netReg

Simon Dirmeier

2017-02-04

Introduction

netReg is a package for graph-regularized linear regression. Graph prior knowledge is incorporated into the likelihood of the linear model in the form of the Laplacian matrices of the graphs. The Laplacians penalize the estimation of the coefficients for smooth solutions.

For high-dimensional data-sets often no unique solutions exist. With graph-regularization high-dimensional data-sets can be fitted efficiently.

Edgenet tutorial

This section explains how to fit a linear model and do parameter estimation using `edgenet`-regularization.

At first we generate some toy data randomly:

```
X <- matrix(rnorm(100*5), 100)
Y <- matrix(rnorm(100*5), 100)
```

Then we load the `netReg` library

```
library(netReg)
```

For `edgenet` we need to create an affinity matrix for the co-variables first. We also can create a graph for the responses, but this is not necessary to demonstrate the method. We could create a random graph like this:

```
aff.mat <- matrix(rpois(5*5, 1),5) * abs(rnorm(5*5))
aff.mat <- t(aff.mat) + aff.mat
diag(aff.mat) <- 0
```

We created the affinity matrix absolutely random, although in practice a *real* (biological) observed affinity matrix should be used, because in the end the affinity matrix decides the shrinkage of the coefficients.

Model fitting

Fitting a model using edge-based regularization is done with:

```
fit <- edgenet(X=X, Y=Y, G.X=aff.mat, lambda=1, psigx=1, family="gaussian")
print(fit)
```

```
##
## Call: edgenet.default(X = X, Y = Y, G.X = aff.mat, lambda = 1, psigx = 1,
##     family = "gaussian")
##
## Coefficients:
##             [,1]        [,2]        [,3]        [,4]        [,5]
## [1,] -0.18268405 -0.02224171  0.00000000 -0.06359686  0.05111031
## [2,]  0.00000000 -0.05086514  0.04313023  0.12833166 -0.03240594
## [3,] -0.01376668  0.01816981  0.01378198  0.17083431  0.09222172
## [4,]  0.02932186  0.03609595 -0.01238275  0.02545892  0.00000000
## [5,]  0.08554348 -0.07201482 -0.07097319 -0.03255783  0.04647016
## Intercept:
## [1]  0.01069500  0.03106659  0.12336473  0.06819523 -0.07508217
## Parameters:
## lambda psi_gx psi_gy
##      1      1      0
## Family:
## [1] "gaussian"
```

The `fit` object contains information about coefficients, intercepts, residuals, etc. Having the coefficients estimated we are able to predict novel data-sets:

```
X.new <-  matrix(rnorm(10*5),10)
pred  <- predict(fit, X.new)
```

The `pred` objects contains the predicted values for the responses.

Model-selection

In some cases we do not know the optimal shrinkage parameters. In that case we can just use `cv.edgenet` to find the best parameters for us. With *best* parameters we mean the ones reducing the *residual sum of squares* for Gaussian regression models:

```
cv <- cv.edgenet(X=X, Y=Y, G.X=aff.mat, family="gaussian")
print(cv)
```

```
##
## Call: cv.edgenet.default(X = X, Y = Y, G.X = aff.mat, family = "gaussian")
##
## Parameters:
##       lambda        psigx        psigy
## 1.357436e+01 3.225137e-04 0.000000e+00
## Family:
## [1] "gaussian"
```

References

Friedman J., Hastie T., Hoefling H. and Tibshirani R. (2007), Pathwise coordinate optimization. The Annals of Applied Statistics

Friedman J., Hastie T. and Tibshirani R. (2010), Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of statistical software

Fu W. J. (1998), Penalized Regression: The Bridge Versus the Lasso. Journal of computational and graphical statistics

Cheng W. and Wang W. (2014), Graph-regularized dual Lasso for robust eQTL mapping. Bioinformatics