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

V.3.6

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

No medical advice

research paper

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

10.2.4.5 delta table

10.3 What-if analysis

10.3.1 Define (**yourChange**).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

10.3.3,5 delta table

Available related case studies:

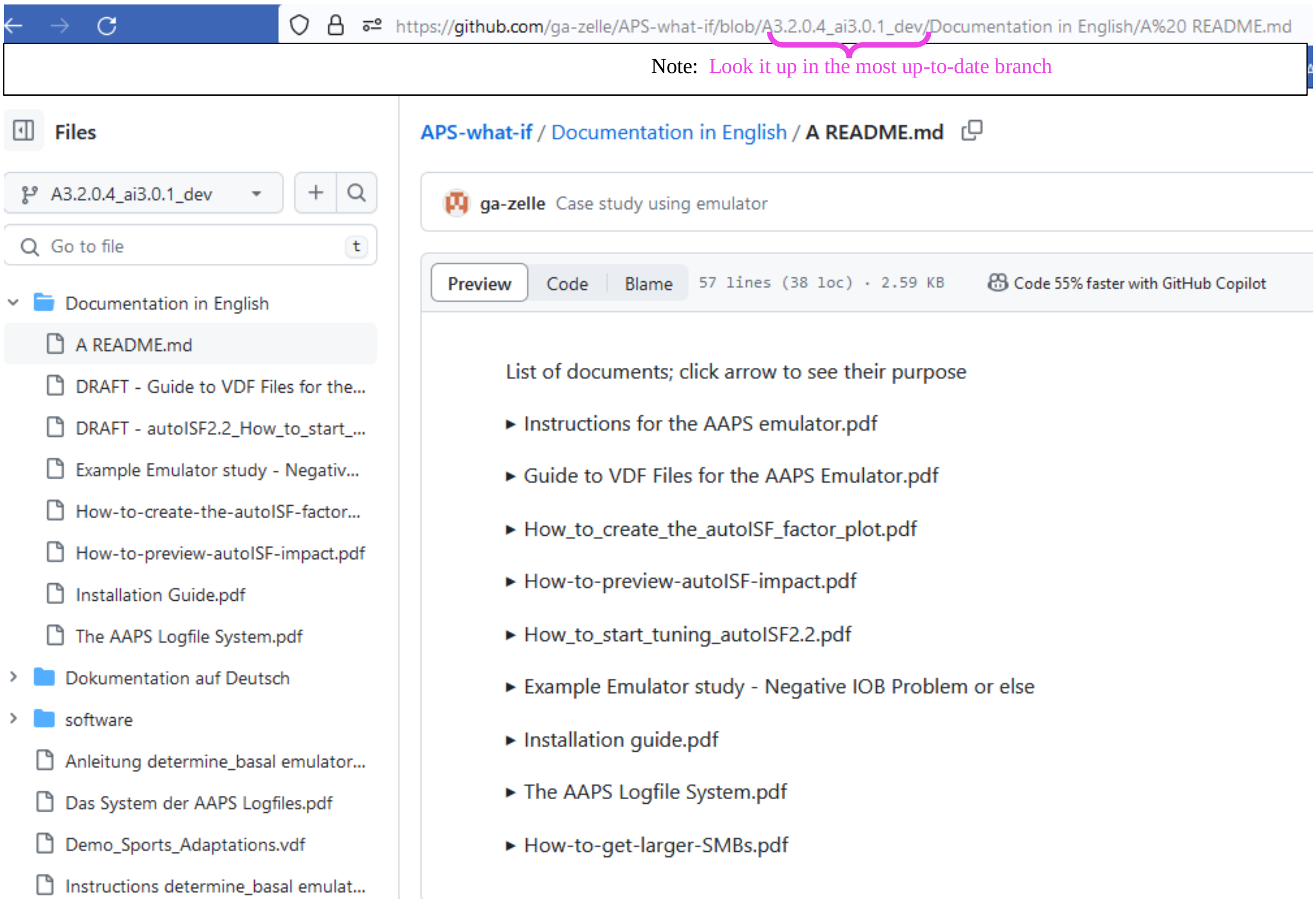
Case studies still missing:

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 in English/A%20 README.md

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

Files

A3.2.0.4_ai3.0.1_dev

Go to file

- 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 need 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-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)

10.1.1 Create your PC folder structure

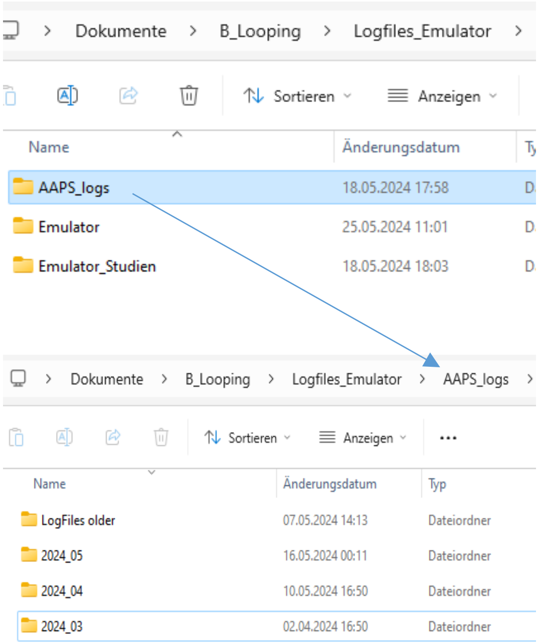
The suggested folder names and structure shown below is of course not mandatory, but only a suggestion.

On your PC, create a folder "**Logfiles_Emulator**" with 3 sub-folders: "AAPS _logs", "Emulator" and "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).

The logfiles you ALWAYS must copy-in from your phone before they get automatically erased there after x days (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.



> Dokumente > B_Looping > Logfiles_Emulator >

Name	Änderungsdatum	Ty
AAPS_logs	18.05.2024 17:58	D
Emulator	25.05.2024 11:01	D
Emulator_Studien	18.05.2024 18:03	D

> Dokumente > B_Looping > Logfiles_Emulator > AAPS_logs >

Name	Änderungsdatum	Typ
LogFiles older	07.05.2024 14:13	Dateiordner
2024_05	16.05.2024 00:11	Dateiordner
2024_04	10.05.2024 16:50	Dateiordner
2024_03	02.04.2024 16:50	Dateiordner

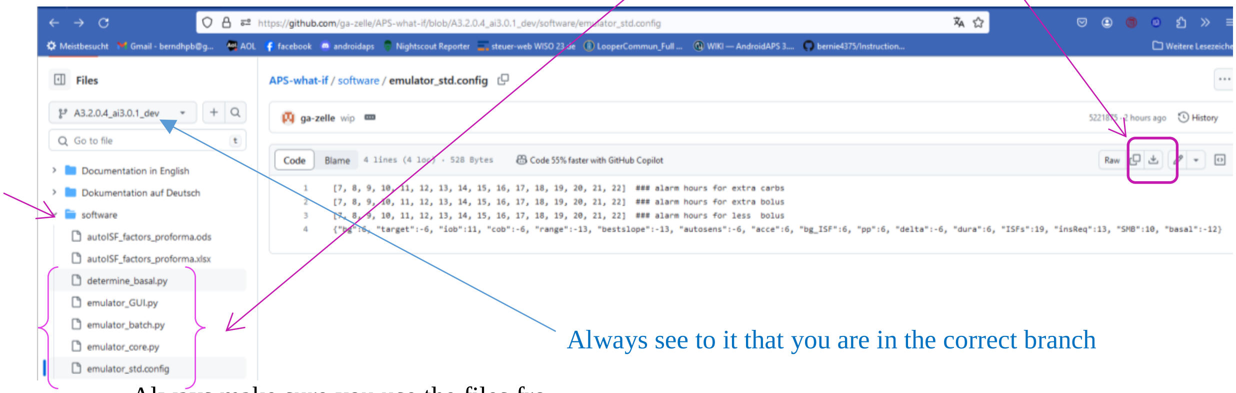
Emulator: Neighboring "AAPS_logs" is the "Emulator" folder into which most downloads from the developer's repo will go in section 10.1.2

Emulator_studies is a folder, where, for now, you should provide some **sub-folders** "Study_1", "Study_2" … **Study_n**. Later, when you use the emulator, you will use these "addresses" for the program to dump results from the emulation into. Additionally you will probably put related AAPS screenshots and Nightscout.Reporter or xDrip/Statistics charts into each project folder to support analysis.

10.1.2 Downloads

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

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



https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1_dev/software/emulator_std.config

APS-what-if / software / emulator_std.config

A3.2.0.4_ai3.0.1_dev

ga-zelle wip

4 lines (4 loc) · 528 Bytes

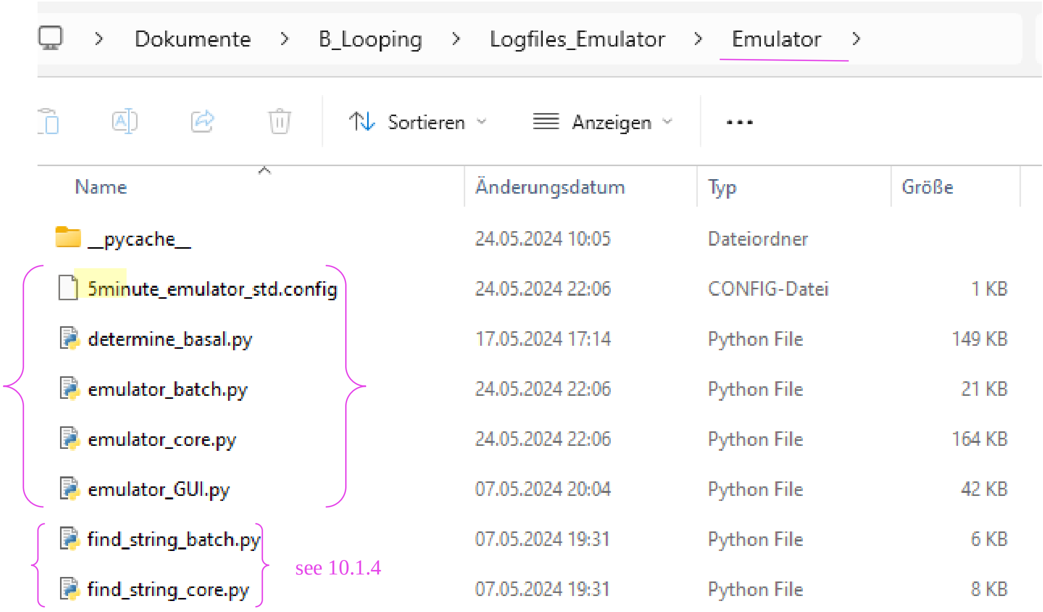
```
[7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22]  ### alarm hours for extra carbs
[7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22]  ### alarm hours for extra bolus
[7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22]  ### alarm hours for less  bolus
{"bg":6, "target":-6, "iob":11, "cob":-6, "range":-13, "bestslope":-13, "autosens":-6, "acce":6, "bg_ISF":6, "pp":6, "delta":-6, "dura":6, "ISFs":19, "insReq":13, "SMB":10, "basal":-12}
```

Always see to it that you are in the correct branch

Always make sure you use the files fro ı (in 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 <u>and also</u> 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:



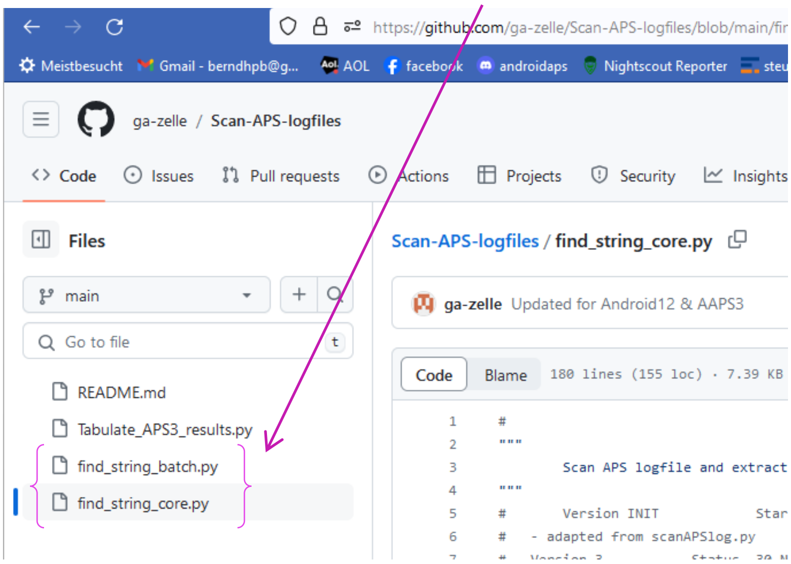
Dokumente > B_Looping > Logfiles_Emulator > Emulator >

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

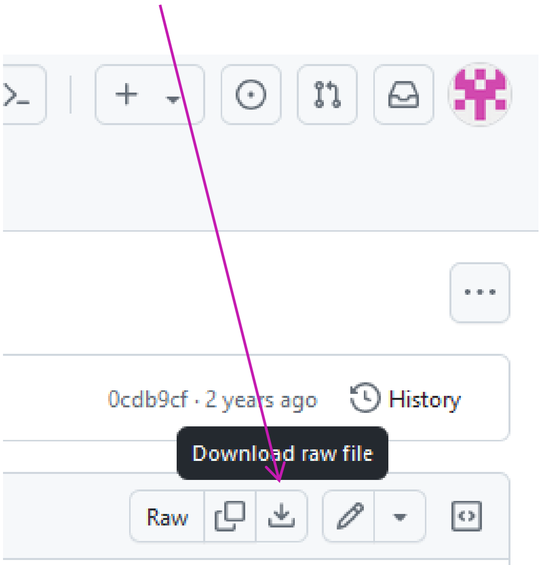
see 10.1.4

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



```
#
"""
        Scan APS logfile and extract
"""
#       Version INIT              Star
#   - adapted from scanAPSlog.py
```



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

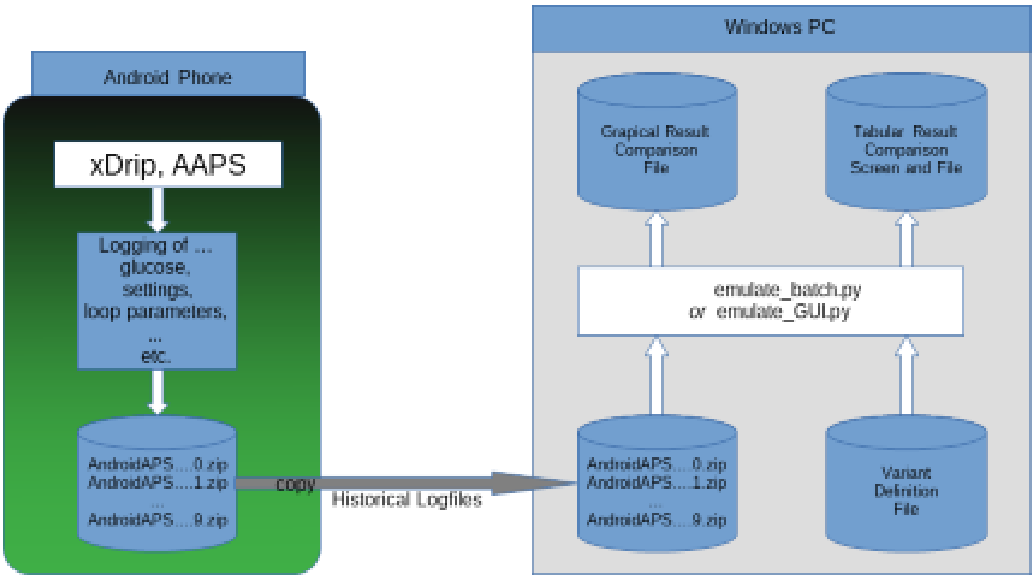
10.1.4 Other software requirements

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

10.2.1 Set up a "no change" .vdf file.

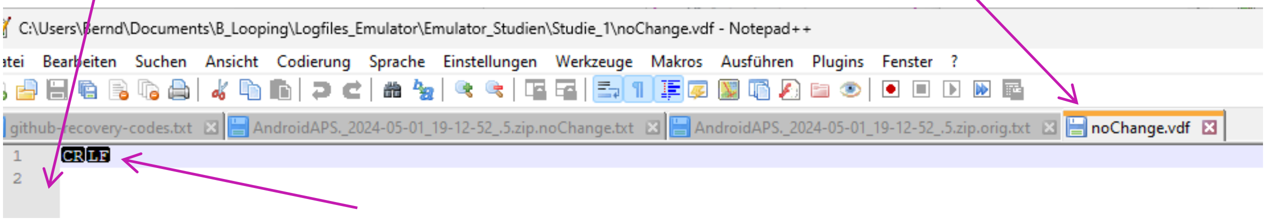
1).To do this, just open **Notepad++** (from list of all programs on your PC).

2). Name your file "noChange.vdf".

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

Note: for "what-if" analysis, entries will be made (in a second .vdf later, see section 10.3)

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


C:\Users\Bernd\Documents\B_Looping\Logfiles_Emulator\Emulator_Studien\Studie_1\noChange.vdf - Notepad++
atei Bearbeiten Suchen Ansicht Codierung Sprache Einstellungen Werkzeuge Makros Ausführen Plugins Fenster ?
github-recovery-codes.txt
AndroidAPS._2024-05-01_19-12-52_.5.zip.noChange.txt
AndroidAPS._2024-05-01_19-12-52_.5.zip.orig.txt
noChange.vdf

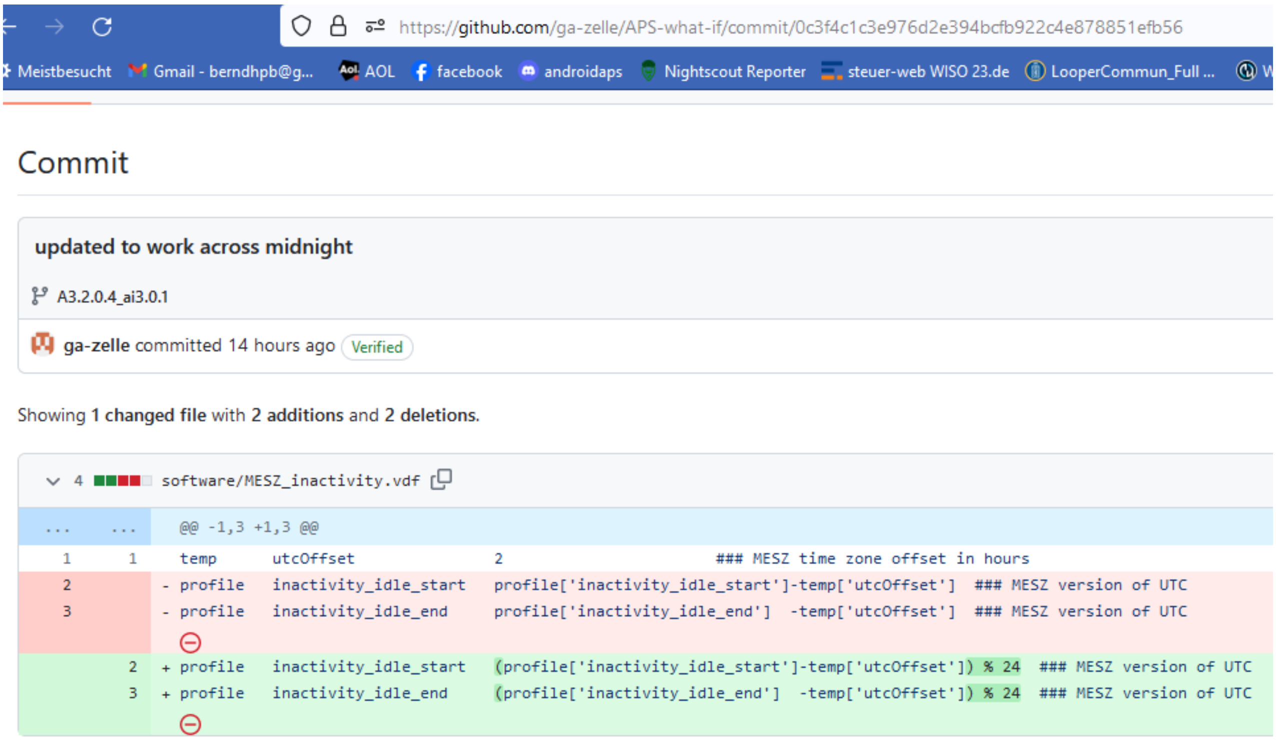
```
CRLF

```

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

3).Temp. bug fix needed if you use Activity Monitor (and emulator)

The temporary bugfix regards the time axis, in context with set sleep time set by you In the Activity Monitor.

All your **vdf** files need 3 additional resp. corrected lines for the set time:



https://github.com/ga-zelle/APS-what-if/commit/0c3f4c1c3e976d2e394bcfb922c4e878851efb56

Commit

updated to work across midnight

A3.2.0.4_ai3.0.1

ga-zelle committed 14 hours ago Verified

Showing **1 changed file** with **2 additions** and **2 deletions**.

```
4  software/MESZ_inactivity.vdf

...   ...   @@ -1,3 +1,3 @@
1     1     temp       utcOffset               2                         ### MESZ time zone offset in hours
2         - profile    inactivity_idle_start   profile['inactivity_idle_start']-temp['utcOffset']  ### MESZ version of UTC
3         - profile    inactivity_idle_end     profile['inactivity_idle_end']  -temp['utcOffset']  ### MESZ version of UTC
      2   + profile    inactivity_idle_start   (profile['inactivity_idle_start']-temp['utcOffset']) % 24  ### MESZ version of UTC
      3   + profile    inactivity_idle_end     (profile['inactivity_idle_end']  -temp['utcOffset']) % 24  ### MESZ version of UTC
```

The “2” in line 1 stands for Central EU Summer Time (CEST) deviation from UTC.

Put in the **applicable time difference** to UTC **from where you are** (e.g. it might be “-4” for East Coast US etc)

In the red marked area, lines 2 and 3, texts must be corrected as highlighted in intensive green: (…..) % 24.   If your current vdf has no text like in the red field, just add line 1 *plus* green lines 2 and 3, e.g. at the bottom.

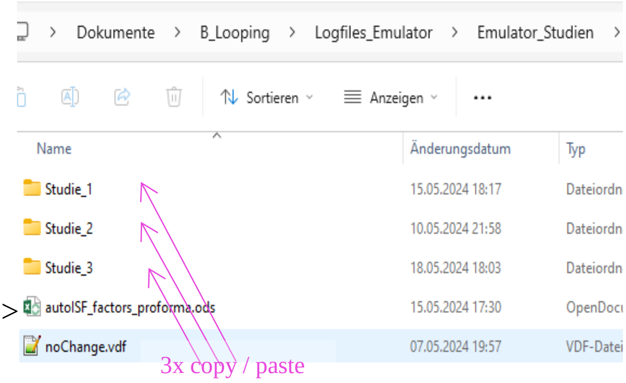
This temporary bugfix only affects the emulator evaluation of data around Activity Monitor sleep time switches. Whether you do something about it or not will in no case have any effect on your actual loop.

After the next autoISF version update, it likely will be no longer needed.

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

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

See section 10.3.3.3. regarding this ->



Dokumente > B_Looping > Logfiles_Emulator > Emulator_Studien >
Sortieren
Anzeigen
...

Name	Änderungsdatum	Typ
Studie_1	15.05.2024 18:17	Dateiordn
Studie_2	10.05.2024 21:58	Dateiordn
Studie_3	18.05.2024 18:03	Dateiordn
autoISF_factors_proforma.ods	15.05.2024 17:30	OpenDocu
noChange.vdf	07.05.2024 19:57	VDF-Datei

3x copy / paste

10.2.2 Locate relevant logfiles and prepare the Study_n folder

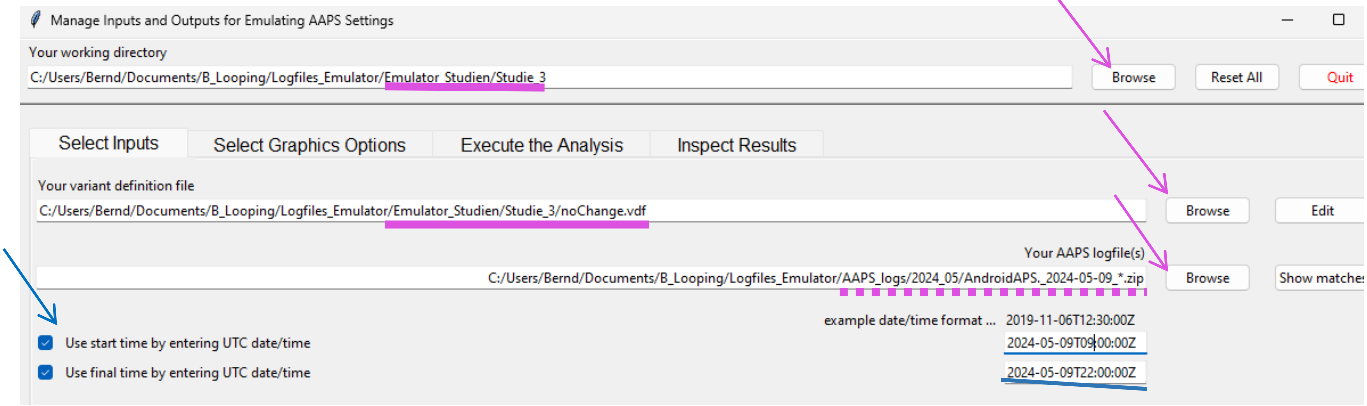
1). Make sure you have the AAPS logfiles that you want to analyze in your ~~Emulator_Studies / Study_n~~ "AAPS_logs folder"

2). In your "Emulator_Studies" folder, create (or use a prepared) "Study_n" sub-folder, with a copied-in (not: moved!) noChange.vdf (It must be *in all* Study_n files).

10.2.3 Prepare your emulator run for Study_n

Now go to your PC **desktop**, and start the emulator by **just pressing the button "emulator start"** that you installed in step 10.1.3

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



Manage Inputs and Outputs for Emulating AAPS Settings

Your working directory: C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3

Your variant definition file: C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/noChange.vdf

Your AAPS logfile(s): C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/AAPS_logs/2024_05/AndroidAPS._2024-05-09_*.zip

example date/time format ... 2019-11-06T12:30:00Z

☑ Use start time by entering UTC date/time: 2024-05-09T09:00:00Z

☑ Use final time by entering UTC date/time: 2024-05-09T22:00:00Z

a) The top box marks the path to your current emulator project ("Studie_3" is my "Study_n" where I want to store results)

b) The middle box marks the path to your current vdf (what kind of analysis; here: "…noChange.vdf" = *read-only*. (For *what-if,* see section 10.3)

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

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

- Replace the time with an asterix * (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.

- As I wanted to look at 11 am –midnight (for lunch and dinner related data), I :
  - clicked the bottom left two boxes

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

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

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

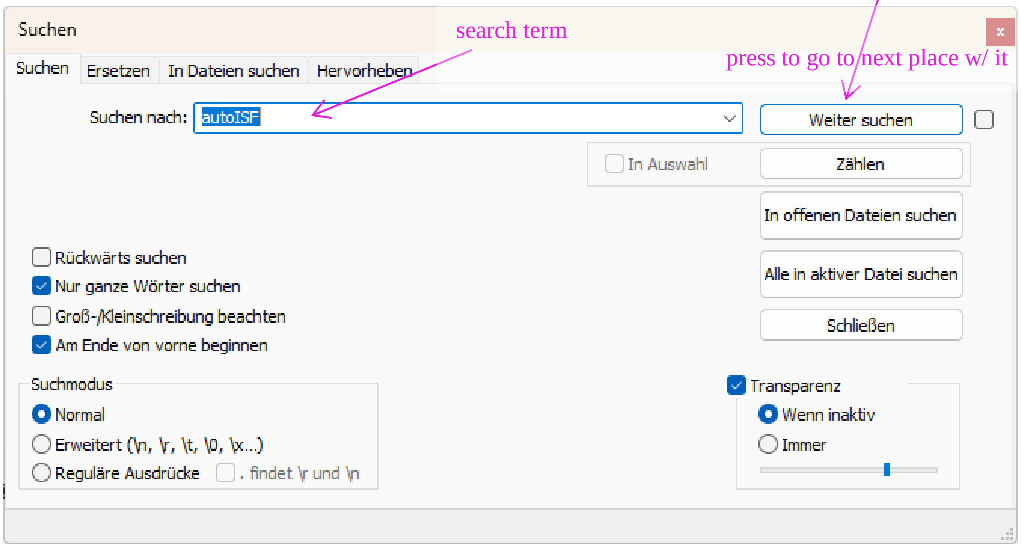
After a short moment results should show up, which you can look into in a couple of ways. First you could have a quick look into the **.log** file to see whether the run had errors (see section 3.8)

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.



search term

press to go to next place w/ it

Suchen nach: autoISF

☐ In Auswahl

☐ Rückwärts suchen
☑ Nur ganze Wörter suchen
☐ Groß-/Kleinschreibung beachten
☑ Am Ende von vorne beginnen

Suchmodus
● Normal
○ Erweitert (\n, \r, \t, \0, \x...)
○ Reguläre Ausdrücke ☐ . findet \r und \n

☑ Transparenz
● Wenn inaktiv
○ Immer

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.

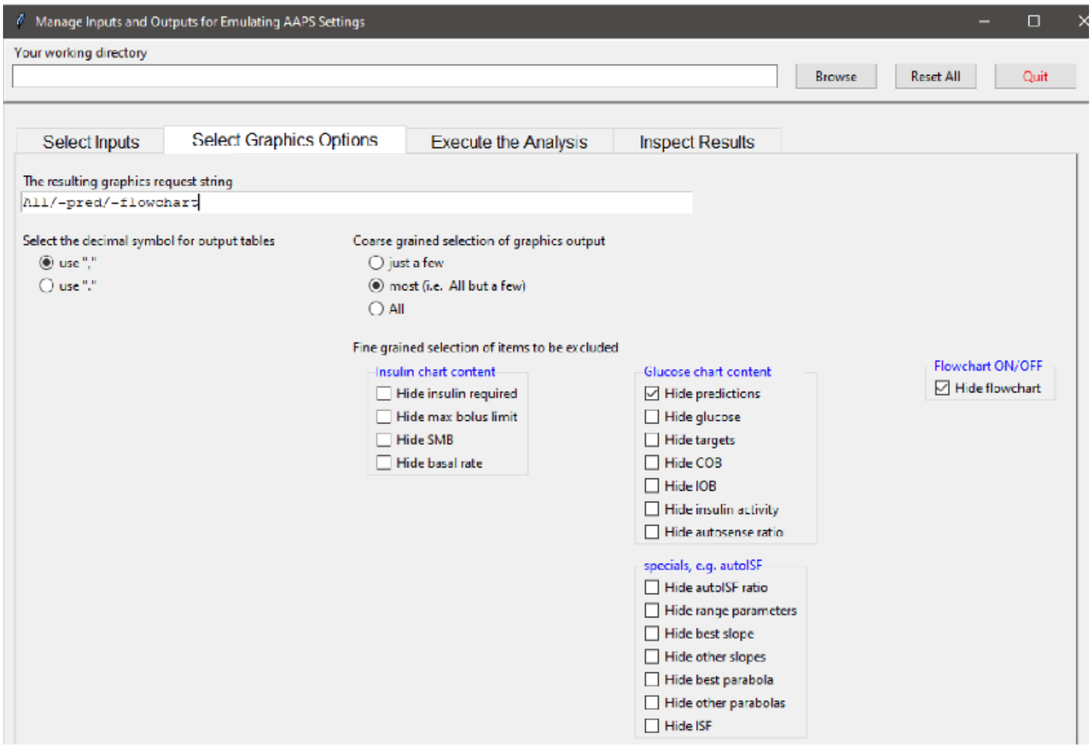
It is a vast table, so you may want to reduce it to something more "digestible", either after transfer to your standard calculation program (next section 10.2.4.3). Or you can also make settings to suppress information you are usually not interested in (or do not know how to interpret, anyways) under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

First, select your preferred way of outputting decimals (point or comma).

Then select whether you want "All" possible outputs in the graph, or "Most" = all except those you tick "off" in the boxes for each output parameter.

> In case you would use "Some/just a few", you would have to tick those few you that do want to see, by ticking the corresponding boxes.

Recommendation is to look at (nearly) everything offered (as your default setting that you can leave untouched in most of your emulator runs):



Your working directory

The resulting graphics request string
`All/-pred/-flowchart`

Select the decimal symbol for output tables
● use ","
○ use "."

Coarse grained selection of graphics output
○ just a few
● most (i.e. All but a few)
○ All

Fine grained selection of items to be excluded

Insulin chart content
□ Hide insulin required
□ Hide max bolus limit
□ Hide SMB
□ Hide basal rate

Glucose chart content
☑ Hide predictions
□ Hide glucose
□ Hide targets
□ Hide COB
□ Hide IOB
□ Hide insulin activity
□ Hide autosense ratio

specials, e.g. autoISF
□ Hide autoISF ratio
□ Hide range parameters
□ Hide best slope
□ Hide other slopes
□ Hide best parabola
□ Hide other parabolas
□ Hide ISF

Flowchart ON/OFF
☑ Hide flowchart

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

10.2.4.3 Analysis of the **noChange.csv** table in Excel or LibreOffice calc.

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

- add right next to the UTC (Unix Time Code) your corresponding "AAPS **time"**

For instance, adding +2/24 translates the UTC column into central European summer time column next to it (where currently a row of Z stands). Likewise, subtract like -5/24 from UTC for an US East Coast time scale.
(Fun fact: Our oref loop stubbornly works on UTC, un-impressed by our folly to jump twice a year into or out of a local summer time, or to travel across time zones. If some data get lost in translation there, it is only to us, with our stupid time change. For the loop, its database (e.g. on insulin activity) remains unambiguously intact).

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

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

  That way, "boxes" (data fields) retain their original position in tables

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

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

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

In section 10.3.4 we present an extra tool that does a standardized table reduction and color marking for you!

You may be able to formulate a hypothesis or two, what settings (…ISF_weights, iobTH%, SMB_range_extention, autoISFmax …) should be changed for improvement (then go to 10.3)

10.2.4.4.. Graph **noChange.pdf**

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

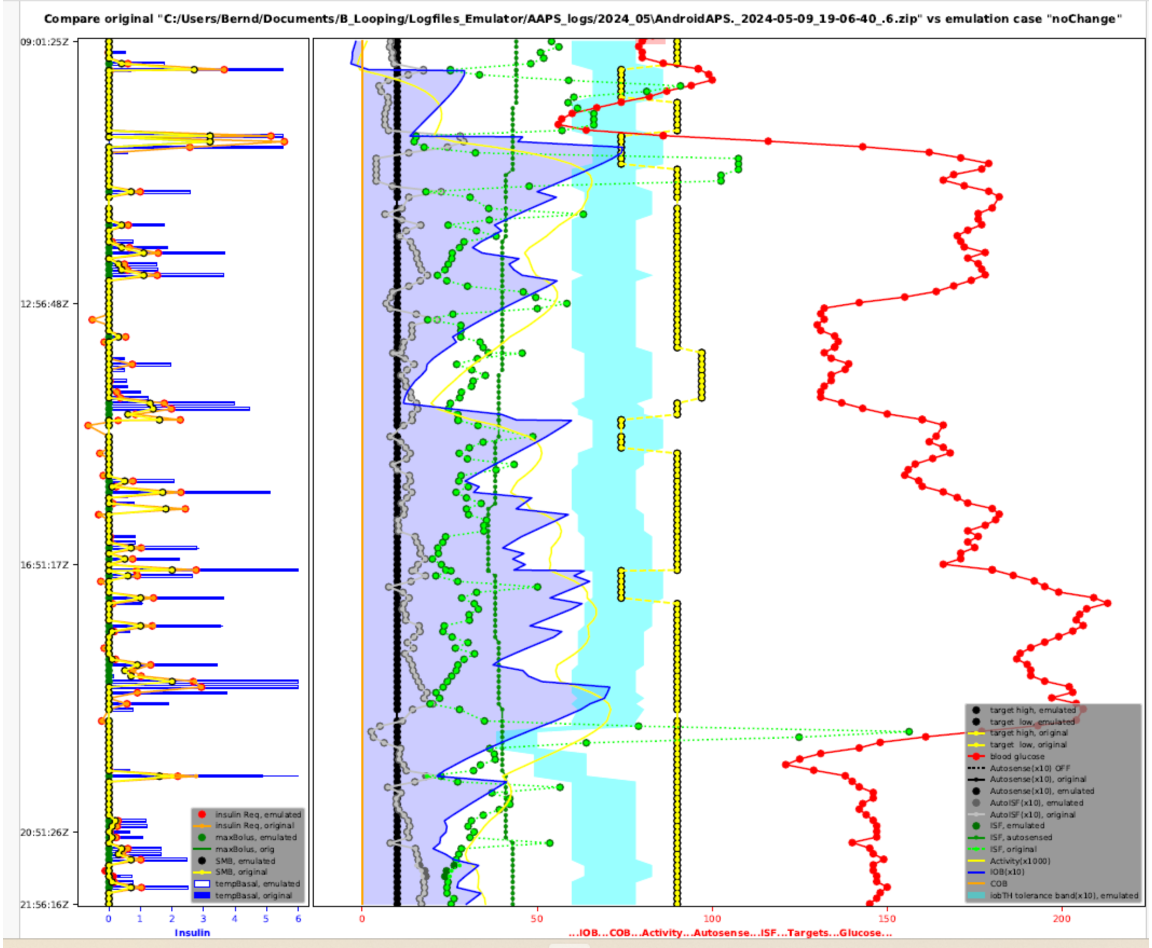
This **noChange.pdf** is a chart that shows along the time axis (down), from right to left:

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

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

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

- 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

10.2.4.5 delta table

In case you want to analyze delta, short and long average deltas, you could do some of that just using your .xls extract from .csv (see section 10.2.4.3).

There is also an extra sheet provided, “delta” (That works only if you have your **emulator_core.py** updated to the one from 02June2024 or newer). Looking into that delta sheet one makes not too much sense when you are FCL looping, unless you are about to define personal Automations that involve Conditions using these glucose curve characteristics

- This is definitely core to FCL using Automations (section 13.1).
- But also in your autoISF FCL you might have an Automation in mind, wondering which Conditions involving any of the deltas would be good to have.

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:.

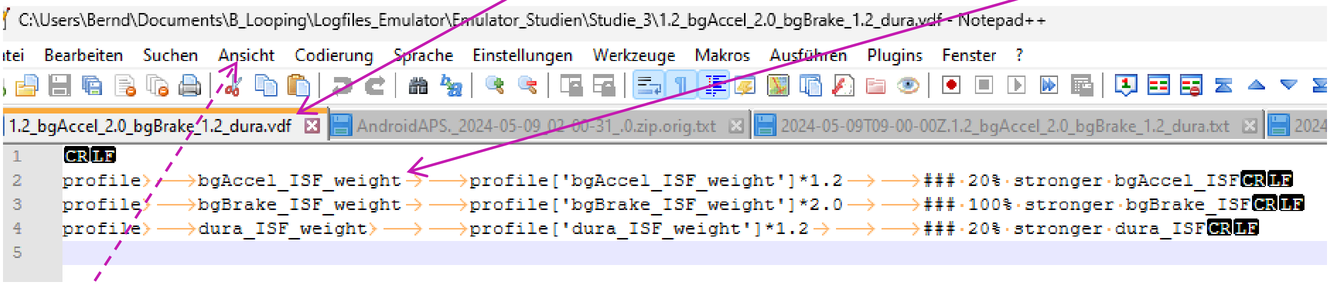
  Alternatively you can also take another pre-existing vdf file, copy it into your current project , edit as desired, and give it a new name (re-name it)

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 (yourChange) .vdf should look like something like this:


C:\Users\Bernd\Documents\B_Looping\Logfiles_Emulator\Emulator_Studien\Studie_3\1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf - Notepad++
itei
Bearbeiten
Suchen
Ansicht
Codierung
Sprache
Einstellungen
Werkzeuge
Makros
Ausführen
Plugins
Fenster
?
1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf
AndroidAPS._2024-05-09_02-00-31_.0.zip.orig.txt
2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.txt
2024

```
CRLF
profile	bgAccel_ISF_weight	profile['bgAccel_ISF_weight']*1.2		### 20% stronger bgAccel_ISFCRLF
profile	bgBrake_ISF_weight	profile['bgBrake_ISF_weight']*2.0		### 100% stronger bgBrake_ISFCRLF
profile	dura_ISF_weight		profile['dura_ISF_weight']*1.2		### 20% stronger dura_ISFCRLF

```

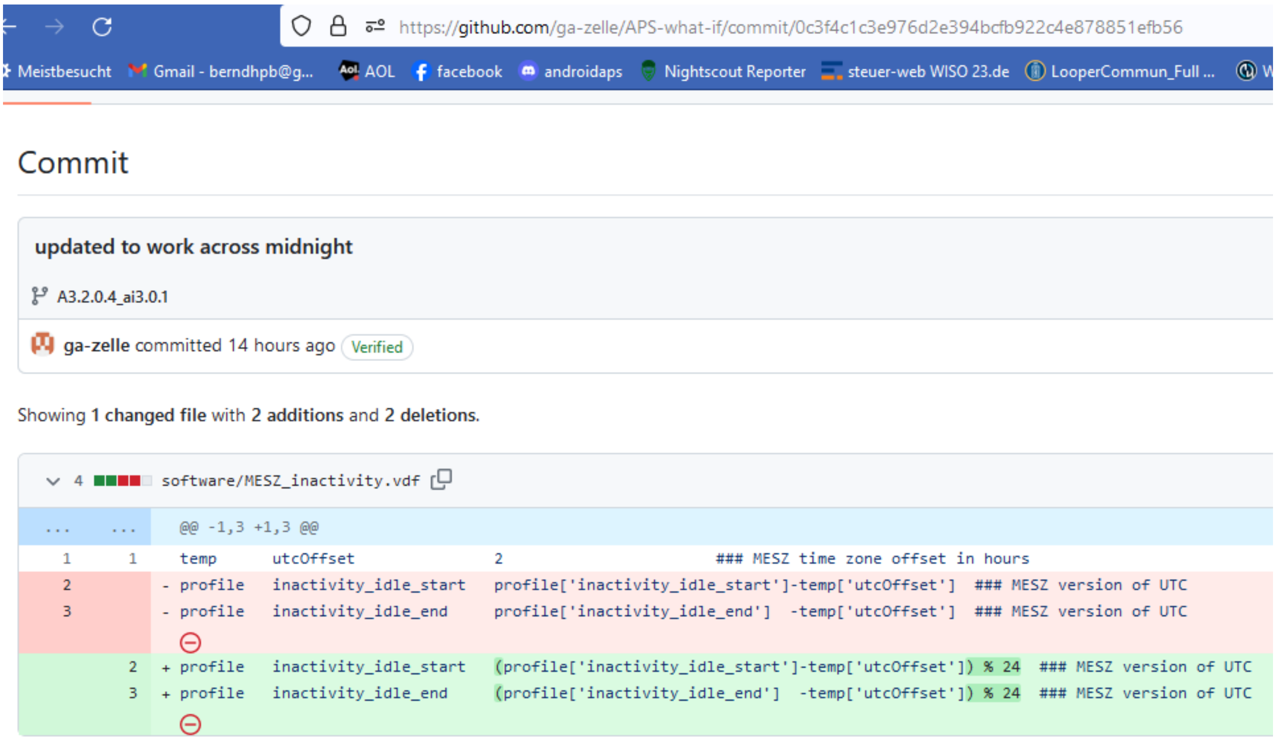
Via view/ show/hide symbols (CR , LF, tabs …), you can have different looks, see other example two pages down.

3). Name your vdf (in example below: 1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf) …

4). Store this (yourChange).vdf 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:)

5). Temp. bug fix needed in step 2, or now, as extra step, if you use Activity Monitor: It regards the time axis in context with set sleep time.in Activity Monitor (from: Github/ga-zelle/APS-what-if …).

All your **vdf** files need 3 additional resp. corrected lines for the set time:



Commit

updated to work across midnight

A3.2.0.4_ai3.0.1

ga-zelle committed 14 hours ago Verified

Showing 1 changed file with 2 additions and 2 deletions.

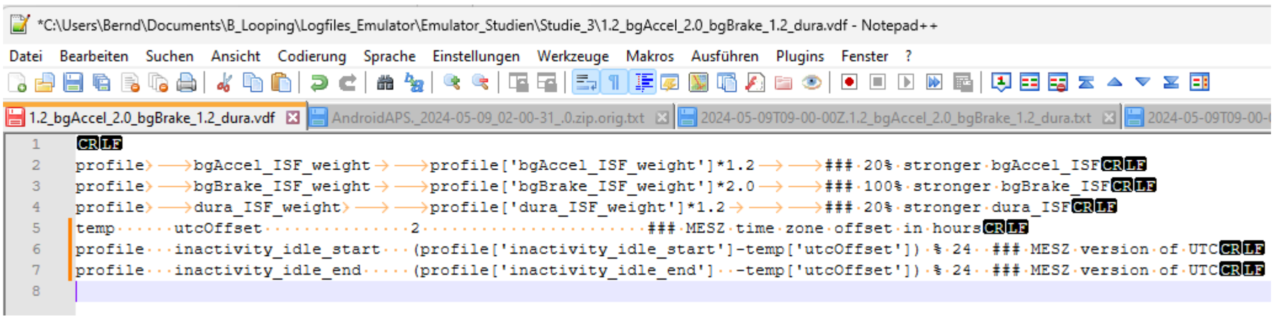
```
4 software/MESZ_inactivity.vdf
@@ -1,3 +1,3 @@
1  1    temp     utcOffset              2                                          ### MESZ time zone offset in hours
2     - profile  inactivity_idle_start  profile['inactivity_idle_start']-temp['utcOffset']   ### MESZ version of UTC
3     - profile  inactivity_idle_end    profile['inactivity_idle_end']  -temp['utcOffset']   ### MESZ version of UTC
   2  + profile  inactivity_idle_start  (profile['inactivity_idle_start']-temp['utcOffset']) % 24   ### MESZ version of UTC
   3  + profile  inactivity_idle_end    (profile['inactivity_idle_end']  -temp['utcOffset']) % 24   ### MESZ version of UTC
```

The “2” in line 1 stands for Central EU Summer Time (CEST) deviation from UTC.

Put in the applicable time difference to UTC from where you are (e.g. it might be “-4” for East Coast US etc)

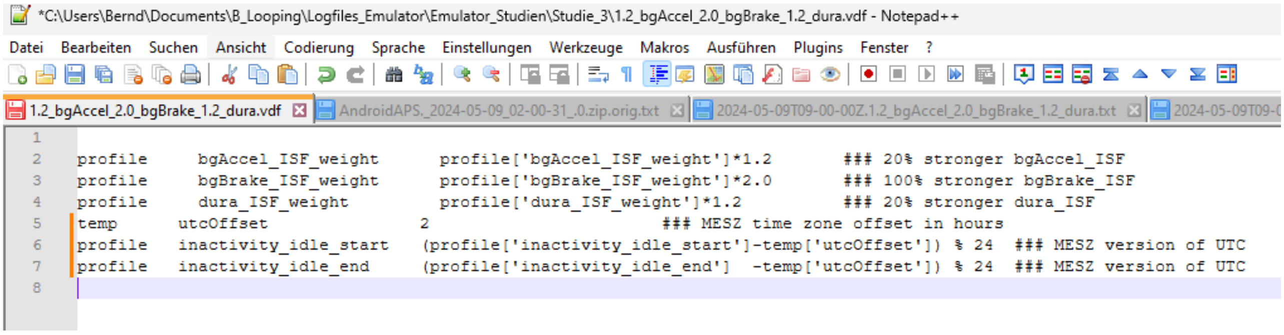
In the red marked area, lines 2 and 3, texts must be corrected as highlighted in intensive green: (…..) % 24. If your current vdf has no text like in the red field, just add line 1 plus green lines 2 and 3.

The (yourChange) .vdf from 2 pages above should now look like something like this:


*C:\Users\Bernd\Documents\B_Looping\Logfiles_Emulator\Emulator_Studien\Studie_3\1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf - Notepad++
Datei
Bearbeiten
Suchen
Ansicht
Codierung
Sprache
Einstellungen
Werkzeuge
Makros
Ausführen
Plugins
Fenster
?
1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf
AndroidAPS._2024-05-09_02-00-31_.0.zip.orig.txt
2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.txt
2024-05-09T09-00-

```
CRLF
profile→	bgAccel_ISF_weight→	profile['bgAccel_ISF_weight']*1.2→	→	###·20%·stronger·bgAccel_ISFCRLF
profile→	bgBrake_ISF_weight→	profile['bgBrake_ISF_weight']*2.0→	→	###·100%·stronger·bgBrake_ISFCRLF
profile→	dura_ISF_weight→	→	profile['dura_ISF_weight']*1.2→	→	→	###·20%·stronger·dura_ISFCRLF
temp······utcOffset···············2·······················###·MESZ·time·zone·offset·in·hoursCRLF
profile···inactivity_idle_start···(profile['inactivity_idle_start']-temp['utcOffset'])·%·24··###·MESZ·version·of·UTCCRLF
profile···inactivity_idle_end·····(profile['inactivity_idle_end']··-temp['utcOffset'])·%·24··###·MESZ·version·of·UTCCRLF
```

Or, with settings/view/remove symbols (for line feed, tab etc) it can also look like this:


*C:\Users\Bernd\Documents\B_Looping\Logfiles_Emulator\Emulator_Studien\Studie_3\1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf - Notepad++
Datei
Bearbeiten
Suchen
Ansicht
Codierung
Sprache
Einstellungen
Werkzeuge
Makros
Ausführen
Plugins
Fenster
?
1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf
AndroidAPS._2024-05-09_02-00-31_.0.zip.orig.txt
2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.txt
2024-05-09T09-0

```

profile     bgAccel_ISF_weight      profile['bgAccel_ISF_weight']*1.2       ### 20% stronger bgAccel_ISF
profile     bgBrake_ISF_weight      profile['bgBrake_ISF_weight']*2.0       ### 100% stronger bgBrake_ISF
profile     dura_ISF_weight         profile['dura_ISF_weight']*1.2          ### 20% stronger dura_ISF
temp      utcOffset               2                       ### MESZ time zone offset in hours
profile   inactivity_idle_start   (profile['inactivity_idle_start']-temp['utcOffset']) % 24  ### MESZ version of UTC
profile   inactivity_idle_end     (profile['inactivity_idle_end']  -temp['utcOffset']) % 24  ### MESZ version of UTC
```

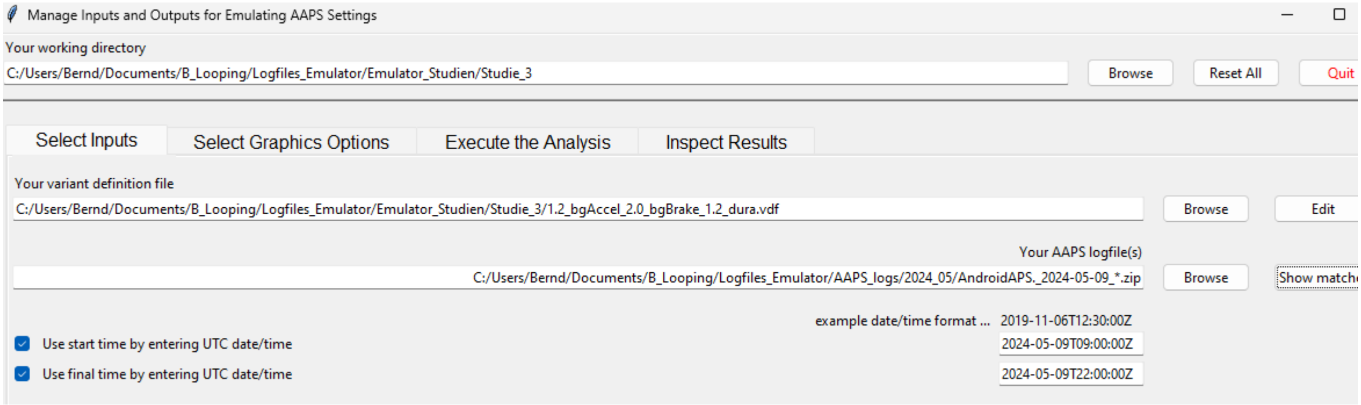
This temporary bugfix only affects the emulator evaluation of data around Activity Monitor sleep time switches. Whether you do something about it or not will in no case have any effect on your actual loop. After the next autoISF version update, it likely will be no longer needed.

10.3.2 Run the emulator with (yourChange).vdf

The “what-if” emulator run is done the same way as you did the noChange.vdf run (section 10.2), which had no (yourChange).vdf on bord

=> No surprise, running an emulation with only that noChange.vdf, yields same result in emul columns as is orig columns. - However, now :

The **(yourChange).vdf** must be loaded into the 2nd input field, where formerly you had the noChange.vdf.:



Manage Inputs and Outputs for Emulating AAPS Settings

Your working directory
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3

Your variant definition file
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf

Your AAPS logfile(s)
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/AAPS_logs/2024_05/AndroidAPS._2024-05-09_*.zip

example date/time format ... 2019-11-06T12:30:00Z

☑ Use start time by entering UTC date/time 2024-05-09T09:00:00Z

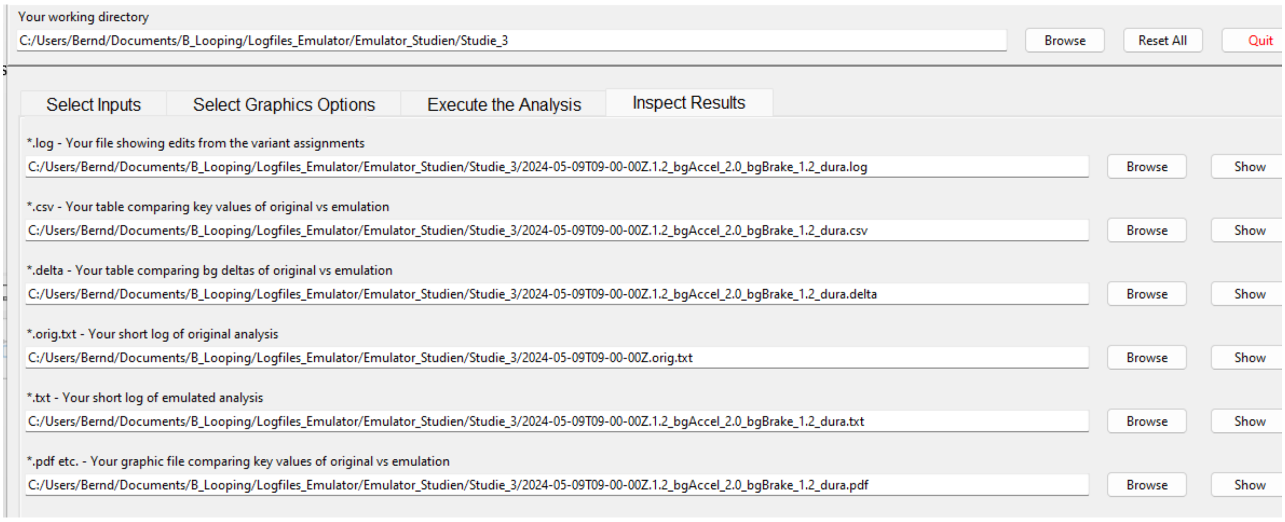
☑ Use final time by entering UTC date/time 2024-05-09T22:00:00Z

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

- Browse in your Windows Explorer to any logfile from the desired day (2024-05-09 in above example)
- Replace the time with an asterix * (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.
- As I wanted to look at 11 am –midnight for lunch and dinner related data, I :
    - clicked the bottom left two boxes
    - copied the date 2024-05-09 over the default date in the bottom right two data fields
    - 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).

After making these entries, press Execute the Analysis, (evtl also Clear old Data) and then press Run Emulation, you can look the results up under “Inspect Results”. First you could have a quick look into the **.log** file to see whether the run had errors (see section 3.)

10.3.3 Emulation results



Your working directory
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3

*.log - Your file showing edits from the variant assignments
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.log

*.csv - Your table comparing key values of original vs emulation
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.csv

*.delta - Your table comparing bg deltas of original vs emulation
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.delta

*.orig.txt - Your short log of original analysis
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/2024-05-09T09-00-00Z.orig.txt

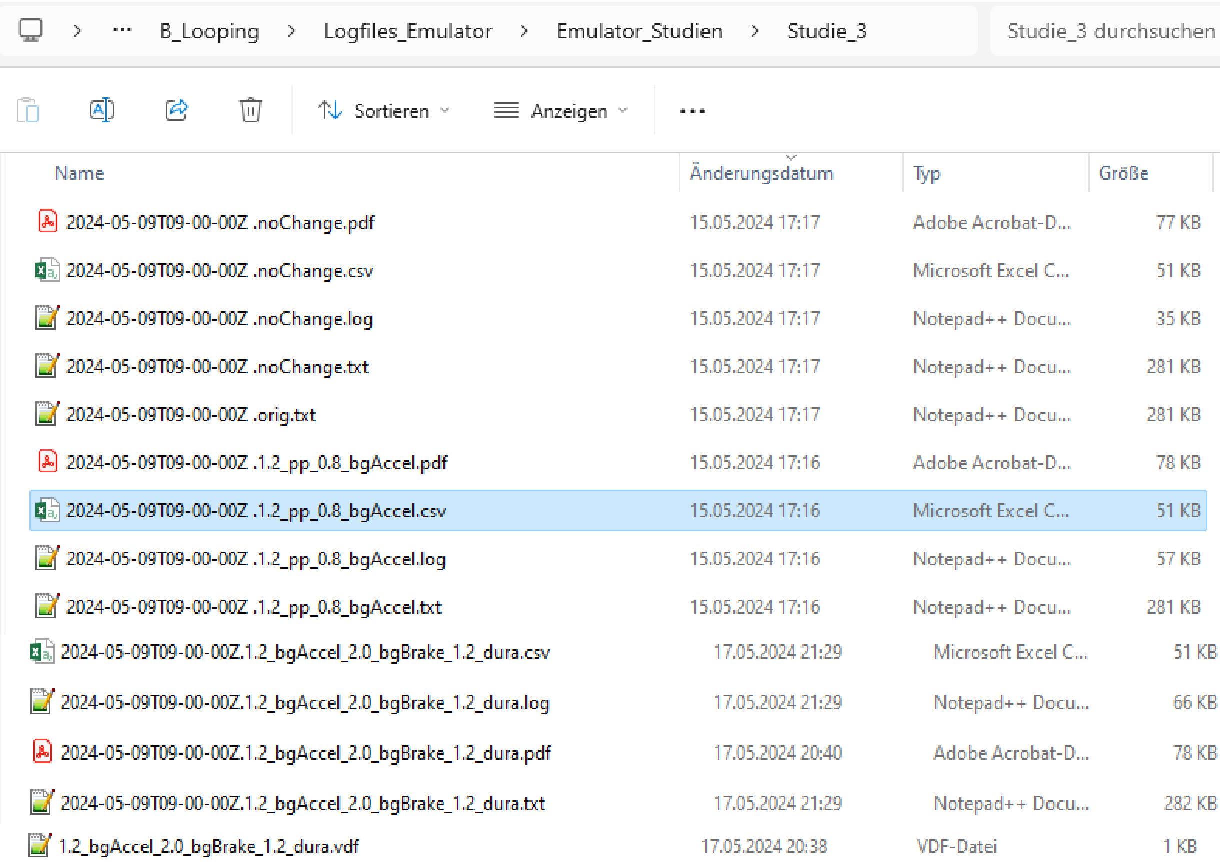
*.txt - Your short log of emulated analysis
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.txt

*.pdf etc. - Your graphic file comparing key values of original vs emulation
C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/Emulator_Studien/Studie_3/2024-05-09T09-00-00Z.1.2_bgAccel_2.0_bgBrake_1.2_dura.pdf

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

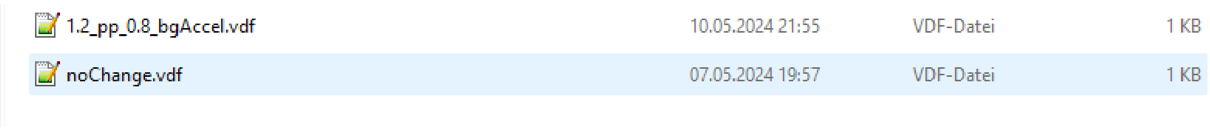
Besides the 1.2_bgAccel_2.0_bgBrake_1.2_dura.vdf case which I like to look into for the present high carb meal, I also prepared another vdf that investigates a factor 1.2 stronger pp_ISF and a weaker, factor 0.8, bgAccel_ISF (with the intention to test this, and a noChange (that ideally would already contain the conclusion on adapting the bgAccel_ISF_weight*), on a low carb meal later.

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



… › B_Looping › Logfiles_Emulator › Emulator_Studien › Studie_3
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



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 **(yourChange).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 (yourChange) 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

10.3.3.2 Tabular results

A) .csv results table and spreadsheet copies of it

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

> Note that the “**acce_ISF**” results are only in case of positive acceleration (that is our main focus) driven by the bgAccel_ISF_weight setting. (These are all positions > 1.0 in the “acce ISF” columns).
>
> **In case of negative acceleration** (decelerating rise, positions **< 1.0** in the **“acce ISF**” columns), **bgBrake_ISF_weight is applied**. As discussed in section 4.4, bgBrake_ISF might be most important (and interesting to analyze) in slowly resorbing meals.
>
> Note: maxBolus=0 means in this table that SMBs were not capped by maxBolus.

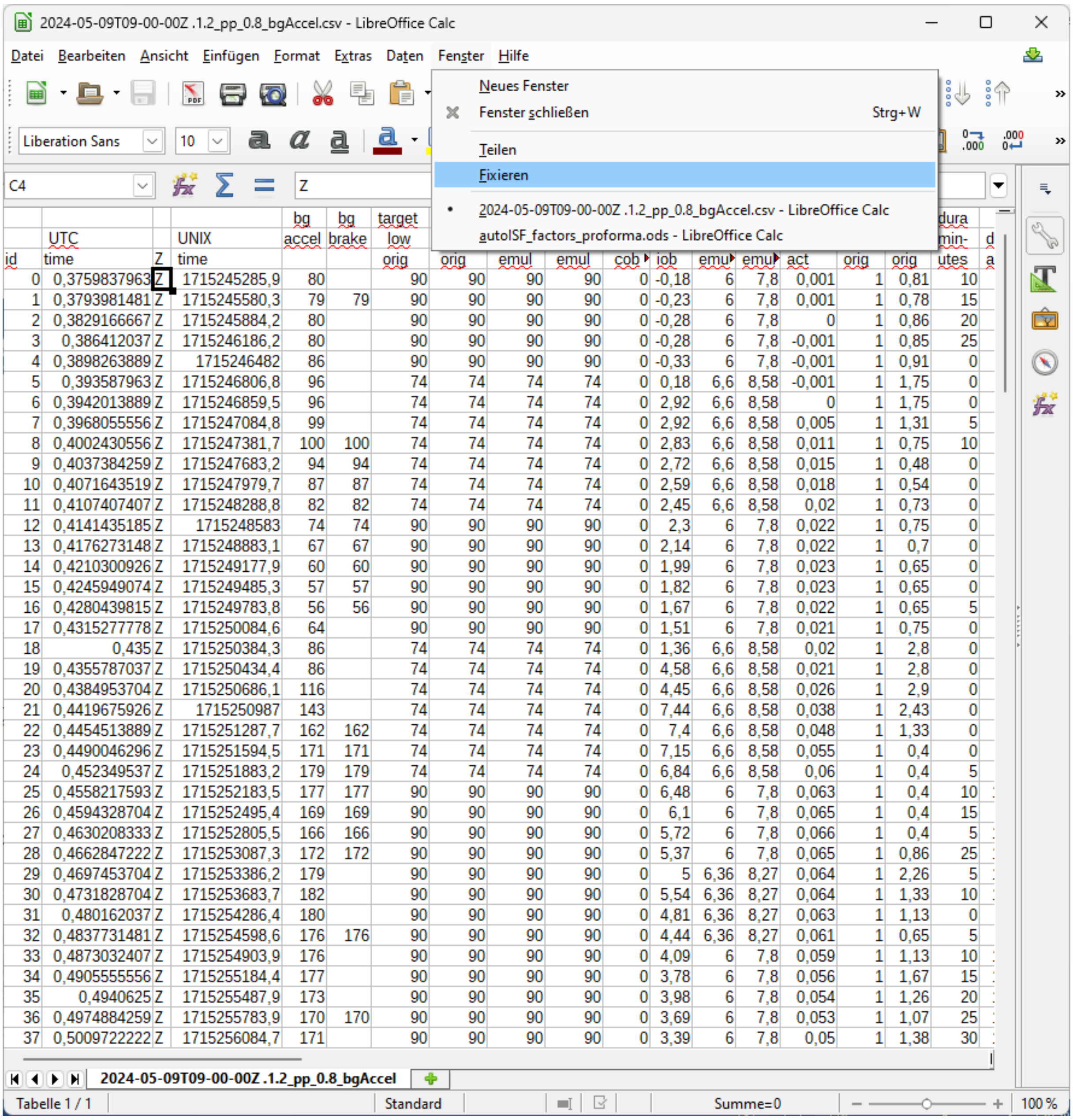
The **(your change).csv** shows in detail how **every single** loop **decision** would be influenced by the different settings you are investigating.

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

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

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

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


2024-05-09T09-00-00Z .1.2_pp_0.8_bgAccel.csv - LibreOffice Calc
Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster Hilfe
Neues Fenster
Fenster schließen Strg+W
Teilen
Fixieren
2024-05-09T09-00-00Z .1.2_pp_0.8_bgAccel.csv - LibreOffice Calc
autoISF_factors_proforma.ods - LibreOffice Calc
Liberation Sans
10
C4
Z
2024-05-09T09-00-00Z .1.2_pp_0.8_bgAccel
Tabelle 1 / 1
Standard
Summe=0
100 %

				bg	bg	target											dura	
	UTC		UNIX	accel	brake	low											min-	d
id	time	Z	time			orig	orig	emul	emul	cob	iob	emu	emu	act	orig	orig	utes	a
0	0,3759837963	Z	1715245285,9	80		90	90	90	90	0	-0,18	6	7,8	0,001	1	0,81	10	
1	0,3793981481	Z	1715245580,3	79	79	90	90	90	90	0	-0,23	6	7,8	0,001	1	0,78	15	
2	0,3829166667	Z	1715245884,2	80		90	90	90	90	0	-0,28	6	7,8	0	1	0,86	20	
3	0,386412037	Z	1715246186,2	80		90	90	90	90	0	-0,28	6	7,8	-0,001	1	0,85	25	
4	0,3898263889	Z	1715246482	86		90	90	90	90	0	-0,33	6	7,8	-0,001	1	0,91	0	
5	0,393587963	Z	1715246806,8	96		74	74	74	74	0	0,18	6,6	8,58	-0,001	1	1,75	0	
6	0,3942013889	Z	1715246859,5	96		74	74	74	74	0	2,92	6,6	8,58	0	1	1,75	0	
7	0,3968055556	Z	1715247084,8	99		74	74	74	74	0	2,92	6,6	8,58	0,005	1	1,31	5	
8	0,4002430556	Z	1715247381,7	100	100	74	74	74	74	0	2,83	6,6	8,58	0,011	1	0,75	10	
9	0,4037384259	Z	1715247683,2	94	94	74	74	74	74	0	2,72	6,6	8,58	0,015	1	0,48	0	
10	0,4071643519	Z	1715247979,7	87	87	74	74	74	74	0	2,59	6,6	8,58	0,018	1	0,54	0	
11	0,4107407407	Z	1715248288,8	82	82	74	74	74	74	0	2,45	6,6	8,58	0,02	1	0,73	0	
12	0,4141435185	Z	1715248583	74	74	90	90	90	90	0	2,3	6	7,8	0,022	1	0,75	0	
13	0,4176273148	Z	1715248883,1	67	67	90	90	90	90	0	2,14	6	7,8	0,022	1	0,7	0	
14	0,4210300926	Z	1715249177,9	60	60	90	90	90	90	0	1,99	6	7,8	0,023	1	0,65	0	
15	0,4245949074	Z	1715249485,3	57	57	90	90	90	90	0	1,82	6	7,8	0,023	1	0,65	0	
16	0,4280439815	Z	1715249783,8	56	56	90	90	90	90	0	1,67	6	7,8	0,022	1	0,65	5	
17	0,4315277778	Z	1715250084,6	64		90	90	90	90	0	1,51	6	7,8	0,021	1	0,75	0	
18	0,435	Z	1715250384,3	86		74	74	74	74	0	1,36	6,6	8,58	0,02	1	2,8	0	
19	0,4355787037	Z	1715250434,4	86		74	74	74	74	0	4,58	6,6	8,58	0,021	1	2,8	0	
20	0,4384953704	Z	1715250686,1	116		74	74	74	74	0	4,45	6,6	8,58	0,026	1	2,9	0	
21	0,4419675926	Z	1715250987	143		74	74	74	74	0	7,44	6,6	8,58	0,038	1	2,43	0	
22	0,4454513889	Z	1715251287,7	162	162	74	74	74	74	0	7,4	6,6	8,58	0,048	1	1,33	0	
23	0,4490046296	Z	1715251594,5	171	171	74	74	74	74	0	7,15	6,6	8,58	0,055	1	0,4	0	
24	0,452349537	Z	1715251883,2	179	179	74	74	74	74	0	6,84	6,6	8,58	0,06	1	0,4	5	
25	0,4558217593	Z	1715252183,5	177	177	90	90	90	90	0	6,48	6	7,8	0,063	1	0,4	10	
26	0,4594328704	Z	1715252495,4	169	169	90	90	90	90	0	6,1	6	7,8	0,065	1	0,4	15	
27	0,4630208333	Z	1715252805,5	166	166	90	90	90	90	0	5,72	6	7,8	0,066	1	0,4	5	
28	0,4662847222	Z	1715253087,3	172	172	90	90	90	90	0	5,37	6	7,8	0,065	1	0,86	25	
29	0,4697453704	Z	1715253386,2	179		90	90	90	90	0	5	6,36	8,27	0,064	1	2,26	5	
30	0,4731828704	Z	1715253683,7	182		90	90	90	90	0	5,54	6,36	8,27	0,064	1	1,33	10	
31	0,480162037	Z	1715254286,4	180		90	90	90	90	0	4,81	6,36	8,27	0,063	1	1,13	0	
32	0,4837731481	Z	1715254598,6	176	176	90	90	90	90	0	4,44	6,36	8,27	0,061	1	0,65	5	
33	0,4873032407	Z	1715254903,9	176		90	90	90	90	0	4,09	6	7,8	0,059	1	1,13	10	
34	0,4905555556	Z	1715255184,4	177		90	90	90	90	0	3,78	6	7,8	0,056	1	1,67	15	
35	0,4940625	Z	1715255487,9	173		90	90	90	90	0	3,98	6	7,8	0,054	1	1,26	20	
36	0,4974884259	Z	1715255783,9	170	170	90	90	90	90	0	3,69	6	7,8	0,053	1	1,07	25	
37	0,5009722222	Z	1715256084,7	171		90	90	90	90	0	3,39	6	7,8	0,05	1	1,38	30	

Still, the csv tables are overwhelming. You could proceed **in either of two** directions now:

A) Convert both (or all 3) csv files into one table in Excel or into Libre office calculator. Hide columns (and eventually also lines) that are of no particular interest for your analysis. Mark differences between noChange and (yourChanges) column data with color, add extra columns with additional calculations …

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

B) For the core data relevant to assessing your autoISF settings, there is an extra tool for convenient analysis - see the following section 10.3.3.3

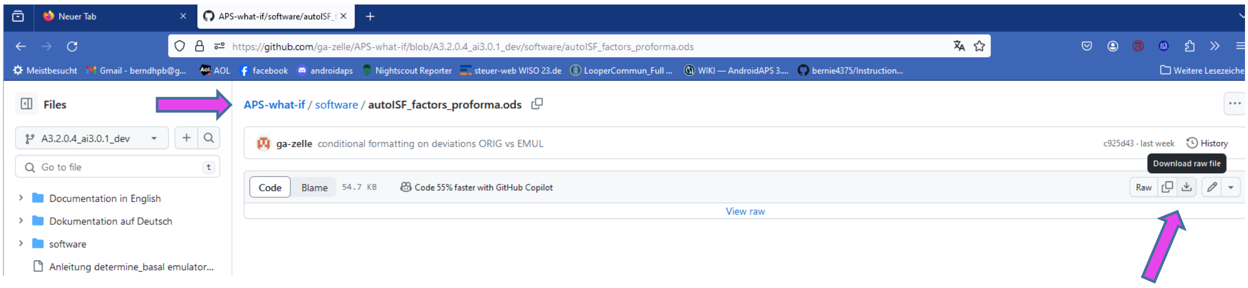
That tool is quite a bit of work to set up. Decide for yourself whether you do it, or whether you rather work with extracting the csv table into Excel (A), and work freely from there.

10.3.3.3 Automated extraction from tabular results *(optional add-on)*

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:

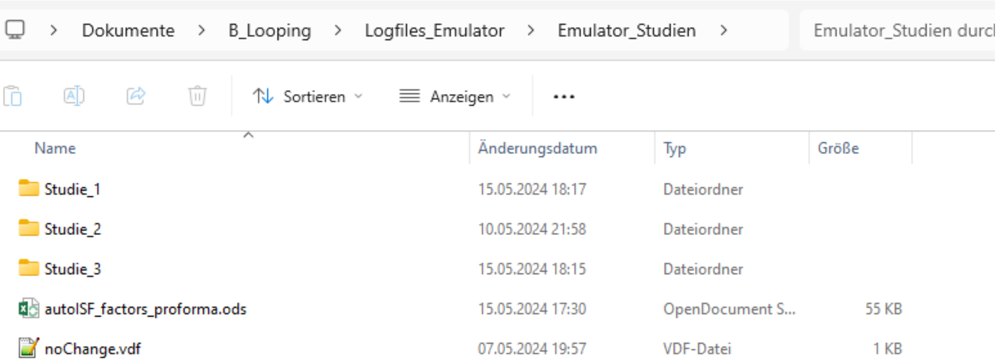


https://github.com/ga-zelle/APS-what-if/blob/A3.2.0.4_ai3.0.1_dev/software/autoISF_factors_proforma.ods

APS-what-if / software / autoISF_factors_proforma.ods

ga-zelle conditional formatting on deviations ORIG vs EMUL

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

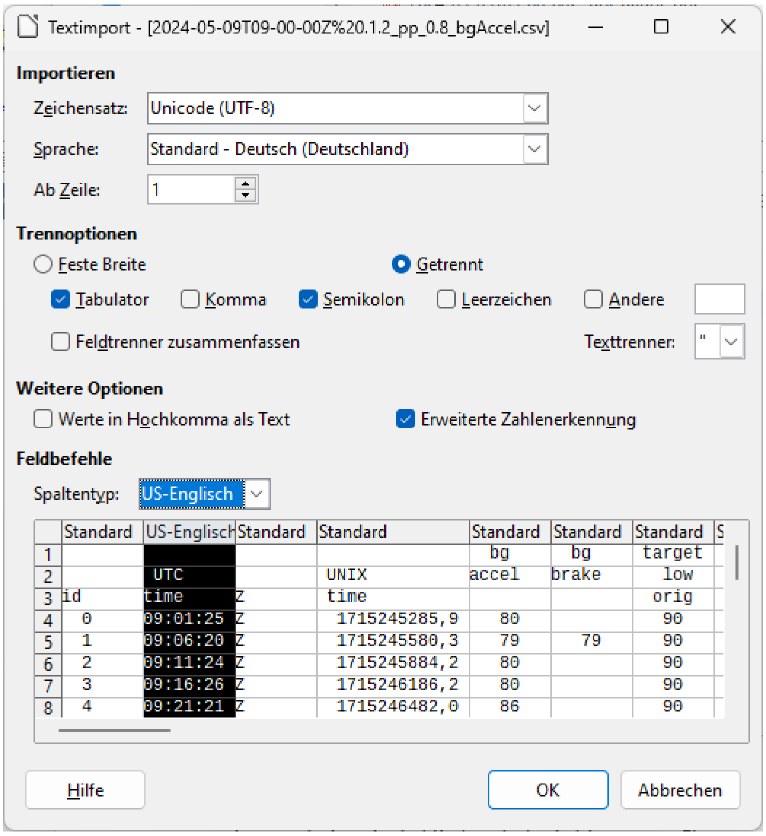


Dokumente > B_Looping > Logfiles_Emulator > Emulator_Studien >

Name	Änderungsdatum	Typ	Größe
Studie_1	15.05.2024 18:17	Dateiordner	
Studie_2	10.05.2024 21:58	Dateiordner	
Studie_3	15.05.2024 18:15	Dateiordner	
autoISF_factors_proforma.ods	15.05.2024 17:30	OpenDocument S...	55 KB
noChange.vdf	07.05.2024 19:57	VDF-Datei	1 KB

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

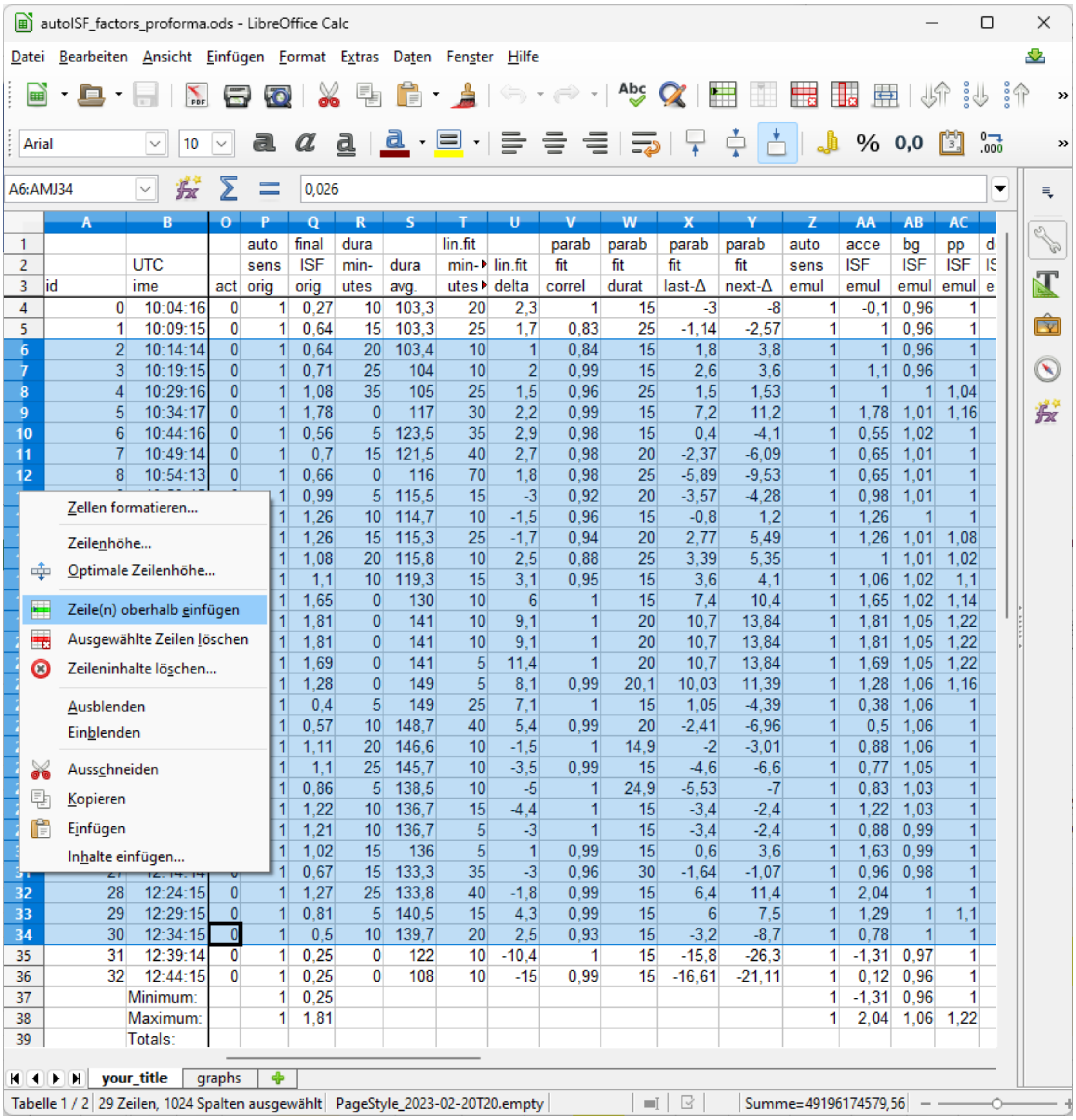
	Standard	US-Englisch	Standard	Standard	Standard	Standard	Standard	S
1					bg	bg	target	
2		UTC		UNIX	accel	brake	low	
3	id	time	Z	time			orig	
4	0	09:01:25	Z	1715245285,9	80		90	
5	1	09:06:20	Z	1715245580,3	79	79	90	
6	2	09:11:24	Z	1715245884,2	80		90	
7	3	09:16:26	Z	1715246186,2	80		90	
8	4	09:21:21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

3).Now start, in Libre office calculator, the autoISF_factors_proforma.ods …

This turns the first 30-some lines of your csv table (left side) into a form in which important effects are highlighted in color, and formatting is improved:

Now, you want this for the entire table.

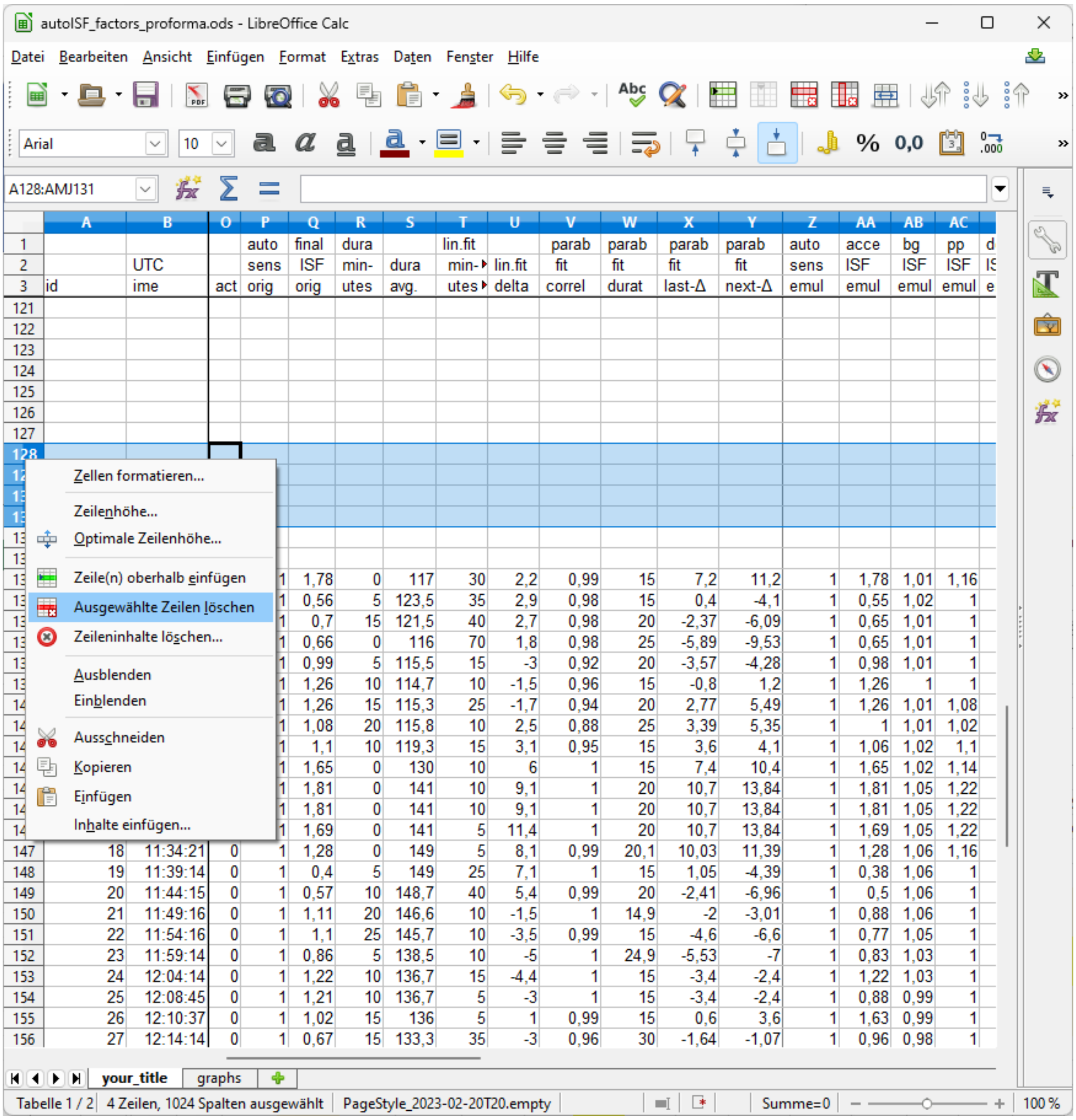
4).In the autoISF_factors_proforma table, highlight 20 or more lines (not including the first or last), and mouse right hand/insert above …



	A	B	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	
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	IS
3	id	ime	act	orig	orig	utes	avg.	utes	delta	correl	durat	last-Δ	next-Δ	emul	emul	emul	emul	e
4	0	10:04:16	0	1	0,27	10	103,3	20	2,3	1	15	-3	-8	1	-0,1	0,96	1	
5	1	10:09:15	0	1	0,64	15	103,3	25	1,7	0,83	25	-1,14	-2,57	1	1	0,96	1	
6	2	10:14:14	0	1	0,64	20	103,4	10	1	0,84	15	1,8	3,8	1	1	0,96	1	
7	3	10:19:15	0	1	0,71	25	104	10	2	0,99	15	2,6	3,6	1	1,1	0,96	1	
8	4	10:29:16	0	1	1,08	35	105	25	1,5	0,96	25	1,5	1,53	1	1	1	1,04	
9	5	10:34:17	0	1	1,78	0	117	30	2,2	0,99	15	7,2	11,2	1	1,78	1,01	1,16	
10	6	10:44:16	0	1	0,56	5	123,5	35	2,9	0,98	15	0,4	-4,1	1	0,55	1,02	1	
11	7	10:49:14	0	1	0,7	15	121,5	40	2,7	0,98	20	-2,37	-6,09	1	0,65	1,01	1	
12	8	10:54:13	0	1	0,66	0	116	70	1,8	0,98	25	-5,89	-9,53	1	0,65	1,01	1	
				1	0,99	5	115,5	15	-3	0,92	20	-3,57	-4,28	1	0,98	1,01	1	
				1	1,26	10	114,7	10	-1,5	0,96	15	-0,8	1,2	1	1,26	1	1	
				1	1,26	15	115,3	25	-1,7	0,94	20	2,77	5,49	1	1,26	1,01	1,08	
				1	1,08	20	115,8	10	2,5	0,88	25	3,39	5,35	1	1	1,01	1,02	
				1	1,1	10	119,3	15	3,1	0,95	15	3,6	4,1	1	1,06	1,02	1,1	
				1	1,65	0	130	10	6	1	15	7,4	10,4	1	1,65	1,02	1,14	
				1	1,81	0	141	10	9,1	1	20	10,7	13,84	1	1,81	1,05	1,22	
				1	1,81	0	141	10	9,1	1	20	10,7	13,84	1	1,81	1,05	1,22	
				1	1,69	0	141	5	11,4	1	20	10,7	13,84	1	1,69	1,05	1,22	
				1	1,28	0	149	5	8,1	0,99	20,1	10,03	11,39	1	1,28	1,06	1,16	
				1	0,4	5	149	25	7,1	1	15	1,05	-4,39	1	0,38	1,06	1	
				1	0,57	10	148,7	40	5,4	0,99	20	-2,41	-6,96	1	0,5	1,06	1	
				1	1,11	20	146,6	10	-1,5	1	14,9	-2	-3,01	1	0,88	1,06	1	
				1	1,1	25	145,7	10	-3,5	0,99	15	-4,6	-6,6	1	0,77	1,05	1	
				1	0,86	5	138,5	10	-5	1	24,9	-5,53	-7	1	0,83	1,03	1	
				1	1,22	10	136,7	15	-4,4	1	15	-3,4	-2,4	1	1,22	1,03	1	
				1	1,21	10	136,7	5	-3	1	15	-3,4	-2,4	1	0,88	0,99	1	
				1	1,02	15	136	5	1	0,99	15	0,6	3,6	1	1,63	0,99	1	
	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	
32	28	12:24:15	0	1	1,27	25	133,8	40	-1,8	0,99	15	6,4	11,4	1	2,04	1	1	
33	29	12:29:15	0	1	0,81	5	140,5	15	4,3	0,99	15	6	7,5	1	1,29	1	1,1	
34	30	12:34:15	0	1	0,5	10	139,7	20	2,5	0,93	15	-3,2	-8,7	1	0,78	1	1	
35	31	12:39:14	0	1	0,25	0	122	10	-10,4	1	15	-15,8	-26,3	1	-1,31	0,97	1	
36	32	12:44:15	0	1	0,25	0	108	10	-15	0,99	15	-16,61	-21,11	1	0,12	0,96	1	
37		Minimum:		1	0,25									1	-1,31	0,96	1	
38		Maximum:		1	1,81									1	2,04	1,06	1,22	
39		Totals:																

Do this as often as you need to create the number of lines that your emulated csv file comes with.

If you ended up with too many lines, erase the superfluous number (any four, in the example):


autoISF_factors_proforma.ods - LibreOffice Calc
Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster Hilfe
Arial
10
A128:AMJ131
Zellen formatieren...
Zeilenhöhe...
Optimale Zeilenhöhe...
Zeile(n) oberhalb einfügen
Ausgewählte Zeilen löschen
Zeileninhalte löschen...
Ausblenden
Einblenden
Ausschneiden
Kopieren
Einfügen
Inhalte einfügen...
your_title
graphs
Tabelle 1 / 2
4 Zeilen, 1024 Spalten ausgewählt
PageStyle_2023-02-20T20.empty
Summe=0
100 %

	A	B	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	
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	IS
3	id	ime	act	orig	orig	utes	avg.	utes	delta	correl	durat	last-Δ	next-Δ	emul	emul	emul	emul	e
121																		
122																		
123																		
124																		
125																		
126																		
127																		
128																		
12																		
13																		
13																		
13																		
13																		
13				1	1,78	0	117	30	2,2	0,99	15	7,2	11,2	1	1,78	1,01	1,16	
13				1	0,56	5	123,5	35	2,9	0,98	15	0,4	-4,1	1	0,55	1,02	1	
13				1	0,7	15	121,5	40	2,7	0,98	20	-2,37	-6,09	1	0,65	1,01	1	
13				1	0,66	0	116	70	1,8	0,98	25	-5,89	-9,53	1	0,65	1,01	1	
13				1	0,99	5	115,5	15	-3	0,92	20	-3,57	-4,28	1	0,98	1,01	1	
13				1	1,26	10	114,7	10	-1,5	0,96	15	-0,8	1,2	1	1,26	1	1	
14				1	1,26	15	115,3	25	-1,7	0,94	20	2,77	5,49	1	1,26	1,01	1,08	
14				1	1,08	20	115,8	10	2,5	0,88	25	3,39	5,35	1	1	1,01	1,02	
14				1	1,1	10	119,3	15	3,1	0,95	15	3,6	4,1	1	1,06	1,02	1,1	
14				1	1,65	0	130	10	6	1	15	7,4	10,4	1	1,65	1,02	1,14	
14				1	1,81	0	141	10	9,1	1	20	10,7	13,84	1	1,81	1,05	1,22	
14				1	1,81	0	141	10	9,1	1	20	10,7	13,84	1	1,81	1,05	1,22	
14				1	1,69	0	141	5	11,4	1	20	10,7	13,84	1	1,69	1,05	1,22	
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	

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)

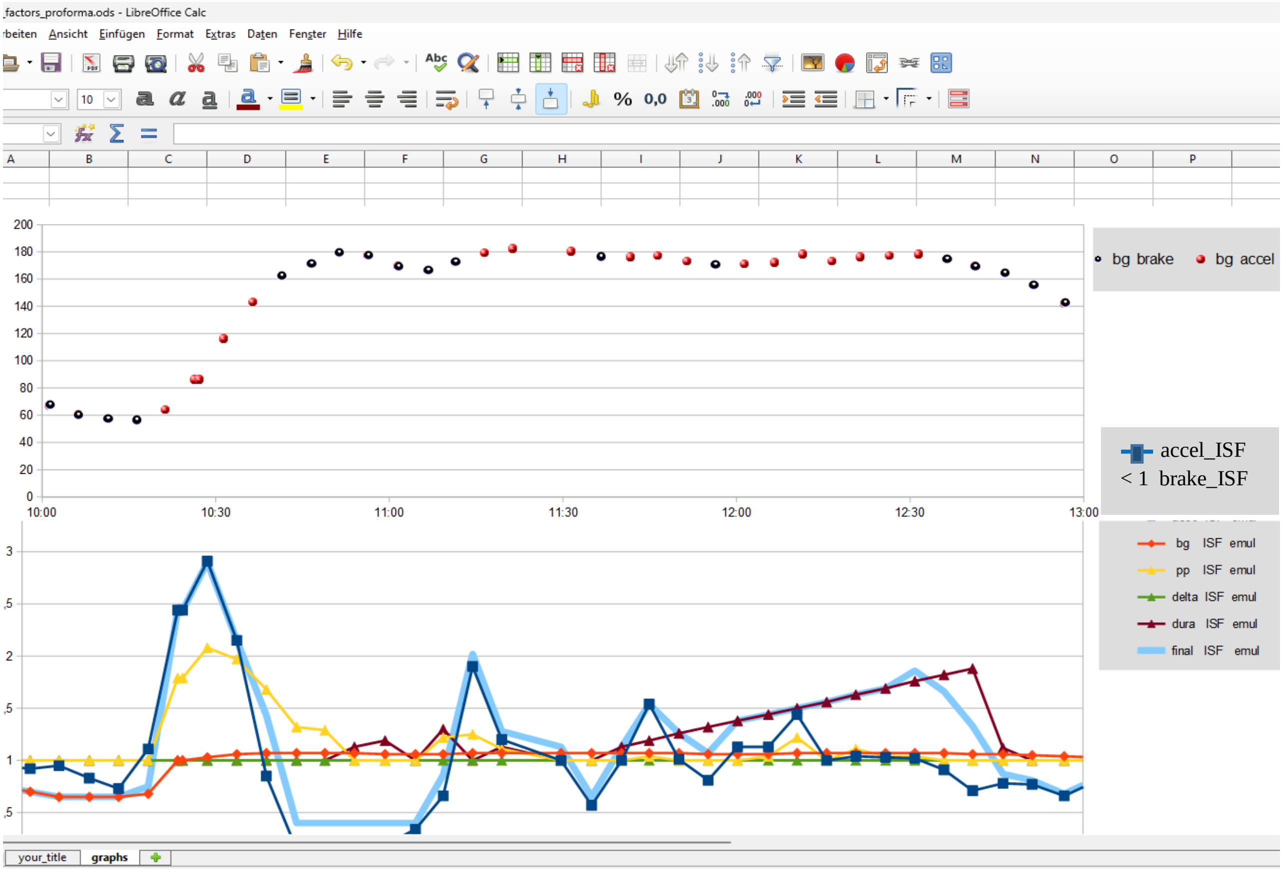
Now you have a table with optimized lay-out that incorporates key data from both your no change AND of your investigated changes.csv files.

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

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

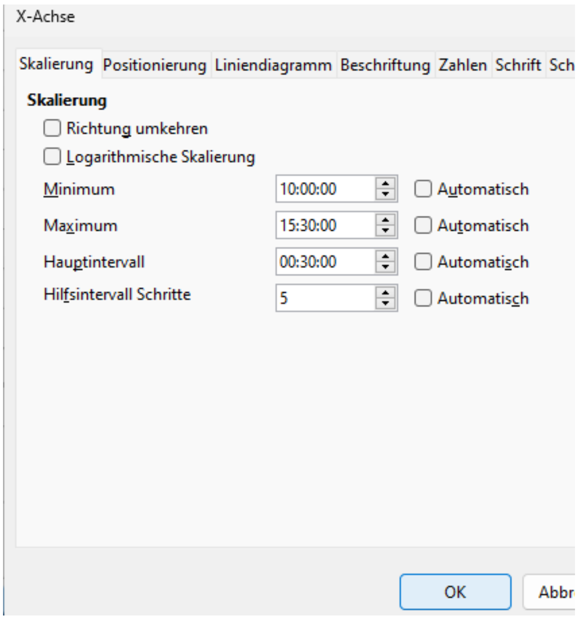
The top graph is the bg curve (the actually seen bg).

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



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

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



Skalierung

- □ Richtung umkehren
- □ Logarithmische Skalierung

Minimum	10:00:00	□ Automatisch
Maximum	15:30:00	□ Automatisch
Hauptintervall	00:30:00	□ Automatisch
Hilfsintervall Schritte	5	□ Automatisch

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 Emulaton 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).

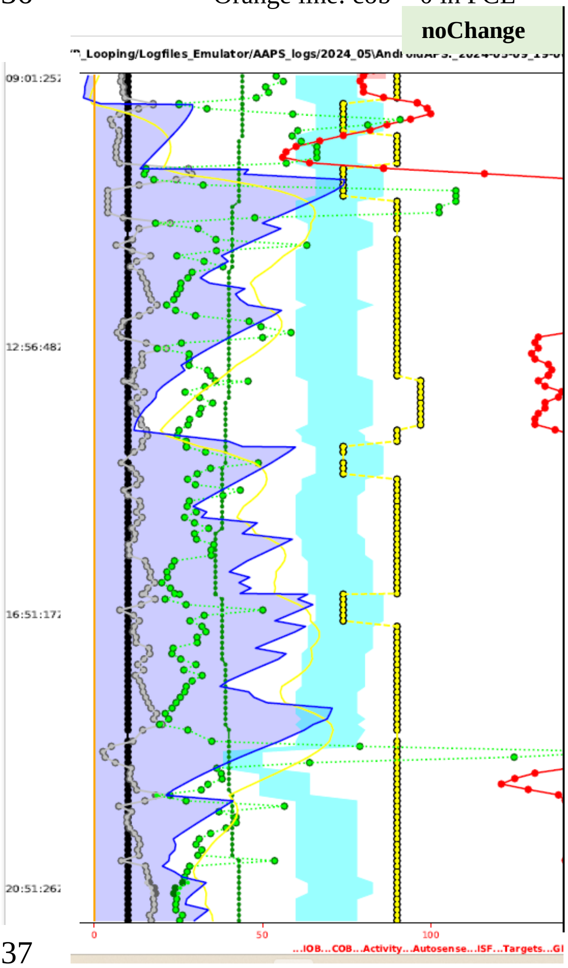
> Note that ideally we want FCL coverage of our entire “normal day” meal spectrum by **one** set of settings.
> So, **not much is gained if you put a lot of effort in optimizing FCL settings for one meal**.
>
> You will need iterations. Do such analysis for **two or three very different meals** that 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 aggressive or mild (category)_ISF_weights you could set.

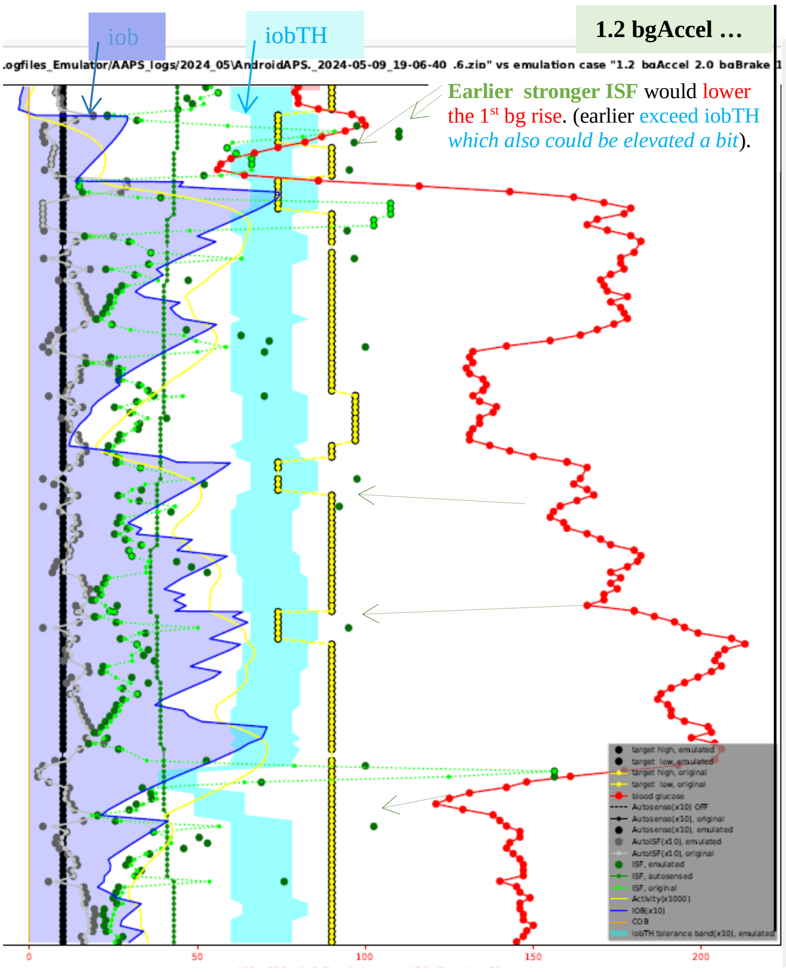
The initial iob received might be limited by allowed SMB sizes, autoISFmax, or the (dynamic!) iobTH. You will have to look into the data table to find out about this (a quick orientation - notably regarding the light blue iobTH band, see next page - is also possible in the pdf result files you have in your project file (project file example “Studie 3” in 2nd 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)
- Light blue corridor: Left edge is set iobTH, and bandwidth +30% (would be +20% at elevated TT)
- Dark blue line: iob (exceeding twice the iobTH, with temp. SMB shut-off
- Orange line: cob = 0 in FCL





Earlier stronger ISF would lower the 1st bg rise. (earlier exceed iobTH *which also could be elevated a bit*).

- As bg did not convincingly come down enough, one could hypothesize that iobTH should be elevated. ((But, again, this would have to be confirmed also with other kinds of meals)).
- Thin yellow line: Insulin activity
- Green dotted line: ISF as would result from AAPS w/Autosens

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

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

- Black line: Profile ISF
- Gray scatter points: ISF weakened (to the left of black line) or strengthened (to the right)

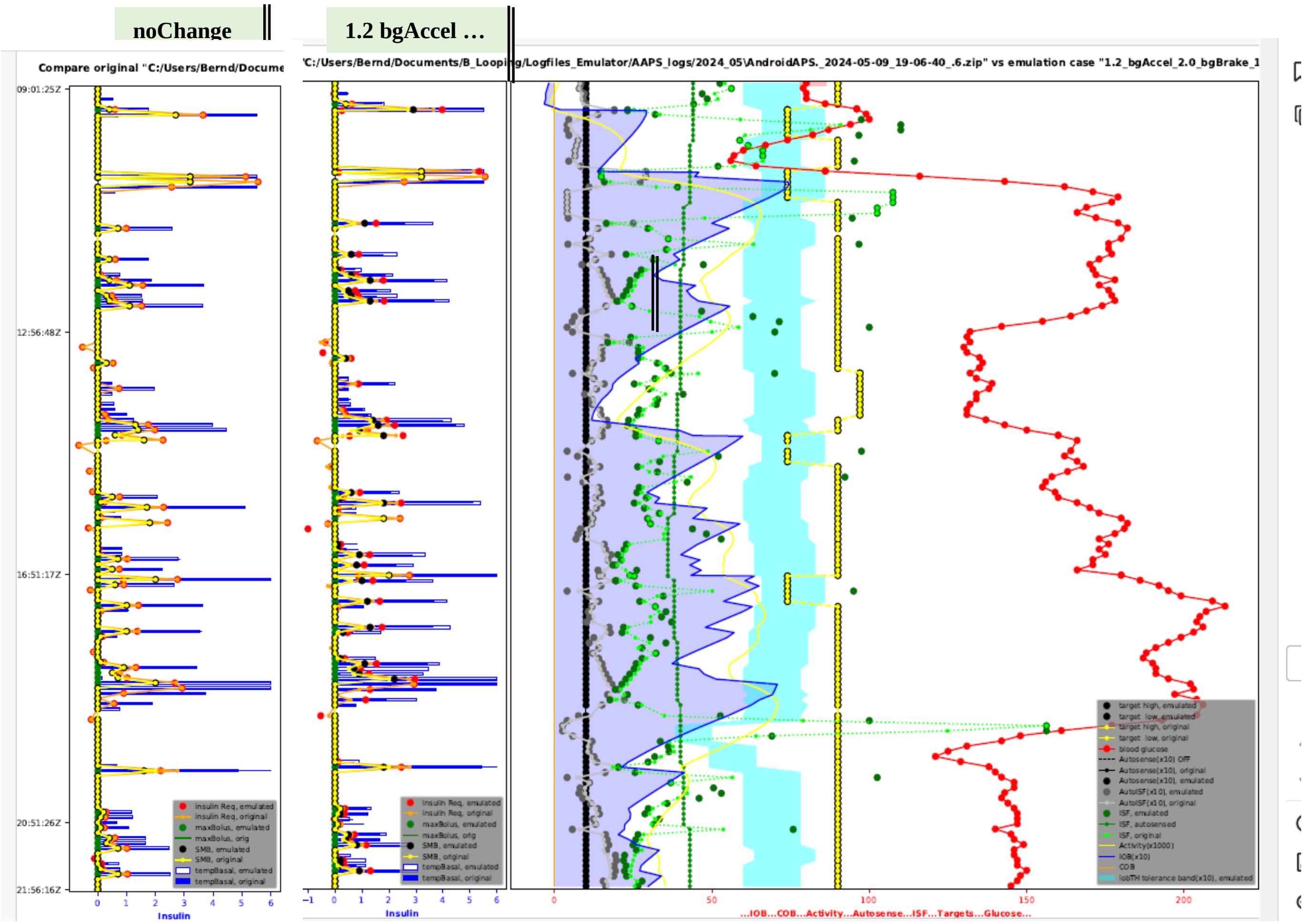
Regarding the other changed parameters: Stronger dura_ISF would suggest more insulin towards the end of plateaus; this should have helped in the 1st plateau (red curve, top right quadrant of the picture). However, same setting would have to work also on 2nd plateau; the chart cuts off there, so too early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

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

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

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

(unfinished / to be explained later)   (**…**note: yourChange = 1.2_bgAccel_2.0_bgBrake_1.2_dura)



noChange

1.2 bgAccel …

Compare original "C:/Users/Bernd/Docume

'C:/Users/Bernd/Documents/B_Looping/Logfiles_Emulator/AAPS_logs/2024_05\AndroidAPS._2024-05-09_19-06-40_.6.zip" vs emulation case "1.2_bgAccel_2.0_bgBrake_1

10.3.3.5 delta table from “what-if” run with (yourChange).vdf

In case you want to analyze delta, short and long average deltas, see section 10.2.4.5

To analyze deltas in a “what-if” scenario really does not make much sense, because effects from each single change ripples through many subsequent situations, and it is impossible to predict how glucose curve, and therefore also how deltas, would develop in the what-if case.