Linnaeus University
Department of Mathematics
Sofia Eriksson

Written Exam on Numerical Methods, 2MA903, 1 hp (5 hp)
Thursday 18th of March 2021, 08.00–13.00.

The solutions should be complete, correct, motivated, well structured and easy to follow.
Aids: Calculator (you may use a scientific calculator but *not* with internet connection).
Please begin each question on a new paper.
Preliminary grades: 15p-17p⇒E; 18p-20p⇒D; 21p-23p⇒C; 24p-26p⇒B; 27p-30p⇒A.

1. (a) Compute the two roots of the equation $x^2 + 200x - 10^{-10} = 0$ with 4 significant digitis.

   (b) Reformulate the function $g(x) = \sin x - x$ to avoid cancellation problems for x very close to zero. (4p)

2. (a) Use the Newton-Raphson method to find all solutions to the equation $x^3 - 6x^2 + 17 = 0$. Answer with 4 significant digits.

   (b) In case of double roots, it is better to use any of the modified versions of the Newton-Raphson methods shown below, that is

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}, \qquad \text{or} \qquad x_{n+1} = x_n - \frac{2f(x_n)}{f'(x_n)}.$$

   What is the numerical consequence if you keep using the original Newton-Raphson method instead of using one of the modified versions, and what characteristic of double roots is causing this problem? (5p)

3. Given the following set of points (x, y): $(-3, -5)$, $(1, 1)$, $(3, 14)$, $(4, 30)$:

   (a) determine the corresponding interpolating polynomial (of lowest possible degree),

   (b) if more points were to be added to the set of points, what would be the preferred method of finding the interpolating polynomial?

   (c) Next, consider the points

x	-4	-3	-2	-1	0	1	2	3	4
y	1	1	1	1	2	1	1	1	1

   Without doing any actual computations, make a sketch of the corresponding interpolating polynomial (together with the given points).
   The scales on the axes do not need to be very exact, but the essential behaviour of the polynomial should be clearly demonstrated. (5p)

4. (a) Use the (composite) trapezoidal rule to compute the integral

$$I = \int_{-1}^{1} \sqrt{1 - x^2}\, \mathrm{d}x$$

   for 3 different h ($h = 2$, $h = 1$ and $h = 1/2$). Use Romberg iteration to improve the approximation of the integral.

(b) Derive a finite difference formula D_h that uses the values $f(x)$, $f(x+h)$ and $f(x+2h)$ to approximate $f'(x)$. The formula should be as accurate as possible.

Find the error term and the order for the approximation formula D_h.

(c) Draw a figure showing the behaviour of the error $|D_h - f'(x)|$ as a function of h. The axes should be in log-log scale. (6p)

5. Let $y(x)$ be the solution of $y'(x) = y(1-y)$ for which $y(0) = 0.2$.

   (a) Find an approximation of $y(2)$ using Euler forward with step length $h = 1$ and another approximation using $h = 0.5$. Use 4 correctly rounded decimals of function values in the written presentation.

   (b) Sketch the corresponding slope field for $y'(x) = y(1-y)$. Include the two approximative solutions from (a), for $x \in [0, 2]$, in your slope field picture.

   c) Using Richardson extrapolation, calculate an improved approximation of y(2) using the results obtained in (a). (5)

6. A 10 cm long rod is exposed to a source of heat along its axis. The temperature $T(x)$ of the rod can be modelled with Poisson's equation

$$-\frac{d^2T}{dx^2} = f(x), \qquad\qquad x \in [0,\ 10],$$

   where $f(x) = x$ is the heat source. At the ends of the rod the temperature is fixed, such that the boundary conditions are $T(x = 0) = 40$ and $T(x = 10) = 200$.

   (a) Approximate the boundary value problem described above as a finite difference problem with step size $\Delta x = h = 2$, and present the resulting system of equations in matrix form.

   (You don't have to solve the system of equations.)

   (b) Reformulate the problem such that it can be solved using the Shooting method.

   (c) Write some Matlab code (or pseudo-code) that shows how the reformulated problem in (b) can be solved using the Shooting method. You may use/refer to existing Matlab functions such as `fzero`, `ode45` or similar. (6)

Good luck!

Linnaeus University
Department of Mathematics
Sofia Eriksson

List of formulas for the exam in Numerical Methods, 2021

These formulas will be attached to the exam. The list is not guaranteed to be complete, and the use, meaning, conditions and assumptions of the formulas are purposely left out.

- **Taylor's formula**

$$f(x) = f(a) + \tfrac{f'(a)}{1!}(x-a) + \tfrac{f''(a)}{2!}(x-a)^2 + \ldots + \tfrac{f^{(n)}(a)}{n!}(x-a)^n + \tfrac{f^{(n+1)}(\xi)}{(n+1)!}(x-a)^{n+1}$$

(ξ between x and a)

- **Absolute and relative error**

$$\Delta_x = \tilde{x} - x, \qquad \frac{\Delta_x}{x} \approx \frac{\Delta_x}{\tilde{x}}, \qquad \Delta_{x+y} = \Delta_x + \Delta_y, \qquad \frac{\Delta_{xy}}{xy} \approx \frac{\Delta_x}{x} + \frac{\Delta_y}{y}$$

- **Error propagation formulas, condition number (1D)**

$$\Delta f \approx f'(x)\Delta x, \qquad \left|\frac{\Delta f/f}{\Delta x/x}\right| \approx \left|\frac{xf'(x)}{f(x)}\right|$$

$$\Delta f \approx f''(x)\frac{\Delta x^2}{2}$$

- **Correct decimals**
  $|\Delta x| \leq 0.5 \cdot 10^{-t}$

- **Numbers in base B**
  $x = x_m B^m + x_{m-1}B^{m-1} + \ldots + x_0 B_0 + x_{-1}B^{-1} + \ldots = (x_m x_{m-1} \ldots x_0 . x_{-1} \ldots)_B$

- **Iterative methods**

Bisection method:

```
c=(a+b)/2;
while (b-a)>2*tol
    if f(c)*f(a)>0
        a=c;
    else
        b=c;
    end
    c=(a+b)/2;
end
```

Newton-Raphson: $$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \qquad J(\mathbf{x}_n)(\mathbf{x}_{n+1} - \mathbf{x}_n) + \mathbf{f}(\mathbf{x}_n) = 0$$

The secant method: $$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})},$$

$$e_n = x_n - x^*, \qquad |x_{n+1} - x^*| < \bar{c}|x_n - x^*|^p, \qquad \lim_{n\to\infty} \tfrac{|e_{n+1}|}{|e_n|^p} = c$$

- **Equation systems**
  $A\mathbf{x} = \mathbf{b}$, residual $\mathbf{r} = \mathbf{b} - A\widetilde{\mathbf{x}}$

LU-factorization:	$A = LU, \quad PA = LU$
QR-factorization:	$A = QR, \quad Q^T Q = I$
(Iterative methods)	$A = D + L + U$
Jacobi methods:	$\mathbf{x}^{(k)} = -D^{-1}(L+U)\mathbf{x}^{(k-1)} + D^{-1}\mathbf{b}$
Gauss-Seidel:	$\mathbf{x}^{(k)} = -(D+L)^{-1}U\mathbf{x}^{(k-1)} + (D+L)^{-1}\mathbf{b}$

Backward: $\|\mathbf{r}\|_\infty$, forward: $\|\mathbf{x} - \widetilde{\mathbf{x}}\|_\infty$

- **Norms and condition numbers**

Let $\mathbf{x} = (x_1, \dots, x_n)$ be a vector:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{1/p}, \quad \|\boldsymbol{x}\|_1 = \sum_{i=1}^n |x_i|, \quad \|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + \cdots + x_n^2}, \quad \|\mathbf{x}\|_\infty = \max_i |x_i|.$$

Let A be a $n \times n$ matrix:

$$\|A\| = \sup_{\mathbf{x} \neq 0} \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|}, \qquad \|A\mathbf{x}\| \le \|A\| \cdot \|\mathbf{x}\|, \qquad \kappa(A) = \|A\| \cdot \|A^{-1}\|$$

$A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \le \kappa(A) \frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}, \qquad econd(A) = \frac{\|\delta\mathbf{x}\|/\|\mathbf{x}\|}{\|\delta\mathbf{b}\|/\|\mathbf{b}\|} \le \kappa(A),$$

- **Interpolation**

Let $(x_0, y_0), \dots, (x_n, y_n)$ be $n+1$ points in the xy-plane.

Monomial:	$P(x) = a_0 + a_1 x + \ldots + a_n x^n$	
Lagrangre:	$P(x) = \sum_{j=0}^n y_j \ell_j(x),$	$\ell_j(x) = \prod_{\substack{0 \le m \le n \\ m \neq j}} \frac{x - x_m}{x_j - x_m}$

Newton's divided differences:

$$P(x) = [y_0] + [y_0, y_1](x - x_0) + \cdots + [y_0, \dots, y_k](x - x_0)(x - x_1) \cdots (x - x_{n-1}).$$
$$[y_0] = f(x_0), \quad [y_0, y_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \quad [y_0, y_1, y_2] = \frac{[y_1, y_2] - [y_0, y_1]}{x_2 - x_0}, \dots$$

Interpolation errors:

$$R(x) = f(x) - P(x) = (x - x_0)(x - x_1) \cdots (x - x_n) \frac{f^{n+1}(\xi)}{(n+1)!}, \qquad x_0 < \xi < x_n,$$

- **Least squares, normal equations** $A^T A\mathbf{x} = A^T\mathbf{b}$, residual $\mathbf{r} = \mathbf{b} - A\mathbf{x}$

- **Finite differences**

$$\frac{f(x+h) - f(x)}{h} = f'(x) + f''(\xi)\frac{h}{2} \qquad \xi \in [x, x+h]$$
$$\frac{f(x) - f(x-h)}{h} = f'(x) - f''(\xi)\frac{h}{2} \qquad \xi \in [x-h, x]$$
$$\frac{f(x+h) - f(x-h)}{2h} = f'(x) + f^{(3)}(\xi)\frac{h^2}{6} \qquad \xi \in [x-h, x+h]$$
$$\frac{f(x+h) - 2f(x) + f(x-h)}{h^2} = f''(x) + f^{(4)}(\xi)\frac{h^2}{12} \qquad \xi \in [x-h, x+h]$$

- **Trapezoidal rule, Simpson's rule**

$$\int_a^b f(x)dx = \frac{h}{2}\left(f(x_0) + 2\sum_{k=1}^{n-1} f(x_k) + f(x_n)\right) - \frac{(b-a)h^2}{12} f''(\xi), \qquad h = \frac{b-a}{n}$$

$$\int_a^b f(x)dx = \frac{h}{3}\left(f(x_0) + 4\sum_{k=1}^{n} f(x_{2k-1}) + 2\sum_{k=1}^{n-1} f(x_{2k}) + f(x_{2n})\right) - \frac{(b-a)h^4}{180} f^{(4)}(\xi), \qquad h = \frac{b-a}{2n}$$

$a < \xi < b$

- **Richardson extrapolation**
  $Q = F(h) + kh^n + \mathcal{O}(h^{n+1}), \qquad Q = \frac{2^n F(h/2) - F(h)}{2^n - 1} + \mathcal{O}(h^{n+1})$

- **Romberg** $R_{i,1} = T(h/2^{i-1})$, $R_{ij} = \frac{4^{j-1} R_{i,j-1} - R_{i-1,j-1}}{4^{j-1} - 1}$

- **Numerical solutions of differential equations**

Differential equation $y' = f(x,y)$ with initial condition $y(x_0) = y_0$

Euler forward ($g_i \sim \mathcal{O}(h)$): $\quad y_{n+1} = y_n + hf(x_n, y_n)$

Euler backward ($g_i \sim \mathcal{O}(h)$): $\quad y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$

Heun's method ($g_i \sim \mathcal{O}(h^2)$): $\quad \begin{cases} y_{n+1} = y_n + \frac{1}{2}(k_1 + k_2) \\ k_1 = hf(x_n, y_n) \\ k_2 = hf(x_n + h, y_n + k_1) \end{cases}$

RK4 ($g_i \sim \mathcal{O}(h^4)$): $\quad \begin{cases} y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 = hf(x_n, y_n) \\ k_2 = hf(x_n + h/2, y_n + k_1/2) \\ k_3 = hf(x_n + h/2, y_n + k_2/2) \\ k_4 = hf(x_n + h, y_n + k_3) \end{cases}$

where $x_{n+1} = x_n + h$.

- **Boundary value problems**

Two-point boundary problem $y'' = f(x, y, y')$ with initial condition $y(a) = \alpha$ and $y(b) = \beta$.

Shooting method: The BVP is rewritten as a IVP. Vary the modified initial condition until the boundary conditions are satisfied with desired accuracy.

Finite difference method: Derivatives are approximated by finite difference quotients. Exact solution y is replaced by y_i such that $y_i \approx y(x_i)$

- **Eigenvalue problems**

The power method: $\mathbf{v}_{k+1} = A\mathbf{v}_k / \|A\mathbf{v}_k\|$ and $\lambda_1 \approx \mathbf{v}_k^T A \mathbf{v}_k$.
The QR-method. Let $A = Q_0 R_0$ be a QR-decomposition of a real matrix A. Set $A_1 = R_0 Q_0$ and inductively (if $A_{n-1} = Q_{n-1} R_{n-1}$ is a QR-decomposition) $A_n = R_{n-1} Q_{n-1}$.