Exercises Lecture Computational Physics (Summer 2011)

Lecturers: Volker Springel & Rainer Spurzem

Tutors: Lei Liu & Justus Schneider

Sheet 7 (June 2, 2011)

1 Numerical linear algebra methods

- Consider the following matrix equation:

$$\begin{pmatrix} \epsilon & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 0.25 \end{pmatrix} \tag{1}$$

  where ϵ is a small number, say, $\epsilon = 10^{-6}$.

  - Solve the above system *numerically* by hand (or write a small program that does this) using either the Gauß-Jordan method or the Gaußian elimination and backsubstitution technique (your choice), but without pivoting. Use single precision, and take $\epsilon = 10^{-6}$ (if you prefer to take double precision, then use $\epsilon = 10^{-12}$). Check the result by back-substituting (x, y) into the above equation and checking if you get the correct right-hand-side, i.e. $(1, 0.25)$.
  - Do the same, but now with row-wise pivoting. What do you notice, compared to the previous attempt? How small can you make ϵ without running into precision problems?

- Solve the above equations using the Numerical Recipes routines `ludcmp` and `lubksb` (or equivalent subroutines for LU-decomposition and back-substitution from a library of your choice, e.g. the routine `gsl_linalg_LU_decomp` from GSL, the GNU Scientific Library), and check if the same results are obtained.

- Now calculate the determinant of the matrix

$$\begin{pmatrix} 3 & 2 & -2 & -3 & 3 \\ 1.5 & 1.5 & -1.2 & 2.5 & 3.5 \\ 12 & 8.125 & -8.55 & -8 & 9.5 \\ -6 & -5 & 3.9 & 5 & -11.5 \\ 0.75 & 0.6 & -0.29 & 6.55 & 7.65 \end{pmatrix} \tag{2}$$

  To this end, use the LU decomposition provided by the numerical library you employed for the previous point.

2 Tridiagonal matrices (homework)

Consider the following tridiagonal matrix equation:

$$\begin{pmatrix} b_1 & c_1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & a_{n-2} & b_{n-2} & c_{n-2} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & a_n & b_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ \vdots \\ r_{n-2} \\ r_{n-1} \\ r_n \end{pmatrix} \tag{3}$$

1. (3 pt) Derive the iterative expressions for Gaußian elimination, in a form that can be directly implemented as a numerical subroutine. Do *not* apply pivoting here[1].

2. (3 pt) Derive the iterative expressions for backward substitution, also for implementation as a numerical subroutine.

3. (10 pt) Program a subroutine that, given the values $a_2 \cdots a_n$, $b_1 \cdots b_n$, $c_1 \cdots c_{n-1}$ and $r_1 \cdots r_n$, finds the solution vector given by $x_1 \cdots x_n$.

4. (2 pt) Take $n = 15$, and set all a values to -1, all b values to 2, all c values to -1 and all r values to 0.2. What is the solution for the $x_1 \cdots x_n$?

5. (2 pt) Put your solution $x_1 \cdots x_n$ back into the original matrix equation (Eq.3) and find how much the result deviates from the original right-hand-side $r_1 \cdots r_n$. Is this satisfactory?

[1] It turns out that, in the *special case of tridiagonal matrix equations* pivoting is rarely necessary in practice; so we're lucky this time.