USER GUIDE

NetSplitter V 1.2

January 2011

Copyright: Wynand Verwoerd

Centre for Advanced Computational Solutions
Dept WF & Molecular Bioscience
Faculty of Agriculture & Life Sciences
Lincoln University
Ellesmere Junction Road
CHRISTCHURCH 7647
New Zealand

Email: wynand.verwoerd@lincoln.ac.nz

Table of Contents

1 Copyright

2 System requirements

This Notebook requires either *Mathematica* version 6 or higher, or the corresponding version of *Mathematica Player Pro* to use its full functionality.

These programs are obtainable from Wolfram Research, Inc.

The free program *Mathematica Player*, which can be downloaded from Wolfram, unfortunately does not support dialog windows and so this notebook will not function in *Mathematica Player*.

3 Program Concepts

This program (Netsplitter) performs a straightforward function: it reads the specification of a large metabolic network as an SBML file or as a stoichiometric matrix, then it splits the network into a collection of smaller component networks, and creates and stores the specification for each of these subnetworks into its own file.

A metabolic network is usually represented with metabolites as nodes and reactions as edges. Because a reaction can involve more than two metabolites, it is formally a hypergrpah rather than a simple graph; alternatively, reactions can be represented as a second set of nodes of a different kind, in which case it would be termed a bipartite graph. Neither of these representations are suitable for the algorithms used in the program. So instead Netsplitter reduces the network to a simple graph in which all nodes are metabolites, and in effect each edge becomes a summation over all reactions that connect the two metabolites.

There is an important distinction between nodes that lie on the periphery of the network and those in the interior: mass balance of the metabolites in the interior is guaranteed by the stoichiometry constraints associated with reactions, but the metabolites on the periphery (termed *external* metabolites) represent the inflows and outflows of the metabolic network, i.e. they are in effect buffered reservoirs such as water, oxygen, CO_2 or nutrients taken up by the cell and waste products.

When the complete network is split into subnets, all nodes that are "cut" become periphery nodes in the subnets. As that means that mass balance for such a metabolite is lost, to retain as much information about the network as possible we would like to a) make as few as possible metabolites external and b) where possible, select these new externals to be metabolites for which a reservoir is biochemically plausible - such as carrier metabolites like NAD(P), ATP, etc.

To facilitate this, Netsplitter produces a display of the current network in matrix form, in which groups of *internal* metabolites that are associated with one another by their network connections are visible as blocks, and block overlaps indicate metabolites that form links between the blocks. The strategy is to decouple the blocks by making some of the linking metabolites external - i.e., the network is "cut" at these nodes. *A fully decoupled block corresponds to a subnetwork*. The program selects a few of the overlap metabolites which are most likely to represent links, to the user for approval. The ones that are approved, are *made external* and the blocking recalculated; then a new set of candidates are presented. This continues through as many *selection rounds* as necessary, until the user is satisfied that the desired degree of splitting of the network has been achieved. There are further aspects of the procedure that the user can interactively control, as elaborated below.

In order to keep track of the metabolic functions associated with each subnet, lists of metabolites in each block are displayed. Moreover, when some particular function (say flavonoid metabolism) is of interest, the user can preselect some typical metabolites involved in this by supplying a list in a special input file. These metabolites (called *target* metabolite*s*) are then visually tracked so that the blocks containing them are pinpointed on the screen.

The matrix that is manipulated by Netsplitter is a probability matrix. Each element of this matrix is associated with two metabolites, identified by its row and column indices. Consider a random walk along the metabolic network that starts at the row metabolite (the *source*) and ends at the column metabolite (the *sink*). The value of the matrix element represents the probability that a random walk along the metabolic network starting from the source will end up at the sink. If this value is zero, it means that there is no path from that source to that sink.

It turns out that when all random walks of all lengths are combined, most metabolites end up as being sources, and the rest as sinks. In other words, if we start a "random walker" on each node of the network, they all end up on a limited subset of sink nodes if we allow random hops to continue indefinitely. So no node is both a source and a sink at the same time, except in the trivial sense that the random walker that starts on a sink node may remain there. (The only exception is that sometimes a small set of 2 or 3 metabolites end up as being fully connected to each other, and in effect form a combined sink).

If we interpret each non-zero element in the probability matrix for all random walks to represent an edge in a graph, the separation of metabolites into two disjoint sets (sources and sinks) means that this graph will be a Directed Acyclic Graph or *DAG*. It is not the same graph as the one that represented the metabolic network; it has the same set of nodes (the metabolites) but a much simpler, star-like structure of edges. Its claim to fame is that it encapsulates only the linking of nodes, rather than the full network structure. Note that it is always a directed graph, irrespective of whether the original was directed or not.

When there are blocks of elements in the DAG matrix that are non-overlapping (i.e., they do not share any rows or columns) these correspond to disconnected subgraphs in the DAG graph. Each such disconnected subgraph contains the internal nodes of a subnetwork of the metabolic network; connections between subnetworks are carried by the external metabolites that have been removed in the elimination rounds before. Once the user halts the elimination process, Netsplitter creates the subnetworks by using the information in the stoichiometry matrix to add all external metabolites that are connected to the internals in a particular block, back to that block. The result is that each internal metabolite is unique to its own subnet, while external metabolites can be duplicated among subnets. In effect, when splitting a network into two subnets, the connecting nodes are cut into two pieces e.g. becoming an output node for one subnet and an input node for the other.

While the stoichiometry matrix is usually very sparse, the matrix obtained when summing over reactions to convert to the simple graph representation is less so, and finally the DAG matrix when represented in terms of sources and sinks usually appears pretty fully connected - i.e., there is a path from almost every source to every sink. In that state it is almost impossible to recognise any block structure. The culprits responsible are usually a small set of very highly connected metabolites. While the vast majority of metabolites only participate in 2 or 3 reactions (independently of the size of the network), metabolites like water, oxygen, ATP, ADP, NADP, NADPH. etc participate in many reactions, typically increasing logarithmically with the network size. As is known from percolation theory, only a relatively small, critical number of the potential links in a network is necessary to create long distance connections. The metabolites creating this degree of connectivity needs to be eliminated (made external) before the underlying network structure is revealed in the DAG matrix.

Netsplitter provides two strategies to achieve this. First, all metabolites with connectivities higher than a threshold value are automatically classified as external. This threshold should in principle depend on network size, but a value of 8 has been found to work well for networks from around 100 metabolites or reactions, to as many as 2000. This value catches most, but not all problem nodes. If the threshold is lowered, the danger is that metabolites that participate in a number of closely related reactions all in one small section of the network are also made external inadvertently. To deal with this, the user can "fine-tune" the selection of externals by supplying a list of common environment molecules, carrier metabolites etc. that the program may take as default externals.

4 Directories and files

The Netsplitter project is hosted by the Bionformatics Organisation and has a home directory at the URL

http://www.bioinformatics.org/groups/?group_id=1067.

The latest Netsplitter version can be downloaded from

ftp://ftp.bioinformatics.org/pub/netsplitter/

The downloaded zip-file unpacks into a directory, e.g. “Netsplitter_V11”, which contains the main *Mathematica* notebook file “Netsplitter.nb” that drives the application, an example data file “Demo.tsv”, and a directory called “Packages” where most of the program code resides. The only critical part of this is that the naming and positioning of the Packages directory relative to the notebook should be maintained in order for Netsplitter to find its code.

Data files can be located anywhere in the file system because they are explicitly selected by the user, and the different types can be in different locations. However, the location of the primary input file, the network specification or S-matrix file, is important because Netsplitter puts all of its output files in the same directory.

The recommended procedure is to create a project directory, named e.g. after the organism being studied, to contain this input file and any others that are specific to the project. Such project directories may be located at any convenient locations in the file system, not necessarily in the Netsplitter home directory.

The name of this project directory is used as an identifying label in e.g. the heading of the printout file that can optionally be produced.

Separating projects in this way avoids overwriting unrelated files, because output files are assigned generic names like “NSPL_Printout.PDF” by the program and replace earlier versions unless the user has manually renamed them. Also, if the option to save subnets is chosen, a new subdirectory “Subnetworks” is created in the project directory to contain them and it similarly needs to be renamed if replacement is to be avoided.

In the “Packages” directory, the file “Netsplitter.m” contains initialisation data for setting some configuration data such as display colours and default choices on the control panel. These can be edited by the user with a text editor, using care not to disturb the formatting.

5 Input files

Some input files need to be prepared before running Netsplitter. They are all simple text files.

5.1 *Stoichiometry matrix (S-matrix)*

The main input file is the stoichiometry matrix. For simple examples it can be typed manually into a text editor, but for realistic networks would probably need to be prepared by a separate program that extracts the data from a database.

- OPTION 1

Netsplitter can read a network specified in a standard .SBML (Systems Biology Markup Language) file and extracts the S-matrix as well as the lists of metabolites and reactions from it. Where available, this is the preferred input option.

- OPTION 2

Alternatively, it reads a tab separated (.TSV) file as described here. The type of input is recognised from the file name extension - only .SBML or .TSV are accepted. The data file DEMO.TSV supplied as part of the download package is an example of this format.

Where the network specification is only available to the user as a spreadsheet file, the TSV option offers a relatively straightforward way to prepare data for Netsplitter.

Alternatively, to extract a network specification from a public metabolic database, a set of AWK scripts (run by a Unix script called Fullnet) is available from the same source as Netsplitter to extract the network from a Biocyc database in flat file format and produce a file Smatrix.tsv as a tab-separated (.TSV) file in the format of the following example:

```
<Title>
Flavonoid Block 1 with 8 internal compounds
<Reactions>
RXN-8088    RXN-8089    MANDELONITRILE-LYASE-RXN      CINNAMYL-
ALCOHOL-DEHYDROGENASE-RXN     CINNAMOYL-COA-REDUCTASE-RXN
RXN-2001    RXN-2002    RXN-2003    RXN-2005    RXN-2006
RXN-2007    RXN-2009    RXN-6722    RXN-6723    RXN-6724
BENZYL-ALC-DEHYDROGENASE-RXN  RXN-1981
<ReversibleReactions>
MANDELONITRILE-LYASE-RXN
<InternalCompounds>
HYDROXYCINNAMOYL-COA SYSTEM   CINNAMOYL-COA SYSTEM    OXOCINNAMOYL-
COA SYSTEM  CINNAMALDEHYDE SYSTEM   BENZOYLCOA SYSTEM
BENZOATE SYSTEM   BENZALDEHYDE SYSTEM     CPD-110 SYSTEM
<ExternalCompounds>
NAD SYSTEM  NADH SYSTEM CPD-6443 SYSTEM   CPD-6442 SYSTEM   CPD-6441
SYSTEM
SALICYLOYL-COA SYSTEM   PYRUVATE SYSTEM   MANDELONITRILE SYSTEM   HCN
SYSTEM      WATER SYSTEM
CO-A SYSTEM BENZYL-ALCOHOL SYSTEM   CPD-674 SYSTEM    NADPH SYSTEM
      NADP SYSTEM
```

```
S-ADENOSYLMETHIONINE SYSTEM   ADENOSYL-HOMO-CYS SYSTEM      OXYGEN-
MOLECULE SYSTEM   HYDROGEN-PEROXIDE SYSTEM      CINNAMYL-ALC SYSTEM
ISOCHORISMATE SYSTEM    ACETYL-COA SYSTEM
<Stoichiometry>
%%MatrixMarket matrix coordinate real general
%
%  Stoichiometry matrix with 30 rows, 17 columns and 70 nonzero
elements.
%
30 17 70
1 8  -1.0000000000000000E+000
1 7   1.0000000000000000E+000
2 7  -1.0000000000000000E+000
2 6   1.0000000000000000E+000
.
.
.
```

There are 6 lines with tags in the format <Title> etc to describe the content of the next line(s). They all need to be there even if there is no content.

The first word of the title (separated by whitespace) is taken as an ID code for the model, and the rest as a descriptive name. This ID is used to construct file names for the separated subnet specifications that Netsplitter generates. When .SBML files are stored, the title is also accordingly split to populate the model “id” and “name” tags in the sbml specification.

Following on the title section, the file contains 3 main parts: a listing of reaction names, a listing of metabolites, and the numerical matrix element values of the stoichiometry matrix. The matrix columns and rows are associated with reactions and metabolites in the order in which they were listed, but as long as this is kept consistent the ordering of names is arbirtrary.

The reaction names are unique identifiers and need to be listed in the same order as the columns of the stoichiometry matrix, and separated by tabs.
Reversible reactions, if any, are listed explicitly in the next section. Names in this section repeat the reaction names already listed in the previous one.

Metabolite identifiers as in the example above, consist of the metabolite ID (a character string without blanks), and its cellular compartment (with a space in between) and the combined identifiers separated by tabs. The two-part identifier is to provide for a naming convention where the same ID (e.g. WATER) is used in different compartments, so adding the compartment is necessary to make identifiers unique e.g. in a transport reaction. In the example above no compartmentalisation was taken into account, so all metabolites were assigned to a default “compartment”, SYSTEM.

However if a naming convention is used where the compartment is identified as part of the ID (e.g. WATER_c and WATER_m for water in the cytosol and mitochondria respectively) the separate compartment specifier can be omitted in the input file, as long as this is consistently done for all metabolites. Netsplitter does not use the compartment specification in its own calculations, although it does pay attention to these when its output is saved in SBML format.

Another common naming convention is to use a numbered metabolite ID such as C00123_m for the first part of the identifier and the second part for the chemical name.

Any embedded quotes are stripped out of names and ID's to avoid complications when saving SBML.

To distinguish between ID+compartment or ID+name conventions, Netsplitter inspects the second part and if there are many duplicates, it assumes that this is because they are compartment names; if they are mostly unique, it is assumed they are names.

So to summarise: the metabolite specifications can be in one of 3 forms: i) A metabolite ID only, containing no blanks; ii) A metabolite ID, then a blank, then a compartment ID; iii) A metabolite ID, then a blank, then a metabolite name. In the latter case, blank characters within the name should be handled correctly but are probably better avoided.

Name truncation. Note that the combined name (ID, ID+compartment or ID+name) is truncated for use by Netsplitter to 50 characters to keep printout formats reasonable. When exporting in the TSV format, these truncated names are stored, but in SBML export where ID's, names and compartments are stored separately no truncation is done.

As both ID's and compartment names are usually reasonably short it is only the chemical name that sometimes get truncated; this is flagged by a "#" character at the end of the name. As long as the ID remains unique, this is only cosmetic. However, in the rare case of a naming convention where the ID itself is very long, its uniqueness may be lost by truncation. Such cases are handled by discarding the name but taking the full ID without truncation. That might give an untidy printout and layout diagrams, but full details are still put in the subnet output files, at least in SBML format. This is usually acceptable, unless ID’s are absurdly long – e.g., sometimes a lot of extraneous information such as the database name, version and path are crammed into metabolite ID's – in which case the only remedy would be to edit the ID’s externally. Generally, Netsplitter tests for non-unique IDs and gives a warning message when it happens.

Listing some metabolites as external in the S-matrix file is optional. This possibility is mainly used by Netsplitter itself so an output file produced in one run of the program can be further split up in a subsequent run. Normally, all metabolites would be listed in the .TSV input file as internal and the externals section left empty (but its tag must still appear). Note that the internal and external metabolite lists do not overlap (unlike the reversible reaction list, which is a subset of the full reaction list).

Everything following on the <Stoichiometry> tag is in standard MatrixMarket (.MTX) format for a sparse matrix. The ordering of columns and rows of the matrix must be the same as the order in which reactions and metabolites were listed respectively.

5.1.1 Converting file formats.

As Netsplitter can read the Stoichiometry input file in one format and save it in the other, it can be used to convert the input files from TSV to SBML or vice versa.

A button is available on the Control Panel to perform this function. When clicked, a new file is created with "converted" appended to the file name and with the appropriate file extension. The modified file name gives some protection that an alternative input file from another source is not overwritten inadvertently.

To ensure that the converted file has the same content as the input stoichiometry file, make sure that a) no flux file is nominated and b) the "Omit reactions with ID containing" text box is empty. Otherwise the file that is saved, will incorporate the allocation of reaction directions and reversibility based on the flux file, and will also exclude any reactions eliminated according to ID criteria. Even if an external metabolite file is specified, its content is not taken into account in the converted file.

Some editing of metabolite and reaction names are done when a file is loaded, such as removal of extraneous blanks and quote characters, and truncation of long names (see above). These changes may also appear in the converted file. Note, however, that Netsplitter does not check that names and ID's conform to SBML conventions, such as not to start with a number and not to contain special characters.

When converting an SBML file to TSV, quite a lot of information may be lost as the TSV only provides for the basics of specifying metabolites and whether they are external, reactions and whether they are reversible, and the stoichiometry matrix. Conversion from TSV to SBML will correspondingly only create a very basic SBML file.

5.2 Default external metabolites

Provision of this file is optional but allows fine-tuning of the preliminary set of recognised externals beyond what is achieved by simply setting the connectivity threshold value, and can thus improve the recognition of block structure. Occasionally identifying one particular external metabolite to Netsplitter by means of this file may be crucial to get the process started.

A secondary function of this file is that provides a way to resume a previous calculation - see below.

It is a simple text file, with one metabolite name per line; blank lines are ignored and "% " and anything following it on the line is considered a comment and ignored. The exception to this is that when Netsplitter creates this file, it uses such comment lines to record the history of a program run and can also read these particular lines. However, there is no need to include these "history" lines in a user-created external metabolite file.

At the end is a special, optional section prefaced by a line reading "<Refusals>"; see below for a description.

Example :

```
%%%%%%%%% Environment molecules and ions %%%%%%%%%%%%%%%%%%%%%%%%%%
WATER
OXYGEN-MOLECULE
CARBON-DIOXIDE
PROTON
E-
NITRITE
NITRATE
SUPER-OXIDE             % O2-
S2O3
```

```
SO3
SULFATE                 % SO4
Pi                      % inorganic phosphate

      % Molecules assumed ubiquitous, e.g. because involved in many
reactions
      % or in important biological functions
HYDROGEN-PEROXIDE
HCO3
PPI                     % pyrophosphate
AMMONIA
AMMONIUM
CHOLINE           % fundamental metabolite for fatty acid metabolism,
                  % synthesised by several pathways
CHORISMATE  % raw material for aromatic amino acids
ISOCHORISMATE     % links to menaquinone and phylloquinone synthesis,
                  % hence to thiamin, chlorophyl synth etc;
%%%%%%%%% From catabolism %%%%%%
            % Sugar phosphates
DIHYDROXY-ACETONE-PHOSPHATE         % Triose
GAP                                 % Triose
ERYTHROSE-4P                        % Tetrose
RIBOSE-1P                           % Pentose
RIBOSE-5P                           % Pentose
RIBULOSE-5P                         % Pentose
L-RIBULOSE-5P                       % Pentose
XYLULOSE-5-PHOSPHATE                % Pentose
FRUCTOSE-6P                         % Hexose
FRUCTOSE-16-DIPHOSPHATE             % Hexose
GALACTOSE-1P                        % Hexose
MANNOSE-1P                          % Hexose
MANNOSE-6P                          % Hexose
GLC-1-P                             % Hexose
GLC-6-P                             % Hexose
ALPHA-GLC-6-P                       % Hexose
            % alpha-keto acids
PYRUVATE
OXALACETIC_ACID
MAL                     % Malate, an isomer of Oxalacetic acid
2-KETOGLUTARATE
PHOSPHO-ENOL-PYRUVATE   % pep
            % from photosynthesis, starch/sucrose degradation
GLC                           % glucose
ALPHA-GLUCOSE
            % from aerobic/anaerobic respiration, TCA cycle
SUC                           % succinate

%%%%%%%%%%%%%%%%% Carrier molecules
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ATP
ADP
AMP
GTP
GDP
GMP
Donor-H2
Acceptor
NADPH
NADP
NADH
NAD
```

```
FADH2
FAD
THF
METHYLENE-THF
S-ADENOSYLMETHIONINE
ADENOSYL-HOMO-CYS
UDP-GLUCOSE % metabolism of glycogen,starch,cellulose, many other
processes
CO-A
MALONYL-COA                  % raw material for fatty acid synthesis
ACETYL-COA
SUC-COA

<Refusals>
GLC
```

Note that the metabolite ID has to be identical to that used in the Stoichiometry input file. The compartment or name can optionally be included (Netsplitter does that when it creates this file). But only the ID is used when matching this list to the metabolites read in from the stoichiometry file. That avoids the need to enter long and perhaps truncated chemical names, especially in SBML input.

The downside is that in the ID+compartment naming convention, any compartment specified in the external metabolites file is ignored. So if there is both WATER CYTOSOL and WATER CHLOROPLAST they will both be taken external if the ID WATER is listed in the Externals file. If that is not desired, the only remedy is to change the ID's for this particular compound in the Stoichiometry file to e.g. WATER_c and WATER_p (keeping the explicit compartment specifier as well for consistency) and then specify the appropriate one in the Externals file.

CAUTION: If the External Metabolites section of the printout states "0 Metabolites proposed in the external file" despite the fact that an external metabolite file was specified in the control panel and this file is recorded in the header section of the printout, this can be a) because all the listed metabolites were accounted for under other categories or reincorporated OR b) this may be because the IDs were incompatible!

There is one important difference between the way externals listed in this "default externals" file and in the main input (S-matrix) file are handled by Netsplitter. Externals in the defaults file are taken as external for the blocking stage; but during the reincorporation stage (see below) they are individually inspected and any of these metabolites that have links to only one block are reincorporated into that block. However, externals from the stoichiometry file are not considered for reincorporation because they are assumed to have further links not explicitly listed in the S matrix. That could be the case, for example, if the input S matrix is in fact one obtained as a subnet in a previous calculation.

So the appropriate way to make sure that a particular metabolite is definitely taken as external is to list it in the stoichiometry file. But remember that if the stoichiometry file is in TSV format, and a metabolite is moved to the externals section, the corresponding matrix row has to be moved as well to maintain consistent ordering. This is not an issue for SBML input files.

The section headed by the tag <Refusals> (on its own line) lists any metabolites that are NOT to be taken as external. This is relevant in particular to automatically identified high connectivity metabolites, and specifying them here will override the automated selection as external. That has to be done with care, as in many cases it will make successful blocking impossible (which is why the automated selection of high connectivity metabolites as external was introduced in the first place).

Specifying a metabolite as a "refusal" will override any listing of it in the first section of this file. That can be used when experimenting with temporarily not taking a particular metabolite as external, but in that case it is probably better to simply comment it out by putting a "%" in front in the name in the first section. However, specifying a metabolite as a refusal will NOT override its specification as external in the main input (S-matrix) file; nor, obviously, if the metabolite is structurally external in the network by always occurring as a substrate (or a product).

5.3 Reaction Fluxes

This file is also optional. Its purpose is to allow reaction directions to be specified in accordance with an actual metabolic state. The reaction directions obtained from a standard database may be based on convention rather than what happens in an actual cell. Especially with reactions specified as "reversible", their actual direction will depend on metabolic conditions. As the presence of too many reversible reactions tend to obscure the blocking in a similar way as highly connected metabolites do, it is useful if as many of these as possible can be pinned down to a specific direction.

If a flux balance calculation on the network is available, its output flux values can be read in from this file. Netsplitter uses these values as follows:

If the flux of a specific reaction is positive, its direction is considered confirmed and it is removed from the list of reversible reactions (if it was there in the first place).

If the flux is negative, its direction is reversed (its column in the stoichiometry matrix multiplied by -1) and it is removed if on the reversible list as well.

If the flux is zero or the reaction is not listed in the flux file, it is left unaffected.

Note that even reactions that were not listed as reversible, will be reversed if a negative flux value was found.

If the option to duplicate reversible reactions was set, that is done only after reaction directions have been fixed based on the flux values.

The file can be supplied either as a simple spreadsheet (in .XLS , .DIF or .CSV format) or as a text file (.TXT or .TSV). It should contain two columns: the reaction ID's (as used in the main input) in the first, and the numerical flux value in the second column. Column headers are optional. Any further columns beyond the second may be present but are ignored. For XLS, this needs to be in the first worksheet, other worksheets are ignored. Reactions for which no flux information is available may be left out giving a file with fewer rows than the total number of reactions.

As Netsplitter only uses the sign of the flux values, this file can also be manually constructed to specify the desired reaction directions simply as notional values of -1, 0 or 1.

CAUTION: If subnet files are saved, they incorporate the reaction direction information gleaned from an input flux file. If such a subnet file is loaded back into Netsplitter in a subsequent run for further analysis, the flux file should not be selected in this subsequent run, because if it is, negative flux values will cause the corresponding reaction to be reversed yet again giving an incorrect subnet. Netsplitter gives a warning message about this, provided that the subnet files have not been renamed in the meantime.

5.4 Target metabolites

This file is optional, it merely allows the visual identification of metabolites of special interest in the display.

It contains one metabolite ID per line, with %-comments, and the same comments stated for the external metabolites file apply here regarding naming conventions.

Example:

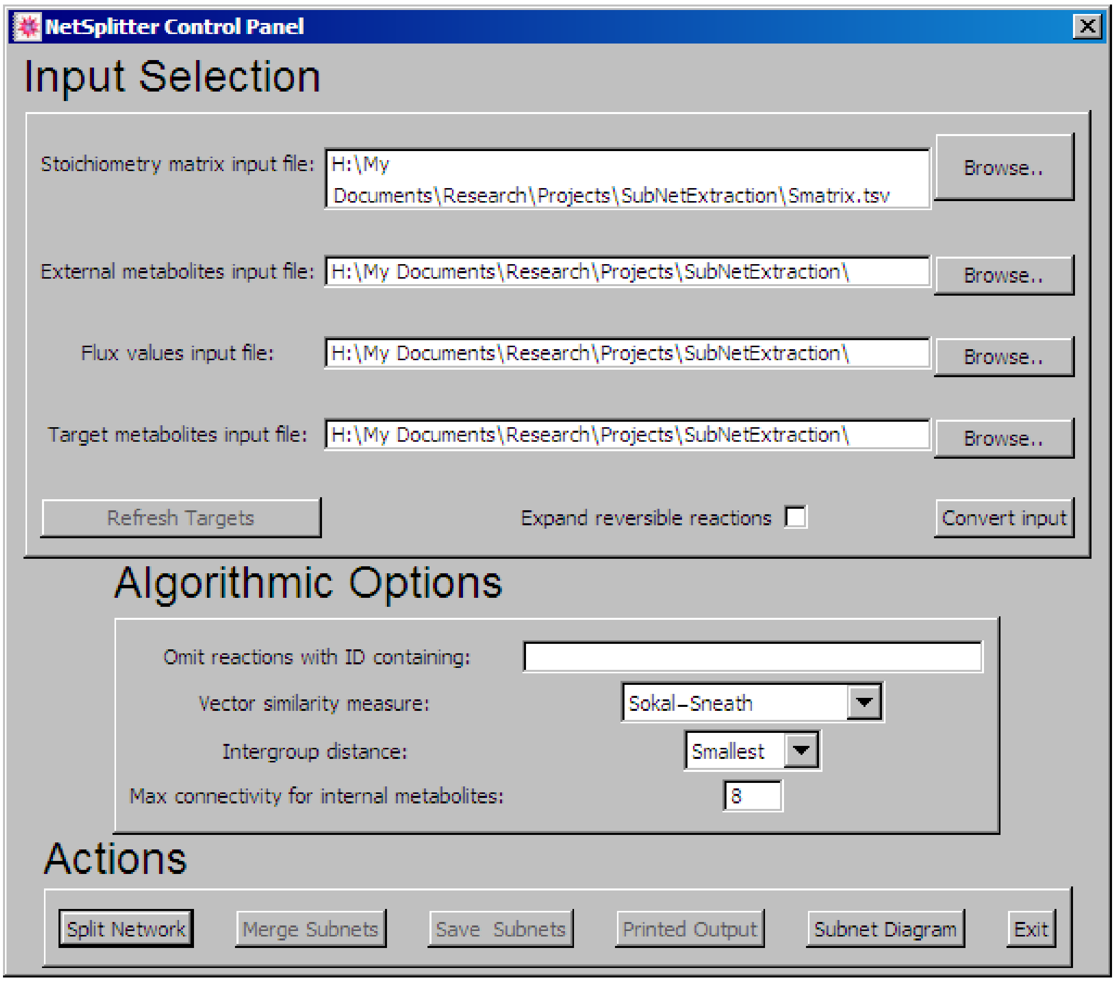
```
%%%%%%%%% metabolites of interest %%%%%%%%%%%%%%%%%%%%%%%%%%

APIGENIN                %% Flavonoids
NARINGENIN              %% Flavonoids
LEUCOPELARGONIDIN-CMPD  %% Flavonoids
CPD1F-90
```

Several target metabolite files may be created e.g. to study different biochemical functions. A new target file can be loaded and displayed during the course of a calculation.

6 Interacting with Netsplitter

6.1 Netsplitter Control Panel

To run the program, open Netsplitter.nb in *Mathematica* or *Mathematica Player Pro.* Then either select Evaluation/Evaluate Notebook from the *Mathematica* menu, or select the contents of the notebook by clicking on the far right cell grouping bracket (or even just the cell bracket for the Control Panel cell group) and press <Shift> <Enter>. If a dialog about initialization cells appears, choose "Yes". The Netsplitter Control Panel will open.



Stoichiometry matrix input file: H:\My Documents\Research\Projects\SubNetExtraction\Smatrix.tsv

External metabolites input file: H:\My Documents\Research\Projects\SubNetExtraction\

Flux values input file: H:\My Documents\Research\Projects\SubNetExtraction\

Target metabolites input file: H:\My Documents\Research\Projects\SubNetExtraction\

Expand reversible reactions □

Omit reactions with ID containing:

Vector similarity measure: Sokal–Sneath

Intergroup distance: Smallest

Max connectivity for internal metabolites: 8

The first step on the control panel is to specify the locations of the four input files in the top panel. With the optional files, the default settings may be left as they are if the corresponding files do not exist; the program will proceed without them.

The "Refresh targets" button is provided to allow the user to load a different set of target metabolites after completing the blocking and can be ignored at first - see below.

The tick box "Expand reversible reactions" allows the user to choose whether both directions of reversible reactions should be included in the calculation.

When this box is ticked, Netsplitter expands each reversible reaction by adding a duplicate to the stoichiometry matrix in the opposite direction. This most accurately reflects the network specification, but it is found that networks with many reversed reactions do not separate into subnets as successfully as those with only a few. That is simply because all the reverse directions create many more alternative pathways through the network, making it more highly connected.

In principle any reaction can only run in one direction under particular metabolic conditions. Which direction this is, might be known from the calculated fluxes, or from the directions required by known pathways, and if the reaction specifications are made reflecting this a better subnet separation is usually achieved by just taking all reactions in the directions explicitly specified.

That is done by unticking the expansion box on the control panel in which case only the explicit reaction as represented by the stoichiometry matrix (possibly modified by an input flux file) is included. This generally gives the best blocking and is the default. It may be worth trying the separation both with and without this option.

The “Convert input” button does no calculations, merely producing an alternative input file as described in section 3.1.1.

Step 2 is to modify the configuration settings on the middle block of the control panel, if desired. For most purposes the default settings should be fine. The available options are:

1. A group of reactions, identified by a substring in the reaction ID, may be deleted from the stoichiometry matrix that was loaded. This option is provided mainly to get rid of pseudo-reactions such as biomass constituents and growth rates, which are not properly part of the metabolic network. For this purpose, if these "reactions" are identified e.g. by identifiers like BIO123456, they can be deleted by filling in the string BIO in the box provided. All metabolites that become unconnected because of deleted reactions, are automatically deleted from the S matrix as well. Several sets of reactions can be eliminated by filling in a string like BIO|Bc . Any number of substrings, separated by vertical strokes "|" (meaning "or") can be specified. Strings are entered without quotes and are case sensitive. This facility may also be used to e.g. experiment with a network not including intercompartment transport, provided that transport reactions can be identified by a common substring in their ID's. Eliminating individual reactions by giving their full ID's should also work although it may be tedious.

2. There is a choice between the Sokal-Sneath (default), Dice and Jaccard measures for comparing the similarity of binary vectors. This is used to group similar rows and columns in the DAG matrix together in order to produce blocks.

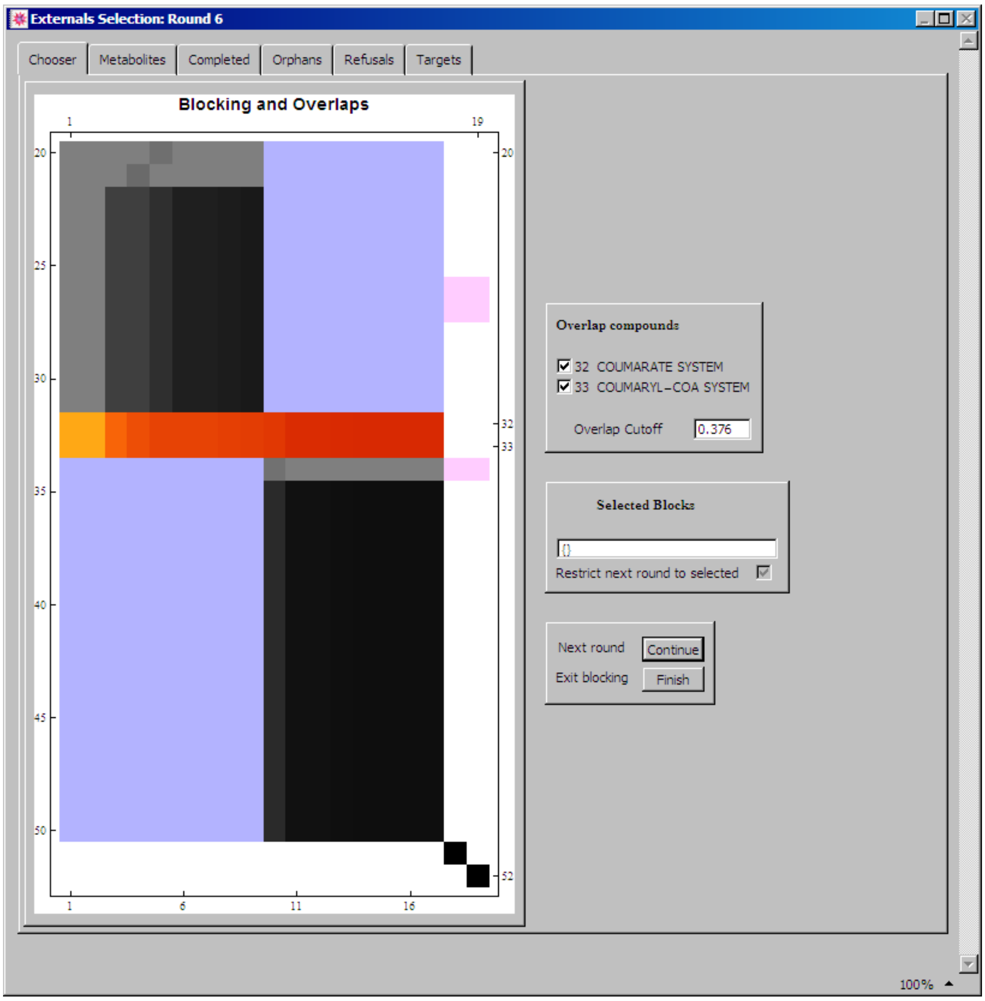
3. In the hierarchical grouping, the method used to calculate the distance between groups of vectors (rather than between individual vectors) can be selected. The default, *single*, takes the smallest separation between vectors belonging to different groups as the group separation. Several other options are available.

4. The minimum reaction participation (connectivity) of a metabolite to be automatically taken as external, can be specified. The default value of 8 seems OK for networks with between 100 and 2000 metabolites and reactions, but could be adjusted otherwise.

The input files will be read and processed once the user clicks the "Split network" button on the bottom of the control panel. After the initial processing, this will display a dialog for selecting external metabolites.

6.2 Selecting externals

The selection dialog is a tabbed panel. The main panel is the **Chooser** tab, which displays a blocking representation of the current DAG matrix on the left hand side. It only shows a submatrix, consisting of sink metabolite columns and source metabolite rows.



Overlap compounds

☑ 32 COUMARATE SYSTEM
☑ 33 COUMARYL-COA SYSTEM

Overlap Cutoff 0.376

Selected Blocks

{}

Restrict next round to selected ☑

Next round Continue

Exit blocking Finish

Matrix elements are displayed not as numbers, but as cells in gray shades. Where all non-zero elements in a row and column of the DAG matrix belong to a single row or column group as determined from the hierarchical grouping (as would be true for perfect blocking) the element is displayed in black. Where there is probability leakage out of the group this is shown by corresponding gray shades: dark gray in the group that contains most of the probability, light gray showing where it has leaked to. Also, the display is a superposition of this leakage analysis applied to rows and columns separately, so a medium gray would result if there is conflicting evidence from rows and columns about whether the probability is localised to within one group. This matrix that expresses both the grouping and deviations from grouping that has been achieved for the DAG, is called a *blocking matrix*.

The net result is that a perfectly blocked matrix will display as pure rectangular black blocks, while with imperfect blocking will show both the dark areas where the strongest and most consistent grouping is found, as well as gray areas that indicate block overlap, and the actual network nodes (metabolites) causing such overlap or inconsistencies.

Netsplitter inspects these grey areas and computes the smallest set of rows and columns that will, between them, cover the light gray cells in the blocking matrix; eliminating these as externals are taken as the most likely to decouple the partially formed blocks. The corresponding rows and columns are displayed in colour, in a "temperature" colour scale where yellow represent high values and red low values. Also, on the right of the matrix display, all metabolites selected in this way are listed, each with a tick box to control whether it will be made external in the next round of calculation. A metabolite can be related to its individual row or column either by its displayed number, or by using the tick boxes to toggle them on/off.

Netsplitter also shows all matrix blocks that it recognises as non-overlapping. These are indicated with a blue background. In the case of a fully completed block the blue background may not be visible because it is overlayed in black, but it is still recognised for computation by the program. This tends to happen especially with small single column or - row blocks which appear particularly during early stages of the splitting.

Another item visble on the matrix display, is the position of any target metabolites Netsplitter was given. The rows or columns that belong to those are highlighted by a pink background outside the range of identified blocks. In the initial stages, when the entire matrix may be a single block, these pink stripes may be invisible, but they become more prominent as block splitting proceeds.

The user can select one or more blocks by clicking on them. The background colour changes to green, and the block number (starting from top left) shown in an editable text field on the right of the matrix display. Selecting blocks are useful for two purposes:

1. Just below the selected block listing box, one may choose to continue further rounds of externals selection with further processing of only the selected blocks. In this way one can avoid fragmenting blocks that are already small enough while still reducing other, larger blocks. When this is done, the unselected blocks will still be converted to subnets, they will just not be split up any further. Similarly, Netsplitter will automatically classify any single

column or single row blocks as "completed", as they can obviously not be further reduced.

2. To make intelligent decisions about which metabolites should be made external and which blocks should be further split, one usually needs more information about which metabolites are contained in a block. To this end, several metabolite listings are provided as separate tab panels that can be selected while in the selection dialog. In particular, the first of these , **Metabolites**, lists the sinks and sources, with their numbering, to relate them to the matrix display. The metabolite names belonging to any blocks that are currently selected, are highlighted by a green background in this listing allowing easy identification.

Other tabs that are available show the following:

- **Completed** - metabolites in blocks already completed in previous rounds (see above) do not appear in the current matrix, so are listed separately here.
- **Orphans** - sometimes when a network is "cut" at a certain node, some individual metabolite nodes get detached from the network. For efficiency these "scraps" are collected into a single block of "orphans" rather than making each a separate little block by itself.
- **Refusals** - whenever the user refuses a certain metabolite proposed by Netsplitter as an external, it is put on this list so it will not be offered again.
- **Targets** - the target metabolites (if any) supplied as an input file.

A final item that the user can adjust on the Chooser tab is a cutoff value that determines how many metabolites will be offered as candidate externals. Normally this cutoff is dynamically calculated by the program in order to give a sensible number of alternatives, but this option allows one to adjust the value used in the next round. However, as any candidates that are accepted will be gone in the next round, while those that are rejected stay rejected, this only gives limited control and it is probably best to leave this value alone.

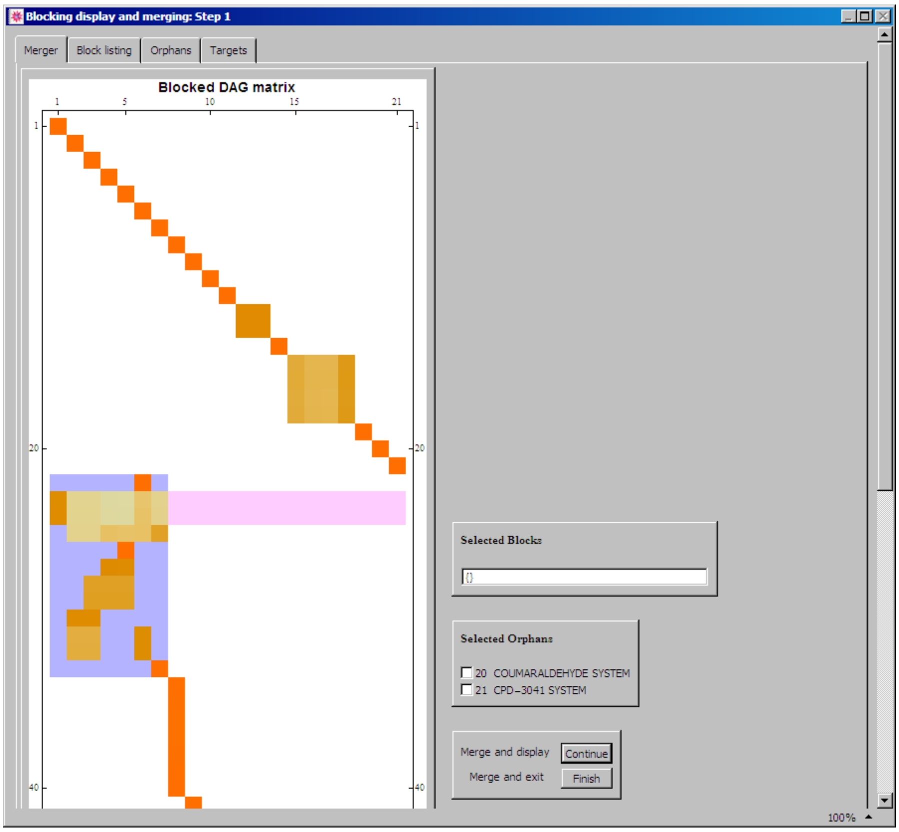
As a general remark, it seems best not to reject the proposed candidates without a strong reason. The situation often encountered is that rejecting one metabolite in a certain round, causes a plethora of alternatives being offered in the next round in order to cover the same set of light grey cells in the matrix. Furthermore, even if a metabolite is chosen as external in this dialog, that does not mean that it is irrevocably external. Once block splitting is finished, Netsplitter performs a further analysis and reincorporates externals where they turn out not to be essential e.g. because another external selected later does the same job. In practice only a fraction of the working set of externals survive the reincorporation. So it is usually best to let Netsplitter run through the process as it wishes at least once, and only if the final allocation seems unacceptable, the program is re-run and the offending metabolite refused.

Having chosen externals, the user has two choices for proceeding. Going into the next round means that all the externals as chosen are eliminated, the blocking calculation is repeated and the user presented with the results and a new set of candidates. This can be repeated as many times as desired, until the user is satisfied that desired degree of splitting has been achieved, or sometimes when Netsplitter cannot find any more candidate externals.

Once the user is satisfied, clicking the "exit" button returns to the Netsplitter Control Panel. Even if there were some candidates that were ticked as future externals, these are not processed when the "exit" button is used. Rather, the previously mentioned reincorporation is performed and the control panel displayed.

6.3 Merging subnets

The next available operation on the control panel is to merge subnets. The idea behind this is that small subnets can always be merged into bigger ones. It sometimes happens that the final collection of blocks is too fragmented for the intended purpose. As described above, the user can exert some control over this during the splitting process, but especially with small fragments such as automatically split off single row or column blocks, it may for example happen that the target metabolites end up spread over several blocks. In this case, the "Merge" dialog allows the user to select any blocks as before by clicking or entering numbers in an editable text box, and have them merged together. In addition, any orphan metabolites that are needed can be specified to be merged in with a block or set of blocks. If this is done, it is in principle possible that the resulting subnet may contain that as a disjoint piece, although mostly it will become linked by common externals to the block.



Selected Blocks

{}

Selected Orphans

□ 20 COUMARALDEHYDE SYSTEM
□ 21 CPD−3041 SYSTEM

The Merge function is in fact also handy just for inspection of the final DAG matrix, even if no merging is done. Its display is similar to that of the Chooser, but it shows the actual DAG rather than its derived blocking matrix. It also shows the result of the reincorporation step, so blocks are usually larger than the corresponding ones in the blocking matrix of the last round that was done in the Chooser. Finally, it also shows the "top" part of the DAG, i.e. the Sinks x Sinks submatrix. This is usually mostly a trivial diagonal matrix, which reflects the "stay put" part of the random walks, but will sometimes show the presence of a combined sink as a small square diagonal submatrix.

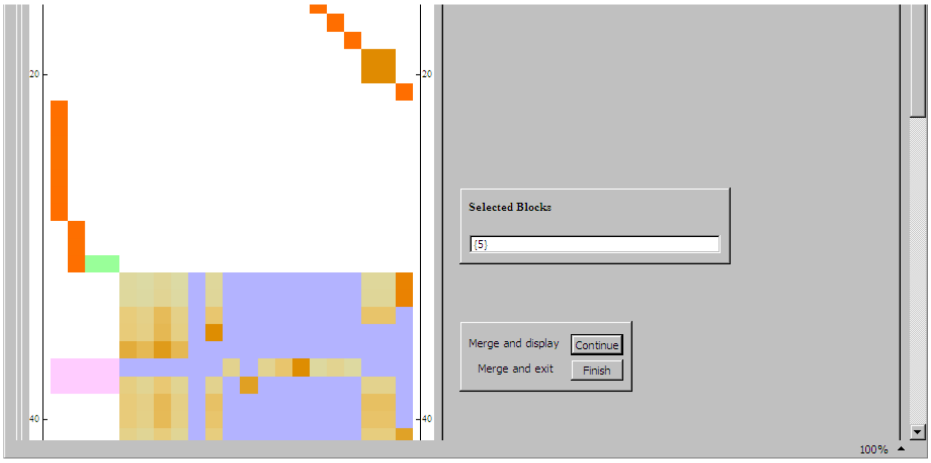
It is also possible to merge just a set of orphans with each other into a new block. In principle, an orphan just represents a very small subnet with only one internal metabolite; i.e., it typically consists of two reactions with one or more input externals, the internal metabolite, and one or more output external metabolites.

The reason that these have to be treated specially, is that the single internal metabolite is always classified as a sink, so the corresponding "block" has no source node and so cannot be properly represented on the (lower) Sink-Source part of the DAG matrix that is used for selecting externals etc.

In the Merge dialog, the collective orphan block is not shown directly but is indicated by the band that remains empty at the right of the diagonally arranged blue blocks. One could picture this as a zero-width blue line extending horizontally across the very bottom of this empty band.

If a specific orphan block is created by merging, this is shown as a quasi-block (blue line) that appears to overlap with the last row of the previous block in the DAG. Conceptually, such a block is again more like a horizontal line covering the set of sinks that it contains, but with zero width in the vertical (source) direction. So the overlap is merely a visual aid to make it visible. For the purpose of selecting such a block by clicking, the row containing the overlap is split in two: the top part is taken to belong to the proper block on the left, and the lower half of the line to the merged orphan block. So by careful positioning of the mouse pointer one or the other can be selected despite the visual overlap. But for large networks where the screen width of a matrix row is very small, it is more practical to select the merged orphan block by manually entering its number into the selection list box.

The figure below shows an example of an orphan block that was first created by merging, and then selected so it is highlighted by appearing green.



Selected Blocks

{5}

In performing a merge operation, Netsplitter first merges the lists of internal metabolites, and then inspects all externals and orphans to reincorporate all those that are not connected to any remaining blocks. The merging process is the complete inverse of the splitting, and allows a targeted assembly of exactly those parts of a network that are relevant for a particular study.

An issue arises sometimes with the rendering of the DAG matrix for large networks. If a dimension of the matrix exceeds the pixel resolution allowed internally for its display, *Mathematica* resamples the image and discrepancies such as apparent small overlaps between blocks or inaccurate colouring around the edges can appear. The same happens in the printout as well. This is merely a display issue; the underlying identification of blocks and overlaps is done by the program using the numerical DAG elements and remains accurate.

The location of target metabolites (if loaded) are also displayed on the Merger to allow the identification of particular blocks that are involved with a function of interest.

If the Merge function is used in this way for display purposes, it is sometimes useful to change to a different set of target metabolites without having to repeat the calculation. To do this, exit Merger, then simply select a different "Targets" file on the control panel and click the "Refresh Targets" button before clicking the Merge button once more.

The tabs available in the Merger dialogue are largely similar to those of the Chooser but on the "Metabolites" tab the metabolite list is formatted by block, with sinks listed first in each block.

Merging can be done in stages, by using the "Merge and Display" button, or at one stroke using the "Merge and exit" button. If no blocks were selected for the merge, no changes are made to the blocks as is the case when this dialog was used for inspection purpose only. One can exit the Merge dialog, e.g. to look at a subnet diagram, and then return to it to do a further merge. However, note that a merge operation cannot be undone; the complete splitting operation may have to be redone if a merge turns out to be unsatisfactory. This can be alleviated by saving the state of a calculation as described in the next section.

When blocks are merged, the new block is placed at the end of the list and given a new number. The constituent blocks are emptied but not deleted altogether, in order to preserve the numbering of other blocks in the matrix. This is done to facilitate performing a series of merge operations without having to re-identify the blocks involved. Also, that preserves consistent numbering with subnet diagrams that were printed out previously. So after a merge operation, numbers of constituent blocks become inactive and are no longer available or shown on printouts. So the sequence of actively allocated block numbers can contain gaps and will extend beyond the total number of actual blocks that are present.

6.4 Blocking efficacy

As a general guide to how successful the current blocking is, Netsplitter displays an *efficacy* index *E* as a percentage, on the selection and merging dialogues as well as in the printout. The idea behind this is that (as shown in the references) under the quite general assumption that the effort in interpreting a network increases as a concave up function of the number N of internal metabolites in a network, the maximal simplification is achieved if there are $\sqrt{N}$ subnets each with identical size $\sqrt{N}$. To calculate a numerical value, a simple power law N^p is assumed as an explicit concave functional form and this leads to the formula

$$E = 100\frac{Log\left[N^p\right] - Log\left[k^p + \frac{1}{k}\sum_{i=1}^{k} n_i^{\,p}\right]}{Log\left[N^p\right] - Log\left[2N^{\frac{1}{2}p}\right]}$$

for a subnet partitioning into k subnets with internal node counts n_i respectively. This expression can be interpreted as the percentage (on a logarithmic graph) of the simplification that is theoretically possible, that has been achieved in a particular subnet split. A value of 0% is obtained for the original network of size N as well as in the limit of complete fragmentation into N subnets of size 1 each, because in that case the metanet is again of size N. The optimal equal distribution of $\sqrt{N}$ subnets gives E = 100%.

Values of E are not very sensitive to the value of the power exponent p. Increasing p mainly increases its sensitivity to the presence of a large monolithic block. As this tends to be a problem for large networks, p- values of 6 to 8 are found to work well for large networks, while p = 2 is adequate for small ones. As implemented the value is adjusted to the network size by the formula

$$p = 0.25\sqrt{N}$$

This calibration formula appears in the configuration file “Netsplitter.m” mentioned above and can be edited if required.

In interpreting E values, it should be borne in mind that the formula is most sensitive near the optimum. Experience shows that E-values of 10 – 20% are obtained even by just splitting a few orphans from a largely monolithic network and is more or less insignificant. Also, to double the efficacy from 40% to 80% requires a six-fold improvement in the number of subnets relative to the optimal number. Splitting alone can achieve values of 80% or higher in small networks, but in large ones this may be as low as 50% due to increasing fragmentation.

The merging operation is necessary to improve on that and the calculated E is quite a useful guideline in the merging process. Not all merges are beneficial; reducing the number of small subnets usually increases E but an increase in the subnet size may have the opposite effect.

The effect of the reincorporation step can also go either way. Firstly, it adds to the total number of internal metabolites so can affect the optimal subnet size, and if these happen to be added to subnets that are already larger than optimal it will lower the efficacy. The values shown in the selection dialog are before reincorporation, while those in the merging dialog include reincorporation. The printout shows the final values of both of these.

The calculated efficacy is always referenced to the current number of internal metabolites. Therefore it may happen that among two splits with identical E values, one may still be better than the other because it retained more internal metabolites in total. More generally, the efficacy score should only be seen as an overall indicator and many other considerations may make a specific partitioning appropriate for a particular purpose even if the *E*-value is not particularly high.

6.5 Saving subnets

This action allows the user to save the calculated subnetworks in either of two formats. This function creates (and overwrites an existing) subdirectory “Subnetworks” in the same filepath where the input S-matrix file was found, and inside it creates a separate file for each subnetwork. The user can choose either :-

- Text - in this case a set of .TSV files are created in the same format described for input to the program. Any of these files can in a subsequent run of the program be chosen as its input, allowing a medium size subnet to be split up further. In this case, all metabolites that were needed as external to separate the medium size network from the original large network is carried forward into the new calculation as external. That is important because it may happen that even though a particular metabolite appears to be an intermediate between internal metabolites in the medium size network and therefore itself internal, it may have had external connections in the large one. So it is not subject to mass balance in the medium network and should therefore be kept as external.

- SBML – If the input was from a .TSV file, the subnetwork files are exported as a relatively basic SBML file that should be suitable for input to standard flux balance software such as CellNetAnalyzer, YANA, etc. If the input came from an SBML file, Netsplitter attempts to transmit all pertinent specifications such as unit definitions etc. to each subnet even if this was not used internally. In particular, full reaction specifications (including kinetics) are passed through to the subnets, only the reversibility may be changed based on the results from a flux file if that was loaded.

Whichever option is chosen, a text file called ExternalMetabolites.txt, listing the complete set of externals, is saved along with the subnet files in the same directory, to enable resumption of the current calculation in a later session, as described below. This file also contains comment lines summarising the choices made interactively in the run that created the saved subnets. These lines are partly for the information of the user, to record the history of the run that produced the subnet files. Additional comment lines can be added manually if required.

Each subnet file as saved, of course only contains reactions, internal and external metabolites for that subnet. It may be noticed that no reactions that only involve metabolites that are external for that subnet, are inlcuded. For example, if both CHORISMATE and ISOCHORISMATE are external, the isomerization reaction between them is not included; similarly for transport reactions between compartments e.g if WATER_c and WATER_m are both external. The same is true for subnet diagrams discussed below.

This is a feature, not a bug! It may appear that this has the effect that links in some well-known pathways go missing. However, as each external metabolite has (by definition) its own infinite reservoir, any sequence of reactions that includes such a link between externals, can still proceed in the subnet. E.g. delivering CHORISMATE to its reservoir while consuming ISOCHORISMATE from its own, is equivalent to a reaction that converts CHORISMATE to ISOCHORISMATE. So explicit inclusion of this reaction would not make sense in the subnet.

6.5.1 Resuming a previous calculation

Whenever the subnet files are saved by Netsplitter, a copy of the external metabolites file that reflects the current state of a calculation is automatically saved as well. One purpose of this file is to record how the subnet files in the same directory were constructed for the information of the user.

But a second purpose is to avoid having to redo a lengthy blocking calculation e.g. to experiment with different merges of blocks, or to generate subnet diagrams. One can simply enter the path of the saved externals file in the Subnetworks directory, in the appropriate box of the Control Panel. The provision of the "refusals" section in the externals file is mainly designed to make this possible, so that information about interactive refusals done in the previous session is conserved.

To make resumption possible in a future run, one must remember to use the "Save subnets" button to store the external metabolites file (along with all subnets).

When Netsplitter detects a "history" section in the Externals file, it loads all program configuration options from this file, overriding the options set either by default or by the user in the Control Panel. If this is not desired, the corresponding option line (or the whole configuration section) can be deleted from the Externals file.

It then presents the user with the choice whether to implement the merge history that was loaded from the file in the new run.

If the choice is "No", only the blocking part of the previous run is repeated, and the user can either continue with this and choose further external metabolites, or simply exit the external selection dialog in order to inspect subnets, or perhaps try an alternative merge sequence.

If the choice is "Yes", the blocking is repeated and the previous merge sequence (or the first part of it, up to a cutoff point the user can select) is directly applied to it. In this case the external selection dialogue is not entered at all since the loaded merge sequence would not be valid any more if the user makes any change to the blocks.

CAUTION: In each merge step, the numbering of metabolites usually changes. As a result, the numbering of orphans that appear in the Merge History (also in the history in the printout) reflect the number applicable at that step and will not necessarily correspond to the numbers listed after subsequent steps, such as in the orphan listing in the prinout that reflects the final state. It can therefore also happen that the same orphan number appears in different steps of the Merge History; if so they refer to different metabolites.

6.6 Producing a printout.

The results of the calculation may be displayed, printed out directly or saved on disk in a variety of formats by using the “Printed Output” button on the Control Panel.

The user can choose between PDF, Postscript and *Mathematica* notebook formats.

Before actually printing or exporting, the output file is displayed on screen and contains a record of input files used, computing time, etc. It shows the final DAG in the same form as the Merger, and lists internal metabolites, formatted by block.

Information uniquely available in the printout is that it shows a listing of external metabolites classified into various categories according to the reason for taking them as external. Also, it compares the interactive choices (and refusals) with the final list of externals after the reincorporation step. The externals are also listed block by block, classified according to whether they are pure inputs or outputs of the full network, or represent crossflows that are either exchanged between subnets or with the environment. Finally all interactive choices made during the run are listed.

On screen, it is possible to scroll through this printout even though a dialogue asking for where the printout should go is displayed. It is also possible to exit this dialogue without actually printing or exporting anything. So this action can also be useful simply for inspection purposes. However, it may be hard to read some of the text on screen as it is formatted to produce a compact printout even for large networks. This can be mostly overcome by use of the zoom% box at the lower right of the window.

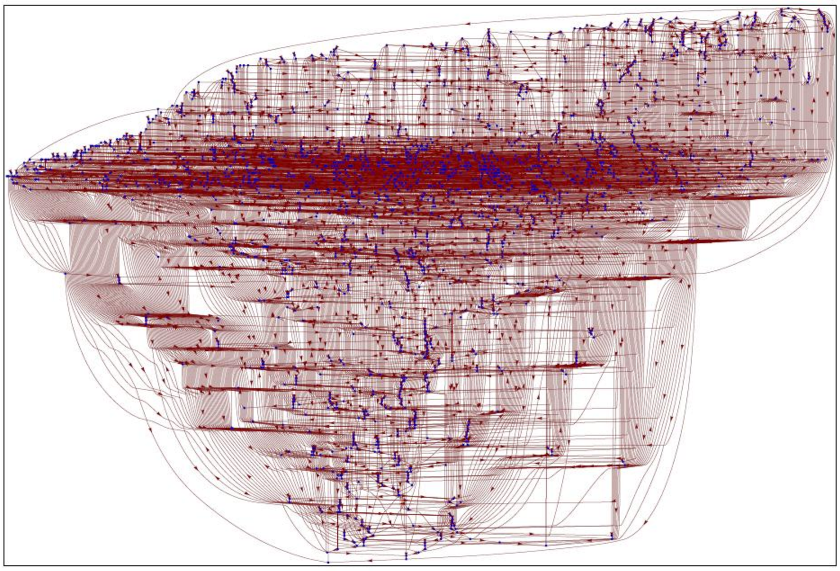
The user also has the option of leaving the printout window on screen, e.g. so it can be compared with the output from a subsequent run without wasting paper.

The results from the Merger, saving and printing actions always reflect the outcomes of the Split Network action. Having completed a run for a particular network, another input file may be chosen on the Netsplitter Control Panel, but e.g. the printout will still reflect the old results until Split Network is executed on the new data.

6.7 Network layout: Metanet and subnets.

An automated layout of the network structure as a bipartite network is available by clicking the “Subnetwork diagram” button on the Control Panel..

If this action is chosen before the splitting has been done, the structure of the full network is shown in a layered representation, in which horizontal layers alternate between reactions and metabolites. An example appears below. As this is usually a very large and complex network, labels are omitted. Also, high connectivity metabolite nodes are left out completely to reduce excessive clutter. This graph is mostly a curiosity, giving some sort of baseline against which the utility of breaking the network into subnets can be compared. But it can also be useful to see if the input network is fully connected. For large networks, it can take quite long to be generated (10 minutes or more for a network of 1800 metabolites x 1700 reactions, on a dual processor PC).

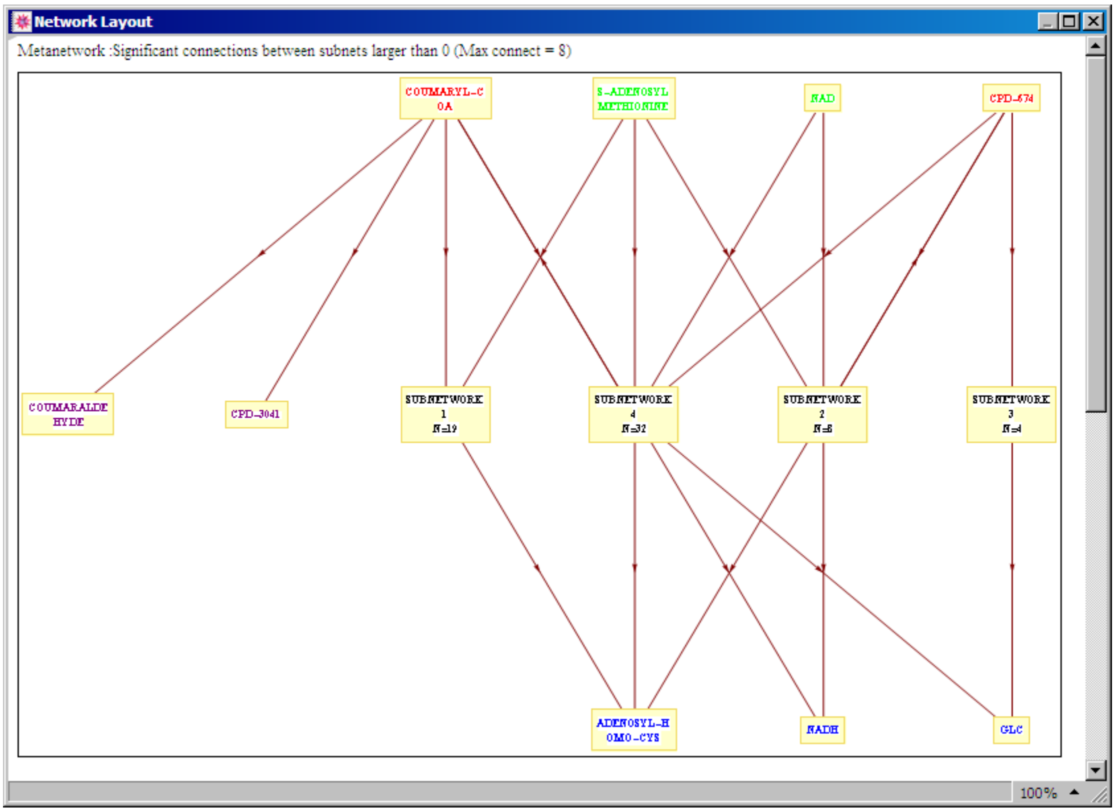


Once the network has been split, one subnetwork can be processed at a time, as chosen from a dropdown menu. The display includes detailed labels and color coding of node labels as explained below. The user can supply a title to print at the top of the network diagram. Output is to the screen, but can be printed or saved to PDF or Postscript files. Block numbers 1 to *N* are as shown e.g. in the merging display and in the printout. Two additional layouts are available: the metanet and the collection of orphan network fragments..

Before displaying the network, the user can choose a maximal metabolite connectivity to be included. The default value is the one set in the main control panel for the initial identification of externals. However, a larger or smaller value can be chosen as appropriate for the complexity of the subnet.

6.7.1 The meta-network.

This displays an overview of the complete network, in which all subnetworks are contracted to meta-reactions. Only metabolites that connect subnets either as shared



Metanetwork :Significant connections between subnets larger than 0 (Max connect = 8) inputs or outputs, or by being passed from one to another are shown. Also, the high connectivity metabolites are not shown even where they connect subnets. This can cause subnets to appear unconnected in the meta-network diagram.

Notice in the example above how the nodes are displayed in horizontal layers, which alternate between a layer of metabolite nodes followed by a layer of subnet nodes. Colour coding, detailed below, is used to make finer distinctions between node types.

The display can also be simplified by giving a threshold value that suppresses subnets smaller (i.e. with fewer internal metabolites) than the threshold. Any value between 1 and the size of the second largest block can be chosen instead. There have to be at least 2 subnets for the meta network to be drawn. With a threshold of 0 or1, "orphan" metabolites (i.e., subnets with a single internal metabolite) are shown if they share external metabolites with any subnets on the meta network. They can be thought of as subnets of size 1, but labelled with the name of their internal metabolite instead of a block number. Where two orphans only share externals with each other, that is not shown on the metanet because such sharing is shown in the orphan layout.

If a subnet appears on the meta-network as an isloated box, that means that it is only connected to other subnets by high connectivity metabolites (which are not displayed). In the case of the orphans, all those that do not connect to subnets of size 2 or more, are collected together as a single node on the meta-net. Their details can be inspected using the orphan layout option.

The connections between subnets are colour coded as described below, allowing subnets that share common inputs (green) or outputs (blue) to be identified. Crossflows, i.e. exchanges between subnets, are coloured red. The fourth type of connection - those coloured cyan - needs further explanation. These indicate situations where the external metabolites of one subnet overlap with internal metabolites of another.

This type of overlap may be unexpected. At the level of a simple graph of metabolite nodes, where the blocking procedure is carried out, there is a strict distinction between internal and external metabolites; they form non-overlapping sets. However, when translated back to the underlying bipartite graph representation, cutting all metabolite nodes that were identified as external, can sometimes still leave subnets connected by a shared reaction node. A typical case is where this reaction node has one input reactant from each subnet; because of the directions of the input links this does not allow probability flow through the reaction so that the two subnets are disconnected as far as the probability matrix representation is concerned. But then, after the blocking procedure, the external metabolites of each subnet are found by collecting metabolites that participate in all the reactions in which each internal metabolite is involved. In the case of the shared reaction, that will include an internal metabolite of the other block. Usually there will also be a complementary link where an external of the other block connects to an internal of the current block. So cyan nodes usually come in pairs connecting the same two blocks; but if one of those blocks is smaller than the block display threshold it may not be visible.

Note that the arrows shown on this type of internal-external overlap connection between blocks is not a reaction direction as such: because the reaction node is not shown on the metanet diagram, it is not possible to assign a direction for this link unambiguously. It would have been better to show these links without any arrow at all, but that is not allowed by the network drawing routines used.

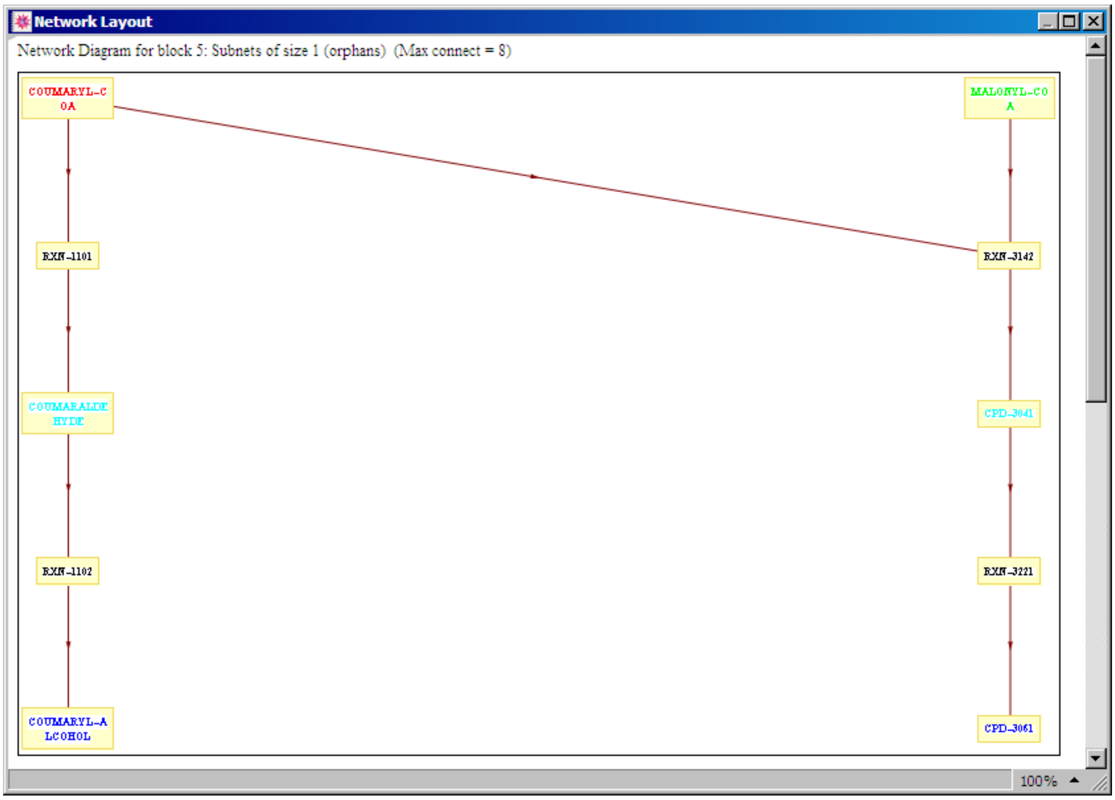
Finally, it is also possible for this type of connection to exist between a subnet and an orphan. For now, these connections are not shown on the metanet diagram at all. They are, however, listed in the printout.

While the metanet gives some overview of the structure of the complete network, its main use is in making decisions about merging subnets and orphans. For example, blocks with internal/external overlaps as described above are probably better to merge together. Also, if all blocks that share a common input or output are merged, that simplifies the connection structure. Merging seems particularly attractive when most, if not all, of the blocks that are merged are small enough that the merged block is still tractable. Conversely, even if two blocks are closely connected in terms of the links shown on the metanet, merging them may not be desirable if they are large, as too much information would be lost in the process.

To facilitate this type of decision, the block size is shown in the format N=123 on each subnet node in the metanet diagram.

6.7.2 The collective orphan layout

Finally, the dropdown offers the choice to display the orphans, i.e. subnets with only a single internal metabolite. All orphan subnets are shown together, inluding any externals that link them to one another.



6.7.3 Colour coding of nodes

Metabolite nodes are displayed in horizontal layers, that are interleaved by reaction layers (in subnets) or subnet layers (in the metanet). In addition the network layout uses colour coding of the node labels to characterise their roles as follows:

BLACK labels identify reaction nodes.

GREEN labels are metabolites that only act as substrates in the complete network, and therefore in the subnet as well.

BLUE labels are metabolites that only act as products in the complete network, and therefore in the subnet as well.

CYAN labels are internal metabolites (appear as both substrates and products) for the subnet.

RED labels are metabolites that are exchanged with other subnets (crossflows); i.e., they are external for the subnet, but internal for the complete network.

MAGENTA labels identify target metabolites (if any).

PURPLE labels are used in the metanetwork to identify "reaction" nodes for the size 1 subnets that are identified by their single internal metabolite or "orphan" metabolite rather than a block number.

Comparison of the crossflows that are shown in a subnet diagram, with those shown in the metanet to connect to that subnet, may show some disparities. This may be caused by using different values of the maximal connectivity filter in producing both layouts. But even if not, additional red-labelled nodes can appear in the subnet diagram. Such a node represents a crossflow not between subnets but rather exchanges with the environment.

To explain that, consider that the list of crossflows for a subnet consists of all its external metabolites that do not appear as pure inflows or outflows for the full network. Any external metabolite that either takes part in a reversible uptake reaction, or is consumed by one reaction and produced by a different reaction, is by this definition treated as a crossflow even though the reversible reaction or the pair of reactions appears in only a single subnet.

A more subtle situation can occur as follows. Suppose subnets *A* and *B* are only linked by both producing metabolite *x*, which in turn is converted to the final product *y* in the full network. So *x* is an internal metabolite in the full network while *y* is external. In the splitting process, *x* is reclassified as external in order to separate *A* and *B*. Now the conversion reaction becomes a reaction between external metabolites *x* and *y*, and as pointed out at the start of section 6.5 would not become part of either subnet. (It could be counted as a subnet of size zero, using the number of internal metabolites as a size measure as before). The result is that the metabolite node *x* would become a crossflow not between A and B, as it is a product of each, but between *A* (or *B*) and the environment. Similarly, if an intermediate metabolite in a single subnet is forced to be external e.g. by its specification in the stoichiometry input file, zero-sized subnets may perhaps unintentionally be split off from the network.

In this way the appearance of a “new” red crossflow node, not present in the metanet nor in other subnets, can be taken as a warning signal that a terminal reaction may have dropped out of the partitioned network as a result of the splitting process. Note that e.g. in terms of flux balance there is no additional information loss from omitting such a reaction – the only loss is that of the mass balance connected with metabolite *x*, when that is made external.

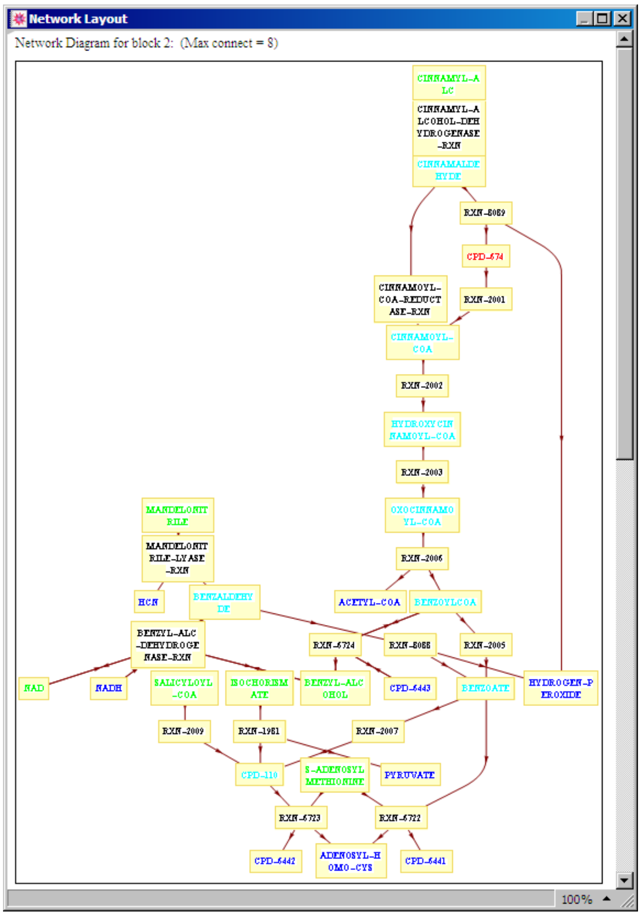
The output diagram can be produced in a variety of sizes. The point of this is that the vertex labels are kept to the same (small) font size irrespective of the total layout size, so overlap of vertices can be reduced and connecting arrows lengthened by using a larger layout size. While this helps to disentangle complicated networks, automated layout should not be expected to be as good as a manual process.

It may or may not be possible to choose the font size used in the labels interactively (due to a *Mathematica* issue, discussed below.). The font size should be chosen mainly for the purpose of the printout; on the screen, the magnification % at the lower right of the display window can be increased to make the labels readable.

The available layout sizes are chosen to coincide with typical paper sizes, but the size of the paper on which the diagram is printed is chosen independently in a standard page setup dialog that appears when a printing output is selected. This dialog also allows selecting and configuring the actual printer to which the printout is sent. Margin settings in this dialog can be ignored, they will automatically be set to 6.3 mm all round by Netsplitter.

It is usually easiest to adjust the paper size in the page setup dialog to the same value that was chosen for the layout. However, a different size may be chosen e.g. to split the printout over multiple pages on a small printer. For example, choosing an A3 Portrait layout, and printing it on A4 Landscape should produce two pages that can be spliced together vertically.

As networks often seem to be elongated either horizontally or vertically, there is also a "double portrait" format listed as **Tall** and a "double landscape" labelled **Wide** . These are meant to be printed out on two pages. So for Tall select the printout paper as Portrait, that should give two pages joined one below the other and for Wide choose Landscape paper, to get two landscape pages next to each other.



Network Diagram for block 2: (Max connect = 8)

6.7.4 Mathematica Issues

Unfortunately, some issues arise in the Network Layout dialog and printing it, depending on *Mathematica* version and its interfacing with printer drivers.

Firstly, in *Mathematica* 6 the dynamic interaction with the layout seems to cause instability and can crash for larger networks. So far no similar problems have been experienced since *Mathematica* 7.

An alternative version of the layout facility is provided that eliminates this problem although it allows somewhat more limited control - font sizes cannot be interactively adjusted and arrangement of the layout is somewhat less flexible. To activate this version, the user needs to edit the “Netsplitter.m” initialisation file, and set the variable SafeDraw = True.

Also, in printing network layouts, especially ones that extend over multiple pages, some issues can arise. Generally, sending the layout to a PDF file and then printing this from a PDF file reader, solves these problems.

Older *Mathematica* versions seem to have problems with an option *PrintMultipleHorizontalPages.* This works for PDF, but when sent directly to a printer blank printouts sometimes result even for printouts contained on a single page. Even in *Mathematica* 7, page counting instructions in PS (postscript) output files are flawed, giving rise to error messages when the file is processed in GSView. Allowing GSView to fix these flaws as it offers to do, normally produces correct output. However, if desired the *Mathematica* option can be disabled for physical printers by setting PrintMultHorPages = False in the “Netsplitter.m” initialisation file as mentioned above. Doing so, however, means that the "Wide" format will not print properly on the printer although it will still work for file output to PDF and PS.

Finally, printing margins for the network layout are set to 6 mm to maximise use of space, but at times the bottom margin appears to revert to a value larger than 50 mm. This behaviour is reasonably well controlled for single page printouts, by Netsplitter in fact setting the bottom margin to 0. Nevertheless the wide margin appears on "Tall" format printouts, and has no known fix at present.

7 Suggested Workflow

A good first step is usually to simply to inspect the appearance of the blocking matrix displayed in the external selection dialog. Ideally, this matrix should contain a clear internal structure of contrasting cells or overlapping blocks in various shades of grey.. If there are already discrete blocks those will be displayed in blue background; but even if the entire matrix is coloured blue, but contains recognisable structure, blocking is likely to proceed well. By contrast, if the matrix is a mostly homogeneous middle grey, there are still too many links in the network to allow blocking and additional metabolites have to be recognised as external before blocking will proceed.

The quickest way to achieve that is to exit the selection dialog to return to the Control Panel, and decrease the maximum connectivity threshold. Values in the range 6 to 10 usually work well, but especially for smaller networks values as low as 4 have been found useful. Alternatively, individual external metabolites can be added to the Externals file if a more focussed strategy is required. Also, making sure that the " Expand reversible reactions" box remains unticked, helps in the initial stages.

Once discernible structure appears in the matrix, one can proceed with the actual blocking process. To explore a new network, the candidate externals proposed by Netsplitter can simply be accepted by repeatedly pressing "Return" until the desired number or size of blue background blocks are found, then using the "Exit" button. If one is only interested in a particular subnet, identified by a list of its internal metabolites, loading this list as "target metabolites" makes it easy to monitor in which blocks they are located. In this case, once one or a few smallish blocks appear containing the targets, that would be the cue to press Exit. Conversely, to inspect the internal metabolites in a given block, one would select it and then look at the metabolite names highlighted in green on the "Metabolites" tab.

As the blocking progresses, one can usually identify blocks with good prospects of further separation by the fact that they contain more than one black or dark grey centre. Further work is speeded up by selecting only such blocks to be further processed in the next round. Another reason to select some blocks, might be e.g. when all target metabolites are found in a block with multiple black centres, and no further splitting of this block is desired. Then one would select all the other multiple centre blocks, but leave that block out. The selection can be done either by clicking directly on a block in the graphical display, or by entering block numbers in the appropriate text box. It may be necessary to click on the display area for the colouring to be updated accordingly.

At each round of the selection process, a number of candidate externals are proposed. Bear in mind that even if they are allowed to be reclassified as externals and removed at this stage, many of them will eventually be restored as internal metabolites during the final housekeeping step so unticking a box is only recommended if one is firmly decided that a metabolite should not be made external and e.g. a previous run has established that it is not automatically restored. Also shown at each round, is the "grey level" cutoff used to select candidates. This value gives a quantitative measure of how much potential for block separation there is. It usually starts at a low value, but tends to increase in subsequent rounds. Once the value approaches 0.5, it means that no metabolite really stands out as a promising candidate and in fact Netsplitter uses this as a cue to stop the selection process if the user has not done so.

Once the "Exit" button has been clicked, the housekeeping step is entered and this can take a while for large networks as the entire blocking procedure needs to be repeated for the complete set of external metabolites collected progressively during the multiple rounds. When finished, the user is returned to the Netsplitter Control Panel. This is a good time to inspect the results so far either by entering the "Merge" dialog or by choosing "Printout".

If "Printout" is chosen, this is first displayed on the screen and whether this is sent to a printer is still optional. The matrix plot at the beginning of the printout gives a good overview of the subnet separation that has been achieved and this may be all that is of interest. The detailed listings of metabolite allocations are formatted for printing so may not be legible on the screen. To save paper, another option is to save to a file where e.g. a PDF reader can be used to magnify the small fonts for reading on the screen.

The same graphical display of the final DAG matrix can also be seen by entering the "Merge" dialog, and internal metabolite listings for individual blocks are also available here on a separate tab. One can simply exit this dialog after inspection if no merging is actually desired.

This would normally be the appropriate point at which to save the results, whether finally or merely in order to be able to recover the current state of the calculation in subsequent runs. That is done by clicking the "Save Subnets" button.

The next major stage is to merge some of the subnets in order to avoid excessive fragmentation. Information for merging decisions can be gleaned from a number of sources.

- **Subnet size:** As an overall guideline, to evenly spread the level of complexity for a network of N metabolites, the ideal would be a metanetwork of $\sqrt{N}$ subnetworks each containing $\sqrt{N}$ metabolite nodes. From this perspective, it is preferable to merge small subnets of only a few nodes with larger ones, while merging two subnets both of a size on the order of $\sqrt{N}$ is less desirable. The weight attached to size considerations reflects how fine-grained one wants the splitting process to be.
- **Overlapping internals:** The printout lists cases where external metabolites of one subnet overlap with internals of another. As explained in connection with the Metanet, separation of such subnets is in some sense an artefact of the simple graph representation used for the blocking procedure. So unless there is a specific reason for keeping such closely linked subnets apart, they are best merged together.
- **Linked Orphans:** Inspecting the layout diagram of the orphan collection (the "subnet" with the highest sequence number) often reveals groups of orphans that are linked by sharing externals. Merging these groups makes sense in reducing the number of trivial size subnets.
- **Target metabolites:** When a list of target metabolites connected with a particular biological function has been loaded, these are highlighted on the matrix display in the merge function and often identify blocks that belong together and should be merged. Once a specific target set has been used, one can

leave the Merge dialog, load another target set, and return to the Merge dialog to process that as well.

- **Shared externals on the metanet:** Inspecting the metanet (subnet no 0) will show which subnets are linked by shared or exchanged externals. Especially in a case where one or several externals link a particular pair of subnets and no others, that would be an indication to merge the subnets (especially where one or both are small). Any orphans that are shown on the metanet in purple are particularly favourable to be merged in with a subnet, as their small size means that such merges simplifies the metanet substantially without complicating the subnet very much. Remember that the network layout suppresses "common" externals according to a threshold connectivity value - this can be adjusted up or down before entering the layout display to facilitate recognising which subnets are linked most closely.
- **Compartmentalisation:** A subnet is usually mainly localised in a particular cellular compartment. When a set of subnets belong to the same compartment, that may be a good reason for merging them to get an overview of the biochemistry in that compartment.

It is usually possible to reduce the total number of subnets substantially - by a factor of two or more - by the listed considerations. There are nevertheless no strict rules, and one may need to experiment with different merge choices to achieve a satisfactory result. Using the efficacy index displayed graphically and by value in the dialog gives some guidance about this.

While a merge cannot be undone directly, that effect can be achieved by leaving the merge dialog, saving subnets (in which case the entire merge history is saved in the ExternalMetabolites.txt file created) and repeating the network splitting with the saved externals file. During the load, the user is offered the choice of executing merge operations only up to a particular one of the saved steps. For this reason, it is usually a good idea to do the most obvious or straightforward merge steps first, and leave the most contentious ones for last.

Note that when alternative merge results are to be compared later, it would be necessary to rename the Subnetworks directory (or at least the ExternalMetabolites.txt file) using the operating system, before the next merging and saving session to prevent files being overwritten.

As is clear from the above, the network layout dialog is very useful to monitor the course of the merging process. Once merging is completed, printing the metanet and any subnets of particular interest would be a way to summarise the results achieved for further study and interpretation.