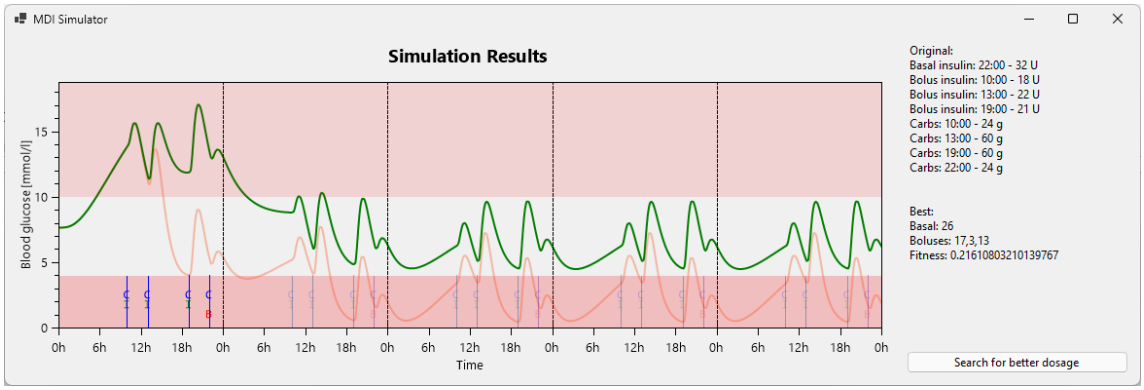
Martin Janiczek

Taking my diabetes treatment into my own hands

2024-07-23

First of all, this blogpost is kinda long. Let me prove to you reading it *will* actually have some payoff:



OK, now that you'll stay, let's start from the beginning...

I'm a Type 1 diabetic. This means my pancreas doesn't produce insulin (which allows cells to use blood glucose for energy) and I have to provide it externally.

This is a finnicky process, because you need to balance your glucose in the right "zone" - not too high (hyperglycemia, >10 mmol/l, is a long-term danger to your body) and not too low (hypoglycemia, <4 mmol/l, is a short-term danger to your body). If it's too high, you need to inject insulin, and if it's too low, you need to eat some sugars.

A commonly used metaphor for this is flying a plane. There are games illustrating the process as well: click blue button, bird flies down, click orange button, bird flies up. Too high bad, too low bad, just right good.

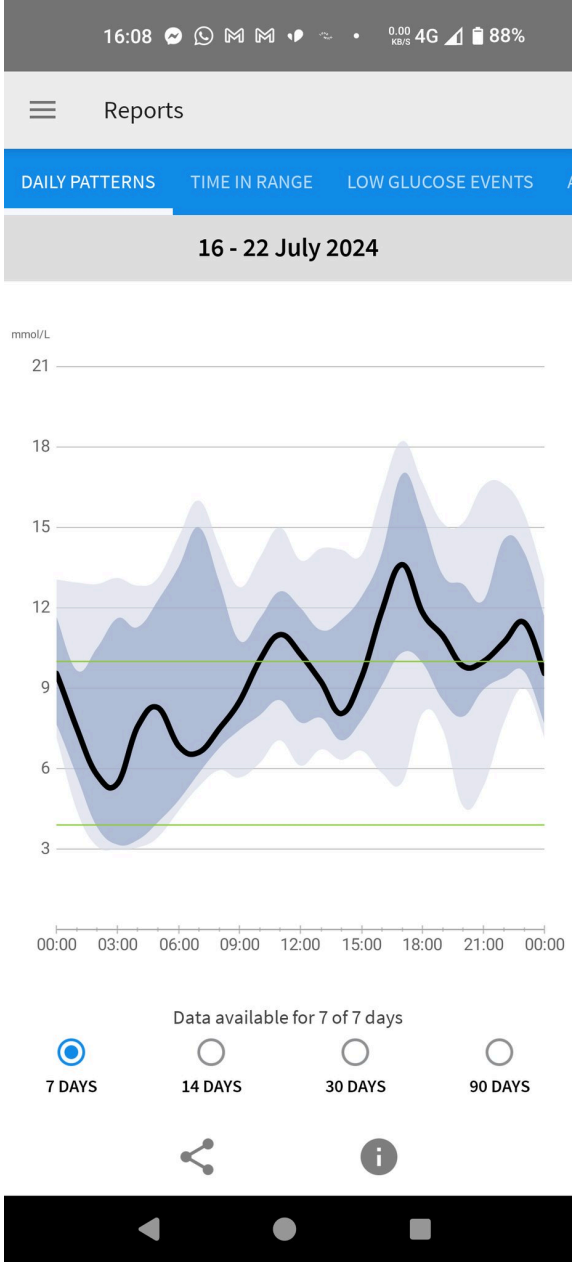
The issues with manually managing this process you've inherited from your douchebag pancreas are manifold:

- the insulin doesn't act immediately, there's around 20min delay (depending on the brand) before your blood glucose goes down
- the *food* doesn't act immediately, there's around 20min delay (depending on the food!) before your blood glucose goes up
  - simple sugars (eg. fruit) are faster (and stop acting faster as well)
  - complex sugars (rice / potatoes / ...) are slower. You usually want your blood glucose as constant and flat as possible. The worst thing you can do is to go from hypo to hyper to hypo to hyper in huge amplitude swings.
  - fats also somehow affect the digestion of sugars. I can't be bothered to remember how, I just ignore them honestly.
- injecting insulin ~15min before you start eating would do *wonders* for neutralizing the BG spike, the issue is, nobody does it, because what if you then get a smaller serving at the restaurant or it gets delayed? What if you get called somewhere urgently and can't finish your meal? People usually inject right before the meal / after the meal as a result.
- there's no generic formula (that I know of) for estimating how much will a gram of sugar increase your blood glucose, nor how much will an unit of insulin decrease your blood glucose. *It's all vibes.*
- mathematical models *do* exist but you need to find your body's parameters - I'll get to it below!
- the body has its own emergency reserves of glucose, which it can sometimes decide to use (though beware, this system turns off when

you're drunk), so *maybe* the Snickers bar you just ate to save your life wasn't actually needed anymore, and you end up with a hyper

- you're not quite yourself during a hypo (you get slower, dumber, I've heard of people getting stuck in thought loops in front of an open fridge), and so even though you intellectually know you just ate enough to get back into the correct levels *eventually*, your brain is screaming at you **"EAT! YOU'RE DYING! I AM LOW ON SUGAR *NOW!!*"** and in my case this leads to overcompensating quite knowingly and willingly. *Yeah one more yoghurt can't hurt.* Well...
- insulin doesn't work when eaten, you need to inject yourself with it (though nowadays we do it under skin, not into veins, I sure am glad I'm not living 50 years ago) or inhale it. Since injections aren't the most pleasant thing in the world, the dosage is usually limited to 4x a day (to counteract the three main meals + a long-acting different type of insulin once a day), even though you'd be more stable if you injected more often, with smaller doses.
    - (no experience with inhalations here, so I'll skip this)
    - (insulin pumps do exist, I'll mention those briefly in a moment... maybe)
- measuring your blood glucose level is painful if you are using test strips and need to prick your fingers to provide a blood drop, so measuring your sugar is usually limited to 4-6x a day – again, even though it would be better to have more data points.
    - this is less of a problem nowadays with Continuous Glucose Monitoring systems like Freestyle Libre, which you install into your arm once every 14 days and get a measurement every minute through Bluetooth to your phone
- things like physical effort, illness, stress, *heck, even seasons of the year* do all affect how your body behaves and reacts to sugar / insulin
- there's this damn thing called dawn phenomenon that some diabetics, me included, experience: in the morning your sugar will just start going up and up. If you wanted to sleep in on the weekend, well tough luck, you're now in the 15 mmol/l range.

I hope this incomplete list gives you an idea of how wonky the process of trying to make your blood glucose stay in the right levels is.



My treatment is usually: keep the Freestyle Libre app on my phone open as much as possible and when I see my BG's getting high, I inject a small amount of insulin. How much? No idea. *IT'S ALL VIBES.*

But sometimes the app is yelling at you: "you're at 15 mmol/l for an hour now, idiot!" and you just don't (want to) pay attention. Alert fatigue is a real thing.

I have some recommended insulin dosages that we've settled on with my diabetologist, whom I visit every three months. So my regimen can look like:

- Breakfast: inject 18 units, eat 24g of sugars
- Lunch: inject 22 units, eat 60g of sugars
- Dinner: inject 21 units, eat 60g of sugars
- Before sleep: inject 32 units of long-acting insulin

And then I see my diabetologist, she looks at the 7d / 14d / 30d averages and says “maybe you can try fixing the 15:00 hypers you’re getting quite regularly, by injecting more insulin before lunch. You should also lose weight, when you started coming here you had 80kg, now you’re a centurion. Like seriously, WTF. OK cool bye, see you in 3 months!”

Lovely.

If you can’t tell, the thing that irks me the most about the whole thing is: [illegible]. I’d seriously appreciate it if my diabetologist used a model, or a simulation of some kind, got my body’s parameters there somehow, and found the improvements to my schedule *that way*. Maybe she has some expert knowledge in her head but from my perspective it’s all guessworks. Err, I mean [illegible].

And this is where my programmer mind comes in.

There are people who take **insulin pumps** (which provide insulin in very small very frequent doses and are ~permanently injected into your body, but are otherwise dumb as a brick) and combine them with **continuous glucose monitors,** and make the glucose measurements inform and control the pump. This is called “closed loop” or “artificial pancreas”, and getting one officially is very hard or impossible: not FDA approved yet / you need to be part of an university study to get one / … It’s one of those things that “will be here in 5 years”, *they say every year for the past 30 years.*

> Aside: I try not to be too butthurt about it: CGMs have just recently started being available and even fully sponsored by the Czech health insurance companies, and having a 1440-datapoints-a-day graph is a *MASSIVE* improvement compared to pricking your finger 4x a day and getting 4 datapoints for your blood glucose graph with nothing in between. So, the artificial pancreas is slowly coming. Unlike nuclear fusion.

The most prominent of these people hacking their devices together, in my social bubble at least, is Scott Hanselman, the Microsoft programmer guy. (Check out his talk.) He’s a T1DM as well and has been promoting the #WeAreNotWaiting initiative where people take their own pump and their own CGM and hack them together despite the healthcare companies’ pleas that it’s not approved and not safe etc. #TheyAreNotWaiting.

And that is really inspirational.

I don't have a pump myself (and to have a chance of getting one I'd first have to find another diabetologist, which makes this into a *"too much work, can't be bothered"* issue for me), so I can't currently do quite what they are doing, but I can go with the high-level idea and #NotWait in my own way.

A few days ago I was fumbling down the stairs to our kitchen at ~3:00 in the morning to fix my hypo. (Night hypoglycemias are *especially* bad: what if you don't wake up?) On the stairs I had the thought: why the hell is there no app into which I'd put my past X blood glucose values, my usual daily schedule, my weight, height, gender, age, whatever, and it would let me play with some kind of prediction (interactively!) and find good dosages / meal times / injection times? Then I would have a potentially good target to get to, and over the course of a few days I could gradually adjust my real dosage to that level and see how it behaves, and hopefully stay in the 4-10 mmol/l range much more easily.

Why doesn't such a thing exist?

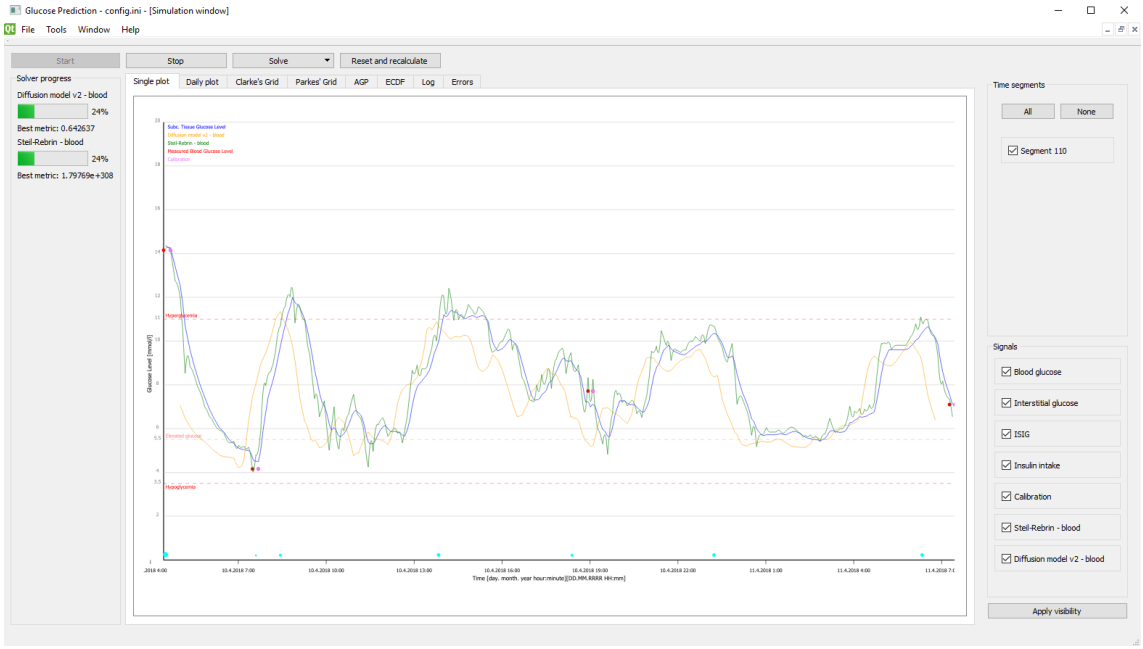
(Coincidentally I've also started chatting about this and other app ideas with John Pavlick. Thanks for your encouragement, John!)

So I started googling. Turns out there are *many* papers detailing differential equations for a model of how glucose and insulin interact, and how an artificial pancreas could get you to the correct range automatically. The issue is, I DON'T HAVE AN ARTIFICIAL PANCREAS. I have four daily injections. Give me something useful for those!

There's not too much to pick from.

> To be honest, I also don't really know how to translate those differential equations into a simulation algorithm, even if I found a good model. I'm guessing I need to write an ODE solver like Runge-Kutta for the specific set of equations and somehow find *my* specific parameters for it. It's been a while since I studied this in uni.

One of the links led me to diabetes.zcu.cz though, and in particular their SmartCGMS app (open-source too!). From the screenshots it seemed kinda relevant, or at least similar to what I had in mind for my dream app.



So I sent an email to the authors. Found one maintainer on GitHub and wrote them an email detailing my woes and the app I'd love to write, and whether they could point me to some existing software or good papers on modelling multiple-dose-injection treatment (as opposed to a pump / artificial pancreas).

What they gave me was better than I could imagine. (Thanks again, Martin!) They had a wrapper for the core SmartCGMS engine (which contains implementations of some of these models already) for the C# language. The API was quite simple:

- `.Create(...)` - initialize the simulation
- `.Step()` - step one time-unit (typically a minute) in the simulation
- `.ScheduleInsulinBasalRate(double unitsPerHour)` - schedule a (pump, sadly) insulin dosage
- `.ScheduleInsulinBolus(double units)` - schedule a short-acting insulin injection
- `.ScheduleCarbohydratesIntake(double grams)` - schedule food consumption
- `.Terminate()` - stop the simulation

And then there was the current simulation state:

- `.BloodGlucose` - current blood sugar, in mmol/l
- `.InterstitialGlucose` - glucose in your interstitial fluid - let's skip it, not important
- `.InsulinOnBoard` - how much insulin is still left to be absorbed
- `.CarbohydratesOnBoard` - how much sugar is still left to be absorbed

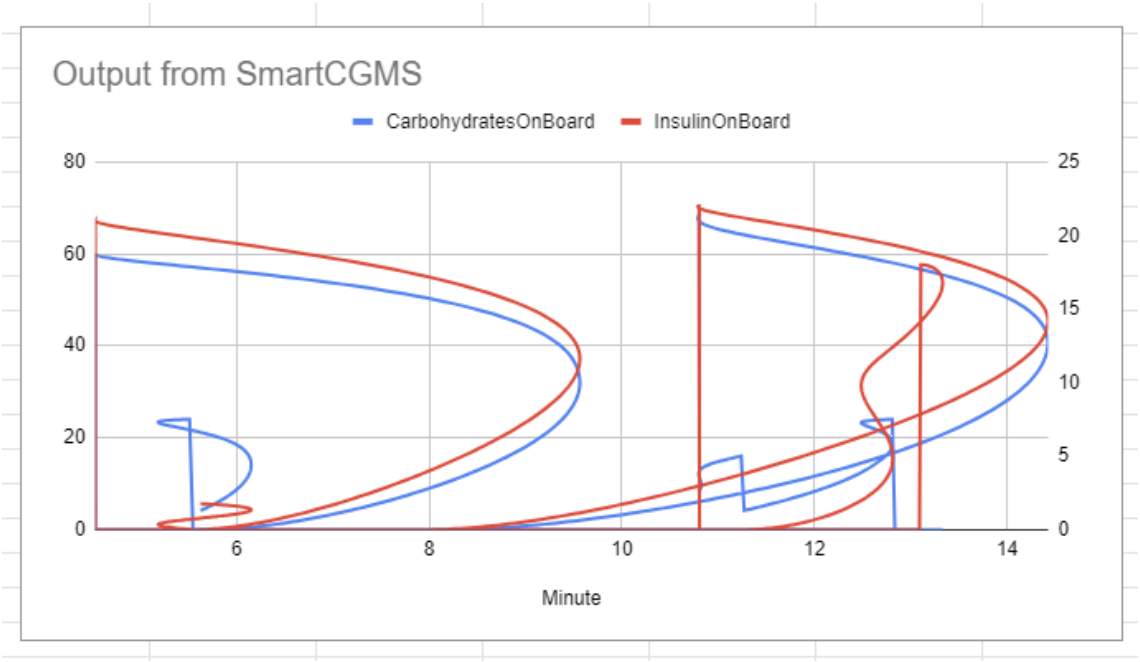
As you can see, with some caveats this would let me make a simulation for my daily schedule.

So a few moments later I had something like this (in C# but here presented as Elm)

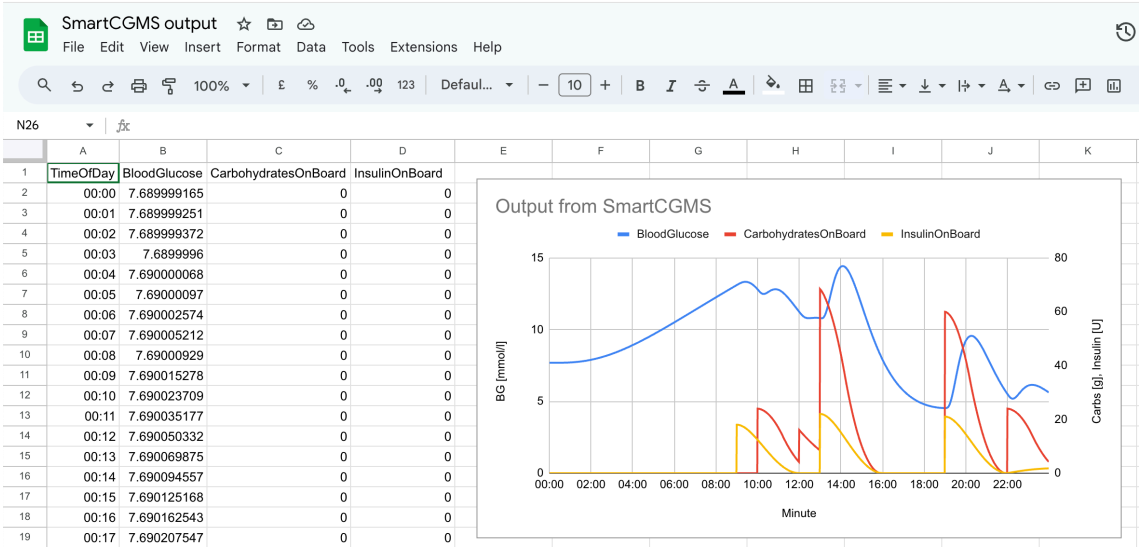
```
type IntakeType
  = BasalInsulin -- long-acting
  | BolusInsulin -- short-acting
  | Carbs        -- fooooooooooooood!

type alias Intake =
  { intakeType : IntakeType
  , amount : Float
  , timeMinutes : Int
  }

type alias Input =
  { basalInsulin : Intake
  , bolusInsulins : List Intake
  , carbs : List Intake
  }

type alias OutputRow =
  { minute : Float
  , bloodGlucose : Float
  , carbohydratesOnBoard : Float
  , insulinOnBoard : Float
  , interstitialGlucose : Float
  }

simulate : Input → Int → List OutputRow
simulate input days =
  ...

mySchedule : Input
mySchedule =
  { basalInsulin = Intake BasalInsulin (22 * 60) 32
  , bolusInsulins =
      [ Intake BolusInsulin (10 * 60) 18
      , Intake BolusInsulin (13 * 60) 22
      , Intake BolusInsulin (19 * 60) 21
      ]
  , carbs =
      [ Intake Carbs (10 * 60) 24
      , Intake Carbs (13 * 60) 60
      , Intake Carbs (19 * 60) 60
      , Intake Carbs (22 * 60) 24
      ]
  }

myPrediction : List OutputRow
```

```
myPrediction =
  simulate mySchedule 3
```

Never have I copied the resulting CSV into Google Sheets so fast. Tada:



Oh, wait, that's not it. Cool piece of accidental art though! Now let me use the correct column for the X axis.



Well hot damn. The graph actually makes sense!

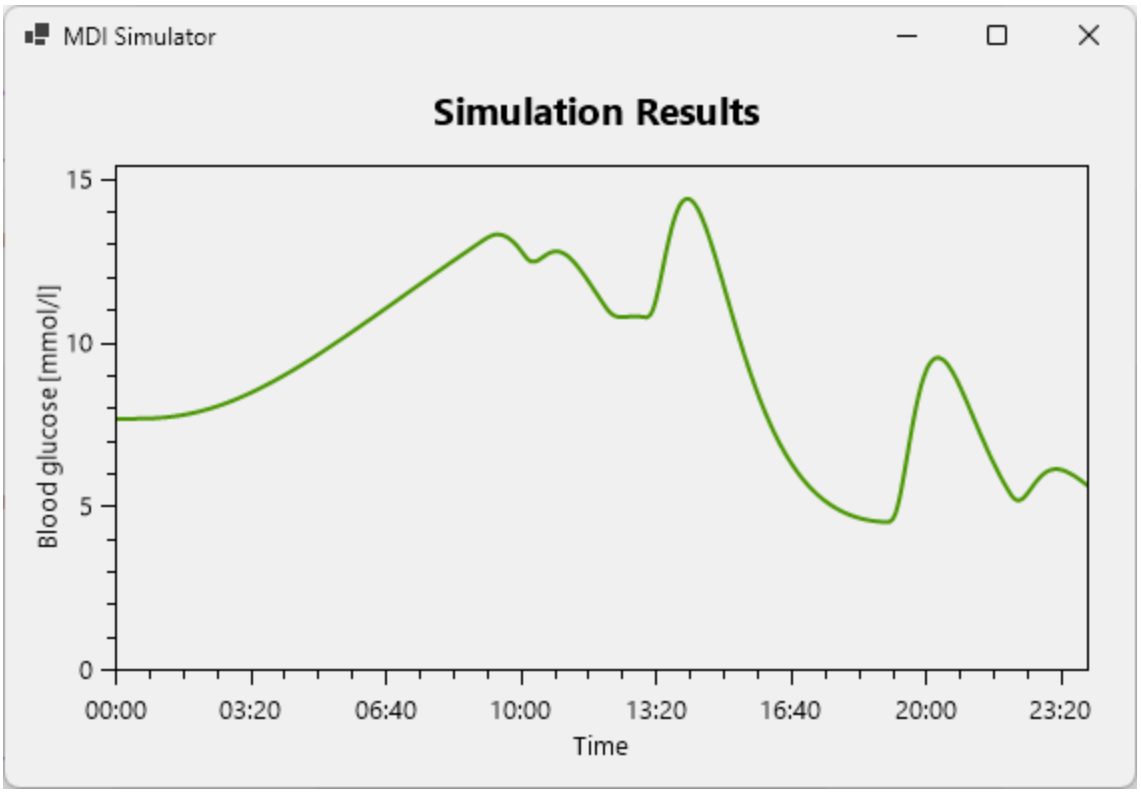
Granted, it's not *me* in the graph but I can simulate *someone* now!

Let's stash away "I need to actually simulate me, *you know" as a TODO for future Martin and continue.*

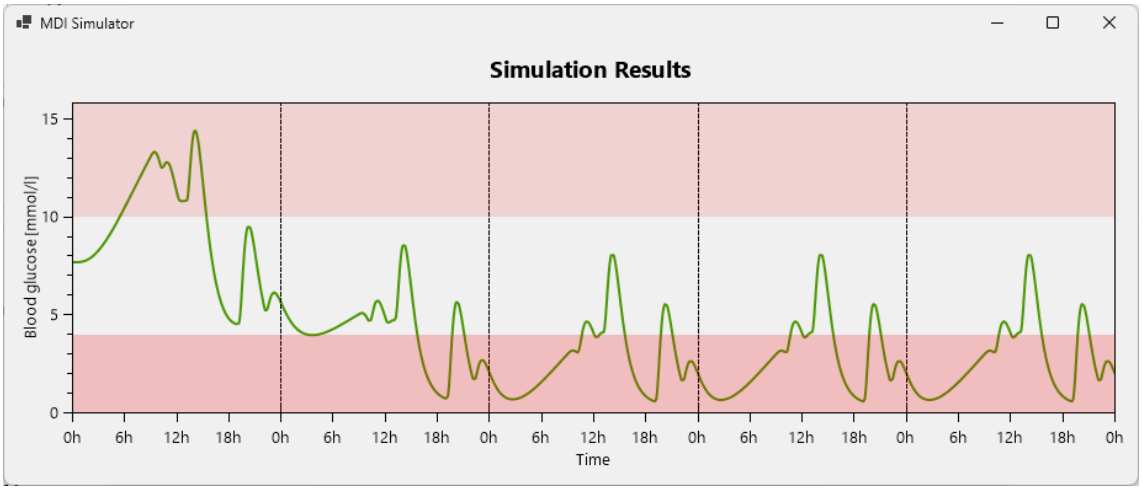
Next I made a Windows Forms application with an OxyPlot chart so that I don't need to write a CSV to a file and manually copy it to Google Sheets.

> Aside: what do you .NET folks use nowadays? MAUI? WPF? Xamarin Forms? It's kinda confusing for an outsider.

This took me a while to figure out (I'm not a C# guy; honestly I've thought about rewriting this into F# instead the moment I had to start learning about event handlers and delegates), but I succeeded:



There's one little issue with seeing just the first day of the simulation though, and that's the fact that I inject my basal (long-acting) insulin at 22:00. So for the majority of the first day the glucose will just be higher because I haven't injected the long-acting insulin yet. So let's simulate more days just to see what happens.



Oh wow, it gets periodic by day 3! Cool!

As you can see I've also added the hypo- and hyperglycemic ranges so that it's clearer where the "Goldilocks zone" lies. As it turns out, my insulin

dosage is absolutely inappropriate for the person simulated by SmartCGMS right now. Hitting such severe hypoglycemia, they'd probably be dead by now (or their liver has to work overtime on dosing that emergency glucose).

OK, well then, now it's just a small step from simulating a hardcoded

```
mySchedule : Input
mySchedule =
  { basalInsulin = Intake BasalInsulin (22 * 60) 32
  , bolusInsulins =
      [ Intake BolusInsulin (10 * 60) 18
      , Intake BolusInsulin (13 * 60) 22
      , Intake BolusInsulin (19 * 60) 21
      ]
  , carbs =
      [ Intake Carbs (10 * 60) 24
      , Intake Carbs (13 * 60) 60
      , Intake Carbs (19 * 60) 60
      , Intake Carbs (22 * 60) 24
      ]
  }
```

to automatically finding a more optimal dosage!

I've opened the NuGet package manager, wrote `genetic` and installed the most popular package: GeneticSharp. Turns out it's *really solid.* It needed me to provide the usual stuff: chromosomes, crossover, selection, population size, termination criteria... and the fitness function.

The fitness function actually deserves a fuller description. It looks like this:

```
fitness : Input → Float
fitness input =
  let
    output : List OutputRow
    output = simulate input 3

    longtermHypos = output ▷ List.takeLast (24*60) ▷ List.count (\r
    longtermHypers = output ▷ List.takeLast (24*60) ▷ List.count (\
    ...

    longtermHyposNormalized = longtermHypos / List.length output
    longtermHypersNormalized = longtermHypers / List.length output
    ...
```

```
  longtermHyposWeight = 15
  longtermHypersWeight = 12
  ...

  weightSum = longtermHyposWeight + longtermHypersWeight + ...
in
( longtermHyposNormalized * longtermHyposWeight
+ longtermHypersNormalized * longtermHypersWeight
+ ...
) / weightSum
```

(Yes I know stuff is computed needlessly here; in the real C# code I'm doing all the intermediate result reuse you wish I did here, but I'm optimizing for understanding instead here.)

Turns out I care about many things. The following list evolved gradually but I'm only giving you the final version:

- minimize # of hypoglycemic readings in the (stabilized) last day
- minimize # of hyperglycemic readings in the last day
- as small amplitude between min and max glucose reading as possible in the last day
- minimize the sum of bolus insulin dosages
- minimize the basal insulin dosage
- minimize # of hypoglycemic readings in the stabilization phase (first 2 days)
- minimize # of hyperglycemic readings in the stabilization phase

Note that I don't care about all of those equally. I've actually sorted the above list by priority: long-term hypos are the most urgent, the temporary hypers while the system settles I care about the least. I'm encoding that via the weights. Each of the normalized measurements is a number `0..1`, which then gets multiplied by the weight.

I'm not quite sure whether this is the right way to encode multiple concerns into a single number, but it's the best I could come up with without consulting math books, and it seems to work well. At least I couldn't find a case where an input with lower (better) fitness was less preferable to me (according to my brain's fuzzy intuition) than another input with higher (worse) fitness.

If I was able to construct all the values and sort them, I'd probably do something like

```
allCombinations
  |> List.sortBy (\output ->
```

```
        [ longtermHypos output
        , longtermHypers output
        , amplitude output
        , ...
        ]
      )
```

(that is, I can order outputs pairwise), but that's not how the genetic programming libraries work. I believe explicitly not going through the whole space is one of their very top priorities :)

> Aside: how many inputs are there? This to me looks like permutations with repetitions (order does matter), `n^r`, so in my case `injectionPossibilities ^ injectionCount`, and for my specific example schedule, `51^4` (assuming I can inject `0..50` units of insulin). That's around 6.7 million.

So this could be bruteforceable. But I have to run the whole simulation inside the fitness function, and it takes around a second or two.

At least it's parallelizable then! (And believe me, I'm making use of my 16 cores.)

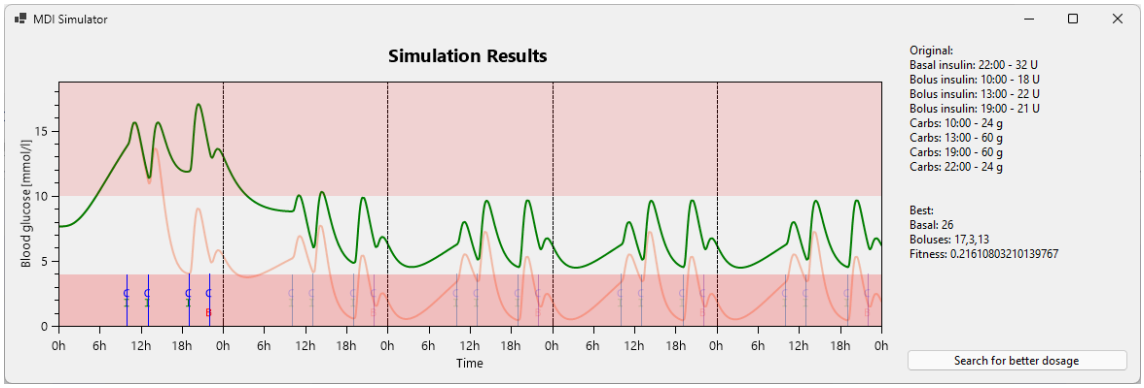
Given it's a pure function (basically... the results vary around the 10th decimal digit - negligible), I can memoize the fitness function for the input of four ints. The genetic algorithm ends up repeating some guesses quite a lot near the end of the simulation, so this actually saves a lot of time.

Writing a memoization function in C# was a breath of fresh air, coming from the pure FP world of Elm:

```
private Dictionary<List<Intake>, double> fitnessCache = new(new Intak

// ...

if (fitnessCache.TryGetValue(amounts, out var cachedFitness))
    return cachedFitness;

// ...

var fitness = Fitness(newInput);
fitnessCache.Add(amounts, fitness);
return fitness;
```

Amazing what you can do with a bit of mutation. Yeah I'll go return my Elm badge now.

So, with this fitness function created, I can now run the genetic algorithm. Let's add a button and some more info to the UI, and start it off!



WELL HOT DAMN. It has optimized the insulin intakes and the patient (again, sadly not *me*) is now stable inside the 4-10 mmol/l range.

That's really magical. I feel like I've solved diabetes. Of course it's not as simple. There will always be unexpected things and changes you need to react to. You'll have hypers, you'll have hypos and it's OK. But still. It managed to squeeze the blood glucose into the correct range.

So, what do I need to make this useful *to me?*

I feel like interactivity would go a long way. Being able to add injections or meals, move them up and down, left and right, and seeing the graph change based on that, would give the diabetic much better understanding than the *"OK I'm gonna inject more before lunch today and see what happens two hours from now"* feedback loop, or even worse, the *"consult a doctor every three months"* one. Did I mention I'm a big fan of short feedback loops?

Of course, it needs to simulate *me* instead of somebody else. I can't use these optimized dosages because my body reacts differently. I need to consult this with the SmartCGMS folks, but the process will likely involve me downloading my historical blood glucose data off my Freestyle Libre sensor and somebody somewhere fitting the model parameters to that data. The math escapes me but it can be done.

Another issue is that the algorithm is optimized for pumps, and my basal (long-acting) insulin will need to be tracked into the model a bit differently. Right now it's as if I was injecting 1/24-th of the dosage every hour, instead of the full dosage once a day. But once the specific insulin brand and its behaviour is tracked in the software, I will be able to call `.ScheduleInsulinBasal(double units)` and all should be well (and more precise), hopefully.

So, there's a bunch of stuff yet to be done and collaborate with the SmartCGMS folks on, but I'm *really* excited by this, and feel empowered taking care of my diabetes better than I thought I could. WE'VE GOT THE TECHNOLOGY! Genetic algorithms and Markov Chains, baby!

What's that? I didn't mention Markov Chains *once* in the article?

Oh yeah, well, I experimented with a bunch of stuff. To end off the blogpost, here's a stupid random walk arriving at a value iteratively. (Change each intake by a random value `-2..+2`, and if the fitness of that tweaked input is better, keep the change, otherwise rollback.)

0:00

Previous Post
Notes from Elm Camp 2024

Archive Twitter Github