

## ✓ BPL\_CHO\_Perfusion\_cspr\_opennloop script with FMPy

The key library FMPy is installed.

After the installation a small application BPL\_CHO\_Fedbatch 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/')
```

```
➤ --2024-08-13 09:36:15-- 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 75.8MB/s in 1.8s
```

```
2024-08-13 09:36:17 (75.8 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
```

```
➤
  ca-certificates                2024.3.11-h06a4308_0 --> 2024.7.2-h06a4308_0
  certifi                        2024.2.2-py312h06a4308_0 --> 2024.7.4-py312h06a4308_0
  conda                          24.3.0-py312h06a4308_0 --> 24.7.1-py312h06a4308_0
  openssl                        3.0.13-h7f8727e_0 --> 3.0.14-h5eee18b_0
```

```
frozendict-2.4.2      | 36 KB      | : 100% 1.0/1 [00:00<00:00, 3.10it/s]
```

```
frozendict-2.4.2      | 36 KB      | : 100% 1.0/1 [00:00<00:00, 3.10it/s]  
conda-24.7.1          | 1.2 MB     | : 100% 1.0/1 [00:00<00:00, 1.56it/s]
```

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

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

```
🔄 conda 24.7.1  
   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 matplotlib --yes
```

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

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

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

## ✓ BPL\_CHO\_Perfusion\_cspr\_openloop setup

Now specific installation and the run simulations. Start with connecting to Github. Then upload the two files:

- FMU - BPL\_CHO\_Perfusion\_cspr\_openloop\_linux\_om\_me.fmu
- Setup-file - BPL\_CHO\_Perfusion\_cspr\_openloop\_fmpy\_explore

```
# Filter out DeprecationWarnings for 'np.float as alias' is needed - wish I could make filter more narrow
import warnings
warnings.filterwarnings("ignore")
```

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

```
📂 Cloning into 'BPL_CHO_Perfusion_cspr'...
```

```
%cd BPL_CHO_Perfusion_cspr
```

```
📂 /content/BPL_CHO_Perfusion_cspr
```

## ✓ BPL\_CHO\_Perfusion\_cspr\_openloop - demo

Author: Jan Peter Axelson

This notebook is about CHO perfusion cultivation and focus on the concept of Cell Specific Perfusion Rate (CSPR). Recombinant protein production is included in the model but not shown in the diagrams here.

The model used here was originally developed and validated for fed-batch cultivation of CHO cultures [1], but also got some influence from perfusion cultivation [2]. The model emphasize the bottlenecks of metabolism and growth and inspired from similar models of microbial cultures. One interesting aspect is that the model brings a theoretical base for the usefulness of the CSPR concept often used experimentally [4] and [5]. The results of simulation shown here are in accordance with experiments in a qualitative way, however not tested more quantitatively, to the authors knowledge.

The main result is that when the perfusion rate should be increased from one set-point to another, it should be changed slowly. An abrupt step-wise increase should be avoided. A slow increase in the perfusion rate minimize the risk of unnecessary by-product formation of lactate and ammonia and make the culture increase in cell concentration quicker to a new steady-state compared to a step-wise perfusion rate change. The rate of change of perfusion rate should not exceed the maximal culture growth rate.

The strategy to limit rate of change of perfusion rate is much more important for mammalian cultures than for microbial cultures. The reason is that mammalian cell cultures increase in cell density after an increase in perfusion rate, while microbial cultures remains constant in cell density. This difference is most likely due to the fact that maintenance metabolism is a larger part in mammalian cultures and at higher growth rates the percentage of maintenance becomes smaller and the overall yield increase, while microbial cultures has a much more constant yield over a broad range of growth rates. These ideas are discussed in section 2.2 in [3], together with other insights into perfusion cultivation obtained from simulation studies of the current model.

Once again, it would be good to compare simulations with relevant experimental data from perfusion cultivation. Hopefully, the material presented here can stimulate work in this direction.

## ✓ Setup of the simulation model

```
run -i BPL_CHO_perfusion_cspr_openloop_explore_fmpy.py
```

```
📂 Linux - run FMU pre-compiled OpenModelica 1.23.0-dev
```

```
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]

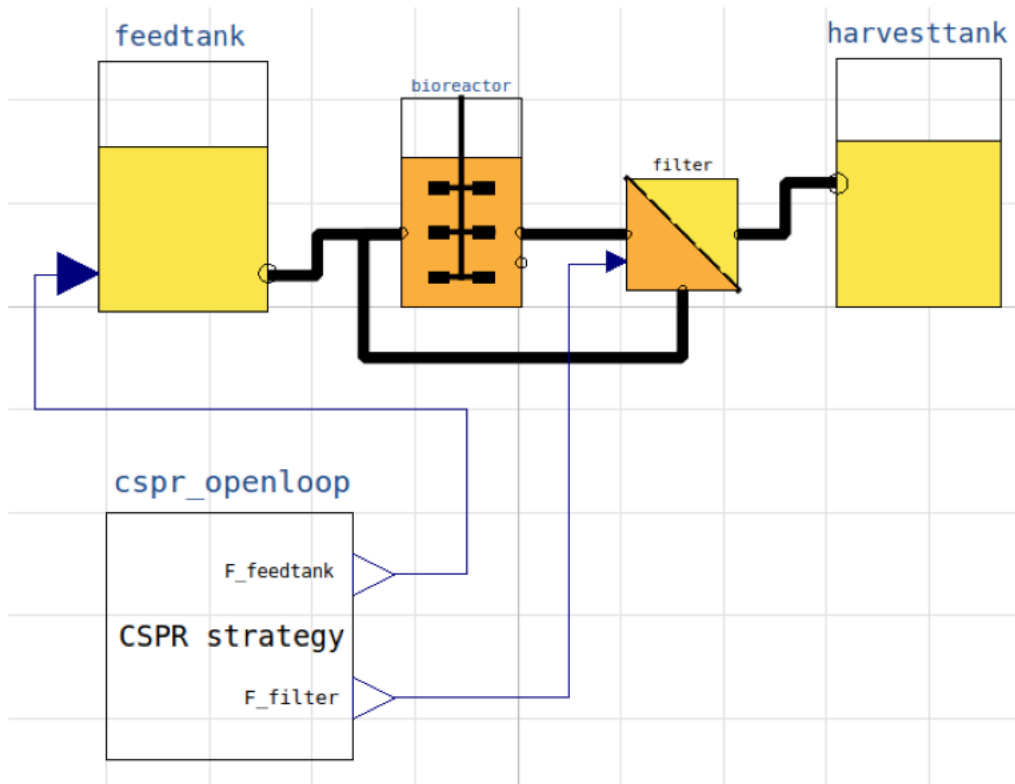
```

## ✓ About the process model

Here a process diagram is shown of the process. Further information about the culture stored in the model code is extracted.

```
process_diagram()
```

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



```
describe('culture'); print(); #describe('liquidphase')
```

🔗 Reactor culture CHO-MAB – cell line HB-58 American Culture Collection ATCC

## ✓ Simulaton study

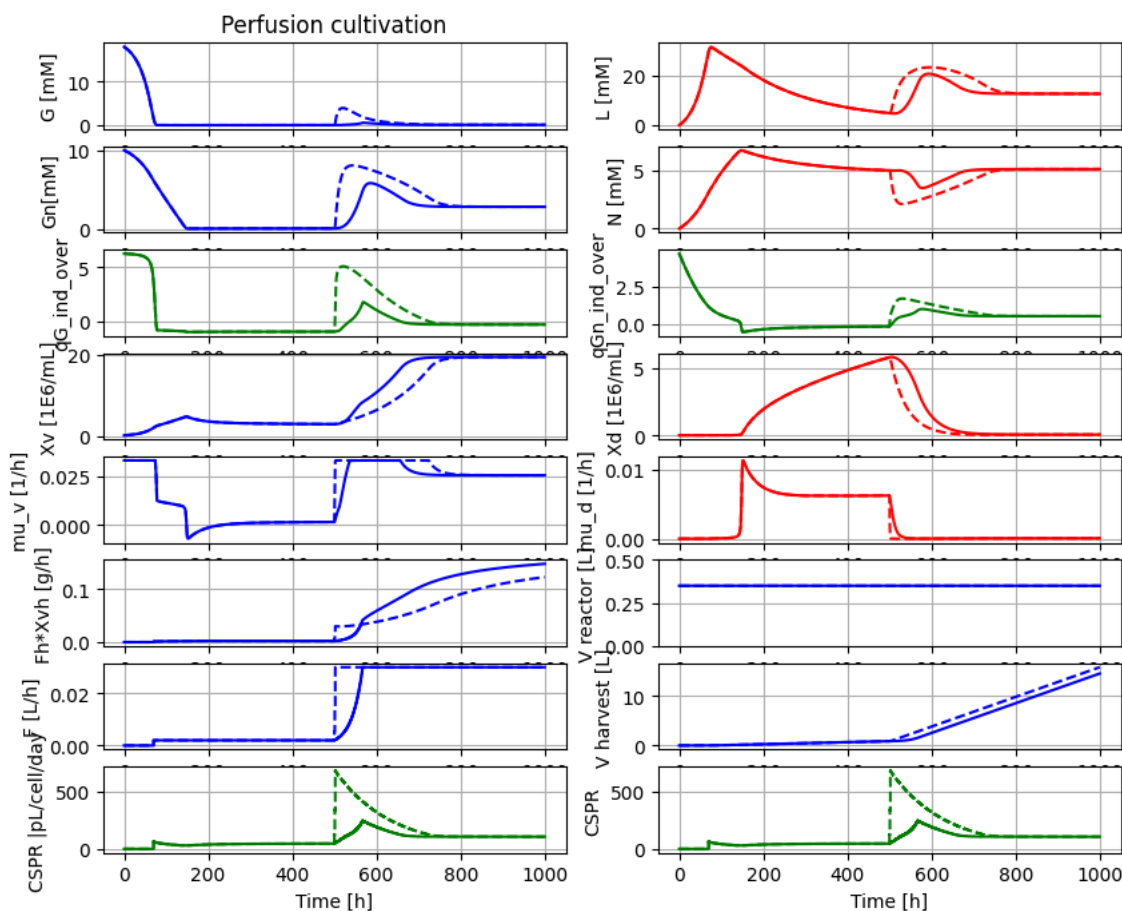
The simulations are run with default parameters from [1]. The parameters for the change of perfusion rate is done here. The parameter `mu_ref` gives the rate of increase in [1/h]. With the value `mu_ref=1` you get an abrupt step directly to the set-point. The parameters `F1` and `F2` give the flow rate [L/h] before and after the change. The corresponding time parameters `t1` and `t2` are given by default.

```

newplot('Perfusion cultivation', plotType='Extended')

par(samplePeriod=1); par(F1=0.0020, F2=0.030) # General parameters
par(mu_ref=0.04); simu()                      # First simulation (solid)
par(mu_ref=1); simu()                         # Second simulation (dashed)

```



We see here the shorter settling time for cell conc  $X_v$  when the slower exponential increase of perfusion rate is used, compared to an abrupt change.

The different sets of parameters that can be changed are shown below

```
disp(mode='long')
```



```
bioreactor.V_start : V_start : 1.0
bioreactor.m_start[1] : VXv_start : 0.0
bioreactor.m_start[2] : VXd_start : 0.0
bioreactor.m_start[3] : VG_start : 0.0
bioreactor.m_start[4] : VGn_start : 0.0
bioreactor.m_start[5] : VL_start : 0.0
bioreactor.m_start[6] : VN_start : 0.0
bioreactor.culture.qG_max1 : qG_max1 : 0.297
bioreactor.culture.qG_max2 : qG_max2 : 0.038
bioreactor.culture.qGn_max1 : qGn_max1 : 0.124
bioreactor.culture.qGn_max2 : qGn_max2 : 0.022
bioreactor.culture.mu_d_max : mu_d_max : 0.13
bioreactor.broth_decay.k_lysis : k_lysis : 0.0
filter.eps : eps : 0.05
filter.alpha[1] : alpha_Xv : 0.5
filter.alpha[2] : alpha_Xd : 0.5
filter.alpha[3] : alpha_G : 0.5
filter.alpha[4] : alpha_Gn : 0.5
filter.alpha[5] : alpha_L : 0.5
filter.alpha[6] : alpha_N : 0.5
filter.alpha[7] : alpha_Pr : 0.5
feedtank.c_in[3] : G_in : 0.0
feedtank.c_in[4] : Gn_in : 0.0
cspr_opennloop.samplePeriod : samplePeriod : 1.0
cspr_opennloop.mu_ref : mu_ref : 0.03
cspr_opennloop.t1 : t1 : 70.0
cspr_opennloop.F1 : F1 : 0.002
cspr_opennloop.t2 : t2 : 500.0
cspr_opennloop.F2 : F2 : 0.03
```

Unsupported cell type. Double-click to inspect/edit the content.

```
describe('bioreactor.broth_decay.k_decay')
```



```
Rate of decay of viable cells : 0.0 [ 1E9 cells/(L*h) ]
```

```
describe('k_lysis')
```

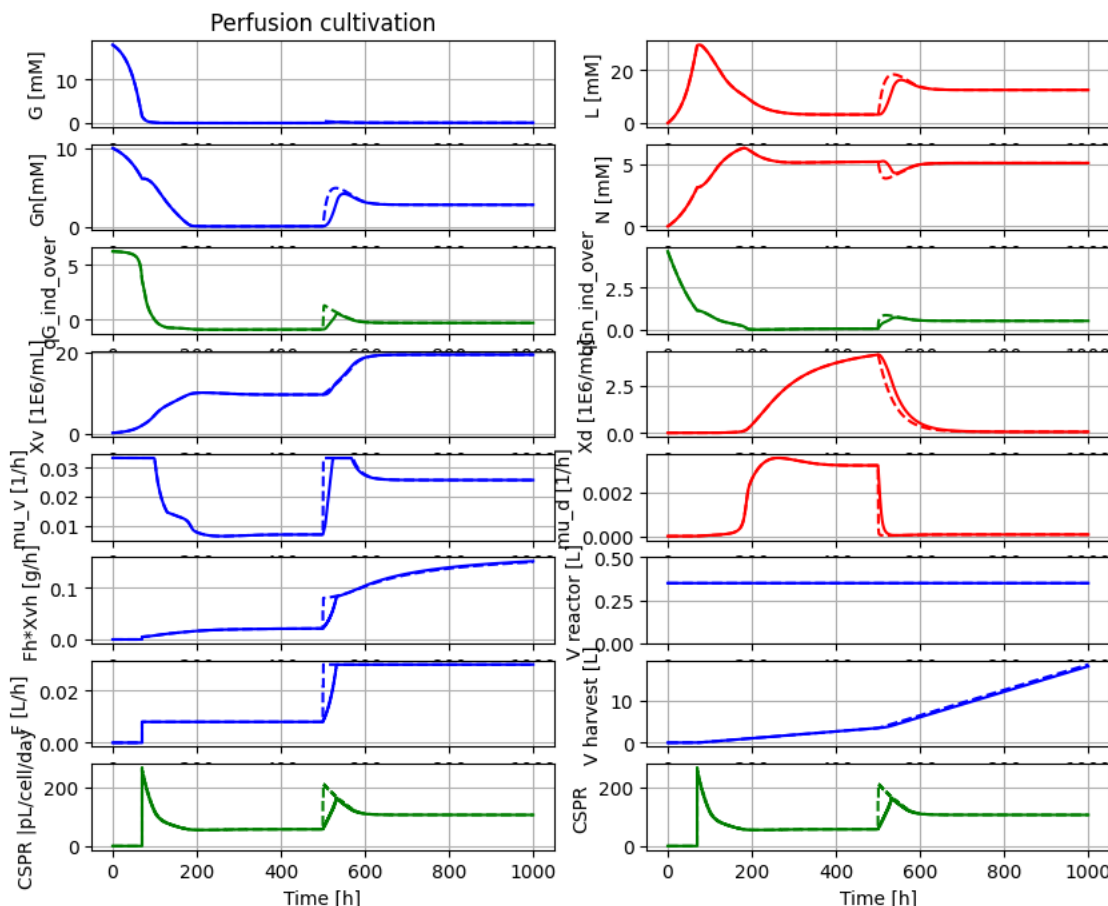
```
↗ Specific rate of lysis of dead cells : 0.0 [ 1/h ]
```

Now a new simulation where the flow rate F1 is higher before change to the new set-point F2.

```
newplot('Perfusion cultivation', plotType='Extended')
```

```
par(samplePeriod=1); par(F1=0.0080, F2=0.03)      # General parameters
par(mu_ref=0.04); simu()                          # First simulation (solid)
par(mu_ref=1); simu()                             # Second simulation (dashed)
```

```
↗
```



We see that a somewhat smaller change in perfusion rate make the difference in settling time of  $X_v$  for the two strategies to almost disappear.

## Summary

The simulation study shows that a slow increase in the perfusion rate that match the maximal culture growth rate gives a quicker increase in cell density than an abrupt step-wise change of perfusion rate.

The effect is more pronounced for a larger change of perfusion rate than for a smaller.

Here the change of perfusion rate was done using a pre-calculated exponential scheme, i.e. open-loop control. A more robust method would be to measure the cell concentration on-line and adjust the perfusion rate change to more exactly maintain constant CSpr.

It is of interest to confirm the results with experimental data.

## References

- [1] Amribt, Z., Niu, H. and Bogaerts P.: "Macroscopic modelling of overflow metabolism and model based optimization of hybridoma cell fed-batch cultures.", Biochem. Eng. Journal, 2013.
- [2] Niu, H., Amribt, Z., Fickers, P., Tan, W. and Bogaerts P.: "Metabolic pathway analysis and reduction for mammalian cell cultures - towards macroscopic modelling", Chem. Eng. Science, 2013.
- [3] Axelsson, J. P.: "Simplified model of CHO-cultivation in Bioprocess Library for Modelica - some experience", conference paper 22nd NPCW Lyngby, Denmark, August 22-23, 2019.
- [4] Hu, W-S: "Cell culture bioprocess engineering", 2nd edition, CRC Press, 2020.

[5] Konstantinov, K. et al: "The push-to-low" approach for optimization of high density perfusion cultures of animal cells", Adv Biochem Engin/Biotechnol, 2006.

## ▼ Appendix

```
describe('parts')
```

```
↔ ['bioreactor', 'bioreactor.broth_decay', 'bioreactor.culture', 'CSPR', 'cspr_openloop', 'D', 'feedtank', 'filter', 'harv
```

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

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

[+ Code](#)[+ Text](#)

```
system_info()
```



```
System information
-OS: Linux
-Python: 3.10.12
-Scipy: not installed in the notebook
-FMPy: 0.3.20
-FMU by: OpenModelica Compiler OpenModelica 1.23.0~dev.beta.1-1-g379f714
-FMI: 2.0
-Type: ME
-Name: BPL_CHO.Perfusion_cspr_openloop
-Generated: 2024-05-20T19:01:47Z
-MSL: 3.2.3
-Description: Bioprocess Library version 2.2.0
-Interaction: FMU-explore for FMPy version 1.0.1
```

Start coding or [generate](#) with AI.