Issues and Solutions in Fitting, Evaluating, and Interpreting Regression Models

Florian Jaeger and Victor Kuperman

July 8, 2009

Hypothesis testing in psycholinguistic research

- Typically, we make predictions not just about the existence, but also the *direction* of effects.
- Sometimes, we're also interested in effect *shapes* (non-linearities, etc.)
- Regression analyses reliably test hypotheses about effect direction and shape without requiring post-hoc analyses if (a) *the predictors in the model are coded appropriately* and (b) *the model can be trusted*.
- **Next:** Provide an overview of (a) and (b).

Overview

- **Introduce sample data and simple models**
- **Towards a model with interpretable coefficients:**
  - outlier removal
  - transformation
  - coding, centering, ...
  - *collinearity*
- **Model evaluation:**
  - fitted vs. observed values
  - model validation
  - investigation of residuals
  - case influence, outliers
- **Model comparison**
- **Reporting the model:**
  - comparing effect sizes
  - back-transformation of predictors
  - visualization

Sample Data and Simple Models

Data: Lexical decision response

- **Outcome:** Correct or incorrect response (`Correct`)
- **Inputs:** same as in linear model

```
> lmer(Correct == "correct" ~ NativeLanguage +
+                              Frequency + Trial +
+                             (1 | Subject) + (1 | Word),
+                  data = lexdec, family = "binomial")
Random effects:
 Groups  Name        Variance Std.Dev.
 Word    (Intercept) 1.01820  1.00906
 Subject (Intercept) 0.63976  0.79985
Number of obs: 1659, groups: Word, 79; Subject, 21

Fixed effects:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.746e+00  8.206e-01  -2.128 0.033344 *
NativeLanguageOther -5.726e-01  4.639e-01   1.234 0.217104
Frequency             5.600e-01  1.570e-01  -3.567 0.000361 ***
Trial                 4.443e-06  2.965e-03   0.001 0.998804
```

- estimates for random effects of `Subject` and `Word` (no residuals).
- estimates for regression coefficients, standard errors → Z- and p-values

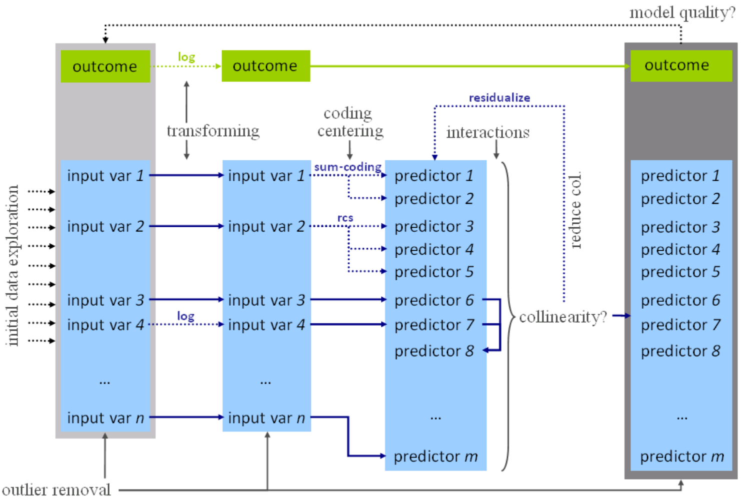
Interpretation of coefficients

- *In theory*, directionality and shape of effects can be tested and immediately interpreted.
  - e.g. logit model

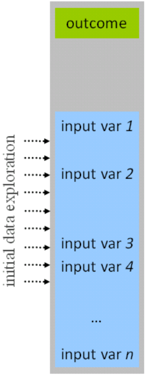
```
Fixed effects:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)         -1.746e+00  8.206e-01  -2.128 0.033344 *
NativeLanguageOther  5.726e-01  4.639e-01   1.234 0.217104
Frequency           -5.600e-01  1.570e-01  -3.567 0.000361 ***
Trial               -5.725e-06  2.965e-03  -0.002 0.998460
```

- . . . but can these coefficient estimates be trusted?

Modeling schema



Data exploration

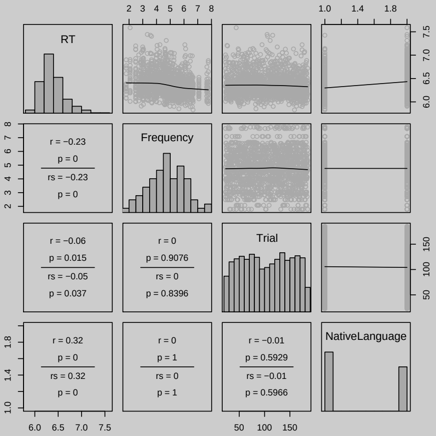


Data exploration

- Select and *understand* input variables and outcome based on a-priori theoretical consideration
  - How many parameters does your data afford (**overfitting**)?
- Data exploration: *Before* fitting the model, explore inputs and outputs
  - Outliers due to missing data or measurement error (e.g. RTs in SPR < 80msecs).
  - **NB:** postpone distribution-based outlier exclusion until after **transformations**)
  - Skewness in distribution can affect the accuracy of model's estimates (**transformations**).

Understanding input variables

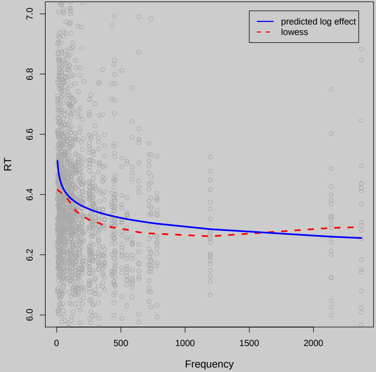
- Explore:
  - correlations between predictors (↷**collinearity**).
  - non-linearities may become obvious (lowess).




```
> pairscor.fnc(lexdec[,c("RT", "Frequency", "T
```

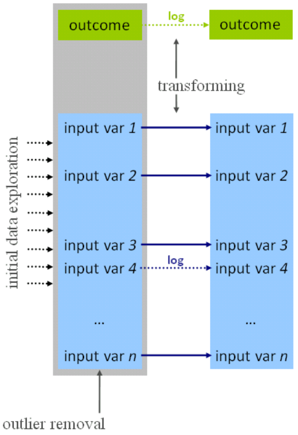
Non-linearities

- Consider `Frequency` (already log-transformed in `lexdec`) as predictor of `RT`:



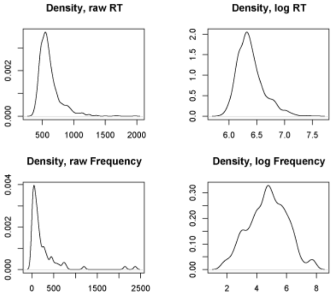
→ Assumption of a linearity may be inaccurate.
  - Select appropriate ↷**transformation**: log, power, sinusoid, etc.
  - or use polynomial `poly()` or splines `rcs()`, `bs()`, etc. to ↷**model non-linearities**.

Transformation

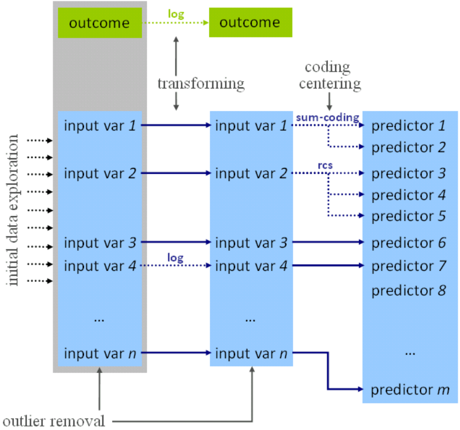


Transformation

- Reasons to transform:
  - Conceptually motivated (e.g. log-transformed probabilities)
  - Can reduce non-linear to linear relations (cf. previous slide)
  - Remove skewness (e.g. by log-transform)
- Common transformation: log, square-root, power, or inverse transformation, etc.



Coding and centering predictors



Coding affects interpretation

Consider a simpler model:

```
> lmer(RT ~ NativeLanguage +
+          (1 | Word) + (1 | Subject), data = lexdec)
    AIC    BIC logLik deviance REMLdev
 -886.1 -853.6  449.1   -926.6  -898.1
Random effects:
 Groups   Name        Variance  Std.Dev.
 Word     (Intercept) 0.0045808 0.067682
 Subject  (Intercept) 0.0184681 0.135897
 Residual             0.0298413 0.172746
Number of obs: 1659, groups: Word, 79; Subject, 21

Fixed effects:
                   Estimate Std. Error t value
(Intercept)         6.32358    0.03783  167.14
NativeLanguageOther 0.15003    0.05646    2.66
```

- **Treatment (a.k.a. dummy) coding** is standard in most stats programs
  - `NativeLanguage` coded as 1 if "`other`", 0 otherwise.
  - Coefficient for `(Intercept)` reflects reference level `English` of the factor `NativeLanguage`.
  - Prediction for `NativeLanguage` = `Other` is derived by $6.32358 + 0.15003 = 6.47361$ (log-transformed reaction times).

Recoding

- Coding affects interpretation of coefficients.
- E.g., we can recode `NativeLanguage` into `NativeEnglish`:

```
> lexdec$NativeEnglish = ifelse(lexdec$NativeLanguage == "English", 1, 0)
> lmer(RT ~ NativeEnglish + Frequency +
+           (1 | Word) + (1 | Subject), data = lexdec)
<...>
    AIC    BIC logLik deviance REMLdev
 -886.1 -853.6  449.1   -926.6  -898.1
Random effects:
 Groups   Name        Variance  Std.Dev.
 Word     (Intercept) 0.0045808 0.067682
 Subject  (Intercept) 0.0184681 0.135897
 Residual             0.0298413 0.172746
Number of obs: 1659, groups: Word, 79; Subject, 21

Fixed effects:
                    Estimate Std. Error t value
(Intercept)          6.32358    0.03783  167.14
NativeEnglish       -0.15003    0.05646    2.66
<...>
```

- NB: **Goodness-of-fit** (AIC, BIC, loglik, etc.) is *not* affected by choice between different sets of *orthogonal contrasts*.

Other codings of factor

- Treatment coding ...
  - makes intercept hard to interpret.
  - leads to **collinearity** with interactions
- Sum (a.k.a. contrast) coding avoids that problem (in balanced data sets) and makes intercept interpretable (in factorial analyses of balanced data sets).
  - Corresponds to ANOVA coding.
  - Centers for balanced data set.
  - Caution when **reporting effect sizes**! (R contrast codes as -1 vs. 1 $\rightarrow$ coefficient estimate is only half of estimated group difference).
- Other contrasts possible, e.g. to test hypothesis that levels are ordered (`contr.poly()`, `contr.helmert()`).

Centering predictors

- **Centering:** removal of the mean out of a variable . . .
  - makes coefficients more interpretable.
  - if all predictors are centered $\rightarrow$ intercept is estimated grand mean.
  - reduces ↷**collinearity** of predictors
    - *with intercept*
    - *higher-order terms that include the predictor* (e.g. interactions)
- **Centering** does not change . . .
  - coefficient estimates (it's a linear transformations); including random effect estimates.
  - ↷**Goodness-of-fit** of model (information in the model is the same)

Centering: An example

- Re-consider the model with `NativeEnglish` and `Frequency`. Now with a centered predictors:

```
> lexdec$cFrequency = lexdec$Frequency - mean(lexdec$Frequency)
> lmer(RT ~ cNativeEnglish + cFrequency +
+          (1 | Word) + (1 | Subject), data = lexdec)
<...>
Fixed effects:
                Estimate Std. Error t value
(Intercept)     6.385090   0.030570  208.87
cNativeEnglish -0.155821   0.060532   -2.57
cFrequency     -0.042872   0.005827   -7.36

Correlation of Fixed Effects:
            (Intr) cNtvEn
cNatvEnglsh 0.000
cFrequency  0.000  0.000
<...>
```

$\longrightarrow$ Correlation between predictors and intercept gone.

$\longrightarrow$ Intercept changed (from 6.678 to 6.385 units): now grand mean (previously: prediction for `Frequency`=0!)

$\longrightarrow$ `NativeEnglish` and `Frequency` coefs unchanged.

Centering: An interaction example

- Let's add an interaction between `NativeEnglish` and `Frequency`.
- Prior to centering: interaction is collinear with main effects.

```
> lmer(RT ~ NativeEnglish * Frequency +
+          (1 | Word) + (1 | Subject), data = lexdec)
<...>
Fixed effects:
                         Estimate Std. Error t value
(Intercept)              6.752403   0.056810  118.86
NativeEnglish           -0.286343   0.068368   -4.19
Frequency               -0.058570   0.006969   -8.40
NativeEnglish:Frequency  0.027472   0.006690    4.11

Correlation of Fixed Effects:
            (Intr) NtvEng Frqncy
NativEnglsh -0.688
Frequency   -0.583  0.255
NtvEnglsh:F  0.320 -0.465 -0.549
<...>
```

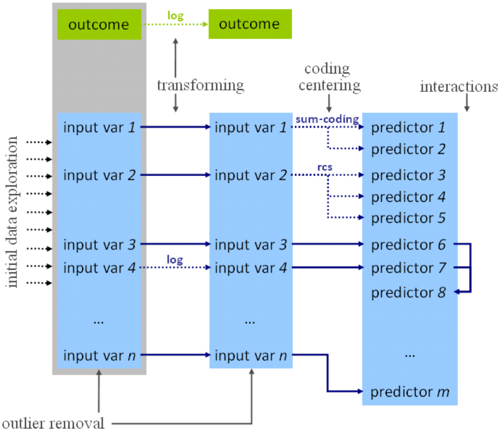
Centering: An interaction example (cnt'd)

- After centering:

```
<...>
Fixed effects:
                            Estimate Std. Error t value
(Intercept)                 6.385090   0.030572  208.85
cNativeEnglish             -0.155821   0.060531   -2.57
cFrequency                 -0.042872   0.005827   -7.36
cNativeEnglish:cFrequency   0.027472   0.006690    4.11

Correlation of Fixed Effects:
            (Intr) cNtvEn cFrqnc
cNatvEnglsh 0.000
cFrequency  0.000  0.000
cNtvEngls:F 0.000  0.000  0.000
<...>
```

Interactions and modeling of non-linearities



Interactions and non-linearities

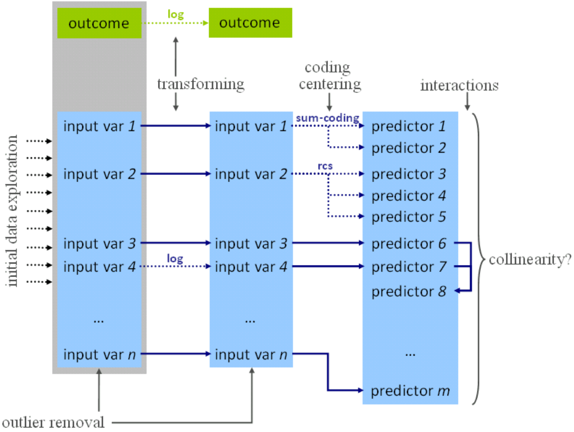
- Include interactions after variables are centered → avoids unnecessary ↷**collinearity**.
- The same holds for higher order terms when non-linearities in continuous (or ordered) predictors are modeled. Though often centering will not be enough.
  - See for yourself: a polynomial of (back-transformed) frequency

```
> lexdec$rawFrequency <- round(exp(lexdec$Frequency),0)
> lmer(RT ~ poly(rawFrequency,2) +
+          (1 | Word) + (1 | Subject), data = lexdec)
```

- ...vs. a polynomial of the centered (back-transformed) frequency

```
> lexdec$crawFrequency = lexdec$rawFrequency - mean(lexdec$rawFrequency)
> lmer(RT ~ poly(crawFrequency,2) +
+          (1 | Word) + (1 | Subject), data = lexdec)
```

Collinearity



Definition of collinearity

- **Collinearity**: a predictor is collinear with other predictors in the model if there are high (partial) correlations between them.
- Even if a predictor is not highly correlated with any single other predictor in the model, it can be highly collinear with the combination of predictors $\longrightarrow$ collinearity will affect the predictor
- This is not uncommon!
  - in models with many predictors
  - when several somewhat related predictors are included in the model (e.g. word length, frequency, age of acquisition)

Consequences of collinearity

$\longrightarrow$ standard errors $\mathrm{SE}(\beta)$s of collinear predictors are biased (*in*flated).

- $\longrightarrow$ *tends* to underestimate significance (but see below)

$\longrightarrow$ coefficients β of collinear predictors become hard to interpret (though not biased)

- 'bouncing betas': minor changes in data might have a major impact on βs
- coefficients will flip sign, double, half

$\longrightarrow$ coefficient-based tests don't tell us anything reliable about collinear predictors!

Extreme collinearity: An example

- **Drastic example of collinearity**: `meanWeight` (rating of the weight of the object denoted by the word, averaged across subjects) and `meanSize` (average rating of the object size) in `lexdec`.

```
lmer(RT ~ meanSize + (1 | Word) + (1 | Subject), data = lexdec)
Fixed effects:
              Estimate Std. Error t value
(Intercept)  6.3891053  0.0427533  149.44
meanSize    -0.0004282  0.0094371   -0.05
```

- n.s. correlation of `meanSize` with RTs.
- similar n.s. weak negative effect of `meanWeight`.
- The two predictors are highly correlated (r$>$ 0.999).

Extreme collinearity: An example (cnt'd)

- If the two correlated predictors are included in the model ...

```
> lmer(RT ~ meanSize + meanWeight +
+           (1 | Word) + (1 | Subject), data = lexdec)
Fixed effects:
            Estimate Std. Error t value
(Intercept)   5.7379     0.1187   48.32
meanSize      1.2435     0.2138    5.81
meanWeight   -1.1541     0.1983   -5.82

Correlation of Fixed Effects:
           (Intr) meanSz
meanSize   -0.949
meanWeight  0.942 -0.999
```

- $\mathrm{SE}(\beta)$s are hugely inflated (more than by a factor of 20)
- large and highly significant significant counter-directed effects (βs) of the two predictors

$\longrightarrow$ collinearity needs to be investigated!

Extreme collinearity: An example (cnt'd)

- Objects that are perceived to be unusually heavy for their size tend to be more frequent ($\rightarrow$ accounts for 72% of variance in frequency).
- Both effects apparently disappear though when frequency is included in the model (but cf. ↷**residualization** $\rightarrow$ `meanSize` *or* `meanWeight` still has small expected effect beyond `Frequency`).

```
Fixed effects:
             Estimate Std. Error t value
(Intercept)   6.64846    0.06247  106.43
cmeanSize    -0.11873    0.35196   -0.34
cmeanWeight   0.13788    0.33114    0.42
Frequency    -0.05543    0.01098   -5.05
```

So what does collinearity do?

- Type II error increases → power loss

```
h <- function(n) {
   x <- runif(n)
   y <- x + rnorm(n,0,0.01)
   z <- ((x + y) / 2) + rnorm(n,0,0.2)

   m <- lm(z ~ x + y)
   signif.m.x <- ifelse(summary(m)$coef[2,4] < 0.05, 1, 0)
   signif.m.y <- ifelse(summary(m)$coef[3,4] < 0.05, 1, 0)

   mx <- lm(z ~ x)
   my <- lm(z ~ y)
   signif.mx.x <- ifelse(summary(mx)$coef[2,4] < 0.05, 1, 0)
   signif.my.y <- ifelse(summary(my)$coef[2,4] < 0.05, 1, 0)
   return(c(cor(x,y),signif.m.x,signif.m.y,signif.mx.x, signif.my.y))
}
result <- sapply(rep(M,n), h)
print(paste("x in combined model:", sum(result[2,])))
print(paste("y in combined model:", sum(result[3,])))
print(paste("x in x-only model:", sum(result[4,])))
print(paste("y in y-only model:", sum(result[5,])))
print(paste("Avg. correlation:", mean(result[1,])))
```

So what does collinearity do?

- Type II error increases → power loss
- Type I error does not increase much (5.165% Type I error for two predictors with $r > 0.9989$ in joined model vs. 5.25% in separate models; 20,000 simulation runs with 100 data points each)

```
set.seed(1)
n <- 100
M <- 20000
f <- function(n) {
 x <- runif(n)
 y <- x + rnorm(n,0,0.01)
 z <- rnorm(n,0,5)
 m <- lm(z ~ x + y)
 mx <- lm(z ~ x)
 my <- lm(z ~ y)
 signifmin <- ifelse(min(summary(m)$coef[2:3,4]) < 0.05, 1, 0)
 signifx <- ifelse(min(summary(mx)$coef[2,4]) < 0.05, 1, 0)
 signify <- ifelse(min(summary(my)$coef[2,4]) < 0.05, 1, 0)
 signifxory <- ifelse(signifx == 1 | signify == 1, 1, 0)
 return(c(cor(x,y),signifmin,signifx,signify,signifxory))
}
result <- sapply(rep(n,M), f)
sum(result[2,])/M # joined model returns >=1 spurious effect
sum(result[3,])/M
sum(result[4,])/M
sum(result[5,])/M # two individual models return >=1 spurious effect
min(result[1,])
```

So what does collinearity do?

- Type II error increases → power loss
- Type I error does not increase (much)

★ But small differences between highly correlated predictors can be highly correlated with another predictors and create 'apparent effects' (like in the case discussed).

→ Can lead to *misleading* effects (not technically spurious, but if they we interpret the coefficients *causally* we will have a misleading result!).

  - This problem is not particular to collinearity, but it frequently occurs in the case of collinearity.

- When coefficients are unstable (as in the above case of collinearity) treat this as a warning sign - check for **mediated effects**.

Detecting collinearity

- Mixed model output in R comes with correlation matrix (cf. previous slide).
  - Partial correlations of fixed effects *in the model*.
- Also useful: correlation matrix (e.g. `cor()`; use Spearman option for categorical predictors) or `pairscor.fnc()` in `languageR` for visualization.
  - apply to predictors (not to untransformed input variables)!

```
> cor(lexdec[,c(2,3,10, 13)])
                  RT        Trial    Frequency       Length
RT         1.0000000 -0.052411295 -0.213249525  0.146738111
Trial     -0.0524113  1.000000000 -0.006849117  0.009865814
Frequency -0.2132495 -0.006849117  1.000000000 -0.427338136
Length     0.1467381  0.009865814 -0.427338136  1.000000000
```

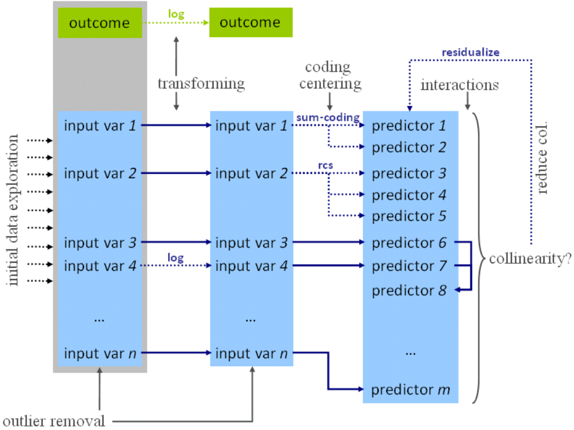
Formal tests of collinearity

- Variance inflation factor (VIF, `vif()`).
  - generally, VIF $> 10 \rightarrow$ absence of absolute collinearity in the model cannot be claimed.
  - ★ VIF > 4 are usually already problematic.
  - ★ but, for large data sets, even VIFs > 2 can lead inflated standard errors.
- Kappa (e.g. `collin.fnc()` in `languageR`)
  - generally, c-number (κ) over 10 $\rightarrow$ mild collinearity in the model.
- Applied to current data set, ...

```
> collin.fnc(lexdec[,c(2,3,10,13)])$cnumber
```

- ...gives us a kappa $> 90 \rightarrow$ Houston, we have a problem.

Dealing with collinearity



Dealing with collinearity

- Good news: Estimates are only problematic for those predictors that are collinear.

→ If collinearity is in the nuisance predictors (e.g. certain controls), nothing needs to be done.

- Somewhat good news: If collinear predictors are of interest but we are *not* interested in the direction of the effect, we can use **model comparison** (rather than tests based on the standard error estimates of coefficients).

- If collinear predictors are of interest and we *are* interested in the direction of the effect, we need to reduce collinearity of those predictors.

Reducing collinearity

- **Centering**: reduces collinearity of predictor with intercept and higher level terms involving the predictor.
  - **pros:** easy to do and interpret; often improves interpretability of effects.
  - **cons:** none?
- **Re-express the variable** based on conceptual considerations (e.g. ratio of spoken vs. written frequency in `lexdec`; rate of disfluencies per words when constituent length and fluency should be controlled).
  - **pros:** easy to do and relatively easy to interpret.
  - **cons:** only applicable in some cases.

Reducing collinearity (cnt'd)

- **Stratification**: Fit separate models on **subsets** of data holding correlated predictor A constant.
- If effect of predictor B persists → effect is probably real.
  - **pros:** Still relatively easy to do and easy to interpret.
  - **cons:** harder to do for continuous collinear predictors; reduces power, → extra caution with null effects; doesn't work for multicollinearity of several predictors.
- **Principal Component Analysis (PCA)**: for n collinear predictors, extract $k < n$ most important orthogonal components that capture $> p\%$ of the variance of these predictors.
  - **pros:** Powerful way to deal with *multi*collinearity.
  - **cons:** Hard to interpret (→ better suited for control predictors that are not of primary interest); technically complicated; some decisions involved that affect outcome.

Reduce collinearity (cnt'd)

- **Residualization**: Regress collinear predictor against combination of (partially) correlated predictors
  - usually using ordinary regression (e.g. `lm()`, `ols()`).
  - **pros:** systematic way of dealing with multicollinearity; directionality of (conditional) effect interpretable
  - **cons:** effect sizes hard to interpret; judgment calls: what should be residualized against what?

An example of moderate collinearity (cnt'd)

- Consider two moderately correlated variables ($r = -0.49$), (centered) word length and (centered log) frequency:

```
> lmer(RT ~ cLength + cFrequency +
+           (1 | Word) + (1 | Subject), data = lexdec)
<...>
Fixed effects:
             Estimate Std. Error t value
(Intercept)  6.385090   0.034415  185.53
cLength      0.009348   0.004327    2.16
cFrequency  -0.037028   0.006303   -5.87

Correlation of Fixed Effects:
           (Intr) cLngth
cLength    0.000
cFrequency 0.000  0.429
<...>
```

- Is this problematic? Let's remove collinearity via **residualization**

Residualization: An example

- Let's regress word length vs. word frequency.

```
> lexdec$rLength = residuals(lm(Length ~ Frequency, data = lexdec))
```

- `rLength`: difference between actual length and length as predicted by frequency. Related to actual length ($r > 0.9$), but crucially not to frequency ($r \ll 0.01$).
- Indeed, collinearity is removed from the model:

```
<...>
Fixed effects:
             Estimate Std. Error t value
(Intercept)  6.385090   0.034415  185.53
rLength      0.009348   0.004327    2.16
cFrequency  -0.042872   0.005693   -7.53

Correlation of Fixed Effects:
           (Intr) rLngth
rLength    0.000
cFrequency 0.000  0.000
<...>
```

$\longrightarrow$ $\mathrm{SE}(\beta)$ estimate for frequency predictor decreased

$\longrightarrow$ larger t-value

Residualization: An example (cnt'd)

- **Q:** What precisely is `rLength`?
- **A:** Portion of word length that is not explained by (a linear relation to log) word frequency.

$\longrightarrow$ Coefficient of `rLength` needs to be interpreted as such

- No trivial way of back-transforming to `Length`.

- NB: We have granted frequency the entire portion of the variance that cannot unambiguously attributed to *either* frequency *or* length!

$\longrightarrow$ If we choose to residualize frequency on length (rather than the inverse), we may see a different result.

Understanding residualization

- So, let's regress frequency against length.
- Here: no qualitative change, but word length is now *highly* significant (random effect estimates unchanged)

```
> lmer(RT ~ cLength + rFrequency +
+          (1 | Word) + (1 | Subject), data = lexdec)
<...>
Fixed effects:
             Estimate Std. Error t value
(Intercept)  6.385090   0.034415  185.53
cLength      0.020255   0.003908    5.18
rFrequency  -0.037028   0.006303   -5.87

Correlation of Fixed Effects:
           (Intr) cLngth
cLength    0.000
rFrequency 0.000  0.000
<...>
```

$\longrightarrow$ Choosing what to residualize, changes interpretation of βs and hence the hypothesis we're testing.

Extreme collinearity: ctn'd

- we can now residualize `meanWeight` against `meanSize` and `Frequency`, and
- and residualize `meanSize` against `Frequency`.
- include the transformed predictors in the model.

```
> lexdec$rmeanSize <- residuals(lm(cmeanSize ~ Frequency + cmeanWeight,
+                                  data=lexdec))
> lexdec$rmeanWeight <- residuals(lm(cmeanWeight ~ Frequency,
+                                    data=lexdec))
> lmer(RT ~ rmeanSize + rmeanWeight + Frequency + (1|Subject) + (1|Word),
+      data=lexdec)
(Intercept)   6.588778   0.043077  152.95
rmeanSize    -0.118731   0.351957   -0.34
rmeanWeight   0.026198   0.007477    3.50
Frequency    -0.042872   0.005470   -7.84
```

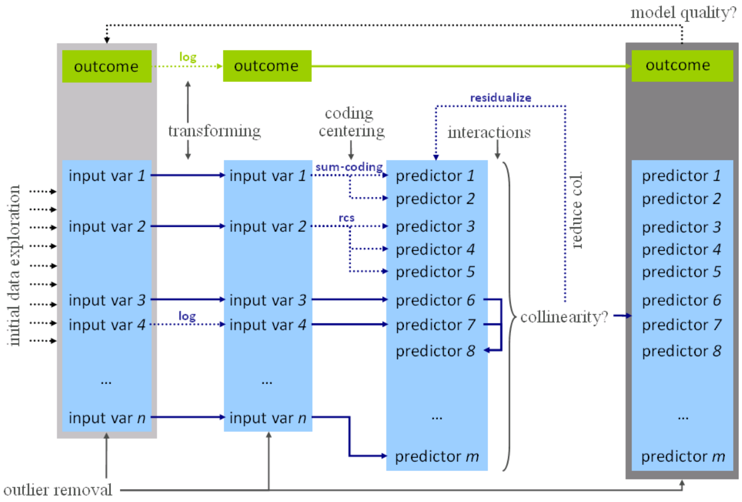
- NB: The frequency effect is stable, but the `meanSize` vs. `meanWeight` effect depends on what is residualized against what.

Residualization: Which predictor to residualize?

- What to residualize should be based on conceptual considerations (e.g. rate of disfluencies = number of disfluencies $\sim$ number of words).
- **Be conservative** with regard to your hypothesis:
  - If the effect only holds under some choices about residualization, *the result is inconclusive*.
  - We usually want to show that a hypothesized effect holds *beyond what is already known* or that it *subsumes other effects*.

  $\longrightarrow$ **Residualize** effect of interest.
    - E.g. if we hypothesize that a word's predictability affects its duration beyond its frequency $\longrightarrow$ `residuals(lm(Predictability ~ Frequency, data))`.
  - (if effect *direction* is not important, see also **model comparison**)

Modeling schema



Overfitting

Overfitting: Fit might be too tight due to the exceeding number of parameters (coefficients). The maximal number of predictors that a model allows depends on their distribution and the distribution of the outcome.

- **Rules of thumb**:
  - **linear models**: >20 observations per predictor.
  - **logit models**: the less frequent outcome should be observed >10 times more often than there predictors in the model.
  - Predictors count: one per each random effect + residual, one per each fixed effect predictor + intercept, one per each interaction.

Validation

Validation allows us to detect **overfitting**:

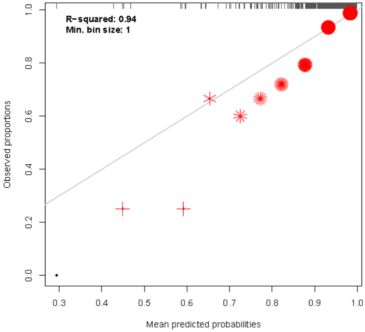
- How much does our model depend on the exact data we have observed?
- Would we arrive at the same conclusion (model) if we had only slightly different data, e.g. a subset of our data?

- **Bootstrap-validate** your model by repeatedly sampling from the population of speakers/items with replacement. Get estimates and confidence intervals for fixed effect coefficients to see how well they generalize (Baayen, 2008:283; cf. `bootcov()` for ordinary regression models).

Visualize validation

- Plot predicted vs. observed (averaged) outcome.
- E.g. for logit models, `plot.logistic.fit.fnc` in `languageR` or similar function (cf. *http://hlplab.wordpress.com*)
  - The following shows a badly fitted model:

```
> lexdec$NativeEnglish = ifelse(lexdec$NativeLanguage == "English", 1, 0)
> lexdec$cFrequency = lexdec$Frequency - mean(lexdec$Frequency)
> lexdec$cNativeEnglish = lexdec$NativeEnglish - mean(lexdec$NativeEnglish)
> lexdec$Correct = ifelse(lexdec$Correct == "correct", T, F)
> l <- glmer(Correct ~ cNativeEnglish * cFrequency + Trial +
+                       (1 | Word) + (1 | Subject),
+                       data = lexdec, family="binomial")
```



Fitted values

So far, we've been worrying about coefficients, but the real model output are the **fitted values**.

Goodness-of-fit measures assess the relation between fitted (a.k.a. predicted) values and actually observed outcomes.

- **linear models**: Fitted values are predicted numerical outcomes.

```
        RT   fitted
1 6.340359 6.277565
2 6.308098 6.319641
3 6.349139 6.265861
4 6.186209 6.264447
```

- **logit models**: Fitted values are predicted log-odds (and hence predicted probabilities) of outcome.

```
  Correct    fitted
1 correct 0.9933675
2 correct 0.9926289
3 correct 0.9937420
4 correct 0.9929909
```

Goodness-of-fit and data likelihood

- **Data likelihood**: What is the probability that we would observe the data we have given the model (i.e. given the predictors we chose and given the 'best' parameter estimates for those predictors).
- Standard model output usually includes such measures, e.g. in R:

```
   AIC    BIC logLik deviance REMLdev
-96.48 -63.41  55.24   -123.5  -110.5
```

- **log-likelihood**, logLik $= log(L)$. This is the maximized model's log data likelihood, no correction for the number of parameters. **Larger (i.e. closer to zero) is better**. The value for log-likelihood should always be *negative*, and AIC, BIC etc. are positive. → current bug in the `lmer()` output for linear models.

Measures built on data likelihood (contd')

- Other measures trade off goodness-of-fit (data likelihood) and model complexity (number of parameters; cf. Occam's razor; see also model comparison).
  - Deviance: -2 times log-likelihood ratio. **Smaller is better.**
  - Aikaike Information Criterion, $\text{AIC} = k - 2ln(L)$, where k is the number of parameters in the model. **Smaller is better.**
  - Bayesian Information Criterion, $\text{BIC} = k * ln(n) - 2ln(L)$, where k is the number of parameters in the model, and n is the number of observations. **Smaller is better.**

Goodness-of-fit: Mixed Logit Models

```
   AIC BIC logLik deviance
499.1 537 -242.6    485.1
```

- ★ but no known closed form solution to likelihood function of mixed logit models → current implementations use **Penalized Quasi-Likelihoods** or better **Laplace Approximation** of the likelihood (default in R; cf. Harding & Hausman, 2007)
- Discouraged:
  - ★ **pseudo-R^2** a la Nagelkerke (cf. along the lines of *http://www.ats.ucla.edu/stat/mult_pkg/faq/general/Psuedo_RSquareds.htm*)
  - ★ **classification accuracy**: If the predicted probability is < 0.5 → predicted outcome $= 0$; otherwise 1. Needs to be compared against baseline. (cf. Somer's D_{xy} and C index of concordance).

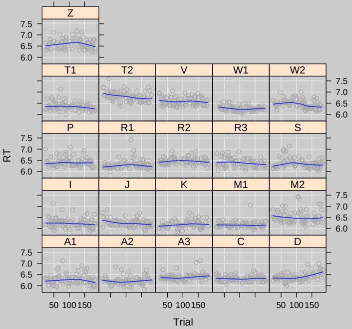
Model comparison

- Models can be compared for performance using any goodness-of-fit measures. Generally, an advantage in one measure comes with advantages in others, as well.
- **To test whether one model is significantly better** than another model:
  - **likelihood ratio test** (for nested models only)
  - (DIC-based tests for non-nested models have also been proposed).

Likelihood ratio test for nested models

- -2 times ratio of likelihoods (or difference of log likelihoods) of nested model and super model.
- Distribution of likelihood ratio statistic follows asymptotically the χ-square distribution with $DF(model_{super}) - DF(model_{nested})$ degrees of freedom.
- χ-square test indicates whether sparing extra df's is justified by the change in the log-likelihood.
  - in R: `anova(model1, model2)`
  - NB: use **restricted maximum likelihood-fitted** models to compare models that differ in random effects.

Example of model comparison




```
> super.lmer = lmer(RT ~ rawFrequency + (1 | Subject) + (1 | Word), data = lexdec)
> nested.lmer = lmer(RT ~ rawFrequency + (1 + Trial| Subject) + (1 | Word), data = lexdec)
> anova(super.lmer, nested.lmer)

            Df     AIC     BIC  logLik  Chisq Chi Df Pr(>Chisq)
super.lmer   5 -910.41 -883.34  460.20
nested.lmer  7 -940.71 -902.81  477.35 34.302      2   3.56e-08 ***
```

$\longrightarrow$ change in log-likelihood justifies inclusion Subject-specific **slopes** for Trial, and the **correlation parameter** between trial intercept and slope.

Model comparison: Trade-offs

- Compared to tests based on $\mathrm{SE}(\beta)$, model comparison ...
  - robust against collinearity
  - does not test directionality of effect

★ **Suggestion:** In cases of high collinearity ...
  - first determine which predictors are subsumed by others (**model comparison**, e.g. $p > 0.7$)) $\longrightarrow$ remove them,
  - then use $\mathrm{SE}(\beta)$-based tests (**model output**) to test effect *direction* on simple model (with reduced collinearity).

Reporting the model's performance

- for the overall performance of the model, report goodness-of-fit measures:
  - D_{xy} or concordance C-number. Report the increase in classification accuracy over and beyond the baseline model.
- for model comparison: report the p-value of the log-likelihood ratio test.

Before you report the model coefficients

- **Transformations**, **centering**, (potentially ↷**standardizing**), **coding**, **residualization** should be described as part of the predictor summary.
  - Where possible, give theoretical, and/or empirical arguments for any decision made.
  - Consider reporting scales for outputs, inputs and predictors (e.g., range, mean, sd, median).

Some considerations for good science

- **Do not** report effects that heavily depend on the choices you have made;
- **Do not** fish for effects. There should be a strong theoretical motivation for what variables to include and in what way.
- To the extent that different ways of entering a predictor are investigated (without a theoretical reason), **do** make sure your conclusions hold for *all* ways of entering the predictor *or* that the model you choose to report is superior (**model comparison**).

What to report about effects

- ↷**Effect size** (What is that actually?)
- Effect direction
- Effect shape (tested by significance of non-linear components & superiority of transformed over un-transformed variants of the same input variable); plus visualization

Interpretation of coefficients

```
Fixed effects:
                       Estimate Std. Error t value
(Intercept)            6.323783   0.037419  169.00
NativeLanguageOther    0.150114   0.056471    2.66
cFrequency            -0.039377   0.005552   -7.09
```

- The increase in 1 log unit of `cFrequency` comes with a -0.039 log units decrease in log-odds.
- Utterly uninterpretable!
- To get estimates in sensible units we need to back-transform both our predictors and our outcomes.
  - decentralize `cFrequency`, and
  - exponentially-transform logged `Frequency` and `RT`.
  - if necessary, we de-residualize and de-standardize predictors and outcomes.

Getting interpretable effects

- estimate the effect in ms across the frequency range (or better from 5th to 95th percentile) and then the effect for a unit of frequency.

```
> intercept = as.vector(fixef(lexdec.lmer4)[1])
> betafreq = as.vector(fixef(lexdec.lmer4)[3])
> eff = exp(intercept + betafreq * max(lexdec$Frequency)) -
> exp(intercept + betafreq * min(lexdec$Frequency)))
[1] -109.0357 #RT decrease across the entire range of Frequency
> range = exp(max(lexdec$Frequency)) -
> exp(min(lexdec$Frequency))
[1] 2366.999
```

- Report that the full effect of `Frequency` on `RT` is a 109 ms decrease.

★ But here there is no simple relation between RTs and frequency, so resist reporting "the difference in 100 occurrences comes with a 4 ms decrease of `RT`".

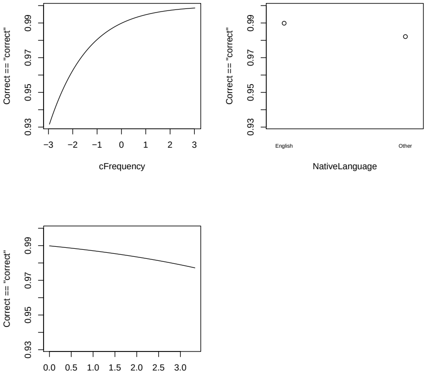
```
> eff/range * 100
[1] -4.606494
```

Comparing effect sizes

- It ain't trivial: What is meant by effect size?
  - Change of outcome if 'feature' is present? → coefficient
    - per unit?
    - overall range?
  - But that does not capture how much an effect affects language processing:
    - What if the feature is rare in *real language use* ('availability of feature')? Could use ...
    - → Variance accounted for (**goodness-of-fit** improvement associated with factor)
    - → **Standardized coefficient** (gives direction of effect)
- ★ **Standardization**: subtract the mean and divide by two standard deviations.
  - standardized predictors are on the same scale as binary factors (cf. Gelman & Hill 2006).
  - makes all predictors (relatively) comparable.

Plotting coefficients of mixed logit models

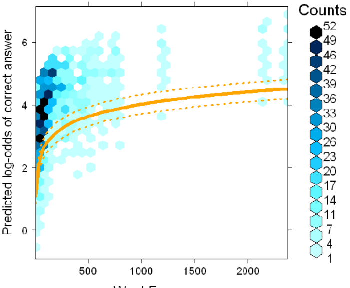
- Log-odd units can be automatically transformed to probabilities.
  - **pros:** more familiar space
  - **cons:** effects are linear in log-odds space, but non-linear in probability space; linear slopes are hard to compare in probability space; non-linearities in log-odd space are hard to interpret

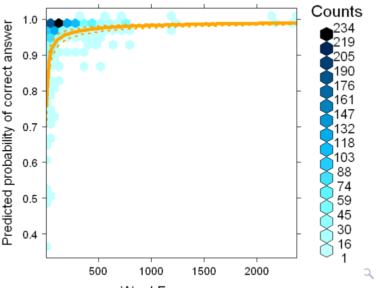


Plotting coefficients of mixed logit models (contd')

- For an alternative way, see *http://hlplab.wordpress.com/*:

```
> data(lexdec)
> lexdec$NativeEnglish = ifelse(lexdec$NativeLanguage == "English", 1, 0)
> lexdec$rawFrequency = exp(lexdec$Frequency)
> lexdec$cFrequency = lexdec$Frequency - mean(lexdec$Frequency)
> lexdec$cNativeEnglish = lexdec$NativeEnglish - mean(lexdec$NativeEnglish)
> lexdec$Correct = ifelse(lexdec$Correct == "correct", T, F)
> l<- lmer(Correct ~ cNativeEnglish + cFrequency + Trial +
+                    (1 | Word) + (1 | Subject), data = lexdec, family="binomial")
> my.glmerplot(l, "cFrequency", predictor= lexdec$rawFrequency,
+                  predictor.centered=T, predictor.transform=log,
+                  name.outcome="correct answer", xlab= ex, fun=plogis)
```





Plotting coefficients of mixed logit models (contd')

- Great for outlier detection. Plot of predictor in log-odds space (actual space in which model is fit):