Homework 1

Stat 597a: Spatial Models

Claire Kelling

Due September 28, 2017

Problem 1:

Suppose we want to simulate a random vector $Y \sim N(\mu, \Sigma)$. If Σ is symmetric and positive definite, it can be represented using the Cholesky decomposition $\Sigma = LL^T$, where L is a lower triangular matrix. Consider the following algorithm for simulating Y:

- **Calculate the matrix L.**
- **Sample $Z \sim N(0, I)$, where I is the n x n identity matrix.**
- **Let $Y = \mu + LZ$.**

Part a: Show that Y generated in this way has the correct distribution. You may use the fact that a linear function of a multivariate normal random variable is again multivariate normal; just show the mean and variance are correct.

First, I will show the mean of this Y.

$$E(Y) = E(\mu + LZ) = \mu + E(LZ) = \mu + LE(Z) = \mu + 0 = \mu$$

Now, I will check the variance of Y.

$$Var(Y) = Var(\mu + LZ) = Var(LZ) = LVar(Z)L^T = LIL^T = LL^T = \Sigma$$

As stated in the problem, since a linear function of a multivariate normal random variable is also a multivariate normal, and since I have verified that the mean and variance are correct, I have shown that Y generated in this way has the correct distribution of $Y \sim N(\mu, \Sigma)$.

Part b: Write a function or a few lines of code in R to implement this method for arguments mu and Sigma. You may use the built-in function chol for the Cholesky decomposition and rnorm to generate Z.

I have included my function in the code shown below, which is commented to show the steps.

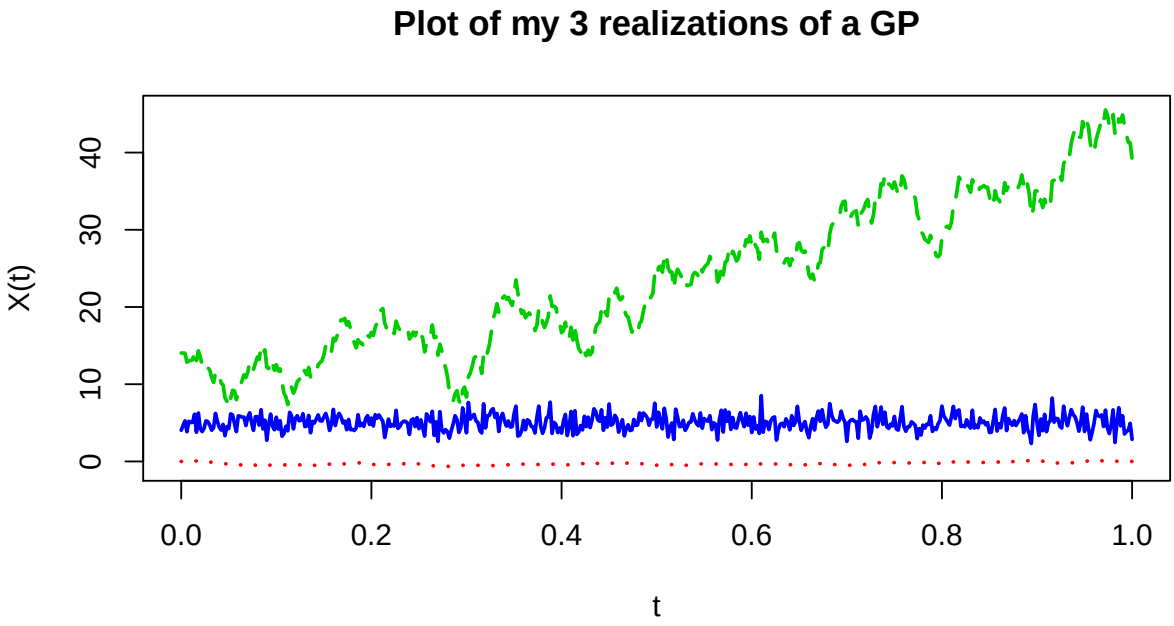
```
#creating a function with mu and Sigma inputs to generate Y~N(mu,Sigma) using N(0,1)
multi_norm <- function(mu, Sigma){
  # Because Sigma may be positive semidefinite, we need the pivot=TRUE option
  # and a few more steps to obtain the matrix L for Cholesky decomposition.
  chol_mat = chol(Sigma, pivot=TRUE)
  pivot = attr(chol_mat, "pivot")
  L = chol_mat[,order(pivot)]

  #Holding Z constant in order to just vary mu and Sigma
  set.seed(3)
  #generating standard normal
  Z = rnorm(nrow(Sigma),0,1)
  #generating N(mu,Sigma) through transformation
  Y = mu + t(L)%*%Z
}
```

Part c: For a mean and covariance function of your choosing, use your code from (b) and make a few plots illustrating realizations of a Gaussian process on [0, 1], but changing the different parameters in the model. These differences will be easier to see if you keep the same Z sample but just change mu and Sigma.

For this problem, I will create 3 realizations of a GP on [0,1] and changing several different parameters in the model. I set the seed in my mulinorm function so that Z does not vary, only mu and Sigma inputs will change.

For the red line on the bottom, we see that this is an example of a Standard Brownian Bridge on the interval [0,1]. It has the covariance matrix given by $\Sigma_{ij} = min(t_i, t_j) \times (1 - max(t_i, t_j))$. It has mean 0. The blue line slightly above this has a mean of 5, which is clearly seen in the plot, and $Sigma = diag(1)$ so there is only variation around the mean of 5. For the top process, it has mean 15, which is where it tends to hang around on the left side of the graph but Σ is a matrix of all one's (so without independence), so it doesn't just fluctuate around the mean, as with the second line mentioned.



Problem 2

The file CAtemps.RData contains two R objects of class SpatialPointsDataFrame, called CAtemp and CAgrid. CAtemp contains average temperatures from 1961- 1990 at 200 locations (latitude and longitude) in California in degrees Fahren- heit, along with their elevations in meters. CAgrid contains elevations in meters over a grid of locations. I've given you some code to get started with this data in HW1.R.

Consider the following model for the temperature data.

$$Y_i = \mu(s_i;\beta) + e(s_i;\sigma^2,\rho,\tau)$$

where $\mu(s;\beta) = \beta_0 + \beta_1$**Longitude**$(s) + \beta_2$**Latitude**$(s) + \beta_3$**Elevation**$(s)$ **and** $e(s_i;\sigma^2,\rho,\tau)$ **is a zero mean stationary Gaussian process with exponential covariance function.**

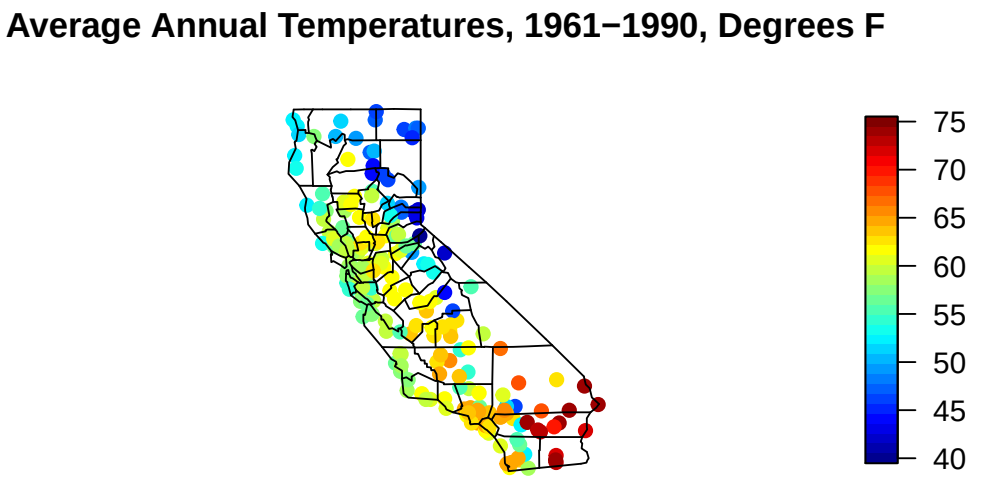
Another way of writing this is as

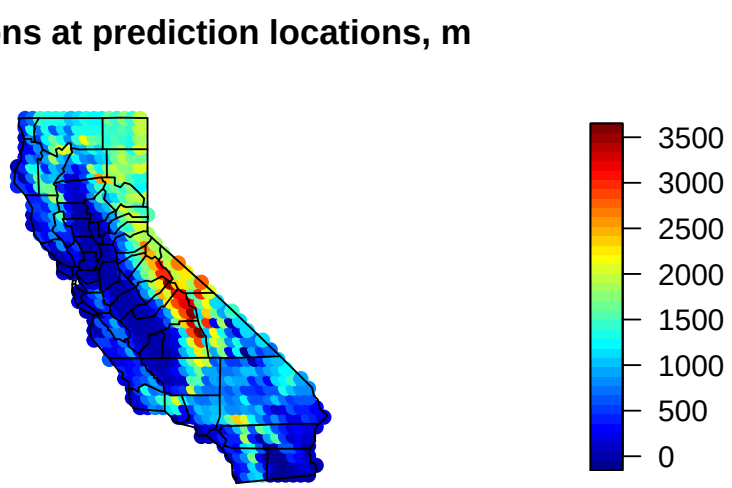
$$Y_i = \mu(s_i;\beta) + Z(s_i;\sigma^2,\rho) + \epsilon_i$$

where now Z is a mean zero Gaussian process like e **but without the nugget term, and the** ϵ_i **are iid** $N(0,\tau^2)$**, indepdendent of Z. This is important because we want to predict** $\mu(s_i;\beta) + Z(s_i;\sigma^2,\rho)$ ***without*** **the measurement error.**

Before I begin the problem, I would like to include the plots that were included in the sample code. I included both the temperatures at the observed locations as well as the elevations over the whole grid.

We see that there is definitely some spatial dependence in time, and latitude, longitude, and elevation all seem to plan an important role, though the linear effect of longitude is less clear.



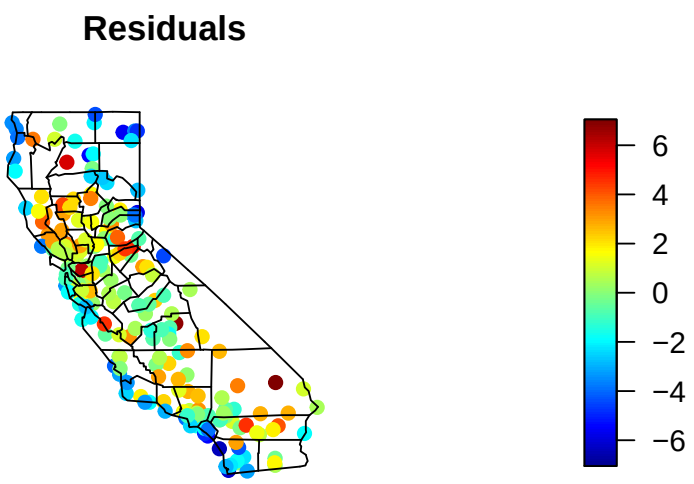


Part a: Using the CAtemp data, form a preliminary estimate of β using ordinary least squares and make a color plot of the residuals. Include your estimates and plot.

Below, I have included my estimates for β through ordinary least squares, as well as a color plot of the residuals. We see that there is some spatial structure to the residuals. According to the summary, also presented below, these covariates seem to have an effect on the temperature.

```
##
## Call:
## lm(formula = avgtemp ~ lon + lat + elevation, data = CAtemp)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.304 -1.780  0.082  1.687  6.954
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.215e+02  1.602e+01   20.06  < 2e-16 ***
## lon          2.324e+00  1.736e-01   13.39  < 2e-16 ***
## lat          5.647e-01  1.586e-01    3.56 0.000465 ***
## elevation   -9.648e-03  3.923e-04  -24.59  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.583 on 196 degrees of freedom
## Multiple R-squared:  0.853,  Adjusted R-squared:  0.8507
## F-statistic: 379.1 on 3 and 196 DF,  p-value: < 2.2e-16
```

```
##   (Intercept)           lon           lat     elevation
## 321.511433492   2.324104683   0.564680460  -0.009647649
```



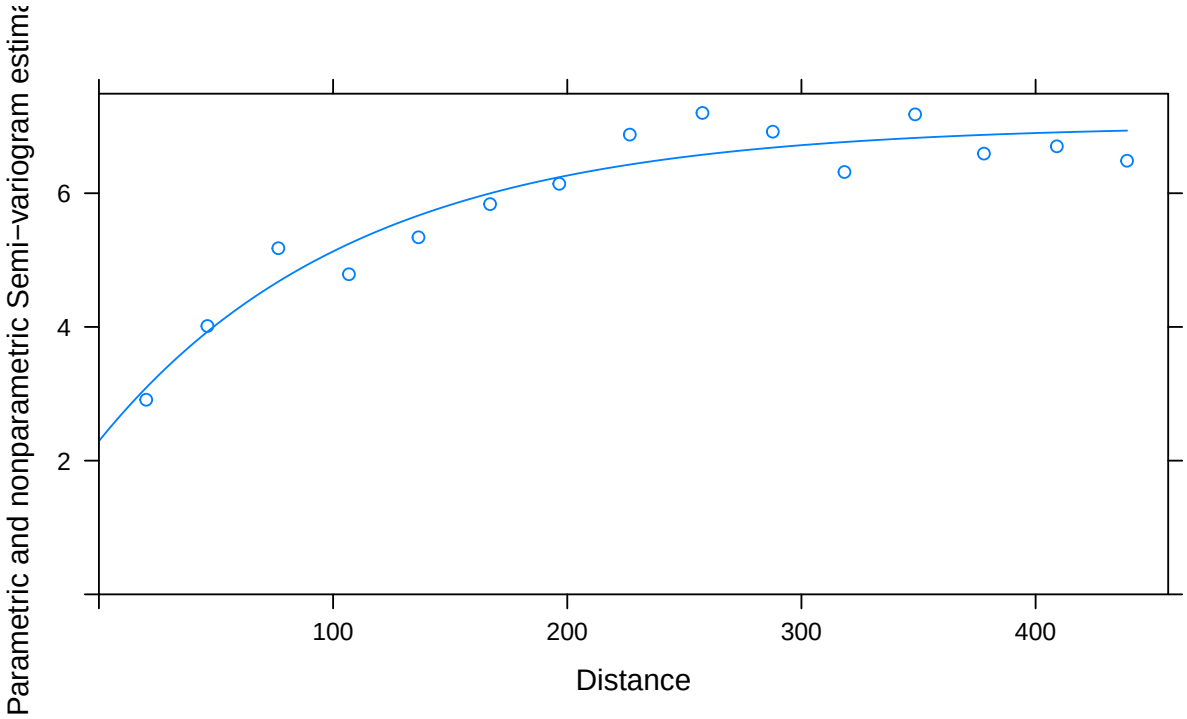
Part b: Estimate the variogram nonparametrically and then fit the exponential variogram to it using weighted least squares. Make and include a plot of the nonparametric and parametric variogram functions. Also store your parameter estimates and report them.

Below, we see that the points on the plot represent the nonparametric variogram estimate and the line representes the fitted exponetnial variogram, using weighted least squares. There seems to be some structure to the variogram, and the exponential seems to be a decent fit. I have also included my parameter estimates below.

```
##   model    psill    range
## 1   Nug 2.292900   0.0000
## 2   Exp 4.728362 109.1394
```

```
## [1] "So, my estimate for sigma^2 is 4.72836154182719 , my estimate for rho is"
```

```
## [1] "109.1394349959 , and my estimate for tau is  2.29290009780337 ."
```



Part c: We will now form the GLS estimate of β by hand, rather than using the gls function. (This function doesn't handle longitude and latitude well, and I also want to give you some practice with matrix calculations in R.)

- Use the rdist.earth function in fields to create a matrix of distances (in miles) between pairs of locations in CAtemp.
- Create the covariance matrix, plugging in your estimates from the fitted variogram. *Hint: Sum two matrices, one without a nugget and one using the diag function to create the matrix $\tau^2 I$.*
- Invert the covariance matrix and store it for later reference.
- Create the X matrix. *Hint: Use cbind.*
- Put all the pieces together to form $\hat{\beta}_{GLS}$

We know that $\hat{\beta}_{GLS} = (X^T \hat{\Sigma}^{-1} X)^{-1} X^T \hat{\Sigma}^{-1} Y$.

I have included my estimates for $\hat{\beta}_{GLS}$ below.

```
##      Intercept      lon       lat    elevation
## [1,]  354.3439 2.612875 0.5669455 -0.009034265
```

These estimates seem to be fairly close to the estimates from the linear model, included below for reference.

```
##   (Intercept)           lon           lat     elevation
## 321.511433492   2.324104683   0.564680460  -0.009647649
```

Part d: Calculate and plot the EBLUP of $\mu + Z$ at the locations in CAgrid, plugging in your estimates from (b) and (c). Calculate and plot the (estimated) standard error of Z at each prediction location.

After we've estimated β and θ, we can apply the kriging predictor:

$$\hat{Y}(s_0) = X^T \hat{\beta}_{GLS} + \gamma^T \Sigma^{-1} (Y - X^T \hat{\beta}_{GLS})$$

As shown in the class notes, this gives us the EBLUP and a naive estimate of our prediction error.

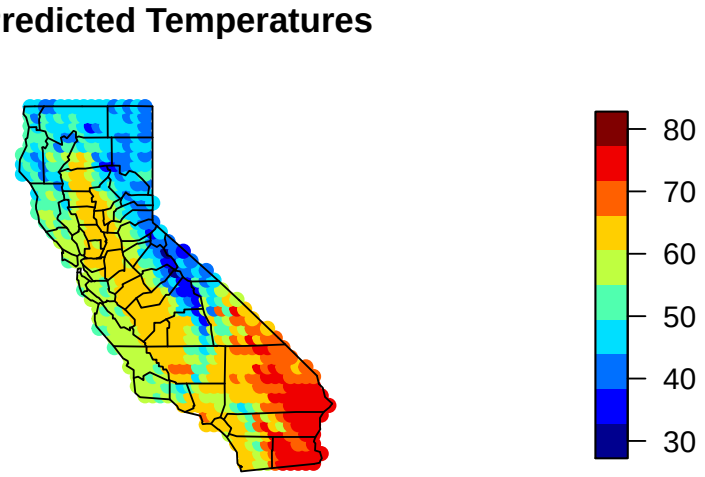
We have also included the estimated standard error of Z at each prediction location, where

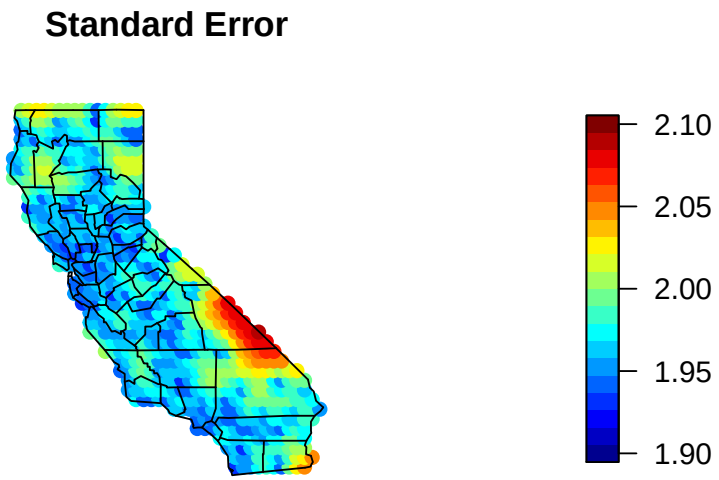
$$Var[\hat{Y}(s_0) - Y(s_0)] = \sigma^2 - \gamma^T \Sigma^{-1} \gamma + b^T (X^T \Sigma^{-1} X)^{-1} b$$

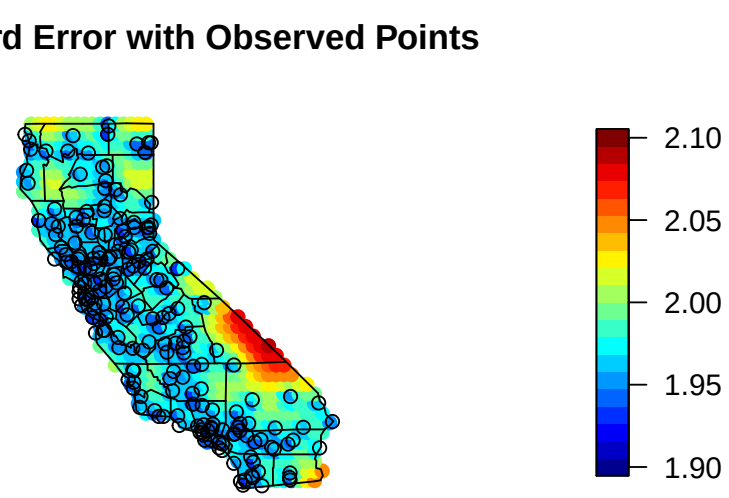
and where $b = x_0 - X^T \Sigma^{-1} \gamma$.

We see that the estimate for the mean temperature is lowest in the mountains and highest in the south, where there are deserts. It is also relatively high in the valley "where all of our vegetables are grown."

We also see that the standard error ($se = \sqrt{(variance)}$) also has some spatial dependence. In the last image, I have included the observation points on the image. We see that the standard errors is the highest where we don't have observation points, especially on the lower east side of California.




Standard Error
2.10
2.05
2.00
1.95
1.90


Standard Error with Observed Points
2.10
2.05
2.00
1.95
1.90

Appendix

I have included my code to all of the problems below.

```
# Problem 1c

# Defining how many points I want on the interval- 500 equally spaced
n <- 500
# I want 3 simulations of a Gaussian Process
nsim <- 3
# I want a Gaussian Process on [0,1]
tmax <- 1
# Initializing a matrix to store all of the realizations of different Gaussian Processes
y.mat <- matrix(NA, n+1, nsim)
# creating a x-vector for the x-axis on the plots, will be used in some of the functions
# for the mean and the variance
x.vec <- (0:n)/n*tmax

#Mean Functions
mu1 <- rep(5,n+1)
mu2 <- rep(0,n+1)
mu3 <- rep(15,n+1)

#Covariance Functions
#diagonal covariance
Sigma1 <- diag(1,n+1)
# Standard Brownian Bridge Process
Sigma2 <- matrix(NA,nrow=length(x.vec),ncol=length(x.vec))
#x.vec <- x.vec[-c(1,501)]
for(i in 1:length(x.vec)){
  for(j in 1:length(x.vec)){
    Sigma2[i,j]=min(x.vec[i],x.vec[j])*(1-max(x.vec[i],x.vec[j]))
  }
}
Sigma3 <- matrix(1, nrow=501,ncol=501)


y.mat[,1] <- multi_norm(mu1, Sigma1)
y.mat[,2] <- multi_norm(mu2, Sigma2)
y.mat[,3] <- multi_norm(mu3, Sigma3)

matplot(x.vec, y.mat, type = "l", col=c(4,2,3),lwd=2, xlab = "t", ylab = "X(t)",
        lty = c(1,3,5), main = "Plot of my 3 realizations of a GP")
```

```
#Problem 2 exploratory analysis
library(sp)
library(gstat)
library(fields)
library(classInt)
library(maps)

load("C:/Users/ckell/OneDrive/Penn State/2017-2018/597/spatial_statistics_597/Homework 1/data/CAtemps.R

ploteqc <- function(spobj, z, breaks, ...){
```

```
  pal <- tim.colors(length(breaks)-1)
  fb <- classIntervals(z, n = length(pal),
                       style = "fixed", fixedBreaks = breaks)
  col <- findColours(fb, pal)
  plot(spobj, col = col, ...)
  image.plot(legend.only = TRUE, zlim = range(breaks), col = pal)
}

## Plotting

#range(CAtemp$avgtemp)
breaks <- 40:75
ploteqc(CAtemp, CAtemp$avgtemp, breaks, pch = 19)
map("county", region = "california", add = TRUE)
title(main = "Average Annual Temperatures, 1961-1990, Degrees F")

#range(CAgrid$elevation)
breaks <- seq(-100, 3600, by = 100)
ploteqc(CAgrid, CAgrid$elevation, breaks, pch = 19)
map("county", region = "california", add = TRUE)
title(main = "Elevations at prediction locations, m")
```

```
##
# Problem 2a
# creating preliminary estimate
##
#head(CAtemp)

#Fitting the preliminary linear model
linmod <- lm(avgtemp~lon+lat+elevation, data=CAtemp)
summary(linmod)
#storing the residuals so I can plot them
CAtemp$resid <- linmod$resid
#storing the coefficients for later
beta_lin <- linmod$coefficients
beta_lin

#plotting the residuals
#range(CAtemp$resid)
breaks <- seq(-7, 7, by = 0.1)
ploteqc(CAtemp, CAtemp$resid, breaks, pch = 19)
map("county", region = "california", add = TRUE)
title(main = "Residuals")
```

```
##
# Problem 2b
##


##  Nonparametric estimation of the variogram
vg <- variogram(resid ~ 1, data = CAtemp)#, width=75)
#plot(vg, xlab = "Distance", ylab = "Nonparametric Semi-variogram estimate", width=5)

##  Fitting the variogram parametrically
```

```r
fitvg <- fit.variogram(vg, vgm(1, "Exp", range=200, nugget=1), fit.method = 2)
print(fitvg)
s2.hat <- fitvg$psill[2]
rho.hat <- fitvg$range[2]
tau2.hat <- fitvg$psill[1]

paste("So, my estimate for sigma^2 is", s2.hat, ", my estimate for rho is")
paste(rho.hat, ", and my estimate for tau is ", tau2.hat, ".")

plot(vg, fitvg, xlab = "Distance", ylab = "Parametric and nonparametric Semi-variogram estimate")
```

```r
##
# Problem 2c
##

# Use the rdist.earth function in fields to create a matrix of distances (in miles)
# between pairs of locations in CAtemp.
#   There are some entries on the diagonal that are not zero, but this is also true
#   in the case in the documentation, so I won't worry too much about it!
d <- rdist.earth(coordinates(CAtemp), miles = TRUE)

# Create the covariance matrix, plugging in your estimates from the fitted variogram.
# add two matrices together
mat1 <- diag(tau2.hat, ncol = 200, nrow=200)
#mat2 <- Matern(dist_mat, scale = s2.hat, range = rho.hat, smoothness = 0.5)
mat2 <- Exponential(d, phi=s2.hat, range=rho.hat)
# phi = Marginal variance, Exponential: phi* exp( -d/range)
cov <- mat1+mat2


# Invert the covariance matrix and store it for later reference.
Sinv <- solve(cov)

# Create the X matrix. Hint: Use cbind.
n <- nrow(CAtemp)
m <- nrow(CAgrid)
X <- cbind(rep(1,n), CAtemp$lon, CAtemp$lat, CAtemp$elevation)
```

```r
# Put all the pieces together to form $\hat{\beta}_{GLS}$

Y <- CAtemp$avgtemp
X <- as.matrix(X)
first <- t(X)%*%Sinv%*%X
second <- t(X)%*%Sinv%*%Y
beta_gls <- solve(first)%*%second

# include estimates below
rownames(beta_gls) <- c("Intercept", "lon", "lat", "elevation")
t(beta_gls)
```

```r
##
# Problem 2d
##
```

```
dcross <- rdist.earth(coordinates(CAtemp), coordinates(CAgrid))
dpred <- rdist.earth(coordinates(CAgrid))
Xpred <- cbind(rep(1,m), CAgrid$lon, CAgrid$lat, CAgrid$elevation)

#Construct the covariance matrixes
Gamma <- exp(-d/rho.hat)
Ginv <- solve(Gamma)
g <- exp(-dcross/rho.hat)
Gpred <- exp(-dpred/rho.hat)

#Kriging equations: mean
y_pred <- Xpred%*%beta_gls + t(g)%*%Ginv%*%(Y-X%*%beta_gls)
# Plot
#range(y_pred)
breaks <- seq(30, 80, by = 5)
ploteqc(CAgrid, y_pred, breaks, pch = 19)
map("county", region = "california", add = TRUE)
title(main = "Predicted Temperatures")

#Kriging equations: standard error
#as in CA temp notes
v_pred <- s2.hat*(Gpred - t(g)%*%Ginv%*%g)

#as in lecture 4 notes
b <- t(Xpred) - t(X) %*% Ginv%*%g
v_pred <- s2.hat-diag(t(g)%*%Ginv%*%g+t(b)%*%solve(t(X)%*%Ginv%*%X)%*%b)
se_pred <- sqrt(v_pred)


#range(se_pred)
breaks <- seq(1.9, 2.1, by = 0.01)
ploteqc(CAgrid, se_pred, breaks, pch = 19)
map("county", region = "california", add = TRUE)
title(main = "Standard Error")

breaks <- seq(1.9, 2.1, by = 0.01)
ploteqc(CAgrid, se_pred, breaks, pch = 19)
map("county", region = "california", add = TRUE)
title(main = "Standard Error with Observed Points")
points(CAtemp)
```