USER GUIDE

NetSplitter V 1.3

December 2011

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

The program may be freely used in accordance with the GPL but it is required that in any publication of results based on its use, the following reference should be cited:

W. S. Verwoerd, *A new computational method to split large biochemical networks into coherent subnets*; BMC Systems Biology (2011) 5:25; DOI: 10.1186/1752-0509-5-25

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, for various operating system environments. Because of its notebook implementation the *Netsplitter* program can be run on any system for which *Mathematica* is available.

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* metabolite*s*) 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, or exclude nominated metabolites in this file.

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_V1.2.3", which contains the main *Mathematica* notebook file "Netsplitter.nb" that drives the application, one or more example data files e.g. "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. Default values of paths to auxiliary input files are set to this directory when the user browses to the primary input file.

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.

By default, at startup Netsplitter looks for data files in the same directory where the notebook file is located. A different startup directory can be specified by manual editing of the Netsplitter.m file – see comments in the file.

5 Input files

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

5.1 Model Definition – the stoichiometry matrix (S-matrix)

The main input needed is the numerical stoichiometry matrix, as well as identifying text for its rows (metabolites) and columns (reactions). Complete model specifications are increasingly made available in standardized form as SBML (Systems Biology Markup Language) files, e.g. from databases such as BIGG and Biomodels. Model specifications are also often made available as supplementary data with research papers, sometimes as spreadsheet or text files. Alternatively, data can be extracted from metabolic databases such as Biocyc to set up a model. To support this diversity, Netsplitter can read files in several formats:

- OPTION 1

Netsplitter can read a network specified in a standard .SBML 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.

Subnetworks can also be exported in SBML format, and input from the same format ensures the most complete transmission of network specification to subnets.

- OPTION 2

The BioOpt format is accepted, including some extensions detailed below to facilitate use of this format as an intermediary for spreadsheet models.

- OPTION 3

A very simple native tab separated format is provided, containing just the minimal information in a .TSV file.

Each of these formats is further described below. The type of input is recognised from the file name extension - only .SBML, .XML, .BPT or .TSV are accepted.

Naming metabolites and reactions. A wide variety of naming conventions are used in published models, and Netsplitter attempts to accommodate as many of these as possible.

As standardized in SBML, a reaction is identified by a unique identifier (ID) as well as optionally a more descriptive name. Similarly, each metabolite has a unique ID, and in addition a descriptive name and separately an allocation to a cellular compartment.

Many models use the names directly as identifiers, but that can be unwieldy or cause ambiguity in compartmentalised models. That is commonly avoided by attaching a compartment suffix to the name; then duplicate copies of reactions have to be supplied for each compartment.

Internally, Netsplitter uses a 2-component composite metabolite identifier. This could be either the ID plus the compartment, or (e.g.where the ID already contains the compartment suffix, or is simply a number code) it uses the ID plus the name. The two components are merged into a text string but separated by a blank character. This implies that IDs should not contain blanks, but otherwise are unrestricted. If neither a name nor a compartment is supplied, Netsplitter adds a fixed compartment name (the default is “Cytosol”) to the ID.

Compartment suffix treatment. When an explicit compartment allocation is not given for any metabolite, Netsplitter inspects the metabolite IDs for a likely looking compartment identifier in the form of a suffix separated from the ID by an understroke “_” character (a commonly used convention). As understrokes are also often used as separators in chemical names, two further criteria are applied. Only suffixes that occur at least 3 times are considered; and of those, no more than the 12 most common ones are presented to the user in a dialog where they can be approved individually for use as compartment identifiers.

In some models the compartment identifier is simply appended to the metabolite ID without an understroke separator. In these cases automated extraction of the suffix is not feasible. Manual editing of the input to insert the understroke is one solution, particularly for BioOpt or spreadsheet files. In BioOpt input, a separate section can also be created in the file (see below) to input compartment allocations in these cases.

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.

5.1.1 SBML input and output files

Netsplitter accepts input from SBML files conforming to the SBML specification Version 2 Level 4, as documented in Nature Precedings : doi:10.1038/npre.2008.2715.1 .

Only the minimal specification of a static model containing the sections listOfCompartments, listOfSpecies and listOfReactions are required. In these, id-attributes are needed but name-attributes are optional. The simplest form of reaction specification, containing just the listOfReactants and listOfProducts is sufficient.

Two rarely used stoichiometry specifications allowed by SBML rules are not currently supported by Netsplitter: i) Repeated occurrence of the same "species"element e.g. on both sides of a reaction equation, giving rise to what is termed the "effective stoichiometry" in the SBML documentation; and ii) the use of stoichiometryMath elements.

More elaborate SBML models are also acceptable for input; for its own use Netsplitter only extracts the items that it needs, but stores the rest to be passed through to any SBML output files that it produces. As the described TSV layout does not provide for the additional features, they are lost when exporting to TSV.

When input was read from a TSV file, output in SBML format (whether conversion or subnets) is confined to the simple minimal structure described. BioOpt input can also contain values for fluxes, constraints and objectives and these are also exported to SBML output.

In addition, the strict SBML rules for ID attributes are implemented when writing the output. An understroke is prepended to any ID starting with a digit; "+" is replaced by "x" and "-", blanks and any other special characters are replaced by an understroke "_". Netsplitter checks that uniqueness of ID's is not compromised and gives a warning message if it happens. If such a converted or subnet file is loaded back into Netsplitter and external metabolite or target files created for the original text input reused, they would need to be edited accordingly with SBML compatible metabolite ID's. To avoid that, it is simpler to choose TSV output when the input was from a text file.

However when both input and output is SBML, Netsplitter tries to transmit input specifications as faithfully as possible to subnet output in SBML format. In particular, the kineticLaw element of each reaction is passed through to the subnet, making it possible to use Netsplitter also to split dynamic network models into subnets.

Attributes and further elements (including those not used internally) of compartment, species and reaction items are mostly copied unchanged to subnet files, for each of these items that belong to the subnet. However the reversibility attribute of each reaction, and the boundaryCondition attribute of each species is changed in accordance with the values resulting from the Netsplitter analysis.

In addition, global SBML specifications such as Unit Defintions, Rules, Constraints etc, are copied in full to each subnet SBML file. These may as a result contain superfluous specifications not relevant to the subnet, which should not cause any problems. More seriously, inconsistencies could arise if a global section refers to a species or other item not contained in the subnet. Such inconsistencies are not currently checked and would need to be weeded out manually from the subnet files.

The SBML specification does not explicitly provide for reaction flux values, but they are sometimes given as purposely defined local parameters in the kineticLaw section of each reaction. Netsplitter checks for such definitions given as values for FLUX, FLUX_VALUE or FLUXVALUE, and processes any non-zero values that are found as described in more detail in section 5.3 below. In addition or alternatively, reaction

directions are in some models specified by restricting flux values to positive or negative values determined by lower and upper bound parameters. These are also processed by Netsplitter by searching for LOWER_BOUND or LOWERBOUND, and UPPER_BOUND or UPPERBOUND parameters, and if found are used in conjunction with explicit flux value specifications Values read from any explicitly nominated flux file overrides values read from the SBML model.

Finally, if the model specifies an objective (as in the BioOpt input discussed below) this is also entered in the SBML as a kineticLaw parameter. The objective consists of an algebraic combination of reaction fluxes, that is to be minimized or maximized. The weighting factor of each flux is specified as an OBJECTIVE_COEFFICIENT in the kinetic law section of the associated reaction. Commonly, there is a single "reaction" that e.g. defines the composition of the biomass, and maximizing biomass production then means maximizing the flux of this reaction. In such a case the default weighting OBJECTIVE_COEFFICIENT=1 is entered as a kinetic law parameter for the biomass reaction.

As there does not appear to be a commonly used method to specify in SBML whether the objective is minimized or maximized, Netsplitter assumes minimization. Consequently, if the input data specifies minimization explicitly or does not specify it either way, the value of the weighting factor from the data is retained unchanged as the objective coefficient; but if maximization is explicitly specified, the sign of all weighting factors are reversed when creating the SBML specification.

5.1.2 BioOpt and spreadsheet models.

The BioOpt format is used for the BioMet Toolbox, available from the Systems Biology Group, Chalmers University of Technology. A detailed description appears at http://129.16.106.142/tools.php?c=bioopt .

A key feature of this format is that reactions are specified by explicit reaction equations of the form A + B <=> C + D. This makes it amenable to use also for spreadsheets following a similar specification.

BioOpt input files need to be renamed with the extension ".bpt" instead of the usual ".txt" in order for Netsplitter to recognise them.

Briefly, BioOpt files are divided into sections by header lines such as "-REACTIONS", "-CONSTRAINTS", etc. A typical reaction line might be:

reactionID: A + 2 B => C + D

On input, metabolite IDs (e.g. A,B,C,D) as well as stoichiometry coefficients are extracted from this. The –REACTIONS section is the only required section and must appear first in the file. All other sections are optional and may follow the reactions section in any order.

Further standard sections specifying constraints, external metabolites and objective functions, are also processed if they are present. A number of consistency checks are performed on the data, such as that explicitly listed external metabolites appear in the

reaction equations, that constraints are unique and refer to valid reactions, etc. Inconsistencies that are flagged often result from typographical errors, such as using both “glucose 6-phosphate” and “glucose-6-phosphate”. This type of error is surprisingly common especially in spreadsheet models, and Netsplitter error messages are helpful to trace these. Spurious leading and trailing blanks are also quite common, but are automatically corrected on input.

For a spreadsheet model, the user needs to export data from the appropriate sheets into text files, and possibly use a text editor to consolidate these into a single file with sections as described. To facilitate retention of specifications not provided for by the standard BioOpt format, Netsplitter supports the following extensions:

- The following are accepted for reaction symbols:
  `->, <->, -->, =>, <=>, <==>`
- Reaction lines can include a reaction name and a flux value as follows:

  reactionID: reaction name: A + 2 B => C + D :3.4

  All terms in the equation, including coefficients, +, and the reaction symbols, have to separated by whitespace.

  Reaction IDs should not contain colons.
  Reaction names can contain blanks and colons, but not adjacently.
  So “`ab cd`” ,“`ab:cd`” and “`a:b cd`” are OK.
  But “`ab: cd`” and “`ab : cd`” are not acceptable names.
  The same restriction applies also to metabolite IDs.

  The reaction name and flux values are each optional, on an individual basis, i.e. either can be present in some equations and absent in others. Absent names will be given the default value “NoName”. If there is no name or flux value, the corresponding colons should not appear either.

- To specify metabolite names and compartments, an extra section can optionally be added with the header line “-METABOLITES”. This is followed by an arbitrary number of lines with the format

  metaboliteID: metabolite name: compartment

  The metabolite ID that appears in this section must be identical to the one that appeared in the reaction equations, in order for names and compartments to be correctly associated with IDs. If the metabolite ID that appears in a reaction equation contains an internal blank, it has to be entered exactly the same way here as well. After Netsplitter has processed the input data, it inspects all metabolite IDs and automatically substitutes any internal blanks by understroke characters (“_”) to construct valid composite metabolites names.

  The METABOLITES section is optional, but if it appears should normally contain a line for each metabolite in the reaction system. Any metabolites that are missing are allocated the default name “NoName” and compartment “cytosol”.

On each metabolite line, either the metabolite name or the compartment or both can be given. Note that the compartment specification is not followed by a colon, to distinguish it when no name is given.

The restriction on adjacent blanks and colons in names applies to metabolite IDs, names and compartment names as well.

- An optional section with the header –FLUXUNITS can be added to specify measurement units for fluxes and constraints. This section contains a single line specifying the unit. The default value “"mmol_per_gDW_per_min" is assumed when this section is missing. The format is:

  Unitname: flow_type flow_scale weight_type weight_scale timefactor

  There are 6 terms are separated by white space and can have values as follows.

  “Unitname” is a text string. It has to be specified according to SBML requirements for IDs, i.e. it can only contain alphabetic characters, numbers and understrokes (but cannot start with a number).”.
  The “type “ fields are text strings and can be either “mole”, “gram” or “kilogram”.
  The “scale” fields are numbers, giving the exponent of 10 that converts the given type unit, to the unit actually used in the data. For example, if the flux is specified in millimoles, the type is “mole” and the scale is “-3”. Note that if the type and data units are the same, the :scale” value is 0 (because $10^0 = 1$).
  The “timefactor” is a number, and represents the multiplier needed to convert data to a time unit of seconds. For example, if flux is given per minute, the value “0.01667” (i.e., 1/60) should be given for “timefactor”.
  A few typical cases are listed in the table below.

Unitname	flow_type	flow_scale	weight_type	weight_scale	timefactor
Mmol_per_gDW_min	mole	-3	gram	0	0.016667
Microgram_per_Kgh	gram	-6	gram	3	2.7778E-4
Mmol_per_mg_per_sec	mole	-3	kilogram	-6	1

- The objective function for an FBA model can be specified in various ways. Such an objective is not used by Netsplitter, but it is read from the BioOpt file for transmission to SBML files that are exported.
  The simplest specification is to include a section headed -OBJECTIVE. The objective itself is given as an algebraic combination of reaction ID’s. It can include (signed) weighting factors to specify a composite objective.
  The algebraic formula can be preceded by either “min “ or “max ” to indicate that the weighted combination of fluxes is to be minimzed or maximized respectively. When this is omitted, minimization is assumed.
  The BioOpt format alternatively allows section headings in the form –MINIMIZE or –MAXIMIZE to specify the type of objective.
  It also allows the specification of a second objective, by allowing a section headed by –DESIGNOBJECTIVE with similar specification as the main objective section.

See below for how the objective is exported to SBML.

Placement of the additional sections in the BioOpt file is arbitrary, except that they should occur after the main –REACTIONS section.

5.1.3 TSV input files

Netsplitter can also read and write a very simple tab separated (.TSV) text file as described here. The data file DEMO.TSV supplied as part of the download package is an example of this format.

Use of this format has largely been superseded by the SBML and BPT formats. However, it is an easy format to prepare from scratch, e.g. for simple test models. It can also occasionally be useful for spreadsheet models that contain the stoichiometry in explicit matrix form rather than as reaction equations. Note that only non-zero elements of the matrix are given by use of the MatrixMarket format for sparse matrices.

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 available in flat file format. This produces 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 ID’s, 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 ID’s 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 in matrix displays, the printout listing and 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.

As described above, compartment suffixes such as _c and _m are extracted out of IDs and offered for approval to the user during processing of input files.

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.

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.

5.1.4 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 is 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.

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 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.

On each line, the metabolite ID and compartment ID is given, separated by a blank. Note that even when using the ID+Name specification in the stoichiometry input file, the second field in the external metabolites file needs to be the **compartment**. For readability, the metabolite name may follow; any data beyond the first two fields are ignored.

It is also admissible to give only the first field, i.e. the metabolite ID; in this case a default compartment specification that includes all compartment IDs, is added automatically. However when the compartment specifier is omitted, it is imperative that any metabolite name is preceded by a "%" so that it is considered a comment and ignored.

The combination (ID, blank, compartment) is in fact interpreted as a Regular Expression. So not only exact ID's, but also an expression to be matched can be given. That is useful in multicompartment models, especially when ID's contain compartment suffixes.

For example, `WATER_e, WATER_cytosol` and `WATER_mit` will all be matched by "`WATER_.*`", and "`.* Extracellular`" (note the blank) will match all metabolites in the extracellular compartment.

Example :

```
%%%%% Common external metabolites using KEGG id's %%%%%%%%%%%%%%%
%%%%%%%%% Environment molecules and ions %%%%%%%%%%%%%%%%%%%%%%%%%
C00001   % H2O
C00007 Cytosol|Chloroplast  % Oxygen
C00011 .*  % CO2
C00014 \w*  % NH3; Ammonia
C00282   % Hydrogen; H2
C00080 Cytosol  % H+
C00238   % Potassium; K+
C01330   % Sodium; Na+
```

```
C00698   % Cl-; Chloride
C00708   % Iodide; I-
C01382   % Iodine; I2
C00533   % Nitric Oxide; NO
C00088   % Nitrite; HNO2
C00244   % Nitrate; HNO3
C00320   % Thiosulfate; HS2O3
C00059   % Sulfate; H2SO4
C00094   % Sulfite; H2SO3
C00009   % Orthophosphate;Pi; H3PO4  % inorganic phosphate
C00076   % Calcium; Ca2+

      % Molecules assumed ubiquitous, e.g. involved in many reactions
      % or in important biological functions
C00027   % Hydrogen peroxide;H2O2
C00060   % Carboxylate; H-COOH
C00067   % Formaldehyde;
C00704   % O2- % SUPER-OXIDE
C00013   % Diphosphate;Pyrophosphate;PPi; H4P2O7
C00536   % Triphosphate;PPPI H5P3O10
C01342   % NH4+   % AMMONIUM
C00114   % Choline     % fatty acid and lipid metabolism,
C00307   % CDPcholine         % synthesised by several pathways
C00251   % Chorismate   % raw material for aromatic amino acids
C00885   % Isochorismate   %

%%%%%%%%% From catabolism  %%%%%%
            % Sugar phosphates
C00111   % DHAP; Dihydroxyacetone phosphate     % Triose
C00118   % D-Glyceraldehyde 3-phosphate; GAP % Triose
C00093   % Glycerol-3-phosphate
C00279   % D-Erythrose 4-phosphate        % Tetrose
C00442   % alpha-D-Ribose 1-phosphate           % Pentose
C00117   % D-Ribose 5-phosphate                % Pentose
C00199   % D-Ribulose 5-phosphate         % Pentose
C01101   % L-Ribulose 5-phosphate         % Pentose
C00231   % D-Xylulose 5-phosphate         % Pentose
C00085   % D-Fructose 6-phosphate         % Hexose
C00354   % D-Fructose 1,6-bisphosphate          % Hexose
C00446   % alpha-D-Galactose 1-phosphate  % Hexose
C00636   % D-Mannose 1-phosphate                % Hexose
C00275   % D-Mannose 6-phosphate                % Hexose
C00103   % D-Glucose 1-phosphate                % Hexose
C00092   % D-Glucose 6-phosphate                % Hexose
C00668   % alpha-D-Glucose 6-phosphate          % Hexose
            % alpha-keto acids
C00022   % Pyruvate
C00033   % Acetate
C00036   % Oxaloacetate
C00058   % Formate
C00149   % Malate, an isomer of Oxalacetic acid
C00026   % 2-Oxoglutarate
C00074   % Phosphoenolpyruvate            % pep

            % from photosynthesis, starch/sucrose degradation
C00031   % D-Glucose
C00159   % D-Mannose

            % from aerobic/anaerobic respiration, TCA cycle
C00042   % Succinate
```

```
    % Amino acids
C00025     % L-Glutamate
C00073     % L-Methionine
C00079     % L-Phenylalanine

%%%%%%%%%%%%%%%% Carrier molecules
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C00002   % ATP
C00008   % ADP
C00020   % AMP
C00044   % GTP
C00035   % GDP
C00144   % GMP
C00015   % UDP
C00030   % Reduced acceptor
C00028   % Acceptor
C00005   % NADPH
C00006   % NADP+
C00004   % NADH
C00003   % NAD+
C01352   % FADH2
C00016   % FAD
C00101   % Tetrahydrofolate
C00445   % 5,10-Methenyltetrahydrofolate
C00019   % S-Adenosyl-L-methionine
C00021   % S-Adenosyl-L-homocysteine
C00029   % UDPglucose    % metabolism of glycogen,starch,cellulose
C00052   % UDP-D-galactose
C00010   % CoA
C00083   % Malonyl-CoA  % raw material for fatty acid synthesis
C00024   % Acetyl-CoA
C00040   % Acyl-CoA
C00229   % Acyl-carrier protein
C00173   % Acyl-[acyl-carrier protein]
C00091   % Succinyl-CoA

%%%%%%%%%%%%% Metabolites assigned to extracellular compartment %%%%
 \w*   Extracellular|Extraorganism
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
<Refusals>
C00031 Cytosol   % D-Glucose
```

Note that the characters \ ^ $. [ ] | ( ) * + ? all have special meanings in regular expressions. These characters are anyway not allowed in ID's by SBML rules, so should not occur in the ID or compartment fields; but if they do, each special character has to be preceded by the '\' character to be interpreted at face value.

In addition to the standard regular expression syntax, Netsplitter also recognises the extensions provided by *Mathematica*. The most relevant ones are:

```
\d    -     matches digits 0-9
\D    -     matches non-digits
\w    -     matches word chars(letters,digits,understroke)
\W    -     matches non-word characters
\s    -     matches whitespace (blank, tab, etc.)
\S    -     matches non-whitespace
```

Only "word characters" are allowed in ID's according to SBML rules, and all matches are case-sensitive.

After the externals listing, the file contains a section headed by the tag <Refusals> (on its own line). This lists any metabolites that are NOT to be taken as external. Again, the specification is read as a regular expression and follows the same rules as explained bove.

Refusals are 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 to give exceptions to a general rule stated as a regular expression in the first part, or when experimenting with temporarily not taking a particular metabolite as external. In the latter case it is probably less confusing 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).

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 ID specification in the stoichiometruy and externals files were incompatible!

To facilitate error-checking in the externals file, any specifications that were not matched to any metabolites are listed on the “Refusals” tab of the externals selection dialog.

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.

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.

Reaction flux and bounds information read from the model specification file, is used in the same way as described above to fix reaction directions. The two sources of information are complementary; but where the same reaction is listed in both, the flux value taken from the flux file overrides the model specs.

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 and compartment per line, with %-comments. Again, this is a regular expression used for matching, and the same comments stated for the external metabolites file apply here regarding naming conventions etc..

Example:

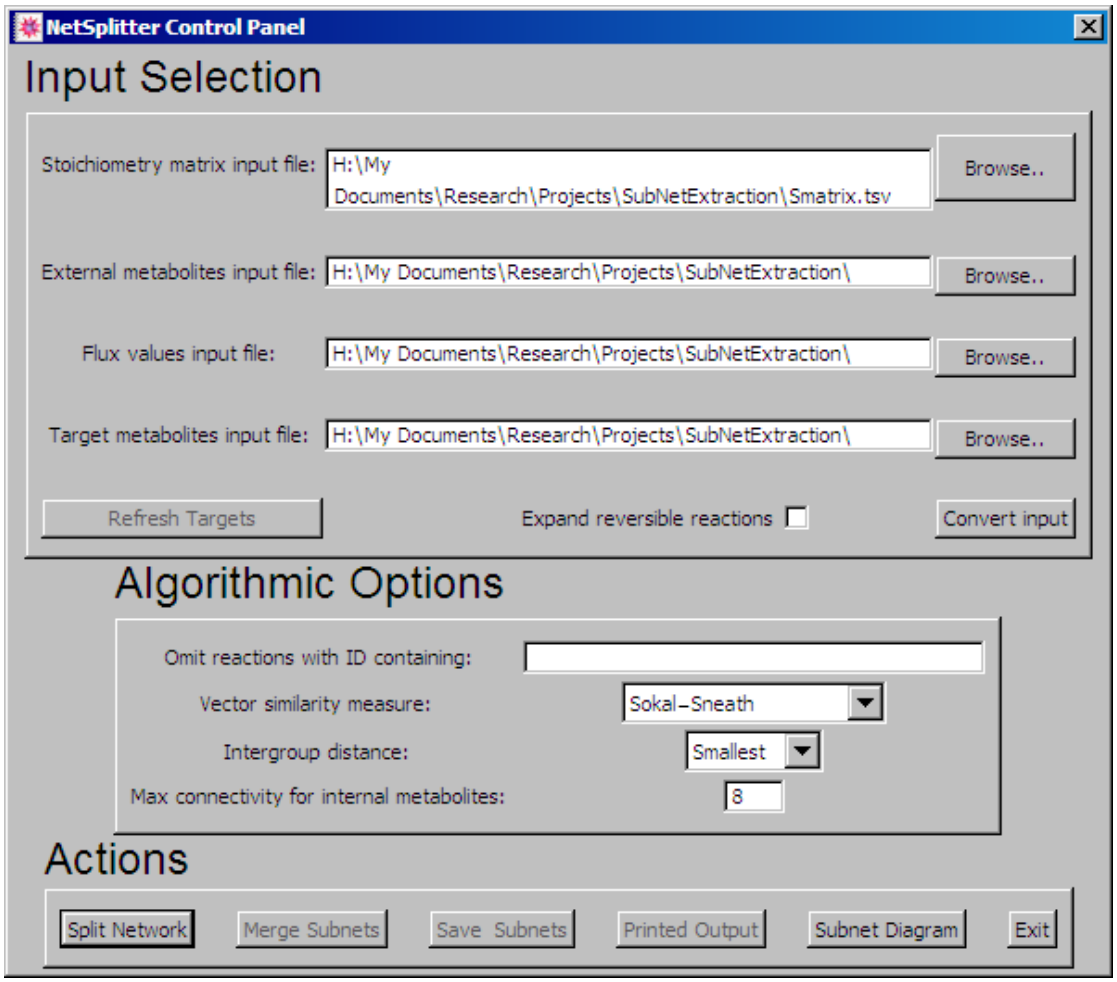
```
%%%%%%%%% metabolites of interest %%%%%%%%%%%%%%%%%%%%%%%%%%

APIGENIN .*                    %% Flavonoids
NARINGENIN .*                  %% Flavonoids
LEUCOPELARGONIDIN-CMPD .*      %% Flavonoids
CPD1F-90 CYTOSOL
CPD1F-4..
```

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.



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.

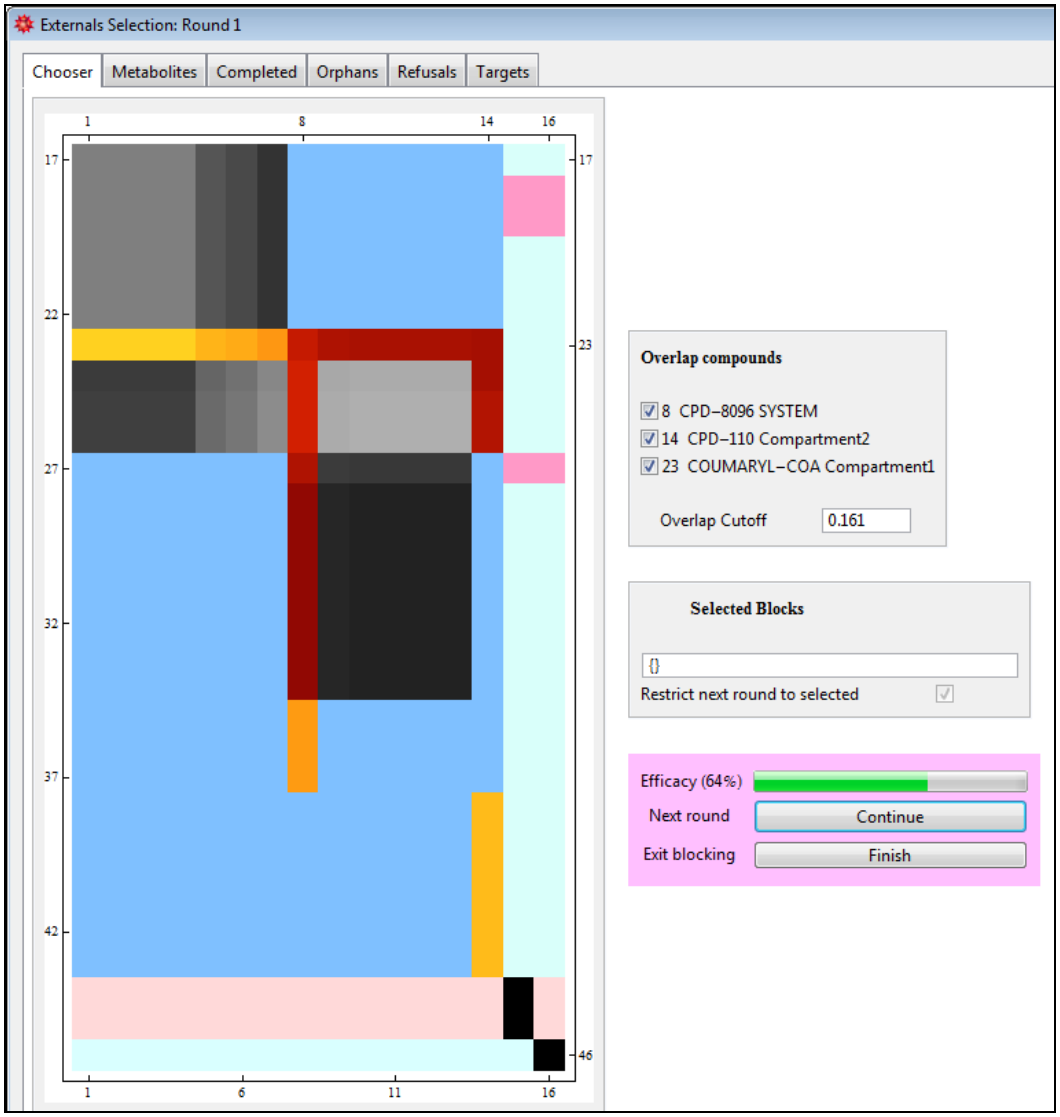
3. In the hierarchical grouping, the method used to calculate the distance between groups of vectors (rather than between individual vectors) can be selected. The default, *single*, takes the smallest separation between vectors belonging to different groups as the group separation. Several other options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



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 it 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. It may be necessary to click on the graphic (outside the blue blocks) to update the colouring accordingly.

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, as seen for the two small blocks at the lower right of the picture above.

The blue blocks are themselves displayed on a background of lightly coloured horizontal bands. The colours are allocated to indicate the cellular compartment that predominates in the metabolites that participate in each block. Where the input data places all metabolites in the same compartment this background is a light turquoise. The same colour results when a block contains a random mix of compartments. That applies to the large first block and small last block in the figure above. The second block in the picture is salmon coloured, because the majority of its metabolites belongs to compartment 1. So the emergence of distinctly coloured bands generally indicate a tendency for blocks to collect metabolites from particular compartments together.

Another item visible 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 shading 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 pink stripes are overlayed on the more lightly coloured compartment bands, and can also usually be distinguished by the fact that they only have the width of a single row or column, not that of the complete block.

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. Instead of graphical selection, block numbers may alternatively be typed straight into this box, taking care to preserve the enclosing curly brackets. 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 three 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.

Alternatively, the choice is offered to proceed automatically. This choice is equivalent to simply accepting all selection choices offered by Netsplitter in all subsequent selection rounds. It is particularly useful for large networks where it can become tedious to attend to each round. By clicking on “Automatic” the user relinquishes control and lets Netsplitter get on with the job as best it can, until it can find no more viable externals.

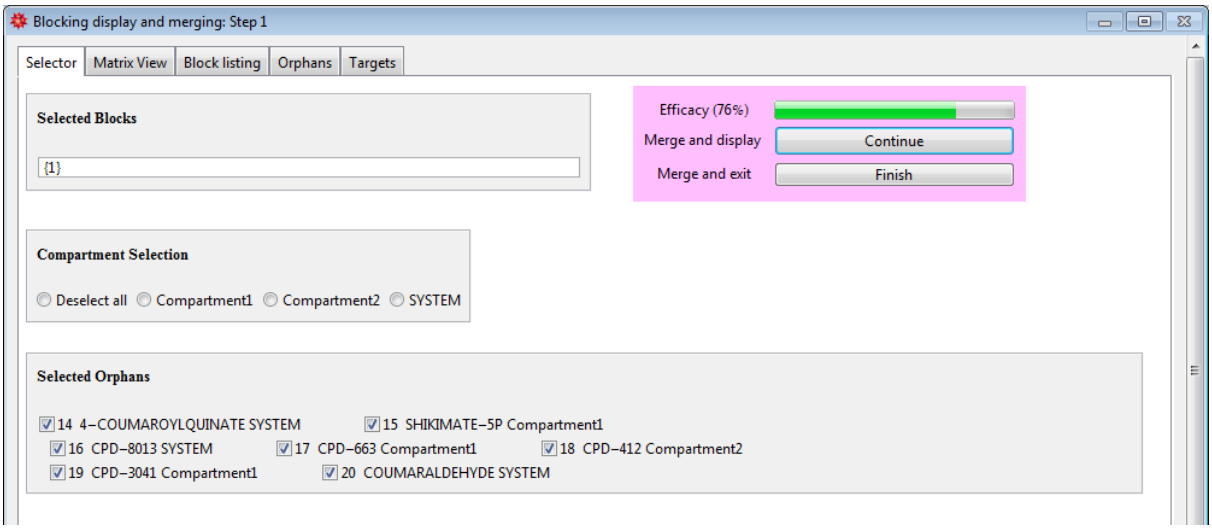
6.3 Merging subnets

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.

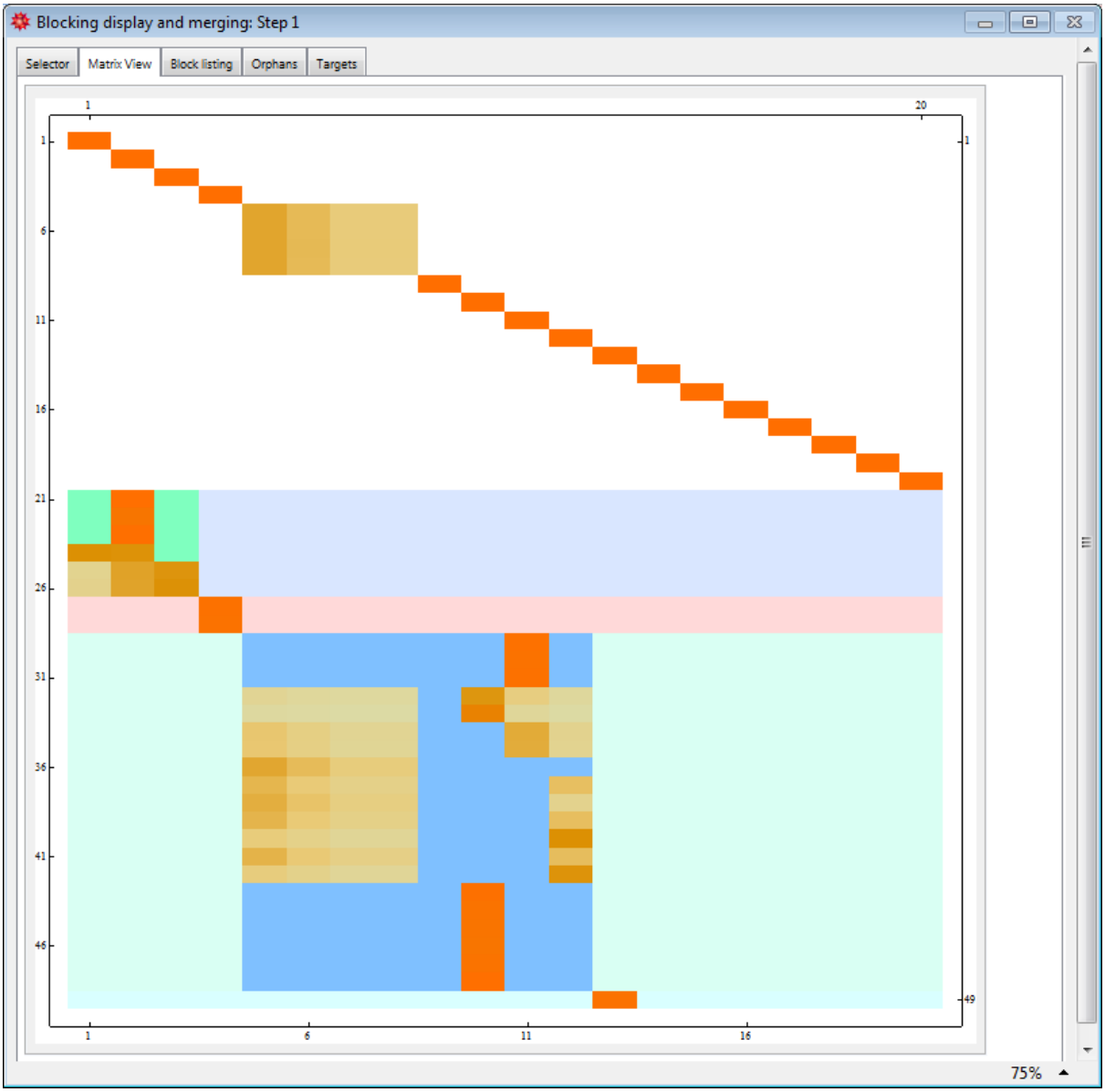
The Merge function is in fact also handy just for inspection of the final DAG matrix, even if no merging is done. Its display is similar to that of the Chooser, but it shows the actual DAG rather than its derived blocking matrix. It also shows the result of the reincorporation step, so blocks are usually 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.

Metabolites are listed blockwise for the merge dialog with sinks listed first in each block. The listing also shows if the block can be allocated to a specific cellular compartment. The criterion used for this purpose is that the single most prevalent compartment for the block must be associated with more than 66% of its internal metabolites. For blocks with more than a single digit metabolite count, this percentage is in practice usually well into the nineties.

In the merge dialog, the selection controls and the matrix display appear on different tabs.



Blocks in the matrix display are once more shown on coloured horizontal background bands, to indicate the cellular compartment with which each block is associated, as well as the pink stripes or stubs to identify target metabolites if any are specified.



In the illustration above, block no 1 has been selected, and this is shown both on the matrix display where it is coloured green, and on the selector tab where it appears in the text box provided for entering explicit numbers.

As in the Externals Selection dialog, blocks can be selected either by mouseclick on a block in the Matrix View tab, or by typing its number inside the curly brackets in the box provided on the Selector tab.

In addition, a set of radio buttons are provided to select all blocks and orphans associated with a particular cellular compartment. This selector will only appear if the input data identified more than one cellular compartment. When a compartment is selected, all blocks belonging to it are highlighted in green on the Block Listing tab,

and its orphans receive ticks in the panel below. This feature is useful to inspect the relation between subnets and compartments. If no selection is visible, either its metabolites are all external or a completely empty compartment was specified in the SBML input file.

Compartment selectors are mutually exclusive, and will also cancel any selections made manually before the radio button is clicked. Once a particular radio button has been clicked, it is nevertheless possible to add or remove selections by either mouseclick or modifying the explicit block selection list. A button for cancelling all current selections is also available.

The Selector tab finally provides a tick box to control individual selection of each orphan metabolite. In the interests of compactness for use with large networks, the layout of these boxes is somewhat untidy. The listing of orphans in columns on the Orphans tab may be helpful to locate particular metabolites. Any orphans selected here will be merged in with the collection of selected blocks when either the “Continue” of “Finish” buttons is pressed.

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 matrix display, 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.

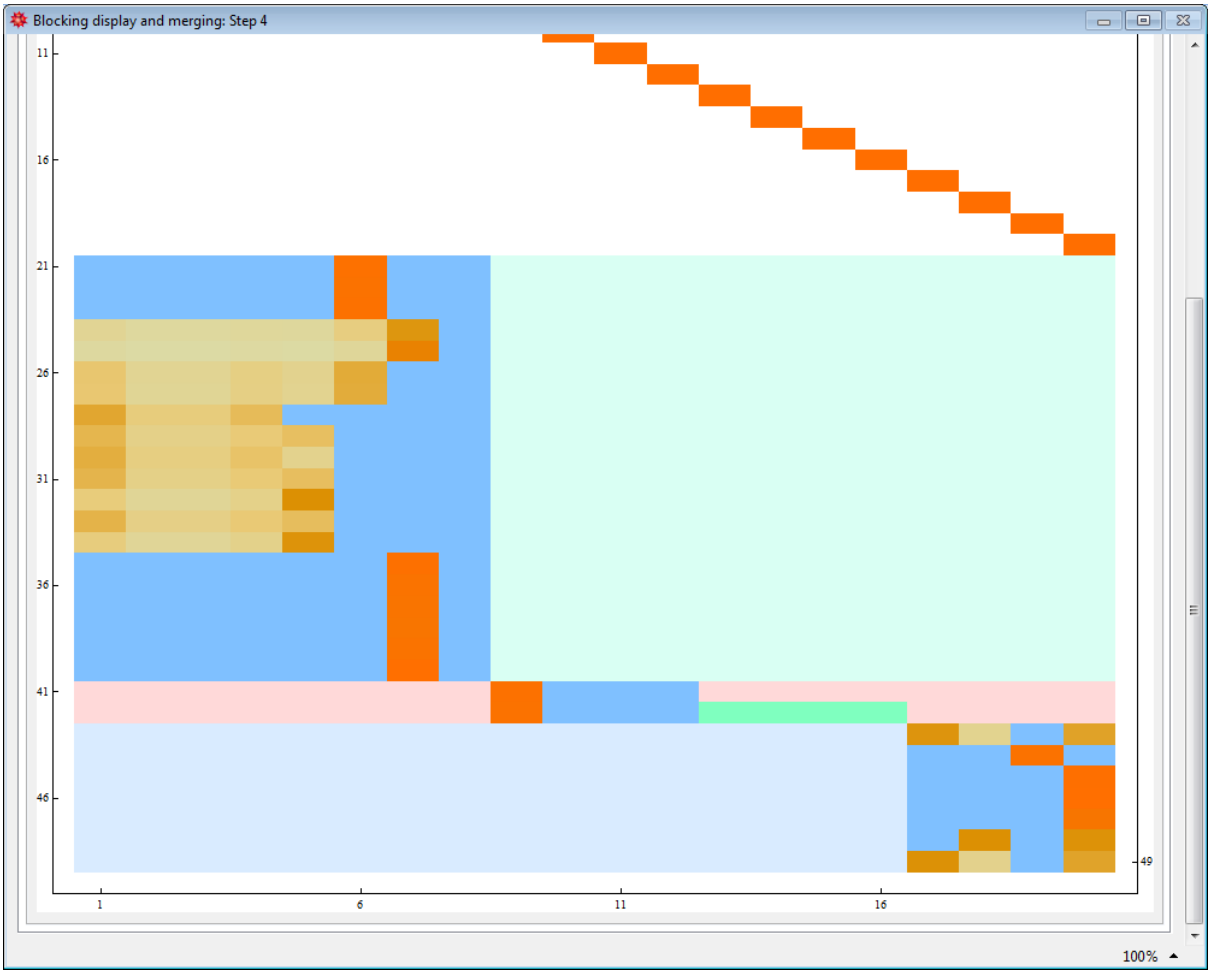
If a specific orphan block is created by merging, this is shown as a quasi-block (blue line) that appears to overlap with the last row of the previous block in the DAG. Conceptually, such a block is again more like a horizontal line covering the set of sinks that it contains, but with zero width in the 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.

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.



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. However, this issue can also affect selection of blocks by mouseclick on the matrix display. If a block does not change colour on clicking, it may need to be selected by entering or deleting its number in the block selection text box on the Selector tab.

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 Output of results

6.5.1 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.

6.5.2 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.

6.5.3 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.

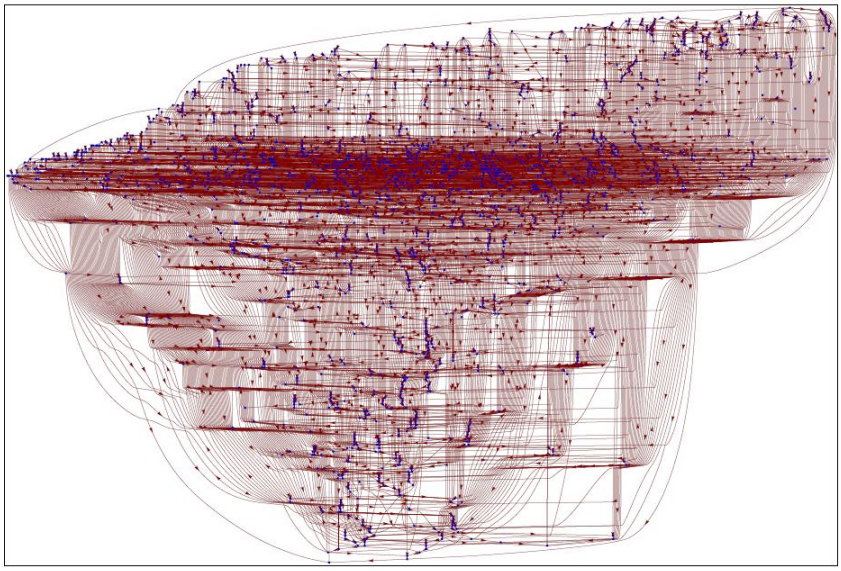
The user also has the option of leaving the printout window on screen, e.g. so it can be compared with the output from a subsequent run without wasting paper.

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.

6.6 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. Subnet numbers are as shown for blocks e.g. in the merging display and in the printout. Two additional layouts are available: the metanet and the collection of orphan network fragments..

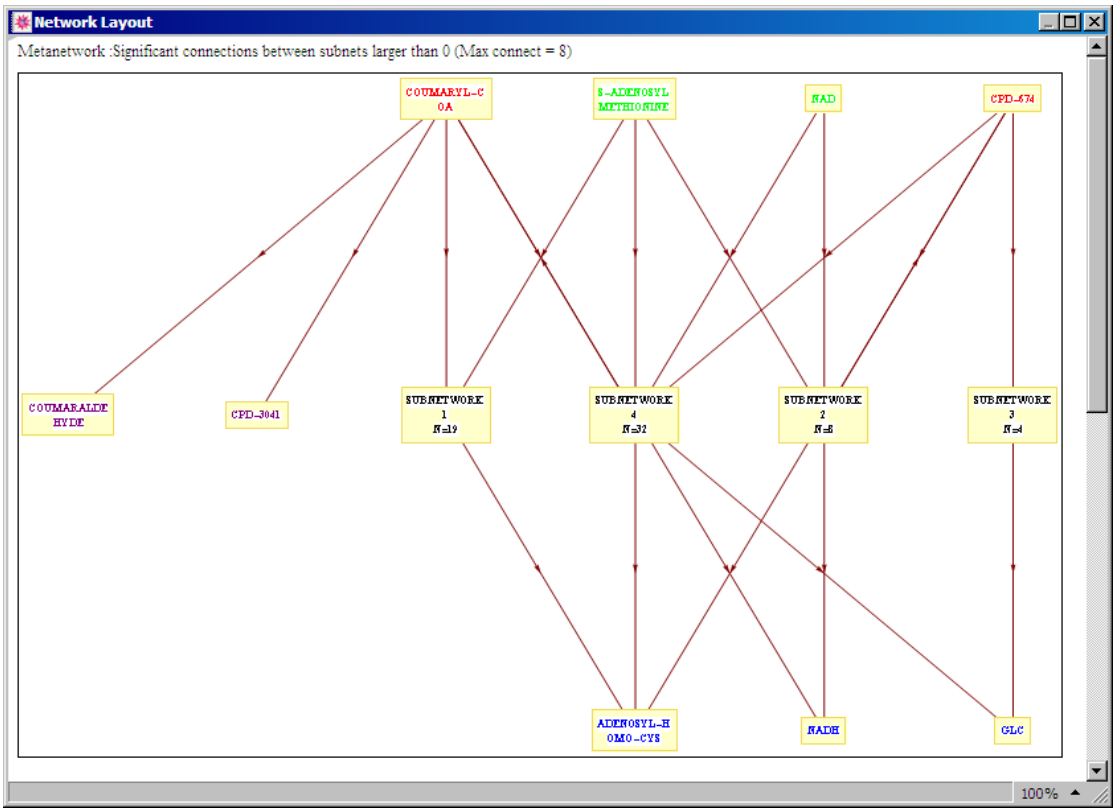
Before displaying the network, the user can choose a maximal metabolite connectivity to be included. The default value is the one set in the main control panel for the initial identification of externals. However, a larger or smaller value can be chosen as appropriate for the complexity of the subnet.

6.6.1 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 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 shown 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.

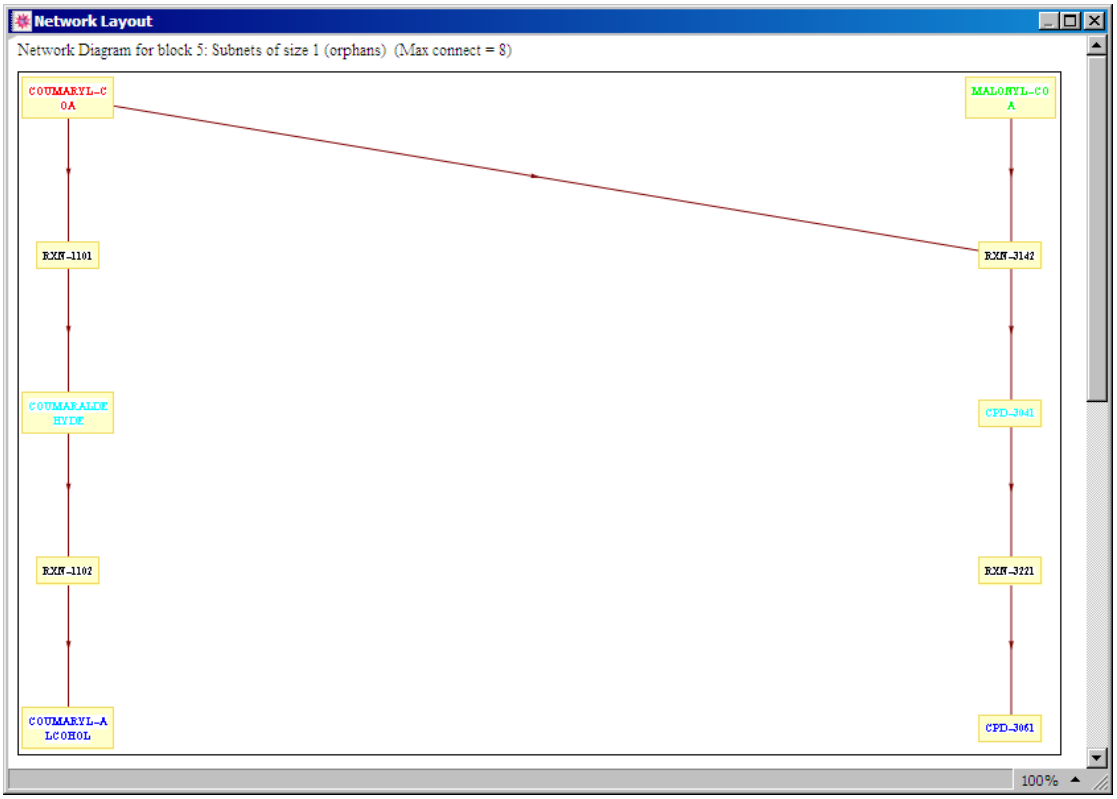
Finally, it is also possible for this type of connection to exist between a subnet and an orphan. For now, these connections are not shown on the metanet diagram at all. They are, however, listed in the printout.

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.

6.6.2 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.



6.6.3 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). To emphasize the distinction, reaction and subnet nodes are shown as ellipses and metabolites as rectangles.

Subnet nodes in the metanet are given coloured backgrounds to indicate their dominant cellular compartment, according to the same colour scheme used in the matrix displays.

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 6.5.1 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.

6.6.4 Configuring network layouts

The display of a network layout is accompanied by the control box shown below that allows considerable flexibility to adjust the display and choose output options. The configuration controls in this box are dynamic – adjustments are implemented live in the displayed network. The output options are only applied when exiting the dialog.

Paper Size	A4 A3 A2 A1 A0
Orientation	Tall Portrait Landscape Wide
Font Size	1 4 5 6 7 8 9 10
Node width	
Node height	
Layout type	Layered Centred 3D Interactive
Output to	None Window PDF File PS File Printer

Firstly, 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.

When generating graph layouts, Mathematica adjusts the aspect ratio of the graphics elements (but not the fonts) to suit the selected overall size. This can result in size mismatches between node boxes and their labels. To alleviate that, the vertical and horizontal dimensions of the boxes can be manually adjusted with sliders.

Three different options for the algorithm used to calculate the layout are available and can be changed interactively in real time. The default layered layout best reflects the bipartite nature of metabolic networks by separating metabolites and reactions into alternate layers, reinforcing the different shapes used to distinguish the two node types. However, this can lead to crowding of nodes making it difficult to recognise whether a connecting arrow terminates at a node or simply passes behind it.

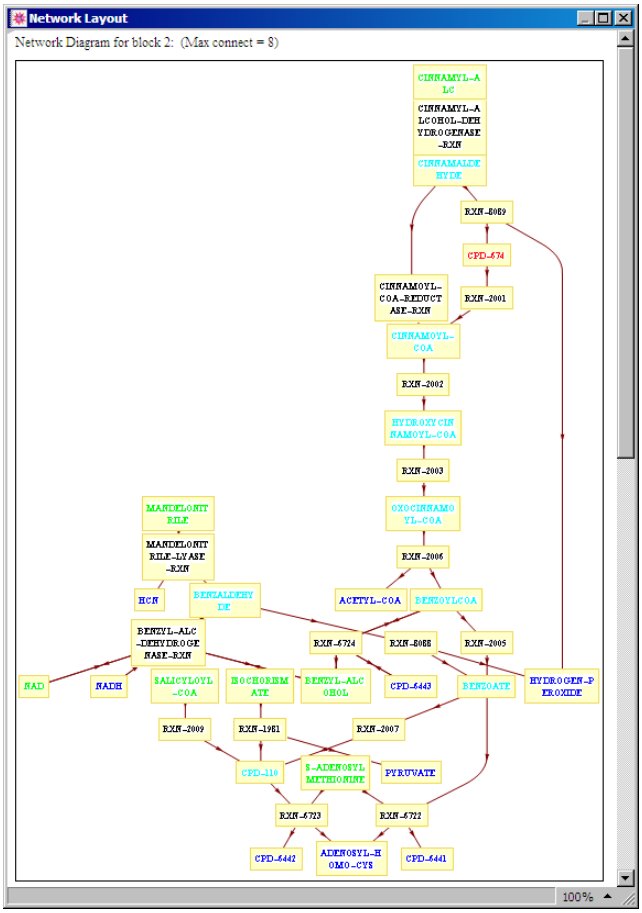
The centred layout associates attractive and repulsive forces with nodes and finds a layout that minimises the associated energy. This gives a completely different arrangement of nodes that can be helpful to clear up ambiguities about connections.

The third alternative uses same energy minimisation scheme to generate a 3-dimensional network layout. This representation can be interactively manipulated on the screen by dragging with the mouse pointer. There are three possibilities. When the mouse pointer is moved close to the network it changes into a “rotation” icon, and while the left mouse button is pressed the image is rotated in 3D by dragging. Once the rotation icon appears, it can be changed to a zoom function by holding in the “Ctrl” or “Alt” key; or to translational dragging by using the “Shift” key. Due to this interactive aspect the 3D representation is not available for hardcopy, although it can be kept on screen for the duration of a Netsplitter session. It can also be saved as a separate notebook that can be run independently in a later *Mathematica* session including 3D mouse interaction.

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 are preset to 6.3 mm; this can be adjusted but larger settings will usually cause undesirable page splitting .

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.



The layout control allows for the layout to be printed to PDF or Postscript files in addition to a printer. Output to a range of other formats is possible by the following trick. In the layout size dialog, choose the option “Keep on screen” which returns the user to the Netsplitter control panel. Now click in the layout window that remained on the screen, then right click on the actual layout graphic. In the context menu that

appears, click on “Save graphic as...” and then choose an appropriate file format in the file selector dialog. This is particularly useful to save a network layout in WMF or EMF format for importing into MS Word etc.

6.7 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. It appears that with multiple page graphical printing, the Mathematica developers intended to print an overlapping strip in duplicate. In practice only the strip at the top of the 2nd page prints, leaving the apparent wide bottom margin on the 1st page. 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.

Other minor issues arise in the matrix display and printout for large networks, as was commented on in describing the interface. These are connected with resampling to fit the matrix display into limited pixel resolution. The result is that small blocks may not be resolved properly and appear overlapping, and/or colouring conventions may not be properly followed. This can also affect the possibility of selecting blocks by mouseclick, but the alternative mechanism of entering explicit block numbers into the selector text box provides a workaround for this issue.

7 Suggested Workflow

7.1 Selection

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 choosing “Automatic” or 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. However, there is a magnification button at the bottom right of the printout window that can be used to make the listings legible.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.

It is sometimes noticed in the printout or the merge dialog window, that a few unacceptably large blocks remain. A strategy that often works is to go back to the control panel and load each of these blocks on its own out of the stored “Subnets” folder. It can then be split on its own, usually using a smaller connectivity threshold. If the block then separates successfully, one inspects its printout to identify which metabolite(s) produced the separation (usually listed under the ones chosen interactively). This can then be noted and included in the original external metabolites file and the separation of the full network repeated with this extended set of external metabolites. In this way recalcitrant blocks can usually be broken up to produce a more homogeneous subnet size distribution.

7.2 Merging

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 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.

  In complicated networks, it is sometimes hard to tell whether a network link terminates at a particular node or simply passes behind the node box. There are several ways to settle this. Choosing the label font size as 1 reduces the nodes almost to a point. Using the sliders to shrink the box size while retaining the text label in readable size, is sometimes preferable. Switching to an alternative layout algorithm is another way to clear up node connections.

  A neat trick to have the best of both worlds, is to choose the output option “Window” to keep one view of the network on the screen, then rerun the subnet layout to create another view for comparison. This works particularly well with the 3D layout; even though it cannot be further reconfigured when kept, it does remain live with respect to 3D mouse manipulation. The 3D view can even be stored and reloaded between sessions for this purpose via an external notebook file
- **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.

  The coloured background bands on the merge dialog display and coloured subnet nodes on the metanet layout give visual clues about which blocks are associated with the same compartment. In addition, the “Block Listing” tab in

this dialog indicates the predominant compartment for each block. Note that the coloured bands gives more nuanced information on this – the actual colour is a blend of the colours contributed by all metabolites in the block, so can show minor variations even for bands that have the same majority compartment allocation.

In large networks, it can be tedious and error-prone to select blocks associated with a compartment manually. The compartment selector should be used instead; selections made by it can be manually modified as needed.

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.

A tutorial introduction, including a stepwise case study, to the use of Netsplitter is available for free online at :

Dissecting Metabolic Networks into Functional Subnets; W.S. Verwoerd, Chapter 7 (*p* 79-98) in *A Practical Guide to Bioinformatics Analysis* (Ed.: Gabriel P.C. Fung), iConcepts Press, Brisbane, 2010.

URL:
http://www.iconceptpress.com/books/aPracticalGuideToBioinformaticsAnalysis/index.php