A Monte Carlo evidence

We carried out several simulation studies in order to evaluate the finite-sample performance of the proposed maximum composite likelihood estimators. In this appendix, we illustrate one of these studies. Similar results were obtained with other simulation settings.

We simulate 200 ordinal response longitudinal data sets with \( m = 200 \) subjects and \( n_i = 50 \) observations per subject, \( i = 1, \ldots, m \). Thus, the total number of observations is equal to 10,000. The ordinal response is simulated from a three categories MAOP model. Hence, the response is obtained by categorizing a hidden Gaussian process

\[
Y_{ij} = y_{ij} \Leftrightarrow \alpha_{y_{ij} - 1} < Y_{ij}^{*} \leq \alpha_{y_{ij}},
\]

with \(-\infty = \alpha_0 < \alpha_1 < \alpha_2 < \alpha_3 = \infty\). The hidden process \( Y_{ij}^{*} \) is generated from a linear mixed model with two covariates, one time-constant \( x_{1i} \) and one time-varying \( x_{2ij} \),

\[
Y_{ij}^{*} = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2ij} + U_i + \epsilon_{ij}.
\]
The observation times $t_{ij}$ are assumed equi-spaced. The time-constant covariate, $x_{1ij} = x_{1i}$, for each $j$, is generated from a Bernoulli distribution with success probability 0.7. The time-varying covariate $x_{2ij}$ is drawn from an autoregressive process of order one with autocorrelation 0.6 and standard error 0.2. Finally, the errors $\epsilon_{ij}$ are simulated from an autoregressive process of order one with autocorrelation parameter $\gamma_{s_i}$. We consider two different autocorrelation regimes: one for the first 100 subjects ($S_i = 1$, $i = 1, \ldots, 100$) and one for the second 100 subjects ($S_i = 2$, $i = 101, \ldots, 200$). For identifiability, we choose to fix the first cutpoint $\alpha_1 = 0$ leaving the intercept $\beta_1$ free to vary. The model has seven parameters, namely one cutpoint $\alpha_2$, three regressor coefficients $\beta_1$, $\beta_2$ and $\beta_3$, two autocorrelation parameters $\gamma_1$ and $\gamma_2$ and the variance of the random effect $\sigma^2$.

Here, we show a set of results obtained with parameter values fixed to $\alpha_2 = 1.5, \beta_0 = -1.5, \beta_1 = 1, \beta_2 = 1, \gamma_1 = 0.5, \gamma_2 = 0.8, \sigma^2 = 1$. The values of parameters $\gamma_1, \gamma_2$ and $\sigma^2$ are chosen in such a way that the correlation between two hidden continuous variables separated by one time-unit, $\rho_1 = (\sigma^2 + \gamma)/(\sigma^2 + 1)$, is equal to 0.75 and 0.9 for the first 100 and the second 100 subjects, respectively. Thus, the simulated data is characterized by strong and very strong correlation regimes. We choose to illustrate this scenario because it is potentially difficult to handle by pairwise likelihood.

Each of the 200 simulated data sets was refitted by pairwise likelihoods with an increasing number of subsequent pairs. The pairwise likelihood formed only by pairs separated by one unit ($q = 1$) cannot identify both the autocorrelation parameters and $\sigma^2$, thus we must consider at least also pairs distant two units, $q \geq 2$.

Starting values for the pairwise likelihood with $q = 2$ are obtained as follow. The cut point $\alpha_2$ and the regression coefficients $\beta$ are provided by fitting ordinal probit regression as if the data were independent. Consequently, the starting values for the autocorrelation parameters $\gamma_1$ and $\gamma_2$ as well as the variance of the random effect $\sigma^2$ were fixed to zero. The starting values for the pairwise likelihoods with $q > 2$ are the estimates by pairwise likelihood with $q - 1$. The maximal distance considered is $q = 20$.

Before summarizing the results, we give some computational details. We implement composite likelihood estimation and model selection for MAOP models using the R programming language (R Development Core Team 2008). The bivariate Gaussian integrals are approximated by the method described in Genz (1994). The vector parameter $\theta$ contains constraints on the cutpoints, the autoregressive parameters and the random effects. Such constraints are met by re-parametrizing the model in terms of a parameter $\tilde{\theta}$ with components $\tilde{\alpha}_2 = \log \alpha_2$, $\tilde{\beta}_r = \beta_r$ ($r = 1, 2, 3$), $\tilde{\gamma}_r = \log \left( \gamma_r/(1-\gamma_r) \right)$ ($r = 1, 2$), $\tilde{\sigma}^2 = \log \sigma^2$. Optimization was performed by the Broyden, Fletcher, Goldfarb and Shanno (BFGS) algorithm as implemented in the R function optim with coded analytic gradient. Computer programs are include in the web supplementary material.

Computations used an iMac 2.2 GHz Intel Core 2 Duo with 2 Gb 667 Mhz DDR2 SDRAM. The average computational time to obtain one estimate (and relative standard errors) ranges from 4.9 CPU seconds for the pairwise likelihood with $q = 2$ to 14.5 CPU seconds for the pairwise likelihood with $q = 20$. 

2
The results displayed in Table 1 suggest that maximum pairwise likelihood estimators behave well for all the parameters despite the high level of serial correlation among the hidden variables.

Table 1: Simulations. Pairwise likelihood estimates based on 200 simulations using a maximal admissible distance between pairs of observations equals to \( q \) units.

<table>
<thead>
<tr>
<th></th>
<th>( q = 2 )</th>
<th>( q = 5 )</th>
<th>( q = 10 )</th>
<th>( q = 15 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>model</td>
<td>est.</td>
<td>s.e.</td>
<td>est.</td>
</tr>
<tr>
<td>( \alpha_2 )</td>
<td>1.500</td>
<td>1.503</td>
<td>0.072</td>
<td>1.501</td>
</tr>
<tr>
<td>( \beta_1 )</td>
<td>−1.500</td>
<td>−1.491</td>
<td>0.177</td>
<td>−1.489</td>
</tr>
<tr>
<td>( \beta_2 )</td>
<td>1.000</td>
<td>0.999</td>
<td>0.179</td>
<td>0.997</td>
</tr>
<tr>
<td>( \beta_3 )</td>
<td>1.000</td>
<td>0.999</td>
<td>0.083</td>
<td>0.997</td>
</tr>
<tr>
<td>( \gamma_1 )</td>
<td>0.500</td>
<td>0.499</td>
<td>0.045</td>
<td>0.501</td>
</tr>
<tr>
<td>( \gamma_2 )</td>
<td>0.800</td>
<td>0.798</td>
<td>0.025</td>
<td>0.798</td>
</tr>
<tr>
<td>( \sigma^2 )</td>
<td>1.000</td>
<td>0.993</td>
<td>0.215</td>
<td>0.986</td>
</tr>
</tbody>
</table>

The precision of the estimates is sensibly affected by the chosen distance \( q \). The behavior is quite different for the various parameters. For some parameters standard errors continue to decrease for increasing \( q \), whereas for other parameters, after an initial decrease, the standard errors increase. As an overall measure of the effect of \( q \) on the parameter estimates, we consider the generalized variance shown in Figure 1. According to this criterion, the overall performance of pairwise likelihood estimates improves until distance \( q = 11 \), after which the generalized variance seem to stabilize or, even, slightly increase.

This simulation study was repeated by varying the values of the autocorrelation parameters \( \gamma_1 \) and \( \gamma_2 \). As expected, we found that the optimal distance \( q \) is strongly related to the level of serial correlation. For example, with parameters chosen in such a way that \( \rho_1 \) and \( \rho_2 \) are both smaller than 0.5, there are little or no improvements by using a pairwise likelihood with \( q \) larger than 2.

**B Software**

The methodology discussed in the article is provided with the R package maop attached to web-supplementary appendix. The code necessary to reproduce the analysis is reported in the following subsections. It is required to install the additional R packages mnormt and xtable both available from the CRAN website.

**B.1 Monte Carlo evidence**

Load packages and set the random number generator seed to make analyses reproducible:

```r
library(maop)
library(mnormt)
```

3
Figure 1: Simulations. Log-generalized variance as a function of the maximal admissible distance $q$ between pairs of observations included in the pairwise likelihood.

```r
set.seed(2410)

Create function `maop.sim` for simulating data from a MAOP model with balanced design and different autocorrelations:

```r
maop.sim <- function(theta, X, n.subj, n.obs, n.cat, corr.factor){
    n.par <- length(theta)
    n.corr <- nlevels(corr.factor)
    n.regr <- dim(X)[2]
    if(n.cat>2){
        cuts <- theta[1:(n.cat-2)]
        beta <- theta[(n.cat-2)+1:n.regr]
    } else
        beta <- theta[1:n.regr]
    rho <- theta[(n.par-n.corr):(n.par-1)]
    autocorr <- rho[as.numeric(corr.factor[seq(by=n.obs, length=n.subj)])]
}
sigma2 <- theta[n.par]

resid <- matrix(nrow=n.subj, ncol=n.obs)
for(i in 1:n.subj){
  R <- diag(n.obs)
  R <- (sigma2+autocorr[i]^abs(row(R)-col(R)))
  resid[i,] <- rmnorm(1, varcov=R)
}

hidden <- matrix(X%*%beta, nrow=n.subj, byrow=TRUE)+resid
hidden <- as.vector(t(hidden))
if(n.cat>2)
  ris <- ordered(cut(hidden, c(-Inf, 0, cuts, Inf)))
else
  ris <- ordered(cut(hidden, c(-Inf, 0, Inf)))
ris

Set the simulation quantities:

n.subj <- 200
n.obs <- 50
n.cat <- 3
n.repl <- 200
corr.levels <- 2
n.par <- 7
dim.answer <- 2*n.par+5
max.order <- 20
true <- c(1.5, -1.5, 1, 1, .5, .8, 1)

Simulate covariates and data and save into files sim_x.Rda and sim_data.Rda, respectively:

x1 <- rep(rbinom(n.subj, 1, .7), each=n.obs)
x2 <- as.vector(1+0.2*replicate(n.subj, arima.sim(n=n.obs, list(ar=.6))))
X <- model.matrix(~x1+x2)
save(X, file="sim_x.Rda")
corr.factor <- rep(gl(2, n.subj/2), each=n.obs)
subj <- as.factor(rep(1:n.subj, each=n.obs))
sim.data <- list(NULL)
for(m in 1:n.repl){
  sim <- maop.sim(true, X, n.subj, n.obs, n.cat, corr.factor)
  sim.data[[m]] <- sim
}
save(sim.data, file="sim_data.Rda")
Fit MAOP models to the simulated data by pairwise likelihood with \( q = 2, \ldots, 20 \):

```r
ris <- array(0, c(dim.answer, (max.order-1), n.repl))
for(m in 1:n.repl) {
    fo <- sim.data[[m]] ~ x1 + x2
    for(k in 2:max.order) {
        time <- system.time(fit <- try(maop(fo,
                subjects=subj,
                randomEffect=TRUE,
                distance=k,
                print.info=FALSE,
                dynamics=corr.factor,
                startValues=if(k>2) fit$coeff)))
        if(class(fit) == "try-error")
            ris[, (k-1), m] <- rep(NA, dim.answer)
        else
            ris[, (k-1), m] <- c(fit$coeff, fit$stdErr, time)
    }
}
save(ris, file="sim_pair.Rda")
```

### B.1.1 Summaries

Next lines produces Figure 1 and Table 1 of this web supplementary appendix. Results are read from the file `sim_pair.Rda` thus next lines may be executed without previous run of the simulations.

```r
load("sim_pair.Rda")
require(xtable)
gen.var <- apply(apply(log(ris[8:14,,]^2), c(2,3), sum),1,mean)
pdf("varin_czado_fig1.pdf")
plot(c(2:20), gen.var, type="l", bty="l", xlab="q", ylab="log-generalized variance")
dev.off()
tab <- cbind(c(1.5, -1.5, 1.0, 1.0, 0.5, 0.8, 1.0),
              apply(ris[1:7,1],1,mean), apply(ris[8:14,1],1,mean),
              apply(ris[1:7,4],1,mean), apply(ris[8:14,4],1,mean),
              apply(ris[1:7,9],1,mean), apply(ris[8:14,9],1,mean),
              apply(ris[1:7,14],1,mean), apply(ris[8:14,14],1,mean))
rownames(tab) <- c("$\alpha_2$", "$\beta_1$", "$\beta_2$", "$\beta_3$",
```

---

\[1\] **Warning:** the complete simulation exercise takes around 11 hours on my laptop MACBook Pro 2.4 GhZ with 2 GB 667 MHz DDR2 SDRAM. If you are not ready to wait so much time, you can simply get the results stored in file “sim_data.Rda”.

6
B.2 Migraine data analysis

Read the migraine data and construct the covariates used in the analysis of Section 4 of the article:

```r
library(maop)
data(migraine)
attach(migraine)
university <- rep(0, length(HA))
university[ED==5] <- 1
university[ED==6] <- 1
university[ED==7] <- 1
university <- as.factor(university==1)
analgesics <- as.factor(MEDANALG == "Y")
wc <- factor(cut(MINWC, breaks=c(-50, -10, 0, 10, 30)))
high.pressure <- PMN > 10130
high.pressure.yesterday <- PMN1P > 10130
change <- rep("persistence", length(high.pressure))
change[(!high.pressure & high.pressure.yesterday)] <- "worse"
change[(high.pressure & !high.pressure.yesterday)] <- "better"
change <- factor(change)
humidity <- cut(RHMN, breaks=c(0, 60, 80, 100))

Fit the larger model by pairwise likelihood with distance $q = 2, \ldots, 20$ and save results in file “tuning_migraine.Rda”:

```r
full <- HA ~ university + analgesics + wc + humidity + change
order.values <- 2:20
n.par <- 17
ris <- array(NA, c(2*n.par, length(order.values)))
for(i in order.values){
  this.ris <- try(maop(full,
                    subjects=SUBJECT,
                    periods=PERIOD,
                    randomEffect=TRUE,
                    distance=i,
                    dynamics=analgesics,
                    order.values=i)
  if(is.null(this.ris))
    ris[,i] <- NA
  else
    ris[,i] <- this.ris
}
```

colnames(tab) <- c("model", "est.", "s.e.", "est.", "s.e.", "est.", "s.e.",
"est.", "s.e.")
print(xtable(tab, digits=3), type="latex",
sanitize.text.function = function(x) {x})
startValues=if(i>2) this.ris$coeff))
if(class(this.ris)="try-error")
ris[, (i-1)] <- rep(NA, 2*n.par)
else
ris[, (i-1)] <- c(this.ris$coeff, this.ris$stdErr)
save(ris, file="tuning_migraine.Rda")
}

Best fitting is provided by the pairwise likelihood with \( q = 12 \):

\[
\text{gen.var} \leftarrow \text{apply}(\text{ris}, 2, \text{function}(i) \{ \sum \log(i[(n.par+1):(2*n.par)]^2) \})
\]

## best value correspond to distance 12
best.order <- 1 + which.min(gen.var)

Fit all possible submodels:

\[
\text{base} \leftarrow \text{HA~university+analgesics}
\]

all.models <- list(NULL)
all.models[[1]] <- base
all.models[[2]] <- update(base, .~.+change)
all.models[[3]] <- update(base, .~.+humidity)
all.models[[4]] <- update(base, .~.+wc)
all.models[[5]] <- update(base, .~.+change+humidity)
all.models[[6]] <- update(base, .~.+change+wc)
all.models[[7]] <- update(base, .~.+humidity+wc)
all.models[[8]] <- update(base, .~.+change+humidity+wc)

fit <- lapply(all.models,
maop,
subjects=SUBJECT,
periods=PERIOD,
randomEffect=TRUE,
distance=best.order,
dynamics=analgesics)

Get the best model accordingly to CLIC statistics:

\[
\text{clics} \leftarrow \text{unlist}(\text{lapply}(\text{fit}, \text{function}(i) i$clic))
\]

\[
\text{clics-min(clics)}
\]

Finally, fit the best model but with an unique autocorrelation parameter and save all models fitted in the file migraine_analysis.Rda:

\[
\text{fit[[9]]} \leftarrow \text{maop(} \text{fit[[2]]$formula,}
\]

\[
\text{subjects=SUBJECT,}
\text{periods=PERIOD,}
\]
randomEffect=TRUE,
  
distance=best.order,
  
dynamics=NULL)

fit[[9]]$clic
save(fit, file="migraine_analysis.Rda")

B.2.1. Summaries

Next lines produce the tables reported in the article. Results are read from file migraine_analysis.Rda, thus next lines can be executed without previous run of the migraine data analysis code.

Table 2:

library(maop)
data(migraine)
attach(migraine)
require(xtable)
y <- HA[SUBJECT==levels(SUBJECT)[1]]
n <- length(y)
twostep <- table(y[2:n], y[1:(n-1)])
for(i in 2:135){
y <- HA[SUBJECT==levels(SUBJECT)[i]]
n <- length(y)
twostep <- twostep+table(y[2:n], y[1:(n-1)])
}
xtable(round(prop.table(twostep,1),2))

Table 3:

load("migraine_analysis.Rda")
logPair <- unlist(lapply(fit, function(i) i$logPair))
clic <- unlist(lapply(fit, function(i) i$clic))
delta <- clic[1:8]-min(clic[1:8])
clic.weights <- exp(-delta/2)/sum(exp(-delta/2))
models <- as.table(cbind(logPair[1:8], clic[1:8], clic.weights))
rownames(models) <- c("base", "+c", "+h", "+w", "+ch", "+cw", "+hw", "+chw")
colnames(models) <- c("logPair", "clic", "clic weights")
print(xtable(models, digits=2))

Table 4:

sd.diff <-
  sqrt(fit[[1]]$varMatrix[8,8]+fit[[1]]$varMatrix[9,9]-2*fit[[1]]$varMatrix[8,9])
base <- cbind(c(fit[[1]]$coeff[1:7], rep(0, 2), fit[[1]]$coeff[8:9],

9
fit[[1]]$coeff[9]-fit[[1]]$coeff[8], fit[[1]]$coeff[10]),
c(fit[[1]]$stdErr[1:7], rep(0, 2), fit[[1]]$stdErr[8:9],
  sd.diff, fit[[1]]$stdErr[10])

sd.diff <- sqrt(fit[[2]]$varMatrix[10,10]+fit[[2]]$varMatrix[11,11]
  -2*fit[[2]]$varMatrix[10,11])
best <- cbind(c(fit[[2]]$coeff[1:11], fit[[2]]$coeff[11]-fit[[2]]$coeff[10],
  fit[[2]]$coeff[12]), c(fit[[2]]$stdErr[1:11], sd.diff, fit[[2]]$stdErr[12]))
best.single <- cbind(c(fit[[9]]$coeff[1:10], rep(0,2), fit[[9]]$coeff[11]),
  c(fit[[9]]$stdErr[1:10], rep(0,2), fit[[9]]$stdErr[11]))

best <- cbind(best, best.single)

rownames(tab) <- c("$\alpha_2$", "$\alpha_3$", "$\alpha_4$", "$\alpha_5$",
  "intercept", "university", "analgesics", "change2", "change3",
  "$\gamma_\text{F}$", "$\gamma_\text{T}$", "$\gamma_\text{T}-\gamma_\text{F}$", "$\sigma^2$"
  
  colnames(tab) <- c("est.", "s.e.", "est.", "s.e.", "est.", "s.e.")

print(xtable(tab, digits=3), type="latex",
       sanitize.text.function = function(x) {x})

References


R Development Core Team (2008). R: A Language and Environment for Statistical Comput-