Convert a reconstruction into a flux balance analysis model

Author: Ronan Fleming, Ines Thiele, University of Luxembourg

Reviewers:

INTRODUCTION

Even with quality control during the reconstruction process, it is not appropriate to assume that any reconstruction can be converted directly into a model and used to make predictions. A model must satisfy certain assumptions before it can be used to make reliable predictions. Depending on the type of model model, these assumptions will be different. Each assumption should be chemically or biologically motivated and expressed in an unambiguous manner and preferably both intuitively and mathematically. Flux balance analysis is a mathematical method widely used for studying genome-scale biochemical network. Here one aims to predict steady-state reaction fluxes, where there is a balance between production and consumption of each molecular species that is not exchanged across the specified boundary of a system. In this situation, one might obtain erroneous predictions if the system boundary is incorrectly specified. If a reconstruction contains one or more supposedly mass balanced reactions, but which are actually not mass balanced, such reactions in a model can lead to inadvertent leakage of a metabolite from the model, in violation of mass balance. Similarly, when generating a model for flux balance analysis, it is important to ensure that the network is flux consistent, that is, each reaction can carry a non-zero steady state flux.

Given a reconstruction with $\hat{m}$ reactants involved in $\hat{n}$ reactions, this tutorial demonstrates a method to identify and extract the largest subset of the reconstruction whose internal reactions are both stoichoimetrically and flux consistent and whose external reactions are flux consistent. This model is then mathematically consistent with the basic requirements for generation of predictions using flux balance analysis. The identification of the component of the reconstruction that does not satisfy the aforementioned modelling conditions is also useful for targeting reconstruction effort towards resolving stoichiometric inconsistency or resolving flux inconsistency. The example used in this tutorial illustrates the process of extracting a model consistent with flux balance analsis, from a ReconX reconstruction.

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 1
        filename='Recon3D_301.mat'
        load(filename);
        model=Recon3D;
    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
```

```
filename =
'Recon3D_301.mat'
```

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/'
resultsPath=[basePath '/courses/2019_Leiden_COBRA/practicalsDemo/Day4/' model.modelID];
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')
```

```
 #mets     #rxns
```

```
fprintf('%6u\t%6u\t%s\n',nMet,nRxn,' totals.')
```

```
  8399     13543     totals.
```

Make sure the stoichiometric matrix is stored in a sparse format as this accelerates computations with large networks

```
model.S=sparse(model.S);
```

Check in case the reconstruction is a model that is already ready for flux balance analysis

There is no need to run this live script any further if the reconstruction already satisfies the conditions necessary for flux balance analysis. That is if all internal reactants and reactions are stoichiometrically consistent, and all reactions are flux consistent, then the reconstruction satisfies the criteria to designate it a model ready for flux balance analysis.

SIntMetBool	m x 1 Boolean of metabolites heuristically though to be involved in mass balanced reactions.
SIntRxnBool	n x 1 Boolean of reactions heuristically though to be mass balanced.
SConsistentMetBool	m x 1 Boolean vector indicating consistent mets
SConsistentRxnBool	n x 1 Boolean vector indicating consistent rxns
fluxConsistentMetBool	m x 1 Boolean vector indicating flux consistent mets
fluxConsistentRxnBool	n x 1 Boolean vector indicating flux consistent rxns

```
if all(isfield(model,{'SIntMetBool','SIntRxnBool','SConsistentMetBool',...
        'SConsistentRxnBool','fluxConsistentMetBool','fluxConsistentRxnBool'}))
    if all(model.SIntMetBool & model.SConsistentMetBool)...
            && nnz(model.SIntRxnBool & model.SConsistentRxnBool)==nnz(model.SIntRxnBool
            && all(model.fluxConsistentMetBool)...
            && all(model.fluxConsistentRxnBool)
        fullyStoichAndFluxConsistent=1;
        fprintf('%s\n','Reconstruction is a model that is already ready for flux balanc
    end
    return
else
    fullyStoichAndFluxConsistent=0;
    fprintf('%s\n','Reconstruction must be tested to check if it is ready for flux bala
end
```

```
Reconstruction must be tested to check if it is ready for flux balance analysis
```

Manually remove certain reactions from the reconstruction

Before attempting to algorithmically remove stoichiometrically or flux inconsistent supposed internal reactions from a reconstruction to generate a model, there is an option to review the content of the reconstruction and manually identify reactions for removal. That is, there are two options:

A. Skip manual review of reconstruction content. Move to the next step.

B. Review the content of the reconstruction and omit any reactions that are assumed to be stoichiometrically or flux inconsistent. With respect to stoichiometric inconsistency, such reactions may be obviously mass imbalanced and not satisfy the heuristic conditions for indentification as an exernal reaction. Alternatively, such reactions may be identified by a previous pass through of this tutorial as being of unknown stoichometric consistent (model.unknownSConsistencyRxnBool(j)==1), after the largest stoichiometrically consistent subset of

the network has been is identified. This is an iterative process where multiple rounds of identification of the largest stoichiometrically consistent set and manual curation of the remainder that is of unknown stoichiometric consistency is necessary.

```
if strcmp(filename,'Recon3.0model')
    modelOrig=model;
    if 0
        if 1
            %Rename some of the biomass reactions to make them more obviously exchange
            %reactions
            model.rxns{strcmp(model.rxns,'biomass_reaction')}= 'EX_biomass_reaction';
            model.rxns{strcmp(model.rxns,'biomass_maintenance')}= 'EX_biomass_maintenar
            model.rxns{strcmp(model.rxns,'biomass_maintenance_noTrTr')}= 'EX_biomass_ma

            %ATP hydrolysis is not imbalanced like all the other demand reactions so
            %give it a different accronym ATPM = ATP Maintenance
            bool=strcmp('DM_atp_c_',model.rxns);
            model.rxns{bool}='ATPM';
        end
        [model,removeMetBool,removeRxnBool] = manuallyAdaptRecon3(model,printLevel);
    else
        [model,removeMetBool,removeRxnBool] = manuallyAdaptRecon3Ines(model,printLevel)
    end
    [nMet0,nRxn0]=size(modelOrig.S);
    [nMet,nRxn]=size(model.S);
    if nMet0==nMet && nRxn0==nRxn && printLevel>0
        fprintf('%s\n','--- Manually removing rows and columns of the stoichiometric ma
        fprintf('%6s\t%6s\n','#mets','#rxns')
        fprintf('%6u\t%6u\t%s\n',nMet0,nRxn0,' totals.')
        fprintf('%6u\t%6u\t%s\n',nMet0-nMet,nRxn0-nRxn,' manually removed.')
        fprintf('%6u\t%6u\t%s\n',nMet,nRxn,' remaining.')
    end
end
```

Remove any trivial rows and columns of the stoichiometric matrix

Remove any zero rows or columns of the stoichiometric matrix

```
modelOrig=model;
model=removeTrivialStoichiometry(model);
[nMet0,nRxn0]=size(modelOrig.S);
[nMet,nRxn]=size(model.S);
if nMet0==nMet && nRxn0==nRxn && printLevel>0
    fprintf('%s\n','---Checking for Remove any trivial rows and columns of the stoichio
    fprintf('%6s\t%6s\n','#mets','#rxns')
    fprintf('%6u\t%6u\t%s\n',nMet0,nRxn0,' totals.')
    fprintf('%6u\t%6u\t%s\n',nMet0-nMet,nRxn0-nRxn,' duplicates removed.')
    fprintf('%6u\t%6u\t%s\n',nMet,nRxn,' remaining.')
end
```

```
---Checking for Remove any trivial rows and columns of the stoichiometric matrix----
 #mets     #rxns
  8399     13543     totals.
```

```
     0         0     duplicates removed.
  8399     13543     remaining.
```

Check for duplicate columns by detecting the columns of the S matrix that are identical upto scalar multiplication.

```
modelOrig=model;
dupDetectMethod='FR';
dupDetectMethod='S';
removeFlag=0;
[modelOut,removedRxnInd, keptRxnInd] = checkDuplicateRxn(model,dupDetectMethod,removeFl
```

Remove any duplicate reactions, and uniquely involved reactants, from the stoichiometric matrix.

```
if length(removedRxnInd)>0
    irrevFlag=0;
    metFlag=1;
    %set all reactions reversible that are duplicates
    model.lb(removedRxnInd)=-model.ub(removedRxnInd);
    %remove duplicates
    model = removeRxns(model,model.rxns(removedRxnInd),irrevFlag,metFlag);
end
```

Display the statistics on the duplicate reactions,

```
[nMet0,nRxn0]=size(modelOrig.S);
[nMet,nRxn]=size(model.S);
if nMet0==nMet && nRxn0==nRxn && printLevel>0
    fprintf('%s\n','---Remove any duplicate reactions----')
    [nMet0,nRxn0]=size(modelOrig.S);
    [nMet,nRxn]=size(model.S);
    fprintf('%6s\t%6s\n','#mets','#rxns')
    fprintf('%6u\t%6u\t%s\n',nMet0,nRxn0,' totals.')
    fprintf('%6u\t%6u\t%s\n',nMet0-nMet,nRxn0-nRxn,' duplicates removed.')
    fprintf('%6u\t%6u\t%s\n',nMet,nRxn,' remaining.')
end
```

```
---Remove any duplicate reactions----
 #mets     #rxns
  8399     13543     totals.
     0         0     duplicates removed.
  8399     13543     remaining.
```

Remove any duplicate reactions upto protons

Remove reactions reactions that differ only in the number of protons involved as substrates or products. Also remove exclusively involved reactants.

Save a temporary model for testing, before making any changes.

```
modelH=model;
```

Find the proton indicies in different compartments. A proton, with index i, is asumed to be represented by an abbreviation within model.mets{i} like h[*], where * denotes the compartment symbol.

```
nMetChars=zeros(length(modelH.mets),1);
for m=1:length(modelH.mets)
    nMetChars(m,1)=length(modelH.mets{m});
end
protonMetBool=strncmp(modelH.mets,'h',1) & nMetChars==length('h[*]');
if printLevel>2
    disp(modelH.mets(protonMetBool))
end
```

Zero out the proton stoichiometric coefficients from the temporary model for testing

```
modelH.S(protonMetBool,:)=0;
```

Check for duplicate columns, upto protons, by detecting the columns of the S matrix that are identical upto scalar multiplication.

```
dupDetectMethod='FR';
removeFlag=0;
[modelOut,removedRxnInd, keptRxnInd] = checkDuplicateRxn(modelH,dupDetectMethod,removeF
```

```
Checking for reaction duplicates by stoichiometry (up to orientation) ...
     Keep:     25HVITD2t    25hvitd2[c]     ->    25hvitd2[e]
Duplicate:     25HVITD2tin    25hvitd2[e]     ->    25hvitd2[c]
     Keep:     25HVITD2tin_m    25hvitd2[c]     ->    25hvitd2[m]
Duplicate:     25HVITD2tm    25hvitd2[m]     ->    25hvitd2[c]
     Keep:     25HVITD3t    25hvitd3[c]     ->    25hvitd3[e]
Duplicate:     25HVITD3tin    25hvitd3[e]     ->    25hvitd3[c]
     Keep:     25HVITD3tin_m    25hvitd3[c]     ->    25hvitd3[m]
Duplicate:     25HVITD3tm    25hvitd3[m]     ->    25hvitd3[c]
     Keep:     3MOBt2im    3mob[c]     ->    3mob[m]
Duplicate:     HMR_3746    3mob[c]     <=>    3mob[m]
     Keep:     5MTHFt    5mthf[e]     <=>    5mthf[c]
Duplicate:     MTHFTe    5mthf[c]     ->    5mthf[e]
     Keep:     ADNt    adn[e]     <=>    adn[c]
Duplicate:     ADNCNT3tc    adn[e]     <=>    adn[c]
     Keep:     ADPRIBt    adprib[e]     ->    adprib[c]
Duplicate:     ADPRIBte    adprib[c]     <=>    adprib[e]
     Keep:     ALAt4    na1[e] + ala_L[e]     ->    na1[c] + ala_L[c]
Duplicate:     HMR_9605    na1[e] + ala_L[e]     ->    na1[c] + ala_L[c]
     Keep:     ALCD21_D    nad[c] + 12ppd_R[c]     ->    nadh[c] + lald_D[c]
Duplicate:     PPDOx    nadh[c] + lald_D[c]     ->    nad[c] + 12ppd_R[c]
     Keep:     ALCD22_D    nad[c] + lald_D[c]     ->    nadh[c] + mthgxl[c]
Duplicate:     LALDO2x    nadh[c] + mthgxl[c]     ->    nad[c] + lald_D[c]
     Keep:     ATPasel    h2o[c] + atp[c]     ->    adp[c] + pi[c]
Duplicate:     DM_atp_c_    h2o[c] + atp[c]     ->    adp[c] + pi[c]
Warning: BTNt2 has more than one replicate
     Keep:     BTNt2    btn[e]     <=>    btn[c]
Duplicate:     BTNTe    btn[c]     ->    btn[e]
     Keep:     C14STRr    nadph[r] + 44mctr[r]     ->    nadp[r] + 44mzym[r]
Duplicate:     r0780    nadp[r] + 44mzym[r]     <=>    nadph[r] + 44mctr[r]
     Keep:     C160CPT1    crn[c] + pmtcoa[c]     <=>    coa[c] + pmtcrn[c]
Duplicate:     C160CPT2rbc    coa[c] + pmtcrn[c]     <=>    crn[c] + pmtcoa[c]
     Keep:     C161CPT2    coa[m] + hdcecrn[m]     <=>    crn[m] + hdcoa[m]
Duplicate:     r0446    crn[m] + hdcoa[m]     <=>    coa[m] + hdcecrn[m]
     Keep:     C181CPT1    crn[c] + odecoa[c]     <=>    coa[c] + odecrn[c]
```

```
Duplicate:     C181CPT2rbc    coa[c] + odecrn[c]     <=>    crn[c] + odecoa[c]
     Keep:     CITtam    cit[c] + mal_L[m]     <=>    cit[m] + mal_L[c]
Duplicate:     HMR_4964    cit[c] + mal_L[m]     ->    cit[m] + mal_L[c]
     Keep:     CRNt    crn[e]     <=>    crn[c]
Duplicate:     CRNtHa    crn[c]     ->    crn[e]
     Keep:     CRNtuNa    na1[e] + crn[e]     ->    na1[c] + crn[c]
Duplicate:     CRNCT2te    na1[c] + crn[c]     <=>    na1[e] + crn[e]
     Keep:     CRVNCtr    crvnc[e]     <=>    crvnc[c]
Duplicate:     CE0328te    crvnc[c]     <=>    crvnc[e]
     Keep:     CYSt4    na1[e] + cys_L[e]     ->    na1[c] + cys_L[c]
Duplicate:     CYSSNAT5tc    na1[e] + cys_L[e]     <=>    na1[c] + cys_L[c]
     Keep:     CYTDt    cytd[e]     <=>    cytd[c]
Duplicate:     CYTDt2r    cytd[e]     <=>    cytd[c]
     Keep:     DALAt2r    ala_D[e]     <=>    ala_D[c]
Duplicate:     ALA-DTDe    ala_D[c]     ->    ala_D[e]
     Keep:     DMHPTCRNte    dmhptcrn[c]     <=>    dmhptcrn[e]
Duplicate:     DMHPTCRNtr    dmhptcrn[e]     <=>    dmhptcrn[c]
     Keep:     DNDPt10m    dadp[c] + dcdp[m]     ->    dcdp[c] + dadp[m]
Duplicate:     DNDPt29m    dcdp[c] + dadp[m]     ->    dadp[c] + dcdp[m]
     Keep:     DNDPt11m    dadp[c] + dgdp[m]     ->    dgdp[c] + dadp[m]
Duplicate:     DNDPt35m    dgdp[c] + dadp[m]     ->    dadp[c] + dgdp[m]
     Keep:     DNDPt14m    dtdp[m] + dudp[c]     ->    dtdp[c] + dudp[m]
Duplicate:     DNDPt22m    dtdp[c] + dudp[m]     ->    dtdp[m] + dudp[c]
     Keep:     DNDPt15m    dgdp[m] + dudp[c]     ->    dgdp[c] + dudp[m]
Duplicate:     DNDPt33m    dgdp[c] + dudp[m]     ->    dgdp[m] + dudp[c]
     Keep:     DNDPt16m    dadp[m] + dudp[c]     ->    dadp[c] + dudp[m]
Duplicate:     DNDPt8m    dadp[c] + dudp[m]     ->    dadp[m] + dudp[c]
     Keep:     DNDPt17m    dcdp[m] + dudp[c]     ->    dcdp[c] + dudp[m]
Duplicate:     DNDPt26m    dcdp[c] + dudp[m]     ->    dcdp[m] + dudp[c]
     Keep:     DNDPt23m    dgdp[m] + dtdp[c]     ->    dgdp[c] + dtdp[m]
Duplicate:     DNDPt34m    dgdp[c] + dtdp[m]     ->    dgdp[m] + dtdp[c]
     Keep:     DNDPt24m    dadp[m] + dtdp[c]     ->    dadp[c] + dtdp[m]
Duplicate:     DNDPt9m    dadp[c] + dtdp[m]     ->    dadp[m] + dtdp[c]
     Keep:     DNDPt25m    dcdp[m] + dtdp[c]     ->    dcdp[c] + dtdp[m]
Duplicate:     DNDPt27m    dcdp[c] + dtdp[m]     ->    dcdp[m] + dtdp[c]
     Keep:     DNDPt28m    dcdp[c] + dgdp[m]     ->    dgdp[c] + dcdp[m]
Duplicate:     DNDPt36m    dgdp[c] + dcdp[m]     ->    dcdp[c] + dgdp[m]
     Keep:     DOPAtu    dopa[e]     <=>    dopa[c]
Duplicate:     DOPAENT4tc    dopa[e]     <=>    dopa[c]
     Keep:     EBP2r    zymstnl[r]     ->    lthstrl[r]
Duplicate:     r1381    lthstrl[r]     <=>    zymstnl[r]
     Keep:     FE2t    fe2[e]     ->    fe2[c]
Duplicate:     FE2DMT1    fe2[e]     ->    fe2[c]
     Keep:     FE2tm    fe2[c]     ->    fe2[m]
Duplicate:     HMR_5420    fe2[c]     ->    fe2[m]
     Keep:     FUCFUCFUCGALACGLC13GALACGLCGAL14ACGLCGALGLUSIDEte    fucfucfucgalacglc13galacglcgal14acglcg
Duplicate:     HMR_9651    fucfucfucgalacglc13galacglcgal14acglcgalgluside_hs[c]     <=>    fucfucfucgalac
     Keep:     FUCFUCFUCGALACGLCGAL14ACGLCGALGLUSIDEte    fucfucfucgalacglcgal14acglcgalgluside_hs[e]
Duplicate:     HMR_9645    fucfucfucgalacglcgal14acglcgalgluside_hs[c]     <=>    fucfucfucgalacglcgal14ac
     Keep:     FUCGALFUCGALACGLCGALGLUSIDEte    fucgalfucgalacglcgalgluside_hs[e]     <=>    fucgalfucgala
Duplicate:     HMR_9643    fucgalfucgalacglcgalgluside_hs[c]     <=>    fucgalfucgalacglcgalgluside_hs[e]
     Keep:     GALFUCGALACGLCGAL14ACGLCGALGLUSIDEte    galfucgalacglcgal14acglcgalgluside_hs[e]     <=>
Duplicate:     HMR_9646    galfucgalacglcgal14acglcgalgluside_hs[c]     <=>    galfucgalacglcgal14acglcgal
     Keep:     GALt1r    gal[e]     <=>    gal[c]
Duplicate:     GALt2_2    gal[e]     <=>    gal[c]
     Keep:     GDPtg    gdp[c]     <=>    gdp[g]
Duplicate:     HMR_7743    gdp[c]     <=>    gdp[g]
Warning: GLCt1r has more than one replicate
     Keep:     GLCt1r    glc_D[e]     <=>    glc_D[c]
Duplicate:     GLCGLUT2    glc_D[c]     ->    glc_D[e]
     Keep:     GLNtm    gln_L[c]     ->    gln_L[m]
Duplicate:     HMR_5101    gln_L[c]     ->    gln_L[m]
     Keep:     GLYC3Ptm    glyc3p[c]     ->    glyc3p[m]
Duplicate:     GLYC3Ptmc    glyc3p[m]     <=>    glyc3p[c]
```

```
     Keep:     GLYt4    na1[e] + gly[e]     ->    na1[c] + gly[c]
Duplicate:     GLYSNAT5tc    na1[e] + gly[e]     <=>    na1[c] + gly[c]
     Keep:     GSNt    gsn[e]     <=>    gsn[c]
Duplicate:     GSNt2r    gsn[e]     <=>    gsn[c]
     Keep:     HISt4    na1[e] + his_L[e]     ->    na1[c] + his_L[c]
Duplicate:     HISSNAT5tc    na1[e] + his_L[e]     <=>    na1[c] + his_L[c]
     Keep:     HIStiDF    his_L[e]     ->    his_L[c]
Duplicate:     HISCAT1    his_L[c]     <=>    his_L[e]
     Keep:     HSD17B7r    nadph[r] + estrone[r]     ->    nadp[r] + estradiol[r]
Duplicate:     HMR_2041    nadph[r] + estrone[r]     ->    nadp[r] + estradiol[r]
Warning: Htg has more than one replicate
     Keep:     Htg        <=>
Duplicate:     Htmi        ->
     Keep:     INSt    ins[e]     <=>    ins[c]
Duplicate:     INSt2    ins[e]     <=>    ins[c]
     Keep:     L_LACtcm    lac_L[c]     ->    lac_L[m]
Duplicate:     L_LACtm    lac_L[c]     ->    lac_L[m]
     Keep:     LNLCCPT1    crn[c] + lnlccoa[c]     <=>    coa[c] + lnlccrn[c]
Duplicate:     LNLCCPT2rbc    coa[c] + lnlccrn[c]     <=>    crn[c] + lnlccoa[c]
Warning: NACUP has more than one replicate
     Keep:     NACUP    nac[e]     ->    nac[c]
Duplicate:     NACHORCTL3le    nac[e]     ->    nac[c]
     Keep:     NADHtpu    nadh[c]     ->    nadh[x]
Duplicate:     NADtpu    nadh[x]     ->    nadh[c]
     Keep:     NAt    na1[e]     <=>    na1[c]
Duplicate:     NAt3_1    na1[c]     <=>    na1[e]
     Keep:     NCAMUP    ncam[e]     ->    ncam[c]
Duplicate:     NCAMDe    ncam[c]     ->    ncam[e]
     Keep:     NH4t3r    nh4[c]     <=>    nh4[e]
Duplicate:     NH4tb    nh4[e]     <=>    nh4[c]
     Keep:     NOt    no[e]     <=>    no[c]
Duplicate:     NODe    no[c]     <=>    no[e]
     Keep:     OCTAt    octa[e]     <=>    octa[c]
Duplicate:     OCTAte    octa[c]     <=>    octa[e]
Warning: ORNt4m has more than one replicate
     Keep:     ORNt4m    orn[m] + citr_L[c]     <=>    orn[c] + citr_L[m]
Duplicate:     r2412    orn[c] + citr_L[m]     ->    orn[m] + citr_L[c]
     Keep:     P5CRxm    nadh[m] + 1pyr5c[m]     ->    nad[m] + pro_L[m]
Duplicate:     PRO1xm    nad[m] + pro_L[m]     ->    nadh[m] + 1pyr5c[m]
     Keep:     PItx    pi[c]     <=>    pi[x]
Duplicate:     HMR_5344    pi[c]     <=>    pi[x]
     Keep:     PRODt2r    pro_D[e]     <=>    pro_D[c]
Duplicate:     PRO_Dtde    pro_D[c]     <=>    pro_D[e]
     Keep:     RIBt    rib_D[e]     <=>    rib_D[c]
Duplicate:     RIBt2    rib_D[e]     ->    rib_D[c]
     Keep:     SRTNtu    srtn[e]     <=>    srtn[c]
Duplicate:     SRTNENT4tc    srtn[e]     <=>    srtn[c]
     Keep:     SUCCtp    succ[c]     <=>    succ[x]
Duplicate:     SUCCTD    succ[x]     <=>    succ[c]
     Keep:     TAGt    tag_hs[e]     <=>    tag_hs[c]
Duplicate:     TAGHSTDe    tag_hs[c]     ->    tag_hs[e]
Warning: THYMDt1 has more than one replicate
     Keep:     THYMDt1    thymd[e]     ->    thymd[c]
Duplicate:     THMDt2r    thymd[e]     <=>    thymd[c]
     Keep:     TRDRm    nadph[m] + trdox[m]     ->    nadp[m] + trdrd[m]
Duplicate:     r1433    nadp[m] + trdrd[m]     ->    nadph[m] + trdox[m]
     Keep:     URIt    uri[e]     <=>    uri[c]
Duplicate:     URIt2r    uri[e]     <=>    uri[c]
     Keep:     VITD3t    vitd3[c]     ->    vitd3[e]
Duplicate:     VITD3t2    vitd3[e]     ->    vitd3[c]
Warning: VITD3tm has more than one replicate
     Keep:     VITD3tm    vitd3[m]     ->    vitd3[c]
Duplicate:     HMR_2116    vitd3[c]     <=>    vitd3[m]
     Keep:     XOLEST2te    xolest2_hs[e]     <=>    xolest2_hs[c]
```

```
Duplicate:     XOLEST2HSTDle    xolest2_hs[c]     ->    xolest2_hs[e]
     Keep:     r0276    nh4[c] + nadp[c] + imp[c]     <=>    nadph[c] + gmp[c]
Duplicate:     GMPR    nadph[c] + gmp[c]     ->    nh4[c] + nadp[c] + imp[c]
     Keep:     r0488    2 nadp[c] + coa[c] + mev_R[c]     <=>    2 nadph[c] + hmgcoa[c]
Duplicate:     HMGCOARc    2 nadph[c] + hmgcoa[c]     ->    2 nadp[c] + coa[c] + mev_R[c]
     Keep:     r0537    ethamp[c] + hxdcal[c]     ->    sph1p[c]
Duplicate:     SGPL11c    sph1p[c]     ->    ethamp[c] + hxdcal[c]
     Keep:     r0561    coa[m] + 2mpdhl[m]     ->    ibcoa[m] + dhlam[m]
Duplicate:     RE3326M    ibcoa[m] + dhlam[m]     <=>    coa[m] + 2mpdhl[m]
     Keep:     r0808    HC00004[c]     ->    HC00004[e]
Duplicate:     HC00004t1e    HC00004[e]     ->    HC00004[c]
     Keep:     r0817    citr_L[c]     <=>    citr_L[e]
Duplicate:     CITRtr    citr_L[e]     <=>    citr_L[c]
     Keep:     r0839    orot[e]     <=>    orot[c]
Duplicate:     OROte    orot[e]     ->    orot[c]
     Keep:     r0899    ala_B[c]     <=>    ala_B[e]
Duplicate:     BALAPAT1tc    ala_B[e]     ->    ala_B[c]
     Keep:     r0913    icit[m] + mal_L[c]     <=>    mal_L[m] + icit[c]
Duplicate:     r2387    mal_L[m] + icit[c]     ->    icit[m] + mal_L[c]
     Keep:     r0915    cit[c] + succ[m]     <=>    cit[m] + succ[c]
Duplicate:     r2382    cit[c] + succ[m]     ->    cit[m] + succ[c]
     Keep:     r0944    spmd[c]     <=>    spmd[e]
Duplicate:     SPMTDe    spmd[e]     <=>    spmd[c]
     Keep:     r1050    chsterol[e]     <=>    chsterol[c]
Duplicate:     CHOLESTTDe    chsterol[c]     ->    chsterol[e]
     Keep:     r1067    his_L[l]     ->    his_L[c]
Duplicate:     HIShPTtc    his_L[l]     ->    his_L[c]
     Keep:     r1078    tyr_L[c]     ->    tyr_L[m]
Duplicate:     HMR_5099    tyr_L[c]     <=>    tyr_L[m]
     Keep:     r1127    HC00005[c]     ->    HC00005[r]
Duplicate:     HC00005t1r    HC00005[r]     ->    HC00005[c]
     Keep:     r1128    HC00009[c]     ->    HC00009[r]
Duplicate:     HC00009t1r    HC00009[r]     ->    HC00009[c]
     Keep:     r1129    HC00004[c]     ->    HC00004[r]
Duplicate:     HC00004t1r    HC00004[r]     ->    HC00004[c]
     Keep:     r1131    HC00006[c]     ->    HC00006[r]
Duplicate:     HC00006t1r    HC00006[r]     ->    HC00006[c]
     Keep:     r1132    HC00007[c]     ->    HC00007[r]
Duplicate:     HC00007t1r    HC00007[r]     ->    HC00007[c]
     Keep:     r1133    HC00008[c]     ->    HC00008[r]
Duplicate:     HC00008t1r    HC00008[r]     ->    HC00008[c]
     Keep:     r1147    akg[c] + icit[m]     <=>    akg[m] + icit[c]
Duplicate:     r2385    akg[m] + icit[c]     ->    akg[c] + icit[m]
     Keep:     r1155    2obut[c]     ->    2obut[m]
Duplicate:     r1454    2obut[m]     ->    2obut[c]
     Keep:     r1423    pi[c]     ->    pi[e]
Duplicate:     PIt6b    pi[e]     <=>    pi[c]
     Keep:     r1427    his_L[c]     ->    his_L[m]
Duplicate:     r2416    his_L[m]     ->    his_L[c]
     Keep:     r1429    glyc3p[c]     <=>    glyc3p[x]
Duplicate:     GLY3Pt    glyc3p[x]     ->    glyc3p[c]
     Keep:     r1441    trdrd[c]     ->    trdrd[m]
Duplicate:     HMR_6618    trdrd[c]     <=>    trdrd[m]
     Keep:     r1455    phe_L[c]     ->    phe_L[m]
Duplicate:     r1456    phe_L[m]     ->    phe_L[c]
     Keep:     r1618    tyr_L[c] + phe_L[e]     <=>    phe_L[c] + tyr_L[e]
Duplicate:     TYRPHELAT2tc    phe_L[c] + tyr_L[e]     ->    tyr_L[c] + phe_L[e]
     Keep:     r1619    cys_L[c] + phe_L[e]     <=>    cys_L[e] + phe_L[c]
Duplicate:     CYSPHELAT2tc    cys_L[e] + phe_L[c]     ->    cys_L[c] + phe_L[e]
     Keep:     r1620    leu_L[c] + phe_L[e]     <=>    leu_L[e] + phe_L[c]
Duplicate:     LEUPHELAT2tc    leu_L[e] + phe_L[c]     <=>    leu_L[c] + phe_L[e]
     Keep:     r1622    asn_L[c] + phe_L[e]     <=>    asn_L[e] + phe_L[c]
Duplicate:     ASNPHELAT2tc    asn_L[e] + phe_L[c]     ->    asn_L[c] + phe_L[e]
     Keep:     r1623    phe_L[e] + val_L[c]     <=>    phe_L[c] + val_L[e]
```

```
Duplicate:     VALPHELAT2tc    phe_L[c] + val_L[e]     ->    phe_L[e] + val_L[c]
     Keep:     r1624    thr_L[c] + phe_L[e]     <=>    thr_L[e] + phe_L[c]
Duplicate:     THRPHELAT2tc    thr_L[e] + phe_L[c]     ->    thr_L[c] + phe_L[e]
     Keep:     r1626    ile_L[c] + phe_L[e]     <=>    ile_L[e] + phe_L[c]
Duplicate:     ILEPHELAT2tc    ile_L[e] + phe_L[c]     ->    ile_L[c] + phe_L[e]
     Keep:     r1644    leu_L[e] + val_L[c]     <=>    leu_L[c] + val_L[e]
Duplicate:     VALLAT1tc    leu_L[c] + val_L[e]     ->    leu_L[e] + val_L[c]
     Keep:     r1647    ile_L[c] + leu_L[e]     <=>    ile_L[e] + leu_L[c]
Duplicate:     ILELAT1tc    ile_L[e] + leu_L[c]     ->    ile_L[c] + leu_L[e]
     Keep:     r1668    arg_L[e] + his_L[c]     <=>    arg_L[c] + his_L[e]
Duplicate:     HISyLATthc    arg_L[c] + his_L[e]     ->    arg_L[e] + his_L[c]
     Keep:     r2009    ala_L[c] + arg_L[e]     ->    ala_L[e] + arg_L[c]
Duplicate:     ALAyLATthc    ala_L[e] + arg_L[c]     ->    ala_L[c] + arg_L[e]
     Keep:     r2010    gln_L[c] + arg_L[e]     ->    gln_L[e] + arg_L[c]
Duplicate:     GLNyLATthc    gln_L[e] + arg_L[c]     ->    gln_L[c] + arg_L[e]
     Keep:     r2012    arg_L[e] + met_L[c]     ->    arg_L[c] + met_L[e]
Duplicate:     METyLATthc    arg_L[c] + met_L[e]     ->    arg_L[e] + met_L[c]
     Keep:     r2014    arg_L[e] + phe_L[c]     ->    arg_L[c] + phe_L[e]
Duplicate:     PHEyLATthc    arg_L[c] + phe_L[e]     ->    arg_L[e] + phe_L[c]
     Keep:     r2017    arg_L[e] + leu_L[c]     ->    arg_L[c] + leu_L[e]
Duplicate:     LEUyLAThtc    arg_L[c] + leu_L[e]     ->    arg_L[e] + leu_L[c]
     Keep:     r2073    zn2[e]     ->    zn2[c]
Duplicate:     r2465    zn2[c]     ->    zn2[e]
     Keep:     r2346    wharachd[e]     <=>    wharachd[c]
Duplicate:     WHARACHDtd    wharachd[c]     <=>    wharachd[e]
     Keep:     r2355    HC02203[e]     <=>    HC02203[c]
Duplicate:     C05953td    HC02203[c]     <=>    HC02203[e]
     Keep:     r2364    HC02213[e]     <=>    HC02213[c]
Duplicate:     C06439td    HC02213[c]     <=>    HC02213[e]
     Keep:     r2373    akg[c] + cit[m]     <=>    akg[m] + cit[c]
Duplicate:     r2381    akg[m] + cit[c]     ->    akg[c] + cit[m]
     Keep:     r2374    cit[m] + oxa[c]     <=>    cit[c] + oxa[m]
Duplicate:     r2384    cit[c] + oxa[m]     ->    cit[m] + oxa[c]
     Keep:     r2375    icit[m] + succ[c]     <=>    icit[c] + succ[m]
Duplicate:     r2386    icit[c] + succ[m]     ->    icit[m] + succ[c]
     Keep:     r2376    icit[m] + oxa[c]     <=>    icit[c] + oxa[m]
Duplicate:     r2388    icit[c] + oxa[m]     ->    icit[m] + oxa[c]
     Keep:     r2377    akg[c] + HC00342[m]     <=>    akg[m] + HC00342[c]
Duplicate:     r2389    akg[m] + HC00342[c]     ->    akg[c] + HC00342[m]
     Keep:     r2378    succ[c] + HC00342[m]     <=>    succ[m] + HC00342[c]
Duplicate:     r2390    succ[m] + HC00342[c]     ->    succ[c] + HC00342[m]
     Keep:     r2379    mal_L[c] + HC00342[m]     <=>    mal_L[m] + HC00342[c]
Duplicate:     r2391    mal_L[m] + HC00342[c]     ->    mal_L[c] + HC00342[m]
     Keep:     r2380    oxa[c] + HC00342[m]     <=>    HC00342[c] + oxa[m]
Duplicate:     r2392    HC00342[c] + oxa[m]     ->    oxa[c] + HC00342[m]
     Keep:     r2395    lys_L[m] + arg_L[c]     ->    lys_L[c] + arg_L[m]
Duplicate:     r2399    lys_L[c] + arg_L[m]     ->    lys_L[m] + arg_L[c]
     Keep:     r2396    orn[c] + lys_L[m]     ->    lys_L[c] + orn[m]
Duplicate:     r2403    lys_L[c] + orn[m]     ->    orn[c] + lys_L[m]
     Keep:     r2397    lys_L[m] + his_L[c]     ->    lys_L[c] + his_L[m]
Duplicate:     r2406    lys_L[c] + his_L[m]     ->    lys_L[m] + his_L[c]
     Keep:     r2398    lys_L[m] + citr_L[c]     ->    lys_L[c] + citr_L[m]
Duplicate:     r2410    lys_L[c] + citr_L[m]     ->    lys_L[m] + citr_L[c]
     Keep:     r2400    orn[c] + arg_L[m]     ->    arg_L[c] + orn[m]
Duplicate:     r2404    arg_L[c] + orn[m]     ->    orn[c] + arg_L[m]
     Keep:     r2401    arg_L[m] + his_L[c]     ->    arg_L[c] + his_L[m]
Duplicate:     r2407    arg_L[c] + his_L[m]     ->    arg_L[m] + his_L[c]
     Keep:     r2402    arg_L[m] + citr_L[c]     ->    arg_L[c] + citr_L[m]
Duplicate:     r2411    arg_L[c] + citr_L[m]     ->    arg_L[m] + citr_L[c]
     Keep:     r2405    orn[m] + his_L[c]     ->    orn[c] + his_L[m]
Duplicate:     r2408    orn[c] + his_L[m]     ->    orn[m] + his_L[c]
     Keep:     r2409    citr_L[c] + his_L[m]     ->    citr_L[m] + his_L[c]
Duplicate:     r2413    citr_L[m] + his_L[c]     ->    citr_L[c] + his_L[m]
     Keep:     r2471    ser_L[e]     ->    ser_L[c]
```

```
Duplicate:     r2526    ser_L[e]     <=>    ser_L[c]
     Keep:     r2516    lac_L[x]     <=>    lac_L[c]
Duplicate:     LACLt    lac_L[x]      ->    lac_L[c]
     Keep:     RE3628M    dc2coa[m]     <=>    dece3coa[m]
Duplicate:     FAOXC101m    dece3coa[m]     ->    dc2coa[m]
     Keep:     BCRNe    3bcrn[c]     ->    3bcrn[e]
Duplicate:     3BCRNtr    3bcrn[e]     <=>    3bcrn[c]
     Keep:     C101CRNe    c101crn[c]     ->    c101crn[e]
Duplicate:     C101CRNtr    c101crn[e]     <=>    c101crn[c]
     Keep:     C10CRNe    c10crn[c]     ->    c10crn[e]
Duplicate:     C10CRNtr    c10crn[e]     <=>    c10crn[c]
     Keep:     C10DCe    c10dc[c]     ->    c10dc[e]
Duplicate:     C10DCtr    c10dc[e]     <=>    c10dc[c]
     Keep:     C12DCe    c12dc[c]     ->    c12dc[e]
Duplicate:     C12DCtr    c12dc[e]     <=>    c12dc[c]
     Keep:     C141OHe    3tetd7ecoacrn[c]     ->    3tetd7ecoacrn[e]
Duplicate:     3TETD7ECOACRNtr    3tetd7ecoacrn[e]     <=>    3tetd7ecoacrn[c]
     Keep:     C142OHe    3ttetddcoacrn[c]     ->    3ttetddcoacrn[e]
Duplicate:     3TTETDDCOACRNtr    3ttetddcoacrn[e]     <=>    3ttetddcoacrn[c]
     Keep:     C162OHe    3thexddcoacrn[c]     ->    3thexddcoacrn[e]
Duplicate:     3THEXDDCOACRNtr    3thexddcoacrn[e]     <=>    3thexddcoacrn[c]
     Keep:     C16DCe    c16dc[c]     ->    c16dc[e]
Duplicate:     C16DCtr    c16dc[e]     <=>    c16dc[c]
     Keep:     C3DCe    c3dc[c]     ->    c3dc[e]
Duplicate:     C3DCtr    c3dc[e]     <=>    c3dc[c]
     Keep:     C4CRNe    c4crn[c]     ->    c4crn[e]
Duplicate:     C4CRNtr    c4crn[e]     <=>    c4crn[c]
     Keep:     C4DCe    c4dc[c]     ->    c4dc[e]
Duplicate:     C4DCtr    c4dc[e]     <=>    c4dc[c]
     Keep:     C5DCe    c5dc[c]     ->    c5dc[e]
Duplicate:     C5DCtr    c5dc[e]     <=>    c5dc[c]
     Keep:     C6CRNe    c6crn[c]     ->    c6crn[e]
Duplicate:     C6CRNtr    c6crn[e]     <=>    c6crn[c]
     Keep:     C6DCe    c6dc[c]     ->    c6dc[e]
Duplicate:     C6DCtr    c6dc[e]     <=>    c6dc[c]
     Keep:     C81CRNe    c81crn[c]     ->    c81crn[e]
Duplicate:     C81CRNtr    c81crn[e]     <=>    c81crn[c]
     Keep:     C8CRNe    c8crn[c]     ->    c8crn[e]
Duplicate:     C8CRNtr    c8crn[e]     <=>    c8crn[c]
     Keep:     C8DCe    c8dc[c]     ->    c8dc[e]
Duplicate:     C8DCtr    c8dc[e]     <=>    c8dc[c]
     Keep:     DDCRNe    3ddcrn[c]     ->    3ddcrn[e]
Duplicate:     3DDCRNtr    3ddcrn[e]     <=>    3ddcrn[c]
     Keep:     DDECCRNe    ddeccrn[c]     ->    ddeccrn[e]
Duplicate:     DDECCRNtr    ddeccrn[e]     <=>    ddeccrn[c]
     Keep:     DDECE1CRNe    ddece1crn[c]     ->    ddece1crn[e]
Duplicate:     DDECE1CRNtr    ddece1crn[e]     <=>    ddece1crn[c]
     Keep:     DECCRNe    3deccrn[c]     ->    3deccrn[e]
Duplicate:     3DECCRNtr    3deccrn[e]     <=>    3deccrn[c]
     Keep:     DECDICRNe    decdicrn[c]     ->    decdicrn[e]
Duplicate:     DECDICRNtr    decdicrn[e]     <=>    decdicrn[c]
     Keep:     HEDCECRNe    3hdececrn[c]     ->    3hdececrn[e]
Duplicate:     3HDECECRNtr    3hdececrn[e]     <=>    3hdececrn[c]
     Keep:     HEXDCRNe    3hexdcrn[c]     ->    3hexdcrn[e]
Duplicate:     3HEXDCRNtr    3hexdcrn[e]     <=>    3hexdcrn[c]
     Keep:     HIVCRNe    3ivcrn[c]     ->    3ivcrn[e]
Duplicate:     3IVCRNtr    3ivcrn[e]     <=>    3ivcrn[c]
     Keep:     HOCTDEC2CRNe    3octdec2crn[c]     ->    3octdec2crn[e]
Duplicate:     3OCTDEC2CRNtr    3octdec2crn[e]     <=>    3octdec2crn[c]
     Keep:     HOCTDECCRNe    3octdeccrn[c]     ->    3octdeccrn[e]
Duplicate:     3OCTDECCRNtr    3octdeccrn[e]     <=>    3octdeccrn[c]
     Keep:     HTDCRNe    3tdcrn[c]     ->    3tdcrn[e]
Duplicate:     3TDCRNtr    3tdcrn[e]     <=>    3tdcrn[c]
     Keep:     IVCRNe    ivcrn[c]     ->    ivcrn[e]
```

```
Duplicate:     IVCRNtr    ivcrn[e]     <=>    ivcrn[c]
     Keep:     OCTDECE1CRNe    3octdece1crn[c]     ->    3octdece1crn[e]
Duplicate:     3OCTDECE1CRNtr    3octdece1crn[e]     <=>    3octdece1crn[c]
     Keep:     TDCRNe    ttdcrn[c]     ->    ttdcrn[e]
Duplicate:     TTDCRNtr    ttdcrn[e]     <=>    ttdcrn[c]
     Keep:     TETDEC2CRNe    tetdec2crn[c]     ->    tetdec2crn[e]
Duplicate:     TETDEC2CRntr    tetdec2crn[e]     <=>    tetdec2crn[c]
     Keep:     TETDECE1CRNe    tetdece1crn[c]     ->    tetdece1crn[e]
Duplicate:     TETDECE1CRNtr    tetdece1crn[e]     <=>    tetdece1crn[c]
     Keep:     TIGCRNe    c51crn[c]     ->    c51crn[e]
Duplicate:     C51CRNtr    c51crn[e]     <=>    c51crn[c]
     Keep:     CARPEPT1tc    carn[e]     ->    carn[c]
Duplicate:     CARNtr    carn[e]     <=>    carn[c]
     Keep:     CBLTDe    adocbl[c]     ->    adocbl[e]
Duplicate:     CBLtle    adocbl[e]     ->    adocbl[c]
     Keep:     FOLTle    fol[e]     ->    fol[c]
Duplicate:     r0963    fol[e]     ->    fol[c]
     Keep:     GLYPROPEPT1tc    glypro[e]     ->    glypro[c]
Duplicate:     GLYPROt    glypro[c]     <=>    glypro[e]
     Keep:     LEULEUPEPT1tc    leuleu[e]     ->    leuleu[c]
Duplicate:     LEULEUt    leuleu[c]     <=>    leuleu[e]
     Keep:     PNTORDe    pnto_R[c]     ->    pnto_R[e]
Duplicate:     PNTOte    pnto_R[e]     <=>    pnto_R[c]
     Keep:     PROGLYPEPT1tc    progly[e]     ->    progly[c]
Duplicate:     PROGLyt    progly[c]     <=>    progly[e]
     Keep:     SBTle    sbt_D[e]     ->    sbt_D[c]
Duplicate:     SBT_Dtde    sbt_D[c]     <=>    sbt_D[e]
     Keep:     TAUPAT1c    taur[e]     ->    taur[c]
Duplicate:     TAURCHAe    taur[c]     ->    taur[e]
     Keep:     GLYCTDle    glyc[e]     <=>    glyc[c]
Duplicate:     GLYCt    glyc[c]     <=>    glyc[e]
     Keep:     KHte    k[e]     <=>    k[c]
Duplicate:     r1492    k[c]     ->    k[e]
     Keep:     PHEMEe    pheme[c]     ->    pheme[e]
Duplicate:     PHEMEt    pheme[e]     ->    pheme[c]
     Keep:     SPRMTDe    sprm[e]     <=>    sprm[c]
Duplicate:     SPRMt2r    sprm[e]     <=>    sprm[c]
     Keep:     BALABETAtc2    cala[e]     <=>    cala[c]
Duplicate:     CALAtr    cala[e]     <=>    cala[c]
     Keep:     CRTNtr    crtn[e]     <=>    crtn[c]
Duplicate:     HMR_9619    crtn[e]     ->    crtn[c]
     Keep:     ALAPAT4te    ala_L[e]     <=>    ala_L[c]
Duplicate:     ALAt2r    ala_L[e]     <=>    ala_L[c]
     Keep:     PROPAT4te    pro_L[e]     <=>    pro_L[c]
Duplicate:     PROt2r    pro_L[e]     <=>    pro_L[c]
     Keep:     5AOPt    5aop[c]     <=>    5aop[e]
Duplicate:     5AOPt2    5aop[e]     ->    5aop[c]
     Keep:     ABT_Dt    abt_D[e]     <=>    abt_D[c]
Duplicate:     ABT_Dt2    abt_D[e]     <=>    abt_D[c]
     Keep:     ELAIDCRNtd    elaidcrn[c]     <=>    elaidcrn[e]
Duplicate:     ELAIDCRNtr    elaidcrn[e]     <=>    elaidcrn[c]
     Keep:     HC02149td    pcrn[c]     <=>    pcrn[e]
Duplicate:     PCRNtr    pcrn[e]     <=>    pcrn[c]
     Keep:     LNLCCRNtd    lnlccrn[c]     <=>    lnlccrn[e]
Duplicate:     LNLCCRNtr    lnlccrn[e]     <=>    lnlccrn[c]
     Keep:     PCSsec    pcs[c]     ->    pcs[e]
Duplicate:     PCSup    pcs[e]     ->    pcs[c]
     Keep:     3HCINNMup    3hcinnm[e]     ->    3hcinnm[c]
Duplicate:     3HCINNMsec    3hcinnm[c]     ->    3hcinnm[e]
     Keep:     3HPPAup    3hppa[e]     ->    3hppa[c]
Duplicate:     3HPPAsec    3hppa[c]     ->    3hppa[e]
     Keep:     PACALDtm    pacald[c]     <=>    pacald[m]
Duplicate:     HMR_4684    pacald[c]     <=>    pacald[m]
     Keep:     ACNAMt2    acnam[e]     ->    acnam[c]
```

```
Duplicate:     ACNAMtr    acnam[c]     ->    acnam[e]
     Keep:     ETHAt    etha[e]     <=>    etha[c]
Duplicate:     ETHAtr    etha[c]     ->    etha[e]
     Keep:     THMtrbc    thm[e]     <=>    thm[c]
Duplicate:     THMt3    thm[e]     <=>    thm[c]
     Keep:     BUTt2r    but[e]     <=>    but[c]
Duplicate:     HMR_0155    but[e]     <=>    but[c]
     Keep:     DIGALSGALSIDESECt    digalsgalside_hs[c]     ->    digalsgalside_hs[e]
Duplicate:     DIGALSGALSIDEt1e    digalsgalside_hs[e]     ->    digalsgalside_hs[c]
     Keep:     PAIL_hs_SECt    pail_hs[c]     ->    pail_hs[e]
Duplicate:     PAIL_hs_t1e    pail_hs[e]     ->    pail_hs[c]
     Keep:     PAILPALM_HSSECt    pailpalm_hs[c]     ->    pailpalm_hs[e]
Duplicate:     PAILPALM_HSt1e    pailpalm_hs[e]     ->    pailpalm_hs[c]
     Keep:     PAILR_HSSECt    pailar_hs[c]     ->    pailar_hs[e]
Duplicate:     PAILR_HSt1e    pailar_hs[e]     ->    pailar_hs[c]
     Keep:     PAILSTE_HSSECt    pailste_hs[c]     ->    pailste_hs[e]
Duplicate:     PAILSTE_HSt1e    pailste_hs[e]     ->    pailste_hs[c]
     Keep:     SPHMYLN180241_hs_SECt    sphmyln180241_hs[c]     ->    sphmyln180241_hs[e]
Duplicate:     SPHMYLN180241_hs_t1    sphmyln180241_hs[e]     ->    sphmyln180241_hs[c]
     Keep:     SPHMYLN18114_hs_SECt    sphmyln18114_hs[c]     ->    sphmyln18114_hs[e]
Duplicate:     SPHMYLN18114_hs_t1    sphmyln18114_hs[e]     ->    sphmyln18114_hs[c]
     Keep:     SPHMYLN18115_hs_SECt    sphmyln18115_hs[c]     ->    sphmyln18115_hs[e]
Duplicate:     SPHMYLN18115_hs_t1    sphmyln18115_hs[e]     ->    sphmyln18115_hs[c]
     Keep:     SPHMYLN18116_hs_SECt    sphmyln18116_hs[c]     ->    sphmyln18116_hs[e]
Duplicate:     SPHMYLN18116_hs_t1    sphmyln18116_hs[e]     ->    sphmyln18116_hs[c]
     Keep:     SPHMYLN181161_hs_SECt    sphmyln181161_hs[c]     ->    sphmyln181161_hs[e]
Duplicate:     SPHMYLN181161_hs_t1    sphmyln181161_hs[e]     ->    sphmyln181161_hs[c]
     Keep:     SPHMYLN18117_hs_SECt    sphmyln18117_hs[c]     ->    sphmyln18117_hs[e]
Duplicate:     SPHMYLN18117_hs_t1    sphmyln18117_hs[e]     ->    sphmyln18117_hs[c]
     Keep:     SPHMYLN18118_hs_SECt    sphmyln18118_hs[c]     ->    sphmyln18118_hs[e]
Duplicate:     SPHMYLN18118_hs_t1    sphmyln18118_hs[e]     ->    sphmyln18118_hs[c]
     Keep:     SPHMYLN181181_hs_SECt    sphmyln181181_hs[c]     ->    sphmyln181181_hs[e]
Duplicate:     SPHMYLN181181_hs_t1    sphmyln181181_hs[e]     ->    sphmyln181181_hs[c]
     Keep:     SPHMYLN18120_hs_SECt    sphmyln18120_hs[c]     ->    sphmyln18120_hs[e]
Duplicate:     SPHMYLN18120_hs_t1    sphmyln18120_hs[e]     ->    sphmyln18120_hs[c]
     Keep:     SPHMYLN181201_hs_SECt    sphmyln181201_hs[c]     ->    sphmyln181201_hs[e]
Duplicate:     SPHMYLN181201_hs_t1    sphmyln181201_hs[e]     ->    sphmyln181201_hs[c]
     Keep:     SPHMYLN18121_hs_SECt    sphmyln18121_hs[c]     ->    sphmyln18121_hs[e]
Duplicate:     SPHMYLN18121_hs_t1    sphmyln18121_hs[e]     ->    sphmyln18121_hs[c]
     Keep:     SPHMYLN18122_hs_SECt    sphmyln18122_hs[c]     ->    sphmyln18122_hs[e]
Duplicate:     SPHMYLN18122_hs_t1    sphmyln18122_hs[e]     ->    sphmyln18122_hs[c]
     Keep:     SPHMYLN181221_hs_SECt    sphmyln181221_hs[c]     ->    sphmyln181221_hs[e]
Duplicate:     SPHMYLN181221_hs_t1    sphmyln181221_hs[e]     ->    sphmyln181221_hs[c]
     Keep:     SPHMYLN18123_hs_SECt    sphmyln18123_hs[c]     ->    sphmyln18123_hs[e]
Duplicate:     SPHMYLN18123_hs_t1    sphmyln18123_hs[e]     ->    sphmyln18123_hs[c]
     Keep:     SPHMYLN1824_hs_SECt    sphmyln1824_hs[c]     ->    sphmyln1824_hs[e]
Duplicate:     SPHMYLN1824_hs_t1    sphmyln1824_hs[e]     ->    sphmyln1824_hs[c]
     Keep:     SPHMYLN1825_hs_SECt    sphmyln1825_hs[c]     ->    sphmyln1825_hs[e]
Duplicate:     SPHMYLN1825_hs_t1    sphmyln1825_hs[e]     ->    sphmyln1825_hs[c]
     Keep:     3AIBt1    3aib[e]     <=>    3aib[c]
Duplicate:     HMR_8090    3aib[c]     ->    3aib[e]
     Keep:     2HXIC_Lt1e    2hxic_L[e]     ->    2hxic_L[c]
Duplicate:     2HXIC_Lt2e    2hxic_L[c]     ->    2hxic_L[e]
     Keep:     MMAt2e    mma[c]     <=>    mma[e]
Duplicate:     MMAte    mma[e]     <=>    mma[c]
     Keep:     CE4890te2    CE4890[c]     <=>    CE4890[e]
Duplicate:     CE4890te    CE4890[c]     <=>    CE4890[e]
     Keep:     MLTHFte    mlthf[e]     ->    mlthf[c]
Duplicate:     MLTHFte3    mlthf[e]     ->    mlthf[c]
     Keep:     TYMte2    tym[c]     <=>    tym[e]
Duplicate:     TYMte    tym[c]     <=>    tym[e]
     Keep:     1A25DHVITD3te    1a25dhvitd3[e]     ->    1a25dhvitd3[c]
Duplicate:     1A25DHVITD3t2e    1a25dhvitd3[c]     ->    1a25dhvitd3[e]
     Keep:     ORN_Dtx    orn_D[x]     <=>    orn_D[c]
```

```
Duplicate:     HMR_9179    orn_D[c]     <=>    orn_D[x]
     Keep:     ORN_Dte    orn_D[c]     <=>    orn_D[e]
Duplicate:     HMR_9180    orn_D[c]     <=>    orn_D[e]
     Keep:     HC00005te    HC00005[c]     ->    HC00005[e]
Duplicate:     HC00005t1e    HC00005[e]     ->    HC00005[c]
     Keep:     HC00006te    HC00006[c]     ->    HC00006[e]
Duplicate:     HC00006t1e    HC00006[e]     ->    HC00006[c]
     Keep:     HC00007te    HC00007[c]     ->    HC00007[e]
Duplicate:     HC00007t1e    HC00007[e]     ->    HC00007[c]
     Keep:     HC00008te    HC00008[c]     ->    HC00008[e]
Duplicate:     HC00008t1e    HC00008[e]     ->    HC00008[c]
     Keep:     HC00009te    HC00009[c]     ->    HC00009[e]
Duplicate:     HC00009t1e    HC00009[e]     ->    HC00009[c]
     Keep:     NO2te    no2[e]     <=>    no2[c]
Duplicate:     HMR_6991    no2[c]     <=>    no2[e]
     Keep:     HMR_0025    M01268[n]     ->    M01268[c]
Duplicate:     HMR_0030    M01268[c]     ->    M01268[n]
     Keep:     HMR_9581    M02035[c]     <=>    M02035[e]
Duplicate:     HMR_9582    M02035[e]     ->    M02035[c]
     Keep:     HMR_9583    M02467[c]     <=>    M02467[e]
Duplicate:     HMR_9584    M02467[e]     ->    M02467[c]
     Keep:     HMR_0031    0.0024 ak2gchol_hs[c] + 0.0008 dak2gpe_hs[c] + 0.0016 pail_hs[c] + 0.19 dag_hs[
Duplicate:     HMR_0032    M02392[c]     ->    0.0024 ak2gchol_hs[c] + 0.0008 dak2gpe_hs[c] + 0.0016 pail_
     Keep:     ALLOP2tu    allop[e]     ->    allop[c]
Duplicate:     ALLOPtepvb    allop[e]     <=>    allop[c]
     Keep:     ATVACIDMCTtu    atvacid[e]     <=>    atvacid[c]
Duplicate:     ATVACIDtdu    atvacid[e]     <=>    atvacid[c]
     Keep:     OXYPthc    oxyp[e]     <=>    oxyp[c]
Duplicate:     OXYPtepv    oxyp[c]     <=>    oxyp[e]
     Keep:     PVSHtu    pvs[e]     <=>    pvs[c]
Duplicate:     PVStep    pvs[c]     <=>    pvs[e]
```

Remove any duplicate reactions from the stoichiometric matrix, but do not remove the protons.

```
if length(removedRxnInd)>0
    irrevFlag=0;
    metFlag=0;%dont remove the protons
    model = removeRxns(model,model.rxns(removedRxnInd),irrevFlag,metFlag);
end
```

Display statistics of the removed reactions

```
if printLevel>0
    [nMet0,nRxn0]=size(modelOrig.S);
    [nMet,nRxn]=size(model.S);
    fprintf('%6s\t%6s\n','#mets','#rxns')
    fprintf('%6u\t%6u\t%s\n',nMet0,nRxn0,' totals.')
    fprintf('%6u\t%6u\t%s\n',nMet0-nMet,nRxn0-nRxn,' duplicate reactions upto protons r
    fprintf('%6u\t%6u\t%s\n',nMet,nRxn,' remaining.')
end
```

```
 #mets     #rxns
  8399     13543     totals.
     0       253     duplicate reactions upto protons removed.
  8399     13290     remaining.
```

```
%model size
[nMet,nRxn]=size(model.S);
```

Heuristically identify exchange reactions and metabolites exclusively involved in exchange reactions

An external reaction is one that is heuristically identified by a single stoichiometric coefficient in the corresponding column of S, or an (abbreviated) reaction name matching a pattern (e.g. prefix EX_) or an external subsystem assignment. Any remaining reaction is assumed to be an internal reaction. If a reaction is not external then it is denoted an internal reaction. External reactants are exclusively involved in exchange reactions, and internal reactants otherwise. The findSExRxnInd function finds the external reactions in the model which export or import mass from or to the model, e.g. Exchange reactions, Demand reactions, Sink reactions.

```
if ~isfield(model,'SIntMetBool')  ||  ~isfield(model,'SIntRxnBool')
     model = findSExRxnInd(model,[],printLevel-1);
end
```

```
Assuming biomass reaction is: biomass_reaction
```

EXPECTED RESULTS

In the returned model, model.SIntRxnBool, is a boolean of reactions heuristically though to be mass balanced, while model.SIntMetBool is a boolean of metabolites heuristically though to be involved in mass balanced reactions.

CAUTION

The aforementioned assignments of external and internal reactions and reactants is the result of a heuristic and might result in one or more errors, either due to misspecification or because the names of external reactions and external subsystems often vary between laboratories.

Find the reactions that are flux inconsistent

Ultimately we seek to identify the set of stoichiometrically consistent reactions that are also flux consistent, with no bounds on reaction rates. However, finiding the stoichiometrically consistent subset can be demanding for large models so first we identify the subset of reactions that are flux consistent and focus on them.

```
modelOrig=model;
model.lb(~model.SIntRxnBool)=-1000;
model.ub(~model.SIntRxnBool)= 1000;
if 1
    if ~isfield(model,'fluxConsistentMetBool') || ~isfield(model,'fluxConsistentRxnBool
        param.epsilon=1e-4;
        param.modeFlag=0;
        param.method='null_fastcc';
        %param.method='fastcc';
        [fluxConsistentMetBool,fluxConsistentRxnBool,...
            fluxInConsistentMetBool,fluxInConsistentRxnBool,model]...
            = findFluxConsistentSubset(model,param,printLevel);
    end
    % Remove reactions that are flux inconsistent
    if any(fluxInConsistentRxnBool)
        irrevFlag=0;
```

```
        metFlag=1;
        model = removeRxns(model,model.rxns(fluxInConsistentRxnBool),irrevFlag,metFlag)
        [nMet0,nRxn0]=size(modelOrig.S);
        [nMet,nRxn]=size(model.S);

        if printLevel>0
            fprintf('%s\n','-------')
            fprintf('%6s\t%6s\n','#mets','#rxns')
            fprintf('%6u\t%6u\t%s\n',nMet0,nRxn0,' totals.')
            fprintf('%6u\t%6u\t%s\n',nMet0-nMet,nRxn0-nRxn,' flux inconsistent reaction
            fprintf('%6u\t%6u\t%s\n',nMet,nRxn,' remaining.')
            fprintf('%s\n','-------')
            if printLevel>1
                for n=1:nRxn0
                    if fluxInConsistentRxnBool(n)
                        fprintf('%15s\t%-100s\n',modelOrig.rxns{n},modelOrig.rxnNames{n
                    end
                end
            end
        end
        %revise model size
        [nMet,nRxn]=size(model.S);

        %Recompute
        %Heuristically identify exchange reactions and metabolites exclusively involved
        %finds the reactions in the model which export/import from the model
        %boundary i.e. mass unbalanced reactions
        %e.g. Exchange reactions
        %     Demand reactions
        %     Sink reactions

        model = findSExRxnInd(model,[],0);
        if printLevel>0
            fprintf('%s\n','------end------')
        end
    end
end
```

```
 12164    Total reactions
  5970    Reversible reactions.
  6194    Irreversible reactions.
 11475    Flux consistent reactions, without flipping.
   290    Flux inconsistent irreversible reactions, without flipping.
   399    Flux inconsistent reactions, without flipping.
 11792    Flux consistent reactions.
    82    Flux inconsistent reversible reactions left to flip.
 11794    Flux consistent reactions.
    80    Flux inconsistent reversible reactions left to flip.
 11796    Flux consistent reactions.
    78    Flux inconsistent reversible reactions left to flip.
 11798    Flux consistent reactions.
    38    Flux inconsistent reversible reactions left to flip.
```

Find mass leaks or siphons within the heuristically internal part, without using the bounds given by the model

```
if 1
    modelBoundsFlag=0;
    leakParams.epsilon=1e-4;
    leakParams.method='dc';
    leakParams.theta=0.5;
    [leakMetBool,leakRxnBool,siphonMetBool,siphonRxnBool,leakY,siphonY,statp,statn] =..
        findMassLeaksAndSiphons(model,model.SIntMetBool,model.SIntRxnBool,...
        modelBoundsFlag,leakParams,printLevel);
end
```

Find the maximal set of reactions that are stoichiometrically consistent

```
if ~isfield(model,'SConsistentMetBool') || ~isfield(model,'SConsistentRxnBool')
    if strcmp(model.modelID,'HMRdatabase2_00')
        massBalanceCheck=0;
    else
        massBalanceCheck=1;
    end
    if 1
        [SConsistentMetBool,SConsistentRxnBool,SInConsistentMetBool,SInConsistentRxnBoo
            =findStoichConsistentSubset(model,massBalanceCheck,printLevel);
    else
        %print out problematic reactions to file
        resultsFileName=[resultsPath filesep model.modelID];
        [SConsistentMetBool,SConsistentRxnBool,SInConsistentMetBool,SInConsistentRxnBoo
            =findStoichConsistentSubset(model,massBalanceCheck,printLevel,resultsFileNa
    end
end

rxnBool=model.SInConsistentRxnBool & model.SIntRxnBool;
if any(rxnBool)
    if printLevel>0
        fprintf('%s\n','Stoichiometrically inconsistent heuristically non-exchange reac
    end
    for n=1:nRxn
        if rxnBool(n)
            fprintf('%20s\t%50s\t%s\n',model.rxns{n},model.rxnNames{n},model.subSystems
        end
    end
    if printLevel>0
        fprintf('%s\n','--------------')
    end
end

rxnBool=model.unknownSConsistencyRxnBool & model.SIntRxnBool;
if any(rxnBool)
    if printLevel>0
        fprintf('%s\n','Unknown consistency heuristically non-exchange reactions:')
    end
    for n=1:nRxn
        if rxnBool(n)
            fprintf('%20s\t%50s\t%s\n',model.rxns{n},model.rxnNames{n},model.subSystems
        end
```

```
    end
    if printLevel>0
        fprintf('%s\n','--------------')
    end
end
```

Sanity check of stoichiometric and flux consistency of model with open external reactions

```
    if  all(model.SIntMetBool & model.SConsistentMetBool)...
            && nnz(model.SIntRxnBool & model.SConsistentRxnBool)==nnz(model.SIntRxnBool
            && all(model.fluxConsistentMetBool)...
            && all(model.fluxConsistentRxnBool)

        [nMet,nRxn]=size(model.S);
        if printLevel>1
            fprintf('%6s\t%6s\n','#mets','#rxns')
            fprintf('%6u\t%6u\t%s\n',nMet,nRxn,' totals.')
            fprintf('%6u\t%6u\t%s\n',nnz(~model.SIntMetBool),nnz(~model.SIntRxnBool),'
        end

        checksPassed=0;
        %Check that all heuristically non-exchange reactions are also stoichiometricall

        %exchange reactions
        model.EXRxnBool=strncmp('EX_', model.rxns, 3)==1;
        %demand reactions going out of model
        model.DMRxnBool=strncmp('DM_', model.rxns, 3)==1;
        %sink reactions going into or out of model
        model.SinkRxnBool=strncmp('sink_', model.rxns, 5)==1;
        %all heuristic non-exchanges, i.e., supposedly all external reactions
        bool=~(model.EXRxnBool | model.DMRxnBool | model.SinkRxnBool);
        if nnz(bool & model.SIntRxnBool & model.SConsistentRxnBool)==nnz(model.SConsist
            checksPassed=checksPassed+1;
            if printLevel>1
                fprintf('%6u\t%6u\t%s\n',nnz(model.SIntMetBool),nnz(model.SIntRxnBool),
            end
        end

        %Check for mass leaks or siphons in the stoichiometrically consistent part
        %There should be no leaks or siphons in the stiochiometrically consistent part
        modelBoundsFlag=0;
        leakParams.epsilon=1e-4;
        leakParams.eta = getCobraSolverParams('LP', 'feasTol')*100;
        leakParams.method='dc';
        [leakMetBool,leakRxnBool,siphonMetBool,siphonRxnBool,leakY,siphonY,statp,statn]
            =findMassLeaksAndSiphons(model,model.SConsistentMetBool,model.SConsistentRx

        if nnz(leakMetBool)==0 && nnz(leakRxnBool)==0 && nnz(siphonMetBool)==0 && nnz(s
            checksPassed=checksPassed+1;
            if printLevel>1
                fprintf('%6u\t%6u\t%s\n',nnz(leakMetBool | siphonMetBool),nnz(leakRxnBo
            end
```

```
        end

        %Check that the maximal conservation vector is nonzero for each the
        %internal stoichiometric matrix
        maxCardinalityConsParams.epsilon=1e-4;%1/epsilon is the largest mass considered
        maxCardinalityConsParams.method = 'quasiConcave';%seems to work the best, but s
        maxCardinalityConsParams.theta = 0.5;
        maxCardinalityConsParams.eta=getCobraSolverParams('LP', 'feasTol')*100;
        [maxConservationMetBool,maxConservationRxnBool,solution]=maxCardinalityConserva

        if nnz(maxConservationMetBool)==size(model.S,1) && nnz(maxConservationRxnBool)=
            checksPassed=checksPassed+1;
            if printLevel>1
                fprintf('%6u\t%6u\t%s\n',nnz(maxConservationMetBool),nnz(maxConservatio
            end
        end

        %Check that each of the reactions in the model (with open external reactions) i
        modelOpen=model;
        modelOpen.lb(~model.SIntRxnBool)=-1000;
        modelOpen.ub(~model.SIntRxnBool)= 1000;
        param.epsilon=1e-4;
        param.modeFlag=0;
        param.method='null_fastcc';
        [fluxConsistentMetBool,fluxConsistentRxnBool,fluxInConsistentMetBool,fluxInCons

        if nnz(fluxConsistentMetBool)==size(model.S,1) && nnz(fluxConsistentRxnBool)==s
            checksPassed=checksPassed+1;
            if printLevel>1
                fprintf('%6u\t%6u\t%s\n',nnz(fluxConsistentMetBool),nnz(fluxConsistentR
            end
        end

        if checksPassed==4
            %save the model with open exchanges as the default generic
            %model
            model=modelOpen;
            if printLevel>0
                fprintf('%s\n','Open external reactions is stoichiometrically and flux
            end
        end
        save([resultsFileName '_consistent.mat'],'model')
    end
```

```
Reference to non-existent field 'SConsistentMetBool'.
```

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.