Some results

2019-04-10

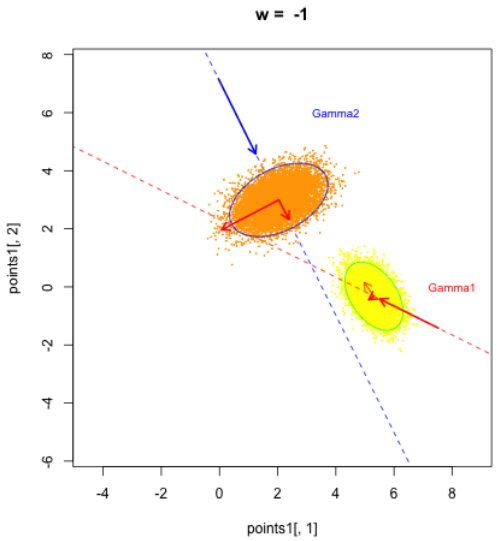
Contents

1 1. Previous plot

The plot of estimated covariates $\beta = (\beta_1, \beta_2)$:

- Add the line to show the direction of Γ_1 and Γ_2.
- Change the x-axis and y-axis in the same scale.

The plot is:



The gif CLICK HERE

2 2. Check the real data

Real data's β, Γ and D.

After calculating the purity, when $\alpha = (0.1494381, 0.9887711)$, then purity meets the max.

Add the combination of covariates (age and baselineCGI) in the the dataset:

```
dat$w = dat$age * 0.1494381 + dat$BaselineCGI * 0.9887711
```

Fit the LMM model:

```
dat_drg = dat[dat$trt == 1,]
dat_pbo = dat[dat$trt == 0,]
fit_drg_est = lmer(y ~ t1 + I(t1^2) + w + w * t1 +
                     w * I(t1^2) + (t1+I(t1^2)|subj),
                   data = dat_drg, REML = FALSE)
fit_pbo_est = lmer(y ~ t1 + I(t1^2) + w + w * t1 +
                     w * I(t1^2) + (t1+I(t1^2)|subj),
                   data = dat_pbo, REML = FALSE)
```

The β:

```
beta1 = as.matrix(fixef(fit_drg_est))[2:3]
beta2 = as.matrix(fixef(fit_pbo_est))[2:3]
beta1; beta2
```

```
## [1] -4.7009166  0.5336457
```

```
## [1] -6.461997  0.762234
```

The Γ:

```
gamma1 = as.matrix(fixef(fit_drg_est))[5:6]
gamma2 = as.matrix(fixef(fit_pbo_est))[5:6]
gamma1; gamma2
```

```
## [1]  0.03929939 -0.01584812
```

```
## [1]  0.21288663 -0.02529197
```

The D matrix:

```
D1 = as.matrix(VarCorr(fit_drg_est)$subj)[2:3, 2:3]
D2 = as.matrix(VarCorr(fit_pbo_est)$subj)[2:3, 2:3]
D1; D2
```

```
##                t1    I(t1^2)
## t1       7.875105 -1.0471405
## I(t1^2) -1.047141  0.1608424
```

```
##                  t1    I(t1^2)
## t1       4.9518831 -0.6652175
## I(t1^2) -0.6652175  0.1172576
```

The eigenvalues and eigenvectors for D:

```
eigen(D1)
```

```
## eigen() decomposition
## $values
## [1] 8.01471753 0.02122937
##
## $vectors
##            [,1]       [,2]
## [1,] -0.9912286 -0.1321584
## [2,]  0.1321584 -0.9912286
```

```
eigen(D2)
```

```
## eigen() decomposition
## $values
## [1] 5.04174309 0.02739757
##
## $vectors
##            [,1]       [,2]
## [1,] -0.9909992 -0.1338678
## [2,]  0.1338678 -0.9909992
```

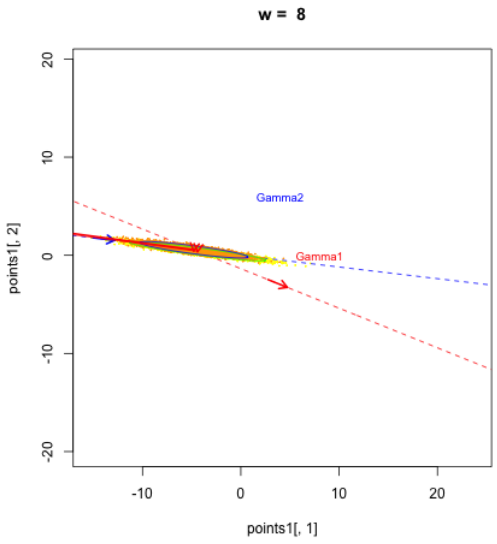
2.0.0.1 Draw the same plot

The range of w:

```
range(dat$w)
```

```
## [1]  6.64497 16.03712
```

Change w, how do the two ellipses move?



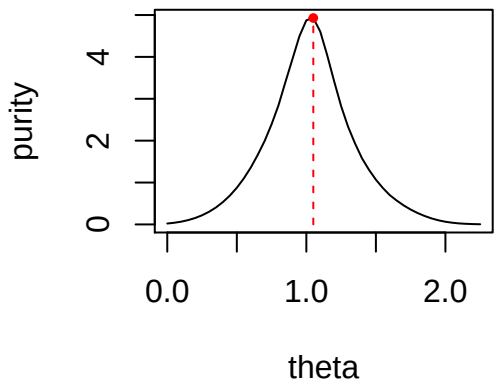
The gif CLICK HERE

3 3. Consider the intercept

Keep the number of covariates the same (still 2 baseline covariates), consider the intercpet. Then the β and Γ changes from 2-dimensions to 3-dimensions.

We can still get the same θ v.s. purity plot:

Purity calculation with consideration of intercep



The max value is close to the true value $\frac{\pi}{3}$

```
data[data$y == max(data$y),]
```

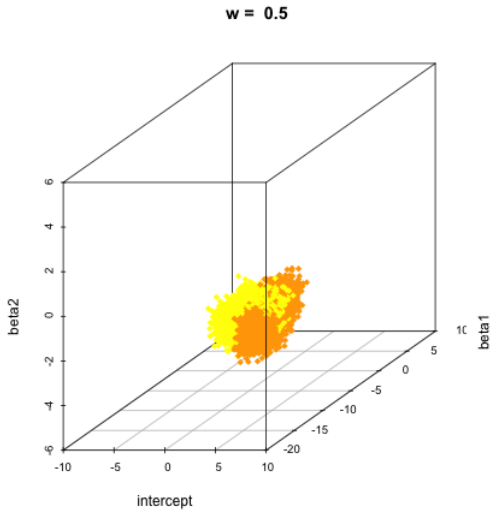
```
##        x        y
## 22 1.05 4.928357
```

```
pi/3
```

```
## [1] 1.047198
```

3.0.0.1 Draw the 3D plot

The dimension increases:



The gif CLICK HERE

4 4. How to change the purity value to cluster classification.

Back to the cvxcluster function. With consideration of covariates, the different is in beta. We may make a new function to calculate the lambda.

```r
lambda = function(data, d,  # data, d = design matrix dim 2
                  beta1, beta2,
                  gamma1, gamma2, D1, D2,
                  pi1, pi2){

  x = as.numeric(data[,1:d])
  miu1 = beta1 + data$W * gamma1 # calculate the new miu
  miu2 = beta2 + data$W * gamma2

  f2x = exp(-d/2 * log(2*pi) -0.5 * log(abs(det(D2))) -
        0.5 * t(x - miu2) %*%  solve(D2) %*% (x - miu2))
  f1x = exp(-d/2 * log(2*pi) -0.5 * log(abs(det(D1))) -
        0.5 * t(x - miu1) %*%  solve(D1) %*% (x - miu1))

  if( pi1 * f1x + pi2 * f2x == 0){
    denom = 10e-10
  }else{
    denom = pi1 * f1x + pi2 * f2x
  }
  return(pi2 * f2x / denom)
}
```

```r
setwd('/Users/yaolanqiu/Desktop/NYU/rotation/Rotation2/Week3/from dr.tarpey')

dat <- read.table("hcaf.dat", header=T)
dat$w = dat$age * 0.1494381 + dat$BaselineCGI * 0.9887711

dat_est = dat
dat_pbo_est = dat_est[dat_est$trt == 1, ]
dat_drg_est = dat_est[dat_est$trt == 0, ]

fit_drg_est = lmer(y ~ t1 + I(t1^2) + w + w * t1 +
                     w * I(t1^2) + (t1+I(t1^2)|subj),
                   data = dat_drg_est, REML = FALSE)

fit_pbo_est = lmer(y ~ t1 + I(t1^2) + w + w * t1 +
                     w * I(t1^2) + (t1+I(t1^2)|subj),
                   data = dat_pbo_est, REML = FALSE)

beta1 = as.matrix(fixef(fit_drg_est))[2:3]
gamma1 = as.matrix(fixef(fit_drg_est))[5:6] # true estimate = -1.9812346 -0.9895642
D1 = as.matrix(VarCorr(fit_drg_est)$subj)[2:3, 2:3]

beta2 = as.matrix(fixef(fit_pbo_est))[2:3]
gamma2 = as.matrix(fixef(fit_pbo_est))[5:6] # 1.994051 1.003545
D2 = as.matrix(VarCorr(fit_pbo_est)$subj)[2:3, 2:3]

# simulation:
xsim_patient = c();xsim_control = c();
m1 = c(); m2 = c();W_1 = c(); W_2 = c()
wrange = unique(dat_est$w) # get the range of w
# for each w value, simulate 100 subject
pns1 = pns2 = 100
for(w in wrange){
```

```
  miu1 = beta1 + w * gamma1 # new miu1
  miu2 = beta2 + w * gamma2 # new miu2
  W_1 = c(W_1, rep(w, pns1)); W_2 = c(W_2, rep(w, pns2))
  xsim_patient = rbind(xsim_patient,
                       data.frame(mvrnorm(pns1, miu1, D1)))
  xsim_control = rbind(xsim_control,
                       data.frame(mvrnorm(pns2, miu2, D2)))
}

# the new dataset
W = c(W_1, W_2)
xsim_patient = as.data.frame(xsim_patient)
xsim_control = as.data.frame(xsim_control)
# group
ns1 = pns1 * length(wrange)
ns2 = pns2 * length(wrange)
xsim_patient$group = rep(1, ns1)
xsim_control$group = rep(2, ns2)
# combine
xsim0 = rbind(xsim_patient, xsim_control)
xsim0 = as.data.frame(xsim0)
xsim0$W = W
xsim0$lambda = NA
```

```
d = 2; k = 4
pi1 = pi2 = 0.5
niter = 100
for (i in 1:dim(xsim0)[1]){
  xsim0$lambda[i] = lambda(xsim0[i,], d,
                           beta1, beta2, gamma1, gamma2,
                           D1, D2, pi1, pi2)
  # calculate the lambda of each point
}
head(xsim0)
```

```
##          t1   I.t1.2. group        W    lambda
## 1 -5.445783 0.8176532     1 8.288789 0.2042158
## 2 -4.271301 0.4873822     1 8.288789 0.4226648
## 3 -4.143873 0.6107268     1 8.288789 0.2523508
## 4 -5.620715 0.7228124     1 8.288789 0.3466583
## 5 -5.135347 0.6671757     1 8.288789 0.3347356
## 6 -3.447847 0.5127307     1 8.288789 0.2744888
```

```
dim(xsim0)
```

```
## [1] 25200     5
```

```
n = ns1 + ns2
p1 = clustering(xsim0, k, d, niter, ns1, ns2)
table(p1$xsim[p1$xsim$group == 1,]$cluster)
```

```
##
##    1    2    3    4
## 4016 3956 3290 1338
```

```r
table(p1$xsim[p1$xsim$group == 2,]$cluster)
```

```
##
##    1    2    3    4
##  710 2517 4674 4699
```

The result without combination of covariates

```
##
##    1    2    3    4
## 3446 3977 3529 1648
```

```
##
##    1    2    3    4
##  651 2292 4656 5001
```

5 5. Higher dimension.

Re-write the function and

Just have a try:

```r
A = c(1,1,1,1)
res = purity_function(A,
                      trt = 'ARM', trtlevel = c(1,3), subj = 'PATID',
                      outcome = 'HAMDT17',
                      varname = c('BASESEV','GENDER','BMI','AGE'),
                      times = 'VISIT',
                      data = hcaf)
```

```
## Warning: Some predictor variables are on very different scales: consider
## rescaling
```

```
## Warning: Some predictor variables are on very different scales: consider
## rescaling
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00288425
## (tol = 0.002, component 1)
```

```r
sum(res$purity)
```

```
## [1] 24.15912
```

There are some questions:

- How could we tune the parameter?
- The covariates have very different scales. Do we need to standardize them?