MOL518 Spring 2026

Homework 1

Due: Friday February 6, 2026

For this assignment, you will generate and then submit a Jupyter notebook on Canvas. You may use generative AI tools as per the syllabus unless otherwise stated in the problem.

First, make a new Jupyter notebook named `MOL518_HW1_<YourLastName>.ipynb`*. All problems will be answered in this notebook. For each problem, start with a Markdown cell using Problem XX as the title, i.e. "# Problem 1".*

Problem 1: Practice with Markdown

Figure 1 is the preface page for Lubert Stryer's classic textbook *Biochemistry*. Recreate this page in a Markdown cell in your notebook. The image at the bottom of the page is provided as `StryerPreface_liverketone.png`. *Do not use AI for this problem.*

PREFACE

The more we learn, the more we discover connections threading through our biochemical world. In writing the sixth edition, we have made every effort to present these connections in a way that will help first-time students of biochemistry understand the subject and how very relevant it is to their lives.

Emphasis on Physiological Relevance

Biochemistry is returning to its roots to renew the study of its role in physiology, with the tools of molecular biology and the information gained from gene sequencing in hand. In the sixth edition, we emphasize that an understanding of biochemical pathways is the underpinning for an understanding of physiological systems. Biochemical pathways make more sense to students when they understand how these pathways relate to the physiology of familiar activities such as digestion, respiration, and exercise. In this edition, particularly in the chapters on metabolism, we have taken several steps to ensure that students have a view of the bigger picture:

- Discussions of metabolic regulation emphasize the **everyday conditions** that determine regulation: exercise versus rest; fed versus fasting.
- New **pathway-integration figures** show how multiple pathways work together under a specific condition, such as during a fast.
- More **physiologically relevant examples** have been added throughout the book.

This physiological perspective is also evident in the new chapter on drug development. The use of a foreign compound to inhibit a specific enzyme sometimes has surprising physiological consequences that reveal new physiological principles.

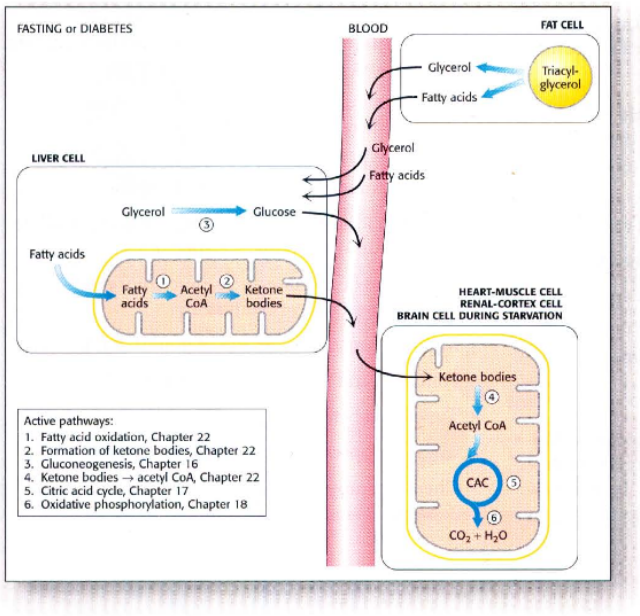

Figure 1: Stryer preface page.

Problem 2: Wet-Lab Arithmetic

You are doing a phage infection assay. You have a tube of virus stock at a known titer, and you want to prepare an inoculum at a target multiplicity of infection (MOI) in a specified final volume. You will now prepare a mini "protocol note" for yourself: define variables clearly, keep track of units, and print a short summary at the end. *Do not use AI to generate code for this problem. You may use AI to debug if you get an error*

a) In a single code cell, define the following variables:

- `cells` = 250,000 (cells)
- `moi` = 1.5 (dimensionless)
- `titer` = 2.0×10^8 (infectious units per mL)
- `inoc_vol_uL` = 500 (μL total inoculation volume)

b) Compute the number of infectious units needed for this infection, `iu_needed`. Briefly explain what MOI means in words.

c) Convert the titer to infectious units per μL, then compute the volume of virus stock to add (in μL) to deliver `iu_needed`.

d) Compute the remaining volume (in μL) that must be filled with media so the final inoculum volume is exactly `inoc_vol_uL`.

e) Print a short summary that includes: `cells`, `moi`, `iu_needed`, virus volume (μL), media volume (μL).

Problem 3: CSV Sanity Check

You ran a plate reader growth experiment and saved the results as a CSV file. Before doing any serious analysis, you want to do a fast sanity check to confirm the file saved correctly, the sampling looks consistent, and the values are in a reasonable range. *Do not use AI to generate code for this problem. You may use AI to debug if you get an error*

a) Load `Lecture_2/data/growth_curve1.csv` as a 2D NumPy array called `data`.

b) Print the shape of `data`. Describe what this tell you.

c) Print the first row and the last row, and briefly describe what those rows mean.

d) Extract `time` in seconds and `od` as 1D arrays.

e) Compute and print the sampling interval, the time between consecutive timepoints, in minutes.

f) Compute and print the total duration of the experiment in units of days.

g) Compute and print the following summary quantities, and label each one clearly in the output:

- absolute OD change from start to end
- total fold change in OD from start to end

Problem 4: Fake data!

In this problem, you will generate a small *synthetic* growth curve dataset. "Fake" data can be very useful for testing code, practicing analysis, or simulating experiments because you know exactly what to expect. *Do not use AI to generate code for this problem. You may use AI to debug if you get an error*

a) Create a NumPy array called `time_hr` with time points in hours the go from 0 to 12 hours in increments of 0.25 hours. Print `time_hr` to check that it looks correct.

b) Define two variables: `od0 = 0.08` and `doubling_time_hr = 1.5`. Explain what "doubling time" means in the context of bacterial growth.

c) Compute the `od` time series using the following fomula

$$OD = OD_0 \times 2^{\frac{t}{t_{doubling}}}. \tag{1}$$

Print `od` to confirm your results.

d) Normalize `od` by its first value. Print `od_norm`.

e) Use `np.column_stack` to make a 2D array with columns `time_hr`, `od`, `od_norm`. Print the resulting array and its shape.

f) Save the result to `simulated_growth.csv`.

g) Reload the file you saved into a varaible `data2`. Print `data2.shape` and the first 3 rows, and confirm that the reloaded numbers match what you expect.

Problem 5: Sample Tracking and Lane Batching

You will often need to organize samples into a workable datastructure for downstream analysis. In this problem, you will treat a Python list as your "sample sheet" and practice counting, appending, sorting, and slicing into batches. *Do not use AI to generate code for this problem. You may use AI to debug if you get an error*

a) Create a Python list called `samples` with the following sample IDs, in this exact order:

```
"WT_A", "WT_B", "WT_C", "KO1_A", "KO1_B", "KO1_C",
    "KO2_A", "KO2_B", "KO2_C", "Rescue_A", "Rescue_B", "Rescue_C"
```

b) Print the number of samples using the list and briefly interpret what that number means in this context.

c) Append two new samples to the end of the list: `"Blank"` and `"PositiveCtrl"`. Print the updated list.

d) Create a new list called `samples_sorted` that is an alphabetically sorted version of `samples`. Print both lists (original and new) to demonstrate that only the new one was sorted.

e) Imagine you can only process 5 samples per "lane" (or batch). Create three new lists by slicing `samples_sorted`:

- `lane1` = first 5 samples
- `lane2` = next 5 samples
- `lane3` = remaining samples

f) Print `lane1`, `lane2`, and `lane3`. Do you think slicing a safer approach than manually retyping sample IDs? What if your list had 1000 samples?