Modelo de la mitocondria cardiaca

Autor: Ines Thiele, Galway University

Editado y traducido: German Preciat, Universidad de Guadalajara

Universidad de Guadalajara

Carrera: Ing. Biomédica

Materia: Ingeniería Metabólica

Contacto: german.preciat@academicos.udg.mx

En este tutorial trabajaremos con el modelo de la mitocondria cardíaca humana. Usaremos COBRA Toolbox para analizar las propiedades de la red del modelo de la mitocondria. Este modelo tiene 235 metabolitos, 225 reacciones y 323 genes. Repetiremos ciertas simulaciones simples como se describe en la información complementaria proporcionada por Orth et al [1]. Además, volveremos a ejecutar las simulaciones en Thiele et al [2].

Para comenzar primero inicializamos COBRA Toolbox y cambiamos de directorio a donde esta el modelo de la mitocondria

```
initCobraToolbox(false)
selpath = uigetdir(pwd, 'Selecciona el directorio con el modelo de la mitocondria');
cd(selpath)
```

Simulaciones a realizar:

1. Cálculo del flujo a través de DM_ATP en condiciones aeróbicas
2. Cálculo del flujo a través de DM_ATP en condiciones anaeróbicas
3. Cálculo del flujo a través de DM_ATP a partir de precursores de energía alternativa
4. Cálculo del flujo a través de la producción de cofactores
5. Cálculo de soluciones óptimas alternas
6. Cálculo de la robustez de la red
7. Análisis del plano de fase fenotípica
8. Simulación de knockouts de genes
9. Practica

Procedimiento:

1. Cálculo del flujo a través de DM_ATP en condiciones aeróbicas

El modelo proporcionado es la mitocondria del corazón de los mamíferos, por lo tanto, el objetivo principal de este sistema es producir energía para varias funciones fisiológicas del corazón y el cuerpo humano en general. La reacción de demanda de ATP dentro del modelo (DM_ATP) es una reacción que implica la hidrólisis de ATP a ADP, Pi y protones en el citosol. Fijaremos esta reacción como nuestro objetivo. A partir de entonces, proporcionaremos glucosa y oxígeno en grandes cantidades y calcularemos el flujo a través de la demanda de ATP.

Fijar los límites inferior y superior del intercambio de glucosa a 20 mmol/gDW/hr.

El flujo de consumo de oxígeno debe de ser básicamente ilimitado.

El flujo a través de la reacción de demanda de ATP (solucion.f) debe de ser alto (1000 mmol/gDW/hr).

```
load('CardicMitochondriaIT.mat');
modelMito = model;
modelMito = changeRxnBounds(modelMito, 'EX_glc(e)', -20, 'b');
modelMito = changeRxnBounds(modelMito, 'EX_o2(e)', -1000, 'l');
modelMito = changeObjective(modelMito, 'DM_atp(c)');
FBAaerobic = optimizeCbModel(modelMito, 'max');
display('El flujo a través de la reacción de demanda de ATP es de: ')
FBAaerobic.f
```

2. Cálculo del flujo a través de DM_ATP en condiciones anaeróbicas

Para ejecutar esta simulación, restringiremos la absorción de oxígeno fijando el límite inferior de la reacción de intercambio de oxígeno.

En comparación con la condición aeróbica, la condición anaeróbica debe de reducir el flujo a través de la demanda de ATP, lo que significa la necesidad de oxígeno para ejecutar la fosforilación oxidativa.

```
modelanaerobic = model;
modelanaerobic = changeRxnBounds(modelanaerobic, 'EX_glc(e)', -20, 'b');
modelanaerobic = changeRxnBounds(modelanaerobic, 'EX_o2(e)', 0, 'l');
modelanaerobic = changeObjective(modelanaerobic, 'DM_atp(c)');
FBAanaerobic = optimizeCbModel(modelanaerobic, 'max')
display('El flujo a través de la reacción de demanda de ATP es de: ')
FBAanaerobic.f
```

3. Cálculo del flujo a través de DM_ATP a partir de precursores de energía alternativa

Para calcular el flujo a través de la demanda de ATP a partir de precursores de energía alternativa, primero cerramos las fuentes de metabolitos importantes. Estos son representados como reacciones de intercambio (EX_).

EX_co2 <=> (Consumo o produccion de dioxido de carbono)

Metabolitos importantes:

```
exchanges = {'EX_12dgr_m(e)';
'EX_acac(e)';
'EX_arachd(e)';
'EX_co2(e)';
'EX_coa(e)';
'EX_crvnc(e)';
'EX_cys-L(e)';
'EX_fe2(e)';
'EX_glc(e)';
'EX_glu-L(e)';
```

```
'EX_gly(e)';
'EX_glyc(e)';
'EX_glyc3p(e)';
'EX_hb(e)';
'EX_hdca(e)';
'EX_lac-L(e)';
'EX_ocdca(e)';
'EX_ocdcea(e)';
'EX_ocdcya(e)';
'EX_pi(e)';
'EX_ps_m(e)';
'EX_urea(e)'};

modelalter = model;
modelalter = changeRxnBounds(modelalter, exchanges, 0, 'l'); % cerramos el consumo
```

Ahora proporcionaremos un precursor a la vez y calcularemos el flujo a través de la demanda de ATP.

Abrir cada fuente una por una para ver el fenotipo.

```
modelocdca = modelalter;
modelocdca = changeRxnBounds(modelocdca, 'EX_ocdcea(e)', -20, 'l');
modelocdca = changeObjective(modelocdca, 'DM_atp(c)');
FBAocdca = optimizeCbModel(modelocdca, 'max')
display('El flujo a través de la reacción de demanda de ATP es de: ')
FBAocdca.f

modelacac = modelalter;
modelacac = changeRxnBounds(modelacac, 'EX_acac(e)', -20, 'l');
modelacac = changeObjective(modelacac, 'DM_atp(c)');
FBAacac = optimizeCbModel(modelacac, 'max')
display('El flujo a través de la reacción de demanda de ATP es de: ')
FBAacac.f
```

De manera similar, agregue hidroxibutirato 'EX_hb(e)', lactato 'EX_lac-L(e)', araquidonato 'EX_arachd(e)', palmitato 'EX_hdca(e)' y glucosa 'EX_glc(e)' y calcule el flujo a través Reacción de demanda de ATP.

Tenga en cuenta en qué casos el valor de flujo es el más alto y el más pequeño. Ejecutar dicho análisis sugiere la fuente de energía favorable para la red mitocondrial del corazón.

```

```

4. Cálculo del flujo a través de la producción de cofactores

La generación de energía celular depende de la producción de cofactores como NADH, NADPH, etc. Aquí, calcularemos el flujo a través de las reacciones de drenaje de NADH y NADPH. Tales reacciones no existen en el modelo, por lo que agregaremos estas reacciones primero (nombre: 'NADH_drain', formula: 'nadh(c) -> nad(c) + h(c)'). Despues, cerrar la reacción de demanda de ATP, de modo que la energía se gaste en cualquier otro lugar.

```
modelcofac = model;
modelcofac = changeRxnBounds(modelcofac, 'EX_glc(e)', -1, 'b');
modelcofac = changeRxnBounds(modelcofac, 'DM_atp(c)', 0, 'b');
modelcofac = addReaction(modelcofac, 'NADH_drain', 'nadh(c) -> nad(c) + h(c)');
modelcofac = changeObjective(modelcofac, 'NADH_drain');
FBAcofac = optimizeCbModel(modelcofac, 'max')
display('El flujo a través de la reacción de demanda de ATP es de: ')
FBAcofac.f
```

En el caso anterior, estamos analizando el drenaje de NADH en condiciones de captación de glucosa. Repita lo anterior para el drenaje de NADPH (nombre: `'NADPH_drain'`, formula: `'nadph(c) -> nadp(c) + h(c)'`).

```

```

5. Cálculo de soluciones óptimas alternas

En todos los ejercicios especificados anteriormente, realizamos la optimización utilizando el análisis de balance de flujo (FBA). FBA genera una de las soluciones óptimas bajo la restricción especificada para el objetivo definido. Sin embargo, la red posee muchas otras soluciones alternativas, y podemos calcular esto de la siguiente manera:

```
modelopt = model;
modelopt = changeRxnBounds(modelopt, 'EX_glc(e)', 0, 'l');
modelopt = changeRxnBounds(modelopt, 'EX_lac-L(e)', -20, 'l');
modelopt = changeObjective(modelopt, 'DM_atp(c)');
FBAlac = optimizeCbModel(modelopt, 'max');

modelopt = changeRxnBounds(modelopt, 'DM_atp(c)', FBAlac.f, 'b');
modelopt = changeObjective(modelopt, 'ME2m');
FBAsolnMin = optimizeCbModel(modelopt, 'min')
FBAsolnMax = optimizeCbModel(modelopt, 'max')
```

En el ejemplo anterior, analizamos la variabilidad de la reacción `ME2m`, es decir, la reacción de la enzima málica bajo la absorción de lactato y el flujo óptimo a través de la demanda de ATP. Tenga en cuenta que los valores `FBAsolnMin.f` y `FBAsolnMax.f` son diferentes, lo que significa que las soluciones variables de esta reacción.

Alternativamente, ejecutar `fluxVariability` en `modelopt` generará los rangos de flujo mínimo y máximo de todas las reacciones en la red.

```
[FVAmin, FVAmax] = fluxVariability(modelopt, 0);
```

Al fijar `optPercentage` en cero, no especificamos ningún objetivo para la simulación realizada.

6. Cálculo de la robustez de la red

En este ejercicio, analizaremos qué tan robusta es la red frente a la reacción de demanda de ATP bajo tasas variables de consumo de glucosa cuando el consumo de oxígeno es fijo. Además, la robustez frente a la variación de la tasa de consumo de oxígeno cuando se fija el consumo de glucosa.

Glucosa

```
modelrobust = modelalter;
modelrobust = changeRxnBounds(modelrobust, 'EX_o2(e)', -17, 'b');
AtpRates = zeros(21, 1);
for i = 0:20
    modelrobust = changeRxnBounds(modelrobust, 'EX_glc(e)', -i, 'b');
    modelrobust = changeObjective(modelrobust, 'DM_atp(c)');
    FBArobust = optimizeCbModel(modelrobust, 'max');
    AtpRates(i + 1) = FBArobust.f;
end
plot(1:21, AtpRates)
xlabel('glucose uptake rate in mmol/gDW/hr')
ylabel('flux through ATP demand in mmol/gDW/hr')
```

Oxigeno

```
modelrobustoxy = modelalter;
modelrobustoxy = changeRxnBounds(modelrobustoxy,'EX_glc(e)', -20, 'b');
AtpRatesoxy = zeros(21, 1);
for i = 0:20
    modelrobustoxy = changeRxnBounds(modelrobustoxy, 'EX_o2(e)', -i, 'b');
    modelrobustoxy = changeObjective(modelrobustoxy, 'DM_atp(c)');
    FBArobustoxy = optimizeCbModel(modelrobustoxy, 'max');
    AtpRatesoxy(i + 1) = FBArobustoxy.f;
end
plot (1:21, AtpRatesoxy)
xlabel('oxygen uptake rate in mmol/gDW/hr')
ylabel('flux through ATP demand in mmol/gDW/hr')
```

7. Análisis del plano de fase fenotípica

Mientras que el análisis de robustez analiza la robustez de la red variando solo un parámetro a la vez. El análisis del plano de fase fenotípica (PhPP) analiza la robustez de la red variando dos parámetros a la vez. Variaremos las tasas de consumo de glucosa y oxígeno dentro de dos bucles for y generaremos un gráfico 3D.

```
modelphpp = modelalter;
ATPphppRates = zeros(21);
for i = 0:10
    for j = 0:20
        modelphpp = changeRxnBounds(modelphpp, 'EX_glc(e)', -i, 'b');
        modelphpp = changeRxnBounds(modelphpp, 'EX_o2(e)', -j, 'b');
        modelphpp = changeObjective(modelphpp, 'DM_atp(c)');
        FBAphpp = optimizeCbModel(modelphpp, 'max');
        ATPphppRates(i + 1,j + 1) = FBAphpp.f;
    end
end
```

```
surfl(ATPphppRates) % 3d plot
xlabel('glucose uptake rate in mmol/gDW/hr')
ylabel('oxygen uptake rate in mmol/gDW/hr')
zlabel('flux through ATP demand in mmol/gDW/hr')
```

Plano en 2D

```
pcolor(ATPphppRates)
xlabel('glucose uptake rate in mmol/gDW/hr')
ylabel('oxygen uptake rate in mmol/gDW/hr')
title('flux through ATP demand in mmol/gDW/hr')
colorbar
```

8. Simulación de knockouts de genes

La realización de estudios de eliminación de genes permite el análisis de los genes knockout en el objetivo del modelo. Esto se puede hacer usando el comando `singleGeneDeletion` o el comando `doubleGeneDeletion`. En `singleGeneDeletion`, todos los genes de la red se eliminan o las reacciones asociadas con una regla 'y' (con la asociación GPR) con el gen especificado se limitan a estar inactivas, y se analiza el efecto en la reacción objetiva. Por el contrario, en `doubleGeneDeletion`, las eliminaciones de genes por pares están enlasadas.

Aquí, usaremos el '`DM_heme(c)`' como objetivo y realizaremos la eliminación de genes simples y dobles.

```
modelknockout = model;
modelknockout = changeObjective(modelknockout, 'DM_heme(m)');
[grRatio, grRateKO, grRateWT, delRxns, hasEffect] = singleGeneDeletion_MitoModel(model
```

La función (`singleGeneDeletion_MitoModel.m`) es la misma `singleGeneDeletion`, excepto que se comenta la parte de la transcripción para permitir la alineación de la solución según la matriz `model.genes`.

Analice qué genes son esenciales para la reacción '`DM_heme(c)`' a partir del output de `grRatio`.

Para realizar la deleción de doble gen:

```
[grRatioDble, grRateKODble, grRateWTDble] = doubleGeneDeletion(modelknockout);
imagesc(grRatioDble)
xlabel('gene knockout 1')
ylabel('gene knockout 2')
```

9. Practica

Se identifico experimentalmente el fenotipo de diferentes condiciones de la mitocondria entre los cuales se encuentran:

- Estado fisiológico normal
- Condición diabética
- Condición isquémica
- Dieta baja en grasas alta en glucosa

- Dieta alta en grasas baja en glucosa

Estado fisiológico normal

```
modelnormal = model;
modelnormal = changeRxnBounds(modelnormal,'DM_atp(c)',7.5,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_acac(e)',-0.001,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_acac(e)',0,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_crvnc(e)',-0.156,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_crvnc(e)',-0.013,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_arachd(e)',-0.156,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_arachd(e)',-0.013,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_glc(e)',-0.875,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_glc(e)',-0.525,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_hb(e)',-0.001,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_hb(e)',0,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_hdca(e)',-1.250,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_hdca(e)',-0.1,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_lac-L(e)',-0.875,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_o2(e)',-39.1,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_o2(e)',-23.438,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_ocdca(e)',-0.547,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_ocdca(e)',-0.044,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_ocdcea(e)',-4.141,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_ocdcea(e)',-0.331,'u');
modelnormal = changeRxnBounds(modelnormal,'EX_ocdcya(e)',-0.547,'l');
modelnormal = changeRxnBounds(modelnormal,'EX_ocdcya(e)',-0.044,'u');
modelnormal = changeRxnBounds(modelnormal,'ATPtm',-32.6,'l');
modelnormal = changeRxnBounds(modelnormal,'ATPtm',32.6,'u');
modelnormal = changeRxnBounds(modelnormal,'CITRtm',-60,'l');
modelnormal = changeRxnBounds(modelnormal,'CITRtm',60,'u');
modelnormal = changeRxnBounds(modelnormal,'OCOAT1m',-16.9,'l');
modelnormal = changeRxnBounds(modelnormal,'OCOAT1m',16.9,'u');
modelnormal = changeRxnBounds(modelnormal,'SUCCt2m',-13.3,'l');
modelnormal = changeRxnBounds(modelnormal,'SUCCt2m',13.3,'u');
```

Condición diabética

```
modeldiab = model;
modeldiab = changeRxnBounds(modeldiab,'EX_acac(e)',-0.121,'l');
%modeldiab = changeRxnBounds(modeldiab,'EX_acac(e)',-0.020,'u');
modeldiab = changeRxnBounds(modeldiab,'EX_hb(e)',-0.106,'l');
modeldiab = changeRxnBounds(modeldiab,'EX_hb(e)',-0.016,'u');
modeldiab = changeRxnBounds(modeldiab,'EX_glc(e)',-0.644,'l');
modeldiab = changeRxnBounds(modeldiab,'EX_glc(e)',-0.386,'u');
modeldiab = changeRxnBounds(modeldiab,'C160CPT1',0.383,'l');
modeldiab = changeRxnBounds(modeldiab,'C160CPT1',0.4,'u');
modeldiab = changeRxnBounds(modeldiab,'C180CPT1',0.168,'l');
modeldiab = changeRxnBounds(modeldiab,'C180CPT1',0.183,'u');
modeldiab = changeRxnBounds(modeldiab,'C181CPT1',1.365,'l');
```

```
modeldiab = changeRxnBounds(modeldiab,'C181CPT1',1.380,'u');
modeldiab = changeRxnBounds(modeldiab,'C182CPT1',0.135,'l');
modeldiab = changeRxnBounds(modeldiab,'C182CPT1',0.151,'u');
```

Condición isquémica

```
modelische = model;
modelische = changeRxnBounds(modelische,'EX_o2(e)',-9.766,'l');
modelische = changeRxnBounds(modelische,'EX_o2(e)',-5.859,'u');
modelische = changeRxnBounds(modelische,'EX_glc(e)',-2,'l');
modelische = changeRxnBounds(modelische,'EX_glc(e)',-0.7,'u');
modelische = changeRxnBounds(modelische,'EX_acac(e)',-1.359,'l');
%modelische = changeRxnBounds(modelische,'EX_acac(e)',-0.140,'u');
modelische = changeRxnBounds(modelische,'EX_hb(e)',-1.171,'l');
modelische = changeRxnBounds(modelische,'EX_hb(e)',-0.110,'u');
```

Dieta baja en grasas alta en glucosa

```
modellowfat = model;
modellowfat = changeRxnBounds(modellowfat,'EX_glc(e)',-1.5,'l');
modellowfat = changeRxnBounds(modellowfat,'EX_glc(e)',-0.875,'u');
modellowfat = changeRxnBounds(modellowfat,'EX_arachd(e)',-0.047,'l');
modellowfat = changeRxnBounds(modellowfat,'EX_arachd(e)',-0.013,'u');
modellowfat = changeRxnBounds(modellowfat,'EX_crvnc(e)',-0.047,'l');
modellowfat = changeRxnBounds(modellowfat,'EX_crvnc(e)',-0.013,'u');
modellowfat = changeRxnBounds(modellowfat,'EX_hdca(e)',-0.375,'l');
modellowfat = changeRxnBounds(modellowfat,'EX_hdca(e)',-0.1,'u');
modellowfat = changeRxnBounds(modellowfat,'EX_ocdca(e)',-0.164,'l');
modellowfat = changeRxnBounds(modellowfat,'EX_ocdca(e)',-0.044,'u');
modellowfat = changeRxnBounds(modellowfat,'EX_ocdcea(e)',-0.521,'l');
modellowfat = changeRxnBounds(modellowfat,'EX_ocdcea(e)',-0.331,'u');
modellowfat = changeRxnBounds(modellowfat,'EX_ocdcya(e)',-0.164,'l');
modellowfat = changeRxnBounds(modellowfat,'EX_ocdcya(e)',-0.044,'u');
```

Dieta alta en grasas baja en glucosa

```
modelhighfat = model;
modelhighfat = changeRxnBounds(modelhighfat,'EX_glc(e)',-0.263,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_glc(e)',-0.158,'u');
modelhighfat = changeRxnBounds(modelhighfat,'EX_arachd(e)',-0.156,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_arachd(e)',-0.154,'u');
modelhighfat = changeRxnBounds(modelhighfat,'EX_crvnc(e)',-0.036,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_crvnc(e)',-0.031,'u');
modelhighfat = changeRxnBounds(modelhighfat,'EX_hdca(e)',-0.467,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_hdca(e)',-0.460,'u');
modelhighfat = changeRxnBounds(modelhighfat,'EX_ocdca(e)',-0.215,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_ocdca(e)',-0.210,'u');
modelhighfat = changeRxnBounds(modelhighfat,'EX_ocdcea(e)',-1.179,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_ocdcea(e)',-1.173,'u');
modelhighfat = changeRxnBounds(modelhighfat,'EX_ocdcya(e)',-0.159,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_ocdcya(e)',-0.153,'u');
```

```
modelhighfat = changeRxnBounds(modelhighfat,'EX_acac(e)',-0.324,'l');
%modelhighfat = changeRxnBounds(modelhighfat,'EX_acac(e)',-0.287,'u');
modelhighfat = changeRxnBounds(modelhighfat,'EX_hb(e)',-0.109,'l');
modelhighfat = changeRxnBounds(modelhighfat,'EX_hb(e)',-0.076,'u');
```

Utilizar las herramientas usadas anteriormente para concluir algo de estos modelos

```

```

Bibliografia

1. Orth et al, Nat Biotechnol. 2010 Mar;28(3):245-8
2. Thiele et al, J Biol Chem. 2005 Mar 25;280(12):11683-95