Tim Vigers

Final Project Description

March 19, 2020

*cgmanalysis Shiny App*

Domain Description

Continuous glucose monitors (CGMs) are devices that people with diabetes wear which measure blood glucose (BG) every 5 – 15 minutes. Use of CGM has increased dramatically over the past 15 years or so, because they allow physicians to monitor free living glucose patterns throughout the day and provide significantly more information than traditional fingerstick BG measurements. In some age groups, as many as 50% of patients regularly wear a CGM1.

However, there are several companies competing in the CGM market (Dexcom, Abbott, and Medtronic seem to be the most popular) and each company has gone through multiple development cycles. Each company also reports different glucose summary statistics (e.g. time in range 70 – 180 mg/dL vs. time in range 70 – 140 mg/dL), and the algorithms for their analyses are proprietary. This can be frustrating for diabetes researchers and physicians, as it makes comparing patients across devices difficult.

As part of my work at the Barbara Davis Center (BDC), I’ve developed an R package for cleaning exported CGM data and generating summary statistics2. The idea is that because the package is open source and works on all the major CGM systems, researchers and physicians can see how the summary statistics are calculated and know they are computed the same way across all devices. However, using R is a rather large barrier to many people who would otherwise find the package useful, and the plots it currently generates are not very good. So, I would like to convert the R package into a Shiny app that provides diabetes physicians and researchers with interactive visualizations and summary statistics, in order to make it more useful to those without programming experience.

Dataset

One of the physicians I work with at the BDC has provided me with 51 Dexcom, 51 Libre, and 50 Medtronic CGM exports from his research participants. Each CGM export is a small CSV spreadsheet, and all together the data only takes up 375.4 MB. We are working on getting additional data from the Senseonics system as well, although this may not happen in time for the final project.

Data Quality and Cleaning

This dataset should not require much cleaning at all. Occasionally CGMs will stop recording for various reasons (the wearer’s movement, lack of calibration, etc.) which leads to missing data “gaps.” There are a few approaches to handling these gaps that I would like to include in the app. Some researchers use linear interpolation (although this is only recommended for shorter gaps), some delete the full 24 hours containing data gaps longer than a specified threshold, and some just ignore missing data entirely. Also, the first couple of hours of CGM data are often inaccurate, as the device needs time to calibrate. So, some researchers will throw out the first 4 – 12 hours of data. My plan is to include these options for trimming and handling missing data when people upload their data to the app.

There is some uncertainty in all CGM data, as blood sugar is measured with error and there is some variation in the accuracy of these devices. I would like to include some sort of confidence interval in my visualizations based on the accuracy of each device. However, this may be difficult to implement considering that the true accuracy of these devices is difficult to study, and the corporations involved obviously have a vested interest in keeping the details of each system secret. If I’m not able to include accuracy information, I would at least like to use a channel such as color to encode the device attribute.

Integration and Deriving Attributes

Virtually all of the attributes I hope to visualize will be derived from the raw CGM data, because the exports are essentially lists of timestamps and sensor glucose (SG) values. This part should not require much work though, because the R package I’ve already written generates all the necessary summary statistics. What will probably be the most difficult aspect for me is making the visualization interactive and intuitive. There are obviously far too many variables to plot all at once, but I would like it to be easy for users to switch back and forth between different views in a way that captures all the important summary variables (see below).

One of the most important features of the app will be the ability to upload multiple CGM exports from multiple different devices without manually cleaning the data. So, the integration of multiple datasets is essential to this project. The trick to this part is writing code such that the app can recognize which columns contain the necessary information without input from the user. Again, this is already included in the R package and shouldn’t be too difficult to adapt to Shiny but is something that I will need to keep updated as export formats change.

Data Abstraction

With this project I will essentially be working with two different kinds of table. The first is the raw CGM data (dexcom\_example.csv), and the second is a table of attributes derived from multiple raw data files (dexcom\_results\_example.csv). The CGM exports are simple time-varying tables, which can be reduced down to 3 fields (dexcom\_example\_cleaned.csv): subject id, time, and SG. The summary table is also a static flat table, where each row represents a single CGM export file, and each column is a different derived attribute from the raw data. All of the derived attributes are continuous, such as average SG, standard deviation of SG, etc. (see Table 1 for more detailed descriptions of the most commonly used attributes).

If there is time, I would also like to use some sort of clustering algorithm to group patients by similarity of their glucose profiles. This should not be too difficult to implement but determining the best clustering algorithm may end up being a bottleneck. However, if I am able to include it, this variable could be either categorical or continuous, and could potentially be visualized using a scatterplot (for example if I use k means clustering).

Task Descriptions and Usage

Here are the most important tasks I would like users to be able to perform, and the actions they would need to take:

1. Lookup: Users should be able to look up target extremes like which subjects have the highest average SG, lowest percent time in range, etc.
   1. This will most likely be in the form of a sortable summary table like dexcom\_results\_example.csv. I think there will be a “summary statistics” tab, which is laid out exactly like dexcom\_results\_example.csv. Users will be able to click each column name to sort the table by that attribute.
2. Browse/identify: I plan on plotting the aggregate glucose profile (AGP) for each participant on a single plot and allowing the user to interact and highlight specific participants. Basically, the AGP plots the average SG at each time of day across multiple days of CGM wear (see AGP\_example.pdf). Currently each participant gets their own separate AGP plot but seeing them all in the same place and being able to browse the profiles would be hugely useful.
   1. Once the user has uploaded all the relevant files, the first visualization they will see is the AGP for the whole cohort, with a stacked bar chart for full cohort time in range below. The full cohort AGP will be a dark line, and each individual will be a transparent line. The more files uploaded the more transparent individual lines will be, in order to reduce clutter. Highlighting an individual’s line will bring up their summary statistics and individual stacked bar chart, so the user can see more information and compare them to the overall cohort.
3. Compare: This is also related to the browsing action, but I would like users to be able to select two (or possibly more) participants and see their AGPs on the same plot, in order to make it easy to visually compare their glucose trajectories.
   1. When two or more individual lines are selected (hopefully by just clicking on the lines), the unselected lines will disappear to facilitate easier comparison between the selections. If two lines are selected, it would be interesting to plot the difference between them as well, but I’m still undecided on the best way to implement this. It might require limiting selection to two lines. This is something I will likely have to experiment with quite a lot.
4. Summarize: This task will be done automatically when the export files are uploaded and presented in a summary table. I would also like users to be able to download the summary table for further statistical analysis not included in the visualization. I would like to talk to physicians more about this, but it might also be useful to include summaries of the summary statistics for the full cohort (e.g. overall average MAGE).
5. Derive: The ability for users to derive their own summary statistics will be fairly limited, but I think there are two important options that users should be able alter themselves. First, some of the summary statistics are generated for “daytime” and “nighttime.” By default, nighttime is defined as 11pm – 6am, but this may need to change depending on the population (e.g. pediatric participants probably go to bed much earlier). Also, users may want to define specific SG ranges not included by default, so they should be able to set a lower and upper bound and derive their own time in range statistics.
   1. One way to plot time in range for every participant is to use small multiples of bar charts (one for each range of interest). I think there will be a separate “time in range” tab with these plots. On this tab there will be a button called something like “add custom range.” When users click this, they will be able to define the lower and upper bound, and the new plot will be added to the small multiples. They will also be able to remove ranges that aren’t of interest (e.g. in some studies it’s possible that nobody will spend any time below 70 mg/dL). The small multiples can also be shown for daytime, nighttime, or overall. I think this might be easiest to implement using another tab selector.

Initial Design Ideas

See design\_ideas.pdf.

**Table 1: Summary Measures**

|  |  |
| --- | --- |
| **CGM Variable** | **Definition** |
| percent\_cgm\_wear | The number of sensor readings as a percentage of the number of potential readings (given time worn). |
| average\_sensor | Mean of all sensor glucose values |
| estimated\_a1c | Estimated HbA1c based on the equation: (46.7 + average glucose in mg/dL) / 28.7 [1] |
| gmi | Glucose management indicator based on the equation: 3.31 + (0.02392 × average glucose in mg/dL)3 |
| q1\_sensor | First quartile sensor glucose value |
| median\_sensor | Median sensor glucose value |
| q3\_sensor | Third quartile sensor glucose value |
| standard\_deviation | Standard deviation of all sensor glucose values |
| cv | Coefficient of variation of all sensor glucose values (SD/mean) |
| min\_sensor | Minimum of all sensor glucose values |
| max\_sensor | Maximum of all sensor glucose values |
| excursions\_over\_\*\*\* | The number of local glucose peaks with an amplitude greater than \*\*\* mg/dL |
| min\_spent\_over\_\*\*\* | The total length of time that sensor glucose was at or above \*\*\* mg/dL |
| percent\_time\_over\_\*\*\* | Minutes spent above \*\*\* mg/dL, as a percentage of the total time CGM was worn |
| avg\_excur\_over\_\*\*\*\_per\_day | The number of glucose peaks above \*\*\* mg/dL averaged per 24-hour period of CGM wear |
| min\_spent\_under\_\*\* | The total length of time that sensor glucose was at or below \*\* mg/dL |
| percent\_time\_under\_\*\* | Minutes spent below \*\* mg/dL, as a percentage of the total time CGM was worn |
| min\_spent\_70\_180 | Minutes spent in the range 70 – 180 mg/dL (inclusive) |
| percent\_time\_70\_180 | Minutes spent in the range 70 – 180 mg/dL (inclusive), as a percentage of the total time CGM was worn |
| daytime\_\*\*\* | \*\*\* of all sensor glucose values during specified daytime hours |
| nighttime\_\*\*\* | \*\*\* of all sensor glucose values during specified nighttime hours |
| auc | Approximate area under the sensor glucose curve, calculated using the trapezoidal rule |
| r\_mage | MAGE calculated according to Baghurst’s algorithm |
| j\_index | Calculated based on the equation: 0.324 × (average glucose in mg/dL + standard deviation of glucose levels)^24 |
| conga | Continuous overall net glycemic action, default n = 1 hour4 |
| modd | Mean of daily differences |
| lbgi | Low blood glucose index |
| hbgi | High blood glucose index |

References

1. DeSalvo DJ, Miller KM, Hermann JM, et al. Continuous glucose monitoring and glycemic control among youth with type 1 diabetes: International comparison from the T1D Exchange and DPV Initiative. *Pediatr Diabetes*. 2018;19(7):1271-1275. doi:10.1111/pedi.12711

2. Vigers T, Chan CL, Snell-Bergeon J, et al. cgmanalysis: An R package for descriptive analysis of continuous glucose monitor data. *PLOS ONE*. 2019;14(10):e0216851. doi:10.1371/journal.pone.0216851

3. Bergenstal RM, Beck RW, Close KL, et al. Glucose Management Indicator (GMI): A New Term for Estimating A1C From Continuous Glucose Monitoring. *Diabetes Care*. 2018;41(November):dc181581. doi:10.2337/dc18-1581

4. McDonnell CM, Donath SM, Vidmar SI, Werther GA, Cameron FJ. A Novel Approach to Continuous Glucose Analysis Utilizing Glycemic Variation. *Diabetes Technol Ther*. 2005;7(2):253-263. doi:10.1089/dia.2005.7.253