11. Emulator on your AAPS smartphone V 2.9

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

Available related case studies:

Case study 11.1: none available yet

*Skip what is in **green writing***:
= Drafted fragments or not implemented ideas. Please contribute, or wait for update with the missing info

The emulator **on the PC** was presented in section 10 as

- a very good tool for making your **initial** tuning for a meal spectrum (= when weighing different effects over the entire course of time after each meal, and for a variety of your meals).

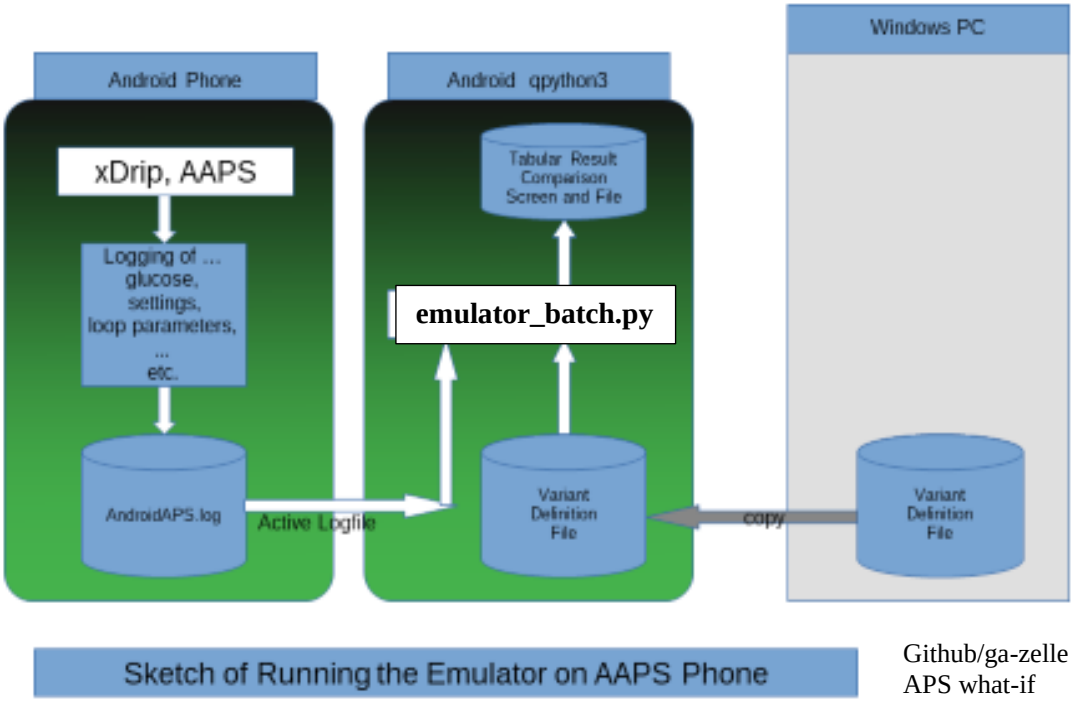
A very useful *additional* tool, is the emulator running **on the AAPS loop smartphone**

- for a quick look how autoISF triggers SMBs after starting a meal (section 11.2).
- and especially for real-time checking „what-if“ you implemented a specific change idea (developed on your PC, or after analyzing many SMB tabs) (section 11.4).

Regarding **i-Phone** options, see section 11.3

The emulator for the AAPS phone is described in https://github.com/ga-zelle/APS-what-if

Join https://discord.gg/n3tD5eXExC for seeking (and giving) help with the Emulator set-up or use, and to share experience.



Sketch of Running the Emulator on AAPS Phone

Github/ga-zelle / APS what-if

11.1 Installing the emulator on your Android smartphone

Note that iOS based autoISF variants cannot use the Emulator on an i-Phone. However, some tabular outputs of ISF-factors are available (see section 11.3 to be provided by iAPS user)

11.1.1 Installing QPython 3L

On your smartphone, go to Google Playstore and download the QPython 3L app. Put the app icon next to your other looping related app icons on the main screen of your smartphone.

With many QPython 3L versions, phone and Android OS versions etc around, you might run into problems and may need to consult detailed installation instructions from https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1/Documentation%20in%20English/Installation%20Guide.pdf, or seek advice via https://discord.gg/n3tD5eXExC

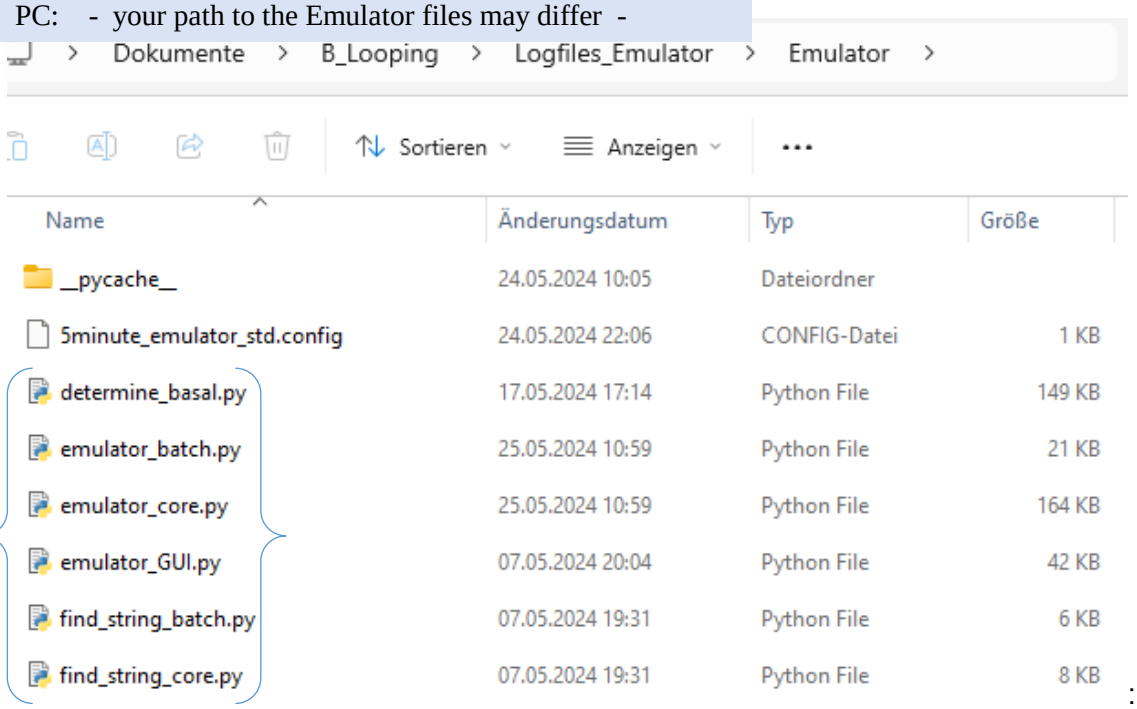
By long pressing on the QPython 3L app icon, go into “app info” and make the settings like for all your other looping related apps, so they do not get killed all the time by energy savings routines.

11.1.2 Copy .py files from your PC into your phone’s QPython/skript3 folder:

1). Connect your phone to your PC for USB data transfer

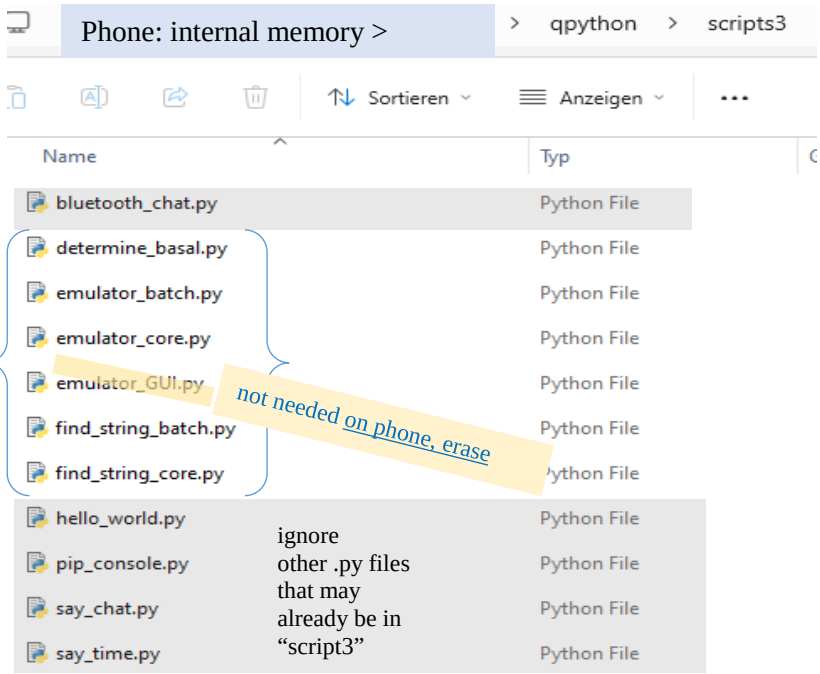
2).Locate your .py files on your **PC** (in Emulator file).

PC: - your path to the Emulator files may differ -



:

3).**Copy** all Python related files except emulator:GUI,py from your PC over the internal memory / QPython / **Scripts3** of your phone:



11.1.3 Put configuration file and noChange.vdf into the phone's AAPS logfiles folder

1). With your phone connected to your PC for data transfer, retrieve **5minute_emulator.std.config** (or 1minute_emulator.std.config if you use Libre3/1minute) on your PC, (picture, left)

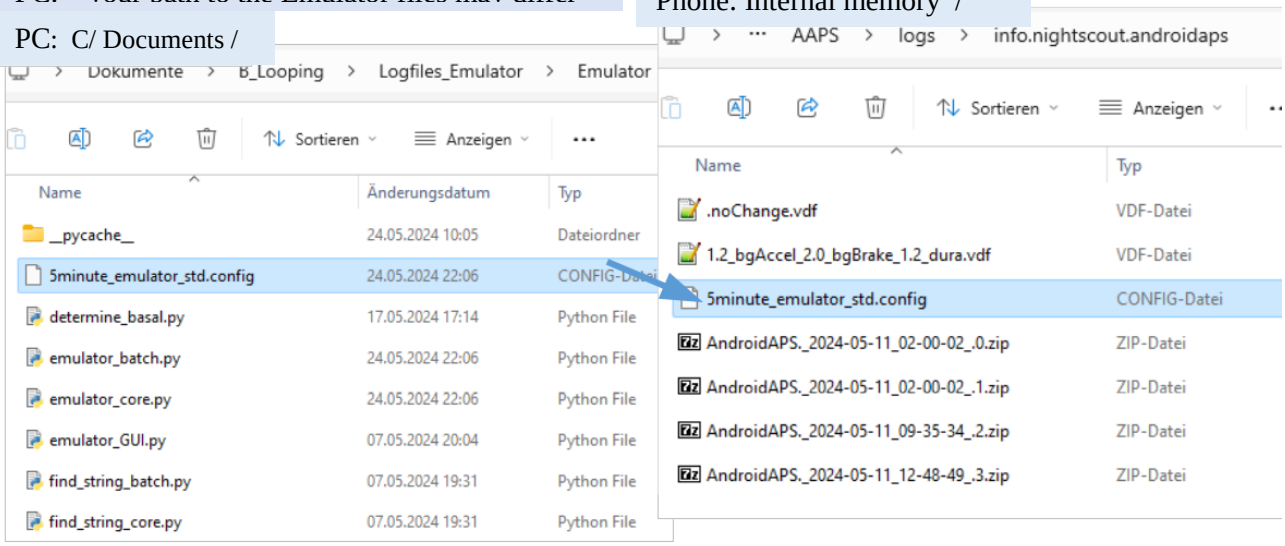
2). This config file contains **your "STANDARD"** programming as to which hours of day there can be speech messages. Default 07-23 h ("your" time, not "Z"scale). How to change: see in section 11.4.3

3).Put a **copy** into the phone into the logfiles (not the QPython!) folder (picture, right):

- Internal memory/AAPS/logs/info.nightscout.androidaps

PC: your path to the Emulator files may differ -

PC: C/ Documents /



Phone: Internal memory /

4). You have the option to produce more than just your „…std.config" file.
For instance you could additionally define and load one, that remains silent at carb-related messages, and gives you only insulin (SMB size) reklated "what-if" suggestions via speech synthesis ; name it for instance „5m_noCarbsAnnounced.config" .
How to switch between the .config files in a run, see section 11.2.2 step 6)

11.1.4 Put noChange.vdf into the phone's AAPS logfiles folder

Retrieve the **noChange.vdf**. on PC in the parallel Studies file

- C:\....\Documents\ Looping\ Logfiles_Emulator\ Emulator_Studies

...and put it on the phone, also into the folder (picture, right):

- Internal memory/AAPS/logs/info.nightscout.androidaps

Later, in section 11.4.1, you will add also (yourChange).vdf files into the same folder. One is already included, line under noChange.vdf, in picture above, right.

11.1.5 Customization of output table

The table should contain the most relevant information that can be displayed reasonably on most smartphone screens
Consult https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1/Documentation%20in%20English/How-to-run-the-emulator-on-the-phone.pdf in case you see a need to customize .config files and output tables for you.

11.2 Inspecting loop decisions on your smartphone

11.2.1 Principal purpose

The result table from the emulator on your smartphone allows you much easier insight than the SMB tab can offer into current and recent determinants of given SMB sizes (e.g. which of the 4 autoISF categories contributed).
So, if you would have benefitted from a bigger (or smaller) SMB at times where, say, bgAccel_SF was the dominant factor, you would increase (or decrease) the associated weight. Before actually making such changes, look in other lines of the table to estimate how this would influence decisions in other time points.

> Testing your hypothesis on the PC (section 10.3) would allow multi-day multi-(kind-of-)-meal judgement on feasibility of your tuning idea.
>
> Before firmly deciding on a settings change, it is advised to run on your phone a what-if emulation (section 11.3) using the less aggressive settings for your active loop run, and the more aggressive defined in (yourChanges).vdf.

11.2.2 Generating a results table (for last 75 minutes' loop decisions)

Note: 1 minute **Libre3 users** will get data only on a significantly shorter time span.
15 minutes is really too short to analyze what is going on, and Libre3 users probably should mostly use the "what-if" part, see section 11.4. That part is principally not impacted (except, cutting one 5 minute change into 5 very tiny changes, often will make it hard to see and interpret "what-if" effects).

Display setting on your phone should be set for automatic switch between landscape and upright viewing (depending how you hold your phone).
Reducing selected font size will *not* help to get more table info on your phone screen, or to avoid broken/double lines. Go to 11.1.4 if, in the following, you are not happy with lay-out.

1).On your main phone screen, press the QPython 3L app icon:

The first black screen then popping up asks to make a **language** selection

In case you don't get an alpha-numerical input field (with <- enter button), touch the upcoming black screen

All black screens have a keypad at the bottom:

2).Press your selected number and <- (for enter).

3).Then proceed in the same screen
to make your capital-"N" entry
and finally " <- " for Next (see illustration)

In the following the instructions from the developer how to navigate through the screens:

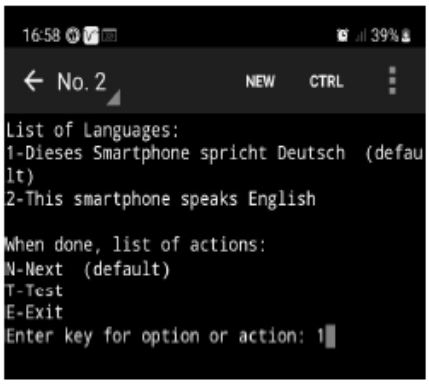
After some Android12 update the previously used GUI dialogues no longer worked and I had to create a keyboard based version similar to the typical telephone dialogues “for option X dial 2”. This dialog system consists of two parts:

- The top part has numerical keys for each option that can be selected. One of those options has **(default)** at the end of the line which indicates that this option is the current selection. If you enter a different number the dialog screen is redrawn and that indicator moved to your new selection. Once your intended selection is OK you focus on the …
- bottom part, which has letters as keys for the action to be selected. Again, the **(default)** indicator highlights what would be done next if you just press enter without any digit or any letter. Those actions typically are Next, Test and Exit.

The first dialogue is used to select the **language** for the speech synthesis.

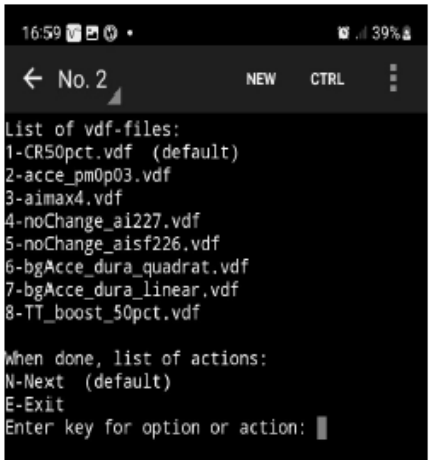
Select „Test“ to listen to a sample speech synthesis.

Select „Next“ to proceed.



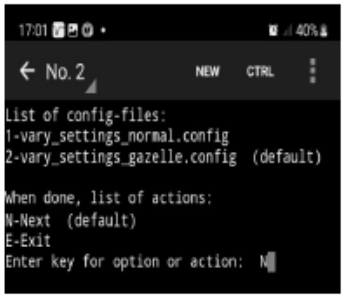
4-5). Repeat steps 2) and 3) with the next screen:

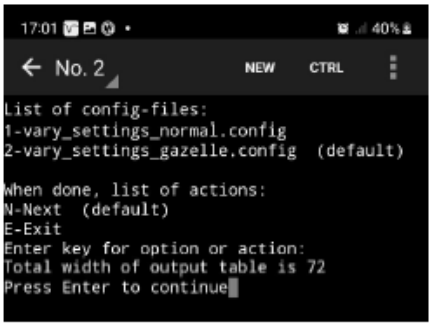
The next dialogue is for selecting your **variant definition file**. All vdf-files found in the logfile folder will be listed.



6-7). Repeat steps 2) and 3) with the next screen:

The last dialogue is used to select your favourite **configuration file** with the content discussed in the preceeding section. After you activate the selection you get an informational message of how many columns the selected tabular output will occupy.





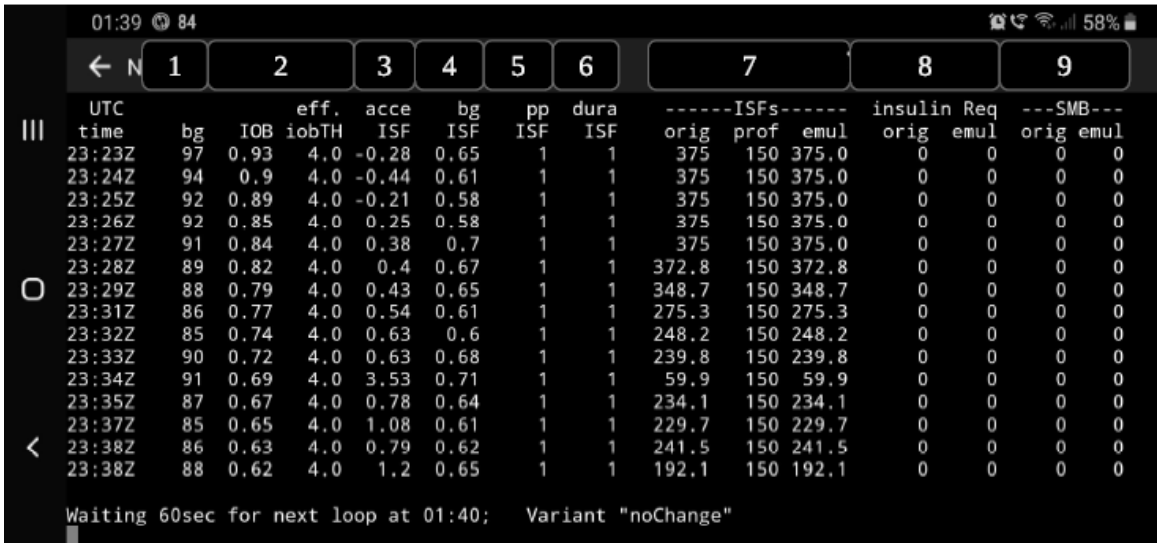
8) Before activating the”**N**(ext)” selection now is a convenient time to rotate the phone to landscape to prepare for the multi column result table display.

Note: Above in step 6), you can also switch between different …config files, e.g. to silence less important outputs. See also step 4) in section 11.1.3,

9).Now a table comes up detailing the loop decisions on SMB size for the last (15 * 5 =) 75 minutes.

In case you forgot to do it in step 4), turn the phone now 90 degrees for landscape format; in this case, give it 5 minutes for the format to straighten out (after a new value has arrived)

In case you want other, or less, columns, see section 11.1.4.



The line at the bottom of the table says the time (hh:mm) when the next bg result and loop decision are expected. Also it shows the (yourChange).vdf file investigated

In sequence of the time(Z!), the table consists default of the columns with info on (1) bg (2) IOB and eff.iobTH (3-6) the adaptation factors on ISF suggested by the 4 autoISF categories (7) resulting ISF that was used, profile_ISF, and emul (“what-if” ISF) (8) insulinRequired calculated by your running loop, and “what-if” result (9) same for resulting SMB

The columns marked “**orig**”.in the table of results shows the ISF actually used to determine insulinReq and SMB size in the actual run.

In the columns marked “**emul**” you find the calculated emulation results that is calculated every 5 minutes

- If a (yourChange).vdf was clicked (in step 2), see also bottom right of the table), the emul columns show the result, how the investigated changes *would have* changed SMB sizes (strictly always looking at the one decision, in that line of the table).
- If only a noChange.vdf was run, the emul columns contain the same results as the orig. columns.

Caution when interpreting the values in the acce, bg, pp and dura ISF columns (3-6)

The factors given there are always those for the emul run.

- So when using (yourChange).vdf, you see in your phone’s table exclusively the ISFs that *would result-if* (yourChange) were made.

We get to “what-if” projects later, in section 11.4.

- Only in the noChange scenario, the values there would be the *orig ones, corresponding to what could be seen also in the SMB tab* at the times

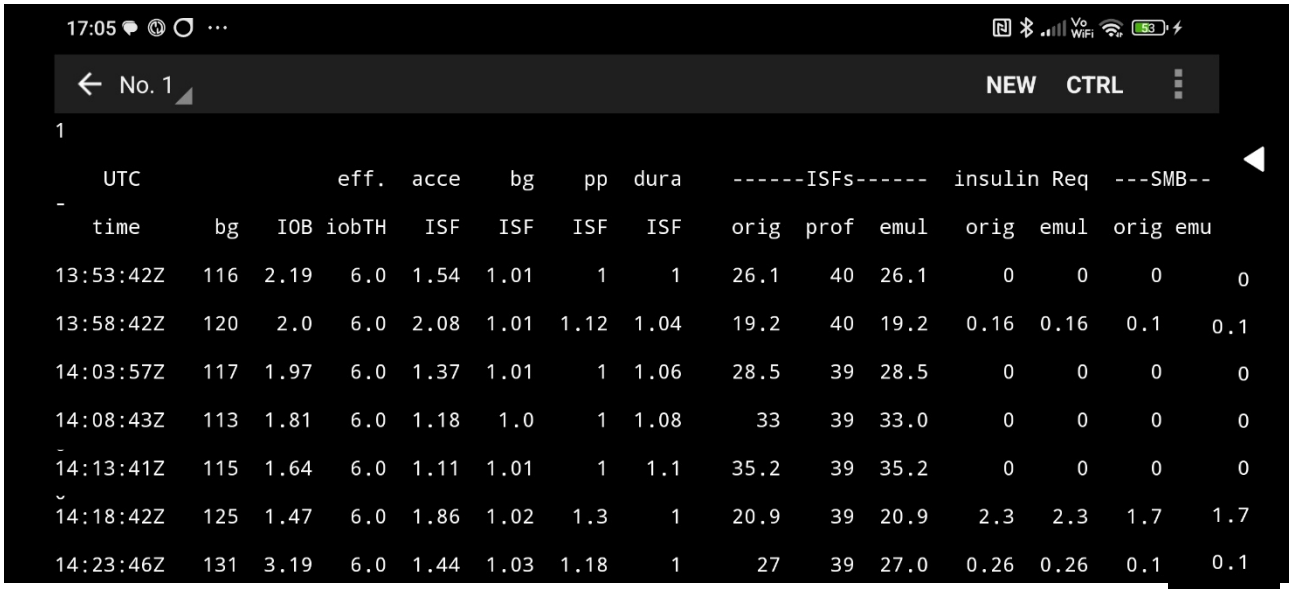
  You easy recognize whether you run the noChange: it says so at the bottom of the table.

orig.ISF is called **sens** in the SMB tab, a couple of lines below “end autoISF”. It is the ISF that replaces, for that one decision you are looking at, the profile ISF (called profile.sens in the SMB tab),

We get back to this topic at end of section 11.4.2 “Understanding how the ISF is emulated by (yourChange) and how SMB or TBR would differ”

“

11.2.3 Analyzing the results



UTC			eff.	acce	bg	pp	dura	------ISFs------			insulin Req		---SMB--	
time	bg	IOB	iobTH	ISF	ISF	ISF	ISF	orig	prof	emul	orig	emul	orig	emu
13:53:42Z	116	2.19	6.0	1.54	1.01	1	1	26.1	40	26.1	0	0	0	0
13:58:42Z	120	2.0	6.0	2.08	1.01	1.12	1.04	19.2	40	19.2	0.16	0.16	0.1	0.1
14:03:57Z	117	1.97	6.0	1.37	1.01	1	1.06	28.5	39	28.5	0	0	0	0
14:08:43Z	113	1.81	6.0	1.18	1.0	1	1.08	33	39	33.0	0	0	0	0
14:13:41Z	115	1.64	6.0	1.11	1.01	1	1.1	35.2	39	35.2	0	0	0	0
14:18:42Z	125	1.47	6.0	1.86	1.02	1.3	1	20.9	39	20.9	2.3	2.3	1.7	1.7
14:23:46Z	131	3.19	6.0	1.44	1.03	1.18	1	27	39	27.0	0.26	0.26	0.1	0.1

In above example (table), all given SMBs were driven by bgAccel_ISF, when glucose rose. The biggest SMB in the time we are looking at (actual local time = Z + 2 hours, so we are looking at late small rises, like 3 hours after lunch) was 1.7 U = 0.74 SMB delivery rate * 2.3 U insulinRequ.

The insulinRequ. Is proportionally amplified by the effective ISF, called sens in your SMB tab, or ISF“orig” in this table. The amplification of 39 profile_ISF / 20.9 effective_ISF = 1.86 happens to be the dominating bgAccel_ISF amplification factor

Note **not** always just the biggest ISF factor “wins”. Consult flowchart in LINK, and occasionally read the real-time explanations in your SMB tab as to which other factors are contributing to the amplification resulting from profile_ISF to effective_ISF (“orig”, or sens). For instance, the Activity Monitor, or a set %temp. profile, or TT, could contribute, or also the question whether glucose already decreases.

Depending on your selected “safety” settings, you might occasionally bump into restrictions. **Tuning up factors that make the system bounce into restrictions is a completely useless, and potentially even dangerous, exercise!**

Therefore, **for your initial set-up** of parameters (section 2 and section 4 of the FCL e-book) it is **advisable to** not **do this analysis** on the smartphone, but **on the PC**, where you can inspect the complete info on each loop decision (see section 10)..

> If for some reason you cannot do this on the PC, you must frequently take screenshots (in very many decisive 5 minute segments), and analyze **more complete data**, than the table on our smartphone could offer, **in the SMB tab**)

In your initial set-up of your FCL, you probably must “loop” a couple of times back into section 2 to adjust the safety settings made for max. SMB sizes.

11.3 Options available in Trio or iAPS

iAPS / Trio offer in their autoISF variants also access to a tabular representation of **autoISF contributions to** resulting **SMB sizes:**

The emulator will not work in the iOS world.

Note that on the iPhone, so far, the what-if emulation and speech synthesis (see next section 11.4 for AAPS) are currently **not** possible. .

More description must be provided by iAPS / Trio user please

11.4 Real-time checking a „what-if“ question using speech synthesis

The emulator on your **Android** smartphone can help clarify "what if..." you implemented a considered change.

11.4.1 Put a (yourChange).vdf into the phone’s AAPS logfiles folder

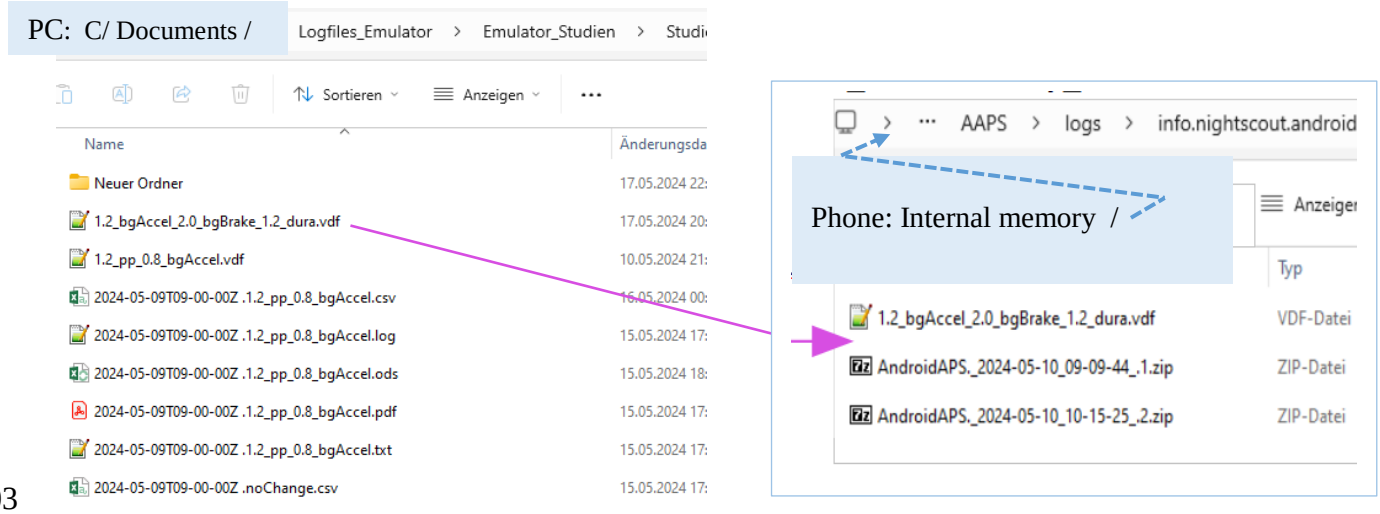
In running the emulator on the phone, you can define in the .vdf file of the emulator, which setting(s) you would like to be differently aggressive than in the active AAPS.

> How to write .vdf files was already explained in the section “Emulator on PC”. See in end of section 10.2.1. You could also pull a vdf file example from the developer’s Github that you could customize further: Access see section 3.8 /5).

Produce or retrieve **(yourChange).vdf**. on PC from one of your studies files.

- C:\....\Documents\ Looping\ Logfiles_Emulator\ Emulator_Studies\ Study_n

PC: your path to the Emulator files may differ -



… and put a **copy** of that **.vdf** into the **smartphone folder with the AAPS logs**: :

- Phone: Internal memory/AAPS/logs/info.nightscout.androidaps

Switch between python scripts running at the same time

In case you have more than one (yourChange).vdf to investigate, you look at the data of your currently running loop (last 15*5 minutes) by just switching between the related vdf files used for emulation.

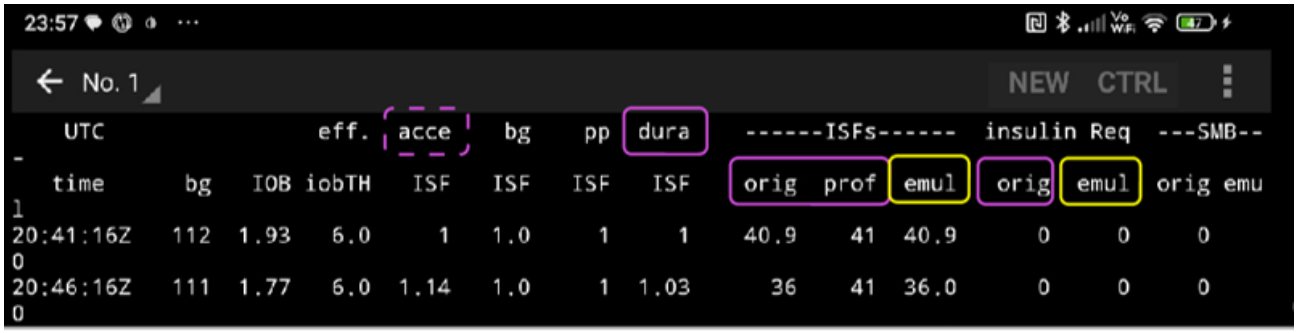
Details see section 3.8, 9) – or try to go direct via https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1/Documentation%20in%20English/How-to-run-the-emulator-on-the-phone.pdf and there p.5, under above sub-headline ".Stop the emulator, or switch…"

Stop the emulator

(see in paper as above)

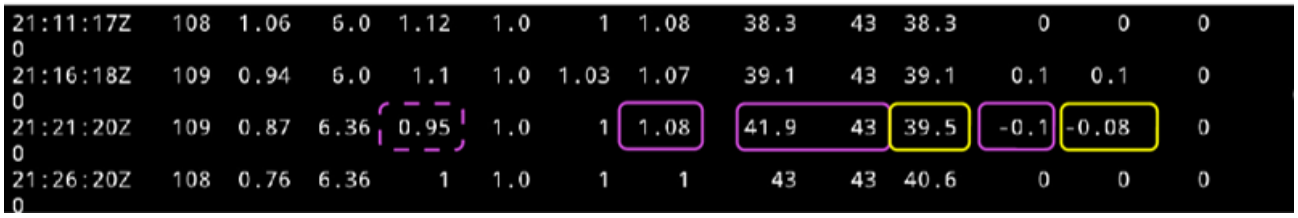
11.4.2 Inspect emulated results

Now, whenever you run QPython 3L emulation (following the steps as described in section 11.2.2) you additionally get the emul. results filled in in the resulting table, showing in which time points your change would lead to adapted insulinRequ, and SMB size data.



UTC time	bg	IOB	iobTH	eff. ISF	acce ISF	bg ISF	pp ISF	dura ISF	ISFs orig	ISFs prof	ISFs emul	insulin Req orig	insulin Req emul	SMB orig	SMB emu
20:41:16Z	112	1.93	6.0	1	1.0	1	1	40.9	41	40.9	0	0	0	0	
20:46:16Z	111	1.77	6.0	1.14	1.0	1	1.03	36	41	36.0	0	0	0	0	

etc.



21:11:17Z	108	1.06	6.0	1.12	1.0	1	1.08	38.3	43	38.3	0	0	0		
21:16:18Z	109	0.94	6.0	1.1	1.0	1.03	1.07	39.1	43	39.1	0.1	0.1	0		
21:21:20Z	109	0.87	6.36	0.95	1.0	1	1.08	41.9	43	39.5	-0.1	-0.08	0		
21:26:20Z	108	0.76	6.36	1	1.0	1	1	43	43	40.6	0	0	0		

etc

In the marked 21;21 Z line, dura_ISF was the dominant factor. The (yourChange)vdf would apply a factor of 1.2 and lead to further strengthening the ISF: profile 43 -> orig.(noChange) 41.9 -> emul (yourChange) 39.5

In this case, late after a meal, and bg=109 mg/dl, the loop saw in the orig. (noChange) case 0.1 U insulin too much; and as the (yourChange) emul case asks for typically more insulin (all weigts in the (yourChange).vdf are > 1), now only 0.08 U are seen as too much (a 20% difference).

The table on your phone is too reduced to show each emulated ISF component. If you need to see more details on how (yourChange).vdf would make a change in a point of time that you like to analyze deeper: Note down interesting Z times, and later look it up in more detail in the Emulator on the PC:

below are “orig.”ISFs:

UTC			eff.	acce	bg	pp	dura	------ISFs------			insulin Req		---SMB--	
time	bg	IOB	iobTH	ISF	ISF	ISF	ISF	orig	prof	emul	orig	emul	orig	emu
21:21:20Z	109	0.87	6.36	0.95	1.0	1	1.08	41.9	43	39.5	-0.1	-0.08	0	

Understanding ISF orig.(the sensitivity used to determine insulinRequ)

To fully understand how acce, bg, pp and dura_ISF determined the orig.ISF, we do an **emulator** run **on the PC,** and look up the **.txt** result:

@ 21:21 Z / Script Debug ----------

Activity monitor disabled inactivity detection: sleeping hours; Autosens ratio: 1;

Basal unchanged: 0.45; ISF unchanged: 43 CR: 9

---

start autoISF 3.0.1

Loop allows APS power level; SMB enabled due to enableSMB_always

acce_ISF adaptation is 0.95

bg_ISF adaptation is 1

pp_ISF adaptation is 1

dura_ISF adaptation is 1.08 because ISF 43 did not do it for 30 m

strongest autoISF factor 1.08 weakened to 1.03 as bg decelerates already

final ISF factor is 1.03

---

end autoISF

---

profile.sens: 43 sens: 41.9 Note: 43 / 1.026 = 41.9

Understanding how the ISF is emulated by (yourChange) and how SMB or TBR would differ

While the result for ISF_emul (and for the SMB size) is given in the table on your phone, the details behind it, and also for finer effects in %TBR, can only be inspected from the .csv table from the (yourChange),vdf **emulator run** on the **PC**:

The following table is an extraction of the most relevant data from the (yourChange).csv:

	B	C	E	F	L	Q	R	S	AA	AB	AC	AD	AE	AF	AG	AH	AI	AP	AQ
1			bg	bg		final	dura		acce	bg	pp	delta	dura	final					
2	UTC	AAPS	accel	brake		ISF	min-	dura	ISF	ISF	ISF	ISF	ISF	ISF	ISF	ISF	ISF	TBR	TBR
3	time	time			iob	orig	utes	avg.	emul	emul	emul	emul	emul	emul	orig	prof	emul	orig	emul
29	21:16:18	23:16	109		0,94	1,1	25	108,2	1,12	1	1,03	1	1,08	1,12	39,1	43	38,4	0,65	0,65
30	21:21:20	23:21	109	109	0,87	1,03	30	108,3	0,9	1	1	1	1,1	1,04	41,9	43	41,2	0	0,46
31	21:26:20	23:26	108		0,76	1	5	108,5	1	1	1	1	1	1	43	43	40,6	0	0
32	C=B+n/24 @ n=+2 h time diff.																		

If there were bigger orig->emul effects, we would show SMB columns, too

It shows that the final (noChange) ISF factor of 1.03 (box Q30) becomes 1.04 (box AF30) in the (yourChange) emulated case. Given that, so late after dinner (box C30), we sit near the 90 mg/dl target (box E30), the difference the changes in the emulated case are minor. Still zero SMB, but the minor changes reflect in elevating the temp. basal rate for fine adjustment (box AP30 -> AQ30).

Note that the orig. values for the four autoISF components (acce, bg, pp, dura “ISF orig”) are **not** contained *in this* csv table (only, in Q30, the final amplification factor for final_ISF in the noChange scenario).

However, you can fetch the autoISF details for 21:20 Z in the actual “noChange” run:

- from the txt result file (as shown above); or
- you also see them on the phone:

orig	orig	orig	orig	ISF
acce	bg	pp	dura	------
ISF	ISF	ISF	ISF	orig
0.95	1.0	1	1.08	41.9

…while the emul_ISFs come from .csv results @ (yourChange).vdf run on the PC:

AA	AB	AC	AE	AI
acce	bg	pp	dura	
ISF	ISF	ISF	ISF	ISF
emul	emul	emul	emul	emul
1,12	1	1,03	1,08	38,4
0,9	1	1	1,1	41,2

¶

Apology: The above example was not well chosen to see relevant effects. The author is struggling to put this chapter first time together, quickly for the V.3.0.1 launch, and just picked from his phone what was available at the moment, I might patch it over with a better example in a later update, or I (or maybe you?) provide an adjunct case study.

Principal limitation of any of your emulations

Note that always the first biggest change regarding insulinRequ and SMB size in emul vs. orig. is the most relevant. This is because:

- Doing that change would change, by the same amount, the iob basis for the next following loop decision
- Doing that change would certainly change the course of the bg curve about half an hour afterwards (to be precise: for the duration of DIA, for the extra insulin), so then it is anybody's guess, for instance how difficult of a job presents itself to the loop to "attack" with duraISF.

This is one of the reasons why setting up your FCL (section 4) should be a iterative process, seeking solutions (with mainly bgAccel_ISF) for the first rise (and a range of different meals) first, AND NOT concurrently already "tuning" the dura_ISF.

> Both, bg peak height, and pattern of insulin activity from the present iob (that is only sluggishly further adjustable driven by dura_ISF and evtl. by bg_ISF) depend on how the first bg rise stage was managed by autoISF (your bgAccel_ISF_weight setting, and others).

Another conclusion you might draw, is to limit use of the emulator *on the PC* largely to analyzing the (in FCL extremely important) first rise, to seek bgAccel_ and pp_ISF_weights.

Then to test these, and many other like ideas for changed settings "in real life" emulated *on the smartphone*: A synthetic voice will announce to you how (if) each actual decision would differ. See next section!

11.4.3 Real time speech synthesized treatment suggestions

> Unfortunately this great feature is **not available on i-Phone**. Look in (updates of) section 11.3 for eventual alternatives.

At time points when the (yourChange) setting would result in smaller or greater difference in SMB insulin delivery (compared to the real "noChange" run), you can get a real-time notification **via speech synthesis**, and you can assess the situation in real-time yourself.

> Also if just the noChange.vdf is running (and even in AAPS without autoISF) , the speech synthesis could alert you to "carbs required" messages, for instance.

If for instance a suggested extra, or bigger, SMB makes sense, **you can add this portion manually*** ….**and observe**, for this meal, whether this bolus was OK and you should switch in

direction of the different setting you were investigating (which would automatically give you that extra in the future).
Likewise, you might choose **not** to intervene, but regret it an hour or so later, seeing the further development **without** implementing the supposed improvement.

In Full Closed Loop, you don't need any buttons at the bottom of the AAPS main screen. But for such test phases it is practical to re-install the insulin button at the bottom of the AAPS main screen (Preferences/Overview/Buttons/Insulin -> ON).

After a couple of days, you will get a feel for whether you want to incorporate your investigated change (or a gradual step towards it) into your active AAPS settings.

Warning: Your settings must always work for a variety of meals. Do not put too much effort into optimizing one situation! (See case study 8.2).

Activating and silencing emulator suggestions

For silencing the suggestions from voice synthesis you have the following options:

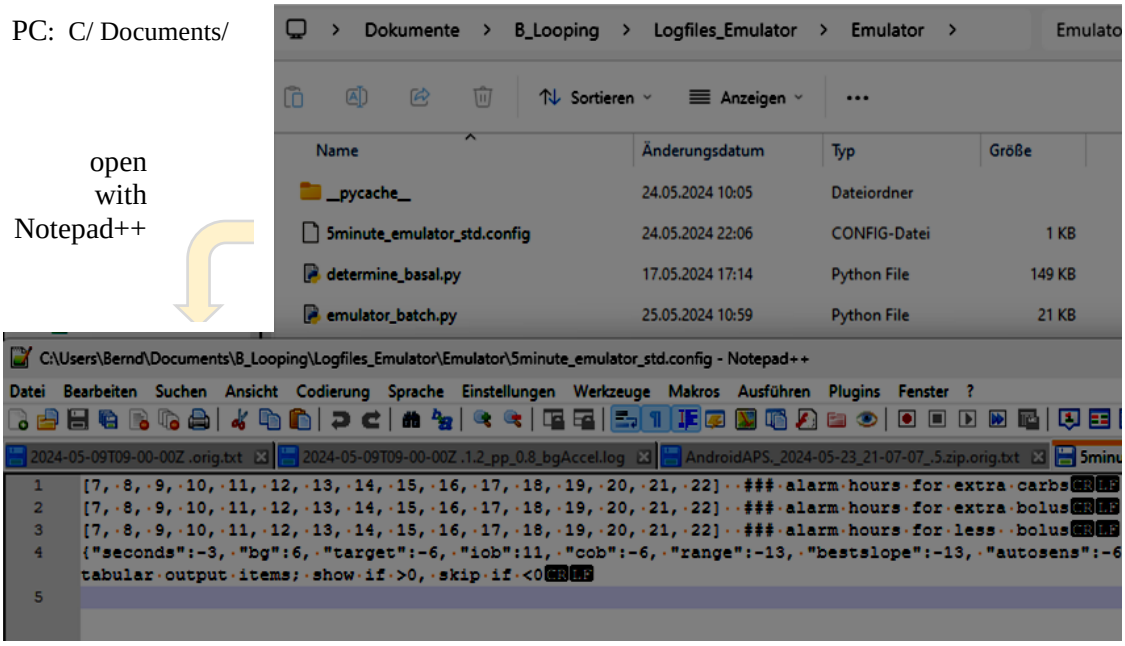
(1) Change **principal settings** what shall be announced (e.g. only if bigger SMB size is suggested, or also warnings about carbs eventually needed?), and in **which hours of day**, to make any announcements via speech synthesis .

These are set in the (1 or) 5minute_emulator_std.config file:

Go on **PC** into the config. file (see illustration). Open it with Notepad++ and edit the hours there for when you would want (no) announcements regarding: extra carb need (line 1), extra bolus need (line 2), or less bolus (line 3):

PC: C/ Documents/

open with Notepad++



Save the changes, and copy the file also into your **phone** at Internal memory/AAPS/logs /info.nightscout.androidaps (see 11.1.3) over the 5minute_emulator_std.config

(2) Turn off **phone volume** (silence media + switch on do not disturb)
Of course, this also shuts off many other potential alerts that you might not want to shut off.

(3) "**Kill" (and later resume) the "what-if" emulation**. This could be done by de-selecting the (yourChange).vdf in step . However, this stops (or interrupts, until you re-start) the entire emulation and you will have no tabular data later for the silenced time.

(4) **De-activate Qpython 3L app** (temporarily?): Press on Qpython 3L app icon, force close or remove necessary permission – re-activate (or need to re-start emulation then ??) when you want to hear again .. However, this stops (or interrupts, until you re-start) the entire emulation and you will have no tabular data later for the silenced time

(5) Probably better than the 2 afore mentioned options would be to **run (for the intended silencing period) exclusively the noChange.vdf,**
Then you will not get any speech outputs (because you are NOT investigating a what-if question, in that case). But you get all data (the un-interrupted noChange actual run) and later on the PC still can investigate any "what-if" scenarios

How to change the .vdf reference during a run, see section 3.8, 9) – or go direct to the instructions in the Github repo at: https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1/Documentation%20in%20English/How-to-run-the-emulator-on-the-phone.pdf and there p.5,_under above sub-headline ".Stop the emulator, or switch…"

(6) A variation of option (5) would be to silence all but the (less frequently occurring but most important) one line in your config definition (done according to section 11.1.5)
Then go through step 6) in section 11.2.2 , you can also switch between different …config files, e.g. to silence the less important outputs. See also step 4) in section 11.1.3,

Note that this area (silencing) has not been researched much, and good answers are likely lifestyle and phone specific.

Please share your experiences with the emulator in Discord / Full-Closed-Looping / HOW TO /_emulate-aaps, at: https://discord.gg/n3tD5eXExC