The *Escherichia coli* Core Model: Modular Energetic Bond Graph Analysis of Glycolysis and Pentose Phosphate Pathways

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

January 22, 2020

Contents

Note: this is the EcoliCoreModelEnergy.ipynb notebook. The PDF version "The Escherichia coli Core Model: Modular Energetic Bond Graph Analysis of Glycolysis and Pentose Phosphate Pathways" is available here.

1 Introduction

As discussed in a companion notebook, the Network Thermodynamics/Bond Graph approach of (Oster et al., 1971, 1973) extended by (Gawthrop and Crampin, 2014, 2016, 2017) to modelling biomolecular systems of interest to systems biologists developed independently from the stoichiometric approach (Palsson, 2006, 2011, 2015).

However, the conceptual point of intersection of the two approaches is the fact that the stoichiometric matrix is the modulus of the conceptual multiport transformer linking reactions to species. This was pointed out by (Cellier and Greifeneder, 2009). This means that the two approaches are complementary and each can build on the strengths of the other.

In particular, as discussed here, the Bond Graph approach adds energy to stoichiometry.

This notebook focuses on building modular models of metabolism and consequent pathway analysis based on the Escherichia coli Core Model (Orth et al., 2010); in particular, the Glycolysis and Pentose Phosphate portion is extracted and analysed. Following the discussion in the textbook of (Garrett and Grisham, 2017), section 22.6d, various possible pathways are examined by choosing appropriate chemostats and flowstats. (Gawthrop and Crampin, 2018)

Assuming steady-state conditions, the corresponding pathway potentials (Gawthrop, 2017) are derived.

1.1 Import some python code

The bond graph analysis uses a number of Python modules:

```
In [1]: ## Maths library
        import numpy as np

        ## BG tools
        import BondGraphTools as bgt

        ## BG stoichiometric utilities
        import stoich as st

        ## Stoichiometric conversion
        import CobraExtract as Extract
        import stoichBondGraph as stbg

        ## Potentials
        import phiData

        ## Faraday constant
        import scipy.constants as con
        F = con.physical_constants['Faraday constant'][0]

        ## Display
```

```
import IPython.display as disp

## Allow output from within functions
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"

## Units etc
factor = 1
units = ' V'

## Control output
quiet = True
computePhi = True
showMu = True
```

1.2 Deriving species potentials

To perform energetic analysis it is necessary to have values of the chemical potential of the species involved. One way of this is to use experimentally derived value of species potentials at standard conditions and then derive potentials corresponding to the concentrations of the species. Another approach used here, is to take experimental values of reaction potentials Φ (Park et al., 2016) and derive a consistent set of species potentials ϕ using $\phi = -N^{\dagger}\Phi$ where N is the stoichiometric matrix of the reaction system and $\dagger$ denotes pseudo inverse.

```
In [2]: def getPhi(s):
            """Extract phi for given system using
            Reaction potentials from ParRubXu16"""

            ## Reaction potentials from ParRubXu16
            PHI = phiData.Phi_ParRubXu16()
            Phi_reac = PHI['Ecoli']

            ## Reaction potential (33.9e3) from GarGri17
            Phi_reac['GLCPTS'] = 33.9e3/F - Phi_reac['PYK']
            print('Setting Phi for reaction GLCPTS to', int(Phi_reac['GLCPTS']*1000),'mV.')

            Phi = np.zeros((len(s['reaction']),1))
            for i,reac in enumerate(s['reaction']):
                if reac in Phi_reac.keys():
                    Phi[i] = Phi_reac[reac]
                else:
                    min = 0.01          # 10mV
                    print('Setting Phi for reaction','\\ch{'+reac+'}','to', min*1000, 'mV. \n')
                    Phi[i] = min

            pinvN =  np.linalg.pinv(s['N'].T)
            phi = -pinvN@Phi
            return Phi,phi
```

2 Extract the model

2.1 Extract full ecoli core model from the CobraPy representation

```
In [3]: sm = Extract.extract(cobraname='textbook',Remove=['_C','__' ], negReaction=['RPI'], quie
```

```
Extracting stoichiometric matrix from: textbook
Cobra Model name: e_coli_core BondGraphTools name: e_coli_core_abg
Extract.Integer only handles one non-integer per reaction
Multiplying reaction BIOMASS_ECOLIORE ( 12 ) by 0.6684491978609626 to avoid non-integer species
Multiplying reaction CYTBD ( 15 ) by 2.0 to avoid non-integer species O2 ( 55 )
Multiplying reaction RPI ( 65 ) by -1
```

2.2 Extract Glycolysis and Pentose Phosphate Pathways

```
In [4]: name = 'GlyPPP_abg'
        reaction = ['GLCPTS','PGI','PFK','FBA','TPI','GAPD','PGK','PGM','ENO','PYK']
        reaction += ['G6PDH2R','PGL','GND','RPI','TKT2','TALA','TKT1','RPE']
        sGlyPPP = Extract.choose(sm,reaction=reaction)
        Phi,phi = getPhi(sGlyPPP)
        sGlyPPP['name'] = name
        stbg.model(sGlyPPP)
        import GlyPPP_abg
```

```
Setting Phi for reaction GLCPTS to 277 mV.
Setting Phi for reaction \ch{G6PDH2R} to 10.0 mV.

Setting Phi for reaction \ch{PGL} to 10.0 mV.
```

2.3 Display the extracted reactions

- () indicates reaction potential in Volts (J/coulomb)
- [] indicates reaction free energy in J/mol

See (Gawthrop, 2017) for a discussion of these two quantities.

```
In [5]: disp.Latex(st.sprintrl(sGlyPPP,chemformula=True,Phi=Phi,showMu=showMu))
```

```
    Out[5]:
```

$$\begin{aligned}
\mathrm{GLCD_E + PEP} &\xrightleftharpoons{\mathrm{GLCPTS}} \mathrm{G_6P + PYR} && (0.28)\ [-26.81]\\
\mathrm{G_6P} &\xrightleftharpoons{\mathrm{PGI}} \mathrm{F_6P} && (0.02)\ [-1.60]\\
\mathrm{ATP + F_6P} &\xrightleftharpoons{\mathrm{PFK}} \mathrm{ADP + FDP + H} && (0.26)\ [-24.71]\\
\mathrm{FDP} &\xrightleftharpoons{\mathrm{FBA}} \mathrm{DHAP + G_3P} && (0.02)\ [-1.98]\\
\mathrm{DHAP} &\xrightleftharpoons{\mathrm{TPI}} \mathrm{G_3P} && (0.01)\ [-0.79]\\
\mathrm{G_3P + NAD + PI} &\xrightleftharpoons{\mathrm{GAPD}} \mathrm{{}_{13}DPG + H + NADH} && (0.01)\ [-1.32]\\
\mathrm{{}_3PG + ATP} &\xrightleftharpoons{\mathrm{PGK}} \mathrm{{}_{13}DPG + ADP} && (0.01)\ [-1.42]\\
\mathrm{{}_2PG} &\xrightleftharpoons{\mathrm{PGM}} \mathrm{{}_3PG} && (0.03)\ [-3.17]\\
\mathrm{{}_2PG} &\xrightleftharpoons{\mathrm{ENO}} \mathrm{H_2O + PEP} && (0.03)\ [-2.75]\\
\mathrm{ADP + H + PEP} &\xrightleftharpoons{\mathrm{PYK}} \mathrm{ATP + PYR} && (0.07)\ [-7.09]
\end{aligned}$$

$$\begin{aligned}
\mathrm{G_6P + NADP} &\xrightleftharpoons{\mathrm{G_6PDH_2R}} \mathrm{{}_6PGL + H + NADPH} && (0.01)\ [-0.96]\\
\mathrm{{}_6PGL + H_2O} &\xrightleftharpoons{\mathrm{PGL}} \mathrm{{}_6PGC + H} && (0.01)\ [-0.96]\\
\mathrm{{}_6PGC + NADP} &\xrightleftharpoons{\mathrm{GND}} \mathrm{CO_2 + NADPH + RU_5PD} && (0.16)\ [-15.08]\\
\mathrm{RU_5PD} &\xrightleftharpoons{\mathrm{RPI}} \mathrm{R_5P} && (0.00)\ [-0.00]\\
\mathrm{E_4P + XU_5PD} &\xrightleftharpoons{\mathrm{TKT_2}} \mathrm{F_6P + G_3P} && (0.02)\ [-1.61]\\
\mathrm{G_3P + S_7P} &\xrightleftharpoons{\mathrm{TALA}} \mathrm{E_4P + F_6P} && (0.06)\ [-5.43]\\
\mathrm{R_5P + XU_5PD} &\xrightleftharpoons{\mathrm{TKT_1}} \mathrm{G_3P + S_7P} && (0.00)\ [-0.40]\\
\mathrm{RU_5PD} &\xrightleftharpoons{\mathrm{RPE}} \mathrm{XU_5PD} && (0.00)\ [-0.08]
\end{aligned}$$

2.4 Code to analyse pathways defined by chemostats and flowstats

```
In [6]: ## Analyse pathways defined by chemostats and flowstats
        def ch(name):
            return '\\ch{'+name+'}'

        def energetics(s,sp,phi):
            """Reaction energetics.
            """

            ## Phi for all reactions
            Phi = -s['N'].T@phi

            ##Phi for pathway
            ## I is the relevant indices of phi
            I = []
```

```
    for spec in sp['species']:
        i = s['species'].index(spec)
        I.append(i)

    Phip = -sp['N'].T@phi[I]

    return Phi,Phip

def pathway(bg,phi,chemostats,flowstats=[],computePhi=False,verbose=False):
    """ Analyse pathways
    """

    print('Chemostats:',sorted(chemostats))
    print('Flowstats:', sorted(flowstats))
    ## Stoichiometry
    ## Create stoichiometry from bond graph.
    s = st.stoich(bg,quiet=True)

    ## Stoichiometry with chemostats
    sc = st.statify(s,chemostats=chemostats,flowstats=flowstats)

    ## Pathway stoichiometry
    sp = st.path(s,sc)

    ## Print info
    if verbose:
        for stat in sorted(chemostats):
            print(ch(stat)+',')

    ## Energetics
    if computePhi:
        Phi,Phip = energetics(s,sp,phi)
        #print('Phi units: kJ/mol')
#         fac = -F/1000
#         units='~\si{\kilo\joule\per\mol}'
        units = '~\si{\volt}'
        print(st.sprintp(sc))
        disp.Latex(st.sprintrl(sp,chemformula=True,Phi=Phip,showMu=showMu))
        #return s,sc,sp,Phi*fac,Phip*fac,units
        return s,sc,sp,Phip
    else:
        print(st.sprintrl(sp,chemformula=True))
        Phip = 0
        return s,sc,sp,Phip
```

3 Analyse Pentose Phosphate Pathway with Glycolysis - Chemostats

3.1 Glycolysis

```
In [7]: print('Glycolysis')
        import GlyPPP_abg
        chemostats = ['H2O','H']
        chemostats += ['ADP','ATP','PI']
        chemostats += ['G6P','PYR','NAD','NADH']
        s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,computePhi=computePhi)
        disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
Glycolysis
Chemostats: ['ADP', 'ATP', 'G6P', 'H', 'H2O', 'NAD', 'NADH', 'PI', 'PYR']
Flowstats: []
1 pathways
0:  + PGI + PFK + FBA + TPI + 2 GAPD - 2 PGK - 2 PGM + 2 ENO + 2 PYK
```

Out[7]:

$$3\,ADP + G_6P + 2\,NAD + 2\,PI \xrightleftharpoons{pr_1} 3\,ATP + H + 2\,H_2O + 2\,NADH + 2\,PYR \quad (0.44)\ [-42.22]$$

- The pathway reaction pr_1 is the overall glycolysis reaction from G6P to PYR Garrett and Grisham (2017, § 18.2).
- The positive reaction potential (negative reaction free energy) indicates that the reaction proceeds in the forward direction.

3.2 R_5P and NADPH generation

```
In [8]: print('R5P and NADPH generation')
        chemostats = ['H2O','H']
        chemostats += ['NADP','NADPH','CO2']
        chemostats += ['G6P','R5P']
        s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,computePhi=computePhi)
        disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
R5P and NADPH generation
Chemostats: ['CO2', 'G6P', 'H', 'H2O', 'NADP', 'NADPH', 'R5P']
Flowstats: []
1 pathways
0:  + G6PDH2R + PGL + GND + RPI
```

Out[8]:

$$G_6P + H_2O + 2\,NADP \xrightleftharpoons{pr_1} CO_2 + 2\,H + 2\,NADPH + R_5P \qquad (0.18)\ [-17.01]$$

- The pathway reaction P_1 corresponds to the R_5P and NADPH synthesis discussed in comment 1 of (Garrett and Grisham, 2017), p787.

- The positive reaction potential (negative reaction free energy) indicates that the reaction proceeds in the forward direction.

3.3 R_5P generation

```
In [9]: print('R5P generation')
        chemostats = ['H2O','H']
        chemostats += ['G6P','R5P']
        chemostats += ['ADP','ATP']
        s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,computePhi=computePhi)
        disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
R5P generation
Chemostats: ['ADP', 'ATP', 'G6P', 'H', 'H2O', 'R5P']
Flowstats: []
1 pathways
0:  - 5 PGI - PFK - FBA - TPI - 4 RPI + 2 TKT2 + 2 TALA + 2 TKT1 + 4 RPE
```

Out[9]:

$$\mathrm{ADP} + \mathrm{H} + 6\,\mathrm{R_5P} \xrightleftharpoons{pr_1} \mathrm{ATP} + 5\,\mathrm{G_6P} \qquad (-0.21)\ [20.30]$$

- The pathway reaction pr_1 corresponds to the R_5P synthesis discussed in comment 2 of (Garrett and Grisham, 2017), p787.
- The *negative* reaction potential (*positive* reaction free energy) indicates that the reaction proceeds in the *reverse* direction.

3.4 NADPH generation

```
In [10]: import GlyPPP_abg
         chemostats = ['H2O','H']
         chemostats += ['G6P']
         chemostats += ['NADP','NADPH','CO2']
         chemostats += ['ATP','ADP']
         s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,computePhi=computePhi)
         disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
Chemostats: ['ADP', 'ATP', 'CO2', 'G6P', 'H', 'H2O', 'NADP', 'NADPH']
Flowstats: []
1 pathways
0:  - 5 PGI - PFK - FBA - TPI + 6 G6PDH2R + 6 PGL + 6 GND + 2 RPI + 2 TKT2 + 2 TALA + 2 TKT1 + 4
```

Out[10]:

$$ADP + G_6P + 6\,H_2O + 12\,NADP \xrightleftharpoons{pr_1} ATP + 6\,CO_2 + 11\,H + 12\,NADPH \quad (0.85)\ [-81.79]$$

- The pathway reaction pr_1 corresponds to the NADPH synthesis discussed in comment 3 of (Garrett and Grisham, 2017), p787.
- The positive reaction potential (negative reaction free energy) indicates that the reaction proceeds in the forward direction.

3.5 NADPH and ATP generation

```
In [11]: import GlyPPP_abg
         chemostats = ['H2O','H']
         chemostats += ['NADP','NADPH','CO2']
         chemostats += ['G6P']
         chemostats += ['ADP','ATP','PI']
         chemostats += ['PYR','NAD','NADH']
         flowstats = ['PGI']
         s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,flowstats=flowstats,computePhi
         disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
Chemostats: ['ADP', 'ATP', 'CO2', 'G6P', 'H', 'H2O', 'NAD', 'NADH', 'NADP', 'NADPH', 'PI', 'PYR'
Flowstats: ['PGI']
2 pathways
0:  + PGI
1:  + 2 PFK + 2 FBA + 2 TPI + 5 GAPD - 5 PGK - 5 PGM + 5 ENO + 5 PYK + 3 G6PDH2R + 3 PGL + 3 GND
```

Out[11]:

$$8\,ADP + 3\,G_6P + 5\,NAD + 6\,NADP + 5\,PI \xrightleftharpoons{pr_1} 8\,ATP + 3\,CO_2 + 8\,H + 2\,H_2O + 5\,NADH + 6\,NADPH + 5\,PYR$$

- The pathway reaction P_1 corresponds to the NADPH and ATP synthesis discussed in comment 4 of (Garrett and Grisham, 2017), p787.
- The positive reaction potential (negative reaction free energy) indicates that the reaction proceeds in the forward direction.

4 Analyse Pentose Phosphate Pathway with Glycolysis - Flowstats

The pathways may also be isolated by using appropriate (zero-flow) flowstats. The comments for each section are the same as in the previous section.

4.1 Common chemostats

```
In [12]: import GlyPPP_abg
         Chemostats = ['G6P','ADP','ATP','CO2','H','H2O','NAD','NADH','NADP','NADPH','PI','PYR']
```

4.2 Glycolysis

- The glycolysis pathway is isolated from the pentose phosphate pathway by replacing the two connecting reactions (G6PDH2R and TKT2) by flowstats.

```
In [13]: print('Glycolysis')
         chemostats = Chemostats
         flowstats = ['G6PDH2R','TKT2']
         s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,flowstats=flowstats,computePhi
         disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
Glycolysis
Chemostats: ['ADP', 'ATP', 'CO2', 'G6P', 'H', 'H2O', 'NAD', 'NADH', 'NADP', 'NADPH', 'PI', 'PYR'
Flowstats: ['G6PDH2R', 'TKT2']
3 pathways
0:  + PGI + PFK + FBA + TPI + 2 GAPD - 2 PGK - 2 PGM + 2 ENO + 2 PYK
1:  + G6PDH2R
2:  + TKT2
```

Out[13]:

$$3\,ADP + G_6P + 2\,NAD + 2\,PI \xrightleftharpoons{pr_1} 3\,ATP + H + 2\,H_2O + 2\,NADH + 2\,PYR \quad (0.44)\ [-42.22]$$

4.3 R_5P and NADPH generation

- This pathway is isolated by setting PGI and TKT2 as flowstats and the product R_5P is added to the chemostat list.

```
In [14]: print('R5P and NADPH generation')
         chemostats = Chemostats + ['R5P']
         flowstats = ['PGI','TKT2']
         #s,sc,sp,Phip,Phi,Phip,units = pathway(GlyPPP_abg.model(),phi,chemostats,flowstats=flow
         s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,flowstats=flowstats,computePhi
         disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
R5P and NADPH generation
Chemostats: ['ADP', 'ATP', 'CO2', 'G6P', 'H', 'H2O', 'NAD', 'NADH', 'NADP', 'NADPH', 'PI', 'PYR'
Flowstats: ['PGI', 'TKT2']
3 pathways
0:  + PGI
1:  + G6PDH2R + PGL + GND + RPI
2:  + TKT2
```

Out[14]:

$$G_6P + H_2O + 2\,NADP \xrightleftharpoons{pr_1} CO_2 + 2\,H + 2\,NADPH + R_5P \qquad (0.18)\ [-17.01]$$

4.4 R_5P generation

- This pathway is isolated by setting GAPD and G6PDH2R as flowstats and the product R_5P is added to the chemostat list.

```
In [15]: print('R5P generation')
         chemostats = Chemostats + ['R5P']
         flowstats = ['GAPD','G6PDH2R']
         s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,flowstats=flowstats,computePhi
         disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
R5P generation
Chemostats: ['ADP', 'ATP', 'CO2', 'G6P', 'H', 'H2O', 'NAD', 'NADH', 'NADP', 'NADPH', 'PI', 'PYR'
Flowstats: ['G6PDH2R', 'GAPD']
3 pathways
0:  + GAPD
1:  + G6PDH2R
2:  - 5 PGI - PFK - FBA - TPI - 4 RPI + 2 TKT2 + 2 TALA + 2 TKT1 + 4 RPE
```

Out[15]:

$$ADP + H + 6\,R_5P \xrightleftharpoons{pr_1} ATP + 5\,G_6P \qquad (-0.21)\ [20.30]$$

4.5 NADPH generation

- This pathway is isolated by setting GAPD as a flowstat.

```
In [16]: print('NADPH generation')
         chemostats = Chemostats
         flowstats = ['GAPD']
         s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,flowstats=flowstats,computePhi
         disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
NADPH generation
Chemostats: ['ADP', 'ATP', 'CO2', 'G6P', 'H', 'H2O', 'NAD', 'NADH', 'NADP', 'NADPH', 'PI', 'PYR'
Flowstats: ['GAPD']
2 pathways
0:  + GAPD
1:  - 5 PGI - PFK - FBA - TPI + 6 G6PDH2R + 6 PGL + 6 GND + 2 RPI + 2 TKT2 + 2 TALA + 2 TKT1 + 4
```

Out[16]:

$$ADP + G_6P + 6\,H_2O + 12\,NADP \xrightleftharpoons{pr_1} ATP + 6\,CO_2 + 11\,H + 12\,NADPH \qquad (0.85)\ [-81.79]$$

4.6 NADPH and ATP generation

This pathway is isolated by setting PGI as flowstat.

```
In [17]: print('NADPH and ATP generation')
         chemostats = Chemostats
         flowstats = ['PGI']
         s,sc,sp,Phip = pathway(GlyPPP_abg.model(),phi,chemostats,flowstats=flowstats,computePhi
         disp.Latex(st.sprintrl(sp,chemformula=True,Phi=factor*Phip,showMu=showMu))
```

```
NADPH and ATP generation
Chemostats: ['ADP', 'ATP', 'CO2', 'G6P', 'H', 'H2O', 'NAD', 'NADH', 'NADP', 'NADPH', 'PI', 'PYR'
Flowstats: ['PGI']
2 pathways
0:  + PGI
1:  + 2 PFK + 2 FBA + 2 TPI + 5 GAPD - 5 PGK - 5 PGM + 5 ENO + 5 PYK + 3 G6PDH2R + 3 PGL + 3 GND
```

Out[17]:

$$8\,ADP + 3\,G_6P + 5\,NAD + 6\,NADP + 5\,PI \xrightleftharpoons{pr_1} 8\,ATP + 3\,CO_2 + 8\,H + 2\,H_2O + 5\,NADH + 6\,NADPH + 5\,PYR$$

References

F.E. Cellier and J. Greifeneder. Modeling chemical reactions in modelica by use of chemo-bonds. In *Proceedings 7th Modelica Conference*, Como, Italy, September 2009.

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

P. J. Gawthrop. Bond graph modeling of chemiosmotic biomolecular energy transduction. *IEEE Transactions on NanoBioscience*, 16(3):177–188, April 2017. ISSN 1536-1241. doi:10.1109/TNB.2017.2674683. Available at arXiv:1611.04264.

P. J. Gawthrop and E. J. Crampin. Modular bond-graph modelling and analysis of biomolecular systems. *IET Systems Biology*, 10(5):187–201, October 2016. ISSN 1751-8849. doi:10.1049/iet-syb.2015.0083. Available at arXiv:1511.06482.

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 J. Gawthrop and Edmund J. Crampin. Energy-based analysis of biomolecular pathways. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 473 (2202), 2017. ISSN 1364-5021. doi:10.1098/rspa.2016.0825. Available at arXiv:1611.02332.

Peter J. Gawthrop and Edmund J. Crampin. Biomolecular system energetics. In *Proceedings of the 13th International Conference on Bond Graph Modeling (ICBGM'18)*, Bordeaux, 2018. Society for Computer Simulation. Available at arXiv:1803.09231.

J. Orth, R. Fleming, and B. Palsson. Reconstruction and use of microbial metabolic networks: the core escherichia coli metabolic model as an educational guide. *EcoSal Plus*, 2010. doi:10.1128/ecosalplus.10.2.1.

George Oster, Alan Perelson, and Aharon Katchalsky. Network thermodynamics. *Nature*, 234: 393–399, December 1971. doi:10.1038/234393a0.

George F. Oster, Alan S. Perelson, and Aharon Katchalsky. Network thermodynamics: dynamic modelling of biophysical systems. *Quarterly Reviews of Biophysics*, 6(01):1–134, 1973. doi:10.1017/S0033583500000081.

Bernhard Palsson. *Systems biology: properties of reconstructed networks*. Cambridge University Press, 2006. ISBN 0521859034.

Bernhard Palsson. *Systems Biology: Simulation of Dynamic Network States*. Cambridge University Press, 2011.

Bernhard Palsson. *Systems Biology: Constraint-Based Reconstruction and Analysis*. Cambridge University Press, Cambridge, 2015.

Junyoung O. Park, Sara A. Rubin, Yi-Fan Xu, Daniel Amador-Noguez, Jing Fan, Tomer Shlomi, and Joshua D. Rabinowitz. Metabolite concentrations, fluxes and free energies imply efficient enzyme usage. *Nat Chem Biol*, 12(7):482–489, Jul 2016. ISSN 1552-4450. doi:10.1038/nchembio.2077.