P7 - Markov Kateak

Prozesu estokastikoak eta Markov kateak

Prozesu estokastiko terminoak bi aspekturi egiten die erreferentzia: alde batetik *prozesu* hitzak, gertaera sekuentzia bati buruz ari garela dio; bestetik, *estokastiko* hitzak ausazko gertaerak ditugula adierazten digu. Hau hala izanik, prozesu estokastiko bat probabilitate espazio berean definitutako zorizko aldagai sekuentzia bat bezala definitu dezakegu:

$$\{X_t \ : t \in T\}$$

Laborategi honetan aztertuko dugun kasuan, $T = \mathbb{N}$ izango da, eta denbora diskretuko aldagai sekuentzia infinitua izango dugu.

Oso garrantzitsua da prozesua bera eta prozesu horren behaketak ezberdintzea. Lehenengoa, esan bezala, zorizko aldagai sekuentzia bat da (teorikoa da, ezin da behatu); bigarrenak aldiz, prozesuaren laginak edo behaketak izango dira eta $x_1, x_2, ..., x_n, ...$ moduan adieraziko ditugu, letra txikiz.

Markov kateak zorizko aldagaien kasu partikularrak dira, non Markov propietatea betetzen den:

$$\forall i \in T \quad P(X_{i+1}|X_1, X_2 \ldots, X_i) = P(X_{i+1}|X_i)$$

Hau da, denbora pausu bakoitzeko probabilitateak justu aurreko pausuan gertatutakoak zehazten ditu, aurretik gertatutako guztiak eraginik izan gabe.

Gardenkietan robotaren adibidea ikusi dugu. Adibide horretan, robota ausaz mugitzen da prozesu estokastiko bat jarraituz. Prozesu horretan, denbora pausu bakoitzean, lau aukera daude: *goruntz*, *beheruntz*, *ezkerreruntz* eta *eskuineruntz*. Mugimendu bakoitzaren probabilitateak zehazteko, aurreko pausuan egindakoa bakarrik hartzen denez kontutan, prozesu estokastiko hori Markov kate bat dela ikusi genuen. Zehazki, mugimendu berdina errepikatzeko probabilitatea 0.7 da eta beste edozein aukeratzekoa 0.1. Trantsitzio-probabilitate hauek matrize baten bidez adierazi ditzazkegu:

$$P = \begin{pmatrix} 0.7 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.7 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.7 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.7 \end{pmatrix}$$

Matrize honek prozesu estokastiko hau guztiz definitzen digu (Markov katea), eta bera aplikatuz edo laginduz, behaketa-sekuentzia bat atera dezakegu. Prozesu estatistiko bat izanda, prozesua aplikatzen dugun bakoitzean, behaketa-sekuentzia desberdin bat lortuko dugu.

Laborategi honetan, Markov kate horren laginketa eskuz inplementatuko dugu. Horretarako, lehenik `factor` motako bektore kategoriko bat definituko dugu, hasierako egoera definituz. Faktore horren balio posibleak `U` (*gora*), `D` (*behera*), `L` (*ezkerrera*) eta `R` (*eskuinera*) izango dira. Gainera, asumituko dugu lehenengo instantean kateak `U` balioa hartuko duela (hau da, robota goruntz joango dela hasieran). Ondoren, transizio-probabilitateak matrize formatuan gordeko ditugu.

```
egoerak <- c("U","D","L","R")  #Egoera posibleak
hasierako.egoera <- factor ("U", levels=egoerak)  #Factor motako bektore bat
trantsizio.matrizea <- rbind(c(0.7, 0.1, 0.1, 0.1),
                             c(0.1, 0.7, 0.1, 0.1),
                             c(0.1, 0.1, 0.7, 0.1),
                             c(0.1, 0.1, 0.1, 0.7))
colnames(trantsizio.matrizea) <- egoerak
rownames(trantsizio.matrizea) <- egoerak
trantsizio.matrizea
```

```
##     U   D   L   R
## U 0.7 0.1 0.1 0.1
## D 0.1 0.7 0.1 0.1
## L 0.1 0.1 0.7 0.1
## R 0.1 0.1 0.1 0.7
```

Markov kateen laginketa

Orain, katea lagintzeari ekingo diogu. Horretarako, aurreko balioa begiratu eta horren arabera hurrengoa aukeratu behar dugu, probabilistikoki, P matrizeko dagokion lerroa erabiliz. Prozesu hau inplementatzeko funtzio bat sortu aurretik, ikus dezagun adibide bat eskuz.

Sekuentziako lehenengo balioa (X_1 aldagaiaren behaketa) `U` dela esan dugu, hau da, robota goruntz mugitu da. Hurrengo balioa zehazteko, hau da, X_2 aldagaiaren behaketa zehazteko, $P(X_2|X_1 = U)$ probabilitateak erabili behar ditugu, trantsizio matrizearen 1. lerroan gordeta daudenak.

```
probabilitateak <- trantsizio.matrizea[hasierako.egoera, ]
probabilitateak
```

```
##   U   D   L   R
## 0.7 0.1 0.1 0.1
```

Ikus dezakegunez, hurrengo pausua goruntz (`U`) emateko probabilitatea 0.7 da eta beste mugimendu mota guztiek 0.1eko probabilitatea dute. Orain, lagindu dezagun X_2 aldagaiaren balioa probabilitate hauek erabiliz. Hau eskuz egitea ez da zaila, baina `sample` funtzioaren bitartez are errazago egin dezakegu:

```
behaketa.berria <- sample(egoerak, size=1, prob=probabilitateak)
behaketa.berria
```

```
## [1] "L"
```

Behin X_2=L finkatuta, X_3 lagintzera pasa gaitezke. Prozesua berdin-berdina da, baina kasu honetan, probabilitateen balioak aurreko pausuan L motako mugimendua egin dugula kontuan hartu beharko dute. Egia esan, probabilitateak aurreko egoeraren menpekoak direnez soilik, prozesua berdina da i guztietarako, eta funtzio batean inplementatu dezakegu:

```
laginduMarkovKatea <- function(lagin.tamaina, hasierako.egoera, trantsizio.matrizea, egoerak) {
  behatutako.balioak <- as.character(hasierako.egoera)
  for (i in 1:lagin.tamaina) {
    azken.behaketa <- tail(behatutako.balioak, n=1)[[1]]
    probabilitateak <- trantsizio.matrizea[azken.behaketa, ]
    behaketa.berria <- sample(egoerak, size=1, prob=probabilitateak)
    behatutako.balioak <- c(behatutako.balioak, behaketa.berria)
  }
  behatutako.balioak <- factor(behatutako.balioak, levels=egoerak)
```

```
  return(behatutako.balioak)
}
```

Funtzioa probatzeko, 25 tamainako behaketa-sekuentzia bat sortuko dugu.

```
sekuentzia.25 <-laginduMarkovKatea(lagin.tamaina=25,
                                   hasierako.egoera="U",
                                   trantsizio.matrizea=trantsizio.matrizea,
                                   egoerak=egoerak)

sekuentzia.25
```

```
##  [1] U U D D D L U U U U L L L L L R R D D D U U U U R R
## Levels: U D L R
```

Gainera, lehen esan bezala, behin prozesu estokastikoa definituta, nahi adina aldiz lagindu dezakegu, behaketa-sekuentzia ezberdinak lortuz.

Ariketa: Errepikatu laginketa behin baino gehiagotan eta konparatu lortutako sekuentziak.

Probabilitate marjinalak

Markov kateen erabileran aspektu oinarrizko bat X_i aldagaien banaketa marjinalak dira, $P(X_i)$ balioak hain zuzen ere. Probabilitate hauek aztertzeko bi modu ikusiko ditugu: teoria erabiliz eta simulazioa erabiliz.

Simulazioa erabiliz

Hasteko, simulazioa erabiliko dugu probabilitate hauek hurbiltzeko. Horretarako, Markov katea laginduko dugu hainbat aldiz, eta $P(X_i)$ valioak estimatuko ditugu behaketa maiztasunak erabiliz.

Robotaren adibidea hartuz, Markov katea 10000 aldiz laginduko dugu, 10000 behaketa sekuentzia lortuz. Normalean 1. balioa (hau da, X_1 aldagaiaen balioa) ausaz hartzen da, baina gure kasuan, sekuentzia guztiak balio berdinetik hasiko ditugu, $X_1 = U$ baliotik hain zuzen ere. Konputazio kostua mugatzeko, 20 tamainako sekuentziak laginduko ditugu (hau da, X_1-tik X_{20}-ra). Kontutan izan X_1-eko balioa finkatuta dagoela, eta, beraz, bakarrik 19 iterazio egin beharko ditugu.

```
lagin.tamaina <- 10000
aux <- lapply(1:lagin.tamaina,
              FUN=function (i){
                laginduMarkovKatea(lagin.tamaina=19,
                                   hasierako.egoera="U",
                                   trantsizio.matrizea=trantsizio.matrizea,
                                   egoerak=egoerak)
              })
```

`lapply` funtzioa erabili dugu katea lagintzen duen funtzioa (`laginduMarkovKatea`) behin eta berriro (10000 aldiz) aplikatzeko. Funtzio honek emaitza bektore zerrenda batean (`aux`) itzultzen dit, hau da, lagindutako behaketa kate bakoitza zerrendako posizio batean gordeko du bektore moduan. Adibidez, 15. behaketa katea honela lortu dezakegu:

```
aux[[15]]
```

```
##  [1] U L L U D D R R L L L L L L R R U U U U
## Levels: U D L R
```

Zerrenda hau matrize bihurtzeko `do.call` funtzioa erabili dezakegu. Kode lerro hauen bitartez, zerrenda osoari `rbind` funtzioa aplikatzen diogu eta zerrendako bektore bakoitza matrizearen errenkada batean jartzen dugu. Arazo bakarra da, hau egitean, faktoreak zenbaki bihurtzen direla (1,2,3 eta 4), `U`, `D`, `L` eta `R` izan beharrean. Adibidez, lehen erakutsi dugun 15. behaketa katea bistaratu dezakegu:

```
mvkatea.behaketa <- do.call(rbind, aux)
mvkatea.behaketa[15,]
```

```
##  [1] 1 3 3 1 2 2 4 4 3 3 3 3 3 3 4 4 1 1 1 1
```

Orain behaketa-kate guztiak matrize batean gordeta ditugula, ohartu, i. zutabean X_i aldagaiaren behaketak gordeta ditugula. `table` funtzioa erabiliz, zutabe jakin bateko balio posible bakoitzaren maiztasunak kalkulatu ditzakegu (aldez aurretik, zutabea berriro fakore bihurtuz) eta horiek erabiliz probabilitate marjinalak estimatu. Adibidez, 1. zutaberako:

```
x1 <- factor(mvkatea.behaketa[,1], levels=1:length(egoerak)) #1) level numerikoak ezarriko ditugu
levels(x1) <- egoerak #2) level numerikoak gure egoerekin ordezkatuko ditugu
head(x1)
```

```
## [1] U U U U U U
## Levels: U D L R
```

```
table(x1) / lagin.tamaina
```

```
## x1
## U D L R
## 1 0 0 0
```

Espero moduan, X_1 aldagaiaren probabilitateak adierazten digu honek beti balio berdina hartzen duela, `U`. Noski, gauza bera errepikatzen badugu 2. zutaberako, hau da X_2 aldagairako, ez dugu emaitza hau lortuko.

Ariketa: Jarraitu aurretik, ikusitako teoria erabiliz, pentsatu zein izango den X_2 aldagaiaren banaketa marjinala kontuan izanda emandako trantsizio matrizea eta jakinda $X_1 = U$ dela beti.

Orain bai, azter dezagun emaitza enpirikoki simulazioak erabiliz eta konparatu emaitzak:

```
x2 <- factor(mvkatea.behaketa[,2], levels=1:length(egoerak))
levels(x2) <- egoerak
head(x2)
```

```
## [1] U L D R R U
## Levels: U D L R
```

```
table(x2) / lagin.tamaina
```

```
## x2
##      U      D      L      R
## 0.6961 0.0960 0.1070 0.1009
```

Teoria erabiliz

Nahiz eta aurreko atalean X_i aldagaien probabilitate banaketa marjinala simulazio bidez hurbildu dugun, Markov kateen kasuan, ez da zaila probabilitate banaketa hau modu analitikoan lortzea.

X_2 aldagaiaren kasuan, probabilitate banaketa marjinala tribiala da, izan ere, X_1 aldagaiak beti balio berdina hartzen du:

$$P(X_2 = x_2) = \sum_{x_1} P(X_2 = x_2, X_1 = x_1) = \sum_{x_1} P(X_2 = x_2|X_1 = x_1)P(X_1 = x_1) = P(X_2 = x_2|X_1 = U)$$

Beraz, $P(X_2 = U) = 0.7$ izango da, eta beste aukera guztiek 0.1eko probabilitatea hartuko dute.

X_3 aldagaiaren kasuan (eta beste aldagai guztien kasuan ere), kontuak apur bat konplikatuagoak dira, aurreko aldagaiak balio posible bat baino gehiago hartzen dute eta:

$$P(X_3 = x_3) = \sum_{x_2} P(X_3 = x_3, X_2 = x_2) = \sum_{x_2} P(X_3 = x_3|X_2 = x_2)P(X_2 = x_2)$$

Ariketa: Aurreko formula erabiliz kalkulatu X_3 aldagaiaren probabilitate marjinalak eta ondoren konparatu emaitzak aurreko atalean simulazio bidez lortutako emaitzekin.

Ondorengo aldagaien probabilitate marginalak era berean lortu daitezke, hau da, behin X_i aldagaiaren probabilitateak kalkulatu ditugunean X_{i+1} aldagaiarenak kalkulatu ditzakegu ondoko formula erabiliz:

$$P(X_{i+1} = x_{i+1}) = \sum_{x_i} P(X_{i+1} = x_{i+1}, X_i = x_i) = \sum_{x_i} P(X_{i+1} = x_{i+1}|X_i = x_i)P(X_i = x_i)$$

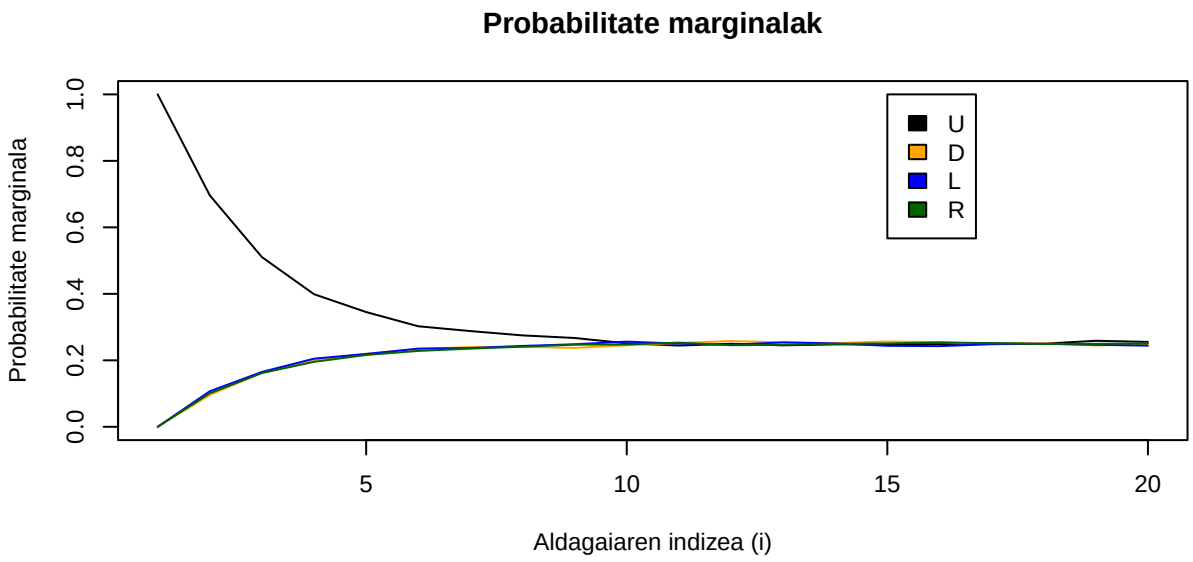
Honek banaketa egonkorrari nola eragiten dion ikusi aurretik, azter ditzagun lehen 50 aldagaien probabilitate marginalak (estimatutakoak), eta irudikatu ditzagun grafiko batean:

```
lortuMarginalak <- function (i, mvkatea.behaketa, egoerak) {
  xi <- factor(mvkatea.behaketa[, i], levels=1:length(egoerak))
  levels(xi) <- egoerak
  return(table(xi) / nrow(mvkatea.behaketa))
}


probabilitate.marginalak <- sapply(1:20, FUN=lortuMarginalak,
                                   mvkatea.behaketa=mvkatea.behaketa, egoerak=egoerak)



plot(1:20, probabilitate.marginalak["U", ], col="black", type="l", ylim=c(0,1),
     main="Probabilitate marginalak",
     xlab="Aldagaiaren indizea (i)",
     ylab="Probabilitate marginala")
lines(1:20, probabilitate.marginalak["D", ], col="orange")
lines(1:20, probabilitate.marginalak["L", ], col="blue")
lines(1:20, probabilitate.marginalak["R", ], col="darkgreen")
legend(15, 1, legend=c("U","D", "L", "R"), fill=c("black", "orange","blue","darkgreen"))
```



Banaketa egonkorra

Aurreko grafikoan ikus dezakegu balio posible bakoitzaren (`U`, `D`, `L`, `R`) probabilitatea aldatu egiten dela X_i bakoitzerako hasieran, baina X_{10} aldagaitik aurrera, probabilitate marginalak egonkortu egiten dira. Hau da, $i > 10$ denean, $P(X_i) \approx P(X_{i+1})$ dela esan dezakegu. Probabilitate banaketa hori Markov katearen *banaketa egonkorra* deitzen da. Gainera, grafikoan begiratuta ikus dezakegu banaketa egonkorra honakoa dela:

$$P(X = U) = P(X = D) = P(X = L) = P(X = R) = 0.25$$

Orain konprobatu dezagun hau teoria erabiliz. Gogoan izan, Markov kate batek banaketa egonkor bakarra baduela bi baldintza hauek betetzen baditu:

- *Ergodikoa*: Edozein egoeratik edozein egoerara pasatzeko probabilitatea ez-nulua da.
- *Aperiodikoa*: Egoera berdinera bueltatzeko ez dago periodizitate zehatz bat.

Gure kasuan, trantsizio matrizea begiratuta hau hala dela argi ikusten da. Gainera, teorian ikusi dugun bezala,

$$\pi_{i+1}^T = \pi_i^T \cdot P,$$

eta hemendik, banaketa egonkorrera iristean, $\pi_i = \pi_{i+1} = \pi_e$ denez, banaketa egonkorrak honako baldintza bete behar du:

$$\pi_e^P = \pi_e^T \cdot P$$

.

Hau ekuazio sistema lineal bat da, era matrizialean idatzita. Ebazteko beste era honetan idatziko dugu:

$$\pi_e^T \cdot P = \pi_e^T \cdot I \rightarrow \pi_e^T \cdot (P - I) = 0^T$$

Sistema linealak Rn `solve` funtzioarekin ebatzi daitezke, baina kontuan izan behar dugu funtzio honek $A \cdot x = b$ moduko sistemak ebazten dituela eta ez $x^T \cdot A = b$.

Ariketa: Gogoratu zer esan nahi zuen ekuazio sistema bat lineala izateak. Betetzen da kasu honetan? Zergatik? Nola moldatu daiteke ekuazio sistema hau 'solve' funtzioarekin ebatzi ahal izateko?

Gure matrizea moldatzeko, transposatuak hartuko ditugu bi aldeetan:

$$(\pi_e^T \cdot (P - I))^T = (0^T)^T \rightarrow (P - I)^T \cdot \pi_e = 0$$

Identitate matrizea `diag` funtzioarekin lortu dezakegu, beraz, prest gaude sistema ebazteko:

```
a.mat <- t(trantsizio.matrizea - diag(4))
b.coef <- c(0,0,0,0)
solve(a=a.mat, b=b.coef)
```

```
## Error in solve.default(a = a.mat, b = b.coef): system is computationally singular: reciprocal conditi
```

Ikus dezakegunez, ez dago soluziorik (soluzio bakarra behintzat), matrizea singularra delako (determinantea 0 da). Honek esan nahi du soluzio kopurua infinitua dela. Baina soluzio horietatik guztietatik bakarrak adierazten du banaketa egonkorra. Hain zuzen, banaketa bat adierazten duenak, hau da, probabilitate guztien batura 1 duena. Ekuazio hori sistemari gehitzen badiogu, soluzio bakarra izango dugu.

Zoritxarrez, `solve` funtzioak ez ditu karratuak ez diren sistemak ebazten eta, hortaz, beste pakete bat erabiliko digu: **matlib**[1].

```
library(matlib)
a.mat.zabaldua <- rbind(a.mat, c(1,1,1,1))
b.coef.zabaldua <- c(b.coef, 1)
showEqn(a.mat.zabaldua, b.coef.zabaldua)
```

```
## -0.3*x1 + 0.1*x2 + 0.1*x3 + 0.1*x4  =  0
##  0.1*x1 - 0.3*x2 + 0.1*x3 + 0.1*x4  =  0
##  0.1*x1 + 0.1*x2 - 0.3*x3 + 0.1*x4  =  0
##  0.1*x1 + 0.1*x2 + 0.1*x3 - 0.3*x4  =  0
##    1*x1   + 1*x2   + 1*x3   + 1*x4  =  1
```

Azkenik, sistema ebazten badugu, banaketa egonkorra lortuko dugu.

```
Solve(a.mat.zabaldua, b.coef.zabaldua)
```

```
## x1        =  0.25
##   x2      =  0.25
##     x3    =  0.25
##       x4  =  0.25
##        0  =     0
```

Ikusten dugunez, banaketa egonkorra uniformea da, hau da lehen grafikoan ikusi dugun berdina. Egiaztatzeko, ikusi behar dugu $\pi_e^T \cdot P = \pi_e^T$ ekuazioa betetzen dela.

```
p.e <- c(0.25,0.25,0.25,0.25)
p.e %*% trantsizio.matrizea
```

[1] Paketea ez badago instalatuta, erabili aurretik instalatu

```
##         U    D    L    R
## [1,] 0.25 0.25 0.25 0.25
```

Esan bezala, behin Markov katea bere banaketa egonkorrera iristen denean, aldagaien banaketa marginalak ez dira denboran zehar aldatuko. Kontuz, honek ez du esan nahi aldagaien balio guztiek probabilitate berdina izango dutenik beti, kasu honetan bezala. Orduan, hasierako balioak kentzen baditugu (lehenengo 10 balioak kasu honetan, katea X_10 aldagaietik aurrera egonkortzen baita), handik aurrerako balioak banaketa egonkorraren laginketa direla esan dezakegu.

Ikus dezakegu zer gertatzen den katearen 10000 pausu lagintzen baditugu (hau da, X_{10000} aldagairarte) eta lehenengo 100 balioak kentzen baditugu:

```
kate.lagina.10000 <- laginduMarkovKatea(lagin.tamaina=10000,
                                        hasierako.egoera="U",
                                        trantsizio.matrizea=trantsizio.matrizea,
                                        egoerak=egoerak)

banaketa.egonkorra.lagina <- kate.lagina.10000[100:10000]
table(banaketa.egonkorra.lagina) / length(banaketa.egonkorra.lagina)
```

```
## banaketa.egonkorra.lagina
##         U         D         L         R
## 0.2518937 0.2576507 0.2488638 0.2415918
```

Ikusten dugun moduan, lagindu dugun banaketa banaketa egonkorraren berdina da (hau da, lau mugimendu motei probabilitate berdina esleitzen die).

Ariketa

Kontsideratu dezagun Markov kate bat, hurrengo trantsizio-matrizea duena:

$$P = \begin{pmatrix} 0.2 & 0.2 & 0.1 & 0.5 \\ 0.1 & 0.3 & 0.5 & 0.1 \\ 0.1 & 0.6 & 0.2 & 0.1 \\ 0.5 & 0.3 & 0.1 & 0.1 \end{pmatrix}$$

Tutorialeko kodea erreferentzia gisa hartuz:

- Kalkula ezazu banaketa egonkorra metodo zehatz bat erabiliz.
- Estima ezazu laginketan oinarritutako metodo bat erabiliz lehendabiziko 15 aldagaien probabilitate marjinalak, eta adieraz itzazu grafikoki hauen konbergentzia ikusteko. Aurretik kalkulatutakoarekin bat egiten du?
- Errepika ezazu aurreko atala hasierako egoera aldatuz. Zein ondorio atera ditzakezu?
- Kalkula ezazu P^{50} eta lortzen duzun emaitza azaldu.