DS298: Random Variates in Computation
Assignment 3

N Surya (22938)

April 18, 2024

Randomized solve for linear system of equations

$Ax = b$ for SPD systems are solved using three randomized linear solve algorithms namely:

1. Randomized Kaczmarz (RK)
2. CD-Least Square (CDLSQ)
3. CD-SPD

1. RK:

The number of scalar multiples per iteration is $2n + 1$ where n is the size of the matrix. This will be useful in estimating the computational effort.

$$x_{k+1}[i] = x_k[i] - A_{(i)}^T \left(A_{(i)} x_k - b_i \right) / ||A^{(i_{k-1})}||^2$$

2. CDLSQ:

The number of scalar multiples per iteration is $2n + 1$ where n is the size of the matrix.

$$x_{k+1}[i] = x_k[i] - A^{(i)^T} \left(A x_k - b \right) / ||A^{(i_{k-1})}||^2$$

Here, α_{k-1} is computed using a temporary vector ($Ax_0 = 0$) starting as all zeros and updating with α_{k-1}. As discussed in class, this might appear as if it is $O(n^2)$ due to the Ax term, but since only one index of x_k is to be changed, Ax need not be computed. The method discussed in class is utilized for updating x_{k+1}.

$$\begin{aligned} \alpha_{k-1} &= A^{(i_{k-1})^T} \left(A x_k - b \right) / ||A^{(i_{k-1})}||^2 \\ A x_k &= A x_{k-1} + \alpha_{k-1} A^{(i_{k-1})} \\ x_k &= x_{k-1} - \alpha_{k-1} e^{(i_{k-1})} \end{aligned}$$

3. CDSPD:

The number of scalar multiples per iteration is $n + 1$ where n is the size of the matrix.

$$x_{k+1}[i] = x_k[i] - \left(A_{(i)} x_k - b_i \right) / ||A^{(i_{k-1})}||^2$$

Plots:

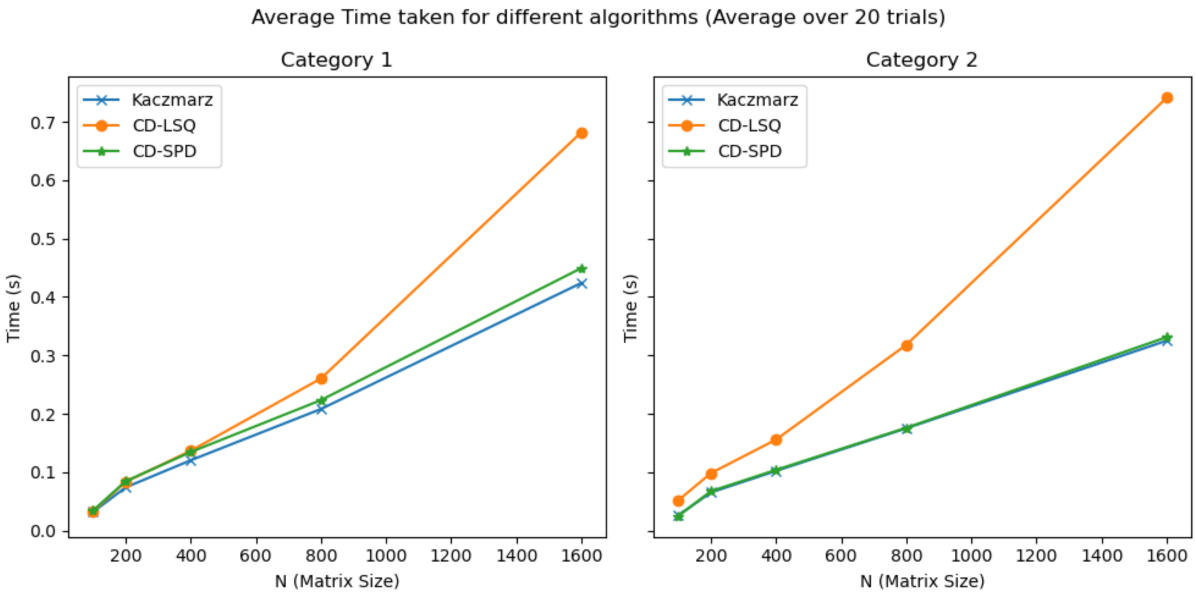

Fig. 1: Wall time

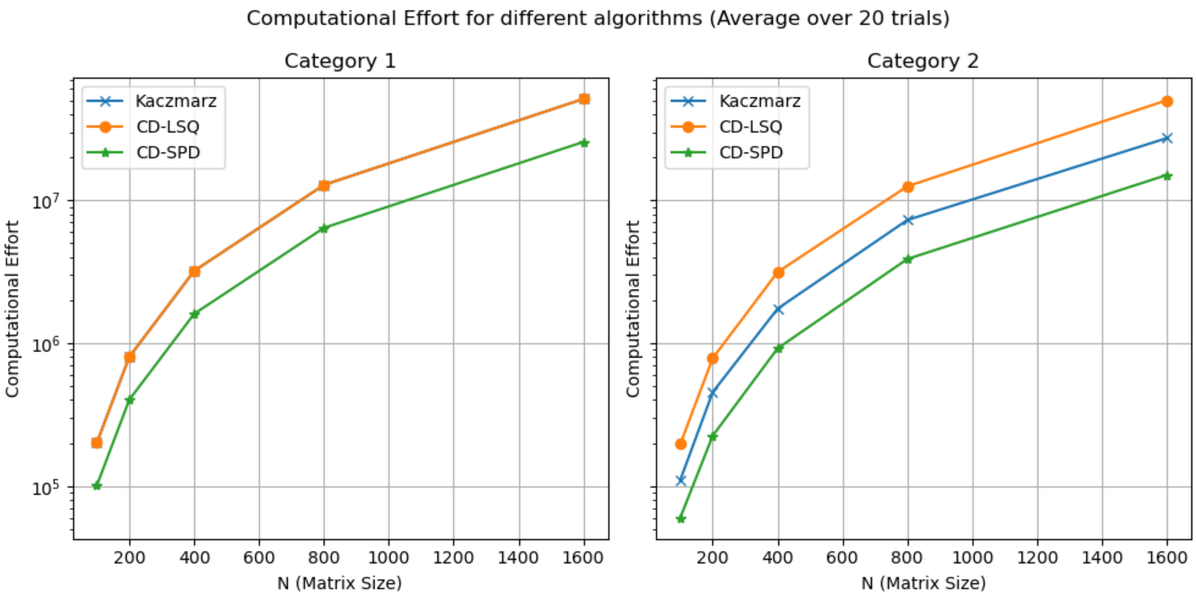

Fig. 2: Computational effort

Conclusions:

1. Class 2 matrices consistently converge within $O(n)$ iterations for all three methods.
2. Conversely, class 2 matrices display favorable conditioning, attributed to the distribution of their eigenvalues.
3. Despite using different methods, class 1 matrices consistently require $10n$ iterations to converge and exhibit no convergence behavior.
4. The high condition number of class 1 matrices renders them resistant to convergence across all methods.
5. The wall time (time per iteration) plot is following $O(n)$, although CDLSQ has a higher slope than the other algorithms.
6. The computational effort is also scaling as expected with CDLSQ having the highest amount of computational effort and correspondingly, higher wall time.
7. It can also be seen that CDSPD, the algorithm specifically designed for SPD matrix is the best algorithm for both of the categories in terms of compute effort and wall time.