Example regression analysis: Party preference and ideological orientation

Susumu Shikano

Last compiled at 18. Juli 2022

Substantive motivation

You are interested in what factor determines citizens' evaluation of political parties. More specifically, you are interested in citizens' like/dislike of parties. You are speculating ideological proximity/distance behind this kind of evaluation. That is, a citizen likes a party whose ideological position is closer to their positions than another party with a more distant position.

To test your idea, you conducted online surveys with students visiting a lecture. To obtain the most important variables, that is, respondents' like/dislike and proximity to different parties, you asked the following questions:

Nun möchten wir Ihnen eine Frage über politische Parteien stellen.

Was halten Sie ganz allgemein von den folgenden einzelnen Parteien?

Benutzen Sie dafür bitte die untenstehende Skala von +5 bis -5 mit einem Nullpunkt in der Mitte. +5 bedeutet, dass Sie sehr viel von der Partei halten; -5 bedeutet, dass Sie überhaupt nichts von der Partei halten. Mit den Werten dazwischen können Sie Ihre Meinung abstufen.

Now we would like to ask you a question about political parties.

How much do you like or dislike each of the following parties?

Please use the scale below ranging from -5 to +5 with 0 indicating the middle. +5 means that you strongly like the party; -5 means that you strongly dislike the party. Choose the numbers in between to grade your statement.

	Ich halte überhaupt nichts von dieser Partei *I strongly dislike this party* -5	-4	-3	-2	-1	0	1	2	3	4	Ich halte sehr viel von dieser Partei *I strongly like this party* 5	Weiß nicht *Don't know*
CDU	○	○	○	○	○	○	○	○	○	○	○	○
CSU	○	○	○	○	○	○	○	○	○	○	○	○
SPD	○	○	○	○	○	○	○	○	○	○	○	○
Grüne	○	○	○	○	○	○	○	○	○	○	○	○
FDP	○	○	○	○	○	○	○	○	○	○	○	○
DIE LINKE	○	○	○	○	○	○	○	○	○	○	○	○
AfD	○	○	○	○	○	○	○	○	○	○	○	○

In der Politik reden die Leute häufig von "links" und "rechts". Wenn Sie diese Skala von 1 bis 11 benutzen, wo würden Sie die folgenden Parteien einordnen, wenn 1 "links" und 11 "rechts" ist?

In politics people sometimes talk of left and right. Where would you place the following parties on the scale if 1 is "left" and 11 is "right"?

	links *left* 1	2	3	4	5	6	7	8	9	10	rechts *right* 11	Weiß nicht *Don't know*
CDU	○	○	○	○	○	○	○	○	○	○	○	○
CSU	○	○	○	○	○	○	○	○	○	○	○	○
SPD	○	○	○	○	○	○	○	○	○	○	○	○
Grüne	○	○	○	○	○	○	○	○	○	○	○	○
FDP	○	○	○	○	○	○	○	○	○	○	○	○
DIE LINKE	○	○	○	○	○	○	○	○	○	○	○	○
AfD	○	○	○	○	○	○	○	○	○	○	○	○

Und wie ist das mit Ihnen selbst? Wo würden Sie sich auf der Skala von 1 bis 11 einordnen, wenn 1 "links" und 11 "rechts" ist?

Thinking about yourself, where would you place yourself on the following scale, with 1 being "left" and 11 being "right"?

	links *left* 1	2	3	4	5	6	7	8	9	10	rechts *right* 11	Weiß nicht *Don't know*
	○	○	○	○	○	○	○	○	○	○	○	○

We can load the dataset and check its content:

```
# Loading the data file
load("data/Lec_Statistics_Ideology_Data.RData")

# Check which object is loaded
ls()
```

```
## [1] "ci.sample.mean" "long.data"      "naive.var"
```

```
# Check the inside of the data
head(long.data,n=10)
```

```
##       id wave party skalo lr.self lr time wg
## 1  10131    3     1     3       1  1   63 NA
## 2  10132    3     2     3       1  2   27 NA
## 3  10133    3     3     2       1 -1   66 NA
## 4  10134    3     4    -2       1 -2  105 NA
## 5  10135    3     5     1       1  0   38 NA
## 6  10136    3     6    -2       1  3   88 NA
## 7  10137    3     7    -2       1 -4   28 NA
## 8  10231    3     1     0       0  1   73 NA
## 9  10232    3     2     0       0  1   19 NA
## 10 10233    3     3     0       0 -1   53 NA
```

The data contains the following variables.

- id: Identification number of responses

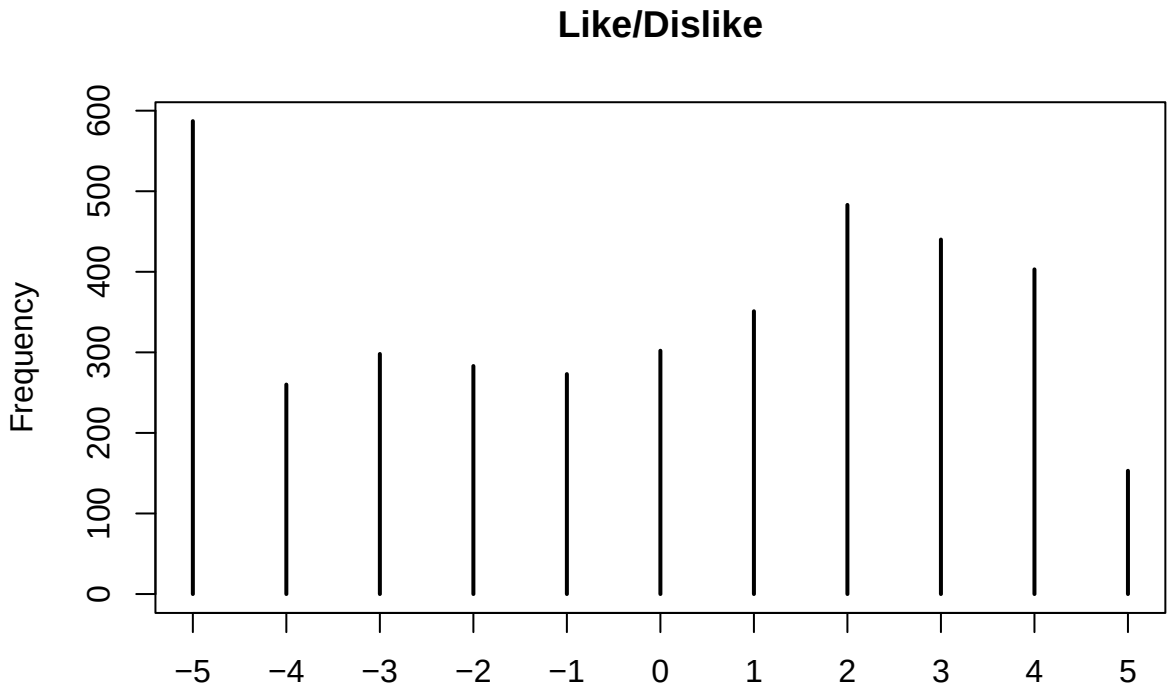
- wave: survey wave
- party: Which party is at stake?
  - 1: CDU; 2: CSU; 3: SPD; 4: Greens; 5: FDP; 6: AfD; 7: Linke
- skalo: Respondents' like/dislike of a party (+5 like/-5 dislike)
- lr.self: Respondents' self placement on the left-right ideology scale (+5 right most/-5 left most)
- lr: Respondents' placement of the party on the left-right ideology scale (+5 right most/-5 left most)
- time: Response time for the whole skalometer question page (in seconds)

In the dataset, we have 3833 observations. Note that each observation corresponds to a response. Since a respondent can give multiple responses for different parties, s/he can appear many times in this dataset.

Univariate discriptive statistics

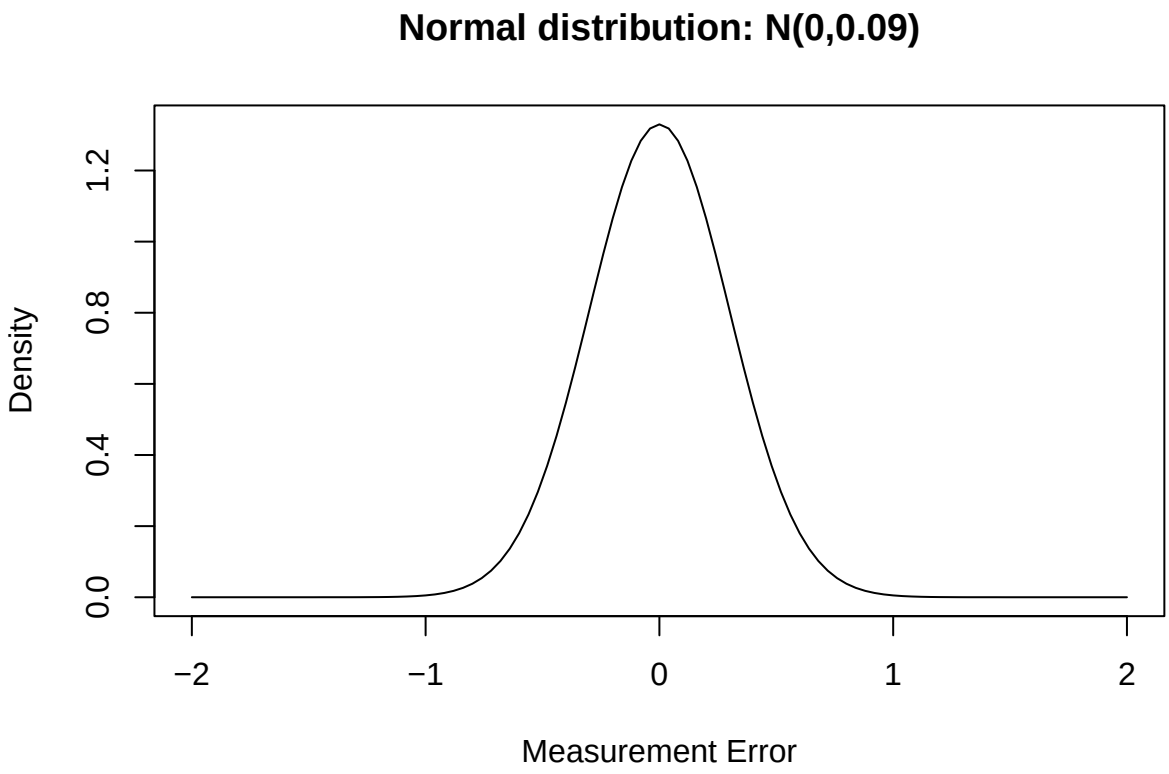
We can start with the respondents' like/dislike of parties. Below its distribution:

```
plot(table(long.data$skalo),type="h",main="Like/Dislike",ylab="Frequency")
```



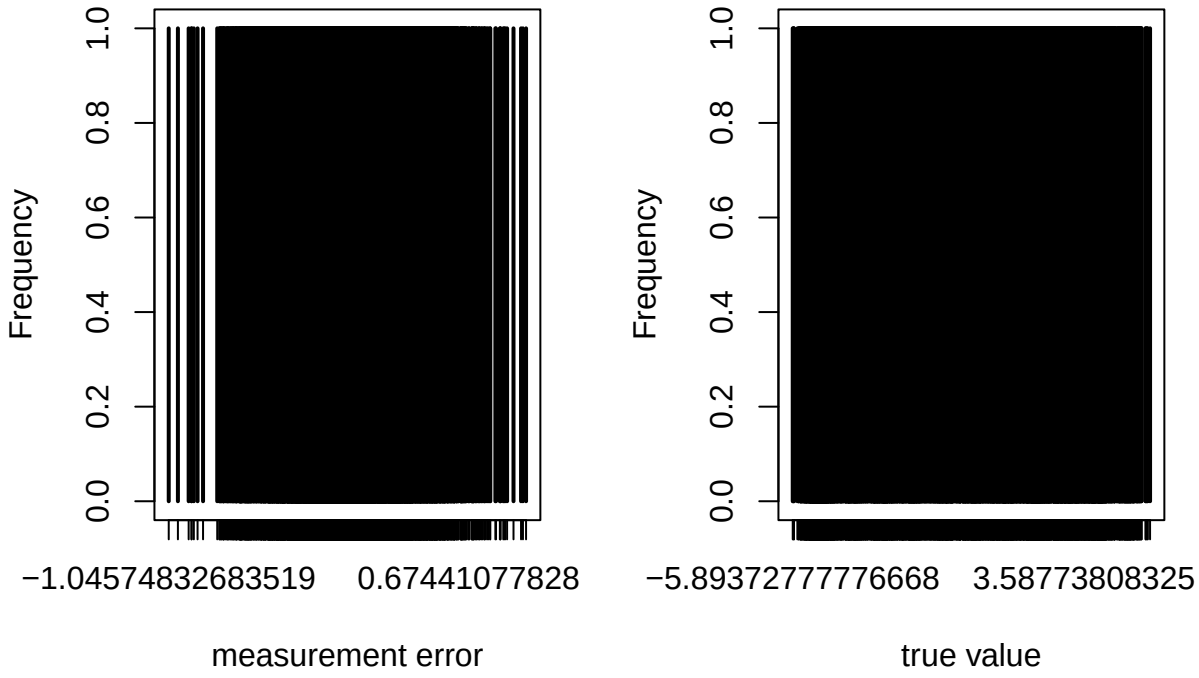
Assume that this measurement contains random errors which follow a normal distribution $N(0, 0.09)$:

```
error.dist.density <- function(x){
  dnorm(x,mean=0,sd=sqrt(0.09))
}
plot(error.dist.density,xlim=c(-2,2),
     main="Normal distribution: N(0,0.09)",
     ylab="Density",
     xlab="Measurement Error")
```



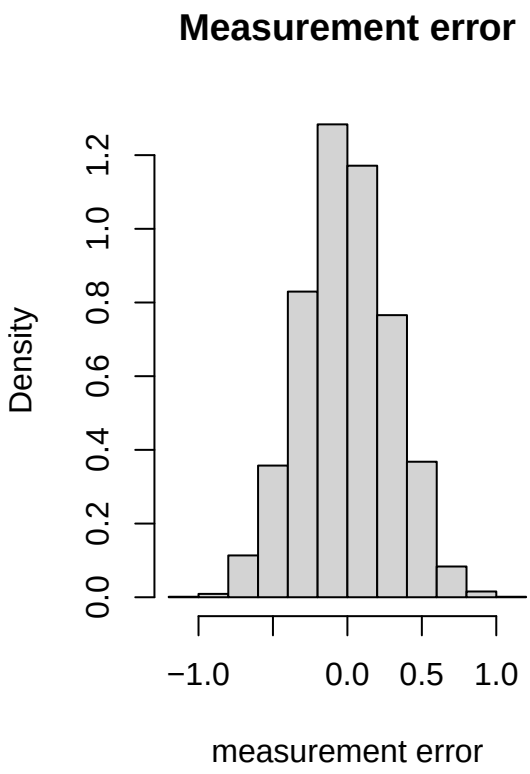
For each of 2504 observations, we generate random errors and compute the true value behind the measurement:

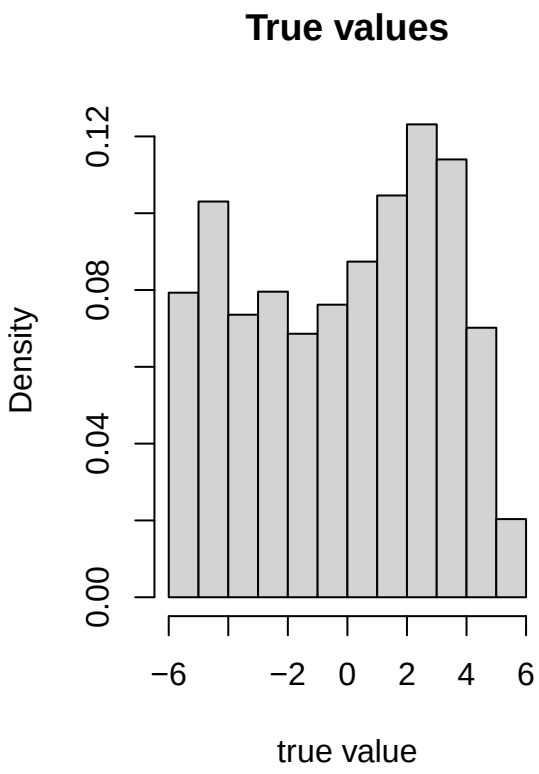
```
set.seed(1234567) # Setting a seed so that the "random" errors can be replicated.
measurement.error <- rnorm(nrow(long.data),mean=0,sd=sqrt(0.09))

long.data$skalo.true <- long.data$skalo - measurement.error

par(mfrow=c(1,2))

plot(table(measurement.error),type="h",xlab="measurement error",ylab="Frequency")
plot(table(long.data$skalo.true),type="h",xlab="true value",ylab="Frequency")
```



This way of plot is not informative since both variables can take all real numbers and each value appears only once. That is, the measurement errors and true values here are continuous random variables. Instead of counting all possible values, we can group the values for certain ranges and plot their densities in histograms.

```
par(mfrow=c(1,2))

hist(measurement.error,xlab="measurement error",ylab="Density",freq=FALSE,
     main="Measurement error")
hist(long.data$skalo.true,xlab="true value",ylab="Density",freq=FALSE,
     main="True values")
```





We can compute some descriptive statistics of the true value. More specifically, we can obtain mean, variance and standard deviation.

```
print("The number of valid observations")
```

```
## [1] "The number of valid observations"
```

```
print(valid.n.skalo <- length(long.data$skalo.true))
```

```
## [1] 3833
```

```
print("The mean response")
```

```
## [1] "The mean response"
```

```
print(mean.skalo <- sum(long.data$skalo.true)/valid.n.skalo)
```

```
## [1] -0.172782
```

```
print("Variance of the responses")
```

```
## [1] "Variance of the responses"
```

```
print(var.skalo <- sum((long.data$skalo.true - mean.skalo)^2)/valid.n.skalo )
```

```
## [1] 10.36044
```

```
print("Standard deviation of the responses")
```

```
## [1] "Standard deviation of the responses"
```

```
print(sd.skalo <- sqrt(var.skalo))
```

```
## [1] 3.218764
```

The above procedure is a bit tedious since you are programming hand-rolled. Fortunately, there are ready-made built-in functions in R:

```
length(long.data$skalo.true)
```

```
## [1] 3833
```

```
mean(long.data$skalo.true)
```

```
## [1] -0.172782
```

```
var(long.data$skalo.true)
```

```
## [1] 10.36315
```

```
sd(long.data$skalo.true)
```

```
## [1] 3.219184
```

If you compare the results with those above, however, there are slight differences in variance and standard deviation. This is because `var` and `sd` functions compute the unbiased estimates. That is, you divide not by the number of observations (3833), but $n-1$ (3832) to obtain the variance.

To obtain just a variance (not an estimate), you should correct as follows:

```
var(long.data$skalo.true)*(valid.n.skalo-1)/valid.n.skalo
```

```
## [1] 10.36044
```

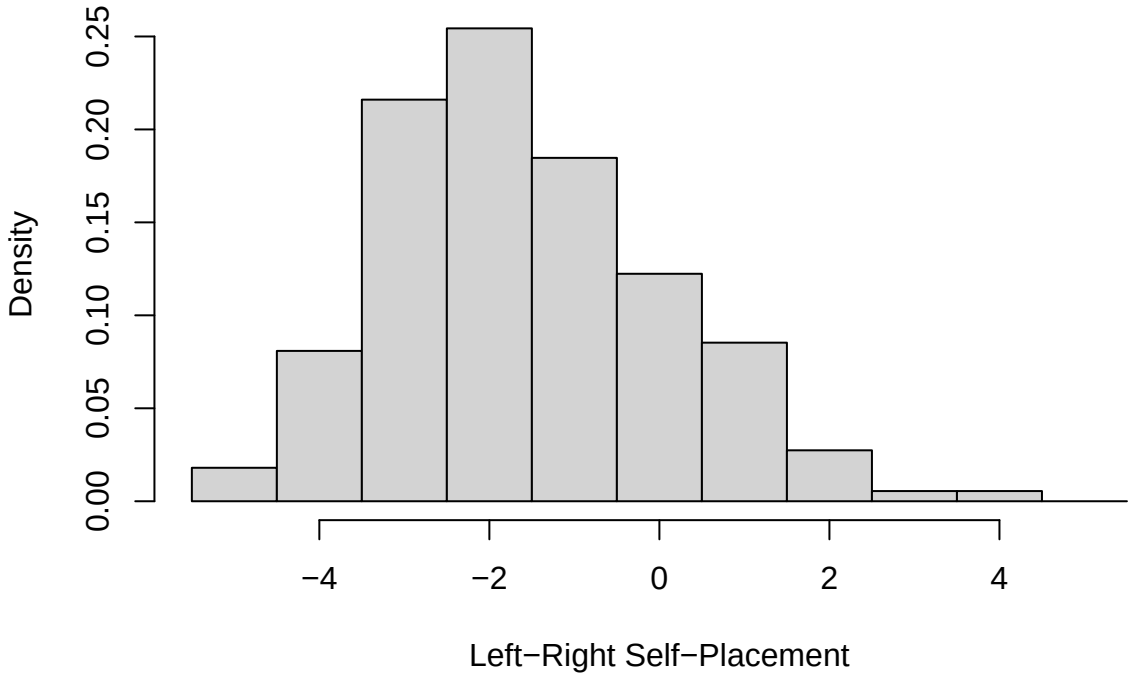
In R, one can also define your own function for the naive variance:

```
naive.var <- function(x){
  diff.to.mean <- x - mean(x)
  squared.diff <- diff.to.mean^2
  output <- mean(squared.diff)
  output
}

naive.var(long.data$skalo.true)
```

```
## [1] 10.36044
```

Below, we can also observe further variables.

```
hist(long.data$lr.self,br=seq(-5.5,5.5,by=1),main="",
     xlab="Left-Right Self-Placement",
     freq=FALSE)
```



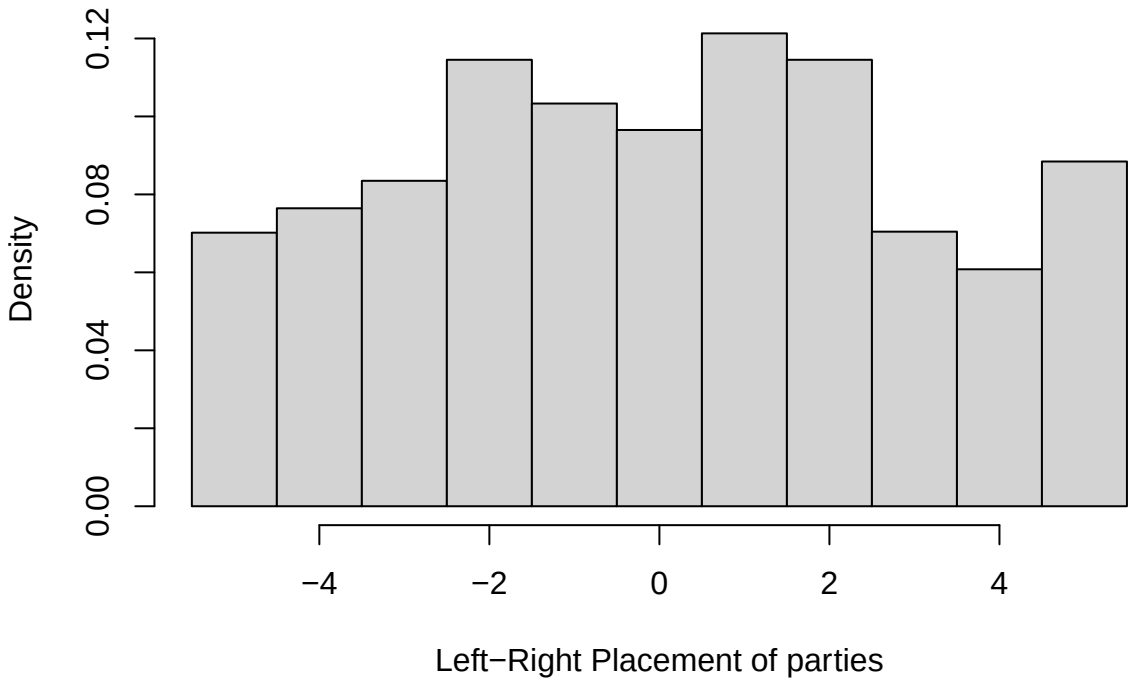

```
mean(long.data$lr.self)
```

```
## [1] -1.576572
```

```
naive.var(long.data$lr.self)
```

```
## [1] 2.736701
```

```
hist(long.data$lr,br=seq(-5.5,5.5,by=1),
     xlab="Left-Right Placement of parties",main="",
     freq=FALSE)
```




```
mean(long.data$lr)
```

```
## [1] 0.007565875
```

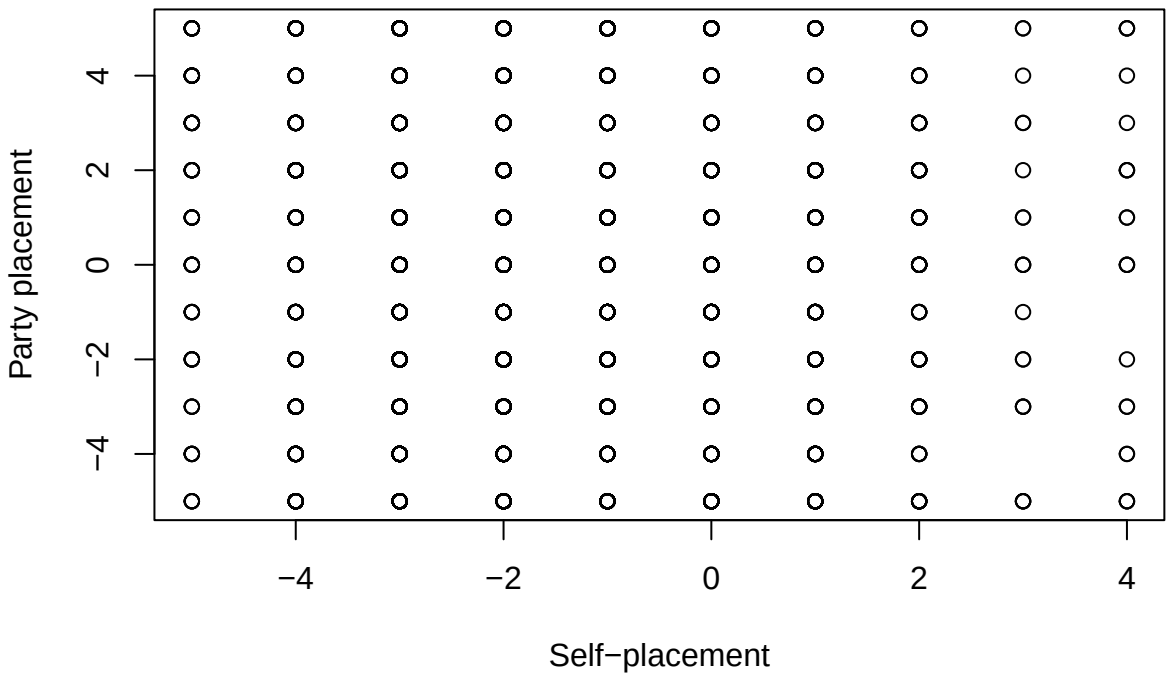
```
naive.var(long.data$lr)
```

```
## [1] 8.687394
```

Bivariate relationship between the party placement and self-placement

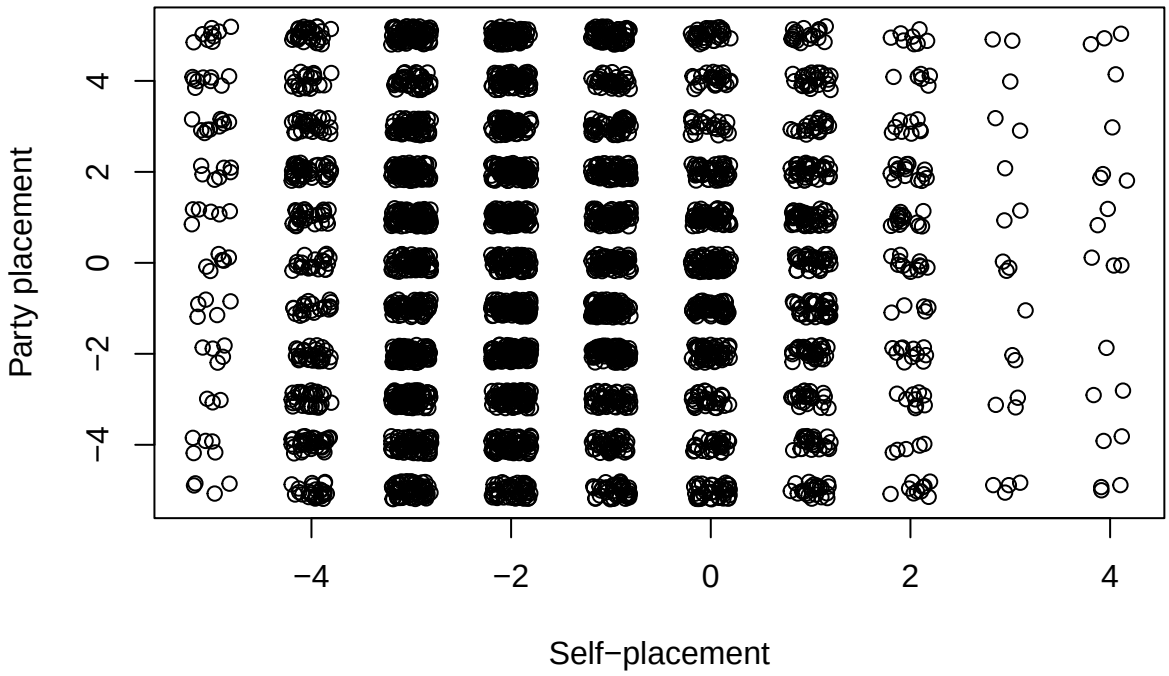
Now we can inspect the joint distribution of the the left-right party-placement and self-placement. We can just plot the distribution:

```
plot(long.data$lr ~ long.data$lr.self,
     xlab="Self-placement",ylab="Party placement")
```



This is however not so informative since both variables are discrete and many responses have the same value combination. Therefore, we cannot see here how many responses are behind different value combinations. Here, we can just add small random values to both measures and plot them again:

```
plot(jitter(long.data$lr) ~ jitter(long.data$lr.self),
     xlab="Self-placement",ylab="Party placement")
```



The relationship of both variables is not clear to see. We can now calculate two summary statistics: covariance and correlation:

```
cov(long.data$lr , long.data$lr.self)
```

```
## [1] -0.06374724
```

```
cor(long.data$lr , long.data$lr.self)
```

```
## [1] -0.01307042
```

Here again, we have to be careful that `cov` gives the unbiased estimate. For simple covariance, you need to write your own function:

```
naive.cov <- function(x,y){
  mean((x - mean(x))*(y - mean(y)))
}

naive.cov(long.data$lr , long.data$lr.self)
```

```
## [1] -0.06373061
```

Creating a new variable

We can now create new variables based on the existing variables:

- Difference between the party placement and self-placement
- Absolute value of the above difference
- Squared value of the above difference

```
long.data$lr.dist.dif <- long.data$lr - long.data$lr.self
long.data$lr.dist.abs <- abs(long.data$lr.dist.dif)
long.data$lr.dist.sqr <- long.data$lr.dist.dif^2
```

```
mean(long.data$lr.dist.dif)
```

```
## [1] 1.584138
```

```
naive.var(long.data$lr.dist.dif)
```

```
## [1] 11.55156
```

These values can be also obtained from the statistics of the variables, based on which the variable was created:

```
mean(long.data$lr) - mean(long.data$lr.self)
```

```
## [1] 1.584138
```

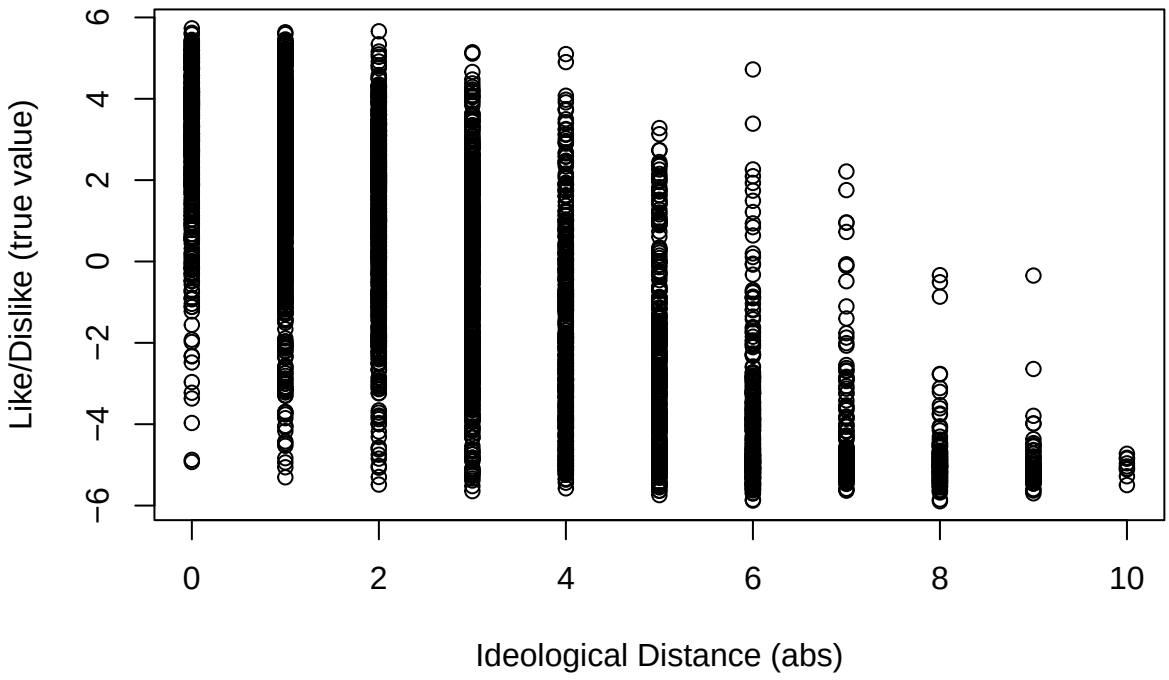
```
naive.var(long.data$lr) + naive.var(long.data$lr.self) - 2*naive.cov(long.data$lr,long.data$lr.self)
```

```
## [1] 11.55156
```

Bivariate relationship between interested variables

Now we can inspect the joint distribution of the like/dislike variable and the absolute distances. We can just again plot the distribution with small random numbers:

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Ideological Distance (abs)",
     ylab="Like/Dislike (true value)")
```




```
naive.cov(long.data$skalo.true, long.data$lr.dist.abs)
```

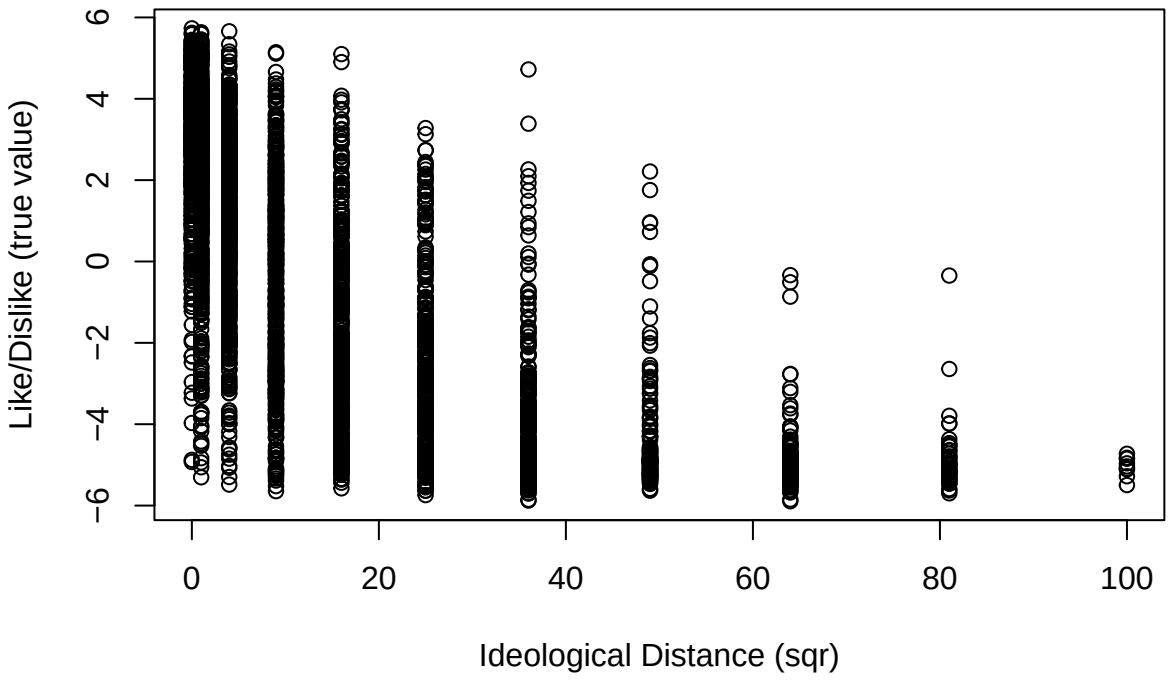
```
## [1] -5.560991
```

```
cor(long.data$skalo.true, long.data$lr.dist.abs)
```

```
## [1] -0.7556228
```

Now, the same exercise for the squared distances:

```
plot(long.data$skalo.true ~ long.data$lr.dist.sqr,
     xlab="Ideological Distance (sqr)",
     ylab="Like/Dislike (true value)")
```




```
naive.cov(long.data$skalo, long.data$lr.dist.sqr)
```

```
## [1] -39.89033
```

```
cor(long.data$skalo, long.data$lr.dist.sqr)
```

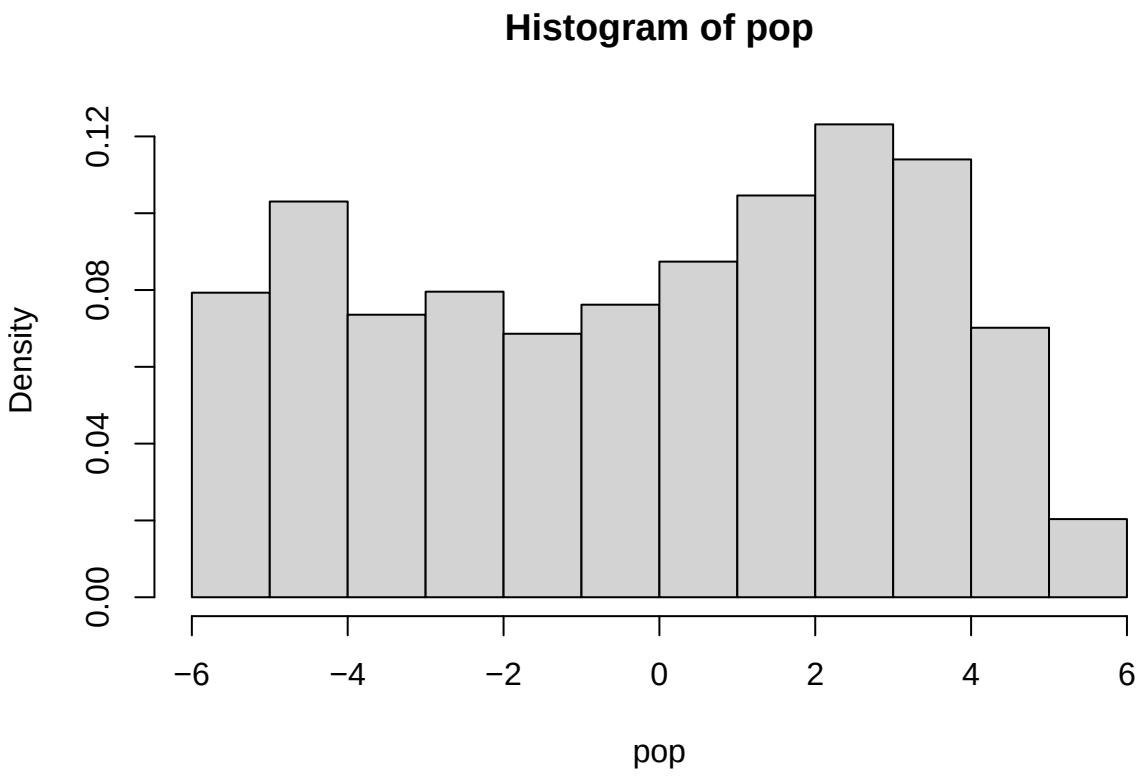
```
## [1] -0.6906155
```

Central limit theorem

In this section, we learn the central limit theorem by using the generated true response to party evaluation. We treat this variable as population, from which we draw multiple samples. Our goal here is to estimate the population mean.

First, we can check the population distribution:

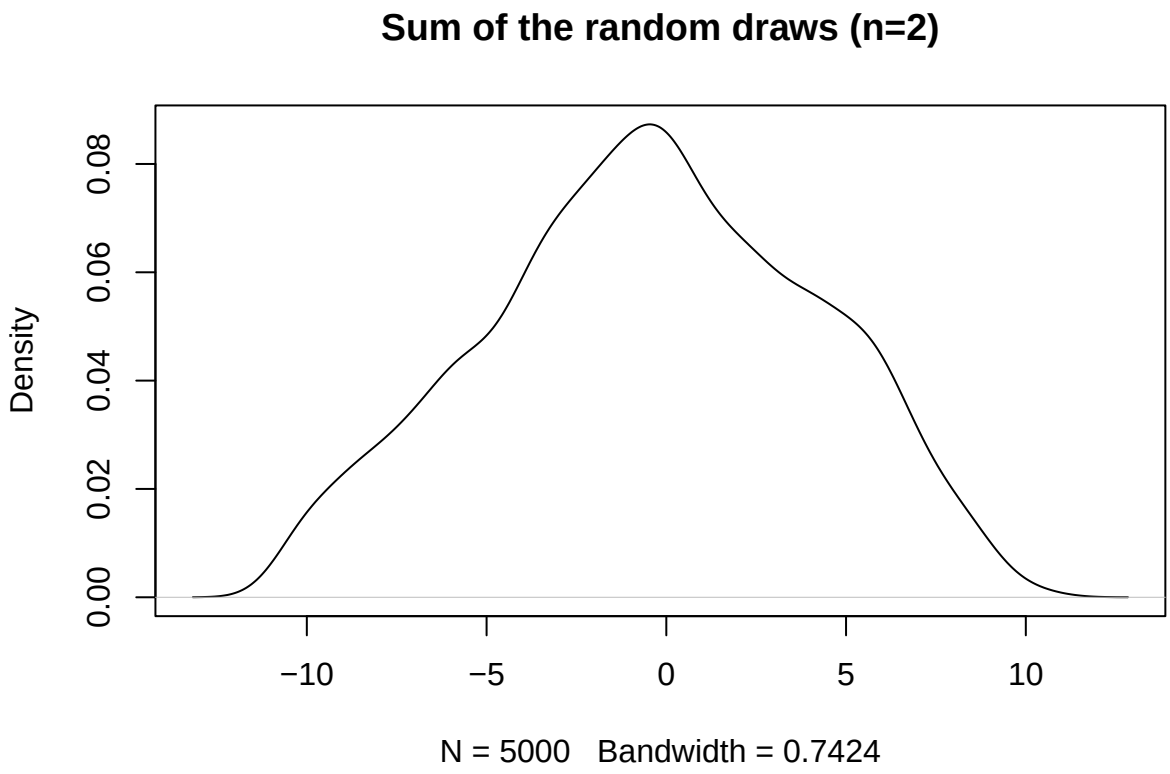
```
pop <- long.data$skalo.true
pop.mean <- mean(pop)
pop.var <- mean((pop-pop.mean)^2)

hist(pop,freq=F)
```



The distribution is multi-modal and skewed. The population mean is -0.173 and the population variance is 10.36‘

From this population, we can draw multiple samples with size of 2, calculate the sample sum and observe its distribution.

```
n.iter <- 5000
sample.size <- 2
all.sample.sum <- rep(NA,n.iter)
for (i in 1:n.iter){
  this.sample <- sample(pop,size=sample.size,replace=TRUE)
  this.sample.sum <- sum(this.sample)
  all.sample.sum[i] <- this.sample.sum
}

plot(density(all.sample.sum),
     main=paste0("Sum of the random draws (n=",sample.size,")"))
```




```
mean(all.sample.sum)
```

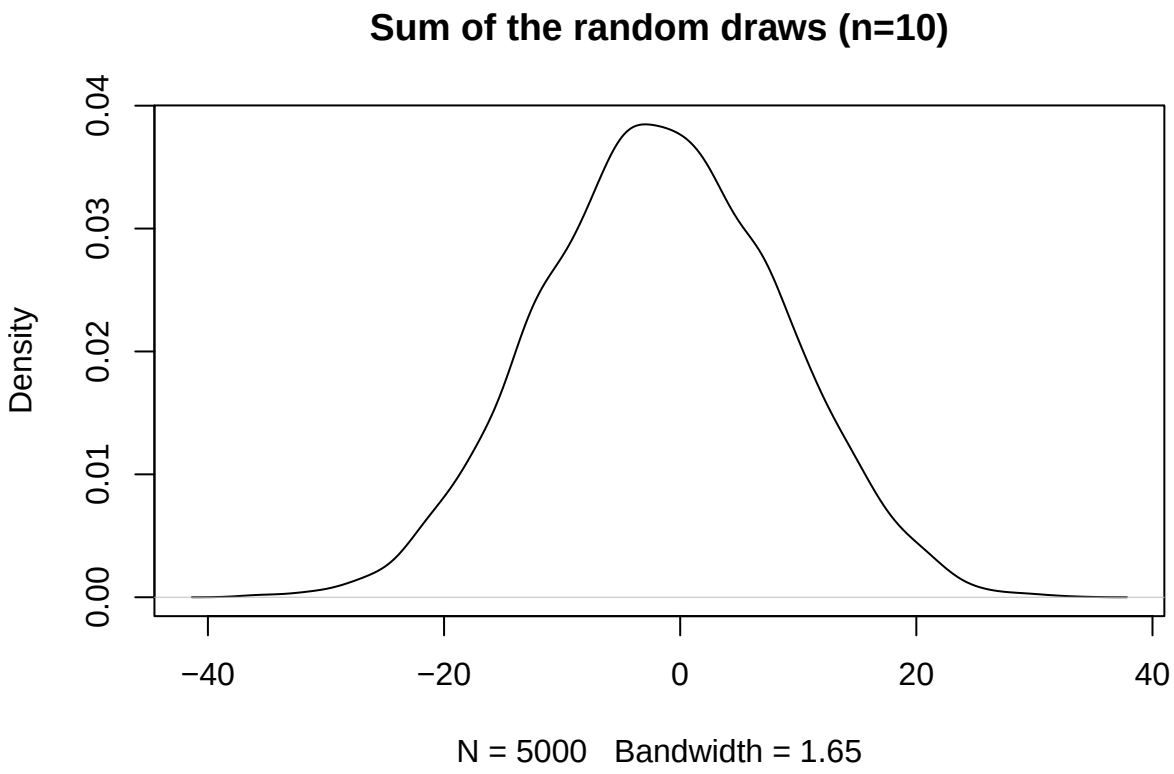
```
## [1] -0.4438867
```

```
var(all.sample.sum)
```

```
## [1] 20.53005
```

From this population, we can draw multiple samples with size of 10, calculate the sample sum and observe its distribution.

```
sample.size <- 10
all.sample.sum <- rep(NA,n.iter)
for (i in 1:n.iter){
  this.sample <- sample(pop,size=sample.size,replace=TRUE)
  this.sample.sum <- sum(this.sample)
  all.sample.sum[i] <- this.sample.sum
}

plot(density(all.sample.sum),
     main=paste0("Sum of the random draws (n=",sample.size,")"))
```




```
mean(all.sample.sum)
```

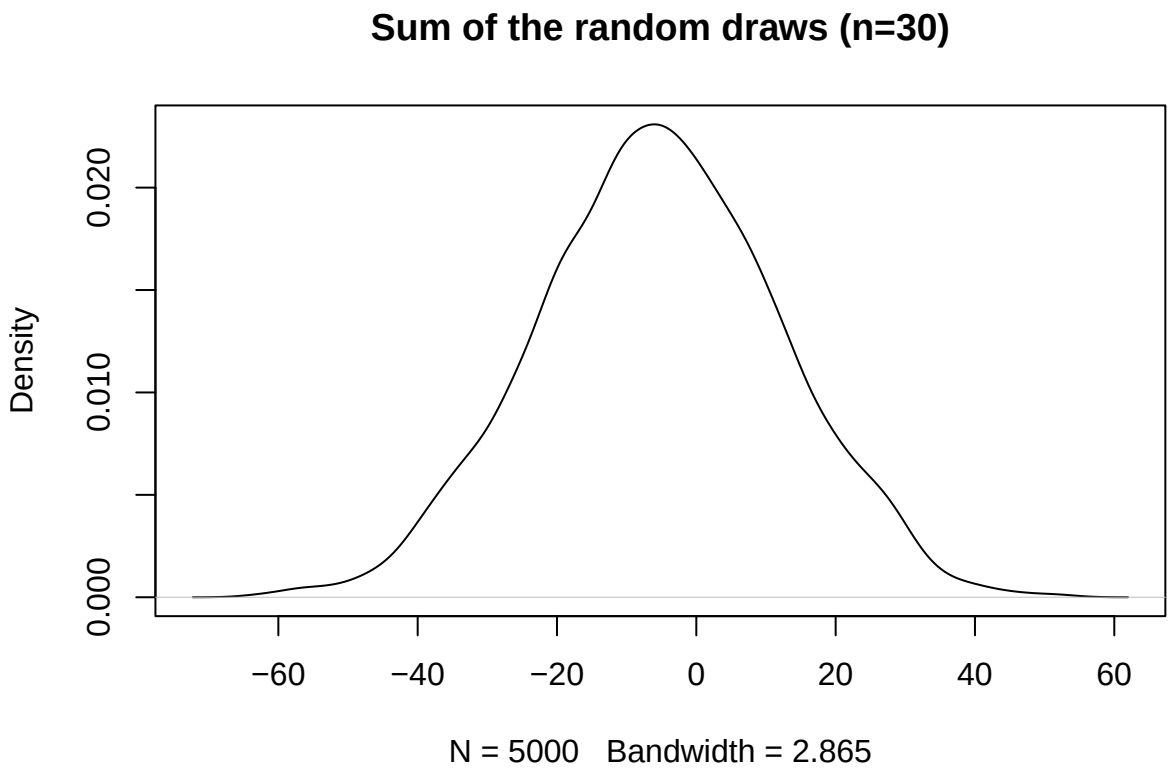
```
## [1] -1.643374
```

```
var(all.sample.sum)
```

```
## [1] 101.3953
```

From this population, we can draw multiple samples with size of 30, calculate the sample sum and observe its distribution.

```
sample.size <- 30
all.sample.sum <- rep(NA,n.iter)
for (i in 1:n.iter){
  this.sample <- sample(pop,size=sample.size,replace=TRUE)
  this.sample.sum <- sum(this.sample)
  all.sample.sum[i] <- this.sample.sum
}

plot(density(all.sample.sum),
     main=paste0("Sum of the random draws (n=",sample.size,")"))
```




```
mean(all.sample.sum)
```

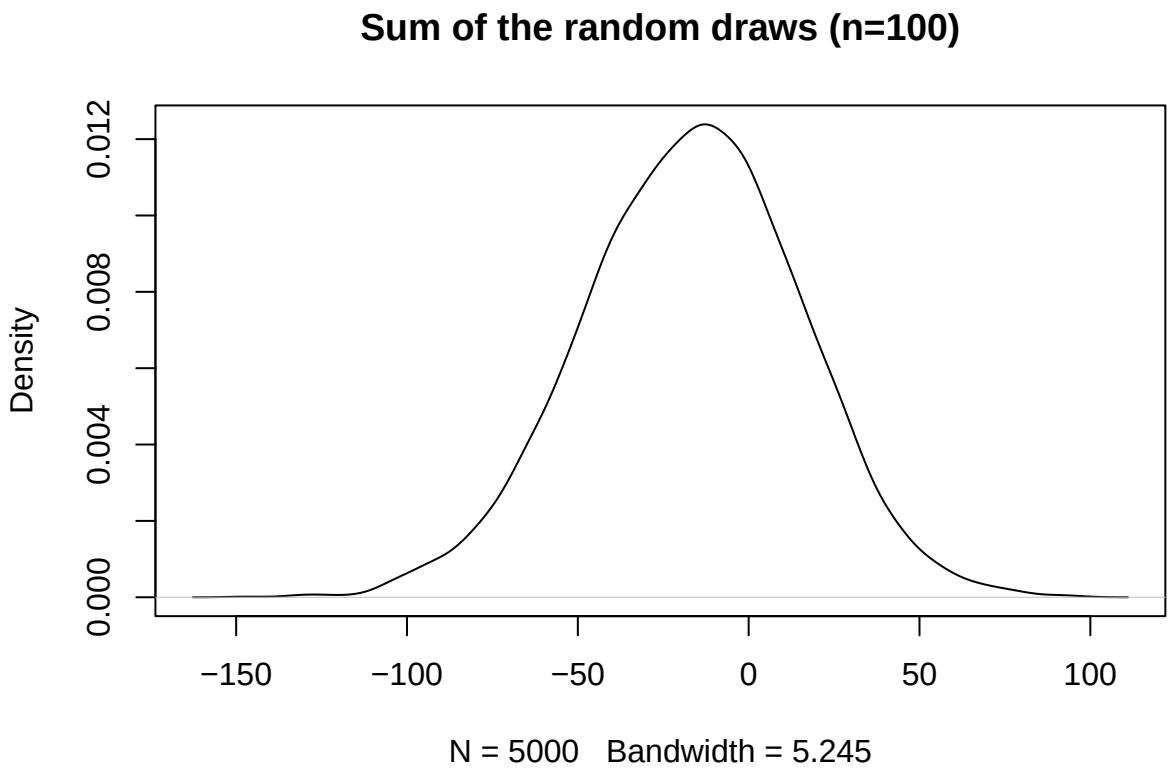
```
## [1] -5.401836
```

```
var(all.sample.sum)
```

```
## [1] 305.6596
```

From the population, we can draw multiple samples with size of 100, calculate the sample sum and observe its distribution.

```
sample.size <- 100
all.sample.sum <- rep(NA,n.iter)
for (i in 1:n.iter){
  this.sample <- sample(pop,size=sample.size,replace = TRUE)
  this.sample.sum <- sum(this.sample)
  all.sample.sum[i] <- this.sample.sum
}

plot(density(all.sample.sum),
     main=paste0("Sum of the random draws (n=",sample.size,")"))
```




```
mean(all.sample.sum)
```

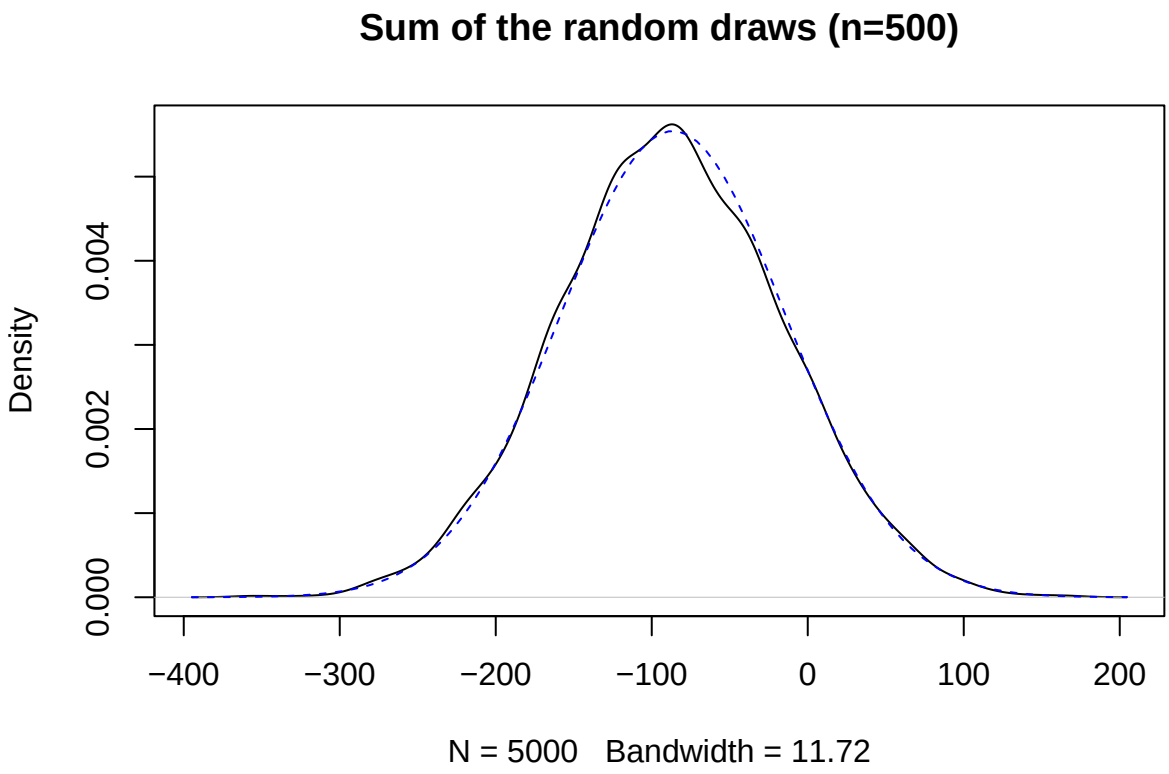
```
## [1] -16.79455
```

```
var(all.sample.sum)
```

```
## [1] 1024.509
```

From the population, we can draw multiple samples with size of 500, calculate the sample sum and observe its distribution.

```
sample.size <- 500
all.sample.sum <- rep(NA,n.iter)
for (i in 1:n.iter){
  this.sample <- sample(pop,size=sample.size,replace=TRUE)
  this.sample.sum <- sum(this.sample)
  all.sample.sum[i] <- this.sample.sum
}

plot(density(all.sample.sum),
     main=paste0("Sum of the random draws (n=",sample.size,")"))
par(new=T)
this.norm <- function(x) dnorm(x,
                               mean=pop.mean*sample.size,
                               sd=sqrt(sample.size*pop.var))
curve(this.norm,add=T,col="blue",lty=2)
```




```
mean(all.sample.sum)
```

```
## [1] -87.85455
```

```
var(all.sample.sum)
```

```
## [1] 5151.608
```

Above, with increasing number of observations, the distribution of the sample sum becomes closer to a normal distribution. If we draw an infinitely large number of samples with a certain size, the distribution converges to a normal distribution whose mean is identical with the population mean times sample size and variance with the population variance times the sample size:

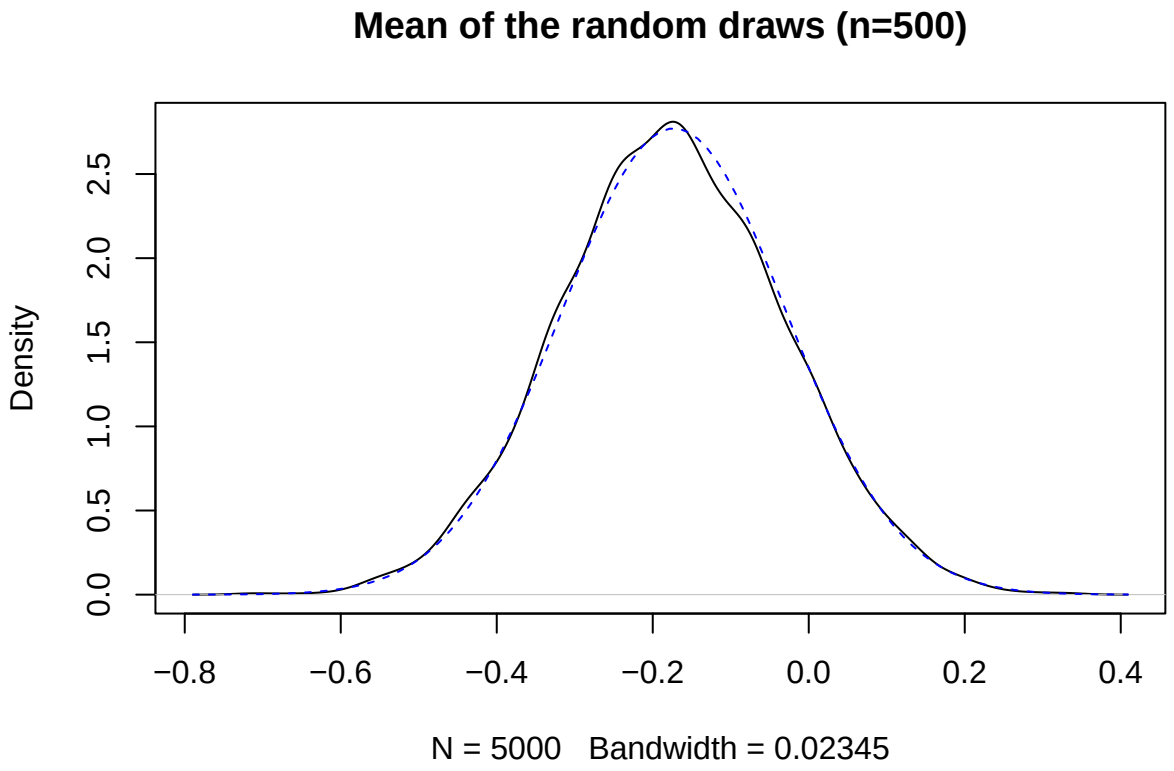
$$\sum x \sim N(n\mu, n\sigma^2)$$

. If we divide the mean and variance of the above distribution (-87.855 and 5150.578) by n (500), we obtain -0.176 and 10.301. These are very close to the population mean and variance (-0.173 and 10.36)

If you divide the sample sum by the sample size, you will obtain the sample mean. In the above figure, we always had the same sample size (n=500), therefore the sample mean also has the same form of the distribution, which also converges to a normal distribution.

```
all.sample.mean <- all.sample.sum/sample.size
plot(density(all.sample.mean),
     main=paste0("Mean of the random draws (n=",sample.size,")"))
par(new=T)
this.norm <- function(x) dnorm(x,
                               mean=pop.mean,
```

```
                                sd=sqrt(pop.var/sample.size))
curve(this.norm,add=T,col="blue",lty=2)
```



The mean and variance of this distribution are -0.176 and 0.021. This should follow the following:

$$\bar{x} \sim N(\mu, \sigma^2/n)$$

.

Regressing like/dislike of parties on left-right ideological distance and log response time.

We first recap the simple regression model, which we estimated above:

```
summary(lm.out.abs <- lm((long.data$skalo.true ~ long.data$lr.dist.abs)))
```

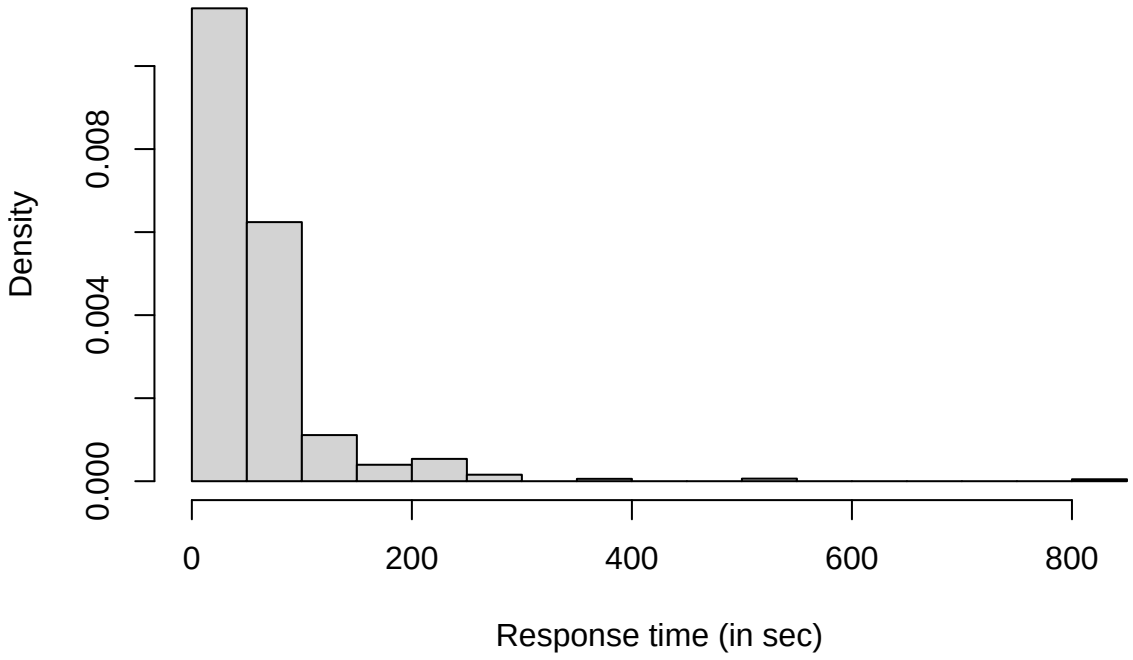
```
##
## Call:
## lm(formula = (long.data$skalo.true ~ long.data$lr.dist.abs))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9169 -1.4955  0.1624  1.4471  8.1145
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            2.98875    0.05586    53.5   <2e-16 ***
```

```
## long.data$lr.dist.abs -1.06374    0.01490   -71.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.109 on 3831 degrees of freedom
## Multiple R-squared:  0.571,  Adjusted R-squared:  0.5709
## F-statistic:  5098 on 1 and 3831 DF,  p-value: < 2.2e-16
```

You think that the time to respond to the questions can affect party evaluation, as well. That is, those who quickly answer may differ from those who deliberate in answering.

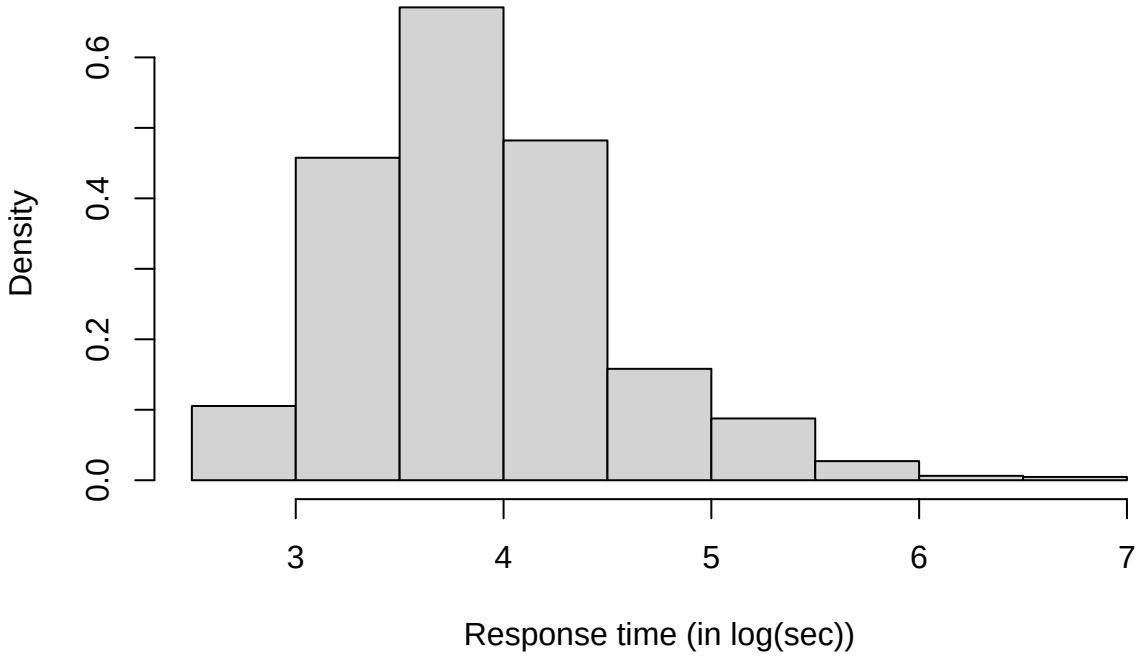
First, you can check the distribution of the time:

```
hist(long.data$time,
     main="",xlab="Response time (in sec)",
     freq=FALSE)
```



You see some outliers who needed much longer time to respond the questions than the others. To reduce the impact of such outliers, you create a new variably by logarithmizing the original variable:

```
long.data$log.time <- log(long.data$time)

hist(long.data$log.time,
     main="",xlab="Response time (in log(sec))",
     freq=FALSE)
```



Now you include the log response time into the regression analysis:

```
summary(lm.out.abs.time <-
          lm(skalo.true ~ lr.dist.abs + log.time,
             data=long.data))
```

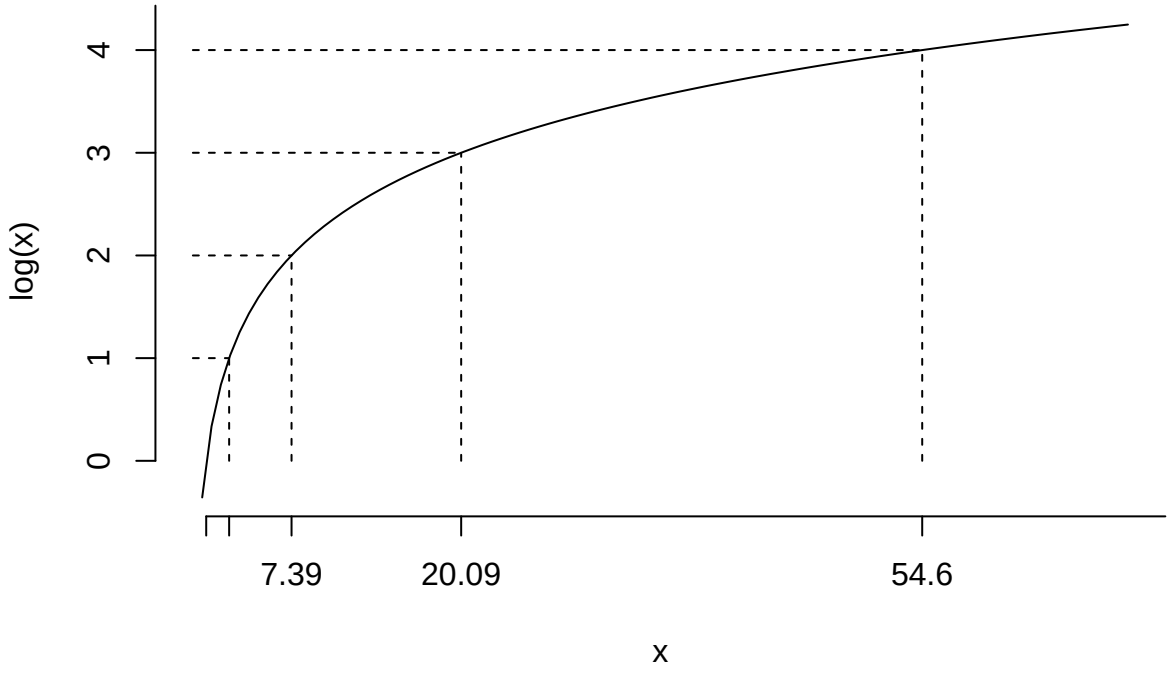
```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + log.time, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9003 -1.5024  0.1592  1.4577  8.0360
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.61475    0.21828  11.979   <2e-16 ***
## lr.dist.abs -1.06416    0.01490 -71.442   <2e-16 ***
## log.time     0.09624    0.05430   1.772   0.0764 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.108 on 3830 degrees of freedom
## Multiple R-squared:  0.5713, Adjusted R-squared:  0.5711
## F-statistic:  2552 on 2 and 3830 DF,  p-value: < 2.2e-16
```

The new variable, log response time, has a positive effect. However, the effect is not significant at 5% level.

To interpret the slope coefficient of a logarithmized variable is tricky.According to the regression result above, unit increase of log response time is associated with increase of 0.096243. However, a unit increase of the log response time is not constant on the scale of the raw response time.

To see what a unit increase of the log response time means, we can first check the logarithm function.

```
curve(log,0,70,axes=F)
axis(1,at=exp(seq(0,5)),
     c("",round(exp(seq(1,5)),2))
     )
axis(2,at=seq(0,5))

for (i in 1:4){
  lines(c(0,exp(i)),rep(i,2),lty=2)
  lines(rep(exp(i),2),c(0,i),lty=2)
}
```



Note that the opposite function of log() is the exponential function (exp()). To solve e.g. log(x)=2 for x, we can calculate just exp(2)=7.3890561.

The above figure clearly demonstrates that increase from log(2) to log(3) and that from log(3) to log(4) correspond different increase in the raw response time. If you however take the growth, it is constant. That is:

- exp(3)/exp(2) = 20.09/7.39 = 2.7182818
- exp(4)/exp(3) = 54.6/20.09 = 2.7182818

That is, we can interpret the coefficient of the log response time (0.096243) as change in Y given X grows constantly. More specifically, given a response time is 1% longer, the evaluation will change with 0.096243/100.

Consequently, the result will not change if we take the log of response time in minutes instead of in seconds:

```
long.data$time.min <- long.data$time/60

long.data$log.time.min <- log(long.data$time.min)

head(long.data[,c("time","log.time","time.min","log.time.min")],n=10)
```

```
##    time log.time  time.min log.time.min
## 1    63 4.143135 1.0500000   0.04879016
## 2    27 3.295837 0.4500000  -0.79850770
## 3    66 4.189655 1.1000000   0.09531018
## 4   105 4.653960 1.7500000   0.55961579
## 5    38 3.637586 0.6333333  -0.45675840
## 6    88 4.477337 1.4666667   0.38299225
## 7    28 3.332205 0.4666667  -0.76214005
## 8    73 4.290459 1.2166667   0.19611488
## 9    19 2.944439 0.3166667  -1.14990558
## 10   53 3.970292 0.8833333  -0.12405265
```

```
summary(lm.out.abs.time.min <-
          lm(skalo.true ~ lr.dist.abs + log.time.min,
             data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + log.time.min, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9003 -1.5024  0.1592  1.4577  8.0360
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.00880    0.05698  52.803   <2e-16 ***
## lr.dist.abs  -1.06416    0.01490 -71.442   <2e-16 ***
## log.time.min  0.09624    0.05430   1.772   0.0764 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.108 on 3830 degrees of freedom
## Multiple R-squared:  0.5713, Adjusted R-squared:  0.5711
## F-statistic:  2552 on 2 and 3830 DF,  p-value: < 2.2e-16
```

Concerning the estimated effect of the left-right distance, it is almost identical with that in the previous model with only one independent variable: -1.0637421. This is because there is alsmot zero covariance between both independent variables. Corresnpondingly, the estimated effect is almost zero if you regress the log response time on the left-right distance:

```
naive.cov(long.data$log.time , long.data$lr.dist.abs)
```

```
## [1] 0.02269566
```

```
summary(lm(log.time ~ lr.dist.abs,data=long.data))
```

```
##
## Call:
```

```
## lm(formula = log.time ~ lr.dist.abs, data = long.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.35582 -0.42898 -0.07062  0.30129  2.79982
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.886037   0.016617  233.86   <2e-16 ***
## lr.dist.abs 0.004341   0.004431    0.98    0.327
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6273 on 3831 degrees of freedom
## Multiple R-squared:  0.0002505,  Adjusted R-squared:  -1.049e-05
## F-statistic: 0.9598 on 1 and 3831 DF,  p-value: 0.3273
```

That is, there is no omitted variable bias due to the response time variable. Note that this does not necessarily mean that there does not exist any omitted variable bias, at all. Potentially there can be further important variable.

Regressing like/dislike of parties on left-right ideological distance, the squared distance and log response time.

We now extend the above multiple regression models with the squared distance (lr.dist.sqr).

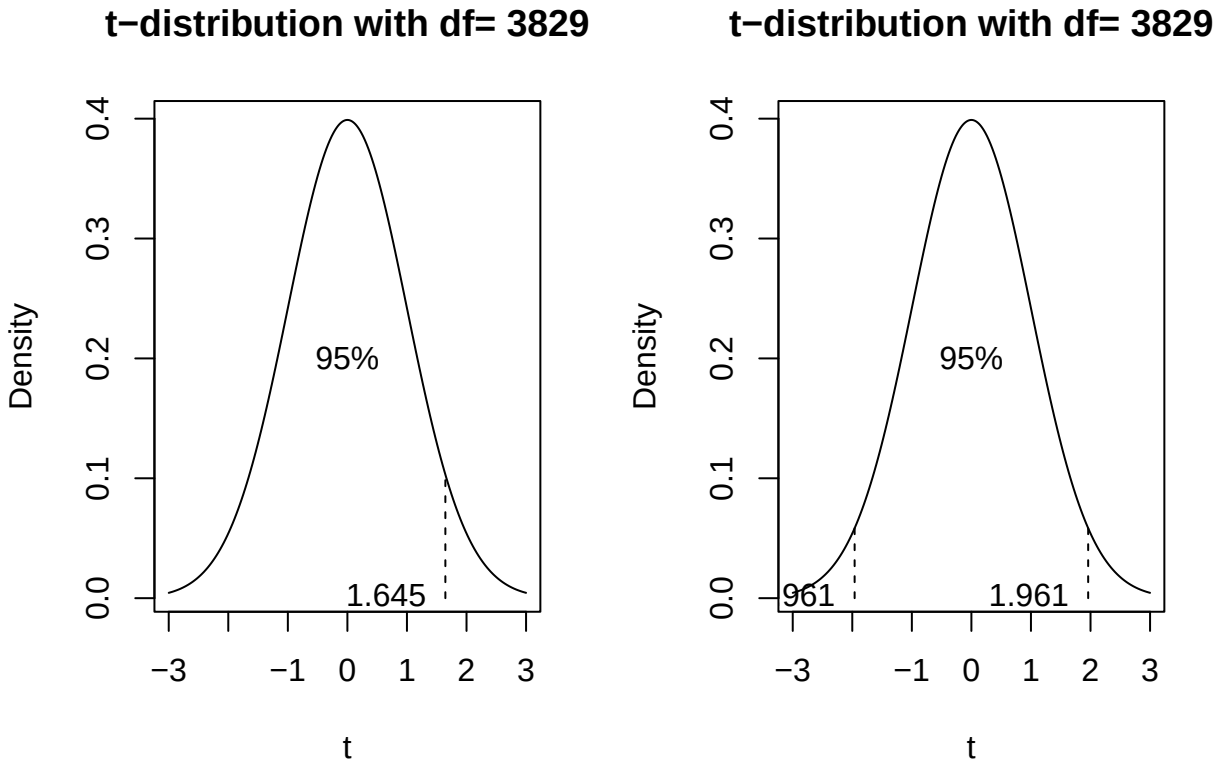
```
summary(lm.out.abs.sqr.time <-
          lm(skalo.true ~ lr.dist.abs +  lr.dist.sqr + log.time ,
             data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + lr.dist.sqr + log.time,
##     data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.2485 -1.4330  0.0771  1.4246  8.0738
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.000875   0.223298  13.439  < 2e-16 ***
## lr.dist.abs -1.381693   0.046302 -29.841  < 2e-16 ***
## lr.dist.sqr  0.042488   0.005871   7.237  5.5e-13 ***
## log.time     0.086030   0.053959   1.594    0.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.094 on 3829 degrees of freedom
## Multiple R-squared:  0.5771, Adjusted R-squared:  0.5768
## F-statistic:  1742 on 3 and 3829 DF,  p-value: < 2.2e-16
```

You have to be able to test the corresponding hypotheses in one-sided and two-sided manners.

t distribution for one-sided and two-sided alternatives

```
x.scale <- seq(-3,3,by=0.01)
this.df <- lm.out.abs.sqr.time$df.residual
this.y <- dt(x.scale,df=this.df)

par(mfrow=c(1,2))

for (zweiseitig in c(FALSE,TRUE)){
  plot(x.scale,this.y,type="l",xlab="t",ylab="Density",
       main=paste("t-distribution with df=",this.df))

  if (zweiseitig) {this.95 <- qt(c(.025,.975),df=this.df)
       } else {
       this.95 <- qt(c(.95),df=this.df)
     }
  for (i in 1:length(this.95)){
    lines(rep(this.95[i],2),c(0,dt(this.95[i],df=this.df)),lty=2)
    text(this.95[i],0,pos=2,round(this.95[i],3))
  }
  text(0,0.2,"95%")
}
```



Constructing the 95% confidence interval

Obtain the point estimates and standard errors.

```
point.estimates <- coefficients(summary(lm.out.abs.sqr.time))[,1]
standard.errors <- coefficients(summary(lm.out.abs.sqr.time))[,2]
```

Compute the critical value.

```
c.95 <- qt(0.975,df= lm.out.abs.sqr.time$df.residual)
c.95
```

```
## [1] 1.960584
```

Calculate the upper and lower bounds of confidence intervals.

```
upperb <- point.estimates + standard.errors * c.95
lowerb <- point.estimates + standard.errors * c.95 *-1

cbind(point.estimates,standard.errors,
      lowerb,upperb)
```

```
##             point.estimates standard.errors      lowerb      upperb
## (Intercept)      3.00087470     0.223298109  2.56308006  3.43866934
## lr.dist.abs     -1.38169327     0.046301875 -1.47247197 -1.29091457
## lr.dist.sqr      0.04248804     0.005870624  0.03097818  0.05399789
## log.time         0.08603009     0.053958668 -0.01976039  0.19182058
```

Is the effect of log reponse time larger than that of the squared distance?

To do this, you have to set up another variable by summing both independent variables.

```
long.data$time.sqr <- long.data$log.time + long.data$lr.dist.sqr
```

Replace the squared distance variable with the new variable.

```
summary(lm.out.abs.sqr.time.2 <-
          lm(skalo.true ~ lr.dist.abs + log.time + time.sqr ,
             data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + log.time + time.sqr,
##     data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.2485 -1.4330  0.0771  1.4246  8.0738
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.000875   0.223298  13.439  < 2e-16 ***
## lr.dist.abs -1.381693   0.046302 -29.841  < 2e-16 ***
## log.time     0.043542   0.054430   0.800    0.424
## time.sqr     0.042488   0.005871   7.237  5.5e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.094 on 3829 degrees of freedom
## Multiple R-squared:  0.5771, Adjusted R-squared:  0.5768
## F-statistic:  1742 on 3 and 3829 DF,  p-value: < 2.2e-16
```

The effect of "log.time" is the difference of the effect of "log.time" and "squared distance" in the original regression. The output tells that the difference is not significant at 5% level.

Testing against the null hypothesis that "log.time" AND "lr.dist.sqr" have no effect on Y.

To test whether "log.time" AND "lr.dist.sqr" variables have no effect on the dependent variable, We first estimate the restricted model without the above variables.

```
summary(lm.out.res <-
          lm(skalo.true ~ lr.dist.abs ,
             data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9169 -1.4955  0.1624  1.4471  8.1145
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.98875    0.05586    53.5   <2e-16 ***
## lr.dist.abs -1.06374    0.01490   -71.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.109 on 3831 degrees of freedom
## Multiple R-squared:  0.571,  Adjusted R-squared:  0.5709
## F-statistic:  5098 on 1 and 3831 DF,  p-value: < 2.2e-16
```

Subsequently, we calculate F value by using SSR.

```
SSR.ur <- sum(lm.out.abs.sqr.time$residuals^2)
SSR.r <-  sum(lm.out.res$residuals^2)
q <- lm.out.res$df.residual - lm.out.abs.sqr.time$df.residual

F <- ((SSR.r - SSR.ur)/q)/(SSR.ur/lm.out.abs.sqr.time$df.residual)
F
```

```
## [1] 27.78173
```

The same value can be also obtained by using R^2.

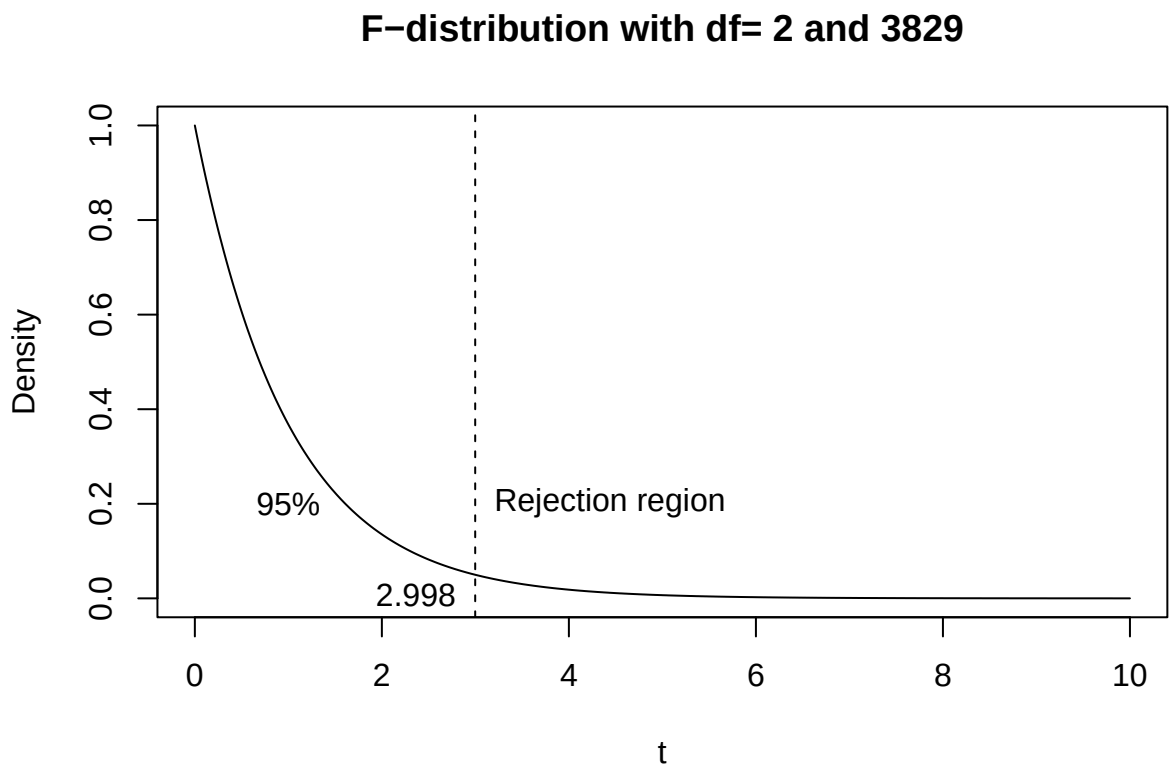
```
R2.ur <- summary(lm.out.abs.sqr.time)$r.squared
R2.r <- summary(lm.out.res)$r.squared



F <- ((R2.ur - R2.r)/q)/((1-R2.ur)/lm.out.abs.sqr.time$df.residual)
F
```

```
## [1] 27.78173
```

We now compare the F value with the F distribution.

```
x.scale <- seq(0,10,by=0.01)
this.y <- df(x.scale,df1=q,df2=lm.out.abs.sqr.time$df.residual)
```

```
plot(x.scale,this.y,type="l",xlab="t",ylab="Density",
       main=paste("F-distribution with df=",q,"and",lm.out.abs.sqr.time$df.residual))

this.95 <- qf(c(.95),df1=q,df2=lm.out.abs.sqr.time$df.residual)
abline(v=this.95,lty=2)
text(1,0.2,"95%")
text(this.95,0,pos=2,round(this.95,3))
text(this.95,0.2,"Rejection region",pos=4)
```



Since the above calculated F is in the rejection region, we can reject the null hypothesis.

Confidence intervals for predictions

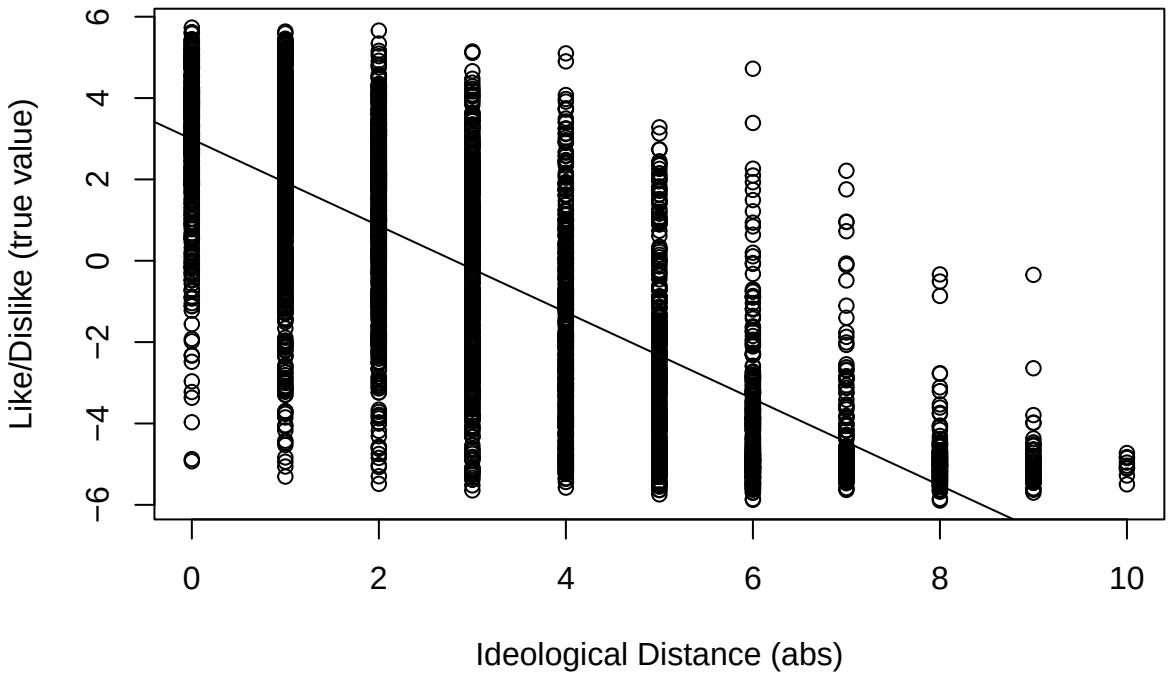
Let's turn to the original question whether ideological distance affects party evaluation. We have already estimated the simple regression model:

```
summary(lm.out.abs <- lm((long.data$skalo.true ~ long.data$lr.dist.abs)))
```

```
##
## Call:
## lm(formula = (long.data$skalo.true ~ long.data$lr.dist.abs))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9169 -1.4955  0.1624  1.4471  8.1145
##
```

```
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              2.98875    0.05586    53.5   <2e-16 ***
## long.data$lr.dist.abs -1.06374    0.01490   -71.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.109 on 3831 degrees of freedom
## Multiple R-squared:  0.571,  Adjusted R-squared:  0.5709
## F-statistic:  5098 on 1 and 3831 DF,  p-value: < 2.2e-16
```

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Ideological Distance (abs)",
     ylab="Like/Dislike (true value)")
abline(lm.out.abs)
```



The regression line represents the expected value given certain x values. Now, we wish to obtain the standard error of the expected value given $x = c$. For this purpose, we can just regress y on $x - c$.

For example, the expected value for ideological distance with 5 units has the following standard error:

```
  new.x <- long.data$lr.dist.abs - 5
  lm.out <- lm(skalo.true ~ new.x,data=long.data)
  summary(lm.out)$coefficients
```

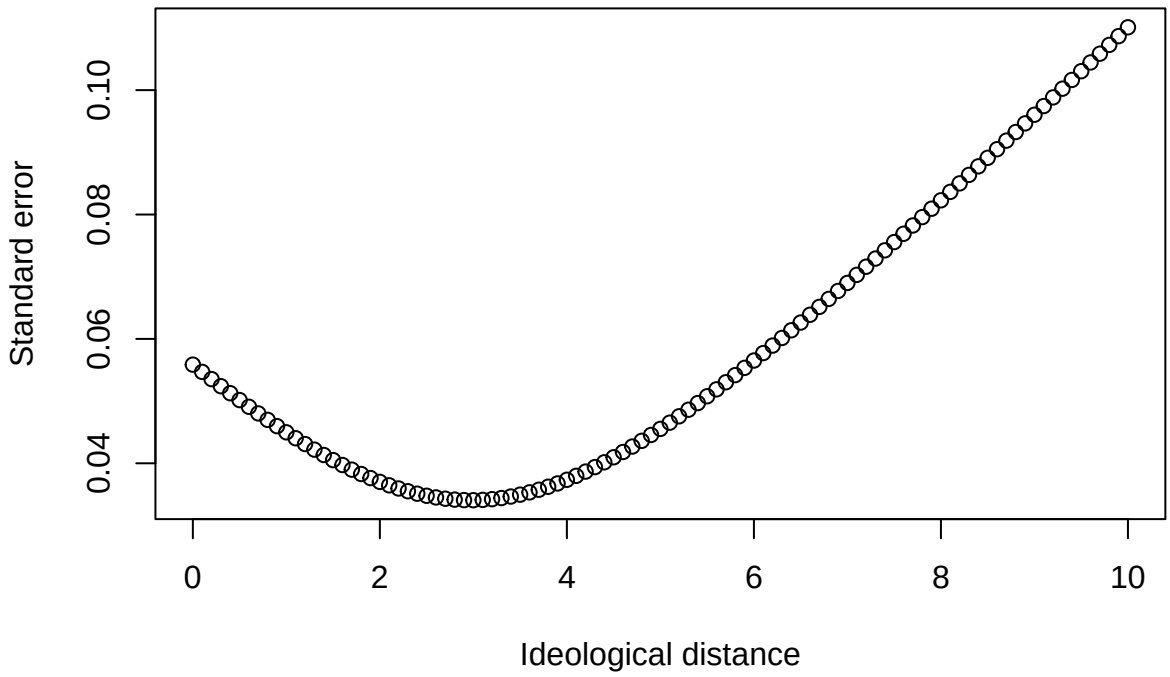
```
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -2.329961 0.04553017 -51.1740        0
## new.x       -1.063742 0.01489777 -71.4028        0
```

The standard error of the intercept -2.33 is the standard error of the expected value.

We repeat the same exercise for the x values between 0 and 10.

We first calculate the standard errors for all x values:

```
x.values <- seq(0,10,by=0.1)

all.se <- rep(NA,length(x.values))
for (i in 1:length(x.values)){
  new.x <- long.data$lr.dist.abs - x.values[i]
  lm.out <- lm(skalo.true ~ new.x,data=long.data)
  all.se[i] <- summary(lm.out)$coefficients[1,2]
}

plot(x.values,all.se,xlab="Ideological distance",ylab="Standard error")
```

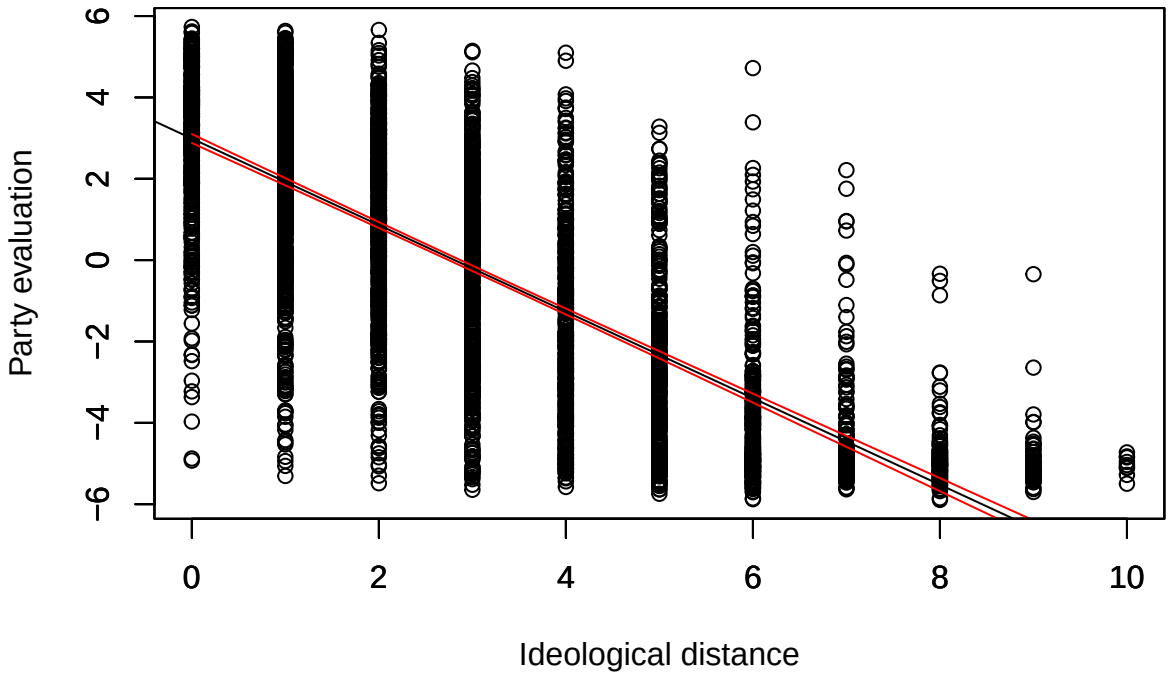


It is obvious that the standard errors of expected values are not constant, but they are larger for extreme values in x. In contrast, the smallest standard error is around 3. More precisely, it is at x=2.972, which is the mean value of x.

We build the confidence intervals by using the calculated standard errors:

```
predicted <- coefficients(lm.out.abs)[1] + coefficients(lm.out.abs)[2]*x.values

critical.value <- qt(0.975,df=lm.out.abs$df.residual)

upper.bounds <- predicted + critical.value*all.se
lower.bounds <- predicted - critical.value*all.se
```

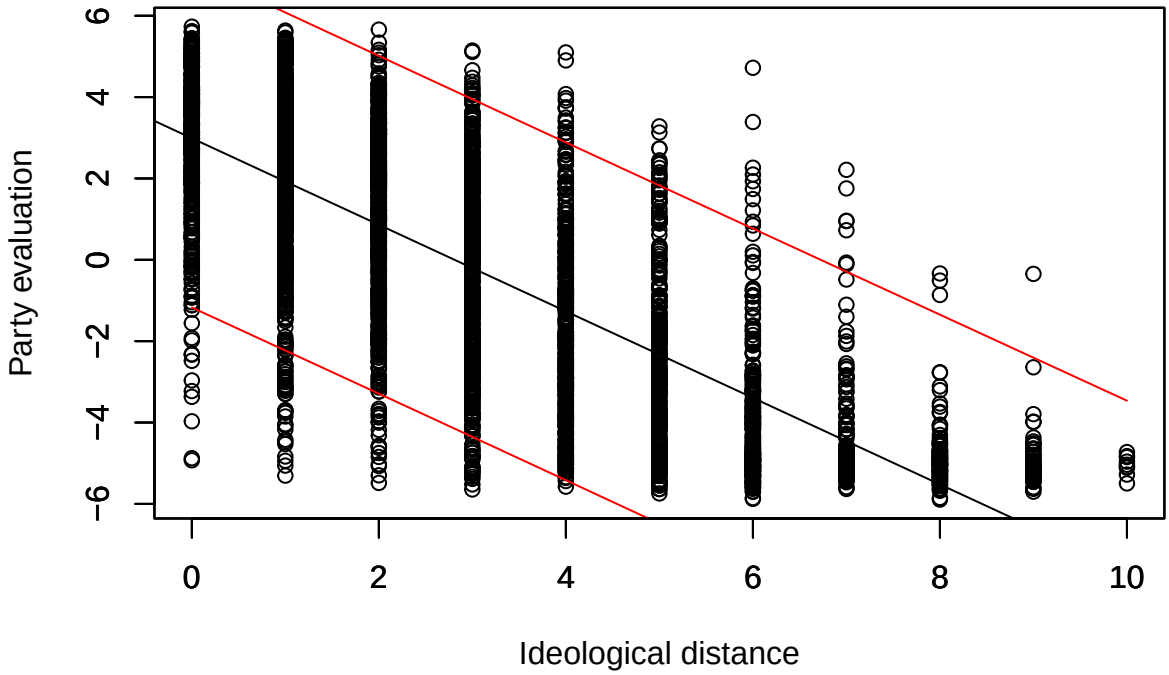
```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,ylab="Party evaluation",xlab="Ideological distance",
            ylim=range(long.data$skalo.true,na.rm=T),
            xlim=range(long.data$lr.dist.abs,na.rm=T))
abline(lm.out.abs)
par(new=T)
plot(x.values,upper.bounds,ann=F,xlab="",ylab="",axes=F,col="red",type="l",
            ylim=range(long.data$skalo.true,na.rm=T),
            xlim=range(long.data$lr.dist.abs,na.rm=T))
par(new=T)
plot(x.values,lower.bounds,ann=F,xlab="",ylab="",axes=F,col="red",type="l",
            ylim=range(long.data$skalo.true,na.rm=T),
            xlim=range(long.data$lr.dist.abs,na.rm=T))
```



Above, we built confidence intervals for the expected values of y given x. We can also build the confidence interval for y given x (prediction interval). The variance of y given x consists of the estimated error variance ($\hat{\sigma}^2$) and the standard error above.

```
predict.var <- all.se + summary(lm.out.abs)$sigma^2
predict.se <- sqrt(predict.var)

critical.value <- qt(0.975,df=lm.out.abs$df.residual)

upper.bounds <- predicted + critical.value*predict.se
lower.bounds <- predicted - critical.value*predict.se

plot(long.data$skalo.true ~ long.data$lr.dist.abs,ylab="Party evaluation",xlab="Ideological distance",
```

```
            ylim=range(long.data$skalo.true,na.rm=T),
            xlim=range(long.data$lr.dist.abs,na.rm=T))
abline(lm.out.abs)
par(new=T)
plot(x.values,upper.bounds,ann=F,xlab="",ylab="",axes=F,col="red",type="l",
            ylim=range(long.data$skalo.true,na.rm=T),
            xlim=range(long.data$lr.dist.abs,na.rm=T))
par(new=T)
plot(x.values,lower.bounds,ann=F,xlab="",ylab="",axes=F,col="red",type="l",
            ylim=range(long.data$skalo.true,na.rm=T),
            xlim=range(long.data$lr.dist.abs,na.rm=T))
```



Including dummy variables of parties as predictors

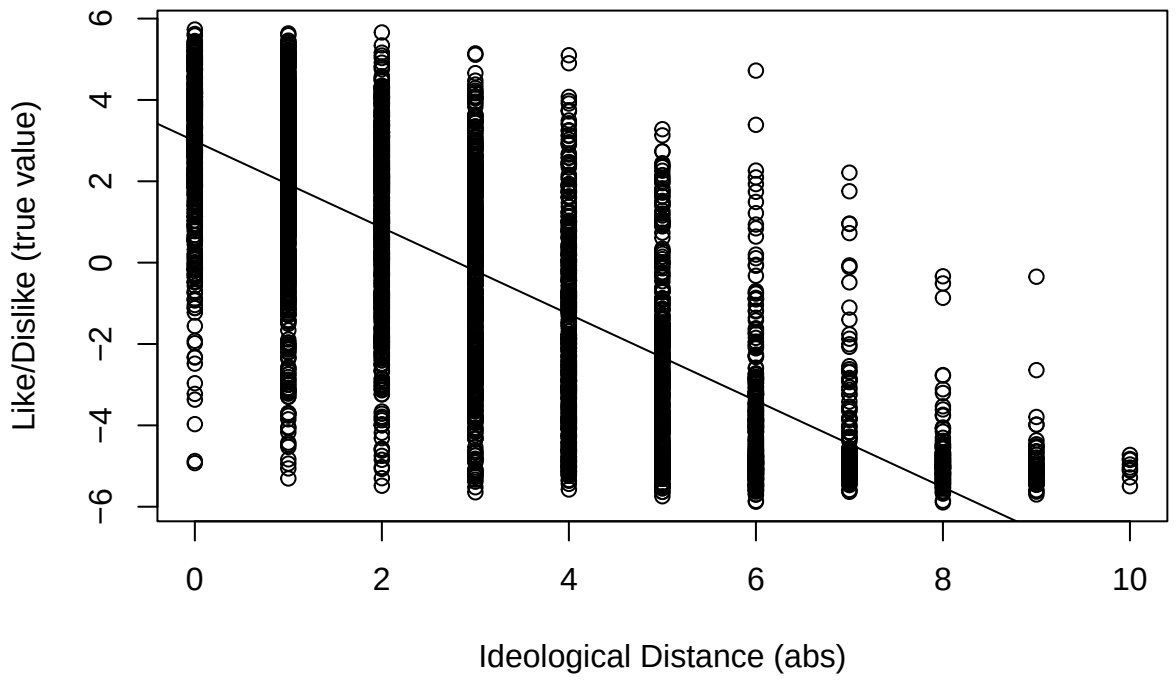
To see whether the ideological distance affects party evaluation, we have already estimated the following model for many times:

```
summary(lm.out.abs <- lm(skalo.true ~ lr.dist.abs,data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9169 -1.4955  0.1624  1.4471  8.1145
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.98875    0.05586    53.5   <2e-16 ***
## lr.dist.abs -1.06374    0.01490   -71.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.109 on 3831 degrees of freedom
## Multiple R-squared:  0.571,  Adjusted R-squared:  0.5709
## F-statistic:  5098 on 1 and 3831 DF,  p-value: < 2.2e-16
```

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Ideological Distance (abs)",
     ylab="Like/Dislike (true value)")
abline(lm.out.abs)
```



While we have evaluation of 7 different parties in the data set, the model above assumes the same relationship between the ideological distance and party evaluation for all parties. In other words, the intercept and slope (and errors) are common for all parties. This is a quite restricted assumption. It is more reasonable to assume that the intercept varies among parties. It is because there can be various factors behind evaluation of different parties, which can result in different intercepts. The intercept is in our context the average evaluation of a party for the respondents with the same ideological position of the party at stake.

To do this, we can first create dummy variables for individual parties:

```
long.data$dummy.cdu <- ifelse(long.data$party==1,1,0)
long.data$dummy.csu <- ifelse(long.data$party==2,1,0)
```

```
long.data$dummy.spd <- ifelse(long.data$party==3,1,0)
long.data$dummy.gru <- ifelse(long.data$party==4,1,0)
long.data$dummy.fdp <- ifelse(long.data$party==5,1,0)
long.data$dummy.afd <- ifelse(long.data$party==6,1,0)
long.data$dummy.lin <- ifelse(long.data$party==7,1,0)

# check the dummy variable

head(long.data,n=10)
```

```
##       id wave party skalo lr.self lr time wg  skalo.true lr.dist.dif
## 1  10131    3     1     3       1  1   63 NA  2.95298887           0
## 2  10132    3     2     3       1  2   27 NA  2.58785664           1
## 3  10133    3     3     2       1 -1   66 NA  1.78079893          -2
## 4  10134    3     4    -2       1 -2  105 NA -1.59475972          -3
## 5  10135    3     5     1       1  0   38 NA  1.00255449          -1
## 6  10136    3     6    -2       1  3   88 NA -2.09629456           2
## 7  10137    3     7    -2       1 -4   28 NA -1.46655548          -5
## 8  10231    3     1     0       0  1   73 NA -0.27285115           1
## 9  10232    3     2     0       0  1   19 NA  0.27582130           1
## 10 10233    3     3     0       0 -1   53 NA  0.04731445          -1
##    lr.dist.abs lr.dist.sqr log.time  time.min log.time.min  time.sqr dummy.cdu
## 1            0           0 4.143135 1.0500000   0.04879016  4.143135         1
## 2            1           1 3.295837 0.4500000  -0.79850770  4.295837         0
## 3            2           4 4.189655 1.1000000   0.09531018  8.189655         0
## 4            3           9 4.653960 1.7500000   0.55961579 13.653960         0
## 5            1           1 3.637586 0.6333333  -0.45675840  4.637586         0
## 6            2           4 4.477337 1.4666667   0.38299225  8.477337         0
## 7            5          25 3.332205 0.4666667  -0.76214005 28.332205         0
## 8            1           1 4.290459 1.2166667   0.19611488  5.290459         1
## 9            1           1 2.944439 0.3166667  -1.14990558  3.944439         0
## 10           1           1 3.970292 0.8833333  -0.12405265  4.970292         0
##    dummy.csu dummy.spd dummy.gru dummy.fdp dummy.afd dummy.lin
## 1          0         0         0         0         0         0
## 2          1         0         0         0         0         0
## 3          0         1         0         0         0         0
## 4          0         0         1         0         0         0
## 5          0         0         0         1         0         0
## 6          0         0         0         0         1         0
## 7          0         0         0         0         0         1
## 8          0         0         0         0         0         0
## 9          1         0         0         0         0         0
## 10         0         1         0         0         0         0
```

Now we can check whether evaluation of CDU is different from that of the other parties:

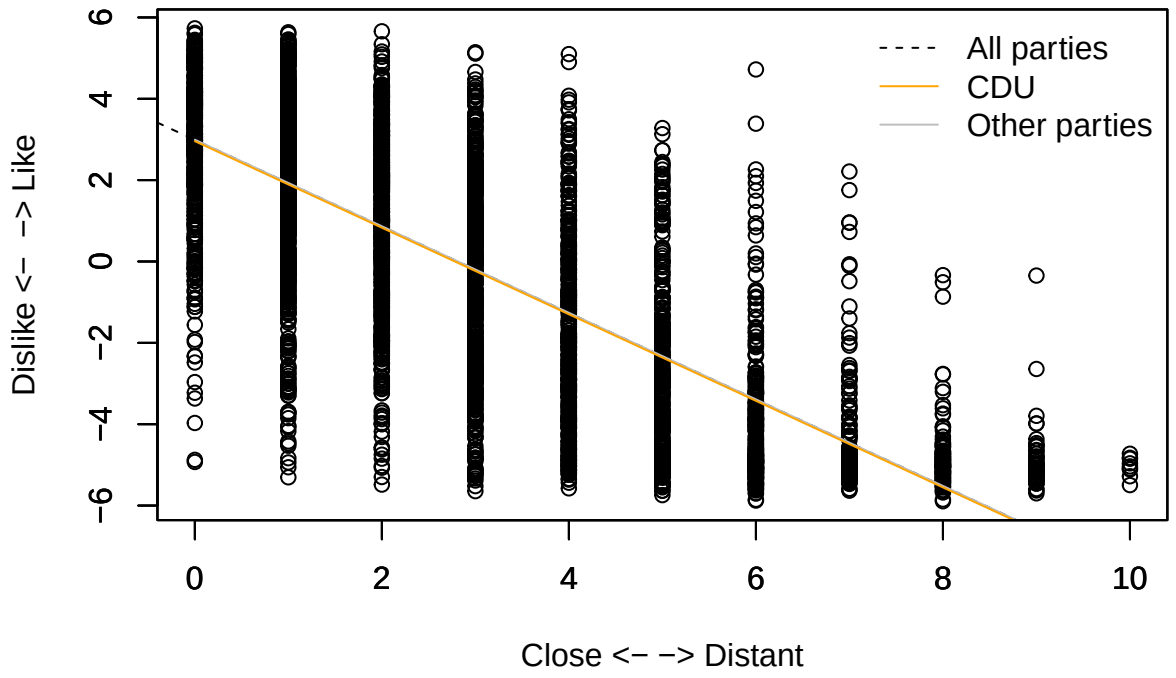
```
summary(lm.out.cdu <- lm(skalo.true ~ lr.dist.abs + dummy.cdu,data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + dummy.cdu, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8866 -1.4892  0.1638  1.4475  8.1099
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.99469    0.05807  51.568   <2e-16 ***
## lr.dist.abs -1.06397    0.01491 -71.353   <2e-16 ***
## dummy.cdu   -0.03624    0.09667  -0.375    0.708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.109 on 3830 degrees of freedom
## Multiple R-squared:  0.571,  Adjusted R-squared:  0.5708
## F-statistic:  2549 on 2 and 3830 DF,  p-value: < 2.2e-16
```

The results can be visualized as follows:

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Close <- -> Distant",ylab="Dislike <-  -> Like",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
abline(lm.out.abs,     lty=2,lwd=1)

predicted.cdu.0 <- coef(lm.out.cdu)[1] + coef(lm.out.cdu)[2]*c(0:10)
predicted.cdu.1 <- coef(lm.out.cdu)[1] + coef(lm.out.cdu)[2]*c(0:10) + coef(lm.out.cdu)[3]

par(new=T)
plot(predicted.cdu.0 ~ c(0:10),col="grey",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.cdu.1 ~ c(0:10),col="orange",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
legend("topright",col=c("black","orange","gray"),
       lty=c(2,1,1),c("All parties","CDU","Other parties"),
       bty="n")
```



We can find the intercept for CDU is slightly lower than the other parties. Its difference is however so small that it can be due to by chance.

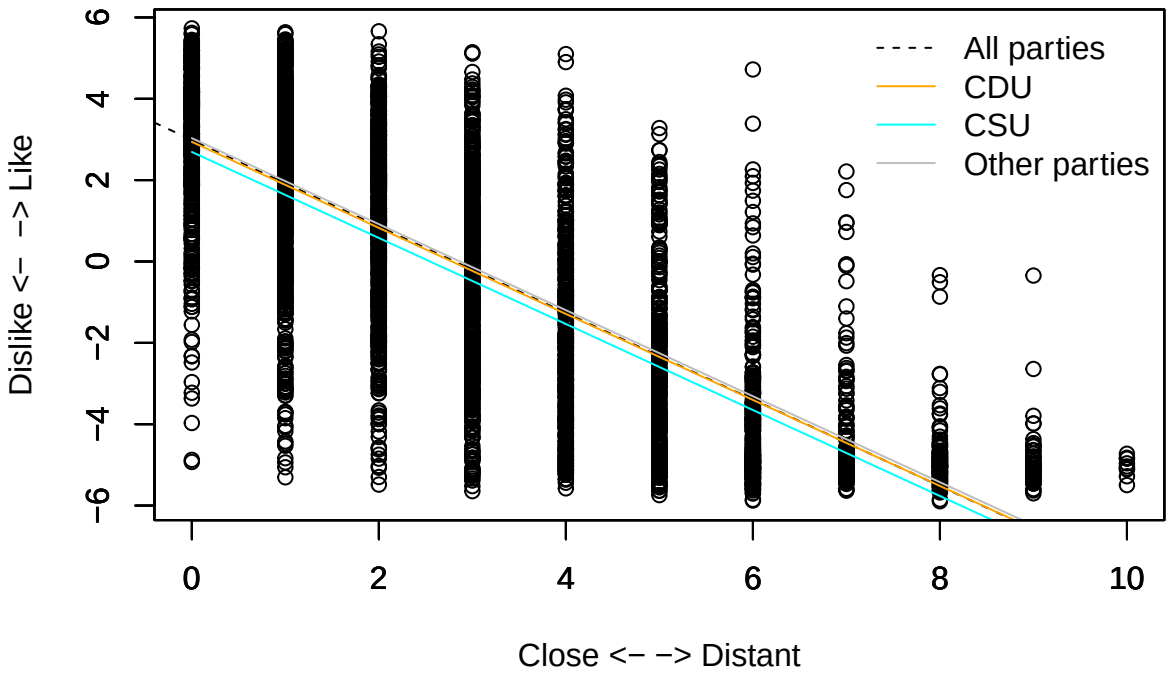
We can now extend the model with the intercept for CSU:

```
summary(lm.out.cdu.csu <- lm(skalo.true ~ lr.dist.abs +
           dummy.cdu + dummy.csu ,
           data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + dummy.cdu + dummy.csu,
##     data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9082 -1.4752  0.1541  1.4499  8.0330
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.03171    0.05899  51.391  < 2e-16 ***
## lr.dist.abs -1.05732    0.01502 -70.409  < 2e-16 ***
## dummy.cdu   -0.09155    0.09788  -0.935 0.349692
## dummy.csu   -0.33959    0.09929  -3.420 0.000632 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 2.106 on 3829 degrees of freedom
## Multiple R-squared:  0.5723, Adjusted R-squared:  0.572
## F-statistic:  1708 on 3 and 3829 DF,  p-value: < 2.2e-16
```

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Close <- -> Distant",ylab="Dislike <-  -> Like",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
abline(lm.out.abs,     lty=2,lwd=1)

predicted.others <- coef(lm.out.cdu.csu)[1] + coef(lm.out.cdu.csu)[2]*c(0:10)
predicted.cdu <- coef(lm.out.cdu.csu)[1] + coef(lm.out.cdu.csu)[2]*c(0:10) + coef(lm.out.cdu.csu)[3]
predicted.csu <- coef(lm.out.cdu.csu)[1] + coef(lm.out.cdu.csu)[2]*c(0:10) + coef(lm.out.cdu.csu)[4]

par(new=T)
plot(predicted.others ~ c(0:10),col="grey",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.cdu ~ c(0:10),col="orange",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.csu ~ c(0:10),col="cyan",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))

legend("topright",col=c("black","orange","cyan","gray"),
       lty=c(2,1,1,1),c("All parties","CDU","CSU","Other parties"),
       bty="n")
```



Here, the CSU's intercept is lower than the other parties and it is significant at 5% level.

To obtain the intercept for all parties, you regress the party evaluation on the left-right distance and all party dummy variables:

```
summary(lm.out.all <- lm(skalo.true ~ lr.dist.abs +
           dummy.cdu + dummy.csu + dummy.spd +
             dummy.gru + dummy.fdp + dummy.afd + dummy.lin,
           data=long.data))
```

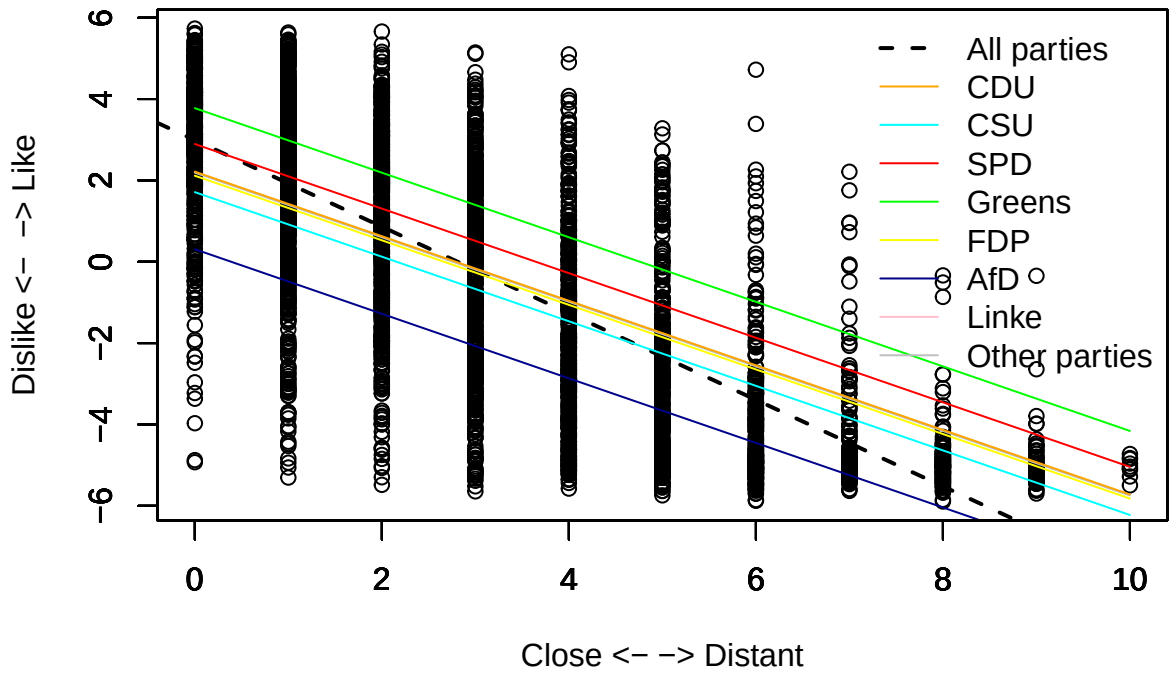
```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + dummy.cdu + dummy.csu +
##     dummy.spd + dummy.gru + dummy.fdp + dummy.afd + dummy.lin,
##     data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.2883 -1.3100  0.1097  1.3444  9.1747
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.18939    0.09845  22.239  < 2e-16 ***
## lr.dist.abs -0.79361    0.01835 -43.253  < 2e-16 ***
## dummy.cdu    0.02533    0.11809   0.214    0.830
## dummy.csu   -0.47876    0.11964  -4.002 6.41e-05 ***
## dummy.spd    0.70372    0.12087   5.822 6.29e-09 ***
```

```
## dummy.gru    1.58706    0.12130  13.084  < 2e-16 ***
## dummy.fdp   -0.07498    0.11970  -0.626    0.531
## dummy.afd   -1.88167    0.13424 -14.017  < 2e-16 ***
## dummy.lin         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.968 on 3825 degrees of freedom
## Multiple R-squared:  0.627,  Adjusted R-squared:  0.6263
## F-statistic: 918.5 on 7 and 3825 DF,  p-value: < 2.2e-16
```

The effect of the dummy variable for the Linke was not estimated. This is due to the perfect collinearity (the dummy variable trap). What is happening here can be seen in the following plot:

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Close <- -> Distant",ylab="Dislike <-  -> Like",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
abline(lm.out.abs,     lty=2,lwd=2)

predicted.others <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10)
predicted.cdu <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10) + coef(lm.out.all)[3]
predicted.csu <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10) + coef(lm.out.all)[4]
predicted.spd <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10) + coef(lm.out.all)[5]
predicted.gru <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10) + coef(lm.out.all)[6]
predicted.fdp <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10) + coef(lm.out.all)[7]
predicted.afd <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10) + coef(lm.out.all)[8]
predicted.lin <- coef(lm.out.all)[1] + coef(lm.out.all)[2]*c(0:10) + coef(lm.out.all)[9]

par(new=T)
plot(predicted.others ~ c(0:10),col="grey",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.cdu ~ c(0:10),col="orange",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.csu ~ c(0:10),col="cyan",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.spd ~ c(0:10),col="red",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.gru ~ c(0:10),col="green",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
```

```
plot(predicted.fdp ~ c(0:10),col="yellow",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.afd ~ c(0:10),col="blue4",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.lin ~ c(0:10),col="pink",type="l",
     ann=F,xlab="",ylab="",axes=F,
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))

legend("topright",col=c("black","orange","cyan","red","green","yellow","blue4","pink","gray"),
       lty=c(2,rep(1,8)),c("All parties","CDU","CSU","SPD","Greens","FDP","AfD","Linke",
                           "Other parties"),
       lwd=c(2,rep(1,8)),
       bty="n")
```



In the figure, you can see the regression lines for all parties except for the Linke. However, you see the gray line for the others. And this corresponds to the regression line for the linke.

The equivalent model can be also estimated as follows:

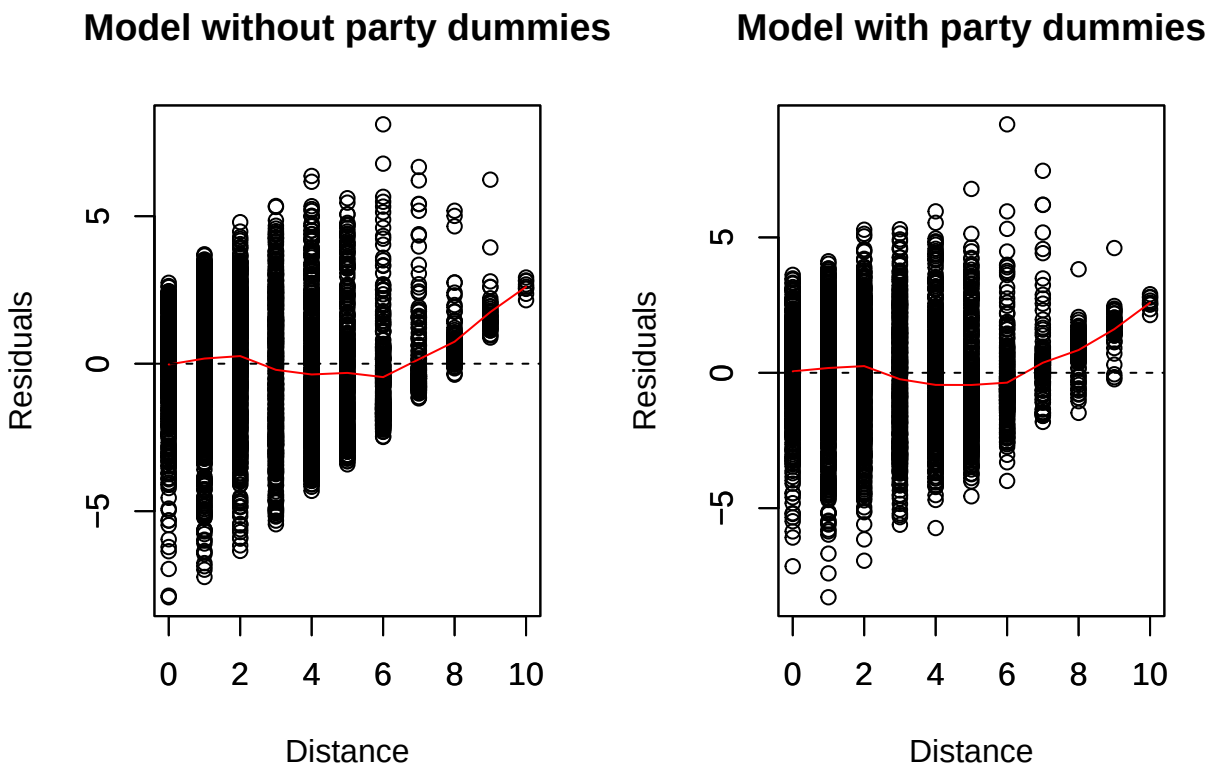
```
summary(lm.out.all2 <- lm(skalo.true ~ lr.dist.abs + as.factor(party),data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + as.factor(party), data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.2883 -1.3100  0.1097  1.3444  9.1747
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)         2.21472    0.09740  22.737  < 2e-16 ***
## lr.dist.abs        -0.79361    0.01835 -43.253  < 2e-16 ***
## as.factor(party)2 -0.50408    0.11958  -4.216 2.55e-05 ***
## as.factor(party)3  0.67839    0.12033   5.638 1.85e-08 ***
## as.factor(party)4  1.56174    0.12073  12.936  < 2e-16 ***
## as.factor(party)5 -0.10031    0.11942  -0.840    0.401
## as.factor(party)6 -1.90700    0.13464 -14.163  < 2e-16 ***
## as.factor(party)7 -0.02533    0.11809  -0.214    0.830
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.968 on 3825 degrees of freedom
## Multiple R-squared:  0.627,  Adjusted R-squared:  0.6263
## F-statistic: 918.5 on 7 and 3825 DF,  p-value: < 2.2e-16
```

In this model, the effect of the CDU-dummy was dropped. But the result is completely equivalent to the last model.

Now, we can observe the residuals to check the zero conditional mean:

```
par(mfrow=c(1,2))
resid.1 <- residuals(lm.out.abs)
plot(resid.1 ~ long.data$lr.dist.abs,
     ylim=range(resid.1),xlim=c(0,10),
     ylab="Residuals",xlab="Distance",
     main="Model without party dummies")
E.u <- tapply(resid.1,as.factor(long.data$lr.dist.abs),mean)
par(new=T)
plot(E.u ~ c(0:10),
     ylim=range(resid.1),xlim=c(0,10),
     type="l",col="red",ann=F,xlab="",ylab="",axes=F)
abline(h=0,lty=2)

resid.5 <- residuals(lm.out.all2)
plot(resid.5 ~ long.data$lr.dist.abs,
     ylab="Residuals",xlab="Distance",
     main="Model with party dummies")
E.u <- tapply(resid.5,as.factor(long.data$lr.dist.abs),mean)
par(new=T)
plot(E.u ~ c(0:10),
     ylim=range(resid.5),xlim=c(0,10),
     type="l",col="red",ann=F,xlab="",ylab="",axes=F)
abline(h=0,lty=2)
```



Here, we still have a problem in the range for large ideological distance values.

7.4 Interactions involving dummy variables

Let's come back to the first model with the dummy variable for CDU evaluation:

```
summary(lm.out.cdu)
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + dummy.cdu, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8866 -1.4892  0.1638  1.4475  8.1099
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.99469    0.05807  51.568   <2e-16 ***
## lr.dist.abs -1.06397    0.01491 -71.353   <2e-16 ***
## dummy.cdu   -0.03624    0.09667  -0.375    0.708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.109 on 3830 degrees of freedom
## Multiple R-squared:  0.571,  Adjusted R-squared:  0.5708
## F-statistic:  2549 on 2 and 3830 DF,  p-value: < 2.2e-16
```

While we estimated different intercepts for the evaluation of CDU and the other parties in the above model, we further assumed that the effect of the ideological distance is same for all parties. It is however not nescessarily true. The ideological distance can have different impacts on the evaluation of CDU and the other parties. To check this, we include now the interaction effect of both variables:

```
summary(lm.out.cdu.int <- lm(skalo.true ~ lr.dist.abs * dummy.cdu,data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs * dummy.cdu, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9060 -1.4900  0.1673  1.4509  8.1444
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            3.02949    0.05982  50.645   <2e-16 ***
## lr.dist.abs           -1.07553    0.01566 -68.679   <2e-16 ***
## dummy.cdu             -0.37590    0.17125  -2.195   0.0282 *
## lr.dist.abs:dummy.cdu  0.12239    0.05095   2.402   0.0163 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.108 on 3829 degrees of freedom
## Multiple R-squared:  0.5716, Adjusted R-squared:  0.5713
## F-statistic:  1703 on 3 and 3829 DF,  p-value: < 2.2e-16
```

This result is a bit tricky to interpret as you have already seen in a previous section. In the previous section, we visualized the results for ease of interpretation. This time, we conduct two separate regression analysis for CDU evaluation and the other evaluations. First we divide the dataset:

```
cdu.dat <- long.data[long.data$party==1,]
non.cdu.dat <- long.data[long.data$party!=1,]
```

Subsequently, the simple regression model is estimated for the CDU evaluation...

```
summary(lm.out.onlycdu <- lm(skalo.true ~ lr.dist.abs,data = cdu.dat))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs, data = cdu.dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.5818 -1.5745  0.2135  1.5887  4.9969
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.65359    0.16213   16.37   <2e-16 ***
## lr.dist.abs -0.95314    0.04898  -19.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.13 on 556 degrees of freedom
## Multiple R-squared:  0.4051, Adjusted R-squared:  0.404
## F-statistic: 378.6 on 1 and 556 DF,  p-value: < 2.2e-16
```

... and the others.

```
summary(lm.out.onlynoncdu <- lm( skalo.true ~ lr.dist.abs,data = non.cdu.dat))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs, data = non.cdu.dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9060 -1.4737  0.1601  1.4338  8.1444
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.02949    0.05971   50.74   <2e-16 ***
## lr.dist.abs -1.07553    0.01563  -68.80   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.104 on 3273 degrees of freedom
## Multiple R-squared:  0.5912, Adjusted R-squared:  0.5911
## F-statistic:  4734 on 1 and 3273 DF,  p-value: < 2.2e-16
```

The point estimates of both separate regression can be found in the first model, while the other elements (standard errors, degrees of freedom) are not always identical.

Now, we wish to test whether both regression models systematically differ from each other. The null-hypothesis is that both models are based on the same population model, which corresponds to the first model estimated without any interaction terms. We call the latter model the pooled model since it pools both groups into one dataset.

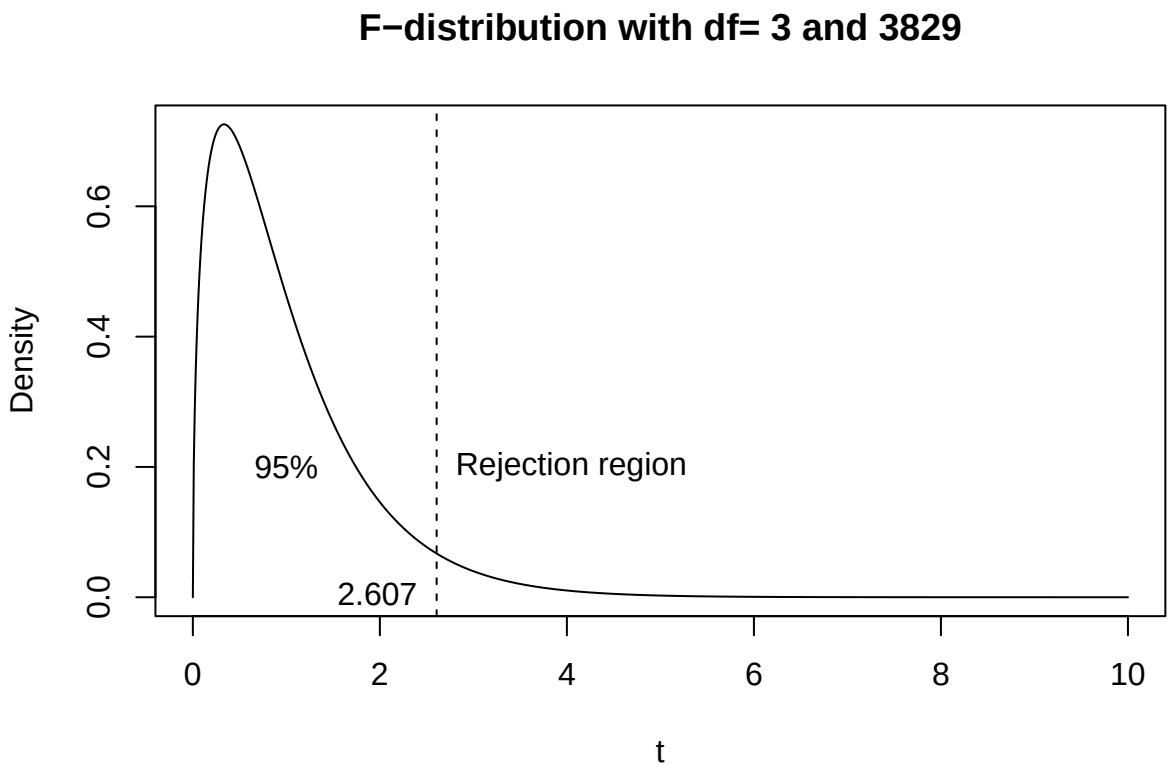
Here, the separated models constitute together an unrestricted model while the pooled model is restricted since we restrict the effects of both independent variables being same for both groups.

This can be tested by F-statistic (which is called "Chow statistic"):

$F = \frac{SSR_P - (SSR_1 + SSR_2)/(k+1)}{(SSR_1 + SSR_2)/[n-2(k+1)]}$

```
SSR.p <- sum(lm.out.abs$residuals^2)
SSR.1 <- sum(lm.out.onlycdu$residuals^2)
SSR.2 <- sum(lm.out.onlynoncdu$residuals^2)
k <- 2

sum.df <- lm.out.onlycdu$df.residual+lm.out.onlynoncdu$df.residual


F <- ((SSR.p - (SSR.1+SSR.2))/(k+1))/((SSR.1+SSR.2)/(sum.df))
F
```

```
## [1] 1.970553
```



We can not reject here the null-hypothesis.

Pooled Cross Sections

We estimate the following model with the quadratic function:

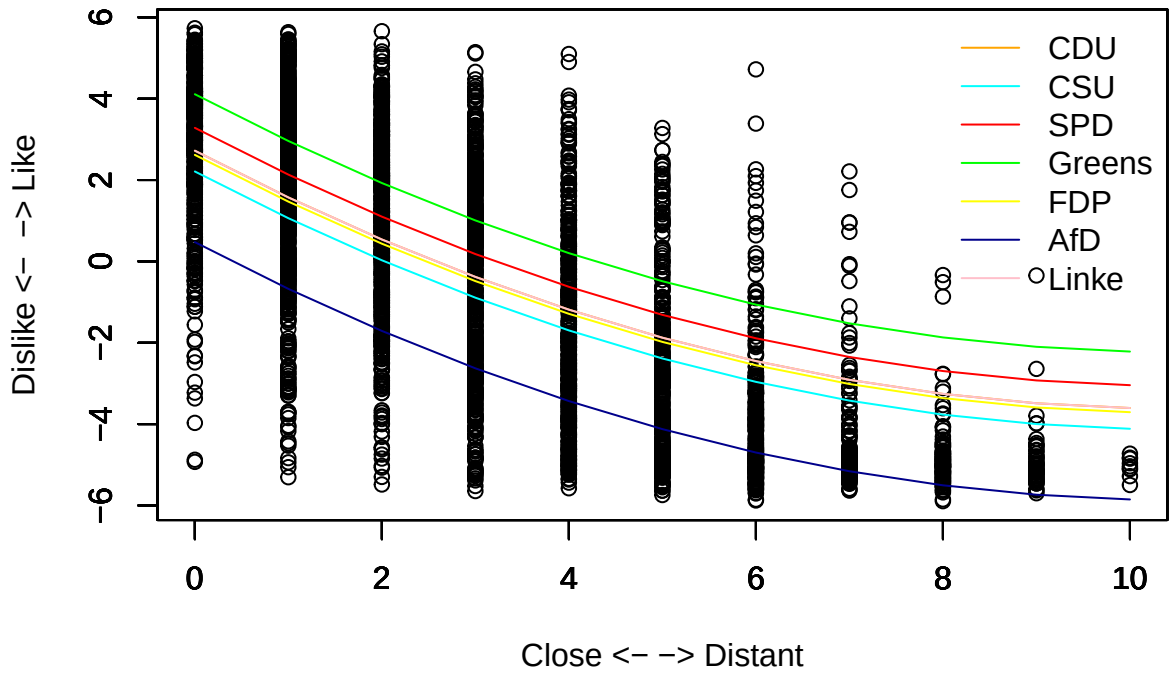
```
summary(lm.out.dummy <- lm(skalo.true ~ lr.dist.abs + lr.dist.sqr + as.factor(party)
                           ,data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + lr.dist.sqr + as.factor(party),
##     data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.2664 -1.2635  0.0597  1.2709  9.4194
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         2.7216632  0.1090836  24.950  < 2e-16 ***
## lr.dist.abs        -1.2064036  0.0456256 -26.441  < 2e-16 ***
## lr.dist.sqr         0.0573810  0.0058205   9.858  < 2e-16 ***
## as.factor(party)2 -0.5107611  0.1181026  -4.325 1.57e-05 ***
## as.factor(party)3  0.5615458  0.1194341   4.702 2.67e-06 ***
## as.factor(party)4  1.3882967  0.1205298  11.518  < 2e-16 ***
## as.factor(party)5 -0.1029053  0.1179475  -0.872    0.383
```

```
## as.factor(party)6 -2.2475507  0.1373943 -16.358  < 2e-16 ***
## as.factor(party)7  0.0003671  0.1166602   0.003    0.997
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.944 on 3824 degrees of freedom
## Multiple R-squared:  0.6362, Adjusted R-squared:  0.6355
## F-statistic: 836.1 on 8 and 3824 DF,  p-value: < 2.2e-16
```

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Close <- -> Distant",ylab="Dislike <-  -> Like",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))

predicted.cdu <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10)+ coef(lm.out.dummy)[3]* (c(0:10)
predicted.csu <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.spd <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.gru <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.fdp <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.afd <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.lin <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)

par(new=T)
plot(predicted.cdu ~ c(0:10),col="orange",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.csu ~ c(0:10),col="cyan",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.spd ~ c(0:10),col="red",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.gru ~ c(0:10),col="green",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.fdp ~ c(0:10),col="yellow",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.afd ~ c(0:10),col="blue4",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.lin ~ c(0:10),col="pink",type="l",
```

```
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))


legend("topright",col=c("orange","cyan","red","green","yellow","blue4","pink"),
       lty=c(rep(1,8)),c("CDU","CSU","SPD","Greens","FDP","AfD",
                         "Linke"),
       lwd=c(rep(1,8)),
       bty="n")
```



Let's look at the data again:

```
head(long.data[,1:7],n=20)
```

```
##       id wave party skalo lr.self lr time
## 1  10131    3     1     3       1  1   63
## 2  10132    3     2     3       1  2   27
## 3  10133    3     3     2       1 -1   66
## 4  10134    3     4    -2       1 -2  105
## 5  10135    3     5     1       1  0   38
## 6  10136    3     6    -2       1  3   88
## 7  10137    3     7    -2       1 -4   28
## 8  10231    3     1     0       0  1   73
## 9  10232    3     2     0       0  1   19
## 10 10233    3     3     0       0 -1   53
## 11 10234    3     4     1       0 -2   34
```

```
## 12 10235    3     5     4       0  0   42
## 13 10236    3     6    -4       0  4   84
## 14 10237    3     7    -4       0 -4   40
## 15 10411    1     1     2       2  2   54
## 16 10416    1     6     1       2  5   20
## 17 10421    2     1     2       2  1   26
## 18 10422    2     2     1       2  0   89
## 19 10423    2     3     0       2 -2   37
## 20 10424    2     4    -1       2 -3   42
```

There is a variable "wave" which codes the time point of data collection:

```
table(long.data$wave)
```

```
##
##    1    2    3
## 1303 1200 1330
```

According to the frequency table, the data were collected at three different time points. That is, we pooled three cross sections.

We might be interested whether the relationship between party evaluation and ideological distance is different at the three time points. One possibility is to estimate different regression models for each time point and compare them with the overall regression (the Chow test). Here, we add the interaction effects with the dummy variable for the panel waves. This is equivalent to estimate three separate regression models:

```
summary(lm.out.chow <- lm(skalo.true ~ lr.dist.abs* as.factor(wave) +
                            lr.dist.sqr* as.factor(wave) +
                            as.factor(party)* as.factor(wave) ,data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs * as.factor(wave) + lr.dist.sqr *
##     as.factor(wave) + as.factor(party) * as.factor(wave), data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.1774 -1.1864  0.0511  1.2058  9.2650
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         3.240880   0.185694  17.453  < 2e-16 ***
## lr.dist.abs                        -1.184191   0.076308 -15.519  < 2e-16 ***
## as.factor(wave)2                   -0.226639   0.266339  -0.851  0.39486
## as.factor(wave)3                   -1.247385   0.259291  -4.811 1.56e-06 ***
## lr.dist.sqr                         0.050942   0.009647   5.281 1.36e-07 ***
## as.factor(party)2                  -0.625045   0.199293  -3.136  0.00172 **
## as.factor(party)3                  -0.101776   0.201804  -0.504  0.61406
## as.factor(party)4                   0.966997   0.205900   4.696 2.74e-06 ***
## as.factor(party)5                  -1.047352   0.199638  -5.246 1.64e-07 ***
## as.factor(party)6                  -2.513897   0.236016 -10.651  < 2e-16 ***
## as.factor(party)7                  -0.640586   0.196615  -3.258  0.00113 **
## lr.dist.abs:as.factor(wave)2        0.129119   0.112864   1.144  0.25269
## lr.dist.abs:as.factor(wave)3       -0.210083   0.106901  -1.965  0.04946 *
## as.factor(wave)2:lr.dist.sqr       -0.007168   0.014729  -0.487  0.62655
## as.factor(wave)3:lr.dist.sqr        0.028950   0.013414   2.158  0.03098 *
## as.factor(wave)2:as.factor(party)2 -0.163555   0.288402  -0.567  0.57067
```

```
## as.factor(wave)3:as.factor(party)2  0.459597   0.279280   1.646  0.09992 .
## as.factor(wave)2:as.factor(party)3  0.116779   0.291091   0.401  0.68831
## as.factor(wave)3:as.factor(party)3  1.807898   0.282874   6.391 1.85e-10 ***
## as.factor(wave)2:as.factor(party)4 -0.015798   0.294807  -0.054  0.95727
## as.factor(wave)3:as.factor(party)4  1.225895   0.286258   4.282 1.89e-05 ***
## as.factor(wave)2:as.factor(party)5  0.272698   0.287604   0.948  0.34310
## as.factor(wave)3:as.factor(party)5  2.453308   0.279512   8.777  < 2e-16 ***
## as.factor(wave)2:as.factor(party)6 -0.244615   0.344067  -0.711  0.47716
## as.factor(wave)3:as.factor(party)6  0.967614   0.322929   2.996  0.00275 **
## as.factor(wave)2:as.factor(party)7  0.010938   0.283802   0.039  0.96926
## as.factor(wave)3:as.factor(party)7  1.835873   0.276539   6.639 3.61e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.908 on 3806 degrees of freedom
## Multiple R-squared:  0.6513, Adjusted R-squared:  0.6489
## F-statistic: 273.4 on 26 and 3806 DF,  p-value: < 2.2e-16
```

The result shows that multiple interaction effects with the 3rd wave is significant. That is, we find significant differences in the regression lines between the first/second waves and the third wave. Further, R-square is larger than that of the model without dummy ('r round(summary(lm.out.dummy)$r.squared,4)'). It is also possible to calculate the exact F-statistics based on the R-square, which is large enough to reject the null hypothesis of no structural change across time.

Panel data and fixed effect models

In the above analysis, we just treated both panel waves as independent (*independently pooled cross sections*). If one looks at the data more closely, this is not the case:

```
head(long.data[24:100,1:7],n=20)
```

```
##       id wave party skalo lr.self lr time
## 24 10511    1     1    -5      -2  1   40
## 25 10512    1     2    -5      -2  1   24
## 26 10513    1     3     2      -2 -2   67
## 27 10514    1     4     5      -2 -2   23
## 28 10515    1     5    -5      -2  0   55
## 29 10516    1     6    -5      -2  4   64
## 30 10517    1     7     0      -2 -4   34
## 31 10521    2     1    -1      -2  1   33
## 32 10522    2     2    -3      -2  2  244
## 33 10523    2     3     1      -2 -1   25
## 34 10524    2     4     4      -2 -2  211
## 35 10525    2     5    -3      -2  1   31
## 36 10526    2     6    -5      -2  4   19
## 37 10527    2     7     0      -2 -4   49
## 38 10611    1     1    -2      -2  2   57
## 39 10612    1     2    -4      -2  4   49
## 40 10613    1     3     3      -2 -2   35
## 41 10614    1     4     5      -2 -1  106
## 42 10615    1     5     2      -2  0   69
## 43 10616    1     6    -5      -2  5   41
```

The first three digits of *id* is the subject number and this indicates that the first 14 rows are of the same respondent. For each panel wave, we have 7 evaluations of the same individual. Therefore, we have here a panel dataset.

It is reasonable to assume that individual respondents use the scale for party evaluation differently. That is, some tend to evaluate parties more positively in general than others. To capture such an unobserved heterogeneity between individual subjects, we include the dummy variable for all respondents:

```
long.data$subject <- round(long.data$id/100)

summary(lm.out.dummy.fe <- lm(skalo.true ~ lr.dist.abs +lr.dist.sqr + as.factor(party) + as.factor(wave)
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + lr.dist.sqr + as.factor(party) +
##     as.factor(wave) + as.factor(subject), data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4089 -1.0808  0.0292  1.0837 10.2201
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              3.714568   0.871440   4.263 2.08e-05 ***
## lr.dist.abs             -1.169170   0.044405 -26.329  < 2e-16 ***
## lr.dist.sqr              0.039785   0.005954   6.682 2.74e-11 ***
## as.factor(party)2       -0.412518   0.109234  -3.776 0.000162 ***
## as.factor(party)3        0.486107   0.110486   4.400 1.12e-05 ***
## as.factor(party)4        1.312333   0.111516  11.768  < 2e-16 ***
## as.factor(party)5       -0.095499   0.108843  -0.877 0.380330
## as.factor(party)6       -1.846606   0.134087 -13.772  < 2e-16 ***
## as.factor(party)7        0.010680   0.107538   0.099 0.920892
## as.factor(wave)2         0.081136   0.079169   1.025 0.305508
## as.factor(wave)3        -1.096009   0.541395  -2.024 0.043005 *
## as.factor(subject)102 -1.088084   0.956880  -1.137 0.255569
## as.factor(subject)104 -0.127905   1.050376  -0.122 0.903088
## as.factor(subject)105 -2.484700   0.988164  -2.514 0.011967 *
## as.factor(subject)106 -0.975558   0.988460  -0.987 0.323738
## as.factor(subject)107  0.792776   1.100287   0.721 0.471255
## as.factor(subject)109 -0.622143   0.988992  -0.629 0.529347
## as.factor(subject)110 -0.697865   1.099492  -0.635 0.525656
## as.factor(subject)111 -0.352853   0.989716  -0.357 0.721474
## as.factor(subject)112 -0.240519   1.099645  -0.219 0.826878
## as.factor(subject)113 -1.036240   0.997100  -1.039 0.298760
## as.factor(subject)114 -1.514897   0.997265  -1.519 0.128842
## as.factor(subject)115 -0.903517   0.956924  -0.944 0.345140
## as.factor(subject)116 -0.471245   0.996988  -0.473 0.636480
## as.factor(subject)117 -2.469312   1.100349  -2.244 0.024889 *
## as.factor(subject)118 -1.382785   0.988213  -1.399 0.161821
## as.factor(subject)119 -1.942758   0.988197  -1.966 0.049384 *
## as.factor(subject)120 -0.469678   0.988473  -0.475 0.634707
## as.factor(subject)121 -0.287654   0.997070  -0.288 0.772982
## as.factor(subject)122 -1.278551   0.988168  -1.294 0.195801
## as.factor(subject)123 -1.041823   0.988176  -1.054 0.291825
## as.factor(subject)124 -1.607301   1.099488  -1.462 0.143871
## as.factor(subject)125  0.526582   0.988288   0.533 0.594191
## as.factor(subject)126  0.584657   0.997096   0.586 0.557673
## as.factor(subject)127 -1.944596   0.988226  -1.968 0.049176 *
## as.factor(subject)128 -0.842064   0.988143  -0.852 0.394180
```

```
## as.factor(subject)129  0.070822   0.988453   0.072 0.942885
## as.factor(subject)130 -1.581636   0.988448  -1.600 0.109664
## as.factor(subject)131 -0.472182   1.098177  -0.430 0.667245
## as.factor(subject)132 -0.986540   1.101207  -0.896 0.370384
## as.factor(subject)133 -0.416446   1.099579  -0.379 0.704910
## as.factor(subject)134  0.227990   0.997080   0.229 0.819149
## as.factor(subject)135 -0.740826   0.989104  -0.749 0.453917
## as.factor(subject)136 -1.590892   0.997325  -1.595 0.110769
## as.factor(subject)137 -1.022045   0.988184  -1.034 0.301085
## as.factor(subject)138 -0.014375   0.988863  -0.015 0.988403
## as.factor(subject)139 -0.545281   0.988147  -0.552 0.581107
## as.factor(subject)140 -2.606882   0.988131  -2.638 0.008373 **
## as.factor(subject)141 -0.358488   0.988336  -0.363 0.716837
## as.factor(subject)142 -1.763546   0.988398  -1.784 0.074472 .
## as.factor(subject)143 -0.902573   0.988573  -0.913 0.361304
## as.factor(subject)144 -1.932975   0.988132  -1.956 0.050524 .
## as.factor(subject)145 -1.223482   1.132115  -1.081 0.279905
## as.factor(subject)146 -2.188235   1.007683  -2.172 0.029958 *
## as.factor(subject)147 -1.157029   0.988226  -1.171 0.241756
## as.factor(subject)148 -1.952371   0.988196  -1.976 0.048270 *
## as.factor(subject)149 -1.516419   1.099540  -1.379 0.167942
## as.factor(subject)150 -1.534385   1.007886  -1.522 0.128007
## as.factor(subject)151 -0.792136   0.988237  -0.802 0.422861
## as.factor(subject)152 -1.787231   0.988253  -1.808 0.070620 .
## as.factor(subject)153 -0.905974   0.988224  -0.917 0.359328
## as.factor(subject)154  0.106067   0.988604   0.107 0.914566
## as.factor(subject)155 -0.991065   0.988181  -1.003 0.315971
## as.factor(subject)156 -1.696639   0.859763  -1.973 0.048533 *
## as.factor(subject)157 -1.710858   0.997426  -1.715 0.086386 .
## as.factor(subject)158 -1.172293   0.988148  -1.186 0.235565
## as.factor(subject)159 -1.711094   0.988129  -1.732 0.083426 .
## as.factor(subject)160 -1.066368   0.988176  -1.079 0.280607
## as.factor(subject)161 -1.866392   0.988309  -1.888 0.059048 .
## as.factor(subject)162 -3.366891   0.988210  -3.407 0.000664 ***
## as.factor(subject)163 -0.282836   0.988430  -0.286 0.774784
## as.factor(subject)164 -1.471321   1.105142  -1.331 0.183166
## as.factor(subject)165 -1.583624   0.988221  -1.603 0.109138
## as.factor(subject)166 -0.548888   0.997053  -0.551 0.582005
## as.factor(subject)167  1.292194   0.988447   1.307 0.191200
## as.factor(subject)168 -2.144536   0.988215  -2.170 0.030067 *
## as.factor(subject)169 -3.799834   0.988161  -3.845 0.000123 ***
## as.factor(subject)170 -1.682601   0.988132  -1.703 0.088695 .
## as.factor(subject)171 -1.381184   0.988189  -1.398 0.162296
## as.factor(subject)172 -0.722762   0.989974  -0.730 0.465390
## as.factor(subject)173 -1.245912   0.988137  -1.261 0.207442
## as.factor(subject)174 -0.562222   0.956973  -0.588 0.556906
## as.factor(subject)175 -0.885018   0.997263  -0.887 0.374901
## as.factor(subject)176 -0.847920   0.988408  -0.858 0.391027
## as.factor(subject)177  0.801403   1.133040   0.707 0.479426
## as.factor(subject)178 -1.481422   0.988163  -1.499 0.133922
## as.factor(subject)179 -0.905902   0.988352  -0.917 0.359428
## as.factor(subject)180 -0.711279   0.988420  -0.720 0.471813
## as.factor(subject)181 -0.383763   1.007387  -0.381 0.703265
## as.factor(subject)182 -1.150412   0.988291  -1.164 0.244489
```

```
## as.factor(subject)183 -0.471984   0.988156  -0.478 0.632937
## as.factor(subject)184 -1.283038   1.097994  -1.169 0.242675
## as.factor(subject)185 -1.173450   0.997175  -1.177 0.239368
## as.factor(subject)186 -1.313459   0.988191  -1.329 0.183886
## as.factor(subject)187 -1.321169   1.098235  -1.203 0.229062
## as.factor(subject)188  0.317481   0.988460   0.321 0.748088
## as.factor(subject)189 -0.239846   0.988152  -0.243 0.808236
## as.factor(subject)190 -1.905105   0.996041  -1.913 0.055873 .
## as.factor(subject)191 -2.196961   0.997194  -2.203 0.027651 *
## as.factor(subject)192 -0.721211   0.988184  -0.730 0.465541
## as.factor(subject)193 -2.345493   0.988176  -2.374 0.017673 *
## as.factor(subject)194 -0.597729   0.988620  -0.605 0.545478
## as.factor(subject)195  0.052548   0.988146   0.053 0.957593
## as.factor(subject)196 -0.609260   0.988304  -0.616 0.537625
## as.factor(subject)197  0.385461   1.099437   0.351 0.725911
## as.factor(subject)198 -1.376675   1.097796  -1.254 0.209915
## as.factor(subject)199  0.418831   0.957995   0.437 0.661997
## as.factor(subject)200 -0.302832   0.988277  -0.306 0.759300
## as.factor(subject)201  0.109719   0.988139   0.111 0.911594
## as.factor(subject)202 -1.192373   0.956901  -1.246 0.212821
## as.factor(subject)203 -1.210903   0.988161  -1.225 0.220505
## as.factor(subject)205  0.324223   0.990791   0.327 0.743509
## as.factor(subject)206 -1.168536   0.988333  -1.182 0.237157
## as.factor(subject)207 -1.345418   0.988164  -1.362 0.173435
## as.factor(subject)208 -1.436599   1.099404  -1.307 0.191400
## as.factor(subject)209 -0.949035   1.098375  -0.864 0.387629
## as.factor(subject)210 -0.200444   0.988373  -0.203 0.839302
## as.factor(subject)211 -1.341305   0.988294  -1.357 0.174810
## as.factor(subject)212 -0.062383   1.007418  -0.062 0.950627
## as.factor(subject)213 -0.073359   0.988164  -0.074 0.940826
## as.factor(subject)214 -1.106654   0.988232  -1.120 0.262864
## as.factor(subject)215 -0.875712   0.989307  -0.885 0.376123
## as.factor(subject)216 -1.813799   0.988139  -1.836 0.066508 .
## as.factor(subject)217 -1.416468   0.988365  -1.433 0.151909
## as.factor(subject)218 -0.861715   0.997205  -0.864 0.387577
## as.factor(subject)219 -1.772938   0.988134  -1.794 0.072865 .
## as.factor(subject)220 -0.282948   0.958406  -0.295 0.767838
## as.factor(subject)221 -1.479867   1.097816  -1.348 0.177745
## as.factor(subject)222 -0.825702   0.988231  -0.836 0.403475
## as.factor(subject)223 -0.171354   0.957134  -0.179 0.857926
## as.factor(subject)224  0.385198   0.988390   0.390 0.696766
## as.factor(subject)225 -0.126055   0.989049  -0.127 0.898592
## as.factor(subject)226 -1.257840   1.099427  -1.144 0.252668
## as.factor(subject)227  0.085265   0.989104   0.086 0.931309
## as.factor(subject)228  0.370724   0.957595   0.387 0.698677
## as.factor(subject)229  1.323472   0.956898   1.383 0.166729
## as.factor(subject)231  0.502430   1.072335   0.469 0.639430
## as.factor(subject)232 -0.861195   0.997178  -0.864 0.387851
## as.factor(subject)233 -1.137552   1.099624  -1.034 0.300980
## as.factor(subject)234 -0.159389   0.997480  -0.160 0.873054
## as.factor(subject)235  0.089255   1.099579   0.081 0.935310
## as.factor(subject)236  0.107856   1.132100   0.095 0.924105
## as.factor(subject)237  0.153482   0.988215   0.155 0.876584
## as.factor(subject)238 -0.454421   0.956910  -0.475 0.634900
```

```
## as.factor(subject)239  0.930993   1.005119   0.926 0.354381
## as.factor(subject)240  2.307737   0.996091   2.317 0.020574 *
## as.factor(subject)241 -0.052640   0.956912  -0.055 0.956133
## as.factor(subject)242 -0.036312   0.988335  -0.037 0.970694
## as.factor(subject)243 -1.452735   0.988269  -1.470 0.141660
## as.factor(subject)244  0.532866   1.098169   0.485 0.627544
## as.factor(subject)245 -0.775529   1.133963  -0.684 0.494078
## as.factor(subject)246 -2.227485   0.988213  -2.254 0.024256 *
## as.factor(subject)247 -1.904108   0.988244  -1.927 0.054092 .
## as.factor(subject)248 -0.408692   0.988762  -0.413 0.679386
## as.factor(subject)249  0.497537   0.997132   0.499 0.617834
## as.factor(subject)250 -0.657830   0.988133  -0.666 0.505629
## as.factor(subject)251 -0.712002   0.989919  -0.719 0.472035
## as.factor(subject)252  0.492355   0.988215   0.498 0.618357
## as.factor(subject)253 -0.896011   0.988185  -0.907 0.364617
## as.factor(subject)254  0.876970   0.988128   0.888 0.374869
## as.factor(subject)255 -0.693172   0.988132  -0.701 0.483040
## as.factor(subject)256 -1.229771   1.132105  -1.086 0.277437
## as.factor(subject)257 -0.542697   0.988414  -0.549 0.583001
## as.factor(subject)258 -0.352610   0.997459  -0.354 0.723729
## as.factor(subject)259 -1.839504   0.988156  -1.862 0.062752 .
## as.factor(subject)260  0.392222   0.988711   0.397 0.691613
## as.factor(subject)261 -2.351155   0.988133  -2.379 0.017396 *
## as.factor(subject)262 -0.193688   0.988698  -0.196 0.844698
## as.factor(subject)263  0.356007   1.099488   0.324 0.746114
## as.factor(subject)265 -0.347907   1.019339  -0.341 0.732894
## as.factor(subject)266 -1.075176   0.997525  -1.078 0.281180
## as.factor(subject)267  0.262197   1.099650   0.238 0.811556
## as.factor(subject)268 -0.674926   0.996952  -0.677 0.498459
## as.factor(subject)269 -0.207153   0.988415  -0.210 0.834007
## as.factor(subject)270  0.064866   1.033605   0.063 0.949963
## as.factor(subject)271 -0.578009   0.988143  -0.585 0.558623
## as.factor(subject)272 -1.133680   0.996250  -1.138 0.255222
## as.factor(subject)273 -0.564086   0.988573  -0.571 0.568304
## as.factor(subject)274  1.711444   0.958233   1.786 0.074181 .
## as.factor(subject)275  0.358836   0.957286   0.375 0.707797
## as.factor(subject)276 -1.192964   0.957642  -1.246 0.212949
## as.factor(subject)277 -1.507463   1.916086  -0.787 0.431488
## as.factor(subject)278  0.148135   0.957119   0.155 0.877011
## as.factor(subject)279  1.488271   0.957259   1.555 0.120105
## as.factor(subject)280 -0.048560   0.956876  -0.051 0.959529
## as.factor(subject)281  0.151966   0.957958   0.159 0.873965
## as.factor(subject)282  2.236712   0.956936   2.337 0.019478 *
## as.factor(subject)283  0.781069   0.957386   0.816 0.414651
## as.factor(subject)284  0.298102   0.957286   0.311 0.755513
## as.factor(subject)285  0.954101   0.956880   0.997 0.318788
## as.factor(subject)286 -0.049957   0.957115  -0.052 0.958376
## as.factor(subject)287 -0.026295   0.959085  -0.027 0.978129
## as.factor(subject)288  1.575999   0.957342   1.646 0.099810 .
## as.factor(subject)289  0.928078   0.956971   0.970 0.332211
## as.factor(subject)290 -1.307082   0.996376  -1.312 0.189664
## as.factor(subject)291  1.654986   0.956982   1.729 0.083831 .
## as.factor(subject)292  0.301966   0.957013   0.316 0.752378
## as.factor(subject)293  0.884478   0.957446   0.924 0.355661
```

```
## as.factor(subject)294 -0.207955   0.996033  -0.209 0.834630
## as.factor(subject)295  1.981271   0.957122   2.070 0.038524 *
## as.factor(subject)296  0.721818   0.956929   0.754 0.450717
## as.factor(subject)297 -0.662695   0.956936  -0.693 0.488660
## as.factor(subject)298  0.362529   0.957139   0.379 0.704887
## as.factor(subject)299 -0.915182   0.956988  -0.956 0.338981
## as.factor(subject)300  1.642163   0.956981   1.716 0.086256 .
## as.factor(subject)301 -0.864532   0.956901  -0.903 0.366340
## as.factor(subject)302  1.507589   0.957075   1.575 0.115302
## as.factor(subject)303 -0.465255   0.958830  -0.485 0.627543
## as.factor(subject)304  0.950283   0.959315   0.991 0.321959
## as.factor(subject)305 -0.361687   0.957232  -0.378 0.705568
## as.factor(subject)307  0.732473   0.957263   0.765 0.444221
## as.factor(subject)308  0.345548   0.957447   0.361 0.718193
## as.factor(subject)309  0.127240   0.956904   0.133 0.894225
## as.factor(subject)310 -0.017966   0.957088  -0.019 0.985025
## as.factor(subject)311 -1.112947   0.956920  -1.163 0.244890
## as.factor(subject)312 -1.615469   0.957029  -1.688 0.091502 .
## as.factor(subject)313 -1.041892   0.957054  -1.089 0.276388
## as.factor(subject)314 -1.047178   0.957013  -1.094 0.273938
## as.factor(subject)315 -0.237852   0.957286  -0.248 0.803790
## as.factor(subject)316 -0.382272   0.956946  -0.399 0.689571
## as.factor(subject)317  0.832070   0.957612   0.869 0.384962
## as.factor(subject)318  0.043203   0.957015   0.045 0.963995
## as.factor(subject)319  0.191520   0.956876   0.200 0.841374
## as.factor(subject)320 -0.224569   0.957474  -0.235 0.814577
## as.factor(subject)321 -0.750222   0.956898  -0.784 0.433086
## as.factor(subject)322 -0.697811   0.956920  -0.729 0.465913
## as.factor(subject)323  0.049894   0.957697   0.052 0.958454
## as.factor(subject)324  0.607976   0.996512   0.610 0.541834
## as.factor(subject)325  0.437814   0.996436   0.439 0.660414
## as.factor(subject)326  0.093875   0.957115   0.098 0.921874
## as.factor(subject)327 -1.021442   0.957102  -1.067 0.285946
## as.factor(subject)328 -0.167033   0.996361  -0.168 0.866874
## as.factor(subject)329 -2.557968   0.956901  -2.673 0.007549 **
## as.factor(subject)330 -1.326439   0.956904  -1.386 0.165783
## as.factor(subject)331  0.146632   0.957538   0.153 0.878301
## as.factor(subject)332 -0.728882   0.957102  -0.762 0.446381
## as.factor(subject)333 -0.593509   0.957301  -0.620 0.535311
## as.factor(subject)334  2.820324   0.957158   2.947 0.003235 **
## as.factor(subject)335  0.513782   0.957115   0.537 0.591439
## as.factor(subject)336 -0.579702   0.996545  -0.582 0.560799
## as.factor(subject)337  0.747801   0.956898   0.781 0.434571
## as.factor(subject)338  0.142850   0.957242   0.149 0.881380
## as.factor(subject)339  0.702266   0.957528   0.733 0.463355
## as.factor(subject)340  0.960219   0.957428   1.003 0.315973
## as.factor(subject)341  0.538833   0.957678   0.563 0.573713
## as.factor(subject)342 -0.124230   0.956898  -0.130 0.896712
## as.factor(subject)344  1.701224   0.961140   1.770 0.076815 .
## as.factor(subject)345  1.160672   0.956988   1.213 0.225275
## as.factor(subject)346 -2.217805   0.996091  -2.227 0.026045 *
## as.factor(subject)347 -1.946807   0.956904  -2.034 0.041980 *
## as.factor(subject)348 -1.049744   0.956909  -1.097 0.272712
## as.factor(subject)349 -1.980837   0.956899  -2.070 0.038522 *
```

```
## as.factor(subject)350 -0.951593   0.996154  -0.955 0.339510
## as.factor(subject)351 -0.913644   0.957342  -0.954 0.339971
## as.factor(subject)352 -0.558036   0.957122  -0.583 0.559908
## as.factor(subject)353 -0.174731   0.957134  -0.183 0.855157
## as.factor(subject)354  1.509774   0.957119   1.577 0.114793
## as.factor(subject)355 -0.071630   0.957953  -0.075 0.940399
## as.factor(subject)356 -0.669822   0.956950  -0.700 0.484003
## as.factor(subject)357 -0.133209   0.956982  -0.139 0.889302
## as.factor(subject)358  0.069922   0.956936   0.073 0.941756
## as.factor(subject)359 -0.676442   0.957040  -0.707 0.479735
## as.factor(subject)360 -0.077056   0.957892  -0.080 0.935890
## as.factor(subject)361 -0.891198   0.996072  -0.895 0.371004
## as.factor(subject)362 -0.523255   0.956892  -0.547 0.584533
## as.factor(subject)363  0.099607   0.957120   0.104 0.917120
## as.factor(subject)364 -0.058195   0.956973  -0.061 0.951512
## as.factor(subject)365  0.736760   0.957063   0.770 0.441464
## as.factor(subject)366  0.520822   0.957489   0.544 0.586514
## as.factor(subject)367  1.349790   0.998375   1.352 0.176469
## as.factor(subject)368 -0.774901   0.828820  -0.935 0.349883
## as.factor(subject)369  0.002593   0.957278   0.003 0.997839
## as.factor(subject)370  1.836198   0.958987   1.915 0.055611 .
## as.factor(subject)371 -0.753193   0.956999  -0.787 0.431315
## as.factor(subject)372  0.804158   0.958199   0.839 0.401394
## as.factor(subject)373  0.887734   0.957473   0.927 0.353907
## as.factor(subject)374 -0.268934   0.957953  -0.281 0.778928
## as.factor(subject)375  0.208980   0.957572   0.218 0.827255
## as.factor(subject)376 -0.199128   0.956982  -0.208 0.835179
## as.factor(subject)377  0.879030   0.957018   0.919 0.358417
## as.factor(subject)378 -0.222139   0.956880  -0.232 0.816436
## as.factor(subject)379 -1.299432   0.996032  -1.305 0.192114
## as.factor(subject)380  1.357501   0.957088   1.418 0.156175
## as.factor(subject)381  0.400960   0.956913   0.419 0.675232
## as.factor(subject)382  0.857190   0.996409   0.860 0.389695
## as.factor(subject)383 -0.177411   0.957159  -0.185 0.852964
## as.factor(subject)384 -0.402535   0.957811  -0.420 0.674318
## as.factor(subject)385  0.948024   0.958050   0.990 0.322472
## as.factor(subject)386 -0.778540   0.956975  -0.814 0.415964
## as.factor(subject)387  1.180249   0.958280   1.232 0.218171
## as.factor(subject)388  0.681249   0.957040   0.712 0.476619
## as.factor(subject)389  0.150081   0.957659   0.157 0.875478
## as.factor(subject)390  0.249345   0.957243   0.260 0.794507
## as.factor(subject)391 -0.015622   0.957191  -0.016 0.986980
## as.factor(subject)392 -0.804340   0.996039  -0.808 0.419412
## as.factor(subject)393  0.060724   0.956936   0.063 0.949406
## as.factor(subject)394 -0.386415   0.956971  -0.404 0.686392
## as.factor(subject)395  0.160242   0.957556   0.167 0.867108
## as.factor(subject)396 -0.724260   0.957174  -0.757 0.449303
## as.factor(subject)397 -0.329523   0.957401  -0.344 0.730729
## as.factor(subject)398  0.725986   0.956999   0.759 0.448140
## as.factor(subject)399  0.028982   0.957489   0.030 0.975855
## as.factor(subject)400  1.930075   0.957695   2.015 0.043948 *
## as.factor(subject)401  0.887714   0.957059   0.928 0.353710
## as.factor(subject)402  1.241289   0.957446   1.296 0.194905
## as.factor(subject)403 -1.206227   0.956892  -1.261 0.207551
```

```
## as.factor(subject)404  0.450078   0.957383   0.470 0.638305
## as.factor(subject)405 -1.443084   0.957014  -1.508 0.131672
## as.factor(subject)406 -2.362381   0.956880  -2.469 0.013604 *
## as.factor(subject)407  0.603863   0.956924   0.631 0.528053
## as.factor(subject)408 -0.852988   0.996436  -0.856 0.392036
## as.factor(subject)409  0.263665   0.956911   0.276 0.782920
## as.factor(subject)410  2.305373   0.957014   2.409 0.016052 *
## as.factor(subject)411  0.594973   0.957871   0.621 0.534549
## as.factor(subject)412  0.199542   0.956920   0.209 0.834831
## as.factor(subject)413 -1.165582   0.957174  -1.218 0.223410
## as.factor(subject)414 -0.789898   0.957120  -0.825 0.409267
## as.factor(subject)415  0.598928   0.964020   0.621 0.534456
## as.factor(subject)417  0.208054   0.996292   0.209 0.834595
## as.factor(subject)418  1.679770   0.957127   1.755 0.079347 .
## as.factor(subject)419  0.507114   0.996083   0.509 0.610710
## as.factor(subject)420  0.361894   0.996581   0.363 0.716526
## as.factor(subject)422  1.048694   0.958031   1.095 0.273754
## as.factor(subject)423  0.269421   0.957995   0.281 0.778547
## as.factor(subject)424  0.946807   0.958938   0.987 0.323541
## as.factor(subject)425  2.101337   0.996239   2.109 0.034994 *
## as.factor(subject)426  1.000729   0.959843   1.043 0.297209
## as.factor(subject)427 -0.181123   0.996157  -0.182 0.855734
## as.factor(subject)428 -0.354796   1.048415  -0.338 0.735074
## as.factor(subject)429 -0.287695   0.956901  -0.301 0.763697
## as.factor(subject)430  0.226280   0.957018   0.236 0.813103
## as.factor(subject)431 -0.452388   0.957020  -0.473 0.636454
## as.factor(subject)432  2.231121   0.965198   2.312 0.020861 *
## as.factor(subject)433 -0.597554   0.996194  -0.600 0.548655
## as.factor(subject)434  0.705675   0.958288   0.736 0.461543
## as.factor(subject)436 -0.645577   1.133626  -0.569 0.569068
## as.factor(subject)437 -1.105897   1.133996  -0.975 0.329519
## as.factor(subject)438 -0.040188   1.179944  -0.034 0.972832
## as.factor(subject)439 -0.702371   1.133673  -0.620 0.535593
## as.factor(subject)440  0.185404   1.102505   0.168 0.866462
## as.factor(subject)441 -3.233693   1.099649  -2.941 0.003297 **
## as.factor(subject)442 -1.331540   0.988267  -1.347 0.177957
## as.factor(subject)443  0.829679   0.957052   0.867 0.386052
## as.factor(subject)444 -1.004919   0.988160  -1.017 0.309245
## as.factor(subject)445 -1.685255   0.988132  -1.705 0.088193 .
## as.factor(subject)446  0.278551   1.101524   0.253 0.800378
## as.factor(subject)447 -1.175897   0.997466  -1.179 0.238526
## as.factor(subject)448  0.614941   0.993586   0.619 0.536016
## as.factor(subject)449 -0.920717   0.988357  -0.932 0.351628
## as.factor(subject)450  0.948174   1.099663   0.862 0.388615
## as.factor(subject)451 -1.821283   1.099516  -1.656 0.097725 .
## as.factor(subject)452 -1.683011   0.988171  -1.703 0.088630 .
## as.factor(subject)453 -0.700822   1.100342  -0.637 0.524224
## as.factor(subject)454 -0.989531   0.988158  -1.001 0.316709
## as.factor(subject)455 -1.987300   1.098030  -1.810 0.070403 .
## as.factor(subject)456 -3.320314   1.097755  -3.025 0.002508 **
## as.factor(subject)457 -0.798262   1.099439  -0.726 0.467850
## as.factor(subject)458  0.187459   1.099478   0.170 0.864628
## as.factor(subject)459  1.439027   0.988969   1.455 0.145740
## as.factor(subject)460  0.962184   0.988323   0.974 0.330348
```

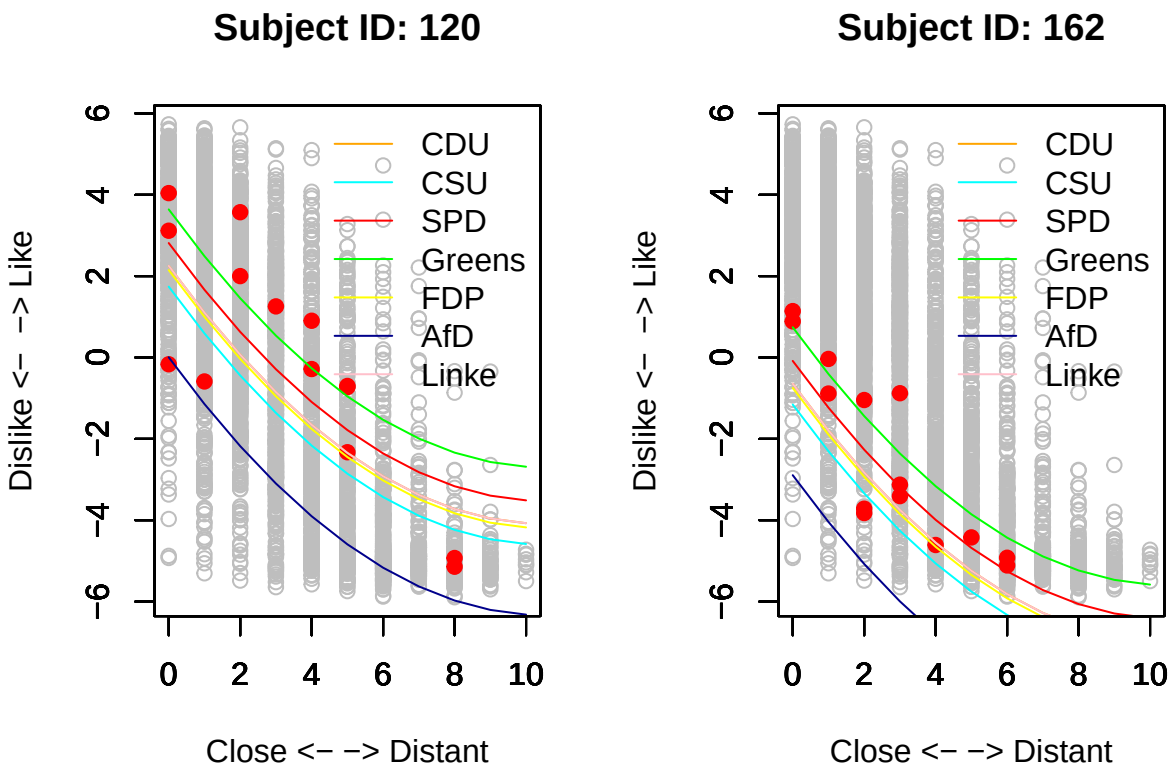
```
## as.factor(subject)461 -0.417927   0.988620  -0.423 0.672513
## as.factor(subject)462 -1.160304   1.097774  -1.057 0.290604
## as.factor(subject)463 -0.633434   0.988731  -0.641 0.521791
## as.factor(subject)464 -1.159843   0.988179  -1.174 0.240590
## as.factor(subject)465 -1.150672   0.871358  -1.321 0.186740
## as.factor(subject)466  0.307776   0.988253   0.311 0.755490
## as.factor(subject)467 -1.512829   0.988399  -1.531 0.125965
## as.factor(subject)468 -0.662299   1.134023  -0.584 0.559241
## as.factor(subject)469 -0.009256   0.957628  -0.010 0.992289
## as.factor(subject)470  1.110966   0.988175   1.124 0.260982
## as.factor(subject)471 -1.045729   1.099525  -0.951 0.341634
## as.factor(subject)472 -1.951524   1.132306  -1.723 0.084889 .
## as.factor(subject)474 -1.385067   1.099690  -1.260 0.207933
## as.factor(subject)475  0.402226   1.097810   0.366 0.714097
## as.factor(subject)476 -1.155647   0.988368  -1.169 0.242385
## as.factor(subject)477 -3.846417   1.097897  -3.503 0.000465 ***
## as.factor(subject)478 -0.552081   0.957339  -0.577 0.564192
## as.factor(subject)479 -0.508445   0.957069  -0.531 0.595278
## as.factor(subject)480  0.567182   0.960359   0.591 0.554832
## as.factor(subject)481  0.349523   0.956910   0.365 0.714938
## as.factor(subject)482 -0.880015   1.101203  -0.799 0.424265
## as.factor(subject)483 -1.318205   1.097769  -1.201 0.229911
## as.factor(subject)484 -0.638307   0.997271  -0.640 0.522180
## as.factor(subject)485  0.005920   1.099789   0.005 0.995706
## as.factor(subject)486 -0.633234   1.099816  -0.576 0.564813
## as.factor(subject)487 -0.137629   1.132104  -0.122 0.903248
## as.factor(subject)488 -1.238608   0.988147  -1.253 0.210122
## as.factor(subject)489 -0.266141   0.957064  -0.278 0.780967
## as.factor(subject)490 -1.160052   0.988513  -1.174 0.240664
## as.factor(subject)491 -3.449631   1.099440  -3.138 0.001718 **
## as.factor(subject)492 -1.133564   1.097869  -1.033 0.301905
## as.factor(subject)493 -0.793244   0.956982  -0.829 0.407218
## as.factor(subject)494  0.232583   0.957068   0.243 0.808007
## as.factor(subject)495 -1.784559   1.099568  -1.623 0.104689
## as.factor(subject)496 -4.833781   1.097758  -4.403 1.10e-05 ***
## as.factor(subject)498 -2.107040   0.956891  -2.202 0.027734 *
## as.factor(subject)499 -1.060190   0.957263  -1.108 0.268146
## as.factor(subject)500 -0.286543   0.997736  -0.287 0.773982
## as.factor(subject)501 -1.382307   1.133687  -1.219 0.222814
## as.factor(subject)502 -3.307539   1.097779  -3.013 0.002606 **
## as.factor(subject)503 -3.042848   1.134074  -2.683 0.007329 **
## as.factor(subject)504  1.539786   0.988416   1.558 0.119366
## as.factor(subject)505 -0.460385   0.988484  -0.466 0.641425
## as.factor(subject)506 -0.657571   1.099457  -0.598 0.549821
## as.factor(subject)507 -1.433421   0.907444  -1.580 0.114285
## as.factor(subject)508 -0.645382   0.956982  -0.674 0.500107
## as.factor(subject)509 -0.302300   1.099513  -0.275 0.783379
## as.factor(subject)510  0.285919   0.997103   0.287 0.774321
## as.factor(subject)511 -1.841308   1.133984  -1.624 0.104521
## as.factor(subject)512  0.267482   0.956924   0.280 0.779861
## as.factor(subject)513 -0.490766   0.957032  -0.513 0.608124
## as.factor(subject)514  1.039275   0.997320   1.042 0.297454
## as.factor(subject)515 -0.421367   0.957242  -0.440 0.659828
## as.factor(subject)516 -1.819502   1.099627  -1.655 0.098086 .
```

```
## as.factor(subject)517 -1.489795   0.988170  -1.508 0.131742
## as.factor(subject)518 -1.127739   0.956929  -1.178 0.238680
## as.factor(subject)527  0.162769   0.996353   0.163 0.870241
## as.factor(subject)529 -0.820087   0.956912  -0.857 0.391498
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.79 on 3415 degrees of freedom
## Multiple R-squared:  0.7244, Adjusted R-squared:  0.6908
## F-statistic: 21.53 on 417 and 3415 DF,  p-value: < 2.2e-16
```

Since we have 'r length(unique(long.data$subject))' respondents, there are so many dummy variables and their coefficients. To get intuition, we can check two respondents (subject.id==120 and 162) in the graphical presentation:

```
par(mfrow=c(1,2))
for (i.subj in c(120,162)){

  this.fe <-coef(lm.out.dummy.fe)[grep(as.character(i.subj),
                                       names(coef(lm.out.dummy.fe)))]

  plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Close <- -> Distant",ylab="Dislike <-  -> Like",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T),col="grey",
     main=paste("Subject ID:",i.subj))
  par(new=T)
  plot(long.data$skalo.true[long.data$subject==i.subj] ~
         long.data$lr.dist.abs[long.data$subject==i.subj],
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T),
     col="red",pch=19)

predicted.cdu <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10)+ coef(lm.out.dummy)[3]* (c(0:10)
predicted.csu <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.spd <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.gru <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.fdp <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.afd <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)
predicted.lin <- coef(lm.out.dummy)[1] + coef(lm.out.dummy)[2]*c(0:10) +coef(lm.out.dummy)[3]* (c(0:10)

par(new=T)
plot(predicted.cdu ~ c(0:10),col="orange",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.csu ~ c(0:10),col="cyan",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.spd ~ c(0:10),col="red",type="l",
     ann=F,axes=F,xlab="",ylab="",
```

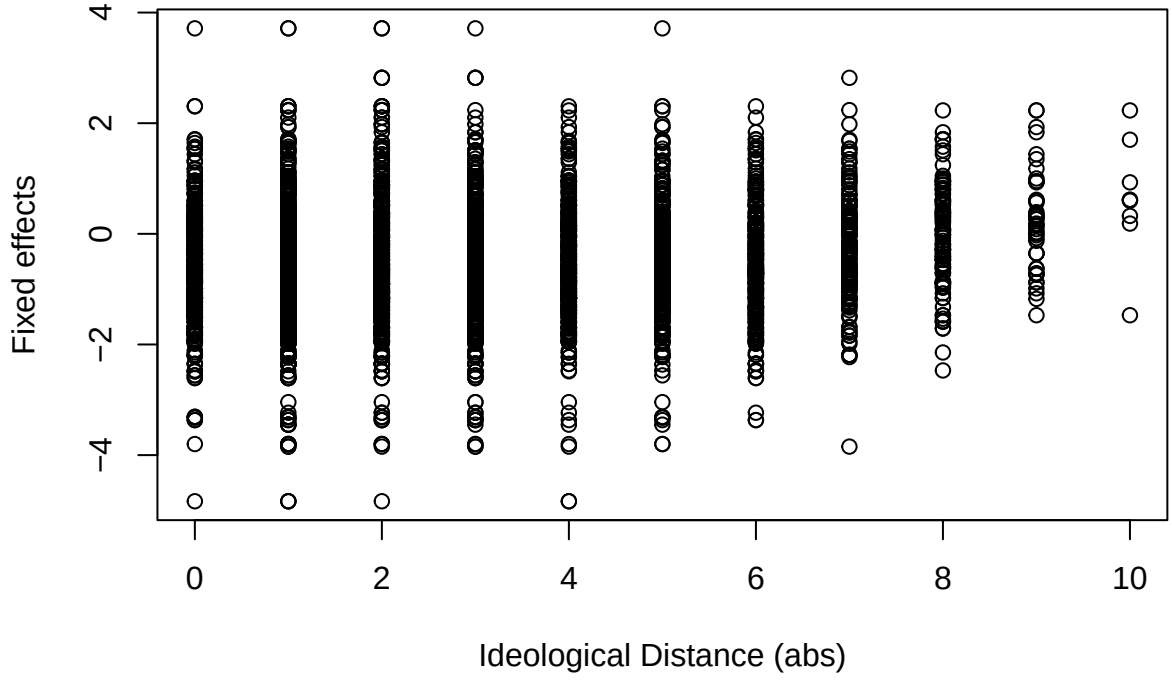
```
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.gru ~ c(0:10),col="green",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.fdp ~ c(0:10),col="yellow",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.afd ~ c(0:10),col="blue4",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.lin ~ c(0:10),col="pink",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))


legend("topright",col=c("orange","cyan","red","green","yellow","blue4","pink"),
       lty=c(rep(1,8)),c("CDU","CSU","SPD","Greens","FDP","AfD",
                         "Linke"),
       lwd=c(rep(1,8)),
       bty="n")
}
```



It is obvious that both respondents use the scale for like/dislike differently. Respondent 120 tend to evaluate parties more positively in general than Repondent 162.

One can also check the correlation between ideological distances as independent variable and the estimated fixed effects:

```
fixed.effects <- coef(lm.out.dummy.fe)[grep("subject",
                                        names(coef(lm.out.dummy.fe)))]
fixed.effects <- c(coef(lm.out.dummy.fe)[1],fixed.effects)
names(fixed.effects) <- unique(long.data$subject)

o <- match(long.data$subject,names(fixed.effects))

long.data$fixed.effects <- fixed.effects[o]


plot(long.data$fixed.effects ~ long.data$lr.dist.abs,
     ylab="Fixed effects",xlab="Ideological Distance (abs)")
```




```
cor.test(long.data$fixed.effects ,long.data$lr.dist.abs)
```

```
##
##  Pearson's product-moment correlation
##
## data:  long.data$fixed.effects and long.data$lr.dist.abs
## t = 7.3939, df = 3831, p-value = 1.742e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.08728307 0.14971201
## sample estimates:
##       cor
## 0.1186148
```

```
cor.test(long.data$fixed.effects ,long.data$lr.dist.sqr)
```

```
##
##  Pearson's product-moment correlation
##
## data:  long.data$fixed.effects and long.data$lr.dist.sqr
## t = 8.3866, df = 3831, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1030481 0.1652266
## sample estimates:
##       cor
## 0.1342695
```

The estimated positive correlation is also significant at 5% level. Therefore, the model without the fixed effect is likely to violate the important assumption of the linear regression model.

First differenced estimator

Another possibility to cope with the individual heterogeneity is to take the first-differences.

In our dataset, each of the respondent-party pairs may have two observations. For example, the first and eighth rows are Respondent 105's evaluation of CDU (party=1) at two different panel waves.

```
head(long.data[24:100,1:7],n=20)
```

```
##       id wave party skalo lr.self lr time
## 24 10511    1     1    -5      -2  1   40
## 25 10512    1     2    -5      -2  1   24
## 26 10513    1     3     2      -2 -2   67
## 27 10514    1     4     5      -2 -2   23
## 28 10515    1     5    -5      -2  0   55
## 29 10516    1     6    -5      -2  4   64
## 30 10517    1     7     0      -2 -4   34
## 31 10521    2     1    -1      -2  1   33
## 32 10522    2     2    -3      -2  2  244
## 33 10523    2     3     1      -2 -1   25
## 34 10524    2     4     4      -2 -2  211
## 35 10525    2     5    -3      -2  1   31
## 36 10526    2     6    -5      -2  4   19
## 37 10527    2     7     0      -2 -4   49
## 38 10611    1     1    -2      -2  2   57
## 39 10612    1     2    -4      -2  4   49
## 40 10613    1     3     3      -2 -2   35
## 41 10614    1     4     5      -2 -1  106
## 42 10615    1     5     2      -2  0   69
## 43 10616    1     6    -5      -2  5   41
```

We are now reoganizing the data which only include the adjacent respondent-party pairs and they are grouped in the next rows:

```
long.data$subject.party <- long.data$subject*10 + long.data$party

long.data <- long.data[order(long.data$subject.party),]

# keeping only adjacent data
freq.subject.party <- table(long.data$subject.party)
adjacent <- as.numeric(names(freq.subject.party[freq.subject.party>=2]))

adjacent.data <- long.data[long.data$subject.party %in% adjacent,]

head(adjacent.data[,1:7],n=20)
```

```
##       id wave party skalo lr.self lr time
## 15 10411    1     1     2       2  2   54
## 17 10421    2     1     2       2  1   26
## 16 10416    1     6     1       2  5   20
## 22 10426    2     6     1       2  3   38
## 24 10511    1     1    -5      -2  1   40
## 31 10521    2     1    -1      -2  1   33
## 25 10512    1     2    -5      -2  1   24
```

```
## 32 10522    2     2    -3      -2  2  244
## 26 10513    1     3     2      -2 -2   67
## 33 10523    2     3     1      -2 -1   25
## 27 10514    1     4     5      -2 -2   23
## 34 10524    2     4     4      -2 -2  211
## 28 10515    1     5    -5      -2  0   55
## 35 10525    2     5    -3      -2  1   31
## 29 10516    1     6    -5      -2  4   64
## 36 10526    2     6    -5      -2  4   19
## 30 10517    1     7     0      -2 -4   34
## 37 10527    2     7     0      -2 -4   49
## 38 10611    1     1    -2      -2  2   57
## 45 10621    2     1    -3      -3  1   20
```

From this data, we create the first differences

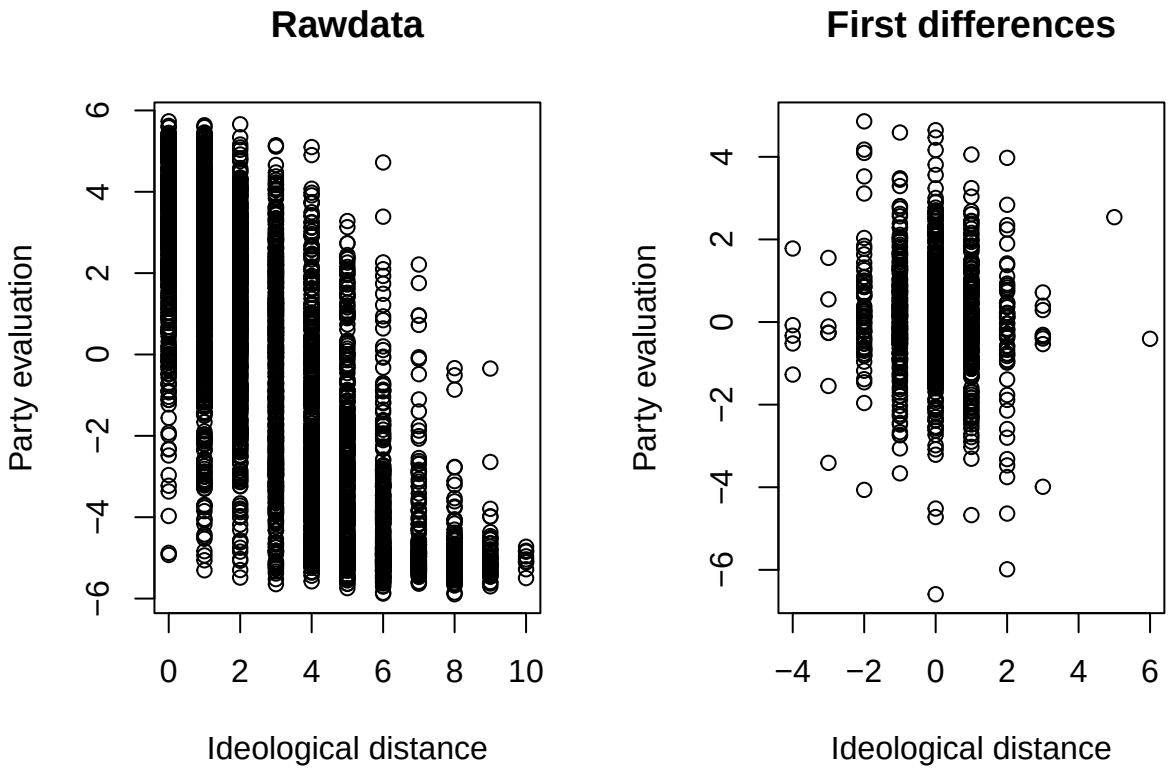
```
# creating a wide format data
unique.subject.party <- unique(long.data$subject.party)

diff.data <- NULL
for (i.wave in 1:2){
  dat.t1 <- adjacent.data[adjacent.data$wave==i.wave,
                    c("subject.party","skalo.true",
                      "lr.dist.abs","lr.dist.sqr","party")]
  dat.t2 <- adjacent.data[adjacent.data$wave==i.wave+1,
                    c("subject.party","skalo.true",
                      "lr.dist.abs","lr.dist.sqr","party")]

  o <- match(dat.t1$subject.party,dat.t2$subject.party)

  this.diff.data <- dat.t2[o,] - dat.t1
  this.diff.data$subject.party <- dat.t1$subject.party
  this.diff.data$party <- dat.t1$party
  this.diff.data$wave <- i.wave+1
  this.diff.data <- this.diff.data[!is.na(this.diff.data$skalo.true),]

  diff.data <- rbind(diff.data,
                    this.diff.data
                    )
}


summary(lm(skalo.true ~ lr.dist.abs +lr.dist.sqr + as.factor(party),data=diff.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs + lr.dist.sqr + as.factor(party),
##     data = diff.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.6495 -0.7099 -0.0326  0.7524  4.7473
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)        0.058658   0.108953   0.538   0.5904
## lr.dist.abs       -0.159964   0.076219  -2.099   0.0361 *
## lr.dist.sqr        0.005919   0.010482   0.565   0.5724
## as.factor(party)2 -0.161633   0.154988  -1.043   0.2973
## as.factor(party)3  0.140484   0.154179   0.911   0.3624
## as.factor(party)4 -0.204338   0.154140  -1.326   0.1852
## as.factor(party)5  0.245844   0.158874   1.547   0.1221
## as.factor(party)6  0.088338   0.156789   0.563   0.5733
## as.factor(party)7  0.086795   0.154304   0.562   0.5739
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.337 on 1017 degrees of freedom
## Multiple R-squared:  0.02188,    Adjusted R-squared:  0.01419
## F-statistic: 2.844 on 8 and 1017 DF,  p-value: 0.003962
```

Now, the coefficients of both ideological distance measures are very small and only the that of the absolute ideological distance measure is significant at 5% more. Here, however, we have to note that we sweeped out individual heterogeneity by first differencing. This strongly reduced the variance of both dependent and independent variables (see the figure below). Furthermore, our independent variables are certainly suffering from measurement errors. The first differencing makes this problem even more serious.

```
par(mfrow=c(1,2))

plot(long.data$skalo.true ~ long.data$lr.dist.abs,main="Rawdata",
     ylab="Party evaluation",xlab="Ideological distance")
plot(diff.data$skalo.true ~ diff.data$lr.dist.abs,main="First differences",
     ylab="Party evaluation",xlab="Ideological distance")
```



Censored models

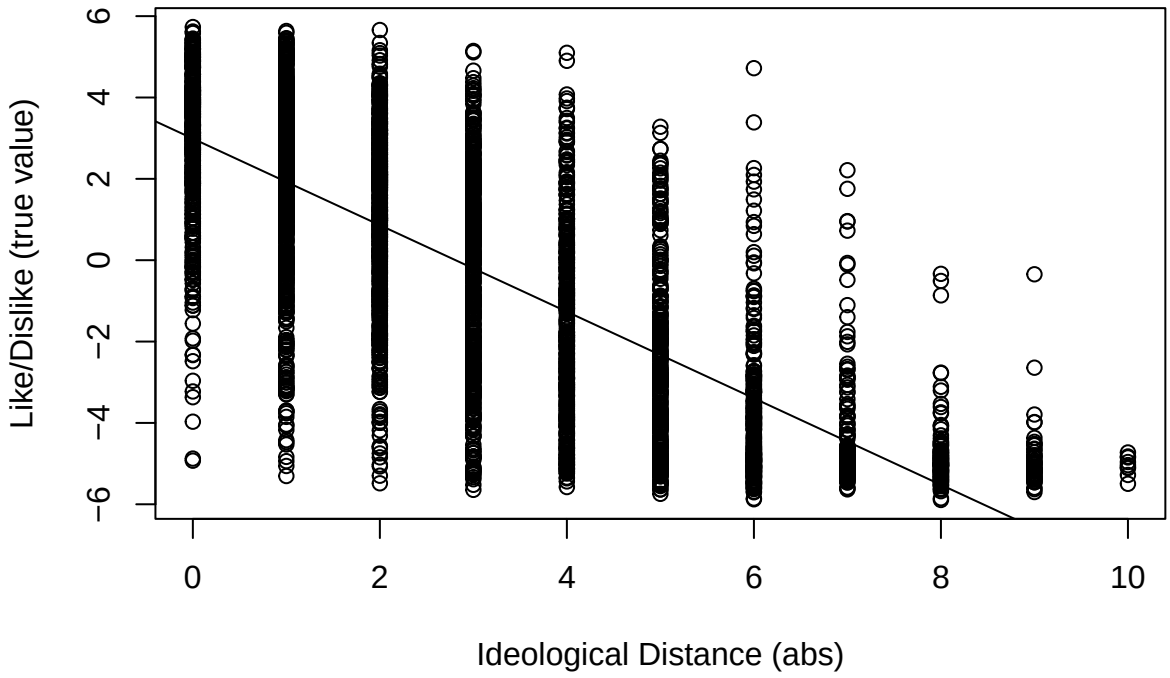
We have already estimated the following model for many times:

```
summary(lm.out.abs <- lm(skalo.true ~ lr.dist.abs,data=long.data))
```

```
##
## Call:
## lm(formula = skalo.true ~ lr.dist.abs, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9169 -1.4955  0.1624  1.4471  8.1145
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.98875    0.05586    53.5   <2e-16 ***
## lr.dist.abs -1.06374    0.01490   -71.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.109 on 3831 degrees of freedom
## Multiple R-squared:  0.571,  Adjusted R-squared:  0.5709
## F-statistic:  5098 on 1 and 3831 DF,  p-value: < 2.2e-16
```

```
plot(long.data$skalo.true ~ long.data$lr.dist.abs,
     xlab="Ideological Distance (abs)",
```

```
     ylab="Like/Dislike (true value)")
abline(lm.out.abs)
```

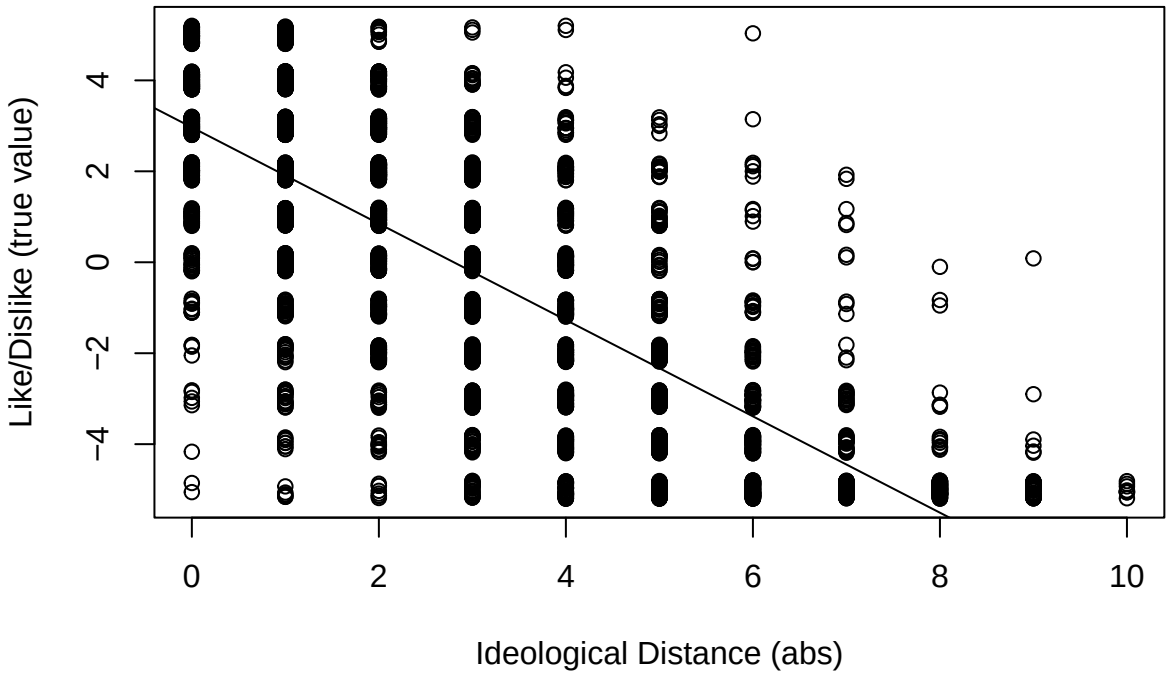


In the very early session, we created the dependent variable under the assumption of random measurement errors from a survey item. If we use the original survey variable, you will obtain the following results:

```
summary(lm.out.abs0 <- lm(skalo ~ lr.dist.abs,data=long.data))
```

```
##
## Call:
## lm(formula = skalo ~ lr.dist.abs, data = long.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9671 -1.6116  0.1514  1.2699  8.3884
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.96714    0.05552   53.44   <2e-16 ***
## lr.dist.abs -1.05926    0.01481  -71.54   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.096 on 3831 degrees of freedom
## Multiple R-squared:  0.5719, Adjusted R-squared:  0.5718
## F-statistic:  5118 on 1 and 3831 DF,  p-value: < 2.2e-16
```

```
plot(jitter(long.data$skalo) ~ long.data$lr.dist.abs,
     xlab="Ideological Distance (abs)",
     ylab="Like/Dislike (true value)")
abline(lm.out.abs0)
```



The original variable of party evaluation was measured by using a scale from -5 to +5. However, the predicted values for x=8 or larger are outside of the range.

Now we can conceive that the party evaluation measure is censored. That is, we assume that among those who rated a party with +5, there are some respondents who would have rated the party with +6 or more. And the same can be also assumed for those who rated a party with -5. This idea is behind the censored regression models:

```
library(censReg)
```

```
## Warning: Paket 'censReg' wurde unter R Version 4.1.3 erstellt
## Lade nötiges Paket: maxLik
## Warning: Paket 'maxLik' wurde unter R Version 4.1.3 erstellt
## Lade nötiges Paket: miscTools
## Warning: Paket 'miscTools' wurde unter R Version 4.1.3 erstellt
##
## Please cite the 'maxLik' package as:
## Henningsen, Arne and Toomet, Ott (2011). maxLik: A package for maximum likelihood estimation in R. C
##
## If you have questions, suggestions, or comments regarding the 'maxLik' package, please use a forum o
```

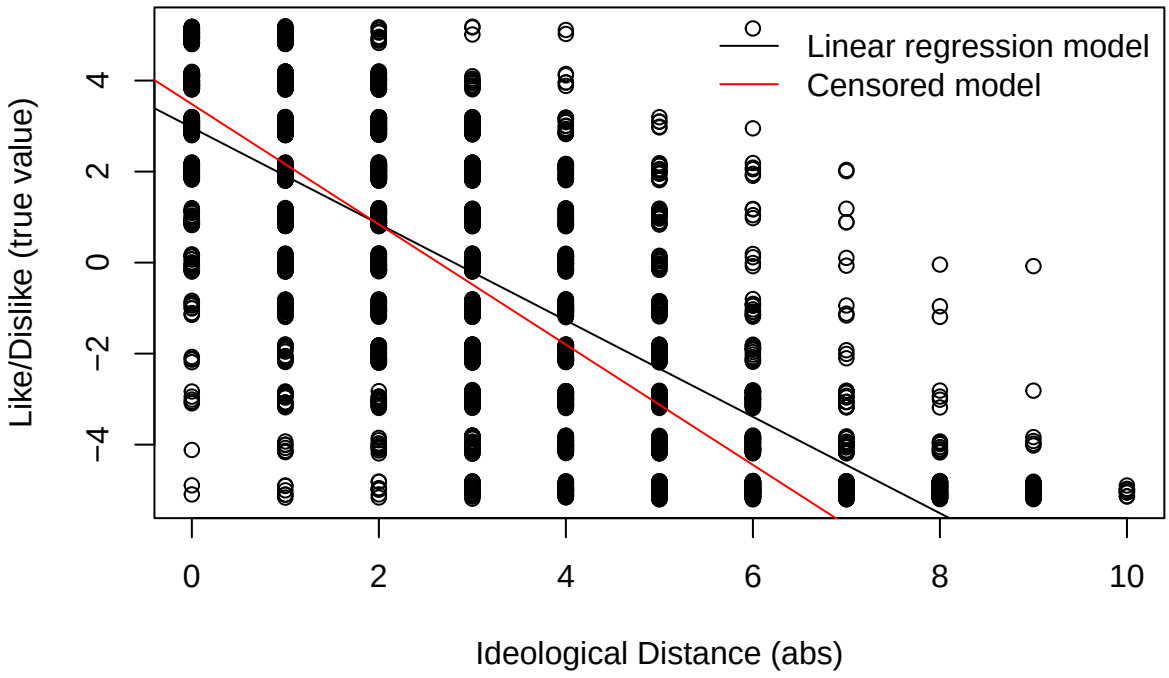
```
## https://r-forge.r-project.org/projects/maxlik/
##
## Please cite the 'censReg' package as:
## Henningsen, Arne (2017). censReg: Censored Regression (Tobit) Models. R package version 0.5. http://
##
## If you have questions, suggestions, or comments regarding the 'censReg' package, please use a forum
## https://r-forge.r-project.org/projects/sampleselection/
```

```
summary(cens.out.abs <- censReg(skalo ~ lr.dist.abs,
                                left=-5,right=5,
                                data=long.data))
```

```
##
## Call:
## censReg(formula = skalo ~ lr.dist.abs, left = -5, right = 5,
##     data = long.data)
##
## Observations:
##          Total  Left-censored     Uncensored Right-censored
##           3833            587           3093            153
##
## Coefficients:
##             Estimate Std. error t value Pr(> t)
## (Intercept)  3.48254    0.06744   51.64  <2e-16 ***
## lr.dist.abs -1.32118    0.01986  -66.53  <2e-16 ***
## logSigma     0.88399    0.01311   67.40  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Newton-Raphson maximisation, 5 iterations
## Return code 1: gradient close to zero (gradtol)
## Log-likelihood: -7718.408 on 3 Df
```

The result of the censored model can be compared with the simple linear regression model in the following figure:

```
plot(jitter(long.data$skalo) ~ long.data$lr.dist.abs,
     xlab="Ideological Distance (abs)",
     ylab="Like/Dislike (true value)")
abline(lm.out.abs0)
abline(coef=cens.out.abs$estimate[1:2],col="red")
legend("topright",lty=1,
       col=c("black","red"),
       legend=c("Linear regression model","Censored model"),
       bty="n")
```



The difference is in particular clear in the region with a large ideological distance (x>6). While the censored model assumes smaller values behind the measurement of y=-5, the linear regression model takes all the measurement value as face value. Therefore, the latter is more flat to minimize the residuals. In contrast, the slope of the censored model is steeper since the range of the dependent variable does not play a role.

We can now extend the model with the party-dummy variables to consider the heterogeneity among parties:

```
summary(cens.out.dummy <- censReg(skalo ~ lr.dist.abs+as.factor(party),
                                  left=-5,right=5,
                                  data=long.data))
```

```
##
## Call:
## censReg(formula = skalo ~ lr.dist.abs + as.factor(party), left = -5,
##     right = 5, data = long.data)
##
## Observations:
##          Total  Left-censored     Uncensored Right-censored
##           3833            587           3093            153
##
## Coefficients:
##                    Estimate Std. error t value  Pr(> t)
## (Intercept)         2.66054    0.11222  23.709  < 2e-16 ***
## lr.dist.abs        -0.96307    0.02226 -43.267  < 2e-16 ***
## as.factor(party)2 -0.42768    0.13561  -3.154 0.001612 **
## as.factor(party)3  0.51802    0.13619   3.804 0.000142 ***
## as.factor(party)4  1.50855    0.13754  10.968  < 2e-16 ***
```

```
## as.factor(party)5 -0.12022    0.13499  -0.891 0.373133
## as.factor(party)6 -4.08257    0.18494 -22.075  < 2e-16 ***
## as.factor(party)7 -0.09731    0.13378  -0.727 0.467000
## logSigma           0.79358    0.01301  61.014  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Newton-Raphson maximisation, 5 iterations
## Return code 8: successive function values within relative tolerance limit (reltol)
## Log-likelihood: -7285.874 on 9 Df
```

We can further include the quadratic function:

```
summary(cens.out.dummy.quad <- censReg(skalo ~ lr.dist.abs+
                                  lr.dist.sqr +
                                  as.factor(party),
                                left=-5,right=5,
                                data=long.data))
```

```
##
## Call:
## censReg(formula = skalo ~ lr.dist.abs + lr.dist.sqr + as.factor(party),
##     left = -5, right = 5, data = long.data)
##
## Observations:
##          Total  Left-censored     Uncensored Right-censored
##           3833            587           3093            153
##
## Coefficients:
##                     Estimate Std. error t value  Pr(> t)
## (Intercept)         2.725252   0.127395  21.392  < 2e-16 ***
## lr.dist.abs        -1.022759   0.059960 -17.057  < 2e-16 ***
## lr.dist.sqr         0.009153   0.008525   1.074 0.282995
## as.factor(party)2 -0.433356   0.135591  -3.196 0.001393 **
## as.factor(party)3  0.506663   0.136494   3.712 0.000206 ***
## as.factor(party)4  1.489528   0.138583  10.748  < 2e-16 ***
## as.factor(party)5 -0.120501   0.134879  -0.893 0.371643
## as.factor(party)6 -4.109909   0.186335 -22.057  < 2e-16 ***
## as.factor(party)7 -0.093428   0.133720  -0.699 0.484747
## logSigma            0.792813   0.013021  60.887  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Newton-Raphson maximisation, 5 iterations
## Return code 1: gradient close to zero (gradtol)
## Log-likelihood: -7285.301 on 10 Df
```

Interestingly, the coefficient of the quadratic distance is not significant (at 5%-level). If we compare this model with the last one in the log-likelihood, the improvement is very small and it is not significant at 5%-level.

This result is reasonable if we consider that the linear model with the quadratic term lead to a convex function. This convex function predicted higher values in party evaluation for those with larger ideological distances. This was an artifact due to the censored dependent variable.

Further, we can extend the previous model (without the quadratic distance) by including the fixed effects for respondents:

```
summary(cens.out.dummy.fe <- censReg(skalo ~ lr.dist.abs+as.factor(party)+
                                       as.factor(subject),
                                     left=-5,right=5,
                                     data=long.data))
```

```
##
## Call:
## censReg(formula = skalo ~ lr.dist.abs + as.factor(party) + as.factor(subject),
##     left = -5, right = 5, data = long.data)
##
## Observations:
##          Total  Left-censored     Uncensored Right-censored
##           3833            587           3093            153
##
## Coefficients:
##                          Estimate Std. error t value  Pr(> t)
## (Intercept)              2.866106   0.715947   4.003 6.25e-05 ***
## lr.dist.abs             -1.050773   0.021199 -49.566  < 2e-16 ***
## as.factor(party)2       -0.323844   0.115997  -2.792 0.005241 **
## as.factor(party)3        0.413375   0.116636   3.544 0.000394 ***
## as.factor(party)4        1.370361   0.117969  11.616  < 2e-16 ***
## as.factor(party)5       -0.113547   0.115140  -0.986 0.324049
## as.factor(party)6       -3.632724   0.166455 -21.824  < 2e-16 ***
## as.factor(party)7       -0.065541   0.114098  -0.574 0.565678
## as.factor(subject)102 -1.007253   1.005158  -1.002 0.316303
## as.factor(subject)104  1.150830   0.947792   1.214 0.224663
## as.factor(subject)105 -2.047462   0.900743  -2.273 0.023021 *
## as.factor(subject)106  0.122968   0.896698   0.137 0.890925
## as.factor(subject)107  2.031656   1.032508   1.968 0.049104 *
## as.factor(subject)109  0.251422   0.893596   0.281 0.778434
## as.factor(subject)110  0.378739   1.050335   0.361 0.718407
## as.factor(subject)111  1.108717   0.904964   1.225 0.220519
## as.factor(subject)112  0.841080   1.034575   0.813 0.416235
## as.factor(subject)113 -0.051394   0.891972  -0.058 0.954053
## as.factor(subject)114 -0.801659   0.893341  -0.897 0.369521
## as.factor(subject)115 -0.958019   1.022296  -0.937 0.348695
## as.factor(subject)116  0.459671   0.889007   0.517 0.605113
## as.factor(subject)117 -2.005863   1.068678  -1.877 0.060524 .
## as.factor(subject)118 -0.466165   0.889082  -0.524 0.600055
## as.factor(subject)119 -1.241852   0.891519  -1.393 0.163632
## as.factor(subject)120  0.450204   0.891612   0.505 0.613606
## as.factor(subject)121  0.967939   0.884096   1.095 0.273589
## as.factor(subject)122 -0.269770   0.873118  -0.309 0.757342
## as.factor(subject)123 -0.328577   0.882372  -0.372 0.709611
## as.factor(subject)124 -0.682328   1.035626  -0.659 0.509989
## as.factor(subject)125  1.642336   0.875139   1.877 0.060565 .
## as.factor(subject)126  1.684468   0.898566   1.875 0.060845 .
## as.factor(subject)127 -1.115485   0.884679  -1.261 0.207348
## as.factor(subject)128  0.498451   0.881507   0.565 0.571766
## as.factor(subject)129  1.406865   0.877501   1.603 0.108876
## as.factor(subject)130 -1.367938   0.911002  -1.502 0.133207
## as.factor(subject)131  0.380880   1.041584   0.366 0.714608
## as.factor(subject)132 -0.116071   1.064200  -0.109 0.913148
## as.factor(subject)133  0.335871   1.035951   0.324 0.745775
```

```
## as.factor(subject)134  1.117168   0.903281   1.237 0.216166
## as.factor(subject)135  0.572089   0.891292   0.642 0.520961
## as.factor(subject)136 -0.713251   0.894202  -0.798 0.425080
## as.factor(subject)137 -0.173812   0.884212  -0.197 0.844162
## as.factor(subject)138  1.430391   0.897747   1.593 0.111090
## as.factor(subject)139  0.703406   0.870501   0.808 0.419063
## as.factor(subject)140 -1.617788   0.880199  -1.838 0.066065 .
## as.factor(subject)141  0.551034   0.885695   0.622 0.533844
## as.factor(subject)142 -1.038476   0.892688  -1.163 0.244702
## as.factor(subject)143 -0.126637   0.895204  -0.141 0.887505
## as.factor(subject)144 -1.041064   0.889480  -1.170 0.241832
## as.factor(subject)145 -0.587815   1.085870  -0.541 0.588279
## as.factor(subject)146 -1.801616   0.956436  -1.884 0.059609 .
## as.factor(subject)147 -0.420468   0.889034  -0.473 0.636249
## as.factor(subject)148 -1.188151   0.894900  -1.328 0.184280
## as.factor(subject)149 -0.891127   1.030045  -0.865 0.386966
## as.factor(subject)150 -0.586902   0.909398  -0.645 0.518685
## as.factor(subject)151  0.115753   0.885217   0.131 0.895964
## as.factor(subject)152 -0.821901   0.890433  -0.923 0.355989
## as.factor(subject)153 -0.081580   0.891222  -0.092 0.927065
## as.factor(subject)154  1.162533   0.885769   1.312 0.189366
## as.factor(subject)155  0.070333   0.893844   0.079 0.937282
## as.factor(subject)156 -1.104381   0.826999  -1.335 0.181743
## as.factor(subject)157 -1.002713   0.914555  -1.096 0.272906
## as.factor(subject)158 -0.163741   0.884405  -0.185 0.853117
## as.factor(subject)159 -0.903675   0.886966  -1.019 0.308280
## as.factor(subject)160  0.010880   0.870501   0.012 0.990028
## as.factor(subject)161 -0.894358   0.881271  -1.015 0.310177
## as.factor(subject)162 -3.004949   0.898416  -3.345 0.000824 ***
## as.factor(subject)163  0.450204   0.891612   0.505 0.613606
## as.factor(subject)164 -1.341272   1.191210  -1.126 0.260176
## as.factor(subject)165 -0.762568   0.888574  -0.858 0.390786
## as.factor(subject)166  0.454740   0.894848   0.508 0.611330
## as.factor(subject)167  2.715390   0.871015   3.117 0.001824 **
## as.factor(subject)168 -1.485243   0.892588  -1.664 0.096118 .
## as.factor(subject)169 -3.174629   0.891026  -3.563 0.000367 ***
## as.factor(subject)170 -0.721541   0.878743  -0.821 0.411586
## as.factor(subject)171 -0.385161   0.888210  -0.434 0.664552
## as.factor(subject)172  0.346546   0.899306   0.385 0.699979
## as.factor(subject)173 -0.312517   0.892958  -0.350 0.726354
## as.factor(subject)174 -0.392305   1.005268  -0.390 0.696352
## as.factor(subject)175 -0.026854   0.903115  -0.030 0.976279
## as.factor(subject)176  0.102728   0.889457   0.115 0.908052
## as.factor(subject)177  2.331835   1.062894   2.194 0.028246 *
## as.factor(subject)178 -0.575042   0.887839  -0.648 0.517187
## as.factor(subject)179 -0.056713   0.891323  -0.064 0.949267
## as.factor(subject)180  0.277409   0.894780   0.310 0.756538
## as.factor(subject)181  0.474275   0.909668   0.521 0.602108
## as.factor(subject)182 -0.201172   0.886346  -0.227 0.820449
## as.factor(subject)183  0.708402   0.892119   0.794 0.427157
## as.factor(subject)184 -0.210320   1.056311  -0.199 0.842178
## as.factor(subject)185 -0.320373   0.898636  -0.357 0.721458
## as.factor(subject)186 -0.366899   0.887057  -0.414 0.679157
## as.factor(subject)187 -0.020375   1.022412  -0.020 0.984101
```

```
## as.factor(subject)188  1.440110   0.900679   1.599 0.109839
## as.factor(subject)189  0.754949   0.887124   0.851 0.394765
## as.factor(subject)190 -2.320100   1.055464  -2.198 0.027936 *
## as.factor(subject)191 -1.206415   0.898526  -1.343 0.179382
## as.factor(subject)192  0.525387   0.870553   0.604 0.546170
## as.factor(subject)193 -1.834563   0.891836  -2.057 0.039680 *
## as.factor(subject)194  0.287312   0.896812   0.320 0.748688
## as.factor(subject)195  1.484118   0.876896   1.692 0.090557 .
## as.factor(subject)196  0.483760   0.898707   0.538 0.590381
## as.factor(subject)197  1.550689   1.035453   1.498 0.134239
## as.factor(subject)198 -0.121097   1.005195  -0.120 0.904110
## as.factor(subject)199 -0.002103   1.059742  -0.002 0.998416
## as.factor(subject)200  0.508533   0.892682   0.570 0.568903
## as.factor(subject)201  1.462604   0.887730   1.648 0.099439 .
## as.factor(subject)202 -1.540637   1.034324  -1.490 0.136353
## as.factor(subject)203 -0.443905   0.884248  -0.502 0.615657
## as.factor(subject)205  1.744087   0.906363   1.924 0.054321 .
## as.factor(subject)206  0.036266   0.870620   0.042 0.966773
## as.factor(subject)207 -0.194193   0.878370  -0.221 0.825027
## as.factor(subject)208 -0.428571   1.005154  -0.426 0.669835
## as.factor(subject)209  0.077820   1.043128   0.075 0.940531
## as.factor(subject)210  0.840956   0.882679   0.953 0.340726
## as.factor(subject)211 -0.432308   0.892942  -0.484 0.628288
## as.factor(subject)212  1.019435   0.901695   1.131 0.258234
## as.factor(subject)213  0.938500   0.881010   1.065 0.286761
## as.factor(subject)214 -0.036829   0.888461  -0.041 0.966935
## as.factor(subject)215  0.071015   0.902660   0.079 0.937292
## as.factor(subject)216 -0.905883   0.881088  -1.028 0.303883
## as.factor(subject)217 -0.712617   0.890002  -0.801 0.423311
## as.factor(subject)218  0.003488   0.907132   0.004 0.996932
## as.factor(subject)219 -1.067722   0.887745  -1.203 0.229079
## as.factor(subject)220 -0.540658   1.079854  -0.501 0.616599
## as.factor(subject)221 -0.007253   1.005158  -0.007 0.994242
## as.factor(subject)222  0.139018   0.887112   0.157 0.875474
## as.factor(subject)223 -0.317449   1.048510  -0.303 0.762071
## as.factor(subject)224  1.337736   0.888734   1.505 0.132269
## as.factor(subject)225  0.913488   0.894615   1.021 0.307209
## as.factor(subject)226 -0.283224   1.026018  -0.276 0.782516
## as.factor(subject)227  1.497109   0.901536   1.661 0.096790 .
## as.factor(subject)228  0.225352   1.042334   0.216 0.828832
## as.factor(subject)229  0.979518   1.023949   0.957 0.338765
## as.factor(subject)231  1.915373   0.989837   1.935 0.052985 .
## as.factor(subject)232  0.039278   0.917692   0.043 0.965861
## as.factor(subject)233 -0.302182   1.039720  -0.291 0.771328
## as.factor(subject)234  1.061125   0.899647   1.179 0.238203
## as.factor(subject)235  0.972861   1.031780   0.943 0.345734
## as.factor(subject)236  1.021973   1.046422   0.977 0.328750
## as.factor(subject)237  0.967843   0.881978   1.097 0.272486
## as.factor(subject)238 -0.859298   1.036668  -0.829 0.407159
## as.factor(subject)239  1.593465   1.104056   1.443 0.148941
## as.factor(subject)240  2.340600   1.055633   2.217 0.026606 *
## as.factor(subject)241 -0.379022   1.033601  -0.367 0.713843
## as.factor(subject)242  1.167426   0.877738   1.330 0.183505
## as.factor(subject)243 -0.440767   0.881611  -0.500 0.617106
```

```
## as.factor(subject)244  1.812384   1.025973   1.767 0.077311 .
## as.factor(subject)245  0.320553   1.084107   0.296 0.767471
## as.factor(subject)246 -1.530271   0.888223  -1.723 0.084916 .
## as.factor(subject)247 -0.941046   0.884521  -1.064 0.287372
## as.factor(subject)248  0.483394   0.891965   0.542 0.587858
## as.factor(subject)249  1.461343   0.889112   1.644 0.100259
## as.factor(subject)250  0.222814   0.875484   0.255 0.799107
## as.factor(subject)251  0.277677   0.914559   0.304 0.761419
## as.factor(subject)252  1.640106   0.883949   1.855 0.063535 .
## as.factor(subject)253  0.069088   0.891060   0.078 0.938199
## as.factor(subject)254  2.475473   0.880214   2.812 0.004918 **
## as.factor(subject)255  0.219164   0.878353   0.250 0.802961
## as.factor(subject)256 -0.319823   1.046400  -0.306 0.759878
## as.factor(subject)257  0.444730   0.895901   0.496 0.619608
## as.factor(subject)258  0.775071   0.912162   0.850 0.395488
## as.factor(subject)259 -0.767141   0.879027  -0.873 0.382818
## as.factor(subject)260  1.903732   0.874523   2.177 0.029489 *
## as.factor(subject)261 -1.760959   0.891640  -1.975 0.048272 *
## as.factor(subject)262  0.878019   0.890606   0.986 0.324198
## as.factor(subject)263  1.657707   1.023130   1.620 0.105183
## as.factor(subject)265  0.738134   0.935710   0.789 0.430200
## as.factor(subject)266 -0.339192   0.908640  -0.373 0.708928
## as.factor(subject)267  1.126691   1.051373   1.072 0.283883
## as.factor(subject)268  0.685949   0.906125   0.757 0.449042
## as.factor(subject)269  0.953987   0.878251   1.086 0.277375
## as.factor(subject)270  0.884584   0.964449   0.917 0.359042
## as.factor(subject)271  0.482418   0.878598   0.549 0.582952
## as.factor(subject)272 -1.452641   1.046558  -1.388 0.165132
## as.factor(subject)273  0.600268   0.899607   0.667 0.504608
## as.factor(subject)274  2.237150   1.005924   2.224 0.026150 *
## as.factor(subject)275  0.030218   1.037961   0.029 0.976775
## as.factor(subject)276 -1.574100   1.045058  -1.506 0.132008
## as.factor(subject)277 -7.598105  85.098478  -0.089 0.928855
## as.factor(subject)278  0.193630   1.005377   0.193 0.847276
## as.factor(subject)279  1.686581   1.042442   1.618 0.105681
## as.factor(subject)280 -0.224757   1.019767  -0.220 0.825560
## as.factor(subject)281  0.360801   1.012581   0.356 0.721602
## as.factor(subject)282  2.314727   1.005227   2.303 0.021296 *
## as.factor(subject)283  0.937730   1.025475   0.914 0.360488
## as.factor(subject)284  0.145731   1.026509   0.142 0.887105
## as.factor(subject)285  0.721069   1.012689   0.712 0.476444
## as.factor(subject)286 -0.099337   1.005318  -0.099 0.921288
## as.factor(subject)287 -0.882622   1.102577  -0.801 0.423417
## as.factor(subject)288  1.808946   1.056570   1.712 0.086879 .
## as.factor(subject)289  0.890733   1.025322   0.869 0.384992
## as.factor(subject)290 -1.700678   1.101595  -1.544 0.122629
## as.factor(subject)291  1.404874   1.017259   1.381 0.167267
## as.factor(subject)292 -0.096663   1.047105  -0.092 0.926449
## as.factor(subject)293  0.715555   1.040081   0.688 0.491465
## as.factor(subject)294 -0.355572   1.066403  -0.333 0.738809
## as.factor(subject)295  1.946768   1.037328   1.877 0.060557 .
## as.factor(subject)296  0.588781   1.013060   0.581 0.561112
## as.factor(subject)297 -0.542415   1.005227  -0.540 0.589476
## as.factor(subject)298  0.086735   1.030533   0.084 0.932925
```

```
## as.factor(subject)299 -1.040525   1.031092  -1.009 0.312903
## as.factor(subject)300  1.893409   1.005268   1.883 0.059634 .
## as.factor(subject)301 -1.381328   1.033275  -1.337 0.181274
## as.factor(subject)302  1.160443   1.037238   1.119 0.263233
## as.factor(subject)303 -1.024469   1.115206  -0.919 0.358286
## as.factor(subject)304  1.111900   1.043789   1.065 0.286762
## as.factor(subject)305 -0.582948   1.048256  -0.556 0.578134
## as.factor(subject)307  0.514142   1.049059   0.490 0.624065
## as.factor(subject)308  0.256801   1.060026   0.242 0.808579
## as.factor(subject)309 -0.066102   1.019020  -0.065 0.948279
## as.factor(subject)310 -0.346704   1.033552  -0.335 0.737287
## as.factor(subject)311 -1.532533   1.034292  -1.482 0.138414
## as.factor(subject)312 -2.138108   1.042996  -2.050 0.040368 *
## as.factor(subject)313 -1.645929   1.041021  -1.581 0.113862
## as.factor(subject)314 -1.004357   1.016860  -0.988 0.323298
## as.factor(subject)315 -0.792849   1.055896  -0.751 0.452726
## as.factor(subject)316 -0.357523   1.030958  -0.347 0.728752
## as.factor(subject)317  0.699030   1.039987   0.672 0.501487
## as.factor(subject)318 -0.053775   1.035921  -0.052 0.958600
## as.factor(subject)319 -0.073758   1.019050  -0.072 0.942300
## as.factor(subject)320 -0.247057   1.052007  -0.235 0.814330
## as.factor(subject)321 -1.064549   1.031156  -1.032 0.301892
## as.factor(subject)322 -0.856502   1.023166  -0.837 0.402531
## as.factor(subject)323 -0.086724   1.043703  -0.083 0.933778
## as.factor(subject)324  0.775894   1.060631   0.732 0.464449
## as.factor(subject)325  0.307130   1.057392   0.290 0.771464
## as.factor(subject)326 -0.475496   1.044245  -0.455 0.648858
## as.factor(subject)327 -1.366823   1.044419  -1.309 0.190639
## as.factor(subject)328 -0.468906   1.085116  -0.432 0.665650
## as.factor(subject)329 -2.920949   1.029500  -2.837 0.004550 **
## as.factor(subject)330 -1.603886   1.043706  -1.537 0.124361
## as.factor(subject)331  0.207819   1.037097   0.200 0.841179
## as.factor(subject)332 -0.987760   1.037602  -0.952 0.341115
## as.factor(subject)333 -0.947920   1.042709  -0.909 0.363301
## as.factor(subject)334  3.231984   1.045202   3.092 0.001987 **
## as.factor(subject)335  0.327723   1.035942   0.316 0.751735
## as.factor(subject)336 -0.585459   1.046907  -0.559 0.576007
## as.factor(subject)337  0.871649   1.005172   0.867 0.385852
## as.factor(subject)338 -0.360255   1.032831  -0.349 0.727237
## as.factor(subject)339  0.695992   1.034093   0.673 0.500918
## as.factor(subject)340  0.846735   1.032977   0.820 0.412385
## as.factor(subject)341  0.732076   1.040221   0.704 0.481576
## as.factor(subject)342 -0.546630   1.034684  -0.528 0.597287
## as.factor(subject)344  1.824820   1.045900   1.745 0.081031 .
## as.factor(subject)345  1.292847   1.028620   1.257 0.208799
## as.factor(subject)346 -2.486490   1.046400  -2.376 0.017490 *
## as.factor(subject)347 -2.344757   1.043546  -2.247 0.024646 *
## as.factor(subject)348 -1.615201   1.052364  -1.535 0.124825
## as.factor(subject)349 -2.518284   1.044041  -2.412 0.015863 *
## as.factor(subject)350 -1.207829   1.062741  -1.137 0.255738
## as.factor(subject)351 -1.739519   1.081759  -1.608 0.107825
## as.factor(subject)352 -0.621242   1.052159  -0.590 0.554892
## as.factor(subject)353 -0.666058   1.044595  -0.638 0.523719
## as.factor(subject)354  1.366922   1.022627   1.337 0.181328
```

```
## as.factor(subject)355 -0.644741   1.099824  -0.586 0.557726
## as.factor(subject)356 -0.435825   1.005158  -0.434 0.664588
## as.factor(subject)357 -0.506458   1.040504  -0.487 0.626440
## as.factor(subject)358 -0.370940   1.033572  -0.359 0.719677
## as.factor(subject)359 -0.995957   1.037593  -0.960 0.337119
## as.factor(subject)360 -0.632892   1.084067  -0.584 0.559346
## as.factor(subject)361 -0.950212   1.063556  -0.893 0.371627
## as.factor(subject)362 -1.028758   1.037668  -0.991 0.321484
## as.factor(subject)363 -0.023723   1.028294  -0.023 0.981594
## as.factor(subject)364 -0.362860   1.033554  -0.351 0.725529
## as.factor(subject)365  0.751269   1.036043   0.725 0.468370
## as.factor(subject)366  0.364260   1.041502   0.350 0.726530
## as.factor(subject)367  1.560708   1.094978   1.425 0.154061
## as.factor(subject)368 -1.174215   0.895958  -1.311 0.190003
## as.factor(subject)369 -0.564494   1.040930  -0.542 0.587613
## as.factor(subject)370  2.348443   1.010671   2.324 0.020144 *
## as.factor(subject)371 -1.165909   1.038523  -1.123 0.261582
## as.factor(subject)372  0.910952   1.039535   0.876 0.380863
## as.factor(subject)373  0.544315   1.040835   0.523 0.601002
## as.factor(subject)374 -0.529032   1.088433  -0.486 0.626932
## as.factor(subject)375  0.054816   1.038131   0.053 0.957889
## as.factor(subject)376 -0.267507   1.025973  -0.261 0.794297
## as.factor(subject)377  0.639409   1.033886   0.618 0.536278
## as.factor(subject)378 -0.150110   1.005158  -0.149 0.881285
## as.factor(subject)379 -1.661618   1.046391  -1.588 0.112297
## as.factor(subject)380  1.213933   1.023496   1.186 0.235596
## as.factor(subject)381  0.570184   1.010833   0.564 0.572704
## as.factor(subject)382  1.046346   1.065672   0.982 0.326166
## as.factor(subject)383 -0.473321   1.040511  -0.455 0.649187
## as.factor(subject)384 -1.191027   1.093662  -1.089 0.276142
## as.factor(subject)385  0.566880   1.044622   0.543 0.587360
## as.factor(subject)386 -1.004155   1.037595  -0.968 0.333158
## as.factor(subject)387  1.397263   1.051413   1.329 0.183868
## as.factor(subject)388  0.433040   1.028292   0.421 0.673664
## as.factor(subject)389  0.265672   1.060121   0.251 0.802119
## as.factor(subject)390  0.022015   1.037926   0.021 0.983078
## as.factor(subject)391 -0.330554   1.033594  -0.320 0.749112
## as.factor(subject)392 -1.048836   1.084279  -0.967 0.333388
## as.factor(subject)393 -0.212087   1.032516  -0.205 0.837253
## as.factor(subject)394 -0.502741   1.045380  -0.481 0.630575
## as.factor(subject)395  0.054816   1.038131   0.053 0.957889
## as.factor(subject)396 -1.030149   1.042887  -0.988 0.323258
## as.factor(subject)397 -0.595350   1.052397  -0.566 0.571592
## as.factor(subject)398  0.840377   1.030711   0.815 0.414879
## as.factor(subject)399 -0.462398   1.045912  -0.442 0.658417
## as.factor(subject)400  2.142693   1.057068   2.027 0.042661 *
## as.factor(subject)401  0.807574   1.057490   0.764 0.445063
## as.factor(subject)402  1.373407   1.046514   1.312 0.189397
## as.factor(subject)403 -1.381328   1.033275  -1.337 0.181274
## as.factor(subject)404  0.380880   1.041584   0.366 0.714608
## as.factor(subject)405 -1.973669   1.042399  -1.893 0.058306 .
## as.factor(subject)406 -2.860370   1.041641  -2.746 0.006032 **
## as.factor(subject)407  0.699779   1.005172   0.696 0.486317
## as.factor(subject)408 -1.203929   1.046521  -1.150 0.249975
```

```
## as.factor(subject)409  0.061564   1.018453   0.060 0.951798
## as.factor(subject)410  2.329234   1.005318   2.317 0.020508 *
## as.factor(subject)411  0.641867   1.025798   0.626 0.531496
## as.factor(subject)412 -0.061869   1.031485  -0.060 0.952171
## as.factor(subject)413 -1.793434   1.051364  -1.706 0.088042 .
## as.factor(subject)414 -0.840110   1.044100  -0.805 0.421035
## as.factor(subject)415  0.535408   1.106906   0.484 0.628600
## as.factor(subject)417  0.214026   1.046558   0.205 0.837959
## as.factor(subject)418  2.225261   1.015039   2.192 0.028358 *
## as.factor(subject)419  0.372092   1.077272   0.345 0.729792
## as.factor(subject)420  0.181149   1.095829   0.165 0.868702
## as.factor(subject)422  1.064584   1.038524   1.025 0.305319
## as.factor(subject)423  0.069474   1.043041   0.067 0.946895
## as.factor(subject)424  1.076585   1.075190   1.001 0.316683
## as.factor(subject)425  2.167276   1.046485   2.071 0.038358 *
## as.factor(subject)426  0.900507   1.062528   0.848 0.396709
## as.factor(subject)427 -0.867885   1.099467  -0.789 0.429897
## as.factor(subject)428 -0.536494   1.101401  -0.487 0.626186
## as.factor(subject)429 -0.435825   1.005158  -0.434 0.664588
## as.factor(subject)430  0.040361   1.050988   0.038 0.969366
## as.factor(subject)431 -0.674188   1.035621  -0.651 0.515047
## as.factor(subject)432  3.272816   1.059012   3.090 0.001999 **
## as.factor(subject)433 -0.652848   1.100762  -0.593 0.553123
## as.factor(subject)434  1.015650   1.057851   0.960 0.337001
## as.factor(subject)436  0.390265   1.061911   0.368 0.713237
## as.factor(subject)437 -0.343270   1.074722  -0.319 0.749421
## as.factor(subject)438  0.935660   1.119210   0.836 0.403154
## as.factor(subject)439  0.121350   1.099613   0.110 0.912126
## as.factor(subject)440  1.642841   1.045948   1.571 0.116259
## as.factor(subject)441 -2.624873   1.044460  -2.513 0.011966 *
## as.factor(subject)442 -0.451351   0.887099  -0.509 0.610896
## as.factor(subject)443  0.444799   1.022581   0.435 0.663579
## as.factor(subject)444  0.150110   0.870494   0.172 0.863089
## as.factor(subject)445 -0.714576   0.878894  -0.813 0.416195
## as.factor(subject)446  1.619943   1.042008   1.555 0.120033
## as.factor(subject)447 -0.232270   0.917658  -0.253 0.800182
## as.factor(subject)448  1.883379   0.908664   2.073 0.038201 *
## as.factor(subject)449 -0.064965   0.886443  -0.073 0.941577
## as.factor(subject)450  2.133369   1.031068   2.069 0.038538 *
## as.factor(subject)451 -0.549669   1.005195  -0.547 0.584497
## as.factor(subject)452 -0.997994   0.892329  -1.118 0.263390
## as.factor(subject)453  0.422412   1.041988   0.405 0.685191
## as.factor(subject)454 -0.057859   0.885814  -0.065 0.947922
## as.factor(subject)455 -0.845746   1.025695  -0.825 0.409622
## as.factor(subject)456 -2.593003   1.058050  -2.451 0.014256 *
## as.factor(subject)457 -0.205708   1.038960  -0.198 0.843050
## as.factor(subject)458  0.820336   1.039104   0.789 0.429840
## as.factor(subject)459  2.675361   0.886278   3.019 0.002539 **
## as.factor(subject)460  1.963414   0.886251   2.215 0.026732 *
## as.factor(subject)461  0.621062   0.889611   0.698 0.485097
## as.factor(subject)462  0.271208   1.005172   0.270 0.787305
## as.factor(subject)463  0.120980   0.902698   0.134 0.893386
## as.factor(subject)464 -0.304552   0.886540  -0.344 0.731201
## as.factor(subject)465 -1.011537   0.888903  -1.138 0.255137
```

```
## as.factor(subject)466  1.704109   0.882662   1.931 0.053527 .
## as.factor(subject)467 -0.502277   0.882854  -0.569 0.569407
## as.factor(subject)468  0.239412   1.046802   0.229 0.819096
## as.factor(subject)469 -0.260916   1.040028  -0.251 0.801911
## as.factor(subject)470  2.245496   0.889799   2.524 0.011616 *
## as.factor(subject)471 -0.275365   1.025990  -0.268 0.788400
## as.factor(subject)472 -1.298653   1.058436  -1.227 0.219840
## as.factor(subject)474 -0.227688   1.005446  -0.226 0.820848
## as.factor(subject)475  1.878903   1.005195   1.869 0.061596 .
## as.factor(subject)476 -0.320612   0.888952  -0.361 0.718352
## as.factor(subject)477 -3.461376   1.046357  -3.308 0.000940 ***
## as.factor(subject)478 -0.939578   1.042771  -0.901 0.367567
## as.factor(subject)479 -0.826617   1.036631  -0.797 0.425214
## as.factor(subject)480  0.557124   1.065929   0.523 0.601207
## as.factor(subject)481  0.095785   1.030426   0.093 0.925938
## as.factor(subject)482 -0.061135   1.070090  -0.057 0.954441
## as.factor(subject)483 -0.165932   1.015237  -0.163 0.870171
## as.factor(subject)484  0.220402   0.900511   0.245 0.806648
## as.factor(subject)485  1.284453   1.043304   1.231 0.218271
## as.factor(subject)486  0.035022   1.048435   0.033 0.973353
## as.factor(subject)487  0.934041   1.059347   0.882 0.377931
## as.factor(subject)488 -0.099364   0.881969  -0.113 0.910298
## as.factor(subject)489 -0.644971   1.041252  -0.619 0.535641
## as.factor(subject)490 -0.167273   0.904647  -0.185 0.853304
## as.factor(subject)491 -3.240378   1.060138  -3.057 0.002239 **
## as.factor(subject)492  0.056478   1.024088   0.055 0.956019
## as.factor(subject)493 -1.365184   1.033222  -1.321 0.186405
## as.factor(subject)494  0.166949   1.036933   0.161 0.872091
## as.factor(subject)495 -0.577729   1.009919  -0.572 0.567285
## as.factor(subject)496 -4.991540   1.080447  -4.620 3.84e-06 ***
## as.factor(subject)498 -2.362562   1.025112  -2.305 0.021184 *
## as.factor(subject)499 -1.032451   1.022068  -1.010 0.312419
## as.factor(subject)500  0.756995   0.905395   0.836 0.403102
## as.factor(subject)501 -0.859654   1.095487  -0.785 0.432616
## as.factor(subject)502 -2.300221   1.005172  -2.288 0.022115 *
## as.factor(subject)503 -2.093921   1.046802  -2.000 0.045468 *
## as.factor(subject)504  2.850994   0.870915   3.274 0.001062 **
## as.factor(subject)505  0.583258   0.887496   0.657 0.511055
## as.factor(subject)506  0.311424   1.035957   0.301 0.763708
## as.factor(subject)507 -0.982648   0.894404  -1.099 0.271916
## as.factor(subject)508 -0.997994   1.042578  -0.957 0.338448
## as.factor(subject)509  0.842636   1.005172   0.838 0.401862
## as.factor(subject)510  1.373393   0.881621   1.558 0.119280
## as.factor(subject)511 -0.983984   1.096523  -0.897 0.369523
## as.factor(subject)512  0.271208   1.005172   0.270 0.787305
## as.factor(subject)513 -0.883151   1.030035  -0.857 0.391225
## as.factor(subject)514  2.072439   0.888288   2.333 0.019644 *
## as.factor(subject)515 -0.546488   1.027997  -0.532 0.595000
## as.factor(subject)516 -1.141236   1.038557  -1.099 0.271826
## as.factor(subject)517 -0.798422   0.890260  -0.897 0.369803
## as.factor(subject)518 -1.588928   1.046758  -1.518 0.129027
## as.factor(subject)527 -0.262240   1.055886  -0.248 0.803855
## as.factor(subject)529 -0.714286   1.005154  -0.711 0.477318
## logSigma               0.631522   0.012975  48.672  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Newton-Raphson maximisation, 7 iterations
## Return code 8: successive function values within relative tolerance limit (reltol)
## Log-likelihood: -6750.522 on 416 Df
```

We can check whether we should include the fixed effect by using the likelihood ratio test. First, we check the log-likelihood values of both models:

```
print('Model without FE')
```

```
## [1] "Model without FE"
```

```
logLik(cens.out.dummy)
```

```
## 'log Lik.' -7285.874 (df=9)
```

```
print('Model with FE')
```

```
## [1] "Model with FE"
```

```
logLik(cens.out.dummy.fe)
```

```
## 'log Lik.' -6750.522 (df=416)
```

We calculate the log-likelihood ratio and compare it with the chi-square distribution

```
print('log likelihood ratio')
```

```
## [1] "log likelihood ratio"
```

```
-2*(as.numeric(logLik(cens.out.dummy)) -
as.numeric(logLik(cens.out.dummy.fe)) )
```

```
## [1] 1070.705
```

```
print('critical value for p=0.95')
```

```
## [1] "critical value for p=0.95"
```

```
qchisq(0.95,df=
(attributes(logLik(cens.out.dummy.fe))$df-
                                          attributes(logLik(cens.out.dummy))$df)
)
```

```
## [1] 455.038
```

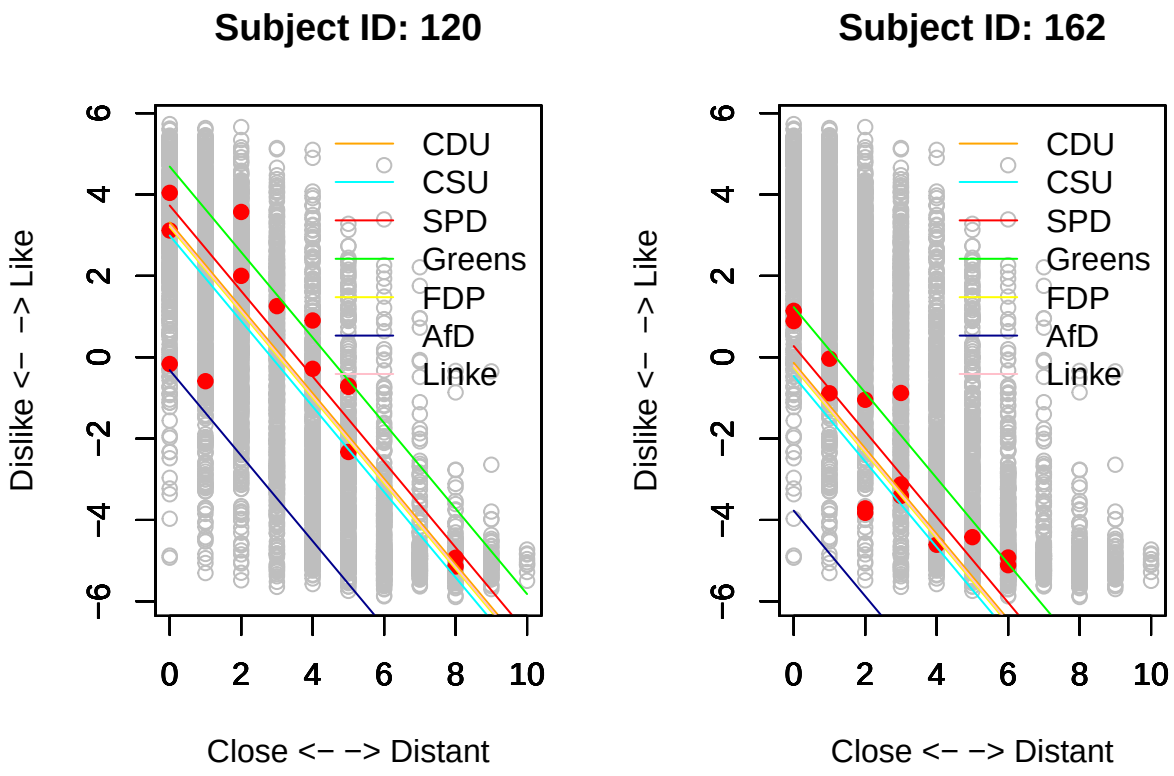
Since the log-likelihood ratio exceeds the critical value, we can reject the null-hypothesis that both models' are at the same level in their performance. In other words, the model with fixed effects has significantly better performance than the model without fixed effects.

Now we can check again the two respondents whose predicted values have already demonstrated in the non-censored linear model (see the section for the panel data analysis):

```
par(mfrow=c(1,2))
for (i.subj in c(120,162)){

  this.fe <-coef(cens.out.dummy.fe)[grep(as.character(i.subj),
                                         names(coef(cens.out.dummy.fe)))]

  plot(long.data$skalo.true ~ long.data$lr.dist.abs,
```

```
     xlab="Close <- -> Distant",ylab="Dislike <-  -> Like",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T),col="grey",
     main=paste("Subject ID:",i.subj))
  par(new=T)
  plot(long.data$skalo.true[long.data$subject==i.subj] ~
         long.data$lr.dist.abs[long.data$subject==i.subj],
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T),
     col="red",pch=19)

predicted.cdu <- coef(cens.out.dummy.fe)[1] + coef(cens.out.dummy.fe)[2]*c(0:10)+  this.fe
predicted.csu <- coef(cens.out.dummy.fe)[1] + coef(cens.out.dummy.fe)[2]*c(0:10) +coef(cens.out.dummy.fe
predicted.spd <- coef(cens.out.dummy.fe)[1] + coef(cens.out.dummy.fe)[2]*c(0:10) +coef(cens.out.dummy.fe
predicted.gru <- coef(cens.out.dummy.fe)[1] + coef(cens.out.dummy.fe)[2]*c(0:10) +coef(cens.out.dummy.fe
predicted.fdp <- coef(cens.out.dummy.fe)[1] + coef(cens.out.dummy.fe)[2]*c(0:10) +coef(cens.out.dummy.fe
predicted.afd <- coef(cens.out.dummy.fe)[1] + coef(cens.out.dummy.fe)[2]*c(0:10) +coef(cens.out.dummy.fe
predicted.lin <- coef(cens.out.dummy.fe)[1] + coef(cens.out.dummy.fe)[2]*c(0:10) +coef(cens.out.dummy.fe

par(new=T)
plot(predicted.cdu ~ c(0:10),col="orange",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.csu ~ c(0:10),col="cyan",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.spd ~ c(0:10),col="red",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.gru ~ c(0:10),col="green",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.fdp ~ c(0:10),col="yellow",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.afd ~ c(0:10),col="blue4",type="l",
     ann=F,axes=F,xlab="",ylab="",
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))
par(new=T)
plot(predicted.lin ~ c(0:10),col="pink",type="l",
     ann=F,axes=F,xlab="",ylab="",
```

```
     xlim=range(long.data$lr.dist.abs,na.rm=T),
     ylim=range(long.data$skalo.true,na.rm=T))


legend("topright",col=c("orange","cyan","red","green","yellow","blue4","pink"),
       lty=c(rep(1,8)),c("CDU","CSU","SPD","Greens","FDP","AfD",
                         "Linke"),
       lwd=c(rep(1,8)),
       bty="n")
}
```




```
knitr::knit_exit()
```