Simulate data from mixed-effect model determined by hcaf depression data

2019-2-22

Dataset generation

In this document, we would like to simulate new dataset, based on the hcaf depression data.

The data were simulated through the following formula:

$$\mathbf{y_i} = \mathbf{X_i}(\beta + \mathbf{b_i} + \mathbf{\Gamma}(\alpha' \mathbf{x_i})) + \epsilon_\mathbf{i}.$$

- Step 1: Fit LME model based on the *hcaf* data and get the estimated β and covariance matrix value for drug and placebo group separately. Then calculate the eigenvalues and eigenvectors for the covariance matrix.
- Step 2: Generate Γ value for drug and placebo group separately. Here, we seet $\Gamma_{drg} = (0, -0.5, -0.1)'$, $\Gamma_{drg} = (0, 0.25, 0.1)'$
- Step 3: Generate baseline covariates from a normal distribution: $X_{drg} \sim N(0, 1)$, $X_{pbo} \sim N(0, 1)$. Since it is a RCT, the baseline covariates should from a same distribution.
- Step 4: Generate the random effect in the above forumla. $b_i = eigen(D)sqrt(diag(eigen(D)))b_{random}$, where b_{random} is a vector, each element comes from $N(0, 1)$
- Step 5: Calculate the outcome y_i based on the above formula. The simulated dataset is generated.

Following the procedures, we can get:

- 1. First try a big dataset with small covariance matrix (estimated value over 100), small variances (estimated value over 100), to make sure that we can get correct estimated parameters by fitting LME models.

Here, the dataset's size is:

```
ndrg
```

```
## [1] 1000
```

```
npbo
```

```
## [1] 1000
```

The variance is:

```
sigma.drg
```

```
## [1] 0.03427945
```

```
sigma.pbo
```

```
## [1] 0.04030258
```

The estimated results:

```
fixef(fitdrg.sim)
```

```
##  (Intercept)           tt      I(tt^2)            x         tt:x
## 23.690791395 -4.292883396  0.368932568 -0.001528058 -0.499165360
##    I(tt^2):x
```

```
## -0.100067476
```

which is close to our settings

```
beta_drg
```

```
##                   [,1]
## (Intercept) 23.6891978
## t1          -4.2926488
## I(t1^2)      0.3689679
```

```
Gamma_drg
```

```
##      [,1]
## [1,]  0.0
## [2,] -0.5
## [3,] -0.1
```

The same with placebo group fitting.

```
fixef(fitpbo.sim)
```

```
## (Intercept)          tt     I(tt^2)           x        tt:x   I(tt^2):x
## 23.17268379 -4.11114889  0.47729238 -0.04976957  0.18639997  0.11155158
```

```
beta_pbo
```

```
##                   [,1]
## (Intercept) 23.0682623
## t1          -4.2604765
## I(t1^2)      0.5013979
```

```
Gamma_pbo
```

```
##      [,1]
## [1,] 0.00
## [2,] 0.25
## [3,] 0.10
```

However, such settings with the small variances will bring large D^{-1} values when fitting the multivariate normal distribution, which makes it is very easy to get a $f(.)$ value close to 0.

```
D_drg_est = as.matrix(VarCorr(fitdrg.sim)$subj)[2:3, 2:3]
solve(D_drg_est)
```

```
##                tt   I(tt^2)
## tt      10575.82  66801.28
## I(tt^2) 66801.28 493998.83
```

Therefore, I think we can try the sigma values estimated from the LME model without over 100. The results are:

```
ndrg
```

```
## [1] 100
```

```
npbo
```

```
## [1] 100
```

```
sigma.drg
```

```
## [1] 3.427945
```

```
sigma.pbo
```

```
## [1] 4.030258
```

```
fixef(fitdrg.sim)
```

```
## (Intercept)          tt     I(tt^2)           x        tt:x   I(tt^2):x
## 23.74286468 -3.46353349  0.29378088  0.12183549 -0.74798262 -0.07910257
```

```
beta_drg
```

```
##                   [,1]
## (Intercept) 23.6891978
## t1          -4.2926488
## I(t1^2)      0.3689679
```

```
Gamma_drg
```

```
##      [,1]
## [1,]  0.0
## [2,] -0.5
## [3,] -0.1
```

```
fixef(fitpbo.sim)
```

```
## (Intercept)          tt     I(tt^2)           x        tt:x   I(tt^2):x
##  23.5062750  -4.5333380   0.5735903   0.1056350  -0.1131354   0.1228694
```

```
beta_pbo
```

```
##                   [,1]
## (Intercept) 23.0682623
## t1          -4.2604765
## I(t1^2)      0.5013979
```

```
Gamma_pbo
```

```
##      [,1]
## [1,] 0.00
## [2,] 0.25
## [3,] 0.10
```

The values are also close.

Purity calculation

The purity for one subject can be calculated as:

$$P_{w_i} = \int_{z_i} \frac{[f_1(z_i|w_i) - f_2(z_i|w_i)]^2}{[f_1(z_i|w_i) + f_2(z_i|w_i)]^2}(f_1(z_i|w_i) + f_2(z_i|w_i))dz_i = \int_{z_i} \frac{[f_1(z_i|w_i) - f_2(z_i|w_i)]^2}{f_1(z_i|w_i) + f_2(z_i|w_i)}dz_i$$

When the the two distributions are totally separated, the $P_{w_i} = 1$.

However, I found the P_{w_i} can excess 1. Maybe the two separated distributions is not the scenario that makes the formula get the max value?

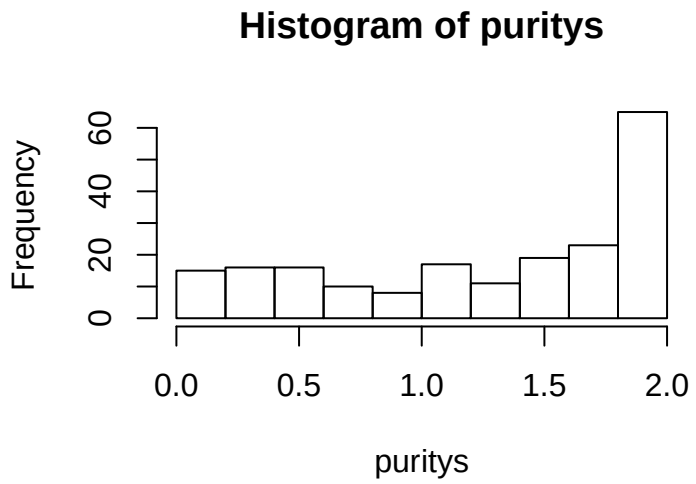
The purity results:

The mean value of P_{w_i} in the dataset is

```r
mean(puritys)
```

```
## [1] 1.281726
```

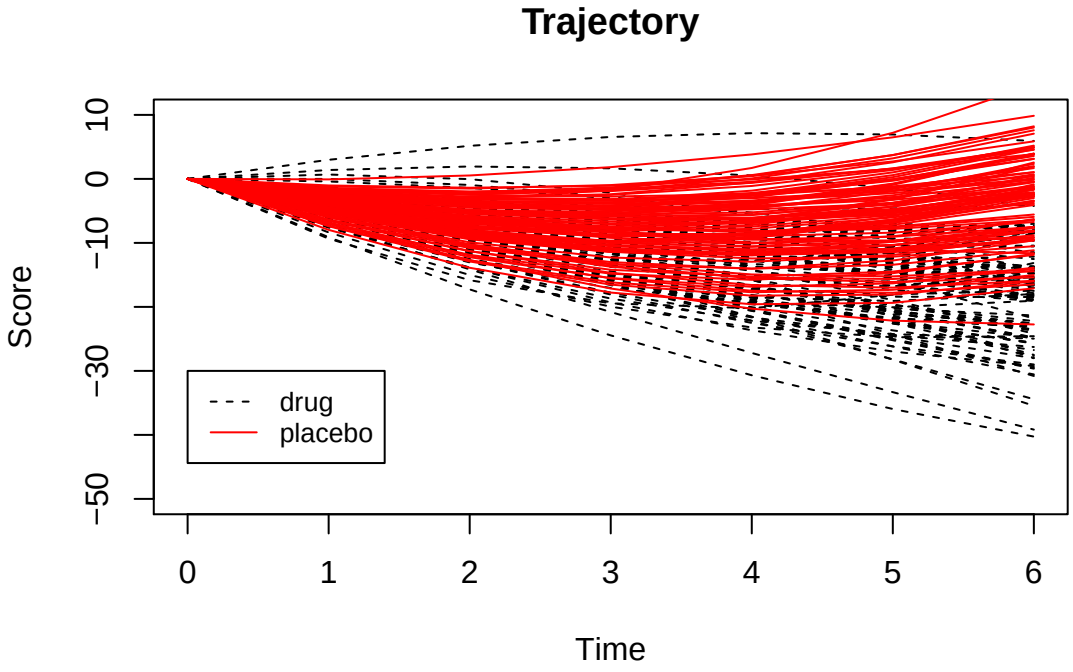
The purity histogram



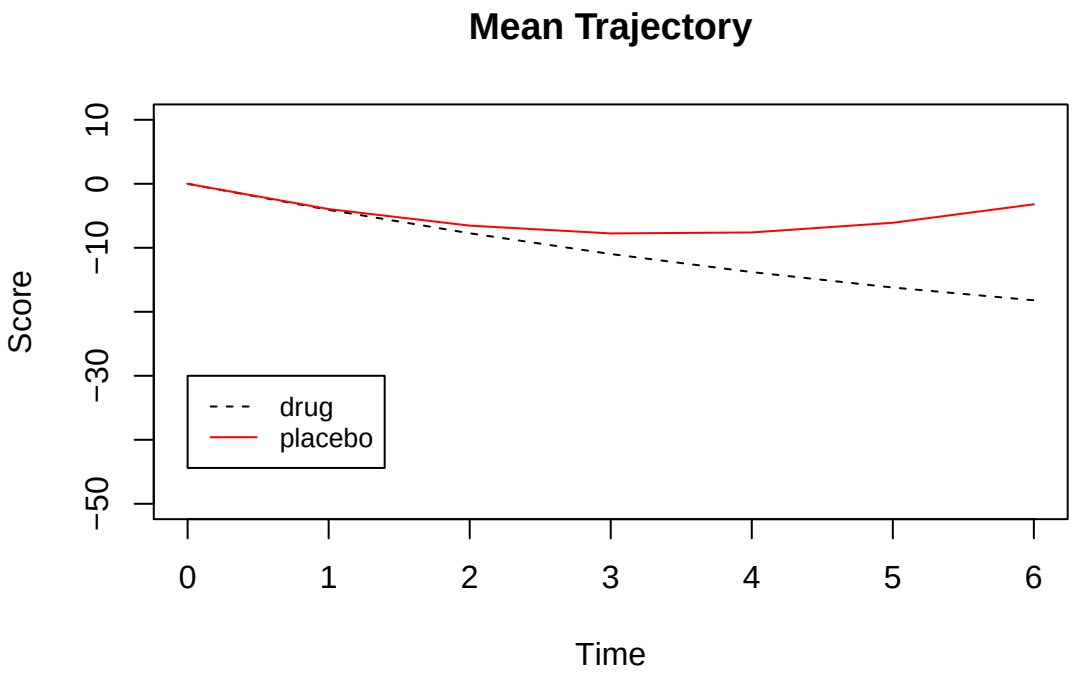
The outcome trajectories

We can fit the LME model, and then get the $z = \beta + b_i + \Gamma(\alpha' x)$ value to fit the polynomial model for the outcome.

We can draw the trajectories of the estimated outcomes.



The mean trajectories of the placebo and drug groups are:

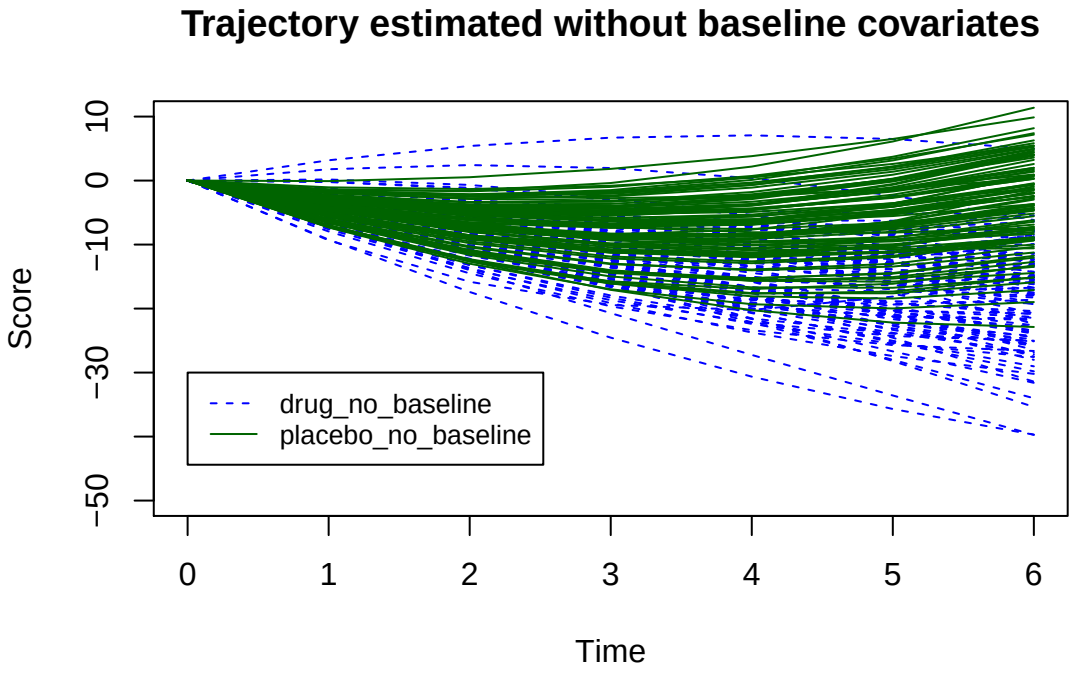


How about trajectories estimated without baseline covariates

Does the baseline covariates help separate the two groups? We may fit LME model without covariates:

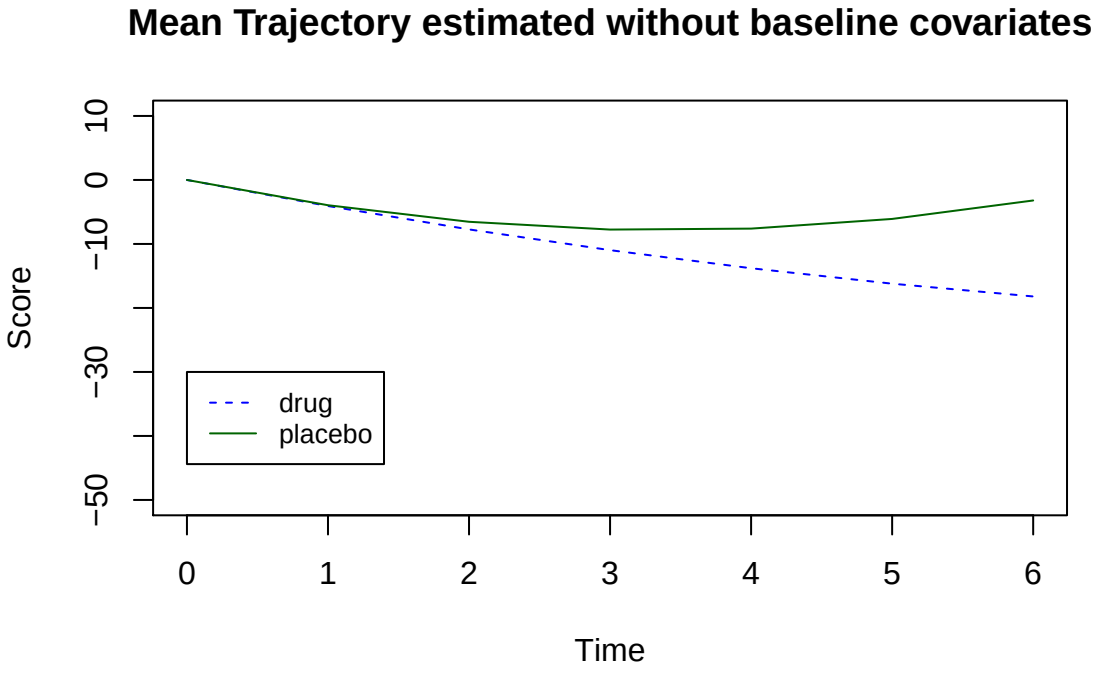
```
# what if we fit the model without covariates x
fitdrg.sim2 = lmer(y ~ tt + I(tt^2) + (tt+I(tt^2)|subj),
                   data = drgsim, REML = FALSE)
fitpbo.sim2 = lmer(y ~ tt + I(tt^2) + (tt+I(tt^2)|subj),
                   data = pbosim, REML = FALSE)
```

The trajectories:

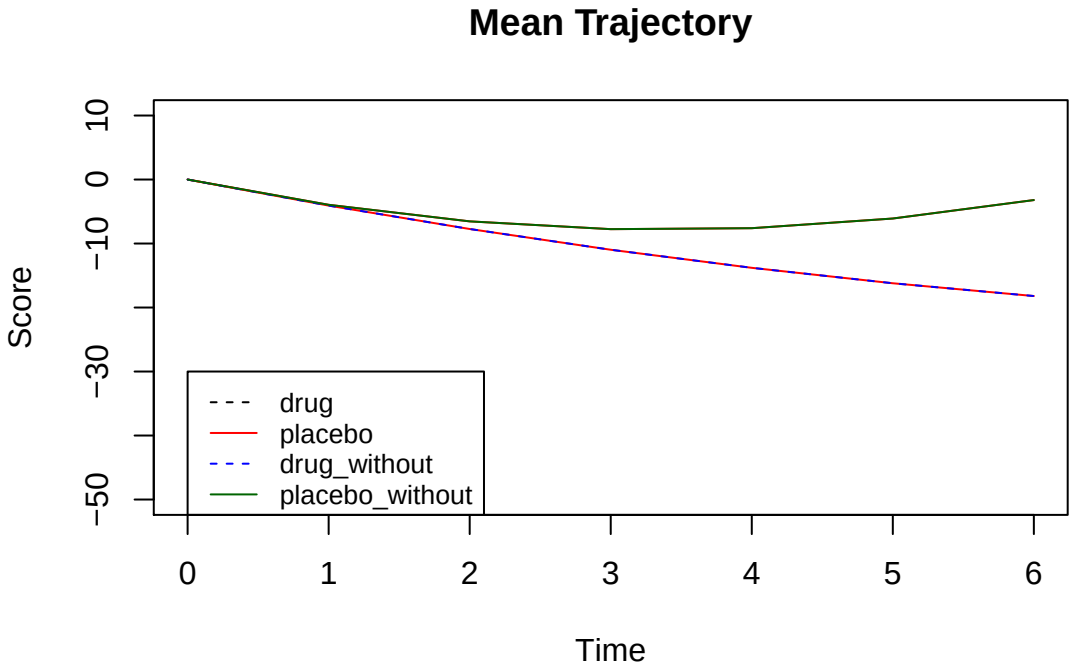


The mean trajectories of the placebo and drug groups are:

xlab = 'Time',ylab = 'Score', main = 'Mean Trajectory'



Draw them together:



The mean tr estimated by those two methods (with covariates and without covariates) are overlapped. We can check their estimated values:

```
apply(mean_beta_drg,2,mean) # Estimated with covariates
```

```
##        tt   I(tt^2)
## -4.279138  0.207527
```

```
apply(coef(fitdrg.sim2)$subj,2,mean) # Estimated without covariates
```

```
## (Intercept)          tt     I(tt^2)
##   23.875715   -4.279138    0.207527
```

Their mean values are the same. But not every value is the same (but they are very close)

```
head(mean_beta_drg)
```

```
##             tt     I(tt^2)
## drg1 -2.381317 -0.03431972
## drg2 -4.078946  0.22167008
## drg3 -7.213512  0.39656614
## drg4 -1.089374 -0.24442088
## drg5 -4.804425  0.41619462
## drg6 -5.151143 -0.09785183
```

```
head(coef(fitdrg.sim2)$subj)
```

```
##      (Intercept)        tt     I(tt^2)
## drg1    27.11683 -2.244571 -0.07103022
## drg2    23.57443 -3.939588  0.18425592
## drg3    23.10600 -7.605017  0.50167504
## drg4    25.58434 -1.208841 -0.21234796
## drg5    21.62719 -4.997856  0.46812396
## drg6    24.68772 -5.264517 -0.06741444
```

Draw all trajectories together