

Other estimators of the expected value

Given data X_1, \dots, X_n , the sample mean \bar{X} is an estimator of the population mean EX .

The sample mean is not the only possible estimate of the population mean. There are other ways of combining the information in the X_i that may be used as alternative estimators.

In certain situations, the sample mean is not the best performing estimator of the expected value.

Trimmed mean: To calculate the “trimmed mean,” let k be an integer between 0 and n . Then remove the k largest and k smallest X_i values and average the rest.

If X is a vector in R , the trimmed mean can be calculated using

```
Y <- sort(X)
n <- length(X)
TM <- mean(Y[(k+1):(n-k)])
```

Before we can compare the trimmed mean to the regular mean, we need an objective way to compare performances of different estimators.

Suppose we wish to estimate a quantity θ based on data X_1, \dots, X_n . For example, θ could be the expected value. What are the important properties of an estimate $\hat{\theta}$ of θ ?

Bias: Bias measures whether over many replications, the estimator yields results that are correct on average.

Positive bias means the estimator is too large on average compared to the true value. Negative bias means that the estimator is too small on average compared to the true value.

The following simulation approximates the bias of the trimmed mean for various value of k . The simulation is carried out for data sets of 20 *iid* standard normal values.

```
B <- NULL

for (k in c(0,1,2,3,4,5))
{
  X <- array(rnorm(20*1000), c(20,1000))
  X <- apply(X, 2, sort)
  TX <- X[(k+1):(20-k),]
  TM <- apply(TX, 2, mean)
  B <- c(B, mean(TM))
}
```

The true value in this case is $\theta = EX = 0$. If you run the program several times, you will see that the numbers in B are small, and fluctuate between positive and negative values. Thus the trimmed mean does not appear to be biased for data sets of 20 standard normal values. You should try other sample sizes and other distributions to see whether this is true more generally.

Mean Squared Error: Another important characteristic of an estimator is the *mean squared error*.

$$\text{MSE} = E(\hat{\theta} - \theta)^2.$$

First, from a notational point of view, it is important to recognize that $E(\hat{\theta} - \theta)^2$ is equivalent to $E((\hat{\theta} - \theta)^2)$, not to $(E(\hat{\theta} - \theta))^2$.

The MSE is a direct measure of the error. The square is taken so that positive and negative errors don't cancel out. To get an error estimate on the same scale as the data, use the *root mean squared error* (RMSE), which is the square root of the MSE.

$$\text{RMSE} = \sqrt{E(\hat{\theta} - \theta)^2}.$$

The following simulation approximates the MSE and RMSE for the trimmed mean applied to standard exponential datasets of size 20. Trimmed means using values of k between 0 and 5 are considered.

```
MSE <- NULL

for (k in c(0,1,2,3,4,5))
{
  X <- array(rexp(20*1000), c(20,1000))
  X <- apply(X, 2, sort)
  TX <- X[(k+1):(20-k),]
  TM <- apply(TX, 2, mean)
  MSE <- c(MSE, mean((TM - 1)^2))
}

RMSE <- sqrt(MSE)
```

Based on this simulation, which estimator appears to perform best?

Contaminated normal distribution: Real data often aren't as clean as simulated normal or exponential data. The contaminated normal distribution is a model for data in which a fixed proportion α of outliers are present.

To simulate a contaminated normal draw, we select from one of two normal populations at random. Specifically, generate a $N(0, \tau^2)$ draw with probability α and a $N(0, \sigma^2)$ draw with probability $1 - \alpha$. The variances are chosen so that $\sigma^2 \ll \tau^2$ (where \ll means "much less than"), e.g., $\tau^2 = 100\sigma^2$. The value of α is usually small, say 0.01 or 0.05.

The following program simulates a 50×1000 array from a contaminated normal distribution.

```

## The expected proportion of draws from component 1.
alpha <- 0.05

## The variance of component 1.
tau2 <- 100

## The variance of component 2.
sig2 <- 1

## A 50x1000 array of iid true/false values in which each entry has
## probability alpha of being true.
A <- array((runif(50*1000) < alpha), c(50, 1000))

## The component 1 draws.
B <- array(sqrt(tau2)*rnorm(50*1000), c(50, 1000))

## The component 2 draws.
C <- array(sqrt(sig2)*rnorm(50*1000), c(50, 1000))

## These are the contaminated normal draws.
X <- A*B + (1-A)*C

```

Here are a few things to note about this program:

- To simulate a list of n draws from a normal distribution with variance s and expected value zero, use `sqrt(s)*rnorm(n)`.
- Although each element of X is generated either from component 1 or component 2, it is easier and faster to generate both the component 1 and component 2 draw for each element of X . The array A is used to select between them, where if A is “true” then $1-A$ is false, and vice versa. This works since “false” times any number x is zero and “true” times any number x is x .

The following program combines the previous two programs to investigate the

MSE of the trimmed mean for the contaminated normal distribution.

```

## The expected proportion of draws from component 1.
alpha <- 0.05

## The variance of component 1.
tau2 <- 100

## The variance of component 2.
sig2 <- 1

## A 50x1000 array of iid true/false values in which each entry has
## probability alpha of being true.
A <- array((runif(50*1000) < alpha), c(50, 1000))

## The component 1 draws.
B <- array(sqrt(tau2)*rnorm(50*1000), c(50, 1000))

## The component 2 draws.
C <- array(sqrt(sig2)*rnorm(50*1000), c(50, 1000))

## These are the contaminated normal draws.
X <- A*B + (1-A)*C

## Sort the columns.
X <- apply(X, 2, sort)

## Storage for the trimmed mean MSE's.
MSE <- NULL

## Use trimmed means for a range of values of k.
for (k in (0:20))
{
  TX <- X[(k+1):(50-k),]
  TM <- apply(TX, 2, mean)
  MSE <- c(MSE, mean(TM^2))
}

RMSE <- sqrt(MSE)

```

Run this program and make a plot of the MSE values. Which is the best estimator? Can you explain why? Compare the results here to what you found when using the trimmed mean to estimate the expected value of the exponential distribution.

Sample median: For a symmetric distribution, the median is an unbiased estimate of the expected value. It's variance is either greater than or less than that of the sample mean, depending on the distribution.

For a non-symmetric distribution, the sample median is a biased estimate of the expected value. However it still may have lower MSE than the sample mean, especially if the distribution has a lot of outliers.

Clustered data: Suppose we observe data in groups, where it is possible to acquire the same measurement for individuals within each group. Every individual has a unique measurement, but individuals in a common group tend to be more similar than individuals in different groups. Examples would be student test scores, which are grouped by school, or patient survival rates, which may be grouped by hospital.

In cases like this, it often happens that there is too little data within each cluster (i.e. per-school or per-hospital) to provide good estimates of the cluster-level means. In this case it makes sense to recall that the clusters may be informative about each other.

Suppose we are interested in the expected value within each cluster. A practical way to proceed is to begin with the cluster level sample means

$$\bar{X}_1, \dots, \bar{X}_m,$$

each of which is an unbiased estimate of the corresponding cluster population mean

$$\mu_1, \dots, \mu_m.$$

We then consider the overall sample mean \bar{X} . We can consider whether shrinking the sample cluster sample means toward \bar{X} improves their performance as estimators. A straightforward way to do the shrinking is to use

$$\hat{\mu}_j = \bar{X} + \lambda(\bar{X}_j - \bar{X})$$

as an estimate of the expected value in cluster j , where $\lambda \geq 0$ is a specified shrinking factor. Note that $\lambda = 0$ shrinks everything to the same value, and $\lambda = 1$ does no shrinkage.

There are many ways to set the value of λ to attempt to optimize performance. Here we will do a simulation study to see the performance in terms of MSE for different value of λ . Note that the estimators $\hat{\mu}_j$ are biased unless $\lambda = 1$.

```
## Number of clusters.
nc <- 100

## Number of observations per cluster.
cs <- 5

## Shrinkage factors.
F <- c(0.5, 0.6, 0.7, 0.8, 0.9, 1)

## MSE results for each shrinkage factor.
MSE <- c(0,0,0,0,0,0)

## Simulation replications.
for (r in 1:100)
{
  ## True cluster means.
  CM <- rnorm(nc)

  ## Estimated cluster means.
  Z <- array(0, nc)

  for (k in 1:nc)
  {
    z <- CM[k] + rnorm(cs)
    Z[k] <- mean(z)
  }
}
```

```

}

## The 'grand mean'
GM <- mean(Z)

for (k in 1:length(F))
{
  ## Modify the cluster mean estimates by shrinking toward
  ## the grand mean.
  S <- GM + F[k]*(Z-GM)

  ## Update the MSE.
  MSE[k] <- MSE[k] + mean((S-CM)^2)
}
}

MSE <- MSE / 100
RMSE <- sqrt(MSE)

```

Bias/Variance tradeoff. If $\hat{\theta}$ is an estimate of θ , then it is a fact that

$$\text{MSE}(\hat{\theta}) = \text{bias}(\hat{\theta})^2 + \text{var}(\hat{\theta}).$$

The following simulation studies the bias/variance tradeoff in the clustered data setting discussed above. Here we focus on estimation of the first cluster mean for simplicity.

```

## Number of clusters.
nc <- 100

## Number of observations per cluster.
cs <- 5

## Shrinkage factors.
F <- c(0.5, 0.6, 0.7, 0.8, 0.9, 1)

```

```

## Save all the estimates.
Estimate <- array(0, c(1000,6))

## True cluster means.
CM <- rnorm(nc)

## Simulation replications.
for (r in 1:1000)
{
  ## Estimated cluster means.
  Z <- array(0, nc)
  for (k in 1:nc)
  {
    z <- CM[k] + rnorm(cs)
    Z[k] <- mean(z)
  }

  ## The 'grand mean'
  GM <- mean(Z)

  for (k in 1:length(F))
  {
    ## Modify the cluster 1 mean estimate by shrinking toward
    ## the grand mean.
    S <- GM + F[k]*(Z[1]-GM)

    Estimate[r,k] <- S
  }
}

Bias <- apply(Estimate, 2, mean) - CM[1]
Var <- apply(Estimate, 2, var)
MSE <- apply((Estimate-CM[1])^2, 2, mean)

```

- What value of F give the least bias?
- What value of F give the least variance?
- What value of F give the least MSE?
- Confirm that the formula relating MSE to bias and variance holds.