2 Laboratorinis Darbas (EM)

Tadas Danielius

1 Dalis

Turime kintamųjų x,y,z duomenis D su praleistomis reikšmėmis.

```
matrices$D
```

```
##          x   y   z
##  [1,]  2.6 0.0 9.2
##  [2,]  0.0 2.4 2.5
##  [3,]  5.6 1.2 6.2
##  [4,] -2.7 0.0 0.0
##  [5,]  7.1 0.0 0.0
##  [6,]  0.0 5.0 0.0
##  [7,]  3.4 0.5 2.3
##  [8,] -6.8 4.2 4.6
##  [9,]  0.0 0.0 0.0
```

(a) Kai duomenys nekoreliuoti

Kadangi duomenys yra nekoreliuoti ir turi Gauso skirstinį, tai pritaikome EM algoritmą (užpildome empyrinemiais vidurkiais) ir gauname rezultatus:

```
t(sapply(1:dim(matrices$D)[1],
                    impute,
                    V=matrices$V_diag,
                    M=matrices$M,
                    D=matrices$D,
                    s=matrices$s))
```

```
##            [,1]     [,2] [,3]
##  [1,]  2.600000 2.216667 9.20
##  [2,]  1.533333 2.400000 2.50
##  [3,]  5.600000 1.200000 6.20
##  [4,] -2.700000 0.000000 4.96
##  [5,]  7.100000 2.216667 4.96
##  [6,]  1.533333 5.000000 4.96
##  [7,]  3.400000 0.500000 2.30
##  [8,] -6.800000 4.200000 4.60
##  [9,]  1.533333 2.216667 4.96
```

(b) Kai duomenys koreliuoti

Atliekame 1-ą EM iteraciją

```
m = load_data()
run_iterations(m, max=1, epsilon = 0.001)$D
```

```
##  iteration 1 ML: 0.206662110582773
```

```
##             [,1]      [,2]     [,3]
##  [1,]  2.6000000 2.5774639 9.200000
##  [2,]  0.1424061 2.4000000 2.500000
##  [3,]  5.6000000 1.2000000 6.200000
##  [4,] -2.7000000 0.0000000 3.114969
##  [5,]  7.1000000 0.9263808 5.519939
##  [6,] -2.8341252 5.0000000 5.306650
##  [7,]  3.4000000 0.5000000 2.300000
##  [8,] -6.8000000 4.2000000 4.600000
##  [9,]  1.5333333 2.2166667 4.960000
```

Atliekame 2-ą EM iteraciją

```
m = load_data()
run_iterations(m, max=2, epsilon = 0.001)$D
```

```
##  iteration 2 ML: 0.0139344400001614
```

```
##             [,1]      [,2]     [,3]
##  [1,]  2.6000000 2.7473164 9.200000
##  [2,] -0.9828586 2.4000000 2.500000
##  [3,]  5.6000000 1.2000000 6.200000
##  [4,] -2.7000000 0.0000000 2.688552
##  [5,]  7.1000000 0.9400901 5.604237
##  [6,] -2.8978741 5.0000000 5.745596
##  [7,]  3.4000000 0.5000000 2.300000
##  [8,] -6.8000000 4.2000000 4.600000
##  [9,]  0.8935127 2.1133902 4.855729
```

Atliekame 3-ą EM iteraciją

```
m = load_data()
run_iterations(m, max=3, epsilon = 0.001)$D
```

```
##  iteration 3 ML: 0.0172843854313036
```

```
##             [,1]      [,2]     [,3]
##  [1,]  2.6000000 3.0201026 9.200000
##  [2,] -1.6016783 2.4000000 2.500000
##  [3,]  5.6000000 1.2000000 6.200000
##  [4,] -2.7000000 0.0000000 2.213643
##  [5,]  7.1000000 0.9665191 5.740753
##  [6,] -2.9738293 5.0000000 6.110547
##  [7,]  3.4000000 0.5000000 2.300000
##  [8,] -6.8000000 4.2000000 4.600000
##  [9,]  0.6903089 2.1223107 4.854902
```

Skaičiuojame tol, kol didžiausio tikėtinumo skirtumas taps mažesnis nei 0.001

```
m = load_data()
res1 = run_iterations(m, max=1000, epsilon = 0.001, ml.cap=10)
```

```
##  iteration 372 ML: 1.77784527500003e-10
```

```
res1$D
```

```
##             [,1]      [,2]      [,3]
##  [1,]  2.6000000 3.0800182  9.200000
##  [2,] -3.0071768 2.4000000  2.500000
##  [3,]  5.6000000 1.2000000  6.200000
##  [4,] -2.7000000 0.0000000 -4.285208
##  [5,]  7.1000000 0.8092924  6.311955
##  [6,] -4.0027421 5.0000000  9.294557
##  [7,]  3.4000000 0.5000000  2.300000
##  [8,] -6.8000000 4.2000000  4.600000
##  [9,]  0.2737601 2.1486638  4.515163
```

Reikėjo atlikti 372 iteracijas kol log-likelihood reikšmė tapo mažesnė už 0.001

(3) Simuliacija

Nekoreliuotos reikšmės

Atliekame simuliaciją 100 kartų kai (x,y,z) komponentės yra tarpusavyje **nekoreliuotos**

```
results = simulate.run(n, 0.001, F, correlated = FALSE)
```

```
## Total simulations 100 Epsilon: 0.001
```

```
message('MAPE Rezultatai')
```

```
## MAPE Rezultatai
```

```
summary(results$mape)
```

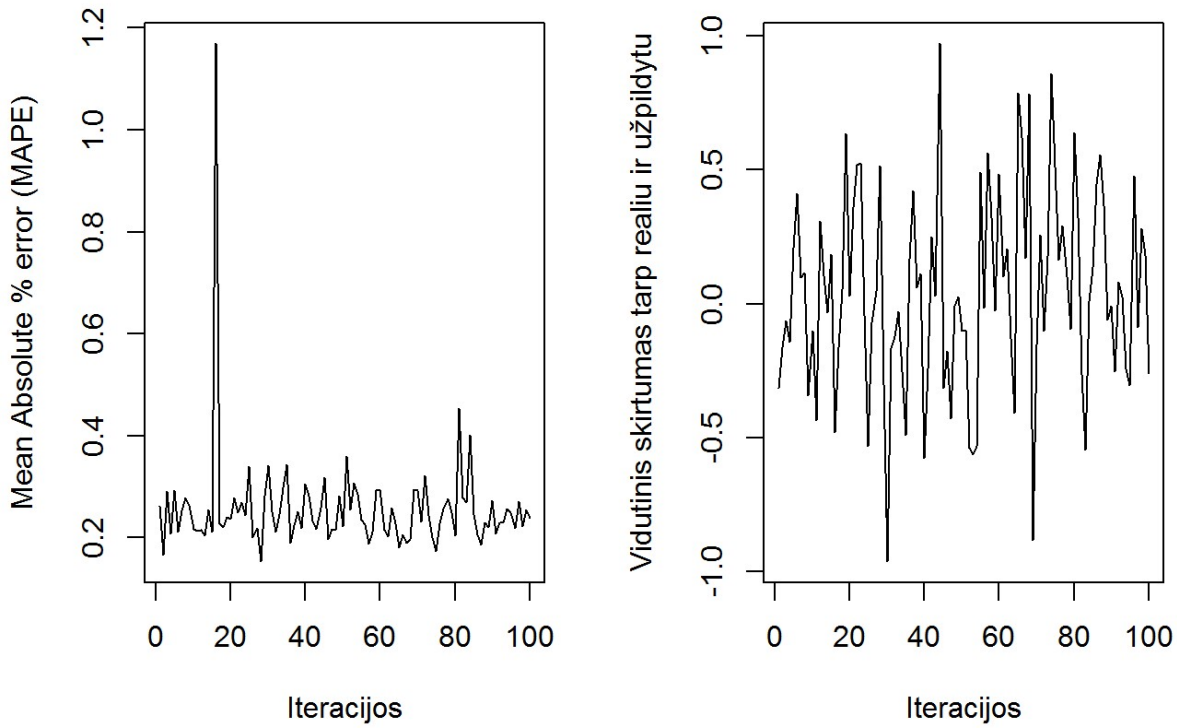
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1528  0.2146  0.2393  0.2560  0.2761  1.1690
```

```
message('Vidutinis skirtumas tarp tikrųjų ir įvertintų reikšmių')
```

```
## Vidutinis skirtumas tarp tikruju ir ivertintu reikšmiu
```

```
summary(results$errors)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.96120 -0.16890  0.02741  0.04025  0.28480  0.97110
```



Koreliuotos reikšmės

Atliekame simuliaciją 100 kartų kai (x,y,z) komponentės yra tarpusavyje koreliuotos

```
res_corr = simulate.run(n, 0.001, F, correlated = T)
```

```
## Total simulations 100 Epsilon: 0.001
```

```
message('MAPE Rezultatai')
```

```
## MAPE Rezultatai
```

```
summary(res_corr$mape)
```

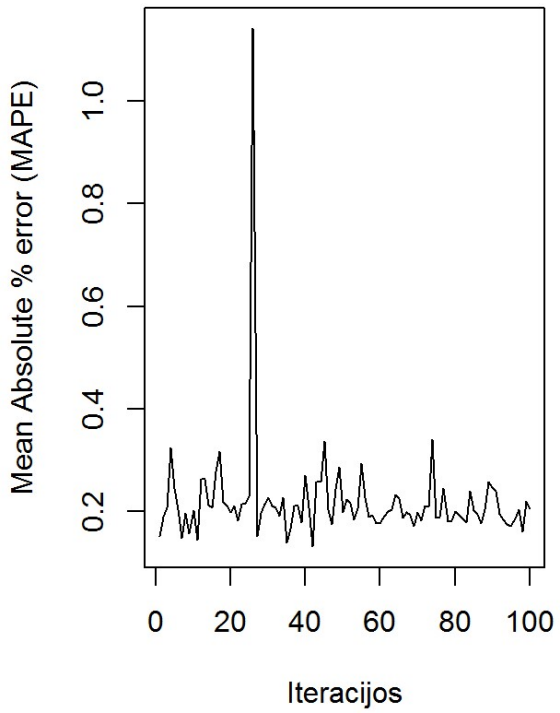
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1320  0.1856  0.2040  0.2190  0.2260  1.1410
```

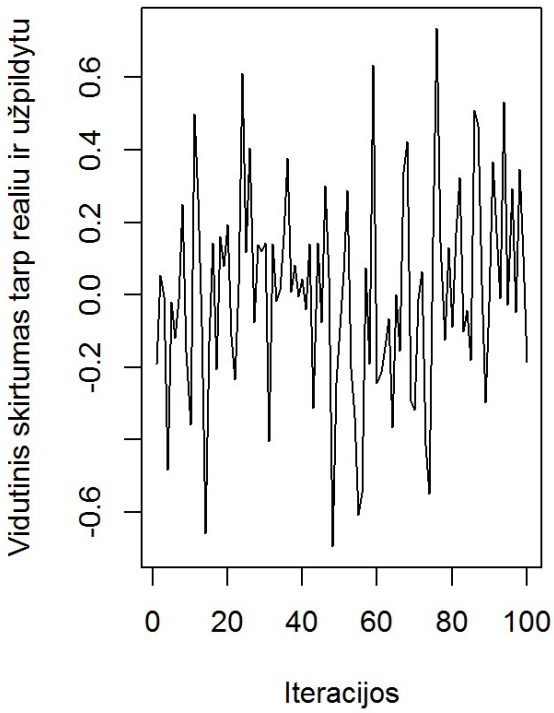
```
message('Vidutinis skirtumas tarp tikrųjų ir įvertintų reikšmių')
```

```
## Vidutinis skirtumas tarp tikruju ir ivertintu reikšmiu
```

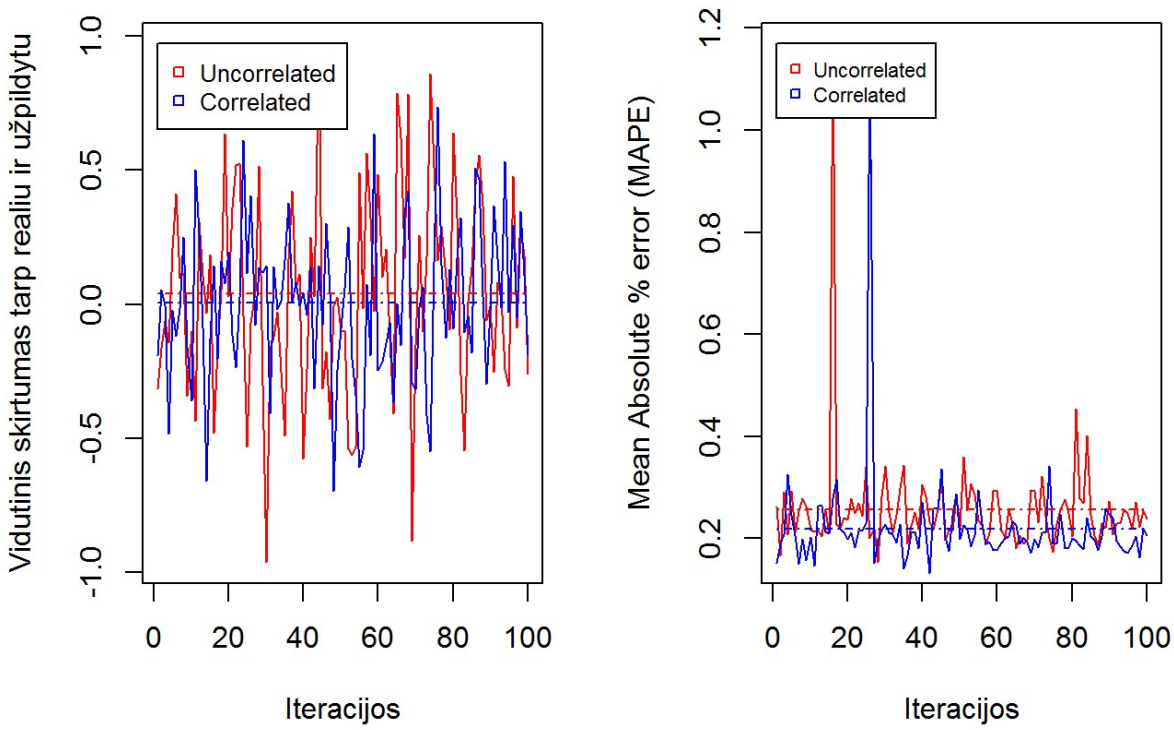
```
summary(res_corr$errors)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.694700 -0.163800 -0.005594  0.005267  0.148800  0.734700
```





Bendras grafikas



Akivaizdu, kai duomenys koreliuoti rezultatai kur kas geresni. Koreliuotų duomenų MAPE reikšmė yra 0.2189797 o nekoreliuotų 0.2560436.

Didžiausio tikėtinumo vertinimas

Kadangi naudojant EM algoritmą maksimizuojam log-likelihood funkciją, tai rezultatai jau yra sukonvergavusios reikšmės.

Kai reikšmės nekoreliuotos

```
summary(results$ml)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.7366  0.8072  0.8484  0.8464  0.8806  0.9377
```

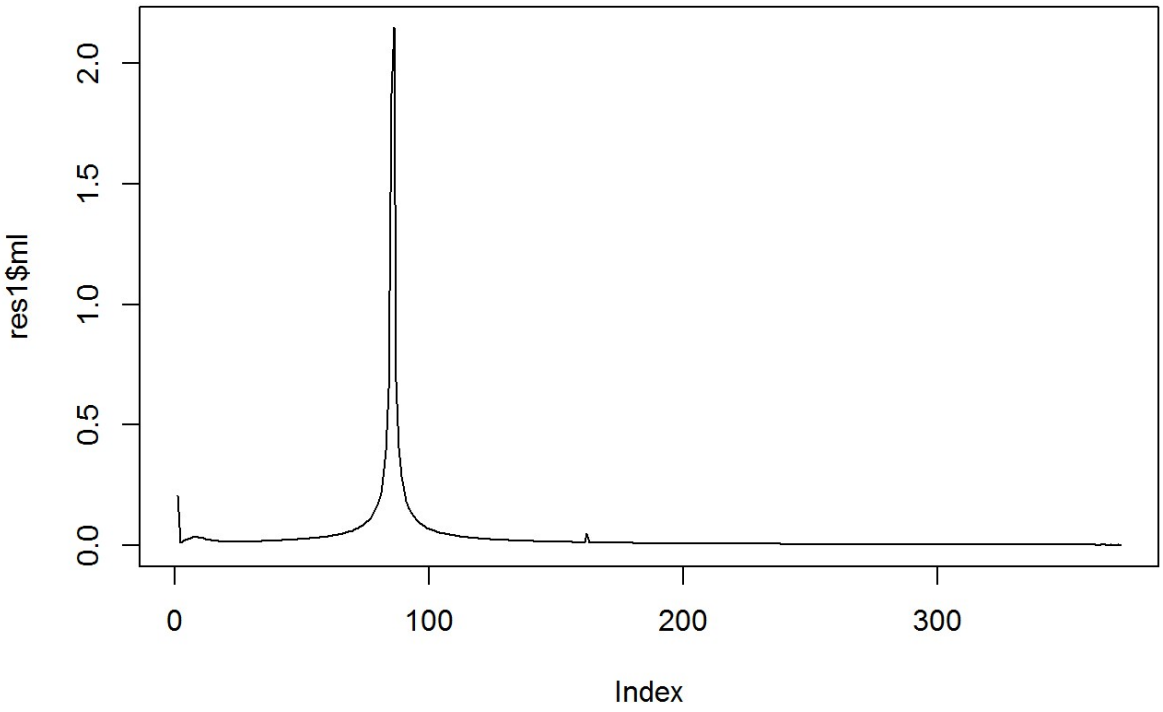
Ir kai reikšmės koreliuotos

```
summary(res_corr$ml)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.007365 0.053210 0.093640 0.133700 0.165300 0.767100
```

Galime grafiškai pavaizduoti kaip kito log-likelihood reikšmė atliekant pirmą užduotį

```
plot(res1$ml, type='l')
```




```
summary(res1$ml)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.000000 0.005099 0.009671 0.038780 0.023850 2.147000
```