EÖTVÖS LORÁND TUDOMÁNYEGYETEM
TERMÉSZETTUDOMÁNYI KAR

TÖBBVÁLTOZÓS FÜGGVÉNYEK INTERPOLÁCIÓJA RADIÁLIS BÁZISFÜGGVÉNYEKKEL

BSc szakdolgozat

Írta: Nagy Ágnes

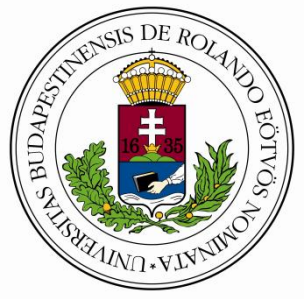
Alkalmazott matematikus szakirány

Témavezető:

Dr. Gáspár Csaba

Numerikus Analízis Tanszék

Eötvös Loránd Tudományegyetem, Informatikai Kar



Budapest, 2013

Köszönetnyilvánítás

Szeretném megköszönni konzulensemnek, Dr. Gáspár Csabának, hogy elvállalta a témavezetői teendőket, és hosszú időn keresztül mindig rendelkezésemre állt. Köszönöm, hogy iránymutatásával, szakmai tanácsaival hozzásegített a szakdolgozatom elkészítéséhez.

Köszönöm családomnak, hogy sok türelemmel és szeretettel segítettek előre tanulmányaim során.

Tartalomjegyzék

1.Fejezet

1.1 Bevezetés

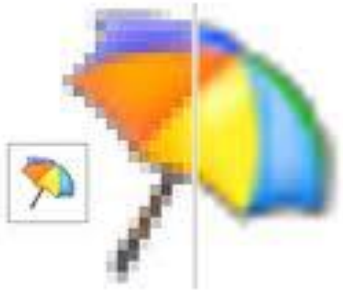
Matekos hallgatóként járva ismerőseim közt két szélsőséges hozzáállással találkozom, a többség bevallja, hogy soha nem szerette a matematikát és olyan messzire elkerüli, ahogy csak lehet. Kevesebben vannak azok, akik szerint ez a tudomány gyönyörű, és meglátja maga körül a világban annak törvényszerűségeit.

Ezért valahányszor megkérdezik, hogy miről írok a dolgozatomban, legegyszerűbb a példát a természetben keresnem.

Rengeteg térkép készül mind a szárazföld, mind a tengerfenék felszínéről. Leolvashatjuk, hogy milyen magas a Bakony egy csúcsa vagy milyen mély a Marianna-árok. Valójában azonban nem tudunk a Föld minden pontjában mérést végezni. Mégis mivel szükségünk van az információra, olyan eljárásokat alkalmazunk, amelyek a már meglévő adatokból képesek a lehető legpontosabb közelítést adni a hiányzó értékekre. Ezt nevezzük interpolációnak.

Az interpoláció a matematika rendkívül látványos része. Nem csak felület közelítésére használható; egyszerűbb eset egy egydimenziós görbe interpolálása, de a feladat általában kiterjeszthető sokváltozós függvényekre is.

Ezen kívül érdekes terület a képinterpoláció, amellyel digitális képek átméretezésekor, forgatásakor találkozhatunk [6,7]. Ha egy ilyen módon készített képet később fel akarunk nagyítani, nem a pixelek méretét növeljük, hanem a mennyiségét. Az új egységek színét és annak intenzitását pedig a környező cellák alapján választja ki az interpolációs algoritmus.



1.1. ábra: Átméretezés során bekövetkezett minőségi romlás

Fontos szerepet kap az interpoláció a meteorológiában is, légköri mozgásrendszerek és más folyamatok leírásában, kezdeti értékek közelítésében. Ezek a folyamatok parciális differenciálegyenlet-rendszer segítségével írhatók fel, melyek megoldásában segítenek az interpolációs eljárások.

Újabb kutatások témája például 3 dimenziós tárgyak felületi pontfelhőjéből való rekonstruálása interpoláció segítségével, illetve ilyen objektumok egyetlen implicit függvénnyel való megadása [9].

De megtalálhatók más földtudományokban, mérnöki tudományokban, orvostudományban, vállalati pénzügyekben is.

1.2 Interpoláció története

Az interpoláció megjelenése visszanyúlik az ókori Babilon és Görögország idejébe, ahol a Nap és más égitestek pozícióját próbálták meghatározni vele. Később, i.sz.625 körül Brahmagupta, indiai matematikus-csillagász bemutatott egy a szinusz függvény közelítésére használható másodrendű interpolációs módszert, majd egy másikat nem egyenlő intervallumokra. A legjelentősebb alkalmazása a tengeri navigáció volt. A franciák olyan táblákat próbáltak gyártani, melyek speciális függvényértékeket tartalmaztak, hogy az alapján szélességi és hosszúsági fokot tudjanak számolni, de a mérések nem bizonyultak pontosnak. (Az alakzatok pontjainak helyét mérték le.) A problémával a nagy világválság idején foglalkoztak újra az Egyesült Államokban, munkafolyamatok modellezésére [8].

Bár a méréseket ma már számítógépek végzik, mégis szükséges az interpoláció használata, hiszen a kapott eredmények egy folytonos függvény diszkrét pontjai, a kimaradó értékeket közelíteni kell. Valójában a gyakorlatban rengetegszer használjuk az interpolációt, akár tudtunkon kívül is.

2.Fejezet

2.1 Az interpoláció definiálása

Az interpoláció lényege tehát az, hogy adott pontokban adott értékekre függvényt illesztünk.

A feladat az egyváltozós függvényekhez hasonlóan megfogalmazható sík-, és térgörbékre is, a dolgozatnak azonban a többváltozós függvények interpolációjának vizsgálata a célja néhány kiemelt módszer alapján, ezért mivel minden feladat az egyváltozósra vezethető vissza, avval kezdem a leírást.

Egy dimenzióban a matematikai megfogalmazása így szól:
adottak $x_0, x_1, \dots, x_n$ értékek az $I \in \mathbb{R}$ intervallumon, nevezzük innentől interpolációs alappontoknak, és a hozzájuk tartozó $f_i = f(x_i)$ függvényértékek. Keressük azt az $F(x)$ függvényt, amely az alappontokban a kívánt értéket veszi fel, $F(x_i) = f_i \;\; i = 0,1, \dots, n.$

Léteznek globális és lokális módszerek is.
Globális módszer alatt azt értjük, hogy az interpoláns minden alapponttól függ, egy

pont elvétele vagy hozzáadása megváltoztathatja a kimenetet.
Lokális módszer esetén az interpoláns egy pontban csak annak egy bizonyos környezetében lévő alappontoktól függ, így egy adat megváltozása ezen a környezeten kívül nem okoz módosulást.

A feladat legegyszerűbb megoldása, ha az interpolánst $p(x) = x^m + a_{m-1} \cdot x^{m-1} + \ldots + a_m$ polinomként kezeljük, és minden egyes alappontot behelyettesítünk a p polinomba: $p(x_i) = {x_i}^m + \ldots + a_m = f_i \;\; i = 0,1, \ldots, n$. Az így kapott egyenletrendszer megoldásának műveletigénye $\sigma(n^3)$, emiatt nagy ponthalmazra nem alkalmazunk globális módszert, helyette lokális módszert használunk, vagy az értelmezési tartomány kisebb részeit interpoláljuk globálisan, majd összekapcsoljuk egy megoldássá.

Jelentősebb eljárások rövid leírása egyváltozós függvényekre

Lagrange-interpoláció

Tegyük fel, hogy az alappontok különbözőek. Keressük azt a legfeljebb n-edfokú $p(x) = x^n + a_1 \cdot x^{n-1} + \ldots + a_n$ polinomot, amely az alappontokban a kívánt értéket veszi fel. Ennek az interpolációs feladatnak a megoldása pedig egyértelmű lesz, $L_n(x) = \sum_{i=0}^{n} f_{x_i} \cdot l_i(x)$, ahol $l_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$, valamint $i \neq j$ esetén $l_i(x_j) = 0$ és $l_i(x_i) = 1$.
Hibabecslése: $f \in C^{n-1}([a,b])$ –re
$|f(x) - L_n(x)| \leq \frac{\|f^{(n+1)}\|_{max}}{(n+1)!} \cdot |w_n(x)|)$ ahol $w_n(x) = (x - x_0) \cdot \ldots \cdot (x - x_n)$ és korlátos $\|f^{(n)}\|_{max}$-ra $L_n(x) \longrightarrow f(x)$.

Newton-interpoláció, osztott differenciák elve

L_n interpolációs polinomot a következő formában keressük:

$$L_n(x) = c_0 + c_1 \cdot (x - x_0) + \cdots + c_n \cdot (x - x0) \cdot \ldots \cdot (x - x_{n-1})$$
$$L_n(x_0) = f_0 = c_0$$
$$L_n(x_1) = f_1 = f_0 + c_1 \cdot (x_1 - x_0),$$

innen $c_1 = \frac{f_1 - f_0}{x_1 - x_0} = f[x_0, x_1]$ elsőrendű osztott differencia,
$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}$ másodrendű differencia, $f[x_0, \ldots, x_k] = \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0}$ k-adrendű osztott differencia, innen

$$L_n(x) = L_{n-1}(x) + f[x_1, .., x_n] \cdot (x - x_0) \cdot \ldots \cdot (x - x_{n-1})$$

Hibabecslés: $\|f^{(n)}\|_{max}$-ra $L_n(x) \longrightarrow f(x)$.

Hermite-interpoláció

Lényege hogy az interpolációs alappontokban nem csak a keresett függvény értéke, hanem a függvény deriváltjainak értéke is ismert:
${f_0}^{(0)}, {f_0}^{(1)}, \ldots, {f_0}^{(m_0 - 1)}, {f_1}^{(0)}, \ldots, {f_1}^{(m_1 - 1)}, \ldots, {f_n}^{(m_n - 1)}$, ahol $m = m_0 + \cdots + m_n$.

Ekkor $P^k(x_j) = f_j^{(k)}$ $k = 0, .., m_{j-1}$ és $j = 0, \dots, n$ -re.

Hibabecslés: $|f(x) - H_{m-1}(x)| \leq \frac{\|f^{(m)}\|_{max}}{m!} \cdot (b-a)^m$.

Spline-interpoláció

Lényege, hogy az alappontokra, mint osztópontokra tekintünk az interpoláció értelmezési tartományán, és ezekben úgy választjuk meg a függvény deriváltját, hogy a szakaszonként értelmezett harmadfokú Hermite interpolánsok minél simábban kapcsolódjanak össze ezeken a helyeken.
Hibabecslés: ha S_h ekvidisztans alappontozású harmadfokú spline interpoláns és $f \in C^4$ $[a, b]$, akkor $\|f - S_h\|_{max} \leq c \cdot h^4$

Bár nem interpolációs eljárás, érdekességképpen mégis megemlíteném a Bernstein-polinomokat, melyekkel inkább egy előre adott görbét közelítünk, és úgy kapunk pontos közelítést, hogy nem követeljük meg, hogy minden alappontbeli függvényértéket felvegyen. Ekvidisztans alappontozás mellett [0,1] -en $B_n(x) = \sum_{k=0}^{n} u_k \cdot P_{n,k}(x)$), ahol $P_{n,k}(x) = \binom{n}{k} \cdot x^k \cdot (1-x)^{n-k}$ Bernstein alappolinom

2.2 Többváltozós függvények interpolációja

Az interpolációs alapfeladat megfogalmazása több változóra:
Legyen adott $P_i = x_i$ $i = 1, \dots, n$ $x_i \in \mathbb{R}^m$ különböző pontok és f_i az ezekben felvett függvényértékek. Keressük azt az F függvényt, amelyre teljesül, hogy minden i-re $F(x_i) = f_i$.

Megjegyzés: az alapfeladatot m dimenziós alappontokra definiáltuk, a későbbiekben azonban sok változó helyett csak a kétváltozós esetet fogom részletezni a könnyebb számolás és reprezentálás végett, az egydimenziós interpolációs eljárásokról előzőleg már röviden esett szó.

Ha az interpolációs alappontok értelmezési tartománya egy k dimenziós kocka és a pontok egy k dimenziós, téglalap alakú rácson fekszenek, akkor a feladat visszavezethető az egyváltozós feladatra. Például 2 dimenzióban a függvényértékeket interpoláljuk az egyik koordináta szerint és a kapott interpoláció együtthatóit a másik koordináta szerint. Ezt nevezzünk tenzorszorzat interpolációnak[5].

Ha a pontok nem ilyen szabályosan helyezkednek el, pl. nem egy téglalaprács vagy háromszögrács rácspontjai, akkor szórt alappontú interpolációról beszélünk. Ilyen módszereknél a pontoknak semmiféle struktúrájára nincs szükség.

Franke az összes szórt alappontú interpolációt átfogó munkájában 6 osztályba sorolja az eljárásokat[3,4]:

1. **Shepard-módszer**
   Az interpolációs függvény az f_i értékek súlyozott közepe:.

$F(x,y) = \frac{\sum f_k \cdot w_k(x,y)}{\sum w_k(x,y)}$, ahol $w_k(x,y) = {d_k}^{\mu}$, μ leggyakrabban 2, de lehet más érték is, és $d_k = \sqrt{(x-x_k)^2 + (y-y_k)^2}$.
Globális módszer, de több variánsa létezik, amelyben az f_i-hez tartozó súlyokat csak az alappont egy R sugarú környezetében lévő pontok alapján számolja, így lokálissá téve az eljárást[1].

2. **Téglalap alapú rácson interpolálás**

$$F(x,y) = \frac{\sum w_k(x,y) \cdot Q_k(x,y)}{\sum w_k(x,y)}$$

ahol Q_k polinom közelíti a függvényt (x_k, y_k)-ban és a hozzá legközelebb lévő 5 pontban, és $w_k = \begin{cases} 1 - \left(\frac{d_k}{R_k}\right)^2 \cdot \left(3 - 2 \cdot \frac{d_k}{R_k}\right) \text{ ha } d_k \leq R_k \\ 0 \text{ egyébként} \end{cases}$.
R_k az x_k és az 5. legközelebbi pont távolsága. Globális eljárás.

3. **Háromszöghálón értelmezett interpoláció**
Az interpolációs alapháló az alappontok konvex burka, és úgy alkotnak háromszögrácsot, hogy a csúcsokat összekötő élek nem metszik egymást. Legyen T_{ijk} az a háromszög, melynek csúcsai $(x_i, y_i), (x_j, y_j), (x_k, y_k)$. Ekkor $F(x,y) = w_i(x,y) \cdot Q_i(x,y) + w_j(x,y) \cdot Q_j(x,y) + w_k(x,y) \cdot Q_k(x,y)$, ahol $w_m(x_m, y_m) = \delta_{m,n}$ és $Q_n(x_n, y_n) = f_n \;\; m,n = i,j,k$ −ra.

4. **Véges elemű módszerek**
A módszer elképzelése, hogy az interpolációs alappontok konvex háromszöghálója fölött véges elemű C^1 függvényeket használ az approximációhoz, és az alappontokban becsüli a deriváltakat, amelyek a módszer által adott elemektől függnek.

5. **Foley módszere**
A módszer valójában több eljárás kombinációja. Először Newton típusú interpolációval készít egy alaprácsot, és utána azon egy másik approximációval, mint például a biharmonikus spline, közelíti a kérdéses függvényt.

6. **Közelítés radiális bázisfüggvényekkel**
Ennek a módszernek a lényege, hogy az alappontokra épülő radiális bázisfüggvények kombinációjaként képzeli el a függvényt. Radiálisnak nevezzük, mert $\Phi_j(x)$: $\mathbb{R}^m \to \mathbb{R}$ függvény valójában x euklideszi normájának függvénye $\Phi_j(x) = \varphi(\|x\|) \;\; \mathbb{R} \to \mathbb{R}$.
$f(x) = \sum \alpha_j \cdot \Phi_j(x - x_j)$, ahol $\Phi_j(x)$ a j. radiális bázisfüggvény.

Interpoláció radiális bázisfüggvényekkel

Láttuk, hogy az interpoláns a következő formában adható meg:
$f(x) = \sum \alpha_j \cdot \Phi_j(x - x_j)$, ahol $\Phi_j(x)$ a j.bázisfüggvény.

Ebből az interpolációs feladat így írható fel egyenletrendszerrel általánosan:

$$\alpha_1 \cdot \Phi_1(x_1 - x_1) + \ldots + \alpha_n \cdot \Phi_n(x_1 - x_n) = f(x_1)$$
$$\alpha_1 \cdot \Phi_1(x_2 - x_1) + \ldots + \alpha_n \cdot \Phi_n(x_2 - x_n) = f(x_2)$$

Írja be az egyenletet ide…

$$\alpha_1 \cdot \Phi_1(x_n - x_1) + \cdots + \alpha_n \cdot \Phi_n(x_n - x_n) = f(x_n)$$

azaz

$$\begin{pmatrix} \Phi_1(x_1 - x_1) & \cdots & \Phi_n(x_1 - x_n) \\ \vdots & \ddots & \vdots \\ \Phi_1(x_n - x_1) & \cdots & \Phi_n(x_n - x_n) \end{pmatrix} * \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}$$

ahol $A = \begin{pmatrix} \Phi_1(x_1 - x_1) & \cdots & \Phi_n(x_1 - x_n) \\ \vdots & \ddots & \vdots \\ \Phi_1(x_n - x_1) & \cdots & \Phi_n(x_n - x_n) \end{pmatrix}$ mátrixot az egyenletrendszer együttható mátrixának nevezzük.

Az ebbe az osztályba sorolható módszerek többnyire a bázisfüggvény definíciójában térnek el egymástól, a későbbiekben a következőket fogjuk vizsgálni:

- Multiquadrics (MQ): $\Phi_j(x) = \sqrt{\|x\|^2 + {r_j}^2}$
- Inverz (vagy reciprok) Multiquadrics (RMQ): $\Phi_j(x) = \frac{1}{\sqrt{\|x\|^2 + {r_j}^2}}$
- Thin Plate Spline (TPS): $\Phi_j(x) = \|x\|^2 \cdot log\|x\|$

Mindezen definíciókban r_j a felhasználó által megadott nemnegatív paraméter, a vektor normát pedig euklideszi norma szerint számoljuk.

A módszer globális jellegéből kifolyólag mindig szükséges lesz az egyenletrendszer kiszámítására, ezért az együttható mátrix tulajdonságai jelentősen befolyásolják a hatékonyságot, későbbiekben a tesztek mind ide lesznek visszavezethetőek.

Megjegyzés: léteznek kompakt tartójú radiális bázisfüggvények is, mint a Wendland-függvények: $\Psi_{l,k}(r) = I^k \Psi_{l,0}(r)$, ahol $\Psi_{l,0}(r) = (1 - r)^l{}_+$, és $If(r) = \int_r^\infty s \cdot f(s)ds$ és $r = \|x\| \in [0,1]$. Jelentőségük, hogy ekkor a felírt egyenletrendszer együttható mátrixa ritka mátrix lesz [2].

Mivel mindhárom eljárást szeretnék jellemezni és összehasonlítani, néhány alapvető szempontot szükséges meghatározni [4].

1. **Pontosság**: Mennyire jól közelít az interpoláns egy előre megadott függvényt? Mi az a legnagyobb fokú polinom, amit az interpoláns hiba nélkül közelít?

Különböző felületekhez milyen hiba társul? Milyen a megoldandó egyenletrendszer kondícionáltsága?

2. **Vizualitás**: (szubjektív) Ugyanolyan pontosság mellett létezik-e jobban közelítő függvény? Nagy pontosság mellett természetesen kicsi a valószínűsége. Hogyan viselkedik az értelmezési tartomány határain?
3. **Érzékenység a paraméterekre**: Ugyanazon ponthalmaz mellett egy paraméter jó lesz-e több típusú függvény közelítésére? Hogyan viselkedik a paraméter a felület változásakor, illetve a paraméter változása hogyan hat a felületre? Tekintsük jobbnak, ha nem érzékeny a paraméter változására és nem függ az értéke túlzottan a függvénytől.
4. **Időigény**: A ténylegesen eltelt idő mérése csak tájékoztató jellegű, hiszen befolyásolja a számítógép erőforrása is. Mekkora az interpolációs algoritmus műveletigénye? Megjegyzendő, hogy általában igaz az, hogy a hatékony módszerek valamilyen más szempont szerint gyengébbek.
5. **Végrehajtás egyszerűsége** (szubjektív): Milyen ötleteket használ, mennyire egyszerű a végrehajtása?
6. **Fejlesztés**: Létezik-e eljárás a módszer gyorsítására, pontosítására, erőforrásigényének csökkentésére?

Mindhárom módszer tesztelése hasonló alapokról indul. Kiválasztok egy origóra szimmetrikus négyzet alapú tartományt, ahol az interpolációs alappontokat kvázivéletlen generálom. Azaz úgy választom meg véletlenszámok segítségével őket, hogy egymástól vett távolságuk egy előre meghatározott epszilon mennyiségnél ne legyen kisebb.(alapgen.m) Definiálok egy adott sűrűségű alaphálót, aminek rácspontjaiban szeretnénk kiszámolni a közelíteni kívánt függvény értékeit. Ezután az RBFI.m program segítségével kirajzolom az interpolációs függvényt egy adott sűrűséggel definiált alaphálón, majd pedig az eredetitől való eltérés nagyságát. Eközben mérem az eltelt időt az interpolációs függvény kiszámítására és kirajzolásra, megnézem az együttható-mátrix kondíciószámát, illetve az eltérés nagyságát jellemzem a max, min és átlag értékkel, valamint a hiba relatív diszkrét L^2 normájával. Végül az eredményeket táblázatba foglalom.

Definíció: *Mátrix kondíciószáma* [5]:
A reguláris mátrix kondíciószáma $condA = \|A\| * \|A^{-1}\|$, értéke függ az alkalmazott vektornorma által indukált mátrixnormától.

Megjegyzés: $condA \geq 1$, mert $\|I\| = 1$ és $1 = \|I\| = \|A * A^{-1}\| \leq \|A\| * \|A^{-1}\| = condA$

Definíció: *Két függvény eltérésének relatív diszkrét L^2 normája*:
Legyen $f, g \in L^2(\mathbb{R})$, azaz $\int_{\mathbb{R}} |f(x)^2|\, dx$ létezik és véges. Valamint $(f)_{ij}$ mátrix elemei az f függvény értékei (x_i, y_j) pontokban. Hasonlóan $(g)_{ij}$ a g függvény

értékei (x_i, y_j) pontokban. Ekkor f és g eltérésének relatív diszkrét L^2 normája $\frac{\sqrt{\Sigma_{i,j}|f_{ij}-g_{ij}|^2}}{\sqrt{\Sigma_{i,j}|f_{ij}|^2}}$.

3.Fejezet

3.1 Multiquadrics

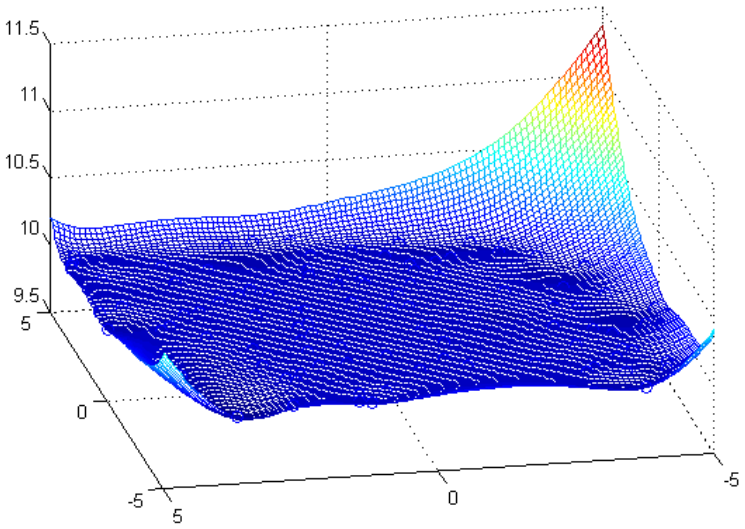
$$\begin{pmatrix} \Phi_1(x_1 - x_1) & \cdots & \Phi_n(x_1 - x_n) \\ \vdots & \ddots & \vdots \\ \Phi_1(x_n - x_1) & \cdots & \Phi_n(x_n - x_n) \end{pmatrix} * \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}$$

ahol $\Phi_j(x) = \sqrt{\|x\|^2 + r_j{}^2}$.

Tehát a feladat, hogy kiszámoljuk α_j -ket, majd ennek segítségével az alaphálóbeli függvényértékeket.

A definícióban r_j-vel jelölt változót skálázó paraméternek nevezzük, amit a felhasználónak magának kell megadni. Később látni fogjuk, hogy egy rosszul beállított paraméterrel milyen silány eredményt kapunk. Sok tanulmány keresett olyan algoritmust, amivel egységesen jó skálázó paramétereket lehet számítani, de a probléma mind a mai napig fennáll. Csak irányelvek léteznek, ezért az első tesztekre konstans $r_j = 1$ értéket állítok be.

Tekintsük most az $f(x) = 10$ konstans polinomot. Legyen az értelmezési tartomány a [-5,5,-5,5] négyzet, $epsz = 0.01$, az interpolációs háló sűrűsége 0.1(sur), a skálázó paraméterek egységesen 1, valamint az alappontok száma 100.



3.1.1. ábra: f(x,y)=10 konstans függvény interpolánsa 100 pontos MQ esetén

Ahogy az az ábrán is látszik, már a konstans 10 függvényt sem közelíti pontosan, ezért nem beszélhetünk polinomiális pontosságról, bár a hiba a tartományhatáron jelentkezik leginkább. Az A együttható mátrix kondíciószáma 2,34110E+07 lesz, és a hiba L^2 normája 1.3309. Valójában ennek a hiánya még nem jelenti, hogy a módszer nem hatékony, azonban létezik olyan eljárás, amivel a MQ polinomiálisan pontossá tehető.

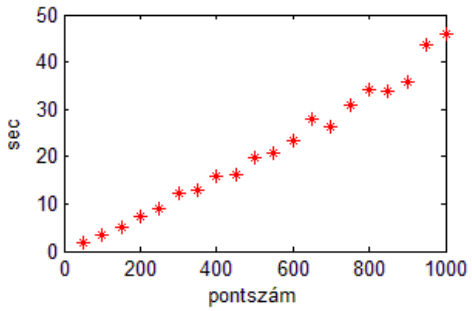
Annak a szemléltetésére, hogy egy sima függvény esetén milyen gyorsan javul az approximáció az interpolációs pontok számának növelésére, tekintsük $f(x,y) = sin(x) \cdot sin(y)$ függvényt. Az alappontokat kétféleképpen generálom: a táblázat baloldalán lévő adatok olyan ponthalmazokból származnak, ahol a meglévő pontokhoz újakat generáltam, a táblázat jobb oldalán lévő adatokhoz pedig minden halmazt újraválasztottam.

condA	Hibaátlag	L2 norma	Felhasznált idő	p	condA	Hibaátlag	L2 norma	Felhasznált idő
453.1801	9.4752	1.1323	0.40144	10	453.1801	9.4752	1.1323	0.40144
185224.8375	0.29989	0.48625	1.8164	50	284300.817	-0.17262	0.51964	1.6878
6,02354E+18	0.012117	0.24878	3.8397	100	2,34110E+07	0.056652	0.44966	3.3931
2,7941E+20	-0.0022043	0.081679	5.891	150	1,11472E+07	0.00078029	0.038998	5.1195
2,51615E+21	0.00022557	0.020283	7.9045	200	7,32035E+07	-0.00038789	0.026781	7.2813
5,41181E+21	-5.2577e-005	0.015011	9.5487	250	2,64171E+08	-0.067038	0.019717	9.1006
2,46463E+22	0.00016823	0.013414	11.3486	300	3,56640E+09	-0.00010474	0.022775	12.3724
8,52915E+22	-1.5317e-006	0.0051464	14.0368	350	5,97167E+09	1.1081e-005	0.0077647	12.8435
2,42105E+23	1.2799e-005	0.004573	15.5101	400	8,61961E+09	8.3072e-006	0.0075398	15.7827
1,0818E+23	-2.1739e-006	0.0043452	17.764	450	7,26252E+09	1.9345e-005	0.008624	16.2284
4,16314E+24	-9.98e-007	0.0041956	19.5164	500	1,68397E+11	-7.086e-006	0.0058678	19.6424
1,08051E+24	-1.1871e-006	0.0050429	22.9004	550	6,62463E+10	-1.3854e-005	0.0088473	20.6034
2,77548E+24	-2.7305e-008	0.0048523	25.1039	600	8,97885E+10	-1.684e-006	0.0065191	23.4424
1,92003E+25	1.5909e-006	0.0030727	25.3087	650	3,63705E+11	-4.6289e-007	0.0015873	27.8324
3,78816E+25	3.7745e-007	0.0029647	27.9483	700	3,64063E+12	1.9716e-007	0.0027082	26.2343
1,78396E+25	-5.963e-007	0.0026191	32.1069	750	2,44293E+12	-3.2803e-007	0.0027154	30.7617
4,18875E+25	1.0854e-007	0.0024322	33.3447	800	2,58921E+12	-3.1879e-006	0.0064274	34.2421
1,09664E+27	-2.6254e-007	0.0023479	35.2972	850	6,17316E+11	1.876e-008	0.0015078	33.8184
3,58466E+26	2.9211e-007	0.0021131	38.671	900	2,57969E+12	8.485e-007	0.004788	35.8834
6,1701E+26	2.2744e-007	0.0020756	41.2252	950	7,40429E+12	-3.7969e-007	0.0021149	43.5557
1,07079E+27	6.0845e-008	0.0019558	43.3403	1000	6,67624E+12	-3.7645e-007	0.0027818	45.9393

3.1.2. ábra: Pontszám növelés hatása bővített(balra) és újragenerált(jobbra) ponthalmazokra

Jól látható, hogy a végrehajtási idő és a hiba normája nem tér el nagyon, de az átlagos hiba nagyság és az együtthatómátrix kondíciója a felére csökkent, amikor a ponthalmazokat újra meghatároztam. Ez nyilván következik abból, hogy ha egy ponthalmazra rossz kondíciószámot kapok, akkor további pontok hozzá vételével csak rontok az értékén, míg egy új halmazban esetleg távolabb kerülnek a pontok egymástól.

Az eredeti függvénytől eltérés kisebb ponthalmazon az értelmezési tartomány belső részén is megfigyelhető, míg később csak a tartomány határánál jelentkezik. A hiba nagysága diszkrét L^2 normában nézve monotonon csökken, 350 pont környékén éri el a néhány ezredes értéket, ahonnan 1000 pontig nem is mozdul el. Eközben a felhasznált idő 50 pontonként legalább 2 másodperccel nő.



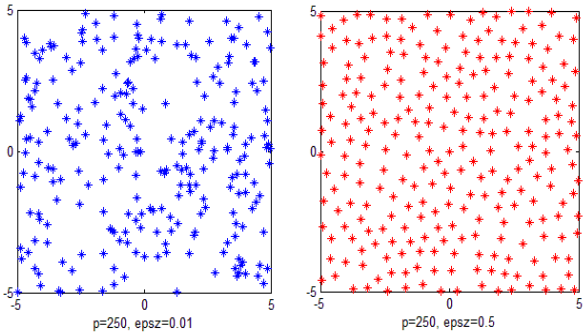
3.1.3. ábra: Felhasznált idő a pontszám függvényében f(x,y)=sin(x)*sin(y) közelítése során MQ-szal

Ha két alappont közel van egymáshoz, akkor a rájuk felírt egyenletek, és így az A együttható mátrixban nekik megfelelő sorok is majdnem azonosak lesznek. Ez elrontja az A mátrix kondícionáltságát, közel szinguláris lesz. Szükséges, hogy úgy válasszuk meg az interpolációs pontokat, hogy azok ne legyenek kollineárisak és egymástól való távolságuk minél nagyobb legyen. Ez utóbbit $epsz = 0.01$-nek rögzítettük. Mi történik, ha változtatjuk az értékét? Figyeljük meg 250 és 500 pont esetén $f(x,y) = sin(x) \cdot sin(y)$-ra:

p=250:

epsz	condA	Hibaátlag	L2 norma
0.01	2,64171E+08	-2.6887e-005	0.019717
0.1	6,16715E+07	-0.00038207	0.029469
0.2	1,18636E+07	-0.00014968	0.01514
0.3	4,42926E+06	-0.00011643	0.014976
0.4	1,81336E+06	7.4612e-006	0.0067889
0.5	7,57818E+05	-1.274e-005	0.0051803

3.1.4. ábra: 250 pontra epsz növelésének hatása f(x,y)=sin(x)*sin(y) interpolánsra

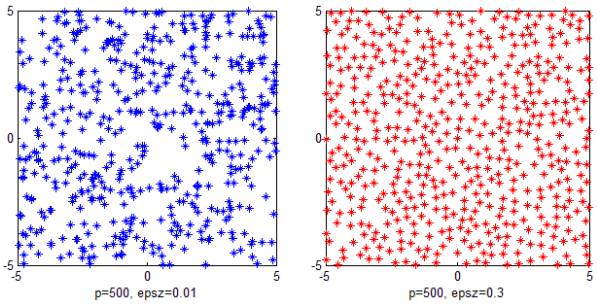


3.1.5. ábra: epsz növelésével az alappontok rendeződnek az É.T.-on

p=500:

epsz	condA	Hibaátlag	L2 norma
0.01	1,68397E+11	-7.086e-006	0.0058678
0.1	1,78790E+09	1.869e-007	0.0060667
0.2	2,73905E+08	1.0205e-006	0.0035893
0.3	4,55681E+07	5.3459e-007	0.0019246

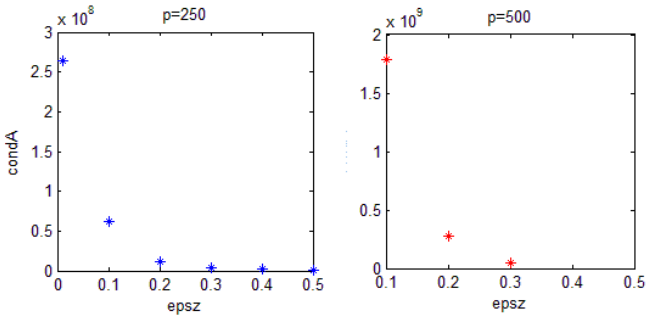
3.1.6. ábra: 500 pontra epsz növelésének hatása f(x,y)=sin(x)*sin(y) interpolánsra



3.1.7. ábra: 0.4-es minimális távolságra nem lehet 500 pontot elhelyezni az É.T.-on

A jobb oldali ábrán látszik, hogy $epsz$ növelésével a pontok egyre szabályosabban helyezkednek el, az intervallum közelít a teljes kihasználtsághoz, 0.4-es ponttávolságra már nem tudunk 500 pontot elhelyezni. A kondíciószám mindkét esetben közel exponenciálisan csökken. A pontosság javul $epsz$ növelésével, és az is

leolvasható, hogy magasabb pontszámra kisebb *epsz* érték is elég ugyanahhoz a hibanagysághoz.

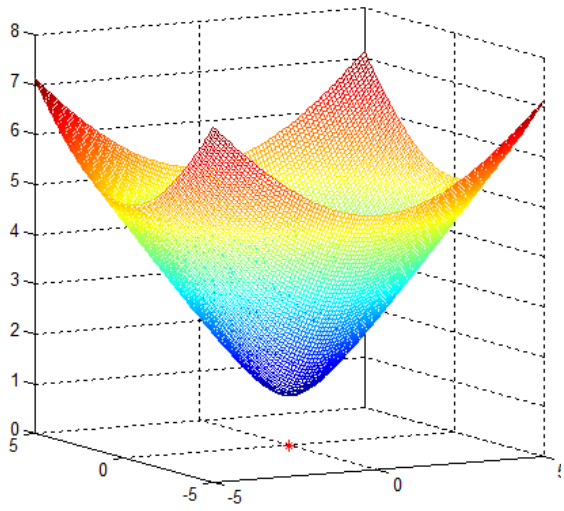


3.1.8. ábra: epsz növelésével A kondíciószáma közel exponenciálisan csökken

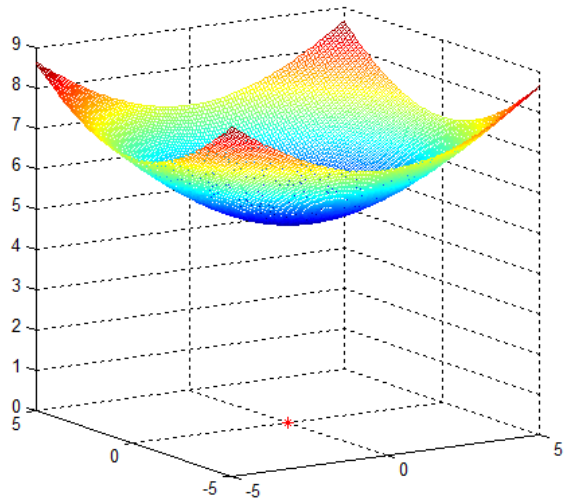
Az interpoláns építőkövei a bázisfüggvények, melyek MQ estén önmagukban paraboloidhoz hasonló felületet alkotnak. $\Phi_j(x - x_j) = \sqrt{\|x - x_j\|^2 + r_j{}^2}$

Látható, hogy a függvény formáját a skálázó paraméter határozza meg. Hogy hogyan befolyásolja $r_j{}^2$ az interpolánst Tarwater, Lancaster és Stalkauss tanulmányaiban olvashatunk.

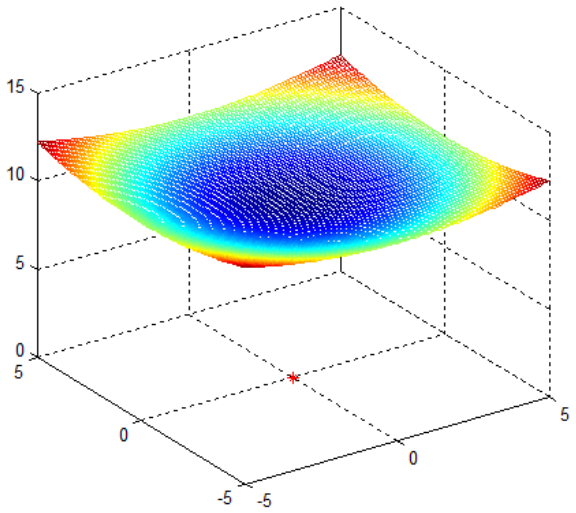
Nyilvánvaló, hogy a $\Phi_j(x) = \sqrt{\|x\|^2 + r_j{}^2}$ j. bázisfüggvény szélsőérték helye az origo, ahol az r_j éréket veszi fel. Tehát r_j szabályozza a bázisfüggvény formáját [3]. Kis r_j értékre szűk, tölcsérszerű alakzatot kapunk, közepes r_j-re „tálszerű" , míg nagy r_j-re sekély „lapszerű" függvényt kapunk. Így a skálázó paraméterek variálásával nagyon pontosan lehet közelíteni vegyes felületeket is. Sőt mondhatjuk, hogy egyenes felületet nehezebb MQ-szal interpolálni, Tarwater a legnehezebben approximálhatónak a „meredek hegy" típusú függvényeket találta.



3.1.9. ábra: MQ bázisfüggvény tölcsérszerű lesz kis r_j -re



3.1.10. ábra: MQ bázisfüggvény tálszerű közepes r_j mellett



3.1.11. ábra: MQ bázisfüggvény lapszerű nagy r_j -re

Azonban a skálázó paraméter befolyásolja az A együttható mátrix kondíciószámát is, hiszen a főátlóban és minden egyes mátrixelemben is megjelennek. Így körültekintően kell eljárni használatukkal, hiszen a rosszul kondícionáltság kihat az eredményre is.

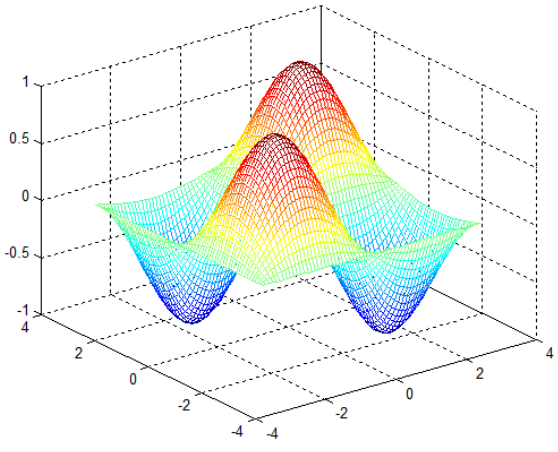
Javítja a kondíciószámot, ha a mátrix szimmetrikus, azaz a mi estünkben minden r_j $j = 1, \dots, n$ egyenlő. Nézzük meg, hogy $f(x, y) = sin(x) \cdot sin(y)$ próbafüggvénynél maradva a skálázó paraméterek változtatása hogyan hat az interpoláció pontosságára, ha az A mátrix szimmetrikusságát meghagyjuk. A többi paramétert a következőképp rögzítettem: É.T.=[-pi,pi,-pi,pi] , $epsz$ =0.2 és a pontok száma 300.

A mért adatokból láthatjuk, hogy $r_j \leq 0.1$-re mind a kondíciószám, mind a hiba normája viszonylag állandó, e fölött viszont mindkét érték gyorsan változik. Nagy r_j értékekre az együtthatók széles skálán mozognak, és előjelük változó, ezért megnöveli a kondíciószámot, A közel szinguláris sá válik [4].

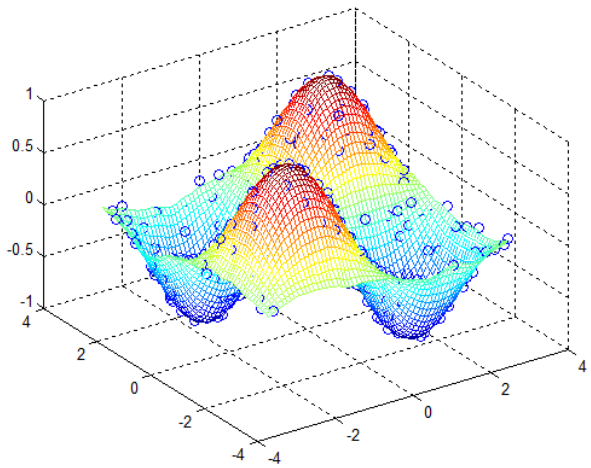
r_j=3-ra elér egy optimális pontosságot az interpoláns, később a hiba nagysága ismét nőni kezd. Némi eltérést tapasztaltam Foley forrás..-hoz képest miszerint ${r_j}^2 > 10$ -re jellemző a hiba növekedése, míg a saját tesztelésemkor csak $r_j = 4$ után következett ez be, de rögtön megjegyzem, hogy ekkor a kondíciószám már olyan nagy (10^{16} nagyságrendű), hogy a számítógép korlátai miatt az értékek hibásak lehetnek.

r	condA	L2 norma
0*ones(p,1)	8,75849E+03	0.039681
0.001*ones(p,1)	8,83402E+03	0.039594
0.01*ones(p,1)	9,55265E+03	0.038828
0.1*ones(p,1)	2,24957E+04	0.032383
0.2*ones(p,1)	6,35283E+04	0.026715
0.3*ones(p,1)	1,83131E+05	0.021984
0.4*ones(p,1)	5,23949E+05	0.017989
0.5*ones(p,1)	1,47444E+06	0.014625
0.6*ones(p,1)	4,07186E+06	0.011822
0.7*ones(p,1)	1,10405E+07	0.0095168
0.8*ones(p,1)	2,94397E+07	0.0076456
0.9*ones(p,1)	7,73620E+07	0.0061442
1*ones(p,1)	2,00745E+08	0.0049491
1.2*ones(p,1)	1,31000E+09	0.0032504
1.4*ones(p,1)	8,26194E+09	0.0021742
1.6*ones(p,1)	5,08694E+10	0.0014733
1.8*ones(p,1)	3,12455E+11	0.0010009
1.9*ones(p,1)	7,72042E+11	0.00082285
2*ones(p,1)	1,89595E+12	0.00067414
2.2*ones(p,1)	1,11941E+13	0.00044571
2.4*ones(p,1)	6,42652E+13	0.00028667
2.6*ones(p,1)	3,59408E+14	0.00017768
2.8*ones(p,1)	1,95833E+15	0.00010477
3*ones(p,1)	1,05495E+16	5.7596e-005
3.2*ones(p,1)	5,27455E+16	2.8341e-005
3.4*ones(p,1)	5,56502E+17	1.1485e-005
3.6*ones(p,1)	2,31363E+18	7.6741e-006
3.8*ones(p,1)	5,36031E+18	5.129e-006
4*ones(p,1)	7,34234E+19	9.7417e-006
4.6*ones(p,1)	1,16238E+19	1.3764e-005
5*ones(p,1)	3,35613E+19	0.0002076

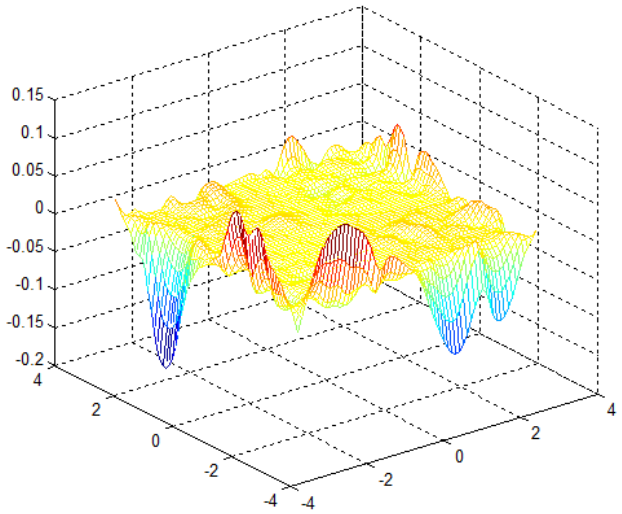
3.1.12. ábra: Skálázó paraméter módosításával mért adatok sin(x)*sin(y) függvényre



3.1.13. ábra: f(x,y)=sin(x)*sin(y)



3.1.14. ábra: sin(x)*sin(y) MQ interpoláltja 300 alappontra



3.1.15. ábra: sin(x)*sin(y) MQ interpolánsának hibafüggvénye

Elképzelhető azonban, hogy ha az r_j értéket egyesével változtatom, bár nem lesz szimmetrikus az együttható mátrix, mégis pontosabb eredményt kapunk. Kansa írása alapján a kondíciószám szinten tartása érdekében csak monoton variációját engedjük meg r_j-nek [3]. Próbaképpen $r = (r_j)$ mátrix elemeit határozzuk meg úgy, hogy egy intervallumot n egyenlő részre osszuk fel. Előző tesztünkből azt kaptuk, hogy a $sin(x) \cdot sin(y)$ függvényre $r_j = 3$ optimális. A következő táblázatban látható, hogy nincs nagy különbség, hogy monoton növekvő vagy csökkenő módon választottuk az $r_j$értékeket.

Kicsi értékeket választva, $r_j \in [0,1]$, illetve $r_j \in [0,3]$ esetén a kondíciószám egész alacsony, de nem túl pontos a közelítés, hasonló eredményt kapunk, ha r_j-t konstans egynek választjuk. Arra gondolva, hogy az interpolálandó függvényünk változatos felületű, széles skálán megválasztva r_j értékeket, $r_j \in [0,5]$ valóban pontos becslést kapunk, de a kondíciószám hatványozódik. Ha az intervallumot leszűkítem a korábbi r_j=3.4-es optimális érték körül, a kondícionáltság tovább romlik, de a pontosság javul.

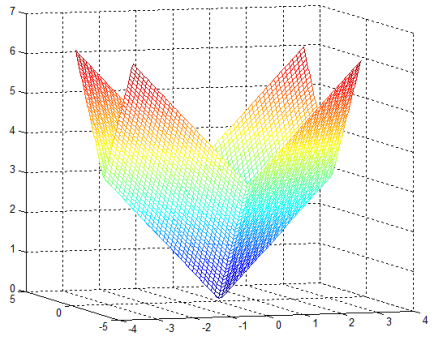
r	condA	L2 norma
linspace(1,0,300)	1,38072E+07	0.018249
linspace(0,1,300)	3,63832E+07	0.025235
linspace(3,0,300)	9,01636E+12	0.0031632
linspace(0,3,300)	2,86663E+12	0.0015105
linspace(5,0,300)	9,64328E+16	6.8628e-005
linspace(0,5,300)	1,73263E+16	0.00050511
linspace(3,1,300)	1,83396E+14	0.00044274
linspace(1,3,300)	5,89608E+13	0.00079959
linspace(4,1,300)	2,43221E+16	0.00023055
linspace(1,4,300)	1,21847E+16	0.0001507
linspace(6,0,300)	5,34763E+17	9.974e-005
linspace(2,4,300)	2,71783E+17	5.7954e-005
linspace(2.5,3.5,300)	1,47402E+17	0.00013965
linspace(3,4,300)	1,55426E+18	2.787e-005
linspace(3.2,3.8,300)	2,87250E+18	1.6014e-005

r	condA	L2 norma
0*ones(p,1)	8,75849E+03	0.039681
1*ones(p,1)	2,00745E+08	0.0049491
2*ones(p,1)	1,89595E+12	0.00067414
3*ones(p,1)	1,05495E+16	5.7596e-005
3.4*ones(p,1)	5,56502E+17	1.1485e-005
4*ones(p,1)	7,34234E+19	9.7417e-006
5*ones(p,1)	3,35613E+19	0.0002076

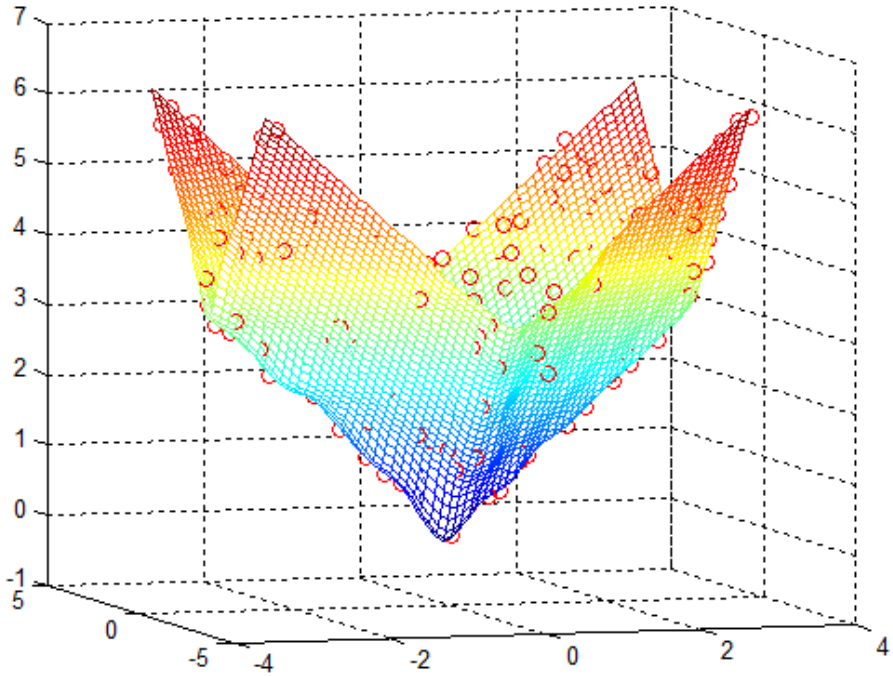
3.1.16. ábra: Skálázó paraméter monoton variációjával mért adatok, valamint a korábban konstans érték mellett mért adatok

A paraméter lineáris és exponenciális variációja is használható a mátrix kondíciószámának csökkentésére [3]. Hardy a következőképpen határozta meg ${r_j}^2$ paramétert: $r^2 = (0.815 \cdot d)^2$, ahol d az interpolációs pontok és azok legközelebbi szomszédjuktól vett távolságok átlaga. A forrásban azonban a legjobb módszert r_j kiválasztására $r^2(j) = \frac{{r_{min}}^2}{n-1} \cdot \left(\frac{{r_{max}}^2}{{r_{min}}^2}\right)^{j-1}$ $j = 1, \dots, n$ definícióval kapták, ahol ${r_{min}}^2$ és ${r_{max}}^2$ adott számok voltak. Kansa

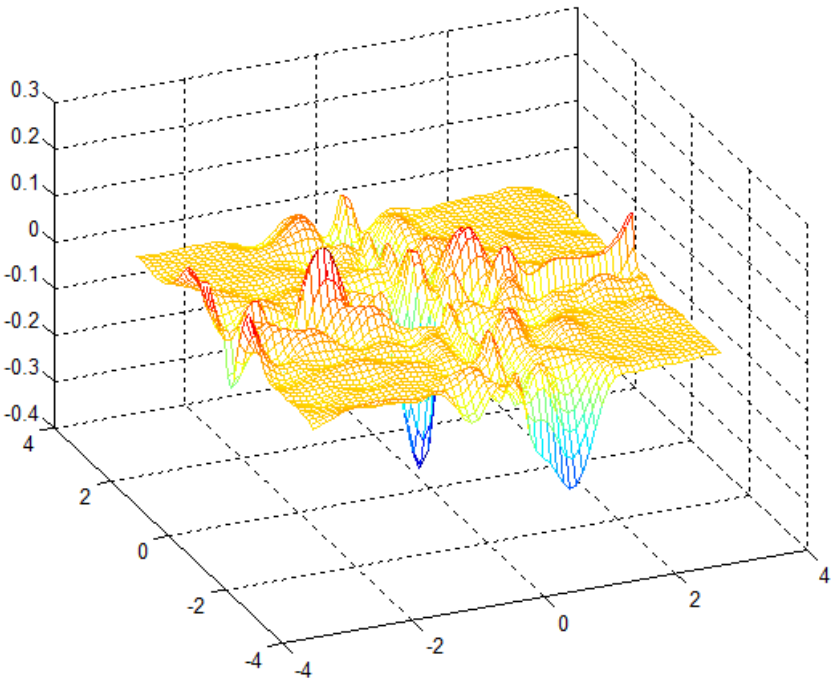
Eddig láttuk hogyan viselkedett a MQ sima függvényre, most nézzük meg hogyan kezeli, ha a függvénynek van olyan pontja, ahol nem differenciálható. Legyen $f(x,y) = abs(x) + abs(y)$. Maradjon az értelmezési tartomány [-pi,pi,-pi,pi], p=300, $epsz = 0.2$, és r_j=1.



3.1.17. ábra: f(x,y)=abs(x)+abs(y) függvény



3.1.18. ábra: abs(x)+abs(y) függvény MQ interpoláltja



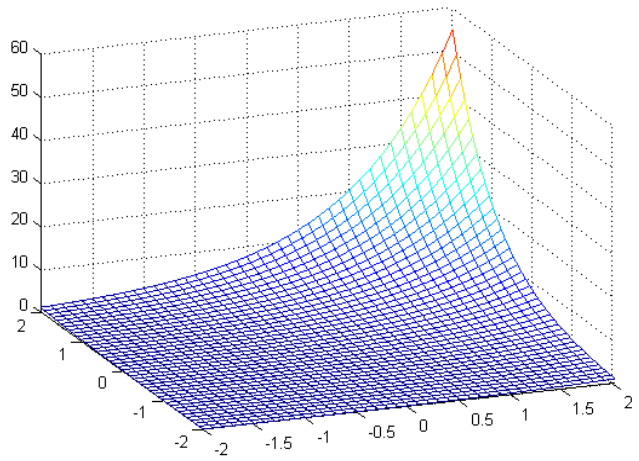
3.1.19. ábra: abs(x)+abs(y) függvény MQ interpoláltjának hibafüggvénye

Jól látszik az ábrákon, hogy azokra a paraméterekre, amelyekre sima függvény esetén egy igen pontos közelítést kaptunk az interpolációs módszerrel, most lekerekíti az éleket és a csúcsot. Elvégezve az interpolálást 500 és 600 pontra látvány szempontjából közelebb kerülünk az eredeti függvényhez, azonban a normában tekintett hiba nagysága nem csökken jelentősen. Skálázó paraméter variálása sem egységes konstansként nem javított a pontosságon, sem egy intervallumból véve az értékeket. A megoldás az lehet, hogy a kritikus helyeken megnöveljük az alappontok számát. Például, ha a [-1,1,-1,1] tartományon, azaz az origonál lévő csúcs környezetében felveszek 10 új pontot, akkor az approximáció során a kondíciószám duplájára ugrik, de a hiba normája csökkeni fog.

A MQ rosszul közelíti az enyhe gradiensű felületeket. Következő tesztfüggvényünk legyen $f(x,y) = exp(x+y)$. Kiindulásképpen legyen az É.T.=[-2,2,-2,2], $epsz$ =0.1, r_j=1 $j = 1, \dots, n$. A következő táblázat balra az $exp(x+y)$ és jobbra a $sin(x) \cdot sin(y)$ függvény teszteredményeit tartalmazza ugyanazon feltételek mellett. A kondíciószámok nyilvánvalóan megegyeznek, hiszen csak az alappontok és skálázó paraméterek határozzák meg az együttható mátrixot, a függvényértékektől nem függ. A módszer pontossága $exp(x+y)$ esetében viszont csak tizede a másiknak.

condA	L2 norma	p	condA	L2 norma
2,96321E+12	0.0042179	400	2,96321E+12	0.00020961
1,93441E+13	0.001309	500	1,93441E+13	0.00029466
7,80903E+13	0.00040081	600	7,80903E+13	8.1287e-005
4,41847E+14	0.00010177	700	4,41847E+14	2.7438e-005
1,68486E+15	0.00043769	800	1,68486E+15	2.6669e-005
4,40757E+15	0.00046643	900	4,40757E+15	1.853e-005
4,64763E+16	4.4854e-005	1000	4,64763E+16	3.1824e-006

3.1.20. ábra: MQ pontosabban közelíti sin(x)*sin(y) függvényt, mint exp(x+y) függvényt, balra az exponenciális interpolánsának adatait, jobbra a sinusos függvényét



3.1.21. ábra: f(x,y)=exp(x+y)

Nézzük meg lehet-e javítani a pontosságon vagy a kondícionáltságon a skálázó paraméter módosításával. 700 pontra vizsgálva a hibát tizedére tudtam csökkenteni a paraméterek növelésével illetve monoton variációjával, de a kondíciószám mindenképpen megnövekedett.

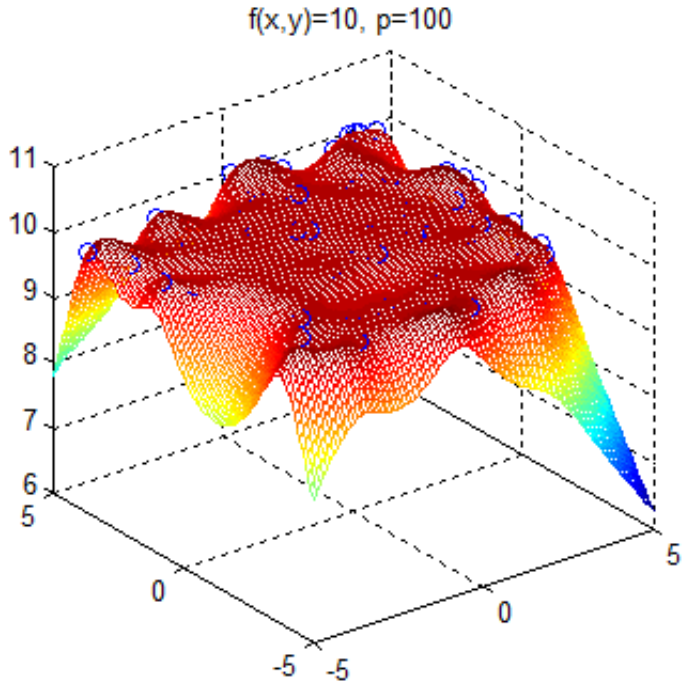
3.2 Inverz (vagy Reciprok) Multiquadrics

$$\begin{pmatrix} \Phi_1(x_1 - x_1) & \cdots & \Phi_n(x_1 - x_n) \\ \vdots & \ddots & \vdots \\ \Phi_1(x_n - x_1) & \cdots & \Phi_n(x_n - x_n) \end{pmatrix} * \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}$$

ahol $\Phi_j(x) = \frac{1}{\sqrt{\|x\|^2 + r_j^2}}$, ahol r_j legyen pozitív.

r_j skálázó paramétereket a felhasználó választja meg, ezért a későbbiekben fontos lesz annak követése, hogy hogyan befolyásolja a RMQ teljesítményét.

Hasonlóan a MQ-hoz, a RMQ-ról is elmondható, hogy nincs polinomiális pontossága, az $f(x, y) = 10$ konstans függvényt nem tudja hiba nélkül közelíteni.



3.2.1 ábra: RMQ nem közelíti hiba nélkül a konstans 10 függvényt

Tudjuk, hogy az interpolációs pontok számának növelése javít a módszer pontosságán, de rontja a kondícionáltságot. MQ-nál alátámasztottuk azt is, hogy érdemes a ponthalmazokat ennek során újragenerálni, mert a kondíciószám így lassabban emelkedik, mintha meglévő pontokhoz újabbakat veszünk hozzá. Igaz lesz ez minden módszerre, ezért itt külön nem vizsgálom a ponthalmazok generálásának hatékonyságát.

Legyen $f(x,y) = sin(x) \cdot sin(y)$, É.T.=[-5,5,-5,5], $epsz$ =0.01, r_j=1 $j = 1, \dots, n$. A pontok számának gyarapodásával a kondíciószám 50 pontonként 10-100-szorosára emelkedik, a hiba relatív diszkrét L^2 normája lassan csökken, 850 pontnál éri el a 0,004 ezredes értéket és a felhasznált idő rohamosan nő.

p	condA	Hibaátlag	L2 norma	Felhasznált idő
10	2,26518E+01	9.7295	1.0073	0.59254
50	5,33429E+03	-0.11263	0.54021	2.917
100	3,68882E+05	0.035915	0.47573	5.7627
150	1,17444E+05	-0.00097909	0.1131	8.0849
200	7,99874E+05	-0.00052291	0.088325	10.6341
250	1,76317E+06	0.0001262	0.064882	14.1378
300	2,12337E+07	-0.00031353	0.067765	16.6908
350	3,76180E+07	0.00012511	0.026516	20.8178
400	5,46856E+07	6.5174e-005	0.032341	21.9492
450	3,96724E+07	0.00010348	0.026975	26.181
500	8,74847E+08	1.6011e-005	0.028224	29.918
550	5,47209E+08	-7.0273e-005	0.048024	32.0835
600	3,86659E+08	8.8402e-006	0.016982	37.2715
650	1,70715E+09	-2.0167e-006	0.002566	39.5203
700	1,73222E+10	-3.8822e-006	0.0044441	46.7885
750	1,07386E+10	-2.5487e-006	0.0048225	44.5864
800	9,50237E+09	1.0083e-005	0.014586	51.4533
850	2,56067E+09	1.7129e-006	0.0040791	52.6613
900	1,05987E+10	-3.9247e-006	0.009787	58.3659
950	2,61957E+10	-2.4585e-007	0.0018533	62.9054
1000	2,53324E+10	5.7636e-007	0.0020089	63.8702
1100	2,99638E+10	-1.1201e-007	0.0012567	71.1484
1200	1,30413E+11	2.7275e-007	0.0015333	79.4245
1300	2,29155E+11	2.5788e-007	0.0018809	86.754
1400	3,72694E+11	5.2025e-008	0.00040499	91.4809
1500	3,70258E+11	-1.8241e-007	0.0011136	106.4078
1600	1,29007E+12	-2.9681e-009	0.00017931	116.9518

3.2.2 ábra: RMQ interpoláció tesztjének adatai növekvő ponthalmazra sin(x)*sin(y) függvény esetén

A kondíciószám csökkentésére először az alappontok egymástól való távolságát növeljük meg. 250, 500 és 700 pontra tekintve a mért adatok a következők:

p=250

epsz	condA	Hibaátlag	L2 norma
0.01	1,76317E+06	0.0001262	0.064882
0.1	4,44504E+05	-0.00084228	0.10385
0.2	8,55607E+04	-0.00029552	0.043798
0.3	3,62949E+04	-0.00027067	0.05292
0.4	1,37452E+04	3.8696e-006	0.018681
0.5	6,23819E+03	-7.7183e-006	0.0077464

p=500

epsz	condA	Hibaátlag	L2 norma
0.01	8,74847E+08	1.6011e-005	0.028224
0.1	9,18920E+06	-5.6059e-006	0.013856
0.2	1,51111E+06	4.2929e-006	0.014707
0.3	2,66420E+05	-1.3863e-006	0.0015973

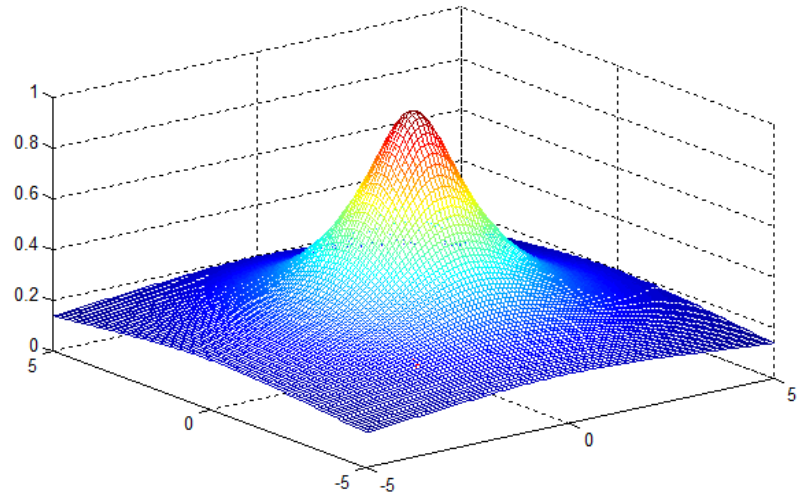
p=700

epsz	condA	Hibaátlag	L2 norma
0.01	1,73222E+10	-3.8822e-006	0.0044441
0.1	9,84083E+07	-1.0821e-005	0.012937
0.2	5,93664E+06	-7.7481e-006	0.0072539
0.3	7,27808E+05	1.2813e-007	0.00060068

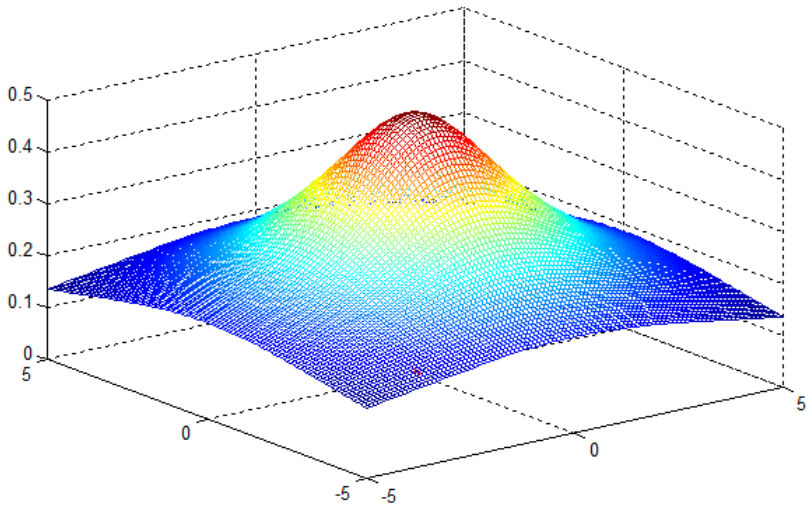
3.2.3 ábra: epsz növelésének hatása RMQ interpolánsra 300, 500 és 700 alappontra

Mindhárom esetben drasztikus változás történt, a kondíciószám 3-5 tizedes jeggyel kisebb lett, a pontosság pedig 700 pontra már tízezredes nagyságig csökkent. Ugyanez az érték $epsz$ =0.01 mellett csak 1400 pont segítségével volt elérhető. Ennek a módszernek az értelmezési tartomány és a ponthalmaz nagysága szab határt, 0.3-nál nagyobb távolsággal nem adható meg 500 pont.

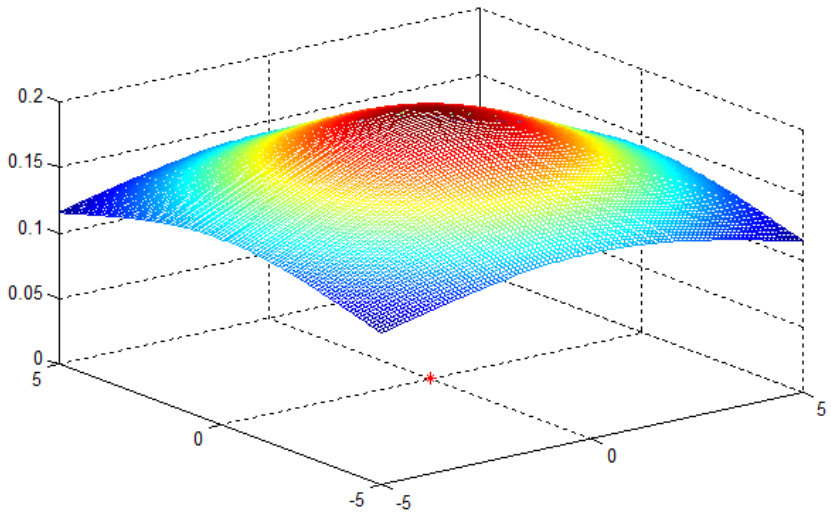
Az interpoláns ezen kívül csak a skálázó paraméterektől függ. Igaz lesz, hogy $\Phi_j(x) = \frac{1}{\sqrt{\|x\|^2 + r_j{}^2}}$ j. bázisfüggvény formáját r_j határozza meg. Az alábbi képek azt mutatják meg, hogy $r_j = 1,2,5$, azaz kicsi, közepes és nagy értékekre folymatosan kilapul a függvény felülete. Ennek oka, hogy a szélsőérték helyén, az origoban $\frac{1}{r_j}$ értéket vesz fel, ami r_j növelésével egyre kisebb lesz.



3.2.4. ábra: RMQ bázisfüggvény r_j=1-re



3.2.5. ábra: RMQ bázisfüggvény r_j=2 -re



3.2.6. ábra: RMQ bázisfüggvény r_j=5 -re

Eddig azonosan 1-nek választottuk r_j -t, ami azért jó, mert $r_1 = r_2 = \ldots = r_n$ esetben az A együttható mátrix szimmetrikus. Tartsuk meg ezt az állapotot, de változtassunk az r_j értékén.

Pontosabb közelítéshez növelni kell r_j -t, de ez maga után vonja a kondíciószám gyors romlását. A hiba L^2 normája $r_j = 2.2$ -nél éri el a tízezredes nagyságrendet, és 3.6-nál válik az A mátrix közel szingulárissá. A mért mennyiségeket a következő táblázat tartalmazza 300 pontra.

r	condA	L2 norma
0.001*ones(p,1)	1,14090E+00	0.96165
0.01*ones(p,1)	2,49170E+00	0.70843
0.1*ones(p,1)	3,12959E+01	0.14368
0.2*ones(p,1)	1,60268E+02	0.065707
0.4*ones(p,1)	2,49892E+03	0.0357
0.6*ones(p,1)	2,83611E+04	0.024015
0.8*ones(p,1)	2,65569E+05	0.015808
1*ones(p,1)	2,17975E+06	0.010073
1.2*ones(p,1)	1,62987E+07	0.0063063
1.4*ones(p,1)	1,13827E+08	0.0039574
1.6*ones(p,1)	7,54869E+08	0.0025383
1.8*ones(p,1)	4,83906E+09	0.0016839
2*ones(p,1)	3,06086E+10	0.001153
2.2*ones(p,1)	1,88479E+11	0.00080427
2.4*ones(p,1)	1,12082E+12	0.00056245
2.6*ones(p,1)	6,44689E+12	0.00038928
2.8*ones(p,1)	3,59740E+13	0.00026415
3*ones(p,1)	1,95283E+14	0.00017447
3.2*ones(p,1)	1,03234E+15	0.00011138
3.4*ones(p,1)	5,36014E+15	6.8092e-005
3.6*ones(p,1)	2,43886E+16	3.9371e-005
3.8*ones(p,1)	1,17352E+17	2.0956e-005
4*ones(p,1)	9,12917E+17	9.6339e-006
5*ones(p,1)	6,98407E+18	1.244e-005

3.2.7. ábra: Skálázó paraméterek konstans megválasztásával kapott teszteredmények sin(x)*sin(y) -ra

Mint ahogy korábban azt a MQ-nál tettük, figyeljük meg, hogyan hat az interpolánsra, ha r_j-ket egy intervallumból választjuk. Mivel $r_j = 2.6$ jó eredményt adott, ezért először ennek környezeteit választom a forrás intervallumoknak. Majd szélesebb skáláról választva tesztelem az interpolánst, figyelembe véve, hogy a $sin(x) \cdot sin(y)$ felülete változatos formájú, így különböző alakú bázisfüggvények kombinációja jobb megoldást adhat.

r	condA	L2 norma
linspace(2,3,300)	1,74908E+13	0.000592
linspace(3,2,300)	7,87090E+13	0.00043072
linspace(1,4,300)	9,66496E+14	0.0001068
linspace(4,1,300)	1,90763E+15	0.00062493
linspace(2.2,3,300)	4,90317E+13	0.00030221
linspace(3,2.2,300)	2,28692E+14	0.00021585
linspace(1.5,2.5,300)	1,80178E+12	0.0032828
linspace(2.5,1.5,300)	1,97613E+12	0.0084435
linspace(1.8,2.2,300)	2,55497E+11	0.0021058
linspace(2.2,1.8,300)	1,56694E+11	0.0052655

3.2.8. ábra: Skálázó paraméterek monoton variációjával végzett tesztek eredményei sin(x)*sin(y) -ra

A módosítások nem jártak eredménnyel, a paraméter változatossága inkább rontott a kondíciószámon, és a pontosságot sem növelte a konstans r_j értékekhez képest.

Ezek után teszteljük az RMQ módszert problémás függvényre. $f(x, y) = abs(x) + abs(y)$ függvény az origoban nem differenciálható, valamint élekkel és lapfelületekkel rendelkezik. Megfigyelhető, hogy kisebb ponthalmazokra pontosabb a közelítés, mint sima függvény esetén, de a pontok számának növelésével a hibanorma csökkenése sokkal lassabb, szinte konstans p=650 pontig. Emiatt nagy ponthalmazra rosszabbul viselkedik.

p	L2 norma (abs)	L2 norma (sin)
10	0.38968	1.0073
50	0.14858	0.54021
100	0.15301	0.47573
150	0.054591	0.1131
200	0.029818	0.088325
250	0.029166	0.064882
300	0.019616	0.067765
350	0.016033	0.026516
400	0.018753	0.032341
450	0.012534	0.026975
500	0.016884	0.028224
550	0.022453	0.048024
600	0.014908	0.016982
650	0.0062187	0.002566
700	0.0077424	0.0044441
750	0.0072268	0.0048225
800	0.01401	0.014586
850	0.0065386	0.0040791
900	0.011116	0.009787
950	0.0064168	0.0018533
1000	0.0086997	0.0020089
1100	0.0062042	0.0012567
1200	0.020237	0.0015333
1300	0.0092931	0.0018809
1400	0.0076081	0.00040499

3.2.9. ábra: RMQ kisebb ponthalmazra jobban megoldja a problémás abs(x)+abs(y) approximációját

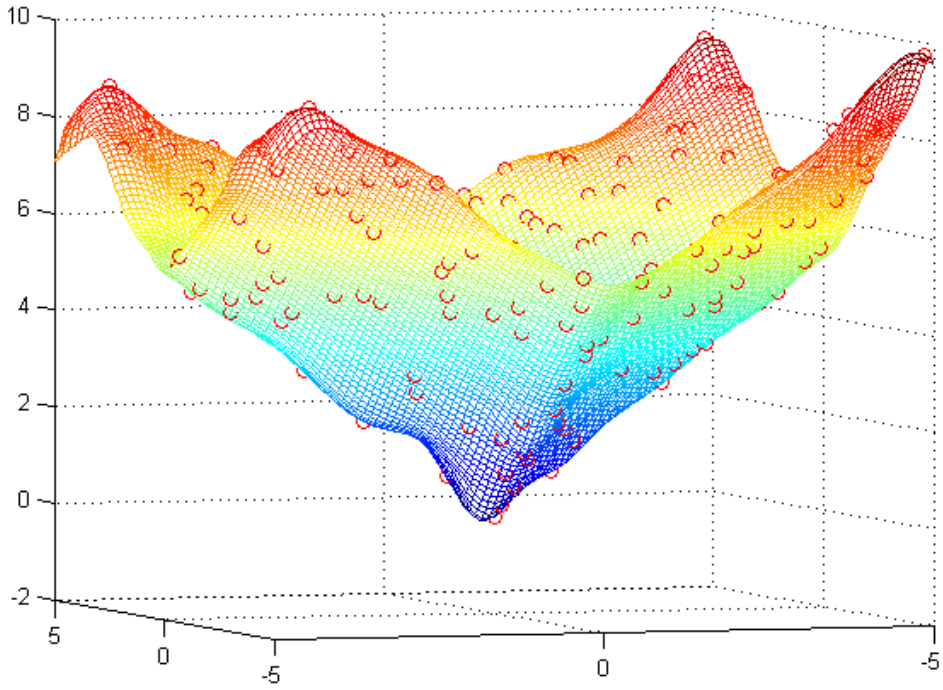
Természetesen $epsz$ növelése itt is jó hatással lesz A kondícionáltságára. Tizedenként emelve értékét tizedére csökken a kondíciószám, 0.4-es távolsággal meghatározott 300 pontra 2,20855E+04-re esik le, a hiba L^2 normája pedig 0.009 ezredre. Hasonlóan viselkedik 500 pontra is.

epsz	condA	Hibaátlag	L2 norma
0.01	8,74847E+08	-0.00036286	0.016884
0.1	9,18920E+06	-0.00025776	0.013519
0.2	1,51111E+06	4.2929e-006	0.014707
0.3	2,66420E+05	-1.3863e-006	0.0015973

3.2.10. ábra: epsz értékét tizedenként növelve, a hiba L normája tizedére esik vissza abs(x)+abs(y) függvénynél

A skálázó paraméter növelése azonban nem hozott jó eredményt. Az interpoláció pontossága végig 0.2 körül mozog 300 pontos feladatra. Ez esetben a sima $sin(x) \cdot sin(y)$ függvényhez képest $r_j < 0.2$ –re mutatott jobb közelítést, és r_j további variációja a kondíciószám erős romlásához vezetett.

Mivel az RMQ módszer lekerekítette az éleket és a csúcsot, további néhány pontot vettem fel a csúcs környezetében. Ez az együttható mátrixot olyan rosszul kondícionálttá tette, hogy a Matlab nem tudta az interpoláns értékeit kiszámolni.



3.2.11. ábra: f(x,y)=abs(x)+abs(y) függvény RMQ interpoláltja

Utolsó tesztfüggvényünk legyen $f(x,y) = exp(x+y)$ a [-2,2,-2,2] értelmezési tartományon.

condA	L2 norma (sin)	p	condA	L2 norma (exp)
4,38369E+06	0.0057575	100	4,38369E+06	0.037245
3,40318E+08	0.001085	200	3,40318E+08	0.028911
7,65411E+09	0.00071128	300	7,65411E+09	0.041992
3,44892E+10	0.00026009	400	3,44892E+10	0.011143
2,02550E+11	0.00034362	500	2,02550E+11	0.0035584
7,55654E+11	0.00011914	600	7,55654E+11	0.0012228
4,00776E+12	4.5267e-005	700	4,00776E+12	0.00024461
1,38541E+13	5.1816e-005	800	1,38541E+13	0.0013302
3,45948E+13	3.6253e-005	900	3,45948E+13	0.0014594
3,45944E+14	6.9545e-006	1000	3,45944E+14	0.00014908

3.2.12. ábra: RMQ rosszabbul teljesít a nagy felületen enyhe gradiensű, majd gyorsan változó exp(x+y) függvényre, mint sin(x)*sin(y) esetében

Látható, hogy A kondíciószáma nem függ a függvényértékektől, azonban az exponenciálishoz hasonló enyhe gradiensű felületet sokkal rosszabbul közelíti az RMQ, mint egy $sin(x) \cdot sin(y)$-hoz hasonló, meredek gradiensű függvényt.

Azt várnám, hogy a nagy felületen lapos függvényhez nagy r_j értékek adnak jó eredményt, helyette a skálázó paraméter 1 körüli értékre ad optimális pontosságot. A 700×700 -as együtthatómátrix kondíciószáma nagyon gyorsan emelkedik r_j növelésével, r_j=1.4-nél A majdnem szinguláris. Rossz eredményt ad az is, ha r_j különböző értékeket vesz fel.

r	condA	L2 norma
0.001*ones(p,1)	1,52240E+00	0.85243
0.01*ones(p,1)	6,90530E+00	0.42671
0.1*ones(p,1)	3,35143E+02	0.071311
0.2*ones(p,1)	7,01392E+03	0.030946
0.4*ones(p,1)	1,63679E+06	0.0064231
0.6*ones(p,1)	2,64949E+08	0.0015513
0.8*ones(p,1)	3,53735E+10	0.00053993
1*ones(p,1)	4,00776E+12	0.00024461
1.2*ones(p,1)	3,93610E+14	0.00011296
1.4*ones(p,1)	3,95127E+16	5.041e-005
1.6*ones(p,1)	9,45541E+18	2.2316e-005
1.8*ones(p,1)	2,20216E+19	2.7706e-005
2*ones(p,1)	8,07020E+19	7.4702e-006

3.2.13. ábra: Skálázó paraméter variációja nem pontosítja exp(x+y) közelítését

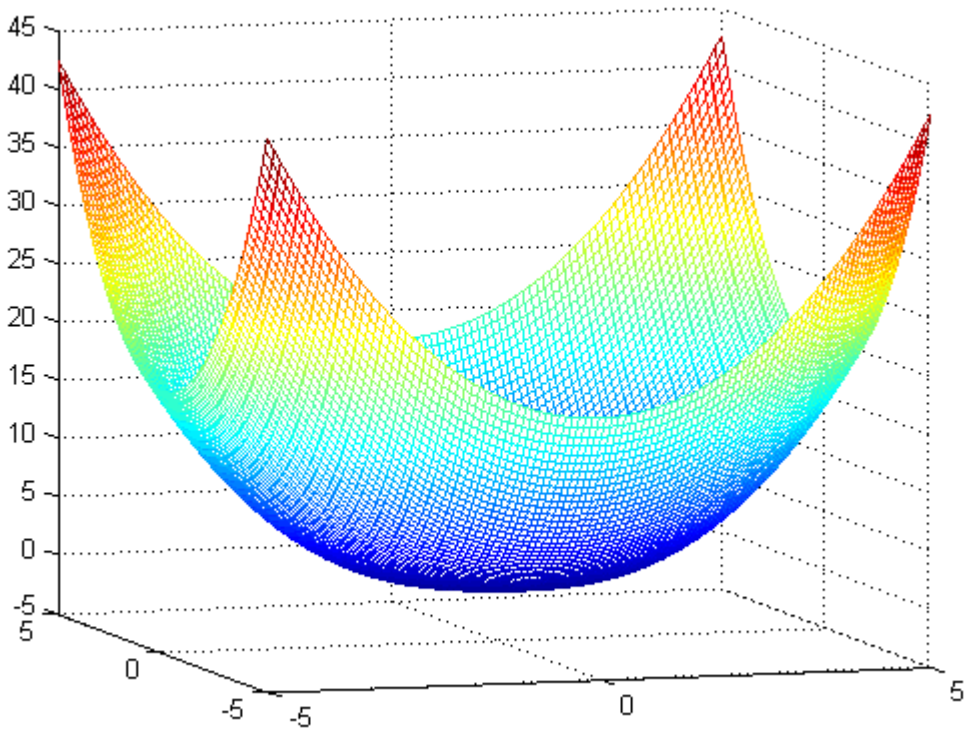
3.3 Thin Plate Spline

$$\begin{pmatrix} \Phi_1(x_1 - x_1) & \cdots & \Phi_n(x_1 - x_n) \\ \vdots & \ddots & \vdots \\ \Phi_1(x_n - x_1) & \cdots & \Phi_n(x_n - x_n) \end{pmatrix} * \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}$$

ahol $\Phi_j(x) = \|x\|^2 \cdot log\|x\|$.

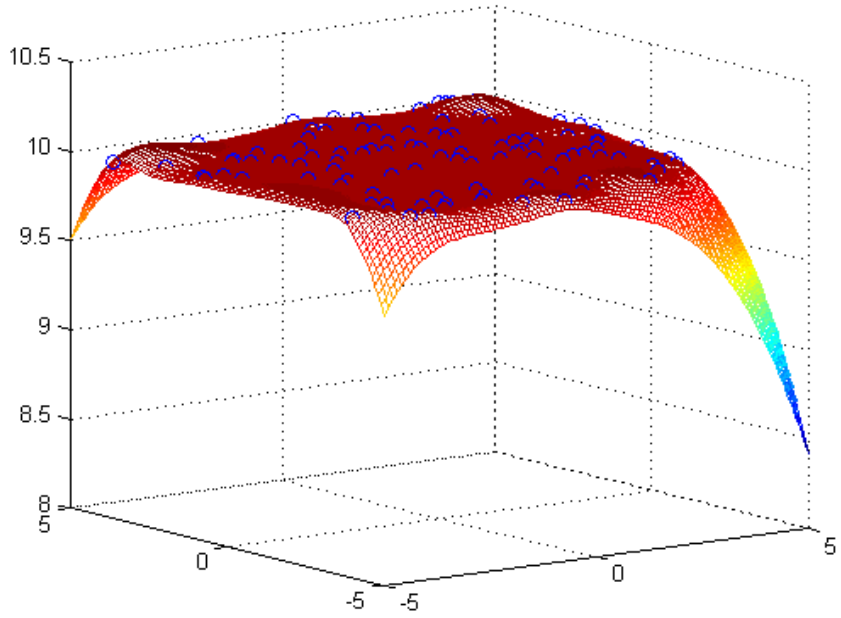
Fontos itt az elején kiemelni, hogy ebben az eljárásban nincsen skálázó paraméter. Teljesítménye és viselkedése matematikailag jobban megfogható a már bemutatott módszereknél. Bizonyítható, hogy $g(r) = r^2 \cdot log(r)$ függvény minimalizálja az $\int_{\mathbb{R}^2} \left(\left| \frac{\partial^2 f}{\partial x^2} \right|^2 + 2 \cdot \left| \frac{\partial^2 f}{\partial x \partial y} \right| + \left| \frac{\partial^2 f}{\partial y^2} \right|^2 \right) dxdy$ thin plate funkcionált, innen kapta a nevét.

A következő kép a bázisfüggvényt ábrázolja a [-5,5,-5,5] értelmezési tartományon.



3.3.1. ábra: TPS bázisfüggvény a [-5,5,-5,5] értelmezési tartományon

Igaz-e, hogy a TPS módszer nem polinomiálisan pontos? Interpoláljuk az $f(x, y) = 10$ konstans függvényt. Legyen É.T.=[-5,5,-5,5], az alappontok egymástól való minimum távolsága $epsz$=0.01 és futtassuk 100 interpolációs pontra. Látható, hogy nem tudja hiba nélkül approximálni $f(x, y)$ konstans függvényt, az együttható mátrix kondíciószáma 5,11631E+06, az átlagos hibanagyság -0.033754 és L^2 normája 1.405.



3.3.2. ábra: TPS nem polinomiálisan pontos eljárás

$f(x,y) = sin(x) \cdot sin(y)$ esetén már sokkal szebb értékeket kapunk. Rögzítve a pontok egymástól vett minimum távolságát $epsz$=0.01-re, egyre nagyobb ponthalmazokat generáltam egymástól függetlenül a [-5,5,-5,5] tartományon. A TPS futtatásával a megoldandó egyenletrendszer együttható mátrixa rendkívül jól kondícionált, 300-tól 1000 alappontig tartja a 10E+07 nagyságrendet. Ellenben a hiba relatív diszkrét L^2 normája lassan éri el a 0.006 ezredes értéket, csak 650 alappontnál és a számítások sok időt vesznek igénybe.

p	condA	Hibaátlag	L2 norma	Felhasznált idő
10	9,60727E+02	6.5968	1.1986	2.5655
50	1,41346E+05	-0.22939	0.67749	10.5891
100	5,11631E+06	0.078811	0.62907	21.3782
150	2,03271E+06	0.0015209	0.071087	31.7709
200	7,97378E+06	-0.00067913	0.061859	42.125
250	7,62071E+06	-0.00013504	0.043336	53.0808
300	1,22575E+07	-0.00012102	0.048544	62.4446
350	3,23813E+07	0.00010705	0.022368	74.4714
400	4,60281E+07	2.8454e-005	0.018069	88.407
450	1,37358E+07	4.4142e-005	0.020715	96.4462
500	8,79154E+07	-9.9056e-006	0.015749	114.9143
550	9,16591E+07	-1.5722e-005	0.021089	126.9155
600	5,88001E+07	-5.0979e-006	0.015241	125.7398
650	7,61833E+07	-2.638e-006	0.0061683	134.3118
700	1,72133E+08	-3.7925e-006	0.007822	150.8566
750	2,66468E+08	-6.1782e-006	0.0088834	156.3857
800	8,17721E+07	-2.3821e-006	0.012771	181.095
850	8,93082E+07	1.1721e-006	0.0067291	188.5285
900	1,10695E+08	1.1734e-006	0.011647	199.0296
950	1,22134E+08	-1.2364e-006	0.0060646	216.1812
1000	9,58298E+07	-2.9756e-007	0.0084233	230.398

3.3.3. ábra: TPS teszteredményei sin(x)*sin(y) függvény esetén különböző nagyságú ponthalmazokra

A következő táblázatok azt mutatják, hogy az $epsz$ növelése tovább javítja a kondíciószámot és ebből kifolyólag néhány századdal a pontosságot is.

p=250:

epsz	condA	Hibaátlag	L2 norma
0.01	7,62071E+06	-0.00013504	0.043336
0.1	1,09206E+06	-0.00060789	0.051014
0.2	5,41609E+05	-0.000355	0.033892
0.3	3,29717E+05	-0.00028066	0.031724
0.4	2,19937E+05	1.7119e-006	0.015398
0.5	1,66299E+05	-3.6929e-005	0.010116

p=500:

epsz	condA	Hibaátlag	L2 norma
0.01	8,79154E+07	-9.9056e-006	0.015749
0.1	3,43466E+06	-2.1005e-006	0.016003
0.2	1,41348E+06	5.8006e-006	0.011359
0.3	8,57455E+05	8.8115e-007	0.0049873

3.3.4. ábra: az alappontok egymástól vett távolságának növelése javítja az együttható mátrix kondíciószámát

Hasonlóan a MQ és RMQ interpolációhoz készült mérés az $f(x,y) = abs(x) + abs(y)$ problémás függvényről, először adott $epsz$=0.01 mellett. Az az érdekes helyzet merült fel, hogy pontosabb közelítést kaptunk egy olyan függvényre, aminek létezik nem differenciálható pontja, minta a szép sima $sin(x) \cdot sin(y)$ -ra. Erre a magyarázat az utóbbi felület változatosságában keresendő.

L2 norma (sin)	p	L2 norma (abs)
1.1986	10	0.10295
0.67749	50	0.033109
0.62907	100	0.020199
0.071087	150	0.013376
0.061859	200	0.011428
0.043336	250	0.0088849
0.048544	300	0.0088128
0.022368	350	0.0086333
0.018069	400	0.0077118
0.020715	450	0.0069216
0.015749	500	0.0066096
0.021089	550	0.0047075
0.015241	600	0.0068587
0.0061683	650	0.0049006
0.007822	700	0.0053459
0.0088834	750	0.004538
0.012771	800	0.0036005
0.0067291	850	0.004187
0.011647	900	0.0035214
0.0060646	950	0.003557
0.0084233	1000	0.0040496

3.3.5. ábra: TPS pontosabban közelíti abs(x)+abs(y) függvényt, mint sin(x)*sin(y) függvényt

Ezután javítsunk A kondícionáltságán $epsz$ növelésével. Megfigyelhető, hogy az alappontok távolságának növelésekor azok egyre rendezettebben helyezkedtek el a szűk tér miatt, ezért a felület egésze általánosan pontosabb lesz, mert mindenhol egyenlő mennyiségű információt tud hasznosítani a TPS módszer.

p=250:

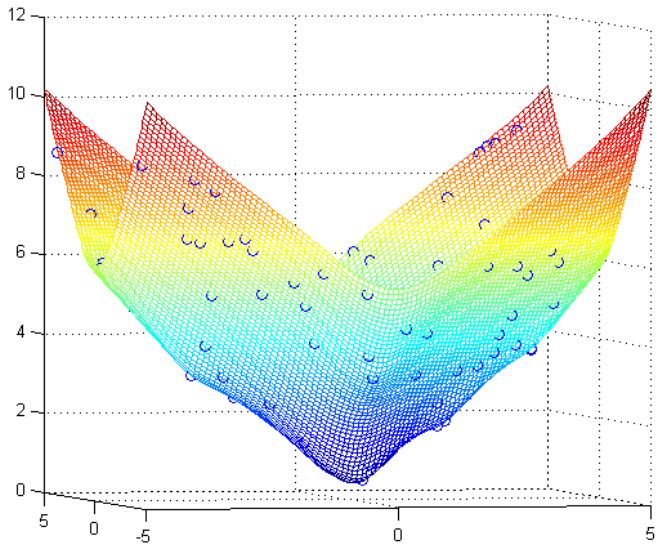
epsz	condA	Hibaátlag	L2 norma
0.01	7,62071E+06	0.00077533	0.0088849
0.1	1,09206E+06	0.00042412	0.010049
0.2	5,41609E+05	1.1554e-005	0.0092698
0.3	3,29717E+05	-3.692e-005	0.0083339
0.4	2,19937E+05	0.00052194	0.0077872
0.5	1,66299E+05	-0.00046988	0.0062102

p=500:

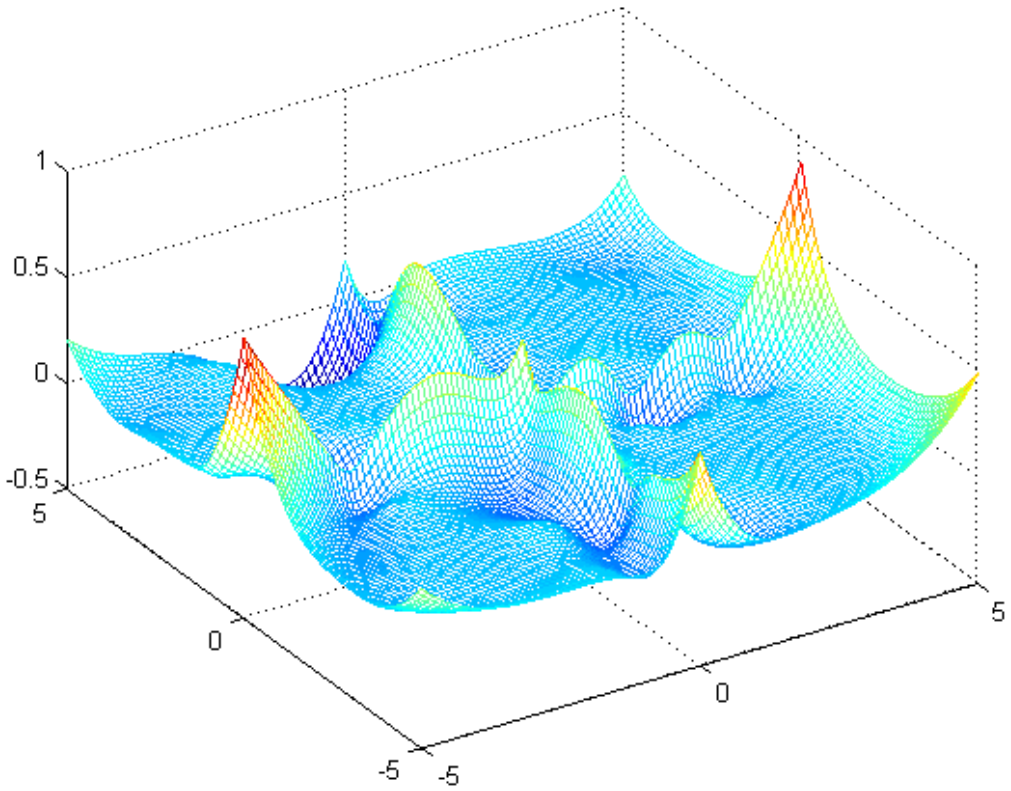
epsz	condA	Hibaátlag	L2 norma
0.01	8,79154E+07	9.6646e-005	0.0066096
0.1	3,43466E+06	6.7492e-005	0.0056895
0.2	1,41348E+06	5.9864e-005	0.0046161
0.3	8,57455E+05	6.45e-006	0.0041406

3.3.6. ábra: epsz növelésével kapott eredmények 250 és 500 alappontra TPS esetén

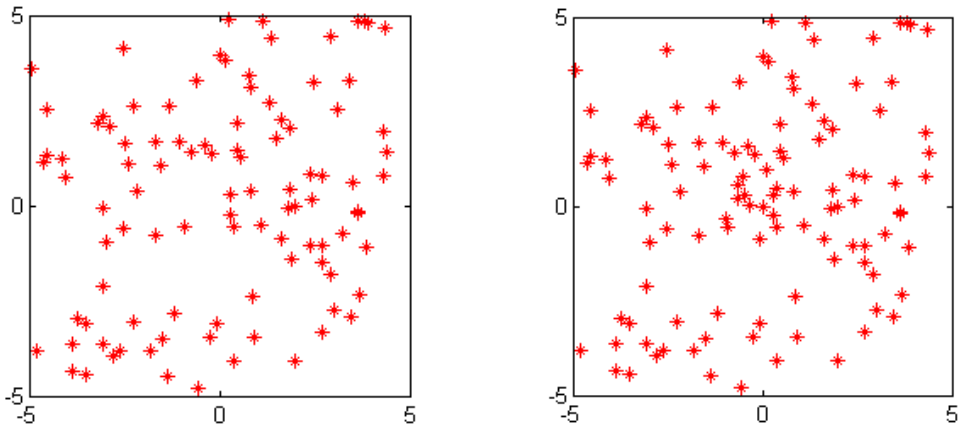
Ennek ellenére az interpoláns csúcsában továbbra is lekerekített élek futnak össze, így meg lehet próbálni a kritikus rész környezetében újabb alappontokat felvenni és úgy approximálni. Az első két ábra 100 pontra mutatja a közelítés eredményét és hibáját, majd 10 pont hozzá vételével új rajz készült. Látszik, hogy az origonál szűkül az alakzat, és a korábban pozitív hiba inkább negatív irányba modult el. Fontos hozzátenni, hogy ha kis területen növeljük az interpolációs pontok számát, akkor a kondíciószám nagyon megugrik.



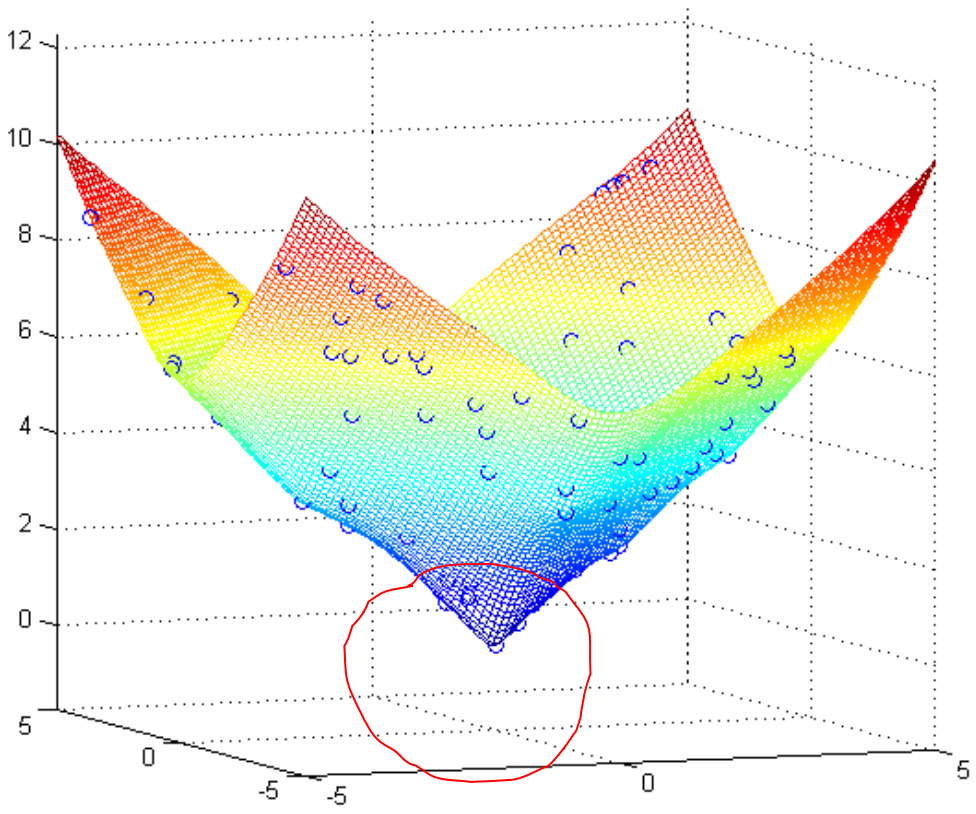
3.3.7. ábra: f(x,y)=abs(x)+abs(y) függvény TPS interpoláltja



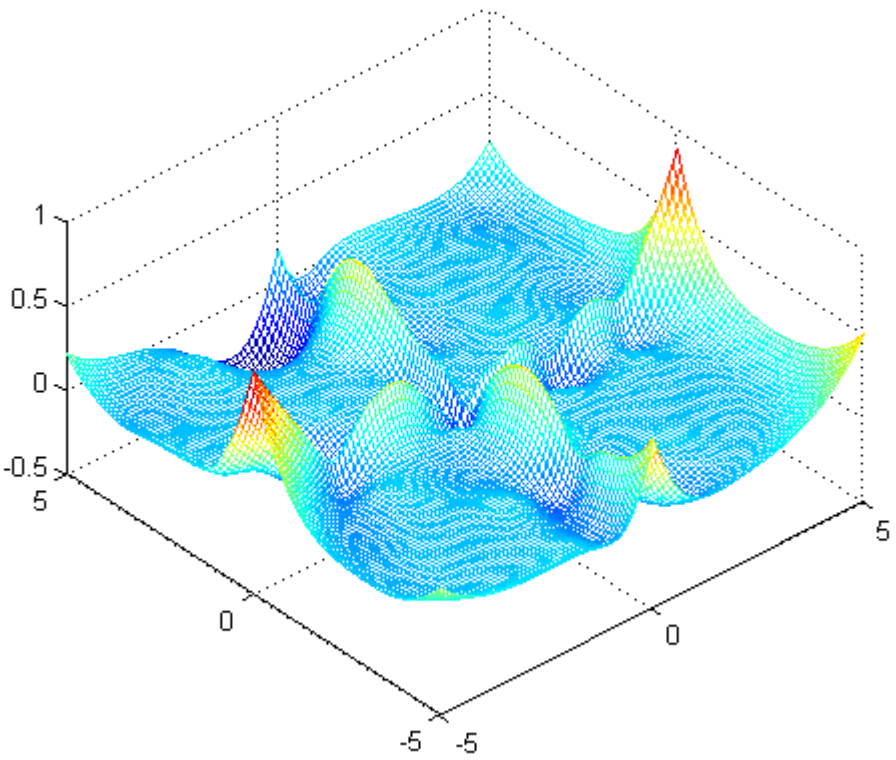
3.3.8. ábra: f(x,y)=abs(x)+abs(y) függvény interpoláltjának hibafüggvénye



3.3.9. ábra: abs(x)+abs(y) pontosabb közelítéséhez az origo körül 10 további pontot vettünk fel



3.3.10. ábra: Szemmel látható eredményt hozott a pontok sűrítés origo körül



3.3.11. ábra: Függvény és interpolánsának különbsége pozitívról negatívra változott

Most válasszuk kisebbre az értelmezési tartományt, és közelítsük $f(x,y) = exp(x+y)$ függvényt [-2,2,-2,2] tartományon $epsz$=0.1 mellett.

L2 norma (sin)	p	condA	L2 norma (exp)
0.62907	**100**	7,13968E+04	0.054738
0.017443	**200**	1,37906E+05	0.036544
0.0093701	**300**	3,19346E+05	0.053825
0.0054286	**400**	4,21456E+05	0.032119
0.0059789	**500**	5,21223E+05	0.0138
0.0032701	**600**	6,94579E+05	0.0070672
0.0014824	**700**	8,85484E+05	0.0031655
0.002228	**800**	1,10338E+06	0.0099648
0.002018	**900**	1,25449E+06	0.012923
0.00082229	**1000**	1,49323E+06	0.0026685

3.3.12. ábra: TPS gyengébben közelíti sin(x)*sin(y)-nál az exp(x+y) függvényt

Jól olvasható a táblázatból, hogy ennek a függvénynek a közelítése TPS esetén is kisebb pontossággal végezhető, mint a $sin(x) \cdot sin(y)$ esetében. Az ezredes hibanormát csak 600 pontra éri el.

Az alappontok egymástól való minimális távolságának növelése a kondíciószámot enyhén csökkentette, de emellett a pontosság ingadozó volt és csak a százados nagyságrendet érte el.

További néhány interpolációs pont felvétele után a [0,5,0,5] tartományon a kondíciószámot százszorosára növelte, és kicsit javított az interpoláció pontosságán.

4.Fejezet

Összegzés

A Multiquadrics módszer Hardytól származik, aki először 1968-ban használta földrajzi felületek, valamint gravitációs- és mágneses szabálytalanságok közelítésére. A módszer akkor vált ismertté, mikor Franke publikációjában a szórt alappontú interpolációs eljárásokról értekezett. Valójában a MQ egy variációja a RMQ. A Thin Plate Spline eljárás ötlete Harder és Desmarais munkáiban jelenik meg, bár ők felületi spline-nak nevezik eleinte, majd Duchontól kapott végleges formájával lett ismert. Lényegük, hogy az interpolációs függvényt különböző radiális bázisfüggvények határozzák meg. Lásd ábra..

Ezen dolgozat célja e három eljárás elemzése és összehasonlítása. Hogy hogyan teljesítenek különböző típusú függvényekre, hogyan lehet javítani a közelítések pontosságán és a megoldandó egyenletrendszer együttható mátrixának kondíciószámán. Teszteket futtattunk két Matlab program segítségével, függelék, kvázivéletlen generált alappontokra, adott értelmezési tartományon. Megfigyeltük a különböző változók módosításának az interpolációra kifejtett hatását, különös tekintettel MQ és RMQ esetén a skálázó paraméterekre. Mivel TPS módszer nem használ a felhasználó által megadott paramétert, a kísérletek során az interpolációs alappontok szisztematikus megválasztására kellett szorítkoznunk.

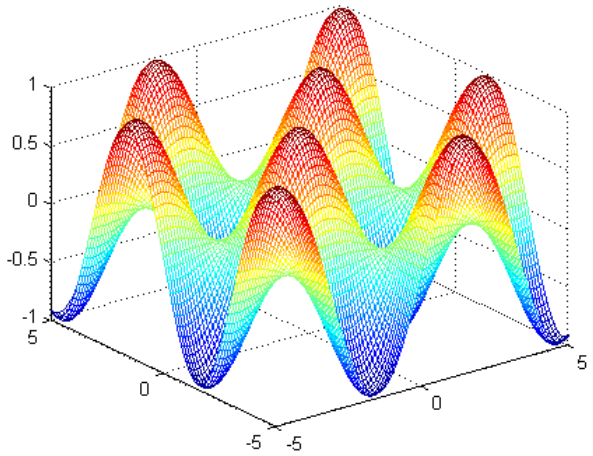
Mindhárom módszernél szükséges n alappontra a $\begin{pmatrix} \Phi_1(x_1 - x_1) & \cdots & \Phi_n(x_1 - x_n) \\ \vdots & \ddots & \vdots \\ \Phi_1(x_n - x_1) & \cdots & \Phi_n(x_n - x_n) \end{pmatrix} * \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}$ $n \times n$-es egyenletrendszer megoldása, ahol az együttható-mátrix teli mátrix. Hasonló lesz az algoritmusok tárigénye: MQ és RMQ esetén $\frac{1}{2} \cdot n \cdot (n+4)$, TPS esetén $\frac{1}{2} \cdot (n+3) \cdot (n+7)$, és műveletigényük $\sigma(n^3)$ -ös. Ez nagy ponthalmazra rengeteg erőforrást igényel, ezért saját számításaimat 1000 pont felett a számítógép korlátai miatt nem tudtam elvégezni.

Bár egyik eljárás sem bír polinomiális pontossággal, mégis jobbnak ítélik más módszereknél, önmagában tehát ennek nincs jelentősége, azonban létezik olyan algoritmus, aminek során MQ polinomiális precíziót nyer.

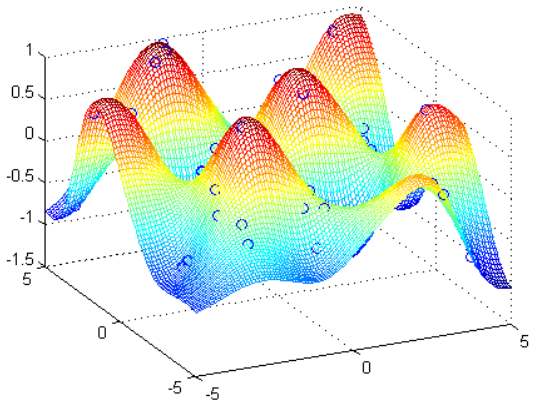
Az összehasonlíthatóság érdekében ugyanazt a 3 próbafüggvényt approximáltuk a 3 módszerrel: $f(x,y) = sin(x) \cdot sin(y)$, f(x,y)= $abs(x) + abs(y)$ és $f(x,y) = exp(x+y)$. Ennek célja, hogy sima, minden pontjában folytonosan differenciálható függvényen, egy pontjában nem differenciálható függvényen, valamint nagy felületen lapos, majd hirtelen változó függvényen keresztül is láthassuk a működésüket.

Ha két alappont közel van egymáshoz, akkor a rájuk felírt egyenletek, és így az A együttható mátrixban nekik megfelelő sorok is majdnem azonosak lesznek. Emiatt A közel szinguláris. Tehát szükséges az interpolációs pontok olyan megválasztása, hogy azok ne legyenek kollineárisak és egymástól való távolságuk minél nagyobb legyen. Ennek egy minimális értékét $epsz$ változóban határoztuk meg. Értékét 0.01-től 0.5-ig emelve A kondíciószáma TPS kivételével egységesen minden függvényre legalább az ezredrészére csökkent, sőt nagyobb ponthalmazra még nagyobb mértékben javult. A TPS-sel meghatározott együtthatómátrix már eleve sokkal jobban kondícionált volt, ezért nem tudott többet változni. Végeredményül általában 10^5 nagyságrendű kondíciószámot kaptunk.

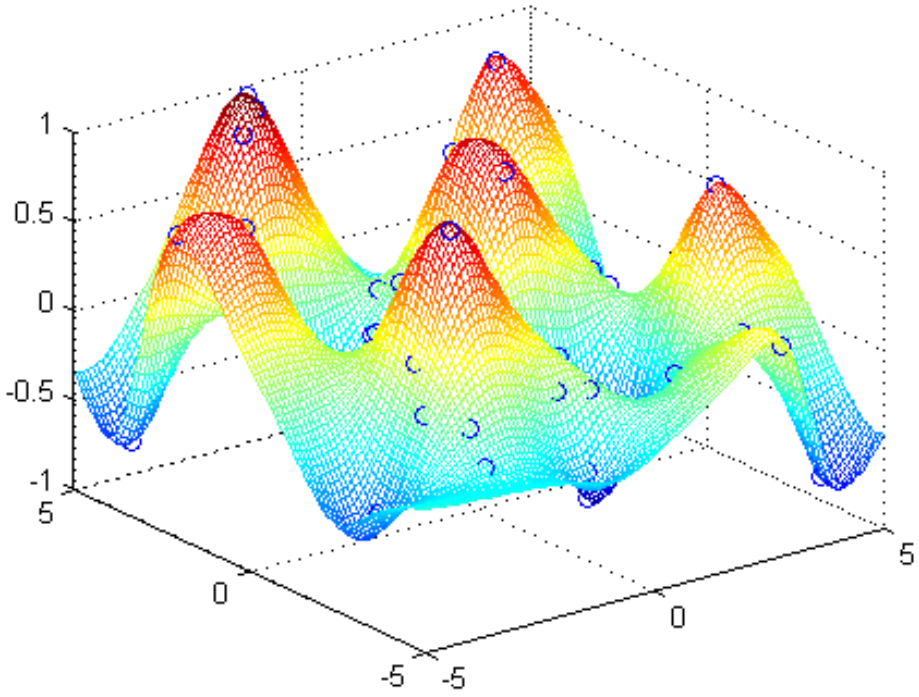
Nyilvánvaló, hogy az interpolációs alappontok számának növelésével pontosabb eredményt érünk el. A legkisebb ponthalmaz a [-5,5,-5,5] értelmezési tartományon, amire már felismerhetők a próbafüggvények általában 50 pont. Kivételt képez $f(x,y) = exp(x+y)$, ahol 10 pont is elegendő volt. A következő ábrák ezekre az alappont halmazokra hivatottak bemutatni az interpoláció módszerek eredményét vizuális tekintetben.



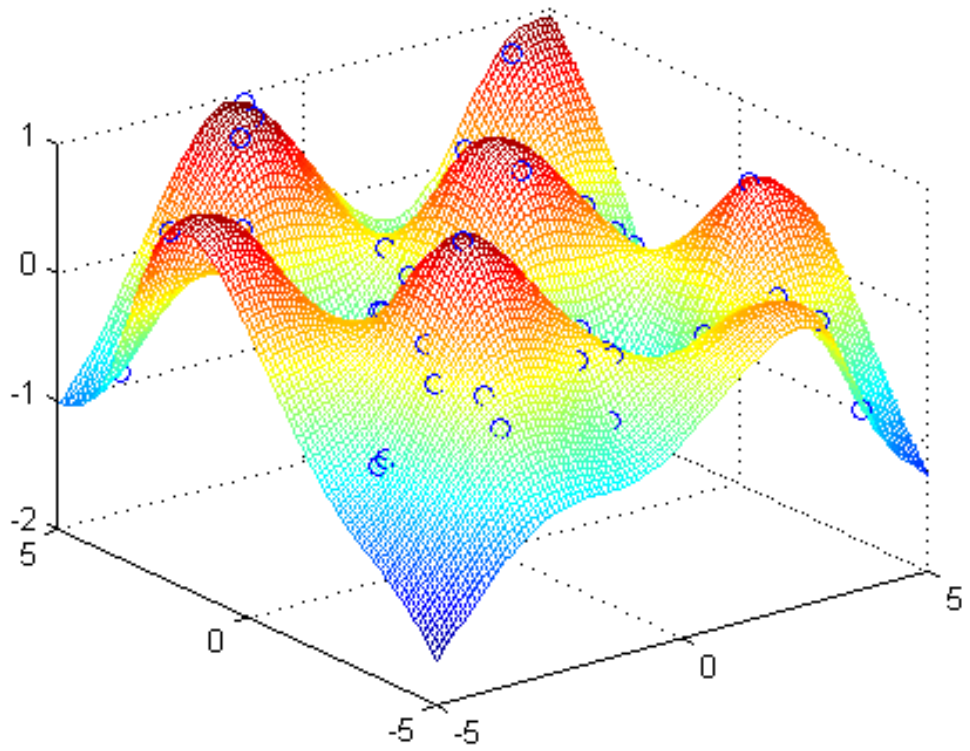
4.1. ábra: f(x,y)=sin(x)*sin(y)



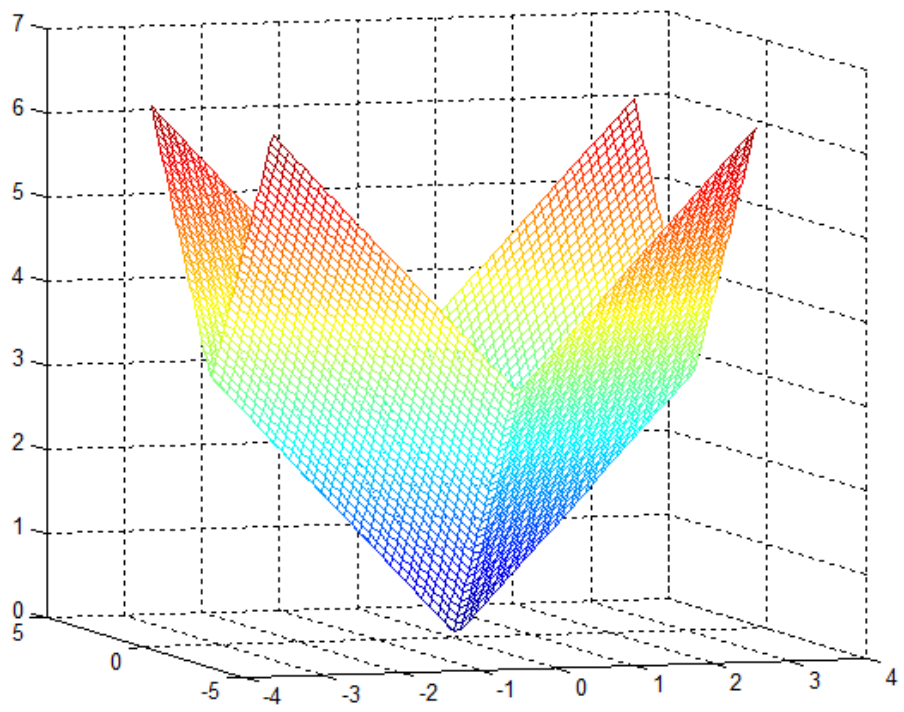
4.2. ábra: f(x,y)=sin(x)*sin(y) MQ interpolációja



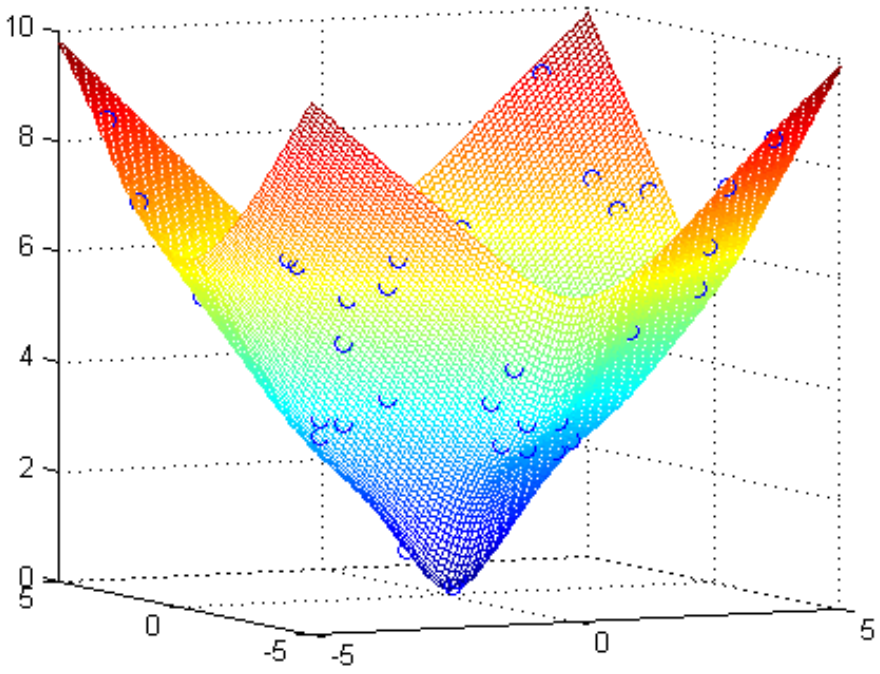
4.3. ábra: f(x,y)=sin(x)*sin(y) RMQ interpoláltja



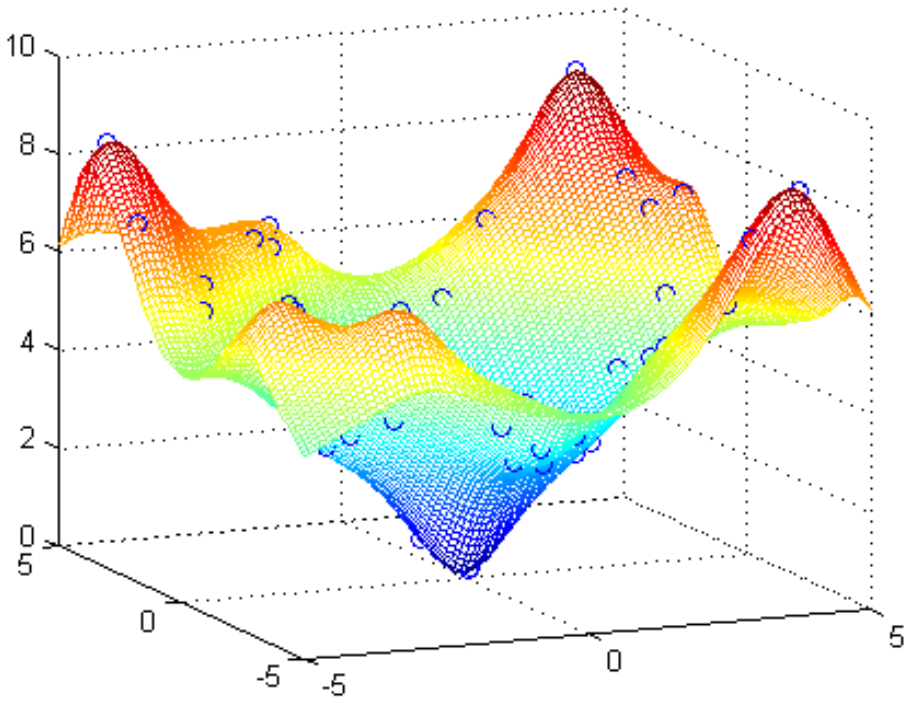
4.4. ábra: f(x,y)=sin(x)*sin(y) TPS interpoláltja



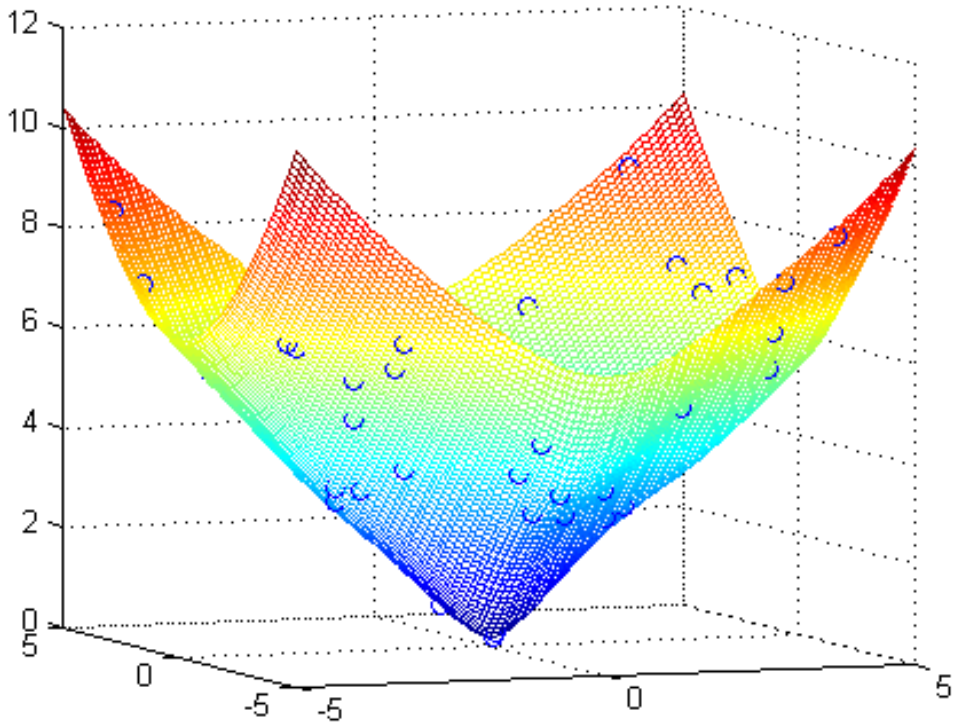
4.5. ábra: f(x,y)=abs(x)+abs(y)



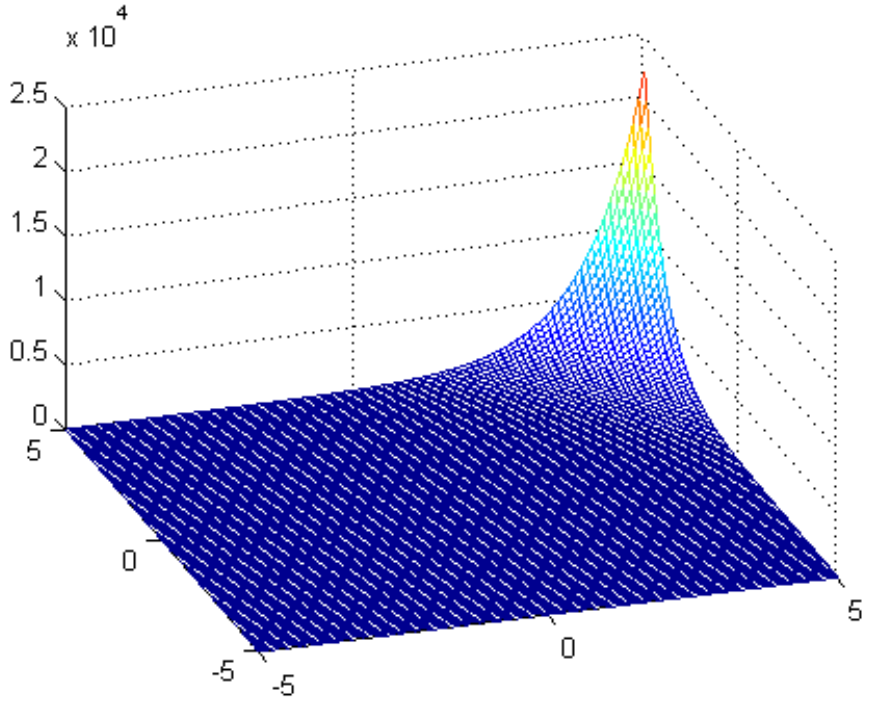
4.6. ábra: f(x,y)=abs(x)+abs(y) MQ interpoláltja



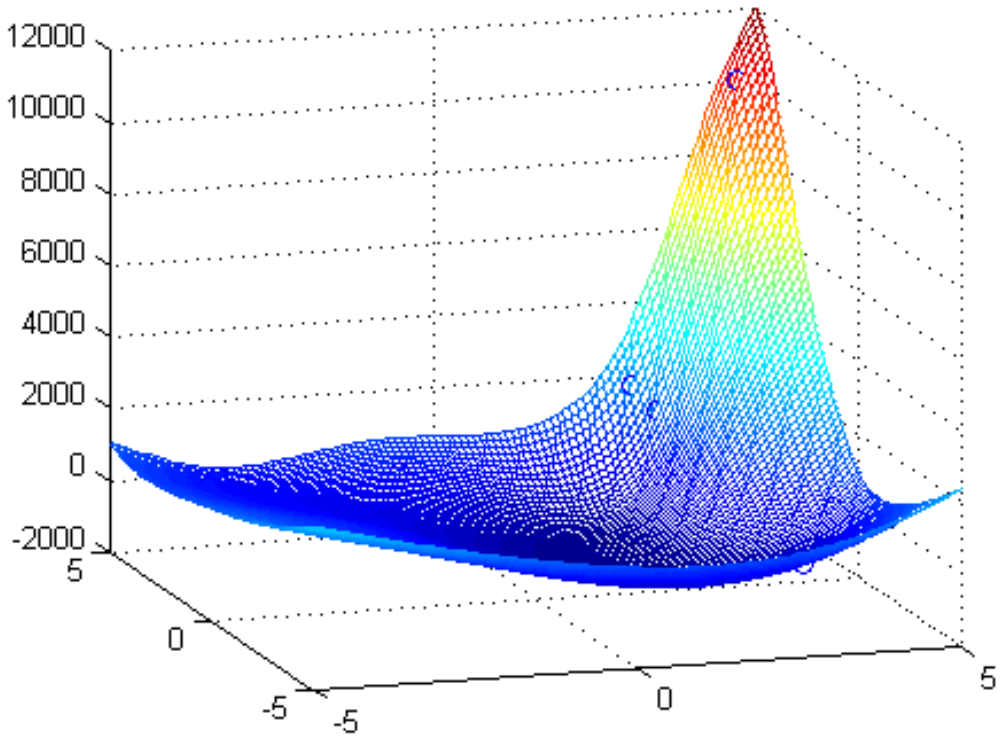
4.7. ábra: f(x,y)=abs(x)+abs(y) RMQ interpoláltja



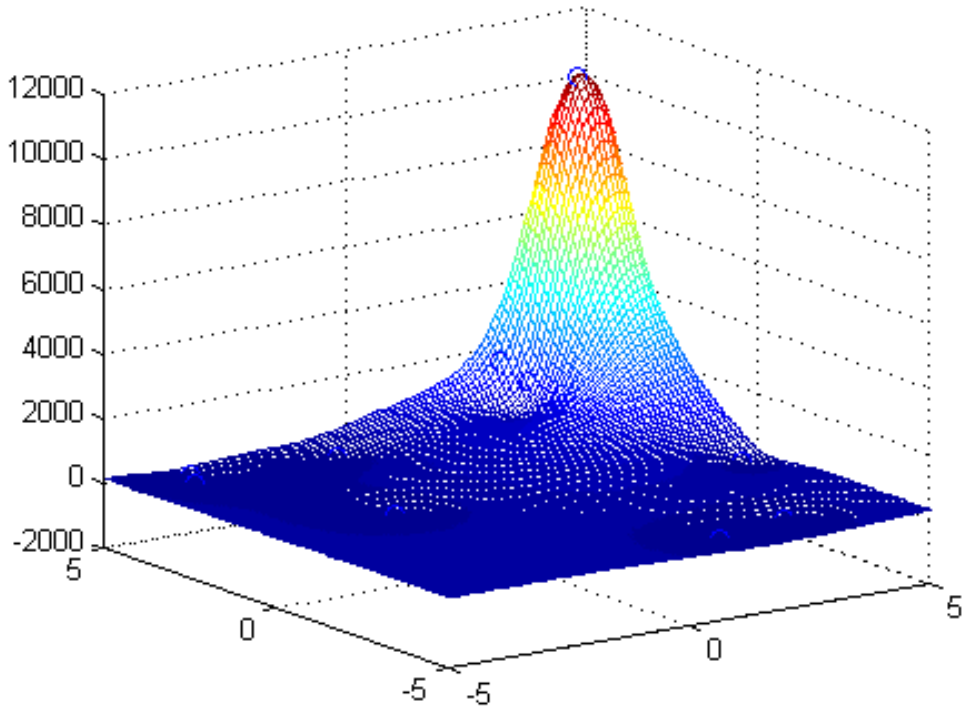
4.8. ábra: f(x,y)=abs(x)+abs(y) TPS interpoláltja



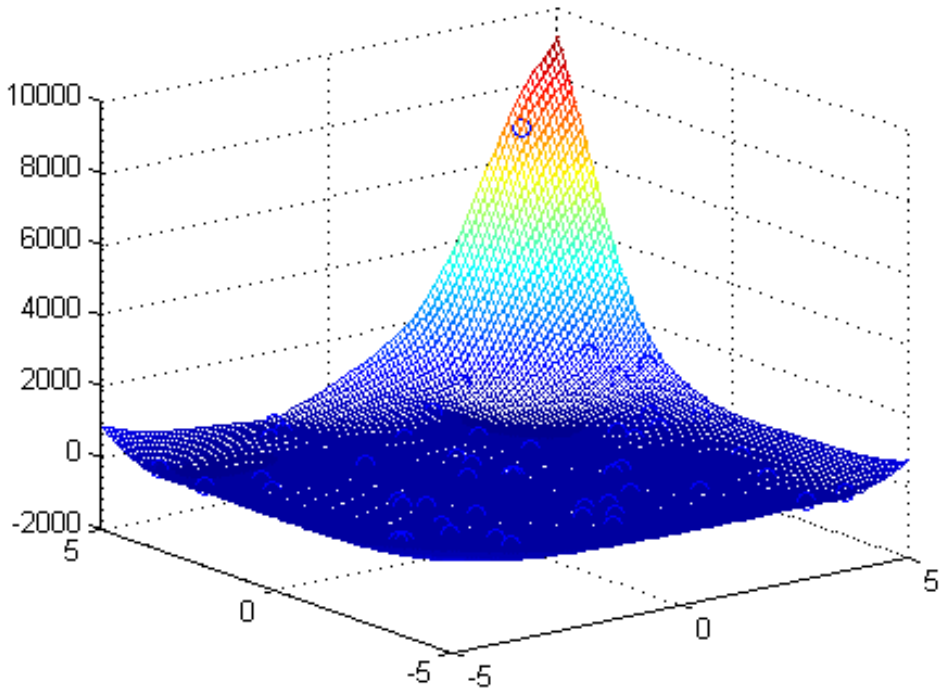
4.9. ábra: f(x,y)=exp(x+y)



4.10. ábra: f(x,y)=exp(x+y) MQ interpoláltja



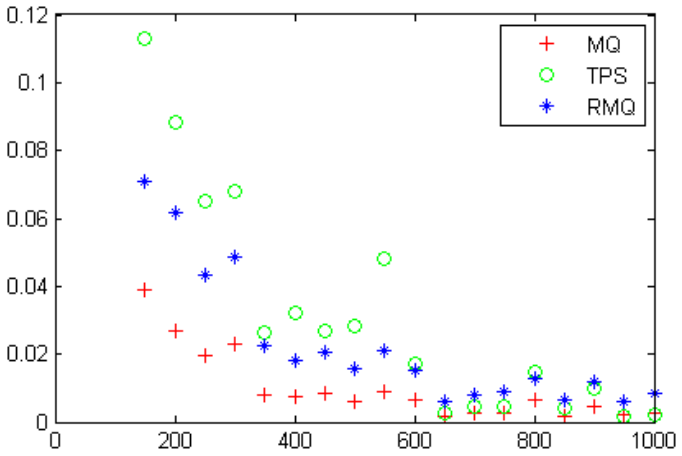
4.11. ábra: f(x,y)=exp(x+y) RMQ interpoláltja



4.12. ábra: f(x,y)=exp(x+y) TPS interpoláltja

A ponthalmaz méretének növekedésével a hiba relatív diszkrét L^2 normája csökken. A $sin(x) \cdot sin(y)$ közelítésében legpontosabbnak a MQ bizonyult, már 350 pontnál

elérte az ezrednyi nagyságrendet, míg a másik kettő csak 650-nél. A második legpontosabb a RMQ, de igaz lesz, hogy a kondícionáltság sorrendje fordított.



4.13. ábra: a ponthalmaz méretének függvényeként tekintve a kondíciószámot, az f(x,y)=sin(x)*sin(y) függvényt approximáló 3 radiális bázisfüggvény interpoláció közül a MQ a legpontosabb

Mivel a kondíciószám nem függ a függvényértékektől, csak az alappontoktól és a módszertől, ezért általában is igaz lesz, hogy a legjobban kondícionált együttható mátrixot TPS esetén kapjuk, és a legrosszabbat MQ-nál.

Az $abs(x) + abs(y)$ függvény az origoban nem differenciálható, meredek gradiensű oldallapjai ott futnak össze egy csúcsban. A hasonló, csúccsal vagy éllel rendelkező függvényeket nehéz jól approximálni olyan sima függvényekkel, mint a radiális bázisfüggvények, az eljárás lekerekíti őket. A korábbi képen is látszott, hogy ezt a feladatot a RMQ oldotta meg a legpontatlanabbul, minden ponthalmazra egyenletesen rosszul teljesít. Ellenben a másik kettő jól interpolálja, 500 pont alatt a TPS, a fölött inkább a MQ interpolánsa jobb.

Az $exp(x + y)$ függvény esetében a RMQ és a TPS eredményei közel azonosak, a MQ pedig kimagaslóan pontos közelítést ad, a hiba relatív diszkrét L^2 normája 10^{-4} nagyságrendű.

A különböző méretű interpolációs ponthalmazokon végzett mérések segítségével megmondható egy olyan halmazméret és olyan ponttávolság, ami a függvény típusának és az értelmezési tartomány nagyságának megfelelően optimális kondíciószámot és pontosságot hoz, jelen esetben ez 650-700 pontot jelent.

További korrekciót a skálázó paraméter segítségével értünk el MQ és RMQ esetén [3,4,10]. Általában véve a két módszer hasonlóan teljesít: r_j növelésével a radiális bázisfüggvények felülete egyre sekélyebb gradiensű lesz, „kilapul”. Azonban a nagy r_j érték a szingularitás felé vezet, mert az együtthatók széles skálán fognak mozogni, és előjelük változó lesz.

Tehát az interpolálandó függvény felületének megfelelő formájú bázisfüggvények kombinációját keressük. Az első méréseket rögzített r_j=1 mellett végeztük, azután pedig úgy módosítottuk az értékét, hogy A együttható mátrix szimmetriája megmaradjon, vagyis r_1=…=r_n. A $sin(x) \cdot sin(y)$ függvény estében az optimális értékek eltérnek, MQ-nál $r_j < 0.1$ -re a mérések nem nagyon változtak, r_j=3-ra a pontosság elérte a 10^{-5} nagyságrendet, majd hamarosan romlani kezdett, míg RMQ-nál kicsi hibanormát r_j=2.2 körül kaptunk és 3.6-ra majdnem szinguláris lett az együttható mátrix. Az $abs(x) + abs(y)$ függvénynél inkább a kis r_j értékek adtak jobb megoldást, ahogy az várható is volt, hiszen felülete meredek gradiensű. Az $exp(x + y)$ függvényt már kisebb értelmezési tartományon közelítettük, és nagy r_j értékekre vártunk pontosabb megoldást, végül r_j=1 körül számoltunk optimális mennyiségeket, és 1.4-től lett közel szinguláris az együttható mátrix.

Megfigyeltük azt az esetet is mikor A nem szimmetrikus, azaz a skálázó paramétereket egyenként határoztuk meg. Főleg olyan függvényeknél szerettünk volna jobb approximációt, amelyeknek változatos felülete van, mint pl. $sin(x) \cdot sin(y)$-nak. Itt a dolgozatban r_j-ket egy adott intervallum ekvidisztans osztópontjainak választottam, de több tanulmány készült egy olyan algoritmus kifejlesztésére, amely jó r_j értékeket generál. A MQ nehézsége valójában abban áll, hogy a mai napig csak irányelvek és tesztek segítségével tudunk egy-egy függvényre pontos közelítést adni, nincs bevált egységesen jó eljárás ezen paraméterek megválasztására [3,10]. Bár minden próbafüggvényre elvégeztem a kísérleteket, igazi eredményt csak $sin(x) \cdot sin(y)$-nál értem el széles skálán [0,5] mozgó r_j változókkal.

Utolsóként úgy próbáltunk az interpolációs feladat megoldásán javítani, hogy főleg nehezebben approximálható függvényeknél, kisebb ponthalmaz mellett a kritikus hely környezetében sűrítettük az alappontokat. Ez nyilván rontja A kondciószámát, de vizuálisan eredményesnek bizonyult pl. $abs(x) + abs(y)$ csúcsának közelítésénél.

Mindhárom módszerben felhasznált ötletek egyszerűen megvalósíthatóak, és nagyon sima, C^∞ interpolánst eredményeznek, melyek invariánsak a forgatásra és tükrözésre. Ahogy azt Franke elemzésében is olvastuk, minden szempontot figyelembe véve a Multiquadrics teljesített az összes szórt alappontú interpoláció közül a legjobban. Azonban megjegyzendő, hogy a második legjobban teljesítő Thin Plate Spline alkalmazása adott esetben vonzóbb lehet, hiszen nincsenek skálázó paraméterek és alapvetően jó kondíciószámot ad az együtthatómátrixnak.

Irodalomjegyzék

[1] *Gonzalo A. Ramos,Wayne Enright: Interpolation of Surfaces over Scattered Data, http://www.dgp.toronto.edu/~bonzo/docs/iasted-paper.pdf, 2013.jan.14.*

[2] *Gregory E. Fasshauer: On Smoothing for Multilevel Approximation with Radial Basis Functions, Charles K. Chui és Larry L. Schumaker (szerk.), Approximation Theory IX., Vanderbilt University Press, Nashville, 1-8.old.*

[3] *E. J. Kansa (1990): Multiquadrics – A Scattered Data Approximation Scheme with Applications to Computational Fluid-Dynamics – I., Computers and Mathematics with Applications, Pergamon Press,Great Britain, 121-145.old.*

[4] *Richard Franke (1982): Scattered Data Interpolation: Test of Some Methods*, Mathematics of Computation, American Mathematical Society, 181-200.old.*

[5] *Stoyan Gisbert, Takó Galina (2002): Numerikus módszerek 1., Typotex, Budapest*

[6] *Wikipédia (2009): Interpoláció, http://hu.wikipedia.org/wiki/Interpol%C3%A1ci%C3%B3, 2013.ápr.10.*

[7] *„Petur" (2005): Interpolációs eljárások, http://pixinfo.com/cikkek/interpolacio, 2013.márc.8.*

[8] *Autar Kaw: History of Interpolation, http://www.saylor.org/site/wp-content/uploads/2011/11/ME205-5.1-TEXT2.pdf, 2013.máj.5.*

[9] *J. C. Carr , R. K. Beatson , J. B. Cherrie , T. J. Mitchell , W. R. Fright , B. C. McCallum , T. R. Evans (2001): Reconstruction and Representation of 3D Objects with Radial Basis Functions, Computer Graphics, SIGGRAPH '01 Conf.Proc., 67-76.old.*

[10] *Ralph E. Carlson, Thomas A. Foley (1991): The Parameter R^2 in Multiquadric Interpolation, Computers and Mathematics with Applications, Pergamon Press,Great Britain, 29-42.old.*

Függelék

Segédprogramok a Radiális bázisfüggvények működésének prezentációjához (MATLAB)

RBFI.m RBF függvények kiszámolása, rajzolása, statisztikája

```
function RBFI(I,p,intxb,intxj,intyb,intyj,intzb,intzj,sur,x,y,z,s)
% Adott véletlen értékekre interpoláció RBF függvényekkel

% Meghívás pl.:
%   RBFI('M',20,0,5,0,5,-10,10,0.1,x,y,z,s)
%   ahol előzőleg már megadom x,y,z,s vektorokat
%   hogy később ugyanezekre az alappontokra tudjam futtatni akár
többször
%   is
%   pl.x=intxb+rand(p,1)*(intxj-intxb); és y=intyb+rand(p,1)*(intyj-
intyb);
%   z=sin(x).*sin(y); és s=unifrnd(0,2,p,1)
% Megjegyzés: a tesztek során az alapgen.m program segítségével
generált
%   x,y vektorokat töltöttem be a
%   save pxyeint p x y epsz intxb intxj intyb intyj; paranccsal

% Változók:
% I - RBF függvény típus: 'M'(MQ),'R'(Reciprocal MQ),'T'(Thin Plate
Spline)
% p - alappontok száma
% intxb - értelmezési tartomány (négyzet)->
% intxj - az x tengelyen bal és jobb végpont
% intyb - az y tengelyen az értelmezési tartomány->
% intyj - bal és jobb végpontjai
% intzb - az értékkészlet tartomány->
% intzj - lenti és fenti végpontja
% sur - alapháló sűrűsége pl. 0.1
%
% alappontok az értelmezési tartományból pl.véletlen
% x - x=intxb+rand(p,1)*(intxj-intxb);
% y - y=intyb+rand(p,1)*(intyj-intyb);
% alappontbeli értékek pl.véletlen vagy függvény
% z - z=unifrnd(intzb,intzj,p,1); vagy z=sin(x).*sin(y);
%
% s - skálázó paraméterek
%       TPS esetén mindig s=zeros(p,1);
%       MQ és RMQ esetén s(p,1) és s>=0

tic
% értelmezési tartomány hálója
h1=intxb:sur:intxj;
h2=intyb:sur:intyj;
[xi,yi]=meshgrid(h1,h2);
zim=length(h1);
zin=length(h2);

% Interpoláció
switch I
    case 'M'
        % bázisfüggvények együtthatóinak kiszámolása -> c
        A=zeros(p);
```

```
        for i=1:p
            for j=1:p
                A(i,j)=sqrt((norm([x(i) y(i)]-[x(j) y(j)]))^2+s(j)^2);
            end
        end
        disp(['condA=',num2str(cond(A))]);
        c=A\z;
        % függvényértékek kiszámolása az alaphálón -> zi
        zi=zeros(zim,zin);
        for i=1:zim
            for j=1:zin
                for k=1:p
                    zi(i,j)=zi(i,j)+c(k)*sqrt((norm([xi(i,j) yi(i,j)]-
[x(k) y(k)]))^2+s(k)^2);
                end
            end
        end
        disp(['toc1=',num2str(toc)]);
    case 'R'
        % bázisfüggvények együtthatóinak kiszámolása -> c
        A=zeros(p);
        for i=1:p
            for j=1:p
                r=sqrt((norm([x(i) y(i)]-[x(j) y(j)]))^2+s(j)^2);
                if r~=0
                     A(i,j)=1/r;
                else
                     A(i,j)=0;
                end
            end
        end
        disp(['condA=',num2str(cond(A))]);
        c=A\z;
        % függvényértékek kiszámolása az alaphálón -> zi
        zi=zeros(zim,zin);
        for i=1:zim
            for j=1:zin
                for k=1:p
                    r=sqrt((norm([xi(i,j) yi(i,j)]-[x(k)
y(k)]))^2+s(k)^2);
                    if r~=0
                        zi(i,j)=zi(i,j)+c(k)/r;
                    end
                end
            end
        end
        disp(['toc1=',num2str(toc)]);
    case 'T'
        % bázisfüggvények együtthatóinak kiszámolása -> c
        A=zeros(p);
        for i=1:p
            for j=1:i
                if i==j
                    A(i,j)=0;
                else
                    r=norm([x(i) y(i)]-[x(j) y(j)]);
                    if r==0
                        A(i,j)=0;
                        A(j,i)=A(i,j);
                    else
                        A(i,j)=r^2*log10(r);
```

```
                        A(j,i)=A(i,j);
                    end
                end
            end
        end
        disp(['condA=',num2str(cond(A))]);
        c=A\z;
        % függvényértékek kiszámolása az alaphálón -> zi
        zi=zeros(zim,zin);
        for i=1:zim
            for j=1:zin
                for k=1:p
                    r=norm([xi(i,j) yi(i,j)]-[x(k) y(k)]);
                    if r~=0
                        zi(i,j)=zi(i,j)+c(k)*r^2*log10(r);
                    end
                end
            end
        end
        disp(['toc1=',num2str(toc)]);
    otherwise
        disp('Rossz RBF megnevezés')
end

% a függvény kirajzolása
mesh(xi,yi,zi);
hold on;
% alappontbeli értékek kirajzolása
plot3(x,y,z,'bo');
disp(['toc2=',num2str(toc)]);
pause;
hold off;
f=sin(xi).*sin(yi); % f az aktuálisan közelítendő függvény
% hiba függvényének kirajzolása
mesh(xi,yi,zi-f);
h=zi-f;
% hiba relatív diszkrét L2 normájának számolása
L2norm=sqrt(sum(sum(h.^2)))/sqrt(sum(sum(f.^2)));
% eredmények kiíratása
disp(['min=',num2str(min(min(h))),', max=',num2str(max(max(h))),',
avg=',num2str(sum(sum(h)/p^2)),', L2norm=',num2str(L2norm)]);
```

alapgen.m Az interpoláció alappontjainak generálása és fájlba mentése

```
function [x,y]=alappgen(p,epsz,intxb,intxj,intyb,intyj,x1,y1)
%  Véletlen alappontokat generál az intxb, intxj, intyb, intyj által
határolt
%  értelmezési tartományban úgy, hogy egymástól vett távolságuk
legalább
%  epsz nagyságú legyen.

% x - alappontok első koordinátája
% y - alappontok második koordinátája
% p - alappontok száma
% epsz - legalább ekkora távolságra legyenek a pontok egymástól
% intxb - x tengely bal határ
% intxj - x tengely jobb határ
% intyb - y tengely bal határ
% intyj - y tengely jobb határ
```

```
% Annak érdekében, hogy később egy meglévő ponthalmazhoz további
pontokat
% generálhassunk:
% x1 - kiinduló alappontok első koordinátája, általában x1=[]
% y1 - kiinduló alappontok második koordinátája, általában y1=[]

%x(1,1)=intxb+rand(1)*(intxj-intxb);
%y(1,1)=intyb+rand(1)*(intyj-intyb);
x=x1;
y=y1;
r=p-length(x1);
for k=1:r-1
    u=intxb+rand(1)*(intxj-intxb);
    v=intyb+rand(1)*(intyj-intyb);
    i=1;
    while sum((u-x).^2+(v-y).^2<epsz^2)>0 || i==r % végtelen ciklus
elkerülésére
        u=intxb+rand(1)*(intxj-intxb);
        v=intyb+rand(1)*(intyj-intyb);
        i=i+1;
    end
    if i<r
        x(length(x1)+k+1,1)=u;
        y(length(x1)+k+1,1)=v;
    else
        disp('sikertelen probalkozas');
        x='rossz';
        y='rossz';
        return;
    end
end
% pontok kirajzolása
plot(x,y,'r*')
axis equal
axis square
% az x és y vektorokat lementem az ap.mat fájlba, ezt át kell majd
nevezni
save pxyeint p x y epsz intxb intxj intyb intyj;
```