# Numerical Linear Algebra
# Computer Labs

## Notes

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

# Computer exercise 1 (1 p.)

**Solve question Q1.21 in the text book**

Question 1.21: Apply Algorithm 1.1 **Bisection**, to find the roots of the polynomial $p(x) = (x-9)^9 = 0$ 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 book. Compare your results with results of Figure 1.3 of the 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 $p(x) = (x-9)^9 = 0$. 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:

```
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 ( max 5 p.)

Consider the nonlinear model equation of some chemical reaction

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

Determine parameters $A, E, T_0$ which are positive constants by knowing $T$ (usually $T$ is a temperature in the model equation, in Kelvin) 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 $\delta$ 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 $\delta$ 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:

$$e_A = \frac{||A - A^*||_2}{||A^*||_2},$$
$$e_E = \frac{||E - E^*||_2}{||E^*||_2}, \quad (1)$$
$$e_{T_0} = \frac{||T_0 - T_0^*||_2}{||T_0^*||_2}.$$

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

# Computer exercise 3 (5 p.)

Suppose that the nonlinear model function is given as

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

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 $c = (c_1, c_2)$. We define the residual function

$$r(c) = y - f(x, c), \tag{4}$$

where $y = y_i, i = 1, ..., m$ are known data points.
Hint: use information in the previous comp.ex. to generate data $y = y_i, i = 1, ..., m$.

Add random noise $\delta$ 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 $\delta$ is the noise level (if noise in data is 5%, then $\delta = 0.05$).
Analyze obtained results by computing the relative errors $e_c$ in the computed parameters $c = (c_1, c_2)$ as:

$$e_c = \frac{||c - c^*||_2}{||c^*||_2}. \tag{5}$$

Here, $c^*$ are exact values and $c$ are computed one.

# 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

$$-a\triangle u(x,y) = f(x,y) \tag{6}$$

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

$$a(x,y) = 1 + A * exp(-((x-x_0)^2/(2*c_x^2) + (y-y_0)^2/(2*c_y^2))) \tag{7}$$

with values of the amplitude $A = 1,12$ and $c_x = 1, c_y = 1$.
Use this program and implement Algorithm 6.1 (Jacobi's method) of the text book instead of LU decomposition for the solution of (6).

Hint: In the program Poisson2D_LU.m we produce the mesh with the points $(x_i, y_j)$ such that $x_i = ih; y_j = jh$ with $h = 1/(N+1)$, where $N+1$ is the number of points in $x$ and $y$ directions. Then we derived the equation

$$a_{i,j}(4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1}) = h^2 f_{i,j} \qquad (8)$$

and wrote it as a single matrix equation in the form $Au = f$ with explicit entries of $A$.

# Computer exercise 5 (1 p.)

Use the program Poisson2D_LU.m of comp.ex. 4 and implement Algorithm 6.4 (Gauss-Seidel method with red-black ordering) of the text book instead of LU decomposition for the solution of (6). See details of problem (6) in comp.ex.4.

# Computer exercise 6 (1 p.)

Use the program Poisson2D_LU.m of comp.ex. 4 and implement Algorithm 6.6 (SOR($\omega$)) of the text book instead of LU decomposition for the solution of (6). See details of problem (6) in comp.ex.4.

# Computer exercise 7 ( 3 p.)

Use the program Poisson2D_LU.m of comp.ex. 4 and implement Algorithm 6.11 (Conjugate gradient algorithm) of the text book instead of LU decomposition for the solution of (6). See details of problem (6) in comp.ex.4.

# Computer exercise 8 (1 p.)

Use the program Poisson2D_LU.m of comp.ex. 4 and implement Algorithm 6.12 (Preconditioned conjugate gradient algorithm) of the text book instead of LU decomposition for the solution of (6). See details of problem (6) in comp.ex.4.