

## 10. Tuning of autoISF settings for Full Closed Loop aided by the emulator

### V.3.51

**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](#)



#### 10.1 Installing the Emulator on your PC

10.1.1 File structure on your PC

10.1.2 Load config and py files

10.1.3 Desktop button "Emulation\_start"

10.1.4 Other software requirements

#### 10.2 Analyzing **loop decisions** in logfiles

10.2.1 **noChange.vdf**

10.2.2/3 Locate logfiles / prepare Emulator

10.2.4 Run emulation and inspect results

10.2.4.1 .txt (all SMB tab infos)

10.2.4.2 Tabular (.csv) presentation of all loop decisions

10.2.4.3 Manual extraction of key data into .xls or .odc

10.2.4.4 .pdf chart

#### 10.3 What-if analysis

10.3.1 Define (**yourChanges**).vdf

10.3.2 Run emulation

10.3.3 Inspect results

10.3.3.1 Logs (all SMB tab infos)

10.3.3.2 Tabular (.csv) presentation of all loop decisions

10.3.3.3 Semi-automated extraction of key data

10.3.3.4 .pdf chart

#### Available related case studies:

Case studies **still missing:**

Based on older autoISF and older emulator versions, examples from emulator use can be found in [case study 6.2](#), in [case study 4.1](#) (last pages there), and [case study 8.2](#)

You can set up and tune the system for Full Closed Loop as described in previous sections.

Doing this by frequently analyzing screenshots that must be taken in real-time of the AAPS **SMB tab** is tedious, however.

More elegant and precise tuning can be done with a special evaluation software for the AAPS logfiles, by using the **emulator**. It is described here: <https://github.com/ga-zelle/APS-what-if/> / Documentation-in-English. There (under / Software) you find the files needed to download on your PC, and the primary instructions:

← → ↻ https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4\_ai3.0.1\_dev/Documentation%20in%20English/A%20README.md

Note: **Look it up in the most up-to-date branch**

**Files**

A3.2.0.4\_ai3.0.1\_dev + 🔍

🔍 Go to file t

Documentation in English
 

A README.md

DRAFT - Guide to VDF Files for the...

DRAFT - autoISF2.2\_How\_to\_start\_...

Example Emulator study - Negativ...

How-to-create-the-autoISF-factor...

How-to-preview-autoISF-impact.pdf

Installation Guide.pdf

The AAPS Logfile System.pdf

Dokumentation auf Deutsch

software

Anleitung determine\_basal emulator...

Das System der AAPS Logfiles.pdf

Demo\_Sports\_Adaptations.vdf

Instructions determine\_basal emulat...

**APS-what-if / Documentation in English / A README.md**

ga-zelle Case study using emulator

Preview Code Blame 57 lines (38 loc) · 2.59 KB Code 55% faster with GitHub Copilot

List of documents; click arrow to see their purpose
 

- ▶ Instructions for the AAPS emulator.pdf
- ▶ Guide to VDF Files for the AAPS Emulator.pdf
- ▶ How\_to\_create\_the\_autoISF\_factor\_plot.pdf
- ▶ How-to-preview-autoISF-impact.pdf
- ▶ How\_to\_start\_tuning\_autoISF2.2.pdf
- ▶ Example Emulator study - Negative IOB Problem or else
- ▶ Installation guide.pdf
- ▶ The AAPS Logfile System.pdf
- ▶ How-to-get-larger-SMBs.pdf

In the emulator, you can see in tabular and graphical form, which autoISF component, and other settings, contributed to SMB values that determined the glucose curve.

In the following, we look into how you create your relevant data.

Application examples for tuning are given in associated case studies (we newer ones).

Note that the iOS based variants of autoISF for Trio or iAPS (oref loops for i-Phone) can not use the emulator. Refer to [section 11.3](#).

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

## 10.1 Installation of the emulator on your PC

Installation is a one-time process, and you best refer to the installation guide of the developer, here: [https://github.com/ga-zelle/APS-what-](https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1/Documentation%20in%20English/Installation%20Guide.pdf)

[if/blob/A3.2.0.4\\_ai3.0.1/Documentation%20in%20English/Installation%20Guide.pdf](https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1/Documentation%20in%20English/Installation%20Guide.pdf)

Below, I attempt to spell out some additional details “for IT dummies” (like myself)

## 60 10.1.1 Create your PC folder structure

The suggested folder names and structure shown below is of course not mandatory, but only a suggestion.

61 On your PC, create a folder “**Logfiles\_Emulator**” with 3 sub-folders: “AAPS\_logs”, “Emulator” and  
62 “Emulator\_Studies”

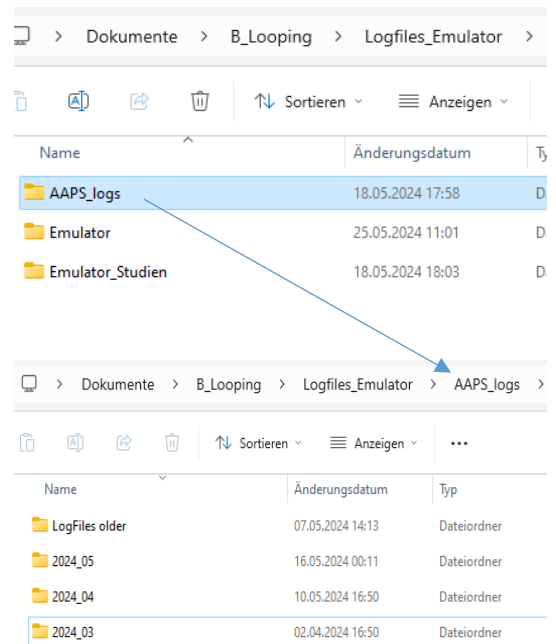
**AAPS\_logs:** Put all your stored AAPS logfiles into that sub-folder. My folder structure for Logfiles and Emulation on the PC has 3 monthly folders, plus one folder with data from previous months and years (which I am less likely to analyze).

68 The logfiles you ALWAYS must copy-in from your phone  
before they get automatically erased there after x days  
69 (about 2 weeks, much shorter for 1-minute Libre3).

It is advisable to additionally store a pdf from **Nightscout Reporter** in the file for every month, with daily glucose charts, 24h scatter graph, etc. From it, you can much easier find which days and times are of high interest to analyze with the emulator.

74 **Emulator:** Neighboring “AAPS\_logs” is the “Emulator” folder into which most downloads from the  
75 developer’s repo will go in [section 10.1.2](#)

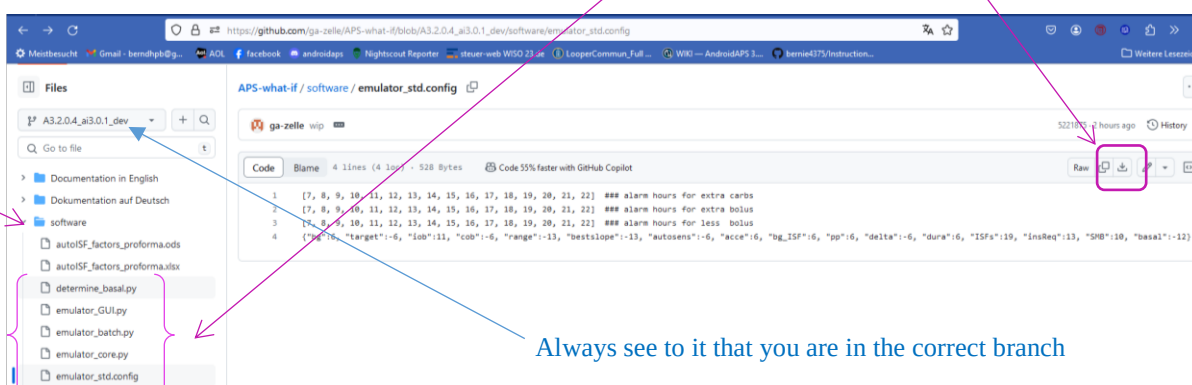
77 **Emulator\_studies** is a folder, where, for now, you should provide some sub-folders “Study\_1”, “Study\_2”  
78 etc.. Later, when you use the emulator, you will use these “addresses” for the program to dump results from  
79 the emulation into.



## 81 10.1.2 Downloads

82 1).Download from: <https://github.com/ga-zelle/APS-what-if/> **software**, the **.config** and four **py.** files.

83 To do this, you must (5x, one at a time): click on the name here, and for download here



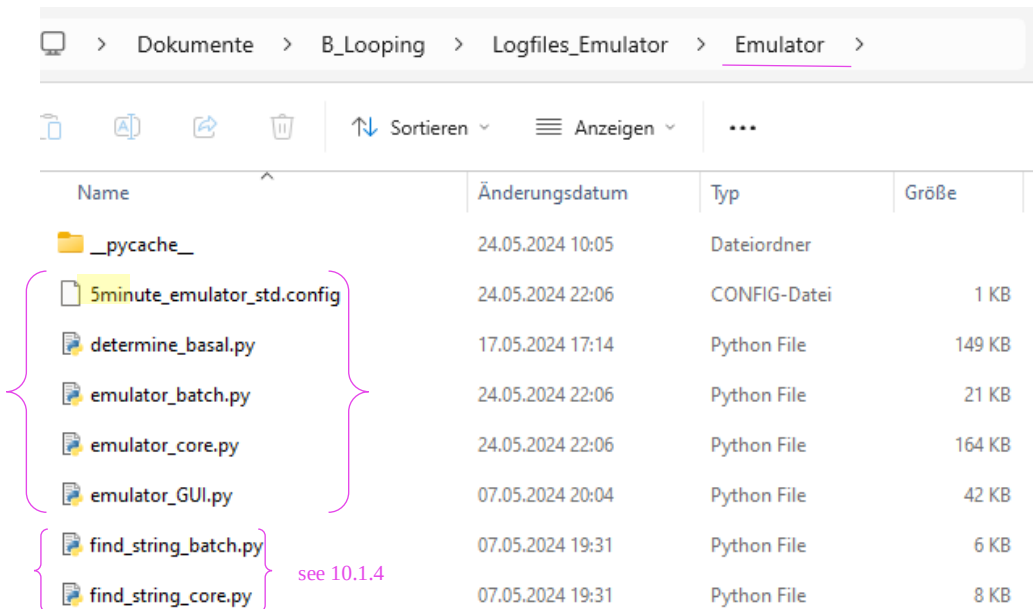
## Always see to it that you are in the correct branch

Always make sure you use the files from the example above: These files will work with AAPS dev version 3.2.0.4 with autoISF version 3.0.1).

Always keep your AAPS x autoISF and also the emulator related files up-to-date. If you can't get your Emulator run, look in the Github repo whether there is a newer .py file (even with the same name; there may be updates that iron out problems that may have been reported only with certain AndroidOS versions etc etc))!!

2). Retrieve these 5 downloaded files on your **PC (list of recent downloads)**, then:

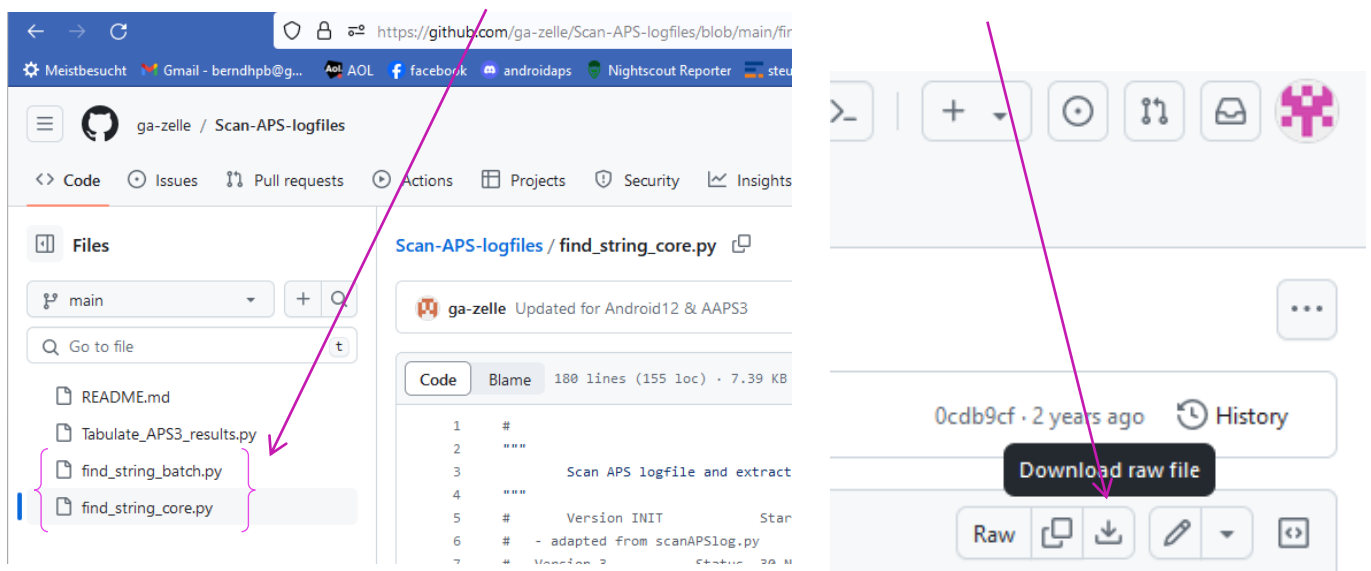
3). Shift each of these 5 into your "Emulator" folder:



Name	Änderungsdatum	Typ	Größe
__pycache__	24.05.2024 10:05	Dateiordner	
5minute_emulator_std.config	24.05.2024 22:06	CONFIG-Datei	1 KB
determine_basal.py	17.05.2024 17:14	Python File	149 KB
emulator_batch.py	24.05.2024 22:06	Python File	21 KB
emulator_core.py	24.05.2024 22:06	Python File	164 KB
emulator_GUI.py	07.05.2024 20:04	Python File	42 KB
find_string_batch.py	07.05.2024 19:31	Python File	6 KB
find_string_core.py	07.05.2024 19:31	Python File	8 KB

Note: Use 1minute:emulator\_std.config in case you use Libre3 (1 min) as your CGM

4). From another section in Github, "Scan-APS-logfiles", fetch two more .py files by repeat steps 1)-3). for these two. They are from: <https://github.com/ga-zelle/Scan-APS-logfiles/blob/main>



The screenshot shows the GitHub repository for 'ga-zelle / Scan-APS-logfiles'. The 'Files' tab is active, displaying a list of files including 'README.md', 'Tabulate\_APS3\_results.py', 'find\_string\_batch.py', and 'find\_string\_core.py'. A purple arrow points from 'find\_string\_core.py' in the list to its detailed view. In the detailed view, a purple arrow points from the 'Download raw file' button to a separate window showing the raw file content.

5)-Retrieve in your PC's downloads folder, and move them into your emulator file (as already was included 2 pictures higher up).

### 10.1.3 Create an “emulation start button” on your desktop

One of the files in your “Emulator” folder is “**emulator\_GUI.py**”

- Create, in your Emulator folder, a **link to it**
- Drag **that link** onto your **desktop**
- Name it something like “Emulator\_start”: This is your **start button** for emulations on the PC

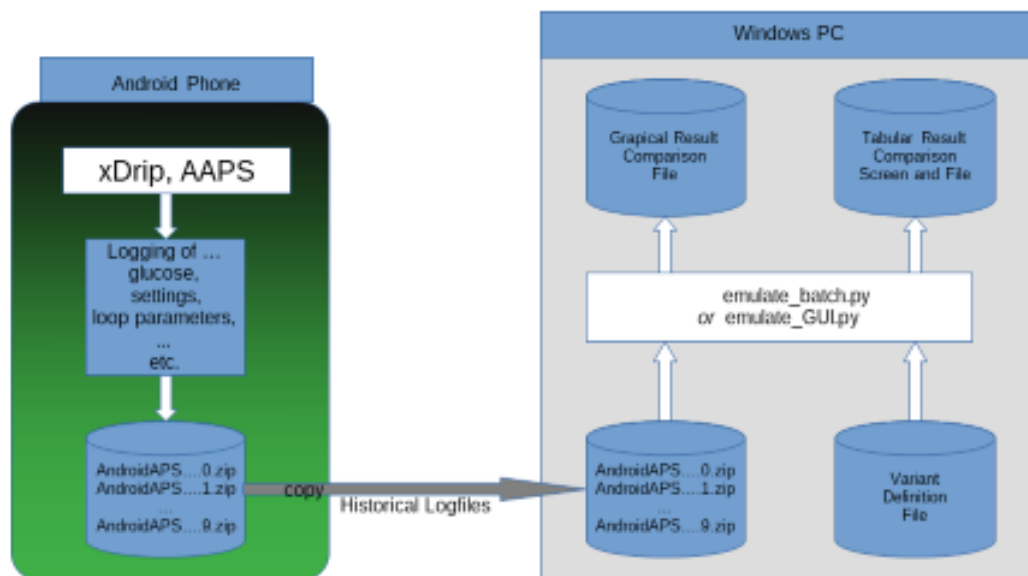
### 10.1.4 Other software requirements

Make sure you have **Notepad++** on your PC (see [section 10.2.1](#)).

QPython 3L will be needed on the smartphone, later (see [section 11](#)).

## 10.2 Analyzing loop decisions in logfiles

Instead of making many screenshots every 5 (or, w/ Libre3, every 1) minutes after meals, and analyzing them later, a much more elegant and powerful way to analyze your loop decisions (and how you might want to influence them with different settings, see [section 10.3](#) for this), is to use the emulator.



Sketch of Running the Emulator on a Windows PC

Github/ga-zelle /  
APS-what-if

124 10.2.1 Set up a “no change” .vdf file.

125

126 1). To do this, just open **Notepad++** (from list of all programs on your PC).

127

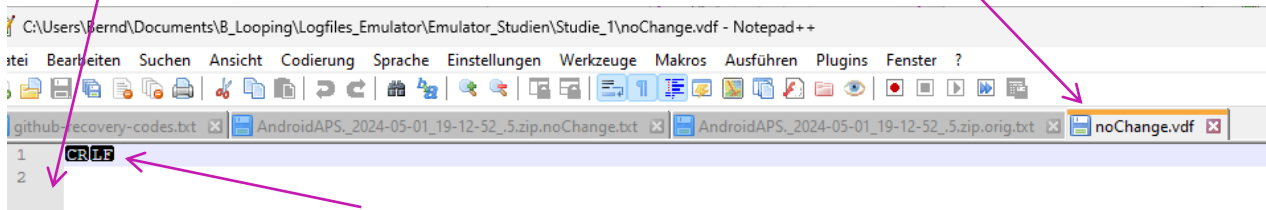
128 2). Name your file “noChange.vdf”.

129 It is just empty in the lines that would define any change to be investigated.

130 Note: for “what-if” analysis, entries will be made (in a second .vdf later, see [section 10.3](#) )

131

The no change .vdf should look like something like this:



132

133 Erase any entries after CR LF and also in lines 2 ff, if any

134

135 3). Store that “noChange.vdf” in your “Emulator studies” folder, on the top level, besides the single studies folders

136

137 4). From that position, you always make a copy, and paste *into each* Studie\_1 ...n :

138

139

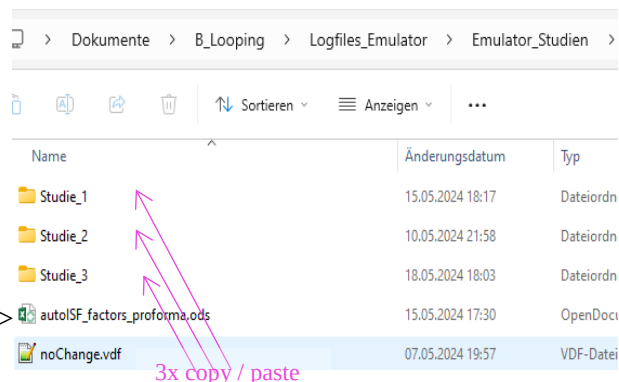
140

141

142 See [section 10.3.3.3](#), regarding this ->

143

144



145 10.2.2 Locate relevant logfiles and prepare the Study\_n folder

146

147 1). Make sure you have the AAPS logfiles that you want to analyze in your ~~Emulator\_Studies/~~  
148 ~~Study\_n~~ “AAPS\_logs folder”

149

150 2). In your “Emulator\_Studies” folder, create (or use a prepared) “Study\_n” sub-folder, with a  
151 copied-in (not: moved!) your noChange.vdf (It must be in all Study\_n files).

152

153

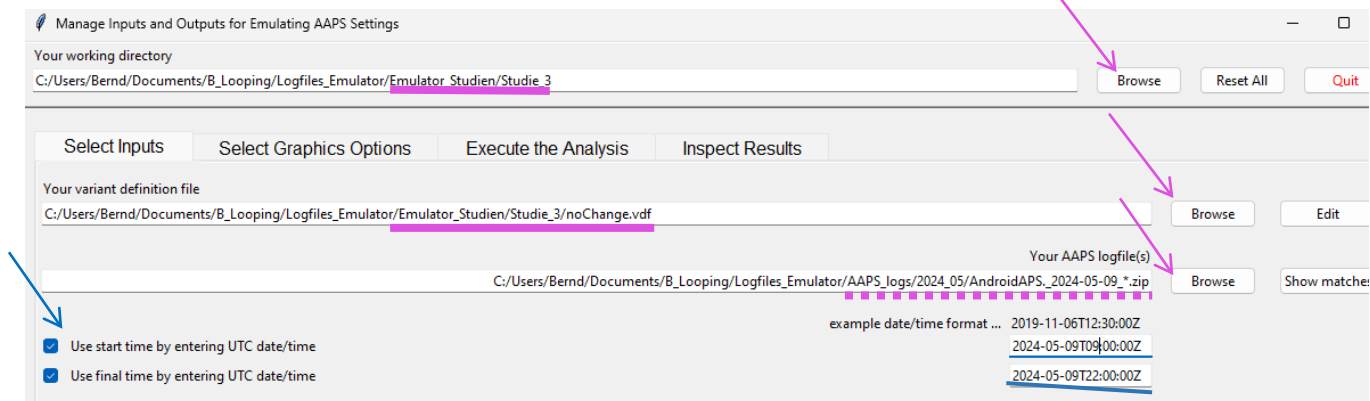
154 10.2.3 Prepare your emulator run for Study\_n

155

156 Now go to your PC **desktop**, and start the emulator by **just pressing the button “emulator start”**  
157 that you installed in step [10.1.3](#)

158

159 This opens a big dialogue box with 3 fields that you must fill in with the applicable path (*without* any  
160 quotation marks “.”) from your Windows Explorer file system, best done via (3x) Browse button:



161

162 a) The top box marks the path to your current emulator project (“Studie\_3” is my “Study\_n”  
163 where I want to store results)

164 b) The middle box marks the path to your current vdf (what kind of analysis; here:  
165 “...noChange.vdf” = *read-only*. (For *what-if*, see [section 10.3](#))

166 c) The third box marks the path to your AAPS logfiles you wish to look into. A good way to do  
167 this is:

- 168 • Browse in your Windows Explorer to any logfile from the desired day (2024-05-09 in above ex-  
169 ample)
- 170 • Replace the time with an asterisk \* (this means you look at **all-day** data, in UTZ time).  
171 Check whether this will work by pressing Show matches .  
172 You should see all logfiles from that day in a pop-up info box.

173 • As I wanted to look at 11 am –midnight (for lunch and dinner related data), I :

- 174 ○ clicked the bottom left two boxes
- 175 ○ copied the date 2024-05-09 over the default date in the bottom right two data fields
- 176 ○ after T (for time), I entered the desired time of analysis AFTER conversion into my local  
177 time (Central EU summer time minus 2 hours = UTZ; so to look at 11 to midnight of  
178 my AAPS screen, I must enter here 09.00:00Z, and below it 22:00:00Z).

179 Entries at the bottom are not mandatory, but when clicking these little boxes (bottom left)  
180 you can define a start and/or an end-point for analyzing, within the logfiles specified in the  
181 field above.

182

183

## 10.2.4 Run emulation

Now we are ready to go: Press “Run emulation”

This produces sometimes an error message (e.g. if you have a syntax error, or incompatible software versions: => seek help, in the Github materials provided by ga-zelle, or in Discord/Full-Closed-Looping/emulate-aaps here: <https://discord.gg/n3tD5eXExC>)

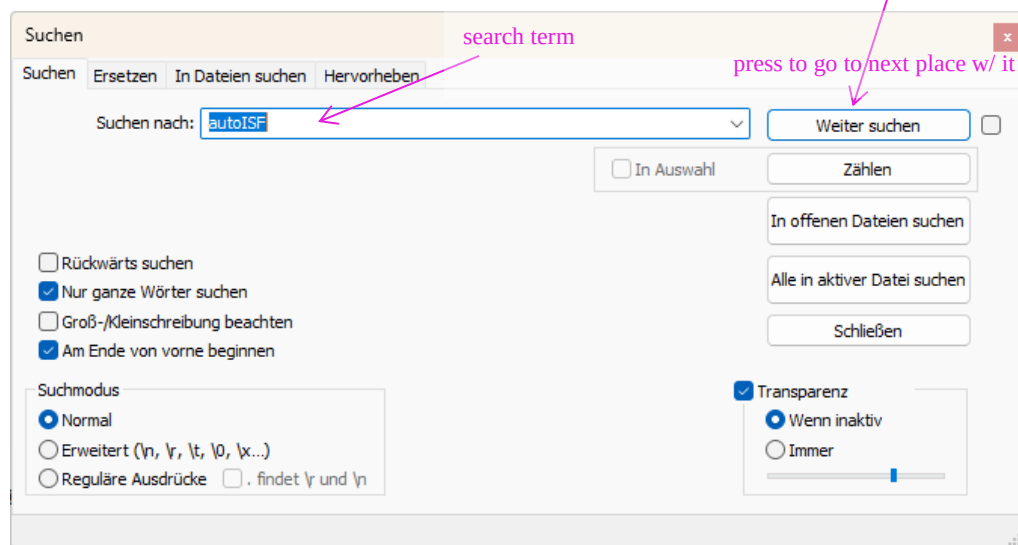
After a short moment results should show up, which you can look into in a couple of ways. First you could have a quick look into the **.log** file to see whether the run had errors (see [section 3.8](#))

### 10.2.4.1 SMB tab contents in (date..) noChange.txt result file

This ...txt file basically gives you “all the SMB tab” infos, in a super long list (but without needing to make screenshots, in real-time, every 5 minutes.)

Search options help find what lines are of interest to your analysis:

By using the **search function** you can jump, in that super long list, to all places that e.g. have „autoISF“ in it or „script debug“, or „SMB disabled“ (if you want to analyze when that happened). Precise spelling, as in this .txt (or in SMB tab) is of course important.



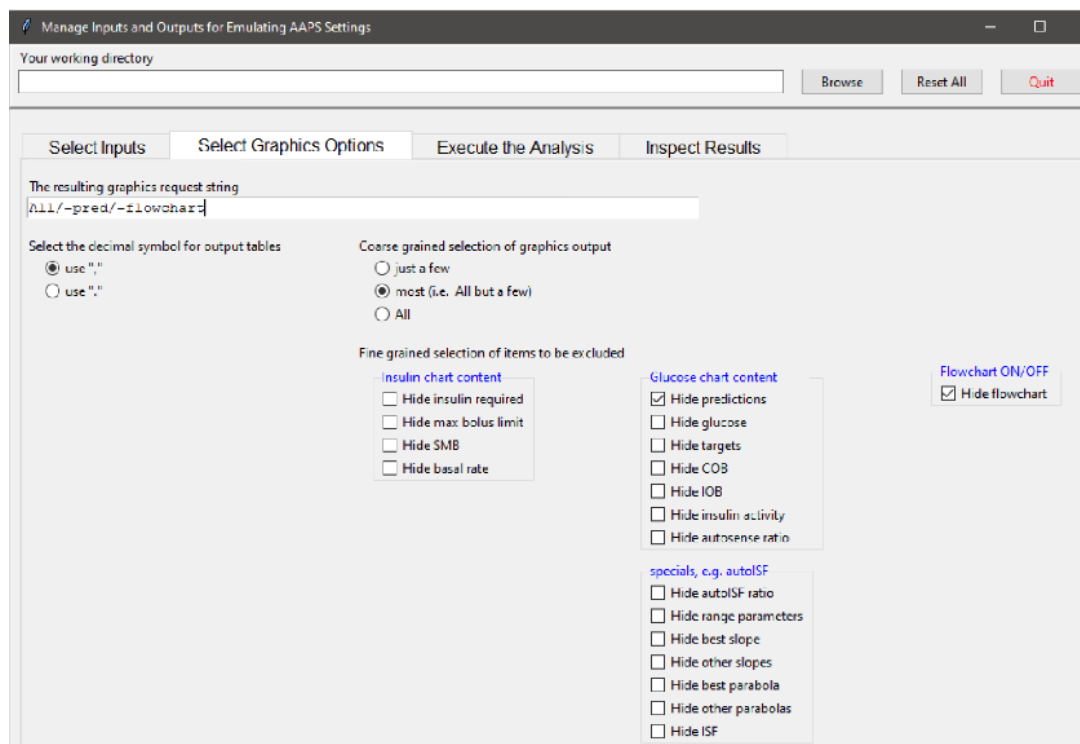
### 10.2.4.2 Table of results (...noChange.csv file)

The .csv file in your project folder gives a tabular presentation of how parameters like bg, iob, iobTH, the various ISF contributors, bg target, insulinRequired etc. develop every 5 minutes, and what SMB size and %TBR resulted.

It is a vast table, so you may want to reduce it to something more “digestible”, either after transfer to your standard calculation program (next [section 10.2.4.3](#)). Or you can also make settings to suppress information you are usually not interested in (or do not know how to interpret, anyways) under “Select Graphics Options” when you open the emulator, before executing any analysis:



213 First, select your preferred way of outputting decimals (point or comma).  
 214 Then select whether you want “All” possible outputs in the graph, or “Most” = all except those you tick “off”  
 215 in the boxes for each output parameter.  
 216 In case you would use “Some/just a few”, you would have to tick those few you that do want to see, by ticking  
 217 the corresponding boxes.  
 218 Recommendation is to look at (nearly) everything offered (as your default setting that you can leave  
 219 untouched in most of your emulator runs):



220  
 221 It might be easier, to not deal with customizing the csv file, and rather copy the data into your  
 222 favorite calculation program:

223  
 224 [10.2.4.3 Analysis of the noChange.csv table in Excel or LibreOffice calc.](#)

225  
 226 Best copy the entire table into a new .xls or .ods sheet, where you can:

- 227 • add right next to the UTC (Unix Time Code) your corresponding “AAPS **time**”

228 For instance, adding +2/24 translates the UTC column into central European summer time column  
 229 next to it (where currently a row of Z stands). Likewise, subtract like -5/24 from UTC for an US East  
 230 Coast time scale.

231 *(Fun fact: Our oref loop stubbornly works on UTC, un-impressed by our folly to jump twice a year*  
 232 *into or out of a local summer time, or to travel across time zones. If some data get lost in translation*  
 233 *there, it is only to us, with our stupid time change. For the loop, its database (e.g. on insulin activity)*  
 234 *remains unambiguously intact).*

235 Highlight all time fields (both entire columns), and switch from hh:mm:ss format to hh:mm.  
 236 *(While the seconds are important for the loop's calculations, for our comparison with Nightscout or*  
 237 *other charts and data, it is much easier without the seconds attached)*

- 238 • **hide** any column you find less important to look at for your intended analysis

239 That way, “boxes” (data fields) retain their original position in tables

240 Also, in case later you want to look into additional info, you can simply un-hide the relevant columns  
 241 (or lines:.)

- 242 • **hide** lines (time segments) you find less important to look at for your intended analysis

243

244 Usually you will color mark where relevant SMBs were given, which of the ISFs (and underlying  
 245 weights) was strongly contributing (note that this can be good or not good). Also where iobTH was  
 246 exceeded, whether an Automation kicked in e.g. setting a TT, or when there were periods with zero  
 247 insulinRequired.

248 In [section 10.3.4](#) we present an extra tool that does a standardized table reduction and color marking  
 249 for you!

250 You may be able to formulate a hypothesis or two, what settings (...ISF\_weights, iobTH%,  
 251 SMB\_range\_extention, autoISFmax ...) should be changed for improvement (then go to [10.3](#))

252

253 [10.2.4.4.. Graph noChange.pdf](#)

254

255 After your emulation run, under Inspect Results, you can open the pdf file that is last in the results list  
 256 offered.

257

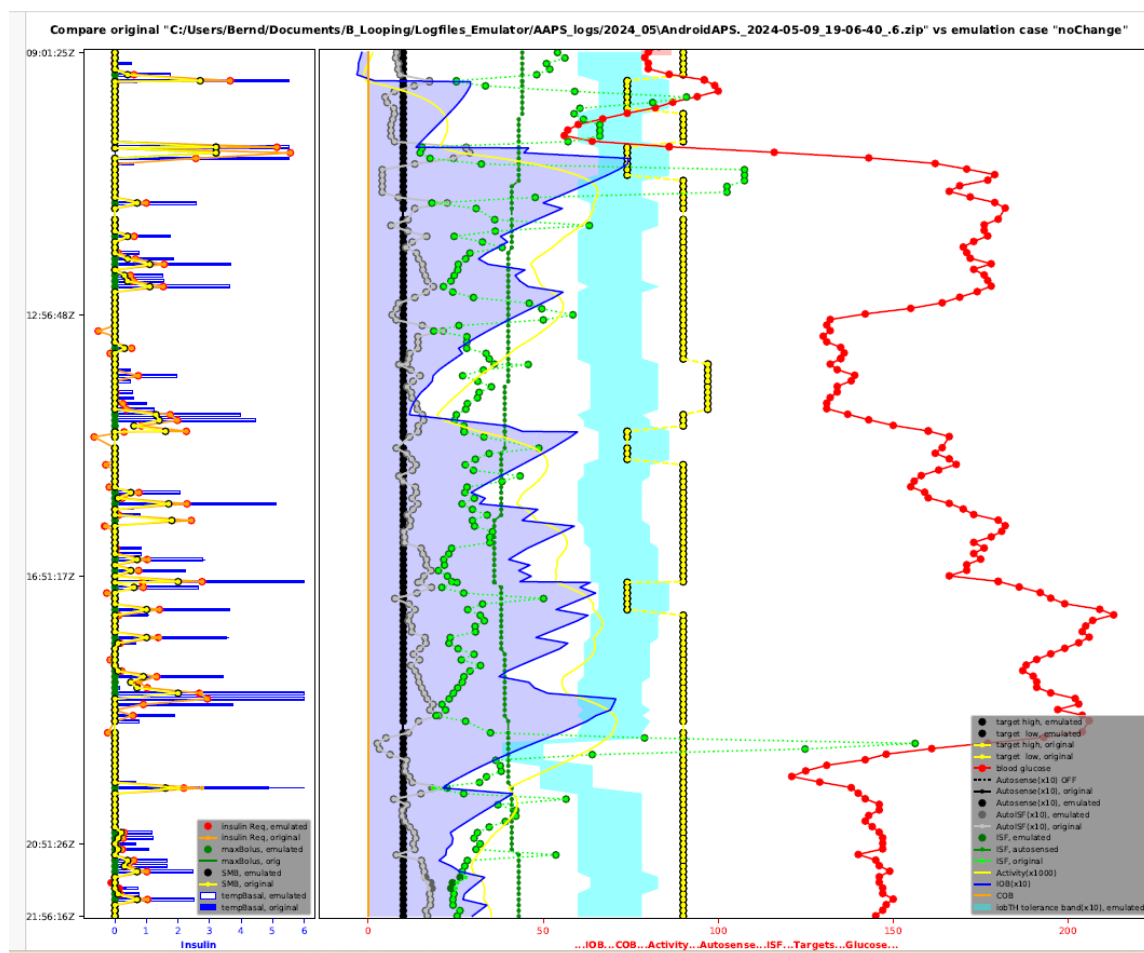
258 This **noChange.pdf** is a chart that shows along the time axis (down), from right to left:

- 259 • Red: the bg curve
- 260 • Yellow: the bg target (note that I do no manual “EatingSoonTT” but for bg rises over +10 mg/dl  
 261 I have an Automation that sets low TT for a couple of minutes)
- 262 • Light blue corridor: Left edge is set iobTH, and bandwidth +30% (would be +20% at elevated  
 263 TT)
- 264 • Dark blue line: iob (exceeding twice the iobTH, with temp. SMB shut-off

265 As bg did not convincingly come down enough, one could hypothesize that iobTH should be  
 266 elevated. ((But, again, this would have to be confirmed also with other kinds of meals)).

- 267 • Thin yellow line: Insulin activity

- Green dotted line: ISF as would result from AAPS w/Autosens
- Green scatter points: autoISF ISF no Chage (lighter points) or what-if (darker points)
- Black line: Profile ISF
- Gray scatter points: ISF weakened (to the left of black line) or strengthened (to the right)
- Orange line: cob=0 at all times (in FCL)



More see discussed together with (yourChanges).pdf in [section 10.3.3.4](#)

### 10.3 “What-if” analysis using the emulator

In the following you see an example how you can analyze a day of logfiles, and selecting the time span of interest, for instance 11-24 h to look at how autoISF managed lunch and dinner.

You will go through the emulator exactly as you already did in [section 10.2](#). where you exclusively had the noChange.vdf on bord.

However, this time you focus on (yourChange).vdf, see below, [10.3.1.](#)

Repeat, if you have two or more such vdf defined.

(Just clear old results before pressing “execute analysis” each time.)

**All results are automatically captured** for all runs, all **in your selected “Study\_n” folder**, together with the noChange results

- Results files with noChange in their name are always your actual loop data ...
- as opposed to results on “what-if , that contain name of the (yourChange).vdf in their file name see e.g. at around [line 380](#)

**How to proceed, step by step:**

**10.3.1 Define your investigated changes in (yourChange).vdf** (one,. or several)

1). Define for which one (I suggest max three) parameter(s) in your current profile settings you want to look into a different setting. Recommendation is to use a factor, like for example: current setting \* 0.9 , or current setting \* 1.2, and use that in your naming for this vdf file, too.

You may want to consult [APS-what-if /Documentation in English/Guide to VDF Files for the AAPS Emulator.pdf](#) Access directly, or via [section 3.8](#)

Within one study, you can make several emulator runs with several (yourChange).vdf files (all based on what really happened, as captured with the noChange.vdf).

All results, like the csv results table, will appear then *several times* in your study file, only *with different name endings* as in the underlying vdf.

Example: I like to check in my actual data (they are in my noChange.vdf emulator run), **in which time points the following parameter changes would make a** (how) big **difference** in the loop’s decision:

- 20% higher bgAccel\_ISF\_weight to boost the first SMBs stronger: How would that tend to ramp up early iob; and might that get too strong in other parts of the data? Or does it bounce into a restriction (maxSMB size; autoISFmax; iobTH...) that I might need to widen?
- Doubling my cautiously set bgBrake\_ISF\_weight shall give me insight into the workings of that parameter (and whether using a much smaller weight than for bgAccel\_ISF\_weight is really what I should keep doing)
- As my bg came down from a persistent high quite slowly, I elevate the dura\_ISF by 20%

*Tuning advice:* Actually, it would make more sense to first find my “optimal”, maybe indeed elevated, bgAccel\_ISF\_weight. *Then*, in a *new project\_n+1*, do (automatically) a noChange run

**with that**, plus a (yourChange) run with the stronger dura weight, investigated *on that* basis.

Reason: 1) As we always say, better do only one change at a time. 2) A better job with bg control via bgAccel\_ISF will reduce the peak height and provide a different (easier) scenario for dura\_ISF to manage.

2). Now, to **write** your (yourChange). **vdf for the emulator** (this is same procedure as you did in section 10.2.1 for the noChange.vdf):

- just open Notepad++ (from list of all programs on your PC) to create a new vdf:.

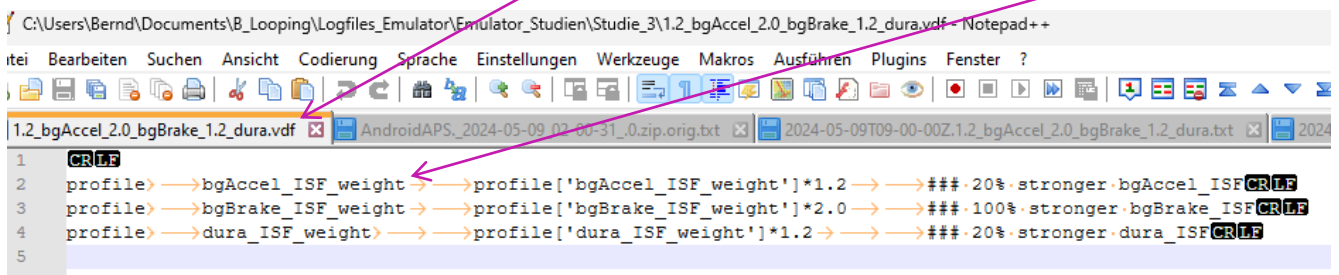
Alternatively you can also take another pre-existing vdf file, copy it into your current project, edit as desired, and give it a new name (re-name it)

**Caution:** Make absolutely sure (best by looking it up in the SMB tab, down in the profile set section) to **spell each term exactly** as your loop uses it (probably w/ decimal points, not comma)

- ...when you make one line per parameter (separating entries with spacers->):

profile->(parameter) ->->profile['(parameter)']\* (factor)->->###(comment as you like)

The (yourChanges) .vdf should look like something like this:



```
1 CR LF
2 profile> ->bgAccel_ISF_weight ->->profile['bgAccel_ISF_weight']*1.2 ->->###.20%.stronger.bgAccel_ISF CR LF
3 profile> ->bgBrake_ISF_weight ->->profile['bgBrake_ISF_weight']*2.0 ->->###.100%.stronger.bgBrake_ISF CR LF
4 profile> ->dura_ISF_weight ->->profile['dura_ISF_weight']*1.2 ->->###.20%.stronger.dura_ISF CR LF
5
```

CR = LF = Erase any entries after CR LF and also any entries in lines below, if any

- name your vdf (in example below: 1.2\_bgAccel\_2.0\_bgBrake\_1.2\_dura.vdf) ...
- ... and store that in the folder for your current Study\_n you are about to start (see my storage path C: : ..... Studie3.....vdf – Notepad++ in the top line:)

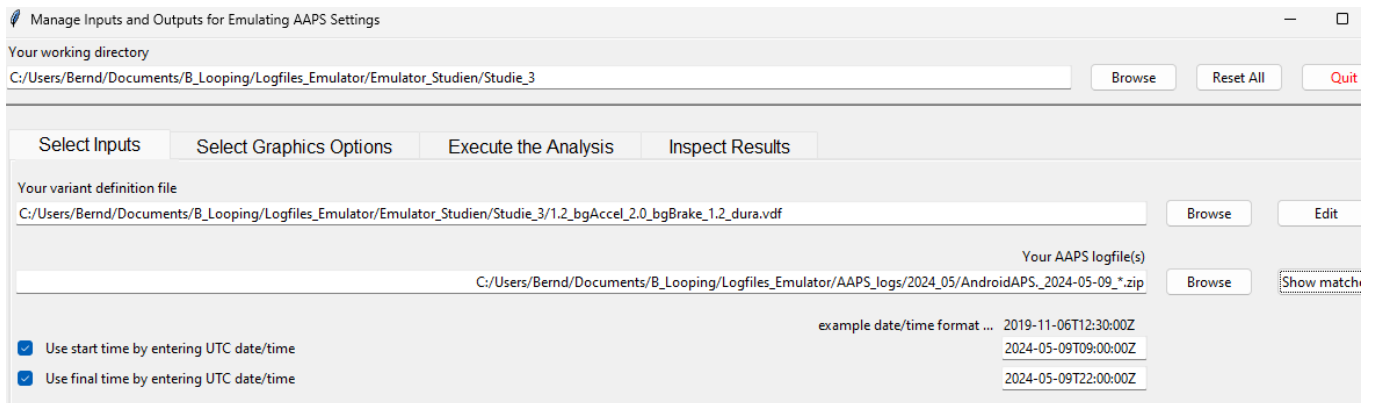
### 349 10.3.2 Run the emulator with (yourChange).vdf

350

351 The “what-if” emulator run is done the same way as you did the noChange.vdf run ([section 10.2](#)), which had no  
352 (yourChange).vdf on board => No surprise, running an emulation with only that noChange.vdf, yields same result in  
353 emul columns as is orig columns (as we earlier had observed, section ), - However, now :

354

355 the **(yourChange).vdf** must be loaded into the 2<sup>nd</sup> input field, where formerly you had the noChange.vdf.:



356

357

358 In the 3<sup>rd</sup> input field, give the path to your stored logfiles. A good way to do this is:

- 359
- Browse in your Windows Explorer to any logfile from the desired day (2024-05-09 in above example)

360

  - Replace the time with an asterisk \* (this means you look at all-day data, in UTZ time). Check whether this will work by pressing Show matches . You should see all logfiles from that day in a pop-up info box.

361

  - As I wanted to look at 11 am –midnight for lunch and dinner related data, I :

362

    - clicked the bottom left two boxes

363

    - copied the date 2024-05-09 over the default date in the bottom right two data fields

364

    - after T (for time), I entered the desired time of analysis AFTER conversion into my local time (Central EU summer time minus 2 hours = UTZ; so to look at 11 to midnight of my AAPS screen, I must enter here 09.00:00Z, and below it 22:00:00Z).

365

366

367

368

369

370 After making these entries, press Execute the Analysis, (evtl also Clear old Data) and then press Run

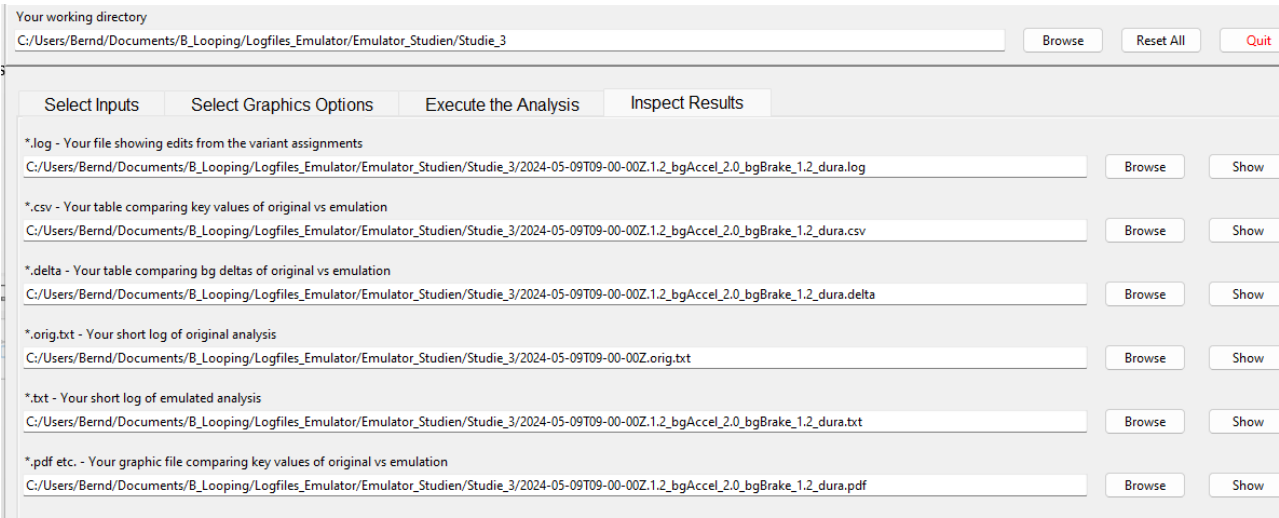
371 Emulation, you can look the results up under “Inspect Results”. First you could have a quick look into  
372 the **.log** file to see whether the run had errors (see [section 3](#).)

373

374

375 10.3.3 Emulation results

376



377

378 All results from your (yourChanges).vdf emulator go automatically where the noChange.vdf results are  
379 already stored, in our example into the “Studie 3” file, below:

380

381 Besides the 1.2\_bgAccel\_2.0\_bgBrake\_1.2\_dura.vdf case which I like to look into for the present high carb  
382 meal, I also prepared another vdf that investigates a factor 1.2 stronger pp\_ISF and a weaker, factor 0.8,  
383 bgAccel\_ISF (with the intention to test this, and a noChange (that ideally would already contain the conclusion  
384 on adapting the bgAccel\_ISF\_weight\*), on a low carb meal later.



385 \* Note the challenge here is to iterate between the typical meals of your personal spectrum to find **one** set  
386 of settings that work good-enough **for all** of them.

Studie_3 durchsuchen			
Sortieren Anzeigen			
Name	Änderungsdatum	Typ	Größe
2024-05-09T09-00-00Z .noChange.pdf	15.05.2024 17:17	Adobe Acrobat-D...	77 KB
2024-05-09T09-00-00Z .noChange.csv	15.05.2024 17:17	Microsoft Excel C...	51 KB
2024-05-09T09-00-00Z .noChange.log	15.05.2024 17:17	Notepad++ Docu...	35 KB
2024-05-09T09-00-00Z .noChange.txt	15.05.2024 17:17	Notepad++ Docu...	281 KB
2024-05-09T09-00-00Z .orig.txt	15.05.2024 17:17	Notepad++ Docu...	281 KB
2024-05-09T09-00-00Z .1.2_pp_0.8_bgAccel.pdf	15.05.2024 17:16	Adobe Acrobat-D...	78 KB
2024-05-09T09-00-00Z .1.2_pp_0.8_bgAccel.csv	15.05.2024 17:16	Microsoft Excel C...	51 KB
2024-05-09T09-00-00Z .1.2_pp_0.8_bgAccel.log	15.05.2024 17:16	Notepad++ Docu...	57 KB
2024-05-09T09-00-00Z .1.2_pp_0.8_bgAccel.txt	15.05.2024 17:16	Notepad++ Docu...	281 KB
2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.csv	17.05.2024 21:29	Microsoft Excel C...	51 KB
2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.log	17.05.2024 21:29	Notepad++ Docu...	66 KB
2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.pdf	17.05.2024 20:40	Adobe Acrobat-D...	78 KB
2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.txt	17.05.2024 21:29	Notepad++ Docu...	282 KB
1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf	17.05.2024 20:38	VDF-Datei	1 KB

387

388

389

 1.2_pp_0.8_bgAccel.vdf	10.05.2024 21:55	VDF-Datei	1 KB
 noChange.vdf	07.05.2024 19:57	VDF-Datei	1 KB

#### 10.3.4 ....(yourChange).txt: “what-if” impact on loop decisions (as in SMB tab )

The **noChange.txt** has all the info your series of SMB tabs had that day.

How to search in this vast list is shown elsewhere (see [section 10.2.4.3](#) ).

Likewise, the **(yourChanges).txt** gives *for each loop decision* in all detail how and why each single decision *would have* changed with the different parameter inputs you are checking out here

In the two (yourChanges) examples here, , it was a check on the difference

- a 20% stronger pp\_weight and 20% weaker bgAccel\_weight
- a 20% stronger weight for both, bgAccel\_ and dura\_ISF, and a doubling of bgBrake\_weight

would make.

Note that all these “what if” data can only give rough hints, notably about **the first** greater change that you would see with the investigated changed setting. So it works quite well for our main problem in FCL, investigating how to ramp up iob quickly after detection of acceleration.

Note that any relevant change would put your bg curve on a different trajectory, so that would influence *all following* results. Therefore, what you get here is **not** a complete modelling how your bg would have developed in the alternative scenario.

But you can investigate in which stages the parameter(s) you are looking at in your current “what-if” had big influence, and in which direction the changes would go. (see also charts shown in [section 10.3.3.4](#)).

Analyzing how to safely come down from a high glucose plateau while limiting hypo danger towards the end of digestion is also to some extent possible.

A good other way to employ the what-if analysis is real time, on your smartphone, using speech synthesis (see [section 11](#)): Then you get real-time info, as to exactly when a significantly different proposal would emerge, and can decide (and watch!) real-time whether to follow the new idea and not was probably better.

Observe that a setting change must work well for you

- not just in one point of time, and
- not just for one kind of meal,

but you must look at all time slots in the investigated meal, plus analyze with the same tool a totally different meal within your usual spectrum, how things work out there



### 425 10.3.3.2 Tabular results

426

#### 427 A) .csv results table and spreadsheet copies of it

428

429 The **noChange.csv** table gives all relevant data. Besides development of bg and iob you see the calculated  
430 insulinRequired in each loop decision, and how each of the autoISF categories contributed to the decision  
431 (notably regarding SMB size).

432

433 Note that the “**acce\_ISF**” results are only in case of positive acceleration (that is our main focus)  
434 driven by the bgAccel\_ISF\_weight setting. (These are all positions > 1.0 in the “acce ISF” columns).

435

436 **In case of negative acceleration** (decelerating rise, positions < 1.0 in the “**acce ISF**” columns),  
437 **bgBrake\_ISF\_weight is applied**. As discussed in [section 4.4](#), bgBrake\_ISF might be most  
438 important (and interesting to analyze) in slowly resorbing meals.

439

440 Note: maxBolus=0 means in this table that SMBs were not capped by maxBolus.

441

442

443 The **(your changes).csv** shows in detail how **every single** loop **decision** would be influenced by the different  
444 settings you are investigating.

445 To inspect that huge table, click on the Z behind the start UTC time entry (see black box in the Z column of  
446 the table, next page).

447 If you like to see the bg in each screen, too, go 3 or 4 columns farther to the right with your black  
448 box.

449 Then, go to window/fix. Now you can scroll through the data and always see headline and time (or time and  
450 bg level).

451 To further ease analysis, feel free to temporarily erase (hide) any columns that you (think you) do not  
452 need for the intended analysis. More suggestions see in [section 10.2.4.2](#)

453  
454

456

461 This route is good to compare quantitative impacts of autoISF categories in critical time points.

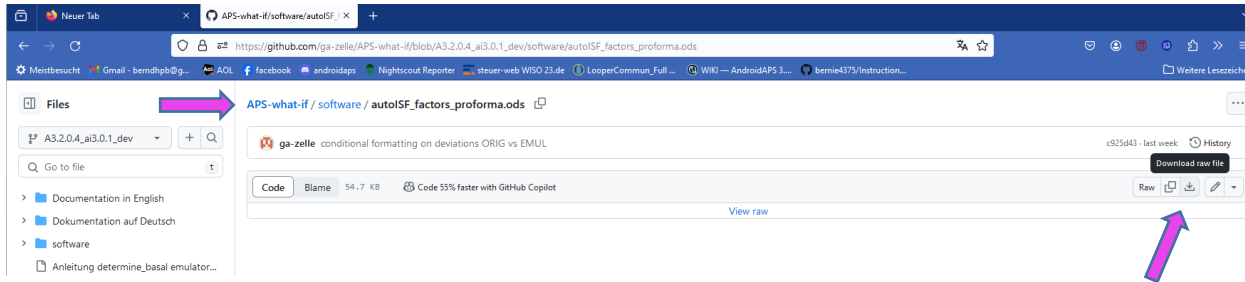
That tool is quite a bit of work to set up. Decide for yourself whether you do it, or whether you rather work with extracting the csv table into Excel (A), and work freely from there.

### 10.3.3.3 Automated extraction from tabular results

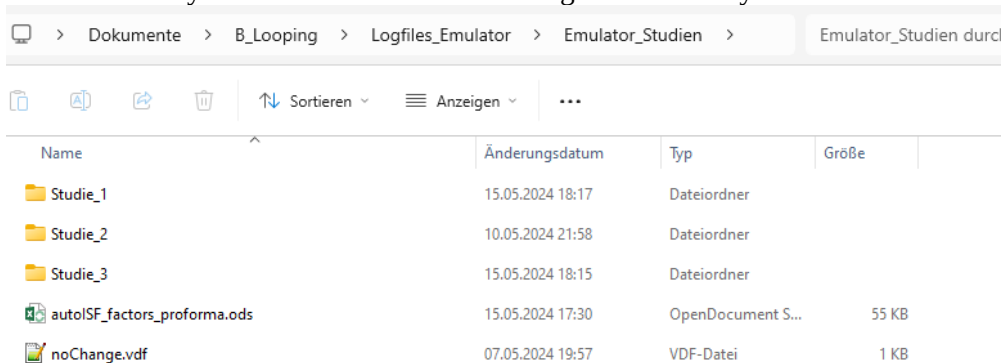
Decide for yourself, whether you rather go from the csv results table into .xls and produce what you want to see there for yourself.

With a bit of extra set-up effort (next 4 pages) you can install an adjunct tool that will always produce the nice graph for you as shown on the end of this [section 10.3.3.3](#):

1).autoISF\_factors\_proforma.ods is provided as an **extra tool** that you download from here:



Put that file on your PC one level above the single files for all your studies:



2).Now, if we want to use this tool on the two csv files of our Studie\_3 file, we must proceed as follows (for each of the two .csv files, separately):

1. Click on the .csv file and open in Libre office calculator.

2. Make sure the time column is set to US\_English:

Textimport - [2024-05-09T09-00-00Z%20.1.2\_pp\_0.8\_bgAccel.csv]

**Importieren**

Zeichensatz: Unicode (UTF-8)

Sprache: Standard - Deutsch (Deutschland)

Ab Zeile: 1

**Trennoptionen**

☐ Feste Breite ☒ Getrennt

☒ Tabulator ☐ Komma ☒ Semikolon ☐ Leerzeichen ☐ Andere

☐ Feldtrenner zusammenfassen Texttrenner: "

**Weitere Optionen**

☐ Werte in Hochkomma als Text ☒ Erweiterte Zahlenerkennung

**Feldbefehle**

Spaltentyp: US-Englisch

	Standard	US-Englisch	Standard	Standard	Standard	Standard	Standard	S
1		UTC		UNIX	bg	bg	target	
2		time		time			low	
3	id						orig	
4	0	09:01:25 Z		1715245285,9	80		90	
5	1	09:06:20 Z		1715245580,3	79	79	90	
6	2	09:11:24 Z		1715245884,2	80		90	
7	3	09:16:26 Z		1715246186,2	80		90	
8	4	09:21:21 Z		1715246482,0	86		90	

Hilfe OK Abbrechen

480

481

482 3).Now start, in Libre office calculator, the autoISF\_factors\_proforma.ods ...

483 This turns the first 30-some lines of your csv table (left side) into a form in which important effects are

484 highlighted in color, and formatting is improved:



485

486

487 Now, you want this for the entire table.

488 4).In the autoISF\_factors\_proforma table, highlight 20 or more lines (not including the first or last), and  
489 mouse right hand/insert above ...

The screenshot shows the LibreOffice Calc application window titled 'autoISF\_factors\_proforma.ods'. The spreadsheet contains a table with columns labeled A through AC. A right-click context menu is open over cell A34, displaying options such as 'Zellen formatieren...', 'Zeilenhöhe...', 'Optimale Zeilenhöhe...', 'Zeile(n) oberhalb einfügen', 'Ausgewählte Zeilen löschen', 'Zeileninhalte löschen...', 'Ausblenden', 'Einblenden', 'Ausschneiden', 'Kopieren', 'Einfügen', and 'Inhalte einfügen...'. The table data includes columns for 'id', 'ime', 'act', 'orig', 'sens', 'ISF', 'min-utes', 'dura', 'avg', 'lin.fit', 'lin.fit', 'parab', 'parab', 'parab', 'parab', 'auto', 'acce', 'bg', 'pp', 'd', and 'e'. The status bar at the bottom indicates 'Tabelle 1 / 2 | 29 Zeilen, 1024 Spalten ausgewählt | PageStyle\_2023-02-20T20.empty | Summe=49196174579,56'.

490  
491  
492 Do this as often as you need to create the number of lines that your emulated csv file comes with.  
493 If you ended up with too many lines, erase the superfluous number (any four, in the example):

autolSF\_factors\_proforma.ods - LibreOffice Calc

Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster Hilfe

Arial 10

A128:AMJ131

	A	B	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD
1				auto	final	dura		lin.fit		parab	parab	parab	parab	auto	acce	bg	pp	d
2		UTC		sens	ISF	min-	dura	min-	lin.fit	fit	fit	fit	fit	sens	ISF	ISF	ISF	ISF
3	id	ime	act	orig	orig	min- utes	avg.	min- utes	delta	correl	durat	last-Δ	next-Δ	emul	emul	emul	emul	e
121																		
122																		
123																		
124																		
125																		
126																		
127																		
128																		
129																		
130																		
131																		
132																		
133																		
134																		
135																		
136																		
137																		
138																		
139																		
140																		
141																		
142																		
143																		
144																		
145																		
146																		
147	18	11:34:21	0	1	1,28	0	149	5	8,1	0,99	20,1	10,03	11,39	1	1,28	1,06	1,16	
148	19	11:39:14	0	1	0,4	5	149	25	7,1	1	15	1,05	-4,39	1	0,38	1,06	1	
149	20	11:44:15	0	1	0,57	10	148,7	40	5,4	0,99	20	-2,41	-6,96	1	0,5	1,06	1	
150	21	11:49:16	0	1	1,11	20	146,6	10	-1,5	1	14,9	-2	-3,01	1	0,88	1,06	1	
151	22	11:54:16	0	1	1,1	25	145,7	10	-3,5	0,99	15	-4,6	-6,6	1	0,77	1,05	1	
152	23	11:59:14	0	1	0,86	5	138,5	10	-5	1	24,9	-5,53	-7	1	0,83	1,03	1	
153	24	12:04:14	0	1	1,22	10	136,7	15	-4,4	1	15	-3,4	-2,4	1	1,22	1,03	1	
154	25	12:08:45	0	1	1,21	10	136,7	5	-3	1	15	-3,4	-2,4	1	0,88	0,99	1	
155	26	12:10:37	0	1	1,02	15	136	5	1	0,99	15	0,6	3,6	1	1,63	0,99	1	
156	27	12:14:14	0	1	0,67	15	133,3	35	-3	0,96	30	-1,64	-1,07	1	0,96	0,98	1	

your\_title graphs

Tabelle 1 / 2 4 Zeilen, 1024 Spalten ausgewählt PageStyle\_2023-02-20T20.empty Summe=0 100 %

5). Then just copy it in, by selecting all data lines in the emulated csv, and pasting (paste special, values only) into box A4 of your “elonged” autoISF\_factors\_performa.ods.

6). The bottom tab “your\_title” should be re-named by you, best with day of log you analyze, and your what-if parameters (so, the name of your csv file could be put in here)

**Now you have a table with optimized lay-out that incorporates key data from both your no change AND of your investigated changes.csv files.**

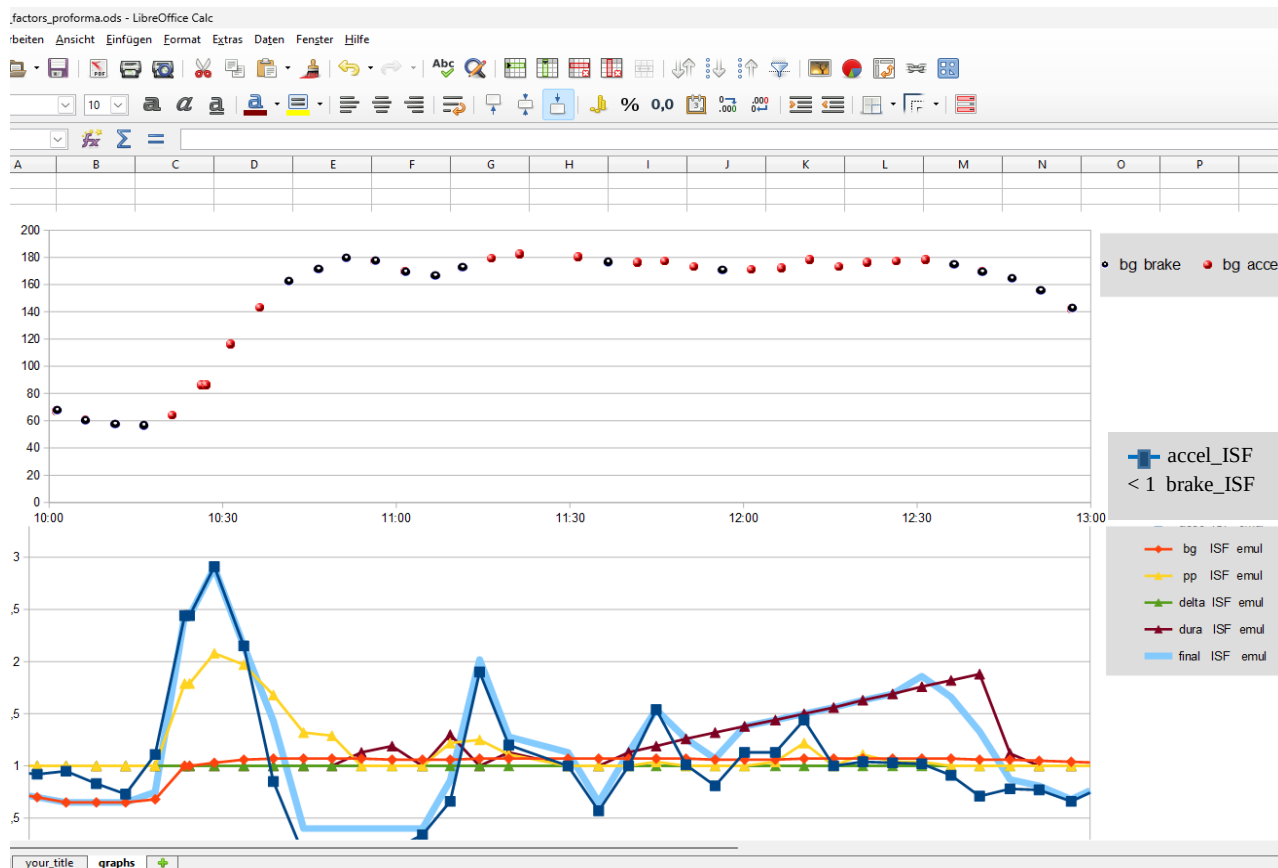
A similar table is available on the (i-)phone if you use the autoISF dev variant of iAPS or Trio (see [section 11.3](#))



507 7). A super neat extra feature is already pre-programmed, which you can see if you click on the bottom **tab**  
508 **“graphs”**.

509 The top graph is the bg curve (the actually seen bg).

510 Note that for the what-if no bg development over the time range is available. (The noChange one is  
511 also given there).



512 The bottom graph (do one for each, the noChange or the (yourChanges) case) shows the amplification factors  
513 coming from each autoISF category, and the overall resulting ISF amplification.

514 You probably have to widen the time scale (double click on the time axis, and type the desired time  
515 span (min and max UTC)(and spacing of data points, 00:30:00 or 00:15:00) into this box:  
516

X-Achse

Skalierung Positionierung Liniendiagramm Beschriftung Zahlen Schrift Sch

**Skalierung**

☐ Richtung umkehren

☐ Logarithmische Skalierung

Minimum 10:00:00 ☐ Automatisch

Maximum 15:30:00 ☐ Automatisch

Hauptintervall 00:30:00 ☐ Automatisch

Hilfsintervall Schritte 5 ☐ Automatisch

OK Abbr

517

*In the given example above, the 2.5 hours were not enough yet to analyze this 10:30 UTC (12:30 AAPs) lunch; we need to look until bg is near target (hopefully before dinner starts).*

#### 10.3.3.4 Chart coming with the emulator

In case you find the extra steps described in the preceding section “too much”, also the emulator offers one chart (the pdf offered at the bottom of the screen as shown below the “[10.3.3](#) Emulation results” headline).

First look at the initial bg rise in the noChange.pdf chart (emulation results from your noChange.vdf run), and see how bgAccel\_ISF and pp\_ISF acted, or could have acted in improved ways.

Then look into in (yourChange).pdf to see potential effects (or what other change to try). (Actually, you probably will have to go into a detailed analysis of several lines and columns of the tables as discussed in sections [10.3.3.2](#) and [10.3.3.3](#)).

Note that ideally we want FCL coverage of our entire “normal day” meal spectrum by one set of settings. So, **not much is gained if you put a lot of effort in optimizing FCL settings for one meal.**

You will need iterations. Do such analysis for **two or three very different meals** that you wish the algorithm to automatically handle. See [section 4.2/4.3](#) on how meals with very different carb loads might benefit or also suffer from too aggressive or too mild (category)\_ISF\_weights you could set.

The initial iob received might be limited by allowed SMB sizes, autoISFmax, or the (dynamic!) iobTH. You will have to look into the data table to find out about this (a quick orientation - notably regarding the light blue iobTH band, see next page - is also possible in the pdf result files you have in your project file (project file example “Studie 3” in 2<sup>nd</sup> chart under the [10.3.3](#). headline).

Only once you found OK weights for bgAccel- and pp\_ISF\_weights, does it make sense to go tune the dura\_ISF\_weight. 12:00 – 12:45 UTC in above graph, the resulting effective ISF is dominated by dura\_ISF. Just judging from the picture, a stronger weight might be worth trying. However, we really need to see the insulinRequired calculation and the further development because impatience about bringing bg values down faster too often results in hypoglycemia later.

The **noChange.pdf** is a chart that shows along the time axis (down), from right to left:

- Red: the bg curve
- Yellow: the bg target (note that I do no manual “EatingSoonTT” but for bg rises over +10 mg/dl I have an Automation that sets low TT for a couple of minutes)



555 • Light blue corridor: Left edge is set iobTH, and bandwidth +30% (would be +20% at elevated  
556 TT)

557 • Dark blue line: iob (exceeding twice the iobTH, with temp. SMB shut-off)

558 As bg did not convincingly come down enough, one could hypothesize that iobTH should be  
559 elevated. ((But, again, this would have to be confirmed also with other kinds of meals)).

560 • Thin yellow line: Insulin activity

561 • Green dotted line: ISF as would result from AAPS w/Autosens

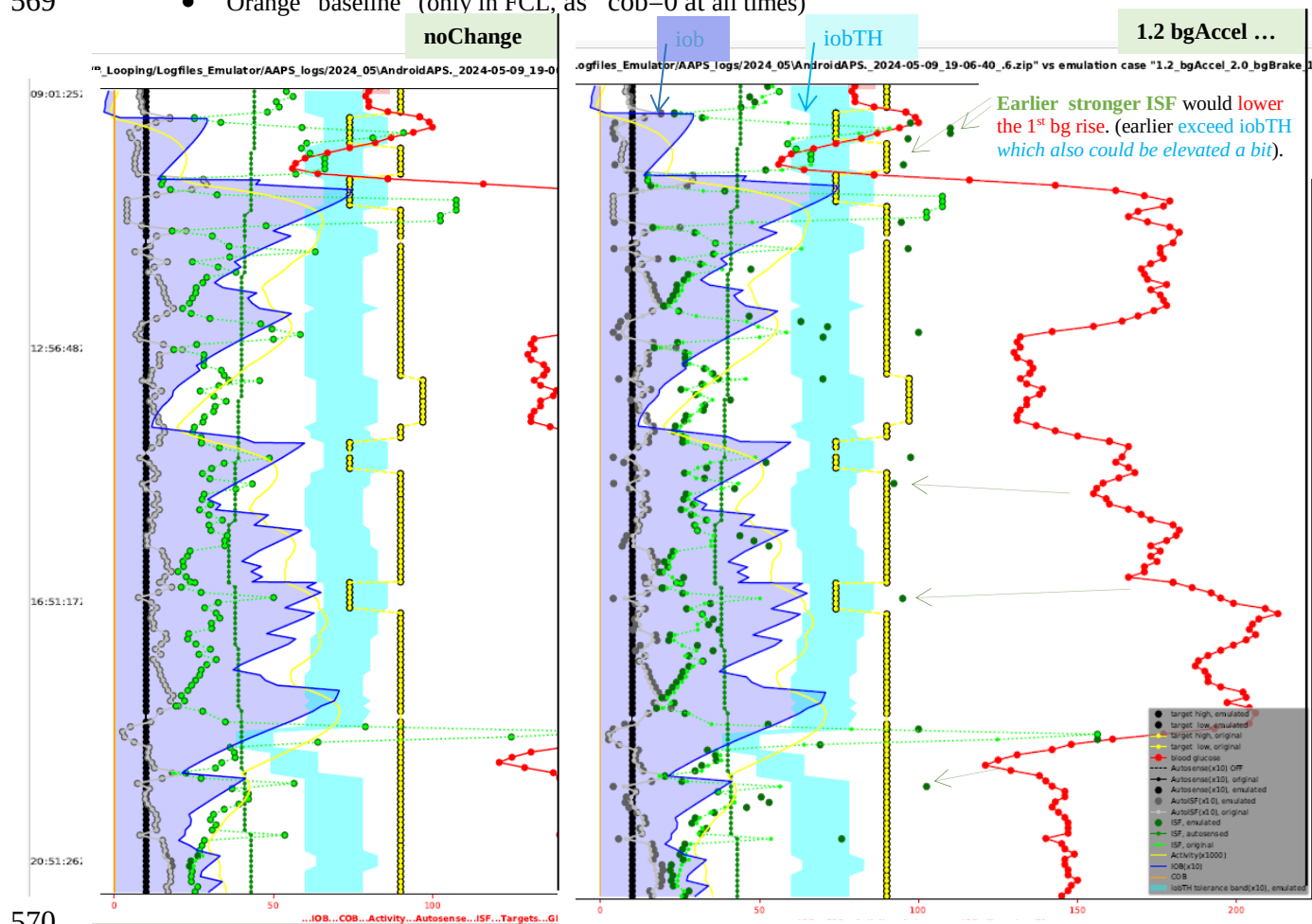
562 • **Green scatter points:** autoISF ISF no Change (lighter points) or **what-if (darker points)**

563 Foreseeably, this is the strongest difference between our noChange (left) and 120% bgAcel\_ISF\_weight  
564 (right) in the picture below. (Note the red bg curve is *both times* the really seen bg, because the what-if  
565 case only looks at each single loop decision). The first ( -> ) time the dark green dot is far to the right, this  
566 *would* get the bg down, we *would start to see* a ( <- ) bg lowering effect, shifting the red curve to the left

567 • Black line: Profile ISF

568 • Gray scatter points: ISF weakened (to the left of black line) or strengthened (to the right)

569 • Orange “baseline” (only in FCL, as cob=0 at all times)



572 Regarding the other changed parameters: Stronger dura\_ISF would suggest more insulin towards the end of  
573 plateaus; this should have helped in the 1<sup>st</sup> plateau (red curve, top right quadrant of the picture). However,  
574 same setting would have to work also on 2<sup>nd</sup> plateau; the chart cuts off there, so too early to see whether a  
575 hypo danger might result.

576 Effect from doubling the bgBrake\_ISF effect are hard to evaluate. Better probably to look in .csv tables, or  
577 run a separate emulation for that change only.

578

579 Always check for 2 or 3 kinds of your meals whether the “new” parameter settings really are on average  
580 better. (See negative example in [case study 8.2!](#))

581

582

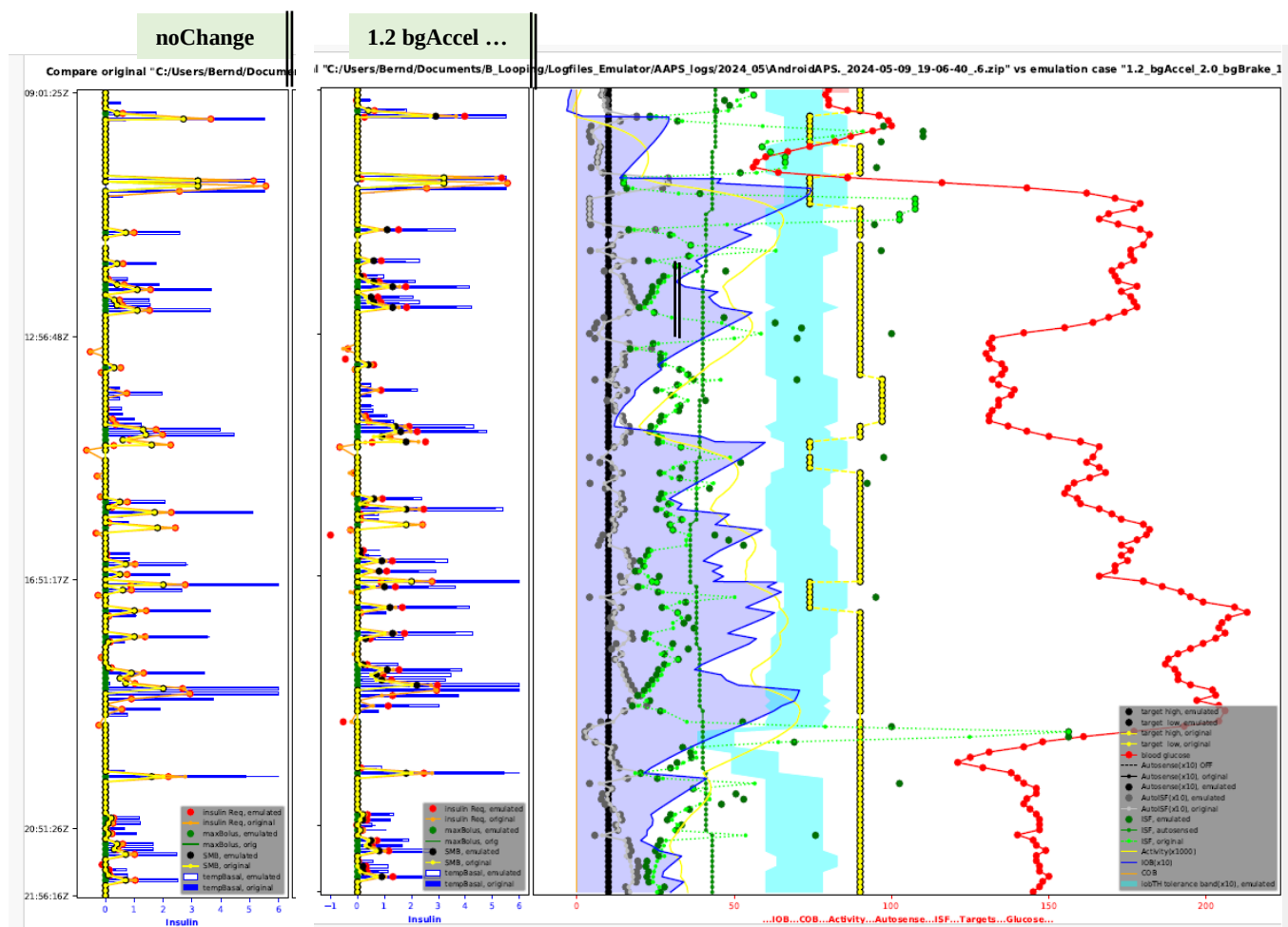
583 Part of both above shown charts (left side of each, with blue peaks) was cut out.....

584

585 (Unfinished / to be explained later) (...note: yourChanges = 1.2\_bgAccel\_2.0\_bgBrake\_1.2\_dura)

586

587



588

589

590

591

592 Please share your experiences with the emulator in Discord / Full-Closed-Looping / HOW

593 TO /\_emulate-aaps, at: <https://discord.gg/n3tD5eXExC>