homework4

February 22, 2019

```
In [6]: %pylab inline
        import scipy.stats as stats
        import astropy.stats as astats
        import numpy.random as random
        from hlmean import hlmean
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
In [7]: def mode2(data,**kwargs):
            counts,edges=np.histogram(data,**kwargs)
            whmax=np.argmax(counts)
            mode=(edges[whmax]+edges[whmax+1])/2
            return(mode)
```

```
In [8]: def two_normal(nsims,ndata,foutlier):
            isoutlier=random.rand(nsims,ndata)<foutlier
            return (1-isoutlier)*(random.randn(nsims,ndata)*930.+3150)\
                +(isoutlier)*(random.randn(nsims,ndata)*200.+4750)
```

```
In [36]: def part1(nsims,ndata,foutlier):
             fakedata=two_normal(nsims,ndata,foutlier)

             means=np.mean(fakedata,axis=1)
             medians=np.median(fakedata,axis=1)
             modes=[mode2(fakedata[i,:],bins=np.arange(np.min(fakedata[i,:]),\
                                                      np.max(fakedata),50))\
                                                        for i in range(nsims)]
             hlmeans=[hlmean(fakedata[i,:]) for i in range(nsims)]

             tmeans=[stats.tmean(fakedata[i,:],limits=np.percentile(fakedata[i,:],(10,90))) for
             biweight_locations=astats.biweight_location(fakedata,axis=1)

             print("ndata=",ndata)
             print("foutlier=",foutlier)
             print("nsims=",nsims)

             print('mean: bias {:.4f}, spread {:.4f}'.\
```

```
            format(np.mean(means)-3150, np.std(means,ddof=1)))
        print('median: bias {:.4f}, spread {:.4f}'.\
            format(np.mean(medians)-3150, np.std(medians,ddof=1)))
        print('mode: bias {:.4f}, spread {:.4f}'.\
            format(np.mean(modes)-3150, np.std(modes,ddof=1)))
        print('hlmean: bias {:.4f}, spread {:.4f}'.\
            format(np.mean(hlmeans)-3150, np.std(hlmeans,ddof=1)))
        print('10%trimmed_mean: bias {:.4f}, spread {:.4f}'.\
            format(np.mean(tmeans)-3150, np.std(tmeans,ddof=1)))
        print('biweight_location: bias {:.4f}, spread {:.4f}'.\
            format(np.mean(biweight_locations)-3150, \
                   np.std(biweight_locations,ddof=1)))

        print("\n")
```

```
In [37]: nsims=int(5E4)
         ndata=100
         foutlier=0
         part1(nsims,ndata,foutlier)
```

```
ndata= 100
foutlier= 0
nsims= 50000
mean: bias -0.8182, spread 92.7816
median: bias -0.5974, spread 114.9933
mode: bias -174.2087, spread 486.5738
hlmean: bias -0.8245, spread 95.7263
10%trimmed_mean: bias -0.7259, spread 95.3357
biweight_location: bias -0.6786, spread 98.9063
```

Except the mode, the other four estimators have nearly the same bias. The mean's spread is the lowest. I will choose the mean.

```
In [46]: nsims=int(5E4)
         ndata=[5,25,100]
         foutlier=0.1

         for i,n in enumerate(ndata):
             part1(nsims,n,foutlier)
```

```
ndata= 5
foutlier= 0.1
nsims= 50000
mean: bias 157.0843, spread 449.1158
median: bias 152.3320, spread 571.5627
mode: bias -863.6486, spread 777.8797
```

```
hlmean: bias 157.3238, spread 480.1949
10%trimmed_mean: bias 158.8326, spread 500.6389
biweight_location: bias 156.5743, spread 544.1693


ndata= 25
foutlier= 0.1
nsims= 50000
mean: bias 160.1378, spread 201.5434
median: bias 135.2752, spread 262.5613
mode: bias -330.9928, spread 764.0954
hlmean: bias 162.0902, spread 216.1344
10%trimmed_mean: bias 165.8871, spread 220.4917
biweight_location: bias 154.5225, spread 226.6276


ndata= 100
foutlier= 0.1
nsims= 50000
mean: bias 159.7944, spread 100.5697
median: bias 130.9527, spread 130.0742
mode: bias -92.3624, spread 608.5616
hlmean: bias 161.7797, spread 107.9801
10%trimmed_mean: bias 169.0696, spread 109.4886
biweight_location: bias 156.9253, spread 111.1472
```

n=5, the mean estimator's bias is resonable and its spread is the lowest. So the mean estimator will be my choice.

n=25, the mode estimator has both the highest bias and spread. I won't choose the mode. The other four estimators have close bias and spreads. They are all ok.

n=100, the mode estimator has the lowest bias, but the spread is to high. The other four estimators have close bias and spreads.

```
In [39]: def normal_uniform(nsims,ndata,foutlier):
             isoutlier=random.rand(nsims,ndata)<foutlier
             return (1-isoutlier)*(random.randn(nsims,ndata)*930.+3150)\
                 +(isoutlier)*(random.rand(nsims,ndata)*6500)

In [40]: def mean_AD(data):
             meanabsdev = np.mean(np.abs(data-data.mean()))/0.7979
             return meanabsdev

In [41]: def MAD(data):
             meddata=np.median(data)
             mad = np.median(np.abs(data-meddata))/0.6745
             return mad
```

```
In [42]: def IQR(data):
             d25,d75 = np.percentile(data,[25,75])
             iqr = (d75-d25)/1.349
             return iqr

In [43]: def part2(nsims,ndata,foutlier):
             fakedata=normal_uniform(nsims,ndata,foutlier)

             stds=np.std(fakedata,axis=1)
             mean_ADs=[mean_AD(fakedata[i,:]) for i in range(nsims)]
             MADs=[MAD(fakedata[i,:]) for i in range(nsims)]
             biweight_stds=np.sqrt(astats.biweight_midvariance(fakedata,axis=1))
             IQRs=[IQR(fakedata[i,:]) for i in range(nsims)]
             trimmed_stds=[stats.tstd(fakedata[i,:],limits=np.percentile(fakedata[i,:],(10,90)))

             print("ndata=",ndata)
             print("foutlier=",foutlier)
             print("nsims=",nsims)

             print('std: bias {:.4f}, spread {:.4f}'.\
               format(np.mean(stds)-930, np.std(stds,ddof=1)))
             print('mean_AD: bias {:.4f}, spread {:.4f}'.\
               format(np.mean(mean_ADs)-930, np.std(mean_ADs,ddof=1)))
             print('MAD: bias {:.4f}, spread {:.4f}'.\
               format(np.mean(MADs)-930, np.std(MADs,ddof=1)))
             print('biweight_std: bias {:.4f}, spread {:.4f}'.\
               format(np.mean(biweight_stds)-930, np.std(biweight_stds,ddof=1)))
             print('IQR: bias {:.4f}, spread {:.4f}'.\
               format(np.mean(IQRs)-930, np.std(IQRs,ddof=1)))
             print('10%trimmed_stds: bia {:.4f}, spread {:.4f}'.\
               format(np.mean(trimmed_stds)-930, np.std(trimmed_stds,ddof=1)))



             print("\n")

In [44]: nsims=int(5E4)
         ndata=100
         foutlier=0
         part2(nsims,ndata,foutlier)

ndata= 100
foutlier= 0
nsims= 50000
std: bias -7.1229, spread 66.0673
mean_AD: bias -4.9478, spread 70.5724
MAD: bias -7.4340, spread 108.0390
biweight_std: bias 2.6057, spread 69.6419
```

```
IQR: bias -13.2744, spread 106.9901
10%trimmed_stds: bia -309.5255, spread 55.4822
```

biweight_std has the lowest bias, and its spread is resonable. It's the best.

```
In [47]: nsims=int(5E4)
         ndata=[5,25,100]
         foutlier=0.1

         for i,n in enumerate(ndata):
             part2(nsims,n,foutlier)
```

```
ndata= 5
foutlier= 0.1
nsims= 50000
std: bias -41.2157, spread 343.8840
mean_AD: bias 10.3543, spread 369.4831
MAD: bias -81.0628, spread 513.6137
biweight_std: bias -77.0517, spread 411.2572
IQR: bias -177.0283, spread 444.5479
10%trimmed_stds: bia -396.5058, spread 314.9349


ndata= 25
foutlier= 0.1
nsims= 50000
std: bias 98.8692, spread 161.7287
mean_AD: bias 91.0941, spread 166.0684
MAD: bias 44.9981, spread 235.9597
biweight_std: bias 94.1622, spread 174.0640
IQR: bias 19.0263, spread 229.1155
10%trimmed_stds: bia -282.2063, spread 124.7067


ndata= 100
foutlier= 0.1
nsims= 50000
std: bias 124.8741, spread 81.0965
mean_AD: bias 105.8536, spread 83.1515
MAD: bias 64.9665, spread 118.2558
biweight_std: bias 116.3016, spread 86.2166
IQR: bias 58.5502, spread 117.0988
10%trimmed_stds: bia -254.1592, spread 62.4100
```

n=5, the mean_AD's bias is appreantly smaller than the others', and its' spread is relatively low. I choose the mean_AD.

n=25, the IQR has the lowest bias and its spread is not too large. I choose the IQR.

n=100, the IQR has the lowest bias and its spread is not too large. I choose the IQR.