BPL_TEST2_Perfusion script with FMPy

The key library FMPy is installed.

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

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

```
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 22.04.3 LTS
Release:        22.04
Codename:       jammy
```

```
%env PYTHONPATH=
```

```
env: PYTHONPATH=
```

```
!wget https://repo.anaconda.com/miniconda/Miniconda3-py312_24.3.0-0-Linux-x86_64.sh
!chmod +x Miniconda3-py312_24.3.0-0-Linux-x86_64.sh
!bash ./Miniconda3-py312_24.3.0-0-Linux-x86_64.sh -b -f -p /usr/local
import sys
sys.path.append('/usr/local/lib/python3.12/site-packages/')
```

```
--2024-10-24 09:43:12--  https://repo.anaconda.com/miniconda/Miniconda3-py312_24.3.0-0-Linux-x86_64.sh
Resolving repo.anaconda.com (repo.anaconda.com)... 104.16.32.241, 104.16.191.158, 2606:4700::6810:bf9e, ...
Connecting to repo.anaconda.com (repo.anaconda.com)|104.16.32.241|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 143351488 (137M) [application/octet-stream]
Saving to: ‘Miniconda3-py312_24.3.0-0-Linux-x86_64.sh’

Miniconda3-py312_24 100%[===================>] 136.71M   126MB/s    in 1.1s

2024-10-24 09:43:13 (126 MB/s) - ‘Miniconda3-py312_24.3.0-0-Linux-x86_64.sh’ saved [143351488/143351488]

PREFIX=/usr/local
Unpacking payload ...

Installing base environment...

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

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

```
The following packages will be UPDATED:

  ca-certificates                     2024.3.11-h06a4308_0 --> 2024.9.24-h06a4308_0
  certifi                         2024.2.2-py312h06a4308_0 --> 2024.8.30-py312h06a4308_0
  conda                             24.3.0-py312h06a4308_0 --> 24.9.2-py312h06a4308_0
  openssl                                3.0.13-h7f8727e_0 --> 3.0.15-h5eee18b_0
```

```
openssl-3.0.15       | 5.2 MB    | :  41% 0.40600721210547036/1 [00:00<00:00,  1.50it/s]

certifi-2024.8.30    | 163 KB    | : 100% 1.0/1 [00:00<00:00,  3.42it/s]

certifi-2024.8.30    | 163 KB    | : 100% 1.0/1 [00:00<00:00,  3.42it/s]
conda-24.9.2         | 1.1 MB    | : 100% 1.0/1 [00:00<00:00,  1.57it/s]
```

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

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

```
conda 24.9.2
Python 3.12.2
```

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

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

```
#!conda install -c conda-forge matplotlib --yes
```

```
#!conda install -c conda-forge scipy --yes
```

```
#!conda install -c conda-forge openpyxl --yes
```

```
#!conda install -c conda-forge xlrd --yes
```

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 some information given in the Python setup file.

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_fmpy_explore.py

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

```
Cloning into 'BPL_TEST2_Perfusion'...
```

```
%cd BPL_TEST2_Perfusion
```

```
/content/BPL_TEST2_Perfusion
```

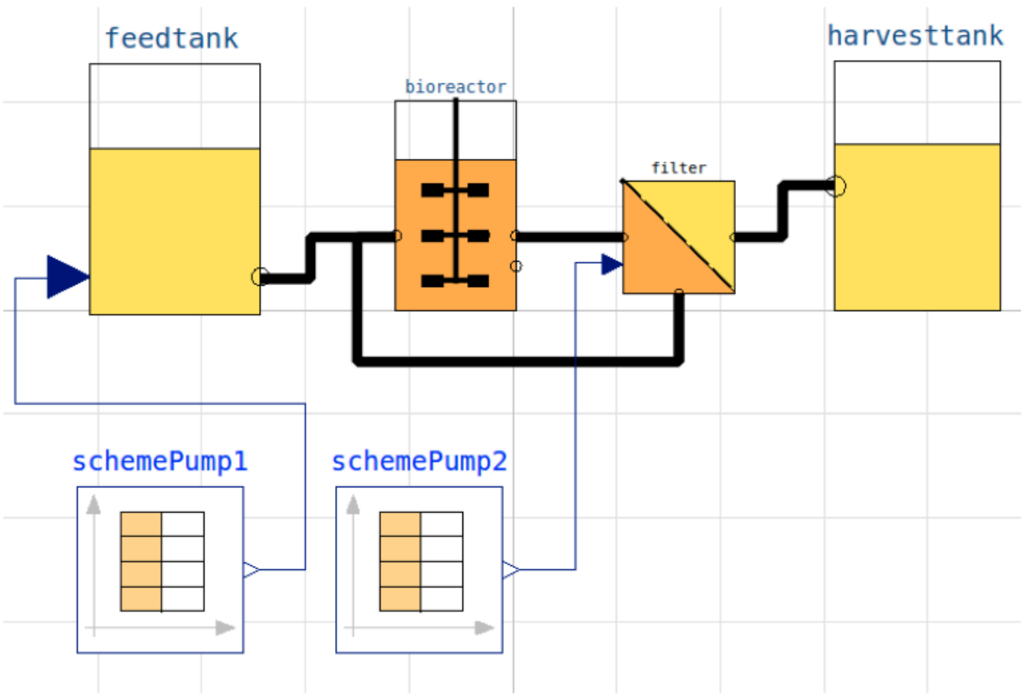
```
run -i BPL_TEST2_Perfusion_fmpy_explore.py
```

```
Linux - run FMU pre-comiled OpenModelica

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 simulation
and the command process_diagram() brings up the main configuration

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

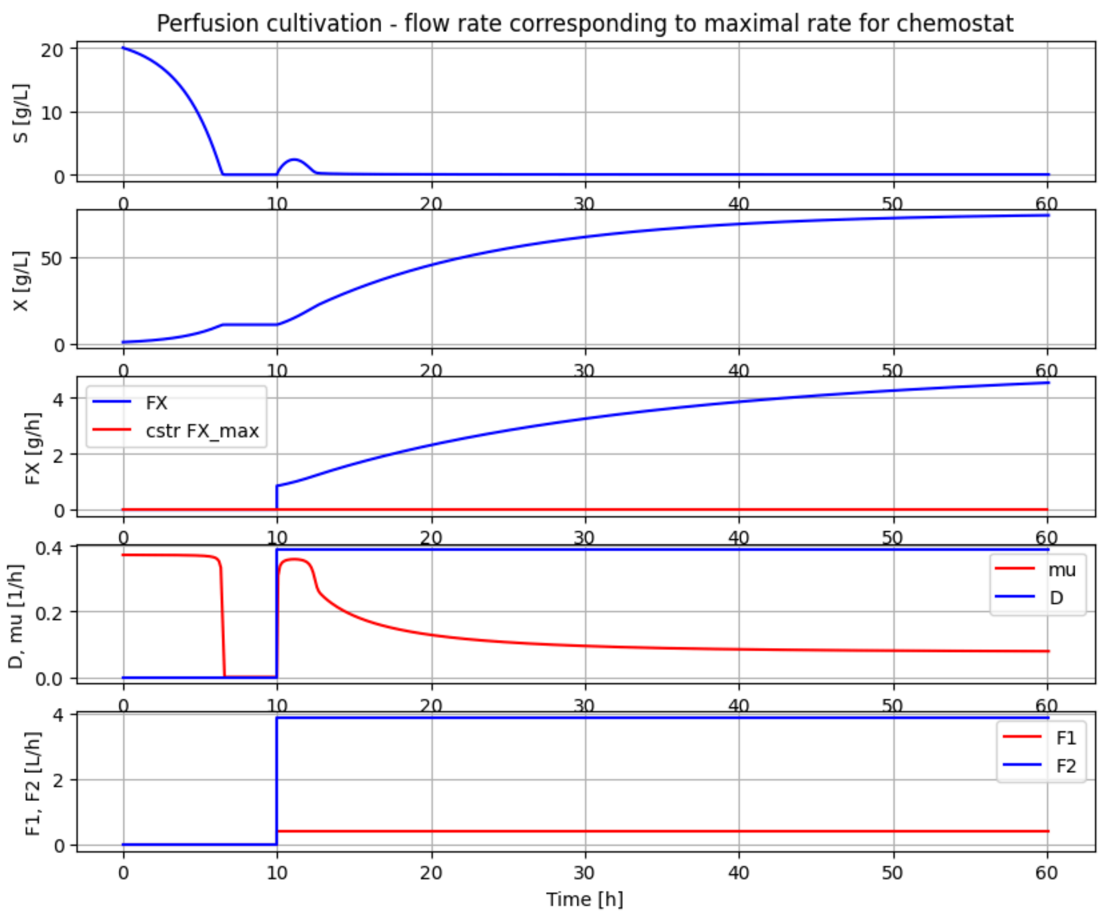
```
%matplotlib inline
plt.rcParams['figure.figsize'] = [25/2.54, 20/2.54]
```

```
process_diagram()
```

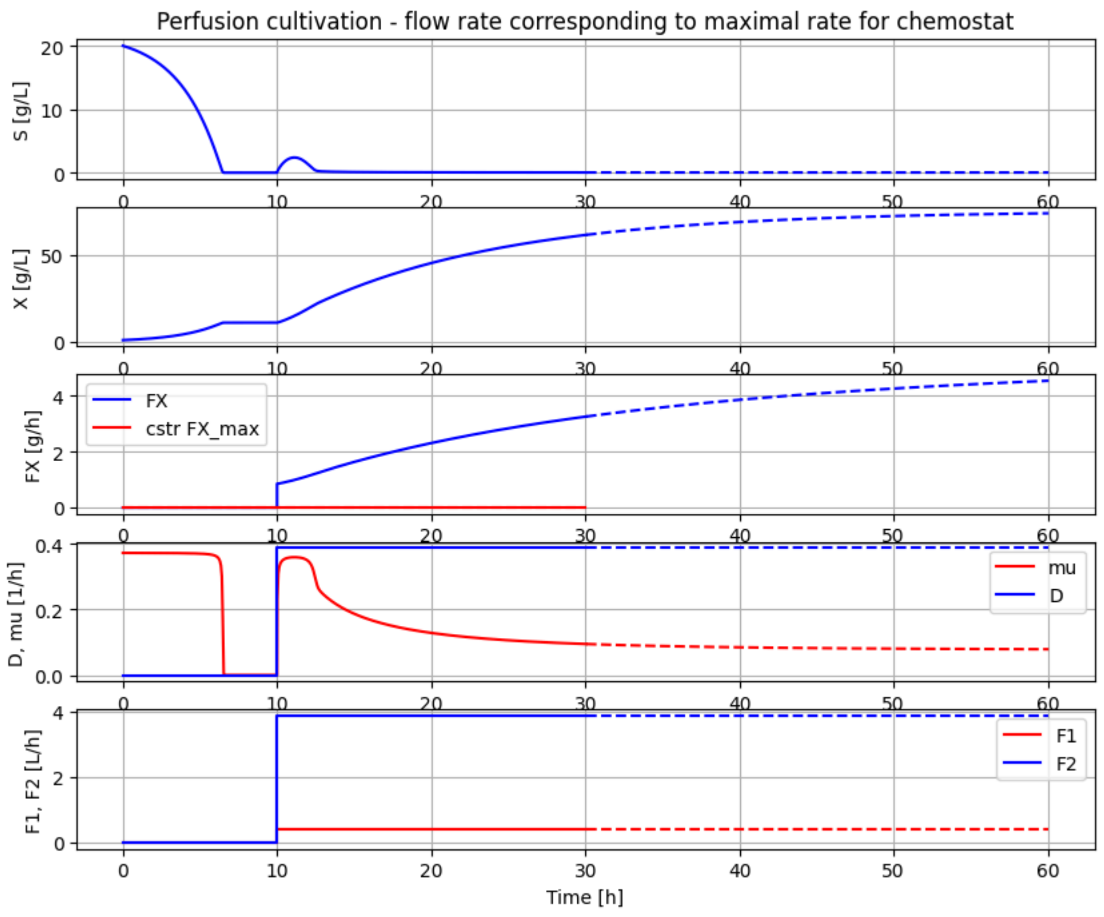
```
No processDiagram.png file in the FMU, but try the file on disk.
```



```
# 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 substrate conc
init(V_start=1.0, VX_start=1.0)                                   # Process initial conditions that are common
eps = parDict['filter_eps']                                       # Pump schedule parameter


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

init(VS_start=20)                                      # Process initial
par(pump1_t1=10, pump2_t1=10)                          # Pump schedule - recycle flow 10 times perfusion flow
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_t4=960)

newplot(title='Perfusion cultivation - flow rate corresponding to maximal rate for chemostat')
simu(60)
```



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

init(VS_start=20)                                       # Process initial
par(pump1_t1=10, pump2_t1=10)                           # Pump schedule - recycle flow 10 times perfusion flow
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_t4=960)

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



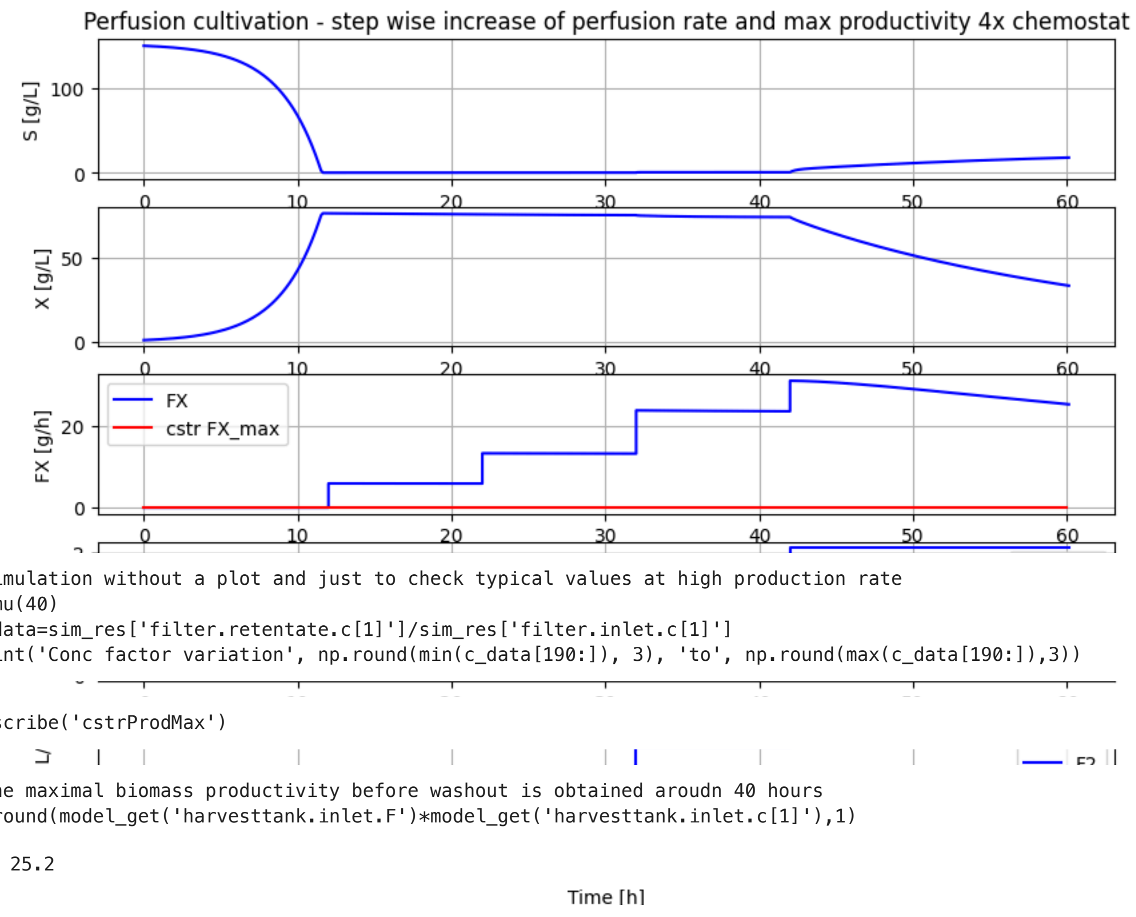
```
# Concentration factor of the filter
c=model_get('filter.retentate.c[1]')/model_get('filter.inlet.c[1]')
print('Conc factor of perfusion filter =', np.round(c,3))
```

```
Conc factor of perfusion filter = 1.179
```

```
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_data[151:]),3))
```

```
Conc factor variation 1.179 1.179
```

```
# 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 rate.

init(VS_start=150)                                  # Process initial varied

par(pump1_t1=12, pump2_t1=12)                       # Pump schedule - recycle flow 10 times perfusion flow
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 max productivity 4x chemostat')
simu(60)
```



```
# Simulation without a plot and just to check typical values at high production rate
#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(c_data[190:]),3))
```

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

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

```
25.2
```

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

```
4.2
```