Data 102, Fall 2025
Homework 3

Due: **5:00 PM** Friday, October 24, 2025

Submission Instructions

Homework assignments throughout the course will have a written portion and a code portion. Please follow the directions below to properly submit both portions.

Written Portion:

- Every answer should contain a calculation or reasoning.
- You may write the written portions on paper or in LaTeX.
- If you type your written responses, please make sure to put it in a markdown cell instead of writing it as a comment in a code cell.
- Please start each question on a new page.
- It is your responsibility to check that work on all the scanned pages is legible.

Code Portion:

- You should append any code you wrote in the PDF you submit. You can either do so by copy and paste the code into a text file or convert your Jupyter Notebook to PDF.
- Run your notebook and make sure you print out your outputs from running the code.
- It is your responsibility to check that your code and answers show up in the PDF file.

Submitting:

You will submit a PDF file to Pensieve containing all the work you want graded (including your math and code).

- When downloading your Jupyter Notebook, make sure you go to File → Save and Export Notebook As → PDF; do not just print page from your web browser because your code and written responses will be cut off.
- Combine the PDFs from the written and code portions into one PDF. Here is a useful tool for doing so. As a Berkeley student, you get free access to Adobe Acrobat, which you can use to merge as many PDFs as you want.
- Please see this guide for how to submit your PDF on Pensieve. In particular, for each question on the assignment, please make sure you understand how to select the corresponding page(s) that contain your solution (see item 2 on the last page).

Late assignments will count towards your slip days; it is your responsibility to ensure you have enough time to submit your work.

Data science is a collaborative activity. While you may talk with others about the homework, please write up your solutions individually. If you discuss the homework with your peers, please include their names on your submission. Please make sure any handwritten answers are legible, as we may deduct points otherwise.

GLM for Dilution Assay

1. In this question, you'll go beyond the four GLM types you saw in class, and explore a new kind of GLM for solving a specific scientific problem.

   Being able to reformulate problems as generalized linear models (GLMs) enables you to solve a wide variety of problems with existing packages. We recommend reviewing the examples of GLMs from Lectures 10 and 11. In particular, make sure you understand that formulating a GLM involves choosing an 1) output distribution and 2) link function that are appropriate for the application at hand.

   In this problem, you'll retrace the footsteps of the statistician R. A. Fisher and develop one of the very first applications of GLMs. In a 1922 paper, Fisher formulated a GLM he used to estimate the unknown concentration ρ_0 of an infectious microbe in a solution. Without specialized technology to directly measure ρ_0 from the solution, Fisher devised the following procedure: we will progressively dilute the original solution, and after each dilution, we'll pour out some small volume v onto a sterile plate. If zero microbes land on the plate, it will remain sterile, but if any microbes land on a plate, they will grow visibly on it (we call this an "infected plate"). By observing whether or not the plate is infected at each dilution, and by formulating the relationship between this data and ρ_0 as a GLM, we can estimate ρ_0 from this data.

   Specifically, let ρ_t denote the concentration at dilution t. Each time, we dilute the solution to be half its concentration, such that

   $$\rho_t = \frac{\rho_0}{2^t} \tag{1}$$

   for $t = 0, 1, \ldots$. When we pour out volume v of the solution onto the plate, and wait awhile to allow for microbe growth, we can observe whether a plate was infected (*i.e.*, has a non-zero number of microbes) or is sterile (*i.e.*, has zero microbes). Therefore, our data $Y_t \in \{0, 1\}$ is whether or not the plate is infected at each dilution.

   In other words, we observe a sequence of binary values $Y_0, \ldots, Y_t$, and from that, we want to estimate the initial concentration ρ_0. In this question, we'll formulate a GLM that relates ρ_0 and t to the data Y_t. Estimating the parameters of this GLM will then allow us to estimate ρ_0, as will become clear in the last part.

   (a) (2 points) At dilution t, the data $Y_t \in \{0, 1\}$ indicates whether or not the plate is infected. The chance that a plate gets infected is denoted by $\mu(t) := \mathbb{E}[Y_t]$. Write down an output distribution for Y_t that is appropriate for the values it takes on, using $\mu(t)$ as a parameter. (We'll derive what $\mu(t)$ should be in the next part).

(b) (3 points) At dilution t, we pour out volume v onto a plate, so the expected number of microbes on the plate is $\rho_t v$. The actual number of microbes is distributed as a Poisson random variable with this mean $\rho_t v$:

$$\#\text{ microbes on plate at dilution } t \sim \text{Poisson}(\rho_t v). \tag{3}$$

Using this fact, write out an expression for $\mu(t) := \mathbb{E}[Y_t]$. Start with

$$\mu(t) = \mathbb{P}(\text{plate is infected at dilution } t) \tag{4}$$
$$= 1 - \mathbb{P}(\text{there are 0 microbes on plate at dilution } t). \tag{5}$$

Your solution should be in terms of ρ_0, t, and other constants.

(c) (3 points) Use your findings from part (b), along with Equation (1), to find a link function g such that

$$g(\mu(t)) = \beta_0 + \beta_1 t \tag{8}$$

for some constants β_0 and β_1. (Remember that in class, we talked about the inverse link function g^{-1}, such that $\mu(t) = g^{-1}(\beta_0 + \beta_1 t)$). Your answer should be of the form "$\beta_0 = \ldots$ and $\beta_1 = \ldots$".

(d) (2 points) Choosing an appropriate output distribution and link function as we've done in Parts (a) and (c) completes the GLM specification. Now, suppose you've estimated β_0 and β_1 (*e.g.*, using maximum-likelihood estimation). Using your findings from part (c), write down an estimate of ρ_0.

Hint: For this question, you do not need to estimate β_0 and β_1: assume you know them, and find a way to estimate ρ_0 from them.

Image Denoising with Gibbs Sampling

2. In this problem, we derive a Gibbs sampling algorithm to restore a corrupted image [1]. A grayscale image can be represented by a 2-dimensional array X of shape $n \times m$, where the intensity of the (i, j)-th pixel is X_{ij}. In this problem, we are given an image X whose pixels have been corrupted by noise, and the goal is to recover the original image Z.

(a) (2 points) Load the grayscale image `X.pkl` as a numpy array X. Visualize the image.

From plotting the image X, it is clear that it has been corrupted with noise. Let Z denote the original image, which we also represent as an $n \times m$ array. Let $\mathcal{I} = \{(i, j) : 1 \le i \le n \text{ and } 1 \le j \le m\}$ denote the collection of all pixels in the image, represented by the corresponding index of the array. Given a pixel (i, j), define the set of *neighboring pixels* to be

$$N_{(i,j)} = \left\{(i', j') \in \mathcal{I} : (i = i' \text{ and } |j - j'| = 1) \textbf{ or } (|i - i'| = 1 \text{ and } j = j')\right\}.$$

We consider the following prior over the original image:

$$p(Z) \propto \exp\left(-\frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\left[aZ_{ij}^2 - b\sum_{(i',j')\in N_{(i,j)}} Z_{ij}Z_{i'j'}\right]\right).$$

(b) (1 point) Explain why this prior captures the fact that, in natural images, neighboring pixels are likely be similar.

Hint: Let $b = a/4$, and consider two cases for a pixel with four neighbors: (i) where the pixel's value is equal to the values for all four neighbors, and (ii) where all four neighbors have a different value.

Hint: Note that $Z_{i,j}$ can take on negative quantities.

Assuming the image has been corrupted with Gaussian noise $X_{(i,j)} \mid Z_{(i,j)} \sim \mathcal{N}(Z_{(i,j)}, \tau^{-1})$ (independently across pixels $(i,j) \in \mathcal{I}$), the complete posterior can be written as

$$p(Z \mid X) \propto \exp\left(-\frac{1}{2}\sum_{(i,j)}\left[(a+\tau)Z_{ij}^2 - 2\tau Z_{ij}X_{ij} - b\sum_{(i',j')\in N_{(i,j)}} Z_{ij}Z_{i'j'}\right]\right) \quad (11)$$

Let $S_{ij} = \sum_{(i',j')\in N_{(i,j)}} Z_{i'j'}$. By completing the square in the posterior (11), we have

$$Z_{ij} \mid (Z_{i'j'})_{(i',j')\neq(i,j)}, X \sim \mathcal{N}\left(\frac{\tau X_{ij} + bS_{ij}}{a+\tau}, \frac{1}{a+\tau}\right) \quad (12)$$

(c) (2 points) Fill in the missing line of pseudocode for a Gibbs sampler of the posterior, $p(Z|X)$. **Be specific with each conditioned variable and sub/superscript!**

- Initialize $Z^{(0)} = X$.
- For $t = 1, \ldots, T$:
  - Sample $Z_{1,1}^{(t)} \sim p(Z_{1,1} \mid Z_{1,2} = Z_{1,2}^{(t-1)}, Z_{1,3} = Z_{1,3}^{(t-1)}, \ldots, Z_{n,m} = Z_{n,m}^{(t-1)}, X)$.
  - Sample $Z_{1,2}^{(t)} \sim p(Z_{1,2} \mid Z_{1,1} = Z_{1,1}^{(t)}, Z_{1,3} = Z_{1,3}^{(t-1)}, \ldots, Z_{n,m} = Z_{n,m}^{(t-1)}, X)$.
  - Sample $Z_{1,3}^{(t)} \sim$ *# TODO: fill this in.*
  - ...
  - Sample $Z_{n,m}^{(t)} \sim p(Z_{n,m} \mid Z_{1,1} = Z_{1,1}^{(t)}, Z_{1,2} = Z_{1,2}^{(t)}, \ldots, Z_{n,m-1} = Z_{n,m-1}^{(t)}, X)$

(d) (3 points) Write the pseudo-code from Part (b) more explicitly both by using a double for-loop over $(i,j) \in \mathcal{I}$ and by being explicit about the conditional distributions of the form $p(Z_{1,1} \mid Z_{1,2} = Z_{1,2}^{(t-1)}, Z_{1,3} = Z_{1,3}^{(t-1)}, \ldots, Z_{n,m} = Z_{n,m}^{(t-1)}, X)$.

(e) (5 points) Implement the Gibbs sampler from Part (c) with $a = 250, b = 62.5$, and $\tau = 0.01$. Run your code for $T = 1$ iteration, i.e. update each coordinate exactly once. Visualize the resulting image $Z^{(1)}$. Time your code and estimate how long it would take to compute $Z^{(100)}$.

Hint: To convert your pseudo-code into real code, it might be helpful to use `np.random.randn()` to generate a $\mathcal{N}(0,1)$ random variable at each step.

(f) (1 point) The bottleneck in running the Gibbs sampler from Part (d) is sampling a single pixel Z_{ij} with the values of all others held fixed. Fortunately, it is possible to speed up the sampling process with an improvement known as *blocked Gibbs sampling*. Specifically, define two subsets of the pixels $\mathcal{I}_{\text{even}} = \{(i,j) : i+j \text{ is even}\}$ and $\mathcal{I}_{\text{odd}} = \{(i,j) : i+j \text{ is odd}\}$. The blocked Gibbs sampler proceeds as follows:

- Initialize $Z^{(0)} = X$.
- For $t = 1, \ldots, T$:
  - Let $Z = Z^{(t-1)}$.
  - Let Δ be an $n \times m$ matrix with $\mathcal{N}(0, \frac{1}{a+\tau})$ entries.

- For $(i, j) \in \mathcal{I}_{\text{even}}$:
  - Let $S_{ij} = \sum_{(i',j') \in N_{(i,j)}} Z_{i'j'}$
- Update $Z_{\mathcal{I}_{\text{even}}} = \frac{\tau}{a+\tau} X_{\mathcal{I}_{\text{even}}} + \frac{b}{a+\tau} S_{\mathcal{I}_{\text{even}}} + \Delta_{\mathcal{I}_{\text{even}}}$.
- For $(i, j) \in \mathcal{I}_{\text{odd}}$:
  - Let $S_{ij} = \sum_{(i',j') \in N_{(i,j)}} Z_{i'j'}$
- Update $Z_{\mathcal{I}_{\text{odd}}} = \frac{\tau}{a+\tau} X_{\mathcal{I}_{\text{odd}}} + \frac{b}{a+\tau} S_{\mathcal{I}_{\text{odd}}} + \Delta_{\mathcal{I}_{\text{odd}}}$.
- Let $Z^{(t)} = Z$.

The advantage of this approach is that the inner for-loops can be *vectorized*. Explain why updating half the variables $Z_{\mathcal{I}_{\text{even}}}$ (and then $Z_{\mathcal{I}_{\text{odd}}}$) at once is justified.
Hint: if you're not sure why, try drawing out a small (e.g., 4×4) grid of pixels and label each one with whether it's in $\mathcal{I}_{\text{odd}}$ or $\mathcal{I}_{\text{even}}$.

(g) (0 points) (**Optional**) Implement the Gibbs sampler from Part (e) using $a = 250, b = 62.5$ and $\tau = 0.01$. Run your code for $T = 100$ iterations, and visualize the resulting image $Z^{(100)}$. Time your code and report how long it took.
Hint: Compute the entire $n \times m$ matrix S at once using matrix operations on Z. You may find it helpful to pad the matrix Z with a border of zeros using `Z_bar = np.pad(Z, 1)`. Then use slicing on the $(n+2) \times (m+2)$ matrix `Z_bar` to compute S.

Fitting and interpreting generalized linear models

3. In this question, you'll work with a dataset of married straight women from 1975 in the file `mroz2.csv`. Each row represents one woman, and the columns represent the following information about her:

   - `inlf`: Whether the woman was in the labor force (i.e., worked for pay) in 1975
   - `hours`: Number of hours the woman worked in 1975
   - `wage`: Woman's hourly wage in 1975
   - `kidslt6`: Number of young children (<6 years old) in the woman's household
   - `kidsge6`: Number of older children (6-18 years old) in the woman's household
   - `age`: Woman's age (years)
   - `educ`: Woman's amount of schooling (years)
   - `hushrs`: Hours worked by husband in 1975
   - `husage`: Husband's age
   - `huseduc`: Husband's years of schooling
   - `huswage`: Husband's hourly wage in 1975
   - `faminc`: Family income in 1975
   - `mtr`: Federal marginal tax rate paid by the woman
   - `motheduc`: Woman's mother's years of schooling

- `fatheduc`: Woman's father's years of schooling
- `unem`: Unemployment rate in county of residence
- `city`: Whether the woman lived in a standard metropolitan statistical area
- `lwage`: log(`wage`)

Throughout this question (except for the last three parts), you should only work with the rows where `inlf==1`: that is, we're only going to make predictions for women who were in the labor force, since for everyone else, the number of hours worked is zero.

For parts (a), (b), (e), and (f), we'll predict the number of hours each woman worked (`hours`) from the number of children she had (`kidslt6` and `kidsge6`).

(a) (2 points) (Frequentist only) Use both Poisson regression and negative binomial regression to predict `hours` from `kidslt6` and `kidsge6`. Does the Poisson model suffer from overdispersion? Explain why or why not.
Hint: You may find section 3.3 of the textbook helpful in understanding overdispersion.

(b) (1 point) (Frequentist only) Using your negative binomial regression results from part (a), interpret the coefficient for `kidslt6` by filling in the blank in the following sentence with a description of how much more or less: "For every additional child under 6 a woman has, the model predicts she'll work _____ hours."

(c) (2 points) (Frequentist only) In this part, you'll compare the following three models:

(i) Negative binomial regression predicting hours worked from number of children (same as part (a), two features and constant)

(ii) Negative binomial regression predicting hours worked from number of children, husband's age, and whether she lives in a city (four features and constant)

(iii) Negative binomial regression predicting hours worked from number of hours husband works, woman's hourly wage, and mother's and father's years of schooling (four features and constant)

For each one, compute and report the AIC. Based on the results, which of these three models is best?

Hint: You may find the `llf` and `params` attributes of model summaries helpful, which contain the log-likelihood and a Series with the coefficients respectively.

(d) (0 points) (Optional, Frequentist only) Split the data into a training and test set. Fit the three models from part (c) on the training set, and report the RMSE on the test set. Which of the three models performs best on the held-out data? Are your results the same as part (c)? Explain why.

(e) (2 points) (Bayesian) Use Bambi to fit a negative binomial regression model with to predict hours worked from number of children (same as in part (a) and (c)(i)). Use the `az.plot_posterior(trace)` function to visualize the posterior for each coefficient, along with 95% credible intervals (specifically, HDIs) for each one.

Are the credible intervals different from the confidence intervals? Why or why not?

(f) (2 points) (Bayesian) Generate posterior predictive samples for the model above. Consider women with two children above 6 and one child under 6. On the same graph, overlay two histograms: (i) the actual distribution of number of hours worked

by these women, and (ii) the posterior predictive distribution of the predicted number of hours for these women.

What does the graph tell you about how well the model fits the data, and why?

Hint: you may find the demo from lecture 12 helpful in determining which Bambi methods to use for the posterior predictive check.

(g) (1 point) Suppose instead of predicting the number of hours worked, we wanted to predict whether the woman is in the labor force (the `inlf` column). What regression family (likelihood model and link function) should we use and why?

Hint: For this question, you only need to specify which likelihood model and link function to use and why; you don't need to write any code.

Hint: You may want to reshape the output of your posterior predictive sample in order to plot it (the lecture notebooks are a helpful resource).

(h) (0 points) (**Optional**) Repeat part (c), but instead of computing the AIC, fit each model on 80% of the data and compute the RMSE on the remaining 20%. Are the results the same or different? Explain why.

(i) (0 points) (**Optional**) Repeat parts (a) - (f) and (h) for predicting whether the woman is in the labor force.

References

[1] Stuart Geman and Donald Geman (1984). *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images.* IEEE Transactions on Pattern Analysis and Machine Intelligence, (6), 721-741.