

Mixture Analysis of the Signal Tandmobiel® Data Using the Package mixAK

Arnošt Komárek

May 7, 2009

This document supplements a paper Komárek (2009) and shows an analysis of a part of Signal Tandmobiel® data introduced in Vanobbergen et al. (2000). We will estimate a joint distribution of the emergence times of six permanent teeth from one quadrant of the mouth. The dataset contains observed intervals of teeth emergence for 4 430 Flemish children resulting from a longitudinal project performed in 1996–2002. The observed intervals result from annual dental examinations and are hence of the length approximately one year.

1 Introduction

- Due to the fact that some code (especially MCMC) is time consuming, some code chunks found in this vignette are not run when compiling the package. You should set the variable `RUN.TIMECONSUMING.CODE` to `TRUE` to run full MCMC and related code.
- Having run full MCMC and related code, setting the variable `RUN.ALLOUT` to `TRUE` will cause that all output shown in this vignette is re-created and not taken from previously computed results.
- Results based on full MCMC runs are stored (chains excluded) in `Tandmob-Result.RData` in `./RESULT_OBJ` directory and are used to create majority of this vignette at the time of compilation of the package. Additionally, file `Tandmob-PDensBiModelK02.RData` in the same directory contains estimated predictive density.

R⇒ Setting variables `RUN.ALLOUT` and `RUN.TIMECONSUMING.CODE`.

```
> RUN.TIMECONSUMING.CODE <- FALSE
> RUN.ALLOUT <- FALSE
```

R⇒ Directory to store postscript files with figures. Figures which require chains are stored in `FIGKEEPDIR` directory all other figures are stored in `FIGDIR` directory.

```
> FIGDIR <- "./figures/"
> FIGKEEPDIR <- "./figuresKeep/"
```

R⇒ Directories with results computed in past. Objects with chains will be stored in directory specified by variable `RESULTDIR`. All other objects will be stored in directory `RESULT2DIR`.

```
> RESULTDIR <- "/home/komarek/RESULT_OBJ/mixAK-Tandmob-S090426/"
> RESULT2DIR <- "./RESULT_OBJ/"      ### available in package as /inst/doc/RESULT_OBJ
```

R⇒ Display options.

```
> options(width = 80)
```

R⇒ Load results computed in past. Variable `Kshow` determines which value for fixed number of components K is used in section 4.

```
> Kshow <- 2
> if (RUN.ALLOUT){
+   load(paste(RESULT2DIR, "Tandmob-Result.RData", sep=""))
+   ## contains ModelK (without chains), PDensUniModelK
+   load(paste(RESULT2DIR, "Tandmob-PDensBiModelK0", Kshow, ".RData", sep=""))
+   ## contains PDensBiModelK[[Kshow]]
+   load(paste(RESULTDIR, "Tandmob-Model0", Kshow, ".RData", sep=""))
+   ## contains Model0=ModelK[[Kshow]] (chains included)
+   PDensUniModel0 <- PDensUniModelK[[Kshow]]
+   PDensBiModel0 <- PDensBiModelK[[Kshow]]
+ }else{
+   load(paste(RESULT2DIR, "Tandmob-Result.RData", sep=""))
+   ## contains ModelK (without chains), PDensUniModelK
+   load(paste(RESULT2DIR, "Tandmob-PDensBiModelK0", Kshow, ".RData", sep=""))
+   ## contains PDensBiModelK[[Kshow]]
+   Model0 <- ModelK[[Kshow]]
+   PDensUniModel0 <- PDensUniModelK[[Kshow]]
+   PDensBiModel0 <- PDensBiModelK[[Kshow]]
+ }
```

R⇒ Load the package `mixAK` and packages `coda` and `colorspace`. Package `coda` is used to perform some basic convergence diagnostics, package `colorspace` is used to draw nicer image plots with estimated bivariate densities.

```
> library("mixAK")
> library("coda")
> library("colorspace")
```

2 Exploration of the data

R⇒ The data are read as follows.

```
> data("Tandmob", package = "mixAK")
> data("TandmobEmer", package = "mixAK")
```

R⇒ The following table shows the number and proportion of each type of censoring for each tooth.

```
> NUM.PROP.CENS <- function(data, tanden)
+ {
+   TABLE <- matrix(NA, nrow=length(tanden), ncol=3)
+   rownames(TABLE) <- paste(tanden)
+   colnames(TABLE) <- c("Left", "Interval", "Right")
+   for (tt in 1:length(tanden)){
+     ebeg <- get(data)[,paste("EBEG.", tanden[tt], sep="")]
+     eend <- get(data)[,paste("EEND.", tanden[tt], sep="")]
+     TABLE[tt, "Left"] <- sum(is.na(ebeg) & !is.na(eend))
+     TABLE[tt, "Interval"] <- sum(!is.na(ebeg) & !is.na(eend))
+     TABLE[tt, "Right"] <- sum(!is.na(ebeg) & is.na(eend))
+     rm(list=c("ebeg", "eend"))
+   }
+   PROP.TABLE <- t(round(apply(TABLE, 1, prop.table)*100, 3))
+   RET <- list(Table=TABLE, Prop.Table=PROP.TABLE)
+   return(RET)
+ }
> tanden <- rep(c(10, 20, 30, 40), 7) + rep(1:7, each=4)
> TAB01 <- NUM.PROP.CENS(data="Tandmob", tanden=tanden)
```

\$Table			
	Left	Interval	Right
11	2461	1745	224
21	2461	1728	241
31	3926	436	68
41	3934	434	62
12	651	3223	556
22	652	3240	538
32	1920	2183	327
42	1952	2163	315
13	12	2095	2323
23	4	2101	2325
33	21	3070	1339
43	18	3077	1335
14	30	2733	1667
24	42	2776	1612
34	25	2851	1554
44	25	2861	1544
15	16	1861	2553
25	30	1873	2527
35	25	1795	2610
45	16	1780	2634
16	3809	554	67
26	3809	555	66
36	3820	535	75
46	3739	604	87
17	0	1053	3377
27	1	1107	3322
37	1	1640	2789
47	1	1509	2920

\$Prop.Table			
	Left	Interval	Right
11	55.553	39.391	5.056
21	55.553	39.007	5.440
31	88.623	9.842	1.535
41	88.804	9.797	1.400
12	14.695	72.754	12.551
22	14.718	73.138	12.144
32	43.341	49.278	7.381
42	44.063	48.826	7.111
13	0.271	47.291	52.438
23	0.090	47.427	52.483
33	0.474	69.300	30.226
43	0.406	69.458	30.135
14	0.677	61.693	37.630
24	0.948	62.664	36.388
34	0.564	64.357	35.079
44	0.564	64.582	34.853
15	0.361	42.009	57.630
25	0.677	42.280	57.043
35	0.564	40.519	58.916
45	0.361	40.181	59.458
16	85.982	12.506	1.512
26	85.982	12.528	1.490
36	86.230	12.077	1.693
46	84.402	13.634	1.964
17	0.000	23.770	76.230
27	0.023	24.989	74.989
37	0.023	37.020	62.957
47	0.023	34.063	65.914

3 Transformation of the data and preparation of the MCMC

R⇒ We will take teeth 1–6 of one quadrant (right maxillary teeth, i.e. teeth 11, 12, 13, 14, 15, 16). To pass the data into **NMixMCMC** function, 6-column matrices (called **y0**, **y1** and **sensor**) have to be created. Matrix **y0** will contain lower limit of observed intervals in the case of interval censoring or lower limits of right-censored observations. Matrix **y1** will contain upper limits of observed intervals in the case of interval censoring and **NA**'s in the case of right censoring. Matrix **sensor** will contain **3** for interval-censored observations and **0** for right-censored observations. All originally left-censored observations have been changed into interval-censored ones with 5 years of age as the lower limit of the observed intervals in the **data.frame TandmobEmer**. The reason for this is the fact that 5 years of age is clinically the minimal age when the permanent teeth emerge (Ekstrand et al., 2003). However note that this is not necessary and it is possible to consider also left-censored observation and pass them into the function **NMixMCMC** (use then the **data.frame Tandmob**).

R⇒ Preparation of the data using the original **data.frame Tandmob**:

```
> Emerg.min <- 5
> Tooth <- 10 + 1:6
> y0 <- Tandmob[, paste("EBEG.", Tooth, sep="")]
> y1 <- Tandmob[, paste("EEND.", Tooth, sep="")]
> y0[is.na(y0)] <- Emerg.min
> ### Left-censored changed into interval-censored with the lower limit=5
> ###
> sensor <- matrix(3, nrow=nrow(y0), ncol=length(Tooth))
> sensor[is.na(y1)] <- 0
> ###
> colnames(y0) <- colnames(y1) <- colnames(sensor) <- Tooth
> rownames(y0) <- rownames(y1) <- rownames(sensor) <- Tandmob$IDNR
```

R⇒ Preparation of the data using the **data.frame TandmobEmer**:

```
> Tooth <- 10 + 1:6
> y0 <- TandmobEmer[, paste("EBEG.", Tooth, sep = "")]
> y1 <- TandmobEmer[, paste("EEND.", Tooth, sep = "")]
> sensor <- TandmobEmer[, paste("CENSOR.", Tooth, sep = "")]
> colnames(y0) <- colnames(y1) <- colnames(sensor) <- Tooth
> rownames(y0) <- rownames(y1) <- rownames(sensor) <- TandmobEmer$IDNR
```

R⇒ Print part of prepared data:

<pre>> print(y0[1:5,])</pre>	<pre>> print(y1[1:5,])</pre>
<pre> 11 12 13 14 15 16 1 5.0 5.0 8.5 7.7 9.5 5.0 2 5.0 8.4 9.3 9.3 10.2 5.0 3 5.0 6.5 10.5 9.5 10.5 5.0 5 7.5 8.2 10.9 10.9 10.9 6.5 6 5.0 7.4 10.9 10.0 10.0 5.0 </pre>	<pre> 11 12 13 14 15 16 1 7.7 7.7 9.5 8.5 10.2 7.7 2 7.4 9.3 10.2 10.2 10.9 7.4 3 6.5 7.7 NA 10.5 NA 6.5 5 8.2 9.1 NA NA NA 7.5 6 7.4 8.4 11.8 10.9 10.9 7.4 </pre>

R⇒ Print censor indicators:

```
> print(censor[1:5, ])
```

```
      11 12 13 14 15 16
1    3  3  3  3  3  3
2    3  3  3  3  3  3
3    3  3  0  3  0  3
5    3  3  0  0  0  3
6    3  3  3  3  3  3
```

R⇒ Length of the MCMC simulation for all models in this document (burn-in of 10 000 iterations, additional 20 000 iterations are kept for the inference, thinning of 1:10):

```
> nMCMC <- c(burn = 10000, keep = 20000, thin = 10, info = 1000)
```

R⇒ Grid of values where we evaluate and subsequently plot the predictive densities for all models in this document (we use different range of values for different teeth to cover the range when a specific tooth emerges with probability approaching one):

```
> lygridBi <- 50
> lygridUni <- 100
> ymin <- c(4, 5, 6, 6, 6, 4)
> ymax <- c(10, 12, 17, 16, 17, 9)
> ygridBi <- list(seq(ymin[1], ymax[1], length=lygridBi),
+               seq(ymin[2], ymax[2], length=lygridBi),
+               seq(ymin[3], ymax[3], length=lygridBi),
+               seq(ymin[4], ymax[4], length=lygridBi),
+               seq(ymin[5], ymax[5], length=lygridBi),
+               seq(ymin[6], ymax[6], length=lygridBi))
> ygridUni <- list(seq(ymin[1], ymax[1], length=lygridUni),
+               seq(ymin[2], ymax[2], length=lygridUni),
+               seq(ymin[3], ymax[3], length=lygridUni),
+               seq(ymin[4], ymax[4], length=lygridUni),
+               seq(ymin[5], ymax[5], length=lygridUni),
+               seq(ymin[6], ymax[6], length=lygridUni))
```

4 Model with two mixture components

In this section, we will fit a mixture model with $K = 2$ components.

R⇒ We will also compute penalized expected deviance and hence two chains will be sampled.

4.1 Specification of the prior distributions and MCMC simulation

R⇒ The minimal specification of the prior distribution with $K = 2$:

```
> Prior0 <- list(priorK = "fixed", Kmax = 2)
```

R⇒ Running MCMC with default values for all prior parameters and $K = 2$ (output is shown from MCMC simulation performed by author):

```
> if (RUN.TIMECONSUMING.CODE){  
+   set.seed(770328)  
+   Model0 <- NMixMCMC(y0=y0, y1=y1, censor=censor, prior=Prior0,  
+                     nMCMC=nMCMC, scale=list(shift=0, scale=1), PED=TRUE)  
+ }
```

Chain number 1

=====

MCMC sampling started on Sun Apr 26 11:25:40 2009.

Burn-in iteration 10000

Iteration 30000

MCMC sampling finished on Sun Apr 26 13:10:24 2009.

Chain number 2

=====

MCMC sampling started on Sun Apr 26 13:10:26 2009.

Burn-in iteration 10000

Iteration 30000

MCMC sampling finished on Sun Apr 26 14:55:48 2009.

Computation of penalized expected deviance started on Sun Apr 26 14:55:50 2009.

Computation of penalized expected deviance finished on Sun Apr 26 15:18:24 2009.

R⇒ The prior distribution for the function `NMixMCMC` was the same as with

```
> Prior0 <- list(priorK="fixed", Kmax=2,  
+               delta=1,  
+               priormuQ="independentC",  
+               xi=c(8.425, 9.508172, 9.769336, 9.75195, 9.789845, 7.675),  
+               D=diag(c(31.9225, 59.54195, 67.87572, 67.30397, 68.55327, 18.0625)),  
+               zeta=7,  
+               g=0.2,  
+               h=c(0.3132587, 0.1679488, 0.1473281, 0.1488796, 0.1458720, 0.5536332))
```

Note that due to the fact that the argument `scale` has been specified in function `NMixMCMC` as `scale=list(shift=0, scale=1)`, the MCMC has been run on the original data.

4.2 Posterior inference

R⇒ Basic posterior summary of the fitted model is obtained using the command `print(Model0)`.

R⇒ Quantities shown in section labeled “Penalized expected deviance” are computed from two sampled chains and have the following meaning: `D.expect` is \hat{D}_e from Komárek (2009), `p(opt)` is estimated optimism \hat{p}_{opt} computed with unit weights, `PED` equals `D.expect` + `p(opt)` gives the estimate of penalized expected deviance with optimism computed without the use of importance sampling. Further, `wp(opt)` is estimated optimism \hat{p}_{opt} computed as described in Komárek (2009), i.e., using importance sampling. Finally, `wPED` equals `D.expect` + `wp(opt)` gives the estimate of penalized expected deviance as described in Komárek (2009).

R⇒ Quantities shown in section labeled “Deviance information criteria” are computed separately from the first and the second sampled chain. They have the following meaning: `DIC` is deviance information criterion denoted as \overline{DIC} in Komárek (2009), `pD` is the effective dimension p_D , `D.bar` is approximated posterior mean \overline{D} of the deviance and `D.in.bar` is \tilde{D} – deviance evaluated in the “estimate”.

R⇒ Section “Posterior summary statistics for moments of mixture for original data” gives posterior summary statistics for $E(\mathbf{Y}) = \mathbf{m} + SE(\mathbf{Y}^*)$ and quantities derived from $\text{var}(\mathbf{Y}) = \mathbf{S} \text{var}(\mathbf{Y}^*) \mathbf{S}'$ in the notation of Komárek (2009), separately for each generated chain.

```
> print(Model0)
```

```
2 component normal mixture estimated using MCMC
=====
```

```
Penalized expected deviance:
```

```
-----
      D.expect      p(opt)      PED      wp(opt)      wPED
74778.2223    459.2338 75237.4561    459.2338 75237.4561
```

```
Deviance information criteria:
```

```
-----
              DIC      pD      D.bar D.in.bar
Chain 1 64062.40 7098.499 56963.9 49865.40
Chain 2 64054.82 7096.818 56958.0 49861.18
```

```
Posterior summary statistics for moments of mixture for original data:
```

```
-----
Means (chain 1):
      y.Mean.1  y.Mean.2  y.Mean.3  y.Mean.4  y.Mean.5  y.Mean.6
Mean      7.00102262  8.10834928  11.29975988  10.54675810  11.51834928  6.37538449
Std.Dev.  0.01375146  0.01527585  0.02815026  0.02266548  0.03371765  0.01519845
Min.      6.94711487  8.04651059  11.19197588  10.45744045  11.39203273  6.32343325
2.5%      6.97411202  8.07860646  11.24651693  10.50249254  11.45442970  6.34505266
1st Qu.   6.99174122  8.09806241  11.28039978  10.53153059  11.49521798  6.36535623
Median    7.00122635  8.10821390  11.29913691  10.54667520  11.51756885  6.37549987
3rd Qu.   7.01032846  8.11863886  11.31816986  10.56194177  11.54052797  6.38565697
```

97.5%	7.02763640	8.13856493	11.35682278	10.59131925	11.58678065	6.40472259
Max.	7.05491105	8.16876362	11.42465980	10.62946205	11.69510593	6.43426475

Means (chain 2):

	y.Mean.1	y.Mean.2	y.Mean.3	y.Mean.4	y.Mean.5	y.Mean.6
Mean	7.00110318	8.10856452	11.29943283	10.54682157	11.51832469	6.37536765
Std.Dev.	0.01377727	0.01527937	0.02809354	0.02249410	0.03370979	0.01513347
Min.	6.94634087	8.05016462	11.19021099	10.46194276	11.37340677	6.31682037
2.5%	6.97360828	8.07906882	11.24624302	10.50315966	11.45460077	6.34575436
1st Qu.	6.99195384	8.09814585	11.28019586	10.53171044	11.49548388	6.36505283
Median	7.00124567	8.10849702	11.29896640	10.54677898	11.51746522	6.37533664
3rd Qu.	7.01036670	8.11892163	11.31795651	10.56178921	11.54061897	6.38561720
97.5%	7.02780596	8.13858318	11.35682863	10.59134962	11.58707869	6.40522481
Max.	7.05318510	8.16998521	11.42133369	10.63809863	11.67877121	6.44064805

Standard deviations and correlations (chain 1):

	y.SD.1	y.Corr.2.1	y.Corr.3.1	y.Corr.4.1	y.Corr.5.1	y.Corr.6.1
Mean	0.73193686	0.69046221	0.62497557	0.53914256	0.54549361	0.51926080
Std.Dev.	0.01229001	0.01260025	0.01630977	0.01647473	0.01828935	0.02218780
Min.	0.68391225	0.63208218	0.54483922	0.46802571	0.47003346	0.43413094
2.5%	0.70803615	0.66512178	0.59212096	0.50633349	0.50853909	0.47457376
1st Qu.	0.72364934	0.68219136	0.61407742	0.52819445	0.53345843	0.50452237
Median	0.73194636	0.69064346	0.62536820	0.53943387	0.54555324	0.51959648
3rd Qu.	0.74015803	0.69911454	0.63608019	0.55031418	0.55774004	0.53458374
97.5%	0.75630350	0.71444541	0.65611875	0.57089306	0.58103201	0.56148304
Max.	0.79280701	0.73473514	0.68552354	0.59769910	0.62247391	0.59981795
	y.SD.2	y.Corr.3.2	y.Corr.4.2	y.Corr.5.2	y.Corr.6.2	y.SD.3
Mean	0.93034835	0.61382150	0.47846794	0.49762726	0.42456799	1.33888038
Std.Dev.	0.01419046	0.01617137	0.01660534	0.01871340	0.02433653	0.03818642
Min.	0.87298810	0.54590257	0.41029615	0.42165911	0.32340115	1.21146151
2.5%	0.90301097	0.58169254	0.44562835	0.46001236	0.37575223	1.27096470
1st Qu.	0.92074441	0.60301670	0.46719011	0.48506071	0.40839809	1.31213939
Median	0.93017796	0.61413383	0.47864335	0.49788239	0.42502482	1.33658615
3rd Qu.	0.93983998	0.62489415	0.48979726	0.51028896	0.44098052	1.36294137
97.5%	0.95860917	0.64484377	0.51027175	0.53345156	0.47171558	1.41970829
Max.	0.99100969	0.66952572	0.53414393	0.57279576	0.51080810	1.52135572
	y.Corr.4.3	y.Corr.5.3	y.Corr.6.3	y.SD.4	y.Corr.5.4	y.Corr.6.4
Mean	0.59830273	0.55772378	0.48610616	1.30003544	0.73632923	0.47032940
Std.Dev.	0.01931984	0.02069057	0.02664376	0.02202476	0.01427254	0.02478035
Min.	0.52453772	0.44489211	0.36435822	1.20902995	0.66905579	0.37431619
2.5%	0.55896787	0.51557089	0.43284903	1.25916574	0.70705975	0.42007667
1st Qu.	0.58558137	0.54426821	0.46847712	1.28513771	0.72696302	0.45404815
Median	0.59869135	0.55813563	0.48645514	1.29927592	0.73684939	0.47077941
3rd Qu.	0.61180554	0.57187389	0.50421487	1.31426079	0.74607173	0.48712673
97.5%	0.63446898	0.59679595	0.53657647	1.34559611	0.76300541	0.51767794
Max.	0.68963845	0.63646183	0.58216992	1.41443024	0.78787508	0.55395921
	y.SD.5	y.Corr.6.5	y.SD.6			
Mean	1.52876064	0.50488853	0.57949181			

Std.Dev.	0.04620749	0.02645607	0.01172826
Min.	1.38098016	0.40281237	0.53351873
2.5%	1.44630482	0.45189276	0.55641042
1st Qu.	1.49642788	0.48720407	0.57157770
Median	1.52566962	0.50541136	0.57943161
3rd Qu.	1.55813809	0.52287338	0.58738606
97.5%	1.62732978	0.55568326	0.60258691
Max.	1.76983288	0.60195652	0.62588187

Standard deviations and correlations (chain 2):

	y.SD.1	y.Corr.2.1	y.Corr.3.1	y.Corr.4.1	y.Corr.5.1	y.Corr.6.1
Mean	0.73193038	0.69047857	0.62495701	0.5391806	0.54571215	0.51955487
Std.Dev.	0.01222552	0.01276867	0.01649097	0.0166532	0.01822416	0.02228140
Min.	0.68419078	0.63451577	0.55786986	0.4569567	0.45900697	0.42806345
2.5%	0.70815456	0.66441375	0.59176430	0.5054894	0.50878992	0.47500115
1st Qu.	0.72368188	0.68219890	0.61401213	0.5282277	0.53359237	0.50473753
Median	0.73197648	0.69079633	0.62528624	0.5394119	0.54592312	0.52004345
3rd Qu.	0.74000928	0.69917178	0.63606359	0.5506265	0.55817953	0.53482575
97.5%	0.75615590	0.71489740	0.65630551	0.5708424	0.58055171	0.56203937
Max.	0.78013331	0.73674696	0.68816049	0.6028364	0.61128646	0.60911313

	y.SD.2	y.Corr.3.2	y.Corr.4.2	y.Corr.5.2	y.Corr.6.2	y.SD.3
Mean	0.93040224	0.61376874	0.47847419	0.49785089	0.42464098	1.33845172
Std.Dev.	0.01432785	0.01627950	0.01667399	0.01870734	0.02419231	0.03810459
Min.	0.87458004	0.53872991	0.41203259	0.41227712	0.31909957	1.21447630
2.5%	0.90300818	0.58135917	0.44508801	0.46047869	0.37724587	1.27012515
1st Qu.	0.92051495	0.60292695	0.46737588	0.48525176	0.40841386	1.31194839
Median	0.93014634	0.61414163	0.47862421	0.49803323	0.42492399	1.33622217
3rd Qu.	0.93990290	0.62493404	0.48986882	0.51053207	0.44128408	1.36273203
97.5%	0.95908202	0.64499439	0.51024479	0.53369959	0.47110881	1.41874295
Max.	0.99097289	0.67162720	0.54293518	0.57946629	0.51892806	1.54957489

	y.Corr.4.3	y.Corr.5.3	y.Corr.6.3	y.SD.4	y.Corr.5.4	y.Corr.6.4
Mean	0.59852591	0.55780945	0.48615027	1.29999036	0.73645422	0.47059316
Std.Dev.	0.01942745	0.02052465	0.02670332	0.02182824	0.01420315	0.02486793
Min.	0.49664176	0.46816619	0.38025760	1.21477424	0.67780441	0.36529847
2.5%	0.55908511	0.51662955	0.43317760	1.25901228	0.70757615	0.42070864
1st Qu.	0.58589315	0.54433447	0.46828976	1.28489900	0.72726704	0.45416075
Median	0.59893110	0.55835707	0.48654016	1.29928296	0.73677950	0.47108515
3rd Qu.	0.61195867	0.57203660	0.50422386	1.31429359	0.74616889	0.48734453
97.5%	0.63489268	0.59616657	0.53764807	1.34488597	0.76314732	0.51805408
Max.	0.66965451	0.63037709	0.58425141	1.40418926	0.78786136	0.55658624

	y.SD.5	y.Corr.6.5	y.SD.6
Mean	1.52865200	0.50536940	0.57946697
Std.Dev.	0.04600838	0.02682988	0.01151379
Min.	1.37806663	0.37240365	0.53640705
2.5%	1.44680507	0.45133093	0.55679329
1st Qu.	1.49652250	0.48759810	0.57181555
Median	1.52570964	0.50590876	0.57948013
3rd Qu.	1.55739033	0.52363467	0.58721213

```
97.5%    1.62658443 0.55602929 0.60216855
Max.     1.75117759 0.60487881 0.62841052
```

R⇒ Computation of the marginal (univariate) predictive densities (separately for chain 1 and chain 2):

```
> if (RUN.TIMECONSUMING.CODE){
+   PDensUniModel0 <- list()
+   PDensUniModel0[[1]] <- NMixPredDensMarg(Model0[[1]], grid=ygridUni)
+   PDensUniModel0[[2]] <- NMixPredDensMarg(Model0[[2]], grid=ygridUni)
+ }
```

R⇒ Default `plot` method for the computed object (see Figure 1):

```
> postscript(paste(FIGDIR, "figTandmob01.ps", sep=""), width=10, height=7,
+           horizontal=FALSE)
> plot(PDensUniModel0[[1]])
> dev.off()
```

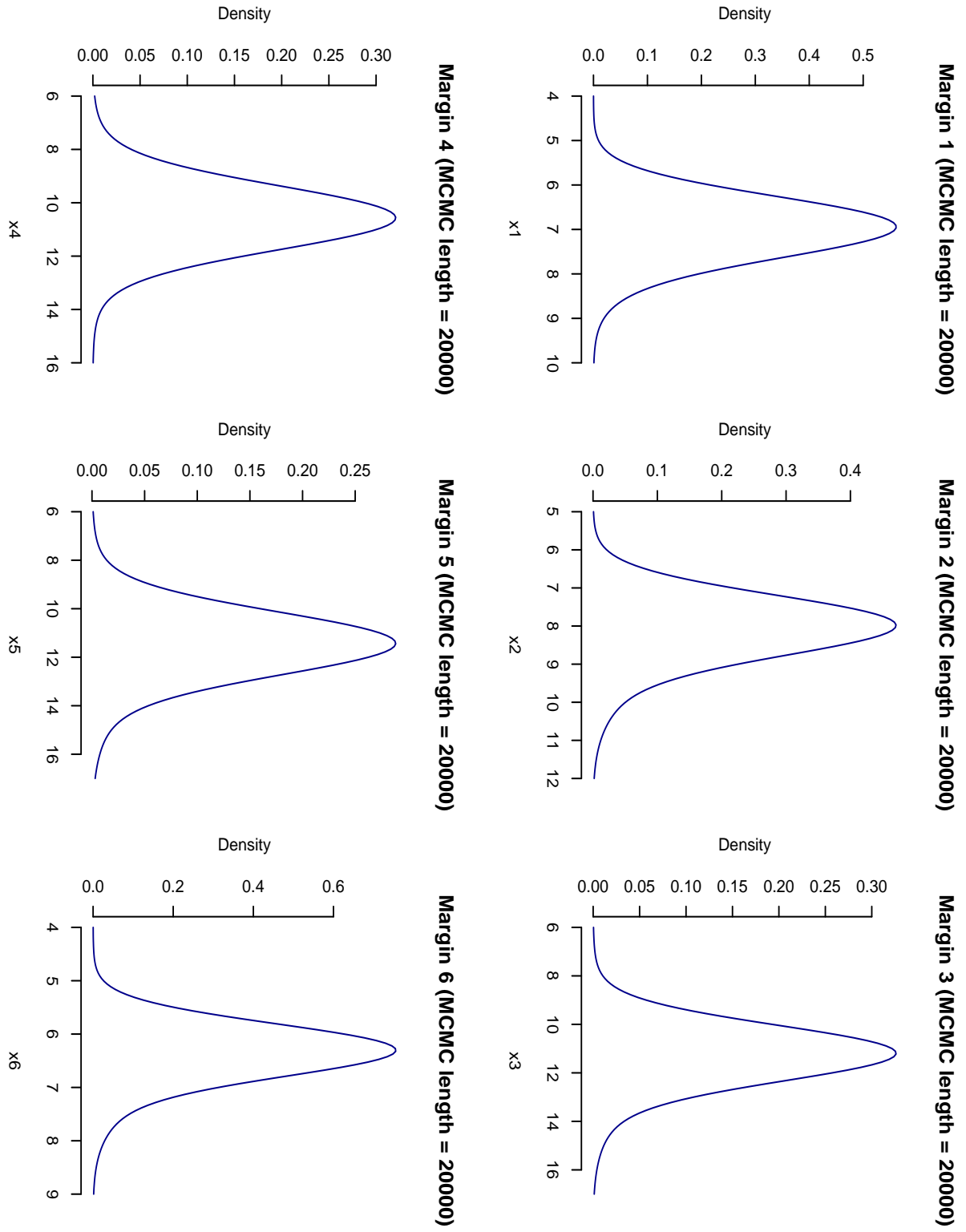


Figure 1: Default `plot` method for the marginal predictive densities based on the model with two mixture components (results from chain 1).

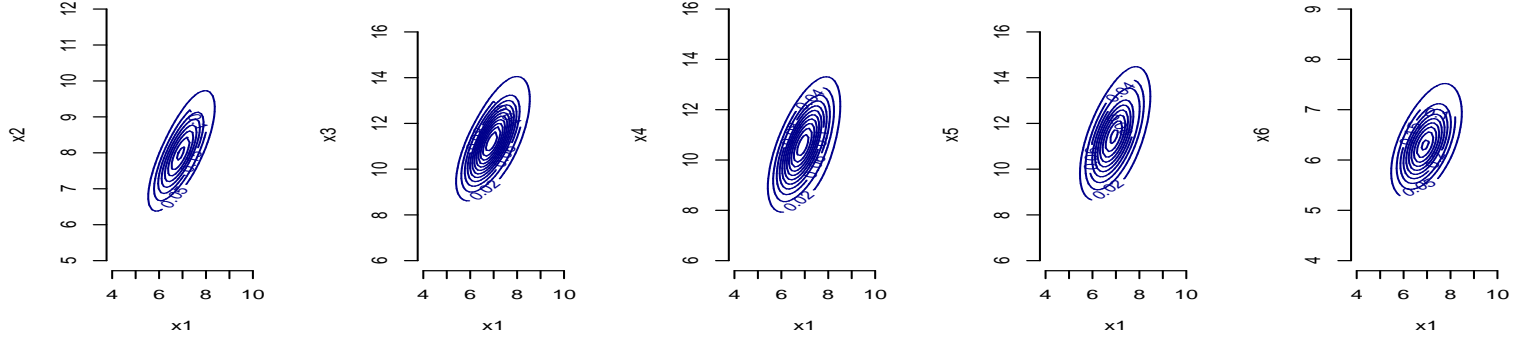
R⇒ Computation of the joint bivariate predictive densities (for each pair and separately for chain 1 and chain 2):

```
> if (RUN.TIMECONSUMING.CODE){  
+   PDensBiModel0 <- list()  
+   PDensBiModel0[[1]] <- NMixPredDensJoint2(Model0[[1]], grid=ygridBi)  
+   PDensBiModel0[[2]] <- NMixPredDensJoint2(Model0[[2]], grid=ygridBi)  
+ }
```

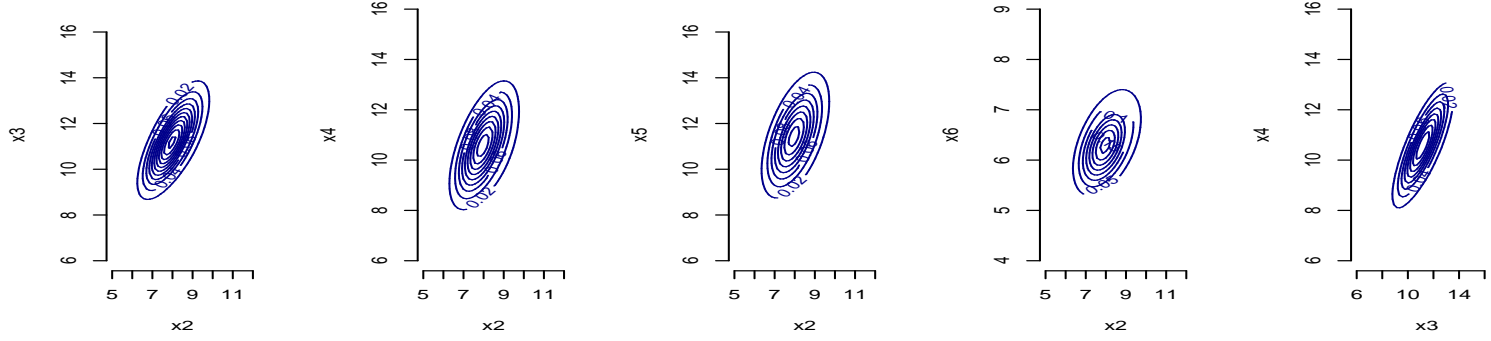
R⇒ Default `plot` method for the computed object (see Figure 2):

```
> postscript(paste(FIGDIR, "figTandmob02.ps", sep=""), width=20, height=14,  
+           horizontal=FALSE)  
> plot(PDensBiModel0[[1]])  
> dev.off()
```

jins (1, 2) (MCMC length jins (1, 3) (MCMC length jins (1, 4) (MCMC length jins (1, 5) (MCMC length jins (1, 6) (MCMC length



jins (2, 3) (MCMC length jins (2, 4) (MCMC length jins (2, 5) (MCMC length jins (2, 6) (MCMC length jins (3, 4) (MCMC length



jins (3, 5) (MCMC length jins (3, 6) (MCMC length jins (4, 5) (MCMC length jins (4, 6) (MCMC length jins (5, 6) (MCMC length

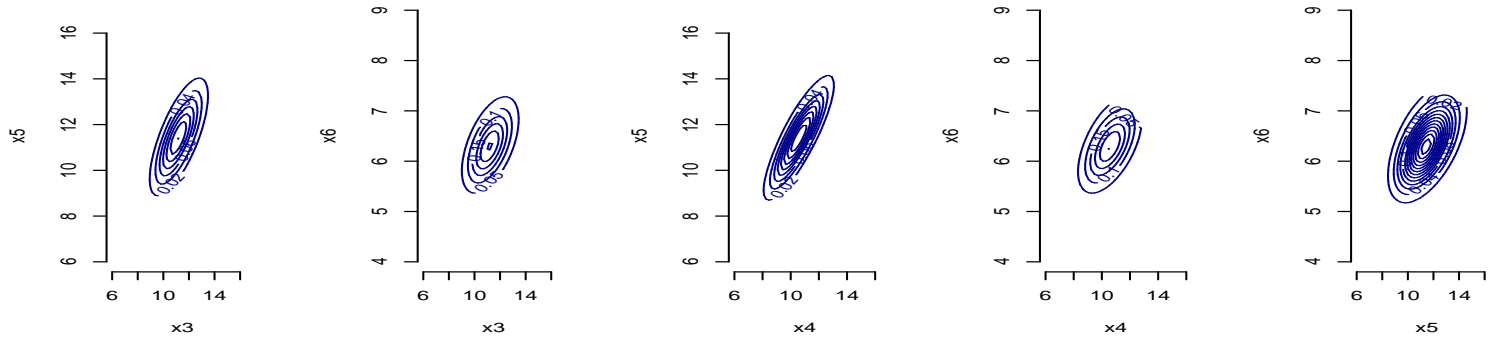


Figure 2: Default **p1ot** method for the joint bivariate predictive densities based on the model with two mixture components (results from chain 1).

4.3 Nicer figures of posterior predictive densities

R⇒ Plot of the marginal predictive densities, results from chain 1 in red, results from chain 2 in blue, (see Figure 3):

```
> postscript(paste(FIGDIR, "figTandmob03.ps", sep=""), width=7, height=5,
+           horizontal=FALSE)
> xlim <- c(4, 17)
> ylim <- c(0, 0.8)
> par(mfrow=c(2, 3), bty="n", mar=c(4, 4, 4, 0)+0.1)
> for (tt in 1:6){
+   plot(PDensUniModel0[[1]]$x[[tt]], PDensUniModel0[[1]]$dens[[tt]],
+        type="l", col="red", xlim=xlim, ylim=ylim,
+        xlab="Age (years)", ylab="Density of emergence",
+        main=paste("Tooth ", 10+tt, sep=""))
+   lines(PDensUniModel0[[2]]$x[[tt]], PDensUniModel0[[2]]$dens[[tt]], col="blue")
+ }
> dev.off()
```

R⇒ Contour plots of the joint bivariate predictive densities for each pair, results from chain 1 (see Figure 4):

```
> CH <- 1
> postscript(paste(FIGDIR, "figTandmob04.ps", sep=""), width=10, height=7,
+           horizontal=FALSE)
> par(mfrow=c(3, 5), bty="n", mar=c(4, 4, 4, 0)+0.1)
> for (tt1 in 1:5){
+   for (tt2 in (tt1+1):6){
+     contour(PDensBiModel0[[CH]]$x[[tt1]], PDensBiModel0[[CH]]$x[[tt2]],
+            PDensBiModel0[[CH]]$dens[[paste(tt1, "-", tt2, sep="")]],
+            col="red", xlab="Age (years)", ylab="Age (years)",
+            main=paste("Teeth ", 10+tt1, "-", 10+tt2, sep=""))
+   }
+ }
> dev.off()
```

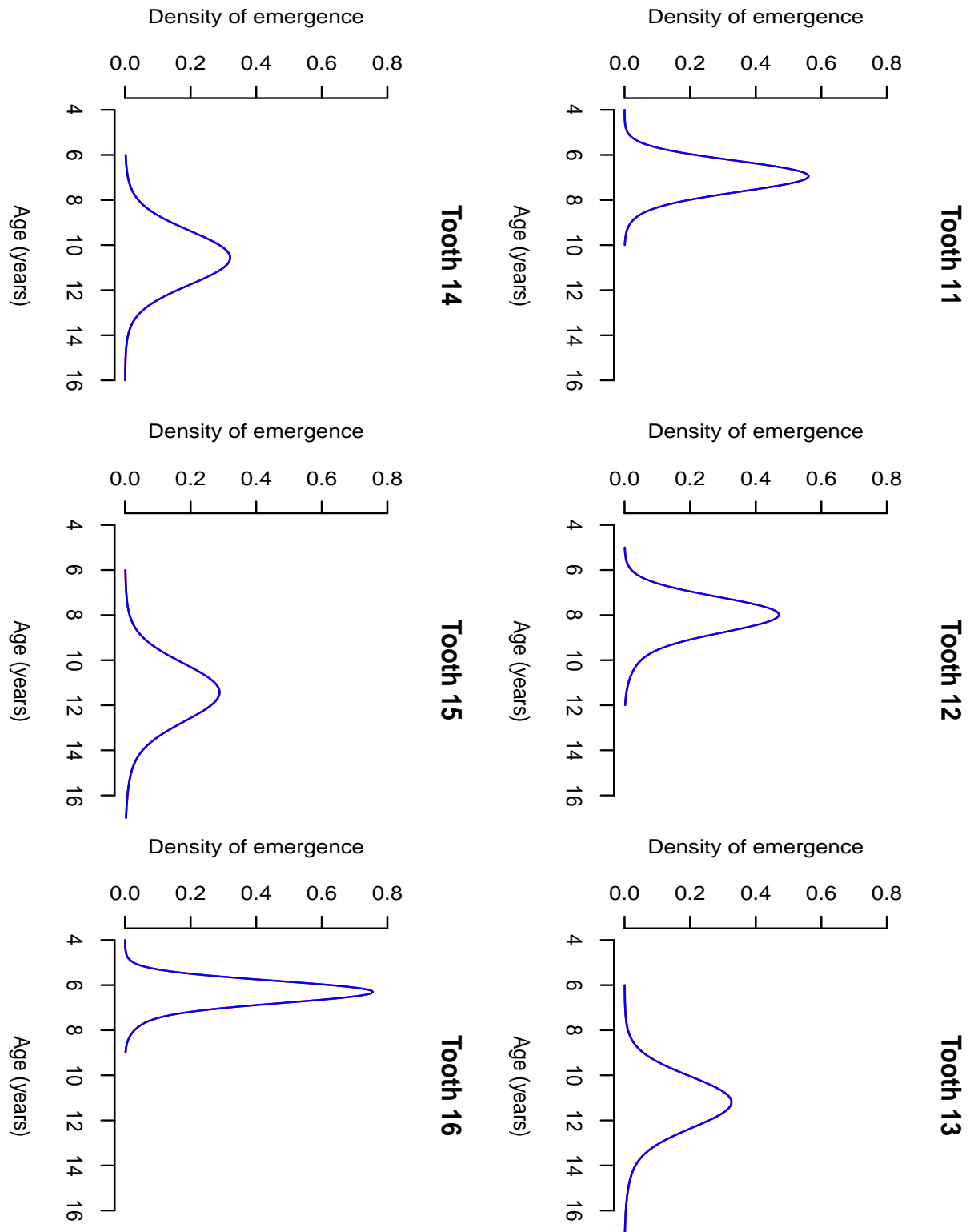



Figure 3: Marginal predictive densities based on the model with two mixture components, results from chain 1 in red, results from chain 2 in blue.

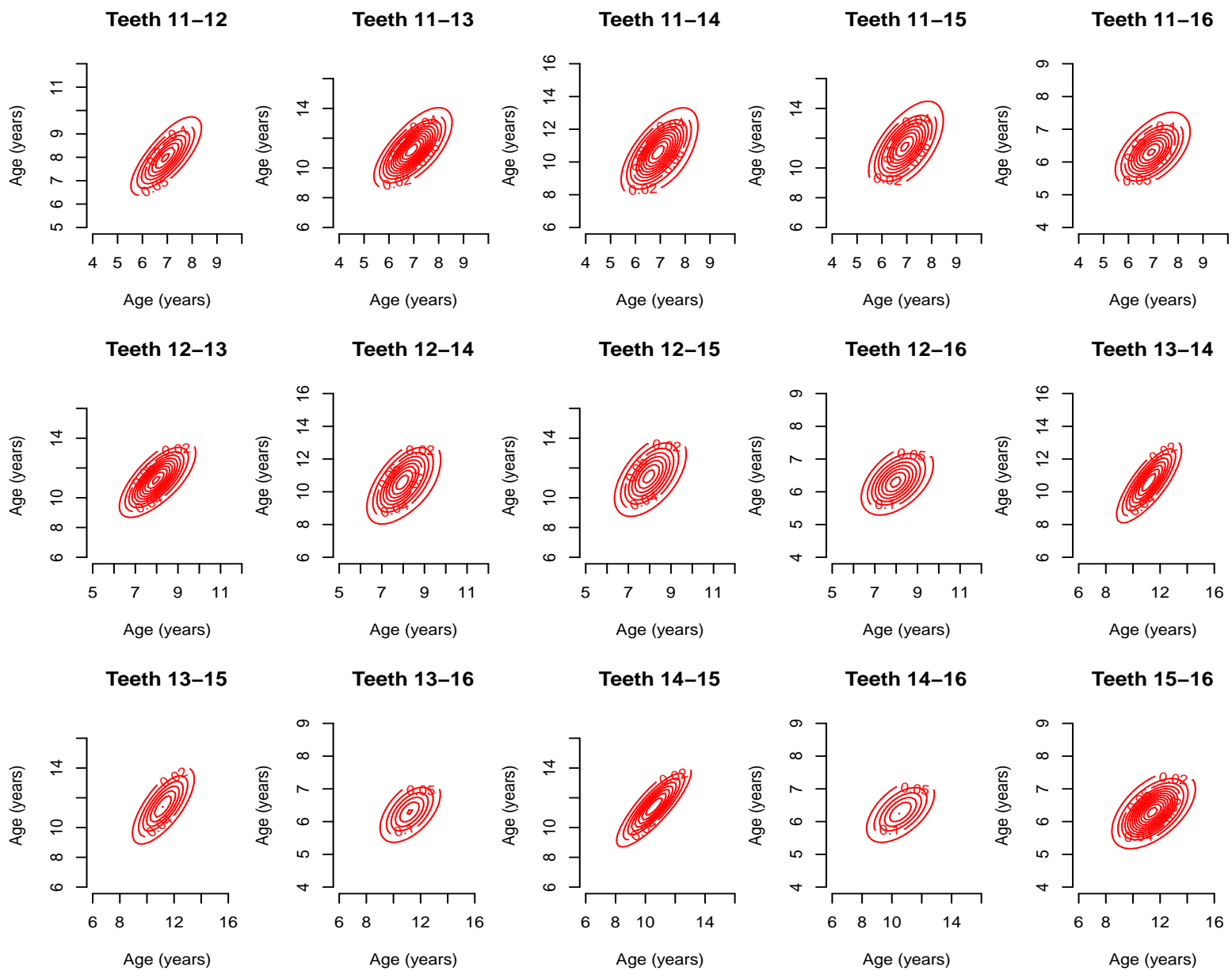


Figure 4: Contour plots of the joint bivariate predictive densities based on the model with two mixture components, results from chain 1.

R⇒ Image plots of the joint bivariate predictive densities for each pair, results from chain 1 (see Figure 5):

```
> CH <- 1
> postscript(paste(FIGDIR, "figTandmob05.ps", sep=""), width=10, height=7,
+           horizontal=FALSE)
> par(mfrow=c(3, 5), bty="n", mar=c(4, 4, 4, 0)+0.1)
> for (tt1 in 1:5){
+   for (tt2 in (tt1+1):6){
+     image(PDensBiModel0[[CH]]$x[[tt1]], PDensBiModel0[[CH]]$x[[tt2]],
+           PDensBiModel0[[CH]]$dens[[paste(tt1, "-", tt2, sep="")]],
+           col=rev(heat_hcl(33, c=c(80, 30), l=c(30, 90), power=c(1/5, 1.3))),
+           xlab="Age (years)", ylab="Age (years)",
+           main=paste("Teeth ", 10+tt1, "-", 10+tt2, sep=""))
+   }
+ }
> dev.off()
```

Note that package `colorspace` is needed to specify the colors in the plot.

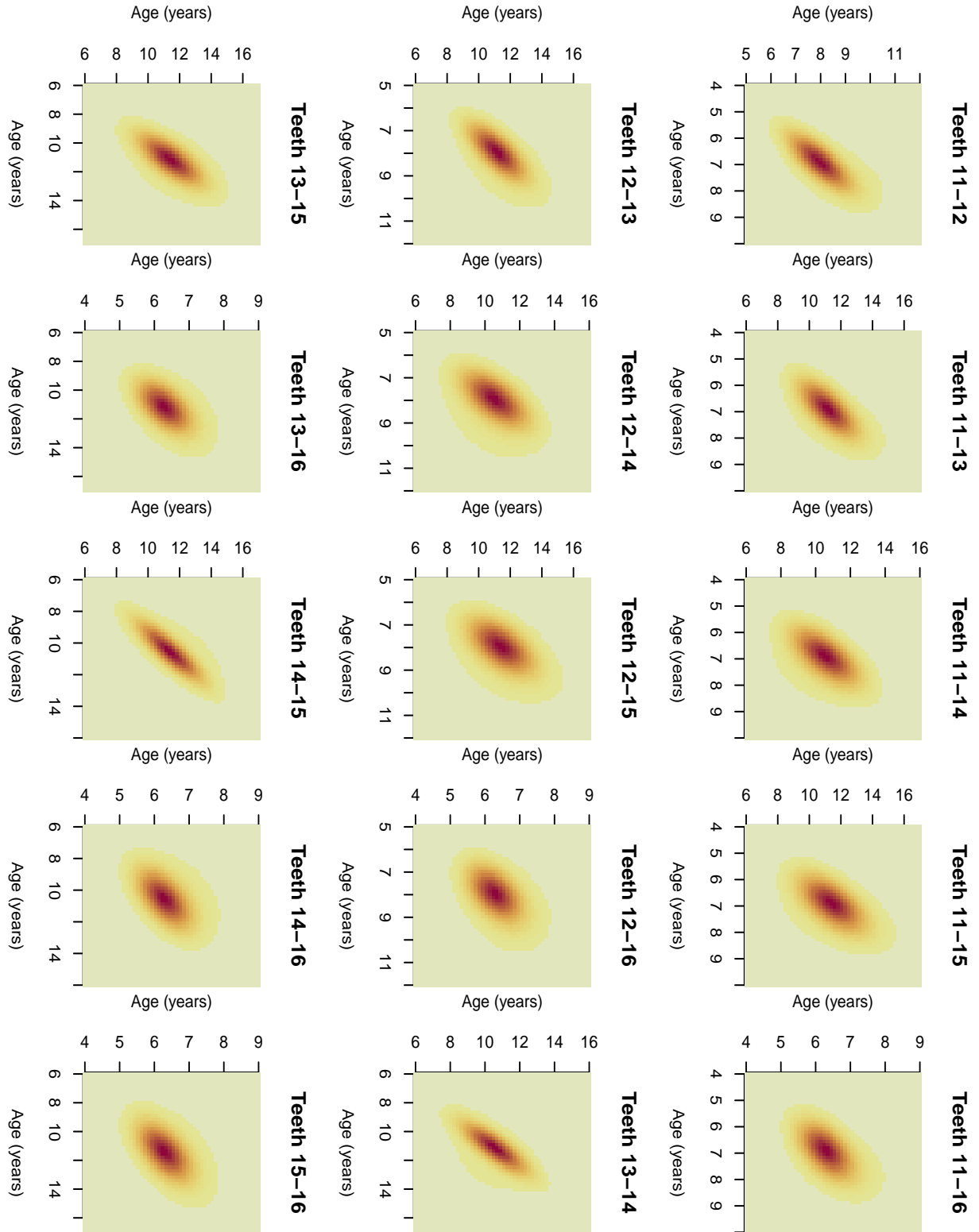


Figure 5: Image plots of the joint bivariate predictive densities based on the model with two mixture components, results from chain 1.

4.4 Convergence diagnostics

R⇒ Single chain convergence diagnostics using chain 1 will be shown here.

```
> CH <- 1
```

R⇒ Converting the chains into **mcmc** objects to be used in the package **coda**:

```
> if (RUN.ALLOUT){  
+   start <- Model0[[CH]]$nMCMC["burn"] + 1  
+   end <- Model0[[CH]]$nMCMC["burn"] + Model0[[CH]]$nMCMC["keep"]  
+   chgammaInv <- mcmc(Model0[[CH]]$gammaInv, start=start, end=end)  
+   chmixture <- mcmc(Model0[[CH]]$mixture, start=start, end=end)  
+   chdeviance <- mcmc(Model0[[CH]]$deviance, start=start, end=end)  
+ }
```

4.4.1 Traceplots

R⇒ Traceplots for the overall means of emergence (see Figure 6):

```
> if (RUN.ALLOUT){  
+   postscript(paste(FIGKEEPDIR, "figTandmob06.ps", sep=""), width=7, height=10,  
+             horizontal=FALSE)  
+   lwd <- 0.5  
+   par(mfrow=c(2, 3), bty="n")  
+   traceplot(chmixture[, "y.Mean.1"], smooth=FALSE, col="darkblue",  
+            lwd=lwd, main="E(Tooth 11)")  
+   traceplot(chmixture[, "y.Mean.2"], smooth=FALSE, col="darkblue",  
+            lwd=lwd, main="E(Tooth 12)")  
+   traceplot(chmixture[, "y.Mean.3"], smooth=FALSE, col="darkblue",  
+            lwd=lwd, main="E(Tooth 13)")  
+   traceplot(chmixture[, "y.Mean.4"], smooth=FALSE, col="darkblue",  
+            lwd=lwd, main="E(Tooth 14)")  
+   traceplot(chmixture[, "y.Mean.5"], smooth=FALSE, col="darkblue",  
+            lwd=lwd, main="E(Tooth 15)")  
+   traceplot(chmixture[, "y.Mean.6"], smooth=FALSE, col="darkblue",  
+            lwd=lwd, main="E(Tooth 16)")  
+   dev.off()  
+ }
```

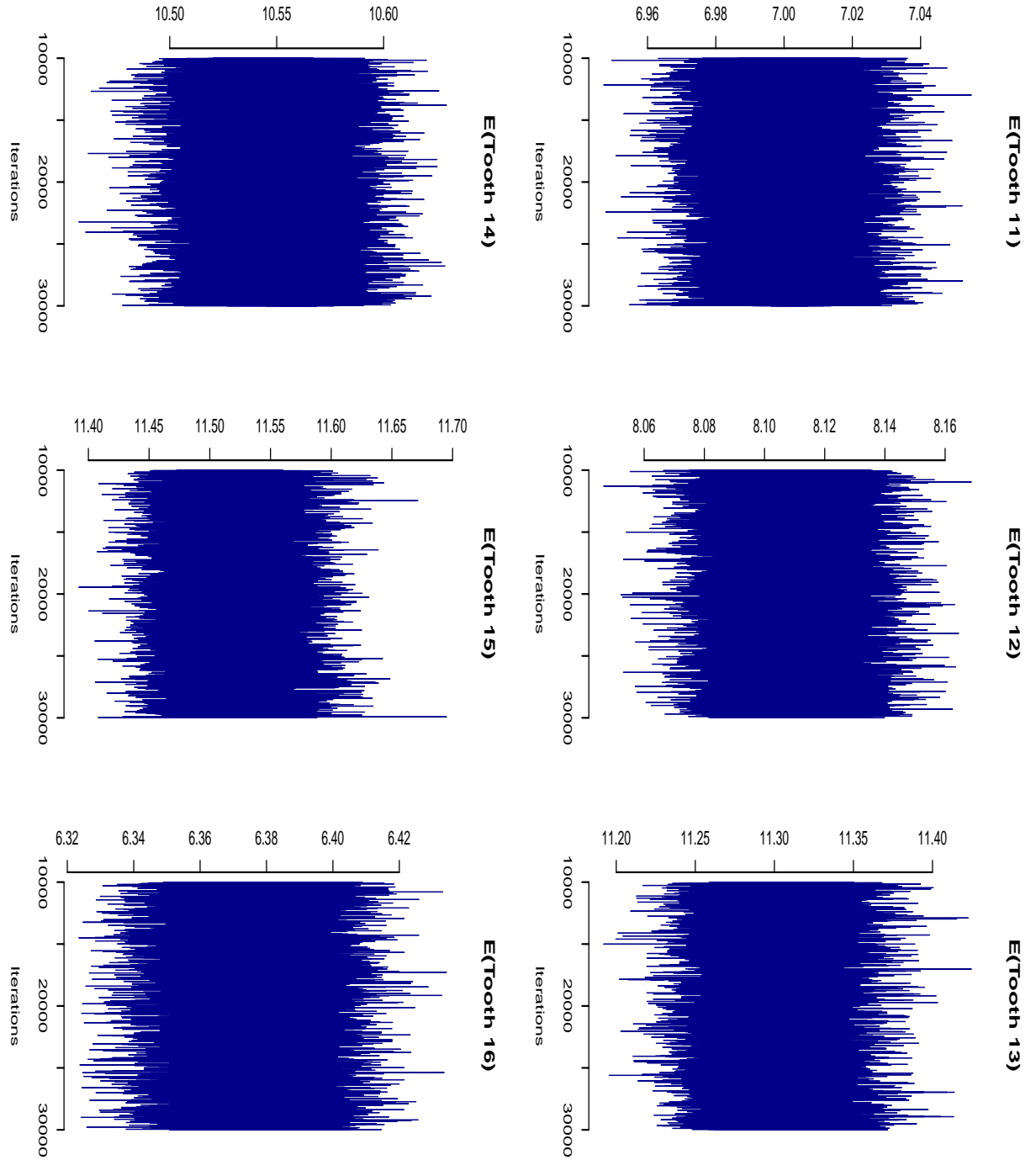


Figure 6: Model with two mixture components. Traceplots for the means of emergence.

R⇒ Traceplots for the standard deviations of emergence (not shown):

```
> if (RUN.ALLOUT){
+   postscript(paste(FIGKEEPDIR, "figTandmob07.ps", sep=""), width=7, height=10,
+             horizontal=FALSE)
+   par(mfrow=c(2, 3), bty="n")
+   traceplot(chmixture[, "y.SD.1"], smooth=FALSE, col="darkgreen",
+             lwd=lwd, main="SD(Tooth 11)")
+   traceplot(chmixture[, "y.SD.2"], smooth=FALSE, col="darkgreen",
+             lwd=lwd, main="SD(Tooth 12)")
+   traceplot(chmixture[, "y.SD.3"], smooth=FALSE, col="darkgreen",
+             lwd=lwd, main="SD(Tooth 13)")
+   traceplot(chmixture[, "y.SD.4"], smooth=FALSE, col="darkgreen",
+             lwd=lwd, main="SD(Tooth 14)")
+   traceplot(chmixture[, "y.SD.5"], smooth=FALSE, col="darkgreen",
+             lwd=lwd, main="SD(Tooth 15)")
+   traceplot(chmixture[, "y.SD.6"], smooth=FALSE, col="darkgreen",
+             lwd=lwd, main="SD(Tooth 16)")
+   dev.off()
+ }
```

R⇒ Traceplots for the pairwise correlations between the emergence times (not shown):

```
> if (RUN.ALLOUT){
+   postscript(paste(FIGKEEPDIR, "figTandmob08.ps", sep=""), width=10, height=7,
+             horizontal=FALSE)
+   par(bty="n", mar=c(1, 1, 4, 0)+0.1)
+   layout(matrix(c(1,2,3,4,5, 0,6,7,8,9, 0,0,10,11,12, 0,0,0,13,14, 0,0,0,0,15),
+                 nrow=5, byrow=TRUE))
+   traceplot(chmixture[, "y.Corr.2.1"], smooth=FALSE, col="darkgreen",
+             main="Corr 2-1")
+   traceplot(chmixture[, "y.Corr.3.1"], smooth=FALSE, col="darkgreen",
+             main="Corr 3-1")
+   traceplot(chmixture[, "y.Corr.4.1"], smooth=FALSE, col="darkgreen",
+             main="Corr 4-1")
+   traceplot(chmixture[, "y.Corr.5.1"], smooth=FALSE, col="darkgreen",
+             main="Corr 5-1")
+   traceplot(chmixture[, "y.Corr.6.1"], smooth=FALSE, col="darkgreen",
+             main="Corr 6-1")
+   traceplot(chmixture[, "y.Corr.3.2"], smooth=FALSE, col="darkgreen",
+             main="Corr 3-2")
+   traceplot(chmixture[, "y.Corr.4.2"], smooth=FALSE, col="darkgreen",
+             main="Corr 4-2")
+   traceplot(chmixture[, "y.Corr.5.2"], smooth=FALSE, col="darkgreen",
+             main="Corr 5-2")
+   traceplot(chmixture[, "y.Corr.6.2"], smooth=FALSE, col="darkgreen",
+             main="Corr 6-2")
+   traceplot(chmixture[, "y.Corr.4.3"], smooth=FALSE, col="darkgreen",
```

```

+         main="Corr 4-3")
+   traceplot(chmixture[, "y.Corr.5.3"], smooth=FALSE, col="darkgreen",
+         main="Corr 5-3")
+   traceplot(chmixture[, "y.Corr.6.3"], smooth=FALSE, col="darkgreen",
+         main="Corr 6-3")
+   traceplot(chmixture[, "y.Corr.5.4"], smooth=FALSE, col="darkgreen",
+         main="Corr 5-4")
+   traceplot(chmixture[, "y.Corr.6.4"], smooth=FALSE, col="darkgreen",
+         main="Corr 6-4")
+   traceplot(chmixture[, "y.Corr.6.5"], smooth=FALSE, col="darkgreen",
+         main="Corr 6-5")
+   dev.off()
+ }

```

R⇒ Traceplots for some other parameters (not shown):

```

> if (RUN.ALLOUT){
+   postscript(paste(FIGKEEPDIR, "figTandmob09.ps", sep=""), width=10, height=7,
+         horizontal=FALSE)
+   par(bty="n")
+   layout(matrix(c(1,1,2,2,3,3, 0,4,4,5,5,0), nrow=2, byrow=TRUE))
+   traceplot(chgammaInv[, "gammaInv1"], smooth=FALSE,
+         col="brown", main="gamma^{-1}")
+   traceplot(chdeviance[, "LogL0"], smooth=FALSE,
+         col="red", lwd=lwd, main="Log(L0)")
+   traceplot(chdeviance[, "LogL1"], smooth=FALSE,
+         col="red", lwd=lwd, main="Log(L1)")
+   traceplot(chdeviance[, "dev.complete"], smooth=FALSE,
+         col="red", lwd=lwd, main="D(complete)")
+   traceplot(chdeviance[, "dev.observed"], smooth=FALSE,
+         col="red", lwd=lwd, main="D(observed)")
+   dev.off()
+ }

```


4.4.2 Posterior density estimates

R⇒ Posterior density estimates for the overall means of emergence (see Figure 7):

```
> if (RUN.ALLOUT){
+   postscript(paste(FIGKEEPDIR, "figTandmob10.ps", sep=""), width=7, height=10,
+             horizontal=FALSE)
+   par(mfrow=c(2, 3), bty="n")
+   densplot(chmixture[, "y.Mean.1"], show.obs=FALSE, col="darkblue",
+            main="E(Tooth 11)")
+   densplot(chmixture[, "y.Mean.2"], show.obs=FALSE, col="darkblue",
+            main="E(Tooth 12)")
+   densplot(chmixture[, "y.Mean.3"], show.obs=FALSE, col="darkblue",
+            main="E(Tooth 13)")
+   densplot(chmixture[, "y.Mean.4"], show.obs=FALSE, col="darkblue",
+            main="E(Tooth 14)")
+   densplot(chmixture[, "y.Mean.5"], show.obs=FALSE, col="darkblue",
+            main="E(Tooth 15)")
+   densplot(chmixture[, "y.Mean.6"], show.obs=FALSE, col="darkblue",
+            main="E(Tooth 16)")
+   dev.off()
+ }
```

R⇒ Posterior density estimates for the standard deviations of emergence (see Figure 8):

```
> if (RUN.ALLOUT){
+   postscript(paste(FIGKEEPDIR, "figTandmob11.ps", sep=""), width=7, height=10,
+             horizontal=FALSE)
+   par(mfrow=c(2, 3), bty="n")
+   densplot(chmixture[, "y.SD.1"], show.obs=FALSE, col="darkgreen",
+            main="SD(Tooth 11)")
+   densplot(chmixture[, "y.SD.2"], show.obs=FALSE, col="darkgreen",
+            main="SD(Tooth 12)")
+   densplot(chmixture[, "y.SD.3"], show.obs=FALSE, col="darkgreen",
+            main="SD(Tooth 13)")
+   densplot(chmixture[, "y.SD.4"], show.obs=FALSE, col="darkgreen",
+            main="SD(Tooth 14)")
+   densplot(chmixture[, "y.SD.5"], show.obs=FALSE, col="darkgreen",
+            main="SD(Tooth 15)")
+   densplot(chmixture[, "y.SD.6"], show.obs=FALSE, col="darkgreen",
+            main="SD(Tooth 16)")
+   dev.off()
+ }
```

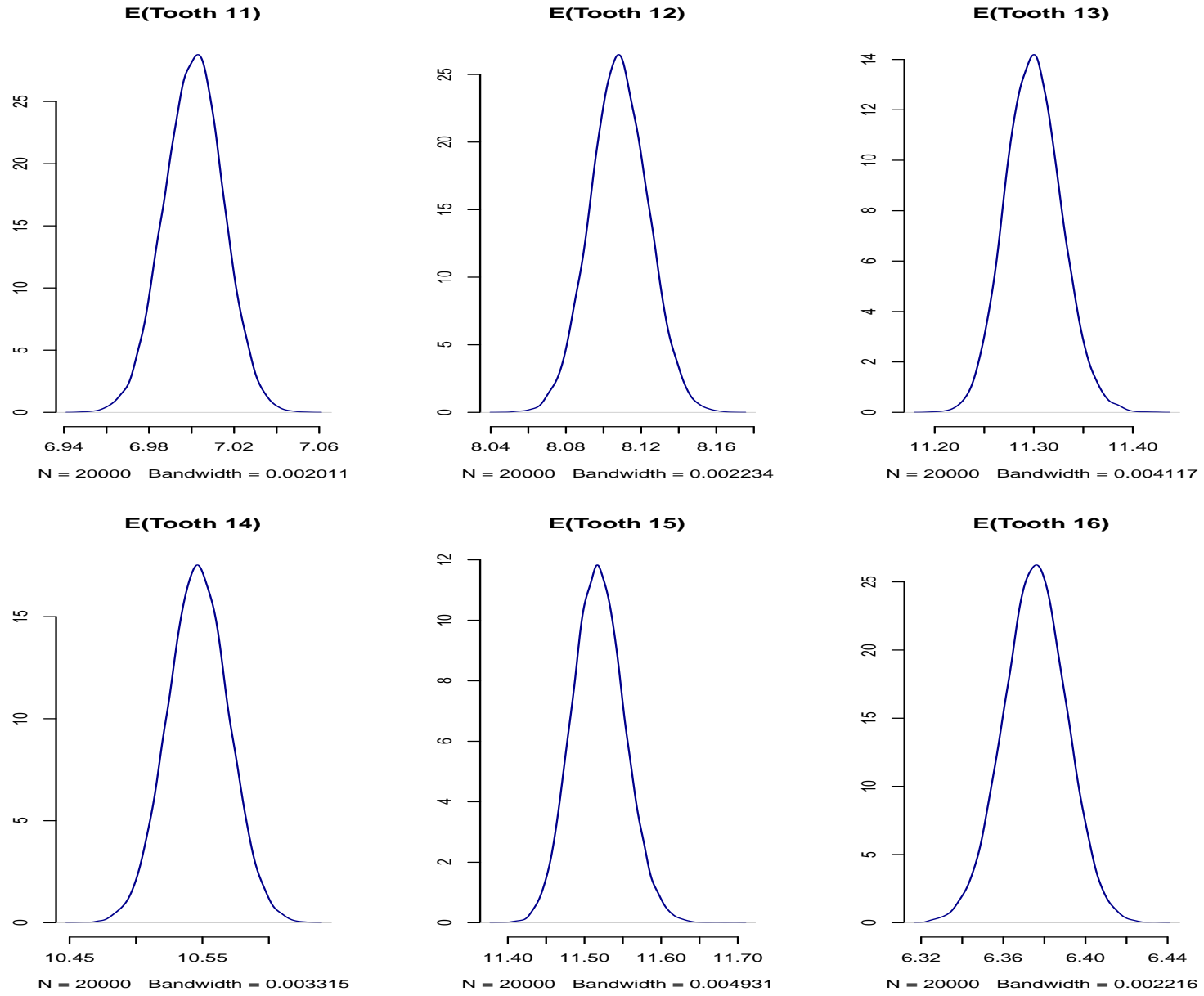


Figure 7: Model with two mixture components. Posterior density estimates for the means of emergence.

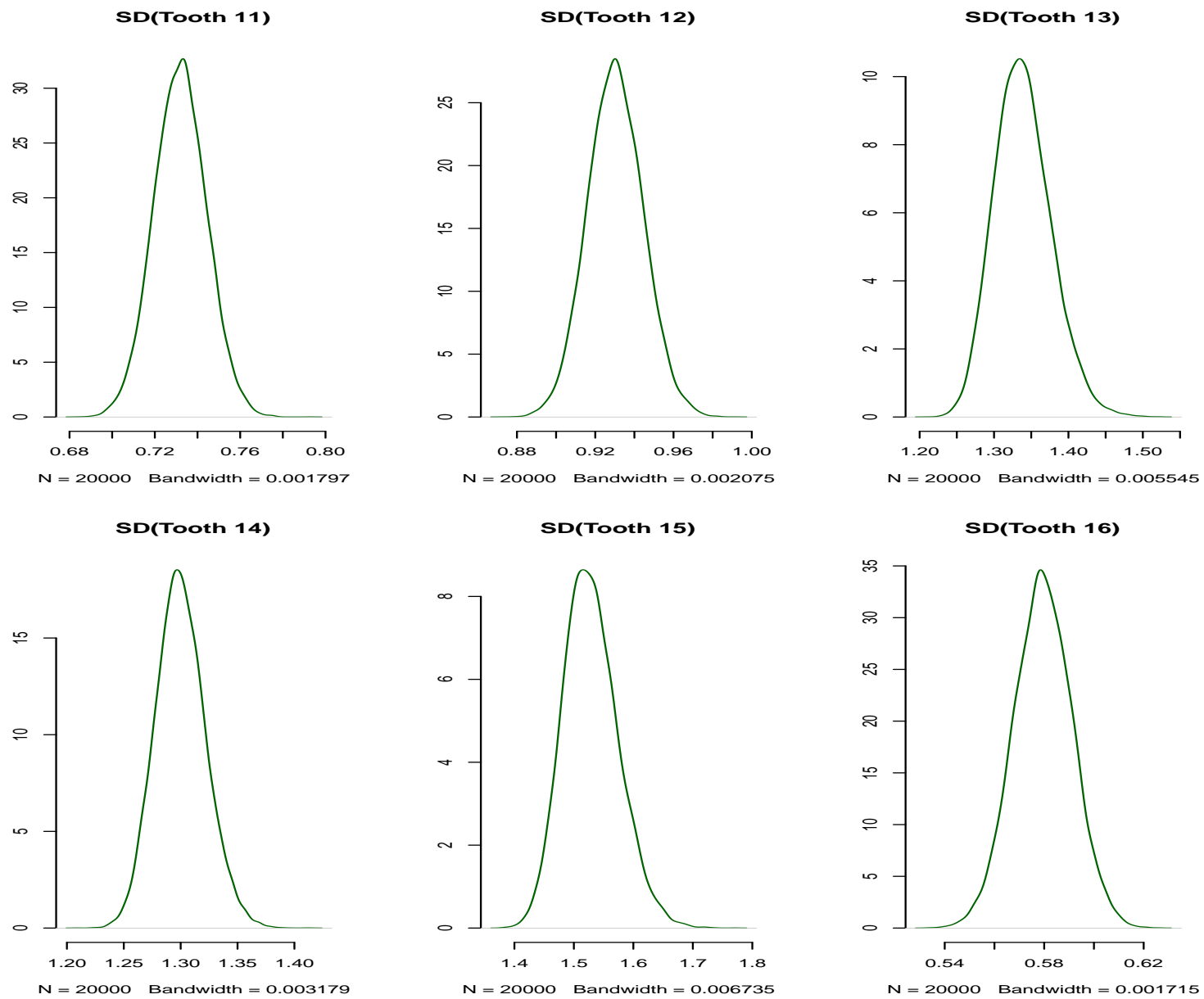


Figure 8: Model with two mixture components. Posterior density estimates for the standard deviations of emergence.

R⇒ Posterior density estimates for the pairwise correlations between the emergence times (see Figure 9):

```
> if (RUN.ALLOUT){
+   postscript(paste(FIGKEEPDIR, "figTandmob12.ps", sep=""), width=10, height=7,
+             horizontal=FALSE)
+   par(bty="n", mar=c(2, 1, 4, 0)+0.1)
+   layout(matrix(c(1,2,3,4,5, 0,6,7,8,9, 0,0,10,11,12,
+                 0,0,0,13,14, 0,0,0,0,15), nrow=5, byrow=TRUE))
+   densplot(chmixture[, "y.Corr.2.1"], show.obs=FALSE, col="darkgreen",
+            main="Corr 2-1")
+   densplot(chmixture[, "y.Corr.3.1"], show.obs=FALSE, col="darkgreen",
+            main="Corr 3-1")
+   densplot(chmixture[, "y.Corr.4.1"], show.obs=FALSE, col="darkgreen",
+            main="Corr 4-1")
+   densplot(chmixture[, "y.Corr.5.1"], show.obs=FALSE, col="darkgreen",
+            main="Corr 5-1")
+   densplot(chmixture[, "y.Corr.6.1"], show.obs=FALSE, col="darkgreen",
+            main="Corr 6-1")
+   densplot(chmixture[, "y.Corr.3.2"], show.obs=FALSE, col="darkgreen",
+            main="Corr 3-2")
+   densplot(chmixture[, "y.Corr.4.2"], show.obs=FALSE, col="darkgreen",
+            main="Corr 4-2")
+   densplot(chmixture[, "y.Corr.5.2"], show.obs=FALSE, col="darkgreen",
+            main="Corr 5-2")
+   densplot(chmixture[, "y.Corr.6.2"], show.obs=FALSE, col="darkgreen",
+            main="Corr 6-2")
+   densplot(chmixture[, "y.Corr.4.3"], show.obs=FALSE, col="darkgreen",
+            main="Corr 4-3")
+   densplot(chmixture[, "y.Corr.5.3"], show.obs=FALSE, col="darkgreen",
+            main="Corr 5-3")
+   densplot(chmixture[, "y.Corr.6.3"], show.obs=FALSE, col="darkgreen",
+            main="Corr 6-3")
+   densplot(chmixture[, "y.Corr.5.4"], show.obs=FALSE, col="darkgreen",
+            main="Corr 5-4")
+   densplot(chmixture[, "y.Corr.6.4"], show.obs=FALSE, col="darkgreen",
+            main="Corr 6-4")
+   densplot(chmixture[, "y.Corr.6.5"], show.obs=FALSE, col="darkgreen",
+            main="Corr 6-5")
+   dev.off()
+ }
```

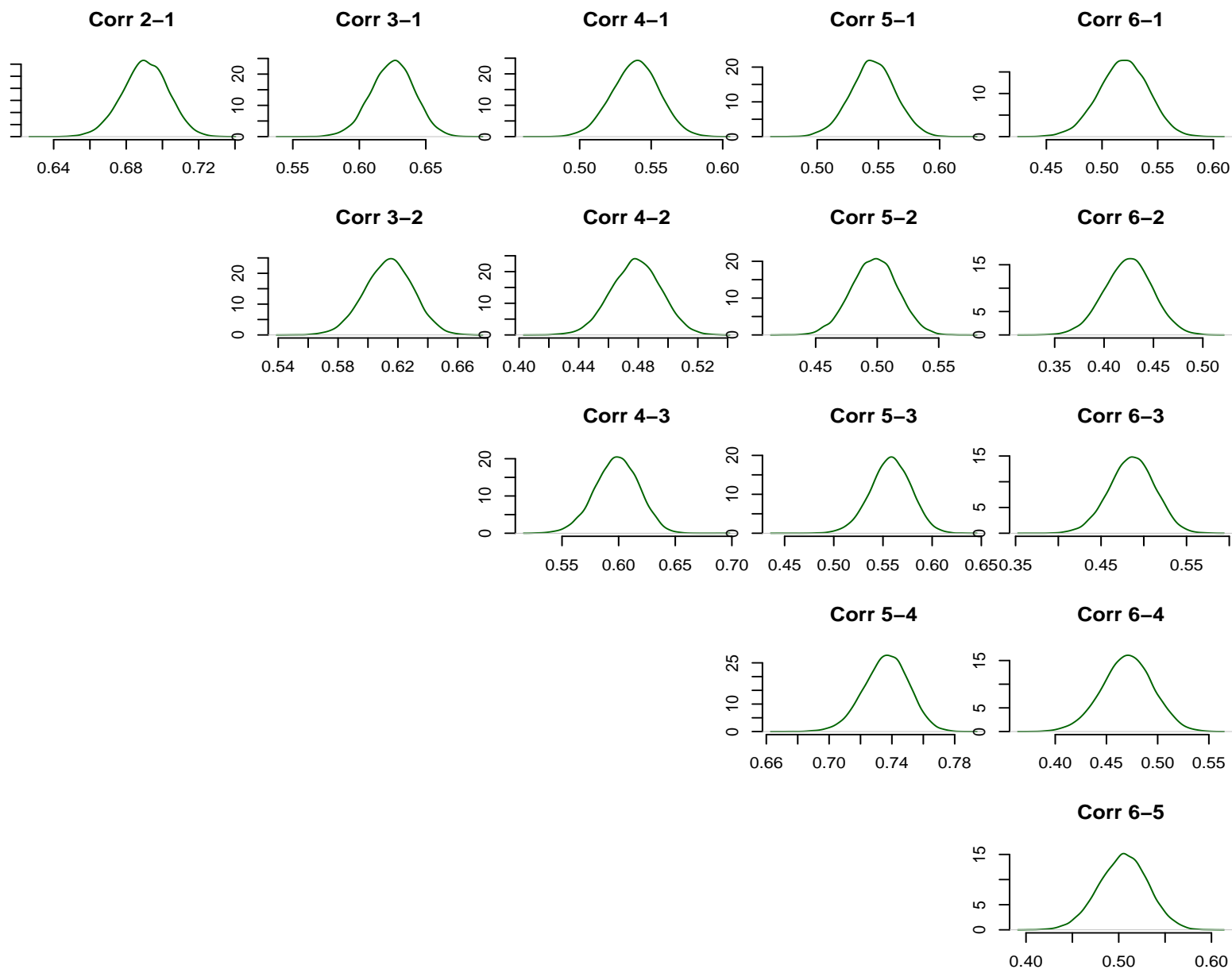


Figure 9: Model with two mixture components. Posterior density estimates for the pairwise correlations between the emergence times.

R⇒ Posterior density estimates for some other parameters (see Figure 10):

```
> if (RUN.ALLOUT){  
+   postscript(paste(FIGKEEPDIR, "figTandmob13.ps", sep=""), width=10, height=7,  
+             horizontal=FALSE)  
+   par(bty="n")  
+   layout(matrix(c(1,1,2,2,3,3, 0,4,4,5,5,0),  
+                 nrow=2, byrow=TRUE))  
+   densplot(chgammaInv[, "gammaInv1"], show.obs=FALSE,  
+            col="brown", main="gamma^{-1}")  
+   densplot(chdeviance[, "LogL0"], show.obs=FALSE,  
+            col="red", main="Log(L0)")  
+   densplot(chdeviance[, "LogL1"], show.obs=FALSE,  
+            col="red", main="Log(L1)")  
+   densplot(chdeviance[, "dev.complete"], show.obs=FALSE,  
+            col="red", main="D(complete)")  
+   densplot(chdeviance[, "dev.observed"], show.obs=FALSE,  
+            col="red", main="D(observed)")  
+   dev.off()  
+ }
```

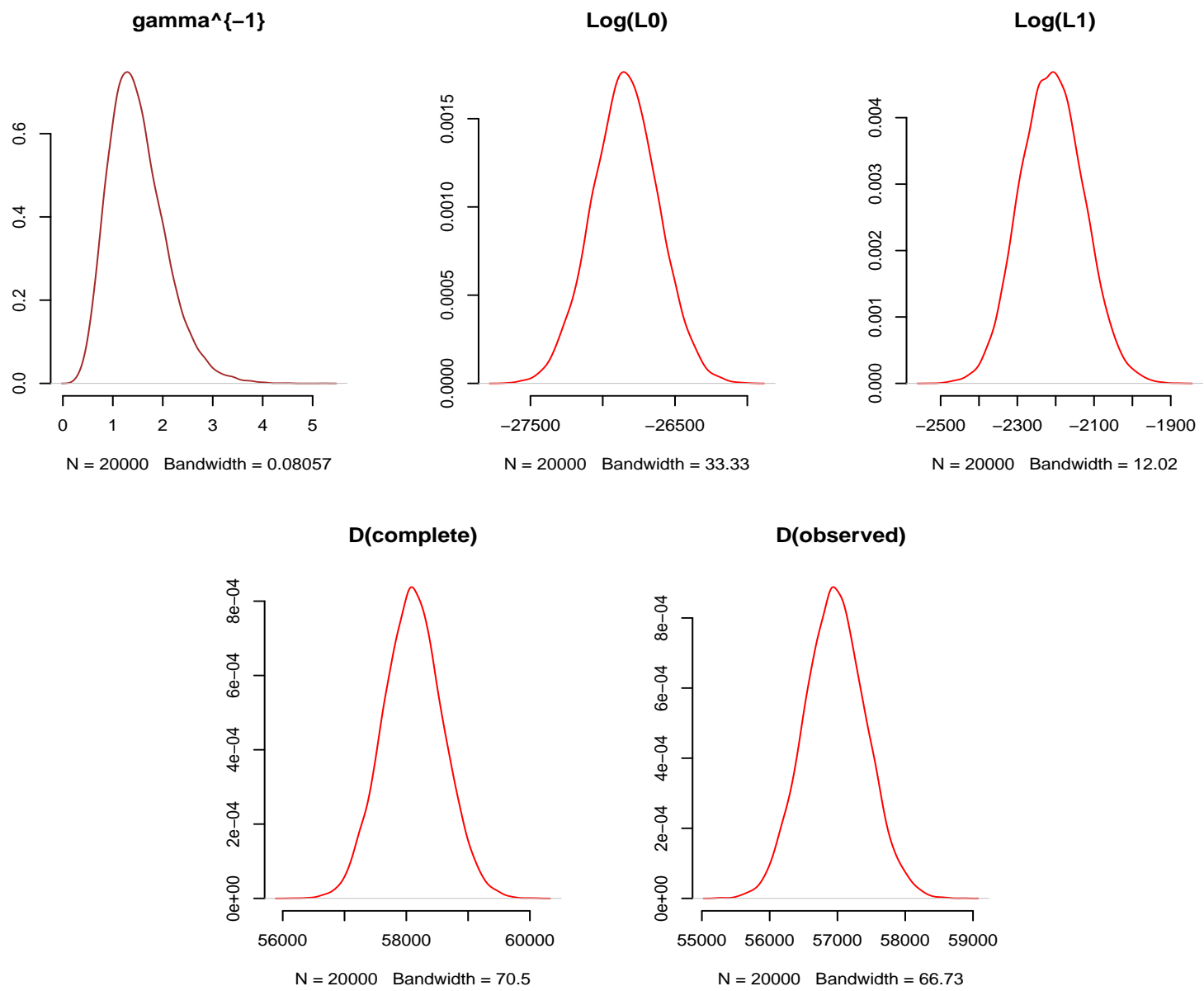


Figure 10: Model with two mixture components. Posterior density estimates for some other parameters.

4.4.3 Autocorrelation plots

R⇒ Autocorrelation plots for the overall means of emergence (see Figure 11):

```
> if (RUN.ALLOUT){  
+   postscript(paste(FIGKEEPDIR, "figTandmob14.ps", sep=""), width=7, height=10,  
+             horizontal=FALSE)  
+   par(mfrow=c(2, 3), bty="n")  
+   autocorr.plot(chmixture[, "y.Mean.1"], auto.layout=FALSE, ask=FALSE,  
+                col="darkblue", main="E(Tooth 11)")  
+   autocorr.plot(chmixture[, "y.Mean.2"], auto.layout=FALSE, ask=FALSE,  
+                col="darkblue", main="E(Tooth 12)")  
+   autocorr.plot(chmixture[, "y.Mean.3"], auto.layout=FALSE, ask=FALSE,  
+                col="darkblue", main="E(Tooth 13)")  
+   autocorr.plot(chmixture[, "y.Mean.4"], auto.layout=FALSE, ask=FALSE,  
+                col="darkblue", main="E(Tooth 14)")  
+   autocorr.plot(chmixture[, "y.Mean.5"], auto.layout=FALSE, ask=FALSE,  
+                col="darkblue", main="E(Tooth 15)")  
+   autocorr.plot(chmixture[, "y.Mean.6"], auto.layout=FALSE, ask=FALSE,  
+                col="darkblue", main="E(Tooth 16)")  
+   dev.off()  
+ }
```

R⇒ Autocorrelation plots for the standard deviations of emergence (see Figure 12):

```
> if (RUN.ALLOUT){  
+   postscript(paste(FIGKEEPDIR, "figTandmob15.ps", sep=""), width=7, height=10,  
+             horizontal=FALSE)  
+   par(mfrow=c(2, 3), bty="n")  
+   autocorr.plot(chmixture[, "y.SD.1"], auto.layout=FALSE, ask=FALSE,  
+                col="darkgreen", main="SD(Tooth 11)")  
+   autocorr.plot(chmixture[, "y.SD.2"], auto.layout=FALSE, ask=FALSE,  
+                col="darkgreen", main="SD(Tooth 12)")  
+   autocorr.plot(chmixture[, "y.SD.3"], auto.layout=FALSE, ask=FALSE,  
+                col="darkgreen", main="SD(Tooth 13)")  
+   autocorr.plot(chmixture[, "y.SD.4"], auto.layout=FALSE, ask=FALSE,  
+                col="darkgreen", main="SD(Tooth 14)")  
+   autocorr.plot(chmixture[, "y.SD.5"], auto.layout=FALSE, ask=FALSE,  
+                col="darkgreen", main="SD(Tooth 15)")  
+   autocorr.plot(chmixture[, "y.SD.6"], auto.layout=FALSE, ask=FALSE,  
+                col="darkgreen", main="SD(Tooth 16)")  
+   dev.off()  
+ }
```

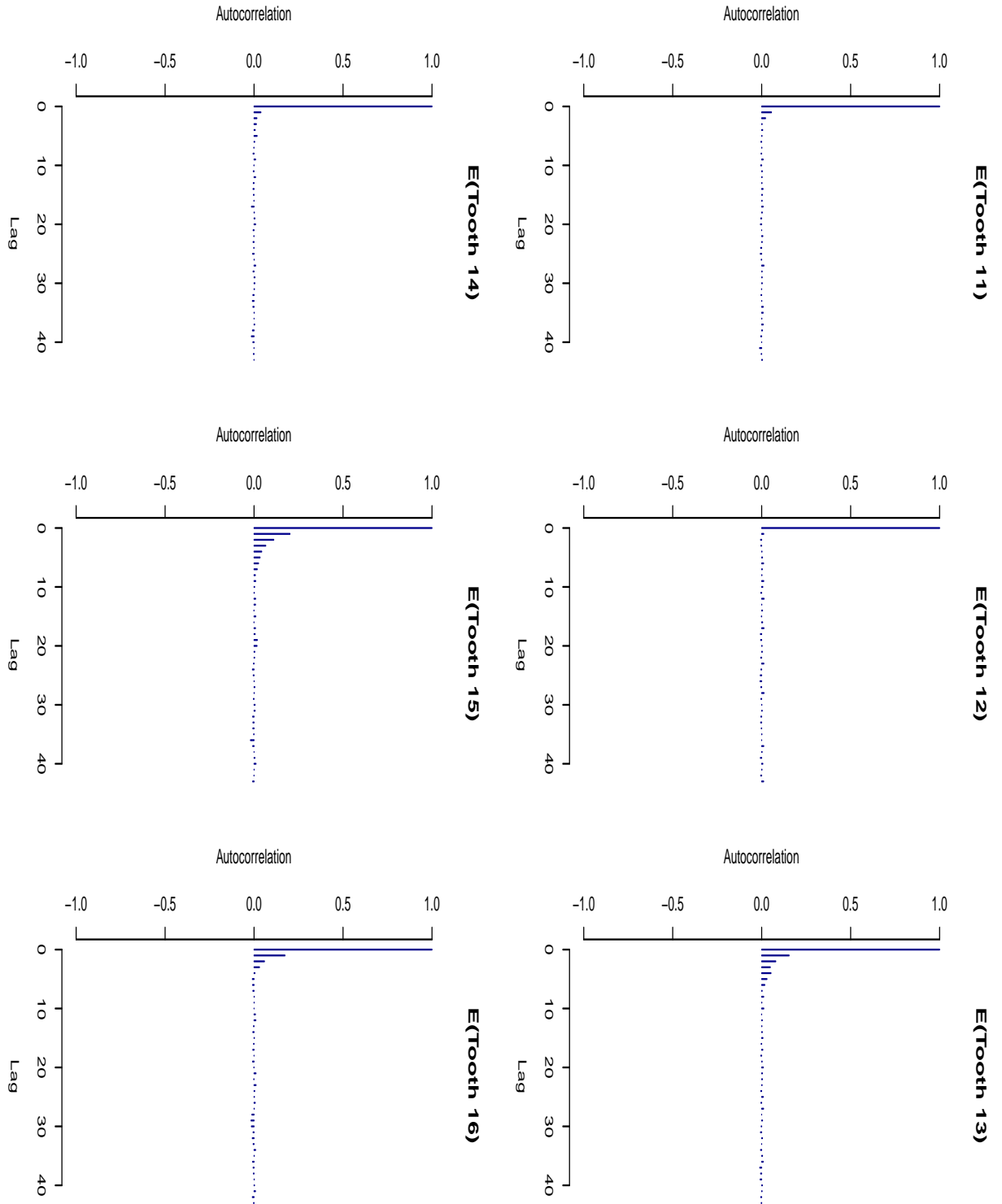



Figure 11: Model with two mixture components. Autocorrelation plots for the means of emergence.

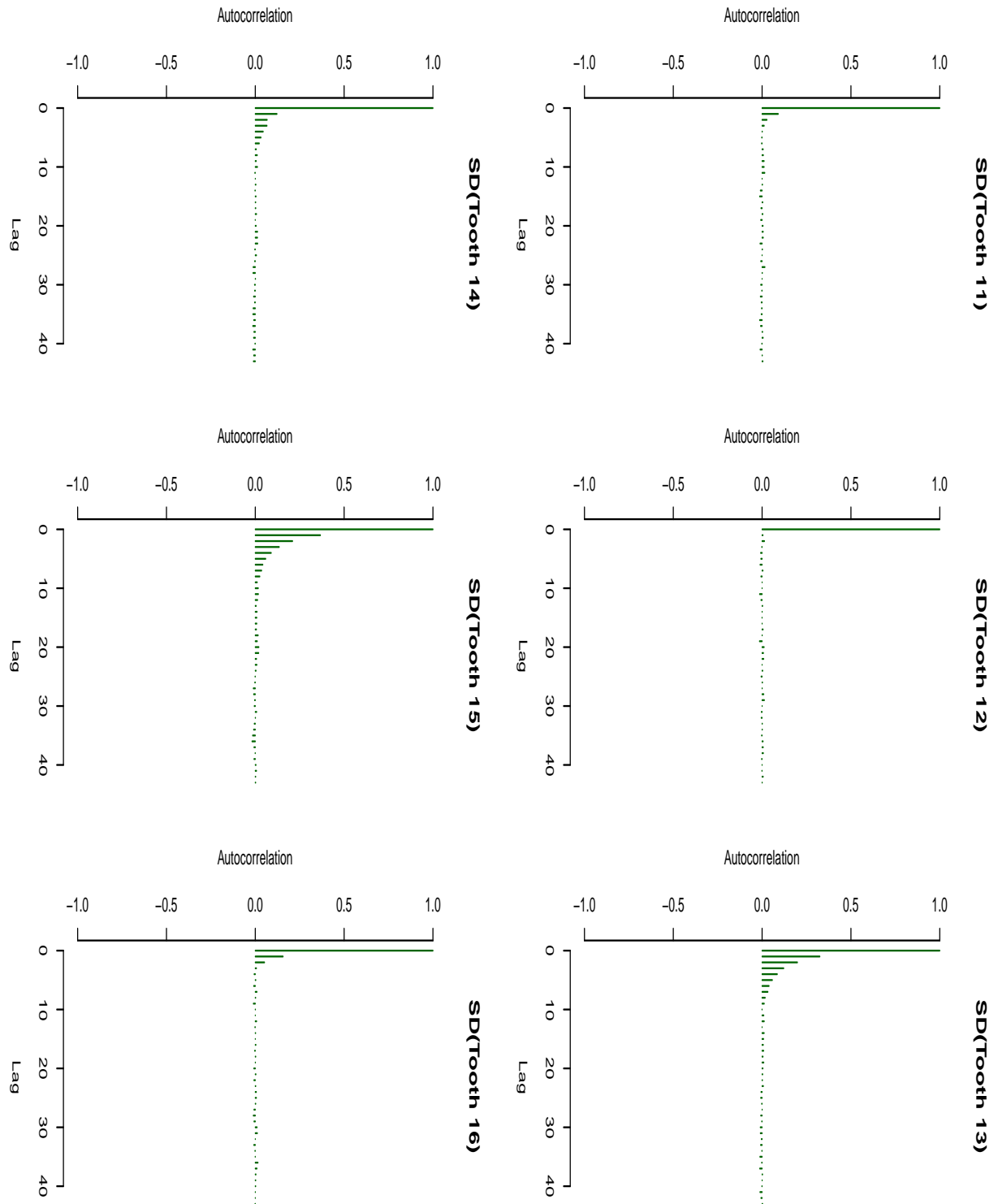


Figure 12: Model with two mixture components. Autocorrelation plots for the standard deviations of emergence.

R⇒ Autocorrelation plots for the pairwise correlations between the emergence times (see Figure 13):

```
> if (RUN.ALLOUT){
+   postscript(paste(FIGKEEPDIR, "figTandmob16.ps", sep=""), width=10, height=7,
+             horizontal=FALSE)
+   par(bty="n", mar=c(1, 1, 4, 0)+0.1)
+   layout(matrix(c(1,2,3,4,5, 0,6,7,8,9, 0,0,10,11,12,
+                   0,0,0,13,14, 0,0,0,0,15), nrow=5, byrow=TRUE))
+   autocorr.plot(chmixture[, "y.Corr.2.1"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 2-1")
+   autocorr.plot(chmixture[, "y.Corr.3.1"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 3-1")
+   autocorr.plot(chmixture[, "y.Corr.4.1"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 4-1")
+   autocorr.plot(chmixture[, "y.Corr.5.1"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 5-1")
+   autocorr.plot(chmixture[, "y.Corr.6.1"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 6-1")
+   autocorr.plot(chmixture[, "y.Corr.3.2"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 3-2")
+   autocorr.plot(chmixture[, "y.Corr.4.2"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 4-2")
+   autocorr.plot(chmixture[, "y.Corr.5.2"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 5-2")
+   autocorr.plot(chmixture[, "y.Corr.6.2"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 6-2")
+   autocorr.plot(chmixture[, "y.Corr.4.3"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 4-3")
+   autocorr.plot(chmixture[, "y.Corr.5.3"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 5-3")
+   autocorr.plot(chmixture[, "y.Corr.6.3"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 6-3")
+   autocorr.plot(chmixture[, "y.Corr.5.4"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 5-4")
+   autocorr.plot(chmixture[, "y.Corr.6.4"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 6-4")
+   autocorr.plot(chmixture[, "y.Corr.6.5"], auto.layout=FALSE,
+                 ask=FALSE, col="darkgreen", main="Corr 6-5")
+   dev.off()
+ }
```

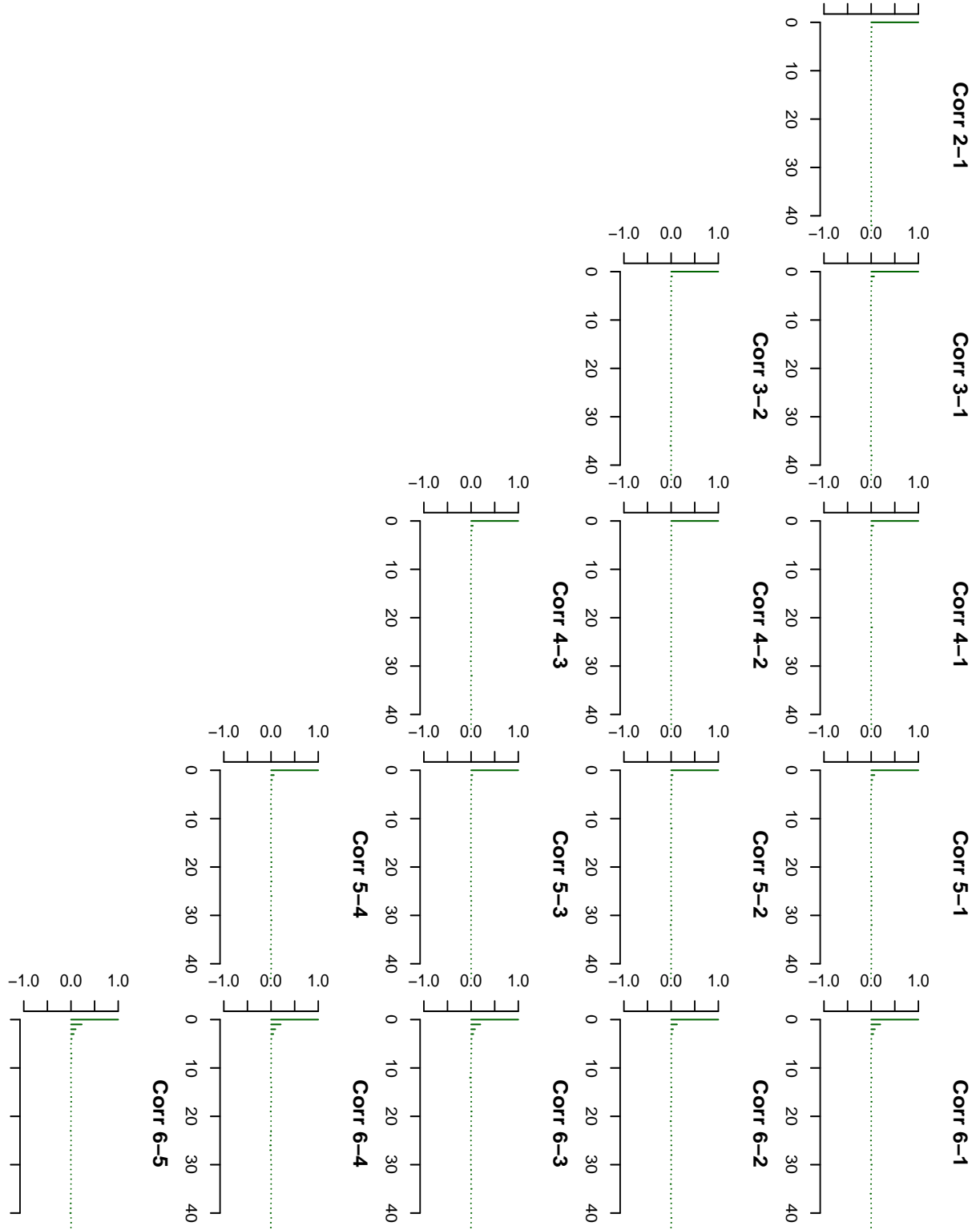


Figure 13: Model with two mixture components. Autocorrelation plots for the pairwise correlations between the emergence times.

R⇒ Autocorrelation plots for some other parameters (see Figure 14):

```
> if (RUN.ALLOUT){  
+   postscript(paste(FIGKEEPDIR, "figTandmob17.ps", sep=""), width=10, height=7,  
+             horizontal=FALSE)  
+   par(bty="n")  
+   layout(matrix(c(1,1,2,2,3,3, 0,4,4,5,5,0), nrow=2, byrow=TRUE))  
+   autocorr.plot(chgammaInv[, "gammaInv1"], auto.layout=FALSE,  
+                ask=FALSE, col="brown", main="gamma^{-1}")  
+   autocorr.plot(chdeviance[, "LogL0"], auto.layout=FALSE,  
+                ask=FALSE, col="red", main="Log(L0)")  
+   autocorr.plot(chdeviance[, "LogL1"], auto.layout=FALSE,  
+                ask=FALSE, col="red", main="Log(L1)")  
+   autocorr.plot(chdeviance[, "dev.complete"], auto.layout=FALSE,  
+                ask=FALSE, col="red", main="D(complete)")  
+   autocorr.plot(chdeviance[, "dev.observed"], auto.layout=FALSE,  
+                ask=FALSE, col="red", main="D(observed)")  
+   dev.off()  
+ }
```

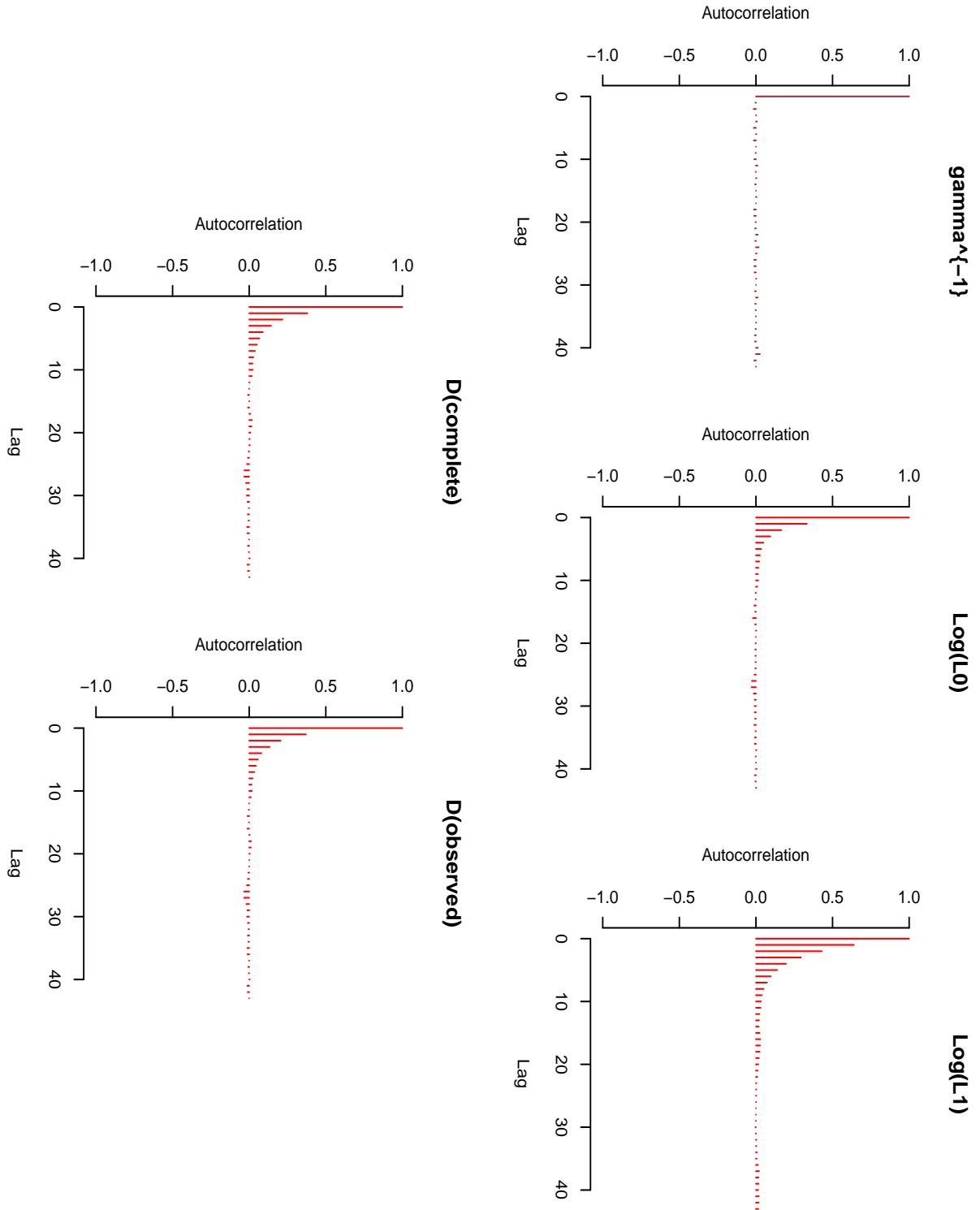


Figure 14: Model with two mixture components. Autocorrelation plots for some other parameters.

5 Models with different fixed numbers of components

In this section, we will fit a mixture model for $K = 1, \dots, 10$, compare the deviance based quantities and predictive densities.

R⇒ Running the MCMC simulation for $K = 1, \dots, 10$ (output printed during the MCMC run on the screen not shown), computation of predictive densities.

R⇒ After predictive densities are computed, we remove all chains from resulting objects (to save some memory).

```
> Seed <- c(770328, 770328, 770328, 770328, 770328,
+          770328, 780830, 761014, 770328, 770328)
> Keep <- c("iter", "nMCMC", "dim", "prior", "init", "RJMCMC",
+          "scale", "freqK", "propK", "DIC", "moves",
+          "summ.y.Mean", "summ.y.SDCorr",
+          "summ.z.Mean", "summ.z.SDCorr")
> if (RUN.TIMECONSUMING.CODE){
+   ModelK <- list()
+   PDensUniModelK <- list()
+   PDensBiModelK <- list()
+   for (K in 1:10){
+     set.seed(Seed[K])
+     cat(paste("K = ", K, "\n-----\n", sep=""))
+     cat("Seed is", Seed[K], "\n")
+     Prior0 <- list(priorK="fixed", Kmax=K)
+     Model0 <- NMixMCMC(y0=y0, y1=y1, censor=censor, prior=Prior0,
+                       nMCMC=nMCMC, scale=list(shift=0, scale=1), PED=TRUE)
+
+     cat(paste("\nComputation of marginal pred. densities started on ", date(),
+               "\n", sep=""))
+     PDensUniModelK[[k]] <- list()
+     PDensUniModelK[[k]][[1]] <- NMixPredDensMarg(Model0[[1]], grid=ygridUni)
+     PDensUniModelK[[k]][[2]] <- NMixPredDensMarg(Model0[[2]], grid=ygridUni)
+     cat(paste("Computation of marginal pred. densities finished on ", date(),
+               "\n\n", sep=""))
+
+     cat(paste("Computation of joint pred. densities started on ", date(),
+               "\n", sep=""))
+     PDensBiModelK[[k]] <- list()
+     PDensBiModelK[[k]][[1]] <- NMixPredDensJoint2(Model0[[1]], grid=ygridBi)
+     PDensBiModelK[[k]][[2]] <- NMixPredDensJoint2(Model0[[2]], grid=ygridBi)
+     cat(paste("Computation of joint pred. densities finished on ", date(),
+               "\n\n\n", sep=""))
+
+     ModelK[[k]][[1]] <- Model0[[1]][Keep]
+     ModelK[[k]][[2]] <- Model0[[2]][Keep]
+     class(ModelK[[k]][[1]]) <- class(ModelK[[k]][[2]]) <- "NMixMCMC"
+   }
+ }
```

```
+ rm(list="Model0")
+ }
+ }
```

R⇒ Summary of PED and DIC's for the fitted models:

```
> PED <- ModelK[[1]]$PED
> DIC <- list(Chain1=ModelK[[1]][[1]]$DIC, Chain2=ModelK[[1]][[2]]$DIC)
> for (k in 2:length(ModelK)){
+   PED <- rbind(PED, ModelK[[k]]$PED)
+   DIC[[1]] <- rbind(DIC[[1]], ModelK[[k]][[1]]$DIC)
+   DIC[[2]] <- rbind(DIC[[2]], ModelK[[k]][[2]]$DIC)
+ }
> rownames(PED) <- paste("K=", 1:length(ModelK), sep="")
> rownames(DIC[[1]]) <- rownames(DIC[[2]]) <- paste("K=", 1:length(ModelK), sep="")

> print(PED)
```

	D.expect	p(opt)	PED	wp(opt)	wPED
K=1	78695.46	157.6598	78853.12	157.6598	78853.12
K=2	74778.22	459.2338	75237.46	459.2338	75237.46
K=3	74923.44	934.1437	75857.58	934.1437	75857.58
K=4	75437.12	1235.8121	76672.94	1235.8121	76672.94
K=5	75632.71	1722.6202	77355.33	1722.6200	77355.33
K=6	76357.36	3171.4515	79528.81	3171.4518	79528.81
K=7	76269.30	3717.2918	79986.59	3717.2861	79986.59
K=8	76476.79	5059.1473	81535.94	5059.1495	81535.94
K=9	76668.48	5614.8044	82283.29	5614.8060	82283.29
K=10	76683.01	6620.5835	83303.59	6620.5875	83303.60

```
> print(DIC)
```

```
$Chain1
      DIC      pD    D.bar D.in.bar
K=1  66735.18 4699.580 62035.60 57336.02
K=2  64062.40 7098.499 56963.90 49865.40
K=3  64041.76 7190.355 56851.40 49661.05
K=4  63212.65 7061.547 56151.10 49089.55
K=5  62768.35 6957.968 55810.39 48852.42
K=6  64483.55 8535.527 55948.02 47412.50
K=7  65167.36 9111.387 56055.97 46944.59
K=8  63117.53 7186.084 55931.44 48745.36
K=9  62986.51 7055.850 55930.66 48874.82
K=10 64409.17 8468.477 55940.69 47472.21
```


\$Chain2

	DIC	pD	D.bar	D.in.bar
K=1	66733.05	4699.827	62033.22	57333.39
K=2	64054.82	7096.818	56958.00	49861.18
K=3	64018.44	7184.664	56833.78	49649.12
K=4	63207.57	7060.128	56147.44	49087.32
K=5	62883.15	6994.865	55888.28	48893.42
K=6	64058.40	8111.954	55946.44	47834.49
K=7	62945.71	7034.567	55911.14	48876.58
K=8	64873.65	8877.754	55995.89	47118.14
K=9	63359.93	7463.778	55896.15	48432.37
K=10	63395.95	7303.974	56091.97	48788.00

R⇒ Plots of the (univariate) predictive densities for different values of K are shown in Figures 15–17:

```
> if (RUN.ALLOUT){
+   CH <- 1
+   for (fig in 1:3){
+     postscript(paste(FIGKEEPDIR, "figTandmob", 17+fig, ".ps", sep=""),
+               width=7, height=10, horizontal=FALSE)
+     par(mfrow=c(2, 1), bty="n")
+     for (tt in (2*(fig-1)+1):(2*fig)){
+       plot(PDensUniModelK[[Kshow]][[CH]]$x[[tt]],
+            PDensUniModelK[[Kshow]][[CH]]$dens[[tt]],
+            type="n", xlab="Age (years)", ylab="Density",
+            main=paste("Tooth ", 10+tt, sep=""))
+       for (k in 1:10){
+         lines(PDensUniModelK[[k]][[CH]]$x[[tt]],
+               PDensUniModelK[[k]][[CH]]$dens[[tt]],
+               col="blue")
+       }
+       lines(PDensUniModelK[[Kshow]][[CH]]$x[[tt]],
+             PDensUniModelK[[Kshow]][[CH]]$dens[[tt]],
+             col="red", lwd=2)
+     }
+     dev.off()
+   }
+ }
```

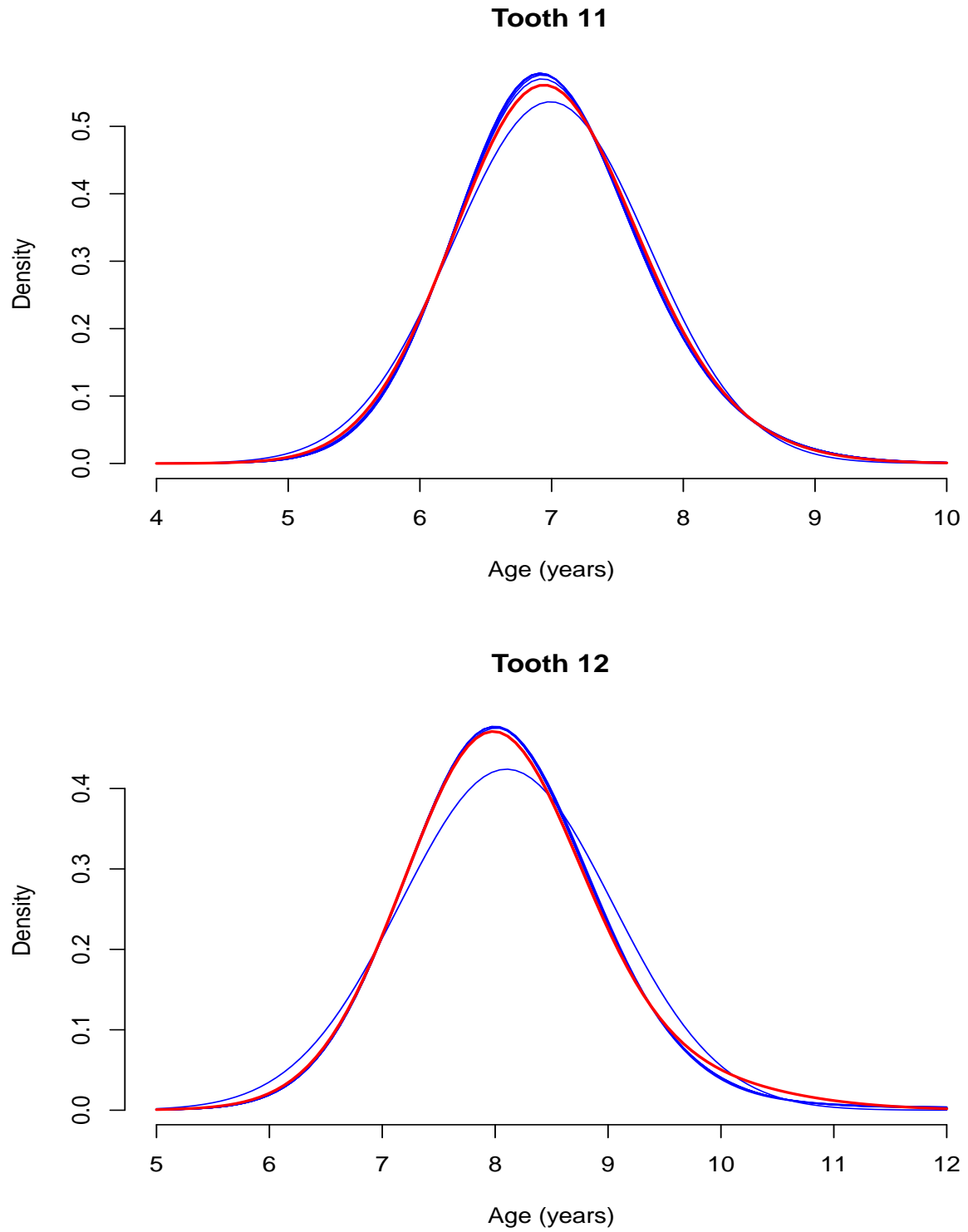


Figure 15: Teeth 11 and 12. Estimated predictive densities of emergence for different values of K ranging from 1 to 10 (blue lines) and for $K = 2$ (red line).

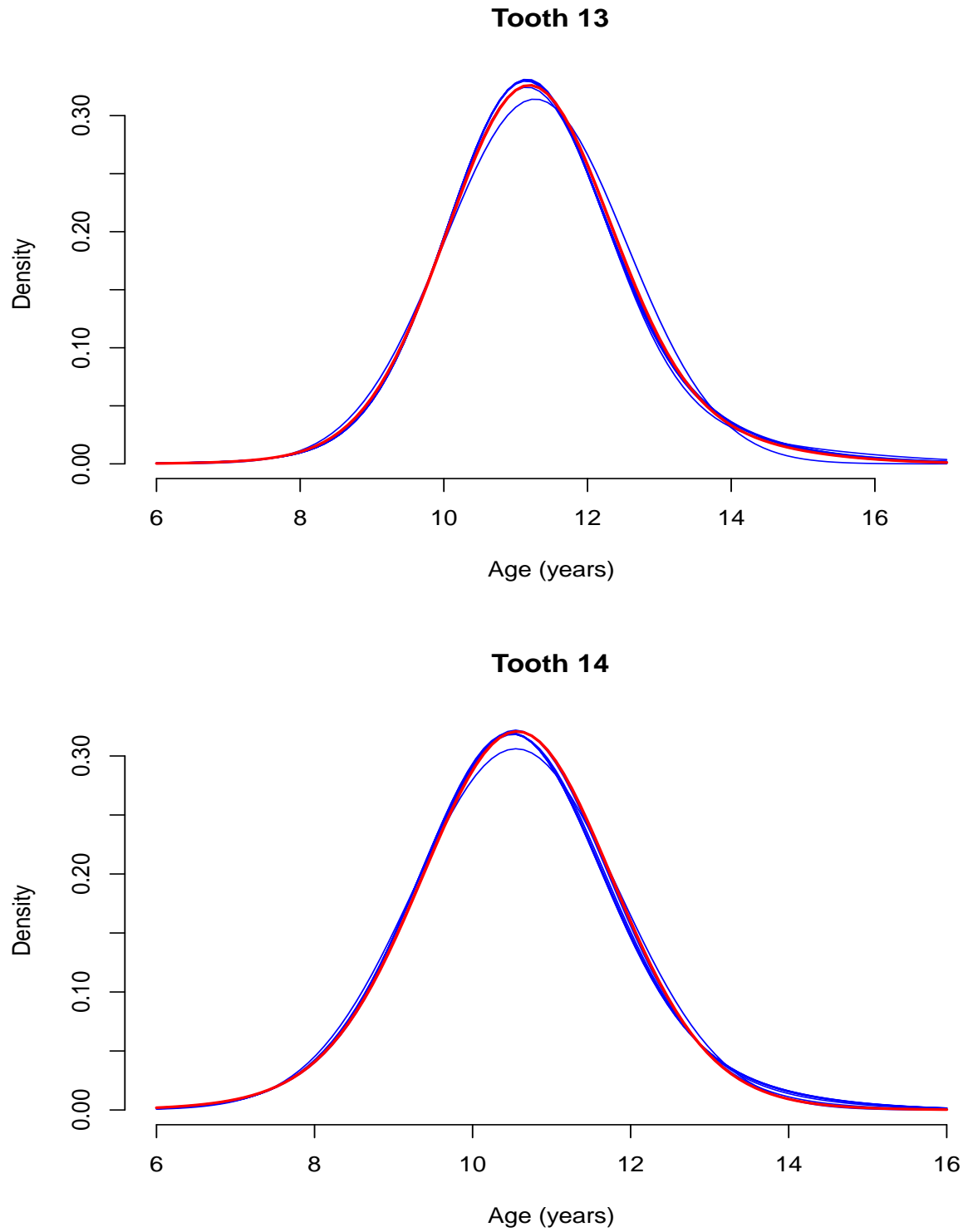


Figure 16: Teeth 13 and 14. Estimated predictive densities of emergence for different values of K ranging from 1 to 10 (blue lines) and for $K = 2$ (red line).

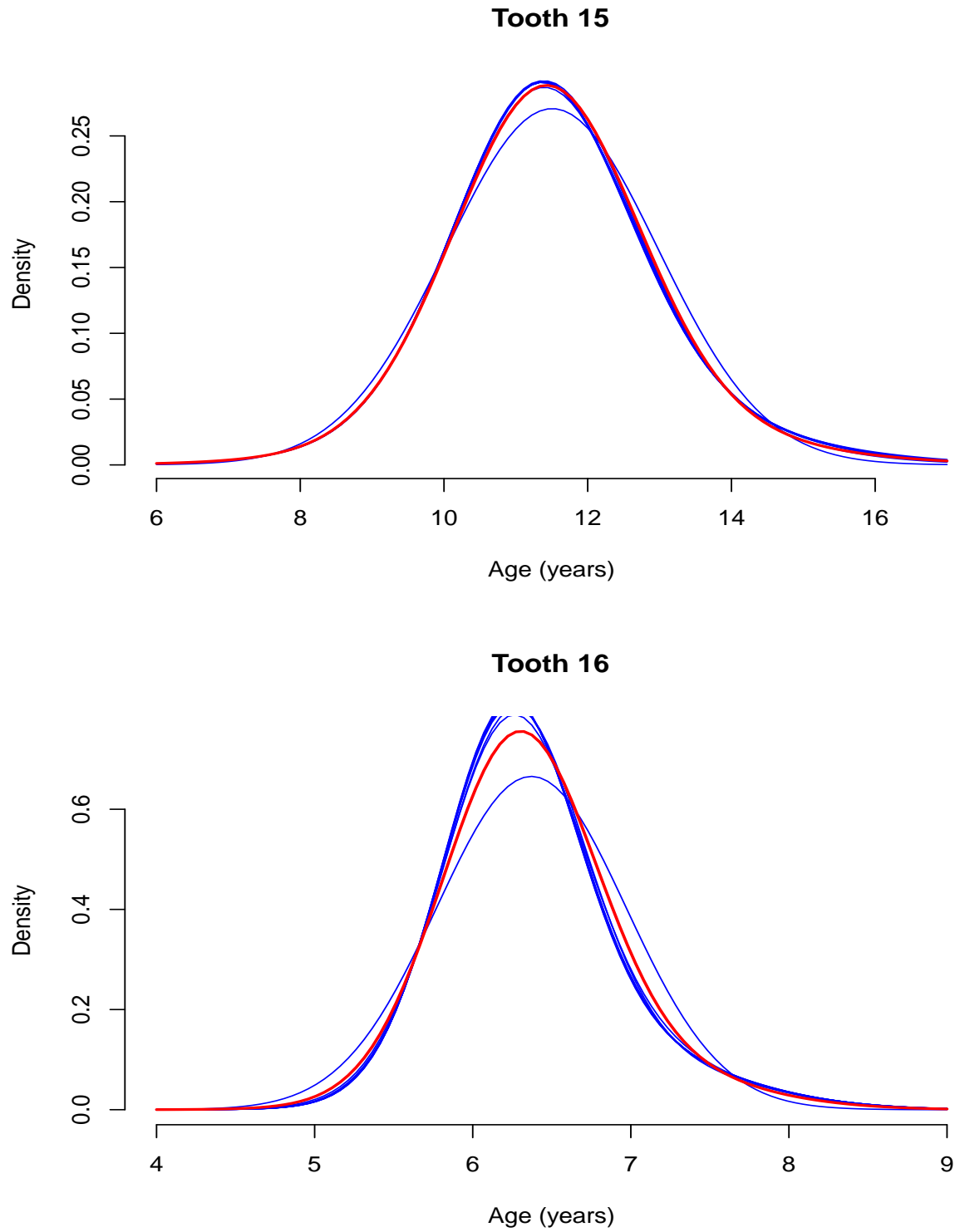


Figure 17: Teeth 15 and 16. Estimated predictive densities of emergence for different values of K ranging from 1 to 10 (blue lines) and for $K = 2$ (red line).

R⇒ Save results for future use:

```
> if (RUN.TIMECONSUMING.CODE){  
+   save(list="Model0",  
+       file=paste(RESULTDIR, "/Tandmob-Model0", Kshow, ".RData", sep=""))  
+  
+   PDensBiModelK <- list()  
+   PDensBiModelK[[Kshow]] <- PDensBiModel0  
+   save(list="PDensBiModelK",  
+       file=paste(RESULT2DIR, "/Tandmob-PDensBiModelK0", Kshow, ".RData", sep=""))  
+  
+   save(list=c("ModelK", "PDensUniModelK", "PDensBiModelK"),  
+       file=paste(RESULT2DIR, "/Tandmob-Result.RData", sep=""))  
+ }
```

References

- EKSTRAND, K. R., CHRISTIANSEN, J., and CHRISTIANSEN, M. E. (2003). Time and duration of eruption of first and second permanent molars: a longitudinal investigation. *Community Dentistry and Oral Epidemiology*, **31**, 344–350.
- KOMÁREK, A. (2009). A new R package for Bayesian estimation of multivariate normal mixtures allowing for selection of the number of components and interval-censored data. *Computational Statistics and Data Analysis*, **53**, To appear.
- VANOBBERGEN, J., MARTENS, L., LESAFFRE, E., and DECLERCK, D. (2000). The Signal-Tandmobiel[®] project – a longitudinal intervention health promotion study in Flanders (Belgium): baseline and first year results. *European Journal of Paediatric Dentistry*, **2**, 87–96.