AR with adaptive Lasso

Changryong Baek

Automatic variable selection using LASSO

$y_i = \beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi} + \varepsilon_i$

Lasso type of estimation/variable selection can also be applied to time series. Recall Lasso is obtained (in the regression setting)

$$\min_{\beta_0,\beta}\left\{\frac{1}{N}\sum_{i=1}^{N}(y_i-\beta_0-x_i^T\beta)^2\right\} \text{ subject to } \underbrace{\sum_{j=1}^{p}|\beta_j| \le t}_{\text{penalty}}.$$

adaptive lasso

$\sum_{j=1}^{p} w_j|\beta_j| \le t$, $w_j = \frac{1}{|\hat\beta^{OLS}|}$

$\Rightarrow \hat\beta^{OLS}\downarrow$, $w_j\uparrow$, $\beta^{adap}=0$

$\hat\beta^{OLS}\uparrow$, $w_j\downarrow$, $\beta^{adap}\uparrow$

or equivalently,

$$\hat\beta^{Lasso} = \min_{\beta\in\mathbb{R}^p}\left\{\frac{1}{N}\underbrace{\|y-X\beta\|_2^2}_{L_2 \text{ norm}}+\lambda\underbrace{\|\beta\|_1}_{L_1 \text{ norm}}\right\}$$

Adaptive lasso is an improvement of lasso and conveniently implemented in parcor package. In the AR(p) setting, we can consdier it as a regression model

$$X_t = \overset{\beta_0}{\mu_s} + \overset{\beta_1}{\phi_1}X_{t-1} + \ldots + \overset{\beta_p}{\phi_p}X_{t-p} + \epsilon_t$$

Then, applying lasso automatically select the order and zero coefficients all together.

```
rm(list=ls(all=TRUE))
library(parcor);

ar.adaplasso = function(y, p, nf){

if(missing(nf)){ nf = 10 };
if(missing(p)){ phat = ar(y, aic = TRUE, order.max=10)$order }

# Check y is a vector
 y = as.vector(y);
 n = length(y);
 mu.s = mean(y);
 id = 1:n;
 X = NULL;
 for(j in 1:p){
 id1 = id-j;
 id2 = id1[id1 <= 0];
 id3 = id1[id1 > 0];
 X = cbind(X, c(rep(mu.s, length(id2)), y[id3]));
 }
 pp = adalasso(X, y, k=nf, intercept=TRUE);
return(pp)
}
```

nf: # of folding in cross-validation

Small simulation result shows that

```
n=250;
#phi = c(.5, .3, .1);
phi = c(.7, .3, 0, -.2);
```

AR(4) , $X_t = 0.7X_{t-1} + 0.3X_{t-2} + 0\cdot X_{t-3} - 0.2X_{t-4} + \varepsilon_t$

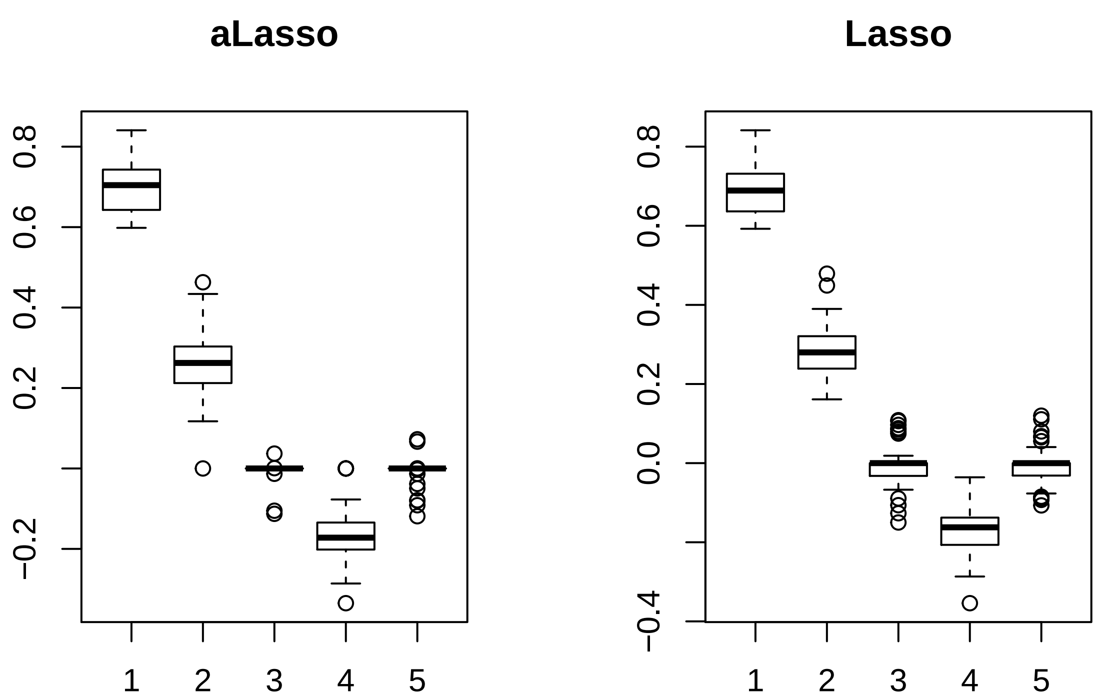
```r
nrep=50;
order=5;
A = B = matrix(0, nrep, order);
for(r in 1:nrep){

data = arima.sim(n = n, list(ar = phi), sd = 1)
y = data/sd(data);
fit = ar.adaplasso(y, p=order)
fit = ar.adaplasso(y, p=order)
A[r,] = fit$coefficients.adalasso
B[r,] = fit$coefficients.lasso
print(r)
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
## [1] 11
## [1] 12
## [1] 13
## [1] 14
## [1] 15
## [1] 16
## [1] 17
## [1] 18
## [1] 19
## [1] 20
## [1] 21
## [1] 22
## [1] 23
## [1] 24
## [1] 25
## [1] 26
## [1] 27
## [1] 28
## [1] 29
## [1] 30
## [1] 31
## [1] 32
## [1] 33
## [1] 34
## [1] 35
## [1] 36
## [1] 37
## [1] 38
## [1] 39
## [1] 40
```

```
## [1] 41
## [1] 42
## [1] 43
## [1] 44
## [1] 45
## [1] 46
## [1] 47
## [1] 48
## [1] 49
## [1] 50
```

```r
par(mfrow=c(1,2))
boxplot(A, main="aLasso");
boxplot(B, main="Lasso");
```



In-class exercise

Use Australian wine example:

Apply adaptive lasso algorithm to find the best model. Also reestimate parameters using constraint optimzation once you get the final model from adaptive lasso.

Cautions: Lasso / adaptive lasso are developed for IID data

It looks like lasso seems work well in time series setting, however, tuning parameter λ in lasso is the key in finite sample performance. It is well documented that CV often fails in time series context.

We can use BIC to select λ