Part1_code

Francisco Mauro, Temesgen Hailemariam & Bryce Frank

Introduction

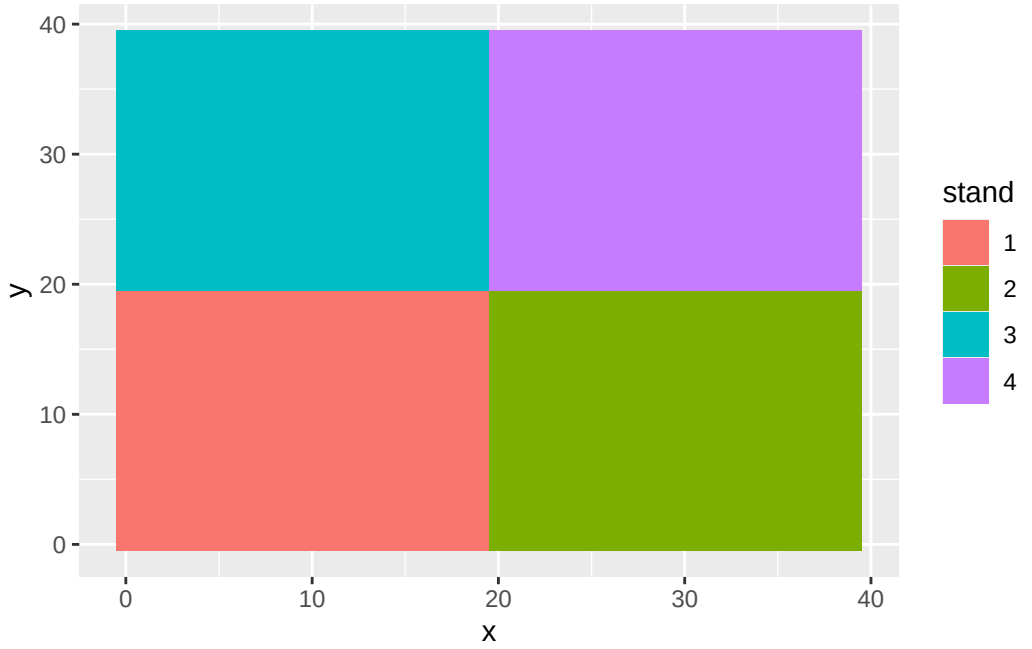
This markdown document covers the principal concepts of regression estimators and introduces unit level models. The code section concludes introducing the unit level model.

Regression estimator

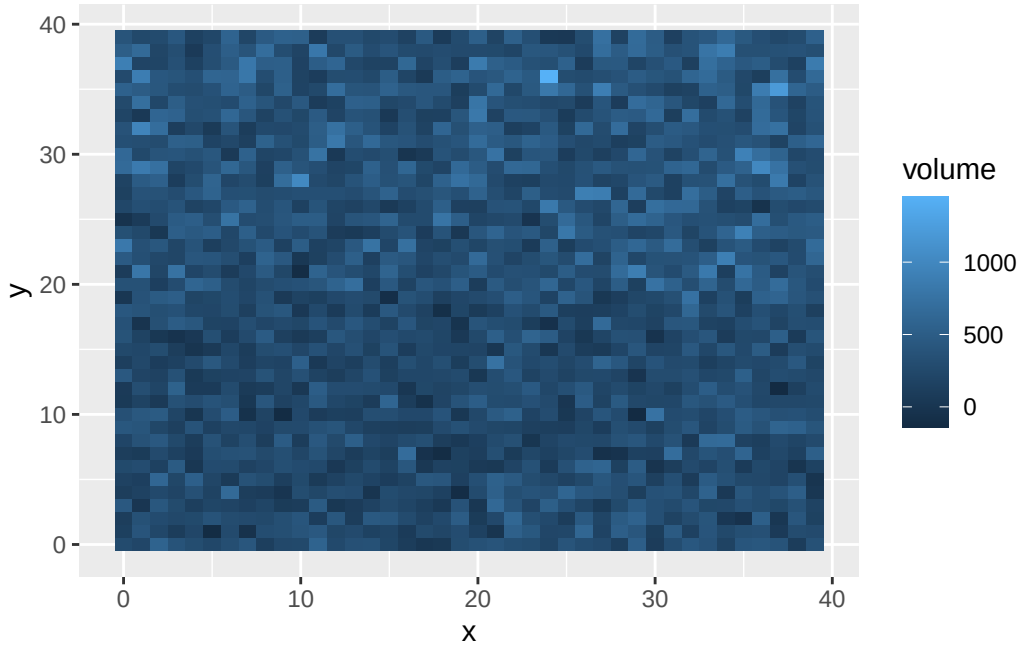
We are going to compare regression estimators with the traditional sample mean for an artificial population that we will use throughout the examples. In this example we have the value of volume for all grid units in the forest along with an auxiliary variable called p95. **THIS NEVER HAPPENS IN REAL LIFE** but will help us understand regression estimators. We will mimic simple random sampling and compute the normal expansion estimator (sample mean) to a regression estimator.

Our population mimics a square tract of forest of 160 acres with four stands 40 of acres each one. Let's take a look at the data.

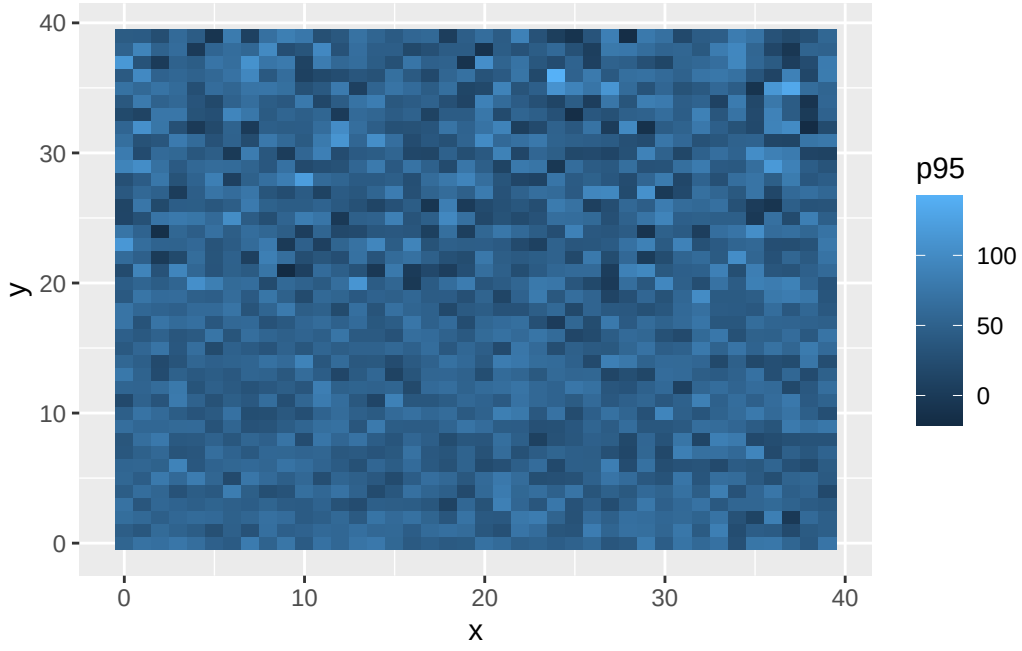
```
library(ggplot2)
example1 <- read.csv("example1.csv")

#PLACEHOLDER
# paste here



example1$stand <- factor(example1$stand)
ggplot(example1,aes(x=x,y=y,fill=stand)) + geom_tile()
```



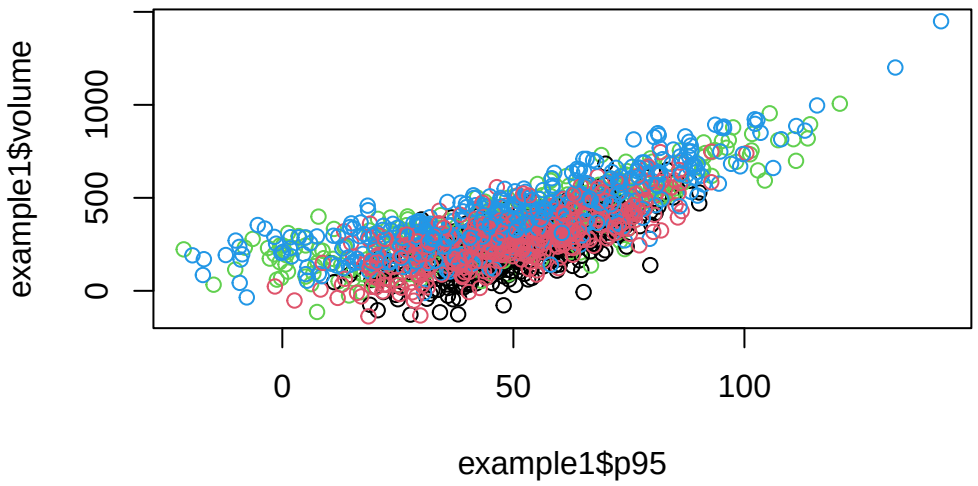

```
ggplot(example1,aes(x=x,y=y,fill=volume)) + geom_tile()
```



```
ggplot(example1,aes(x=x,y=y,fill=p95)) + geom_tile()
```




```
plot(example1$p95,example1$volume,col=example1$stand)
```



We will start estimating the average volume in the forest and in the mean volume in each stand doing simple random sampling without replacement. Having the volume of all grid units allows us to obtain the true average volume in the forest.

```
true_mean_forest <- mean(example1$volume)
true_mean_stand1 <- mean(example1$volume[example1$stand==1])
true_mean_stand2 <- mean(example1$volume[example1$stand==2])
true_mean_stand3 <- mean(example1$volume[example1$stand==3])
true_mean_stand4 <- mean(example1$volume[example1$stand==4])
print("True mean")
```

```
[1] "True mean"
```

```
true_mean_forest
```

```
[1] 322.715
```

```
print("True mean stand 1")
```

```
[1] "True mean stand 1"
```

```
  true_mean_stand1
```

```
[1] 234.6624
```

```
  print("True mean stand 2")
```

```
[1] "True mean stand 2"
```

```
  true_mean_stand2
```

```
[1] 286.2552
```

```
  print("True mean stand 3")
```

```
[1] "True mean stand 3"
```

```
  true_mean_stand3
```

```
[1] 357.4505
```

```
  print("True mean stand 4")
```

```
[1] "True mean stand 4"
```

```
  true_mean_stand4
```

```
[1] 412.4917
```

```
  range_volume <- range(example1$volume)
```

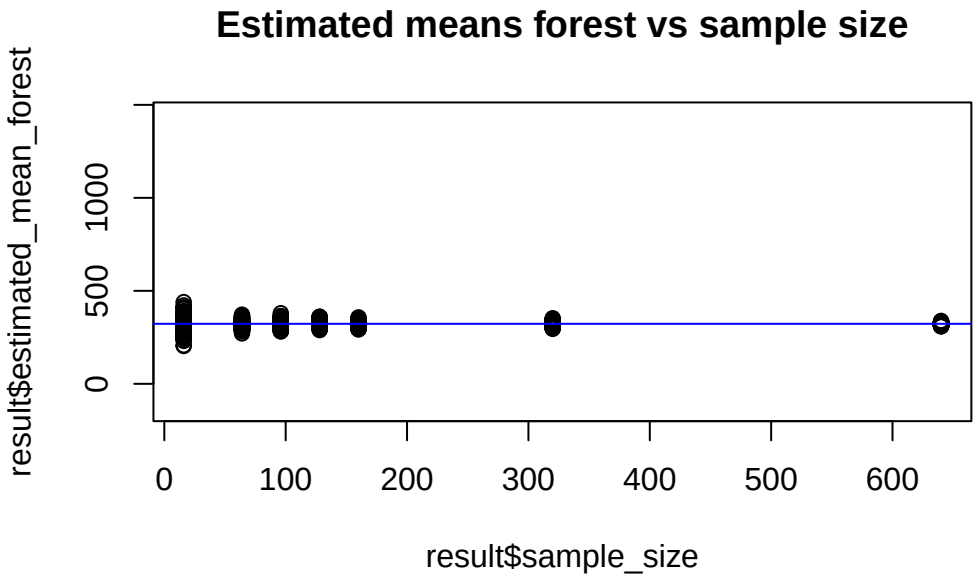
Sample mean with simple random sampling

Let's simulate our simple random sampling. We will repeat the simple random sampling 100 with sample sizes of 16, 64, 96, 128, 160, 320 and 640 and plot our estimates to visualize the variance of doing simple random sampling. We will store the result of each iteration in a data.frame with our estimate, the sample size and the iteration.

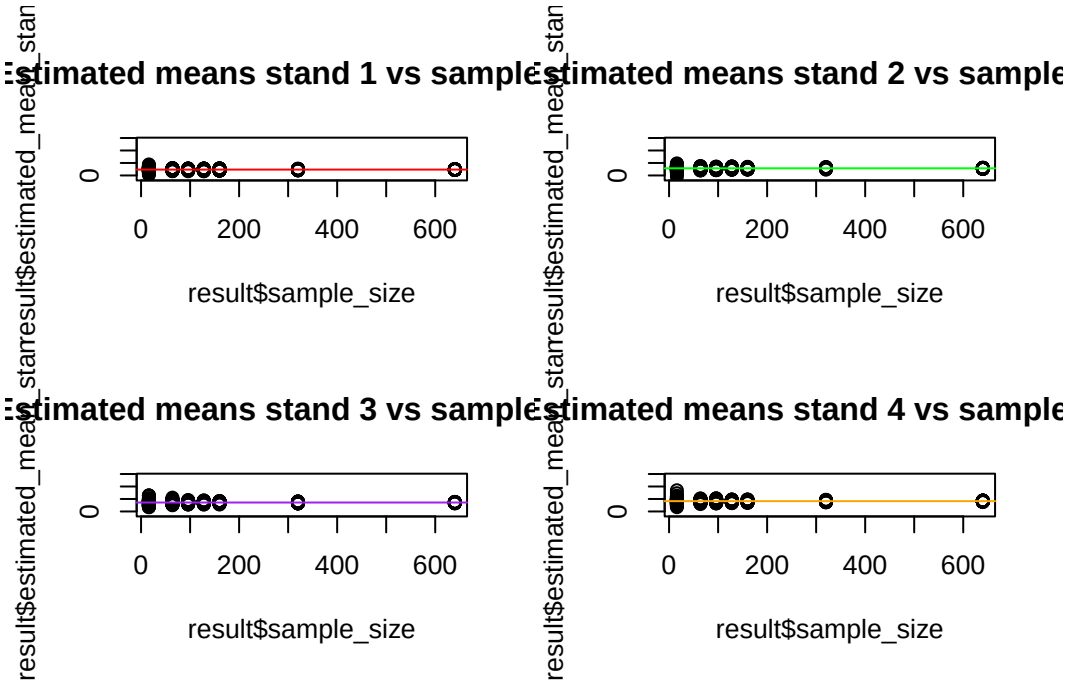
```
repeats <- 100
sample_sizes <- c(16, 64, 96, 128, 160, 320, 640 )
result <- data.frame(rep=c(),sample_size=c(),
                     estimated_mean_forest=c(),
                     estimated_mean_stand1 = c(),
                     estimated_mean_stand2 = c(),
                     estimated_mean_stand3 = c(),
                     estimated_mean_stand4=c())
cont <- 1
for(i in 1:repeats){
  for(j in sample_sizes){
    # get a sample of the indices of the data
    my_sample<- sample(dim(example1)[1],j,replace = FALSE)
    # subset the forest
    sampled_data <- example1[my_sample,]
    # store the rep and sample size
    result[cont,"rep"]<-i
    result[cont,"sample_size"]<-j
    # calculate the sample means for the forest and each stand
    result[cont,"estimated_mean_forest"]<-mean(sampled_data$volume)
    result[cont,"estimated_mean_stand1"]<-mean(sampled_data$volume[sampled_data$stand==1])
    result[cont,"estimated_mean_stand2"]<-mean(sampled_data$volume[sampled_data$stand==2])
    result[cont,"estimated_mean_stand3"]<-mean(sampled_data$volume[sampled_data$stand==3])
    result[cont,"estimated_mean_stand4"]<-mean(sampled_data$volume[sampled_data$stand==4])
    # pass to next iteration
    cont <- cont+1
  }
}
# Store the pre-calculated true values
result$true_mean_forest<-true_mean_forest
result$true_mean_stand1 <- true_mean_stand1
result$true_mean_stand2 <- true_mean_stand2
result$true_mean_stand3 <- true_mean_stand3
result$true_mean_stand4 <- true_mean_stand4
```

Let's now plot our results as a function of the sample size. We will simulate a precision requirement of +- 10% in our estimates piloting discontinuous lines around the true value

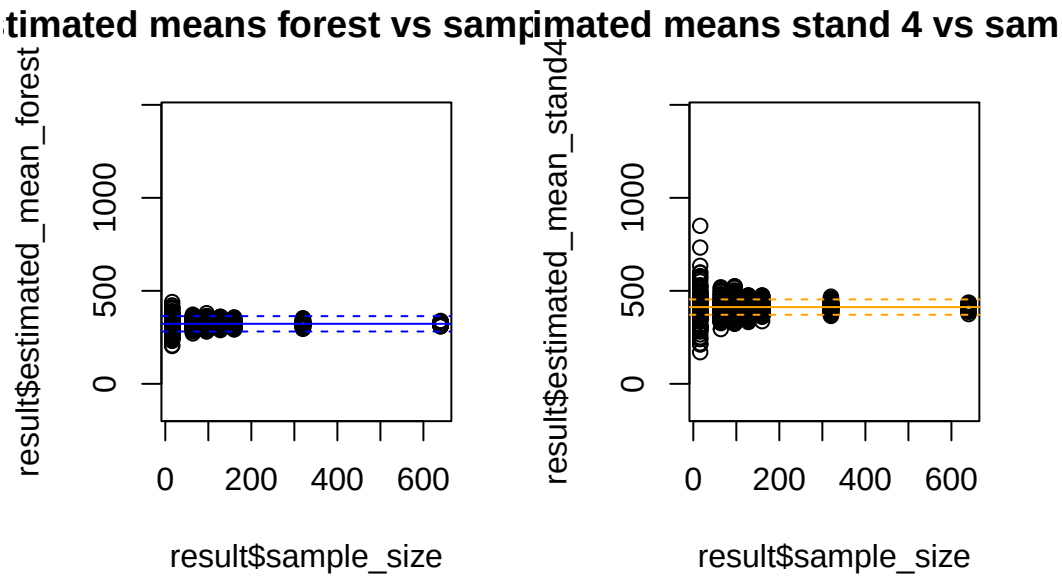
```
plot(result$sample_size,result$estimated_mean_forest,ylim=range_volume,
      main="Estimated means forest vs sample size")
abline(true_mean_forest,0,col="blue")
```




```
par(mfrow=c(2,2))
plot(result$sample_size,result$estimated_mean_stand1,ylim=range_volume,
      main="Estimated means stand 1 vs sample size")
abline(true_mean_stand1,0,col="red")
plot(result$sample_size,result$estimated_mean_stand2,ylim=range_volume,
      main="Estimated means stand 2 vs sample size")
abline(true_mean_stand2,0,col="green")
plot(result$sample_size,result$estimated_mean_stand3,ylim=range_volume,
      main="Estimated means stand 3 vs sample size")
abline(true_mean_stand3,0,col="purple")
plot(result$sample_size,result$estimated_mean_stand4,ylim=range_volume,
      main="Estimated means stand 4 vs sample size")
abline(true_mean_stand4,0,col="orange")
```




```
par(mfrow=c(1,2))
plot(result$sample_size,result$estimated_mean_forest,ylim=range_volume,
     main="Estimated means forest vs sample size")
abline(true_mean_forest,0,col="blue")
abline(true_mean_forest+0.1*true_mean_stand4,0,col="blue",lty=2)
abline(true_mean_forest-0.1*true_mean_stand4,0,col="blue",lty=2)
plot(result$sample_size,result$estimated_mean_stand4,ylim=range_volume,
      main="Estimated means stand 4 vs sample size")
abline(true_mean_stand4,0,col="orange")
abline(true_mean_stand4+0.1*true_mean_stand4,0,col="orange",lty=2)
abline(true_mean_stand4-0.1*true_mean_stand4,0,col="orange",lty=2)
```



Inspecting the generated figures we can observe the variance of our estimates as a function of sample size and draw some interesting conclusions.

1. The variability of our estimates over repeated samples decreases with a pattern that resembles the graph of $\sqrt{\frac{1}{n}}$
2. As we know the sample mean is unbiased and, on average, tend to coincide with the true means for the forest and the stands.
3. When we compare the figure for the total forest and the figure for stand 4, we see that at some point, the sample size for stand 4 becomes so small that the variability is very large.

Now we are going to repeat the same process but instead of using the sample mean we will use the auxiliary information to calculate the regression estimator using p95 as auxiliary variable (X).

```
repeats <- 100
sample_sizes <- c(16, 64, 96, 128, 160, 320, 640 )
result_reg <- data.frame(rep=c(),sample_size=c(),
                         estimated_mean_forest=c(),
                         estimated_mean_stand1 = c(),
                         estimated_mean_stand2 = c(),
```

```
                      estimated_mean_stand3 = c(),
                      estimated_mean_stand4=c())
cont <- 1
for(i in 1:repeats){
  for(j in sample_sizes){
    # get a sample of the indices of the data
    my_sample_reg<- sample(dim(example1)[1],j,replace = FALSE)
    # subset the forest and obtain the sample fit using OLS (lm function)
    sampled_data_reg <- example1[my_sample_reg,]
    model_forest <- lm(volume~p95,data = sampled_data_reg)
    # calculate the regression estimator
    media_reg_forest <- mean(predict(model_forest,example1)) + mean(model_forest$residuals

    # now we will create subsamples for stands, obtain subsample fits for the stand and ap
    # the regression estimator
    stand1 <- sampled_data_reg[sampled_data_reg$stand==1,]
    media_reg_stand1 <- try({
      model_stand1 <- lm(volume~p95, data=stand1)
      mean(predict(model_stand1,example1[example1$stand==1,]))+ mean(model_stand1$residual
    })
    if(inherits(media_reg_stand1,"try-error")){
      media_reg_stand1<-NA
    }
    # repeat for stand 2
    stand2 <- sampled_data_reg[sampled_data_reg$stand==2,]
    media_reg_stand2 <- try({
      model_stand2 <- lm(volume~p95, data=stand2)
      mean(predict(model_stand2,example1[example1$stand==2,])) + mean(model_stand2$residua
    })
    # bypass very low stand sample size
    if(inherits(media_reg_stand2,"try-error")){
      media_reg_stand2<-NA
    }
    # repeat for stand 3
    stand3 <- sampled_data_reg[sampled_data_reg$stand==3,]
    media_reg_stand3 <- try({
      model_stand3 <- lm(volume~p95, data=stand3)
      mean(predict(model_stand3,example1[example1$stand==3,])) + mean(model_stand3$residua
    })
    # bypass very low stand sample size
    if(inherits(media_reg_stand3,"try-error")){
```

```
      media_reg_stand3<-NA
    }

    # repeat for stand 4
    stand4 <- sampled_data_reg[sampled_data_reg$stand==4,]
    media_reg_stand4 <- try({
      model_stand4 <- lm(volume~p95, data=stand4)
      mean(predict(model_stand4,example1[example1$stand==4,])) + mean(model_stand4$residua
    })
    # bypass very low stand sample size
    if(inherits(media_reg_stand4,"try-error")){
      media_reg_stand4<-NA
    }



    stand2 <- sampled_data_reg[sampled_data_reg$stand==2,]
    stand3 <- sampled_data_reg[sampled_data_reg$stand==3,]
    stand4 <- sampled_data_reg[sampled_data_reg$stand==4,]
    # store the rep and sample size
    result_reg[cont,"rep"]<-i
    result_reg[cont,"sample_size"]<-j
    # calculate the sample means for the forest and each stand
    result_reg[cont,"estimated_mean_forest"]<-media_reg_forest
    result_reg[cont,"estimated_mean_stand1"]<-media_reg_stand1
    result_reg[cont,"estimated_mean_stand2"]<-media_reg_stand2
    result_reg[cont,"estimated_mean_stand3"]<-media_reg_stand3
    result_reg[cont,"estimated_mean_stand4"]<-media_reg_stand4
    # pass to next iteration
    cont <- cont+1
  }
}
```

```
Warning in predict.lm(model_stand1, example1[example1$stand == 1, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand1, example1[example1$stand == 1, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

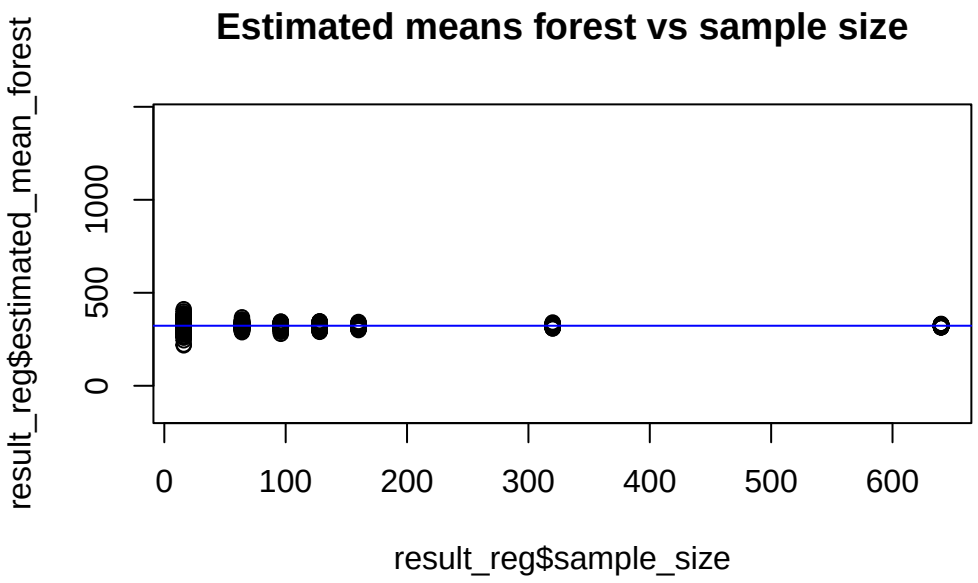
```
Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand4, example1[example1$stand == 4, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand2, example1[example1$stand == 2, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand4, example1[example1$stand == 4, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand1, example1[example1$stand == 1, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
  0 (non-NA) cases

Warning in predict.lm(model_stand2, example1[example1$stand == 2, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand2, example1[example1$stand == 2, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```
Warning in predict.lm(model_stand4, example1[example1$stand == 4, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand1, example1[example1$stand == 1, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
Warning in predict.lm(model_stand1, example1[example1$stand == 1, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand2, example1[example1$stand == 2, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
  0 (non-NA) cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand1, example1[example1$stand == 1, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
  0 (non-NA) cases

Warning in predict.lm(model_stand1, example1[example1$stand == 1, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand4, example1[example1$stand == 4, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
Warning in predict.lm(model_stand4, example1[example1$stand == 4, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
Warning in predict.lm(model_stand4, example1[example1$stand == 4, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

Warning in predict.lm(model_stand3, example1[example1$stand == 3, ]):
prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
# Store the pre-calculated true values
result_reg$true_mean_forest<-true_mean_forest
result_reg$true_mean_stand1 <- true_mean_stand1
result_reg$true_mean_stand2 <- true_mean_stand2
result_reg$true_mean_stand3 <- true_mean_stand3
result_reg$true_mean_stand4 <- true_mean_stand4
```

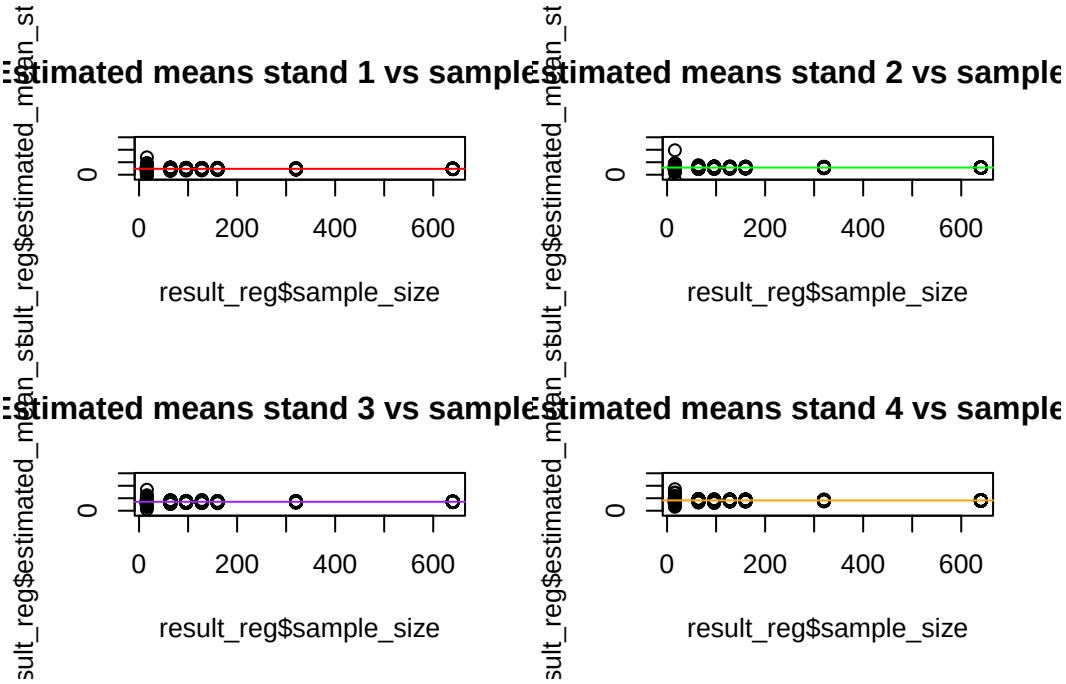
No let's plot the results

```r
plot(result_reg$sample_size,result_reg$estimated_mean_forest,ylim=range_volume,
      main="Estimated means forest vs sample size")
abline(true_mean_forest,0,col="blue")
```

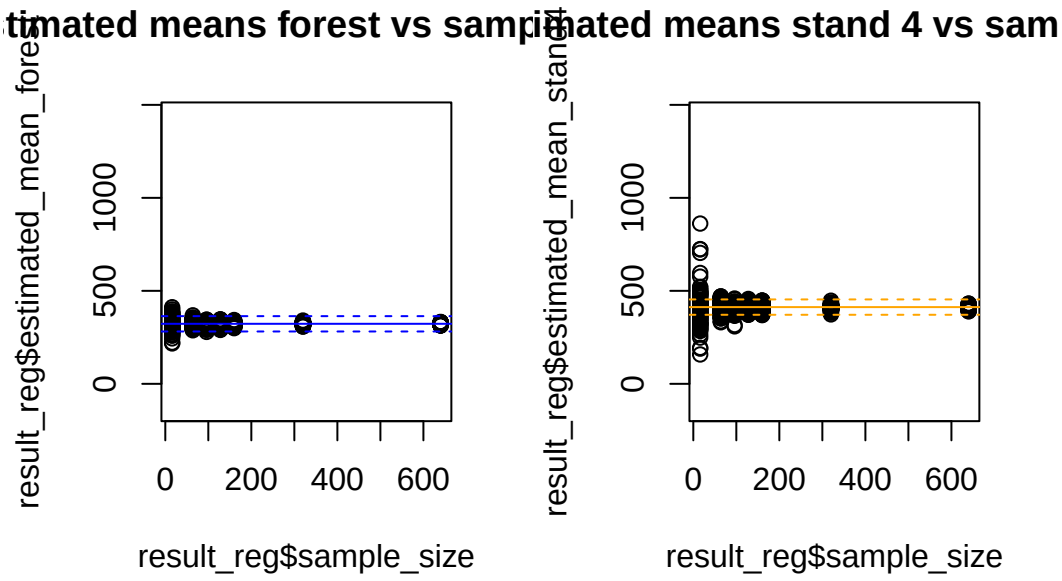



```r
par(mfrow=c(2,2))
plot(result_reg$sample_size,result_reg$estimated_mean_stand1,ylim=range_volume,
      main="Estimated means stand 1 vs sample size")
abline(true_mean_stand1,0,col="red")
plot(result_reg$sample_size,result_reg$estimated_mean_stand2,ylim=range_volume,
      main="Estimated means stand 2 vs sample size")
abline(true_mean_stand2,0,col="green")
plot(result_reg$sample_size,result_reg$estimated_mean_stand3,ylim=range_volume,
```

```
      main="Estimated means stand 3 vs sample size")
abline(true_mean_stand3,0,col="purple")
plot(result_reg$sample_size,result_reg$estimated_mean_stand4,ylim=range_volume,
      main="Estimated means stand 4 vs sample size")
abline(true_mean_stand4,0,col="orange")
```




```
par(mfrow=c(1,2))
plot(result_reg$sample_size,result_reg$estimated_mean_forest,ylim=range_volume,
     main="Estimated means forest vs sample size")
abline(true_mean_forest,0,col="blue")
abline(true_mean_forest+0.1*true_mean_stand4,0,col="blue",lty=2)
abline(true_mean_forest-0.1*true_mean_stand4,0,col="blue",lty=2)
plot(result_reg$sample_size,result_reg$estimated_mean_stand4,ylim=range_volume,
      main="Estimated means stand 4 vs sample size")
abline(true_mean_stand4,0,col="orange")
abline(true_mean_stand4+0.1*true_mean_stand4,0,col="orange",lty=2)
abline(true_mean_stand4-0.1*true_mean_stand4,0,col="orange",lty=2)
```



If we observe the plots for the regression estimator we can see:

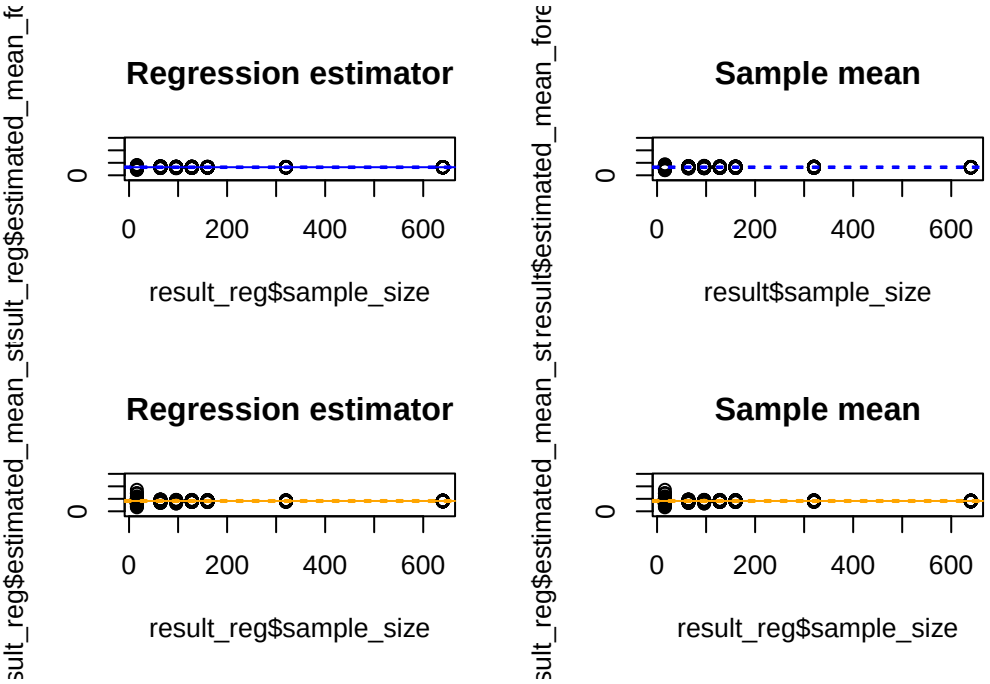
- The pattern with $\sqrt{\frac{1}{n}}$ is also there, but this time the variability is smaller.
- For all sample sizes, the estimates over repeated samples concentrate around the true values.
- As before, the estimates for the entire forest have much lower variability than estimates for single stands.
- You probably obtained some warnings in the process because some stands were never sampled or had very small sampled sizes.

Regression estimator with simple random sampling

Let's now compare use the regression estimator. We will simulate a precision requirement of +- 10% in our estimates plotting discontinuous lines around the true value

```
par(mfrow=c(2,2))
plot(result_reg$sample_size,result_reg$estimated_mean_forest,ylim=range_volume,
     main="Regression estimator")
abline(true_mean_forest,0,col="blue")
abline(true_mean_forest+0.1*true_mean_forest,0,col="blue",lty=2)
```

```
abline(true_mean_forest-0.1*true_mean_forest,0,col="blue",lty=2)
plot(result$sample_size,result$estimated_mean_forest,ylim=range_volume,
     main="Sample mean")
abline(true_mean_forest+0.1*true_mean_forest,0,col="blue",lty=2)
abline(true_mean_forest-0.1*true_mean_forest,0,col="blue",lty=2)


plot(result_reg$sample_size,result_reg$estimated_mean_stand4,ylim=range_volume,
      main="Regression estimator")
abline(true_mean_stand4,0,col="orange")
abline(true_mean_stand4+0.1*true_mean_stand4,0,col="orange",lty=2)
abline(true_mean_stand4-0.1*true_mean_stand4,0,col="orange",lty=2)
plot(result_reg$sample_size,result_reg$estimated_mean_stand4,ylim=range_volume,
      main="Sample mean")
abline(true_mean_stand4,0,col="orange")
abline(true_mean_stand4+0.1*true_mean_stand4,0,col="orange",lty=2)
abline(true_mean_stand4-0.1*true_mean_stand4,0,col="orange",lty=2)
```



When comparing the sample mean to the regression estimator, we can observe that:

- For the entire forest the regression estimator does much better for all sample sizes. This is because the relationship between volume and p95 can be approximated well by a linear

function. The relationship, however, is not linear, it was created with the code spinet below, but this method does not rely on assuming that a linear relationship exists.

- For both methods variability for stands was larger and for small sample sizes variances were large.

This is the small area estimation problem. We have reach to a point where even using auxiliary information with a large correlation with our response variable does not allow us to obtain reliable estimates.

```
id <- 1:1600
x <- rep(0:39,each=40)
y <- rep(0:39,times=40)
stand <- (1+x%/%20) +2*(y%/%20)
p95 <- rnorm(1600,50,10+5*stand)
volume <- rnorm(1600,p95+0.05*p95^2+50*stand,100)
example1 <- data.frame(id=id,x=x,y=y,stand=stand,volume=volume, p95=p95)
```

Take the code snippet above and replace the #PLACEHOLDER comment in the first cell of code with a data pattern generated by you for volume. You can make the data have a more or less linear relationship. Then re-run the code and see what happens.

Important notes

All our assessment were based on repeating the sampling process. That sampling process implemented the design that we chose (simple random sampling). There are some things to consider for general sampling designs:

- For simple random sampling and systematic designs we use ordinary least square to estimate the line of best fit. For other general sampling designs that does not necessarily hold. Read Särndal et al., (2003) section 6.5 for a more general formulation of regression estimators for general sampling designs.
- We have used an example in which we know the entire population. This is appropriate to see how regression estimators work. In real applications we will only have the variable of interest for the sample. In those cases we will have to use formulas to estimate the variance of our regression estimator. Särndal et al., (2003) also provides those formulas and they will provide estimates of the spread of the points that we obtain for each sample size.

Model-based synthetic estimator

To start with the regression estimator we are going to use the data in example2.csv. It is the data we used in example one but the volume column has been removed so we can simulate different models.

```
example2 <- read.csv("example2.csv")
```

Lets start assuming some model form.

$$Vol_i \sim N(25 + 6 * P95, 130^2)$$

With $Cov(Vol_i, Vol_j) = 0$ for all i and j

Element 200 in the population has a value of P95 equal to 43.06279, according to this value the volume for this unit is distributed normally with mean 25+6*43.06279 = 283.3767, and variance 200.

$$Vol_{200} \sim N(283.3767, 130^2)$$

We can calculate the distribution of the mean volume.

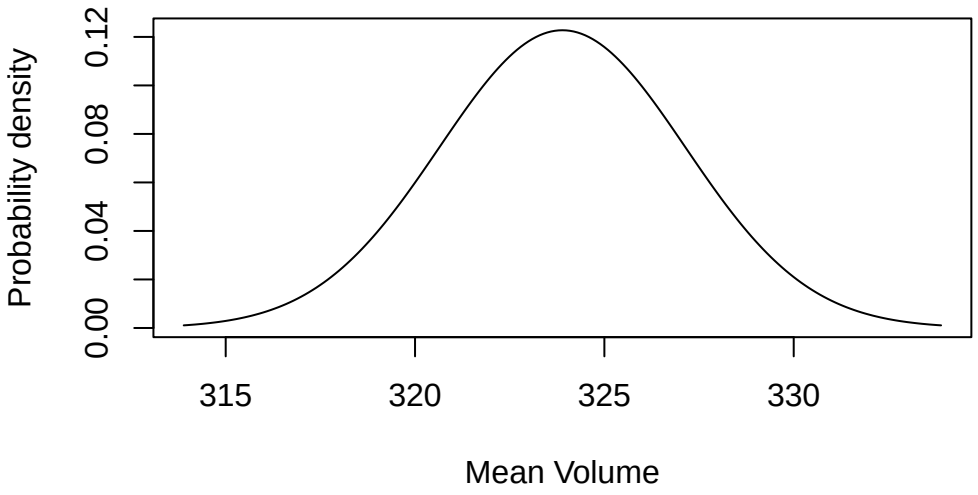
$$\bar{Vol} \sim N(\frac{1}{1600}\sum_{i=1}^{1600}(25 + 6P95_i), \frac{1}{1600}130^2) = N(323.8917, 3.25^2)$$

According to the model the distribution of the mean volume in the forest is:

```
mean_vol_dist <- mean(25+6*example2$p95)
mean_vol_dist
```

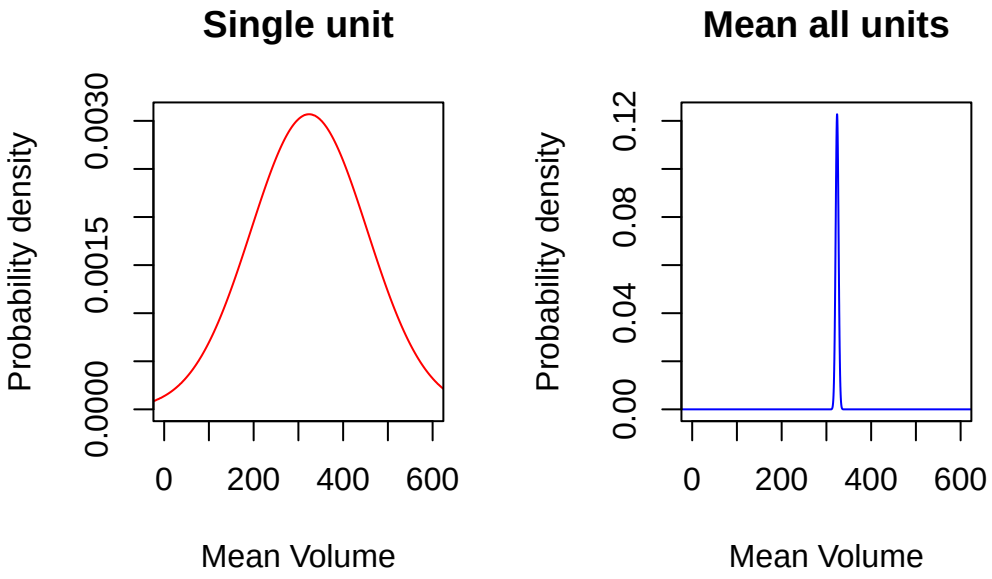
```
[1] 323.8917
```

```
plot(mean_vol_dist+c(-100:100)/10,
     dnorm(mean_vol_dist+c(-100:100)/10,
           mean=mean_vol_dist,
           130/sqrt(1600)),
     type="l",ylab="Probability density",xlab="Mean Volume")
```



Lets compare the distribution of the mean to the distribution of given unit centered also at 323.8917.

```
par(mfrow=c(1,2))
plot(x=mean_vol_dist+c(-10000:10000)/10,xlim=c(0,600),
     y=dnorm(mean_vol_dist+c(-10000:10000)/10,
     mean=mean_vol_dist, 130),col="red",
     type="l",ylab="Probability density",xlab="Mean Volume",main="Single unit")
plot(x=mean_vol_dist+c(-10000:10000)/10,xlim=c(0,600),
     y=dnorm(mean_vol_dist+c(-10000:10000)/10,
     mean=mean_vol_dist, 130/sqrt(1600)),col="blue",
     type="l",ylab="Probability density",xlab="Mean Volume",main="Mean all units")
```



We can see that assuming independence is a very strong assumption. With independece departures there is a large compensation. Some units deviate above the mean and some units deviate below the mean causing the variance of the mean to be very small. We will see that this assumption is not realistic in many settings. For example, we can expect volume for units within a give stand to show some correlation. Stands are differentiated units, someone draw the stand boundaries for a reason, and it is reasoble to expect some within stand correlation.

References

Särndal, C.-E., Swensson, B., Wretman, J., 2003. Model Assisted Survey Sampling (Springer Series in Statistics).