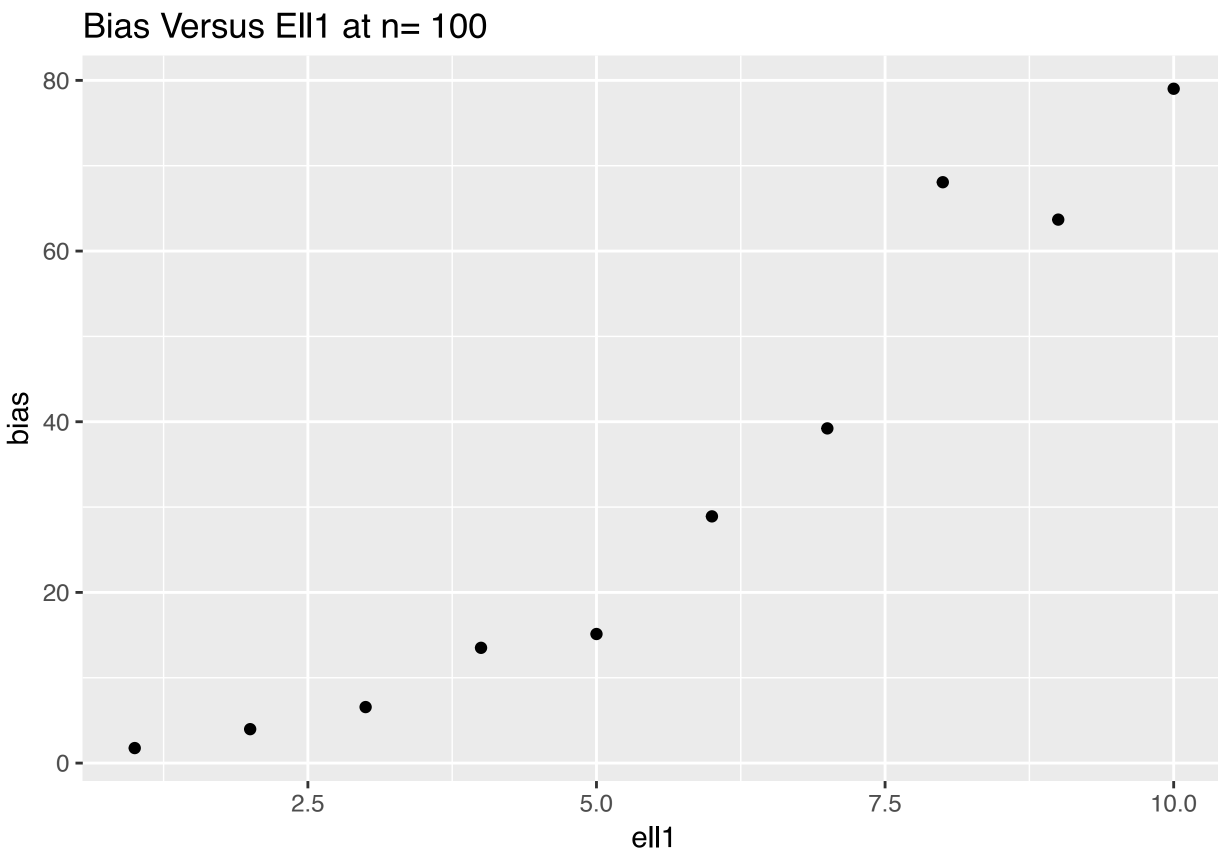
hw4.r

Tue May 2 08:26:50 2017

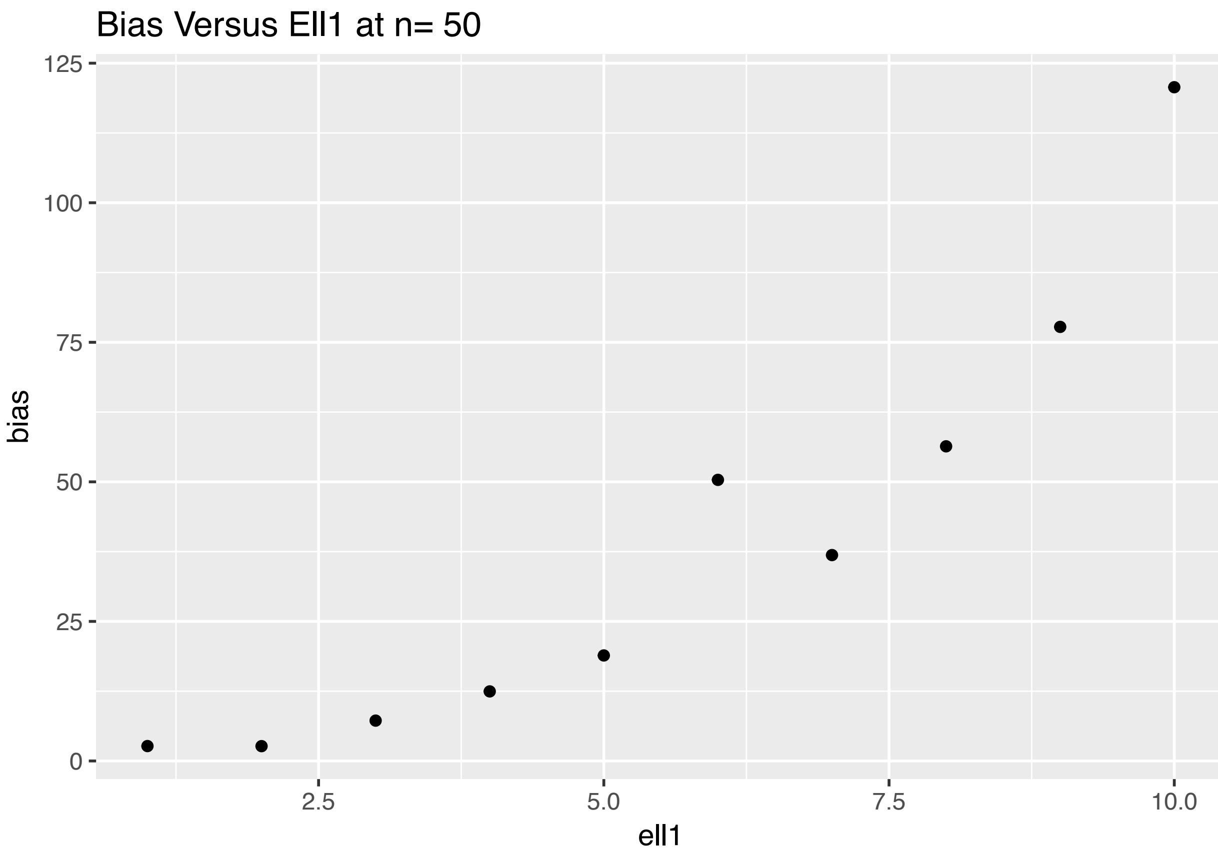
```
### PROBLEM 1 ###

# accepts an n*p matrix
# x and returns the largest eigenvalue of cov(x)
topeval = function(x){
  mat.cov <- cov(x)
  ev <- eigen(mat.cov)
  values <- ev$values
  return(max(values))
}

#creates an n*p matrix with columns 2 through p iid N(0,1),
# column 1 has entries iid N(0,ell1)
rspikenorm = function(ell1,n,p) {
  col.one <- rnorm(n, 0, ell1)
  mat <- as.data.frame(col.one)
  for (i in 2:p) {
    col <- rnorm(n, 0, 1)
    c <- as.data.frame(col)
    mat <- cbind(mat, c)
  }
  colnames(mat) <- 1:p
  return(mat)
}

# generates a series of M datasets which are n by p,
# with spike eigenvalue ell1. For each dataset it should
# evalute  the top eigenvalue of its corresponding covariance
# matrix. It should return the mean of the so-obtained top
# eigenvalues and the e standard deviation of the obtained
# top eigenvalues
sampling.moments = function(ell1,n,p,M) {
  x <- rep(NA, M)
  for (i in 1:M) {
    data <- rspikenorm(ell1,n,p)
    x[i] <- topeval(data)
  }
  my.struct <- rep(NA,2)
  my.struct[1] <- mean(x)
  my.struct[2] <- sd(x)
  return(my.struct)
}

### PROBLEM 2 ###

# "evaluate the bias of the
#top eigenvalue as an estimate of ell1,
```

```r
# at problem size n=100, p=50 "

estm.bias <- function(ell1,n) {
  bias <- rep(NA, length(ell1))
  for (i in ell1) {
    x <- rspikenorm(i,n,50)
    y <- topeval(x)
    bias[i] <- y-(i)
  }
  df <- cbind(as.data.frame(ell1), as.data.frame(bias))
  g <- ggplot(df, aes(ell1,bias)) + geom_point()
  g + ggtitle(paste("Bias Versus Ell1 at n=",n))
}
library(ggplot2)
### PROBLEM 2.A ###
estm.bias(1:10, 100)
```

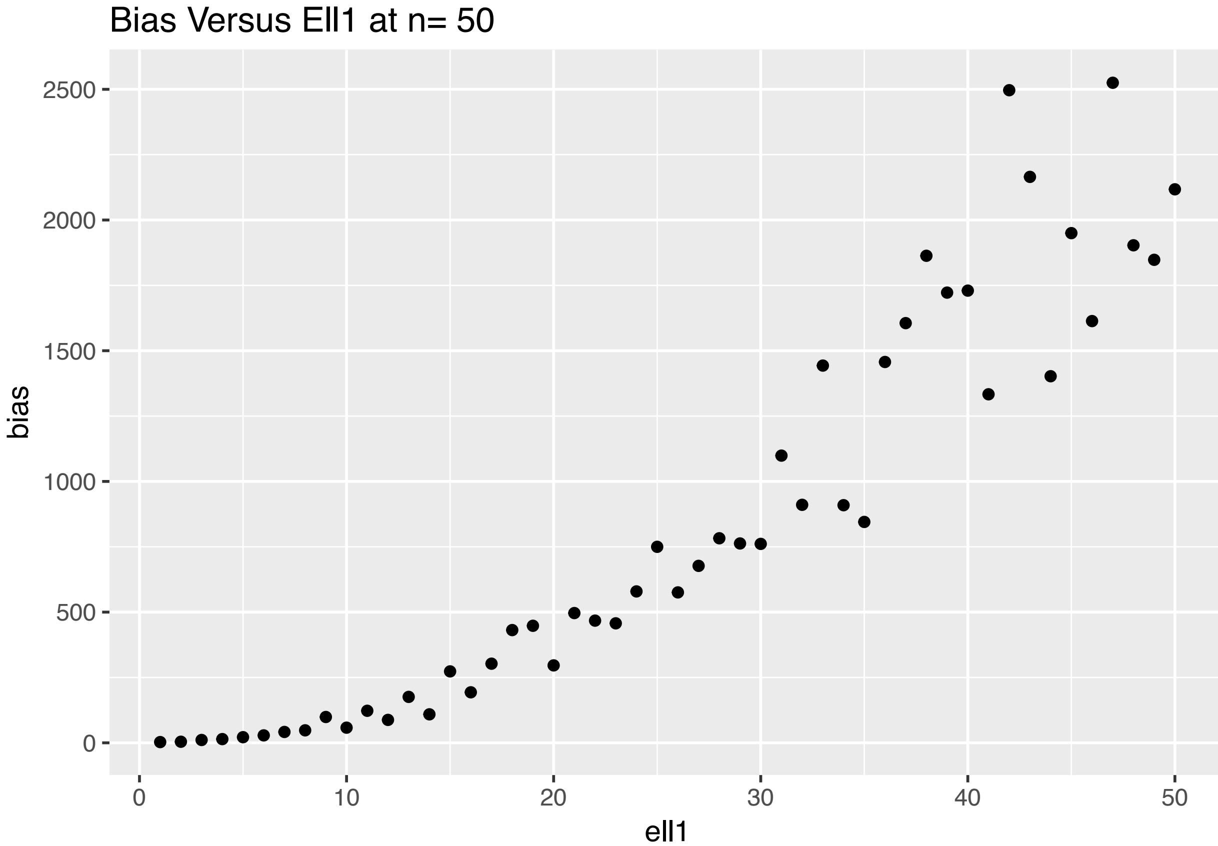



```r
estm.bias(1:50, 100)
```

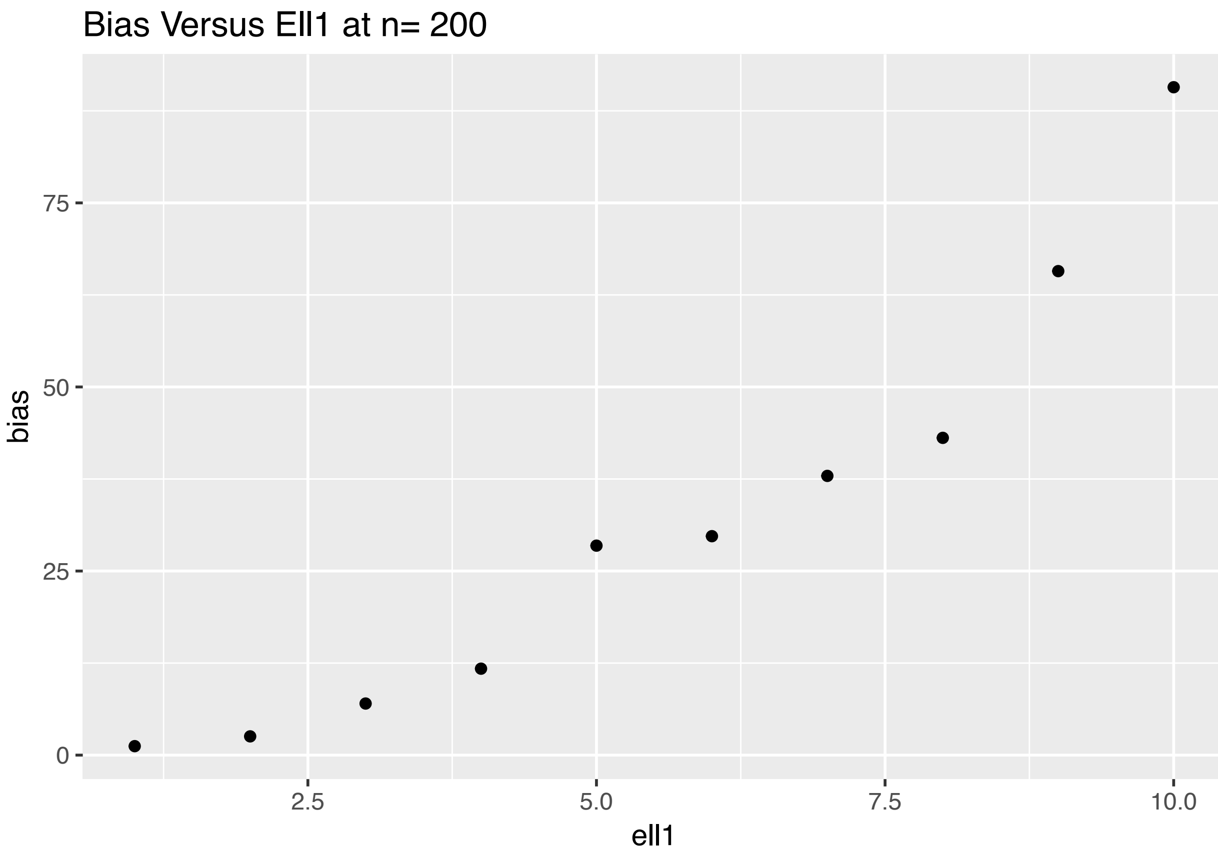

```
### PROBLEM 2.B ###
estm.bias(1:10, 50)
```



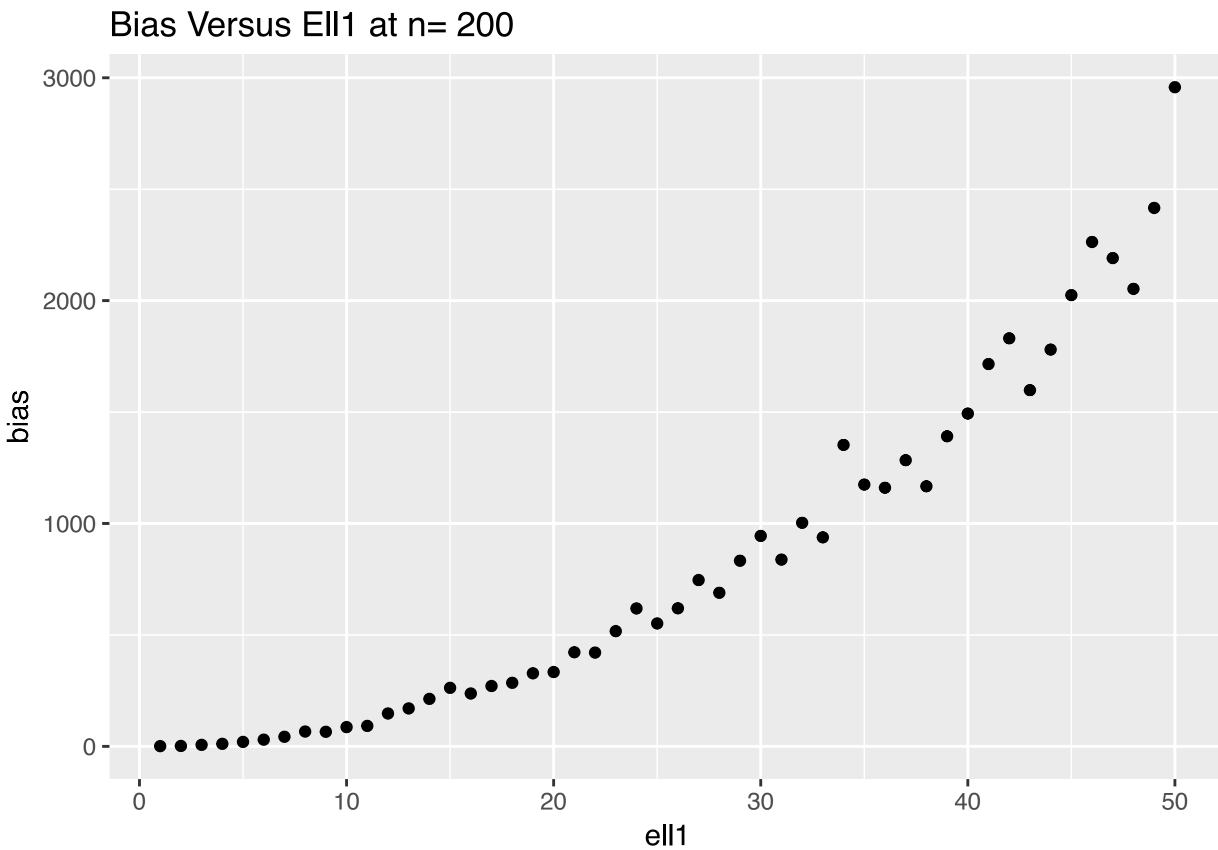

```
estm.bias(1:50, 50)
```




```
### PROBLEM 2.C ###
estm.bias(1:10, 200)
```

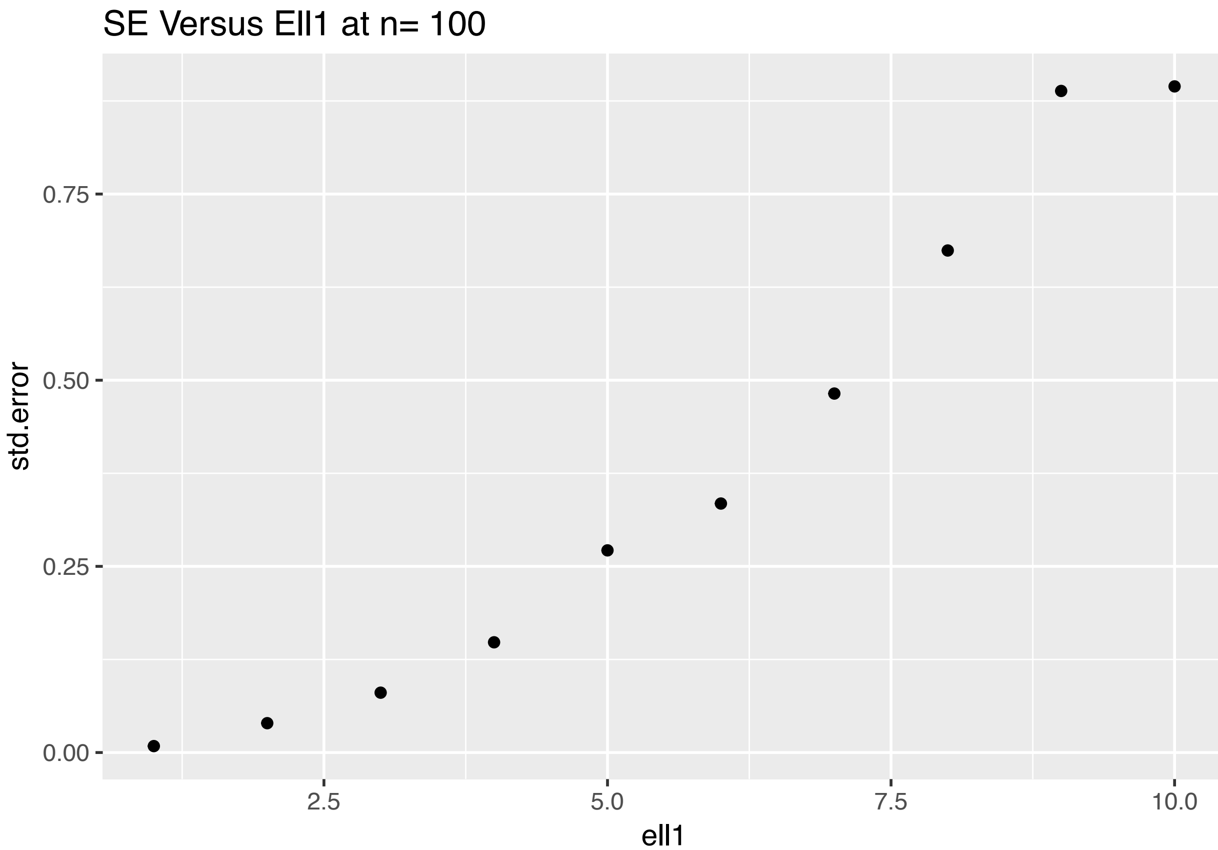



```
estm.bias(1:50, 200)
```

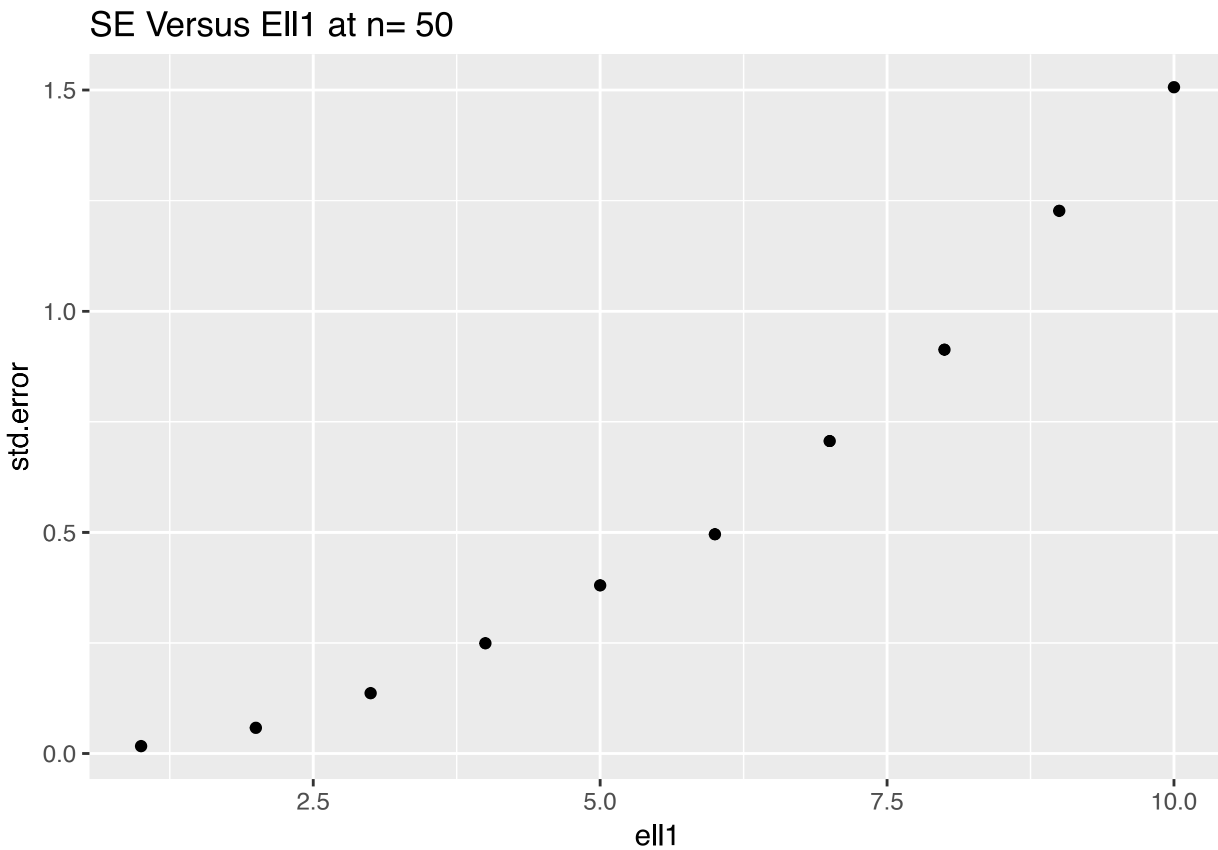



```
### PROBLEM 2.D ###
# For n=50,100,and 200, the distributions of bias as a function of ell1 are remarkably similar.
# At n=10, the the difference between true and empirical is about 100 for all three.
#The main takeaway is that the bias increases faster than linear with ell1, probably about O(n2).

### PROBLEM 3 ###

# In the setting of problem 1, evaluate the standard error of the
# top eigenvalue as an estimate of ell1, at problem size n=100, p=50

# In #2, I was able to find the bias at many different points
# Here, I will use a range of ell1 to compute a single standard error

estm.se <- function(ell1, n) {
  std.error <- rep(NA, length(ell1))
  sim.power <- 200
  for (i in ell1) {
    sim.vec <- rep(NA, sim.power)
    for (j in 1:sim.power) {
      x <- rspikenorm(i,n,50)
      sim.vec[j] <- topeval(x)
    }
    std.error[i] <- (sd(sim.vec)/sqrt(length(sim.vec)))
  }
  df <- cbind(as.data.frame(ell1), as.data.frame(std.error))
  g <- ggplot(df, aes(ell1,std.error)) + geom_point()
  g + ggtitle(paste("SE Versus Ell1 at n=",n))
```

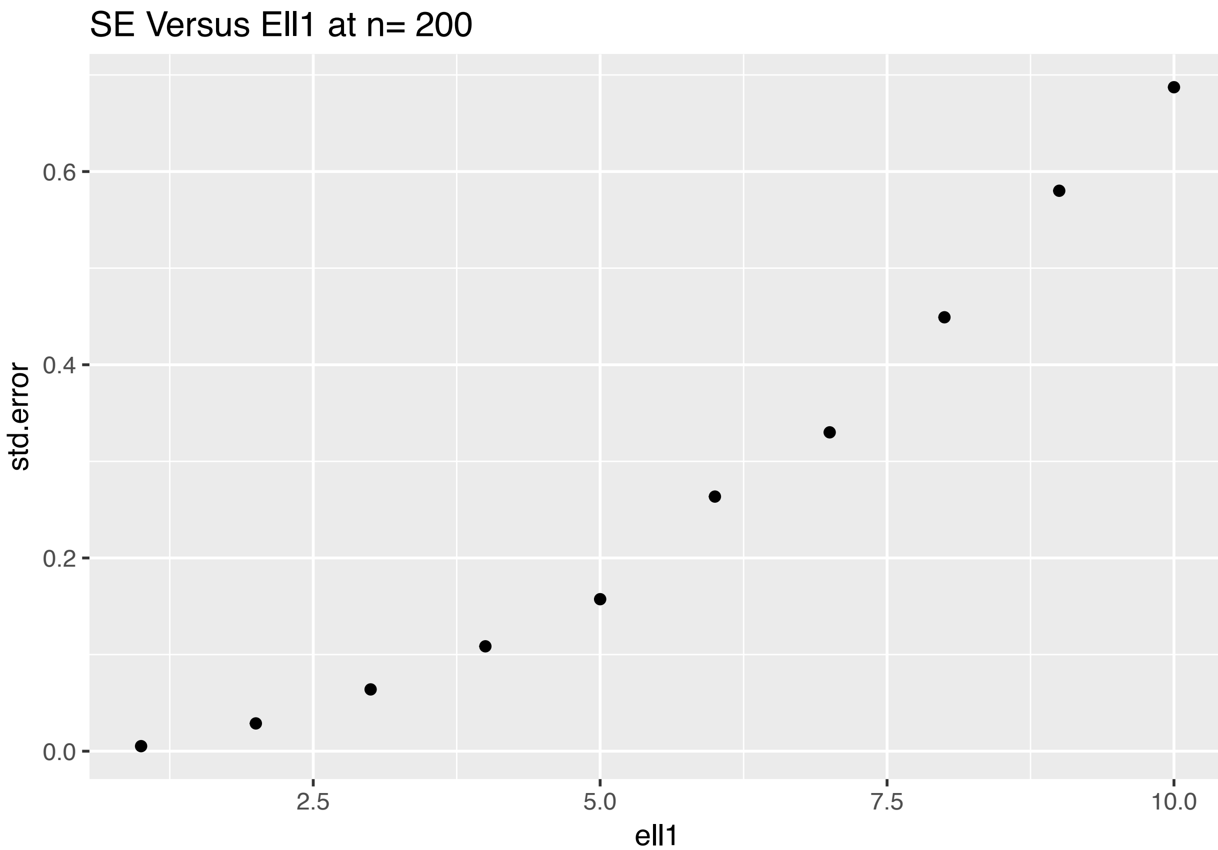
```
}
### PROBLEM 3.A ###
estm.se(1:10,100)
```




```
### PROBLEM 3.B ###
estm.se(1:10,50)
```

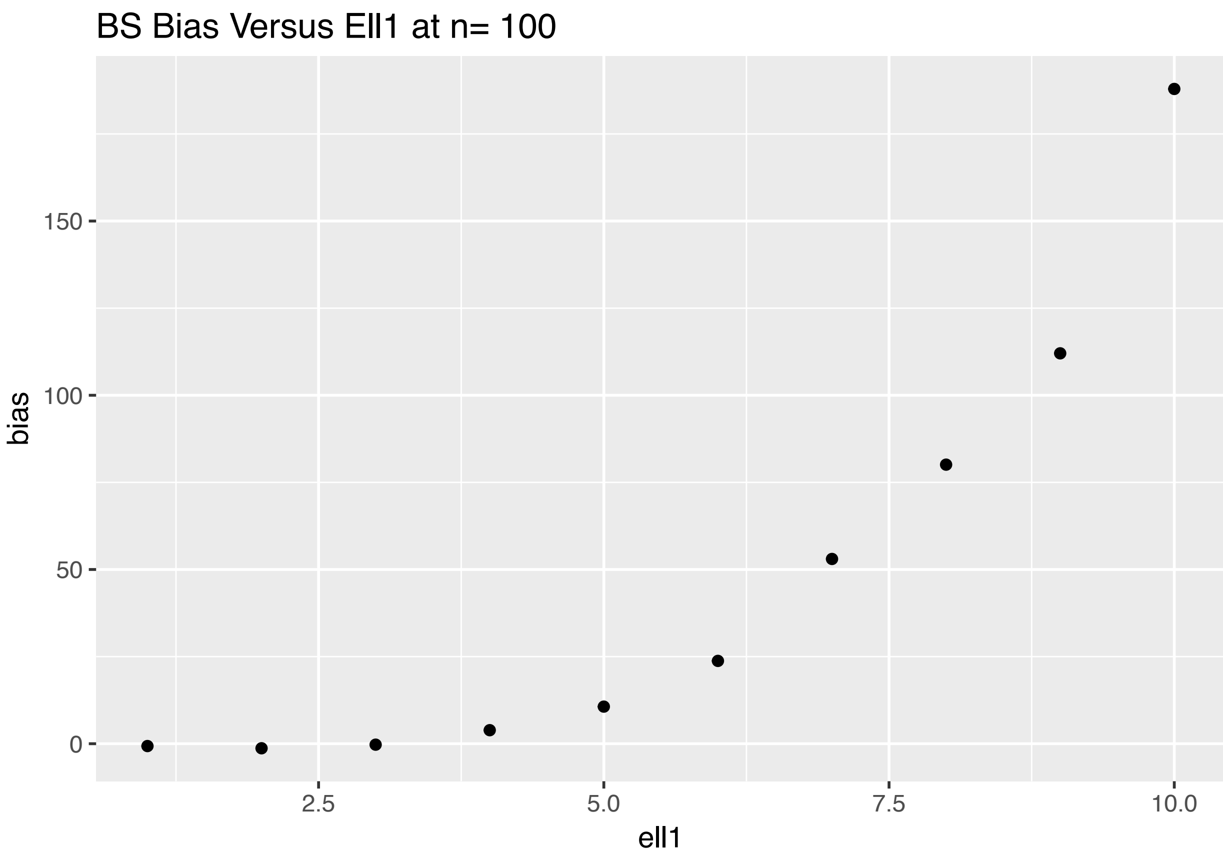



```
### PROBLEM 3.C ###
estm.se(1:10,200)
```

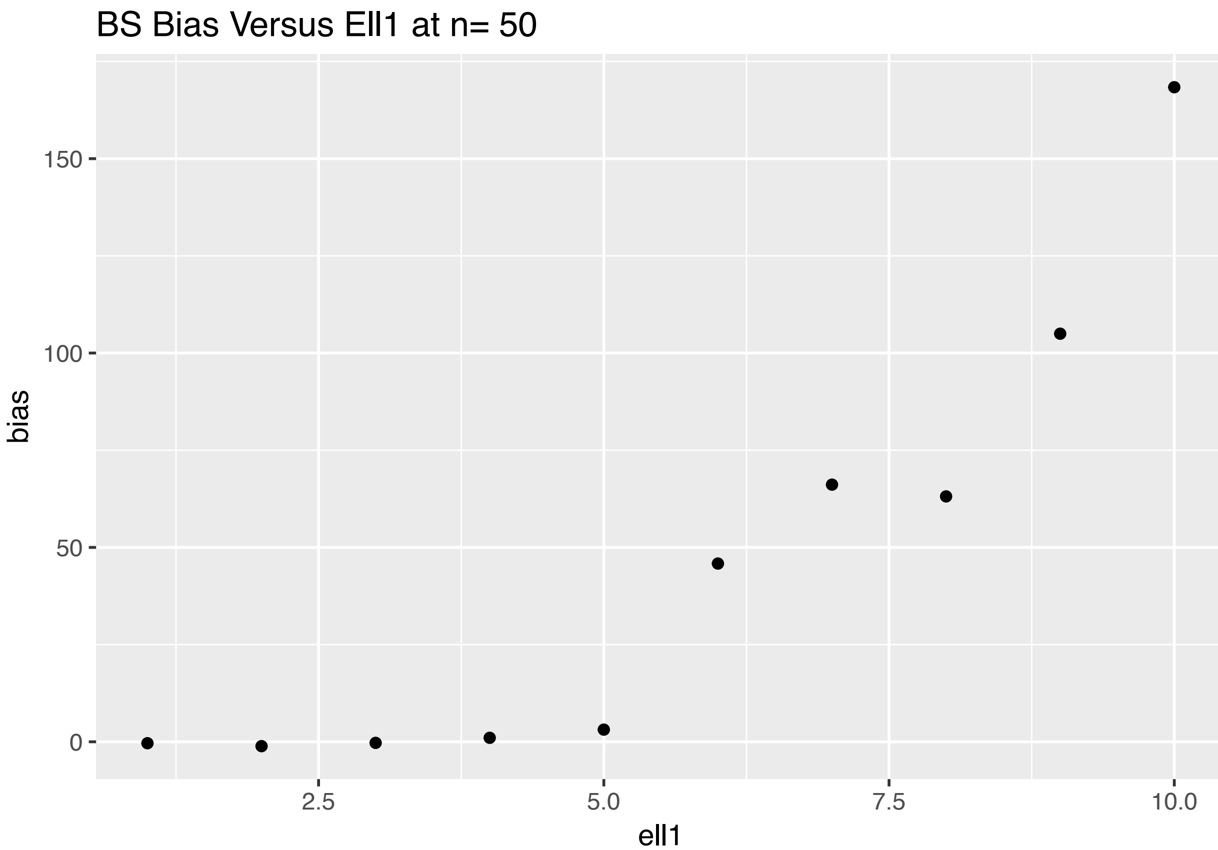



```
### PROBLEM 3.D ###

# The standard error also seems to increase as ell1 increases.
# n=50 had the highest SE (at ell1=10, the highest value tested)
# and n=200 had the lowest, suggesting an inverse relationship
# between n and SE

### PROBLEM 4 ###

# successively generates a series of B
# bootstrap realizations x* from the n by p dataset x,
# by resampling rows. For each dataset x* it should
# evaluate the top eigenvalue of its corresponding covariance
# matrix. It should return the mean of the so-obtained top eigenvalues.
# and the e standard deviation of the obtained top eigenvalues.

resampling.moments = function(x,B) {
  top.evals <- rep(NA, B)
  bs.ind <- 1:B
  for (i in bs.ind) {
    boot.inds <- sample(1:nrow(x),replace=TRUE)
    boot.sample <- x[boot.inds,]
    boot.cov <- cov(boot.sample)
    top.evals[i] <- topeval(boot.cov)
  }
  list <- rep(NA,2)
  list[1] <- mean(top.evals)
```

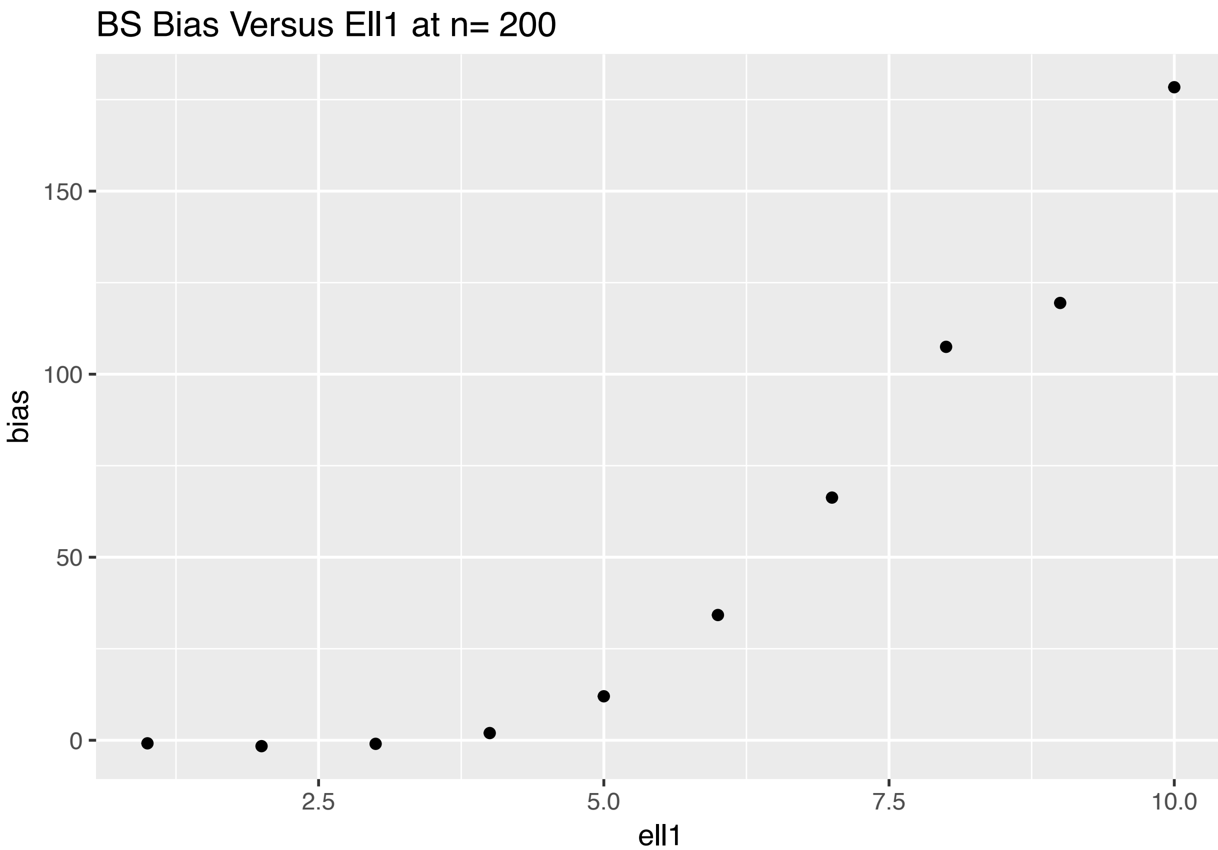
```r
  #print(paste("mean:",mean(top.evals)))
  #print(paste("sd:",sd(top.evals)))
  list[2] <- sd(top.evals)
  return(list)
}

### PROBLEM 5 ###

estm.bootbias <- function(ell1, n) {
  bias <- rep(NA,length(ell1))
  for (i in ell1) {
    x <- rspikenorm(i,n,50)
    y <- resampling.moments(x,200)
    bias[i] <- y[1]-i
  }
  df <- cbind(as.data.frame(ell1),as.data.frame(bias))
  g <- ggplot(df,aes(ell1,bias))+geom_point()
  g + ggtitle(paste("BS Bias Versus Ell1 at n=",n))
}

### PROBLEM 5A ###
library(ggplot2)
estm.bootbias(1:10,100)
```



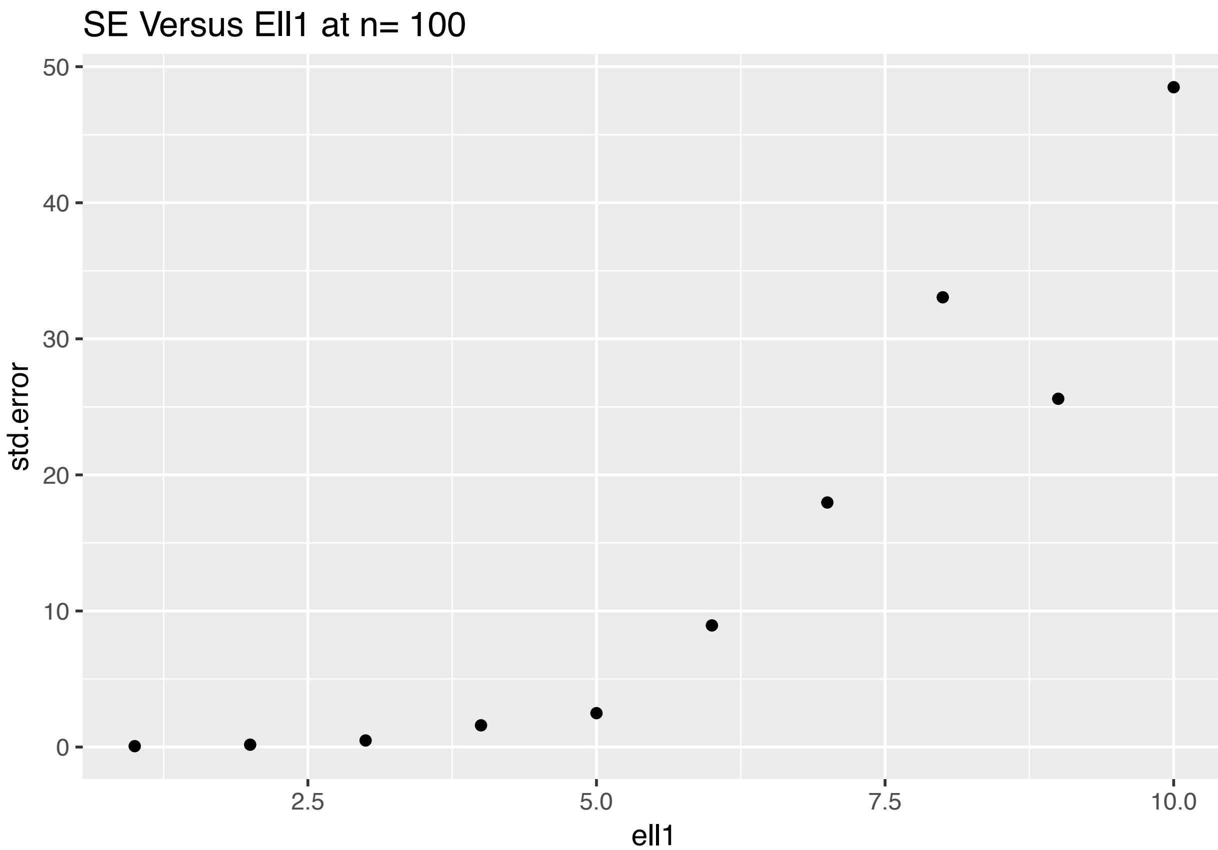

```r
### PROBLEM 5B ###
estm.bootbias(1:10,50)
```



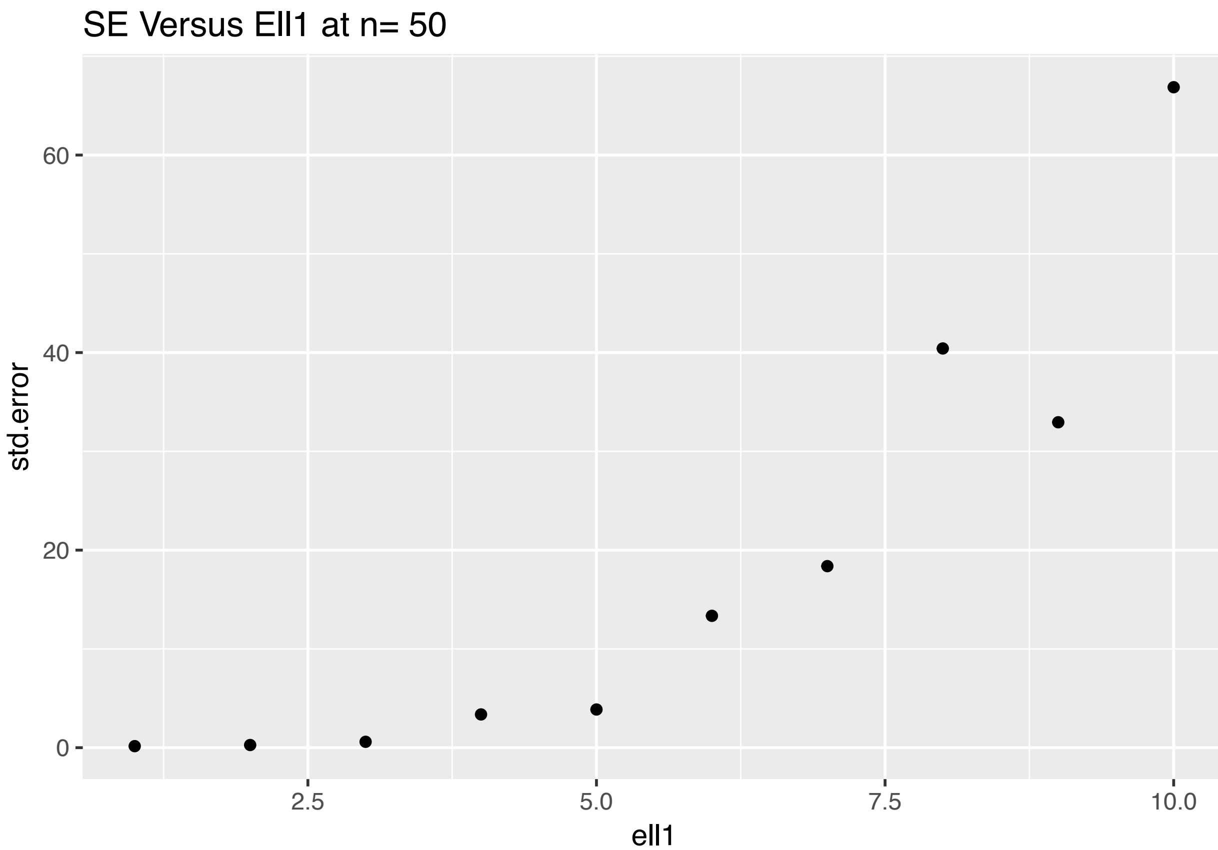

```
### PROBLEM 5C ###
estm.bootbias(1:10,200)
```



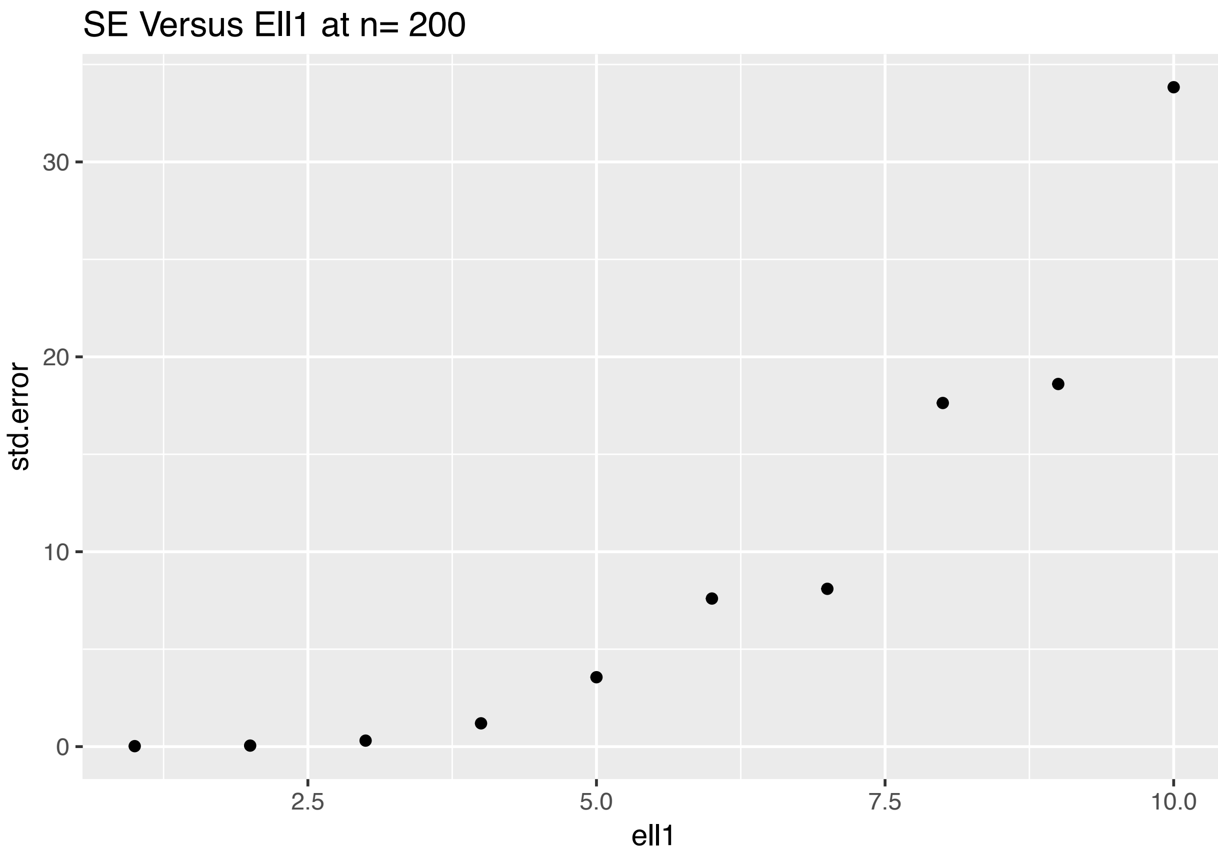

```
### PROBLEM 5D ###
# Bias again grows with ell1. My results actually show that the bootstrap has worse bias
# than the more analytic solution we tried earlier, which is surprising.'

### PROBLEM 6 ###
estm.bootse <- function(ell1, n) {
  std.error <- rep(NA,length(ell1))
  for (i in ell1) {
    x <- rspikenorm(i,n,50)
    y <- resampling.moments(x,200)
    std.error[i] <- y[2]
  }
  df <- cbind(as.data.frame(ell1),as.data.frame(std.error))
  g <- ggplot(df,aes(ell1,std.error))+geom_point()
  g + ggtitle(paste("SE Versus Ell1 at n=",n))
}

### PROBLEM 6A ###
estm.bootse(1:10,100)
```




```
### PROBLEM 6B ###
estm.bootse(1:10,50)
```




```
### PROBLEM 6C ###
estm.bootse(1:10,200)
```




```
### PROBLEM 6D ###
# My results also show that the bootstrap has a higher standard error

### PROBLEM 7 ###
# My conclusion would be that the bootstrap is ineffetive for this estimation, though
# I'm not sure why. Alternatively, I made an implementation error, since the
# bootstrap is known for its versatility. I'm not sure if my non-BS method
# of finding standard error was correct.
```