Gene/Reaction Knockouts
Learning Objectives

• Explain the purpose of a gene/reaction knockout

• Explain growth-coupled bioproduction.

• Explain the purpose of a production envelope plot.

• 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 recombinant DNA technology to selectively alter cell metabolism (new strain design) and improve 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 analysis 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 gene/reaction knockouts that can generate a desired phenotype to produce specific metabolites by 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 (mutant) could have significant metabolite production at a desired growth rate.

• These knockout strains would theoretically be stable strains that can produce specific metabolites.
Simulating Gene/Reaction Knockouts

• Just as growth in different environments can be simulated with FBA, gene/reaction 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 reactions associated with a given gene. Then FBA may be used to predict the model phenotype with gene knockouts.
Creating a Mutant Strain: Anaerobic Ethanol Production

EthanolProduction_WildType.m

EthanolProduction_Mutants.m
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 required 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.

Bilevel optimization structure of OptKnock

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

Number of knockouts ≤ limit

Lesson Outline

• Overview

• OptKnock
  ✓ OptKnock Philosophy
  ✓ Identifying Potential Knockout Reactions
    ✓ Production Envelopes
    ✓ Knockout Calculation & Simulation
    ✓ OptKnock Supporting Functions in the Cobra Toolbox

• GDLS

• OptGene
Reducing OptKnock Computational Time

- OptKnock can take a large amount of time to identify the reactions to knockout. The computational time can be significantly reduced by limiting the number of potential 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'
  ✓ Transport Reactions:
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

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

<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>'ACKr'</td>
</tr>
<tr>
<td>'CYTBD'</td>
<td>'GLUDy'</td>
</tr>
<tr>
<td>'D_LACt2'</td>
<td>'GLUN'</td>
</tr>
<tr>
<td>'ETOHt2r'</td>
<td>'GLUN'</td>
</tr>
<tr>
<td>'EX_ac(e)'</td>
<td>'H2Ot'</td>
</tr>
<tr>
<td>'EX_acald(e)'</td>
<td>'H2Ot'</td>
</tr>
<tr>
<td>'EX_acyg(e)'</td>
<td>'H2Ot'</td>
</tr>
<tr>
<td>'EX_co2(e)'</td>
<td>'H2Ot'</td>
</tr>
<tr>
<td>'EX_for(e)'</td>
<td>'H2Ot'</td>
</tr>
<tr>
<td>'EX_fum(e)'</td>
<td>'H2Ot'</td>
</tr>
<tr>
<td>'EX_glc(e)'</td>
<td>'MALt2_2'</td>
</tr>
<tr>
<td>'EX_gln_L(e)'</td>
<td>'NADH16'</td>
</tr>
<tr>
<td>'EX_glL(e)'</td>
<td>'NH4t'</td>
</tr>
<tr>
<td>'EX_h2o(e)'</td>
<td>'O2t'</td>
</tr>
</tbody>
</table>

IdentifyingUnwantedKnockoutReactions.m
Identifying Unwanted Knockout Reactions

% IdentifyingUnwantedKnockoutReactions.m

clear; clc;

% Input the E.coli core model and set environmental conditions
load('ecoli_textbook.mat');
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');

% Perform FBA with Biomass_Ecoli_core_N(w/GAM)_Nmet2 as the objective,
FBAsolution = optimizeCbModel(model,'max')

% Identify non-transport reactions
[transRxns,nonTransRxns] = findTransRxns(model,true);

% Removing ATPM and biomass function
[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}};

disp('Transport Reactions');
disp(transRxns);

disp('Nontransport Reactions');
disp(nonTransRxns);

disp('selectedRxns');
disp(selectedRxns);
## Transport, Non-transport & Selected Reactions

### Transport (transRxns) Reactions

<table>
<thead>
<tr>
<th>Reaction</th>
<th>Transport (transRxns) Reactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>'ACALD'</td>
<td>'EX_h(e)'</td>
</tr>
<tr>
<td>'ACt2r'</td>
<td>'EX_lac_D(e)'</td>
</tr>
<tr>
<td>'AKGt2r'</td>
<td>'EX_mal_L(e)'</td>
</tr>
<tr>
<td>'ATPS4r'</td>
<td>'EX_nh4(e)'</td>
</tr>
<tr>
<td>'CO2t'</td>
<td>'EX_o2(e)'</td>
</tr>
<tr>
<td>'CYTBD'</td>
<td>'EX_pi(e)'</td>
</tr>
<tr>
<td>'D_LACT2'</td>
<td>'EX_pyr(e)'</td>
</tr>
<tr>
<td>'ETOHt2r'</td>
<td>'EX_succ(e)'</td>
</tr>
<tr>
<td>'EX_ac(e)'</td>
<td>'FORt2'</td>
</tr>
<tr>
<td>'EX_acald(e)'</td>
<td>'FRUupts2'</td>
</tr>
<tr>
<td>'EX_ako(e)'</td>
<td>'FUMt2_2'</td>
</tr>
<tr>
<td>'EX_co2(e)'</td>
<td>'GLCqts'</td>
</tr>
<tr>
<td>'EX_for(e)'</td>
<td>'GLNabc'</td>
</tr>
<tr>
<td>'EX_fru(e)'</td>
<td>'GLUt2r'</td>
</tr>
<tr>
<td>'EX_fum(e)'</td>
<td>'H2O(e)'</td>
</tr>
<tr>
<td>'EX_gle(e)'</td>
<td>'MALt2_2'</td>
</tr>
<tr>
<td>'EX_gln_L(e)'</td>
<td>'NADH16'</td>
</tr>
<tr>
<td>'EX_glu_L(e)'</td>
<td>'NH4t'</td>
</tr>
<tr>
<td>'EX_h2o(e)'</td>
<td>'O2t'</td>
</tr>
</tbody>
</table>

### Non-transport (nonTransRxns) Reactions

<table>
<thead>
<tr>
<th>Reaction</th>
<th>Non-transport (nonTransRxns) Reactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>'ACALD'</td>
<td>'GLUSy'</td>
</tr>
<tr>
<td>'ACKr'</td>
<td>'GND'</td>
</tr>
<tr>
<td>'ACONTa'</td>
<td>'ICDHyr'</td>
</tr>
<tr>
<td>'ACONTb'</td>
<td>'ICl'</td>
</tr>
<tr>
<td>'ADK1'</td>
<td>'LIU'</td>
</tr>
<tr>
<td>'AKGDH'</td>
<td>'MALS'</td>
</tr>
<tr>
<td>'ALCD2x'</td>
<td>'MDH'</td>
</tr>
<tr>
<td>'ATPM'</td>
<td>'ME1'</td>
</tr>
<tr>
<td>'Biomass'</td>
<td>'ME2'</td>
</tr>
<tr>
<td>'CS'</td>
<td>'NADTRHD'</td>
</tr>
<tr>
<td>'ENO'</td>
<td>'PDH'</td>
</tr>
<tr>
<td>'FBA'</td>
<td>'PFK'</td>
</tr>
<tr>
<td>'FBP'</td>
<td>'PFL'</td>
</tr>
<tr>
<td>'FRD7'</td>
<td>'PGI'</td>
</tr>
<tr>
<td>'FUM'</td>
<td>'PGK'</td>
</tr>
<tr>
<td>'G6PDH2r'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'GAPD'</td>
<td>'PGM'</td>
</tr>
<tr>
<td>'GLNS'</td>
<td>'PPC'</td>
</tr>
<tr>
<td>'GLUDy'</td>
<td>'PPCK'</td>
</tr>
<tr>
<td>'GLUN'</td>
<td>'PPS'</td>
</tr>
</tbody>
</table>

### Selected Reactions

<table>
<thead>
<tr>
<th>Reaction</th>
<th>Selected Reactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>'ACALD'</td>
<td>'GLUSy'</td>
</tr>
<tr>
<td>'ACKr'</td>
<td>'GND'</td>
</tr>
<tr>
<td>'ACONTa'</td>
<td>'ICDHyr'</td>
</tr>
<tr>
<td>'ACONTb'</td>
<td>'ICl'</td>
</tr>
<tr>
<td>'ADK1'</td>
<td>'LIU'</td>
</tr>
<tr>
<td>'AKGDH'</td>
<td>'MALS'</td>
</tr>
<tr>
<td>'ALCD2x'</td>
<td>'MDH'</td>
</tr>
<tr>
<td>'CS'</td>
<td>'ME1'</td>
</tr>
<tr>
<td>'ENO'</td>
<td>'ME2'</td>
</tr>
<tr>
<td>'FBA'</td>
<td>'NADTRHD'</td>
</tr>
<tr>
<td>'FBP'</td>
<td>'PDH'</td>
</tr>
<tr>
<td>'FRD7'</td>
<td>'PFK'</td>
</tr>
<tr>
<td>'FUM'</td>
<td>'PFL'</td>
</tr>
<tr>
<td>'G6PDH2r'</td>
<td>'PGI'</td>
</tr>
<tr>
<td>'GLNS'</td>
<td>'PGK'</td>
</tr>
<tr>
<td>'GLUDy'</td>
<td>'PPC'</td>
</tr>
<tr>
<td>'GLUN'</td>
<td>'PPCK'</td>
</tr>
<tr>
<td>'GAPD'</td>
<td>'PPS'</td>
</tr>
</tbody>
</table>

---

IdentifyingUnwantedKnockoutReactions.m
## Potential Knockout Reactions

<table>
<thead>
<tr>
<th>Reaction ID</th>
<th>Reaction Name</th>
<th>Gene/Enzyme</th>
<th>Reaction ID</th>
<th>Reaction Name</th>
<th>Gene/Enzyme</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>'ACALD'</td>
<td>'ACALD'</td>
<td>26</td>
<td>'EX_fru(e)'</td>
<td>'GLNS'</td>
</tr>
<tr>
<td>2</td>
<td>'ACALDt'</td>
<td>'ACALD'</td>
<td>27</td>
<td>'EX_fum(e)'</td>
<td>'GLNabc'</td>
</tr>
<tr>
<td>3</td>
<td>'ACKr'</td>
<td>'GLUDy'</td>
<td>28</td>
<td>'EX_glc(e)'</td>
<td>'GLUDy'</td>
</tr>
<tr>
<td>4</td>
<td>'ACONTa'</td>
<td>'GLUN'</td>
<td>29</td>
<td>'EX_glu_L(e)'</td>
<td>'Pit2r'</td>
</tr>
<tr>
<td>5</td>
<td>'ACONTb'</td>
<td>'GLUSy'</td>
<td>30</td>
<td>'EX_glu_L(e)'</td>
<td>'PPCK'</td>
</tr>
<tr>
<td>6</td>
<td>'ACt2r'</td>
<td>'GLUh2r'</td>
<td>31</td>
<td>'EX_h2o(e)'</td>
<td>'Pit2r'</td>
</tr>
<tr>
<td>7</td>
<td>'ADK1'</td>
<td>'GND'</td>
<td>32</td>
<td>'EX_h(e)'</td>
<td>'PTAr'</td>
</tr>
<tr>
<td>8</td>
<td>'AKGDH'</td>
<td>'H2Ot'</td>
<td>33</td>
<td>'EX_lac_D(e)'</td>
<td>'PYK'</td>
</tr>
<tr>
<td>9</td>
<td>'AKGl2r'</td>
<td>'ICDHyr'</td>
<td>34</td>
<td>'EX_mal_L(e)'</td>
<td>'PYRt2r'</td>
</tr>
<tr>
<td>10</td>
<td>'ALCD2x'</td>
<td>'ICL'</td>
<td>35</td>
<td>'EX_nh4(e)'</td>
<td>'RPE'</td>
</tr>
<tr>
<td>11</td>
<td>'ATPM'</td>
<td>'LDH_D'</td>
<td>36</td>
<td>'EX_o2(e)'</td>
<td>'RPI'</td>
</tr>
<tr>
<td>12</td>
<td>'ATPS4r'</td>
<td>'MALS'</td>
<td>37</td>
<td>'EX_pi(e)'</td>
<td>'SUCCt2_2'</td>
</tr>
<tr>
<td>13</td>
<td>Biomass</td>
<td>'ME1'</td>
<td>38</td>
<td>'EX_pyr(e)'</td>
<td>'SUCCt2_2'</td>
</tr>
<tr>
<td>14</td>
<td>'CO2t'</td>
<td>'MDH'</td>
<td>39</td>
<td>'EX_succ(e)'</td>
<td>'SUCCI'</td>
</tr>
<tr>
<td>15</td>
<td>'CS'</td>
<td>'ME2'</td>
<td>40</td>
<td>'FBA'</td>
<td>'SUCOAS'</td>
</tr>
<tr>
<td>16</td>
<td>'CYTBD'</td>
<td>'ME2'</td>
<td>41</td>
<td>'FBP'</td>
<td>'TALa'</td>
</tr>
<tr>
<td>17</td>
<td>'D_LAc2'</td>
<td>'NADHt16'</td>
<td>42</td>
<td>'FORt2'</td>
<td>'THD2'</td>
</tr>
<tr>
<td>18</td>
<td>'ENO'</td>
<td>'NADTRHD'</td>
<td>43</td>
<td>'FORti'</td>
<td>'TKT1'</td>
</tr>
<tr>
<td>19</td>
<td>'ETOHt2r'</td>
<td>'NHz4'</td>
<td>44</td>
<td>'FRD7'</td>
<td>'TKT2'</td>
</tr>
<tr>
<td>20</td>
<td>'EX_ac(e)'</td>
<td>'O2t'</td>
<td>45</td>
<td>'FRUpts2'</td>
<td>'TPI'</td>
</tr>
<tr>
<td>21</td>
<td>'EX_acald(e)'</td>
<td>'PDH'</td>
<td>46</td>
<td>'FUM'</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>'EX_aka(e)'</td>
<td>'FUMt2_2'</td>
<td>47</td>
<td>'FUMt2_2'</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>'EX_co2(e)'</td>
<td>'G6PDH2r'</td>
<td>48</td>
<td>'G6PDH2r'</td>
<td>'FKK'</td>
</tr>
<tr>
<td>24</td>
<td>'EX_etoh(e)'</td>
<td>'GAPD'</td>
<td>49</td>
<td>'GAPD'</td>
<td>'PGI'</td>
</tr>
<tr>
<td>25</td>
<td>'EX_for(e)'</td>
<td>'GLCpts'</td>
<td>50</td>
<td>'GLCpts'</td>
<td>'PGK'</td>
</tr>
<tr>
<td>51</td>
<td>'GLNS'</td>
<td></td>
<td>76</td>
<td>'PGL'</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>'GLNabc'</td>
<td></td>
<td>77</td>
<td>'PGM'</td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>'GLUDy'</td>
<td></td>
<td>78</td>
<td>'Pit2r'</td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>'GLUN'</td>
<td></td>
<td>79</td>
<td>'PPC'</td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>'GLUSy'</td>
<td></td>
<td>80</td>
<td>'PPCK'</td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>'GLUh2r'</td>
<td></td>
<td>81</td>
<td>'Pit2r'</td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>'GND'</td>
<td></td>
<td>82</td>
<td>'PTAr'</td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>'H2Ot'</td>
<td></td>
<td>83</td>
<td>'PYK'</td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>'ICDHyr'</td>
<td></td>
<td>84</td>
<td>'PYRt2r'</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>'ICL'</td>
<td></td>
<td>85</td>
<td>'RPE'</td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>'LDH_D'</td>
<td></td>
<td>86</td>
<td>'RPI'</td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>'MALS'</td>
<td></td>
<td>87</td>
<td>'SUCCt2_2'</td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>'MALt2_2'</td>
<td></td>
<td>88</td>
<td>'SUCCI'</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>'MDH'</td>
<td></td>
<td>89</td>
<td>'SUCCI'</td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>'ME1'</td>
<td></td>
<td>90</td>
<td>'SUCOAS'</td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>'ME2'</td>
<td></td>
<td>91</td>
<td>'TALa'</td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>'NADHt16'</td>
<td></td>
<td>92</td>
<td>'THD2'</td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>'NADTRHD'</td>
<td></td>
<td>93</td>
<td>'TKT1'</td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>'NHz4'</td>
<td></td>
<td>94</td>
<td>'TKT2'</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>'O2t'</td>
<td></td>
<td>95</td>
<td>'TPI'</td>
<td></td>
</tr>
</tbody>
</table>

### Diagram Notes
- **Black** - Potential knockout reactions
- **Red** - Reactions to NOT include on optKnock process
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.m)

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

```
lcolor = 'b';
targetRxn = 'EX_etoh(e)';
biomassRxn = 'Biomass_Ecoli_core_N(w/GAM)_Nmet2';
geneDelFlag = false; % Genes(true) or reactions(false)
nPts = 50;
deletions = {'FUM','G6PDH2r','GLUDy','PTAr','SUCDi'};
[biomassValues,targetValues] = productionEnvelope(model,deletions,lineColor,targetRxn,biomassRxn,geneDelFlag,nPts);
xlabel('Biomass (mmol/g DW-hr)')
ylabel('EX-etoh(e)(mmol/g DW-hr)')
```

Production Envelopes
(EthanolProduction_OptKnock.m)

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

Desired production state
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);
```

Don't plot all secreted metabolites only those that are growth coupled
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);
```

Plot all secreted metabolites
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
load('ecoli_textbook.mat');
% Set uptake rates
model = changeRxnBounds(model,'EX_glc(e)',-10,'b');
model = changeRxnBounds(model,'EX_o2(e)',-0,'b');
% 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 in wild type cell.

Set as minimum Growth rate

Maximum ethanol production with no growth

Robustness Analysis
Alternate Optimal Flux Vectors for Ethanol Production

% findingOptimalSolutionsEthanol.m

clear;

load('ecoli_textbook.mat');

model = changeRxnBounds(model,'EX_glc(e)',-10,'1');
model = changeRxnBounds(model,'EX_o2(e)',0,'1');

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

solverOK = changeCobraSolver('glpk');

% List optimal solutions
[solutions] = enumerateOptimalSolutions(model);
FVA Chart for Glucose-based Anaerobic Ethanol Production
Ethanol Production: Wild Type

- Pumps protons to create protomotive force
- Creates nad\([c]\) and nadph\([c]\)

Graph showing production rate vs. growth rate for wild type with three lines:
- EX_ac(e)
- EX_ etoh(e)
- EX_for(e)
Enhanced OptKnock Example: Ethanol Production
(EthanolProduction_OptKnock.m)

% EthanolProduction_OptKnock_Gurobi_Revised.m
clear; % Input the E.coli core model
load('ecoli_textbook.mat');
% 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 = 3;
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'
% Print out growth rate and minimum & maximum secretion rate
[growthRate,minProd,maxProd] = testOptKnockSol(model,'EX_etoh(e)',optKnockSol.rxnList)
deletions = 'ACKr','G6PDH2r','GLUDy'
growthRate = 0.1772
minProd = 17.31; maxProd = 17.31

Candidate Reactions to be knocked out (want to keep as small as possible)
1 Knockout

EthanolProduction_Mutant.m
EthanolProduction_multiProductionEnvelope.m

Knockout Results:

- EthanolProduction_Mutant.m
- EthanolProduction_multiProductionEnvelope.m

Graph showing production rates:
- Ethanol
- Succinate

Equation:
(0.1800, 16.59)

Diagram highlighting:
- Block Formate Pathway
- Secreting Succinate
- Ethanol Secretion
- ATP Production
- NADPH Production
2 Knockouts

'PGL', 'G6PDH2r', and 'GND' are a correlated reaction set

'PTAr', 'PGL'

Correlated Reaction Set

Ethanol Secretion

Formate Secretion

Oxidative Pathway

Acetate Pathway

ATP Production

NADPH Production

Ethanol

Formate

(0.1876, 17.15)
3 Knockouts
EthanolProduction_Mutant.m

'ACKr' and 'PTAr' are a correlated reaction set

'ACKr', 'GLUDy', 'G6PDH2r'

Etheral Production

Formate
(0.1772, 17.31)

Formate Secretion

NADPH Production

ATP Production

Block Oxidative Pathway

Block Acetate Pathway

Reduces NADPH Demand

Ethanol Secretion
Production Envelopes for Different Ethanol Producing Knockouts

Maximum Growth-rate ≥ 0.14

EthanolProduction_OptKnock.m
% AnaerobicEthanolRA.m

clear; clc;

% Input the E.coli core model

load('ecoli_textbook.mat');

% Set uptake rates

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

% Set optimization objective

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);
OptKnock Example: Maximizing Ethanol Production
(EthanolProduction_OptKnock.m)

% EthanolProduction_OptKnock.m
clear; clc;
% Input the E.coli core model
load('ecoli_textbook.mat');

% 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'

deletions = 'PTAr' , 'GLUDy', 'PYK'
growthRate = 0.0852
minProd = 18.3845; maxProd = 18.7054
(0.0852, 18.71)

Reduce the growth rate to increase bioproduction (0.14 -> 0.05)

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

EthanolProduction_Mutant.m

\[(0.0852, 18.71)\]

**Formate Secretion**

**ETH**

**Acetate Pathway**

\[\text{Force NADPH Production in ME2}\]

\[\text{Reduces NADPH Demand}\]

\[\text{ATP Production}\]

\[\text{Ethanol Secretion}\]
Production Envelopes for Different Ethanol Producing Knockouts

Maximum Growth-rate ≥ 0.05

Production curves for different ethanol producing knockouts:
- (‘PFL’) (0.1800, 16.59)
- (‘ACKr’, ‘PYK’) (0.0913, 18.61)
- (‘ACKr’, ‘PYK’, ‘GLUDy’) (0.0852, 18.71)

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

Maximum Growth-rate \( \geq 0.05 \)

Note that only one case includes secondary secretion.

EthanolProduction_OptKnock.m (CPLEX solver)
iAF1260 Model

iAF1260 Metabolic Core
(GlucoseEthanol_iaf1260.pdf)

iAF1260
Amino Acid Metabolism

(GlucoseEthanol_iaf1260.pdf)

iAF1260
Nucleotide Metabolism
(GlucoseEthanol_iaf1260.pdf)

iAF1260
Alternate Carbon Sources
(GlucoseEthanol_iaf1260.pdf)

iAF1260
Cofactor Biosynthesis
(GlucoseEthanol_iaf1260.pdf)

iAF1260
Inorganic Ion Transport
(GlucoseEthanol_iAF1260.pdf)

Fatty Acid Biosynthesis

(GlucoseEthanol_iaf1260.pdf)

Enterobacterial common antigen (ECA), also referred to as an endotoxin, is a family-specific surface antigen shared by all members of the Enterobacteriaceae and is restricted to this family. It is found in freshly isolated wild-type strains as well as in laboratory strains like Escherichia coli K-12. ECA is located in the outer leaflet of the outer membrane. It is a glycopospholipid built up by an aminosugar heteropolymer linked to an L-glycerophosphatidyl residue.

"optKnock" Knockouts for Ethanol Production using the iaf1260 Model

% GlucoseEthanolOptKnock_iaf1260_Reduced_Reactions.m
 clearance;

% Set constraints
load('ecoli_iaf1260.mat');
model = changeRxnBounds(model, {'EX_o2(e)', 'EX_glc(e)'}, [0 -10], 'l');
model = changeObjective(model,'Ec_biomass_iAF1260_core_59p81M');

[transRxns,nonTransRxns] = findTransRxns(model,true); % Remove transport reactions
includedSubSystems = {'Transport, Inner Membrane','Glycerophospholipid Metabolism','Transport, Outer Membrane Porin',...  
'Cell Envelope Biosynthesis','Nucleotide Salvage Pathway','Murein Recycling','Membrane Lipid Metabolism',...  
'Glycerophospholipid Metabolism','Inorganic Ion Transport and Metabolism','Lipopolysaccharide Biosynthesis / Recycling',...  
'tRNA Charging','Unassigned','Membrane Lipid Metabolism','Murein Biosynthesis'};
unwantedReactions = model.rxns(ismember(model.subSystems,includedSubSystems));
[tf,rids] = ismember([unwantedReactions;'ATPM';'Ec_biomass_iAF1260_core_59p81M'], nonTransRxns);
idfinal=rids(tf);
nonTransRxns(idfinal) = [];
selectedRxns = nonTransRxns;

% Optknock analysis for Ethanol secretion
disp('Executing optKnock');
options.targetRxn = 'EX_etoh(e)';
options.vMax = 1000;
options.numDel = 3;
options.numDelSense = 'L';
constrOpt.rxnList = {'Ec_biomass_iAF1260_core_59p81M','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)
3 Knockouts – iaf1260

GlycerolEthanolOptknock_iaf1260.m

GlucoseEthanolOptknock_iaf1260_Reduced_Reactions.m

GlucoseEthanol_Mutant_iaf1260.m

('PFL', 'GLUDy', 'TPI')

(0.0644, 19.05)
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
% 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'

d = deletions

growthRate = 0.1772
minProd = 17.3062
maxProd = 17.3062

% 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
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 metabolites?
• Why is there a trade-off between biomass growth and 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.

\[
\text{[gdlsSolution, biLevelMILPproblem, gdlsSolutionStructs] = GDLS(model, varargin)}
\]

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

**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).

Change the model to reduce the number of reactions, metabolites, and boundary conditions to reduce the search space for optimization.
[gdlsSolution, bilevelMILPProblem, 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:
- `'nbhdsz'`: 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)
- `bilevelMILPProblem`: Problem structure used in computation
GDLS Example: Maximizing Ethanol Production

% 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', 3, 'nbhdsz', 2);

deletions = 'ACKr', 'GLUDy', 'PYK'
growthRate = 0.0852
maxProd = 18.7054
## GDLS Example: Iterations

**EthanolProduction_GDLS.m**

### Iteration 1

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass flux</td>
<td>0.091291</td>
</tr>
<tr>
<td>Synthetic flux</td>
<td>[18.268541, 18.612509]</td>
</tr>
<tr>
<td>Knockout cost</td>
<td>2</td>
</tr>
<tr>
<td>Knockouts</td>
<td>ACKr, PYK</td>
</tr>
</tbody>
</table>

### Iteration 2

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass flux</td>
<td>0.085178</td>
</tr>
<tr>
<td>Synthetic flux</td>
<td>[18.384498, 18.705430]</td>
</tr>
<tr>
<td>Knockout cost</td>
<td>3</td>
</tr>
<tr>
<td>Knockouts</td>
<td>ACKr, GLUDy, PYK</td>
</tr>
</tbody>
</table>

### Iteration 3

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass flux</td>
<td>0.085178</td>
</tr>
<tr>
<td>Synthetic flux</td>
<td>[18.384498, 18.705430]</td>
</tr>
<tr>
<td>Knockout cost</td>
<td>3</td>
</tr>
<tr>
<td>Knockouts</td>
<td>ACKr, GLUDy, PYK</td>
</tr>
</tbody>
</table>

**GDLS Example: Iterations**

**EthanolProduction_GDLS.m**

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass flux</td>
<td>0.091291</td>
</tr>
<tr>
<td>Synthetic flux</td>
<td>[18.268541, 18.612509]</td>
</tr>
<tr>
<td>Knockout cost</td>
<td>2</td>
</tr>
<tr>
<td>Knockouts</td>
<td>ACKr, PYK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass flux</td>
<td>0.085178</td>
</tr>
<tr>
<td>Synthetic flux</td>
<td>[18.384498, 18.705430]</td>
</tr>
<tr>
<td>Knockout cost</td>
<td>3</td>
</tr>
<tr>
<td>Knockouts</td>
<td>ACKr, GLUDy, PYK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass flux</td>
<td>0.085178</td>
</tr>
<tr>
<td>Synthetic flux</td>
<td>[18.384498, 18.705430]</td>
</tr>
<tr>
<td>Knockout cost</td>
<td>3</td>
</tr>
<tr>
<td>Knockouts</td>
<td>ACKr, GLUDy, PYK</td>
</tr>
</tbody>
</table>
GDLS Example: Location of Knockouts for the GDLS Mutant

Ethanol

ACKr, GLUDy, PYK

EthanolProduction_Mutant.m

Same as OptKnock Results

EthanolProduction_multiProductionEnvelope.m
GDLS Example: Maximizing Ethanol Production using Genes

```matlab
% EthanolProduction_GDLS_Gene.m
clear;

% Set operating conditions
load('ecoli_textbook.mat');
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');

selectedRxns = model.genes; % Using all model genes

% GDLS analysis for Ethanol secretion
[gdlsSolution, bilevelMILPproblem, gdlsSolutionStructs] = GDLS(model, 'EX_etoh(e)', 'minGrowth', 0.05, ...'
    'koType','genes'
    'selectedRxns', selectedRxns, 'maxKO', 3, 'nbhdsz', 2);

gdlsSolution.KOs
[results ListResults] = findRxnsFromGenes(model, gdlsSolution.KOs); % Reaction names located in "results" structure
```

Gene deletions = b1676, b1761, 1849
Reaction deletions = 'ACKr', 'GLUDy', 'PYK'
growthRate = 0.0852
maxProd = 18.7054
**GDLS Gene Example:**
Location of Knockouts for the GDLS Mutant

- **Ethanol Production Mutant.m**
- **EthanolProduction_multiProductionEnvelope.m**

ACKr, GLUDy, PYK

Same as OptKnock Results
“GDLS” Gene Knockouts for Ethanol Production Using the iaf1260 Model

% EthanolProduction_GDLS_Gene_iaf1260.m

clear;

% Set operating conditions
load('ecoli_iaf1260.mat');
model = changeRxnBounds(model,'EX_glc(e)',-10,'l');
model = changeRxnBounds(model,'EX_o2(e)',0,'l');
model = changeObjective(model,'Ec_biomass_iAF1260_core_59p81M');

selectedRxns = model.genes;

% GDLS analysis for Ethanol secretion
[gdlsSolution, bilevelMILPproblem, gdlsSolutionStructs] = GDLS(model, 'EX_etoh(e)', 'minGrowth', 0.05, ...
   'koType','genes','selectedRxns', selectedRxns, 'maxKO', 3, 'nbhdsz', 2);

gdlsSolution.KOs
[results ListResults] = findRxnsFromGenes(model, gdlsSolution.KOs)
3 Gene Knockouts - GDLS

GlucoseEthanolGDLS_iaf1260.m

Production Rate (mmol/GDW h) vs. Growth Rate (1/h)

('b1761', 'b3919', 'b3951')
('GLUDy', 'TPI', 'PFL')

GlucoseEthanol_Mutant_iaf1260.m
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

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.

\[ [x, \text{population}, \text{scores}, \text{optGeneSol}] = \text{OptGene}(\text{model}, \text{targetRxn}, \text{substrateRxn}, \text{generxnList}, \text{maxKOs}, [\text{population}]) \]

where: \text{targetRxn} specifies the reaction to optimize; \text{substrateRxn} specifies the exchange reaction for the carbon source; \text{generxnList} is a cell array of strings that specifies which genes or reactions are allowed to be deleted; and \text{maxKOs} sets the maximum number of knockouts; \text{x} is the best scoring set as determined by the functions \text{optGeneFitness} or \text{optGeneFitnessTilt}; \text{population} is the binary matrix representing the knockout sets; and \text{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.

% Ethanol_OptGene_core.m
clear; clc;
% Set conditions
load('ecoli_textbook.mat');
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,3);

% 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.m
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
New Cobra Toolbox Functions

Identify non-transport/exchange reactions
[transRxns,nonTransRxns] = findTransRxns(model,true);

Plot a production envelope
[biomassValues,targetValues] = productionEnvelope(model,deletions,lineColor,targetRxn,biomassRxns,geneDelFlag,nPts);

Plot a multiproduction envelope
[biomassValues,targetValues] = productionEnvelope(model,deletions,lineColor,targetRxn,biomassRxns,geneDelFlag,nPts);

OptKnock Analysis
optKnockSol = OptKnock(model, selectedRxns, options, constrOpt);

OptKnock Supporting Functions
[type,maxGrowth,maxProd] = analyzeOptKnock(model,deletions,target,biomassRxns,geneDelFlag)
[growthRate,minProd,maxProd] = testOptKnockSol(model,target,optKnockSol.rxnList)
[wtRes,delRes] = simpleOptKnock(model,target,deletions,false,minGrowth,doubleDelFlag)

GDLS Analysis
[gdlSolution,bilevelMILPProblem,gdlSolutionStructs] = GDLS(model,varargin)

optGene Analysis
[x,population,scores,optGeneSol] = OptGene(model,targetRxn,substrateRxn,generxnList,maxKOs,[population])
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';
doubleProductionEnvelope

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

```matlab
% 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');

del deletions = {};
bio biomassRxn = {'Biomass_Ecoli_core_N(w/GAM)_Nmet2'};
[x1,x2,y] = doubleProductionEnvelope(model,deletions,'EX_etoh(e)','EX_for(e)',biomassRxn,false,20);
```
Ethanol Production: Exchange Fluxes
3 Knockouts (EthanolProduction_Mutant.m)

clear; clc;
load('ecoli_textbook.mat');

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,'G6PDH2r',0,'b');
model = changeRxnBounds(model,'GLUDy',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);
Ethanol Production: Fluxes

3 Knockouts (EthanolProduction_Mutant.m)

clear; clc;
load('ecoli_textbook.mat');

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,'G6PDH2r',0,'b');
model = changeRxnBounds(model,'GLUDy',0,'b');

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
3 Knockouts (EthanolProduction_Mutant.m)

clear; clc;
load('ecoli_textbook.mat');

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,'G6PDH2r',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);
Ethanol Production: Robustness Analysis
3 Knockouts (EthanolProduction_Mutant.m)

```matlab
clear; clc;
load('ecoli_textbook.mat');

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,'G6PDH2r',0,'b');
model = changeRxnBounds(model,'GLUDy',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);
```
Ethanol Production: Production Envelope

3 Knockouts (EthanolProduction_multiProductionEnvelope.m)

```matlab
% EthanolProduction_multiProductionEnvelope.m

clear;

load('ecoli_textbook.mat');

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','G6PDH2r'};

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

%Show only growth coupled metabolites

figure(1)

[biomassValues,targetValues] = multiProductionEnvelope(model,deletions,biomassRxn,false,20,false);
```

Knockouts

- ACKr
- G6PDH2r
- GLUDy

Ethanol: (0.1772, 17.31)

Formate
### Mapping FVA Classifications

(Removing Transport and Exchange Reactions)

<table>
<thead>
<tr>
<th>Partially-Coupled Reactions</th>
<th>Not-Coupled Reactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>'ACONTa'</td>
<td>'ACALD'</td>
</tr>
<tr>
<td>'ACONTb'</td>
<td>'ME1'</td>
</tr>
<tr>
<td>'CS'</td>
<td>'ME2'</td>
</tr>
<tr>
<td>'FBA'</td>
<td>'NADH16'</td>
</tr>
<tr>
<td>'FRD7'</td>
<td>'NADTRHD'</td>
</tr>
<tr>
<td>'GLNS'</td>
<td>'PGK'</td>
</tr>
<tr>
<td>'ICDHyr'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'PFK'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'PFL'</td>
<td>'PGM'</td>
</tr>
<tr>
<td>'G6PDH2r'</td>
<td>'PPCK'</td>
</tr>
<tr>
<td>'PGI'</td>
<td>'RPE'</td>
</tr>
<tr>
<td>'PPC'</td>
<td>'RPE'</td>
</tr>
<tr>
<td>'PTAr'</td>
<td>'RPI'</td>
</tr>
<tr>
<td>'PYK'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'TPi'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'ACALD'</td>
<td>'ME1'</td>
</tr>
<tr>
<td>'ACKr'</td>
<td>'ME2'</td>
</tr>
<tr>
<td>'ADK1'</td>
<td>'NADH16'</td>
</tr>
<tr>
<td>'AKGDH'</td>
<td>'NADTRHD'</td>
</tr>
<tr>
<td>'AKGt2r'</td>
<td>'PDH'</td>
</tr>
<tr>
<td>'ALCD2x'</td>
<td>'PGK'</td>
</tr>
<tr>
<td>'FBA'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'FRD7'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'GLNS'</td>
<td>'PGM'</td>
</tr>
<tr>
<td>'ICDHyr'</td>
<td>'PPCK'</td>
</tr>
<tr>
<td>'PFK'</td>
<td>'RPE'</td>
</tr>
<tr>
<td>'PFL'</td>
<td>'RPI'</td>
</tr>
<tr>
<td>'G6PDH2r'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PGI'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PPC'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PTAr'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PYK'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'TPi'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'ACALD'</td>
<td>'ME1'</td>
</tr>
<tr>
<td>'ACKr'</td>
<td>'ME2'</td>
</tr>
<tr>
<td>'ADK1'</td>
<td>'NADH16'</td>
</tr>
<tr>
<td>'AKGDH'</td>
<td>'NADTRHD'</td>
</tr>
<tr>
<td>'AKGt2r'</td>
<td>'PDH'</td>
</tr>
<tr>
<td>'ALCD2x'</td>
<td>'PGK'</td>
</tr>
<tr>
<td>'FBA'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'FRD7'</td>
<td>'PGL'</td>
</tr>
<tr>
<td>'GLNS'</td>
<td>'PGM'</td>
</tr>
<tr>
<td>'ICDHyr'</td>
<td>'PPCK'</td>
</tr>
<tr>
<td>'PFK'</td>
<td>'RPE'</td>
</tr>
<tr>
<td>'PFL'</td>
<td>'RPI'</td>
</tr>
<tr>
<td>'G6PDH2r'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PGI'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PPC'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PTAr'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'PYK'</td>
<td>'SUCDi'</td>
</tr>
<tr>
<td>'TPi'</td>
<td>'SUCDi'</td>
</tr>
</tbody>
</table>
### Parsimonious FBA For Glucose in an Anaerobic Environment

#### Essential pFBA Optima
- 'ACONTa'
- 'ACALD'
- 'FRD7'
- 'ACALD1'
- 'ME2'
- 'CYTBD'

#### Enzymatically Less Efficient
- 'ACONTb'
- 'ACKr'
- 'SUCDi'
- 'ADK1'
- 'NADH16'
- 'EX_fru(e)'

#### Metabolically Less Efficient
- 'ACI2r'
- 'AKGDH'
- 'NADTRHD'
- 'EX_fum(e)'

#### pFBA No-flux
- 'CS'
- 'ALCD2x'
- 'AKG12r'
- 'PDH'
- 'EX_gln_L(e)'

#### Blocked
- 'ENO'
- 'ATPM'
- 'D_LAC12'
- 'PGL'
- 'EX_mal_L(e)'

#### Other
- 'EX_glucose(e)'
- 'ATPS4r'
- 'EX_acald(e)'
- 'PPCK'
- 'EX_o2(e)'

### Diagram
- **pFBA_Ecoli_Core.m (Anaerobic)**
  - **PPP**
  - **Glyc**
  - **Ana**
  - **TCA**
  - **Ferm**

---

**Constraint-based Metabolic Reconstructions & Analysis**

H. Scott Hinton, 2017

**Lesson: Gene/Reaction Knockouts**

---

**Utah State University**

**BIE 5500/6500**
H. Scott Hinton, 2017

FVA Chart for Glucose-based Anaerobic Ethanol Production

FluxVariabilityEthanol.m

FVA Ethanol Production.xlsx

PYK

GLUDy

PFL

PTAr