Relative Efficiency of $\tilde{X}$ and $\bar{X}$

```
> means <- medians <- numeric(0)  # make empty vectors
> for (i in 1:1000) {
+   x <- rnorm(30,mean=10,sd=5)
+   means[i] <- mean(x)
+   medians[i] <- median(x)
+ }
> var(means)
[1] 0.8111457
> var(medians)
[1] 1.228001
> var(means)/var(medians)  # simulated RE, normal case
[1] 0.6605414
> for (i in 1:1000) {
+   x <- 10+5*rt(30,df=5)
+   means[i] <- mean(x)
+   medians[i] <- median(x)
+ }
> var(means)
[1] 1.350055
> var(medians)
[1] 1.319712
> var(means)/var(medians)  # simulated RE, t(5) case
[1] 1.022992
>
```


Trimmed Mean

```
> x <- 10+5*rt(70,df=3)  # sample size 70
> mean(x)                # default trim = 0
[1] 11.09950
> mean(x,trim=0)
[1] 11.09950
> mean(x,trim=0.05)      # 5% off each end
[1] 10.34636
> 70*0.05
[1] 3.5                  # that is, 3 obs off each end
> mean(sort(x)[4:67])
[1] 10.34636
> mean(x,trim=0.5)       # trimming off everything...
[1] 10.18221
> median(x)              # ...gives the median
[1] 10.18221
>
```


Efficiency of Trimmed Mean

```
> trims <- seq(0,0.5,length=10)
> trimmeans <- matrix(NA,nrow=1000,ncol=10)
> for (i in 1:1000) {
+   x <- rnorm(30,mean=10,sd=5)
+   for (j in 1:10) {
+     trimmeans[i,j] <- mean(x,trim=trims[j])
+   }
+ }
> trims
 [1] 0.0000000 0.0555556 0.1111111 0.1666667 0.2222222
 [6] 0.2777778 0.3333333 0.3888889 0.4444444 0.5000000
> (vars <- apply(trimmeans,2,var))
 [1] 0.8272082 0.8308127 0.8708342 0.9230062 0.9517421
 [6] 1.0231556 1.0956334 1.1251648 1.1916106 1.2344185
> vars[1]/vars
 [1] 1.0000000 0.9956615 0.9499032 0.8962109 0.8691516
 [6] 0.8084872 0.7550045 0.7351885 0.6941934 0.6701197

[ repeat with x <- 10+5*rt(30,df=5) ]
> vars[1]/vars
 [1] 1.0000000 1.1433810 1.2279008 1.2284159 1.2215654
 [6] 1.1857230 1.1303785 1.0970575 1.0152993 0.9801351

[ repeat with x <- rnorm(30,mean=10,
           sd=ifelse(rbinom(30,1,.05),10,5)) ]
> vars[1]/vars
 [1] 1.0000000 1.0318539 1.0208543 0.9858267 0.9587991
 [6] 0.8952624 0.8441214 0.8151472 0.7547057 0.7171817
>
```


Robust Scale Estimates

- Some scale estimates:

```
> attach(Traffic)
> y.yes <- y[limit=="yes"]; y.no <- y[limit=="no"]
> c(sd(y.yes),sd(y.no))
[1] 7.474937 9.157991
> c(mad(y.yes),mad(y.no))
[1] 7.4130 8.8956
> c(IQR(y.yes)/1.349,IQR(y.no)/1.349)
[1] 6.671609 8.895478
>
```

- Simulated relative efficiency (under normality):

```
> mads <- numeric(0)
> iqrs <- numeric(0)
> sds  <- numeric(0)
> for (i in 1:1000) {
+   x <- rnorm(30,mean=10,sd=5)
+   mads[i] <- mad(x)
+   iqrs[i] <- IQR(x)/1.349
+   sds[i]  <- sd(x)
+ }
> var(sds)/var(mads)
[1] 0.3784513
> var(sds)/var(iqrs)
[1] 0.3923162
>
```

Huber's Estimate

```
> x <- 10+5*rt(70,df=3)
> huber(x,k=10000)
$mu
[1] 11.09950
$s
[1] 6.260165
> mean(x)
[1] 11.09950
> huber(x,k=0)
$mu
[1] 10.18221
$s
[1] 6.260165
> median(x)
[1] 10.18221
> huber(x)           # default k=1.5
$mu
[1] 10.38661
$s
[1] 6.260165
>
```


Huber's Proposal 2

```
> x <- rnorm(20,mean=10,sd=5)
> hubers(x,k=10000)
$mu
[1] 8.769674
$s
[1] 4.816868
> mean(x)
[1] 8.769674
> sd(x)
[1] 4.816868
> hubers(x,k=0.00001)
$mu
[1] 9.667696
$s
[1] 21120.81
> median(x)
[1] 9.667696
> hubers(x)          # default k=1.5
$mu
[1] 8.773504
$s
[1] 5.451605
>
```


Efficiency of Proposal 2

```
> means <- numeric(0)
> sds <- numeric(0)
> hubers.means <- numeric(0)
> hubers.sds <- numeric(0)
> for (i in 1:1000) {
+   x <- rnorm(25,mean=30,sd=10)
+   means[i] <- mean(x)
+   sds[i] <- sd(x)
+   huber1 <- hubers(x)
+   hubers.means[i] <- huber1$mu
+   hubers.sds[i] <- huber1$s
+ }
> var(means)/var(hubers.means)
[1] 0.969263
> var(sds)/var(hubers.sds)
[1] 0.7299158

[ repeat with x <- 30+10*rt(25,df=5) ]
> var(means)/var(hubers.means)
[1] 1.213951
> var(sds)/var(hubers.sds)
[1] 2.442511
>
```