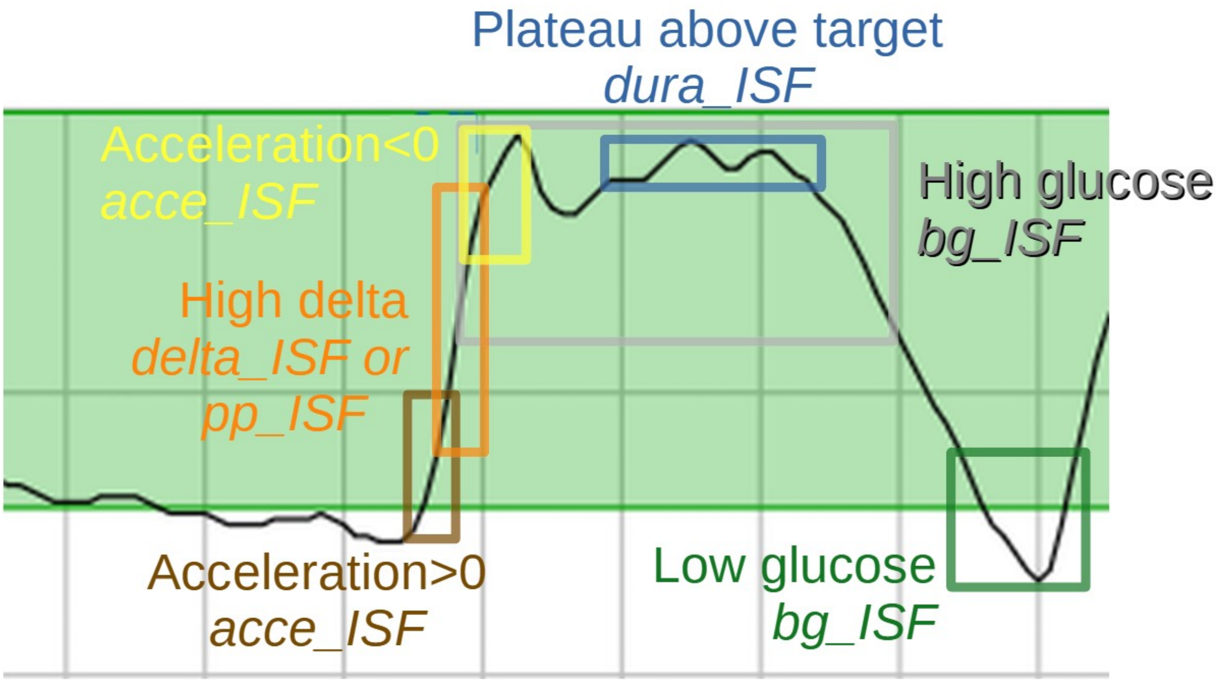
AutoISF – How to start the tuning in general

This document goes into more detail how to find out which of the effects has the strongest influence and by how much it could be changed. It covers some general hints but for specific and more precise understanding of what is going on you should get familiar with using the emulator (documentation to be updated!). The emulator was adapted continuously with exactly this usage in mind.

Understanding the various phases of glucose behaviour

In the schematic picture shown below you can see times when specific behaviours can be expected to dominate. But keep in mind it is only schematic. For instance, the box "Low glucose, bg_ISF" bottom



right could also be labelled "Acceleration>0, acce_ISF". Further is appears to be below target and therefore bg_ISF could play a role to weaken ISF.

As a general hint regarding acceleration taken from mathematics:

- acceleration is positive whenever the curve takes a "left-hand-turn"
- acceleration is negative whenever the curve takes a "right-hand-turn"

In order to find out which effect contributes how much at this point in time you need to take into account the various weights, too. Even though the area in the middle is clearly recognizable as a plateau, i.e. presumably being dominated by dura_ISF, it could still be dominated by bg_ISF if weights distort the picture. For deeper insight you need to check the SMB-tab. However, that explains only the current moment. To get a complete view over a certain time window you can extract the SMB-tab info about the various xx_ISF factors from the logfiles. Look for rows containing strings like these:

```
18:21:35.326  D/APS: [LoggerCallback.jsFunction_log():42]: acce_ISF adaptation is 1.24
18:21:35.327  D/APS: [LoggerCallback.jsFunction_log():42]: bg_ISF adaptation is 0.49
18:21:35.327  D/APS: [LoggerCallback.jsFunction_log():42]: bg_ISF adaptation lifted to 0.61 as bg accelerates already
```

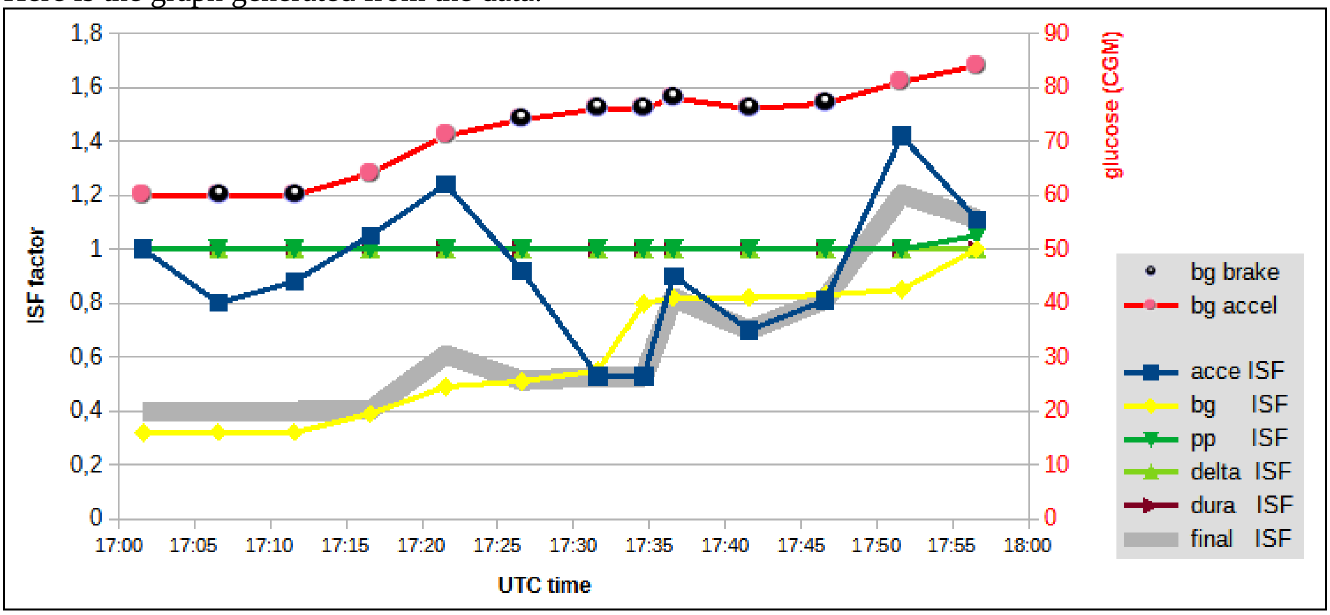
Watch out, this actual extract is from AAPS2.8.2 (I myself have not upgraded yet) and in AAPS3 you need to replace the code ":42]" by ":21]".

Using the emulator to gain first insight

Collecting all that information over a whole time window is very tedious and luckily the emulator can deliver that as a bye product in the form of a CSV-File. In this example I used a 1 hour time window from 18:00 – 19:00 Central European Time which includes the data shown above. This time window translates to 17:00-18:00 UTC time, which is the UNIX time zone used internally by AAPS and equals Greenwich without daylight savings adjustments. The example was from Feb.10th, so one hour difference. Here is an extract of the relevant columns after import:

UTC time	bg accel	bg brake	parab fit correl	parab fit durat	parab fit last-Δ	parab fit next-Δ	auto sens emul	acce ISF emul	bg ISF emul	pp ISF emul	delta ISF emul	dura ISF emul	final ISF	ISF orig	ISF prof	ISF emul
17:01	60						1	1	0,32	1	1	1	0,4	225	90	225
17:06	60	60	1	15	0	-1	1	0,8	0,32	1	1	1	0,4	225	90	225
17:11	60	60	0,98	20	-0,37	-1,09	1	0,88	0,32	1	1	1	0,4	225	90	225
17:16	64		0,93	15	3,19	5,18	1	1,05	0,39	1	1	1	0,4	223,1	90	223,1
17:21	71		1	15	7,19	10,68	1	1,24	0,49	1	1	1	0,61	147,8	90	147,8
17:26	74	74	0,98	15	4,41	3,93	1	0,92	0,51	1	1	1	0,51	175,6	90	175,6
17:31	76	76	0,99	15	1,4	-1,1	1	0,53	0,55	1	1	1	0,53	163,6	90	170,8
17:34	76	76	0,99	15	1,4	-1,1	1	0,53	0,8	1	1	1	0,53	113,9	90	170,8
17:36	78	78	1	15	1,8	1,3	1	0,9	0,82	1	1	1	0,82	109,8	90	109,8
17:41	76	76	0,99	25	-1,13	-2,88	1	0,7	0,82	1	1	1	0,7	109,8	90	128,6
17:46	77	77	0,98	35	-1,09	-2,25	1	0,81	0,83	1	1	1	0,81	109	90	110,6
17:51	81		1	15	4	7	1	1,42	0,85	1	1	1	1,2	74,6	90	74,8
17:56	84		0,98	15	3,8	4,8	1	1,11	1	1,05	1	1	1,11	80,9	90	80,9

Here is the graph generated from the data:



In this example dura_ISF, delta_ISF and pp_ISF all equal 1, i.e. they do not contribute with the exception of pp_ISF at 17:56. In that sense this example is simpler than normal but provides already enough data to start discussing what can be learned from that graph and the table. After importing the CSV-File you can insert an extra column for the final ISF which simply is “profile ISF” divided by “emul ISF”. In standard cases the final ISF is the envelope of the other factors, i.e. the lowest below 1 and the highest above 1 because the extremest factor wins. This is clearly seen in the second half of the time window between 17:21 and 17:46. Between 17:01 and 17:11 bg_ISF lies outside the envelope, but that is because of the limiting autoISF_min=0.4. At 17:21, at 17:36 and at 17:51 there are three more exceptions to the envelope principle because the acce_ISF and bg_ISF were multiplied together.

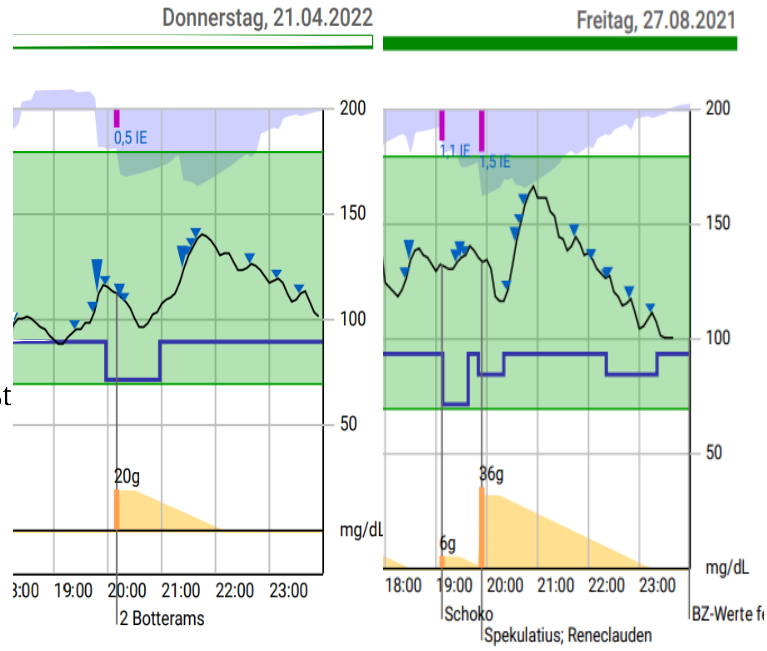
The acceleration business is quite interesting and needs some explaining. Whenever acce_ISF >1 the curve for it is above the line for factor=1 and we have positive acceleration and vice versa. This can also be deduced from the blood glucose line. As a trick that line is plotted twice, namely the regular “red line with red dots “ superimposed by “no line with black dots and no gaps filled” if acceleration is negative. That “bg brake” column was inserted also and contains the formulae `=IF(acce_ISF(row)<1; bg(row)+F$3; "")` which means if acce_ISF<1 then use glucose value, otherwise enter null text, which will not plot. The extra “+F$3” is a trick to initially offset the duplicated BG curves s that can be picked individually while formatting the chart. Later cell F$3 is emptied.

Now, when following the glucose curve there is not always a positive acceleration in every left hand turn like at 17:36. Why is that so? The reason is the fit algorithm. A parabola is defined exactly by 3 data points but that introduces all the “noise” from the CGM. A fit needs a minimum of 4 data points, i.e. 15 minutes of CGM data and smoothens out spikes. At time 17:36 the fits says negative acceleration “fits” better and has a very good correlation of 100% (including rounding effects). Visually it is less obvious but knowing that the last 15 Minutes are used it appears acceptable. Until 17:46 the time window of the fit grows steadily up to 35 minutes and we still have a right hand turn situation and negative acceleration. Therefore the simple schema from the beginning is a good first indication but you really need to check the results of the fit, namely sign, correlation and window length and look at the graph of the factors.

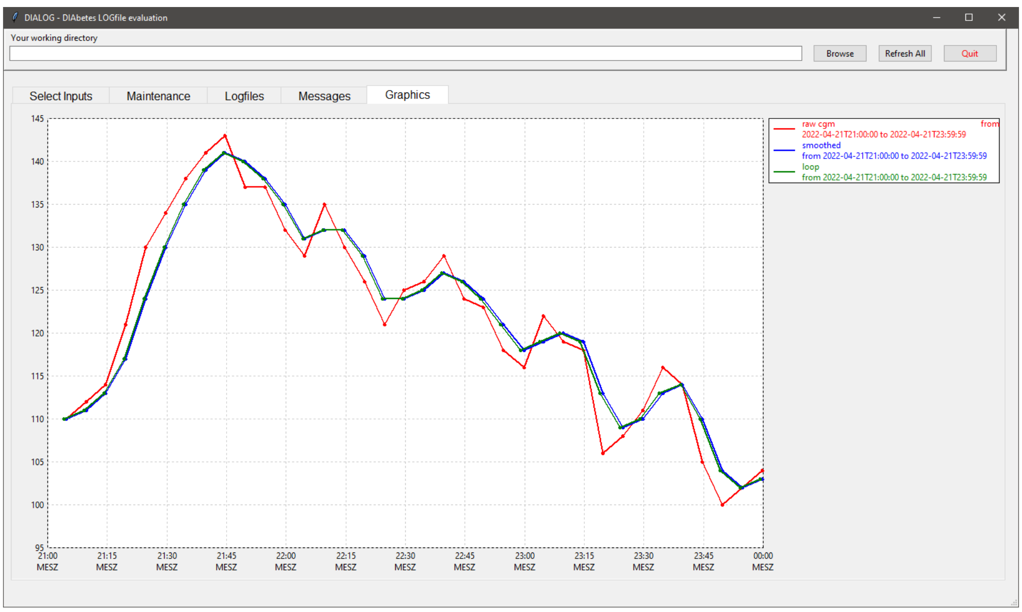
Using the emulator to see all contributions

Here is the process I went through as the very latest study. The problem to be addressed is a pattern happening again and again when glucose goes up and down for no apparent reason. The example I picked was from 21.Apr.2022 after 21:00 hours having a rise after meal and the later decline was overlaid by ups and downs. In this case it was not “Full Loop” but a test to see whether the settings are compatible with regular use of AAPS.

Further right I found the very effect end of August last year while I used autoISF2.1. This probably means the effect was not introduced by using acce_ISF. But let me focus on the recent event.



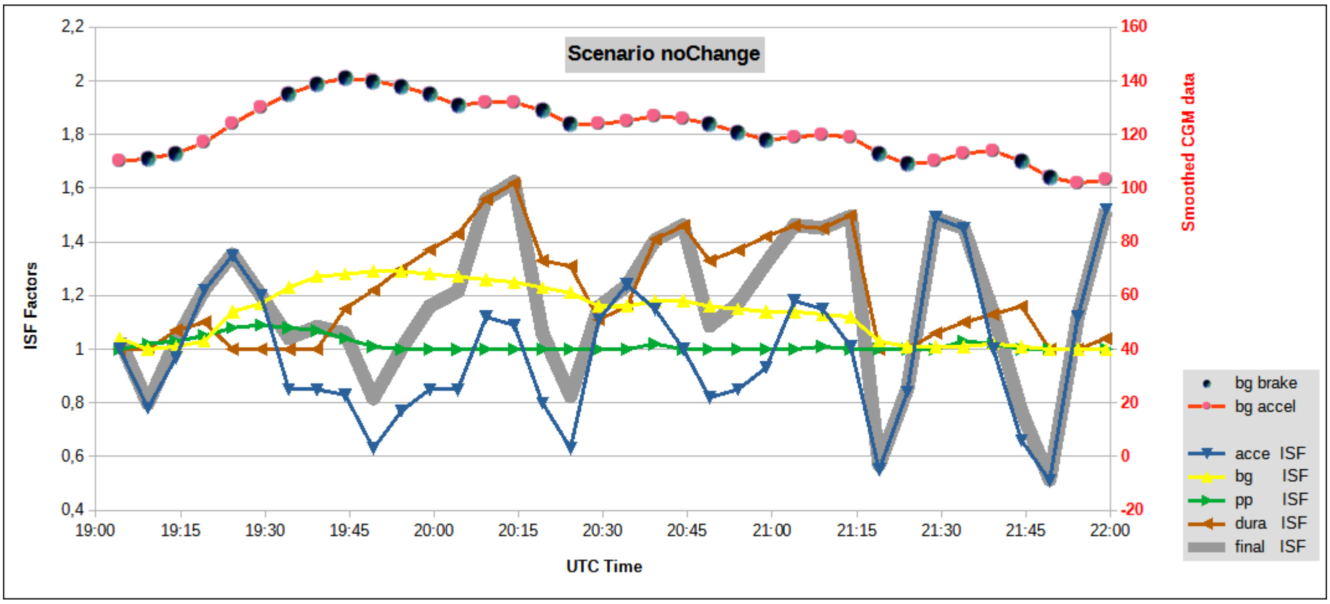
The first thing I did was checking raw cgm versus smoothed values and which value was used by the loop. Here is that comparison showing no obvious problem. Looking at the graph the first option which comes to mind is to strengthen the smoothing, but that



is the last resort I want because it irons out the initial rise and as a consequence the acceleration would detect meal absorption too late and too weak.

I see that each up phase and each down phase lasts for about 20m. With the smoothing delaying things by about 5m and assuming the delay from blood glucose to interstitial glucose around the sensor is 15m that could explain a total of 20 minutes. The loop makes a decision and the feedback from the blood takes 20m and shows a behaviour differing from the loops expectation. Then the loop would take the opposite action, viola. But then the insulin released takes another 10-30m until it arrives in the blood and becomes active. So, this does not look like a probable explanation.

Therefore I ran the emulator with the original settings, i.e. the so-called noChange scenario to allow detailed inspection.



During the whole decline beyond 19:45 dura_ISF would normally take control as in earlier versions of autoISF. Now, however, negative acceleration reduces its full influence. So, bgBrake_weight should be reduced. I ran the emulator with 33% less, but that had little effect. After a few iterations I ended up with reducing bgBrake_weight by 20% and increasing bgAccel_weight by 20%. The summary row at the bottom of that CSV-File shows the incremental insulin likely to be delivered. This is only a rough estimate because the emulator cannot predict a new glucose value, just the change in insulin at each single moment. It needs experience in looping to judge the effect a change has on things to happen downstream. Nevertheless the extra SMB of 0.2U is of the right order to bring glucose down by 20mg/dl. Both settings were at 0.15 on that day and you can see the new values in the VDF-File, which defined the changes in settings:

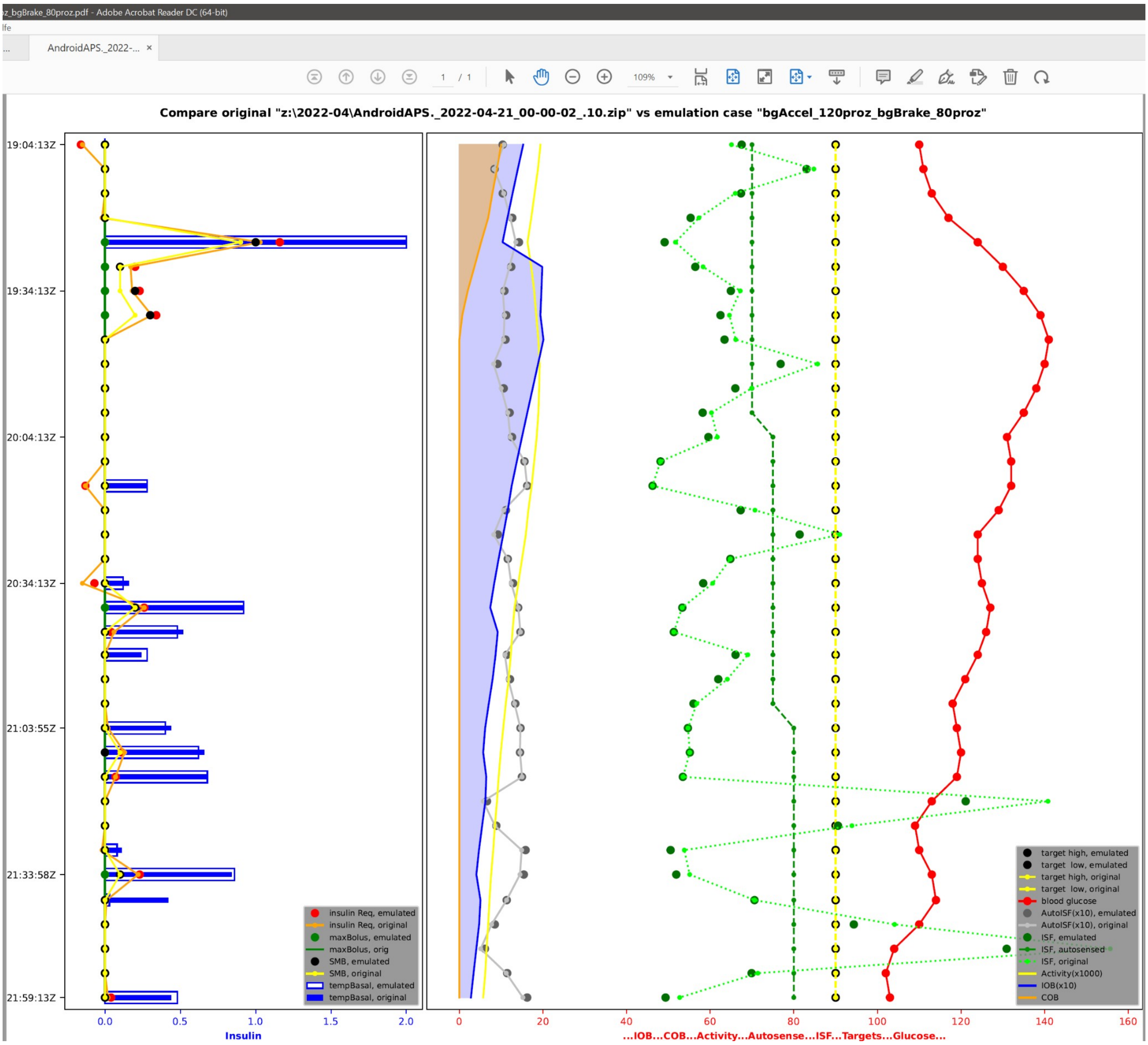
	A	AJ	AK	AL	AM	AN	AO
1							
2		SMB	SMB	TBR	TBR		
3	id	orig	emul	orig	emul		
40	Minimum:	0	0				
41	Maximum:	0,9	1				
42	Totals:	1,7	1,9	0,65	0,61		

AndroidAPS._2022-04-21_00-00-02_.0.zip.bgAccel_120proz_bgBrake_80proz

Sheet 1 of 1 | Default | English (UK) | Average: ; Sum: 0 | 140%

```
profile         bgBrake_ISF_weight      0.12         ### was 0.15
profile         bgAccel_ISF_weight      0.18         ### was 0.15
```

The acce_ISF was strengthened to allow for that incremental insulin early during the initial rise after the start of meal absorption as can be seen in the graphical output of the emulator:



In the left frame of the picture the yellow dots show the original SMBs, the black ones the SMBs in the emulated case. The main changes happen between 19:24 and 19:39. Delivering required insulin early is something BerNie preaches all the time so this change goes in the right direction.

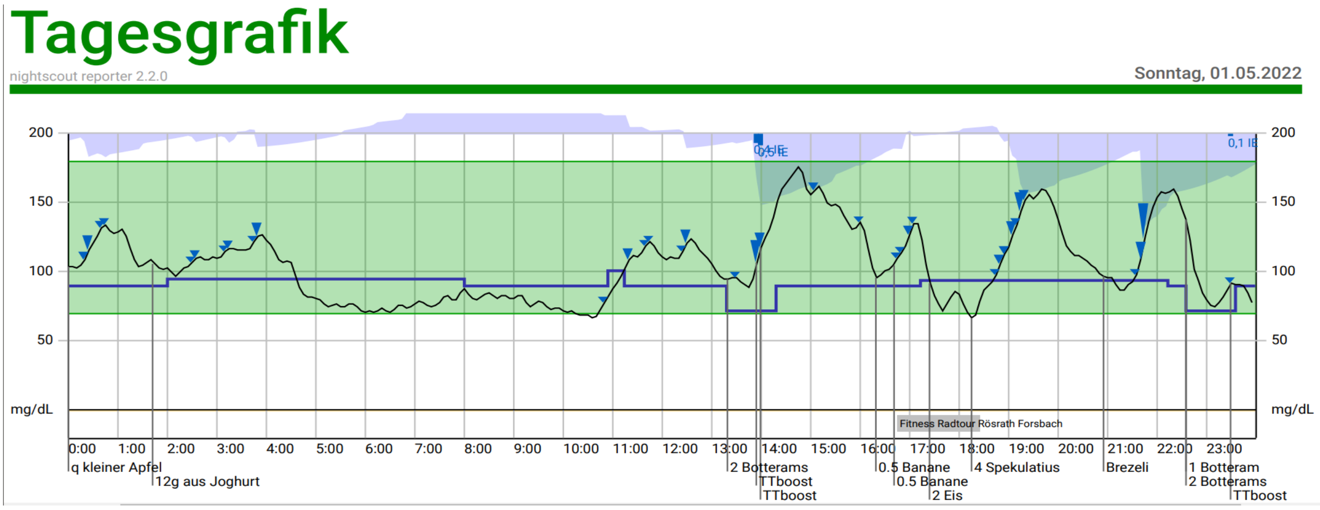
In the right frame of the picture the dark green dots show the emulated ISF, the lighter green ones show the original ISF. During positive acceleration the ISF is somewhat stronger than before and it is weaker during deceleration. Included in that frame is the blue area showing IOB and it is fairly straight and therefore cannot be the reason for the oscillations.

Result of adapting these weights during acceleration and breaking

Did it help? Subjectively things the oscillations are less but still visible. With hindsight that was obvious because the same oscillations were there in the other example on 28.Aug.2021 shown above when the acceleration effect was not yet included in autoISF.

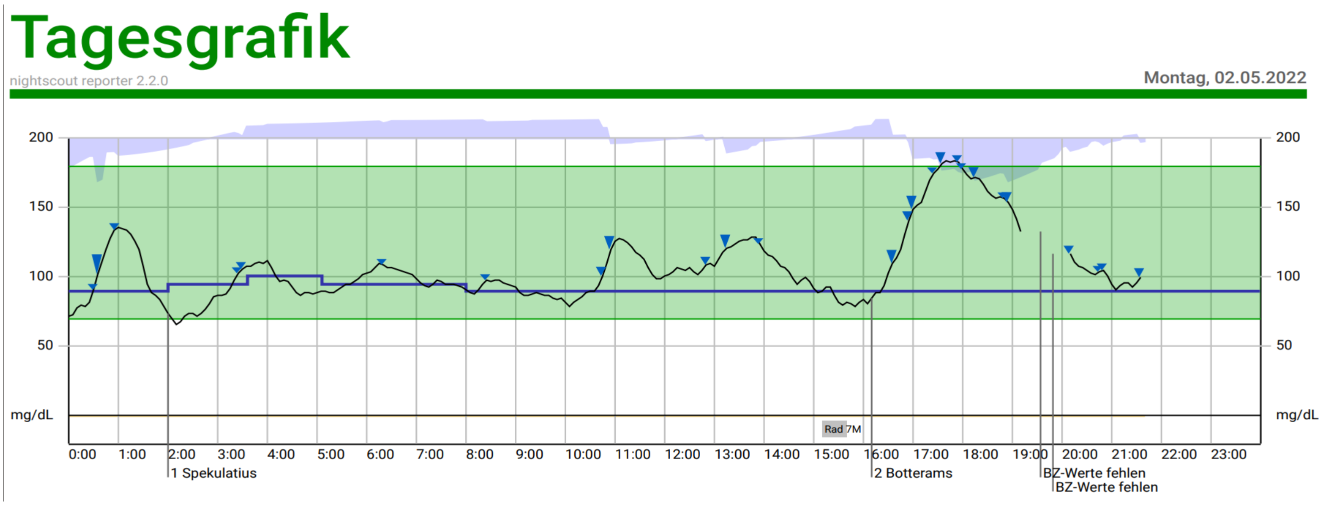
Then I remembered reading several hints that oscillations occur when ISF is too strong. Yesterday I cycled for 2 hours including a few hills. I had not done that for many months and expected it would have an stronger effect than usual because the glucose depots in the body would fill up during the night. To protect against the expected drop in blood glucose and testing a weaker ISF I reduced the pump profile overall to 90%, an easy and quick measure.

The result was quite astonishing. First here is the day with the preceding night showing a long stretch



of being low, having negative IOB and showing those oscillations. I created a VDF to swap the weights of acceleration and braking. The idea was the acceleration needs more insulin to bring me down to target while below target I need that braking effect to be stronger to allow glucose to rise to target and vice versa. It was a very interesting exercise in using the emulator but the end result was hardly different from the master run.

That low ín the preceding night was another reason why I had reduced the profile. The next night and day went much smoother with far less oscillations and staying close to target.



Conclusion

Tonight I will run at 90% profile again and if that looks as promising I can weaken ISF overall and basal during the night. Earlier I had already weakened ISF in two steps after starting autoISF because it would strengthen ISF as and when needed.

I always knew that the documentation for the emulator needed updating. Although it is generations out of date the principles published in https://github.com/ga-zelle/APS-what-if/blob/master/Instructions%20determine_basal%20emulator_282.pdf are still valid.

Status: 02.May.2022