HD_regression

Haotian Xu

9/2/2021

This is a simple guide for changepoint detection in high-dimensional linear regression.

Function *simu.change.regression* simulates sparse regression model with changepoints in coefficients.

Dynamic programming is implemented for regression changepoint detection:

- *DP.regression*: performs dynamic programming for regression changepoint detection through l0 penalty.
  - *CV.search.DP.regression*: perform grid search to select the tuning parameters (gamma for l0, lambda for l1) through Cross-Validation.

In addition, function *local.refine.regression* performs local refinement for an initial changepoint estimation.

Simulate data

```
library(changepoints)
```

```
## Loading required package: gglasso
```

```
## Loading required package: glmnet
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-2
```

```
## Loading required package: penalized
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:gglasso':
##
##     colon
```

```
## Welcome to penalized. For extended examples, see vignette("penalized").
```

```
## Loading required package: ks
```

```
## Loading required package: MASS
```

```
## parameters for simulating data
d0 = 5 # the number of nonzero elements
sigma = 1 # error standard deviation
kappa = 5 # minimum jump size in l2 norm
delta = 5 # minimal gap between boundaries
n = 200 # sample size
p = 50 # dimensionality
change.point = c(80, 170)
```

```
set.seed(0)
data = simu.change.regression(d0, change.point, p, n, sigma, kappa)
X = data$X
y = data$y
cpt_true = data$cpt.true
```

Perform dynamic programming

```
gamma.dp.set = c(0.01, 0.5, 1, 5, 10, 50) # a set of tuning parameters for DP
lambda.dp.set = c(0.01, 0.1, 1, 1.5) # a set of tuning parameters for lasso
DP_result = CV.search.DP.regression(y, X, gamma.dp.set, lambda.dp.set, delta) # grid search through cro
```

```
##       [,1]       [,2]       [,3]      [,4]
##  [1,] numeric,3  numeric,11 numeric,9 numeric,8
##  [2,] 3          11         9         8
##  [3,] 881.6197   780.3093   750.3877  791.0653
##  [4,] 0.06585709 1.584812   110.3635  194.1428
##  [5,] numeric,2  numeric,3  numeric,6 numeric,8
##  [6,] 2          3          6         8
##  [7,] 764.6688   554.4619   608.9422  791.0653
##  [8,] 0.1440713  4.898734   111.2579  194.1428
##  [9,] numeric,2  numeric,2  numeric,5 numeric,5
## [10,] 2          2          5         5
## [11,] 764.6688   950.2495   552.9739  483.8277
## [12,] 0.1440713  8.21851    114.369   204.563
## [13,] 97         numeric,2  numeric,2 numeric,2
## [14,] 1          2          2         2
## [15,] 7033.705   950.2495   307.2387  403.3635
## [16,] 6.969676   8.21851    134.5234  226.9008
## [17,] 97         83         numeric,2 numeric,2
## [18,] 1          1          2         2
## [19,] 7033.705   787.1442   307.2387  403.3635
## [20,] 6.969676   36.18725   134.5234  226.9008
## [21,] 97         numeric,0  numeric,0 numeric,0
## [22,] 1          0          0         0
## [23,] 7033.705   1019.984   689.6947  697.7489
## [24,] 6.969676   249.1271   409.5372  469.2157
```

```
min_idx = as.vector(arrayInd(which.min(DP_result$test_error), dim(DP_result$test_error)))# select gamma
cpt_DP_hat = unlist(DP_result$cpt_hat[min_idx[1], min_idx[2]]) # estimated changepoints by DP
cpt_DP_hat
```

```
## [1]  79 167
```

```
Hausdorff.dist(cpt_DP_hat, cpt_true)
```

```
## [1] 3
```

```
zeta = 1 # tuning parameter for group lasso
cpt_DPlr_hat = local.refine.regression(cpt_DP_hat, y, X, zeta, w = 1/3) # perform local refinement
cpt_DPlr_hat
```

```
## [1]  79 170
```

```
Hausdorff.dist(cpt_DPlr_hat, cpt_true)
```

```
## [1] 1
```