Gene Knockouts
Learning Objectives

• Explain the capabilities and limitations of OptKnock.
• Explain the capabilities and limitations of the Genetic Design Local Search (GDLS) tool.
• Explain the capabilities and limitations of OptGene.
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
  ✓ Knockout calculation & simulation
  ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

• OptGene
Gene/Reaction Knockouts

• Metabolic engineering has been successful in using the recombinant DNA technology to selectively alter cell metabolism (new strain design) and improve a targeted cellular function (bioproduct production).

• The use of metabolic genome scale metabolic reconstructions represents a major opportunity for the field of metabolic engineering to use whole-cell networks and systems-level analyses to determine optimal metabolic engineering strategies.

• Constraint-based techniques can be used for metabolic engineering where FBA-based algorithms, such as OptKnock, GDLS, or OptGene, predict the gene knockouts that can generate a desired phenotype to produce specific metabolites in an organism.

• Using this approach, the desired phenotype will show an increase in biomass yield coupled to an increase in the production rate of a desired by-product (metabolite). In other words, the cell will be able to grow faster only by producing more of the desired metabolite. The resulting knockout strain will have significant metabolite production at its maximal growth rate.

• These knockout strains would theoretically be stable strains that can produce specific metabolites.
Simple Case:

Selecting one of the Alternate Optimal Solutions

\[
\begin{align*}
\text{EX}_{\text{succ}}(e) &= -20 \\
\text{EX}_{\text{o2}}(e) &= -40
\end{align*}
\]
Alternate Optimal Solutions

findingOptimalSolutionsSuccVisualize.m

Solution #1

Desired

Solution #2

Eliminate

Remove

Solution #3

Eliminate

Remove
SIMULATING GENE KNOCKOUTS

- Just as growth in different environments can be simulated with FBA, gene knockouts can also be simulated by changing reaction bounds.

- To simulate the knockout of any gene, its associated reaction or reactions can simply be constrained to not carry flux. By setting both the upper and lower bounds ('b') of a reaction to 0 mmol gDW\(^{-1}\) hr\(^{-1}\), a reaction is essentially knocked out, and is restricted from carrying flux.

- The COBRA Toolbox contains a function called `deleteModelGenes` that uses the GPRs to constrain the correct reactions. Then FBA may be used to predict the model phenotype with gene knockouts.
Knocking Out All but the Desired Flux Vector

```matlab
% succinateOptimalSolutionsKnockouts.m
clear; clc;

model = readCbModel('ecoli_textbook');
model = changeRxnBounds(model,'EX_glc(e)',0,'l');
model = changeRxnBounds(model,'EX_o2(e)',-40,'l');
model = changeRxnBounds(model,'EX_succ(e)',-20,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Delete reactions to select desired alternate flux vector
model = changeRxnBounds(model,'ME1',0,'b'); % Knockout the ME1 reaction
model = changeRxnBounds(model,'PYK',0,'b'); % Knockout the PYK reaction

% List optimal solutions
solverOK = changeCobraSolver('glpk','MILP'); % Won't work with Gurobi
[solutions] = enumerateOptimalSolutions(model);
```

Knocking out a reaction from the undesired optimal flux vectors to force the cell into the desired flux vector.
Removing Optimal Flux Vectors

No Knockouts

ME1 Knocked Out (zero flux)

ME1 & PYK Knocked Out (zero flux)
Knocking Out All but the Desired Flux Vector

```matlab
% succinateOptimalSolutionsKnockouts.m
clear; clc;

model = readCbModel('ecoli_textbook');
model = changeRxnBounds(model,'EX_glc(e)',0,'l');
model = changeRxnBounds(model,'EX_o2(e)',-40,'l');
model = changeRxnBounds(model,'EX_succ(e)',-20,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Delete reactions to select desired alternate flux vector
model = changeRxnBounds(model,'ME1',0,'b');  % Knockout the ME1 reaction
model = changeRxnBounds(model,'PYK',0,'b');   % Knockout the PYK reaction

% List optimal solutions
solverOK = changeCobraSolver('glpk','MILP');  % Won't work with Gurobi
[solutions] = enumerateOptimalSolutions(model);
```
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
  ✓ Knockout calculation & simulation
  ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

• OptGene
**OptKnock**

**A Reaction Deletion Strategy**

- The OptKnock framework suggests a reaction deletion strategy that leads to the overproduction of specific chemical compounds.
- This is accomplished by ensuring that the production of the desired chemical becomes an obligatory byproduct of growth by "shaping" the connectivity of the metabolic network.
- OptKnock identifies and subsequently removes metabolic reactions that are capable of uncoupling cellular growth from chemical production.
- To reduce the computation time of OptKnock the number of candidate reactions for knockout should be minimized.
- Requires Gurobi or CPLEX solvers! Built-in Matlab solvers will not work.

Maximize: Bioengineering Objective
(through reaction knockouts)

Subject to: Maximize: cellular objective
(over fluxes)
Subject to: Fixed substrate uptake
Network Stoichiometry
Blocked reactions identified by the outer problem

Bilevel optimization structure of OptKnock

---

Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
    ✓ Production Envelopes
    ✓ Knockout calculation & simulation
    ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

• OptGene
Identifying Unwanted Knockout Reactions

- When building the list of reactions, for deletion, "selectRxns," exclude exchange and transport reactions, and biomass and ATP maintenance requirements.
  - ATP maintenance requirements: ATPM
  - Exchange Reactions: EX_?
  - Biomass Objective Function: 'Biomass_Ecoli_core_N(w/GAM)_Nmet2'

- Removing unwanted reactions

```matlab
[transRxns,nonTransRxns] = findTransRxns(model,true); % Identify non-transport/exchange reactions
[tmp,ATPMnumber] = ismember('ATPM',nonTransRxns); % Identify ATPM reaction number
[tmp,BioMassnumber] = ismember('BiomassEcoli',nonTransRxns); % Identify biomass reaction number
nonTransRxnsLength = length(nonTransRxns); % Find number of non-transport reactions

selectedRxns = {nonTransRxns{[1:ATPMnumber-1, ATPMnumber+1:BioMassnumber-1, ... BioMassnumber+1:nonTransRxnsLength]}}, % Reactions to be used by OptKnock
```
## Transport & Non-transport Reactions

\[
[\text{transRxns},\text{nonTransRxns}] = \text{findTransRxns(model, true)}
\]

<table>
<thead>
<tr>
<th>Transport (transRxns) Reactions</th>
<th>Non-transport (nonTransRxns) Reactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>'ACALDt'</td>
<td>'ACALD'</td>
</tr>
<tr>
<td>'ACt2r'</td>
<td>'GLUSy'</td>
</tr>
<tr>
<td>'AKGt2r'</td>
<td>'PTAr'</td>
</tr>
<tr>
<td>'ATPS4r'</td>
<td>'ACKr'</td>
</tr>
<tr>
<td>'CO2t'</td>
<td>'GND'</td>
</tr>
<tr>
<td>'CYTBD'</td>
<td>'ADK1'</td>
</tr>
<tr>
<td>'D_LACT2'</td>
<td>'AKGDH'</td>
</tr>
<tr>
<td>'ETOHt2r'</td>
<td>'ALCD2x'</td>
</tr>
<tr>
<td>'EX_ac(e)'</td>
<td>'ATPM'</td>
</tr>
<tr>
<td>'EX_acald(e)'</td>
<td>'Biomass'</td>
</tr>
<tr>
<td>'EX_akg(e)'</td>
<td>'CS'</td>
</tr>
<tr>
<td>'EX_co2(e)'</td>
<td>'ENO'</td>
</tr>
<tr>
<td>'EX_etoht(e)'</td>
<td>'FBA'</td>
</tr>
<tr>
<td>'EX_fru(e)'</td>
<td>'FBP'</td>
</tr>
<tr>
<td>'EX_fum(e)'</td>
<td>'FRD7'</td>
</tr>
<tr>
<td>'EX_glc(e)'</td>
<td>'FUM'</td>
</tr>
<tr>
<td>'EX_gln_l(e)'</td>
<td>'G6PDH2r'</td>
</tr>
<tr>
<td>'EX_glu_l(e)'</td>
<td>'GAPD'</td>
</tr>
<tr>
<td>'EX_h2o(e)'</td>
<td>'GLN5'</td>
</tr>
<tr>
<td>'EX_h(e)'</td>
<td>'GLUDy'</td>
</tr>
<tr>
<td>'EX_h2(e)'</td>
<td>'GLUN'</td>
</tr>
<tr>
<td>'EX_i(e)'</td>
<td>'GLUD'</td>
</tr>
<tr>
<td>'EX_lac_d(e)'</td>
<td>'GLUy'</td>
</tr>
<tr>
<td>'EX_nh4(e)'</td>
<td>'GLUN'</td>
</tr>
<tr>
<td>'EX_o2(e)'</td>
<td>'GND'</td>
</tr>
<tr>
<td>'EX_pi(e)'</td>
<td>'PYK'</td>
</tr>
<tr>
<td>'EX_pyr(e)'</td>
<td>'RPE'</td>
</tr>
<tr>
<td>'EX_succ(e)'</td>
<td>'RPI'</td>
</tr>
<tr>
<td>'EX_thd'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'EX_water(e)'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'FORt2'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'FORti'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'FRUpts2'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'GLCpts'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'GLNabc'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'GLU2r'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'H2Ot'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'HLT2r'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'MALT2_2'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'MALt2_2'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'NADH16'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'NH4t'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'O2t'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'PTAr'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'PYK'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'RPI'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'SUCDi'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>'TALa'</td>
<td>'TALa'</td>
</tr>
</tbody>
</table>

IdentifyingUnwantedKnockoutReactions.m
Potential Knockout Reactions

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>'ACALD'</td>
<td>26</td>
<td>'EX_fru(e)'</td>
<td>51</td>
<td>'GLNS'</td>
</tr>
<tr>
<td>2</td>
<td>'ACALDt'</td>
<td>27</td>
<td>'EX_fum(e)'</td>
<td>52</td>
<td>'GLNabc'</td>
</tr>
<tr>
<td>3</td>
<td>'ACKr'</td>
<td>28</td>
<td>'EX_glc(e)'</td>
<td>53</td>
<td>'GLUDy'</td>
</tr>
<tr>
<td>4</td>
<td>'ACONTa'</td>
<td>29</td>
<td>'EX_gln_L(e)'</td>
<td>54</td>
<td>'GLUN'</td>
</tr>
<tr>
<td>5</td>
<td>'ACONTb'</td>
<td>30</td>
<td>'EX_glu_L(e)'</td>
<td>55</td>
<td>'GLUSy'</td>
</tr>
<tr>
<td>6</td>
<td>'ACt2r'</td>
<td>31</td>
<td>'EX_h2o(e)'</td>
<td>56</td>
<td>'GLUH2r'</td>
</tr>
<tr>
<td>7</td>
<td>'ADK1'</td>
<td>32</td>
<td>'EX_h(e)'</td>
<td>57</td>
<td>'GND'</td>
</tr>
<tr>
<td>8</td>
<td>'AKGDH'</td>
<td>33</td>
<td>'EX_lac_D(e)'</td>
<td>58</td>
<td>'H2Ot'</td>
</tr>
<tr>
<td>9</td>
<td>'AKG2r'</td>
<td>34</td>
<td>'EX_mal_L(e)'</td>
<td>59</td>
<td>'ICDHy'</td>
</tr>
<tr>
<td>10</td>
<td>'ALCD2x'</td>
<td>35</td>
<td>'EX_nh4(e)'</td>
<td>60</td>
<td>'ICL'</td>
</tr>
<tr>
<td>11</td>
<td>'ATPM'</td>
<td>36</td>
<td>'EX_o2(e)'</td>
<td>61</td>
<td>'LDH_D'</td>
</tr>
<tr>
<td>12</td>
<td>'ATPS4r'</td>
<td>37</td>
<td>'EX_p(i(e)'</td>
<td>62</td>
<td>'MALS'</td>
</tr>
<tr>
<td>13</td>
<td>'Biomass'</td>
<td>38</td>
<td>'EX_pyr(e)'</td>
<td>63</td>
<td>'MALt2_2'</td>
</tr>
<tr>
<td>14</td>
<td>'CO2t'</td>
<td>39</td>
<td>'EX_succ(e)'</td>
<td>64</td>
<td>'MDH'</td>
</tr>
<tr>
<td>15</td>
<td>'CS'</td>
<td>40</td>
<td>'FBA'</td>
<td>65</td>
<td>'ME1'</td>
</tr>
<tr>
<td>16</td>
<td>'CYTBD'</td>
<td>41</td>
<td>'FBP'</td>
<td>66</td>
<td>'ME2'</td>
</tr>
<tr>
<td>17</td>
<td>'D_LAc2'</td>
<td>42</td>
<td>'FORt2'</td>
<td>67</td>
<td>'NADH16'</td>
</tr>
<tr>
<td>18</td>
<td>'ENO'</td>
<td>43</td>
<td>'FORti'</td>
<td>68</td>
<td>'NADTRHD'</td>
</tr>
<tr>
<td>19</td>
<td>'ETOHt2r'</td>
<td>44</td>
<td>'FRD7'</td>
<td>69</td>
<td>'NHi4'</td>
</tr>
<tr>
<td>20</td>
<td>'EX_ac(e)'</td>
<td>45</td>
<td>'FRUpts2'</td>
<td>70</td>
<td>'O2t'</td>
</tr>
<tr>
<td>21</td>
<td>'EX_acald(e)'</td>
<td>46</td>
<td>'FUM'</td>
<td>71</td>
<td>'PDH'</td>
</tr>
<tr>
<td>22</td>
<td>'EX_akg(e)'</td>
<td>47</td>
<td>'FUMt2_2'</td>
<td>72</td>
<td>'PFK'</td>
</tr>
<tr>
<td>23</td>
<td>'EX_co2(e)'</td>
<td>48</td>
<td>'G6PDH2r'</td>
<td>73</td>
<td>'PFL'</td>
</tr>
<tr>
<td>24</td>
<td>'EX_etoH(e)'</td>
<td>49</td>
<td>'GAPD'</td>
<td>74</td>
<td>'PGI'</td>
</tr>
<tr>
<td>25</td>
<td>'EX_for(e)'</td>
<td>50</td>
<td>'GLCpts'</td>
<td>75</td>
<td>'PGK'</td>
</tr>
</tbody>
</table>

Black – Potential knockout reactions; Red - Unwanted knockout Reactions;
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
    ✓ Knockout calculation & simulation
    ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

• OptGene
Production Envelopes
(EthanolProduction_OptKnock_Gurobi_Revised.m)

Production envelope - A graph that shows both the maximum and minimum bioproduct production with respect to the growth rate

```matlab
lineColor = 'b';
targetRxn = 'EX_etoh(e)';
biomassRxn = 'Biomass_Ecoli_core_N(w/GAM)_Nmet2';
geneDelFlag = false; % Genes(true) or reactions(false)
nPts = 50;
deletions = {'ACKr','GLUDy','ME2','PGL','PYK'};
[biomassValues,targetValues] = productionEnvelope(model,deletions,lineColor,targetRxn,biomassRxn,geneDelFlag,nPts);
xlabel('Biomass (mmol/g DW-hr)');
ylabel('EX-etoh(e)(mmol/g DW-hr)');
```

**Knockouts**

'FUM', 'G6PDH2r', 'GLUDy', 'PTAr', 'SUCDi'

**Desired production state**

(0.1772, 17.31)

**Coupled growth & metabolite production**

(0.1772, 17.31)
Multiproduction Envelopes of All Growth-coupled Secreted Metabolites

```matlab
% Reactions to be deleted
deletions = {'ACKr','GLUDy','ME2','PGL','PYK'};

% Biomass function
biomassRxn = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2'};

% Show only growth coupled metabolites
[biomassValues,targetValues] = multiProductionEnvelope(model,deletions,biomassRxn,false,20,false);
```
Multiproduction Envelopes of All the Secreted Metabolites

% Reactions to be deleted
deletions = {'ACKr','GLUDy','ME2','PGL','PYK'};

% Biomass function
biomassRxn = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2'};

% Show all secreted metabolites
[biomassValues,targetValues] = multiProductionEnvelope(model,deletions,biomassRxn,false,20,true);
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
  ✓ Knockout calculation & simulation
    ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

• OptGene
What Do We Know About Ethanol Production?

```matlab
% AnaerobicEthanolRA.m

clear; clc;

% Input the E.coli core model
model=readCbModel('ecoli_textbook');

% Set uptake rates
model = changeRxnBounds(model,'EX_glc(e)',-10,'b');
model = changeRxnBounds(model,'EX_o2(e)',-0,'b');
model = changeRxnBounds(model,'EX_etoh(e)',0,'l');

% Set optimization objective to Biomass_Ecoli_core_N(w/GAM)_Nmet2
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Using robustnessAnalysis, plot the objective function as a function
% of the ethanol secretion rate
[controlFlux, objFlux] = robustnessAnalysis(model,'EX_etoh(e)',100);
```

This curve represents the maximum possible growth rate.

Set as minimum growth rate.

Maximum ethanol production with no growth.

Robustness Analysis
Alternate Optimal Flux Vectors for Ethanol Production

```matlab
% findingOptimalSolutionsEthanol.m

clear;

model = readCbModel('ecoli_textbook');

model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',0,'l');

model = changeObjective(model,'EX_etoh(e)');

solverOK = changeCobraSolver('glpk');

% List optimal solutions
[solutions] = enumerateOptimalSolutions(model);
```
FVA Chart for Ethanol Production

FVA Ethanol Production.xlsx
Enhanced OptKnock Example: Ethanol Production
(EthanolProduction_OptKnock_Gurobi_Revised.m)

% EthanolProduction_OptKnock_Gurobi_Revised.m
clear; clc;
% Input the E.coli core model
model=readCobraModel('ecoli_textbook');
solverOK = changeCobraSolver('gurobi5','all');
% Set carbon source and oxygen uptake rates
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',-0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Select reactions to be explored by the optKnock algorithm
[transRxns,nonTransRxns] = findTransRxns(model,true);
[tmp,ATPMnumber] = ismember('ATPM',nonTransRxns); % Identify ATPM reaction number
[tmp,BioMassnumber] = ismember('Biomass_Ecoli_core_N(w/GAM)_Nmet2',nonTransRxns); % Identify biomass reaction number
nonTransRxnsLength = length(nonTransRxns); % Find number of non-transport reactions
selectedRxns = {nonTransRxns{[1:ATPMnumber-1, ATPMnumber+1:BioMassnumber-1, BioMassnumber+1:nonTransRxnsLength]});

% Optknock analysis for Ethanol secretion
options.targetRxn = 'EX_etoh(e)';
options.vMax = 1000;
options.numDel = 5;
options.numDelSense = 'L';
constrOpt.rxnList = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2','ATPM'};
constrOpt.values = [0.14, 8.39];
constrOpt.sense = 'GE';

optKnockSol = OptKnock(model, selectedRxns, options, constrOpt);
deletions = optKnockSol.rxnList'

deletions = 'FUM','G6PDH2r','GLUDy','PTAr','SUCDi'
growthRate = 0.1772
minProd = 17.31; maxProd = 17.31
Ethanol Production: Exchange Fluxes
5 Knockouts (EthanolProduction_Mutant.m)

clear; clc;
model=readCbModel('ecoli_textbook');

model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Knockout reactions
model = changeRxnBounds(model,'FUM',0,'b');
model = changeRxnBounds(model,'G6PDH2r',0,'b');
model = changeRxnBounds(model,'GLUDy',0,'b');
model = changeRxnBounds(model,'PTAr',0,'b');
model = changeRxnBounds(model,'SUCDi',0,'b');

FBAsolution = optimizeCbModel(model,'max');
printFluxVector(model,FBAsolution.x, true, true)

map=readCbMap('ecoli_Textbook_ExportMap');
options.lb = -10;
options.ub = 10;
options.zeroFluxWidth = 0.1;
options.rxnDirMultiplier = 10;
drawFlux(map, model, FBAsolution.x, options);
robustnessAnalysis(model,'EX_etoh(e)',100);

<table>
<thead>
<tr>
<th>Exchange Fluxes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass</td>
</tr>
<tr>
<td>EX_co2(e)</td>
</tr>
<tr>
<td>EX_etoh(e)</td>
</tr>
<tr>
<td>EX_for(e)</td>
</tr>
<tr>
<td>EX_glc(e)</td>
</tr>
<tr>
<td>EX_h2o(e)</td>
</tr>
<tr>
<td>EX_h(e)</td>
</tr>
<tr>
<td>EX_nh4(e)</td>
</tr>
<tr>
<td>EX_pi(e)</td>
</tr>
</tbody>
</table>
Ethanol Production: Fluxes
5 Knockouts (EthanolProduction_Mutant.m)

clear; clc;
model=readCbModel('ecoli_textbook');

model = changeRxnBounds(model, 'EX_glc(e)', -10, 'l');
model = changeRxnBounds(model, 'EX_o2(e)', 0, 'l');
model = changeObjective(model, 'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Knockout reactions
model = changeRxnBounds(model, 'FUM', 0, 'b');
model = changeRxnBounds(model, 'G6PDH2r', 0, 'b');
model = changeRxnBounds(model, 'GLUDy', 0, 'b');
model = changeRxnBounds(model, 'PTAr', 0, 'b');
model = changeRxnBounds(model, 'SUCDi', 0, 'b');

model = changeRxnBounds(model, 'EX_etoh(e)', 100);

FBAsolution = optimizeCbModel(model, 'max');
printFluxVector(model, FBAsolution.x, true)

map=readCbMap('ecoli_Textbook_ExportMap');
options.lb = -10;
options.ub = 10;
options.zeroFluxWidth = 0.1;
options.rxnDirMultiplier = 10;
drawFlux(map, model, FBAsolution.x, options);
robustnessAnalysis(model, 'EX_etoh(e)', 100);
Ethanol Production: Flux Map
5 Knockouts (EthanolProduction_Mutant.m)
Ethanol Production: Robustness Analysis
5 Knockouts (EthanolProduction_Mutant.m)

clear; clc;
model=readCbModel('ecoli_textbook');

model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Knockout reactions
model = changeRxnBounds(model,'ACKr',0,'b');
model = changeRxnBounds(model,'GLUDy',0,'b');
model = changeRxnBounds(model,'ME2',0,'b');
model = changeRxnBounds(model,'PGL',0,'b');
model = changeRxnBounds(model,'PYK',0,'b');

FBAsolution = optimizeCbModel(model,'max')
printFluxVector(model,FBAsolution.x, true, true)

map=readCbMap('ecoli_Textbook_ExportMap');
options.lb = -10;
options.ub = 10;
options.zeroFluxWidth = 0.1;
options.rxnDirMultiplier = 10;
drawFlux(map, model, FBAsolution.x, options);

robustnessAnalysis(model,'EX_etoh(e)',100);
Production Envelopes for Different Ethanol Producing Knockouts

- 'PFL'
  - (0.1800, 16.59)

- 'ACKr', 'GLUDy', 'GND'
  - (0.1772, 17.31)

- 'FUM', 'G6PDH2r', 'GLUDy', 'PTAr', 'SUCCI'
  - (0.1772, 17.31)

- 'PTAr', 'G6PDH2r'
  - (0.1876, 17.15)

- 'ADK1', 'GLUDy', 'GND', 'PTAr'
  - (0.1772, 17.31)
1 Knockout

EthanolProduction_Mutant.m
EthanolProduction_multiProductionEnvelope.m

(PFL) (0.1800, 16.59)

Secreting Succinate

Block Formate Pathway
2 Knockouts

EthanolProduction_Mutant.m
EthanolProduction_multiProductionEnvelope.m

\[ \text{Ethanol Production} \]

\[ \text{Formate} \]

\( (0.1876, 17.15) \)

\[ \text{PTAr}, 'G6PDH2r' \]

\[ \text{Ethanol} \rightarrow \text{Formate} \]

\[ \text{NADPH Production} \]

\[ \text{ATP Production} \]

\[ \text{Block Acetate Pathway} \]

\[ \text{Reduce NADPH} \]

\[ \text{Secreting Formate} \]

\[ \text{Extracellular Protons} \]
3 Knockouts
EthanolProduction_Mutant.m

- 'ACKr', 'GLUDy', 'GND'
- (0.1772, 17.31)

- Ethanol
- Formate

- EthanolFormate

- Use Glutamine to Produce Glutamate
- Block Acetate Pathway
- NADPH Production
- ATP Production

- Secretory Formate
- Extracellular Protons
4 Knockouts

EthanolProduction_Mutant.m

- 'ADK1', 'GLUDy', 'GND', 'PTAr'

- NADPH Production
- Reduce ATP -> ADP
- Use Glutamine to Produce Glutamate
- Block Acetate Pathway
- Extracellular Protons
- Secreting Formate
- Formate

Ethanol

(0.1772, 17.31)
5 Knockouts

EthanolProduction_Mutant.m

'FUM', 'G6PDH2r', 'GLUDy', 'PTAr', 'SUCDi'

(0.1772, 17.31)
## FVA Classifications for Ethanol Production

<table>
<thead>
<tr>
<th>Hard-Coupled Reactions</th>
<th>Partially-Coupled Reactions</th>
<th>Not-Coupled Reactions</th>
<th>Zero-flux Reactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>'Biomass'</td>
<td>'ACONTa'</td>
<td>'EX_pyr(e)'</td>
<td>'PFL'</td>
</tr>
<tr>
<td>'ACONTb'</td>
<td>'ACALD'</td>
<td>'EX_succ(e)'</td>
<td>'PGK'</td>
</tr>
<tr>
<td>'Biomass'</td>
<td>'ACALDt'</td>
<td>'FBP'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'CS'</td>
<td>'ACKr'</td>
<td>'FORt2'</td>
<td>'PGM'</td>
</tr>
<tr>
<td>'EX_co2(e)'</td>
<td>'Act2r'</td>
<td>'FORTi'</td>
<td>'PPCK'</td>
</tr>
<tr>
<td>'EX_etoh(e)'</td>
<td>'ADK1'</td>
<td>'FRD7'</td>
<td>'PPS'</td>
</tr>
<tr>
<td>'EX_h(e)'</td>
<td>'AKGDH'</td>
<td>'FUM'</td>
<td>'PTAr'</td>
</tr>
<tr>
<td>'FBA'</td>
<td>'AKGt2r'</td>
<td>'G6PDH2r'</td>
<td>'PYRt2r'</td>
</tr>
<tr>
<td>'GLNS'</td>
<td>'ALCD2x'</td>
<td>'GLUDy'</td>
<td>'RPE'</td>
</tr>
<tr>
<td>'ICDHyr'</td>
<td>'ATPS4r'</td>
<td>'GLUN'</td>
<td>'RPI'</td>
</tr>
<tr>
<td>'NH4t'</td>
<td>'CO2t'</td>
<td>'GLUSy'</td>
<td>'SUCC12_2'</td>
</tr>
<tr>
<td>'PDH'</td>
<td>'D_LACT2'</td>
<td>'GLU2r'</td>
<td>'SUCCT3'</td>
</tr>
<tr>
<td>'PFK'</td>
<td>'ETOHt2r'</td>
<td>'GND'</td>
<td>'SUCOAS'</td>
</tr>
<tr>
<td>'PGI'</td>
<td>'EX_ac(e)'</td>
<td>'H2Ot'</td>
<td>'TALA'</td>
</tr>
<tr>
<td>'PIT2r'</td>
<td>'EX_acald(e)'</td>
<td>'ICL'</td>
<td>'THD2'</td>
</tr>
<tr>
<td>'PPC'</td>
<td>'EX_aka(g(e)'</td>
<td>'LDH_D'</td>
<td>'TKTI'</td>
</tr>
<tr>
<td>'PYK'</td>
<td>'EX_for(e)'</td>
<td>'MALS'</td>
<td>'TKT2'</td>
</tr>
<tr>
<td>'TPI'</td>
<td>'EX_glc(e)'</td>
<td>'MDH'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>'EX_glu_L(e)'</td>
<td>'ME1'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>'EX_h2o(e)'</td>
<td>'ME2'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>'EX_lac_D(e)'</td>
<td>'NADH16'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>'EX_nh4(e)'</td>
<td>'NADTRHD'</td>
<td></td>
</tr>
</tbody>
</table>

FVA_SuccinateClassification.m
Parsimonious FBA For Glucose in Anaerobic Environment

- Essential pFBA Optima
- Enzymatically Less Efficient
- Metabolically Less Efficient
- pFBA No-flux
- Blocked

<table>
<thead>
<tr>
<th>Essential</th>
<th>pFBA Optima</th>
<th>Enzymatically Less Efficient</th>
<th>Metabolically Less Efficient</th>
<th>pFBA No-flux</th>
<th>Blocked</th>
</tr>
</thead>
<tbody>
<tr>
<td>'ACONTa'</td>
<td>'ACALD'</td>
<td>'FRD7'</td>
<td>'ACALDI'</td>
<td>'ME2'</td>
<td>'CYTBD'</td>
</tr>
<tr>
<td>'ACONTb'</td>
<td>'ACKr'</td>
<td>'SUCDi'</td>
<td>'ADK1'</td>
<td>'NADHI6'</td>
<td>'EX_fru(e)'</td>
</tr>
<tr>
<td>Biomass</td>
<td>'AC12r'</td>
<td>'AKGDH'</td>
<td>'NADTRHD'</td>
<td>'EX_fum(e)'</td>
<td></td>
</tr>
<tr>
<td>'CS'</td>
<td>'ALCD2x'</td>
<td>'AKG12r'</td>
<td>'PDH'</td>
<td>'EX_gln_L(e)'</td>
<td></td>
</tr>
<tr>
<td>'ENO'</td>
<td>'ATPM'</td>
<td>'D_LAC12'</td>
<td>'PGL'</td>
<td>'EX_mal_L(e)'</td>
<td></td>
</tr>
<tr>
<td>'EX_glc(e)'</td>
<td>'ATPS4r'</td>
<td>'EX_acald(e)'</td>
<td>'PPCK'</td>
<td>'EX_o2(e)'</td>
<td></td>
</tr>
<tr>
<td>'EX_h(e)'</td>
<td>'CO2t'</td>
<td>'EX_akhir(e)'</td>
<td>'PPS'</td>
<td>'FruUpts2'</td>
<td></td>
</tr>
<tr>
<td>'EX_nh4(e)'</td>
<td>'ETOH12r'</td>
<td>'EX_glu_L(e)'</td>
<td>'PYRt2r'</td>
<td>'Fumt2_2'</td>
<td></td>
</tr>
<tr>
<td>'EX_pi(e)'</td>
<td>'EX_ac(e)'</td>
<td>'EX_lac_D(e)'</td>
<td>'Succ12_2'</td>
<td>'GLNabc'</td>
<td></td>
</tr>
<tr>
<td>'FBA'</td>
<td>'EX_co2(e)'</td>
<td>'EX_pyr(e)'</td>
<td>'Succ13'</td>
<td>'Malt1_2'</td>
<td></td>
</tr>
<tr>
<td>'GAPD'</td>
<td>'EX_ether(e)'</td>
<td>'EX_succ(e)'</td>
<td>'SuccoAS'</td>
<td>'O2t'</td>
<td></td>
</tr>
<tr>
<td>'GLCpts'</td>
<td>'EX_for(e)'</td>
<td>'FBP'</td>
<td>'FOR12'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'GLNS'</td>
<td>'EX_h2o(e)'</td>
<td>'FUM'</td>
<td>'66PDH2r'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'ICDHyr'</td>
<td>'FORTi'</td>
<td>'GPDh'</td>
<td>'GLUN'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'NH4t'</td>
<td>'GLUDy'</td>
<td>'PGI'</td>
<td>'GLUSy'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'PFK'</td>
<td>'H2Ot'</td>
<td>'PGK'</td>
<td>'GLY12r'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'PGI'</td>
<td>'PPL'</td>
<td>'PGM'</td>
<td>'GPDK'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'PGK'</td>
<td>'PTAr'</td>
<td>'PFL'</td>
<td>'IICL'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'PGM'</td>
<td>'PYK'</td>
<td>'PTAr'</td>
<td>'IICL'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'PI2r'</td>
<td>'RPE'</td>
<td>'PCc'</td>
<td>'LDH_D'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'PCc'</td>
<td>'TALA'</td>
<td>'RPI'</td>
<td>'MASt'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'RPI'</td>
<td>'TKT1'</td>
<td>'TP'</td>
<td>'MDH'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'TP'</td>
<td>'TKT2'</td>
<td>'TP'</td>
<td>'ME1'</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram: pFBA_Ecoli_Core.m (Anaerobic)
Can we Increase the Ethanol Production?

```matlab
% AnaerobicEthanolRA.m

clear; clc;

% Input the E.coli core model
model=readCbModel('ecoli_textbook');

% Set uptake rates
model = changeRxnBounds(model,'EX_glc(e)',-10,'b');
model = changeRxnBounds(model,'EX_o2(e)',-0,'b');
model = changeRxnBounds(model,'EX_etoh(e)',0,'l');

% Set optimization objective to Biomass_Ecoli_core_N(w/GAM)_Nmet2
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Using robustnessAnalysis, plot the objective function as a function of the ethanol secretion rate
[controlFlux, objFlux] = robustnessAnalysis(model,'EX_etoh(e)',100);
```

Robustness Analysis

- This curve represents the maximum possible growth rate.
- Set as minimum growth rate.
- Larger production, lower growth rate.
- Maximum ethanol production with no growth.
OptKnock Example: Maximizing Ethanol Production

(EthanolProduction_OptKnock_Gurobi.m)

```matlab
% EthanolProduction_OptKnock_Gurobi_Revised.m
clear; clc;
% Input the E.coli core model
model = readCobraModel('ecoli_textbook');
solverOK = changeCobraSolver('gurobi5','all');
% Set carbon source and oxygen uptake rates
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',-0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Select reactions to be explored by the optKnock algorithm
[transRxns,nonTransRxns] = findTransRxns(model,true);
[tmp,ATPMnumber] = ismember('ATPM',nonTransRxns); % Identify ATPM reaction number
[tmp,BioMassnumber] = ismember('Biomass_Ecoli_core_N(w/GAM)_Nmet2',nonTransRxns); % Identify biomass reaction number
nonTransRxnsLength = length(nonTransRxns); % Find number of non-transport reactions
selectedRxns = {nonTransRxns{[1:ATPMnumber-1, ATPMnumber+1:BioMassnumber-1, BioMassnumber+1:nonTransRxnsLength]}];

% Optknock analysis for Ethanol secretion
options.targetRxn = 'EX_etoh(e)';
options.vMax = 1000;
options.numDel = 5;
options.numDelSense = 'L';
constrOpt.rxnList = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2','ATPM'};
constrOpt.values = [0.05, 8.39];
constrOpt.sense = 'GE';

optKnockSol = OptKnock(model, selectedRxns, options, constrOpt);
deletions = optKnockSol.rxnList'

% Print out growth rate and minimum & maximum secretion rate
[growthRate,minProd,maxProd] = testOptKnockSol(model,'EX_etoh(e)',optKnockSol.rxnList)
deletions = 'ACKr'    'GLUDy'    'ME2'    'PGL'    'PYK'
growthRate = 0.0767
minProd = 18.5452; maxProd = 18.8342
```

Output:

- Deletions: 'ACKr', 'GLUDy', 'ME2', 'PGL', 'PYK'
- Growth Rate: 0.0767
- Minimum Production: 18.5452
- Maximum Production: 18.8342
Production Envelopes for Different Knockouts

Notice the decrease in the slope of the production curve which implies better coupling with growth.
Multiproduction Envelopes for Different Knockouts

Note that only one case includes secondary secretion.
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
  ✓ Knockout calculation & simulation
  ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS
• OptGene
analyzeOptKnock

% EthanolProduction_analyzeOptKnock.m

clear; clc;

model=readCbModel('ecoli_textbook');
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',-0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Analyze OptKnok results for growth coupling

deletions = {'ACKr','GLUDy','ME2','PGL','PYK'};
biomassRxn = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2'};
minGrowth = 0,05;
geneDelFlag = false;
target = 'EX_etoh(e)';

[type,maxGrowth,maxProd] = analyzeOptKnock(model,deletions,target,biomassRxn,geneDelFlag)

Determines whether an optKnock solution is growth coupled or not and what the maximum growth and production rates are

minGrowth =

0

type -

growth coupled non unique

maxGrowth -

0.0767

maxProd =

18.8342
% EthanolProduction_OptKnock_CPLEX.m

clear; clc;

% Input the E.coli core model
model = readCbModel('ecoli_textbook');
solverOK = changeCobraSolver('tomlab_cplex','all');
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',-0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Select reactions to be explored by the optKnock algorithm
[transRxns,nonTransRxns] = findTransRxns(model,true);
[tmp,ATPMnumber] = ismember('ATPM',nonTransRxns); % Identify ATPM reaction number
[tmp,BioMassnumber] = ismember('Biomass_Ecoli_core_N(w/GAM)_Nmet2',nonTransRxns); % Identify biomass reaction number
nonTransRxnsLength = length(nonTransRxns); % Find number of non-transport reactions
selectedRxns = {nonTransRxns{[1:ATPMnumber-1, ATPMnumber+1:BioMassnumber-1, BioMassnumber+1:nonTransRxnsLength]}};

% Optknock analysis for Ethanol secretion
options.targetRxn = 'EX_etoh(e)';
options.vMax = 1000;
options.numDel = 1;
options.numDelSense = 'L';
constrOpt.rxnList = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2','ATPM'};
constrOpt.values = [0.14, 8.39];
constrOpt.sense = 'GE';

optKnockSol = OptKnock(model, selectedRxns, options, constrOpt);
deletions = optKnockSol.rxnList';

growthRate = testOptKnockSol(model,'EX_etoh(e)',optKnockSol.rxnList)

% Print out growth rate and minimum & maximum secretion rate
[growthRate,minProd,maxProd] = testOptKnockSol(model,'EX_etoh(e)',optKnockSol.rxnList)
simpleOptKnock

% EthanolProduction_simpleOptKnock.m

clear; clc;

model = readCbModel('ecoli_textbook');
model = changeRxnBounds(model, 'EX_glc(e)', -10, 'l');
model = changeRxnBounds(model, 'EX_o2(e)', -0, 'l');
model = changeObjective(model, 'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

deletions = {'ACKr', 'GLUDy', 'ME2', 'PGL', 'PYK'};

biomassRxn = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2'};
minGrowth = 0.05;
geneDelFlag = false;
doubleDelFlag = false;

[wtRes, delRes] = simpleOptKnock(model, 'EX_etoh(e)', deletions, false, minGrowth, doubleDelFlag)

Check all one gene or reaction deletions for growth-coupled metabolite production

Maximum production for each individual knockout reaction
doubleProductionEnvelope

Plots maximum growth rate as a function of the output of two specified products.

% EthanolProduction_doubleProductionEnvelope.m

clear; clc;
model=readCbModel('ecoli_textbook');
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',-0,'l');

% Calculate doubleProductionEnvelope

deletions = {};
biomassRxn = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2'};
[x1,x2,y] = doubleProductionEnvelope(model,deletions,'EX_etoh(e)','EX_for(e)',biomassRxn,false,20);
OptKnock Review Questions

• What is OptKnock?
• Why should the number of potential knockout reactions be limited?
• What type of reactions should not be included in an OptKnock search?
• How do you knockout a reaction using the Cobra Toolbox?
• What does it mean to couple the growth and metabolite production?
• What is a production envelope? What is a multiproduction envelope?
• How can a production envelope be created for all secreted metabolite?
• Why is there a trade-off between biomass growth and bioproduct production?
• What are some of the unintended consequences from optimized bioproduct production?
• How many knockouts can be identified by OptKnock?
• What are some of the key parameters needed for OptKnock?
• How can you simulate the engineered mutant cell using the knockouts identified by OptKnock?
• What are the limitations of OptKnock?
• What are some of the OptKnock supporting functions in the Cobra Toolbox?
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
  ✓ Knockout calculation & simulation
  ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

• OptGene
Genetic Design Local Search (GDLS)

This algorithm typically runs faster than the global search performed by OptKnock, however, it is not guaranteed to identify the global optima.

\[
[gdlSSolution, biLevelMILPproblem, gdlsSolutionStructs] = GDLS(model, varargin)
\]

where: varargin are optional parameters; gdlsSolution is the knockout solution; biLevelMILPproblem is the bi-level MILP problem for the solution; and gdlsSolutionStructs contains the intermediate solutions.

Overview of the Genetic Design Local Search (GDLS)

Model Reduction

Change the model to reduce the number of reactions, metabolites, and boundary conditions to reduce the search space for optimization.

**GDLS Model Reduction**

- Remove dead-end reactions
- Replace linked reactions or equivalent variables (pairs of fluxes that are constrained to have the same value.)
- Finds the minimal bounds for the flux through each reaction.

**reduceModel Cobra Function**

- Removes from the model all of the reactions that are never used (max and min are < tol).
- Finds the minimal bounds for the flux through each reaction.
- Also returns the results for flux variability analysis (maxes, mins).
[gdlsSolution, bilevelMILPPProblem, gdlsSolutionStructs] = GDLS(model, varargin)

INPUTS
model             Cobra model structure
targetRxn        Reaction(s) to be maximized (Cell array of strings)

OPTIONAL INPUTS
varargin         parameters entered using either a structure or list of parameter, parameter value

List of optional parameters
'nbdhpsz'       Neighborhood size (default: 1)
'M'             Number of search paths (default: 1)
'maxKO'         Maximum number of knockouts (default: 50)
'koCost'        Cost for knocking out a reaction, gene set, or gene
                 A different cost can be set for each knockout. (default: 1 for each knockout)
'selectedRxns'  List of reactions/geneSets that can be knocked out
'koType'        What to knockout: reactions, gene sets, or genes {('rxns'), 'geneSets', 'genes')
'iterationLimit' Maximum number of iterations (default: 70)
'timeLimit'     Maximum run time in seconds (default: 252000)
'minGrowth'     Minimum growth rate

OUTPUTS
gdlsSolution    GDLS solution structure (similar to OptKnock sol struct)
bilevelMILPPProblem Problem structure used in computation
GDLS Example: Maximizing Ethanol Production

EthanolProduction_GDLS_Revised.m

% EthanolProduction_GDLS_Revised.m
clear; clc;

% Set operating conditions
model=readCbModel('ecoli_textbook');
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Select reactions
[transRxns,nonTransRxns] = findTransRxns(model,true);
[tmp,ATPMnumber] = ismember('ATPM',nonTransRxns); % Identify ATPM reaction number
[tmp,BioMassnumber] = ismember('Biomass_Ecoli_core_N(w/GAM)_Nmet2',nonTransRxns); % Identify biomass reaction number
nonTransRxnsLength = length(nonTransRxns); % Find number of non-transport reactions
selectedRxns = {nonTransRxns{[1:ATPMnumber-1, ATPMnumber+1:BioMassnumber-1, BioMassnumber+1:nonTransRxnsLength]});

% GDLS analysis for Ethanol secretion
[gdlsSolution, bilevelMILPproblem, gdlsSolutionStructs] = GDLS(model, 'EX_etoh(e)', 'minGrowth', 0.05, ...'
'selectedRxns', selectedRxns, 'maxKO', 5, 'nbhdsz', 2);

deletions = 'ACKr', 'G6PDH2r', 'GLUDy', 'ME2', 'PYK'
growthRate = 0.0767
maxProd = 18.8342
GDLS Example: Final Three Iterations
EthanolProduction_GDLS_Revised.m

**Iteration 2**
------------------
Biomass flux:  0.081627  
Synthetic flux:  [18.451847, 18.759400]  
Knockout cost:  4  
Knockouts:  
  ACKr  
  G6PDH2r  
  ME2  
  PYK  

**Iteration 3**
------------------
Biomass flux:  0.076704  
Synthetic flux:  [18.545214, 18.834218]  
Knockout cost:  5  
Knockouts:  
  ACKr  
  G6PDH2r  
  GLUDy  
  ME2  
  PYK  

**Iteration 4**
------------------
Biomass flux:  0.076704  
Synthetic flux:  [18.545214, 18.834218]  
Knockout cost:  5  
Knockouts:  
  ACKr  
  G6PDH2r  
  GLUDy  
  ME2  
  PYK  

GDLS Example: Final Three Iterations
EthanolProduction_GDLS_Revised.m
**GDLS Example:**

Location of Knockouts for the GDLS Mutant

Knockouts = ACKr, G6PDH2r, GLUDy, ME2, PYK

EthanolProduction_GDLS_Mutant_Revised.m
Robustness Analysis & Production Envelope

Robustness Analysis

Production multiProductionEnvelope_GDLS.m

EthanolProduction_GDLS_Mutant_Revised.m

Multiproduction Envelope

Ethanol

Succinate
## GDLS Example: Final Three Iterations

**EthanolProduction_GDLS_Revised.m**

<table>
<thead>
<tr>
<th>Iteration 2</th>
<th>Biomass flux: 0.081627</th>
<th>Synthetic flux: [18.451847, 18.759400]</th>
<th>Knockout cost: 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knockouts: ACKr, G6PDH2r, ME2, PYK</td>
<td>Biomass 0.0816265</td>
<td>EX_co2(e) 18.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_etoh(e) 18.7594</td>
<td>EX_for(e) 0.307552</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_glc(e) -10</td>
<td>EX_h2o(e) 0.535192</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_h(e) 1.94498</td>
<td>EX_nh4(e) -0.445093</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_pi(e) -0.300279</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Knockouts: ACKr, G6PDH2r, GLUDy, ME2, PYK</td>
<td>Biomass 0.0767038</td>
<td>EX_co2(e) 18.7784</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_etoh(e) 18.8342</td>
<td>EX_for(e) 0.289004</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_glc(e) -10</td>
<td>EX_h2o(e) 0.502916</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_h(e) 1.82768</td>
<td>EX_nh4(e) -0.41825</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_pi(e) -0.300279</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Knockouts: ACKr, G6PDH2r, GLUDy, ME2, PYK</td>
<td>Biomass 0.0767038</td>
<td>EX_co2(e) 18.7784</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_etoh(e) 18.8342</td>
<td>EX_for(e) 0.289004</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_glc(e) -10</td>
<td>EX_h2o(e) 0.502916</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_h(e) 1.82768</td>
<td>EX_nh4(e) -0.41825</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EX_pi(e) -0.300279</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

GDLS allows loops!
GDLS Review Questions

- What is the purpose of GDLS?
- What is the difference between OptKnock, GDLS?
- What are some of the key parameters required by GDLS?
- What does it mean to reduce a model?
- How many iterations are required to complete the GDLS algorithm?
- How many knockouts can be identified by GDLS?
- What are the limitations of GDLS?
- Will OptKnock and GDLS give the same knockouts?
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
  ✓ Knockout calculation & simulation
  ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

→ • OptGene
OptGene is an evolutionary programming-based method to determine gene knockout strategies for overproduction of a specific product. It can handle non-linear objective functions such as product flux multiplied by biomass.

```matlab
[x, population, scores, optGeneSol] = OptGene(model, targetRxn, ...
    substrateRxn, generxnList, maxKOs, [population])
```

where: targetRxn specifies the reaction to optimize; substrateRxn specifies the exchange reaction for the carbon source; generxnList is a cell array of strings that specifies which genes or reactions are allowed to be deleted; and maxKOs sets the maximum number of knockouts; x is the best scoring set as determined by the functions optGeneFitness or optGeneFitnessTilt; population is the binary matrix representing the knockout sets; and optGeneSol is the structure summarizing the results. If resuming a previous simulation, the binary matrix (population) can be specified.

OptGene requires the CPLEX solver and will not work with the Gurobi or gplk solver.

OptGene Algorithm

- OptGene begins with a predefined number of individuals, forming a population. Each column corresponds to a reaction.
- The fitness score of an individual is calculated using the desired objective function value. The best individuals are selected for crossover.
- Selected individuals are then crossed to produce a new offspring.
- Individuals propagating to the new population are mutated (in our formulation, a gene is deleted) with a given probability.
- Mutation and crossover give rise to a new population, which can then again be subjected to a new round of evaluation, crossover and mutations.
- This cycle is repeated until an individual with a satisfactory phenotype is found.

OptGene Code Changes

• optGene.m (Fitness function) - Optional
  • From
    0102 %FitnessFunction = @(x) optGeneFitness(x,model,targetRxn, generxnList, geneok);
    0103 FitnessFunction = @(x) optGeneFitnessTilt(x,model,targetRxn, generxnList, geneok);
  • To
    0102 FitnessFunction = @(x) optGeneFitness(x,model,targetRxn, generxnList, geneok);
    0103 %FitnessFunction = @(x) optGeneFitnessTilt(x,model,targetRxn, generxnList, geneok);

• From (multiple locations within optGene)
  removeindex = ind2(randint(1,1,length(ind2))+1);
  • To
    removeindex = ind2(randi(1,1,length(ind2))+1);

• optGeneSol.m - Required
  • From
    0021 writeDirect = 'C:/';
  • To
    0021 writeDirect = 'C:\Users\hinton\Documents\MATLAB';
optGene Example

```matlab
% Ethanol_OptGene_core_revised.m

clear; clc;

% Set conditions
model = readCbModel('ecoli_textbook');
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',0,'l');
model = changeObjective(model,'Biomass_Ecoli_core_N(w/GAM)_Nmet2');

% Select reactions
[transRxns,nonTransRxns] = findTransRxns(model,true);
[tmp,ATPMnumber] = ismember('ATPM',nonTransRxns); % Identify ATPM reaction number
[tmp,BioMassnumber] = ismember('Biomass_Ecoli_core_N(w/GAM)_Nmet2',nonTransRxns); % Identify biomass reaction number
nonTransRxnsLength = length(nonTransRxns); % Find number of non-transport reactions
generxnList = {nonTransRxns([1:ATPMnumber-1, ATPMnumber+1:BioMassnumber-1, BioMassnumber+1:nonTransRxnsLength])};

% Run optGene
targetRxn = {'EX_etoh(e)'};
substrateRxn = {'EX_glc(e)'};
[x, population, scores, optGeneSol] = optGene(model, targetRxn, substrateRxn,generxnList,1);

% Graph production envelope
lineColor = 'b';
targetRxn = 'EX_etoh(e)';
biomassRxn = 'Biomass_Ecoli_core_N(w/GAM)_Nmet2';
geneDelFlag = false;
nPts = 50;
[biomassValues,targetValues] = productionEnvelope(model,optGeneSol.rxnList,lineColor,targetRxn,biomassRxn,geneDelFlag,nPts);
xlabel('Biomass (mmol/g DW-hr)')
ylabel('EX-etoh(e)(mmol/g DW-hr)')
```
optGene Display Window
Ethanol_OptGene_core_revised.m

![Graph showing biomass (mmol/g DW-hr) vs. EX-etch (mmol DW/hr).]

- Fitness of Each Individual:
- Score Histogram:
- Mutation Histogram:
- Stopping Criteria:

Best: -16.5863 Mean: -15.2668
OptGene Review Questions

• What is the purpose of OptGene?

• What is the difference between OptKnock, GDLS, and OptGene?

• Explain how to use OptGene?

• How many knockouts can be identified by OptGene?

• What are the limitations of OptGene?

• What solvers are required by OptGene?

• Are there changes that need to be made to the OptGene function before it can be used?

• How does OptGene’s speed compare to OptKnock and GDLS?
Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying potential knockout reactions
  ✓ Production Envelopes
  ✓ Knockout calculation & simulation
  ✓ OptKnock supporting functions in the Cobra Toolbox

• GDLS

• OptGene