```
In[7]:= Quit[]
```

Import packages, set up model and functions

Import MASS toolbox

```
In[1]:= << Toolbox`
        << Toolbox`Style`
```

Define working directory

```
In[3]:= SetDirectory[NotebookDirectory[]];
```

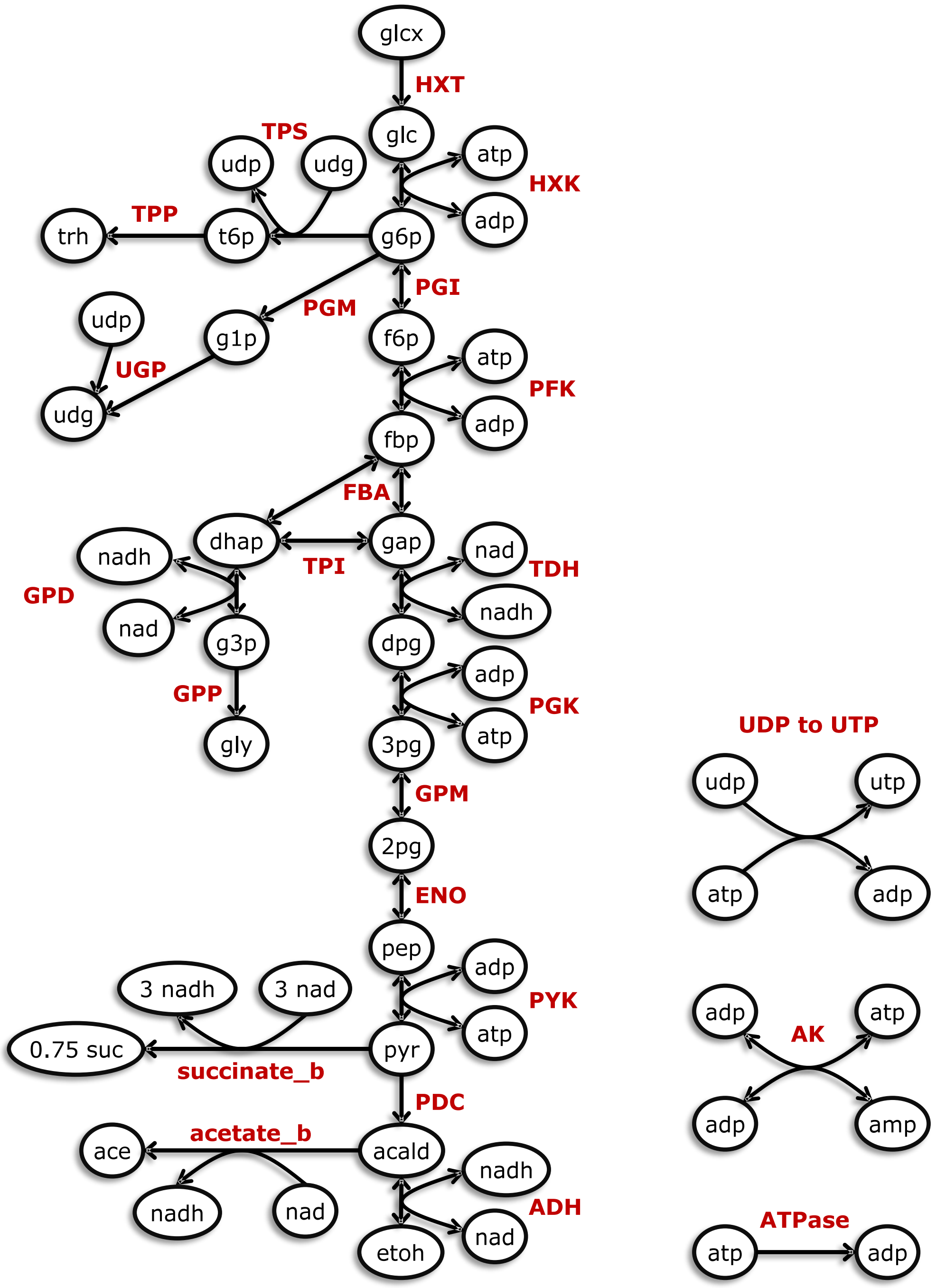
Import Smallbone model

```
In[4]:= modelSmallbone18 = sbml2model["MODEL1303260018.xml"];
```

See model structure (no regulation included here)

In[5]:=
```
First@Import["glycolysis_yeast.pdf"]
```

Out[5]=



Define starveSystem function

```
In[6]:= starveSystem[starveTime_, starveGlc_, originalModel_] :=
         Block[{modelTemp, cSmallbone, fSmallbone},

          modelTemp = originalModel;
          {cSmallbone, fSmallbone} = simulate[modelTemp,
              {t, 0, starveTime}, Parameters → {GLCx^extracellular → starveGlc}];
          updateInitialConditions[modelTemp, cSmallbone /. t → starveTime];

          Return[{modelTemp, cSmallbone, fSmallbone}];];
```

Define exportData function

```
In[7]:= exportData[outputFile_, data_, simLength_, timeInterval_: 0.001] :=
           Block[{concDataToExport},

             concDataToExport =
                  Table[
                       Flatten@{tPoint, Values@data /. t → tPoint},
                    {tPoint, 0, simLength, timeInterval}];

                concDataToExport =
              Insert[concDataToExport, Flatten@{"t", Map[getID@# &, Keys@data]}, 1];

                Export[outputFile, concDataToExport, "CSV"];

           ];
```

Simulate cell starvation

Do starvation simulation

Define starvation time, glucose concentration during starvation, and glucose concentration for the glucose pulse

```
In[8]:= starveGlc = 0.1; (* in mM/L *)
        starveTime = 1200; (* in seconds *)
```

Starve the model, a new model is returned whose initial metabolite concentrations are the concentrations at the end of the starvation period

```
In[10]:= {model, concStarv, fluxStarv} = starveSystem[starveTime, starveGlc, modelSmallbone18];
```

```
NDSolve: NDSolve has computed initial values that give a zero residual for the differential-algebraic system,
  but some components are different from those specified. If you need them to be satisfied, giving initial
  conditions for all dependent variables and their derivatives is recommended.
```

Export simulation data

```
In[11]:= exportData["conc_starvation.csv", concStarv, starveTime, 0.1];
         exportData["flux_starvation.csv", fluxStarv, starveTime, 0.1];
```

Plot metabolite and flux time courses during starvation

Simulate glucose pulse, with external glucose concentration set to 4 mM

Simulate the glucose pulse

```
In[13]:= glcPulse = 4; (*glucose concetration in mmol/L*)
         glcPulseSimulationTime = 60; (*in seconds*)
```

```
In[15]:= {concentrationGlcPulse, fluxGlcPulse} = simulate[model,
            {t, 0, glcPulseSimulationTime}, Parameters → {GLCx^extracellular → glcPulse}];
```

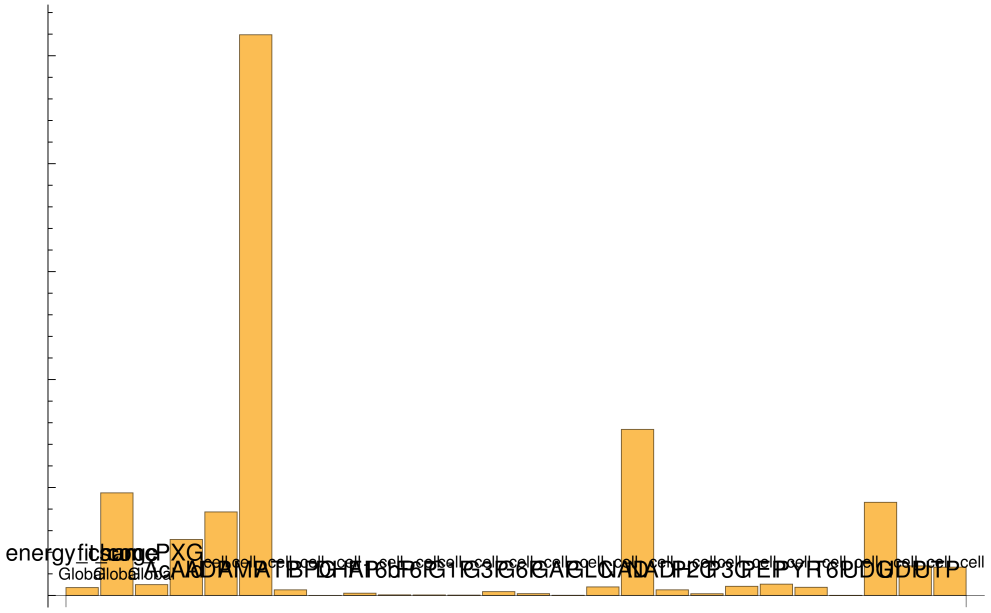
Export simulation data

```
In[16]:= exportData["conc_glc_pulse.csv", concentrationGlcPulse, 60, 0.01];
         exportData["flux_glc_pulse.csv", fluxGlcPulse, 60, 0.01];
```

Plot initial metabolite concentrations, to see which metabolites accumulated during starvation

```
In[18]:= BarChart[Values@model["InitialConditions"],
          ChartLabels → Placed[Keys@model["InitialConditions"], Bottom]]
```

Out[18]=

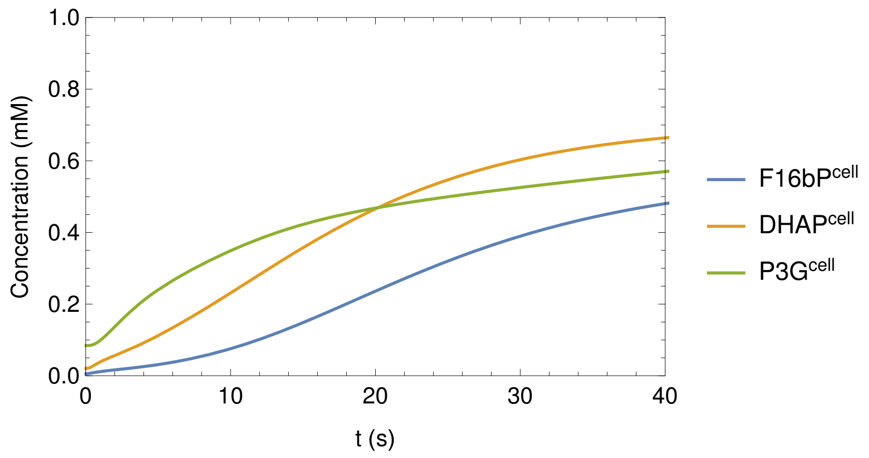


Plot simulation results

Figure 5B

```
In[19]:= plot = plotSimulation[filter[concentrationGlcPulse, {F16bP^cell, DHAP^cell, P3G^cell}],
            PlotFunction → Plot, PlotRange → {{0, 40}, {0, 1}},
            Legend → True, FrameLabel → {"t (s)", "Concentration (mM)"}]
```

Out[19]=



Other simulation results

Simulate glucose and acetaldehyde coinjection, with external glucose concentration set to 4 mM and internal acetaldehyde concentration set to 2.3 mM

Simulate the glucose+acetaldehyde pulse

```
In[20]:= {concentrationCoinjection1, fluxCoinjection1} = simulate[model, {t, 0, 60},
            Parameters → {GLCx^extracellular → 4}, InitialConditions → { AcAld^cell → 2.3}];
```

Export simulation data

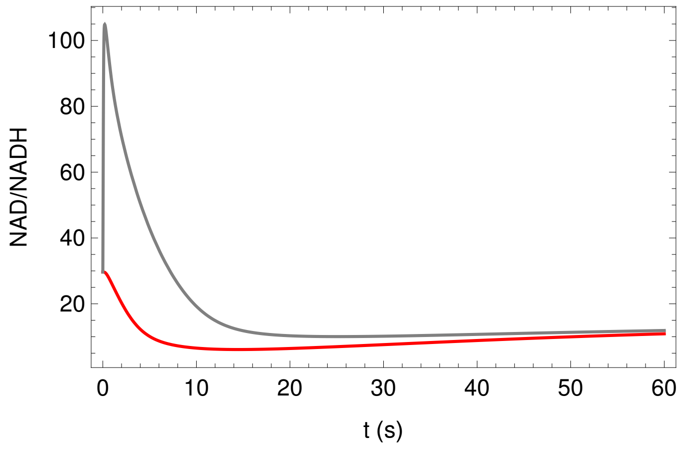
```
In[21]:= exportData["conc_glc_pulse_acald23.csv", concentrationCoinjection1, 60, 0.01];
         exportData["flux_glc_pulse_acald23.csv", fluxCoinjection1, 60, 0.01];
```

Plot simulation results

Figure S5

```
In[23]:= plotGlcPulse = Plot[Values@filter[concentrationGlcPulse, {NAD^cell}] /
             Values@filter[concentrationGlcPulse, {NADH^cell}], {t, 0, 60}, PlotStyle → Red];
         plotCoinjection = Plot[Values@filter[concentrationCoinjection1, {NAD^cell}] / Values@
              filter[concentrationCoinjection1, {NADH^cell}], {t, 0, 60}, PlotStyle → Gray];
         Show[plotGlcPulse, plotCoinjection, FrameLabel → {"t (s)", "NAD/NADH"}]
```

Out[25]=



Simulate second glucose and acetaldehyde coinjection, with external glucose concentration set to 4 mM and internal acetadehyde concentration set to 12 mM

Simulate the glucose+acetaldehyde pulse

```
In[26]:= {concentrationCoinjection2, fluxCoinjection2} = simulate[model, {t, 0, 60},
            Parameters → {GLCx^extracellular → 4}, InitialConditions → { AcAld^cell → 12}];
```

Export simulation data

In[27]:=
```
exportData["conc_glc_pulse_acald12.csv", concentrationCoinjection2, 60, 0.01];
exportData["flux_glc_pulse_acald12.csv", fluxCoinjection2, 60, 0.01];
```

Plot simulation results

Figure S7 - 3PG evolution

In[29]:=
```
plot3PGnoAcald = plotSimulation[filter[concentrationGlcPulse, {P3Gcell}],
    PlotFunction → Plot, PlotRange → {{0, 40}, {0, 1}}, PlotStyle → Blue];
plot3PG12Acald = plotSimulation[filter[concentrationCoinjection2, {P3Gcell}],
    PlotFunction → Plot, PlotRange → {{0, 40}, {0, 1}}, PlotStyle → Green];
Show[plot3PGnoAcald, plot3PG12Acald, FrameLabel → {"t (s)", "3PG (mM)"}]
```

Out[31]=

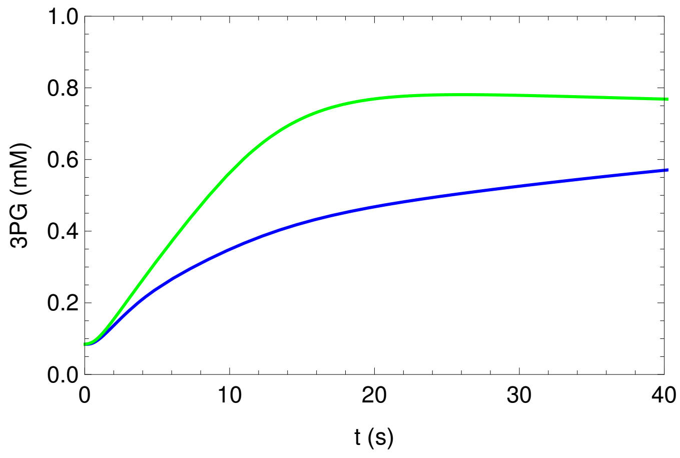


Figure S7 - pyruvate evolution

In[32]:=
```
plotPYRnoAcald = plotSimulation[filter[concentrationGlcPulse, {PYRcell}],
    PlotFunction → Plot, PlotRange → {{0, 40}, {0, 2}}, PlotStyle → Blue];
plotPYR12Acald = plotSimulation[filter[concentrationCoinjection2, {PYRcell}],
    PlotFunction → Plot, PlotRange → {{0, 40}, {0, 2}}, PlotStyle → Green];
Show[plotPYRnoAcald, plotPYR12Acald, FrameLabel → {"t (s)", "Pyr (mM)"}]
```

Out[34]=

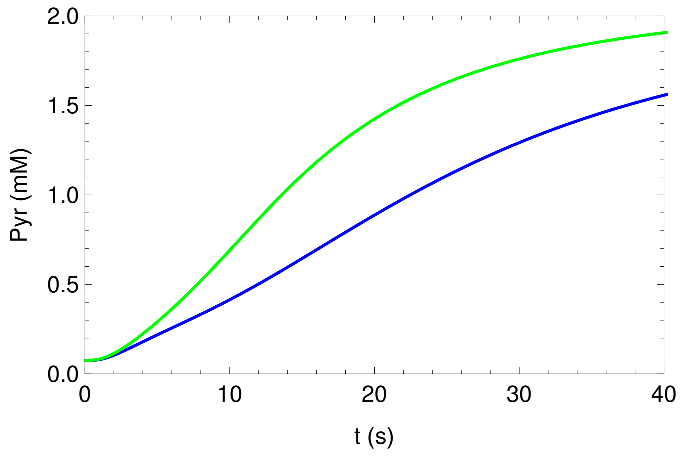
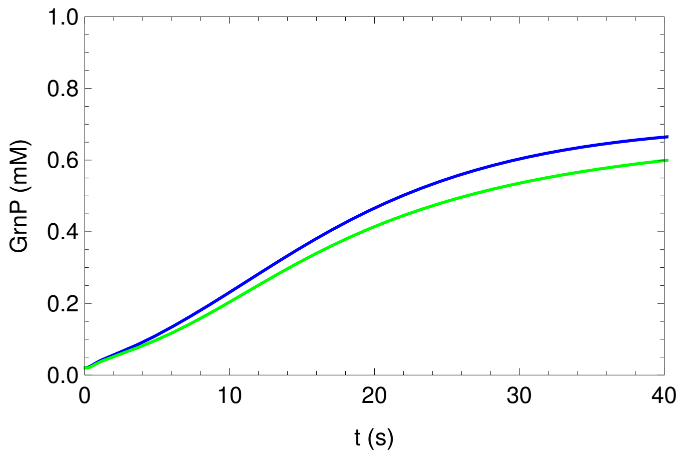


Figure S7 - GrnP evolution

In[35]:=
```
plotGrnPnoAcald = plotSimulation[filter[concentrationGlcPulse, {DHAPcell}],
    PlotFunction → Plot, PlotRange → {{0, 40}, {0, 1}}, PlotStyle → Blue];
plotGrnP12Acald = plotSimulation[filter[concentrationCoinjection2, {DHAPcell}],
    PlotFunction → Plot, PlotRange → {{0, 40}, {0, 1}}, PlotStyle → Green];
Show[plotGrnPnoAcald, plotGrnP12Acald, FrameLabel → {"t (s)", "GrnP (mM)"}]
```

Out[37]=



Energy charge ratio = 0.37: set ADP = 3.01, and ADP/ATP ratio = 4

Simulate glucose pusel with energy charge and acetaldehyde increase

```
In[38]:= {concentrationEnergyCharge, fluxEnergyCharge} = simulate[model, {t, 0, 60},
    InitialConditions → {ADP^cell → 3.01, ATP^cell → 0.75 , AcAld^cell → 1.6426428464870837`},
    Parameters → {GLCx^extracellular → 4}];
```

```
NDSolve: NDSolve has computed initial values that give a zero residual for the differential–algebraic system,
  but some components are different from those specified. If you need them to be satisfied, giving initial
  conditions for all dependent variables and their derivatives is recommended.
```

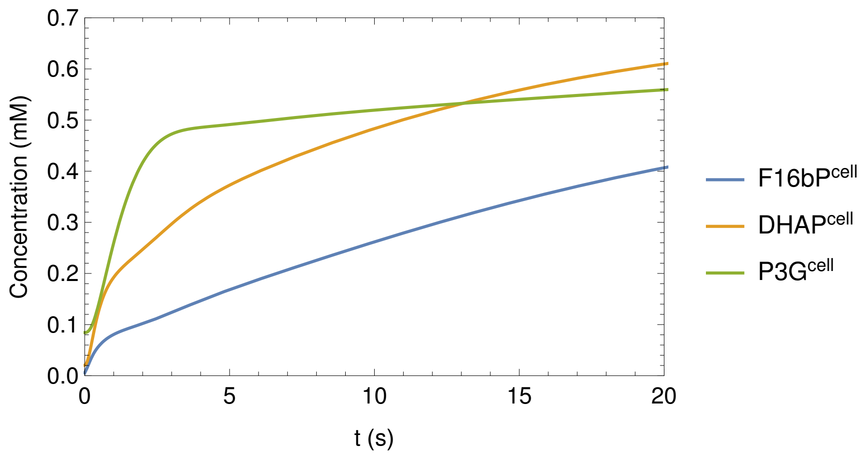
Export simulation data

```
In[39]:= exportData["conc_glc_pulse_S6.csv", concentrationEnergyCharge, 60, 0.01];
exportData["flux_glc_pulse_S6.csv", fluxEnergyCharge, 60, 0.01];
```

Plot simulation results

Figure S6

```
In[41]:=
plot = plotSimulation[
  filter[concentrationEnergyCharge, {F16bP^cell, DHAP^cell, P3G^cell}],
  FrameLabel → {"t (s)", "Concentration (mM)"},
  PlotFunction → Plot, PlotRange → {{0, 20}, {0, 0.7}}, Legend → True]
```

Out[41]=


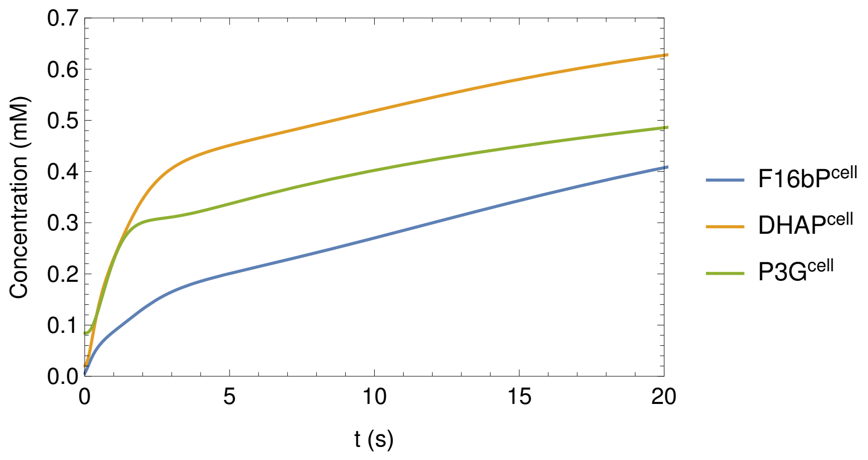
Simulation without AcAld increase

```
In[42]:= {concentrationEnergyChargeNoAcaldChange, fluxEnergyChargeNoAcaldChange} =
           simulate[model, {t, 0, 60}, InitialConditions → {ADP^cell → 3.01, ATP^cell → 0.75},
            Parameters → {GLCx^extracellular → 4}];

         plot = plotSimulation[
           filter[concentrationEnergyChargeNoAcaldChange, {F16bP^cell, DHAP^cell, P3G^cell}],
           FrameLabel → {"t (s)", "Concentration (mM)"}, PlotFunction → Plot,
           PlotRange → {{0, 20}, {0, 0.7}}, Legend → True]
```

```
NDSolve: NDSolve has computed initial values that give a zero residual for the differential-algebraic system,
  but some components are different from those specified. If you need them to be satisfied, giving initial
  conditions for all dependent variables and their derivatives is recommended.
```

Out[43]=


t (s)
Concentration (mM)
F16bP^cell
DHAP^cell
P3G^cell