Addressing residual autocorrelation with generalized additive mixed models

R. Harald Baayen & Jacolien van Rij
University of Tübingen

1 Preliminaries

Packages that we will use are:

```
require(xtable,quietly=TRUE)
require(lme4,quietly=TRUE)
suppressMessages(require(mgcv))
require(RePsychLing,quietly=TRUE)
suppressMessages(require(texreg))
suppressMessages(require(itsadug))
require(lattice, quietly=TRUE)
```

2 The KKL dataset

We take the KKL dataset from `RePsychLing`, and create a separate variable for the interaction of `spt` and `orn`.

```
dat <- KKL
dat$FirstTrial <- dat$first==1
mm <- model.matrix(~ sze*(spt+obj+grv)*orn, data=KKL)
dat$spt_orn = mm[,11]
```

2.1 The LMM

As starting point, we take the LMM obtained with `lmer`, but now refit it with `mgcv`. Random effects are modeled as smooths with `re` basis functions. Random slopes for subject are specified as `s(subj, bs="re")`, and by-subject random contrasts for `obj` as `s(subj, orn, bs="re")`. We use the `bam()` function rather than the `gam` function because it evaluates more quickly, albeit at the potential cost of a slight loss of precision. Even so, be aware that `bam` evaluates very much more slowly than `lmer`.

```
dat.gam0 <- bam(lrt ~ sze * (spt + obj + grv) * orn +
                      s(subj, bs="re") +
                      s(subj, spt, bs="re") +
                      s(subj, grv, bs="re") +
                      s(subj, obj, bs="re") +
                      s(subj, orn, bs="re") +
                      s(subj, spt_orn, bs="re"),
                data=dat, method="ML")
summary(dat.gam0)

Family: gaussian
Link function: identity

Formula:
```

```
lrt ~ sze * (spt + obj + grv) * orn + s(subj, bs = "re") + s(subj,
    spt, bs = "re") + s(subj, grv, bs = "re") + s(subj, obj,
    bs = "re") + s(subj, orn, bs = "re") + s(subj, spt_orn, bs = "re")

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.69103    0.01727  329.50  < 2e-16
sze          0.18414    0.03454    5.33  9.8e-08
spt          0.07440    0.00770    9.66  < 2e-16
obj          0.04086    0.00450    9.09  < 2e-16
grv         -0.00153    0.00533   -0.29   0.7748
orn          0.04087    0.01044    3.92  9.0e-05
sze:spt      0.04880    0.01540    3.17   0.0015
sze:obj     -0.01069    0.00899   -1.19   0.2345
sze:grv     -0.03616    0.01066   -3.39   0.0007
sze:orn      0.01650    0.02087    0.79   0.4291
spt:orn      0.02020    0.00669    3.02   0.0025
obj:orn      0.00922    0.00734    1.26   0.2093
grv:orn      0.01105    0.00737    1.50   0.1338
sze:spt:orn -0.01267    0.01337   -0.95   0.3433
sze:obj:orn -0.00195    0.01469   -0.13   0.8945
sze:grv:orn -0.04353    0.01474   -2.95   0.0031

Approximate significance of smooth terms:
                 edf Ref.df      F p-value
s(subj)         83.6     84 8022.1 < 2e-16
s(subj,spt)     76.2     84 6497.1 < 2e-16
s(subj,grv)     47.8     84  152.6 2.3e-11
s(subj,obj)     31.7     84  176.8 0.00028
s(subj,orn)     80.8     84   99.9 < 2e-16
s(subj,spt_orn) 44.0     84   23.1 9.2e-09

R-sq.(adj) =  0.477   Deviance explained = 48.1%
-ML = -12294  Scale est. = 0.036202  n = 53765
```

Inspection of the residuals (Figure 1)

```
acf(resid(dat.gam0))
```

reveals autocorrelational structure. However, when applying the general `acf` function to the residuals, we do not distinguish between the individual time series constituted by the data for the separate subjects, and may therefore obtain imprecise and sometimes misleading information about the autocorrelations. We therefore inspect the autocorrelations for the individual subjects using a trellis graph (Figure 2).

```
dfr = acf_resid(dat.gam0,        # acf_resid from package itsadug
  split_pred=list(subj=dat$subj),
  plot=FALSE,
  return_all=TRUE)$dataframe
civec = dfr[dfr$lag==0,]$ci      # vector of confidence intervals for xyplot
xyplot(acf ~ lag | subj, type = "h", data = dfr, col.line = "black",
  panel = function(...) {
    panel.abline(h = civec[panel.number()], col.line = "grey")
    panel.abline(h = -civec[panel.number()], col.line = "grey")
```

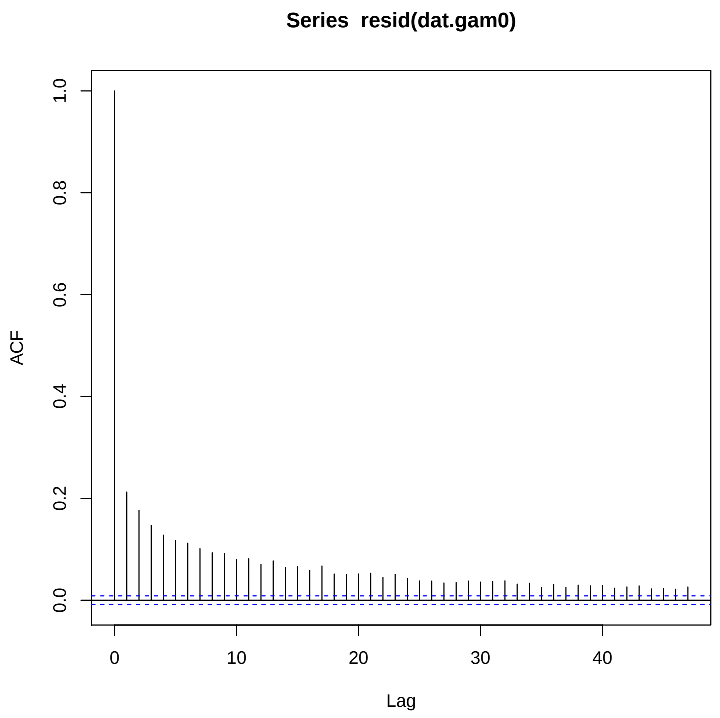

Figure 1: Acf function for the initial model without differentiating for individual time series.

```
    panel.abline(h = 0, col.line = "black")
    panel.xyplot(...)
  },
  strip = strip.custom(bg = "grey90"),
  par.strip.text = list(cex = 0.8),
  xlab="lag", ylab="autocorrelation")
```

Note the presence of substantial variation between subjects with respect to the magnitude of the autocorrelations, also with respect to the number of lags at which these autocorrelations persist.

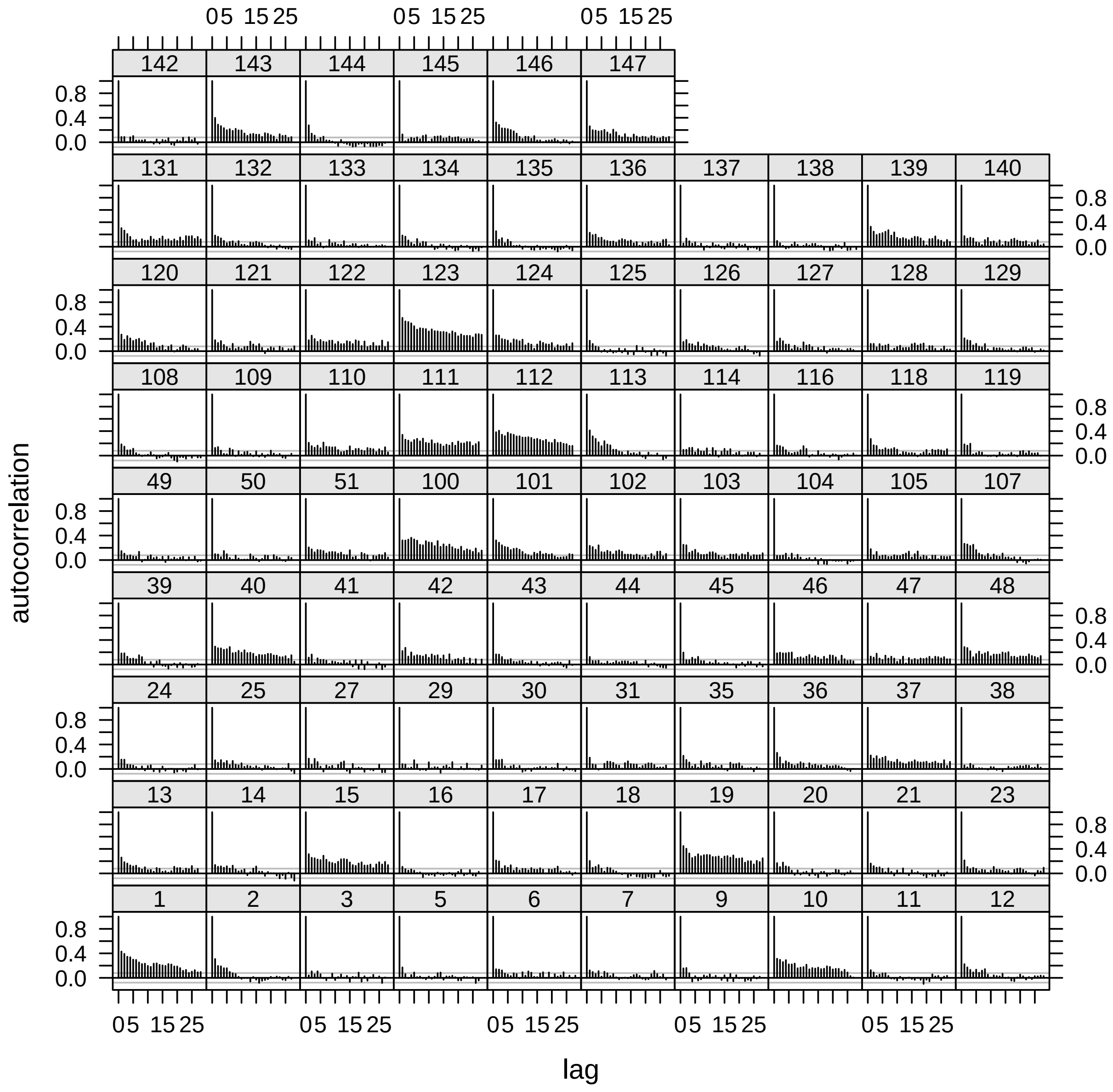

Figure 2: Autocorrelations in the initial model for each subject-specific time series.

2.2 By-subject factor smooths for trial

To reduce the autocorrelation in the errors, we add a by-subject factor smooth for trial with the directive

```
s(trial, subj, bs="fs", m=1).
```

Here, a factor smooth basis is requested (`bs="fs"`) with shrinkage (`m=1`). Factor smooths are appropriate when smooths are required for a factor with a large number of levels, and each smooth should have the same smoothing parameter. The `fs` smoothers have penalties on each null space component, which with `m=1` are set to order 1, so that we have the nonlinear 'wiggly' counterpart of what in a linear mixed model would be handled with by-subject random intercepts and by-subject random slopes. The model with by-subject factor smooths for trial is:

```
dat.gam1 <- bam(lrt ~ sze * (spt + obj + grv) * orn +
                      s(subj, spt, bs="re") +
                      s(subj, grv, bs="re") +
                      s(subj, obj, bs="re") +
                      s(subj, orn, bs="re") +
                      s(subj, spt_orn, bs="re") +
                      s(trial, subj, bs="fs", m=1),
                data=dat, method="ML")
```

As the factor smooths 'absorb' the random intercepts, no separate request for random intercepts is required. The model summary

```
summary(dat.gam1)
```

```

Family: gaussian
Link function: identity

Formula:
lrt ~ sze * (spt + obj + grv) * orn + s(subj, spt, bs = "re") +
    s(subj, grv, bs = "re") + s(subj, obj, bs = "re") + s(subj,
    orn, bs = "re") + s(subj, spt_orn, bs = "re") + s(trial,
    subj, bs = "fs", m = 1)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.728103   0.017819  321.47  < 2e-16
sze          0.179647   0.035637    5.04  4.6e-07
spt          0.074483   0.007763    9.59  < 2e-16
obj          0.041530   0.004425    9.39  < 2e-16
grv         -0.000546   0.004921   -0.11  0.91161
orn          0.019875   0.009761    2.04  0.04173
sze:spt      0.046126   0.015526    2.97  0.00297
sze:obj     -0.008455   0.008849   -0.96  0.33933
sze:grv     -0.036411   0.009842   -3.70  0.00022
sze:orn      0.049243   0.019521    2.52  0.01165
spt:orn      0.022882   0.006500    3.52  0.00043
obj:orn      0.008648   0.007027    1.23  0.21843
grv:orn      0.012527   0.007055    1.78  0.07579
sze:spt:orn -0.013802   0.013001   -1.06  0.28843
sze:obj:orn -0.004611   0.014053   -0.33  0.74281
```

```
sze:grv:orn -0.047642   0.014110   -3.38  0.00073

Approximate significance of smooth terms:
                  edf Ref.df       F p-value
s(subj,spt)      76.9     84   28.02 < 2e-16
s(subj,grv)      44.5     84    1.82 4.2e-09
s(subj,obj)      35.0     84    1.77 3.0e-05
s(subj,orn)      53.4     84    1.88 < 2e-16
s(subj,spt_orn)  46.1     84    1.27 3.1e-10
s(trial,subj)   643.8    772 1266.20 < 2e-16

R-sq.(adj) =  0.526   Deviance explained = 53.4%
-ML = -14212  Scale est. = 0.032854   n = 53765
```

indicates that the factor smooths are well supported, but it might be that a simpler model with by-subject random intercepts only is just as good. Model comparison shows this not to be the case.

```
compareML(dat.gam0, dat.gam1)    # from package itsadug

dat.gam0: lrt ~ sze * (spt + obj + grv) * orn + s(subj, bs = "re") + s(subj,
     spt, bs = "re") + s(subj, grv, bs = "re") + s(subj, obj,
     bs = "re") + s(subj, orn, bs = "re") + s(subj, spt_orn, bs = "re")

dat.gam1: lrt ~ sze * (spt + obj + grv) * orn + s(subj, spt, bs = "re") +
     s(subj, grv, bs = "re") + s(subj, obj, bs = "re") + s(subj,
     orn, bs = "re") + s(subj, spt_orn, bs = "re") + s(trial,
     subj, bs = "fs", m = 1)

Chi-square test of ML scores
-----
     Model  Score Edf    Chisq    Df  p.value Sig.
1 dat.gam0 -12294  22
2 dat.gam1 -14212  23 1917.248 1.000  < 2e-16  ***

AIC difference: 4686.84, model dat.gam1 has lower AIC.
```

We again inspect the residuals for autocorrelation, which is substantially reduced but not completely eliminated (Figure 3).

```
dfr = acf_resid(dat.gam1,
  split_pred=list(subj=dat$subj),
  plot=FALSE,
  return_all=TRUE)$dataframe
civec = dfr[dfr$lags==0,]$ci
xyplot(acf ~ lag | subj, type = "h", data = dfr, col.line = "black",
  panel = function(...) {
    panel.abline(h = civec[panel.number()], col.line = "grey")
    panel.abline(h = -civec[panel.number()], col.line = "grey")
    panel.abline(h = 0, col.line = "black")
    panel.xyplot(...)
  },
  strip = strip.custom(bg = "grey90"),
  par.strip.text = list(cex = 0.8),
  xlab="lag", ylab="autocorrelation")
```

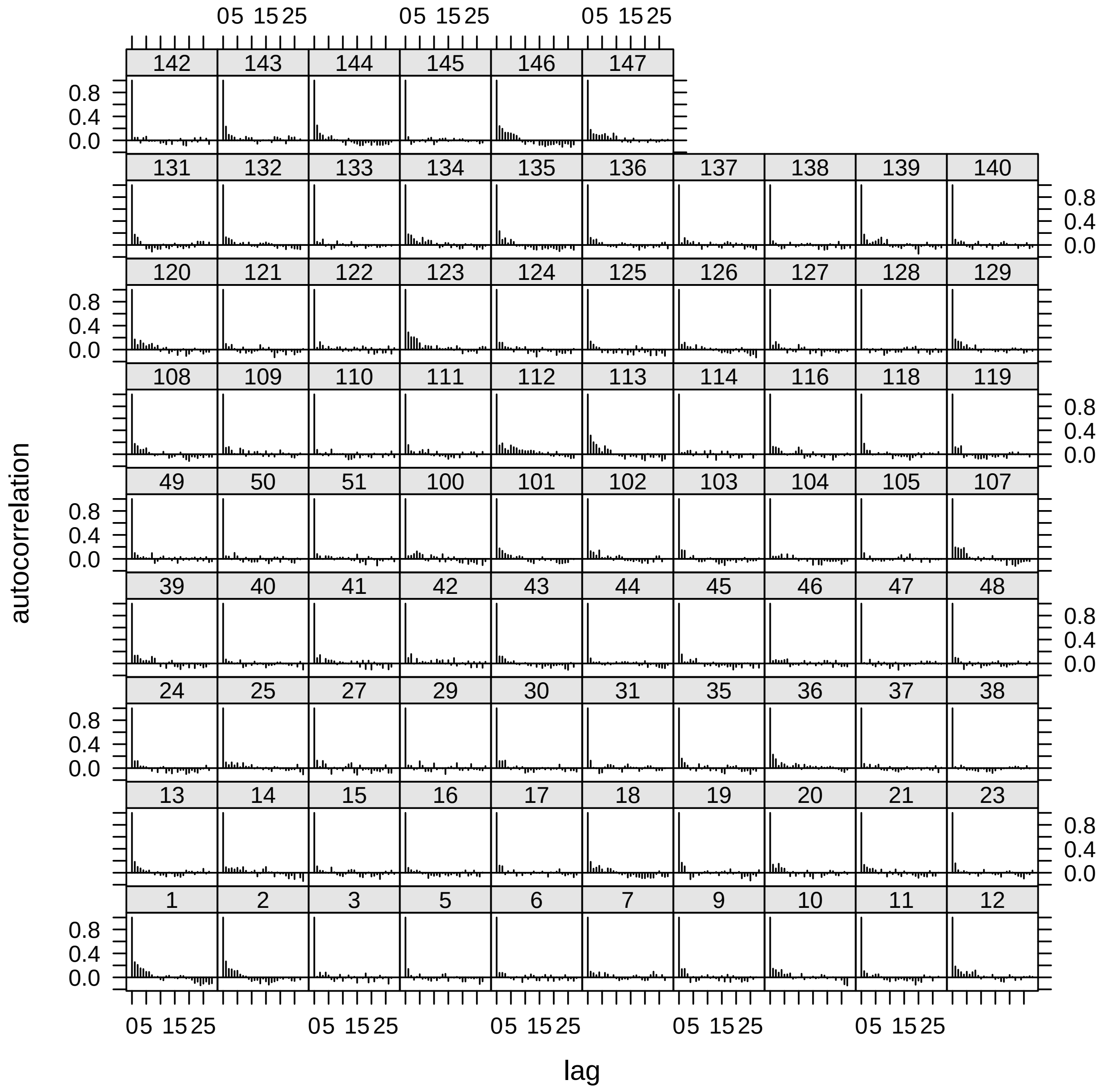

Figure 3: Autocorrelation functions for each subject-specific time series in the model with factor smooths.

2.3 An AR1 process in the errors

Since there is still some small autocorrelational structure in the residuals, we further whiten the errors by filtering out a mild autocorrelative process with proportionality $\rho = 0.15$. To do so, we need to tell `bam` where new time series begin. This is accomplished with the variable `FirstTrial`, which is true whenever a new time series starts (which is the case whenever trial==1), and false elsewhere. Crucially, the rows in the data frame should be ordered by subject, and within subject, by trial. The directives for `bam` are:

```
AR.start=dat$FirstTrial, rho=0.15
```

We fit the model and summarize it.

```
dat.gam2 <- bam(lrt ~ sze * (spt + obj + grv) * orn +
                      s(subj, spt, bs="re") +
                      s(subj, grv, bs="re") +
                      s(subj, obj, bs="re") +
                      s(subj, orn, bs="re") +
                      s(subj, spt_orn, bs="re") +
                      s(trial, subj, bs="fs", m=1),
                      AR.start=dat$FirstTrial, rho=0.15,
                data=dat, method="ML")
summary(dat.gam2)
```

```
Family: gaussian
Link function: identity

Formula:
lrt ~ sze * (spt + obj + grv) * orn + s(subj, spt, bs = "re") +
    s(subj, grv, bs = "re") + s(subj, obj, bs = "re") + s(subj,
    orn, bs = "re") + s(subj, spt_orn, bs = "re") + s(trial,
    subj, bs = "fs", m = 1)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.725191   0.017730  322.90  < 2e-16
sze          0.180231   0.035461    5.08  3.7e-07
spt          0.073017   0.007893    9.25  < 2e-16
obj          0.040762   0.004103    9.93  < 2e-16
grv         -0.000501   0.004892   -0.10  0.91849
orn          0.019850   0.008984    2.21  0.02714
sze:spt      0.047950   0.015787    3.04  0.00239
sze:obj     -0.007886   0.008206   -0.96  0.33657
sze:grv     -0.035036   0.009783   -3.58  0.00034
sze:orn      0.048386   0.017967    2.69  0.00708
spt:orn      0.021655   0.006405    3.38  0.00072
obj:orn      0.008665   0.006881    1.26  0.20794
grv:orn      0.008930   0.006907    1.29  0.19604
sze:spt:orn -0.009092   0.012811   -0.71  0.47790
sze:obj:orn -0.007498   0.013762   -0.54  0.58584
sze:grv:orn -0.048455   0.013813   -3.51  0.00045

Approximate significance of smooth terms:
                  edf Ref.df      F p-value
s(subj,spt)      77.7     84  31.76 < 2e-16
```

```
s(subj,grv)      46.3     84   2.04 3.4e-10
s(subj,obj)      28.3     84   1.24  0.0011
s(subj,orn)      41.9     84   1.05 < 2e-16
s(subj,spt_orn)  46.9     84   1.31 9.5e-11
s(trial,subj)   604.0    772 666.70 < 2e-16

R-sq.(adj) =  0.525   Deviance explained = 53.2%
-ML = -14756  Scale est. = 0.033118  n = 53765
```

By-subject acf plots

```
dfr = acf_resid(dat.gam2,        # acf_resid from package itsadug
  split_pred=list(subj=dat$subj),
  plot=FALSE,
  return_all=TRUE)$dataframe
civec = dfr[dfr$lag==0,]$ci      # vector of confidence intervals for xyplot
xyplot(acf ~ lag | subj, type = "h", data = dfr, col.line = "black",
  panel = function(...) {
    panel.abline(h = civec[panel.number()], col.line = "grey")
    panel.abline(h = -civec[panel.number()], col.line = "grey")
    panel.abline(h = 0, col.line = "black")
    panel.xyplot(...)
  },
  strip = strip.custom(bg = "grey90"),
  par.strip.text = list(cex = 0.8),
  xlab="lag", ylab="autocorrelation")
```

show that an occasional subject still has some remaining autocorrelation. However, further increasing ρ would induce spurious autocorrelations for subjects with hardly any autocorrelations in their residuals. Ideally, the ρ parameter would be tuneable to each individual subject. With current software, this is not possible. Hence, the chosen value of $\rho = 0.15$ is a compromise that allows reducing strong autocorrelations without creating many artificial autocorrelations where none are present.

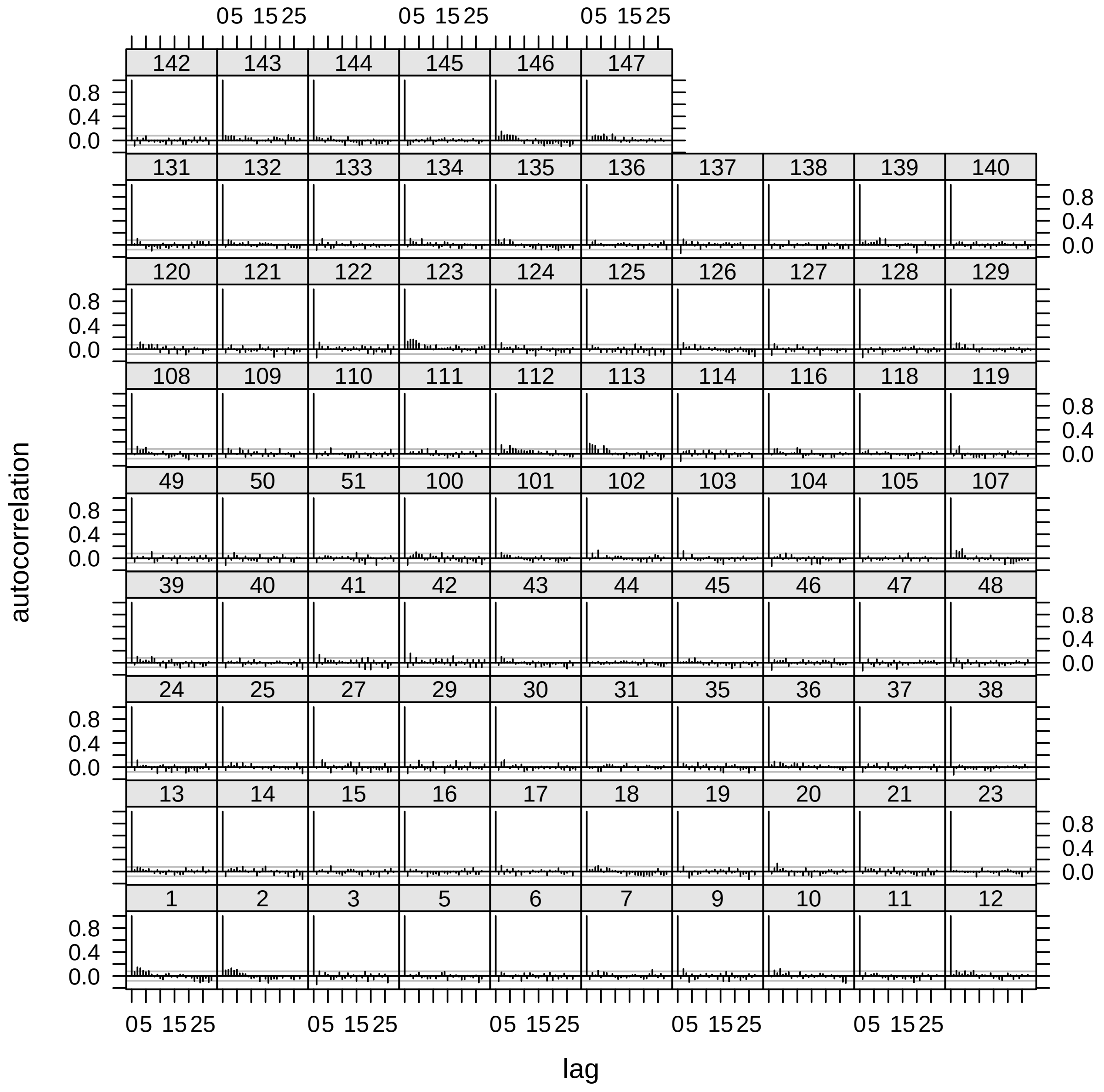

Figure 4: Autocorrelation functions for each subject-specific time series in the model with factor smooths and AR1 correction for the errors.

2.4 A nonlinear main effect of SOA

Next, we bring into the model the nonlinear effect of SOA with a thin plate regression spline (the default spline in `mgcv`).

```
dat.gam3 <- bam(lrt ~ sze * (spt + obj + grv) * orn +
                      s(subj, spt, bs="re") +
                      s(subj, grv, bs="re") +
                      s(subj, obj, bs="re") +
                      s(subj, orn, bs="re") +
                      s(subj, spt_orn, bs="re") +
                      s(SOA) +
                      s(trial, subj, bs="fs", m=1) ,
                      AR.start=dat$FirstTrial, rho=0.15,
                data=dat, method="ML")
summary(dat.gam3)
```

```
Family: gaussian
Link function: identity

Formula:
lrt ~ sze * (spt + obj + grv) * orn + s(subj, spt, bs = "re") +
    s(subj, grv, bs = "re") + s(subj, obj, bs = "re") + s(subj,
    orn, bs = "re") + s(subj, spt_orn, bs = "re") + s(SOA) +
    s(trial, subj, bs = "fs", m = 1)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.725232   0.017774  322.12  < 2e-16
sze          0.180390   0.035547    5.07  3.9e-07
spt          0.072801   0.007844    9.28  < 2e-16
obj          0.041138   0.004091   10.05  < 2e-16
grv         -0.000518   0.004895   -0.11  0.91568
orn          0.019187   0.009004    2.13  0.03309
sze:spt      0.048046   0.015688    3.06  0.00220
sze:obj     -0.008696   0.008183   -1.06  0.28793
sze:grv     -0.036482   0.009790   -3.73  0.00019
sze:orn      0.047304   0.018007    2.63  0.00862
spt:orn      0.021357   0.006387    3.34  0.00083
obj:orn      0.008053   0.006836    1.18  0.23878
grv:orn      0.007746   0.006862    1.13  0.25893
sze:spt:orn -0.009870   0.012774   -0.77  0.43973
sze:obj:orn -0.007979   0.013672   -0.58  0.55948
sze:grv:orn -0.048638   0.013724   -3.54  0.00039

Approximate significance of smooth terms:
                   edf Ref.df      F p-value
s(subj,spt)      77.69  84.00  31.93 < 2e-16
s(subj,grv)      46.82  84.00   2.11 1.6e-10
s(subj,obj)      28.74  84.00   1.28 0.00095
s(subj,orn)      42.28  84.00   1.08 < 2e-16
s(subj,spt_orn)  47.29  84.00   1.34 4.8e-11
s(SOA)            5.48   6.62 104.42 < 2e-16
s(trial,subj)   606.39 772.00 676.96 < 2e-16
```

```
R-sq.(adj) =   0.53   Deviance explained = 53.8%
-ML = -15091  Scale est. = 0.032687   n = 53765
```

The final model in Figure 1 in the paper is based on this model. The next code snippet recreates this figure.

```
# randomly select 5 subjects
set.seed(314)
Events <- sample(levels(dat$subj))[1:5]
# collect acf data for the three models
dfr1 = acf_resid(dat.gam3, split_pred=list(subj=dat$subj),
    cond=list(subj=Events),
    plot=FALSE, return_all=TRUE)$dataframe
dfr2= acf_resid(dat.gam1, split_pred=list(subj=dat$subj),
    cond=list(subj=Events),
    plot=FALSE, return_all=TRUE)$dataframe
dfr3 = acf_resid(dat.gam0, split_pred=list(subj=dat$subj),
    cond=list(subj=Events),
    plot=FALSE, return_all=TRUE)$dataframe
# combine and add column specifying models
Dfr <- rbind(dfr1, dfr2, dfr3)
Dfr$model <- rep(c("final model","model with fs","initial model"),
  c(nrow(dfr1), nrow(dfr2), nrow(dfr3)))
# beautify event names
levels(Dfr$event) = c("time series 1",
                      "time series 2",
                      "time series 3",
                      "time series 4",
                      "time series 5")
# order models for xyplot
Dfr$model = ordered(Dfr$model,
                    c("final model",
                      "model with fs",
                      "initial model"))
# conf. intervals for xyplot
civec = Dfr[Dfr$lag==0,]$ci
xyplot(acf ~ lag | event + model, type = "h", data = Dfr, col.line = "black",
  panel = function(...) {
      panel.abline(h = civec[panel.number()], col.line = "grey")
      panel.abline(h = -civec[panel.number()], col.line = "grey")
      panel.abline(h = 0, col.line = "black")
      panel.xyplot(...)
  },
  strip = strip.custom(bg = "grey90"),
  par.strip.text = list(cex = 0.8),
  xlab="lag", ylab="autocorrelation")
```

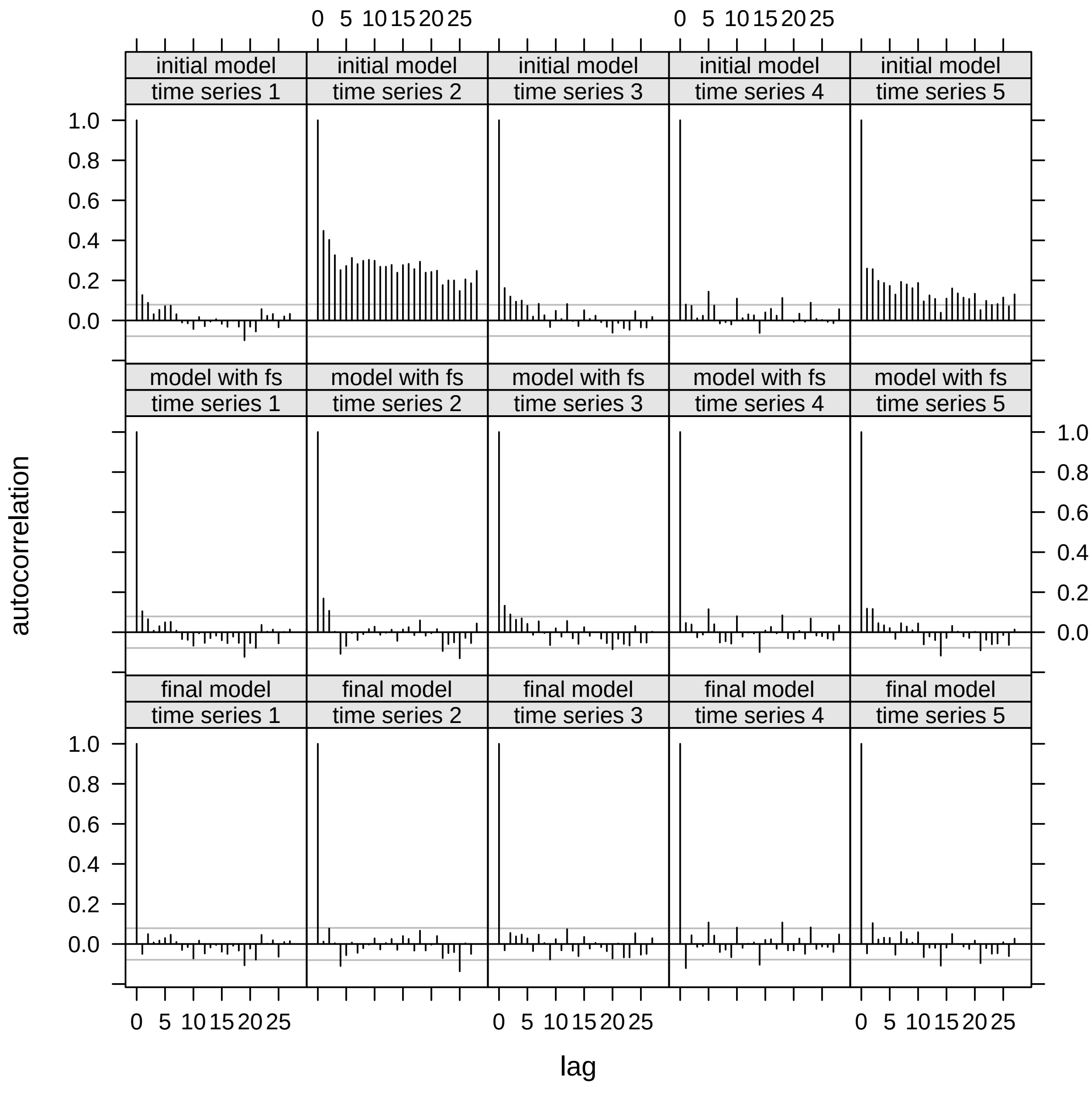

Figure 5: Figure 1 in the paper: ACF for 5 subjects for three models.

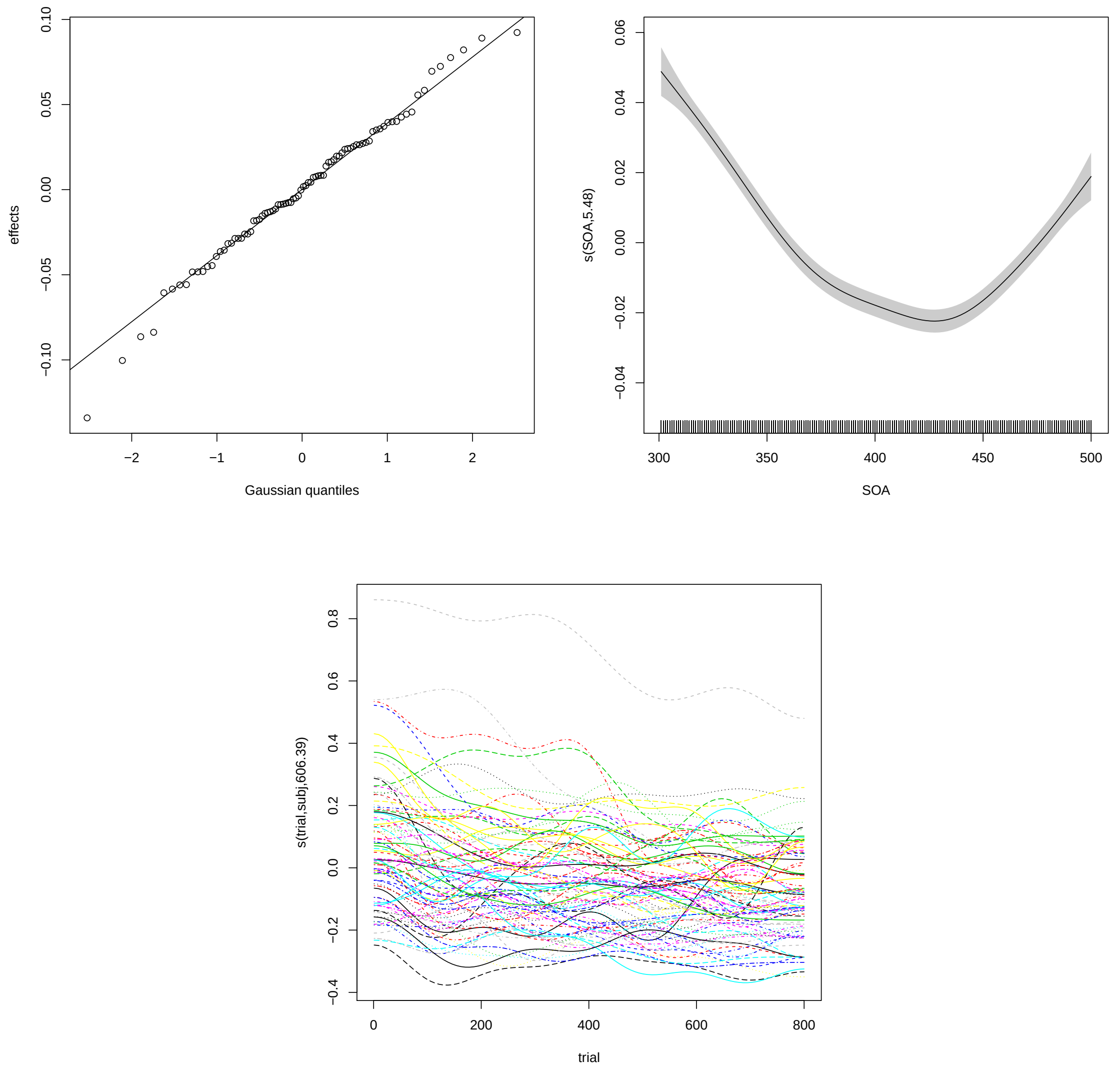

Figure 6: Smooth terms produced with plot.gam (mgcv).

The nonlinear effects in this final model (`dat.gam3`) can be visualized in various ways. Using the plot functionality of `mgcv`, we can select smooth terms by their row number in the smooths table of the summary. For plots of `s(subj,orn)`, `s(SOA)` and `s(trial,subj)` we proceed as follows (see Figure 6):

```
plot(dat.gam3, select=4, main=" ")
plot(dat.gam3, select=6, scheme=1, ylim=c(-0.05, 0.06))
plot(dat.gam3, select=7)
```

For random contrasts, a quantile-quantile plot for the BLUPs is shown, for the covariate, a smooth with 95% confidence intervals, and for the factor smooths, the partial effects for each subject.

The plot for the factor smooths is overcrowded. We inspect the individual subjects' partial effects for trial with a trellis graph.

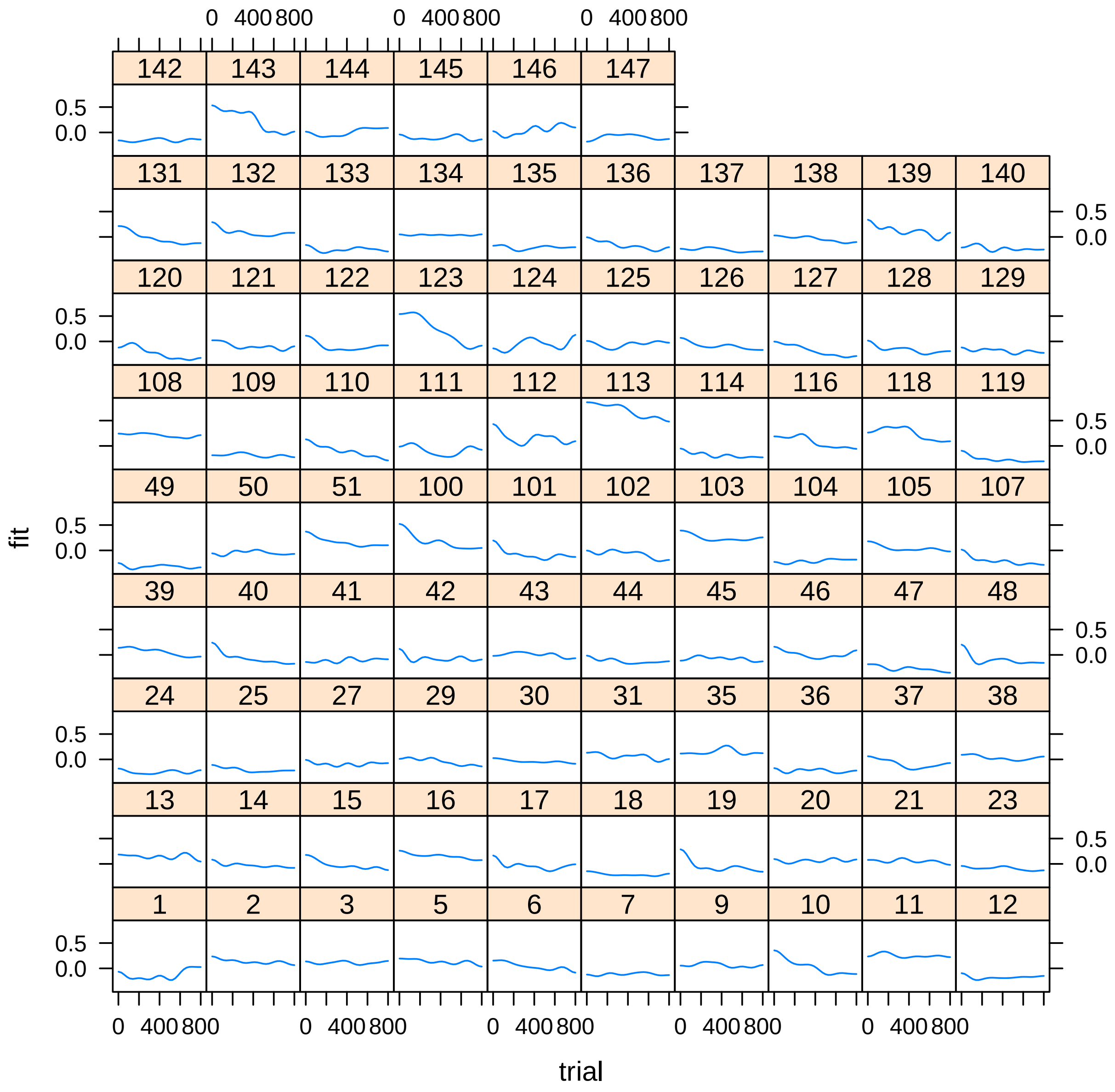

Figure 7: Factor smooths for subjects.

```
xyplot(fit~trial|subj, data=get_modelterm(dat.gam3, select=7, as.data.frame=TRUE), type="l")
```

The left panel of Figure 2 in the paper is obtained by selecting a subset of these subjects that illustrate the range of nonlinear patterns.

```
subjects = c("1", "124", "19", "46", "118", "146", "143", "108", "123")
pp = get_modelterm(dat.gam3, select=7,
  cond=list(subj=subjects),
  as.data.frame=TRUE)
xyplot(fit~trial|subj, data=pp, type="l")
```

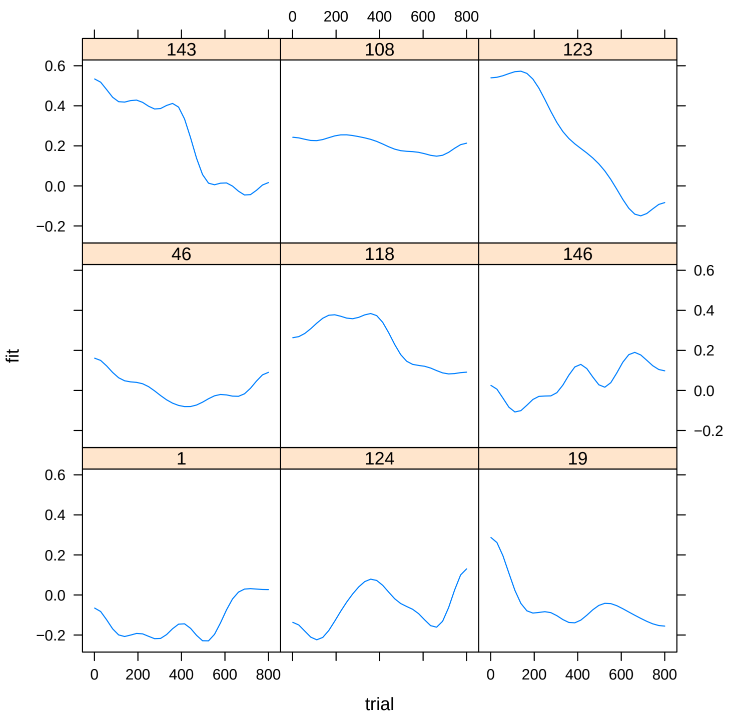

Figure 8: Factor smooths for selected subjects (Figure 2 in the paper).

It is noteworthy that subject 123 has the largest autocorrelations in Figure 2 and displays the largest changes in response speed in the course of the experiment. This illustrates how slow changes in response behavior as an experiment proceeds may induce marked non-independence in the response.

2.5 Model comparisons

To assess the importance of the factor smooth, and the wiggliness it captures, we compare the GAMM with an LMM that incorporates by-subject random slopes for trial.

```
dat.lmer = lmer(lrt ~ sze * (spt + obj + grv) * orn +
                      (spt + grv | subj) +
                      (0 + obj | subj) +
                      (0 + orn | subj) +
                      (0 + spt_orn | subj) +
                      (0 + trial|subj),
                data=dat, REML=FALSE)
```

A blunt way of comparing goodness of fit is to compare the proportion of variance explained by all predictors jointly, both random and fixed.

```
Rsquareds = c(
  cor(fitted(kkl4), dat$lrt)^2,
  cor(fitted(dat.gam0), dat$lrt)^2,
  cor(fitted(dat.gam1), dat$lrt)^2,
  cor(fitted(dat.lmer), dat$lrt)^2,
  cor(fitted(dat.gam2), dat$lrt)^2,
```

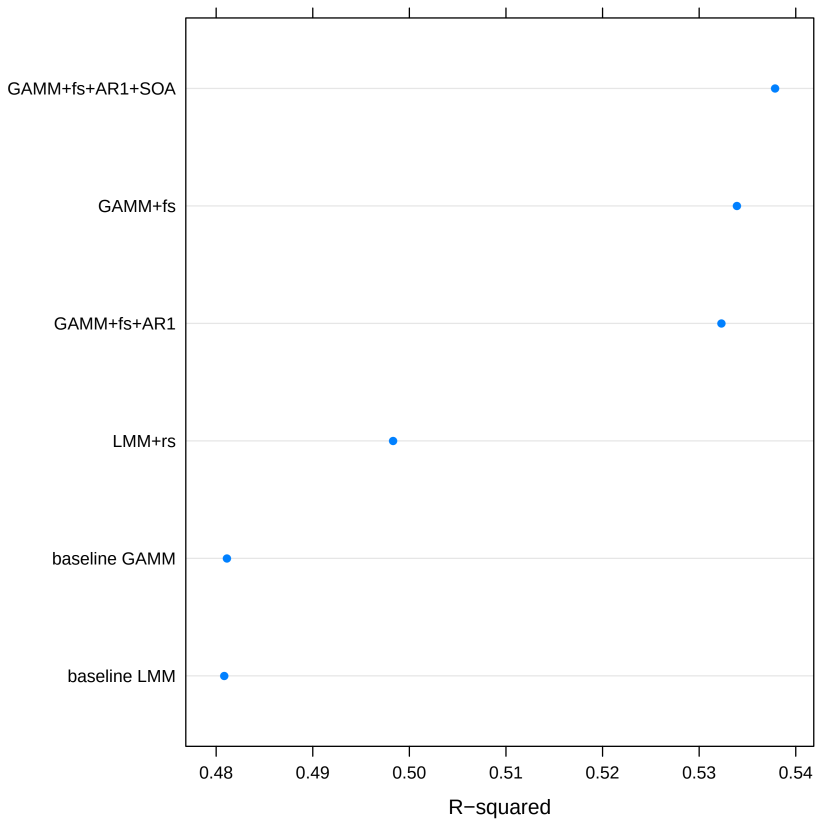

Figure 9: Overall R-squared for different models for the KKL dataset.

```
  cor(fitted(dat.gam3), dat$lrt)^2)
names(Rsquareds) = c("baseline LMM", "baseline GAMM", "GAMM+fs", "LMM+rs", "GAMM+fs+AR1",
                     "GAMM+fs+AR1+SOA")
dotplot(sort(Rsquareds), xlab="R-squared")
```

Figure 9 indicates that the factor smooths are well supported. Note that the inclusion of an AR1 process for the errors (with $\rho = 0.15$) makes the model a bit more conservative: the data can be predicted less well because of the autocorrelative process in the errors.

Table 1 in the paper, which compares the initial model with the final model, is reproduced as follows.

```
load("models/kkl4.rda")
# table comparing coefficients and t values for the two models
tab = cbind(summary(kkl4)$coefficients[,1],
            dat.gam0.smry$p.table[,1],
            dat.gam3.smry$p.table[,1],
            summary(kkl4)$coefficients[,3],
            dat.gam0.smry$p.table[,3],
            dat.gam3.smry$p.table[,3])
colnames(tab) = c("beta initial lmer",
                  "beta initial gamm",
                  "beta final (gamm)",
                  "t 0 (lmer)",
                  "t 0 (gamm)",
                  "t 3 (gamm)")
tab = round(tab, 3)
```

```
tab    # formatted as Table 1
```

	beta initial lmer	beta initial gamm	beta final (gamm)	t 0 (lmer)	t 0 (gamm)	t 3 (gamm)
(Intercept)	5.691	5.691	5.725	327.683	329.496	322.120
sze	0.184	0.184	0.180	5.302	5.331	5.075
spt	0.074	0.074	0.073	9.824	9.663	9.281
obj	0.041	0.041	0.041	9.360	9.091	10.055
grv	-0.002	-0.002	-0.001	-0.323	-0.286	-0.106
orn	0.041	0.041	0.019	3.916	3.917	2.131
sze:spt	0.049	0.049	0.048	3.221	3.169	3.063
sze:obj	-0.011	-0.011	-0.009	-1.218	-1.189	-1.063
sze:grv	-0.036	-0.036	-0.036	-3.445	-3.390	-3.727
sze:orn	0.017	0.017	0.047	0.792	0.791	2.627
spt:orn	0.020	0.020	0.021	3.025	3.022	3.344
obj:orn	0.009	0.009	0.008	1.263	1.256	1.178
grv:orn	0.011	0.011	0.008	1.519	1.499	1.129
sze:spt:orn	-0.013	-0.013	-0.010	-0.952	-0.948	-0.773
sze:obj:orn	-0.002	-0.002	-0.008	-0.140	-0.133	-0.584
sze:grv:orn	-0.044	-0.044	-0.049	-2.989	-2.954	-3.544

Table 1: Estimates of the fixed-effects coefficients and associated t-values for model 0 (without correction for autocorrelation) fitted with lmer and with bam and model 3 (with full correction for autocorrelation), fitted with bam.

This table indicates changes in the fixed-effect estimates for terms involving `orn`, the orientation of the picture presented to the participants.

A more precise way of clarifying what changes between the initial and final model is to compare the importance of the different terms in the model. We assess term importance by leaving it out of the model specification, and assessing the decrease in goodness of fit by means of the change for the worse in the ML score. We first consider the GAMM.

```
fmla = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
              )
# leave out spt from random effects
fmla1 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out grv from random effects
fmla2 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out obj from random effects
fmla3 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out orn from random effects
fmla4 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out spt_orn from random effects
fmla5 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out factor smooth from random effects, we have to put random intercepts back in
fmla6 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(subj, bs="re") +
                     s(SOA),
               )
# leave out SOA
fmla7 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out sze
fmla8 = formula(lrt ~ (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out orn and all associated random effect terms
fmla9 = formula(lrt ~ sze * (spt + obj + grv) +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
               )
# leave out tar (spt, obj, grv) and all associated random effect terms

fmla10 = formula(lrt ~ sze * orn +
                     s(subj, orn, bs="re") +
                     s(SOA) +
                     s(trial, subj, bs="fs", m=1)
                )

formulae = list(fmla1, fmla2, fmla3, fmla4, fmla5,
                fmla6, fmla7, fmla8, fmla9, fmla10)
# fixed effects in caps, variance components in lower case
# trial represents the factor smooth
names(formulae) = c("spt", "grv", "obj", "orn", "spt_orn",
                    "trial", "SOA", "SZE", "ORN", "TAR")
mls = as.numeric(dat.gam3$gcv.ubre)
# the next loop collects fREML scores
# this takes a couple of hours
for (i in 1:length(formulae)) {
  m = bam(formulae[[i]],
          AR.start=dat$FirstTrial, rho=0.15,
          data=dat, method="ML")
  mls = c(mls, as.numeric(m$gcv.ubre))
}
names(mls) = c("baseline", names(formulae))
```

Next, we consider the baseline LMM, for which we always retain the by-subject random intercepts.

```
fmla = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(subj, bs="re")
              )
# leave out spt from random effects
fmla1 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(subj, bs="re")
               )

# leave out grv from random effects
fmla2 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(subj, bs="re")
               )
# leave out obj from random effects
fmla3 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(subj, bs="re")
               )
# leave out orn from random effects
fmla4 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(subj, bs="re")
               )
# leave out spt_orn from random effects
fmla5 = formula(lrt ~ sze * (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, bs="re")
               )
# leave out sze
fmla6 = formula(lrt ~ (spt + obj + grv) * orn +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, orn, bs="re") +
                     s(subj, spt_orn, bs="re") +
                     s(subj, bs="re")
               )
# leave out orn and all associated random effect terms
fmla7 = formula(lrt ~ sze * (spt + obj + grv) +
                     s(subj, spt, bs="re") +
                     s(subj, grv, bs="re") +
                     s(subj, obj, bs="re") +
                     s(subj, bs="re")
               )
# leave out tar (spt, obj, grv) and all associated random effect terms
fmla8 = formula(lrt ~ sze * orn +
                     s(subj, orn, bs="re") +
                     s(subj, bs="re")
                )

formulaeLMM = list(fmla1, fmla2, fmla3, fmla4, fmla5,
                fmla6, fmla7, fmla8)
# fixed effects in caps, variance components in lower case
names(formulaeLMM) = c("spt", "grv", "obj", "orn", "spt_orn", "SZE", "ORN", "TAR")
mlsLMM = as.numeric(dat.gam0$gcv.ubre)
# collect fREML statistics
# the next loop takes about an hour
for (i in 1:length(formulaeLMM)) {
  cat(i, " ")
  m = bam(formulaeLMM[[i]], data=dat, method="ML")
  mlsLMM = c(mlsLMM, as.numeric(m$gcv.ubre))
}
names(mlsLMM) = c("baseline", names(formulaeLMM))
cat("\n")
```

```
# calculate changes in fREML scores compared to the full model
m = rep(mls[1], length(mls)-1)
m = m - mls[2:length(mls)]
names(m)=c("spt", "grv", "obj", "orn", "spt_orn", "trial", "SOA", "SIZE", "ORN", "TAR")
m = -m
m

     spt      grv      obj      orn  spt_orn    trial      SOA     SIZE
 302.646   16.689    3.999   13.296   19.472 1317.720  335.112   32.765
     ORN      TAR
  74.899 2768.349

mLMM = rep(mlsLMM[1], length(mlsLMM)-1)
mLMM = mLMM - mlsLMM[2:length(mlsLMM)]
names(mLMM)=c("spt", "grv", "obj", "orn", "spt_orn", "SIZE", "ORN", "TAR")
mLMM = -mLMM
mLMM

     spt      grv      obj      orn  spt_orn     SIZE      ORN      TAR
 239.285   18.562    5.285  758.659   15.135   28.243 1676.725 2445.818
```

```
m2 = c(mLMM, 0, 0)
names(m2)[9:10] = c("TRIAL", "SOA")
m2 = m2[order(names(m2))]
m = m[order(names(m))]
# bring information together in data frame
dfr = data.frame(decrease = c(m, m2),
                 term = rep(names(m),2),
                 model = c(rep("final", length(m)), rep("initial", length(m))))
dfr$term = gsub("_", ":", dfr$term)
dfr$val = rep(m,2)
dfrA = dfr[dfr$model=="final",]
dfrB = dfr[dfr$model!="final",]
dfrA = dfrA[order(dfrA$val),]
dfrB = dfrB[order(dfrB$val),]
dfr2 = rbind(dfrA, dfrB)
dfr2$term = factor(dfr2$term)
dfr2$F = factor(rep(strsplit("abcdefghij", "")[[1]],2))
dfr2$F = factor(rep(1:10, 2), labels=as.character(dfr2$term[1:10]))
```

```
dotplot(F~decrease, groups=model, data=dfr2,
  pch=c(19, 2), col="black", cex=c(1, 1.3),
  xlab="decrease in goodness of fit when a term is dropped",
  key=list(text=list(c("final model", "initial model")),
           points=list(pch=c(19,2), cex=c(1, 1.3)))
  )
```

Figure 10 indicates that once the autocorrelations are dealt with, a model is obtained that assigns slightly greater importance to `TAR` and substantially less importance to `ORN` (and `orn`). Apparently, changes in orientation of the picture presented are more prone to give rise to attentional shifts that linger on to the next trial.

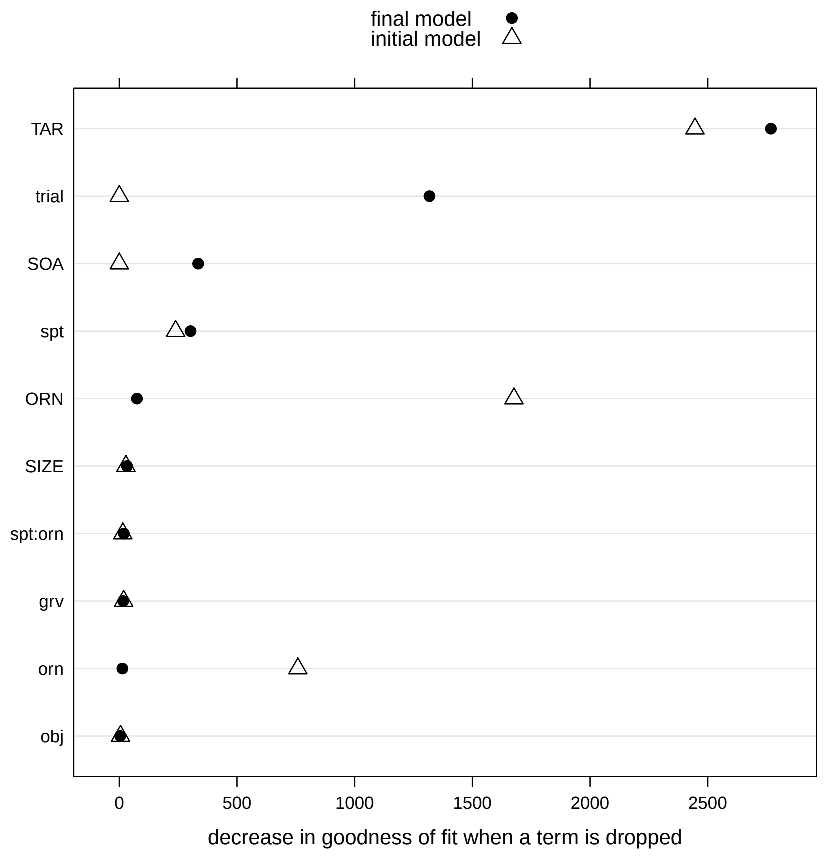

Figure 10: Variable importance for the LMM and GAMM.

3 Parsimony in regression

Baayen and Milin (2010) reported a self-paced reading experiment with overspecified random effects structure. The response variable in this study is log-transformed reaction time. Of the many predictors considered by these authors, we select three to illustrate the overspecification problem: word frequency, participant age in years, and participant's reaction times for a multiple-choice question probing their reading habits. The latter variable was included to probe between-subject variance and to capture variance that would otherwise have to be accounted for through by-participant random intercepts. All three predictors were log-transformed and scaled.

```
load("data/poems.rda")
# abbreviate names
poems$Fre = scale(poems$LogWordFormFrequency)
poems$Mul = scale(poems$LogMultipleChoiceRT)
poems$Age = scale(log(poems$Age))
poems$Lrt = poems$LogReadingTime
# order by subject and trial
poems = poems[order(poems$Subject, poems$Trial),]
# scale Trial
poems$TrialSc = as.numeric(scale(poems$Trial))
# add logical for starting point of time series
pos = tapply(poems$Trial, poems$Subject, min)
poems$MinTrial = pos[as.character(poems$Subject)]
poems$Start = poems$MinTrial==poems$Trial
```

A total of 275996 data points are available, from 326 subjects, for 2315 appearing across 87 modern Dutch poems. Words are partially nested under poems. Any given subject read only a subset of poems.

3.1 Linear mixed models for the poems data set

We first fit a simple model with the three predictors and the three random-effect factors,

```
poems.lmer0 = lmer(Lrt ~ Fre + Mul + Age +
  (1|Poem)+ (1|Subject)+ (1|Word),
  data=poems, REML=FALSE)
```

and inspect the summary.

```
print(summary(poems.lmer0), corr=FALSE)

Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: Lrt ~ Fre + Mul + Age + (1 | Poem) + (1 | Subject) + (1 | Word)
   Data: poems

     AIC      BIC   logLik deviance df.resid
  174760   174845   -87372   174744   275988

Scaled residuals:
   Min     1Q Median     3Q    Max
-5.977 -0.616 -0.107  0.493  5.193

Random effects:
 Groups   Name        Variance Std.Dev.
 Word     (Intercept) 0.01435  0.1198
 Subject  (Intercept) 0.04771  0.2184
 Poem     (Intercept) 0.00275  0.0524
 Residual             0.10750  0.3279
Number of obs: 275996, groups:  Word, 2315; Subject, 326; Poem, 87

Fixed effects:
            Estimate Std. Error t value
(Intercept)   6.0403     0.0142     426
Fre          -0.0580     0.0035     -17
Mul           0.0945     0.0122       8
Age           0.0445     0.0121       4
```

Next, we fit a series of models that add by-subject slopes for `Fre` and by-word slopes for `Mul`, with and without correlation parameters.

```
poems.lmer1 = lmer(Lrt ~ Fre + Mul + Age +
  (1|Poem) + (1+Fre|Subject) + (1|Word),
  data=poems, REML=FALSE)
```

```
poems.lmer2 = lmer(Lrt ~ Fre + Mul + Age +
  (1|Poem) + (1+Fre|Subject) + (1+Mul|Word),
  data=poems, REML=FALSE)
```

```
poems.lmer3 = lmer(Lrt ~ Fre + Mul + Age +
  (1|Poem) + (1|Subject) + (0+Fre|Subject) + (1+Mul|Word),
  data=poems, REML=FALSE)
```

```
poems.lmer4 = lmer(Lrt ~ Fre + Mul + Age +
  (1|Poem) + (1+Fre|Subject) + (1|Word) + (0+Mul|Word),
  data=poems, REML=FALSE)
```

Likelihood ratio tests

```
# testing for random slopes and correlations, first subject, then also word
round(as.data.frame(anova(poems.lmer0, poems.lmer1, poems.lmer2)),4)

            Df    AIC    BIC logLik deviance   Chisq Chi Df Pr(>Chisq)
poems.lmer0  8 174761 174845 -87372   174745      NA     NA         NA
poems.lmer1 10 172376 172481 -86178   172356 2388.35      2          0
poems.lmer2 12 172302 172429 -86139   172278   77.83      2          0

# testing for by-subject correlation parameter for Fre
round(as.data.frame(anova(poems.lmer3, poems.lmer2)),4)

            Df    AIC    BIC logLik deviance Chisq Chi Df Pr(>Chisq)
poems.lmer3 11 172461 172577 -86219   172439    NA     NA         NA
poems.lmer2 12 172302 172429 -86139   172278 160.4      1          0
```

```
# testing for by-word correlation parameter for Mul
round(as.data.frame(anova(poems.lmer4, poems.lmer2)),4)

            Df    AIC    BIC logLik deviance Chisq Chi Df Pr(>Chisq)
poems.lmer4 11 172372 172488 -86175   172350    NA     NA         NA
poems.lmer2 12 172302 172429 -86139   172278  71.5      1          0
```

support the maximal model (`poems.lmer2`), which we summarize:

```
print(summary(poems.lmer2), corr=FALSE)

Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: Lrt ~ Fre + Mul + Age + (1 | Poem) + (1 + Fre | Subject) + (1 +
    Mul | Word)
   Data: poems

     AIC      BIC   logLik deviance df.resid
  172302   172429   -86139   172278   275984

Scaled residuals:
   Min     1Q Median     3Q    Max
-6.061 -0.610 -0.108  0.488  5.130

Random effects:
 Groups   Name        Variance Std.Dev. Corr
 Word     (Intercept) 0.014624 0.1209
          Mul         0.000114 0.0107   0.91
 Subject  (Intercept) 0.049583 0.2227
          Fre         0.001253 0.0354   -0.68
 Poem     (Intercept) 0.002776 0.0527
 Residual             0.106265 0.3260
Number of obs: 275996, groups:  Word, 2315; Subject, 326; Poem, 87

Fixed effects:
            Estimate Std. Error t value
(Intercept)  6.04262    0.01438     420
Fre         -0.05719    0.00401     -14
Mul          0.05511    0.00977       6
Age          0.05786    0.00944       6
```

Both correlation parameters are interpretable. Words that take more time to respond to may also be words that take especially long to respond to for participants who are slow deciders in a multiple choice situation ($r = 0.91$). Subjects that respond very quickly show little of a frequency effect, suggesting a trade-off between signal-driven responding and responding on the basis of long-term lexical priors ($r = -0.68$).

However, the large value of the correlation parameter for `Word` is an informal indicator of overparameterization, pointing to collinearity in the by-word random effects structure. Given a word's intercept, one has a very good estimate of that word's slope, and vice versa, see Figure 11. In other words, there is not much evidence in the data that would allow separation of the two sources of by-word variation.

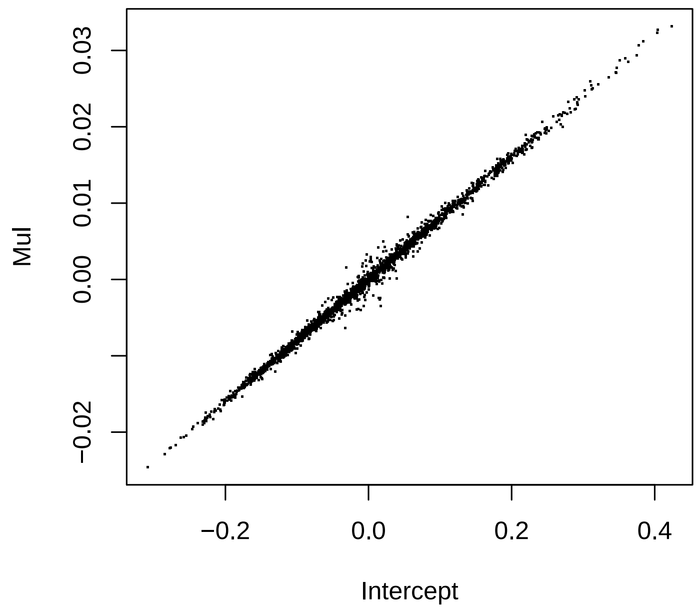

Figure 11: By-word BLUPS in the maximal model for the poems dataset.

```r
plot(ranef(poems.lmer2)$Word, xlab="Intercept", pch=".")
```

This strong collinearity becomes apparent as well when we subject the random effects structure to a singular value decomposition, and inspect the proportions of squared singular values of the random-effects variance-covariance estimates (Figure 12). It is clear that the random intercepts are important. Whether the tiny contributions of the random slopes are worth including in the model requires further reflection.
Inspection of the variance explained compared to a baseline model with random intercepts for the three factors only

```r
poems.lmer00 =  lmer(Lrt ~ 1 +
  (1|Poem) + (1|Subject) + (1|Word),
  data=poems, REML=FALSE)
```

indicates that, as expected given the preceding results, a model that has only by-subject slopes captures most of the explainable variance, with little additional variance to be captured with the help of by-word slopes for `Mul`.

```r
# full model compared to intercept-only model
cor(fitted(poems.lmer2),  poems$Lrt)^2 -
  cor(fitted(poems.lmer00), poems$Lrt)^2

[1] 0.007656

# model with by-subject slopes but no by-word slopes compared to intercept-only model
cor(fitted(poems.lmer1),  poems$Lrt)^2 -
  cor(fitted(poems.lmer00), poems$Lrt)^2
```

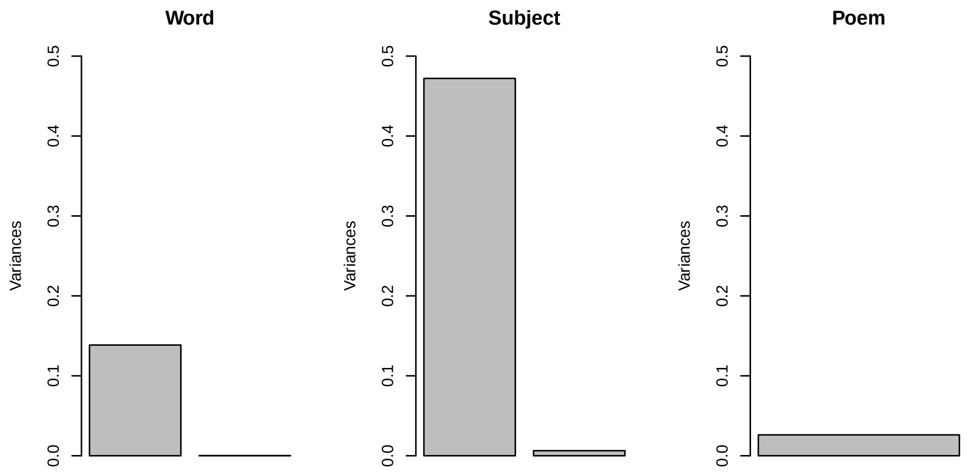

Figure 12: PCA of random-effects variance-covariance estimates.

```
[1] 0.007266
```

As is often the case in studies of lexical processing, almost all of the variance is accounted for by random intercepts for random-effect factors such as subject and item.
Caution with respect to random slopes for `Word` is advisable given that words are partially crossed with subject and poem. Some words are specific to a given poem, others are shared by many poems. Furthermore, not every participant read every poem. In addition, the distribution of the words is not balanced, but Zipfian, as participants were reading real text. As a consequence, the data are relatively sparse, and not optimal for estimating interactions of subject properties with word. Since removal of the by-word random slopes results in only a minor reduction in goodness of fit, a model without by-word random slopes is a well-motivated alternative to the maximal model.

Inspection of the residuals of the simplified linear mixed model reveals substantial autocorrelation.

```
acf(resid(poems.lmer1), main=" ")
```

A model with scaled trial as covariate and corresponding by-subject random slopes,

```
poems.trial.lmer = lmer(Lrt ~ Fre + Mul + Age + TrialSc +
  (1|Poem) + (1+Fre|Subject) + (0+TrialSc|Subject) + (1|Word),
  data=poems, REML=FALSE)
```

```
summary(poems.trial.lmer)

Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula:
Lrt ~ Fre + Mul + Age + TrialSc + (1 | Poem) + (1 + Fre | Subject) +
    (0 + TrialSc | Subject) + (1 | Word)
   Data: poems

     AIC      BIC   logLik deviance df.resid
```

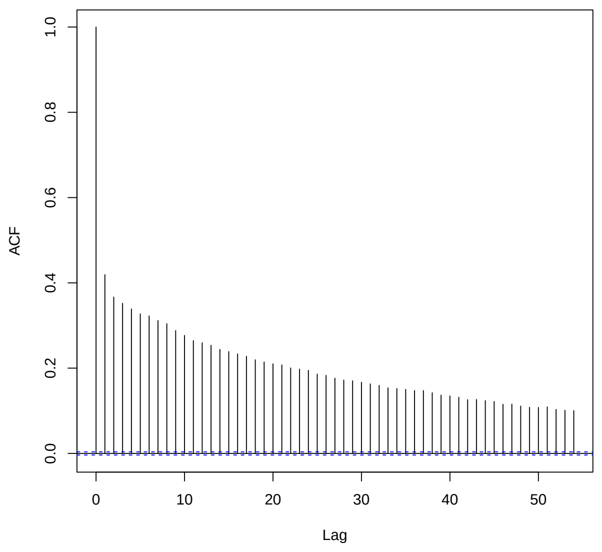

Figure 13: Autocorrelation function for simplified LMM for the poems data.

```
  140010   140136   -69993   139986   275984

Scaled residuals: 
   Min     1Q Median     3Q    Max 
-5.993 -0.605 -0.112  0.469  6.023 

Random effects:
 Groups    Name        Variance Std.Dev. Corr 
 Word      (Intercept) 0.01480  0.1216        
 Subject   TrialSc     0.00967  0.0983        
 Subject.1 (Intercept) 0.05021  0.2241        
           Fre         0.00128  0.0357   -0.65
 Poem      (Intercept) 0.00206  0.0454        
 Residual              0.09395  0.3065        
Number of obs: 275996, groups:  Word, 2315; Subject, 326; Poem, 87

Fixed effects:
            Estimate Std. Error t value
(Intercept)  6.03211    0.01417     426
Fre         -0.05979    0.00404     -15
Mul          0.05826    0.00982       6
Age          0.05984    0.00974       6
TrialSc     -0.07756    0.00550     -14

Correlation of Fixed Effects:
        (Intr) Fre    Mul    Age   
Fre     -0.036                     
Mul     -0.002  0.000              
Age     -0.001  0.000  0.004       
TrialSc  0.001  0.000  0.000 -0.001
```

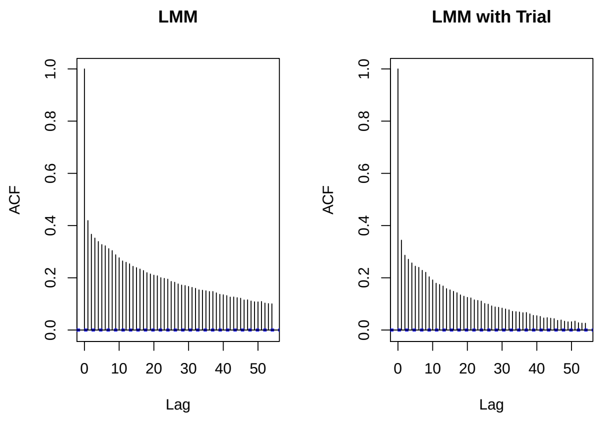

Figure 14: ACF for simplified LMM and the corresponding model with in addition random slopes for Trial

which improves considerably in terms of R-squared,

```
cor(fitted(poems.lmer1), poems$Lrt)^2

[1] 0.3932

cor(fitted(poems.trial.lmer), poems$Lrt)^2

[1] 0.4645
```

affords only a slight reduction in the autocorrelations of the residual errors (see Figure 14).

```
par(mfrow=c(1,2))
acf(resid(poems.lmer1), main="LMM")
acf(resid(poems.trial.lmer), main="LMM with Trial")
```

3.2 Generalized additive modeling of the poems dataset

Often, effects of trial are non-linear. We therefore relax the linearity constraint on trial and consider a generalized additive mixed model with by-subject factor smooths for trial. In addition, we model a nonlinear interaction of Frequency and Age with the help of a tensor product smooth (`te(Fre, Age)`). We return to this interaction below.

```
poems.trial.gamA = bam(Lrt~ te(Fre, Age) + Mul +
  s(TrialSc, Subject, bs="fs", m=1)+s(Subject, Fre, bs="re"),
  data=poems)
```

The autocorrelation is now reduced more substantially, as can be seen in the left panel of Figure 15. Inclusion of an AR1 autocorrelative process in the residuals with $\rho = 0.30$ further reduces the autocorrelations (right panel). (Baayen & Milin, 2010, used the response latency at the preceding trial to whiten the errors, but it is preferable to address the autocorrelations directly in the residuals.)

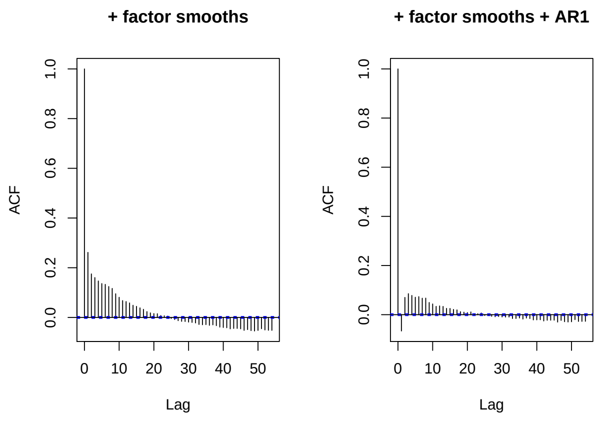

Figure 15: Autocorrelation functions for GAMMs with factor smooths and AR1 process for the errors.

```
poems.trial.gamB = bam(Lrt~ te(Fre, Age) + Mul +
   s(TrialSc, Subject, bs="fs", m=1)+s(Subject, Fre, bs="re"),
   rho=0.3, AR.start = poems$Start,
   data=poems)
```

The autocorrelation functions shown thus far for the poems data are imprecise, because they ignore the individual time series of the different subjects. This can give rise to artifacts. We therefore zoom in on these individual time series for the final model. These time series consist of the button presses of a given subject as she was reading through the set of poems assigned to her (Figures 16–18).

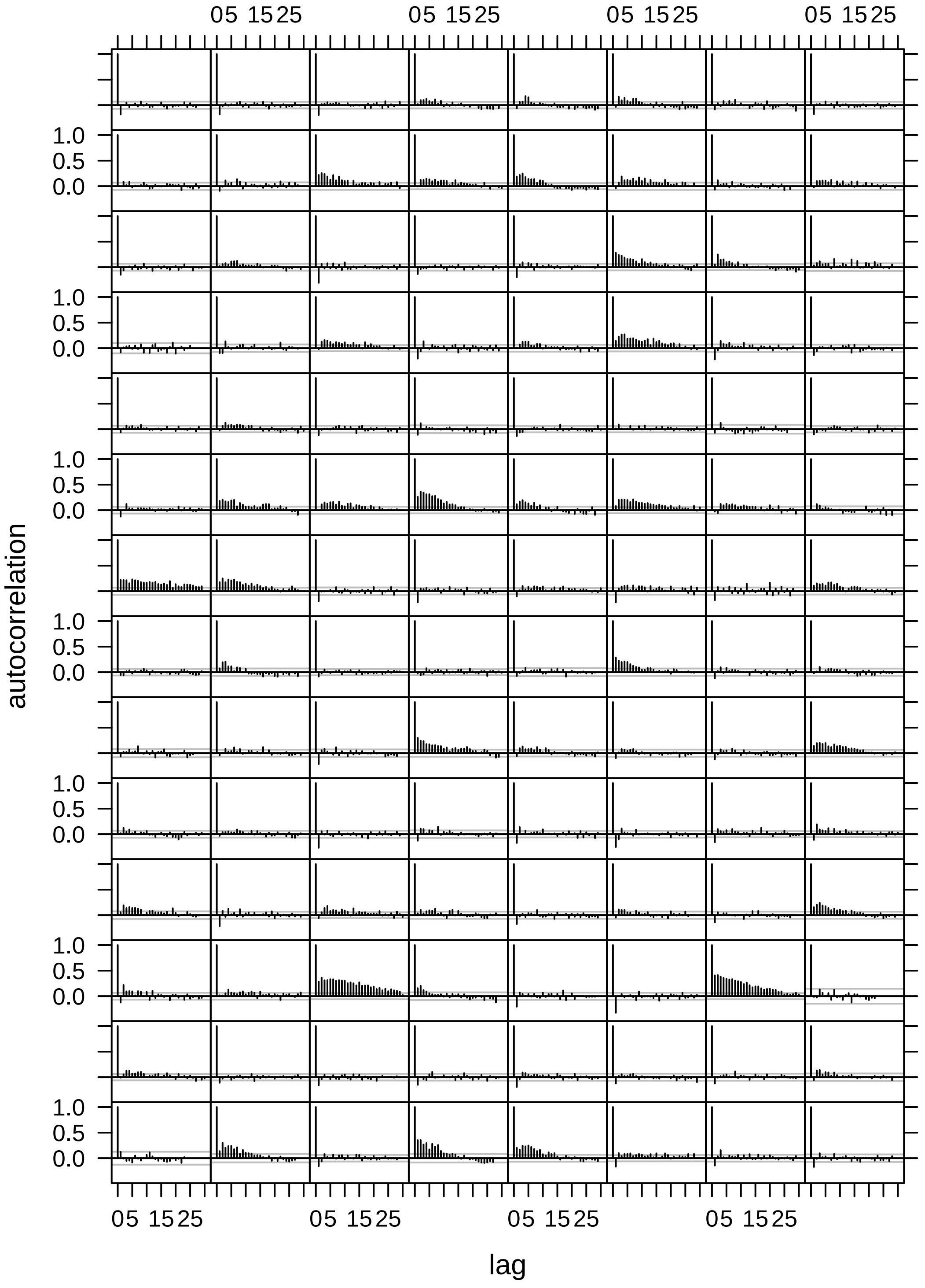

Figure 16: Autocorrelation functions for individual readers in the poems data.

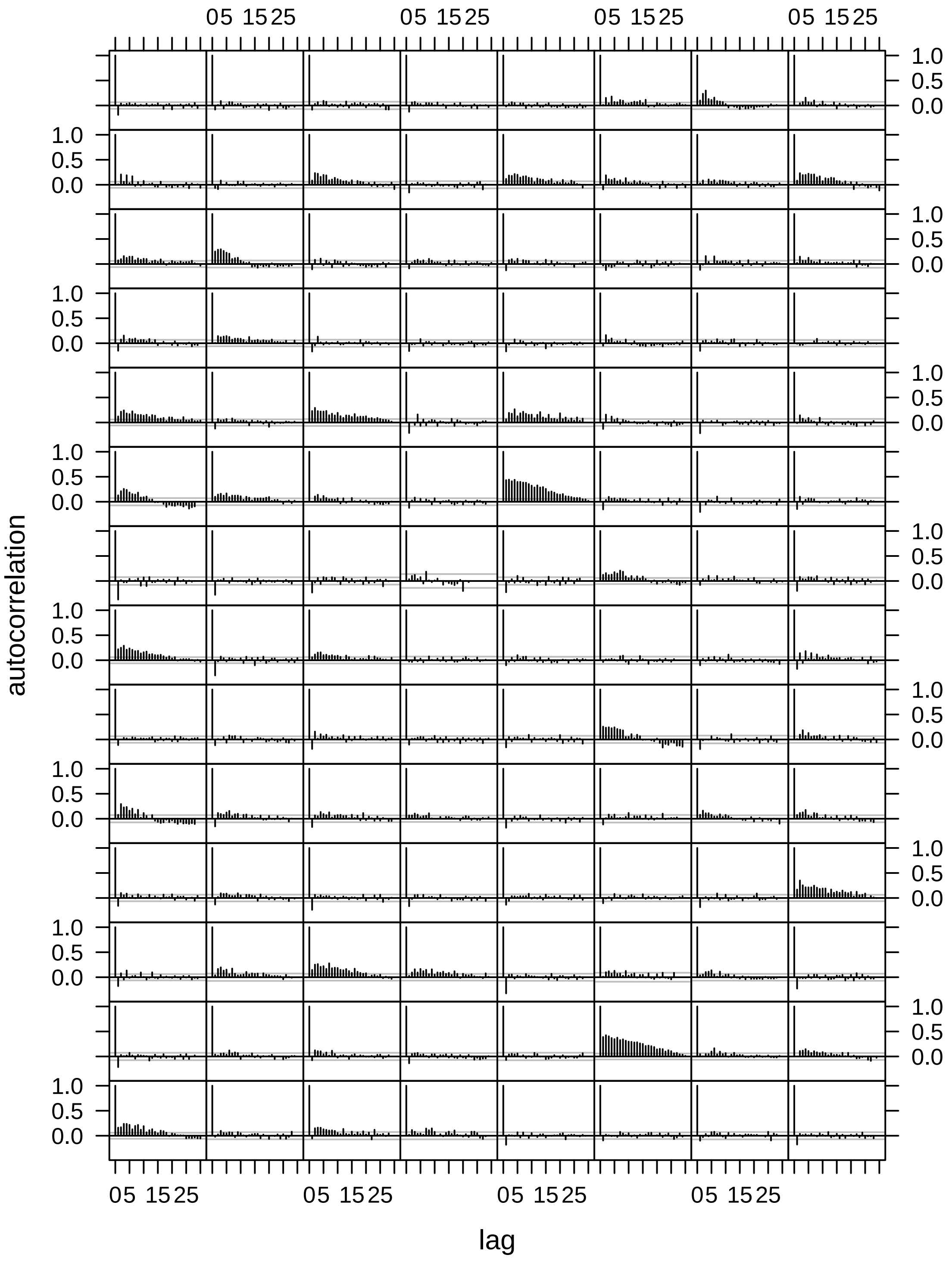

Figure 17: Autocorrelation functions for individual readers in the poems data.

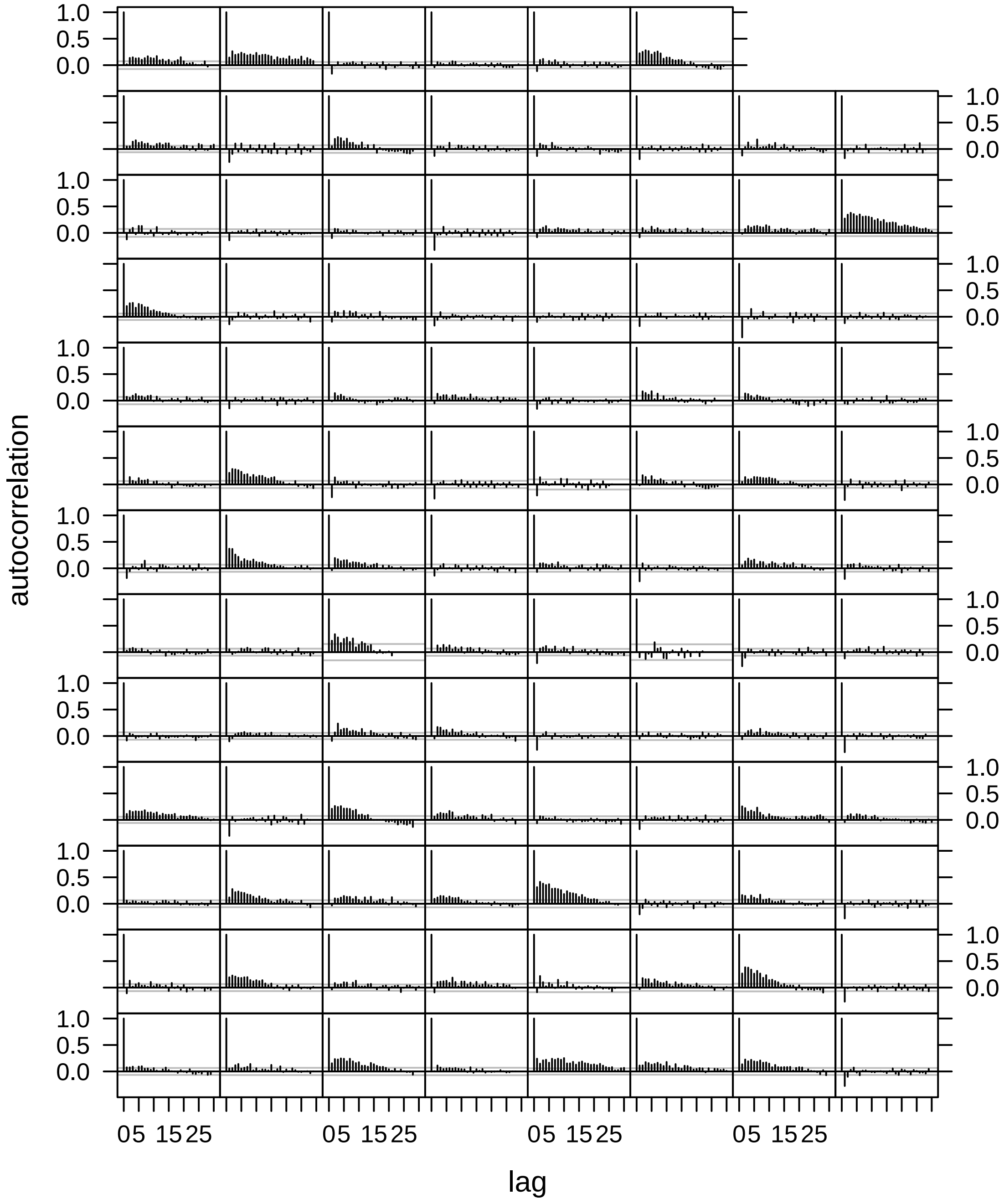

Figure 18: Autocorrelation functions for individual readers in the poems data.

For a majority of readers, the errors are appropriately whitened. Some readers, however, still show autocorrelations across many lags. As just a single autocorrelation parameter can be specified, that will be applied across all subjects, we again have to settle for a compromise such that artefactual induced (often negative) autocorrelations for many other subjects are avoided. Several strategies could be persued. If significance is crucial, subjects with strong autocorrelations could be removed, and the analysis repeated without them. The timeseries could be refined. Instead of taking all data from a given subject as a timeseries, one could define shorter series, one for each combination of subject and poem. However, models with large numbers of time series may become unestimable. The subjects with strong remaining autocorrelations are also of genuine interest by themselves. Are these subjects the ones who enjoy reading poetry? Or are these the subjects who read through the poems with little interest and enjoyment? Answers to questions such as these are beyond the scope of this vignette.

By taking nonlinearities into account, we have obtained a model that not only has substantially reduced autocorrelations in the errors, but that also provides a slightly tighter fit to the data.

```
# LMM without trial
cor(fitted(poems.lmer1), poems$Lrt)^2

[1] 0.3932

# LMM with trial and by-subject random slopes for trial
cor(fitted(poems.trial.lmer), poems$Lrt)^2

[1] 0.4645
```

```
# GAMM with by-subject factor smooths for trial and rho=0.30
cor(fitted(poems.trial.gamB), poems$Lrt)^2
```

```
[1] 0.4997
```

The final model (`poems.trial.gamB`) is described by the paramatric and smooth subtables from the summary.

```
poems.trial.gamB.smry = summary(poems.trial.gamB) # this takes a long time
poems.trial.gamB.smry$p.table
             Estimate Std. Error t value  Pr(>|t|)
(Intercept)  6.12576    0.01397 438.514 0.000e+00
Mul          0.08116    0.01402   5.788 7.118e-09
```

```
poems.trial.gamB.smry$s.table
                         edf   Ref.df     F    p-value
te(Fre,Age)            5.996    7.088 78.36 6.824e-115
s(TrialSc,Subject) 2275.558 2931.000 41.30  0.000e+00
s(Subject,Fre)      302.987  324.000 14.83  0.000e+00
```

3.3 Nonlinear interactions of covariates

As mentioned above, the final model incorporates a nonlinear interaction of Age by Frequency, modeled with a tensor product smooth. This interaction is supported by comparison with a model with main effects only,

```
poems.trial.gamC = bam(Lrt ~ Fre + Age + Mul +
   s(TrialSc, Subject, bs="fs", m=1) + s(Subject, Fre, bs="re"),
   rho=0.3, AR.start = poems$Start,
   data=poems)
```

```
compareML(poems.trial.gamC, poems.trial.gamB)
```

```
             Model     Score Edf Chisq    Df   p.value
1 poems.trial.gamC 50172.36   7
2 poems.trial.gamB 50163.23  10 9.137 3.000 3.863e-04
```

The partial effect of the interaction is shown in Figure 19, left panel.

```
plot(poems.trial.gamB,select=1, rug=FALSE, main=" ", lwd=1.6)  # Fig 19, left panel
x = unique(poems[,c("Fre", "Age")])
points(x, pch=".", col="gray")
```

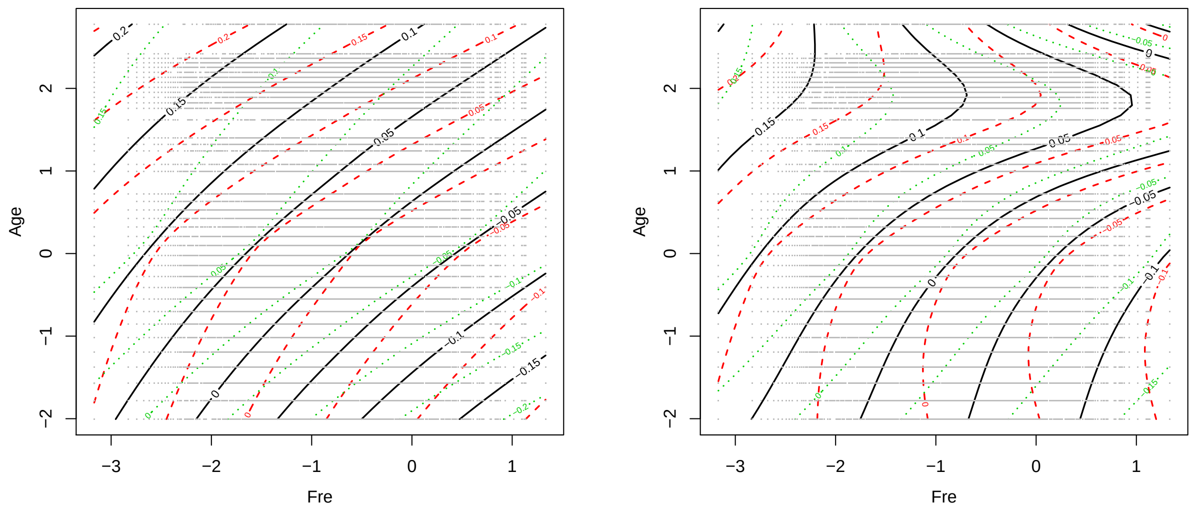

Figure 19: Tensor product smooth for the interaction of Age and Frequency in the poems data. Left: correct model with by-subject random slopes for frequency. Right: incorrect model without this variance component. Dotted lines represent 1 SE confidence regions, red dashed lines are 1 SE down from their contour line, and green dotted lines 1 SE up. Grey dots represent data points.

The age effect is slightly smaller for low-frequency words, and the frequency effect is slightly stronger for the younger participants. The interaction is mild, but of potential theoretical significance, as in our previous work on frequency and age (Ramscar et al., 2014, 2013) only aggregate data were considered in which subject-specific linear slopes for frequency where not partialed out. The right panel illustrates the consequences of not doing so: a much more irregular and less well interpretable surface is generated for the interaction. Returning to the left panel, if frequency is understood as a lexical prior, then older readers of poetry depend less on these priors, suggesting they get more out of the poetry.

In summary, Baayen & Milin (2010) proposed a model with several by-word and by-subject random slopes. Inclusion of these random slopes was motivated in part by the wish to provide stringent tests for the significance of main effects (cf. Barr et al., 2013), and in part by interest in individual differences. Upon closer inspection, the by-word random slopes turned out to contribute very little to the model fit, while at the same time suffering from data sparseness and collinearity. Here, a more parsimoneous model without the by-word random slopes seems justified. By contrast, we maintained the by-subject random slopes for frequency. This variance component contributed more substantially to the model fit, and furthermore turned out to be essential for a proper assessment of the interaction of age by frequency. Thus, we kept the model maximal within the boundaries set by what the data can support on the one hand, and by what makes sense theoretically on the other.

4 Concluding comments

The statistician George Box is famous for stating that all models are wrong, but some are more useful than others. The present model for the poems data is wrong in several ways. We have already seen that for some subjects, persistent autocorrelations are present in the residuals. Furthermore, there are many other variables that could have been brought into the analysis. Important for the present discussion is that it is quite possible that there is subject-specific variation that has not been accounted for, especially in relation to trial. For instance, the interaction of frequency and age might be modulated by how far a participant has progressed through the experiment. It might also vary by poem. Interactions of subject by poem by trial by frequency by age, however realistic, are beyond what can currently be modelled, and are probably also far beyond what we can integrate into our theories of language processing. Nevertheless, the present model may be useful as a window on a complex dataset, the modulation of frequency effects by age, subject-specific differences in the effect of frequency, and subject-specific variation in local coherence in reading.

References

Baayen, R. H. and Milin, P. (2010). Analyzing reaction times. *International Journal of Psychological Research*, 3:12–28.

Barr, D. J., Levy, R., Scheepers, C., and Tily, H. J. (2013). Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of Memory and Language*, 68(3):255–278.

Ramscar, M., Hendrix, P., Love, B., and Baayen, R. (2013). Learning is not decline: The mental lexicon as a window into cognition across the lifespan. *The Mental Lexicon*, 8:450–481.

Ramscar, M., Hendrix, P., Shaoul, C., Milin, P., and Baayen, R. (2014). Nonlinear dynamics of lifelong learning: the myth of cognitive decline. *Topics in Cognitive Science*, 6:5–42.