Cyclic Flow Modulation

Peter Gawthrop (peter.gawthrop@unimelb.edu au)

April 16, 2020

Contents

1 Introduction **2**
1.1 Import some python code . . . 3

2 Modulated Cooperative Enzyme-catalysed Reaction (mECR) **3**
2.1 Two-stage cooperative enzyme-catalysed reaction (N=2) with modulation . . . 3

3 Cyclic Flow Modulation (CFM) **5**
3.1 Stoichiometry and reactions . . . 6

4 Simulation of Steady-state properties **7**
4.1 Set up some parameters for simulation . . . 7
4.2 Simulation code . . . 8
4.3 Vary the substrate concentration. . . . 10
4.4 Vary the activation species concentration. . . . 12
4.5 Vary the substrate concentration. . . . 13
4.6 Vary the inhibition species concentration. . . . 15
4.7 Discussion . . . 16

5 Fructose-2,6-phosphate ($F_{26}P$) **16**
5.1 Fructose-2,6-phosphate ($F_{26}P$) CFM as an integrator . . . 17
5.2 Simulation . . . 17
5.3 Discussion . . . 19

Note: this is the CyclicFlowModulation.ipynb notebook. The PDF version "Cyclic Flow Modulation" is available here.

1 Introduction

The reaction F_6P + ATP $\overset{PFK}{\rightleftharpoons}$ $F_{16}P$ + ADP catalysed by the enzyme PFK is a key step in glycolysis where:

- PFK phosphofructokinase
- F_6P fructose-6-phosphate
- $F_{16}P$ fructose-1,6-biphosphate

As pointed out by (Cornish-Bowden, 2013), section 12.1.1., the PFK-catalysed reaction forms a cycle with the reaction: $F_{16}P + H_2O \overset{FBP}{\rightleftharpoons} F_6P$ + Pi where:

- FBP fructose biphosphatase
- Pi inorganic phosphate

This cycle is *modulated* by a number of species which simultaneously activate the PFK reaction and inhibit the FBP reaction or *vice-versa*.
(Cornish-Bowden, 2013) [section 12.1.1], (Garrett and Grisham, 2017) [sections 18.3c,22.1 (3), 22.2a]. Indeed (Garrett and Grisham, 2017) [section 22.2b] explicitly states that "substrate cycles provide metabolic control mechanisms".
The species which activate PFK and inhibit FBP include:

- AMP
- $F_{26}P$ fructose-2,6-phosphate

The species which inhibit PFK and activate FBP include:

- ATP
- Cit citrate

Because of the cyclic nature of these two reactions, and the fact that flow is modulated, the term **Cyclic Flow Modulation** (CFM) is used to decribe such reaction systems.

- This note gives a bond graph (Gawthrop and Crampin, 2014) interpretation of such Cyclic Flow Modulation and uses BondGraphTools (Cudmore et al., 2019) to build an analyse a simple example of Cyclic Flow Modulation.
- The note also provides an example of graphical computational modularity where graphical represention in SVG format are converted using svgBondGraph -- see Tutorial svgBondGraph
- Cooperativity is discussed in the notebook Cooperativity and modulated cooperativity is discussed in the notebook modulated Cooperativity.

1.1 Import some python code

The bond graph analysis uses a number of Python modules:

In [1]:
```
## Some useful imports

import BondGraphTools as bgt
import numpy as np
import sympy as sp
import matplotlib.pyplot as plt
import IPython.display as disp

## Stoichiometric analysis
import stoich as st

## SVG bg representation conversion
import svgBondGraph as sbg

## Modular bond graphs
import modularBondGraph as mbg

## Data structure copy
import copy

## For reimporting: use imp.reload(module)
import importlib as imp

## Set quiet=False for verbose output
quiet = True
```

2 Modulated Cooperative Enzyme-catalysed Reaction (mECR)

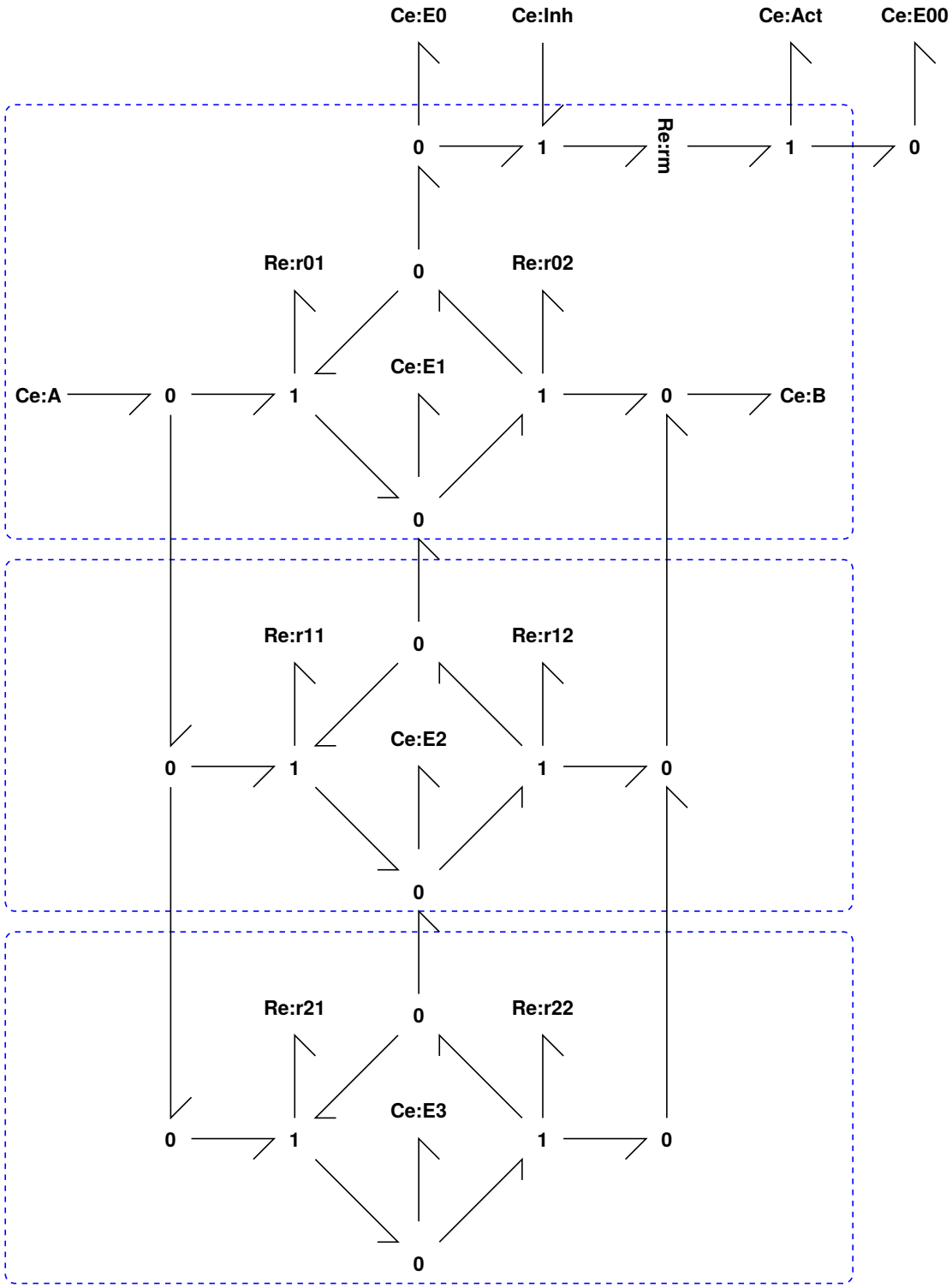
As discussed in the notebook notebook modulatedCooperativity a modulated Cooperative Enzyme-catalysed Reaction may be modelled using the following bond graph. Two instances of this model are used in the sequel to model Cyclic Flow Modulation.

2.1 Two-stage cooperative enzyme-catalysed reaction (N=2) with modulation

The cooperative enzyme-catalysed reaction is modulated by the activation species (Act) and the inhibition species (Inh).

In [2]:
```
## Two-stage cooperative enzyme-catalysed reaction (N=2)
sbg.model('mCoop_abg.svg',quiet=quiet)
import mCoop_abg
disp.SVG('mCoop_abg.svg')
```

Out[2]:




```
In [3]: s = st.stoich(mCoop_abg.model(),quiet=quiet)
        sc = st.statify(s,chemostats=['A','B','Act','Inh'])
        disp.Latex(st.sprintrl(s,chemformula=True))
```

Out[3]:

$$A + E_0 \rightleftharpoons^{r_{01}} E_1 \tag{1}$$

$$E_1 \rightleftharpoons^{r_{02}} B + E_0 \tag{2}$$

$$A + E_1 \rightleftharpoons^{r_{11}} E_2 \tag{3}$$

$$E_2 \rightleftharpoons^{r_{12}} B + E_1 \tag{4}$$

$$A + E_2 \rightleftharpoons^{r_{21}} E_3 \tag{5}$$

$$E_3 \rightleftharpoons^{r_{22}} B + E_2 \tag{6}$$

$$E_0 + Inh \rightleftharpoons^{rm} Act + E_{00} \tag{7}$$

3 Cyclic Flow Modulation (CFM)

As discussed in the context of the PFK/FBP cycle in the introduction, CFM involves a cycle formed of two modulated enzyme-catalysed reactions. Such a cycle is shown in the following bond graph with the following components and interpretation:

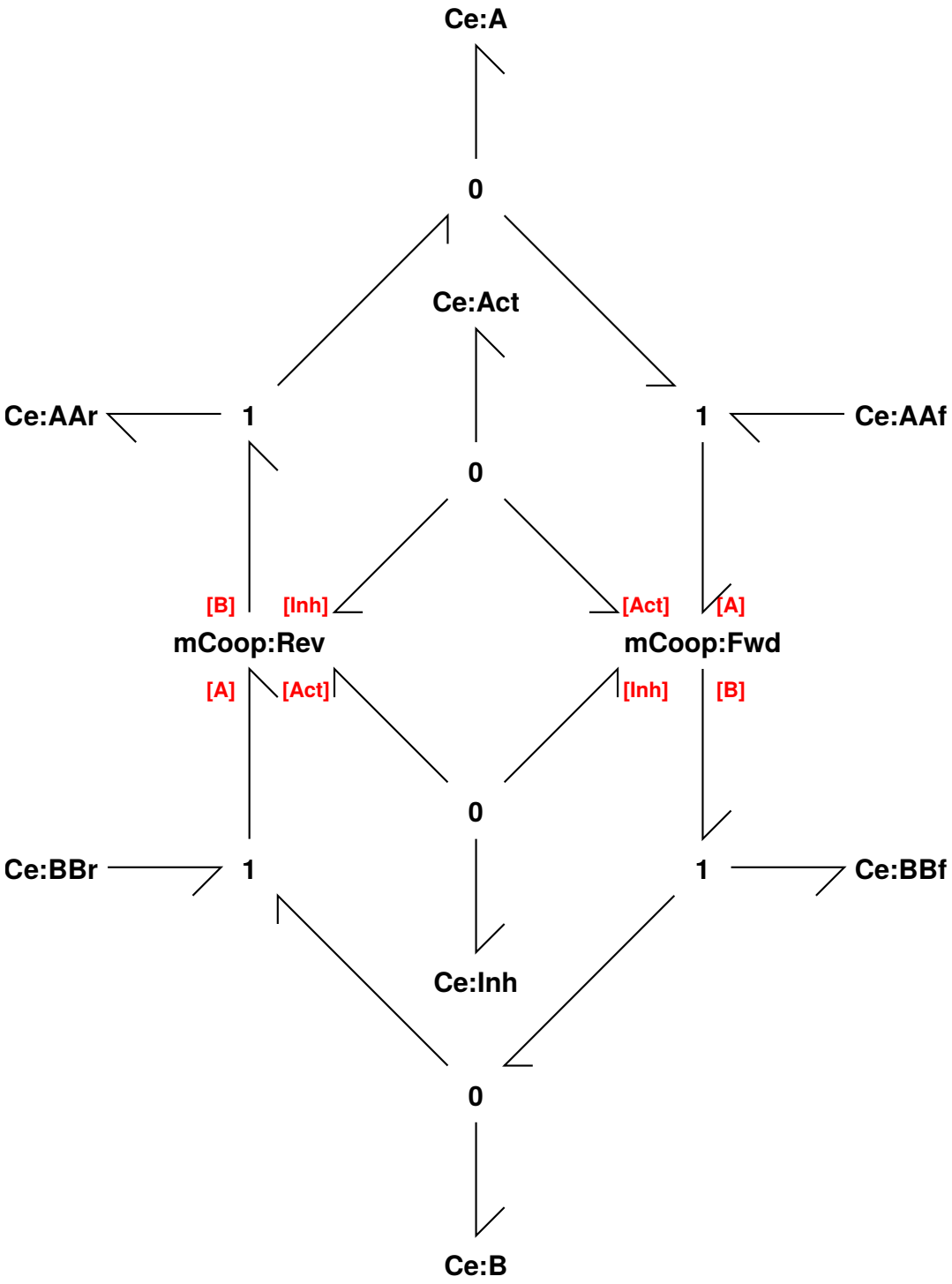
- mCoop:Fwd an instance of the mECR representing the forward reaction [PFK]
- mCoop:Rev an instance of the mECR representing the reverse reaction [FBP]
- Ce:A The substrate species [F_6P]
- Ce:B The product species [$F_{16}P$]
- Ce:Act The activation species [AMP + $F_{26}P$]
- Ce:Inh The inhibition species [ATP + Cit]
- Ce:AAf,Ce:BBf,Ce:AAr,Ce:BBr Additional species [ATP,ADP,Pi,H_2O]

Note that the activator Ce:Act activates mCoop:Fwd and inhibits mCoop:Rev and Ce:Inh inhibits mCoop:Fwd and activates mCoop:Rev.

In [4]:
```
sbg.model('CFM_abg.svg',quiet=quiet)
import CFM_abg
imp.reload(CFM_abg)
disp.SVG('CFM_abg.svg')
```

```
Creating subsystem: mCoop:Fwd
Creating subsystem: mCoop:Rev
```

Out[4]:



3.1 Stoichiometry and reactions

```
In [5]: s = st.stoich(CFM_abg.model(),quiet=quiet)
        sc = st.statify(s,chemostats=['A','B','Act','Inh','AAf','BBf','AAr','BBr'])
```

```
disp.Latex(st.sprintrl(s,chemformula=True))
```

Out[5]:

$$\mathrm{A} + \mathrm{AAf} + \mathrm{Fwd}_{\mathrm{E0}} \rightleftharpoons \mathrm{Fwd}_{\mathrm{E1}} \quad (8)$$

$$\mathrm{Fwd}_{\mathrm{E1}} \rightleftharpoons \mathrm{B} + \mathrm{BBf} + \mathrm{Fwd}_{\mathrm{E0}} \quad (9)$$

$$\mathrm{A} + \mathrm{AAf} + \mathrm{Fwd}_{\mathrm{E1}} \rightleftharpoons \mathrm{Fwd}_{\mathrm{E2}} \quad (10)$$

$$\mathrm{Fwd}_{\mathrm{E2}} \rightleftharpoons \mathrm{B} + \mathrm{BBf} + \mathrm{Fwd}_{\mathrm{E1}} \quad (11)$$

$$\mathrm{A} + \mathrm{AAf} + \mathrm{Fwd}_{\mathrm{E2}} \rightleftharpoons \mathrm{Fwd}_{\mathrm{E3}} \quad (12)$$

$$\mathrm{Fwd}_{\mathrm{E3}} \rightleftharpoons \mathrm{B} + \mathrm{BBf} + \mathrm{Fwd}_{\mathrm{E2}} \quad (13)$$

$$\mathrm{Inh} + \mathrm{Fwd}_{\mathrm{E0}} \rightleftharpoons \mathrm{Act} + \mathrm{Fwd}_{\mathrm{E00}} \quad (14)$$

$$\mathrm{B} + \mathrm{BBr} + \mathrm{Rev}_{\mathrm{E0}} \rightleftharpoons \mathrm{Rev}_{\mathrm{E1}} \quad (15)$$

$$\mathrm{Rev}_{\mathrm{E1}} \rightleftharpoons \mathrm{A} + \mathrm{AAr} + \mathrm{Rev}_{\mathrm{E0}} \quad (16)$$

$$\mathrm{B} + \mathrm{BBr} + \mathrm{Rev}_{\mathrm{E1}} \rightleftharpoons \mathrm{Rev}_{\mathrm{E2}} \quad (17)$$

$$\mathrm{Rev}_{\mathrm{E2}} \rightleftharpoons \mathrm{A} + \mathrm{AAr} + \mathrm{Rev}_{\mathrm{E1}} \quad (18)$$

$$\mathrm{B} + \mathrm{BBr} + \mathrm{Rev}_{\mathrm{E2}} \rightleftharpoons \mathrm{Rev}_{\mathrm{E3}} \quad (19)$$

$$\mathrm{Rev}_{\mathrm{E3}} \rightleftharpoons \mathrm{A} + \mathrm{AAr} + \mathrm{Rev}_{\mathrm{E2}} \quad (20)$$

$$\mathrm{Act} + \mathrm{Rev}_{\mathrm{E0}} \rightleftharpoons \mathrm{Inh} + \mathrm{Rev}_{\mathrm{E00}} \quad (21)$$

4 Simulation of Steady-state properties

The steady state properties are investigated using dynamic simulation where slowly varing exogenous quantities are used to induce quasi-steady-state behaviour. In each case, the variable is at a constant value to start with followed by a slowly increasing ramp. The response after the initial reponse is plotted to remove artefacts due to the initial transient.

All parameters are unity except for $K_B = 10^{-6}$ (to approximate an irreversible reaction) and initial states are chosen so that the total enzyme is $e_0 = 1$.

4.1 Set up some parameters for simulation

```
In [6]: ## Set up some parameters for simulation
        def setParameter(s,N,e0,K_0=1,Integrator=False):
            ## Set up the non-unit parameters and states

            parameter = {}
```

```
    ## Cycle driving potentials
    K_BB = 1e-6
    parameter['K_BBf'] = K_BB
    parameter['K_AAr'] = K_BB

    #parameter['K_BBr'] = 0.1

    ## Reaction constants for forward and reverse reactions
    kappa = {}
    kappa['Fwd'] = 2

    if Integrator:
        kappa['Rev'] = 2
        parameter['K_A']= 1
        parameter['K_B']= 0.01
    else:
        kappa['Rev'] = 2
        parameter['K_A']= 1
        parameter['K_B']= 1

    X0 = np.ones(s['n_X'])
    for fr in ['Fwd','Rev']:
        K_i = 1/(K_0**(N+1))
        X0[s['spec_index'][fr+'_E00']] = (e0/(N+3))
        for i in range(N+2):
            Ei = '_E'+str(i)
            X0[s['spec_index'][fr+Ei]] = (e0/(N+3))
            Ki = 'K_'+fr+Ei
            parameter[Ki] = K_i
            K_i *= K_0
            if i<N+1:
                kappa_i = 'kappa_'+fr+'_r'+str(i)
                parameter[kappa_i+'1'] = kappa[fr]
                parameter[kappa_i+'2'] = kappa[fr]

    return parameter,X0
```

4.2 Simulation code

The flow v is a dynamical function of substrate x_A, activation x_{Act}, inhibition x_{Inh} and cooperativity index N. An approximate steady-state is acjieved by varying one of the three concentrations slowly whilst fixing the other two. The following function does this by declaring the varying function species by the string sX, a fixed species with a number of discrete values as sX1 with values XX1 and the other species as sX2 with value X2. N can take on a range of values.

deriv=True gives a plot of the derivative of the flow with respect to $\log_{10} X$.

```
In [7]: def label(sX1,sX2,X1,X2,N,Loop=False):
            if N<0:
```

```
            return f'{sX1}={X1}, N={-N} (graphical)'
        else:
            if Loop:
                return f'{sX1}={X1}(Loop flow)'
            else:
                return f'{sX1}={X1}'

def VaryX(sX='A',sX1='Act',sX2='Inh',Xrange=[1e-2,1e2],XX1=[1],X2=1,NN=[2],K_B=1e-6,
          IntPar=False,deriv=False,quiet=True):

    ## Time
    t_max = int(1e4)
    t = np.linspace(0,t_max,100000)
    t_0 = 100
    t_1 = t_max-t_0
    i_max = len(t)
    i_0 = int(i_max*t_0/t_max)
    i_1 = i_max-i_0
    #print(i_0,i_1)

    ## Set up the chemostats: vary X
    x_max = Xrange[1]
    x_min = Xrange[0]
    chemo = '{3} + ({0}-{3})*np.heaviside(t-{1},1)*((t-{1})/{2})'.format(x_max,t_0,t_1,x
    X_chemo =  {sX:chemo}

    for N in NN:
        for X1 in XX1:

            ## Non-unit parameters and states
            e0 = 1 # Total enzyme
            parameter,X0 = setParameter(s,abs(N),e0,K_0=1,Integrator=IntPar)
            X0[s['spec_index'][sX1]] = X1
            X0[s['spec_index'][sX2]] = X2
            #print(X0)
            dat = st.sim(s,sc=sc,t=t,parameter=parameter,X0=X0,X_chemo=X_chemo,quiet=qui

            ## Extract flows at the chemostatted species
            VV = dat['V']
            dX  = s['N']@(VV.T)
            dX_B = dX[s['spec_index']['B'],:]
            dX_BBf = dX[s['spec_index']['BBf'],:]
            dX_AAr = dX[s['spec_index']['AAr'],:]
            V = dX_B
            V_C_1 = dX_BBf-V
            V_C = dX_AAr

            ## Extract the state being varied
```

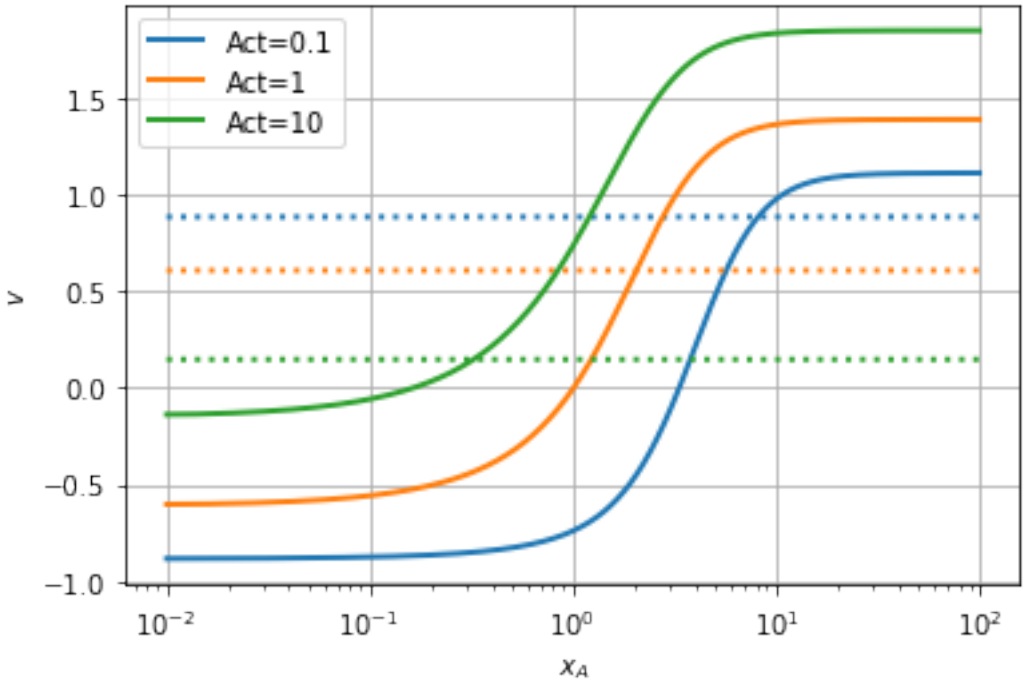
```
            X = dat['X'][:,s['spec_index'][sX]]


            lw = 2
            ls = None
            if deriv:
                slope = np.gradient(V[-i_1:],np.log10(X[-i_1:]))
                plt.semilogx(X[-i_1:],slope,lw=lw,label=label(sX1,sX2,X1,X2,N),linestyle
                ylabel = '$dv/d \log_{10}{x}$'

            else:
                p = plt.semilogx(X[-i_1:],V[-i_1:],lw=lw,label=label(sX1,sX2,X1,X2,N),li
                colour = p[0].get_color()
                plt.semilogx(X[-i_1:],V_C[-i_1:],color=colour,lw=lw,linestyle='dotted')
                ylabel = '$v$'



        plt.xlabel('$x_{'+sX+'}$')
        plt.ylabel(ylabel)
        plt.legend()
        plt.grid()
        #plt.title('N = '+str(N))
        plt.show()

        return dat,X
```

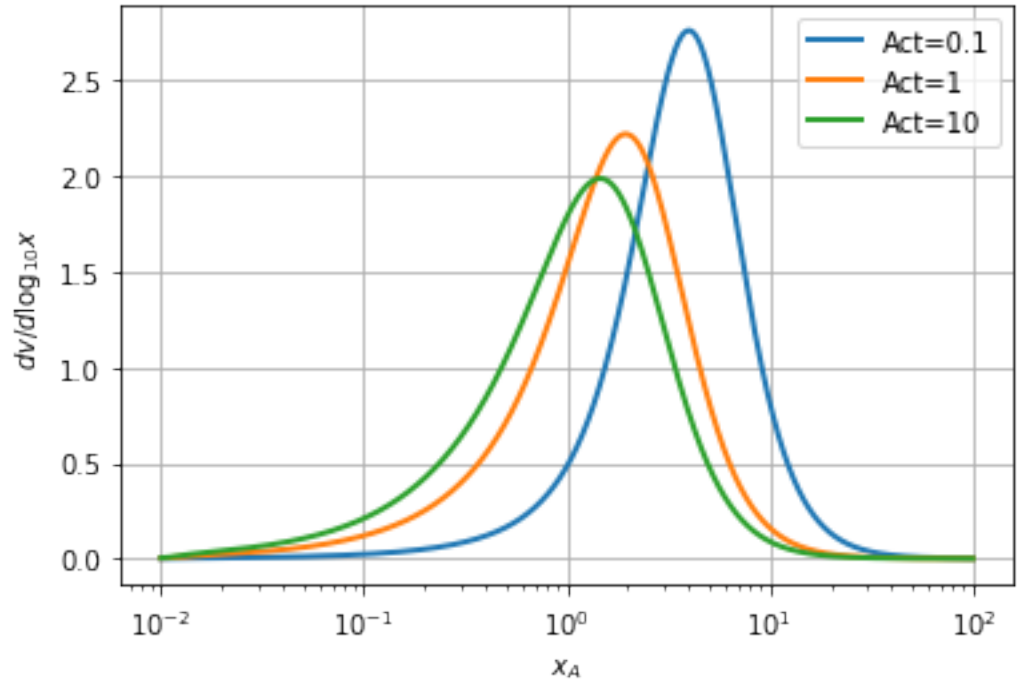
4.3 Vary the substrate concentration.

The substrate concentration x_A is varied for two values of activation x_{Act}.

- dotted lines give the cyclic flow.
- The derivative is also plotted.

```
In [8]: Act = [0.1,1,10]
        dat,x = VaryX(sX='A',sX1='Act',sX2='Inh',XX1=Act,X2=1)
        dat,x = VaryX(sX='A',sX1='Act',sX2='Inh',XX1=Act,X2=1,deriv=True)
```


Act=0.1
Act=1
Act=10
1.5
1.0
0.5
0.0
−0.5
−1.0
v
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
x_A

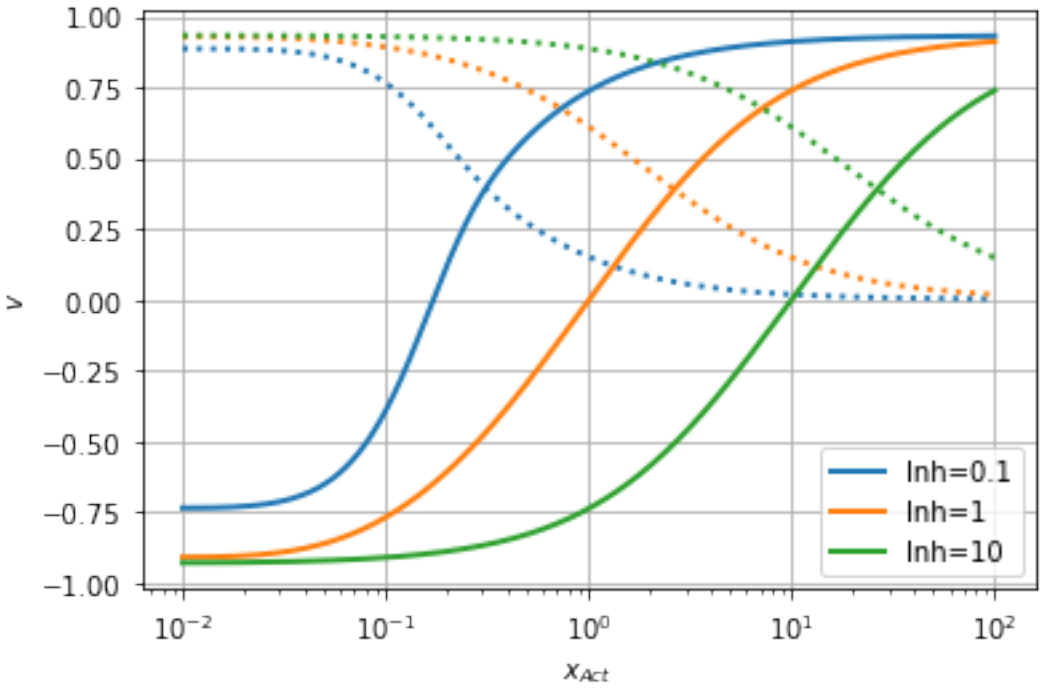

Act=0.1
Act=1
Act=10
2.5
2.0
1.5
1.0
0.5
0.0
$dv/d\log_{10}x$
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
x_A

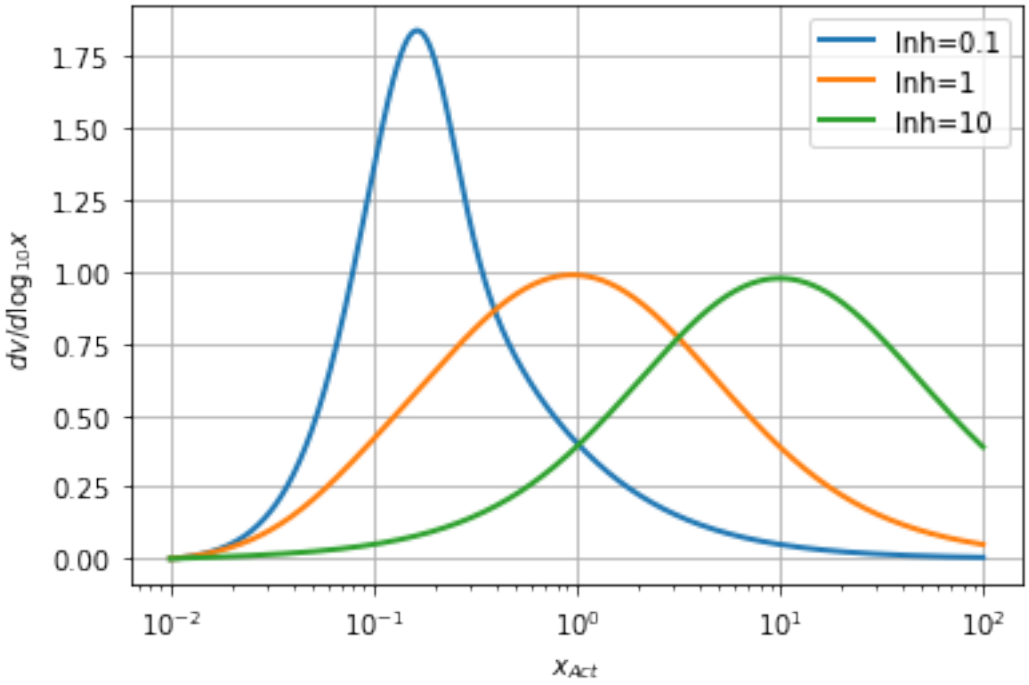
4.4 Vary the activation species concentration.

The activation species concentration x_{Act} is varied for three values of x_{Inh}.

- dotted lines give the cyclic flow.
- The derivative is also plotted.

```
In [9]: Inh = [0.1,1,10]
        dat,x = VaryX(sX='Act',sX1='Inh',sX2='A',XX1=Inh,X2=1)
        dat,x = VaryX(sX='Act',sX1='Inh',sX2='A',XX1=Inh,X2=1,deriv=True)
```



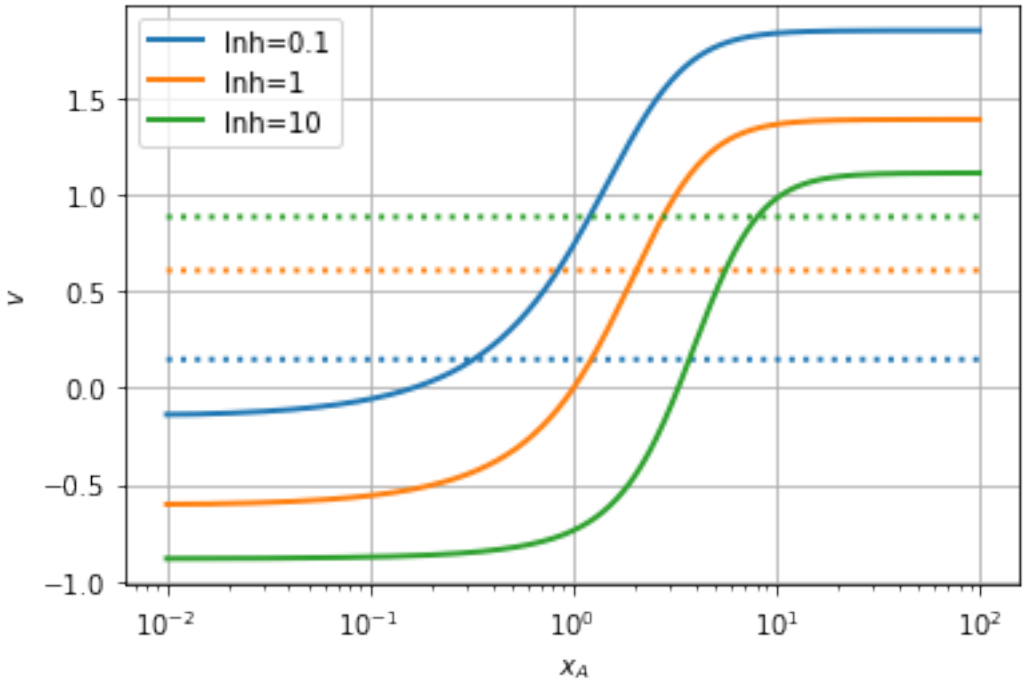


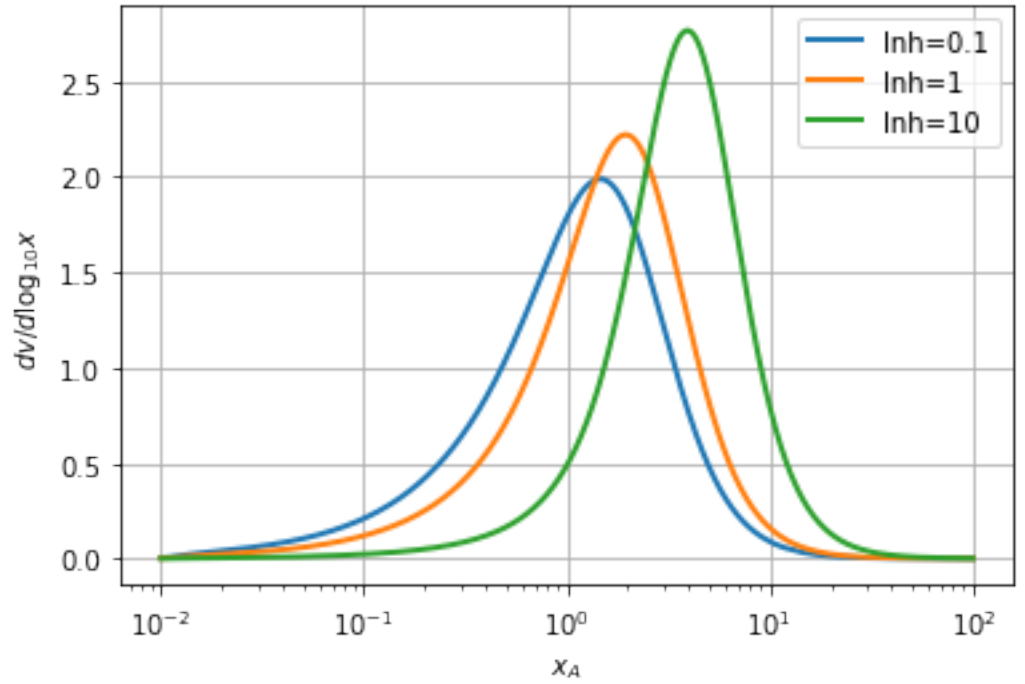
4.5 Vary the substrate concentration.

The substrate concentration x_A is varied for three values of inhibition x_{Inh}.

- dotted lines give the cyclic flow.
- The derivative is also plotted.

```
In [10]: dat,x = VaryX(sX='A',sX1='Inh',sX2='Act',XX1=Inh,X2=1)
         dat,x = VaryX(sX='A',sX1='Inh',sX2='Act',XX1=Inh,X2=1,deriv=True)
```


Inh=0.1
Inh=1
Inh=10
1.5
1.0
0.5
0.0
−0.5
−1.0
v
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
x_A

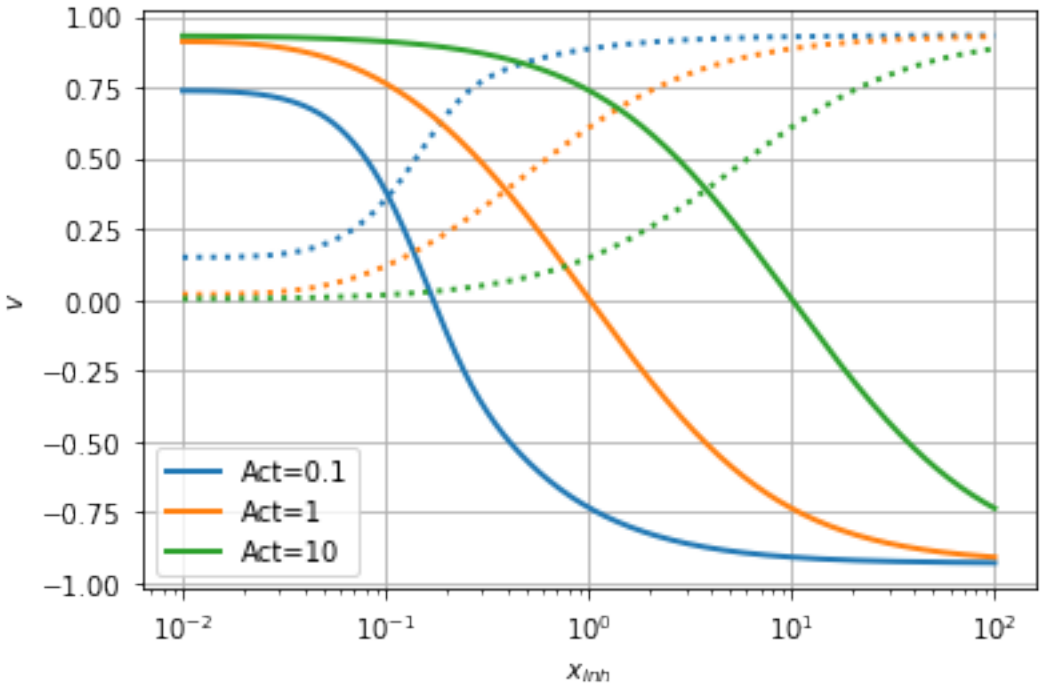

Inh=0.1
Inh=1
Inh=10
2.5
2.0
1.5
1.0
0.5
0.0
$dv/d\log_{10}x$
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
x_A

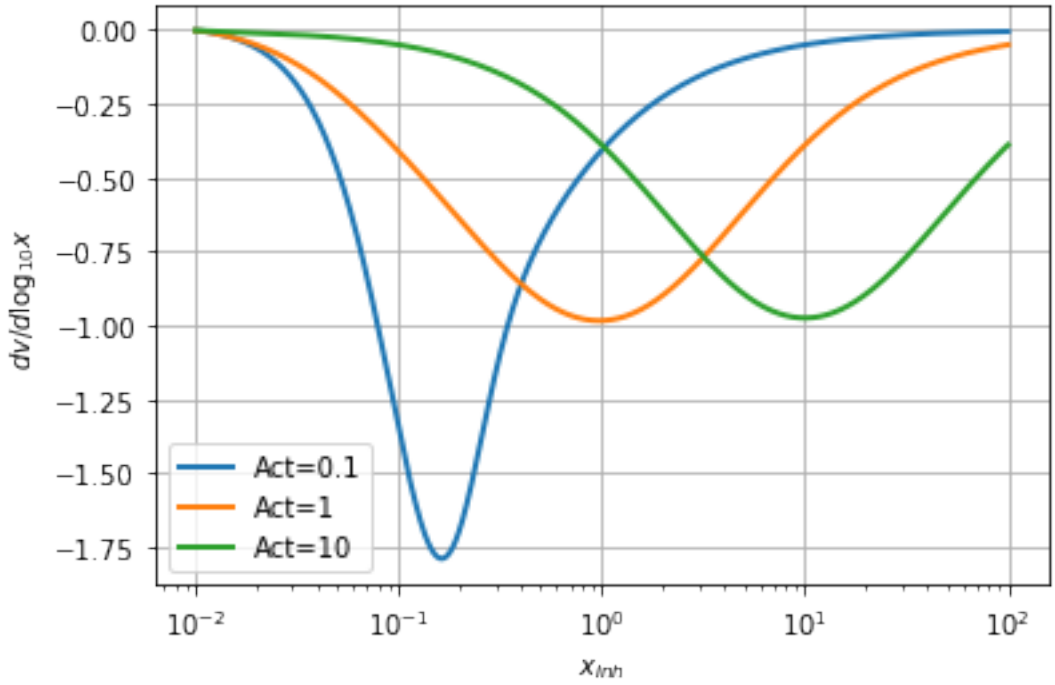
4.6 Vary the inhibition species concentration.

The inhibition species concentration x_{Inh} is varied for two values of N.

- dotted lines give the cyclic flow.
- The derivative is also plotted.

```
In [11]: dat,x = VaryX(sX='Inh',sX1='Act',sX2='A',XX1=Act,X2=1)
         dat,x = VaryX(sX='Inh',sX1='Act',sX2='A',XX1=Act,X2=1,deriv=True)
```





4.7 Discussion

- Both positive and negative flow rates are possible
- The substrate concentration affects net flow but not loop flow.
- Loop flow is affected by the degree of activation/inhibition as well as the driving species AAf, BBf, AAr and BBr.
- Note that species A potential $\phi_A = \ln K_A X_A$. Thus plotting against $\log_{10} X$ is equivalent to plotting against potential with a constant factor.
- Increasing activation increases flow - this corresponds to positive feedback with positive incremental gain given by the derivative plots.
- Increasing inhibition decreases flow - this corresponds to negative feedback with negative incremental gain given by the derivative plots.
- the behaviour is dependent on the parameters of the particular enzyme-catalysed reaction; those used here are for illustration.

5 Fructose-2,6-phosphate ($F_{26}P$)

The reaction F_6P + ATP $\overset{\text{PFK2}}{\rightleftharpoons}$ $F_{26}P$ + ADP is catalysed by the enzyme PFK2 where

- PFK_2 phosphofructokinase-2
- F_6P fructose-6-phosphate
- $F_{26}P$ fructose-2,6-biphosphate

As pointed out by (Garrett and Grisham, 2017) section 22.2a, the PFK2-catalysed reaction forms a cycle with the reaction: $F_{26}P + H_2O \xrightleftharpoons{F26BP} F_6P + Pi$ where:

- $F_{26}BP$ fructose-2,6-biphosphatase
- Pi inorganic phosphate

The species which activate PFK2 and inhibit F26BP include:

- AMP
- F_6P fructose-6-phosphate

Thus this pair of reactions is a further example of Cyclic Flow Modulation (CFM).
Moreover, the PFK and PFK2 CFMs stongly interact:

- The PFK CFM is positively modulated by the product of the PFK2 CFM: $F_{26}P$
- the PFK2 CFM is positively modulated by the substrate of the PKF (and PFK2) CFM: F_6P
- both are positively modulated by AMP.
- this has been sugested as a mechanism for **integral action** (Cloutier and Wellstead, 2010).

TIGAR (TP53-induced gycolysis and apoptosis regulator) mimics $F_{26}P$; this is related to oncogenesis (Garrett and Grisham, 2017)

5.1 Fructose-2,6-phosphate ($F_{26}P$) CFM as an integrator

(Cloutier and Wellstead, 2010) suggest that the reaction catalysed by PFK2 generating $F_{26}P$ can be used as an integrator based on the fact that $F_{26}P$ is a strong activator of PFK. Their model involves a single irreversible reaction $F_6P + ATP \xrightarrow{PFK2} F_{26}P + ADP$; the basic idea is that the the concentration $F_{26}P$ is the integral of the molar flow which is modulated by AMP.

Within the CFM context, a similar effect can be acheived by *not* setting species B to be a chemostat and its state will indeed be the integral of the net CFM flow. However, unlike a true integrator, this flow will depend on the amount of B x_B anf thus the CFM in these circumstances will only approximate an integrator. This appproximation will depend on the parameters of the CFM itself.

The approximation will look like a high-gain low-pass filter rather than an integrator.

5.2 Simulation

The following simulation illustrates the basic properties for a particular set of parameter. The key changes are:

- **Ce:B** is no longer a chemostat
- $K_B = 0.01$

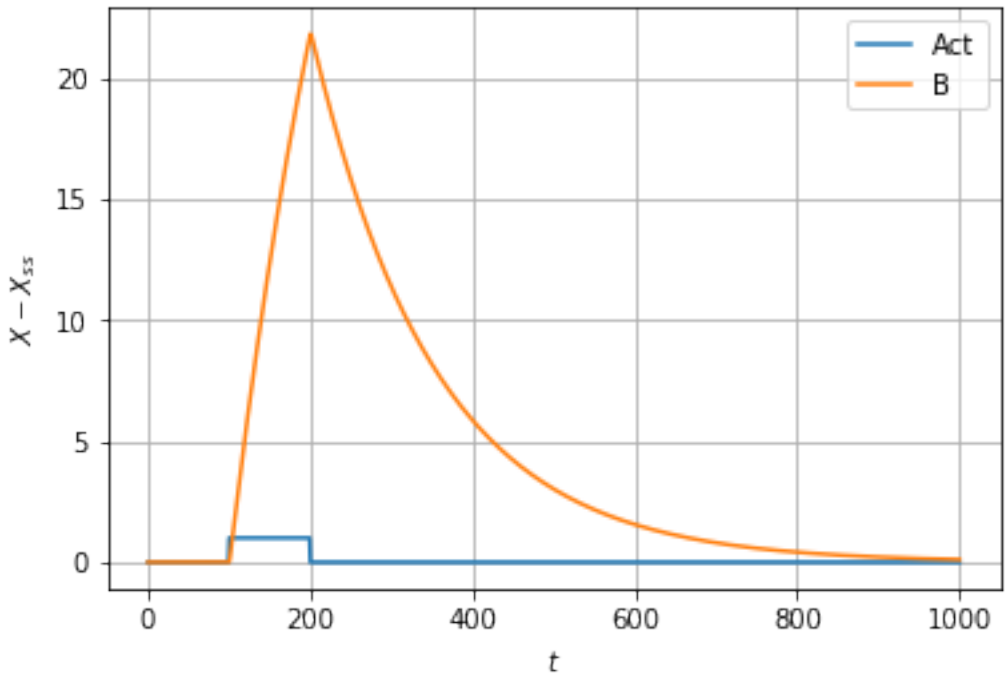
The steady-state is computed when $x_{Act} = 1$. The simulation starts from the steady-state and

$$x_{Act} = \begin{cases} 2 & \text{for } 100 < t < 200 \\ 1 & \text{otherwise} \end{cases} \tag{22}$$

The plot shows deviations of x_{Act} and x_B from the steady state.

In [12]:
```
t = np.linspace(0,10000)
scB = st.statify(s,chemostats=['A','Act','Inh','AAf','BBf','AAr','BBr'])
X_chemo=None
chemo = '1+{0}*(np.heaviside(t-{1},1) - np.heaviside(t-{2},1))'.format(1,100,200)
X_chemo =  {'Act':chemo}
print(X_chemo)
N=2
e0 = 1

for Integrator in [True]:
    parameter,X0 = setParameter(s,abs(N),e0,Integrator=Integrator)

    ## Get the steady-state
    dat = st.sim(s,sc=scB,t=t,parameter=parameter,X0=X0,X_chemo=None,quiet=quiet)
    X_ss = dat['X'][-1]


    ## Simulate from steady-state
    t = np.linspace(0,1000,1000)
    dat = st.sim(s,sc=scB,t=t,parameter=parameter,X0=X_ss,X_chemo=X_chemo,quiet=quiet)
    st.plot(s,dat,species=['Act','B'],reaction=[],x_ss=X_ss)
```

```
{'Act': '1+1*(np.heaviside(t-100,1) - np.heaviside(t-200,1))'}
```



5.3 Discussion

- In the context of the fructose-2,6-phosphate ($F_{26}P$) CFM, the activator Act is AMP and the product B is $F_{26}P$
- The step change in AMP activation at time t=100 gives rise to an increasing value of $F_{26}P$: this is similar to an integrator response.
- When the activation ceases, the amount of $F_{26}P$ decays.
- As $F_{26}P$ is an activator of PFK, the behaviour would give rise to a similar increase and then decrease of the flow through the PFK reaction.
- Thus PFK + PFK-2 act as a proportional + integral (PI) controller in the context of regulating energy levels (as measured by AMP) via metabolism.

References

Athel Cornish-Bowden. *Fundamentals of enzyme kinetics*. Wiley-Blackwell, London, 4th edition, 2013. ISBN 978-3-527-33074-4.

Reginald H. Garrett and Charles M. Grisham. *Biochemistry*. Cengage Learning, Boston, MA, 6th edition, 2017.

Peter J. Gawthrop and Edmund J. Crampin. Energy-based analysis of biochemical cycles using bond graphs. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, 470(2171):1–25, 2014. doi:10.1098/rspa.2014.0459. Available at arXiv:1406.2447.

Peter Cudmore, Peter J. Gawthrop, Michael Pan, and Edmund J. Crampin. Computer-aided modelling of complex physical systems with BondGraphTools. Available at arXiv:1906.10799, Jun 2019.

Mathieu Cloutier and Peter Wellstead. The control systems structures of energy metabolism. *Journal of The Royal Society Interface*, 7(45):651–665, 2010. doi:10.1098/rsif.2009.0371.