NumCPP

By: Christopher Hainzl

Ramapo ID #R00736043

Contents

[Background / Introduction 2](#_Toc162164615)

[Installation Instructions (Double-Click for More Information) 4](#_Toc162164616)

[User’s Manual 4](#_Toc162164617)

[Design 6](#_Toc162164618)

[Code 8](#_Toc162164619)

[Test Plan and Test Results 11](#_Toc162164620)

[Summary and Conclusions 14](#_Toc162164621)

[References 15](#_Toc162164622)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Background / Introduction

NumPy is a Python module that comes equipped with functions for performing various array-oriented actions. Two of those actions include creating and manipulating arrays. The other key functionalities of NumPy involve performing logical and mathematical operations, including trigonometry (such as sine, cosine, and tangent), linear algebra, and statistics. NumPy also has functions for sorting arrays using specific algorithms, as well as searching for and counting values that meet certain criteria.

Some of the linear algebra operations that NumPy can perform include but are not limited to computing the row-reduced echelon form of a matrix, as well as the eigenvalues of a matrix. For a matrix to be in row-reduced echelon form means that it meets all of the following criteria: the first non-zero entry in any row is 1 (called the leading-one), and all of the entries above or below that leading-one are 0; in any two successive rows, not consisting entirely of zeros, the leading-one in the lower row appears further to the right than the leading-one in the row above it; all of the rows that consist entirely of zeros are at the bottom of the matrix (Viglino 7). An eigenvalue of a square matrix A (meaning that its row and column counts are equal) is a scalar value Y corresponding to a vector X that satisfies the following equation: AX = YX (Viglino 218).

Some of the statistics NumPy can compute include but are also not limited to the variance and standard deviation of a dataset. The variance of a dataset is the sum of squared deviations from its mean divided by the size of the dataset. The standard deviation is the positive square root of the variance (Peck et al. 162).

The purpose of this project was to implement most of NumPy’s key functionalities from scratch in C++. Doing this gave an understanding of the algorithms that are being run whenever the corresponding NumPy functions are called, as well as how much the implementation process varies when taking the syntax differences between Python and C++ into account.

# Installation Instructions (Double-Click for More Information)



## User’s Manual

The following are examples of valid declarations of Matrix instances with their data initialized:



WARNING: MAKE SURE THAT THE ROW VECTORS ARE ALL THE SAME LENGTH! OTHERWISE, THE MATRIX WILL BE DECLARED AS EMPTY!

These are examples of valid declarations of Matrix instances that do not have any data to begin with:



Below are examples of how to properly call NumCPP’s template functions that return another Matrix instance (some more examples of how to call NumCPP’s functions can be found in the “Test Plan and Test Results” section):

A screen shot of a computer code

Description automatically generated

This is an example of how to access a specific element in a Matrix instance:



## Design

Summary: Using a custom-made Matrix template class (to ensure that matrices containing values of any data type could be created; the full details on this are in the “Classes” section), as well as some of C++’s built-in data and file structures, most of NumPy’s key functionalities were successfully replicated from scratch. The main data structures used were a vector of vectors (for representing the matrix’s data), a pair, and a tuple (the pair and tuple were set as the return values for two of the functions; this will be explained further in the “Data & File Structures” section). The main file structures used were ostream, ifstream, istringstream, and ofstream (how they were all used will also be explained further in the “Data & File Structures” section).

Classes: The main class used while implementing this project was a Matrix template class. This allowed for the creation of not just matrices containing integers, but also matrices containing other numeric data types such as doubles and floats. An overview of the attributes and functions associated with this class can be observed in the following UML diagram.

|  |
| --- |
| Matrix |
| * m\_data * m\_rows * m\_cols |
| + Matrix() |
| + Matrix(vector<vector<T>> a\_vectors) |
| + ~Matrix() |
| + SetRows(int a\_num\_rows) : void |
| + SetCols(int a\_num\_cols) : void |
| + SetData(vector<vector<T>> a\_vectors) : void |
| + GetRows() : int |
| + GetCols() : int |
| + GetData(): vector<vector<T>> |

Data & File Structures:

* To help with storing the data of Matrix instances in a way such that the user can easily access them, a vector of vectors was used. For example, this creation of a Matrix instance is done with the help of a vector of vectors:

Matrix<int> matrix1 = Matrix<int>({{1, 2, 3}, {4, 5, 6}});

* To implement the function that computes the eigenvalue of a matrix in a way such that it can be easily understood by an entry-level software developer, a pair was set to be returned. The first element is the eigenvalue, and the second element is a Boolean value stating if the eigenvalue satisfies the equation that was described in the background section:

template <typename T>

pair<T, bool> Eigenvalue(Matrix<T> a\_matrix, Matrix<T> a\_eigenvector);

* To implement the function that finds the row and column indices of the non-zero elements in a matrix, a tuple containing two vectors of integers was used. The first vector contains the row indices, and the second vector contains the column indices:

template <typename T>

tuple<vector<int>, vector<int>> NonZero(Matrix<T> a\_matrix);

* To overload the “<<” operator so the data in a Matrix instance can be printed out easily, a reference to an “ostream” object was called upon. This allowed for the storage of the Matrix instance’s data inside that object.
* In the function that loads in a .txt file, the file structures used to help with the implementation were “ifstream” and “istringstream” (how they were used will be found in the section containing the code for the “Matrix.cpp” file).
* In the function that saves the contents of a Matrix instance as a CSV file, the file structure used was “ofstream” (how it was used will also be found in the section describing the code for the “Matrix.cpp” file).

## Code

NOTE: THE FILES ARE REPRESENTED AS EMBEDDED WORD DOCUMENTS. IF YOU WANT TO SEE THE FULL DETAILS, DOUBLE-CLICK ON THE DOCUMENTS.





## Test Plan and Test Results

This is the test program that was used to demonstrate that the “Matrix.h” and “Matrix.cpp” files, when combined, accomplish their specified tasks (DOUBLE-CLICK ON THE PROGRAM AND TEST RESULTS IF YOU WANT THE FULL DETAILS):



And these are screenshots of what the output of this test program looks like:



Here is a video of the test program in action:



## Summary and Conclusions

This project enables the performance of many of NumPy’s key functionalities in C++, such as creating & manipulating matrices, performing logical operations, and mathematical operations such as trigonometry, linear algebra, and statistics. It also allows the user to sort the data stored in matrices, as well as search for and count elements in a matrix that meet specific criteria. All of this was accomplished with the help of object-oriented programming, as part of working with this project’s functions involves creating instances of a custom-made Matrix class in C++.

What I learned from this project was not only how to implement the different algorithms used by NumPy from scratch, but also how to do so in C++. In addition, I learned how to work with other useful data structures in C++ besides vectors, such as tuples and pairs. This enabled me to further broaden my knowledge of writing programs in C++.

If I were to recreate NumPy in C++ again, the main thing that I would have done differently is use more of C++’s Standard Library to help me out. For example, the Standard Library contains a function named “inner\_product” that can compute the inner product of two vectors. Utilizing more functions from the Standard Library such as this would enable me to not only gain a stronger understanding of how helpful it can be when typing up C++ programs but also explain to entry-level software developers what is happening inside those corresponding functions.

## References

* Generative AI
  + GitHub Copilot
  + Microsoft Copilot
  + ChatGPT
  + Gemini
* Publications
  + Peck, Roxy, et al. *Introduction to Statistics & Data Analysis.* 3rd ed., Thomson Brooks/Cole, 2008.
  + Rajan, S., Wang, S., Inkol, R., & Joyal, A. (2006, May). Efficient Approximations for the Arctangent Function. IEEE Signal Processing Magazine, 108–111.
  + Viglino, Giovanni. *Linear Algebra: A Pathway to Abstract Mathematics.* 3rd ed., Ramapo College of New Jersey, 2023.
* Webpages
  + https://numpy.org/doc/stable/reference/constants.html
  + https://en.cppreference.com/w/cpp/types/numeric\_limits/infinity
  + https://en.cppreference.com/w/cpp/types/numeric\_limits/signaling\_NaN
  + https://www.geeksforgeeks.org/stdis\_same-template-in-c-with-examples/
  + https://gist.github.com/mortenpi/f20a93c8ed3ee7785e65
  + https://github.com/compute-io/Logspace/blob/master/lib/index.js
  + https://www.geeksforgeeks.org/how-to-find-common-ratio-with-first-and-last-terms/
  + https://www.geeksforgeeks.org/round-in-cpp/
  + https://stackoverflow.com/questions/6406356/how-to-write-vector-values-to-a-file
  + https://cplusplus.com/forum/general/5898/
  + https://github.com/pankajkompella/Hackerrank-SI-Basic/blob/master/Transpose%20matrix.c
  + https://stackoverflow.com/questions/571394/how-to-find-out-if-an-item-is-present-in-a-stdvector
  + https://www.geeksforgeeks.org/sort-c-stl/
  + https://www.geeksforgeeks.org/2d-vector-in-cpp-with-user-defined-size/
  + https://stackoverflow.com/questions/47930428/how-to-rotate-an-array-by-%c2%b1-180-in-an-efficient-way
  + https://www.codeproject.com/Tips/676287/C-and-Cplusplus-Use-isnan-to-Test-for-nan
  + https://www.geeksforgeeks.org/log-function-cpp/
  + https://www.geeksforgeeks.org/log2-function-in-c-with-examples/
  + https://www.geeksforgeeks.org/gcd-two-array-numbers/
  + https://www.geeksforgeeks.org/lcm-of-given-array-elements/
  + https://stackoverflow.com/questions/5707324/using-non-template-functions-in-template-classes?rq=3
  + https://www.geeksforgeeks.org/program-for-factorial-of-a-number/
  + https://www.geeksforgeeks.org/binomial-coefficient-dp-9/
  + https://www.codingconnect.net/cpp-program-sine-series/
  + https://www.codingconnect.net/cpp-program-cosine-series/
  + https://math.stackexchange.com/questions/421892/how-to-calc-arc-sine-without-a-calculator
  + https://proofwiki.org/wiki/Power\_Series\_Expansion\_for\_Real\_Arccosine\_Function
  + https://www.mathsisfun.com/sets/function-hyperbolic.html
  + https://numpy.org/doc/stable/reference/generated/numpy.tanh.html
  + https://btechgeeks.com/python-numpy-arcsinh-function/
  + https://btechgeeks.com/python-numpy-arccosh-function/
  + https://btechgeeks.com/python-numpy-arctanh-function/
  + https://www.geeksforgeeks.org/pair-in-cpp-stl/
  + https://en.cppreference.com/w/cpp/utility/pair/make\_pair
  + https://github.com/mmarinaa/Determinant-Calculator/blob/master/src/DeterminantCalc.java
  + https://rosettacode.org/wiki/Reduced\_row\_echelon\_form#C#
  + https://stackoverflow.com/questions/52490184/raising-a-matrix-to-the-nth-power
  + https://www.geeksforgeeks.org/program-for-mean-and-median-of-an-unsorted-array/
  + https://www.statology.org/numpy-average-vs-mean/
  + https://numpy.org/doc/stable/reference/generated/numpy.std.html
  + https://en.cppreference.com/w/cpp/algorithm/max\_element
  + https://en.cppreference.com/w/cpp/algorithm/min\_element
  + https://code.activestate.com/recipes/511478-finding-the-percentile-of-the-values/
  + https://big-o.io/examples/quicksort/c++/
  + https://big-o.io/examples/heapsort/c++/
  + https://big-o.io/examples/merge-sort/c++/
  + https://www.geeksforgeeks.org/timsort/
  + https://www.geeksforgeeks.org/tuples-in-c/