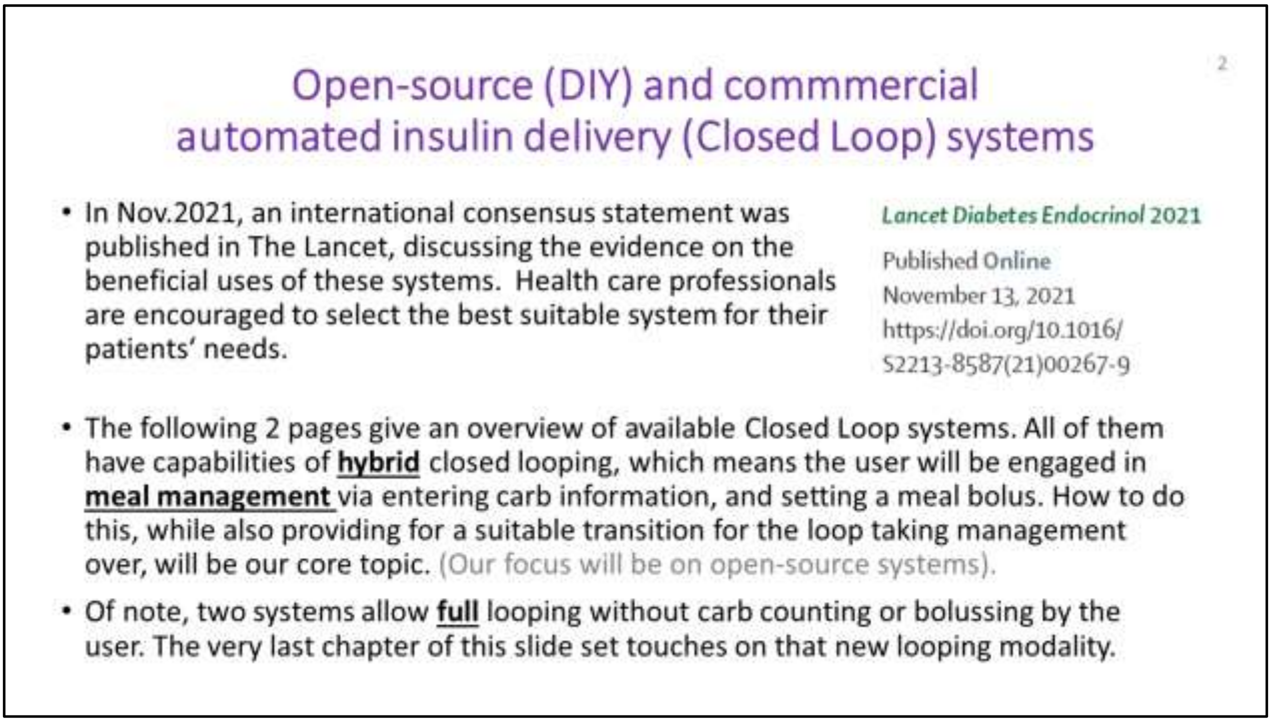
For most loopers, meals are the major problem for keeping glucose in range.
This presentation looks at the principle challenges and available solutions from a lay user perspective. It builds on the published learnings of looping pioneer Dana Lewis, and other related literature. It also reflects own experiences, and insights shared on the internet platforms of the international looper communities.
The author is type1 diabetic for many years, and looper for several years.
He uses AndroidAPS. Both his education in technical chemistry and his nautical hobbies helped him understand aspects of the meal control problem, and, hopefully, also proves helpful for communicating about it.
There are numerous references to what other open source loopers have published. We all cannot guarantee correctness of all content, or applicability to other patients, and assume no liablity. In line with our #WeAreNotWaiting philosophy, we like to encourage critiques and debates so we all can deepen our knowledge in moving forward..
Please use everything that is said only to trigger thoughts for cautious self-responsible conclusions. Seek further clarification in the open-source looper communities, and, most importantly, with your doctor.
Luckily, just very recently, an international consensus was published in one of the globally most renowned medical journals, encouraging our doctors to embrace open-source loops as viable options for their patients, besides the increasing number, but more limited nature, of commercial loops (next slide):



Before we get to our topic, we like to share a nice overview that an international expert group issued in The Lancet, just in time for the 2021 World Diabetes Day. It lists core features for all available looping systems, and discusses considerations for medical professionals how to best make use of these offerings for their patients.
All of them have capabilities of **hybrid** closed looping, which means the user will be engaged in **meal management** via entering carb information, and setting a meal bolus. How to do this, while also providing for a suitable transition for the loop taking management over, will be our core topic. We will focus on open-source systems. Users of commercial systems should refer to the training material for specifics on their systems.
Of note, two systems allow **full** looping without carb counting or bolussing by the user. The very last chapter of this slide set touches on that fairly new looping modality.

	Open-source AID systems			Commercial AID systems				
	OpenAPS	AndroidAPS	Loop and FreeAPS	Medtronic 670G/770G	Medtronic 780G	Tandem Control IQ	Diabeloop DBLG1	Cam APS FX
Type of closed loop	Hybrid to full	Hybrid to full	Hybrid	Hybrid	Hybrid	Hybrid	Hybrid	Hybrid
Type of algorithm	Heuristic*	Heuristic*	MPC	Proportional-integral-derivative and insulin on board	Basal rate modulation: proportional-integral-derivative, insulin on board; corrections: fuzzy logic	Predictive control (ie, Kalman filter with prediction)	MPC	MPC
Licence status	Open source	Open source	Open source	US FDA and CE mark (for people aged ≥7 years)	CE mark	US FDA and CE mark (for people aged ≥6 years)	CE mark (for people aged ≥22 years, total daily dose <90 units per day)	CE mark (for people aged ≥1 years)
Availability	Worldwide	Worldwide	Worldwide	USA, Canada, Australia, some countries in Latin America, Middle East, Europe, South Africa, and Hong Kong	Some countries in Europe	USA, Canada, some countries in Europe	France	UK
Compatible CGM systems	Dexcom G4, G5, or G6; Medtronic Real-Time Revel and Enlite; other CGM systems and CGM-like devices (eg, FreeStyle Libre with MiaoMiao or BluCon) via Nightscout	Dexcom G4, G5, or G6; FreeStyle Libre (via MiaoMiao, BluCon, or Bubble); FreeStyle Libre 2; Eversense; Medtronic Guardian 2 (via 600 series pump); Medtrum A6; PocTech; Gluco24	Dexcom G4 (with share receiver), G5, or G6; Medtronic Enlite; FreeStyle Libre (via Spike)	Medtronic Guardian 3	Medtronic Guardian 3 and future generation sensors	Dexcom G6 and future generation sensors	Dexcom G6 and future generation sensors	Dexcom G5, G6, and future generation sensors

Source: The Lancet (see p.2). Note: FreeAPS-X is another Open-source system in late development stage using the heuristic OpenAPS algorithm (for Apple phones)

	Open-source AID systems			Commercial AID systems				
	OpenAPS	AndroidAPS	Loop and FreeAPS	Medtronic 670G/770G	Medtronic 780G	Tandem Control IQ	Diabeloop DBLG1	Cam APS FX
Compatible insulin pumps	Medtronic 512/712,† 515/715,† 522/722,† 523/723,‡ 554/754§	AccuChek Spirit Combo; AccuChek Insight; Dana R or RS; Medtronic 512/712†, 515/715†, 522/722†, 523/723‡, 554/754§; OmniPod Eros	OmniPod Eros; Medtronic 515/715†, 522/722†, 523/723‡, 554/754§	Medtronic 670G	Medtronic 780G	t:slim X2	Kaleido, AccuChek Insight	Dana RS, Dana-i
Compatible mobile phones	Optional (ie, Android and Apple)	Android	Apple	None for 670G; Apple and Android for 770G (view only, insulin pump cannot be remotely controlled)	Apple and Android (view only, insulin pump cannot be remotely controlled)	None	None	Android (ie, all models compatible with Dexcom G6 app)
Compatible smartwatches	Any	Wear OS by Google	Apple Watch	None	None	None	None	None
Connectivity	900 mHZ (between rig and pump)	BLE*, Bluetooth, RF2BT bridge, NFC2BLE bridge (depending on pump and CGM system)	BLE (between mobile phone and RileyLink), 916 MHz (between RileyLink and Medtronic pump), 433 MHz (between RileyLink and OmniPod)	2·4 GHz	BLE	BLE	BLE, mobile internet connection to cloud via virtual private network	BLE
Required additional hardware for use	Rig (eg, Raspberry Pi or Intel Edison, Explorer Board)	RileyLink (only for Medtronic pumps and Omnipod), otherwise none	RileyLink, EmaLink, OrangeLink, or similar	None	None	None	Dedicated handheld device	None
Required for setup	Any computer	Any computer, Android Studio software	Mac or virtual machine, XCode software, Apple developer licence	NA	NA	Any computer (only for updates)	NA	Compatible Android smartphone

BLE=Bluetooth Low Energy. CGM=continuous glucose monitoring. FDA=Food and Drug Administration. MPC=model predictive control. NA=not applicable. *The heuristic algorithm makes multiple predictions and delivery is altered to ensure that the lowest of these predictions falls in the target range for the individual. †Compatible with all firmware. ‡Compatible with firmware 2·4A or lower. §Worldwide Veo is compatible with firmware 2·6A or lower and Canadian or Australian Veo is compatible with firmware 2·7A or lower.

Source: The Lancet (see p.2).

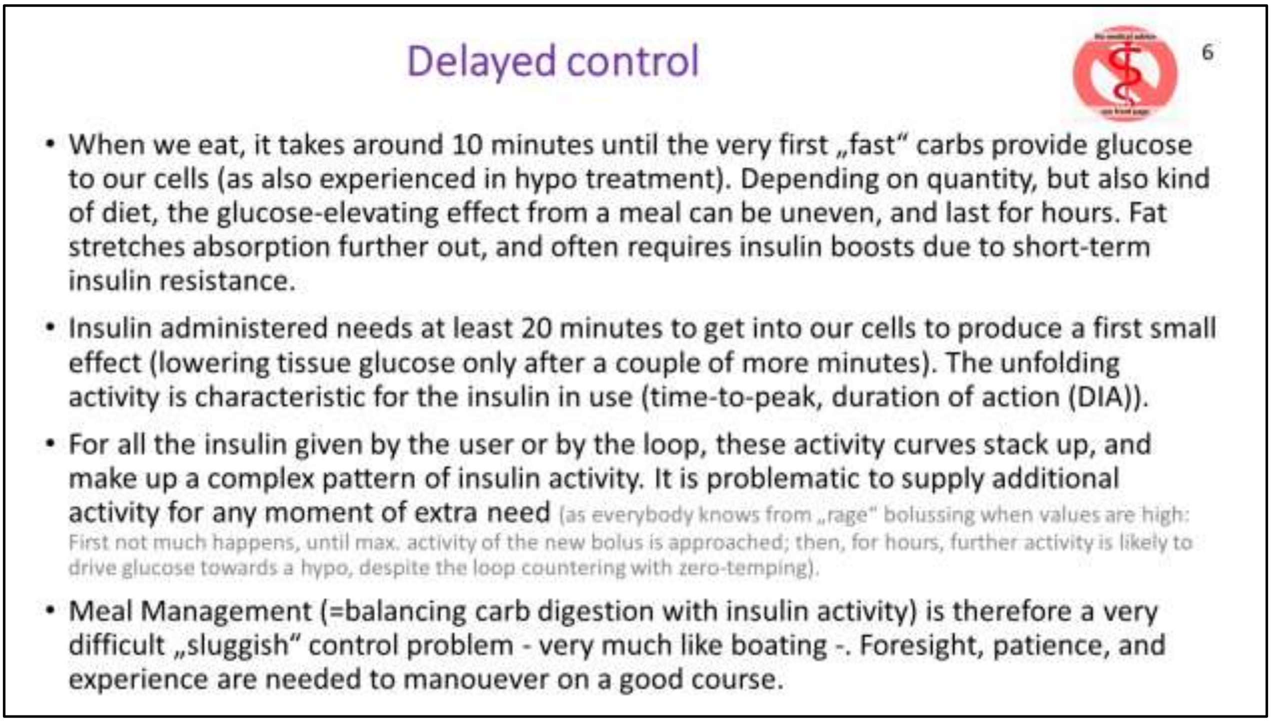
Topics

This presentation on Meal Management describes the general challenges that loopers and (any) loops face when managing carb absorption (meals).

We look at our topic, meal management in closed loop, in four main chapters.
First, the principal challenge when approaching meal time is laid out. We look into the time-patterns of insulin activity and of carb absorption, to gain some understanding of „sluggish“ delayed control. The key parameters loops use to navigate our blood sugar through the disturbances created by meals are presented. For the further discussion, a 3 phase model of meal management is laid out.
The **second** part will be about how to reasonably interact with the looping algorithm in the **phase before meals;** and about the **early meal phase**, where, in <u>hybrid</u> closed looping, our meal bolus is highly active and forces the loop to retreat into a co-management role.
The **third** part discusses the **late phase** of a meal. This is when the bolus initiated by the user fades out, while more late carbs are being absorbed. Effects from fat and protein complicate the picture. Now the looping algorithm must increasingly take over. We focus here on the two main algorithms as used in the open-source systems. To enable the algorithm to take best care, the user must give appropriate inputs about the meal, and select appropriate settings in her/his looping system. Also, tuning some parameters may be required from time to time. We know it is sometimes tempting to interfere with the loop, so we will touch on the pros and cons of doing this, as well.
The **last chapter** describes **simplified meal management** when working with the **„oref(1)“** algorithm. It principally can work without carb inputs for meals. This

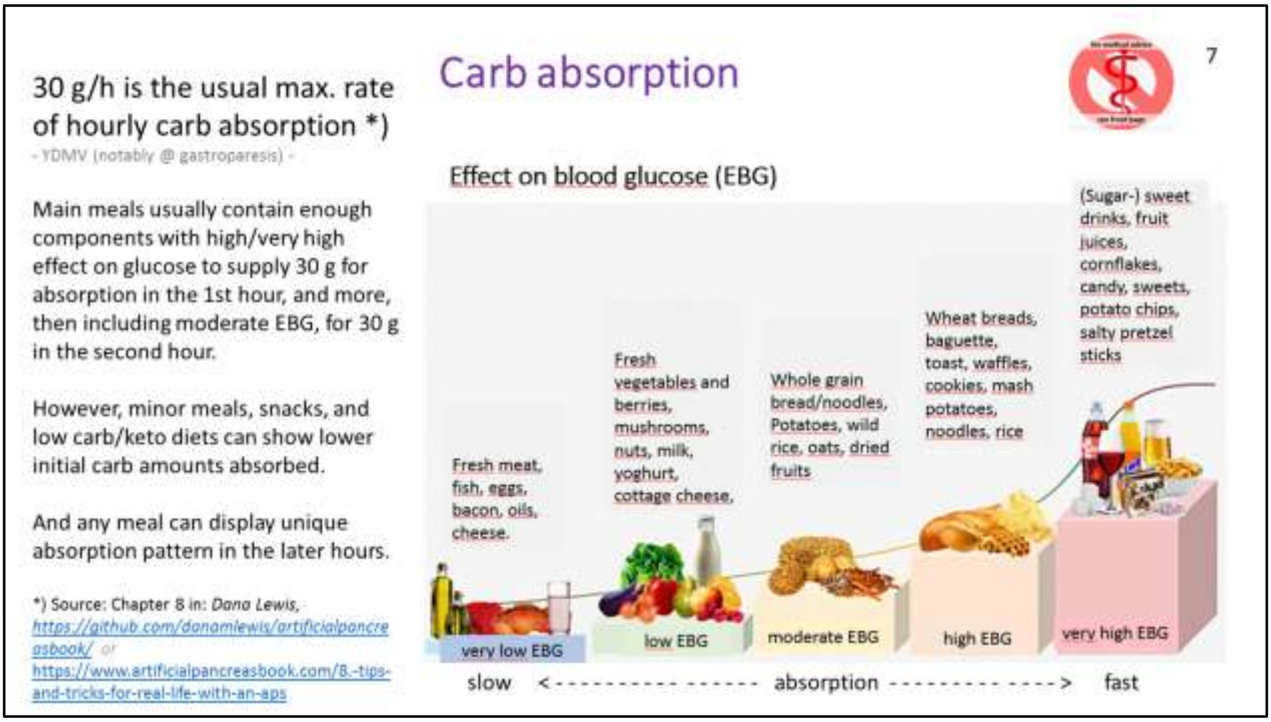
enables a very much simplified „carb counting" in hybrid closed looping. Moreover, full looping is a possibility if certain criteria are met, which will be sketched in the very last section.

This slide set builds on a more extensive 3-parts presentation on Meal Management which focusses on AndroidAPS and is avaliable in /Files of Facebook Group „AndroidAPS Users" (and in German language in de.loopercommunity.org).
Part III. of it is an advanced user guide for full closed looping without any carb inputs or user boli. (Feasibility documented for AndroidAPS or other oref(1) system, Lyumjev or Fiasp, Dexcom CGM, and basal rates of ~0.5U/h or higher)(See p. 44 ff for some more info).

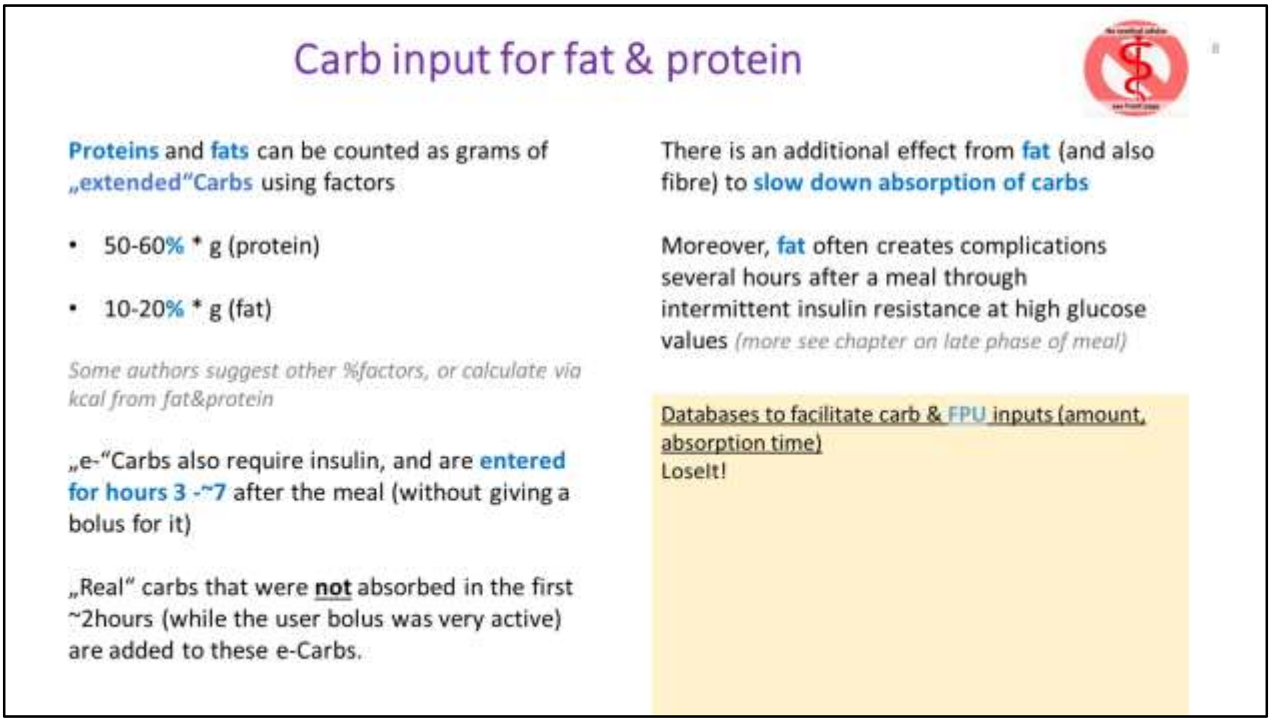


One of the core challenges in type 1 diabetes management is to provide our body the suitable amount of insulin for any meal. While „historically“ the focus was on roughly providing a balance without incurring severe hypos, or prolonged hyperglycemias, the advent of continuous glucose monitors was, literally, an eye opener to most of us:
It allows us, every 5 minutes, to have a look how carb absorption and insulin activity work out to result in the glucose curve we see developing:
When we eat, it takes around 10 minutes until the very first „fast“ carbs provide glucose to our cells (as also experienced in hypo treatment). Depending on quantity, but also kind of diet, the glucose-elevating effect from a meal can be uneven, and last for hours. Fat stretches absorption further out, and often requires insulin boosts due to short-term insulin resistance.
Insulin administered needs at least 20 minutes to get into our cells to produce a first small effect (lowering tissue glucose only after a couple of more minutes). The unfolding activity is characteristic for the insulin in use (time-to-peak, duration of action (DIA)).
For all the insulin given by the user or by the loop, these activity curves stack up, and make up a complex pattern of insulin activity. It is problematic to supply additional activity for any moment of extra need (as everybody knows from „rage“ bolussing when values are high: First not much happens, until max. activity of the new bolus is approached; then, for hours, further activity is likely to drive glucose towards a hypo, despite the loop countering with zero-temping).

Constantly balancing carb digestion with insulin activity is the task. It is really a mission-impossible, given the sluggish nature of anything we already did (eat, past insuilin additions) or could do (further insulin additions, withdrawing basal, adding more carbs).
Meal Management therefore is a very difficult control problem - very much like boating -. Foresight, patience, and experience are needed to manouever on a good-enough course. And a good algorithm will have some of this modelled-in, or could also have some self-learning capabilities (like autotune, autosense in open-source systems, or non-disclosed machine learning in some of the newer commercial closed loops).



To picture how carb absorption and need for active insulin, develops over time after eating, we first must look at the type of foods our meal consisted of. By glycemic index, i.e. by how fast they impact blood glucose, we can for instance group them into the 5 categories as shown here.
The fastest absorbing foods have a very high (fast) effect on blood glucose. So this is what we need to prevent or resolve a hypoglycemia. But as component of a meal this makes glucose control very dfifficult (possible really only with a suitable pre-bolussing time).
However, even the slowest absorbing foods can be challenging to deal with. Typically, they are absorbed too late for a meal bolus to take effective care. Notably fats can also interfere with absorption speed and insulin requirements for carbohydrates.
Also, there is a maximum hourly capacity: 30 g is the usual maximum hourly carb absorption (YDMV, notably with gastroparesis).
Main meals often contain enough components with high or very high effect on glucose, to supply 30 g for absorption in the 1st hour, and more, then including moderate EBG, for 30 g in the second hour.
So, for the fast insulins (Fiasp or Lyumjev) we usually use for looping, our initial bolus usually should cover 60g.
However, low carb/keto diets can show lower initial carb amounts absorbed (as is the case for minor meals or snacks).
Any meal can display unique absorption pattern in the later hours



Proteins and fats can be counted as grams of **„extended"Carbs** using factors 50-60% per gram of protein, and 10-20% per gram of fat.
(There is an additional effect from fat (and also from fibre) to slow down absorption of carbs, so it may seem that fats translate to a higher % into late carbs.)
Instead of using these % conversions, there is another calculation via kcal from fat and protein (see FPU; p.29-30).
„e-"Carbs also require insulin, and are entered for hours 3 -~7 after the meal (without giving a bolus for it)
„Real" carbs that were not absorbed while the user bolus was very active are added to these e-Carbs.
Often, fats create complications several hours after a meal through intermittent insulin resistance at high glucose values. *(More see chapter on late phase of meal, p.27 ff).*

Discussion regarding factor for fats: Alternatives are (A) Use a low factor of 10 – max 20% for fats; plus apply stronger ISF when glucose is stuck on high values. - Or (B), Apply a bigger % for the conversion that exaggerates carbs from fats, and this way also gets them covered with more insulin.
For the AndroidAPS loop, which does not react directly on grams of „e"-carbs entered, the mode with temporary elevated ISF is more effective.

"Insulin on bord" (iob)

9

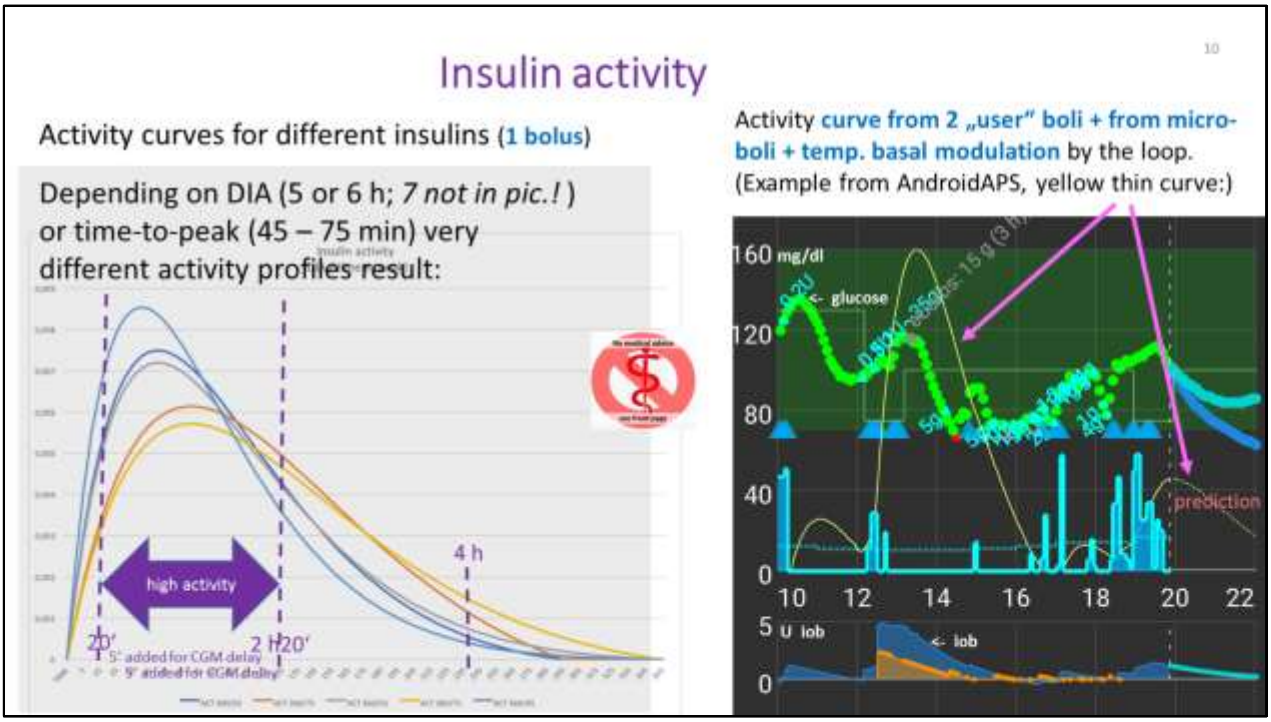
Insulin on bord shows how much insulin is available for activity in the upcoming hours. Ideally (like in AndroidAPS) it is defined as insulin *above the basal needs* as defined in profile.

- Negative iob then indicates that less than basal need is on bord. This is usually not the case around meal-times.
- Posititive iob indicates there is insulin available to cover more expected carb absorption, or to correct high glucose values. Therefore, the iob should „match" the needs for unabsorbed carbs (cob/IC), plus for corrections (delta-glu/ISF).
- However, the iob information **cannot tell when/how strong this insulin actually provides activity.** Therefore an „overall match for the next couple of hours" of iob with cob might not be good enough. Fortunately, the loop records all insulin deliveries, and calculates the aggregate activity curve. This enables predictions for every 5-minute segment in the future. PLUS the loop can calculate, how insulin above (or also below) basal need it gives (withholds), works out „on top" of that **activity pattern**, increasing (or decreasing) available „power" every 5 minutes in the future.

Insulin on bord shows how much insulin is available for activity <u>in the upcoming hours</u>.
Ideally (like in AndroidAPS) iob is defined as insulin *above the basal needs* as defined in profile. That way, it is always clear how much is there for correction-to-target, or for more carbs becoming absorbed.

- Negative iob then indicates that less than basal need is on bord. This will not occur at meal time.
- Posititive iob indicates there is insulin available to cover more expected carb absorption, or to correct high glucose values. Therefore, the iob should „match" the insulin need for unabsorbed carbs (which can be calculated as =cob/IC factor), plus for corrections (which can be calculated via =delta-glucose/ISF factor).

However, the iob information **cannot tell when/how strong this insulin actually provides activity.** Therefore an „overall match for the next couple of hours" of iob with cob might not be good enough.
Fortunately, the loop records all insulin deliveries, and calculates the aggregate activity curve. This enables predictions for every 5-minute segment in the future. PLUS the loop can calculate, how insulin above (or also below) basal need it gives (withholds), works out „on top" of that **activity pattern**, increasing (or decreasing) available „power" every 5 minutes in the future. (It immediately adds fully to iob, but that is only the trivial part of the story).



The **graph on the left** shows for various types of Insulin the time of peak activity, and how it decreases after the peak depending on the set DIA (duration of insulin action; unfortunately, the chart does not include the important 7 h DIA curves). We can see (dotted lines) that active insulin amout can differ by factor of 2 after 15 minutes, or up to factor 3 at 4 hours. So it is very important to give the loop the correct data on the insulin in use.

(Beginner loopers often select the DIA too short: One consequence is, that several hours after dinner, their loop is falsely „told" the bolus is already very weak, practically gone, and will give more correction insulin than would be the case with a longer set DIA. So, it is not necessarily a too strong nightly ISF that can lead to low glucose values!)

The rapid insulins we prefer for looping (blue and gray curves) show a time window with high activity between 20 minutes and 2 hours and 20 minutes after the bolus was given. *(Slower insulins would show a longer albeit on average weaker period of high activity, with onset shifted by ~15 minutes, too).*

For the meal bolus in hybrid closed loop, it is very important to picture how many grams of carbs will become absorbed in this time window.

With realistic inputs for time-to-peak and DIA, the loop can exactly calculate, for past AND for future hours, how much insulin was or will be active in each 5 minute segment. This is very powerful for the loop's decision whether and how much insulin activity to add - or to withdraw via zero-temping - going forward.

The **graph on the right** (main screen AndroidAPS example) shows how the aggregate insulin activity (thin yellow curve) developed between 10 and 20 h, plus how it will develop in the next 2 hours. The prediction assumes, no extra insulin above basal need is given from 20:00h on.

After a big bolus was given at 12:30h, we see in the smaller bottom graph an iob maximum of about 5 units. But – it is not active much, the green curve in the top graph, the glucose, keeps rising!

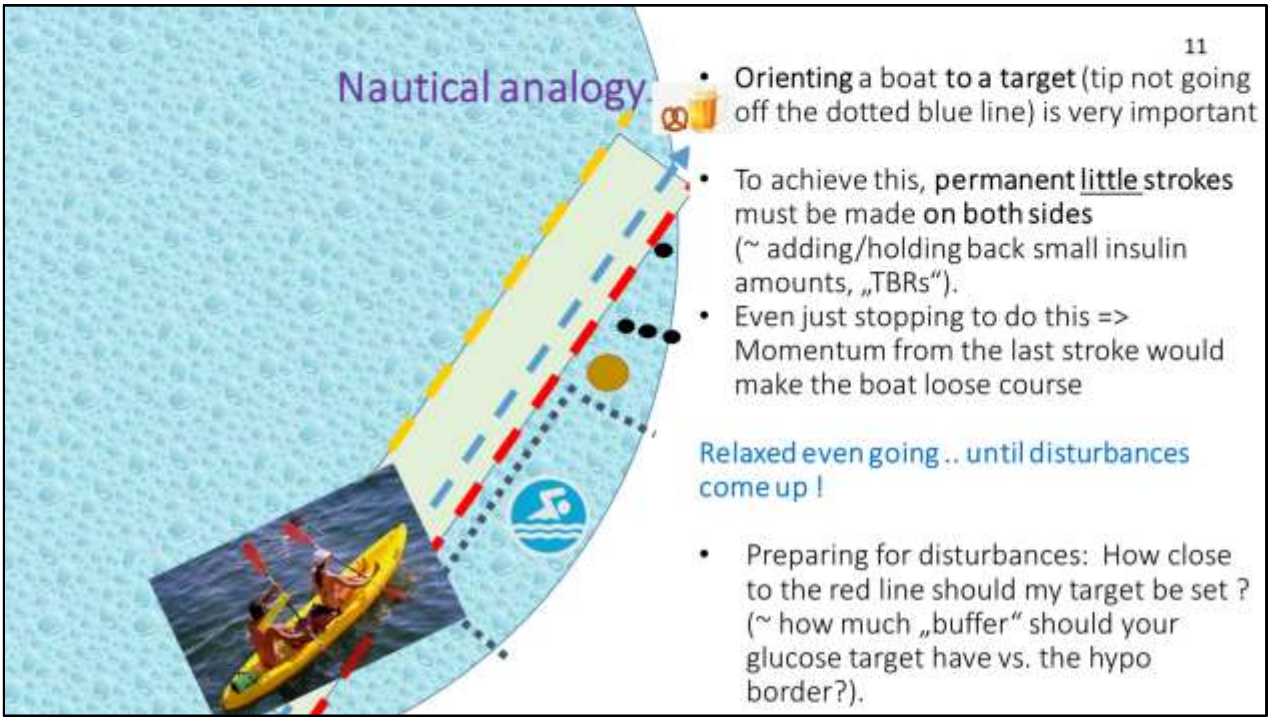
It is the (thin yellow) insulin activity curve that tells us better where glucose will be headed.

But we must additionally look at carb absorption…

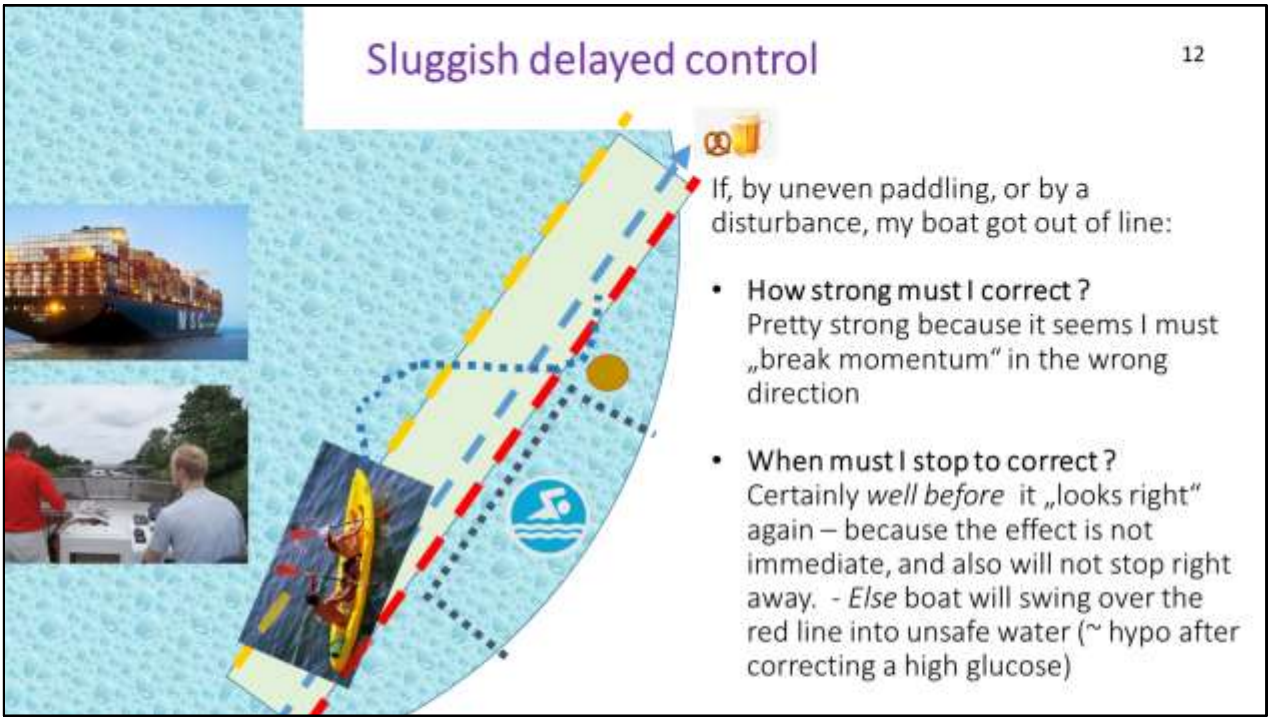
----

http://seemycgm.com/2017/10/21/exponential-insulin-curves-fiasp/

http://seemycgm.com/2017/08/09/why-dia-matters/



On a paddling tour to the beer garden we could learn a lot about glucose control and looping.
When paddling, it is very important to always orient the tip of the boat towards the target, and, to achieve this, always slight paddling is needed, alternating on the right and the left. (~ In our case, the loop does this fine-tuning by elevating and reducing the temporary basal rates (TBR) all the time).
We are free to select our target. (~ Many loopers select a very low goal. Beginners sometimes think to „lure" the loop in the desired direction with this measure. However, it is generally a good idea to have some room for disturbances, to avoid hectic moves and alarm fatique. Also experienced loopers could be more relaxed, and utilze more options,) if not „always" near the red line). (…as we shall see later, p.18).
Also, everybody should be aware of the performance of the CGM system (eventually depending on body site, how long in use etc.).

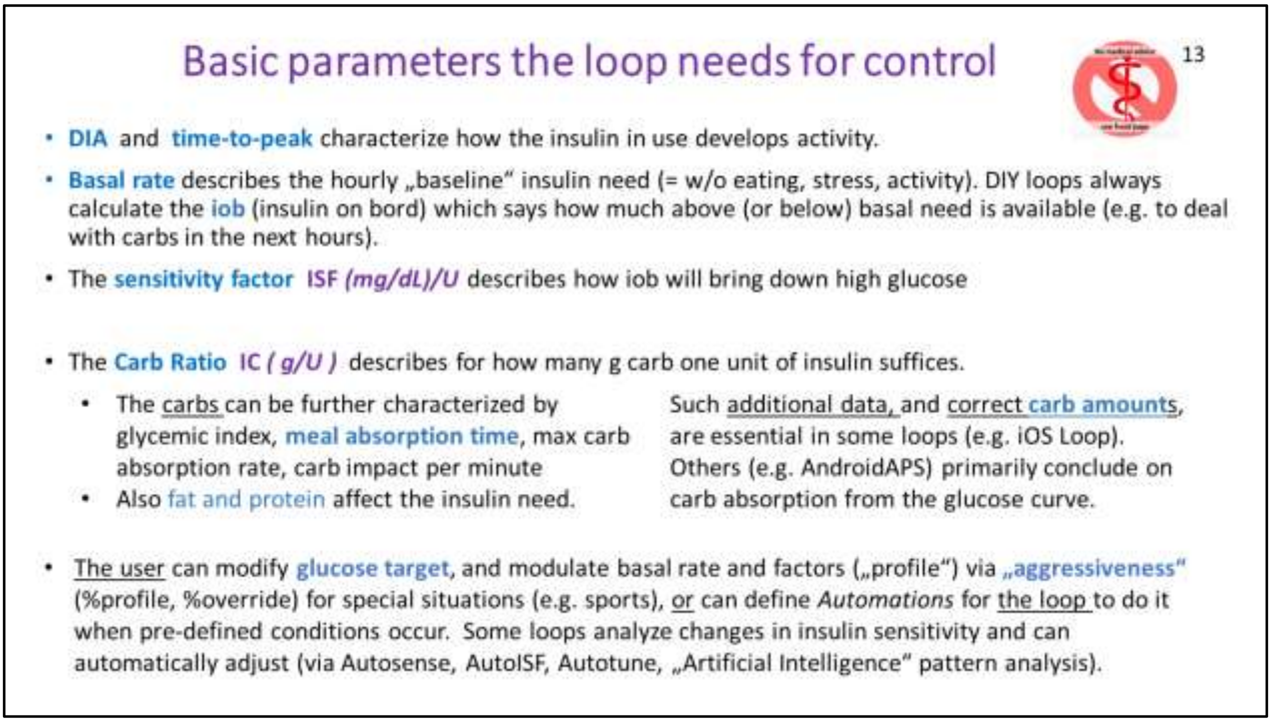


Our key problem is, that anything we do, just starts to become effective in a 15 to 30 minutes time frame, and smears in its effects over a couple of hours..
Seasoned navigators learned over years how to deal with sluggish control of their vessel.
However, not knowing enough about it, makes many manoevers dangerous.
In type1 diabetes we face very similar challenges. Neither the insulin nor the food (glucose) we take will immediately be active in our body. And their effects will be very uneven, but last for hours.
We face a threefold challenge:

1) We must tell our loop certain basic parameters that it needs to keep us in a certain range
2) We must watch and analyze our loop occasionally, in order to learn how to tune certain parameters for better performance in the next days
3) We must become experienced „sugar-captains"

NOTE: This is different to „sugar surfing". Surfing „was" a strategy developed, when CGMs became popular, and people tried to „tweak" their glucose curve in a desired direction by giving boli. Who learned to factor-in a bit about the time delays, eventually could make some progress over where she/he was coming from. But this is far from what we advocate here: (a) To give the loop ever improving parameters to stay better in range („tuning") (b) To give the loop decisive information for its job (like announcing a meal, the absorption time for some components of it, etc.) (c) To

know also about LIMITATIONS of our system (d) To be able to recognize MISFUNCTION of the system, and have a plan B for that scenario.
Generally, it is important to remain calm, analytic and open to learn, and in good spirit, to make it better next time.
To remain in the target range is the goal. How to achieve this more and more, and also around challenging meals, is our core topic.



We already saw that the kinetic characteristics (time-to-peak and DIA) of the insulin in use are very important for the loop.
We tell our loop our basal rate to cover our hourly „baseline“ insulin need w/o eating, stress, or activity. DIY loops always calculate the iob (insulin on bord) which says how much above (or below) basal need is available (e.g. to deal with carbs in the next hours). **)
The sensitivity factor ISF, expressed in *(mg/dL)/U or in (mmol/dl)/U,* describes by how much a unit of insulin (from the available iob) will bring down high glucose. This describes our sensitivity to insulin, and usually differs between times-of-day.
The Carb Ratio IC *(g/U)* describes for how many grams of carb one unit of insulin suffices. In iOS Loop, the absorption times of carbohydrates are also a very important input for each meal. AndroidAPS primarily concludes on carb absorption from the glucose curve
The user can use his knowledge about upcoming meals or activity to help the loop regulate. Modifying the glucose target creates counter-momentum for expected disturbances.
Modulated basal rate and factors („%profile“-changes, „overrides“) alter the „aggressiveness“ of the loop in special situations (e.g. sports),
Moreover, open-source systems allow the user to define *Automations* for the loop to act differently when pre-defined conditions occur. Automations are a fully integrated feature in AndroidAPS (programmable on the smartphone within seconds), and an

available add-on option in iOS Loop (via extra software like IFTTT or Automate!). Some loops analyze changes in insulin sensitivity, and can automatically adjust (Autosense, AutoISF, Autotune, machine learning).

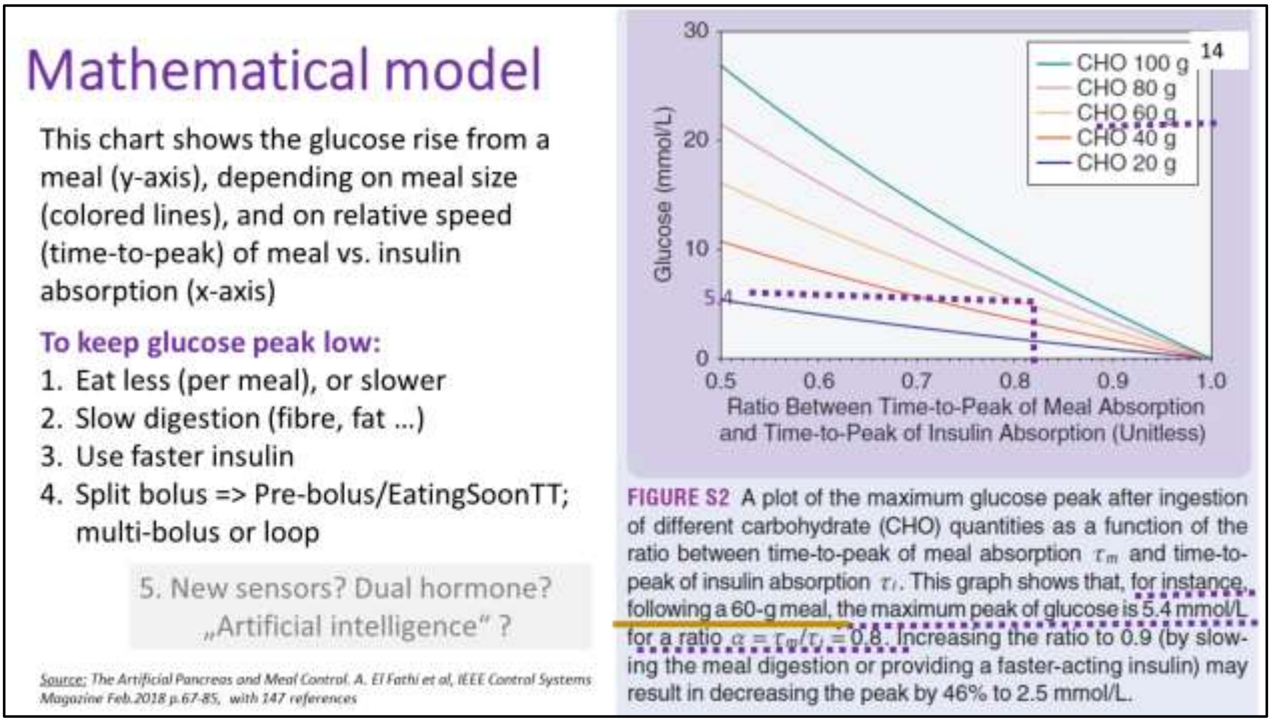
**) Brief reference to the flat basal / circadian basal debate:

Pro flat: It works OK to go from a flat (in case of doubt, low) basal, and „invest" care into validating different IC and ISF factors for several times during the the day and night. *(A contoured basal, but single ISF and IC, is likely worse - and this is what Autotune leads some „trusting" loopers to do!).*

Pro circadian: Factors AND basal run in typical „biorhythm" patterns***). The loop has best-possible data if it exactly knows which part of insulin is basal requirement each hour, and hence what the „true" iob is that is available for managing meals and for bg corrections. Also, if loop does not work (system failure), the set profile basal is truely what keeps you stable.

Con making a big deal out of this question: As insulin sensitivity is affected by many factors (stress, sports, infection … see also p.59), a lot of more things (like also: does your loop have an Autosense feature etc) is playing into this debate, and everybody must really decide how much she/he is willing to invest, and which result is „good enough". After all, the loop looks, calculates, every 5 minutes, and CAN correct for things that are a bit „off".

***) 24 hour circadian patterns change strongly during childhood – see e.g. here: https://www.facebook.com/groups/AndroidAPSUsers/permalink/3091340191087377



Modelling the course of carb absorption and of insulin activity, the effects on glucose peak heights can be quantified. The graph shows on the y axis peak over baseline (in mmol/dl), and on the x-axis the relative speed of insulin absorption to carb absorption. (Carb absorption is faster, therefore all values are under 1.0).

The model allows to calculate how post-prandial glucose peaks can be reduced via:

1. Eating less (per meal; see colored curves), or slower
2. Slower digestion (fibre, fat …). (This would move the RATIO of insulin/carb absorption peak on the x-axis to the right)
3. Using faster insulin (Effect as in 2.)
4. Splitting bolus => Pre-bolus/EatingSoonTT; multi-bolus or loop.

New developments like dual hormone systems, or faster responding sensors, might also have potential to lower glucose peaks.

Source: The Artificial Pancreas and Meal Control. A. El Fathi et al, IEEE Control Systems Magazine Feb.2018 p.67-85. This is a very thorough mathematical modelling study, with 147 references to other literature. However, any modelling must make simplifying assumptions. So, our everyday reality is more complex, but the authors do carve out, even in quantitative terms, some core mechanisms at play. We revisit the model on p.45.

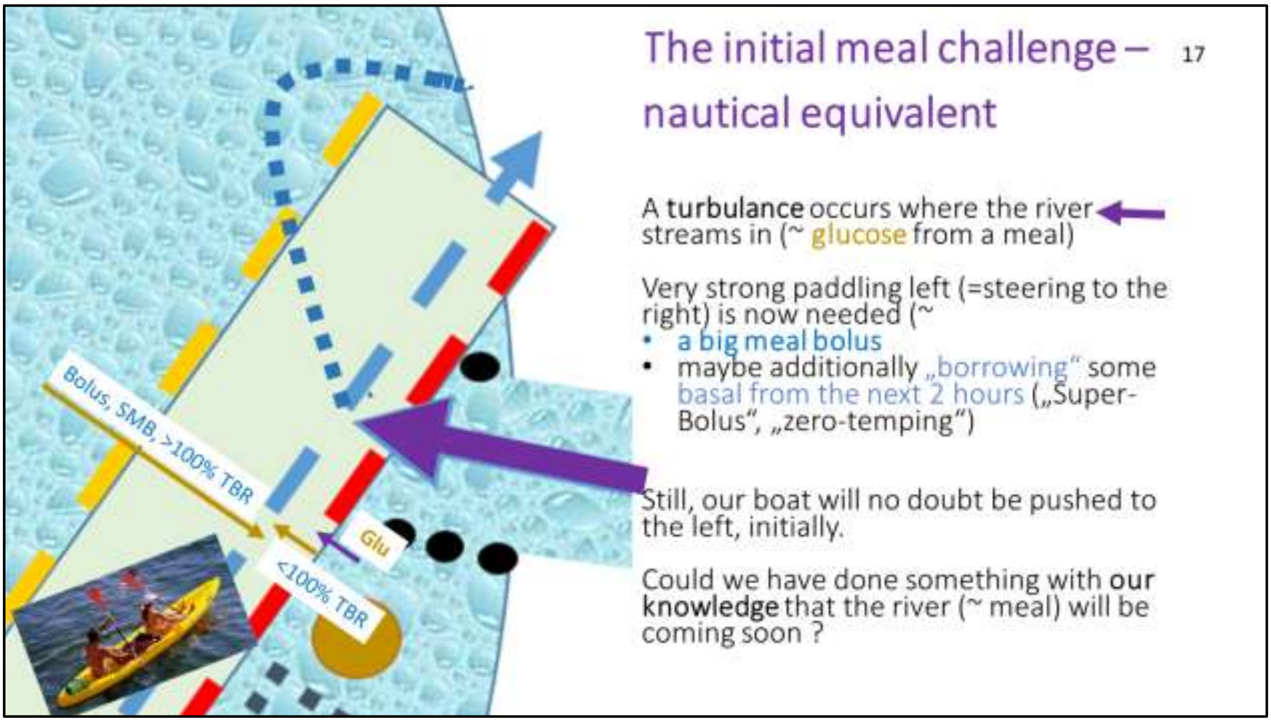
As our table of contents already indicated, we see three phases to each meal that require special attention.
Often we face an unbalanced situation right after meal start, with strongly rising glucose levels. Or, when pre-bolussing, also the opposite could happen, a tendency towards hypoglycemia right around meal start. It is therefore suggested to seek appropriate action already in a **pre-phase**, with the goal to provide active insulin (and a low starting glucose) when first carbs will be absorbed ~ 10-15 minutes after meal start
This marks the beginning of the **early** meal **phase**, in which a constant stream of carbs is absorbed, while the meal bolus develops, and sustains, high activity. In the following main section on the early phase, we look into the options how and when to set the meal bolus.
In the **late phase** of meals, activity from meal bolus fades out. The total carb amount exceeding ~60g, and fat/protein units (FPU) determine the carb absorption in this last phase. In a later chapter we will look how the utilized looping algorithm, and the settings, deliver suitable insulin then.

AndroidAPS: COB calculation — https://androidaps.readthedocs.io/en/latest/Usage/COB-calculation.html?highlight=cob%20calculation#cob-calculation
https://androidaps.readthedocs.io/en/latest/CROWDIN/de/Usage/COB-

calculation.html#erkennung-fehlerhafter-cob-werte
Dana Lewis: https://de.loopercommunity.org/uploads/short-url/grgfNjbECtyau5nwtQH8xhaMRyI.pdf

Topics

In the second main chapter we look into the main meal phase, where in hybrid closed loop, the user issues a bolus.
Very much like in conventional diabetes management, the bolus size is calculated from the carbs, by dividing through a user-specific (and time-of-day-specific) carb ratio (IC).
The kinetics of carb absorption vs insulin activity must be considered for timing the bolus right.
This does really not have much to do with looping. So it is a brief chapter, and we will conclude it with some considerations on the accompanying, and during this phase increasing, co-management role the closed loop assumes.



The start of a meal, with carbs getting absorbed, can be compared to a river streaming in sideways on our paddling tour on the lake. We will experience a strong drift towards the yellow border of our range (~glucose sharply rising). We must try to counter with strong paddling strokes.

So we give a big insulin bolus; both AndroidAPS and iOS Loop also allow additional 2 hours worth of basal to be pulled into our bolus – the so-called Superbolus - for maximum boost and counter-action „against" the carbs.

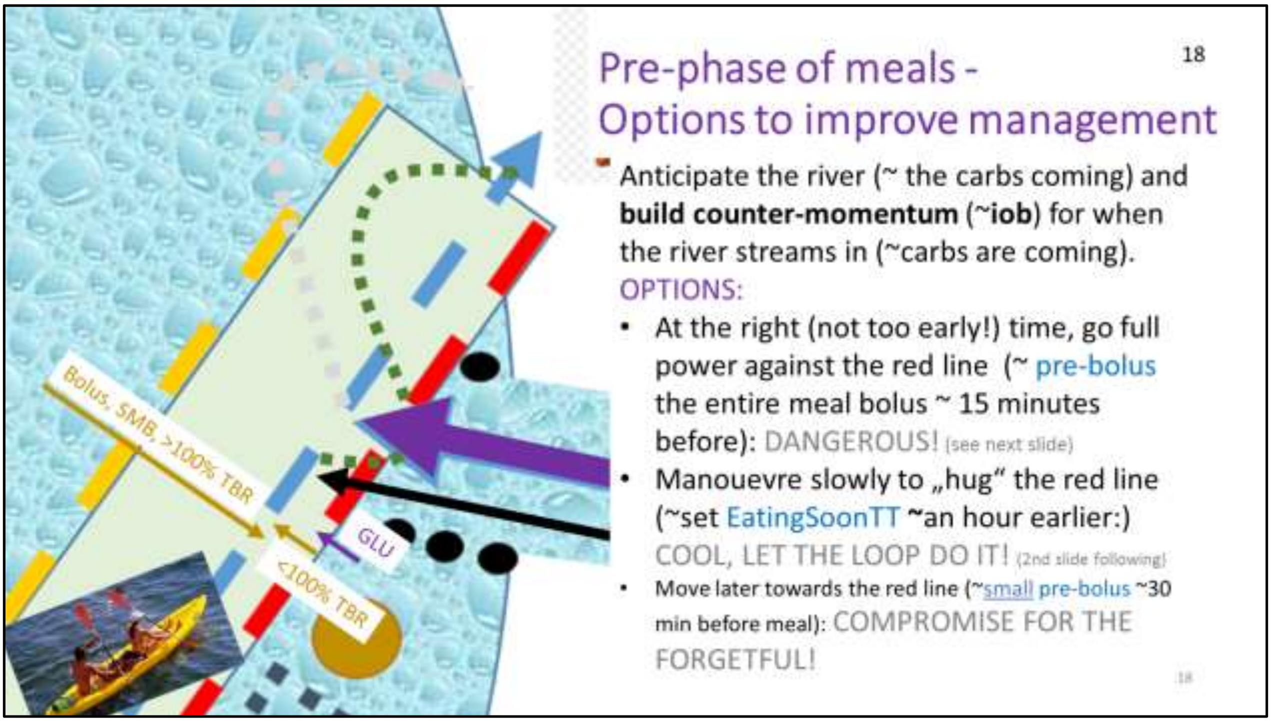
Still, our boat will no doubt be pushed to the left, and eventually also surpass the yellow line

Could we have done something to prevent this, using **our knowledge,** that the river (~ meal) will be coming soon ?

Yes. One also by non-loopers widely used method is to pre-bolus, meaning to give the meal bulus already 10 – 20 minutes before the meal starts.

This can work pretty well, also when hybrid closed looping, if a precisely set pre-bolus time is strictly adhered to. Notably when using one of the newer fastest insulins, or also when you have no control over when the meal will actually be served (restaurants!), pre-bolussing can be dangerous!

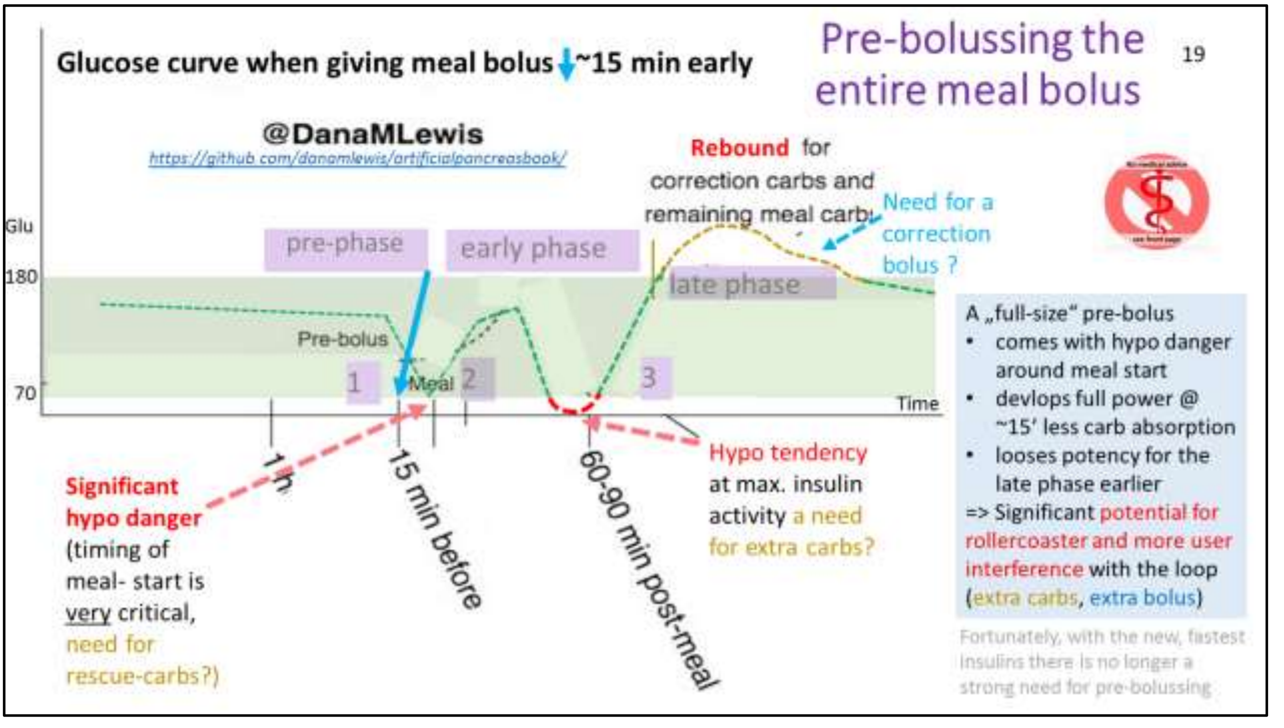
When we loop, there is a much safer option…



…Instead of, briefly before the stream strikes, going full speed against the red border (~pre-bolussing) we could also do the following:
Manouevre slowly, to „hug" the red line (~set EatingSoonTT **~**an hour earlier:). This gives us three benefits:

1. It creates additional room to drift to the left when the current from the incoming river strikes (~ a lower starting glucose value)
2. It provides some momentum against the current (~ some positive iob for the first carbs getting absorbed)
3. The manoevre can be automated (~ COOL, LET THE LOOP TAKE FULL CARE by setting an EatingSoonTT; more see 2nd slide following)

If you forget to set an EatingSoonTT in time (about 1 hour before meal), a compromise solution is, to give a small part of the meal bolus before the meal starts.



This slide shows the strategy to give the (entire) meal bolus about 15 or 20 minutes before meal start. (The number of minutes depends on the insulin in use, and on the actual glucose level).

Only a small glucose rise is experienced in the first minutes after meal start.

At max activity of the early meal bolus, only a certain fraction of the meal carbs will have become absorbed,. Therefore there is a tendency for low glucose values. Also without receiving extra carbs at this point, the glucose curve is likely to rise from there into the late phase.

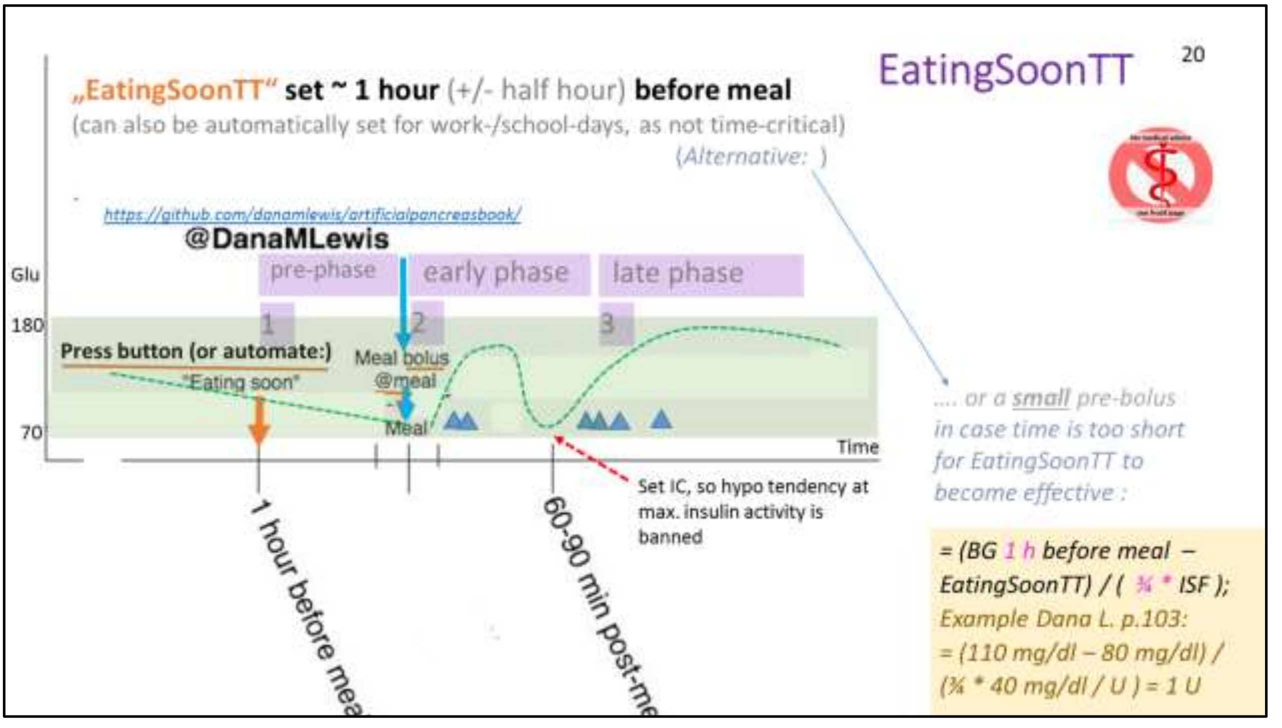
However, a 20 minutes early given bolus also fades out 20 minutes earlier; for a major or FPU-rich meal, this complicates the meal management for the loop in the late phase.

There is a significant potential for a roller coaster, unless all inputs and the timing are precisely executed. (In closed loop, the loop will attempt, via zero-temping, or via adding insulin, to limit the roller-coaster movements, but its capabilities can be limited when challenged by a „stupid" big bolus given by the user).

Fortunately, with the new, fastest insulins, there is no longer a strong need for pre-bolussing

----

Source: Dana Lewis: https://github.com/danamlewis/artificialpancreasbook/



So, here is the alternative that looping pioneer Dana Lewis proposes, based on her findings that:

- In the pre-phase it is important to build up a bit of active insulin for the time of meal start. (Note, that giving even an impressive amount of iob via a bolus, does not get you any active insulin in the next 10 or even more minutes!)
- Carb absorptions is typically (limited at) 30 g/h
- Absorption runs at a relatively even speed
- The glycemic index of meal components plays a minor role (except in iOS Loop for absorption times, for the late phase)

In the **pre-phase** (1), about one hor before meal start, a low „EatingSoon" temporary target is set. This makes the loop drive glucose carefully towards that low temporary target, thereby creating iob followed by active insulin for when the first carbs get into absorption.

A similar effect can be created via giving a small bolus at least 30 minutes before meal start.

Early phase (2)*:* At meal start, a bolus is given for the **60g** of carbs. That is the amount that can be absorbed in the first 2 – 2 1/2 hours*. (YDMV, gastroparesis?)*

Variants for determining the bolus for the early phase:

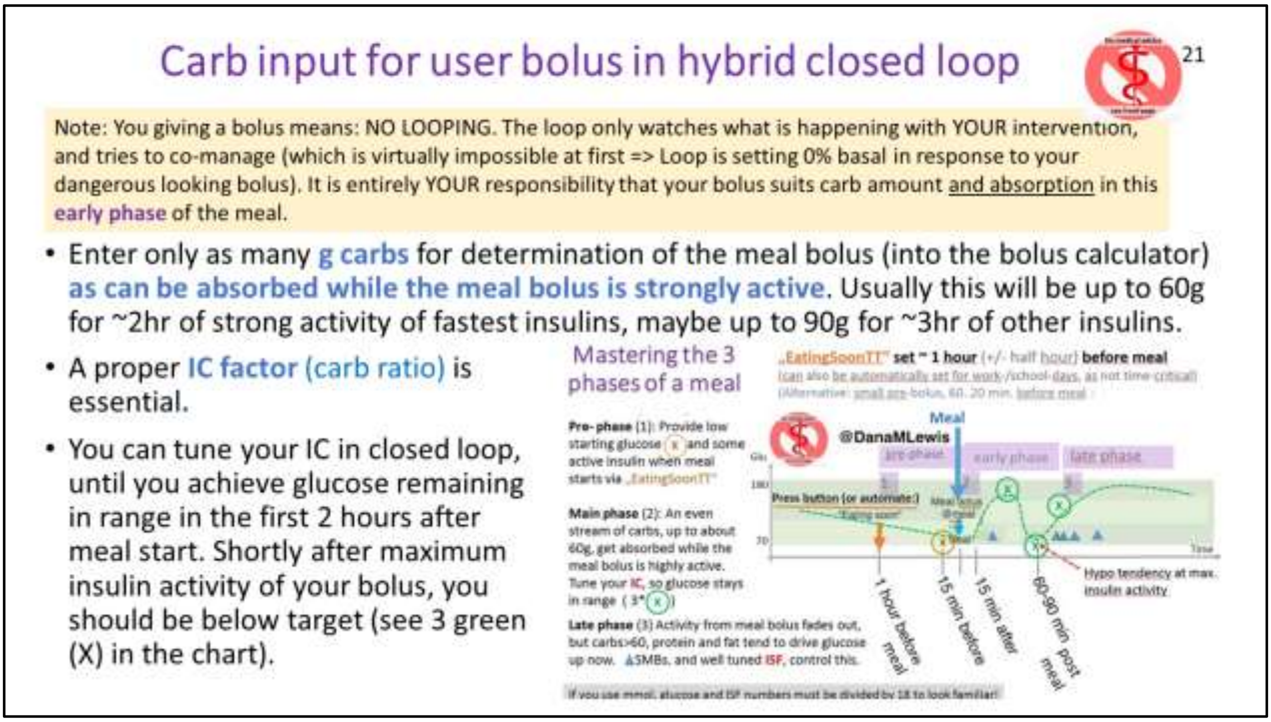
I. Enter in bolus calculator the grams of carbs that can be absorbed while the intended bolus is highly active, usually no more than 60g and bolus for it. Input the extended carbs/FPUs, icluding info about absorption time window (and, ideally, pattern) without triggering a bolus.

II. Enter grams of carbs in total (early carbs, extended cartbs inkl. FPU), plus a percentage to which these shall be covered by an upfront meal bolus.

It may be necessary to tune the IC value so the swinging of the glucose curve in the early phase remains in the desired green band. Another strategy would be to employ a „Superbolus", meaning, 2 hours worth of basal are pulled intoi the bolus, followed by two hours of zero-temping.

In the **late phase** (3) not yet absorbed carbs and transformed fat and protein components are absorbed, and the loop must now provide suitable insulin activity. The challenge there is that the activity from the meal bolus is fading out, and the loop algorithm must move front stage now to manage the meal. This will differ bewteen systemns and is discussed in the section on the late meal phase.

Source: Dana Lewis, https://github.com/danamlewis/artificialpancreasbook/



The on-board bolus calculator will suggest a bolus suitable to the carb input. You giving a bolus essentially means: NO LOOPING. The loop only watches what is happening with YOUR intervention, and tries to co-manage - which is virtually impossible at first . As a consequence, the Loop is setting 0% basal in response to your dangerous looking bolus. It is entirely YOUR responsibility that your bolus suits carb amount <u>and absorption</u> in this **early phase** of the meal.

It is important to enter only as many **g carbs** for determination of the meal bolus into the bolus calculator **as can be absorbed while the meal bolus is strongly active**. Usually this will be up to 60g for ~2hr of strong activity of fastest insulins, maximum up to 90g for ~3hr of slower insulins.

(An alternative could be to add more carbs, but a %age to which they shall receive an insulin bolus. This mode leads to higher cob when the late phase starts. (That could actually be beneficial for loops (iOS) that act strongly on carb inputs. An alternative would be to make 2 or more „eCarb inputs for the late phase, and that way front-loading absorbed carbs for the late phase).

You can tune your IC factor in closed loop, until you achieve glucose remaining in range in the first 2 hours after meal start. Try to match the pattern given with the 3 green (X) in the chart! Notably, shortly after maximum insulin activity of your bolus, you should be below target (if not, consider to strengthen (=to lower) your IC).

Timing of bolus

22

- The early meal management in hybrid closed loop resembles very much that of doing multiple daily injections.
  Differences between the looping systems are minor *), and relate to
  - how, and by how much, the algorithm of the loop will co-manage
  - how important exact user inputs regarding carb amount and absorption times are
  - how flexible the user is, to set different „aggressiveness" and targets (e.g. if sports follow)
- If you use one of the fastest insulins (Lyumjev, Fiasp), or eat low carb/high fat, or have unusual slow digestion (gastroparesis), it is usually OK to bolus at meal start.
- In all other cases, giving the bolus before meal start can lead to a better result (pre-bolus see p 18/19).

Note: Fastest insulin is not necessarily best for hybrid closed looping. A slower insulin, like Humalog, with proper pre-bolussing time, is likely to cover a meal better than the new super fast ones could.

*) Differences between looping systems are more pronounced in the late meal phase (when actually full closed looping takes place).

(read the slide)

Trouble shooting

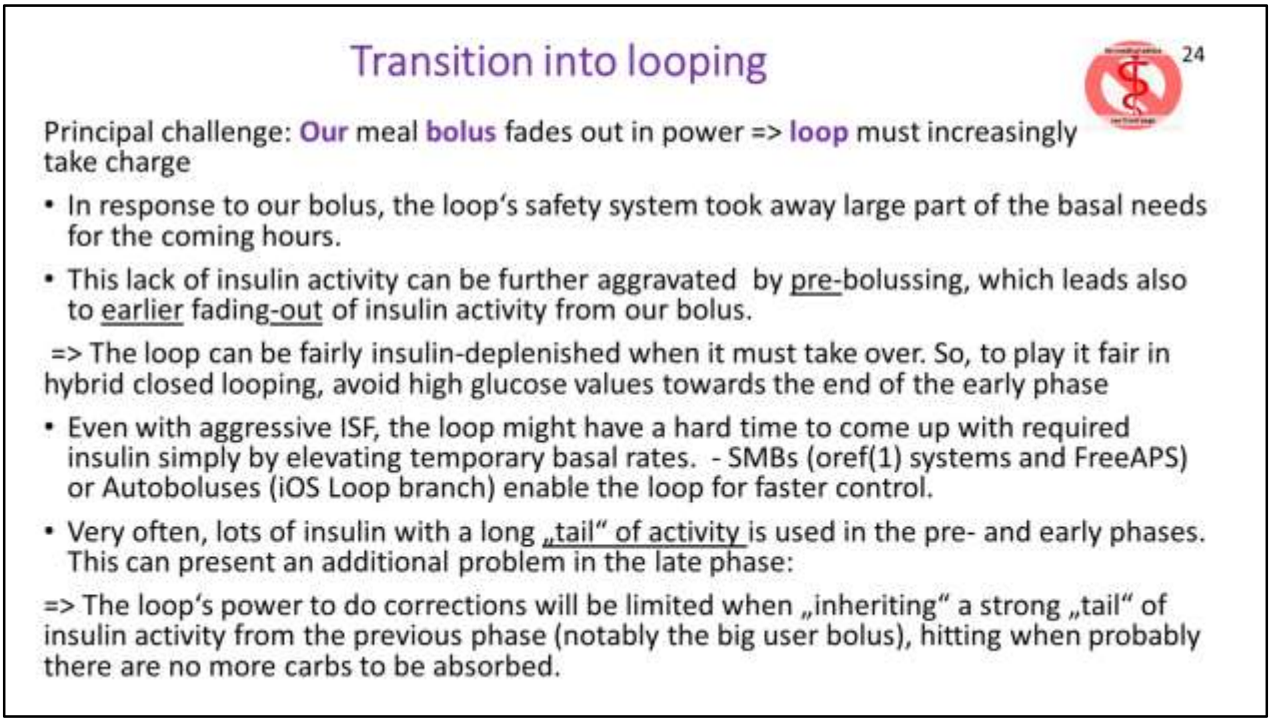
23

Matching insulin activity with carb absorption for a straight glucose curve is impossible.
However, to keep values in range (70-180 mg/dl) during the pre- and early meal phases should be your goal.
Avoid hypoglycemia and high values:

category	problem	likely cause	action, immediate ...	...for next day
hypoglycemia	around meal start	pre.bolus time too long	take glucose	reduce pre-bolus time
	shortly after max bolus activity	IC too strong (low)	take glucose	elevate IC
		too many carbs entered in calculator	take glucose	bolus for „early" carbs only
too high glucose		occlusion	new cannula, erase iob, add insulin	
		IC too weak (high)	watch and evaluate	reduce IC
		carbs underestimated	watch and evaluate	find out the true carb content

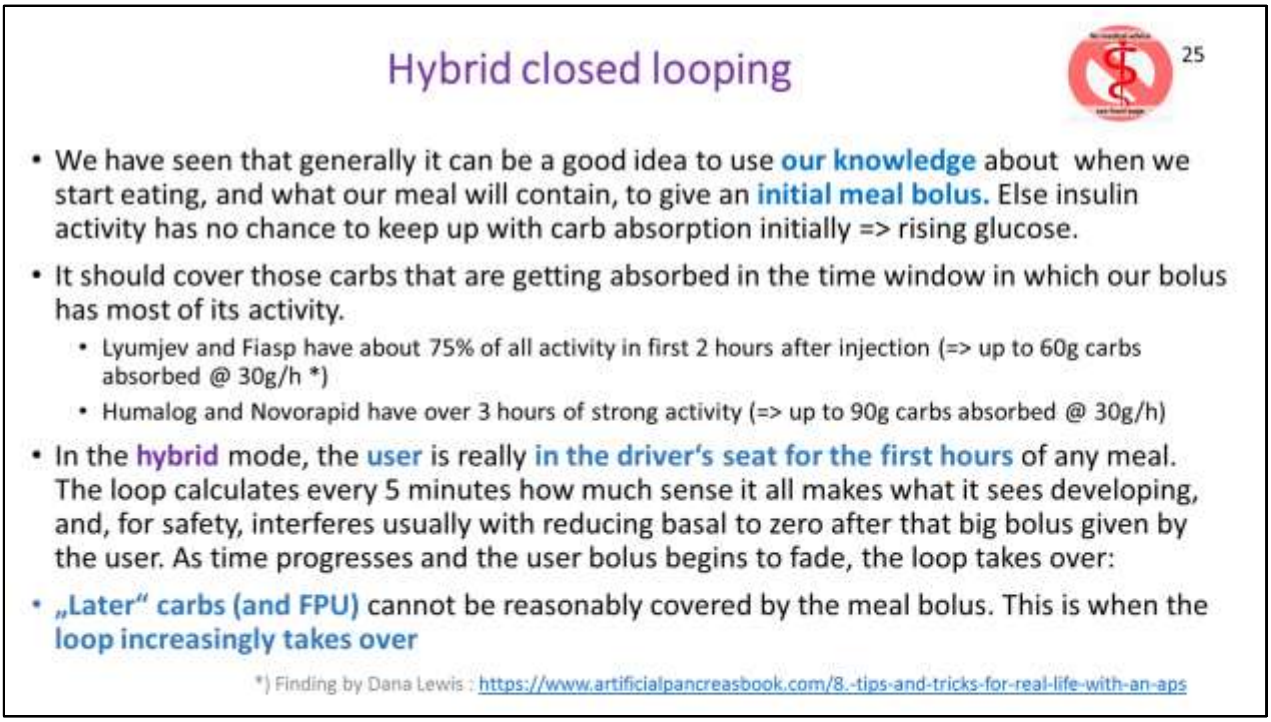
We broadly discussed the problem that constantly matching carb absorption and insulin activity is virtually impossible, and therefore we always will encounter ups and downs in our glucose level. Moreover, a lot of other factors can interfere. A list of 42 such factors is included as the very last slide.
Foremostly, we should avoid **hypoglycemia**. Especially with the newer super-fast insulins these can be dangerous. If they happen around meal start, look whether you need a shorter pre-bolus time. Also, setting an aggressive eating soon temporary target could make you drift towards a hypo, notably when being active prior to meal start, or if your ISF is set a bit too aggressive.
Another time-window in which hypoglycemia could appear is shortly after the maximum activity of your meal bolus. This would point either to a too strong IC (too low of a numerical value), or to too many carbs entered in the calculator when determining your bolus. This is actually a common beginner mistake, to enter all meal carbs there, regardless whether they can be absorbed in the time window of the bolus you are giving. So, in the future, limit the carb amount in the calculator to the early or fast carbs for your bolus; and reserve the „extended carbs" for the loop in the late phase. We will look into this in detail in the next chapter.
Regarding **glucose running high** beyond 180 mg/dl (or even over 250 mg/dl, with meals rich in high BG impact foods, see p.7): After excluding an occlusion as the cause (this is not our topic here; use search term elsewhere), there are two highly likely causes, a too-weak IC, or underestimating your carbs. Tweak your IC only if you were

able to rule out gross underestimation of carb amount (which I believe happens quite frequently regarding breakfast components).



As our meal bolus fades out in power, our loop must increasingly take charge to keep glucose under control while being faced with a stream of late carbs and sometimes ugly effects from so-called FPUs (fat and protein units).
Besides it being a principal challenge, to provide seemless performance when management totally changes (from you in the driver‘s seat to the loop), we specifically have the following issues.
„As a measure of courtesy“ we should not hand over to our loop with ridiculously high glucose value towards the end of the early phase (see last two lines of the table on the preceding page). The loop is faced with a tough job anyways:
In response to our bolus, the loop‘s safety system took away large part of the basal needs for the coming hours. The resulting lack of insulin activity can be further aggravated by pre-bolussing, which leads also to earlier fading-out of insulin activity from our bolus. So, the loop does not have a lot of active insulin when it must take over.
In fact, even with an aggressive ISF, it might have a hard time to come up with required insulin simply on basis of elevating temporary basal rates. SMBs (oref(1) systems and FreeAPS) or Autoboluses (iOS Loop branch) enable the loop for faster control. We will look into this in more detail in the next chapter.
(Alternatively, additional bolussing could be a solution, see 2 slides further down).
Very often, lots of insulin with a long „tail“ of activity is used in the pre- and early phases. This can present an additional problem in the late phase, because

the loop‘s power to do corrections will be limited when „inheriting“ a strong „tail“ of insulin activity from the previous phase (notably the big user bolus), hitting when probably there are no more carbs to be absorbed.



We have seen that generally it can be a good idea to use our knowledge about when we start eating, and what our meal will contain, to give an initial meal bolus. Else insulin activity has no chance to keep up with carb absorption initially, resulting in rising glucose.

The initial bolus should cover those carbs that are getting absorbed in the time window in which our bolus has most of its activity.

Lyumjev and Fiasp have about 75% of all activity in first 2 hours after injection (=> up to 60g carbs absorbed @ 30g/h *)

Humalog and Novorapid have over 2.5 - 3 hours of strong activity (=> 75 -90g carbs are absorbed @ 30g/h)

In the hybrid mode, the user is really in the driver‘s seat for the first hours of any meal. The loop calculates every 5 minutes how much sense it all makes what it sees developing, and, for safety, interferes usually with reducing basal to zero after that big bolus given by the user. As time progresses and the user bolus begins to fade, the loop takes over:

„Later“ carbs (and FPU) cannot be reasonably covered by the meal bolus. This is when the loop increasingly takes over

Additional bolussing for dessert or 2nd course?

26

In case you want to bolus for another course, or for a dessert:

1. Assume that you need not interfere with the situation from your last bolus and carbs from the prior course, no matter what the glucose, iob and cob levels currently are
2. **Divide the new amount of carbs by the IC** (carb ratio) **to calculate the new bolus.** (To do this in the bolus calculator, un-tick the boxes that, otherwise, would consider current glucose level, iob and „old" cob). <u>Do not enter more grams of carbs than can be absorbed while your new bolus is very active!</u> (Rest & FPEs -> carbs in the future entry.)
3. If the <u>amount</u> of new „extra" carbs is high and/or the <u>time gap</u> to the previously taken carbs and bolus is short, you might need to look into:

(a) whether a dangerously high iob, followed by an insulin activity maximum, might result – which could be dangerous …

(b) …if there is a bottleneck, so the extra carbs can really not be absorbed right away (usually there seems to be a 30g/h limit, for overall carb absorption).

If this is the case, reduce the bolus for the second course; ideally, also input some of the carbs as carbs in the future (eCarbs).

If you are having a substantial second (or third) course, or a desert, you could be tempted to bolus for that yourself, instead of „waiting for the loop" to take care. In case your loop has only temporary basal rates to work with (i.e. no Autobolus or SMBs), this can actually be a very good idea,
To define the appropriate bolus for the next course or dessert:

1. Assume that you need not interfere with the situation from your last bolus and carbs from the prior course, no matter what the glucose, iob and cob levels currently are
2. Divide the new amount of carbs by the IC (carb ratio) to calculate the new bolus. (To do this in the bolus calculator, un-tick the boxes that, otherwise, would consider current glucose level, iob and „old" cob). <u>Do not enter more grams of carbs than can be absorbed while your new bolus is very active!</u> For the rest, plus for any FPU equivalents, make carb entries with absorption time in the next hours.
3. A tricky problem arises, if the amount of new „extra" carbs is high, and/or the time gap to the previously taken carbs (and bolus) is short. Then look into:

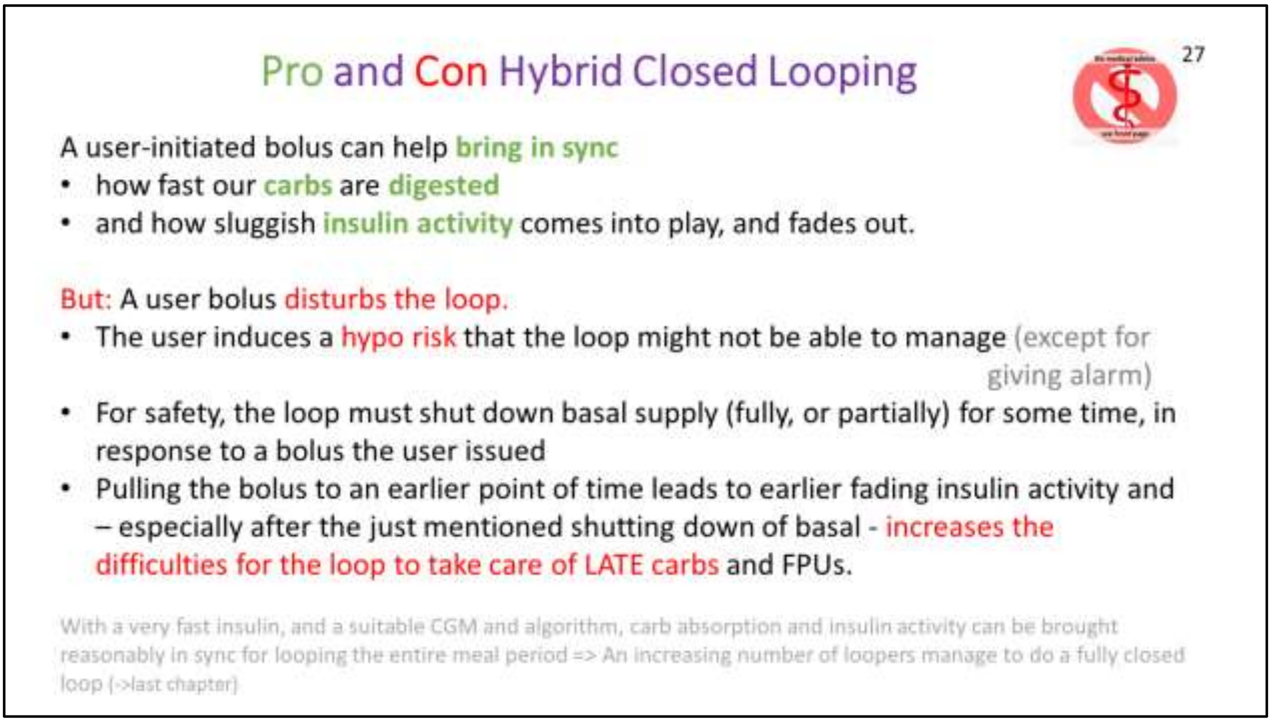
(a) whether a dangerously high iob, followed by an insulin activity maximum, might result – which could lead to a hypoglycemia …

(b) whether there is a bottleneck, so the extra carbs can really not be absorbed right away. (Usually there seems to be a 30g/h limit; less in gastroparesis).

In that case, reduce the bolus for the second course, and input some of the carbs, and any fat and protein units from the second course, into a new future carbs entry for the

loop to take care of later, when your 2nd bolus begins to fade out.

If activity/sports follows after a meal, see p.40.



A user-initiated bolus can help bring in sync

- how fast our carbs are digested
- and how sluggish insulin activity comes into play, and fades out.

And, it is what we were used to. (Actually, we may miss some of the refinements we had, like multi-bolus).

However: A user bolus disturbs the loop.

The loop has a set of rules (an algorithm) how to deal, every 5 minutes, and based on mathematical predictions, with any situation. And then you come in, with a very coarse idea, and brute bolus … It is like on a ship with 2 captains with extremely different attitudes …

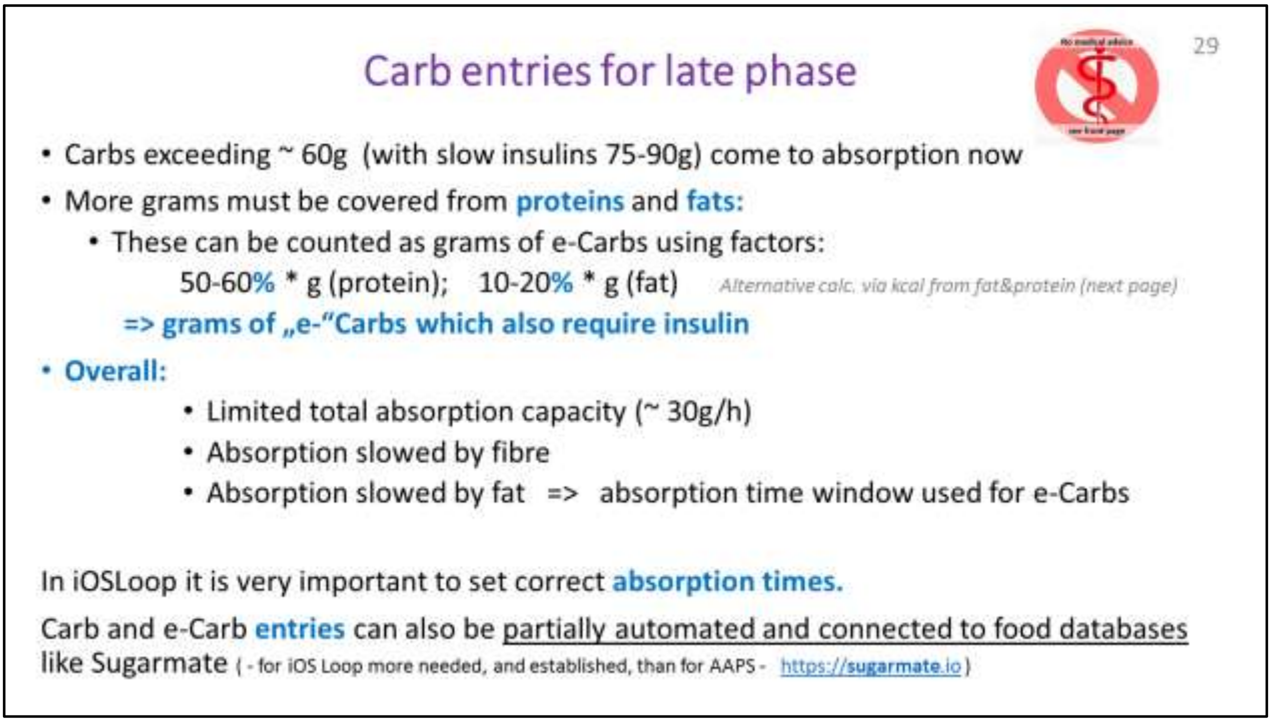
- The user induces a hypo risk that the loop might not be able to manage (except for giving alarm)
- For safety, the loop must shut down basal supply (fully, or partially) for some time, in response to a bolus the user issued
- Specifically when pre-bolussing, pulling the bolus to an earlier point of time leads to earlier fading insulin activity, which – especially after the just mentioned shutting down of basal - increases the difficulties for the loop to take care of LATE carbs and FPUs.

Still, it is usually a good strategy to focus on the immediate problem first (to pre-bolus against a strong glucose rise). Any „side-effects“ from that, plus new occurring challenges, can be managed (in more than 1 way), thereafter. Managing that late meal phase is the topic of our next chapter.

With a very fast insulin, a suitable CGM and algorithm, carefully determined, not too-unstable „profile“, and avoiding meal extremes, carb absorption and insulin activity can be brought good-enough in sync for looping the entire meal period => An increasing number of loopers manage to do a fully closed loop (->last chapter) .

Some medical authors even postulated that in pprincuipal they could see full closed loop outperform hybrid closed loop: https://www.researchgate.net/publication/351273207_Full_closed_loop_open-source_algorithm_performance_comparison_in_pigs_with_diabetes „As algorithm design for closed loop systems continues to develop, the strategies employed in the OpenAPS algorithm (known as oref1) as implemented in AndroidAPS for unannounced meals may result in a better overall control for full closed loop systems.”

Now we are getting to the late phase of a meal which is where the loops really take over the management, and take care of the late carbs and fat/protein units (FPU). How they do this 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-correcct high values, and how to deal with thosem 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.



For the late phase, i.e. the time window when our meal bolus lost most of its power *) we must enter the carbs that will be absorbed after this time window.
We just inform the loop of how much (gram amounts) and about when (absorption times) more carbs will be absporbed 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. Carbs button in AndroidAPS).
The **„eCarb“ amount** consists of:

- Carbs exceeding ~ 60g (with slow insulins 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); or 10-20**%** * g (fat).
- Also lower %age for protein and higher %age for fats is sometimes suggested *(see remarks in comments at bottom)*
- *Alternatively there is a conversion via kcal from fat&protein (see next page)*

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 .

3 pages later we will look into this in more detail.
Carb and e-Carb **entries** (amounts and absorption time info) can also be partially automated and connected to food databases like Sugarmate. (As we will discuss in

the last chapter, detailing the eCarbs carefully is not necessary with oref(1) systems).
----
*) Rule of thumb, 75-80% lost, this is about 3 hours after bolus for Humalog; or 2.5 hours for Fiasp or 2 hrs with Lyumjev
Food databases: https://**sugarmate**.io Fleisch - Tierische Produkte - Kalorien – **Fddb**
General overwiew, effects from proteiens and fats:
https://www.youtube.com/watch?v=4_5K9zH9CFc
Regarding lower %age for proteien see e.g. here:
https://www.youtube.com/watch?v=4_5K9zH9CFc
Regarding higher %age for fats, see e.g. here: and p.51

30

Fat-protein units

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**	[illegible]	27	4	23	4,1	185	4,9
Alcohol	**20**	**0%**				7,1	142	
total			64	39	25	*5	773	

Two main meals (example)

meal rich in carbs

component	g	% carb	carb	g carb in 1st 2,5 h	late carbs	kcal/g	kcal	FPU
Carb	**60**	**100%**	60	55	5	4,1	246	
Fat	**30**	**10%**	3	1	2	9,3	279	361
Protein	**20**	**60%**	12	2	10	4,1	82	3,6
Alcohol	**20**					7,1	142	
total			75	58	17	*5	749	

Late carbs from fats and proteins can be calculated:

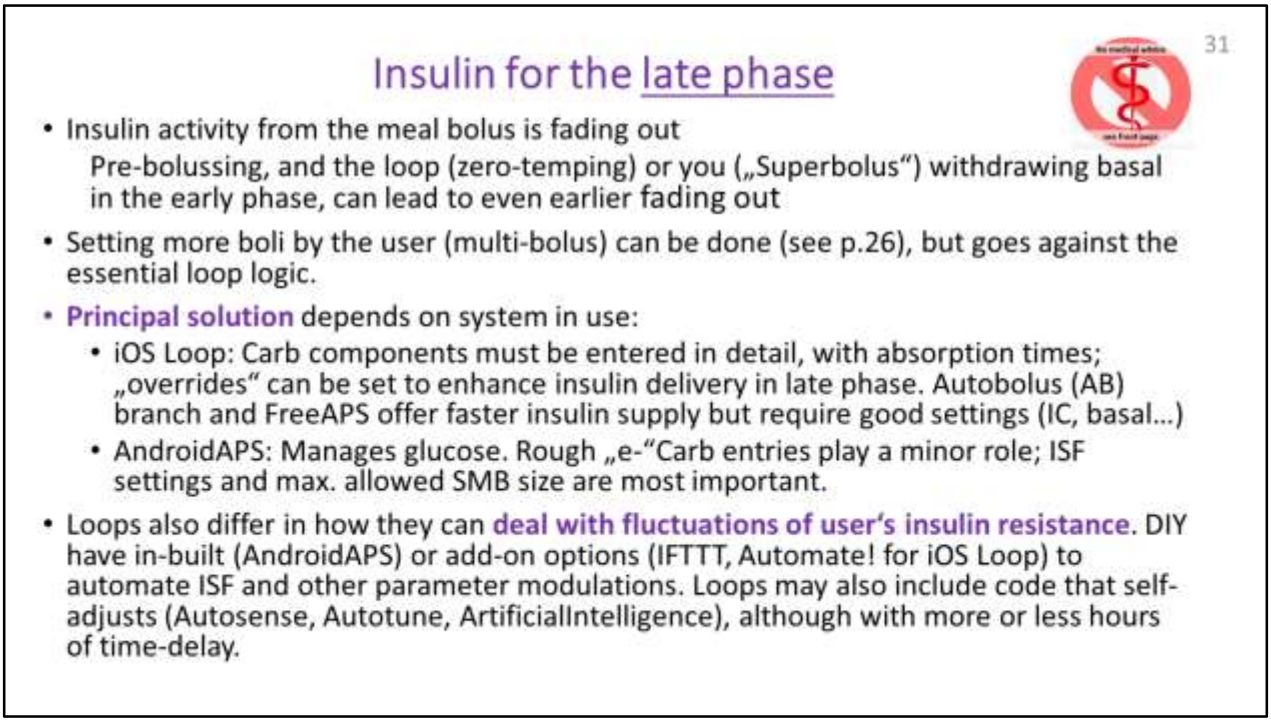
a) via FPU, see yellow fields
FPU*5 gives the grams eCarbs
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5686679 ;
(up to *10 is used by some *Pediatr Diabetes. 2009;10(5):298–303. [PubMed]* , who deduct for „safety" afterwards)

b) via factors like 50-60% converting g Protein to g *mostly late* Carbs, and 10-20% for fat, see green darts.

High fat (or fibre) also slows absorption of originary carbs

With this table we like to show for a meal rich in protein, and anothert one richer in carbs, how fat and protein can be counted into late carbs, and what reasonable entries would be in the bolus calculator (first 2.5 hours covered by bolus with rapid insulin) and into eCarbs (late carbs amount, absorbed in 3rd – 7th hour).
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 arrows).
A (safer) factor of 5 is suggested e.g. by https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5686679 ;
up to *10 is used by others, e.g. *Pediatr Diabetes. 2009;10(5):298–303. [PubMed]*.
Alternatively, and faster to calculate in everyday life, the grams of protein and of fat can be converted into eCarbs (green arrows) by using factors of 60%, or 10%, respectively.
The factor on fat may seem low. This may, at least in part, be owed to the secondary effect of fats to stretch out the absorption of all carbs, which makes it seem that more late carbs originate from fats. (See also slide 51).

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



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 (p.10) 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.

Or absorption follow largely from inputs about **carbs eaten** and their absorption times *), which is the basic approach of iOS Loop.

Of course, in the end, both systems take into account all available information, and as user errors as well as a myriad of secondary influencing factors (see p.58) 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.

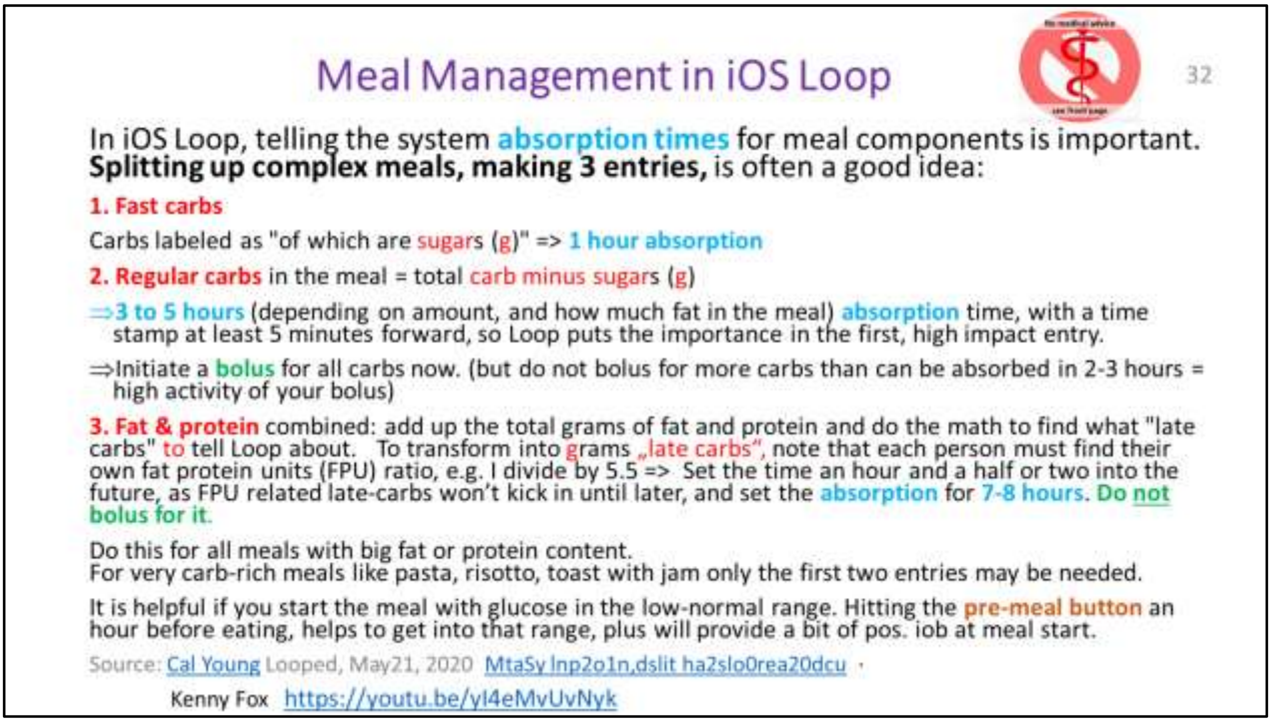
Predictions are the key part driving the loop‘s conservative behaviour regarding **hypo** prevention, **from the „tail“ of any insulin, that might be desirable to have on bord**

earlier.
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 AndroidAPS 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 decison.
Loops differ also in how they can deal with fluctuations of user‘s **insulin resistance**. DIY have in-built (AndroidAPS) or add-on options (IFTTT, Automate! for iOS Loop) to automate ISF and other parameter modulations. Loops may also include code that self-adjusts (Autosense, Autotune, ArtificialIntelligence), although with more or less hours of time-delay
----
*) 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/
How each algo calculates InsulinRequired differs:
(a) AndroidAPS (and similar OpenAPS, FreeAPS-X):
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 remains within a pre-specified target. The user, similar to standard insulin pump therapy, inputs their personal basal rate, insulin-to-carbohydrate ratio (IC), insulin sensitivity factor (ISF), and curve of insulin pharmacodynamics. OpenAPS and AndroidAPS also **enable automated insulin coverage of meals without carbohydrate announcement via the “unannounced meal” feature**, whereby meals are anticipated based on available data suggesting an otherwise unanticipated glycemic excursion. These unannounced meals may then be acted upon by a supermicrobolus (SMB) wherein tiny boluses are delivered to more rapidly affect rising glucose levels. Other features, not specific to unannounced meals, such as “**auto-sensitivity**” respond to glucose fluctuations beyond the scope of predictions. These features used in combination enable some OpenAPS and AndroidAPS users to utilize the open-source system in a full closed loop mode.
The actual used data (every 5 minutes) can be looked up „live“ in the AndroidAPS SMB

tab; or, for any 5 minute segment in the past, in the AndroidAPS logfiles; -> phone storage/emulated/0/Android/data/info.nightscout.androidaps/files. There: „AndroidAPS.log“ contains the most recent data, and „AndroidAPS_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 Github / ga-zelle / determine-basal-emulator: This 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 (and similar AB branch, FreeAPS):

The other open-source AID system, 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, ICR, 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.



iOS Loop benefits from differentiated inputs of meal components by clusters of absorption times.
Therefore, **splitting up complex meals, making 3 entries,** will usually be a good idea:

1. Fast carbs

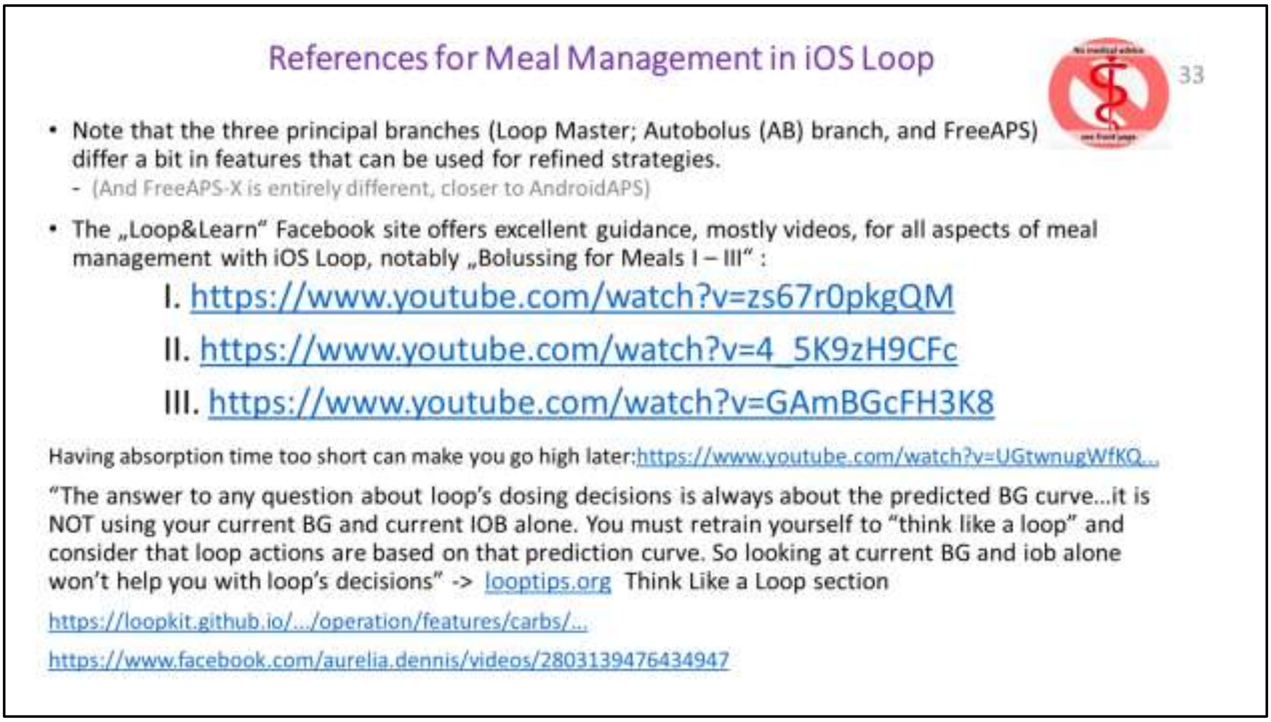
Carbs labeled as "of which are sugars (g)" are entered with a **1 hour absorption** time

2. Regular carbs in the meal are the total carbs minus the sugars (g). They are entered with a **3 to 5 hours absorption** window (depending on amount, and how much fat is in the meal); a time stamp of at least 5 minutes forward is applied, so Loop puts the importance on the first, high impact entry from the fast carbs.
⇒Initiate a **bolus** for all carbs now. (But do not bolus for more carbs than can be absorbed during the high activity of your bolus, which should be 2-3 hours worth of carbs absorbed, or ~60 (Lyumjev) – ~90 (Humalog) grams. YDMV, e.g. less in gastroparesis. But less also if accompanied by high fibre or fat).

3. Fat & protein combined: Add up the total grams of fat and protein and do the math to find what "late carbs" to tell Loop about. To transform into grams „late carbs", note that each person must find their own fat protein units (FPU) ratio, e.g. I divide by 5.5 (See also p.30 for how to calculate, or use software like FreeFPU).
=> Set the time an hour and a half or **two hours into the future**, as FPU related late-carbs won't kick in until later, and set the **absorption** for **7-8 hours**. **Do not bolus for it**.
Do this for all meals with big fat or protein content. For very carb-rich meals like

pasta, risotto, toast with jam only the first two entries may be needed.
It is helpful if you start the meal with glucose in the low-normal range. Hitting the pre-meal button an hour before eating, helps to get into that range, plus will provide a bit of pos. iob at meal start.
Users of AB branch might enter carbs 30 minutes before eating with time stamp(s) at the correct time(s). At a moderate diet, autoboluses might be able to take care of the rest (AB does default 40% bolussing, plus corrections as needed).
Regarding future carb entries...Think of this as entering a future rise that will need insulin. Loop will dose based on the prediction and correction range.
For a deep dive on how Loop chooses to dose based on this information, watch this video. The core content is 30 minutes or so.
https://youtu.be/yI4eMvUvNyk
After watching this, you'll understand why you could consider entering future carbs but also why a higher correction range override would make that safer. (Source: Kenny Fox/L&L May2022)

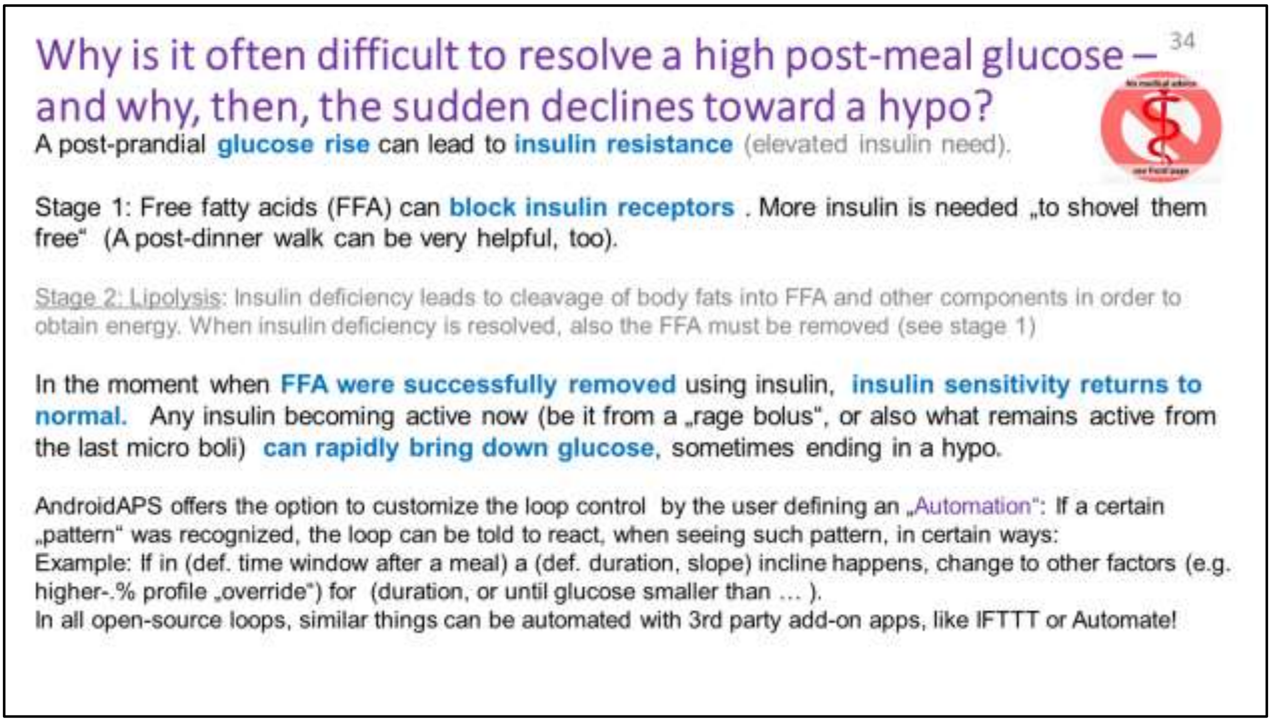
Source: Cal Young Looped, May21, 2020 MtaSy lnp2o1n,dslit ha2slo0rea20dcu ·
Programming icons for frequently used foods in iOS Loop see e.g.:
https://www.loopandlearn.org/custom-code/?fbclid=IwAR1_B--X6gj4WGMR-lTdnYkNpA5-sgzHD4zWc7ePUT-7CXZnrxU8zaG5yG0#food-emoji
Apps that supports calculating carbs : LoseIt! ……………….(add others)



Here are a few of the available references specific to iOS Loop.
Note that the three principal branches (Loop Master; Autobolus (AB) branch, and FreeAPS) differ a bit in features that can be used for refined strategies.
- (And FreeAPS-X is entirely different, closer to AndroidAPS) -
The „Loop&Learn" Facebook site offers excellent guidance, mostly videos, for all aspects of meal management with iOS Loop, notably „Bolussing for Meals I – III" **).
The answer to any question about loop's dosing decisions is always about the predicted BG curve…it is NOT using your current BG and current IOB alone. You must retrain yourself to "think like a loop" and consider that loop actions are based on that prediction curve. So looking at current BG and iob alone won't help you with loop's decisions – Occasionally studying a couple of pages in the linked materials (and further references you will find there) is highly recommended.
(Still, no need to make an exact science out of it. If nothing else, the sheer number of influencing factors we struggle with - see very last slide (p.59) – should teach us, that gradually increasing our understanding OVER THE YEARS, while facing occasional puzzles and setbacks "out of nowhere", is what we must deal with , accept, and move on.)

**) It would be nice if a AndroidAPS Hybrid Closed Looper would compile a case study file like section III. from Loop&Leran here.
Some available material from Meal Management I. und II. in AndroidAPS Users / Files

could be extracted for that, but more and newer examples should be added.
(The author of this slide set moved on to full looping, wants to focus on that topic from now on, and also simply cannot come up with new material for meals in hybrid closed loop).



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

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

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 (p.11-12) the nautical parallel 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

36

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

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 also 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.7) should help to dial in all parameters right.

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

(read through the cases)

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.) favourite 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“.

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.

----

*) 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 <u>within</u> the AndroidAPS 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

	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 <u>normal</u> 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.

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.
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, Our 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 ist 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 discussed p.33-34.
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 <u>normal</u> glucoose target. A sportsTT would need to be set very early, but lead to a weaker overall meal management. A good alternative would be a normal managed <u>reduced</u> meal size, then a sports <u>snack</u> accompanied by a late set ActivityTT.

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 girlfiend Afrezza" does a CANNONBALL JUMP next to the front left side of our boat. The resulting >>>-wave orients our boat back towards target
Pro: Like p.33/34 we address 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.
But it is over when it is over. Or, is it? The loop saw the problem, and that it resolved, but in his model cannot attribute how.=> False Autosense, false insulin and carbs data => The short term problem is solved, but there are consequences in the upcoming hours from skewed calculations and eventually also (via Autotune-driven basal and factor adjustments) for the next days: Taking the effect from Afrezza for granted, but really not knowing about it, the message for our loop is, we are doing fine with the loop-provided insulin!
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).

However, best try to always **stay in range**. Crazy actions like in the picture will rarely be needed by experienced navigators.

Temptation to cheat the loop („fake“ data entry)

39

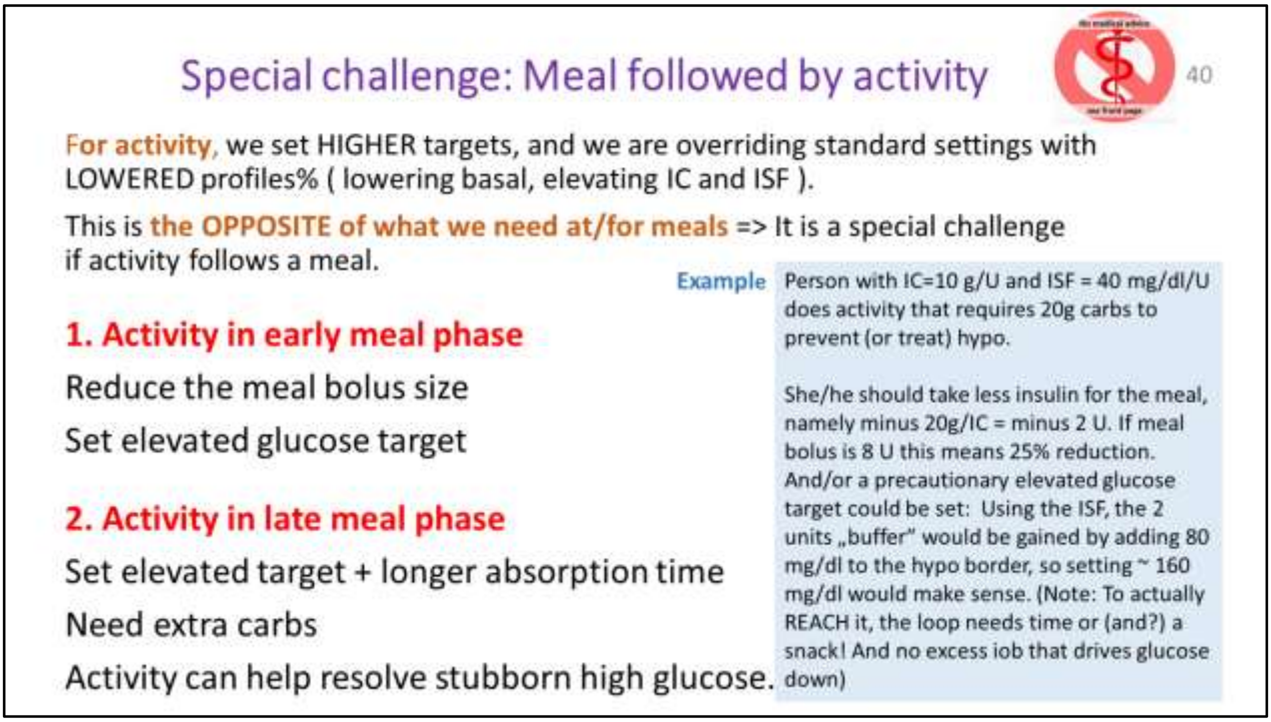
Trying to provoke a certain behaviour of the system via „nudging“ it with fake inputs can certainly work for some intended effects. BUT: Generally, we want our loop to work smoothly, and have the best information we can give it.

	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

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.

To be discussed / eventual milder evaluation by iOS loopers ?



Generally for activity/sports you will benefit from getting less insulin, and from having an elevated glucose level - with „room" for it to go down **)
The blue box gives an example *(read it).*
So, **for activity**, 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 activity 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 activiry 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 activity, it very much depends on kind, duration, starting time relative to the meal. And as we have seen throughout this presentation, meals and kind of insulin in use/bolussing strategy plays a decisive role as well.

There are extra resources (videos, blogs, FB posts) on sports management when looping. 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.

**) Exception: In very stressful situations, glucose may be rising temporarily also during sports, and might even briefly require extra insulin to supply „panic“ energy

41

Topics

We have touched on a lot of strategies and measures that can be taken in an effort to manage meals when using closed loop systems. The next slide sums up the options for meal management, and should be considered a „toolbox“ if for special challenges new ideas are sought.
Specifically, users with systems that incorporate the oref(1) algorithm (OpenAPS, AndroidAPS, Free-APS-X) could „ease up“ after investigating for themselves, whether the „good practice“ proposal for easy meal management in hybrid closed looping (p.42) - or even going all the way into full closed loop (p.43-46) - could work for them.

„Toolbox“ for meal management when looping

	Start	Carbs	FPUs	Absorption kinetics	IC factors	Interfering with the loop	Variations 42
	Do nothing	No entries	Not considered	Not considered	IC unimportant	None (except evtl. TT)	Tuning as you go (IC, ISF, „Aggressive-ness“)
	Do nothing prior to meal start, then enter carbs	Bolus for <= 60g	Very rough estimate -> eCarbs	% set in bolus calculator	IC from „trial and error“	Revised/fake late carb entries	
Red: simple mode that allowed me to reach *in AndroidAPS* ~ 87%TIR	Pre-bolus the entire meal bolus (~ 15 min before start)	Estimate for total, and %split for bolussing	FPU ~ kcal/100; factor -> C: 5 ...10?	Carb>60 + rough est. for FPUs => eCarb input for hours 3 – 5...7 *(if hi fat, fibre)*	IC from ~3 h experimental observation (< 60g)	Open Loop/ multi-bolus Afrezza Addit. bolus (if calculator advises in 3rd h)	Vary as seems suitable for different challenges
Efforts to consider FPUs etc. -> Good practice for *other loops*	Small pre-bolus (>30 min before start)	Exact inputs (amounts)	P -> C factor 50-60%; F -> C factor 10-20%?	carbs equally distributed over absorption time	IC from Autotune or from „AI“	Temp.% profile (iOS: overrides) at hi glucose	Only vary things if certain goals (like 7d TIR) are no longer met
	EatingSoonTT (~1 h before meal)	Exact inputs (+ abs.time)		Differentiated eCarbs inputs for meal components	IC from daily total balance including FPUs	Bolus out of impatience or frustration	

With this table we try to give an overview of the **options** for meal management.
For all open-source loops, the **golden framed** options should work well.. They incorporate a very detailed treatment of late carbs and FPUs.
*So, you would pre-bolus for the meal and enter exact carb amounts and absorption times, considerimng also fats and proteins.
*This is not just one entry, but differentiated for meal components.

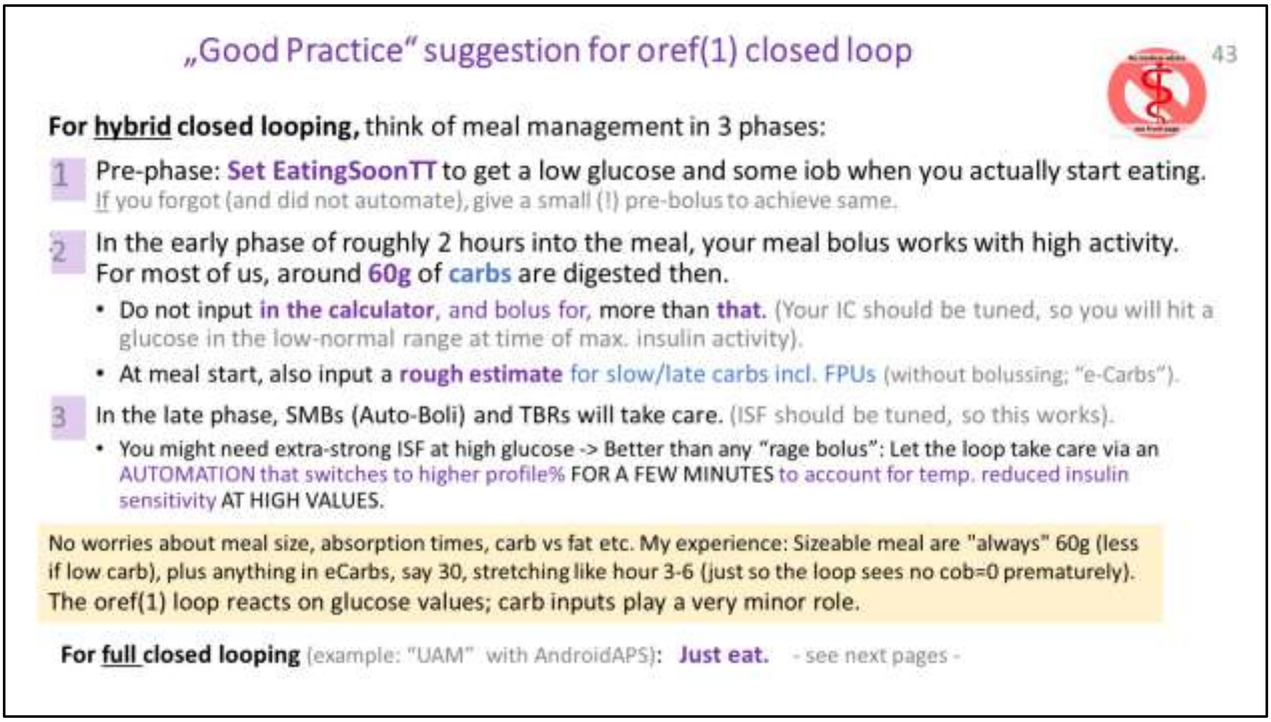
- An well-determnined, and tuned, carb ratio „IC“ is very important
- Complicatiobns are likely managed via poverrides/profile changes; also Afrezza might be helpful in certain situations.

With AndroidAPS, the **red** boxes can provide a sufficiently good starting point:

- Enter total carbs at meal start, and estimate the %of it getting bolussed right away.
- Do not worry about absorption times or FPUs
- Experimentally determine, and fine-tune, your carb ratio (IC).
- Optional additional bolus in 3rd hour (notably if not in SMB / Autobolus mode).

Rather than discussing more details here, we move on to the next slide with a slightly refined procedure proposed as „good practice“ when using AndroidAPS in hybrid closed loop.
In the end, everybody must decide for her/himself how to handle meals in looping.
Suitable solutions depend on a variety of **factors:** The alogorithm in use; the prevalent kinds of diet, the lifestyle,, the targeted TIR, and the accepted effort.
Some systems or modes require more **upfront** effort to get the system going well, but

less of an effort **everyday**, and vice versa. The extreme case in this respect is the full closed loop we will look into in just a moment.
This table can be looked at as a **toolbox** from which to pick whenever special challenges are ahead, maybe requiring sharpened tools, or alternative things to try.



The oref(1) loop is included in OpenAPS, AndroidAPS and the new FreeAPS-X. This algorithm reacts primarily on glucose values, and carb inputs play only a minor role. However, tuning the profile is of great importance for the oref(1) loop to work well. Main components of the profile are the insulin model, and basal rate, IC and ISF values suitable to the personal 24h sensitivity pattern, and selecting suitable safety settings,

Meal management then can be simplified with the se systems **in hybrid closed looping** as follows;

1) In the pre-phase, **setting EatingSoonTT** to get a low glucose and some iob when you actually start eating. Of note, in contrast to pre-bolussing, this is not time-critical at all and can be automated e.g. for school or work days. Also, if you (did not automate and) forgot, you can give a small (!) pre-bolus to achieve the same. Even doing nothing at all is a viable possibility coming with about a 30-40 mg/dl higher peak for half an hour or an hour, which can still be compatible with the desired time-in-range.
2) Use the **bolus calculator** to define a meal bolus for the early phase of roughly 2 hours into the meal. While your **meal bolus** works with high activity, for most of us (and depending a bit on the insulin in use), around **60g** of **carbs** are digested.
   Do not input **in the calculator**, and bolus for, more than **that**. Tune your IC, so you will hit a glucose in the low-normal range shortly after time of maximum

insulin activity.

At meal start, also input a **rough estimate** for slow/late carbs incl. FPUs (without bolussing; “e-Carbs”).

In the **late phase**, SMBs (Auto-Boli) and TBRs will take care. (ISF should be tuned, so this works well).

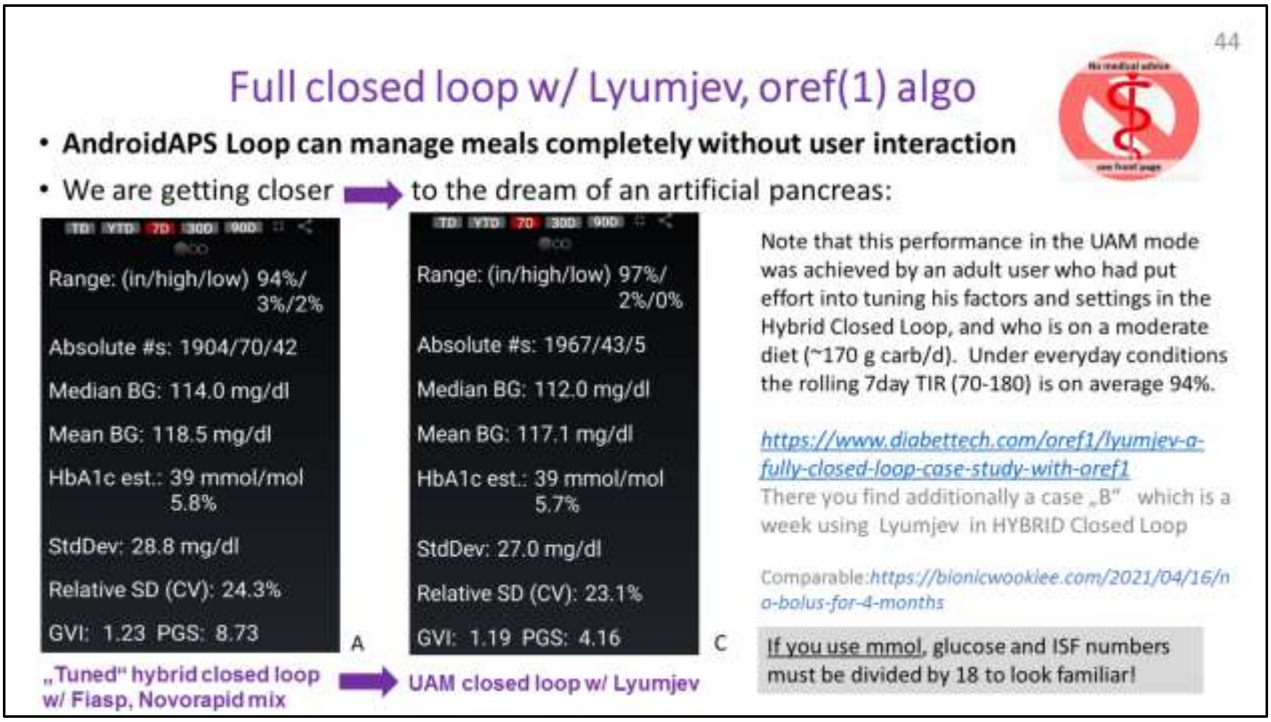
You might need extra-strong ISF at high glucose -> Better than any “rage bolus”: Let the loop take care via an AUTOMATION that switches to higher profile% FOR A FEW MINUTES to account for temp. reduced insulin sensitivity AT HIGH VALUES (see p.33)..

So, no worries about meal size, absorption times, carb vs fat etc. Sizeable meal are "always" 60g (less if low carb), plus anything in eCarbs, say 30, stretching like hour 3-6 (just so the loop sees no cob=0 prematurely).

For <u>full</u> closed looping (as possible with the “SMB+UAM” setting in AndroidAPS):

Just eat!

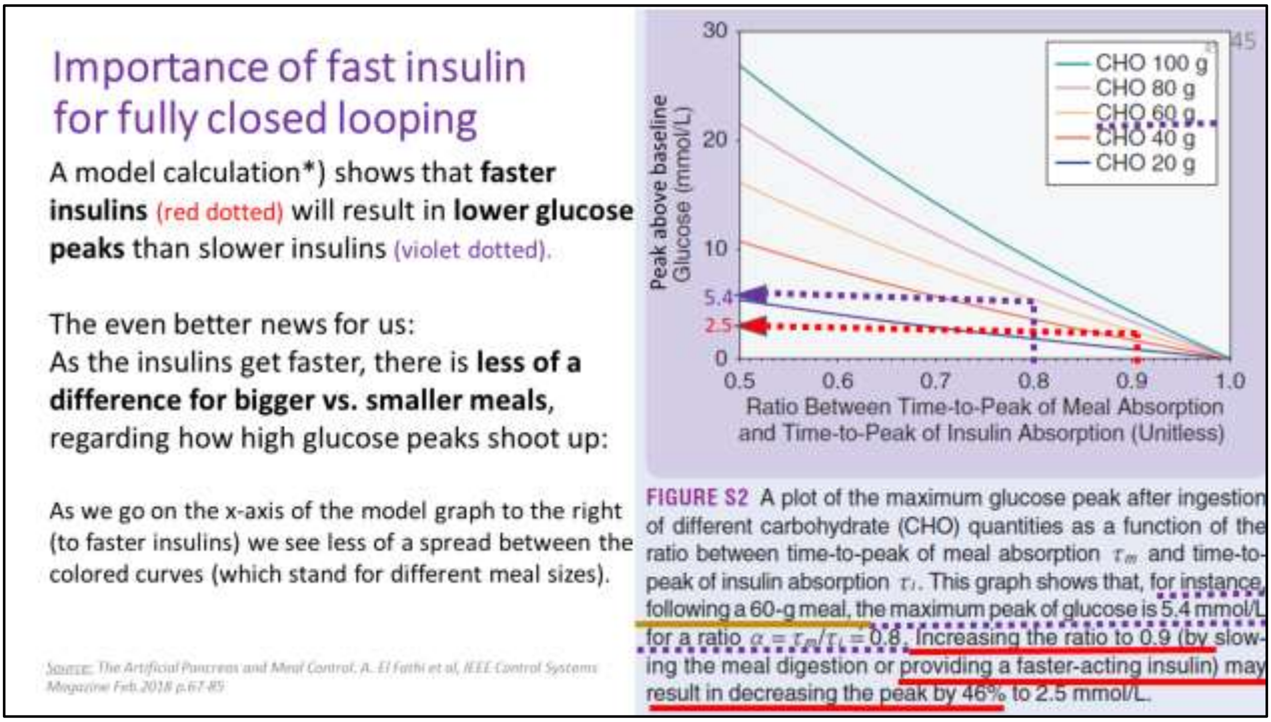
- May sound unbelievable, but see next pages -



https://www.youtube.com/watch?v=IEpEgMdnrAA : Did somebody say „Just eat?" (SnoopDog).
With a very fast insulin, a suitable CGM and algorithm, carefully determined, not too-unstable „profile", and avoiding meal extremes, carb absorption and insulin activity can be brought good-enough in sync for looping the entire meal period. An increasing number of loopers manage to do a fully closed loop. Here an example of an AndroidAPS looper switching from hybrid to full closed loop.
It is possible to **just eat!**
These xDrip statistics compare:
LEFT: one week in hybrid closed loop (A) giving carb inputs and user boli (Fiasp/Novorapid 50/50).
94% TIR (70-180 mg/dl) was achieved with 42 values under 70 mg/dl.
RIGHT: one week in full closed loop, unannouced meals (UAM) using Lyumjev.
97% TIR (70-180 mg/dl) was reached at only 5 values under 70 mg/dl.
During this week (C) no carbs were entered into the system, and the user gave no bolus.
These results were achieved under carefully controlled trial conditions. The user finds about 4%points lower TIR on average in everyday situations. Also it should probably be noted that he follows no special diet, but avoids sweet drinks and bigger breakfasts. YDMV
We can conclude that a significant step towards having an aritificial pancreas is being

made, by an increasing number of loopers with oref(1) algorithm and rapid insulin.

Source: https://www.diabettech.com/oref1/lyumjev-a-fully-closed-loop-case-study-with-oref1
https://bionicwookiee.com/2021/04/16/no-bolus-for-4-months
(in German:) https://de.loopercommunity.org/t/loopen-ohne-bolus/5955



Modelling the course of carb absorption and of insulin activity, the effects on glucose peak heights can be quantified. The graph shows on the y axis peak over baseline (in mmol/dl), and on the x-axis the relative speed of insulin absorption to carb absorption (carb absorption is faster, therefore all values are under 1.0). (See also p.14).

A model calculation shows that **faster insulins** (red dotted) will result in **lower** glucose **peaks** than slower insulins (violet dotted).

As probably everybody painfully is aware of, bigger meals aggravate the situation (see different colored curves).

Another great piece of news for us:

As the insulins get faster, there is **less of a difference for bigger vs. smaller meals**, regarding how high glucose peaks shoot up:

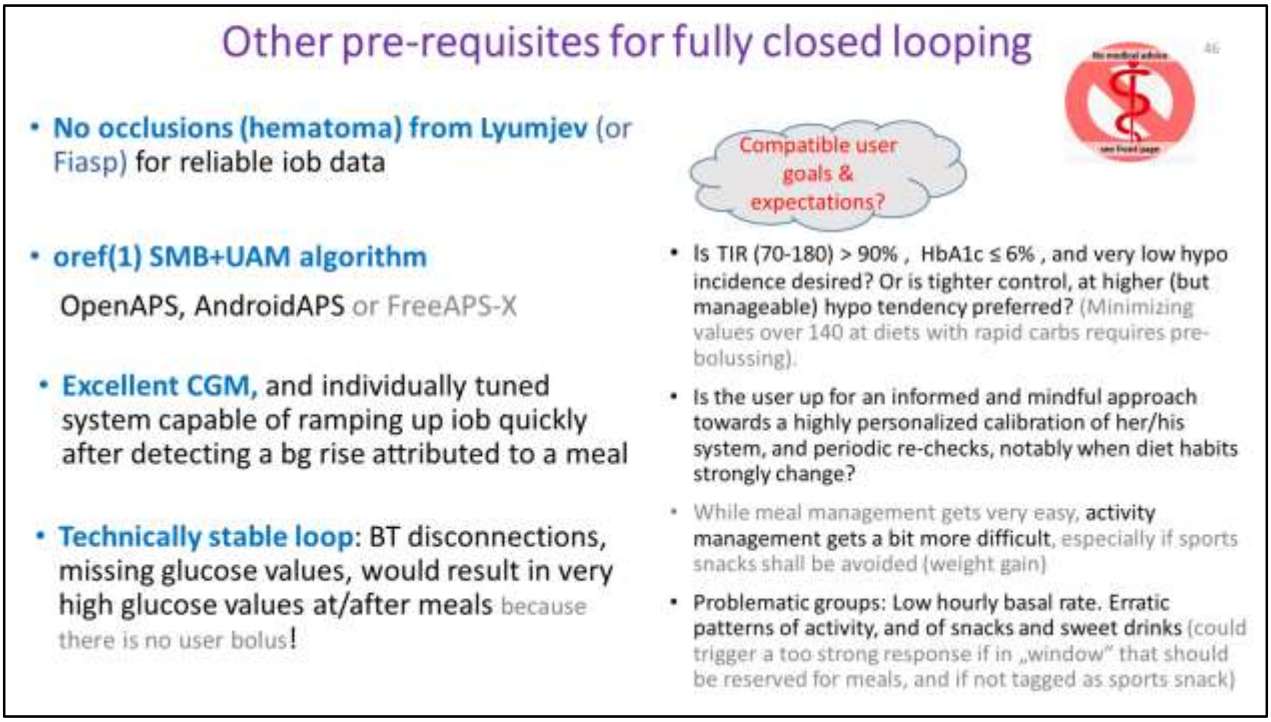
As we go on the x-axis of the model graph to the right (to faster insulins) we see less of a spread between the colored curves (which stand for different meal sizes).

So, this model supports that using a faster insulin will

- lead to less high glucose peaks, notably for bigger meals
- or might **tolerate** a couple of minutes **delayed** first meal bolus while not incurring unacceptable height of peaks.

The latter is a pre-requisite for full closed loop, in which we leave it up to the loop to notice that a meal „must have started“, and to come forward with SMBs that are typically delayed compared to the bolus as given in hybrid closed loop.

Source: The Artificial Pancreas and Meal Control. A. El Fathi et al, IEEE Control Systems Magazine Feb.2018 p.67-85. This is a very thorough mathematical modelling study, with 147 references to other literature. However, any modelling must make simplifying assumptions. So, our everyday reality is more complex, but the authors do carve out, even in quantitative terms, some core mechanisms at play.



Then, why is it that full closed loop is not broadly in use yet?
We just discussed the importance of a fast insulin. **Lyumjev** is not available in all markets, and generally has no registration for kids. Fiasp seems a somewhat inferior but possible alternative. However, many users of these rapid insulins suffer from side-effects from the additives. Some are just inconvenient or painful, others (tendencies for hematoma and occlusions) threaten the function of the full closed loop.
Other factors that limit broad adoption are:
The **oref(1)** algorithm is essential *) but only available in AndroidAPS and OpenAPS; in a late developmental stage of system development now also for Apple phones in FreeAPS-X. People attempting full closed loop should first establish a very well functioning hybrid closed loop for themselves. Main reason: Full closed loop requires detailed knowledge of the individual patterns after meals, so the insulination in full closed loop can be modelled after the best patterns seen in hybrid closed loop (settings, automations; there is a separate slide set on „UAM“ Looping, with more details on that).
In the full closed loop mode, it is absolutely essential to have a well-functioning **CGM system.** This is the basis for an individually tuned system, capable of ramping up iob quickly after detecting a bg rise which automatically and safely can be attributed to a meal.
Likewise, it is essential that CGM and pump **communication** (Bluetooth) is absolutely **stable,** notably after starting a meal.

But even if all these preconditions are met, the (currently available) **full loop may not be for everyone:**

- Is TIR (70-180) > 90% , HbA1c ≤ 6% (at very low hypo incidence) desired? Or is tighter control, at higher (but manageable) hypo tendency preferred? Minimizing values over 140 at diets with rapid carbs probably requires pre-bolussing.
- Is the user up for an informed and mindful approach towards a highly personalized calibration of her/his system, and periodic re-checks, notably when diet habits strongly change?
- While meal management gets very easy, activity management gets a bit more difficult, especially if sports snacks shall be avoided (because of weight gain)
- Kids could be a problematic group for the currently available full closed loop if they have low hourly basal rate, instable profiles, erratic patterns of activity and of consuming snacks and sweet drinks. Unreliable tagging of sport snacks would be a problem, too. (Snacks could trigger a full „meal detected" response by the full closed loop).

----

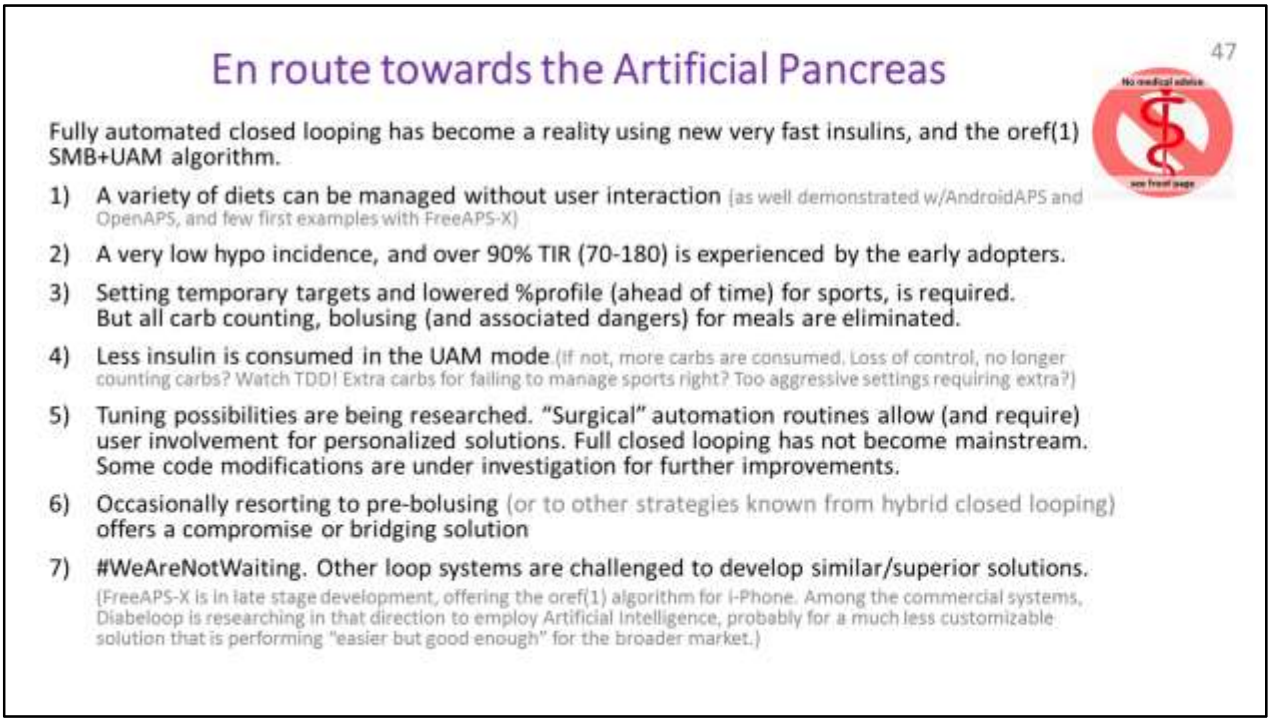
*) iOS Loop and its branches (Autobolus and FreeAPS) are not doing well enough without carb inputs:

- Looping study (Stanford Univ.) Diabetic pigs were studied for 3 weeks each using AndroidAPS or iOS Loop. Time in Range was only satisfactory with AndroidAPS (SMB und UAM)

Source: Poster bei ATTD Feb.2020, Madrid. COMPARING DIY FULL CLOSED-LOOP PERFORMANCE IN PIGS WITH STREPTOZOCIN-INDUCED DIABETES CLOSED-LOOP SYSTEM AND ALGORITHM Link aus dem Flyer: [scroll to abstract 246] / Abstract ID 474 - See also: https://www.researchgate.net/publication/351273207_Full_closed_loop_open-source_algorithm_performance_comparison_in_pigs_with_diabetes*

- See also Discussion in: *https://www.diabettech.com/oref1/lyumjev-a-fully-closed-loop-case-study-with-oref1*

- Using Autobolus or FreeAPS branch of iOS Loop, and very moderate TIR expectations looping without bolussing might be possible, too (Kenny Fox @ Loop and Learn Facebook, Nov.13/14, 2020; loopers with elderly relatives in poor diabetes control might want to explore that idea further).



To summarize on full closed loop: (read text)

Body weight control

More difficult when (hybrid or full) looping?

- Untreated type 1 dies under-weight => Optimally treated (looper!) gains weight?
- Rage bolussing? => More carbs needed afterwards.
- Pizza etc. managed better => Consumed more?
- Often around 100 mg/dl and lower => Any **activity may require additional food**

Simpler when looping?

- Less „roller coasters“ => Less carbs and less insulin
- More knowledge and data at hand to manage health & body weight. Options to integrate fitness watch and various automations. Good to sustain self-motivation.

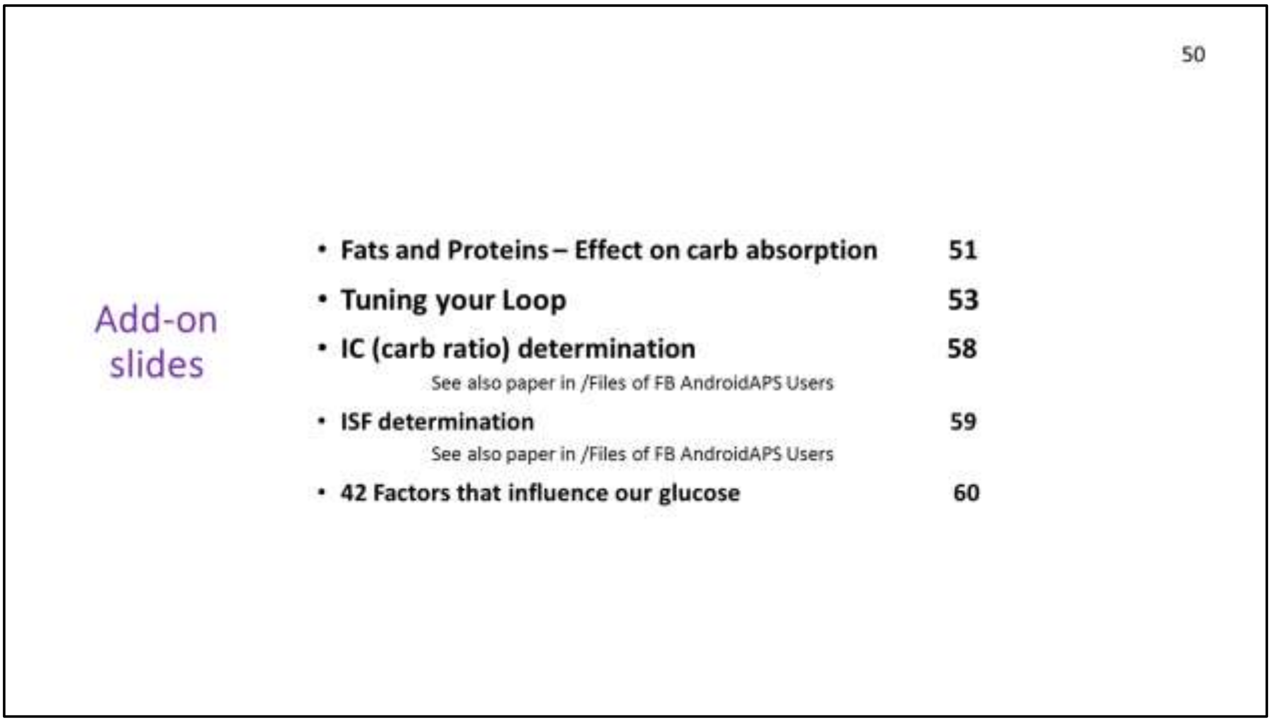
=> Reduce insulin delivery prior to activity:
Pro-actively set elevated temp. targets and lower %profile (overrides).

Note: Only DIY Loops are very flexible regarding settings for sports.
However, commercial loops generally operate with higher targets, may have an extra mode with lower insulin delivery, and, at moderate excercise, usually work out too.

We like to conclude the presentation with a word on **body weight control**.
History showed that untreated type 1 diabetics were dying underweight.
We could suspect that perfectly treated type 1s – loopers - might lean towards gaining weight.
That could, indeed, be the case:
The permanent availabilty of glucose values, but sluggish nature of control capability, very often leads loopers to interfere with the loop via rage bolussing, and, as low values appear, eating too much additional carbs., The ensuing roller coaster inheritantly increases both insulin and carb use.
Pizza and other formally challenging meals can now be enjoyed more frequently, and less often this is accompanied by „losing glucose“ via urine.
In full closed loop, we even lose any control on carbs, which are not counted any longer.
People with an active lifestyle also have the problem that they may need unplanned-for snacks very often, because their blood glucose (or predicted glucose, given their iob) rarely leaves them enough „room“ to exercise.
Still, weight gain can be averted:
Roller coasters can be avoided by careful calibration of the loop system, and with patience from the user.
Loopers have knowledge and data at hand, to manage health & body weight. (Also in full closed loop, the TDD allows indirect monitoring of carb consumption, long before body weight would show the trend.).

There are far reaching options to integrate fitness watches and various automations, which is good to sustain self-motivation.
To avoid frequent sports snacks, active people must learn to manage exercise by setting elevate glucose targets and reduced aggressiveness of factors/lowered basal rates, and do this well ahead of the activity already.
In conclusion, loopers certainly **can** successfully integrate a weight control goal into their diabetes management.
There will always be setbacks by diet sins, illness, stress, or just plain forgetfulness. Also technical system instability can in times be challenging. But it is absolutely worth it to stay motivated for a well running loop.
So, enjoy eating, remain (mostly) in range, and stay healthy!

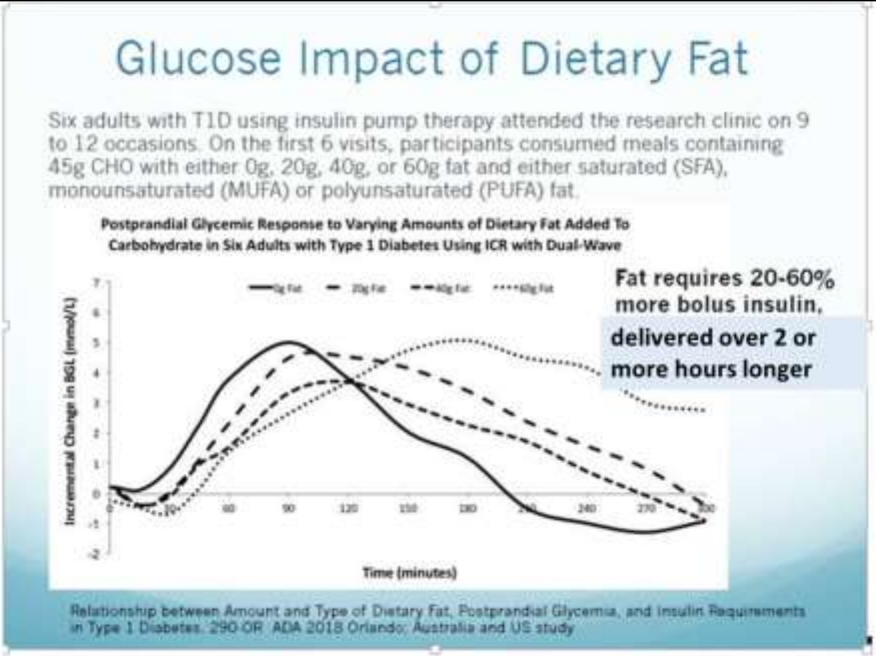
Open source loopers, please stay connected with the community, draw from the experience shared by others, and contribute where you can!
Users of commercial systems should primarily stay in contact with the respective manufacturers but are more than welcome to draw from the applicable shared experience in the DIY groups. As demonstrated by the full loop capability, by the remote small-kid management capabilities, and by the many customization possibilities, the open-source communities blaze ahead. # WeAreNotWaiting!
Stay safe. Do not just copy for yourself what works for others. Authors/cited colleagues are no medical professionals, and cannot guarantee correctness of content or applicability to other patients



In the following, a few slides are added on effect of fats and proteins, and on tuning the loop.
The very last slide is on „42 factors that influence our glucose“, maybe a proper reminder not to chase a phantom of perfect control, but rather to be happy with (maybe even amazed about) how well things are working for you.

Sorry, some of the following add-on slides have comments that were not translated into English. (But they pretty much say what the slide contains).

51



From: John Walsh, 2018: https://www.diabetesnet.com/diabetes-resources/diabetes-presentations/

Glucose Impact from Protein and Fat: Mechanism

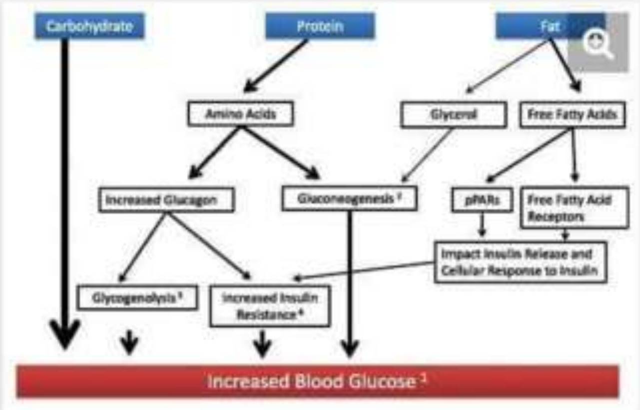
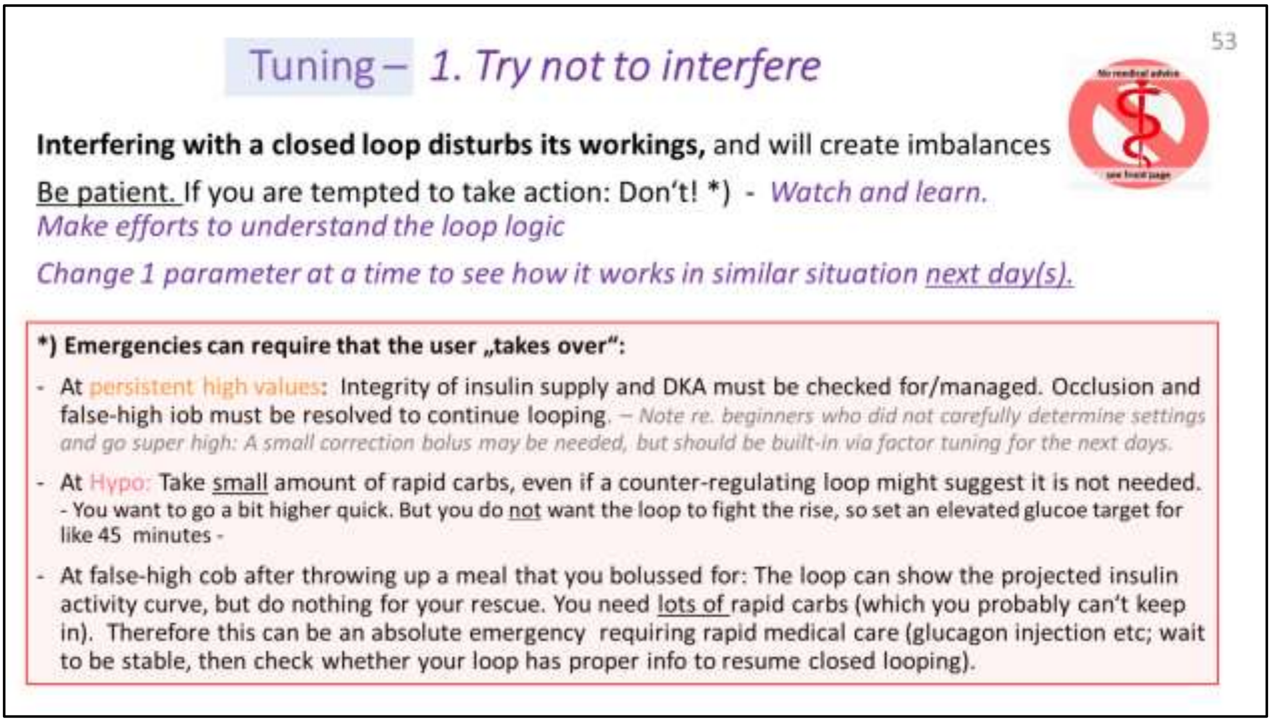


Figure 1

Download figure | Open in new tab | Download powerpoint

Mechanisms by which carbohydrate, protein, and fat contribute to increasing blood glucose levels and insulin requirements in type 1 diabetes. Numbers indicate points of insulin requirement. 1, insulin is required to metabolize glucose; 2, insulin is required to stop gluconeogenesis; 3, insulin is required to stop conversion of glycogen to glucose; 4, insulin is required to counteract insulin resistance. Width of arrows indicates relative contribution to increase in blood glucose. pPARs, peroxisome proliferator-activated receptors.



Auch wenn man sich zurecht gelegt hat, wie es gut genug funktioniert – Man wird oft Grund finden zu Unzufriedenheit mit dem aktuellen Ergebnis ..

.. und versucht sein, in die Steuerung des Closed Loop einzugreifen.

Das sollte man aber lassen. Vornehmlich kommt es darauf an zu beobachten, zu lernen, **den Loop** zu **verstehen** versuchen und **ihm** mit den Faktoren und Sicherheitsparametern das nötige **Werkzeug und „Leitplanken"** zu **geben** (Tuning).

Ich kenne nur 2 Indikationen, wo ich aktiv in den Loop eingreifen sollte *(bzw. muss, bzw. er sich evtl auch selbst abstellt oder zumindest laute Alarme aussendet):*

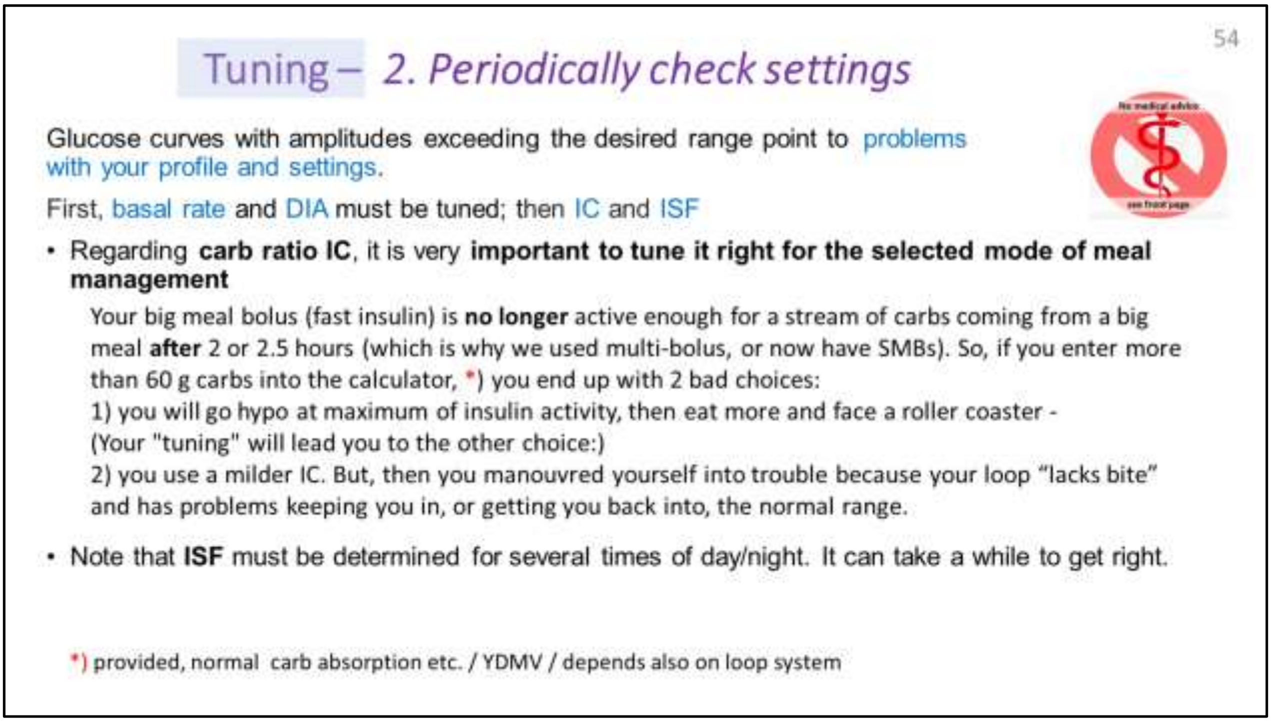
Bei Hypo kann es indiziert sein ein paar g Traubenzucker zu nehmen um schneller als der Loop dies (vielleicht) auch kann aus der roten Zone zu kommen. *(- Nicht so viel wie wir aus vor-Loop Zeiten gewohnt sind; und HypoTT setzen, damit der Loop nicht sofort wieder Insulin dagegen ballert)*

Anhaltend (über Stunden) steigende BZ-Werte, bei steigendem angezeigten iob, deuten darauf hin, dass etwas mit der Insulinzufuhr nicht stimmt *(„fake" hohes iob)* . Da hilft meist nur ein Bolus mit dem Pen, der aber gut überlegt sein will, was die Größe anbelangt. Nach der Wiederherstellung der Insulinzufuhr muss auch die Datenbasis des Loop um fake Werte bereinigt werden.

Wie kann man sich das Tuning vorstellen? …

____

Mehr zum Vorgehen bei Problemen mit Insulinzufuhr, siehe: *https://de.loopercommunity.org/uploads/short-url/lFgUkZnGkO83RmheUUxnbfNRU6C.docx*



Glukose-Kurven, die sich auch Stunden nach Störungen nicht an die Ziel-Linie anschmiegen, oder die diese in einer „Achterbahnfahrt“ täglich wiederholt kreuzen, deuten darauf hin, dass die Grundeinstellungen überprüft werden sollten. Ferner deutet ein unerwartetes Verhalten des Loops, z.B. wenn man eine Mahlzeit auslässt, oder wenn man für einige Stunden in den OpenLoop Modus geht, darauf hin, dass Einstellungen wahrscheinlich nicht stimmen.

DIA und Basalrate sind unbedingt immer zuerst zu verifizieren. Denn Fehler dort müssen sich beim weiteren Tuning in „Gegen-“Fehlern bei anderen Parametern ausbalancieren (was als labile Balance durchaus gelingen kann).

Ich habe da selbst eine leidvolle Geschichte. Aufgrund unkritischer Verwendung von Autotune hat sich meine Basalrate immer weiter vom eigentlich guten Muster entfernt. Ich konnte das aber mit andren Tuning-Maßnahmen gut genug abfangen, und merkte das Problem erst als ich (wegen Umstieg auf G6 – 8G, danke, Dexcom) den Loop halbe Tage lang abschalten musste (an Tag-1 bzw. -11 jedes 8G Sensors).

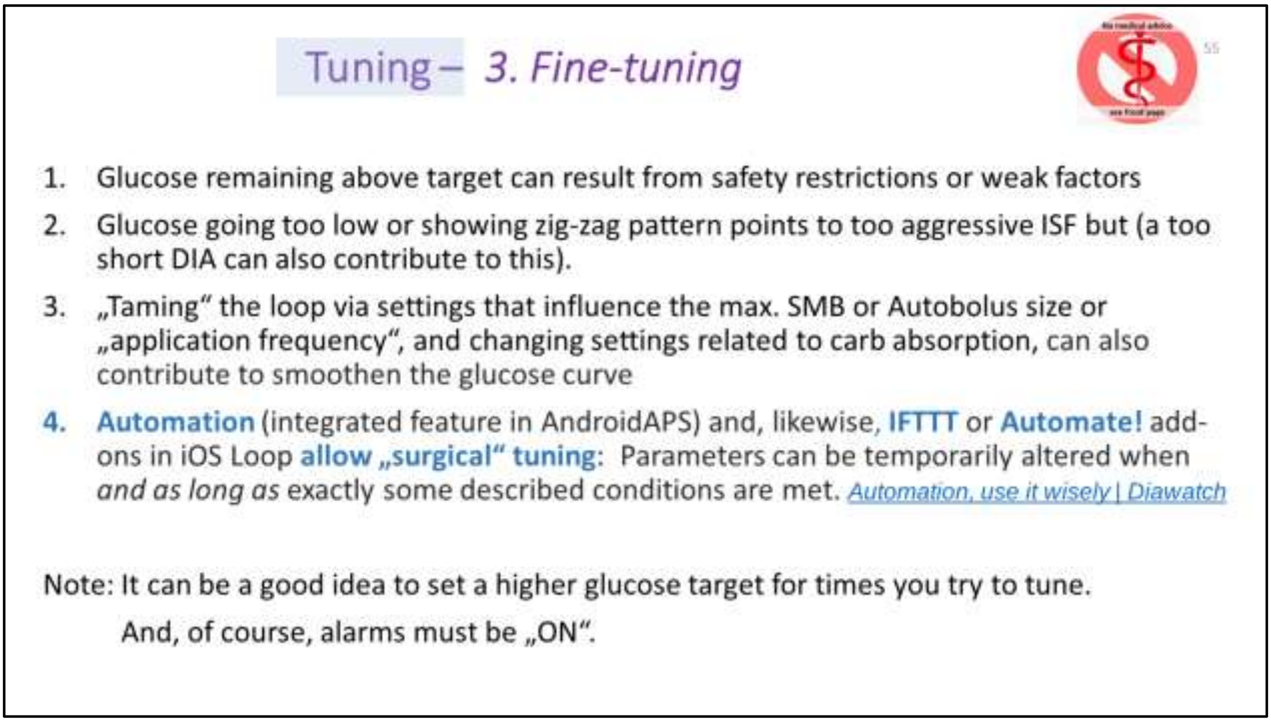
Sofern DIA und Basalrate stimmig sind, überlegt man sich, je nach Art des Problems, ob die **Carb ratio IC** oder der Sensitivitätsfaktor ISF verändert werden sollen

Der Mahlzeiten-Bolus ist nach 2,5 Stunden **nicht mehr aktiv genug** für die KH die jetzt noch zur Absorption kommen. (Deshalb nutzten wir “früher” Multi-Bolus, und haben jetzt den

Loop, der weiteres Insulin gibt). Man *) sollte nie **mehr als 60 g** KH eingeben im Bolusrechner; den man hat dann nur die Wahl zwischen zwei Übeln:
1) Beim Maximum der Insulin-Aktivität herrscht Hypo-Gefahr, man nimmt KH, und eine Achterbahnkurve entwickelt sich. - (Spätestens nach dem "Tuning" kommt man dann zur andren Option:
2) Man nutzt einen milderen (= **zu milden**) **IC**. Dadurch bekommt der Loop generell zu wenig "Biss" und Probleme, den Zielwert zu erreichen.

*) CAVEAT: normale KH-Absorption usw. vorausgesetzt / YDMV / abhängig vom gewählten Modus (vgl. S.9) – Vor allem bei proteinreicher Diät ist wichtig, wie man FPEs/late Carbs konsistent eingibt und mit dem passenden IC Wert behandelt - Evtl. auch abhängig vom gewählten Loop System (?). Loop System

Beim „tunen" des **ISF** ist zu berücksichtigen, dass er besonders empfindlich auf Tageszeit und körperlich/gesundheitlichen Zustand empfindlich reagiert. *Auch sollte man Autosense dabei eher abschalten (oder mitinterpretieren können).*



(1) Neben den Daten des Profils können r auch manche **Sicherheitseinstellungen** und weitere in AAPS/Preferences eingestellte Parameter eine entscheidende Rolle spielen.

(2) Setzen temporärer Ziele sowie Profilwechsel (in iOS Loop: overrides) sind oft sehr gute Maßnahmen zur Entschärfung von Problemfeldern

(3) Schwingende Glukose-Kurven lassen sich manchmal *(neben ISF auch)* dämpfen durch Änderungen bei DIA des verwendeten Insulins, SMBs lassen sich "zähmen" (limit SMB min basal, aber auch Einstellungen, wann SMBs überhaupt eingeschaltet sein sollen), und die obere und untere Default-Grenze für KH-Absorption kann man ebenfalls variieren.
beim maximal „erlaubten" basal iob, oder bei den Limits für Autosense.

(4) Die grundsätzliche Möglichkeit in AndroidAPS mit „Automation" wurde ja schon skizziert *(Folie 12 unten)* und es folgt gleich noch ein Beispiel *(Folie 22).* Daneben gibt es – nutzbar auch z.B. für iOS Loop - Automations-Möglichkeiten mit Hilfe von speziellen Programmen aus dem Google Playstore wie IFTTT oder Automate!

Auf die vielen Optionen zum Tunen können wir im Einzelnen jetzt nicht näher eingehen. Aber man kann ggf. den Suchbegriff in der Wiki oder in

LooperCommunity.org eingeben, und konkret nachfragen …

Immer mehr Looper entdecken die Automation für sich, und, mit Bezug zu unserem Hauptthema, sehen da einen Lösungsweg für die Bekämpfung der gfett-induziert hohen Glukosewerte nach manchen Mahlzeiten: ….

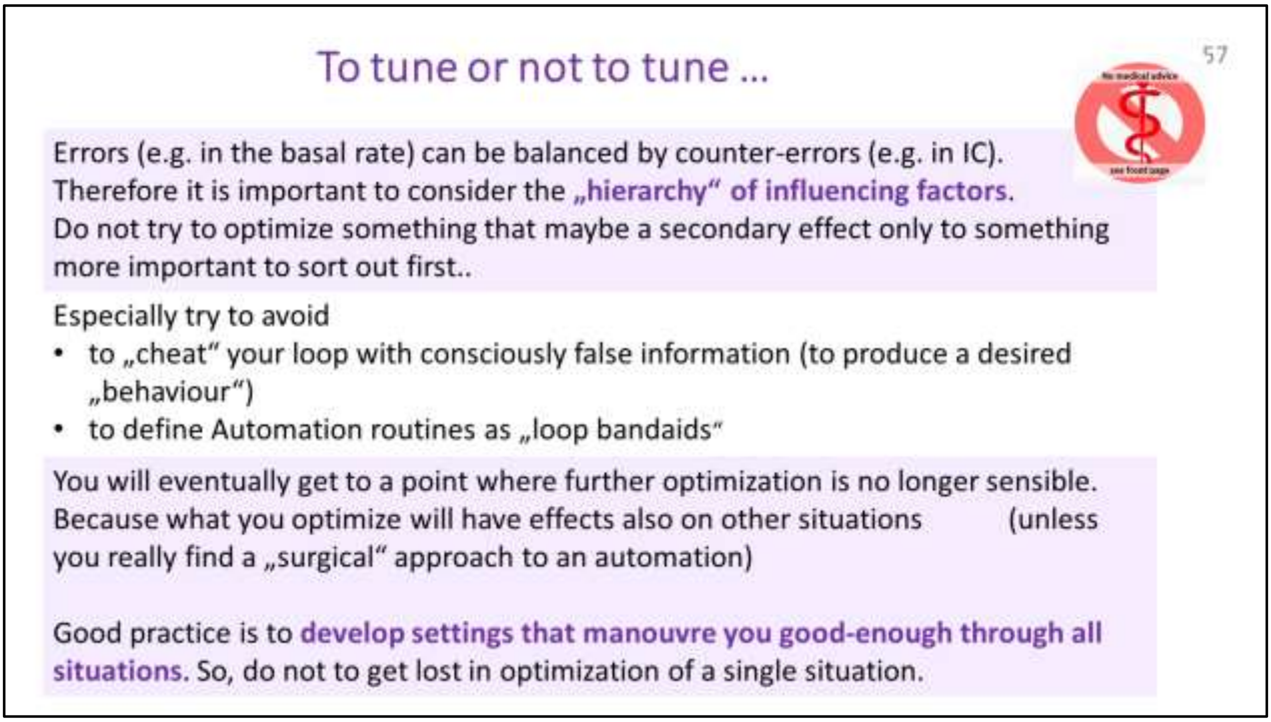
Tuning – 4. Automatic „self-"tuning capabilities

56

Some loop algorithms have capabilities to analyze data from the very recent past (minutes, hours or weeks, see last column!) to (quickly, or very slowly) adjust certain parameters:

Parameter	Loop	integr. or extra	Feature name	Remark	Time scope
Basal rate, IC, ISF	Open APS, AndroidAPS	<- integr. <- external	Autotune	IC, ISF: only 24h avg.	>24 h. >2 w.
Basal rate, IC, ISF	Open APS, AndroidAPS	<- integr. <- integr.	Autosense	detected insulin resistance from last ~8 h applied to the parameters	~ 8 h. ~ 8 h.
Basal rate, IC, ISF	iOS Loop				
IC, ISF	Diabeloop	Integr.	Artificial Intellig.		>2 w.
ISF	AndroidAPS	<- integr. (in dev.)	AutoISF	rapid modulation depending on glucose curve, settings	>15 m.
Carb absorption	iOS Loop	<- extra		connectable food databases	real-t.
Carb absorption	OpenAPS, AndroidAPS Free-APS-X	eliminated need to tune	oref(1) SMB+UAM (un-announced meals)		n/a n/a n/a

=> => Needs work, notably for iOS Loop <= <=



So, jetzt wird‘s poetisch: To tune or not to tune, that is the question.
Es ist eine gute Idee seinen Loop zu pflegen mit wohlüberlegtem Tuning. Aber dabei sollte man nicht aus der Hüfte schießen.

Da Fehler (z.B. in der Basalrate) mit Gegenfehlern (z.B. beim IC Wert) ausbalanciert sein können ist es wichtig immer an die **„Hierarchie“ der potentiellen Einflussfaktoren** zu denken – und nicht an einer kleinen Stellschraube zu drehen, während eigentlich „vorgelagert“ erst mal woanders das Tunen erforderlich wäre.

Vorsichtig sollte man mit Automation umgehen…damit es nicht ein grobschlächtiges „Loop-Pflaster“ wird für Probleme, die eigentlich innerhalb des „Loop“ gelöst werden könnten (..oder **die** auch **nicht lösbar sind**, und wenn man sie trotzdem „lösen“ möchte (a la Wut-Bolus) eben **zu andren Problemen führen und den Loop** an anderer Stelle **überfordern**).

Es ist nur bedingt sinnvoll Dinge, die, gut genug laufen, weiter zu optimieren. Denn was man für diese eine Situation ändert, wirkt sich auf ähnliche, aber eben doch i.d.R. andre, Situationen ebenfalls aus (es sei denn, ich bekomme das „chirurgisch in den Griff“ mit Automation) – **Die Kunst ist, ALLE vorkommenden Situationen gut genug zu durchlaufen**, nicht, sich verrückt zu machen an der Optimierung konkreter Einzelfälle.

IC factor (carb ratio)

58

You can **ESTIMATE** your **daily average IC** if you (1) count up the g carbs in 24 hrs and (2) divide it by the amount of 24 hr bolus insulin. For the latter, look up the **TDD** (total daily insulin given), then substract the "real" 24hr basal need <u>as in your profile</u>: **IC (g/U) = (daily g carb) / (TDD - 24h Basal as in profile).**

Adult example: TDD = 37U; Profile Basal = 16U; daily carbs 200g => IC = 200 g / (37U-16U) ~ 9.5 g/U

Average data from days without extreme sports, unusual stress, or infection from that evaluation. Later you will modify insulin delivery for such scenarios (via %profile switch, override, aggressivness).

DETERMINE, for each relevant meal time, your IC on days without preceding major activity, stress, infection, and relative steady glucose in the normal range (and cob=0): Shut closed loop off. Eat a well defined **smaller meal** (20 .. 45 g of preferably „rapid" carbs) and divide by your estimated IC to determine the amount of **insulin** for this meal. *Example: 38g / 9.5 g/U = 4.0 U*

With ClosedLoop off, but profile basal running, watch for 3 hours (Lyumjev or Fiasp).

If your glucose levels about where you started, the IC can be used.

If curve goes too low (eat some carbs and) try again next day with a higher IC value.

If curve remains too high, the IC was too weak and needs to be lowered.

TUNE your IC in hybrid closed loop so glucose is in low part of desired range after maximum activity from your bolus

More see: https://www.facebook.com/groups/AndroidAPSUsers/permalink/3092045561016840/

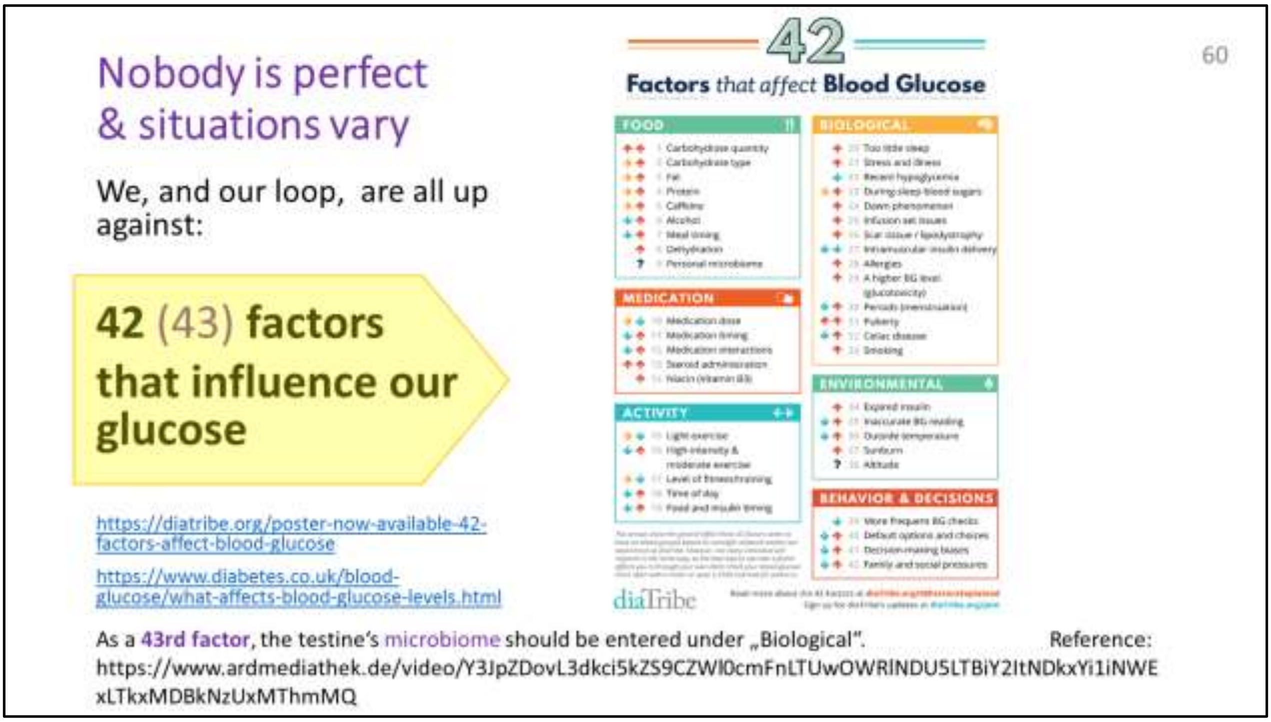
ISF determination

You can **ESTIMATE** your **daily average ISF** using: *ISF*=1700 / *TDD*, where *TDD* is the total daily insulin.
Example TDD = 40 U; ISF = 1700 / 40 = 42 mg/dl/U (or divided by 18 for mmol).

DETERMINE, for several times during day and night, your ISF: Do a determination of your **CarbRiseRatio CRR** first: Take a sweet drink with a known carb content, and watch (in Open Loop, not giving any insulin except profile basal rate) by how many (mg/dl) glucose rises, until reaching a plateau (in about 1 hour).
Example: After consuming 20g, my glucose rises from 90 to 190 mg/dl. CRR = (190-90)mg/dl / 20g = 5 mg/dl / g
Now, having reached a high plateau – *at 190 mg/dl in our example* ,-we move on to determine **the ISF**.
Inject (still in Open Loop, with profile basal running), an amount of **insulin** as you typically would (considering your personal insulin sensitivity). *Example: At 42 mg/dl, 2 units should bring glucose down by 84 mg/dl.*
Then watch for about three hours until a deeper plateau is reached, *for example at 110 mg/dl.* Then you can evaluate: *Example: 2 U of insulin lowered my glucose by 80 mg/dl . Hence, ISF = 40 mg/dl / U*
Note: As IC = ISF / CRR you can also calculate IC, *in the example ISF / CRR = 40 (mg/dl)/U / 5 (mg/dl)/g = 8 g/U*

TUNE your ISF in hybrid closed loop so corrections of high values after meals, and notably also in the night, bring glucose to target. Not reaching target is a sign for too big (mild) ISF, rollercoasters speak for too low (aggressive) values.
Special **boosting of ISF** potency for a couple of minutes when, and as long as, glucose is high after a fatty meal can be helpful, and can be automated in AndroidAPS, or via using add-ons like IFTTT also in other DIY loops.

More see: https://www.facebook.com/groups/AndroidAPSUsers/permalink/3091340191087377/



The very last slide is on „42 factors that influence our glucose", maybe a proper reminder not to chase a phantom of perfect control, but rather to be happy with (maybe even amazed about) how well things are working for you with your loop.