AutoISF2.2 – Quick User Guide

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

Introduction

AutoISF is meant for the advanced user who has a deep understanding of AAPS and who has tuned the system to achieve a TIR of about 90% or better. If such a user is ambitious and wants to improve further, the methods contained can very well help.

This document describes how and when ISF is adapted automatically and provides short descriptions of the weighting factors to tune it to your individual needs. Detailed guides covering special situations and examples will follow later as separate documents including results of using it in full loop mode, i.e. without carb or insulin entry by the user.

The adaptation of ISF is based on special glucose behaviour and results in an adaptation factor just like Autosense. However, here only ISF is adapted and no other setting. The scenarios analysed typically cover the last 10-30 minutes and therefore autoISF reacts much faster to problems or recoveries. Often Autosense would drive me into hypos because of its delayed reaction even after things had come back to target and I disabled Autosense although both can coexist.

AutoISF is part of oref1 in OpenAPS SMB and cannot coexist with the recent DynamicISF which therefore is included in its own plug-in as an alternative to OpenAPS AMA and OpenAPS SMB.

There are 4 different effects in glucose behaviour that autoISF checks and reacts to:

1. **acce_ISF** is a factor derived from acceleration of glucose levels
2. **bg_ISF** is a factor derived from the deviation of glucose from target
3. **delta_ISF** and **pp_ISF** are factors derived from glucose delta
4. **dura_ISF** is a factor derived from glucose being stuck at high levels

Finally these factors are compared among each other and against Autosense. Normally the strongest of them will be used with some exceptions as detailed further below. These factors work the same way as the Autosense factor works, i.e. the profile sensitivity is divided by the factor to deliver a final sensitivity ISF.

In the SMB-tab, section Script debug, you can always see which values were assigned during the last loop execution. It also lists explanations in case the factor had to be modified or why it cannot be used. Some interim values like dura_ISF_average are listed, too. All that can be seen in the SMB-tab, sections Glucose status, Profile and Script debug, respectively.

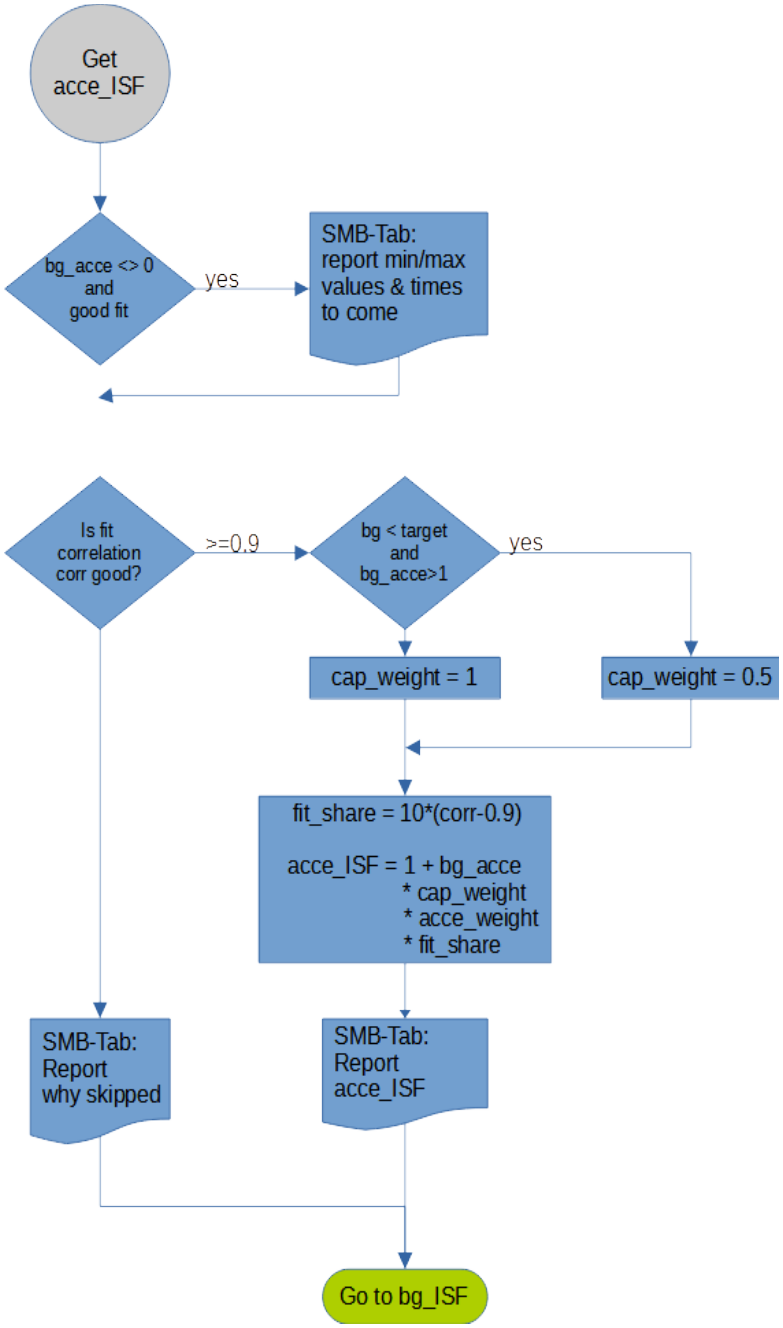
Again in analogy to Autosense there are upper (autoISF_max) and lower (autoISF_min) limits for how far ISF can be modified in total.

In analogy to enabling SMB there is a setting enable_autoISF which determines whether any part of autoISF is enabled or none at all.

The settings specific to autoISF are collected in its own menu found at the end of the OpenAPS SMB menu. A screenshot is shown as attachment. Another trick for finding them is to use the filter method at the top of the Preferences page which searches for all settings containing the string entered in the filter field.

acce_ISF determination and its impact

This is the latest contribution to autoISF and has been in action since late December 2021.



For things like glucose or delta to change, there first must be an acceleration. Therefore acceleration recognises such changes earlier and is used by autoISF to take pre-emptive action.

Glucose and delta, its 1st derivative, play a significant role in AAPS in determining the insulin required. Acceleration, its 2nd derivative, was not included so far. One reason might be that it is harder to extract from the glucose history considering that delta needed to be averaged already to provide a reliable signal. In autoISF a best fit algorithm is used to determine the parabola which best matches the glucose data.

Once the formula for the parabola is known it is then very easy to determine the acceleration. Sometimes the fit has bad correlation, i.e. it deviates too much from the glucose readings. In such cases there is no contribution from acceleration and acce_ISF = 1.

Otherwise acce_ISF is calculated by

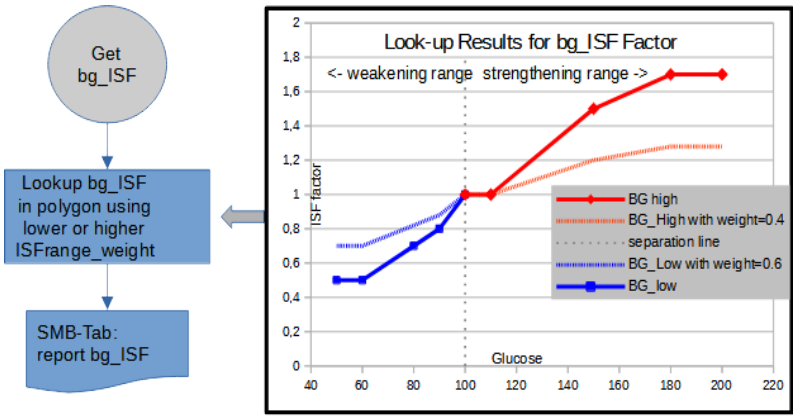
acce_ISF = 1 + acce_weight * fit_share * cap_weight* acceleration

where	fit_share	a measure of fit quality, is 0% if unacceptable up to 100% if perfect;
	cap_weight	is 0.5 below target and 1.0 otherwise;
	acce_weight	is bgAccel_ISF_weight for positive acceleration or bgBrake_ISF_weight for negative acceleration.

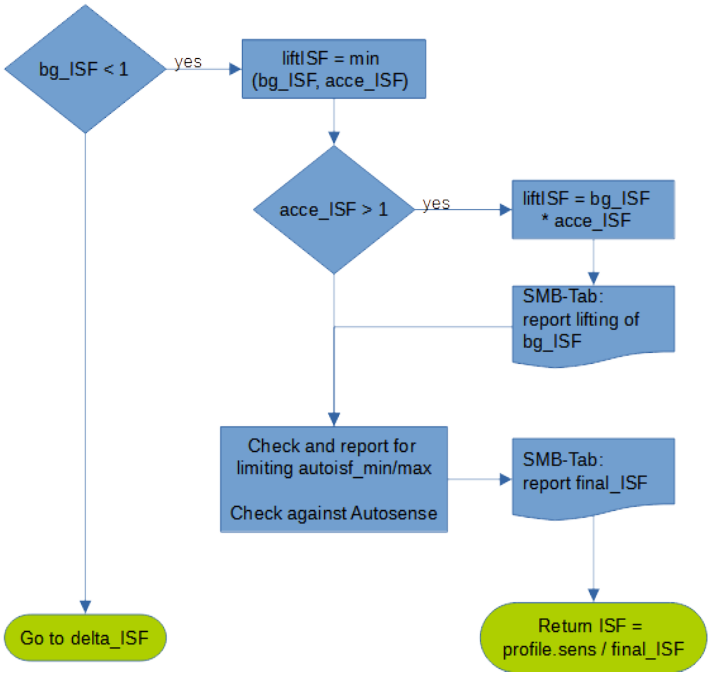
It is still early days but it can be assumed that the weights for accelerating and braking are of similar size. Quite often the acce_ISF contribution plays the dominant role inside autoISF and is therefore very important and delicate. Weights for acce_ISF of 0 disable this contribution. Start small with weights like 0.02 and observe the results before increasing them. Keep in mind that negative acceleration will start to happen while glucose is apparently still rising but the slope reduces. Here, acce_ISF will be <1, i.e. sensitivity grows and less insulin than normal will be required even before the glucose peak is reached.

bg_ISF determination and its impact

There are indicators that higher glucose needs stronger ISF. This was evident from all the successful AAPS users defining automation rules which strengthen the profile at higher glucose levels. The drawback is that there are sudden jumps in ISF at switch points and no further or minor adaptations in between.



In autoISF a polygon is provided that defines a relationship between glucose and ISF and interpolates in between. This is currently hard coded but the user can apply weights to easily strengthen or weaken it in order to fit personal needs. In principle the polygon itself can be edited and the apk rebuilt if a different shape is required. Developing a GUI for that purpose was considered very tedious especially before knowing whether the results warrant the effort. With this approach you could even approximate the formula well enough that is used in DynamicISF for the ISF dependency.



There are two weighting factors depending on whether glucose is below or above target:

lower_ISFrange_weight	Used below target, weakens ISF the more the higher this weight is; 0 disables this contribution, i.e. ISF is constant in the whole range below target. This weight is less critical as the loop is probably running at TBR=0 anyway and you can start around 1.
higher_ISFrange_weight	Used above target, strengthens ISF the more the higher this weight is 0 disables this contribution, i.e. ISF is constant in the whole range above target. Start with a weight of 0.2 and observe the reactions and check the SMB-tab before you increase it with care.

The result is:

bg_ISF = 1 + *xxx*_ISFrange_weight * glucose_polygon_Lookup

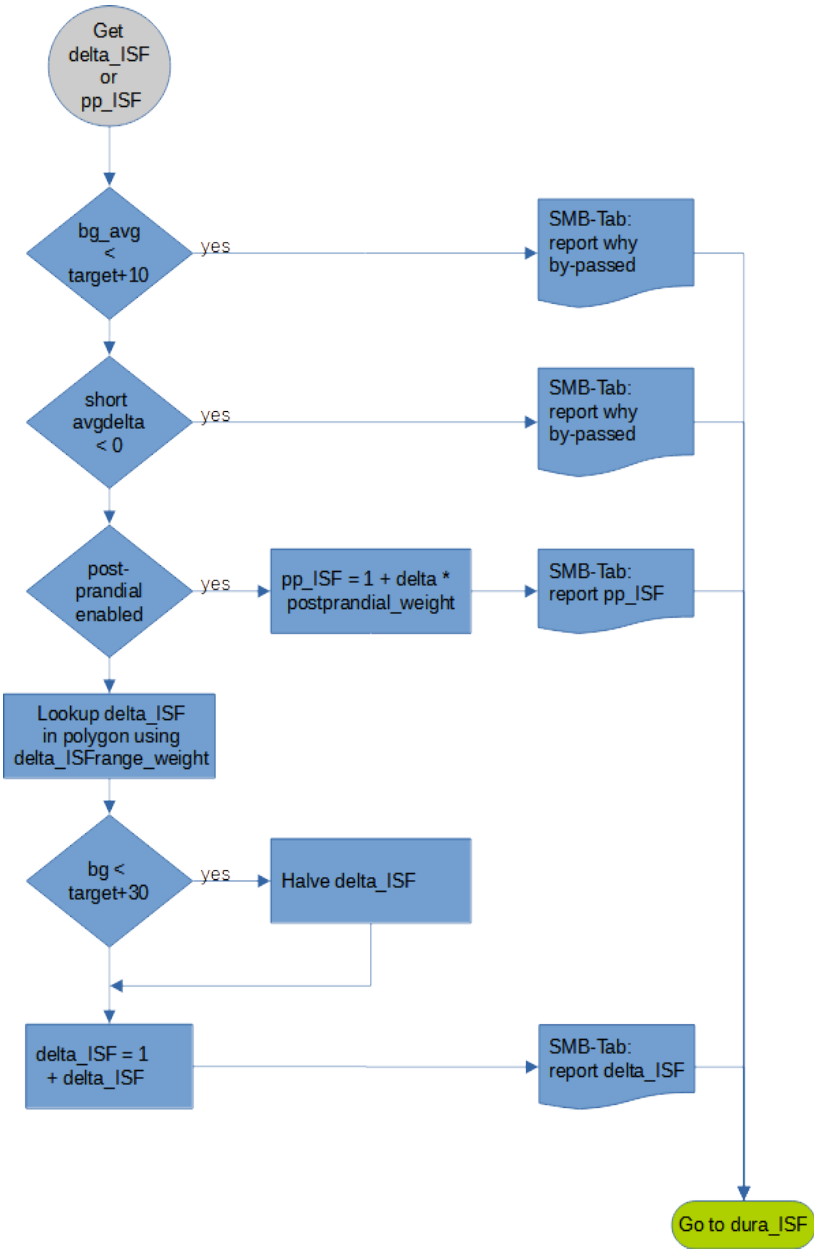
There is a special case possible, namely below target i.e. when bg_ISF<1. ISF will be weakened and there is no point in checking the remaining effects. Only with positive acceleration the weakening will be less pronounced as that is a sign of rising glucose to come soon.

pp_ISF and delta_ISF determination and its impact

There are two alternative effects in autoISF based on delta. The first one becomes active during meal absorption and was introduced to help users with gastroparesis. They can define a time window (*pp_ISF_hours*) during which this effect is active. Alternatively any user can set *enable_pp_ISF_always*=true to activate it all the time. The latter is also the recommended setting for UAM users because in their case no meal start can be detected. Given a positive *short_avgdelta* and average glucose being above target+10 the result is:

pp_ISF = 1 +
delta * pp_ISF_weight.

As a starting value for *pp_ISF_weight* use 0.005. Observe the reactions and check the SMB-tab before you increase it with care. A weight of 0 disables this contribution.



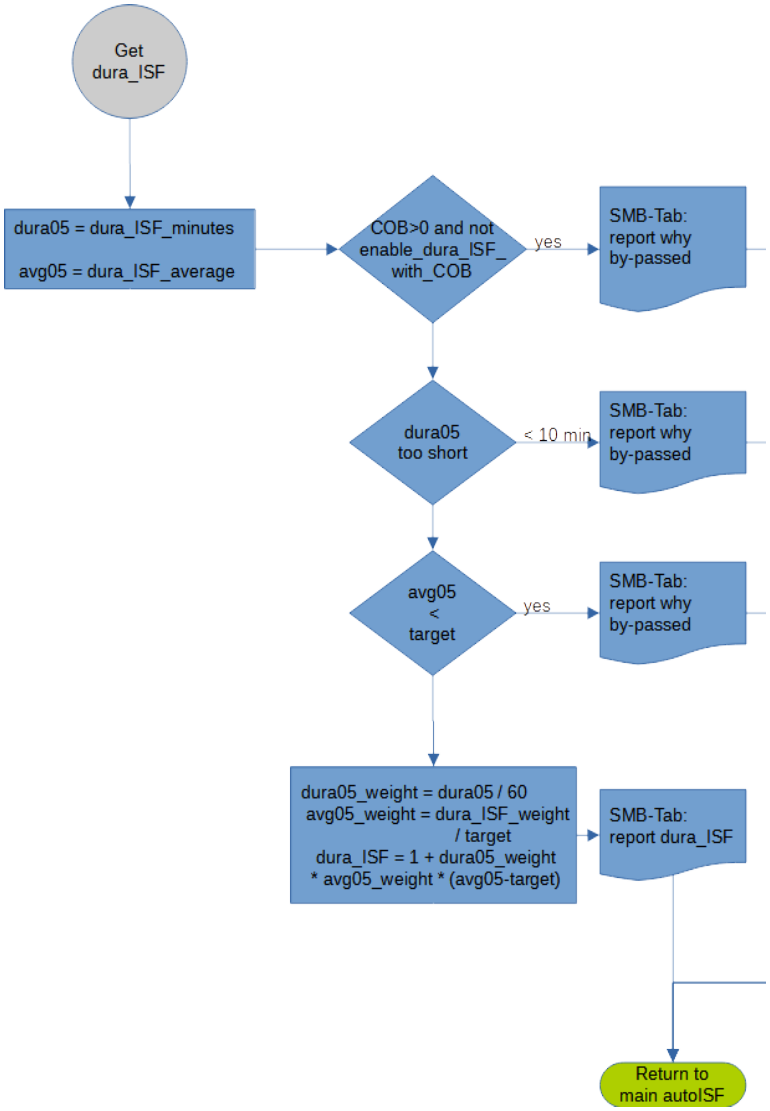
If postprandial settings are not enabled then the alternative method mimics what AAPS users did in defining automation rules that modify profiles or targets based on the level if delta. As in the case of bg_ISF again a polygon is used to lookup an ISF modification. For safety reasons this modification is halved (target_penalty) if bg<target+30. Finally the *delta_ISFrange_weight* is applied:

POLYGON x_data (delta)	POLYGON y_data (Lookup)	delta_ISF with weight=1.0	delta_ISF with weight=0.06
2	0	1	1,000
7	0	1	1,000
12	0,4	1,4	1,024
16	0,7	1,7	1,042
20	0,7	1,7	1,042

delta_ISF = 1 + *delta_ISFrange_weight* * *target_penalty* * *delta_polygon_Lookup*

As a starting value for *delta_ISFrange_weight* use 0.02. Observe the reactions and check the SMB-tab before you increase it with care. A weight of 0 disables this contribution.

dura_ISF determination and its impact



This is the original effect of autoISF in action since August 2020. Because autoISF is now a toolbox of several effects this original effect was renamed *dura_…* It addresses situations when

- glucose is varying within a +/- 5% interval only;
- the average glucose (*dura_ISF_average)* within that interval is above target;
- this situation lasted at least for the last 10 minutes (*dura_ISF_minutes)*.

This is a classical insulin resistance and is typically caused by free fatty acids which grab available insulin before glucose can. Quite often user get impatient in such a situation and administer one or even more rogue boluses. Again and again that leads to hypos later which the dura_ISF approach avoids if carefully tuned.

The method is active if

- no COB is detected like long-time after a meal or in pure UAM mode;
- or *enable_dura_ISF_with_COB*=true, i.e. even while a meal is still being absorbed.

The strengthening if ISF is stronger the longer the situation lasts and the higher the average glucose is above target:

$$\text{dura\_ISF} = 1 + \frac{\text{avg05-target\_bg}}{\text{target\_bg}} * \frac{\text{dura05}}{60} * \text{dura\_ISF\_weight}$$

where

avg05 = dura_ISF_average
dura05 = dura_ISF_minutes

The user can apply his personal weighting by using *dura_ISF_weight*. Start cautiously with a value of 0.2 and be very careful when you approach 1.5 or even higher. By using 0 this effect is disabled.

CAUTION for users with Libre CGMs: There can be error states in your CGM which look like long lasting resistance in this context. Therefore Libre CGM users should use *dura_ISF_weight* = 0. This way they can still benefit from the other three effects described before.

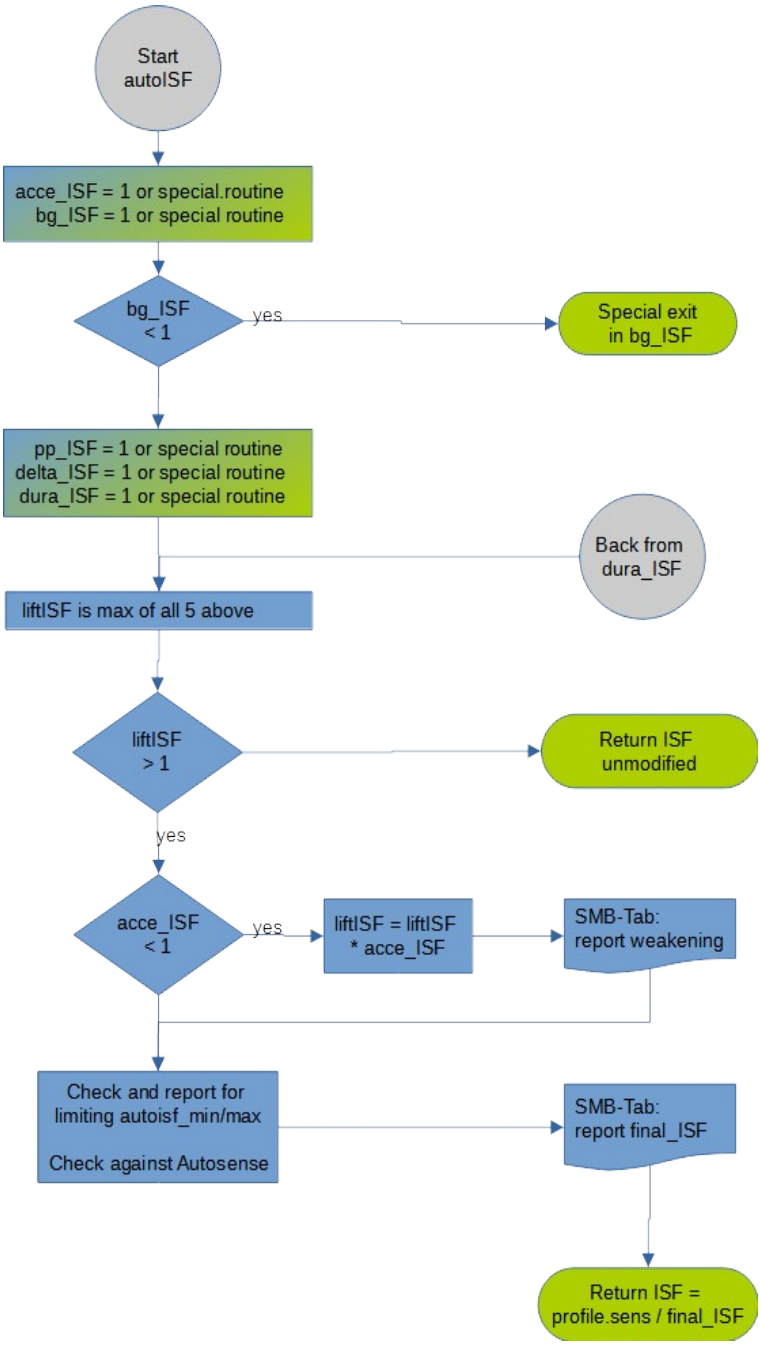
The combined result from the above factors

Now that the factors for all 4 effects are known how to deduce an end result? The normal case is to pick the strongest factor as the one and only factor to be applied. Here autosense is also part of the game. But how about the exceptions, i.e. when different factors pull in different directions? In order of precedence they are:

- bg_ISF < 1, i.e. glucose is below target
  If acce_ISF>1, i.e. glucose is accelerating although below target, multiply both factors as a trade-off between them.
  Then use the weaker of bg_ISF and Autosens as the final sensitivity ISF.

- acce_ISF < 1, i.e. glucose is decelerating while all other effects want to strengthen ISF.
  In this case the strongest of the remaining, positive factors will be multiplied by acce_ISF to reach a compromise. This overall factor will be compared with autosense and the stronger of the two will be used in calculating the final sensitivity ISF.

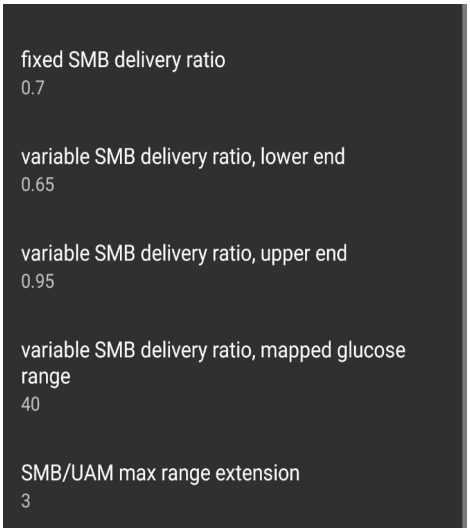
In all of the above the autoISF limits for maximum and minimum changes will also be applied.



Adapting SMB delivery

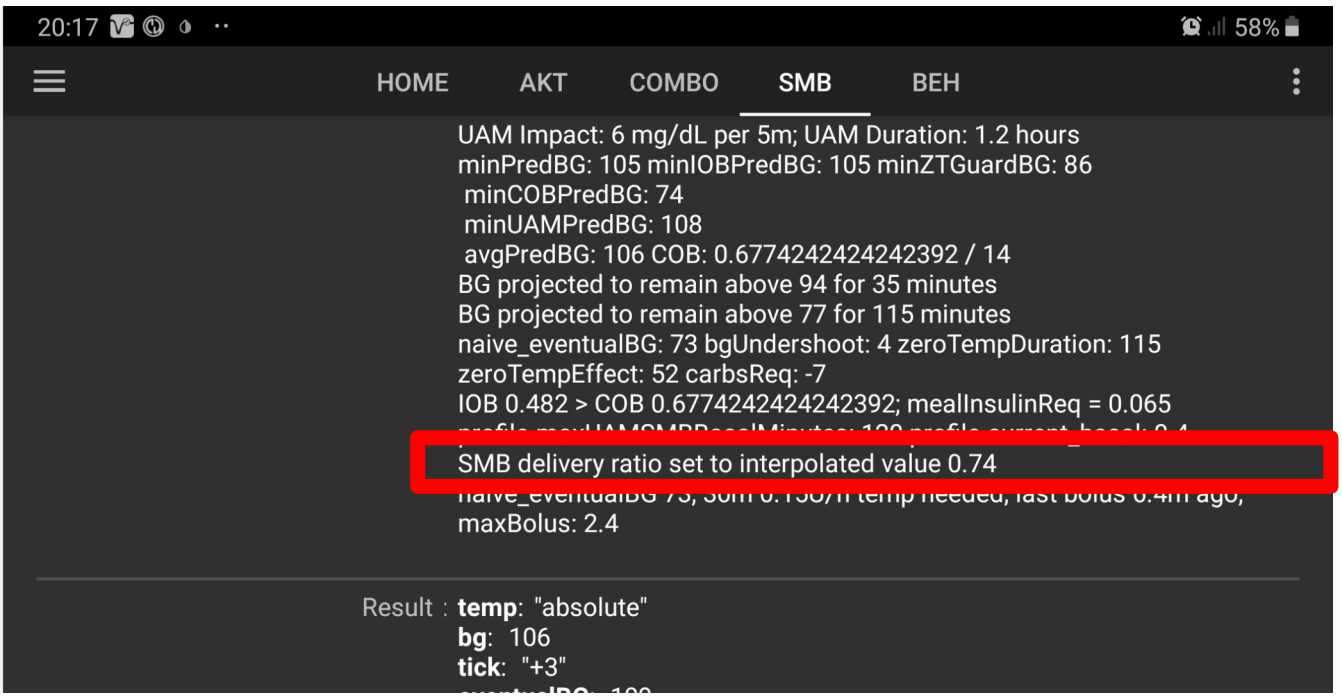
After reading all the methods to strengthen ISF you may wonder whether your maxBolus will always allow as much SMB as requested by the loop with such stronger ISF. This is especially important for pure UAM mode, where you want a few but strong SMBs as soon as meal absorption is detected by acce_ISF in order to catch up with pre-bolus and meal bolus in standard use of AAPS. Several extensions in autoISF can be used to get there:

- *smb_delivery_ratio* is normally hard coded as 0.5 of the insulin requested. This is a safety feature for master/follower setups in case both phones trigger an SMB in the same situation. If this does not apply in your case you may increase this setting to a value above 0.5 and up to even 1.0 if you are very courageous. Best is to leave some margin like the max delivery setting in the calculator.



- Alternatively to a higher but fixed ratio you can use a linearly rising ratio, starting with *smb_delivery_ratio_min* at *target_bg* and rising to *smb_delivery_ratio_max* at *target_bg+smb_delivery_ratio_bg_range.* If *smb_delivery_ratio_bg_range=0* then this linear rise is disabled and the above *smb_delivery_ratio* is used instead.

- smb_max_range_extension is a factor that increases the current *maxSMBBasalMinutes* and *maxUAMSMBBasalMinutes* beyond the 120 minute limit.



Attachment: Table of all autoISF settings

Name	Use	Min – Max	Default	Useful initial value
Settings related to autoISF as a whole				
enable_autoISF	Use any of the autoISF methods	False - True	False	
autoISF_min	Lowest ISF factor allowed	0.3 – 1.0	1	0.7 like autosense_min
autoISF_max	Highest ISF factor allowed	1.0 – 3.0	1	1.2 like autosense_max
Settings related to acce_ISF, i.e. related to glucose acceleration				
bgAccel_ISF_weight	Strength of acce_ISF contribution with positive acceleration	0.0 – 1.0	0	0.02
bgBrake_ISF_weight	Strength of acce_ISF contribution with negative acceleration	0.0 – 1.0	0	0.02
Settings related to pp_ISF and delta_ISF, i.e. glucose delta				
enable_pp_ISF_always	Flag to use it even without COB	False - True	False	True for UAM mode
pp_ISF_hours	How many hours after start of a meal the effect is active	1 – 10	3	6 for gastroparesis
pp_ISF_weight	Strength of effect	0.0 – 1.0	0	0.005
delta_ISFrange_weight	Strength of effect outside the above postprandial time window	0.0 – 1.0	0	0.02
Settings related to bg_ISF, i.e. glucose deviating from target				
lower_ISFrange_weight	Strength of bg_ISF effect when bg<target	0.0 – 2.0	0	1
higher_ISFrange_weight	Strength of bg_ISF effect when bg>target	0.0 – 2.0	0	0.2
Settings related to dura_ISF, i.e. related to glucose “frozen” at high levels				
enable_dura_ISF_with_COB	Enable dura_ISF even if COB is present	False - True	False	
dura_ISF_weight	Strength of dura_ISF effect; will also grow with time above target	0.0 – 3.0	0	0.2
Settings related to SMB delivery size				
smb_delivery_ratio	Increase the AAPS standard 0.5 of Insulin required	0.5 – 1.0	0.5	0.6
smb_delivery_ratio_min	Minimum for linearly rising ratio at bg=target_bg	0.5 – 1.0	0.5	0.5
smb_delivery_ratio_max	Maximum for linearly rising ratio at bg=target_bg + smb_delivery_ratio_bg_range	0.5 – 1.0	0.9	0.8
smb_delivery_ratio_bg_range	Width of bg range to reach maximum ratio	0.0 – 100	0	40
smb_max_range_extension	Multiplier for maxSMBBasalMinutes and maxUAMSMBBasalMinutes	1.0 – 5.0	1	1.5

Attachment: Screenshots of autoISF menu