UC San Diego

Libraries and Tools:

Exploring various Linear Algebra operations offered by several libraries

Group number : 8

Tanvi Joshi, Sarthak Doshi, Sarvesh Khire, Vivek Sharma

Nov. 8, 2022

Sec 1: Introduction

Overview of the field/topic

- Linear algebra involves complex mathematical operations involving large multi-dimensional vectors and matrices
- Applications:
  - Machine Learning
  - NLP
  - Computer Vision
  - Deep learning
  - Data mining
- These tasks involve:
  - Huge amount of data
  - Computational power to process this data

Importance of the topic

- Several program libraries have been developed that allow computers to perform common operations in linear algebra.
- LAPACK was the first linear algebra library written in Fortran 90 that provides routines for solving systems of simultaneous linear equations eigenvalue problems, and singular value problems.
- New frameworks like NumPy, TensorFlow, PyTorch were developed
  - Provide high level of abstraction
  - User friendly syntax
  - Offer extensions for popular Machine learning frameworks.

Sec 2: Problem Formulation

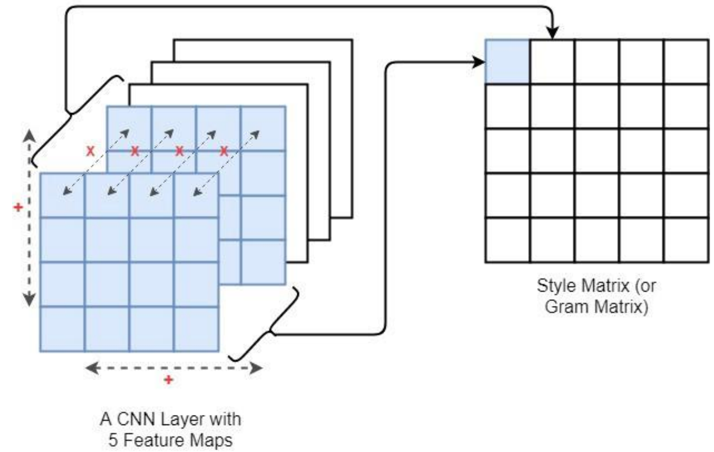
Problem formulation

- To explore and compare various frameworks and libraries that offer linear algebra operations.
  - Numpy
  - PyTorch
  - Tensorflow
- Comparison Factors
  - Time
- To assess the advantages and disadvantages of using these tools.
- To see the internal working of these libraries*

Approach of Numerical Linear Algebra (NLA)

Modern Approach

- Machine Learning Frameworks
- Hardware Support
- Operational Performance Enhancements



How the Style Matrix is Computed for a CNN Layer with 5 Feature Maps

Experiment setup and result

- Created random matrices of size in the range of 1 to 1000, implementing the NLA operations on all these matrices.
- Checking the time taken by each library as the matrix size increases.
- Changing the runtime to check differences in output with GPU and TPU Hardware accelerator.
- For small matrices, it was observed that all the Libraries had a similar computational speed
- The differences in their computation speed were observed as the size increased.
- It was observed that Pytorch had the fastest computation speed when GPU hardware accelerator was used.
- Similarly, TensorFlow was the fastest when TPU hardware accelerator was used.
- Numpy had the slowest efficiency for almost all the operations that we checked for.

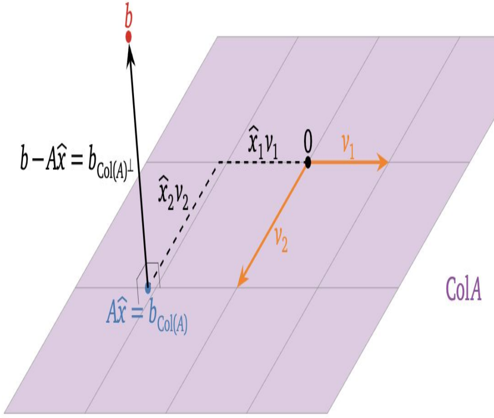
Sec 3: State of the Art (SOTA)

- **Least Square:**

Definition. Let A be an $m \times n$ matrix and let b be a vector in $\mathbf{R}^m$. A ***least-squares solution*** of the matrix equation $Ax = b$ is a vector $\widehat{x}$ in $\mathbf{R}^n$ such that

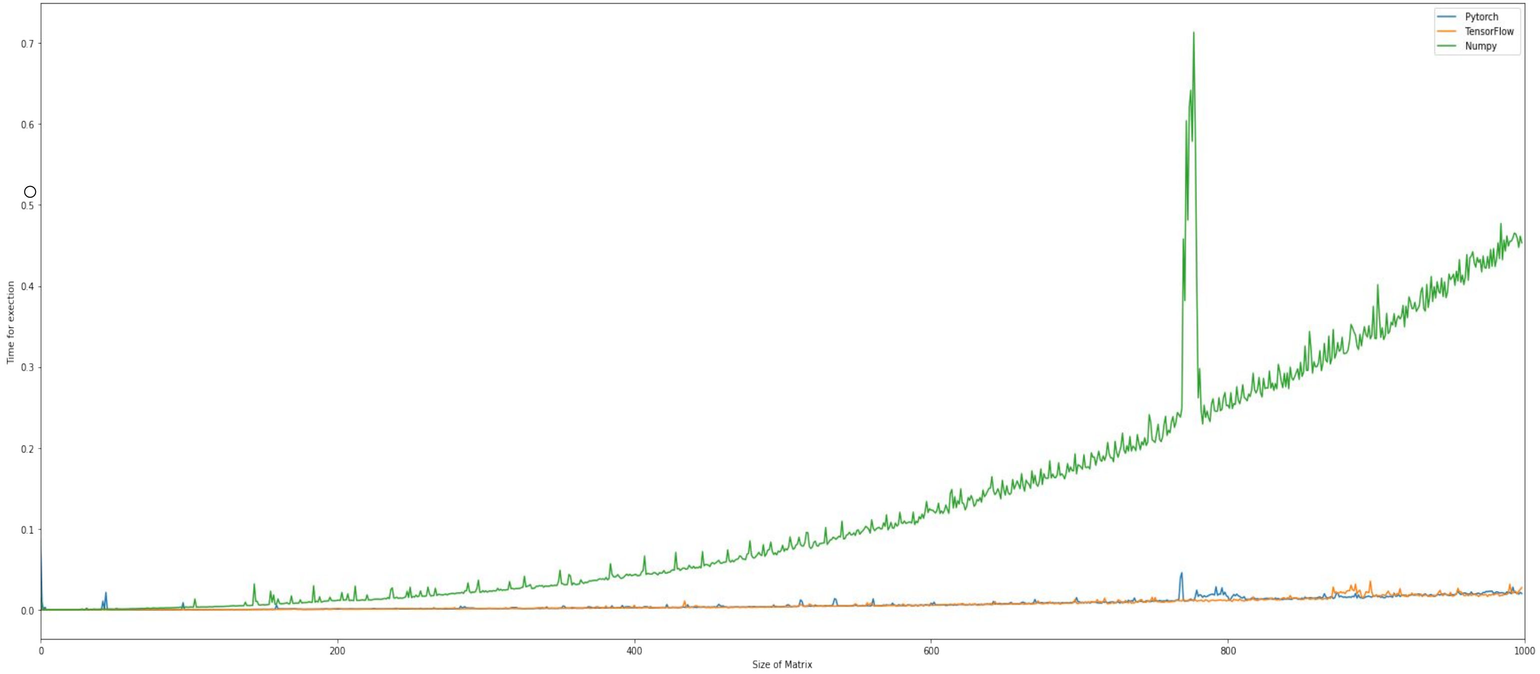
$$\text{dist}(b, A\widehat{x}) \leq \text{dist}(b, Ax)$$

for all other vectors x in $\mathbf{R}^n$.



- **Least Square:**

Comparing the three libraries: PyTorch vs. TensorFlow vs. Numpy

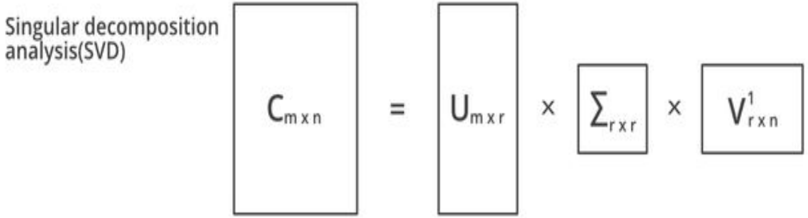


- **SVD(Singular Value Decomposition):**

Factorization of that matrix into three matrices.The SVD of mxn matrix A is given by the formula :

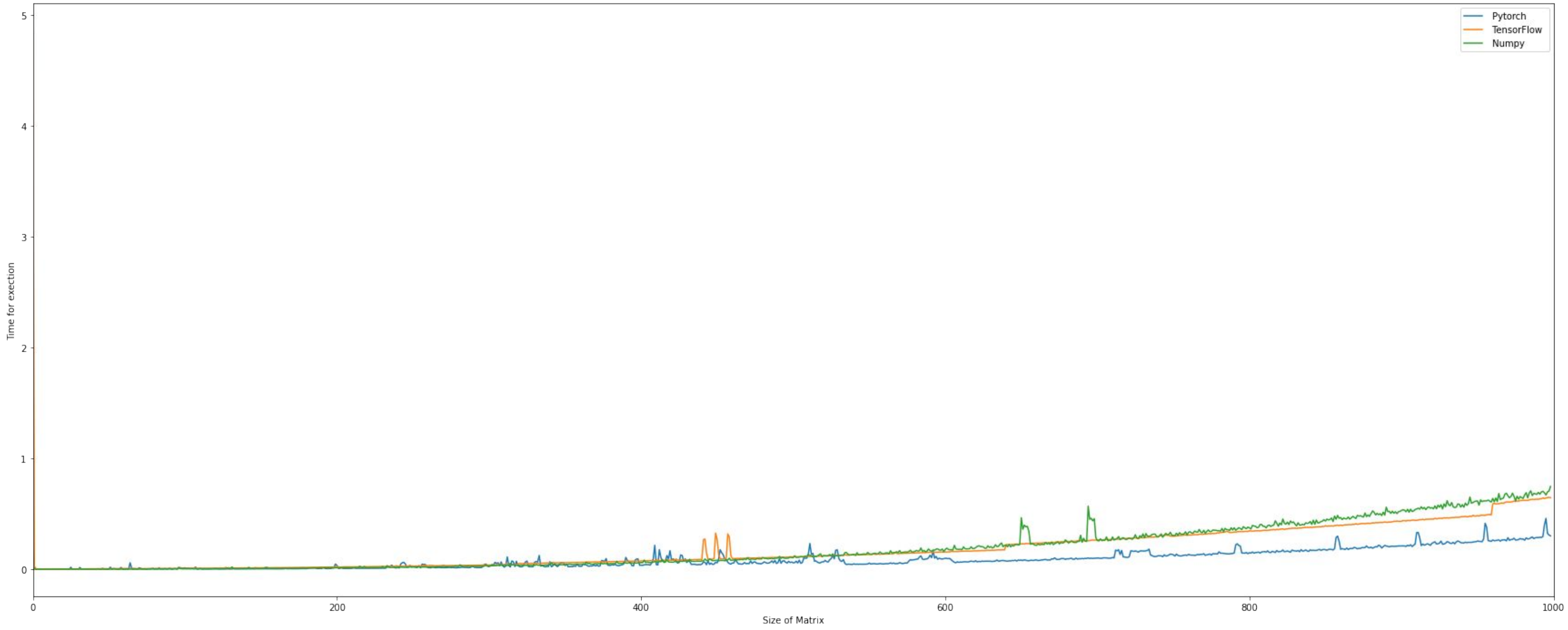
$$A = UWV^T$$

- U: mxn matrix of the orthonormal eigenvectors of AA^T
- V^T: transpose of a nxn matrix containing the orthonormal eigenvectors A^TA
- W: a nxn diagonal matrix of the singular values which are the square roots of the eigenvalues A^TA



- **SVD(Singular Value Decomposition):**

  Comparing the three libraries: PyTorch vs. TensorFlow vs. Numpy



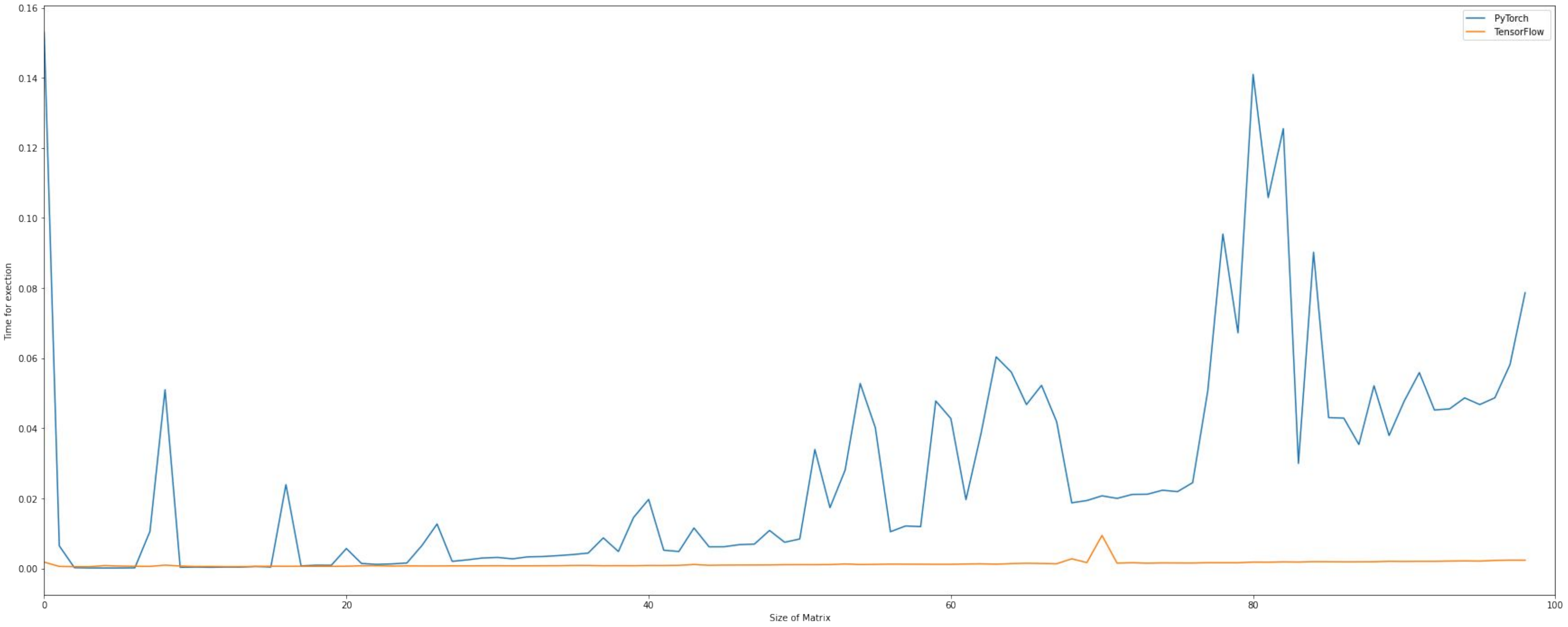
- **LU Decomposition(lower–upper):**

> L is lower triangular matrix.
> U is upper triangular matrix.
> M = LU is the LU decomposition of M.
> LU Decomposition in used to solve equations like : MX = LUX

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} = \begin{pmatrix} L_{11} & & \\ L_{21} & L_{22} & \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} & U_{13} \\ & U_{22} & U_{23} \\ & & U_{33} \end{pmatrix}$$

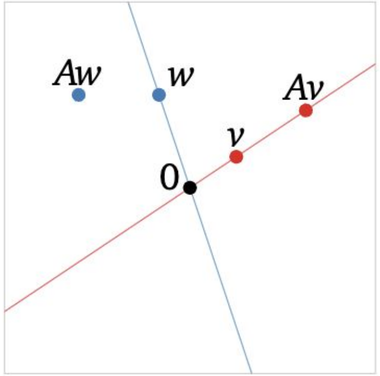
- **LU Decomposition(lower–upper):**

Comparing the three libraries: PyTorch vs. TensorFlow vs. Numpy



- **Eigenvalue Decomposition:**

1. An ***eigenvector*** of A is a *nonzero* vector v in $\mathbf{R}^n$ such that $Av = \lambda v$, for some scalar λ.
2. An ***eigenvalue*** of A is a scalar λ such that the equation $Av = \lambda v$ has a *nontrivial* solution.

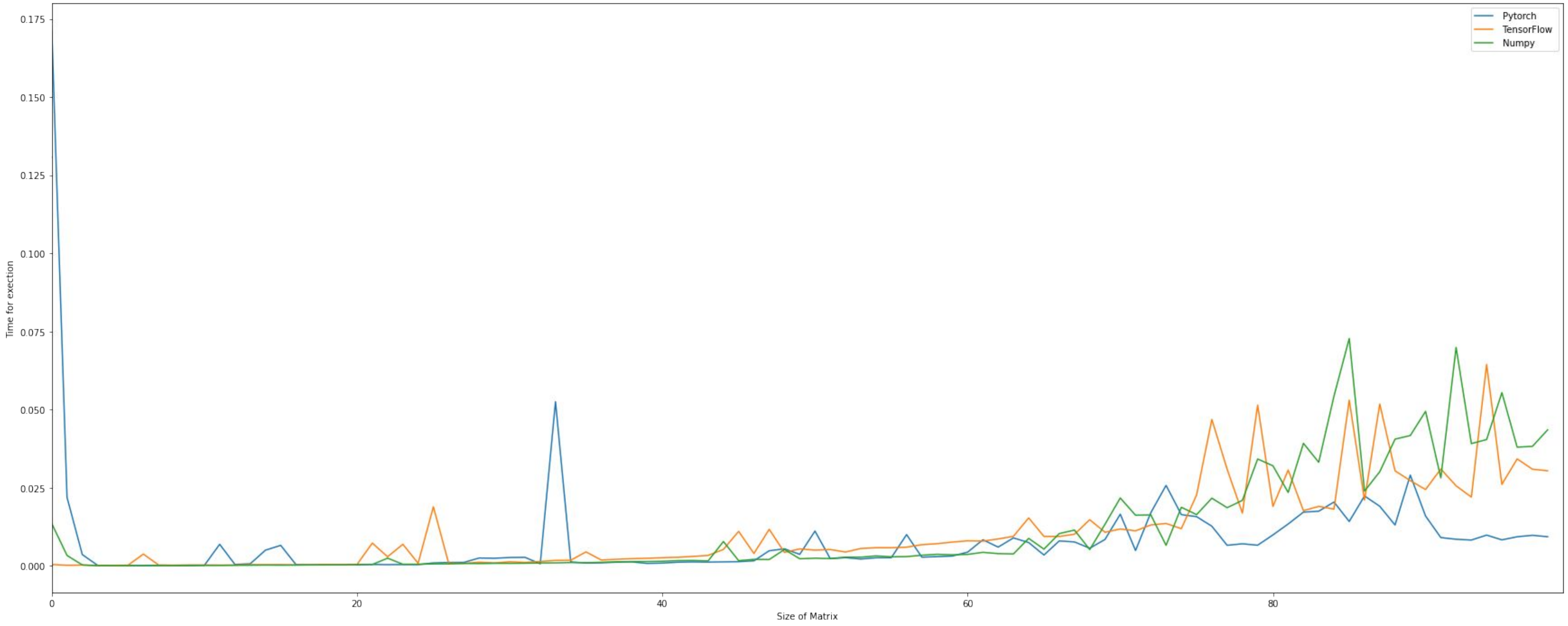


v is an eigenvector

w is not an eigenvector

- **Eigenvalue Decomposition:**

Comparing the three libraries: PyTorch vs. TensorFlow vs. Numpy



- **Norm:**

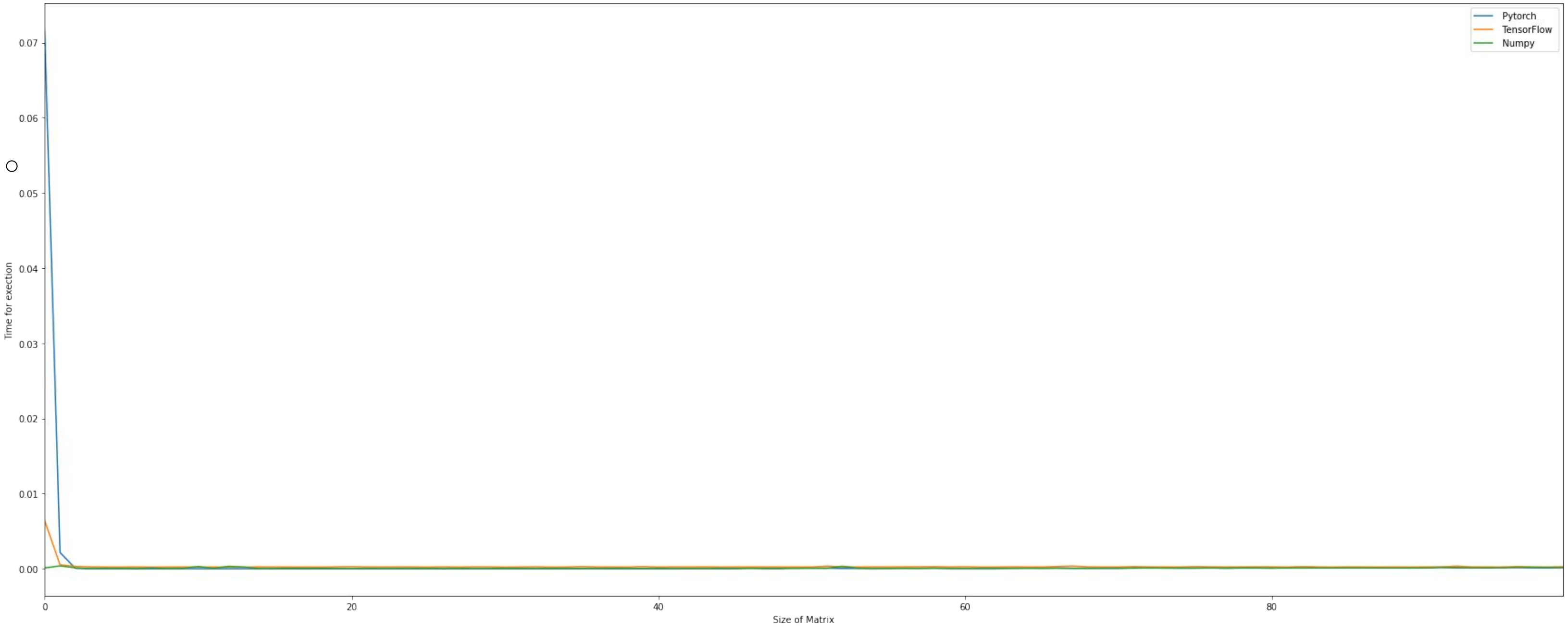
  - The essential notion of size and distance in a vector are captured by norms.
  - Norm is a function that returns length/size of any vector (except zero vector).
  - The norm gives the measure of magnitude.

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \cdots \\ u_n \end{bmatrix}$$

$$||u||_2 = \sqrt{u_1^2 + u_2^2 + \cdots + u_n^2}$$

- **Norm:**

Comparing the three libraries: PyTorch vs. TensorFlow vs. Numpy



Comparison Matrix:

Operation	PyTorch	TensorFlow	Numpy
Least Square	1 (similar to Tf)	1	3
SVD(Singular Value Decomposition)	1	2	3
LU Decomposition	3	1	-
Eigenvalue Decomposition	1	2	3
Norm	2	1	1

1:Fastest

3: Slowest

Sec 4: Concluding remarks

Conclusions

- This project involved exploring various Linear Algebra operations offered by several libraries.
- We compared various frameworks and libraries that offer linear algebra operations.
  - Numpy
  - PyTorch
  - Tensorflow
- We compared various Linear Algebra operations on varying data size and noted the computational speed offered by these libraries.

Conclusions

- We also explored how hardware acceleration has an effect on the working of each of these libraries.
- We ran our code on Google Colab and used the GPU,TPU Hardware acceleration.
- Finally, we visualized all the outputs that we received for better understanding of the results.

Conclusions

- John T Foster(2019) Matlab-vs-python -A rebuttal. https://johnfoster.pge.utexas.edu/blog/posts/matlab-vs-python/
- - Bryan Weber matlab-vs-python: Why and How to make the switch https://realpython.com/matlab-vs-python/
- - Fei-Fei Li,Justin Johnson,Serena Yung (2019) Hardware and software for convolutional neural networks http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture06.pdf
- - Numpy Developers (2018) Numpy for Linear Algebra networks https://numpy.org/doc/stable/reference/routines.linalg.html
- - The PyTorch Foundation(2022) Linear Algebra Operations https://pytorch.org/docs/stable/linalg.html
- - Google Developers Network(2019) TensorFlow Linear Algebra Operations https://www.tensorflow.org/api_docs/python/tf/linalg
- - Numpy Linear Algebra Library Documentation- Numpy.lin.alg
- https://numpy.org/doc/stable/reference/routines.linalg.html
- - TensorFlow LU Decomposition Function
- https://www.tensorflow.org/api_docs/python/tf/linalg/lu
- - PyTorch Common Linear Algebra Operations
- https://pytorch.org/docs/stable/linalg.html