# Package 'xrnet'

July 21, 2025

**Type** Package

**Title** Hierarchical Regularized Regression

**Version** 1.0.0

**URL** https://github.com/USCbiostats/xrnet,
https://uscbiostats.github.io/xrnet/

**Description** Fits hierarchical regularized regression models
to incorporate potentially informative exter-
nal data, Weaver and Lewinger (2019) <doi:10.21105/joss.01761>.
Utilizes coordinate descent to efficiently fit regularized regression models both with and without external information with the most common penalties used in practice (i.e. ridge, lasso, elastic net).
Support for standard R matrices, sparse matrices and big.matrix objects.

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.1

**Suggests** knitr, rmarkdown, testthat, Matrix, doParallel

**LinkingTo** Rcpp, RcppEigen, BH, bigmemory

**Imports** Rcpp (>= 0.12.19), foreach, bigmemory, methods

**Depends** R (>= 4.0)

**BugReports** https://github.com/USCbiostats/xrnet/issues

**NeedsCompilation** yes

**Author** Garrett Weaver [aut, cre] (ORCID:
<https://orcid.org/0000-0002-9918-8386>),
Dixin Shen [aut],
Juan Pablo Lewinger [ctb, ths]

**Maintainer** Garrett Weaver <gmweaver.usc@gmail.com>

**Repository** CRAN

**Date/Publication** 2024-07-16 00:40:02 UTC

# Contents



---

`coef.tune_xrnet` *Get coefficient estimates from "tune_xrnet" model object.*

---

## Description

Returns coefficients from 'xrnet' model. Note that we currently only support returning coefficient estimates that are in the original path(s).

## Usage

```
## S3 method for class 'tune_xrnet'
coef(object, p = "opt", pext = "opt", ...)
```

## Arguments

| | |
|---|---|
| `object` | A `tune_xrnet` object. |
| `p` | vector of penalty values to apply to predictor variables. Default is optimal value in tune_xrnet object. |
| `pext` | vector of penalty values to apply to external data variables. Default is optimal value in tune_xrnet object. |
| `...` | pass other arguments to xrnet function (if needed). |

## Value

A list with coefficient estimates at each of the requested penalty combinations.

| | |
|---|---|
| beta0 | matrix of first-level intercepts indexed by penalty values, NULL if no first-level intercept in original model fit. |
| betas | 3-dimensional array of first-level penalized coefficients indexed by penalty values. |
| gammas | 3-dimensional array of first-level non-penalized coefficients indexed by penalty values, NULL if unpen NULL in original model fit. |
| alpha0 | matrix of second-level intercepts indexed by penalty values, NULL if no second-level intercept in original model fit. |
| alphas | 3-dimensional array of second-level external data coefficients indexed by penalty values, NULL if external NULL in original model fit. |

## Examples

```
## Cross validation of hierarchical linear regression model
data(GaussianExample)

## 5-fold cross validation
cv_xrnet <- tune_xrnet(
  x = x_linear,
  y = y_linear,
  external = ext_linear,
  family = "gaussian",
  control = xrnet_control(tolerance = 1e-6)
)

## Get coefficient estimates at optimal penalty combination
coef_opt <- coef(cv_xrnet)
```

---

# coef.xrnet *Get coefficient estimates from "xrnet" model object.*

---

## Description

Returns coefficients from 'xrnet' model. Note that we currently only support returning coefficient estimates that are in the original path(s).

## Usage

```
## S3 method for class 'xrnet'
coef(object, p = NULL, pext = NULL, ...)
```

## Arguments

| | |
|---|---|
| object | A xrnet object. |
| p | vector of penalty values to apply to predictor variables. |
| pext | vector of penalty values to apply to external data variables. |
| ... | pass other arguments to xrnet function (if needed). |

## Value

A list with coefficient estimates at each of the requested penalty combinations.

| | |
|---|---|
| beta0 | matrix of first-level intercepts indexed by penalty values, NULL if no first-level intercept in original model fit. |
| betas | 3-dimensional array of first-level penalized coefficients indexed by penalty values. |
| gammas | 3-dimensional array of first-level non-penalized coefficients indexed by penalty values, NULL if unpen NULL in original model fit. |
| alpha0 | matrix of second-level intercepts indexed by penalty values, NULL if no second-level intercept in original model fit. |
| alphas | 3-dimensional array of second-level external data coefficients indexed by penalty values, NULL if external NULL in original model fit. |

## Examples

```
data(GaussianExample)

fit_xrnet <- xrnet(
  x = x_linear,
  y = y_linear,
  external = ext_linear,
  family = "gaussian"
)

lambda1 <- fit_xrnet$penalty[10]
lambda2 <- fit_xrnet$penalty_ext[10]

coef_xrnet <- coef(
  fit_xrnet,
  p = lambda1,
  pext = lambda2,
)
```

## define_enet *Define elastic net regularization object for predictor and external data*

### Description

Helper function to define a elastic net penalty regularization object. See `define_penalty` for more details.

### Usage

```
define_enet(
  en_param = 0.5,
  num_penalty = 20,
  penalty_ratio = NULL,
  user_penalty = NULL,
  custom_multiplier = NULL
)
```

### Arguments

| | |
|---|---|
| `en_param` | elastic net parameter, between 0 and 1 |
| `num_penalty` | number of penalty values to fit in grid. Default is 20. |
| `penalty_ratio` | ratio between minimum and maximum penalty for x. Default is 1e-04 if $n > p$ and 0.01 if $n <= p$. |
| `user_penalty` | user-defined vector of penalty values to use in penalty path. |
| `custom_multiplier` | variable-specific penalty multipliers to apply to overall penalty. Default is 1 for all variables. 0 is no penalization. |

### Value

A list object with regularization settings that are used to define the regularization for predictors or external data in `xrnet` and `tune_xrnet`. The list elements will match those returned by `define_penalty`, but with the penalty_type set to match the value of `en_param`.

## define_lasso *Define lasso regularization object for predictor and external data*

### Description

Helper function to define a lasso penalty regularization object. See `define_penalty` for more details.

## Usage

```
define_lasso(
  num_penalty = 20,
  penalty_ratio = NULL,
  user_penalty = NULL,
  custom_multiplier = NULL
)
```

## Arguments

| | |
|---|---|
| num_penalty | number of penalty values to fit in grid. Default is 20. |
| penalty_ratio | ratio between minimum and maximum penalty for x. Default is 1e-04 if $n > p$ and 0.01 if $n <= p$. |
| user_penalty | user-defined vector of penalty values to use in penalty path. |
| custom_multiplier | variable-specific penalty multipliers to apply to overall penalty. Default is 1 for all variables. 0 is no penalization. |

## Value

A list object with regularization settings that are used to define the regularization for predictors or external data in xrnet and tune_xrnet. The list elements will match those returned by define_penalty, but with the penalty_type automatically set to 1.

---

# define_penalty *Define regularization object for predictor and external data.*

---

## Description

Defines regularization for predictors and external data variables in xrnet fitting. Use helper functions define_lasso, define_ridge, or define_enet to specify a common penalty on x or external.

## Usage

```
define_penalty(
  penalty_type = 1,
  quantile = 0.5,
  num_penalty = 20,
  penalty_ratio = NULL,
  user_penalty = NULL,
  custom_multiplier = NULL
)
```

## Arguments

| | |
|---|---|
| penalty_type | type of regularization. Default is 1 (Lasso). Can supply either a scalar value or vector with length equal to the number of variables the matrix.<br>• 0 = Ridge<br>• (0,1) = Elastic-Net<br>• 1 = Lasso / Quantile |
| quantile | specifies quantile for quantile penalty. Default of 0.5 reduces to lasso (currently not implemented). |
| num_penalty | number of penalty values to fit in grid. Default is 20. |
| penalty_ratio | ratio between minimum and maximum penalty for x. Default is 1e-04 if $n > p$ and 0.01 if $n <= p$. |
| user_penalty | user-defined vector of penalty values to use in penalty path. |
| custom_multiplier | variable-specific penalty multipliers to apply to overall penalty. Default is 1 for all variables. 0 is no penalization. |

## Value

A list object with regularization settings that are used to define the regularization for predictors or external data in xrnet and tune_xrnet:

| | |
|---|---|
| penalty_type | The penalty type, scalar with value in range [0, 1]. |
| quantile | Quantile for quantile penalty, 0.5 defaults to lasso (not currently implemented). |
| num_penalty | The number of penalty values in the penalty path. |
| penalty_ratio | The ratio of the minimum penalty value compared to the maximum penalty value. |
| user_penalty | User-defined numeric vector of penalty values, NULL if not provided by user. |
| custom_multiplier | User-defined feature-specific penalty multipliers, NULL if not provided by user. |

## Examples

```
# define ridge penalty with penalty grid split into 30 values
my_penalty <- define_penalty(penalty_type = 0, num_penalty = 30)

# define elastic net (0.5) penalty with user-defined penalty
my_custom_penalty <- define_penalty(
  penalty_type = 0.5, user_penalty = c(100, 50, 10, 1, 0.1)
)
```

## define_ridge *Define ridge regularization object for predictor and external data*

### Description

Helper function to define a ridge penalty regularization object. See `define_penalty` for more details.

### Usage

```
define_ridge(
  num_penalty = 20,
  penalty_ratio = NULL,
  user_penalty = NULL,
  custom_multiplier = NULL
)
```

### Arguments

| | |
|---|---|
| `num_penalty` | number of penalty values to fit in grid. Default is 20. |
| `penalty_ratio` | ratio between minimum and maximum penalty for x. Default is 1e-04 if $n > p$ and 0.01 if $n <= p$. |
| `user_penalty` | user-defined vector of penalty values to use in penalty path. |
| `custom_multiplier` | variable-specific penalty multipliers to apply to overall penalty. Default is 1 for all variables. 0 is no penalization. |

### Value

A list object with regularization settings that are used to define the regularization for predictors or external data in `xrnet` and `tune_xrnet`. The list elements will match those returned by `define_penalty`, but with the penalty_type automatically set to 0.

## ext_linear *Simulated external data*

### Description

Simulated external data

### Usage

```
ext_linear
```

### Format

A matrix with 50 rows and 4 columns

plot.tune_xrnet *Plot k-fold cross-validation error grid*

## Description

Generates plots to visualize the mean cross-validation error. If no external data was used in the model fit, a plot of the cross-validated error with standard error bars is generated for all penalty values. If external data was used in the model fit, a contour plot of the cross-validated errors is created. Error curves can also be generated for a fixed value of the primary penalty on x (p) or the external penalty (pext) when external data is used.

## Usage

```
## S3 method for class 'tune_xrnet'
plot(x, p = NULL, pext = NULL, ...)
```

## Arguments

| | |
|---|---|
| x | A tune_xrnet class object |
| p | (optional) penalty value for x (for generating an error curve across external penalties). Use value "opt" to use the optimal penalty value. |
| pext | (optional) penalty value for external (for generating an error curve across primary penalties). Use value "opt" to use the optimal penalty value. |
| ... | Additional graphics parameters |

## Details

The parameter values p and pext can be used to generate profiled error curves by fixing either the penalty on x or the penalty on external to a fixed value. You cannot specify both at the same time as this would only return a single point.

## Value

None

## Examples

```
## load example data
data(GaussianExample)

## 5-fold cross validation
cv_xrnet <- tune_xrnet(
  x = x_linear,
  y = y_linear,
  external = ext_linear,
  family = "gaussian",
  control = xrnet_control(tolerance = 1e-6)
```

```
)

## contour plot of cross-validated error
plot(cv_xrnet)

## error curve of external penalties at optimal penalty value
plot(cv_xrnet, p = "opt")
```

---

`predict.tune_xrnet` *Predict function for "tune_xrnet" object*

---

## Description

Extract coefficients or predict response in new data using fitted model from a `tune_xrnet` object. Note that we currently only support returning results that are in the original path(s).

## Usage

```
## S3 method for class 'tune_xrnet'
predict(
  object,
  newdata = NULL,
  newdata_fixed = NULL,
  p = "opt",
  pext = "opt",
  type = c("response", "link", "coefficients"),
  ...
)
```

## Arguments

| | |
|---|---|
| `object` | A `tune_xrnet` object |
| `newdata` | matrix with new values for penalized variables |
| `newdata_fixed` | matrix with new values for unpenalized variables |
| `p` | vector of penalty values to apply to predictor variables. Default is optimal value in tune_xrnet object. |
| `pext` | vector of penalty values to apply to external data variables. Default is optimal value in tune_xrnet object. |
| `type` | type of prediction to make using the xrnet model, options include:<br>• response<br>• link (linear predictor)<br>• coefficients |
| `...` | pass other arguments to xrnet function (if needed) |

## Value

The object returned is based on the value of type as follows:

- response: An array with the response predictions based on the data for each penalty combination
- link: An array with linear predictions based on the data for each penalty combination
- coefficients: A list with the coefficient estimates for each penalty combination. See `coef.xrnet`.

## Examples

```
data(GaussianExample)

## 5-fold cross validation
cv_xrnet <- tune_xrnet(
  x = x_linear,
  y = y_linear,
  external = ext_linear,
  family = "gaussian",
  control = xrnet_control(tolerance = 1e-6)
)

## Get coefficients and predictions at optimal penalty combination
coef_xrnet <- predict(cv_xrnet, type = "coefficients")
pred_xrnet <- predict(cv_xrnet, newdata = x_linear, type = "response")
```

---

# predict.xrnet *Predict function for "xrnet" object*

---

## Description

Extract coefficients or predict response in new data using fitted model from an `xrnet` object. Note that we currently only support returning coefficient estimates that are in the original path(s).

## Usage

```
## S3 method for class 'xrnet'
predict(
  object,
  newdata = NULL,
  newdata_fixed = NULL,
  p = NULL,
  pext = NULL,
  type = c("response", "link", "coefficients"),
  ...
)
```

## Arguments

| | |
|---|---|
| `object` | A `xrnet` object |
| `newdata` | matrix with new values for penalized variables |
| `newdata_fixed` | matrix with new values for unpenalized variables |
| `p` | vector of penalty values to apply to predictor variables |
| `pext` | vector of penalty values to apply to external data variables |
| `type` | type of prediction to make using the xrnet model, options include:<br>• response<br>• link (linear predictor)<br>• coefficients |
| `...` | pass other arguments to xrnet function (if needed) |

## Value

The object returned is based on the value of type as follows:

- response: An array with the response predictions based on the data for each penalty combination
- link: An array with linear predictions based on the data for each penalty combination
- coefficients: A list with the coefficient estimates for each penalty combination. See `coef.xrnet`.

## Examples

```
data(GaussianExample)

fit_xrnet <- xrnet(
  x = x_linear,
  y = y_linear,
  external = ext_linear,
  family = "gaussian"
)

lambda1 <- fit_xrnet$penalty[10]
lambda2 <- fit_xrnet$penalty_ext[10]

coef_xrnet <- predict(
  fit_xrnet,
  p = lambda1,
  pext = lambda2,
  type = "coefficients"
)

pred_xrnet <- predict(
  fit_xrnet,
  p = lambda1,
  pext = lambda2,
  newdata = x_linear,
  type = "response"
)
```

tune_xrnet *k-fold cross-validation for hierarchical regularized regression*

## Description

k-fold cross-validation for hierarchical regularized regression xrnet

## Usage

```
tune_xrnet(
  x,
  y,
  external = NULL,
  unpen = NULL,
  family = c("gaussian", "binomial"),
  penalty_main = define_penalty(),
  penalty_external = define_penalty(),
  weights = NULL,
  standardize = c(TRUE, TRUE),
  intercept = c(TRUE, FALSE),
  loss = c("deviance", "mse", "mae", "auc"),
  nfolds = 5,
  foldid = NULL,
  parallel = FALSE,
  control = list()
)
```

## Arguments

| | |
|---|---|
| x | predictor design matrix of dimension $nxp$, matrix options include:<br>• matrix<br>• big.matrix<br>• filebacked.big.matrix<br>• sparse matrix (dgCMatrix) |
| y | outcome vector of length $n$ |
| external | (optional) external data design matrix of dimension $pxq$, matrix options include:<br>• matrix<br>• sparse matrix (dgCMatrix) |
| unpen | (optional) unpenalized predictor design matrix, matrix options include:<br>• matrix |
| family | error distribution for outcome variable, options include:<br>• "gaussian"<br>• "binomial" |

`penalty_main` specifies regularization object for x. See `define_penalty` for more details.

`penalty_external` specifies regularization object for external. See `define_penalty` for more details. See `define_penalty` for more details.

`weights` optional vector of observation-specific weights. Default is 1 for all observations.

`standardize` indicates whether x and/or external should be standardized. Default is c(TRUE, TRUE).

`intercept` indicates whether an intercept term is included for x and/or external. Default is c(TRUE, FALSE).

`loss` loss function for cross-validation. Options include:

- "deviance"
- "mse" (Mean Squared Error)
- "mae" (Mean Absolute Error)
- "auc" (Area under the curve)

`nfolds` number of folds for cross-validation. Default is 5.

`foldid` (optional) vector that identifies user-specified fold for each observation. If NULL, folds are automatically generated.

`parallel` use `foreach` function to fit folds in parallel if TRUE, must register cluster (`doParallel`) before using.

`control` specifies xrnet control object. See `xrnet_control` for more details.

## Details

k-fold cross-validation is used to determine the 'optimal' combination of hyperparameter values, where optimal is based on the optimal value obtained for the user-selected loss function across the k folds. To efficiently traverse all possible combinations of the hyperparameter values, 'warm-starts' are used to traverse the penalty from largest to smallest penalty value(s). Note that the penalty grid for the folds is generated by fitting the model on the entire training data. Parallelization is enabled through the `foreach` and `doParallel` R packages. To use parallelization, `parallel = TRUE`, you must first create the cluster `makeCluster` and then register the cluster `registerDoParallel`. See the `parallel`, `foreach`, and/or `doParallel` R packages for more details on how to setup parallelization.

## Value

A list of class `tune_xrnet` with components

`cv_mean` mean cross-validated error for each penalty combination. Object returned is a vector if there is no external data (external = NULL) and matrix if there is external data.

`cv_sd` estimated standard deviation for cross-validated errors. Object returned is a vector if there is no external data (external = NULL) and matrix if there is external data.

`loss` loss function used to compute cross-validation error

`opt_loss` the value of the loss function for the optimal cross-validated error

| | |
|---|---|
| opt_penalty | first-level penalty value that achieves the optimal loss |
| opt_penalty_ext | second-level penalty value that achieves the optimal loss (if external data is present) |
| fitted_model | fitted xrnet object using all data, see xrnet for details of object |

## Examples

```
## cross validation of hierarchical linear regression model
data(GaussianExample)

## 5-fold cross validation
cv_xrnet <- tune_xrnet(
  x = x_linear,
  y = y_linear,
  external = ext_linear,
  family = "gaussian",
  control = xrnet_control(tolerance = 1e-6)
)

## contour plot of cross-validated error
plot(cv_xrnet)
```

---

# xrnet *Fit hierarchical regularized regression model*

---

## Description

Fits hierarchical regularized regression model that enables the incorporation of external data for predictor variables. Both the predictor variables and external data can be regularized by the most common penalties (lasso, ridge, elastic net). Solutions are computed across a two-dimensional grid of penalties (a separate penalty path is computed for the predictors and external variables). Currently support regularized linear and logistic regression, future extensions to other outcomes (i.e. Cox regression) will be implemented in the next major update.

## Usage

```
xrnet(
  x,
  y,
  external = NULL,
  unpen = NULL,
  family = c("gaussian", "binomial"),
  penalty_main = define_penalty(),
  penalty_external = define_penalty(),
  weights = NULL,
  standardize = c(TRUE, TRUE),
  intercept = c(TRUE, FALSE),
```

```
  control = list()
)
```

## Arguments

| | |
|---|---|
| x | predictor design matrix of dimension $nxp$, matrix options include:<br>• matrix<br>• big.matrix<br>• filebacked.big.matrix<br>• sparse matrix (dgCMatrix) |
| y | outcome vector of length $n$ |
| external | (optional) external data design matrix of dimension $pxq$, matrix options include:<br>• matrix<br>• sparse matrix (dgCMatrix) |
| unpen | (optional) unpenalized predictor design matrix, matrix options include:<br>• matrix |
| family | error distribution for outcome variable, options include:<br>• "gaussian"<br>• "binomial" |
| penalty_main | specifies regularization object for x. See `define_penalty` for more details. |
| penalty_external | specifies regularization object for external. See `define_penalty` for more details. |
| weights | optional vector of observation-specific weights. Default is 1 for all observations. |
| standardize | indicates whether x and/or external should be standardized. Default is c(TRUE, TRUE). |
| intercept | indicates whether an intercept term is included for x and/or external. Default is c(TRUE, FALSE). |
| control | specifies xrnet control object. See `xrnet_control` for more details. |

## Details

This function extends the coordinate descent algorithm of the R package `glmnet` to allow the type of regularization (i.e. ridge, lasso) to be feature-specific. This extension is used to enable fitting hierarchical regularized regression models, where external information for the predictors can be included in the `external=` argument. In addition, elements of the R package `biglasso` are utilized to enable the use of standard R matrices, memory-mapped matrices from the `bigmemory` package, or sparse matrices from the `Matrix` package.

## Value

A list of class `xrnet` with components:

| | |
|---|---|
| beta0 | matrix of first-level intercepts indexed by penalty values |

| | |
|---|---|
| betas | 3-dimensional array of first-level penalized coefficients indexed by penalty values |
| gammas | 3-dimensional array of first-level non-penalized coefficients indexed by penalty values |
| alpha0 | matrix of second-level intercepts indexed by penalty values |
| alphas | 3-dimensional array of second-level external data coefficients indexed by penalty values |
| penalty | vector of first-level penalty values |
| penalty_ext | vector of second-level penalty values |
| family | error distribution for outcome variable |
| num_passes | total number of passes over the data in the coordinate descent algorithm |
| status | error status for xrnet fitting |

- 0 = OK
- 1 = Error/Warning

| | |
|---|---|
| error_msg | description of error |

## References

Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software, 33(1), 1-22. URL http://www.jstatsoft.org/v33/i01/.

Zeng, Y., and Breheny, P. (2017). The biglasso Package: A Memory- and Computation-Efficient Solver for Lasso Model Fitting with Big Data in R. arXiv preprint arXiv:1701.05936. URL https://arxiv.org/abs/1701.05936.

Michael J. Kane, John Emerson, Stephen Weston (2013). Scalable Strategies for Computing with Massive Data. Journal of Statistical Software, 55(14), 1-19. URL http://www.jstatsoft.org/v55/i14/.

## Examples

```
### hierarchical regularized linear regression ###
data(GaussianExample)

## define penalty for predictors and external variables
## default is ridge for predictors and lasso for external
## see define_penalty() function for more details

penMain <- define_penalty(0, num_penalty = 20)
penExt <- define_penalty(1, num_penalty = 20)

## fit model with defined regularization
fit_xrnet <- xrnet(
  x = x_linear,
  y = y_linear,
  external = ext_linear,
  family = "gaussian",
  penalty_main = penMain,
  penalty_external = penExt
)
```

`xrnet_control` *Control function for xrnet fitting*

## Description

Control function for `xrnet` fitting.

## Usage

```
xrnet_control(
  tolerance = 1e-08,
  max_iterations = 1e+05,
  dfmax = NULL,
  pmax = NULL,
  lower_limits = NULL,
  upper_limits = NULL
)
```

## Arguments

| | |
|---|---|
| `tolerance` | positive convergence criterion. Default is 1e-08. |
| `max_iterations` | maximum number of iterations to run coordinate gradient descent across all penalties before returning an error. Default is 1e+05. |
| `dfmax` | maximum number of variables allowed in model. Default is $ncol(x)+ncol(unpen)+ncol(external)+intercept[1]+intercept[2]$. |
| `pmax` | maximum number of variables with nonzero coefficient estimate. Default is $min(2*dfmax+20, ncol(x)+ncol(unpen)+ncol(external)+intercept[2])$. |
| `lower_limits` | vector of lower limits for each coefficient. Default is -Inf for all variables. |
| `upper_limits` | vector of upper limits for each coefficient. Default is Inf for all variables. |

## Value

A list object with the following components:

| | |
|---|---|
| `tolerance` | The coordinate descent stopping criterion. |
| `dfmax` | The maximum number of variables that will be allowed in the model. |
| `pmax` | The maximum number of variables with nonzero coefficient estimate. |
| `lower_limits` | Feature-specific numeric vector of lower bounds for coefficient estimates |
| `upper_limits` | Feature-specific numeric vector of upper bounds for coefficient estimates |

## x_linear *Simulated example data for hierarchical regularized linear regression*

### Description

Simulated example data for hierarchical regularized linear regression

### Usage

```
x_linear
```

### Format

A matrix with 100 rows and 50 variables

## y_linear *Simulated outcome data*

### Description

Simulated outcome data

### Usage

```
y_linear
```

### Format

A vector with 100 elements

# Index