Norges teknisk–naturvitenskapelige universitet
Institutt for Matematiske Fag

[S]=T. Sauer, Numerical Analysis, Second International Edition, Pearson, 2014

"Teorioppgaver"

Cholesky faktorisering

1 Prøv å kjøre Cholesky faktorisering manuelt på så mange små matriser som dere kan; f.eks oppgavene 2.6.3-8.

Sparse matriser

2 Consider a tri-diagonal system $Ax = b$, where the matrix A is defined by $A_{i,i} = \alpha_i$, $i = 1, \ldots, n$; $A_{i,i+1} = \gamma$, $i = 1, \ldots, n-1$; $A_{i,i-1} = \beta_i$, $i = 2, \ldots, n$. Rest of the elements are zeros:

$$A = \begin{pmatrix} \alpha_1 & \gamma_1 & 0 & 0 & \dots & 0 \\ \beta_2 & \alpha_2 & \gamma_2 & 0 & \dots & 0 \\ 0 & \beta_3 & \alpha_3 & \gamma_3 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \beta_{n-1} & \alpha_{n-1} & \gamma_{n-1} \\ 0 & \dots & 0 & 0 & \beta_n & \alpha_n \end{pmatrix}$$

Let $LU = A$ be the LU-factorization of A (without pivoting; we assume that such a factorization exists). Describe explicitly the spartity structure (that is, the position of non-zero elements) in matrices L and U. Describe the algorithm for computing the LU-factorization of such a matrix. Further describe an algorithm for solving a linear system $Ax = b$ for the tri-diagonal matrix based on the previously computed LU factorization.

Iterative metoder

3 Oppgave 2.5.1

4 Oppgave 2.5.2

Systemer av ikke-lineære likninger

5 Oppgave 2.7.1

6 Oppgave 2.7.4

"Computeroppgaver"

Cholesky faktorisering

—

Sparse matriser

7 Implement a function for solving a tri-diagonal system of equations using LU-factorization in Python (see exercise 2). It should take three numpy-arrays α, β, γ, and b as inputs and produce the solution x as output.

Test your algorithm on some randomly generated tri-diagonal matrices (e.g., generate random arrays β and γ, and then generate a random α such that the resulting matrix is strictly diagonally dominant, hence also non-singular).

Compare the results produced by your algorithm with those produced by the sparse linear solver `scipy.sparse.linalg.spsolve`. (In order to do this you need to create a sparse tri-diagonal matrix. The easiest way to do this is to create it as `scipy.sparse.dia_matrix` and then convert it to `scipy.sparse.csr_matrix` format).

Iterative metoder

8 Oppgave 2.5.1-2.5.3

Systemer av ikke-lineære likninger

9 Oppgave 2.7.5