

# Package ‘SIS’

February 14, 2012

**Version** 0.6

**Date** 2010-09-06

**Title** Sure Independence Screening

**Author** Jianqing Fan, Yang Feng, Richard Samworth, Yichao Wu

**Maintainer** Yang Feng <yangfeng@stat.columbia.edu>

**Depends** survival, R (>= 2.0)

**Description** (Iterative) Sure Independence Screening for Generalized Linear Models and Cox’s Proportional Hazards Models

**License** GPL-2

**Repository** CRAN

**Date/Publication** 2010-09-06 18:12:23

## R topics documented:

COXvanISISscad . . . . .	2
getfinalSCADcoef . . . . .	4
getfinalSCADcoefCOX . . . . .	6
GLMvanISISscad . . . . .	7
heart.data . . . . .	10
lung.data . . . . .	11
scadcox . . . . .	11
scadglm . . . . .	13
SIS . . . . .	15
wlassocox . . . . .	17
wlassoglm . . . . .	19

<b>Index</b>	<b>21</b>
--------------	-----------

---

COXvanISISscad                    *(Iterative) Sure Independence Screening ((I)SIS) in the Cox proportional hazards regression model*

---

## Description

These functions implement the iterative sure independence screening with `COXvanISISscad` for vanilla ISIS and `COXvarISISscad` for variant ISIS in Cox proportional hazards regression model.

## Usage

```
COXvanISISscad(x, time, status, method = "efron", nsis=NULL, folds=NULL,
rank.method="obj", eps0=1e-5, inittype='NoPen', tune.method="AIC",
ISISStypeCumulative=FALSE, DOISIS=TRUE, maxloop=5)
```

```
COXvarISISscad(x, time, status, method = "efron", nsis=NULL, folds=NULL,
rank.method="obj", eps0=1e-5, inittype='NoPen', tune.method="AIC",
vartype="First", ISISStypeCumulative=FALSE, DOISIS=TRUE, maxloop=5)
```

## Arguments

<code>x</code>	an (n * p) matrix of features.
<code>time</code>	an (n) vector of the follow up time for right censored data.
<code>status</code>	an (n) vector of the status indicator, normally 0=alive, 1=dead.
<code>method</code>	indicates how to handle observations that have tied (i.e., identical) survival times. The default "efron" method is generally preferred to the once-popular "breslow" method.
<code>nsis</code>	number of predictors recruited by (I)SIS.
<code>folds</code>	fold information for cross validation.
<code>rank.method</code>	the criterion for ranking predictor variables in (I)SIS. It can be either obj or coeff.
<code>tune.method</code>	method for tuning regularization parameter.
<code>inittype</code>	inittype specifies the type of initial solution for the one-step SCAD. It can be either NoPen or L1.
<code>vartype</code>	vartype specifies variant (I)SIS of first type or second type.
<code>ISISStypeCumulative</code>	ISISStypeCumulative specifies whether to penalize variables selected by the previous step of the ISIS iteration in the following SCAD step. (ISISStypeCumulative=FALSE put penalties on all variables. In this case, the procedure is more likely to delete variables from previous step. )
<code>DOISIS</code>	DOISIS specifies whether to do iterative SIS.
<code>maxloop</code>	maximum number of loops in iterative SIS.
<code>eps0</code>	an effective zero relative to the scale of the maximum absolute marginal regression coefficients.

**Value**

Returns an object with

`initRANKorder` initial predictor ranking order for vanilla SIS.  
`detail.pickind`, `detail.ISISind`  
 details of each loop of ISIS.  
`SISind` the vector of indices selected by SIS.  
`ISISind` the vector of indices selected by ISIS.  
`normal.exit` indicator of normal exit.  
`nsis` number of predictors recruited by (I)SIS.  
`initRANKorder1`, `initRANKorder2`  
 initial predictor ranking order for variant SIS.

**Author(s)**

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

**References**

Jianqing Fan and Jinchi Lv (2008) Sure independence screening for ultra-high dimensional feature space (with discussion) *Journal of Royal Statistical Society B*, **36**, 849-911.  
 Jianqing Fan, Richard Samworth, and Yichao Wu (2009) Ultrahigh dimensional variable selection: beyond the linear model *Journal of Machine Learning Research*, to appear.  
 Jianqing Fan and Rui Song (2009) Sure Independence Screening in Generalized Linear Models with NP-Dimensionality, technical report.

**See Also**

[scadcox](#), [fullscadcox](#)

**Examples**

```
set.seed(0)
n=150
p=200
truerho=0.5
beta <- c(4,4,4,-6*sqrt(2),4/3, rep(0,p-5))

corrmat=diag(rep(1-truerho, p))+matrix(truerho, p, p)
corrmat[,4]=sqrt(truerho)
corrmat[4, ]=sqrt(truerho)
corrmat[4,4]=1
corrmat[,5]=0
corrmat[5,]=0
corrmat[5,5]=1
cholmat=chol(corrmat)

x=matrix(rnorm(p*n, mean=0, sd=1), n, p)
```

```

x=x%%cholmat

myrates <- exp(x%%beta)

ytrue <- rexp(n, rate = myrates)
cen <- rexp(n, rate = 0.1 )
time <- pmin(ytrue, cen)
status <- as.numeric(ytrue <= cen)

cox.van.sis <- COXvanISISscad(x, time, status)

cox.var.sis <- COXvarISISscad(x, time, status, vartype='Second')

####compare the result
cox.van.sis$SIS

cox.van.sis$ISIS

cox.var.sis$SIS

cox.var.sis$ISIS

```

---

getfinalSCADcoef

*SCAD regularized loglikelihood for generalized linear models*


---

## Description

This function gets the final regression coefficients for the SCAD regularized loglikelihood for generalized linear models after applying (I)SIS.

## Usage

```
getfinalSCADcoef(x, y, pickind, folds=NULL, eps0=1e-5, family=binomial(),
tune.method="AIC", inittype='NoPen', detailed=FALSE)
```

```
INDEPgetfinalSCADcoef(x, y, pickind, folds=NULL, xtune, ytune,
family=binomial(), inittype='NoPen', eps0=1e-5, detailed=FALSE)
```

## Arguments

x	an (n * p) matrix of features.
y	an (n) vector of response.
pickind	predictor indice selected by (I)SIS.
folds	fold information for cross validation.
family	a description of the error distribution and link function to be used in the model.

tune.method	method for tuning regularization parameter.
inittype	inittype specifies the type of initial solution for the one-step SCAD. It can be either NoPen or L1.
xtune, ytune	independent tuning dataset.
eps0	an effective zero.
detailed	indicates whether detailed information will be provided. Default is FALSE.

### Value

An initial solution vector `wt.initsoln` and final solution (p+1) vector `SCADcoef`.

### Author(s)

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

### References

Jianqing Fan and Runze Li (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of American Statistical Association*, **96**, 1348-1360.

Hui Zou and Runze Li (2008) One-step Sparse Estimates in Nonconcave Penalized Likelihood Models (with discussion). *The Annals of Statistics*, **36**, 1509-1533

### See Also

[scadglm](#), [fullscadglm](#)

### Examples

```
set.seed(0)
b <- c(4,4,4,-6*sqrt(2))
n=150
p=200
truerho=0.5
corrmat=diag(rep(1-truerho, p))+matrix(truerho, p, p)
corrmat[,4]=sqrt(truerho)
corrmat[4, ]=sqrt(truerho)
corrmat[4,4]=1
cholmat=chol(corrmat)
x=matrix(rnorm(n*p, mean=0, sd=1), n, p)
x=x%%cholmat
feta=x[, 1:4]%%b
fprob=exp(feta)/(1+exp(feta))
y=rbinom(n, 1, fprob)

nsis=floor(n/log(n)/4)
xtune=matrix(rnorm(n*p, mean=0, sd=1), n, p)
xtune=xtune%%cholmat
feta=xtune[, 1:4]%%b
fprob=exp(feta)/(1+exp(feta))
ytune=rbinom(n, 1, fprob)
```

```
ISIScoef = INDEPgetfinalSCADcoef(x = x, y = y,
                                pickind = 1:4, xtune = xtune, ytune = ytune,
                                family = binomial())
```

---

getfinalSCADcoefCOX     *SCAD regularized loglikelihood for Cox proportional hazards regression models*

---

### Description

This function gets the final regression coefficients for the SCAD regularized loglikelihood for Cox proportional hazards regression models after applying (I)SIS

### Usage

```
getfinalSCADcoefCOX(x, time, status, method = "efron", pickind,
                    folds = NULL, eps0 = 1e-5, tune.method = "AIC", inittype = "NoPen",
                    detailed = FALSE)
```

### Arguments

x	an (n * p) matrix of features.
time	an (n) vector of the follow up time for right censored data.
status	an (n) vector of the status indicator, normally 0=alive, 1=dead.
method	indicates how to handle observations that have tied (i.e., identical) survival times. The default "efron" method is generally preferred to the once-popular "breslow" method.
pickind	predictor indice selected by (I)SIS.
folds	fold information for cross validation.
eps0	an effective zero.
tune.method	method for tuning regularization parameter.
inittype	inittype specifies the type of initial solution for the one-step SCAD. It can be either NoPen or L1.
detailed	indicates whether detailed information will be provided. Default is FALSE.

### Value

An initial solution vector `wt.initsoln` and final solution (p) vector `SCADcoef`.

### Author(s)

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

## References

Jianqing Fan and Runze Li (2002) Variable Selection for Cox's Proportional Hazards Model and Frailty Model. *The Annals of Statistics*, **30**, 74-99.

Hui Zou and Runze Li (2008) One-step Sparse Estimates in Nonconcave Penalized Likelihood Models (with discussion). *The Annals of Statistics*, **36**, 1509-1533

## See Also

[scadcox](#), [fullscadcox](#)

## Examples

```
set.seed(0)
n=150
p=200
truerho=0.5
beta <- c(4,4,4,-6*sqrt(2),4/3, rep(0,p-5))

corrmat=diag(rep(1-truerho, p))+matrix(truerho, p, p)
corrmat[,4]=sqrt(truerho)
corrmat[4, ]=sqrt(truerho)
corrmat[4,4]=1
corrmat[,5]=0
corrmat[5,]=0
corrmat[5,5]=1
cholmat=chol(corrmat)

x=matrix(rnorm(p*n, mean=0, sd=1), n, p)
x=x%%cholmat

myrates <- exp(x%%beta)

ytrue <- rexp(n, rate = myrates)
cen <- rexp(n, rate = 0.1 )
time <- pmin(ytrue, cen)
status <- as.numeric(ytrue <= cen)

SIScoef <- getfinalSCADcoefCOX(x = x, time = time, status = status,
                               pickind = 1:5)
```

---

GLMvanISISscad

*(Iterative) Sure Independence Screening ((I)SIS) in Generalized Linear Models*

---

## Description

These functions implement the iterative sure independence screening with [GLMvanISISscad](#) for vanilla ISIS and [GLMvarISISscad](#) for variant ISIS in Generalized Linear Models.

**Usage**

```
GLMvanISISscad(x, y, nsis=NULL, family=binomial(), folds=folds,
rank.method="obj", eps0=1e-5, inittype='NoPen', tune.method="AIC",
ISISStypeCumulative=FALSE, DOISIS=TRUE, maxloop=5)
```

```
GLMvarISISscad(x, y, nsis=NULL, family=binomial(), folds=folds,
rank.method="obj", tune.method="AIC", vartype="First", eps0=1e-5,
inittype='NoPen', ISISStypeCumulative=FALSE, DOISIS=TRUE, maxloop=5)
```

**Arguments**

<code>x</code>	an (n * p) matrix of features.
<code>y</code>	an (n) vector of response.
<code>nsis</code>	number of predictors recruited by (I)SIS.
<code>family</code>	a description of the error distribution and link function to be used in the model.
<code>folds</code>	fold information for cross validation.
<code>rank.method</code>	the criterion for ranking predictor variables in (I)SIS. It can be either <code>obj</code> or <code>coeff</code> .
<code>tune.method</code>	method for tuning regularization parameter.
<code>inittype</code>	<code>inittype</code> specifies the type of initial solution for the one-step SCAD. It can be either <code>NoPen</code> or <code>L1</code> .
<code>vartype</code>	<code>vartype</code> specifies variant (I)SIS of first type or second type.
<code>ISISStypeCumulative</code>	<code>ISISStypeCumulative</code> specifies whether to penalize variables selected by the previous step of the ISIS iteration in the following SCAD step. ( <code>ISISStypeCumulative=FALSE</code> put penalties on all variables. In this case, the procedure is more likely to delete variables from previous step. )
<code>DOISIS</code>	<code>DOISIS</code> specifies whether to do iterative SIS.
<code>maxloop</code>	maximum number of loops in iterative SIS.
<code>eps0</code>	an effective zero relative to the scale of the maximum absolute marginal regression coefficients.

**Value**

Returns an object with

<code>initRANKorder</code>	initial predictor ranking order for vanilla SIS.
<code>detail.pickind</code> , <code>detail.ISISind</code>	details of each loop of ISIS.
<code>normal.exit</code>	indicator of normal exit.
<code>SISind</code>	the vector of indices selected by SIS.
<code>ISISind</code>	the vector of indices selected by ISIS.
<code>initRANKorder1</code> , <code>initRANKorder2</code>	initial predictor ranking order for variant SIS.

**Author(s)**

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

**References**

Jianqing Fan and Jinchi Lv (2008) Sure independence screening for ultra-high dimensional feature space (with discussion) *Journal of Royal Statistical Society B*, **36**, 849-911.

Jianqing Fan, Richard Samworth, and Yichao Wu (2009) Ultrahigh dimensional variable selection: beyond the linear model *Journal of Machine Learning Research*, to appear.

Jianqing Fan and Rui Song (2009) Sure Independence Screening in Generalized Linear Models with NP-Dimensionality, technical report.

**See Also**

[scadglm](#), [fullscadglm](#)

**Examples**

```

set.seed(0)
b <- c(4,4,4,-6*sqrt(2))
n=150
p=200
truerho=0.5
corrmat=diag(rep(1-truerho, p))+matrix(truerho, p, p)
corrmat[,4]=sqrt(truerho)
corrmat[4, ]=sqrt(truerho)
corrmat[4,4]=1
cholmat=chol(corrmat)
x=matrix(rnorm(n*p, mean=0, sd=1), n, p)
x=x%%cholmat
feta=x[, 1:4]%%b
fprob=exp(feta)/(1+exp(feta))
y=rbinom(n, 1, fprob)

nsis=floor(n/log(n)/4)

binom.van.sis=GLMvanISISscad(x, y, nsis, family=binomial(), tune.method='BIC')
binom.var.sis=GLMvarISISscad(x, y, nsis, family=binomial(), vartype='Second',
tune.method='BIC')

#####compare the result
binom.van.sis$SIS

binom.van.sis$ISIS

binom.var.sis$SIS

binom.var.sis$ISIS

```

---

heart.data	<i>Dataset for SIS</i>
------------	------------------------

---

## Description

*South African Heart Disease dataset* used to test SIS algorithm

## Usage

```
data(heart.data)
```

## Format

A dataset with 462 observations on 9 variables and a binary response.

**x** **x** contains 9 columns of the following variables: sbp (systolic blood pressure); tobacco (cumulative tobacco); ldl (low density lipoprotein cholesterol); adiposity; famhist (family history of heart disease); typea (type-A behavior); obesity; alcohol (current alcohol consumption); age (age at onset)

**y** response, coronary heart disease

## References

Hastie, T., Tibshirani, R., and Friedman, J. (2001) *Elements of Statistical Learning; Data Mining, Inference, and Prediction* Springer-Verlag, New York.

## Examples

```
data(heart.data)
attach(heart.data)
n=length(y)
p=dim(x)[2]
set.seed(0)
trainset=sample(1:n,floor(n*2/3))
testset=setdiff(1:n,trainset)

traindata=list(x=x[trainset,],y=y[trainset])

fit <- SIS(traindata, family=binomial(), xtune=x[testset,], ytune=y[testset])

detach(heart.data)
```

---

`lung.data`*Dataset for SIS*

---

**Description**

*Lung cancer dataset used to test SIS algorithm*

**Usage**

```
data(lung.data)
```

**Format**

A dataset consisting of 137 observations with their survival time, censor status as well as 6 features.

**x** `x` contains 6 columns of the following variables: `trt` (1=standard treatment, and 2=test); `celltype` (1=squamous, 2=smallcell, 3=adeno, and 4=large); `karno` (Karnofsky performance score); `diagtime` (months from diagnosis to randomization); `age` (in years); `prior` (prior therapy 0=no, and 1=yes)

**time** survival time

**status** censor status

**References**

Kalbfleisch, J. and Prentice, R. (2002) *The Statistical Analysis of Failure Time Data* J. Wiley, Hoboken, N.J.

**Examples**

```
data(lung.data)
attach(lung.data)
fit <- SIS(lung.data, model='cox')
detach(lung.data)
```

---

`scadcox`*SCAD regularized loglikelihood for Cox proportional hazards regression models*

---

**Description**

These functions solve SCAD regularized loglikelihood for Cox proportional hazards regression models; `scadcox` does the one-step SCAD while `fullscadcox` solves the SCAD in a fully iterative method.

**Usage**

```
scadcox(x, time, status, method="efron", wt.initsoln=NULL, lambda,
        initsoln=NULL, weight = NULL, function.precision=1e-10,
        nopenalty.subset=NULL)
```

```
fullscadcox(x, time, status, method="efron", lambda,
            initsoln=NULL, weight = NULL, function.precision=1e-10,
            nopenalty.subset=NULL, eps0=1e-5, maxloop=10)
```

**Arguments**

x	an (n * p) matrix of features.
time	an (n) vector of the follow up time for right censored data.
status	an (n) vector of the status indicator, normally 0=alive, 1=dead.
method	indicates how to handle observations that have tied (i.e., identical) survival times. The default "efron" method is generally preferred to the once-popular "breslow" method.
wt.initsoln	a (p) vector of initial solution for one-step SCAD.
lambda	the regularization parameter.
initsoln	a (p) vector of initial solution.
weight	an optional (n) vector of weights to be used in the fitting process.
function.precision	function.precision parameter used in the internal solver. Default is 1e-10.
nopenalty.subset	a set of indices for the predictors that are not subject to the L1 penalty.
eps0	an effective zero.
maxloop	the maximum number of loops for the SCAD iteration.

**Value**

They return a (p) vector of estimated coefficients.

**Author(s)**

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

**References**

Jianqing Fan and Runze Li (2002) Variable Selection for Cox's Proportional Hazards Model and Frailty Model. *The Annals of Statistics*, **30**, 74-99.

Hui Zou and Runze Li (2008) One-step Sparse Estimates in Nonconcave Penalized Likelihood Models (with discussion). *The Annals of Statistics*, **36**, 1509-1533

**See Also**

[wtlassocox](#)

**Examples**

```

set.seed(0)
n=400
p=1000
truerho=0.5
beta <- c(4,4,4,-6*sqrt(2),4/3, rep(0,p-5))

corrmat=diag(rep(1-truerho, p))+matrix(truerho, p, p)
corrmat[,4]=sqrt(truerho)
corrmat[4, ]=sqrt(truerho)
corrmat[4,4]=1
corrmat[,5]=0
corrmat[5,]=0
corrmat[5,5]=1
cholmat=chol(corrmat)

x=matrix(rnorm(p*n, mean=0, sd=1), n, p)
x=x%%cholmat

myrates <- exp(x%%beta)

ytrue <- rexp(n, rate = myrates)
cen <- rexp(n, rate = 0.1 )
time <- pmin(ytrue, cen)
status <- as.numeric(ytrue <= cen)

w1 <- scadcox(x[,1:100],time,status,lambda=0.003)
w2 <- coxph(Surv(time,status)~x[,1:100])$coef

```

---

scadglm

*SCAD regularized loglikelihood for generalized linear models*


---

**Description**

These functions solve SCAD regularized loglikelihood for generalized linear models; scadcox does the one-step SCAD while fullscadcox solves the SCAD in a fully iterative method.

**Usage**

```

scadglm(x, y, wt.initsoln=NULL, lambda, initsoln=NULL,
family = binomial(), weight = NULL, offset = NULL,
function.precision=1e-10, nopenalty.subset=NULL)

fullscadglm(x, y, lambda, initsoln=NULL, family = binomial(),
weight = NULL, offset = NULL, function.precision=1e-10,
nopenalty.subset=NULL, eps0=1e-5, maxloop=10)

```

**Arguments**

<code>x</code>	an (n * p) matrix of features.
<code>y</code>	an (n) vector of response.
<code>wt.initsoln</code>	a (p+1) vector of initial solution for one-step SCAD.
<code>lambda</code>	regularization parameter for the SCAD.
<code>initsoln</code>	a (p+1) vector of initial solution.
<code>family</code>	a description of the error distribution and link function to be used in the model.
<code>weight</code>	an optional (n) vector of weights to be used in the fitting process.
<code>offset</code>	this can be used to specify an a priori known component to be included in the linear predictor during fitting.
<code>function.precision</code>	function.precision parameter used in the internal solver. Default is 1e-10.
<code>nopenalty.subset</code>	a set of indices for the predictors that are not subject to the L1 penalty.
<code>eps0</code>	an effective zero.
<code>maxloop</code>	the maximum number of loops for the SCAD iteration.

**Value**

They return a (p+1) vector of estimated coefficients.

**Author(s)**

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

**References**

Jianqing Fan and Runze Li (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of American Statistical Association*, **96**, 1348-1360.

Hui Zou and Runze Li (2008) One-step Sparse Estimates in Nonconcave Penalized Likelihood Models (with discussion). *The Annals of Statistics*, **36**, 1509-1533

**See Also**

[wtlassoglm](#)

**Examples**

```
set.seed(0)
b <- c(1,1,1,-3*sqrt(2)/2)
n=400
p=30
truerho=0.5
x=matrix(rnorm(n*p, mean=0, sd=1), n, p)
feta=x[, 1:4]%*%b
fprob=exp(feta)/(1+exp(feta))
```

```

y=rbinom(n, 1, fprob)
scadglm(x,y,lambda=0.0015)
coef(glm(y~x,family=binomial()))

```

---

SIS *(Iterative) Sure Independence Screening ((I)SIS) and fitting in Generalized Linear Models and Cox proportional hazards regression models*

---

## Description

This function first implements the iterative sure independence screening with functions [GLMvanISISscad](#), [GLMvanISISscad](#), [COXvanISISscad](#), [COXvarISISscad](#) for different variants of (I)SIS, then gets the final regression coefficients with functions [getfinalSCADcoef](#), [INDEPgetfinalSCADcoef](#), [getfinalSCADcoefCOX](#) for the SCAD regularized loglikelihood for the variables picked by (I)SIS.

## Usage

```

SIS(data=NULL, model='glm', family=NULL, method='efron', vartype=0, nsis=NULL,
rank.method='obj', eps0=1e-5, inittype='NoPen', tune.method='BIC', folds=NULL,
post.tune.method='CV', post.tune.folds=NULL, DOISIS=TRUE,
ISIStypeCumulative=FALSE, maxloop=5, xtune=NULL, ytune=NULL, detail=FALSE)

```

## Arguments

<code>data</code>	a list that contains the data.
<code>model</code>	the model used, the implemented ones are 'glm' and 'cox'.
<code>family</code>	a description of the error distribution and link function to be used in the generalized linear model.
<code>method</code>	indicates how to handle observations that have tied (i.e., identical) survival times. The default "efron" method is generally preferred to the once-popular "breslow" method.
<code>vartype</code>	vartype specifies variant (I)SIS of first type or second type.
<code>nsis</code>	number of predictors recruited by (I)SIS.
<code>rank.method</code>	the criterion for ranking predictor variables in (I)SIS. It can be either obj or coeff.
<code>eps0</code>	an effective zero relative to the scale of the maximum absolute marginal regression coefficients.
<code>inittype</code>	inittype specifies the type of initial solution for the one-step SCAD. It can be either NoPen or L1.
<code>tune.method</code>	method for tuning regularization parameter.
<code>folds</code>	fold information for cross validation.
<code>post.tune.method</code>	method for tuning regularization parameter in the final step for getting SCAD coefficients.

<code>post.tune.folds</code>	fold information for cross validation in the final step for getting SCAD coefficients.
<code>DOISIS</code>	<code>DOISIS</code> specifies whether to do iterative SIS.
<code>ISISTypeCumulative</code>	<code>ISISTypeCumulative</code> specifies whether to penalize variables selected by the previous step of the ISIS iteration in the following SCAD step. ( <code>ISISTypeCumulative=FALSE</code> put penalties on all variables. In this case, the procedure is more likely to delete variables from previous step. )
<code>maxloop</code>	maximum number of loops in iterative SIS.
<code>xtune, ytune</code>	independent tuning dataset.
<code>detail</code>	indicates whether return detailed information or not. Default is <code>FALSE</code> .

**Value**

Returns an object with

<code>SISind</code>	the vector of indices selected by SIS.
<code>ISISind</code>	the vector of indices selected by ISIS.
<code>SIScoef</code>	the vector of solution by SIS followed by SCAD.
<code>ISIScoef</code>	the vector of solution by ISIS.

**Author(s)**

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

**References**

Jianqing Fan and Jinchi Lv (2008) Sure independence screening for ultra-high dimensional feature space (with discussion) *Journal of Royal Statistical Society B*, **36**, 849-911.

Jianqing Fan, Richard Samworth, and Yichao Wu (2009) Ultrahigh dimensional variable selection: beyond the linear model *Journal of Machine Learning Research*, to appear.

Jianqing Fan and Rui Song (2009) Sure Independence Screening in Generalized Linear Models with NP-Dimensionality, technical report.

**See Also**

[GLMvanISISscad](#), [GLMvanISISscad](#), [COXvanISISscad](#), [COXvarISISscad](#), [getfinalSCADcoef](#), [INDEPgetfinalSCADcoef](#), [getfinalSCADcoefCOX](#)

**Examples**

```
set.seed(0)
b <- c(2,2,2,-3*sqrt(2))
n=150
p=200
truerho=0.5
corrmat=diag(rep(1-truerho, p))+matrix(truerho, p, p)
```

```

corrmat[,4]=sqrt(truerho)
corrmat[4, ]=sqrt(truerho)
corrmat[4,4]=1
cholmat=chol(corrmat)
x=matrix(rnorm(n*p, mean=0, sd=1), n, p)
x=x%%cholmat
feta=x[, 1:4]%%b
fprob=exp(feta)/(1+exp(feta))
y=rbinom(n, 1, fprob)

xtune=matrix(rnorm(n*p, mean=0, sd=1), n, p)
xtune=xtune%%cholmat
feta=xtune[, 1:4]%%b
fprob=exp(feta)/(1+exp(feta))
ytune=rbinom(n, 1, fprob)

binom.result1=SIS(data=list(x=x, y=y), family=binomial(), xtune=xtune, ytune=ytune)
binom.result2=SIS(data=list(x=x, y=y), family=binomial(), xtune=xtune, ytune=ytune,
vartype=1)
binom.result1$ISISind
binom.result2$ISISind

myrates <- exp(x[,1:4]%%b)

ytrue <- rexp(n, rate = myrates)
cen <- rexp(n, rate = 0.1 )
time <- pmin(ytrue, cen)
status <- as.numeric(ytrue <= cen)

cox.result1=SIS(data=list(x=x,time=time,status=status), model='cox', vartype=0)
cox.result2=SIS(data=list(x=x,time=time,status=status), model='cox', vartype=1)
cox.result3=SIS(data=list(x=x,time=time,status=status), model='cox', vartype=2)

```

---

wtlassocox

*Weighted L1 regularized loglikelihood for Cox proportional hazards regression model*


---

## Description

This functions solves weighted L1 regularized loglikelihood for Cox proportional hazards regression model.

## Usage

```
wtlassocox(x, time, status, method="efron", lassoweight=NULL, initsoln=NULL,
weight = NULL, lambda2=0, function.precision=1e-10)
```

**Arguments**

<code>x</code>	an (n * p) matrix of features.
<code>time</code>	an (n) vector of the follow up time for right censored data.
<code>status</code>	an (n) vector of the status indicator, normally 0=alive, 1=dead.
<code>method</code>	indicates how to handle observations that have tied (i.e., identical) survival times. The default "efron" method is generally preferred to the once-popular "breslow" method.
<code>lassoweight</code>	an optional (p) vector of weights specifying the weighted L1 penalty.
<code>initsoIn</code>	an optional (p) vector of initial solution.
<code>weight</code>	an optional (n) vector of weights to be used in the fitting process.
<code>lambda2</code>	regularization parameter for the L2 norm of the coefficients. Default is 0.
<code>function.precision</code>	<code>function.precision</code> parameter used in the internal solver. Default is 1e-10.

**Details**

This function solves weighted L1 regularized loglikelihood for Cox proportional hazards regression model. It is based on the source code of R package `glm`path.

**Value**

An object is returned with

<code>lambda2</code>	$\lambda_2$ used.
<code>xnames</code>	column names of <code>x</code> .
<code>weight</code>	an optional (n) vector of weights to be used in the fitting process.
<code>lassoweight</code>	a (p) vector of weights specifying the weighted L1 penalty.
<code>initsoIn</code>	a (p) vector of initial solution.
<code>w</code>	a (p) vector of weight L1 solution.

**Author(s)**

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

**See Also**

[scadglm](#), [fullscadglm](#)

**Examples**

```
set.seed(0)
n=400
p=1000
truerho=0.5
beta <- c(4,4,4,-6*sqrt(2),4/3, rep(0,p-5))
```

```

corrmat=diag(rep(1-truerho, p))+matrix(truerho, p, p)
corrmat[,4]=sqrt(truerho)
corrmat[4, ]=sqrt(truerho)
corrmat[4,4]=1
corrmat[,5]=0
corrmat[5,]=0
corrmat[5,5]=1
cholmat=chol(corrmat)

x=matrix(rnorm(p*n, mean=0, sd=1), n, p)
x=x%%cholmat

myrates <- exp(x%%beta)

ytrue <- rexp(n, rate = myrates)
cen <- rexp(n, rate = 0.1 )
time <- pmin(ytrue, cen)
status <- as.numeric(ytrue <= cen)

weights <- rep(0.01*n, 100)
w1 <- wtlassocox(x[,1:100],time,status,lassoweight=weights)$w
w2 <- coxph(Surv(time,status)~x[,1:100])$coef

```

wtlassoglm

*Weighted L1 regularized loglikelihood for generalized linear models***Description**

This functions solves weighted L1 regularized loglikelihood for generalized linear models.

**Usage**

```
wtlassoglm(x, y, lassoweight=NULL, initsoln=NULL, family = binomial(),
weight = NULL, offset = NULL, lambda2=0, function.precision=1e-10)
```

**Arguments**

x	an (n * p) matrix of features.
y	an (n) vector of response.
lassoweight	a (p) vector of weights specifying the weighted L1 penalty.
initsoln	a (p+1) vector of initial solution.
family	a description of the error distribution and link function to be used in the model.
weight	an optional (n) vector of weights to be used in the fitting process.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting.

lambda2 regularization parameter for the L2 norm of the coefficients. Default is 0.  
 function.precision function.precision parameter used in the internal solver. Default is 1e-10.

### Details

This function solves weighted L1 regularized loglikelihood for generalized linear models. It is based on the source code of R package glmpath.

### Value

An object is returned with

lambda2	$\lambda_2$ used.
xnames	column names of $x$ .
family	a description of the error distribution and link function to be used in the model.
weight	an optional (n) vector of weights to be used in the fitting process.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting.
lassoweight	a (p) vector of weights specifying the weighted L1 penalty.
initsoIn	a (p+1) vector of initial solution.
w	a (p+1) vector of weight L1 solution.

### Author(s)

Jianqing Fan, Yang Feng, Richard Samworth, and Yichao Wu

### See Also

[scadglm](#), [fullscadglm](#)

### Examples

```
set.seed(0)
b <- c(2,2,2,-3*sqrt(2))
n=400
p=30
truerho=0.5
x=matrix(rnorm(n*p, mean=0, sd=1), n, p)
feta=x[, 1:4]*%*%b
fprob=exp(feta)/(1+exp(feta))
y=rbinom(n, 1, fprob)
lassoweight<-rep(0.6,30)
wtlassoglm(x,y,lassoweight)$w
coef(glm(y~x,family=binomial()))
```

# Index

## \*Topic **datasets**

heart.data, 10

lung.data, 11

## \*Topic **file**

COXvanISISscad, 2

getfinalSCADcoef, 4

getfinalSCADcoefCOX, 6

GLMvanISISscad, 7

scadcox, 11

scadglm, 13

SIS, 15

wlassocox, 17

wlassoglm, 19

COXvanISISscad, 2, 2, 15, 16

COXvarISISscad, 2, 15, 16

COXvarISISscad (COXvanISISscad), 2

fullscadcox, 3, 7

fullscadcox (scadcox), 11

fullscadglm, 5, 9, 18, 20

fullscadglm (scadglm), 13

getfinalSCADcoef, 4, 15, 16

getfinalSCADcoefCOX, 6, 15, 16

GLMvanISISscad, 7, 7, 15, 16

GLMvarISISscad, 7

GLMvarISISscad (GLMvanISISscad), 7

heart.data, 10

INDEPgetfinalSCADcoef, 15, 16

INDEPgetfinalSCADcoef  
(getfinalSCADcoef), 4

lung.data, 11

scadcox, 3, 7, 11

scadglm, 5, 9, 13, 18, 20

SIS, 15

wlassocox, 12, 17

wlassoglm, 14, 19