university of groningen

NWO
Netherlands Organisation for Scientific Research

Analyzing EEG data using GAMs

Lecture 4 of advanced regression for linguists

Martijn Wieling

Department of Humanities Computing, University of Groningen

Cambridge, January 14, 2015

This lecture

- Introduction
  - ERPs to study gender violations
  - Research question
- Design
- Methods: `R` code
- Results
- Discussion

ERPs to study grammatical gender violations

- A P600 (a positivity 'around' 600 ms. after stimulus onset) is sensitive to grammatical violations
- An N400 (a negativity 'around' 400 ms. after stimulus onset) is modulated by semantic context and lexical properties of a word
- The P600/N400 are found by comparing the incorrect sentences to the correct sentences

- Native speakers appear to show a P600 for gender violations
  - But analyzed by averaging over items and over subjects!
- Here we are interested in how non-native speakers respond to gender violations (joint work with Nienke Meulman)

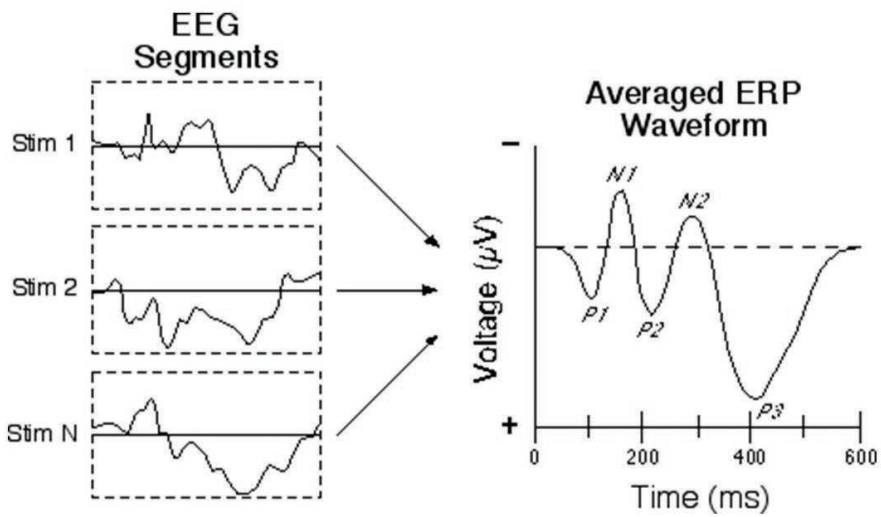
- Gender is very hard to learn for L2 learners
- Even though behaviorally L2 learners might show correct responses, the brain may reveal differences in processing gender

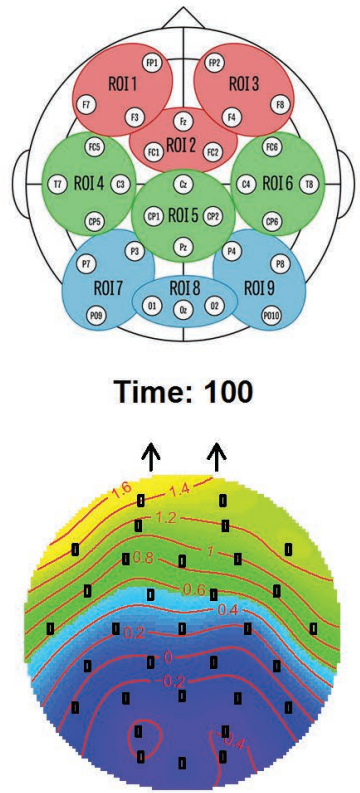
Research question

- Is the P600 for gender violations dependent on age of arrival for the L2 learners of German?

ERP data

- Today: analysis of single region of interest (ROI 8)
  - GAMs allow for spatial distribution analyses





Design

- 67 L2 speakers of German (Slavic L1)
- Auditory presentation of correct sentences or sentences with a gender violation (incorrect determiner; no determiners in L1)
- 48 items in each condition: 96 trials per participant (minus artifacts)
- Example:

  *Nach der Schlägerei ist das/*der Auge des Angestellten von der Krankenschwester versorgt worden.*

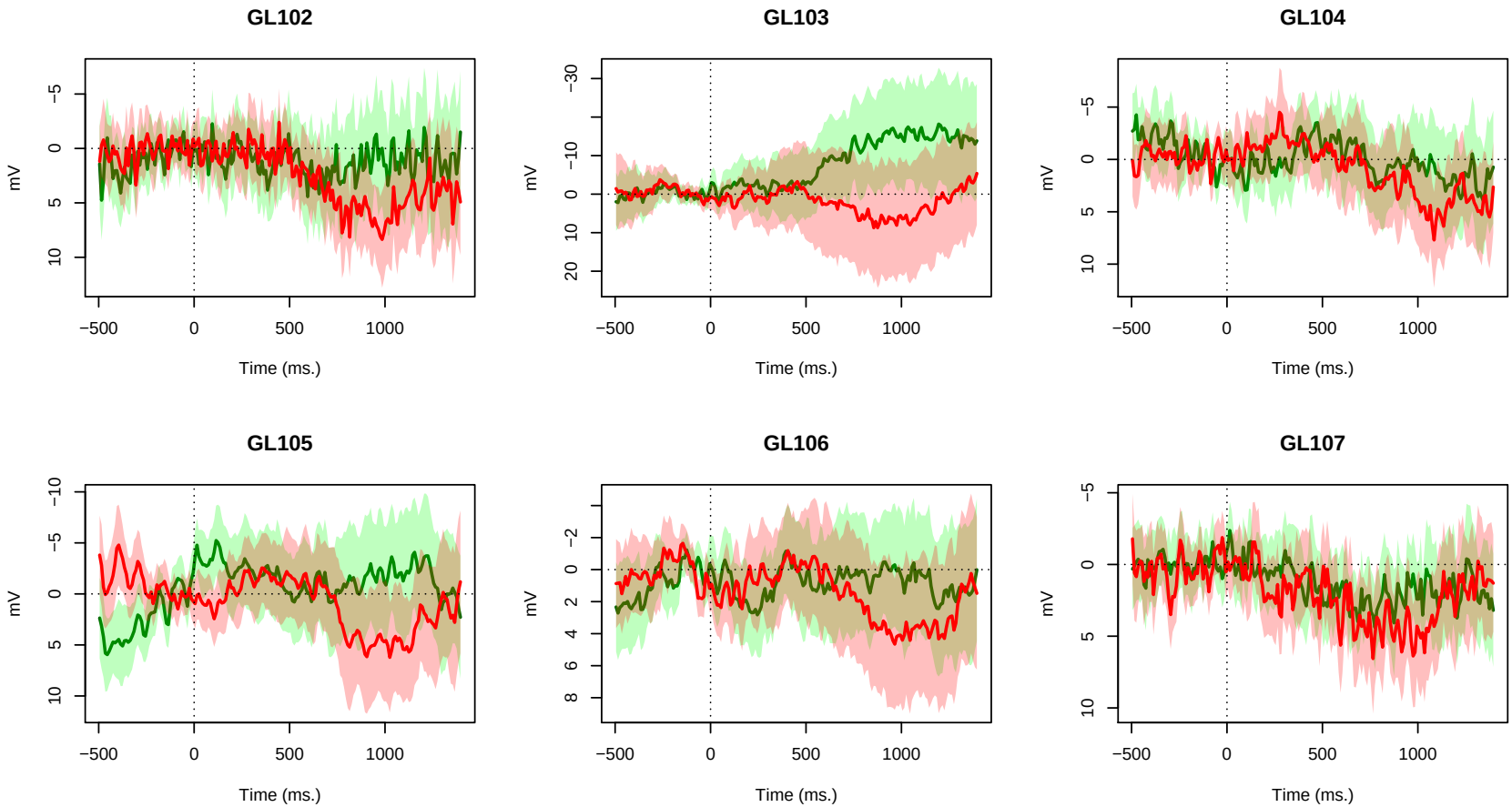
  After the fight the$_{\text{neut}}$/*the$_{\text{masc}}$ eye of the worker was treated by the nurse.

Data overview

```
> load('dat.rda')

# data needs to be sorted to test for autocorrelation problems
> dat = dat[order(dat$Subject, dat$TrialNr, dat$Time, decreasing=F),]

> head(dat)
            uV Time Subject    Group Word TrialNr      Roi
44947  8.94400  505   GL102 GenEarly Wald       2 post.mid
28121 15.56267  515   GL102 GenEarly Wald       2 post.mid
40909 21.30807  525   GL102 GenEarly Wald       2 post.mid
1294  13.31573  535   GL102 GenEarly Wald       2 post.mid
9446  19.10700  545   GL102 GenEarly Wald       2 post.mid
81324 17.95607  555   GL102 GenEarly Wald       2 post.mid
      Structure Correctness L1 AoArr LoR Age Edu SeqStart
44947        DN       incor PL     8  24  32   3     TRUE
28121        DN       incor PL     8  24  32   3    FALSE
40909        DN       incor PL     8  24  32   3    FALSE
1294         DN       incor PL     8  24  32   3    FALSE
9446         DN       incor PL     8  24  32   3    FALSE
81324        DN       incor PL     8  24  32   3    FALSE

> dim(dat)
[1] 442160     15
```

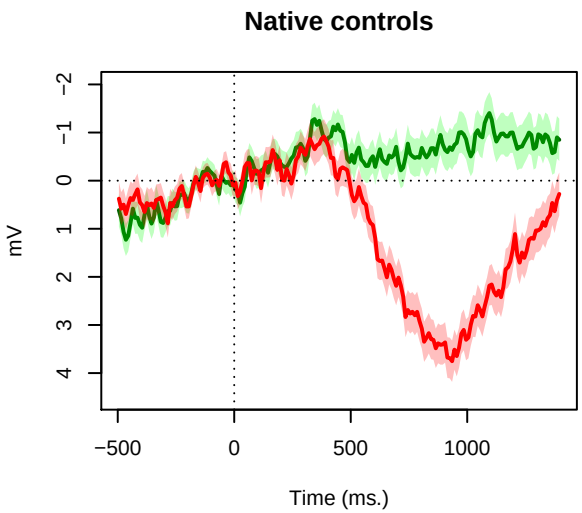
Much individual variation

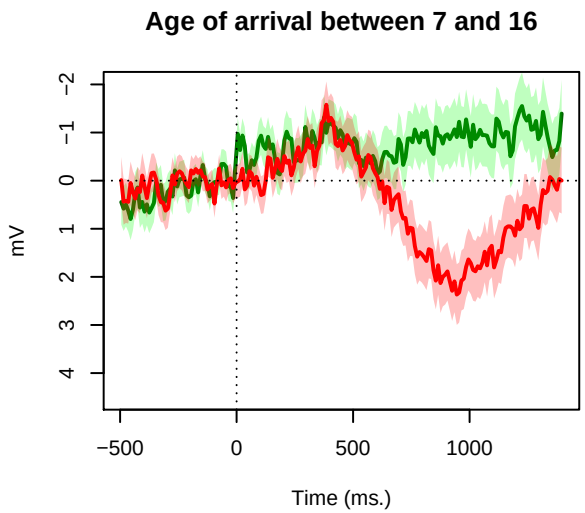
(the signal has been downsampled to 100 Hz)

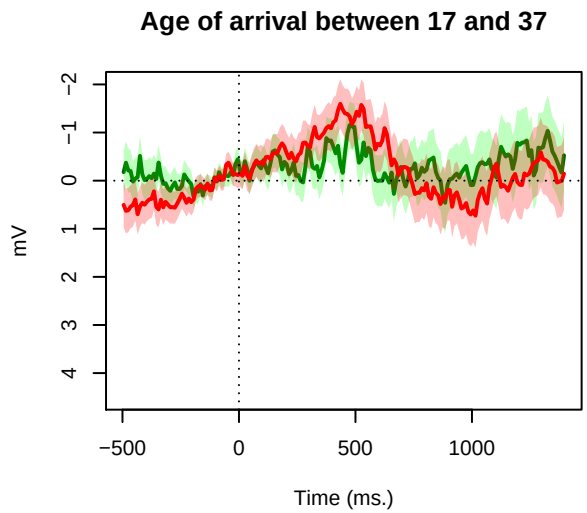


General patterns exist

(note the arbitrary age split, however)







A first model: a general effect of time

(we focus here on the time span from 500 to 1300 ms)

```
> library(mgcv) # 1.8.3
> m0 = bam(uV ~ s(Time), data=dat, gc.level=2, method='ML')
> summary(m0)

Family: gaussian
Link function: identity

Formula:
uV ~ s(Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.06120    0.02774  -2.206   0.0274 *

Approximate significance of smooth terms:
          edf Ref.df     F p-value
s(Time) 3.917   4.84 25.95  <2e-16 ***

R-sq.(adj) =  0.000284   Deviance explained = 0.0293%
ML score = 1.9162e+06  Scale est. = 340.13    n = 442160
```

- Why was maximum-likelihood (ML) estimation used for fitting?

A first model: a general effect of time

(we focus here on the time span from 500 to 1300 ms)

```
> library(mgcv) # 1.8.3
> m0 = bam(uV ~ s(Time), data=dat, gc.level=2, method='ML')
> summary(m0)

Family: gaussian
Link function: identity

Formula:
uV ~ s(Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.06120    0.02774  -2.206   0.0274 *

Approximate significance of smooth terms:
           edf Ref.df     F p-value
s(Time) 3.917   4.84 25.95  <2e-16 ***

R-sq.(adj) =  0.000284   Deviance explained = 0.0293%
ML score = 1.9162e+06  Scale est. = 340.13    n = 442160
```

- Why was maximum-likelihood (ML) estimation used for fitting?

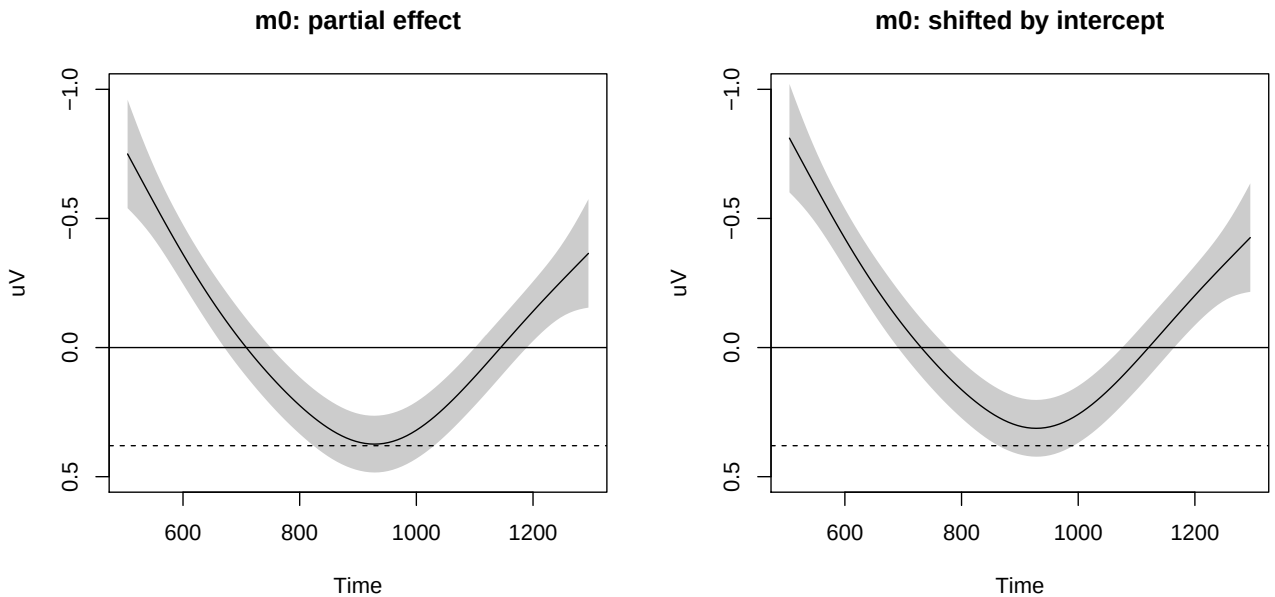
Fitting methods for GAMs

- For model comparison, the approach is similar as for `lmer` models:
  - For comparing differences in the fixed effects (constant random effects) of two models, fit both models using `method="ML"` (`lmer`: `REML=F`)
  - For comparing random effects (constant fixed effects) between two models, fit using `method="fREML"` (`bam` only) or `method="REML"` (`lmer`: `REML=T`)
- Choosing the fitting method of your final model (using `method="..."`):
  - `ML`: conservative, but biased variance components (oversmoothing)
  - `fREML/REML`: better smooths, but less conservative *p*-values
  - `GCV.Cp`: good for prediction, but not robust to autocorrelation problems
- I generally use `method="ML"` for my final model (conservative)
  - (but note that fitting a model with non-problematic residuals is currently only possible by using `method="fREML"`)

Visualizing the non-linear time pattern of the model

(Interpreting GAM results **always** involves visualization)

```
> par(mfrow=c(1,2)) # 2 plots in one window
> plot(m0, rug=F, shade=T, ylim=c(0.5,-1), seWithMean=T,
       ylab='uV', main='m0: partial effect')
> abline(h = c(0,0.38), lty = c(1,2)) # horizontal lines
> ic = m0$coef["(Intercept)"] # intercept (-0.06) from model
> plot(m0, rug=F, shade=T, ylim=c(0.5-ic,-1-ic), seWithMean=T, shift = ic,
       ylab='uV', main='m0: shifted by intercept')
> abline(h = c(0,0.38), lty = c(1,2)) # horizontal lines
```



Check for additional wigglyness

(if p-value is low and edf close to k'; note that the default *k* is arbitrary and higher for `s` than for `te`)

```
> gam.check(m0)

Method: ML   Optimizer: outer newton
full convergence after 2 iterations.
Gradient range [-0.6306798,0.1689593]
(score 1916153 & scale 340.1335).
Hessian positive definite, eigenvalue range [1.948326,221080.6].
Model rank =  10 / 10

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

           k'   edf k-index p-value
s(Time) 9.000 3.917   0.997    0.46
```

Increasing the wigglyness of a smooth with *k*

(double *k* if higher *k* is needed, but do not set it too high, i.e. max $\frac{1}{2}$ × unique time points)

```
> m0b = bam(uV ~ s(Time, k=20), data=dat, gc.level=2, method='ML')
> summary(m0b)

Family: gaussian
Link function: identity

Formula:
uV ~ s(Time, k = 20)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.06119    0.02774  -2.205   0.0274 *

Approximate significance of smooth terms:
          edf Ref.df     F p-value
s(Time) 4.038   5.04 24.94  <2e-16 *** # was 3.917

R-sq.(adj) =  0.000284   Deviance explained = 0.0293%
ML score = 1.9162e+06  Scale est. = 340.13     n = 442160
```

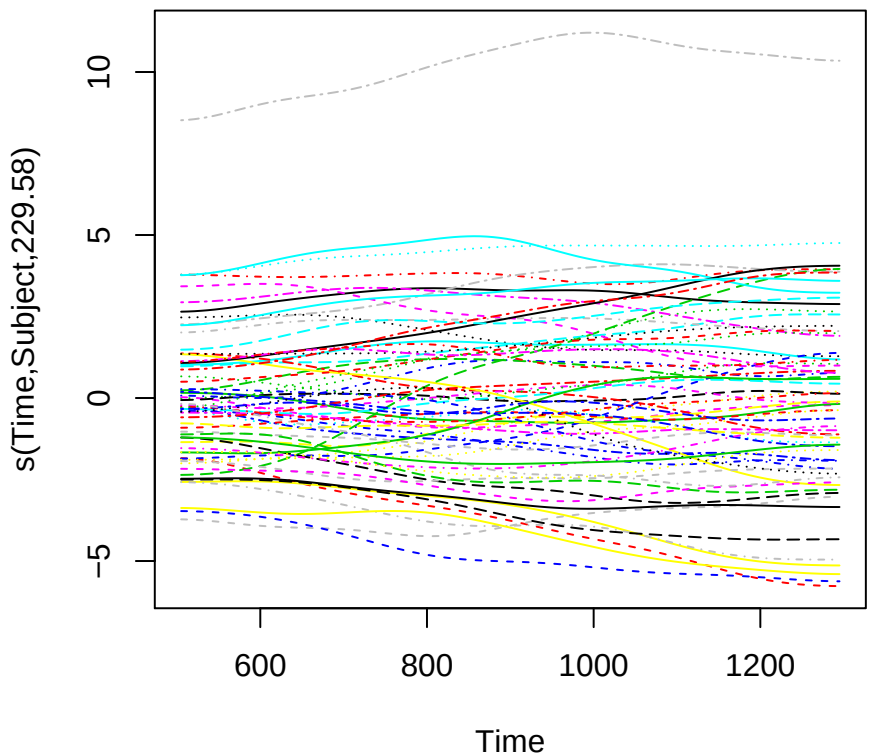
Taking into account individual variation

- The effect of time is non-linear and is variable per subject
- We need a random intercept/slope which is non-linear
  - (instead of random intercept and random slope of time per subject)

```
> m1 = bam(uV ~ s(Time) + s(Time,Subject,bs='fs',m=1),
           data=dat, gc.level=2, method='ML')
> summary(m1)
...
Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.08612    0.29531  -0.292    0.771

Approximate significance of smooth terms:
                    edf  Ref.df     F  p-value
s(Time)           3.741   4.482 11.49 6.09e-10 ***
s(Time,Subject) 229.578 602.000 14.48  < 2e-16 ***

R-sq.(adj) =  0.0196   Deviance explained =  2.01% # was 0.0293%
-ML = 1.9121e+06  Scale est. = 333.56    n = 442160
```

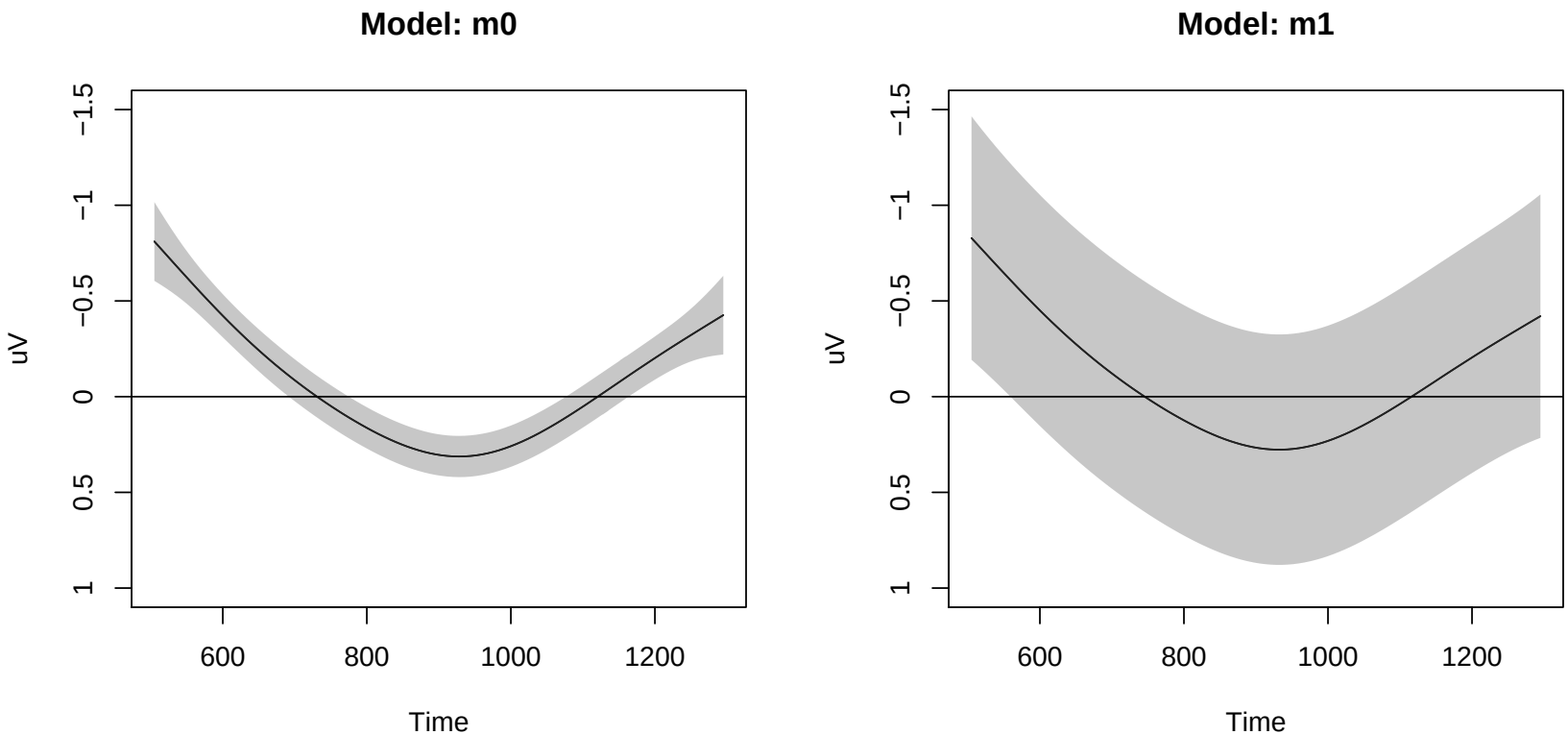
Visualization of individual variation

```
> plot(m1, select=2)
```



Smooths become more uncertain

```
> source('plotting.R') # custom plotting functions
> par(mfrow=c(1,2))
> plotSmooths(m0, "Time", ylim=c(1,-1.5)) # using complete model context
> plotSmooths(m1, "Time", ylim=c(1,-1.5), dropRanef="Subject")
```



Assessing correct versus incorrect

(smooths are centered, so the factorial predictor also needs to be included in the fixed effects)

```
> m2 = bam(uV ~ s(Time,by=Correctness) + Correctness +
                s(Time,Subject,bs='fs',m=1),
           data=dat, gc.level=2, method='ML')
> summary(m2)

...

Parametric coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.55608    0.29553  -1.882   0.0599 .
Correctnessincor   0.88609    0.05493  16.132   <2e-16 ***

Approximate significance of smooth terms:
                               edf  Ref.df      F p-value
s(Time):Correctnesscor       1.295   1.458  2.901  0.0704 .
s(Time):Correctnessincor     4.220   5.145 26.494  <2e-16 ***
s(Time,Subject)            229.770 602.000 14.490  <2e-16 ****

R-sq.(adj) =  0.0205   Deviance explained =  2.1%
-ML = 1.9119e+06  Scale est. = 333.26    n = 442160
```

- Does the correct-incorrect distinction improve the model?

Assessing correct versus incorrect

(smooths are centered, so the factorial predictor also needs to be included in the fixed effects)

```
> m2 = bam(uV ~ s(Time,by=Correctness) + Correctness +
                s(Time,Subject,bs='fs',m=1),
           data=dat, gc.level=2, method='ML')
> summary(m2)

...

Parametric coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.55608    0.29553  -1.882   0.0599 .
Correctnessincor   0.88609    0.05493  16.132   <2e-16 ***

Approximate significance of smooth terms:
                               edf  Ref.df      F p-value
s(Time):Correctnesscor       1.295   1.458  2.901  0.0704 .
s(Time):Correctnessincor     4.220   5.145 26.494  <2e-16 ***
s(Time,Subject)            229.770 602.000 14.490  <2e-16 ****

R-sq.(adj) =  0.0205   Deviance explained =  2.1%
-ML = 1.9119e+06  Scale est. = 333.26    n = 442160
```

- Does the correct-incorrect distinction improve the model?

Model comparison and GAMs

- There are three possibilities for comparing GAM models:
  - Compare (fRE)ML values: `compareML(m0,m1)` (custom function)
  - Compare AIC values (at least reduction of 2): `compareML(m0,m1)` or `AIC(m0) - AIC(m1)`
  - Look at the *p*-values of the model summary (recommended)
- Important: not all methods can be applied in all situations!
- The table below shows when each method can be applied:

(N.B. rho is used to correct for autocorrelation in the residuals and explained later)

	Gaussian without rho	Gaussian with rho	Non-Gaussian
`gam`	(RE)ML / AIC / summary	-	(RE)ML / AIC / summary
`bam`	(fRE)ML / AIC / summary	(fRE)ML / summary	AIC / summary

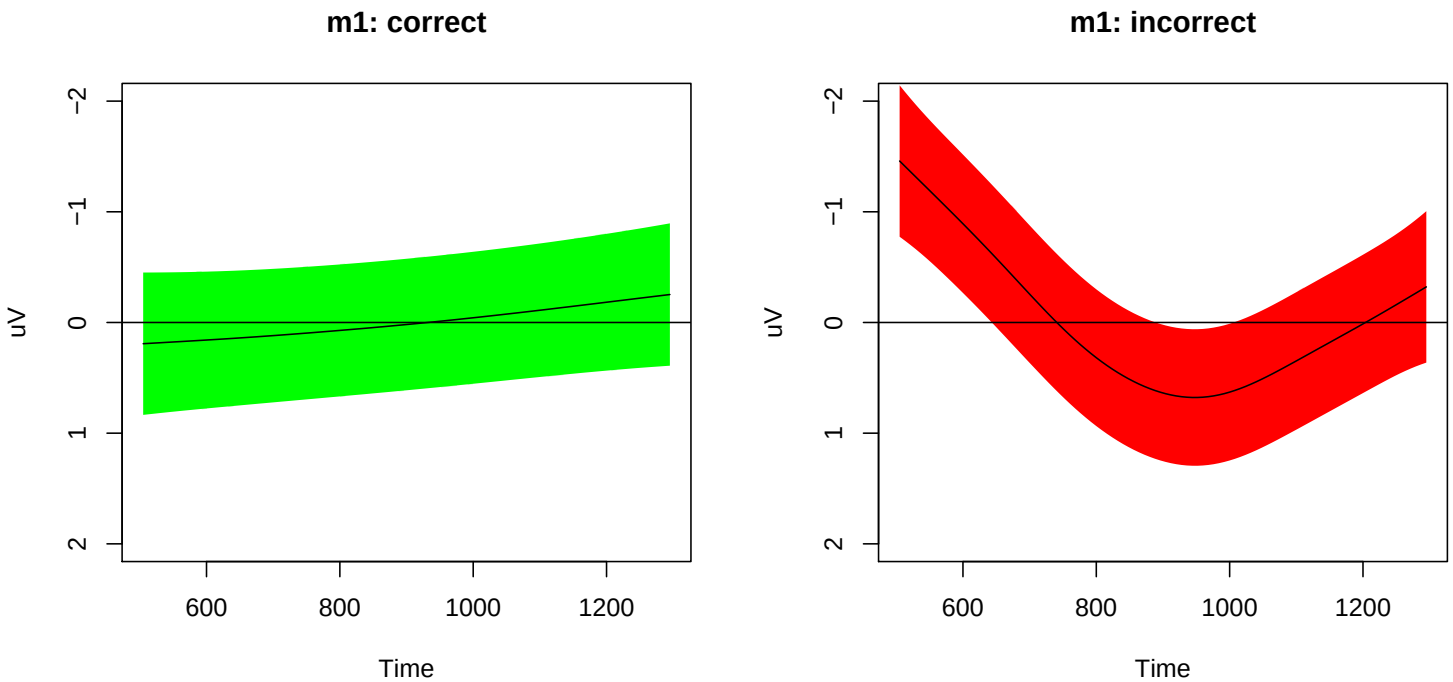
Assessing model improvement

```
> source('compareML.R')  # custom model comparison function
> compareML(m1,m2)

m1: uV ~ s(Time) + s(Time, Subject, bs = "fs", m = 1)

m2: uV ~ s(Time, by = Correctness) + Correctness +
         s(Time, Subject, bs = "fs", m = 1)

Chi-square test of ML scores
-----
  Model   Score Edf   Chisq    Df  p.value Sig.
1    m1 1912123   5
2    m2 1911925   8 197.563 3.000  < 2e-16  ***

AIC difference: 393.34, model m2 has lower AIC.
```

- Model `m2` is much better (AIC decrease > 2)!

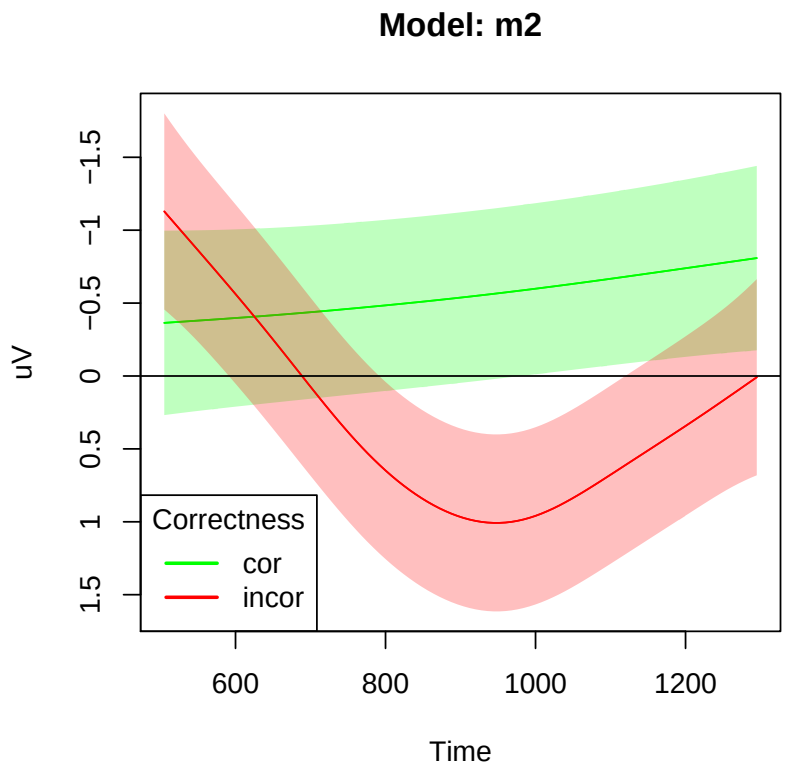
Visualizing both patterns: partial effects

```
> par(mfrow=c(1,2))
> plot(m2, select=1, shade=T, rug=F, ylim=c(2,-2), shade.col='green',
       seWithMean=T, ylab='uV', main='m1: correct'); abline(h = 0)
> plot(m2, select=2, shade=T, rug=F, ylim=c(2,-2), shade.col='red',
       seWithMean=T, ylab='uV', main='m1: incorrect'); abline(h = 0)
```



Visualizing both patterns: complete model context

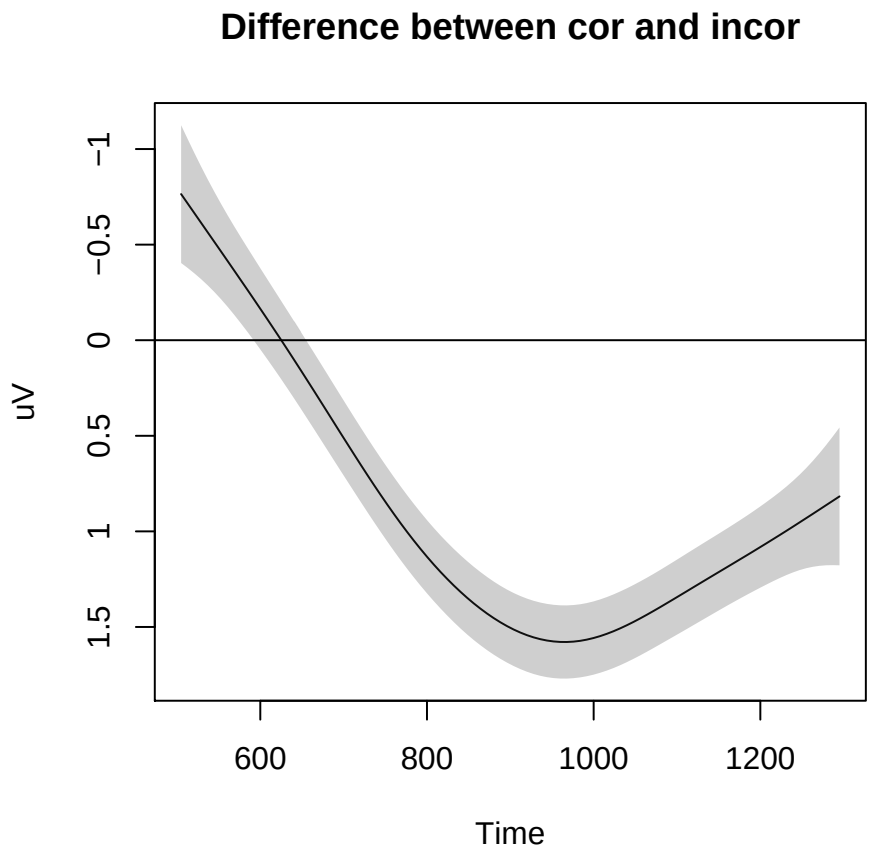
```
> plotSmooths(m2, "Time", "Correctness", colors=c('green','red'),
              dropRanef='Subject')
# Note that if there are additional variables in the model, plotSmooths
# sets these to the specified value (via the 'cond' parameter) or to their
# mean / most frequent value. Random-effect factors are excluded via
# dropRanef.
```



Visualizing the differences

(no individual variation w.r.t. the difference is taken into account: confidence bands **too** thin)

```
> plotDiff(m2, "Time", "Correctness") # custom function
```

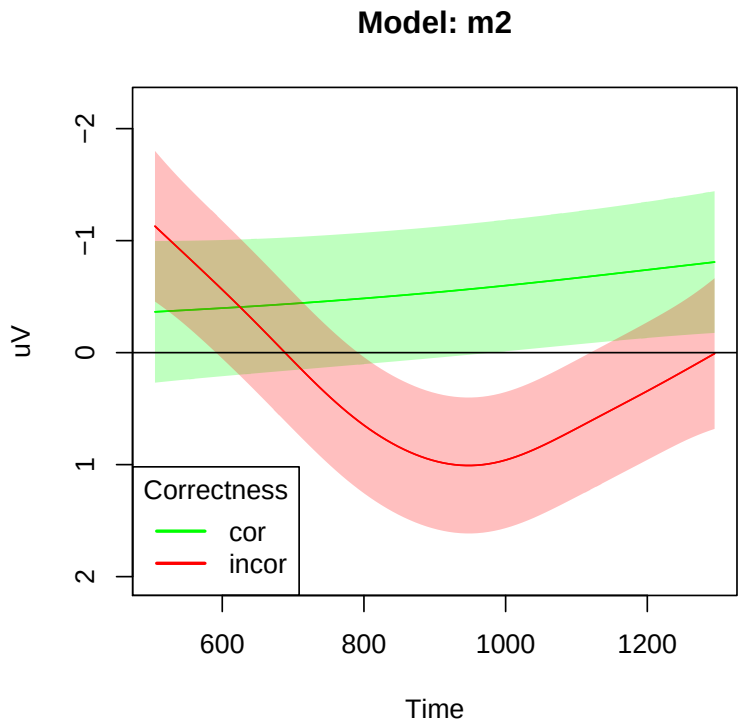


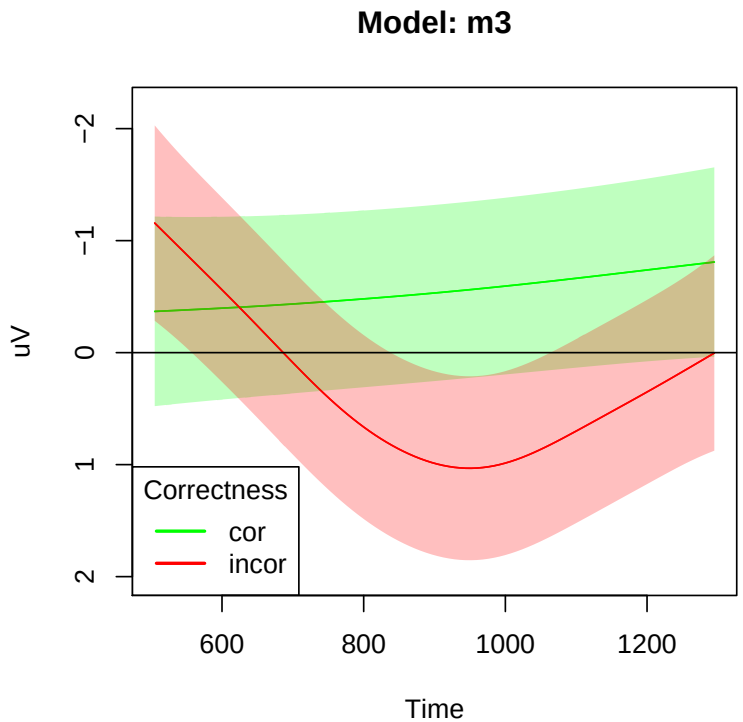
Including individual variation for correct vs. incorrect

```
> dat$SubjectCor = interaction(dat$Subject,dat$Correctness) # new factor
> m3 = bam(uV ~ s(Time,by=Correctness) + Correctness +
                s(Time,SubjectCor,bs='fs',m=1),
           data=dat, gc.level=2, method='ML')
> summary(m3)

...
Parametric coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.5538     0.3974  -1.393    0.164
Correctnessincor   0.8965     0.5638   1.590    0.112

Approximate significance of smooth terms:
                               edf   Ref.df      F p-value
s(Time):Correctnesscor       1.316    1.449  0.965   0.339
s(Time):Correctnessincor     4.125    4.939 17.724  <2e-16 ***
s(Time,SubjectCor)         469.508 1204.000 13.266  <2e-16 ***

R-sq.(adj) =  0.036   Deviance explained =  3.7% # was 2.1%
-ML = 1.9087e+06  Scale est. = 327.98    n = 442160
```

More uncertainty in the patterns

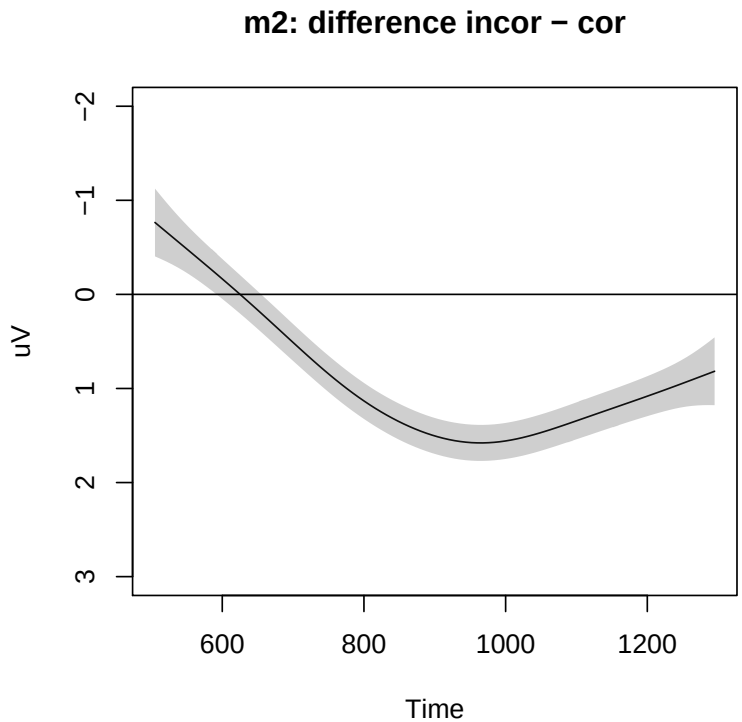
```
> par(mfrow=c(1,2))
> plotSmooths(m2, "Time", "Correctness", colors=c('green','red'),
              dropRanef='Subject', ylim=c(2,-2.2))
> plotSmooths(m3, "Time", "Correctness", colors=c('green','red'),
              dropRanef='SubjectCor', ylim=c(2,-2.2))
```

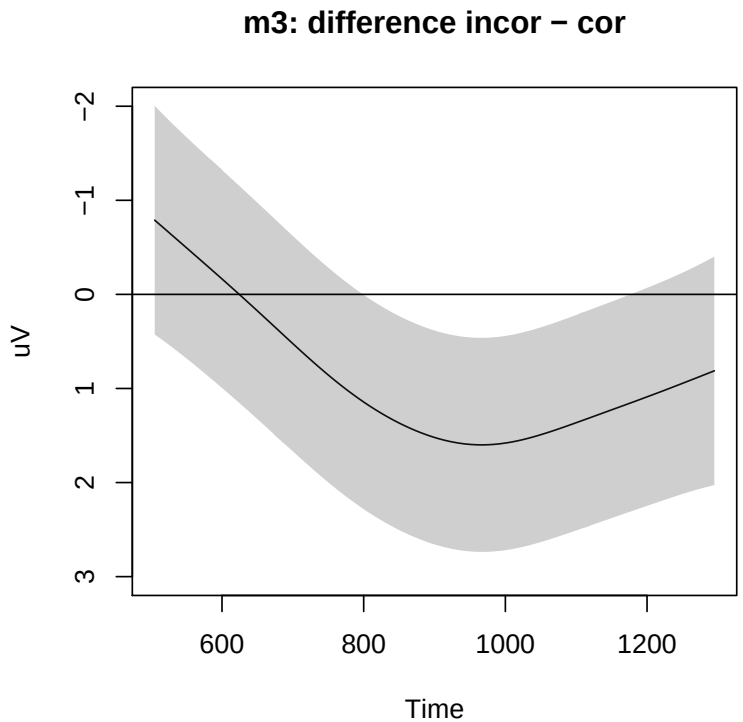




More uncertainty in the difference curve

```
> par(mfrow=c(1,2))
> plotDiff(m2, "Time", "Correctness", ylim=c(3,-2),
           main="m2: difference incor - cor")
> plotDiff(m3, "Time", "Correctness", ylim=c(3,-2),
           main="m3: difference incor - cor")
```





Explicit test for significant difference between levels

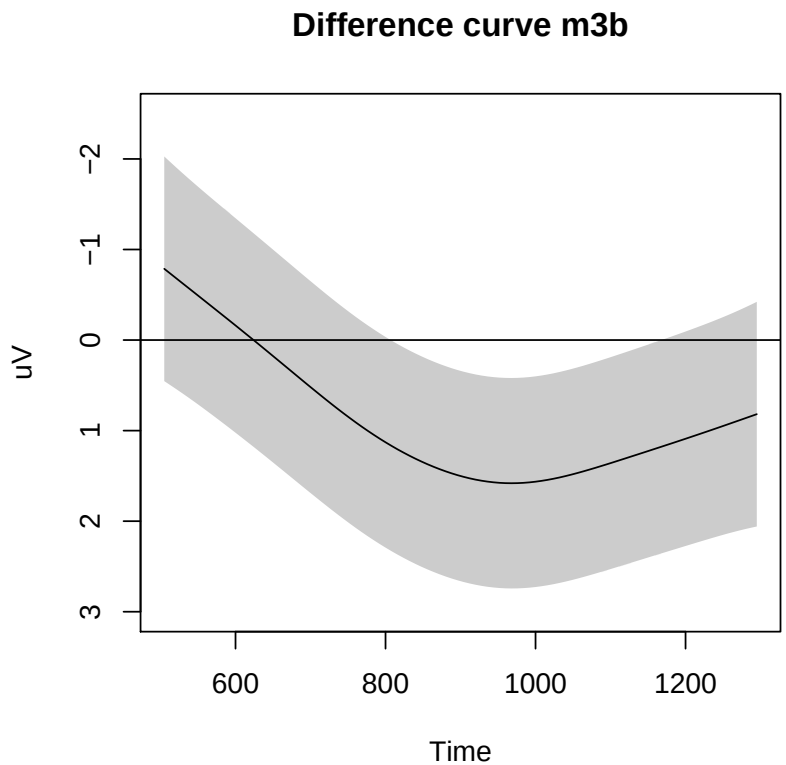
(smooths for a binary by-variable are **not** centered, so no extra term in the fixed effects)

```
# Note that the reference level influences the model. It makes sense
# to choose the reference level on the basis of your research question.
> dat$IsIncorrect = (dat$Correctness == 'incor')*1
> m3b = bam(uV ~ s(Time) + s(Time,by=IsIncorrect) +
                 s(Time,SubjectCor,bs='fs',m=1),
            data=dat, gc.level=2, method='ML')
> summary(m3b)

...
Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.5486     0.3983  -1.377    0.168

Approximate significance of smooth terms:
                           edf   Ref.df      F  p-value
s(Time)                  1.353    1.495  0.913    0.358
s(Time):IsIncorrect      4.947    5.724 12.007 9.27e-13 ***
s(Time,SubjectCor)     469.375 1204.000 13.266  < 2e-16 ***

R-sq.(adj) =  0.036   Deviance explained =  3.7%
-ML = 1.9087e+06  Scale est. = 327.98    n = 442160
```

Visualizing correct and incorrect difference

(identical shape as the difference curve of model m3)

```
> plot(m3b, shade=T, rug=F, ylim=c(1.5,-3.5), select=2,
       main='Difference curve m3b', ylab='uV', seWithMean=T)
> abline(h = 0)
```



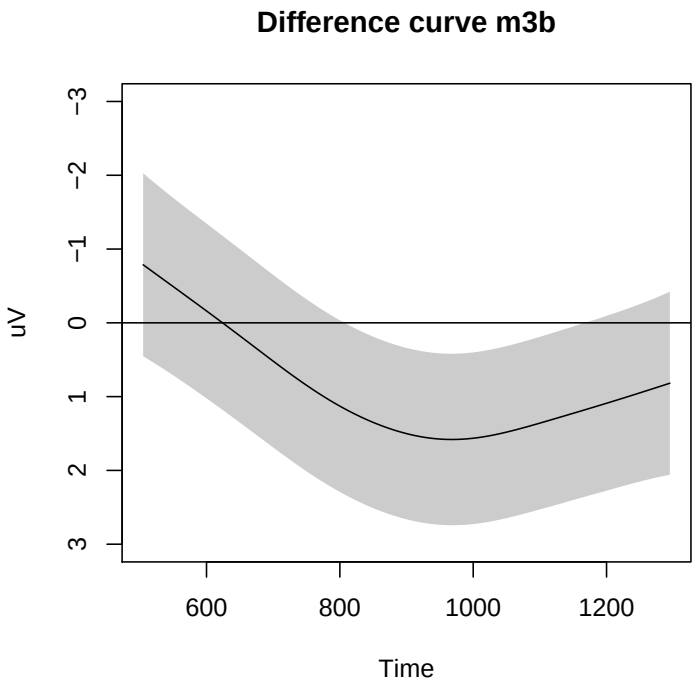
Separating intercept and smooth difference

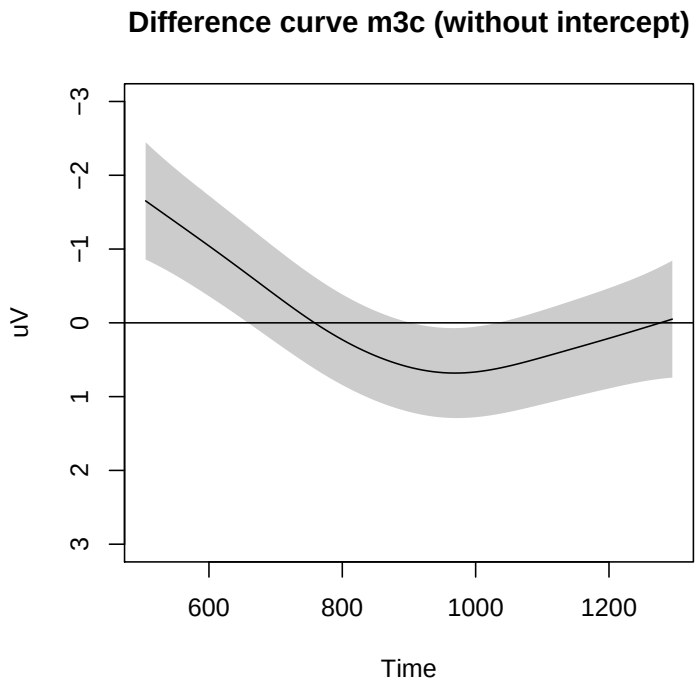
(the binary by-variable combines the intercept and smooth difference)

```
> dat$CorrectnessO = as.ordered(dat$Correctness)
> contrasts(dat$CorrectnessO) = 'contr.treatment' # contrast treatment
> m3c = bam(uV ~ s(Time) + s(Time,by=CorrectnessO) + CorrectnessO +
                 s(Time,SubjectCor,bs='fs',m=1),
            data=dat, gc.level=2, method='ML')
> summary(m3c)
...
Parametric coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)        -0.5380     0.3979  -1.352    0.176
CorrectnessOincor   0.8799     0.5640   1.560    0.119

Approximate significance of smooth terms:
                              edf   Ref.df      F  p-value
s(Time)                     1.572    1.783   1.02    0.342
s(Time):CorrectnessOincor   3.968    4.748 12.80 1.1e-11 ***
s(Time,SubjectCor)        469.380 1204.000 13.27 < 2e-16 ***

R-sq.(adj) =  0.036   Deviance explained =  3.7%
-ML = 1.9087e+06  Scale est. = 327.98    n = 442160
```

Visualizing difference between correct and incorrect

```
> plot(m3b, shade=T, rug=F, ylim=c(3,-4), select=2, ylab='uV',
      main='Difference curve m3b', seWithMean=T)
> abline(h = 0)
> plot(m3c, shade=T, rug=F, ylim=c(3,-4), select=2, ylab='uV',
      main='Difference curve m3c (without intercept)', seWithMean=T)
> abline(h = 0)
```

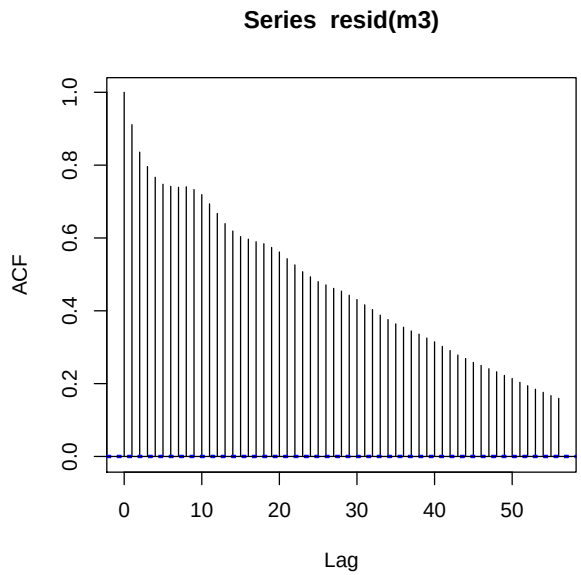




Autocorrelation in the data is a huge problem!

(residuals should be independent, otherwise the standard errors and *p*-values are wrong)

```
# It is essential that the data used in m3 is sorted per individual EEG
# signal sequence (per subject, per trial, per ROI, etc.). The best rho
# value can be found by trying different values and comparing models. (The
# autocorrelation at lag 1 for the same model w/o rho seems to work well,
# however.) A good approach is to use the lowest value possible which has
# a good acf plot and is not significantly worse than the best rho model.
> m3acf = acf(resid(m3)) # show autocorrelation
> rhoval = as.vector(m3acf[1]$acf)
> rhoval
[1] 0.91104 # correlation of residuals at time t with those at time t-1
```



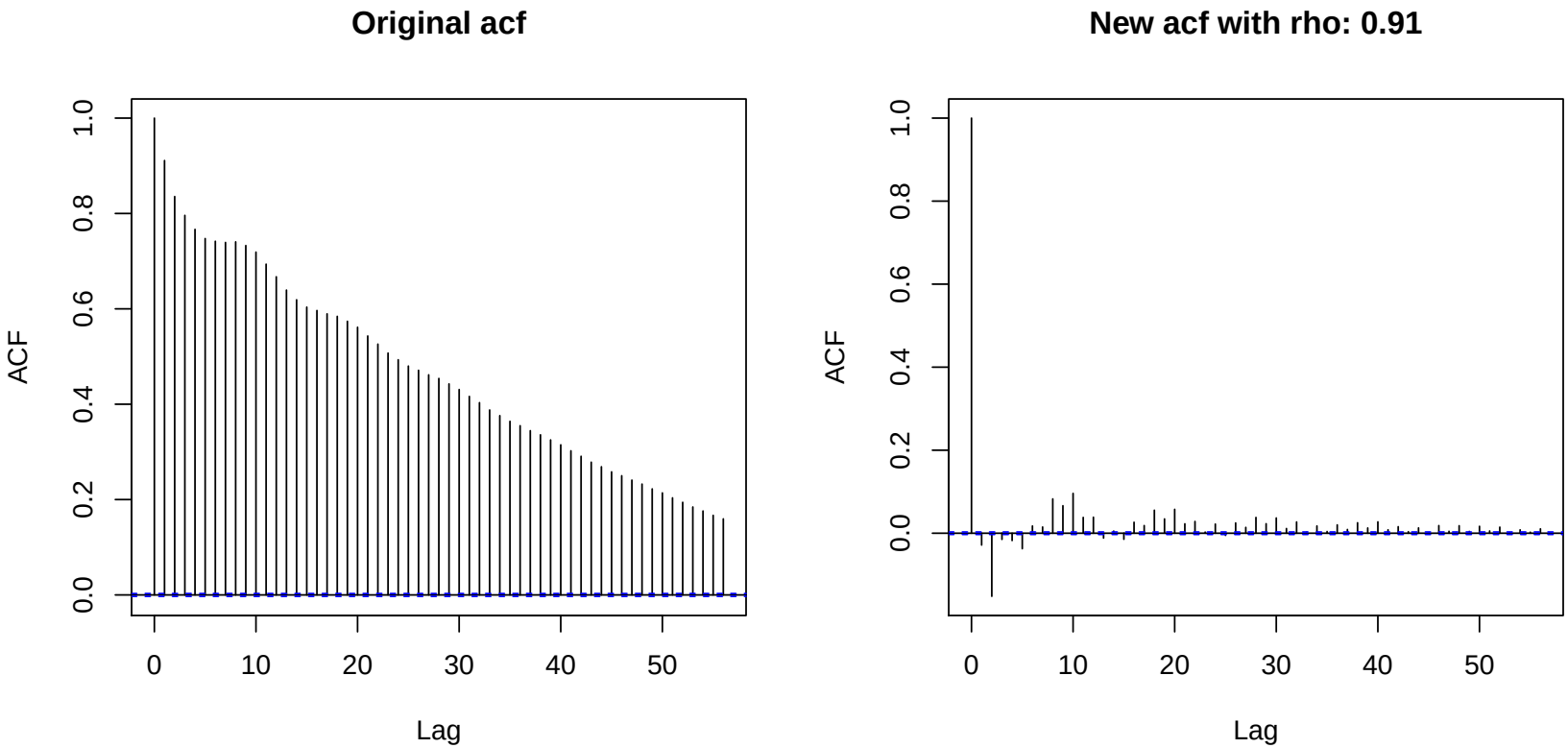
Correcting for autocorrelation

(`rho` can only be used with `bam`, not with `gam`)

```
> m4 = bam(uV ~ s(Time,by=Correctness) + Correctness +
                s(Time,SubjectCor,bs='fs',m=1), data=dat,
            gc.level=2, method='ML', rho=rhoval, AR.start=SeqStart)
> summary(m4)

...
Parametric coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.5453     0.4054  -1.345    0.179
Correctnessincor   0.8932     0.5746   1.554    0.120

Approximate significance of smooth terms:
                              edf   Ref.df     F  p-value
s(Time):Correctnesscor      1.011    1.022 0.323    0.575
s(Time):Correctnessincor    3.099    4.028 6.859 1.56e-05 ***
s(Time,SubjectCor)        116.095 1204.000 0.657  < 2e-16 ***

R-sq.(adj) =  0.0321   Deviance explained = 3.24%
-ML = 1.4923e+06  Scale est. = 287.48    n = 442160
```

Autocorrelation has been removed

```
> acf.new(m3, m4, "Time") # custom plotting function
```

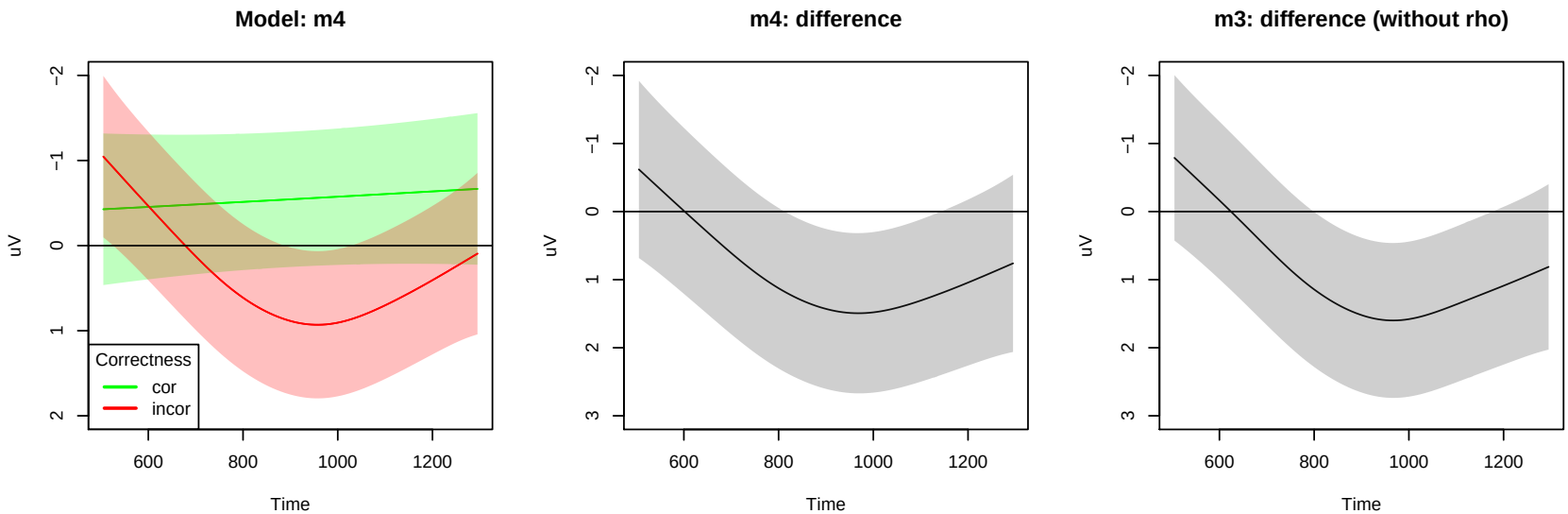


Clear model improvement

```
> compareML(m3,m4)

m3: uV ~ s(Time, by = Correctness) + Correctness +
         s(Time, SubjectCor, bs = "fs", m = 1)

m4: uV ~ s(Time, by = Correctness) + Correctness +
         s(Time, SubjectCor, bs = "fs", m = 1)

Chi-square test of ML scores
-----
  Model    Score Edf       Chisq    Df  p.value Sig.
1    m3 1908692   8
2    m4 1492261   8 416431.071 0.000  < 2e-16  ***

Warning message:
In compareML(m3, m4) :
AIC is not reliable, because an AR1 model is included ...

> AIC(m3) - AIC(m4) # AIC indeed does not work when using rho
[1] -1428.872
```

Resulting smooths: slightly wider confidence bands

```
> par(mfrow=c(1,3))
> plotSmooths(m4, "Time", "Correctness", colors=c('green','red'),
              dropRanef="SubjectCor", ylim=c(2,-2))
> plotDiff(m4, "Time", "Correctness", ylim=c(3,-2), main='m4: difference')
> plotDiff(m3, "Time", "Correctness", ylim=c(3,-2),
           main='m3: difference (without rho)')
```

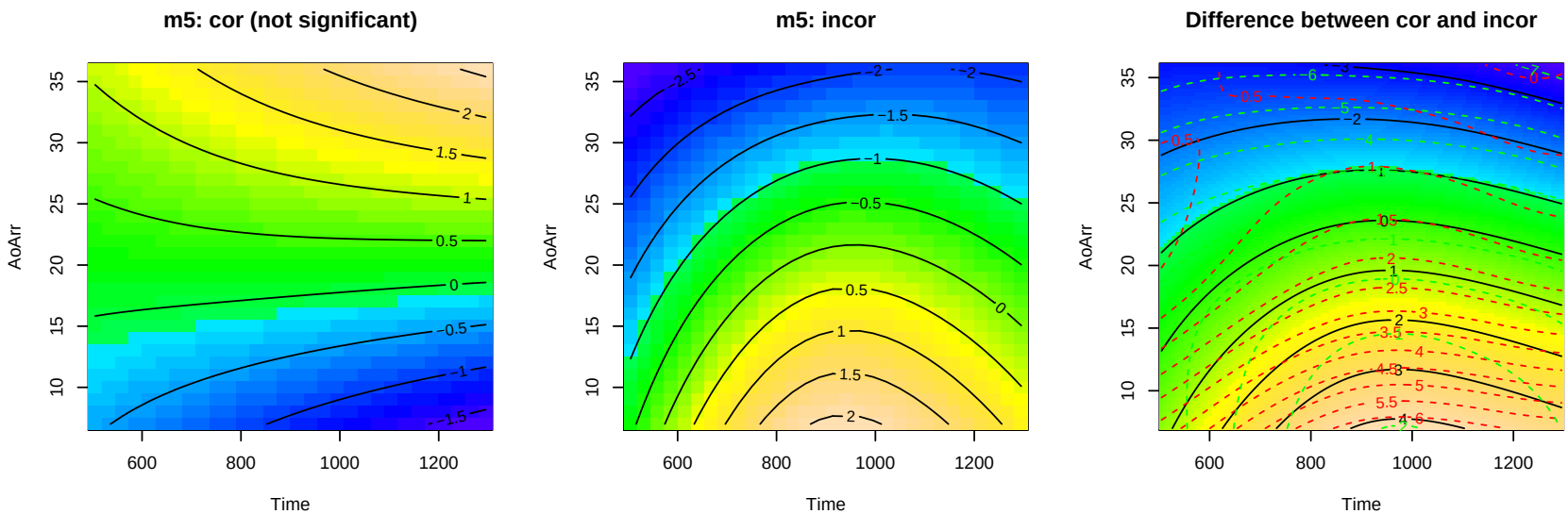


Our research question: the effect of age of arrival

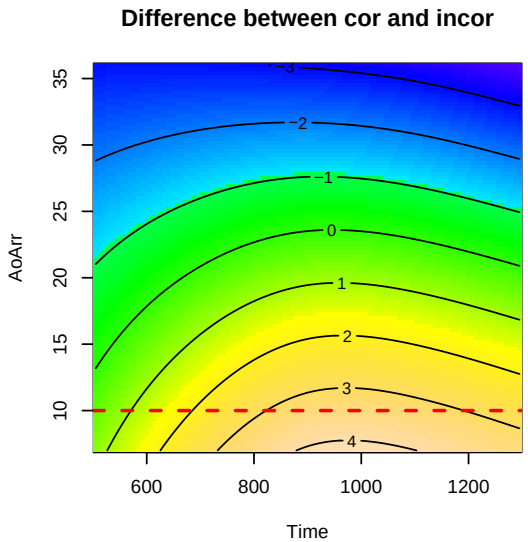
```
# in this lecture, rho is held constant at 0.91
> m5 = bam(uV ~ te(Time,AoArr,by=Correctness) + Correctness +
                s(Time,SubjectCor,bs='fs',m=1), data=dat,
            gc.level=2, method='ML', rho=rhoval, AR.start=SeqStart)

> summary(m5)

...
Parametric coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.5589     0.3962  -1.411   0.1584
Correctnessincor   0.9260     0.5615   1.649   0.0991 .

Approximate significance of smooth terms:
                                    edf   Ref.df     F p-value
te(Time,AoArr):Correctnesscor     3.250    3.482 1.565   0.186
te(Time,AoArr):Correctnessincor   5.871    6.956 4.707 2.93e-05 ***
s(Time,SubjectCor)              113.435 1202.000 0.628  < 2e-16 ***

R-sq.(adj) =  0.0322   Deviance explained = 3.25%
-ML = 1.4923e+06  Scale est. = 287.48    n = 442160
```

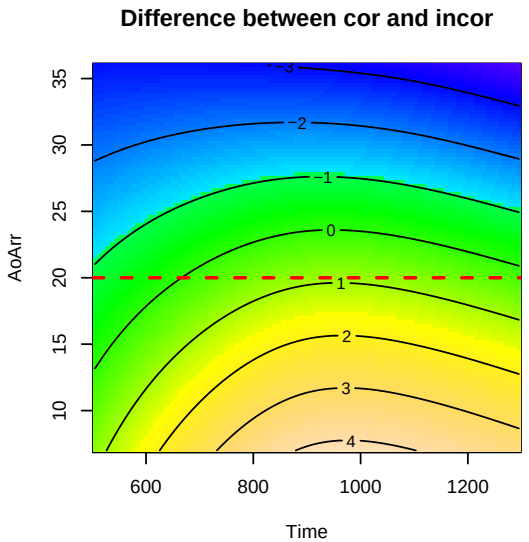
Visualization of the 2-dimensional smooths

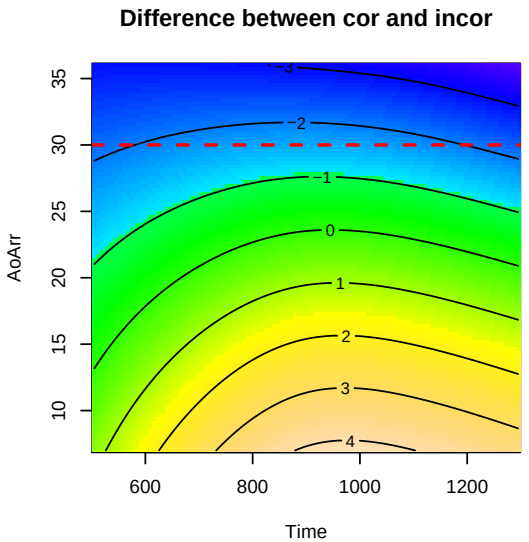
```
# Custom plotting functions are used to plot the partial effects
# (i.e. the general pattern) and difference surfaces.
# N.B. a surface including all effects (at pre-specified levels) can be
# visualized using vis.gam(...) - see additional slides at the end.

> par(mfrow=c(1,3))
> pvis.gam(m5, plot.type='contour', view=c('Time','AoArr'), select=1,
           color='topo', main='m5: cor (not significant)')
> pvis.gam(m5, plot.type='contour', view=c('Time','AoArr'), select=2,
           color='topo', main='m5: incor')
> plotDiff2D(m5, "Time", "AoArr", "Correctness", plotCI=T)
```

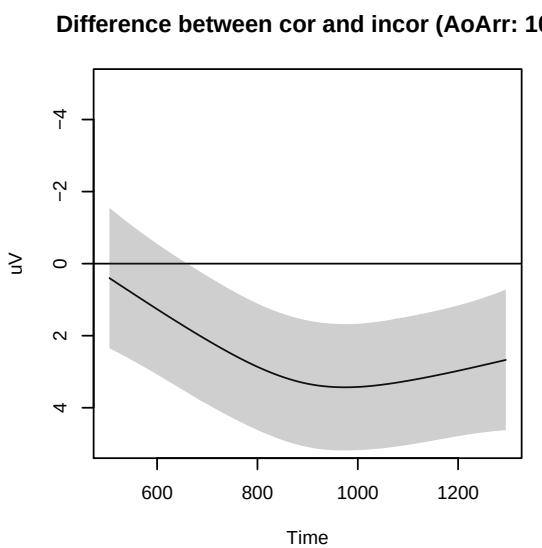


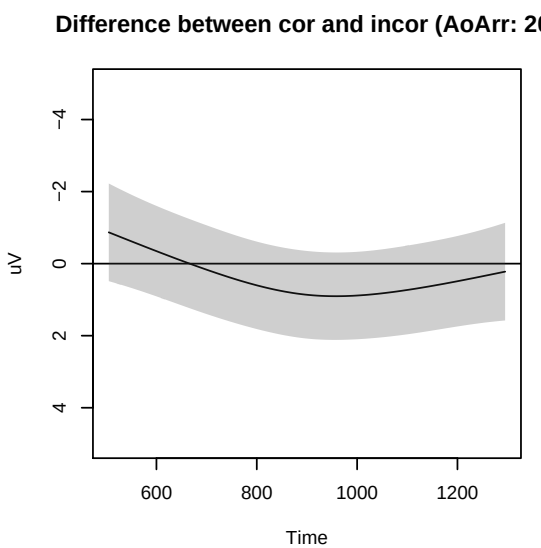
Interpreting 2-dimensional smooths

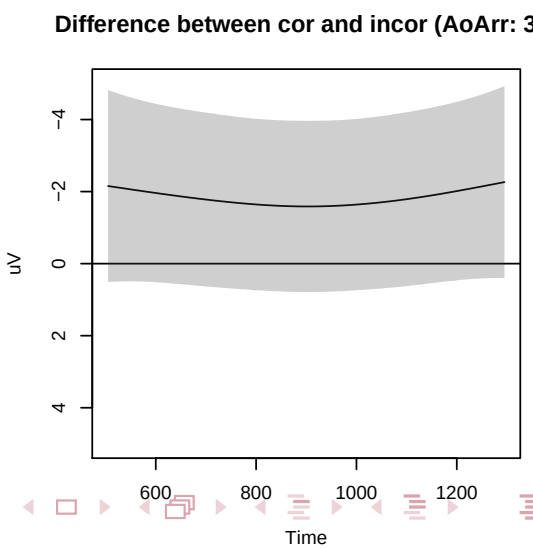












Significance testing using a binary variable

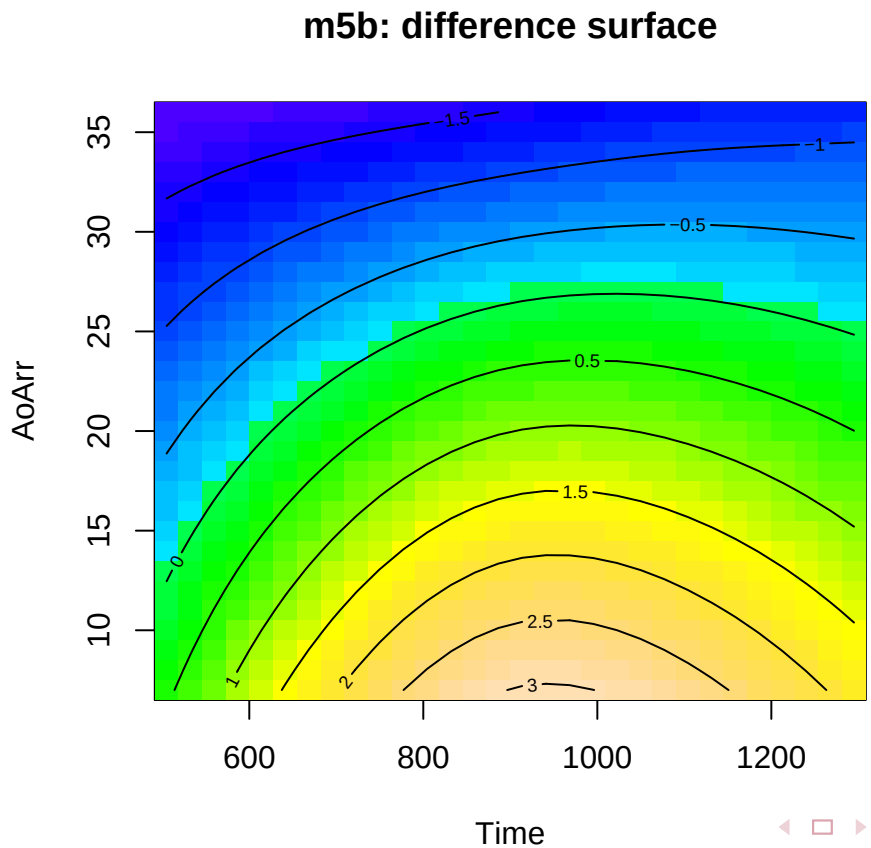
(an ordered factor may be used to separate the intercept difference from the difference surface)

```
> m5b = bam(uV ~ te(Time,AoArr) + te(Time,AoArr,by=IsIncorrect) +
                 s(Time,SubjectCor,bs='fs',m=1), data=dat,
            gc.level=2, method='ML', rho=rhoval, AR.start=SeqStart)
> summary(m5b)
...
Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.5222     0.4004  -1.304    0.192

Approximate significance of smooth terms:
                                edf   Ref.df     F p-value
te(Time,AoArr)                1.088   20.000 0.083 0.12947
te(Time,AoArr):IsIncorrect    6.610    7.631 3.402 0.00087 ***
s(Time,SubjectCor)          114.647 1203.000 0.636 < 2e-16 ***

R-sq.(adj) =  0.0321   Deviance explained = 3.24%
-ML = 1.4923e+06  Scale est. = 287.48    n = 442160
```

Visualizing the difference surface

(note that `m5b` is a different model than `m5`, so the difference surfaces do not **need** to be identical)

```
> pvis.gam(m5b, plot.type='contour', view=c('Time','AoArr'),
           contour.col='black', select=2, color='topo',
           main='m5b: difference surface')
```



Assessing if k needs to be increased

(note that a single `k` value specifies the wigglyness per dimension in `te()` / in total in `s()`)

```
# the wigglyness may also specified separately per dimension for te()
> m5c = bam(uV ~ te(Time,AoArr,by=Correctness,k=c(10,10)) + Correctness +
                 s(Time,SubjectCor,bs='fs',m=1), data=dat,
             gc.level=2, method='ML', rho=rhoval, AR.start=SeqStart)
> compareML(m5,m5c) # the additional complexity is not needed
...
Chi-square test of ML scores
-----
  Model    Score Edf Chisq    Df  p.value Sig.
1   m5c 1492256  14
2    m5 1492255  14 0.271 0.000  < 2e-16  ***
...

> summary(m5c) # max edf: 99 (10^2 - 1)
...
                                    edf Ref.df     F  p-value
te(Time,AoArr):Correctnesscor     3.239  3.465 1.644    0.167     # was 3.3
te(Time,AoArr):Correctnessincor   6.397  8.010 4.072 7.31e-05 *** # was 5.9
...
```

Adding factor smooths per word(-correctness)

```
> dat$WordCor = interaction(dat$Word,dat$Correctness)
> m6 = bam(uV ~ te(Time,AoArr,by=Correctness) +
                Correctness + s(Time,SubjectCor,bs='fs',m=1) +
                s(Time,WordCor,bs='fs',m=1), data=dat, gc.level=2,
            method='ML', rho=rhoval, AR.start=SeqStart) # 15 hours...
> summary(m6)
...
Parametric coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.5888     0.5571  -1.057    0.291
Correctnessincor   0.7473     0.7804   0.958    0.338

Approximate significance of smooth terms:
                                      edf   Ref.df     F p-value
te(Time,AoArr):Correctnesscor       3.106    3.207 1.774    0.145
te(Time,AoArr):Correctnessincor     5.864    6.943 4.793 2.32e-05 ***
s(Time,SubjectCor)                111.028 1202.000 0.624  < 2e-16 ***
s(Time,WordCor)                   133.202 1726.000 0.260  < 2e-16 ***

R-sq.(adj) =  0.0511   Deviance explained = 5.17% # was 3.25%
-ML = 1.4921e+06  Scale est. = 287.19    n = 442160
```

Model comparison for factor smooths

```
> compareML(m5r, m6r)

# m5r and m6r are the same as m5 and m6, but with method='REML'
# (required when comparing random effects)

m5r: uV ~ te(Time, AoArr, by = Correctness) + Correctness +
          s(Time, SubjectCor, bs = "fs", m = 1)
m6r: uV ~ te(Time, AoArr, by = Correctness) + Correctness +
          s(Time, SubjectCor, bs = "fs", m = 1) +
          s(Time, WordCor, bs = "fs", m = 1)

Chi-square test of REML scores
-----
  Model   Score  Edf    Chisq    Df p.value Sig.
1   m5r 1492248  14
2   m6r 1492137  16 110.657 2.000  < 2e-16  ***
```

Decomposition: the pure effect of age of arrival

(note that the difference between the 2 ti-surfaces is also non-significant, not shown)

```
> m6b = bam(uV ~ s(Time,by=Correctness) + s(AoArr,by=Correctness) +
                 ti(Time,AoArr,by=Correctness) + Correctness +
                 s(Time,SubjectCor,bs='fs',m=1) +
                 s(Time,WordCor,bs='fs',m=1), data=dat, gc.level=2,
            method='ML', rho=rhoval, AR.start=SeqStart)
> summary(m6b)
...
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)        -0.5790     0.4606  -1.257    0.209
Correctnessincor    0.8279     0.6527   1.269    0.205

Approximate significance of smooth terms:
                                     edf   Ref.df     F p-value
s(Time):Correctnesscor             1.003    1.006 0.322   0.5715
s(Time):Correctnessincor           3.100    4.029 6.866 1.54e-05 ***
s(AoArr):Correctnesscor            1.002    1.002 2.883   0.0894 .
s(AoArr):Correctnessincor          1.002    1.002 4.268   0.0388 *
ti(Time,AoArr):Correctnesscor      1.263    1.491 2.059   0.1346
ti(Time,AoArr):Correctnessincor    1.390    1.723 0.163   0.8177
s(Time,SubjectCor)               111.134 1202.000 0.624  < 2e-16 ***
s(Time,WordCor)                  133.225 1726.000 0.260  < 2e-16 ***
```

A simpler model without the non-linear interaction

```
> m6c = bam(uV ~ s(Time,by=Correctness) + s(AoArr,by=Correctness) +
                 Correctness + s(Time,SubjectCor,bs='fs',m=1) +
                 s(Time,WordCor,bs='fs',m=1), data=dat, gc.level=2,
             method='ML', rho=rhoval, AR.start=SeqStart)
> summary(m6c)
...
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)        -0.5789     0.4607  -1.257    0.209
Correctnessincor    0.8279     0.6527   1.268    0.205

Approximate significance of smooth terms:
                                edf   Ref.df     F p-value
s(Time):Correctnesscor        1.008    1.016 0.329   0.5700
s(Time):Correctnessincor      3.101    4.030 6.868 1.53e-05 ***
s(AoArr):Correctnesscor       1.009    1.010 2.793   0.0941 .
s(AoArr):Correctnessincor     1.013    1.015 4.131   0.0415 *
s(Time,SubjectCor)          111.255 1202.000 0.623  < 2e-16 ***
s(Time,WordCor)             134.127 1726.000 0.260  < 2e-16 ***

R-sq.(adj) =  0.0511   Deviance explained = 5.16%
-ML = 1.4921e+06  Scale est. = 287.19    n = 442160
```

Model comparison: the simpler model is sufficient

```
> compareML(m6b,m6c)

m6b: uV ~ s(Time, by = Correctness) + s(AoArr, by = Correctness) +
          ti(Time, AoArr, by = Correctness) + Correctness +
          s(Time, SubjectCor, bs = "fs", m = 1) +
          s(Time, WordCor, bs = "fs", m = 1)

m6c: uV ~ s(Time, by = Correctness) + s(AoArr, by = Correctness) +
          Correctness + s(Time, SubjectCor, bs = "fs", m = 1) +
          s(Time, WordCor, bs = "fs", m = 1)

Chi-square test of ML scores
-----
  Model   Score Edf Chisq    Df p.value Sig.
1   m6c 1492148  14
2   m6b 1492146  20 1.568 6.000   0.792

Warning messages:
1: In compareML(m6b, m6c) : AIC is not reliable, ...
2: In compareML(m6b, m6c) : Only small difference in ML...
```

Testing for significant smooth differences

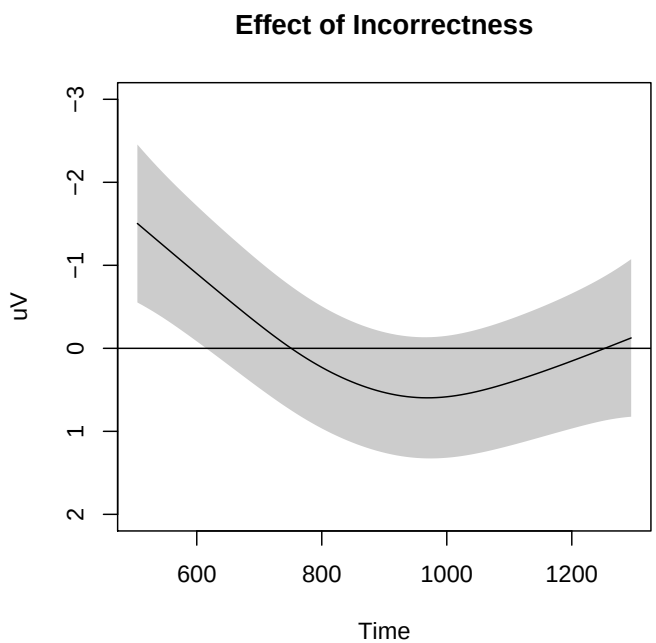
(a binary variable may only occur once in a model, so ordered factors are essential here)

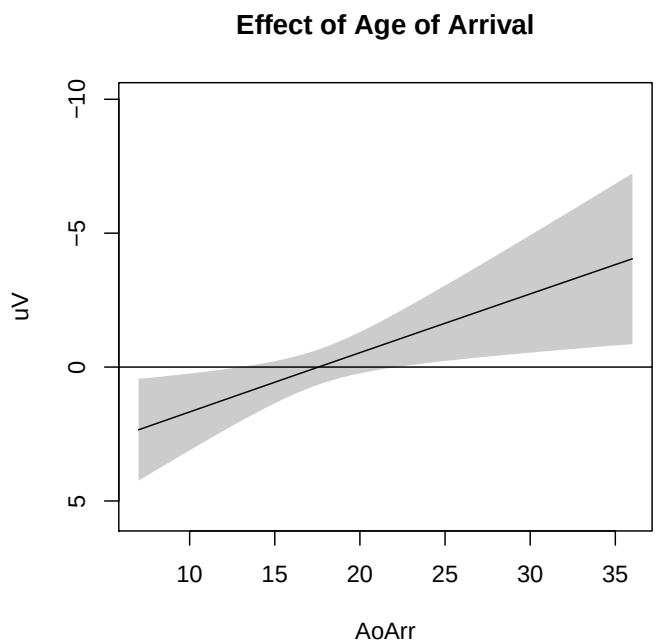
```
> m7 = bam(uV ~ s(Time) + s(Time,by=CorrectnessO) + s(AoArr) +
               s(AoArr,by=CorrectnessO) + CorrectnessO + ...)
> summary(m7)
...
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         -0.5768     0.4607  -1.252    0.211
CorrectnessOincor    0.8262     0.6527   1.266    0.206

Approximate significance of smooth terms:
                                edf   Ref.df     F p-value
s(Time)                       1.055    1.102 0.389  0.55289
s(Time):CorrectnessOincor     3.096    4.023 6.118 6.26e-05 ***
s(AoArr)                      1.000    1.000 2.801  0.09423 .
s(AoArr):CorrectnessOincor    1.013    1.015 6.837  0.00871 **
s(Time,SubjectCor)          110.670 1202.000 0.623  < 2e-16 ***
s(Time,WordCor)             134.023 1726.000 0.260  < 2e-16 ***
...
```

- Thus: the data clearly show a non-linear effect of time and age of arrival

Difference curves (smooths only)

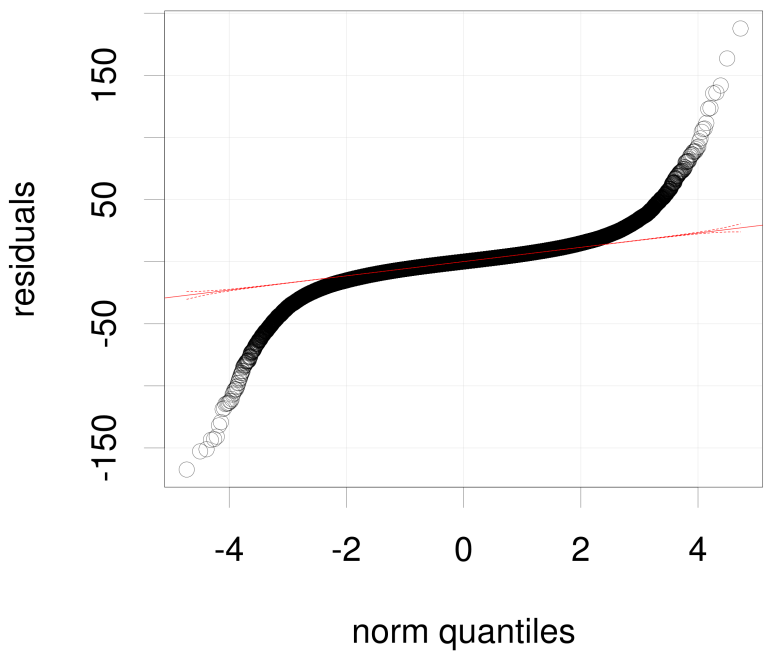
```
> par(mfrow=c(1,2))
> plot(m7, shade=T, rug=F, select=2, ylim=c(2,-3),
       main='Effect of Incorrectness', ylab='uV', seWithMean=T)
> abline(h = 0)
> plot(m7, shade=T, rug=F, select=4, ylim=c(5.5,-10),
       main='Effect of Age of Arrival', ylab='uV', seWithMean=T)
> abline(h = 0)
```

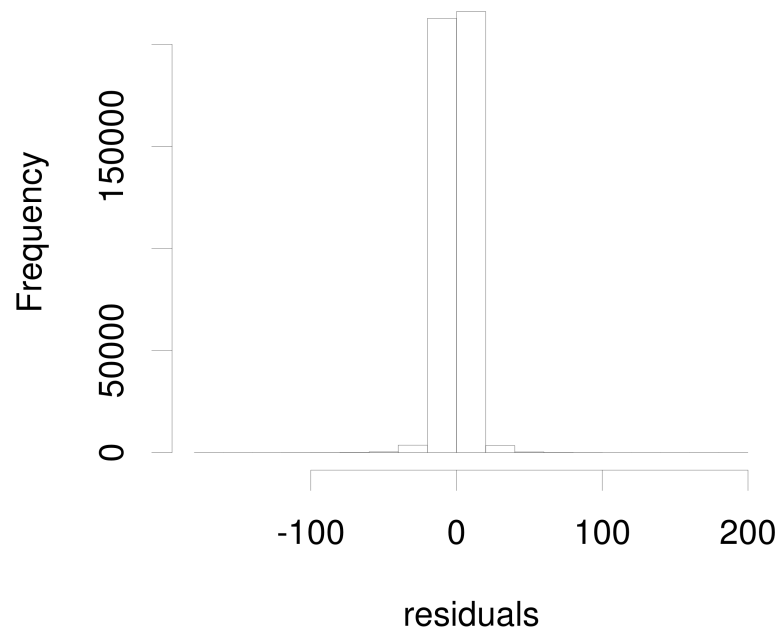




Finally: model criticism

```
> library(car)
> par(mfrow=c(1,2))
> qqplot.rho(m7) # custom qqplot of residuals taking rho into account
> hist.rho(m7) # custom histogram of residuals taking rho into account
```





Approach to normalize residuals

- This type of residual distribution is common for EEG data
- These deviations are very problematic: incorrect *p*-values...
- Solution: model fitting using scaled-*t* distribution
  - Implemented in `gam`: `family="scat"`
  - Adding rho to `gam` possible via custom function (created by Natalya Pya)
  - Unfortunately, `gam` is too slow, so a `bam` approach is in development
  - Current state: implemented for `fREML` estimation with pre-specified scaled-*t* parameters (θ)
  - Thus `gam` still necessary to determine θ (but applied to subset / simpler model specification)

Estimating scaled-t parameters efficiently

1. Fit an intercept-only scaled-*t* `gam` model (including rho, `method="REML"`) to all data and extract θ
2. Fit the complete model without random effects as a scaled-*t* `gam` model (including rho, `method="REML"`) to all data and extract θ
3. Compare the models' θ values:
   - Relatively small differences: use θ of the complex model in the `bam` estimation
   - Relatively large differences: determine θ by fittting the complete model specification (including random effects) to a small subset of the data (randomly select trials, compare θ across a few runs)

Fitting a scaled-*t* model

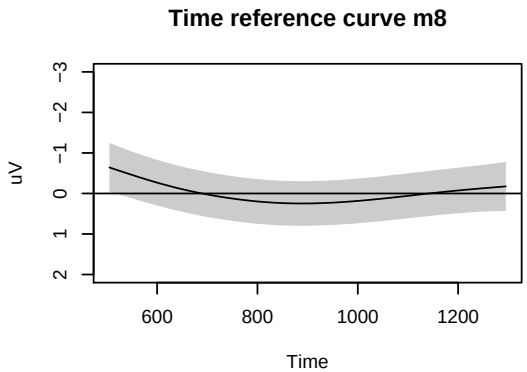
(does not work for binomial models)

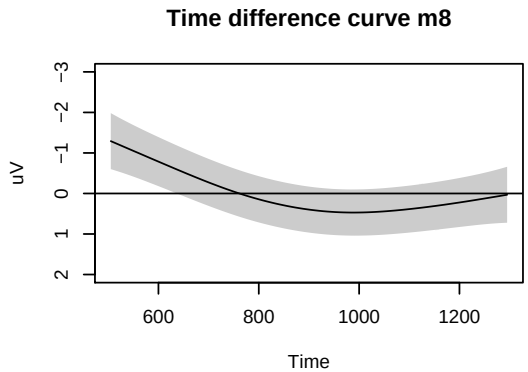
```
> source('bam.art.fit.R') # custom functions by Natalya Pya
> g0 <- gam(uV ~ 1, data=dat, method='REML',
            family=scat(rho=rhoval, AR.start=dat$SeqStart))
> g1 <- gam(uV ~ s(Time) + s(Time,by=CorrectnessO) + s(AoArr) +
                 s(AoArr,by=CorrectnessO) + CorrectnessO,
            data=dat, method='REML',
            family=scat(rho=rhoval, AR.start=dat$SeqStart)))
> theta0 = g0$family$getTheta(TRUE)
> theta1 = g1$family$getTheta(TRUE) # 4.566764 5.264096
> theta0 - theta1
[1] -0.003382087 -0.002104720 # OK, not too high

# for simplicity and speed we ignore by-word factor smooths here
# (those should be included in the final model, however)
> m8 = bam(uV ~ s(Time) + s(Time,by=CorrectnessO) + s(AoArr) +
               s(AoArr,by=CorrectnessO) + CorrectnessO +
               s(Time,SubjectCor,bs='fs',m=1), data=dat, gc.level=2,
               method='fREML', family=art(theta=theta1,rho=rhoval),
               AR.start=SeqStart)) # 12 hours (35 min. for Gaussian)!

# for a fair comparison a Gaussian model (m8g) was fitted using fREML
```

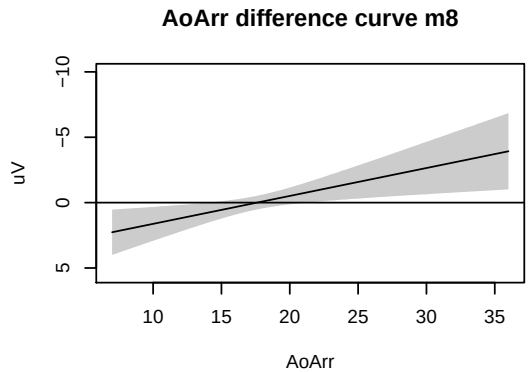
Using the scaled-*t* distribution: *p*-values change

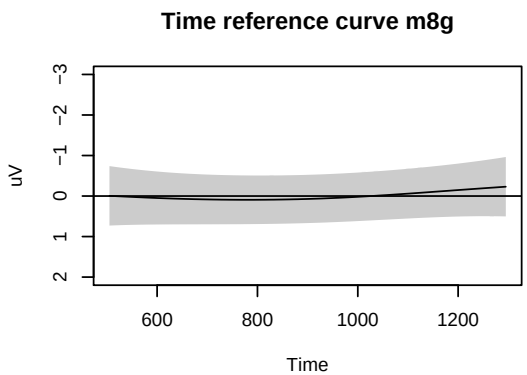
```
> summary(m8)
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.002761   0.369947  -0.007    0.994
CorrectnessOincor  0.842288   0.523107   1.610    0.107

s(Time)                          3.600    4.643 4.280  0.00107 **
s(Time):CorrectnessOincor        3.475    4.485 7.940 8.71e-07 ***
s(AoArr)                         1.692    1.714 1.164  0.29437
s(AoArr):CorrectnessOincor       1.001    1.001 7.496  0.00617 **
s(Time,SubjectCor)             122.258 1202.000 1.795  < 2e-16 ***
R-sq.(adj) =  0.0291   Deviance explained = 1.9%

> summary(m8g)
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         -0.5370     0.4006   -1.34     0.18
CorrectnessOincor    0.9076     0.5672    1.60     0.11

s(Time)                          1.660    2.085 0.577 0.56858
s(Time):CorrectnessOincor        3.058    3.968 4.414 0.00152 **
s(AoArr)                         1.698    1.758 1.501 0.21636
s(AoArr):CorrectnessOincor       1.004    1.004 6.431 0.01114 *
s(Time,SubjectCor)             110.461 1202.000 0.609 < 2e-16 ***
R-sq.(adj) =  0.0321   Deviance explained = 3.24%
```

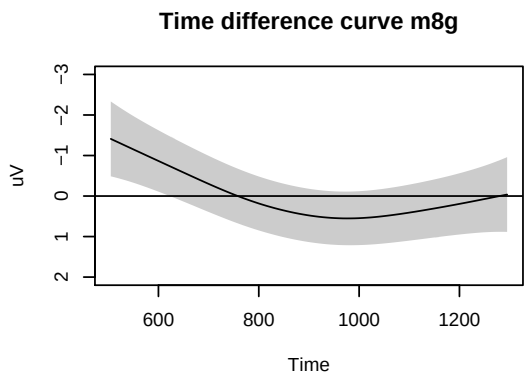
Using the scaled-*t* distribution: patterns remain similar

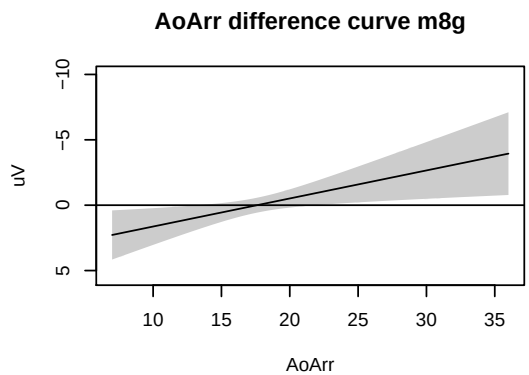






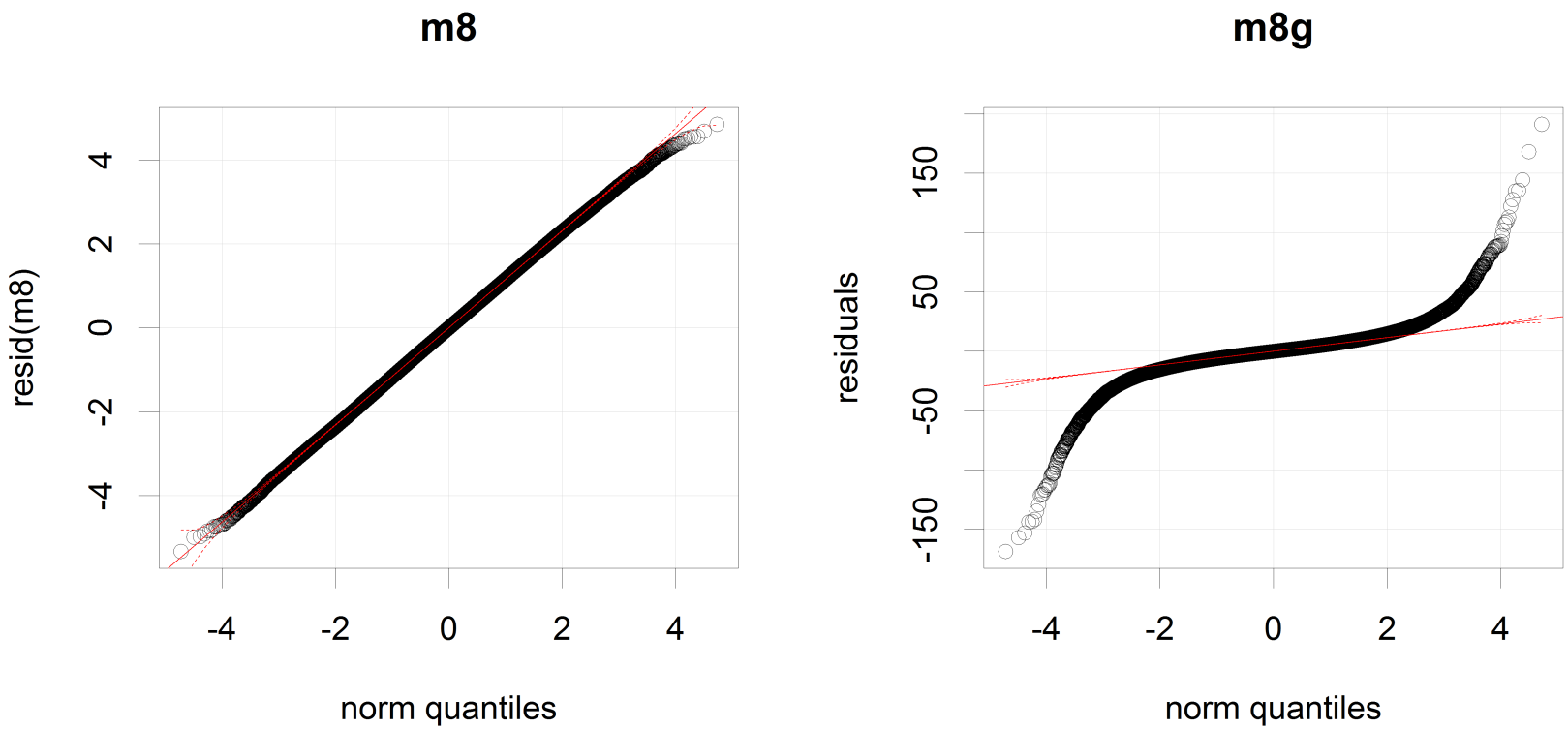






Model criticism: much improved!

```
> library(car)
> par(mfrow=c(1,2))
> qqp(resid(m8), main='m8') # resid of scaled-t model takes rho into account
> qqplot.rho(m8g, main='m8g') # custom qqplot of resid. accounting for rho
```



Discussion

- Still much to do: e.g., testing the significance of other possibly important variables (proficiency, age, etc.)
- But don't make it too complex!
  - There is much variation present in EEG data and adding very complex surfaces will almost certainly improve your model significantly
  - keep it simple, otherwise you won't be able to compute/interpret the results
  - Also note that for hypothesis testing, it is essential not to conduct step-wise approaches

Conclusion

- GAMs are very useful to analyze EEG and other time-series data
  - The method is very suitable to detect non-linear patterns, while taking into account individual variation and correcting for autocorrelation
- If you would like to get some hands-on experience, you can use the lab session available here:
  `http://www.let.rug.nl/wieling/statscourse/lecture4`

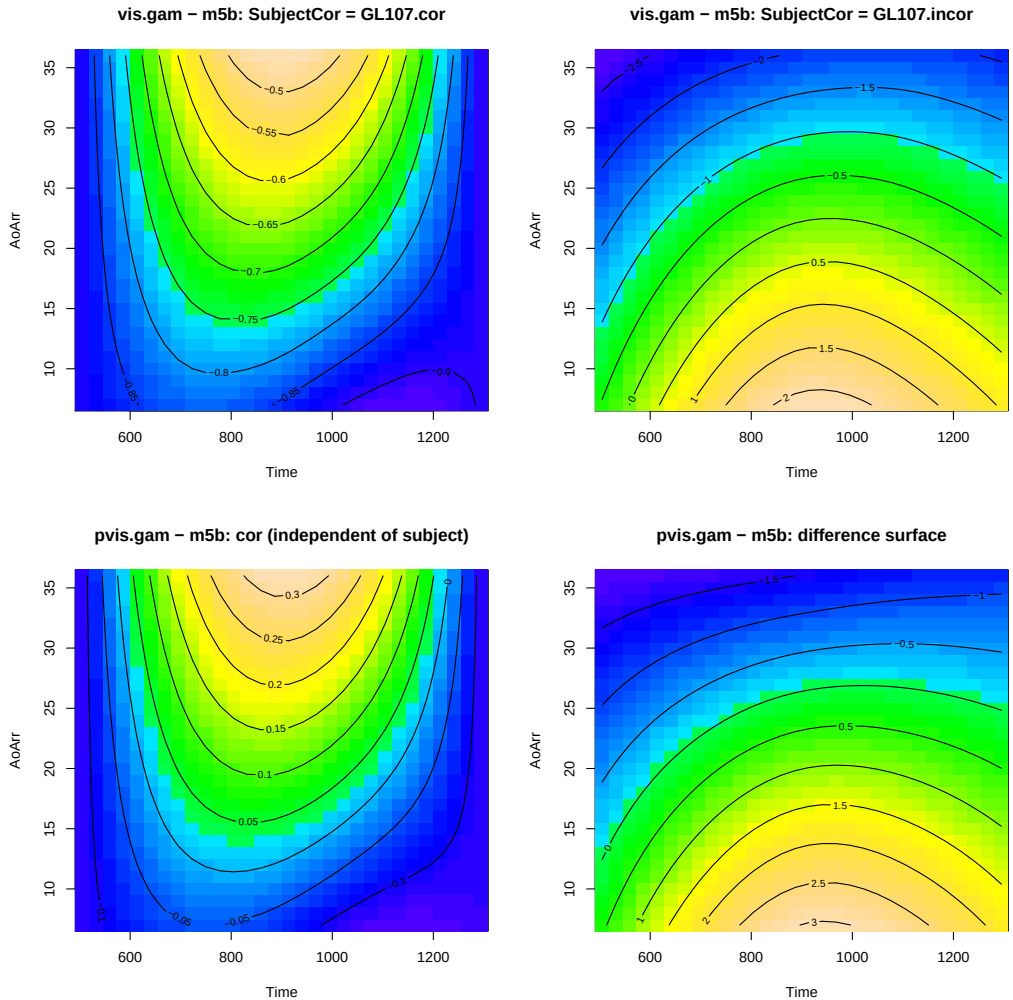
Thank you for your attention!

Visualization: vis.gam vs. pvis.gam (1)

```
> m5b = bam(uV ~ te(Time,AoArr) + te(Time,AoArr,by=IsIncorrect) +
                 s(Time,SubjectCor,bs='fs',m=1), data=dat,
            gc.level=2, method='ML', rho=rhoval, AR.start=SeqStart)

> par(mfrow=c(2,2))
> vis.gam(m5b, plot.type='contour', view=c('Time','AoArr'),
          cond=list(IsIncorrect=0,Subject='GL107.cor'),
          contour.col='black', color='topo',
          main='vis.gam - m5b: SubjectCor = GL107.cor')
> vis.gam(m5b, plot.type='contour', view=c('Time','AoArr'),
          cond=list(IsIncorrect=1,Subject='GL107.incor'),
          contour.col='black', color='topo',
          main='vis.gam - m5b: SubjectCor = GL107.incor')
> pvis.gam(m5b, plot.type='contour', view=c('Time','AoArr'),
           contour.col='black', select=1, color='topo',
           main='pvis.gam - m5b: cor (independent of subject)')
> pvis.gam(m5b, plot.type='contour', view=c('Time','AoArr'),
           contour.col='black', select=2, color='topo',
           main='pvis.gam - m5b: difference surface')
```

Visualization: vis.gam vs. pvis.gam (2)

(vis.gam shows the resulting surface of the conditions, pvis.gam shows the smooths in your model)



Differences between ML, REML and GCV

- `ML` is conservative, but the variance component (i.e. smooth) is biased (oversmoothed)
- `(f)REML` is less conservative, but robust to moderate autocorrelation, and the smooth is not biased
- `GCV.Cp` is better for prediction, but not robust to autocorrelation
- Strongest approach is to compare results across different methods (i.e. using AIC/(RE)ML)