2. General Settings for Full Closed Loop V.2.31

Please note that with autoISF 3.0 you are in an early-dev. environment, where the user interface is **not optimized for safety** of users who stray away from intended ways to use. Good safety features exist, but these are only as good as the development-oriented user understands and implements them. This is not a medical product, refer to disclaimer in section 0

Available related case studies:
Case study 2.1: (nothing available yet)

When in Hybrid Closed Loop, big boli were given by the user, and the loop had no business to give big ones on top. But this is fundamentally different now.

So, first we must **enable our loop to overcome the narrow safety restrictions for SMB sizes** that were appropriate in HCL.

2.1 SMB Range Extention

(preferences/OpenAPS SMB/autoISF settings/smb delivery settings/smb_max_range_extension)

Full Closed Looping requires bigger SMB sizes. Setting 120 minutes of basal as max. SMB size as enabled in AAPS Master will rarely suffice.

In AAPS Preferences/OpenAPS SMB/autoISF settings/smb delivery settings, set SMB/UAM max range extention (**smb_max_range_extension**) to 2.0. That doubles the allowed max. size, for a start.

Even better, you could determine an estimate for your initial setting as in the following description, I w*ill use the symbol, **.....** , to denote where you would use your numbers. My numbers that I use for the same situation will be in parentheses (U).*

*In full loop, you want to get at least half of our required meal bolus in 10 minutes, through 2 SMBs. To do that, you need ..... U (2 U) per SMB on average, and because the bolus sizes tend not to be equal when requested by the Loop, you should have at least **.....** U (3 U) as the allowable SMB size. Your hourly basal is around **.....** U (0.6 U), i.e. AAPS Master will allow a max. 2 times that hourly basal which = **.....** U (1.2 U) per SMB, because of the max min of basal setting's max 120 min. size limitation.)*

The profile helper in section 4.6 might be available for doing this calculation, and for a cross-check.

If you have an *extremely low hourly basal* rate, you may have to use a *higher* range extension along with adjusting settings such as bgAccel_ISF_weight, as discussed in section 4.2

To avoid the requested SMBs reduced in size by safety settings, you need to make further setting adjustments in AAPS/ Preferences:

2.2 Max and Min autoISF Ratio

(preferences/OpenAPS SMB/autoISF settings/autoISF_min and autoISF_max)

Set **autoISF_max** = 2.0

This allows *up to doubling* of ISF aggressiveness if "requested by the "… _ISF_weights" (see section 4). You can sharpen (elevate autoISF_max) further, later.

For situations of reduced insulin sensitivity, you must specify, in your settings (preferences) also what your *lower* limit (for weakening of ISF, compared to profile_ISF) may be.

autoISF_min should be set to 0.5, or even lower.

2.3 SMB Delivery Ratio

(preferences/OpenAPS SMB/autoISF settings/smb delivery settings/smb_delivery_ratio)

Increase fixed **smb_delivery_ratio** from 0.5 to 0.6.

This results in the algorithm generally demanding 20% more insulin (in the calculation it does every 5 minutes). This can be sharpened significantly more (going toward 1.0), later.

The smb_delivery_ratio *can also be set dynamically* (changing with glucose level).

This seems a feature geared more towards hybrid closed loop applications.

2.4 Safety Against too Aggressive Settings: iobTH

(preferences/OpenAPS SMB/ autoISF settings/smb delivery settings/iob_threshold_percent

…which gets multiplied with preferences/OpenAPS SMB/Maximum total IOB OpenAPS can't go over (U)

A safety net is needed because autoISF shoots big SMBs when glucose levels begin to rise, and you do not want to bounce into your ultimate iobMAX safety setting too often.

Therefore we install an **iob threshold** (iobTH) which, **when**, and as long as, **exceeded, shuts SMBs off.**

autoISF 3.0 totally changes how iobTH is accessed and modulated. (Before it had been done via an Automation).

iobTH is now a parameter in **AAPS preferences**, defined there as fraction of your set maxIOB:

/OpenAPS_SMB/autoISF_settings/Full_Loop_settings: iob_threshold_percent,

First, check whether your **maxIOB** is set reasonably, near the max level of iob you had ever needed in your past looping history*; for instance 10 U*.
Then look at your meal spectrum, and what bolus size, and iob level (including SMBs) was useful in high carb meals to control your glucose (*for instance 8 U*).

Then assume you want to approach no more than about 75% of that level by rapid SMB „fire“, after a meal related bg rise is seen (*for instance 6 U*).

That gives you the **iob_threshold_percent** to enter = desired SMB amount / iobMAX (*In the example it would be = 6 U / 10 U = 0.6*). In section 4.7 a profile helper might be available for more guidance.

The default iobTH is then:

default set iobTH = iobTH_percent x maxIOB

Note that

- the last SMB given **can exceed** that threshold by up to **+30%** of the effective iobTH.
- thereafter (until iob falls below effective iobTH) only %TBRs supply more insulin, if calculated as still required.
- in low carb meals, that iobTH level should not be reached => the autoISF parameters („weights“) need to be tuned carefully, so SMB sizes are *not always* huge and bounce against the iobTH restriction, but show different behaviour for different meals
- autoISF 3.0 and higher contains a function to **auto-adjust** iobTH **with TT** set: Dynamic iobTH, The formula for the resulting **effective iobTH** is complicated and also depends on whether the exercise mode is active. See sections 3.3 and 6.1.3, and example in case study 6.2
- Likewise, a **%profile** setting < 100% (or > 100%) will proportionally lower (or elevate) the iobTH that will be used. It will automatically revert to the default iobTH after the temporary % profile expired

  effective iobTH = % temp.profile x iobTH

  If both, a % profile and a TT are set, both effects multiply.

Note: With older autoISF variants, the odd TT SMB shut-off above iobTH in an Automation had been a bridging solution. Now, shutting off SMBs due to iobTH **does not rely on an odd TT or target** running.

The even/odd logic remains useful in different contexts, see e.g. section 5.1.2 and 5.1.3)

All above discussed settings must be made in AAPS/preferences (except for the temp. modulations of sensitivity which can be done from the AAPS home screen via %profile or via TT inputs).

To enable going into FCL mode, additional settings must be made in /preferences/Open APS SMB/autoISF, as discussed in detail in sections 3-4.

Once all this set-up is in place, you can enter/exit FCL (for initial tuning or for everyday utilization) via

- in AAPS/Preferences/OpenAPS SMB/autoISF settings/"**Enable ISF adaptation by glucose behavior**" ON / OFF
- or, by just tapping on the **violet**/ green **closed loop icon** of your AAPS home screen, after this feature (section 5.3.1) becomes integrated in a future autoISF version update.

2.5 EatingSoon TT ?

Your FCL works best if you start meals at below-target glucose values, and ideally have a bit of positive iob at meal start. Also, a low temp. glucose target helps making SMBs (that „aim at it") a bit bigger.

Setting an EatingSoonTT well ahead of meal start therefore is in principle a good idea

- If you have relatively fixed meal time slots in the 24 hours of the day, you could set the **target glucose** values **in your profile** accordingly. So *e.g. 11-15h target 76 mg/dl* if you almost always start a lunch between 11:45 and 14:30h. (*If you do exercise or physical work in that time, this would be too aggressive, and probably also un-necessary*).
- If you have rather irregular habits, it might be worthwhile to **manually** set an **EatingSoonTT** (which is quite time-uncritical) well before the start of a meal, or even (latest) when the first SMB is about to be triggered by your loop. – **Or, just forget about it:**

However, if (as to be expected) your loop anyways always regulates you down to near-target, the effect from setting an EatingSoonTT will be limited:

- If your meals are spaced by a couple of hours your glucose should not be elevated as you approach the next meal …

- If you eat more at a still elevated glucose, your loop should provide you with “a balancing” iob (and hence a prediction to get to target soon) …

…In both cases (which are the prevalent norm) setting an EatingSoonTT would only have a very minor effect.

The good news therefore is: Setting an **EatingSoonTT is not required** for autoISF FCL.

My preferred solution (*as e.g. consistently used in* case study 4.3) is as follows

I am never setting an EatingSoonTT ahead of the meal, but just - automatically – have my loop set a low TT just around the time when it gives first SMBs (“to orient the calculated insulinRequired towards a more aggressive target”).

For this you can define an **Automation** like: CONDITIONS: likely meal time of day + 1st sign of a beginning meal + iob under (?) U (to kick in only in initial phase) + no TT running => ACTION: set TT=74 mg/dl for ~30 minutes

Note for users of previous autoISF versions: You do not need the other Automation any longer, that previously was needed to shut off EatingSoonTT before iobTH is reached. autoISF 3.0 ff integrates iobTH and the associated temp. SMB shut-off into the algorithm, rather than requiring yet another user Automation for that.

FCL works in principle also without setting an EatingSoonTT. Try for yourself whether you lose any performance via the totally hands-off way.