Fall 2017: Numerical Methods I
Assignment 6 (due Nov 30, 2017)

This assignment contains three sets of basic questions that check your understanding of the concepts from class. These questions require either a true/false, or another short answer. You should be able to do them quickly. Similar questions will also be part of the final. For each block of 10 questions, you will get 0.5 points if you have at least 6 correct answers, 1 points for 7, 1.5 for 8, 2 for 9, and 2.5 points for 10 correct answers.

1. **[Basic understanding questions on optimization and interpolation, 3×2.5 points]**

We consider $f : \mathbb{R}^n \to \mathbb{R}$ a twice continuously differentiable function, and are interested in solving $\min_{\boldsymbol{x}\in\mathbb{R}^n} f(x)$.

1	If f is convex, then the optimization problem has a global minimum.	
2	The Hessian matrix of f at a local minimum $\boldsymbol{x}^\star$ must be positive semidefinite.	
3	If f is bounded from below, the minimization problem can have multiple local minimia, but must have a unique global minimum.	
4	For $n = 2$ and $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$, give an example for A such that the minimization problem has no solution.	
5	For $n = 2$ and $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$ with $A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$, give an example for $\boldsymbol{b}$ such that the minimization problem does not have a solution.	
6	For $n = 2$ and $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$ with positive definite matrix A, the speed of convergence of the steepest descent method with linesearch depends crucially on the condition number of A.	
7	For $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$ with positive definite matrix A, Newton's method without line search always converges.	
8	An iterative minimization method with iterates $\boldsymbol{x}^0, \boldsymbol{x}^1, \ldots$ for which $f(\boldsymbol{x}^0) < f(\boldsymbol{x}^1) < \ldots$ holds, converges to a stationary point if f is bounded from below.	
9	An initial step length of 1 in backtracking is always a good choice for the steepest descent method.	
10	The Newton search direction at a point $\boldsymbol{x}$ is a descent direction if the Hessian matrix at $\boldsymbol{x}$ is positive definite.	

Denote by $P(f|t_0,\ldots,t_n)$ the Lagrange polynomial that interpolates $f \in C^2([a,b])$ at the nodes $t_0 < \ldots < t_n$. Denote by $[t_0,\ldots,t_k]f$ the k-th divided difference of f.

1	The Lagrange basis polynomial $L_0(t)$ for the nodes $t_0 = 0$, $t_1 = 1, t_2 = 2$ is $t^2 - 3t + 2$.	
2	If one uses the Lagrange basis for interpolation, one needs to solve a system with the Vandermonde matrix.	
3	The conditioning of polynomial interpolation depends on the node locations.	
4	The Lebesgue constant is only an upper bound for the absolute condition number of polynomial interpolation in the infinity norm—the actual conditioning may be smaller.	
5	Give the Newton polynomial $\omega_2(t)$ for the nodes $t_0 = 0, t_1 = 2, t_2 = 17$.	
6	Let $f(t) = 3t^3 - t + 2$. Then $[t_0,t_1,t_2,t_3]f = 9$.	
7	Let $f(t) = 8t^2 - t$. Then $[t_0,t_1,t_2,t_3]f = 0$.	
8	$P(f|t_0,\ldots,t_n)(t_j) = f(t_j)$ for $j = 0,\ldots,n$.	
9	Choosing the equidistant nodes $t_j = a + j(b-a)/n$ for $j = 0,\ldots,n$ is a good choice for polynomial interpolation.	
10	Let $t_0 = 0$, $t_1 = 3$ and $f(t_0) = 0$, $f(t_1) = 6$. What is the value of $P(f|t_0,t_1)(1)$?	

Denote by $P(f|t_0,\ldots,t_n)$ the Hermite interpolation of $f \in C^2([a,b])$ with nodes $t_0 \le \ldots \le t_n$, where for duplicated nodes additionally to the function value also the first derivative is interpolated, for twice duplicated nodes additionally the second derivative is interpolated, and so on. Denote by $[t_0,\ldots,t_k]f$ the k-th divided difference of f.

1	The polynomial $P(f|t_0,\ldots,t_n)$ is uniquely defined.	
2	If f is a polynomial of degree n, then $P(f|t_0,\ldots,t_n)(t) = f(t)$ for all $t \in [a,b]$.	
3	The divided differences are the coefficients of the interpolating polynomial for the monomial basis.	
4	For $t_0 = t_1 = t_2 = 0$ and $f(t) = \exp(2t)$, what is $[t_0,t_1,t_2]f$?	
5	For $t_0 = t_1 = t_2 = 1$, give the Newton polynomial $\omega_1(t)$.	
6	The cubic Hermite interpolation polynomial of $f(t) = \cos(t)$ for the nodes $t_0 = t_1 = 0$, $t_2 = t_3 = 2\pi$ is $P(f|t_0,t_1,t_2,t_3) = 1$.	
7	For $t_0 = t_1 = 0$, $t_2 = t_3 = 1$ and $f(t) = 3t^3 - 5$, $[t_0,t_1,t_2,t_3]f = 9$.	
8	In general, $[t_0,t_1,t_2]f \neq [t_2,t_0,t_1]f$.	
9	If $t_0 \neq t_n$, then $[t_0,\ldots,t_n]f = ([t_1,\ldots,t_n]f - [t_0,\ldots,t_{n-1}]f)/(t_n - t_0)$.	
10	If $f \in C^\infty([(a,b)])$ and one increases the number n of nodes, then the interpolating polynomial becomes a better and better approximation of f.	

2. **[3-term recursion and orthogonal polynomials, 2+2pt]** We consider the polynomial recursion $l_0(x) = 1$, $l_1(x) = 1 - x$, and

$$l_{k+1}(x) = \frac{2k+1-x}{k+1} l_k(x) - \frac{k}{k+1} l_{k-1}(x) \quad \text{for } k = 1, 2, \dots$$

   (a) Derive the polynomials for $l_2(x)$ and $l_3(x)$ from the recursion.[1] Verify[2] that

$$\int_0^\infty \exp(-t) l_i(t) l_j(t)\, dt = 0 \quad \text{for } 0 \leq i < j \leq 3.$$

   Since this orthogonality relation holds for all $i \neq j$ (you do not need to show that), the polynomials $l_i()$ are orthogonal on $[0, \infty)$ with weight function $\omega(t) = \exp(-t)$.

   (b) Write a function `my_l(k, x)`, which returns $l_k(x)$. The function should also allow vector inputs $\boldsymbol{x} = (x_1, \dots, x_n)$ and return $(l_k(x_1), \dots, l_k(x_n))$. Your function should not derive the polynomials analytically, but just compute their value at the points $\boldsymbol{x}$ using the recursion. Hand in a code listing, and plot graphs of the first several polynomials l_i (these polynomials are called Laguerre polynomials).

3. **[Chebyshev polynomials, 1+1+1+1+1+2+2pt]** The recurrence relation for Chebyshev polynomials T_k is $T_0(x) = 1$, $T_1(x) = x$ and $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$. Show from this relation that:

   (a) For even k, T_k is symmetric, i.e., $T_k(-x) = T_k(x)$. For odd k, T_k satisfies $T_k(-x) = -T_k(x)$.

   (b) By showing that $T_k'(x) = \cos(k \arccos(x))$ satisfies this 3-term recurrence relation, argue that that $T(x) = T'(x)$. Note that this implies that $|T_k(x)| \leq 1$ for all x.

   (c) By showing that

$$T_k''(x) = \frac{1}{2}\left((x + \sqrt{x^2-1})^k + (x - \sqrt{x^2-1})^k\right)$$

   satisfies the same 3-term recurrence relation, argue that $T''(x) = T(x)$. Note that differently from $T'(x)$, the form $T''(x)$ is defined for all $x \in \mathbb{R}$.

   (d) Show that the zeros of $T_k(x)$ are given by

$$x_j = \cos\left(\frac{(2j+1)\pi}{2k}\right) \qquad \text{for } j = 0, \dots, k-1.$$

   These x_j are called Chebyshev nodes (corresponding to the k-th Chebyshev polynomial).

[1] Note that in this recursion the polynomials are not normalized as in the problems we discussed in class, where the leading coefficient was assumed to be 1. Such a (or another) normalization can easily be achieved by multiplication of $L_i(x)$ with an appropriate scaling factor.

[2] Feel free to look up the values for the indefinite integrals $\int_0^\infty \exp(-t)t^k\, dx$ ($k = 0, 1, 2, 3$)—I use Wolfram Alpha for looking up things like that: http://www.wolframalpha.com/.

(e) The Newton polynomial ω_{n+1} that is based on Chebyshev nodes (i.e., the roots of T_{n+1}) satisfies

$$|\omega_{n+1}(x)| \leq \frac{1}{2^n} \quad \text{for all } x \in [-1,1]. \tag{1}$$

Hint: Think about what the difference is between the Newton polynomial with node points given by the roots of the Chebyshev polynomial T_{n+1}, and the Chebyshev polynomial T_{n+1} itself. Note that it can be shown that the Chebyshev nodes minimize the pointwise bound given by (1), which is referred to as the min-max property of the Chebyshev nodes and explains their usefulness for interpolation.

(f) Show that this implies the following error estimate for the interpolation error $E_f(x) := f(x) - p_n(x)$, where $p_n \in \boldsymbol{P}_n$ is a polynomial that interpolates a function $f \in C^{n+1}$ at the Chebyshev nodes $x_0, \ldots, x_n$:

$$|E_f(x)| \leq \frac{1}{2^n(n+1)!}\|f^{(n+1)}\|_\infty, \tag{2}$$

where for continuous functions g, $\|g\|_\infty = \|g\|_{C^0([-1,1])} := \max_{-1\leq t\leq 1}|g(t)|$ is the supremum norm on the interval $[-1,1]$.

(g) Compute and plot the approximation of the function $f(x) = 1/(1+12x^2)$ with a polynomial of order $N = 14$ on the interval $[-1,1]$. Choose $N+1$ equidistant nodes (including the points ± 1) and the Chebyshev nodes given by the roots of $T_{N+1}(x)$ as interpolation points, and compare the results.[3] What do you observe?

(h) Repeat the Chebyshev point-based interpolation with at least two larger numbers N of Chebyshev interpolation points, and approximate the maximal interpolation error.[4] Plot the maximal error as a function of N and compare with the estimate (2) (without evaluating (2) explicitly, just in terms of how it changes as you change N). Use a logarithmic scale for plotting the error (in MATLAB, you can use `semilogy`). Can you spot a trend for the error?

4. **[Recursion formula for divided differences, 2+1pt]** For nodes $t_0 \leq t_1 \leq \ldots \leq t_n$ in $[a,b]$ we consider Hermite interpolation of a function $f \in C([a,b])$. We want to verify the following recursion: For $t_i \neq t_j$ holds

$$[t_0,\ldots,t_n]f = \frac{\left([t_0,\ldots,\hat{t_i},\ldots,t_n]f - [t_0,\ldots,\hat{t_j},\ldots,t_n]f\right)}{t_j - t_i}, \tag{3}$$

where the "hat" means these nodes are removed.

(a) Show that

$$P(t) = \frac{(t_i - t)P(f|t_1,\ldots,\hat{t_i},\ldots,t_n) - (t_j - t)P(f|t_1,\ldots,\hat{t_j},\ldots,t_n)}{t_i - t_j}$$

is a polynomial of degree n, and that it interpolates f at $t_0,\ldots,t_n$. Hence, $P = P(f|t_0,\ldots,t_n)$.

[3] You can use MATLAB's polynomial interpolation functions `polyfit`.

[4] You can do that by choosing a very fine mesh with, say, 10,000 uniformly distributed points s_i in [-1,1], and compute the maximum of $|E_f(s_i)|$ for all i.

(b) By comparing the leading coefficients, argue that (3) holds.

5. **[Polynomial interpolation and error estimation, 1+1+1+2+2pt]** Let us interpolate the function $f : [0,1] \to \mathbb{R}$ defined by $f(t) = \exp(3t)$ using the nodes $t_i = i/2$, $i = 0, 1, 2$ by a quadratic polynomial $p_2 \in \boldsymbol{P}_2$.

   (a) Use the monomial basis $1, t, t^2$ and compute (numerically) the coefficients $c_j \in \mathbb{R}$ such that $p_2(t) = \sum_{j=0}^{2} c_j t^j$.

   (b) Give an alternative form for p_2 using Lagrange interpolation polynomials.

   (c) Give yet another alternative form of p_2 using the Newton polynomial basis $\omega_0(t) = 1$, and $\omega_j(t) = \prod_{i=0}^{j-1}(t - t_i)$ for $j = 1, 2$. Compute the coefficients of p_2 using divided differences.

   (d) Compare the exact interpolation error $E_f(t) := f(t) - p_2(t)$ at $t = 3/4$ with the estimate
$$|E_f(t)| \leq \frac{\|\omega_{n+1}\|_\infty}{(n+1)!}\|f^{(n+1)}\|_\infty,$$
   where $f^{(n+1)}$ is the $(n+1)$st derivative of f, and $\|\cdot\|$ is the supremum norm for the interval $[0, 1]$.

   (e) Find a (Hermite) polynomial $p_3 \in \boldsymbol{P}_3$ that interpolates f in t_0, t_1, t_2 and additionally satisfies $p_3'(t_3) = f'(t_3)$, where $t_2 = t_3 = 1$. Give the polynomial p_3 using the Newton basis.[5]

6. **[Property of polynomial interpolation, 2pt extra credit]** We consider Lagrange interpolation with distinct interpolation nodes $t_0, \ldots, t_n$.

   (a) Show that the nth divided difference $[t_0, \ldots, t_n]f$, i.e., the leading coefficient of the polynomial interpolant, satisfies:
$$[t_0, \ldots, t_n]f = \sum_{k=0}^{n} \frac{f(t_k)}{\prod_{i\neq k}(t_i - t_k)}. \tag{4}$$
   Hint: Differentiate the expressions of the interpolant in the monomial and the Newton basis n-times and compare.

   (b) Show that, for any continuous function $f(t)$ holds
$$(P_n(tf) - tP_n(f))(t) = (-1)^{n+1}[t_0, \ldots, t_n]f\ \omega_{n+1}(t),$$
   where $P_n(g) = P_n(g|t_0, \ldots, t_n)$ is the polynomial interpolant for a continuous function g, and $\omega_{n+1}(t) = \prod_{i=0}^{n}(t_i - t)$ is the $(n+1)$st Newton polynomial. *Hint: Multiply for $t \neq t_j$, $1 \leq j \leq n$, the numerator and the denominator in (4) by $L_k(t)(t_k - t)$, where L_i is the Lagrange polynomial for the node t_i. Argue separately for $t = t_j$.*

[5] You should only have to add a row to the derivation you did when you computed p_2 using the Newton basis. Note that since $t_2 = t_3$, you can use that the divided difference $[t_2 t_3]f = f'(t_2)$. Moreover, note that $[t_3]f = f(t_3) = f(t_2)$.