Managing the late phase of a meal

Meal Management, section 3 – V.0.2 (draft)

needs more work (but is better than preceding ppt and therefore made temp available

Now we are getting to the **late phase of a meal** which is where the **loop really takes over the management**, and takes care of the late carbs and fat/protein units (FPU). How that is done differs by the used algorithm, so we can only roughly sketch the common key points, and must give some references for further study.

We then touch on common troubles in the late phase, notably hard-to-correct high values, and how to deal with those and with other troubles.

We close this chapter with a critical view on some interventions into the loop, which may or may not be smart when we are dissatisfied with our loop.

Table of contents

Carb entries for late phase

For the late phase, i.e. the time window when our meal bolus lost most of its power (Rule of thumb: 75-80% is lost about 3 hours after bolus for Humalog; or 2.5 hours for Fiasp; or 2 hrs with Lyumjev) we must enter the carbs that will be absorbed **after** this time window. Roughly, carbs exceeding ~ 60g (with slow insulins 75-90g) come to absorption now.

We just inform the loop of how much (gram amounts) and about when (absorption times) more carbs will be absorbed and require insulin activity. We do not bolus for those, and – this may depend on your looping system – you do **not** enter them into the calculator but via a special „eCarbs“ entry (e.g. via Carbs button in AAPS).

The **„eCarb“ amount** consists of:

- Carbs exceeding ~ 60g (with slower insulins: exceeding 75-90g)
- plus a „FPU“ contribution from proteins and fats

  …which can be counted as grams of e-Carbs (using factors 50-60**%** * g (protein); and 10-20**%** * g (fat)). Also lower %age for protein and higher %age for fats, or a different approach via kcal, is sometimes suggested (see next chapter, “FPU”)

Notably iOS Loop systems also require details about **absorption times.** These are derived for the e-Carbs by major food component, and considering:

- Limited total absorption capacity (~ 30g/h)
- Absorption is slowed by fibre
- Absorption is slowed by fat
- Proteins take long to digest

Carb and e-Carb **entries** (amounts and absorption time info) can also be partially automated and connected to food databases like Sugarmate.

This can be valueable for iOS Loop.

As we will further elaborate on in section 4, detailing the eCarbs carefully is not necessary with oref(1) systems.

Fat Protein Units (FPU)

There are two principal approaches how to count protein and fat:

(1) Calculation of **e-Carb** equivalents **via kcal from fat and protein**.

With the following table we like to show for a meal rich in protein, how fat and protein can be counted into late carbs, and what reasonable late (e-)Carb entries would result (2g from fat + 23 g from protein, plus a bit also considered in the calculator or bolus wizard by adding 1g resp. 4g in for the first 2.5 hours):

Formula (1) to calculate g eCarbs

FPU = (kcal from fat + protein) / 100

eCarb ~ #FPU * 5 g/FPU

meal rich in proteien

component	g	% carb	carb	g carb in 1st 2,5 h	late carbs	kcal/g	kcal	FPU
Carb	**34**	**100%**	34	34	0	4,1	139	
Fat	**33**	**10%**	3,3	1	2	9,3	307	491
Protein	**45**	**60%**	27	4	23	4,1	185	4,9
Alcohol	**20**	**0%**				7,1	142	
total			64	39	25	*5	773	

Late carbs from fats and proteins can be calculated:

a) via FPU, see yellow fields
FPU*5 gives the grams eCarbs
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5686679 ;

The top right yellow box shows how FPUs (fat-protein-units) are calculated via the kcal/g from these components.

To check for overall plausibility, you might occasionally add such a table up for an entire day, and see whether it is plausible with your daily caloric needs.

The protein-rich meal delivered 491 kcal from protein and fat, which, divided by 100, is 4.9 FPU.

There is a controversy about which factor to use to convert these FPU into grams of eCarbs (red arrow).

A (safer) factor of *5 is suggested e.g. by https:// www.ncbi.nlm.nih.gov/pmc/articles/PMC5686679

while *10 (orig."Warsaw method") is used by others, e.g. Pediatr Diabetes. 2009;10(5):298–303. [PubMed]

As caloric information on components on your plate is not easy figured out, in daily T1D life mostly the other route is taken:

Formula (2) to calculate g eCarbs

eCarb = 50…60% * protein + 10..20% fat

(2) Using **factors 50-60% per gram of protein**, and **10-20% per gram of fat**.

As our example in the table above shows, this (green arrows, effects adding up to 25g eCarbs) can be fairly consistent with the caloric-derived values, when using the *5 factor (red arrow arriving at 25g eCarbs, too).

As the *factor, also the **% applied for fat** is discussed quite controversially.

This is actually quite understandable, because fats bring about two **effects that increase insulin requirements** in the late meal phase:

- There is an additional effect from fat (and also from fibre) to slow down absorption of “real carbs”, so it may seem that fats translate to a higher % into late carbs.
- Often, fats create complications several hours after a meal through **intermittent insulin resistance** at high glucose values).

Important here is, that the **reasonable conversion factor depends on the used strategy** (A or B) to deal with these effects:

- Strategy (A): Apply a **bigger %** for the conversion, and in consequence **exaggerate carbs** from fats, and this way also gets them covered with more insulin.
  For iOS Loop, this method (A) is preferable.

- Strategy (B) is to use the suggested **low** 10 … max 20 **%** for fats; **plus** apply **stronger ISF** temporarily, only when glucose is stuck on high values.
  This method (B) is preferable for oref(1) loops, which do not react directly on grams of „e“-carbs entered, but do have special features that can even automatically address temporary insulin resistance (dynamicISF, Automations, and especially the dura_ISF component of autoISF).

--- References: --

Balancing Carbs, Protein, and Fat. Clinical review by Mary Hanson, registered dietitian. Kaiser Permanente. 03/01/2014 https://wa.kaiserpermanente.org

General overview, effects from proteins and fats: https://www.youtube.com/watch?v=4_5K9zH9CFc

FPE calculator via GooglePlay: EasyFPU. FPU calculators für iOS:: https://www.icloud.com/shortcuts/ac31808d809f40e1a9e63175f27152ac ICLOUD.COM Loop FPU
Robert's video explaining his method: https://youtu.be/k8amkz9E66k Loop FPU Food Entry (iOS Shortcut version)

Food databases: https://sugarmate.io or Fleisch - Tierische Produkte - Kalorien – Fddb

Reference for Automations:
https://androidaps.readthedocs.io/en/latest/AdvancedOptions/FullClosedLoop.html#stagnation-at-high-bg-values

…and for dura_ISF: Section 4.5.5 in: https://github.com/bernie4375/FCL-potential-autoISF/blob/FCL-e-book/04_Meals.ISF-weights_FCL-book_V%204.7.pdf

Insulin for the late phase

In the late phase of the meal, which can last several hours, the loop must provide insulin as required – but without the „tails“ of the aggregated TBRs and microboli dipping the glucose level into the hypo zone later, when the flow of to-be-absorbed carbs has ceased.

Based on the insulin activity curves, the loop has always a pretty good picture how much insulin activity it has to work with in all upcoming 5 minute segments, and how to increase it via extra TBR>100%, or via Autobolus/SMB, or to decrease it via TBR<100%.

The challenge then is the carb absorption.

The carb absorption can either be inferred from the **glucose** movement in light of insulin activity. That is the basic approach of the oref(1) algorithm (more see section 1.2 in https://github.com/bernie4375/HCL-Meal-Mgt.-ISF-and-IC-settings/blob/HCL-.-settings-main-repo-(pdf)/IC%20(carb%20ratio)_V.3.1.pdf).

Or absorption follow largely from inputs about **carbs eaten** and their absorption times *), which is the basic approach of iOS Loop. More see https://www.youtube.com/watch?v=4_5K9zH9CFc

Of course, in the end, both systems take into account all available information. As user errors as well as a myriad of secondary influencing factors play into it, the so-called **deviations** play a big and similar role in both approaches . Deviations try to explain discrepancies the model sees, by saying, one of the parameters, e.g. carbs, must have been misjudged (in quantity or absorption time). Depending on which factor „gets the blame“, **predictions** for the next 2 hours can look vastly different … but **will converge as uncertainties decrease** over time.

Predictions are the key part that drives the loop‘s conservative behavior regarding **hypo** prevention **from the „tail“ of** remaining active **insulin** (from insulin that had been used earlier “to fight high bg”).

How calculating the required and to-be-delivered (or withdrawn) insulin is exactly done, differs in many details between the open source loops, but it can be looked up **). Of note, in AAPS much of it can be read (over like 5 full screens on the smartphone) in the SMB tab, every 5 minutes, if one desires to completely understand the loop's decision.

Loops differ also in how they can deal with fluctuations of user's **insulin resistance**. DIY have in-built (AAPS) or add-on options (IFTTT, Automate! for iOS Loop) to automate ISF and other parameter modulations. Loops may also include code that self-adjusts (Autosens, Autotune, dynamicISF, autoISF, Machine Learning/Artificial Intelligence), although with more or less hours of time-delay

-------------------------------- footnotes---

*) In one minute, Kenny explains why **absorption time** matters in iOS Loop:. https://youtu.be/UGtwnugWfKQ.

“Absorbing too fast, which is what you are seeing, generally leads to highs. If it doesn't, then your ISF is probably far too low and helps compensate for the incorrect settings.

When carbs absorb too quickly, the settings that could be impacted are:

1) basal too weak. Not enough insulin all around.

2) Carb ratio too weak (big). Most obvious from being high as well.

3) ISF too low/small. Sometimes leads to highs, but not if way too low.” (Kenny F., L&L-FB, 16Jan'22)

) More on **algorithms: https://www.diabettech.com/looping-a-guide/comparing-the-loop-and-openaps-algorithms/ and (partially cited below): https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8087942/

Calculation of Insulin_Required (differs between algorithms):

(a) **AAPS** (and similar OpenAPS, Trio, iAPS): combine w preceding page and with 1.2 of IC..pdf

The current heuristic algorithm of oref1 generates multiple future blood glucose predictions based on (i) action of approximated insulin remaining in the body; (ii) the scenario in which all carbohydrate intake ceases, and the system stops insulin delivery; (iii) action of approximated carbohydrates remaining in the body; and (iv) unannounced meal absorption. The predictions are then combined to estimate the lowest predicted glucose, and insulin delivery is adjusted to ensure that the local minima remain within a pre-specified target. The user, similar to standard insulin pump therapy, inputs their personal basal rate, carb ratio (IC), insulin sensitivity factor (ISF), and curve of insulin pharmacodynamics.

AAPS and the other oref loops also **enable automated insulin coverage of meals without carbohydrate announcement via the "unannounced meal" (UAM) feature**, whereby meals are anticipated based on available data suggesting an otherwise unanticipated glycemic excursion. More see chapter 1.2 in: https://github.com/bernie4375/HCL-Meal-Mgt.-ISF-and-IC-settings/blob/HCL-.-settings-main-repo-(pdf)/IC%20(carb%20ratio)_V.3.1.pdf

Other features, not specific to unannounced meals, such as "**auto-sensitivity**" respond to glucose fluctuations beyond the scope of predictions.

> The actual used data (every 5 minutes) can be looked up „live" in the AAPS SMB tab; or, for any 5 minute segment in the past, in YOUR AAPS logfiles; -> phone storage/emulated/0/Android/ data/ info.nightscout.androidaps/files. There: „AAPS.log" contains the most recent data, and „AAPS_yyyy-mm-dd_(time in univ.time zone)_#.zip" are the „historic" ones). Note this is not easy-to-read stuff. There are tools for analysis that load desired files from a time window of interest to you. If you look in https://github.com/ga-zelle/APS-what-if__/: The **emulator** offered there (for AAPS) even allows what-if analysis on some changed settings you may like to investigate on paper … or even „live", with a voice telling you every 5 minutes what would have been done differently! So, even that could be done, before jumping right into trying it out.

(b) iOS Loop See https://www.youtube.com/watch?v=4_5K9zH9CFcv

The other open-source AID system, iOS Loop, uses a different algorithm that employs coincidence point control (model predictive control),[16] and generates a single future prediction based on insulin delivery history, **carbohydrates entered**, and other entered settings, including the basal rate, IC, ISF, and curve of insulin pharmacodynamics. When used in humans, users enter not only carbohydrate counts but also an estimate of the **absorption time** of that particular meal or food. Loop uses two forms of short-term adaptation called "glucose momentum" and "retrospective correction" to enact temporary basal rates to push the projected glucose toward a specified target range. Glucose momentum uses the 5-min rates of change from the prior 20 min to influence future predictions, with most weight given to the most recent rate of change. In retrospective correction, glucose differing from the predicted value causes that difference to be added to the next prediction. An extension of this feature called “integral retrospective correction” takes not only the difference but also the accumulated differences into account for more rapid adaptation.

Loops also differ in how they can deal with **fluctuations of the user‘s insulin resistance**. (is already touched on in preceding sections .. try to unite, and add ActivityMonitor) DIY have in-built (AAPS) or add-on options (IFTTT, Automate! for iOS Loop) to automate ISF and other parameter modulations. Loops may also include code that self-adjusts (Autosens, Autotune, dynamicISF, autoISF, Machine Learning/Artificial Intelligence), although with more or less hours of time-delay.

Why is it often difficult to resolve a high post-meal glucose – and why, then, the sudden declines toward a hypo? combine better with the FPU section

A post-prandial glucose rise can lead to insulin resistance (elevated insulin need). This often happens after fatty meals, and mechanisms behind it seem to be the following:

Stage 1: Free fatty acids (FFA) can block insulin receptors . More insulin is needed „to shovel them free“ (A post-dinner walk can be very helpful, too).

Stage 2: Lipolysis: Insulin deficiency leads to cleavage of body fats into FFA and other components in order to obtain energy. When insulin deficiency is resolved, also the FFA must be removed (see stage 1)
In the moment when FFA were successfully removed using insulin, insulin sensitivity returns to normal. Any insulin becoming active now (be it from a „rage bolus“, or also what remains active from the last micro boli) can rapidly bring down glucose, sometimes ending in a hypo.

AAPS offers the option to customize the loop control by the user defining an „Automation“: If a certain „pattern“ was recognized, the loop can be told to react, when seeing such pattern, in certain ways:

> Example: If in (def. time window after a meal) a (def. duration, slope) incline happens, change to other factors (e.g. higher-.% profile „override“) for (duration, or until glucose smaller than …). See also https://androidaps.readthedocs.io/en/latest/AdvancedOptions/FullClosedLoop.html#automated-big-smbs-at-bg-rise

In all open-source loops, similar things can be automated with 3rd party add-on apps, like IFTTT or Automate!

------------------------- references -- -----

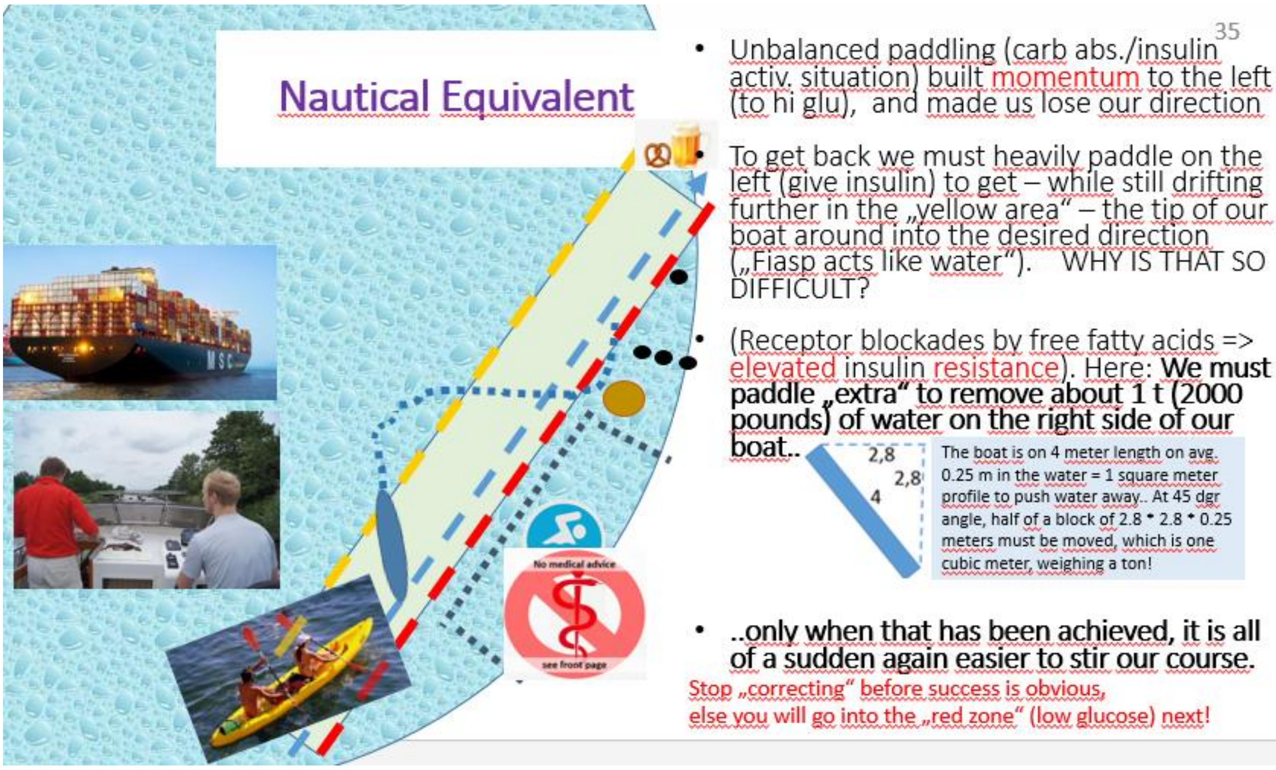
Source about FFA (in German): Loopercommunity.org/ LongboatAline / Dr. Teupe. Mehr zu FFS: s. S.19-25 in: geb.uni-giessen.de › volltexte › pdf O. Wüsten, Beeinflussung der Insulinresistenz durch freie Fettsäuren

Zu Profilwechsel bei Resistenz siehe auch die gute Diskussion in: https://de.loopercommunity.org/t/profilwechsel-bei-resistenz/2969

Automation: see (in English) Automation, use it wisely | Diawatch

Nautical equivalent to temp. insulin resistance

The situation with temporary insulin resistance because of blocked insulin receptors that we just discussed, is paralleled in nautical terms as shown here:



We had used the nautical parallel before, to understand the implications from the sluggish and delayed nature of control. Even just the momentum from pausing after the last executed paddle stroke on the right, could make our boat, as symbolized with the blue ellipse in the picture, turn away from the blue target line.

To get back on course, we would have to paddle strongly on the left (=to the right) (~give SMBs or Autoboli); still, this would keep us drifting over the yellow border (~ „Fiasp works like water at high values"…). Why?

The equivalent of the receptor blockages is here, that we need to somehow move about one ton of water from the front right side of our boat to the other side, in order to face towards our goal line again with our boat. (Calculation see sketch and blue insert)

Only when heavy paddling has achieved this, it is all of a sudden easier again (~ sudden return to normal insulin sensitivity) to stir our course.

It is a typical beginner mistake to keep „correcting" until success is obvious,. But that is too much, and the boat will head into the red zone next. (~ glucose roller coasters triggered when neglecting the delayed control implications – as the loop algorithm uses in ist predictions).

Trouble shooting for the late meal phase

If you frequently face troubles in the late phase of a meal, one logical approach would be to first solve major problems that might carry over from the early meal phase, so you do not start with a super high glucose value into the late phase (and also not with a burden of too high iob, in a meal situation with not many eCarbs to deal with).

Generally it is a good strategy to develop a stable loop with meals that do not push the limits. So, limiting the amounts of components with very high glucose impact (see chart p.xxx) should help to dial in all parameters right.

This table gives some basic ideas what might be wrong if certain problems are encountered.

The **late meal phase** can last for 6 and more hours; an incorrect basal rate or DIA could become noticeable, besides „the usual suspects". Problems might influence the next-following meal, and can be clearest seen in the night after dinner. The following table can only give rough guidance because the looping algorithms differ.

category	problem	likely cause	action, immediate	...for next day
glucose increases	in the beginning of the late phase	not enough active insulin (zero-temping in early phase)	„ride it out"; evtl.small bolus/Afrezza	front-load eCarbs enable bigger SMB/AB
	prolonged in the middle of the late phase	temp. resistance from fatty acids	set override /high%profile; Afrezza?	Automate a high override / %profile for a couple of AB/SMB
	stays high	ISF too weak absorption times too short too low eCarb input	evtl.small bolus/Afrezza; add carbs	lower the ISF increase absorption times input more late eCarbs
glucose dips low	towards the end of the late phase	DIA too short basal too high too much carbs entered user bolussed extra	take glucose & elevate TT	elevate DIA reduce basal improve carb input
„roller coaster"	repeatedly bouncing low	ISF too strong	take glucose & elevate TT	elevate the ISF reduce SMB/AB size limit

For super big meals with lots of fat and protein, problems might surface again, or get worse. Then your choice would be to accept that and „ride it out". Or to prepare some of the immediate or even „next day" actions from the table. My (author B.) favorite one is to provide my loop an automated response to high glucose values *) , something like „if glucose >170, switch for 12 minutes to 20% lower ISF".

Consult also „Loop tips - stuck on high": https://loopkit.github.io/looptips/how-to/high-bg/

Trouble very often is coming also from user interferences, notably from rage bolussing when the glucose seems to stay too high for too long. We were not sure whether even to include small (!!) extra boli (using the pump, or Afrezza) in the immediate remedies column. Try to manage without it, your loop should have a better idea on where things are heading, notably for hypo prevention an hour or more down the road.

--remark---*) Generally though, try to avoid „home constructing“ a loop within the loop! The justification here is, that we do know of short-term elevated insulin resistance after fatty meals, as just discussed on the preceding 2 pages, and a short-term (e.g. 12 minutes = 2 ABs or SMBs) strengthening of ISF, only when values became high after a meal, can therefore be justified (and has nothing to do with a sloppily set-up loop being patched over with new tricks). Actually this flexibilization is being developed as a user tune-able new feature within the AAPS algorithm for 2022 („autoISF“); but users of current iOS or oref(1) master looping software can meanwhile coarsly „home construct“ it via an Automation add-on.

Co-managing the loop in the late phase

While the user took center stage with her/his bolus in the early meal phase, now, in the late phase, the loop is supposed to manage the remaining meal time. However, we are often tempted to „help“ the loop.

The traffic light symbol on the left of the table symbolizes what we can (or maybe better, should not) do in a desired co-manager role.

	Interference	Example	Note
	setting temp. targets (early phase)	EatingSoonTT	even if not reached: insulin given by the loop considers the TT
	setting temp. targets	ActivityTT for sports after the meal	Difficult because the loop front-loads insulin and zero-temps a lot in the late meal phase to hit normal target. A sportsTT would need to be set very early, leading to a weaker overall meal management. A good alternative would be a normal managed reduced meal size, then a sports snack with a late set ActivityTT
	setting over-rides/%profile	(Automation with) elevat-ed %profile for insulin resistance from FPUs	(see page 33)
	Afrezza	for stopping glucose rise	(see next page)
	changing key loop parameters depending on glucose level	glucose ⇔ ISF modulation (e.g. via an Automation)	tuning the ISF correctly agrees better with the algorithm
	extra bolus (with info entered)	for correcting high glucose	= core job of the loop; not helpful to get the loop work right; loop will counter-regulate; still, likely ending in a hypo.

Starting from the bottom: What we really should avoid, is to give an extra „rage“ bolus when values go high and we are dissatisfied with our loop. Correcting high glucose is really the core job of the loop. Your job is to **enable** the loop do a better job next day. So better watch, analyze what you see, and adjust parameters for the next days. Your intervention would highly likely result in a hypoglycemia a bit later.

A somewhat milder interference would be when the user modulates „aggressiveness“ of the loop based on the reached glucose level; this could also be automated. However, this would go against the logic of the loop, which should really work with appropriate settings, notably ISF, without any additional patching-up by the user.

To stop a glucose rise with the inhaleable insulin Afrezza is a possible but somewhat ambivalent choice, as will be discussed on the following page.

Setting stronger ISF when temporary insulin resistance from fat happens, is a good co-managing strategy that was already discussed ref.?.

Setting temporary targets is a benign way to „nudge“ the loop in a desired direction. We had discussed this already in context of EatingSoonTT in the pre-phase. Now, in the late phase, setting a high ActivityTT might be helpful, not so much for meal management, but in case activity follows that requires room for sinking glucose values. Actually, this can be quite a challenge for people who do have meal management under good control: The loop front-loads insulin and zero-temps a lot in the late meal phase, to hit the normal glucose target. An ActivityTT would need to be set very early, but lead to a weaker overall meal management. A good alternative would be a normal managed reduced meal size, then a sports snack accompanied by a late set ActivityTT.

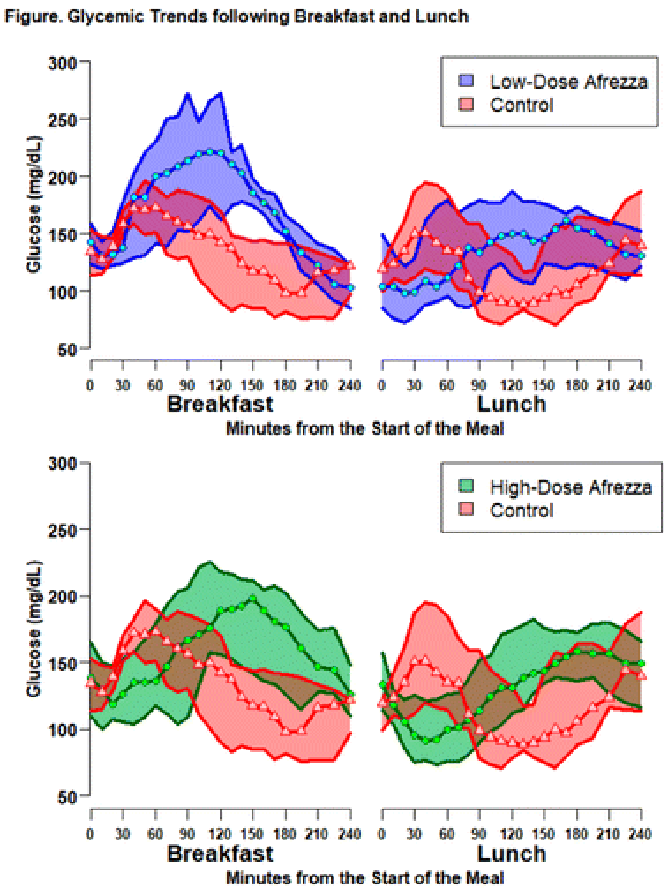
Afrezza to the rescue!

Afrezza is an inhaleable very fast (and also short) acting insulin which could be useful **to correct high glucose levels**.

In our nautical analogy: The tip of our boat has drifted to the left (blue ellipse). We move it back to the right when „my girlfriend Afrezza“ does a CANNONBALL JUMP next to the front left side of our boat. The resulting **>>>**-wave orients our boat back towards target

Pro: Afrezza addresses a temporary extra need, and even without the hours-long tail of effects.

Con: Afrezza spray is hard to dose. Also it is not adviseable to enter data into loop because the kinetics of this insulin are very different. Available data (see charts on next page) suggest that Afrezza solves a problem in the first 60 minutes of application. However, things get worse thereafter, as both, previous loop insulin AND Afrezza fade out quickly just as (around minute 90 …120 after a meal) there is still strong carb absorption requiring active (!) insulin:



Source: A. Galderisi et al, Abstract ID 290 ATTD Madrid 2020

Observations/Questions Bernie:

90 minutes **after** meal starts, bg curve is WORSE with Afrezza: *Is this because a pre-bolus with more of a activity at around 60..120 minutes is fully replaced by a faster waning spray shot? Can the blue resp. green curves, made come down in minutes 90-…-210 * via more aggressve ISF and SMBs acting on the bg rise?*

**via a second Afrezza shot?*

** via complementing the 1st Afrezza shot with a (reduced) pre-bolus?*

Why is breakfast so well managed by HCL, and much more worse with Afrezza after 60 or after 90 minutes (also when we compare to the differences seen at lunch)? *Probably we must see the insulin activity curves (Afrezza not available in AAPS, though), + estimate the carb absorption characteristics for both meals …*

Upside: *FCL w/autoISF should be able to manage the 60 .. 150 minutes situation*

In the 60 minutes after Afrezza application the loop saw bg movements but by far did not have all related iob info (Afrezza contributions missing for the calculations). It is a problem when up to about minute 60 after Afrezza application the loop sees good bg control at supposedly low active insulin; But immediately afterwards (Afrezza effects then faded away quickly), data look like suddenly super high insulin sensitivity - for which the loop can not suddenly provide active insulin. Hence the glucose rises in minutes 90…120.

The situation will also result in false Autosens (if in use), with negative consequences in the upcoming hours from skewed calculations .

Still, Afrezza can be a reasonable remedy in times. **Solving the problem at hand** as best as we can, **even if it makes the time afterwards a bit more complicated, is the name of**

the game. **We and our loop do this all the time** (for instance, by giving more upfront insulin, then reducing basal (zero-temping)).

A second shot when bg rises again could actually help, too.

But preferably, seek strategies that make you **stay better in range, and** avoid crazy extra moves with Afrezza. (Experienced navigators can do without cannonball jumps by their crew!)

Temptation to cheat the loop via „fake“ data entry

Generally, we want our loop to work smoothly, and have the best information we can give it.

To try to provoke a certain behaviour of the system via „nudging“ it with fake inputs can certainly work for some intended effect; however it is likely to create an imbalance and develop, at least within the time frame of carb absorption from the given meal, and DIA of the insulin on bord, errors and suboptimal further regulation capability of the loop. The most prevalent cases probably are:

- the user rage bolus from a pen, without announcing it to the loop
- the un-announced anti-hypo snack.

Trying to provoke a certain behaviour of the system via „nudging“ it with fake inputs can certainly work for some intended effects. BUT: Generally, we want our loop to work smoothly, and have the best information we can give it.

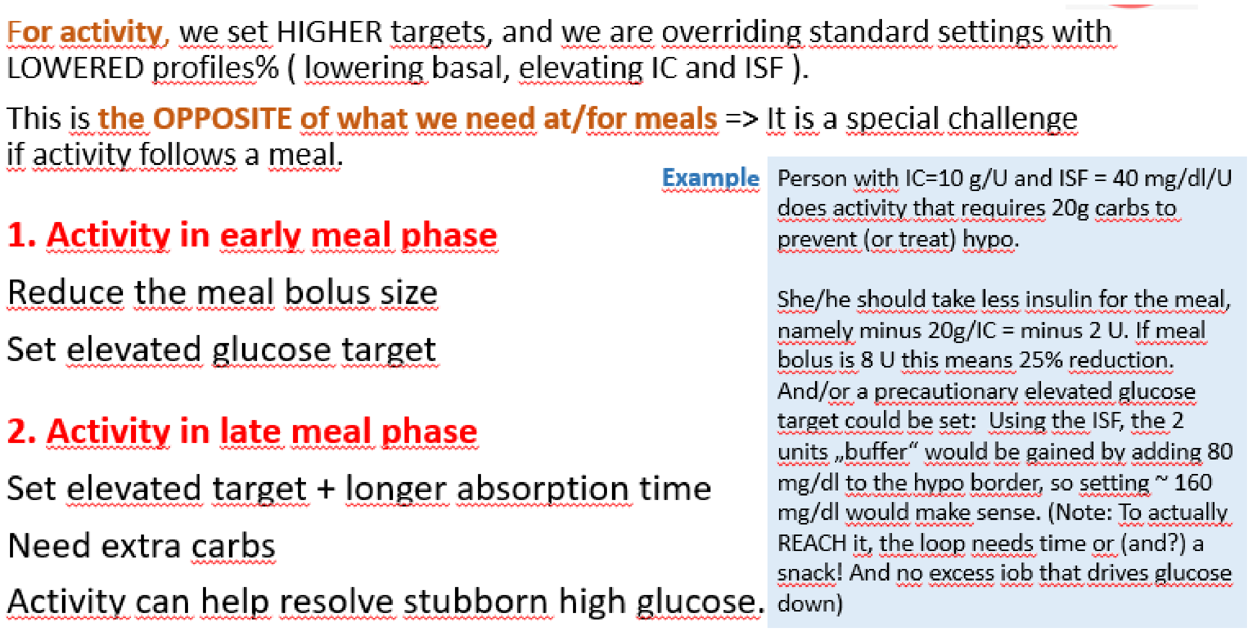
	Interference	Example	Better strategy
	un-announced Afrezza	input would be difficult because of totally different kinetic characteristics	optimizing early phase tuning the loop anti-FPU automation (p.33)
	un-announced extra bolus (user)	e.g. to fight high glucose	patience; tune the loop
	un-announced extra carbs consumed	e.g. to avert hypoglycemia	enter carbs, set high TT
	under-reporting carbs		tune the loop
	over-reporting carbs		tune the loop
	basal rate or factors unrealistic		tune the loop
	erasing past boli to lower iob and trigger more insulin by the loop	to fight high glucose (alternative to unanounced bolus)	tune the loop

Special challenge: Meal followed by exercise

Generally for exercise you will benefit from getting less insulin, and from having an elevated glucose level - with „room“ for it to go down

> Exception: In very stressful (or super exciting) situations, glucose may be rising temporarily and might even briefly require extra insulin

The blue box describes a standard situation with reduced insulin need…



So, **for exercise**, we set a HIGHER target, and we are overriding standard settings with LOWERED profiles% (lowering or completely shutting off basal; elevating IC and ISF) so a desired %age of reduced insulin supply results..

This is **the OPPOSITE of what we need at/for meals** => It is a special challenge if exercise follows a meal.

1. Activity in early meal phase

Make **partial bolus** and set **higher glucose target**

Deduct as much insulin from the normally given bolus, as is needed less due to activity. For instance, if activity would require 20g extra carbs to prevent hypo, use your IC to calculate the units of insulin to reduce, in order to prevent the hypo without taking extra carbs: At IC = 10 g/U, 2 units of insulin less should be given. So for instance 5 U insted of 7 U. Likewise, if your ISF is 40 mg/dl / U, you may want to be about 80 mg/dl above hypo warning and should set a target around 160 mg/dl during activity.

So you need to collect knowledge, for each of your typical activities, how much glucose you (!!) typically need extra. Then use your (!) factors to estimate needed modification of inulin for meal, and for set glucose target.

If it is unclear in the beginning how strenuous the activity will be, your options are to reduce bolus for the meal even stronger, or to risk going low and requiring sports snacks (or sweet drink etc) later.

2. Activity in late meal phase

If activity starts when your loop is mostly zero-temping after having delivered the insulin as likely needed for (most of) the meal, this is unfortunate. You have a risk of going low as is, and the loop cannot do anything for you now. Before starting activity, be sure you do (or can) cover your needed extra energy, for instance the 20g in the example used above.

On a positive note, a bit of activity (like the post-dinner walk) can actually be good to control glucose in case you tend to go high for hours after meals, for instance, to resolve fat-induced high glucose values due to increased resistance as sometimes seen 3-4 hours after a fatty meal (as discussed p.34).

Depending on the amount of caution applied to prevent hypo during activity, a correction bolus may be required when activity comes to an end.

If the activity is not too strenuous, you still may require insulin for late carbs. In that case, use an override (a reduction of %profile), so the loop gives TBRs or microboli at a reduced rate, when glucose gets above a certain BG #. In iOS loop, you can also play with extending the absorption time of carbs.

Overall, it's a personal experiment to find what works good enough for you. Regarding exercise, it very much depends on kind, duration, starting time relative to the meal.

Could mention here: ActivityMonitor for light activity/in-activity modulation of sens

---more info---

There are extra resources (videos, blogs, FB posts) on sports management when looping Add ref. . Certainly the open source systems offer much more adjustability to all kinds of also extreme sports. And there are other groups discussing specifically kids, with their sometimes unpredictable eating and hopping-around habits.

Add references to “ultimate” examples:

- exercise mode x autoISF x FCL How even a full closed loop can be prepared to automatically deal with this challenge, can be seen in (FCL e-book Case study 6.2)
- Dana Lewis ultra-marathon

Link or here?: Activity Monitor description (better integrate at beginning when talking about calibrating for normal situations .. Modulating (manual, auto) for temp sens/resist.