Rendszerintegráció és -felügyelet laboratórium (VIMIM309)

Felügyeleti adatok vizuális elemzése

Mérési segédlet

Készítette: Szombath István

Utolsó módosítás: 2012. április 17.

Verzió: 1.0

Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

1 Bevezető

A mérés célja, hogy a hallgatókkal megismertesse a vizuális adatelemzés előnyeivel és fontosságával. A vizuális adatelemzés egy összetett folyamat, mely számos más szakterült specifikus tudását használja, például elsősorban az adatbányászatból és az adatvizualizációból merít.

Az adatelemzéshez valós infrastruktúrából (egy virtuális gépen futó konferencia szerverről) származó felügyeleti adatokat fogunk felhasználni. A mérés célja a kiadott adathalmaz „megértése", alapvető szakterület-specifikus ökölszabályok és trivialitások megtalálása, visszafejtése, majd összetett függőségek keresése a statisztikában és adatvizualizációban használt RStudio és Mondrian programok segítségével. Ilyen összetett függőség lehet például a felhasználók által érzékelhető szolgáltatási szint minőség (QoS) és az alacsony szintű fizikai erőforrások kihasználtságának kapcsolata. A segédlet betekintést enged a vizuális adatelemzéshez milyen elengedhetetlen ismeretek szükségesek illetve milyen eszközöket, módszereket célszerű használni. A laborgyakorlat előtt a hallgatóknak egy ún. beugró formájában kell bizonyítania, hogy felkészültek a laborgyakorlatra. A felkészülés a segédletből történik, a számonkérés anyagát képezik az alapvető fogalmak és a leírt tények és felsorolások (tehát a konkrét kód szintaxis és a hivatkozott irodalom nem).

A segédlet először betekintést enged három egymástól részben független informatikai ágazatba, melyek minimális ismerete szükséges a felügyeleti adatok vizuális elemzésének megértéséhez. Sorra vesszük az adatbányászat főbb lépéseit, majd bemutatjuk a mérésen használt főbb adatvizualizációs technikákat, utána pedig egy rövid leírás található a felügyeleti adatok sajátosságairól.

Ezek után kerül sor az adatok vizuális elemzésének rövid bemutatására. A következő pont a felügyeleti adatok vizuális elemzése, ami a vizuális adatelemzés azon speciális esete, mikor a feldolgozandó adathalmaz forrása az IT infrastruktúra naplózása és monitorozása során keletkezett.

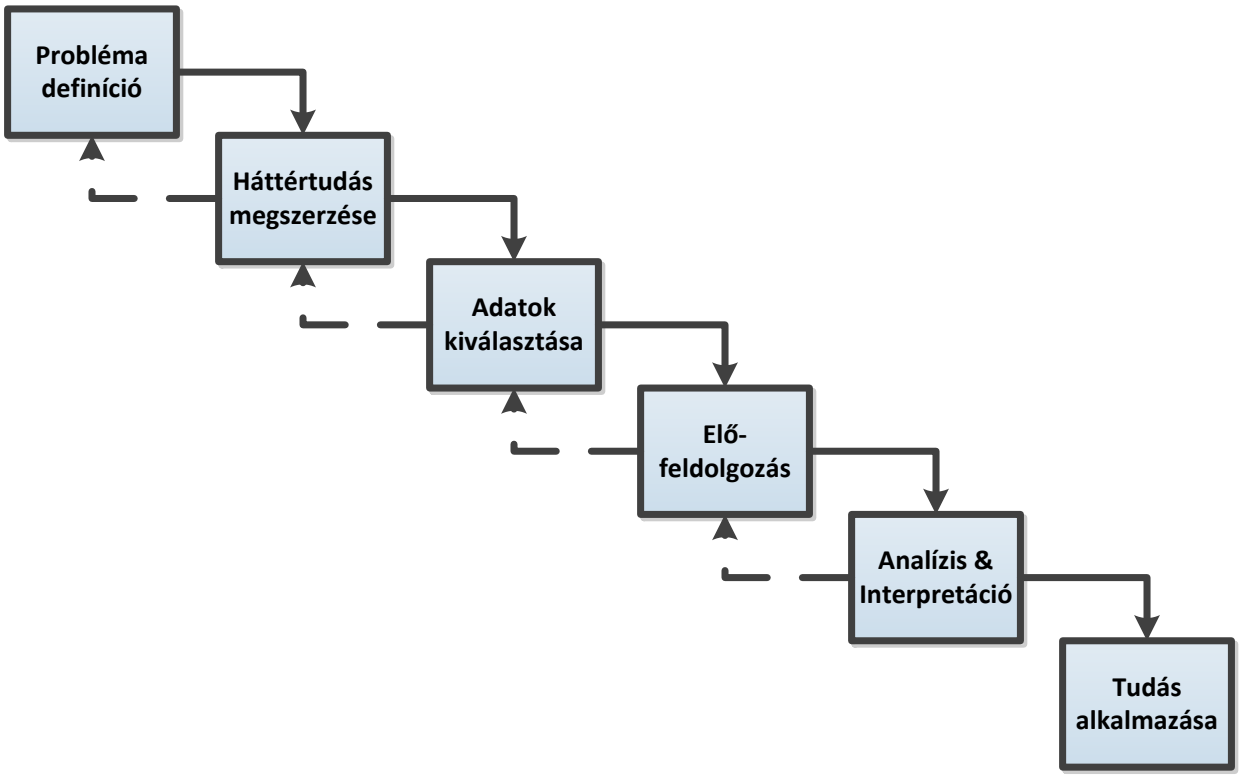
Továbbiakban megismerkedhetünk az IT infrastruktúra architektúrával, ahonnan az adatok származnak. Ez a tudás nélkülözhetetlen, ha az adatokat meg akarjuk érteni. Majd az R és a Mondrian nevű eszköz kerül bemutatásra, melyek segítségével lehetőség nyílik az adatfeldolgozásra, transzformálásra, és megjelenítésre.

Végül alapvető statisztikai ismeretek kerülnek felfrissítésre. A függelék és a hivatkozásjegyzés a mérés során felmerülő problémákban nyújthat majd segítséget.

2 Alapfogalmak

2.1 Adatbányászat

Az adatbányászat eljárások és módszerek összessége, mely alkalmas nagy adathalmazokból hasznos és értelmes mintákat felderíteni, melyek korábban explicit nem voltak ismertek. Az adatbányászat iteratív folyamat, továbbá érdemes megfigyelni, hogy az első két lépés a probléma definíció, és a háttértudás megszerzése. A vizuális adatelemzés éppen azokban az esetekben tud hatékony lenni, mikor a hagyományos adatbányászati eszközök nem, például, mert a szükséges háttértudás nem ismert, vagy nem ismert pontosan a cél.



ábra 1 Az adatbányászat lépései

http://en.wikipedia.org/wiki/Data_mining

2.2 Adatvizualizáció

Az adatvizualizáció adatok vizuális reprezentálása, emberek számára nyújt gyors és hatékony kommunikációs lehetőséget grafikus módon.

http://en.wikipedia.org/wiki/Data_visualization

2.3 Felügyeleti adatok

A felügyeleti adatok általában nagy és heterogén IT infrastruktúra környezetből származnak. Az IT infrastruktúrák logikailag több rétegre különíthetőek el. A rétegek megválasztása részben önkényes részben függhet egy adott szakterület sajátosságaitól. Egy lehetséges elrendezés: fizikai réteg (hálózati csatoló, switchek), ip réteg (routerek, gatewayek, számítógépek), és logikai réteg (alkalmazások). Mindegyik réteg számos elemet és elemek közötti kapcsolatot tartalmazhat, mindegyik elem számos attribútumot tartalmazhat, melyek állapota részben megfigyelhető.

Példák:

- a virtualizációs környezet paraméterei (nem mindig megfigyelhető, több száz paraméter)
  - Hoszt CPU kihasználtsága
  - Hoszt memória kihasználtsága
  - Hoszt hálózati csatolójának kihasználtsága
  - …
- PC / virtuális gép (akár több ezer paraméter)
  - Guest CPU kihasználtsága
  - Guest memória kihasználtsága
  - Guest hálózati csatolójának kihasználtsága
  - …
- JVM paraméterek (többszáz/JVM)

    - Heap Size
    - Stack Size
    - …
- Alkalmazás paraméterek (pl. WebSphere, …)
    - Bejelentkezett felhasználók száma
    - …

Általában elmondható, hogy a felügyeleti adatok számos objektum számos paraméterét mérik, általában hosszú időn keresztül, akár nagy felbontással is. Az adatok továbbá nagyon zajosak, főleg a valódi IT infrastruktúrából származóak. Elmondható, hogy adatszűrés nagyon fontos részfeladat és az erőforrások nagy részét ez emészti fel, azonban mi a mérésen már részben tisztított (azonban zajos, esetenként redundáns) adatokkal fokunk dolgozni, azaz ezt a lépést nem fogjuk elvégezni.

3 Adatok vizuális elemzése

Az adatok vizuális elemzése az adatbányászat és az adatvizualizáció (és esetenként más szakterületek, melyeket itt nem tárgyalunk) összefonódásából jött létre. Képes nagy adathalmazok hatékony megjelenítésére és elemzésére. A feldolgozást és a megjelenítést számítógépek végzik, azonban az adatbányászat számos ágával, itt az interaktív elemző az ember. Tehát az ember szerves része az elemzés folyamatának, ugyanis az emberi agy bizonyos problémákat intuitív módon gyorsabban és hatékonyabban tud megoldani.

Fontos kitétel, hogy a vizuális adatelemzés ne igényeljen mély és komplex matematikai, statisztikai és algoritmikus ismereteket. Inhomogén és zajos adatok elemzésére is kiválóan alkalmas módszer.

Számos szakterületen használják, különösen alkalmas akkor, ha keveset tudunk az adatról, az adott szakterület sajátosságairól, vagy ha a cél nem teljesen világos. Ezt nevezik ún. EDA-nak (Exploratory data analysis), ahol az adathalmazok vizuális elemzése kezdeti hipotézis előállítása céljából történik.

http://en.wikipedia.org/wiki/Exploratory_data_analysis

3.1 Az EDA (Exploratory data analysis) lépései

A következő lépéseket célszerű végigjárni, amennyiben vizuális adatelemzés a feladatunk:

- adatok áttekintése
    - érdekes minták azonosítsa, szükség szerinti fókuszálás, érdekes részhalmazok azonosítása
- szűkítés és szűrés
- részletekre koncentrálás, szükség szerinti mélységben

Példa egy felügyeleti adatsor vizuális elemzéséről, ahol az adatelemző még az oszlopneveit és jelentésüket se tudta kezdetben: először a redundáns oszlopok kiszűrése következett (CPU terhelést több ágens is mért), majd az egyedi azonosító (időbélyeg) került megállapításra. Gyorsan kiderült az is, hogy pár attribútum között lineáris összefüggés van (CPU terheltség, alkalmazás áteresztőképesség), legalábbis normális működés tartományokban. Így a kezdeti százról húszra sikerült az adatok dimenzionalitását csökkenteni. Ez után megadtuk az elemzőnek az oszlopok jelentéseit, majd a memóriára fókuszálva megállapítható volt, hogy az alkalmazás lefagyott, az erőforrásokat fogyasztotta és nem engedte el, azonban értelmes kiszolgálást nem végzett. Egy belső timer vagy külső reset hatására azonban újra visszaállt a megszokott működési tartományba.

3.2 Felügyeleti adatok vizuális elemzése

A felügyeleti adatok tipikusan nagy dimenziószámmal rendelkeznek (a metrikák számából következően), továbbá időben nagyon finom felbontású és hosszú időtartamú is mérés eredményét is

tárolhatják, így a konkrét adatpontok száma is nagy. Egy-egy ágens egy attribútum halmazt mér, és azokat továbbítja a központi naplózó vagy adatfeldolgozó felé. Például az ESXi szerver méri a gazdagép és az egyes hosztgépek metrikáit, és ezeket képes adott felbontással és időközönként naplóállományba menteni. Tipikusan minden ágens az attribútum halmaz értékeivel elküldi a saját azonosítóját és az időbélyeget, mely az attribútumok rögzítésének időpontja (tehát ha konzisztens központosított adatfeldolgozást szeretnénk, akkor az ágensek óráit szinkronizálni kell!).

Az interaktív adatelemzés megjelenítés többnyire monitoron történik. 2D-s felületen sok (3-nál jóval több) dimenzionalitású adatot megjeleníteni problémás lehet, főleg ha az elemző ember (például sok 2D-s ábra minden dimenziópárra nem biztos, hogy előnyös, bár néha elkerülhetetlen).

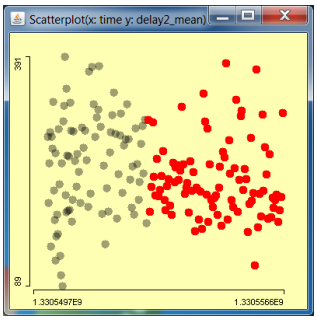
A felügyeleti adatok akár több ezer dimenzióját azonban a hagyományos módokon már nem lehet megjeleníteni értelmezhető formában. Célszerű lehet ilyenkor a nem releváns és / vagy redundáns attribútumok szűrése (például az CPU L2 cache állapota nem feltétlenül érdekes, ha nem a CPU a szűk keresztmetszet). Redundáns attribútum esetén előfordulhat, hogy egy attribútumot többször mérünk, de az is, hogy egy attribútum halmaz lineárisan nem független. Korrelációszámítással gyorsan kiszűrhetőek a plusz információval nem rendelkező attribútum párok.

Gyakran szükség lehet gráfok, azaz objektumok és azok kapcsolatainak a megjelenítésére. Itt különösen nagy szerepe lehet a hierarchiába rendezésnek, mert az ember nem tud átlátni egy több ezer csomópontból és élből álló ábrát.

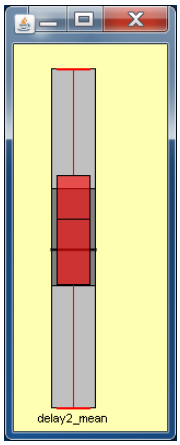
3.3 Adatvizualizációs technikák adatok vizuális elemzés esetén

Általános recept nem adható hogyan kell vizuális analízist végezni, azonban van pár bevett gyakorlat, ami jó kiindulási alap lehet:

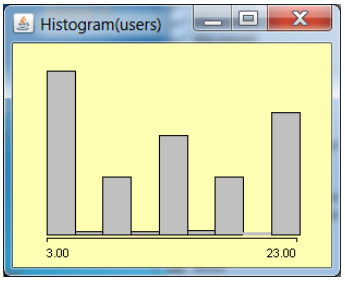
- attribútumok x-y(-z) tengely szerinti ábrázolása (scatterplot)



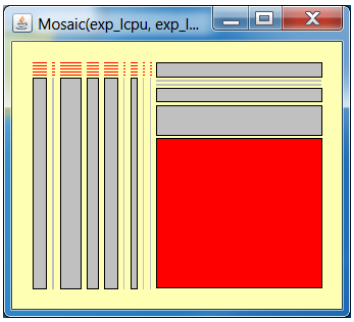
- oszlopok szórásának, várható értékének vizsgálata (boxplot). Adott populáció eloszlását mutatja. Alsó 25 % és felső 25 %, átlag, minimum és maximum érték, esetleges kívülálló értékek.



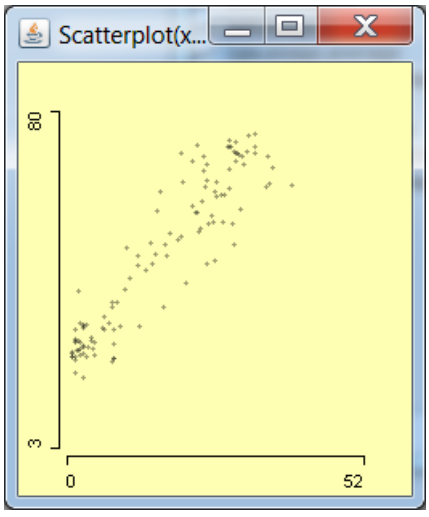
- Hisztogram, mikor volt valódi szolgáltatás, mikor nem, mikor mennyi felhasználó volt bejelentkezve (histogram).



- működési tartományok azonosítása (mosaic plot) (az x tengelyen a memória mennyisége y tengelyen a CPU mennyisége figyelhető meg, a téglalap nagysága pedig arányos az adott működési tartományban lévő egyedek számával)



- A korreláció vizsgálatával a fölösleges oszlopok (lineárisan összefüggő oszlopok) hagyhatóak el, vagy akár nem várt összefüggésekre is fény derülhet (például két ágens CPU metrikája ugyanarról a gépről valószínűleg redundáns, azonban a virtuális host és guest CPU órái nem feltétlenül lesznek korreláltak, gondoljunk utána!)!



ábra 2 Példa a gyenge korrelációra

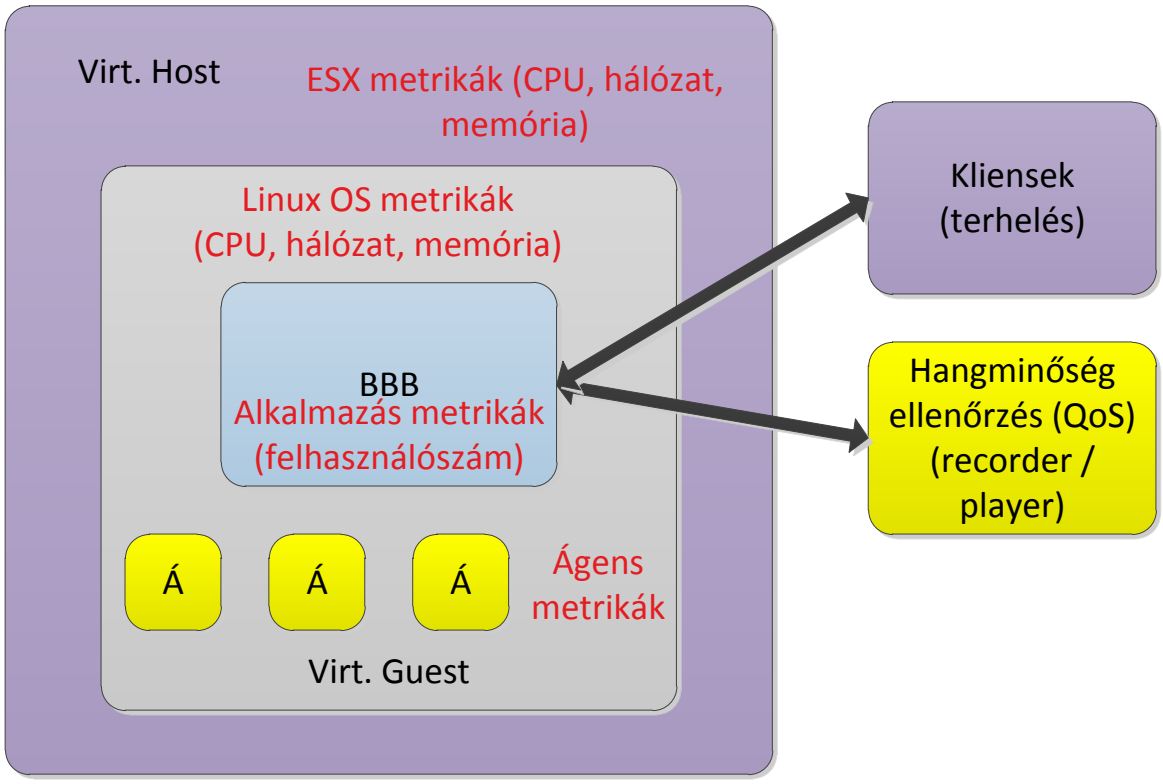
Nagyon fontos alapkövetelmény a vizuális analízistől, hogy legyen interaktív. Bizonyos alkalmazott technikák, melyek segítik az interaktivitást:

- dinamikus projekció (több 2D-s ábra ugyanarról az adathalmazról)
- interaktív zoom (magas szinten részletek elrejtése, ráközelítve pedig a megjelenítése)
- interaktív torzítás (fontos részletek magas felbontásban, nem fontosak alacsonyban, pl mikor egy tree editorban böngészünk alapból minden csomópont kibontatlan, azonban tetszés szerinti mélységben kibonthatunk csomópontokat)
- interaktív színezés (pl. kijelölés)
- összekapcsolt ábrák (linked plot), mikor pl. kijelölünk egy részhalmazt az egyik ábrán, és a többi ábrán is megjelenik a kijelölésünk

4 A mérési környezet bemutatása

Adott egy konferencia szerver (BigBlueButton), mely alkalmas konferencia beszélgetés lebonyolítására, azaz képes többek között kép és hanganyag átvitelére a kliensek között. Kérdés, hogy mi befolyásolja a hangminőséget (QoS). Azaz mely erőforrás változására érzékeny a QoS? (például a latency mely alacsony szintű erőforrásra érzékeny)

Az alábbi ábrán látható a rendszer vázlatos architektúrája. Egy virtuális gépeket futtató gazdagép (virtual host) futtat egy Linux alapú virtuális gépet (guest). A virtuális gépre telepítettük a BBB szervert, továbbá egyéb ágenseket, mely a virtuális gép állapotát képesek figyelni és jelenteni, naplózni. A szervert webes felületen keresztül érik el a kliensek, a terhelést több kliens együttes bejelentkezésével végezzük. A hangminőség mérésére egy dedikált lejátszó (player) és egy felvevő (recorder) kliens szolgál. A player lejátszik egy hangmintát, amit elküld a szervernek, a szerver pedig szétküldi a konferencia részvevőinek, azaz az összes bejelentkezett kliensnek. A recorder ezt a hangmintát veszi fel és analizálja. A bemenő és kijövő hangmintát feldolgozva különböző (hang) minőségi metrikákat tudunk számolni, mint például késleltetés (latency), késleltetés szórása (jitter), stb.



ábra 3 A mérési környezet architektúrális vázlata

Fontos látni, hogy egyszerre mérünk virtualizációs platform szintű metrikákat, (virtuális) operációs rendszer szintű metrikákat, alkalmazás szintű metrikákat, egyéb metrikákat, valamint QoS metrikákat. A kérdés kicsit másképp megfogalmazva az, hogy az alacsony szintű erőforrások változása milyen kapcsolatban van a magas szintű metrikákkal, amit a felhasználó is érzékel.

Például megállapítható, hogy egy alkalmazás memória, hálózat, vagy processzor erőforrásra érzékeny, adott konfigurációval körülbelül hány kiszolgálót tud kiszolgálni. De az is felismerhető, ha beragadt egy folyamat, például mert sok erőforrást fogyaszt, de magas szinten nem látni aktivitást, azaz kérések nem jönnek, és nem távoznak a rendszerből a felhasználók fele.

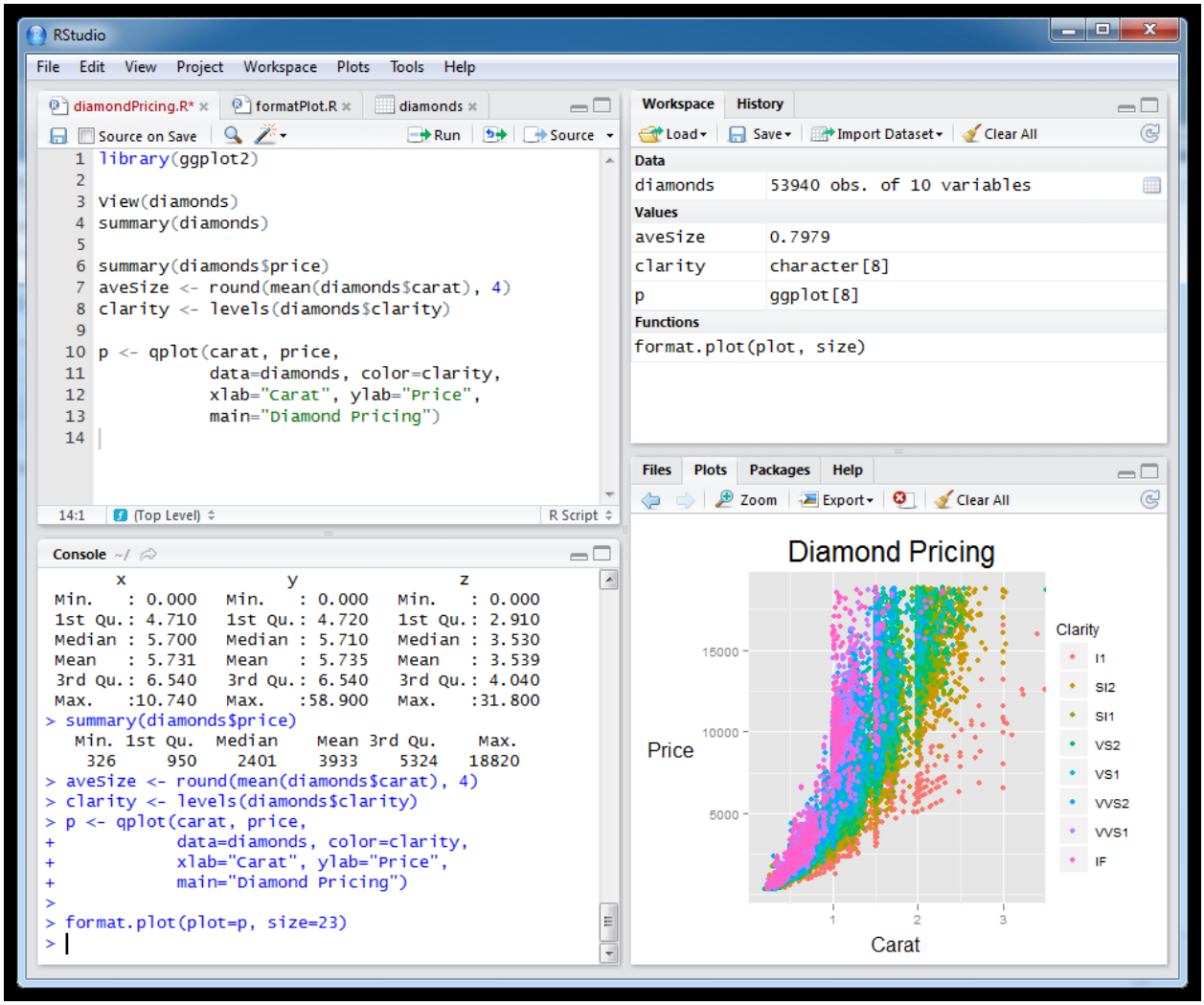
4.1 Adatvizualizációs és statisztikai eszközök

4.1.1 R, RStudio

Az R egy (GNU, azaz nyílt forráskódú) szoftvercsomag, mely adatok analízisére és grafikai megjelenítésére lett kifejlesztve. Az R nagyon sokféle statisztikai és grafikai megjelenítő technikát tartalmaz, és jól / könnyen bővíthető. Az R tulajdon képen egy (programozói) környezet, vagy egy parancssori értelmező, amiben statisztikai technikákat implementáltak. Az R kiadása számos csomagot alapból tartalmaz, de az Internetről is letölthetőek.

Az R-t számos tudományos területeket előszeretettel használják, ugyanis rendkívül könnyen lehet vele publikáció szintű grafikonokat és ábrákat előállítani. Ennek oka például, hogy a grafikonok alapbeállítására nagy figyelem fordul (de természetesen minden testre szabható a kellő tudás birtokában), és LaTex szerű dokumentum formátum kimenete is van.

Az R képes a parancssori értelmezőn begépelt parancsokra kimenetet produkálni, vagy batch jelleggel parancssori állományt feldolgozni. A könnyebb kezelhetőség érdekében jött létre az RStudio. Ez egy ingyenes nyílt forráskódú fejlesztői környezet (IDE) az R-hez. Számos szolgáltatást nyújt, ami kényelmessé teszi az R használatát. Segítségével például egyszerre láthatjuk az adatainkat (változókat), a batch állományunkat és a parancssori értelmezőt és kimenetét, valamint grafikonokat. Segít továbbá a szintaxis kiemelésben, tördelésben.



ábra 4 Az RStudio kezelői felülete

A mérés során a hallgatóknak csak a mérési környezet adataival kell dolgozniuk, egy csv állománnyal, azaz feltételezzük, hogy az adatok már előálltak. Így a mérés nagy részét az RStudio fejlesztői környezettel való ismerkedés fogja kitenni.

Az R-hez számos segédlet található az Interneten, pár fontosabb hivatkozás megtalálható a segédletben, továbbá pár alapvető művelet megtalálható a függelékben.

http://www.r-project.org/
http://rstudio.org/

4.1.2 Mondrian

Általános célú statisztikai adatvizualizációs eszköz. Képes az R által elmentett adatokat beolvasni és azokat megjeleníteni. Kezeléséhez nincs szükség statisztikai ismeretre, vagy az R ismeretére, emiatt prezentációs célokra kész adatok esetén igen jól alkalmas (azonban adatok tisztítására manipulálására nem célszerű). A mérés célja, hogy bemutassa a Mondrian könnyű használhatóságát és az R széles körű használhatóságát.

http://stats.math.uni-augsburg.de/mondrian/

5 Statisztikai alapfogalmak

5.1 Faktorváltozók

A faktorváltozók kategóriai változók, felvehetnek numerikus és karakterlánc értéket is. A kategorikus változók működési tartományokat, egyedi beállításokat jelölhetnek értékkészletük általában kis diszkrét halmaz. Például dohányzik-e valaki, 0 ha nem 1 ha igen.

Bővebben:

http://www.ats.ucla.edu/stat/R/modules/factor_variables.htm

5.2 Származtatott attribútumok

Gyakran előfordul, hogy az adathalmazban van egy attribútum, mely tartalmazza a számunkra fontos információt, azonban direkt nem tudjuk felhasználni. Ekkor származtatott attribútumot célszerű előállítani, mely már direkt felhasználható formában tartalmazza az információt. Például nagyon sok rekordnál megtalálható az idő attribútum, amit informatikában gyakran célszerű Unix epoch formában megadni (1970 óta eltelt idő másodpercekben). Azonban az is előfordulhat, hogy nekünk a kísérlet kezdete óta eltelt idő a fontos. Ekkor mindegyik időértékből az adathalmazban ki kell vonni a legelső időértéket. Javasolt, hogy ne írjuk felül az idő attribútumot, ilyenkor célszerű származtatott attribútumot létrehozni.

5.3 Várható érték, variancia

Ezek az alapfogalmak már szerepeltek Valószínűségszámításból, a mérés elvégzéséhez fontos tudni a legfontosabb definíciókat, alapfogalmakat.

http://hu.wikipedia.org/wiki/V%C3%A1rhat%C3%B3_%C3%A9rt%C3%A9k
http://hu.wikipedia.org/wiki/Sz%C3%B3r%C3%A1s_%28val%C3%B3sz%C3%ADn%C5%B1s%C3%A9g-sz%C3%A1m%C3%ADt%C3%A1s%29

5.4 Korreláció

A korreláció egy matematikai, statisztika fogalom. A korreláció jelzi két tetszőleges érték (például valószínűségi változó) közötti lineáris kapcsolat nagyságát (és irányát):

$$RX,Y = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

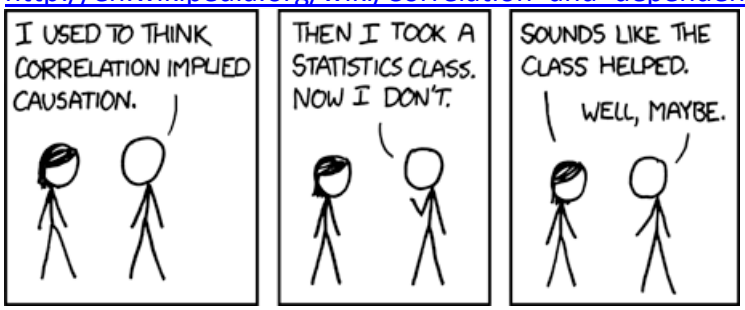
A statisztikában nem állnak rendelkezésre az elméleti értékek (mivel nem tudjuk a várható értéket és a szórást), így a tapasztalati korrelációt a következőképpen számoljuk (a tapasztalati várható értékből és a tapasztalati szórásból):

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y},$$

A korreláció értéke -1 és 1 közé esik, és egyenlőség akkor és csak akkor áll fenn, ha a két változó lineáris kapcsolatban áll egymással. A korreláció skálafüggetlen.

Fontos megjegyezni, hogy a (statisztika) korreláció nem jelent kauzalitást. Ugyanis a korreláció oka lehet egy harmadik (ismeretlen) paraméter is (root cause), vagy lehet egyszerű véletlen is.

http://en.wikipedia.org/wiki/Correlation_and_dependence



5.5 Regresszió

A regresszió definíciója: a statisztikában a regresszió számítás során két vagy több véletlen változó között fennálló kapcsolatot modellezzük. A regressziós modell tulajdonságai alapján megkülönböztetünk lineáris és nemlineáris regressziót. Bővebben:

http://hu.wikipedia.org/wiki/Regresszi%C3%B3sz%C3%A1m%C3%ADt%C3%A1s

6 Mérési feladatok

6.1 Mérési feladatok 1

Adott egy tiszta, preparált kísérleti adathalmaz. A csv első sora tartalmazza a fejlécet, valamint a mellékelt fájl magyarázza, hogy az adott csv mező neve mit jelent, valamint, hogy melyik komponenstől származik az adott mező adata.

- Tanulmányozzuk az adathalmazt, nyissuk meg egy megfelelő alkalmazással. Különösen tanulmányozzuk a fejlécet és a fejlécben szereplő nevek definícióját. Ehhez szükséges lesz az adathalmazt előállító kísérleti architektúra ismerete.
- Mondrian kipróbálása: Indítsuk el a Mondriant, töltsük be az adathalmazt, és készítsünk néhány grafikont. Milyen grafikon készített és miért? Mit láthatunk a grafikonon, mi a jelentése?
- Indítsuk el az RStudiot. Próbáljuk ki az alapvető műveleteket.
- Olvassa be az állomány egy data frame-be. Honnan tudjuk, hogy ez sikeres volt?
- Adjunk egy új oszlopot az adathalmazhoz, egy származtatott attribútum formájában (pl relatív idő, azaz a kísérlet kezdete óta eltelt idő).
- Készítsünk egy faktorváltozót és azt csatoljuk a data frame-hez (például a kísérlet első és utolsó 3 percét jelöljük meg, mint egyfajta keretet).

- Készítsünk grafikonokat az iplots használatával. Mit tapasztalunk? Értelmezze a kapott ábrákat?
  - Scatterplotok: time x CPU, mem, QoS …
  - Boxplot a QoS metrikára
  - Boxplot QoS metrika × faktorváltozó
  - Histogramm ábra a késleltetésre
  - Működési tartományok beazonosítása faktor változók segítségével
  - …
- Számoljunk korrelációt! Például a belső és VM host memória között, valamint a belső és külső CPU erőforrás használat között. Mit tapasztalunk? Értelmezze a kapott adatokat?
- Adathalmaz megjelenítése párhuzamos koordinátákkal.
- Illesszünk regressziós görbét egy tetszőleges oszlop értékeire.

6.2 Mérési feladatok 2

Adott egy valós infrastruktúrából származó (zajos) adat, csv formában. A csv első sora tartalmazza a fejlécet, valamint a mellékelt fájl magyarázza, hogy az adott csv mező neve mit jelent, valamint, hogy melyik komponenstől származik az adott mező adata.

A tanultak alapján elemezze az adatsort. Milyen alapvető összefüggéseket lehet kinyerni (trivialitások, esetleg mérnöki best practice-k visszafejtése)? Mi vonható le a vizsgált szoftverre (BigBlueButton) vonatkozóan? Mi tapasztalat vonható le a kísérletből? Mely (QoS) metrikák használhatóak, és melyek nem?

Az adatelemzést a mérésvezető segíti majd interaktív módon, így egy valós adatelemzés kerül majd megvalósításra.

7 Ellenőrző kérdések

- Mi a különbség az adatvizualizáció és az adatok vizuális elemzése között?
- Mire használják és mire előnyös az adatok vizuális elemzése?
- Mik a főbb különbségek a klasszikus adatbányászati eljárások és az adatok vizuális elemzése között?
- Mondjon példákat (3-3) alacsony és magas szintű metrikára? Egy alkalmazást futtató gép CPU terhelése alacsony vagy magas szintű metrika? Válaszát indokolja!
- Milyen grafikonokat ismer, melyeket lehet adatok vizuális elemzésére használni? Ismertesse ezket!
- Mi az a QoS? Írjon egy példát, akár ábrával, hogy egy kiszolgáló rendszerben hogyan mérné.
- Ha A-ból következik B, akkor A és B korrelál-e? Fordítva is igaz? Mondjon példát vagy ellenpéldát!
- Mik az alapvető elvárások egy interaktív vizuális adatelemző programtól?

8 További segédanyagok / függelék

8.1 R Alapok

Az R sima kalkulátorként is használható. A parancssorba beütve a következőket kapjuk:

```
> 2 + 3 * 5      # Note the order of operations.
> log (10)       # Natural logarithm with base e=2.718282
```

```
> 4^2            # 4 raised to the second power
> 3/2            # Division
> sqrt (16)      # Square root
> abs (3-7)      # Absolute value of 3-7
> pi             # The mysterious number
> exp(2)         # exponential function
> 15 %/% 4       # This is the integer divide operation
> # This is a comment line
```

Azonban nagyon sok beépített függvény van. Például az „assigment operator” (<-) ami a változóba (objektumba) menti az operátor jobb oldalán lévő értéket. Hozzárendelés után a változót használhatjuk számításokra. Az objektum nevének beütésére az objektum megjelenítésre kerül. Az R „case sensitive”.

```
> x<- log(2.843432) *pi
> x
[1] 3.283001
> sqrt(x)
[1] 1.811905
> floor(x)        # largest integer less than or equal to x (Gauss number)
[1] 3
> ceiling(x)      # smallest integer greater than or equal to x
[1] 4
```

8.2 Vektorok

Az R-rel hatékonyan és könnyen kezelhetők a vektorok.

```
> x<-c(1,3,2,10,5)    #create a vector x with 5 components
> x
[1]  1  3  2 10  5
> y<-1:5              #create a vector of consecutive integers
> y
[1] 1 2 3 4 5
> y+2                 #scalar addition
[1] 3 4 5 6 7
> 2*y                 #scalar multiplication
[1]  2  4  6  8 10
> y^2                 #raise each component to the second power
[1]  1  4  9 16 25
> 2^y                 #raise 2 to the first through fifth power
[1]  2  4  8 16 32
> y                   #y itself has not been unchanged
[1] 1 2 3 4 5
> y<-y*2
> y                   #it is now changed
[1]  2  4  6  8 10
```

Összetettebb vektor aritmetikák:

```
> x<-c(1,3,2,10,5); y<-1:5 #two or more statements are separated by semicolons
> x+y
[1]  2  5  5 14 10
> x*y
[1]  1  6  6 40 25
> x/y
[1] 1.0000000 1.5000000 0.6666667 2.5000000 1.0000000
> x^y
[1]     1     9     8 10000  3125

> sum(x)            #sum of elements in x
[1] 21
```

```
> cumsum(x)          #cumulative sum vector
[1]  1  4  6 16 21
> diff(x)            # first difference
[1]  2 -1  8 -5
> diff(x,2)          #second difference
[1] 1 7 3
> max(x)             #maximum
[1] 10
> min(x)             #minimum
[1] 1
```

Rendezés:

```
> x
[1]  1  3  2 10  5
> sort(x)                 # increasing order
[1]  1  2  3  5 10
> sort(x, decreasing=T)  # decreasing order
[1] 10  5  3  2  1
```

Komponens kinyerése:

```
> x
[1]  1  3  2 10  5
> length(x)           # number of elements in x
[1] 5
> x[3]                # the third element of x
[1] 2
> x[3:5]              # the third to fifth element of x, inclusive
[1]  2 10  5
> x[-2]               # all except the second element
[1]  1  2 10  5
> x[x>3]              # list of elements in x greater than 3
[1] 10  5
```

Logikai vektor:

```
> x>3
[1] FALSE FALSE FALSE  TRUE  TRUE
> as.numeric(x>3)     # as.numeric() function coerces logical components to
numeric
[1] 0 0 0 1 1
> sum(x>3)            # number of elements in x greater than 3
[1] 2
> (1:length(x))[x<=2] # indices of x whose components are less than or equal to 2
[1] 1 3
> z<-as.logical(c(1,0,0,1)) # numeric to logical vector conversion
> z
[1]  TRUE FALSE FALSE  TRUE
```

Character vektor:

```
> colors<-c("green", "blue", "orange", "yellow", "red")
> colors
[1] "green"  "blue"   "orange" "yellow" "red"
```

Egyedi komponenseknek nevet lehet adni, és ezzel a névvel lehet rájuk hivatkozni:

```
> names(x)            # check if any names are attached to x
NULL
> names(x)<-colors    # assign the names using the character vector colors
> names(x)
[1] "green"  "blue"   "orange" "yellow" "red"
> x
 green   blue orange yellow    red
     1      3      2     10      5
> x["green"]          # component reference by its name
green
    1
```

```
> names(x)<-NULL      # names can be removed by assigning NULL
> x
[1]  1  3  2 10  5
```

Bizonyos repetitív mintákat könnyű csinálni:

```
> seq(10)
 [1]  1  2  3  4  5  6  7  8  9 10
> seq(0,1,length=10)
 [1] 0.0000000 0.1111111 0.2222222 0.3333333 0.4444444 0.5555556 0.6666667
 [8] 0.7777778 0.8888889 1.0000000
> seq(0,1,by=0.1)
 [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
> rep(1,3)
[1] 1 1 1
> c(rep(1,3),rep(2,2),rep(-1,4))
[1]  1  1  1  2  2 -1 -1 -1 -1
> rep("Small",3)
[1] "Small" "Small" "Small"
> c(rep("Small",3),rep("Medium",4))
[1] "Small"  "Small"  "Small"  "Medium" "Medium" "Medium" "Medium"
> rep(c("Low","High"),3)
[1] "Low"  "High" "Low"  "High" "Low"  "High"
```

8.3 Data Frame

Gyakorlatilag egy CSV (comma separated value) fájlt képes tárolni, azaz egy vektor, mely oszlopokat tartalmaz, ahol az egyes oszlopok különböző adattípusok lehetnek. Egy data frame összerakása (a végeredményen látható az összerakott adathalmas):

```
> Make<-c("Honda","Chevrolet","Ford","Eagle","Volkswagen","Buick","Mitsbusihi",
+ "Dodge","Chrysler","Acura")
> Model<-c("Civic","Beretta","Escort","Summit","Jetta","Le Sabre","Galant",
+ "Grand Caravan","New Yorker","Legend")

> Cylinder<-c(rep("V4",5),"V6","V4",rep("V6",3))
> Cylinder
 [1] "V4" "V4" "V4" "V4" "V4" "V6" "V4" "V6" "V6" "V6"
> Weight<-c(2170,2655,2345,2560,2330,3325,2745,3735,3450,3265)
> Mileage<-c(33,26,33,33,26,23,25,18,22,20)
> Type<-
c("Sporty","Compact",rep("Small",3),"Large","Compact","Van",rep("Medium",2))

> Car<-data.frame(Make,Model,Cylinder,Weight,Mileage,Type)
> Car
         Make          Model Cylinder Weight Mileage    Type
1       Honda          Civic       V4   2170      33  Sporty
2   Chevrolet       Beretta       V4   2655      26 Compact
3        Ford        Escort       V4   2345      33   Small
4       Eagle        Summit       V4   2560      33   Small
5  Volkswagen         Jetta       V4   2330      26   Small
6       Buick      Le Sabre       V6   3325      23   Large
7  Mitsbusihi        Galant       V4   2745      25 Compact
8       Dodge Grand Caravan       V6   3735      18     Van
9    Chrysler    New Yorker       V6   3450      22  Medium
10      Acura        Legend       V6   3265      20  Medium

> names(Car)
[1] "Make"     "Model"    "Cylinder" "Weight"   "Mileage"  "Type"
```

Az indexelés a következőképp működik:

```
> Car[1,]
   Make Model Cylinder Weight Mileage   Type
1 Honda Civic       V4   2170      33 Sporty
```

Az egyes oszlopokra így hivatkozhatunk:


```
> Car[1,]
   Make Model Cylinder Weight Mileage   Type
1 Honda Civic       V4   2170      33 Sporty
```


In addition, individual columns can be referenced by their labels:

```
> Car$Mileage
 [1] 33 26 33 33 26 23 25 18 22 20
> Car[,5]        #equivalent expression, less informative
> mean(Car$Mileage)    #average mileage of the 10 vehicles
[1] 25.9
> min(Car$Weight)
[1] 2170
```

Rendezés adott oszlop szerint:

```
> i<-order(Car$Weight);i
 [1]  1  5  3  4  2  7 10  6  9  8
> Car[i,]
         Make         Model Cylinder Weight Mileage    Type
1       Honda         Civic       V4   2170      33  Sporty
5  Volkswagen         Jetta       V4   2330      26   Small
3        Ford        Escort       V4   2345      33   Small
4       Eagle        Summit       V4   2560      33   Small
2   Chevrolet       Beretta       V4   2655      26 Compact
7  Mitsbusihi        Galant       V4   2745      25 Compact
10      Acura        Legend       V6   3265      20  Medium
6       Buick     Le Sabre       V6   3325      23   Large
9    Chrysler    New Yorker       V6   3450      22  Medium
8       Dodge Grand Caravan       V6   3735      18     Van
```

Az adatok utólagos szerkesztése

```
> y<-edit(y)
```

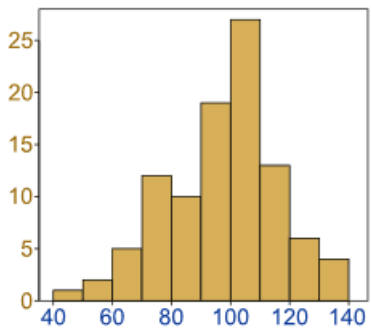
Vagy grafikus felülettel:

```
> data1<-edit(data.frame())
```

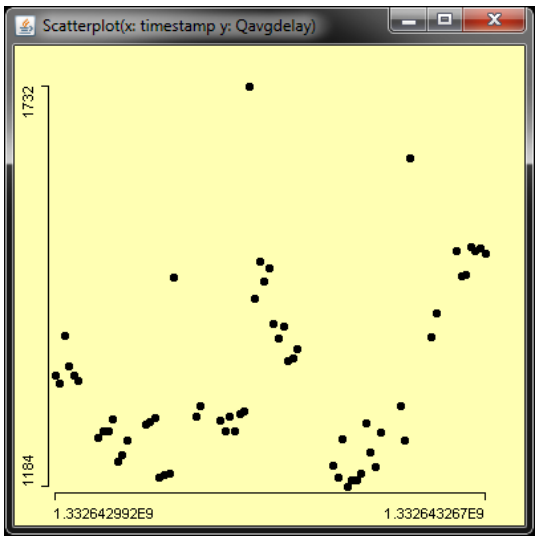
8.4 Plotok (iplots csomag)

Az iplots az R egy statisztikai csomagja, mely interaktív statisztikai grafikonok Java alapú megjelenítésére képes.

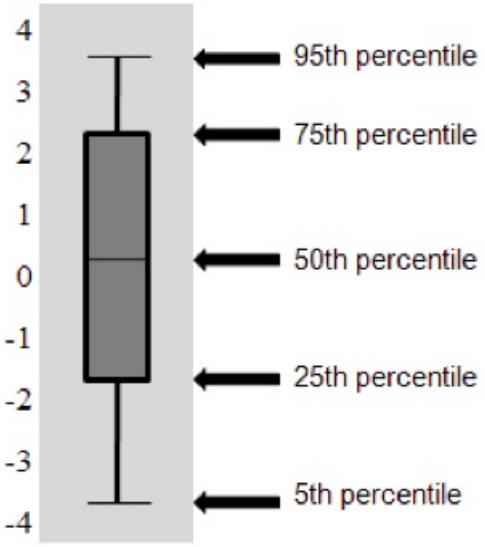
Histogram - ihist(...) - Magáért beszél...



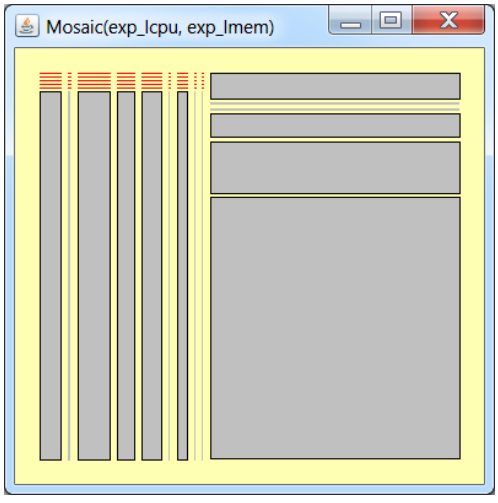
Scatterplot - iplot(...) - Interaktív x – y koordináta készítése



Boxplot - ibox(...) - Adott populáció eloszlását mutatja. Alsó 25 % és felső 25 %, átlag, minimum és maximum érték, esetleges kívülálló értékek.



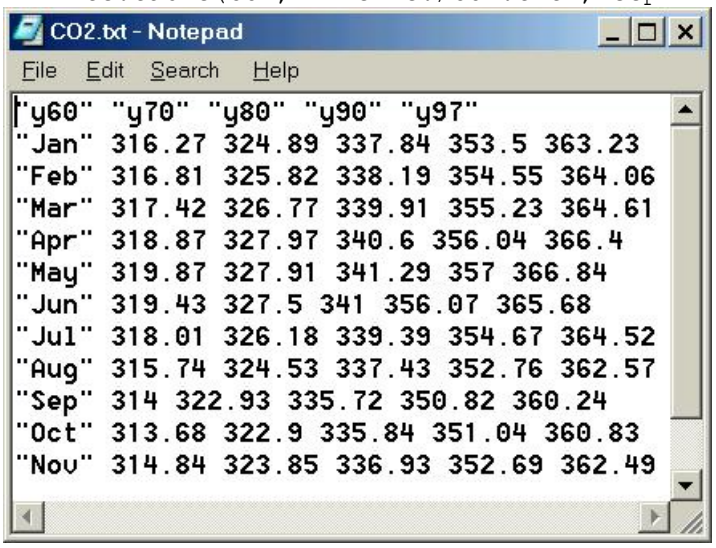
Mosaic plot - imosaic(...) - segítségével kivallóan lehet működési tartományokat megjenesíteni.



8.5 Adat exportálás és importálás

Az R-ből egyszerűen lehet szöveges állományokba exportálni, vagy onnan importálni. Exportálásra egy példa:

```
> CO2     # data frame
> write.table(CO2, file="c:/CO2.txt", sep=" ")
```



Importálásra (és az elérési út manipulálása a):

```
> getwd()
[1] "C:\\Program Files\\R\\rw1070"
> setwd("c:/")     # set the root directory as the working directory
> getwd()
[1] "c:\\"
> read.table(file="CO2.txt")
> # now pathname is not required to read data files in the root directory
```

8.6 Hiányos értékek (NA)

NA (NotApplicable) a hiányzó értékeket jelöli.

```
> x                         #contains a missing value
[1]  1  2  3  4  5 NA
> mean(x)                   #doesn't work
[1] NA
> is.na(x)                  #returns a logical vector
[1] FALSE FALSE FALSE FALSE FALSE  TRUE
> sum(is.na(x))             #number of NA's in the vector
[1] 1
> x1<-x[!is.na(x)];x1       #retain only non-missing cases
[1] 1 2 3 4 5
> a<-mean(x[!is.na(x)]);a   #compute the average value of the non-missing cases
[1] 3
> x2<-x
> x2[is.na(x)]<-a;x2        #impute the missing by the average value
[1] 1 2 3 4 5 3
```

Még több példa:

```
> data2
  var1 var2 var3
1    1  2.3   aa
2    4  3.2 <NA>
3    3  5.4   bc
4   NA  2.7   ed
```

```
5    3  4.1   dd
> a1<-!is.na(data2$var1);a1    #TRUE if nonmissing for var1
[1]  TRUE  TRUE  TRUE FALSE  TRUE
> a2<-!is.na(data2$var2);a2
[1] TRUE TRUE TRUE TRUE TRUE
> a3<-!is.na(data2$var3);a3
[1]  TRUE FALSE  TRUE  TRUE  TRUE
> data3<-data2[a1*a2*a3==1,]
> #select those rows if all of the elements are nonmissing.
> data3
  var1 var2 var3
1    1  2.3   aa
3    3  5.4   bc
5    3  4.1   dd
```

8.7 Help

Az R-nek számos kézikönyve van pdf formában („An Introduction to R”: http://cran.r-project.org/doc/manuals/R-intro.pdf). A kézikönyvek eléréséhez a …

Beépített parancs a segítségre: help()

```
> help(read.table)
```

8.8 Metrikák (a 2. Feladat metrikái)

Metrics:	Source:	Description:	Dimension:
%user	SAR-CPU	The percentage of time the CPU is spending on user processes, such as applications, shell scripts, or interacting with the user.	%
%nice	SAR-CPU	Percentage of CPU utilization that occurred while executing at the user level with nice priority.	%
%system	SAR-CPU	The percentage of time the CPU is spending executing kernel tasks. In this example, the number is high, because I was pulling data from the kernel's random number generator.	%
%iowait	SAR-CPU	The percentage of time the CPU is waiting for input or output from a block device, such as a disk.	%
%steal	SAR-CPU	???	%
%idle	SAR-CPU	The percentage of time the CPU isn't doing anything useful.	%
kbmemfree	SAR_Mem	Amount of free memory available in kilobytes.	kb

kbmemused	SAR_Mem	Amount of used memory in kilobytes. This does not take into account memory used by the kernel itself.	kb
%memused	SAR_Mem	Percentage of used memory.	%
kbbuffers	SAR_Mem	Amount of memory used as buffers by the kernel in kilobytes.	kb
kbcached	SAR_Mem	Amount of memory used to cache data by the kernel in kilobytes.	kb
kbcommit	SAR_Mem	Amount of memory in kilobytes needed for current work load. This is an estimate of how much RAM/swap is needed to guarantee that there never is out of memory.	kb
%mem.commit	SAR_Mem	Percentage of memory needed for current workload in relation to the total amount of memory (RAM+swap). This number may be greater than 100% because the kernel usually overcommits memory.	%
users	bbb	Number of users logged into BBB	number
ethernetBytes	ntop	Summ. Of bytes send by the server. Aggregated.	bytes
net.usage.average	ESXi	Network utilization (combined transmit- and receive-rates) during the interval	Kb
cpu.usage.average	ESXi	CPU usage as a percentage during the interval	%
cpu.usagemhz.average	ESXi	CPU usage, as measured in megahertz, during the interval	Mhz
sys.uptime.latest	ESXi	Total time elapsed, in seconds, since last system startup	second
mem.usage.average	ESXi	Memory usage as percentage of total configured or available memory	%
cpu.usagemhz.average	ESXi	CPU usage, as measured in megahertz, during the interval	Mhz
fs	MP3	Frequency	Hz
b	MP3	Bitrate	bit

drive	MP3	"kivezérlési határhoz képest mennyit használok"	%
clip	MP3	Volt-e túlvezérlés	boolean
clip_rate	MP3	0.1 seces intervallumok hány %-ban volt túlvezérlés	%
B	MP3	??? Sávszélesség	Hz
B_ratio	MP3	"bemenő / kijövő sávszélesség aránya"	%
delay_1	MP3	Késleltetés	ms
delay2_mean	MP3	kereszkorreláció - középérték (???)	ms
delay2_dev	MP3	kereszkorreláció - szórás	ms

9 Egyéb Hivatkozások:

http://www.cs.bme.hu/~bodon/magyar/adatbanyaszat/tanulmany/adatbanyaszat.pdf
http://nm.merz-akademie.de/~jasmin.sipahi/drittes/images/Keim2002.pdf
http://math.illinoisstate.edu/dhkim/rstuff/rtutor.html
http://cran.r-project.org/doc/manuals/R-intro.pdf