# 「データ解析」（下平英寿）　講義資料４

# 推定量のバイアスとバラツキ

- 目標：
  1. 「講義資料３」で学んだ統計量を復習する：中心の統計量（平均，メディアン，刈り込み平均，t-分布の位置パラメータ m の最尤推定）．バラツキの統計量 (標準偏差，四分位偏差，ＭＡＤ，t-分布のスケールパラメータ s の最尤推定)
  2. 擬似乱数を用いたシミュレーションによってデータを人工的に多数生成して，統計量の性質を調べる．つまり推定量としての誤差（バイアスとバラツキ）を調べる．
  3. バイアスやバラツキをデータから推定する手法（理論値，ブートストラップ法）について学ぶ．

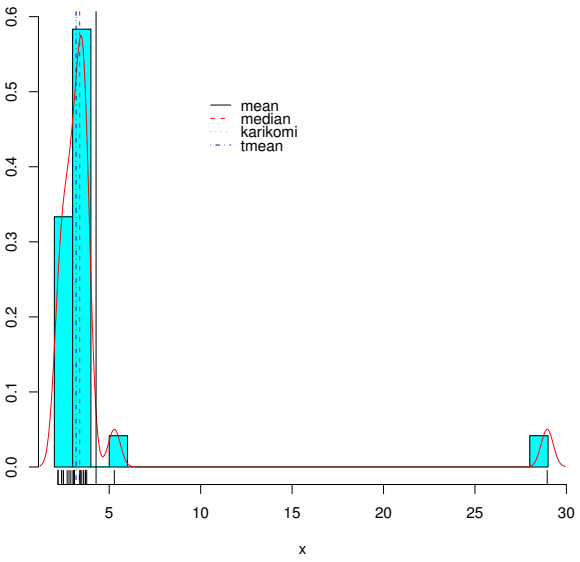
## 1　統計量の復習

- MASS ライブラリの chem データセットに大して，統計量（中心４個，バラツキ４個，バラツキの２乗が４個）を計算する．

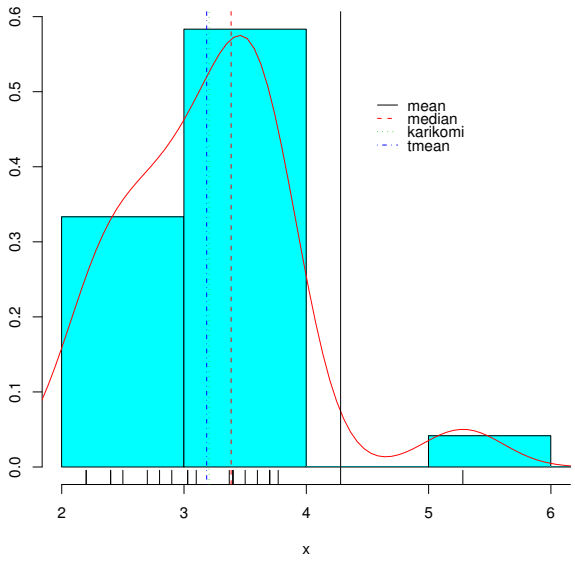
```
# run0042.R
# 中心とバラツキの統計量の復習
library(MASS)
tsdcorrection <- function(m) { # t 分布の補正項の計算
  dlik <- function(x) m/(1+m) + (pnorm(x)-1)/(dnorm(x)/x)
  f <- uniroot(dlik,c(0,sqrt(m)))
  f$root/sqrt(m)
}
mymeansd <- function(x,trim=0.1,df=5) { # 各種統計量の計算
  m1 <- mean(x) # 平均
  m2 <- median(x) # メディアン
  m3 <- mean(x,trim=trim) # 刈り込み平均 (刈り込み率=trim)
  f4 <- fitdistr(x,"t",df=df) # t-分布の最尤推定 (自由度=df)
  m4 <- f4$estimate[[1]] # t-分布の m
  s1 <- sqrt(var(x)) # 標準偏差
  s2 <- IQR(x)/1.3489795 # 四分位偏差 (Interquartile Range) を補正する
  s3 <- mad(x) # MAD（補正済み）
  s4 <- f4$estimate[[2]]/tsdcorrection(df) # t-分布の s
  list(mean=m1,median=m2,karikomi=m3,tmean=m4, # 中心
       sd=s1,iqr=s2,mad=s3,tsd=s4, # バラツキ
       sd2=s1^2,iqr2=s2^2,mad2=s3^2,tsd2=s4^2 ) # バラツキの２乗
}
```

```
# run0043.R
# 統計量のデータへの適用
drawhist <- function(x,nbins,xlim=range(x),t=NULL,leg=NULL) {
  truehist(x,xlim=xlim,nbins=nbins) # ヒストグラム (nbins 分割)
  rug(x) # 「ラグ」をプロット (下部にデータを示す線分)
  lines(density(x),col=2) # 密度関数をカーネル法で推定しプロット
  if(!is.null(t)) {
    abline(v=t$mean,col=1,lty=1)
    abline(v=t$median,col=2,lty=2)
    abline(v=t$karikomi,col=3,lty=3)
    abline(v=t$tmean,col=4,lty=4)
  }
  if(!is.null(leg))
    legend(leg[1],leg[2],
           c("mean","median","karikomi","tmean"),col=1:4,lty=1:4,bty="n")
}
x <- chem # chem データ
t <- mymeansd(x) # 統計量の計算
print(unlist(t)) # 表示
drawhist(x,20,t=t,leg=c(10,0.5))
dev.copy2eps(file="run0043-h1.eps")
drawhist(x,20,xlim=c(2,6),t=t,leg=c(4.5,0.5))
dev.copy2eps(file="run0043-h2.eps")
```

```
> source("run0042.R")
> source("run0043.R")
      mean     median   karikomi      tmean         sd        iqr        mad
 4.2804167  3.3850000  3.2050000  3.1853242  5.2973960  0.6857035  0.5263230
       tsd        sd2       iqr2       mad2       tsd2
 0.7496211 28.0624042  0.4701893  0.2770159  0.5619318
```




run0043-h1




run0043-h2

## 2 シミュレーション

- これまで説明してきた統計量の性質を調べる．
- 統計量は期待値や標準偏差の推定量である．
- 擬似乱数によってデータセットを人工的に多数個生成して，各データセットに統計量を適用する．
- 統計量（＝推定量）の平均，標準偏差，RMSE を調べる．
- 推定量を $T$，その真値を $t_0$ とする．たとえばデータ $x_1, \ldots, x_n$ の平均 $\bar{x}$ が $T$，確率変数 $X$ の期待値が $t_0$ である．
- $T$ の平均は $E(T)$，標準偏差は $\sqrt{V(T)}$ である．$E(T)$ と $t_0$ の差が小さいほうが良い．もし $E(T) = t_0$ ならば，推定量 $T$ は不偏という．また分散 $V(T)$ も小さいほうが良い．
- 誤差 (error) $T - t_0$ のＲＭＳ (Root Mean Square)，すなわち RMSE は，$T$ の $t_0$ からの平均的なズレを表す量．平方平均二乗誤差

$$\mathrm{RMSE} = \sqrt{E((T - t_0)^2)}$$

（説明）

$$\begin{aligned} E((T - t_0)^2) &= E(((T - E(T)) + (E(T) - t_0))^2) \\ &= E((T - E(T))^2) + 2E(T - E(T))(E(T) - t_0) + (E(T) - t_0)^2 \\ &= V(T) + (E(T) - t_0)^2 \end{aligned}$$

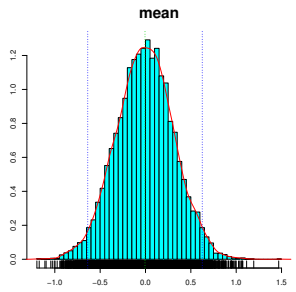
つまり「RMSE の２乗」＝「$T$ の分散」＋「$T$ のバイアスの２乗」．

## 2.1 その１

- ハズレ値なし．サンプルサイズ＝小
- データ $x = (x_1, \ldots, x_n)$，各要素は独立に平均０，分散１の正規分布 $x_i \sim N(0,1)$ に従うとする．
- データサイズ $n = 10$
- シミュレーション回数 $b = 10000$

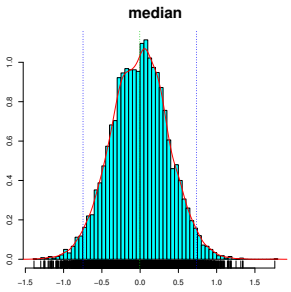
```
# run0037.R
# シミュレーション：データセットの生成（n=10, b=10000）
source("run0042.R")
myrnormb <- function(n,mean=0,sd=1,b=100) # n=データサイズ,
  matrix(rnorm(n*b,mean=mean,sd=sd),n,b) # b=シミュレーション回数
mydrawhist <- function(x,title="") {
  truehist(x,xlab="") # ヒストグラム
  rug(x) # ラグ
  lines(density(x),col=2)
  m <- mean(x); s <- sd(x) # 平均と標準偏差
  abline(v=m,col=3,lty=3) # 平均
  abline(v=m-2*s,col=4,lty=3) # 平均-2*標準偏差
  abline(v=m+2*s,col=4,lty=3) # 平均+2*標準偏差
  title(main=title,cex.sub=2,cex.main=2,
        sub=paste("mean=",signif(m,5),", sd=",signif(s,5)))
}
mycalcrms <- function(x) { # RMS等の計算
  x0 <- x[1]; x <- x[-1]  # 最初の１個は真値，残りがシミュレーション値
  m <- mean(x); s <- sd(x); rms <- sqrt(mean((x-x0)^2))
  list(mean=m,sd=s,rms=rms) # 平均，標準偏差，RMS
}
simx <- myrnormb(10,b=10000) # n=10, b=10000
t0 <- c(rep(0,4),rep(1,4),rep(1,4)) # 統計量の真値（平均=0, 標準偏差=1, 分散=1）
simt <- apply(simx,2,function(x) unlist(mymeansd(x))) # 統計量を繰り返し適用
print(simt[,1:5]) # 最初の５セットだけ表示してみる
sims <- apply(cbind(t0,simt),1,
              function(x) unlist(mycalcrms(x))) # 各統計量のmean,sd,rms
print(sims) # 表示
for(i in rownames(simt)) {
  mydrawhist(simt[i,],i)
  dev.copy2eps(file=paste("run0037-",i,".eps",sep=""))
}
```

```
> source("run0037.R")
              [,1]      [,2]         [,3]       [,4]        [,5]
mean     0.2808414 0.2591123 -0.053392635 -0.5492171  0.01545792
median   0.2838463 0.3463480  0.161148121 -0.3326795 -0.14101321
karikomi 0.2479640 0.2599638  0.008254481 -0.5319985  0.03282927
tmean    0.2602826 0.2881586  0.074402046 -0.5140467  0.02318557
sd       0.5382827 0.6876270  1.048607409  1.1939441  1.23777723
iqr      0.5297210 0.7698082  0.893801258  1.4749687  1.29557598
mad      0.5337794 0.8035744  0.919865965  1.5528675  1.65203348
tsd      0.5296125 0.7053568  1.015799691  1.2516894  1.27516332
sd2      0.2897483 0.4728309  1.099577499  1.4255026  1.53209248
iqr2     0.2806043 0.5926047  0.798880688  2.1755326  1.67851712
mad2     0.2849205 0.6457318  0.846153394  2.4113975  2.72921462
tsd2     0.2804894 0.4975282  1.031849012  1.5667263  1.62604150
              mean      median    karikomi       tmean        sd       iqr
mean -0.00551989 -0.00494938 -0.00504308 -0.00520544 0.9704863 0.8686196
sd    0.31553400  0.37060023  0.32250944  0.32666835 0.2329940 0.3126584
rms   0.31556651  0.37061475  0.32253274  0.32669349 0.2348443 0.3391258
           mad       tsd       sd2      iqr2      mad2      tsd2
mean 0.9123029 0.9336213 0.9961244 0.8522456 0.9438892 0.9268779
sd   0.3340714 0.2350205 0.4724127 0.6094750 0.6861952 0.4627812
rms  0.3453742 0.2442033 0.4724049 0.6270997 0.6884513 0.4684996
> sims0037 <- sims
```



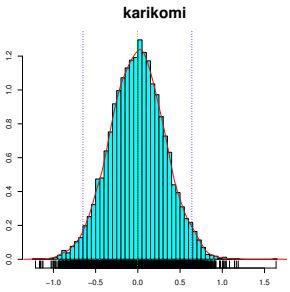

mean= −0.0055199 , sd= 0.31553

run0037-mean



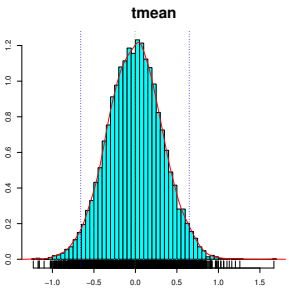

mean= −0.0049494 , sd= 0.3706

run0037-median



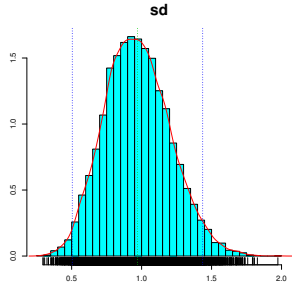

mean= −0.0050431 , sd= 0.32251

run0037-karikomi



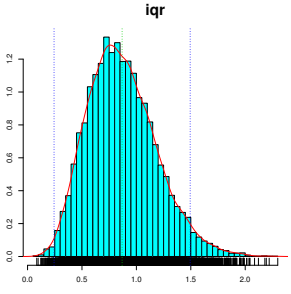

mean= −0.0052054 , sd= 0.32667

run0037-tmean



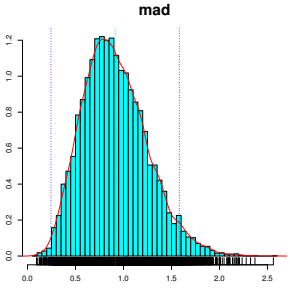

mean= 0.97049 , sd= 0.23299

run0037-sd



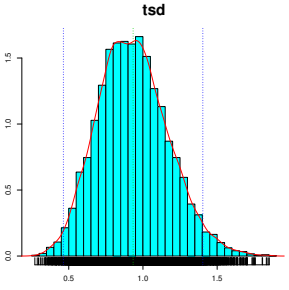

mean= 0.86862 , sd= 0.31266

run0037-iqr



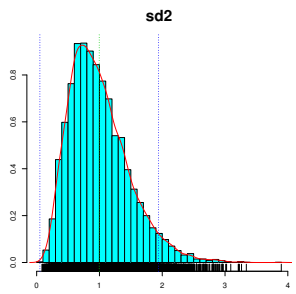

mean= 0.9123 , sd= 0.33407

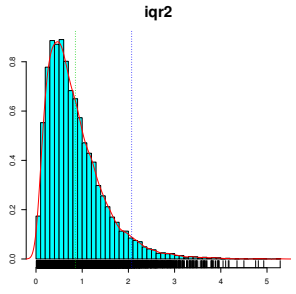
run0037-mad

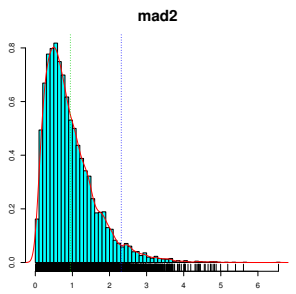


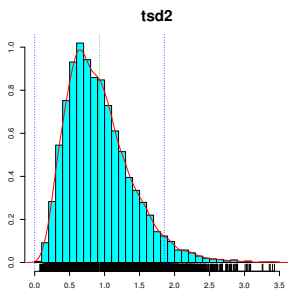

mean= 0.93362 , sd= 0.23502

run0037-tsd










- データは正規分布 (期待値 $\mu$，分散 $\sigma^2$)
- mean は期待値 $\mu$ の不偏推定量．sd2 は分散 $\sigma^2$ である．(mean の行を参照せよ.）なお，sd は標準偏差 $\sigma$ の不偏推定量ではないことに注意.
- mean, median, karikomi, tmean の中で推定量のバラツキ (sd の行) を比べると，mean がバラツキを最小にする．mean はバイアスもゼロである．ここでは分布が $\mu = 0$ の周りで対称であり統計量も対称なので，mean, median, karikomi, tmean すべてバイアスはゼロである．結果として mean が RMSE を最小にする（rms の行を参照）．各行の間には

$$\mathtt{rms}^2 = (\mathtt{mean} - t_0)^2 + \frac{9999}{10000}\mathtt{sd}^2$$

  の関係がある．$\mu$ に関しては $t_0 = 0$ である．$\sigma$ と $\sigma^2$ に関しては $t_0 = 1$ である．sd の計算では分母が 9999 であるので，上式の第 2 項では 9999/10000 をかけて調整している．
- sd2,iqr2, mad2, tsd2 の中で推定量のバラツキ (sd の行) を比べると，tsd2 がバラツキ最小．sd2 も同じくらい小さい．これに対して，iqr2 と mad2 はバラツキが多きい.sd2 はバイアスもゼロだが tsd2 はバイアスが多少あるので，結果として sd2 が RMSE を最小にする．

## 2.2 その 2

- ハズレ値なし．サンプルサイズ＝大
- データ $x = (x_1, \ldots, x_n)$，各要素は独立に平均 0，分散 1 の正規分布 $x_i \sim N(0, 1)$ に従うとする．
- データサイズ $n = 100$
- シミュレーション回数 $b = 10000$

```
# run0038.R
# n=100, b=10000
simx <- myrnormb(100,b=10000) # n=100, b=10000
simt <- apply(simx,2,function(x) unlist(mymeansd(x))) # 統計量を繰り返し適用
sims <- apply(cbind(t0,simt),1,
              function(x) unlist(mycalcrms(x))) # 各統計量の mean,sd,rms
print(sims) # 表示
```

```
> source("run0038.R")
             mean       median     karikomi        tmean         sd       iqr
mean -0.002193727 -0.002392857 -0.002157381 -0.002211460 0.99757808 0.9866762
sd    0.099506990  0.124668535  0.102795207  0.104075987 0.07044747 0.1151082
rms   0.099526194  0.124685264  0.102812705  0.104094277 0.07048557 0.1158710
           mad        tsd       sd2      iqr2      mad2      tsd2
mean 0.9928688 0.99363375 1.0001244 0.9867784 0.9993569 0.9929115
sd   0.1164893 0.07485978 0.1409518 0.2302896 0.2345734 0.1492883
rms  0.1167016 0.07512626 0.1409448 0.2306573 0.2345626 0.1494491
> sims0038 <- sims
```

- mean, median, karikomi, tmean の中では，mean が RMSE を最小にする．
- sd2, iqr2, mad2, tsd2 の中では，sd2 のバイアスがゼロ（不偏）であるが，他の 3 個も，バイアスはほぼゼロになる．これら 3 個は，$n$ が十分大きければ，ほぼ不偏になる．
- sd2, iqr2, mad2, tsd2 の中では，sd2 のバラツキが最も小さく，RMSE を最小にする．tsd2 の RMSE もほぼ同じ．

## 2.3 その 3

- 中程度のハズレ値が 5 %あり．サンプルサイズ＝大
- データ $x = (x_1, \ldots, x_n)$，各要素の 95%は独立に平均 0，分散 1 の正規分布 $x_i \sim N(0, 1)$ に従うとする．残りの 5%は平均 3，分散 1 の正規分布 $x_i \sim N(3, 1)$ に従うとする．
- データサイズ $n = 100$
- シミュレーション回数 $b = 10000$

```
# run0039.R
# n=100, b=10000 (mean=3 を 5%混合)
simx1 <- myrnormb(95,b=10000) # mean=0, sd=1, n=95, b=10000
simx2 <- myrnormb(5,mean=3,b=10000) # mean=3, sd=1, n=10, b=10000
simx <- rbind(simx1,simx2) # 混合する n=100, b=10000
simt <- apply(simx,2,function(x) unlist(mymeansd(x))) # 統計量を繰り返し適用
sims <- apply(cbind(t0,simt),1,
              function(x) unlist(mycalcrms(x))) # 各統計量の mean,sd,rms
print(sims) # 表示
```

```
> source("run0039.R")
           mean     median   karikomi      tmean         sd       iqr       mad
mean 0.15061938 0.06570235 0.08652966 0.07775998 1.1954449 1.0494428 1.0553901
sd   0.09971118 0.12722193 0.10446674 0.10613099 0.0818505 0.1230098 0.1245980
```

```
rms  0.18063091 0.14318031 0.13564509 0.13156472 0.2118904 0.1325688 0.1363494
            tsd       sd2      iqr2      mad2      tsd2
mean 1.11458334 1.4357875 1.1164600 1.1293714 1.2487836
sd   0.08054976 0.1959607 0.2612120 0.2662897 0.1801988
rms  0.14006054 0.4778153 0.2859857 0.2960406 0.3071834
> sims0039 <- sims
```

- ハズレ値が５％ある場合．真値 $t_0$ の定義では，ハズレ値を無視する．
- mean, median, karikomi, tmean のなかでは，median のバイアスが最小．karikomi, tmean のバイアスもほぼ同じ．mean のバイアスが最大で，ハズレ値に弱いことを示す．バラツキはどれも大差ない．結果として，tmean がＲＭＳＥ最小，mean が最大．
- sd2, iqr2, mad2, tsd2 の中では，sd2 のバイアスが最大で，ＲＭＳＥも最大．やはり sd2 はハズレ値に弱い．

## 2.4 その４

- 大きなハズレ値が１％あり．サンプルサイズ＝大
- データ $x = (x_1, \ldots, x_n)$，各要素の 99%は独立に平均０，分散１の正規分布 $x_i \sim N(0,1)$ に従うとする．残りの 1%は平均３０，分散１の正規分布 $x_i \sim N(30,1)$ に従うとする．
- データサイズ $n = 100$
- シミュレーション回数 $b = 10000$

```
# run0041.R
# n=100, b=10000 (mean=30 を 1%混合)
simx1 <- myrnormb(99,b=10000) # mean=0, sd=1, n=99, b=10000
simx2 <- myrnormb(1,mean=30,b=10000) # mean=30, sd=1, n=1, b=10000
simx <- rbind(simx1,simx2) # 混合する n=100, b=10000
simt <- apply(simx,2,function(x) unlist(mymeansd(x))) # 統計量を繰り返し適用
sims <- apply(cbind(t0,simt),1,
              function(x) unlist(mycalcrms(x))) # 各統計量の mean,sd,rms
print(sims) # 表示
```

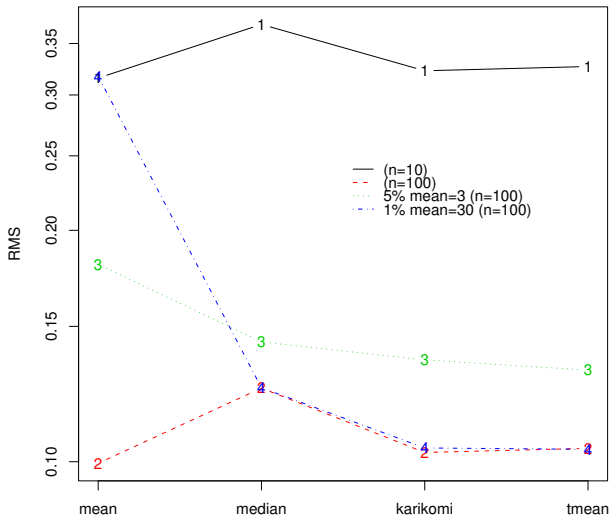
```
> source("run0041.R")
          mean     median   karikomi       tmean         sd       iqr       mad
mean 0.3011617 0.01397016 0.01769648 0.003502758 3.16135441 0.9947431 1.0016417
sd   0.1000034 0.12401011 0.10270399 0.103721946 0.09737281 0.1158773 0.1173871
rms  0.3173296 0.12478836 0.10421238 0.103775891 2.16354648 0.1159907 0.1173927
            tsd        sd2      iqr2      mad2      tsd2
mean 1.03089212 10.0036422 1.0029400 1.0170645 1.0689682
```

```
sd   0.07893174  0.6159101 0.2335482 0.2382560 0.1637370
rms  0.08475801  9.0246817 0.2335550 0.2388544 0.1776619
> sims0041 <- sims
```

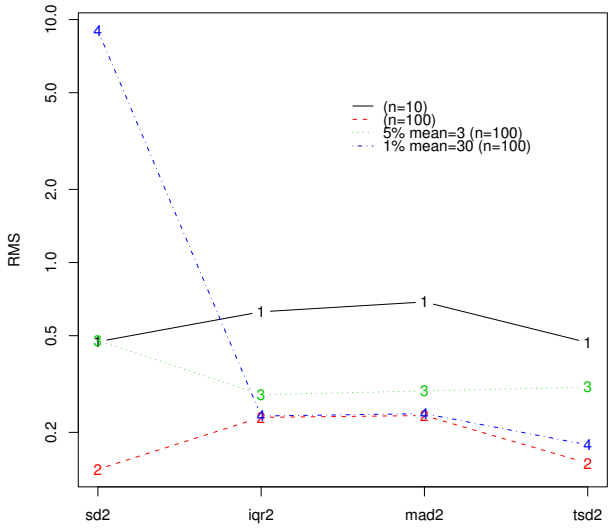
- 大きなハズレ値１％
- mean と sd2 がハズレ値に弱くＲＭＳＥ最大になっていることが際立つ.

## 2.5　シミュレーションのまとめ

```
> simsrms <- rbind(sims0037["rms",],sims0038["rms",],sims0039["rms",],sims0041["rms",])
> simsrms[,1:4]
          mean    median  karikomi     tmean
[1,] 0.3155665 0.3706148 0.3225327 0.3266935
[2,] 0.0995262 0.1246853 0.1028127 0.1040943
[3,] 0.1806309 0.1431803 0.1356451 0.1315647
[4,] 0.3173296 0.1247884 0.1042124 0.1037759
> matplot(t(simsrms[,1:4]),type="b",xaxt="n",ylab="RMS",log="y")
> axis(1,at=1:4,labels=dimnames(simsrms)[[2]][1:4])
> legend(2.5,0.25,c("(n=10)","(n=100)","5% mean=3 (n=100)","1% mean=30 (n=100)"),col=1:4
> dev.copy2eps(file="run0037-sims1.eps")
X11
  2
> simsrms[,9:12]
           sd2      iqr2      mad2      tsd2
[1,] 0.4724049 0.6270997 0.6884513 0.4684996
[2,] 0.1409448 0.2306573 0.2345626 0.1494491
[3,] 0.4778153 0.2859857 0.2960406 0.3071834
[4,] 9.0246817 0.2335550 0.2388544 0.1776619
> matplot(t(simsrms[,9:12]),type="b",xaxt="n",ylab="RMS",log="y")
> axis(1,at=1:4,labels=dimnames(simsrms)[[2]][9:12])
> legend(2.5,5,c("(n=10)","(n=100)","5% mean=3 (n=100)","1% mean=30 (n=100)"),col=1:4,lt
> dev.copy2eps(file="run0037-sims2.eps")
X11
  2
```




run0037-sims1




run0037-sims2

# 3 推定量の誤差評価

- 「シミュレーション」で見たように，統計量には誤差（バイアスとバラツキ）がある．
- シミュレーションではデータの従う分布や真値 $t_0$ があらかじめ分かっている（こちらで設定している）ので，推定量の従う分布，バイアス，バラツキ等を数値的に求めることができた．
- 現実のデータ解析では，データの従う分布も $t_0$ も未知なので，シミュレーションはつかえない．データだけからすべてを計算する必要がある．
- データから，推定量の値を計算するだけでなく，その推定量の誤差（バイアス，バラツキ）を推定したり，さらには推定量の従う分布を推定する．データ解析の信頼性を知るには，このような誤差評価が重要である．

## 3.1 誤差の理論値とプラグイン推定量

- 非常に簡単な統計量ならば，誤差の表す式（理論値）を与えられる．
- その式には通常未知パラメタが含まれているが，そのパラメタに推定量を代入（プラグイン推定）すれば，誤差を推定できる．
- データとその確率値を区別して次のように書く

1. 観測したサイズ $n$ のデータ（確率変数の実現値）

$$x_1, \ldots, x_n$$

2. データに対応する確率変数

$$X_1, \ldots, X_n$$

これらは互いに独立に密度関数 $X \sim f(x)$ に従うものとする．

3. $X$ の期待値と分散

$$E(X) = \mu, \quad V(X) = \sigma^2$$

この二つは未知パラメタで，これらをデータから推定したい．

4. データの平均と不偏分散

$$\bar{x} = \frac{x_1 + \cdots + x_n}{n}, \quad S_x^2 = \frac{(x_1 - \bar{x})^2 + \cdots + (x_n - \bar{x})^2}{n-1} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

5. それに対応する確率変数

$$\bar{X} = \frac{X_1 + \cdots + X_n}{n}, \quad S_X^2 = \frac{(X_1 - \bar{X})^2 + \cdots + (X_n - \bar{X})^2}{n-1} = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2$$

- $\bar{X}$ の実現値が $\bar{x}$ であり，これは $\mu$ の推定値．$S_X^2$ の実現値が $S_x^2$ であり，これは $\sigma^2$ の推定値．$\bar{X}$ と $\sigma_X^2$ は不偏な推定量

$$E(\bar{X}) = \mu, \quad E(S_X^2) = \sigma^2$$

つまりバイアスの理論値はゼロ

$$E(\bar{X}) - \mu = 0, \quad E(S_X^2) - \sigma^2 = 0$$

ここまでは単純計算で確かめられる．

- $X$ の標準偏差 $\sigma$ の推定は $\sqrt{S_X^2} = S_X$ である．

$$E(S_X) \approx \sigma \quad \text{であるが} \quad E(S_X) \neq \sigma$$

つまり $S_X$ は $\sigma$ の不偏な推定量ではない．

- 以下では $f(x)$ が平均 $\mu$，分散 $\sigma^2$ の正規分布 $N(\mu, \sigma^2)$ であると仮定する．

- 推定量のバラツキは次のようにわかる

1. まず $\bar{X}$ について．

$$V(\bar{X}) = \frac{V(X_1) + \cdots + V(X_n)}{n^2} = \frac{V(X)}{n} = \frac{\sigma^2}{n}$$

一般に正規分布に従う確率変数の線形結合も正規分布に従うので，

$$\bar{X} \sim N(\mu, \sigma^2/n)$$

となり，統計量の分布形まで理論的に与えられる．また，標準誤差（推定量の分散の平方根）は

$$\sqrt{V(\bar{X})} = \frac{\sigma}{\sqrt{n}}$$

で与えられる．

2. 次に $S_X^2$ について.

$$(n-1)S_X^2/\sigma^2 \sim \chi_{n-1}^2$$

つまり自由度 $n-1$ のカイ二乗分布に従うことが知られている．したがって，一般に自由度 $m$ のカイ二乗分布の期待値は $m$, 分散は $2m$ である．これより,

$$E(S_X^2) = (n-1) \times \frac{\sigma^2}{n-1} = \sigma^2, \quad V(S_X^2) = 2(n-1) \times \frac{\sigma^4}{(n-1)^2} = \frac{2\sigma^4}{n-1}$$

標準誤差は

$$\sqrt{V(S_X^2)} = \frac{\sqrt{2}\sigma^2}{\sqrt{n-1}}$$

3. 実際のデータ解析では，$\sigma^2$ は未知である．そこで $\sigma^2$ の不偏推定量 $S_x^2$ を

$$V(\bar{X}) = \frac{\sigma^2}{n}, \quad V(S_X^2) = \frac{2\sigma^4}{n-1}$$

に代入すると，分散の推定量 $\hat{V}(\cdot)$ は

$$\hat{V}(\bar{X}) = \frac{S_x^2}{n}, \quad \hat{V}(S_X^2) = \frac{2S_x^4}{n-1}$$

となる．このように，あらかじめ理論値を求めておき，その未知パラメータを推定値で置き換えて得られる推定量は，プラグイン推定量と呼ばれる.

- 分散 $S_X^2$ の分散推定 $\hat{V}(S_X^2)$ から標準偏差 $S_X$ の分散推定 $\hat{V}(S_X)$ を求めるには，次のように考える.(一般にはデルタ法という手法)．まず，統計量 $T$ の分散を $V(T)$ と書く．滑らかな関数 $g(t)$ を使って変換した $g(T)$ のテーラー展開

$$g(T) = g(t_0) + g'(t_0)(T-t_0) + \tfrac{1}{2}g''(t_0)(T-t_0)^2 + \cdots$$

を

$$g(T) \approx g(t_0) + g'(t_0)(T-t_0) = \text{定数} + g'(t_0)T$$

で近似すると,

$$V(g(T)) \approx (g'(t_0))^2 V(T)$$

従って，$g(T) = \sqrt{T}$，$T = S_X^2$ と置けば，$g'(t) = 1/(2\sqrt{t})$ なので,

$$\hat{V}(S_X) = \hat{V}(S_X^2)/(4S_x^2) = S_x^2/(2(n-1))$$

- 以下の数値例 `run0047.R` では，$n = 100, \mu = 0, \sigma = 2, X \sim N(0,4)$ としている.

```
# run0044.R
# 「平均，分散」の「分散，標準偏差」
library(MASS) # truehist の定義
func0044 <- function(x) {
  n <- length(x) # データサイズ
  m <- sum(x)/n # 平均
```

```
  v <- sum( (x-m)^2 )/(n-1) # 不偏分散
  s <- sqrt(v) # 標準偏差
  vm <- v/n # 平均の分散推定
  vv <- 2*v^2/(n-1) # 不偏分散の分散推定
  vs <- v/(2*(n-1)) # 標準偏差の分散推定
  sm <- sqrt(vm); sv <- sqrt(vv); ss <- sqrt(vs)
  list(mean=m,var=v,sd=s,varmean=vm,varvar=vv,varsd=vs,
       sdmean=sm,sdvar=sv,sdsd=ss)
}
drawhist <- function(x,nb,name=NULL,prefix=NULL) {
  truehist(x,nbins=nb,xlab=""); rug(x); lines(density(x),col=2)
  m <- mean(x) ; s <- sd(x)
  abline(v=m,col=3,lty=3) # 平均
  title(main=name,cex.sub=2,cex.main=2,
        sub=paste("mean=",signif(m,5),", sd=",signif(s,5)))
  if(!is.null(prefix))
    dev.copy2eps(file=paste(prefix,name,".eps",sep=""))
}
```

```
# run0047.R
# 誤差の理論値 n=100
n <- 100 # データサイズ
sigma <- 2 # sigma の真値
v <- list(mean=sigma^2/n,var=2*sigma^4/(n-1), # 平均, 分散の分散の理論値
          sd=sigma^2/(2*(n-1))) # 標準偏差の分散の理論値（近似）
cat("\n分散の理論値\n"); print(unlist(v))
cat("\n標準偏差の理論値\n"); print(sqrt(unlist(v)))
simx <- myrnormb(n,mean=0,sd=sigma,b=10000) # データの生成
simt <- apply(simx,2,function(x) unlist(func0044(x))) # 統計量を繰り返し適用
cat("\n統計量の平均, 分散, 標準偏差\n")
a <- apply(simt,1,function(x) unlist(list(mean=mean(x),var=var(x),sd=sd(x))))
print(a) # シミュレーション結果の表示
cat("\n分散のシミュレーション値\n")
print(a["var",c("mean","var","sd")])
cat("\n標準偏差のシミュレーション値\n")
print(sqrt(a["var",c("mean","var","sd")]))
cat("\n分散の推定値の平均値\n")
print(a["mean",c("varmean","varvar","varsd")])
cat("\n標準偏差の推定値の平均値\n")
print(a["mean",c("sdmean","sdvar","sdsd")])
for(i in rownames(simt)) drawhist(simt[i,],20,i,"run0047-") # ヒストグラム
```

```
> source("run0044.R")
> source("run0047.R")
```

分散の理論値

```
      mean        var         sd
0.04000000 0.32323232 0.02020202
```

標準偏差の理論値

```
     mean       var        sd
0.2000000 0.5685352 0.1421338
```

統計量の平均，分散，標準偏差

```
             mean       var         sd      varmean     varvar        varsd
mean -0.003385110 3.9977999 1.99441662 3.997800e-02 0.32938154 2.019091e-02
var   0.040029976 0.3220148 0.02010425 3.220148e-05 0.00882014 8.213824e-06
sd    0.200074927 0.5674634 0.14178944 5.674634e-03 0.09391560 2.865977e-03
           sdmean       sdvar         sdsd
mean 0.1994416617 0.568222532 0.1417370171
var  0.0002010425 0.006505349 0.0001015366
sd   0.0141789440 0.080655742 0.0100765367
```

分散のシミュレーション値

```
      mean        var         sd
0.04002998 0.32201476 0.02010425
```

標準偏差のシミュレーション値

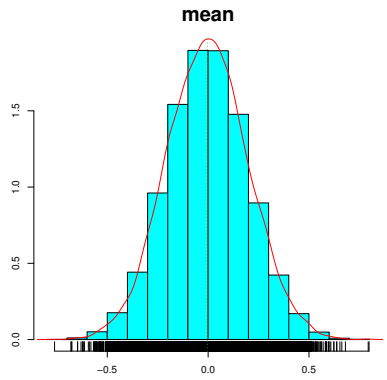
```
     mean       var        sd
0.2000749 0.5674634 0.1417894
```

分散の推定値の平均値

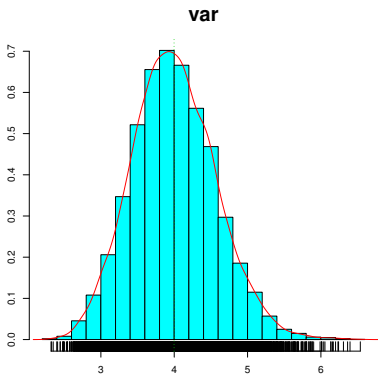
```
   varmean     varvar      varsd
0.03997800 0.32938154 0.02019091
```

標準偏差の推定値の平均値

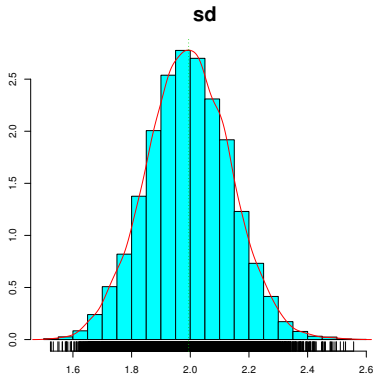
```
   sdmean     sdvar      sdsd
0.1994417 0.5682225 0.1417370
> simx0047 <- simx
```



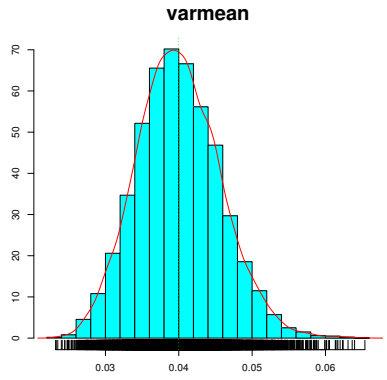

mean= –0.0033851 , sd= 0.20007
run0047-mean



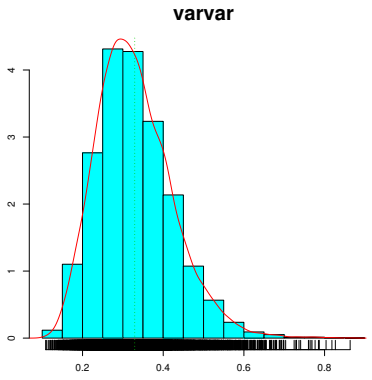

mean= 3.9978 , sd= 0.56746
run0047-var



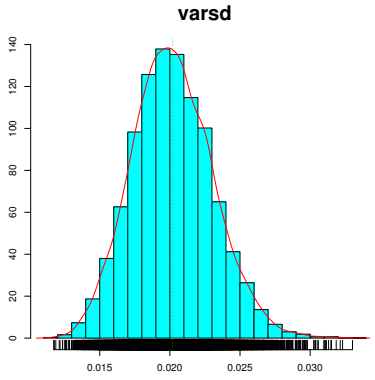

mean= 1.9944 , sd= 0.14179
run0047-sd



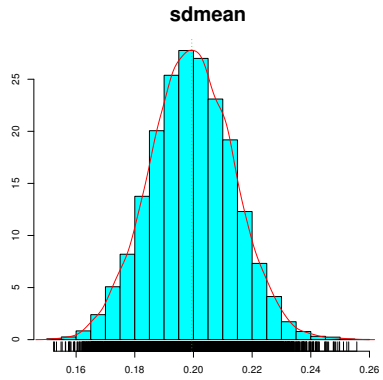

mean= 0.039978 , sd= 0.0056746
run0047-varmean



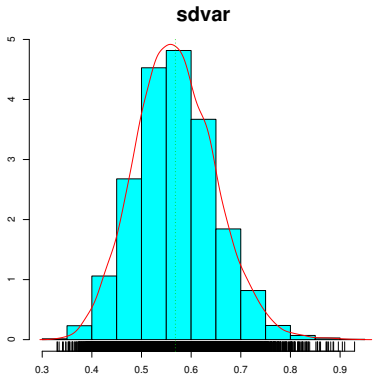

mean= 0.32938 , sd= 0.093916
run0047-varvar



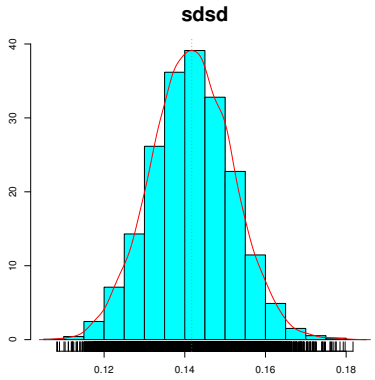

mean= 0.020191 , sd= 0.002866
run0047-varsd




mean= 0.19944 , sd= 0.014179
run0047-sdmean




mean= 0.56822 , sd= 0.080656
run0047-sdvar




mean= 0.14174 , sd= 0.010077
run0047-sdsd

- $n = 100, \mu = 0, \sigma^2 = 4$
- 分散の理論値

$$V(\bar{X}) = 0.0400, \quad V(S_X^2) = 0.323, \quad V(S_X) \approx 0.0202$$

- 標準偏差の理論値

$$\sqrt{V(\bar{X})} = 0.200, \quad \sqrt{V(S_X^2)} = 0.569, \quad \sqrt{V(S_X)} \approx 0.142$$

- 分散のシミュレーション値（理論値にほぼ一致）

$$V(\bar{X}) \approx 0.0400, \quad V(S_X^2) \approx 0.322, \quad V(S_X) \approx 0.0201$$

- 標準偏差のシミュレーション値（理論値にほぼ一致）

$$\sqrt{V(\bar{X})} \approx 0.200, \quad \sqrt{V(S_X^2)} = 0.567, \quad \sqrt{V(S_X)} \approx 0.142$$

- 分散の推定量の平均値のシミュレーション値

$$E(\hat{V}(\bar{X})) \approx 0.0400, \quad E(\hat{V}(S_X^2)) \approx 0.329, \quad E(\hat{V}(S_X)) \approx 0.0202$$

- 標準偏差の推定量の平均値のシミュレーション値

$$E\left(\sqrt{\hat{V}(\bar{X})}\right) \approx 0.199, \quad E\left(\sqrt{\hat{V}(S_X^2)}\right) \approx 0.568, \quad E\left(\sqrt{\hat{V}(S_X)}\right) \approx 0.142$$

しかしながら，分散の推定量 $\hat{V}(\bar{X})$, $\hat{V}(S_X^2)$ 等はバラツキがあり，個々のデータに対して計算した推定値は誤差がある．（グラフ varmean, varvar, varsd, sdmean, sdvar, sdsd を見ると，バラツキの様子がわかるだろう）

## 3.2 最尤推定量の分散＊

- 最尤推定量の分散は近似値が理論的に与えられる．
- 確率モデル $X \sim f(x|\theta)$ を仮定する．$m$ 次元の未知パラメタベクトル $\theta = (\theta_1, \ldots, \theta_m)$ の最尤推定量 $\hat{\theta}$ は

$$\ell(\theta) = \sum_{i=1}^{n} \log f(x_i|\theta)$$

を最大にする $\theta$ の値として定義される．$\hat{\theta}$ はデータ $x_1, \ldots, x_n$ の関数 $\hat{\theta}(x_1, \ldots, x_n)$ と考えられる．

- 最尤推定量の分散（実際にはサイズ $m \times m$ の分散共分散行列）$V(\hat{\theta}(X_1, \ldots, X_n))$ は近似的に次式で与えられる．

$$V(\hat{\theta}(X_1, \ldots, X_n)) \approx H^{-1}(\theta)/n$$

ただし $H(\theta)$ は Fisher 情報行列と呼ばれその $(i, j)$ 成分は

$$H_{ij}(\theta) = E\left\{-\frac{\partial^2 \log f(X|\theta)}{\partial\theta_i \partial\theta_j}\right\}$$

- 現実のデータ解析では $\theta$ の真値は未知なので，その推定量で置き換え，$\hat{\theta}$ の分散を

$$\hat{V}(\hat{\theta}(X_1, \ldots, X_n)) = H^{-1}(\hat{\theta})/n$$

で推定する．もしくは

$$\hat{V}(\hat{\theta}(X_1, \ldots, X_n)) = \hat{H}^{-1}/n$$

$$\hat{H}_{ij} = -\frac{1}{n}\sum_{i=1}^{n} \frac{\partial^2 \log f(X|\theta)}{\partial\theta_i \partial\theta_j}\Big|_{\hat{\theta}} = -\frac{1}{n}\frac{\partial^2 \ell(\theta)}{\partial\theta_i \partial\theta_j}\Big|_{\hat{\theta}}$$

で推定しても良い．これらも一種のプラグイン推定量である．行列 $\hat{H}$ は目的関数の 2 回微分（ヘシアン）である．つまり，対数尤度関数の $\hat{\theta}$ の周辺でのピークの鋭さの逆数が分散に対応している．（ピークが鋭いほど分散が小さく推定精度が高いことを意味する．）

- パラメタの各成分 $\theta_i$, $i = 1, \ldots, m$ の分散は，$\hat{V}(\hat{\theta}(X_1, \ldots, X_n))$ の対角項 $(i, i)$ によって与えられる．

- fitdistr 関数の与える sd は $\hat{H}^{-1}/n$ の対角成分である．(fitdistr の定義を見てみよ．sds <- sqrt(diag(solve(res$hessian))) とある．res$hessian は対数尤度の 2 回微分から求めた Fisher 情報行列の推定，solve は逆行列を計算する関数．diag は対角項を取り出す関数.)

```
# run0045.R
# 最尤推定量の分散
x <- rnorm(100,mean=0,sd=2) # データセット（サイズ n=100) を 1 個生成する.
a <- unlist(func0044(x)) ; print(a)
cat("\n 平均と標準偏差の標準誤差\n")
print(sqrt(a[c(4,6)]))
cat("\n 平均と標準偏差の最尤推定 (標準誤差)\n")
print(fitdistr(x,"normal"))
```

```
> source("run0045.R")
       mean          var           sd     varmean      varvar       varsd
-0.05801715  3.91772064  1.97932328  0.03917721  0.31007141  0.01978647
     sdmean       sdvar        sdsd
 0.19793233  0.55684057  0.14066438

平均と標準偏差の標準誤差
  varmean     varsd
0.1979323 0.1406644

平均と標準偏差の最尤推定 (標準誤差)
      mean           sd
  -0.05801715    1.96940180
 ( 0.19694018) ( 0.13925774)
> x0045 <- x
```

- データ x0045：統計量 $\bar{x}$ と $S_x$ の標準偏差（理論値へのプラグイン推定値）

$$\sqrt{\hat{V}(\bar{X})} = 0.198, \quad \sqrt{\hat{V}(S_X)} = 0.141$$

- データ x0045：統計量 $\bar{x}$ と $S_x$ の標準偏差（最尤推定に伴うヘシアンから推定）

$$\sqrt{\hat{V}(\bar{X})} = 0.197, \quad \sqrt{\hat{V}(S_X)} = 0.139$$

## 3.3 誤差のブートストラップ推定

- 統計量の誤差推定のための一般的なシミュレーション技法.
- 理論値が分からなくても，結果がだせる．複雑な問題でも使える.(その代わり，計算機を酷使する．これからの主流.)

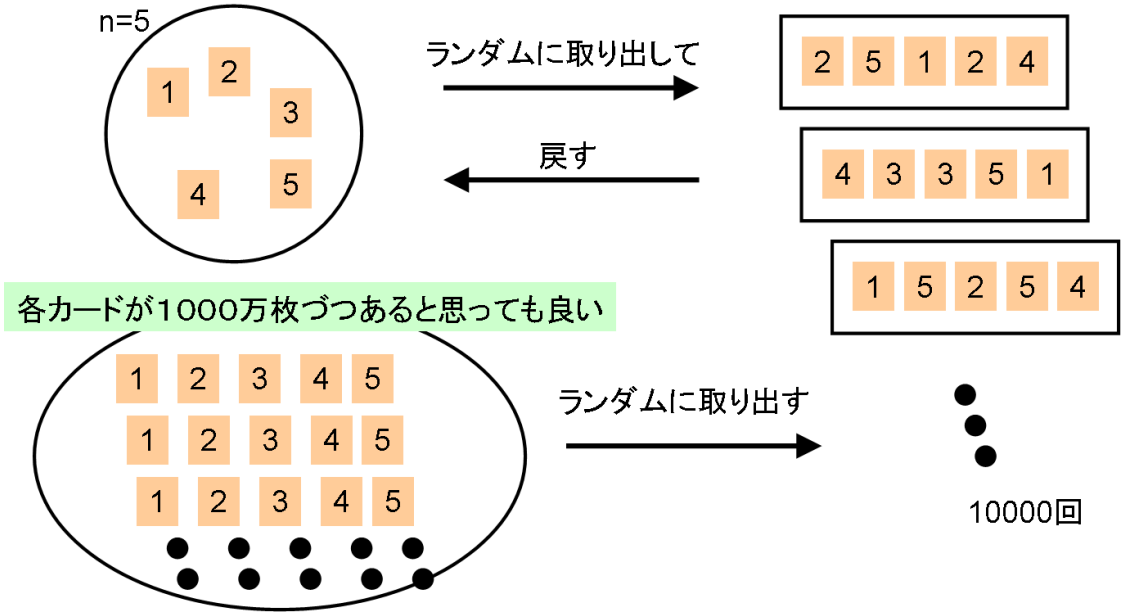
- 通常のシミュレーションでは真の分布からデータセットを多数生成する．ところが実際のデータ解析では真の分布は未知である．そこで観測した 1 個のデータセット（サイズ $n$）から多数のデータセットを生成するというアイデア．

- データ $(x_1, \ldots, x_n)$ からシミュレーションデータ $x(x_1^*, \ldots, x_n^*)$ を生成する手続きは以下のとおり：

  1. $n$ 個の整数値 $\{1, \ldots, n\}$ を等確率 $(1/n)$ で生成する．これを $n$ 回繰り返した結果を $i_1, \ldots, i_n$ とする．たとえていうと，1 から $n$ までの数字の書いてサイコロを $n$ 回振り，結果を記録する．
  2. $x_1^* = x_{i_1}, x_2^* = x_{i_2}, \ldots, x_n^* = x_{i_n}$ と代入する．つまり，$j = 1, \ldots, n$ に対して，新しいデータの $j$ 番目の値 $x_j^*$ は，元のデータの $i_j$ 番目の値 $x_{i_j}$ である．
  3. たったこれだけ．

# 復元抽出

ブートストラップ法＝データからの復元抽出（resampling with replacement）




- 上記の手続きを多数回 ($B = 10000$ 回) くりかえし適用し，$B$ 個のシミュレーションデータセットを生成する．

- シミュレーションによって得られた各データセットはブートストラップ標本と呼ばれる．各ブートストラップ標本にたいして計算した統計量は，その統計量のブートストラップ複製と呼ばれる．

- ブートストラップ複製の分布から，統計量のバイアスやバラツキなどが推定できる．

- データセットは未知の母集団からのサンプリングである．ブートストラップ標本は，データからの「リサンプリング」という．

```
# run0046.R
# ブートストラップ法による分散推定
x <- x0045 # run0045で使ったデータ
n <- length(x) # データサイズ
b <- 10000 # シミュレーションの繰り返し回数
simx <- matrix(0,n,b) # ブートストラップ標本のアレイを準備
for(i in 1:b) simx[,i] <- sample(x,replace=T) # ブートストラップ法
simt <- apply(simx,2,function(x) unlist(func0044(x))) # 統計量を繰り返し適用
cat("\n統計量の平均，分散，標準偏差\n")
a <- apply(simt,1,function(x) unlist(list(mean=mean(x),var=var(x),sd=sd(x))))
print(a)
cat("\n平均と標準偏差の標準誤差\n")
print(sqrt(a["var",c("mean","sd")]))
for(i in rownames(simt)) drawhist(simt[i,],20,i,"run0046-") # ヒストグラム
```

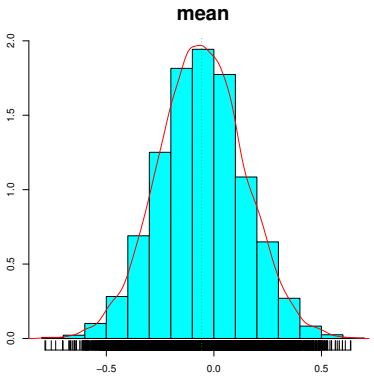
```
> source("run0046.R")
```

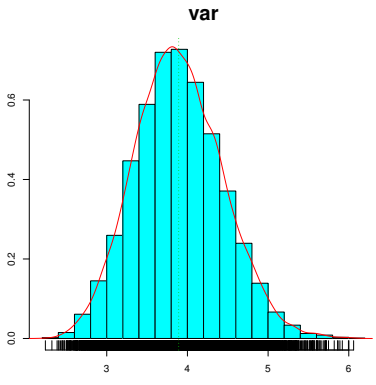
統計量の平均，分散，標準偏差

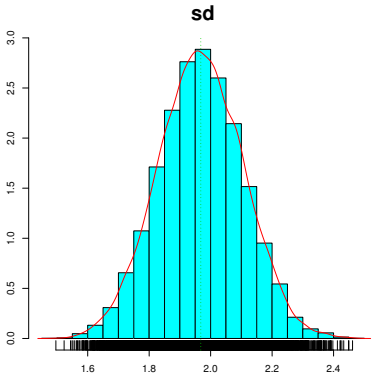
```
            mean       var          sd      varmean      varvar         varsd
mean -0.05652554 3.8919287 1.96799977 3.891929e-02 0.311945889 1.965621e-02
var    0.03985005 0.2942423 0.01890747 2.942423e-05 0.007582336 7.505415e-06
sd     0.19962476 0.5424410 0.13750444 5.424410e-03 0.087076610 2.739601e-03
           sdmean       sdvar          sdsd
mean 0.1967999766 0.553174652 1.398597e-01
var  0.0001890747 0.005944288 9.549227e-05
sd   0.0137504436 0.077099211 9.772015e-03
```

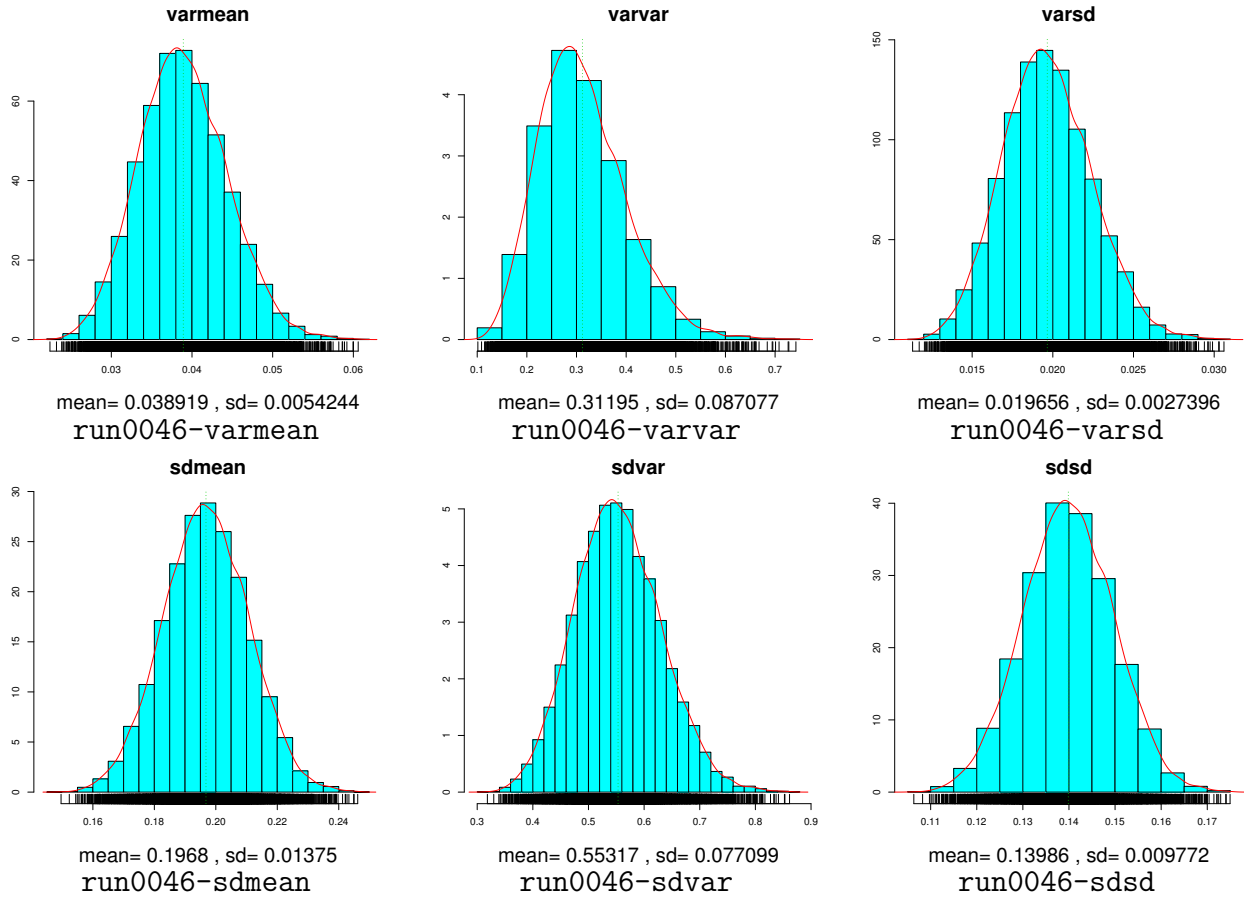
平均と標準偏差の標準誤差

```
     mean        sd
0.1996248 0.1375044
```






- データ x0045：統計量 $\bar{x}$ と $S_x$ の標準偏差（ブートストラップ推定値）

$$\sqrt{\hat{V}(\bar{X})} = 0.200, \quad \sqrt{\hat{V}(S_X)} = 0.138$$

## 3.4 ブートストラップ法の適用例（その１）

- Galaxies データセット $n = 82$ にブートストラップ法を適用する．
- 各種統計量の誤差を評価
- 以下の run0048.R において，ブートストラップ法に相当するのは，sample(x,replace=T) のところ．help(sample) を確認すること．sample(x) はデータ $x$ をランダムに並べ替え．sample(x,replace=T) はデータ $x$ のブートストラップ標本を 1 個生成．

```
> x <- 0:9
> x
 [1] 0 1 2 3 4 5 6 7 8 9
> sample(x)
 [1] 8 4 5 1 0 7 3 6 2 9
> sample(x)
 [1] 9 5 7 8 3 1 4 6 0 2
> sample(x)
 [1] 3 0 8 7 2 1 5 4 6 9
> sample(x,replace=T)
```

```
 [1] 4 5 1 0 6 3 8 7 7 2
> sample(x,replace=T)
 [1] 3 2 4 9 8 9 1 0 2 1
> sample(x,replace=T)
 [1] 5 9 5 3 3 7 2 0 1 5
```

```
# run0048.R
# 各種統計量にブートストラップ法の適用
# x <- galaxies/10000 # in library(MASS)
# filename <- "run0048-"
source("run0042.R") # 中心とバラツキの統計量の復習
source("run0044.R") # 「平均, 分散」の「分散, 標準偏差」
drawhist(x,20,"x",filename) # in run0044.R
cat("\nデータセットに各種統計量を計算（その１）\n")
print(unlist(mymeansd(x))) # 統計量１
cat("\nデータセットに各種統計量を計算（その２）\n")
print(unlist(func0044(x))) # 統計量２
n <- length(x) # データサイズ
b <- 10000 # シミュレーションの繰り返し回数
simx <- matrix(0,n,b) # ブートストラップ標本のアレイを準備
for(i in 1:b) simx[,i] <- sample(x,replace=T) # ブートストラップ法
simt <- apply(simx,2,function(x) unlist(mymeansd(x))) # 統計量を繰り返し適用
cat("\n統計量の平均, 標準偏差\n")
a <- apply(simt,1,function(x) unlist(list(mean=mean(x),sd=sd(x))))
print(a) # シミュレーション結果の表示
cat("\n統計量のバイアス推定\n")
print(a["mean",] - unlist(mymeansd(x)))
cat("\n統計量の RMSE の推定\n")
print(sqrt((a["mean",] - unlist(mymeansd(x)))^2 + a["sd",]^2 ))
for(i in rownames(simt)) drawhist(simt[i,],20,i,filename) # ヒストグラム
```

```
> library(MASS)
> x <- galaxies/10000 # in library(MASS)
> filename <- "run0048-"
> source("run0048.R")
```

データセットに各種統計量を計算（その１）

```
      mean     median   karikomi      tmean         sd        iqr        mad
2.08281707 2.08335000 2.11469242 2.11443600 0.45637580 0.26694253 0.23736426
       tsd        sd2       iqr2       mad2       tsd2
0.34754599 0.20827887 0.07125831 0.05634179 0.12078822
```

データセットに各種統計量を計算（その２）

```
       mean         var          sd     varmean      varvar       varsd
2.082817073 0.208278870 0.456375799 0.002539986 0.001071113 0.001285672
     sdmean       sdvar        sdsd
0.050398276 0.032727867 0.035856269
```

統計量の平均，標準偏差

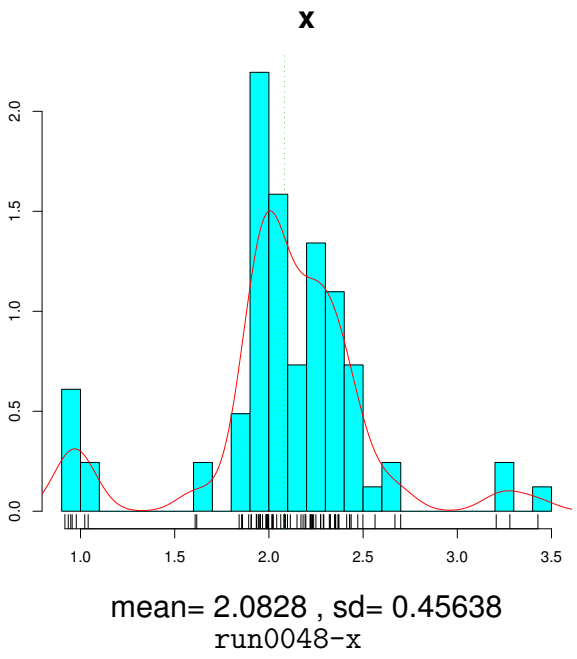
```
           mean     median   karikomi      tmean         sd        iqr
mean 2.08323249 2.08747362 2.10865742 2.11324202 0.45032618 0.25937529
sd   0.05036561 0.05190322 0.03921362 0.03344895 0.05214537 0.02944672
            mad       tsd        sd2       iqr2       mad2       tsd2
mean 0.25360792 0.3482585 0.20551253 0.06814256 0.06561677 0.12421679
sd   0.03605450 0.0541583 0.04683001 0.01523341 0.01867089 0.03930798
```

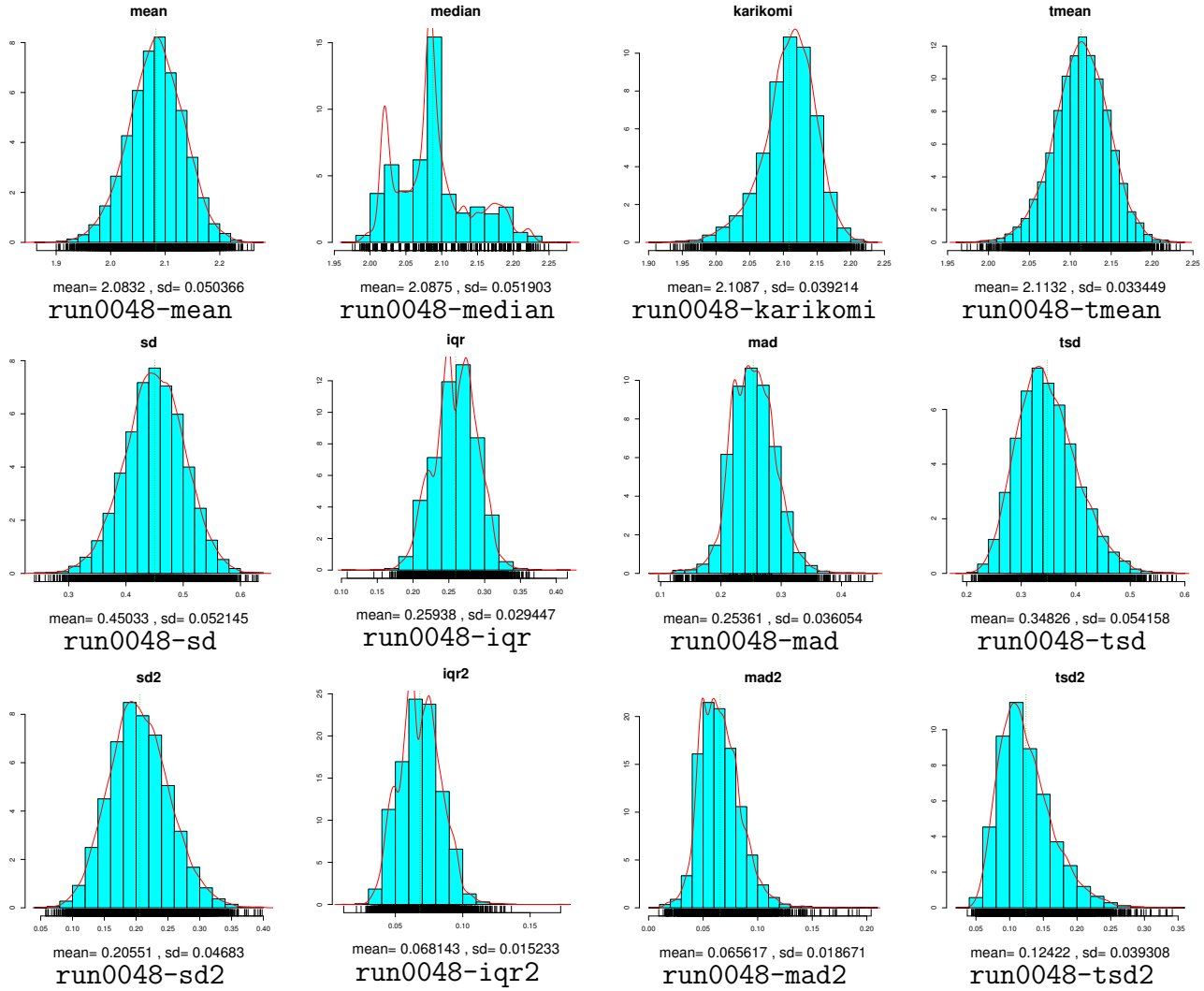
統計量のバイアス推定

```
         mean        median      karikomi         tmean            sd
 0.0004154198  0.0041236200 -0.0060350073 -0.0011939811 -0.0060496224
          iqr           mad           tsd           sd2          iqr2
-0.0075672425  0.0162436621  0.0007124732 -0.0027663368 -0.0031157522
         mad2          tsd2
 0.0092749829  0.0034285706
```

統計量の RMSE の推定

```
      mean     median   karikomi      tmean         sd        iqr        mad
0.05036732 0.05206677 0.03967530 0.03347025 0.05249512 0.03040349 0.03954470
       tsd        sd2       iqr2       mad2       tsd2
0.05416299 0.04691164 0.01554879 0.02084772 0.03945722
There were 50 or more warnings (use warnings() to see the first 50)
```






## 3.5 ブートストラップ法の適用例（その２）

- 自分でマウスクリックして入力したデータにブートストラップ法を適用する．
- 各種統計量の誤差を評価

```
# run0049.R
# 座標の入力
getpoints <- function(n=100,xlim=c(0,1),ylim=c(0,1)) {
  plot(0,0,xlab="x",ylab="y",xlim=xlim,ylim=ylim,type="n") # 枠を描く
  locator(n,type="p") # 左ボタンクリックでデータ点の入力．他のボタンで終了
}
```

```
> source("run0049.R")
> xy <- getpoints() # 100 個入力すると終了する
> dev.copy2eps(file="run0049-xy.eps")
X11
  2
> x <- xy$x # x 成分だけ使う．
```

```
> length(x)
[1] 100
> filename <- "run0049-"
> source("run0048.R")
```

データセットに各種統計量を計算（その１）

```
       mean      median    karikomi       tmean          sd         iqr
0.222820818 0.188054689 0.195130278 0.190165475 0.157858818 0.056432883
        mad         tsd         sd2        iqr2        mad2        tsd2
0.056779012 0.080432130 0.024919406 0.003184670 0.003223856 0.006469328
```

データセットに各種統計量を計算（その２）

```
        mean          var           sd      varmean       varvar        varsd
2.228208e-01 2.491941e-02 1.578588e-01 2.491941e-04 1.254499e-05 1.258556e-04
      sdmean        sdvar         sdsd
1.578588e-02 3.541890e-03 1.121854e-02
```

統計量の平均，標準偏差

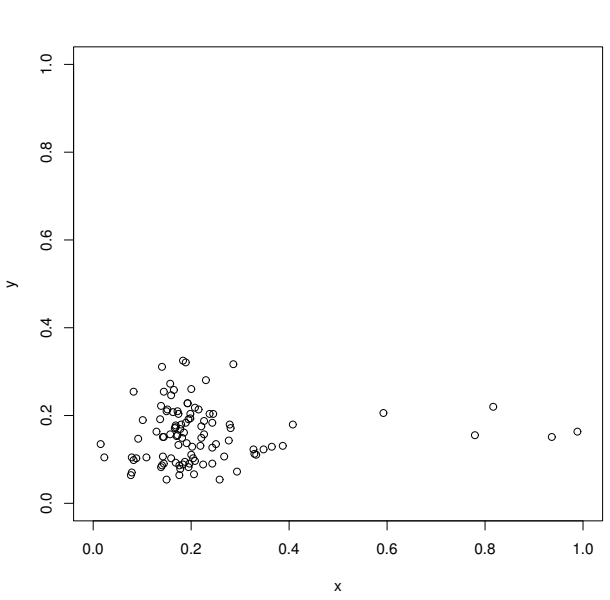
```
           mean      median    karikomi       tmean         sd        iqr
mean 0.22253342 0.187231492 0.195146190 0.190503257 0.15408508 0.05851134
sd   0.01558411 0.006269289 0.008200379 0.007123813 0.02894646 0.01004548
            mad        tsd        sd2        iqr2         mad2        tsd2
mean 0.05644789 0.08076099 0.02458003 0.003524479 0.0032592103 0.006679574
sd   0.00853539 0.01254001 0.00889349 0.001242645 0.0009803288 0.002153355
```

統計量のバイアス推定

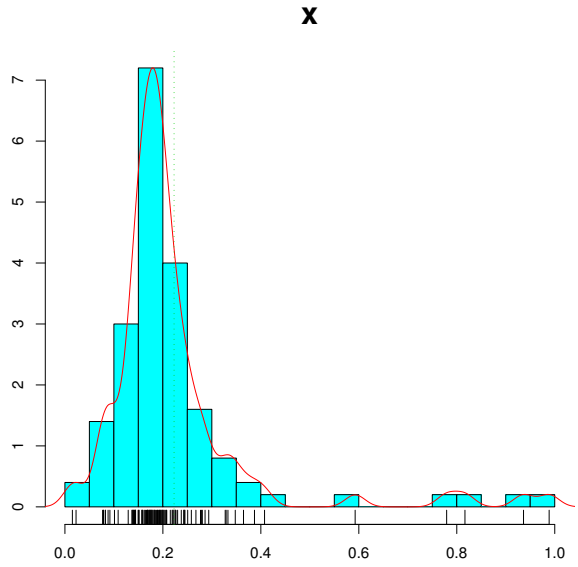
```
         mean        median      karikomi         tmean            sd
-2.873950e-04 -8.231969e-04  1.591190e-05  3.377823e-04 -3.773735e-03
          iqr           mad           tsd           sd2          iqr2
 2.078461e-03 -3.311186e-04  3.288613e-04 -3.393796e-04  3.398086e-04
         mad2          tsd2
 3.535406e-05  2.102462e-04
```

統計量の RMSE の推定

```
        mean       median     karikomi        tmean           sd          iqr
0.0155867606 0.0063231035 0.0082003946 0.0071318162 0.0291914177 0.0102582444
         mad          tsd          sd2         iqr2         mad2         tsd2
0.0085418100 0.0125443191 0.0088999638 0.0012882687 0.0009809661 0.0021635940
```

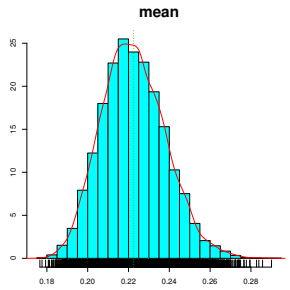



run0049-xy



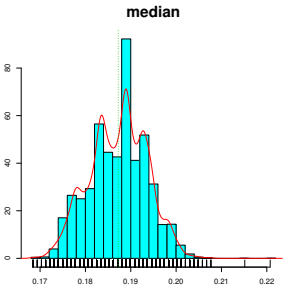

mean= 0.22282 , sd= 0.15786

run0049-x



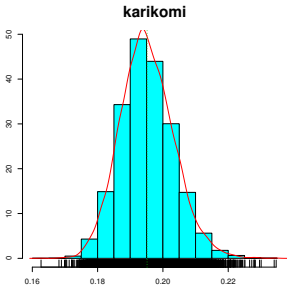

mean= 0.22253 , sd= 0.015584

run0049-mean



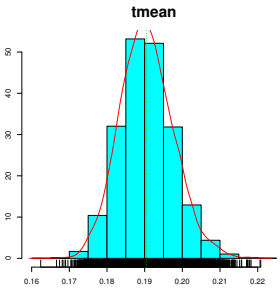

mean= 0.18723 , sd= 0.0062693

run0049-median



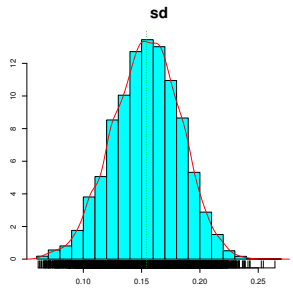

mean= 0.19515 , sd= 0.0082004

run0049-karikomi



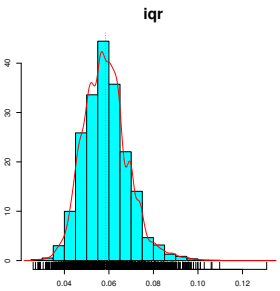

mean= 0.1905 , sd= 0.0071238

run0049-tmean



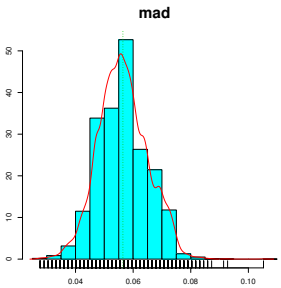

mean= 0.15409 , sd= 0.028946

run0049-sd



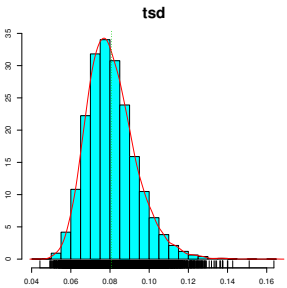

mean= 0.058511 , sd= 0.010045

run0049-iqr



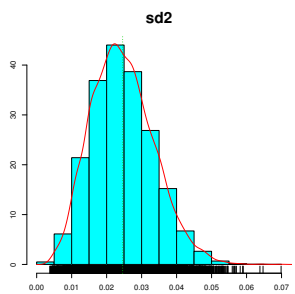

mean= 0.056448 , sd= 0.0085354

run0049-mad



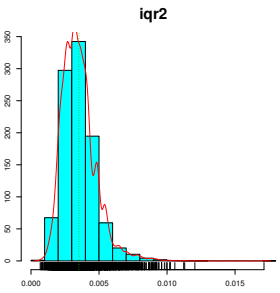

mean= 0.080761 , sd= 0.01254

run0049-tsd



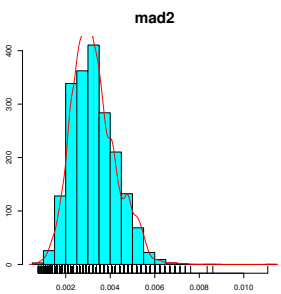

mean= 0.02458 , sd= 0.0088935

run0049-sd2



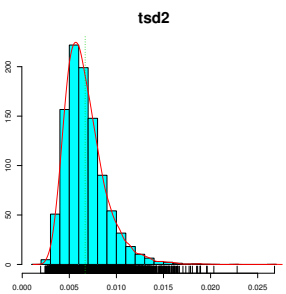

mean= 0.0035245 , sd= 0.0012426

run0049-iqr2




mean= 0.0032592 , sd= 0.00098033

run0049-mad2




mean= 0.0066796 , sd= 0.0021534

run0049-tsd2

## 3.6　ブートストラップ法の適用例（その３）

- `dat0002$Zouka`： X2000 のコード A05201 自然増加率 (%) $n = 47$ にブートストラップ法を適用する.

- 各種統計量の誤差を評価

```
> x <- dat0002$Zouka
> filename <- "zouka-"
> source("run0048.R")
```

データセットに各種統計量を計算（その１）

```
      mean     median   karikomi      tmean         sd        iqr        mad
0.07957447 0.07000000 0.07282051 0.07117966 0.18004779 0.14826022 0.14826000
       tsd        sd2       iqr2       mad2       tsd2
0.16661943 0.03241721 0.02198109 0.02198103 0.02776204
```

データセットに各種統計量を計算（その２）

```
        mean          var           sd      varmean       varvar        varsd
7.957447e-02 3.241721e-02 1.800478e-01 6.897278e-04 4.569023e-05 3.523609e-04
      sdmean        sdvar         sdsd
2.626267e-02 6.759455e-03 1.877128e-02
```

統計量の平均，標準偏差

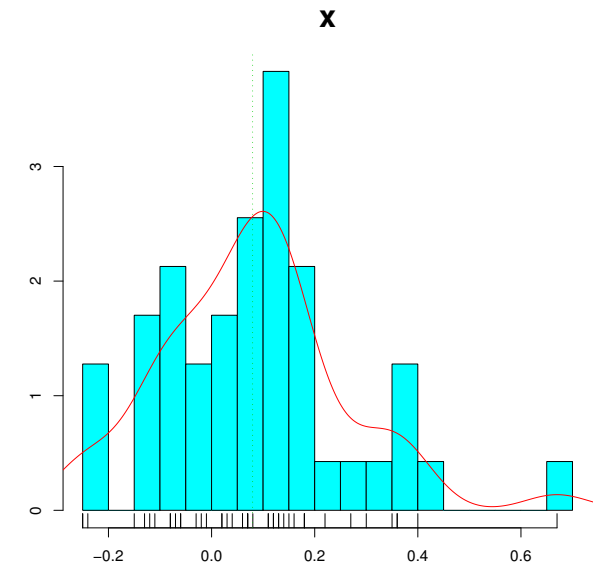
```
           mean     median   karikomi      tmean         sd        iqr
mean 0.07958540 0.08105100 0.07265149 0.07165539 0.17666091 0.14466083
sd   0.02588437 0.02726352 0.02522510 0.02391654 0.02362894 0.02996778
            mad        tsd         sd2        iqr2       mad2        tsd2
mean 0.14535262 0.16494973 0.031767348 0.021824735 0.02242045 0.027703791
sd   0.03596102 0.02225820 0.008459186 0.009109195 0.01123171 0.007472447
```

統計量のバイアス推定

```
         mean        median      karikomi         tmean            sd
 1.093617e-05  1.105100e-02 -1.690256e-04  4.757284e-04 -3.386881e-03
          iqr           mad           tsd           sd2          iqr2
-3.599388e-03 -2.907379e-03 -1.669703e-03 -6.498587e-04 -1.563584e-04
         mad2          tsd2
 4.394227e-04 -5.824396e-05
```

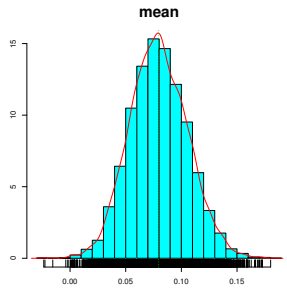
統計量の RMSE の推定

```
       mean      median    karikomi       tmean          sd         iqr
0.025884369 0.029418095 0.025225663 0.023921268 0.023870439 0.030183164
        mad         tsd         sd2        iqr2        mad2        tsd2
0.036078358 0.022320741 0.008484112 0.009110537 0.011240305 0.007472674
```



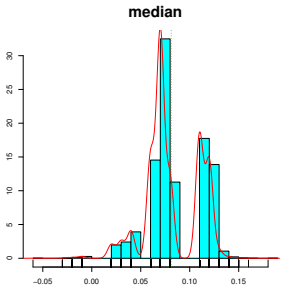

mean= 0.079574 , sd= 0.18005

zouka-x



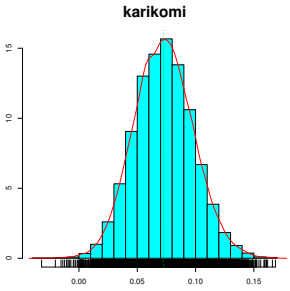

mean= 0.079585 , sd= 0.025884

zouka-mean



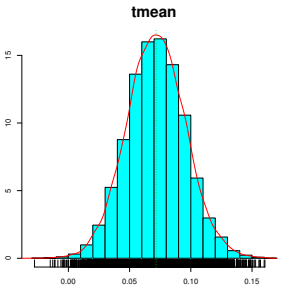

mean= 0.081051 , sd= 0.027264

zouka-median



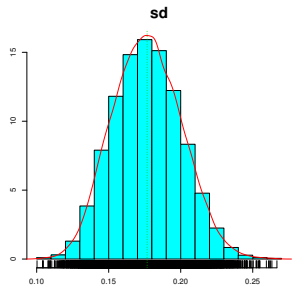

mean= 0.072651 , sd= 0.025225

zouka-karikomi



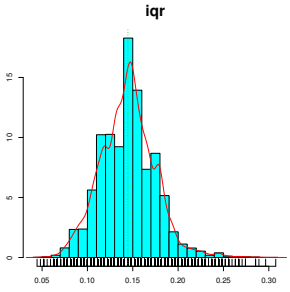

mean= 0.071655 , sd= 0.023917

zouka-tmean



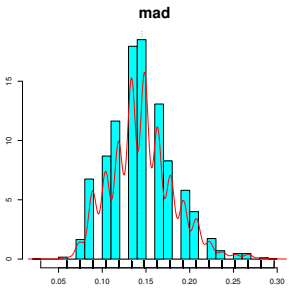

mean= 0.17666 , sd= 0.023629

zouka-sd



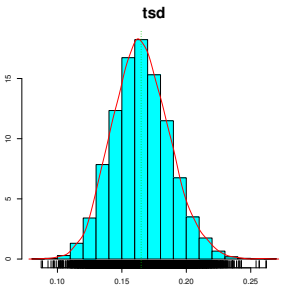

mean= 0.14466 , sd= 0.029968

zouka-iqr



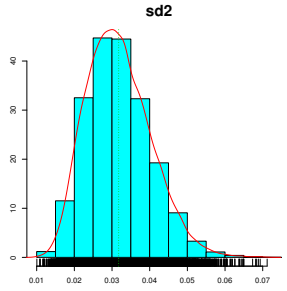

mean= 0.14535 , sd= 0.035961

zouka-mad



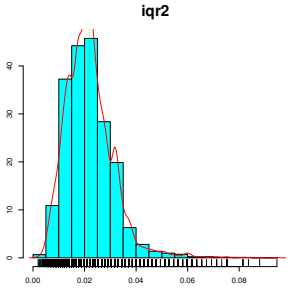

mean= 0.16495 , sd= 0.022258

zouka-tsd



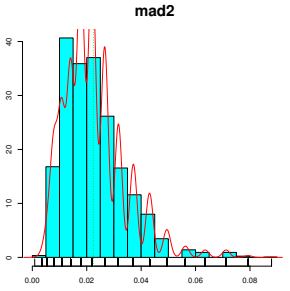

mean= 0.031767 , sd= 0.0084592

zouka-sd2



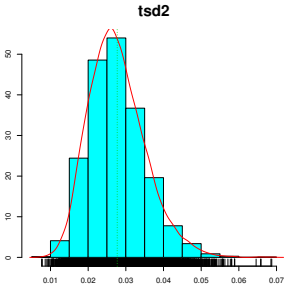

mean= 0.021825 , sd= 0.0091092

zouka-iqr2




mean= 0.02242 , sd= 0.011232

zouka-mad2




mean= 0.027704 , sd= 0.0074724

zouka-tsd2

# 4 課題

## 4.1 課題 4-1

3.5 節を参考にして，以下を実行する．run0048.R，run0049.R 等を利用してよい．

- マウスクリックでデータサイズ $n = 100$ のデータを作成し，それを $x$ とする．データの山の個数や，ハズレ値を意識して作成すると良い．
- $x$ のヒストグラム（ラグ付）を示せ．
- 「中心」の推定値：平均 ($\bar{x}$)，メディアン，刈り込み平均 (刈り込み率=0.1)，$t$-分布の最尤推定 ($df = 5$) から得られる m を計算せよ．
- 「バラツキ」の推定値：標本標準偏差 ($S_x$)，四分位偏差 (IRQ)，MAD，$t$-分布の最尤推定 (自由度=5) から得られる s を計算せよ．いずれも正規分布の標準偏差に相当する修正を施したものを用いよ．(run0042.R の mymeansd 関数を利用すればよい．)
- $\bar{x}$ と $S_x$ の標準誤差（理論値から求めたプラグイン推定値）を示せ．
- ブートストラップ標本を $b = 10000$ 個生成せよ．
- 「中心」と「バラツキ」の８個の統計量について，そのブートストラップ複製のヒストグラム（ラグ付）を示せ．
- 「中心」と「バラツキ」の８個の統計量について，ブートストラップ複製から求めた標準誤差を示せ．
- データ作成時に意識した点，および，それがどのように統計量に反映されたかを述べよ．

## 4.2 課題 4-2

sample 関数を利用して，次のような関数を作成し，その関数定義と実行例を示せ．

1. cards 関数：１から１２までのカード（各４枚）をランダムにシャッフルして，順番に取り出したときの動作をシミュレーションする関数 cards を作成する．カードを取り出す枚数 n を引数とする．関数の返す値は，擬似乱数によってシミュレーションされた，カードの系列 ($n$ 次元のベクタ) である．たとえば cards(10) とすれば，１０枚のカードを取り出した結果を返す．
2. saikoro 関数：１から６までの目が等確率 (1/6) で出るサイコロの動作をシミュレーションする関数 saikoro を作成する．サイコロを振る回数 n を引数とする．関数の返す値は，擬似乱数によってシミュレーションされた，サイコロの出た目の系列 ($n$ 次元のベクタ) である．たとえば saikoro(10) とすれば，１０回サイコロを振った結果を返す．