Testing chemical and biochemical fidelity

Authors: Ronan Fleming, Ines Thiele, University of Luxembourg.

Reviewer: Almut Heinken, University of Luxembourg.

Introduction

Once a context-specific model is generated, but before a it is used to make predictions of biological relevance, it should be subjected to a range of quantitative and qualitative chemical and biochemical fidelity tests. The stoichiometric consistency tests should not be necessary if one starts with a generic model where the internal reactions are all stoichiometrically consistent then a context-specific model extracted from it should also be stoichiometrically consistent. Beyond chemical fidelity, it is also very important to test biochemical fidelity. Such tests are very specific to the particular biological domain one is modelling. Here we focus on human metabolism and use the Recon3.0model or Recon2.0 model.

PROCEDURE

Load a model

Load Recon2.0model. You may also load your own model.

```
modelFileName = 'Recon2.0model.mat';
modelDirectory = getDistributedModelFolder(modelFileName); %Look up the folder for the
modelFileName= [modelDirectory filesep modelFileName]; % Get the full path. Necessary t
model = readCbModel(modelFileName);
```

Display the size of the model

```
[nMet,nRxn] = size(model.S);
fprintf('%6s\t%6s\n','#mets','#rxns'); fprintf('%6u\t%6u\t%s%s\n',nMet,nRxn,' totals in
```

Set the threshold to classify flux into non-zero and zero flux:

```
threshold=1e-6;
```

Set a solver

```
% changeCobraSolver ('gurobi', 'all', 1);
changeCobraSolver ('glpk', 'all', 1);
```

Production of mthgxl from 12ppd-S

Add sink reactions for either end of the proposed pathway:

```
model = addSinkReactions(model,{'12ppd_S[c]','mthgxl[c]'},[-100 -1; 0 100]);
```

Change the objective to maximise the sink reaction for mthgxl[c]

```
model = changeObjective(model,'sink_mthgxl[c]');
```

Test if it is possible to attain a nonzero objective, and if it is compute a sparse flux vector:

```
sol = optimizeCbModel(model,'max','zero');
```

Check to see if there is a non-zero flux through the objective

```
if sol.stat==1
    fprintf('%g%s\n',sol.v(model.c~=0),' flux through the sink_mthgxl[c] reaction')
end
```

Display the sparse flux solution, but only the non-zero fluxes, above a specified threshold.

```
if sol.stat==1
    for n=1:nRxn
        if abs(sol.v(n))>threshold
            formula=printRxnFormula(model, model.rxns{n}, 0);
            fprintf('%10g%15s\t%-60s\n',sol.v(n),model.rxns{n}, formula{1});
        end
    end
end
```

ANTICIPATED RESULTS

If FBAsol.stat==1 then it is feasible to produce methylglyoxal from (S)-propane-1,2-diol. If FBAsol.stat==0, then this metabolic function is infeasible. This is not anticipated and indicates that further gap filling is required (cf Gap Filling Tutorial).

Metabolic task: 4abut -> succ[m]

Add sink reactions for either end of the proposed pathway:

```
model = addSinkReactions(model,{'4abut[c]','succ[m]'},[-100 -1; 0 100]);
```

Change the objective to maximise the sink reaction for nh4[c]

```
model = changeObjective(model,'sink_succ[m]');
```

Test if it is possible to attain a nonzero objective, and if it is compute a sparse flux vector:

```
sol = optimizeCbModel(model,'max','zero');
```

Check to see if there is a non-zero flux through the objective

```
if sol.stat==1
    fprintf('%g%s\n',sol.v(model.c~=0),' flux through the sink_succ[m] reaction')
end
```

Display the sparse flux solution, but only the non-zero fluxes, above a specified threshold.

```
if sol.stat==1
```

```
    for n=1:nRxn
        if abs(sol.v(n))>threshold
            formula=printRxnFormula(model, model.rxns{n}, 0);
            fprintf('%10g%15s\t%-60s\n',sol.v(n),model.rxns{n}, formula{1});
        end
    end
end
```

ANTICIPATED RESULTS

If FBAsol.stat==1 then it is feasible to produce mitochondrial succinate from 4-Aminobutanoate. If FBAsol.stat==0, then this metabolic function is infeasible. This is not anticipated and indicates that further gap filling is required (cf Gap Filling Tutorial).

Metabolic task: gly -> co2 and nh4 (via glycine cleavage system)

Add sink reactions for either end of the proposed pathway:

```
model = addSinkReactions(model,{'gly[c]','co2[c]','nh4[c]'},[-100 -1; 0.1 100; 0.1 100]
```

Change the objective to maximise the sink reaction for nh4[c]

```
model = changeObjective(model,'sink_nh4[c]');
```

Test if it is possible to attain a nonzero objective, and if it is compute a sparse flux vector:

```
sol = optimizeCbModel(model,'max','zero');
```

Check to see if there is a non-zero flux through the objective

```
if sol.stat==1
    fprintf('%g%s\n',sol.v(model.c~=0),' flux through the sink_nh4[c] reaction')
end
```

Display the sparse flux solution, but only the non-zero fluxes, above a specified threshold.

```
if sol.stat==1
    for n=1:nRxn
        if abs(sol.v(n))>threshold
            formula=printRxnFormula(model, model.rxns{n}, 0);
            fprintf('%10g%15s\t%-60s\n',sol.v(n),model.rxns{n}, formula{1});
        end
    end
end
```

ANTICIPATED RESULTS

If FBAsol.stat==1 then it is feasible to produce CO2 and NH4 from glycine. If FBAsol.stat==0, then this metabolic function is infeasible. This is not anticipated and indicates that further gap filling is required (cf Gap Filling Tutorial).

REFERENCES

[fleming_cardinality_nodate] Fleming, R.M.T., et al., Cardinality optimisation in constraint-based modelling: illustration with Recon 3D (submitted), 2017.

[sparsePaper] Le Thi, H.A., Pham Dinh, T., Le, H.M., and Vo, X.T. (2015). DC approximation approaches for sparse optimization. European Journal of Operational Research 244, 26–46.