

Power of Wald test for interaction

Ziang Zhang

09/03/2021

Contents

1	Power of Wald test:	2
1.1	For one parameter:	2
1.2	General Expression and Non-centrality parameter:	2
2	When will other parameters matter for our power?	3
2.1	Wald test for <code>lm</code> :	3
2.2	Wald test for <code>glm</code> :	3
3	Computational Experiment:	4
3.1	Assume data generated from ordinary linear model	4
3.2	Assume data generated from probit regression model	6

1 Power of Wald test:

1.1 For one parameter:

If we consider using Wald test to test the null hypothesis $H_0 : \theta = \theta_0$, the test statistic will be

$$\sqrt{\frac{n}{I^{-1}(\hat{\theta})}}(\hat{\theta} - \theta_0)$$

where $\hat{\theta}$ is the MLE of θ and $I^{-1}(\hat{\theta})$ is inverse of the fisher information evaluated at the MLE.

Under null hypothesis, this test statistic follows a standard normal distribution. If the alternative hypothesis $H_a : \theta = \theta_1 \neq \theta_0$, then the power of our test can be computed as:

$$1 - \Phi(\Delta + z_{\alpha/2}) + \Phi(\Delta - z_{\alpha/2})$$

where $\Delta = \sqrt{\frac{n}{I^{-1}(\hat{\theta})}}(\theta_0 - \theta_1)$.

The important information above is that the power of Wald test will depend on several things at the same time:

- The difference of $\theta_0 - \theta_1$
- The sample size n
- The **true** information's inverse $I^{-1}(\theta_1)$

In general, higher power is achieved when Δ is large. That implies, we have higher power if we have larger $|\theta_0 - \theta_1|$, n or smaller $I^{-1}(\theta_1)$.

1.2 General Expression and Non-centrality parameter:

In general for a vector of parameter $\beta \in \mathbb{R}^p$, if we are interested in testing the null hypothesis of linear combination of β : $L\beta = 0$, then the test statistic will be

$$(L\hat{\beta})^T L I_n^{-1}(\hat{\beta}) L^T (L\hat{\beta})$$

which follows a chi-square distribution with 1 degree of freedom under the null hypothesis. The matrix $I_n^{-1}(\hat{\beta})$ is the inverse of the fisher information matrix of the whole sample, evaluated at $\beta = \hat{\beta}$.

Specifically, if we want to test the case where $L = (0, 0, \dots, 1, 0, \dots, 0)$, then our test statistic reduced to

$$I_n^{-1}(\hat{\beta})_{[i,i]}(\hat{\beta}_i)^2$$

where β_i is the i -th component of β , $I_n^{-1}(\hat{\beta})_{[i,i]}$ is the i -th diagonal term of $I_n^{-1}(\hat{\beta})$. Under the alternative hypothesis that $\beta_i = \beta_{i1} \neq 0$, the limiting distribution will be a non-central chi-square distribution with non-centrality parameter being

$$I_n^{-1}(\beta)_{[i,i]} \beta_{i1}$$

here $I_n^{-1}(\beta)$ is inverted fisher information matrix evaluated at the true parameter vector β .

Note that to compute $I_n^{-1}(\beta)$, we may need more than just β_{i1} . Even though we are only interested in testing β_i , that does not imply we can compute $I_n^{-1}(\beta)$ just using the true value of β_i . We will discuss this problem in the next section in details.

2 When will other parameters matter for our power?

2.1 Wald test for `lm`:

To answer this question, we need to know when will $I_n^{-1}(\beta)$ depend on parameters other than β_i . It turns out that if the true underlying model of our data is a linear regression model, $I_n^{-1}(\beta)$ will not depend on any parameter. However, if the true underlying model is not an ordinary linear regression model, but a generalized linear regression model, then $I_n^{-1}(\beta)$ will depend on all the regression parameters.

If we consider the linear regression model:

$$y = \beta_0 + \beta_G G + \beta_E E + \beta_{GE} G \times E + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$

Then the fisher information matrix at β can be computed as

$$XX^T / \sigma^2$$

Note that this matrix is only affected by the variance parameter σ^2 , and not affected by any regression parameter β . Therefore, the power function of Wald test for $\beta_{GE} = 0$ will only depend on the magnitude of the true value of β_{GE} .

2.2 Wald test for `glm`:

On the other hand, if the true data generating model is

$$\mu = g^{-1}(\beta_0 + \beta_G G + \beta_E E + \beta_{GE} G \times E)$$

where $g(\cdot)$ is a specified link function connecting the linear predictor with the mean of y .

In this case, the information matrix can be written as

$$XW(\beta)X^T$$

where the matrix $W(\beta)$ is a diagonal matrix with each term depends on all the regression parameter β .

For a specific example, if we are using probit regression model, then the matrix $W(\beta)$ will be

$$W(\beta) = \text{diag}\left\{\frac{\phi^2(s_i)}{\Phi(s_i)(1 - \Phi(s_i))}\right\}$$

where $s_i = \beta_0 + \beta_G G_i + \beta_E E_i + \beta_{GE} G_i \times E_i$ and ϕ, Φ are the density and cdf of standard normal distribution respectively.

3 Computational Experiment:

Here we illustrate the phenomenon above using some computational experiment, and compare the empirical power and the theoretical power:

3.1 Assume data generated from ordinary linear model

In this section, we consider continuous trait generated from a linear regression model:

First case, we consider the true $\beta_{GE} = 0.1$ and true main effect $\beta_E = 1$:

```
set.seed(100)
N <- 1000
G <- sample(c(0,1,2),size = N, replace = T, prob = c(0.16,0.48, 0.36))
E <- rnorm(N)
beta0 <- -1
betaG <- 0.3
betaE <- 1
betaGE <- 0.1
ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)

mod <- glm(ylat~G+E+I(G*E), family = gaussian(link = "identity"))

#### Get the design matrix:
X <- cbind(rep(1,N),mod$model[, -1])

### Compute the weight matrix W:
beta <- c(beta0,betaG,betaE,betaGE)

### The true information matrix
I <- as.matrix(t(X)) %*% as.matrix(X)

#### Invert to get the true covariance matrix
V <- solve(I)

### Compute the power function

### Assume d = beta0 - beta1, where beta0 = 0 is what we are testing as null:
delta <- sqrt(1/V[4,4])*(0-beta[4])
alpha <- 0.05
Power <- 1- pnorm(delta - qnorm(alpha/2)) + pnorm(delta + qnorm(alpha/2))
Power

## [1] 0.6030156
```

```
#### Illustrate that this power is correct:
set.seed(100)
p1 <- c()
for (i in 1:1000) {
  ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)
  mod <- glm(ylat~G+E+I(G*E), family = gaussian(link = "identity"))
  p1[i] <- summary(mod)$coefficient[4,4]
}
emp_power <- mean(p1 <= alpha)
emp_power
```

```
## [1] 0.575
```

Second case, we consider the true $\beta_{GE} = 0.1$ fixed but change the true main effect β_E from 1 to 0:

```
set.seed(100)
N <- 1000
G <- sample(c(0,1,2),size = N, replace = T, prob = c(0.16,0.48, 0.36))
E <- rnorm(N)
beta0 <- -1
betaG <- 0.3
betaE <- 0
betaGE <- 0.1
ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)

mod <- glm(ylat~G+E+I(G*E), family = gaussian(link = "identity"))

#### Get the design matrix:
X <- cbind(rep(1,N),mod$model[, -1])

### Compute the weight matrix W:
beta <- c(beta0,betaG,betaE,betaGE)

### The true information matrix
I <- as.matrix(t(X)) %*% as.matrix(X)

#### Invert to get the true covariance matrix
V <- solve(I)

### Compute the power function

### Assume d = beta0 - beta1, where beta0 = 0 is what we are testing as null:
delta <- sqrt(1/V[4,4])*(0-beta[4])
alpha <- 0.05
Power <- 1- pnorm(delta - qnorm(alpha/2)) + pnorm(delta + qnorm(alpha/2))
Power
```

```
## [1] 0.6030156
```

```
#### Illustrate that this power is correct:
set.seed(100)
p1 <- c()
for (i in 1:1000) {
  ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)
  mod <- glm(ylat~G+E+I(G*E), family = gaussian(link = "identity"))
  p1[i] <- summary(mod)$coefficient[4,4]
}
emp_power <- mean(p1 <= alpha)
emp_power
```

```
## [1] 0.575
```

We can see that both the theoretical power and empirical power are not changed at all.

3.2 Assume data generated from probit regression model

Here, we consider the binary trait generated from a probit regression model:

First case, we consider the true $\beta_{GE} = 0.1$ and true main effect $\beta_E = 1$:

```
set.seed(100)
N <- 1000
G <- sample(c(0,1,2),size = N, replace = T, prob = c(0.16,0.48, 0.36))
E <- rnorm(N)
beta0 <- -1
betaG <- 0.3
betaE <- 1
betaGE <- 0.1
ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)
y <- ifelse(ylat >= 0, 1, 0)

mod <- glm(y~G+E+I(G*E), family = binomial(link = "probit"))

#### Get the design matrix:
X <- cbind(rep(1,N),mod$model[, -1])

### Compute the weight matrix W:
beta <- c(beta0,betaG,betaE,betaGE)
#beta <- as.numeric(mod$coefficients)

w <- c()
for (i in 1:N) {
  si <- as.numeric(as.numeric(X[i,]) %*% beta)
  w[i] <- (dnorm(si)^2)/(pnorm(si)*(1-pnorm(si)))
}

### The true information matrix
I <- as.matrix(t(X)) %*% diag(w,nrow = N,ncol = N) %*% as.matrix(X)
```

```
#### Invert to get the true covariance matrix
V <- solve(I)

### Compute the power function

### Assume d = beta0 - beta1, where beta0 = 0 is what we are testing as null:
delta <- sqrt(1/V[4,4])*(0-beta[4])
alpha <- 0.05
Power <- 1- pnorm(delta - qnorm(alpha/2)) + pnorm(delta + qnorm(alpha/2))
Power
```

```
## [1] 0.1723627
```

```
#### Illustrate that this power is correct:
set.seed(100)
p1 <- c()
for (i in 1:1000) {
  ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)
  y <- ifelse(ylat >=0, 1, 0)
  mod <- glm(y~G+E+I(G*E), family = binomial(link = "probit"))
  p1[i] <- summary(mod)$coefficient[4,4]
}
emp_power <- mean(p1 <= alpha)
emp_power
```

```
## [1] 0.167
```

Second case, we consider the true $\beta_{GE} = 0.1$ fixed but change the true main effect β_E from 1 to 0:

```
set.seed(100)
N <- 1000
G <- sample(c(0,1,2),size = N, replace = T, prob = c(0.16,0.48, 0.36))
E <- rnorm(N)
beta0 <- -1
betaG <- 0.3
betaE <- 0
betaGE <- 0.1
ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)
y <- ifelse(ylat >=0, 1, 0)

mod <- glm(y~G+E+I(G*E), family = binomial(link = "probit"))

#### Get the design matrix:
X <- cbind(rep(1,N),mod$model[, -1])

### Compute the weight matrix W:
beta <- c(beta0,betaG,betaE,betaGE)
#beta <- as.numeric(mod$coefficients)
```

```

w <- c()
for (i in 1:N) {
  si <- as.numeric(as.numeric(X[i,]) %*% beta)
  w[i] <- (dnorm(si)^2)/(pnorm(si)*(1-pnorm(si)))
}

### The true information matrix
I <- as.matrix(t(X)) %*% diag(w,nrow = N,ncol = N) %*% as.matrix(X)

#### Invert to get the true covariance matrix
V <- solve(I)

### Compute the power function

### Assume d = beta0 - beta1, where beta0 = 0 is what we are testing as null:
delta <- sqrt(1/V[4,4])*(0-beta[4])
alpha <- 0.05
Power <- 1- pnorm(delta - qnorm(alpha/2)) + pnorm(delta + qnorm(alpha/2))
Power

```

```
## [1] 0.3541068
```

```

#### Illustrate that this power is correct:
set.seed(100)
p1 <- c()
for (i in 1:1000) {
  ylat <- beta0 + betaG*G + betaE*E + betaGE*G*E + rnorm(N)
  y <- ifelse(ylat >=0, 1, 0)
  mod <- glm(y~G+E+I(G*E), family = binomial(link = "probit"))
  p1[i] <- summary(mod)$coefficient[4,4]
}
emp_power <- mean(p1 <= alpha)
emp_power

```

```
## [1] 0.35
```

We can see that both the theoretical power and empirical power increased greatly.