Detecting interaction with unknown environmental covariate

Ziang Zhang

15/10/2020

1 Summary of current idea:

1.1 Latent Model for binary data

For binary response variable, it is often assumed that the reponse variable y_i conditioning on the regressors G_1, G_2 come from a latent model such that:

$$\begin{aligned} Y_i^* &= \beta_0 + \beta_1 G_1 + \beta_2 G_2 + \epsilon_i \\ Y_i &= I\{Y_i^* > 0\} \end{aligned} \tag{1}$$

The unobserved latent variable Y_i^* determines whether the observed response variable Y_i is 0 or 1. The error term ϵ_i in Y_i^* needs to have a completely known distribution, which can be $\mathrm{N}(0,1)$ for the model to become a probit model, or a logistic distribution with mean 0 and variance 3.28 for the model to become a logistic regression model.

1.2 Potential Method 1: By checking linearity:

1.2.1 When the true model does not contain interaction with environmental factor

First, consider that the true underlying model for the response variable Y_i is a probit model without interaction effect, i.e:

$$\begin{aligned} Y_i^* &= \beta_0 + \beta_1 G_1 + \beta_2 G_2 + \epsilon_i \\ Y_i &= I\{Y_i^* > 0\} \\ \epsilon_i &\sim \mathrm{N}(0,1) \end{aligned} \tag{2}$$

Therefore, it can be shown that:

$$\begin{aligned} \mathrm{P}(Y_i = 1|G_1, G_2) &= \mathrm{P}(\epsilon_i > -(\beta_0 + \beta_1 G_1 + \beta_2 G_2)) \\ &= 1 - \Phi(-(\beta_0 + \beta_1 G_1 + \beta_2 G_2)) \\ &= \Phi(\beta_0 + \beta_1 G_1 + \beta_2 G_2) \end{aligned} \tag{3}$$

Where $\Phi(.)$ denote the CDF function of standard normal distribution. Therefore, $\Phi^{-1}\Big(\mathrm{P}(Y_i = 1|G_1, G_2)\Big)$ shoud be a linear function of both G_1 and G_2.

1.2.2 When the true model does contain gene-environment interaction

Assume for simplicity that E_i the environmental variable has a normal distribution with mean μ_E and variance σ_E^2, and suppose that the true underlying model is:

$$\begin{aligned} Y_i^* &= \beta_0 + \beta_1 G_1 + \beta_2 G_2 + \beta_3 G_1 \times E + \epsilon_i \\ Y_i &= I\{Y_i^* > 0\} \\ \epsilon_i &\sim \mathrm{N}(0,1) \end{aligned} \tag{4}$$

Furthermore, we can compute that:

$$\begin{aligned} \mathrm{E}(Y_i^*|G_1,G_2) &= \beta_0 + (\beta_1 + \beta_3\mu_E)G_1 + \beta_2 G_2 \\ \mathrm{Var}(Y_i^*|G_1,G_2) &= (\beta_3 G_1)^2\sigma_E^2 + 1 \\ Y_i^*|G_1,G_2 &\sim \mathrm{N}\Big(\beta_0 + (\beta_1 + \beta_3\mu_E)G_1 + \beta_2 G_2, \big(\beta_3 G_1\big)^2\sigma_E^2 + 1\Big) \end{aligned} \tag{5}$$

That implies that the probability we get a case for different levels of G_1 and G_2 will be:

$$\begin{aligned} \mathrm{P}(Y=1|G_1,G_2) &= \mathrm{P}(Y^*>0|G_1,G_2) \\ &= \mathrm{P}\Big(\frac{Y^* - \mathrm{E}(Y^*|G_1,G_2)}{\sqrt{\mathrm{Var}(Y^*|G_1,G_2)}} > \frac{-\mathrm{E}(Y^*|G_1,G_2)}{\sqrt{\mathrm{Var}(Y^*|G_1,G_2)}}\Big) \\ &= \Phi\Big(\frac{\mathrm{E}(Y^*|G_1,G_2)}{\sqrt{\mathrm{Var}(Y^*|G_1,G_2)}}\Big) \end{aligned} \tag{6}$$

Therefore, applying the inverse CDF on both sides, we get

$$\Phi^{-1}\Big(\mathrm{P}(Y=1|G_1,G_2)\Big) = \frac{\beta_0 + (\beta_1 + \beta_3\mu_E)G_1 + \beta_2 G_2}{\sqrt{(\beta_3^2 G_1^2\sigma_E^2 + 1)}}$$

.

This is not a linear function of G_1, but is a linear function of G_2.

1. If the true underlying model also contains another regressor Z but Z is uncorrelated with G_2 for example. Then eventhough ignoring that regressor breaks the structural assumption of probit model, so that the fitted model without Z is no longer a probit model (since now ϵ does not follow standard normal), but $\Phi^{-1}(\mathrm{P}(Y_i=1|G_1,G_2))$ will still be a linear function of G_2. So detecting based on the linearity of $\Phi^{-1}\mathrm{P}$ will not be affected by omitted exogenous regressors.
2. Since $P(Y_i=1|G_1,G_2)$ is actually unknown in practice, we can estimate it using the sample proportion $\hat{P}(Y=1|G_1=g_1,G_2=g_2) = \frac{\sum_{i=1}^n \mathrm{I}\{y_i=1,G_{1i}=g_1,G_{2i}=g_2\}}{\sum_{i=1}^n \mathrm{I}\{G_{1i}=g_1,G_{2i}=g_2\}}$. We shouldn't use the fitted model to estimate them since our fitted model may be wrong.
3. The reason we used probit model instead of logistic model here is that assuming E follows normal distribution, $Y^*|G_1,G_2$ will still be normal if we omit the interaction term, since linear combination of normal is normal. But assuming E follows logistic distribution does not imply that $Y^*|G_1,G_2$ will be logistically distributed as logistic distribution is not closed under linear combination. However, based on the literatures, it seems like probit model and logistic model have really closed results in real applications.
4. If this method is feasible, I will try to find a test statistic that has a nice asymptotic null distribution for the testing of linearity.

1.2.3 A simple simulation study:

Let the sample size be 300000. Let G_1 and G_2 be randomly generated from two multinomial distribution. Assuming their effects are additive with coefficient 1.5 and 1 respectively. First consider the case when no interaction is present:

```
###### For a simulation of size n:
n = 300000
set.seed(123)



##### Generate random genotype for G1 and G2, and a normal environmental factor that is unknown:
G1 = apply(X = rmultinom(n,1,prob = c(0.2,0.5,0.3))>0, FUN = "which",MARGIN = 2) - 1
G2 = apply(X = rmultinom(n,1,prob = c(0.3,0.5,0.2))>0, FUN = "which",MARGIN = 2)  -1
E <- rnorm(n, mean = 3, sd = 1)

### Case 1: If the true model is nicely additive without interaction (Assuming probit model, inverse no
beta0 <- -3
beta1 <- 1.5
beta2 <- 1
beta3 <- 1

latent_y <- beta0 + beta1*G1 + beta2*G2 + rnorm(n = n)
y <- ifelse(latent_y > 0,1,0)
data <- data.frame(y = y, G1 = G1, G2 = G2)



p <- data %>% group_by(G1,G2) %>% summarise(p = mean(y))
```

```
## `summarise()` regrouping output by 'G1' (override with `.groups` argument)
```

```
a1 <- diff(qnorm(p$p[1:3]))
a2 <- diff(qnorm(p$p[4:6]))
a3 <- diff(qnorm(p$p[7:9]))

resultG2 <- data.frame(rbind(a1,a2,a3))
rownames(resultG2) <- c("G1=0","G1=1","G1=2")
colnames(resultG2) <- c("1-0","2-1")
knitr::kable(resultG2)
```

	1-0	2-1
G1=0	1.0274514	0.9804426
G1=1	0.9972533	1.0021970
G1=2	1.0163421	0.9899940

```
### A weighted sum for G2's difference:
((table(G1)[1]) * a1 + (table(G1)[2]) * a2 + (table(G1)[3]) * a3)/n
```

```
## [1] 1.0089930 0.9942029
```

```
#### Similarly for G1:

p <- data %>% group_by(G2,G1) %>% summarise(p = mean(y))
```

```
## `summarise()` regrouping output by 'G2' (override with `.groups` argument)
```

```
a1 <- diff(qnorm(p$p[1:3]))
a2 <- diff(qnorm(p$p[4:6]))
a3 <- diff(qnorm(p$p[7:9]))
### A weighted sum:
resultG1 <- data.frame(rbind(a1,a2,a3))
rownames(resultG1) <- c("G2=0","G2=1","G2=2")
colnames(resultG1) <- c("1-0","2-1")
knitr::kable(resultG1)
```

	1-0	2-1
G2=0	1.511206	1.502123
G2=1	1.481008	1.521211
G2=2	1.502763	1.509008

```
((table(G2)[1]) * a1 + (table(G2)[2]) * a2 + (table(G2)[3]) * a3)/n
```

```
## [1] 1.494402 1.513055
```

Based on the simulation above, it can be seen that in this case, $\Phi^{-1}\mathrm{P}$ is both linear in G_1 and G_2, with the linear differences be very closed to their true coefficents. In reality, even when these differences tend to be linear, we cannot conclude that they are the correct estimates. Linearity can only help us to conclude whether interaction effect is present.

Now for the same setup above, let's add a interaction between G_1 and E with interaction effect $\beta_3 = 1$. The environmental factor E is generated from $\mathrm{N}(3, 1)$.

```
set.seed(123)
latent_y <- beta0 + beta1*G1 + beta2*G2 + beta3*G1*E + rnorm(n = n)
y <- ifelse(latent_y > 0,1,0)
data <- data.frame(y = y, G1 = G1, G2 = G2)


p <- data %>% group_by(G1,G2) %>% summarise(p = mean(y))
```

```
## `summarise()` regrouping output by 'G1' (override with `.groups` argument)
```

```
a1 <- diff(qnorm(p$p[1:3]))
a2 <- diff(qnorm(p$p[4:6]))
a3 <- diff(qnorm(p$p[7:9]))

resultG2 <- data.frame(rbind(a1,a2,a3))
rownames(resultG2) <- c("G1=0","G1=1","G1=2")
colnames(resultG2) <- c("1-0","2-1")
knitr::kable(resultG2)
```

	1-0	2-1
G1=0	1.0183975	0.9669660
G1=1	0.7145236	0.6647753
G1=2	0.4875553	0.5203277

```
### A weighted sum for G2's difference:
((table(G1)[1]) * a1 + (table(G1)[2]) * a2 + (table(G1)[3]) * a3)/n
```

```
## [1] 0.7073273 0.6819229
```

```
#### Similarly for G1:

p <- data %>% group_by(G2,G1) %>% summarise(p = mean(y))
```

```
## `summarise()` regrouping output by 'G2' (override with `.groups` argument)
```

```
a1 <- diff(qnorm(p$p[1:3]))
a2 <- diff(qnorm(p$p[4:6]))
a3 <- diff(qnorm(p$p[7:9]))
### A weighted sum:
resultG1 <- data.frame(rbind(a1,a2,a3))
rownames(resultG1) <- c("G2=0","G2=1","G2=2")
colnames(resultG1) <- c("1-0","2-1")
knitr::kable(resultG1)
```

	1-0	2-1
G2=0	4.082256	1.615465
G2=1	3.778382	1.388497
G2=2	3.476191	1.244049

```
((table(G2)[1]) * a1 + (table(G2)[2]) * a2 + (table(G2)[3]) * a3)/n
```

```
## [1] 3.808773 1.427475
```

Now, it can be seen from the R output above that, $\Phi^{-1}\mathrm{P}$ seems to be linear in G_2 with linear rate approximately 0.7. But $\Phi^{-1}\mathrm{P}$ is definitely not linear in G_1 as 3.8 is not closed to 1.42. This is within our expectation since the interaction is only between G_1 and E.

1.3 Potential Method 2: By modeling the interaction term as a random slope:

First, let's rewrite our previous latent variable specification:

$$\begin{aligned} Y_i^* &= \beta_0 + \beta_1 G_1 + \beta_2 G_2 + \beta_3 G_1 \times E_i + \epsilon_i \\ &= \beta_0 + \beta_1 G_1 + \beta_2 G_2 + U_i * G_1 + \epsilon_i \\ Y_i &= I\{Y_i^* > 0\} \\ U_i &= \beta_3 * E_i \end{aligned} \tag{7}$$

Here U_i can be thought as a random effect (random slope), being drawn from distribution $\mathrm{N}(0, \sigma_u^2)$. Notice that $\sigma_u^2 = \beta_3^2 \sigma_E^2$. Therefore, testing for $\beta_3 = 0$ is equivalent to testing $\sigma_u^2 = 0$ for the random effects. In this case, we do not need to restrict our distribution to the probit model anymore. Since both probit model and logistic model are flexible enough to incorporate an observations-level random slopes. (There shouldn't be any identifiability problem with have too many random slopes(same number as observations), as including an observations-level random intercepts is a common trick to account for overdispersion in Poisson regression.)

1.3.1 A simple simulation study:

Fitting this probit model with observations-level random slope is computationally very hard, and lme4 seems to converge very slow when there is an interaction effect in the true model, so we fit models only using the first 30000 rows of the dataset for computational efficiency. We still use the same setting as previous to generate our data, i.e. true model is probit model, and we will try to fit both probit regression and logistic regression to see how they behave:

Let's fit a probit model with observations-level random slope using lme4: First with interaction between G_1 and E

```
set.seed(123)
latent_y <- beta0 + beta1*G1 + beta2*G2 + beta3*G1*E + rnorm(n = n)
y <- ifelse(latent_y > 0,1,0)
data <- data.frame(y = y, G1 = G1, G2 = G2)
data$OLRE <- 1:nrow(data)
model1 <- glmer(y~ G1 + G2 + (0+G1|OLRE) ,family = binomial(link = "probit"), data = data[1:30000,])
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.0202522 (tol = 0.002, component 1)
```

```
summary(model1)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( probit )
## Formula: y ~ G1 + G2 + (0 + G1 | OLRE)
##    Data: data[1:30000, ]
##
##      AIC      BIC   logLik deviance df.resid
##   4820.3   4853.6  -2406.2   4812.3    29996
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -0.4301  0.0000  0.0000  0.0001 26.2608
##
## Random effects:
##  Groups Name Variance Std.Dev.
##  OLRE   G1   237.4    15.41
## Number of obs: 30000, groups:  OLRE, 30000
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.978577   0.004425  -673.0   <2e-16 ***
## G1           7.631677   0.004448  1715.8   <2e-16 ***
## G2           0.983989   0.004130   238.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##    (Intr) G1
## G1 -0.020
## G2 -0.040 -0.036
## convergence code: 0
## Model failed to converge with max|grad| = 0.0202522 (tol = 0.002, component 1)
```

```
#### The variance of the random slope seems to be very large, suggesting the presence of interaction
```

There is some warnings from lme4 about the convergence, but the fitted result can still be valid. It can be seen that the estimated variance of the random slope is quite large (237.4). The estimated parameters are not closed to the truth in this case.

Then, try again to fit a probit model when the true model doesn't have interaction term:

```r
### If the true model does not have interaction
set.seed(123)
latent_y <- beta0 + beta1*G1 + beta2*G2 + rnorm(n = n)
y <- ifelse(latent_y > 0,1,0)
data <- data.frame(y = y, G1 = G1, G2 = G2)
data$OLRE <- 1:nrow(data)
model2 <- glmer(y~ G1 + G2 + (0+G1|OLRE) ,family = binomial(link = "probit"), data = data[1:30000,])
summary(model2)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( probit )
## Formula: y ~ G1 + G2 + (0 + G1 | OLRE)
##    Data: data[1:30000, ]
##
##      AIC      BIC   logLik deviance df.resid
##  24990.9  25024.2 -12491.5  24982.9    29996
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -6.3731 -0.6630 -0.1496  0.4311 28.7505
##
## Random effects:
##  Groups Name Variance Std.Dev.
##  OLRE   G1   0.004607 0.06788
## Number of obs: 30000, groups:  OLRE, 30000
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.03359    0.03831  -79.19   <2e-16 ***
## G1           1.50949    0.02141   70.50   <2e-16 ***
## G2           1.01741    0.01726   58.93   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##    (Intr) G1
## G1 -0.907
## G2 -0.806  0.580
```

This time the probit model doesn't take that long to be fitted. We can see that without the interaction effect in the true model, the estimated random slope has very small variance (0.004607), which is consistent with our expectation. The other parameters in the model are estimated accurately. The estimated parameters are very accurate.

Since fitting probit model is computationally hard for lme4, what if we fit a logistic regression instead? First try it when the true model doesn't contain interaction:

```r
set.seed(123)
latent_y <- beta0 + beta1*G1 + beta2*G2 + rnorm(n = n)
y <- ifelse(latent_y > 0,1,0)
data <- data.frame(y = y, G1 = G1, G2 = G2)
data$OLRE <- 1:nrow(data)
model3 <- glmer(y~ G1 + G2 + (0+G1|OLRE) ,family = binomial(link = "logit"), data = data[1:30000,])
summary(model3)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: y ~ G1 + G2 + (0 + G1 | OLRE)
##    Data: data[1:30000, ]
##
##      AIC      BIC   logLik deviance df.resid
##  25023.2  25056.4 -12507.6  25015.2    29996
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.2610 -0.6385 -0.1707  0.4037 14.3478
##
## Random effects:
##  Groups Name Variance Std.Dev.
##  OLRE   G1   0.06464  0.2542
## Number of obs: 30000, groups:  OLRE, 30000
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.32720    0.08066  -66.04   <2e-16 ***
## G1           2.65670    0.04492   59.15   <2e-16 ***
## G2           1.79198    0.03395   52.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##    (Intr) G1
## G1 -0.934
## G2 -0.850  0.678
```

This time lme4 doesn't have any convergence warnings, maybe becasue fitting logistic regression is computationally easier than a probit model. The result is still consistent if we use logistic regression instead, σ_u^2 is estimated to be 0.06464, which is still very small. The estimated coefficents are not very closed to the truth becasue the true underlying model in this simulation is a probit model instead.

Let's try again with the true model does have the interaction effect:

```
set.seed(123)
latent_y <- beta0 + beta1*G1 + beta2*G2 + beta3 * G1*E + rnorm(n = n)
y <- ifelse(latent_y > 0,1,0)
data <- data.frame(y = y, G1 = G1, G2 = G2)
data$OLRE <- 1:nrow(data)
model3 <- glmer(y~ G1 + G2 + (0+G1|OLRE) ,family = binomial(link = "logit"), data = data[1:30000,])
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.127451 (tol = 0.002, component 1)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, : Model is nearly unide
##  - Rescale variables?;Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
```

```
summary(model3)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
```

```
## Formula: y ~ G1 + G2 + (0 + G1 | OLRE)
##    Data: data[1:30000, ]
##
##      AIC      BIC   logLik deviance df.resid
##   4685.8   4719.0  -2338.9   4677.8    29996
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -0.4419  0.0000  0.0000  0.0007 21.1998
##
## Random effects:
##  Groups Name Variance Std.Dev.
##  OLRE   G1   2833     53.23
## Number of obs: 30000, groups:  OLRE, 30000
##
## Fixed effects:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.1079801  0.0006049  -10097   <2e-16 ***
## G1          18.4230853  0.0005878   31342   <2e-16 ***
## G2           2.2374098  0.0006045    3701   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##    (Intr) G1
## G1  0.020
## G2 -0.231  0.019
## convergence code: 0
## Model failed to converge with max|grad| = 0.127451 (tol = 0.002, component 1)
## Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
```

Again, lme4 seems to have some convergence warnings when the true model has interaction. Now with the interaction term in the true model, σ_u^2 is estimated to be 3034 which is very large. To deal with the potential convergence issue, we could try some other optimizers in the lme4 package.

For the next step, we can implement some boundary test to formally test the hypothesis $\sigma_u^2 = 0$ based on likelihood ratio, and try to come up with some way to do the joint testing of G's main effect and interaction effect.

1.4 Difference between two potential methods

1. The first method relies on the assumption that the true underlying model is probit model, and the distribution of E is normal. These assumptions shouldn't be too restrictive as it is said in the literature that probit model and logistic model tend to give similar results. However, the second method can be used for both probit model and logistic model. The only assumption in the second method is that E follows a normal distribution.
2. The next step for the first method is to develop a test statistic for testing the linearity. While for the second method, it seems like there are plenty of tools of testing at boudnary to test $\sigma_u = 0$, using likelihood ratio. It seems like in the second method, jointly testing for the main effect and interaction effect

3. For the simulations of sample size 300000, the first method is very efficent to compute as it basically just computes nine sample proportions and compute their difference. If we can find a good test statistic for this, the hypothesis testing will be efficient to carry out and scale to larger sample. The second method takes a very long time to converge when the interaction is actually present in the model, and lme4 tends to give some warnings about the potential convergence problems if a probit model is fitted and underlying model has the interaction effect. For a larger sample with more regressors, the computational loads will be bigger for the second method.