

**Progress Report (Summer 2024)**

**Jainil Kakka (10 Weeks – 20/06/2024 – 29/08/2024)**

# Timeline

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Week** | **Date** | **Tasks** |
| 1 | 1-2 | 20/06/2024 | Running and debugging the existing work and research done on the topic (code and notebook). |
| 2 | 25/06/2024 | Setting up PSU roar collab environment to run high computation neural networks |
| 3 | 27/06/2024 | Identifying Potential issues with the previous approach using images |
| 4 | 3-4 | 04/07/2024 | Using 2D vectors and Linear approach for GAN |
| 5 | 11/07/2024 | Using 2D vectors and Conv2D approach for GAN |
| 6 | 11/07/2024 | Using 3D vectors and Conv3D approach for GAN |
| 7 | 5-6 | 15/07/2024 | Creating data engineering pipelines to store 2D and 3D data in .npz format locally for fast processing. |
| 8 | 23/07/2024 | Converting data into derivatives to extract features and make images more readable. |
| 9 | 7-8 | 29/07/2024 | Creating visualizer file to create multiple visual techniques in the third dimension using matplotlib gallery |
| 10 | 12/08/2024 | Creating simple linear GAN to use a benchmark. |
| 11 | 9-10 | 15/08/2024 | Using multiple GAN techniques and tweaking models |
| 12 | 22/08/2024 | Least Squares GAN |
| 13 | 25/08/2024 | Relativistic GAN |

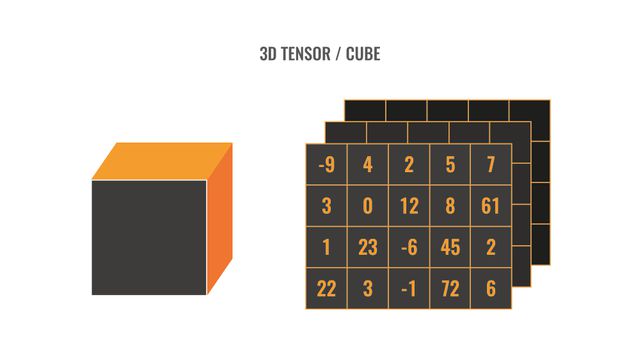
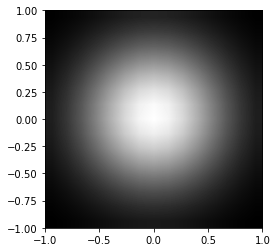
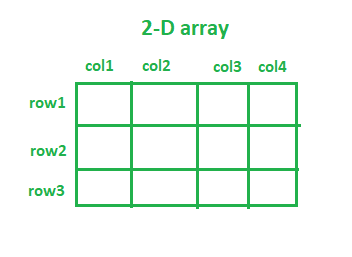
# Detailed Report

## Week 1-2

**Potential Issues with previous approach using images.**

There are some things I did not quite understand with the previous approach and the notebook, here is where I think it was lacking.

* We are converting the array generated using our plot\_function\_u (formula) to generate image (color image in the notebook).
* While we plot image the in the form of cmaps the final output is range invariant. If an array has a range of (-3,3) it will apply black pixel to -3 and white to +3.
* Hence be it any range, the information is lost over conversion of the array to image and then back to tensor.
* If the design of the neural network is to be range invariant, then this could work in favor.

## Week 3-4

**Using 2D vectors and Linear approach for GAN**

I used the classic GAN architecture using Linear neural networks and not convolution. I was able to do it with a 32x32 dimension array. I was able to get decent results, below are some generated images and loss over epochs. It can be improved using various techniques, but this is just a benchmark we could use for future approaches.

A group of white dots

Description automatically generated A group of squares with white spots

Description automatically generated

A light on a black background

Description automatically generatedA close-up of a white light

Description automatically generatedA red and blue dotted line

Description automatically generated

**Using 2D vectors and Conv2D approach for GAN**

Instead of using the approach of image storage and transformation I used the array to pass into convolution itself, there were better results, but the modelling needs a bit more work. A graph on a white sheet

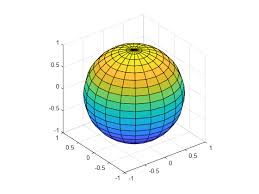
Description automatically generated

**Using 3D vectors and Conv3D approach for GAN**

In essence, I was able to create stacks of images for different x=[0,1]. The visual is shown below. A group of squares with yellow and green lights

Description automatically generated I was able to run conv3D using the above idea but the output did not resemble the input and the model had heavy losses and took time to run in general.

Instead I think we could use x = [0,1] as part of a supervised question where x could be the classes and conv3d could be used to find features contributing to x.

We could use 3d plotting tools or visuals to get such results in the data.  A light on a black background

Description automatically generated

I am not sure how we can achieve such data using plot\_function\_u but I think in general the distribution, or the plot generated here on the right is a 2d representation of a 3d object.

## Week 5-6

**Creation of data loader Numpy file directory for 465 replica files (32x32x1) images and (32x32x9) data for magnitude of the derivatives along x and y axis.**

Sample Image

A blurry image of a black and white image

Description automatically generated

## Week 7-8

**Creating visualizer file to create multiple visual techniques in the third dimension using matplotlib gallery, sample images are attached below.**

A graph of a colorful graph

Description automatically generated with medium confidenceA graph with a diagram

Description automatically generated with medium confidence

A graph of a cone

Description automatically generatedA graph of a graph

Description automatically generated

**Simple Linear Model for magnitude plot showed above, results and code are available (will work as a benchmark for future models)**

A grid of white squares

Description automatically generated

## Week 9-10

Identified a GitHub repository with 30+ GAN models, trying to read up and implement specific GANs that would work best for us.

We used more than 10 different types of GAN prebuilt models, based on which we could make custom GANs which would be tailored for our situation. Below is the exhaustive list of the models used and results of the best performing techniques

* Adversarial Autoencoder
* BEGAN
* Bicycle GAN
* Deep Convolution GAN
* Enhanced Super Resolution GAN
* Energy-Based GAN
* GAN
* Info GAN
* Least Squares GAN
* Relativistic GAN

Below are the results of best performing models

A group of squares with black border

Description automatically generated with medium confidence A grey square grid with black squares

Description automatically generated with medium confidence

Least Squares GAN Relativistic GAN