horizons.

With the present challenge setup, we see two opportunities to increase its level of realism:

1. Now, the researcher obtains a post-mortem database of measured signals as a whole. In reality, sensors provide new measurements incrementally. We offer a framework that can replay the post-mortem databases as if they come from real sensors. The framework would isolate the competing researcher from the database.
2. Due to the use of post-mortem databases, researchers cannot compete in BG control. We offer a framework that provides in-silico patient that reacts to changes in insulin dosage – i.e., it responds dynamically to insulin boluses and IBR changes.

We propose to use SmartCGMS – continuous glucose monitoring and controlling framework. SmartCGMS abstracts the chain of devices (from a sensor to an insulin pump) with a set of filters [20]. Each filter represents a single step in the entire processing of measured physiological signals. As SmartCGMS makes no assumption about a particular filter implementation, real, prototyped and simulated devices can work together to reduce differences between laboratory and practical conditions.

Second section describes individual features of SmartCGMS, while the following section specializes on programming with SmartCGMS. Fourth section presents an experimental setup of a predictive low-glucose management system. Fifth section concludes the paper and outlines future work.

2. SmartCGMS

SmartCGMS is directly available as x86-64 build for MS Windows, macOS and Debian GNU/Linux, as well as armeabi-v7a and arm64-v8a builds for Android. Other systems are available upon a request. For non-profit, academic research, SmartCGMS is available under the GPLv3 license. Other use requires a different license.

SmartCGMS ships with two executables (x86-64 only) and a set of dynamically loaded libraries (SmartCGMS itself, core libraries, available on all platforms). One executable, gpredict3, offers a graphical user interface (GUI) to visualize, edit and execute custom configurations of the signal processing. The second executable, console3, executes the configuration only.

2.1. Reading the dataset

Let us connect three filters – file reader, drawing and log. The file reader would read the given dataset, while the drawing filter would visualize respective graphs. To verify that such a setup works correctly, the third filter records all events to an external, comma-separated value (CSV) log file. For example, to load the OhioT1DM dataset, we select respective .xml file of a particular patient and set the segment spacing to 1200 seconds. The segment spacing determines the gap, which separates individual segments of measuring.

To replay post-mortem measurements, SmartCGMS can replay the given dataset, SQL-compliant database, log file and a number of CGMS profiles as exported by authorized software of Medtronic, Dexcom, etc. – a functionality inherited from the preceding project on on-line BG calculation [19].

2.2. Closed-Loop

To improve the degree of realism, let us replace the post-mortem signal-replay with an in-silico patient. SmartCGMS ships with three in-silico models. (1) Bergman model [28] combined with Hovorka's absorption model [10] and Koutny's diffusion model, which calculates the interstitial fluid glucose level (IG) [15, 16], UVA/Padova (2) S2013[31] and (3) S2017 [32] models. Researcher can develop another model as needed. With a valid license, SmartCGMS works with FDA-approved diabetic-patient simulators T1DMS and DMMS.R [29]. FDA stands for U.S. Food & Drug Administration.

With a patient, we need to deliver insulin. SmartCGMS filter calculates insulin bolus or IBR, based on measured signals. Then, it uses a feedback connection to deliver requested insulin boluses and IBR requests to the insulin pump. The in-silico patient contains a simulated insulin pump that reacts to these requests. The fifth section describes, executes and discuss such an experimental setup.

With a real patient, we would replace only the in-silico patient filter with a CGMS-sensor reading filter and a filter that controls real insulin pump. Other filters do not change.

2.3. Calculating Time-in-Range

To quantify effectiveness of BG control, let us calculate a time in range [2]. It is a time that the patient spent between given low and high BG thresholds, during a given period. SmartCGMS ships with several filters, which can produce different statistical markers. One of them is a decoupling filter. This filter evaluates a logical formula. If it evaluates to true, the filter changes signal id of the particular device event (a unit of communication between filters). In addition, it updates statistics of these device events. Eventually, the filter saves the statistics to an external CSV file. Hence, we can use it to gather statistics about hypo- and hyperglycemic episodes, time-in-range, etc. Fig. 1 depicts configuration of the decoupling filter to collect information about the time in range for T1D and T2D patients [2]. Setting the output signal to the *Null signal* causes the filter to update the statistical markers only.

2.4. Metric

When evaluating different competing filters, we need a scalar-represented score. SmartCGMS implements a filter, which calculates a chosen metric such as average absolute error, standard deviation, area under the curve, Crosswalk [18], etc. It takes times and levels of a reference signal. At these times, it calculates difference between the reference-signal levels and calculated levels. For example, predicted or controlled BG can be the calculated signal, while measured or target BG would be the reference one. Eventually, the metric reduces the resulting set of difference to a single scalar.

As needed, the metric filter can periodically output the metric scalar as another signal so that we can observe its time course. We used this approach e.g. with the Icarus game [30]. Fig. 1 depicts respective metric-filter configuration. Less metric value means better fitness. Hence, the Icarus-game frontend further recalculates the score signal to be monotonically increasing.

By emitting the metric-value as a signal, while possibly remapping particular signals, we can evaluate multiple metrics, thus exerting a multi-objective optimization. When remapping, a dedicated filter rewrites signal id to a different one. For example, it can change calculated-BG id to measured-BG id.

2.5. Batch Processing

To evaluate multiple competing filters (e.g. BG predicting filter, or insulin dosing filter), there would be a need to use multiple input files and to produce output files with custom names to identify a particular competing filter and/or an evaluation scenario. For such a purpose, it is possible to create a single SmartCGMS configuration, which will read actual parameters from system variables. Then, a batch file can execute SmartCGMS to produce desired results with a constant configuration for all competing filters.

Fig. 2 depicts the batch-processing configuration for a log replay.

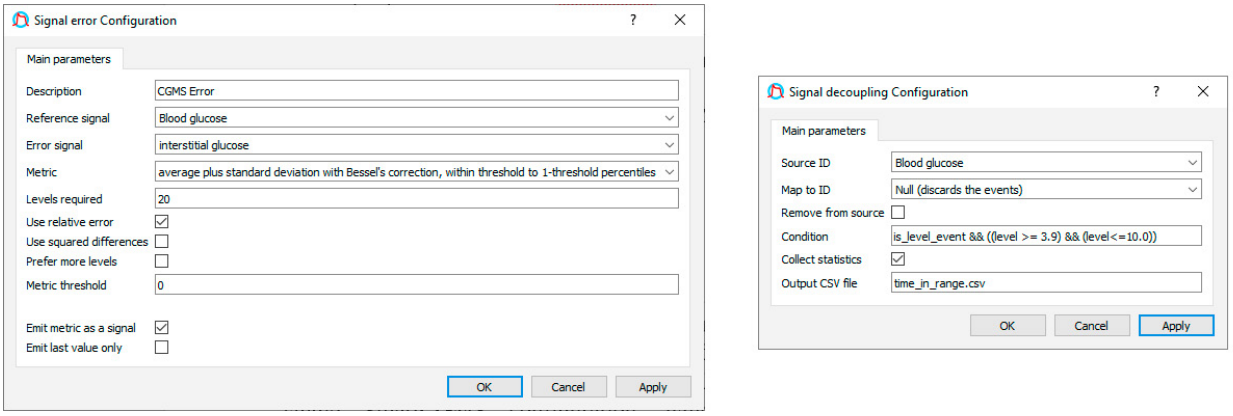

Fig. 1. Configuring Error Metric filter (left) and Signal Decoupling to obtain Time-In-Range Statistics (right).

```
@for /R %%G in (*.log) do @call :SCGMS "%%G"
@exit /B 0


:SCGMS
@set Input_File=%*
@echo Processing a log: %Input_File%

@rem console3 runs SmartCGMS processing without GUI
@rem The first console3 parameter is optional,
@rem and it is specific configuration file.
console3 config.ini

@exit /B 0
```

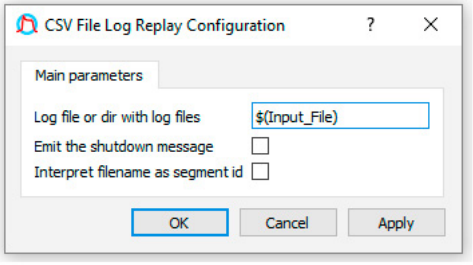

Fig. 2. Batch Processing with SmartCGMS

2.6. Decomposition

For example, IBR calculation may require additional signals such as insulin-on-board (IOB) and carbohydrates-on-board (COB) [26]. Although IBR-calculating filter can produce them internally, it would considerably increase its logic. Therefore, SmartCGMS allows extraction of e.g. IOB and COB-calculation to separate filters. These filters produce respective IOB and COB signals, possibly utilized by the IBR-calculating filter. As a result, we support multiple IOB and COB models, e.g. bilinear and exponential ones.

Such an approach reduces maintenance costs of individual filters, and improves a verification process. As we are concerned about processing medical signals, we are concerned about verification of correct functions of individual SmartCGMS components, e.g. the filters.

2.7. Masking – Training and Testing Datasets

When testing a particular calculation, there is a need for training and testing datasets. One possibility is to create different scenarios – one for training and one for testing. Depending on the needs, another possibility is to use masking. SmartCGMS implements a masking filter, for which a researcher selects particular signal and enter a binary mask. Bits in this mask identify whether to keep a level of the given signal unmodified, or whether to mask it. For example, we can omit every third and fourth measured level of the CGMS sensor to evaluate how much it would degrade BG prediction or IBR calculation.

2.8. Solvers – Parameter Identification

With SmartCGMS, we decouple a physiological model implementation from an identification of its parameters. To SmartCGMS, an array of floating-point numbers represents the model parameters. Then, the model implementation calculates particular signals only, by read-only accessing the parameters array. To identify the parameters, we offer a number of solvers, which we expose with a uniform interface. Then, a signal-calculation filter can automatically apply these solvers to determine model parameters progressively. In this process, the solver uses the metric as described in Section 2.4.

By default, SmartCGMS ships with Meta-differential evolution [17] and Pathfinder – a deterministic evolutionary method of our own design. On request, we can provide a custom build that supports the NLOpt [14] and PaGMO2 [3] implemented algorithms. Alternatively, a researcher can develop a specialized solver for a particular model.

3. Programming with SmartCGMS

A SmartCGMS front-end (e.g.; the gpredict3 and console3 executables) loads the scgms (SmartCGMS) dynamic library. This library loads other dynamic libraries, which implement individual filters and models. A researcher competing in the challenge would simply create a set of custom filters or models. For example, one filter can predict BG, while another one can calculate IBR. The latter filter can reuse the calculation of the preceding one.

The filters interoperate using a message called device event. Device event contains event code (e.g.; measured level), time segment id, time stamp, signal's id and signal level [20]. Time segment represents a continuous period of measuring, e.g. a lifetime of CGMS sensor from its insertion to its removal. On receiving a device event, a filter can output none, single or multiple device events. As needed, a filter can output device events without waiting for an input event. For example, the in-silico patient filter or CGMS-sensor filter do so. SmartCGMS ships with example projects, which provide respective source code.

3.1. Programming a Filter

A straightforward way to process the signals is to develop a custom filter. SmartCGMS components builds on the concept of Component Object Model (COM) [23]. Each component inherits from the same interface refcnt::IReferenced. This interface provides methods to increment and decrement reference counter of the object, together with a method to query the interface for another interface. Globally Unique ID (GUID) [22] identifies each interface. The component releases itself, once its reference counter decreases to zero. Hence, it uses the correct memory manager so that multiple programming languages and runtimes can be used together.

To implement the filter, the containing dynamic library must export C functions *do_create_filter* and *do_get_filter_descriptors*. The latter one returns static structures only.

The generic filter implements two methods only – *Configure* and *Execute*. The *Configure* method has one parameter. It is another component, which allows the filter to obtain the configuration based on the key-value paradigm.

SmartCGMS calls the *Execute* method, whenever a preceding filter produced an event. Hence, the component can either discard this event, or produce one or more events, while performing a custom calculation. It is a simple

and straightforward process. As a result, it provides no specialization. Signal and discrete models are specialized filters, which further reduce the amount of source code needed to implement a model.

3.2. Signal Model

The signal model interface inherits from a signal interface. The signal interface defines access to measured levels and their interpolation, using the Akima spline. As time segment contains multiple signals, it implements a time segment interface, which enumerates and makes individual signals accessible to other, time-segment contained signals. For example, the diffusion model calculates with BG and IG to produce future IG – thus accessing the BG and IG signals in the same time segment.

Let us predict BG by calculating weighted average of 10 recent BG levels. Hence, the model would have 11 parameters. Ten parameters would express the individual weights, while the 11th parameter would express the prediction horizon.

With the filter implementation, we would periodically update a buffer of 10 floating-point numbers – the recent BGs. With each update, we would calculate the weighted average and emit the result as the predicted BG. This is a stateful approach. On contrary, signal model presents a stateless approach.

With the signal model, the developer must implement a method called *Get_Continuous_Levels*. Its arguments are model parameters, vector of times at which to calculate the levels and desired derivative order of the calculated levels. When implementing this method, we would predict BG for a particular time, after requesting 10 preceding BG levels via the time segment container, prior the particular time less the prediction horizon. Hence, we can calculate prediction for any past time. Such a functionality may be needed under certain circumstances.

SmartCGMS provides a calculation filter, which calls the signal models to produce calculated signals. In addition, it enables automatic solving of their parameters as new data are incrementally measured – e.g.; as CGMS sensors provides new readings.

3.3. Discrete Model

Let us consider an in-silico patient model. When implementing such a model, we need two methods – one to initialize its state and the second to advance its simulation time, while emitting signals such as BG. The in-silico patient has to respond to carbohydrates and insulin, both represented with device events.

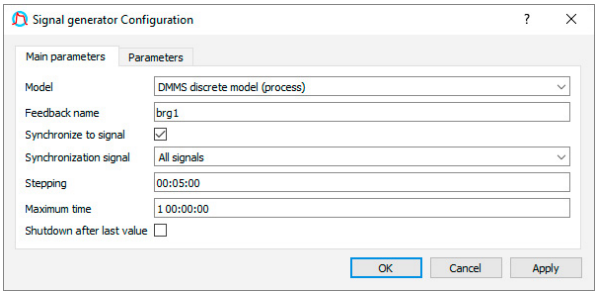

Fig. 3. SmartCGMS connects with FDA-accepted DMMS.R.

There are two possible configurations. First, the discrete model executes in a dedicated thread, thus producing device events asynchronously to its input. This is the case with the Icarus game. Second, the discrete model is synchronized to another signal, produced by a preceding filter. For example, this is the case with FDA-accepted scenarios for testing IBR calculation. In such a scenario, meals and insulin boluses are delivered exactly at scheduled times of the in-silico experiment.

Implementing the support for both, synchronous and asynchronous, configurations would be demanding and repeating task across different discrete models. Therefore, SmartCGMS provides Signal Generator filter. This filter executes discrete models in both configurations.

Fig. 3 depicts configuration of the Signal Generator filter. Particularly, it depicts a connection between SmartCGMS and the recent FDA-accepted DMMS.R T1D simulator.

4. Experimental Setup of a Predictive Low-Glucose Management System

To demonstrate the capability of predicting and controlling BG levels, we configured SmartCGMS to act as a Predictive Low-Glucose Management (PLGM) system [1]. PLGM insulin pump suspends insulin infusion for a pre-defined period of time, when PLGM predicts that BG would fall below a given threshold.

To demonstrate the PLGM functionality, we adopted FDA-accepted scenario. This scenario describes times and amounts of ingested CHO and associated insulin boluses. Specifically, we followed study [21]. We fed these events to our modification of the Bergman model, using the log-replay filter. The log-replay filter clocks the Bergman model as both filters execute synchronously. Then, the model computes BG and IG.

We chose our modification of the Bergman model for this experimental setup, because SmartCGMS delivers it free for academic research. Switching to FDA-accepted T1D simulator is as easy to accomplish as to change a single line in the SmartCGMS configuration file or selecting a different model in a GUI combo box.

Using the Steil-Rebrin (SR) model [7], we calculate SR-BG from Bergman-IG. Then, we use the diffusion model [15] to predict IG 30-minutes ahead. The signal filter automatically solves the prediction parameters, as new IG levels become available.

To suspend insulin infusion, we use the LGS insulin pump. This pump uses IG as the reference signal. Therefore, we use the diffusion model to transform the future IG to the present IG, while calculating a temporal BG signal. After these steps, the LGS acts as PLGM, because it actually calculates with predicted IG. Let us note that the LGS developer did not need to develop a complex PLGM logic, due to the signal remapping.

Fig. 4 depicts SmartCGMS filter-sequence needed to build the PLGM system. Fig. 5 depicts calculated signals as displayed in the SmartCGMS GUI. The red curve depicts patient's BG, while the blue one depicts IG. A green circle depicts CHO intake, while a purple dot represents insulin bolus. The turquoise steps indicate IBR. The green curve is predicted IG. The yellow curve is the signal consumed by LGS to produce the PLGM behavior.

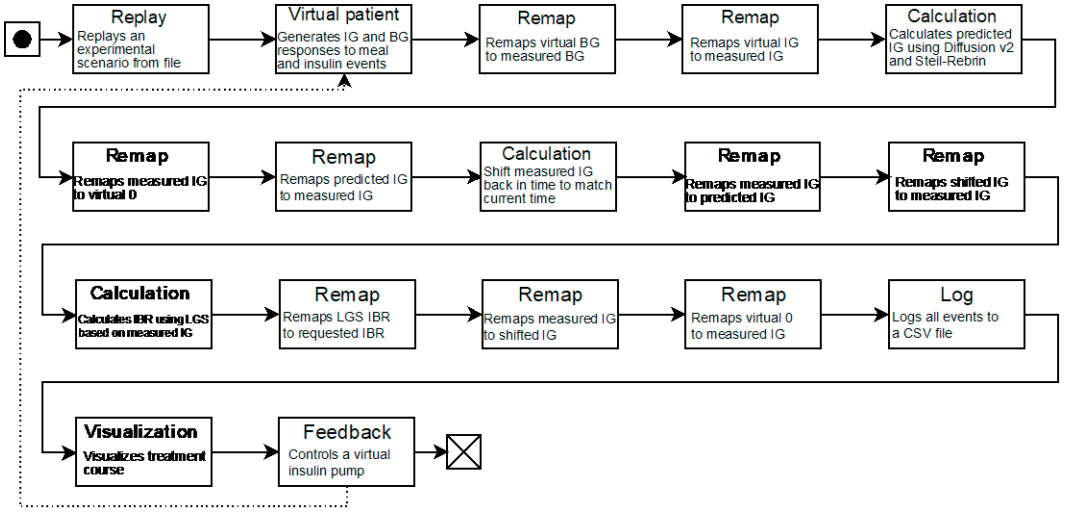

Fig. 4. SmartCGMS configuration to obtain the Predictive Low-Glucose Management system behavior.

Fig. 5. Visualization of Predictive Low-Glucose Management system built with SmartCGMS.

5. Conclusion and Future Work

In this paper, we demonstrated main features of the SmartCGMS framework, which are relevant to design a BG prediction and/or control challenge with an increased level of realism. Using these features, we demonstrated SmartCGMS capability to predict BG as well as to control it. With SmartCGMS, competing researchers can implement custom models, while SmartCGMS provides the infrastructure for automated testing and evaluation of the submitted glucose-level prediction and/or control models. Hence, the competition organizers would be relived from the burden of assuring that each model is fairly compared to another model across several reviewers and the reviewers can focus just on the methodology.

In the future work, we will focus on implementing the community feedback as received by fellow researchers and improving the frontend. Particularly, we focus on designing a new frontend for mobile phones as SmartCGMS compiles and executes on low-power hardware as well.

To accelerate SmartCGMS-connected research, we focus on interoperability with different languages. A number of languages support COM natively. Where COM is not available, SmartCGMS can invoke an external, discrete model using a network protocol. Currently, this protocol enters the beta-testing phase. To facilitate SmartCGMS invocation from different languages, SmartCGMS implements ISO/IEC 9899:2018 (C18) compliant interface.

Finally, let us stress that SmartCGMS architecture is independent on the diabetes specifics. While we develop SmartCGMS primarily for treating diabetes, we can apply SmartCGMS to any signal processing. This benefits diabetic patients, because SmartCGMS can process additional signals. For example, it can consider the accelerometer signal to estimate physical activity of the patient. Hence, SmartCGMS can process all signals of e.g.; the OhioT1DM data set without any need for a non-systematic, ad-hoc solution.

Acknowledgements

This publication was supported by the project LO1506 of the Czech Ministry of Education, Youth and Sports and university specific research project SGS-2019-016.

SmartCGMS is available for download at https://diabetes.zcu.cz/smartcgms

References

[1] Abraham Mary, Smith Grant, Nicholas Jennifer, Fairchild Janice, King Bruce, Ambler Geoffrey, Cameron Fergus, Davis Elizabeth, and Jones Timothy, ‘Characteristics of automated insulin suspension and glucose responses with the predictive low-glucose management system’, *Diabetes technology & therapeutics*, (2019).

[2] Tadej Battelino, Thomas Danne, Richard M Bergenstal, Stephanie Amiel, Roy Beck, Torben Biester, Emanuele Bosi, Bruce Buckingham, William efalu, Kelly Close, et al., 'Clinical targets for continuous glucose monitoring data interpretation: recommendations from the international consensus on time in range', *Diabetes Care*, **42**(8),1593–1603, (2019).

[3] Francesco Biscani, Dario Izzo, and Chit Hong Yam, 'A global optimization toolbox for massively parallel engineering optimisation', *arXiv preprint arXiv:1004.3824*, (2010).

[4] Bruce W Bode, 'Insulin pump use in type 2 diabetes', *Diabetes technology & therapeutics*, **12**(S1), S–17, (2010).

[5] Bruce Buckingham, Erin Cobry, Paula Clinton, Victoria Gage, Kimberly Caswell, Elizabeth Kunselman, Fraser Cameron, and Peter Chase, 'Preventing hypoglycemia using predictive alarm algorithms and insulin pump suspension', *Diabetes technology & therapeutics*, **11**(2), 93–97, (2009).

[6] Ivan Contreras and Josep Vehi, 'Mid-term prediction of blood glucose from continuous glucose sensors, meal information and administered insulin', in *XIV Mediterranean Conference on Medical and Biological Engineering and Computing 2016*, pp. 1137–1143. Springer, (2016).

[7] S. Del Favero, A. Facchinetti, G. Sparacino, and C. Cobelli, 'Improving accuracy and precision of glucose sensor profiles: retrospective fitting by constrained deconvolution', *IEEE Trans Biomed Eng*, **61**(4), 1044–1053, (Apr 2014).

[8] Sandrine Ding and Michael Schumacher, 'Sensor monitoring of physical activity to improve glucose management in diabetic patients: a review', *Sensors*, **16**(4), 589, (2016).

[9] Andrea Facchinetti, 'Continuous glucose monitoring sensors: past, present and future algorithmic challenges', *Sensors*, **16**(12), 2093, (2016).

[10] Fernando Garcıa-Garcıa, Roman Hovorka, Malgorzata E Wilinska, Daniela Elleri, and M Elena Hernando, 'Modelling the effect of insulin on the disposal of meal-attributable glucose in type 1 diabetes', *Medical & biological engineering & computing*, **55**(2), 271–282, (2017).

[11] John E. Hall, *Guyton and Hall Textbook of Medical Physiology, 13e (Guyton Physiology)*, Saunders, 2015.

[12] Peter G Jacobs, Navid Resalat, Joseph El Youssef, Ravi Reddy, Deborah Branigan, Nicholas Preiser, John Condon, and Jessica Castle, 'Incorporating an exercise detection, grading, and hormone dosing algorithm into the artificial pancreas using accelerometry and heart rate', *Journal of diabetes science and technology*, **9**(6), 1175–1184, (2015).

[13] J. Larry Jameson, Anthony S. Fauci, Dennis L. Kasper, Stephen L. Hauser, Dan L. Longo, and Joseph Loscalzo, *Harrison's Principles of Internal Medicine, Twentieth Edition (Vol.1 & Vol.2)*, McGraw-Hill Education/ Medical, 2018.

[14] Steven G Johnson. The nlopt nonlinear-optimization package, 2014.

[15] Tomas Koutny, 'Prediction of interstitial glucose level', *IEEE Trans Inf Technol Biomed*, **16**(1), 136–142, (Jan 2012).

[16] Tomas Koutny, 'Blood glucose level reconstruction as a function of transcapillary glucose transport', *Comput. Biol. Med.*, **53**, 171–178, (Oct 2014).

[17] Tomas Koutny, 'Using meta-differential evolution to enhance a calculation of a continuous blood glucose level', *Comput Methods Programs Biomed*, **133**, 45–54, (Sep 2016).

[18] Tomas Koutny, 'Crosswalk–a time-ordered metric', in *EMBEC & NBC 2017*, 884–887, Springer, (2017).

[19] Tomas Koutny, Michal Krcma, Josef Kohout, Petr Jezek, Jana Varnuskova, Petr Vcelak, and Jan Strnadek, 'On-line blood glucose level calculation', *Procedia Computer Science*, **98**, 228–235, (2016).

[20] Tomas Koutny and Martin Ubl, 'Parallel software architecture for the next generation of glucose monitoring', *Procedia Computer Science*, **141**, 279–286, (2018).

[21] Boris P Kovatchev, Marc Breton, Chiara Dalla Man, and Claudio Cobelli.In silico preclinical trials: a proof of concept in closed-loop control of type 1 diabetes, 2009.

[22] Paul Leach, Michael Mealling, and Rich Salz, 'A universally unique identifier (uuid) urn namespace', (2005).

[23] Jinxun Liu, Changhui Peng, Qinglai Dang, Mike Apps, and Hong Jiang, 'A component object model strategy for reusing ecosystem models', *Computers and Electronics in Agriculture*, **35**(1), 17–33, (2002).

[24] Cindy Marling and Razvan C Bunescu, 'The OhioT1DM dataset for blood glucose level prediction.', in *KHD@ IJCAI*, pp. 60–63, (2018).

[25] Leonid Poretsky, *Principles of diabetes mellitus*, volume 21, Springer, 2010.

[26] Paolo Rossetti, Carmen Quiros, Vanessa Moscardo, Anna Comas, Marga Gim´enez, F Javier Ampudia-Blasco, Fabi´an Le´on, Eslam Montaser, Ignacio Conget, Jorge Bondia, et al., 'Closed-loop control of postprandial glycemia using an insulin-on-board limitation through continuous action on glucose target', *Diabetes technology & therapeutics*, **19**(6), 355–362, (2017).

[27] Stephane Roze, Jayne Smith-Palmer,William Valentine, Vincent Payet, Simona de Portu, Natalie Papo, Michel Cucherat, and Helene Hanaire, 'Cost-effectiveness of sensor-augmented pump therapy with low glucose suspend versus standard insulin pump therapy in two different patient populations with type 1 diabetes in France', *Diabetes technology & therapeutics*, **18**(2), 75–84, (2016).

[28] Maria Rettian Anggita Sari et al., 'Optimal blood glucose level control using dynamic programming based on minimal bergman model', in *Journal of Physics: Conference Series*, volume 974, p. 012036. IOP Publishing, (2018).

[29] Martin Ubl and Tomas Koutny, 'SmartCGMS as an environment for an insulin-pump development with fda-accepted in-silico pre-clinical trials', *Procedia Computer Science*, **160**, 322–329, (2019).

[30] Martin Ubl, Zuzana Majdisova, and Tomas Koutny, 'Icarus has diabetes', in *Technology in Diabetology*, (2019).

[31] Man, Chiara Dalla, Francesco Micheletto, Dayu Lv, Marc Breton, Boris Kovatchev, and Claudio Cobelli. 'The UVA/PADOVA Type 1 Diabetes Simulator: New Features.' Journal of Diabetes Science and Technology 8, no. 1 (January 2014): 26–34.

[32] Visentin, Roberto, Enrique Campos-Náñez, Michele Schiavon, Dayu Lv, Martina Vettoretti, Marc Breton, Boris P. Kovatchev, Chiara Dalla Man, and Claudio Cobelli. "The UVA/Padova Type 1 Diabetes Simulator Goes From Single Meal to Single Day." Journal of Diabetes Science and Technology 12, no. 2 (March 2018): 273–81.