Dr. Tolga Soyata *© George Mason University, Department of ECE* *tsoyata@gmu.edu*

ECE655 (Advanced GPU Programming and Deep Learning) Project2 (10 points)

Deadline: Sep 28, 2025 11:59PM. 2 point penalty per each late day

- In this project, you will use your knowledge from Lectures 2 and 3 to apply Linear Regression to a circuits application. You are required to generate results in two ways: 1) solely by using numpy, and 2) by using PyTorch. **You are not allowed to use the sci-kit package.**
- Document your results in a report, written in LaTeX (Overleaf). The report is expected to be 4-5 pages, which is packed with plots and tables.
- Your computer must have the full Anaconda and PyTorch packages installed to complete this project. You are only asked to use CPU tensors for the PyTorch implementation.
- Name your code **`P2A.py`**, **`P2B.py`**, … Parts A-B are data generation, Parts C-E are numpy only, Part F is PyTorch.
- Submit your report and code files under **Project 2 Submission Area**, *as a single ZIP file*, which must include:
  - Your Python source files and other necessary files for me to generate your results.
  - Your Report PDF and source files (the entire Overleaf directory)

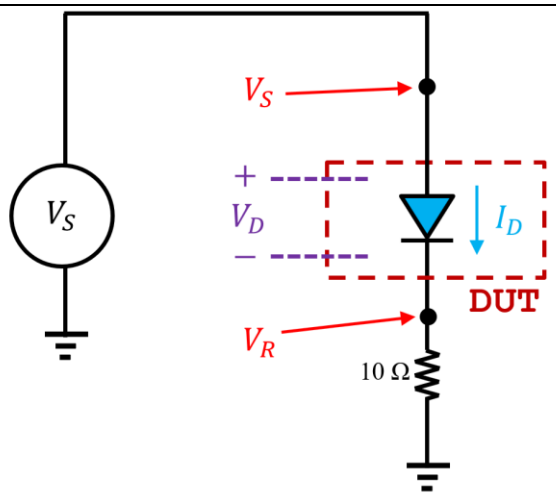
PROJECT 2 DESCRIPTION:

First, you need a dataset. Build your dataset using the I-V measurements of a diode of your own choosing. This is very simple by building the testbench shown on the right ➔

Diodes are characterized by the following formula:

$$I_D = I_S \cdot \left(e^{V_D/V_T} - 1\right)$$

where I_S is a constant, V_D is the voltage drop across the diode's terminals, and I_D is the current flowing through the diode. Notice that in this circuit, $V_D = V_S - V_R$.



DUT = Device Under Test

Dr. Tolga Soyata *© George Mason University, Department of ECE* *tsoyata@gmu.edu*

HOW TO GENERATE THE DATA THROUGH DIODE MEASUREMENTS:

You need 4 circuit elements to build the test bench shown previously:

1) **An adjustable power source**: You will need to sweep the voltage V_S from 0V to approximately 2V for diodes and from 0V to 4V for LEDs.
2) **A voltmeter**: The more accurate, the better. Simple 3.5 digit DMM is a little inadequate, but it will work. Ideally, you need a 4.5 digit one. I have one if you can't find it. Using the 10Ω resistor eliminates the need for a current meter. With the voltmeter, you will measure V_R and calculate the diode voltage and diode current as follows:

   $V_D = V_S - V_R$ $\quad\quad$ $I_D = V_R/10\Omega$

3) **A current limiting resistor**: Any resistor between 10-22 Ω is fine.
4) **Device Under Test (DUT)**: In our case, a diode. Use one of the following, which are widely available through our lab or I can give you one: 1N4001, 1N4007, 1N4148. Their voltage will usually be around 0.7-0.8V. Do not exceed 300mA for the 1N400x and do not exceed 100mA for 1N4148.
   You can use an LED as your DUT. Red, yellow, green LED voltages are around 2V and blue LED voltage is around 2.8V.
   Do not exceed 40mA for any LED. The higher the current, the brighter they will be until you read higher than 40mA, which will burn them.

You are allowed to use LTSpice to generate this data. Simply use a diode model from Digikey. To simulate the noise, you can simply emulate the DMM's quantization error and accuracy. You are welcome to add the noise in Python after you simulate the diode voltages in LTSpice.

There are published error metrics for each DMM. For example, my 4.5 digit DMM is Agilent U1272A: https://www.tme.eu/Document/fd88e63f97d358f31b140c4a61d1e1fb/U1271-90010.pdf

On Page 138, the voltage measurement accuracy is given as: 0.05%+2 when measuring at the 30V scale. Different scales may have a different error, but since the voltage you are measuring can be outside the range below this (0-3 V), it is a good idea to use the 0-30V range, where the measured voltages are in 0.0000 to 29.999 V. The reason the DMM is called 4.5 digits is because it has full 4 digits and the highest digit can only be (0, 1, or 2).

Assuming that you measure 2.7565 Volts, what is the expected noise?
The datasheet says 0.05%+2, which has two components:

- 0.05% is referring to the mean error, which is 2.7565 * 0.05 = 0.001378 V
- +2 is referring to the quantization error of the ADC, which is 2 units, implying that 2.7565 can be between 2.7563 and 2.7567.

To summarize, you will add two noise components, first of which is a normal-distributed multiplier anywhere from [-0.05% ... 0.05%]; call this **`noiseM`**. Second noise components is a uniform-distributed integer between -2 and +2, which adds between [-0.0002 … +0.0002], call this **noiseQ**;

So, the finalized noisy voltage measurement is:

$$v_{noisy} = v_{measured} \cdot (1 + noiseM) + noiseQ$$

Example: We measure 2.7565 Volts. What is the actual (noisy) voltage?

Assume our random number generator gave us noiseM= -0.02% and noiseQ=-2.

$$v_{noisy} = 2.7565\left(1 - \frac{0.02}{100}\right) + (-0.0002) = 2.7557487\ V$$

In other words, we would be plugging in 2.7557487 V as the training data.

PART A (1 point):

Measurement is very simple. Make an Excel sheet. Start Vs at 0V, increase it in small steps and keep recording V_D, I_D pairs until you reach the current limit of the diode. Get at least 40-50 data points. Program your Excel sheet to calculate the V_D, I_D, $\ln I_D$ triplets according to the formulas above. You also need to include a third column in your measurements, $\ln I_D$, which is the natural logarithm of the current. The reason for this third column will become clear shortly.

PART B (1 point):

Split your measurement data randomly into two pieces: training and validation, exactly the way described in the class notes. 80-20 split is fine; for example, if you have 40 measurements, you will have 32 and 8 data points in training vs. validation. As you will see shortly, we do not care about I_D. We only want the V_D, $\ln I_D$ pairs, where V_D is the independent variable (diode voltage) and $\ln I_D$ is the dependent variable (ln diode current).

PART C (2 points):

We know that the diode model is $I_D = I_S \cdot \left(e^{V_D/V_T} - 1\right)$. Since "1" is an enormously small number compared to e^{V_D/V_T}, a very reasonable simplification is $I_D \approx I_S \cdot e^{V_D/V_T}$

We are trying to find a relationship between V_D and I_D and it is exponential! We only know how to do linear regression. Check out the trick below: What if we took the natural log of both sides ?

$$\ln I_D \approx \ln I_S + \frac{V_D}{V_T} \quad \Rightarrow \quad y = b + \omega x$$

Your bias (b) is the log of the diode's saturation current (I_S), which is a constant. Your weight is $\left(\frac{1}{V_T}\right)$, which is a physical value. Now, we are back to exactly how we normally view a linear model ➔ $y = b + \omega x$. Here, the dependent variable y is the log current of the diode current, and the independent variable x is the diode voltage. Without this transformation, we would have to deal with a very ugly exponential model.

This means that we curve-fit $\ln I_D$ and use it to determine the exponential formula later by reversing the $\ln$ thing.

Using your training data, find the best linear regression curve fit model parameters using **numpy only**. When done, reverse the $\ln$ thing to determine a formula for your diode.

Your finalized formula should look like this $I_D \approx I_S \cdot e^{V_D/V_T}$. You will determine this formula by reversing the rick we applied and using the model parameters you determined (b, w).

Use a sufficient number of epochs that will yield a satisfactory error.. Report what you used.

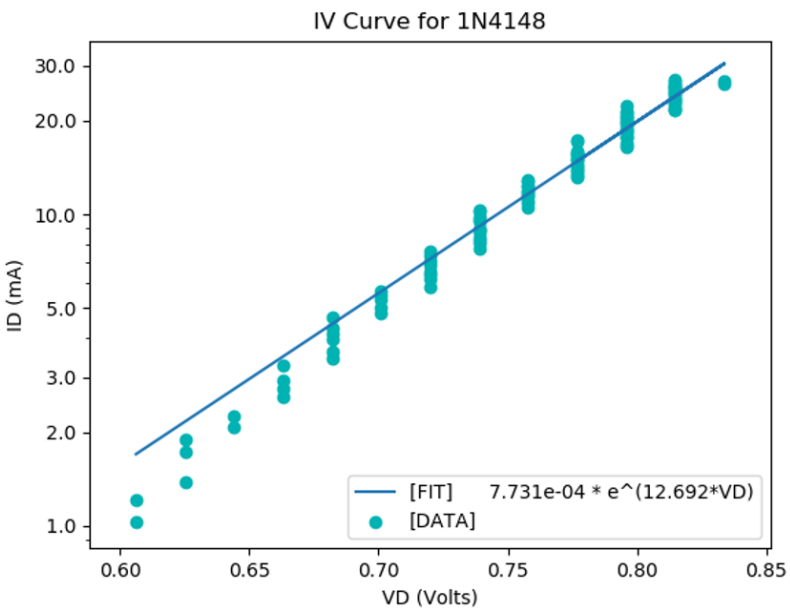
PART D (1 point):

- You will now validate your results in PART C. Use only the validation set to apply to your formula in PART C and determine validation errors and a validation loss.
- Note: validation only requires forward propagation, so, there is no "epochs." This is not in training mode; rather, in evaluation mode.

GEORGE MASON UNIVERSITY

PART E (2 points):

- Plot your measurement data, where the x axis is the Voltage in linear scale, and y axis is the current in log scale. This plot will be shown to the user, who doesn't care about your intermediate log trick! They want to see voltage vs. current. Draw your predicted model (the blue line below).

As an example, I am showing a diode I-V plot that I generated when I was teaching ECE350, which was in a lecture that asked the students to do the same experiments and plot their results in matplotlib. Use this only as a loose example, since your results could differ vastly. Furthermore, the measurements were done with an 8-bit ADC, which generated very low quality data points; your measurements should be drastically better; hence the reason for me to suggest a 4.5 digit DMM. Without an accurate DMM, you will see the multiple current measurements for the same voltage.	 1N4148 $I_D = 7.39 \cdot 10^{-4} \times e^{12.693 V_D}$

PART F (3 points):

Make a copy of your previous code and name it **`P2F.py`**. You will be converting your entire code into PyTorch. Use only PyTorch CPU tensors. Repeat Parts C-E (training and validation) in PyTorch and compare results to numpy-only results. You do not have to repeat the plots; instead, just reporting the results is fine.

GEORGE MASON UNIVERSITY