

# Analysis of Continuous Glucose Monitoring

---

Yale-NUS College Special Project in Science

**Hrishi Olickel**

*Yale-NUS College*

hrishi.olickel@u.yale-nus.edu.sg

**Hebe Hilhorst**

*Yale-NUS College*

hebe.hilhorst@u.yale-nus.edu.sg

# Abstract

In this work, we investigate current methods for Continuous Glucose Monitoring and their accuracy, in order to develop a framework for better reporting and data collection as well as prediction. Additional investigation is done into the use of Blood or Interstitial Glucose as a predictor of general health.

# Contents

<b>1</b>	<b>Background</b>	<b>1</b>
1.1	History . . . . .	1
1.2	Sensor Choice . . . . .	3
1.3	Freestyle Libre . . . . .	4
<b>2</b>	<b>Technical Analysis</b>	<b>5</b>
2.1	Event Analysis and Review . . . . .	5
2.2	Sensor Chemistry . . . . .	6
2.2.1	Amperometric Glucose Biosensors . . . . .	6
<b>3</b>	<b>Development</b>	<b>10</b>
3.1	Architecture . . . . .	10
3.2	Communication Protocol Analysis . . . . .	12
3.2.1	Related Work . . . . .	12
3.3	HEX Analysis . . . . .	13
3.4	Application Development . . . . .	17
3.5	Reverse Engineering the Reader . . . . .	18
<b>4</b>	<b>Analysis</b>	<b>20</b>
4.1	Data Comparison . . . . .	20
4.2	Prediction . . . . .	22
4.2.1	Linear Regression . . . . .	23
4.2.2	Neural Networks . . . . .	25
4.2.3	Reader Prediction Arrow . . . . .	26
4.2.4	Clinical Analysis . . . . .	28
<b>5</b>	<b>Appendix</b>	<b>32</b>
5.1	Source Code Organization . . . . .	32
	<b>Bibliography</b>	<b>34</b>

# Background

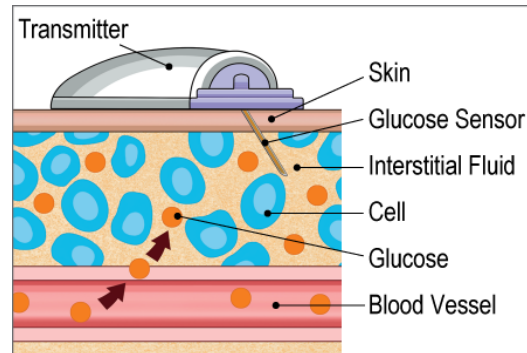
Diabetes means that a patient must manually control their glucose levels. Glucose levels outside of normal levels, typically considered 4.0 mmol/L to 8.0 mmol/L pose a significant danger. Being below optimal - **hypoglycaemia** - can cause unconsciousness and lasting brain damage within minutes. Being above optimal - **hyperglycaemia** - can cause fatigue, thirst, headaches, etc in the short term. In the long term, it is very problematic and can lead to nerve damage and many chronic health issues such as nerve damage, blindness, and heart disease.

## 1.1 History

While it has other applications, glucose measurement technology has always been driven by diabetes treatment and care. Diabetics need glucose measurements in order to inform immediate management, to analyze historical trends for future changes in care, and as a performance metric to measure whether they've met their goals. For the performance metric, glycated haemoglobin (HbA1c) measurements are most commonly used. This provides a rough average of glucose over the past three months. Everyday management, however, requires a user-friendly system that provides a good idea of the current location and behaviour of blood glucose.

Accurate glucose measurement is a key aspect of diabetes control, and has been developed for that purpose over the past century or so. Initially, it was measured through urine. This was a very crude method with poor accuracy and a significant time lag, so capillary blood glucose measurement took over in the form of fingerprick tests. While significantly more timely and accurate, these are uncomfortable and only provide a snapshot view. Subcutaneous Continuous Glucose Monitors, or CGMs, have been developed over the past decade to provide a semi-continuous (typically every 15 minutes) reading of interstitial fluid glucose levels. More systems have been sporadically developed, such as GlucoWatch, near infrared spectroscopy, and microdialysis. However, these have all fizzled out, usually because of issues with inaccuracy or user discomfort. Venous blood plasma is still in use, but due to the obvious impracticalities this is typically limited to hospital inpatients. This leaves capillary blood glucose spot checks and semi-continuous interstitial fluid monitoring as the current existing technology.

Capillary blood glucose is considered the clinical gold standard. It's easy to self-administer and quick to display changes in glucose level, making it the most responsive and up-to-date for patient management. Mostly, however, it has simply long been the only option. That means decades of medical practices have been developed using it for reference, so modern clinical care and diabetes management respond to venous blood information. It's been institutionalized in.



**Fig. 1.1:** Illustration of a CGM sensor.

State-of-the-art ground truth measurement for glucose levels is typically ascribed to the Yellow Springs Instrument Glucose Analyzer, which measures venous plasma, but day to day use is handled through handheld Blood Glucose Meters (BGMs). Although accepted as ground truth in many studies, these are far from perfect. The accepted Diabetes Technology Society standard is that a BGM must be accurate to within 15% at least 95% of the time, and within 20% at least 99% of the time, compared to YSI readings [Noaf]. Many meters do not meet this standard<sup>1</sup>. Another popular assessment for BGMs is Clarke Error Grid Analysis (EGA). This is intended to measure clinical accuracy, or the likelihood that a glucose reading will provoke a detrimental management decision, and is equally important.

Over the past half decade, CGMs have been working their way into common practice. For immediate management, they give both an instantaneous spot check and an idea of current glucose behaviour, which is very helpful for tailoring action. On top of this, they also give semi-continuous historical data that can be used to get a better idea of glucose response to various environmental variables, which helps significantly in perfecting management. Analysis of this historical record can also lead to a more high definition performance metric.

Although CGMs directly measure interstitial fluid, their accuracy is judged by how close they are to blood glucose readings - officially to YSI, but most studies use BGM readings as an acceptable ground truth. A CGM accuracy will typically be reported as how close the spot checks are to a corresponding BGM reading. Although CGMs also give trend data, performance metrics for this are still being developed within

<sup>1</sup>A formal study on accuracy is [Cla+87], more recent informal studies are [Sch],[Noad],[Ede13].

the community. When it comes to clinical accuracy, the basic EGA is often used. However, Continuous Glucose Error Grid Analysis (CG-EGA) is in the late stage of development to judge the clinical accuracy of CGMs based on the wider range of information they provide.

BGM capillary blood glucose is the accepted treatment basis for glucose management, because it's the most reactive metric and already embedded in diabetes care. All other systems, regardless of how they sample glucose, strive to match it. However, continuous monitoring also provides a wealth of other useful information relevant to both diabetics and others.

## 1.2 Sensor Choice

Among the many commercially available sensors, we used Abbott's Freestyle Libre. Among other reasons, it was the only one we could use. DexCom is not available in Singapore, and Medtronic only displays to an insulin pump. Being based in Singapore without an insulin pump, Abbott was the only option. It is not strictly speaking a CGM, since it doesn't broadcast the results, which must be manually checked. Since that's the only difference, for ease of reference it's typically included under the umbrella term anyway.



**Fig. 1.2:** The Nightscout ecosystem.

The Freestyle Libre has several other advantages. It is the cheapest and longest lived sensor available, with appreciable accuracy in all papers. The website is easily the

most user-friendly, which is far more beneficial than it sounds. Most relevant to our purposes, it has the largest hacker community already formed. In particular, protocols and proof-of-concept software had been developed for accessing values straight from the sensor, and as we later found out, from the reader. Achieving something similar with DexCom apparently requires soldering together a xDrip device from scratch and online instructions. Data sent from sensor to display can be captured from both DexCom and Medtronic, with associated GitHub repositories. However, it's more complicated and not intended for personal data analysis, being part of the bigger Nightscout[Noam] project, which aims to provide a complete framework for managing glucose information.

## 1.3 Freestyle Libre

The Freestyle Libre CGM<sup>2</sup> consists of a subcutaneously implanted sensor and hand-held reader. Interstitial fluid readings are taken every 60 seconds, by the method described in detail below. The sensor records these raw values. Scanning the sensor with the reader automatically transmits the stored data through NFC. The reader then processes the raw data and displays a current glucose value, 8hr historical trend line, and an arrow indicating trend direction.

Studies have shown [Abb] that using continuous glucose monitors (the Freestyle Libre in particular) have resulted in a 16% reduction in HbA1c levels, as well as significantly lower occurrence and duration of hypoglycemic episodes. This is an extraordinarily positive result that lowers both short and long term risks due to diabetes complications by a significant margin.

---

<sup>2</sup>It must be noted that the FreeStyle Libre is officially classified as a Flash Glucose Monitor (FGM), in that it does not alert the user directly, requiring a scan to do so. However, for the purposes of this report and the framework being built, we will use the terms CGM and FGM interchangeably.

# Technical Analysis

## 2.1 Event Analysis and Review

The FreeStyle Libre sensor takes readings every 60 seconds, but it does not retain all of these, because of what we presume are storage limitations. Instead, it keeps half an hour of high definition data (30 values separated by 60 seconds) and an additional 7.5 hrs of coarse 15 minute data (45 values separated 15 min), disregarding the rest. The sensor also contains a power source, for both the electronics and the measurement mechanism, and a thermometer. When scanned, the sensor provides the stored data (30min fine, 7.5hr coarse), associated internal time, and uncalibrated temperature information. This information was gained during data analysis, as discussed later.






The sensor provides a series of raw, unsmoothed datapoints at a lag from ground truth, which the reader aims to transform into clinically useful information. Their precise algorithm is appropriately a trade secret, but provides transformed historical data, point estimate, and trend. Sometimes it will refuse to display any data, typically if glucose levels are changing very rapidly. Since the company provided accuracy study uses industry-standard Yellow Springs Instrument levels as ground truth, the provided data can safely be assumed to be intended to represent venous blood glucose. Their user manual also provides a chart for interpreting the glucose trend arrow, as shown in Figure 2.1<sup>1</sup>.

We found that the reader point estimate provides greater mathematical accuracy to blood glucose than the associated raw sensor value. The historical trend line is displayed in too low a resolution for much accuracy discussion, but as discussed later, more accurate data can be gained. The trend arrow will be discussed more in depth in future. Of course, clinical accuracy must also be considered.

As well as processing and displaying the raw data, the reader provides several extra capabilities. It allows users to set date, time and target glucose range. With each scan, it also allows the user to add flags to record extra events such as insulin, food, medication and exercise. Flags for other events can be added using the software. Historical records can be viewed as well as limited analytics such as weekly averages,

<sup>1</sup>from the FreeStyle Libre manual supplied with the sensor.



	<b>Glucose is rising quickly</b> (more than 0.1 mmol/L per minute)
	<b>Glucose is rising</b> (between 0.06 and 0.1 mmol/L per minute)
	<b>Glucose is changing slowly</b> (less than 0.06 mmol/L per minute)
	<b>Glucose is falling</b> (between 0.06 and 0.1 mmol/L per minute)
	<b>Glucose is falling quickly</b> (more than 0.1 mmol/L per minute)

**Fig. 2.1:** Conditions where arrows are displayed.

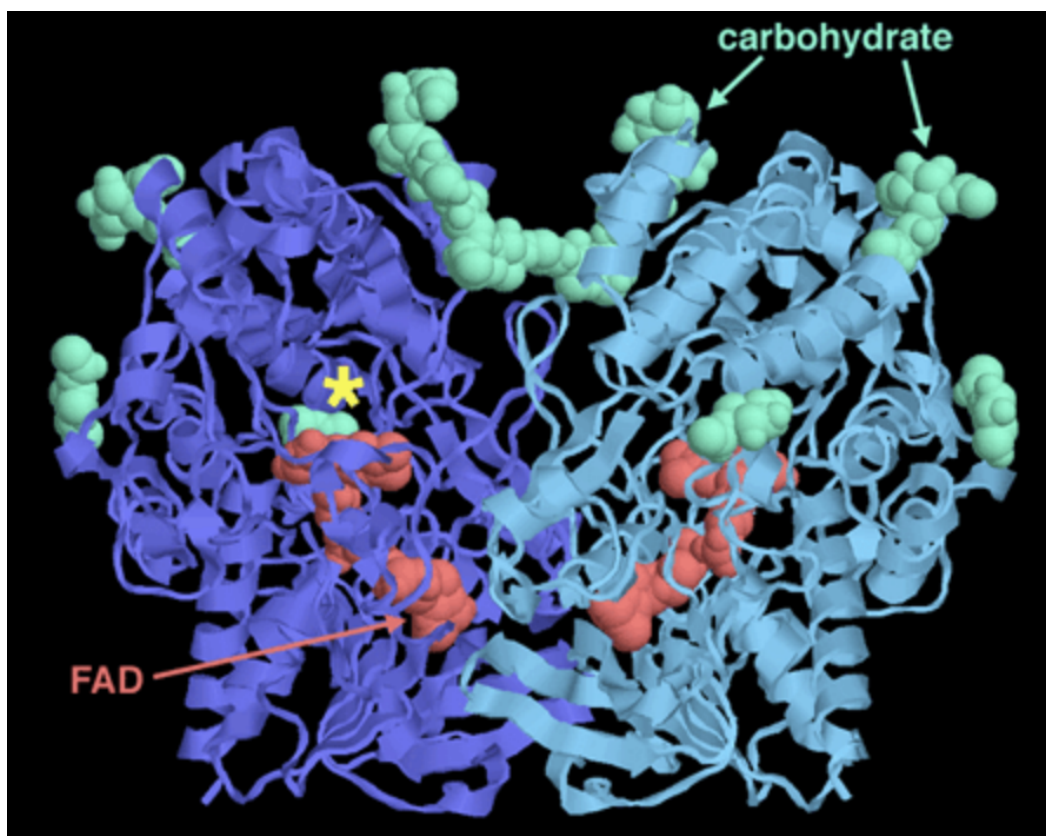
time in range, and typical daily trend. The reader can also act as an independent blood glucose or ketone meter.

## 2.2 Sensor Chemistry

The technology underlying most commercially available continuous glucose monitors is the same. They are inserted subcutaneously, typically into the arm or stomach. They consist of an implanted, flexible catheter connected to the external plastic shell, which is typically attached to the skin with adhesive. The catheter acts as an amperometric biosensor and takes a raw measurement of interstitial glucose. The external hardware provides power, keeps time, records the measurements, and is sometimes attached to an optional transmitter. Currently, there are three main commercial brands; DexCom, Medtronic, and Abbott.

### 2.2.1 Amperometric Glucose Biosensors

An amperometric glucose biosensor uses redox electrodes to create and measure a current proportional to interstitial glucose concentration. From the current, they can closely estimate the actual glucose concentration. If there's too much noise for the current and concentration to have a constant ratio, the device will require regular recalibration by matching with blood glucose tests. Since most CGMs aim to eliminate the need for BGMs, manufacturers aim to reduce noise, while maintaining sensor lifespan and comfort.

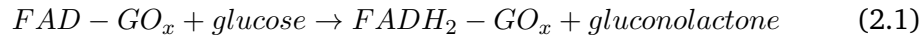


**Fig. 2.2:** Location of flavin adenine dinucleotide (FAD) co-factor in glucose oxidase.

In order to produce the proportional current, an amperometric biosensor electro-oxidizes glucose with the flavin adenine dinucleotide (FAD) co-factor of glucose oxidase as shown in Equation 2.1, then measures the current provided when re-oxidizing the reduced glucose oxidase. It is difficult to directly oxidize the FAD co-factor, since it's buried deep within the molecule (Figure 2.2), so a mediator is used. This creates a three step system: oxidize glucose with glucose oxidase, react with the mediator, then oxidize the reduced mediator. At each of these steps, the concentration of the products remains proportional to the concentration of glucose. At the last step, the electro-oxidation of the mediator, the current produced will also be proportional. The speed of electron movement will be proportional to the reaction rate, which will be proportional to the reactant (reduced mediator) concentration [Noac]. This provides an acceptable raw representation for interstitial fluid glucose concentration. Anything that interferes with any of these proportions creates noise.

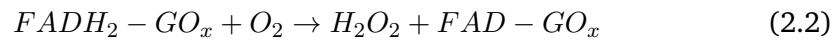
Most often, oxygen is used for the mediator. This results in the chain of reactions shown in Equations 2.2 and 2.3. Oxygen is a useful mediator, because instead of needing to be immobilized in on the electrode, it can be taken in vivo. However, the reactions need a higher ratio of oxygen to glucose that is found in interstitial fluid to be successful. To overcome this, both Dexcom and Medtronic have developed

complicated membranes that only allow the desired ratios in - remaining, of course, in proportion to the global body glucose[Noai]. However, this introduces more noise. The heavy design requirements for the membrane also mean that it lacks flexibility to control other noise-inducing factors. This contributes to both those systems requiring twice-daily calibration.



Reduction half-equation:  $FAD - GO_x + 2H_+ + 2e_- \rightarrow FADH_2 - GO_x$

Oxidation half-equation:  $glucose \rightarrow gluconolactone + 2H_+ + 2e_-$

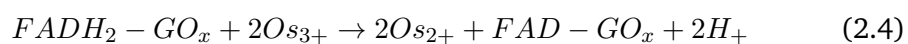


Reduction half-equation:  $O_2 + 2H_+ + 2e_- \rightarrow H_2O_2$

Oxidation half-equation:  $FADH_2 \rightarrow FAD - GO_x + 2H_+ + 2e_-$



Abbott overcame this issue with their trademarked Wired Enzyme technology[Noap]. This uses immobilized osmium complexes as their mediator, resulting in the reactions shown in Equation 2.4. This has several benefits. It avoids involving several confounding variables into the equation by removing the need to use internal oxygen and by oxidizing the glucose oxidase in place, instead of in solution. Most usefully, re-oxidizing the osmium requires a much lower voltage than doing so for the oxygen. This avoids interference from other substances, like uric acid or medical acetaminophen[Noai]. Overall, the Wired Enzyme decreases noise enough that it need only be calibrated in the factory, since the current to glucose concentration ratio stays constant[Hos+14]. On the other hand, it means that Abbott devices have a clear expiration date, since they become completely unusable once they run out of osmium.



Reduction half-equation:  $2Os_{3+} + 2e_- \rightarrow 2Os_{2+}$

Oxidation half-equation:  $FADH_2 \rightarrow FAD - GO_x + 2H_+ + 2e_-$

Other factors also confound accuracy. Regardless of what mediator a sensor uses, it has to resolve biofouling. Tissue response to a foreign body injection can mean that local glucose stops accurately representing global glucose. Companies increasingly minimize this with improving biocompatible material technology. Another problem which is yet to be properly solved is tissue inflammation upon injection. This typically means that initial results will be less accurate, getting more so as the inflammation goes down.

One environmental variable that hasn't been properly addressed is temperature. All amperometric biosensors rely on enzyme reactions, which vary heavily with temperature[Noab]. Enzyme activity increases alongside temperature, which carries over throughout the process and hence to the reported glucose values. No mention can be found of how any CGM controls for this within the sensor hardware, and it's difficult to imagine how they could. No mention was found of whether this was controlled for in software, and anecdotal evidence suggests that the current methods are ineffective.

CGM sensors record interstitial fluid glucose by using an amperometric biosensor to measure a current that's directly proportional to glucose concentration. In order to make sure that the proportionality remains, companies have put a lot of work into minimizing chemical and biological interference.

## Development

Below is the process of developing a glucose monitoring app using the FreeStyle Libre FGM sensor, which involved reverse engineering the protocol of the Libre, and developing a system for calibration and conversion of raw sensor and temperature values into readings. The application also features speed improvements over current state of the art, and is designed to be open source to function as a platform for building predictive and analytics tools based on such data, in addition to data aggregation.

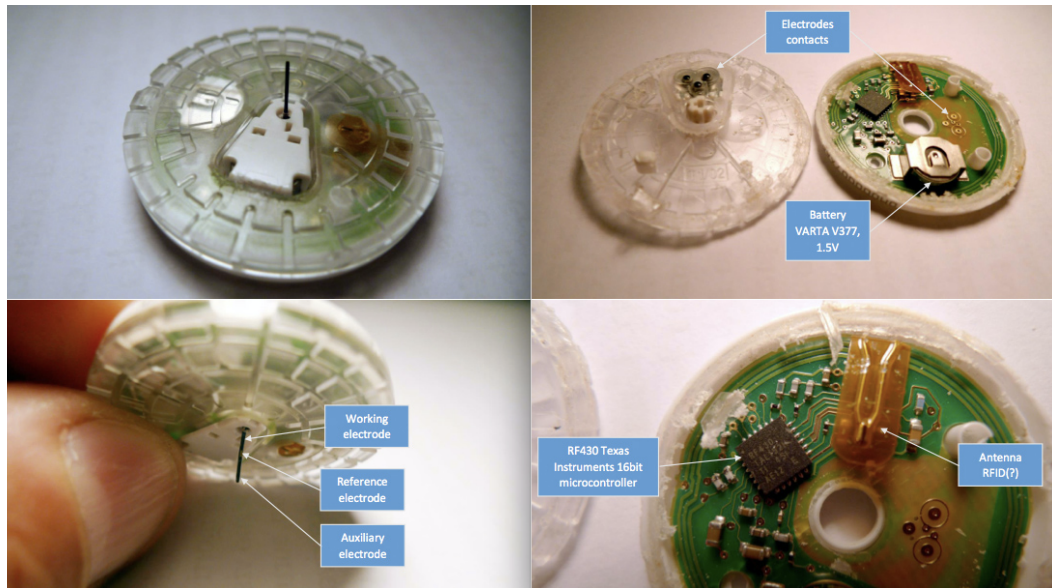
### 3.1 Architecture



**Fig. 3.1:** Freestyle Libre sensor and reader in use[Noag].

The sensor used in the Freestyle Libre (hereafter referred to as *the sensor* for brevity) is described in multiple patents[Noan], which outline the design and electrochemical composition of the system. Figure 3.1 shows the scale of the sensor and the reader. Most relevant here are the ones that describe the sensor chemistry as well as notes on the encoding patterns. The chemical method being used[Say+00] is described to vary linearly with temperature as well as the concentration of interstitial fluid glucose. The raw values thus extracted from the electrode on the sensor are then stored in a processing unit on the sensor, as underlined by the patent.

A teardown view of the sensor is presented in Figure 3.2, where the processor being used as well as the internal components can be identified. Glucose measurement, as previously discussed, is taken via an amperometric biosensor which is primarily encapsulated within the flexible needle. The sensor apparatus also contains a thermocouple tape [Noaq] which reacts linearly(within thresholds underlined in the patent) to temperature. Single-point and dual-point calibration systems are mentioned, by the use of which the raw sensor glucose information can be calibrated



**Fig. 3.2:** External and internal components of the sensor[Ilk14].

to the internal temperature of the sensor. The only visible temperature sensor<sup>1</sup> is on-board the sensor body, taped to the plastic housing on the side of the user's skin. This leads us to believe that a single-point calibration is being used, which will be confirmed later.

The processor (from the product number) is an RF430FRL152H[Noao]<sup>2</sup>, which is used to read and store raw values from the sensor, and communicate them over a Near-Field Communication(NFC)[Noaj] protocol to the reader apparatus. Unfortunately there are no accessible debugging points on the circuit board, so most of the work will need to be done through the wireless protocol on board.

An on-board battery powers the sensor during operation, however, the patent does not describe any mechanisms for calibrating the voltage drift from the battery provided DC voltage. This was surprising, and leaves potential room for additional work on whether this is a factor.

The reader apparatus (Figure 3.3) is unsurprisingly complex, as it incorporates the mechanism for reading and processing sensor data, USB interfacing components as well as a blood-glucose test strip reader. The internals are well weatherproofed, and the construction is remarkably robust. The existence of a number of test pads (possibly for in-factory calibration and quality control) as well as an unexposed connector leave future avenues for extracting information from the reader.

<sup>1</sup>It must be noted that the processor carried an on-board temperature sensor. However, it is unknown how this is used.

<sup>2</sup>a member of the MSP430[Noaa] family of low-power microcontrollers by Texas Instruments

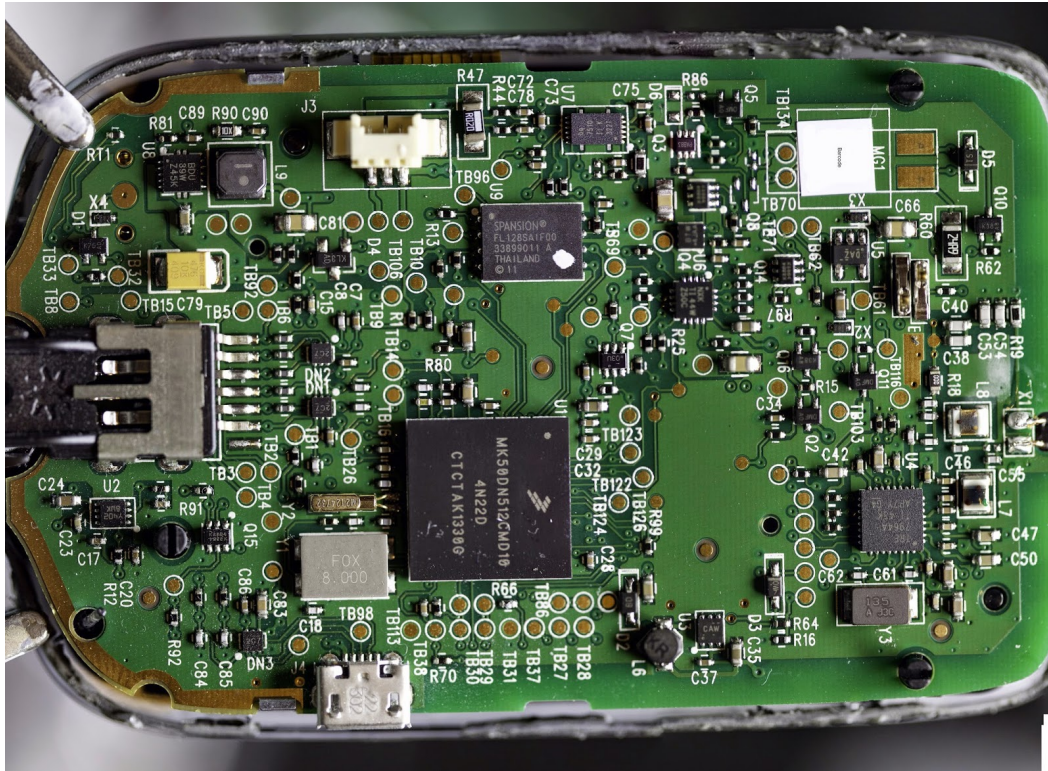


Fig. 3.3: Internal PCB of the reader[Noar].

## 3.2 Communication Protocol Analysis

The datasheet[Noao] for the on-board processor of the sensor describes use of the ISO 15692 protocol[Noah], and reading the information using an NFC reader application on an Android phone confirms it conforms to the Nfc-V[Noal] standard, which is a distance-limiting version of the ISO protocol. This limits the effective use of a reader apparatus to a few centimeters[Noak], but also means that all new Android smartphones with NFC technology will be able to read the sensor with the same protocol. The manufacturer is listed as Texas Instruments, with 1952 bytes of memory being transmitted on a full read. Armed with this information, we can proceed.

### 3.2.1 Related Work

A number of projects have attempted to use the Freestyle Libre as a Continuous Glucose Monitor. There have also been a number of attempts to reverse engineer the protocol being used and to extract the values therein. The most popular (with about 50,000 downloads) of which is the Android application Glimp[Sof17], which reads the FreeStyle Libre sensor and claims to "calibrate" the sensor using Blood Glucose readings independently provided by the user. Unfortunately, this project is not open

source, and the available open source offerings do not go further than reading raw glucose values without calibration or temperature data, and suffer from long read-times (exceeding 2 seconds) for the sensor. Users have also reported that the measurements thus provided are often wrong, and often dangerous - in fact, as one particular analysis shows, the calibration methods used lead to often fatal predictions for users[Ve]. The closest project that could be found was the FreeStyleLibre-NFC-Reader[Bau17] project, which displayed the most recently recorded sensor value<sup>3</sup>. Looking at available solutions for working with the FreeStyle Libre system indicated the dire need for open source projects that aim to improve on intelligent reporting as well as detailed documentation on internal working.

The accompanying Wiki article was useful in providing the following information:

1. The glucose data being recorded on the sensor is split into 16 measurements spaced one minute apart and 32 measurements spaced 15 minutes apart.
2. A circular array is used for writing the data (presumably to minimize rewrites to memory), and the next write position is recorded in the hex before the two arrays.
3. The numbers are encoded in Little Endian Binary[Noae].

However, the author states that his analysis is purely experimental, and that it fails on a sensor that is in use for more than 10 days (the labeled use period of the sensor is 14 days). However, this provides a suitable starting point if we proceed with caution.

### 3.3 HEX Analysis

Visual analysis of the tag memory indicates that a large amount of whitespace remains (Figure 3.4), and comparing to sensor read times for the full tag as opposed to existing solutions (Glimp, Liapp, etc), we can predict that reading only the required bits of the sensor will let us speed up this process considerably.

A python program<sup>4</sup> to process the resulting XML and convert it to an addressable array of hex values enables analysis. Considering a differential of consecutive sensor data dumps, we can see the bytes that differ.

---

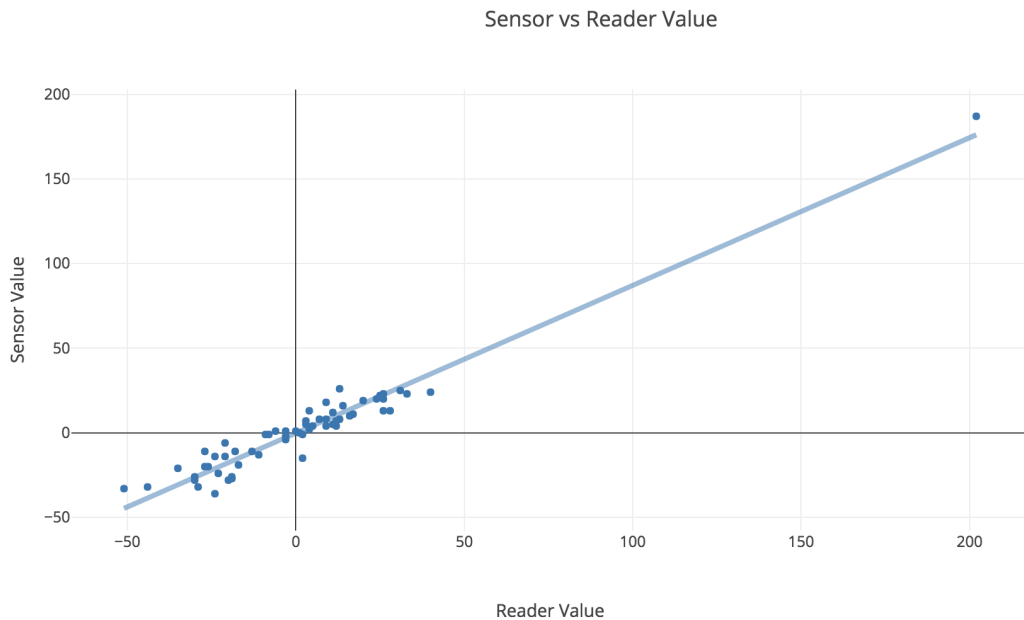
<sup>3</sup>Small caveat being that the code was largely uncommented and labeled in Spanish.

<sup>4</sup>processrawNFC.py in source code





Next, the glucose information was calibrated using previously collected raw glucose data from Glimp[Sof17] which could be matched with processed data extracted from the reader using protocols defined in the open source project Glucometer-protocols[Pet17]. The raw and calibrated data seemed to possess mostly a first order aka linear correlation across sensors (Figure 3.6, and the value of intercept and slope were used to further adjust raw data collected.

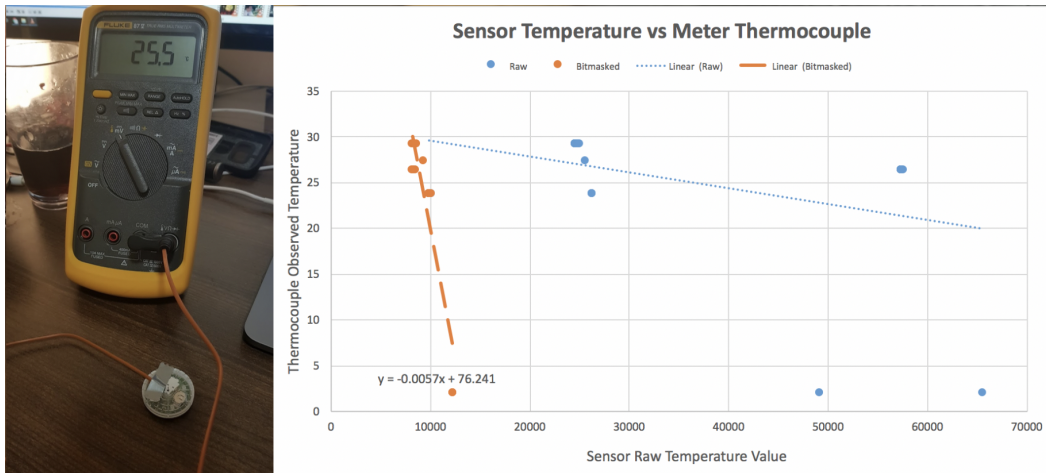


**Fig. 3.6:** Graph of raw sensor vs processed reader glucose values.

Next was the process of extracting and calibrating temperature sensor information. A simple experimental setup using a multimeter thermocouple taped to the sensor (which was placed closest to the external thermocouple in Figure 3.2), following which the apparatus was placed in multiple temperature systems, the average of which was calculated. Once again, the datasheet [Noao] was helpful in expecting a linear correlation between temperature and raw values, so a slope, intercept, and a bitmask for the flags was all that was required.

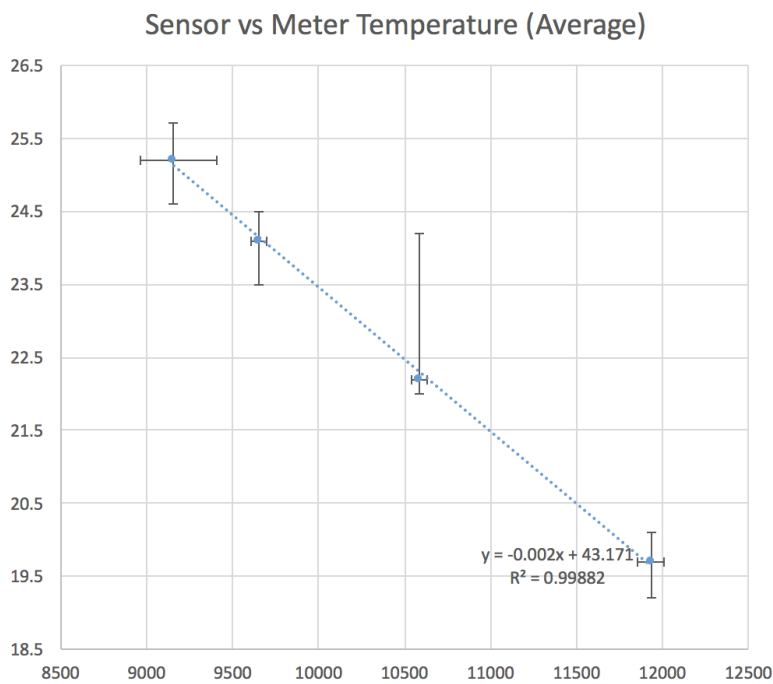
Graphing the resulting data revealed certain flags being turned on and off, which combined with the knowledge that a flag change could be removed with a power of 2, resulted in the bitmask being set at **0x2FFF**, applied to the upper three bytes of the six-byte reading extracted (the same value can be applied to the entire reading, left shifted). The experimental setup and bitmask evaluation can be found in Figure 3.7.

Next, the setup was placed in more accurate ovens that provided better temperature references. In each case, the entire apparatus was placed for a period of 16 minutes. This was due to the uncertainty of the sensor's reading within a minute. Once all



**Fig. 3.7:** Experimental Setup and bitmasking results of Temperature Calibration.

previous short term memory had been overwritten, the average of these results along with the average of the meter reading was used to create a calibration line to find the slope and intercept<sup>5</sup>.



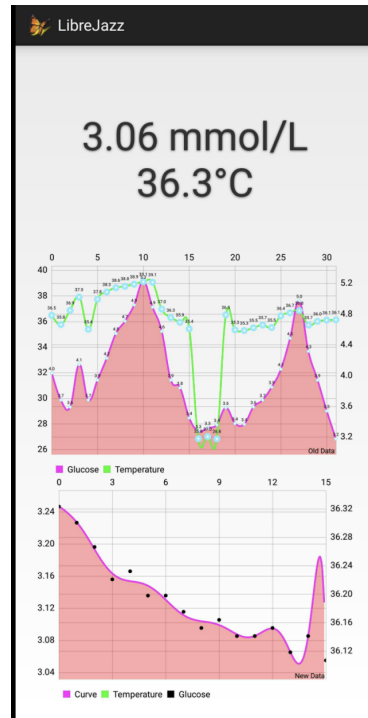
**Fig. 3.8:** Detailed calibration of temperature values.

Figure 3.8 shows how the raw sensor values were related to measured temperature values from the meter. Once the bitmask was applied, the linear relationship was evident.

<sup>5</sup>The meaning of flags were not ascertained due to time constraints and limited testing conditions, and was left for future work.

## 3.4 Application Development

The next step involved developing an Android application using the information found through reverse-engineering the NfcV protocol. The standard Java stack for Android was used, and the resulting application can be seen in Figure 3.9.



**Fig. 3.9:** Android application for reading sensor raw values.

Some optimizations were made to increase read times beyond state-of-the-art smartphone apps. From a preliminary study, current solutions were unable to get sensor scan times less than 1.5 seconds. This was found to be due to the large memory of the NC chip in the sensor, and the slow read times of the protocol. Information could only be read in packets of eight bytes - scanning the entire chip took over five seconds, and reading valid program memory took over a second.

Reverse engineering the Glimp app (which provided no source code) allowed for decrypting the sensor tag label from the metadata provided by the sensor. The full algorithm is shown in Figure 3.10. The Libre sensor uses a reduced set of alphanumeric characters and some form of compression to encode the sensor label that is printed on the side of each device.

Using this data, it was possible to match previously read values to the values being presently read, and to stop when a previously known value was encountered. This greatly reduced read times by removing the need to read the entire memory, or even the entire array of values. For a subsequent read of the same sensor within 60

```

public String getSensorID(String tagID) {
    int [] Is = new int[]{0,0,0,0,0,0};
    int [] Js = new int[]{0,0,0,0,0,0,0,0,0,0};
    final String l = "0123456789ACDEFGHJKLMNPQRTUVWXYZ";
    for(int i=0;i<Is.length;i++) {
        Is[i] = Integer.parseInt(tagID.substring(((6 - i) * 2)-2, \
            ((6 - i) * 2)), 16) & 255;
    }
    Js[0] = (Is[0] >> 3);
    Js[1] = (((Is[0] & 7) << 2) | (Is[1] >> 6));
    Js[2] = ((Is[1] >> 1) & 31);
    Js[3] = (((Is[1] & 1) << 4) | (Is[2] >> 4));
    Js[4] = (((Is[3-1] & 15) << 1) | (Is[3] >> 7));
    Js[5] = ((Is[3] >> 2) & 31);
    Js[6] = (((Is[3] & 3) << 3) | (Is[4] >> 5));
    Js[7] = (Is[4] & 31);
    Js[8] = (Is[5] >> 3);
    Js[9] = ((Is[5] << 2) & 31);

    String sensorID = "";

    for(int i=0;i<Js.length;i++)
        sensorID+=l.charAt(Js[i]);

    return sensorID;
}

```

**Fig. 3.10:** Algorithm used to decrypt sensor tag information.

seconds, the total read time was less than 50 ms, which only included the time it took to read one short term value and one long term to confirm. On average, read times were decreased to less than 500 ms, which provided a significant advantage over existing apps and systems. The only known faster scanning system is the physical reader provided by Abbott, making this a competitive option due to low cost and greater accessibility.

### 3.5 Reverse Engineering the Reader

The goal is to get the app to be more accurate to BGM readings, with the Freestyle Libre providing the current gold standard. Our app provides raw sensor data, while BGM readings are straightforward to collect and record. However, we were also hopeful that the Freestyle Libre reader would be able to provide more data. There had already been significant work towards this. Xavier Claessens reverse engineered many of the previous Abbott devices[[Cla18](#)]. However, most of the prior work reverse engineering the Freestyle Libre was done by Diego Elio Pettenò. He details the reverse engineering journey online[[Pet16](#)] and published the finished product GitHub[[Pet17](#)]. However, this still does not deliver all the wanted data, so needed further adaption.

The Freestyle Libre is charged by USB and further acts as a USB HID, or Human Interface Device, class. This means it can be queried through libusb for information packets, as when connected to the Abbott software. Mostly through trial and error,

the community found two particularly useful commands for data collection. All manual readings - reader spotchecks, blood fingerprick results for glucose and ketones, alongside any associated flags and other information - are returned by the `$arresult` command. Interestingly, a historical dataset of values fifteen minutes apart can also be gained, with the `$history` command. These correspond to the fifteen minute values stored and returned by the sensor.

Unfortunately, the returned data is heavily obfuscated. It comes in 64-bit packets of hexadecimal dump with a significant amount of noise and no hints of how to identify specific information within the data. Luckily, the necessary parts had already been identified within the community. The code necessary to query the raw data from the reader and map most of the important parts was publicly available[Pet17].

In order to end up with the desired data, the device must be queried appropriately, then the returned data must be unobfuscated, separated into instances, and processed. Most of this could be done through the `glucometer-utils` python code available on GitHub, but needed to be extended to include more information and return in a different format. This was achieved by updating the mapping of the unobfuscated data for each instance, to include date, time, value, sensor runtime, arrow, flags, error and reading type information, where applicable. Processing of time and value was changed, and the formatting was adapted to include the new data in preferred format. This was then easily exported to csv format.

# Analysis

With this data, we were able to further investigate accuracy and general glucose analysis. As previously mentioned, capillary blood glucose is considered the clinical gold standard for most treatment. While our app provided a quick, easily accessible way to read the sensor, the returned values were not a perfect fit for either simultaneous BGM readings or the Freestyle Libre reader, which processes the raw values for greater accuracy. We hoped that, with further analysis, we would be able to improve on this accuracy.

The reverse engineering mentioned above provided four datasets in total. The blood glucose readings individually added from fingerpricks, the raw sensor values from our app, the manual readings reported by the reader, and the extracted dataset contained in the reader memory.

## 4.1 Data Comparison

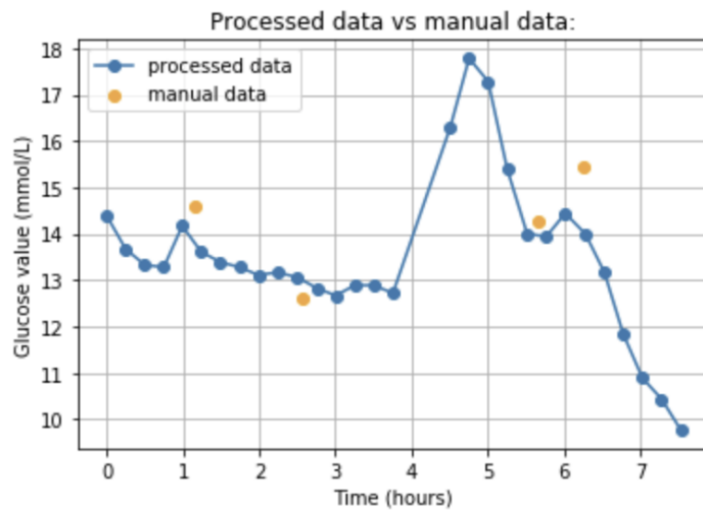
The processed data<sup>1</sup> - provided by \$history - is distinctly different from the raw data, as shown by Figure ???. It is also (excluding the separate manual dataset) evenly spaced 15 minutes apart. This suggests that the processed data is a transformed version of the 15 minute raw datasets. The data displayed on demand with each scan is intended to estimate the current blood glucose level - let's call this read-time data (alternatively manual data, since it is affected by the manual scanning of the sensor). Interestingly, these manually pulled values heavily deviate from the other processed data, as seen in Figure 4.1.

The most obvious reason for this would be an attempt to make up for the time lag between interstitial fluid and blood glucose. On the other hand, this is probably taken into account with all the processed data, which Figure ?? would appear to support. Another option is that the difference is due to the calculation method. The majority of the processed data may be calculated solely off the sensor-provided 15 minute data, even if higher definition data is available, while the read-time value incorporates the 30 minutes of dense data into the calculation. The latter theory

---

<sup>1</sup>referring to the 15-minute data collected from the reader, and raw data likewise refers to data directly extracted from the sensor.

suggests that the read-time values should be more accurate than the rest of the processed data.



**Fig. 4.1:** Graph of read-time reader data.

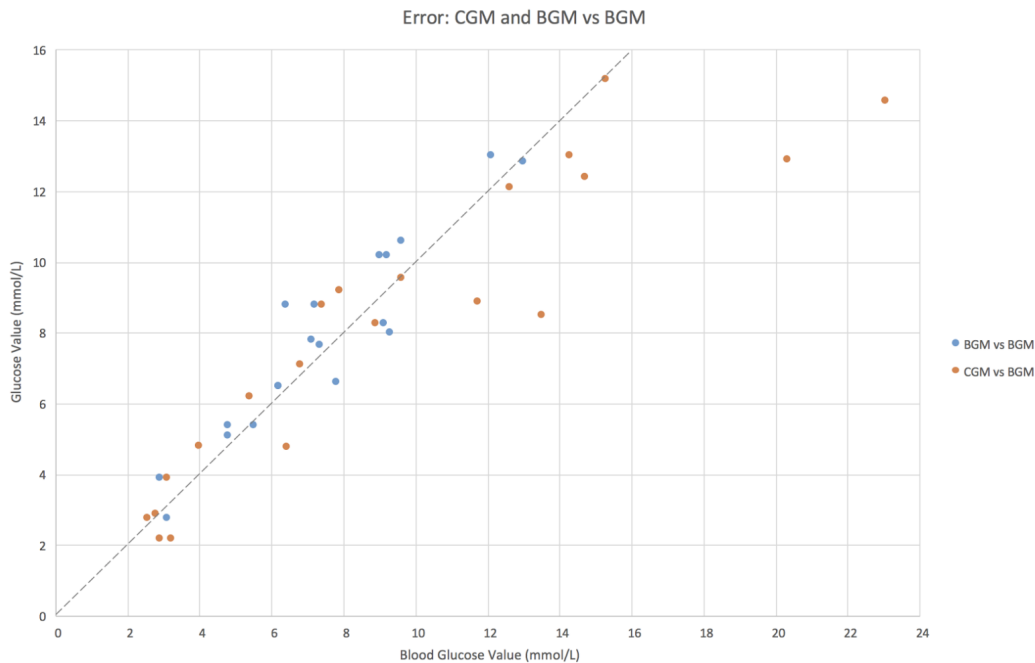
We lacked the data to do a proper accuracy evaluation, and many papers have already done so.

When it comes to accuracy, the Freestyle Libre consistently underperforms BGMs, but not to a debilitating extent. However, in some studies it has been found to fall short of common specifications - Fokket et al. found only 85.5% of results fell within Zone A of the Clarke Error Grid, which has a generally accepted requirement of 95%. Over use, we found that the average difference between almost simultaneous BGM and reader readouts was comparable to the average difference between two semi-simultaneous readings taken with the same BGM, as shown in Figure 4.2. The exception to this were three troubling outliers, where the Libre read significantly under (5-8mmol/L) the BGM ground truth. This is a noticeable trend - the CGM routinely underestimates the BGM glucose reading, especially during hypoglycemia.

Figure 4.3 shows the distribution of the differences between the BGM reading and another simultaneous reading from either the same BGM or the CGM. The CGM clearly does not perform as well as the BGM, although this is significantly improved if the outliers are removed from consideration. Given the 15 minute delay due to intravenous fluid, this is understandable. If anything, it's surprising how much variation there was between two semi-simultaneous readings on the exact same meter.

These findings held true when comparing the limited blood glucose (fingerprick *ground truth*) readings against the available time data, as in Figure 4.4. The Freestyle Libre was usually reasonably accurate to the BGM measurement, even when glucose



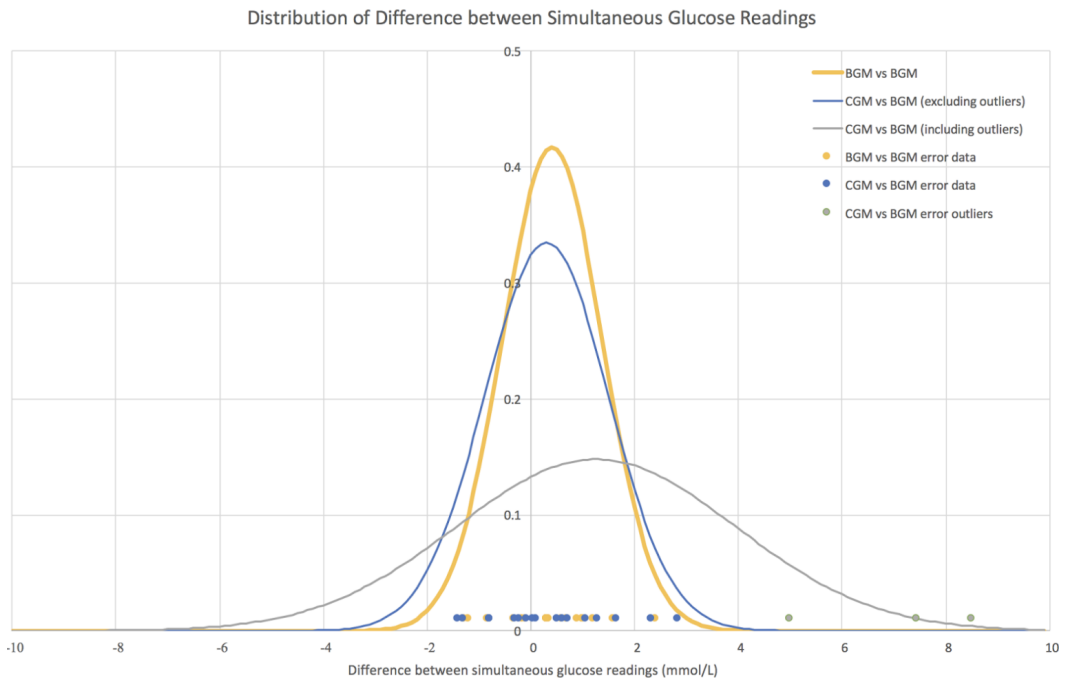


**Fig. 4.2:** Comparison of the correlation between the Freestyle Libre and a BGM measurement with a simultaneous BGM ground truth.

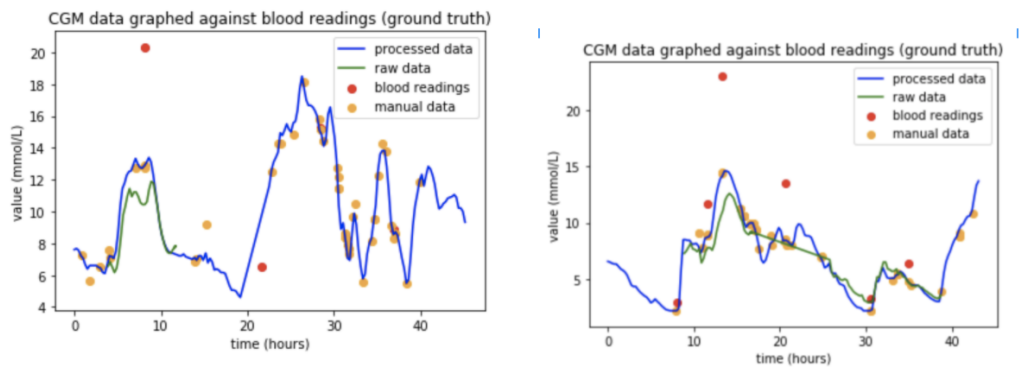
was changing quickly. The processed reader values were more accurate than the raw values, as expected. Where applicable, the manual readings were usually a bit closer still. As previously mentioned, it sometimes wildly underestimated particularly high blood glucose reading, so it is possible that these points of apparent mismatch may be user error, with the BGM reading incorrectly high due to sugar on the fingers or the needle, for instance. Of course, there isn't enough data to form a true conclusion either way.

## 4.2 Prediction

One of the major additional features that a CGM brings is an element of prediction through data analysis. Improving accuracy of this is very helpful. As previously explained, Freestyle Libre communicates predictions with arrows (see Figure 2.1), which translate to the direction glucose is currently trending in mmol/L per minute. First, the accuracy of this needed to be evaluated. The majority of the data we had came in fifteen minute intervals, but it was simple arithmetic to translate the arrow meaning into mmol/L per 15 minute. Matching the time of each manual scan (and hence arrow) to the closest 15 minute change in processed data, an acceptable ground truth, allowed me to get the accuracy of the Freestyle Libre arrows. Depending on the time frame, this varied around 63% +/-5. This seemed fairly low, so effort was made to improve on it using both least squares linear regression and neural networks.



**Fig. 4.3:** Gaussian distributions of the difference between simultaneous BGM readings and simultaneous BGM and Libre readings.



**Fig. 4.4:** Multiple graphs comparing blood glucose data with CGM.

### 4.2.1 Linear Regression

The initial attempt at linear regression was at seeing whether curve fitting data could be extended to predict future values. As Figure 4.5 shows, this was predictably unsuccessful. Extending the curve quickly got out of hand, with even the first values being very poorly fit.

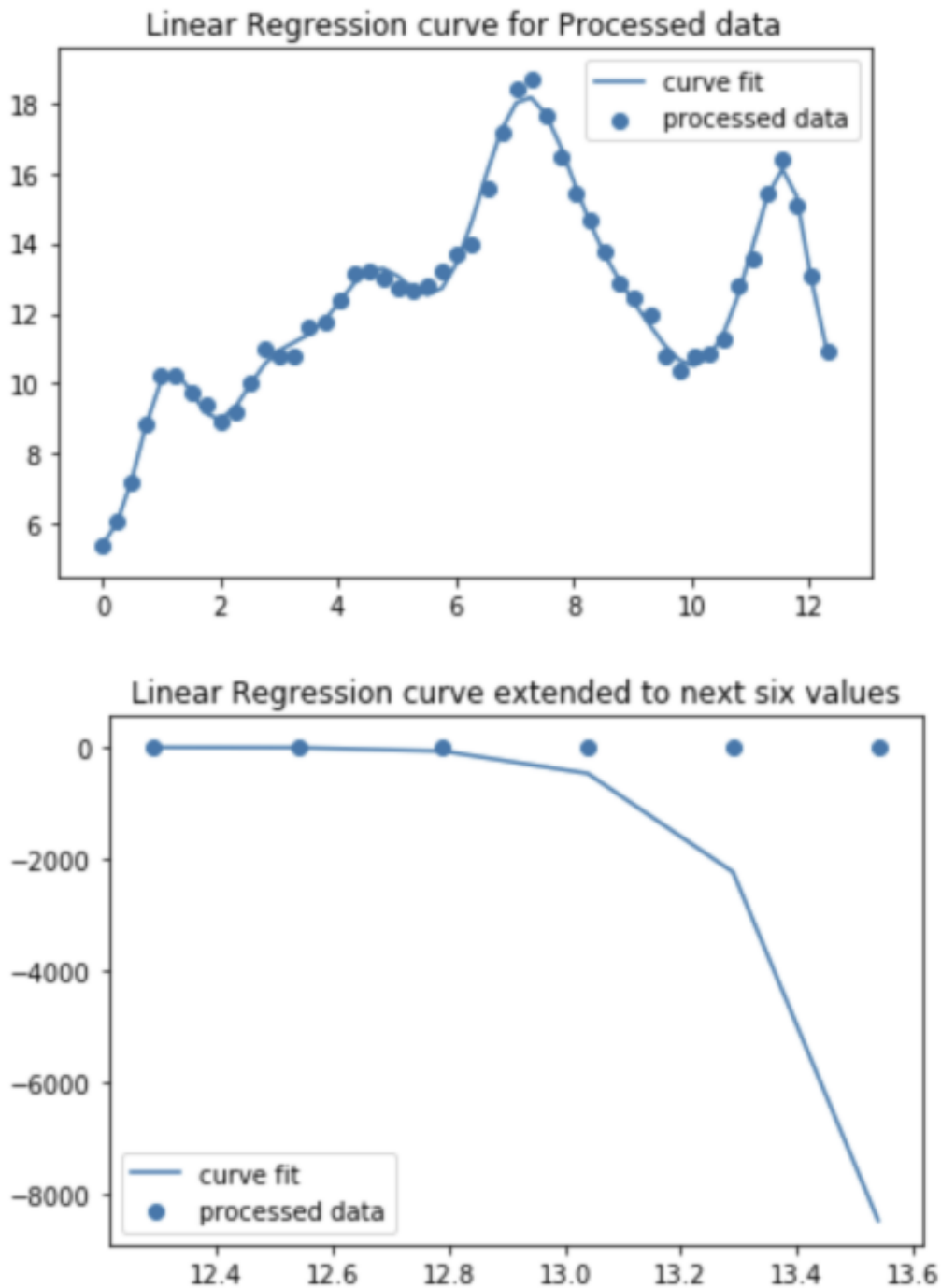


Fig. 4.5: Linear Regression results.

Next, I tried predicting the change in value based on the past 20 datapoints. This initially seemed far more successful than the prior attempt, and even than the reader itself, as shown in Figure 4.6. The accuracy was taken from testing with separate data, not the training set. While there was some fluctuation depending on the training/test sets chosen, it was consistently above 70%. Attempting to improve on this - to predict over an hour, to take derivative values as input, to add prior

- made little improvement, or caused a backtrack. Nevertheless, it seemed pretty successful.

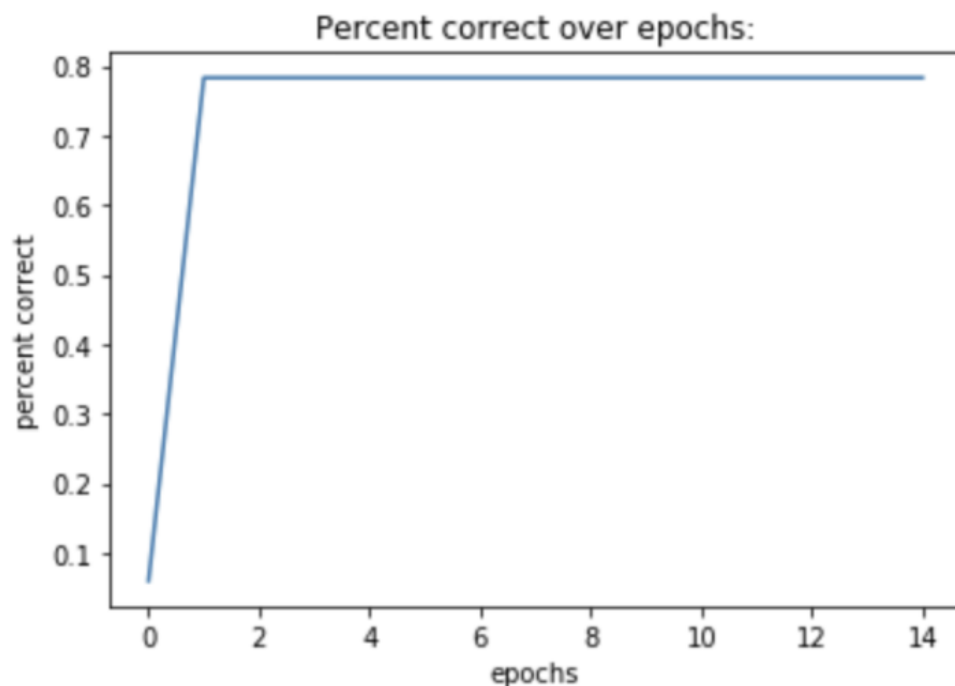
```
Reader accuracy over 15 min, total: 0.631802120141
Reader accuracy over 15 min, only when given: 0.687692307692

Basicest regression accuracy over 15 min, total: 0.781818181818
```

Fig. 4.6: Linear Regression results on prediction.

## 4.2.2 Neural Networks

The next attempt at improving on the prediction arrow was a basic neural network, which I expected to be quite successful. However, it would always immediately go to 78% accuracy and refuse to move through epochs or re-runs, as in Figure 4.7. It fluctuated somewhat depending on the testing data, but it was generally settled. This behaviour seemed strange, so I looked further into it.



```
test correct: 0.783042394015
training correct: 0.650557620818
```

Fig. 4.7: Training error against epoch for neural network.

As it turned out, the problem was that the data was generally too easy to predict. Averaging  $\frac{3}{4}$  of it was going straight. This meant the neural network was just persistently returning *straight*, getting it right most of the time, and not changing. In

fact, looking back at the linear regression, it suffered the same fault. As Figure 4.8 shows, the ground truth for the change in value varied between -4 and 4, albeit heavily clustered around zero. The prediction, on the other hand, remains tightly between -1 and 1. This means it still captures most of the data accurately, but ignores two thirds of the possible range.



```
Training data error is: 0.61684580418
Training data percent correct is: 0.745475113122
```

**Fig. 4.8:** Correlation data for prediction.

### 4.2.3 Reader Prediction Arrow

The reader's glucose trend arrow had a poorer mathematical accuracy than either of my algorithms, while displaying the full range of directions instead of just straight. To better understand how this went, I broke down their correct and incorrect instances, as shown in Figure 4.9. This shows the distribution of the correct matches, the incorrect arrows, and the incorrectly represented ground truths across the trend directions; the distribution of the differences between a displayed arrow and the corresponding true direction, and a more detailed breakdown of the errors. These numbers tell us a few things.

```

Reader Glucose Trend Arrow:

arrow direction key:      ['very down', ' down ', ' straight ', ' up ', ' very up']
correct values percents: [ 0.06040268 0.0704698 0.76733781 0.04362416 0.05816555]
incorrect arrow values:  [ 0.05665025 0.30788177 0.24137931 0.31773399 0.07635468]
incorrect true values:   [ 0.0862069 0.1773399 0.56650246 0.08128079 0.08866995]

difference arrow:value:  ['-2: 18', '-1: 165', '0: 894', '1: 192', '2: 29', '4: 2']

errors when going down: ['same direction: 34', 'straight: 70', 'opposite direction: 3']
errors when going up:   ['same direction: 41', 'straight: 28', 'opposite direction: 0']

errors when going straight: ['up: 116', 'down: 114']

```

**Fig. 4.9:** Trend arrow statistics.

First, while the reader is often inaccurate, it's rarely completely incorrect. Almost 93% of the predictions is within one degree of the ground truth. Less than 2% is further away than two degrees. On the other hand, that 2% is an extreme enough difference to possibly cause harmful management decisions (eg: the difference between 4.2 going straight down, and going straight up).

The reader exaggerates change. When the reader is correct, it displays a straight arrow 77% of the time. When the reader is incorrect, it is much more likely to be signalling a change in glucose, only showing straight 24% of the time. This suggests that the reader algorithm is more responsive to changes in blood glucose than steadiness, and is likely to over exaggerate change in blood glucose. This is in direct contrast to my attempts, which actively conformed everything to 'straight', so all the incorrect values were due to rapidly changing bloods being typed as 'straight' nonetheless.

On the other hand, the true gradient is more likely to be changing when the reader is incorrect. When the reader was incorrect, the gradient was changing 44% of the time. This is almost twice the amount when the reader was correct, 23%. The reader is more likely to read incorrectly on a quickly changing value. Some of this might be overestimating the rate of change, but the rest will be due to underestimating the change. This is in contrast to the early statistics, which suggested that the reader perpetually overestimated change.

With the incorrect readings, the distribution of the true gradient was slightly skewed down. The given arrow value was slightly skewed up. This suggests the reader is more likely to incorrectly predict a higher gradient.

Of the errors that occur while the true value is straight, the uncertainty is very equally distributed between over and underestimating. This suggests a lack of bias. Similarly, the reader has only pointed in completely the wrong direction once.

When true trend is going down, 60% of the reader's incorrect predictions go straight. Only 30% of the time will an incorrect arrow get the direction, at least, correct. In contrast, if the true trend is up, an incorrect prediction is 60% likely to be in the right direction, just with the wrong severity. This suggests that the reader arrow exaggerates upwards trends and downplays downwards trends.

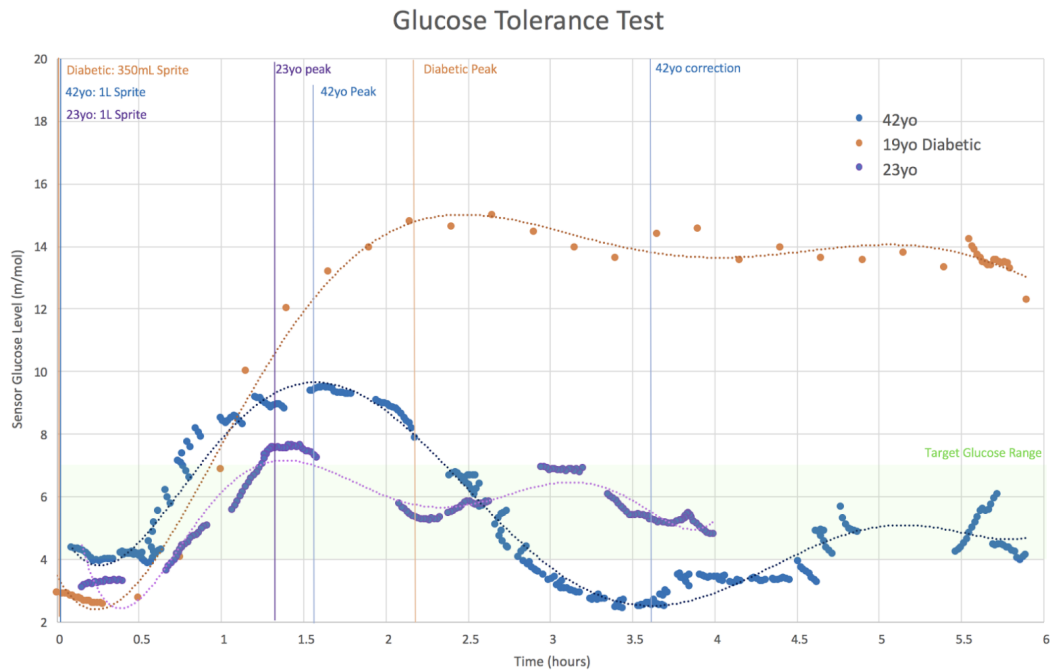
The reader's glucose trend arrow has avoided the trap of only ever predicting straight, while maintaining decent accuracy ratings. It appears to do this by magnifying any change there is in the trend, often resulting in incorrectly predicting more change than there is. This could be due to technical issues preventing greater accuracy. On the other hand, it is most likely due to the arrow's purpose as a clinical treatment aid, not a mathematical instrument. Abbott's internal research may suggest that better treatment decisions are made when glucose rate of change is made more obvious.

#### 4.2.4 Clinical Analysis

Improving the precision and accuracy of these sensors is useful, as a tool to improve monitoring. However, that tool must then be put to use. While the sensors are currently prolific within the diabetic community, they have strong potential for application elsewhere. The sensors are very useful in understanding internal response to glucose. This has a range of possible uses; improving diet, calculating biological age, preventing glucose lows (including in non-diabetics) and controlling 'food comas'. Poor glucose response is indicative of a higher biological age, increases risk of diabetes and cardiovascular disease, and may decrease height [Wil+91][Law+04]. On the other hand, an overactive glucose response can lead to hypoglycemia.

The current state-of-the-art testing for glucose response is limited to the straightforward glucose tolerance test. This comprises of taking a blood glucose test (BGM) to establish baseline, consuming a 1L soft drink, then taking a second blood glucose test (BGM) two hours later. The difference between the two measurements indicates glucose tolerance. If the second test is above baseline, they are considered to have an impaired glucose response. Needless to say, this is a fairly low definition test. CGM monitoring can give a closer look at what the typical test might miss for better diagnosis, while also giving an opportunity to further explore how glucose response relates to other biological factors. As well as giving greater information on overall health and age, a better resolution analysis of glucose levels can give more insight into diet, hyperglycemia (food comas), and hypoglycemia (dizziness, fainting spells, nausea).

Our testing, while far from extensive, did give some insight into the glucose response of three people - 19yo female diabetic, 23yo male non-diabetic, and 42yo male



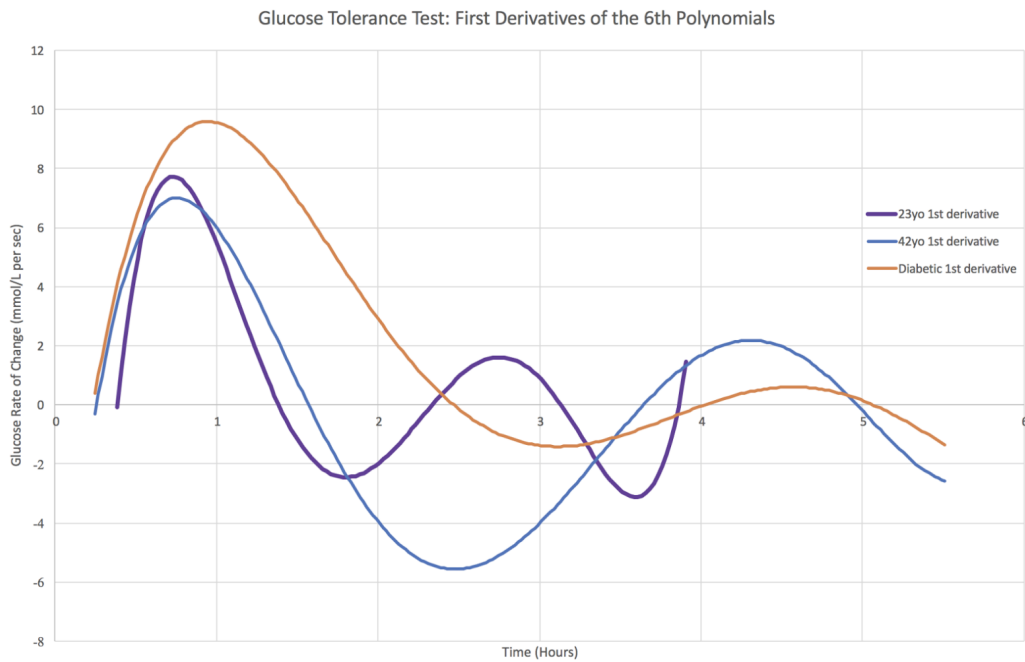
**Fig. 4.10:** Glucose Tolerance Test comparison.

non-diabetic. Figure 4.10 shows the aftermath of a glucose tolerance test (note, the diabetic’s serving size was lowered to avoid health concerns). It’s clear that the younger male has a quicker insulin response to the sugar, peaking at 7.6 mmol/L just 1hr 20 mins after consumption. The older male peaks 15min later at 1hr 35 mins, with a high of 9.5 mmol/L. The post-peak decline is particularly interesting. Although the 42yo peaked later, his glucose response actively turned his blood glucose around and actually pushed him into hypoglycemia for an hour before correcting. The 23yo’s reaction, on the other hand, kicked in earlier and leveled out the high. This is apparent from the first derivative (Figure 4.11), which shows the 23yo’s blood glucose rate of change dropping off sharply. However, although it then decreases somewhat, the drop is comparatively restrained. Interestingly, despite the lack of hypoglycemia, there still appears to be a corrective response.

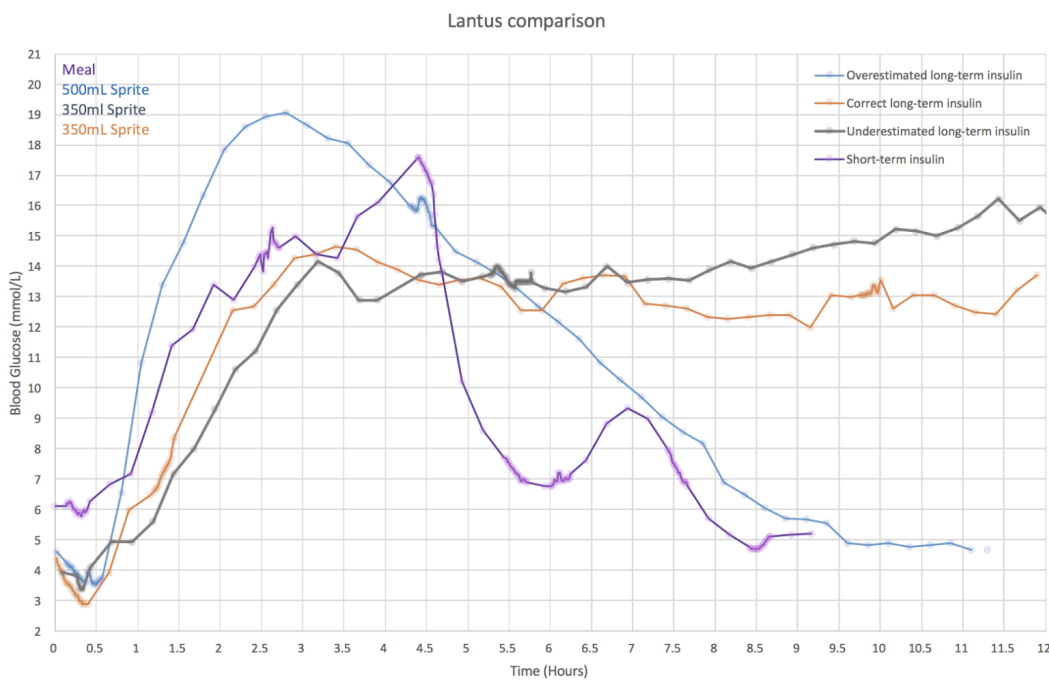
In contrast, the diabetic does not appear to have any counter-reaction to the sugar. Her blood glucose continues to rise to 15mmol/L, when it appears to have run through the sugar over two hours past consumption. It then plateaus, although appears to decrease over time. This is understandable, since a diabetic has limited to zero internal glucose regulation, and instead has to rely on insulin (typically both short-term and long-term) to manage levels. Continuous monitoring can also provide an interesting insight into the effectiveness of this management.

Figure 4.12 shows revealing CGM data from different insulin combinations in the course of everyday life. The appropriateness of long-term insulin, in particular, becomes readily apparent from CGM data, which show whether blood glucose





**Fig. 4.11:** First derivative of the glucose responses (from fitted sixth degree polynomial).



**Fig. 4.12:** Comparison of the effects of different insulin treatments post-food in everyday life.

decreases over time without further action, increases, or stays steady as desired. This level of understanding is very difficult to gain from the discrete and limited nature of BGM data, especially as temporary interference (from food, short-term insulin, exercise, etc) can confound any of the few spotchecks. The CGM data is very easy to interpret on this front. This also provides better monitoring of short-term

insulin. In Figure 4.12, this is given away by the steep post-peak curve. However, it's surprising how long it took to take noticeable effect - three hours, on the inside. This may, however, be more indicative of eating patterns (spending hours nibbling over dinner instead of rushing it down) than issues with the insulin. This would be an interesting point to further research.

# Appendix

All source code and data are located at the primary repo for this paper - <https://github.com/hrishioa/Juventas>

## 5.1 Source Code Organization

Primary organization of the source code is split into three folders: Code, Data and Paper. The **Code** folder contains all of the applications and utilities written and used for the paper, the **Data** folder contains raw data read from multiple versions of the Libre app, along with Liapp and Glimp. The **Paper** folder, understandably contains the tex files used to generate this paper as a form of paperception.

Below are the utilities and scripts included in the **Code** folder:

- **BGMLogger** is a handy script for manually logging blood glucose and tagging information, and produces an output to csv. Usage: `python BGMLogger.py <filename - default is BGMlog.csv>`
- **apk-reverse-engineer** contains all of the files used in the reverse engineering process of the Glimp and Liapp applications. Here the binaries of **dex2jar** and **apktool** are included for repeatability, as well as extracted source code and recompiled binaries. Tread at your own risk.
- **glucometerutils** is a fork of [Pet18], with some modifications added to make logging to csv easier for proper debugging. Some fixes were made to improve performance with the Freestyle Libre, and some locks were removed to improve debugging verbosity<sup>1</sup>.
- **Juventus App** is the Android developed as part of the paper. It is functioning as of the time of writing, and the repository contains the gradle files needed to import and compile in Android Studio, as well as a pre-built apk that will work on Android Versions 17 and up.

---

<sup>1</sup>Please check <https://github.com/flameeyes/glucometerutils> for dependencies and usage

- **Misc Utilities** contains processing scripts, and may therefore be more cluttered than the rest. The contents are -
  1. **glimp\_process.py** can be used to fix unicode errors and patch missing data from the output of `glucometerutils`. Usage: `python glimpse_process.py <input_file> <output_file>`
  2. `processNFCcsv.py` parses the csv hex dumps from the Android app (disabled by default for performance) to compute glucose and temperature information for testing. The color added console output seen in Figure 3.5 is also produced by this script. The input filename is stored in the script and will need to be modified.
  3. `processrawNFC.py` is very helpful is extracting information directly from the sensor, and therefore is more versatile when used on different sensor. The input is a xml hex dump from NXP TagInfo, which is freely available for Android smartphones. Bypassing any other application dedicated to blood sugar also allows for independent algorithm confirmation. Same as before, the input filename is stored in the script and will need to be modified.
  4. `Graphing.ipynb` is the Jupiter notebook containing raw glucose and temperature plots as well as some sanitization functions for datasets.
- **Data** contains all of the raw data used in this experiment. Most common organisation is as a csv, with self-explanatory headings.

# Bibliography

- [Abb] Abbott. *Real-World Data from Abbott's FreeStyle® Libre Show Association Between Higher Frequency of Glucose Monitoring and Improved Glucose Control for People with Diabetes* (cit. on p. 4).
- [Bau17] Victor Bautista. *FreeStyleLibre-NFC-Reader: Read data from a FreeStyleLibre with Android*. original-date: 2014-11-14T11:28:51Z. Nov. 2017 (cit. on p. 13).
- [Cla+87] William L. Clarke, Daniel Cox, Linda A. Gonder-Frederick, William Carter, and Stephen L. Pohl. „Evaluating Clinical Accuracy of Systems for Self-Monitoring of Blood Glucose“. en. In: *Diabetes Care* 10.5 (Sept. 1987), pp. 622–628 (cit. on p. 2).
- [Cla18] Xavier Claessens. *OpenGlucose is an application for diabetics that reads data from supported glucometer devices and display statistics and graphs*. original-date: 2014-08-17T01:43:41Z. Jan. 2018 (cit. on p. 18).
- [Ede13] David Edelman. *Blood Glucose Meter Accuracy Comparison (Chart)*. July 2013 (cit. on p. 2).
- [Hos+14] Udo Hoss, Erwin S. Budiman, Hanqing Liu, and Mark P. Christiansen. „Feasibility of Factory Calibration for Subcutaneous Glucose Sensors in Subjects With Diabetes“. In: *Journal of Diabetes Science and Technology* 8.1 (Jan. 2014), pp. 89–94 (cit. on p. 8).
- [Law+04] C. M. M. Lawes, V. Parag, D. A. Bennett, et al. „Blood glucose and risk of cardiovascular disease in the Asia Pacific region“. eng. In: *Diabetes Care* 27.12 (Dec. 2004), pp. 2836–2842 (cit. on p. 28).
- [Noaa] *16-bit 32-bit MCU | Low-power MCUs | Overview | Microcontrollers (MCU) | TI.com* (cit. on p. 11).
- [Noab] *Accuracy of a Flash Glucose Monitoring System in Diabetic Dogs* (cit. on p. 9).
- [Noac] *Amperometric biosensors* (cit. on p. 7).
- [Noad] *Are Blood Glucose Meters Accurate? New Data on 18 Meters*. Aug. 2017 (cit. on p. 2).
- [Noae] *Endianness*. en. Page Version ID: 812521270. Nov. 2017 (cit. on p. 13).
- [Noaf] *FDA Publishes Final Recommendations on Blood Glucose Meter Accuracy*. Oct. 2016 (cit. on p. 2).

- [Noag] „How to Get Diabetics Addicted to Data“. In: *Bloomberg.com* (Mar. 2017) (cit. on p. 10).
- [Noah] *ISO/IEC 15693*. en. Page Version ID: 804776582. Oct. 2017 (cit. on p. 12).
- [Noai] *Mary Ann Liebert, Inc. - Home* (cit. on p. 8).
- [Noaj] *Near-field communication*. en. Page Version ID: 811370705. Nov. 2017 (cit. on p. 11).
- [Noak] *NFC and ISO15693: Let's be clear! - Global Tag Srl* (cit. on p. 12).
- [Noal] *NfcV | Android Developers* (cit. on p. 12).
- [Noam] *Nightscout – Javorek.eu* (cit. on p. 4).
- [Noan] *Patents | Diabetes | Abbott U.S.* (Cit. on p. 10).
- [Noao] *RF430FRL152H NFC ISO15693 Sensor Transponder With SPI/I2C Interface and 14-Bit Sigma-Delta ADC | TI.com* (cit. on pp. 11, 12, 15).
- [Noap] *Sensor Technology | FreeStyle* (cit. on p. 8).
- [Noaq] „Thermocouple tape“. Pat. Jan. 1971 (cit. on p. 10).
- [Noar] *Type I Diabetes, Coeliac Disease, Tennis.* (Cit. on p. 12).
- [Pet16] Diego Elio Pectenò. *Reverse engineering the FreeStyle Libre CGM, chapter 1*. en-us. Mar. 2016 (cit. on p. 18).
- [Pet17] Diego Elio Pectenò. *glucometer-protocols: A shared repository to provide a description of reverse-engineered glucometer protocols*. original-date: 2016-02-09T00:36:31Z. Dec. 2017 (cit. on pp. 15, 18, 19).
- [Pet18] Diego Elio Pectenò. *glucometerutils: Glucometer access utilities*. original-date: 2013-08-03T08:08:52Z. Jan. 2018 (cit. on p. 32).
- [Say+00] James Say, Michael F. Tomasco, Adam Heller, et al. „Electrochemical analyte“. Pat. US6134461 A. U.S. Classification 600/345, 600/309; International Classification G01N27/327, A61B5/00, C12Q1/00, G01N33/487; Cooperative Classification C12Q1/001, A61B5/14865, C12Q1/006, A61B5/14542, A61B5/01, A61B5/14546, A61B5/1486, A61B5/14735, A61B5/14532; European Classification A61B5/145G, A61B5/1486B, G01N27/327B, C12Q1/00B, C12Q1/00B6B. Oct. 2000 (cit. on p. 10).
- [Sch] Gary Scheiner. *2016 Blood Glucose Meter Comparisons* (cit. on p. 2).
- [Sof17] CTAPP Software. *Glimp*. Oct. 2017 (cit. on pp. 12, 15).
- [Ve] Pierre V and evenne. *Libre Data Interpretation (continued - and probably final for parameters)* (cit. on p. 13).
- [Wil+91] D R Williams, P M Clark, N E Day, et al. „Impaired glucose tolerance and height.“ In: *BMJ : British Medical Journal* 303.6810 (Nov. 1991), p. 1134 (cit. on p. 28).
- [Ilk14] Ilka. *Freestyle Libre - Blick ins Innere*. Nov. 2014 (cit. on p. 11).