Package `dynamo`

February 23, 2018

Type Package

Title Fit a Stochastic Dynamical Array Model to Array Data

Version 1.0

Date 2018-02-23

Author Adam Lund

Maintainer Adam Lund <adam.lund@math.ku.dk>

Description An implementation of the method proposed in Lund and Hansen (2018) for fitting 3-dimensional dynamical array models. The implementation is based on the glamlasso package, see Lund et al. (2017) <doi:10.1080/10618600.2017.1279548>, for efficient design matrix free lasso regularized estimation in a generalized linear array model. The implementation uses a block relaxation scheme to fit each individual component in the model using functions from the glamlasso package.

License GPL (>= 2)

Depends glamlasso (>= 3.0), abind, MortalitySmooth

Imports Rcpp (>= 0.12.12)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-02-23 17:28:37 UTC

R topics documented:

fitmodel .......................................................... 2
V ................................................................. 5

Index 7
Sparse Estimation of Dynamical Array Models

Description

An implementation of the method proposed in Lund and Hansen, 2018 for fitting 3-dimensional dynamical array models. The implementation is based on the glamlasso package, see Lund et. al, 2017, for efficient design matrix free lasso regularized estimation in a generalized linear array model. The implementation uses a block relaxation scheme to fit each individual component in the model using functions from the glamlasso package.

Usage

```r
fitmodelHvL(phixL, phiyL, philL, phitL, penaltyfactorL, nlambda = 15, lambdaminratio = 0.0001, reltolinner = 1e-4, reltola = 1e-4, maxalt = 10)
```

Arguments

- `V` is the $N_x \times N_y \times N_t \times G$ array containing the data
- `phix` is a $N_x \times px$ matrix containing spatial basis functions
- `phiy` is a $N_y \times py$ matrix containing spatial basis functions
- `phil` is a $Lp1 \times pl$ matrix containing temporal basis functions
- `phit` is a $Nt \times pt$ matrix containing temporal basis functions
- `penaltyfactor` An array of size $p1 \times \cdots \times p_d$. Is multiplied with each element in `lambda` to allow differential shrinkage on the coefficients.
- `nlambda` the number of lambda values to fit
- `lambdaminratio` the smallest value for lambda, given as a fraction of
- `reitolinner` the convergence tolerance used with glamlasso
- `reitol` the convergence tolerance used for the alternation loop
- `maxalt` maximum number of alternations

Details

This package contains an implementation of the method proposed in Lund and Hansen, 2018 for fitting a (partial) 3-dimensional 3-component dynamical array model to a $N_x \times N_y \times N_t \times G$ data array $V$, where $N_x, N_y, N_t, G \in \{1, 2, \ldots\}$. Note that $N_x, N_y, N_t$ gives the number of observations in the two spatial dimensions and the temporal dimension respectively and $G$ gives the number of
Let $L$ be a positive integer giving the length of the modelled delay, $M := N_t - L - 1$ and $\phi^i$ be a $N_i \times p_i$ matrix for $i \in \{x, y, l\}$. Then define a $M \times p_x p_y p_l$ matrix

$$\Phi_{xy}^{g} = \left( \begin{array}{c} \text{vec}(\Phi_{1,g}^{xy}) \\ \vdots \\ \text{vec}(\Phi_{M,g}^{xy}) \end{array} \right),$$

for each $g \in \{1, \ldots, G\}$. Here using the so called rotated $H$-transform $\rho$ from Currie et al., 2006, $\Phi_{k,g}^{xy} \times p_x p_y p_l$ array that, for each $k \in \{1, \ldots, M\}$, can be computed as

$$\Phi_{k,g}^{xy} := \rho((\phi^l)^T, \rho((\phi^y)^T, \rho((\phi^x)^T, V_{i,(k-L):(k-1),g}))).$$

Then we can write the model equation as for each trial $g$ as

$$V_{i,(L+1):N_t,g} = \text{vec}(\rho(\phi^l, \rho(\phi^y, \rho(\phi^x, A)))) + \rho(\Phi_{xy}^{g}, \rho(\phi^y, \rho(\phi^x, \Gamma))) + V_{i,-1:(M-1),g} \odot C + E$$

where $A$ and $B$ are resp. $p_x \times p_y \times p_t$ and $p_x \times p_y \times p_x p_y p_l$ coefficient arrays that are estimated and $C$ is a $N_x \times N_y \times M$ array containing $M$ copies of the $N_x \times N_y$ matrix $\rho(\phi^y, \rho(\phi^x, \Gamma))$, where $\Gamma$ is a $p_x \times p_y$ coefficient matrix that is estimated. Finally $E$ is a $N_x \times N_y \times M$ array with Gaussian noise. See Lund and Hansen, 2018 for more details.

**Value**

An object with S3 Class "dynamo".

- **Out**
  A list where the first $G$ items are the individual fits to the trials containing:
  - **out$v**
The $N_x \times N_y \times N_t$ data array for trial $g$.
  - **out$Phixyl**
a $M \times p_x p_y p_l$ matrix, the convolution tensor.
  - **out$BetaS**
p$_x p_y p_l \times nlambda$ matrix containing the estimated parameters for the stimulus component for each $\lambda$ value
  - **out$BetaF**
p$_x p_y p_x p_y p_l \times nlambda$ matrix containing the estimated parameters for the filter component for each $\lambda$ value
  - **out$BetaH**
p$_x p_y \times nlambda$ matrix containing the estimated parameters for the instantaneous memory component for each $\lambda$ value
  - **out$lambda**
  vector of length $nlambda$ containing the penalty parameters.
  - **out$Obj**
  maxalt $\times nlambda$ matrix containing the objective values for each alternation and each $\lambda$
  - **phix**
is a $N_x \times p_x$ matrix containing the basis functions over the spatial $x$ domain.
  - **phiy**
is a $N_y \times p_y$ matrix containing the basis functions over the spatial $y$ domain.
  - **phil**
is a $L + 1 \times p_l$ matrix containing the basis functions over the temporal domain for the delay.
  - **phit**
is a $M \times p_l$ matrix containing the basis functions over the temporal domain for the stimulus.

**Author(s)**

Adam Lund

Maintainer: Adam Lund, <adam.lund@math.ku.dk>
References


Examples

```r
## Not run:
# Example showcasing the application from Lund and Hansen (2018).

# data
data(V)

# constants
Nx <- dim(V)[1]
Ny <- dim(V)[2]
Nt <- dim(V)[3]
L <- 50               # lag length in steps
Lp1 <- L + 1          # number of lag time points (= initial points)
t0 <- 0
M <- Nt - Lp1         # number of modelled time points
sl <- floor(200 / 0.6136) - 0 + 1    # stim start counted from -tau
sr <- sl + floor(250 / 0.6136)      # stim end counted from -tau

# no. of basis func.
px <- 8
py <- 8
pl <- max(ceiling(Lp1 / 5), 4)
pt <- max(ceiling((Nt - sl) / 25), 4)
degx <- 2
degy <- 2
degl <- 3
degt <- 3

# basis functions
library(MortalitySmooth)
phix <- round(MortalitySmooth_bb(x = 1:Nx, xl = 1, xr = Nx, ndx = px - degx, deg = degx), 10)
phyi <- round(MortalitySmooth_bb(x = 1:Ny, xl = 1, xr = Ny, ndx = py - degy, deg = degy), 10)
phil <- round(MortalitySmooth_bb(x = -tau:(t0 - 1), xl = -tau, xr = (t0 - 1),
                                 ndx = pl - degl, deg = degl), 10)
phit <- round(MortalitySmooth_bb(x = sl:Nt, xl = sl, xr = Nt, ndx = pt - degt, deg = degt), 10)
phit <- rbind(matrix(0, (sl - 1) - Lp1, dim(phit)[2]), phit)

# penalty weights
wt <- c(1, 1, 2, 2, 3, 3, 3, 3, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 2, 1, 1)
penSL <- list(matrix(1, px, py), matrix(1 / wt, dim(phit)[2], 1))
penF <- array(1, c(px, py, py * px * pl1))
penH <- matrix(1, px, py)
penaltyfactor <- list(penSL, penF, penH)

# run algorithm
system.time(Fit <- fitmodel(V,
```
VSD imaging of a live ferret brain.

Description

This data set contains one trial of processed voltage sensitive dye recordings from animal 308. The original raw data set contains integer values indicating the fluorescence. See Roland et al., 2006

Format

A three dimensional $25 \times 25 \times 977$ array.
References

Index

*Topic package
  fitmodel, 2

fitmodel, 2

V, 5