Veri Üretimi

1-0


Dr. Kübra Atalay Kabasakal

Veri Üretimi

- İstatistiğin en önemli konularından birisi **olasılık dağılımlarıdır.**
- R'da veri üretimi yaparken çıktıların ne olduklarını bilmek yerine **çıktıların olma olasılıkları üzerinden veri üretmek** gerekebilir.
- Örneğin, bir sınıftaki öğrencilerin madeni para atışında, paranın yazı ya da tura gelme olasılığı üzerine veri üretimi yapılırken,
  - yazı gelme olasılığının 0.5; tura gelme olasılığının 0.5 olduğu bu durumda **Bernoulli dağılımı** yardımıyla yüzlerce ya da binlerce para atışı için veri üretmek mümkündür.

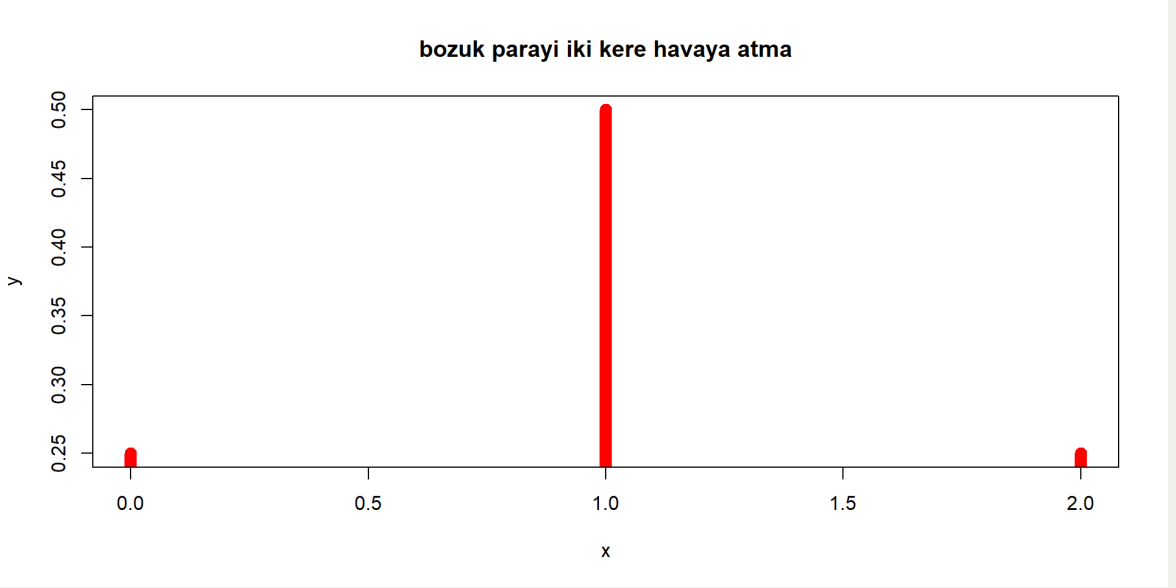
Veri Üretimi

- Paranın iki kere hava atılması
- YY (1/4)
- TT (1/4)
- TY (2/4)

```r
x <- 0:2
y <- dbinom(x,size=2,p=0.5)
y
```

```
[1] 0.25 0.50 0.25
```

```r
plot(x, y ,type="h",col="red", lwd=10,
main="bozuk parayi iki kere havaya atma")
```



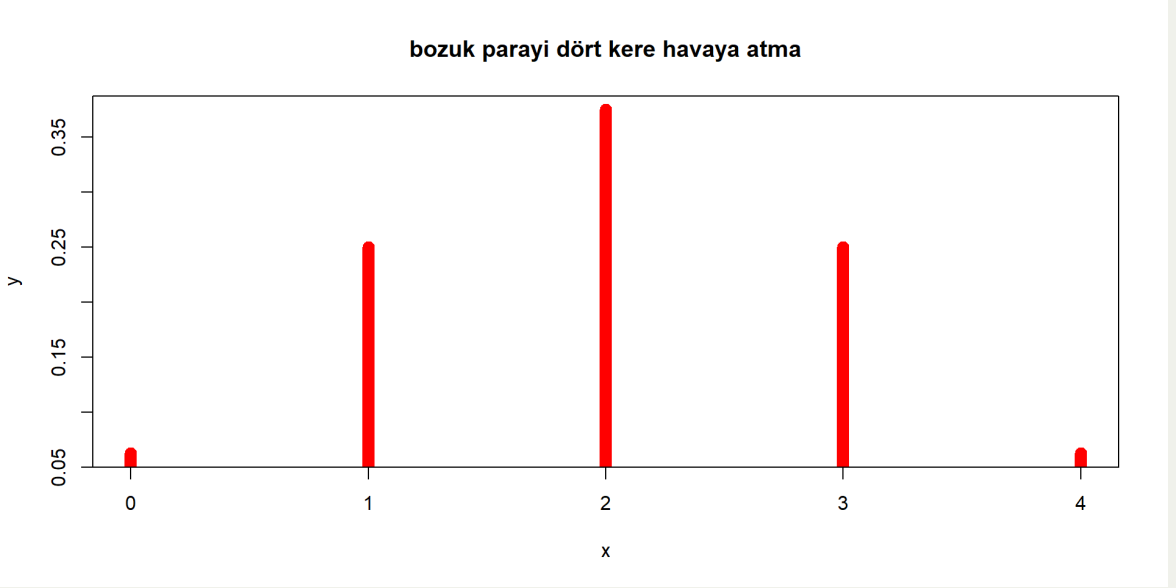
Veri Üretimi

- Paranın dört kez kere hava atılması
- YYYY (1/16)
- TTTT (1/16)
- YTTT (4/16)
- TYYY (4/16)
- YYTT(6/16)

```r
x <- 0:4
y <- dbinom(x,size=4,p=0.5)
y
```

```
[1] 0.0625 0.2500 0.3750 0.2500 0.0625
```

```r
plot(x, y ,type="h",col="red", lwd=10,
main="bozuk parayi dört kere havaya atma")
```



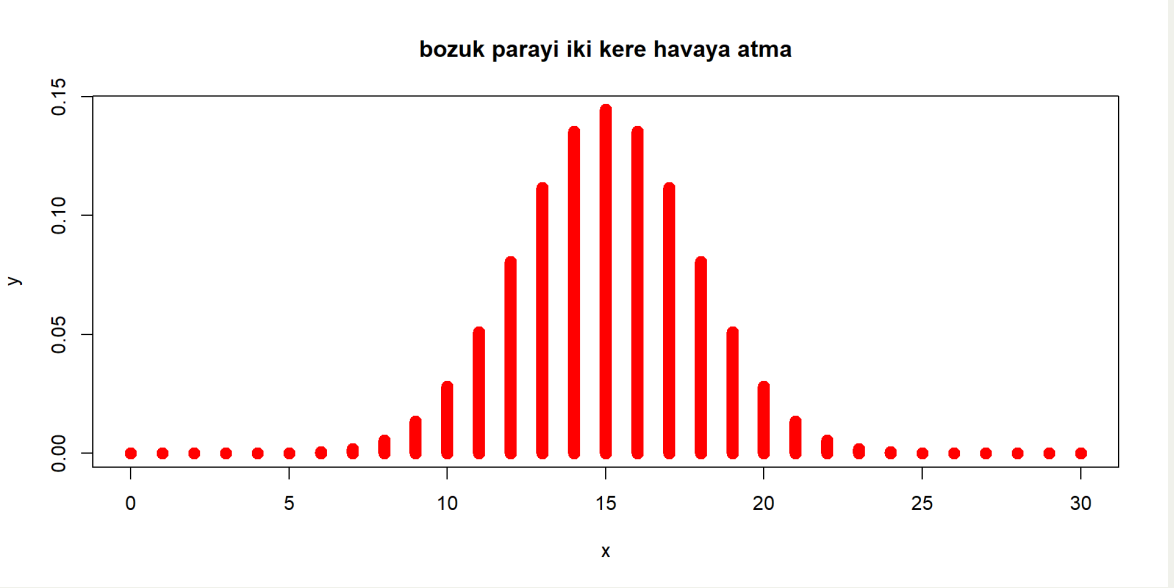
Veri Üretimi

- Paranın 30 kez kere hava atılması

```r
x <- 0:30
y <- dbinom(x,size=30,p=0.5)
y
```

```
 [1] 9.31e-10 2.79e-08 4.05e-07 3.78e-06
2.55e-05 1.33e-04 5.53e-04 1.90e-03
 [9] 5.45e-03 1.33e-02 2.80e-02 5.09e-02
8.06e-02 1.12e-01 1.35e-01 1.44e-01
[17] 1.35e-01 1.12e-01 8.06e-02 5.09e-02
2.80e-02 1.33e-02 5.45e-03 1.90e-03
[25] 5.53e-04 1.33e-04 2.55e-05 3.78e-06
4.05e-07 2.79e-08 9.31e-10
```

```r
plot(x, y ,type="h",col="red", lwd=10,
     main="bozuk parayi iki kere havaya atma")
```



Veri Üretimi

- Normal, log-normal, ki-kare, binom, poisson, uniform, gamma, beta vb. gibi farklı dağılımlar mevcuttur.
- **p**, olasılık (probability) kümülatif dağılım fonksiyonuna ilişkin bilgileri
- **q**, çeyreklik (quantile) kümülatif dağılım fonksiyonunun tersine ilişkin bilgileri
- **d**, yoğunluk (density) yoğunluk fonksiyonuna ilişkin bilgileri
- **r**, rastgele (random) belirlenen dağılımdaki rastgele veriyi

Veri Üretimi

- Normal dağılama uygun veri üretmek için **rnorm()**

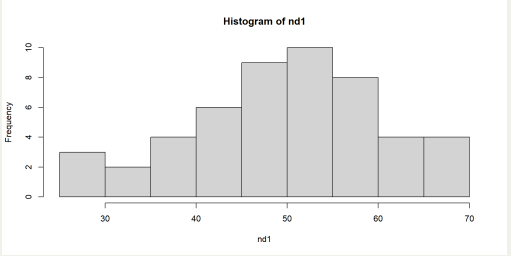
```
1 n=50
2 nd1 <- rnorm(n=n, mean=50, sd=10)
3 mean(nd1)
```

```
[1] 49.7
```

```
1 sd(nd1)
```

```
[1] 10.1
```

```
1 hist(nd1)
```



Veri Üretimi

- Tek biçimli (uniform) dağılıma ilişkin bir veri üretmek için **runif()**

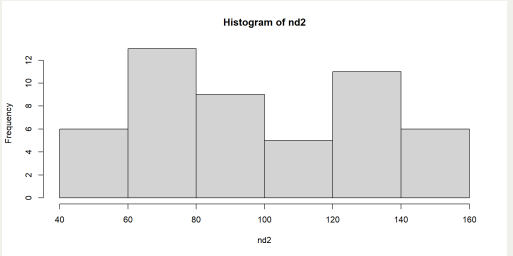
```
1 nd2 <- runif(n=n, min=50, max=150)
2 min(nd2)
```

```
[1] 50.8
```

```
1 sd(nd2)
```

```
[1] 31.1
```

```
1 hist(nd2)
```



İterasyon

- Veri üretimlerinde tek bir örneklemin elde edilmesi **güvenilir olmayabilir.**
- Örneğin, **ortalaması 100**, **standart sapması 20** olan normal dağılıma ilişkin veri üretildiğinde, elde edilen ortalama değer bir örneklem için **örnekleme hatasından** dolayı **100 değerinden göreceli olarak farklı** çıkabilir.

```
set.seed(41)
x <- rnorm(2, mean=100, sd=20)
mean(x)
```

```
[1] 94
```

- veri üretme çalışmalarında **tekrarlamaların yapılması önemlidir.**
- İterasyon, veri üretiminin fonksiyonlar ve döngüler kullanılarak genişletilmesi ve **genellenebilir sonuçlar** elde edilmesi olarak tanımlanabilir.

İterasyon

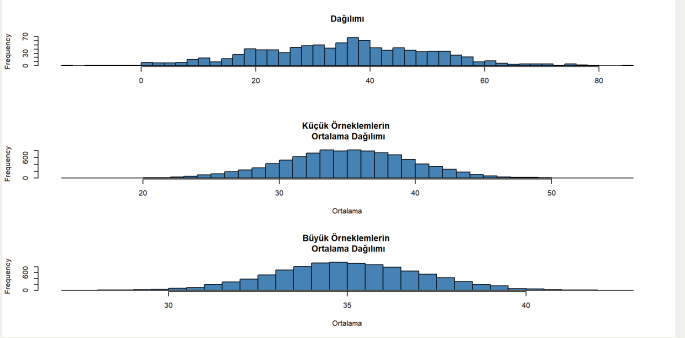
```r
n_k <- 10                    # Küçük örneklem
n_b <- 50                    # Büyük örneklem

tekrar <- 10000               # Tekrar sayısı
kucuk_orn <- numeric(tekrar) # Küçük örneklem ort.
buyuk_orn <- numeric(tekrar) # Büyük örneklem ort.

orneklem.normal <- rnorm(n = 1000, mean = 35, sd = 15)#Sekil

for (i in 1:tekrar) {
  kucuk_orn[i] <- mean(rnorm(n = n_k,mean =  35, sd = 15)  )
  buyuk_orn[i] <- mean(rnorm(n = n_b,mean =  35, sd = 15)  )

}
```



İterasyon

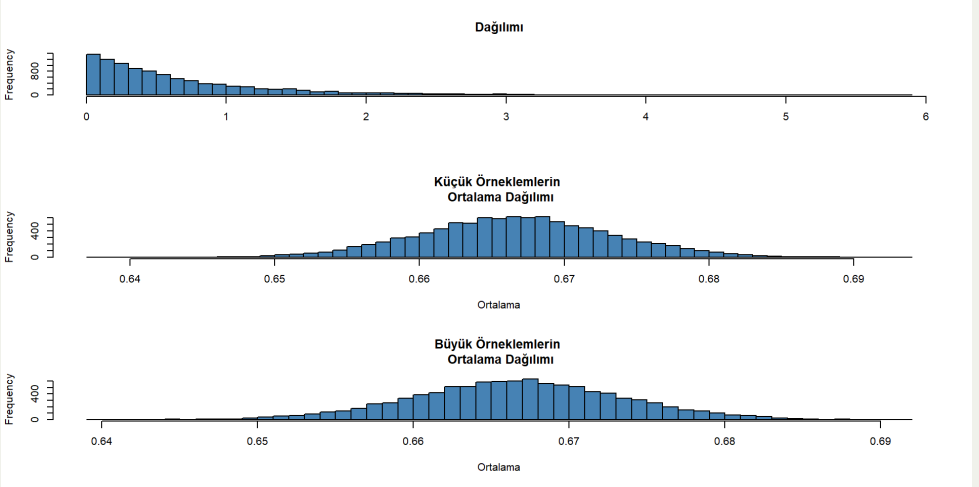
```r
n_k <- 10                    # Küçük örneklem
n_b <- 50                    # Büyük örneklem

tekrar <- 10000               # Tekrar sayısı
kucuk_orn <- numeric(tekrar) # Küçük örneklem ort.
buyuk_orn <- numeric(tekrar) # Büyük örneklem ort.

orneklem.carpik <- rexp(n = 10000, rate = 1.5)#Sek

for (i in 1:tekrar) {
  kucuk_orn[i] <- mean(rexp(n = 10000, rate = 1.5)
  buyuk_orn[i] <- mean(rexp(n = 10000, rate = 1.5)

}
```



İterasyonların Kaydedilmesi ve Tekrar Okutulması

- Özellikle büyük verilerle çalışırken, **iterasyonlar uzun zamanlar alabilir.**
- Her seferinde **iterasyonların tekrarlanması** yerine iterasyon sonuçlarının **kaydedilip** yeniden okutularak analizlere devam edilmesi gerekebilir.
- **kucuk_orn** ve **buyuk_orn** veri setlerini **iki ayrı klasöre yazdırnız.**

İterasyonların Kaydedilmesi ve Tekrar Okutulması

- Daha önce oluşturduğunuz **kucuk_orn** ve **buyuk_orn** veri setlerini iki ayrı klasörden okutup R nesnesi olarak yazdırınız.

Psikometri Alanında Veri Üretimi

- Ölçme ve değerlendirme alanında gerçekleştirilen bazı çalışmalarda, çalışmanın amacı doğrultusunda, veri setlerinin üretilmesi gerekebilir.
- Bu çalışmalarda veri setleri çoğunlukla **Monte Carlo (MC) teknikleri** kullanılarak üretilir.
- **MC** yaklaşımında **matematiksel bir model** kullanılarak bilgisayar yazılımı aracılığıyla **rastgele örneklemler elde edilebilir.**

Psikometri Alanında Veri Üretimi

- **MC** çalışmaları ile parametre değerlerini belirlemek, çalışmada **bazı faktörleri (madde sayısı gibi) sabit tutup bazı faktörleri (örneklem büyüklüğü gibi) manipüle ederek** değişkenlerin etkilerini incelemek mümkündür.
- Böylece belli özellikleri önceden bilinen veri setleri, ele alınan koşullar altında, **yöntemlerin veya modellerin değerlendirilmesine** ve karşılatırılmasına olanak sağlar.
- Ancak sonuçların kullanışlılığı için MC çalışmalarında modellenen koşulların **gerçek uygulamalara** ne kadar yakın olduğuna, replikasyon sayısına, vb. dikkat edilmesi gerekir (Harwell & Others, 1996).

R Paketleri ile Veri Üretimi

- Revelle (2019) tarafından üretilen **psych** ile
- YEM, MTK, ve KTK ya uygun veriler üretilebilir.
- Partchev (2017) tarafından yazılan **irtoys** paketinde bulunan
  - 1, 2 ve 3 parametreli lojistik modele uygun veriler üretilebilir.
- Chalmers (2019) tarafından hazırlanan **mirt** paketiyle hem tamamlayıcı hem de tamamlayıcı olmayan ÇBMTK verisi üretilebilir.
- ** …**

R Paketleri ile Veri Üretimi

- hangi model?
- kaç madde?
- madde ve yetenek parametreleri ne olacak?

3PL modele dayali **8 maddelik** veri veri üretilmesi İlk aşama madde parametrelerinin belirlenmesi

```r
set.seed(41)
madde <- 8
maddepar <- cbind(
rnorm(madde, mean = 0, sd = 0.75)*1.702,     #a
rnorm(madde, mean = 0.30, sd = 0.51)*1.702, #b
rnorm(madde, mean = 0.16, sd = 0.05))        #c
```

```r
maddepar
```

```
       [,1]      [,2]  [,3]
[1,] -1.014  1.379159 0.193
[2,]  0.252  2.409834 0.204
[3,]  1.279 -0.539118 0.170
[4,]  1.645  0.000801 0.274
[5,]  1.156  1.427334 0.115
[6,]  0.630  0.235798 0.267
[7,]  0.765  0.463257 0.102
[8,] -2.016  0.796831 0.158
```

R Paketleri ile Veri Üretimi

- İkinci aşama yetenek parametrelerinin belirlenmesi
- 1000 kişilik normal dağılıma dayalı veri üretimi

```r
set.seed(41)
birey <- 1000
yetenek <- rnorm(birey, mean = 0, sd = 1)
yetenek[1:10]
```

```
 [1] -0.794  0.197  1.002  1.289  0.906  0.494  0.599 -1.580  1.001  2.188
```

R Paketleri ile Veri Üretimi

- **irtoys** paketi **sim** fonksiyonu ile veri üretilmesi

```r
veri <- irtoys::sim(ip = maddepar, x = yetenek)
  colnames(veri) <- paste0("madde", 1:madde)
  head(veri)
```

```
     madde1 madde2 madde3 madde4 madde5 madde6 madde7 madde8
[1,]      0      1      1      0      0      0      1      1
[2,]      1      0      1      1      0      1      0      0
[3,]      1      1      1      1      0      1      0      1
[4,]      1      1      0      1      0      1      1      0
[5,]      1      1      0      1      0      1      1      1
[6,]      1      0      1      1      0      1      0      0
```

R Paketleri ile Veri Üretimi

- **irtoys** paketi **sim** fonksiyonu ile veri üretilmesi

```
1 library(irtoys)
2 ip = maddepar
3 x = yetenek
4 sim
```

```
function (ip, x = NULL)
{
    if (is.list(ip))
        ip = ip$est
    i = irf(ip = ip, x = x)
    d = dim(i$f)
    u = runif(d[1] * d[2])
    dim(u) = d
    return(ifelse(i$f > u, 1, 0))
}
<bytecode: 0x000001e16a595500>
<environment: namespace:irtoys>
```

R Paketleri ile Veri Üretimi

- Üretilen verinin parametrelerinin kestirimi

```r
library(mirt)
model3PL <- mirt(veri, # cevap matrisi
                 1,  # tek boyutlu model
                 itemtype = "3PL", # 3PL
                 verbose = FALSE,
                 technical = list(NCYCLES = 1000,
                 message = FALSE))

mirt::coef(model3PL, IRTpars = TRUE, simplify = TRUE)$items
```

```
            a      b       g u
madde1 -1.096  1.541 0.00484 1
madde2  0.308  2.155 0.21580 1
madde3  1.093 -0.932 0.02929 1
madde4  1.353  0.071 0.24129 1
madde5  0.975  1.547 0.08119 1
madde6  1.007  1.079 0.47165 1
madde7  0.697  0.711 0.08023 1
madde8 -1.890  0.970 0.00211 1
```

R Paketleri ile Veri Üretimi

- Veri üretimini tekrarlamak için fonksiyon yazmamız lazım.
- Fonksiyon1
  - madde sayısı ve birey sayısına bağlı olarak
    - madde parametresi üretsin
    - yetenek parametresi üretsin
    - madde cevapları üretsin
    - ciktida madde parametrelerini, yetenek parametrelerini ve cevap matrisini tutsun

```
1 veri_uretimi <- function(madde, birey ){
2   .....
3 }
```

R Paketleri ile Veri Üretimi

```r
veri_1 <- veri_uretimi(madde = 8, birey = 1000, seed = 666)
head(veri_1$yetenek)
```

```
[1]  0.0306 -1.4823 -1.1266 -1.7638 -1.0626 -1.3430
```

```r
head(veri_1$maddepar,3)
```

```
     [,1]   [,2]  [,3]
[1,] 2.24 -1.198 0.203
[2,] 2.78  0.321 0.177
[3,] 1.77  2.224 0.131
```

```r
head(veri_1$cevaplar,3)
```

```
     madde1 madde2 madde3 madde4 madde5 madde6 madde7 madde8
[1,]      1      1      0      1      1      1      0      0
[2,]      0      0      0      0      1      0      1      0
[3,]      1      0      0      1      0      1      0      1
```

Paramtere kestirimi

Fonksiyon2 - **veri_uretimi** fonksiyonu ile üretilen veri girdi olsun - hangi modele göre kestirim yapacağı argüman olsun. (2PL ya da 3PL) - Cikti olarak kestirilen parametre matrisi verilsin

```
kestirilen_par <- function(veri, par=3){

  ....
}
```

Kestirilen değerler ve Gerçek Değerler

- **RMSE** her bir madde için her bir replikasyonda kestirilen parametre değeri ile gerçek parametre değeri arasındaki farkın karesinin ortalamasının karekökü̈dür

\[RMSE(\tau) = \sqrt\frac{\sum_{r}^{R} (\hat{\tau}-\tau)^2}{R})\]

Kestirilen değerler ve Gerçek Değerler

- **BIAS** kestirime ilişkin sistematik hatayı ifade eden yanlılık her bir madde için her bir replikasyonda kestirilen parametre değerinin ortalaması ile gerçek parametre değeri arasındaki farktır ve aşağıdaki formül ile hesaplanır:

$$BIAS(\tau) = \frac{\sum_{r}^{R}\hat{\tau}}{R} - \tau$$

Kestirilen değerler ve Gerçek Değerler

- **SE** kestirime ilişkin rastgele hatayı ifade eden standart hata her bir madde için her bir replikasyonda kestirilen parametre değeri ile ortalama parametre kestirimi arasındaki farkın karesinin ortalamasının karekköüdür. Diğer bir ifadeyle replikasyonlarda kestirilen parametre değerlerinin standart sapmasıdır ve aşağıdaki formül ile hesaplanır:

\[SE(\tau) = \sqrt\frac{\sum_{r}^{R} (\hat{\tau}- \overline{\hat{\tau}})^2}{R})\] - Yanlılığın karesi ile standart hatanın karesinin toplamı RMSE'nin karesinin toplamına eşittir. RMSE, yanlılık ve standart hata arasındaki ilişki aşağıdaki eşitlik ile gösterilebilir:

\[RMSE^2= Bias^2 +SE^2\]

RMSE BIAS ve SE hesapalama

```r
hata <- function(kestirim, gercek) {

}
```

```
  parametreler    bias  rmse korelasyon
1            a  0.7953 1.861      0.814
2            b  0.0864 0.309      0.978
3            c -0.0148 0.133      0.451
```

R Paketleri ile Veri Üretimi

Fonksiyon yazarken nelere dikkat etmeliyiz?

```r
temp2 <- veri_uretimi(maddesay = 10, bireysay = 1000)
```

```
Error in set.seed(seed): argument "seed" is missing, with no default
```

```
atanan seed =  5593
[1] "Madde parametresi uretme"
[1] "Üretilen veri"
[1] "Cıktıların birleştirilmesi"
```

Veri Üretiminde Tekrar

- Tekrar etmek amacıyla **döngülerden** ya da **apply** fonksiyonlarından yararlanalılabilir.

```
replicate(tekrar sayisi, fonksiyon)
```

- replicate fonksiyonu kullanıldığında oluşan çıktı liste biçiminde olacaktır.

```
tekrarsayisi=5L
x <- replicate(tekrarsayisi, veri_uretimi(maddesay = 10, bireysay = 1000))
```

```
atanan seed =  7116
[1] "Madde parametresi uretme"
[1] "Üretilen veri"
[1] "Cıktıların birleştirilmesi"
atanan seed =  727
[1] "Madde parametresi uretme"
[1] "Üretilen veri"
[1] "Cıktıların birleştirilmesi"
atanan seed =  6970
[1] "Madde parametresi uretme"
[1] "Üretilen veri"
[1] "Cıktıların birleştirilmesi"
atanan seed =  4875
[1] "Madde parametresi uretme"
[1] "Üretilen veri"
[1] "Cıktıların birleştirilmesi"
atanan seed =  789
[1] "Madde parametresi uretme"
[1] "Üretilen veri"
[1] "Cıktıların birleştirilmesi"
```

Veri Üretiminde Tekrar

- Oluşan nesnenin sadece maddepar bileşenin almak için ciktilar liste biçimindedir

```
lapply(1L:tekrarsayisi, function(i) x[,i][1])
```

```
[[1]]
[[1]]$maddepar
       [,1]   [,2]   [,3]
 [1,] 2.63 -0.570 0.0752
 [2,] 1.53  0.065 0.1859
 [3,] 1.27  1.798 0.0994
 [4,] 1.93  2.325 0.1950
 [5,] 2.36  0.461 0.1757
 [6,] 1.81  0.827 0.1840
 [7,] 2.26  1.668 0.1773
 [8,] 2.21  0.170 0.1471
 [9,] 2.46  0.392 0.1278
[10,] 2.23  0.370 0.1622


[[2]]
```

Veri Üretiminde Tekrar

- Ciktilarin liste olmaması için sapply de kullanılabilir.

```
1 sapply(1L:tekrarsayisi, function(i) x[,i][1])
```

```
$maddepar
      [,1]   [,2]   [,3]
 [1,] 2.63 -0.570 0.0752
 [2,] 1.53  0.065 0.1859
 [3,] 1.27  1.798 0.0994
 [4,] 1.93  2.325 0.1950
 [5,] 2.36  0.461 0.1757
 [6,] 1.81  0.827 0.1840
 [7,] 2.26  1.668 0.1773
 [8,] 2.21  0.170 0.1471
 [9,] 2.46  0.392 0.1278
[10,] 2.23  0.370 0.1622

$maddepar
      [,1]   [,2]   [,3]
```

Veri Üretiminde Tekrar

```
tekrar = 4
seed = sample.int(10000, 100)
maddesay = 10 # 10, 15, 20, or 25
bireysay = 1000 # 250, 500, 750, or 1000
```

```
library("doParallel")
detectCores()
```

```
[1] 12
```

```
cl <- makeCluster(4) # en fazla n-2
registerDoParallel(cl)
```

foreach

- Döngüler yavas olabilir,
- apply ailesi ciktilari kullanışlı olmayabilir.

```
1  foreach(i=1:4) %dopar% sqrt(i)
```

```
[[1]]
[1] 1

[[2]]
[1] 1.41

[[3]]
[1] 1.73

[[4]]
[1] 2
```

foreach

```
1 #coklu arguman
2 foreach(i=1:4, j=1:4) %do%
3     sqrt(i+j)
```

```
[[1]]
[1] 1.41

[[2]]
[1] 2

[[3]]
[1] 2.45

[[4]]
[1] 2.83
```

```
1 stopCluster(cl) # cekirdek atama işini bitirir.
```

foreach

```
1   for(i in 1:5) {
2     sum(rnorm(1e6))
3     }
```

```
1   foreach(i=1:5) %dopar% {
2     sum(rnorm(1e6))
3     }
```

foreach

```
cl <- makeCluster(4)
registerDoParallel(cl)

tekrar = 4
seed = sample.int(10000, 100)
maddesay = 10 # 10, 15, 20, or 25
bireysay = 1000 # 250, 500, 750, or 1000

simulasyon <- foreach(i=1:4,
              .packages = c("mirt", "doParallel"),
              .combine = rbind) %dopar% {
# Adım 1 madde parametrelerini ve veri setini üretme
adim1 <- veri_uretimi(maddesay =maddesay, bireysay =bireysay, seed=seed[i])
# Adım 2 üretilen veri seti üzerinden ketsirim yapma
adim2 <- kestirilen_par(adim1$cevaplar)
# adim 3 raporlama
hata(adim2, adim1$maddepar)
}
```

foreach

Cikti düzenleme

```
1 simulasyon
```

	parametreler	bias	rmse	korelasyon
1	a	0.09127	0.3951	0.277
2	b	-0.06086	0.0955	0.998
3	c	-0.03573	0.0698	0.847
4	a	0.14192	0.3825	0.770
5	b	0.00934	0.1453	0.979
6	c	-0.00052	0.0750	0.566
7	a	-0.22460	0.6986	0.688
8	b	0.05469	0.1792	0.991
9	c	-0.03235	0.0736	0.608
10	a	0.11741	0.5982	0.382
11	b	0.01070	0.1925	0.978
12	c	-0.00743	0.0832	0.613

foreach

Cikti düzenleme

```
1 library("dplyr")
2 simulasyon %>%
```

```
1 simulasyon_v1
```

```
  parametreler   bias  rmse korelasyon maddesay bireysay
1            a  0.032 0.519      0.529       10     1000
2            b  0.003 0.153      0.986       10     1000
3            c -0.019 0.075      0.658       10     1000
```

foreach

```r
tekrar = 1;seed = sample.int(10000, 100);maddesay = c(10, 20 ,40);bireysay = c(250, 500, 1000)

# kumeler
cl <- makeCluster(6);registerDoParallel(cl)

# İc ice foreachler
sonuc <- foreach(i=1:tekrar,
                      .packages = c("mirt", "doParallel", "dplyr"),
                      .combine = rbind) %:%
  foreach(j=maddesay,
          .packages = c("mirt", "doParallel", "dplyr"),
          .combine = rbind)  %:%
  foreach(k=bireysay,
          .packages = c("mirt", "doParallel", "dplyr"),
          .combine = rbind)    %dopar% {
            adim1 <- veri_uretimi(maddesay=j, bireysay=k, seed=seed[i])
            adim2 <- kestirilen_par(adim1$cevaplar)
            adim3 <- hata(adim2, adim1$maddepar)
            adim3 %>%
              group_by(parametreler) %>%
              summarise(bias = round(mean(bias),3),
                        rmse = round(mean(rmse),3),
                        korelasyon = round(mean(korelasyon),3)) %>%
              mutate(maddesay = j, bireysay = k,
              ) %>%              as.data.frame()
```

foreach

```
write.table(adim3, "results.csv",
            sep = ",",
            col.names = FALSE,
           row.names = FALSE,
           append = TRUE) # Sonucları yazmak icin acik tutar
```

foreach

```
iterations = 4; seed = sample.int(10000, 100)
maddesay = 10 ;bireysay = 1000 #250, 500, 750, or 1000
library("doSNOW")
cl <- makeCluster(4);registerDoSNOW(cl)

pb <- txtProgressBar(max = iterations, style = 3) # Initiate progress bar
```

```
  |
  |                                                                      |
0%
```

```
progress <- function(n) {setTxtProgressBar(pb, n)}
opts <- list(progress = progress)
sonuc <- foreach(i=1:iterations,
                        .packages = c("mirt", "doSNOW"),
                        .options.snow = opts, # see the additional line for d
                        .combine = rbind) %dopar% {
                          # Generate item parameters and data
                          adim1 <- veri_uretimi(maddesay  , bireysay, seed=se
                          # Estimate item parameters
                          adim2 <- kestirilen_par(adim1$cevaplar)
                          # Summarize results
```

```
12                          hata(adim2, adim1$maddepar)
13                        }
```

```
  |
  |==================                                                    |
25%
  |
  |====================================                                  |
50%
  |
  |======================================================                |
75%
  |
  |======================================================================|
100%
```

```
1 close(pb) # kapat progress bar
```

```
1 stopCluster(cl)
```

Bitti