Análisis de datos proyecto Final de experimentos I

Daniel Sibaja Salazar

2024-05-16

Preparación

Paquetes

```
library(readxl)
library(lme4)
```

```
## Loading required package: Matrix
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(qqplotr)
```

```
##
## Attaching package: 'qqplotr'

## The following objects are masked from 'package:ggplot2':
##
##     stat_qq_line, StatQqLine
```

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode
```

Se cargan los datos. Es necessario pasar las observaciones a tipo factor.

```
base1=read_xlsx("final_registro_cloud.xlsx",col_names = FALSE)
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
```

```
names(base1)=c("Tiempo","BL","Nube","Denied","Right","Solicitudes","ID","Bloque")
base1$BL=as.factor(base1$BL)
base1$Solicitudes=as.factor(base1$Solicitudes)
base1$Nube=as.factor(base1$Nube)




base2=read_xlsx("final_registro_local.xlsx",col_names = FALSE)
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
```

```
names(base2)=c("Tiempo","BL","Nube","Denied","Right","Solicitudes","ID","Bloque")
base2$BL=as.factor(base2$BL)
base2$Solicitudes=as.factor(base2$Solicitudes)
base2$Nube=as.factor(base2$Nube)
```

Podemos unir las bases para que sea solo una tabla grande y ver los tratamientos

```
base=rbind(base1,base2)
table(base$BL,base$Nube,base$Solicitudes)
```

```
## , ,  = 10
##
##
##       Nube Local
##   BL    90    80
##   ML    90    80
##
## , ,  = 100
##
##
##       Nube Local
##   BL   900   800
##   ML   895   800
##
## , ,  = 500
##
##
##       Nube Local
##   BL 3759  3866
##   ML 3758  3865
```

Vemos que hay un desbalance con los tratamientos, según el experto se pueden eliminar parcelas, se eliminan las que causan el desbalance

```
quitar1=which(base1$Bloque=="N109")
base1=base1[-quitar1,]

quitar2=which(base1$Bloque=="N1009")
base1=base1[-quitar2,]

quitar3=which(base1$Bloque=="N5009")
base1=base1[-quitar3,]

quitar4=which(base1$Bloque=="N5006")
base1=base1[-quitar4,]

quitar5=which(base2$Bloque=="L5006")
base2=base2[-quitar5,]

quitar6=which(base1$Bloque=="N5008")
base1=base1[-quitar6,]

quitar7=which(base2$Bloque=="L5008")
base2=base2[-quitar7,]

quitar8=which(base1$Bloque=="N5004")
base1=base1[-quitar8,]

quitar9=which(base2$Bloque=="L5004")
base2=base2[-quitar9,]

quitar10=which(base1$Bloque=="N1007")
base1=base1[-quitar10,]
```

```r
quitar11=which(base2$Bloque=="L1007")
base2=base2[-quitar11,]

base=rbind(base1,base2)
table(base$BL,base$Nube,base$Solicitudes)
```

```
## , ,  = 10
##
##
##      Nube Local
##   BL   80    80
##   ML   80    80
##
## , ,  = 100
##
##
##      Nube Local
##   BL  700   700
##   ML  700   700
##
## , ,  = 500
##
##
##      Nube Local
##   BL 2500  2500
##   ML 2500  2500
```

Se soluciona el desbalance, ahora los bloques deben de coincidir.

Descriptivos

```r
options(scipen = 999)
tapply(base$Tiempo,list(base$BL,base$Nube,base$Solicitudes),mean)
```

```
## , , 10
##
##         Nube       Local
## BL 0.0008677 0.002882075
## ML 0.1918961 0.898266438
##
## , , 100
##
##            Nube       Local
## BL 0.0008544457 0.002883869
## ML 0.1564820486 0.907232110
##
## , , 500
##
##            Nube      Local
## BL 0.0009379336 0.00289288
## ML 0.1561837404 0.89883775
```

```
tapply(base$Tiempo,list(base$BL,base$Nube,base$Solicitudes),var)
```

```
## , , 10
##
##               Nube          Local
## BL 0.0000002244848 0.000008591479
## ML 0.0315259459741 0.002696818658
##
## , , 100
##
##               Nube          Local
## BL 0.0000004528595 0.000009014041
## ML 0.0000608513538 0.007039751061
##
## , , 500
##
##              Nube          Local
## BL 0.000004491367 0.000009103962
## ML 0.000038641015 0.005070112955
```

Observamos que son valores que están cercanos a 0, esto se debe a que la variable respuesta está dada en segundos y pero los valores de tiempo son mucho más pequeños que un segundo.

```
base$TiempoT=base$Tiempo*100
tapply(base$TiempoT,list(base$BL,base$Nube,base$Solicitudes),mean)
```

```
## , , 10
##
##        Nube      Local
## BL  0.08677  0.2882075
## ML 19.18961 89.8266438
##
## , , 100
##
##           Nube      Local
## BL  0.08544457  0.2883869
## ML 15.64820486 90.7232110
##
## , , 500
##
##           Nube     Local
## BL  0.09379336  0.289288
## ML 15.61837404 89.883775
```

```
tapply(base$TiempoT,list(base$BL,base$Nube,base$Solicitudes),var)
```

```
## , , 10
##
##             Nube       Local
## BL   0.002244848  0.08591479
## ML 315.259459741 26.96818658
```

```
##
## , , 100
##
##           Nube       Local
## BL 0.004528595  0.09014041
## ML 0.608513538 70.39751061
##
## , , 500
##
##          Nube       Local
## BL 0.04491367  0.09103962
## ML 0.38641015 50.70112955
```

Parece que en centisegundos todo tiene un poco más de coherencia.

ESTO SE DEBE DE PONER EN UNA TABLA

El modelo

El modelo es un modelo mixto de parcelas divididas, el cual debe de cumplir los supuestos de normalidad y de homocedasticidad.

Es el siguiente:

$$\mu_{ijk} = \mu + \alpha_i + \beta_j + \delta_k + (\alpha\beta)_{ij} + (\alpha\delta)_{ik} + (\beta\delta)_{jk} + (\alpha\beta\delta)_{ijk} + \nu_l$$

Donde:

- μ es la media general
- α_i es el efecto del nivel i del Factor BlackList
- β_j es el efecto del nivel j del Factor Nube
- δ_k es el efecto del nivel k del Factor Solicitudes
- $(\alpha\beta)_{ij}$ es el efecto de la interacción entre BlackList y Nube
- $(\alpha\delta)_{ik}$ es el efecto de la interacción entre BlackList y Solicitudes
- $(\beta\delta)_{jk}$ es el efecto de la interacción entre Nube y Solicitudes
- ν_l Es el efecto de la parcela.

Se ajusta un modelo mixto

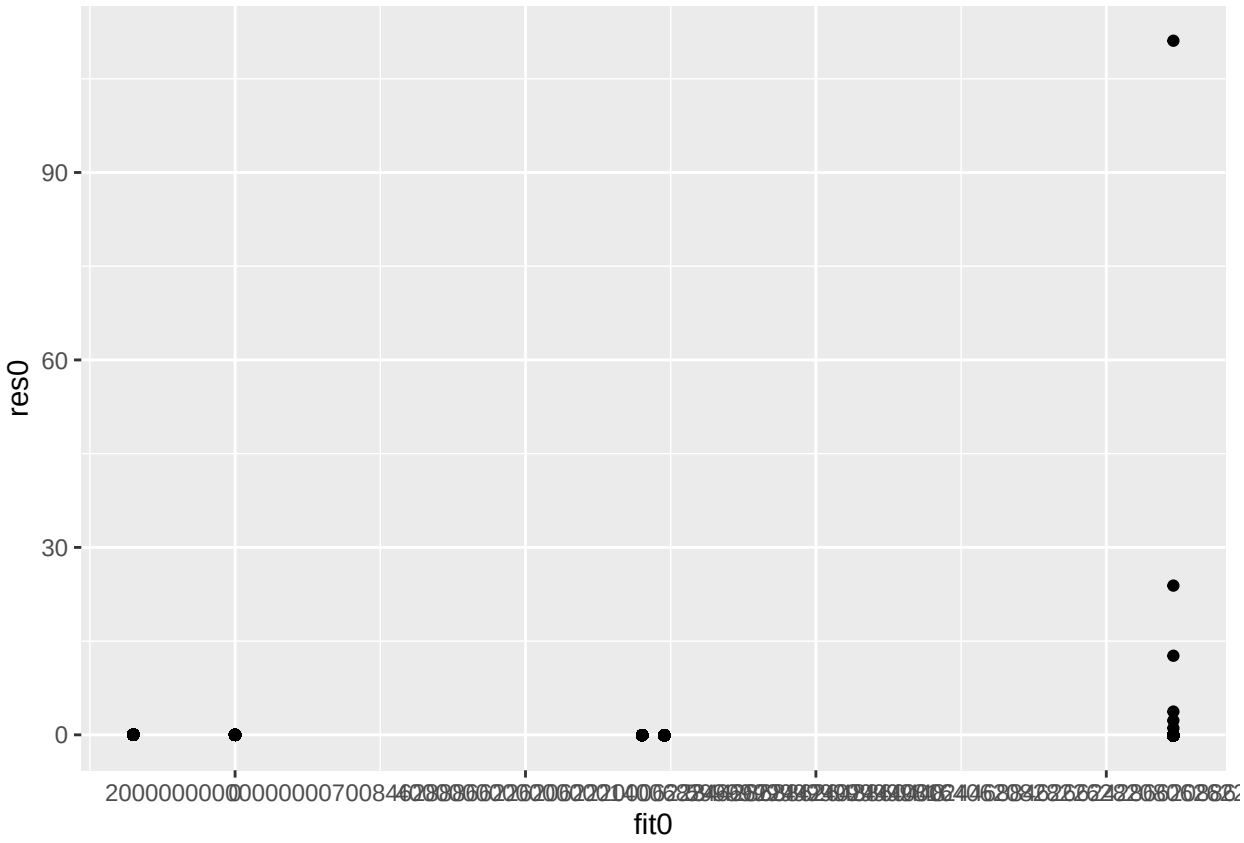
```
modp0=lmer(exp(TiempoT)~BL*Nube*Solicitudes+(1|Bloque),data = base)
summ0=summary(modp0)
```

Supuestos del modelo

Se asume el supuesto de independencia, se analiza entonces homocedasticidad y normalidad. Se oueden usar qqPlots, la prueba de Kolmogorov-Smirnov, el gráfico de predichos contra residuos y la prueba de levene.

```
res0=summ0$residuals
fit0=fitted(modp0)
dffp=data.frame(res0,fit0)
```

```
ggplot(data = dffp, aes(x = fit0, y = res0)) +
  geom_point()
```




```
leveneTest(res0~BL*Nube*Solicitudes,data = base)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value Pr(>F)
## group    11  0.9522 0.4884
##       13108
```

Tanto en el gráfico de residuos contra predichos como en la prueba de Levene, no hay razón para creer que hay homocedasticidad.

```
ks.test(res0,"pnorm")
```

```
## Warning in ks.test.default(res0, "pnorm"): ties should not be present for the
## Kolmogorov-Smirnov test

##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  res0
## D = 0.49479, p-value < 0.00000000000000022
## alternative hypothesis: two-sided
```

```
residuos=as.data.frame(res0)
res.stnd = scale(residuos$res0) # <- Residuos estandarizados.
ggplot(data = residuos, mapping = aes(sample = res.stnd )) + stat_qq_point() +
      ylab("Residuos Estandarizados") + xlab("Norm Quantiles") + ggtitle("Normal QQ Plot")  +
```

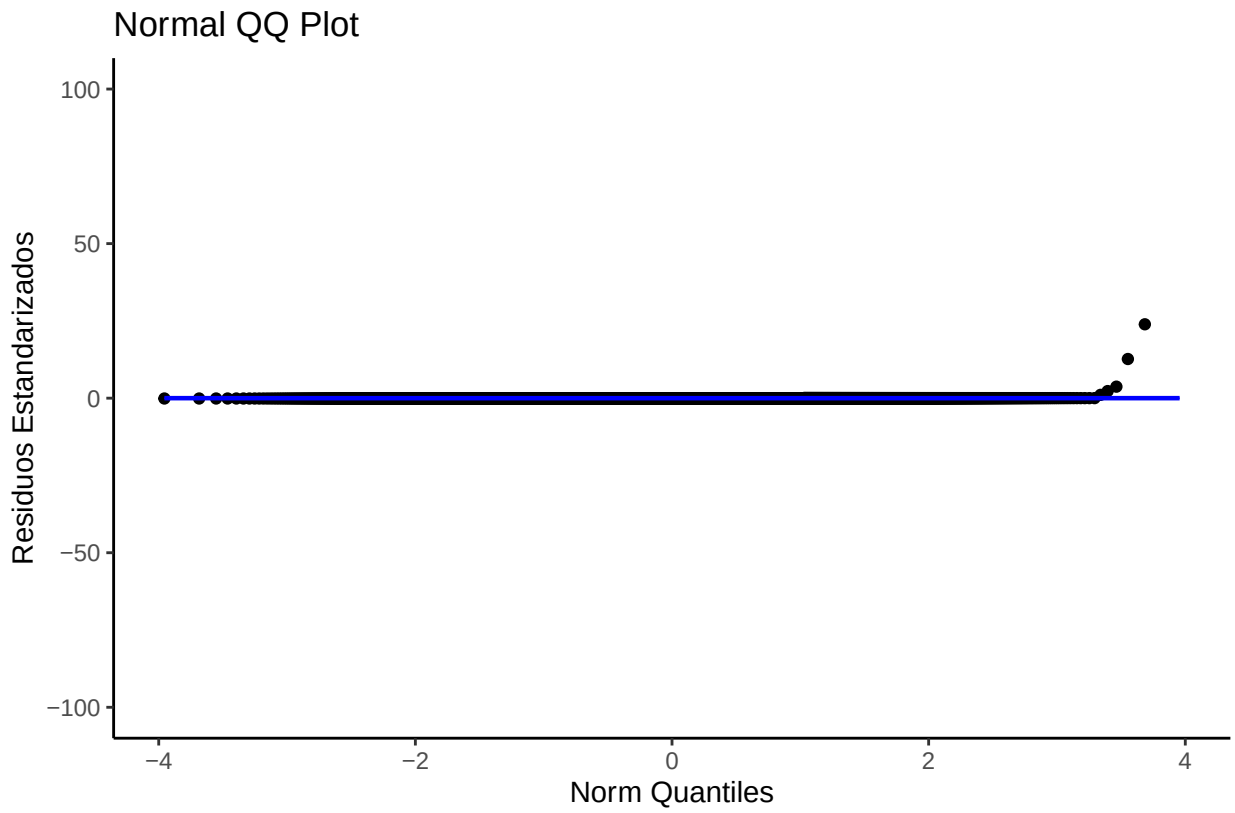


Grafico 1

De igual forma, tanto en el gráfico como en la prueba de Kolmogorov se puede distinguir normalidad.

En ambos gráficos hay un dato extremo.

Vamos a intentar cambiar la respuesta por el promedio del tratamiento al que pertenecen

```
which(res0>20,)
```

```
## 12290 12292
## 12290 12292
```

```
#tapply(base$TiempoT,list(base$BL,base$Nube,base$Solicitudes),mean) ##El promedio es 19 y estos tienen
baseA=base
baseA[c(5325,5327),9]=c(19.18961,19.18961)
```

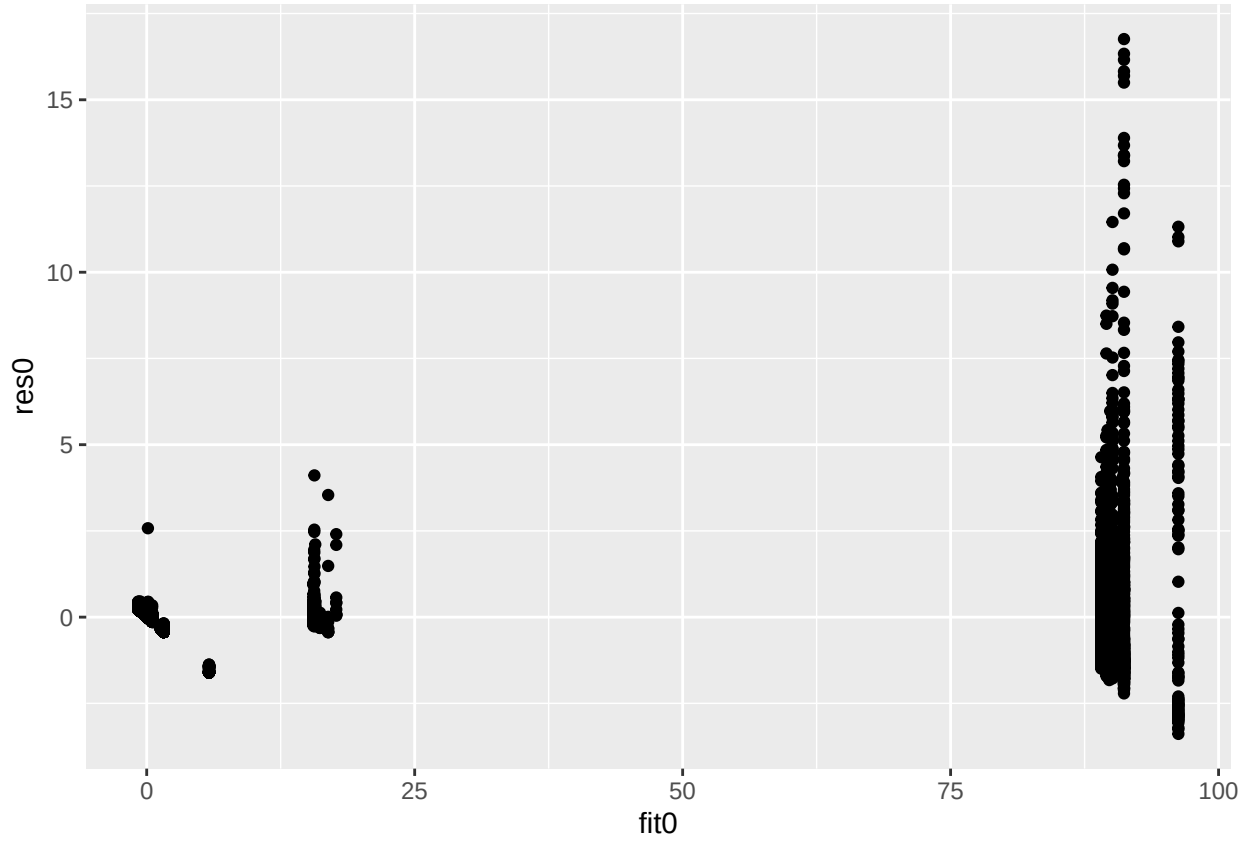
Podemos volver a hacer todo a ver qué tal nos va

```
modp0=lmer(TiempoT~BL*Nube*Solicitudes+(1|Bloque),data = baseA)
summ0=summary(modp0)
```

```r
# Homocedasticidad

res0=summ0$residuals
fit0=fitted(modp0)
dffp=data.frame(res0,fit0)


ggplot(data = dffp, aes(x = fit0, y = res0)) +
  geom_point()
```




```r
leveneTest(res0~BL*Nube*Solicitudes,data = baseA)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value                Pr(>F)
## group    11  310.22 < 0.00000000000000022 ***
##       13108
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Normalidad

ks.test(res0,"pnorm")
```

```
## Warning in ks.test.default(res0, "pnorm"): ties should not be present for the
## Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  res0
## D = 0.28311, p-value < 0.00000000000000022
## alternative hypothesis: two-sided
```

```
residuos=as.data.frame(res0)
res.stnd = scale(residuos$res0) # <- Residuos estandarizados.
ggplot(data = residuos, mapping = aes(sample = res.stnd )) + stat_qq_point() +
      ylab("Residuos Estandarizados") + xlab("Norm Quantiles") + ggtitle("Normal QQ Plot")  +
```

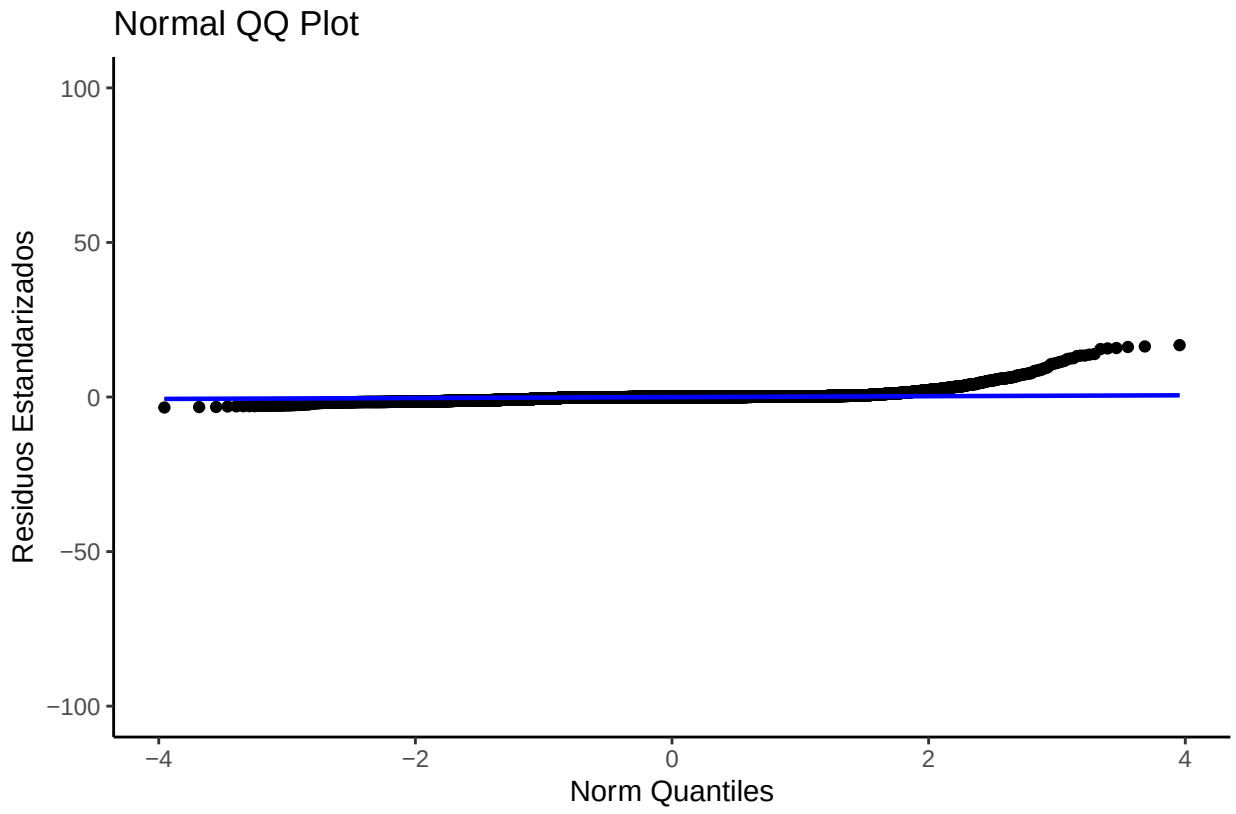


Grafico 1

No mejoró mucho

Por esta razón vamos a intentar a justar un modelo Generalizado mixto, a ver si eso nos ayuda a arreglar los supuestos.

Modelo generalizado Mixto

```
modp1=glmer(TiempoT~BL*Nube*Solicitudes+(1|Bloque),family = Gamma(link = "log"),data = base)
```

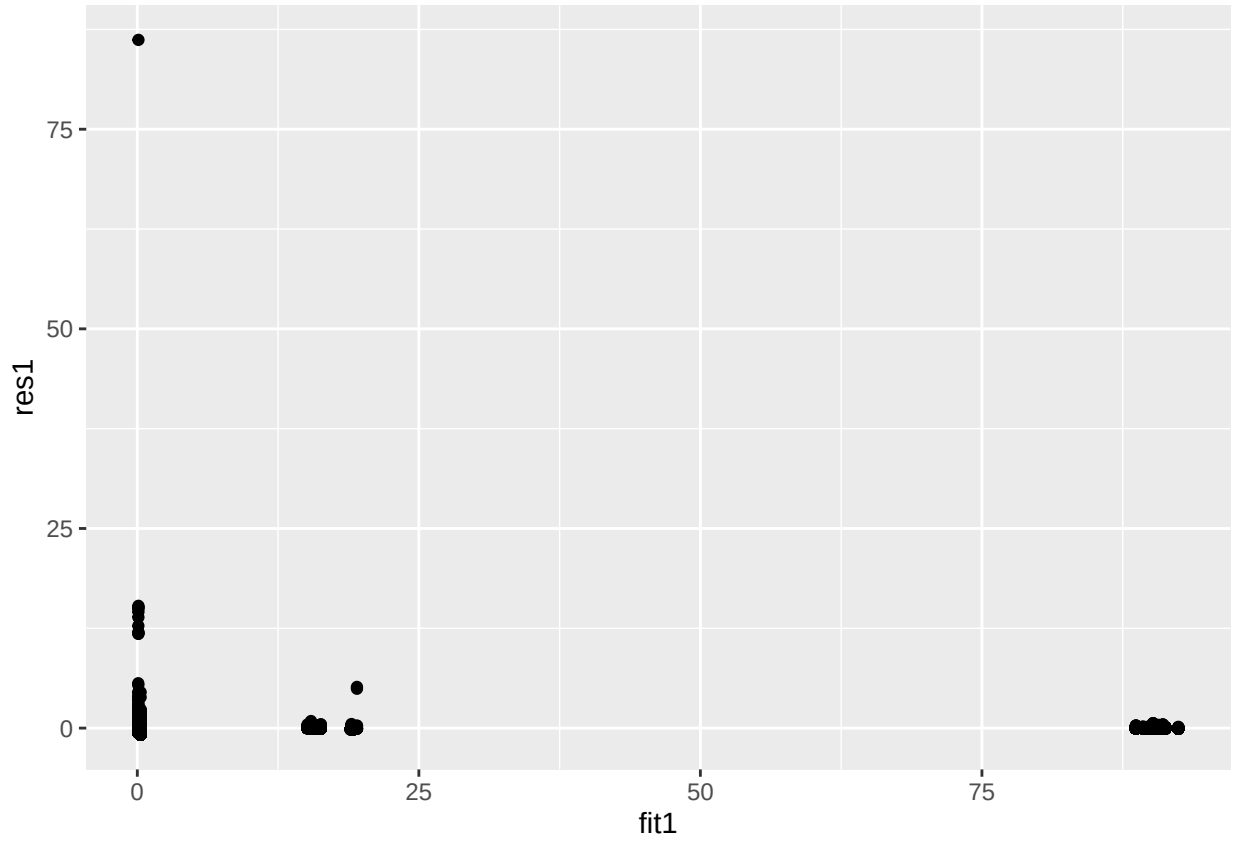
```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.00799993 (tol = 0.002, component 1)
```

```
summ=summary(modp1)
```

Supuestos

```
res1=summ$residuals
fit1=fitted(modp1)
dffp1=data.frame(res1,fit1)


ggplot(data = dffp, aes(x = fit1, y = res1)) +
  geom_point()
```




```
leveneTest(res1~BL*Nube*Solicitudes,data = base)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value                Pr(>F)
## group    11  118.66 < 0.00000000000000022 ***
##       13108
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
ks.test(res1,"pnorm")
```

```
## Warning in ks.test.default(res1, "pnorm"): ties should not be present for the
## Kolmogorov-Smirnov test

##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  res1
## D = 0.27265, p-value < 0.00000000000000022
## alternative hypothesis: two-sided
```

```r
residuos1=as.data.frame(res1)
res.stnd1 = scale(residuos1$res1) # <- Residuos estandarizados.
ggplot(data = residuos1, mapping = aes(sample = res.stnd1 )) + stat_qq_point() +
      ylab("Residuos Estandarizados") + xlab("Norm Quantiles") + ggtitle("Normal QQ Plot")  +
```

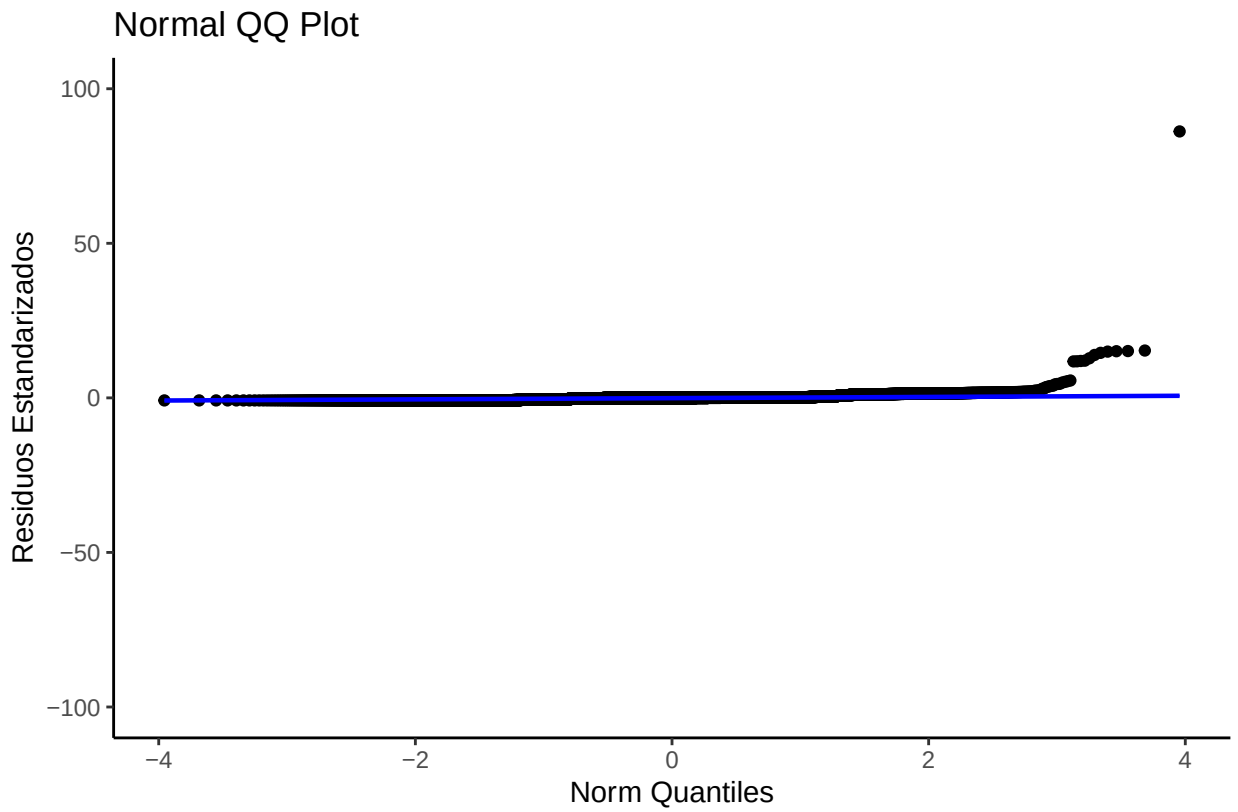


Grafico 1

Observamos que la normalidad se arregla un poco, pero la homocedasticidad sigue dando problemas. Puede que los valores extemos sigan afectando.

En este caso hay un punto en especifico que causa problemas, se va a cambiar por el promedio del tratamiento

```r
which(res1>75);which(res.stnd1>75) ## Se trata del mismo
```

```
## 1465
## 1465
```

```
## [1] 1465
```

```
base[1465,]
```

```
## # A tibble: 1 x 9
##   Tiempo BL    Nube  Denied Right Solicitudes    ID Bloque TiempoT
##    <dbl> <fct> <fct> <chr>  <chr> <fct>       <dbl> <chr>    <dbl>
## 1 0.0940 BL    Nube  Denied Right 500          1465 N5002     9.40
```

```
0.09379336  ## Es el promedio del tratamiento que es BL,Nube y 500 y Tarda 9!
```

```
## [1] 0.09379336
```

```
baseB=base
baseB[1465,9]=0.09379336
```

Y podemos volver a hacer todo a ver como nos da.

```
modp1=glmer(TiempoT~BL*Nube*Solicitudes+(1|Bloque),family = Gamma(link = "log"),data = baseB)
```

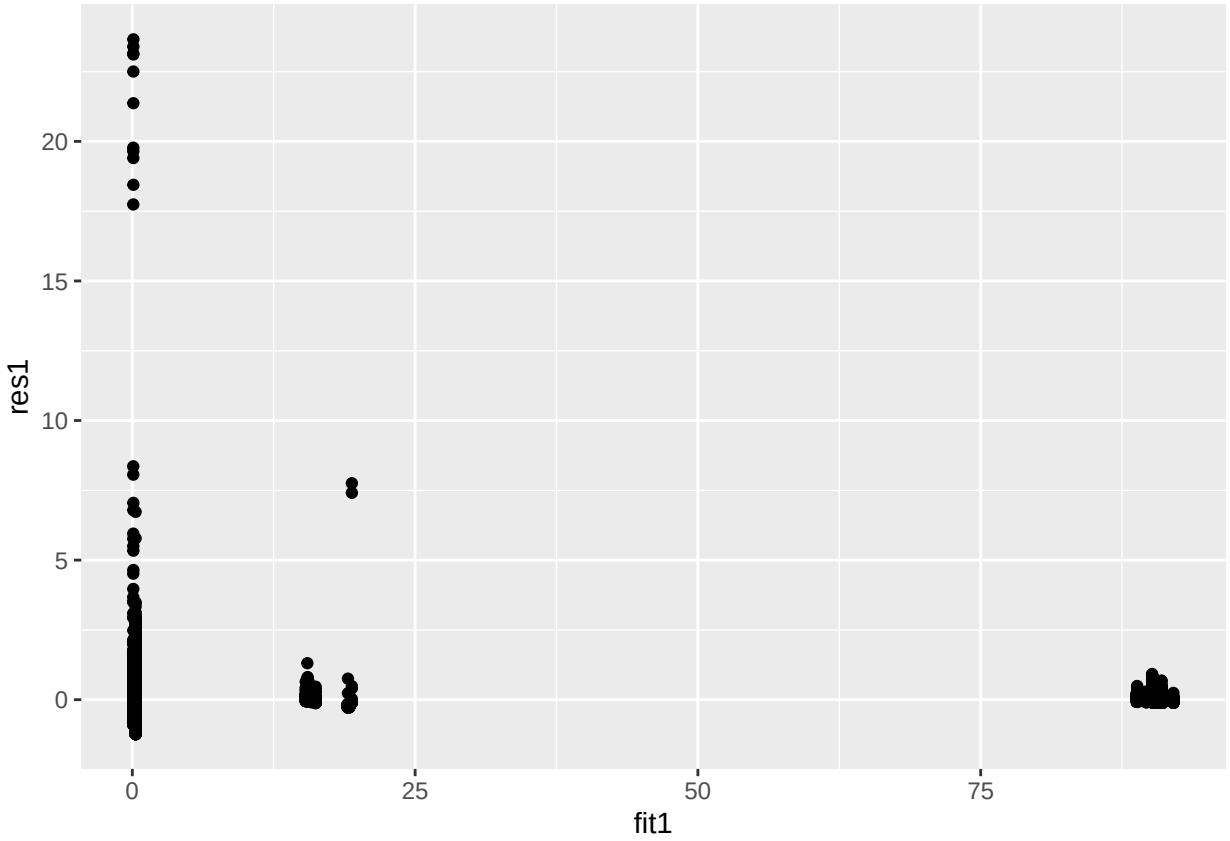
```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.00309844 (tol = 0.002, component 1)
```

```
summ=summary(modp1)

# Homocedasticidad

res1=summ$residuals
fit1=fitted(modp1)
dffp1=data.frame(res1,fit1)


ggplot(data = dffp, aes(x = fit1, y = res1)) +
  geom_point()
```




```
leveneTest(res1~BL*Nube*Solicitudes,data = base)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value                Pr(>F)
## group    11  299.17 < 0.00000000000000022 ***
##       13108
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Normalidad

ks.test(res1,"pnorm")
```

```
## Warning in ks.test.default(res1, "pnorm"): ties should not be present for the
## Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  res1
## D = 0.24686, p-value < 0.00000000000000022
## alternative hypothesis: two-sided
```

```
residuos1=as.data.frame(res1)
res.stnd1 = scale(residuos1$res1) # <- Residuos estandarizados.
ggplot(data = residuos1, mapping = aes(sample = res.stnd1 )) + stat_qq_point() +
      ylab("Residuos Estandarizados") + xlab("Norm Quantiles") + ggtitle("Normal QQ Plot")  +
```

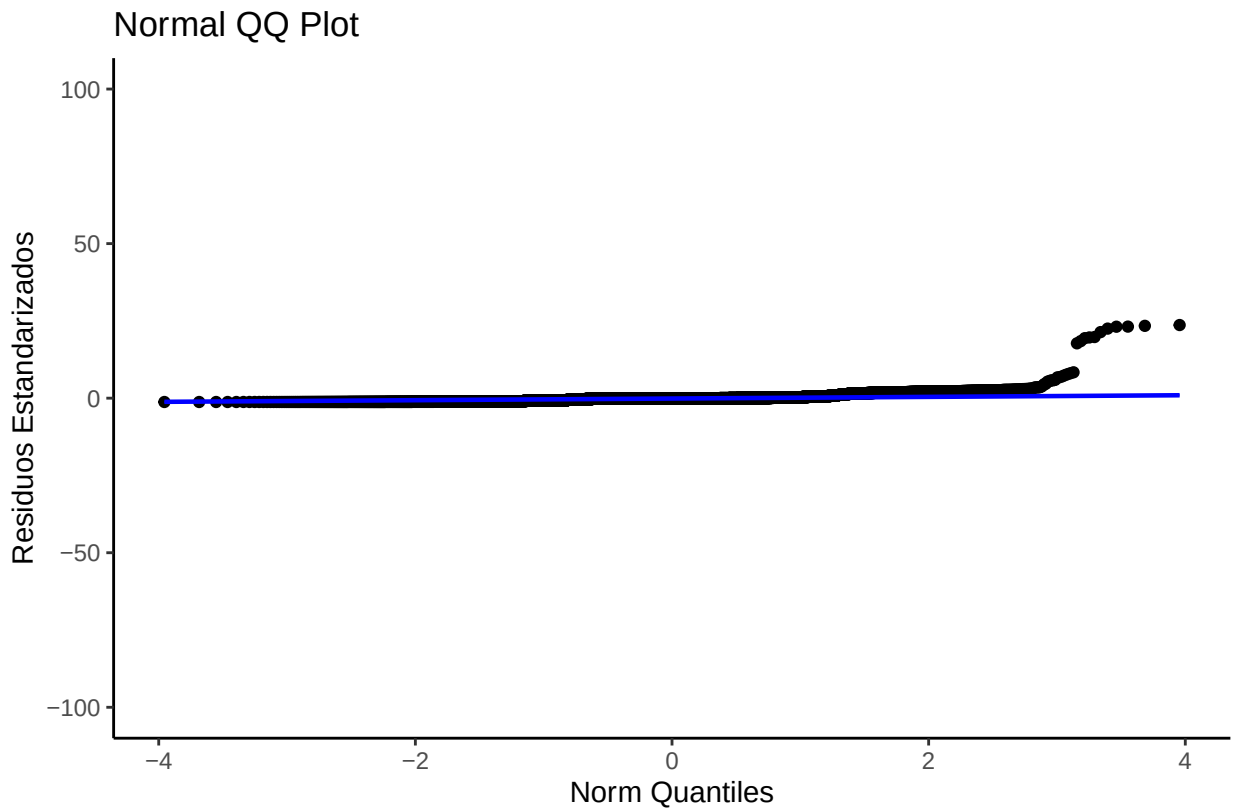


Grafico 1

Protocolo UDP Protocolo TCP