Direct Methods for Solving Linear Systems

1) Matrix multiplication.

There are different ways to interpret the product of two matrices $\mathbf{AB}$. Each component of $\mathbf{C} = \mathbf{AB}$ can be viewed as the dot product of vectors $c_{ij} = \mathbf{A}(i,:)\mathbf{B}(:,j)$. Similarly, eacy column of $\mathbf{AB}$ can be interpreted as a linear combination of column vectors $\mathbf{A}(:,j)\mathbf{B}(j,k)$, $k = 1,\ldots,m$. Further $\mathbf{AB}$ can be represented as the summation of n vector outer products $\mathbf{A}(:,j)\mathbf{B}(j,:)$. This gives different ways to program the product of two matrices that are not entirely equivalent in CPU time. Compare these algorithms

2) Use the MATLAB Linear solver to compare the CPU time of the solution of full random matrices and lower triangular matrices of sizes $n = 1000 : 500 : 6000$.

3) **Gauss-Jordan**. Write a function to solve nonsingular linear systems using the Gauss-Jordan Method.

4) The Hilbert Matrix is defined as:

$$\mathbf{H}_n = (h_{ij}) = \frac{1}{i+j-1}$$

a) Use the Naive Gaussian elimination to solve the system

$$\mathbf{H}_5\mathbf{x} = \mathbf{b}$$

where

$$\mathbf{b} = (5, 3.550, 2.81428571428571, 2.34642857142857, 2.01746031746032)^T$$

The correct answer is:

$$\mathbf{x} = (1,2,3,4,5)^T.$$

b) Repeat the problem with $b(1) = 5.0001$. How much does the solution change?

5) Given the matrices:

$$\mathbf{T}_n = (t_{ij}) = \begin{cases} 4, & i = j \\ 1, & |i-j| = 1 \\ 0, & \text{else} \end{cases},$$

$$\mathbf{K}_n = (k_{ij}) = \begin{cases} 2, & i = j \\ -1, & |i-j| = 1 \\ 0, & \text{else} \end{cases}$$

and the Hilbert matrix $\mathbf{H}_n$, solve the systems

$$\mathbf{T}_n\mathbf{x} = \mathbf{b}$$

$$\mathbf{K}_n\mathbf{x} = \mathbf{b}$$

$$\mathbf{H}_n\mathbf{x} = \mathbf{b}$$

usign the Gauss Naive method, where $4 \le n \le 20$ and $\mathbf{b}$ is the vector:

$$\mathbf{b} = (1,1,\ldots,1)^T$$

of appropiate size. For each n, compute the value of $\max_{1\le i\le n}|r_i|$, where $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$. This vector is know as the **residual** of the numerical solution.

6) Use the LU factorization algorithm to solve the following systems of equations:

a) The system

$$\mathbf{T}_5\mathbf{x} = \mathbf{b}_1$$

where $\mathbf{b}$ is the vector

$$\mathbf{b} = (-1, 2, 4, 6, 13)^T.$$

b) The system

$$\mathbf{K}_5\mathbf{x} = \mathbf{b}_2$$

with

$$\mathbf{b}_2 = (-1, 0, 0, 0, 5)^T$$

c) The system

$$\mathbf{H}_5\mathbf{x} = \mathbf{b}_3$$

with

$$\mathbf{b}_3 = (163/60, 21/10, 241/140, 307/210, 641/504)$$

Compare your solutions with the correct answer:

$$\mathbf{x} = (0, 1, 2, 3, 4)^T.$$

7) Given a non-singular matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, we can compute its inverse as follows, first we obtain its $\mathbf{LU}$ decomposition. Then we obtain the inverse as

$$\mathbf{A}^{-1} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$$

where $\mathbf{x}_i$ are the solution of the systems

$$\mathbf{A}\mathbf{x}_i = \mathbf{e}_i$$

and $\mathbf{e}_i$ is the ith element in the canonical basis. Use this method to compute the inverse of the matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & -1 \\ -1 & 1 & 1 & 0 & 0 & 1 \\ 1 & -1 & 1 & 1 & 0 & -1 \\ -1 & 1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 \end{pmatrix}$$

The exact solution should be:

$$A^{-1} = \begin{pmatrix} \frac{1}{2} & \frac{1}{4} & -\frac{1}{8} & \frac{1}{16} & -\frac{1}{32} & \frac{1}{32} \\ 0 & \frac{1}{2} & \frac{1}{4} & -\frac{1}{8} & \frac{1}{16} & -\frac{1}{16} \\ 0 & 0 & \frac{1}{2} & \frac{1}{4} & -\frac{1}{8} & \frac{1}{8} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{4} & -\frac{1}{4} \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{4} & \frac{1}{8} & -\frac{1}{16} & \frac{1}{32} & -\frac{1}{32} \end{pmatrix}$$

8) Consider the matrix

$$\mathbf{A} = \begin{pmatrix} 1.8 & -3.8 & 0.7 & -3.7 \\ 0.7 & 2.1 & -2.6 & -2.8 \\ 7.3 & 8.1 & 1.7 & -4.9 \\ 1.9 & -4.3 & -4.9 & -4.7 \end{pmatrix}$$

Compute the inverse A^{-1}, solving the systems

$$\mathbf{Ax}_1 = \mathbf{e}_1,\ \mathbf{Ax}_2 = \mathbf{e}_2,\ \mathbf{Ax}_3 = \mathbf{e}_3,\ \mathbf{Ax}_4 = \mathbf{e}_4$$

where $\mathbf{e}_1,\ \mathbf{e}_2,\ \mathbf{e}_3,\ \mathbf{e}_4$ are the vectors of the canocial basis. Use the Gaussian Elimination algorithm

Repeat the analysis using the vandermonde matrix obtained with the MATLAB facility

$$vander(\mathbf{v})$$

with

$$\mathbf{v} = (1,2,3,4,\ldots,10).$$

9) Given the Hilbert matrix of size n :

$$\mathbf{H}_n = (h_{ij}) = \frac{1}{i+j-1}$$

find the inverse of $\mathbf{H}_n$ and the inverse of $\mathbf{H}_n^T\mathbf{H}_n$ for $n = 3,4,5.$ Then noting that

$$(H_n^T H_n)^{-1} = H_n^{-1}(H_n^{-1})^T$$

compare the accuracy of the results using the MATLAB command *invhilb* to find the accurate inverse

10) Compute the determinant of the matrix $\mathbf{A} = (a_{ij})$, where

$$a_{ij} = a_{ij-1} + a_{i-1j}$$

with $a_{1j} = 1 = a_{i1}$.

11) Solve the linear system

$$\begin{pmatrix} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -100 \\ 0 \\ 0 \\ -100 \\ 0 \\ 0 \\ -100 \end{pmatrix}$$

using Gaussian Elimination and the LDL^T factorization algorithm.

12) Compare the efficiency of the MATLAB solver with the function *chol* in order to solve linear systems of equations with symmetric positive definite matrices of different size.

13) Use the Cholesky decomposition to compute the inverse of the matrix

$$\mathbf{A} = \begin{pmatrix} 1 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & -1 & 2 & -1 & 0 \\ \vdots & \ddots & -1 & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{pmatrix}$$

The result should be the array

$$\mathbf{A}^{-1} = \begin{pmatrix} n & n-1 & n-2 & n-3 & \cdots & 3 & 2 & 1 \\ n-1 & n-1 & n-2 & n-3 & \cdots & 3 & 2 & 1 \\ n-2 & n-2 & n-2 & n-3 & \cdots & 3 & 2 & 1 \\ n-3 & n-3 & n-3 & n-3 & \cdots & 3 & 2 & 1 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ 3 & 3 & 3 & 3 & \ddots & 3 & 2 & 1 \\ 2 & 2 & 2 & 2 & \cdots & 2 & 2 & 1 \\ 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \end{pmatrix}$$