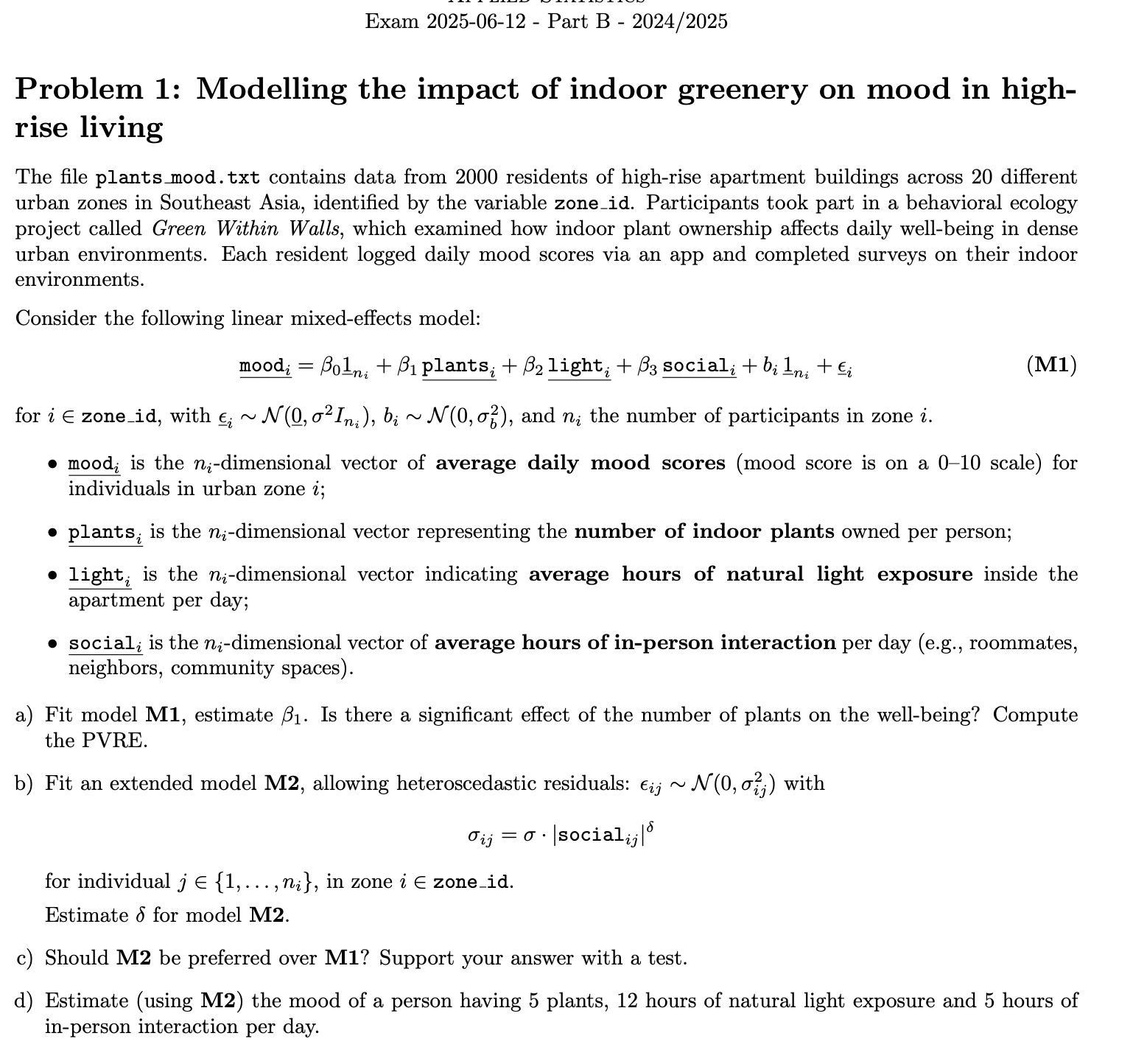
# 2025-06-12

## LMM

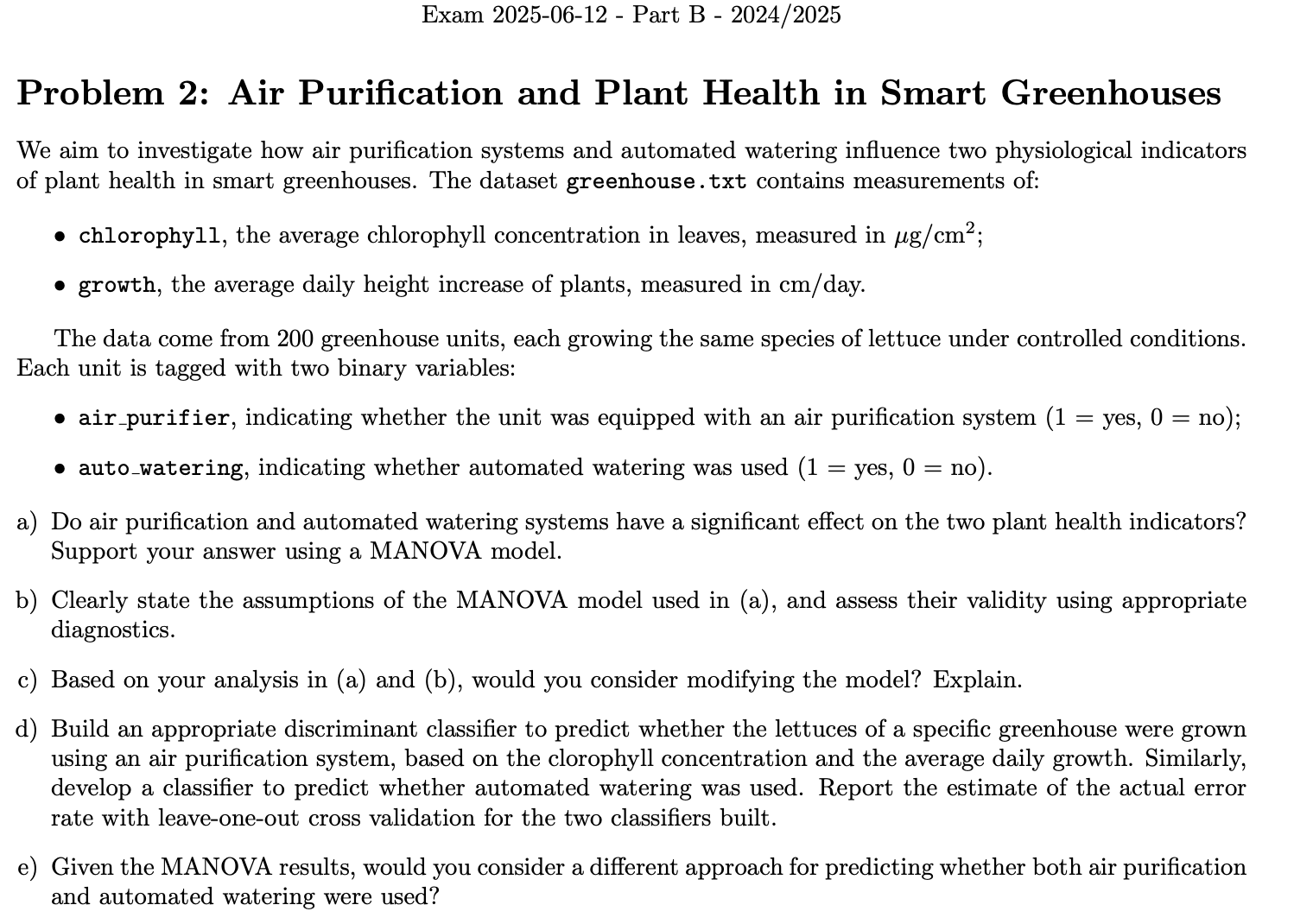
****

* 1. Random intercept
  2. PVRE. get\_variance\_random(lme\_model), get\_variance\_residual(lme\_model)
  3. heteroscedastic residuals (estimate params) lme(mood ~ plants + light + social, random = ~1|as.factor(zone\_id), weights = varPower(form = ~social) ,data = plants\_mood)
  4. anova(model1, model2)
  5. value of the test statistic (from anova - L.Ratio)
  6. Prediction in new group (lme)

predict(lme\_model2, test\_data, level = FALSE)

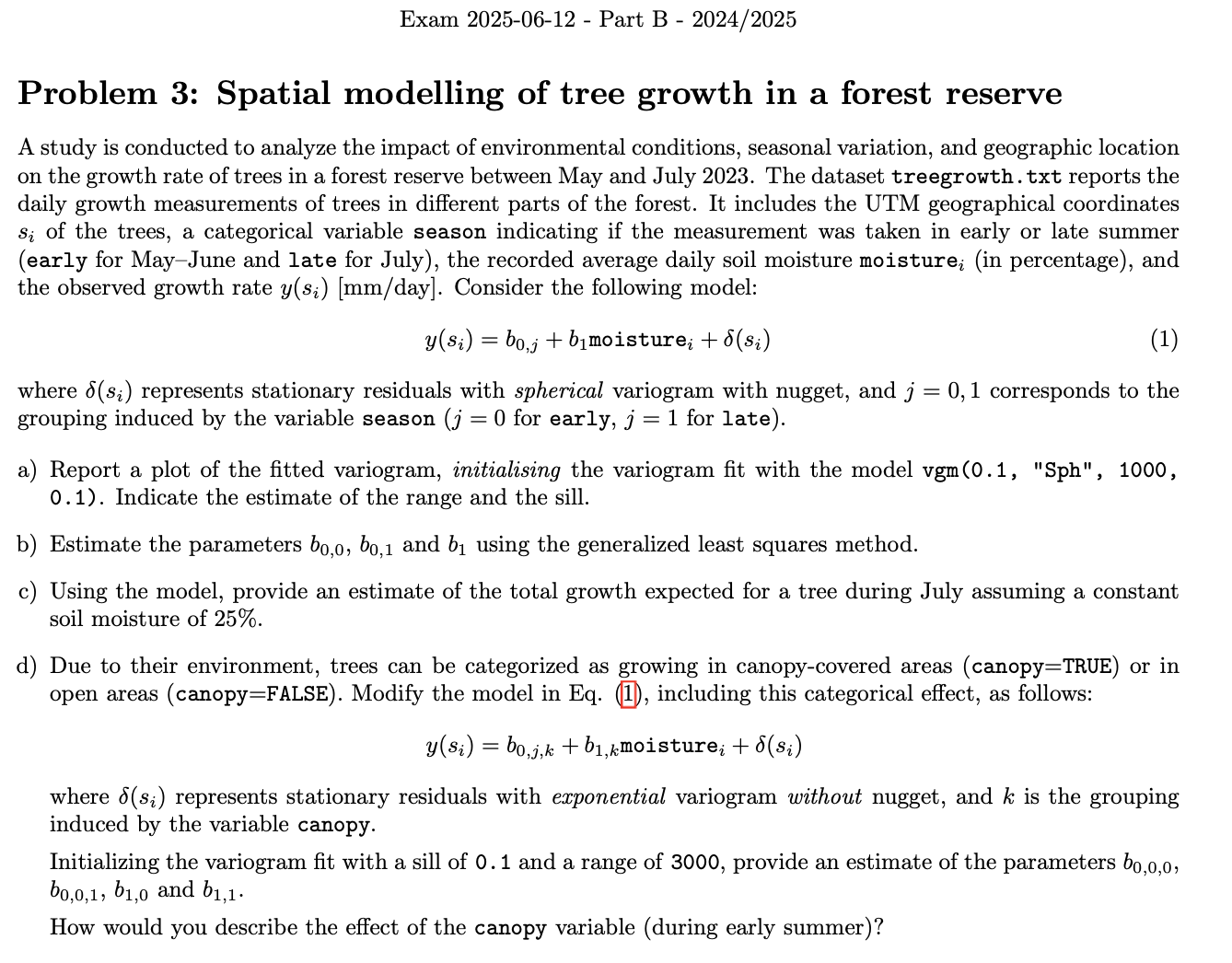
# level = FALSE - for new group, TRUE - for existing group

## Two-ways MANOVA + LDA

****

* 1. Do they have an effect? (YES)
  2. Verify MANOVA assumptions
  3. Make two classifier (LDA - because of satisfy assumptions)
  4. Leave-one-out cross-validation. lda(greenhouse\_data, AP, CV=TRUE)
  5. One classifier for 4 classes, instead of two classifiers (because we have interaction effect)

## GEOSTAT

****

* 1. variogram
  2. Range and Sill (sill - SUM OF sill from nugget and model)
  3. Estimate parameters of model (b0, b01 …). make prediction for different points and solve system

s0.new=data.frame(x=0.0, y=0.0, moisture=0.0, season=FALSE)

coordinates(s0.new)=c('x','y')

predict(g.tr, s0.new, BLUE = TRUE)

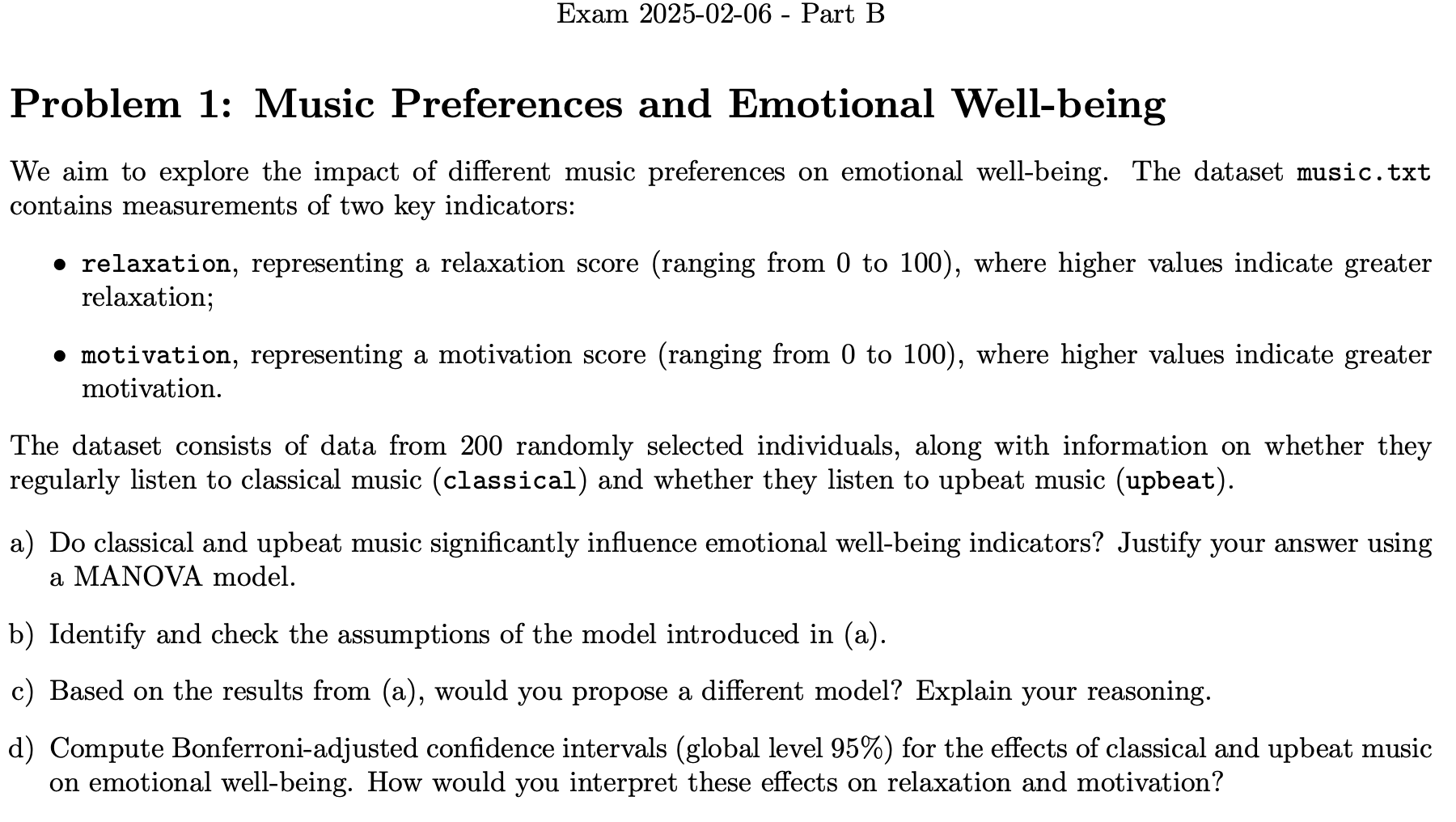
* 1. geostat model with factors:

g2.tr <- gstat(formula = growth ~ -1 + as.factor(season):as.factor(canopy) + moisture:as.factor(canopy), data = treegrowth, model = v2.fit)

* 1. Estimate parameters of model (b00, b001 …)

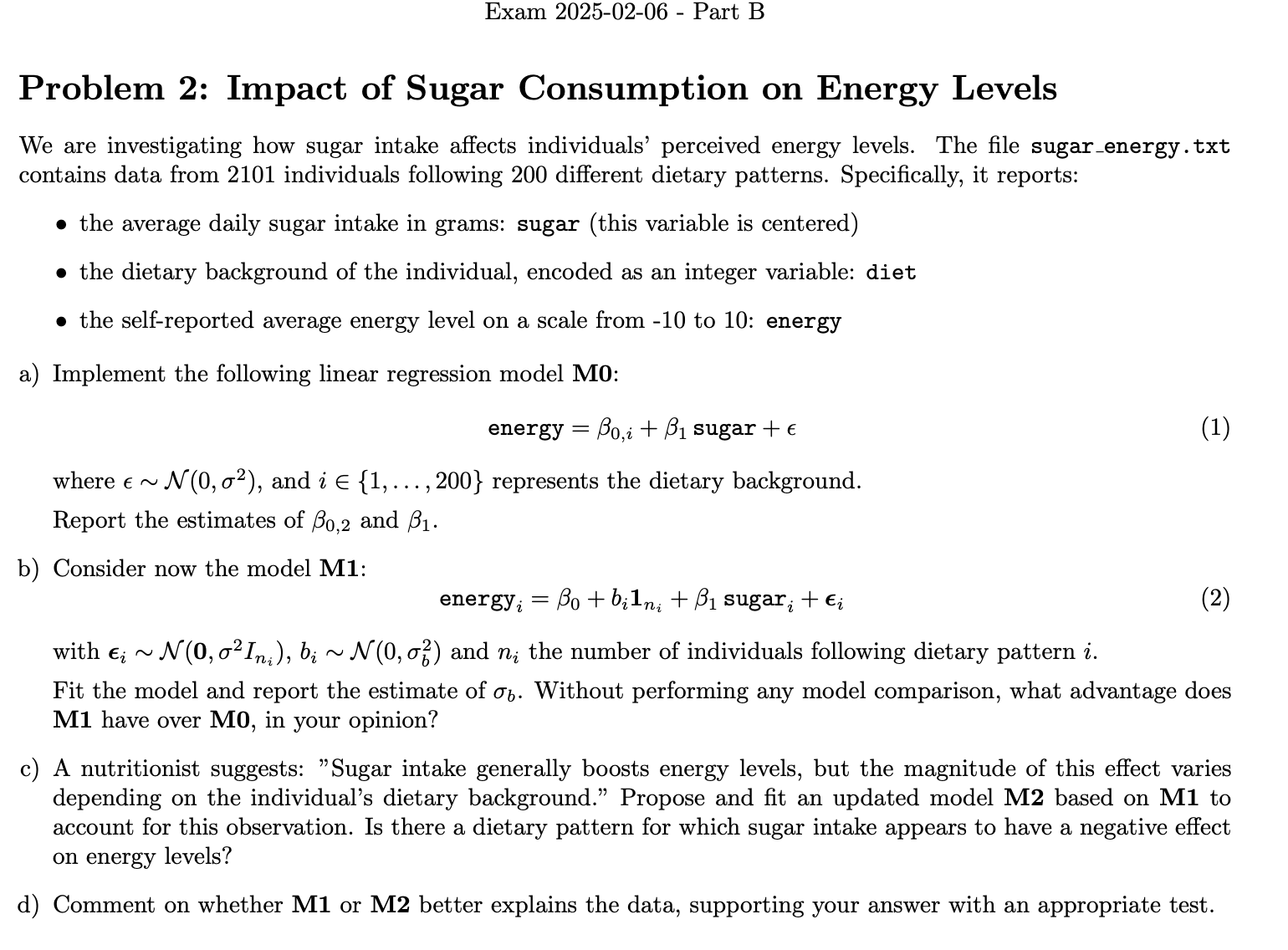
# 2025-02-06

## Two-ways MANOVA + CI



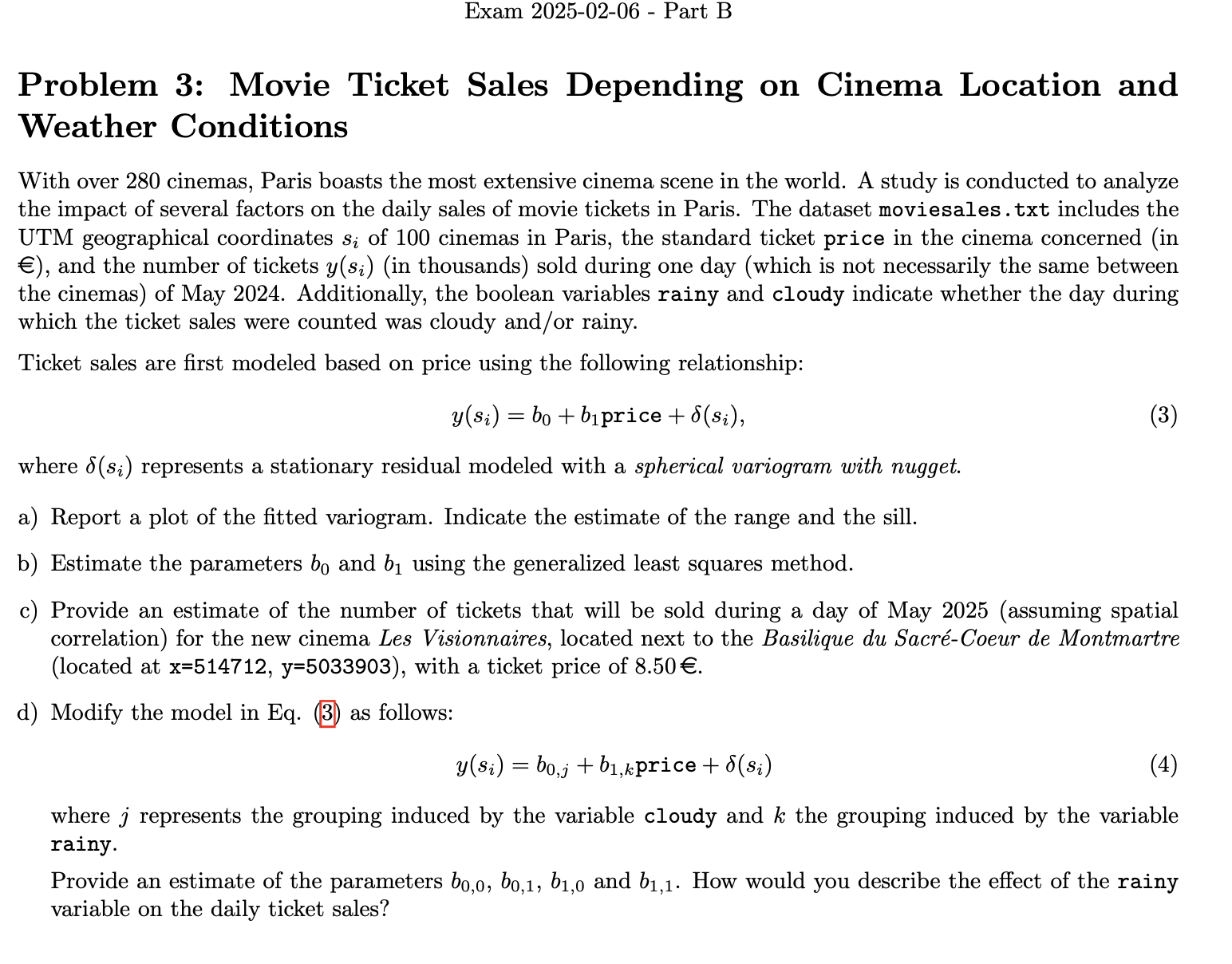
* 1. Do they have an interaction effect? (NO)
  2. Verify MANOVA assumptions
  3. Confidence Intervals for two factors and interpretation (no interaction)
  4. Confidence Intervals if there is interaction effect

## LMM (description of difference between standard model/intercept/slope)



* 1. Random intercept
  2. sigma\_b - from random intercept
  3. Random random intercept and random slope
  4. VarCorr(lme\_model22) - to get sigma for intercept
  5. Random intercept and random slope use less params, that using of dummy variables for each group random = ~1+sugar|as.factor(diet)
  6. Number of params (my task)

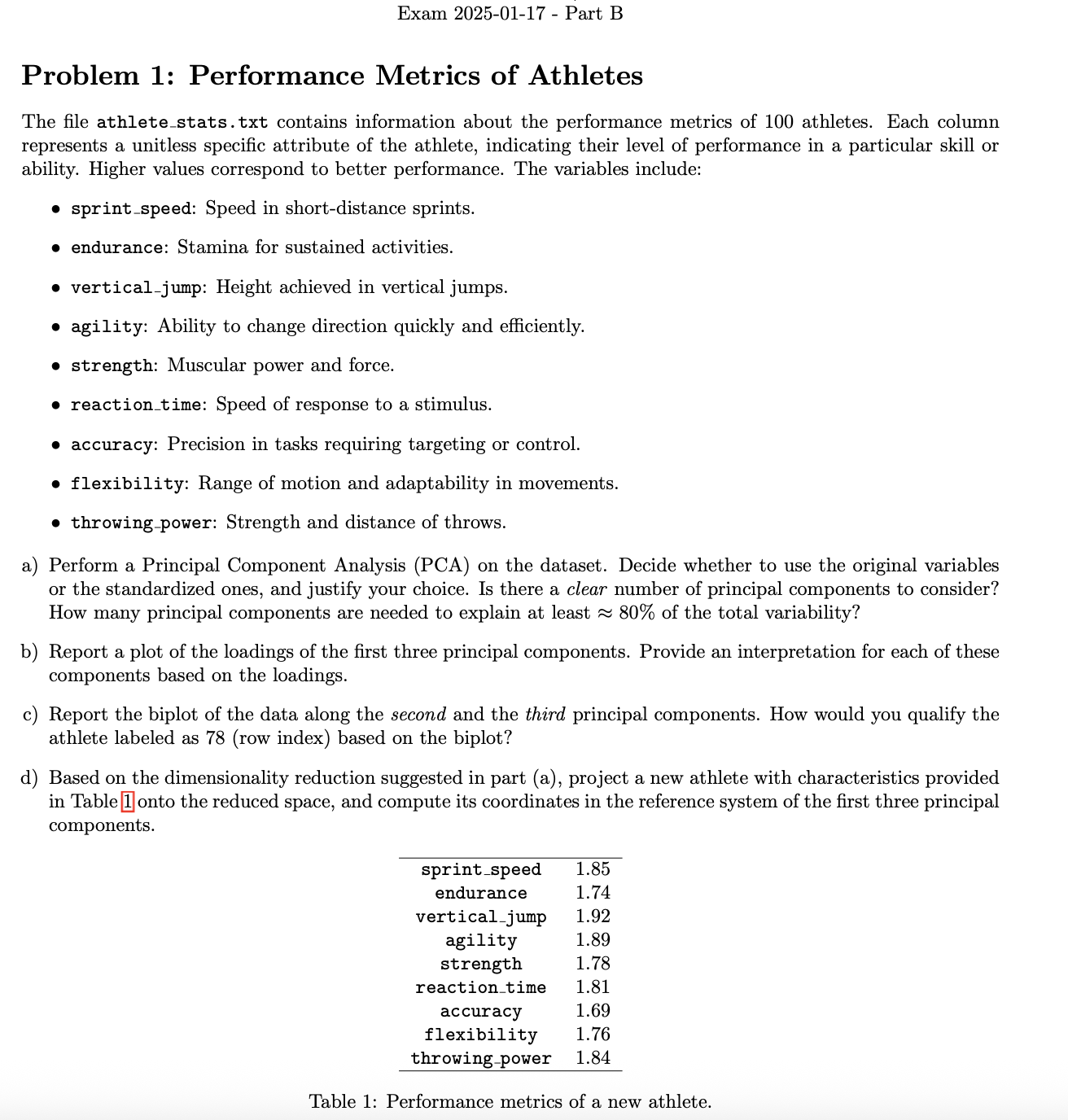
## GEOSTAT



* 1. variogram
  2. Range and Sill
  3. model with factors
  4. Estimate parameters

# 2025-01-17

## PCA

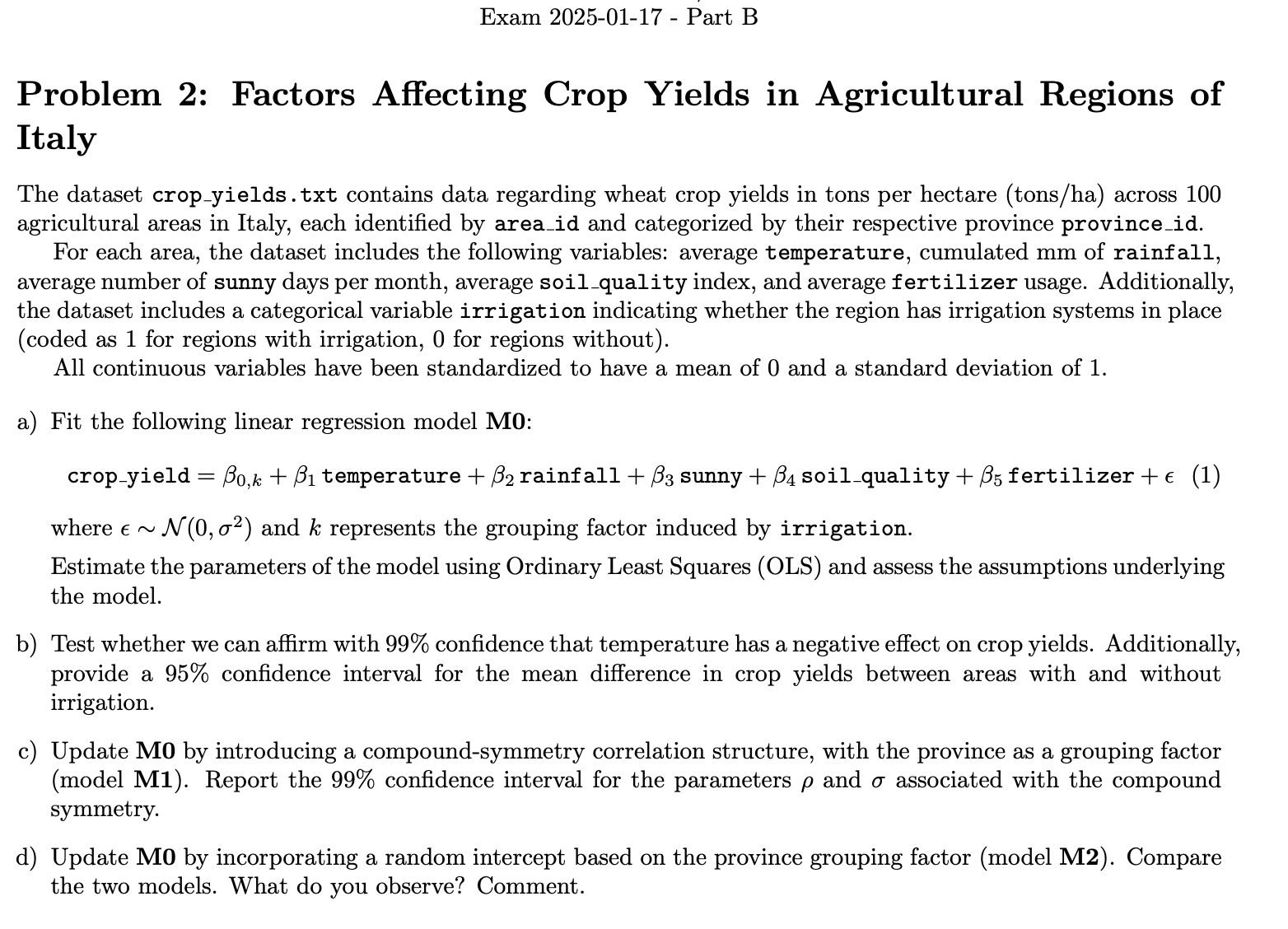


* 1. Perform PCA, Should we standartise?, How many PCs need to explain 80% of variance
  2. Plot of loadings and explain first 3 components
  3. Biplot - in PC2 and PC3 biplot(pc.athlete\_stats, choices = c(2, 3))
  4. Explain one of sample on Biplot (How would you qualify the

athlete labeled as 78 (row index) based on the biplot?

* 1. Prediction new data (provide scores) (Don’t forget, that if you centralised source data, you also have to centralise new observation)

## LMM (mean CI)



* 1. Standart OLS model, need check assumptions (residuals normality and homoscedasticity) + plot to confirm

plot(lm0, which = 1) # plot to confirm that residuals are normal

plot(lm0, which = 2) # plot showing homoscedasticity

shapiro.test(lm0$residuals) # not check normality

plot(lm\_model$residuals) # ALSO plot this to check homogenity

* 1. Check if there is effect of some variable (by its value positive/negative) and Confidence Interval or p-value

confint(lm0, level=0.99)

* 1. Confidence Intervals for means, using lm model

lm0\_b0 <- lm(formula = yield ~ as.factor(irrigation) + temperature + rainfall + sunny + soil\_quality + fertilizer,

data=crop\_yields) # in that case we have common intercept and additional factor for irrigation Yes

# this factor plays role of delta between two groups, so we can use if to compare means in two groups

summary(lm0\_b0)

confint(lm0\_b0, parm="as.factor(irrigation)Yes", level = 0.95)

# [0.3066912 0.5894834] ~ [0.307, 0.589]

* 1. compound-symmetry correlation structure

correlation = corCompSymm(form =~1|as.factor(province\_id))

* 1. Confidence Intervals for Rho and sigma

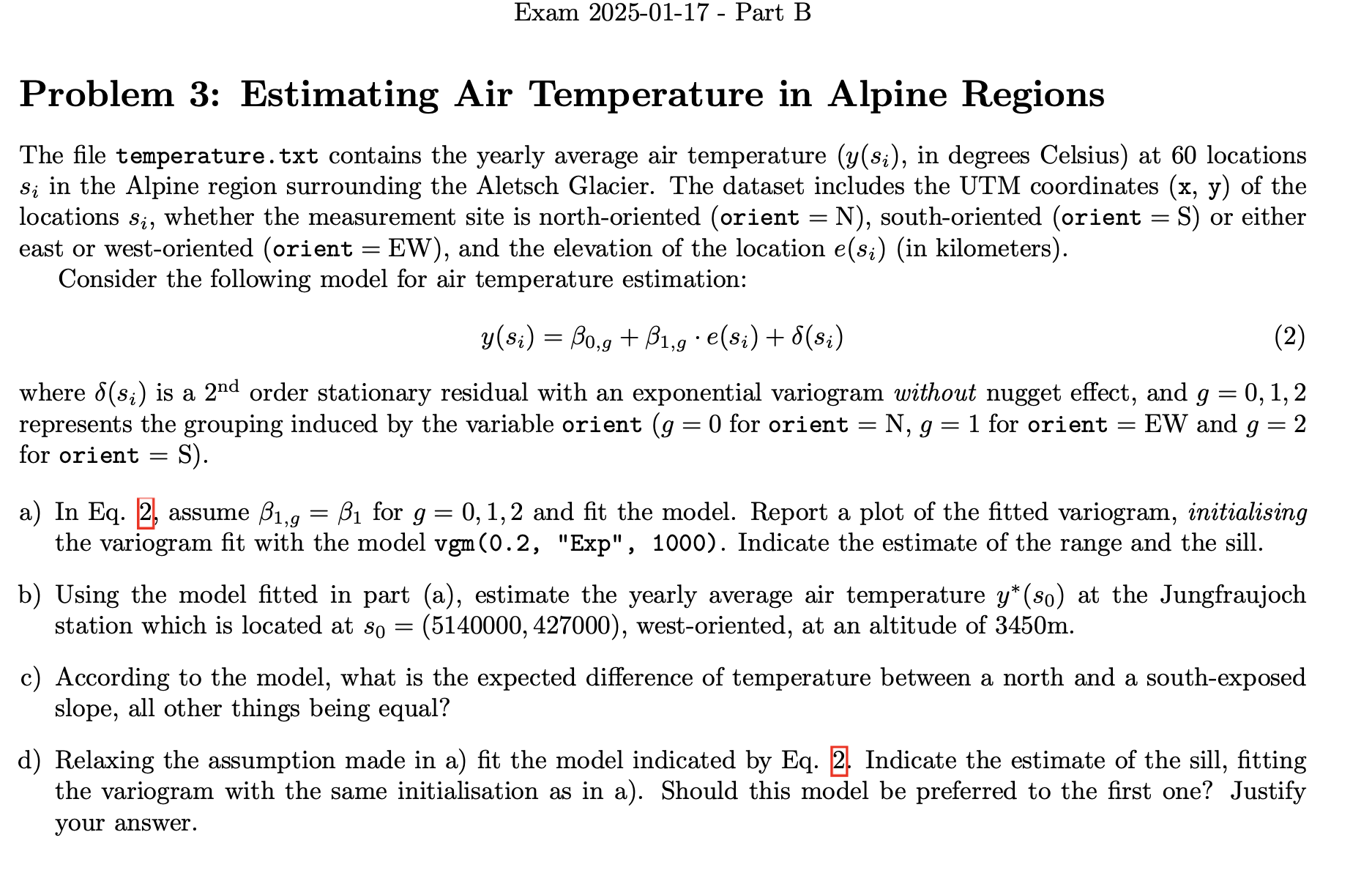
intervals(gls1, which = "var-cov", level = 0.99)

* 1. Random intercept

random = ~1|as.factor(province\_id)

* 1. Compare two models with anova(gls1, lmm2) (have same AIC, BIC => equal)

## GEOSTAT

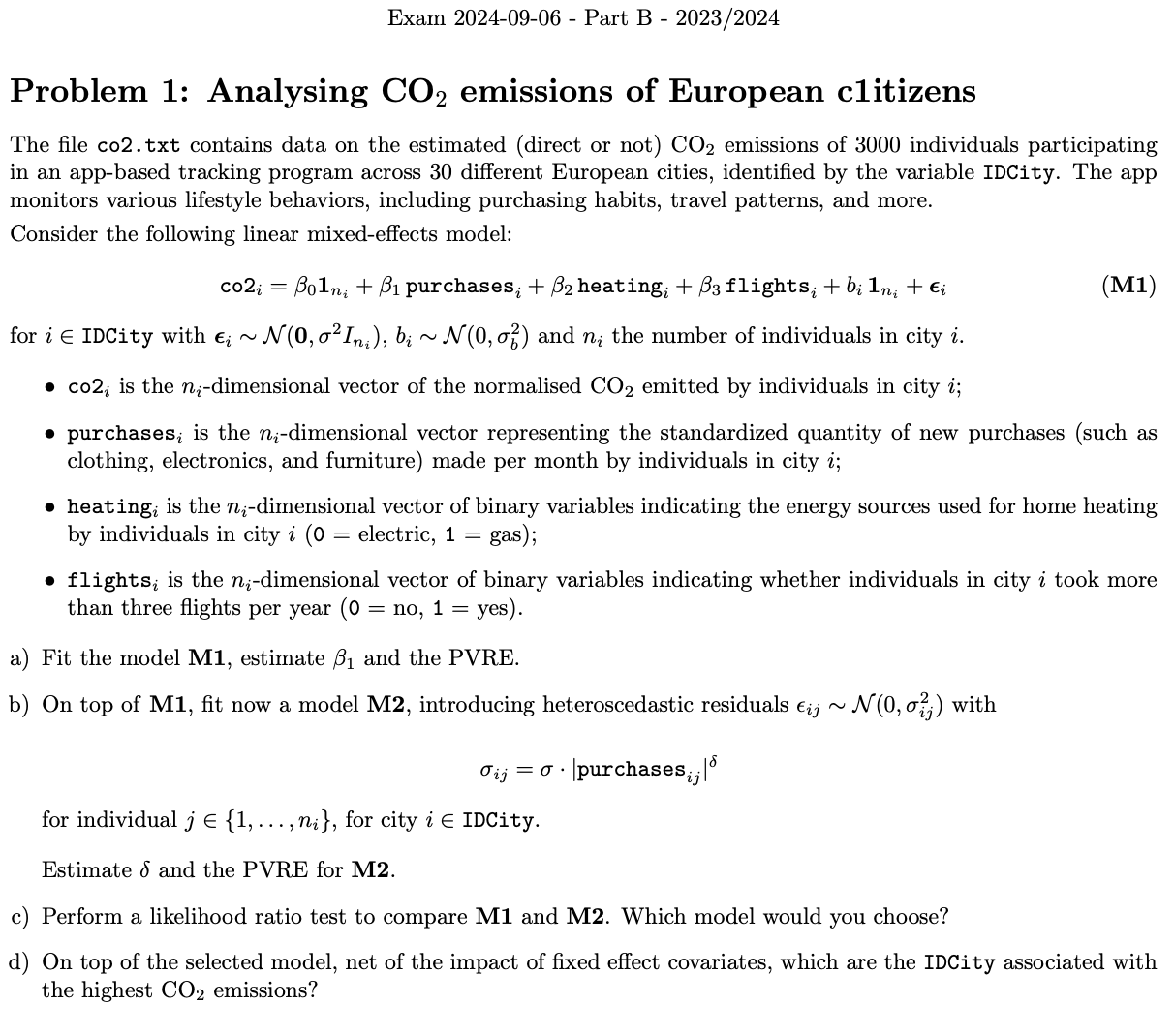


* 1. Variogram, indicate sill and range (For Exp model, computed range will not coincide with plot)
  2. Predict point (with existing group) predict(g.tr, s0.new, BLUE = FALSE)
  3. Difference among groups (b\_0\_1 - b\_0\_2)
  4. Variogram with more factors
  5. Which model is better? (look on smoothness)

Lower sill, bigger range - better model

# 2024-09-06

## LMM



* 1. Model with random intercept random = ~1|as.factor(IDCity)

NOTICE: betta0 - common for all groups, multiplying by 1 just show repeating for all elements

* 1. PVRE,

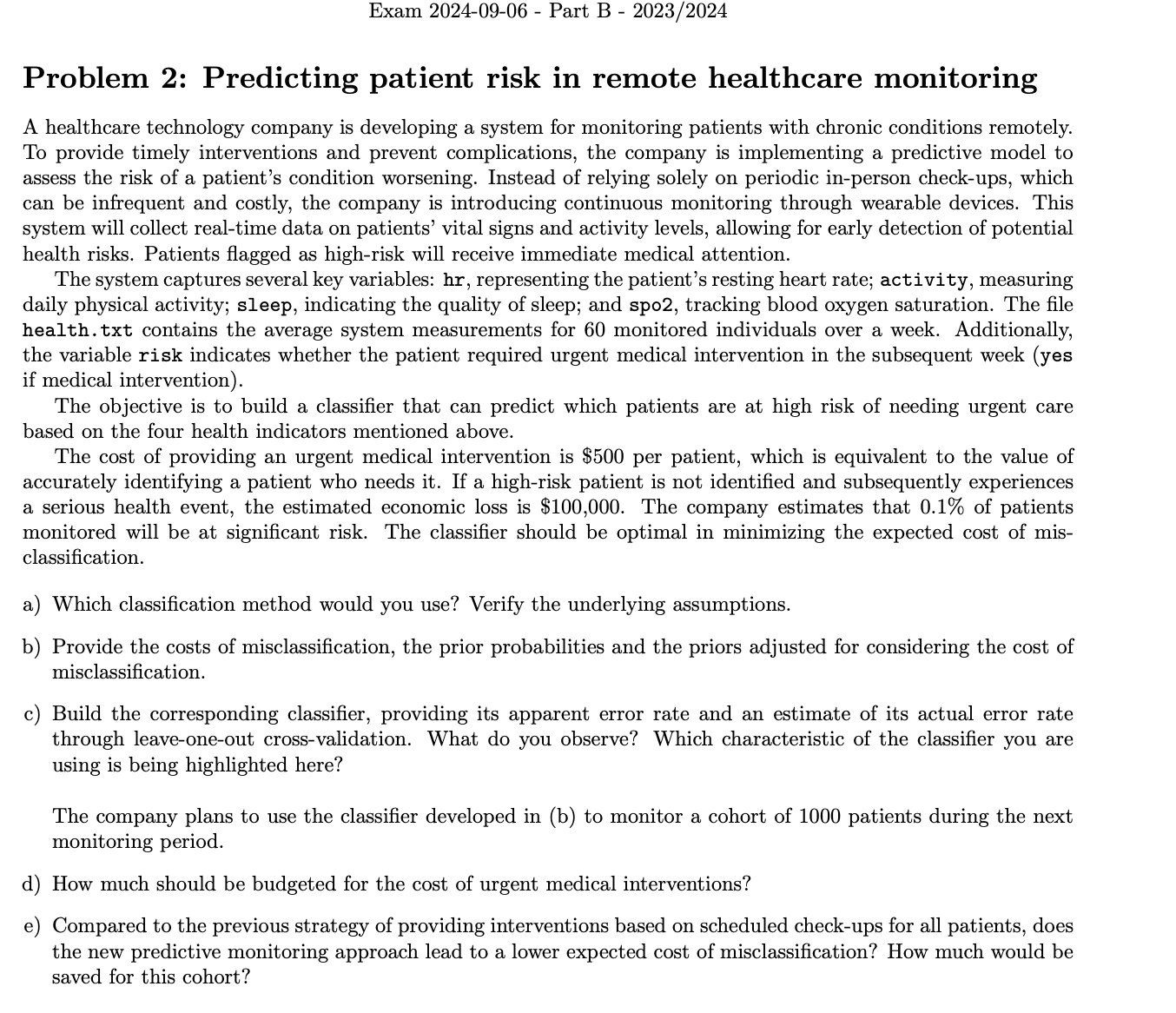
var\_b <- get\_variance\_random(lme1)

var\_eps <- get\_variance\_residual(lme1)

PVRE <- var\_b/(var\_b+var\_eps)

* 1. Heteroscedastic residuals weights = varPower(form = ~purchases)
  2. Compare model anova(lme1, lme2)
  3. Which group have highest (target), based on random intercept ranef(lme2) - to get coefficient of random effect

## Classifier (LDA)



* 1. Which classification model to use

mvn(health\_data[health$risk == "yes", ])

mvn(health\_data[health$risk == "no", ])

boxM(health\_data, health$risk)

S\_yes <- cov(health\_data[yes\_index,])

S\_no <- cov(health\_data[no\_index,])

max(abs(S\_yes/S\_no)) # less than 10

min(abs(S\_yes/S\_no)) # more than 0.1

* + 1. Normal distributed & Homoschedasticity (same covariance structure) => LDA
    2. Normal distributed & NOT homoschedastic => QDA
    3. Not normal distributed & homoschedastic => FDA
  1. Cost of misclassification, prior probabilities, prior adjusted

c\_no\_yes <- 100000 # cost of misclassification c(no|yes)

c\_yes\_no <- 500 # cost of misclassification c(yes|no) c(predicted|real)

p\_yes <- 0.001

p\_no <- 1 - p\_yes

p\_adj\_yes <- p\_yes\*c\_no\_yes / (p\_yes\*c\_no\_yes + p\_no\*c\_yes\_no)

p\_adj\_no <- p\_no\*c\_yes\_no / (p\_yes\*c\_no\_yes + p\_no\*c\_yes\_no)

* 1. Build classifier (in our case it was LDA)

lda.m\_cv <- lda(health\_data, health$risk, prior=c(p\_adj\_no, p\_adj\_yes), CV=TRUE) # for leave one out CV and AER

lda.m <- lda(health\_data, health$risk, prior=c(p\_adj\_no, p\_adj\_yes)) # for APER

* 1. Compute APER, AER

Lda.s <- predict(lda.m)

table(class.true=health$risk, classe.assigned=Lda.s$class)

n\_no <- sum(health$risk == "no")

n\_yes <- sum(health$risk == "yes")

APER <- 2\*p\_no/n\_no + 4\*p\_yes/n\_yes # 2 and 4 taken from Confusion Matrix

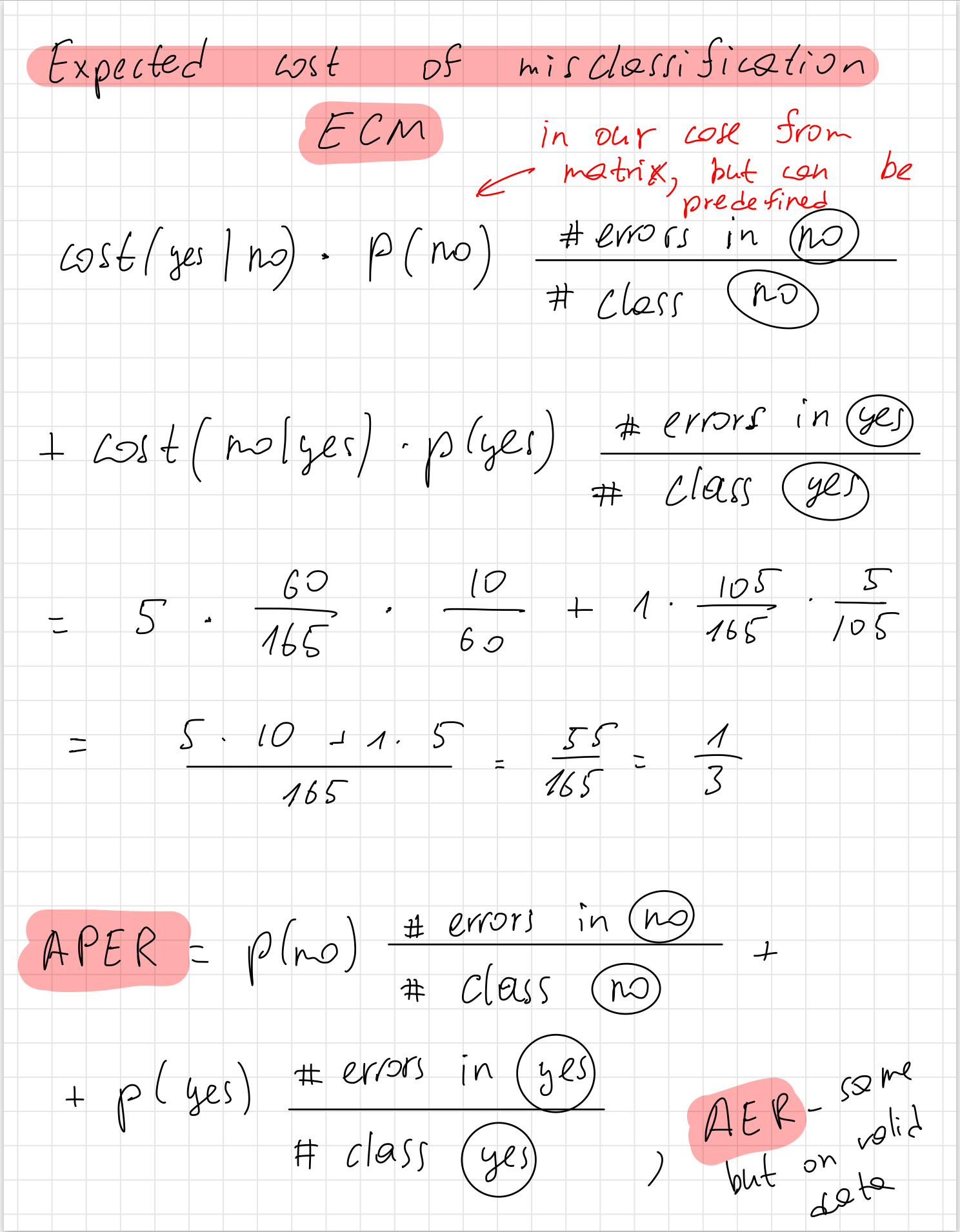
table(class.true=health$risk, class.assignedCV=lda.m\_cv$class)

AER <- p\_no\*2/n\_no + p\_yes\*5/n\_yes

* 1. Expected Misclassification Cost (EMC) (for 1000 patients) (BUT here was trickt question about budget - use column of predicted as risk patients - all of them reuire 500$)

(c\_yes\_no\*2\*p\_no/n\_no + c\_no\_yes\*4\*p\_yes/n\_yes) \* 1000 # EMC

* 1. Comparison with other strategy (when everyone predicted as risk patient - (yes in label))



## FunctionalDA (B-splines)



* 1. Fit B-spline, GCV, Penalization

m <- 4 # "using a basis of CUBIC B-splines", then we have to use norder = 4

nrow(stelvio) # 126 points

# nbasis = norder + number\_of\_knots

# number\_of\_knots = 126 - 2 (knot divide space, so -2)

# => 124 knots, norder = 124 + 4 = 128

nbasis <- 128 # number of splines same basis$nbasis

basis <- create.bspline.basis(rangeval=c(0,25), nbasis=nbasis, norder=m)

functionalPar <- fdPar(fdobj=basis, Lfdobj=m-2, lambda=1)

Xobs0 <- stelvio$altitude # like Y - target

Xss <- smooth.basis(abscissa, Xobs0, functionalPar)

gcv <- Xss$gcv # the value of the gcv statistic

* 1. Estimate approximate dimension (df) and plots

df <- Xss$df # number of parameters (so dimension of space)

Xss0 <- eval.fd(abscissa, Xss$fd, Lfd=0) # zero derivative (fitted curve)

Xss1 <- eval.fd(abscissa, Xss$fd, Lfd=1) # firs derivative

par(mfrow=c(1,2))

plot(abscissa, Xss0, type = "l", lwd = 2, col="red", ylab="fitted curve (RED)")

points(abscissa, Xobs0)

plot(abscissa, Xss1, type = "l", lwd = 2, col="blue", ylab="first derivative")

* 1. Optimal lambda

lambda <- 10^seq(-1,3, by = 0.5)

gcv <- numeric(length(lambda)) # GCV - general cross validation

for (i in 1:length(lambda)){

functionalPar <- fdPar(fdobj=basis, Lfdobj=m-2, lambda=lambda[i])

gcv[i] <- smooth.basis(abscissa, Xobs0, functionalPar)$gcv

}

par(mfrow=c(1,1))

plot(log10(lambda),gcv)

lambda[which.min(gcv)]

abline(v = log10(lambda[which.min(gcv)]), col = 2)

opt\_lambda <- lambda[which.min(gcv)]

cat("Optimal lambda:", opt\_lambda, "\nGCV error:", min(gcv), "\n")

* 1. Calculate the slope at the steepest point of the ascent.

max\_slope <- max(Xss1\_refit)

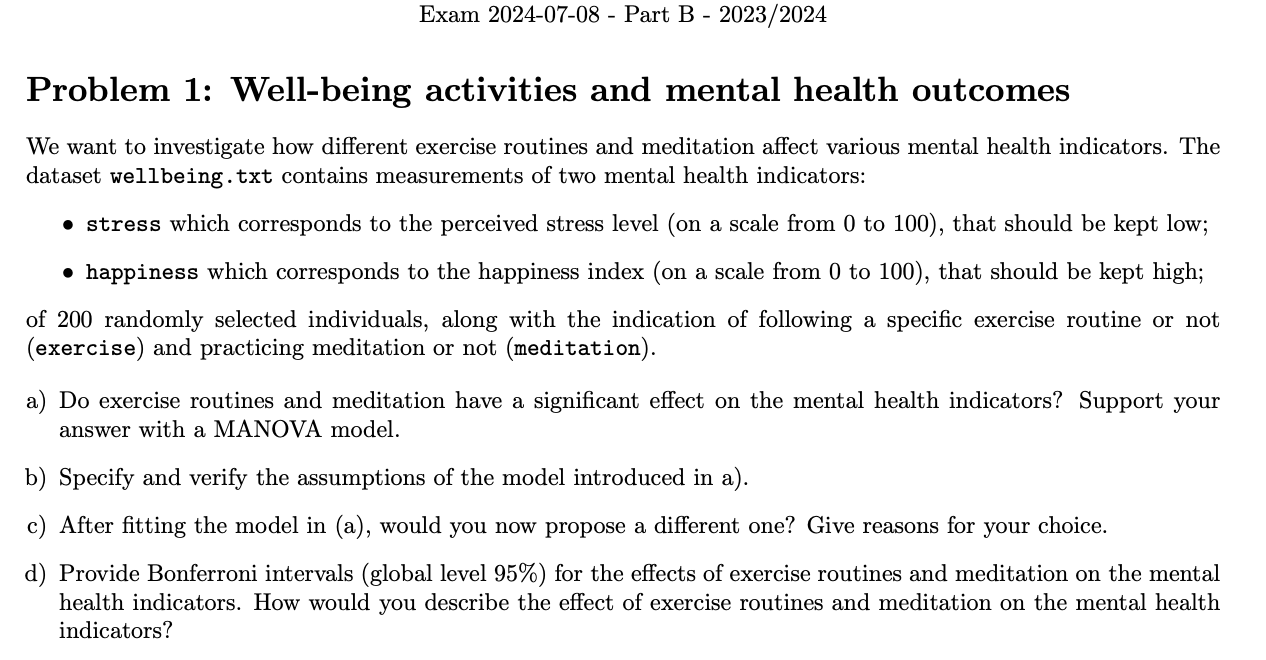
max\_index <- which.max(Xss1\_refit)

max\_distance <- abscissa[max\_index]

max\_distance # 11.2

# 2024-07-08

## Two-ways MANOVA + CI



* 1. Do factors have an effect?
  2. Verify assumptions
  3. Here we didn’t have interaction effect => simplify model
  4. Bonferroni CI (no interaction effect)

alpha <- 0.05

g <- 2 # number of levels of first factor

b <- 2 # number of levels of second factor

p <- 2 # num of features which means we compare

n <- 50

N <- n\*g\*b # 200

W <- summary.manova(fit2)$SS$Residuals

# how many comparisons?

k <- p\*g\*(g-1)/2 + p\*b\*(b-1)/2

# because we have: g levels on the first factor on p components

# b levels on the second factor on p components

k

qT <- qt(1 - alpha/(2\*k), g\*b\*n-g-b+1) # g\*b\*n-g-b+1 = 197 degree of freedom from manova

# the degrees of freedom of the residuals on the additive model are

# g\*b\*n-g-b+1

sum(meditation=='TRUE') # 100,

sum(meditation=='FALSE') # 100

# so divide by 1/100 + 1/100

m\_med\_true <- sapply(wellbeing\_data[meditation=='TRUE',], mean)

m\_med\_false <- sapply(wellbeing\_data[meditation=='FALSE',], mean)

inf\_med <- m\_med\_true-m\_med\_false - qT \* sqrt(diag(W)/(g\*b\*n-g-b+1) \* (1/100+1/100))

sup\_med <- m\_med\_true-m\_med\_false + qT \* sqrt(diag(W)/(g\*b\*n-g-b+1) \* (1/100+1/100))

inf\_med

sup\_med

sum(exercise=='TRUE')

sum(exercise=='FALSE')

m\_exe\_true <- sapply(wellbeing\_data[exercise=='TRUE',],mean)

m\_exe\_false <- sapply(wellbeing\_data[exercise=='FALSE',],mean)

inf\_exe <- m\_exe\_true-m\_exe\_false - qT \* sqrt(diag(W)/(g\*b\*n-g-b+1) \* (1/100+1/100))

sup\_exe <- m\_exe\_true-m\_exe\_false + qT \* sqrt(diag(W)/(g\*b\*n-g-b+1) \* (1/100+1/100))

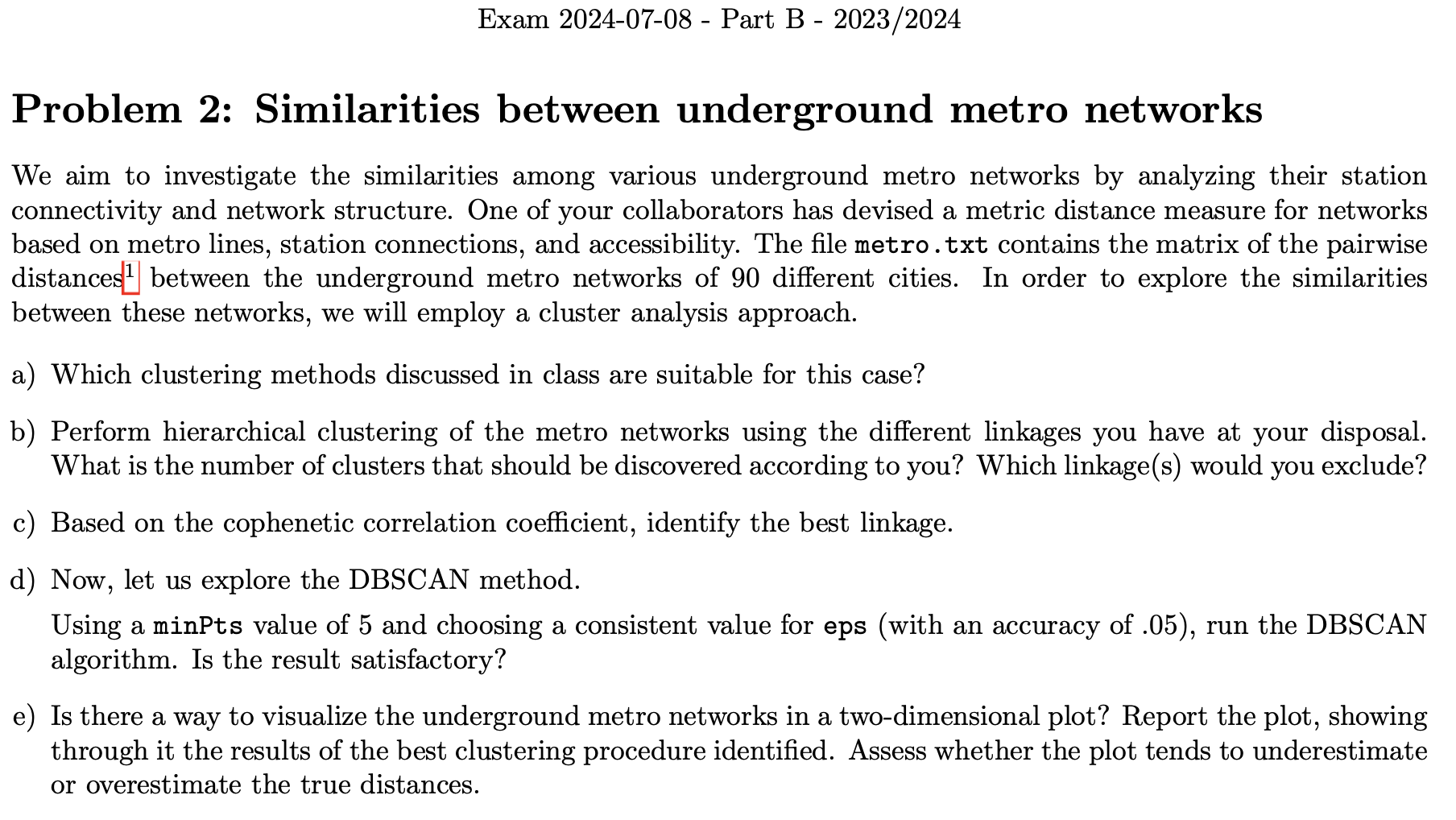
inf\_exe

sup\_exe

IC2 <- list(med\_true\_false=cbind(inf\_med, sup\_med), exe\_true\_false=cbind(inf\_exe, sup\_exe))

IC2

## Clustering



* 1. Clustering methods, that may work only with dissimilarity matrix

# As we work with dissimilarity matrix we can apply these approaches:

# 1. DBSCAN

# 2. Hierarchical clustering with average linkage

# 3. Hierarchical clustering with single linkage

# 4. Hierarchical clustering with complete linkage

# Requires source variabels

# 1. K-means

# 2. Hierarchical clustering with Ward linkage

* 1. Perform Hierarchical clustering (trying different linkage from previous question)

# data.dist <- dist(data, method='euclidean') # how we usually define dist from source data

m.dist <- as.dist(metro) # how to get appropriate format of dist from dist matrix

metro.hclust\_single <- hclust(m.dist, method='single')

plot(metro.hclust\_single)

From hierarchical plots, we can estimate number of clusters

* 1. Cophenetic correlation coefficient

coph.single <- cophenetic(metro.hclust\_single)

single\_cor <- cor(m.dist, coph.single)

single\_cor # 0.5746898 ~ 0.575

* 1. Perform DBSCAN

minPts <- 5 # defince our core points

dbs <- dbscan(m.dist, eps = 0.05, minPts)

dbs

# class 0 - unlabeled points, if they are too many => bad clustering

* 1. From distance to 2-D space

location <- cmdscale(m.dist, k=2)

# perform clustering after transforming to 2-D space

location.dist <- dist(location)

minPts <- 5 # defince our core points

dbs\_loc <- dbscan(location.dist, eps = 0.25, minPts)

dbs\_loc

plot(location[,1], location[,2], type='n', asp=1, axes=FALSE, main="MDS of Metro stations",xlab='',ylab='')

points(location, col = dbs\_loc$cluster + 1L, pch=19)

text(location[,1], location[,2], labels=colnames(as.matrix(m.dist)), cex = 0.75, pos = 3)

* 1. Overestimates/Underestimates

# compare the original matrix d\_ij = d(x\_i,x\_j) and delta\_ij = d(y\_i,y\_j)

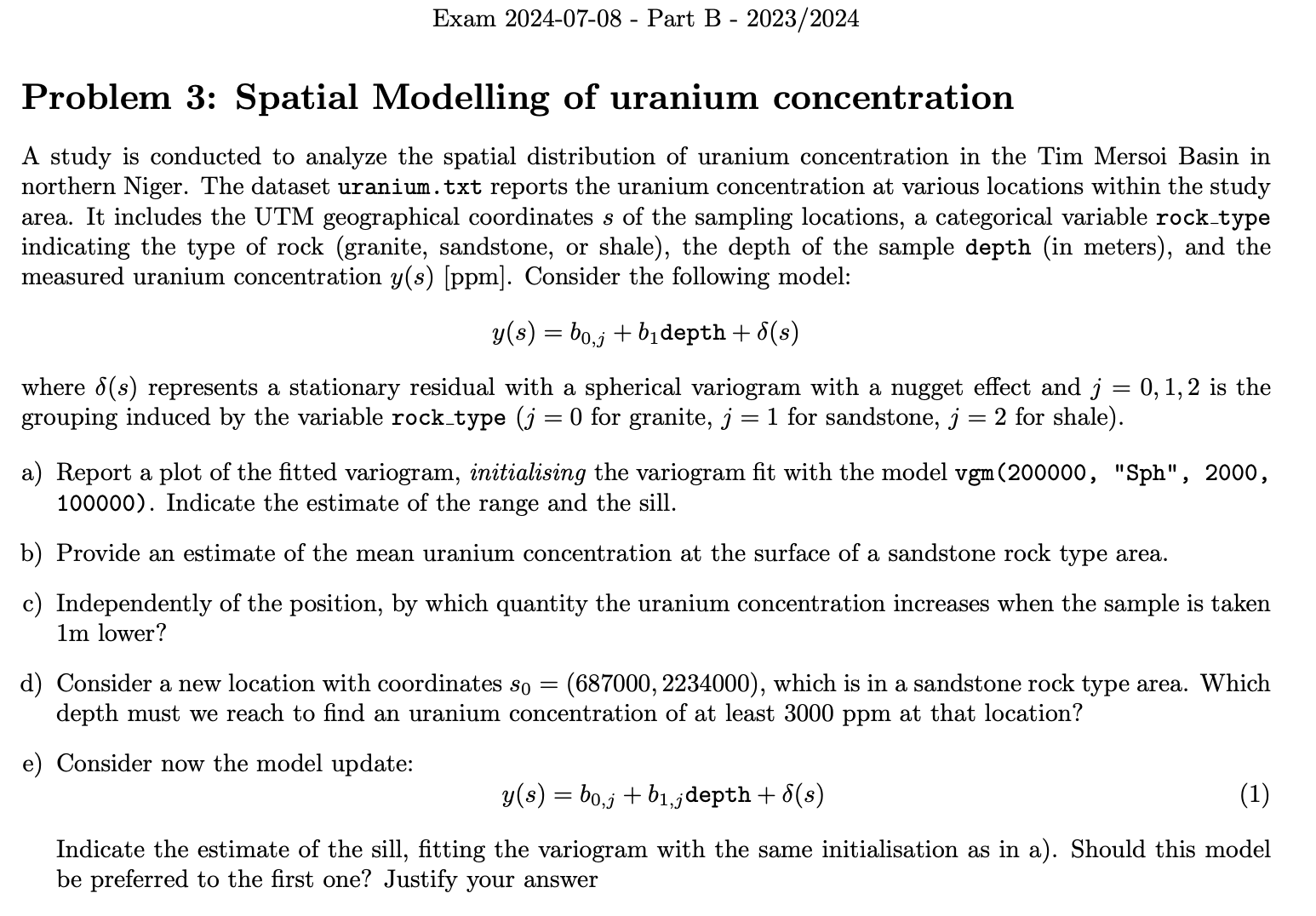
plot(m.dist, dist(location)) # have to be close to line (new distance have to be proportional to source distance)

If points lies **above** diagonal - **overestimates** (distance after MDS > true distance)

If points lies **under** diagonal - **underestimates** (distance after MDS < true distance)

# In our case many points lies above line => overestimates

## GEOSTAT

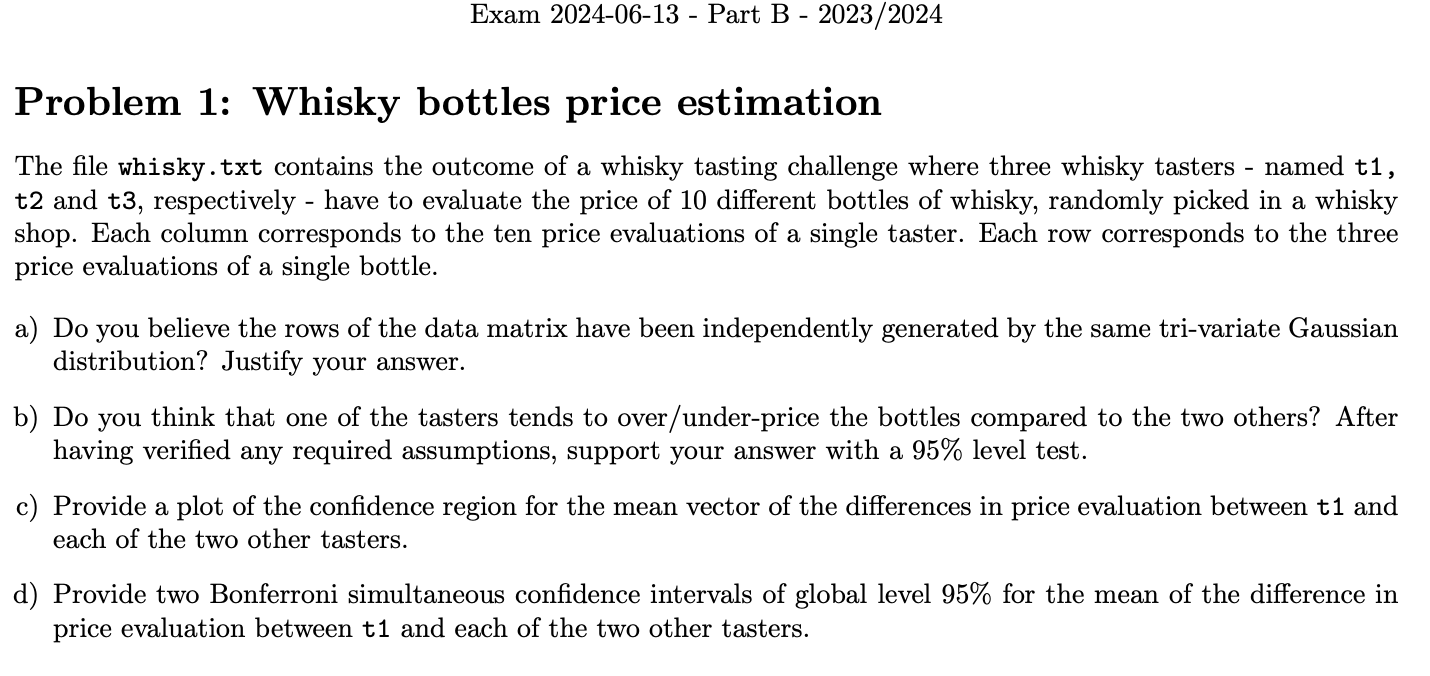


* 1. Fit and plot variogram
  2. Find coefficient (to estimate mean value for specific group) betta\_0\_i (BLUE=TRUE)
  3. Find coefficient betta\_1 (BLUE=TRUE)
  4. Predict in new point (BLUE=FALSE)
  5. Update model with more factors
  6. Which model better

Lower sill, bigger range - better model

# 2024-06-13

## TestMean



4\_TestMean.R - at the end (row 650) for unpaired data (different size of groups)

“Test for the mean of two independent Gaussian populations”

* 1. Check for MVN mvn(whisky)
  2. Hotelling T^2 test (compare means) and Confidence region

boxplot(whisky)

data\_difference <- data.frame(

t12 = whisky$t1 - whisky$t2,

t13 = whisky$t1 - whisky$t3

)

boxplot(data\_difference)

plot(data\_difference)

mvn(data\_difference) # differences

mu0 <- c(0, 0) # H0, that difference is zero

x.mean <- colMeans(data\_difference)

x.cov <- cov(data\_difference)

x.invcov <- solve(x.cov)

n <- 10 # each group has 10 observations

p <- 2 # check two means on zero

x.T2 <- n \* (x.mean-mu0) %\*% x.invcov %\*% (x.mean-mu0)

Pb <- 1-pf(x.T2\*(n-p)/(p\*(n-1)), p, n-p)

Pb # 0.006252078 < 0.05, so we can reject, that mean difference = 0

# Confidence region plot

# mean under H0 (blue)

points(mu0[1], mu0[2], col='blue', pch=16)

# sample mean (black)

points(x.mean[1], x.mean[2], col='black', pch=16)

# we represent the confidence region of level 95%: where does mu0 fall?

alpha <- .05

cfr.fisher <- (p\*(n-1)/(n-p))\*qf(1-alpha,p,n-p)

ellipse(center=x.mean, shape=x.cov/n, radius=sqrt(cfr.fisher), lwd=2)

# as we see, CI doesn't contain 0, so we can reject H0 of zero mean

* 1. Bonferroni CI for means (but in code also presented for variance)

k <- 2 # if we also wanted to plot for variances => k = 4

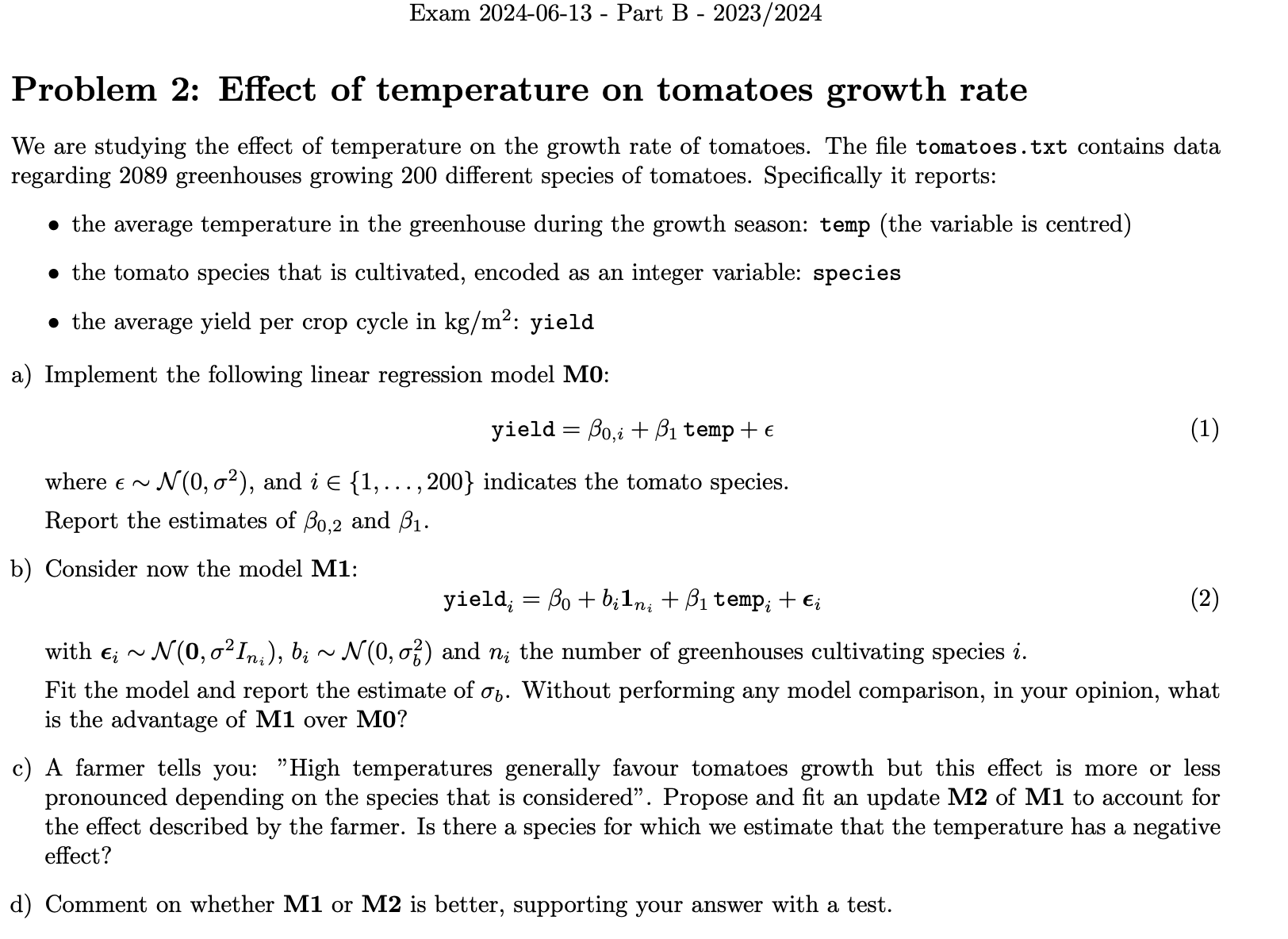
alpha <- 0.05

ICmean <- cbind(inf=x.mean - sqrt(diag(x.cov)/n) \* qt(1 - alpha/(2\*k), n-1),

center= x.mean,

sup= x.mean + sqrt(diag(x.cov)/n) \* qt(1 - alpha/(2\*k), n-1))

## LMM



* 1. First model with Dummy variables and estimate corresponding coefficients
  2. Model with random effect

lme\_model <- lme(yield ~ 1 + temp, random = ~1|as.factor(species), data=tomatoes)

* 1. What the advantage of model with random effect, compared to model with dummy variables:

M1 have less params than M0, so bigger degree of freedom, and will not overfit the data

* 1. Estimate sigma\_b (from random effect)

sqrt(get\_variance\_random(lme\_model))

* 1. Random intercept and random slope

lme\_model2 <- lme(yield ~ 1 + temp, random = ~1+temp|as.factor(species), data=tomatoes)

summary(lme\_model2)

ranef(lme\_model2) # as we can see from coefficents of random effect, there are

# species, that have negative effect from temperature

* 1. What is better Random intercept + slope or only Random Intercept

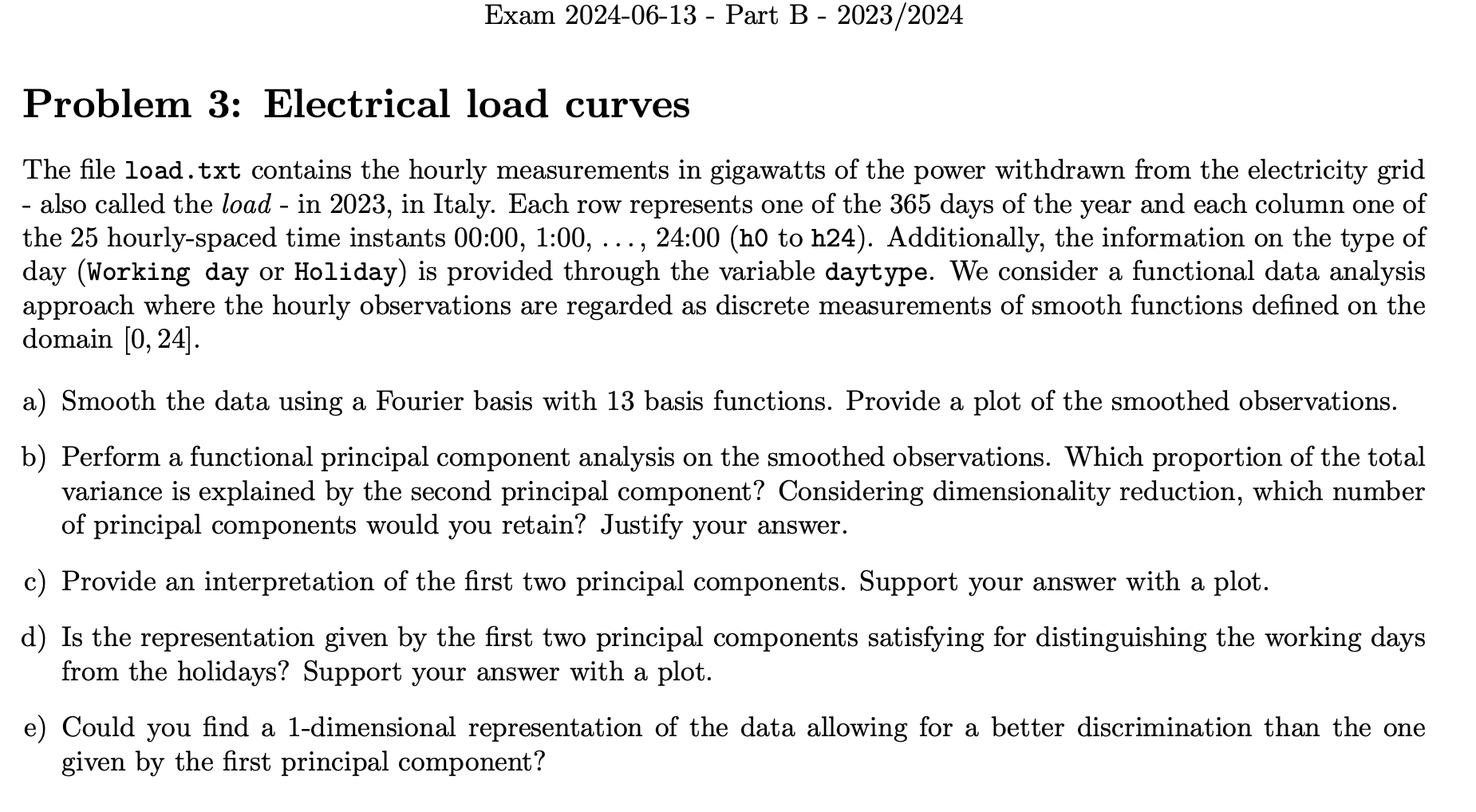
anova(lme\_model, lme\_model2)

# model 2 better, due to p-value < 0.05 (and we reject H0 of equal zero rzndom effects)

# also AIC, BIC values lower for second model

# and Model 2 consider how temeprature affect on growth for each species group

## FunctionalDA (Fourier basis)



* 1. Smoothing with Fourier basis (Look in which domain you have to smooth [0,24] - hours)

nbasis <- 13

time <- seq(0,24, by=1)

basis <- create.fourier.basis(rangeval = c(0,24), nbasis = nbasis)

plot(basis)

data\_W.fd <- Data2fd(y=data, argvals=time, basisobj=basis)

plot.fd(data\_W.fd)

data\_W.fd$coefs[1:3, 1] # 116.344769 -12.747130 -2.863551

data\_W.fd$coefs[, 1]

# Same procedure:

Xsp <- smooth.basis(argvals=time, y=data, fdParobj=basis) #

Xsp0 = eval.fd(time, Xsp$fd)

plot(Xsp) # same as previously

coef(Xsp$fd)[1:3, 1]

* 1. Perform a functional principal component analysis on the smoothed observations. Proportion of total variance, proportion explained by second variabel

pca\_W <- pca.fd(data\_W.fd, nharm=5, centerfns=TRUE)

# scree plot

# pca.fd computes all the 25 eigenvalues, but only the first

# N-1=34 are non-null

plot(pca\_W$values[1:35],xlab='j',ylab='Eigenvalues')

plot(cumsum(pca\_W$values)[1:35]/sum(pca\_W$values),xlab='j',ylab='CPV',ylim=c(0.8,1))

# we have 35 data, (features), so we can not plot more principal components

pca\_W$values

cumsum(pca\_W$values)[1:35]/sum(pca\_W$values) # 13 components are not NAN, so to explain whole data I need 13 components

pca\_W$values[2]/sum(pca\_W$values) # 0.03984729

# second PC exaplins 0.9668994 - 0.9270522 = 0.0398472 part of variance (took from cumsum)

* 1. Interpretation of first two components

media <- mean.fd(data\_W.fd)

plot(media,lwd=2,ylim=c(0,60),ylab='temperature',main='FPC1')

lines(media+pca\_W$harmonics[1,]\*sqrt(pca\_W$values[1]), col=2)

lines(media-pca\_W$harmonics[1,]\*sqrt(pca\_W$values[1]), col=3)

plot(media,lwd=2,ylim=c(0,60),ylab='temperature',main='FPC2')

lines(media+pca\_W$harmonics[2,]\*sqrt(pca\_W$values[2]), col=2)

lines(media-pca\_W$harmonics[2,]\*sqrt(pca\_W$values[2]), col=3)

# temperate climate or not

# Does same as previous Command of the library fda that automatically does these plots

par(mfrow=c(1,2))

plot(pca\_W, nx=100, pointplot=TRUE, harm=c(1,2), expand=0, cycle=FALSE)

* 1. Find 1-dimension representation of data, to separate two groups (I used FDA)

data\_pca <- data.frame(

pc1=scores[, 1],

pc2=scores[, 2]

)

g <- 2

i1 <- which(load$daytype=='Holiday')

i2 <- which(load$daytype=='Working day')

n1 <- length(i1)

n2 <- length(i2)

n <- n1+n2

m <- colMeans(data\_pca)

m1 <- colMeans(data\_pca[i1,])

m2 <- colMeans(data\_pca[i2,])

S1 <- cov(data\_pca[i1,])

S2 <- cov(data\_pca[i2,])

Sp <- ((n1-1)\*S1+(n2-1)\*S2)/(n-g)

# covariance between groups (estimate) # should be here 1/(g-1)?

B <- 1/(g-1)\*(cbind(m1 - m) %\*% rbind(m1 - m) +

cbind(m2 - m) %\*% rbind(m2 - m)

)

B

# covariance within groups (estimate)

Sp

# how many coordinates?

g <- 2

p <- 2

s <- min(g-1,p)

s

# system on which we project the data (in PCA it's orhogonal basis, but here no)

# Matrix Sp^(-1/2)

val.Sp <- eigen(Sp)$val

vec.Sp <- eigen(Sp)$vec

invSp.2 <- 1/sqrt(val.Sp[1])\*vec.Sp[,1]%\*%t(vec.Sp[,1]) + 1/sqrt(val.Sp[2])\*vec.Sp[,2]%\*%t(vec.Sp[,2])

invSp.2

# ration variability between groups compared to variability within groups

# spectral decomposition of Sp^(-1/2) B Sp^(-1/2)

spec.dec <- eigen(invSp.2 %\*% B %\*% invSp.2)

# first canonical coordinate

a1 <- invSp.2 %\*% spec.dec$vec[,1] # not exact eigen vector, but multiplyed by invSp

a1

# second canonical coordinate

a2 <- invSp.2 %\*% spec.dec$vec[,2]

a2

cc1.data <- as.matrix(data\_pca)%\*%a1

cc2.data <- as.matrix(data\_pca)%\*%a2

coord.cc <- cbind(cc1.data,cc2.data)

plot.fisher.directions <- function() {

plot(data\_pca[, 1], data\_pca[, 2], col = ifelse(load$daytype == "Working day", "blue", "red"),

pch = 19, xlab = "PC1", ylab = "PC2", main = "Scores of First Two Principal Components", ylim=c(-50,50), xlim=c(-70, 70))

legend("topright", legend = c("Working day", "Holiday"), col = c("blue", "red"), pch = 19)

abline(h=0,v=0, col='grey35')

arrows(x0=0, y0=0, x1=a1[1], y1=a1[2], length=.1)

arrows(x0=0, y0=0, x1=a2[1], y1=a2[2], length=.1)

text(a1[1], a1[2], 'a1', pos=2)

text(a2[1], a2[2], 'a2',pos=3)

arrows(x0=0, y0=0, x1=a1[1], y1=a1[2], length=.1)

arrows(x0=0, y0=0, x1=a2[1], y1=a2[2], length=.1)

abline(coef=c(0,(a1[2]/a1[1])), col='grey55',lty=2)

abline(coef=c(0,(a2[2]/a2[1])), col='grey55',lty=2)

points(cc1.data\*a1[1]/(sum(a1^2)),cc1.data\*a1[2]/(sum(a1^2)),col=ifelse(load$daytype == "Working day", "blue", "red"))

points(cc2.data\*a2[1]/(sum(a2^2)),cc2.data\*a2[2]/(sum(a2^2)),col=ifelse(load$daytype == "Working day", "blue", "red"))

}

# as we see among first component we have higher variability between - so better

# classification

# if we don't have lot of groups - we more try to separate our groups,

# (course of dimensionality - as bigger dimension - as easier separate groups)

# purpose of

# and our a1, a2 vectors not orthogonal

plot.fisher.directions()

a1 # so by a1, we can find better direction to discriminate our data

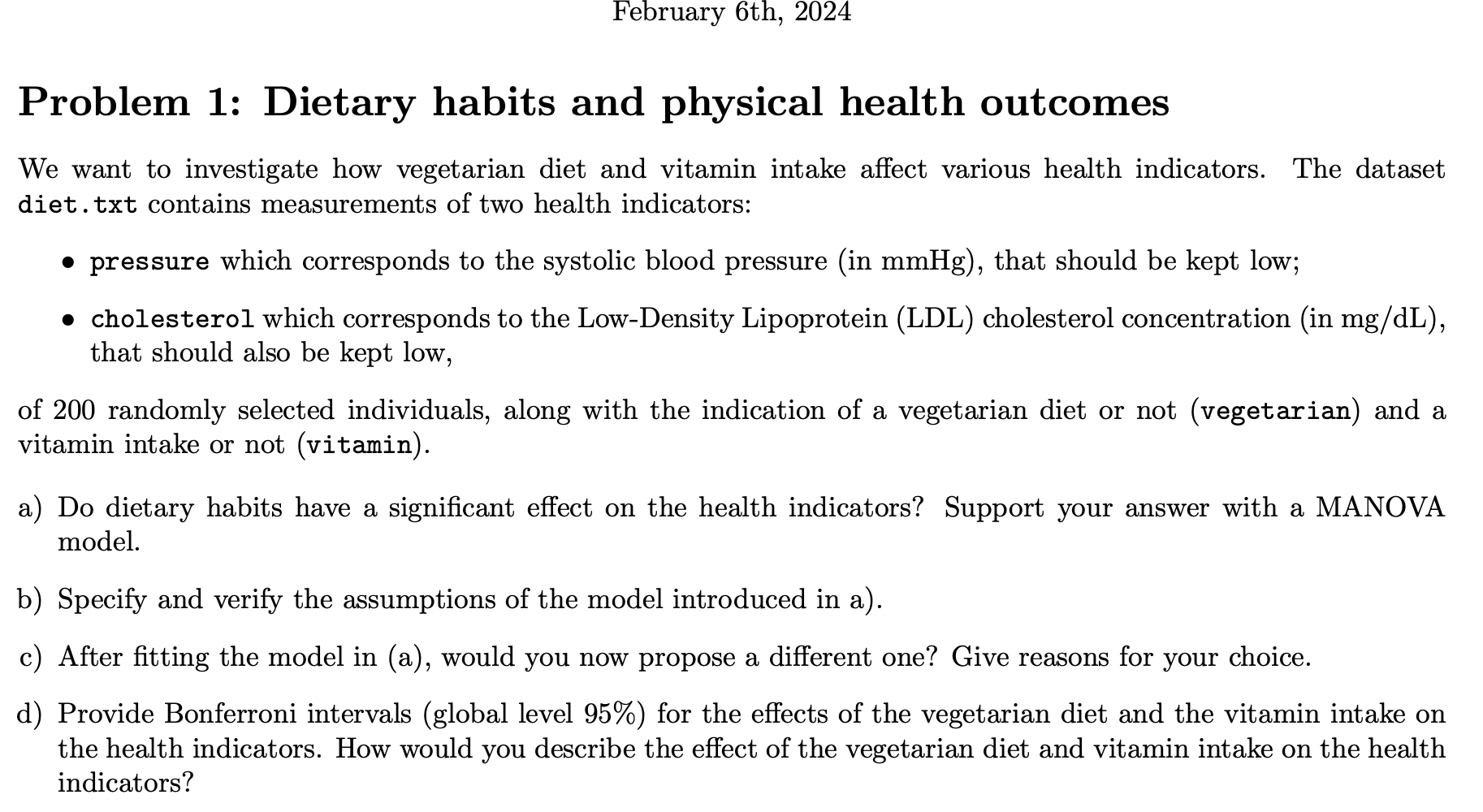
plot(seq(-5,5))

points(cc1.data[i1], rep(0, length(cc1.data[i1])), pch = 16, col = 'blue')

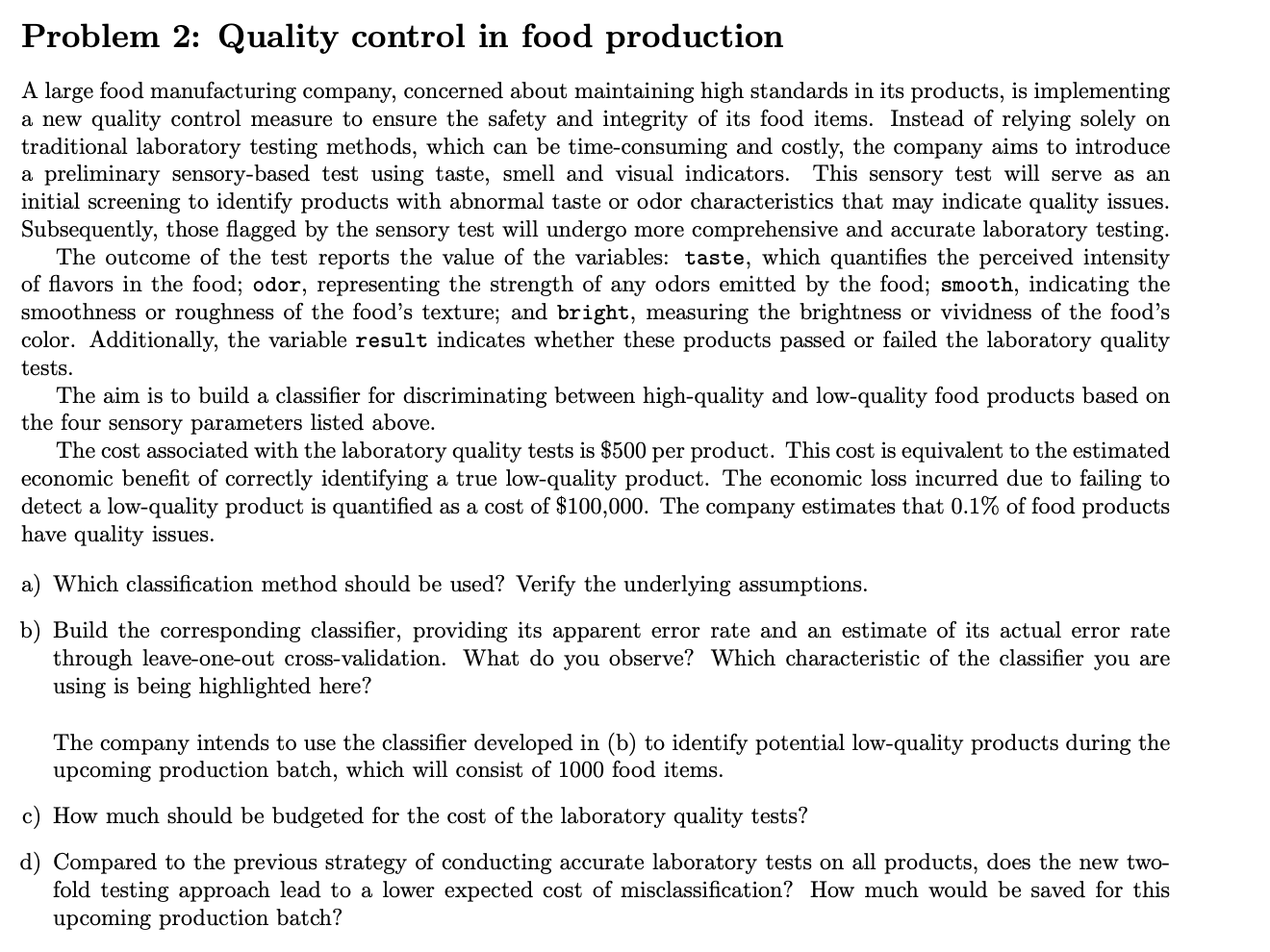
points(cc1.data[i2], rep(0, length(cc1.data[i2])), pch = 16, col = 'red')

# 2024-02-06

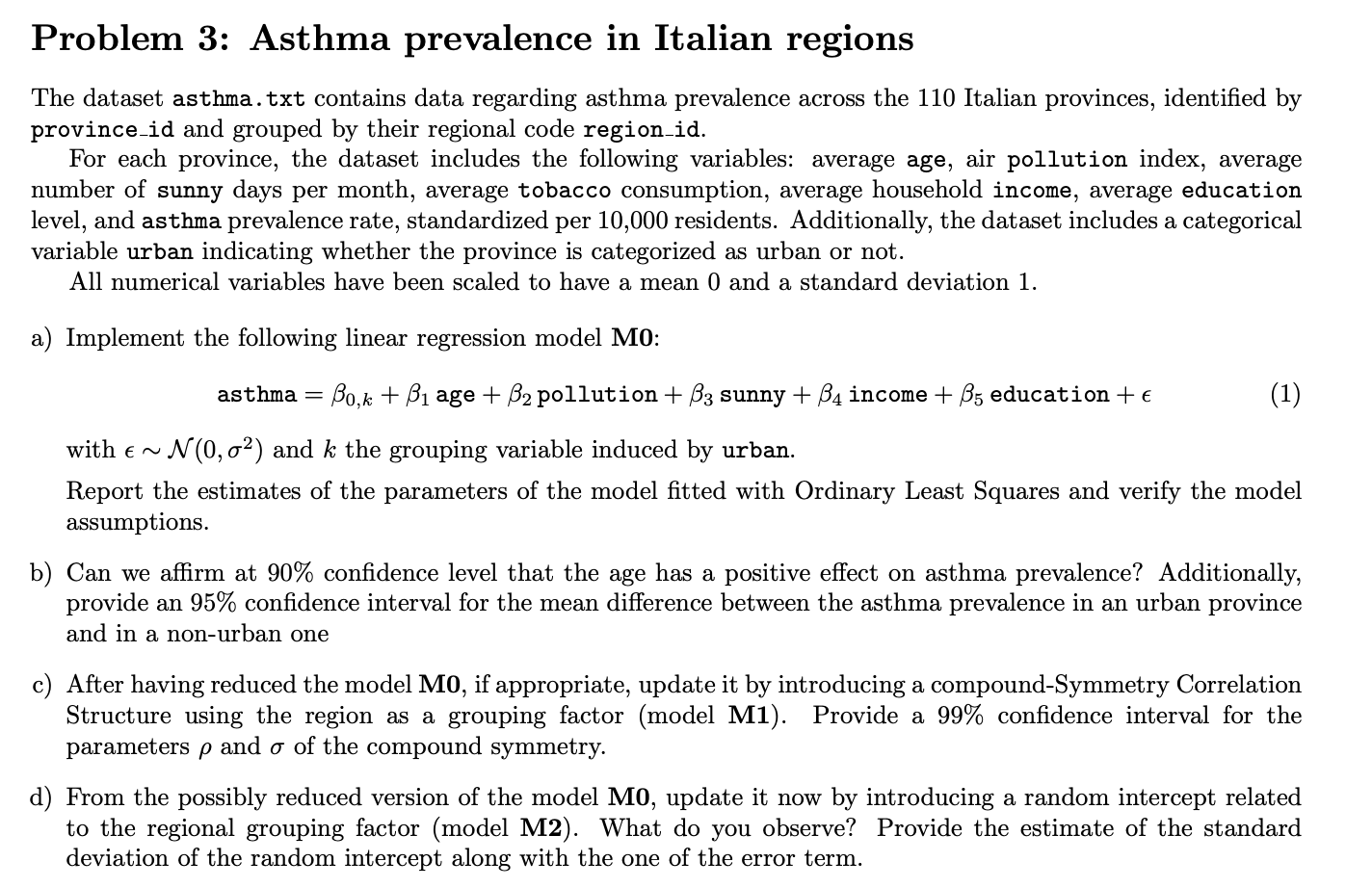
## Two-ways MANOVA + CI



## Classifier (LDA)

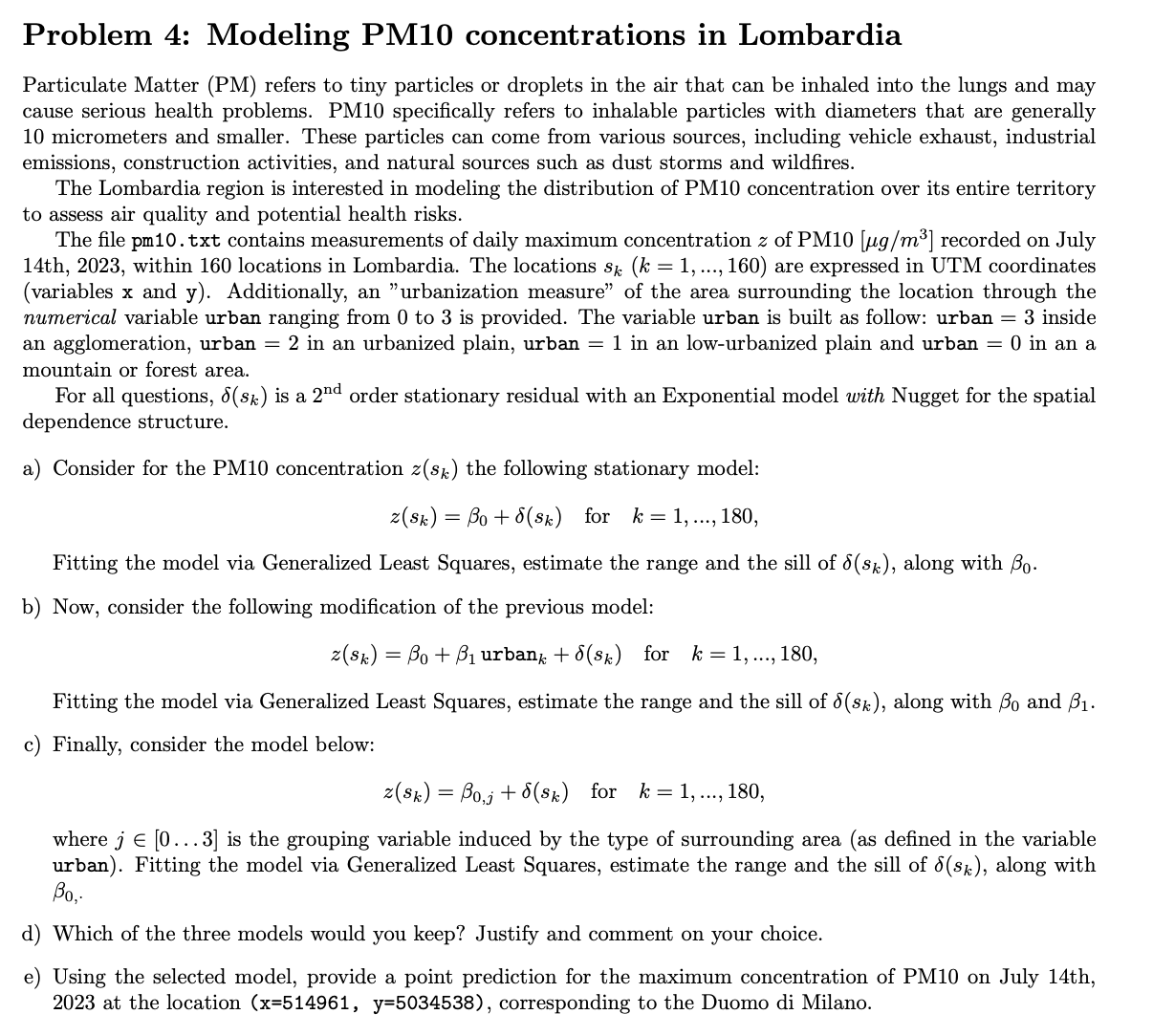


## LMM



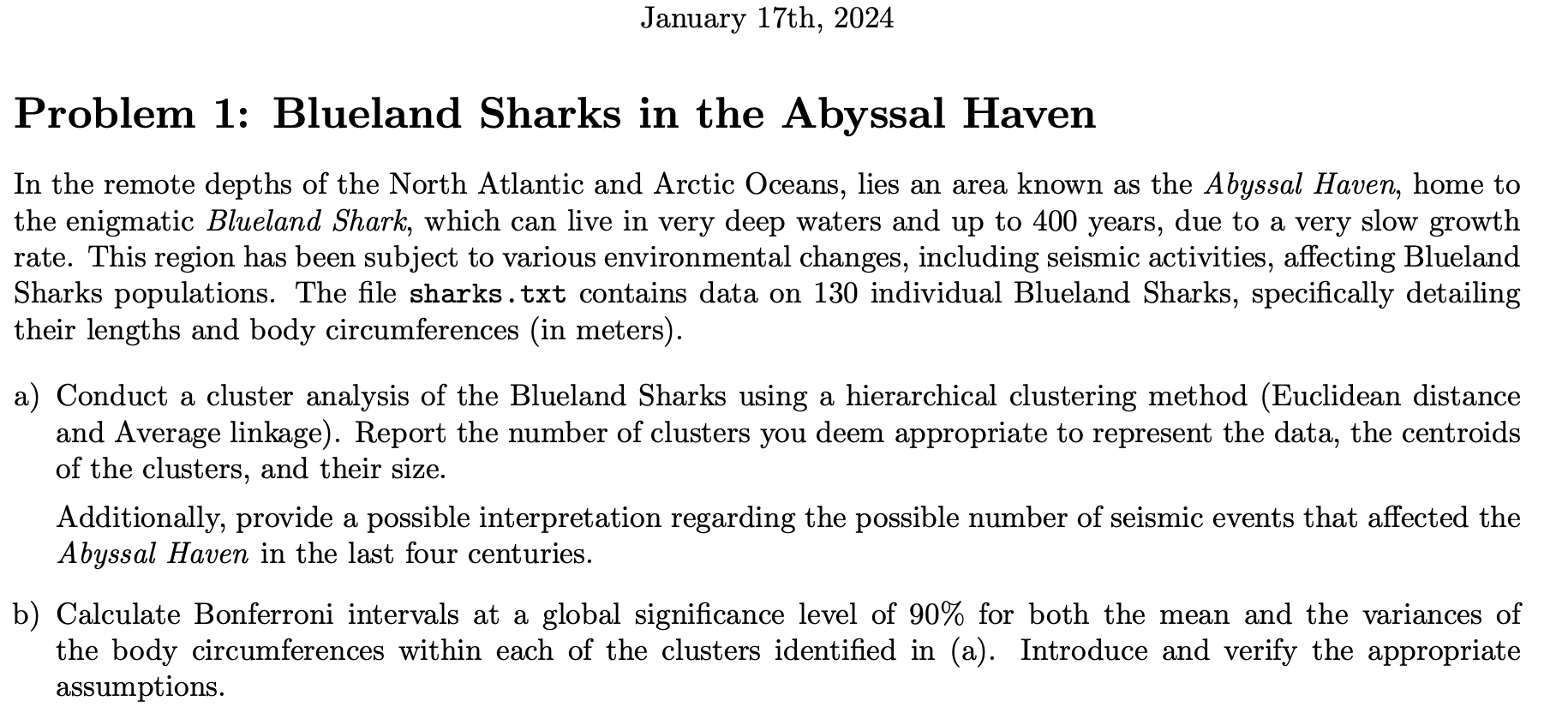
plot(lm\_model$residuals) # ALSO plot this to check homogenity

## GEOSTAT



# 2024-01-17

## Clustering + CI



* 1. Perform Hierarchical clustering, find centroids, and size of each cluster
  2. Compute CI with Bonferroni correction (just for mean and variance, NO difference)

alpha <- 0.1 # глобальный уровень значимости

k <- 2 \* 2 \* 4 # 2 статистики (mean + var) × 2 переменные × 4 кластера

alpha.bf <- alpha / k # уровень для одного интервала

# Group 1 ----------------------------------------------------------------------

cov1 <- cov(sharks[cluster.ec==1,])

ICmean1 <- cbind(inf=m1 - sqrt(diag(cov1)/n1) \* qt(1 - alpha/(2\*k), n1-1),

center= m1,

sup= m1 + sqrt(diag(cov1)/n1) \* qt(1 - alpha/(2\*k), n1-1))

ICmean1

# inf center sup

# circumference 0.916496 1.001071 1.085647

# length 1.338416 1.468214 1.598013

ICvar1 <- cbind(inf=diag(cov1)\*(n1-1) / qchisq(1 - alpha/(2\*k), n1-1),

center=diag(cov1),

sup=diag(cov1)\*(n1-1) / qchisq(alpha/(2\*k), n1-1))

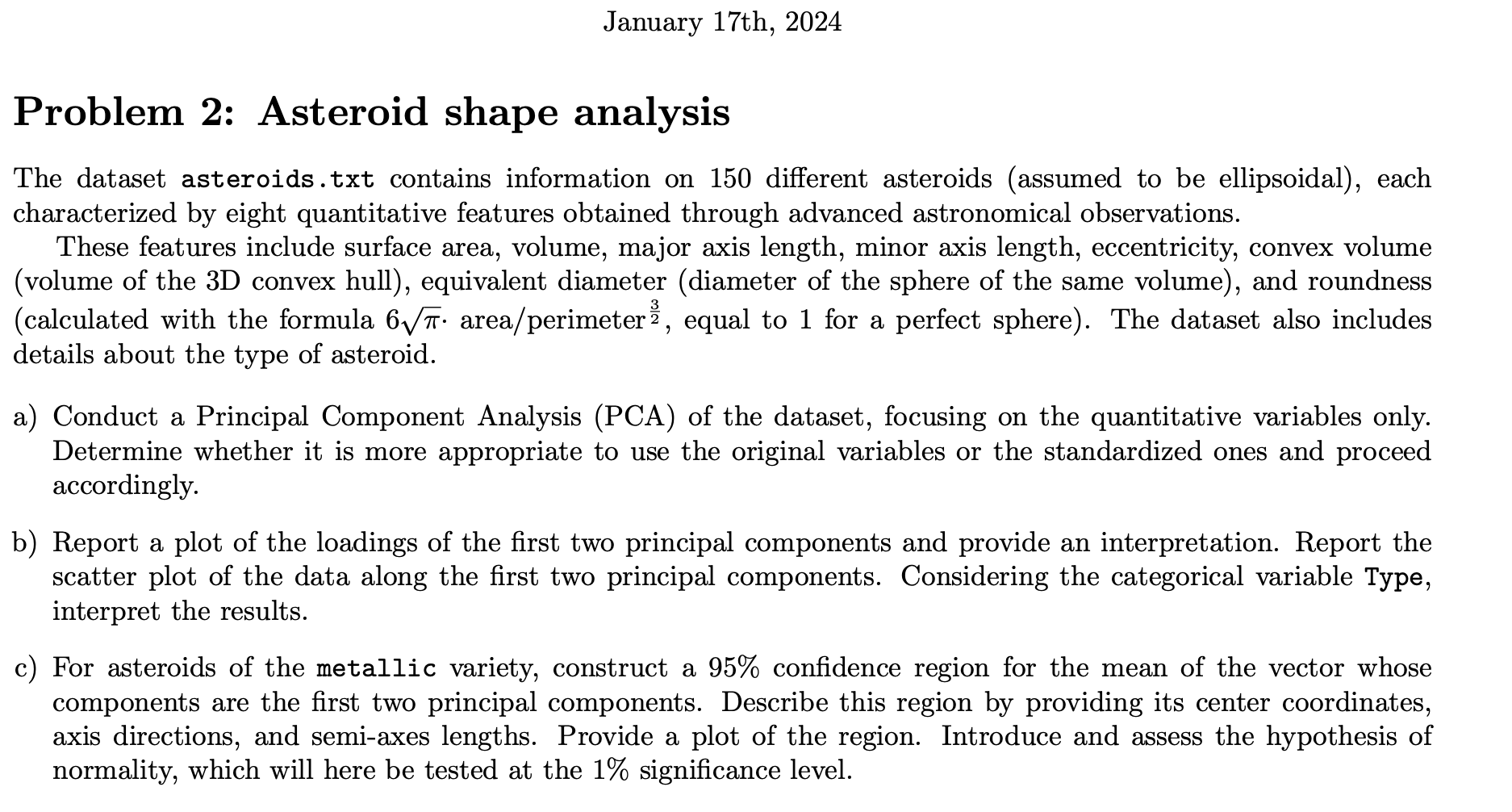
ICvar1

# inf center sup

# circumference 0.01196274 0.02277288 0.0550907

# length 0.02817608 0.05363743 0.1297562

## PCA + CR (Ellipse)



* 1. Perform PCA (need scale data)

boxplot(asteroids\_data, col = 'gold', main = 'Original Variables')

asteroids\_data.sd <- scale(asteroids\_data, center=TRUE, scale=TRUE)

asteroids\_data.sd <- data.frame(asteroids\_data.sd)

boxplot(asteroids\_data.sd, col = 'gold', main = 'Standartised Variables')

asteroids\_data.sd

pc.asteroids\_data.sd <- princomp(asteroids\_data.sd, scores=T)

pc.asteroids\_data.sd

summary(pc.asteroids\_data.sd)

* 1. Explain Loadings, plot 2-D plot of two components

n\_lod\_show <- 2

load.asteroids\_data.sd <- pc.asteroids\_data.sd$loadings

par(mar = c(1, n\_lod\_show, 0, 2), mfrow = c(n\_lod\_show, 1))

for (i in 1:n\_lod\_show) {

barplot(load.asteroids\_data.sd[, i], ylim = c(-1, 1))

}

# first components represents general effect of variabels, as:

# SurfArea, Volume, MajAxis, MinAxis, ConvVol, EqDiam, Round

# while second component represents contrast of:

# MajAxis and Ecc, with MinAxis and Round

cols <- as.factor(asteroids$Type)

plot(pc.asteroids\_data.sd$scores[,1], pc.asteroids\_data.sd$scores[,2],

col = cols, pch = 19, xlab = "PC1", ylab = "PC2", main = "PCA of Asteroids")

text(pc.asteroids\_data.sd$scores[,1], pc.asteroids\_data.sd$scores[,2],

labels = asteroids$Type, pos = 3, cex = 0.7)

* 1. 95% CI Region (ellipse), with axes, length of axes, and check MVN

metalic\_data <- data.frame(

pc1=pc.asteroids\_data.sd$scores[,1][asteroids$Type == "metallic"],

pc2=pc.asteroids\_data.sd$scores[,2][asteroids$Type == "metallic"]

)

mvn(metalic\_data) # 1% significance level => if p-value > 0.01 => normal

metalic.m <- sapply(metalic\_data, mean)

# 0.06248007 -0.08163318

metalic.cov <- cov(metalic\_data)

#. pc1 pc2

# pc1 0.009653561 0.001505507

# pc2 0.001505507 0.007900650

# Directions and axis length ---------------------------------------------------

eig <- eigen(metalic.cov / n\_metalic)

directions <- eig$vectors # направления (ось эллипса)

lengths <- sqrt(eig$values \* cfr.fisher) # длины полуосей

# Результаты:

directions # Each column — direction of axis

# [,1] [,2]

# [1,] -0.8669251 0.4984385

# [2,] -0.4984385 -0.8669251

lengths # длины полуосей (вдоль направлений)

# 0.03702053 0.03027514

n\_metalic <- nrow(metalic\_data)

p <- 2

alpha <- 0.05

cfr.fisher <- ((n\_metalic - 1) \* p / (n\_metalic - p)) \* qf(1 - alpha, p, n\_metalic - p)

cfr.fisher

# Ellipsoidal confidence region with confidence level 95%

plot(metalic\_data, asp=1, pch=1, main='Metallic data')

# plottin ellipse, don't forget to divide cov/n

ellipse(center=metalic.m, shape=metalic.cov/n\_metalic, radius=sqrt(cfr.fisher), lwd=2)

arrows(metalic.m[1], metalic.m[2],

metalic.m[1] + lengths[1]\*directions[1,1],

metalic.m[2] + lengths[1]\*directions[2,1],

col="red", lwd=2)

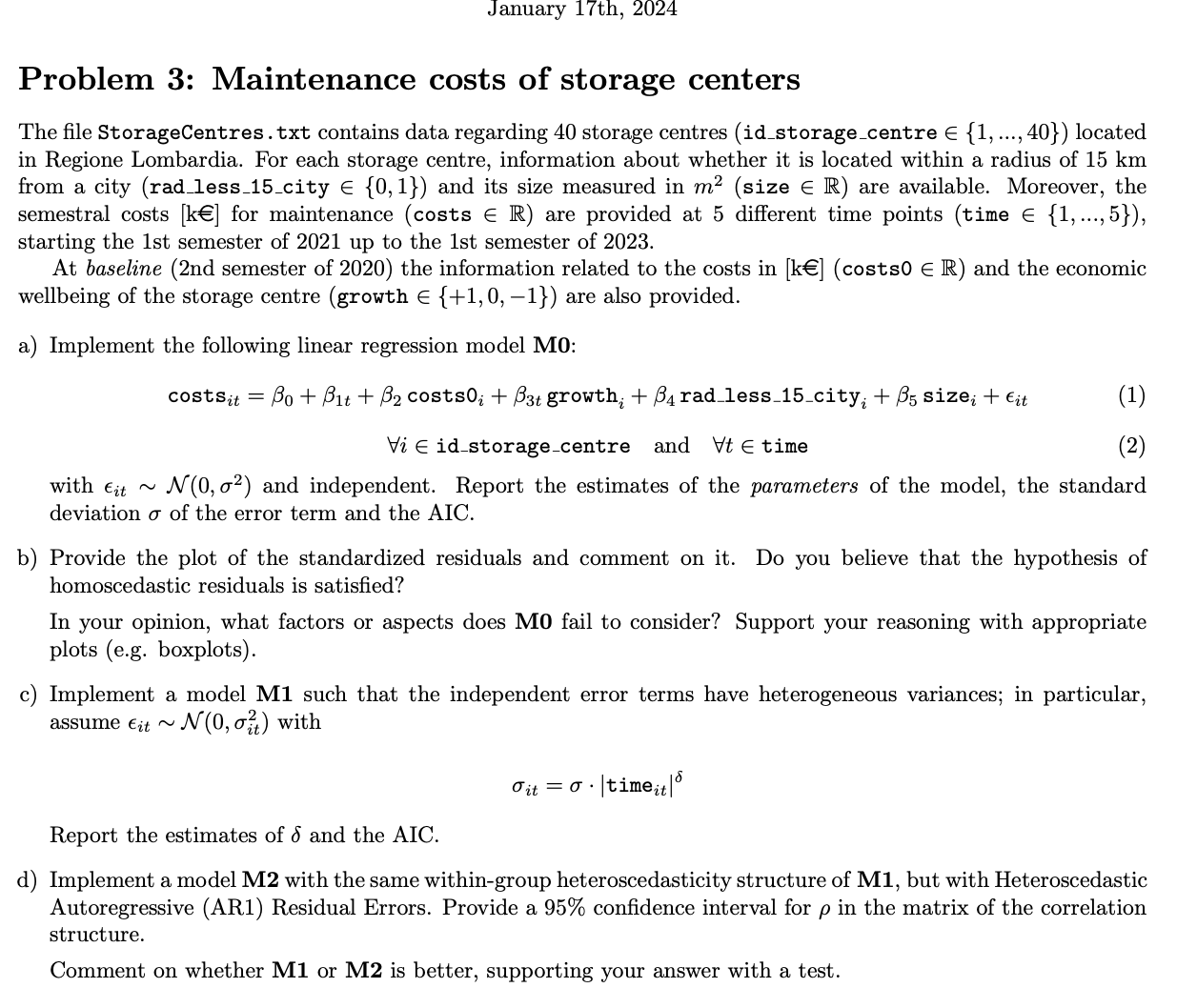
arrows(metalic.m[1], metalic.m[2],

metalic.m[1] + lengths[2]\*directions[1,2],

metalic.m[2] + lengths[2]\*directions[2,2],

col="darkgreen", lwd=2)

## LMM (AR1)



* 1. Model with groups factors and AIC

lm\_model <- lm(costs ~ 1 + as.factor(time) + costs0 + growth:as.factor(time) + rad\_less\_15\_city + size,

data=storage)

summary(lm\_model)

AIC(lm\_model)

* 1. Check hypothesis of OLS (residuals normality and homoscedasticity) (why assumptions failed)

par(mfrow=c(2,2)); plot(lm\_model)

# plot to confirm that residuals are normal

qqnorm(lm\_model$residuals)

qqline(resid(lm\_model), col='red', lwd=2)

plot(lm\_model, which = 1) # plot to confirm that residuals are normal

plot(lm\_model, which = 2) # plot showing homoscedasticity,

# homoscedascisity looks true, there are no specific pattorn of residuals with fitted values

shapiro.test(lm\_model$residuals) # So we reject H0 of residuals normality

boxplot(storage)

plot(lm\_model$residuals) # ALSO plot this to check homogenity

* 1. Heterogeneous variances

gls\_model1 <- gls(

costs ~ 1 + as.factor(time) + costs0 + growth:as.factor(time) + rad\_less\_15\_city + size,

weights = varPower(form = ~time),

data=storage)

summary(gls\_model1)

* 1. Autoregressive (AR1) Residuals Errors, CI for correlation structure

gls\_model2 <- gls(

costs ~ 1 + as.factor(time) + costs0 + growth:as.factor(time) + rad\_less\_15\_city + size,

weights = varPower(form = ~time),

correlation = corAR1(form = ~time|id\_storage\_centre),

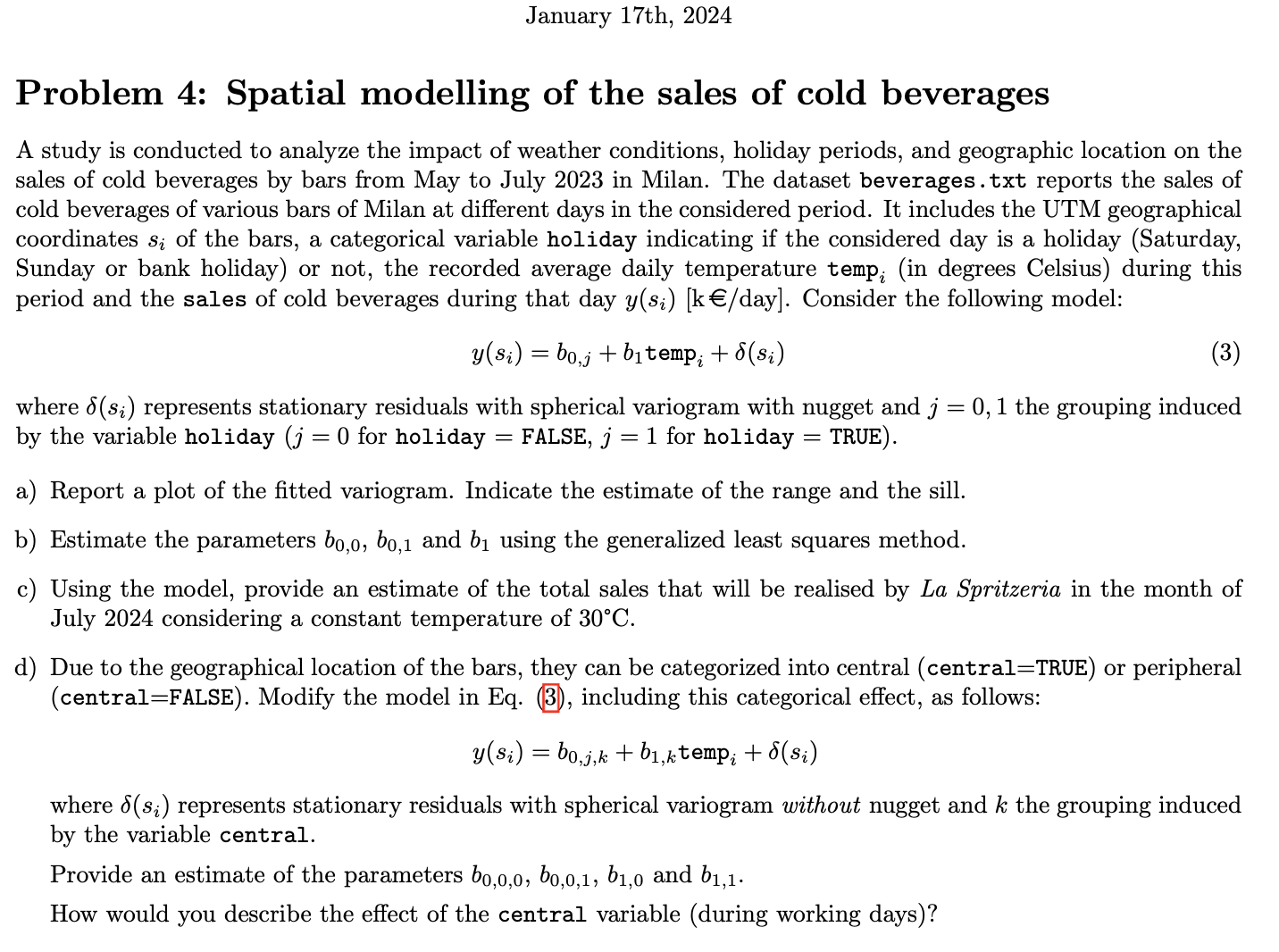
data=storage

)

summary(gls\_model2)

intervals(gls\_model2, which = "var-cov", level = 0.95)

## GEOSTAT

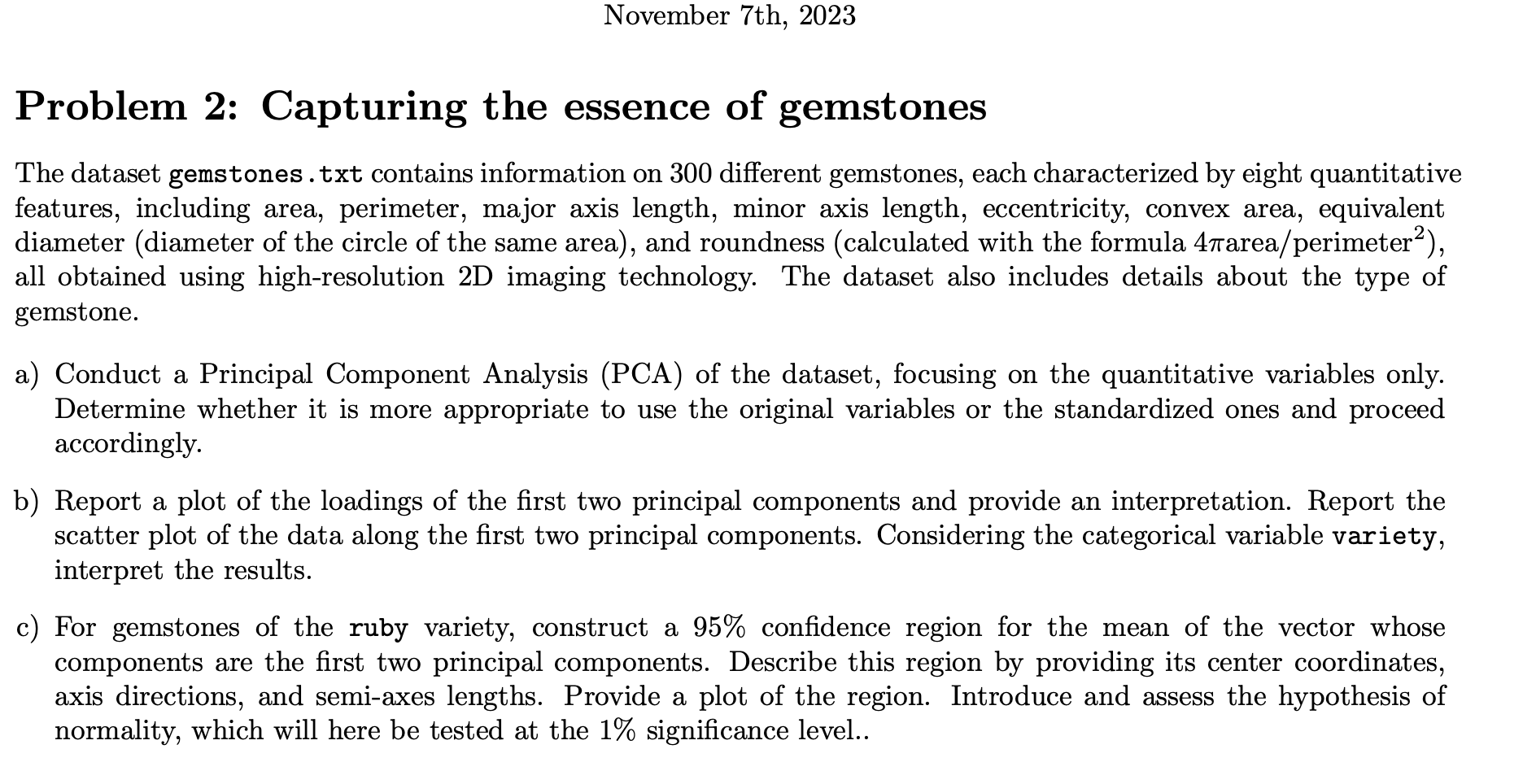


# 2023-11-07

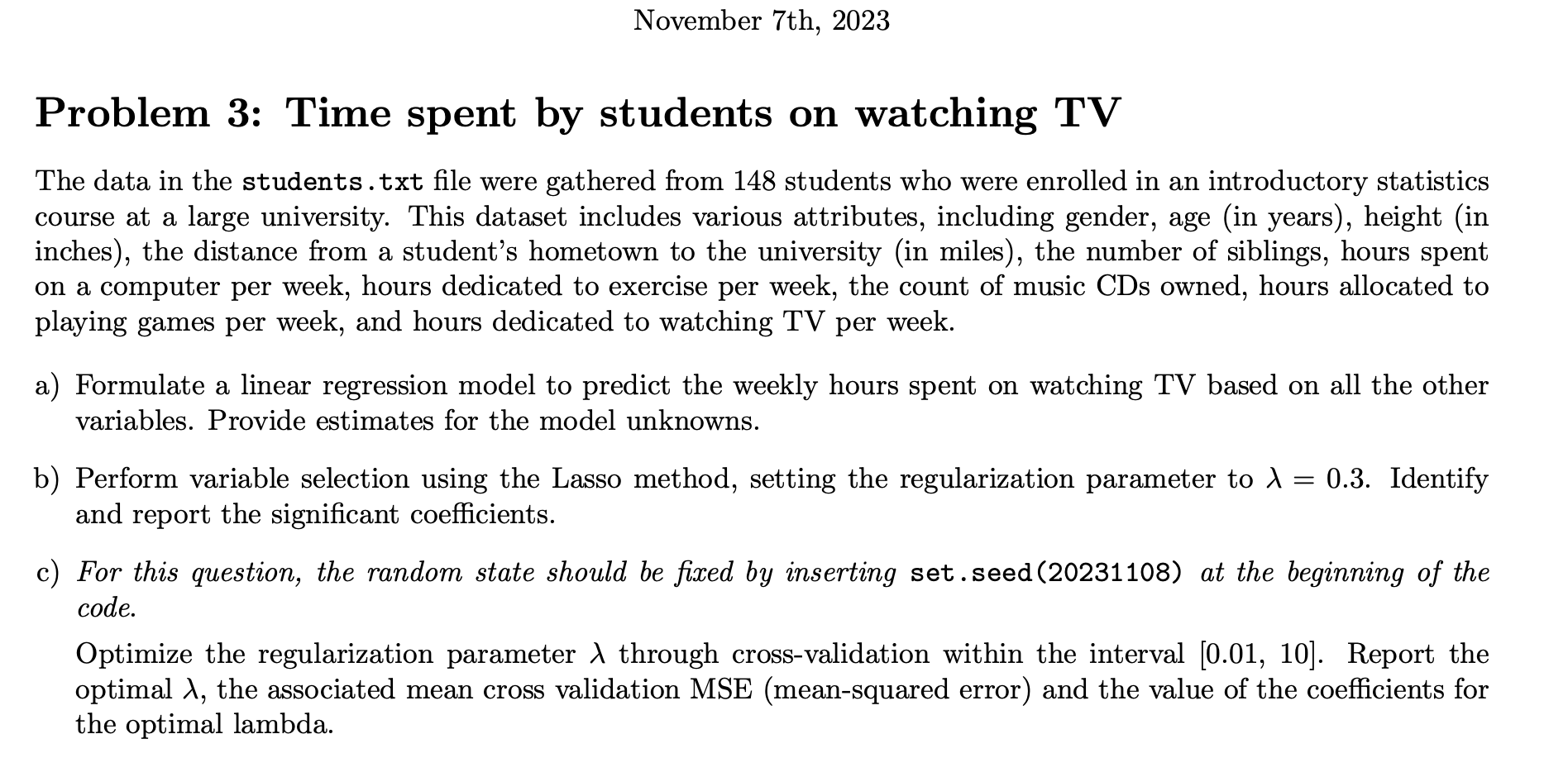
## Clustering + CI



## PCA + CR (Ellipse)



## LASSO



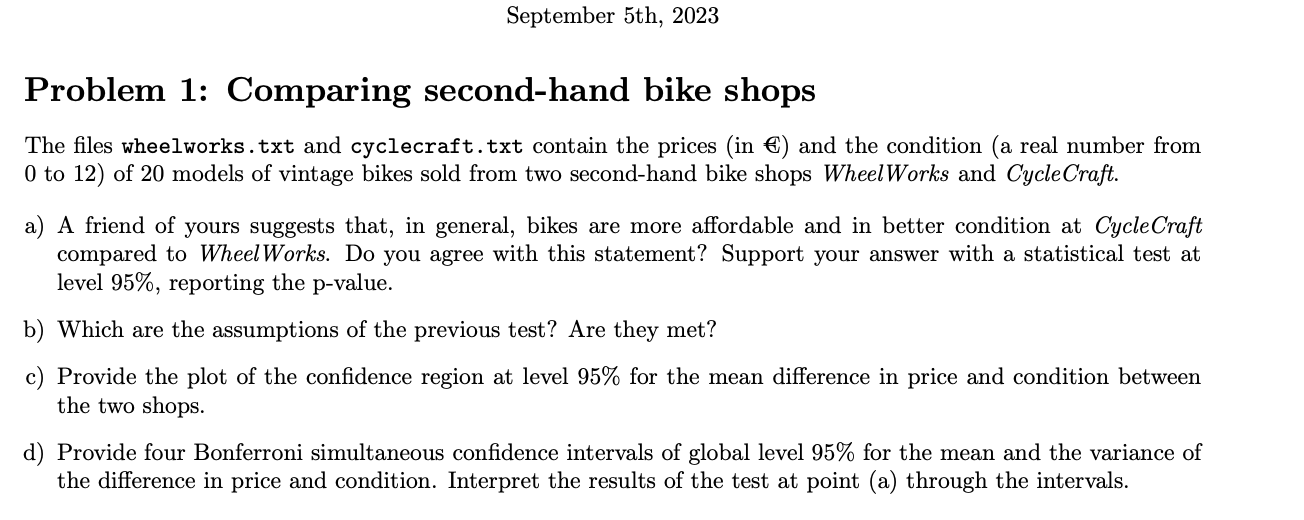
## GEOSTAT



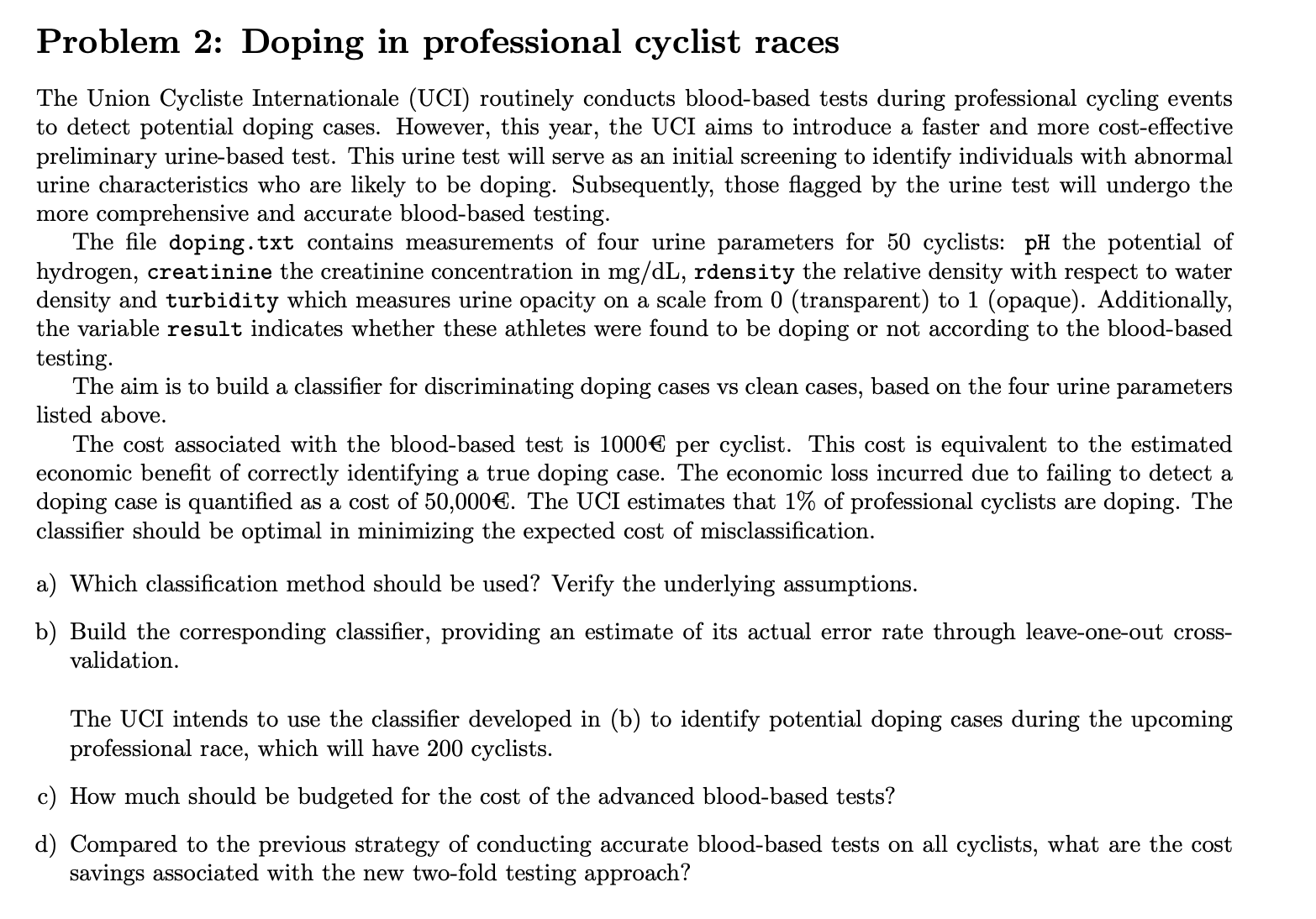
DID NOT SOLVE

# 2023-09-04

## TestMean

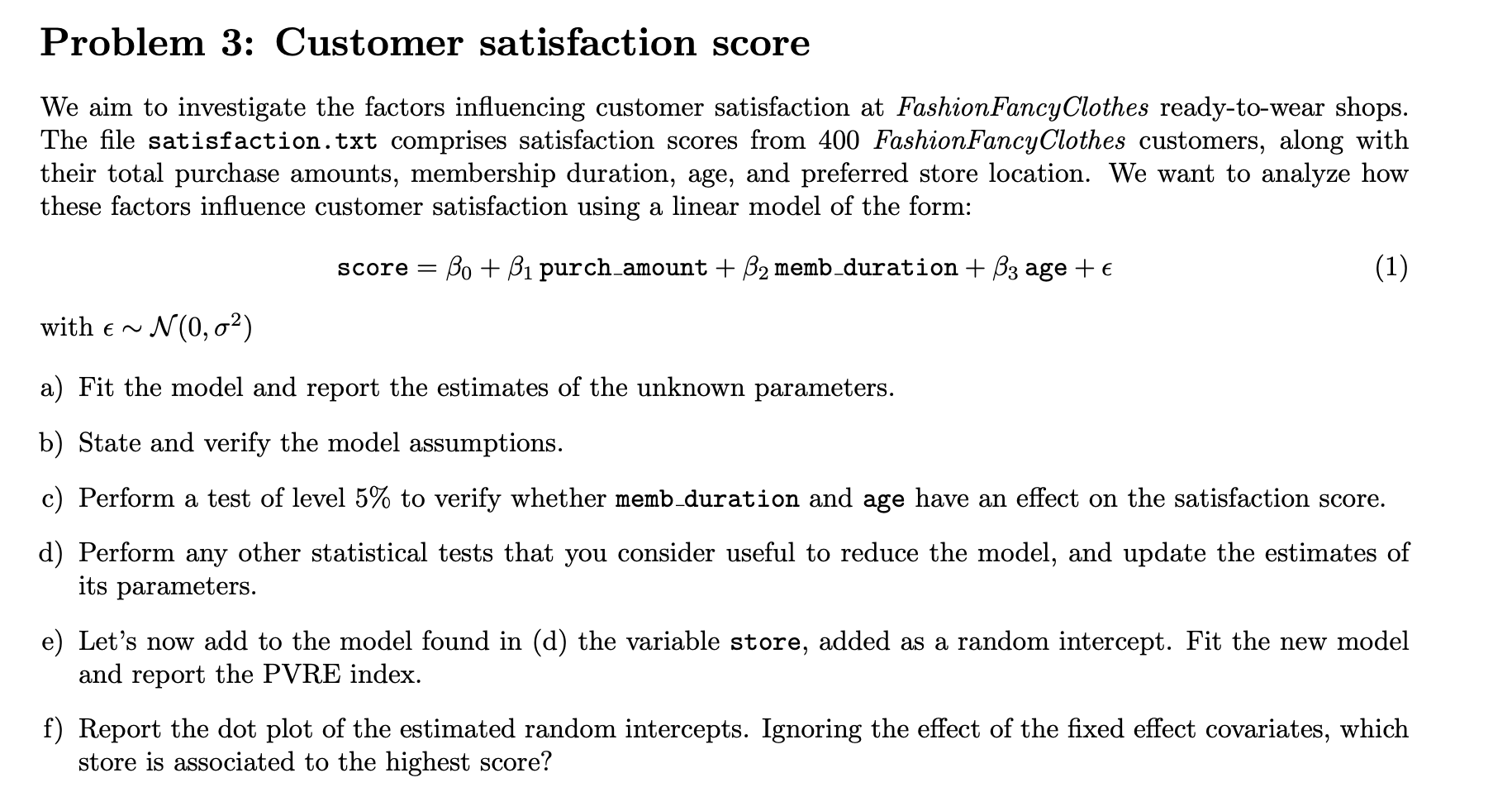


## Classifier



DID NOT SOLVE

## LMM (PVRE, CI, Random intercept plot)



## FunctionalDA (B-splines)

