BPL IEC validation

Author: Jan Peter Axelsson

Here I try to reproduce somre results from Jonas Månssons master thesis report "Control of chromatography comlumn in production scale", TFRT-5599.

The interaction withe the model has changed slightly so that linear flow rate LF = F/area is used. The column volume is now specified in terms of (cross-section) area times height. Thus by changing area the scale up works fine and what you do in practice. Further the time unit changed from seconds to minutes.

The model has 5 states as listed below, 3 states in liquid mobile phase and 2 states for the gel for bound proteins. In a few of Jonas figures also bound ions of the buffer are plotted but I do not do that. Bound ions of the buffer can be calculated from the difference of total binding capaciety Q_{av} and bound protein and bound protein antagonist, see section 5.1 in the report.

The molecular weights listed below are not used in the simulations. They just give typical values and the molecular weight for bound protein and antagonist protein is just arbitrary here.

Parameters of the model in general as well as time scale are all arbitrary and focus is on qualitative aspects of the model.

The height of the column is 20 cm which is a common size in the industry.

In [1]:
```
run -i BPL_IEC_explore.py
```

```
Windows - run FMU pre-compiled JModelica 2.14

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

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

In [2]:
```
plt.rcParams['figure.figsize'] = [30/2.54, 24/2.54]
```

In [3]:
```
describe('chromatography'); print(); describe('liquidphase')
```

```
Ion exchange chromatorgraphy controlled with varying salt-concentration. The pH is
kept constant.

Chromatography liquidphase (or mobilephase) substances included in the model

Protein                 - index =  1 - molecular weight =  22000.0 Da
Antagonist protein      - index =  2 - molecular weight =  15000.0 Da
Ion                     - index =  3 - molecular weight =  58.4 Da
Protein bound           - index =  4 - molecular weight =  100000.0 Da
Antagonist protein bound - index =  5 - molecular weight =  100000.0 Da

Note that both proteins P and A as well as the salt-ion E is modelled to the same
mobile phase volume.
```

Loading

The parameter notation and values are the same as in the referred report. However the flow rate is here denoted F while q in the report. The column is diveded in n=8 sections and set at compilation time. The values are arbitrarly chosen in the report and the focus is on qualitative aspects of the model.

The simplified model describe only the column in terms of volume and does not distinguish a high column with a small diameter from a lower with larger diameter.

The parameters k1, k2, k3, k4 and Q_av are given relative volume and with increased column volume a larger capacity is thus obtained.

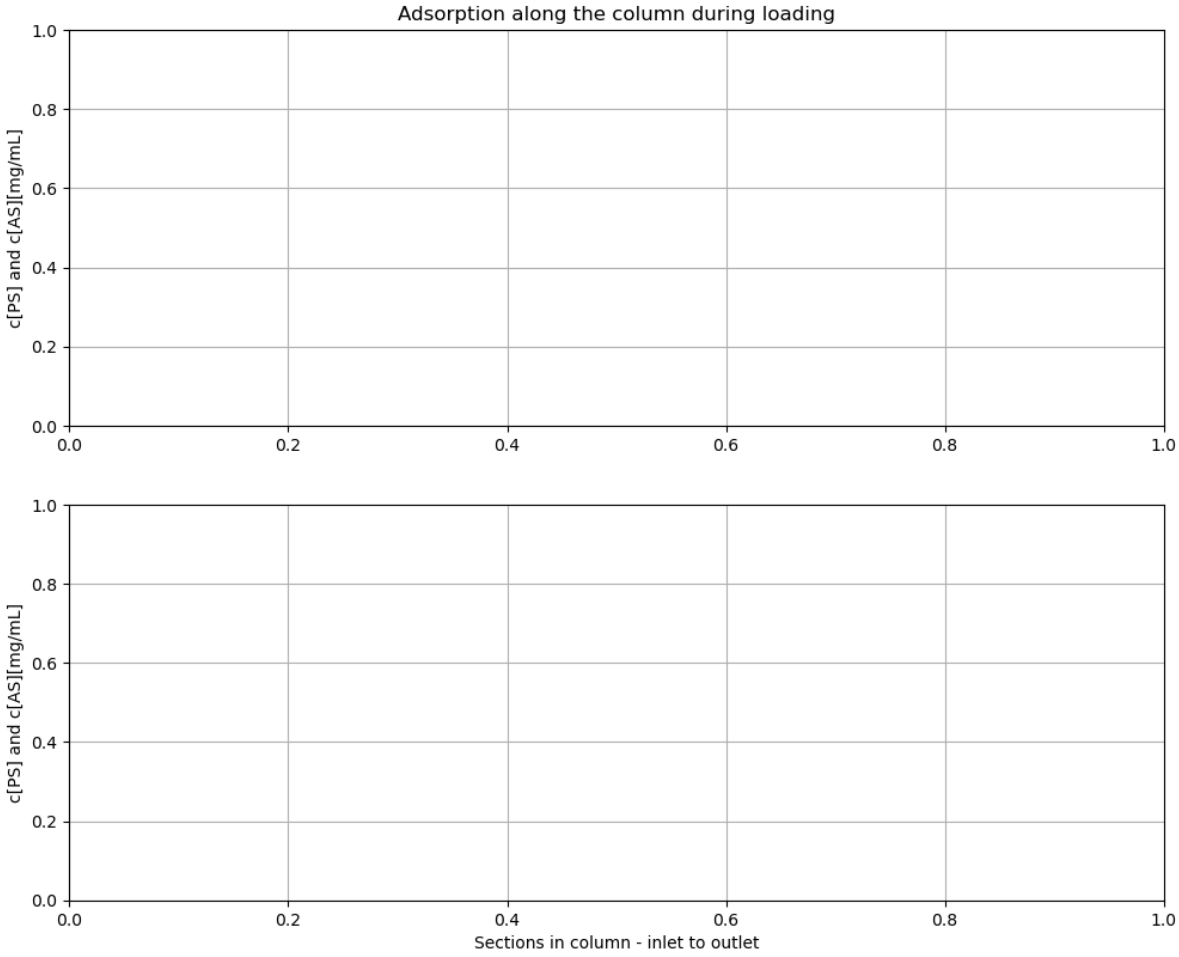
In [4]:
```
describe('column.n')
```

```
Number of sections of the column : 8
```

In [5]:
```
# Loading of the column - try to reproduce Jonas figure 13.
newplot(title='Adsorption along the column during loading', plotType='Loading')

# Column properties
par(LFR=12.0)
par(k1=0.3, k2=0.05, k3=0.05, k4=0.3, Q_av=3.0)
par(height=20, diameter=0.714)

# Operation
par(P_in=1.0, A_in=1.0, E_in=8)
par(scale_volume=False, stop_adsorption=50, start_desorption=150, stop_desorption=4
par(start_pooling=220, stop_pooling=450)

# Simulation
simu(100)
```

```
Error - the following requirements do not hold:
parDict['stationary_desorption'] <= parDict['stop_desorption']
Error - the following requirements do not hold:
parDict['stationary_desorption'] <= parDict['stop_desorption']
```

```
---------------------------------------------------------------------------
FMUException                              Traceback (most recent call last)
Cell In [5], line 15
     12 par(start_pooling=220, stop_pooling=450)
     14 # Simulation
---> 15 simu(100)

File \\VBoxSvr\Modelica\GitHub\Colab\BPL_IEC_validation\BPL_IEC_explore.py:1156, i
n simu(simulationTimeLocal, mode, options, diagrams, timeDiscreteStates)
   1154       model.set(parLocation[key],parDict[key])
   1155   # Simulate
-> 1156   sim_res = model.simulate(final_time=simulationTime, options=options)
   1157   simulationDone = True
   1158 elif mode in ['Continued', 'continued', 'cont']:

File src\pyfmi\fmi.pyx:7453, in pyfmi.fmi.FMUModelCS2.simulate()

File src\pyfmi\fmi.pyx:378, in pyfmi.fmi.ModelBase._exec_simulate_algorithm()

File src\pyfmi\fmi.pyx:372, in pyfmi.fmi.ModelBase._exec_simulate_algorithm()

File ~\miniconda3\envs\pyfmi2100\lib\site-packages\pyfmi\fmi_algorithm_drivers.py:
996, in FMICSAlg.__init__(self, start_time, final_time, input, model, options)
    994 elif isinstance(self.model, fmi.FMUModelCS2):
    995     self.model.setup_experiment(start_time=start_time, stop_time_defined=s
elf.options["stop_time_defined"], stop_time=final_time)
--> 996     self.model.initialize()
    998 else:
    999     raise fmi.FMUException("Unknown model.")

File src\pyfmi\fmi.pyx:4746, in pyfmi.fmi.FMUModelBase2.initialize()

File src\pyfmi\fmi.pyx:4740, in pyfmi.fmi.FMUModelBase2.initialize()

File src\pyfmi\fmi.pyx:4692, in pyfmi.fmi.FMUModelBase2.enter_initialization_mode
()

FMUException: Enter Initialize returned with an error. Enable logging for more inf
ormation, (load_fmu(..., log_level=4)).
```



In [7]:
```
parDict
```

Out[7]:
```
{'diameter': 0.714,
 'height': 20,
 'x_m': 0.3,
 'k1': 0.3,
 'k2': 0.05,
 'k3': 0.05,
 'k4': 0.3,
 'Q_av': 3.0,
 'E_0': 0.0,
 'P_in': 1.0,
 'A_in': 1.0,
 'E_in': 8,
 'E_in_desorption_buffer': 0.3,
 'LFR': 12.0,
 'scale_volume': False,
 'gradient': True,
 'start_adsorption': 0,
 'stop_adsorption': 50,
 'start_desorption': 150,
 'x_start_desorption': 0.2,
 'stationary_desorption': 500,
 'stop_desorption': 450,
 'start_pooling': 220,
 'stop_pooling': 450,
 'start_uv': -1,
 'stop_uv': -2}
```

In [ ]:
```
# We just check that we had the same volume flow rate as Jonas
describe('F')
```

Looks like Figure 13 in Jonas report!

In [ ]:
```
# Impact of shorter time for loading and then less material
newplot(title='Loading', plotType='Loading-combined')
show()

# Simulation with changed parameter t2
par(stop_adsorption=30); simu(100)

# Reset changed parameter
par(stop_adsorption=50)
```

To the left the inlet loading over time. To the right upper concentration along the column at different times and in steady states finally To the right lower concentrations along the column in steady state.

We see that a shorter time and then less material makes less of the column capacity used.

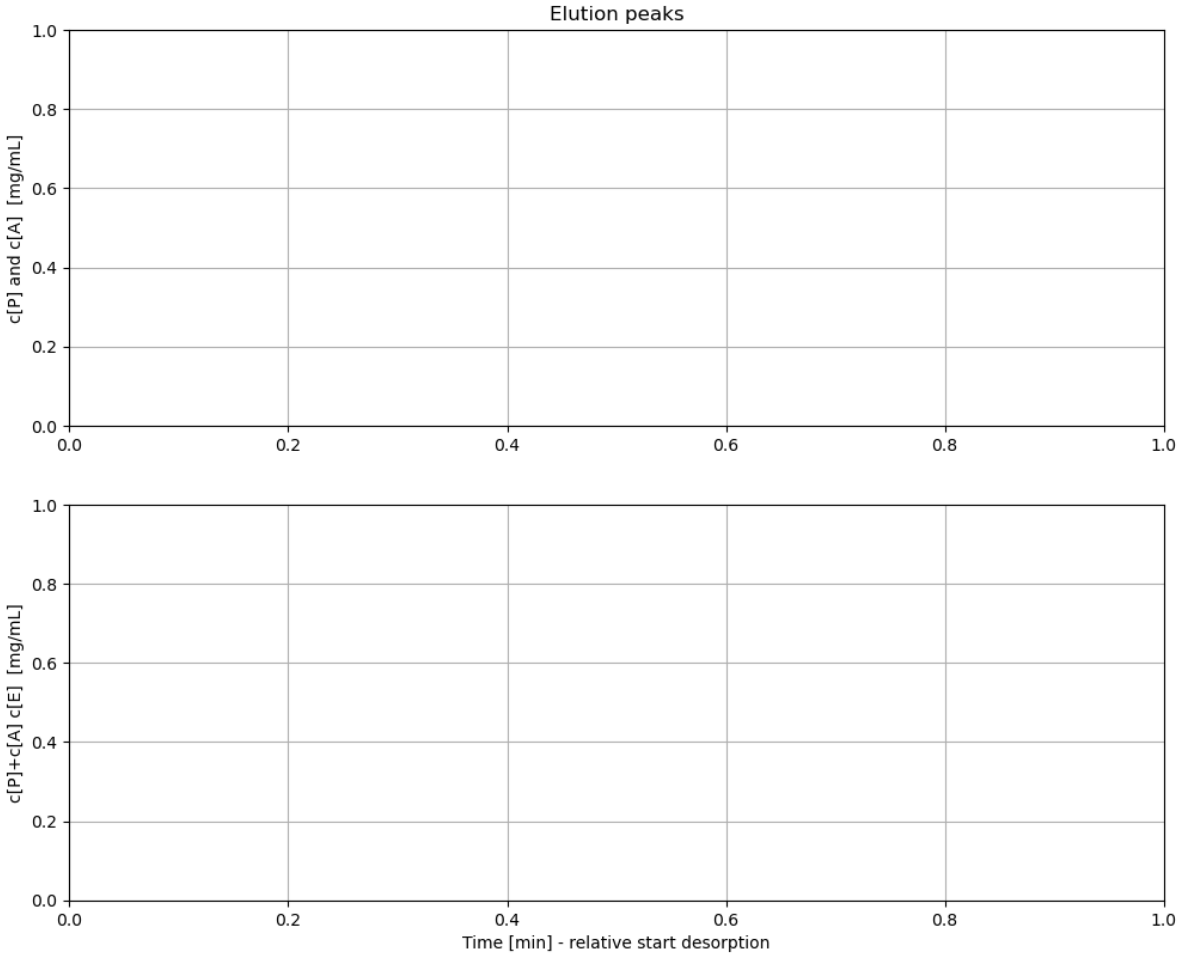
Note that the flow through the column is constant despite change from sample to just buffer 1, and shown in how the volume of the waste tank increase with time.

Elution

In [11]:
```
# Elution of the column
newplot(title='Elution peaks', plotType='Elution')

# Operation
par(P_in=1, A_in=1.0, E_in=8)
par(LFR=12.0, stop_adsorption=50, start_desorption=150, stop_desorption=450)

# Simulation
simu(550)
```

```
Error - the following requirements do not hold:
parDict['stationary_desorption'] <= parDict['stop_desorption']
Error - the following requirements do not hold:
parDict['stationary_desorption'] <= parDict['stop_desorption']
```

```
---------------------------------------------------------------------------
FMUException                              Traceback (most recent call last)
Cell In [11], line 9
      6 par(LFR=12.0, stop_adsorption=50, start_desorption=150, stop_desorption=45
0)
      8 # Simulation
----> 9 simu(550)

File \\VBoxSvr\Modelica\GitHub\Colab\BPL_IEC_validation\BPL_IEC_explore.py:1156, i
n simu(simulationTimeLocal, mode, options, diagrams, timeDiscreteStates)
   1154       model.set(parLocation[key],parDict[key])
   1155    # Simulate
-> 1156    sim_res = model.simulate(final_time=simulationTime, options=options)
   1157    simulationDone = True
   1158 elif mode in ['Continued', 'continued', 'cont']:

File src\pyfmi\fmi.pyx:7453, in pyfmi.fmi.FMUModelCS2.simulate()

File src\pyfmi\fmi.pyx:378, in pyfmi.fmi.ModelBase._exec_simulate_algorithm()

File src\pyfmi\fmi.pyx:372, in pyfmi.fmi.ModelBase._exec_simulate_algorithm()

File ~\miniconda3\envs\pyfmi2100\lib\site-packages\pyfmi\fmi_algorithm_drivers.py:
996, in FMICSAlg.__init__(self, start_time, final_time, input, model, options)
    994 elif isinstance(self.model, fmi.FMUModelCS2):
    995     self.model.setup_experiment(start_time=start_time, stop_time_defined=s
elf.options["stop_time_defined"], stop_time=final_time)
--> 996     self.model.initialize()
    998 else:
    999     raise fmi.FMUException("Unknown model.")

File src\pyfmi\fmi.pyx:4746, in pyfmi.fmi.FMUModelBase2.initialize()

File src\pyfmi\fmi.pyx:4740, in pyfmi.fmi.FMUModelBase2.initialize()

File src\pyfmi\fmi.pyx:4692, in pyfmi.fmi.FMUModelBase2.enter_initialization_mode
()

FMUException: Enter Initialize returned with an error. Enable logging for more inf
ormation, (load_fmu(..., log_level=4)).
```



Very close to Jonas Figure 14!

The upper diagrams shows the column outlet concentrations of P and A over time.

The lower diagram shows the sum (or possibly the UV signal) at column outlet as well as the salt concentration. We have some separation between the two peaks.

Note that the salt concentration deviates slightly from the linear increse between time 50 to 100. This is due to ion interaction with P and A in the column. The ion-salt concentration is scaled with factor 0.05 to get comparable concentrations to P and A.

I have here simulated time 150 of adsorbtion and then started elution. Here is time counted as zero at time of start of elutions. Not sure how long Jonas simulated to get steady state before he did elution.

In [ ]:
```
# More complete visualization of the elution phase and the different flows
newplot(title='Elution and corresponding flows', plotType='Elution-combined')
show()

# Simulation with changed parameter t2
par(stop_adsorption=30); simu(550)

# Reset changed parameter
par(stop_adsorption=50)
```

Here a diagram that shows the peaks at the outlet as shown in the previous diagram. Below the flow rates of the three differnet sorces. Here time is 0 at start of adsorbtion and elution starts at time 150.

All material goes at the end to the waste tank. Automatic pooling to a product tank is not implemented yet.

The comparison with a smaller loading and no effect on the P and A peaks confuse me.

Acknowledgment

The author thank Karl Johan Brink for sharing his know-how of chromatography operation. He has especially given input of how to parametrize the model in terms often used in practice.

References

1. Månsson, Jonas, "Control of chromatography comlumn in production scale", Master thesis TFRT-5599, Department of Automatic Control, LTH, Lund Sweden, 1998.

2. Pharmacia LKB Biotechnology. "Ion Exchange chromatography. Principles and Mathods.", 3rd edition, 1991.

Appendix

In [9]:
```
describe('MSL')
```

```
MSL: RealInput, RealOutput, Constants, Hysteresis, CombiTimeTable, Types
```

In [10]:
```
system_info()
```

```
System information
 -OS: Windows
 -Python: 3.10.6
 -Scipy: not installed in the notebook
 -PyFMI: 2.10.0
 -FMU by: JModelica.org
 -FMI: 2.0
 -Type: FMUModelCS2
 -Name: BPL_IEC.Column_system
 -Generated: 2023-04-21T12:28:38
 -MSL: 3.2.2 build 3
 -Description: Bioprocess Library version 2.1.1
 -Interaction: FMU-explore version 0.9.7
```

In [ ]:
```

```