AutoISF Prototype in the Emulator on the PC
Test and Calibrate Adaptation of ISF

The Concept

Before changing APS to include such automated ISF adaptations it is wise to first try it out in a virtual environment. By using the emulator in this repo such virtual scenarios can be investigated for any interesting period in the past. This way various algorithms can be tested and calibrated. For that purpose the emulator was extended to include code for adapting ISF based on special variant definition parameters. These are not included in the main emulator document and are prepared for various situations in the BG curve:

BG is nearly constant for at least 10 minutes, above target and COB=0

BG values are within +/- 5% for *dura05* minutes and their average is *avg05*. The label *...05* comes from the range width of 5%. An example of the VDF (Variant Definition File) looks like this:

```
new_parameter   autoISF_flat        True         ### additional parameter; AAPS is fix at False; enable autoISF_flat

new_parameter   maxISFAdaptation    1.7                              ### later replace it by profile['autosens_max']
temp            dura05_weight       glucose_status['dura05'] / 60    ### 1.0 at 60 min duration
temp            avg05_weight        1.0/profile['target_bg']         ### target_bg may be modified in AAPS, e.g. by autosense
temp            deltaBG             abs(glucose_status['avg05']-profile['target_bg'])

new_parameter   prodISF_flat        temp['dura05_weight']*temp['avg05_weight']*pow(temp['deltaBG'],1)
new_parameter   liftISF_flat        min(new_parameter['maxISFAdaptation'], 1+new_parameter['prodISF_flat'])
```

- *autoISF_flat* is a flag to enable this method which is ignored otherwise
- *maxISFAdaptation* defines the upper limit of adaptation like in *autosens_max.* With a value of 1 there is no change, 2 means ISF can only be reduced down numerically to half its original value.
- The various *temp* assignments are factors to hold interim values
- *prodISF_flat* is the product of all those interim factors and is the relative increase in ISF strength. The pow() term was a try to achieve a very smooth transition close to target. However, power of 2 turned out to be too weak close to target and too strong far above. So I could have used just *temp['deltaBG']* instead.
- *liftISF_flat* is the final factor like in Autosense and is limited by *maxISFAdaptation.* The final ISF is then profile ISF divided by *liftISF_flat.* In the emulator it is also checked against lift*ISF_slope* described below and against the Autosense adaptation of ISF. The stronger of them prevails.

Of course the file „*...<VDF>.log*“ should be checked for evaluation of these formulae as intended. The file „*...<VDF>.txt*“ created by the emulator includes echoes of the final assignments. The output option „*/range*“ can be used to include *dura05* and *avg05* in the on screen table and the graph.

Over the course of 2 weeks I fine tuned the algorithm and ended up with the above version which I implemented in AAPS and have been using it since August 2020.

BG is nearly constant for at least 10 minutes and below target

BG values are within +/- 5% for *dura05* minutes and their average is *avg05*. In principle the VDF (Variant Definition File) could look like this:

```
new_parameter   autoISF_low         True                   ### additional parameter; AAPS is fix at False; enable autoISF_low

temp            dura05_weight       ... your expression …  ### 1.0 at 60 min duration
temp            avg05_weight        ... your expression …  ### target_bg may be modified in AAPS, e.g. by autosense
temp            deltaBG             ... your expression …

new_parameter   weakISF              ... your expression … ### include any min/max limits here
```

- *autoISF_low* is a flag to enable this method which is ignored otherwise
- there are no checks inside the emulator for minimum or maximum adaptations
- The various *temp* assignments can hold interim values
- *weakISF* is the final factor by which ISF is numerically increased. There is no check whether other influences like Autosense might deliver a different or weaker result.

My hope was to create something like a wall with high ISF factors so BG could not drop much further. So far I did not find a useful algorithm. Especially with negative IOB it even backfired.

BG is rising nearly linearly for at least 10 minutes and COB=0

BG values are increasing by *slope70* (in mg/dl/5min) along a nearly straight line for *dura70* minutes. That line is determined by a linear regression analysis using a length deliveríng maximum correlation but at least 70%. The label *...70* comes from that minimum correlation coefficient. An example of the VDF (Variant Definition File) looks like this:

```
new_parameter   autoISF_slope       True        ### additional parameter; AAPS is fix at False; enable autoISF_slope

new_parameter   maxISFAdaptation    1.7                                     ### later replace it by profile['autosense_max']
temp            deltaBG             abs(glucose_status['avg05']-profile['target_bg'])
temp            dura70_weight       glucose_status['dura70'] / 30           ### 1.0 at 30 min duration
temp            slope70_weight      max(0, 1+glucose_status['slope70']/120)  ### 0.0 at <=0, 1.1 at 12mg/dl/5min

new_parameter   prodISF_slope       temp['slope70_weight']*temp['dura70_weight']*temp['avg05_weight']*pow(temp['deltaBG'],1)
new_parameter   liftISF_slope       min(new_parameter['maxISFAdaptation'], 1+new_parameter['prodISF_slope'])
```

- *autoISF_slope* is a flag to enable this method which is ignored otherwise
- *maxISFAdaptation* defines the upper limit of adaptation like in *autosens_max.* With a value of 1 there is no change, 2 means ISF can only be reduced down numerically to half its original value.
- The various *temp* assignments are factors to hold interim values
- *prodISF_slope* is the product of all those interim factors and is the relative increase in ISF strength. The pow() term was a try to achieve a very smooth transition close to target. However, power of 2 turned out to be too weak close to target and too strong far above. So I could have used just *temp['deltaBG']* instead.
- *liftISF_slope* is the final factor like in Autosense and is limited by *maxISFAdaptation.* The final ISF is then profile ISF divided by *liftISF_slope.* In the emulator it is also checked against lift*ISF_flat* described above and against the Autosense adaptation of ISF. The stronger of them prevails.

Of course the file „*...<VDF>.log*“ should be checked for evaluation of these formulae as intended. The file „*...<VDF>.txt*“ created by the emulator includes echoes of the final assignments. The output option „*/slope*“ can be used to include *dura70* and *slope70* in the on screen table and the graph.

The algorithm shown above is not satisfactory. It either results in an incremental bolus of just 0.1 IE or delivers more which leads to hypo alarm shortly after. So without further research and trials this does not look useful.

Example of tabular output with options *„/slope/range“*

Thank you to BerNie from de.loopercommunity.org for providing the original, non-autoISF data. The original run is with autosense and automation both enabled and the emulated run shows the changes if autoISF would have been used, too.

UTC		avg.			Auto	--5% range-		--lin.fit-		---------ISFs---------				insulin Req		---SMB---		--tmpBasal--	
time	bg	targ	IOB	COB	sens	dura	avg.	dura	rate	orig	prof	auto	emul	orig	emul	orig	emul	orig	emul
01:52Z	110	90.0	0.75	0	1.0	0.0	110.0	0	0	34	34	34.0	34.0	0	0	0	0	0	0
01:57Z	109	90.0	0.66	0	1.0	5.0	109.5	0	0	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	10.0	109.3	10.0	-0.5	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	15.0	109.0	15.0	-0.6	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	20.0	109.0	20.0	-0.3	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	25.0	109.2	10.0	1.0	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	30.0	109.4	10.0	1.0	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	35.0	109.6	20.0	0.8	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	40.0	109.6	10.0	-1.0	33	33	33.0	29.6	0	0	0	0	0	0
02:42Z	105	90.0	0.78	0	1.0	50.0	109.1	15.0	-2.0	33	33	33.0	28.9	0	0	0	0	0	0.08
02:47Z	104	90.0	0.69	0	1.0	55.0	108.6	20.0	-1.8	33	33	33.0	28.7	0	0	0	0	0	0
02:52Z	103	90.0	0.59	0	1.0	20.0	105.2	10.0	-1.0	33	33	33.0	31.6	0	0	0	0	0	0
02:57Z	103	90.0	0.5	0	1.0	15.0	103.8	35.0	-1.3	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	20.0	103.4	40.0	-1.3	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	25.0	103.2	45.0	-1.1	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	30.0	102.9	30.0	-0.6	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	35.0	102.6	35.0	-0.6	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	40.0	102.4	40.0	-0.5	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	45.0	102.2	40.0	-0.5	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	50.0	101.9	10.0	-1.0	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	55.0	101.6	15.0	-1.0	24.6	32	24.6	24.6	0	0.0	0	0	0.17	0.17

In the table above the 2 columns for „--5% range-“ mean:

- dura the length of the time window in which BG is within a +/- 5% range
- avg. the average BG value within that time window

In the table above the 2 columns for „--lin.fit-“ show the result of the linear regression analysis and mean:

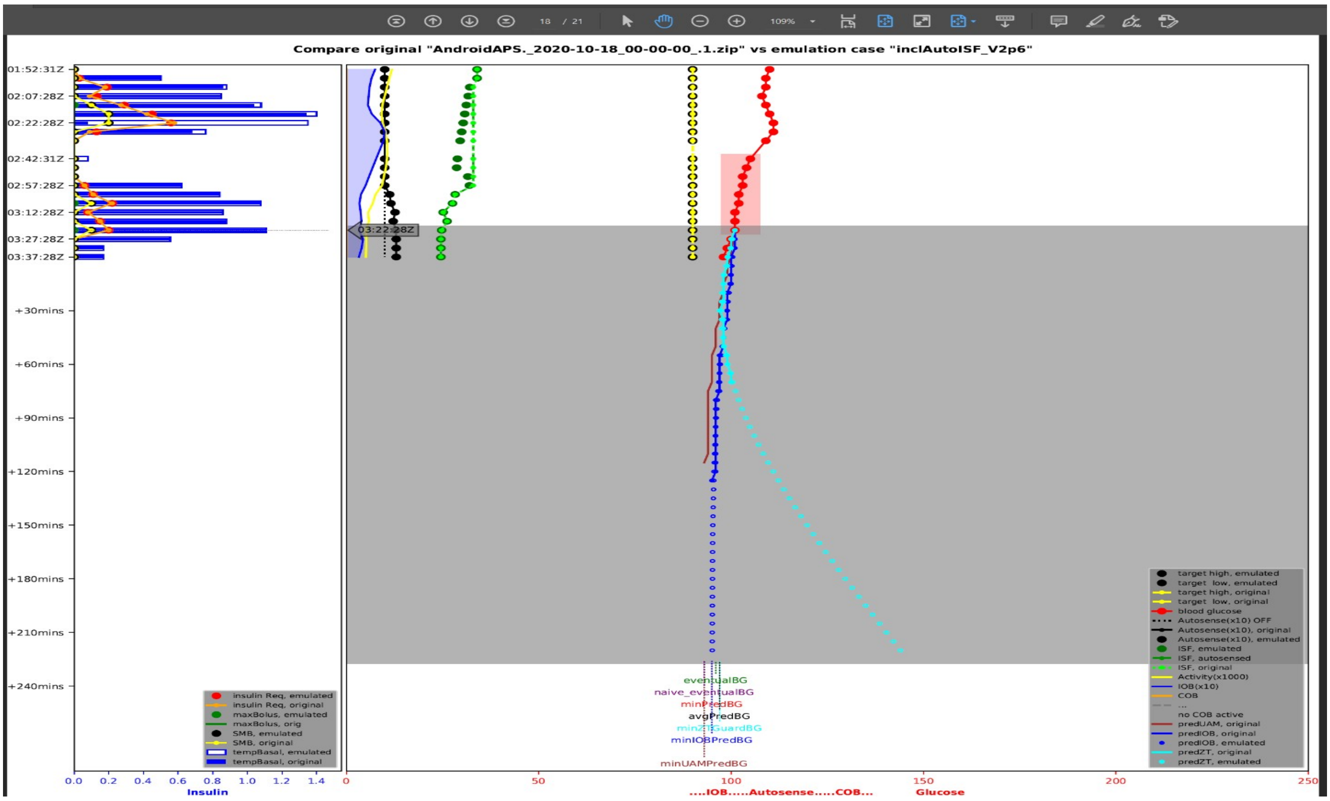
- dura the lenght of the time window in which BG is following a nearly straight trend
- rate the slope of that trend line in mg/dl/5min

In the table above the 4 columns for ISF mean:

- orig the original, real AAPS use case, in this case without autoISF
- 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 with autoISF activated and the dominant effect being activated

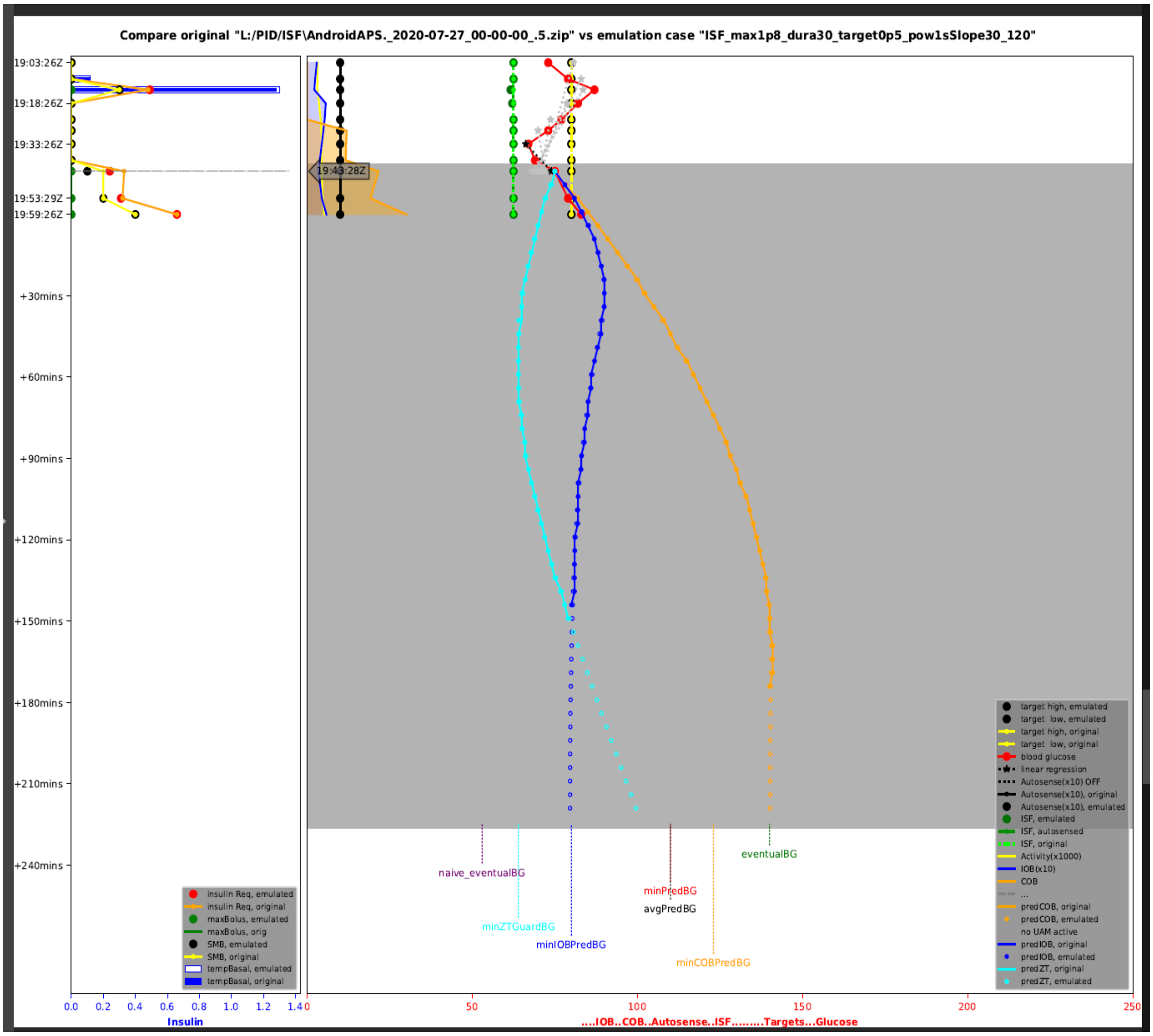
Example of graphical output with options *„/pred/range“*

This graph is the same situation as in the tabular output example above and for the same time window.



Looking at the dark green dots we see autoISF gradually strengthening ISF. 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.

<u>Example of graphical output with options „/pred/slope“</u>



The results of the regression analysis are overlayed on the BG graph as dotted lines with stars as endpoints:

- the grey ones are all the shorter or longer time range possibilities with correlation at least 70%
- the black dotted line is the one with the best correlation.

Status: 14-DEC-2020 @ 16:20