Homework 4 — assigned on Friday 30 March, due 11:59PM on Friday 13 April

General instructions

Total number of points available: 150 pts.

The goal of this assignment is to use ODE modeling and Python-based simulation techniques to explore the behavior of a DNA strand displacement reaction network.

For questions that do not require a programming answer, please include your answer in your submitted file in some appropriate format, e.g., as a comment in your code.

Note that if you use any Python programming techniques beyond those covered in class, you must supply comments demonstrating that you understand how and why your code works.

Also note that points will be deducted for bad software engineering practices, in particular, large-scale duplication of code blocks where an iterative solution would be preferable.

How to submit

Your submission should consist of the following parts:

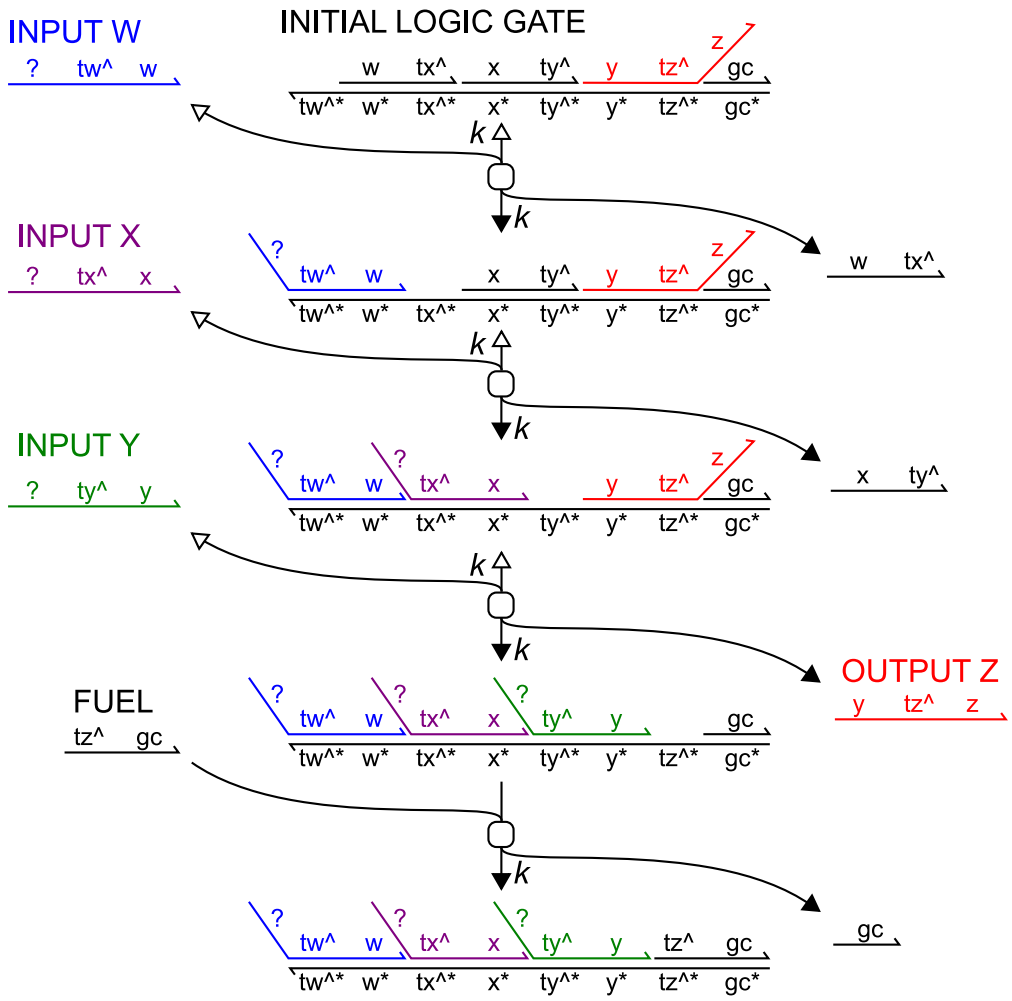
- A commented source file that contains your name and your code. The comments should identify which code answers which question, and should provide any necessary explanation of the code's operation.
- A text file that contains the log of you interacting with the Python toplevel interpreter to test the code from each question, to demonstrate that you understand how to use your code, and that your code works correctly.
- Image files containing any plots produced by your code.
- Answers to any non-programming questions, either as separate files or integrated as comments into your main code file.

All submissions must be submitted via UNM Learn. Please archive your files into a zipfile and submit the zipfile.

4.1 Modeling a strand displacement logic gate (150 pts)

The following diagram illustrates the reactions for a 3-input DNA strand displacement "AND" logic gate, which extends the 2-input version presented in the lectures. This logic gate detects

input species W, X, and Y, and produces the output species Z, and uses the "presence-absence" encoding of logic signals.



In order to model this system, carry out the following tasks:

(a) (30 pts) Convert the above reaction diagram into an abstract CRN model, by assigning species names to each of the DNA structures illustrated above and listing the reactions between those abstract species.

In constructing this model, you should assume that all reactions (both the forward and reverse directions of reversible toehold exchange reactions, and irreversible strand displacement reactions) proceed with the same rate constant:

$$k = 5 \times 10^5 \, \mathrm{M}^{-1} \, \mathrm{s}^{-1} = 5 \times 10^{-4} \, \mathrm{nM}^{-1} \, \mathrm{s}^{-1}.$$

(b) (40 pts) Convert your abstract CRN model from above into a vector ODE model, using the techniques covered in class.

(c) (50 pts) Implement your vector ODE model from above in Python, and integrate this ODE model for each combination of True/False for the three Boolean input variables (W, X, and Y), that is, eight combinations in total.

The initial concentrations of the species in the model should be as follows:

- 10 nM for each input species (W, X, and Y) if the corresponding input is True, or 0 nM if that input is False;
- 0 nM for the output species, Z;
- 100 nM for the initial gate species and for the fuel strand;
- 0 nM for all other intermediate gate species and waste species.

You should run each simulation for 7200 seconds (that is, units of simulation time). ***Hint:*** *when setting up the initial conditions, you will need to ensure that the units of your rate constant match the concentration units.*

Use `matplotlib` to produce a plot of the concentrations of the four signal species (W, X, Y, and Z) against time, for each of the eight simulations.

(d) (20 pts) Using the vector ODE model from above, run another collection of eight simulations. Use the same rate constant as above, and run each for 7200 seconds of simulation time. In each simulation, all three input species (W, X, and Y) should be present initially with the same concentration, but that concentration will vary between simulations. Thus, the initial concentrations of the species in each simulation should be as follows:

- C for each input species, where C is a different one of the following five concentrations in each simulation: 10 nM, 25 nM, 50 nM, 100 nM, 150 nM, or 200 nM;
- 0 nM for the output species, Z;
- 100 nM for the initial gate species and for the fuel strand;
- 0 nM for all other intermediate gate species and waste species.

Use `matplotlib` to plot a bar chart of the final concentration of Z from each of these simulations, expressed as a **percentage of the original input concentration** in each case (the concentration of just one input, not the sum of all three). You should label each of the five bars with the corresponding input concentration.

(e) (10 pts) Explain why the output yields drop towards 50% for the input concentrations above 100 nM.