# 1. zárthelyi feladatsor

## Programcsomagok a numerikus módszerekben

*2011. március 29.*

Oldd meg az alábbi feladatokat MATLAB segítségével! Munkádhoz minden **saját** kézzel írott jegyzetet használhatsz. Netet, szomszédot nem, ennek megsértése 0 pontos eredményt von maga után. 45 perc áll a rendelkezésedre. Törekedj a szép, átlátható kódolásra.

Az elkészített fájlokat egyetlen `zip` fájlba tömörítve töltsd fel az *eLearning* rendszerbe!

**1.** Készítsd el a `szelomodszer` nevű függvényt, amely 2 adott kezdeti pontból $(x_1, x_2)$ indulva, a szelőmódszer képleteit alkalmazva végrehajt $n$ lépést.

A függvény adja vissza az $n$-dik lépés után kapott pontot, valamint az itt felvett függvényértéket, továbbá készítsen két ábrát, az egyiken a kapott pontok sorozatát kirajzolva ($x_1$ és $x_2$ pontokat is beleértve), míg a másik ábrán magát a függvényt megjelenítve $[min(x_i), max(x_i)]$ intervallumon.

A mintafüggvényt - amire végrehajtjuk a módszert - „inline" módon adjátok meg az alábbi formában: `f = @(x)(cos(x)+4*x-2);`

Beküldendő a `szelomodszer.m` fájl, valamint az `[xmo, fxmo] = szelomodszer(0,3,5);` hívás eredményeként kapott két ábra (képként).

Egy kis segítség: `function [eredm, fvertek] = szelomodszer(x1, x2, n)`; `figure` parancs; valamint mivel a kirajzoláshoz szükséges az összes pontunk, ezért érdemes egy vektorban tárolni őket.

(5 pont)

**2.** Készíts egy `hasonlit` nevű függvényt, amely adott $[-1, 1]$-en értelmezett függvényhez elkészíti az egyenletes felosztáshoz tartozó, valamint a Csebisev-alappontokhoz tartozó $n$-edfokú ($n+1$ alappont) interpolációs polinomokat, és egy ábrán megjeleníti az eredeti függvényt (piros szín), és a két interpolációs polinomot (kék és fekete szín), 0.001 finomságú felosztás mellett, az interpoláló pontokat az órán látott módon kiemelve.

A függvény megint „inline" módon adott (feltehetjük, hogy $[-1, 1]$-en értelmezett), példaként használjuk az `f = @(x)(1./(1+25*x.^2));` függvényt.

Beküldendő a `hasonlit.m` fájl, valamint egy kép, mégpedig a `hasonlit(9)` hívás eredményeként kapott ábra (a fenti mintafüggvény esetén).

Segítség: `function hasonlit(n)`; `polyfit, polyval` parancsok használata; szinek: 'r', 'b', 'k'. (esetleg még a `hold` parancs, bár nélküle is megoldható)

(4 pont)

**3.** Készítsd el a `fejerhermite` függvényt, amely elkészíti adott alappontok, függvényértékek, és függvény derivált értékekhez tartozó Fejér–Hermite-féle interpolációs polinomot, az órán szereplő képlet alapján.
Emlékeztető: $A_i(x) = [1 - 2\,(x - x_i)\,l_i'(x_i)]\,l_i^2(x), B_i(x) = (x - x_i)\,l_i^2(x)$.

A paraméterek az egyszerűség kedvéért: `function fejerhermite(x,f,fd)`, sorrendben az alappontok vektora, a hozzá tartozó függvényértékek, és végül a függvény deriváltjainak értékei az `fd` vektorban.

Például próbáld ki az $f(x) = x \sin x$ függvény interpolálására a $\{0, \frac{\pi}{4}, \frac{\pi}{2}\}$ pontokon. Ábrázold a közelítés hibáját (abszolút értékét) a $[min(x), max(x)]$ intervallumon. A visszatérési érték pedig legyen az interpoláció maximális hibája az előbb említett intervallumon. (0.001 finomságú felosztás mellett)

Beküldendő `fejerhermite.m` fájl. (képet itt nem kell)

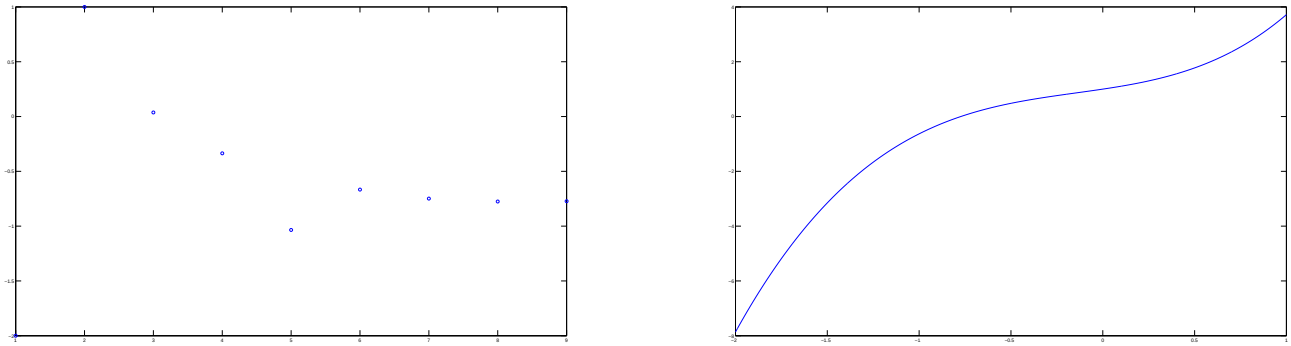
Segítség: a kérdéses hívás:
`maxhiba = fejerhermite([0,pi/4,pi/2],[0,sqrt(2)*pi/8,pi/2],[0,sqrt(2)/2+sqrt(2)*pi/8,1]);`
a hiba számítása: `hibaxx = abs(fxx-hxx)`, feltéve, hogy `fxx` változó a megfelelő felosztás melletti függvényértékek vektora (ehhez a függvényt inline módon „beégetve" használhatod), `hxx` pedig az interpolációs polinom értékeinek vektora ugyanazon felosztás mellett; ezek után `maxhiba = max(hibaxx);` utasítással megkapható a maximális hiba.

(6 pont)

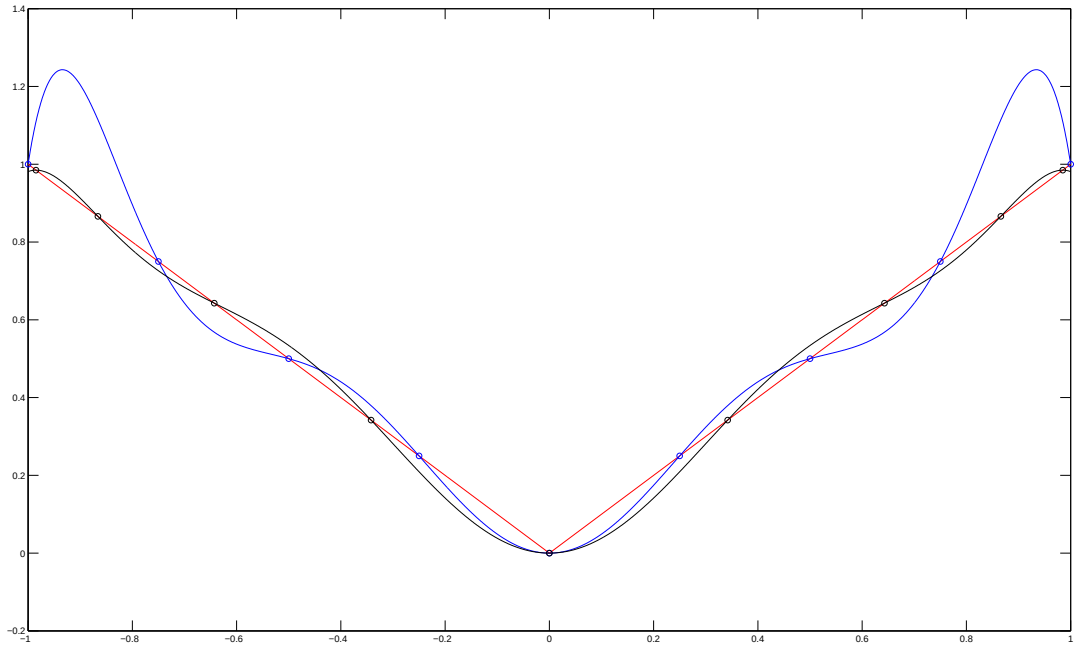
Ha így nem menne, akkor megoldható a Newton-féle táblázattal is, de ekkor a maximális pontszám:

(4 pont)

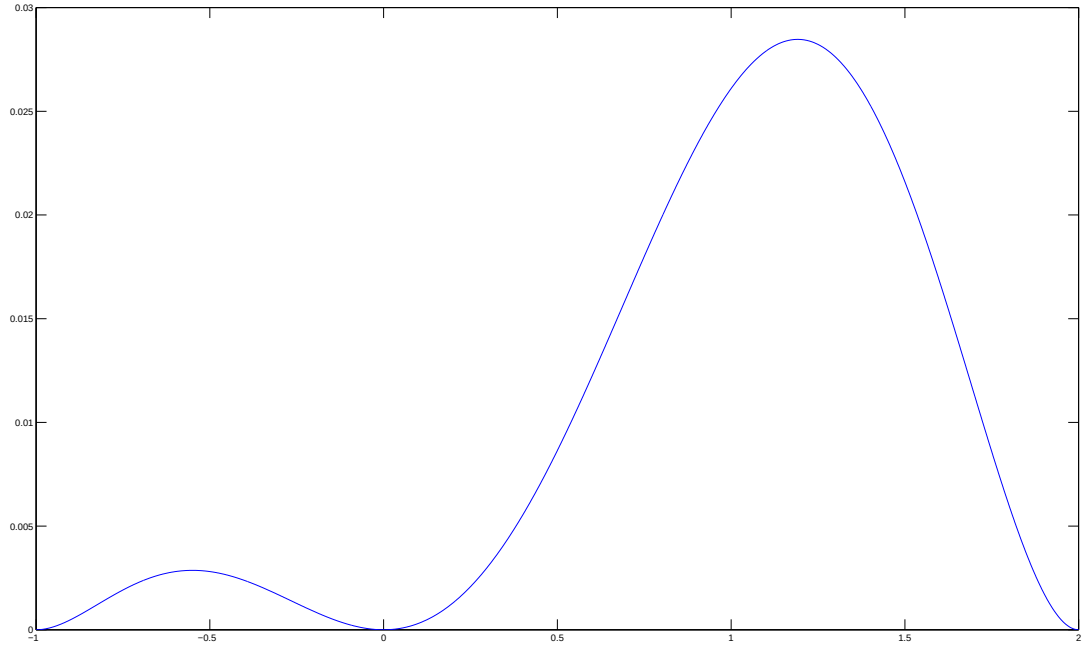
Jó munkát!



(a) 1. feladat, $e^x + x^3$ fv, (-2,1,7) paraméterekkel történő hívás esetén




(b) 2. feladat, $|x|$ függvény, $n = 8$ paraméterrel




(c) 3. feladat, példafüggvény, de $-1, 0, 2$ pontokon interpolálva