Table of Contents

Introduction to Jupyter notebook and Binder

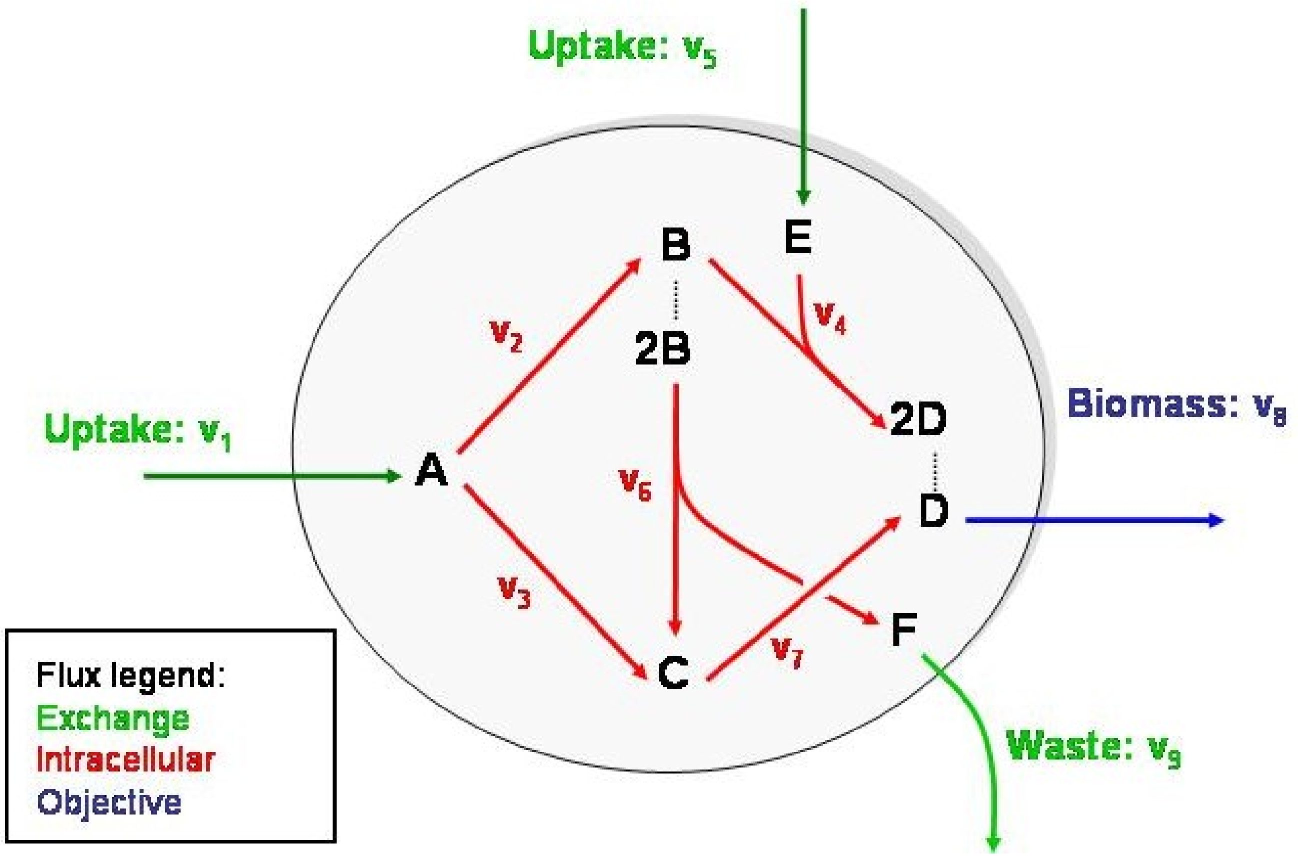
Launch the course in Binder!

launch binder
(https://mybinder.org/v2/gh/agilebiofoundry/2021-07-16-cell-factory-design/master)

ABC model of metabolism

Everything we talk about here extends to genome-scale models, and the only reason we are discussing this simple model is so that the entire metabolism fits in your head, and you can gain intuitions about how the genome-scale methods actually work.

Imagine we have a complete model of all the biochemical reactions in the cell, so nutrients A and E enter the cell, bioproducts D and F leave the cell. There is a reaction that takes A to C to D and another reaction that takes A to B then either 2 moles of B recombine to form C and F, or B combines with E to form 2 moles of D.



The chemical equations for the ABC model are:

$$\begin{array}{rrcl}
R_1: & & \overset{v_1}{\rightarrow} & A \\
R_2: & A & \overset{v_2}{\rightarrow} & B \\
R_3: & A & \overset{v_3}{\rightarrow} & C \\
R_4: & B+E & \overset{v_4}{\rightarrow} & 2D \\
R_5: & & \overset{v_5}{\rightarrow} & E \\
R_6: & 2B & \overset{v_6}{\rightarrow} & C+F \\
R_7: & C & \overset{v_7}{\rightarrow} & D \\
R_8: & D & \overset{v_8}{\rightarrow} & \\
R_9: & F & \overset{v_9}{\rightarrow} &
\end{array}$$

where R_j are the reaction equations, the v_j are the reaction rates, or fluxes, and the arrow direction indicates the reaction is irreversible. These chemical equations can also be represented mathematically as a Stoichiometric matrix S:

$$S = \left[\begin{array}{cccccccccc}
 & R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 & R_8 & R_9 \\
A & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
B & 0 & 1 & 0 & -1 & 0 & -2 & 0 & 0 & 0 \\
C & 0 & 0 & 1 & 0 & 0 & 1 & -1 & 0 & 0 \\
D & 0 & 0 & 0 & 2 & 0 & 0 & 1 & -1 & 0 \\
E & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\
F & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1
\end{array}\right]$$

where each column corresponds to a reaction, each row corresponds to a metabolite, and the element corresponds to the stoichiometry of each metabolite in that reaction. The number is negative if the metabolite is a reactant of the reaction, positive if it is a product of the reaction, and zero, otherwise.

Implementation of the ABC model using Cobrapy

We can also represent this toy model in python using the cobrapy library

In [1]:
```
import cobra
cobra_config = cobra.Configuration()
cobra_config.solver='glpk_exact'

# Define the model
abc_model = cobra.Model('ABC_model')
# create new metabolites
A  = cobra.Metabolite('A',compartment='c')
B  = cobra.Metabolite('B',compartment='c')
C  = cobra.Metabolite('C',compartment='c')
D  = cobra.Metabolite('D',compartment='c')
E  = cobra.Metabolite('E',compartment='c')
F  = cobra.Metabolite('F',compartment='c')

# Add the new metabolites to the model
abc_model.add_metabolites([A,B,C,D,E,F])

# Create new reactions
R_1 = cobra.Reaction('R_1')
R_2 = cobra.Reaction('R_2')
R_3 = cobra.Reaction('R_3')
R_4 = cobra.Reaction('R_4')
R_5 = cobra.Reaction('R_5')
R_6 = cobra.Reaction('R_6')
R_7 = cobra.Reaction('R_7')
R_8 = cobra.Reaction('R_8')
R_9 = cobra.Reaction('R_9')

# Add reactions to the model
abc_model.add_reactions([R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9])

# Generate the stoichiometry from
R_1.build_reaction_from_string('--> A')
R_2.build_reaction_from_string('A --> B')
R_3.build_reaction_from_string('A --> C')
R_4.build_reaction_from_string('B + E --> 2 D')
R_5.build_reaction_from_string('--> E')
R_6.build_reaction_from_string('2 B --> C + F')
R_7.build_reaction_from_string('C --> D')
R_8.build_reaction_from_string('D -->')
R_9.build_reaction_from_string('F -->')

cobra.io.save_json_model(abc_model, 'ABC/ABC_model.json')
cobra.util.array.create_stoichiometric_matrix(abc_model,
                                              array_type='DataFrame').as
type(int)
```

Out[1]:

	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9
A	1	-1	-1	0	0	0	0	0	0
B	0	1	0	-1	0	-2	0	0	0
C	0	0	1	0	0	1	-1	0	0
D	0	0	0	2	0	0	1	-1	0
E	0	0	0	-1	1	0	0	0	0
F	0	0	0	0	0	1	0	0	-1

As you can see, this is identical to the Stoichiometric matrix we defined above:

$$S = \begin{bmatrix} & R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 & R_8 & R_9 \\ A & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 1 & 0 & -1 & 0 & -2 & 0 & 0 & 0 \\ C & 0 & 0 & 1 & 0 & 0 & 1 & -1 & 0 & 0 \\ D & 0 & 0 & 0 & 2 & 0 & 0 & 1 & -1 & 0 \\ E & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix}$$

Steady state analysis of the ABC model

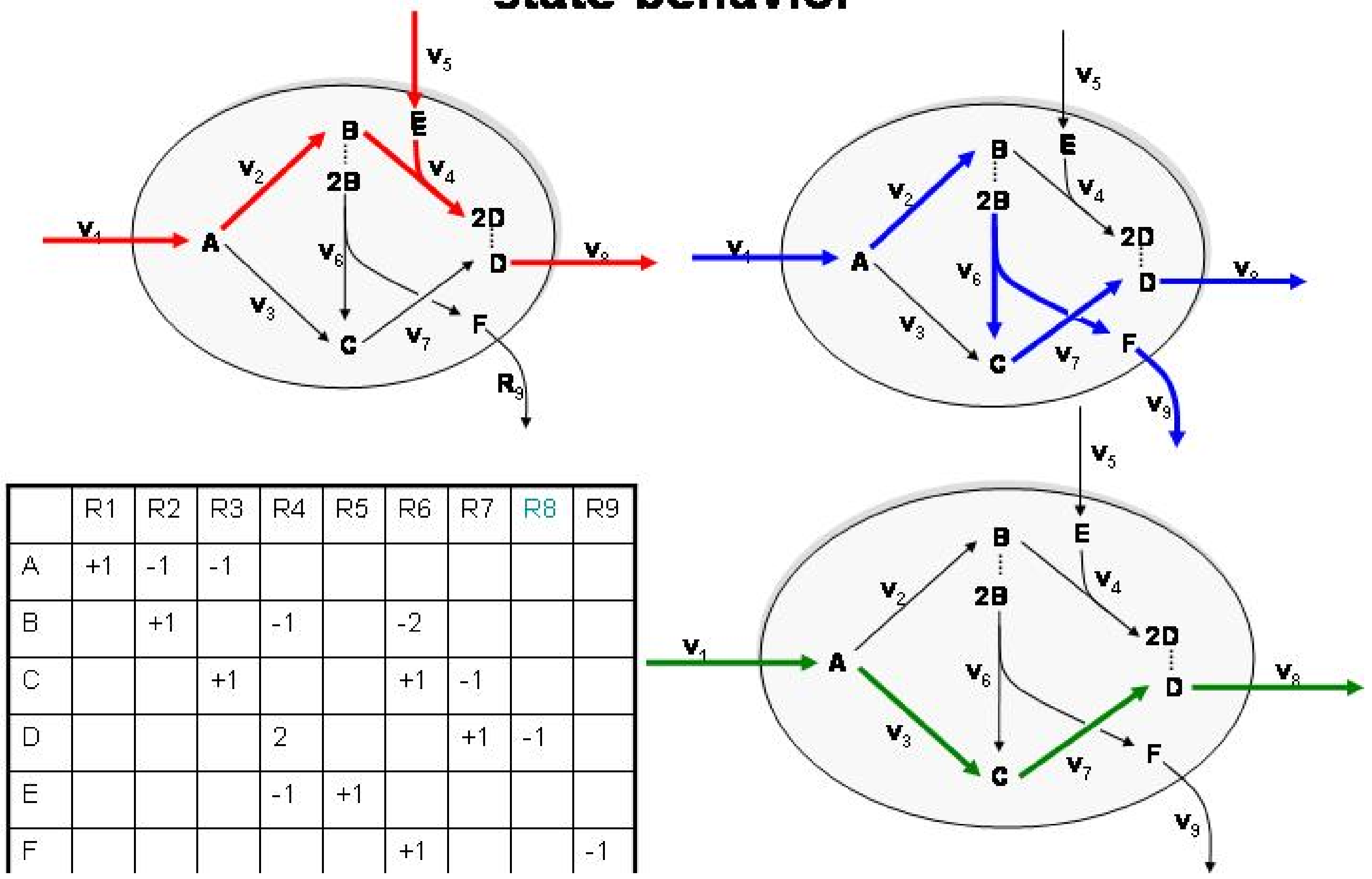
Like the terraced pools of water in the geothermal hot springs of [Pamukkale, Turkey (https://rustytraveltrunk.com/pamukkale/)](https://rustytraveltrunk.com/pamukkale/), when the metabolic network is in steady state, the concentrations of the internal metabolites do not change. Therefore

$$\frac{d\vec{c}}{dt} = S \cdot \vec{v} = 0$$

where $\frac{d\vec{c}}{dt}$ represents the change in metabolite concentrations with respect to time, $\vec{v}$ represents the reaction rates (also known as fluxes), and S is the stoichiometric matrix.

$$\begin{bmatrix} \frac{dA}{dt} \\ \frac{dB}{dt} \\ \frac{dC}{dt} \\ \frac{dD}{dt} \\ \frac{dE}{dt} \\ \frac{dF}{dt} \end{bmatrix} = S \cdot \vec{v} = \left[\begin{array}{cccccccccc} & R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 & R_8 & R_9 \\ A & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 1 & 0 & -1 & 0 & -2 & 0 & 0 & 0 \\ C & 0 & 0 & 1 & 0 & 0 & 1 & -1 & 0 & 0 \\ D & 0 & 0 & 0 & 2 & 0 & 0 & 1 & -1 & 0 \\ E & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \end{array}\right] \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \\ v_9 \end{bmatrix} = \begin{bmatrix} v_1 - v_2 - v_3 \\ v_2 - v_4 - 2v_6 \\ v_3 + v_6 - v_7 \\ 2v_4 + v_7 - v_8 \\ -v_4 + v_5 \\ v_6 - v_9 \end{bmatrix} =$$

Using Elementary modes to study the steady state-behavior



	R1	R2	R3	R4	R5	R6	R7	R8	R9
A	+1	-1	-1						
B		+1		-1		-2			
C			+1			+1	-1		
D				2			+1	-1	
E				-1	+1				
F						+1			-1

An **elementary mode** is a minimal set of reactions that forms a steady-state. It is minimal in the sense that removing any reaction from the set results in a network that cannot achieve steady state.
For the ABC model, all feasible steady-state flux distributions can be decomposed into non-negative combinations of just 3 elementary modes. Although elementary modes are a useful conceptual framework for analyzing small networks, they are not practical for genome-scale network analysis because the number of elementary modes increases exponentially with the size of the network. Nevertheless, for this ABC model keeping in mind these three elementary modes will be helpful when solving the cell factory design problems below.

Cell factory design questions for the ABC model

How can we use a model to predict:

1. the optimal theoretical growth rate?
2. the optimal bioproduct yield?
3. the effect of modulating environmental conditions on bioproduct yield?
4. the effect of genetic perturbations on bioproduct yield?
5. the tradeoff between growth rate and bioproduct yield?

How can we use a model to predict the optimal theoretical growth rate?

Suppose that the uptake rate of A is limited to $10\frac{mmol}{hour}$, and let's imagine that D is biomass. We can assume the uptake rate is constant if we place the organism in a chemostat where we provide a carbon source at replacement rate. What is the maximum growth rate achievable given the constraints of this network? We can find the answer by solving the following optimization problem:

$$\begin{array}{lll} \underset{\vec{v}}{\text{maximize}} & v_8 & \text{Cellular objective (growth)} \\ \text{subject to} & S \cdot \vec{v} = 0 & \text{Balanced Steady-state} \\ & 0 \leq \vec{v} & \text{Irreversible reactions} \\ & v_1 \leq 10 & \text{Uptake rate is } 10\frac{mmol}{hour} \end{array}$$

Cobrapy implementation of the optimization problem

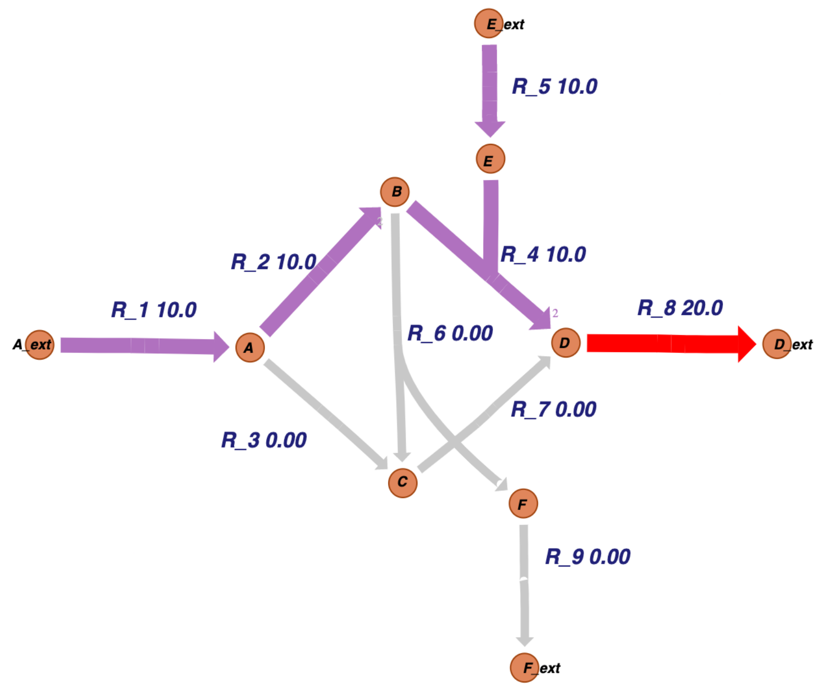
In [2]:
```
abc_model = cobra.io.load_json_model('ABC/ABC_model.json')  # Balanced s
teady state
abc_model.objective = 'R_8'                                   # Cellular o
bjective (growth)
for rxn in abc_model.reactions:
    rxn.lower_bound = 0                                      # Irreversibl
e reactions
abc_model.reactions.R_1.upper_bound = 10                      # Uptake rat
e is 10 mmol/gDW/hour

optimal_growth_rate = abc_model.optimize()                    # Find fluxe
s that maximize growth rate
optimal_growth_rate
```

Out[2]: ***Optimal* solution with objective value 20.000**

	fluxes	reduced_costs
R_1	10.0	4.0
R_2	10.0	0.0
R_3	0.0	-2.0
R_4	10.0	0.0
R_5	10.0	0.0
R_6	0.0	-6.0
R_7	0.0	0.0
R_8	20.0	0.0
R_9	0.0	0.0

We can also visualize the fluxes on the network

In [3]:
```
import escher
reaction_scale = [ { 'type': 'min',  'color': '#c8c8c8', 'size': 12 },
                   { 'type': 'mean', 'color': '#9696ff', 'size': 20 },
                   { 'type': 'max',  'color': '#ff0000', 'size': 25 } ]
escher.Builder( map_json       = 'ABC/ABC_map.json',
                model          = abc_model,
                reaction_data  = optimal_growth_rate.fluxes.to_dict(),
                reaction_scale = reaction_scale
              )
```



Notice that R_1 is a limiting reagent, or bottleneck. It doesn't matter how much E is available, we can only convert E to D at the rate supplied to B, since they are stoichiometrically constrained by R_4 to be equal. Notice also that this solution is just one of the elementary modes.

How can we use a model to predict the optimal bioproduct yield?

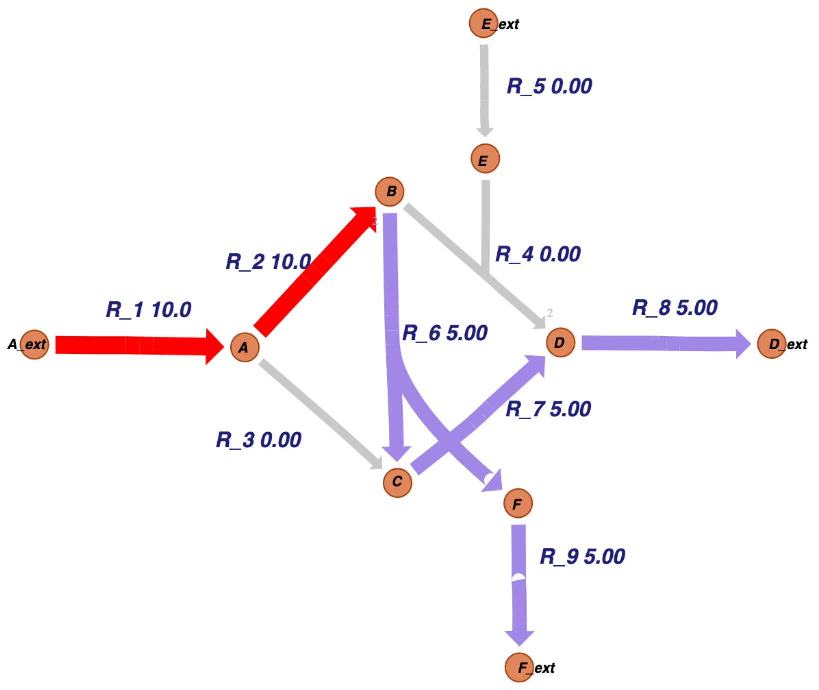
Since we are interested in microbial cell factories, let's think of F as a high-value product. What happens to the flux distribution and the growth rate if we maximize the production of F?

$$\begin{array}{lll}\underset{\vec{\mathbf{v}}}{\text{maximize}} & v_9 & \text{Engineering objective (Bioproduct)} \\ \text{subject to} & S \cdot v = 0 & \text{Balanced steady-state} \\ & 0 \leq v & \text{Irreversible reactions} \\ & v_1 \leq 10 & \text{Uptake rate is } 10\frac{mmol}{hour}\end{array}$$

In [4]:
```
abc_model = cobra.io.load_json_model('ABC/ABC_model.json')     # Balanced
steady-state
abc_model.objective = abc_model.reactions.R_9                   # Engineer
ing objective
for rxn in abc_model.reactions:
    rxn.lower_bound = 0                                         # Irrevers
ible reactions
abc_model.reactions.R_1.upper_bound = 10                        # Uptake r
ate is 10 mmol/hour

optimal_bioproduct_yield = abc_model.optimize()
display(optimal_bioproduct_yield)
escher.Builder( map_json       ='ABC/ABC_map.json',
                model          = abc_model,
                reaction_data  = optimal_bioproduct_yield.fluxes.to_dict
(),
                reaction_scale = reaction_scale
              )
```

***Optimal* solution with objective value 5.000**

	fluxes	reduced_costs
R_1	10.0	1.0
R_2	10.0	0.0
R_3	0.0	-1.0
R_4	0.0	-1.0
R_5	0.0	0.0
R_6	5.0	0.0
R_7	5.0	0.0
R_8	5.0	0.0
R_9	5.0	0.0


E_ext
R_5 0.00
E
B
R_4 0.00
R_2 10.0
R_1 10.0
R_8 5.00
A_ext
A
R_6 5.00
D
D_ext
R_7 5.00
R_3 0.00
C
F
R_9 5.00
F_ext

How can we model the effect of altering environmental conditions on bioproduct yield?

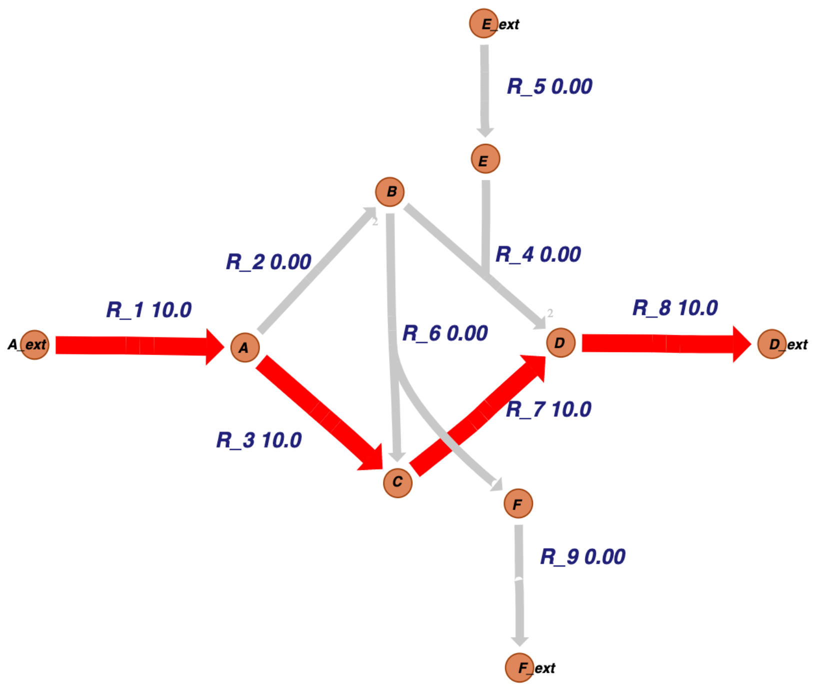
We see from the previous exercise that if the cell wanted to produce the product, then there is a pathway that enables both growth and bioproduction. But cells don't want to produce bioproducts, they just want to grow. How can we align the cellular objective with our engineering objective? Well, by looking at the difference in flux distributions between optimal biomass growth and optimal bioproduct yield, it seems that there is a decision point at the production of B. To optimize growth, B combines with E to generate 2 moles of D. On the other hand to optimize bioproduct, 2 moles of B split to form C and F. Perhaps we can alter the environmental conditions to induce bioproduction. In the ABC model, let's imagine that E plays the role of oxygen in the metabolism of a facultative aerobe: having some makes the growth rate increase, but it is not strictly necessary for growth. What happens to the flux distribution if the cell tries to grow without E (an-E-robically?) Will this result in the desired pathway being utilized?

$$\begin{array}{lll}
\underset{\vec{\mathbf{v}}}{\text{maximize}} & v_8 & \text{Cellular objective (growth)} \\
\text{subject to} & S \cdot v = 0 & \text{Balanced steady-state} \\
& 0 \leq v & \text{Irreversible reactions} \\
& v_1 \leq 10 & \text{Uptake rate is } 10\,\frac{mmol}{hour} \\
& v_5 \leq 0 & \text{An-}E\text{-robic environmental condition}
\end{array}$$

In [5]:
```
abc_model = cobra.io.load_json_model('ABC/ABC_model.json')  # Stoichiome
tric matrix loaded
abc_model.objective = R_8                                    # Cellular o
bjective (growth)
for rxn in abc_model.reactions:
    rxn.lower_bound = 0                                      # Irreversib
le reactions
abc_model.reactions.R_1.upper_bound = 10                     # Uptake rat
e is 10 mmol/hour
abc_model.reactions.R_5.upper_bound = 0                      # An-E-robic
environmental condition

environment_solution = abc_model.optimize()                  # Effect of
 altering environment on bioproduct yield
display(environment_solution)

escher.Builder( map_json       ='ABC/ABC_map.json',
                model          = abc_model,
                reaction_data  = environment_solution.fluxes.to_dict(),
                reaction_scale = reaction_scale,
              )
```

Optimal **solution with objective value 10.000**

	fluxes	reduced_costs
R_1	10.0	2.0
R_2	0.0	0.0
R_3	10.0	0.0
R_4	0.0	0.0
R_5	0.0	2.0
R_6	0.0	-2.0
R_7	10.0	0.0
R_8	10.0	0.0
R_9	0.0	0.0


E_ext
R_5 0.00
E
B
R_4 0.00
R_2 0.00
R_1 10.0
R_8 10.0
A_ext
A
R_6 0.00
D
D_ext
R_7 10.0
R_3 10.0
C
F
R_9 0.00
F_ext

No! The optimal flux distribution in the absence of E is to convert A to C to D

How can we model the effect of genetic perturbations on bioproduct yield?

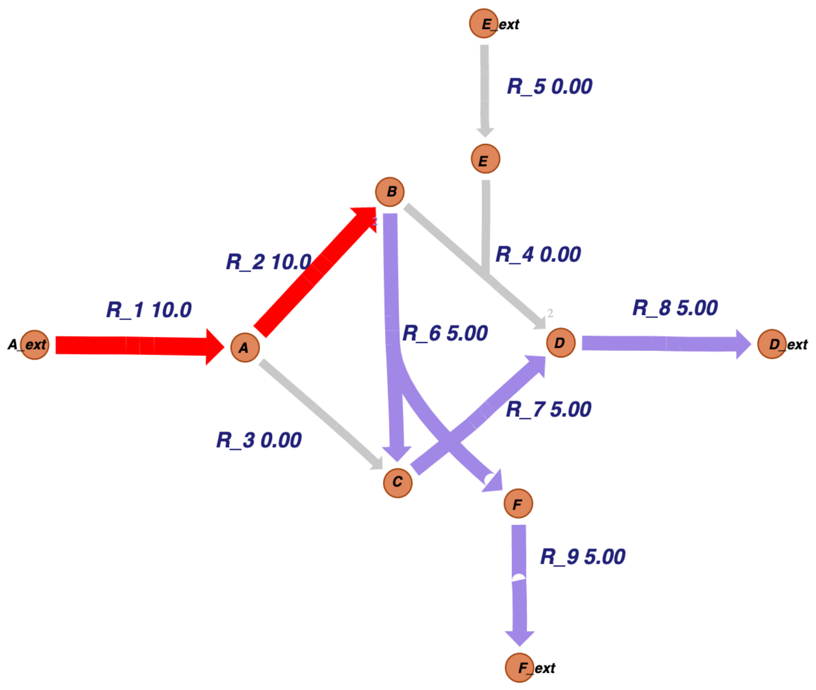
As we saw in the previous exercise, even a simple model can generate counterintuitive results. Perhaps by knocking out the genes that enabled this alternate pathway, we can sculpt the metabolic network towards our objectives.

$$\begin{array}{lll} \underset{\vec{\mathbf{v}}}{\text{maximize}} & v_8 & \text{Cellular objective (growth)} \\ \text{subject to} & S \cdot v = 0 & \text{Balanced steady-state} \\ & 0 \leq v & \text{Irreversible reactions} \\ & v_1 \leq 10 & \text{Uptake rate is } 10\frac{mmol}{hour} \\ & v_3 \leq 0 & \text{Genetic perturbation} \\ & v_5 \leq 0 & \text{An-}E\text{-robic environmental condition} \end{array}$$

In [6]:
```
abc_model = cobra.io.load_json_model('ABC/ABC_model.json')  # Stoichiome
tric matrix loaded
abc_model.objective = R_8                                    # Cellular o
bjective (growth)
for rxn in abc_model.reactions:
    rxn.lower_bound = 0                                      # Irreversib
le reactions
abc_model.reactions.R_1.upper_bound = 10                     # Uptake rat
e is 10 mmol/hour
abc_model.reactions.R_3.upper_bound = 0                      # Genetic pe
rturbation
abc_model.reactions.R_5.upper_bound = 0                      # An-E-robic
environmental condition

environment_and_ko_solution = abc_model.optimize()           # Find fluxe
s that balance steady-state

display(environment_and_ko_solution)
escher.Builder( map_json       ='ABC/ABC_map.json',
                model          = abc_model,
                reaction_data  = environment_and_ko_solution.fluxes.to_d
ict(),
                reaction_scale = reaction_scale
              )
```

Optimal **solution with objective value 5.000**

	fluxes	reduced_costs
R_1	10.0	1.0
R_2	10.0	0.0
R_3	0.0	1.0
R_4	0.0	0.0
R_5	0.0	3.0
R_6	5.0	0.0
R_7	5.0	0.0
R_8	5.0	0.0
R_9	5.0	0.0


E_ext
R_5 0.00
E
B
R_4 0.00
R_2 10.0
R_1 10.0
A_ext
A
R_6 5.00
R_8 5.00
D
D_ext
R_7 5.00
R_3 0.00
C
F
R_9 5.00
F_ext

How can we model the tradeoff between growth rate and bioproduct yield?

By knocking out R_3 and restricting ourselves to an-E-robic conditions, we are able to maximize bioproduct yield, but growth rate is pretty low. Perhaps we can improve E-robic conditions and see how that affects the trade-off between growth rate and bioproduct yield.

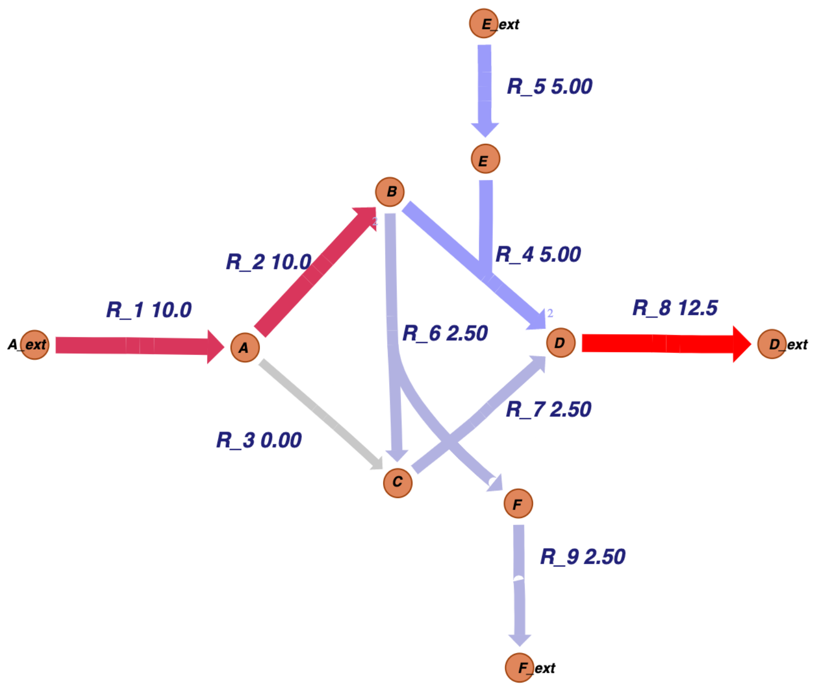
Modeling the effect of the gene perturbation on growth rate and bioproduct yield under low E-robic conditions

$$\begin{array}{lll}
\underset{\vec{\mathbf{v}}}{\text{maximize}} & v_9 & \text{Engineering objective (bioproduct)} \\
\text{subject to} & S \cdot v = 0 & \text{Balanced steady-state} \\
 & 0 \leq v & \text{Irreversible reactions} \\
 & v_1 \leq 10 & \text{Uptake rate is } 10\ \frac{mmol}{hour} \\
 & v_3 \leq 0 & \text{Genetic perturbation} \\
 & v_5 \leq 5 & \text{Low } E\text{-robic environmental condition}
\end{array}$$

In [7]:
```
abc_model = cobra.io.load_json_model('ABC/ABC_model.json')  # Stoichiome
tric matrix loaded
abc_model.objective = R_8                                    # Cellular o
bjective (growth)
for rxn in abc_model.reactions:
    rxn.lower_bound = 0                                      # Irreversib
le reactions
abc_model.reactions.R_1.upper_bound = 10                     # Uptake rat
e is 10 mmol/hour
abc_model.reactions.R_3.upper_bound = 0                      # Genetic pe
rturbation
abc_model.reactions.R_5.upper_bound = 5                      # An-E-robic
environmental condition

environment_solution = abc_model.optimize()                  # Find fluxe
s that balance steady-state
display(environment_solution)

escher.Builder( map_json       ='ABC/ABC_map.json',
                model          = abc_model,
                reaction_data  = environment_solution.fluxes.to_dict(),
                reaction_scale = reaction_scale,
              )
```

***Optimal* solution with objective value 12.500**

	fluxes	reduced_costs
R_1	10.0	1.0
R_2	10.0	0.0
R_3	0.0	1.0
R_4	5.0	0.0
R_5	5.0	3.0
R_6	2.5	0.0
R_7	2.5	0.0
R_8	12.5	0.0
R_9	2.5	0.0


E_ext
R_5 5.00
E
B
R_4 5.00
R_2 10.0
R_1 10.0
A_ext
A
R_6 2.50
D
R_8 12.5
D_ext
R_7 2.50
R_3 0.00
C
F
R_9 2.50
F_ext

![

Visualizing the tradeoff between growth rate and bioproduct yield with the Production envelope

We have discussed the tradeoff between growth rate and bioproduct yield at an intuitive level, but we would like to extend our intuitions to genome-scale models. One valuable way to visualize this tradeoff is with the production envelope.

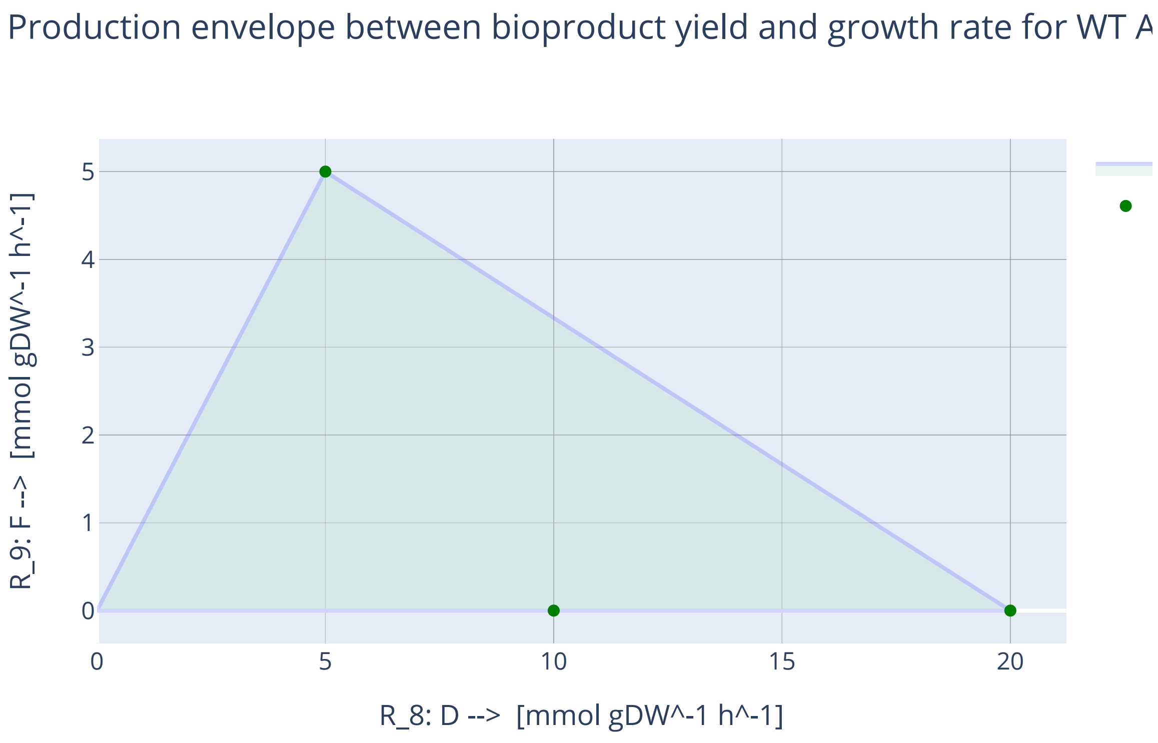
Let's generate a production envelope for the original unperturbed ABC model

Production envelope for wild-type ABC model

For $i = 0..20$

$$\begin{array}{lll} \underset{\vec{\mathbf{v}}}{\text{maximize/minimize}} & v_9 & \text{Engineering objective (bioproduct)} \\ \text{subject to} & S \cdot v = 0 & \text{Balanced steady-state} \\ & 0 \leq v & \text{Irreversible reactions} \\ & v_1 \leq 10 & \text{Uptake rate is } 10\ \frac{mmol}{hour} \\ & v_8 = i & \text{Growth rate is } i \frac{mmol}{gDW \cdot hour} \end{array}$$

In [14]:
```
from cameo.flux_analysis import phenotypic_phase_plane as ppp
from cameo.visualization.plotting.with_plotly import PlotlyPlotter
from cameo.visualization import plotting

abc_model = cobra.io.load_json_model('ABC/ABC_model.json')  # Stoichiome
tric matrix loaded
for rxn in abc_model.reactions:
    rxn.lower_bound = 0                                      # Irreversib
le reactions
abc_model.reactions.R_1.upper_bound = 10                     # Uptake rat
e is 10 mmol/hour

production_envelope = ppp( abc_model,
                              variables=[abc_model.reactions.R_8],  # Gr
owth rate <= i
                              objective=abc_model.reactions.R_9,
                              points=21)    # Engineering objective (bio
product)

result_df = production_envelope.data_frame.rename( columns  = dict(
                                                             R_8 = 'growt
h_rate',
                                                objective_upper_bound = 'biopr
oduct_maximum',
                                                objective_lower_bound = 'biopr
oduct_minimum'))
plotter = PlotlyPlotter()
production_envelope.plot(plotter,
                         title='Production envelope between bioproduct y
ield and growth rate for WT ABC model',
                         points=[(5,5), (20,0),(10,0)])
```



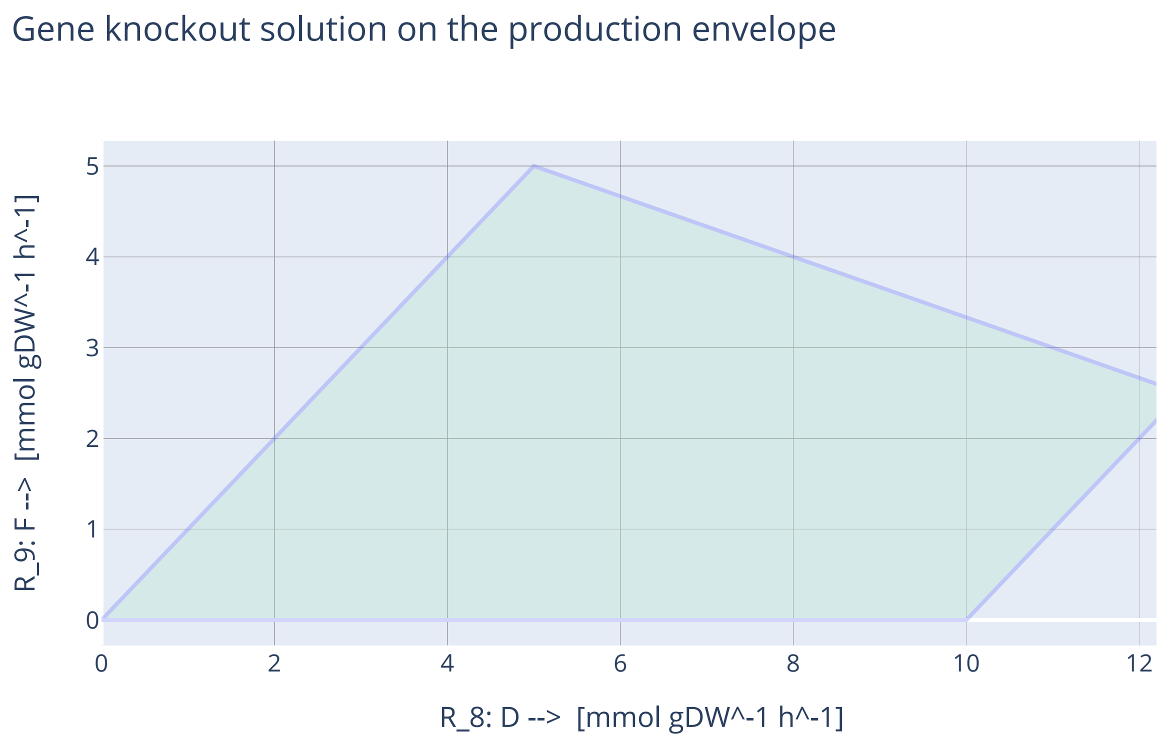
Production envelope for genetic perturbation and environmental condition

As we can see there is an interesting tradeoff between growth rate and bioproduct yield for the original unperturbed ABC model. We see that growth is required for bioproduct formation, but bioproduct formation is not required for growth. Also note that this production envelope indicates that the optimal bioproduct yield is not evolutionarily stable. A faster-growing mutant can outcompete our bioproduct-producing strain. How does the production envelope for our engineered strain under low E-robic conditions look?

For $i = 0..20$

$$\begin{array}{lll}
\underset{\vec{\mathbf{v}}}{\text{maximize/minimize}} & v_9 & \text{Engineering objective (bioproduct)} \\
\text{subject to} & S \cdot v = 0 & \text{Balanced steady-state} \\
 & 0 \leq v & \text{Irreversible reactions} \\
 & v_1 \leq 10 & \text{Uptake rate is } 10 \frac{mmol}{hour} \\
 & v_3 \leq 0 & \text{Genetic perturbation} \\
 & v_5 \leq 5 & \text{Low } E\text{-robic environmental condition} \\
 & v_8 = i & \text{Growth rate is } i \frac{mmol}{gDW/hour}
\end{array}$$

In [15]:
```
from cameo.flux_analysis import phenotypic_phase_plane as production_env
elope

abc_model = cobra.io.load_json_model('ABC/ABC_model.json')  # Stoichiome
tric matrix loaded
for rxn in abc_model.reactions:
    rxn.lower_bound = 0                                      # Irreversib
le reactions
abc_model.reactions.R_1.upper_bound = 10                     # Uptake rat
e is 10 mmol/hour
abc_model.reactions.R_3.upper_bound = 0                      # Genetic pe
rturbation
abc_model.reactions.R_5.upper_bound = 5                      # Low E-robi
c environmental condition

result = production_envelope( abc_model,
                              variables=[abc_model.reactions.R_8],  # Gr
owth rate is i mmol/gDW/hour
                              objective=abc_model.reactions.R_9,    # En
gineering objective (bioproduct)
                              points=21 )

result_df = result.data_frame.rename(columns=dict(R_8='growth_rate',
                                objective_upper_bound='bioproduct_maximu
m',
                                objective_lower_bound='bioproduct_minimu
m'))
result.plot(plotter,title='Gene knockout solution on the production enve
lope')
```



Using Flux variability analysis to study the steady-state behavior

A number of cell factory design algorithms use flux variablity analysis (FVA) as a pre-processing step, so it is worth taking some time to understand the problem it is designed to solve, how it works, and what you can do with it.

What does flux variablity analysis do?

Although each reaction has explicit lower and upper bounds, sometimes the constraint on one reaction imposes implicit constraints on other reactions. Flux variability analysis is an estimate of these implicit bounds. Why do I say an estimate? Because the implicit bounds may actually be tighter than what flux variability analysis predicts. This can be seen in the figure from [Mahadevan 2003]

FVA

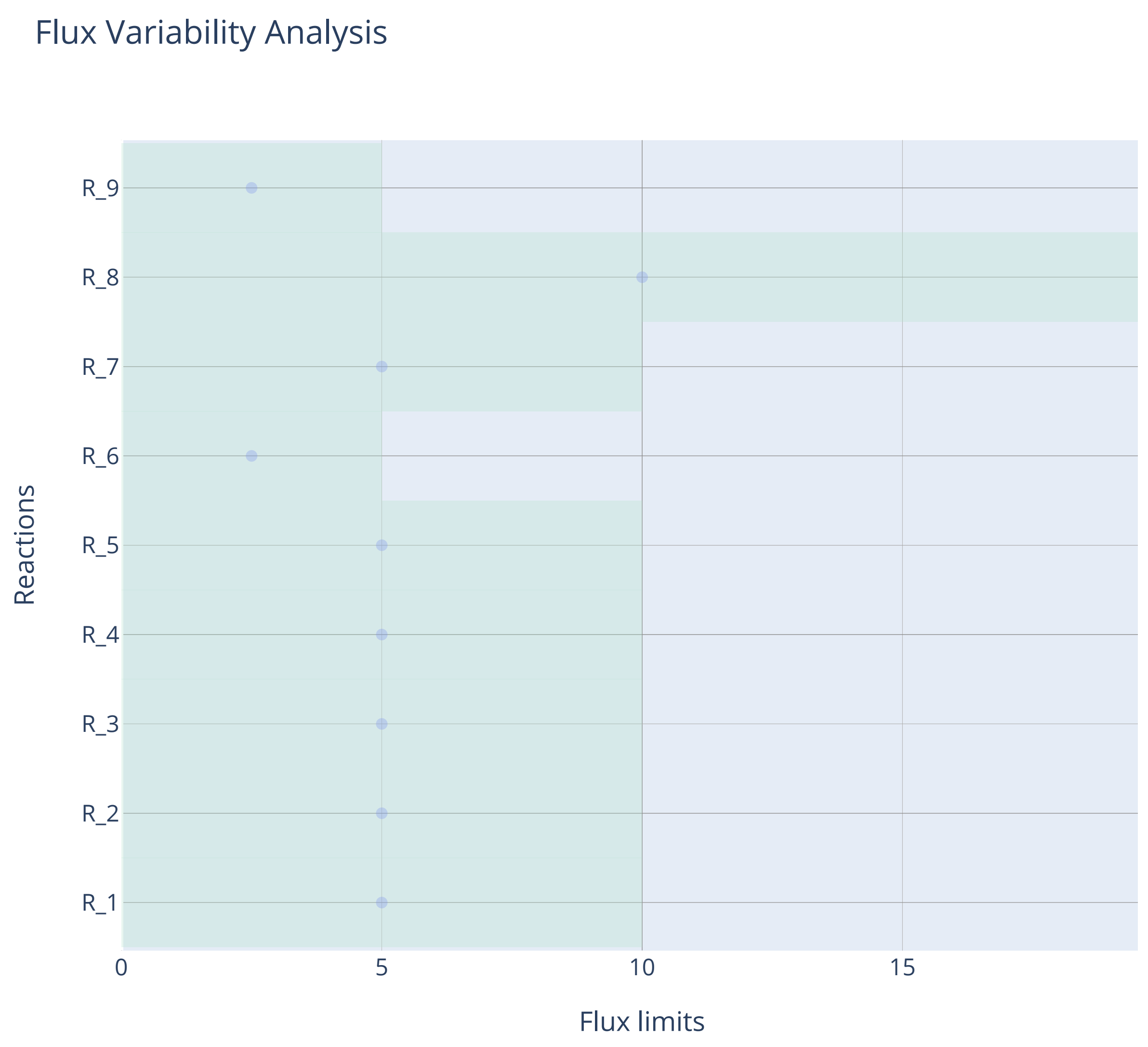
As you can see, when there are only two fluxes, FVA forms the tightest rectangle around the actual solution, which is a polygon. FVA forms a parallelopiped around the actual solution, which is a polyhedra in 3 dimensions, and in general, FVA forms a hyperrectangle around the polytope in n dimensions. The work required to find an exact solution, unfortunately, grows exponentially in the number of reactions, but for genome-scale models, FVA is usually good enough.

In [16]:
```
import escher
import cobra
from cameo import flux_variability_analysis

abc_model =  cobra.io.load_json_model('ABC/ABC_model.json')
abc_model.reactions.R_1.upper_bound=10
fva_result = flux_variability_analysis(abc_model)

#abc_fva = fva(abc_model)

fva_result.data_frame
```

Out[16]:

	lower_bound	upper_bound
R_1	0.0	10.0
R_2	0.0	10.0
R_3	0.0	10.0
R_4	0.0	10.0
R_5	0.0	10.0
R_6	0.0	5.0
R_7	0.0	10.0
R_8	0.0	20.0
R_9	0.0	5.0

In [17]:
```
fva_result.plot(plotter,index=result.data_frame.index, height=600)
```



How does FVA work?

Conceptually, FVA works by looping through each rection in the network, and solving for the maximum flux and then solving for the minimum flux associated with that reaction.

$$\text{for } i \text{ in } 1..n$$

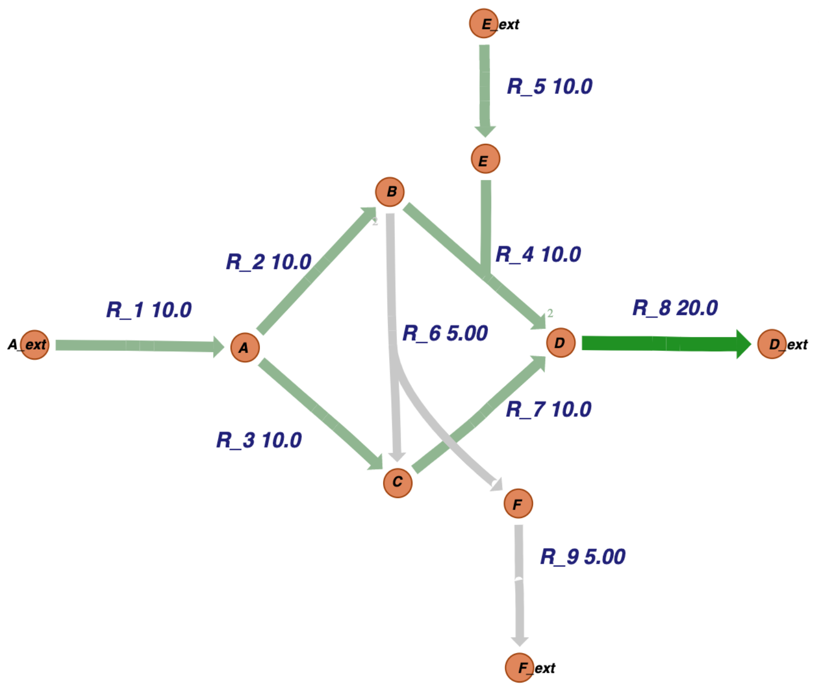
$$\begin{aligned}
\underset{\vec{v}}{\text{minimize}} \quad & v_i \\
\text{subject to} \quad & S \cdot \vec{v} = 0 \\
& 0 \leq \vec{v} \\
& v_1 \leq 10 \\
\underset{\vec{v}}{\text{maximize}} \quad & v_i \\
\text{subject to} \quad & S \cdot \vec{v} = 0 \\
& 0 \leq \vec{v} \\
& v_1 \leq 10
\end{aligned}$$

In [18]:
```
import pandas as pd

def fva( model ):
    fva = {'minimum':{},
           'maximum':{}}
    for reaction in model.reactions:
        with model:
            model.objective = {reaction: -1}
            fva['minimum'][reaction.id] = model.slim_optimize()
        with model:
            model.objective = {reaction: 1}
            fva['maximum'][reaction.id] =  model.slim_optimize()
    return pd.DataFrame(fva)[['minimum','maximum']]
```

What can you do with flux variability analysis?

In [19]:
```python
import escher
import cobra
from cameo import flux_variability_analysis

abc_model =  cobra.io.load_json_model('ABC/ABC_model.json')
abc_model.reactions.R_1.upper_bound=10

abc_fva = fva(abc_model)
display(abc_fva)
escher.Builder(map_json='ABC/ABC_map.json',
               model=abc_model,
               reaction_data=abc_fva['maximum'].to_dict(),
              )
```

	minimum	maximum
R_1	0.0	10.0
R_2	0.0	10.0
R_3	0.0	10.0
R_4	0.0	10.0
R_5	0.0	10.0
R_6	0.0	5.0
R_7	0.0	10.0
R_8	0.0	20.0
R_9	0.0	5.0


E_ext
R_5 10.0
E
B
R_2 10.0
R_4 10.0
R_1 10.0
R_8 20.0
A_ext
A
R_6 5.00
D
D_ext
R_7 10.0
R_3 10.0
C
F
R_9 5.00
F_ext