Chapter 4

Tutorial: R package `lori`

Contents

4.1 Simulated example 59
4.1.1 Generating simulated data 60
4.1.2 Estimation and single imputation 61
4.1.3 Multiple imputation 63
4.2 Analysis of the Aravo data set 65
4.2.1 Loading the data 65
4.2.2 Estimation of the lori model 66
4.2.3 Bootstrap and intervals of variability 66

The R package `lori` (https://CRAN.R-project.org/package=lori) contains methods for the analysis, single imputation, and multiple imputation of incomplete count data with side information. The implemented functions correspond to the methods described in Chapters 2 and 3, but this tutorial can be read and used independently: the models and procedures are recalled as concisely as possible. In this chapter, we display the main functionalities of the `lori` package, namely a single imputation procedure–which also returns estimates of main effects and interactions–a cross-validation function to select the regularization parameters, and a function to perform multiple imputation. We also provide reusable code using a simulated toy example.

4.1 Simulated example

Before going on with the tutorial, here are the necessary commands to install and load the package `lori`:

```
install.packages("lori")
library(lori)
```

To obtain the exact results that are presented here, you may set the following seed:

```
set.seed(123)
```

Consider an abundance table $\boldsymbol{Y} \in \mathbb{N}^{m_1 \times m_2}$, with missing values. Assume that three other tables are available: $\boldsymbol{R} \in \mathbb{R}^{m_1 \times K_1}$, containing covariates about the rows of $\boldsymbol{Y}$ (geographical information for instance), $\boldsymbol{C} \in \mathbb{R}^{m_2 \times K_2}$, containing covariates about the columns of $\boldsymbol{Y}$ (yearly meteorological indices), and $\boldsymbol{E} \in \mathbb{R}^{m_1 m_2 \times K_3}$ containing covariates about the row-column pairs (yearly meteorological information at the sites' scale, yearly economical indices of the sites' country, etc.). For $i \in [\![m_1]\!]$, the vector

$\boldsymbol{R}_{i,\cdot} = (\boldsymbol{R}_{i,1}, \ldots, \boldsymbol{R}_{i,K_1})$ contains all the information about the row i. Similarly, for $j \in [\![m_2]\!]$, the vector $\boldsymbol{C}_{j,\cdot} = (\boldsymbol{C}_{i,1}, \ldots, \boldsymbol{C}_{j,K_1})$ contains all the information about the column j. Finally, the vector $\boldsymbol{E}_{(j-1)m_1+i,\cdot} = (\boldsymbol{E}_{(j-1)m_1+i,1}, \ldots, \boldsymbol{E}_{(j-1)m_1+i,K_3})$ contains all the information about the row-column pair (i, j). The model implemented in the `lori` package is the following log-linear model:

$$\boldsymbol{Y}_{i,j} \sim \mathcal{P}(\exp(\boldsymbol{X}^0_{i,j})), \quad \boldsymbol{X}^0_{i,j} = \alpha^0_i + \beta^0_j + \sum_{k=1}^{K_1+K_2} \boldsymbol{U}_{(j-1)m_1+i,k}\epsilon^0_k + \boldsymbol{\Theta}^0_{i,j}, \tag{4.1}$$

where $\boldsymbol{U} \in \mathbb{R}^{m_1 m_2 \times (K_1+K_2+K_3)}$ denotes the covariate matrix whose rows are defined by: $\boldsymbol{U}_{(j-1)m_1+i,\cdot} = (\boldsymbol{R}_{i,\cdot}, \boldsymbol{C}_{j,\cdot}, \boldsymbol{E}_{(j-1)m_1+i,\cdot})$. Note that options are implemented to handle the cases where either singleton or pair of the matrices $(\boldsymbol{R}, \boldsymbol{C}, \boldsymbol{E})$ are available; row and column effects can also be removed optionally: this will be detailed later on.

4.1.1 Generating simulated data

Consider the following simulated example.

```
## covariates
m1 <- 30 # number of rows
m2 <- 10 # number of columns
K1 <- 2 # number of row covariates
K2 <- 2 # number of column covariates
K3 <- 3 # number of (rowxcolumn) covariates
q <- K1+K2+K3
R <- matrix(rnorm(m1*K1), nrow=m1) # matrix of row covariates
C <- matrix(rnorm(m2*K2), nrow=m2) # matrix of column
   covariates
E <- matrix(rnorm(m1*m2*K3), nrow=m1*m2) # matrix of  (
   rowxcolumn) covariates
```

To compute the matrix $\boldsymbol{U}$, a simple command is available in the `lori` package:

```
U <- covmat(m1, m2, R, C, E)
```

Note that the function also supports creating $\boldsymbol{U}$ from any singleton or pair of the matrices $(\boldsymbol{R}, \boldsymbol{C}, \boldsymbol{E})$, provided that the arguments are in the correct order. In other words, all the commands below may also be used:

```
U <- covmat(m1, m2, R)
U <- covmat(m1, m2, C)
U <- covmat(m1, m2, R, C)
U <- covmat(m1, m2, R=R, E=E)
U <- covmat(m1, m2, C=C, E=E)
```

An important practical point is that the rows of $\boldsymbol{R}$, $\boldsymbol{C}$ and $\boldsymbol{E}$ should be sorted in the correct order. If $\boldsymbol{R}$ and $\boldsymbol{C}$ are sorted as follows:

Row ID	Covariate 1	Covariate 2
Row 1	-0.6	0.4
Row 2	-0.2	-0.3
Row 3	1.6	0.9

Table 4.1: First three rows of table $\boldsymbol{R}$

Column ID	Covariate 1	Covariate 2
Column 1	0.4	-0.5
Column 2	-0.5	-2.3
Column 3	-0.3	1.0

Table 4.2: First three rows of table $\boldsymbol{C}$

Then $\boldsymbol{E}$ should be sorted as displayed in Table 4.3, with the row indices varying *inside* the colum indices.

Row ID	Column ID	Covariate 1	Covariate 2	Covariate 3
Row 1	Column 1	0.0	0.0	0.4
Row 2	Column 1	0.4	0.2	1.0
Row 3	Column 1	-0.4	0.2	-0.7
⋮	⋮	⋮	⋮	
Row 1	Column 2	-0.6	0.5	1.1
Row 2	Column 2	0.6	0.3	-0.2
Row 3	Column 2	-1.6	0.7	-0.1
⋮	⋮	⋮	⋮	

Table 4.3: $\boldsymbol{E}$

Now that we have our covariate matrices in the correct ordering, let us construct artificial row, column, and covariate effects, α^0, β^0 and ϵ^0, and artificial interactions $\boldsymbol{\Theta}^0$.

```
## parameters
alpha0 <- rep(0, m1);alpha0[1:6] <- 1
beta0 <- rep(0, m2);beta0[1:4] <- 1
epsilon0 <- rep(0, q);epsilon0[5:6] <- 0.2
r <- 2 #rank of interaction matrix theta0
theta0 <- 0.1*matrix(rnorm(m1*r), nrow=m1)%*%diag(r:1)%*%
   matrix(rnorm(m2*r), nrow=r)
theta0 <- sweep(theta0, 2, colMeans(theta0))
theta0 <- sweep(theta0, 1, rowMeans(theta0))
```

Finally, we can construct the parameter matrix $\boldsymbol{X}^0$ and sample our count data $\boldsymbol{Y}$:

```
## construct x0
U <- scale(U)#center and normalize the covariates
x0 <- matrix(rep(alpha0,m2),nrow=m1)#row effects
x0 <- x0 + matrix(rep(beta0,each=m1),nrow=m1)#add col effects
x0 <- x0 + matrix(U%*%epsilon0,nrow=m1) #add cov effects
x0 <- x0 + theta0 #add interactions

## sample count data y
y0 <- matrix(rpois(m1*m2, lambda = c(exp(x0))), nrow = m1)

## add missing values
p <- 0.2
y <- y0
y[sample(1:(m1*m2), round(p*m1*m2))] <- NA
```

4.1.2 Estimation and single imputation

Model and estimation The purpose of the `lori` package is dual: to estimate the parameters α^0, β^0, ϵ^0, $\boldsymbol{\Theta}^0$, and to impute the count table $\boldsymbol{Y}$. The corresponding input are the incomplete count data $\boldsymbol{Y}$ and (optionally) the covariate matrix $\boldsymbol{U}$. The estimation procedure is:

$$(\hat{\alpha}, \hat{\beta}, \hat{\epsilon}, \hat{\boldsymbol{\Theta}}) \in \operatorname{argmin} \mathcal{L}(\boldsymbol{Y}; \alpha, \beta, \epsilon, \boldsymbol{\Theta}) + \lambda_1 \|\boldsymbol{\Theta}\|_* + \lambda_2 \left(\|\alpha\|_1 + \|\beta\|_1 + \|\epsilon\|_1\right), \quad (4.2)$$

where $\mathcal{L}(\boldsymbol{Y}; \alpha, \beta, \epsilon, \boldsymbol{\Theta})$ is the Poisson negative log-likelihood:

$$\sum_{(i,j)} \Omega_{i,j}[-\boldsymbol{Y}_{i,j}(\alpha_i + \beta_j + \sum_{k=1}^{K} \epsilon_k \boldsymbol{U}(i,j)_k + \boldsymbol{\Theta}_{i,j}) + \exp(\alpha_i + \beta_j + \sum_{k=1}^{K} \epsilon_k \boldsymbol{U}(i,j)_k + \boldsymbol{\Theta}_{i,j})],$$

and $\Omega_{i,j} = 1$ if $\boldsymbol{Y}_{i,j}$ is observed, and 0 otherwise. The intuition behind (4.2) is that the negative log-likelihood is minimized (the parameters should fit the data as much as possible) with additional regularization terms which constrain the parameters and perform model selection automatically. The first penalty term, $\|\boldsymbol{\Theta}\|_*$ corresponds to the nuclear norm of $\boldsymbol{\Theta}$ and induces low-rank solutions: this is interpreted as assuming that a few latent factors summarize the interactions. The second term, $\|\alpha\|_1 + \|\beta\|_1 + \|\epsilon\|_1$, corresponds to the sum of the ℓ_1 norms of the vectors α, β and ϵ:

$$\|\alpha\|_1 + \|\beta\|_1 + \|\epsilon\|_1 = \sum_{i=1}^{m_1} |\alpha_i| + \sum_{j=1}^{m_2} \beta_j + \sum_{k=1}^{K} |\epsilon_k|.$$

This penalty induces sparse solutions (vectors α, β and ϵ containing many zeros), meaning that not all rows, columns and covariates have an effect on the counts. The trade-off between the data-fitting term and the penalties is controlled by the regularization parameters λ_1 and λ_2. As λ_1 increases, the rank of the solution $\hat{\boldsymbol{\Theta}}$ decreases (the number of latent factors decreases). As λ_2 increases, the number of nonzero values in $\hat{\alpha}$, $\hat{\beta}$ and $\hat{\epsilon}$ decreases (the number of active rows, columns and covariates decreases).

In R, the estimation is done as follows, for some predefined regularization parameters λ_1 and λ_2:

```
## lori estimation
lambda1 <- 0.1
lambda2 <- 0.1
res.lori <- lori(y, U, lambda1, lambda2)
```

Cross-validation and estimation However, the quality of the estimation is highly dependent on the choice of λ_1 and λ_2, and thus we provide a cross-validation function to select them automatically:

```
## cross-validation
res.cv <- cv.lori(y, U, trace.it = T) #this takes a few
   minutes, the trace.it argument states that information
   about the progress should be printed
## estimation
res.lori <- lori(y, U, res.cv$lambda1, res.cv$lambda2)
```

```
res.lori$alpha
res.lori$beta
res.lori$espilon
res.lori$theta
```

In the present case, we obtain point estimates for the main effects (with a one-digit precision), displayed in Table 4.4 for $\hat{\alpha}$, Table 4.5 for $\hat{\beta}$ and Table 4.6 for $\hat{\epsilon}$.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1.0	1.0	0.1	0.6	0.9	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 4.4: First 20 coefficients in $\hat{\alpha}$ (res.lori$alpha[1:20]). The cells colored in green correspond to the nonzero coefficients in α^0 ($\alpha_1^0 = \ldots = \alpha_6^0 = 1$).

1	2	3	4	5	6	7	8	9	10
1.2	0.7	0.9	0.6	0.0	0.0	0.0	0.0	0.0	0.2

Table 4.5: $\hat{\beta}$ (res.lori$beta). The cells colored in green correspond to the nonzero coefficients in β^0 ($\beta_1^0 = \ldots = \beta_4^0 = 1$).

1	2	3	4	5	6	7
0.1	0.1	-0.1	0.0	0.3	0.2	0.0

Table 4.6: $\hat{\epsilon}$ (res.lori$epsilon). The cells colored in green correspond to the nonzero coefficients in ϵ^0 ($\epsilon_5^0 = \epsilon_6^0 = 0.2$).

Single imputation The function `lori` also returns imputed values for the missing entries in $\boldsymbol{Y}$, based on these point estimates. For each missing entry, the predicted value is given by

$$\hat{\boldsymbol{Y}}_{i,j} = \exp\left(\hat{\alpha}_i + \hat{\beta}_j + \sum_{k=1}^{K_1+K_2} \boldsymbol{U}_{(j-1)m_1+i,k}\hat{\epsilon}_k + \hat{\boldsymbol{\Theta}}_{i,j}\right). \tag{4.3}$$

The imputed data set $\hat{\boldsymbol{Y}}$, such that $\hat{\boldsymbol{Y}}_{i,j} = \boldsymbol{Y}_{i,j}$ if $\boldsymbol{Y}_{i,j}$ is observed, and $\hat{\boldsymbol{Y}}_{i,j}$ given by (4.3) otherwise, is accessed as follows:

```
res.lori$imputed
```

	1	2	3	4	5	6	7	8	9	10
1	9	7	1	3	12	8	3	5	1	6
2	14	6	6.6	5	4	5	2.2	1	3	6
3	5	3.2	9	3.7	3	5	3	1	2	1
4	11	6	1	7	3.7	1.8	5	4	4	2
5	8.8	7	8	4	2	3	7	10	2	2.9
6	17	4.2	8	4.1	1	0	3	4	5	4

Table 4.7: First six rows of the imputed data set $\hat{\boldsymbol{Y}}$ (res.lori$imputed[1:5,]). The red cells correspond to imputed values, originally missing in $\boldsymbol{Y}$.

Removing row and column effects Note that, by default, the row and column effects α^0 and β^0 are estimated. To estimate model (4.1) without row and column effects, one can force them to zero by using the arguments `reff` (boolean indicating whether row effects should be fit) and `ceff` (boolean indicating whether column effects should be fit) in `lori`, which are set to TRUE by default:

```
res.noreff <- lori(y, U, lambda1, lambda2, reff=F)
res.noceff <- lori(y, U, lambda1, lambda2, ceff=F)
res.noreffceff <- lori(y, U, lambda1, lambda2, reff=F, ceff=F)
```

4.1.3 Multiple imputation

The function `mi.lori` performs multiple imputation, based on the single imputation procedure described above. To produce M imputed data sets, a two-step bootstrap

procedure is applied. First, the observed entries are sampled uniformly at random; second, Poisson noise centered around the observations is added. The M new data sets are each imputed using the previous single imputation procedure (function `lori`). Then, the multiple imputations may be pooled using the function `pool.lori`. Note that, by default, the parameter M is set to 100. The entire procedure goes as follows in R:

```
res.mi <- mi.lori(y, U, res.cv$lambda1, res.cv$lambda2)
res.pool <- pool.lori(res.mi)
```

The function `pool.lori` returns estimates for the mean and variance of the imputed values, and for all the parameters involved in the model ($\hat{\alpha}$, $\hat{\beta}$, $\hat{\epsilon}$, $\hat{\theta}$). Table 4.8 shows the result obtained for the imputed values. Compared to the single imputation result displayed in Table 4.7, we obtain intervals of variability for the predicted values.

	1	2	3	4	5	6	7	8	9	10
1	9	7	1	3	12	8	3	5	1	6
2	14	6	5.5(3.1)	5	4	5	1.7(1.6)	1	3	6
3	5	3.1(2.7)	9	3.1(2.1)	3	5	3	1	2	1
4	11	6	1	7	3.2(2.0)	1.4(1.7)	5	4	4	2
5	8.3(3.8)	7	8	4	2	3	7	10	2	2.2(2.1)
6	17	3.9(2.8)	8	3.4(2.4)	1	0	3	4	5	4

Table 4.8: Results for the first six rows of the imputed data set $\hat{\boldsymbol{Y}}$ after multiple imputation and pooling. The red cells correspond to imputed values, originally missing in $\boldsymbol{Y}$. The standard deviation of the multiple imputation (computed via Rubin's formula (Rubin, 1987)) is given between parenthesis.

One may also visualize the variability of the parameter estimates with boxplots. Indeed, for every single data set generated by the bootstrap procedure, denoted $\boldsymbol{Y}^1, \ldots, \boldsymbol{Y}^M$, one obtains point estimates $\hat{\alpha}^k$, $\hat{\beta}^k$, $\hat{\epsilon}^k$ and $\hat{\boldsymbol{\Theta}}^k$, $k \in [\![M]\!]$. For example for the column effects:

```
boxplot(res.mi$mi.beta, pch="", names=paste("col", 1:10))
```

The resulting boxplot is displayed in Figure 4.1.

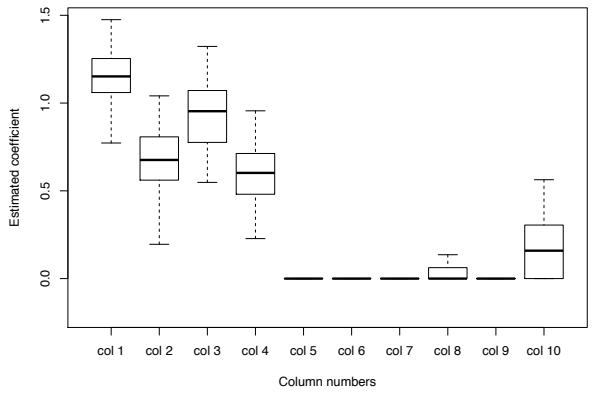

Figure 4.1: Boxplot of the estimators $\hat{\beta}^1, \ldots, \hat{\beta}^M$ produced by the multiple imputation procedure.

4.2 Analysis of the Aravo data set

4.2.1 Loading the data

The Aravo data set (Choler, 2005) consists of three main data tables. First, a count table collecting the abundance of 82 species of alpine plants in 75 sites in France (the rows correspond to the environments, and the column to species). We will denote this abundance table, displayed in Table 4.9, $\boldsymbol{Y} \in \mathbb{R}^{m_1 \times m_2}$. Second, a matrix containing 6 geographical and meteorological characteristics of the sites. Third, a matrix containing 8 species traits (height, spread, etc.). We denote $\boldsymbol{R}$ the matrix of row covariates, and $\boldsymbol{C}$ the matrix of column covariates, respectively displayed in Tables 4.10 anc 4.11.

```
library(ade4)
data("aravo")
```

	Agro.rupe	Alop.alpi	Anth.nipp	Heli.sede	Aven.vers	Care.rosa
AR07	0	0	0	0	0	1
AR71	0	0	0	0	0	2
AR26	3	0	1	0	1	2
AR54	0	0	0	2	0	2
AR60	0	0	0	0	0	0

Table 4.9: First 5 rows (environments) and 6 columns (species) of the Aravo count table (aravo$spe[1:5, 1:6]).

```
aravo$env[1:5,]
aravo$traits[1:6,]
```

	Aspect	Slope	Form	PhysD	ZoogD	Snow
AR07	7	2	1	50	no	140
AR71	1	35	3	40	no	140
AR26	5	0	3	20	no	140
AR54	9	30	3	80	no	140
AR60	9	5	1	80	no	140

Table 4.10: First 5 rows (environments) of the Aravo row covariates (aravo$env[1:5,]).

	Height	Spread	Angle	Area	Thick	SLA	N_mass	Seed
Agro.rupe	6	10	80	60.0	0.12	8.10	218.70	0.08
Alop.alpi	5	20	20	190.9	0.20	15.10	203.85	0.21
Anth.nipp	15	5	50	280.0	0.08	18.00	219.60	0.54
Heli.sede	0	30	80	600.0	0.20	10.60	233.20	1.72
Aven.vers	12	30	60	420.0	0.14	12.50	156.25	1.17
Care.rosa	30	20	80	180.0	0.40	6.50	208.65	1.68

Table 4.11: First 6 rows (species) of the Aravo column covariates (aravo$traits[1:6,]).

First we put the data in the right shape for the `lori` function.

```
Y <- aravo$spe # count table
R <- aravo$env # row covariates
R <- R[, c(1,2,4,6)] # keep quantitative variables (for
   simplicity)
C <- aravo$traits # column covariates
d <- dim(Y)
n <- d[1]
```

```
p <- d[2]
U <- covmat(n,p,R,C) # construct covariate matrix for lori
   input
U <- scale(U) # scale the covariates
```

4.2.2 Estimation of the lori model

Then, we tune the regularization parameters, and apply the lori function.

```
# Tune regularization parameter
res_cv <- cv.lori(Y, U, reff=F, ceff=F, trace.it=T, len=10)
res_lori <- lori(Y, U, lambda1 = res_cv$lambda1, lambda2=res_
   cv$lambda2, reff=F, ceff=F)
```

Aspect	Slope	PhysD	Snow	Height	Spread	Angle	Area	Thick	SLA	Nmass	Seed
0.00	0.02	-0.00	-0.01	0.00	-0.04	-0.02	-0.03	-0.02	-0.00	0.01	-0.01

Table 4.12: Estimated covariate effects in the Aravo data set using `lori`. The regularization parameters are selected using cross-validation.

The obtain covariates coefficients are reported in Table 4.12. The estimated rank of the interaction matrix $\hat{\mathbf{\Theta}}$ is 3. The environments (rows) and species (columns) can be visualized on a biplot (de Rooij and Heiser, 2005, Section 2.5), where rows and columns are represented simultaneously in a normalized Euclidean space. In such plots, the dimensions of the Euclidean space are given by the principal directions of $\hat{\mathbf{\Theta}}$, scaled by the square root of the singular values of $\hat{\mathbf{\Theta}}$. Such displays can be interpreted in terms of distance between points: a species and an environment that are close interact highly, and two species or two environments that are close have similar profiles. Justifications for such a distance interpretation can be found in (de Rooij and Heiser, 2005, Section 2.5) or (Fithian and Josse, 2017, Section 2). To visualize the two-dimensional display, one may use the following command, whose output plot in given in Figure 4.2:

```
plot.interaction(res_lori$theta)
```

4.2.3 Bootstrap and intervals of variability

Finally, even though the table $\boldsymbol{Y}$ in complete, the multiple imputation function may be used to obtain intervals of variability for the coefficients reported in Table 4.12. The following command performs multiple imputation (which may be seen as bootstrap in this case), with 20 replications. Then, one can visualize the variability of the main effects coefficients with a boxplot (given in Figure 4.3).

```
res_mi <- mi.lori(Y, U, lambda1 = res_cv$lambda1, lambda2=res_
   cv$lambda2, reff=F, ceff=F, M=20)
boxplot(res_mi$mi.epsilon)
```

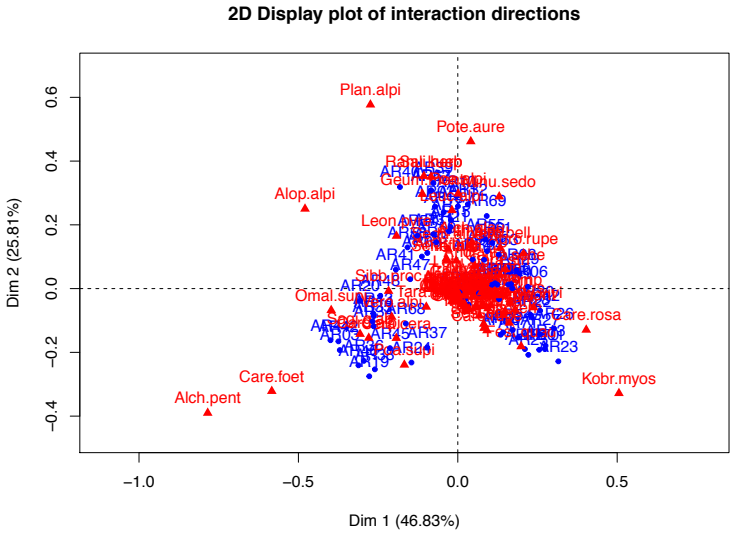

Figure 4.2: Two-dimensional display of the first dimensions of interaction.

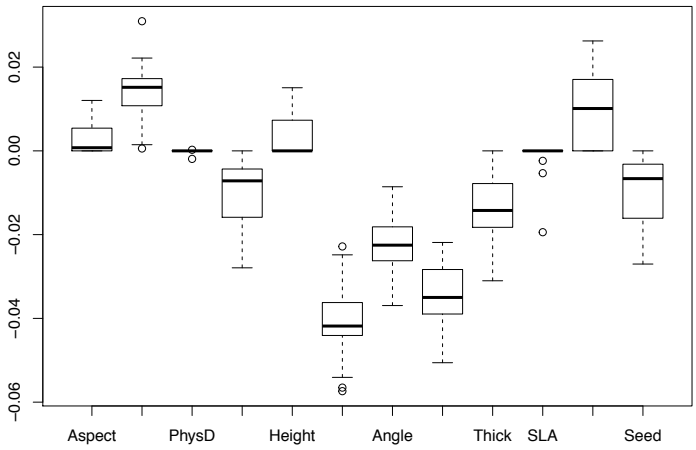

Figure 4.3: Boxplot of the main effects coefficients of the Aravo data set estimated with lori, across 20 bootstrap replications.