Analyse properties of conserved moieties

Author(s): Ronan M.T. Fleming, School of Medicine, University of Galway

Reviewer(s):

INTRODUCTION

Given a set of conserved moieties, this tutorial analyses several of its properties.

These tutorials should generally be used in the following order:

1. Initialise and set the paths to inputs and outputs

COBRA.tutorials/driver_initConservedMoietyPaths.mlx

2. Build an atom transition graph

tutorial_buildAtomTransitionMultigraph.mlx

3. Identify conserved moieties, given an atom transition graph

tutorial_identifyConservedMoieties.mlx

4. Analyse the output of #3

tutorial_analyseConservedMoieties.mlx

5. Prepare for visualisation of individual conserved moieties (beta)

tutorial_visualiseConservedMoieties.mlx

```
tutorial_initConservedMoietyPaths
```

```
projectDir =
'~/work/sbgCloud/programExperimental/projects/tracerBased'
dataDir =
'~/work/sbgCloud/programExperimental/projects/tracerBased/data'
softwareDir =
'~/work/sbgCloud/programExperimental/projects/tracerBased/software'
visDataDir =
'~/work/sbgCloud/programExperimental/projects/tracerBased/data/visualisation'
resultsDir =
'~/work/sbgCloud/programExperimental/projects/tracerBased/results/conservedMoieties/centralMetabolism_fast
```

```
if ~recompute
    load([resultsDir  modelName '_ConservedMoietiesAnalysis.mat'])
    return
end
```

Load the atomically resolved models derived from identifyConservedMoieties.m

```
load([resultsDir modelName '_arm.mat'])
```

Basic properties of atomically resolved models

```
disp(arm)
```

```
     MRH: [1×1 struct]
    dATM: [1×1 digraph]
    M2Ai: [45×1342 double]
    Ti2R: [2132×33 double]
    Ti2I: [2132×15 double]
     ATG: [1×1 graph]
     M2A: [45×1342 double]
     A2R: [1115×33 double]
    A2Ti: [1115×2132 double]
     I2A: [15×1342 double]
     A2I: [1115×15 double]
     I2C: [15×347 double]
     C2A: [347×1342 double]
     A2C: [1115×347 double]
     MTG: [1×1 graph]
     I2M: [15×582 double]
     M2I: [680×15 double]
     M2M: [45×582 double]
     M2R: [680×33 double]
       L: [15×45 double]
```

Load the model, unless it is also saved with the results.

```
if 0
    load([dataDir modelName '.mat'])
    model = iDopaNeuro;
else
    model=arm.MRH;
end
```

Identify the stoichiometrically consistent subset of the model

```
massBalanceCheck=0;
printLevel=1;
[SConsistentMetBool, SConsistentRxnBool1, SInConsistentMetBool,
SInConsistentRxnBool, unknownSConsistencyMetBool,
unknownSConsistencyRxnBool, model]...
    = findStoichConsistentSubset(model,massBalanceCheck,printLevel);
```

```
--- findStoichConsistentSubset START ----
--- Summary of stoichiometric consistency ----
    45        41     totals.
     0         8     heuristically external.
    45        33     heuristically internal:
    45        33     ... of which are stoichiometrically consistent.
     0         0     ... of which are stoichiometrically inconsistent.
     0         0     ... of which are of unknown consistency.
    45        33     Confirmed stoichiometrically consistent by leak/siphon testing.
--- findStoichConsistentSubset END ----
```

Remove non-atom mapped part of the model, but keep the external reactions

```
keepRxnBool = getCorrespondingCols(arm.MRH.S, arm.MRH.metAtomMappedBool,
true(size(arm.MRH.S,2),1), 'inclusive');
keepRxnBool = keepRxnBool & ~SConsistentRxnBool1;
removeRxnBool = ~(arm.MRH.rxnAtomMappedBool | keepRxnBool);
model = removeRxns(arm.MRH, arm.MRH.rxns(removeRxnBool));
```

Identify the stoichiometrically consistent subset of the model

```
massBalanceCheck=1;
printLevel=1;
[SConsistentMetBool, SConsistentRxnBool2, SInConsistentMetBool,
SInConsistentRxnBool, unknownSConsistencyMetBool,
unknownSConsistencyRxnBool, model]...
    = findStoichConsistentSubset(model,massBalanceCheck,printLevel);
```

```
--- findStoichConsistentSubset START ----
--- Summary of stoichiometric consistency ----
    45        41     totals.
     0         8     heuristically external.
    45        33     heuristically internal:
    45        33     ... of which are stoichiometrically consistent.
     0         0     ... of which are stoichiometrically inconsistent.
     0         0     ... of which are of unknown consistency.
---
     0         0     heuristically internal and stoichiometrically inconsistent or unknown consistency.
     0         0     ... of which are elementally imbalanced (inclusively involved metabolite).
     0         0     ... of which are elementally imbalanced (exclusively involved metabolite).
    45        33     Confirmed stoichiometrically consistent by leak/siphon testing.
--- findStoichConsistentSubset END ----
```

Table of model properties

```
rankN=getRankLUSOL(arm.MRH.S(arm.MRH.metAtomMappedBool,arm.MRH.rxnAtomMappedB
ool));
rankL=getRankLUSOL(arm.L);
rankdATM=getRankLUSOL(incidence(arm.dATM));
rankATG=getRankLUSOL(incidence(arm.ATG));
rankMTG=getRankLUSOL(incidence(arm.MTG));

TT={'Model', 'm+'  , 'Metabolites', size(arm.MRH.S,1);
       ''     , 'm'   , 'Mapped metabolites', nnz(arm.MRH.metAtomMappedBool);
   ''     , 'n+'  , 'Reactions', size(arm.MRH.S,2);
   ''     , ''    ,  'Internal reactions', nnz(SConsistentRxnBool1);
   ''     , ''    ,  'External reactions', nnz(~SConsistentRxnBool1);
   ''     , 'n'   , 'Mapped reactions', nnz(arm.MRH.rxnAtomMappedBool);
   'Mapped model'     , 'm'   , 'size(model.S,1)', rankN;
   ''     , 'n+k'   , 'size(model.S,2)', size(model.S,2);
   ''     , ''    ,  'Internal reactions', nnz(SConsistentRxnBool2);
   ''     , ''    ,  'External reactions', nnz(~SConsistentRxnBool2);
   ''     , 'r'   , 'Rank(N)', rankN;
```

```
   ''     , 'm-r' , 'Row rank deficiency(N)', nnz(arm.MRH.metAtomMappedBool)
- rankN;
   ''     , ''    ,  'Isomorphism classes', size(arm.L,1);
   ''     , ''    ,  'Independent isomorphism classes', rankL;
   'MTG'  , ''    ,  'Moieties', size(arm.I2M,2);
   ''     , ''    ,  'Moiety transitions', size(arm.M2I,1);
   ''     , ''    ,  'Rank(M)', rankMTG;
  'ATG'   , ''    ,  'Atoms', size(arm.I2A,2);
   ''     , ''    ,  'Atom transitions', size(arm.A2I,1);
   ''     , ''    ,  'Rank(A)', rankATG;
   ''     , ''    ,  'Row rank deficiency(A)', size(arm.I2A,2) - rankATG;
   ''     , ''    ,  'Components', size(arm.C2A,1);
'dATM'    , ''    ,  'Atoms', size(arm.dATM.Nodes,1);
   ''     , ''    ,  'Directed atom transition instances',
size(arm.dATM.Edges,1);
   ''     , ''    ,  'Rank(dATM)', rankdATM;
   ''     , ''    ,  'Row rank deficiency(dATM)', size(arm.dATM.Edges,1) -
rankdATM;
   ''     , ''    ,  '', NaN;
   };
disp(TT)
```

```
    {'Model'       }    {'m+'    }    {'Metabolites'                      }    {[  45]}
    {0×0 char      }    {'m'     }    {'Mapped metabolites'               }    {[  45]}
    {0×0 char      }    {'n+'    }    {'Reactions'                        }    {[  41]}
    {0×0 char      }    {0×0 char}    {'Internal reactions'               }    {[  33]}
    {0×0 char      }    {0×0 char}    {'External reactions'               }    {[   8]}
    {0×0 char      }    {'n'     }    {'Mapped reactions'                 }    {[  33]}
    {'Mapped model'}    {'m'     }    {'size(model.S,1)'                  }    {[  33]}
    {0×0 char      }    {'n+k'   }    {'size(model.S,2)'                  }    {[  41]}
    {0×0 char      }    {0×0 char}    {'Internal reactions'               }    {[  33]}
    {0×0 char      }    {0×0 char}    {'External reactions'               }    {[   8]}
    {0×0 char      }    {'r'     }    {'Rank(N)'                          }    {[  33]}
    {0×0 char      }    {'m-r'   }    {'Row rank deficiency(N)'           }    {[  12]}
    {0×0 char      }    {0×0 char}    {'Isomorphism classes'              }    {[  15]}
    {0×0 char      }    {0×0 char}    {'Independent isomorphism classes'  }    {[  10]}
    {'MTG'         }    {0×0 char}    {'Moieties'                         }    {[ 582]}
    {0×0 char      }    {0×0 char}    {'Moiety transitions'               }    {[ 680]}
    {0×0 char      }    {0×0 char}    {'Rank(M)'                          }    {[ 567]}
    {'ATG'         }    {0×0 char}    {'Atoms'                            }    {[1342]}
    {0×0 char      }    {0×0 char}    {'Atom transitions'                 }    {[1115]}
    {0×0 char      }    {0×0 char}    {'Rank(A)'                          }    {[ 995]}
    {0×0 char      }    {0×0 char}    {'Row rank deficiency(A)'           }    {[ 347]}
    {0×0 char      }    {0×0 char}    {'Components'                       }    {[ 347]}
    {'dATM'        }    {0×0 char}    {'Atoms'                            }    {[1342]}
    {0×0 char      }    {0×0 char}    {'Directed atom transition instances'}    {[2132]}
    {0×0 char      }    {0×0 char}    {'Rank(dATM)'                       }    {[ 995]}
    {0×0 char      }    {0×0 char}    {'Row rank deficiency(dATM)'        }    {[1137]}
    {0×0 char      }    {0×0 char}    {0×0 char                           }    {[ NaN]}
```

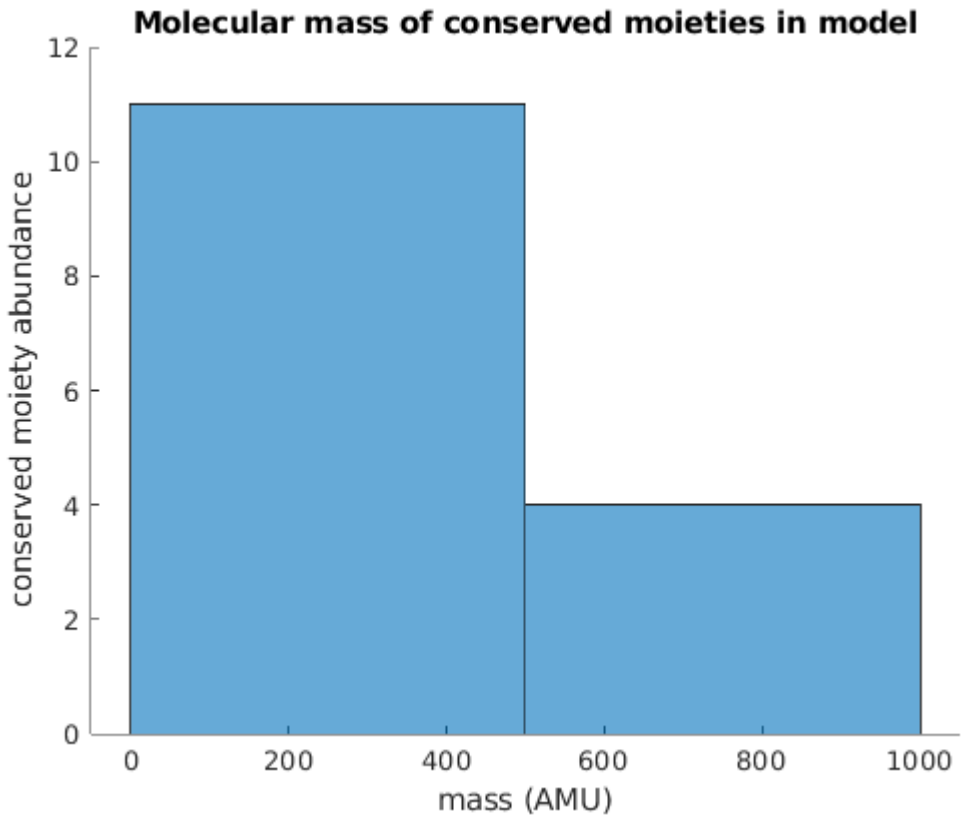
Properties of conserved moieties

```
[moietyMasses, knownmoietyMasses, unknownElements, Ematrix, elements] =
getMolecularMass(moietyFormulae);
```

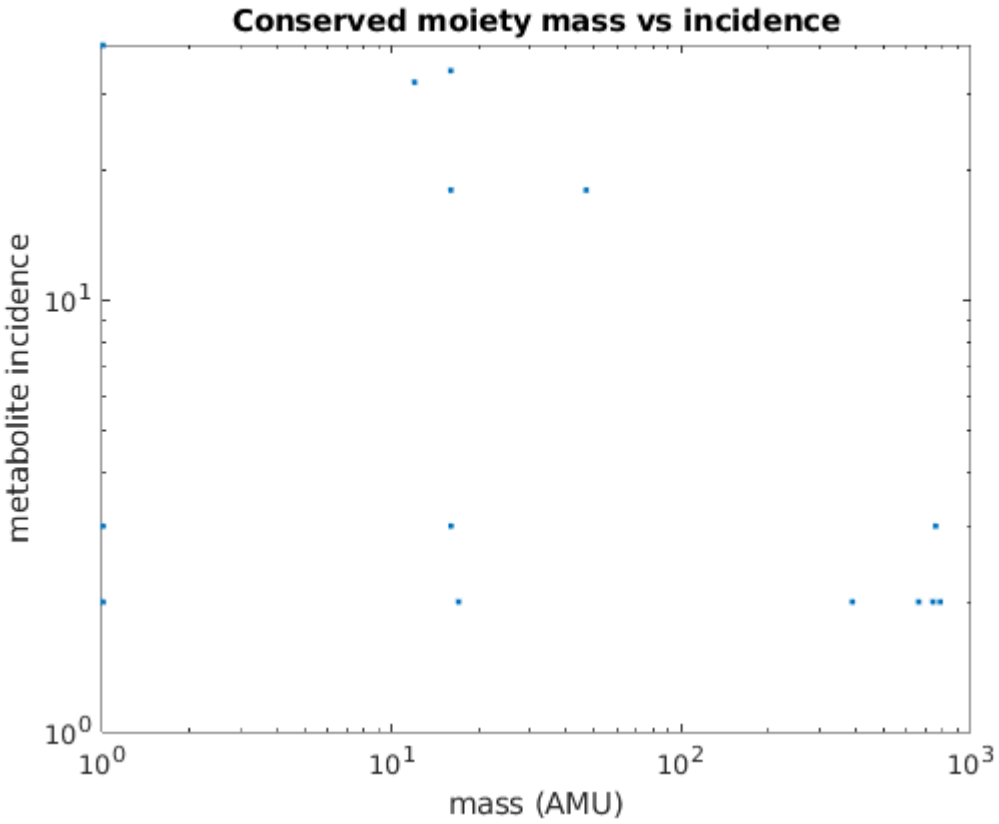
```
[metMasses, knownmetMasses, unknownElements, Ematrix, elements] =
getMolecularMass(model.metFormulas);
```

Compare the distributions of the molecular moietyMasses

```
figure
hold on
h = histogram(moietyMasses);
xlabel('mass (AMU)')
ylabel('conserved moiety abundance')
title('Molecular mass of conserved moieties in model')
```

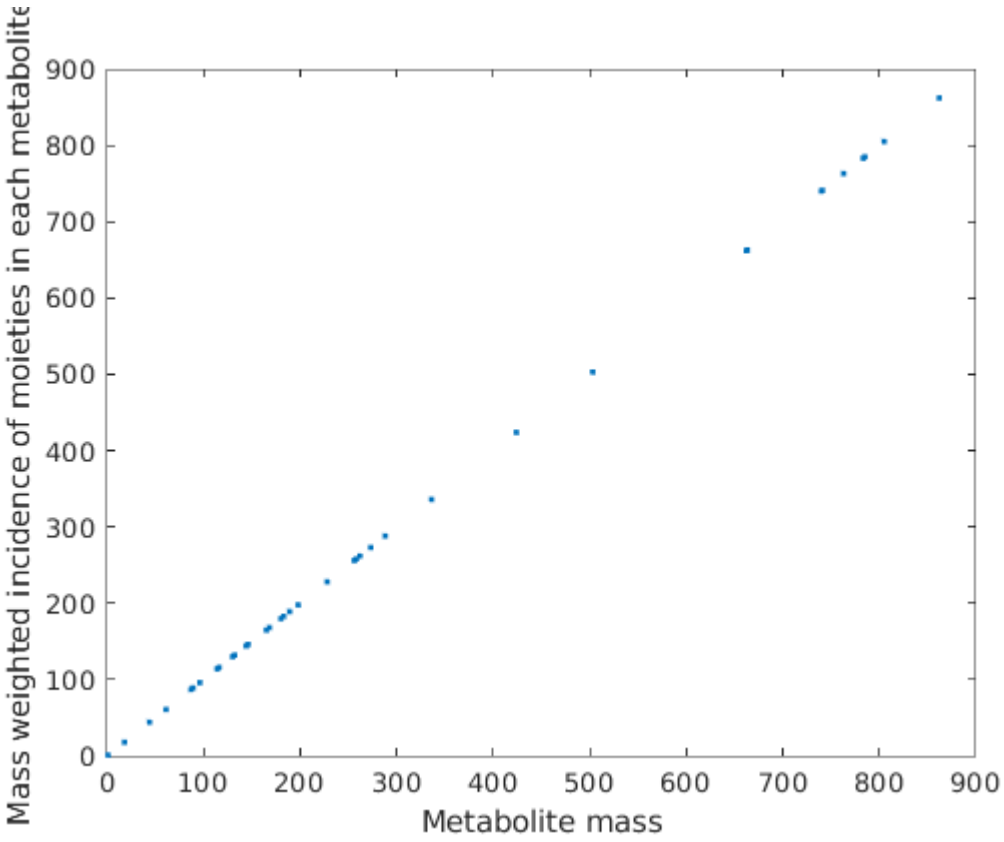



```
%h2.BinWidth = 0.25;
figure
moietyIncidence = sum(arm.L~=0,2);
loglog(moietyMasses,moietyIncidence,'.')
title('Conserved moiety mass vs incidence')
xlabel('mass (AMU)')
ylabel('metabolite incidence')
```



The metabolite mass vs mass weighted incidence of moieties in each metabolite should be a straight line trough the origin if all of the moieties that make up a metabolite are present and the formula for the metabolite is correct. Sometimes the metabolite formula is ambiguous, e.g., FULLR in the formula, so the mass will be incorrect.

```
figure
approxMetMasses = arm.L'*moietyMasses;
plot(metMasses,approxMetMasses,'.')
xlabel('Metabolite mass')
ylabel('Mass weighted incidence of moieties in each metabolite')
```




```
massDifference = abs(approxMetMasses-metMasses)./metMasses;
bool=massDifference >0.1;
n=1;
C = cell(nnz(bool),6);
for i=1:size(model.S,1)
    if bool(i)
        C(n,:)={i,
massDifference(i),model.mets{i},model.metFormulas{i},metMasses(i),approxMetMa
sses(i)};
        n=n+1;
    end
end
C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','massDifference','mets','metFormula','met
Mass','approxMetMass'};
disp(T)
```

Metabolites with mass most similar to the mass of the moiety they contain

```
[minimalMassMetabolite, minimalMassFraction, numMinimalMassMetabolites] =
representativeMolecule(arm.L,moietyFormulae,model.mets);
if ~isfield(model,'metNames')
```

```
    model.metNames = model.mets;
end
```

High molecular weight moieties that are present in many metabolites.

```
massWeightedIncidence=diag(moietyMasses)*arm.L*ones(size(arm.L,2),1);
[massWeightedIncidenceSorted, xi] = sort(massWeightedIncidence,'descend');
bool=false(size(arm.L,1),1);
bool(xi(1:min(length(xi),30)))=1;
C = cell(nnz(bool),9);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(strcmp(minimalMassMetabolite{i},model.mets));
        C(n,1:9) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),minimalMassMetabolite{i},model.metNames{ind},model.metFormulas{in
d},minimalMassFraction(i)};
        n=n+1;
    end
end

C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Minimalmassmetabolite','Name','Formula','Massfraction'};
size(T,1)
```

```
ans =
    15
```

```
disp(T)
```

index	metabolites	rxns	moietyformula	mass	Minimalmassmetabolite	
15	2	4	{'C27H31N9O15P2' }	783.1414843934	{'fad' }	{'Fla
6	3	8	{'C21H25N7O16P3S'}	756.0291330125	{'coa' }	{'Coe
10	2	8	{'C21H25N7O17P3' }	740.0519769446	{'nadp'}	{'Nic
4	2	14	{'C21H24N7O14P2' }	660.0856465362	{'nad' }	{'Nic
11	2	12	{'C10H10N5O8P2' }	390.0004603438	{'adp' }	{'Ade
13	18	50	{'OP' }	46.9686761321	{'pi' }	{'Ort
12	2	12	{'HO' }	17.0027396542	{'adp' }	{'Ade
9	3	7	{'H2N' }	16.0187240694	{'nh4' }	{'Amm
2	34	97	{'O' }	15.9949146221	{'h2o' }	{'Wat
14	18	50	{'O' }	15.9949146221	{'pi' }	{'Ort
3	32	88	{'C' }	12	{'co2' }	{'Car
1	39	121	{'H' }	1.0078250321	{'h' }	{'Pro
5	2	14	{'H' }	1.0078250321	{'nad' }	{'Nic
7	3	8	{'H' }	1.0078250321	{'coa' }	{'Coe
8	3	8	{'H' }	1.0078250321	{'coa' }	{'Coe

Moieties that are present in a near maximal number of metabolites.

```
if 1
    bool=moietyIncidence>=mean(moietyIncidence) + std(moietyIncidence);
else
    bool=moietyIncidence>=100;
end
C = cell(nnz(bool),9);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(strcmp(minimalMassMetabolite{i},model.mets));
        C(n,1:9) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),minimalMassMetabolite{i},model.metNames{ind},model.metFormulas{in
d},minimalMassFraction(i)};
        n=n+1;
    end
end

C=sortrows(C,2,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Minimalmassmetabolite','Name','Formula','Massfraction'};
size(T,1)
```

```
ans =
     3
```

```
disp(T)
```

index	metabolites	rxns	moietyformula	mass	Minimalmassmetabolite	Name
1	39	121	{'H'}	1.0078250321	{'h' }	{'Proton'
2	34	97	{'O'}	15.9949146221	{'h2o'}	{'Water'
3	32	88	{'C'}	12	{'co2'}	{'Carbon Di

Moieties that are present in a moderate number of metabolites.

```
if 1
    bool= moietyIncidence>=(mean(moietyIncidence)- std(moietyIncidence)) &
moietyIncidence<=(mean(moietyIncidence)+ std(moietyIncidence));
else
    bool= moietyIncidence>=10 & moietyIncidence<=100;
end
C = cell(nnz(bool),9);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(strcmp(minimalMassMetabolite{i},model.mets));
        C(n,1:9) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
```

```
tyMasses(i),minimalMassMetabolite{i},model.metNames{ind},model.metFormulas{in
d},minimalMassFraction(i)};
        n=n+1;
    end
end

C=sortrows(C,9,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Minimalmassmetabolite','Name','Formula','Massfraction'};
size(T,1)
```

```
ans =
    12
```

```
disp(T)
```

index	metabolites	rxns	moietyformula	mass	Minimalmassmetabolite	
10	2	8	{'C21H25N7O17P3' }	740.0519769446	{'nadp'}	{'Nic
15	2	4	{'C27H31N9O15P2' }	783.1414843934	{'fad' }	{'Fla
4	2	14	{'C21H24N7O14P2' }	660.0856465362	{'nad' }	{'Nic
6	3	8	{'C21H25N7O16P3S'}	756.0291330125	{'coa' }	{'Coe
11	2	12	{'C10H10N5O8P2' }	390.0004603438	{'adp' }	{'Ade
9	3	7	{'H2N' }	16.0187240694	{'nh4' }	{'Amm
13	18	50	{'OP' }	46.9686761321	{'pi' }	{'Ort
14	18	50	{'O' }	15.9949146221	{'pi' }	{'Ort
12	2	12	{'HO' }	17.0027396542	{'adp' }	{'Ade
5	2	14	{'H' }	1.0078250321	{'nad' }	{'Nic
7	3	8	{'H' }	1.0078250321	{'coa' }	{'Coe
8	3	8	{'H' }	1.0078250321	{'coa' }	{'Coe

Moieties that are present in a small number of metabolites.

```
bool= moietyIncidence>2 & moietyIncidence<=10;
C = cell(nnz(bool),9);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(strcmp(minimalMassMetabolite{i},model.mets));
        C(n,1:9) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),minimalMassMetabolite{i},model.metNames{ind},model.metFormulas{in
d},minimalMassFraction(i)};
        n=n+1;
    end
end

C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Minimalmassmetabolite','Name','Formula','Massfraction'};
size(T,1)
```

```
ans =
     4
```

```
disp(T)
```

index	metabolites	rxns	moietyformula	mass	Minimalmassmetabolite	
6	3	8	{'C21H25N7O16P3S'}	756.0291330125	{'coa'}	{'Coe
9	3	7	{'H2N' }	16.0187240694	{'nh4'}	{'Amm
7	3	8	{'H' }	1.0078250321	{'coa'}	{'Coe
8	3	8	{'H' }	1.0078250321	{'coa'}	{'Coe

Moieties that are present in a minimal number of metabolites.

```
bool=moietyIncidence==2;
C = cell(nnz(bool),11);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(arm.L(i,:)~=0);
        C(n,1:11) =
{i,moietyFormulae{i},moietyMasses(i),model.mets{ind(1)},model.metNames{ind(1)
},model.metFormulas{ind(1)},model.mets{ind(2)},model.metNames{ind(2)},model.m
etFormulas{ind(2)},minimalMassMetabolite{i},minimalMassFraction(i)};
        n=n+1;
    end
end

C=sortrows(C,3,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','moietyformula','mass','met1','name1','fo
rmula1','met2','name2','formula2','Minimalmassmetabolite','Massfraction'};
size(T,1)
```

```
ans =
     6
```

```
disp(T)
```

index	moietyformula	mass	met1	name1
15	{'C27H31N9O15P2'}	783.1414843934	{'fad' }	{'Flavin Adenine Dinucleotide Oxidized'
10	{'C21H25N7O17P3'}	740.0519769446	{'nadph'}	{'Nicotinamide Adenine Dinucleotide Phosp
4	{'C21H24N7O14P2'}	660.0856465362	{'nadh' }	{'Nicotinamide Adenine Dinucleotide - Red
11	{'C10H10N5O8P2' }	390.0004603438	{'atp' }	{'Adenosine Triphosphate'
12	{'HO' }	17.0027396542	{'atp' }	{'Adenosine Triphosphate'
5	{'H' }	1.0078250321	{'nadh' }	{'Nicotinamide Adenine Dinucleotide - Red

Classification of conserved moieties

```
moietyTypes = classifyMoieties(arm.L, model.S);
```

An 'Internal' moiety is one that either does not participate in any exchange reaction or is conserved by all exchange reactions

```
isInternalMoiety = strcmp('Internal',moietyTypes);
bool = isInternalMoiety;
C = cell(nnz(bool),8);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(arm.L(i,:)~=0);
        C(n,1:8) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),model.mets{ind(1)},model.metNames{ind(1)},model.metFormulas{ind(1
)}};
        n=n+1;
    end
end

C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Examplemet','Examplename','Exampleformula'};
size(T,1)
```

```
ans =
    11
```

```
disp(T)
```

index	metabolites	rxns	moietyformula	mass	Examplemet	
15	2	4	{'C27H31N9O15P2' }	783.1414843934	{'fad' }	{'Flavin Adenine
6	3	8	{'C21H25N7O16P3S'}	756.0291330125	{'coa' }	{'Coenzyme A'
10	2	8	{'C21H25N7O17P3' }	740.0519769446	{'nadph'}	{'Nicotinamide A
4	2	14	{'C21H24N7O14P2' }	660.0856465362	{'nadh' }	{'Nicotinamide A
11	2	12	{'C10H10N5O8P2' }	390.0004603438	{'atp' }	{'Adenosine Trip
13	18	50	{'OP' }	46.9686761321	{'atp' }	{'Adenosine Trip
12	2	12	{'HO' }	17.0027396542	{'atp' }	{'Adenosine Trip
14	18	50	{'O' }	15.9949146221	{'atp' }	{'Adenosine Trip
5	2	14	{'H' }	1.0078250321	{'nadh' }	{'Nicotinamide A
7	3	8	{'H' }	1.0078250321	{'coa' }	{'Coenzyme A'
8	3	8	{'H' }	1.0078250321	{'coa' }	{'Coenzyme A'

A 'Transitive' moiety is only found in primary metabolites

```
isTransititiveMoiety= strcmp('Transitive',moietyTypes);
bool = isTransititiveMoiety;
C = cell(nnz(bool),9);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(strcmp(minimalMassMetabolite{i},model.mets));
```

```
        C(n,1:9) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),minimalMassMetabolite{i},model.metNames{ind},model.metFormulas{in
d},minimalMassFraction(i)};
        n=n+1;
    end
end

C=sortrows(C,9,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Minimalmassmetabolite','Name','Formula','Massfraction'};
size(T,1)
```

```
ans =
     1
```

```
disp(T)
```

index	metabolites	rxns	moietyformula	mass	Minimalmassmetabolite	Name
9	3	7	{'H2N'}	16.0187240694	{'nh4'}	{'Ammonium'

An 'Integrative' moiety is one that is not conserved in the open network and found in both primary and secondary metabolites.

```
bool= strcmp('Integrative',moietyTypes);
C = cell(nnz(bool),8);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(arm.L(i,:)~=0);
        C(n,1:8) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),model.mets{ind(1)},model.metNames{ind(1)},model.metFormulas{ind(1
)}};
        n=n+1;
    end
end

C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Examplemet','Examplename','Exampleformula'};
size(T,1)
```

```
ans =
     3
```

```
disp(T)
```

index	metabolites	rxns	moietyformula	mass	Examplemet	Examplename	Exampl
2	34	97	{'O'}	15.9949146221	{'h2o'}	{'Water' }	{'H2
3	32	88	{'C'}	12	{'pyr'}	{'Pyruvate'}	{'C3
1	39	121	{'H'}	1.0078250321	{'h2o'}	{'Water' }	{'H2

Mitochondrially localised moieties

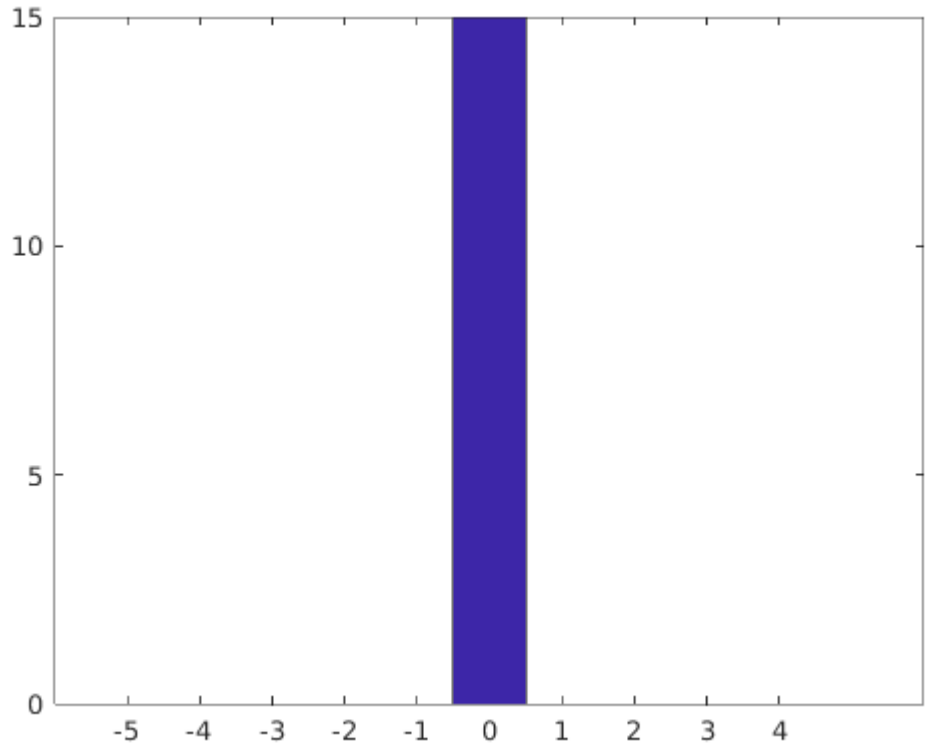
```
[compartments, uniqueCompartments] = getCompartment(model.mets);
isMitochondrial=strcmp('m',compartments);
nnz(isMitochondrial)
```

```
ans =
     0
```

```
isCompletelyMitochondrialMoiety = ~any(arm.L(:,~isMitochondrial),2);
nnz(isCompletelyMitochondrialMoiety)
```

```
ans =
    15
```

```
mitochondrialMoietyFraction = sum(arm.L(:,isMitochondrial),2)./sum(arm.L,2);
figure;
title('Fraction of moiety incidence that is mitochondrial')
hist(mitochondrialMoietyFraction)
```




```
bool= mitochondrialMoietyFraction==1;
C = cell(nnz(bool),8);
```

```
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(arm.L(i,:)~=0);
        C(n,1:8) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),model.mets{ind(1)},model.metNames{ind(1)},model.metFormulas{ind(1
)}};
        n=n+1;
    end
end

C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Examplemet','Examplename','Exampleformula'};
size(T,1)
```

```
ans =
     0
```

```
disp(T)
```

Transitive moiety, of sufficient mass, with moderate incidence

```
isTransititiveMoiety= strcmp('Transitive',moietyTypes);
isModerateIncidence = moietyIncidence>=7 & moietyIncidence<=100;
isSufficientMass = moietyMasses > 2;
isSufficientMinimalMassFraction = minimalMassFraction > 0.1;
bool = isTransititiveMoiety & isModerateIncidence & isSufficientMass &
isSufficientMinimalMassFraction;
C = cell(nnz(bool),9);
n=1;
for i=1:size(arm.L,1)
    if bool(i)
        ind = find(strcmp(minimalMassMetabolite{i},model.mets));
        C(n,1:9) =
{i,nnz(arm.L(i,:)),nnz(model.S((arm.L(i,:)~=0)',:)~=0),moietyFormulae{i},moie
tyMasses(i),minimalMassMetabolite{i},model.metNames{ind},model.metFormulas{in
d},minimalMassFraction(i)};
        n=n+1;
    end
end

C=sortrows(C,9,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','metabolites','rxns','moietyformula','mas
s','Minimalmassmetabolite','Name','Formula','Massfraction'};
```

```
size(T,1)
```

```
ans =
     0
```

```
disp(T)
```

Individual moiety subnetwork

Examine the metabolites and reactions in an individual moiety subnetwork.

```
ind = min(size(arm.L,1),32);% anth moiety
mBool=arm.L(ind,:)~=0;
nnz(mBool)
```

```
ans =
     2
```

```
rBool = getCorrespondingCols(model.S, mBool, true(size(model.S,2),1),
'inclusive');
nnz(rBool)
```

```
ans =
     2
```

Metabolites

```
bool=mBool;
C = cell(nnz(bool),5);
n=1;
for i=1:size(model.S,1)
    if bool(i)
        C(n,1:5) =
{i,model.mets{i},model.metNames{i},model.metFormulas{i},metMasses(i)};
        n=n+1;
    end
end
C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','met','name','formula','mass'};
disp(T)
```

index	met	name	formula	mass
22	{'fadh2'}	{'Flavin Adenine Dinucleotide Reduced' }	{'C27H33N9O15P2'}	785.1571344576
21	{'fad' }	{'Flavin Adenine Dinucleotide Oxidized'}	{'C27H31N9O15P2'}	783.1414843934

Reactions

```
formulas = printRxnFormula(model, model.rxns(rBool));
```

```
SUCD1m    fad + succ     <=>    fadh2 + fum
DM_fad    fadh2     <=>    2 h + fad
```

```
return
```

Another Individual moiety subnetwork

Specify the index of a particular moiety

```
ind = min(size(arm.L,1),215);% Nicotinate moiety in iDopaNeuro1.
mBool=arm.L(ind,:)~=0;
nnz(mBool)
rBool = getCorrespondingCols(model.S, mBool, true(size(model.S,2),1),
'inclusive');
nnz(rBool)
```

Metabolites

```
bool=mBool;
C = cell(nnz(bool),5);
n=1;
for i=1:size(model.S,1)
    if bool(i)
        C(n,1:5) =
{i,model.mets{i},model.metNames{i},model.metFormulas{i},metMasses(i)};
        n=n+1;
    end
end
C=sortrows(C,5,'descend');
T=cell2table(C);
T.Properties.VariableNames={'index','met','name','formula','mass'};
disp(T)
```

Reactions

```
formulas = printRxnFormula(model, model.rxns(rBool));
```

Save analysis results

```
save([resultsDir  modelName '_ConservedMoietiesAnalysis.mat'])
```

Acknowledgments

Co-funded by the European Union's Horizon Europe Framework Programme (101080997)

REFERENCES

1. Ghaderi, S., Haraldsdóttir, H. S., Ahookhosh, M., Arreckx, S., & Fleming, R. M. T. (2020). Structural conserved moiety splitting of a stoichiometric matrix. Journal of Theoretical Biology, 499, 110276. https://doi.org/10.1016/j.jtbi.2020.110276

2. Rahou, H., Haraldsdóttir, H. S., Martinelli, F., Thiele, I., & Fleming, R. M. T. (2026). Characterisation of conserved and reacting moieties in chemical reaction networks. Journal of Theoretical Biology, 112348. https://doi.org/10.1016/j.jtbi.2025.112348