TPFBA_ex1

June 13, 2023

[1]:
```
#pip install cobra
```

[2]:
```
#pip install escher
```

[3]:
```
import cobra
import escher
```

[4]:
```
# Loading a model
model_path = 'toy_model.xml'
model = cobra.io.read_sbml_model(model_path)

model
```

```
Adding exchange reaction EX_S with default bounds for boundary metabolite: S.
Adding exchange reaction EX_biomass with default bounds for boundary metabolite:
biomass.
Adding exchange reaction EX_fuel1 with default bounds for boundary metabolite:
fuel1.
Adding exchange reaction EX_fuel2 with default bounds for boundary metabolite:
fuel2.
```

[4]:
```
<Model toy_model_xml at 0x7fea90b94fd0>
```

[5]:
```
environmental_conditions = {
    'EX_S': (-100, 0),
    'EX_biomass': (0, 1000),
    'EX_fuel1': (0, 1000),
    'EX_fuel2': (0, 1000),

}
environmental_conditions
```

[5]:
```
{'EX_S': (-100, 0),
 'EX_biomass': (0, 1000),
 'EX_fuel1': (0, 1000),
 'EX_fuel2': (0, 1000)}
```

[6]:
```
for reaction_id, bound in environmental_conditions.items():
    model.reactions.get_by_id(reaction_id).bounds = bound
```

[7]:
```
model.objective='EX_biomass'
```

[8]:
```
model.objective.expression
```

[8]:
$$1.0 \cdot EX_{biomass} - 1.0 \cdot EX_{biomassreverse91cf2}$$

[9]:
```
for i in range(0,len(model.reactions)):
    print(model.reactions[i].id,' : ', model.reactions[i].reaction)
```

```
EX_S  :  S <--
EX_biomass  :  biomass -->
EX_fuel1  :  fuel1 -->
EX_fuel2  :  fuel2 -->
R01  :  S --> M1
R02  :  M1 <=> M2
R03  :  M2 --> M3
R04  :  M3 <=> M4
R05  :  M4 <=> M5
R06  :  M3 --> M6
R07  :  M6 --> M7
R08  :  M7 --> 0.5 M5 + 0.5 M8
R09  :  M8 --> fuel1
R10  :  M1 <=> M9
R11  :  M9 --> M2
R12  :  M9 --> M10
R13  :  M9 --> M11
R14  :  M11 --> 0.7 M12 + 0.3 M13
R15  :  M13 --> fuel2
R16  :  M12 --> M5
R17  :  M5 --> biomass
R18  :  M3 --> 2.0 M8
```

[10]:
```
solution = model.optimize()
solution.fluxes
```

[10]:
```
EX_S         -100.0
EX_biomass    100.0
EX_fuel1       -0.0
EX_fuel2       -0.0
R01           100.0
R02           100.0
R03           100.0
R04           100.0
R05           100.0
```

```
R06             0.0
R07             0.0
R08             0.0
R09             0.0
R10             0.0
R11             0.0
R12             0.0
R13             0.0
R14             0.0
R15             0.0
R16             0.0
R17           100.0
R18             0.0
Name: fluxes, dtype: float64
```

[ ]:

[11]:
```
builder = escher.Builder(map_json='toy_escher_map.json', model=model,␣
↪reaction_data=solution.fluxes)
#builder.save_html('map2')
#from IPython.core.display import display, HTML
#display(HTML('map2'))
builder
```

```
Builder(reaction_data={'EX_S': -100.0, 'EX_biomass': 100.0, 'EX_fuel1': -0.0,␣
↪'EX_fuel2': -0.0, 'R01': 100.0, …
```

[12]:
```
model.objective='EX_fuel1'
solution = model.optimize()
solution.fluxes
```

[12]:
```
EX_S         -100.0
EX_biomass     -0.0
EX_fuel1      200.0
EX_fuel2       -0.0
R01           100.0
R02           100.0
R03           100.0
R04             0.0
R05             0.0
R06             0.0
R07             0.0
R08             0.0
R09           200.0
R10             0.0
R11             0.0
R12             0.0
```

```
R13             0.0
R14             0.0
R15             0.0
R16             0.0
R17             0.0
R18           100.0
Name: fluxes, dtype: float64
```

[13]:
```
builder = escher.Builder(map_json='toy_escher_map.json', model=model,␣
 ↪reaction_data=solution.fluxes)
#builder.save_html('map2')
#from IPython.core.display import display, HTML
#display(HTML('map2'))
builder
```

```
Builder(reaction_data={'EX_S': -100.0, 'EX_biomass': -0.0, 'EX_fuel1': 200.0,␣
 ↪'EX_fuel2': -0.0, 'R01': 100.0, …
```

[14]:
```
model.objective='EX_biomass'
solution = model.optimize()
solution.fluxes
```

[14]:
```
EX_S         -100.0
EX_biomass    100.0
EX_fuel1       -0.0
EX_fuel2       -0.0
R01           100.0
R02           100.0
R03           100.0
R04           100.0
R05           100.0
R06             0.0
R07             0.0
R08             0.0
R09             0.0
R10             0.0
R11             0.0
R12             0.0
R13             0.0
R14             0.0
R15             0.0
R16             0.0
R17           100.0
R18             0.0
Name: fluxes, dtype: float64
```

[15]:
```
from cobra.flux_analysis import flux_variability_analysis
flux_variability_analysis(model)
```

[15]:

	minimum	maximum
EX_S	-100.0	-100.0
EX_biomass	100.0	100.0
EX_fuel1	0.0	0.0
EX_fuel2	0.0	0.0
R01	100.0	100.0
R02	-900.0	100.0
R03	100.0	100.0
R04	100.0	100.0
R05	100.0	100.0
R06	0.0	0.0
R07	0.0	0.0
R08	0.0	0.0
R09	0.0	0.0
R10	0.0	1000.0
R11	0.0	1000.0
R12	0.0	0.0
R13	0.0	0.0
R14	0.0	0.0
R15	0.0	0.0
R16	0.0	0.0
R17	100.0	100.0
R18	0.0	0.0

[16]:
```
environmental_conditions = {
    'R11': (1000, 1000)
}
for reaction_id, bound in environmental_conditions.items():
    model.reactions.get_by_id(reaction_id).bounds = bound
```

[17]:
```
solution = model.optimize()
solution.fluxes
```

[17]:

EX_S	-100.0
EX_biomass	100.0
EX_fuel1	-0.0
EX_fuel2	-0.0
R01	100.0
R02	-900.0
R03	100.0
R04	100.0
R05	100.0
R06	0.0
R07	0.0

```
R08              0.0
R09              0.0
R10           1000.0
R11           1000.0
R12              0.0
R13              0.0
R14              0.0
R15              0.0
R16              0.0
R17            100.0
R18              0.0
Name: fluxes, dtype: float64
```

[18]:
```
builder = escher.Builder(map_json='toy_escher_map.json', model=model,␣
 ↪reaction_data=solution.fluxes)
#builder.save_html('map2')
#from IPython.core.display import display, HTML
#display(HTML('map2'))
builder
```

```
Builder(reaction_data={'EX_S': -100.0, 'EX_biomass': 100.0, 'EX_fuel1': -0.0,␣
 ↪'EX_fuel2': -0.0, 'R01': 100.0, …
```

[28]:
```
model.objective={model.reactions.EX_fuel1:1}
solution = model.optimize()
model.summary()
```

[28]:
```
<cobra.summary.model_summary.ModelSummary at 0x7feae92d2880>
```

[29]:
```
solution.fluxes
```

[29]:
```
EX_S         -100.0
EX_biomass     -0.0
EX_fuel1      200.0
EX_fuel2       -0.0
R01           100.0
R02           100.0
R03           100.0
R04             0.0
R05             0.0
R06             0.0
R07             0.0
R08             0.0
R09           200.0
R10             0.0
R11             0.0
R12             0.0
```

```
R13             0.0
R14             0.0
R15             0.0
R16             0.0
R17             0.0
R18           100.0
Name: fluxes, dtype: float64
```

[30]:
```
model.objective={model.reactions.EX_biomass:1}
```

[31]:
```
environmental_conditions = {
    'R11': (0, 1000),
    'R04': (0,0),
    'R10': (0,0),
}
for reaction_id, bound in environmental_conditions.items():
    model.reactions.get_by_id(reaction_id).bounds = bound
```

[32]:
```
solution = model.optimize()
solution.fluxes
```

[32]:
```
EX_S         -100.0
EX_biomass     50.0
EX_fuel1       50.0
EX_fuel2       -0.0
R01           100.0
R02           100.0
R03           100.0
R04             0.0
R05             0.0
R06           100.0
R07           100.0
R08           100.0
R09            50.0
R10             0.0
R11             0.0
R12             0.0
R13             0.0
R14             0.0
R15             0.0
R16             0.0
R17            50.0
R18             0.0
Name: fluxes, dtype: float64
```

[33]:
```
builder = escher.Builder(map_json='toy_escher_map.json', model=model,␣
 ↪reaction_data=solution.fluxes)
```

```
#builder.save_html('map2')
#from IPython.core.display import display, HTML
#display(HTML('map2'))
builder
```

```
Builder(reaction_data={'EX_S': -100.0, 'EX_biomass': 50.0, 'EX_fuel1': 50.0,␣
↪'EX_fuel2': -0.0, 'R01': 100.0, '…
```

[ ]:

[ ]:

[ ]:
```
# Dessiner une map escher
'''
cobra.io.save_json_model(model,'toy_model.json')
from escher import Builder
b=Builder()
b.model_json='toy_model.json'
b.height=1600
b.width=1600
b.save_html('map')
from IPython.core.display import display, HTML
display(HTML('map'))
'''
```

[ ]:

[ ]: