Ps 3 Problem 2

Adam Guerin

September 26, 2023

Abstract

This document has an outline of a program that compares a triple for loop for calculating a matrix product versus the built in numpy dot product function.

1 Introduction

For loops are slow and not preferred in python, instead you should probably use arrays and numpy functions. We expect the runtime for matrix multiplication of square two dimensional arrays to scale as the size cubed when using for loops. This is because you must iterate over every element in the array you are saving to, and each of those iterations must compare N elements of each of the arrays being multiplied. Using built in numpy functions is much faster.

2 Methods

I calculated NxN matrix multiplication runtimes in two ways: using a triple for loop and using numppy.dot(). I timed using time.time() because I still can't figure out how to use timeit outside of an interactive python script.

3 Results

The for loop method does increase at the size cubed, but much more interestingly is numpy.dot() which while it does on average increase in run time with size. The increase in basically negligible with the time between the largest and shortest being a factor of 5, this ends up being 40 milliseconds with the for loop method having an increase of multiple seconds instead. See below for figure.

4 Discussion

Python doesn't handle for loops effectively try to avoid using them if you can help it because they scale very poorly. People have spent a lot of time making numpy very good instead.

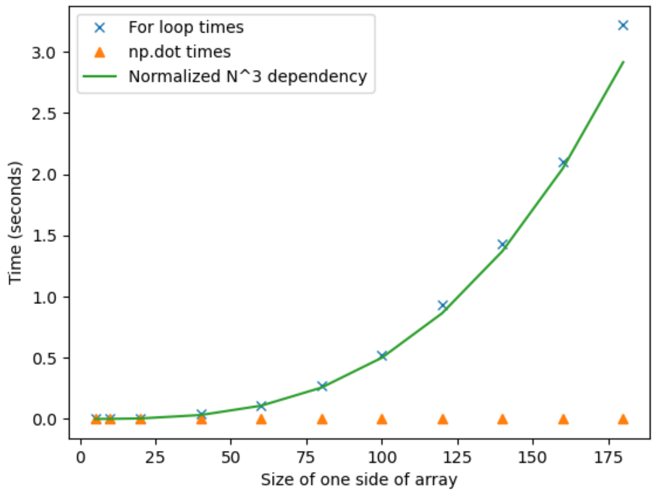

Figure 1: Run times for increasing sizes, the for loops follow an N^3 dependency, the numpy.dot() times are negligible in comparison.