We can break down the code into three tasks:

• Generating the data for 5 different scenarios.
• Carrying out different techniques for each scenario etc.
• Comparing the methods

Different methods used:
a) Elastic net with alpha and lambda decided via cross-validation
b) Lad lasso based on flare package
c) sqrt lasso based on flare package
d) l_q lasso based on flare package
e) Dantzig selector based on flare package
f) Adaptive lasso based on the least squares approximation proposed by Wang and Leng (2007, JASA).
g) 4 variants of adaptive lad lasso with penalty terms chosen in accordance with the suggestions in Section 4.1 of the article below

A brief review of the various methods used…

Elastic Net

- We try to minimize the below quantity:

$\hat{\beta} = \arg\min_\beta ||Y - X\beta||^2 + \lambda_2||\beta||^2 + \lambda_1|| \beta||_1$

- Elastic net is an intermediary method between Ridge(alpha=0) and Lasso(alpha=1). We can say that it tries to get the best of both the methods and it does in some cases for example when we see multicollinearity.
- It is expected that elastic net performs well when there is strong correlation among the variables. The lasso penalty will end to pick just one from a set of strong but correlated variables and ignore the rest. The ridge penalty will tend to shrink the coefficients of correlated variables toward each other the elastic net penalty is a good compromise.
- The ***L2*** part of the solution encourages highly correlated features to be averaged, while the first ***L1*** encourages a sparse solution in the coefficients of these averaged features.
- For the p>N case, whereas the lasso can select less than N, elastic net can include more than N predictors into the model.

Implementation

- Simultaneous Cross Validation of Alpha and Lambda was performed to find the best combination of Alpha and Lambda.
- Alpha values depend on the loop variable . it was defined as i/10. for i =1:10.
- 10-fold Cross validation for each alpha = 0, 0.1, ... , 0.9, 1.0

```
for (i in 1:10)        # i is alpha level
  {
      cvfit <- cv.glmnet(Data$X, Data$y, foldid=fold, alpha=i/10,
      type.measure="mse",family="gaussian")
      lambda_values[i] <- cvfit$lambda.min       # best lambda for alpha=i/10
      min.index <- which(cvfit$lambda==cvfit$lambda.min) # store location of best lambda
      cv.error[i] <- cvfit$cvm[min.index]    # store errors
  }
min.index<-which.min(cv.error)        # locate smallest error
selected_model<-glmnet(Data$X, Data$y, alpha=alpha_values[min.index],
lambda=lambda_values[min.index])         # refit model using optimal alpha and lambda
```

Adaptive lasso

- Adaptive lasso follows the below rule:

$$\hat{\boldsymbol{\beta}}^{*(n)} = \arg\min_{\boldsymbol{\beta}} \left\| \mathbf{y} - \sum_{j=1}^{p} \mathbf{x}_j \beta_j \right\|^2 + \lambda_n \sum_{j=1}^{p} \hat{w}_j |\beta_j|$$

- Adaptive weights are used for penalizing different coefficients in the L1 penalty
- Adaptive Lasso expected to have oracle properties.

Least-angle regression

It is effective in contexts where *p* >> *n.*

Adaptive LAD lasso

- It is similar to Adaptive Lasso except
- Minimizes(**LAD + Penalty)** instead of RSS + Penalty. The penalty term is the **weighted L1 norm** mentioned in the previous case.

Implementation

It is implemented as described in ”**The Adaptive Lasso and Its Oracle Properties**”. In the paper Beta$_{(OLS)}$ was mentioned as the weight. Since we are performing LAD LASSO, Beta$_{(RQ)}$ was used in this case.

four ADAPTIVE LAD Lasso Methods from the Paper

- Four penalties were tried for **Adaptive LAD Lasso**

$\lambda_1 = \sqrt{1.5n \log p}$, $\lambda_2 = \sqrt{2n \log p}$, $\lambda_3 = \sqrt{4n \log p}$, and $\lambda_4 = \sqrt{10n \log p}$.

```
Lambda=sqrt(2*n*log(p))  #original penalty
weight<- coef(rq(Data$y~Data$X))  #Weight=LAD coefficents
weight<-weight[-1] # Remove the intercept for the weights
weighted_lambda=Lambda/(abs(weight)) # calculate Weighted penalty term
weighted_lambda<-c(0,weighted_lambda) # add zero penatlty for the intercept
tempcfQR<- coef(rq(Data$y~Data$X,.5,method="lasso",lambda = weighted_lambda))
```

FLARE PACKAGE

- Variable selection by the L1 penalization

Here **y** represents the observations, **X** represents the basis functions for each observation, β are the regression coefficients. The subscripts indicate the choice of the norms used. λ is a positive coefficient determined by cross-validation.

METHOD	**Function to minimize**
LASSO	$\min_\beta \lvert y-X\beta\rvert^2 + \lambda\lvert\beta\rvert_1$
LAD – LASSO	$\min_\beta \lvert y-X\beta\rvert_1 + \lambda\lvert\beta\rvert_1$
Dantzig Selector	$\min_\beta \lVert X_T(y-X\beta)\rVert_\infty + \lambda\lvert\beta\rvert_1$

From Vignette of FLARE package:

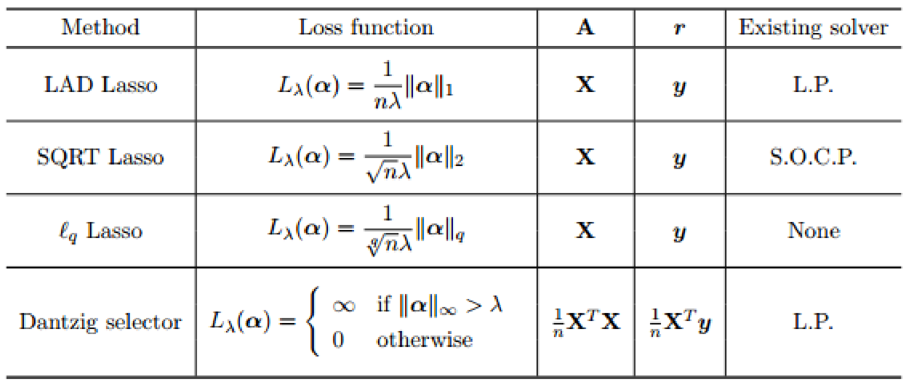
- **LAD Lasso** is robust to heavy tail random noise and outliers (Wang, 2013).
- **SQRT Lasso** is tuning insensitive (the optimal regularization parameter selection does not depend on any unknown parameter, Belloni et al. (2011)).
- **LQ Lasso** shares the advantage of LAD Lasso and SQRT Lasso.
- **Dantzig selector** can tolerate missing values in the design matrix and response vector (Candes and Tao, 2007).

After we specify ***method="lq "***
Where 1<= q <=2.

Method	**q value**
LQ Lasso	1.5
LAD Lasso	1
SQR Lasso	2

At q=1.5-**LQ norm regrelarized regression (with screening)**.

Table 1: All regression methods provided in the flare package



Method	Loss function	A	r	Existing solver
LAD Lasso	$L_\lambda(\alpha) = \frac{1}{n\lambda}\|\alpha\|_1$	$\mathbf{X}$	$\boldsymbol{y}$	L.P.
SQRT Lasso	$L_\lambda(\alpha) = \frac{1}{\sqrt{n}\lambda}\|\alpha\|_2$	$\mathbf{X}$	$\boldsymbol{y}$	S.O.C.P.
ℓ_q Lasso	$L_\lambda(\alpha) = \frac{1}{\sqrt[q]{n}\lambda}\|\alpha\|_q$	$\mathbf{X}$	$\boldsymbol{y}$	None
Dantzig selector	$L_\lambda(\alpha) = \begin{cases} \infty & \text{if } \|\alpha\|_\infty > \lambda \\ 0 & \text{otherwise} \end{cases}$	$\frac{1}{n}\mathbf{X}^T\mathbf{X}$	$\frac{1}{n}\mathbf{X}^T\boldsymbol{y}$	L.P.

$X \in R^{n\times d}$ denotes the design matrix, and $y \in R^{n}$ denotes the response vector. "L.P." denotes the general linear programming solver, and "S.O.C.P" denotes the second-order cone programming solver

- LAD-LASSO is less sensitive to outliers in contrast to the LASSO so it is preferred when we have heavy-tailed errors present.

Implementation

How to choose the best performing lambda for LAD,SQRT and LQ Lasso?

1) First obtain slim object

```
result_2 = slim(Data$X,Data$y,method="lq",nlambda=10,q=2)
```

2) Calculate the MSE for each lambda by subtracting the Original Response from the New Response obtained from the lambda.

```
residuals=abs((Data$X%*%result_2$beta)-Y_original)
mse_sqr_lasso<<-(apply(residuals, 2,mean))^2
Y_original= matrix(rep(Data$X%*%beta, each=10), ncol=10)
mse_lad_lasso
```

```
 8.7686   31.75   18.323   9.8079   13.242   9.7045   11.214   10.889   20.868   25.2003
```

```
result_2$lambda
```

```
0.4273919 0.3865510   0.3496129   0.3162045   0.2859885 0.2586599 0.2339428 0.2115877
0.1913687 0.1730818
```

3)We look for the Lambda that gives the lowest MSE. This is the Best Performing lambda.

```
min.index=which.min(mse_lad_lasso)
[1] 0.4273919
```

We can follow the same procedure for SQRT and LQ Lasso.

- The only difference for Dantzig Lasso is we need to specify method="dantzig"

```
slim(Data$X,Data$y,method="dantzig",nlambda=10)
```

Scenario 1 : Uncorrelated Variables:

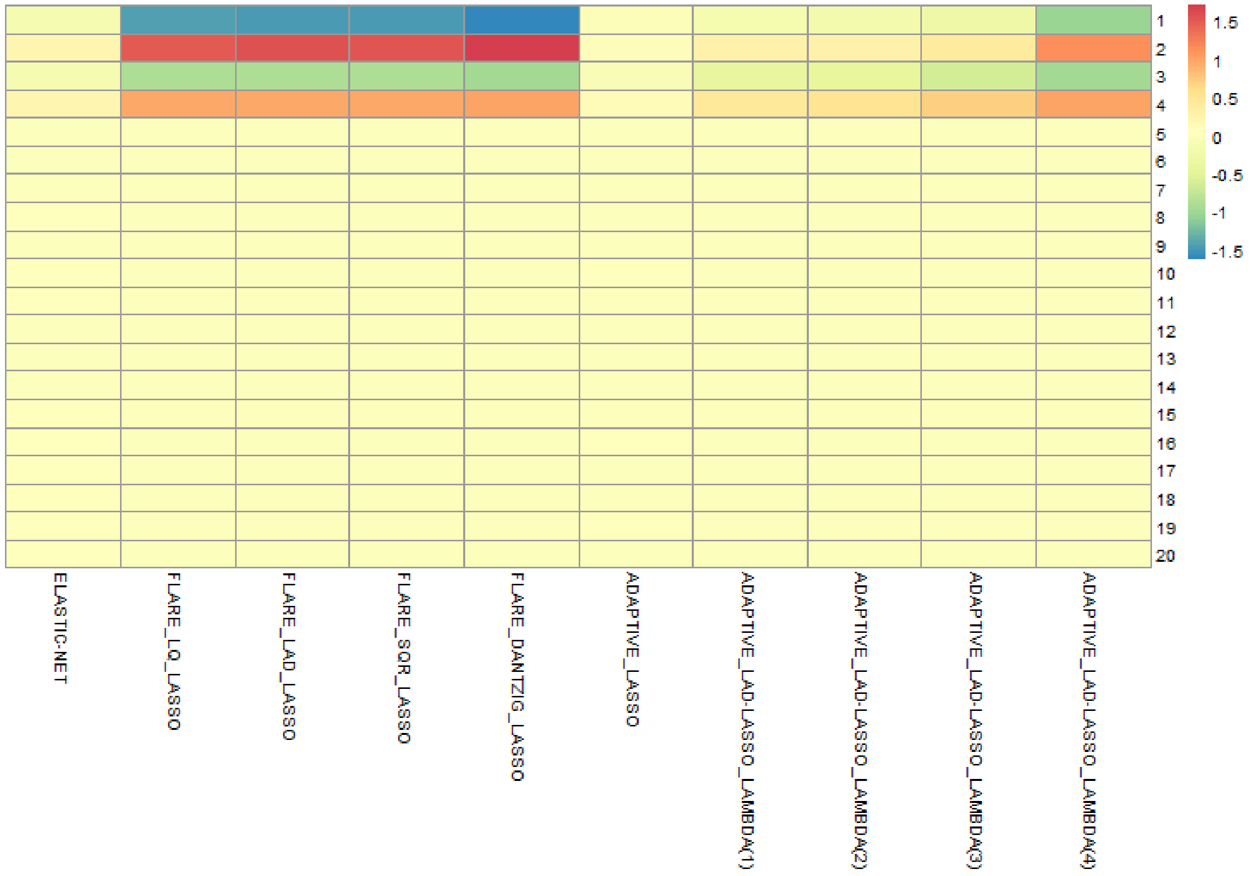
Scenario 1 is defined by:

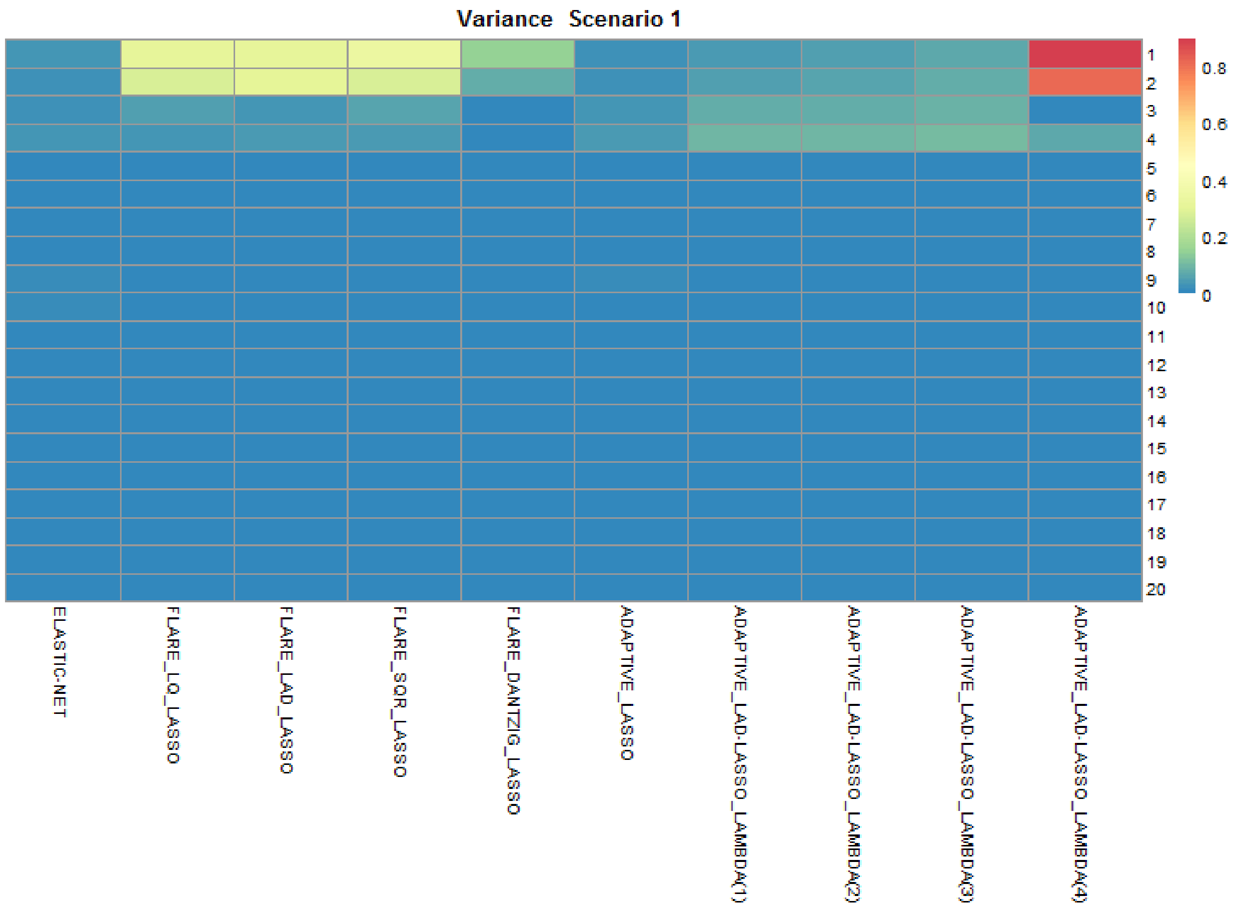
```
n=100
N <- 150
beta <- c(2,-2,1,-1,rep(0,16))
p <- length(beta)
X <- matrix(rt(n*p,5),nrow=n,ncol=p)
y <- rnorm(n,X%*%beta,2)
```

RESULTS:

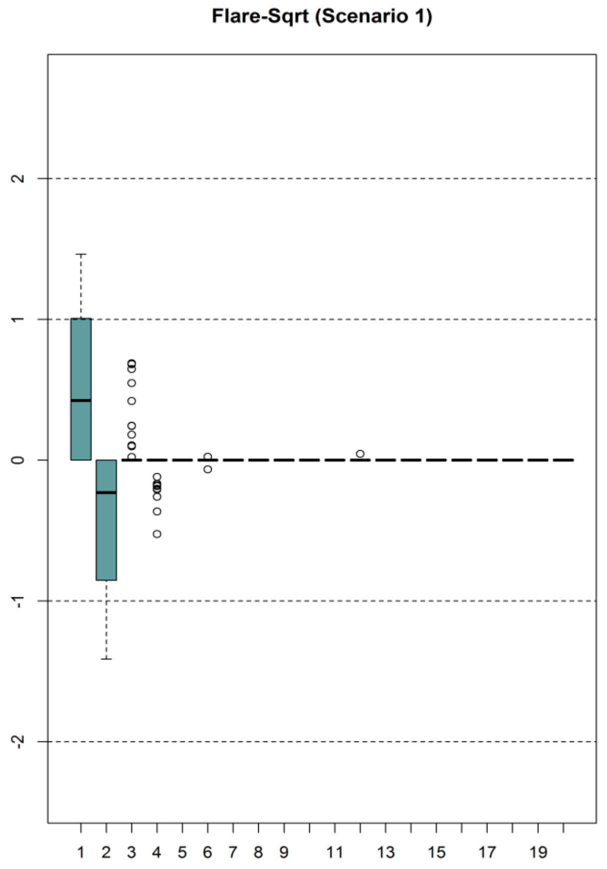
ELASTIC-NET	LQ_LASSO	LAD_LASSO	SQR_LASSO	DANTZIG_LASSO
0.3208773	6.6060218	6.625781762	6.720175	7.12659
ADAPTIVE_LASSO	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(3)	ADAPTIVE_LAD-LASSO LAMBDA(4)
0.174311	0.622954	0.781596	1.41585	6.129742

Bias Scenario 1

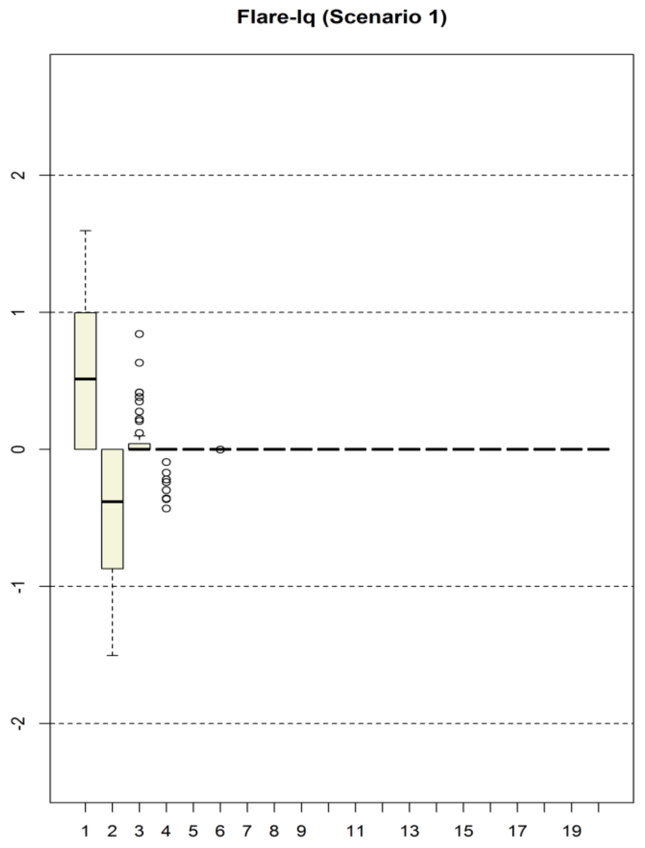




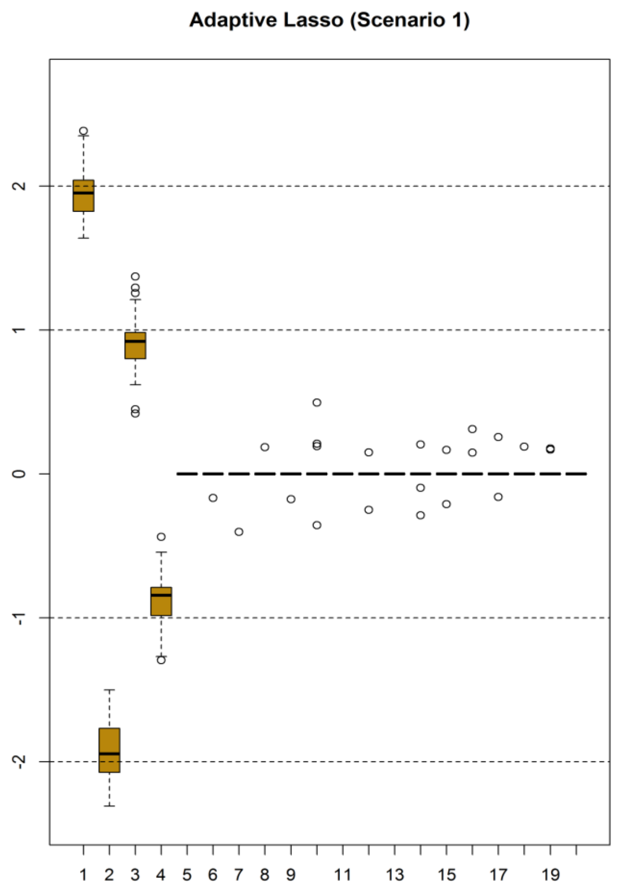
Variance Scenario 1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
ELASTIC-NET
FLARE_LQ_LASSO
FLARE_LAD_LASSO
FLARE_SQR_LASSO
FLARE_DANTZIG_LASSO
ADAPTIVE_LASSO
ADAPTIVE_LAD-LASSO_LAMBDA(1)
ADAPTIVE_LAD-LASSO_LAMBDA(2)
ADAPTIVE_LAD-LASSO_LAMBDA(3)
ADAPTIVE_LAD-LASSO_LAMBDA(4)

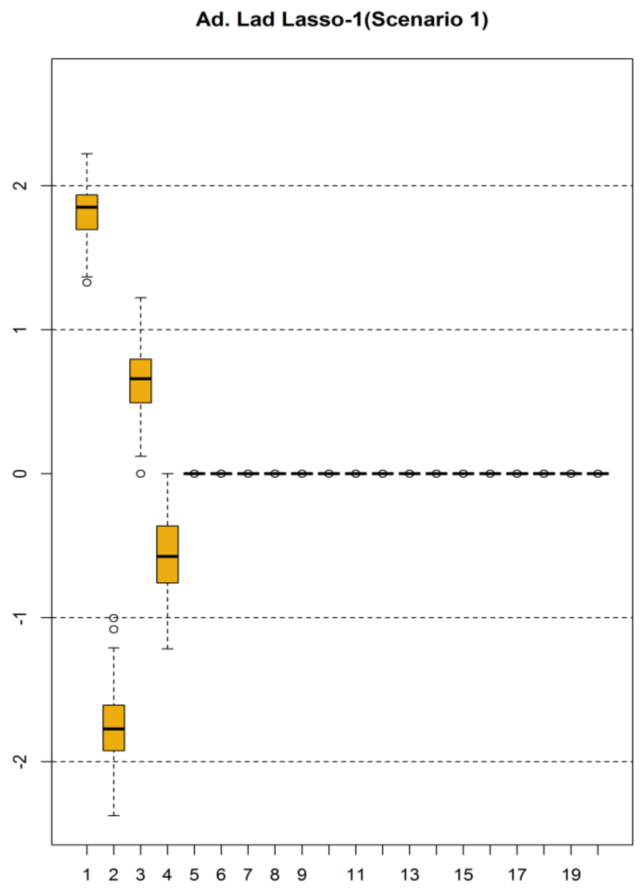


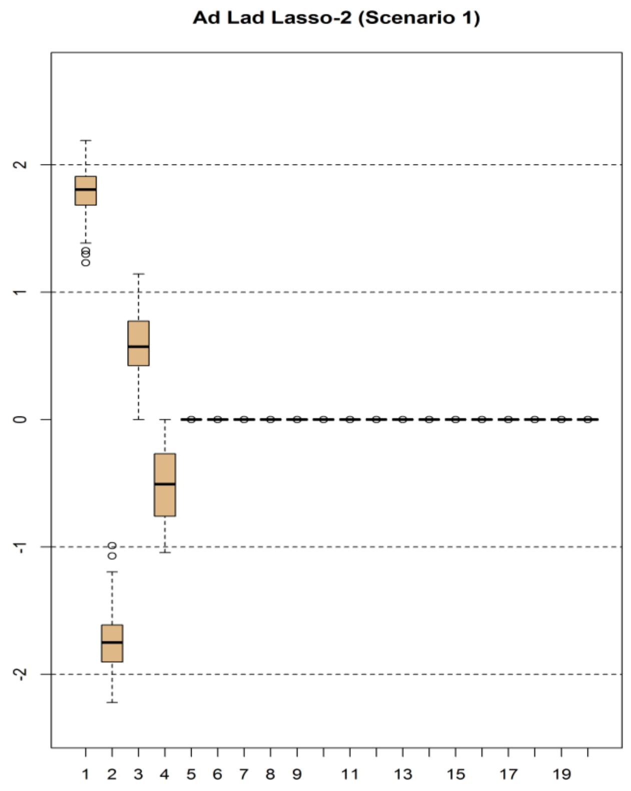
Flare-Sqrt (Scenario 1)
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19

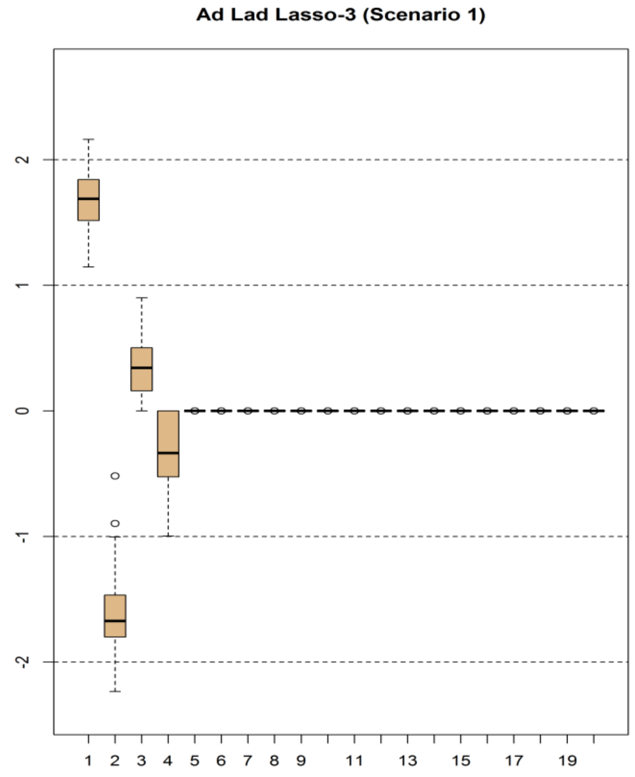


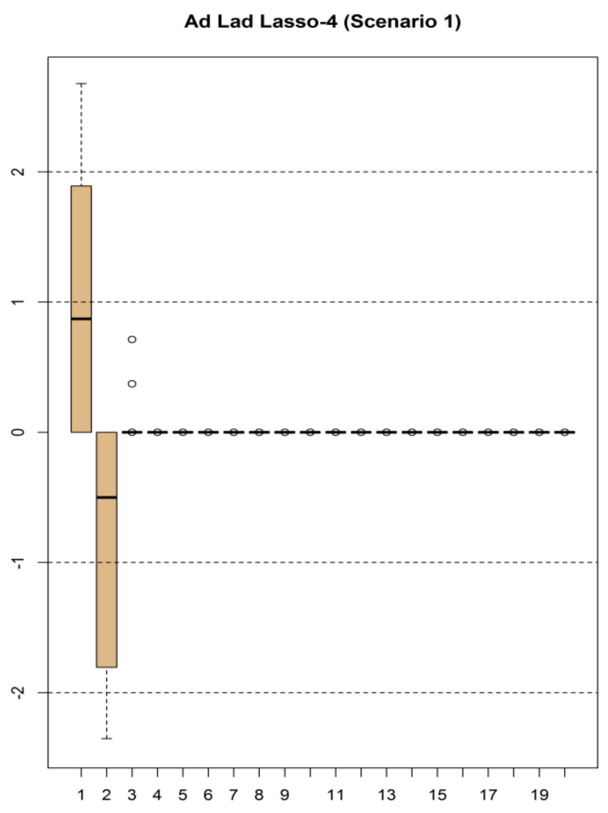
Flare-lq (Scenario 1)
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19

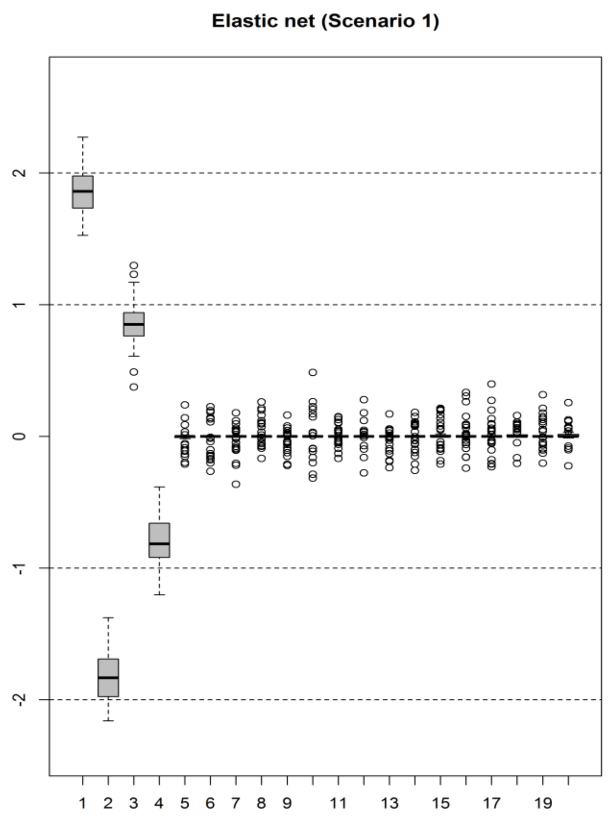

Adaptive Lasso (Scenario 1)
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19

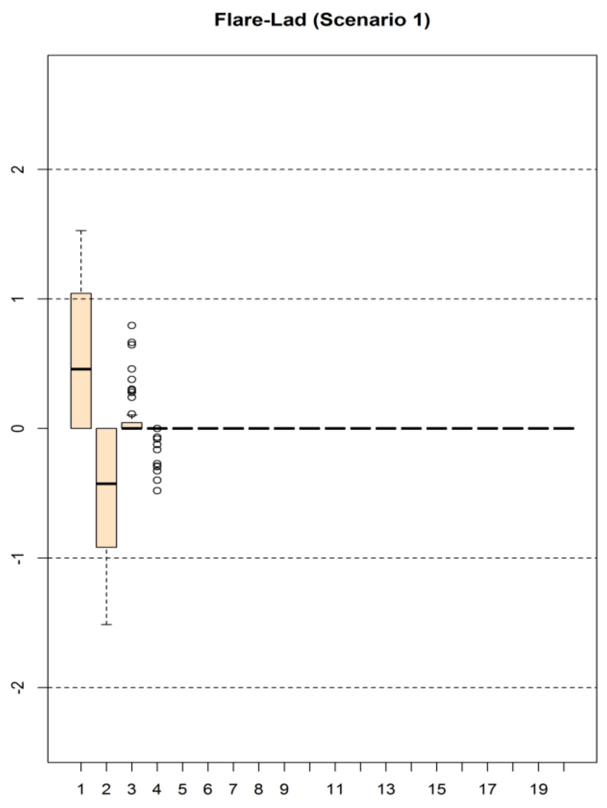

Ad. Lad Lasso-1(Scenario 1)
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19

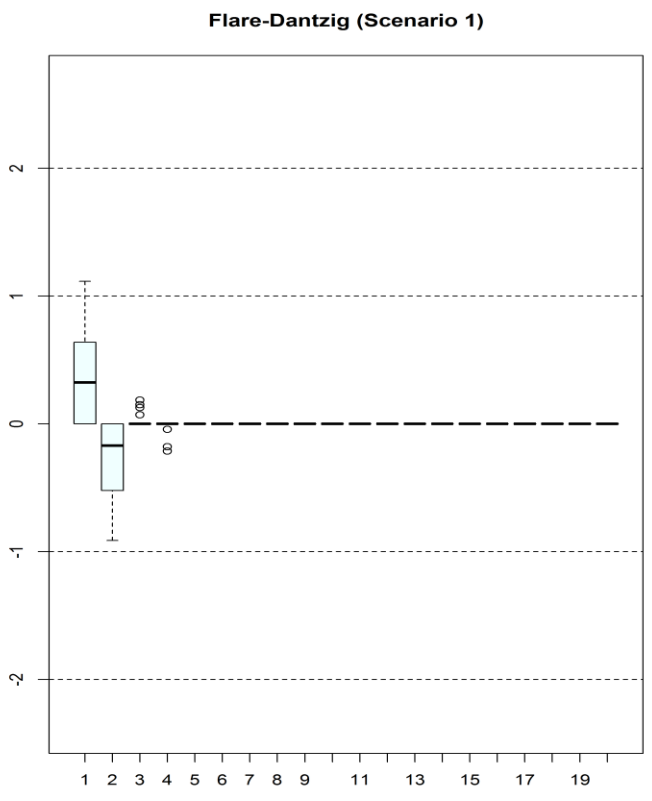

Ad Lad Lasso-2 (Scenario 1)
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Ad Lad Lasso-3 (Scenario 1)
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Ad Lad Lasso-4 (Scenario 1)


Elastic net (Scenario 1)


Flare-Lad (Scenario 1)


Flare-Dantzig (Scenario 1)

Some Obervations:

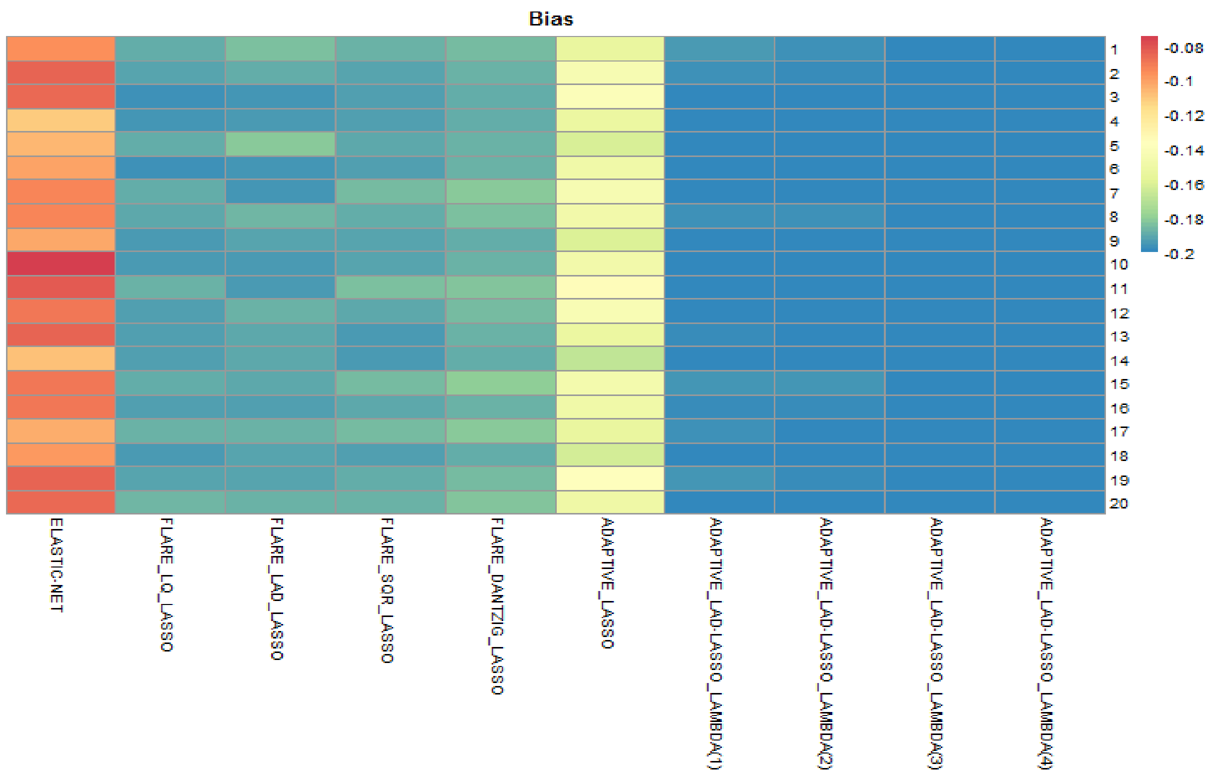
- Best performer : "**Adaptive Lasso**"
- Elastic Net also did very well better than the Flare Lassos and Adaptive Lad Lassos.
- Lambda 1 penalty showed good performance
- Lambda 4 penalty and Flare methods did not do well.
- In general we can say that adaptive methods performed better than non-adaptive methods
- Adaptive lasso and Elastic net predict very close to real values
- Lambda4 shows very high variance

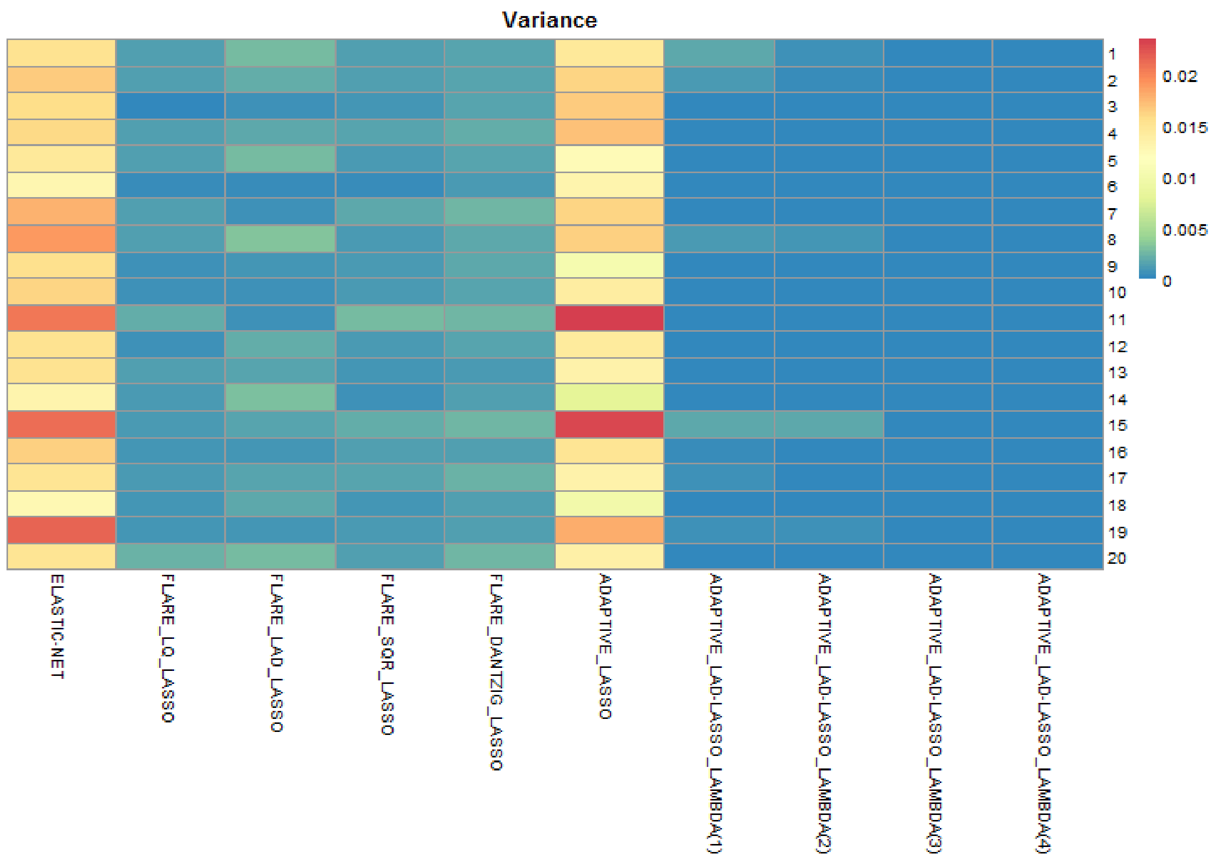
Scenario 2: ALL PREDICTORS ARE EQUAL

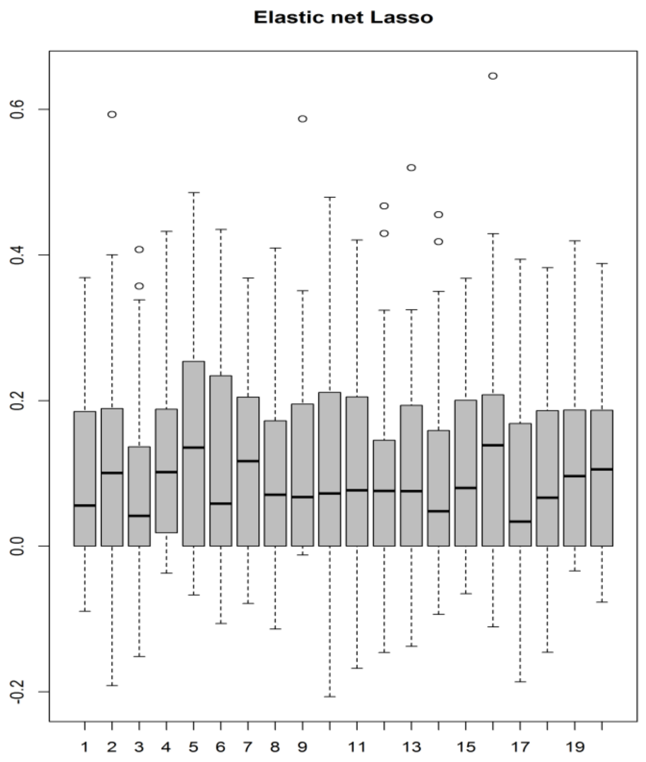
Scenario 2 is defined by:

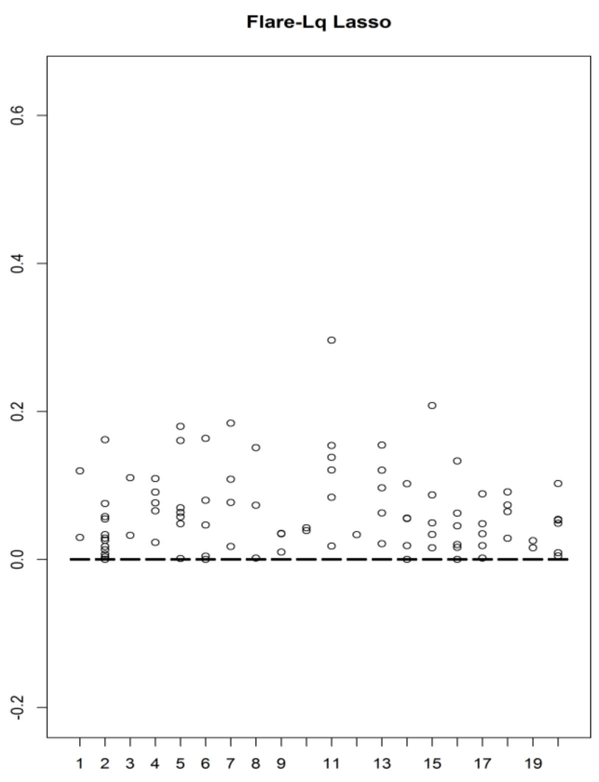
```
n=100
N=10
beta=rep (0.2,20) #ALL PREDICTORS ARE EQUAL
p=length(beta)
X <- matrix(rt(n*p,5),nrow=n,ncol=p)
y <- rnorm(n,X%*%beta,2)
```

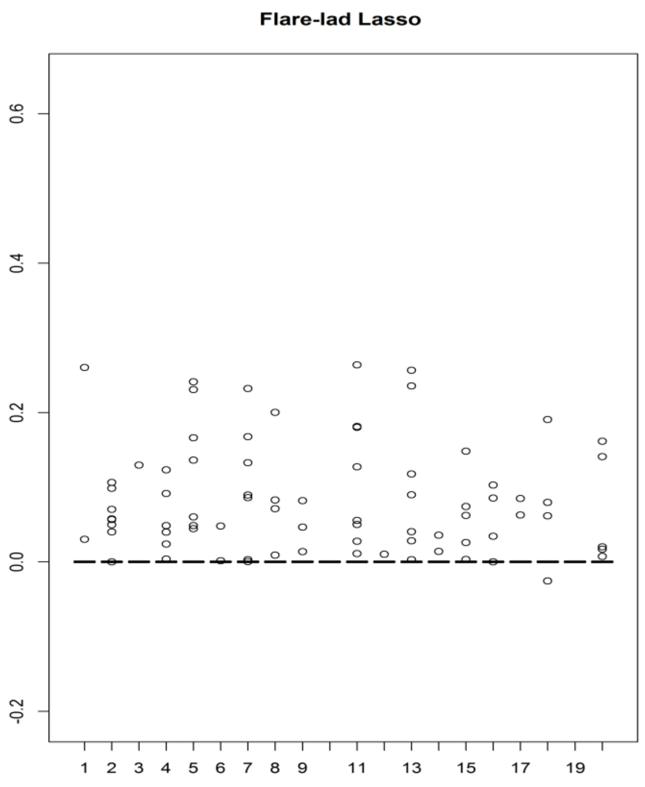
ELASTIC-NET	LQ_LASSO	LAD_LASSO	SQR_LASSO	DANTZIG_LASSO
0.499826	0.7578339	0.760539	0.7477682	0.7284788
ADAPTIVE_LASSO	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(3)	ADAPTIVE_LAD-LASSO LAMBDA(4)
0.7462397	0.7949623	0.7973586	0.7998412	0.8000000

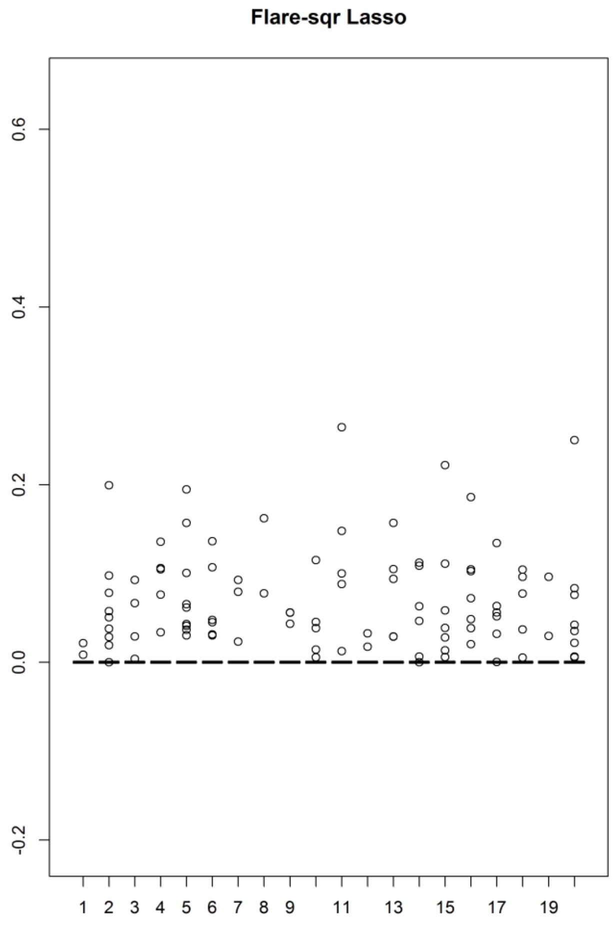


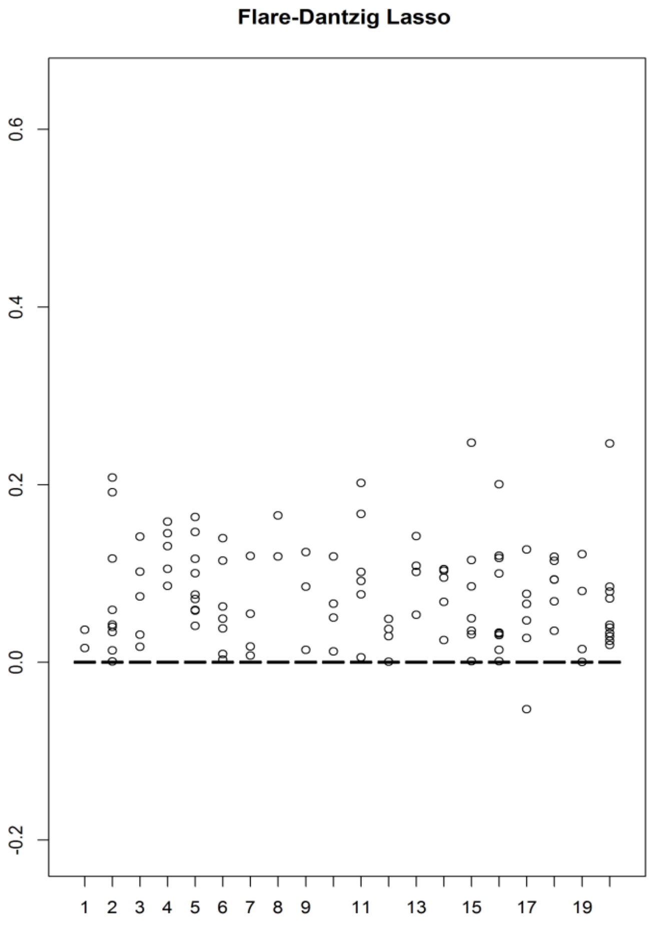

Variance
ELASTIC-NET
FLARE_LQ_LASSO
FLARE_LAD_LASSO
FLARE_SQR_LASSO
FLARE_DANTZIG_LASSO
ADAPTIVE_LASSO
ADAPTIVE_LAD-LASSO_LAMBDA(1)
ADAPTIVE_LAD-LASSO_LAMBDA(2)
ADAPTIVE_LAD-LASSO_LAMBDA(3)
ADAPTIVE_LAD-LASSO_LAMBDA(4)
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
0.02
0.015
0.01
0.005
0

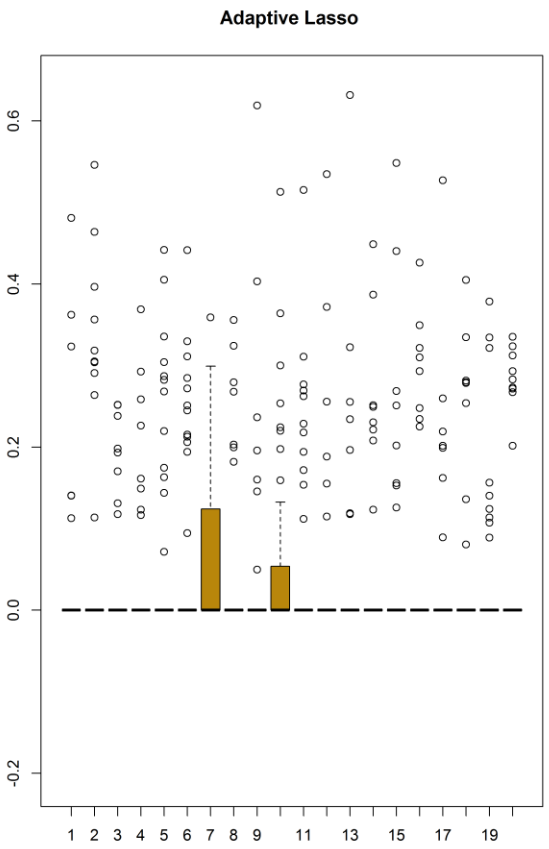

Elastic net Lasso
0.6
0.4
0.2
0.0
-0.2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Flare-Lq Lasso
-0.2
0.0
0.2
0.4
0.6
1 2 3 4 5 6 7 8 9 11 13 15 17 19

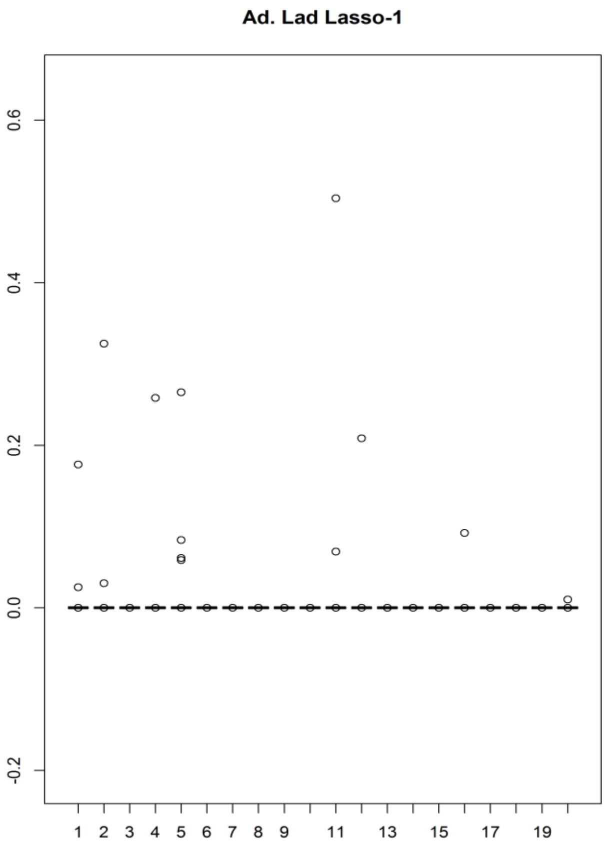

Flare-lad Lasso
-0.2
0.0
0.2
0.4
0.6
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Flare-sqr Lasso
-0.2
0.0
0.2
0.4
0.6
1 2 3 4 5 6 7 8 9 11 13 15 17 19

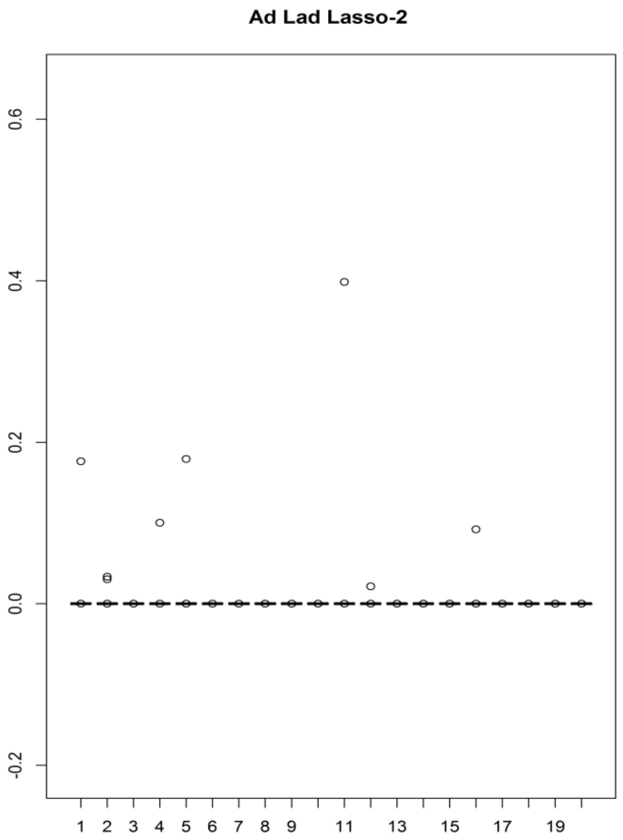

Flare-Dantzig Lasso
-0.2
0.0
0.2
0.4
0.6
1 2 3 4 5 6 7 8 9 11 13 15 17 19



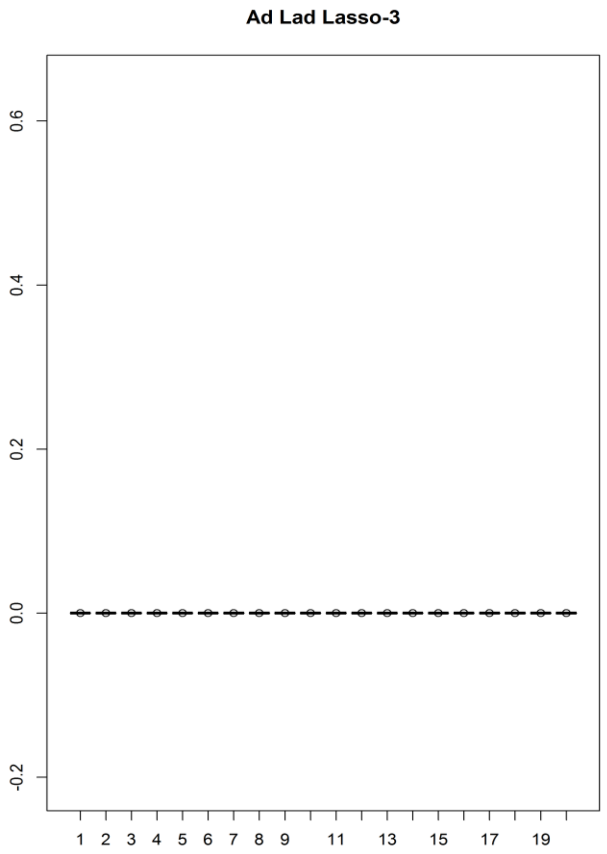
Adaptive Lasso
0.6
0.4
0.2
0.0
-0.2
1 2 3 4 5 6 7 8 9 11 13 15 17 19



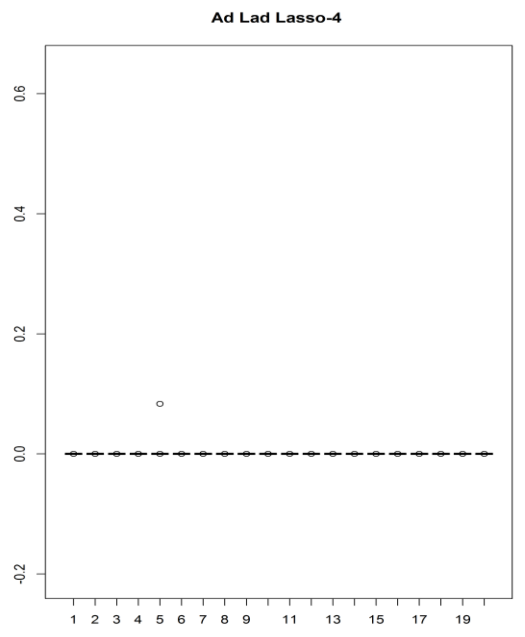
Ad. Lad Lasso-1
0.6
0.4
0.2
0.0
-0.2
1 2 3 4 5 6 7 8 9 11 13 15 17 19



Ad Lad Lasso-2
0.6
0.4
0.2
0.0
-0.2
1 2 3 4 5 6 7 8 9 11 13 15 17 19



Ad Lad Lasso-3
0.6
0.4
0.2
0.0
-0.2
1 2 3 4 5 6 7 8 9 11 13 15 17 19



Obervations

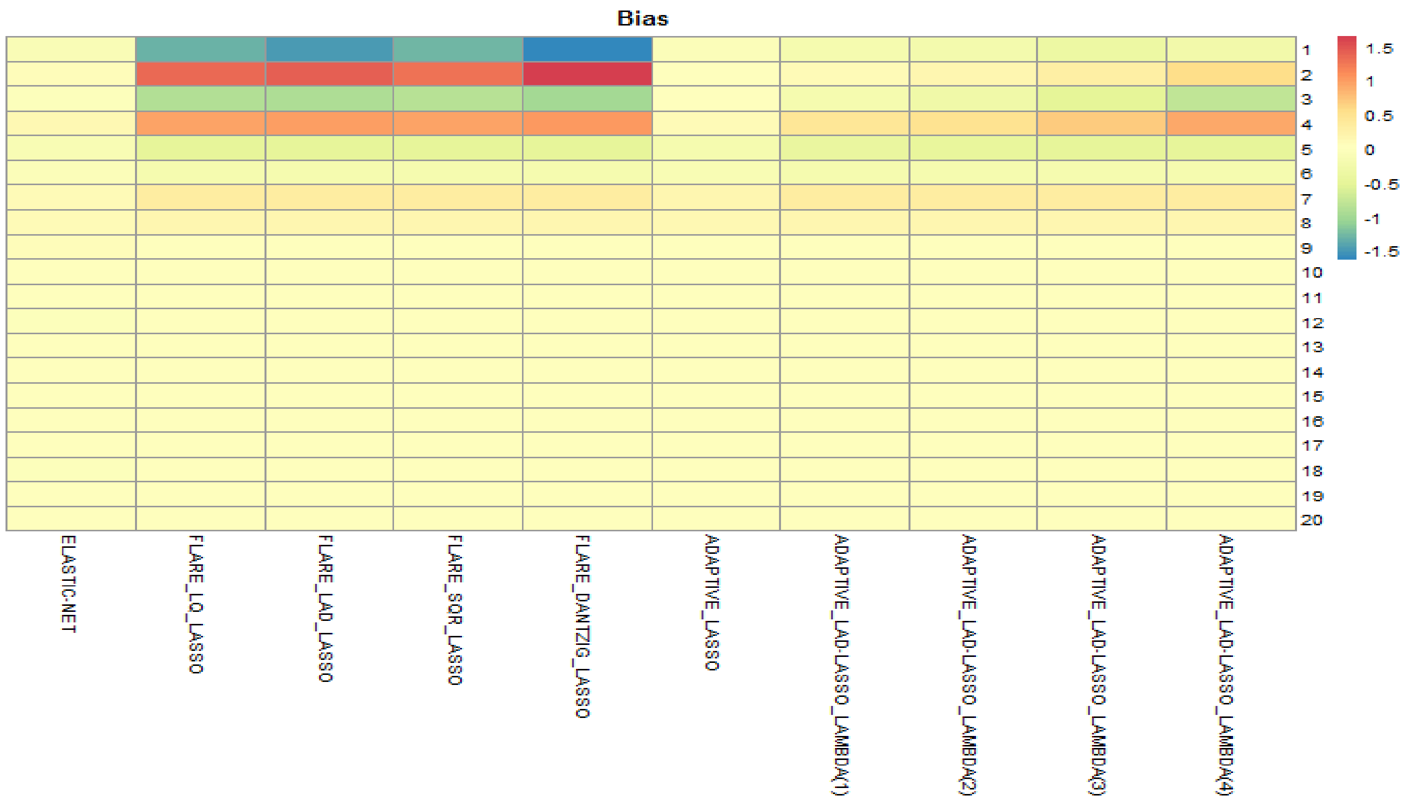
- Elastic net gives the best performance though it shows high variability.
- Adaptive LAD Lambda was shrinking almost all predictors close to zero.Especially lambda4 since it was the largest penalty.
- Elastic coefficents are pretty good. 0.24733, 0.211860, 0.1891178, 0.2040464, 0.1878290, 0.2411749, 0.1996526, 0.2133963, 0.1923646, 0.1811714, 0.1821808, 0.2021172, 0.1811008, 0.1796055, 0.1848787, 0.2060155, 0.1848596, 0.1877940, 0.1767627, 0.1929914. All coefficents are close to 0.2

Scenario 3: ALL PREDICTORS ARE STRONGLY CORRELATED

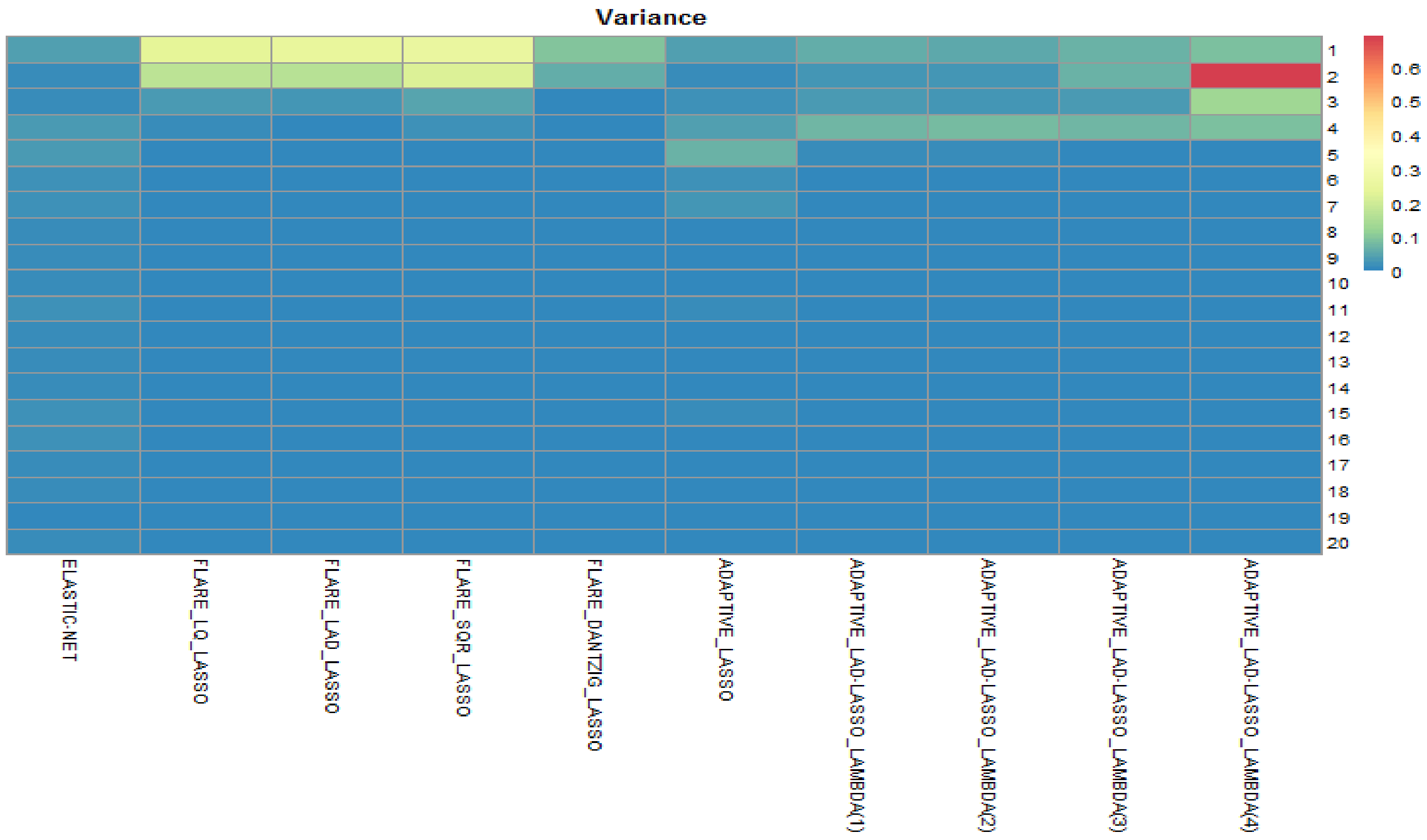
Scenario 3 is defined by:

```
n=100;N=150
beta=c(2,-2,1,-1,0.5,0.2,-0.3,-0.15,rep(0,12))
p=length(beta)
rho=0.8
x=c(1,rho^seq(1:(length(beta)-1)))
x=toeplitz(x)
X=matrix(rmvt(n*p,x,5),nrow=n ,ncol=p)
y=rnorm(n,X%*%beta,2.2)
```

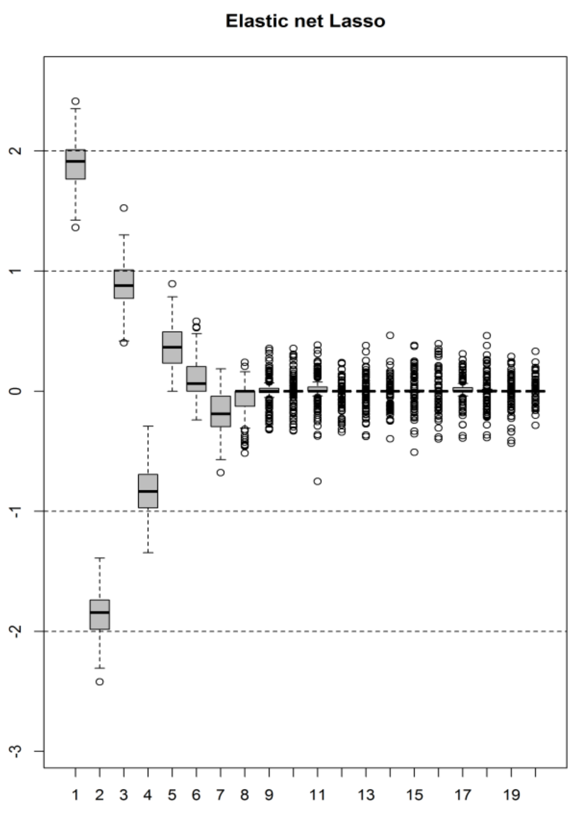
ELASTIC-NET	LQ_LASSO	LAD_LASSO	SQR_LASSO	DANTZIG_LASSO
0.5074423	6.8077823	6.9750984	6.8247627	7.9745115
ADAPTIVE_LASSO	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(3)	ADAPTIVE_LAD-LASSO LAMBDA(4)
0.4468155	1.200836	1.4015227	2.1318310	7.0767288

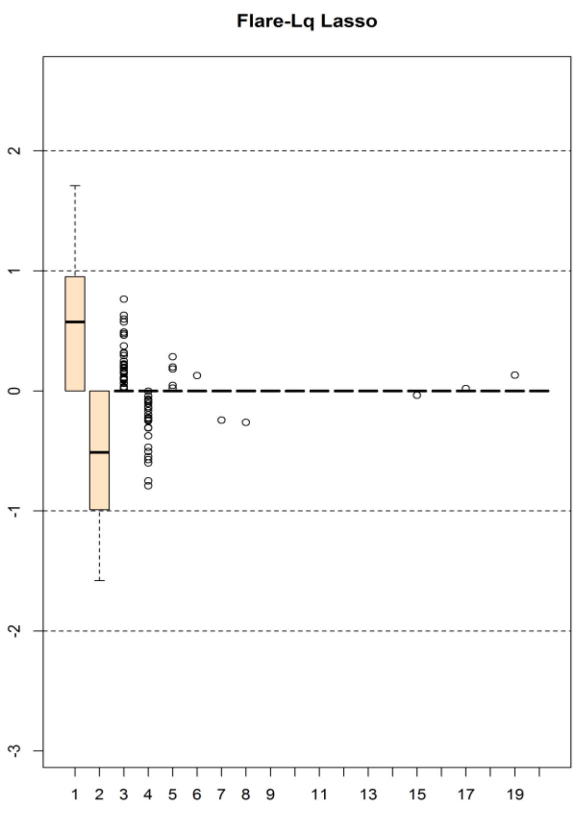


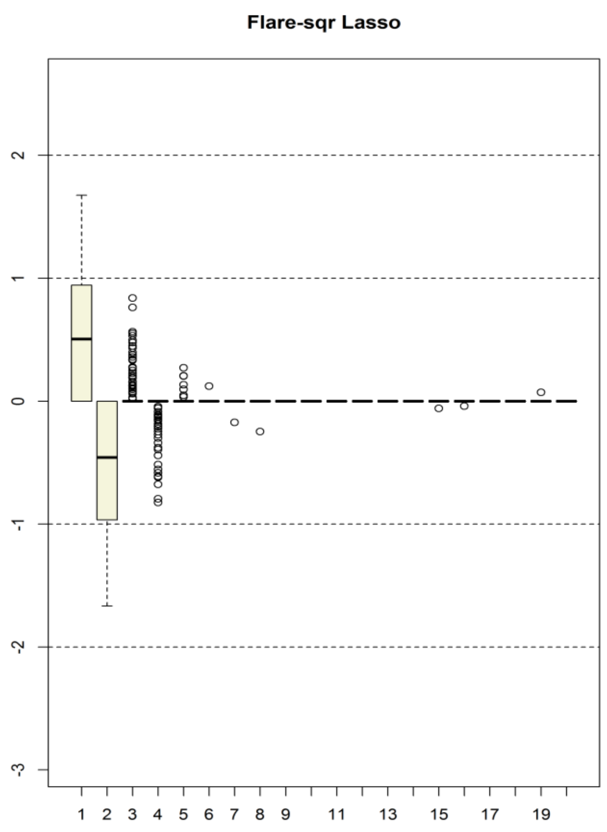
Bias
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
1.5
1
0.5
0
-0.5
-1
-1.5
ELASTIC-NET
FLARE_LQ_LASSO
FLARE_LAD_LASSO
FLARE_SQR_LASSO
FLARE_DANTZIG_LASSO
ADAPTIVE_LASSO
ADAPTIVE_LAD-LASSO_LAMBDA(1)
ADAPTIVE_LAD-LASSO_LAMBDA(2)
ADAPTIVE_LAD-LASSO_LAMBDA(3)
ADAPTIVE_LAD-LASSO_LAMBDA(4)

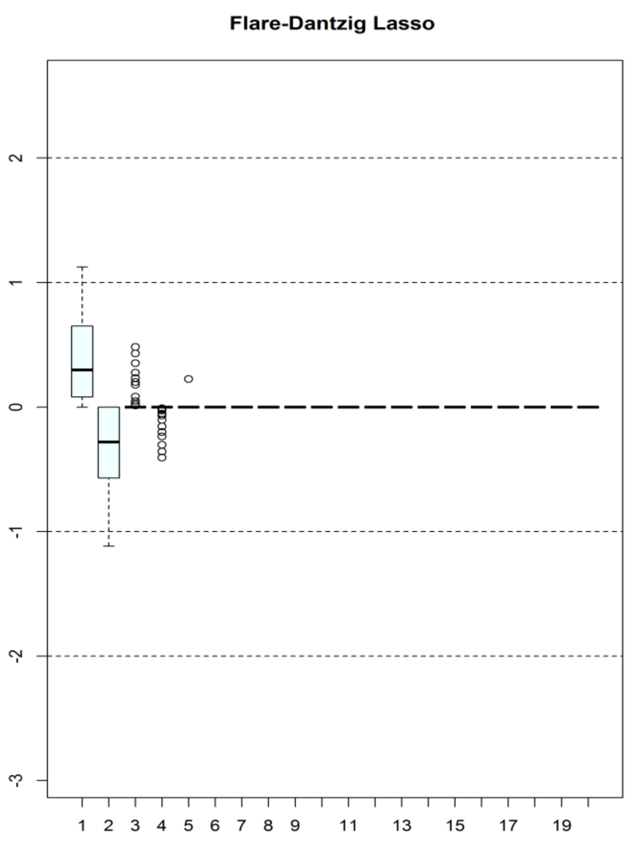


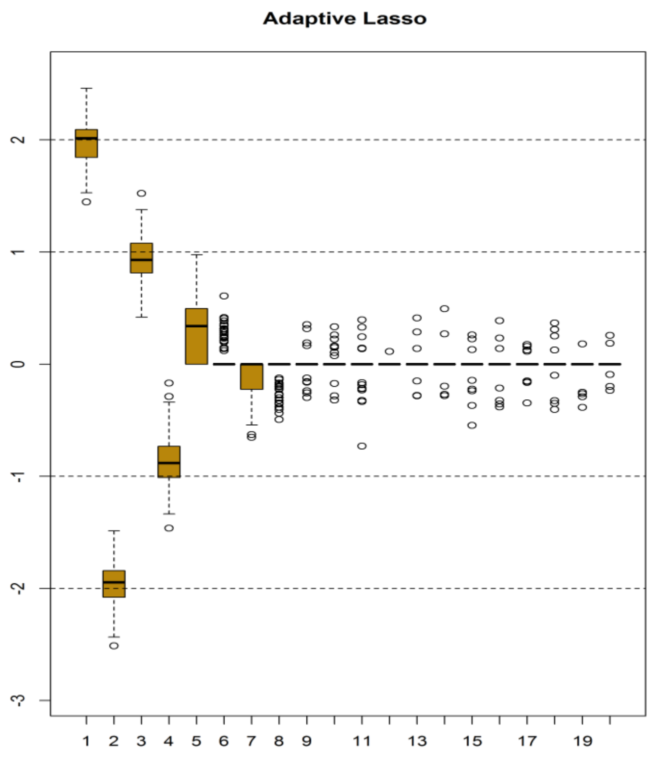
Variance
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
0.6
0.5
0.4
0.3
0.2
0.1
0
ELASTIC-NET
FLARE_LQ_LASSO
FLARE_LAD_LASSO
FLARE_SQR_LASSO
FLARE_DANTZIG_LASSO
ADAPTIVE_LASSO
ADAPTIVE_LAD-LASSO_LAMBDA(1)
ADAPTIVE_LAD-LASSO_LAMBDA(2)
ADAPTIVE_LAD-LASSO_LAMBDA(3)
ADAPTIVE_LAD-LASSO_LAMBDA(4)

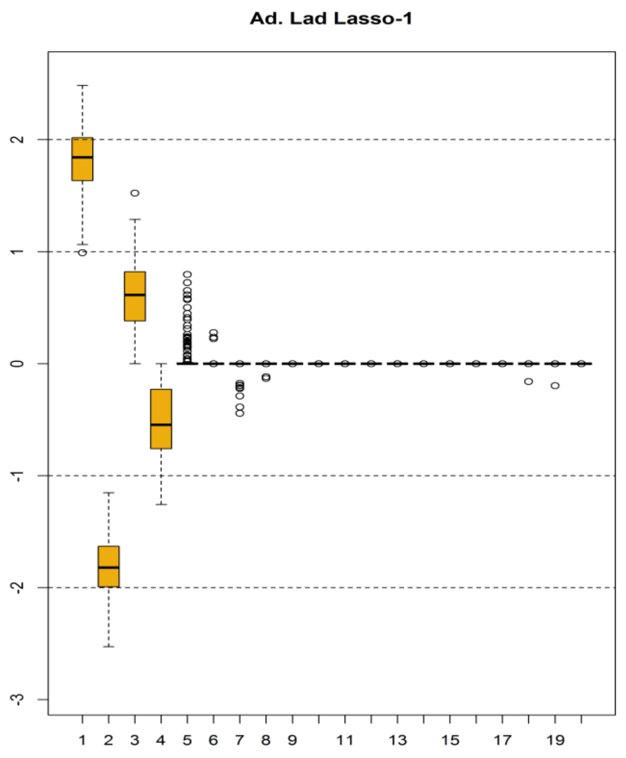

Elastic net Lasso
2
1
0
-1
-2
-3
1
2
3
4
5
6
7
8
9
11
13
15
17
19

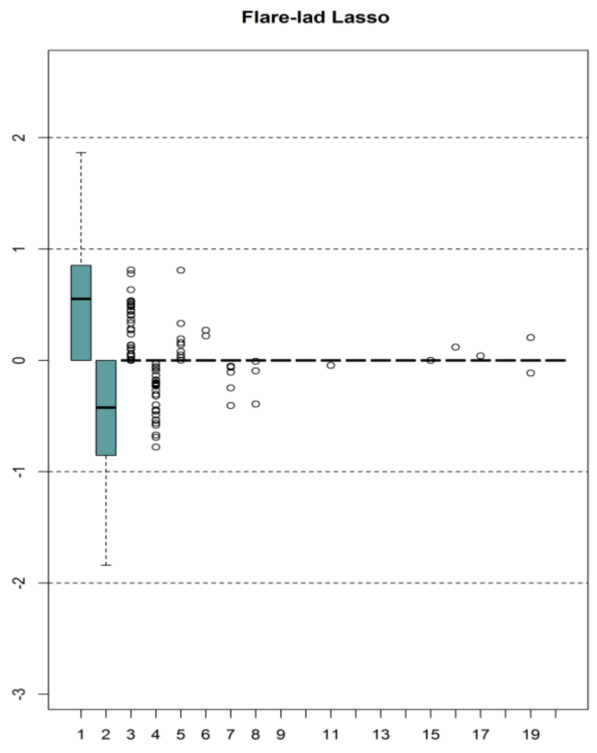

Flare-Lq Lasso
2
1
0
-1
-2
-3
1
2
3
4
5
6
7
8
9
11
13
15
17
19

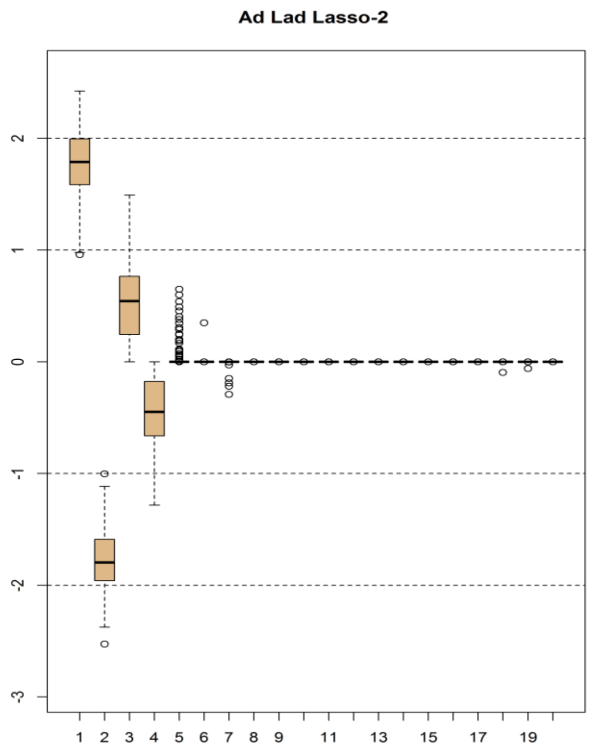

Flare-sqr Lasso
2
1
0
-1
-2
-3
1
2
3
4
5
6
7
8
9
11
13
15
17
19

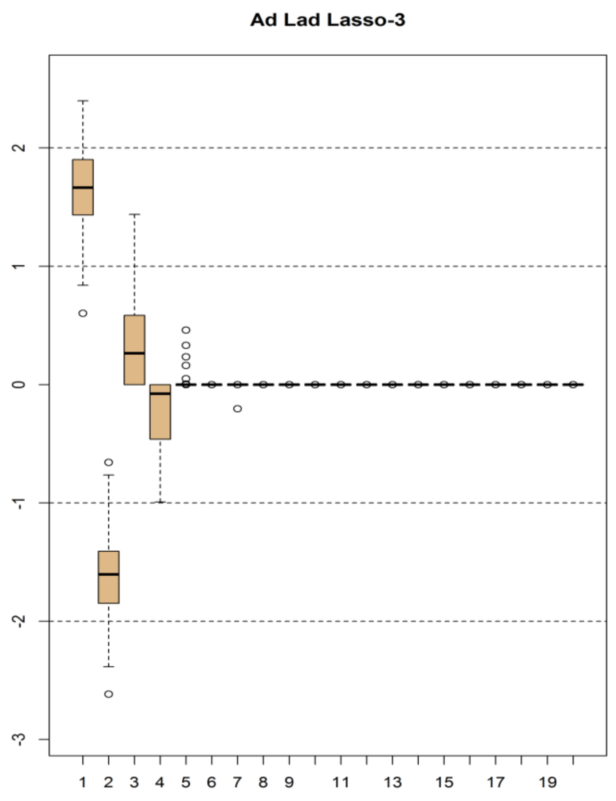

Flare-Dantzig Lasso
2
1
0
-1
-2
-3
1
2
3
4
5
6
7
8
9
11
13
15
17
19

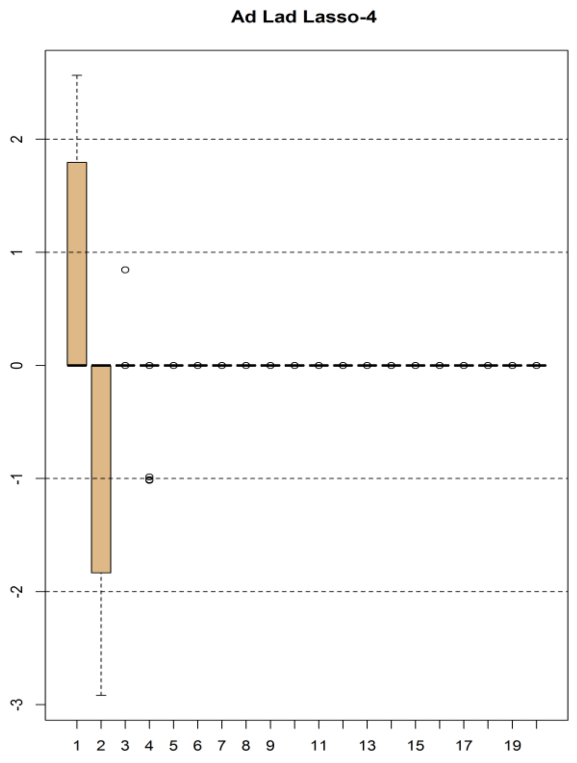

Adaptive Lasso
2
1
0
-1
-2
-3
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Ad. Lad Lasso-1
2
1
0
-1
-2
-3
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Flare-lad Lasso
2
1
0
-1
-2
-3
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Ad Lad Lasso-2
2
1
0
-1
-2
-3
1 2 3 4 5 6 7 8 9 11 13 15 17 19





All

Observations:

- Best performance: Adaptive Lasso
- the flare methods and Lambda4 show high variance
- There is fluctuation between under and overestimating the coefficents in flare.

Scenario 4:HIGH DIMENSIONAL SETTING

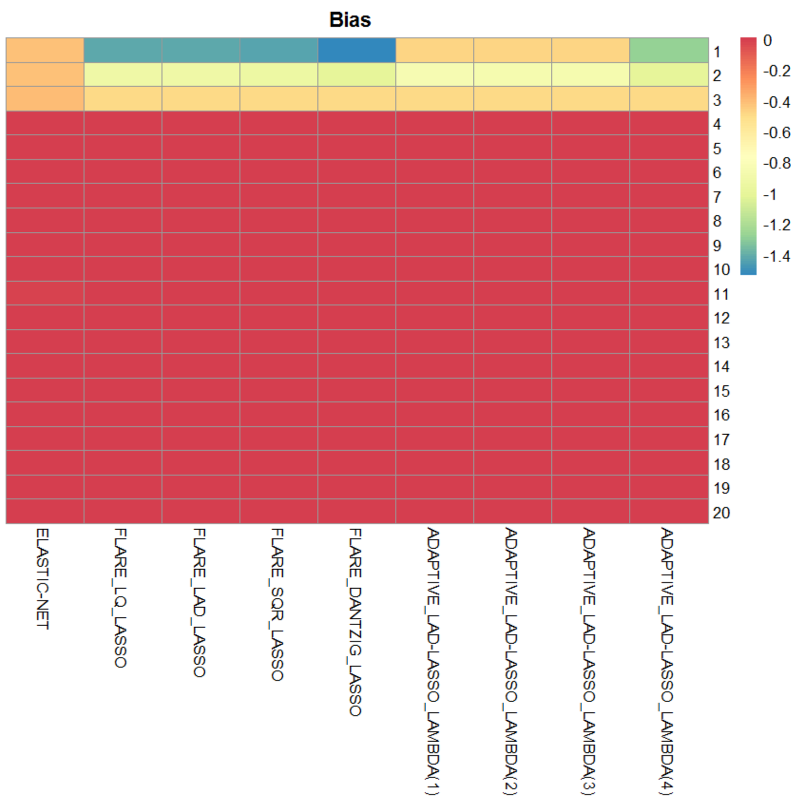
In the High Dimensional case when we have 997 uncontributing predictors and 3 useful predictors. There is a lot of noise. This is a small signal Big noise situation.

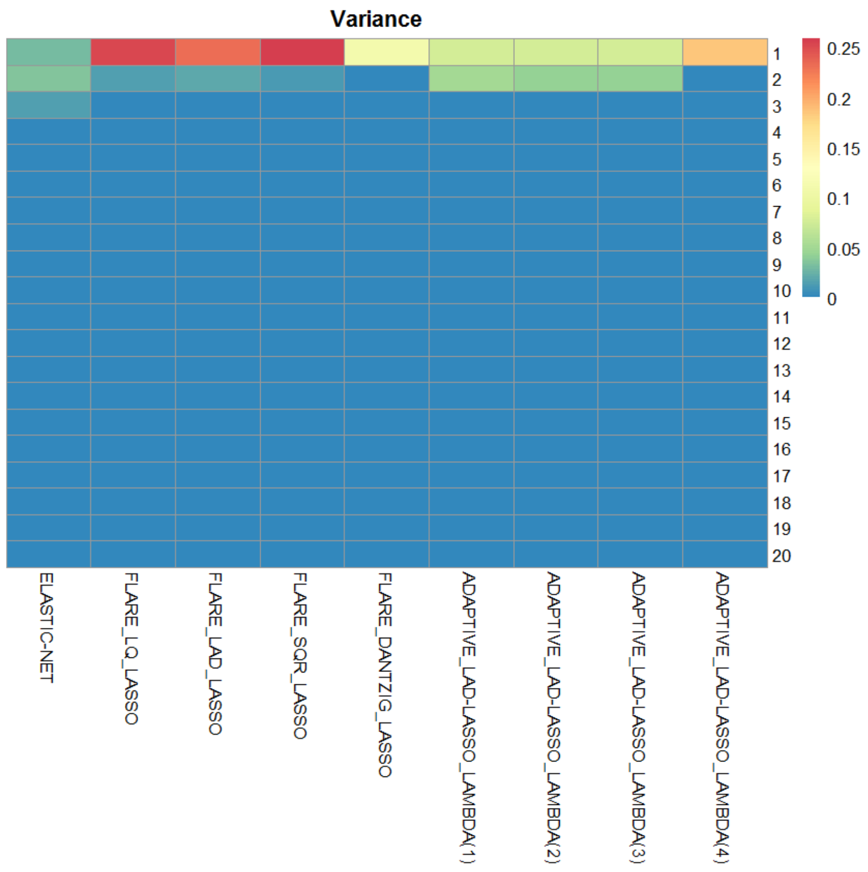
Scenario 4 is defined by:

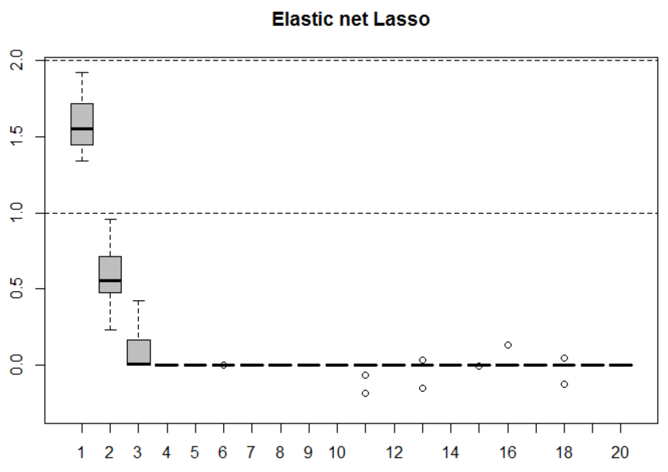
```
n=100;N=10
beta=c(2,1,0.5,rep(0,997)) #1000 predictors
p=length(beta)
X=matrix(rt(n*p,5),nrow=n,ncol=p)
y=rnorm(n,X%*%beta,2)
```

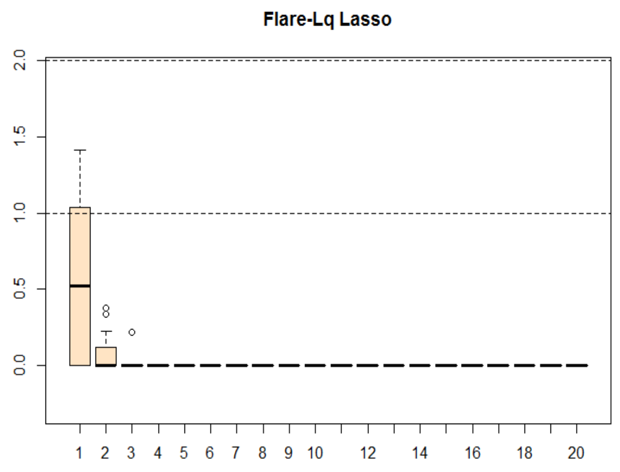
ELASTIC-NET	LQ_LASSO	LAD_LASSO	SQR_LASSO	DANTZIG_LASSO
0.8652662	3.3757059	3.3701016	3.4783493	3.7224816
ADAPTIVE_LASSO	ADAPTIVE_LAD-LASSO_LAMBDA(1)	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(3)	ADAPTIVE_LAD-LASSO LAMBDA(4)
NA	1.3363953	1.3514643	1.3514643	3.0743282

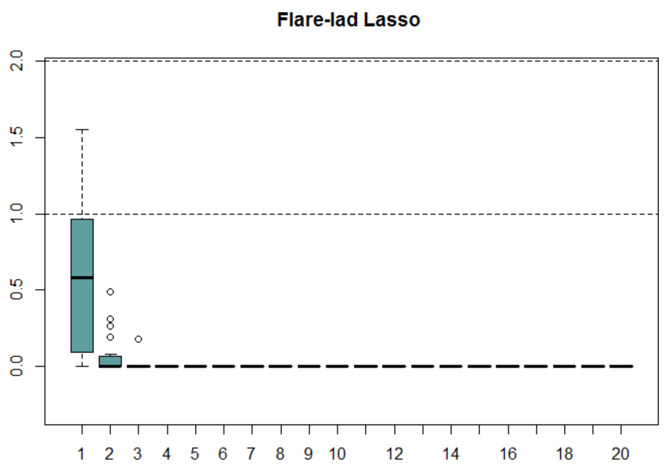
Only the first twenty predictors in high dimensions were considered since rest of them were insignficant and this helps us focus on the useful variables.

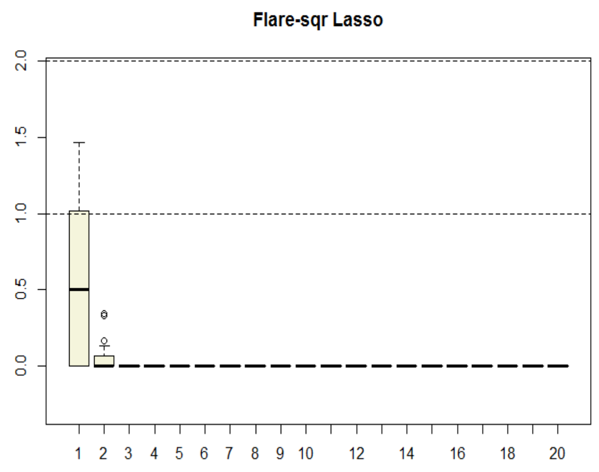


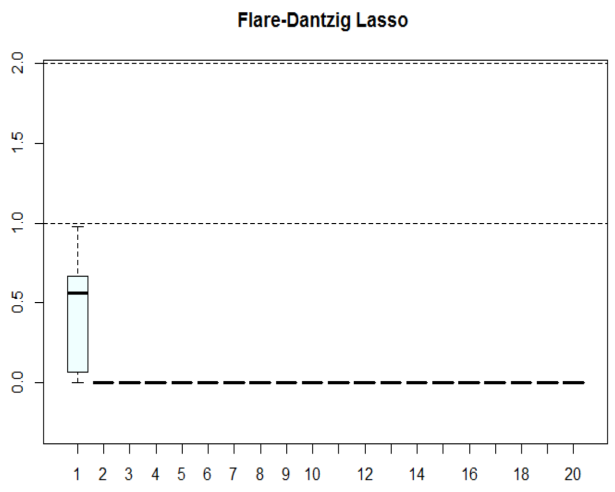


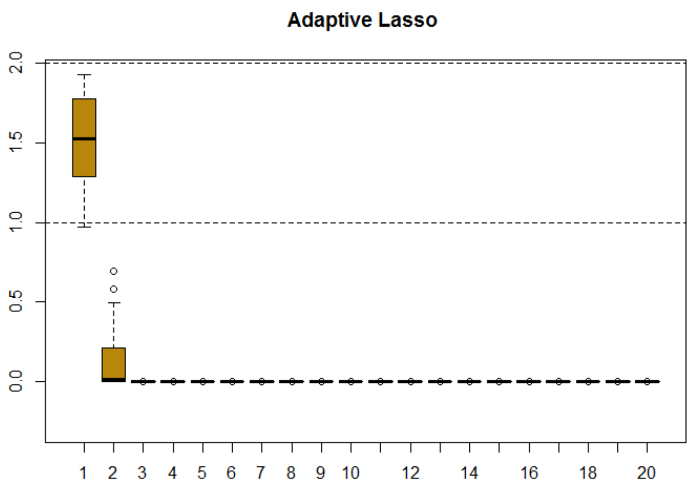

Elastic net Lasso
0.0
0.5
1.0
1.5
2.0
1 2 3 4 5 6 7 8 9 10 12 14 16 18 20

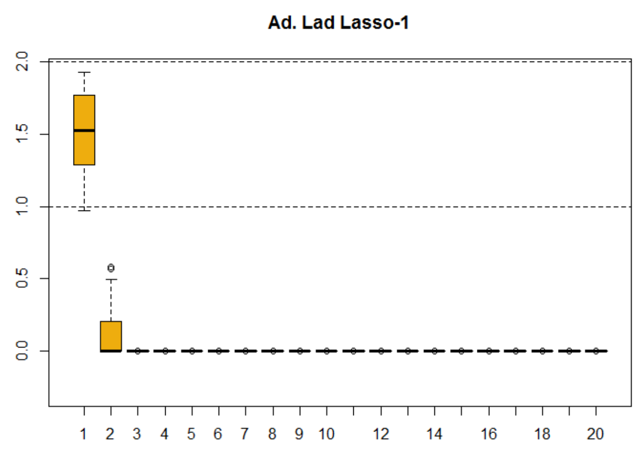

Flare-Lq Lasso
0.0
0.5
1.0
1.5
2.0
1 2 3 4 5 6 7 8 9 10 12 14 16 18 20

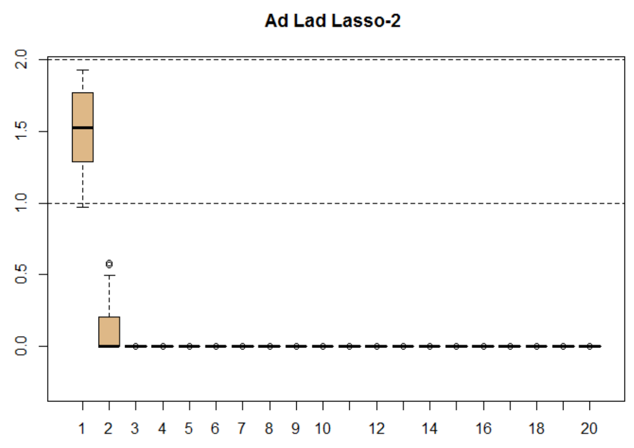

Flare-lad Lasso
0.0
0.5
1.0
1.5
2.0
1 2 3 4 5 6 7 8 9 10 12 14 16 18 20

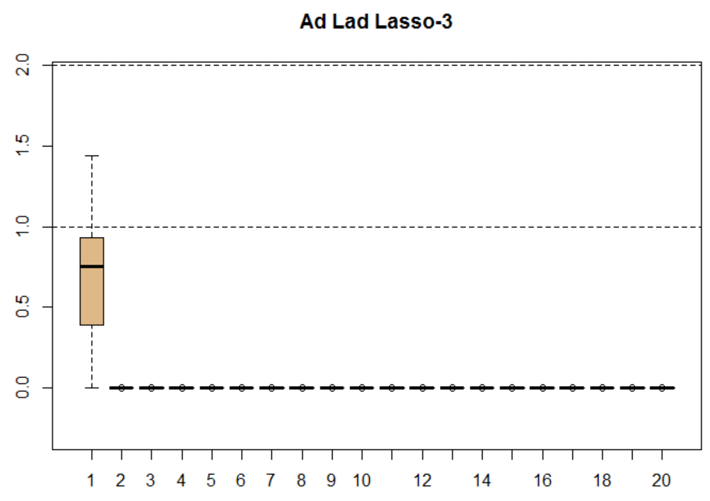

Flare-sqr Lasso
0.0
0.5
1.0
1.5
2.0
1 2 3 4 5 6 7 8 9 10 12 14 16 18 20


Flare-Dantzig Lasso
0.0
0.5
1.0
1.5
2.0
1 2 3 4 5 6 7 8 9 10 12 14 16 18 20


Adaptive Lasso
0.0
0.5
1.0
1.5
2.0
1 2 3 4 5 6 7 8 9 10 12 14 16 18 20







Observations

- Adpative methods have an improved capability of uncovering ‘zeroes’ in comparison to non-adaptive methods. Hence they perform well in high dimensions.
- Adpative methods have higher variance but they do not constrain the predictors to zero when we use the right penalty hence they have lower MSE.

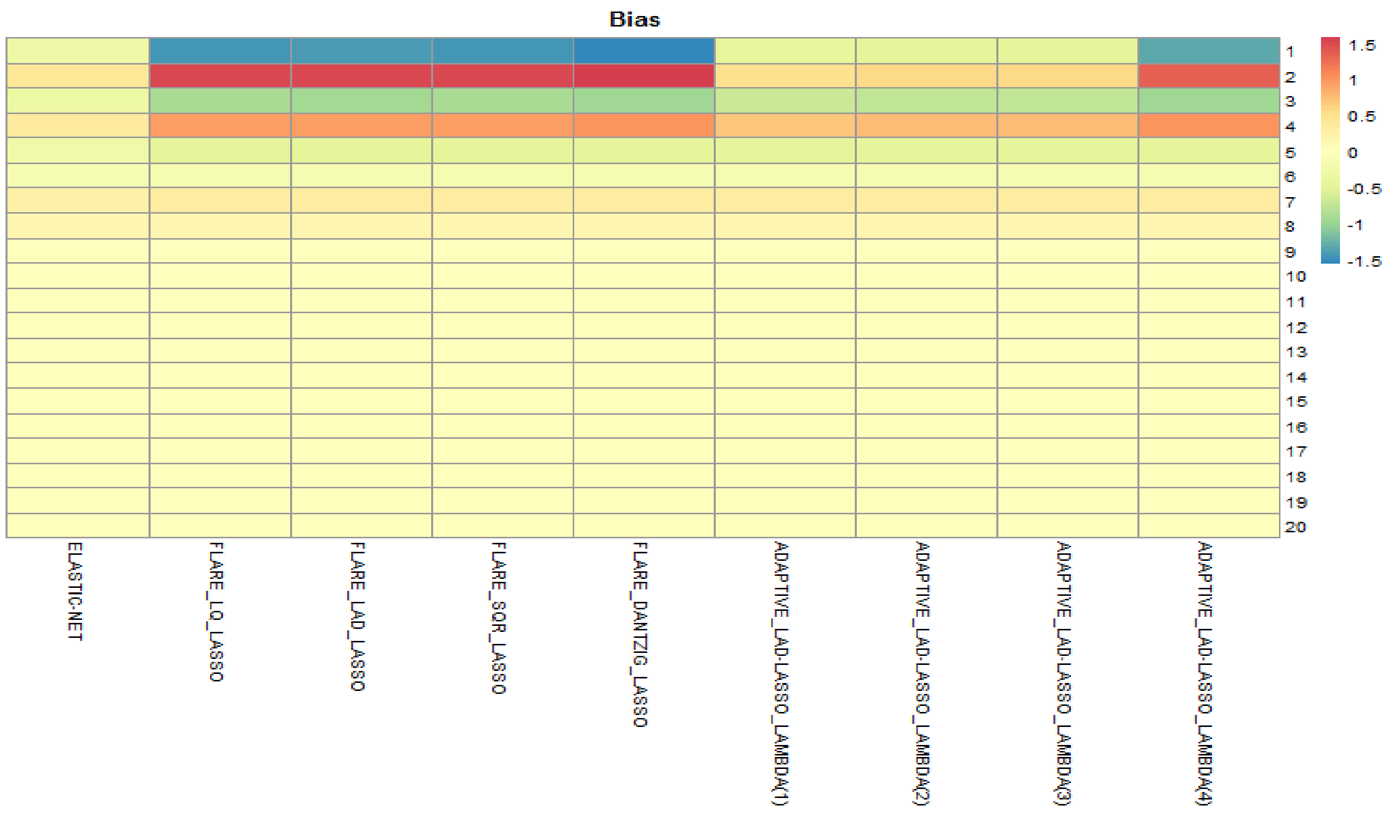
Scenario 5: HIGH DIMENSIONAL SETTING AND STRONGLY CORRELATED

Scenario 5 is defined by:

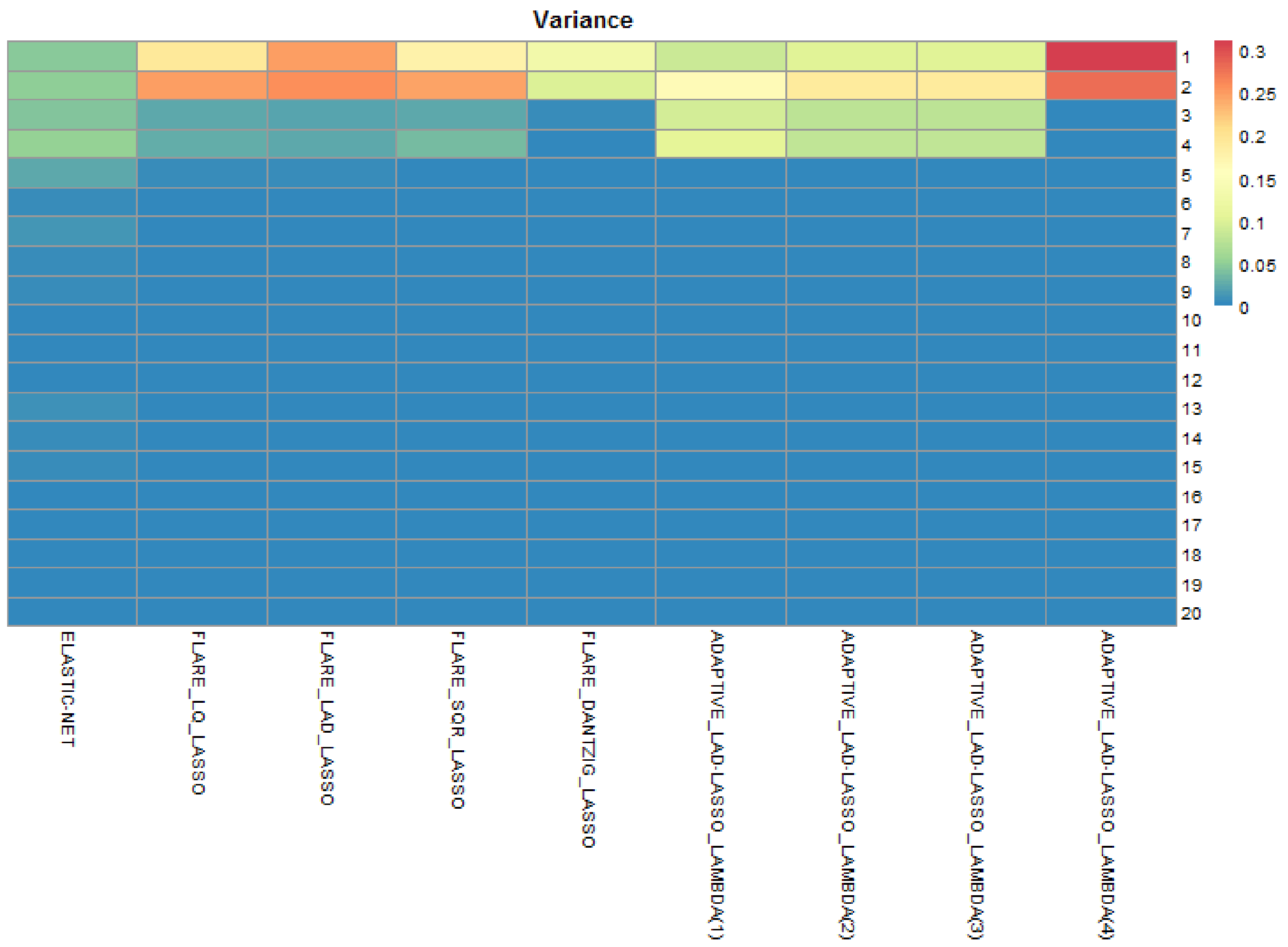
```
rho=0.8
n=10
N=150
beta=c(2,-2,1,-1,0.5,0.2,-0.3,-0.15,rep(0,212))
p=length(beta)
x=c(1,rho^seq(1:(length(beta)-1)))
```

```
x=toeplitz(x)                                                          #create variance matrix
X=matrix(rmvt(n*p,x,5),nrow=n,ncol=p ) # degrees of freedom=5
y=rnorm(n,X%*%beta,2)
```

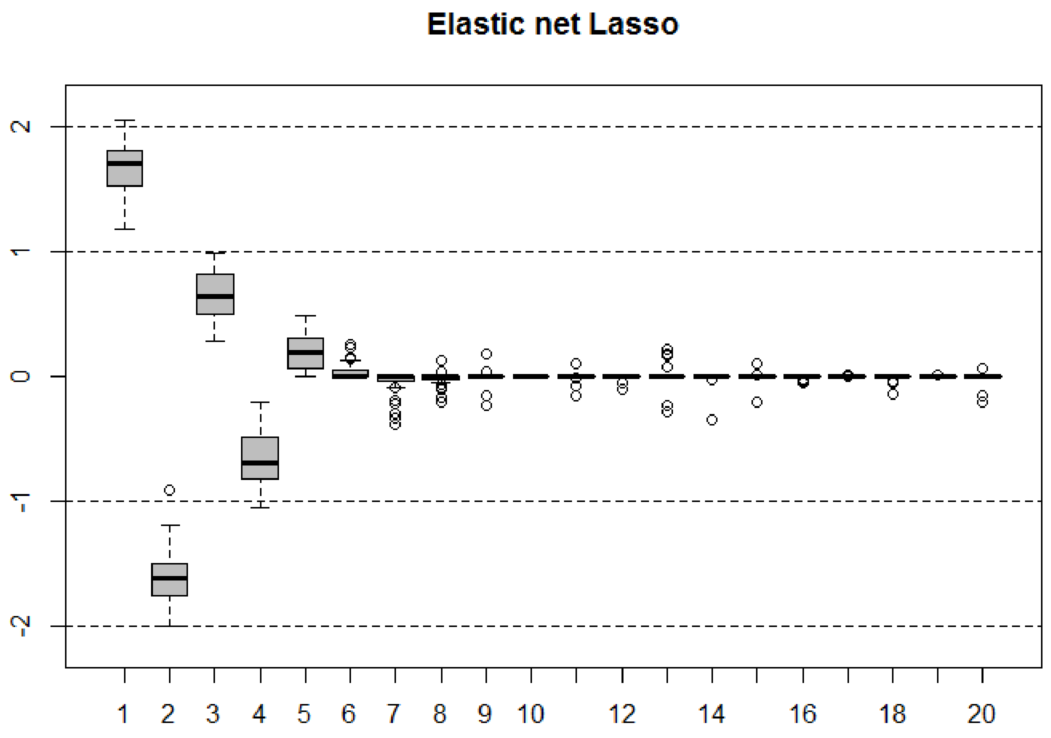
ELASTIC-NET	LQ_LASSO	LAD_LASSO	SQR_LASSO	DANTZIG_LASSO
1.144504	7.561426	7.914729	7.629567	8.085618
ADAPTIVE_LASSO	ADAPTIVE_LAD-LASSO_LAMBDA(1)	ADAPTIVE_LAD-LASSO_LAMBDA(2)	ADAPTIVE_LAD-LASSO_LAMBDA(3)	ADAPTIVE_LAD-LASSO LAMBDA(4)
NA	2.052445	2.441425	2.441425	5.963532



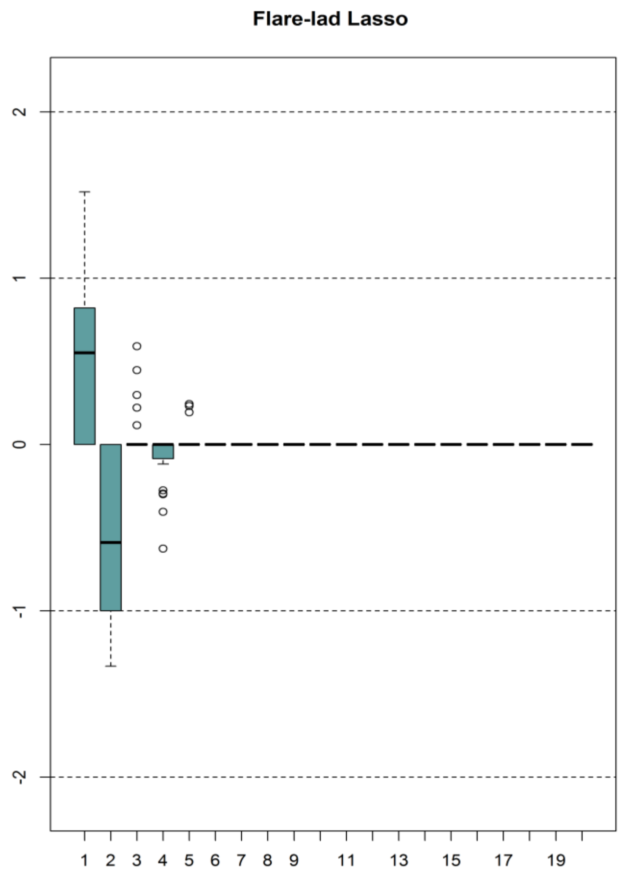
- Elastic net performs well it includes the group of correlated variables in its model.
- Best performance: ADAPTIVE_LAD-LASSO_LAMBDA(1)
- All models tend to show higher variance in high dimensions
- FLARE shows fluctuating bias.

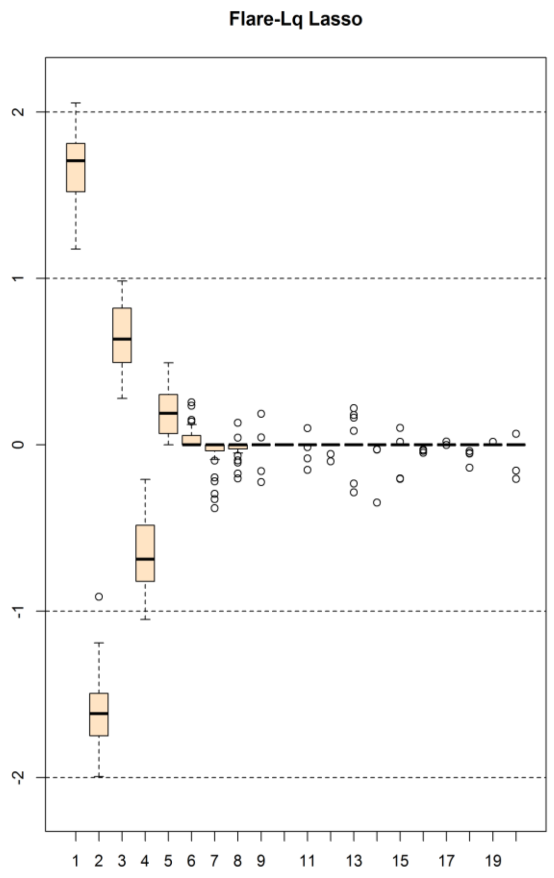


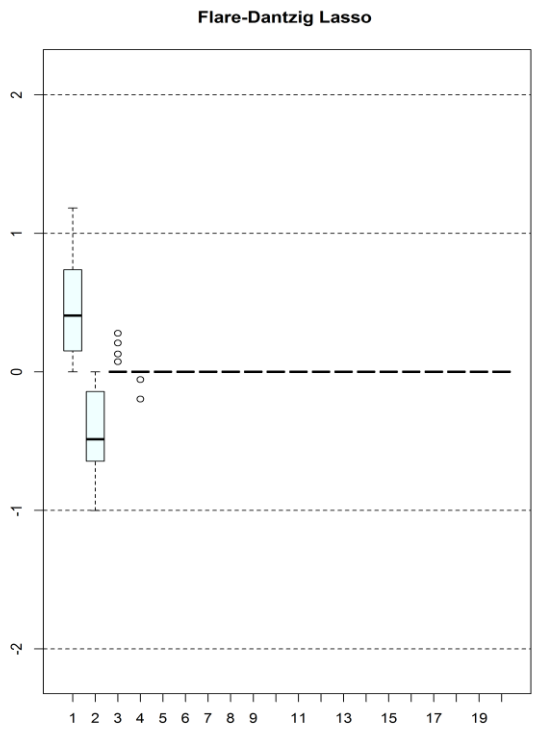
Variance
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
0.3
0.25
0.2
0.15
0.1
0.05
0
ELASTIC-NET
FLARE_LQ_LASSO
FLARE_LAD_LASSO
FLARE_SQR_LASSO
FLARE_DANTZIG_LASSO
ADAPTIVE_LAD-LASSO_LAMBDA(1)
ADAPTIVE_LAD-LASSO_LAMBDA(2)
ADAPTIVE_LAD-LASSO_LAMBDA(3)
ADAPTIVE_LAD-LASSO_LAMBDA(4)

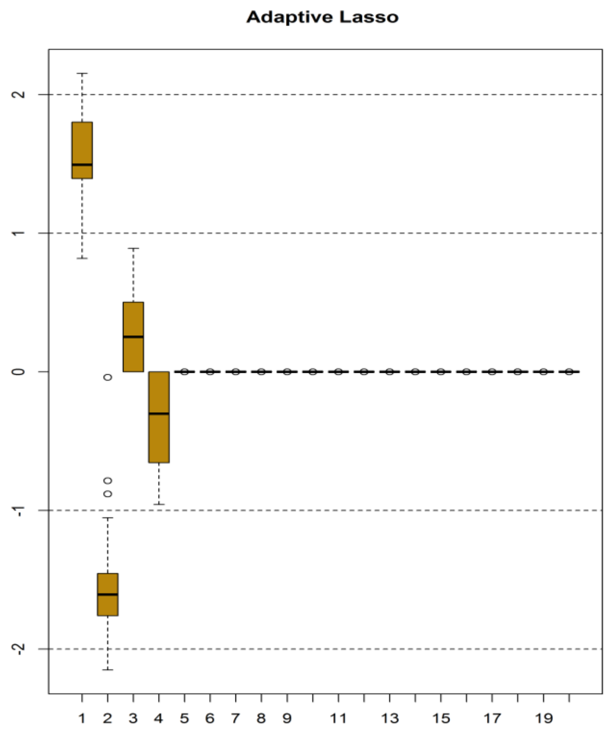


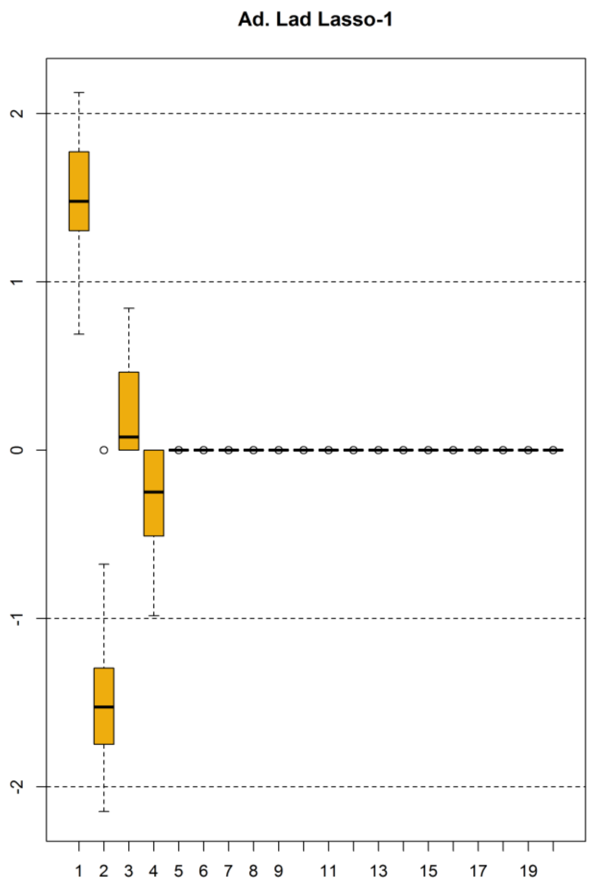
Elastic net Lasso
2
1
0
-1
-2
1
2
3
4
5
6
7
8
9
10
12
14
16
18
20

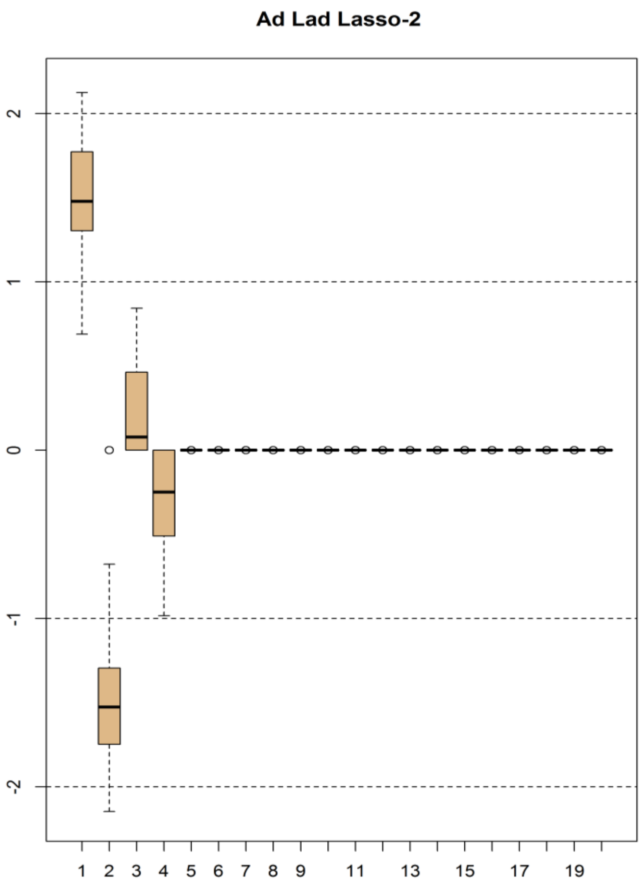

Flare-lad Lasso
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19

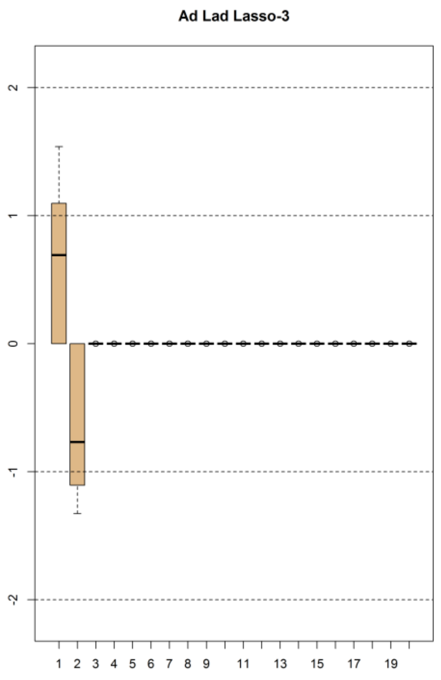

Flare-Lq Lasso
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Flare-Dantzig Lasso
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Adaptive Lasso
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Ad. Lad Lasso-1
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Ad Lad Lasso-2
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19


Ad Lad Lasso-3
2
1
0
-1
-2
1 2 3 4 5 6 7 8 9 11 13 15 17 19

Scenario	**BEST Performer**	**Second Best Performer**
One	Adaptive Lasso	Elastic Net
Two	Elastic Net	Dantzig
Three	Adaptive Lasso	Elastic Net
Four	Elastic Net	Adaptive LAD Lasso lambda1
Five	Elastic Net	Adaptive LAD Lasso lambda1

General observations

- ADAPTIVE_LAD-LASSO LAMBDA 4 was tending to kill all the significant varibales in lower dimensions. Hence it did very badly for scenario 1,2,3 and was expected to perform well in higher dimensions. But it failed to impress even when p>>n.
- The best performing elastic net showed some traits of Ridge. It brought all the coefficents very close to zero but never actually zero. If LASSO was chosen in some case (when alpha=1) all the coefficents would have been zero.
- The use of smaller penalties in adaptive LAD Lasso gave good models.
- Adaptive methods and Elastic net did quite well when there was high correlation – in scenarios 3 and 5
- Adaptive methods and elastic net did well in higher dimensions -in scenarios 4 and 5.
- Elastic net regression with dual-crossvalidation is very powerful since it can perform at worst as good as the lasso or ridge.

More Analysis:

Signal to noise ratio

- Now that we have taken a look at all the 5 5cenarios for reccomended values we can makes some changes to see the effect on MSE.
- We reduce the noise in the response represented by sigma to 0.5 and observe the results.
- We see that as expected the average MSE of almost all the methods have gone down. But the best performer(shown in yellow) remain the same.
- In the High Dimensional case when we have 997 uncontributing predictors and 3 useful predictors. There is a lot of noise by default.

MSE Values for various scenarios before and after changing sigma

	SCENARIO 4		**SCENARIO 5**	
	SIGMA	**SIGMA**	**SIGMA**	**SIGMA**
methods	**2**	**0.5**	**2**	
ELASTIC-NET	0.865266	0.0518	1.144504	0.093653
FLARE_LQ_LASSO	3.375706	3.172507	7.561426	6.871671
FLARE_LAD_LASSO	3.370102	3.181188	7.914729	7.346537
FLARE_SQR_LASSO	3.478349	3.274706	7.629567	6.796996
FLARE_DANTZIG_LASSO	3.722482	3.725129	8.085618	7.875244
ADAPTIVE_LASSO	NA	NA	NA	NA
ADAPTIVE_LAD-LASSO_LAMBDA(1)	1.336395	0.311968	2.052445	0.596753
ADAPTIVE_LAD-LASSO_LAMBDA(2)	1.351464	0.37505	2.441425	0.743439
ADAPTIVE_LAD-LASSO_LAMBDA(3)	1.351464	0.667588	2.441425	1.652156
ADAPTIVE_LAD-LASSO_LAMBDA(4)	3.074328	1.593767	5.963532	3.668061

	SCENARIO 1		**SCENARIO 2**		**SCENARIO 3**	
	SIGMA	**SIGMA**	**SIGMA**	**SIGMA**	**SIGMA**	**SIGMA**
methods	**2.2**	**0.5**	**2**	**0.5**	**2**	**0.5**
ELASTIC-NET	0.320877	0.021502	0.499826	0.02949	0.507442	0.55466
FLARE_LQ_LASSO	6.606022	7.412725	0.757834	7.054982	6.807782	6.323844
FLARE_LAD_LASSO	6.625782	7.518369	0.760539	7.520957	6.975098	6.76379
FLARE_SQR_LASSO	6.720175	7.456598	0.747768	7.137033	6.824763	6.319848
FLARE_DANTZIG_LASSO	7.12659	7.284979	0.728479	7.767765	7.974512	7.592256
ADAPTIVE_LASSO	0.174311	0.008715	0.74624	0.022132	0.446816	0.51146
ADAPTIVE_LAD-LASSO_LAMBDA(1)	0.622954	0.036392	0.794962	0.347489	1.200836	1.290749
ADAPTIVE_LAD-LASSO_LAMBDA(2)	0.781596	0.048993	0.797359	0.439152	1.401523	1.542271
ADAPTIVE_LAD-LASSO_LAMBDA(3)	1.41585	0.109539	0.799841	0.754523	2.131831	2.341216
ADAPTIVE_LAD-LASSO_LAMBDA(4)	6.129742	1.191511	0.8	2.261269	7.076729	3.890899

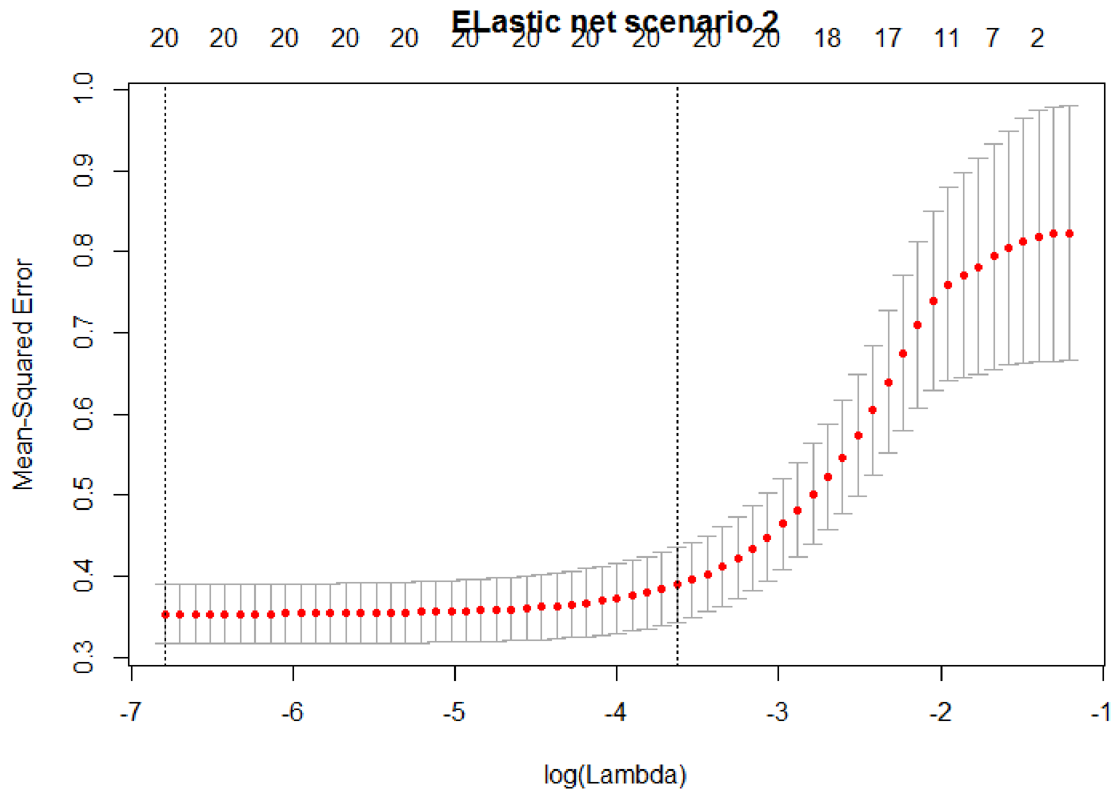
Elastic net performance

This is to understand which aspect of elastic net was giving the best performance whether it was Lasso, ridge or somwehere in between that gave the lowest MSE .

Testing on Unseen Data(taking 2/3 test and 1/3 training data)

```
train_rows <- sample(1:n, .66*n)
x.train <- x[train_rows, ]
x.test <- x[-train_rows, ]
y.train <- y[train_rows]
y.test <- y[-train_rows]
fit    <-    cv.glmnet(x.train,    y.train,    type.measure="mse",    alpha=i/10,
family="gaussian")
MSE <- mean((y.test - Yhat)^2)
```

After double crossvalidation original scenario 2.



Scenario 2 This is the case where all the betas are equal. **Ridge is the best**

alpha values	**MSE**	How good?
α=0 (**Ridge)**	0.361964306	BEST
α=0.1	0.363265052	
α=0.2	0.36362691	
α=0.3	0.363795051	
α=0.4	0.363919033	
α=0.5	0.363985191	
α=0.6	0.364077181	
α=0.7	0.364118083	
α=0.8	0.364149442	
α=0.9	0.364231461	
α=1 (**LASSO**)	0.364231461	WORST

Scenario 3: highly correlated values: **Lasso is the best**

alpha values	**MSE**	How good?
α=0(**Ridge**	5.611247461	WORST
α=0.1	5.327628207	
α=0.2	5.162226367	
α=0.3	5.057578179	
α=0.4	5.003720325	
α=0.5	4.966464154	
α=0.6	4.94313986	
α=0.7	4.927355201	
α=0.8	4.914193935	
α=0.9	4.903475592	
α=1(**LASSO**)	4.903475592	BEST