Document Loading Speed

In this exercise, we are comparing the loading speeds of the *.csv, *.npy, and *.hdf5 files based on a known set of data described in Table 1.

Table 1. Data Parameters to be Stored in Various Files. From Assignment 5 definition file.

Matrix	Min. value	Max. value	Filling	Order	Shape	Type
A	2	9	Arbitrary	Fortran	(5000 × 5000)	64-bit integer
B	100	127	Arbitrary	C	(5000 × 5000)	8-bit integer
C	0.33333	0.33333	Exact	C	(5000 × 5000)	8-byte float
D	1001	1100	Sequential	Fortran	(10,10)	2-byte integer
E	350.0	350.3	Sequential	C	(2,2)	4-byte float

This resulted in the following data that took a required time and size of memory to write, shown in Table 2.

Table 2. Matrices, Files, and Corresponding Write Times and Size.

Matrix	Filename	CPU Time (seconds)	File Size (bytes)	File Size (MB)
A	**A.csv**	2.97E+0	50.0E+6	50.0E+0
	A.npy	117E-3	200E+6	200E+0
B	**B.csv**	3.04E+0	100E+6	100E+0
	B.npy	16.9E-3	25.0E+6	25.0E+0
C	**C.csv**	12.0E+0	625E+6	625E+0
	C.npy	119E-3	200E+6	200E+0
D	**D.csv**	2.23E-3	500E+0	500E-6
	D.npy	3.24E-3	328E+0	328E-6
E	**E.csv**	1.77E-3	56.0E+0	56.0E-6
	E.npy	1.19E-3	144E+0	144E-6
HD5 Database	**Matrix_db.hdf5**	1.39E+0	35.6E+6	35.6E+0

There is clearly a strong trend between the inherent size of the data type, the size of the matrix, and the storage medium type, and the size of the file written.

However, this also leads to different loading times for retrieving the data, shown in Figure 1. These larger datasets take orders of magnitude longer to load than the smaller ones. This makes sense because this data

has more to read. Additionally, it is obvious that for large data, the *.csv format is inferior to the other data storage mediums in its speed.

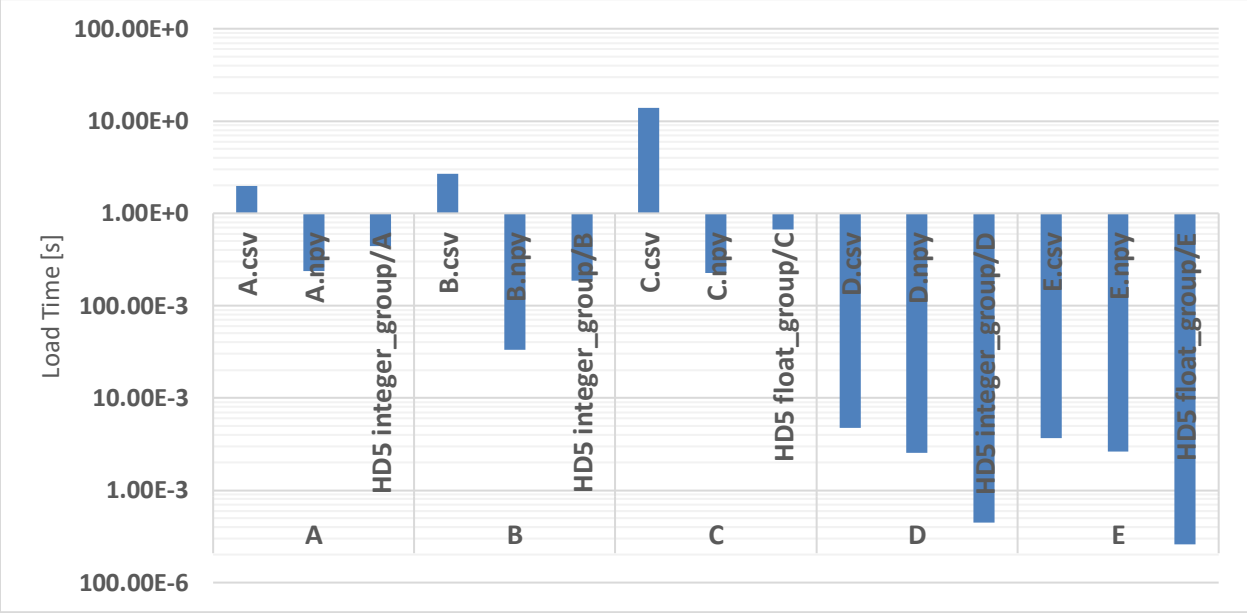

Figure 1. Loading Times for the Various Files. Data can be seen in Table 3 from the Appendix.

This trend is rather obvious. However, if the data load time is normalized to the size of the data, this reveals a more interesting trend. It is obvious that loading data in bulk takes less time relative to the data desired. Additionally, the advantage that the *.npy files have over the *.csv files in the raw load time is diminished, illustrated in Figure 2. This is where the speed of the *.hdf5 file takes dominance. If multiple file sets are desired, then clearly the HDF5 format can load all of them quicker than as individual sets. If we take this a step further and sum all the normalized load times across all data, the HDF5 file format presents a 205,000% improvement over the CSV format and a 70,800% improvement over the NPY format, shown in Table 4 in the Appendix.

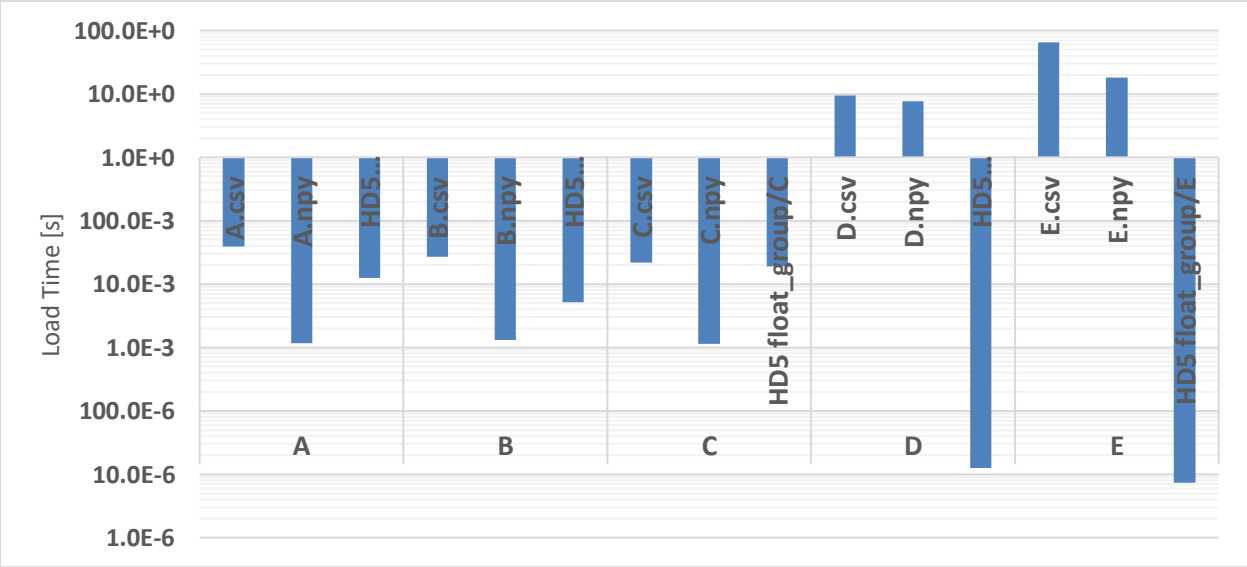

Figure 2. Normalized Loading Time Relative to the File Size. Data can be seen in Table 3 from the Appendix.

Appendix: Load Times

Table 3. Data Load Times and Normalized Load Times.

Matrix	File	Load Time (seconds)	Normalized Load Time [sec/MB]
A	A.csv	1.97E+0	39.4E-3
	A.npy	236E-3	1.18E-3
	HD5 integer_group/A	444E-3	12.5E-3
B	B.csv	2.68E+0	26.8E-3
	B.npy	33.1E-3	1.32E-3
	HD5 integer_group/B	186E-3	5.22E-3
C	C.csv	13.8E+0	22.1E-3
	C.npy	227E-3	1.13E-3
	HD5 float_group/C	669E-3	18.8E-3
D	D.csv	4.73E-3	9.47E+0
	D.npy	2.53E-3	7.71E+0
	HD5 integer_group/D	443E-6	12.5E-6
E	E.csv	3.66E-3	65.3E+0
	E.npy	2.61E-3	18.2E+0
	HD5 float_group/E	258E-6	7.27E-6

Table 4. Total Load Times Between Formats.

CSV Total Normalized Load Time [sec/MB]	
74.9E+0	
NPY Total Normalized Load Time [sec/MB]	
25.9E+0	
HDF5 Total Normalized Load Time [sec/MB]	
36.5E-3	
HDF5 Improvement	
CSV	NPY
204892%	70762%