

BPL_IEC_operation script with FMPy

The key library FMPy is installed.

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

```
In [1]: !lsb_release -a # Actual VM Ubuntu version used by Google
```

```
No LSB modules are available.  
Distributor ID: Ubuntu  
Description:    Ubuntu 22.04.4 LTS  
Release:        22.04  
Codename:       jammy
```

```
In [2]: !python --version
```

```
Python 3.11.11
```

```
In [3]: !pip install fmpy
```

```
Collecting fmpy  
  Downloading FMPy-0.3.22-py3-none-any.whl.metadata (1.9 kB)  
Requirement already satisfied: attrs in /usr/local/lib/python3.11/dist-packages (from fmpy) (25.3.0)  
Requirement already satisfied: Jinja2 in /usr/local/lib/python3.11/dist-packages (from fmpy) (3.1.6)  
Collecting lark (from fmpy)  
  Downloading lark-1.2.2-py3-none-any.whl.metadata (1.8 kB)  
Requirement already satisfied: lxml in /usr/local/lib/python3.11/dist-packages (from fmpy) (5.3.1)  
Requirement already satisfied: msgpack in /usr/local/lib/python3.11/dist-packages (from fmpy) (1.1.0)  
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from fmpy) (2.0.2)  
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from Jinja2->fmpy) (3.0.2)  
Downloading FMPy-0.3.22-py3-none-any.whl (4.9 MB)  
----- 4.9/4.9 MB 16.4 MB/s eta 0:00:00  
Downloading lark-1.2.2-py3-none-any.whl (111 kB)  
----- 111.0/111.0 kB 4.4 MB/s eta 0:00:00  
Installing collected packages: lark, fmpy  
Successfully installed fmpy-0.3.22 lark-1.2.2
```

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

- FMU - BPL_IEC_operation_linux_om_me.fmu
- Setup-file - BPL_IEC_operation_fmpy_explore.py

```
In [4]: %%bash
git clone https://github.com/janpeter19/CONF_2023_08_NPCW24
```

Cloning into 'CONF_2023_08_NPCW24'...

```
In [5]: %cd CONF_2023_08_NPCW24

/content/CONF_2023_08_NPCW24
```

BPL_IEC_operation

Authors: Karl Johan Brink and Jan Peter Axelsson

In this notebook we show operation of a typical ion-exchange chromatography step. The impact of pH is also illustrated.

The model is based on the simplified model [1].

```
In [6]: run -i BPL_IEC_fmпы_explore.py
```

Linux - run FMU pre-compiled OpenModelica

Model for the process 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

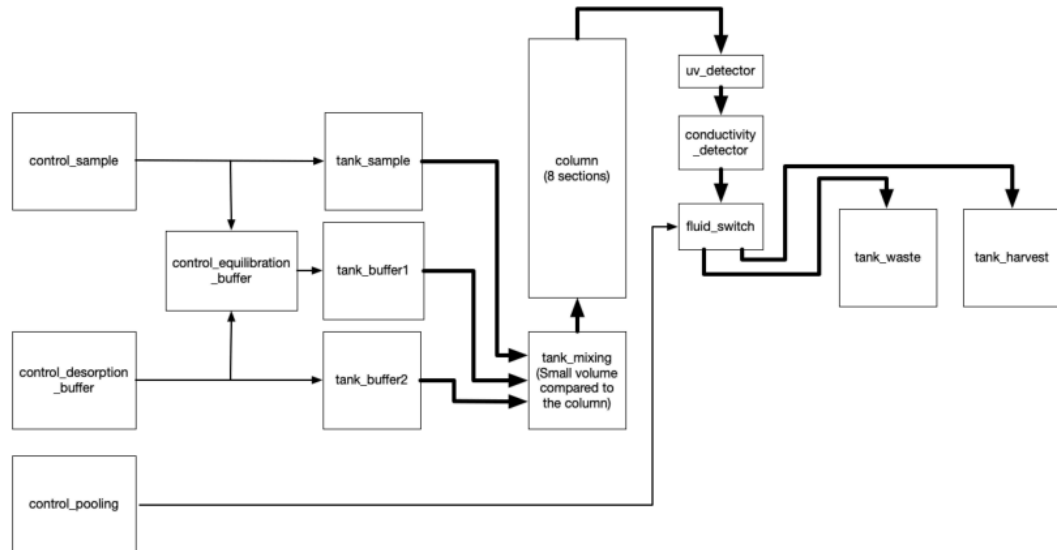
Brief information about a command by help(), eg help(simu)

Key system information is listed with the command system_info()

```
In [7]: plt.rcParams['figure.figsize'] = [30/2.54, 24/2.54]
```

```
In [8]: process_diagram()
```

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



1 Typical parameters an ion exchange chromatography column step

In [9]: *# From given column height (h) diameter (d) and Linear flow rate (lfr)
actual column volume (V) and volume flow rate (VFR) are calculated below.*

```

from numpy import pi
h = 20.0
d = 1.261
a = pi*(d/2)**2
V = h*a
print('V =', np.round(V,1), '[mL]')

lfr = 48
VFR = a*lfr/60
print('VFR =', np.round(VFR,1), '[mL/min]')

```

V = 25.0 [mL]
VFR = 1.0 [mL/min]

In [10]: *# Sample concentration product P_in and antagonist A_in*
par(P_in = 1.0)
par(A_in = 1.0)
par(E_in = 0.0)

Column properties are described by the size and binding capacity of the resin Q_a
par(height = h)
par(diameter = d)
par(Q_av = 6.0)

Resin parameters - default values used

Remaining salt koncentration in the column from prvious batch and eliminated duri
init(E_0 = 50)

Salt concentration of the desorption buffer

```

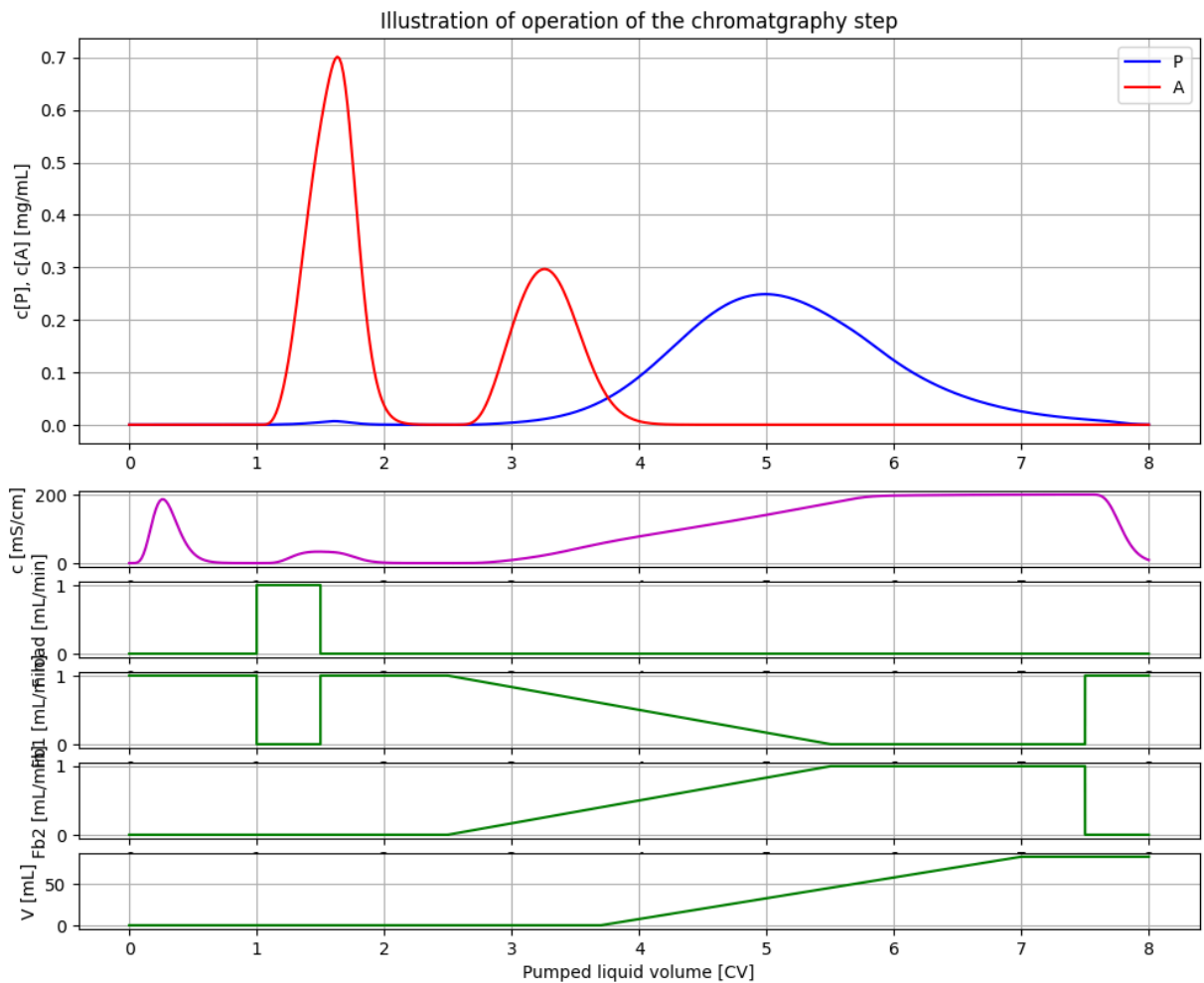
par(E_in_desorption_buffer = 8.0)

# Flow rate rate through the
par(LFR=lfr)

# Switching points during operation are conveniently described in terms of multiple
CV_ekv = 1.0
CV_ads = 0.5
CV_wash = 1.0
CV_desorb = 3.0
CV_start_pool = 1.2
CV_stop_pool = 4.5
CV_wash2 = 2.5
par(scale_volume=True, start_adsorption=CV_ekv*V, stop_adsorption=(CV_ekv+CV_ads)*V
par(start_desorption=(CV_ekv+CV_ads+CV_wash)*V, stationary_desorption=(CV_ekv+CV_ad
par(stop_desorption=7.5*V)
par(start_pooling=(CV_ekv+CV_ads+CV_wash+CV_start_pool)*V, stop_pooling=(CV_ekv+CV_

# Simulation and plot of results
newplot(title='Illustration of operation of the chromatgraphy step', plotType='Elut
simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

```



Comments of steps of operations:

1. Time: 0-1 hours - equilibration. Just to illustrate the equilibration process the first part of the column is given an initial value of salt concentration.
2. Time: 1-1.5 hours - sample is loaded on the column. The product P is adsorbed to the column and just a small amount passes through and goes to the waste. The antagonist A is much less adsorbed.
3. Time: 1.5-2.5 hours - washing 1. The column comes to equilibrium and both antagonist and product comes down to low levels.
4. Time: 2.5-5.5 hours - desorption. A linear gradient of increasing salt concentration is applied. First the antagonist and later the product comes out.
5. Time: 5.5-7.5 hours - washing 2. The column has constant salt concentration and stationary desorption.
6. Time: 3.7-7.0 hours - pooling of product. The start- and stop of pooling are chosen with trade-off between maximizing the product pooled and minimize the amount of antagonist in the pooling.
7. Time: 7.5-8.0 hours - desorption stopped and salt is washed out and preparation of the next batch to come.

Note that step 4 and 5 is parallel to step 6.

```
In [11]: # Check mass-balance of P and A
P_mass = model_get('tank_harvest.m[1]') + model_get('tank_waste.m[1]')
A_mass = model_get('tank_harvest.m[2]') + model_get('tank_waste.m[2]')
print('P_mass [mg] =', P_mass)
print('A_mass [mg] =', A_mass)
```

```
P_mass [mg] = 12.422130344708323
A_mass [mg] = 12.488781164505506
```

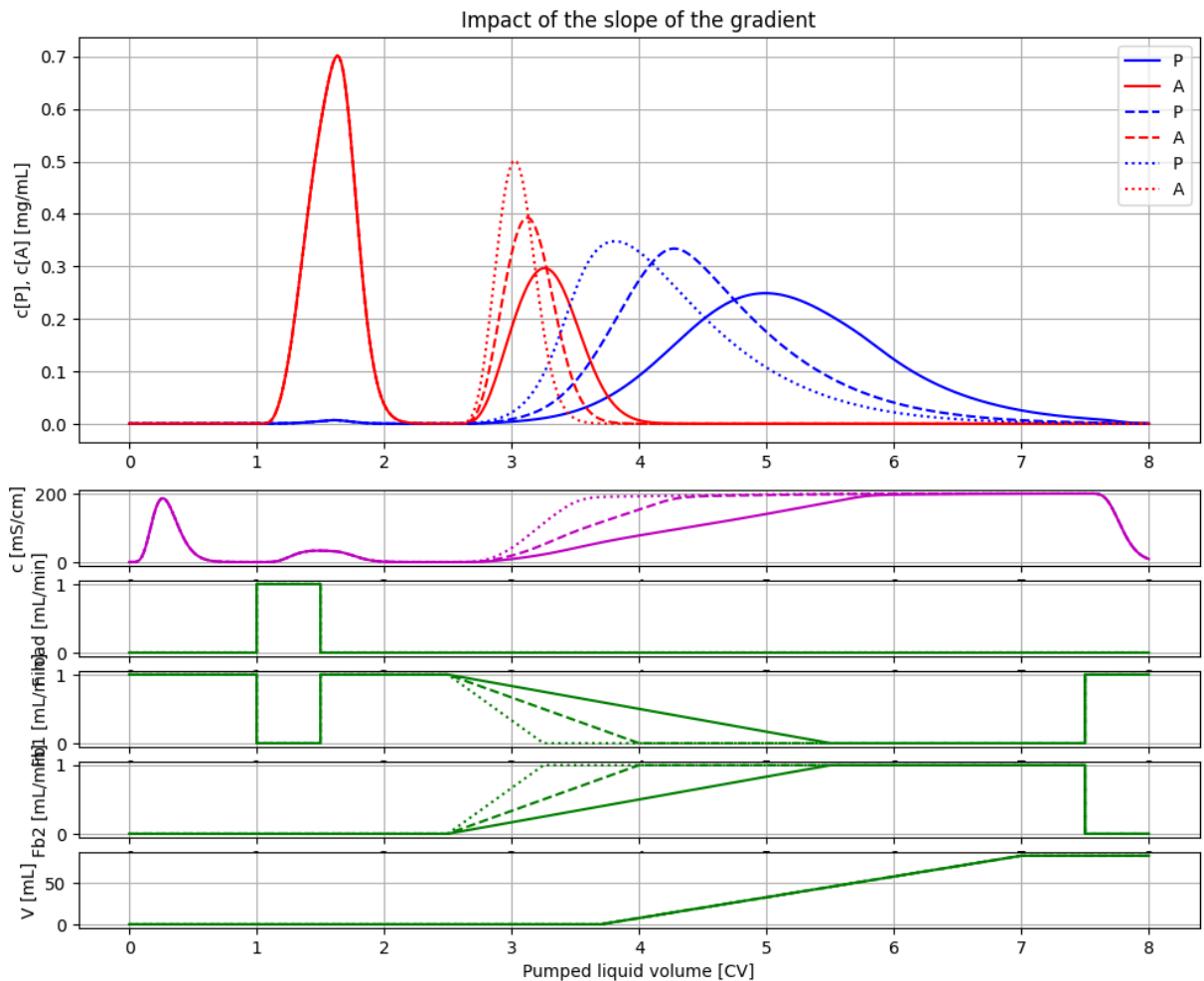
2 The impact of the slope of the desorption gradient

```
In [12]: # Simulations showing the impact of change of slope of the desorption gradient
newplot(title='Impact of the slope of the gradient', plotType='Elution-conductivity')

# Same gradient as before
par(start_desorption=(CV_ekv+CV_ads+CV_wash)*V, stationary_desorption=(CV_ekv+ CV_a
par(stop_desorption=7.5*V)
simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

# Gradient finishes after 0.5 of the volume
par(stationary_desorption = (CV_ekv + CV_ads + CV_wash + 0.5*CV_desorb)*V )
simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

# Gradient finishes after 0.25 of the volume
par(stationary_desorption = (CV_ekv + CV_ads + CV_wash + 0.25*CV_desorb)*V )
simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)
```



3 The impact of salt concentration in the sample

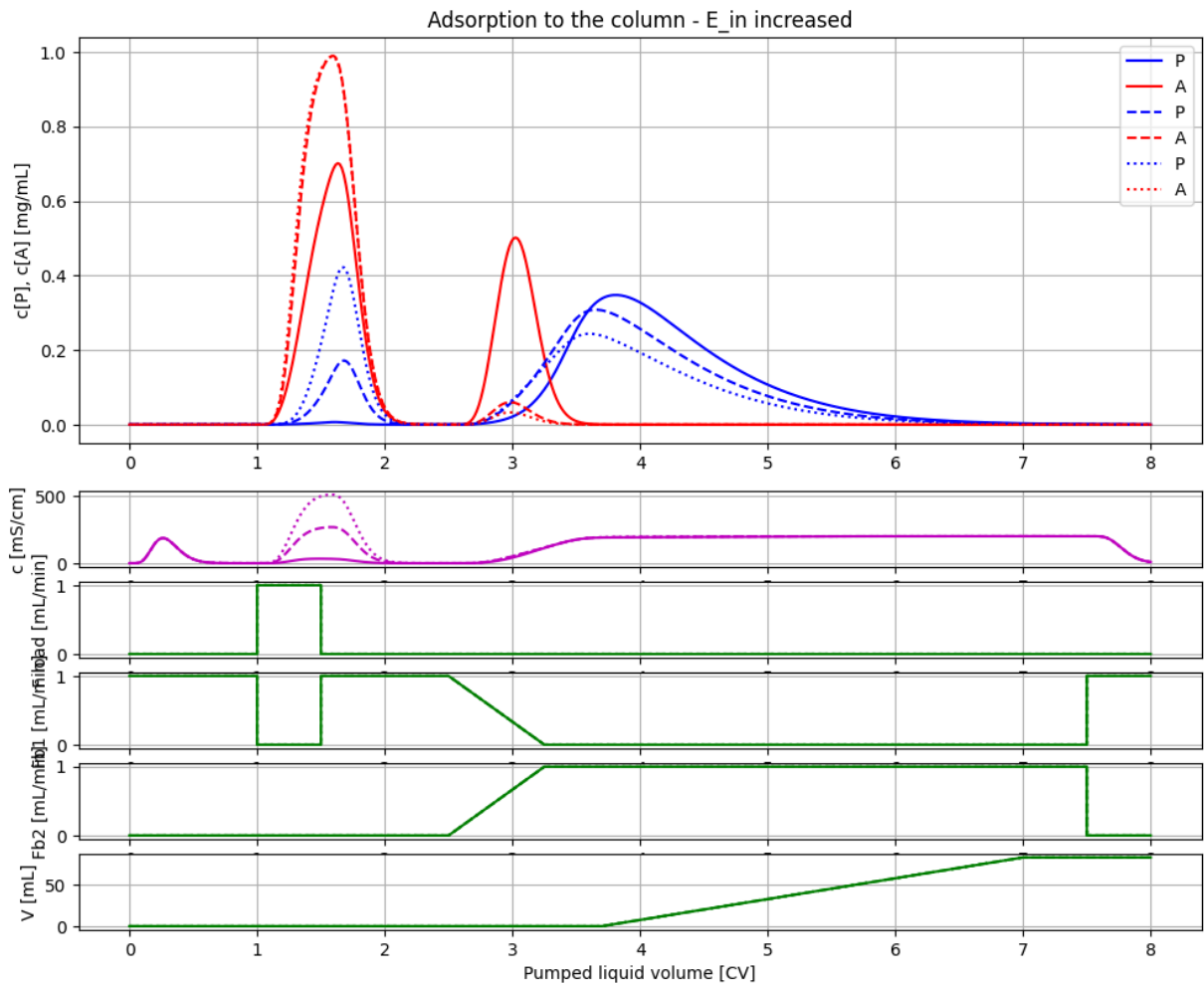
These values should be compared with the expected value 12.5 mg, i.e. half a column volume with sample concentration 1 mg/L. The difference is due to numerical errors during simulation.

```
In [13]: # Let us investigate the impact of increasing salt concentration in the sample E_in

# Simulate and plot the results
newplot(title='Adsorption to the column - E_in increased', plotType='Elution-conduc

for value in [0, 10, 20]:
    par(E_in=value)
    simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

# Restore default values
par(k2=0.05, k4=0.3, E_in=0)
```



Note, that increased salt concentration in the sample affect binding of both proteins. During adsorption less is bound. During desorption less product P can be harvested but the fraction of antagonist A may be lowered. Thus, some product is lost but the quality in terms of purity is improved.

4 The impact of change of binding strength due to pH

There are many factors that contribute to the binding strength. A most important factor is the pH-value of the resin and the characteristic iso-electric point of the protein. The binding strenght can be seen as proportional to the difference.

The binding strength of the resin is described by the quotient $K_P = k_1/k_2$ for the protein P and similarly $K_A = k_3/k_4$ for the protein A.

Below a few help-functions that describe this idea of the pH difference and its impact on binding strength in terms of the parameters k_1 , k_2 , k_3 , and k_4 of the protein-resin interaction.

```
In [14]: # Define function that describe the proportionality of binding strength of
# the pH difference of the iso-electric point and the resin

def KP_pH_sensitivity(pI_P=8.0, pH_resin=7.0):
    K_P_nom = 0.0
    coeff_pH = 6.0
    return K_P_nom + coeff_pH*(pI_P-pH_resin)

def KA_pH_sensitivity(pI_A=7.1667, pH_resin=7.0):
    K_A_nom = 0.0
    coeff_pH = 1.0
    return K_A_nom + coeff_pH*(pI_A-pH_resin)

def par_pH(pI_P=8.0, pI_A=7.1667, pH_resin=7.0, TP=3.33, TA=20.0):
    if (pI_P > pH_resin) & (pI_A > pH_resin):
        par(k2 = 1/(TP*KP_pH_sensitivity(pI_P=pI_P, pH_resin=pH_resin)))
        par(k4 = 1/(TA*KA_pH_sensitivity(pI_A=pI_A, pH_resin=pH_resin)))
    else:
        print('Both pI_P > pH_resin and pI_A > pH_resin must hold - no parameter ch
```

```
In [15]: # The default parameters of the column
disp('column')
```

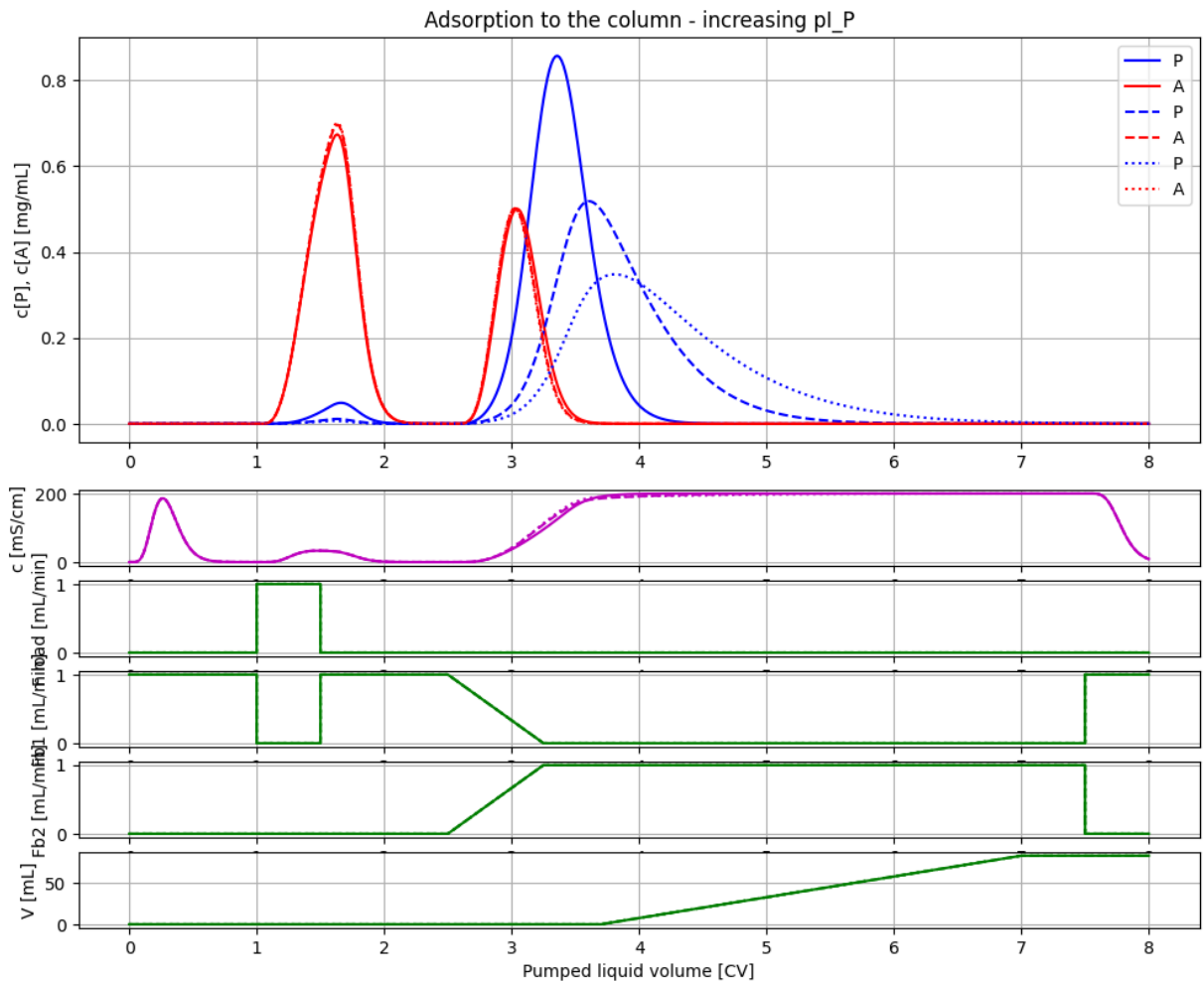
```
diameter : 1.261
height : 20.0
x_m : 0.3
k1 : 0.3
k2 : 0.05
k3 : 0.05
k4 : 0.3
Q_av : 6.0
E_0 : 50
```

```
In [16]: # Let us investigate the impact of change of the iso-electric pH for protein P

# Simulate and plot the results
newplot(title='Adsorption to the column - increasing pI_P', plotType='Elution-condu

for value in [7.2, 7.6, 8.0]:
    par_pH(pI_P=value, pI_A=7.1667, pH_resin=7.0)
    simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

# Restore default values
par(k2 = 0.05, k4 = 0.3)
```

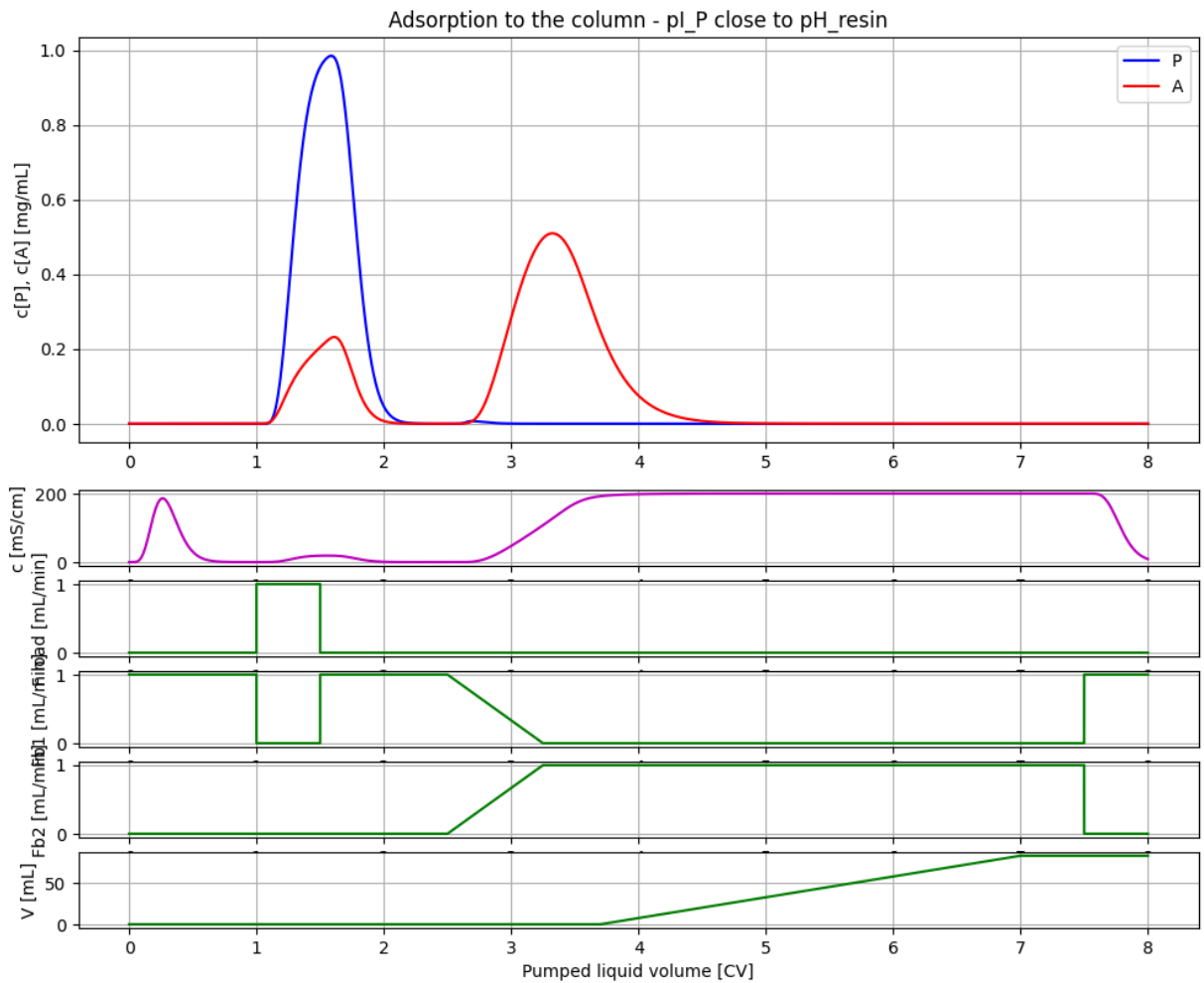
Note, with increasing pI_P the binding of P increase which leads less loss of product during adsorption. During desorption the peak height is lower with increasing binding strenght, but the total amoingt of product P that can be harvested is higher, due to the smaller loss during adsorption.

```
In [17]: # Let us investigate the impact of pI_P close to pH_resin

# Simulate and plot the results
newplot(title='Adsorption to the column - pI_P close to pH_resin', plotType='Elutio

for value in [7.0001]:
    par_pH(pI_P=value, pI_A=8)
    simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

# Restore default values
par(k2=0.05, k4=0.3)
```

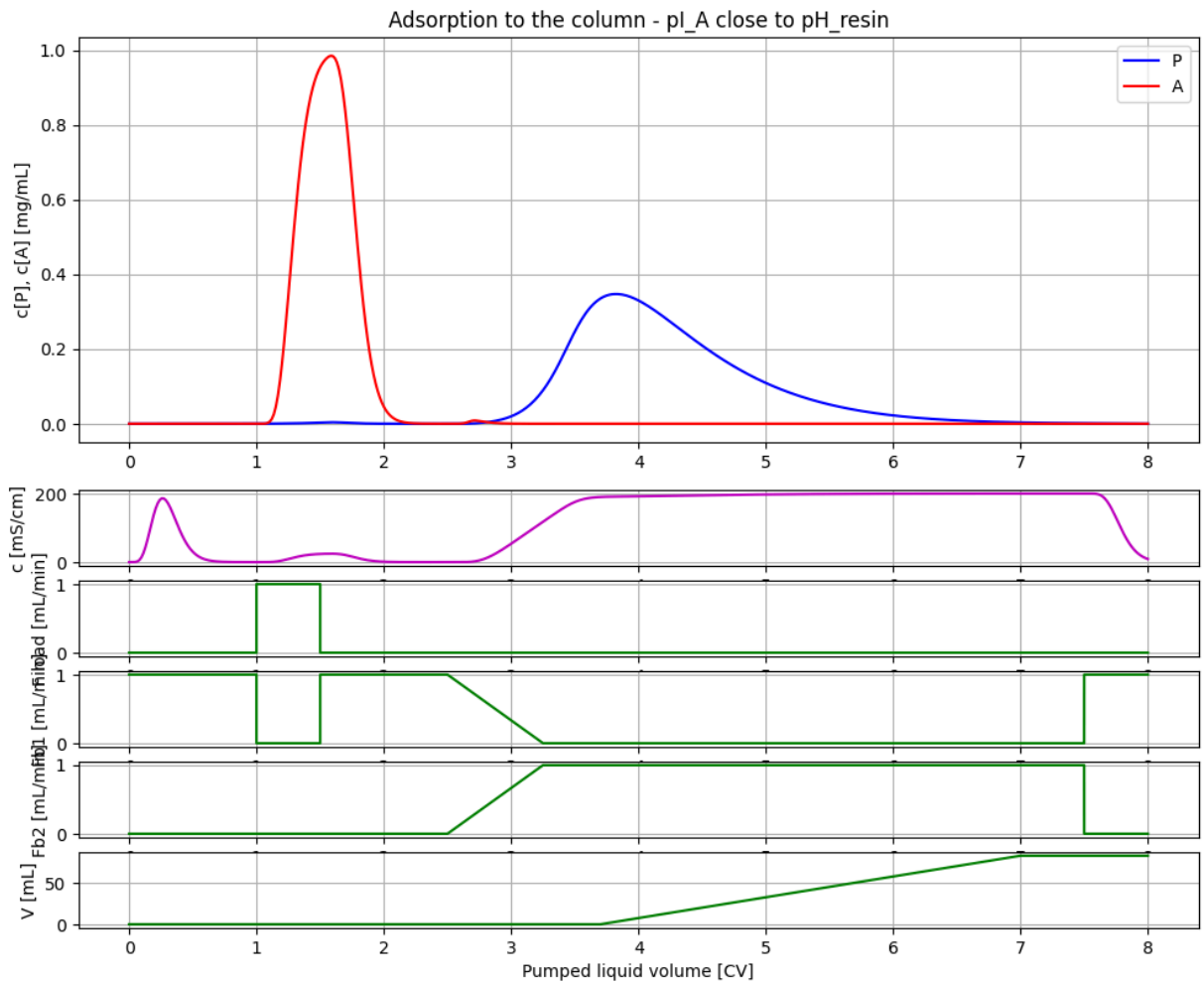


```
In [18]: # Let us investigate the impact of pI_A close to pH_resin

# Simulate and plot the results
newplot(title='Adsorption to the column - pI_A close to pH_resin', plotType='Elutio

for value in [7.001]:
    par_pH(pI_P=8.0, pI_A=value)
    simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

# Restore default values
par(k2=0.05, k4=0.3)
```

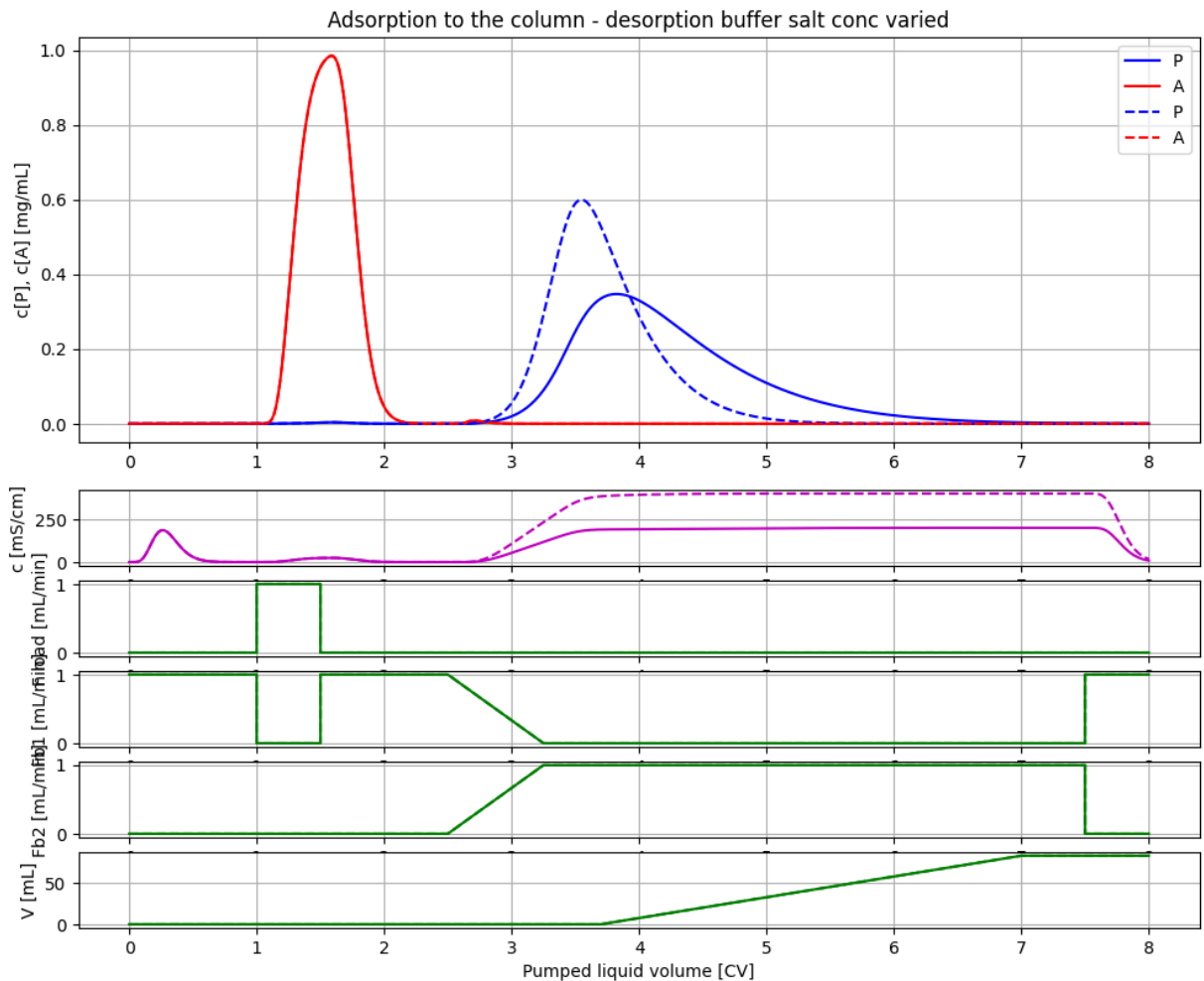


```
In [19]: # Let us also investigate the impact of salt concentration of the desorption buffer

# Simulate and plot the results
newplot(title='Adsorption to the column - desorption buffer salt conc varied', plot

for value in [8.0, 16.0]:
    par(E_in_desorption_buffer=value)
    par_pH(pI_P=8.0, pI_A=7.001, pH_resin=7.0)
    simu((CV_ekv+CV_ads+CV_wash+CV_desorb+CV_wash2)*V/VFR)

# Restore default values
par(E_in_desorption_buffer=8.0)
par(k2=0.05, k4=0.3)
```



5 Breakthrough curve often used during process development

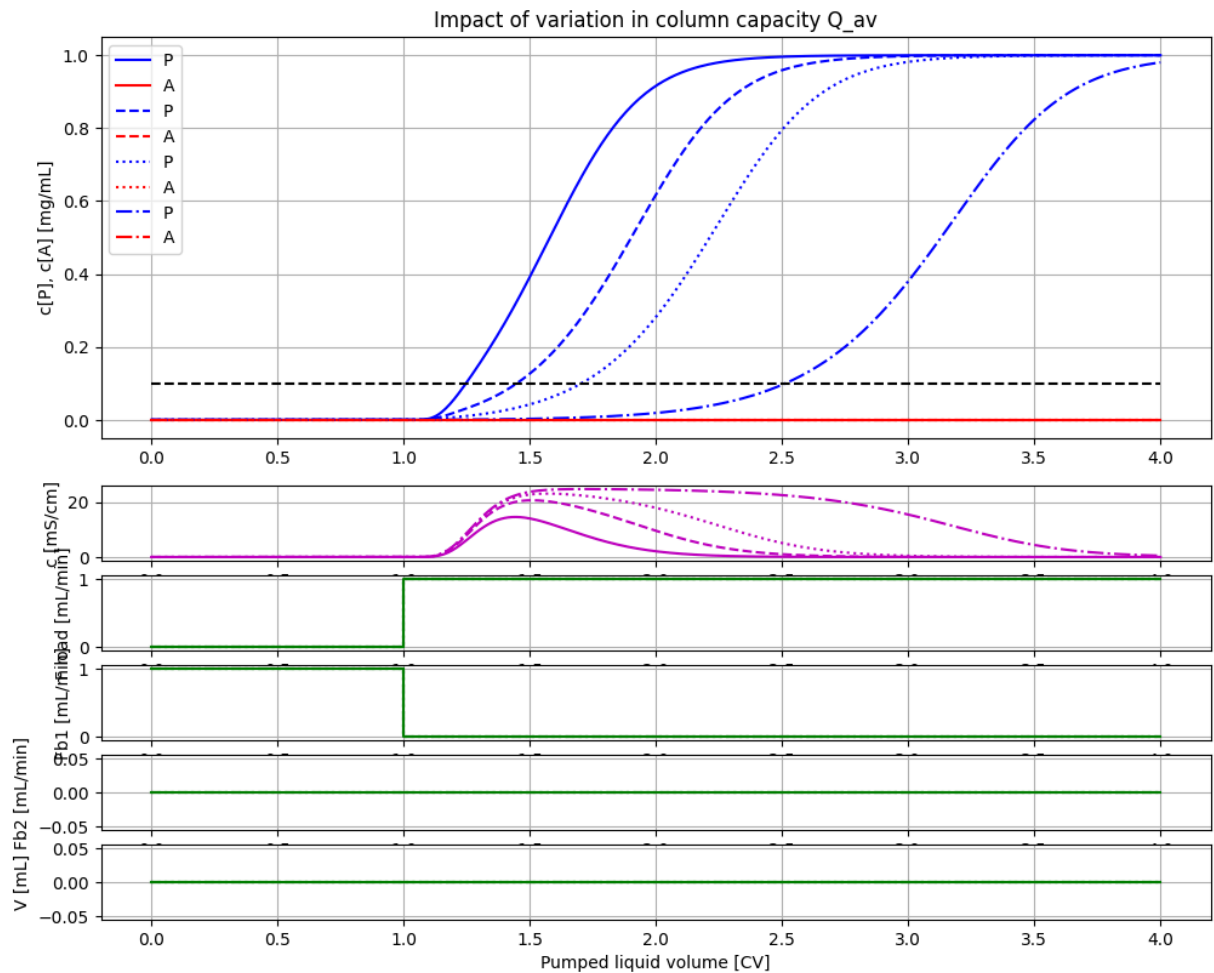
```
In [20]: # Experiment to check column capacity  $Q_{av}$  often called breakthrough curve
par(P_in=1, A_in=0, E_in=0)
init(E_0 = 0)
par( $Q_{av}$ =6.0)

par(scale_volume=True, start_adsorption=1*V, stop_adsorption=4.01*V)
par(start_desorption=10*V, stationary_desorption=10.5*V, stop_desorption=11*V)
par(start_pooling=11*V, stop_pooling=12*V)

newplot(title='Impact of variation in column capacity  $Q_{av}$ ', plotType='Elution-cond
for value in [1, 2, 3, 6]: par( $Q_{av}$ =value); simu(4.0*V/VFR)

# Linje för 10% UV
ax1.plot([0,4], [0.1,0.1], 'k--')

# Restore default parameters
par( $Q_{av}$ =6.0, A_in=1.0)
```



With greater column capacity Q_{av} the longer it takes before the concentration of protein start to increase. Note, that the salt concentration increase initially during adsorption but then go back to low levels. This phenomenon is also seen experimentally.

6 Summary

The simplified simulation model was found useful to describe operational aspects of ion exchange chromatography.

- The model describe qualitatively well the impact of typical operational changes in the flow rate.
- The model also describe qualitatively well the impact of changes in iso-electric point of the proteins relative the pH of the resin.
- The small deviations in salt concentration from linear increase during the gradient in the salt buffer is also what you see in reality.

References

1. Månsson, Jonas, "Control of chromatography column in production scale", Master thesis TFRT-5599, Department of Automatic Control, LTH, Lund Sweden, 1998.
2. Pharmacia LKB Biotechnology. "Ion Exchange chromatography. Principles and Methods.", 3rd edition, 1991.

Appendix

In [21]: `describe('MSL')`

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

In [22]: `system_info()`

System information

- OS: Linux
- Python: 3.11.11
- Scipy: not installed in the notebook
- FMPy: 0.3.22
- FMU by: OpenModelica Compiler OpenModelica 1.21.0
- FMI: 2.0
- Type: ME
- Name: BPL_IEC.Column_system
- Generated: 2023-04-21T10:58:26Z
- MSL: 3.2.3
- Description: Bioprocess Library version 2.1.1
- Interaction: FMU-explore for FMPy version 0.9.8

In [22]: