

# An Introduction to `islasso`

Gianluca Sottile

Giovanna Cilluffo

Vito M.R. Muggeo

Novembre 15, 2021

## Contents

<b>Abstract</b>	<b>1</b>
<b>Introduction</b>	<b>1</b>
<b>The R functions</b>	<b>1</b>
<b>A worked example: the Diabetes data set</b>	<b>2</b>
<b>References</b>	<b>13</b>

## Abstract

In this short note we present and briefly discuss the R package `islasso` dealing with regression models having a large number of covariates. Estimation is carried out by penalizing the coefficients via a quasi-lasso penalty, wherein the nonsmooth lasso penalty is replaced by its smooth counterpart determined iteratively by data according to the induced smoothing idea. The package includes functions to estimate the model and to test for linear hypothesis on linear combinations of relevant coefficients. We illustrate R code throughout a worked example, by avoiding intentionally to report details and extended bibliography.

## Introduction

Let  $\mathbf{y} = \mathbf{X}\beta + \epsilon$  be the linear model of interest with usual zero-means and homoscedastic errors. As usual,  $\mathbf{y} = (y_1, \dots, y_n)^T$  is the response vector,  $\mathbf{X}$  is the  $n \times p$  design matrix (having  $p$  quite large) with regression coefficients  $\beta$ . When interest lies in selecting the non-noise covariates and estimating the relevant effect, one assumes the lasso penalized objective function (Tibshirani, 1996),

$$\frac{1}{2} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda \|\beta\|_1$$

## The R functions

The main function of the package are `islasso()` where the user supplies the model formula as in the usual `lm` or `glm` functions, i.e.

```
islasso(formula, family = gaussian, lambda, alpha = 1, data, weights, subset, offset,
        unpenalized, contrasts = NULL, control = is.control())
```

and *islasso.path* used to fit the regularization path via the induced smoothed lasso framework, i.e.

```
islasso.path(formula, family = gaussian, lambda = NULL, nlambda = 100, lambda.min.ratio = ifelse(nobs <
  nvars, 0.001, 1e-05), alpha = 1, data, weights, subset, offset, unpenalized,
  contrasts = NULL, control = is.control())
```

*family* accepts specification of family and link function as in Table 1, *lambda* is the tuning parameter, *alpha* is elastic-net mixing parameter, *nlambda* is the number of lambda values, *lambda.min.ratio* is the smallest value for lambda (as a fraction of lambda.max), and *unpenalized* allows to indicate covariates with unpenalized coefficients.

**Table 1. Families and link functions allowed in islasso**

family	link
gaussian	identity
binomial	logit, probit
poisson	log
gamma	identity, log, inverse

The fitter functions are *islasso.fit()* and *islasso.path.fit()* which reads as

```
islasso.fit(X, y, family = gaussian(), lambda, alpha = 1, intercept = FALSE, weights = NULL,
  offset = NULL, unpenalized = NULL, control = is.control())
```

and

```
islasso.path.fit(X, y, family = gaussian(), lambda, nlambda, lambda.min.ratio, alpha = 1,
  intercept = FALSE, weights = NULL, offset = NULL, unpenalized = NULL, control = is.control())
```

whose actually implements the estimating algorithm as described in the paper. The *lambda* argument in *islasso.fit* and *islasso* specifies the positive tuning parameter in the penalized objective. Any non-negative value can be provided, but if missing, it is computed via *K*-fold cross validation by the function *cv.glmnet()* from package **glmnet**. The number of folds being used can be specified via the argument *nfolds* of the auxiliary function *is.control()*. The *lambda* argument in *islasso.path.fit* and *islasso.path* specifies the sequence of positive tuning parameters, user supplied or automatically computed based on *nlambda* and *lambda.min.ratio*.

## A worked example: the Diabetes data set

We use the well-known **diabetes** dataset available in the **lars** package. The data refer to  $n = 442$  patients enrolled to investigate a measure of disease progression one year after the baseline. There are ten covariates, (age, sex, bmi (body mass index), map (average blood pressure) and several blood serum measurements (tc, ldl, hdl, tch, ltg, glu). The matrix  $x^2$  in the dataframe also includes second-order terms, namely first-order interactions between covariates, and quadratic terms for the continuous variables.

To select the important terms in the regression equation we could simply apply the lasso using the R package **glmnet**

```
library(islasso)

data("diabetes", package = "islasso")

a1 <- with(diabetes, cv.glmnet(x2, y))
n <- nrow(diabetes)
a1$lambda.min * n
```

```
> [1] 1344.186
```

```
b <- drop(coef(a1, "lambda.min", exact = TRUE))
length(b[b != 0])
```

```
> [1] 15
```

Ten-fold cross validation “selects”  $\lambda = 1344.186$ . corresponding to 15 non null coefficients

```
names(b[b != 0])
```

```
> [1] "(Intercept)" "sex"          "bmi"          "map"          "hdl"
> [6] "ltg"           "glu"          "age^2"        "bmi^2"        "glu^2"
> [11] "age:sex"       "age:map"      "age:ltg"      "age:glu"      "bmi:map"
```

The last six estimates are

```
tail(b[b != 0])
```

```
>      glu^2    age:sex    age:map    age:ltg    age:glu    bmi:map
> 69.599081 107.479925 29.970061  8.506032 11.675332 85.530937
```

A reasonable question is if all the “selected” coefficients are significant in the model. Unfortunately lasso regression does not return standard errors due to nonsmoothness of objective, and some alternative approaches have been proposed., including the (Lockhart et al., 2013). Among the (few) strategies, including the ‘covariance test’, the ‘post-selection inference’ and the ‘(modified) residual bootstrap’, here we illustrate the R package **islasso** implementing the recent ‘quasi’ lasso approach based on the induced smoothing idea (Brown and Wang, 2005) as discussed in Cilluffo et al. (2019)

While the optimal lambda could be selected (without supplying any value to *lambda*), we use optimal value minimizing a specific criterion chosen between AIC, BIC, AICc, BIC, GCV or GIC. From version 1.4.0 of the R package **islasso** optimal strategy is to built the regularization path

```
out <- islasso.path(y ~ x2, data = diabetes, nlambda = 30L)
out
```

```
>
> Call:
> islasso.path(formula = y ~ x2, nlambda = 30L, data = diabetes)
>
> Coefficients:
>      lambda      df      phi    deviance    logLik
> 1    0.0096 63.9746 2825.8687 1068250.0676 -2349.8100
```

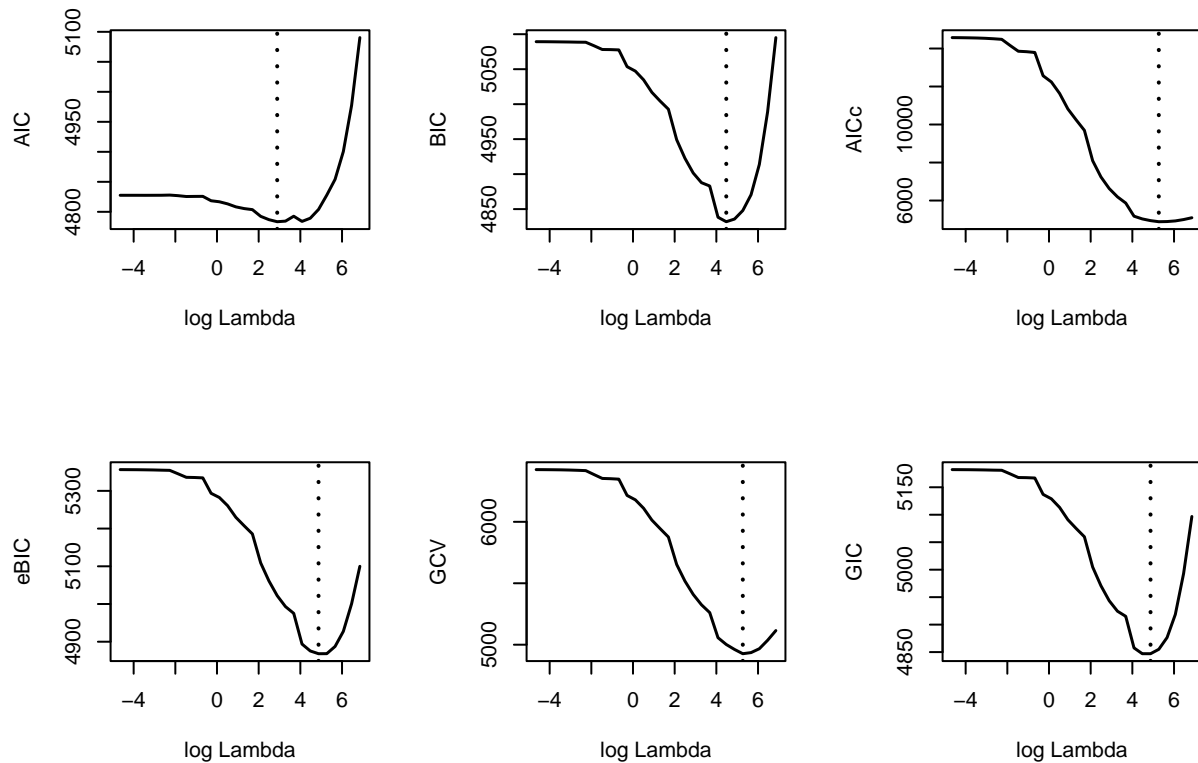
```

> 2    0.0143 63.9617 2825.8380 1068275.0565 -2349.8152
> 3    0.0212 63.9418 2825.8335 1068329.5460 -2349.8265
> 4    0.0315 63.9110 2825.9155 1068447.6038 -2349.8509
> 5    0.0469 63.8632 2826.2291 1068701.1497 -2349.9033
> 6    0.0697 63.7908 2827.1134 1069240.3444 -2350.0148
> 7    0.1036 63.6954 2829.4514 1070394.4412 -2350.2532
> 8    0.1540 62.7071 2828.4990 1072829.4824 -2350.7554
> 9    0.2290 61.7457 2824.6327 1074078.8282 -2351.0126
> 10   0.3405 61.6696 2825.9734 1074803.4980 -2351.1616
> 11   0.5063 61.5412 2827.3134 1075676.3710 -2351.3411
> 12   0.7528 57.4755 2802.4977 1077628.9025 -2351.7418
> 13   1.1193 56.3378 2797.8701 1079032.8088 -2352.0296
> 14   1.6643 54.1428 2787.5392 1081167.2561 -2352.4663
> 15   2.4745 51.0020 2773.3184 1084361.9345 -2353.1184
> 16   3.6792 48.6396 2768.5883 1089053.0990 -2354.0724
> 17   5.4705 46.2344 2770.5193 1096476.3647 -2355.5737
> 18   8.1337 38.2892 2745.4988 1108387.5451 -2357.9615
> 19  12.0935 33.1720 2739.9608 1120172.5610 -2360.2989
> 20  17.9812 28.8615 2743.3998 1133403.9865 -2362.8940
> 21  26.7351 25.2455 2770.4703 1154606.0227 -2366.9900
> 22  39.7509 22.0821 2841.4537 1193177.1836 -2374.2521
> 23  59.1033 13.2863 2839.7888 1217456.2782 -2378.7040
> 24  87.8773 10.4271 2892.3610 1248264.6332 -2384.2269
> 25 130.6596  7.7723 3007.4993 1305939.6066 -2394.2091
> 26 194.2702  4.8935 3195.3473 1396706.9730 -2409.0591
> 27 288.8491  4.0324 3396.8386 1487705.1863 -2423.0081
> 28 429.4729  3.2662 3779.3045 1658108.6614 -2446.9739
> 29 638.5583  2.8155 4502.7339 1977531.0001 -2485.9078
> 30 949.4353  1.1610 5835.2820 2572420.1586 -2544.0304

```

and then to choose the best tuning parameter through the one of the criteria listed above using the function *GoF.islasso.path*, e.g.,

```
lmb.best <- GoF.islasso.path(out)
```

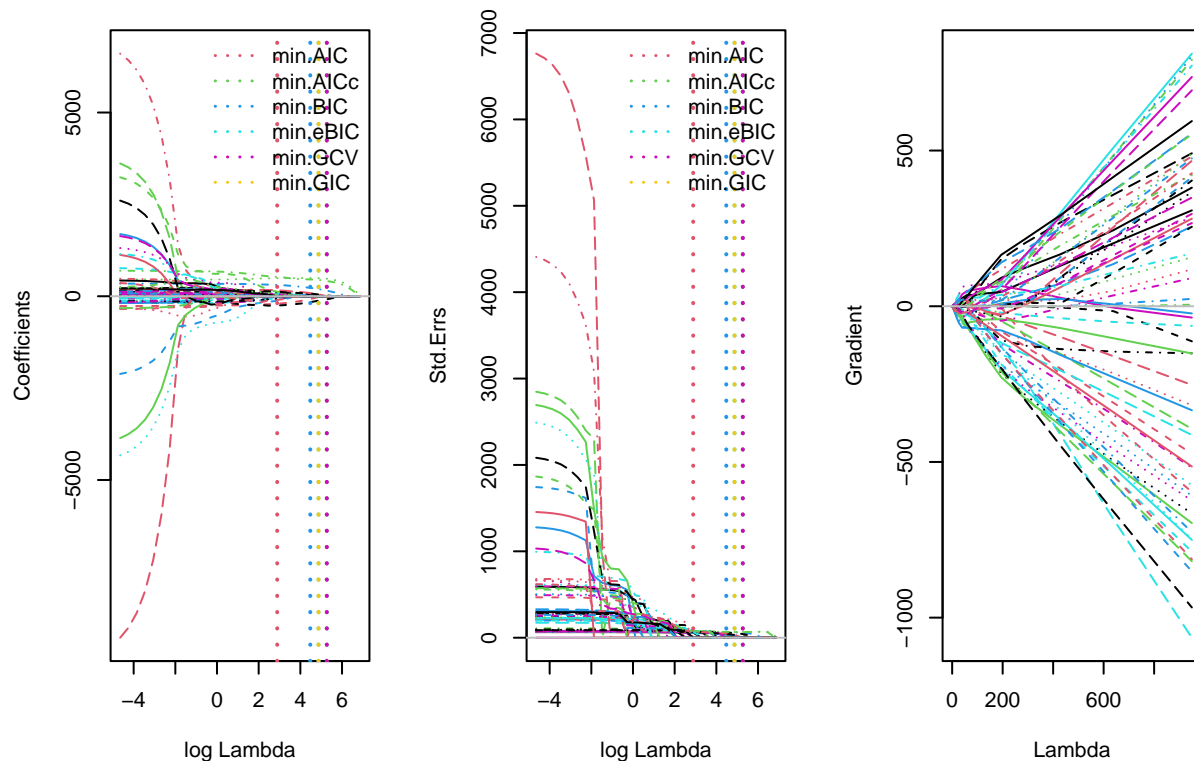


```
lmb.best$lambda.min
```

```
>      AIC      BIC      AICc      eBIC      GCV      GIC
> 17.98115  87.87729 194.27022 130.65964 194.27022 130.65964
```

Using also the regularization path is very useful to have more insights about coefficients, standard errors and gradient profile

```
par(mfrow = c(1, 3))
plot(out, yvar = "coefficients")
plot(out, yvar = "se")
plot(out, yvar = "gradient")
```



Once selected the best lambda value minimizing for example the BIC criterion, the last step of the strategy consists on fitting a new islasso model.

```
lambda.bic <- lmb.best$lambda.min["BIC"]
out2 <- islasso(y ~ x2, data = diabetes, lambda = lambda.bic)
out2
```

```
>
> Call:
> islasso(formula = y ~ x2, lambda = lambda.bic, data = diabetes)
>
> Coefficients:
> (Intercept)      x2age      x2sex      x2bmi      x2map      x2tc
>    152.13         0.00    -76.08    494.73    228.28         0.00
>      x2ldl      x2hdl      x2tch      x2ltg      x2glu      x2age^2
>         0.00    -160.32         0.00    457.12     23.46         0.00
>    x2bmi^2    x2map^2    x2tc^2    x2ldl^2    x2hdl^2    x2tch^2
>        40.60         0.00         0.00         0.00         0.00         0.00
>    x2ltg^2    x2glu^2    x2age:sex    x2age:bmi    x2age:map    x2age:tc
>         0.00        56.15        85.50         0.00        33.05         0.00
>   x2age:ldl   x2age:hdl   x2age:tch   x2age:ltg   x2age:glu   x2sex:bmi
>         0.00         0.00         0.00         0.00        19.41         0.00
>   x2sex:map   x2sex:tc   x2sex:ldl   x2sex:hdl   x2sex:tch   x2sex:ltg
>         0.00         0.00         0.00         0.00         0.00         0.00
>   x2sex:glu   x2bmi:map   x2bmi:tc   x2bmi:ldl   x2bmi:hdl   x2bmi:tch
>         0.00        71.62         0.00         0.00         0.00         0.00
>   x2bmi:ltg   x2bmi:glu   x2map:tc   x2map:ldl   x2map:hdl   x2map:tch
>         0.00         0.00         0.00         0.00         0.00         0.00
>   x2map:ltg   x2map:glu   x2tc:ldl   x2tc:hdl   x2tc:tch   x2tc:ltg
```

```

>      0.00      0.00      0.00      0.00      0.00      0.00
> x2tc:glu x2ldl:hdl x2ldl:tch x2ldl:ltg x2ldl:glu x2hdl:tch
>      0.00      0.00      0.00      0.00      0.00      0.00
> x2hdl:ltg x2hdl:glu x2tch:ltg x2tch:glu x2ltg:glu
>      0.00      0.00      0.00      0.00      0.00
>
> Degrees of Freedom: 441 Total (i.e. Null); 431.6 Residual
> Null Deviance: 2621000
> Residual Deviance: 1248000
> AIC: 4789
> Lambda: 87.88

```

The **summary** method quickly returns the main output of the fitted model, including point estimates, standard errors and *p*-values. Visualizing estimates for all covariates could be somewhat inconvenient, especially when the number of covariates is large, thus we decide to print estimates only if the *p*value is less than a threshold value. We use *0.10*

```
summary(out2, pval = 0.1)
```

```

>
> Call:
> islasso(formula = y ~ x2, lambda = lambda.bic, data = diabetes)
>
> Residuals:
>      Min       1Q   Median       3Q      Max
> -136.951  -40.268   -4.937   36.309  144.918
>
>      Estimate Std. Error    Df z value Pr(>|z|)
> (Intercept)  152.133     2.558  1.000  59.472 < 2e-16 ***
> x2sex        -76.079    45.233  0.754  -1.682  0.092582 .
> x2bmi        494.732    67.792  1.000   7.298  2.92e-13 ***
> x2map        228.284    62.552  0.998   3.649  0.000263 ***
> x2hdl       -160.323    60.826  0.957  -2.636  0.008395 **
> x2ltg        457.117    66.243  1.000   6.901  5.18e-12 ***
> x2age:sex     85.503    45.663  0.828   1.872  0.061144 .
> x2bmi:map     71.616    42.846  0.754   1.671  0.094632 .
> ---
> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
> (Dispersion parameter for gaussian family taken to be 2892.361)
>
>      Null deviance: 2621009  on 441.0  degrees of freedom
> Residual deviance: 1248265  on 431.6  degrees of freedom
> AIC: 4789.3
> Lambda: 87.877
>
> Number of Newton-Raphson iterations: 308

```

In addition to the usual information printed by the **summary** method, the output also includes the column *Df* representing the degrees of freedom of each coefficient. Their sum is used to quantify the model complexity

```
sum(out2$internal$hi)
```

```
> [1] 10.42709
```

and the corresponding residual degrees of freedom () as reported above. The Wald test (column *z value*) and *p*-values can be used to assess important or significant covariates. Results suggest that variables bmi, map, hdl and ltg to predict the measure of diabetes progression, while sex and two interactions age:sex and bmi:map are borderline informative. Just to be clear, another way to obtain a similar result without computing the regularization path, is to use the function *aic.islasso* which requires a preliminary islasso fit object and a specification of the criterion to be used. Hence

```
lambda.bic2 <- aic.islasso(out2, method = "BIC", interval = c(1, 100))
```

```
>
> Optimization through BIC
>
> lambda = 38.8146 BIC = 4878.46285
> lambda = 62.1854 BIC = 4833.27143
> lambda = 76.6293 BIC = 4829.91482
> lambda = 71.9900 BIC = 4829.24032
> lambda = 72.4230 BIC = 4829.30009
> lambda = 68.2450 BIC = 4828.75161
> lambda = 65.9304 BIC = 4834.31100
> lambda = 69.6754 BIC = 4828.93141
> lambda = 67.3609 BIC = 4828.64601
> lambda = 66.8145 BIC = 4834.53781
> lambda = 67.6986 BIC = 4828.68585
> lambda = 67.1522 BIC = 4834.63262
> lambda = 67.4899 BIC = 4828.65682
> lambda = 67.2812 BIC = 4828.65740
> lambda = 67.3868 BIC = 4828.64808
> lambda = 67.3304 BIC = 4828.64520
> lambda = 67.3317 BIC = 4828.64464
> lambda = 67.3448 BIC = 4828.64534
> lambda = 67.3367 BIC = 4828.64488
> lambda = 67.3339 BIC = 4828.64456
> lambda = 67.3334 BIC = 4828.64471
> lambda = 67.3350 BIC = 4828.65172
> lambda = 67.3343 BIC = 4828.65172
> lambda = 67.3337 BIC = 4828.64470
> lambda = 67.3341 BIC = 4828.64455
> lambda = 67.3342 BIC = 4828.65171
> lambda = 67.3340 BIC = 4828.64456
> lambda = 67.3341 BIC = 4828.64456
> lambda = 67.3341 BIC = 4828.64455
```

```
out3 <- update(out2, lambda = lambda.bic2)
summary(out3, pval = 0.1)
```

```
>
> Call:
> islasso(formula = y ~ x2, lambda = lambda.bic2, data = diabetes)
```

```

>
> Residuals:
>      Min       1Q   Median       3Q      Max
> -138.642  -40.405   -4.437   34.847  144.269
>
>      Estimate Std. Error    Df z value Pr(>|z|)
> (Intercept)  152.133      2.538  1.000  59.935 < 2e-16 ***
> x2sex        -107.826     53.154  0.888  -2.029  0.04250 *
> x2bmi         495.530     68.682  1.000   7.215  5.40e-13 ***
> x2map         246.444     62.758  0.999   3.927  8.60e-05 ***
> x2hdl        -182.407     63.095  0.983  -2.891  0.00384 **
> x2ltg         463.173     66.163  1.000   7.001  2.55e-12 ***
> x2age:sex     104.217     49.889  0.908   2.089  0.03671 *
> x2bmi:map      82.933     47.338  0.829   1.752  0.07978 .
> ---
> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
> (Dispersion parameter for gaussian family taken to be 2847.796)
>
>      Null deviance: 2621009  on 441.0  degrees of freedom
> Residual deviance: 1227042  on 430.9  degrees of freedom
> AIC: 4783.1
> Lambda: 67.334
>
> Number of Newton-Raphson iterations: 1000

```

Comparisons between methods to select the tuning parameter and further discussions are out of the scope of this short note. We conclude this note by emphasizing that **islasso** also accepts the so-called elastic-net penalty, such that

$$\frac{1}{2} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda \{ \alpha \|\beta\|_1 + \frac{1}{2} (1 - \alpha) \|\beta\|_2^2 \}$$

where  $0 \leq \alpha \leq 1$  is the mixing parameter to be specified in *islasso()* and *islasso.path()* via the argument *alpha*, e.g.

```

# update the islasso path to fit an elastic-net model
out4 <- update(out, alpha = 0.5)
out4

```

```

>
> Call:
> islasso.path(formula = y ~ x2, nlambdas = 30L, alpha = 0.5, data = diabetes)
>
> Coefficients:
>      lambda      df      phi    deviance    logLik
> 1    0.0192 53.3378 2788.1530 1083649.6469 -2352.9731
> 2    0.0285 51.8074 2785.2072 1086767.2213 -2353.6080
> 3    0.0424 50.1275 2783.2995 1090698.4552 -2354.4060
> 4    0.0630 48.2721 2782.6748 1095616.6721 -2355.4003
> 5    0.0937 46.2059 2783.6291 1101743.9506 -2356.6328
> 6    0.1394 43.8866 2786.8431 1109479.6886 -2358.1791
> 7    0.2072 41.0769 2792.4528 1119558.9740 -2360.1778
> 8    0.3081 38.1354 2806.0099 1133248.0879 -2362.8636
> 9    0.4581 34.7083 2830.2195 1152724.9908 -2366.6296

```

```

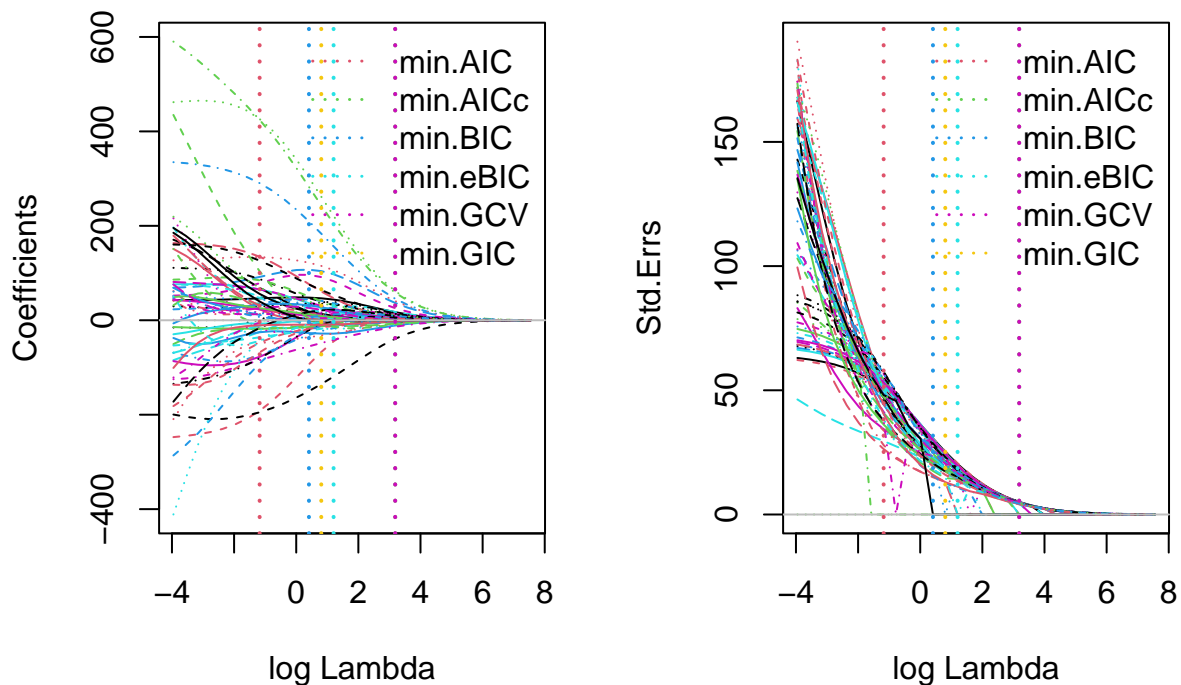
> 10  0.6811 31.3060 2874.6662 1180608.0556 -2371.9117
> 11  1.0127 27.5587 2943.2851 1219818.8439 -2379.1324
> 12  1.5057 23.4888 3042.6670 1273390.0639 -2388.6311
> 13  2.2387 19.6313 3182.1778 1344052.3655 -2400.5665
> 14  3.3286 16.1734 3368.4809 1434388.7081 -2414.9424
> 15  4.9491 13.0060 3603.1211 1545717.4272 -2431.4621
> 16  7.3585 10.2915 3884.5006 1676971.8784 -2449.4739
> 17 10.9409  7.8529 4197.9048 1822508.3980 -2467.8664
> 18 16.2674  6.0660 4525.2435 1972707.3606 -2485.3681
> 19 24.1870  4.4757 4836.5190 2116094.6451 -2500.8746
> 20 35.9623  3.3819 5114.2550 2243204.8073 -2513.7663
> 21 53.4703  2.5663 5344.8466 2348705.8621 -2523.9232
> 22 79.5018  1.9574 5525.4481 2431432.4045 -2531.5734
> 23 118.2066  1.5965 5661.8543 2493500.2273 -2537.1441
> 24 175.7546  1.3227 5760.3452 2538453.4577 -2541.0928
> 25 261.3193  1.1559 5829.6730 2569977.0286 -2543.8204
> 26 388.5404  1.0666 5877.0083 2591369.2249 -2545.6524
> 27 577.6982  1.0271 5907.8717 2605211.2844 -2546.8297
> 28 858.9459  1.0138 5926.8651 2613665.6463 -2547.5457
> 29 1277.1167  1.0051 5938.4931 2618845.1503 -2547.9833
> 30 1898.8705  1.0002 5943.1748 2620938.7814 -2548.1599

```

```

# some diagnostic plot
par(mfrow = c(1, 2))
plot(out4, yvar = "coefficients")
plot(out4, yvar = "se")

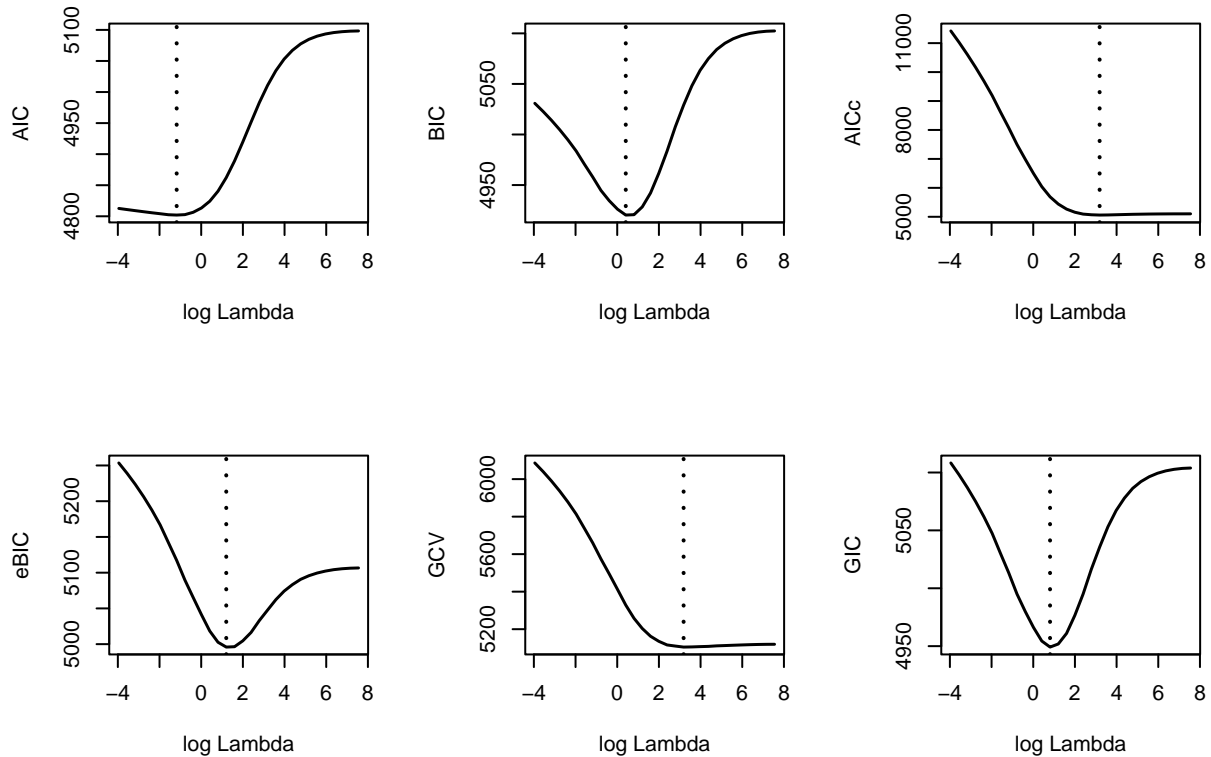
```



```

# select the best tuning parameter
lmb.best2 <- GoF.islasso.path(out4)

```



```
lmb.best2$lambda.min
```

```
>      AIC      BIC      AICc      eBIC      GCV      GIC
> 0.3080847 1.5056683 24.1870426 3.3285790 24.1870426 2.2386907
```

```
# fit a new islasso model with elastic-net penalty
lambda.bic3 <- lmb.best2$lambda.min["BIC"]
out5 <- update(out2, alpha = 0.5, lambda = lambda.bic3)
summary(out5, pval = 0.1)
```

```
>
> Call:
> islasso(formula = y ~ x2, lambda = lambda.bic3, alpha = 0.5,
> data = diabetes)
>
> Residuals:
>      Min       1Q   Median       3Q      Max
> -120.696  -40.098   -5.179   39.143  154.979
>
>      Estimate Std. Error    Df z value Pr(>|z|)
> (Intercept)  152.133      2.623  1.000  58.009 < 2e-16 ***
> x2sex        -92.139     30.576  0.524  -3.013 0.002583 **
> x2bmi         300.885     28.831  0.443  10.436 < 2e-16 ***
> x2map         209.877     30.044  0.484   6.986 2.84e-12 ***
> x2hdl        -148.335     23.292  0.365  -6.369 1.91e-10 ***
> x2tch         121.563     19.401  0.280   6.266 3.71e-10 ***
> x2ltg         281.328     27.947  0.417  10.066 < 2e-16 ***
> x2glu         106.431     30.327  0.481   3.509 0.000449 ***
```

```

> x2bmi^2      95.651      28.761  0.455   3.326 0.000882 ***
> x2glu^2      45.792      27.564  0.396   1.661 0.096655 .
> x2age:sex     75.138      29.930  0.495   2.510 0.012058 *
> x2age:ldl    -54.473      22.136  0.336  -2.461 0.013861 *
> x2age:ltg     48.020      27.292  0.395   1.760 0.078491 .
> x2sex:hdl     46.184      25.423  0.404   1.817 0.069278 .
> x2bmi:map     76.316      28.571  0.434   2.671 0.007559 **
> x2tch:glu     38.543      20.768  0.273   1.856 0.063466 .
> ---
> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
> (Dispersion parameter for gaussian family taken to be 3040.087)
>
> Null deviance: 2621009  on 441  degrees of freedom
> Residual deviance: 1270727  on 418  degrees of freedom
> AIC: 4824.4
> Lambda: 1.5057
>
> Number of Newton-Raphson iterations: 154

```

```

# or select the best tuning parameter using BIC with an islasso object
lambda.bic4 <- aic.islasso(out5, method = "BIC", interval = c(1, 100))

```

```

>
> Optimization through BIC
>
> lambda = 38.8146 BIC = 5051.72582
> lambda = 62.1854 BIC = 5068.72021
> lambda = 24.3707 BIC = 5029.72592
> lambda = 15.4439 BIC = 5005.53468
> lambda = 9.9268 BIC = 4979.49142
> lambda = 6.5171 BIC = 4956.70746
> lambda = 4.4097 BIC = 4937.73402
> lambda = 3.1073 BIC = 4929.13054
> lambda = 2.3024 BIC = 4924.94408
> lambda = 1.8049 BIC = 4923.41105
> lambda = 1.4975 BIC = 4922.61320
> lambda = 1.3075 BIC = 4923.73094
> lambda = 1.5751 BIC = 4922.40274
> lambda = 1.5950 BIC = 4923.47355
> lambda = 1.5750 BIC = 4922.40283
> lambda = 1.5827 BIC = 4922.38740
> lambda = 1.5874 BIC = 4922.37837
> lambda = 1.5903 BIC = 4922.37296
> lambda = 1.5921 BIC = 4922.36974
> lambda = 1.5932 BIC = 4922.36785
> lambda = 1.5939 BIC = 4922.36681
> lambda = 1.5943 BIC = 4922.36734
> lambda = 1.5939 BIC = 4922.36686
> lambda = 1.5940 BIC = 4922.36669
> lambda = 1.5941 BIC = 4922.36664
> lambda = 1.5942 BIC = 4922.36654
> lambda = 1.5942 BIC = 4922.36650
> lambda = 1.5943 BIC = 4922.36649

```

```

> lambda =      1.5943 BIC =  4922.36649

out6 <- update(out5, lambda = lambda.bic4)
summary(out6, pval = 0.1)

>
> Call:
> islasso(formula = y ~ x2, lambda = lambda.bic4, alpha = 0.5,
>   data = diabetes)
>
> Residuals:
>      Min       1Q   Median       3Q      Max
> -120.008  -40.854   -5.302   39.039  155.229
>
>      Estimate Std. Error    Df z value Pr(>|z|)
> (Intercept)   152.13      2.63  1.000  57.842 < 2e-16 ***
> x2sex         -88.30     29.83  0.511  -2.960 0.003074 **
> x2bmi         295.19     27.99  0.431  10.546 < 2e-16 ***
> x2map         206.12     29.23  0.472   7.052 1.76e-12 ***
> x2hdl        -146.33     22.71  0.357  -6.442 1.18e-10 ***
> x2tch         120.71     18.90  0.275   6.388 1.68e-10 ***
> x2ltg         275.65     27.13  0.406  10.160 < 2e-16 ***
> x2glu         106.26     29.49  0.468   3.603 0.000315 ***
> x2bmi^2        95.31     27.95  0.443   3.410 0.000650 ***
> x2glu^2        44.65     26.69  0.385   1.673 0.094325 .
> x2age:sex       72.88     29.15  0.482   2.500 0.012406 *
> x2age:ldl     -53.54     21.61  0.329  -2.478 0.013228 *
> x2age:ltg      46.70     26.46  0.385   1.765 0.077604 .
> x2sex:hdl      45.20     24.78  0.395   1.824 0.068123 .
> x2bmi:map      74.55     27.73  0.423   2.688 0.007181 **
> x2tch:glu      37.90     20.16  0.266   1.880 0.060064 .
> ---
> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
> (Dispersion parameter for gaussian family taken to be 3057.643)
>
>      Null deviance: 2621009  on 441.0  degrees of freedom
> Residual deviance: 1279745  on 418.5  degrees of freedom
> AIC: 4826.4
> Lambda: 1.5943
>
> Number of Newton-Raphson iterations: 974

```

## References

- Tibshirani R. *Regression shrinkage and selection via the lasso*. J R Stat Soc: Series B 1996; 58: 267–288
- Cilluffo, G, Sottile, G, La Grutta, S and Muggeo, VMR (2019) *The Induced Smoothed lasso: A practical framework for hypothesis testing in high dimensional regression*. Statistical Methods in Medical Research, online doi: 10.1177/0962280219842890.
- Brown B and Wang Y. *Standard errors and covariance matrices for smoothed rank estimators*. Biometrika 2005; 92: 149–158.