Using Cello

Verilog.html

1. Entering Verilog code: Click Choose and select the type of Verilog code.
    a. Choosing an option such as 'case' or 'assign' from the select menu will automatically build skeleton Verilog code as a starting point, based on the number of inputs/outputs.
2. Inputs/Outputs:
    a. Inputs are promoters
    b. Outputs are the actuator DNA sequences (including RBS if it is a gene)
    c. Creating a new Input/Output:
        i. Click on choose
        ii. Select the option **(new)** from the dropdown menu.
        iii. Enter the **name**, **low REU**, **high REU**, and **DNA sequence** for a new Input. Enter the **name** and **DNA sequence** for a new Output.
        iv. Click **add**
    d. Deleting an Input/Output:
        i. Click on choose
        ii. Select the option **(delete)** from the dropdown menu.
        iii. Enter the name of the input/output (to be deleted)
        iv. Click **delete**
    e. Clear current list of selected Inputs/Outputs:
        i. Click on **clear**
3. Sigma Outputs:
    a. The 'sigma' outputs can be used to control actuators. Citation: *Segall-Shapiro et. al. (2014) "A resource allocator for transcription based on a highly fragmented T7 RNA polymerase". Mol Syst Biol Jul 30;10:742. doi: 10.15252/msb.20145299*.
4. Run: Enter a **design name** and click on **Run**
    a. You will be notified if this design name already exists in your results.
    b. During the run, the Cello print statements will be displayed on the right side of the page. This same text can be viewed in the output log file in the 'results.html' page.

Options.html

1. Select a UCF: Select the UCF Cello must use while designing the genetic circuit.
    a. The UCF contains the genetic parts, gates library, response functions, Eugene rules, etc.
2. View collection: View a particular section of the UCF.
3. Upload UCF
    a. New UCF's should be produced programmatically. The Cello source code comes with a python-based UCF writer (~/cello/tools/pycello/ucf_writer.py) and a Java-based UCF writer (~/cello/src/main/java/org/cellocad/adaptors/ucfwriters/).
    b. Once the UCF is produced, it can be uploaded here.
4. Delete UCF
5. Download UCF
    a. View the JSON file in a plain text editor.

Results.html

The results page is divided into 13 sections spread over 5 tabs. These results can be viewed by clicking on a tab or by downloading all the results as a zipped folder (by clicking on the **Download Zip** button).

- **Cello Input and Output Log**

1. input Verilog:
    a. Verilog Input for Cello (specified in Verilog.html)
2. output log file:
    a. Print statements produced by Cello software. The log file is divided into sections based on the order of operations in the software.

i. UCF Validation: Checks to see if the UCF used for the current job is a valid UCF. Checks to see if each section has valid values.
ii. Logic synthesis, Wiring diagram: Specifies a textual representation for the abstract directed acyclic graph generated for the input Verilog.
iii. Loading Parts: Loads all the individual parts (promoters, rbs, ribozyme, terminator, scar, cds) from the UCF.
iv. Loading Gate Library: Loads the genetic gates (repressor) from the UCF
v. Loading input and output gates: Loads the inputs and outputs specified for the current circuit.
vi. Loading Response Functions: Loads the response function (equation for the hill function) that specifies the functional behavior of each genetic gate (repressor) from the UCF.
vii. Loading Cytometry Data: Loads the cytometry data for each gate from the UCF
viii. Assignment Algorithm: Specifies the assignment algorithm used to produce an assigned genetic circuit (genetic regulatory network) from the abstract genetic regulatory network specified in **Logic synthesis, Wiring diagram**. The assignment algorithm is simulated annealing by default.
ix. Processing best circuits: Simulates and predicts REU values for the assigned genetic circuit with the best score. See the **Circuit Assignment** section for additional details.
x. Figures: Generates figures for the wiring diagrams, histograms and plots.
xi. Plasmid DNA sequences: Runs Eugene to generate valid combinations of circuit designs and generates SBOL and Genbank files.

- **Circuit Assignment**

3. gate assignment:
    a. A schematic representation of the genetic circuit along with the gate assignments
    b. Each box represents a gate and contains the following information
        i. Gate Type: NOT or NOR (1st Row 1st Column)
        ii. Genetic gate: (1st Row 2nd Column) e.g.: S1_SrpR
        iii. Stage (1st Row 3rd Column): Specifies how many levels a particular gate or output is away from the inputs. The stage of a circuit input is always 0. The stage of a gate or an output equals one plus the maximum stage of all the inputs (not to be confused with the circuit input) of that gate.
        iv. Gate Score (2nd Row): Ratio of the lowest ON state (among all expected ON states) and highest OFF state (among all expected OFF states). Refer: SI Section II.C.
        v. Truth Table (3rd Row): 1 represents an ON state and 0 represents an OFF state. Each column of the values (1 or 0) represents the corresponding row of the truth table. e.g.: 1011 states that the output value is 1 (the 1 in column 1/the left most 1) for the first row of the truth table. Similarly, the output value is 0 (the 2nd column) for the second row of the truth table.
    c. Inputs and Outputs of the circuit
        i. INPUT/OUTPUT type (1st Row 1st Column)
        ii. Name of the input/output (1st Row 2nd Column)
        iii. Stage (1st Row 3rd Column): Specifies how many levels a particular gate or output is away from the inputs. The stage of a circuit input is always 0. The stage of a gate or an output equals one plus the maximum stage of all the inputs (not to be confused with the circuit input) of that gate.
        iv. Score (2nd Row): For circuit outputs, score is the ratio of the lowest ON state (among all expected ON states) and highest OFF state (among all expected OFF states). Refer: SI Section II.C. For circuit inputs score is the ratio of ON state REU value and the OFF state REU value.
        v. Truth Table (3rd Row): 1 represent an ON state and 0 represents an OFF state. Each column of the values (1 or 0) represents the corresponding row of the truth table. e.g.: 1011 states that the output value is 1 (the 1 in column 1/the

left most 1) for the first row of the truth table. Similarly, the output value is 0 (the 2nd column) for the second row of the truth table.

4. response functions:
    a. A schematic representation of the response function of each assigned gate where each box represents a graph (in log scale, where the range for the x and y axis is 10^{-3} to 10^{3}) showing the response function of the gate (repressor). The bigger ticks on the axes represent powers of 10 (10^{-3}, 10^{-2}, 10^{-1}, 0, 10^{1}, 10^{2}, and 10^{3}). The smaller ticks indicate multiples of each power of 10 (e.g.: between 10^{1} and 10^{2}, the smaller ticks indicate 20, 30, 40, 50, 60, 70, 80, and 90). Black dots indicate low and high noise margin thresholds for that gate. The dashed lines show the highest OFF input (leftmost dashed line) and the lowest ON input (rightmost dashed line) for that gate.
5. predicted expression levels (REU) :
    a. A truth table showing the simulated REU values for each Input, Gate and Output in the circuit, for each row of the truth table. The order of values (0 or 1) in the truth table is determined by the order of the inputs of the circuit.
6. predicted toxicity (relative OD600):
    a. A truth table showing the predicted toxicity (measured in relative OD600. Refer SI Section VII.C.10.). For each row in the truth table, all gates have a relative cell growth value (normalized OD600) where 1.0 indicates normal cell growth and 0.0 indicates no cell growth. The circuit growth value is computed by multiplying all gate growth values for that row.
7. logic circuit simulation:
    a. A detailed simulation showing the truth table for each gate in the circuit.
        i. Logic Circuit: This sections is a textual representation of the circuit. The 1st column specifies the type of gate, input or output. The 2nd column specifies the truth table. See Section 3.b.v. (under Results.html) of this document for details. The 3rd column specifies the genetic gate (repressor) or input or output. The 4th Column specifies the index of the gate/input/output. The 5th Column specifies the index of the inputs to that gate/output. Circuit inputs will not have any value specified in the 5th column. The 6th Column specifies the score for the gate/input/output. See Section 3.b.iv. (under Results.html) of this document for details. The 7th Column specifies the toxicity of each gate/output. See Section 6 (under Results.html) of this document for details.
        ii. Circuit Score and Cell growth. This specifies the overall score and toxicity for the entire circuit.
        iii. Detailed simulation for each gate:
            I. The first line, 1st column specifies the genetic gate/input/output. The 2nd column specifies the individual gate score.
            II. The rest of the lines show the truth table simulation for that gate/input/output. The 1st column is the type of input/output/gate (e.g. INPUT, OUTPUT, NOT, NOR). The 2nd column specifies the gate stage See Section 3.b.iii. (under Results.html) of this document for details. The 3rd column specifies the truth table values for that gate/input/output. The values in square brackets are input values and the value after the colon is the output value. The 4th column specifies the simulated REU value of the 1st input to that gate/output. The 5th column specifies the simulated REU value of the 2nd input (if the 4th column is not empty) or the simulated REU value of the 1st input (if the 4th column is empty). Circuit inputs do not have any value in the 4th or 5th columns. The 6th column specifies the simulated output REU value of that gate/input/output. The 7th column specifies the toxicity value for that gate/output.
8. bionetlist: A simplified text representation of the circuit.
    a. 1st Column: Gate/Input/Output
    b. 2nd Column: First input for the gate or output specified in column 1. If the 2nd column shows a truth table, it implies that the first column is an Input.
    c. 3rd Column: Second input for the gate or output specified in column 1.

- **Histograms and plots**

9. predicted gate REUs:
    a. A schematic representation of showing the predicted gate REU data for each repressor. Each box shows histograms (of predicted gate REUs) for a gate and each row in the box represents a row in the truth table. Light grey histograms represent predicted OFF states and dark grey histograms represent predicted ON states. The number on the left of each row of the graph, specifies the states of the Inputs of the circuit for each row of the truth table. 0 represents an OFF state and 1 represents an ON state for an input. The order of the values specifies the state of the inputs. For e.g.: **10** indicates that Input 1 of the circuit is in the ON state and Input 2 of the circuit in the OFF state.
10. predicted output REUs:
    a. Histograms showing the predicted REU data for the outputs of the circuit. Each row represents a row in the truth table. Light grey histograms represent predicted OFF states and dark grey histograms represent predicted ON states. The number on the left of each row of the graph, specifies the states of the Inputs of the circuit for each row of the truth table. 0 represents an OFF state and 1 represents an ON state for an input. The order of the values specifies the state of the inputs. For e.g.: **10** indicates that Input 1 of the circuit is in the ON state and Input 2 of the circuit in the OFF state.

- **Eugene Specification**

11. Eugene rules for plasmid design
    a. One assignment for one circuit can be physically realized in many different genetic layouts by varying gate order and orientation. Eugene rules are auto-generated based on the 'eugene' collection in the UCF and the circuit assignment, and these rules govern the space of allowed solutions for plasmid DNA sequences. Citation: *Oberortner, Ernst, et al. "A rule-based design specification language for synthetic biology." ACM Journal on Emerging Technologies in Computing Systems (JETC) 11.3 (2014): 25.*
12. Circuit variants: Variants of the circuit based on the Eugene rules.
    a. The default UCF, named Eco1C1G1T1.UCF.json, contains Eugene rules that constrain the combinatorial design of genetic layouts to a single solution. To achieve more than one plasmid, upload a new UCF with certain Eugene constraints removed (such as 'ALL_FORWARD').

- **SBOL and Plasmid Files**

13. SBOL files and Plasmid files:
    a. Each plasmid is encoded as either a Genbank file or SBOL file.
    b. The 'genetic_locations' collection in the UCF determines the plasmid backbones and the locations within those plasmids where the sensor module, circuit module, and output module will be inserted.
    c. In the Eco1C1G1T1 UCF, the circuit and sensor modules are inserted into one backbone (p15A), and the output module is inserted into a second backbone (pSC101).