The concept of autoISF – a prototype in AAPS 2.8.2

This document describes a prototype extension of AAPS the adapt ISF when glucose is stuck at high levels.

Caveat: *I am not a medically trained person and developed this methd purely based on trial and numerical experimentation. I had no mathematical model of the reaction kinetics of the free fatty acids which are one reason for temporary insulin resistance.*

This prototype cannot be used with Freestyle Libre CGMs as long as there is no check whether stable signals are „fozen" or real. It may be wrongly interpreted as a long lasting resistance and result in too much insulin being released.

Many users occasionally complain about „water" in the pump. Especially at high BG levels things appear to be stuck and users ask for a dynamic adaptation of ISF. This is what I intended to address. In my later use of it I saw that it also helps bringing down glucose from moderately high but constant levels.

In many control systems a PID controller is used, where

- the P-term stands for proportional behaviour. i.e. the higher the glucose deviation the higher the insulin dose. This contribution was already included in AAPS.
- the I-term stands for integrated error, i.e. the integral of the glucose deviation over time. Such a contribution is missing in AAPS and I tried to add this in the autoISF. But keep in mind that I am not a trained controller person either.
- the D-term stands for differential behaviour, i.e. the larger the change in glucose deviation the larger the reaction in insulin dosing. This contribution was already included in AAPS.

After playing around with several algorithms to acheive such an adaptation I ended up with the following autoISF method:

1. Check there is no COB, otherwise there is interference with the IC settings. In UAM mode and for enthusiasts of the recent „eat, don't bolus" fraction this is a given.
2. Check that BG is above target_bg.
3. Determine length (*autoisf_duration*) and height(*autoisf_average*) of a BG plateau where BG is within +/- 5%
4. Use a weighting system applied to duration and *target_bg* as well as the deviation from *target_bg* to calculate an ISF change factor *liftISF*. This factor behaves similarly to autosense in the way it modifies ISF and is always larger than or equal to 1:

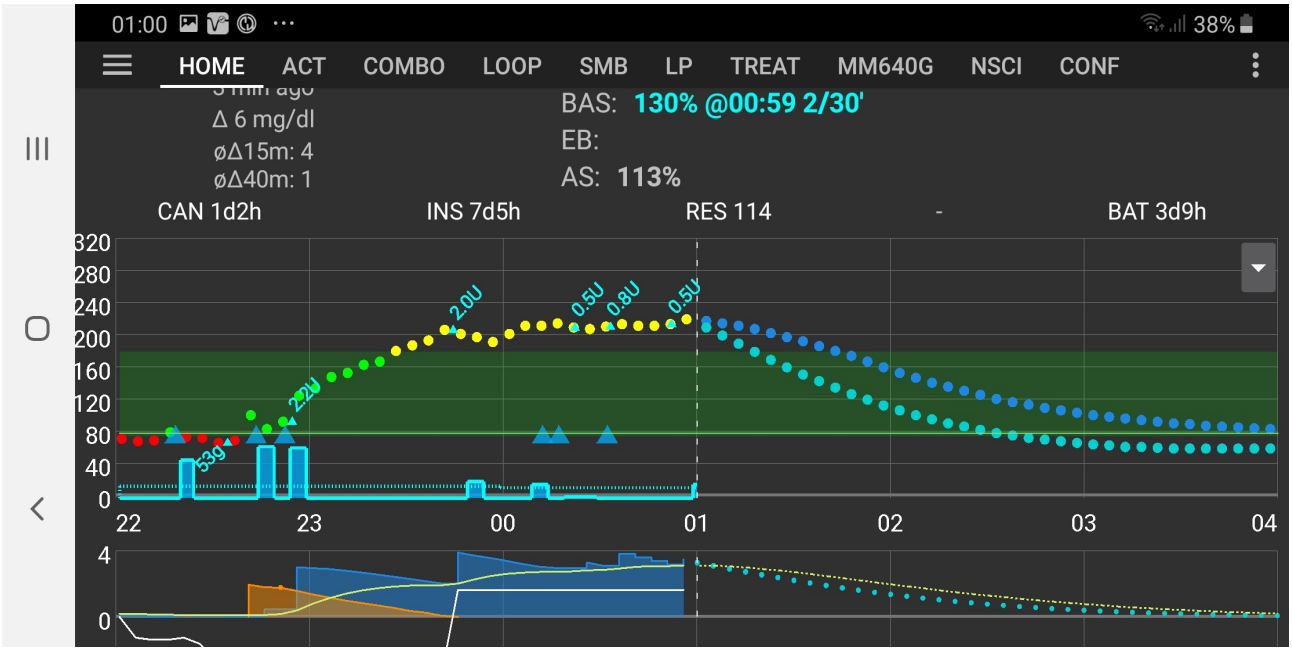
$$\text{liftISF} = 1 + \frac{\text{avg05-target\_bg}}{\text{target\_bg}} * \frac{\text{dura05}}{60} * \text{autoisf\_hourlychange}$$

where

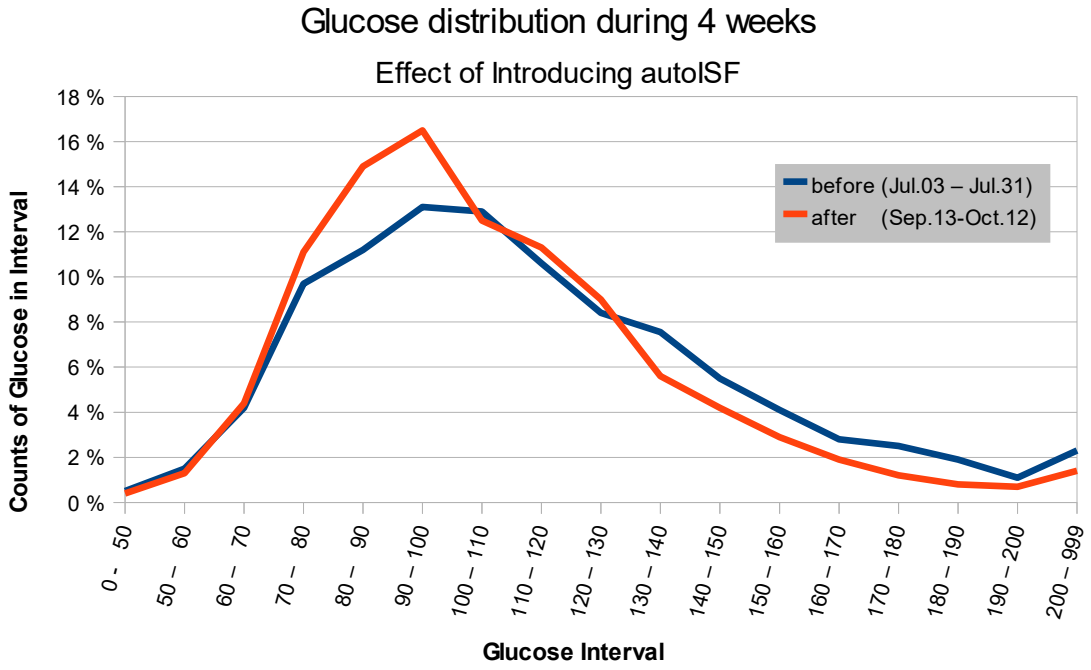
avg05 = autoisf_average
dura05 = autoisf_duration

The resulting factor has linear impact on the change of ISF, e.g. *autoisf_hourlychange* of 0.3 is twice as strong as 0.15. To get a fealing for the magnitudes consider the example of a plateau *autoisf_average* at twice the *target_bg* and lasting already for 60 minutes then *liftISF* would be 1+*autoisf_hourlychange*. In the training class for ICT and CSII we all learned the rule of thumb that the factors should be doubled if a resistance lasts

for 2 hours. This would be acheived here with *autoisf_hourlychange*=0.5 and limiting the maximum impact of this factor according to that rule by setting *autoISF_max*=2.
For algorithm selection and tuning of these factors I made extensive use of my AAPS emulator which I had extended for this purpose to include several prototype algorithms. Such emulation as available on github[1] is still the highly recommended starting point because all the initial calibration happens without risk in the virtual world. The screen shot below is from the early testing phase when the algorithm was still only in the



emulator and I figured that applying half of its result might work – which it did in this case. Once I saw an effect large enough but it would not bring me too low, I implemented it directly in AAPS and have been using it since August instead of the regular autosense. The positive effect on glucose values can be seen in the diagramme below by comparing their distributions before and after using autoISF.



1 https://github.com/ga-zelle/APS27-What-if ; meanwhile there is an upate for AAPS 2.8.2

Meanwhile I even weakened my regular ISF profile by 10-20% to hover above the target level and autoISF will pull me towards the target from time to time. This reduces my time spent below target, at least in theory. Too many other things were happening during that time so I cannot see the evidence in numbers.

I tried to develop an equivalent algorithm for glucose being too low and then weakening ISF. So far I did not find anything promising.

Please be aware that autoISF only changes ISF, nothing else like TTs or similar. Because it leaves the pump profile intact, it saves a lot of the pumps battery life compared to a method resulting in profile changes driven by automation.

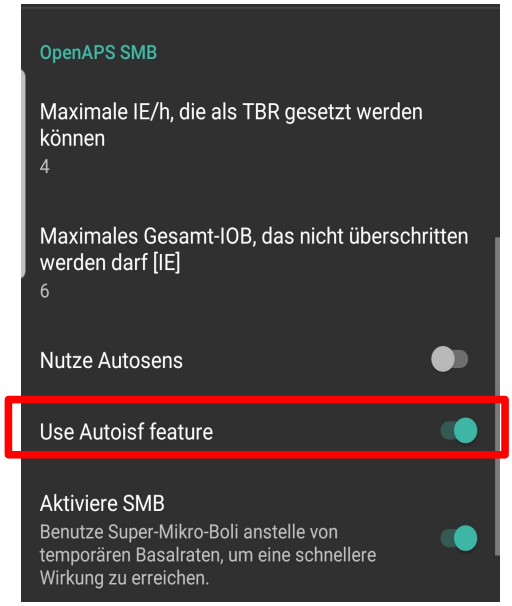
Late November 2020 BerNie from de.loopercommunity.org also started using this prototype, even in conjunction with Autosense. He observes slightly lower average glucose, slightly less time spent above 180 and minor increase of time spent below 70. The lows could be offset by increasing the target a bit.

In January 2021 Wolf, Daniel and Doraberta from de.loopercommunity.org also started using this prototype after adaptation to AAPS 2.8.x.

The use of autoISF in AAPS 2.8.2

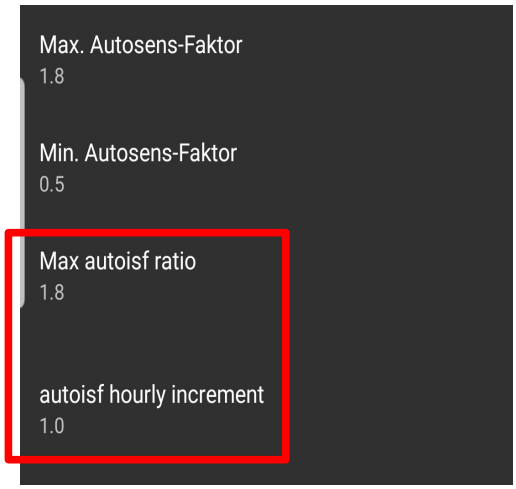
Although I am not a java developer I managed to add extra parameters starting with version 2.8.1.1 of AAPS for the autoISF scale factor and its upper limit. The autoisf settings are used in this way:

1. *autoisf enabled* or not: The method can be enabled or disabled by ticking the newly added menu item in „Preferences → OpenAPS SMB“.



2. *autoisf_hourlychange* is used to derive the resulting weighting factor *liftISF*. For initial and cautious approximation you should start weak, e.g. with the default 0.2 and increase it with caution,. For that purpose I strongly recommend to run the emulator on the phone parallel to AAPS with slightly higher settings to see whether higher values would be acceptable. The paramerer can be set in „Preferences → Resorption Settings – Extended Settings“.

3. *autoisf_max* is used as the upper limit of the resulting weighting factor *liftISF* just like in the autosense case. For reference, my setup uses 1.8 but for safety reasons you should start closer to 1.0 and increase it step by step. The default is set at 1.2.The paramerer can be set in „Preferences → Resorption Settings – Extended Settings“.



Does it interfere with autosense?

No longer, the potential interference was only present in the AAPS 2.7 prototype.

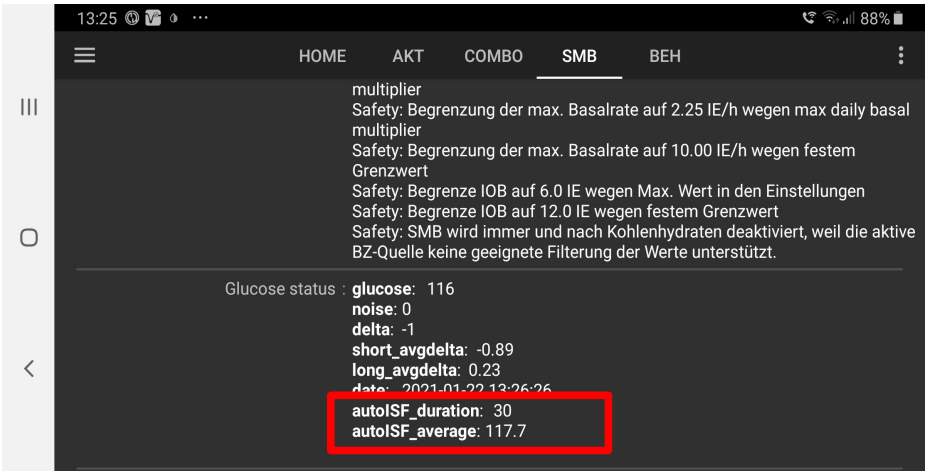
The code will pick whichever of the two methods has the larger impact on ISF at any point in time as can be seen further down in the second example.

When the BG situation changes then autoISF reacts promptly compared to the sluggish autosens. In that sense there is less of an interference than thaught because they mostly act at different times.

How can you see what it does?

The SMB-tab is the place to check:

1. In *Glucose_status* there are two new items appended, namely *autoisf_duration* and *autoisf_average* as mentioned above in the concept section. As long as *autoisf_duration* is less than 10m autoISF is not active and there is no point checking elsewhere.



2. In *Profile*, right at the end *autoisf_max* and *autoisf_hourlychange* were appended, joining up with good old *autosense_max*. For you it serves as a reminder what their current values are.

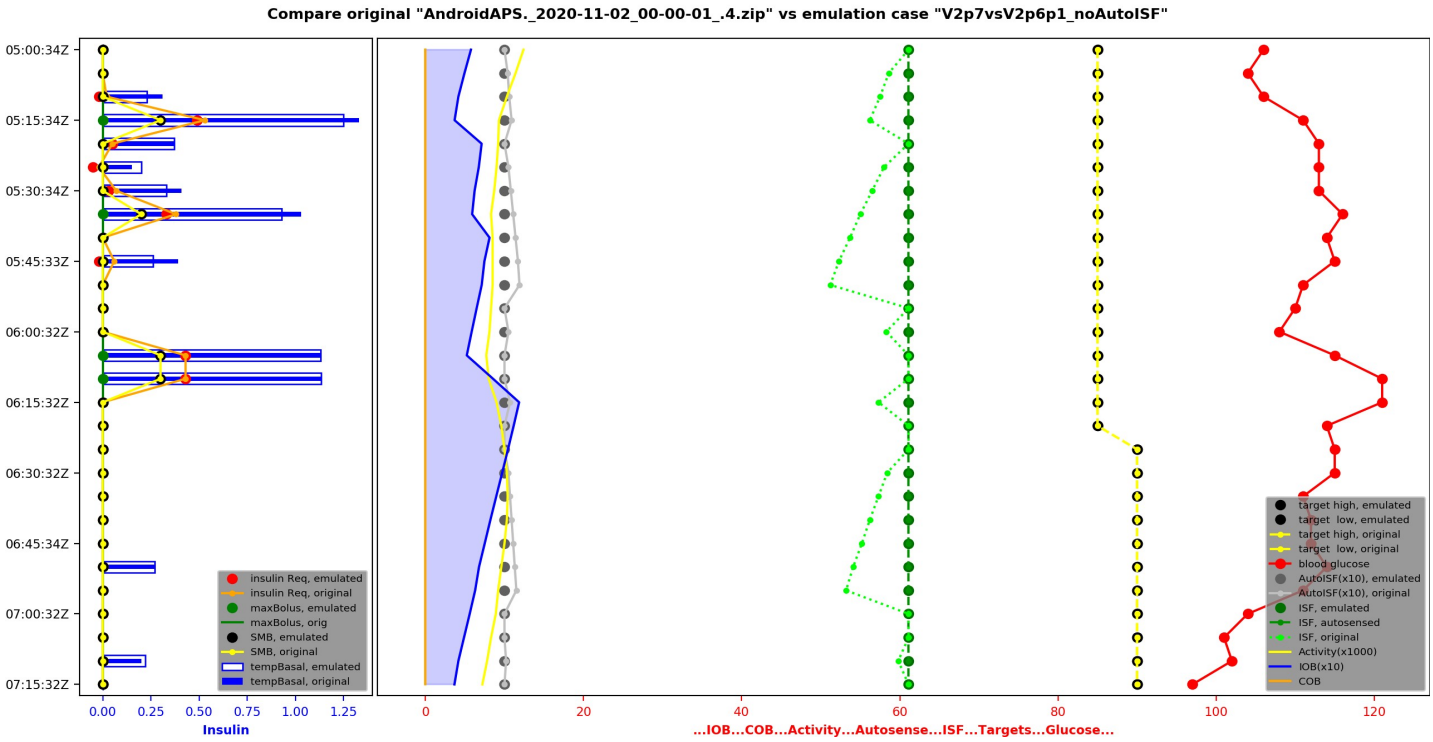
3. In *Meal data* you can check for *mealCOB* = 0 or not. But of course that was already clear from the carb area on the home screen.

4. The *Result / Script debug* holds the really interesting information. The first few lines are an echo of what autosens did to ISF, basal and CR.
   Thereafter the line(s) starting with „*autoISF ...*“ are an echo of autoISF, e.g.
   - *autoISF disabled in Preferences*
   - *autoISF by-passed; BG is only 5m at level 107.5*
     Obviously the 10 minutes minimum time window was not acheived
   - *autoISF reports 100 did not do it for 20 m; go more aggressive by 1.25*
     Here the autoISF will become active and strengthen ISF of 100 by a factor 1.25
   - *autoISF reduction 1.6 limited by autoisf_max 1.5*
     Here the calculated strengthening of 1.6 is capped to 1.5 as defined in *autoisf_max*
   - *autoISF by-passed; avg. glucose 90 below target 95*
     Here BG is already below target and ISF should not be strengthened
   - *autoISF by-passed; mealCOB of 9.6*
     Here autoISF is deactivated due to presence of carbs

   A few lines further down AAPS explicitely shows the ISF as defined in the profile and the final value after applying autosens and autoISF.

Example with autosense disabled

This prototype implementation does not yet include a capability to plot the overall *liftISF* factor, especially overlayed with the autosense factor. Therefore, if you missed the time to check the SMB tab it is hard to find out what autoISF did before. You can dig through the logfiles but that is cumbersome. I use my emulator to get graphs and tables to see what happened.



The above figure compares the original run including active autoISF against a virtual case where it was not used. Right at the beginning you see a plateau at about 105mg/dl which lasts up to 15m. Shortly after that another plateau develops at about 113mg/dl and lasts for 45m. In both periods the light green line shows continued strengthening of ISF for the original use and that ISF is back to normal as soon as the plateau ends. The smaller, lighter grey dots show the overall *liftISF* factor applied in the original run, the lager darker ones are at 1.0 because autoISF was off in the emulated run.

The bars to the left show more insulin being delivered when autoISF is active. After 06:30Z there is another plateau with strengthening ISF but in spite of that the insulin delivered is unchanged because enough of it was on board.

The table below tells the same story. The 4 columns for ISF mean:

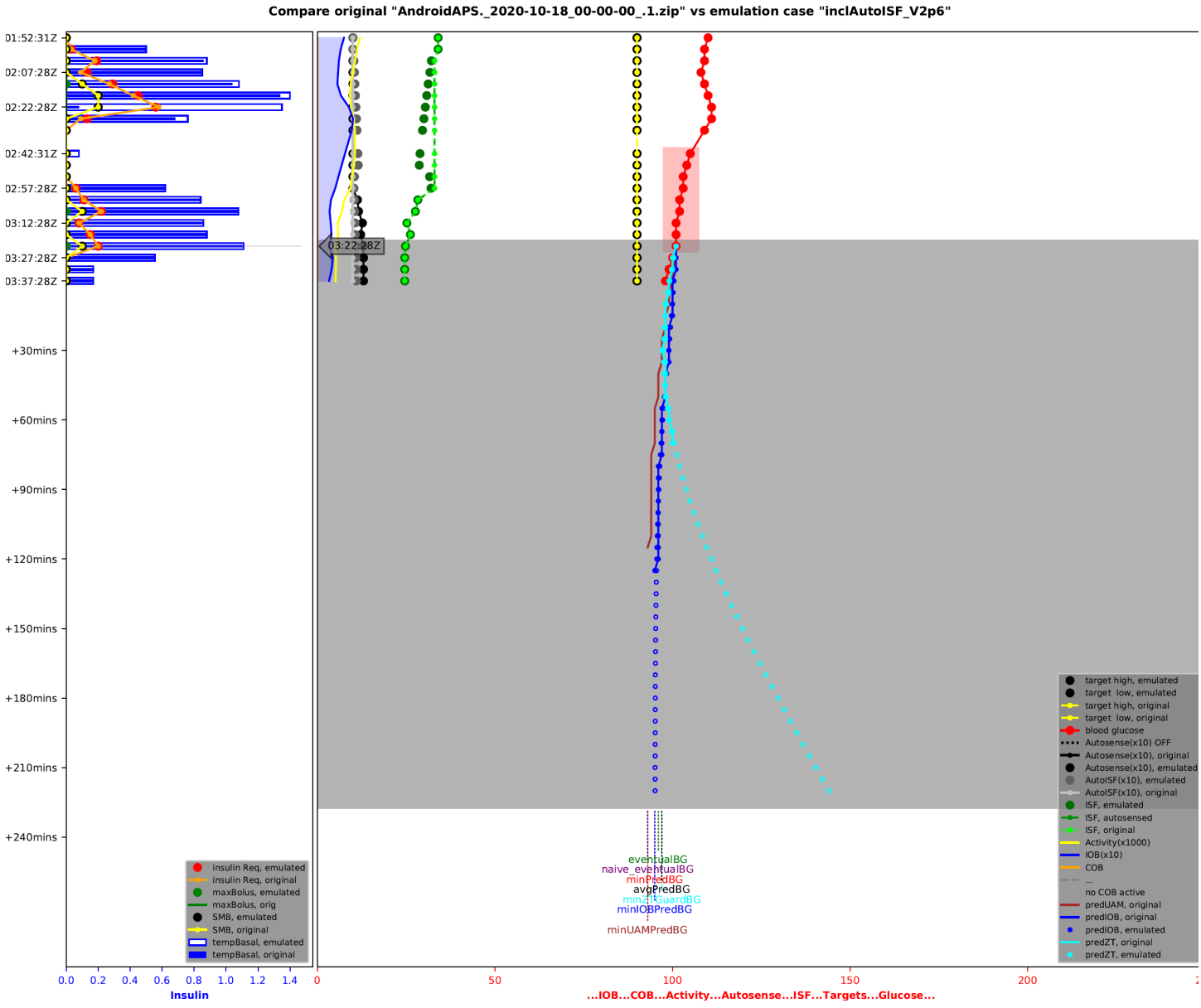
- orig: the original, real use case
- prof: as defined ín the active pump profile
- auto: the prof value potentially modified by autosense
- emul: value in the virtual, simulated situation, in this case no autoISF

UTC time	bg	avg. targ	IOB	COB	-AutoISF- orig	emul	--5% dura	range- avg.	---------ISFs--------- orig	prof	auto	emul	insulin Req orig	emul	---SMB--- orig	emul	--tmpBasal-- orig	emul
05:00Z	106	85.0	0.58	0	1.0	1.0	0.0	106.0	61.1	61.1	61.1	61.1	0	0	0	0	0	0
05:05Z	104	85.0	0.49	0	1.04	1.0	5.0	105.0	58.6	61.1	61.1	61.1	0	0	0	0	0	0
05:10Z	106	85.0	0.42	0	1.06	1.0	10.0	105.3	57.5	61.1	61.1	61.1	0.02	-0.02	0	0	0.31	0.23
05:15Z	111	85.0	0.37	0	1.09	1.0	15.0	106.8	56.2	61.1	61.1	61.1	0.53	0.49	0.3	0.3	1.33	1.25
05:20Z	113	85.0	0.71	0	1.0	1.0	5.0	112.0	61.1	61.1	61.1	61.1	0.05	0.05	0	0	0.37	0.37

05:25Z	113	85.0	0.68	0	1.05	1.0	10.0	112.3	58	61.1	61.1	61.1	-0.02	-0.05	0	0	0.15	0.2
05:30Z	113	85.0	0.62	0	1.08	1.0	15.0	112.5	56.5	61.1	61.1	61.1	0.07	0.03	0	0	0.41	0.33
05:35Z	116	85.0	0.59	0	1.11	1.0	20.0	113.2	55	61.1	61.1	61.1	0.38	0.33	0.2	0.2	1.03	0.93
05:40Z	114	85.0	0.81	0	1.14	1.0	25.0	113.3	53.7	61.1	61.1	61.1	0	0	0	0	0	0
05:45Z	115	85.0	0.75	0	1.17	1.0	30.0	113.6	52.3	61.1	61.1	61.1	0.06	-0.02	0	0	0.39	0.26
05:50Z	111	85.0	0.71	0	1.19	1.0	35.0	113.2	51.2	61.1	61.1	61.1	0	0	0	0	0	0
05:55Z	110	85.0	0.65	0	1.0	1.0	40.0	112.9	61.1	61.1	61.1	61.1	0	0	0	0	0	0
06:00Z	108	85.0	0.58	0	1.05	1.0	45.0	112.4	58.3	61.1	61.1	61.1	0	0	0	0	0	0
06:05Z	115	85.0	0.52	0	1.0	1.0	0.0	115.0	61.1	61.1	61.1	61.1	0.43	0.43	0.3	0.3	1.13	1.13
06:10Z	121	85.0	0.86	0	1.0	1.0	5.0	118.0	61.1	61.1	61.1	61.1	0.43	0.43	0.3	0.3	1.13	1.13
06:15Z	121	85.0	1.19	0	1.07	1.0	10.0	119.0	57.3	61.1	61.1	61.1	0	0	0	0	0	0
06:20Z	114	85.0	1.12	0	1.0	1.0	0.0	114.0	61.1	61.1	61.1	61.1	0	0	0	0	0	0
06:25Z	115	90.0	1.04	0	1.0	1.0	5.0	114.5	61.1	61.1	61.1	61.1	0	0	0	0	0	0
06:30Z	115	90.0	0.97	0	1.05	1.0	10.0	114.7	58.4	61.1	61.1	61.1	0	0	0	0	0	0
06:35Z	111	90.0	0.9	0	1.07	1.0	15.0	113.8	57.3	61.1	61.1	61.1	0	0	0	0	0	0
06:40Z	112	90.0	0.82	0	1.09	1.0	20.0	113.4	56.2	61.1	61.1	61.1	0	0	0	0	0	0
06:45Z	112	90.0	0.75	0	1.11	1.0	25.0	113.2	55.2	61.1	61.1	61.1	0	0	0	0	0	0
06:50Z	114	90.0	0.68	0	1.13	1.0	30.0	113.3	54.1	61.1	61.1	61.1	0	0	0	0	0.27	0.27
06:55Z	111	90.0	0.63	0	1.15	1.0	35.0	113.0	53.2	61.1	61.1	61.1	0	0	0	0	0	0
07:00Z	104	90.0	0.56	0	1.0	1.0	0.0	104.0	61.1	61.1	61.1	61.1	0	0	0	0	0	0
07:05Z	101	90.0	0.49	0	1.0	1.0	5.0	102.5	61.1	61.1	61.1	61.1	0	0	0	0	0	0
07:10Z	102	90.0	0.42	0	1.02	1.0	10.0	102.3	59.8	61.1	61.1	61.1	0	0	0	0	0.2	0.22
07:15Z	97	90.0	0.37	0	1.0	1.0	0.0	97.0	61.1	61.1	61.1	61.1	0	0	0	0	0	0

Example with concurrent use of autosense

Thank you to BerNie from de.loopercommunity.org for providing the original, non-autoISF data. This time the comparison is the other way round, i.e. the original run is from BerNie with autosense and automation both enabled and the emulated run shows the changes if autoISF would have been used, too.



Looking at the dark green dots we see autoISF gradually strengthening ISF. The factor *lifISF*, i.e. how much it is strengthened by autoISF, are shown by the grey dots. From the black dots we see that autosens is only detecting resistance towards the end of this time window. It then dominates the strengthening of ISF. The transparent red-ish box shows the +/- 5% range identified by autoISF as its reason for action at time 03:22Z. However, assuming *autoisf_hourlychange*=0.8 its influence is less than that of autosens. The numerical comparisons of ISF look like this:

UTC time	bg	avg. targ	IOB	COB	Auto sens	-AutoISF- orig	-AutoISF- emul	--5% range- dura	--5% range- avg.	---------ISFs--------- orig	---------ISFs--------- prof	---------ISFs--------- auto	---------ISFs--------- emul	insulin Req orig	insulin Req emul	---SMB--- orig	---SMB--- emul	--tmpBasal-- orig	--tmpBasal-- emul
01:52Z	110	90.0	0.75	0	1.0	1.0	1.0	0.0	110.0	34	34	34.0	34.0	0	0	0	0	0	0
01:57Z	109	90.0	0.66	0	1.0	1.0	1.0	5.0	109.5	34	34	34.0	34.0	0.03	0.03	0	0	0.5	0.5
02:02Z	109	90.0	0.61	0	1.0	1.0	1.03	10.0	109.3	33	33	33.0	32.1	0.18	0.19	0	0	0.86	0.88
02:07Z	108	90.0	0.58	0	1.0	1.0	1.04	15.0	109.0	33	33	33.0	31.7	0.09	0.13	0	0	0.85	0.85
02:12Z	109	90.0	0.56	0	1.0	1.0	1.06	20.0	109.0	33	33	33.0	31.2	0.27	0.29	0.1	0.1	1.04	1.08
02:17Z	110	90.0	0.66	0	1.0	1.0	1.07	25.0	109.2	33	33	33.0	30.8	0.42	0.45	0.2	0.2	1.34	1.4
02:22Z	111	90.0	0.89	0	1.0	1.0	1.09	30.0	109.4	33	33	33.0	30.4	0.58	0.56	0.2	0.2	0.08	1.35

02:27Z	111	90.0	1.0	0	1.0	1.0	1.1	35.0	109.6	33	33	33.0	30.0	0.09	0.13	0	0	0.68	0.76
02:32Z	109	90.0	0.97	0	1.0	1.0	1.12	40.0	109.6	33	33	33.0	29.6	0	0	0	0	0	0
02:42Z	105	90.0	0.78	0	1.0	1.0	1.14	50.0	109.1	33	33	33.0	28.9	0	0	0	0	0	0.08
02:47Z	104	90.0	0.69	0	1.0	1.0	1.15	55.0	108.6	33	33	33.0	28.7	0	0	0	0	0	0
02:52Z	103	90.0	0.59	0	1.0	1.0	1.05	20.0	105.2	33	33	33.0	31.6	0	0	0	0	0	0
02:57Z	103	90.0	0.5	0	1.0	1.0	1.03	15.0	103.8	33	33	33.0	32.0	0.06	0.06	0	0	0.62	0.62
03:02Z	102	90.0	0.38	0	1.13	1.0	1.04	20.0	103.4	28.3	32	28.3	28.3	0.11	0.11	0	0	0.84	0.84
03:07Z	102	90.0	0.34	0	1.16	1.0	1.05	25.0	103.2	27.6	32	27.6	27.6	0.22	0.22	0.1	0.1	1.08	1.08
03:12Z	101	90.0	0.38	0	1.27	1.0	1.06	30.0	102.9	25.2	32	25.2	25.2	0.08	0.08	0	0	0.86	0.86
03:17Z	101	90.0	0.4	0	1.22	1.0	1.07	35.0	102.6	26.2	32	26.2	26.2	0.15	0.15	0	0	0.88	0.88
03:22Z	101	90.0	0.34	0	1.29	1.0	1.07	40.0	102.4	24.8	32	24.8	24.8	0.2	0.2	0.1	0.1	1.11	1.11
03:27Z	100	90.0	0.44	0	1.3	1.0	1.08	45.0	102.2	24.6	32	24.6	24.6	0	0	0	0	0.56	0.56
03:32Z	99	90.0	0.4	0	1.3	1.0	1.09	50.0	101.9	24.6	32	24.6	24.6	0	0.0	0	0	0.17	0.17
03:37Z	98	90.0	0.33	0	1.3	1.0	1.09	55.0	101.6	24.6	32	24.6	24.6	0	0.0	0	0	0.17	0.17