The Tutorials

Index

tutorial1.nb Shows how to compute elementary flux vectors given a stoichiometry matrix.

tutorial2.nb Introduces the computational geometry routine used by SNA.

tutorial3.nb Basic introduction into constructing, manipulating and analyzing metabolic networks.

tutorial4.nb Application of the ideas in Tutorial 3 to a medium sized real world network.

tutorial5.nb Importing metabolic networks into SNA based on an stoichiometry matrix.

tutorial6.nb Importing metabolic networks into SNA based on a textual representation.

tutorial7.nb Flux cone calculations for large networks I.

tutorial8.nb Conversion cone calculations for large networks and flux cone calculations for large networks II.

tutorial9.nb Flux Balance Analysis.

Notes

It is perhaps best to start with Tutorial 3. This, together with Tutorials 4 and 9 contains all the information you need to analyze networks of moderate complexity with SNA.
Tutorial 1 is intended mainly for people who wish to avoid using SNA proper and just need a computational engine for calculating elementary fluxes. However, the material in Tutorial 1 (and 2) is also used in the more advanced Tutorials 7 and 8.
The content of all of the tutorials is appended below. Since there is currently no function by function documentation for SNA, you can use this, together with Acroread's *Find*, as a workaround.

Computing elementary fluxes given the stoichiometry matrix

*HINT: Evaluating the tutorial will be a much more lively experience, if you now choose "*Delete all output*" in the* Kernel *menu.*

■ *Since we shall need to read in some data, first check that you are in the SNA tutorials directory. Executing the command*

```
Directory[]
```

```
/home/robert/SNA/tutorials
```

should give above a line with a path ending with "SNA/tutorials".

■ *If this is ok, we can now load the SNAmat package using*

```
<< "../mathcode/SNAmat.m"
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

If the package is correctly loaded the above line should look similar to LinkObject[../mathcode/../pairelvs/pairelvsIFsh, ...]

■ *We now read in a stoichiometry matrix, but first the source of the matrix is acknowledged:*

```
Import["!cat succinate.dat | grep @", "Lines"] // TableForm
```

```
@ Stoichiometry matrix of the Succinate metabolism of the E. Coli network,
@ taken from:
@            S.Klamt, J. Stelling, Mol. Biol. Rep, 29, 2002, 233-236
@
@ The first 18 columns are reversible.
@
```

```
stoichreal = Import["!cat succinate.dat | grep -v @", "Table"];
```

■ *This is a 30 by 45 matrix,*

```
stoichreal // Dimensions
```

```
{30, 45}
```

with the second row :

```
stoichreal[[2]]
```

```
{0., 0., -1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., -1., 1., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.}
```

■ *The numbers in the matrix are reals, notice the decimal point. This is unsuitable for SNAmat, so we convert* ***stoichreal*** *to rational numbers using:*

```
stoich = real2rat[stoichreal];
```

```
stoich[[2]]
```

```
{0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

■ *We can now calculate the elementary fluxes using the following command (the second argument states that the first 18 columns of* ***stoich*** *are reversible):*

```
{elfluxs, rev} = fluxelvs[stoich, 18];
```

The list ***elfluxs*** *consists of the 4250 elementary fluxes*

```
Dimensions[elfluxs]
```

```
{4250, 45}
```

The first

```
rev
```

```
0
```

of these are reversible. (So in this case none are reversible.)
The second elementary flux, for example, is

```
elfluxs[[2]]
```

$$\left\{8, 0, 0, 0, 0, 0, 0, 0, 4, 6, 0, 0, 0, 0, -4, 6, 0, \frac{40}{3}, 0, 7, 2, 0,\right.$$
$$\left. 0, 0, 0, 0, 0, 4, 0, 0, 2, 0, 4, 0, 2, 0, 4, 0, \frac{28}{3}, 0, 0, 0, 0, 4, 0\right\}$$

- *If you don't need all elementary fluxes, but just a minimal generating set of the flux cone use:*

```
{gsfluxs, rev} = fluxgset[stoich, 18];
Dimensions[gsfluxs]
```

```
{230, 45}
```

- *In the course of computation the SNA routines can generate quite a lot of informative messages. If you want to look at the messages generated so far, open the Messages notebook from the Window menu. However,to find out what all the numbers mean, you'll have to peruse the source code in SNAmat.nb. The output of informative messages can be controlled by setting the variable* ***chatter****.*

```
?chatter
```

```
With default setting, chatter = 1, the SNA routines print out all kinds
  of intermediate stuff to the Messages notebook. chatter = 0:
  shut  up, chatter = -1: output goes to the evaluation notebook.
```

Computational Geometry with SNAmat

This tutorial describes functions which give a more direct access to the computational geometry routines in SNAmat than the ones described in Tutorial 1. The presented material is rather technical and it is perhaps best to skip it, at least initially.

■ *Make sure you are in the SNA tutorials directory, before loading SNAmat.*

```
<< "../mathcode/SNAmat.m"
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *The routines described below carry out computations on a polyhedral convex cone* ***C****, given*
by two matrices ***Z*** *and* ***H*** *as the set of points* ***x*** *satisfying* $\boldsymbol{Zx = 0}$ *and* $\boldsymbol{Hx \geq 0}$.
For example, let us assume

```
Z = {}; H = {{0, 1, 1}, {1, 1, 0}, {2, 3, 1}};
```

So there are no equality but only inequality constraints.

■ *To compute a minimal generating set of* ***C****, use:*

```
{gset, rev} = ZH2gset[Z, H];
```

the result **gset** *is a list of row vectors generating the cone:*

```
gset // MatrixForm
```

$$\begin{pmatrix} 1 & -1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & -1 \end{pmatrix}$$

The first **rev** r*ows in* **gset** *are reversible. Since*

```
rev
```

```
1
```

gset[[1]] *is reversible, i.e.* **−gset[[1]]** *also lies in* ***C.***

■ *Some of the above claims are easy to verify. Calculating*

```
H.Transpose[gset] // MatrixForm
```

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix}$$

yields only non−negative values in the resulting matrix, showing that the rows of **gset** *indeed lie in* ***C*** *. Further*

```
H.gset[[1]]
```

```
{0, 0, 0}
```

so **gset[[1]]** *is indeed reversible.*

■ *As a problem with equality constraints consider*

```
Znew = Take[gset, 1]; Hnew = Drop[gset, 1];
```

```
General::spell1 : Possible spelling error: new symbol
   name "Hnew" is similar to existing symbol "Znew". More…
```

So

```
Znew // MatrixForm
```

```
(1  -1  1)
```

is the reversible row in **gset** *and the other rows go into*

```
Hnew // MatrixForm
```

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & -1 \end{pmatrix}$$

The cone given by **Znew** *and* **Hnew** *is pointed, since*

```
{gsetnew, revnew} = ZH2gset[Znew, Hnew];
revnew
```

```
0
```

and its edges are

```
gsetnew // MatrixForm
```

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

Notice that these are just the first two rows of ***H****. This is a consequence of duality and the fact that the inequality given by that last row of* ***H*** *is redundant.*

■ *You can also use SNAmat to calculate the elementary vectors of* ***C****, using:*

```
{elvs, elvsrev} = ZH2elvs[Z, H];
elvs // MatrixForm
```

$$\begin{pmatrix} 1 & -1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & -1 \\ 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

```
elvsrev
```

```
1
```

SNAsym basics

The basic data type SNAsym operates on, is called **mnet**. *The tutorial shows how to input* **mnet** *metabolic networks and some basic analysis of networks.*

■ *Make sure you are in the SNA tutorials directory, before loading SNAsym. Note that executing the following command automatically loads SNAmat as well.*

```
<< "../mathcode/SNAsym.m";
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *Now we construct an* **mnet** *starting from the list of reactions given below. The third reaction is reversible, while the others are not.*

```
reacs =
 {"x" ⇀ 2 "c",
  "x" ⇀ "e" + "d",
  "a" + "b" ⇌ "c",
  "e" ⇀ "c",
  "d" ⇀ "c",
  "e" + 2 "b" ⇀ "y",
  "d" + 2 "a" ⇀ "y"}
```

```
{x ⇀ 2 c, x ⇀ d + e, a + b ⇌ c, e ⇀ c, d ⇀ c, 2 b + e ⇀ y, 2 a + d ⇀ y}
```

In the above reactions, metabolite names are strings. This is inappropriate for SNAsym, where metabolites are more complicated constructs of the form

meta[role, compound, compartment].

role *can be one of the four symbols* **Int** , **Xt**, **Xtin**, **Xtout**. *An internal metabolite has* **Int** *, an external one* **Xt**, *and* **Xtin**, **Xtout** *denote external metabolites which are restricted to being inputs or, respectively, outputs.*

compound *is a string , typically giving the chemical species of the metabolite.*

compartment *is a string giving the cellular compartment of the metabolite, you might e.g. use* **"c"** *for cytosol and* **"m"** *for mitochondria.*

The next input line converts the string in **reacs** *to proper metabolites, making all metabolites internal and being in the same compartment ,* **"".**

```
reacs1 = reacs /. x_String → meta[Int, x, ""]
```

```
{x ⟶ 2 c, x ⟶ d + e, a + b ⇌ c, e ⟶ c, d ⟶ c, 2 b + e ⟶ y, 2 a + d ⟶ y}
```

Comparing to **reacs**, *it seems that nothing has changed. The outputs seems the same, because metabolites are pretty printed on output.*
But using **InputForm** *on the first element of* **reacs** *and* **reacs1** *reveals the difference.*

```
First[reacs] // InputForm
```

```
"x" ⇁ 2*"c"
```

```
First[reacs1] // InputForm
```

```
meta[Int, "x", ""] ⇁ 2*meta[Int, "c", ""]
```

The next thing is, to give each reaction a name. Here we shall just number them sequentially.

```
names = Range[Length[reacs]]
```

```
{1, 2, 3, 4, 5, 6, 7}
```

Finally, we can construct the network.

```
net = constructmnet[reacs1, names];
```

■ *SNAsym provides functions for inspecting metabolic nets, which we illustrate on the* **net** *just constructed.*
metabolites *lists all metabolites in the net;* **reactions** *lists the reactions;* **tags** *lists the names, and* **trpairs** *makes the association between names and reactions more explicit.*

```
metabolites@net
```

```
{a, b, c, d, e, x, y}
```

```
reactions@net
```

```
{a + b ⇌ c, d ⟶ c, 2 a + d ⟶ y, e ⟶ c, 2 b + e ⟶ y, x ⟶ 2 c, x ⟶ d + e}
```

```
tags@net
```

```
{R[3], R[5], R[7], R[4], R[6], R[1], R[2]}
```

```
trpairs@net
```

```
{{R[3], a + b ⇌ c}, {R[5], d ⟶ c}, {R[7], 2 a + d ⟶ y},
 {R[4], e ⟶ c}, {R[6], 2 b + e ⟶ y}, {R[1], x ⟶ 2 c}, {R[2], x ⟶ d + e}}
```

Note that the reactions are listed in canonical ordering and not in the order they appeared in **reacs1** *. However using the names you could reconstruct the original ordering. Further,* **R[..]** *has been wrapped around the numbers we used to name the reactions. This is so that you can safely use not just numbers but also strings or even general Mathematica expressions to name the reactions. For instance, in a large model, you might name reactions by something like* **{** mnemonic, pathway**}.**

■ *Finally we use SNAsym to actually calculate something meaningful for this simple* **net.**

First we compute the elementary flux vectors.

```
{fluxes, nrev} = symfluxelvs[net];
```

```
nrev
```

```
0
```

```
fluxes
```

```
{}
```

This somewhat disappointing result means that there are no elementary fluxes and 0 of them are reversible. This is not surprising, since **net** *has no futile cycles and all metabo·*

lites are internal.

So, next we use the function **setrole** *to change the role of* **"x"** *and* **"y"** *from* **Int** *to* **Xt**.

```
net1 = setrole[net, {meta[Int, "x", ""], meta[Int, "y", ""]}, Xt];
reactions@net1
```

$$\{a + b \rightleftharpoons c,\ d \rightharpoonup c,\ 2\,a + d \rightharpoonup y_{Xt},\ e \rightharpoonup c,\ 2\,b + e \rightharpoonup y_{Xt},\ x_{Xt} \rightharpoonup 2\,c,\ x_{Xt} \rightharpoonup d + e\}$$

Note that in the output, external metabolites have their role shown in the subscript.

Now calculating the elementary fluxes yields:

```
{fluxes, nrev} = symfluxelvs[net1];
nrev
```

```
0
```

```
fluxes
```

$$\{R[1] + R[2] - 2\,R[3] + R[6] + R[7] - 2\,Rx[x_{Xt} \rightleftharpoons 0] + 2\,Rx[y_{Xt} \rightleftharpoons 0],\ 2\,R[2] - 2\,R[3] + R[4] + R[5] + R[6] + R[7] - 2\,Rx[x_{Xt} \rightleftharpoons 0] + 2\,Rx[y_{Xt} \rightleftharpoons 0]\}$$

So **net1** *has two irreversible elementary fluxes. An elementary flux is listed by showing the tag for each reaction with the scalar pre−factor of the tag giving the flow through the reaction. Note that* **symfluxelvs** *has automatically added the appropriate exchange reactions. For simplicity the exchange reactions themselves are also their names, wrapped in* **Rx** *to distinguish them from the internal reactions. In effect the output of* **symfluxelvs** *refers to the following network:*

```
trpairs@addexchanges@net1 // TableForm
```

$R[3]$	$a + b \rightleftharpoons c$
$Rx[x_{Xt} \rightleftharpoons 0]$	$x_{Xt} \rightleftharpoons 0$
$Rx[y_{Xt} \rightleftharpoons 0]$	$y_{Xt} \rightleftharpoons 0$
$R[5]$	$d \rightharpoonup c$
$R[7]$	$2\,a + d \rightharpoonup y_{Xt}$
$R[4]$	$e \rightharpoonup c$
$R[6]$	$2\,b + e \rightharpoonup y_{Xt}$
$R[1]$	$x_{Xt} \rightharpoonup 2\,c$
$R[2]$	$x_{Xt} \rightharpoonup d + e$

Now, to illustrate **Xtin**.

```
net2 = setrole[net1, {meta[Xt, "x", ""]}, Xtin];
metabolites@net2
```

```
{a, b, c, d, e, y_Xt, x_Xtin}
```

```
{fluxes, nrev} = symfluxelvs[net2];
fluxes
```

```
{R[1] + R[2] - 2 R[3] + R[6] + R[7] + 2 Rx[y_Xt ⇌ 0] + 2 Rx[0 ⟶ x_Xtin],
 2 R[2] - 2 R[3] + R[4] + R[5] + R[6] + R[7] + 2 Rx[y_Xt ⇌ 0] + 2 Rx[0 ⟶ x_Xtin]}
```

■ *To calculate the elementary conversions of* **net1** *use:*

```
conversionelvs[net1]
```

```
{x_Xt ⟶ y_Xt}
```

```
conversionelvs[net]
```

```
{}
```

Analyzing a metabolic network of moderate complexity.

■ *Make sure you are in the SNA tutorials directory, before loading SNAsym.*

```
<< "../mathcode/SNAsym.m";
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *The following reads in the network, acknowledges the source and displays some basic information about the network.*

```
{name, humit} = << (SNApath <> "tutorials/HumanMitochondria.m");
name
```

```
Metabolic network of human cardiac mitochondria adapted from

    Vo, T.D., Greenberg, H.J., and Palsson, B.O.
    Journal of Biological Chemistry, 2004.

Only reactions involving mitochondrial metabolites are included
in the present network.
```

```
reactions@humit // Length
```

```
123
```

```
metabolites@humit
```

```
{10fthf[m], 12dgr_m[m], 2mop[m], 34hpp[m], 3hacoa[m], 3oacoa[m], 5aop[m], 5fthf[m], aacoa[m],
 acac[m], accoa[m], adp[m], aglyc3p_m[m], ahcys[m], akg[m], amet[m], amp[m], arg-L[m],
 ascb-L[m], asp-L[m], atp[m], c204coa[m], c204crn[m], c226coa[m], c226crn[m], cbp[m], cdp[m],
 cdpdag_m[m], cdpea[m], chol[m], cit[m], citr-L[m], clpn_m[m], cmp[m], co2[m], coa[m], creat[m],
 crn[m], ctp[m], dgmp[m], dgsn[m], dhap[m], dhdascb[m], dhor-S[m], dtmp[m], dump[m], dutp[m],
 facoa_m[m], fad[m], fadh2[m], fe2[m], ficytC[m], focytC[m], for[m], fum[m], gdp[m], glu-L[m],
 gly[m], glyc[m], glyc3p[m], gthox[m], gthrd[m], gtp[m], gudac[m], h[m], h2co3[m], h2o[m],
 h2o2[m], hco3[m], hmgcoa[m], icit[m], idp[m], itp[m], lys-L[m], mal-L[m], methf[m], mlthf[m],
 mmcoa-S[m], nad[m], nadh[m], nadp[m], nadph[m], nh4[m], o2[m], o2-[m], oaa[m], ocdycacoa[m],
 ocdycrn[m], odecoa[m], odecrn[m], orn[m], orot[m], pa_m[m], pc_m[m], pcreat[m], pe_m[m],
 pep[m], pg_m[m], pgp_m[m], pheme[m], pi[m], pmtcoa[m], pmtcrn[m], ppcoa[m], ppi[m], ppp9[m],
 pppg9[m], ps_m[m], pyr[m], q10[m], q10h2[m], ser-L[m], stcoa[m], stcrn[m], succ[m], succoa[m],
 thf[m], thymd[m], tyr-L[m], udp[m], 5aop[c][Xt], adp[c][Xt], akg[c][Xt], arg-L[c][Xt], asp-L[c][Xt], atp[c][Xt],
 c204crn[c][Xt], c226crn[c][Xt], cdp[c][Xt], cit[c][Xt], citr-L[c][Xt], co2[c][Xt], coa[c][Xt], crn[c][Xt], fe2[c][Xt],
 fum[c][Xt], gdp[c][Xt], glu-L[c][Xt], gly[c][Xt], glyc[c][Xt], glyc3p[c][Xt], gthrd[c][Xt], gtp[c][Xt], h[c][Xt], h2o[c][Xt],
 lys-L[c][Xt], mal-L[c][Xt], nh4[c][Xt], o2[c][Xt], ocdycrn[c][Xt], odecrn[c][Xt], orn[c][Xt], pc_m[c][Xt], pep[c][Xt],
 pheme[e][Xt], pi[c][Xt], PL[e][Xt], pmtcrn[c][Xt], pppg9[c][Xt], ps_m[c][Xt], pyr[c][Xt], stcrn[c][Xt], succ[c][Xt], udp[c][Xt]}
```

Note that mitochondrial metabolites have the superscript **"m"** *, metabolites in the cytosol have* **"c"** *, and extracellular metabolites* **"e"** *. Mitochondrial metabolites are treated as internal, whereas the others are external.*

■ *While listing all 123 reactions yields a result which is still just about readable, this is a good occasion to show how to pick parts of a network.*
The following lists all reaction involving asp−L.

```
with["asp-L", reactions@humit]
```

```
{akg[m] + asp-L[m] ⇌ glu-L[m] + oaa[m], asp-L[m] + glu-L[c][Xt] + h[c][Xt] ⇌ glu-L[m] + h[m] + asp-L[c][Xt]}
```

And this, all reactions involving asp−L or glu−L.

```
with[{"asp-L", "glu-L"}, reactions@humit] // TableForm
```

```
akg[m] + asp-L[m] ⇌ glu-L[m] + oaa[m]
akg[m] + tyr-L[m] ⇌ 34hpp[m] + glu-L[m]
glu-L[c][Xt] + h[c][Xt] ⇌ glu-L[m] + h[m]
asp-L[m] + glu-L[c][Xt] + h[c][Xt] ⇌ glu-L[m] + h[m] + asp-L[c][Xt]
```

with *does not only work for reactions, but for lists in general.*

```
with[Xt, metabolites@humit]
```

```
{5aop$_{Xt}^{c}$, adp$_{Xt}^{c}$, akg$_{Xt}^{c}$, arg-L$_{Xt}^{c}$, asp-L$_{Xt}^{c}$, atp$_{Xt}^{c}$, c204crn$_{Xt}^{c}$, c226crn$_{Xt}^{c}$,
 cdp$_{Xt}^{c}$, cit$_{Xt}^{c}$, citr-L$_{Xt}^{c}$, co2$_{Xt}^{c}$, coa$_{Xt}^{c}$, crn$_{Xt}^{c}$, fe2$_{Xt}^{c}$, fum$_{Xt}^{c}$, gdp$_{Xt}^{c}$,
 glu-L$_{Xt}^{c}$, gly$_{Xt}^{c}$, glyc$_{Xt}^{c}$, glyc3p$_{Xt}^{c}$, gthrd$_{Xt}^{c}$, gtp$_{Xt}^{c}$, h$_{Xt}^{c}$, h2o$_{Xt}^{c}$, lys-L$_{Xt}^{c}$,
 mal-L$_{Xt}^{c}$, nh4$_{Xt}^{c}$, o2$_{Xt}^{c}$, ocdycrn$_{Xt}^{c}$, odecrn$_{Xt}^{c}$, orn$_{Xt}^{c}$, pc_m$_{Xt}^{c}$, pep$_{Xt}^{c}$, pheme$_{Xt}^{e}$,
 pi$_{Xt}^{c}$, PL$_{Xt}^{e}$, pmtcrn$_{Xt}^{c}$, pppg9$_{Xt}^{c}$, ps_m$_{Xt}^{c}$, pyr$_{Xt}^{c}$, stcrn$_{Xt}^{c}$, succ$_{Xt}^{c}$, udp$_{Xt}^{c}$}
```

yields all external metabolites.

If you need the list indices of the elements and not elements themselves, **iwith** *can be used instead of* **with**. *In fact the definition of* **with** *is just:*

```
?with
```

```
SNAsym`with

with[pat_, l_List] := l⟦iwith[pat, l]⟧
```

- *Many of the cytosolic compounds can, in fact, be synthesized by the mitochondria. So there is no need to feed the mitochondria with them. Hence, we change the role of some cytosolic compounds to* **Xtout**.

```
humit1 = setrole[
   humit,
   {"adp"$_{Xt}^{"c"}$, "atp"$_{Xt}^{"c"}$, "gdp"$_{Xt}^{"c"}$, "gtp"$_{Xt}^{"c"}$, "glyc3p"$_{Xt}^{"c"}$,
    "akg"$_{Xt}^{"c"}$, "cit"$_{Xt}^{"c"}$, "citr-L"$_{Xt}^{"c"}$, "succ"$_{Xt}^{"c"}$, "mal-L"$_{Xt}^{"c"}$, "fum"$_{Xt}^{"c"}$},
   Xtout];
```

The above syntax for the metabolites is a bit more readable than **meta[...]**. *There are arcane ways of actually typing things like* $\mathbf{adp_{Xt}^{c}}$ *in Mathematica. I tend to avoid this by just pasting the output of* **metabolites**.

Further, we don't want the mitochondria to produce or consume protons.

```
humit1 = setrole[humit1, "h"$_{Xt}^{"c"}$, Int];
```

The flux cone of **humit1**

■ *First we study the flux cone of* **humit1** *by computing a minimal generating set of the cone.*

```
{fluxes, nrev} = symfluxgset[humit1];
nrev
```

```
1
```

```
fluxes // Length
```

```
1150
```

So the first one of the 1150 fluxes is reversible. Let's inspect this flux.

```
fluxes[[1]]
```

```
R[CITRtm] - R[ORNt3m] + R[ORNt4m]
```

```
with[{R["CITRtm"], R["ORNt3m"], R["ORNt4m"]}, trpairs@humit1] // TableForm
```

R[CITRtm]	$\text{citr-L}^{m} \rightleftharpoons \text{citr-L}^{c}_{\text{Xtout}}$
R[ORNt3m]	$\text{h}^{c} + \text{orn}^{m} \rightleftharpoons \text{h}^{m} + \text{orn}^{c}_{\text{Xt}}$
R[ORNt4m]	$\text{h}^{c} + \text{orn}^{m} + \text{citr-L}^{c}_{\text{Xtout}} \rightleftharpoons \text{citr-L}^{m} + \text{h}^{m} + \text{orn}^{c}_{\text{Xt}}$

The second flux is already much more complicated:

```
fluxes[[2]]
```

$$R[ACONTm] - R[ASPGLUm] + R[ASPTAm] + R[ATPS4m] + R[CO2tm] + R[CSm] - R[FUMm] + R[HCO3Em] + R[ICDHxm] - R[MDHm] + R[NADH2-u10m] + R[PCm] + R[PDHm] - R[PIt2m] + 2\,R[PYRt2m] - R[SUCCt2m] - R[SUCD1m] - R[SUCD3-u10m] - Rx[asp\text{-}L^{c}_{Xt} \rightleftharpoons 0] - Rx[co2^{c}_{Xt} \rightleftharpoons 0] + Rx[glu\text{-}L^{c}_{Xt} \rightleftharpoons 0] - 2\,Rx[pyr^{c}_{Xt} \rightleftharpoons 0] + Rx[succ^{c}_{Xtout} \rightharpoonup 0]$$

■ *Clearly, just printing* **fluxes** *does not make good bedtime reading. So we use Mathematica to analyze*

fluxes *focussing on phospholipid synthesis.*

The phospholipid reaction is:

```
with["PL", trpairs@humit1]
```

$$\{\{R[DMPL],\ 18\,clpn\_m^{m} + 43\,pc\_m^{m} + 34\,pe\_m^{m} \rightharpoonup 100\,PL^{e}_{Xt}\}\}$$

So we can pick all fluxes generating PL by

```
plfluxes = with["DMPL", fluxes];
plfluxes // Length
```

```
791
```

Note,that since the exchange reactions are part of a flux, we could also have obtained **plfluxes** *by asking for fluxes with PL.*

```
plfluxes === with["PL", fluxes]
```

```
True
```

Next, we obtain all internal reactions which cam be involved in PL synthesis by

```
somewhere = Cases[plfluxes, R[_], ∞] // Union;
somewhere // Length
```

```
89
```

```
somewhere
```

```
{R[5AOPtm], R[ACONTm], R[ADK1m], R[AGATm], R[AKGDm], R[AKGMALtm], R[ALASm],
 R[ASPGLUm], R[ASPTAm], R[ATPS4m], R[C160CPT2], R[C160CRN], R[C180CPT2],
 R[C180CRN], R[C181CRN2], R[C181CRN3], R[C182CRN2], R[C182CRN3], R[C204CRN2],
 R[C204CRN3], R[C226CRN2], R[C226CRN3], R[CATm], R[CBMKm], R[CITRtm],
 R[CITtam], R[CITtbm], R[CLPNSm], R[CO2tm], R[CRNtim], R[CSm], R[CYOOm3],
 R[CYOR-u10m], R[DAGKm], R[DASYNm], R[DMheme], R[DMPL], R[FAOXC160],
 R[FAOXC180], R[FAOXC181], R[FAOXC182], R[FAOXC204], R[FAOXC226],
 R[FASYNm], R[FCLTm], R[FE2tm], R[FUMm], R[G3PATm], R[GLYCtm], R[GLYKm],
 R[GLYtm], R[GTHOm], R[GTHPm], R[H2Otm], R[HCO3Em], R[HMGCOASm], R[HMGLm],
 R[ICDHxm], R[ICDHym], R[MALtm], R[MDHm], R[ME2m], R[NADH2-u10m],
 R[NH4tm], R[O2tm], R[OCBTm], R[OCOAT1m], R[ORNt3m], R[PAPAm], R[PCm],
 R[PCtm], R[PDHm], R[PEPCKm], R[PGPPm], R[PGSAm], R[PIt2m], R[PPAm],
 R[PPPG9tm], R[PPPGOm], R[PSDm], R[PStm], R[PYRt2m], R[Satpctp],
 R[SPODMm], R[SUCCt2m], R[SUCD1m], R[SUCD3-u10m], R[SUCOASm], R[THD1m]}
```

and compare this with the reactions which occur in every flux of **plfluxes**.

```
everywhere = Intersection @@ Map[ Cases[#, R[_], ∞] &, plfluxes];
everywhere // Length
```

```
34
```

```
with[everywhere, trpairs@humit1] // TableForm
```

R[ASPTAm]	$akg^m + asp\text{-}L^m \rightleftharpoons glu\text{-}L^m + oaa^m$
R[ADK1m]	$amp^m + atp^m \rightleftharpoons 2\, adp^m$
R[PGSAm]	$cdpdag\_m^m + glyc3p^m \rightleftharpoons cmp^m + h^m + pgp\_m^m$
R[DASYNm]	$ctp^m + h^m + pa\_m^m \rightleftharpoons cdpdag\_m^m + ppi^m$
R[CLPNSm]	$2\, pg\_m^m \rightleftharpoons clpn\_m^m + glyc^m$
R[CO2tm]	$co2_{Xt}^c \rightleftharpoons co2^m$
R[ASPGLUm]	$asp\text{-}L^m + h^c + glu\text{-}L_{Xt}^c \rightleftharpoons glu\text{-}L^m + h^m + asp\text{-}L_{Xt}^c$
R[GLYCtm]	$glyc_{Xt}^c \rightleftharpoons glyc^m$
R[H2Otm]	$h2o_{Xt}^c \rightleftharpoons h2o^m$
R[PIt2m]	$h^c + pi_{Xt}^c \rightleftharpoons h^m + pi^m$
R[Satpctp]	$atp^m + cmp^m \rightharpoonup amp^m + ctp^m$
R[C204CRN3]	$c204crn^m + coa^m \rightharpoonup c204coa^m + crn^m$
R[C226CRN3]	$c226crn^m + coa^m \rightharpoonup c226coa^m + crn^m$
R[CRNtim]	$crn^m \rightharpoonup crn_{Xt}^c$
R[AGATm]	$aglyc3p\_m^m + facoa\_m^m \rightharpoonup coa^m + pa\_m^m$
R[GLYKm]	$atp^m + glyc^m \rightharpoonup adp^m + glyc3p^m + h^m$
R[G3PATm]	$facoa\_m^m + glyc3p^m \rightharpoonup aglyc3p\_m^m + coa^m$
R[C182CRN3]	$coa^m + ocdycrn^m \rightharpoonup crn^m + ocdycacoa^m$
R[C181CRN3]	$coa^m + odecrn^m \rightharpoonup crn^m + odecoa^m$
R[DMPL]	$18\, clpn\_m^m + 43\, pc\_m^m + 34\, pe\_m^m \rightharpoonup 100\, PL_{Xt}^e$
R[PGPPm]	$h2o^m + pgp\_m^m \rightharpoonup pg\_m^m + pi^m$
R[ATPS4m]	$adp^m + 4\, h^c + pi^m \rightharpoonup atp^m + 3\, h^m + h2o^m$
R[PPAm]	$h2o^m + ppi^m \rightharpoonup h^m + 2\, pi^m$
R[PSDm]	$h^m + ps\_m^m \rightharpoonup co2^m + pe\_m^m$
R[NADH2-u10m]	$5\, h^m + nadh^m + q10^m \rightharpoonup 4\, h^c + nad^m + q10h2^m$
R[FASYNm]	$\frac{c204coa^m}{4} + \frac{c226coa^m}{20} + \frac{ocdycacoa^m}{5} + \frac{odecoa^m}{10} + \frac{pmtcoa^m}{5} + \frac{stcoa^m}{5} \rightharpoonup facoa\_$
R[C180CPT2]	$coa^m + stcrn^m \rightharpoonup crn^m + stcoa^m$
R[C204CRN2]	$c204crn_{Xt}^c \rightharpoonup c204crn^m$
R[C226CRN2]	$c226crn_{Xt}^c \rightharpoonup c226crn^m$
R[C182CRN2]	$ocdycrn_{Xt}^c \rightharpoonup ocdycrn^m$
R[C181CRN2]	$odecrn_{Xt}^c \rightharpoonup odecrn^m$
R[PCtm]	$pc\_m_{Xt}^c \rightharpoonup pc\_m^m$
R[PStm]	$ps\_m_{Xt}^c \rightharpoonup ps\_m^m$
R[C180CRN]	$stcrn_{Xt}^c \rightharpoonup stcrn^m$

■ *The irreversible reactions in* **everywhere** *are truly essential for PL−synthesis. However, for the reversible reactions in* **everywhere** *this might not be the case, since*

cancellations could occur when combining the vectors in **fluxes.**

To check this, you might want to carry out the above analysis using the elementary vectors of the flux cone of **humit1,** *starting with being a little patient until following computation terminates:*

```
{efluxes, nrev} = symfluxelvs[humit1];
nrev
```

```
General::spell1 : Possible spelling error: new symbol
   name "efluxes" is similar to existing symbol "fluxes". More…
```

```
1
```

```
efluxes // Length
```

```
34690
```

```
with["DMPL", efluxes] // Length
```

```
31410
```

■ *Sometimes it is convenient to transform symbolic representations like*

```
fluxes[[2]]
```

$$\begin{aligned}&R[ACONTm] - R[ASPGLUm] + R[ASPTAm] + R[ATPS4m] + R[CO2tm] + R[CSm] - R[FUMm] + \\ &R[HCO3Em] + R[ICDHxm] - R[MDHm] + R[NADH2\text{-}u10m] + R[PCm] + R[PDHm] - R[PIt2m] + \\ &2\,R[PYRt2m] - R[SUCCt2m] - R[SUCD1m] - R[SUCD3\text{-}u10m] - Rx[asp\text{-}L^{c}_{Xt} \rightleftharpoons 0] - \\ &Rx[co2^{c}_{Xt} \rightleftharpoons 0] + Rx[glu\text{-}L^{c}_{Xt} \rightleftharpoons 0] - 2\,Rx[pyr^{c}_{Xt} \rightleftharpoons 0] + Rx[succ^{c}_{Xtout} \rightharpoonup 0]\end{aligned}$$

into proper vectors.

This is achieved with the function **factorsof***, which extracts the pre−factors of the specified symbols, e.g.*

```
factorsof[{R["PYRt2m"], R["PPAm"], R["ASPGLUm"]}, fluxes[[2]]]
```

```
{2, 0, -1}
```

factorsof *will not only work with a single symbolic flux but also with a list of such fluxes (or more generally with a list of sums of products).*

Hence, the following extracts the flows for all internal reactions in the fluxes generating PL.

```
intcoefficientmatrix = factorsof[tags@humit1, plfluxes];
intcoefficientmatrix // Dimensions
```

```
{791, 123}
```

Note that a more numeric way of computing the set **everywhere** *could have been to use the above matrix. e.g.*

```
(tags@humit1)[[iwith[Length@plfluxes,
   Apply[Plus, Abs@Sign@intcoefficientmatrix]]]]
```

```
{R[ASPTAm], R[ADK1m], R[PGSAm], R[DASYNm], R[CLPNSm],
 R[CO2tm], R[ASPGLUm], R[GLYCtm], R[H2Otm], R[PIt2m], R[Satpctp],
 R[C204CRN3], R[C226CRN3], R[CRNtim], R[AGATm], R[GLYKm], R[G3PATm],
 R[C182CRN3], R[C181CRN3], R[DMPL], R[PGPPm], R[ATPS4m], R[PPAm],
 R[PSDm], R[NADH2-u10m], R[FASYNm], R[C180CPT2], R[C204CRN2],
 R[C226CRN2], R[C182CRN2], R[C181CRN2], R[PCtm], R[PStm], R[C180CRN]}
```

To get at flows through the exchange reactions can require a lot of typing. For convenience, the function **exch** *construct the reaction tag for a given external metabolite.*

```
exch["co2"["c"]
xt ]
```

```
Rx[co2^c_Xt ⇌ 0]
```

```
exch[{"co2"["c"]
xt , "coa"["m"], "succ"["c"]
xtout }]
```

```
{Rx[co2^c_Xt ⇌ 0], Rx[succ^c_Xtout ⇀ 0]}
```

Note, that **exch** *skips internal metabolites. So to get at the flows through the exchange reactions in* **plfluxes** *you can simply use:*

```
exchcoefficientmatrix = factorsof[exch@metabolites@humit1, plfluxes];
exchcoefficientmatrix // Dimensions
```

```
{791, 43}
```

The conversion cone of **humit1**

■ *We first a compute a minimal generating set of the conversion cone*

```
conversiongset[humit1] // Length
```

```
641
```

Since this generating set is not all that large, we go straight for the elementary conversions.

```
econvs = conversionelvs[humit1] ;
econvs // Length
```

```
1535
```

```
econvs // Last
```

$$\begin{aligned}&9629190\ \mathrm{c204crn}^{c}_{Xt} + 1925838\ \mathrm{c226crn}^{c}_{Xt} + 23685632\ \mathrm{fe2}^{c}_{Xt} + \\ &\quad 11243430\ \mathrm{gly}^{c}_{Xt} + 28887570\ \mathrm{glyc}^{c}_{Xt} + 57292212\ \mathrm{o2}^{c}_{Xt} + 7703352\ \mathrm{ocdycrn}^{c}_{Xt} + \\ &\quad 3851676\ \mathrm{odecrn}^{c}_{Xt} + 23003065\ \mathrm{pc\_m}^{c}_{Xt} + 19258380\ \mathrm{pi}^{c}_{Xt} + 23685632\ \mathrm{pppg9}^{c}_{Xt} + \\ &\quad 18188470\ \mathrm{ps\_m}^{c}_{Xt} + 11243430\ \mathrm{pyr}^{c}_{Xt} + 15800046\ \mathrm{stcrn}^{c}_{Xt} \longrightarrow \\ &\ 11243430\ \mathrm{5aop}^{c}_{Xt} + 40675330\ \mathrm{co2}^{c}_{Xt} + 38910102\ \mathrm{crn}^{c}_{Xt} + \\ &\quad 131007666\ \mathrm{h2o}^{c}_{Xt} + 23685632\ \mathrm{pheme}^{e}_{Xt} + 53495500\ \mathrm{PL}^{e}_{Xt}\end{aligned}$$

■ *Similarly to the above analysis, we now ask which external metabolites can /must occur as inputs in PL−synthesis.*

```
pleconvs = with["PL", econvs];
pleconvs // Length
```

```
1155
```

```
pleconvs // Last
```

$$9629190\ c204crn^{c}_{Xt} + 1925838\ c226crn^{c}_{Xt} + 23685632\ fe2^{c}_{Xt} + 11243430\ gly^{c}_{Xt} + 28887570\ glyc^{c}_{Xt} + 57292212\ o2^{c}_{Xt} + 7703352\ ocdycrn^{c}_{Xt} + 3851676\ odecrn^{c}_{Xt} + 23003065\ pc\_m^{c}_{Xt} + 19258380\ pi^{c}_{Xt} + 23685632\ pppg9^{c}_{Xt} + 18188470\ ps\_m^{c}_{Xt} + 11243430\ pyr^{c}_{Xt} + 15800046\ stcrn^{c}_{Xt} \rightharpoondown 11243430\ 5aop^{c}_{Xt} + 40675330\ co2^{c}_{Xt} + 38910102\ crn^{c}_{Xt} + 131007666\ h2o^{c}_{Xt} + 23685632\ pheme^{e}_{Xt} + 53495500\ PL^{e}_{Xt}$$

The first thing is, to get rid of the outputs and of the stoichiometric factors.

```
tmp = pleconvs /. a_ ⇀ b_ → a;
tmp // Last
```

$$9629190\ c204crn^{c}_{Xt} + 1925838\ c226crn^{c}_{Xt} + 23685632\ fe2^{c}_{Xt} + 11243430\ gly^{c}_{Xt} + 28887570\ glyc^{c}_{Xt} + 57292212\ o2^{c}_{Xt} + 7703352\ ocdycrn^{c}_{Xt} + 3851676\ odecrn^{c}_{Xt} + 23003065\ pc\_m^{c}_{Xt} + 19258380\ pi^{c}_{Xt} + 23685632\ pppg9^{c}_{Xt} + 18188470\ ps\_m^{c}_{Xt} + 11243430\ pyr^{c}_{Xt} + 15800046\ stcrn^{c}_{Xt}$$

```
plins = Map[Cases[#, meta[__], ∞] &, tmp];
plins // Last
```

$$\{c204crn^{c}_{Xt}, c226crn^{c}_{Xt}, fe2^{c}_{Xt}, gly^{c}_{Xt}, glyc^{c}_{Xt}, o2^{c}_{Xt}, ocdycrn^{c}_{Xt}, odecrn^{c}_{Xt}, pc\_m^{c}_{Xt}, pi^{c}_{Xt}, pppg9^{c}_{Xt}, ps\_m^{c}_{Xt}, pyr^{c}_{Xt}, stcrn^{c}_{Xt}\}$$

And now, we proceed as for the fluxes.

```
insomewhere = Cases[plins, meta[__], ∞] // Union
```

$$\{asp\text{-}L^{c}_{Xt}, c204crn^{c}_{Xt}, c226crn^{c}_{Xt}, fe2^{c}_{Xt}, glu\text{-}L^{c}_{Xt}, gly^{c}_{Xt}, glyc^{c}_{Xt}, nh4^{c}_{Xt}, o2^{c}_{Xt}, ocdycrn^{c}_{Xt}, odecrn^{c}_{Xt}, orn^{c}_{Xt}, pc\_m^{c}_{Xt}, pi^{c}_{Xt}, pmtcrn^{c}_{Xt}, pppg9^{c}_{Xt}, ps\_m^{c}_{Xt}, pyr^{c}_{Xt}, stcrn^{c}_{Xt}\}$$

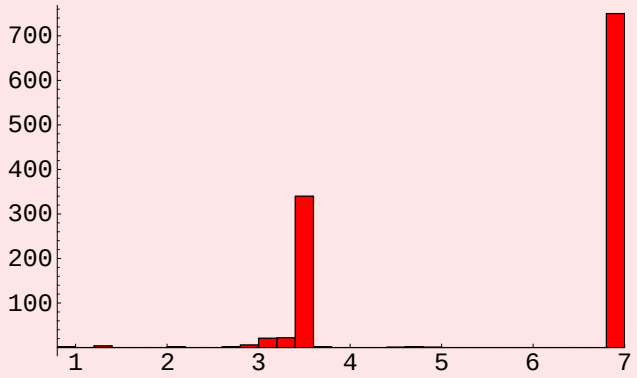
```
ineverywhere = Intersection @@ Map[ Cases[#, meta[__], ∞] &, plins]
```

```
{c204crn$_{Xt}^{c}$, c226crn$_{Xt}^{c}$, glyc$_{Xt}^{c}$,
 ocdycrn$_{Xt}^{c}$, odecrn$_{Xt}^{c}$, pc_m$_{Xt}^{c}$, pi$_{Xt}^{c}$, ps_m$_{Xt}^{c}$, stcrn$_{Xt}^{c}$}
```

Since we have considered the elementary conversions, the metabolites in **ineverywhere** *are essential for PL−synthesis. Of course we could , hopefully, have arrived at the same result by considering the exchange reactions of the elementary fluxes. But then we'd have to deal with ≥30000 fluxes.*

Now we know that $\mathbf{stcrn_{Xt}^{c}}$ *is essential, we can ask what the production rates of Pl can be.*

```
stcrnfactors = factorsof[ "stcrn"^"c"_Xt, pleconvs /. a_ ⇀ b_ → a ];
plefactors   = factorsof["PL"^"e"_Xt, pleconvs /. a_ ⇀ b_ → b ];
<< Graphics'Graphics'
Histogram[plefactors / stcrnfactors]
```




```
- Graphics -
```

Feasible and infeasible reactions

Before calculating elementary vectors , the above routines simplify the network. Among other things, this means that they determine which reactions are feasible. i.e. can actually have a non−zero flow in steady state given the current assignment of roles. Since this can be of independent interest, SNA provides the function **feasiblemnet**, *which returns the subnetwork consisting of the feasible reactions.*

```
reactions@feasiblemnet[humit] // Length
```

```
94
```

```
reactions@feasiblemnet[humit1] // Length
```

```
90
```

So of the total 123 reactions, 94 are feasible if any non mitochondrial metabolite is treated as external. But the more restrictive assignment of roles used for **humit1** *reduces this to 90.*

Importing external formats I.

This tutorial shows how to construct an **mnet**, *if you have a stoichiometry matrix (and some other information) .*

```
<< "../mathcode/SNAsym.m";
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *The data for our network is in CSV−format in the subdirectory* humit.

We first read in the stoichiometry matrix, note that the "54rev " in the filename means that the first 54 columns are reversible.

```
stoich = Import[SNApath <> "tutorials/humit/stoich54rev.csv", "CSV"];
Dimensions[stoich]
```

```
{164, 123}
```

```
stoich[[1]]
```

```
{0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.}
```

Since the entries of stoich are real numbers, we convert them to rational numbers.

```
stoich = real2rat[stoich];
stoich[[1]]
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

■ *Next, we read in some auxiliary information regarding metabolite & reaction names.*

```
metaspec =
 Import[SNApath <> "tutorials/humit/metabolites_species.csv", "CSV"]
```

```
{{10fthf, 12dgr_m, 2mop, 34hpp, 3hacoa, 3oacoa, 5aop, 5fthf, aacoa, acac,
  accoa, adp, aglyc3p_m, ahcys, akg, amet, amp, arg-L, ascb-L, asp-L,
  atp, c204coa, c204crn, c226coa, c226crn, cbp, cdp, cdpdag_m, cdpea,
  chol, cit, citr-L, clpn_m, cmp, co2, coa, creat, crn, ctp, dgmp, dgsn,
  dhap, dhdascb, dhor-S, dtmp, dump, dutp, facoa_m, fad, fadh2, fe2,
  ficytC, focytC, for, fum, gdp, glu-L, gly, glyc, glyc3p, gthox, gthrd,
  gtp, gudac, h, h2co3, h2o, h2o2, hco3, hmgcoa, icit, idp, itp, lys-L,
  mal-L, methf, mlthf, mmcoa-S, nad, nadh, nadp, nadph, nh4, o2, o2-,
  oaa, ocdycacoa, ocdycrn, odecoa, odecrn, orn, orot, pa_m, pc_m, pcreat,
  pe_m, pep, pg_m, pgp_m, pheme, pi, pmtcoa, pmtcrn, ppcoa, ppi, ppp9,
  pppg9, ps_m, pyr, q10, q10h2, ser-L, stcoa, stcrn, succ, succoa, thf,
  thymd, tyr-L, udp, 5aop, adp, akg, arg-L, asp-L, atp, c204crn, c226crn,
  cdp, cit, citr-L, co2, coa, crn, fe2, fum, gdp, glu-L, gly, glyc,
  glyc3p, gthrd, gtp, h, h2o, lys-L, mal-L, nh4, o2, ocdycrn, odecrn, orn,
  pc_m, pep, pheme, pi, PL, pmtcrn, pppg9, ps_m, pyr, stcrn, succ, udp}}
```

Mathematica reads in CSV as a matrix. We just want a simple list, so we replace the above with:

```
metaspec =
 Import[SNApath <> "tutorials/humit/metabolites_species.csv", "CSV"][[1]]
```

```
{10fthf, 12dgr_m, 2mop, 34hpp, 3hacoa, 3oacoa, 5aop, 5fthf, aacoa, acac,
 accoa, adp, aglyc3p_m, ahcys, akg, amet, amp, arg-L, ascb-L, asp-L,
 atp, c204coa, c204crn, c226coa, c226crn, cbp, cdp, cdpdag_m, cdpea,
 chol, cit, citr-L, clpn_m, cmp, co2, coa, creat, crn, ctp, dgmp, dgsn,
 dhap, dhdascb, dhor-S, dtmp, dump, dutp, facoa_m, fad, fadh2, fe2,
 ficytC, focytC, for, fum, gdp, glu-L, gly, glyc, glyc3p, gthox, gthrd,
 gtp, gudac, h, h2co3, h2o, h2o2, hco3, hmgcoa, icit, idp, itp, lys-L,
 mal-L, methf, mlthf, mmcoa-S, nad, nadh, nadp, nadph, nh4, o2, o2-,
 oaa, ocdycacoa, ocdycrn, odecoa, odecrn, orn, orot, pa_m, pc_m, pcreat,
 pe_m, pep, pg_m, pgp_m, pheme, pi, pmtcoa, pmtcrn, ppcoa, ppi, ppp9,
 pppg9, ps_m, pyr, q10, q10h2, ser-L, stcoa, stcrn, succ, succoa, thf,
 thymd, tyr-L, udp, 5aop, adp, akg, arg-L, asp-L, atp, c204crn, c226crn,
 cdp, cit, citr-L, co2, coa, crn, fe2, fum, gdp, glu-L, gly, glyc,
 glyc3p, gthrd, gtp, h, h2o, lys-L, mal-L, nh4, o2, ocdycrn, odecrn, orn,
 pc_m, pep, pheme, pi, PL, pmtcrn, pppg9, ps_m, pyr, stcrn, succ, udp}
```

```
metacomp = Import[
   SNApath <> "tutorials/humit/metabolites_compartments.csv", "CSV"][[1]]
```

```
{m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m,
 m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m,
 m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m,
 m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m, m,
 m, m, m, m, m, m, m, m, m, c, c, c, c, c, c, c, c, c, c, c, c, c, c, c, c,
 c, c, c, c, c, c, c, c, c, c, c, c, c, c, c, c, c, e, c, e, c, c, c, c, c, c}
```

```
rnames = Import[SNApath <> "tutorials/humit/reactionnames.csv", "CSV"][[1]]
```

```
General::spell1 : Possible spelling error: new symbol
   name "rnames" is similar to existing symbol "names". More…
```

```
{ASPTAm, ADK1m, ETHAPTm, ACONTm, CITRtm, CK, GLYAMDTR, PGSAm, GACMTR,
 H2CO3Dm, HCO3Em, FUMm, GTHPm, HMGCOASm, ADK4m, HACDm, MDHm, DHORDm,
 DASYNm, CLPNSm, SUCD3_u10m, SUCD1m, SUCOASm, OCOAT1m, FTHFLm, GHMT2rm,
 TYRTAm, 5AOPtm, ARGtm, DNC1C, CITtam, CITtbm, CO2tm, COAtm, DNC1G, GLYtm,
 GLYCtm, THD1m, ORNt3m, ORNt4m, GLUt2m, ASPGLUm, H2Otm, GTHRDt, LYStm,
 AKGMALtm, MALtm, NH4tm, O2tm, PIt2m, PPPG9tm, PYRt2m, SUCCt2m, DNC1U,
 ACACT1m, DAGKm, Satpctp, C204CRN3, C226CRN3, CRNtim, DGNSKm, AGATm,
 GLYKm, G3PATm, G3PDm, GTHDHm, DUTPDPm, CATm, HMGLm, THFATm, MMSAD1m,
 AKGDm, ICDHxm, FAOXC204, FAOXC226, ICDHym, ME2m, GTHOm, CBMKm, CYOOm3,
 SPODMm, PEPCKm, CSm, FAOXC182, C182CRN3, FAOXC181, C181CRN3, OCBTm,
 PAPAm, PCHOLPm, DMPL, PGPPm, DMheme, FAOXC160, C160CPT2, PPCOACm, PPAm,
 FCLTm, PPPGOm, PSDm, PCm, PDHm, NADH2_u10m, CYOR_u10m, FASYNm, FAOXC180,
 C180CPT2, ALASm, TMDK1m, ATPtm, C204CRN2, C226CRN2, FRDcm, GTPtm,
 GLYC3Ptm, FE2tm, ATPS4m, C182CRN2, C181CRN2, PCtm, C160CRN, PStm, C180CRN}
```

■ *From* **metaspec** *and* **metacomp** *we construct the metabolites of the network by:*

```
metas = MapThread[meta[Int, #1, #2] &, {metaspec, metacomp}]
```

```
{10fthf^m, 12dgr_m^m, 2mop^m, 34hpp^m, 3hacoa^m, 3oacoa^m, 5aop^m, 5fthf^m, aacoa^m,
 acac^m, accoa^m, adp^m, aglyc3p_m^m, ahcys^m, akg^m, amet^m, amp^m, arg-L^m,
 ascb-L^m, asp-L^m, atp^m, c204coa^m, c204crn^m, c226coa^m, c226crn^m, cbp^m,
 cdp^m, cdpdag_m^m, cdpea^m, chol^m, cit^m, citr-L^m, clpn_m^m, cmp^m, co2^m,
 coa^m, creat^m, crn^m, ctp^m, dgmp^m, dgsn^m, dhap^m, dhdascb^m, dhor-S^m, dtmp^m,
 dump^m, dutp^m, facoa_m^m, fad^m, fadh2^m, fe2^m, ficytC^m, focytC^m, for^m, fum^m,
 gdp^m, glu-L^m, gly^m, glyc^m, glyc3p^m, gthox^m, gthrd^m, gtp^m, gudac^m, h^m,
 h2co3^m, h2o^m, h2o2^m, hco3^m, hmgcoa^m, icit^m, idp^m, itp^m, lys-L^m, mal-L^m,
 methf^m, mlthf^m, mmcoa-S^m, nad^m, nadh^m, nadp^m, nadph^m, nh4^m, o2^m, o2-^m,
 oaa^m, ocdycacoa^m, ocdycrn^m, odecoa^m, odecrn^m, orn^m, orot^m, pa_m^m, pc_m^m,
 pcreat^m, pe_m^m, pep^m, pg_m^m, pgp_m^m, pheme^m, pi^m, pmtcoa^m, pmtcrn^m,
 ppcoa^m, ppi^m, ppp9^m, pppg9^m, ps_m^m, pyr^m, q10^m, q10h2^m, ser-L^m, stcoa^m,
 stcrn^m, succ^m, succoa^m, thf^m, thymd^m, tyr-L^m, udp^m, 5aop^c, adp^c, akg^c,
 arg-L^c, asp-L^c, atp^c, c204crn^c, c226crn^c, cdp^c, cit^c, citr-L^c, co2^c, coa^c,
 crn^c, fe2^c, fum^c, gdp^c, glu-L^c, gly^c, glyc^c, glyc3p^c, gthrd^c, gtp^c, h^c,
 h2o^c, lys-L^c, mal-L^c, nh4^c, o2^c, ocdycrn^c, odecrn^c, orn^c, pc_m^c, pep^c,
 pheme^e, pi^c, PL^e, pmtcrn^c, pppg9^c, ps_m^c, pyr^c, stcrn^c, succ^c, udp^c}
```

Now we construct the reactions, note that 54 is the number of reversible reactions.

```
reacs = stoich2reacs[stoich, metas, 54];
reacs // First
```

```
akg^m + asp-L^m ⇌ glu-L^m + oaa^m
```

```
reacs // Last
```

```
stcrn^c ⇀ stcrn^m
```

Using the reactions and their names, we first construct the network

```
newhumit = constructmnet[reacs, rnames];
```

and then change the role of all extramitochondrial metabolites to **Xt***.*

```
newhumit = setrole[newhumit, with[{"c", "e"}, metabolites@newhumit], Xt];
```

Now **newhumit** *should be just the network we used in Tutorial 4.*
To check if this is indeed the case:

```
{name, humit} = << (SNApath <> "tutorials/HumanMitochondria.m");
newhumit === humit
```

```
True
```

Importing external formats II.

This tutorial shows how to construct an **mnet**, *given a textual representation of the reaction system.*

```
<< "../mathcode/SNAsym.m";
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *We first acknowledge and then inspect the data in* humit.csv.

```
Import["!grep \"#\" humit.csv", "Lines"] // TableForm
```

```
#
#  Metabolic network of human cardiac mitochondria
#
#  Vo, T.D., Greenberg, H.J., and Palsson, B.O.
#  Journal of Biological Chemistry, 2004.
#
#  This file was generated by converting the reaction sheet of
#
#    http://gcrg.ucsd.edu/organisms/mitochondria/Supplemental_data.xls
#
#  to CSV format, and replacing the header.
#
```

```
rawtext = Import["!grep -v \"#\" humit.csv", "CSV"];
rawtext // First // InputForm
```

```
{1, "HEX1", "hexokinase (D-glucose:ATP)", "[c] : atp + glc-D --> adp + g6p
+ h", 4, "Glycolysis ",
 "2.7.1.1", "yes", "(Voet, Voet et al. 1999)"}
```

The second column of **rawtext** *has the reaction name and the fourth column has the reaction.*

The following sample shows the reaction syntax.

```
indexes = {1, 10, 21, 26};
sample = Transpose[rawtext][[4]][[indexes]]; sample // TableForm
```

```
[c] : atp + glc-D --> adp + g6p + h
[c] : 2pg <==> 3pg
[m] : fum + h2o <==> mal-L
(4) focytC[m] + (7.92) h[m] + o2[m] --> (4) ficytC[m] + (4) h[c] + (1.96) h2
```

If all in involved metabolites are within one compartment, the compartment is noted as the head of the reaction but not in the names of the metabolites.

If, as in the last reaction, metabolites from different compartments are involved, the compartments are noted as part of the metabolite name.

- *Our goal will be to let Mathematica do the parsing of this syntax. The problem is that the above reactions are not Mathematica expressions, e.g. some metabolite names start with a number and thus are not valid symbols in Mathematica.*
  However, we can use substitutions based on regular expression to 'beautify' the reactions and make them conform to Mathematica syntax. From Mathematica 5.1, this could be done within Mathematica. Here we shall use the Unix utility sed *to do the beautifying.*
  The following preprocesses humit.csv *and inspects the result.*

```
text = Import["!grep -v \"#\" humit.csv | sed -f humitsedscript", "CSV"];
text // First // InputForm
```

```
General::spell : Possible spelling error: new symbol
   name "text" is similar to existing symbols {ext, Text}. More…
```

```
{1, "HEX1", "hexokinase (DDashglucose:ATP)", "c: atp + glcDashD > adp +
g6p + h", 4,
 "Glycolysis ", "2.7.1.1", "yes", "(Voet, Voet et al. 1999)"}
```

```
sample = Transpose[text][[4]][[indexes]]; sample // TableForm
```

```
c: atp + glcDashD > adp + g6p + h
c: NuM2pg == NuM3pg
m: fum + h2o == malDashL
(4) focytC[m] + (7.92) h[m] + o2[m] > (4) ficytC[m] + (4) h[c] + (1.96) h2o[
```

The syntax of the transformed reactions now conforms to Mathematica. The following shows what happens

when Mathematica parses the syntax.

```
Map[FullForm[ToExpression[#]] &, sample] // TableForm
```

```
Pattern[c, Greater[Plus[atp, glcDashD], Plus[adp, g6p, h]]]
Pattern[c, Equal[NuM2pg, NuM3pg]]
Pattern[m, Equal[Plus[fum, h2o], malDashL]]
Greater[Plus[Times[4, focytC[m]], Times[7.92', h[m]], o2[m]], Plus[Times[4, fic
```

■ *While Mathematica can make sense of the syntax, parsing does not result in expressions which are valid SNAsym reactions.*

So we shall use the following function which further transforms the result of **ToExpression***:*

```
toreaction[s_] := Block[ {r, c, a, b},
   r = ToExpression[s];
   If[Head[r] == Pattern,
    c = r [[1]];
    r = r [[2]];
    r = Replace[r , a_Symbol → a[c], {-1}]];
   r /. a_[b_] → meta[Int, a, b] /. (a_ > b_) → (a ⇁ b) /.
    (a_ == b_) → (a ⇌ b)];
```

We now apply **toreaction** *to the fourth column of* **text** *and inspect the result.*

```
reacs1 = Map[toreaction[#[[4]]] &, text];
reacs1[[indexes]] // TableForm
```

```
meta[Int, atp, c] + meta[Int, glcDashD, c] ⇁ meta[Int, adp, c] + meta[Int, g6p, c]
meta[Int, NuM2pg, c] ⇌ meta[Int, NuM3pg, c]
meta[Int, fum, m] + meta[Int, h2o, m] ⇌ meta[Int, malDashL, m]
4 meta[Int, focytC, m] + 7.92 meta[Int, h, m] + meta[Int, o2, m] ⇁ 4 meta[Int, ficy
```

This is already quite good, since the reaction syntax is now what we want. However, the metabolite syntax needs some more work, because the compounds and compartments are now symbols and not strings.

So we transform them back to strings and, while we are at it, we might as well do something about the fact that beautified compound names such as **glcDashD** *are less than beautiful.*

```
reps = {
   "NuM" → "",
   "Dash" → "-",
   "Sub" -> "_",
   "Prime" → "'"};
sym2string[sym_] := Fold[StringReplace, ToString[sym], reps];
reacs2 =
  reacs1 /. meta[a_, b_, c_] :> meta[a, sym2string@b, sym2string@c];
reacs2[[indexes]] // TableForm
```

```
General::spell : Possible spelling error: new symbol
   name "reps" is similar to existing symbols {rep, res}. More…
```

$$\text{atp}^{c} + \text{glc-D}^{c} \rightharpoonup \text{adp}^{c} + \text{g6p}^{c} + \text{h}^{c}$$

$$2\text{pg}^{c} \rightleftharpoons 3\text{pg}^{c}$$

$$\text{fum}^{m} + \text{h2o}^{m} \rightleftharpoons \text{mal-L}^{m}$$

$$4\ \text{focytC}^{m} + 7.92\ \text{h}^{m} + \text{o2}^{m} \rightharpoonup 4\ \text{ficytC}^{m} + 4\ \text{h}^{c} + 1.96\ \text{h2o}^{m} + 0.02\ \text{o2-}^{m}$$

We are nearly done with the reactions. The last thing is to transform any real stoichiometric factors to rartionals.

```
reacs3 = reacs2 /. a_ meta[x_, y_, z_] :> real2rat[a] meta[x, y, z];
reacs3[[indexes]] // TableForm
```

$$\text{atp}^{c} + \text{glc-D}^{c} \rightharpoonup \text{adp}^{c} + \text{g6p}^{c} + \text{h}^{c}$$

$$2\text{pg}^{c} \rightleftharpoons 3\text{pg}^{c}$$

$$\text{fum}^{m} + \text{h2o}^{m} \rightleftharpoons \text{mal-L}^{m}$$

$$4\ \text{focytC}^{m} + \frac{198\ \text{h}^{m}}{25} + \text{o2}^{m} \rightharpoonup 4\ \text{ficytC}^{m} + 4\ \text{h}^{c} + \frac{49\ \text{h2o}^{m}}{25} + \frac{\text{o2-}^{m}}{50}$$

■ *Now we are truly done with the reactions. And it seems that all we still need is to get the reaction names*

```
rnames = Map[
  Composition[sym2string, First, Rest], text]
```

```
General::spell1 : Possible spelling error: new symbol
   name "rnames" is similar to existing symbol "names". More…
```

```
{HEX1, G6PI, G6PI2, PGI, PFK, FBA, TPI, GAPD, PGK, PGM, ENO, PYK, PDHm, CSm,
 ACONTm, ICDHxm, ICDHym, AKGDm, SUCOASm, SUCD1m, FUMm, MDHm, NADH2-u10m,
 SUCD3-u10m, CYOR-u10m, CYOOm3, ATPS4m, GLUCYS, GTHDHm, GTHOm, GTHPm,
 GTHRDt, GTHS, SPODMm, THD1m, CATm, ASPGLUm, ASPTA, MDH, AKGMALtm, ASPTAm,
 ALASm, 5AOPtm, PPBNGS, HMBS, UPP3S, UPPDC1, CPPPGO, PPPG9tm, PPPGOm,
 FCLTm, FAOXC160, FAOXC180, FAOXC181, FAOXC182, FAOXC204, FAOXC226, C160,
 C160CPT1, C160CPT2, C160CRN, C180, C180CPT1, C180CPT2, C180CRN, C181,
 C181CRN1, C181CRN2, C181CRN3, C182, C182CRN1, C182CRN2, C182CRN3, C204,
 C204CRN1, C204CRN2, C204CRN3, C226, C226CRN1, C226CRN2, C226CRN3, CRNtim,
 CHLPCTD, DAGCPTm, PCtm, PSDm, PStm, G3PDm, CLPNSm, PAPAm, DAGKm, ETHAPTm,
 HMGLm, OCOAT1m, ACACT1m, MMSAD1m, PPCOACm, HACDm, GLYKm, FASYNm, G3PATm,
 AGATm, DASYNm, PCHOLPm, PGSAm, PGPPm, NH4tm, UREAt, CBMKm, OCBTm, ARGSS,
 ARGSL, ARGN, ORNt4m, GLYAMDTR, GACMTR, CK, ADK1, ADK1m, ADK4m, DGNSKm,
 DHORDm, DUTPDPm, NDPK1, TMDK1m, GLUDx, GLUDy, TYRTAm, LDH_L, ME2m, PCm,
 PEPCKm, FTHFLm, GHMT2rm, HMGCOASm, THFATm, H2CO3Dm, H2OD, HCO3Em, PPA,
 PPAm, ARGtm, ATPtm, CITRtm, CITtam, CITtbm, CO2tm, COAtm, DNC1C, DNC1G,
 DNC1U, FE2tm, FRDcm, GLUt2m, GLYC3Ptm, GLYCtm, GLYtm, GTPtm, H2Otm, Htm,
 LYStm, MALtm, O2tm, ORNt3m, PIt2m, PYRt2m, SUCCt2m, 12DGRt1, C204t, C226t,
 CO2t, COAt, CYSt2r, FE2t1, GLCt1, GLUt2r, GLYC3Pt1, GLYCt1, GLYt2r,
 H2Ot, HDCAt, Ht, L-LACt2r, O2t, OCDCAt, OCDCEAt, OCDCYAt, PIt2r, PSt}
```

and use **constructmnet** *to build the network.*

Unfortunately some reactions, which are necessary for the proper functioning of the network , are not listed in the reactions sheet of the original .xls file, but on the constraint sheet.

So we add these by hand and then call **constructmnet** .

```
more =
 { {"DMatp",  "h2o"["c"] + "atp"["c"] → "adp"["c"] + "h"["c"] + "pi"["c"]},
  {"DMheme", "pheme"["m"] → "pheme"["e"]},
  {"DMPL",    18 "clpn_m"["m"] + 43 "pc_m"["m"] + 34 "pe_m"["m"] → 100 "PL"["e"]},
  {"Satpctp", "atp"["m"] + "cmp"["m"] → "amp"["m"] + "ctp"["m"]}};
reacs3 = Join[reacs3, Last[Transpose[more]]];
rnames = Join[rnames, First[Transpose[more]]];
humitall = constructmnet[reacs3, rnames];
```

For Tutorial 4, we do not want all of the reactions but only those involving at least one mitochondrial metabolite.

submnetwith *extracts the appropriate subnetwork and, finally, we change the role of all non−mitochondrial metabolites to* **Xt***.*

```
mithumit = submnetwith[ with["m", metabolites@humitall], humitall];
mithumit1 = setrole[mithumit ,  with[{"c", "e"}, metabolites@mithumit ], Xt];
```

Now mithumit1 should be the network used in Tutorial 4. Let's check

```
{name, humit} = << HumanMitochondria.m;
mithumit1 === humit
```

```
True
```

Analyzing larger metabolic networks

```
<< "../mathcode/SNAsym.m";
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *The network*

```
{name, ecoli} = << (SNApath <> "tutorials/ecoli.m");
name
```

```
Central carbon metabolism of Ecoli.
Adapted from: Stelling et. al., Nature,420 (2002),190-193.
```

```
reactions@ecoli // Length
```

```
99
```

```
metabolites@ecoli
```

```
{2PG^c, 3PG^c, AcCoA^c, AcP^c, Adh^c, ADPGlc^c, Ala^c, alKG^c, Arg^c, Asn^c, Asp^c,
 AspSAld^c, C14_0_FS^c, CDPEth^c, Chor^c, Cit^c, CMP_KDO^c, Cys^c, dATP^c, dCTP^c,
 dGTP^c, DHAP^c, di_am_pim^c, DPG^c, dTTP^c, E4P^c, F16P^c, F6P^c, Fum^c, G3P^c, G6P^c,
 Gln^c, Glu^c, Gly^c, Glyc3P^c, Glyox^c, H_ex^c, His^c, HSer^c, ICit^c, Ile^c, Leu^c,
 Lys^c, Mal^c, Met^c, mit_FS^c, MTHF^c, NADH^c, NADPH^c, NDPHep^c, OH_myr_ac^c,
 OxA^c, PEP^c, PGlac^c, PGluc^c, Phe^c, Pro^c, PRPP^c, Pyr^c, QuiH2^c, R5P^c, rATP^c,
 rCTP^c, rGTP^c, Rl5P^c, rUTP^c, S^c, S7P^c, Ser^c, SuccCoA^c, TDPGlcs^c, Thr^c,
 Trp^c, Tyr^c, UDPGlc^c, UDP_NAG^c, UDP_NAM^c, Val^c, X5P^c, Ac_Xt^x, CO2_Xt^x, Glyc_Xtin^x,
 N_Xtin^x, O2_Xtin^x, Succ_Xtin^x, ATP_Xtout^x, Biomass_Xtout^x, Eth_Xtout^x, Form_Xtout^x, Lac_Xtout^x}
```

In contrast to the 110 reactions in the above reference, our network has only 99 reactions since the exchange reactions for the 10 external metabolites are not included here. Further , in the reference, $\mathbf{Ac^x_{Xt}}$ *, has two irreversible exchange reactions instead of a single reversible one. 99+10+1 = 110.*

■ *Judging by the number of reactions, this network has about the same size as the one used in Tutorial 4. Indeed, due to the relatively few external metabolites the conversion cone is not more complicated and can be dealt with by the same techniques as in Tutorial 4. So we shall only consider the flux cone, which is more*

complicated, probably due the fact that the E. Coli network has some lumped reactions and in effect correspond to a larger network.

The output of the function **symfluxelvs** *may use up a lot memory and we shall discuss how this can be controlled.*

symfluxelvs *first adds the exchange reactions, then simplifies the network , and obtains the stoichiometry matrix, similarly to the following two commands*

```
secoli = fluxsimp[addexchanges[ecoli]]; {stoich, nrev} = mnet2stoich[secoli];
```

```
General::spell1 : Possible spelling error: new symbol
   name "secoli" is similar to existing symbol "ecoli". More…
```

Then **symfluxelvs** *calls the SNAmat routine* **fluxelvs** *to compute the elementary fluxes of the reduced network and, if you really need the elementary fluxes, this cannot be avoided. For* **secoli** *the calculation is doable even on a moderately sized machine but may take 5 to 10 minutes to complete. Hence, I suggest that you wait till your coffee break before evaluating the next two commands and, for the moment, proceed to the next section.*

```
AbsoluteTiming[ {fluxese, reve} = fluxelvs[stoich, nrev];]
```

```
{436.802760 Second, Null}
```

```
fluxese // Length
```

```
507631
```

■ *We shall make do with a minimal generating set for the flux cone of* **secoli.**

```
 {fluxesg, revg} = fluxgset[stoich, nrev];
fluxesg // Dimensions
```

```
General::spell1 : Possible spelling error: new symbol
   name "fluxesg" is similar to existing symbol "fluxese". More…

General::spell1 : Possible spelling error: new symbol
   name "revg" is similar to existing symbol "reve". More…
```

```
{1427, 46}
```

```
fluxesg // First
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0}
```

We now have the flux vectors for the simplified network and would like to relate them to the ones of the full network. For this we need the tags of **secoli**, *since they keep track of the relationship. For instance,*

```
with["Eth", trpairs@secoli] // TableForm
```

R[AcCoA::Adh] + R[Adh::Eth] + Rx[$\text{Eth}^{x}_{\text{Xtout}} \rightharpoonup 0$] $\text{AcCoA}^{c} + 2\,\text{NADH}^{c} \rightharpoonup 0$

shows that **fluxsimp** *has determined that in steady state the three reactions* **R[AcCoA::-Adh]** , **R[Adh::Eth]** *and* **Rx[$\text{Eth}^{x}_{\text{Xtout}} \rightharpoonup 0$]** *must operate in conjunction and effectively become* **$\text{AcCoA}^{c} + 2\,\text{NADH}^{c} \rightharpoonup 0$**.

Sometimes **fluxsimp** *can make more ambitious simplification marked by the symbol* **ALT.** *These cases will be considered at the end of Tutorial 8, here we just check that we don't have to worry about* **ALT**.

```
with[ALT, tags@secoli]
```

```
{}
```

Since we have the fluxes of **secoli** *and the relationship between the reactions of* **secoli** *and* **ecoli** *given by* **tags@secoli** , *it is very easy to obtain the fluxes of the full network using a matrix product.*

```
ecolisymfluxesg = fluxesg.(tags@secoli) // Expand;
ecolisymfluxesg // First
```

```
R[Fum::Succ] + R[Succ::Fum]
```

ecolisymfluxesg *is essentially what we'd have obtained by just using* **symfluxgset[ecoli],** *except that, in addition,* **symfluxgset** *normalizes the result:*

```
symfluxgset[ecoli] // First // First
```

```
R[Fum::Succ] + R[Succ::Fum]
```

But , using **fluxesg** *and* **tags@secoli,** *we can make statements about the full network without explicitly computing* **ecolisymfluxesg.** *For instance to get the first flux of* **ecolisymfluxesg** *directly we could use:*

```
fluxesg[[1]].(tags@secoli) // Expand
```

```
R[Fum::Succ] + R[Succ::Fum]
```

As a more meaningful example, let us determine the average number of internal reactions per flux in the minimal generating set of **ecoli**.

```
counts = Map[Count[#.(tags@secoli) // Expand, R[_], ∞] &, fluxesg];
Plus @@ counts / Length[counts] // N
```

```
52.398
```

```
counts = Map[Count[#, R[_], ∞] &, ecolisymfluxesg ];
Plus @@ counts / Length[counts] // N
```

```
52.398
```

The first method is more complicated, slower but uses less memory than the second one because it does not need **ecolisymfluxesg**.

■ *If your coffee break is over, and you have computed the full set* **fluxese** *of elementary vectors of*

secoli, *you can determine the average number of internal reactions per elementary flux in* **ecoli** *by*

```
counts = Map[Count[#.(tags@secoli) // Expand, R[_], ∞] &, fluxese];
Plus @@ counts / Length[counts] // N
```

```
76.1311
```

Analyzing genome scale networks

```
<< "../mathcode/SNAsym.m";
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *The network*

```
{name, yeast} = << yeast.m;
name
```

```
S.cerevisiae iND750

Duarte, N.C., Herrgard, M.J., and Palsson, B.O.
Genome Research, 14: 1298–1309,(2004)
```

```
reactions@yeast // Length
```

```
1150
```

```
metabolites@yeast // Length
```

```
1062
```

Just calculating a generating set for the flux cone of a network of this magnitude seems computationally intractable.

■ *Hence, we shall go for the conversion cone but even this is daunting, as the following long list of extracellular metabolites shows.*

```
with["e", metabolites@yeast]
```

```
{13BDglcn[e], 4abut[e], 5aop[e], 8aonn[e], abt[e], ac[e], acald[e], ade[e], adn[e], akg[e],
 ala-L[e], alltn[e], alltt[e], amet[e], arab-D[e], arab-L[e], arg-L[e], asn-L[e], asp-L[e],
 Biomass[e], btn[e], chol[e], cit[e], co2[e], crn[e], csn[e], cys-L[e], cytd[e], dad-2[e],
 dann[e], dcyt[e], dgsn[e], din[e], dttp[e], duri[e], ergst[e], etoh[e], fmn[e], for[e],
 fru[e], fum[e], gal[e], gam6p[e], gcald[e], glc-D[e], gln-L[e], glu-L[e], gly[e], glyc[e],
 gsn[e], gthox[e], gua[e], h[e], h2o[e], hdca[e], hdcea[e], his-L[e], hxan[e], ile-L[e],
 inost[e], ins[e], k[e], lac-L[e], leu-L[e], lys-L[e], mal-L[e], malt[e], man[e], melib[e],
 met-L[e], mmet[e], na1[e], nh4[e], nmn[e], o2[e], ocdca[e], ocdcea[e], ocdcya[e], orn[e],
 pap[e], pepd[e], phe-L[e], pi[e], pnto-R[e], pro-L[e], ptrc[e], pyr[e], rib-D[e], ribflv[e],
 sbt-D[e], sbt-L[e], ser-L[e], so4[e], spmd[e], sprm[e], srb-L[e], succ[e], sucr[e],
 thm[e], thmmp[e], thmpp[e], thr-L[e], thym[e], thymd[e], tre[e], trp-L[e], ttdca[e],
 tyr-L[e], ura[e], urea[e], uri[e], val-L[e], xan[e], xtsn[e], xyl-D[e], xylt[e], zymst[e]}
```

But it is possible to analyze meaningful restricted scenarios.

First, we choose to ignore the organisms consumption/productions of water by simply zeroing **h2o[e]**

```
yeast1 = constructmnet[
            reactions@yeast /. "h2o"["e"] → 0,
            tags@yeast /. R[x_] → x];
```

The transformation **R[x_] → x** *strips the* **R** *from the tags, because* **constructmnet** *insists on wrapping an* **R** *around them.*

More importantly we shall only treat the following subset of the extracellular metabolites as external.

```
Xts = {"ac"["e"], "acald"["e"], "ala-L"["e"], "Biomass"["e"], "co2"["e"],
  "csn"["e"], "ergst"["e"], "etoh"["e"], "gam6p"["e"], "glc-D"["e"], "hdcea"["e"],
  "ocdcea"["e"], "ocdcya"["e"], "so4"["e"], "xylt"["e"], "zymst"["e"],
  "nh4"["e"], "asp-L"["e"], "ser-L"["e"], "fum"["e"], "gly"["e"], "thr-L"["e"]}
```

```
{ac[e], acald[e], ala-L[e], Biomass[e], co2[e], csn[e], ergst[e],
 etoh[e], gam6p[e], glc-D[e], hdcea[e], ocdcea[e], ocdcya[e], so4[e],
 xylt[e], zymst[e], nh4[e], asp-L[e], ser-L[e], fum[e], gly[e], thr-L[e]}
```

```
yeast2 = setrole[yeast1, Xts, Xt];
```

Calculating the conversion cone

- *While it might just about work, simply typing* **conversiongset[yeast2]** *or* **conversionelvs[yeast2]** *is not a really a good idea for a network of this size.*
  But the first thing **conversiongset** *and* **conversionelvs** *do, is to use* **convfullsimp** *to compute a smaller network which has the same conversion cone. This should always be done,even if takes a while.*

```
ysimp = convfullsimp[yeast2];
```

```
General::spell1 : Possible spelling error: new symbol
   name "ysimp" is similar to existing symbol "simp". More…
```

```
reactions@ysimp // Length
```

```
45
```

```
metabolites@ysimp
```

```
{accoa^m, adp^m, atp^m, co2^c, coa^m, dtmp^c, glu-L^m, hcys-L^c, nadp^c, nadp^m,
 nadph^c, nadph^m, pep^c, pro-L^c, r5p^c, s7p^c, thr-L^c, tyr-L^c, tyr-L^m,
 ura^c, uri^c, xu5p-D^c, ac^e_Xt, acald^e_Xt, ala-L^e_Xt, asp-L^e_Xt, Biomass^e_Xt,
 co2^e_Xt, csn^e_Xt, ergst^e_Xt, etoh^e_Xt, fum^e_Xt, gam6p^e_Xt, glc-D^e_Xt, gly^e_Xt, hdcea^e_Xt,
 nh4^e_Xt, ocdcea^e_Xt, ocdcya^e_Xt, ser-L^e_Xt, so4^e_Xt, thr-L^e_Xt, xylt^e_Xt, zymst^e_Xt}
```

While, compared to **yeast2,** *the achieved reduction is impressive, it is not quite a large as the reaction count indicates.*

The reasons is that the reactions of **ysimp** *tend to be much more complicated than in* **yeast2.** *For instance*

```
(reactions@ysimp)[[4]]
```

```
adp^m + nadp^m + nadph^c + 2 r5p^c + 2 tyr-L^m + ura^c + xu5p-D^c + acald^e_Xt ⟶
 atp^m + nadp^c + nadph^m + pep^c + s7p^c + 2 tyr-L^c + uri^c + etoh^e_Xt
```

■ *From* **ysimp** *the functions* **conversiongset** *and* **conversionelvs** *calculate in one go a minimal generating set or, respectively, the elementary vectors of the conversion cone.*

Doing this in one go , may not be a good idea. The reason is that a few reactions in **ysimp** *do no longer involve any internal metabolites, e.g.*

```
(reactions@ysimp)[[13]]
```

$$\text{glu-L}^{m} + \text{r5p}^{c} + \text{tyr-L}^{c} + \text{xu5p-D}^{c} + \text{asp-L}^{e}_{Xt} + \text{etoh}^{e}_{Xt} \rightharpoonup \text{pep}^{c} + \text{pro-L}^{c} + \text{s7p}^{c} + \text{tyr-L}^{m} + \text{acald}^{e}_{Xt} + \text{fum}^{e}_{Xt} + \text{nh4}^{e}_{Xt}$$

and are thus already in the conversion cone..

So we can split **ysimp** *into two subnets* ***ni*** *and* ***nxt****, where the reactions in* ***nxt*** *do not involve internal metabolites while the ones in* ***ni*** *do. We then find a minimal generating set of conversion for* ***ni*** *and append the reactions from* ***nxt*** *to this set. This results in a generating set for the conversions cone of* **ysimp.** *This is just what the following two commands do:*

```
ints = with[Int, metabolites@ysimp];
yeastgset = partialconversions[ysimp, ints];
```

As you will have noted, the computation took quite long, even though we focussed on the subnetwork ***ni*** *.*

While we now have a generating set for the conversion cone of **yeast2,** *this set may not be minimal. The reason is that, due to splitting, any redundancies between the conversions of* ***ni*** *and the reactions of* ***nxt*** *have been ignored.*

Since **partialconversions** *returns an intermediate result, to facilitate further processing, the output is not a list of reactions but an* **mnet.** *So to count the number of conversions:*

```
reactions@yeastgset // Length
```

```
950
```

Finally, a minimal generating set of the conversion cone of **yeast2,** *is computed by*

```
mingsetyeast2conv = conversions[yeastgset, ZH2gset];
reactions@mingsetyeast2conv // Length
```

```
372
```

conversions *computes conversions of the network given as the first argument even if, as in the above case, there a no longer any internal metabolites. The reactions of the output* **mnet** *are a minimal generating set if the second argument is the SNAmat routine* **ZH2gset;** *if instead* **ZH2elvs** *is specified, all elementary vectors of the conversion coen are enumerated.*

■ *So, if this is the right way to do it, why isn't it just build into* **conversiongset***? The reason is, that it may not be the best way of doing it.*
Note that **partialconversions** *has a second argument and, above, we gave the list of all internal metabolites for this argument. One can also pass a subset of internal metabolites and then* **partialconversions** *will proceed as if only the metabolites in the subset where internal , treating all other metabolites as* **Xt***. In the next step, one can then throw out the remaining internal metabolites.*

```
si = {"accoa"[m], "adp"[m], "atp"[m], "coa"[m],
   "glu-L"[m], "nadp"[m], "nadph"[m], "pro-L"[c], "tyr-L"[m]};
ysimp1 = partialconversions[ysimp, si];
reactions@mingsetyeast2conv === reactions@conversions[ysimp1, ZH2gset]
```

```
True
```

So we have the same result as before, but you will have noticed the more than fivefold reduction in computing time.

■ *Of course you can choose any subset of the internal metabolites of* **ysimp** *for* **si.** *While the final result does not depend on the choice, the computation time does.*
The problem of finding the optimal choice for **si,** *probably is at least as hard as the one of computing a generating set for the conversions. However, SNA can help a bit by making it easy to inspect the network.*
This is how I arrived at the above choice for **si***. As a starting point I thought that throwing out all of the remaining mitochondrial metabolites might be good idea. To investigate this:*

```
si1 = with["m", metabolites@ysimp]
```

```
{accoa[m], adp[m], atp[m], coa[m], glu-L[m], nadp[m], nadph[m], tyr-L[m]}
```

```
{nisi1, nxtsi1} = splitmnet[ysimp, si1];
```

This returns two **mnet***s with* **nisi1** *having all reaction in* **ysimp** *which use at least one metabolite of* **si1** *and* **nxtsi1** *all other reactions. Now*

```
reactions@nisi1 // Length
```

```
18
```

So to get rid of the mitochondrial metabolites, the function **partialconversions** *just has to consider 18 reactions. This is not all that demanding. However*

```
with[Int, Complement[metabolites@nisi1, metabolites@nxtsi1]]
```

```
{accoa[m], adp[m], atp[m], coa[m], glu-L[m], nadp[m], nadph[m], pro-L[c], tyr-L[m]}
```

is a superset of **si1**, *containing in addition* **pro-L[c]**. *So* **pro-L[c]** *is only used by* **nisi1**. *Hence, it makes sense to add* **pro-L[c]** *to the list of metabolites we eliminate already in the first step because this does not increase the subnetwork* **partialconversions** *has to consider.*

```
si === si1⋃{"pro-L"["c"]}
```

```
True
```

- *Finally, to compute all elementary conversion of* **yeast2,** *we can again use* **conversions** *but with the SNAmat routine* **ZH2elvs** *instead of* **ZH2gset**. *As first argument we might as well use* **mingsetyeast2conv**, *the minimal generating set of the conversion cone.*

```
elvsyeast2conv = conversions[mingsetyeast2conv, ZH2elvs];
reactions@elvsyeast2conv // Length
```

```
40969
```

Decomposing elementary conversions into elementary fluxes

■ *Allow me to pick my favorite conversion.*

```
myconv = with[27266477, reactions@elvsyeast2conv] // First
```

$$90626\ ac^{e}_{Xt} + 23710211\ acald^{e}_{Xt} + 5530\ csn^{e}_{Xt} + 35\ ergst^{e}_{Xt} + 23052269\ fum^{e}_{Xt} + 9885\ gam6p^{e}_{Xt} + 4033772\ glc\text{-}D^{e}_{Xt} + 476\ hdcea^{e}_{Xt} + 23401575\ nh4^{e}_{Xt} + 672\ ocdcea^{e}_{Xt} + 252\ ocdcya^{e}_{Xt} + 3865\ so4^{e}_{Xt} + 75\ zymst^{e}_{Xt} \rightharpoonup 22812604\ asp\text{-}L^{e}_{Xt} + 50000\ Biomass^{e}_{Xt} + 15522497\ co2^{e}_{Xt} + 27266477\ etoh^{e}_{Xt} + 323346\ ser\text{-}L^{e}_{Xt} + 12475\ thr\text{-}L^{e}_{Xt}$$

The goal will be to find out how many elementary fluxes give rise to this single conversion.

For this first invert the conversion

```
vnocym = myconv /. (a_ ⇀ b_) → (b ⇀ a)
```

$$22812604\ asp\text{-}L^{e}_{Xt} + 50000\ Biomass^{e}_{Xt} + 15522497\ co2^{e}_{Xt} + 27266477\ etoh^{e}_{Xt} + 323346\ ser\text{-}L^{e}_{Xt} + 12475\ thr\text{-}L^{e}_{Xt} \rightharpoonup 90626\ ac^{e}_{Xt} + 23710211\ acald^{e}_{Xt} + 5530\ csn^{e}_{Xt} + 35\ ergst^{e}_{Xt} + 23052269\ fum^{e}_{Xt} + 9885\ gam6p^{e}_{Xt} + 4033772\ glc\text{-}D^{e}_{Xt} + 476\ hdcea^{e}_{Xt} + 23401575\ nh4^{e}_{Xt} + 672\ ocdcea^{e}_{Xt} + 252\ ocdcya^{e}_{Xt} + 3865\ so4^{e}_{Xt} + 75\ zymst^{e}_{Xt}$$

append **vnocym** *to the reactions in* **yeast2** *and make all metabolites internal.*

```
restricted = joinmnet[
               yeast2,
               constructmnet[{vnocym}, {"vnocym"}]];
restricted = setrole[restricted, metabolites@restricted, Int];
```

Now any elementary flux of **restricted** *corresponds to either a futile cycle of* **yeast2** *or to an elementary flux of* **yeast2** *which has* **myconv** *as conversion.*

We first calculate a minimal generating set of the flux cone by

```
{gfluxes, nrev} = symfluxgset[restricted];
nrev
```

```
9
```

```
{gfluxes // Length, with["vnocym", gfluxes] // Length}
```

```
{1044, 1024}
```

So there only 20 futile cycles and the 1024 fluxes have **myconv** *as conversion.*

■ *Now, one might further wish to enumerate all elementary fluxes for* **restricted.** *But their number is* 3.7×10^7, *so trying to use* **symfluxelvs** *would fail disastrously. The method presented below, however, might just about make it possible to enumerate all the elementary fluxes, even if we only illustrate it for the minimal generating set.*
Initially we proceed as in Tutorial 7 and find the generating fluxes in vector form for the reduced network of **restricted.**

```
rrestricted = fluxsimp[restricted];
{stoich, nrev} = mnet2stoich[rrestricted];
{rgfluxes, rnrev} = fluxgset[stoich, nrev];
rnrev
```

```
General::spell1 :
 Possible spelling error: new symbol name "rrestricted" is
   similar to existing symbol "restricted". More…

General::spell1 : Possible spelling error: new symbol
   name "rgfluxes" is similar to existing symbol "gfluxes". More…

General::spell : Possible spelling error: new symbol
   name "rnrev" is similar to existing symbols {nnrev, nrev}. More…
```

```
9
```

```
rgfluxes // Length
```

```
144
```

So the reduced network has fewer generating fluxes than **restricted.**
The reason, becomes apparent by inspecting the tags of **rrestricted.**

```
{with[ALT, tags@rrestricted][[2]]} // TableForm
```

```
65250 ALT[R[CRNCARtm] + R[CSNATifm], R[ACRNtim] + R[CRNtim] + R[CSNATifm]] + 25925
```

The symbol **ALT** *indicates that the simplification procedure has generated duplicate reactions. To illustrate this, we look at the second* **ALT** *in the line above.*

```
with[{"GLCS2", "GBEZ", "GLYGS"}, trpairs@restricted] // TableForm
```

```
R[GBEZ]     14glun^c ⇀ glycogen^c + h2o^c
R[GLCS2]    udpg^c ⇀ glycogen^c + h^c + udp^c
R[GLYGS]    h2o^c + udpg^c ⇀ 14glun^c + h^c + udp^c
```

The simplification procedure has found that GBEZ *and* GLYGS *can only run in tandem and are then equivalent to* $h2o^c + udpg^c \rightharpoonup h2o^c + glycogen^c + h^c + udp^c$, *which is the same as* GLCS2. *So for any elementary flux using* GLCS2 *there is another one using* GBEZ *as well as* GLYGS.

Computationally it makes sense to generate such alternatives only at the very end of the calculation. This is just what **symfluxgset** *does, applying* **ALTexpfull** *to the intermediate result of* **fluxgset** *to obtain the final 1044 fluxes.*

```
Map[ALTexpfull, rgfluxes.(tags@rrestricted)] // Flatten // Length
```

```
1044
```

But of course, one can also expand the elements of **rgfluxes** *individually.*

```
ALTexpfull[ rgfluxes[[1]].(tags@rrestricted)] // Length
```

```
1
```

```
ALTexpfull[ rgfluxes[[20]].(tags@rrestricted)] // Length
```

```
8
```

So **symfluxelvs[restricted]** *is bound to run out of memory in the final stage of the calculation, due to the sheer size of the result. Enumerating the elementary fluxes of*

rrestricted *is a much more reasonable proposition, since their number is only* 4.6×10^6 *. So if you have 4GB of memory and can do without your computer for upto 10 hours, you might try* **fluxelvs[stoich,nrev];**

In fact, SNAmat provides the function **signfluxelvs,** *which manages to squeeze the computation into 2GB by returning just the signs of the flows in each flux and using a compressed output format. But this is so messy that I am not sure whether I really consider* **signfluxelvs** *to be 'officially' part of SNAmat.*

Flux Balance Analysis

```
<< "../mathcode/SNAsym.m";
```

```
LinkObject[../mathcode/../pairelvs/pairelvsIFsh, 2, 2]
```

■ *We shall look at the yeast network and consider the same scenario as in Tutorial 8 .*

```
{name, yeast} = << yeast.m;
name
```

```
S.cerevisiae iND750

Duarte, N.C., Herrgard, M.J., and Palsson, B.O.
Genome Research, 14: 1298-1309,(2004)
```

```
yeast1 = constructmnet[
            reactions@yeast /. "h2o"^"e" → 0,
            tags@yeast /. R[x_] → x];

Xts = {"ac"^"e", "acald"^"e", "ala-L"^"e", "Biomass"^"e", "co2"^"e",
   "csn"^"e", "ergst"^"e", "etoh"^"e", "gam6p"^"e", "glc-D"^"e", "hdcea"^"e",
   "ocdcea"^"e", "ocdcya"^"e", "so4"^"e", "xylt"^"e", "zymst"^"e",
   "nh4"^"e", "asp-L"^"e", "ser-L"^"e", "fum"^"e", "gly"^"e", "thr-L"^"e"};

yeast2 = setrole[yeast1, Xts, Xt];
```

■ *The first thing is, to prepare the network for flux balance analysis by*

```
fbayeast2 = FBAprep[yeast2];
```

We next set up a list of constraints on the flows through some of the reactions:

```
constr = {
   { exch["csn"^"e"_"xt"], {-1, -1 / 2} },
   {R["ORNt3m"], {0, 1}}
  };
constr // MatrixForm
```

$$\begin{pmatrix} \text{Rx}[\text{csn}^{\text{e}}_{\text{Xt}} \rightleftharpoons 0] & \{-1, -\frac{1}{2}\} \\ \text{R}[\text{ORNt3m}] & \{0, 1\} \end{pmatrix}$$

Each constraint is a list consisting of a reaction tag and a pair of numbers. The first number of the pair gives the lower and the second one the upper bound for the allowed flow through the reaction. The numbers in the bounds should be integers, rationals or $\pm\infty$ *unless the option* **numeric** *(see below) is used. Then reals may be also occur in the bounds.*

The default admissible range for reactions not specified in the constraints is $\{-\infty,\infty\}$ *for reversible and* $\{0,\infty\}$ *for irreversible reactions.*

Now we are ready to call **FBA** *, passing as the first argument the structure returned by* **FBAprep ,** *specifying which flow should be maximized by the tag of the corresponding reaction passed as the second argument, and giving the constraints in the third argument.*

```
{opt, flux} = FBA[fbayeast2 , exch["Biomass"^"e"_"xt"], constr];
opt
```

$$\frac{10000}{1607}$$

opt *gives the maximal flow through* $\mathtt{Rx[Biomass^{e}_{xt} \rightleftharpoons 0]}$ *given the constraints* **constr***;*

flux *is one (of often many) flux vectors realizing* **opt** .

The symbolic form familiar from e.g. **symfluxelvs** *is used for* **flux**.

```
flux
```

$$\frac{11348\,R[13GS]}{1607} + \frac{2862\,R[2OXOADPtim]}{1607} - \frac{5610\,R[3MOBtm]}{1607} - \frac{1927\,R[3MOPtm]}{1607} + \frac{2862\,R[AASAD2]}{1607} + \frac{2862\,R[AATA]}{1607} + \frac{8288419\,R[ACALDt]}{8035} + \frac{9156\,R[ACCOACr]}{8035} + R[ACGKm] + \frac{1927\,R[ACHBSm]}{1607} + \frac{5610\,R[ACLSm]}{1607} + \frac{93491\,R[ACOAH]}{8035} + \frac{10188\,R[ACONTm]}{1607} + R[ACOTAim] + \frac{13050\,R[ACRNtim]}{1607} + \frac{90626\,R[ACt2r]}{8035} + \frac{247\,R[ADHAPR\_SC]}{1607} + \frac{9689\,R[ADK1]}{1607} + \frac{240\,R[ADNK1]}{1607} + \frac{573\,R[ADSK]}{1607} + \frac{1159\,R[ADSL1r]}{1607} + \frac{980\,R[ADSL2r]}{1607} + \frac{1159\,R[ADSS]}{1607} + \frac{247\,R[AGAT\_SC]}{1607} + R[AGPRim] + \frac{180\,R[AHCi]}{1607} + \frac{507\,R[AHSERL2]}{1607} + \frac{1643\,R[AICART]}{1607} + \frac{980\,R[AIRCr]}{1607} + \frac{228126\,R[ALAt2r]}{8035} + \frac{205186\,R[ALATA\_L]}{8035} - \frac{8288419\,R[ALCD2x]}{8035} + \frac{284\,R[ANPRT]}{1607} + \frac{284\,R[ANS]}{1607} + R[ARGSL] + R[ARGSSr] + \frac{507\,R[ASADi]}{1607} + \frac{1017\,R[ASNS1]}{1607} + \frac{507\,R[ASPKi]}{1607} + \frac{8933\,R[ASPt2m]}{1607} - \frac{3989292\,R[ASPt2r]}{8035} - \frac{4075182\,R[ASPTA]}{8035} + \frac{8933\,R[ASPTAm]}{1607} + \frac{663\,R[ATPPRT]}{1607} + \frac{307101\,R[ATPS]}{8035} + \frac{7537\,R[ATPtm\text{-}H]}{1607} +$$

$$\frac{10000\,R[BioM]}{1607} + \frac{573\,R[BPNT]}{1607} + R[CBPS] + \frac{2359\,R[CHORM]}{1607} + \frac{2643\,R[CHORS]}{1607} - \frac{4035222\,R[CO2t]}{8035} - \frac{49274\,R[CO2tm]}{1607} + \frac{13050\,R[CRNtim]}{1607} + \frac{10188\,R[CSm]}{1607} + \frac{13050\,R[CSNAT]}{1607} + \frac{13050\,R[CSNATifm]}{1607} + \frac{1106\,R[CSND]}{1607} + \frac{1106\,R[CSNt2]}{1607} + \frac{447\,R[CTPS1]}{1607} + \frac{66\,R[CYSS]}{1607} - \frac{272\,R[CYTK1]}{1607} - \frac{96\,R[DADK]}{1607} + \frac{66\,R[DAGPYP\_SC]}{1607} + \frac{175\,R[DASYN\_SC]}{1607} - \frac{24\,R[DCMPDA]}{1607} + \frac{2643\,R[DDPA]}{1607} - \frac{24\,R[DGK1]}{1607} + \frac{5610\,R[DHAD1m]}{1607} + \frac{1927\,R[DHAD2m]}{1607} + \frac{36\,R[DHFRi]}{1607} + \frac{2643\,R[DHQS]}{1607} + \frac{2643\,R[DHQTi]}{1607} + \frac{8079\,R[DOLPMMer]}{1607} + \frac{8079\,R[DOLPMTcer]}{1607} - \frac{8079\,R[DOLPt2er]}{1607} + \frac{60\,R[DURIK1]}{1607} - \frac{60\,R[DURIPP]}{1607} + \frac{5286\,R[ENO]}{1607} + \frac{7\,R[ERGSTt]}{1607} - \frac{8288419\,R[ETOHt]}{8035} - \frac{896\,R[FACOAL140]}{8035} + \frac{756\,R[FACOAL160]}{8035} + \frac{476\,R[FACOAL161]}{8035} + \frac{28\,R[FACOAL180]}{1607} + \frac{672\,R[FACOAL181]}{8035} + \frac{252\,R[FACOAL182]}{8035} + \frac{280\,R[FAS100COA]}{1607} + \frac{1344\,R[FAS120COA]}{8035} + \frac{1176\,R[FAS140COA]}{8035} + \frac{896\,R[FAS160]}{8035} + \frac{28\,R[FAS180]}{1607} + \frac{280\,R[FAS80COA\_L]}{1607} - \frac{2003\,R[FBA]}{1607} + \frac{2003\,R[FBP]}{1607} + \frac{4057533\,R[FDH]}{8035} + \frac{9144\,R[FORtm]}{1607} - \frac{4011813\,R[FTHFL]}{8035} - \frac{9144\,R[FTHFLm]}{1607} + \frac{3940586\,R[FUM]}{8035} + \frac{3921856\,R[FUMt2r]}{8035} + \frac{1647\,R[G5SADr]}{1607} + \frac{1647\,R[G5SD2]}{1607} + \frac{1977\,R[G6PDA]}{1607} + \frac{16767\,R[GALU]}{1607} + \frac{1977\,R[GAM6Pt]}{1607} + \frac{5286\,R[GAPD]}{1607} + \frac{980\,R[GARFTi]}{1607} + \frac{247\,R[GAT2\_SC]}{1607} + \frac{5185\,R[GBEZ]}{1607} + \frac{4074263\,R[GHMT2r]}{8035} + \frac{24\,R[GK1]}{1607} + \frac{7069\,R[GLNS]}{1607} + \frac{1647\,R[GLU5K]}{1607} - \frac{3951411\,R[GLUDy]}{8035} + \frac{980\,R[GLUPRT]}{1607} - \frac{5719\,R[GLUt2m]}{1607} + \frac{5185\,R[GLYGS]}{1607} - \frac{4054843\,R[GLYt2r]}{8035} + \frac{484\,R[GMPS2]}{1607} + \frac{3675511\,R[H2Ot]}{8035} + \frac{14657\,R[H2Otm]}{1607} + \frac{2862\,R[HACNHm]}{1607} + \frac{2862\,R[HCITSm]}{1607} + \frac{300972\,R[HCO3E]}{8035} + \frac{476\,R[HDCEAt]}{8035} + \frac{2862\,R[HICITDm]}{1607} + \frac{663\,R[HISTD]}{1607} + \frac{663\,R[HISTP]}{1607} + \frac{507\,R[HSDxi]}{1607} + \frac{507\,R[HSERTA]}{1607} + \frac{663\,R[HSTPT]}{1607} + \frac{10188\,R[ICDHxm]}{1607} + \frac{663\,R[IG3PS]}{1607} + \frac{663\,R[IGPDH]}{1607} + \frac{284\,R[IGPS]}{1607} - \frac{1927\,R[ILETA]}{1607} - \frac{1643\,R[IMPC]}{1607} + \frac{484\,R[IMPD]}{1607} + \frac{2964\,R[IPMD]}{1607} - \frac{2964\,R[IPPMIa]}{1607} - \frac{2964\,R[IPPMIb]}{1607} + \frac{2964\,R[IPPS]}{1607} + \frac{5610\,R[KARA1im]}{1607} + \frac{1927\,R[KARA2im]}{1607} - \frac{2964\,R[LEUTA]}{1607} - \frac{13050\,R[MALtm]}{1607} + \frac{8079\,R[MAN1PT]}{1607} - \frac{8079\,R[MAN6PI]}{1607} - \frac{8079\,R[MANNANter]}{1607} + \frac{2862\,R[MCITDm]}{1607} + \frac{4005836\,R[MDH]}{8035} - \frac{41632\,R[MDHm]}{1607} + \frac{28582\,R[ME1m]}{1607} + \frac{180\,R[METAT]}{1607} + \frac{687\,R[METS]}{1607} + \frac{60\,R[MFAPS\_SC]}{1607} + \frac{53\,R[MI1PP]}{1607} + \frac{53\,R[MI1PS]}{1607} + \frac{9144\,R[MLTHFtm]}{1607} + \frac{4024928\,R[MTHFC]}{8035} + \frac{9144\,R[MTHFCm]}{1607} + \frac{4024928\,R[MTHFD]}{8035} +$$

$$\frac{9144\,R[\text{MTHFDm}]}{1607}+\frac{687\,R[\text{MTHFR3}]}{1607}+\frac{10308\,R[\text{NDPK1}]}{1607}+\frac{17214\,R[\text{NDPK2}]}{1607}-\frac{272\,R[\text{NDPK3}]}{1607}-\frac{24\,R[\text{NDPK5}]}{1607}+\frac{3964061\,R[\text{NH4t}]}{8035}-\frac{1927\,R[\text{NH4tm}]}{1607}+\frac{60\,R[\text{NTD6}]}{1607}+\frac{42887\,R[\text{OAAt2m}]}{1607}+R[\text{OCBTi}]+\frac{672\,R[\text{OCDCEAt}]}{8035}+\frac{252\,R[\text{OCDCYAt}]}{8035}+\frac{2964\,R[\text{OMCDC}]}{1607}+R[\text{ORNt3m}]+R[\text{ORNTACim}]+\frac{2862\,R[\text{OXAGm}]}{1607}+\frac{1647\,R[\text{P5CR}]}{1607}+\frac{573\,R[\text{PAPSR}]}{1607}+\frac{283781\,R[\text{PC}]}{8035}-\frac{21\,R[\text{PEtm\_SC}]}{32140}+\frac{60\,R[\text{PETOHM\_SC}]}{1607}-\frac{17054\,R[\text{PGI}]}{1607}-\frac{5286\,R[\text{PGK}]}{1607}-\frac{5286\,R[\text{PGM}]}{1607}-\frac{16767\,R[\text{PGMT}]}{1607}-\frac{1339\,R[\text{PHETA1}]}{1607}+\frac{53\,R[\text{PINOS\_SC}]}{1607}-\frac{5513\,R[\text{PIt2m}]}{1607}-\frac{8079\,R[\text{PMANM}]}{1607}+\frac{60\,R[\text{PMETM\_SC}]}{1607}+\frac{34041\,R[\text{PPA}]}{1607}-\frac{1106\,R[\text{PPM}]}{1607}+\frac{1020\,R[\text{PPND}]}{1607}+\frac{1339\,R[\text{PPNDH}]}{1607}+\frac{980\,R[\text{PRAGSr}]}{1607}+\frac{284\,R[\text{PRAIi}]}{1607}+\frac{980\,R[\text{PRAIS}]}{1607}+\frac{663\,R[\text{PRAMPC}]}{1607}+\frac{980\,R[\text{PRASCS}]}{1607}+\frac{663\,R[\text{PRATPP}]}{1607}+\frac{980\,R[\text{PRFGS}]}{1607}+\frac{663\,R[\text{PRMICIi}]}{1607}+\frac{1927\,R[\text{PRPPS}]}{1607}+\frac{2643\,R[\text{PSCVTi}]}{1607}+\frac{105\,R[\text{PSERDm\_SC}]}{1607}+\frac{122\,R[\text{PSERS\_SC}]}{1607}+\frac{21\,R[\text{PStm\_SC}]}{32140}-\frac{60\,R[\text{PUNP1}]}{1607}+\frac{60\,R[\text{PUNP2}]}{1607}-\frac{1046\,R[\text{PYNP2r}]}{1607}-\frac{15435\,R[\text{PYRt2m}]}{1607}+\frac{96\,R[\text{RNDR1}]}{1607}+\frac{24\,R[\text{RNTR2}]}{1607}-\frac{14931\,R[\text{RPE}]}{1607}-\frac{14931\,R[\text{RPI}]}{1607}+\frac{2862\,R[\text{SACCD1}]}{1607}+\frac{2862\,R[\text{SACCD2}]}{1607}+\frac{66\,R[\text{SERATi}]}{1607}+\frac{4085893\,R[\text{SERt2r}]}{8035}+\frac{2643\,R[\text{SHK3D}]}{1607}+\frac{2643\,R[\text{SHKK}]}{1607}+\frac{573\,R[\text{SLFAT}]}{1607}+\frac{773\,R[\text{SO4ti}]}{1607}-\frac{573\,R[\text{SULR}]}{1607}+\frac{11898\,R[\text{TALA}]}{1607}-\frac{9144\,R[\text{THFtm}]}{1607}+\frac{1927\,R[\text{THRD\_Lm}]}{1607}+\frac{1927\,R[\text{THRt2m}]}{1607}+\frac{3841\,R[\text{THRt2r}]}{1607}+\frac{11898\,R[\text{TKT1}]}{1607}+\frac{9255\,R[\text{TKT2}]}{1607}+\frac{36\,R[\text{TMDS}]}{1607}-\frac{2250\,R[\text{TPI}]}{1607}+\frac{693\,R[\text{TRDR}]}{1607}+\frac{234\,R[\text{TRE6PP}]}{1607}+\frac{234\,R[\text{TRE6PS}]}{1607}+\frac{66\,R[\text{TRIGS\_SC}]}{1607}+\frac{284\,R[\text{TRPS1}]}{1607}-\frac{1020\,R[\text{TYRTA}]}{1607}+\frac{447\,R[\text{UMPK}]}{1607}+\frac{1046\,R[\text{URIK2}]}{1607}-\frac{2646\,R[\text{VALTA}]}{1607}+\frac{36084\,R[\text{XYLK}]}{1607}+\frac{36084\,R[\text{XYLTD\_D}]}{1607}+\frac{36084\,R[\text{XYLTt}]}{1607}+\frac{15\,R[\text{ZYMSTt}]}{1607}-\frac{90626\,Rx[\text{ac}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{8288419\,Rx[\text{acald}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{228126\,Rx[\text{ala-L}^{e}_{Xt}\rightleftharpoons 0]}{8035}+\frac{3989292\,Rx[\text{asp-L}^{e}_{Xt}\rightleftharpoons 0]}{8035}+\frac{10000\,Rx[\text{Biomass}^{e}_{Xt}\rightleftharpoons 0]}{1607}+\frac{4035222\,Rx[\text{co2}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{1106\,Rx[\text{csn}^{e}_{Xt}\rightleftharpoons 0]}{1607}-\frac{7\,Rx[\text{ergst}^{e}_{Xt}\rightleftharpoons 0]}{1607}+\frac{8288419\,Rx[\text{etoh}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{3921856\,Rx[\text{fum}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{1977\,Rx[\text{gam6p}^{e}_{Xt}\rightleftharpoons 0]}{1607}+\frac{4054843\,Rx[\text{gly}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{476\,Rx[\text{hdcea}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{3964061\,Rx[\text{nh4}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{672\,Rx[\text{ocdcea}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{252\,Rx[\text{ocdcya}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{4085893\,Rx[\text{ser-L}^{e}_{Xt}\rightleftharpoons 0]}{8035}-\frac{773\,Rx[\text{so4}^{e}_{Xt}\rightleftharpoons 0]}{1607}-\frac{3841\,Rx[\text{thr-L}^{e}_{Xt}\rightleftharpoons 0]}{1607}-\frac{36084\,Rx[\text{xylt}^{e}_{Xt}\rightleftharpoons 0]}{1607}-\frac{15\,Rx[\text{zymst}^{e}_{Xt}\rightleftharpoons 0]}{1607}$$

To optimize Biomass production one could also have used :

```
{opt, flux} = FBA[fbayeast2 , R["BioM"], constr];
opt
```

$$\frac{10000}{1607}$$

To minimize the flow through a reaction precede its tag with a minus sign.

```
{opt, flux} = FBA[fbayeast2 , -R["BioM"], constr];
opt
```

$$\frac{2500}{553}$$

For solving thousands of FBA problems, **FBA** *is a bit slow. Hence* **FBAprep** *has some options to speed up subsequent calls to* **FBA .**

```
Options[FBAprep]
```

```
{simplification → 1, exchonly → False, numeric → False}
```

■ *The most straightforward option is* **numeric**, *which instructs* **FBA** *to do the linear programming numerically:*

```
nfbayeast2 = FBAprep[yeast2, numeric → True];
```

```
General::spell1 :
 Possible spelling error: new symbol name "nfbayeast2" is
   similar to existing symbol "fbayeast2". More…
```

```
{opt, flux} = FBA[nfbayeast2 , exch["Biomass"ᵉ"xt" ], constr];
opt
```

```
6.22278
```

■ *The option* **simplification** *takes values 0,1,2 or 3; it controls how much effort* **FBAprep** *puts into simplifying the network when preparing it for* **FBA.** *Using* **simplification → 0** *means no simplification, while* **FBAprep** *will itself use linear programming if you specify a value greater than 1. While speeding up* **FBA,** *this may substantially increase the execution time of* **FBAprep**.

```
sfbayeast2 = FBAprep[yeast2, simplification → 2];
```

```
General::spell :
 Possible spelling error: new symbol name "sfbayeast2" is
   similar to existing symbols {fbayeast2, nfbayeast2}. More…
```

```
{opt, flux} = FBA[sfbayeast2 , exch["Biomass"$_{xt}^{e}$], constr];
opt
```

$$\frac{10000}{1607}$$

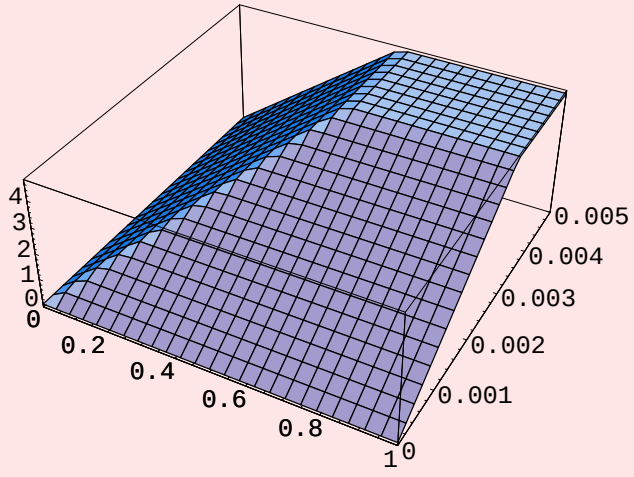
■ *If you only need to specify constraints for exchange reactions and the optimization target is an exchange reaction as well, you are in fact asking a question about the conversion cone. You should then use* **exchonly → True,** *since this will allow* **FBAprep** *to simplify the network to a much greater extent. Then FBA becomes fast enough to be used in 3D−plots if you are a little patient.*

```
cnbayeast2 = FBAprep[yeast2,
               exchonly → True, simplification → 3, numeric → True];

ppconstr[csn_, ergst_] =
  {
   { exch["csn"$_{xt}^{e}$], {-csn, 0} },
   { exch["ergst"$_{xt}^{e}$], {-ergst, 0} },
   {exch["zymst"$_{xt}^{e}$], {-0.007, 0}}
  };
ppconstr[csn, ergst] // MatrixForm
```

$$\begin{pmatrix} \text{Rx}[\text{csn}_{xt}^{e} \rightleftharpoons 0] & \{-\text{csn}, 0\} \\ \text{Rx}[\text{ergst}_{xt}^{e} \rightleftharpoons 0] & \{-\text{ergst}, 0\} \\ \text{Rx}[\text{zymst}_{xt}^{e} \rightleftharpoons 0] & \{-0.007, 0\} \end{pmatrix}$$

```
Plot3D[FBA[cnbayeast2 , exch["Biomass"_"e"], ppconstr[csn, ergst]] // First,
 {csn, 0, 1}, {ergst, 0, 0.005}]
```




```
- SurfaceGraphics -
```