Stat 4620: Day 13

(3/12)

1 Section 4.8 Monte Carlo Methods

Suppose $X_1, \ldots, X_n$ is a random sample from

1. $N(\mu, \sigma^2)$
2. t-distribution with 2 df
3. .75 N(0,1) + .25 N(0,25)

Which estimates perform better (e.g. have smaller variance)?

- mean versus median
- t-test versus Wilcoxon

When the theory is limited because estimates and test are more complex, then Monte Carlo methods provide an alternative to theoretical analyses.

Justification:
The Monte Carlo method aims to estimate a distribution (and hence, its parameters) by simulating it with random sampling.

Example: mean versus median

- Theory approach

  When $f(\cdot)$ is symmetric about μ, then $E(\overline{X}) = \mu$ and $V_f(\overline{X}) = \frac{1}{n}\int (x-\mu)^2 f(x)dx$. Also $E(\text{Med}) = \mu$ and $V_f(\text{Med}) = \frac{1}{4nf^2(0)}$.

- Monte Carlo approach

  1. Generate $x_1, \ldots, x_n$ from f
  2. Calculate $\overline{X}$, Med.
  3. Repeat say, 10,000 times. We get

  $$\overline{X}_{(1)}, \overline{X}_{(2)}, \ldots, \overline{X}_{(10000)} \quad (1)$$

  $$\text{Med}_{(1)}, \text{Med}_{(2)}, \ldots, \text{Med}_{(10000)} \quad (2)$$

  4. The histogram of (1) is an estimate of the distribution of the sample mean, and so the average and variance of (1) are estimates of $E(\overline{X})$ and $V(\overline{X})$, respectively. Similarly, the distribution of the sample median is estimated by the histogram of (2), so then the mean and variance of (2) are estimates of $E(\text{Med}X)$ and $V(\text{Med}X)$.

Ex. Compare variance of sample mean versus sample median:

```
> # GENERATE 21 OBSERVATIONS from N(0,25)
> rnorm(21, 0, 5)
 [1]   1.7520488 -10.5342269  -5.5744215   0.4318463  -7.2656240
 [6]   4.8900702  -8.0917928  -7.1690461   2.5454584  -2.0446031
[11]   3.2847050 -10.7551782  -2.6461101   2.9666078   0.9359345
[16]  -3.2420609  -4.1704664   8.7097997  -0.6876553  -4.3153376
[21]   1.4995644
> # GENERATE 21 by 10000 MATRIX of OBSERVATIONS from N(0,25)
> A1<-matrix(rnorm(21*10000,0,5), ncol=21)
> A1[1,]
 [1] -2.366766 -6.193805  3.733646 -1.681459  4.563442 -4.805903
 [7]  5.332507 -1.058404  3.864665  1.638579 -6.863568 -3.527249
[13] -4.297159 -1.041060 -1.049405  4.284408 -6.092839  4.900293
[19] -1.190889 -1.626793 -1.038477
> u1<-apply(A1,1,mean)        # CALCULATE MEAN OF EACH ROW
> length(u1)
[1] 10000
> head(u1,10)
 [1] -0.69124930 -1.75314372  1.35882932  1.59557218  0.42969832
 [6]  0.34690172  1.55288779  0.94511570 -1.67065778  0.03220971
> hist(u1)                    # DISTRIBUTION OF MEANS
> u2<-apply(A1,1,median)      # CALCULATE MEDIAN OF EACH ROW
> head(u2,10)
 [1] -1.0584039 -0.2608988  1.5384211  2.0834606  0.4493463
 [6]  0.1642841  0.6034383  0.6850637 -1.0776656 -0.6364295
> hist(u2)                    # DISTRIBUTION OF MEDIANS
> cbind(var(u1), var(u2))     # COMPARE THE VARIANCES OF 2 DIST'NS
         [,1]     [,2]
[1,] 1.183194 1.852354
>
> # NOTE THAT WE KNOW THE TRUE VALUES OF THESE MC VARIANCES
> 25/21
[1] 1.190476
> 1/(4*21*dnorm(0,0,5)^2)
[1] 1.869996
>
> # GENERATE 21 by 10000 MATRIX of OBSERVATIONS from t(3 df)
> A2<-matrix(rt(21*10000, df=3), ncol=21)
> v1<-apply(A2,1,mean)
> v2<-apply(A2,1,median)
> cbind(var(v1), var(v2))
          [,1]      [,2]
[1,] 0.1396077 0.0929398
>
```

```
> # COMPARE VARIANCES of mean, median, and midrange)=(min+max)/2
> u3<-(apply(A1,1,min)+apply(A1,1,max))/2      # CALCULATE MIDRANGE: N(0,25) data
> cbind(var(u1), var(u2), var(u3))
         [,1]     [,2]     [,3]
[1,] 1.183194 1.852354 3.482622
> v3<-(apply(A2,1,min)+apply(A2,1,max))/2      # CALCULATE MIDRANGE: t(3) data
> cbind(var(v1), var(v2), var(v3))
          [,1]      [,2]     [,3]
[1,] 0.1396077 0.0929398 3.331376
> head(cbind(apply(A2,1,min), apply(A2,1,max), v3), 20)
                                    v3
 [1,] -3.469538 4.264274  0.39736784
 [2,] -4.315455 2.931515 -0.69196976
 [3,] -4.015488 2.497757 -0.75886547
 [4,] -1.619958 2.231435  0.30573835
 [5,] -1.462167 4.964913  1.75137273
 [6,] -4.062752 4.407131  0.17218947
 [7,] -3.140452 4.153518  0.50653299
 [8,] -1.081613 3.520287  1.21933720
 [9,] -1.319669 3.179154  0.92974252
[10,] -3.024387 2.049439 -0.48747392
[11,] -1.794011 1.339674 -0.22716828
[12,] -2.084250 1.832133 -0.12605840
[13,] -2.580364 2.767060  0.09334809
[14,] -2.077733 2.216724  0.06949518
[15,] -2.119251 3.117655  0.49920212
[16,] -1.721693 2.367183  0.32274492
[17,] -2.882547 3.359477  0.23846463
[18,] -2.475485 4.190855  0.85768488
[19,] -1.387333 4.314093  1.46337970
[20,] -1.063153 1.853816  0.39533184
```

Comments:

- In the simulation from $N(0,1)$, we know that
  - $V(\overline{X}) = \frac{\sigma^2}{n} = \frac{1}{20} = .05$
  - $V(\text{Med}) \doteq \frac{1}{4nf^2(0)} = \frac{\pi}{40} = .0785$
  - $V(Y_1 + Y_n)/2) = ?$

  so the variance of more complex statistics are analytically harder to get
- In Monte Carlo, the variance of all statistics are equally simple to estimate *as long as they can be computed from data*.

- The Monte Carlo estimates of variance require knowing the underlying distribution, e.g. $N(0,1)$ or $t(2 \text{ df})$. In the section on the bootstrap, we will estimate the underlying distribution with the sample-based empirical distribution.

1.1 How to generate observations from f

We can generate observations using R provided the distribution is in the built-in library, like $N(\mu, \sigma^2$ or Binomial(n, p). But how do we generate observations from, say

1. $f_1(x) = 3x^2, \;\; 0 < x < 1$
2. $f_2(x) = |x|/4, \;\; -2 < x < 2$

The following theorem tells us how.

Theorem 1. *Suppose that U has a Uniform distribution on (0,1). Let $F(\cdot)$ be a valid continuous cdf. Then the transformed random variable*

$$X = F^{-1}(U)$$

has cdf F.

Proof.

$$P[X \leq t] = P[F^{-1}(U) \leq t] = P[U \leq F(t)] = F(t)$$

□

Example:
Using R, generate $n = 1000$ observations from $f_1(x) = 3x^2, \;\; 0 < x < 1$.

Solution. The theorem says we should generate $n = 1000$ observations from $U(0,1)$, then apply the transformation F^{-1}. Since $F(x) = x^3$ then

$$y = F(x) = x^3$$

implies that

$$x = y^{1/3} = F^{-1}(y)$$

```
> x<-runif(100)
> y<-x^{1/3}
> head(cbind(x,y), 10)
              x         y
 [1,] 0.6077556 0.8470512
 [2,] 0.5225812 0.8054735
 [3,] 0.5530387 0.8208274
 [4,] 0.2955492 0.6661059
 [5,] 0.7418665 0.9052640
 [6,] 0.3523041 0.7062729
 [7,] 0.8090632 0.9318103
 [8,] 0.9779602 0.9925987
```

```
 [9,] 0.2502251 0.6301495
[10,] 0.3064387 0.6741883
> t<-seq(.1, 1.0, by=.1)
> sto1<-foreach(a=1:length(t),.combine='rbind') %do% c(t[a],mean(x<t[a]),mean(y<t[a]))
> sto1
          [,1] [,2] [,3]
result.1   0.1 0.11 0.00
result.2   0.2 0.17 0.00
result.3   0.3 0.26 0.02
result.4   0.4 0.36 0.07
result.5   0.5 0.50 0.12
result.6   0.6 0.62 0.19
result.7   0.7 0.72 0.31
result.8   0.8 0.79 0.51
result.9   0.9 0.95 0.73
result.10  1.0 1.00 1.00
```

■