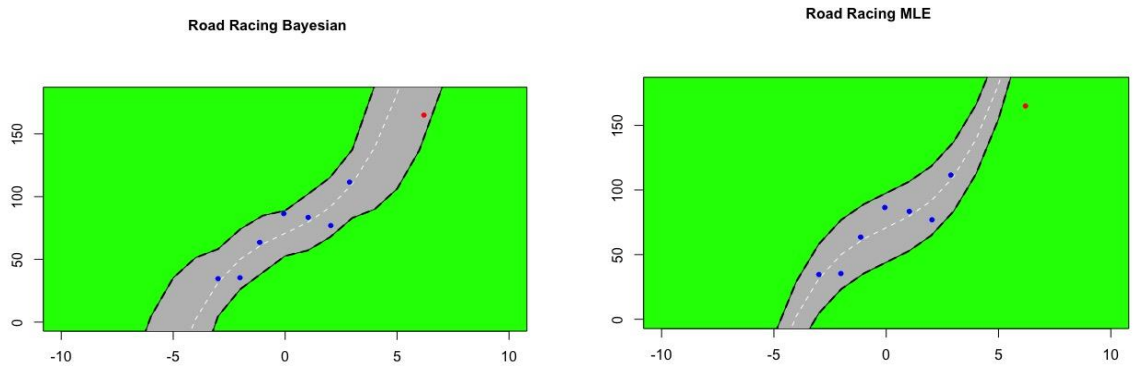
Ans:3)

We have a Formula One racing track.It is seen that car racing is so competitive that a car wins by minute difference, hence, mostly the cars are very near to each other and the probability of them crashing is very high.



Note: We have assumed, FI racing in this scenario does not have a road. The grey area is the assumed road, calculated from CIs, grey area is the confidence interval not the actual road.There are 11 cars racing at a particular instance, let's say April 8th, 10:00am. The blue dots represent the cars that we can see using our GPS camera.

The GPS system is strong but cannot capture all the cars due to bad weather and terrain. At the same time, we need to keep an eye on the cars because when there is a crash, the drivers need immediate action.For example, if a blue car in the front crashes into another car and the respective driver informs their team through their mikes, we should have paramedics ready to attend to that driver.**But, we can send some help only when we know where that car is.**

To note, the paramedics change their locations depending on the driver's location. Hence, paramedics keep on changing their locations.

But, as our GPS system is very weak, we were not able to capture the image of a red car. We need red car’s location because of the same reason above. **Therefore, we calculate confidence intervals.**The aim to calculate the confidence interval is to understand in what area the red car can be found.

We found confidence intervals (95%) using frequentist/classical method. The grey area is the confidence interval, the black lines are boundaries of upper and lower limit, the white line is the predicted lines, the blue cars are observed data, whereas the red car is new data to be predicted. Technically, the boundaries of the road will dictate where the paramedic should be.

The area in grey (looks like road) shows that by just taking into consideration the ten blue cars, our method, says that out of 100 random times, in positions of blue cars something like what we have (10 blue cars present at almost same locations) 95 times the red car will be in that grey (road area). But, as stated, our red car is among the rest 5%, that means it couldn't

be predicted using MLE. If the red car crashes, we will know the car crashed as the driver will tell their team, but we won't be able to say what path it will be on and hence, will fail to send paramedics.

On the other hand,

We have Bayesian intervals, Bayesian intervals , which will take into account the data (the blue cars) and the prior distribution, that is, how usually cars are believed to be driven in an F1 racing (In other words the style of a car is driven).

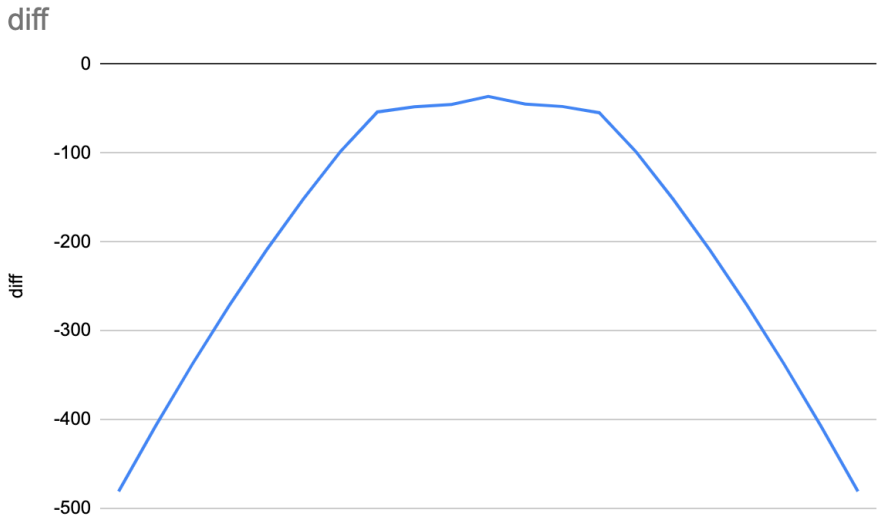
In Bayesian intervals, paramedics, even though we might need more of paramedics, because the interval is large, but still we can predict the red cars location.

Understanding the width of intervals:

Bayesian intervals:

lowerCI	**upperCI**	**Diff**
-819.008967	-338.19082	-480.818147
-619.87945	-212.77872	-407.10073
-453.358734	-116.23015	-337.128584
-316.114257	-45.04741	-271.066847
-204.815715	4.26946	-209.085175
-116.130754	35.21835	-151.349104
-46.783937	51.35409	-98.138027
4.560986	58.17069	-53.609704
26.266768	74.13566	-47.868892
39.53919	84.87346	-45.33427
52.724024	88.8686	-36.144576
57.020024	101.75257	-44.732546
67.61662	115.1662	-47.54958
82.990821	137.46274	-54.471919
89.959196	188.65589	-98.696694
106.197499	257.90015	-151.702651
137.197377	346.53412	-209.336743
186.544014	457.80289	-271.258876
257.746086	595.02804	-337.281954
354.308128	761.53529	-407.227162
479.730094	960.65494	-480.924846

Plot of column difference (Bayesian)

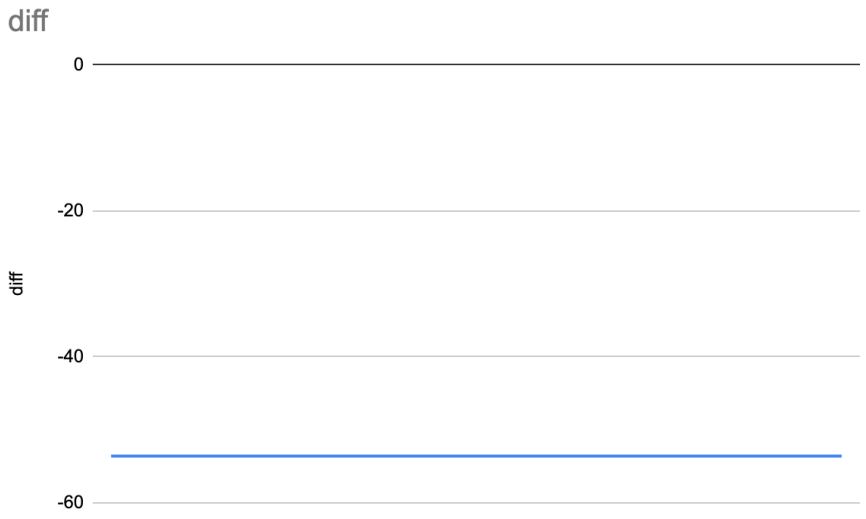


MLE intervals:

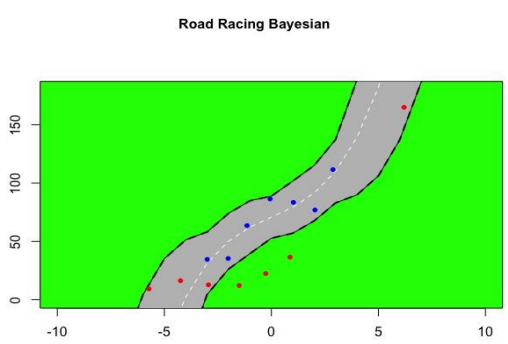
lowerCI	upperCI	Diff
-599.759611	-546.17316	-53.586451
-439.125438	-385.53899	-53.586448
-308.887102	-255.30065	-53.586452
-205.667288	-152.08084	-53.586448
-126.088681	-72.50223	-53.586451
-66.773966	-13.18752	-53.586446
-24.345827	29.24062	-53.586447
4.573051	58.1595	-53.586449
23.359983	76.94643	-53.586447
35.392284	88.97873	-53.586446
44.047271	97.63372	-53.586449
52.702257	106.2887	-53.586443
64.734558	118.32101	-53.586452
83.52149	137.10794	-53.58645
112.440368	166.02682	-53.586452
154.868507	208.45495	-53.586443
214.183222	267.76967	-53.586448
293.761829	347.34828	-53.586451
396.981643	450.56809	-53.586447
527.219979	580.80643	-53.586451

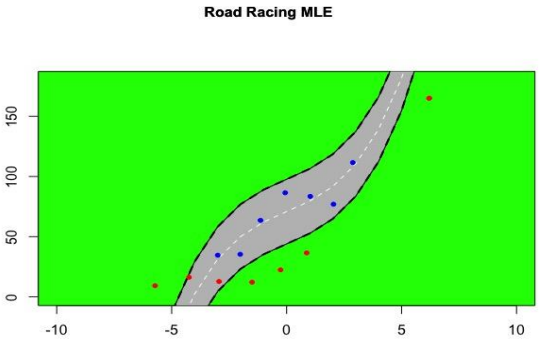
687.854153	741.4406	-53.586447

Plot of column difference MLE:



The idea is to pick the new data from the same distribution (an extreme values) , let's see for more examples, where new data points come from the same distribution.





As we can see there are a lot of red cars, these red cars we need to predict their location.

Bayesian is 43% successful whereas MLE is 26% successful. With this example, we can say Bayesian performs better than MLE.

BayesianCodeAssignment2.R

sarah

2022-04-08

```
library("sp")


#Plot original data

set.seed(10)

n <- 7

x_orig <- seq(-3, 3, length = n) + rnorm(n, 0, 0.1)

x_orig

## [1] -2.99812538 -2.01842525 -1.13713305 -0.05991677  1.02945451  2.03897943

## [7]  2.87919238

y <- 70 + x_orig +  x_orig^3 + rnorm(n, 0, 15)


#Creating the model

n<-length(y)

X <- cbind(rep(1, n), x_orig, x_orig^3)

colnames(X) <- NULL

p<-dim(X)[2]

fit.ls<-lm(y~-1+ X)

beta.0<-fit.ls$coef

nu.0<-1  ; sigma2.0<-sum(fit.ls$res^2)/(n-p)

Sigma.0<- solve(t(X)%*%X)*sigma2.0*n

S<-5000


rmvnorm<-function(n,mu,Sigma)
```

```
{ # samples from the multivariate normal distribution
  E<-matrix(rnorm(n*length(mu)),n,length(mu))
  t(  t(E%*%chol(Sigma)) +c(mu))
}

## some convenient quantites
n<-length(y)
p<-length(beta.0)
iSigma.0<-solve(Sigma.0)
XtX<-t(X)%*%X

## store mcmc samples in these objects
beta.post<-matrix(nrow=S,ncol=p)
sigma2.post<-rep(NA,S)
Sigma.post <- matrix(nrow = S, ncol = p*p)

## starting value
set.seed(1)
sigma2<- var( residuals(lm(y~0+X)) )

for( scan in 1:S) {

  #update beta
  V.beta<- solve(  iSigma.0 + XtX/sigma2 )
  E.beta<- V.beta%*%( iSigma.0%*%beta.0 + t(X)%*%y/sigma2 )
  beta<-t(rmvnorm(1, E.beta,V.beta) )

  #update sigma2
```

```
  nu.n<- nu.0+n
  ss.n<-nu.0*sigma2.0 + sum(  (y-X%*%beta)^2 )
  sigma2<-1/rgamma(1,nu.n/2, ss.n/2)

  #save results of this scan
  beta.post[scan,]<-beta
  sigma2.post[scan]<-sigma2
  Sigma.post[scan,] <- c(V.beta)
}

round( apply(beta.post,2,mean), 3)
## [1] 70.796  8.021  0.569
BP <- apply(beta.post, 2, mean)
BP
## [1] 70.7963108  8.0207964  0.5691882
s2P <- mean(sigma2.post)
s2P
## [1] 254.8178
SigmaP <- matrix(apply(Sigma.post, 2, mean), ncol = p, byrow = F)
SigmaP
##                  [,1]       [,2]       [,3]
## [1,] 28.3681117 -1.121874  0.2084256
## [2,] -1.1218736 52.612059 -6.8111495
## [3,]  0.2084256 -6.811150  1.0196722
plot(x_orig, y, xlab = "", ylab = "", xlim = c(-10, 10), ylim = c(0, 180), cex = 1, pch = 20, col =
"black")

xnew <- seq(-10, 10, by = 1)
Xnew <- cbind(rep(1, length(xnew)), xnew, xnew^3)
```

```
colnames(Xnew) <- NULL
predP <- Xnew %*% BP
s2P <- sqrt(s2P * diag(Xnew %*% SigmaP %*% t(Xnew)))
ll <- predP - c(qnorm(0.975) * sqrt(s2P))
ul <- predP + c(qnorm(0.975) * sqrt(s2P))
rect(par("usr")[1], par("usr")[3],
          par("usr")[2], par("usr")[4], par("usr")[5],
          col = "green") # Color

polygon(c(xnew, rev(xnew)), c(ll, rev(ul)),
          col = "grey")
lines(xnew, predP, col = "white", lwd =1,lty = "dashed")
lines(xnew, ll, col = "black", lwd = 2, lty = 2)
lines(xnew, ul, col = "black", lwd = 2, lty = 2)
points(x_orig, y, col = "blue",pch = 20)

#Some more random cars
points(6.2, 165, col = "red",pch = 20)
mtext("Road Racing Bayesian", side = 1, line = -24, outer = TRUE,font=2)

set.seed(4)
n <- 7

x_red <- seq(-7, 1, length = n) + rnorm(n, 0, 0.1)
x_red
## [1] -6.9783245 -5.7209159 -4.2442189 -2.9404019 -1.5031049 -0.2644058  0.8718753
y_red <- 10 + x_orig +  x_orig^3 + rnorm(n, 0, 5)
```

```
points(x_red, y_red, col = "red",pch = 20)
in_or_out = point.in.polygon(x_red,y_red,c(xnew, rev(xnew)),c(ll, rev(ul)))
total_vals = mean(in_or_out)*100
paste(round(total_vals,2) , "% red cars are inside the CI")
## [1] "42.86 % red cars are inside the CI"
#"42.86 % red cars are inside the CI"
```

MLECodeBassignment2.R

sarah

2022-04-08

```
library("sp")

#Plot original data

set.seed(10)
n <- 7
x_orig <- seq(-3, 3, length = n) + rnorm(n, 0, 0.1)
x_orig
## [1] -2.99812538 -2.01842525 -1.13713305 -0.05991677  1.02945451  2.03897943
## [7]  2.87919238
y <- 70 + x_orig +  x_orig ^ 3 + rnorm(n, 0, 15)

#Creating the model

X <- cbind(rep(1, n), x_orig, x_orig ^ 3)
colnames(X) <- NULL
X
##       [,1]       [,2]       [,3]
## [1,]  1 -2.99812538 -2.694942e+01
```

```
## [2,]  1 -2.01842525 -8.223146e+00
## [3,]  1 -1.13713305 -1.470394e+00
## [4,]  1 -0.05991677 -2.151024e-04
## [5,]  1  1.02945451  1.090992e+00
## [6,]  1  2.03897943  8.476929e+00
## [7,]  1  2.87919238  2.386778e+01
p <- dim(X)[[2]]
p
## [1] 3
B <- solve(t(X) %*% X) %*% t(X) %*% y
B
##                   [,1]
## [1,] 70.8404944
## [2,]  8.0921003
## [3,]  0.5628859
s2 <- t(y - X %*% B) %*% (y - X %*% B) / (n - p)
s2
##          [,1]
## [1,] 186.8761
plot(
  x_orig,
  y,
  xlab = "",
  ylab = "",
  xlim = c(-10, 10),
  ylim = c(0, 180),
  cex = 1,
  pch = 20,
```

```
 col = "black"
)



xnew <- seq(-10, 10, by = 1)
Xnew <- cbind(rep(1, length(xnew)), xnew, xnew ^ 3)
colnames(Xnew) <- NULL
pred <- Xnew %*% B
ll <- pred - c(qnorm(0.975) * sqrt(s2))
ul <- pred + c(qnorm(0.975) * sqrt(s2))
rect(par("usr")[1],
          par("usr")[3],
          par("usr")[2],
          par("usr")[4],
          par("usr")[5],
          col = "green",
          par(new = TRUE)) # Color
polygon(c(xnew, rev(xnew)), c(ll, rev(ul)),
          col = "grey")
lines(xnew,pred,col = "white",lwd = 1,lty = "dashed")
lines(xnew,ll,col = "black",lwd = 2,lty = 2)
lines(xnew,ul,col = "black",lwd = 2,lty = 2)
points(x_orig, y, col = "blue", pch = 20)


points(6.2, 165, col = "red", pch = 20)
mtext(
 "Road Racing MLE",
 side = 1,
```

```
 line = -24,
 outer = TRUE,
 font = 2
)
polygon(c(xnew, rev(xnew)), c(ll, rev(ul)),
          col = "grey")



lines(xnew,pred,col = "white",lwd = 1,lty = "dashed")
lines(xnew,ll,col = "black",lwd = 2,lty = 2)
lines(xnew,ul,col = "black",lwd = 2,lty = 2)
points(x_orig, y, col = "blue", pch = 20)
points(6.2, 165, col = "red", pch = 20)
mtext("Road Racing MLE",side = 1, line = -24,outer = TRUE, font = 2)
#Some extra red cars
set.seed(4)
n <- 7
x_red <- seq(-7, 1, length = n) + rnorm(n, 0, 0.1)
x_red
## [1] -6.9783245 -5.7209159 -4.2442189 -2.9404019 -1.5031049 -0.2644058  0.8718753
y_red <- 10 + x_orig +  x_orig ^ 3 + rnorm(n, 0, 5)



points(x_red, y_red, col = "red", pch = 20)
in_or_out = point.in.polygon(x_red, y_red, c(xnew, rev(xnew)), c(ll, rev(ul)))
total_vals = mean(in_or_out) * 100
paste(round(total_vals, 2) , "% red cars are inside the CI")
## [1] "28.57 % red cars are inside the CI"
```

```
#"28.57 % red cars are inside the CI"
```