Find leakage and siphon modes in a reconstruction

Note: This tutorial is a draft and needs completion. Contributions welcome!

Author: Ronan Fleming, University of Luxembourg

Reviewers:

INTRODUCTION

In standard notation, flux balance analysis is the linear optimisation (1)

$$\begin{aligned} \min_{v} \quad & \rho(v) \equiv c^T v \\ \text{s.t.} \quad & Sv = b, \\ & l \leq v \leq u, \end{aligned}$$

where c is a parameter vector that linearly combines one or more reaction fluxes to form what is termed the objective function and where b represents some fixed output, or input, of species. If b=0 , a species may be produced from nothing (leak species) or consumed for nothing (siphon species) in violation of mass conservation, if all reactions are reversible and S is stoichiometrically consistent. However, even with a stoichiometrically inconsistent S, mass conservation may still not be violated if the lower and upper bounds on reaction rates, l,u prevent it. For instance, the reactions A+B C and C A are stoichiometrically inconsistent because it is impossible to assign a positive molecular mass to all species whilst ensuring that each reaction conserves mass. However, if the bounds are such that both reactions are unidirectional in the forward direction, A+B C and C A , then leakage of B is no longer possible and so mass is conserved.

Testing for leaks or siphons has been widely used in genome-scale metabolic modelling, prior to distribution of a model for prediction of mass conserved steady states. One can identify all leaks in a network with the cardinality optimisation problem (2)

$$\begin{aligned} \min_{v,y} \quad & \|y\|_0 \\ \text{s.t.} \quad & Sv - y = 0, \\ & l \leq v \leq u, \\ & 0 \leq y, \end{aligned}$$

where y_i ≠ 0 indicates that the corresponding species is a leak and can be produced from nothing. To test for siphons, replace the equality constraint in with Sv+y=0. This approach is useful when double checking stoichiometric consistency, with the bounds for all reactions that are known to be stoichiometrically inconsistent set to zero. A minimal number of reactions required to be active and result in a leak of the species subset can be found using the cardinality optimisation problem (3)

$$\begin{aligned} \min_{v,y} \quad & \|v\|_0 \\ \text{s.t.} \quad & Sv - y = 0, \\ & l \leq v \leq u, \\ & 0 \leq y, \\ & 1 \leq y_i,\ i \in \Omega \end{aligned}$$

with the optimal v referred to as a minimal leakage mode [gevorgyan_detection_2008]. A minimal siphon mode would be obtained if the equality constraint was replaced with Sv+y=0. Similarly, a minimal number of reactions contributing to leakage of a minimal number of species can be found using the cardinality optimisation problem (4)

$$\begin{aligned} \min_{v,y} \quad & \|v\|_0 + \|y\|_0 \\ \text{s.t.} \quad & Sv - y = 0, \\ & l \leq v \leq u, \\ & 1 \leq v_j,\ j \in \Omega \\ & 0 \leq y, \end{aligned}$$

where the additional inequality enforces the rate of a subset reactions to be active. The utility of Problems (3) and (4) is in the identification of reactions within the suspect stoichiometrically inconsistent set, to remove prior to reiteration of a test for stoichiometric consistency with a reduced network. This requires manual curation of the results of Problems (3) and (4), where the advantage is that they often narrow down significantly the search for stoichiometrically inconsistent reactions, especially in the context of genome-scale networks. This tutorial illustrates the methods to implement the aforementioned problems when debugging the source of stoichiometric inconsistency within a reconstruction as it can be used to find leak or siphon metabolites, as well as minimal cardinality leakage modes and minimal cardinality siphon modes, of various types.

TIMING

Hours to days, depending on how long it takes to biochemically interpret the minimal leakage and siphon modes.

PROCEDURE

Select reconstruction to convert into a model and enter parameters

Load the ReconX reconstruction, and save the original reconstruction in the workspace, unless it is already loaded into the workspace.

```
clear model
if ~exist('modelOrig','var')
    %select your own model, or use Recon2.0model instead
    if 0
        filename='Recon3.0model';
        directory='~/work/sbgCloud/programReconstruction/projects/recon2models/data/rec
        model = loadIdentifiedModel(filename,directory);
    else
        filename='Recon2.0model.mat';
```

```
        if exist('Recon2.0model.mat','file')==2
            model = readCbModel(filename);
        end
    end
    model.csense(1:size(model.S,1),1)='E';
    modelOrig = model;
else
    model=modelOrig;
end
```

Set the level of printing, zero for silent, higher for more output.

```
printLevel=2;
```

Choose the directory to place the results

```
basePath='~/work/sbgCloud/';
resultsPath=[basePath '/programReconstruction/projects/recon2models/results/reconXs/' m
resultsFileName=[resultsPath filesep model.modelID];
```

Create and enter the folder for the results if it does not already exist

```
if ~exist(resultsPath,'dir')
    mkdir(resultsPath)
end
cd(resultsPath)
```

Optionally create a diary to save the output in case it is very long, this makes it easier to search, especially when debugging the process during the early stages.

```
if 0
    diary([resultsFileName '_diary.txt'])
end
```

Overview some of the key properties of the reconstruction

Noting the initial size of the reconstruction is useful for comparisons later with subsets derived according to mathematical specifications.

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

Find leakage or siphons in heuristically internal part using the bounds given with the model

```
        if 1
            [leakMetBool,leakRxnBool,siphonMetBool,siphonRxnBool,leakY,siphonY,statp,st
                = findMassLeaksAndSiphons(model,model.SIntMetBool,model.SIntRxnBool,mod
        end
```

For each leaking metabolite find a minimal cardinality leakage mode

```
leakParams.epsilon=1e-4;
minLeakParams.eta = getCobraSolverParams('LP', 'feasTol')*100;
leakParams.method='dc';

if 1
    minLeakParams.monoMetMode=1;
    [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
        findMinimalLeakageModeMet(model,leakMetBool,model.SIntRxnBool,modelBoun
end
```

For each siphon metabolite find a minimal cardinality siphon mode

```
if 1
    minLeakParams.monoMetMode=1;
    [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
        findMinimalLeakageModeMet(model,siphonMetBool,model.SIntRxnBool,modelBo
end
```

For each leaking metabolite find a minimal cardinality leakage mode

```
if 1
    minLeakParams.monoMetMode=0;
    [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
        findMinimalLeakageModeMet(model,leakMetBool,model.SIntRxnBool,modelBoun
end
```

For each siphon metabolite find a minimal cardinality siphon mode

```
if 1
    minLeakParams.monoMetMode=0;
    [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
        findMinimalLeakageModeMet(model,siphonMetBool,model.SIntRxnBool,modelBo
end
```

For each heuristically internal but stoichiometrically inconsistent reaction (one at a time), find the min cardinality leakage mode

```
if 1
    rxnBool=model.SIntRxnBool & model.SInConsistentRxnBool;
    metBool=true(nMet,1);
    minLeakParams.monoRxnMode=1;
    [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
        findMinimalLeakageModeRxn(model,rxnBool,metBool,modelBoundsFlag,minLeak
end

% An example of a minimal cardinality leakage mode that causes leakage of no2[c
%
%  #mets     #rxns      mode            rxnAbbr        lb        ub     newlb
%     1         2      leak            RE2704C         0         0         1
```

```
%
% If there are a small number of reactions, the formulae for the set of
% reactions in the minimal leakage mode are displayed:
%
% RE2704C    yvite[c]     ->    h[c] + no2[c] + CE7047[c]
% HMR_6456    h[c] + CE7047[c]     <=>    yvite[c]
%
% If RE2704C goes in the forward direction and HMR_6456 in reverse, then
% no2[c] will leak from the network. One or other of these reactions have
% the incorrect stoichiometry.
%
% Here is a minimal leakage mode resulting in leakage of one metabolite
%  #mets     #rxns      mode            rxnAbbr        lb        ub     newlb
%      1         4      leak           HMR_3288         0         0         1
% FADH2tx    fadh2[c]     <=>    fadh2[x]
% r1290    fadh2[m] + fad[c]     ->    fad[m] + fadh2[c]
% HMR_3288    fad[m] + C05280[m]     ->    fadh2[m] + C05279[m]
% HMR_3321    dece4coa[x]     ->    2 h[x] + dec24dicoa[x]

% If there are a large number of reactions, the formulae for the set of
% reactions in the minimal leakage mode are not displayed. To access such
% reaction and metabolite sets,
%
%
```

For each heuristically internal but stoichiometrically inconsistent reaction, find the min cardinality leakage mode

```
if 1
    rxnBool=model.SIntRxnBool & model.SInConsistentRxnBool;
    metBool=true(nMet,1);
    minLeakParams.monoRxnMode=0;
    [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
        findMinimalLeakageModeRxn(model,rxnBool,metBool,modelBoundsFlag,minLeak
end
```

For each heuristically internal but unknown stoichiometric consistency reaction (one at a time), find a minimal cardinality leakage mode

```
if 1
    rxnBool=model.SIntRxnBool & model.unknownSConsistencyRxnBool;
    metBool=true(nMet,1);
    minLeakParams.monoRxnMode=1;
    [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
end
```

For each heuristically internal but unknown stoichiometric consistency reaction, find a minimal cardinality leakage mode

```
if 1
    rxnBool=model.SIntRxnBool & model.unknownSConsistencyRxnBool;
    metBool=true(nMet,1);
```

```
            minLeakParams.monoRxnMode=0;
            [minLeakMetBool,minLeakRxnBool,minSiphonMetBool,minSiphonRxnBool,leakY,siph
        end

    if printLevel>0
        fprintf('%6u\t%6u\t%s\n',nnz(~model.SIntMetBool),nnz(~model.SIntRxnBool),' heur
        fprintf('%6u\t%6u\t%s\n',nnz(model.SIntMetBool),nnz(model.SIntRxnBool),' All in
        fprintf('%6u\t%6u\t%s\n',nnz(model.fluxConsistentMetBool),nnz(model.fluxConsist
        fprintf('%s\n','Input model assumed to be stoichiometrically and flux consisten
    end
```

REFERENCES

Gevorgyan, A., Poolman, M. G., Fell D., Detection of stoichiometric inconsistencies in biomolecular models. Bioinformatics, 24(19):2245–51, 2008.

Fleming, R.M.T., et al., Cardinality optimisation in constraint-based modelling: Application to Recon 3D (submitted), 2017.

Brunk, E. et al. Recon 3D: A resource enabling a three-dimensional view of gene variation in human metabolism. (submitted) 2017.