Numerical Linear Algebra
Computer Labs

Notes

- To pass this course you should do any **2** computer assignments presented here.
- You can work in groups by 2 persons.
- Sent final report with your assignment to my e-mail before deadline. Report should have description of used techniques, tables and figures confirming your investigations. Attach also corresponding programs for testing.

Computer exercise 1 (1 p.)

Apply Algorithm 1.1 (you can find this algorithm in the book BKK, see question 8.3, Algorithm 8.11) **Bisection**, to find the roots of the polynomial $p(x) = (x-1)^3(x-5)^2(x-7)^3 = 0$ as well as the roots **of some of your own polynomial**, where $p(x)$ is evaluated using Horner's rule.
Write your own matlab program. Confirm that changing the input interval for $x = [x_{low}, x_{high}]$ slightly changes the computed root drastically. Modify the algorithm to use the error bound in the text (the relative condition number for polynomial evaluation) to stop bisecting when the roundfoff error in the computed value of $p(x)$ gets so large that its sign cannot be determined. Present your results similarly with results of Figure 1.3 of the Demmel's book or Fig. 8.2, 8.3 of BKK book. Compare your results with results of Figure 1.3 of the Demmel's book or results of Fig. 8.2, 8.3 of BKK book.

Notes:
In Horner's rule a_d is the coefficient in the polynomial $p(x) = \sum_{i=0}^{d} a_i x^i$. Initialization: $p = a(d), bp = abs(a(d))$. You need first compute coefficients of $p(x)$. For example, exact coefficients of polynomial $p(x) = (x-9)^9$ are: a = [-387420489; 387420489; -172186884; 44641044; -7440174; 826686; -61236; 2916; -81; 1].

Compute error bound *bp* as in the algorithm on page 17 of the course book, see Matlab function below:

```
for i = d - 1:(-1):1
p = x*p + a(i);
bp = abs(x)*bp + abs(a(i));
end
bp = 2*(d - 1)*eps*bp;
```

Here, eps is the machine epsilon. Machine eps in matlab: eps = 2.220446049250313e-16;

Computer exercise 2 (1 p.)

Consider the nonlinear equation

$$y(T) = A \cdot \exp^{-\frac{E}{T - T_0}} .$$

Determine parameters A, E, T_0 which are positive constants by knowing T and output data $y(T)$.
Hint: Transform first the nonlinear function $y(T)$ to the linear one and solve then linear least squares problem. Discretize T by N points and compute discrete values of $y(T)$ as $y_i = y(T_i)$ for the known values of parameters A, E, T_0. Then forget about these parameters (we will call them exact parameters A^*, E^*, T_0^*) and solve the linear least squares problem using the method of normal equations (optionally QR decomposition, Algorithm 3.1) in order to recover these exact parameters.

Try add random noise δ to data $y(T)$ using the formula $y_\sigma(T) = y(T)(1 + \delta\alpha)$, where $\alpha \in (-1, 1)$ is randomly distributed number and δ is the noise level (if noise in data is 5%, then $\delta = 0.05$).
Analyze obtained results by computing the relative errors e_A, e_E, e_{T_0} in the computed parameters as:

$$e_A = \frac{|A - A^*|}{|A^*|},$$
$$e_E = \frac{|E - E^*|}{|E^*|}, \quad (1)$$
$$e_{T_0} = \frac{|T_0 - T_0^*|}{|T_0^*|}.$$

Here, A^*, E^*, T_0^* are exact values and A, E, T_0 are computed one.

Computer exercise 3 (2 p.)

Suppose that the nonlinear model function is given as

$$f(x, c) = A\mathrm{e}^{\mathrm{c_1 x}} + \mathrm{Be}^{\mathrm{c_2 x}}, \quad \mathrm{A, B} = \mathrm{const.} > 0, \tag{2}$$

and our goal is to fit this function using Gauss-Newton method. In other words, we want to use

$$c^{k+1} = c^k - [J^T(c^k)J(c^k)]^{-1}J^T(c^k)r(c^k), \tag{3}$$

where k is the number of iteration and $J(c^k)$ is the Jacobian matrix of the residual $r(c^k)$, for iterative update of $c = (c_1, c_2)$.
Try also to use Levenberg-Marquardt method. This method is based on the finding of minimum of the regularized function

$$F(c) = \frac{1}{2}r(c)^T r(c) + \frac{1}{2}\gamma(c-c_0)^T(c-c_0) = \frac{1}{2}\|r(c)\|_2^2 + \frac{1}{2}\gamma\|c-c_0\|_2^2, \tag{4}$$

where c_0 is a good initial guess for c and γ is a small regularization parameter. In the Levenberg-Marquardt method the linear system which should be solved at every iteration k is

$$(J^T(c^k)J(c^k) + \gamma^k I)(c^{k+1} - c^k) = -J^T(c^k)r(c^k). \tag{5}$$

Hint: we define the residual function

$$r(c) = y - f(x, c), \tag{6}$$

where $y = y_i, i = 1, ..., m$ are known data points. Use information in the previous comp.ex. to generate data $y = y_i, i = 1, ..., m$. Add random noise δ to data $y = y_i, i = 1, ..., m$ using the formula $y_\sigma(x, c) = f(x, c)(1 + \delta\alpha)$, where $\alpha \in (-1, 1)$ is randomly distributed number and δ is the noise level (if noise in data is 5%, then $\delta = 0.05$).

Test the Gauss-Newton and the Levenberg-Marquardt methods for different initial guesses for the parameter $c = (c_1, c_2)$ and different noise level δ in data $y = y_i, i = 1, ..., m$.

Analyze obtained results by computing the relative errors e_1, e_2 for computed parameters c_1, c_2 as:

$$e_i = \frac{|c_i - c_i^*|}{|c_i^*|}, \ \ i = 1, 2. \tag{7}$$

Here, (c_1^*, c_2^*) are exact values and $c = (c_1, c_2)$ are computed values.

Computer exercise 4 (1 p.)

The program Poisson2D_LU.m (available download from the course page together with programs DiscretePoisson2D.m, LU_factor.m, ForwSub.m, BackSub.m used by Poisson2D_LU.m) solves the Poisson's equation in two dimensions

$$-a\triangle u(x,y) = f(x,y) \tag{8}$$

on the unit square $\{(x,y) : 0 < x, y < 1\}$ with boundary conditions $u = 0$ on the boundary of this square using LU decomposition.
In (8) the coefficient $a(x,y)$ is such that

$$a(x,y) = 1 + A * exp(-((x-x_0)^2/(2*c_x^2) + (y-y_0)^2/(2*c_y^2))) \tag{9}$$

and we define the right hand side as

$$f(x,y) = A_f * exp(-((x-x_0)^2/(2*c_x^2) + (y-y_0)^2/(2*c_y^2))). \tag{10}$$

In (18), (19) functions A, A_f are the amplitudes of Gaussian, x_0, y_0 are constants which show the location of the center of the Gaussian functions, and c_x, c_y are constants which show spreading of the functions in x and y directions.

Use these programs and solve the Poisson's equation

$$-a(x,y,z)\triangle u(x,y,z) = f(x,y,z) \;\; in \;\; \Omega \tag{11}$$

in 3D on the unit cube $\Omega = [0,1] \times [0,1] \times [0,1]$ with boundary condition $u = 1$ on the whole boundary. Choose coefficient $a(x,y,z)$ as

$$a(x,y,z) = 1 + A * exp(-((x-x_0)^2/(2*c_x^2) + (y-y_0)^2/(2*c_y^2) + (z-z_0)^2/(2*c_z^2))), \tag{12}$$

and define the right hand side f of (11) as

$$f(x,y,z) = A_f * exp(-((x-x_0)^2/(2*c_x^2) + (y-y_0)^2/(2*c_y^2) + (z-z_0)^2/(2*z_y^2))). \tag{13}$$

Choose different amplitudes A, A_f and different constants $x_0, y_0, z_0, c_x, c_y, c_z$ (12), (13) .

Hint. We discretize the unit cube Ω with $x_{1i} = ih_1, x_{2j} = jh_2, x_{3k} = kh_3$, where

$$h_1 = \frac{1}{n_i - 1}, \quad h_2 = \frac{1}{n_j - 1}, \quad h_3 = \frac{1}{n_k - 1}$$

are the steps of the discrete finite difference mesh and n_i, n_j, n_k are number of discretization points in the directions x_1, x_2, x_3, respectively. Indexes (i, j, k) are such that $0 \leq i < n_i, 0 \leq j < n_j, 0 \leq j < n_k$. Global nodes numbers n_{glob} in the three dimensional case can be computed as

$$n_{glob} = j + n_j \left((i - 1) + n_i (k - 1) \right). \tag{14}$$

We take $n_i = n_j = n_k = n = N + 2$,
$h_1 = h_2 = h_3 = 1/(n-1) = 1/(N+1)$ and obtain the following scheme for the solution of Poisson's equation (11) in three dimensions:

$$-\frac{u_{i+1,j,k} - 2u_{i,j,k} + u_{i-1,j,k}}{h_1^2} - \frac{u_{i,j+1,k} - 2u_{i,j,k} + u_{i,j-1,k}}{h_2^2} - \frac{u_{i,j,k+1} - 2u_{i,j,k} + u_{i,j,k-1}}{h_3^2} = \frac{f_{i,j,k}}{a_{i,j,k}}, \tag{15}$$

where $u_{i,j,k}, f_{i,j,k}, a_{i,j,k}$ are values of u, f, a, respectively, at the discrete point n_{glob} with indices (i, j, k). We rewrite equation (15) with $h = h_1 = h_2 = h_3$ as

$$6u_{i,j,k} - u_{i+1,j,k} - u_{i-1,j,k} - u_{i,j+1,k} - u_{i,j-1,k} - u_{i,j,k+1} - u_{i,j,k-1} = h^2 \frac{f_{i,j,k}}{a_{i,j,k}}. \tag{16}$$

We recognize that scheme (16) is the system of linear equations $Au = b$. The matrix A is of the size $(n_i - 2)(n_j - 2)(n_k - 2) = N^3$ and on the unit cube is given by the block matrix

$$A = \left(\begin{array}{c|c|c|c|c} A_N & -I_N & O_N & -I_N & \ddots \\ \hline -I_N & A_N & -I_N & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ -I_N & \ddots & -I_N & A_N & -I_N \\ \hline \ddots & -I_N & O_N & -I_N & A_N \end{array}\right)$$

with zero-blocks O_N of order N. Now the blocks A_N of the size N-by-N on the main diagonal of this matrix are given by

$$A_N = \begin{pmatrix} 6 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 6 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 6 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 0 & -1 & 6 \end{pmatrix}.$$

Computer exercise 5 (2 p.)

Use the Matlab programs of comp.ex. 4 and solve the Poisson's equation in two dimensions

$$\begin{aligned} -a\triangle u(x,y) &= f(x,y) \ in \ \Omega, \\ u &= 1 \ on \ \partial\Omega \end{aligned} \tag{17}$$

on the L-shaped 2D domain Ω (choose this domain).
In (17) the coefficient $a(x,y)$ is such that

$$a(x,y) = 1 + A * exp(-((x-x_0)^2/(2*c_x^2) + (y-y_0)^2/(2*c_y^2))) \tag{18}$$

and we define the right hand side as

$$f(x,y) = A_f * exp(-((x-x_0)^2/(2*c_x^2) + (y-y_0)^2/(2*c_y^2))). \tag{19}$$