Final project for the Course High Performance Programming, Philipp Noel von Bachmann

March 16, 2022

1 Introduction

Machine Learning and Data analysis are some of the most influencial fields in our current society. One of the main task in Machine Learning is to give a prediction based on some input variables. A common algorithm to use is linear-least-square, which finds the best fit for a linear model. Here, we need an efficient way to calculate this fit. In Data analysis, a common tool is the Principal Component analysis, which tries to reduce the dimensionality of the data in a way, that the reconstruction error gets minimized. PCA relies on an eigenvalue decomposition. For both methods, we can use the QR decomposition too speed up the computation.

2 Problem description

Suppose A is a real, square matrix. Then it can be shown that A can be decomposed into

$$A = QR \tag{1}$$

where Q is orthogonal and R is upper triangular. Finding Q and R is the task of the algorithm.

3 Solution

We will implement an algorithm known as **Givens Rotation**. This algorithm relies on construction a sequences of matrices G_i, such that when multiplying A with G_i, we get a new matrix with a zero at a predefined place. Choosing G_i such that we eliminate the lower diagonal of A, we end up with a R and by multiplying all G_i with Q. We will first show how to eliminate one value at the time by constructing G_i and then how to combine these.

3.1 Givens rotation

First, we define a Givens rotation matrix as

$$G(i,j,\theta) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c_{ii} & \cdots & -s_{ij} & & \\ & & \vdots & \ddots & \vdots & & \\ & & s_{ji} & \cdots & c_{jj} & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \tag{2}$$

where $c = \cos(\theta)$, $s = \sin(\theta)$ and any not filled out values are 0. Note: Therefore we can equally represent G by $G(i,j,c,s)$

3.2 Eliminating one value

We will how to find θ to solve

$$G(i,j,\theta)^T \begin{bmatrix} \times \\ \vdots \\ a \\ \vdots \\ b \\ \vdots \\ \times \end{bmatrix} = \begin{bmatrix} \times \\ \vdots \\ r \\ \vdots \\ 0 \\ \vdots \\ \times \end{bmatrix} \tag{3}$$

where $\times$ are arbitrary numbers $a, b \in \mathbb{R}$, $r = \sqrt{a^2 + b^2}$. A trivial solution would be

$$c = \frac{a}{r} \tag{4}$$

$$s = \frac{b}{r} \tag{5}$$

However, r is prone to overflow, so we can instead store it in a different way. If

- $|b| \geq |a|$:
  $t = \frac{a}{b}$, $s = \frac{sgn(b)}{\sqrt{1+t^2}}, c = st$
- else:
  $t = \frac{b}{a}$, $c = \frac{sgn(a)}{\sqrt{1+t^2}}, s = ct$

3.3 Final algorithm

Algorithm 1 Givens rotation

procedure STEP
  set $R = A$, $Q = I$
  for j in 1 to n **do**
    for i in n down to $j+1$ **do**
      compute the Givens rotation $G(i,j,c,s)$ eliminating R_{ij} with $a = i$, $b = i-1$.
      set $R = G(i,j,c,s)R$, $Q = QG(i,j,c,s)$

Note that we have to do it in this order, since each update affects row and row above.

4 Experiments

4.1 Check for correctness

4.2 Results

4.2.1 Format

In the foloowing section, we will show timing results for different matrix sizes. We denote the sizes by i, j, which we defined as decomposing the matrix $A \in \mathbb{R}^{i \times j}$.

4.2.2 Baseline Results

i \ j	10	25	50
10	0.0005	0.0008	0.001
25	0.018	0.034	0.049
50	0.235	0.611	1.076

We see that the if we vary i, we get a superlinear increase of runtime. If we increase j instead, we see that we get a nearly linear increase in runtime.

(We should only get a minor increase in runtime for j, since we dont need to calculate new 0s). (For i this makes sense, since increasing i by one adds i new 0s to compute.)

5 Random

- we can order them by affecting row and above
- best is probably to create tasks because upper rows require less work
- maybe need memory optimizations, but most of the operations seem to be near anyway

TODO:

- implement checks
- implement good storage of G
- implement efficient matmul of G
- implement one givens rotation
- implement outer loop
- serial optimizations
- parallelizations, probably with openmp

6 Experiments

7 Conclusion

8 References