**Designing stoichiometric conversion (optStoic) and *de novo* enzyme-catalyzed pathways (novoStoic) for desired substrate/product transformations**

**Installation**

In order to use opStoic/novoStoic codes (in the folder “core”) to identify potential pathways for desired substrate/product conversions, we first need to install the following software and python packages:

* Python Anaconda == 2.7
* numpy == 1.10.0
* pandas == 0.20.0
* rdkit == 2018.09.1
* pulp == 1.6.8
* networkx == 2.1
* sortedcontainers == 1.5.10
* Pillow == 5.1.0
* matplotlib == 2.2.2
* graphviz == 0.10.1
* xlrd == 1.1.0
* Optimization solver CPLEX == 12.8.0.0 (or guborbi == 7.5.2)

**Installing on Windows:**

1. Install Anaconda
   1. Download Python 2.7 version 32-Bit (64 bit if running on 64-bit OS) graphical installer from <https://www.anaconda.com/distribution/#windows>, open installer file
   2. Double-click on installation file, follow installation instructions
   3. Under Advanced Installation Options, check the following box:
      1. Add Anaconda to my PATH environment
   4. Add anaconda scripts folder to path:
      1. Control Panel -> System and Security -> System -> Advanced system settings -> Environment variables -> User variables for ‘user’ -> Select Path, click ‘edit’
      2. Click new
      3. Add Anaconda Scripts folder location (e.g. C:\ProgramData\Anaconda2\Scripts\)
      4. Open new Command Prompt, test conda install (command line: conda --version)
2. Create a new conda environment with rdkit and compatible versions of pandas and numpy (command line: conda create -c rdkit -n my-rdkit-env python=2.7 numpy=1.10 pandas=0.20 rdkit)
3. Install pulp (command line: pip install pulp)
4. Install networkx (command line: pip install network)
5. Install sortedcontainers (command line: pip install sortedcontainers)
6. Install matplotlib (command line: pip install matplotlib)
7. Install PIL (command line: pip install Pillow)
8. Install graphviz (command line: pip install graphviz)
9. Install xlrd (command line: pip install xlrd)
10. Install gurobi (optional, must acquire gurobi license, gurobi quick start guide: https://www.gurobi.com/documentation/8.1/quickstart\_windows.pdf):
    1. Command line: conda config --add channels http://conda.anaconda.org/gurobi
    2. Command line: conda install gurobi
11. Install cplex (optional, must acquire licence from IBM, requires 64 bit windows)
    1. download and run installer from IBM
    2. in command prompt, navigate to CPLEX\Python folder (e.g C:\ProgramFiles\CPLEX\Python)
    3. run steup.py (command line: python setup.py)

**Navigating to working directory:**

1. Unzip PathwayDesignToolbox.zip and move the folder to a working direction (e.g., C:\Users\user\_id\Desktop\PathwayDesignToolbox)
2. From command prompt, change to PathwayDesignToolbox working directory (i.e., cd C:\Users\user\_id\Desktop\PathwayDesignToolbox)

**Examples:**

1. *1,4-butanediol synthesis from succinyl-CoA*
   * 1. **Component contribution: Calculating Gibbs free energy of formation for all metabolites participating in 1,4-butanediol synthesis** 
        + use dG\_calculator instructions and codes to calculate Gibbs free energy of formations
     2. **Activate rdkit Anaconda environment (command line: activate my-rdkit-env)**
     3. **optStoic: Designing stoichiometric conversion of succinyl-CoA to 1,4-butanediol**

Directions for running code:

* Save’dG\_output.csv’ file from dG\_calculator folder as ‘bdo\_dG\_output.csv’ in ‘PathwayDesignToolkit’ folder
* Open ‘calculate\_stoic.py’
  1. Change line 4 to name = ‘bdo’
  2. Save calculate\_stoic.py
* From command prompt run ‘calculate\_stoic.py (python calculate\_stoic.py)

**Data Input**:

1. bdo\_optstoic\_input.xlsx
   1. reactant\_stoichs(sheet): Column 1: metabolite IDs of reactants, Column 2: stoichiometry of reactants in desired conversion. Value of primary substrate (succinyl-CoA, MNX92) is set to -1, all other stoichiometries are set to None and optimized by optStoic.
   2. product\_stoichs (sheet): Column 1: metabolite IDs of products, Column 2: stoichiometry of reactants in desired conversion. Value of all products set to ‘None’ and optimized by optStoic.
   3. Objective (sheet): Primary product (1,4-butanediol, ChEBI\_41189), value to be maximized
   4. missingMet (sheet): Column 1: Metabolite ID for any reactants or products in problem missing from database, Column 2: InChI strings corresponding to metabolites missing from database, Column 3: dG of formation corresponding to metabolites missing from database (taken from bdo\_dG\_output.csv)

**Data Output**:

* + - 1. bdo\_optstoic\_output.csv:

1. Column 1: metabolite ID of all metabolites participating in conversion, Column 2: stoichiometric coefficients of metabolites in conversion
   * 1. **novoStoic: Designing a pathway for conversion of succinyl-CoA to 1,4-butanediol**

Directions for running code:

* + - * Open ‘run\_novoStoic.py’
        1. Change line 17 to name = ‘bdo’
        2. Save run\_novoStoic.py
      * From command prompt run ‘run\_novoStoic.py (python run\_novoStoic.py)

**Data Input**:

1. bdo\_input.xlsx: column 1:
   * + - 1. Project: name of project (defalult name is ‘test’)
         2. Substrates: column 1: list of substrate metabolite ID, column 2: corresponding stoichiometry in transformation
         3. Products: column 1: list of product metabolite IDs, column 2: corresponding stoichiometry in transformation
         4. novel\_mets: column 1: list of metabolites participating in transformation not included in database, column 2: InChI strings corresponding to novel metabolites, Column 2: Smarts string corresponding to novel metabolites
         5. primary\_substrate: primary substrate in transformation
         6. primary\_product: primary product in transformation
         7. product\_name: name of product used in transformation (14bdo)
         8. vard\_data:

iterations: number of solutions for novoStoic to generate

distance: size of molecular signature

nRule: maximum number of transformations to consider in desired pathway

**Data Output**:

* + - * 1. solution\_14bdo\_XX\_YY\_ZZ.csv (XX = number of biotransformations, YY = number of rules, ZZ = solution index): column 1: list of metabolite ID, column 2: list of standard Gibbs free energy of formation for each metabolite (in kJ/mol)
        2. solution\_14bdo\_XX\_YY\_ZZ.png (XX = number of biotransformations, YY = number of rules, ZZ = solution index): image linking source metabolite to intermediate metabolites to product metabolite
  1. *styrene synthesis from 3-phenylpyruvate*

1. **Component contribution: Calculating Gibbs free energy of formation for all metabolites participating in styrene synthesis**
   * + - use dG\_calculator instructions and codes to calculate Gibbs free energy of formations
2. **Activate rdkit Anaconda environment (command line: activate my-rdkit-env)**
3. **optStoic: Designing stoichiometric conversion of 3-phenylpyruvate to styrene**

Directions for running code:

* Save’dG\_output.csv’ file from dG\_calculator folder as ‘styrene\_dG\_output.csv’ in ‘PathwayDesignToolkit’ folder
* Open ‘calculate\_stoic.py’
  1. Change line 4 to name = ‘styrene’
  2. Save calculate\_stoic.py
* From command prompt run ‘calculate\_stoic.py (python calculate\_stoic.py)

**Data Input**:

1. styrene\_optstoic\_input.xlsx
   1. reactant\_stoichs(sheet): Column 1: metabolite IDs of reactants, Column 2: stoichiometry of reactants in desired conversion. Value of primary substrate (3-phenylpyruvate, MNX210) is set to -1, all other stoichiometries are set to None and optimized by optStoic.
   2. product stoichs (sheet): Column 1: metabolite IDs of products, Column 2: stoichiometry of reactants in desired conversion. Value of MNX13 (CO2) is set to 1 to ensure carbon balance. Value of all other products set to None and optimized by optStoic.
   3. Objective (sheet): Primary product (styrene, MNXM2434), value to be maximized
   4. missingMet (sheet): Column 1: Metabolite ID for any reactants or products in problem missing from database, Column 2: InChI strings corresponding to metabolites missing from database, Column 3: dG of formation corresponding to metabolites missing from database (taken from styrene\_dG\_output.csv), empty for this scenario, all metabolites are included in database

**Data Output**:

* + - 1. styrene\_optstoic\_output.csv:
  1. Column 1: metabolite ID of all metabolites participating in conversion Column 2: stoichiometric coefficients of metabolites in conversion

1. **novoStoic: Designing a pathway for conversion of 3-phenylpyruvate to styrene**
   * + - Open ‘run\_novoStoic.py’
         1. Change line 17 to name = ‘styrene’
         2. Save run\_novoStoic.py
       - From command prompt run ‘run\_novoStoic.py (python run\_novoStoic.py)

**Data Input**:

1. styrene\_input.xlsx:
   * + - 1. Project: name of project (defalult name is ‘test’)
         2. Substrates: column 1: list of substrate metabolite ID, column 2: corresponding stoichiometry in transformation
         3. Products: column 1: list of product metabolite IDs, column 2: corresponding stoichiometry in transformation
         4. novel\_mets: column 1: list of metabolites participating in transformation not included in database, column 2: InChI strings corresponding to novel metabolites, Column 2: Smarts string corresponding to novel metabolites
         5. primary\_substrate: primary substrate in transformation
         6. primary\_product: primary product in transformation
         7. product\_name: name of product used in transformation (14bdo)
         8. vard\_data:

iterations: number of solutions for novoStoic to generate

distance: size of molecular signature

nRule: maximum number of transformations to consider in desired pathway

**Data Output**:

1. solution\_styrene\_XX\_YY\_ZZ.csv (XX = number of biotransformations, YY = number of rules, ZZ = solution index): column 1: list of metabolite ID, column 2: list of standard Gibbs free energy of formation for each metabolite (in kJ/mol)
2. solution\_styrene\_XX\_YY\_ZZ.png (XX = number of biotransformations, YY = number of rules, ZZ = solution index): image linking source metabolite to intermediate metabolites to product metabolite
3. *xylitol synthesis from D-xylose*
4. **Component contribution: Calculating Gibbs free energy of formation for all metabolites participating in xylitol synthesis**
   * + - use dG\_calculator instructions and codes to calculate Gibbs free energy of formations
5. **Activate rdkit Anaconda environment (command line: activate my-rdkit-env)**
6. **optStoic: Designing stoichiometric conversion of D-xylose to xylitol**

Directions for running code:

* Save’dG\_output.csv’ file from dG\_calculator folder as ‘styrene\_dG\_output.csv’ in ‘PathwayDesignToolkit’ folder

1. Open ‘calculate\_stoic.py’
   1. Change line 4 to name = ‘xylitol’
   2. Save calculate\_stoic.py
2. From command prompt run ‘calculate\_stoic.py (python calculate\_dG.py)

**Data Input**:

1. xylitol\_optstoic\_input.xlsx
   1. reactant\_stoichs(sheet): Column 1: metabolite IDs of reactants, Column 2: stoichiometry of reactants in desired conversion. Value of primary substrate (D-xylose, MNX348) is set to -1, all other stoichiometries are set to None and optimized by optStoic.
   2. product\_stoichs (sheet): Column 1: metabolite IDs of products, Column 2: stoichiometry of reactants in desired conversion. Value of all products set to None and optimized by optStoic.
   3. Objective (sheet): Primary product (xylitol, MXNM510), value to be maximized
   4. missingMet (sheet): Column 1: Metabolite ID for any reactants or products in problem missing from database, Column 2: InChI strings corresponding to metabolites missing from database, Column 3: dG of formation corresponding to metabolites missing from database (taken from xylitol\_dG\_output.csv) , empty for this scenario, all metabolites are included in database

**Data Output**: xylitol\_optstoic\_output.csv: Column 1: metabolite ID of all metabolites participating in conversion Column 2: stoichiometric coefficients of metabolites in conversion

1. **novoStoic: Designing a pathway for conversion of D-xylose to xylitol**
   * + - Open ‘run\_novoStoic.py’
         1. Change line 17 to name = ‘xylitol’
         2. Save run\_novoStoic.py
       - From command prompt run ‘run\_novoStoic.py (python run\_novoStoic.py)

**Data Input**:

1. styrene\_input.xlsx:
   * + - 1. Project: name of project (defalult name is ‘test’)
         2. Substrates: column 1: list of substrate metabolite ID, column 2: corresponding stoichiometry in transformation
         3. Products: column 1: list of product metabolite IDs, column 2: corresponding stoichiometry in transformation
         4. novel\_mets: column 1: list of metabolites participating in transformation not included in database, column 2: InChI strings corresponding to novel metabolites, Column 2: Smarts string corresponding to novel metabolites
         5. primary\_substrate: primary substrate in transformation
         6. primary\_product: primary product in transformation
         7. product\_name: name of product used in transformation (14bdo)
         8. vard\_data:

iterations: number of solutions for novoStoic to generate

distance: size of molecular signature

nRule: maximum number of transformations to consider in desired pathway

**Data Output**:

1. solution\_xylitol\_XX\_YY\_ZZ.csv (XX = number of biotransformations, YY = number of rules, ZZ = solution index): column 1: list of metabolite ID, column 2: list of standard Gibbs free energy of formation for each metabolite (in kJ/mol)
2. solution\_xylitol\_XX\_YY\_ZZ.png (XX = number of biotransformations, YY = number of rules, ZZ = solution index): image linking source metabolite to intermediate metabolites to product metabolite