Az EM-algoritmus használata hálózattomográfiában

Mérési jegyzőkönyv

Rudolf Ádám, Kozics György

Fizikus MSc.

Mérésvezető: Stéger József

Mérés időpontja: 2012. április 25.

Leadás időpontja: 2012. június 9.

1. Bevezető

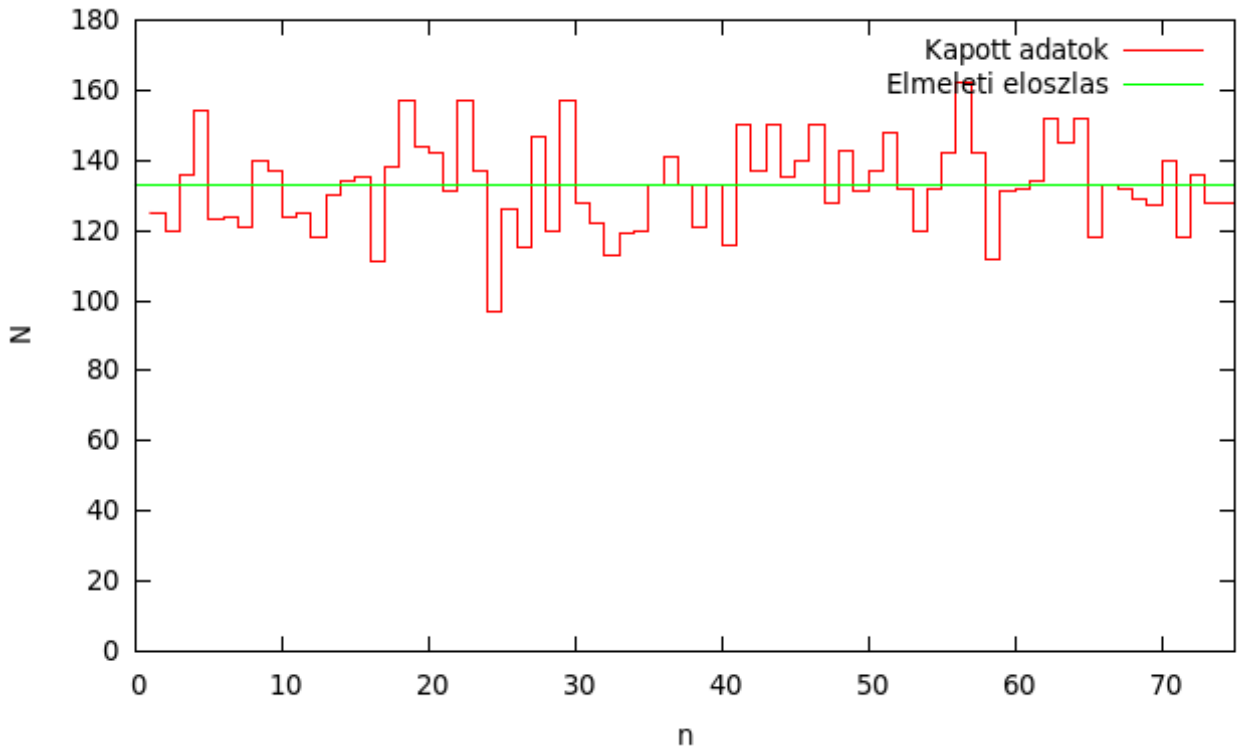
A gyakorlat során a számítógépes hálózatok vizsgálatában használt EM algoritmust valósítjuk meg, és használjuk általunk generált, valamint korábban mért valós adatok vizsgálatára. Az algoritmus célja (esetünkben), hogy egy egyszerű Y elágazás két ágában mért késleltetésekből meghatározzuk a nódusokra jellemző sorbanállási időket. Az elméleti hátteret nem részletezem, az megtalálható a http://complex.elte.hu/komplexlabor/halozatok.html internetes oldalon a felhasznált adatsorral együtt. Csak a közvetlenül felhasznált elméletet fogom közölni.

2. Véletlenszám generátorok, és vizsgálatuk

Első feladatunk a szintetikus adatsor létrehozásához használt véletlenszám generátorok megalkotása, és a valószínűségeloszlások ellenőrzése. Célunk megfelelő paraméterű egyenletes, Poisson és Gauss eloszlású egész számok legenerálása. Ehhez C++ programot írtam, és a GSL függvénykönyvtár mt19937 jelű véletlenszám generátorát használtam kiindulásként. Ebből ki lehet csikarni a kívánt eloszlású számhalmazt, amiből megfelelő transzformációk, és egész számmá alakítás után megkapjuk az általunk kívánt adatokat.

1.1. Egyenletes eloszlás

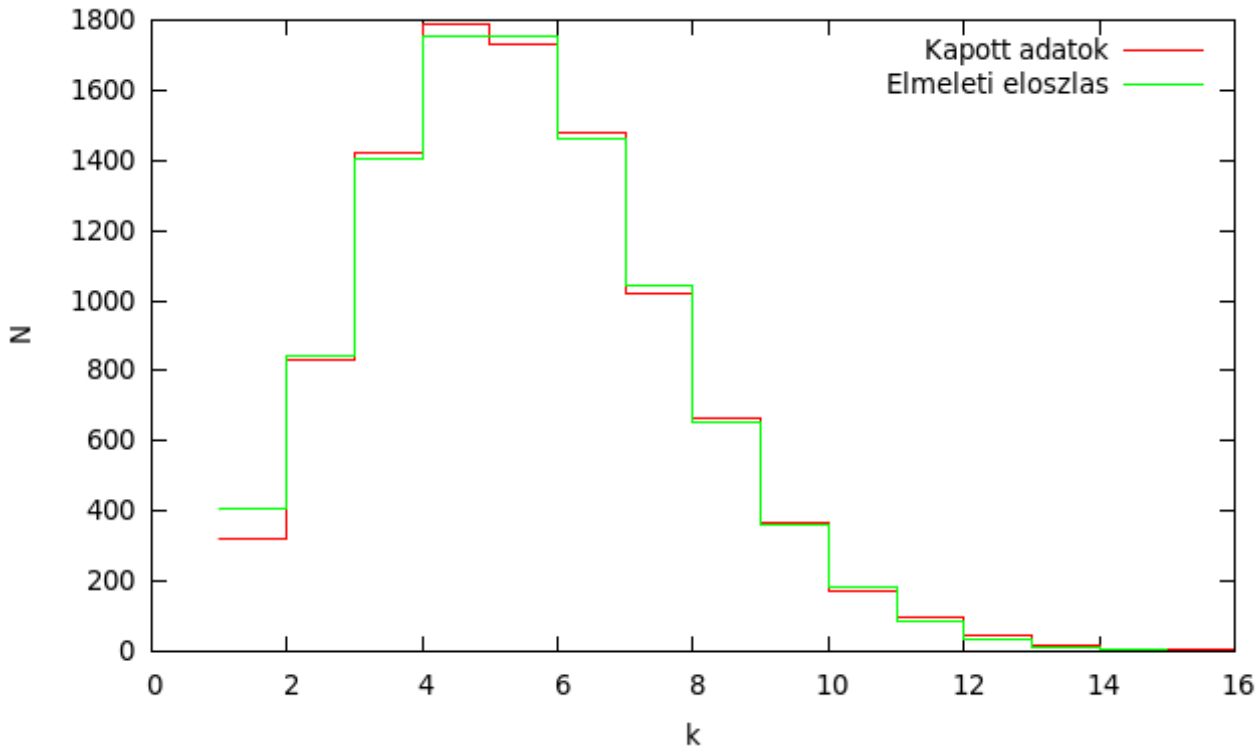
Először egyenletes eloszlású egész számokat generáltam 0 és 75 között. Legeneráltam N = 10 000 értéket, és hisztogramot készítettem belőlük. Itt az elméleti eloszlás egyszerűen $f_{egyenletes}(x) \equiv \frac{N}{75} \approx 133$. A hisztogramot és az elméleti függvényt az *1. ábra* tartalmazza.



1. ábra: 0 és 75 közötti, egyenletes eloszlású egész számok generált és elméleti valószínűségsűrűség függvénye.

1.2. Poisson eloszlás

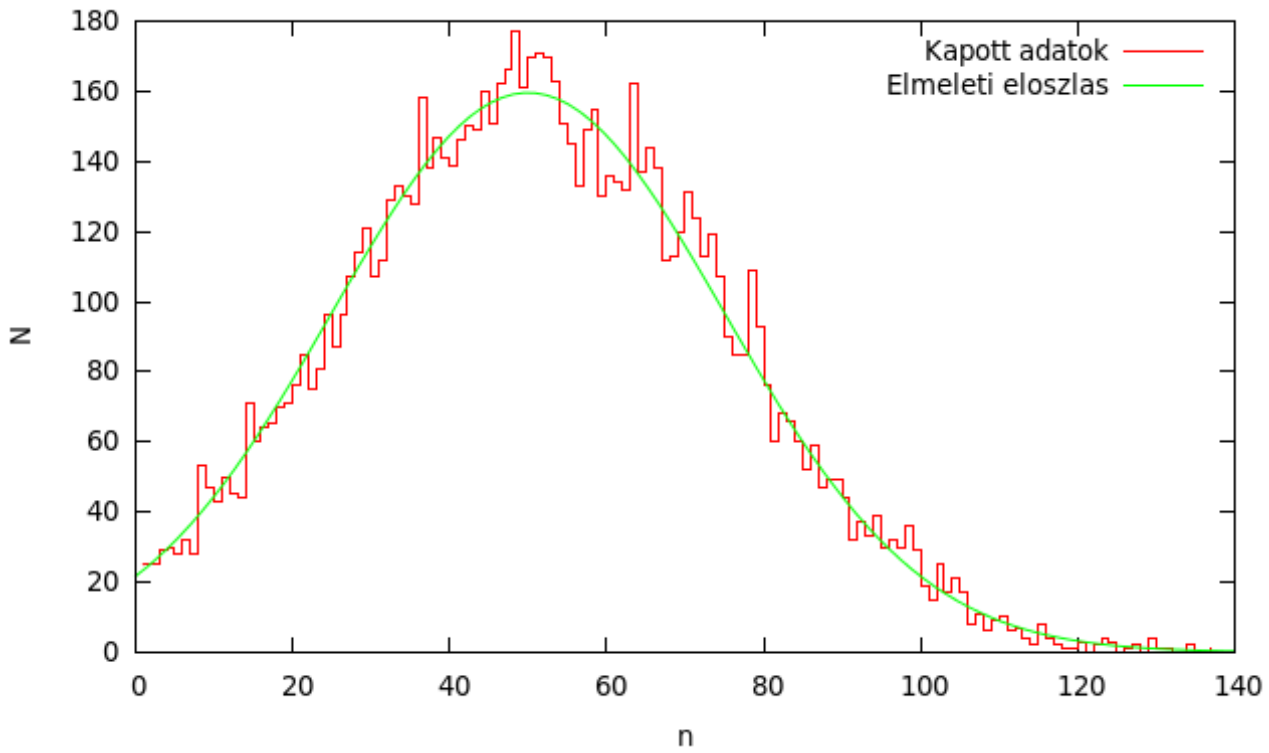
$\mu=5$ paraméterű Poisson eloszlást is generáltam. A hisztogram előzőhöz hasonló összevetése az elméleti függvénnyel a *2. ábrán* látható. Az elméleti eloszlás ez esetben $f_{poisson}(k;\lambda)=\frac{N\cdot\lambda^k e^{-\lambda}}{k!}$



2. ábra: μ = 5 paraméterű, Poisson eloszlású egész számok generált és elméleti valószínűségsűrűség függvénye.

1.3. Gauss eloszlás

Végül egy μ = 50 középértékű, σ = 25 szórású Gauss eloszlást generáltam. Az elméleti eloszlás $f_{Gauss}(x) = N \cdot \frac{1}{\sigma\sqrt{2\pi}} \cdot \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$. Ezt, és a kapott adatok hisztogramját a *3. ábra* tartalmazza.



3. ábra: μ = 50 középértékű, σ = 25 szórású Gauss eloszlású egész számok generált és elméleti valószínűségsűrűség függvénye.

Látszik, hogy mindegyik eloszlás jól visszaadja az elméleti értékeket.

3. EM algoritmus implementációja

Az EM algoritmus lényege, hogy meghatározzuk az adott szakaszokon elszenvedett késleltetéseket. 3 szakaszból 2 útvonal áll össze, és mi az útvonalakon elszenvedett késleltetéseket ismerjük. A szakaszokat v = {1, 2, 3}, az útvonalakat λ = {2, 3} indexeli. Az időt kvantáljuk, így értéke pozitív egész számokat vehet fel, amiket k-val jelölünk. Az idő kvantálása után $P_{\nu,k}$ annak a valószínűsége, hogy a v szakaszon a sorbanállás ideje a k-adik binbe esett. (Ezen túl a kvantált adatokat fogom időnek nevezni az egyszerűség kedvéért.) Ezeket az eloszlásokat adottnak vesszük. A 2-es útvonal az 1-es és 2-es szakaszokból, a 3-as útvonal az 1-es és 3-as szakaszokból áll össze.

$P_{\nu,k}$ -k segítségével így meg tudjuk határozni egy adott mérés valószínűségét. Tegyük fel, hogy az (a, b) időket mérjük a két útvonalon. Ez azt jelenti, hogy az adott eloszlású változók összegének ezeknek a számoknak kell lennie, vagyis ha az 1-es szakaszon a késleltetés k, a másodikon $a - k$-nak, a harmadikon $b - k$-nak kell lennie, ennek a valószínűsége pedig:

$$\Pi_{a,b} = \sum_k P_{1,k} P_{2,a-k} P_{3,b-k} \quad ,$$

ahol az összes index nemnegatív. Elméleti megfontolások alapján felírhatók a következő egyenletek:

$$P_{\nu,k} = \frac{1}{N} \sum_{i=1}^{N} P\left(x_\nu = k \,|\, (y_2^i, y_3^i); \Theta\right) \quad ,$$

ahol

$$P\left(x_1 = k \,|\, (a,b); \Theta\right) = \frac{P_{1,k} P_{2,a-k} P_{3,b-k}}{\Pi_{a,b}}$$

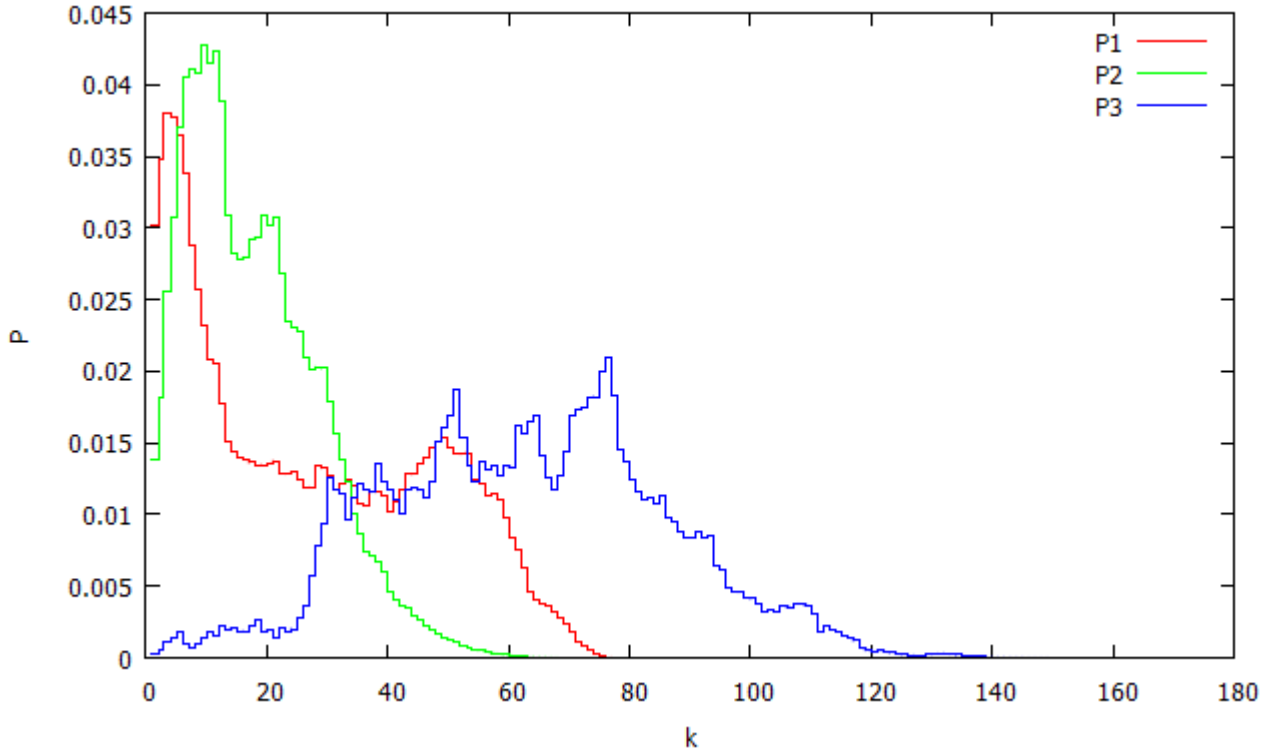
$$P\left(x_2 = k \,|\, (a,b); \Theta\right) = \frac{P_{1,a-k} P_{2,k} P_{3,b-a+k}}{\Pi_{a,b}}$$

$$P\left(x_3 = k \,|\, (a,b); \Theta\right) = \frac{P_{1,b-k} P_{2,a-b+k} P_{3,k}}{\Pi_{a,b}} \quad ,$$

ahol Θ a háttérben meghúzódó valószínűségsűrűségek halmaza. Ezek alapján MatLab programot írtam, ami a következőket csinálta: az adatok megfelelő kvantálása után (ha ez szükséges) beállítottam a feltételezett valószínűségeloszlásokat. Ezek egyenletes eloszlással indultak. Megkereste az értelmezési tartomány maximumát (B), majd elindult az iteráció. Ennek során kiszámolta $\Pi_{a,b}$ mátrixot (feltételekkel megadva azokat a k-kat, amire összegezni kell). Ha megvolt, a többi képlet segítségével léptette P-ket. Ezeknél is oda kellett figyelni az indexelés helyességére: csak pozitív értékeken futhattak végig. Itt azt is figyelembe kell vennünk, hogy $\Pi_{a,b}$ sem lehet 0. Az iterációs ciklus végén az összes P érték változását vizsgáltam. Ha ez egy előre megadott küszöbérték alatt volt, az iteráció leállt. Megjegyzendő, hogy a program egy konvergáló szakasz után numerikus hibák miatt elszállt, vagyis az ε küszöbérték beállítása kritikus volt.

4. Szintetikus adatsor kiértékelése

Elsőnek 1000 mintából álló szintetikus adatsort hoztam létre a fent leírt véletlenszám generátorokkal. A rejtett paramétereket hoztam létre, mégpedig úgy, hogy az 1-es szakasz eloszlása egyenletes, a 2-esé Poisson, a 3-asé Gauss legyen. Ezután megfelelően összeadva őket létrehoztam az y^2 és y^3 adatpárokat. Ezeket kiértékeltem az általam írt programmal, a kapott eloszlások a *4. ábrán* látszanak.



4. ábra: Az EM algoritmust megvalósító program szintetikus adatsorral végzett tesztje. Elég nagy hibával ugyan, de felismerhetők az eredeti eloszlások.

A kezdeti eloszlásokból ezred nagyságrendű ε küszöbérték mellett az adatsorok formája ugyan nem követi a pontos értékeket, de látszik, hogy megközelítették az elméleti értékeket. Az egyenletes eloszlásnak ugyan van egy csúcsa 0 közelében, de aztán közel egyenletesen halad tovább és 70 körül levág 0-ra. A Poisson eloszlás is megfelelően 0-hoz közel húzódik, a Gauss alakunk pedig egy durván szimmetrikus, 50-60-ra centrált csúcsot hozott létre. Ezt sikernek könyvelem el, és viszonylag nagy hibával ugyan, de a kapott adatokat hitelesnek értékelem.

5. Valódi mérés kiértékelése

Megtettük a megfelelő előkészületeket, hogy egy valódi adatsoron használjuk a programunkat. A megadott idősor mintából számolt adatokban a megfelelő fenntartások mellet megbízhatunk.

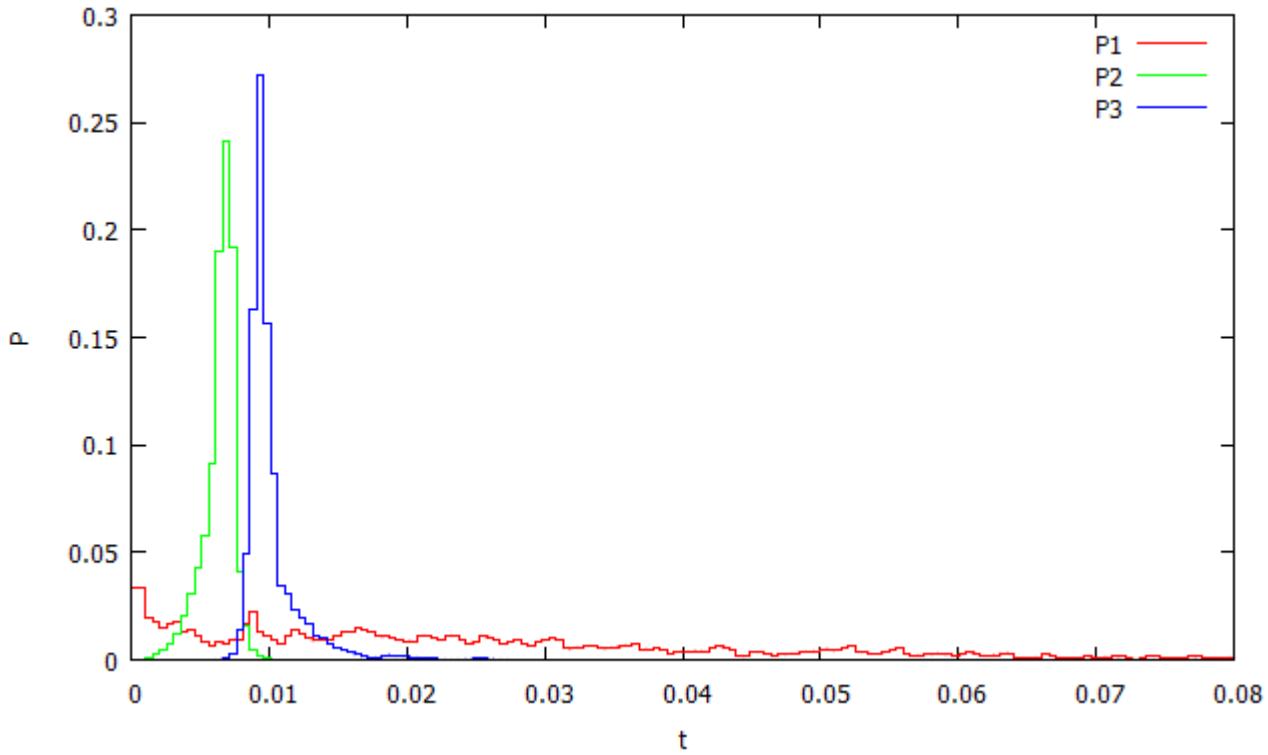
5.1. Kvantálás

Az általunk kapott minta még (adott pontossággal) időben van megadva, így első feladatunk, hogy a kiértékelő program számára feldolgozható kvantált adatsorrá alakítsuk. Természetesen a kvantálással információt vesztünk, ezért a binek méretét úgy kell meghatározni, hogy ne legyen túlzottan elaprózott, mert az lassítja a program futását és megszaporítja az adatokat, viszont ne is legyenek túl nagyok, mert akkor sok információt vesztünk és pontatlan értékeket kapunk.

A kvantálást a következőképpen végeztem. Először megkerestem az adatsor maximumát (bármely oszlopban előforduló adatok között). 0 és a maximum közötti részt kell egyenlő binekre felosztani. 200 részre osztottam fel, két okból. Az előző mérésben 170 volt a maximum ugyanekkora mennyiségű adatpárnál. A másik ok, hogy a maximum 6 értékes jegy pontossággal van megadva, a többi adat tipikusan 5-re, így a kvantálással okozott hiba % nagyságrendű lesz.

A kvantálás elvégzésére is MatLab programot írtam. Bevezettem a $q = 200/max$ számot, végigszoroztam vele az összes adatot és vettem az egészrészüket, így 0 és 200 közötti egész számokat kaptam. Ezt az adatsort már át lehet adni a kiértékelőprogramnak.

5.2. Eloszlások meghatározása



5. ábra: Mért adatsorból EM algoritmussal kapott valószínűségeloszlás a késleltetésekre.

A kapott adatsort ezután átadtam a programnak. Az eredmények az *5. ábrán* láthatóak. A k értékeket q-val leosztva visszakalibráltam időre, hogy szemléletesebb legyen az eredmény. Látszik, hogy az első szakaszon a késleltetés kicsi, lassan lecsengő eloszlást mutat. Itt a csomagok még nem találkoztak útvonalválasztóval. A kettes és hármas úton egy-egy éles csúcsot látunk, amiből arra következtetek, hogy ez főleg a propagációs idő és routerre jellemző kiszolgálási idő miatt van, és kevésbé a sorbanállás miatt (akkor nagyobb lenne a szórás). A két csúcs távolsága viszont abból kell, hogy fakadjon, hogy a hármas úton közlekedő csomagnak meg kell várnia, míg az útvonalválasztó kiszolgálja a kettes úton haladót. Ebből azt is meg lehet határozni, hogy a kiszolgálási idő kb. 0,0025 s (feltéve, hogy az adatok másodpercben voltak megadva).