S.1 Description of a COBRA-compliant SBML file structure

The format of the SBML files used in this work were generated using the standards outlined in http://sbml.org/documents/specifications. COBRA compliant SBML files should contain the following data:

- Unit definitions
- Compartments
- Metabolites (format: ‘M_*<metabolite abbreviation>*_*<compartment id>*’)
    - Metabolite name, compartment, charge, formula
- Reactions (format: ‘R_*<reaction abbreviation>*’)
    - Reaction name, reversibility, reaction stoichiometry, gene-protein-reaction (GPR) association, subsystem, E. C. number, lower bound, upper bound, flux value, objective coefficient

Unit definitions are listed in the <listOfUnitDefinitions> section of the SBML file in the following format (example from ecoli_core_model.xml):

```
<listOfUnitDefintions>
    <unitDefinition id=”mmol_per_gDW_per_hr”>
        <listOfUnits>
            <unit kind=”mole” scale=”-3”/>
            <unit kind=”gram” exponent=”1”/>
            <unit kind=”second” exponent=”-1” multiplier=” 0.000277777777777778"/>
        </listOfUnits>
    </unitDefinition>
</listOfUnitDefinitions>
```

Compartments are listed in the <listOfCompartments> section of the SBML file in the following format (example from ecoli_core_model.xml):

```
<listOfCompartments>
        <compartment id=”c” name=”Cytoplasm”/>
        <compartment id=”e” name=”ExtraOrganism”/>
</listOfCompartments>
```

The compartment id should correspond to the compartment abbreviation appended to metabolite names. The full name of the compartment is defined using the name parameter.

Metabolites are listed in the <listOfSpecies> section of the SBML file in the following format (example: acetaldehyde in the cytoplasm):

```
<listOfSpecies>
    …
    <species id=”M_ACALD_c” name=”Acetaldehyde” compartment=”c”/>
        <notes>
            <body xmins=”http://www.w3.org/1999/xhtml”>
                <p>FORMULA: C2H4O</p>
```

```
                <p>CHARGE: 0</p>
            </body>
        </notes>
    …
</listOfSpecies>
```

The compartment abbreviation should be appended to the end of the metabolite id. The formula for metabolite is defined in the <notes> section. Note that as the charge parameter was deprecated as of SBML Level 2 Version 2, metabolite charge is now specified in the <notes> section.

Reactions are listed in the <listOfReactions> section of the SBML file in the following format (example: D-lactate dehydrogenase):

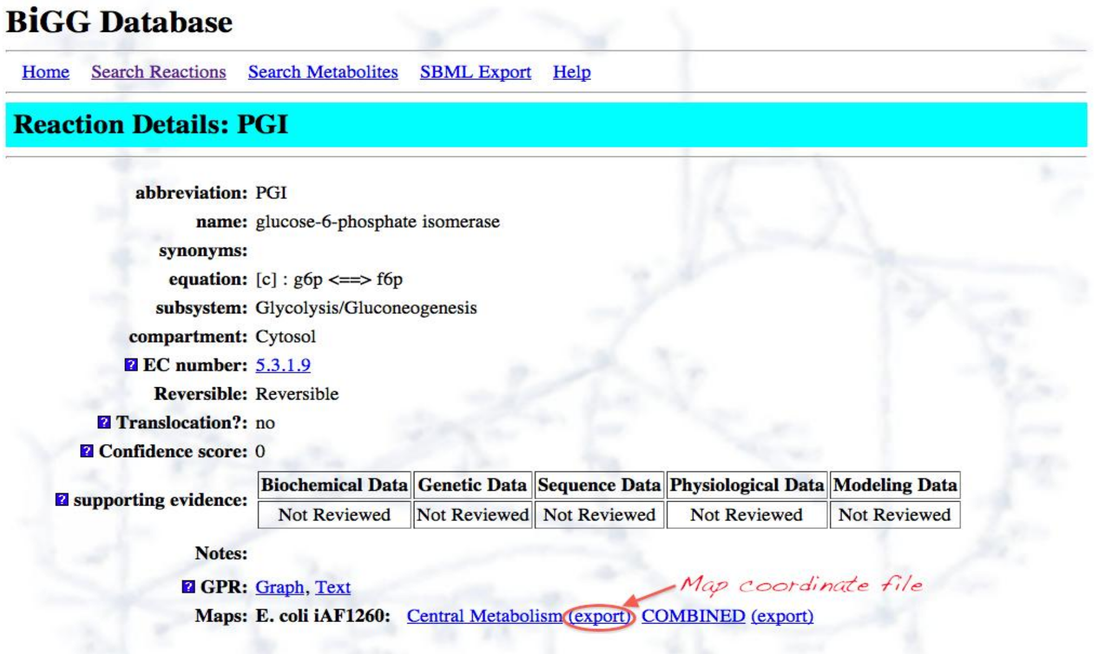
```
<listOfReactions>
    …
    <reaction id=”R_LDH” name=”D-lactate dehydrogenase”>
        <notes>
            <body xmlns=”http://www.w3.org/1999/xhtml”>
                <p>GENE_ASSOCATION: (b1380) or (b2133)</p>
                <p>SUBSYSTEM: Pyruvate Metabolism</p>
                <p>EC Number: 1.1.1.28</p>
                <p>Confidence level: 0</p>
            </body>
        </notes>
        <listOfReactants>
            <speciesReference species=”M_lac_DASH_D_c”/>
            <speciesReference species=”M_nad_c”/>
        </listOfReactants>
        <listOfProducts>
            <speciesReference species=”M_h_c”/>
            <speciesReference species=”M_nadh_c”/>
            <speciesReference species=”M_pyr_c”/>
        </listOfProducts>
        <kineticLaw>
            <math xmlns=”http://www.w3.org/1998/Math/MathML”>
                <ci> FLUX_VALUE </ci>
            </math>
            <listOfParameters>
                <parameter id="LOWER_BOUND" value="-1000" units="mmol_per_gDW_per_hr"/>
                <parameter id="UPPER_BOUND" value="1000" units="mmol_per_gDW_per_hr"/>
                <parameter id="FLUX_VALUE" value="0" units="mmol_per_gDW_per_hr"/>
                <parameter id="OBJECTIVE_COEFFICIENT" value="0" units="mmol_per_gDW_per_hr"/>
            </listOfParameters>
        </kineticLaw>
    </reaction>
    …
</listOfReactions>
```

The gene-protein-reaction associations and subsystem for a reaction are specified in the <notes> section. Reactants and products for a reaction are listed in <listOfReactants> and <listOfProducts> sections respectively. The lower bound, upper bound, and objective coefficient for a reaction are listed in the <listOfParameters> section within the <kineticLaw> section.

Note that Sid entries in an SBML format file are limited to ‘0 to 9’, ‘A to Z’, ‘a to z’, and’ _’. To compensate for this, several substitutions are used in place of certain characters. convertSBMLToCobra and convertCobraToSBML are able to handle translating these substitutions back and forth.

S.2 Description of a COBRA map coordinate file

Map coordinate files for a variety of pathways are available from the BiGG database (http://bigg.ucsd.edu) (Fig. S1)



Reaction Details: PGI

abbreviation: PGI
name: glucose-6-phosphate isomerase
synonyms:
equation: [c] : g6p <==> f6p
subsystem: Glycolysis/Gluconeogenesis
compartment: Cytosol
EC number: 5.3.1.9
Reversible: Reversible
Translocation?: no
Confidence score: 0
supporting evidence:

Biochemical Data	Genetic Data	Sequence Data	Physiological Data	Modeling Data
Not Reviewed	Not Reviewed	Not Reviewed	Not Reviewed	Not Reviewed

Notes:
GPR: Graph, Text
Maps: E. coli iAF1260: Central Metabolism (export) COMBINED (export)

Map coordinate file

Figure S1. Map coordinate files are available for a variety of pathways. If the reaction being explored through the BiGG database is included in a pathway then the map figure and associated coordinate file will be accessible.

The map coordinate files are text tab-delimited files that have four sections: (i) Molecules, (ii) Reaction Nodes, (iii) Reactions, and (iv) Texts. An example map (ecoli_core_map.txt) is available in the testing/testMaps/ folder. The columns for each section are described below:

(i) Molecules

(1) Molecule Id: String. Text used to label the molecule. Used to associate metabolites from the mets field of a COBRA model struct (Section S.3) with the map.

(2) Compartment: Integer indicating which cellular compartment in which the molecule is located. Currently not used in drawing maps.

(3) Primary: Y or N. Y indicates the molecule should be drawn larger. N indicates the molecule should be drawn smaller.

(4) Label X-Coordinate: Integer. Distance in points from upper left corner.

(5) Label Y-Coordinate: Integer. Distance in points from upper left corner.
(6) Molecule X-Coordinate: Integer. Distance in points from upper left corner.
(7) Molecule Y-Coordinate: Integer. Distance in points from upper left corner.
(8) Node Id: Unique Integer for Molecules section. Used to associate molecules with reactions.
(9) Molecule Name: String. Descriptive name for the molecule.

(ii) Reaction Nodes
(1) Empty
(2) Empty
(3) Empty
(4) Reaction X-Coordinate: Integer. Distance in points from upper left corner.
(5) Reaction Y-Coordinate: Integer. Distance in points from upper left corner.
(6) Label X-Coordinate (Optional): Integer. Distance in points from upper left corner.
(7) Label Y-Coordinate (Optional): Integer. Distance in points from upper left corner.
(8) Node Id: Unique Integer for Reaction Nodes section. Used to associate molecules with reactions.
(9) Empty

(iii) Reactions
(1) Reaction Id: String. Text used to label Reaction Node, if Label coordinates are provided for the reaction node. Used to associate reactions from the rxns field of a COBRA model struct (Section S.3) with the map.
(2) Empty
(3) Directionality: Reversible or Irreversible
(4) Start Node: Integer. References a Node Id from Molecules or Reaction Nodes.
(5) End Node: Integer. referencing a Node Id from Molecules or Reaction Nodes.
(6) Empty
(7) Empty
(8) Empty
(9)Empty

(iv) Texts
(1) Style. String. Either a shape (‘Circle’, ‘Line’, ‘Rectangle’) or the Font specification for text. Rows starting with ‘Sans’ are currently not used by the COBRA Toolbox.
(2) Empty
(3) Empty
(4) X-Coordinate for upper left corner of text or shape. Integer.
(5) Y-Coordinate for upper left corner of text or shape. Integer.
(6) Height of text or shape in points. Integer
(7) Width of text or shape in points. Integer
(8) Empty
(9) Formatting Specifications. String. R/G/B:LWD@Type. R/G/B specifies the Red, Green, Blue color scheme; R, G, and B are integers between 1-255. LWD is the line width for the shape and is an Integer >= 1. Type indicates the line type: ‘PLAIN’ specifies a solid line. ‘DASHES’ specifies a dashed line.

S. 3 Description of the COBRA Toolbox model structure

The model structure contains the following required fields:

- rxns: reaction name abbreviation; reaction ID; order corresponds to S matrix.
- mets: metabolite name abbreviation; metabolite ID; order corresponds to S matrix.

- S: Stoichiometric matrix in sparse format.
- rev: logical array; true for reversible reactions, otherwise false
- lb: lower flux bound for corresponding reactions
- ub: upper flux bound for corresponding reactions
- c: objective coefficient for corresponding reactions
- metCharge: value of charge for corresponding metabolite
- metFormulas: Elemental formula for each metabolite
- rules: Boolean rule for the corresponding reaction which defines gene-reaction relationship.
- genes: List of all genes within the model.
- rxnGeneMat: matrix with rows corresponding to reactions and columns corresponding to genes
- grRules: rules field in a format readable format
- subSystems: subSystem assignment for each reaction
- description: A string describing the model (i.e. model name)

Additional fields which contain supplemental information may also be provided using the following fields:

- rxnNames: Full name of each corresponding reaction
- rxnReferences: Cell array of strings which can contain optional information on references for each specific reaction.
- rxnECNumbers: E. C. number for each reaction
- rxnNotes: Cell array of strings which can contain optional information for each specific reaction.
- confidenceScores: Confidence score for each reaction
- proteins: proteins associated with each reaction
- metNames: Full name of each corresponding metabolite
- metChEBIID: ChEBI ID for each corresponding metabolite
- metKeggID: KEGG ID for each corresponding metabolite
- metPubChemID: Pub Chem ID for each corresponding metabolite
- metInchiString: Inichi String for each corresponding metabolite

S. 4 Description of the COBRA format for C13 tracing

Network Description

The matlab structure must contain separate forward and reverse reactions for every Carbon traced reaction. The Carbon information is stored in the following additional field:

.isotopomer: a vector of strings of length n (number of reactions).

For each entry, if the corresponding reaction is carbon tracked, .isotopomer must contain the following string:

Unique_label <space> coefficient/metabolite pairs describing reactants '>' coefficient/metabolite pairs describing products '!' corresponding carbon mapping indicated with '#'.

If there are multiple possible mappings due to compound symmetry, another '!' may follow. All possible mappings are treated with equal probability.

Example:

.isotopomer{i} = ‘CYSS 1 xacser > 1 xac 1 xcysL!#abcde > #de #abc’

Experimental data

Each experiment is stored as a structure comprising the following fields.

.input – vector of length 2^(#carbons in input)
corresponds to cumumer distribution of isotopomers. If only Isotopomer distribution is available then function iso2cdv(input, carbons) will convert to cumomer.

.inputfrag – generated automatically from .input. Contains distribution of C13 in input compound fragments. This field is used by the EMU method.

.fragments – A structure containing the actual measured data. Each field in the structure is one fragment and is identified by a .fragment_name. These fragments contain fields:
.met – string of metabolite measured
.fragment – binary vector of which carbons were measured. A 1 indicates inclusion in a fragment, 0 exclusion.
.data – A vector of length #carbons+1 corresponding to the fraction of carbon labeled 0, 1, … #carbons times.
.metfrag – a string comprising the metabolite name (.met) and .fragment concatenated as a string.

.std2 – The standard deviation of experimental measurements. Used as a scaling factor for goodness of fit testing.

S.5 Description of COBRA problem structures

The required fields for each type of problem structure is listed below. See documentation for additional information. A COBRA model structure requires some manipulation to be converted into a COBRA problem structure. For example, to change an iAF1260 COBRA model structure named model to an COBRA LP problem structure call the following:

```
>> LPproblem = struct('A', model.S, 'lb', model.lb, 'ub', model.ub, 'b', zeros(size(model.S,1),1),
    'osense', -1, 'csense', repmat('E',size(model.S,1),1));
```

S.5.1 Linear Programming problem

The linear programming problem structure must contain the following fields:

- A: Constraints matrix; Left hand side matrix (size m x n)
- lb: Lower bound vector (size n x 1)
- ub: Upper bound vector (size n x 1)
- b: Right hand side matrix (size m x 1)
- osense: Objective sense (-1 max, +1 min) (size 1 x 1)
- csense: Constraints sense ('E' equality, 'G' greater than, 'L' less than). (size m x 1)

S.5.2 Mixed Integer Linear Programming problem

The mixed integer linear programming problem structure must contain the following fields:

- A: Constraints matrix; Left hand side matrix (size m x n)
- lb: Lower bound vector (size n x 1)
- ub: Upper bound vector (size n x 1)
- b: Right hand side matrix (size m x 1)
- osense: Objective sense (-1 max, +1 min) (size 1 x 1)
- csense: Constraints sense ('E' equality, 'G' greater than, 'L' less than). (size m x 1)
- vartype: Variable type vector ('C' continuous, 'I' integer, 'B' binary). (size n x 1)

S.5.3 Quadratic Programming problem

The quadratic programming problem structure must contain the following fields:

- A: Constraints matrix; Left hand side matrix (size m x n)
- lb: Lower bound vector (size n x 1)
- ub: Upper bound vector (size n x 1)
- b: Right hand side matrix (size m x 1)
- osense: Objective sense (-1 max, +1 min) (size 1 x 1)
- csense: Constraints sense ('E' equality, 'G' greater than, 'L' less than). (size m x 1)
- F: Matrix defining quadratic objective (size n x n)

S.5.4 Mixed Integer Quadratic Programming problem

The mixed integer quadratic programming problem must contain the following fields:

- A: Constraints matrix; Left hand side matrix (size m x n)
- lb: Lower bound vector (size n x 1)
- ub: Upper bound vector (size n x 1)
- b: Right hand side matrix (size m x 1)
- osense: Objective sense (-1 max, +1 min) (size 1 x 1)
- csense: Constraints sense ('E' equality, 'G' greater than, 'L' less than). (size m x 1)
- vartype: Variable type vector ('C' continuous, 'I' integer, 'B' binary). (size n x 1)
- F: matrix Defining quadratic objective (size n x n)

S.5.5 Non-linear Programming problem

The non-linear programming problem structure must contain the following fields:

- A: Constraints matrix; Left hand side matrix (size m x n)
- lb: Lower bound vector (size n x 1)
- ub: Upper bound vector (size n x 1)
- b: Right hand side matrix (size m x 1)
- csense: Constraints sense ('E' equality, 'G' greater than, 'L' less than). (size m x 1)
- objFunction: Name of function to evaluate as the objective