Preamble: Frequentis Foundations

Shravan Vasishth (vasishth.github.io)

October 2025

Contents

Textbook

Introduction to Bayesian Data Analysis for Cognitive Science

Nicenboim, Schad, Vasishth

- Online version:

https://bruno.nicenboim.me/bayescogsci/

- Source code:

https://github.com/bnicenboim/bayescogsci

- Physical book:

here

Be sure to read the textbook's chapter 1 before watching this lecture.

Introduction

This lecture covers some basic ideas in frequentist statistics that everyone should know. These ideas are very useful as background knowledge when studying Bayesian methods.

Reminder from chapter 1: Maximum likelihood estimates (MLEs)

For the normal distribution, where $X \sim N(\mu, \sigma)$, and given $i = 1, \dots, n$ independent data points, we can get MLEs of μ and σ by computing:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i = \bar{x} \tag{1}$$

and

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2 = s^2 \tag{2}$$

you will sometimes see the "unbiased" estimate (and this is what `R` computes) but for large sample sizes the difference is not important:

$$\hat{\sigma}^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 = s^2 \tag{3}$$

I use $\bar{x}$ and s to represent the **estimates** of the mean and standard deviation from a particular data-set. $\hat{\mu}$ and $\hat{\sigma}$ are the formulas (analytically derived) for estimating the mean and standard deviation, and are called the **estimators**.

The significance of these MLEs is that, having assumed a particular underlying pdf, we can estimate the (unknown) parameters (the mean and variance/standard deviation) of the distribution that generated our particular data.

This leads us to the distributional properties of the mean **under (hypothetical) repeated sampling**.

The central limit theorem

For large enough sample sizes, the sampling distribution of the means will be approximately normal, regardless of the underlying distribution (as long as this distribution has a mean and variance defined for it).

- So, from a sample of size n, and sd σ, we can compute **the standard deviation of the sampling distribution of the means**.
- We will call this standard deviation the **standard error**.

$SE = \frac{\sigma}{\sqrt{n}}$

When estimated from data, we will write

$SE = \frac{s}{\sqrt{n}}$

I say **estimated** because we are estimating SE using an estimate of σ.

The estimated standard error allows us to define a so-called **95% confidence interval**:

$$\bar{x} \pm 1.96 SE \tag{4}$$

So, for a given sample mean, we define a 95% confidence interval as follows:

$$\bar{x} \pm 1.96 \frac{s}{\sqrt{n}} \tag{5}$$

I usually just write:

$$\bar{x} \pm 2 \frac{s}{\sqrt{n}} \tag{6}$$

Example with simulated data:

```r
n<-100
x<-rnorm(n,mean=500,sd=100)
mu_hat<-mean(x)
hat_sigma<-sd(x)
```

```r
## lower bound:
mu_hat-(2*hat_sigma/sqrt(n))
```

```
## [1] 486.031
```

```r
## upper bound:
mu_hat+(2*hat_sigma/sqrt(n))
```

```
## [1] 524.4064
```

What does the 95% CI mean?

If you take repeated samples from a particular distribution, and compute the CI each time, 95% of those repeatedly computed CIs will contain the true population mean.

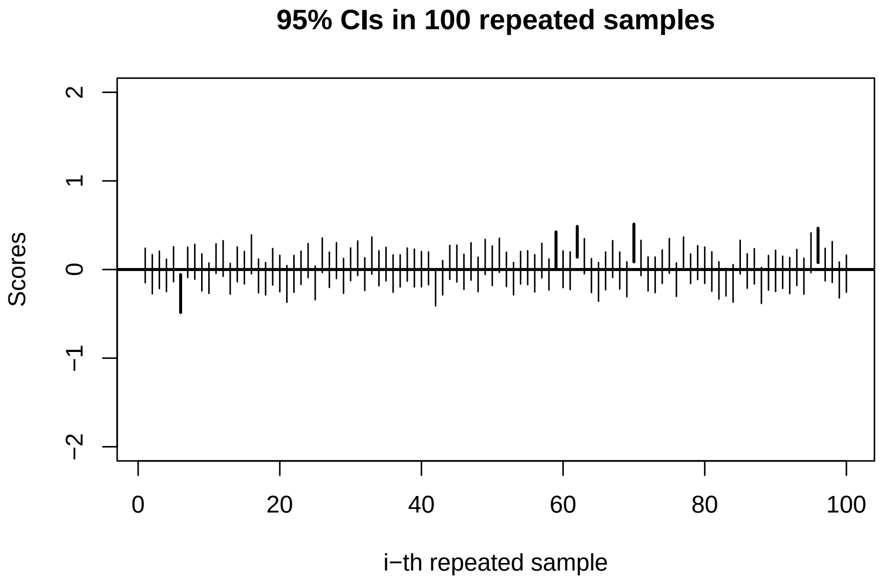
```r
nsim<-100
mu<-0
sigma<-1
lower<-rep(NA,nsim)
upper<-rep(NA,nsim)
for(i in 1:nsim){
  x<-rnorm(n,mean=mu,sd=sigma)
  lower[i]<-mean(x) - 2 * sd(x)/sqrt(n)
  upper[i]<-mean(x) + 2 * sd(x)/sqrt(n)
}
```

```r
## check how many CIs contain mu:
CIs<-ifelse(lower<mu & upper>mu,1,0)
table(CIs)
```

```
## CIs
##  0  1
##  5 95
## approx. 95% of the CIs contain true mean:
table(CIs)[2]/sum(table(CIs))
```

```
##    1
## 0.95
```

Graphical visualization:



There is a correspondence between the **correctly computed** CI and the hypothesis testing procedure (see below).

Example of incorrectly computed CIs

If you have repeated measures/dependent data, then the correct CI is computed after aggregation such that you have only one data point per subject per condition.

Consider these repeated measures data:

```
library(bcogsci)
data("df_gg05_rc")
head(df_gg05_rc)
```

```
##   subj item condition  RT residRT qcorrect experiment
## 1    1    1    objgap 320  -21.39        0     tedrg3
## 2    1    2   subjgap 424   74.66        1     tedrg2
## 3    1    3    objgap 309  -40.34        0     tedrg3
## 4    1    4   subjgap 274  -91.24        1     tedrg2
## 5    1    5    objgap 333   -8.39        1     tedrg3
## 6    1    6   subjgap 266  -87.32        1     tedrg2
```

```
## 8 data points per subject per condition:
t(xtabs(~subj+condition,df_gg05_rc))
```

```
##          subj
## condition 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
##   objgap  8 8 8 8 8 8 8 8 8  8  8  8  8  8  8  8  8  8  8  8
##   subjgap 8 8 8 8 8 8 8 8 8  8  8  8  8  8  8  8  8  8  8  8
##          subj
## condition 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42
##   objgap   8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8
##   subjgap  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8
```

Incorrectly computed CIs per condition:

```
(means<-with(df_gg05_rc,tapply(RT,condition,mean)))
```

```
##   objgap  subjgap
## 471.3601 369.0744
```

```
(sds<-with(df_gg05_rc,tapply(RT,condition,sd)))
```

```
##   objgap  subjgap
## 464.4060 177.2674
```

```
## what should n be?
(n<- length(unique(df_gg05_rc$subj)))
```

```
## [1] 42
```

```
## wrong lower bound for objgap condition:
means[1]-2*sds[1]/sqrt(n)
```

```
##   objgap
## 328.0413
```

```
## wrong upper bound:
means[1]+2*sds[1]/sqrt(n)
```

```
##   objgap
## 614.6789
```

Correctly computed CIs:

```
agg_gg05<-aggregate(RT~subj+condition,mean,
                           data=df_gg05_rc)
t(xtabs(~subj+condition,agg_gg05))
```

```
##          subj
## condition 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
##   objgap  1 1 1 1 1 1 1 1 1  1  1  1  1  1  1  1  1  1  1  1
##   subjgap 1 1 1 1 1 1 1 1 1  1  1  1  1  1  1  1  1  1  1  1
##          subj
## condition 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42
##   objgap   1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##   subjgap  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

```
## Correct CIs:
(means<-with(agg_gg05,tapply(RT,condition,mean)))
```

```
##   objgap  subjgap
## 471.3601 369.0744
```

```
(sds<-with(agg_gg05,tapply(RT,condition,sd)))
```

```
##   objgap  subjgap
## 259.8924 117.6722
```

```
## what should n be?
(n<- length(unique(agg_gg05$subj)))
```

```
## [1] 42
## correct lower bound for objgap condition:
means[1]-2*sds[1]/sqrt(n)
```

```
##   objgap
## 391.1556
```

```
## correct upper bound:
means[1]+2*sds[1]/sqrt(n)
```

```
##   objgap
## 551.5647
```

The t-test

The hypothesis test

Suppose we have a random sample of size n, and the data come from a $N(\mu,\sigma)$ distribution, and the data are independent and identically distributed (for now).

We can estimate sample mean $\bar{x}=\hat{\mu}$ and and sample standard deviation $s=\hat{\sigma}$, which in turn allows us to estimate **the sampling distribution of the mean under (hypothetical) repeated sampling** (thanks to the central limit theorem):

$$N(\bar{x},\frac{s}{\sqrt{n}}) \tag{7}$$

The NHST approach is to set up a null hypothesis that μ has some fixed value. For example:

$$H_0:\mu=\mu_0=0 \tag{8}$$

This amounts to assuming that the true distribution of sample means is (approximately) normally distributed and centered at 0, **with the standard error estimated from the data**.

The intuitive idea is that

- if the sample mean $\bar{x}$ is "near" the hypothesized μ (here, 0), the data are (possibly) "consistent with' ' the null hypothesis distribution.
- if the sample mean $\bar{x}$ is far from the hypothesized μ, the data are inconsistent with the null hypothesis distribution.

We formalize "near" and "far" by determining the value of the number t, which represents how many standard errors the sample mean is distant from the hypothesized mean:

$$t \times SE = \bar{x} - \mu \tag{9}$$

The above equation quantifies the distance of sample mean from μ in SE units.

So, given a sample and null hypothesis mean μ, we can compute the quantity:

$$t = \frac{\bar{x} - \mu}{SE} \tag{10}$$

We will call this the **observed t-value**.

The random variable T:

$$T = \frac{\bar{X} - \mu}{SE} \tag{11}$$

has a t-distribution, which is defined in terms of the sample size n. We will express this as: $T \sim t(n-1)$.

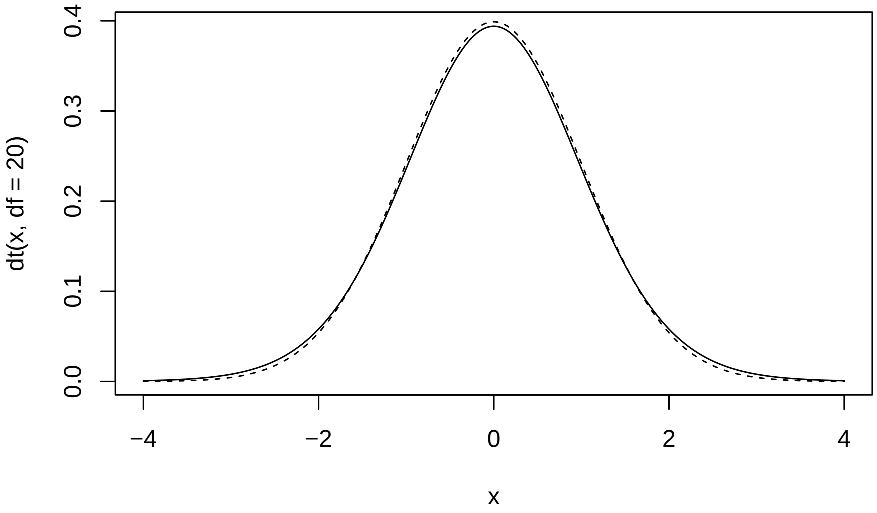
Note also that, as n approaches infinity, $T \sim N(0,1)$.

Thus, given a sample size n, and given our null hypothesis, we can draw t-distribution corresponding to the null hypothesis distribution.

For large n, we could even use N(0,1), although it is traditional to always use the t-distribution no matter how large n is.

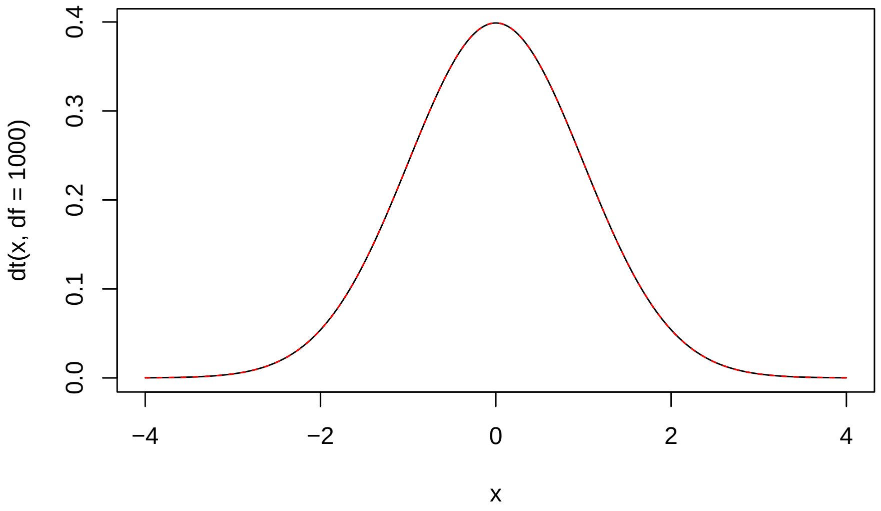
Compare the t-distribution t(21) (solid line) with Normal(0,1) (broken line).

```r
x<-seq(-4,4,by=0.01)
plot(x,dt(x,df=20),type="l")
lines(x,dnorm(x),lty=2)
```



Now compare the t-distribution t(1000) (solid line) with Normal(0,1) (broken line).

```r
plot(x,dt(x,df=1000),type="l")
lines(x,dnorm(x),lty=2,col="red")
```



The hypothesis testing procedure

So, the null hypothesis testing procedure is:

- Define the null hypothesis: for example, $H_0 : \mu = 0$.
- Given data of size n, estimate $\bar{x}$, standard deviation s, standard error $s/\sqrt{n}$.
- Compute the observed t-value:

$$t_{observed} = \frac{\bar{x} - \mu}{s/\sqrt{n}} \tag{12}$$

- Reject null hypothesis if the observed t-value is large (defined below).

Rejection region

So, for large sample sizes, if $\mid t \mid > 2$ (approximately), we can reject the null hypothesis.

For a smaller sample size n (say 42), you can compute the exact critical t-value:

```r
n<-42
qt(0.025,df=n-1)
```

```
## [1] -2.019541
```

This is the **critical t-value** on the **left**-hand side of the t-distribution. The corresponding value on the right-hand side is:

```
qt(0.975,df=n-1)
```

```
## [1] 2.019541
```

Their absolute values are of course identical (the distribution is symmetric when the t-distribution is centered on 0).

R syntax you should know

Given iid data (Note: aggregated!):

```
OR<-subset(agg_gg05,condition=="objgap")$RT
SR<-subset(agg_gg05,condition=="subjgap")$RT
diff<-OR-SR
## one sample t-test:
t.test(diff)
```

```
##
##  One Sample t-test
##
## data:  diff
## t = 3.1093, df = 41, p-value = 0.003404
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##   35.85024 168.72119
## sample estimates:
## mean of x
##  102.2857
```

```
## paired t-test:
t.test(OR,SR,paired=TRUE)
```

```
##
```

```
##  Paired t-test
##
## data:  OR and SR
## t = 3.1093, df = 41, p-value = 0.003404
## alternative hypothesis: true mean difference is not equal to
## 95 percent confidence interval:
##   35.85024 168.72119
## sample estimates:
## mean difference
##        102.2857
```

You should know when to aggregate data to meet the one sample (=paired) t-test's assumptions.

A very common mistake is to forget or neglect to aggregate the data. The following is wrong:

```r
OR<-subset(df_gg05_rc,condition=="objgap")$RT
SR<-subset(df_gg05_rc,condition=="subjgap")$RT
diff<-OR-SR
## one sample t-test (WRONG):
t.test(diff)
```

```
##
##  One Sample t-test
##
## data:  diff
## t = 3.9997, df = 335, p-value = 7.81e-05
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##   51.98059 152.59084
## sample estimates:
## mean of x
##  102.2857
```

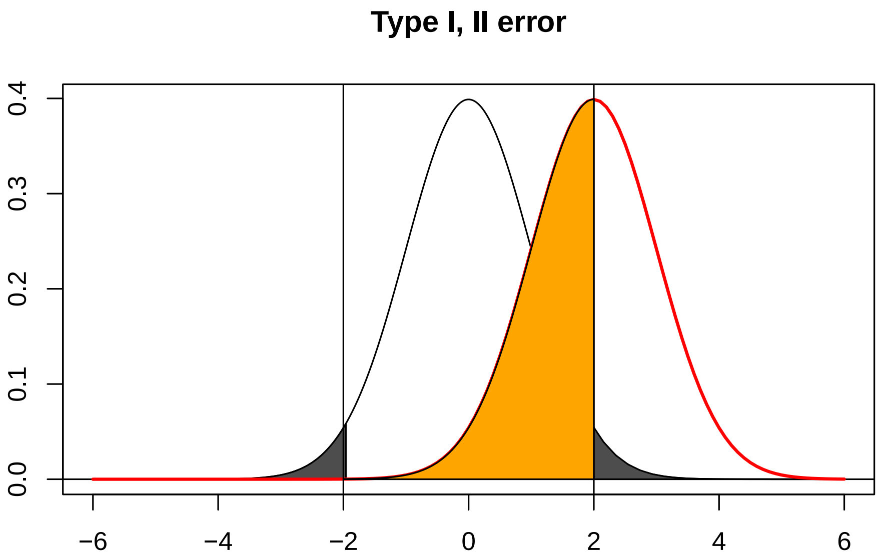
```r
## paired t-test (WRONG):
t.test(OR,SR,paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  OR and SR
## t = 3.9997, df = 335, p-value = 7.81e-05
## alternative hypothesis: true mean difference is not equal to
## 95 percent confidence interval:
##   51.98059 152.59084
## sample estimates:
## mean difference
##        102.2857
```

Look at the degrees of freedom–they are wrong (we have only 42 subjects, so it should have been 41).

Type I, II error, power

Reality:	H_0 TRUE	H_0 FALSE
Decision: 'reject':	α **Type I error**	$1 - \beta$ **Power**
Decision: 'fail to reject':	$1-\alpha$	β **Type II error**



Computing power

Power, which is calculated **before** a study is conducted, is a function of three variables:

- effect size
- standard deviation
- sample size

A quick way to get a ballpark estimate of power is by using the `power.t.test` function in R.

Example: what sample size do we need in a standard within-subjects design (like the two-condition relative clause study mentioned above) to reach 80% power if the true effect size were 15 ms, with a standard deviation of 150 ms?

```
power.t.test(n=NULL,
             delta=15,
             sd=150,
             sig.level=0.05,
             power=0.80,
             alternative="two.sided",
             type="one.sample",
             strict=TRUE)
```

```
##
##      One-sample t test power calculation
##
##               n = 786.8089
##           delta = 15
##              sd = 150
##       sig.level = 0.05
##           power = 0.8
##     alternative = two.sided
```

In this example, something close to 800 subjects would be needed to achieve 80% power.

For more complex designs, we use simulation to compute power. See my frequentist textbook draft for more:

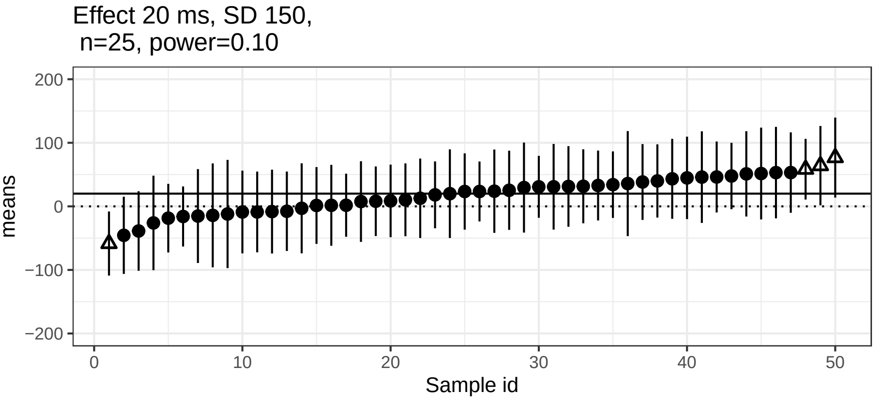
https://vasishth.github.io/Freq_CogSci/

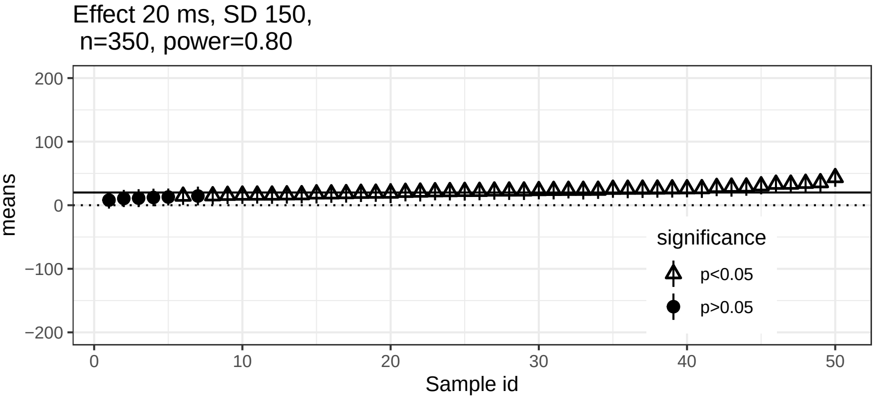
Type M error

If your true effect size is believed to be D, then we can compute (apart from statistical power) this error rate, which is defined as follows:

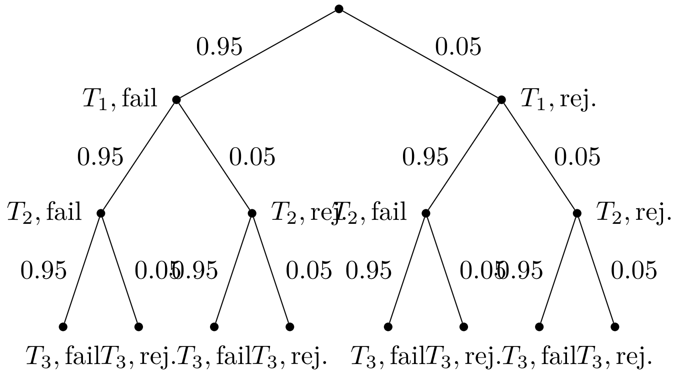
Type M error: the expectation of the ratio of the absolute magnitude of the effect to the hypothesized true effect size, given that result is significant. Gelman and Carlin also call this the exaggeration ratio, which is perhaps more descriptive than “Type M error.’ ’

Here’s a visualization of Type M error in action, under low statistical power.





Multiple comparisons inflate Type I error



The theoretical probability of rejecting at least one test incorrectly: $1 - 0.95^3 = 0.143$.

- It is common practice in linguistics, psychology, and other areas, to carry out multiple t-tests/ANOVA comparisons, fixing Type I error at 0.05.
- It seems to be not well-understood (even among established scientists) that multiple

comparisons will inflate Type I error.

Example: $29 \times 5 = 145$ **tests**

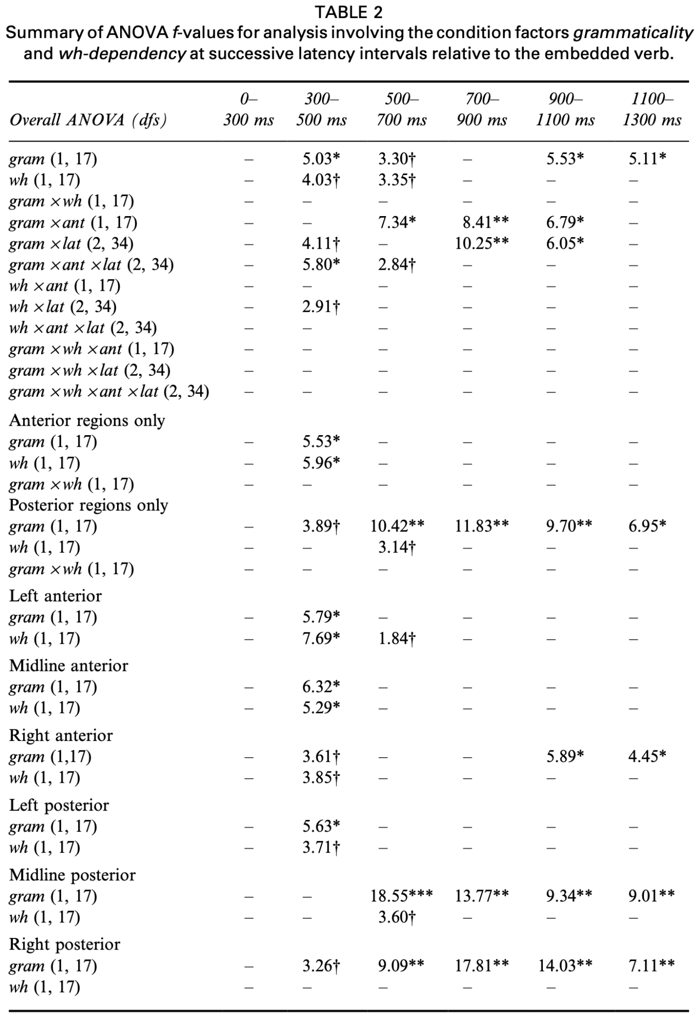


TABLE 2
Summary of ANOVA *f*-values for analysis involving the condition factors *grammaticality* and *wh-dependency* at successive latency intervals relative to the embedded verb.

Overall ANOVA (dfs)	*0– 300 ms*	*300– 500 ms*	*500– 700 ms*	*700– 900 ms*	*900– 1100 ms*	*1100– 1300 ms*
gram (1, 17)	–	5.03*	3.30†	–	5.53*	5.11*
wh (1, 17)	–	4.03†	3.35†	–	–	–
gram ×*wh* (1, 17)	–	–	–	–	–	–
gram ×*ant* (1, 17)	–	–	7.34*	8.41**	6.79*	–
gram ×*lat* (2, 34)	–	4.11†	–	10.25**	6.05*	–
gram ×*ant* ×*lat* (2, 34)	–	5.80*	2.84†	–	–	–
wh ×*ant* (1, 17)	–	–	–	–	–	–
wh ×*lat* (2, 34)	–	2.91†	–	–	–	–
wh ×*ant* ×*lat* (2, 34)	–	–	–	–	–	–
gram ×*wh* ×*ant* (1, 17)	–	–	–	–	–	–
gram ×*wh* ×*lat* (2, 34)	–	–	–	–	–	–
gram ×*wh* ×*ant* ×*lat* (2, 34)	–	–	–	–	–	–
Anterior regions only						
gram (1, 17)	–	5.53*	–	–	–	–
wh (1, 17)	–	5.96*	–	–	–	–
gram ×*wh* (1, 17)	–	–	–	–	–	–
Posterior regions only						
gram (1, 17)	–	3.89†	10.42**	11.83**	9.70**	6.95*
wh (1, 17)	–	–	3.14†	–	–	–
gram ×*wh* (1, 17)	–	–	–	–	–	–
Left anterior						
gram (1, 17)	–	5.79*	–	–	–	–
wh (1, 17)	–	7.69*	1.84†	–	–	–
Midline anterior						
gram (1, 17)	–	6.32*	–	–	–	–
wh (1, 17)	–	5.29*	–	–	–	–
Right anterior						
gram (1,17)	–	3.61†	–	–	5.89*	4.45*
wh (1, 17)	–	3.85†	–	–	–	–
Left posterior						
gram (1, 17)	–	5.63*	–	–	–	–
wh (1, 17)	–	3.71†	–	–	–	–
Midline posterior						
gram (1, 17)	–	–	18.55***	13.77**	9.34**	9.01**
wh (1, 17)	–	–	3.60†	–	–	–
Right posterior						
gram (1, 17)	–	3.26†	9.09**	17.81**	14.03**	7.11**
wh (1, 17)	–	–	–	–	–	–

Figure 1: Gouvea et al 2010. Language and Cognitive Processes

Source: Gouvea et al 2010. Language and Cognitive Processes.

Example: 20 tests

Source: Liversedge et al., 2024. Cognition.

Example: At least 18 tests (probably more)

All my own work, published during the period 2002 to 2016, has this Type I inflation problem.

Source: Vasishth and Lewis, 2006. Language.

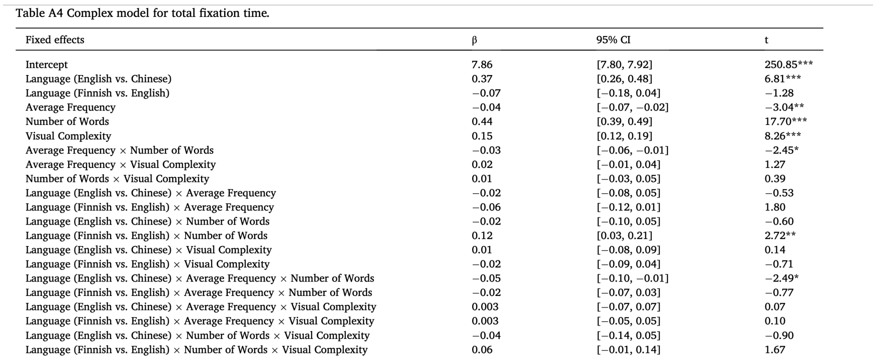
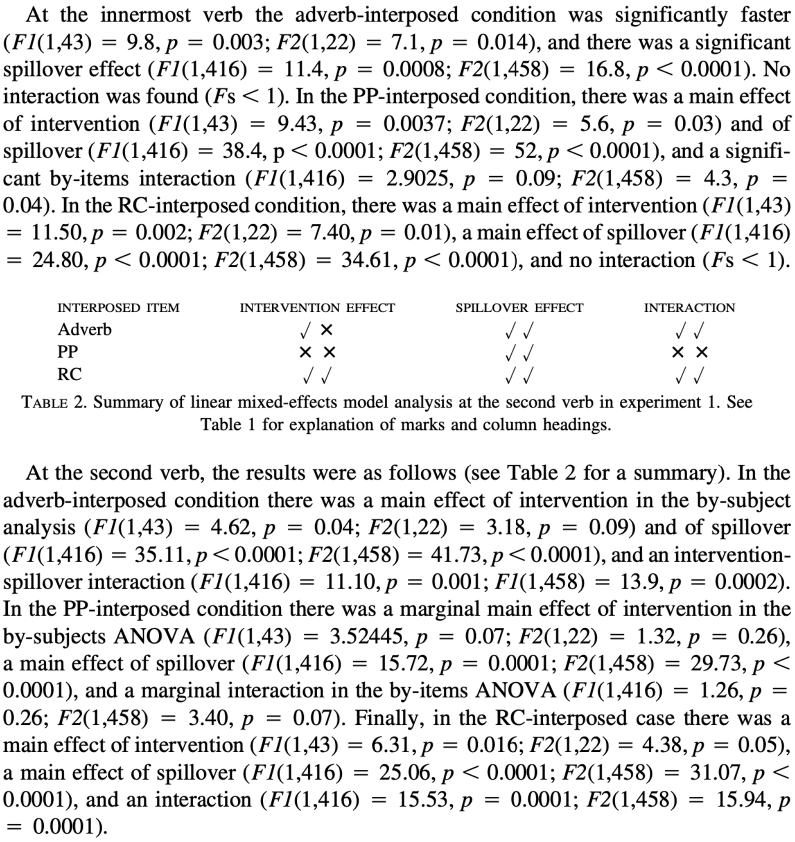


Table A4 Complex model for total fixation time.

Fixed effects	β	95% CI	t
Intercept	7.86	[7.80, 7.92]	250.85***
Language (English vs. Chinese)	0.37	[0.26, 0.48]	6.81***
Language (Finnish vs. English)	−0.07	[−0.18, 0.04]	−1.28
Average Frequency	−0.04	[−0.07, −0.02]	−3.04**
Number of Words	0.44	[0.39, 0.49]	17.70***
Visual Complexity	0.15	[0.12, 0.19]	8.26***
Average Frequency × Number of Words	−0.03	[−0.06, −0.01]	−2.45*
Average Frequency × Visual Complexity	0.02	[−0.01, 0.04]	1.27
Number of Words × Visual Complexity	0.01	[−0.03, 0.05]	0.39
Language (English vs. Chinese) × Average Frequency	−0.02	[−0.08, 0.05]	−0.53
Language (Finnish vs. English) × Average Frequency	−0.06	[−0.12, 0.01]	1.80
Language (English vs. Chinese) × Number of Words	−0.02	[−0.10, 0.05]	−0.60
Language (Finnish vs. English) × Number of Words	0.12	[0.03, 0.21]	2.72**
Language (English vs. Chinese) × Visual Complexity	0.01	[−0.08, 0.09]	0.14
Language (Finnish vs. English) × Visual Complexity	−0.02	[−0.09, 0.04]	−0.71
Language (English vs. Chinese) × Average Frequency × Number of Words	−0.05	[−0.10, −0.01]	−2.49*
Language (Finnish vs. English) × Average Frequency × Number of Words	−0.02	[−0.07, 0.03]	−0.77
Language (English vs. Chinese) × Average Frequency × Visual Complexity	0.003	[−0.07, 0.07]	0.07
Language (Finnish vs. English) × Average Frequency × Visual Complexity	0.003	[−0.05, 0.05]	0.10
Language (English vs. Chinese) × Number of Words × Visual Complexity	−0.04	[−0.14, 0.05]	−0.90
Language (Finnish vs. English) × Number of Words × Visual Complexity	0.06	[−0.01, 0.14]	1.67

Figure 2: Liversedge et al., 2024. Cognition.



At the innermost verb the adverb-interposed condition was significantly faster ($F1$(1,43) = 9.8, p = 0.003; $F2$(1,22) = 7.1, p = 0.014), and there was a significant spillover effect ($F1$(1,416) = 11.4, p = 0.0008; $F2$(1,458) = 16.8, p < 0.0001). No interaction was found (Fs < 1). In the PP-interposed condition, there was a main effect of intervention ($F1$(1,43) = 9.43, p = 0.0037; $F2$(1,22) = 5.6, p = 0.03) and of spillover ($F1$(1,416) = 38.4, p < 0.0001; $F2$(1,458) = 52, p < 0.0001), and a significant by-items interaction ($F1$(1,416) = 2.9025, p = 0.09; $F2$(1,458) = 4.3, p = 0.04). In the RC-interposed condition, there was a main effect of intervention ($F1$(1,43) = 11.50, p = 0.002; $F2$(1,22) = 7.40, p = 0.01), a main effect of spillover ($F1$(1,416) = 24.80, p < 0.0001; $F2$(1,458) = 34.61, p < 0.0001), and no interaction (Fs < 1).

INTERPOSED ITEM	INTERVENTION EFFECT	SPILLOVER EFFECT	INTERACTION
Adverb	✓ ✕	✓ ✓	✓ ✓
PP	✕ ✕	✓ ✓	✕ ✕
RC	✓ ✓	✓ ✓	✓ ✓

TABLE 2. Summary of linear mixed-effects model analysis at the second verb in experiment 1. See Table 1 for explanation of marks and column headings.

At the second verb, the results were as follows (see Table 2 for a summary). In the adverb-interposed condition there was a main effect of intervention in the by-subject analysis ($F1$(1,43) = 4.62, p = 0.04; $F2$(1,22) = 3.18, p = 0.09) and of spillover ($F1$(1,416) = 35.11, p < 0.0001; $F2$(1,458) = 41.73, p < 0.0001), and an intervention-spillover interaction ($F1$(1,416) = 11.10, p = 0.001; $F1$(1,458) = 13.9, p = 0.0002). In the PP-interposed condition there was a marginal main effect of intervention in the by-subjects ANOVA ($F1$(1,43) = 3.52445, p = 0.07; $F2$(1,22) = 1.32, p = 0.26), a main effect of spillover ($F1$(1,416) = 15.72, p = 0.0001; $F2$(1,458) = 29.73, p < 0.0001), and a marginal interaction in the by-items ANOVA ($F1$(1,416) = 1.26, p = 0.26; $F2$(1,458) = 3.40, p = 0.07). Finally, in the RC-interposed case there was a main effect of intervention ($F1$(1,43) = 6.31, p = 0.016; $F2$(1,22) = 4.38, p = 0.05), a main effect of spillover ($F1$(1,416) = 25.06, p < 0.0001; $F2$(1,458) = 31.07, p < 0.0001), and an interaction ($F1$(1,416) = 15.53, p = 0.0001; $F2$(1,458) = 15.94, p = 0.0001).

Figure 3: Vasishth and Lewis, 2006. Language.

Demonstration using simulation

If we do a single t-test when the null is actually true, our Type I error is 0.05:

```
nsim<-1000
pvals<-rep(NA,nsim)
for(i in 1:nsim){
  y<-rnorm(10,mean=0,sd=1)
  pvals[i]<-t.test(y)$p.value
}
mean(pvals<0.05)
```

```
## [1] 0.06
```

If we do two t-tests when the null in all the analyses is actually true, our Type I error is no longer 0.05:

```
nsim<-1000
ntests<-2
pvals<-matrix(rep(NA,nsim*ntests),ncol=ntests)
 for(j in 1:ntests){
     for(i in 1:nsim){
     y<-rnorm(10,mean=0,sd=1)
     pvals[i,j]<-t.test(y)$p.value
  }
 }

head(pvals)
```

```
##           [,1]       [,2]
## [1,] 0.2227763 0.04618541
## [2,] 0.7433136 0.97112086
## [3,] 0.8559875 0.58501021
## [4,] 0.3284245 0.90432025
## [5,] 0.8683764 0.69789140
## [6,] 0.4849970 0.86804439
```

What is the probability that *at least one of the two tests comes out significant* despite the null being true in both cases?

```r
sig<-rep(NA,nsim)
for(i in 1:nsim){
if(pvals[i,1]<0.05 | pvals[i,2]<0.05){
  sig[i]<-1
} else {
  sig[i]<-0
}
}

## The probability that at least
## one t-test comes out significant:
mean(sig>0)
```

```
## [1] 0.103
```

This inflation of Type I error is called the multiple comparisons problem.

The more t-tests/F-tests you do, the higher the Type I error.

Let's write a function that computes the Type I error when we do n hypothesis tests, where n can be 1,2,3,. . .

The full function is not visible here (see the R source code).

Let's test the function with some ntest values.

```
## ntests=1
computeTypeI(ntests=1)
```

```
## [1] 0.048
```

```
## ntests=2
computeTypeI(ntests=2)
```

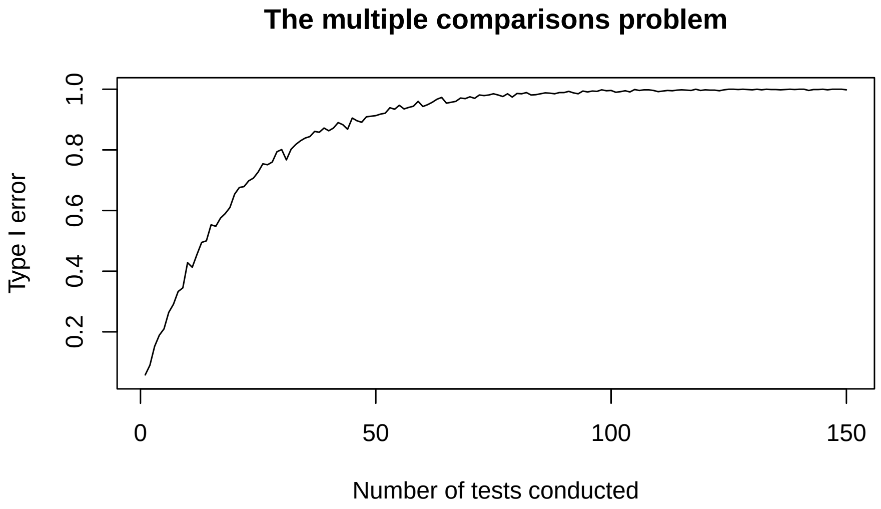
```
## [1] 0.105
```

```
## ntests=3
computeTypeI(ntests=3)
```

```
## [1] 0.142
```

Let's plot a figure showing how Type I error will inflate as we increase the number of tests:

```
n<-150
inflation<-rep(NA,n)
for(i in 1:n){
  inflation[i]<-computeTypeI(ntests=i)
}
```



So, once you have done some 100 statistical tests, you are basically **guaranteed** to obtain some significant effect or the other, even if the null were in fact true.

A solution to the multiple comparisons problem

If working in the frequentist framework, just do a Bonferroni correction. If you do n tests, the new α is 0.05/n.

Comparison to Bayes

In the classical Bayesian framework, there is no concept of Type I/II error. There, the focus is rather on **uncertainty quantification**. More on that later.

Of course, one can think about the frequentist properties of Bayesian hypothesis tests; in that approach, you will run into the same Type I error inflation problems as in the classical frequentist approach.

Linear models

We consider the case where we have two conditions (e.g., subject and object relatives), and a repeated measures design. The dependent measure is reading times in milliseconds.

If you are not familiar with relative clauses, just imagine doing an experiment with two types of sentences, an easy-to-read sentence type and a difficult-to-read sentence type, and measuring reading time difference between the hard and easy conditions.

Treatment contrast coding

The alphabetically first condition level is coded 0, and the other condition level is coded 1. E.g., if condition labels are objgap and subjgap, then objgap is coded 0 and subjgap 1. You can change this with the command (not run):

```
## this code has not been run:
## code subj as 0 and obj as 1:
df_gg05_rc$condition<-
  factor(df_gg05_rc$condition,
                                    levels=c("subjgap","objgap"))
```

In mathematical form, the linear model is:

$$rt = \beta_0 + \beta_1 condition + \epsilon \qquad (13)$$

where

- β_0 is the mean for the object relative
- *condition& has value 0 (object relative) or 1 (subject relative)
- β_1 is the amount by which the object relative mean must be changed to obtain the mean for the subject relative.

```
agg_gg05$condition<-factor(agg_gg05$condition)
contrasts(agg_gg05$condition)
```

```
##         subjgap
## objgap        0
## subjgap       1
```

```
## this model is wrong for these data:
m<-lm(RT ~ condition, agg_gg05)
round(summary(m)$coefficients,2)
```

```
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)          471.36      31.13   15.14     0.00
## conditionsubjgap  -102.29      44.02   -2.32     0.02
```

The null hypothesis of interest is that the difference in means between the two relative clause types β_1 is:

$H_0 : \beta_1 = 0$

We will make a distinction between the **unknown true mean** β_0, β_1 and the **estimated mean from the data** $\hat{\beta}_0, \hat{\beta}_1$. These estimated means are maximum likelihood estimates of the parameters.

- Estimated mean object relative processing time: $\hat{\beta}_0 = 471$ ms.
- Estimated mean subject relative processing time: $\hat{\beta}_0 + \hat{\beta}_1 = 471 + (-102) = 369$.

Sum contrast coding

Alternatively, we can code objgap as $+1$ and subjgap as -1 (or vice versa).

Equivalently: objgap as $+1/2$ and subjgap as $-1/2$ (or vice versa).

With ± 1 coding:

```
agg_gg05$so<-ifelse(agg_gg05$condition=="objgap",
                     1,-1)

## this model is wrong:
m_sum<-lm(RT~so,agg_gg05)
round(summary(m_sum)$coefficients,2)
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   420.22       22.01   19.09      0.00
## so             51.14       22.01    2.32      0.02
```

- Estimated **grand mean** processing time: $\hat{\beta}_0 = 420$ ms.
- Estimated mean object relative processing time: $\hat{\beta}_0 + \hat{\beta}_1 = 420 + 1 \times 51 = 471$.
- Estimated mean subject relative processing time: $\hat{\beta}_0 - \hat{\beta}_1 = 420 + (-1) \times 51 = 369$.

This kind of parameterization is called **sum-to-zero contrast** or more simply **sum contrast** coding. This is the coding we will use.

The null hypothesis for the slope is

$$H_0 : \mathbf{1}\times\mu_{obj} + (-\mathbf{1}\times)\mu_{subj} = 0 \tag{14}$$

or:

$$H_0 : \mu_{obj} = \mu_{subj} \tag{15}$$

The sum contrasts are referring to the ±1 terms in the null hypothesis:

- object relative: +1
- subject relative: -1

The model is:

Estimated object relative reading times:

$$rt = 420\times\mathbf{1} + 51\times\mathbf{1} \tag{16}$$

Estimated subject relative reading times:

$$rt = 420\times\mathbf{1} + 51\times - \mathbf{1} \tag{17}$$

The ϵ has been dropped here because the mean of the random variable ϵ is 0.

The normality assumption of the residuals in the linear models

The model is:

$$rt = \beta_0 + \beta_1 + \epsilon \text{ where } \epsilon \sim Normal(0, \sigma) \quad (18)$$

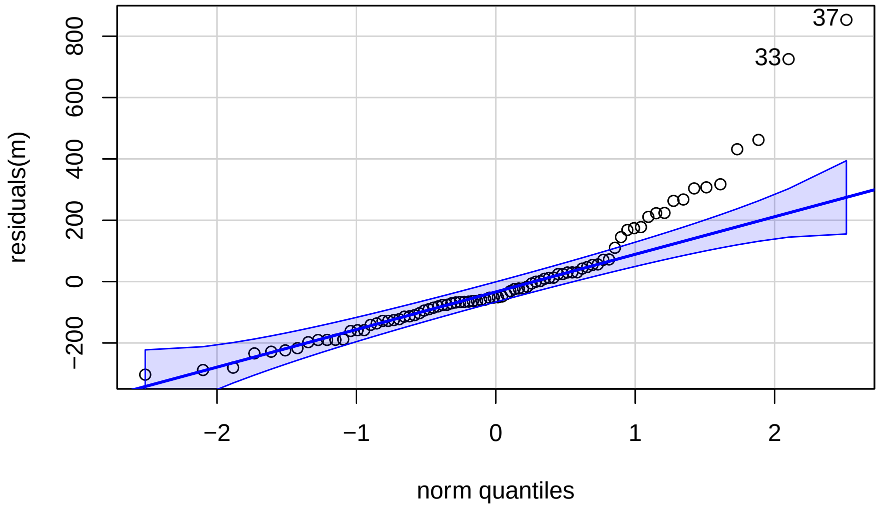
It is an assumption of the linear model that the residuals are (approximately) normally distributed.

This assumption is not crucial if our goal is only to estimate the parameters.

However, this assumption is crucial if we are doing hypothesis testing.

We can check this assumption in R. **This model is wrong for these data**.

```
m<-lm(RT ~ condition, agg_gg05)
car::qqPlot(residuals(m))
```

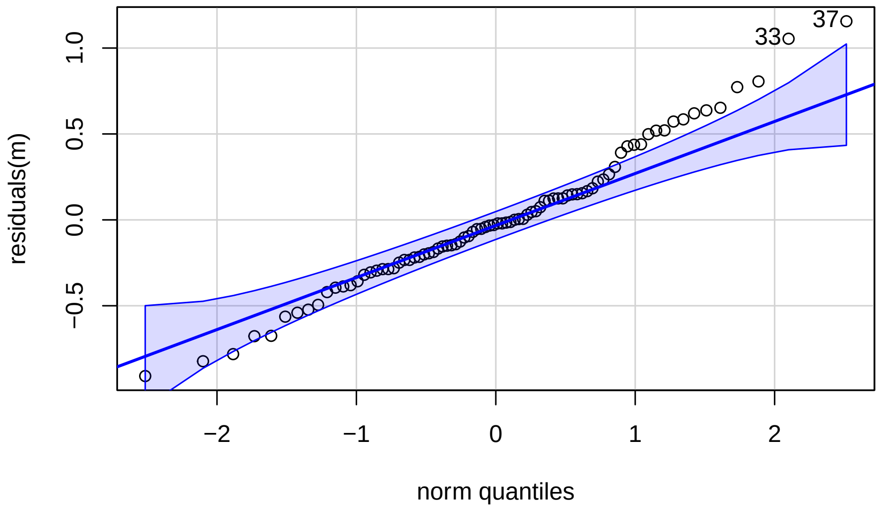


```
## [1] 37 33
```

If the residuals were approximately normally distributed, the quantiles of the standard normal and the residuals would align, leading to a diagonal line angled at 45 degrees (not the case here).

A log-transform would improve the situation here:

```
m<-lm(log(RT) ~ condition, agg_gg05)
car::qqPlot(residuals(m))
```



```
## [1] 37 33
```

Linear mixed models

The correct model for the **aggregated** data:

```
library(lme4)
```

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

```
m1<-lmer(RT ~ condition + (1|subj), agg_gg05)
## compare with the paired t-test result above!
## They are exactly the same.
summary(m1)$coefficients
```

```
##                    Estimate Std. Error   t value
## (Intercept)        471.3601   31.12777 15.142753
## conditionsubjgap -102.2857   32.89632 -3.109336
```

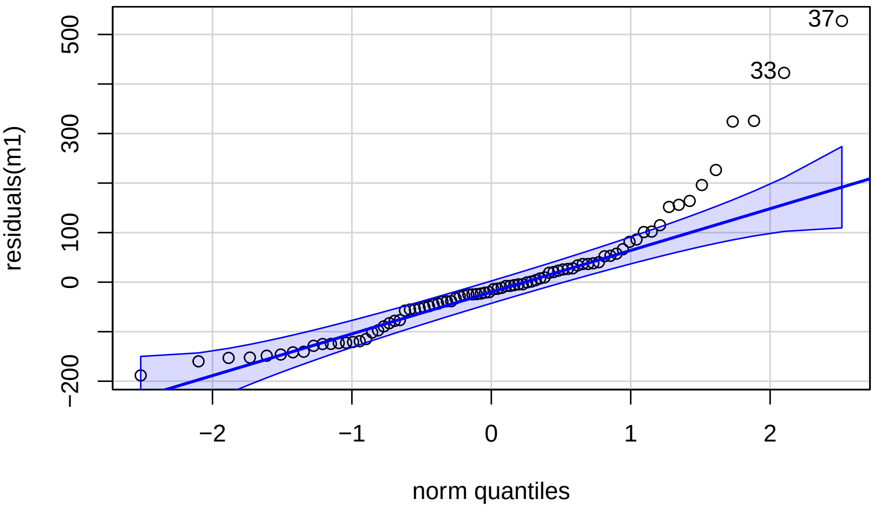
```
OR<-subset(agg_gg05,condition=="objgap")$RT
SR<-subset(agg_gg05,condition=="subjgap")$RT
diff<-SR-OR
## one sample t-test:
t.test(diff)
```

```
##
##  One Sample t-test
##
## data:  diff
## t = -3.1093, df = 41, p-value = 0.003404
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -168.72119  -35.85024
## sample estimates:
## mean of x
## -102.2857
```

The linear mixed model with varying intercepts (on the aggregated data) is exactly the one-sample (paired) t-test.

Residuals check:

```
car::qqPlot(residuals(m1))
```



```
## [1] 37 33
```

The correct way to analyze these data are using linear mixed models on the **unaggregated** data, and carrying out a log transform on the data:

```
df_gg05_rc$so<-ifelse(df_gg05_rc$condition=="objgap",
                                 1,-1)
m2<-lmer(log(RT) ~ so +
              (1+so|subj) + (1+so| item),
           df_gg05_rc)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(m2)$coefficients
```

```
##               Estimate Std. Error   t value
## (Intercept) 5.88305598 0.05202442 113.08258
## so          0.06201673 0.02466207   2.51466
```

The convergence warning is due to data sparsity, leading to an inability to estimate the varying intercepts/slopes correlation for items.

For a full and formal review of linear models (including linear mixed modeling), see:

https://github.com/vasishth/LM