(https://colab.research.google.com/github/janpeter19/BPL_TEST2_Perfusion/blob/main/BPL_TEST2_Perfusior

BPL_TEST2_Perfusion script with PyFMI ver 2.9.8

The key library PyFMI v2.9.8 is installed.

After the installation a small application BPL_TEST2_Perfusion is loaded and run. You can continue with this example if you like.

In [ ]:

```
!lsb_release -a # Actual VM Ubuntu version used by Google
```

```
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.5 LTS
Release:        20.04
Codename:       focal
```

In [ ]:

```
%env PYTHONPATH=
```

```
env: PYTHONPATH=
```

In [ ]:

```
!wget https://repo.anaconda.com/miniconda/Miniconda3-py38_22.11.1-1-Linux-x86_64.
!chmod +x Miniconda3-py38_22.11.1-1-Linux-x86_64.sh
!bash ./Miniconda3-py38_22.11.1-1-Linux-x86_64.sh -b -f -p /usr/local
import sys
sys.path.append('/usr/local/lib/python3.8/site-packages/')
```

```
--2023-01-25 10:17:20--  https://repo.anaconda.com/miniconda/Minicon
da3-py38_22.11.1-1-Linux-x86_64.sh (https://repo.anaconda.com/minico
nda/Miniconda3-py38_22.11.1-1-Linux-x86_64.sh)
Resolving repo.anaconda.com (repo.anaconda.com)... 104.16.131.3, 10
4.16.130.3, 2606:4700::6810:8203, ...
Connecting to repo.anaconda.com (repo.anaconda.com)|104.16.131.3|:44
3... connected.
HTTP request sent, awaiting response... 200 OK
Length: 64630241 (62M) [application/x-sh]
Saving to: ‘Miniconda3-py38_22.11.1-1-Linux-x86_64.sh’

Miniconda3-py38_22. 100%[===================>]  61.64M   124MB/s
in 0.5s

2023-01-25 10:17:21 (124 MB/s) - ‘Miniconda3-py38_22.11.1-1-Linux-x8
6_64.sh’ saved [64630241/64630241]

PREFIX=/usr/local
Unpacking payload ...

Installing base environment...


Downloading and Extracting Packages


Downloading and Extracting Packages

Preparing transaction: done
Executing transaction: done
installation finished.
```

In [ ]:

```
!conda update -n base -c defaults conda --yes
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /usr/local

  added / updated specs:
    - conda


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ca-certificates-2023.01.10 |       h06a4308_0         120 KB
    conda-23.1.0               |   py38h06a4308_0         942 KB
    conda-package-handling-2.0.2|   py38h06a4308_0         267 KB
    conda-package-streaming-0.7.0|   py38h06a4308_0          26 KB
```

In [ ]:

```
!conda --version
!python --version
```

```
conda 23.1.0
Python 3.8.15
```

In [ ]:

```
!conda install -c conda-forge pyfmi --yes # Install the key package
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /usr/local

  added / updated specs:
    - pyfmi


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    _libgcc_mutex-0.1          |      conda_forge           3 KB
conda-forge
    _openmp_mutex-4.5          |       2_kmp_llvm           6 KB
conda-forge
```

Notes of BPL_TEST2_Perfusion

This notebook explore perfusion cultivation in comparison with ordinary continuous cultivation (chemostat) and use comparable settings to earlier notebook. Further you see here examples of interaction with the simplified commands par(), init(), simu() etc as well as direct interaction with the FMU which is called "model" here. The last simulation is always available in the workspace and called "sim_res". Note that describe() brings mainly up from descriptive information from the Modelica code from the FMU but is complemented by

Now specific installation run a simulation and notebook for that Start with connecting to Github. Then upload the two files:

- FMU - BPL_TEST2_Perfusion_linux_om_me.fmu
- Setup-file - BPL_TEST2_Perfusion_explore.py

In [1]:

```
%%bash
git clone https://github.com/janpeter19/BPL_TEST2_Perfusion
```

```
bash: line 1: git: command not found

---------------------------------------------------------------------------
CalledProcessError                        Traceback (most recent call last)
Cell In[1], line 1
----> 1 get_ipython().run_cell_magic('bash', '', 'git clone https://github.com/janpeter19/BPL_TEST2_Perfusion (https://github.com/janpeter19/BPL_TEST2_Perfusion)\n')

File ~/miniconda3/envs/pyfmi/lib/python3.8/site-packages/IPython/core/interactiveshell.py:2422, in InteractiveShell.run_cell_magic(self, magic_name, line, cell)
   2420 with self.builtin_trap:
   2421     args = (magic_arg_s, cell)
-> 2422     result = fn(*args, **kwargs)
   2423 return result

File ~/miniconda3/envs/pyfmi/lib/python3.8/site-packages/IPython/core/magics/script.py:153, in ScriptMagics._make_script_magic.<locals>.named_script_magic(line, cell)
    151 else:
    152     line = script
--> 153 return self.shebang(line, cell)

File ~/miniconda3/envs/pyfmi/lib/python3.8/site-packages/IPython/core/magics/script.py:305, in ScriptMagics.shebang(self, line, cell)
    300 if args.raise_error and p.returncode != 0:
    301     # If we get here and p.returncode is still None, we must have
    302     # killed it but not yet seen its return code. We don't wait for it,
    303     # in case it's stuck in uninterruptible sleep. -9 = SIGKILL
    304     rc = p.returncode or -9
--> 305     raise CalledProcessError(rc, cell)

CalledProcessError: Command 'b'git clone https://github.com/janpeter19/BPL_TEST2_Perfusion\n'' (https://github.com/janpeter19/BPL_TEST2_Perfusion\n'') returned non-zero exit status 127.
```

In [ ]:

```
%cd BPL_TEST2_Perfusion
```

In [30]:

```
run -i BPL_TEST2_Perfusion_explore.py
```

```
Linux - run FMU pre-comiled OpenModelica 1.21.0

Model for bioreactor has been setup. Key commands:
 - par()       - change of parameters and initial values
 - init()      - change initial values only
 - simu()      - simulate and plot
 - newplot()   - make a new plot
 - show()      - show plot from previous simulation
 - disp()      - display parameters and initial values from the last
simulation
 - describe()  - describe culture, broth, parameters, variables with
values/units

Note that both disp() and describe() takes values from the last simu
lation

Brief information about a command by help(), eg help(simu)
Key system information is listed with the command system_info()

<Figure size 984.252x787.402 with 0 Axes>
```

In [31]:

```
# Filter out DepracationWarnings for 'np.float as alias' is needed - wish
import warnings
warnings.filterwarnings("ignore")
```

In [32]:

```
%matplotlib inline
plt.rcParams['figure.figsize'] = [25/2.54, 20/2.54]
```

In [33]:

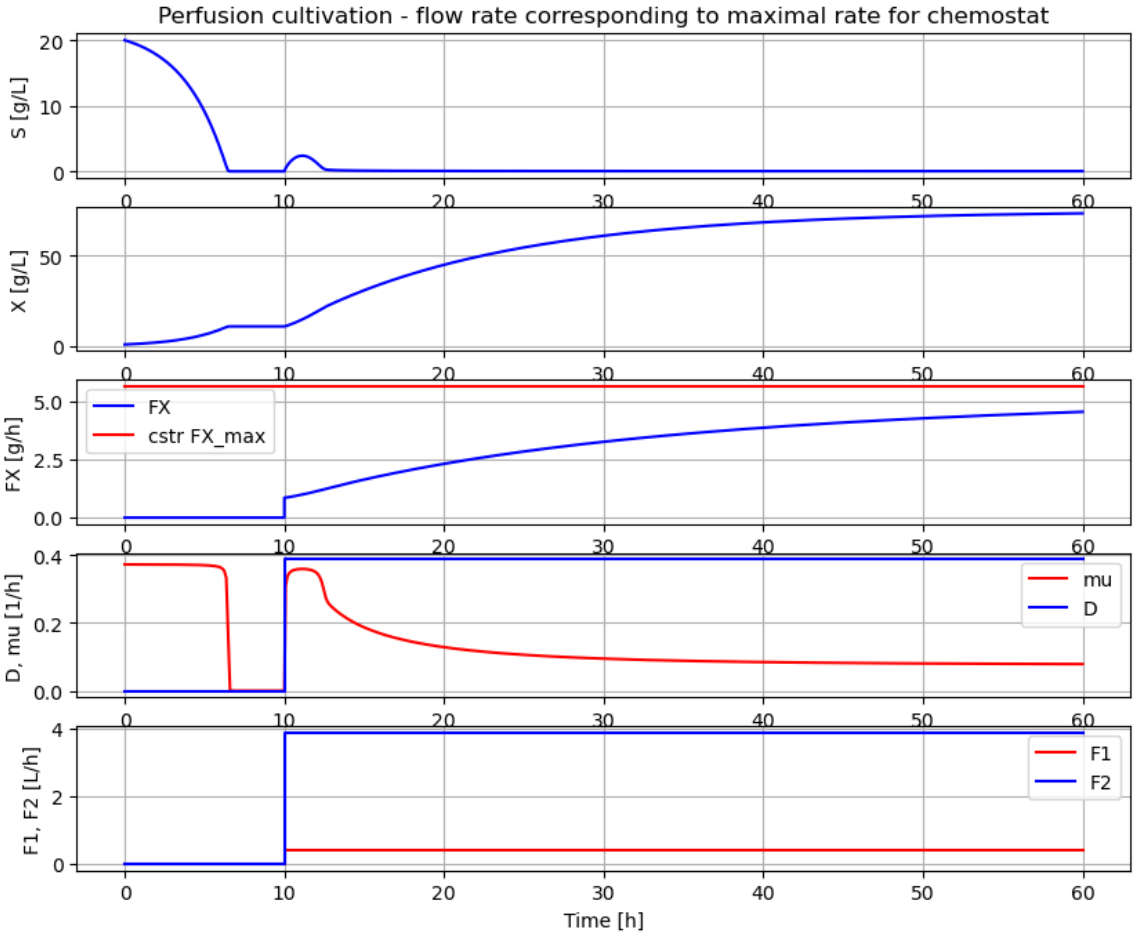
```
# Process parameters used throughout
par(Y=0.5, qSmax=0.75, Ks=0.1)                                    # Culture
par(filter_eps=0.10, filter_alpha_X=0.02, filter_alpha_S=0.10)    # Filter
par(S_in=30.0)                                                    # Inlet subst
init(V_0=1.0, VX_0=1.0)                                           # Process ini
eps = parDict['filter_eps']                                       # Pump schedu
```

In [34]:

```
# Simulation of process with flow rate clot to wash-out for chemostat

init(VS_0=20)                                                    # Process initial
par(pump1_t1=10, pump2_t1=10)                                    # Pump schedule - recycl
par(pump1_F1=2.5*0.155, pump2_F1=2.5*0.155/eps)
par(pump1_t2=940, pump2_t2=940, pump1_t3=950, pump2_t3=950, pump1_t4=960, pump2_t
newplot(title='Perfusion cultivation - flow rate corresponding to maximal rate fo
simu(60)
```

```
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
```



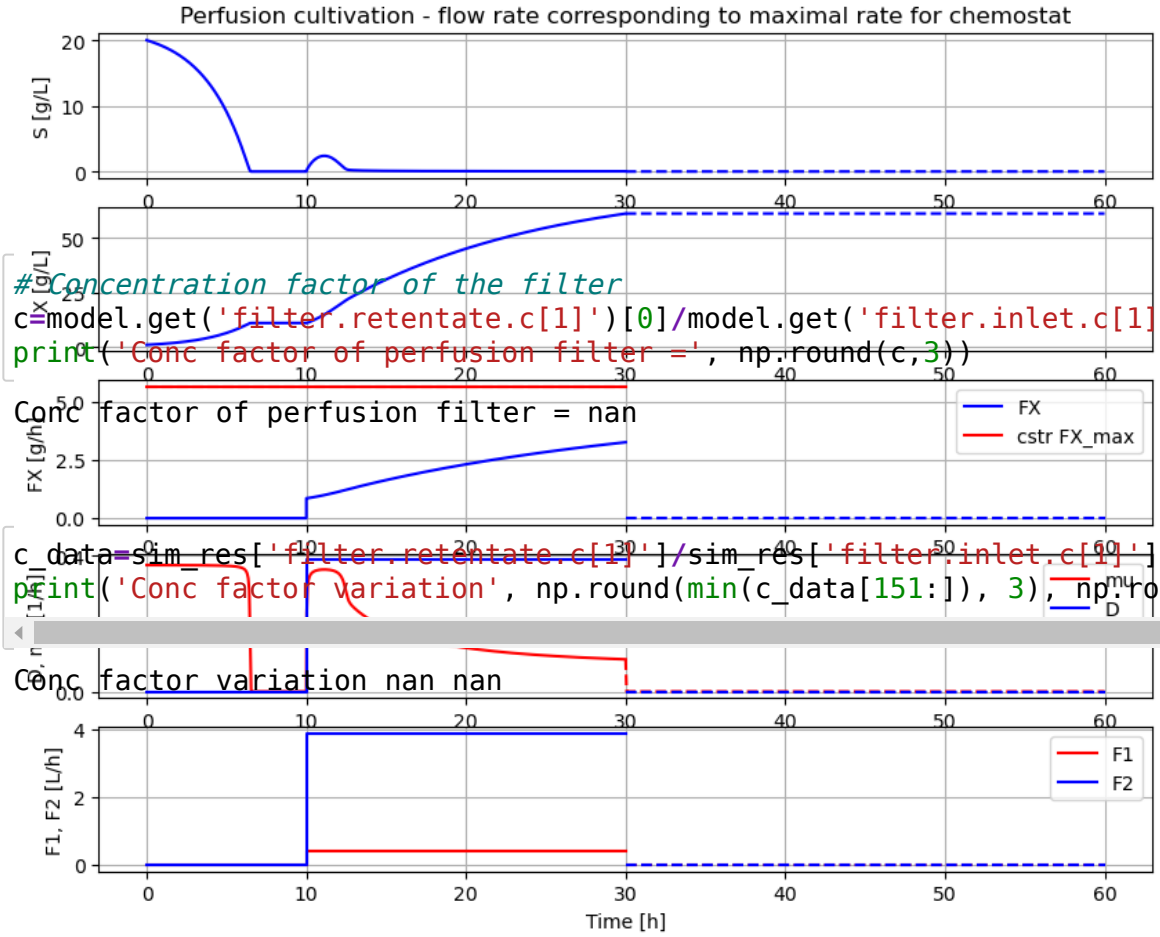
In [35]:

```
# Simulation of process with flow rate clot to wash-out for chemostat

init(VS_0=20)                                                # Process initial
par(pump1_t1=10, pump2_t1=10)                                # Pump schedule - recycl
par(pump1_F1=2.5*0.155, pump2_F1=2.5*0.155/eps)
par(pump1_t2=940, pump2_t2=940, pump1_t3=950, pump2_t3=950, pump1_t4=960, pump2_t

newplot(title='Perfusion cultivation - flow rate corresponding to maximal rate fo
simu(30)
simu(30,'cont')
```

```
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
```




```
#Concentration factor of the filter
c=model.get('filter.retentate.c[1]')[0]/model.get('filter.inlet.c[1]')[0]
print('Conc factor of perfusion filter =', np.round(c,3))
```

```
Conc factor of perfusion filter = nan
```

```
c_data=sim_res['filter.retentate.c[1]']/sim_res['filter.inlet.c[1]']
print('Conc factor variation', np.round(min(c_data[151:]), 3), np.round(max(c_dat
```

```
Conc factor variation nan nan
```

In [38]:

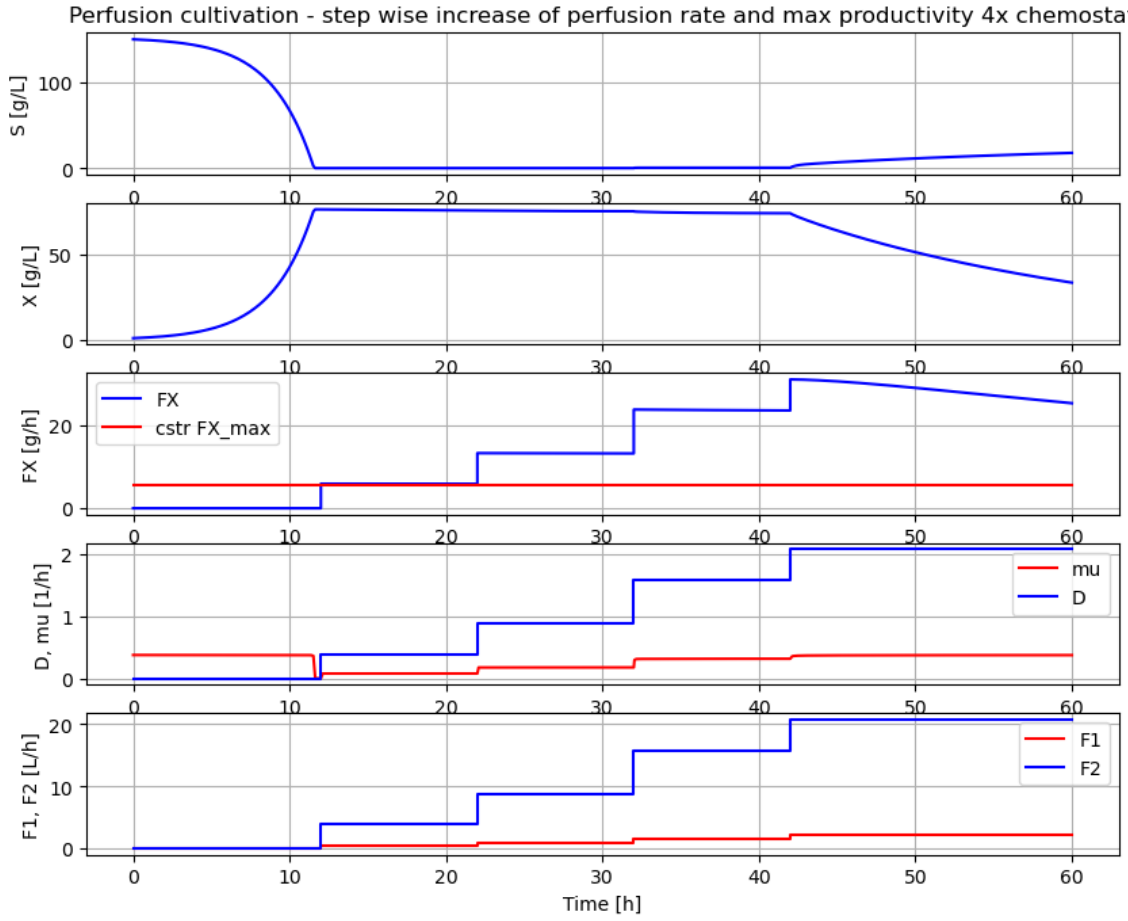
```
# Simulation of process with step-wise increase of pefusion rate until wash-out.
# This means that re-circulation rate change at the same time as the perfusion ra

init(VS_0=150)                                          # Process initial varied

par(pump1_t1=12, pump2_t1=12)                           # Pump schedule - recycl
par(pump1_F1=2.5*0.155, pump2_F1=2.5*0.155/eps)
par(pump1_t2=22, pump2_t2=22)
par(pump1_F2=2.5*0.35, pump2_F2=2.5*0.35/eps)
par(pump1_t3=32, pump2_t3=32)
par(pump1_F3=2.5*0.63, pump2_F3=2.5*0.63/eps)
par(pump1_t4=42, pump2_t4=42)
par(pump1_F4=2.5*0.83, pump2_F4=2.5*0.83/eps)

newplot(title='Perfusion cultivation - step wise increase of perfusion rate and m
simu(60)
```

```
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
```



In [39]:

```
# Simulation without a plot and just to check typical values at high production r
simu(40)
c_data=sim_res['filter.retentate.c[1]']/sim_res['filter.inlet.c[1]']
print('Conc factor variation', np.round(min(c_data[190:]), 3), 'to', np.round(max
```

```
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
stdout            | warning | The default linear solver fails, the f
allback solver with total pivoting is started at time 0.000000. That
might raise performance issues, for more information use -lv LOG_LS.
Conc factor variation 1.089 to 1.089
```

In [40]:

```
#describe('cstrProdMax')
```

In [41]:

```
# The maximal biomass productivity before washout is obtained aroudn 40 hours
np.round(model.get('harvesttank.inlet.F')[0]*model.get('harvesttank.inlet.c[1]')[
```

Out[41]:

```
23.5
```

In [42]:

```
# Thus perfusion (with this filter) brings a productivity improvement of about
np.round(23.5/5.6,1)
```

Out[42]:

```
4.2
```

In [43]:

```
# Finally we check the filter flow rates at time 40 hour - note the negative sign
model.get('filter.inlet.F')[0]
```

Out[43]:

```
15.749999999999998
```

In [44]:

```
model.get('filter.filtrate.F')[0]
```

Out[44]:

```
-1.575
```

In [45]:

```
model.get('filter.retentate.F')[0]
```

Out[45]:

```
-14.174999999999999
```

Summary

- The perfusion filter had a concentration factor of cells around 1.08 and re-cycling flow was set to a factor 10 higher than the perfusion rate and changed when perfusion rate was change to keep the ratio factor 10.
- The first simulation showed that by cell retention using perfusion filter the process could be run at a perfusion flow rate at the maximal flow rate possible for corresponding chemostat culture and cell concetration increased steadily.
- The second simulation showed that with a proper startup cell concentration, the cell concentration remained constant when perfusion rate increased in a similar way as what we see in a chemostat.
- The second simulation also showed that biomass productivity in this case was increased by a factor 4.2 compared to chemostat.
- If the perfusion rate increased to higher levels washout started but the decrase of cell concentration was slow.

Some of you who read this may have your perfusion experience with CHO-cultures. For such cultures the cell concentration do increase with increase of perfusion rate and there are understood reasons for that. But for this simplifed process as well as microbial processes they typically keep cell concentration constant when flow rate is chaged, and that under quite wide conditions. I will try come back to this phenomena in a later notebook.

In [46]:

```
# List of components in the process setup and also a couple of other things like
describe('parts')
```

```
['bioreactor', 'bioreactor.culture', 'D', 'feedtank', 'filter', 'har
vesttank', 'schemePump1', 'schemePump2']
```

In [47]:

```
describe('MSL')
```

```
MSL: 3.2.3 - used components: RealInput, RealOutput, CombiTimeTable,
Types
```

In [48]:

```
system_info()
```

```
System information
 -OS: Linux
 -Python: 3.8.16
 -Scipy: not installed in the notebook
 -PyFMI: 2.9.8
 -FMU by: OpenModelica Compiler OpenModelica 1.21.0~dev-185-g9d983b8
 -FMI: 2.0
 -Type: FMUModelME2
 -Name: BPL_TEST2.Perfusion
 -Generated: 2023-01-25T11:02:31Z
 -MSL: 3.2.3
 -Description: Bioprocess Library version 2.1.1-beta
 -Interaction: FMU-explore version 0.9.6
```

In [ ]:

```

```