Számítógépes Szimulációs Labor

5. Multigrid módszerek

Készítette:
Sztreha Balázs
Informatikus fizikus

December 16, 2007

1 Bevezetés

Használható multigrid módszert először Brandt és függetlenül Hackbusch mutatott be. Megoldották diszkréten az elliptikus parciális differenciál egyenletek N pontból álló rácson lineáris időben.
A multigrid módszerek lényegében ismétlődő eljárások, melyekben a rács nagysága változik minden lépésben, például növekszik vagy csökken a pontok száma egyes dimenziókban egy kettes faktorral.
A következő egyenletet akarjuk megoldani (**E** az elektromos tér, ρ az elektromos töltés x,y,z pontban):

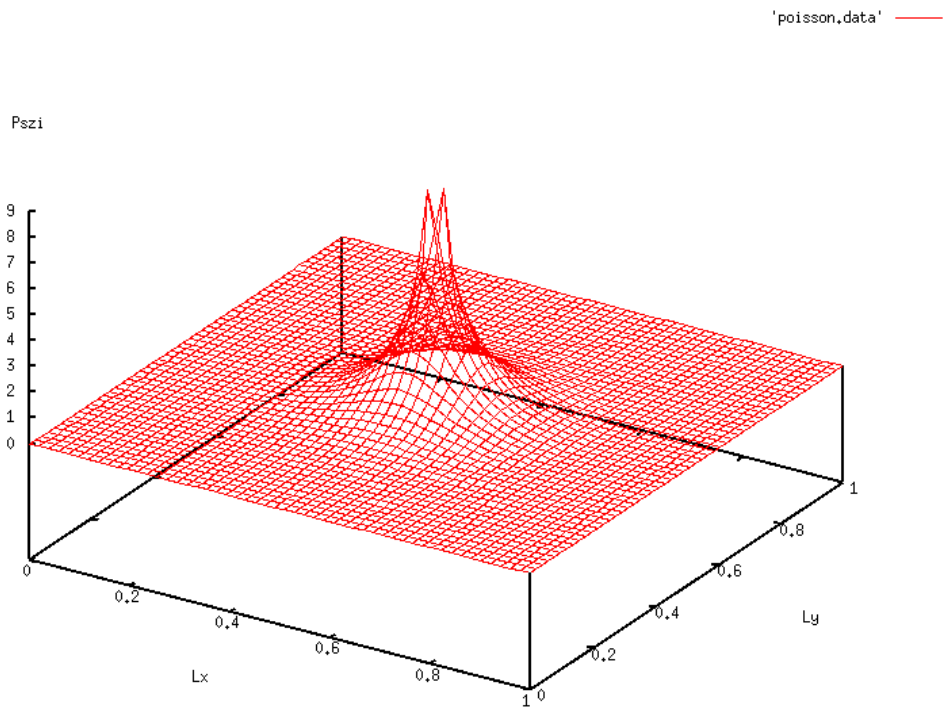

Figure 1: Dipol tere

$$\nabla \cdot E = \frac{\rho(x,y,z)}{\epsilon_0}, \tag{1}$$

ahol ϵ_0 permittivitás állandó. A mi esetünkben csak két dimnezióra (x,y) szorítkozunk.

2 Dipol tere (módosított poisson.cpp)

Az eredeti program csak 1 töltésre írja le a teret, de nekünk dipol kell, ezért berakunk még egy ugyanakkora töltést közvetlen mellé:

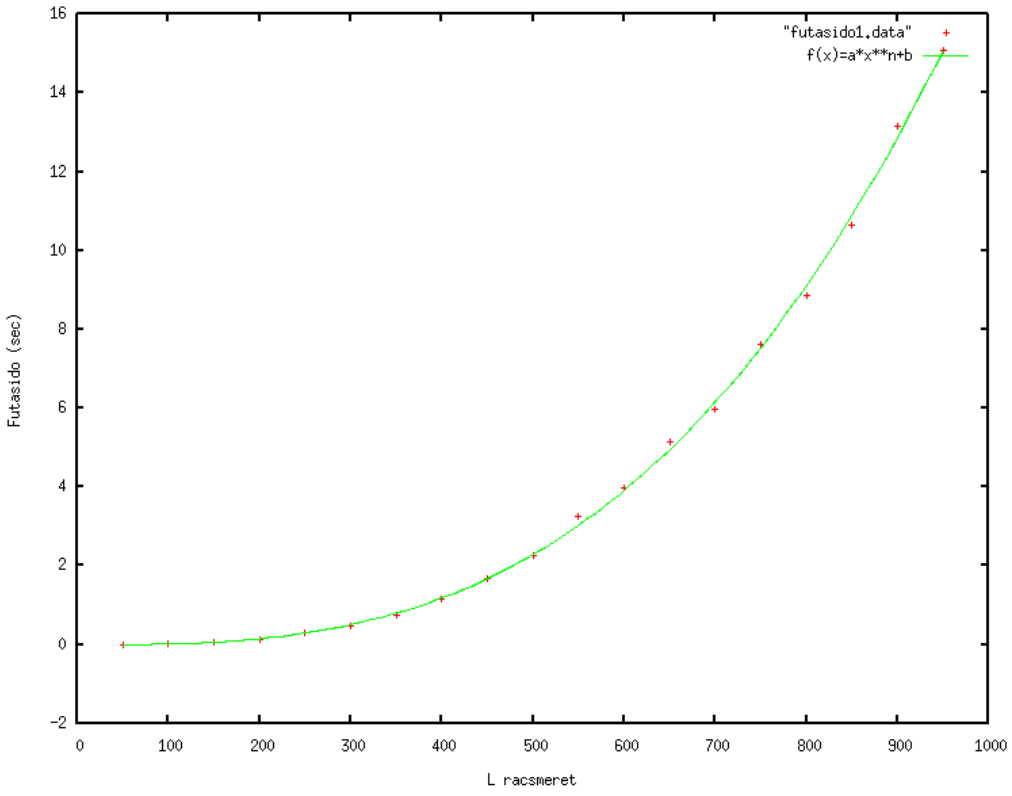

Figure 2: Jacobi futásideje, 0.5 pontossággal

```
rho[i][i+1] = q / (h * h); // 2. charge density
```

Futásidőt úgy mértem, hogy amikor meghívja a kiválasztott algoritmust, ott többször lefuttatom más L értékekkel. Majd a kapott rácsméret-futásidő grafikonra illesztettem az:

$$f(x) = a \cdot x^n + b$$

függvényt. Mindegyik metódust 0.5 pontossággal hívtam meg, és a következőket kaptam.

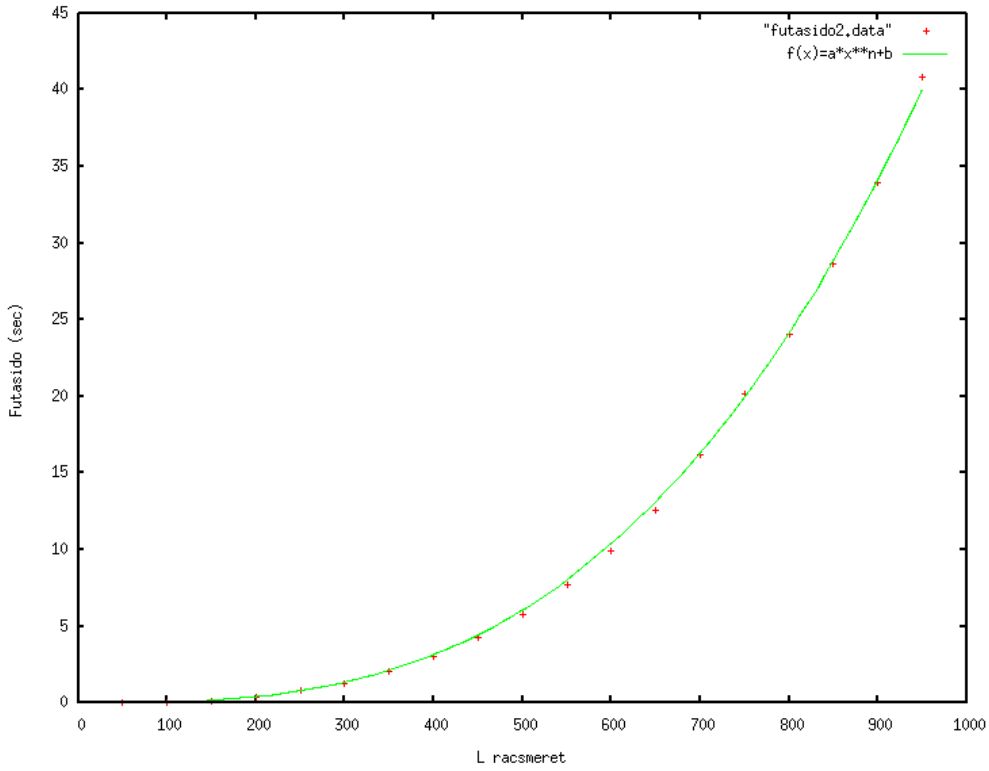

Figure 3: Gauss-Seidel algoritmus futásideje, 0.5 pontossággal

2.1 Jacobi

$$\begin{aligned} a &= 2.71 \pm 33\% \\ n &= 2.93 \pm 1.67\% \\ b &= 0 \end{aligned}$$

2.2 Gauss-Seidel algoritmus

$$\begin{aligned} a &= 7.21 \pm 26\% \\ n &= 2.93 \pm 1.31\% \\ b &= 0 \end{aligned}$$

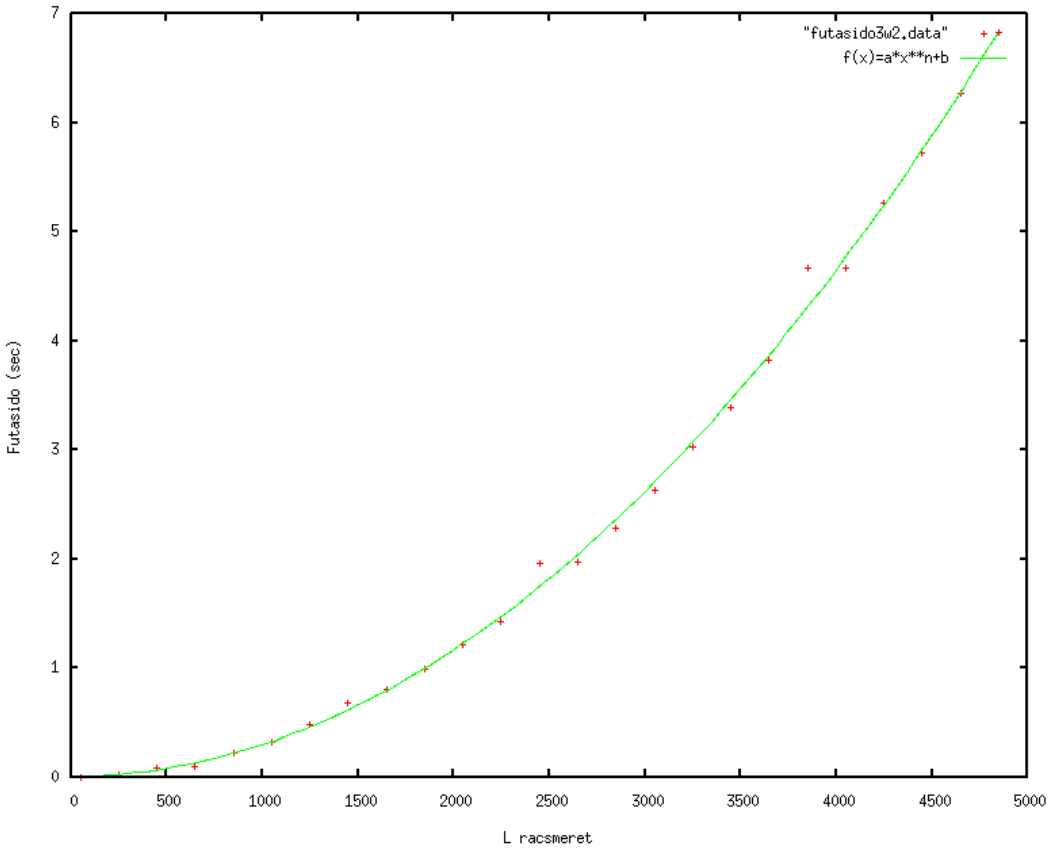

Figure 4: Successive Over Relaxation algoritmus futásideje, 0.5 pontossággal, $\omega = 2$

2.3 Successive Over Relaxation

$$\begin{aligned} a &= 2.9 \pm 39\% \\ n &= 2 \pm 2.3\% \\ b &= 0 \end{aligned}$$

3 Dipol tere (módosított poisson-mg.cpp)

A másik töltés behelyezése a térbe, itt is ugyanúgy történt, mint az előző esetben. Futásidő lemérésére a következő shellscriptet írtam:

```
i=2; //rácsméret
while test $((i=i*2)) -lt 513; do
   echo "L=" $i " "
```

```
        ./poisson-mg $i 54 >> futasido.out  //54 a preSmooth lépések száma
done;
```

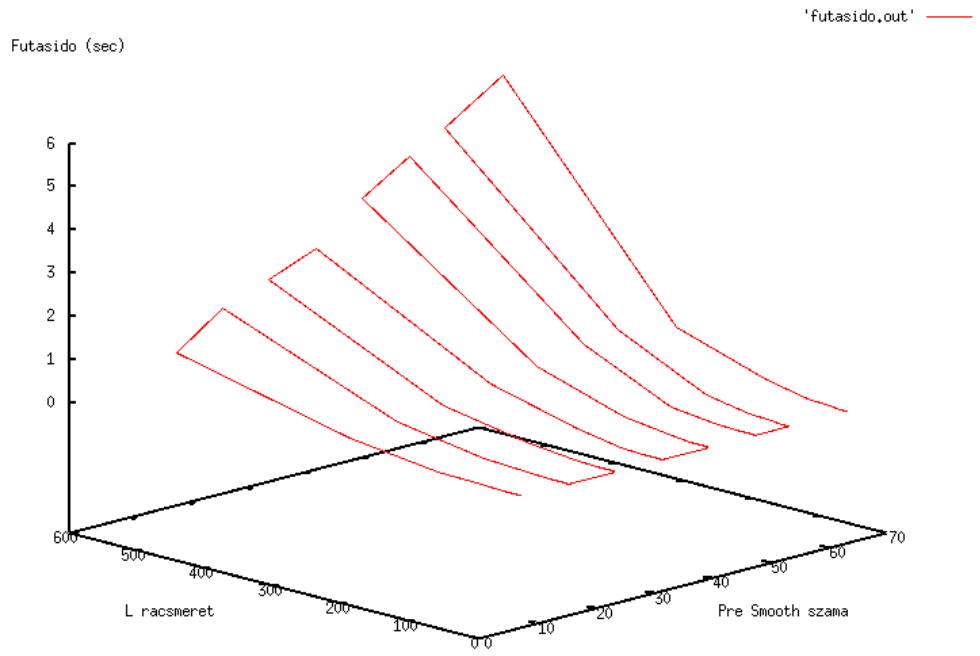

Figure 5: Futásidő, preSmooth és rácsméret függvényében

A kapott eredmény preSmooth függésnél, L=512 esetben lineáris lett, L függésében viszont négyzetes.