

# Package: RAMP (via r-universe)

July 2, 2026

**Type** Package

**Title** Regularized Generalized Linear Models with Interaction Effects

**Version** 2.0.2

**Date** 2020-01-03

**Description** Provides an efficient procedure for fitting the entire solution path for high-dimensional regularized quadratic generalized linear models with interactions effects under the strong or weak heredity constraint.

**License** GPL-2

**RoxygenNote** 7.0.2

**NeedsCompilation** yes

**Author** Yang Feng [aut, cre], Ning Hao [aut], Hao Helen Zhang [aut]

**Maintainer** Yang Feng <yangfengstat@gmail.com>

**Repository** <https://cran.r-universe.dev>

**Date/Publication** 2020-01-16 12:20:29 UTC

**RemoteUrl** <https://github.com/cran/RAMP>

**RemoteRef** HEAD

**RemoteSha** 8695e26f0eada99a873da35b5f330f6b1f299b07

## Contents

predict.RAMP . . . . .	2
print.RAMP . . . . .	3
RAMP . . . . .	3

<b>Index</b>	<b>6</b>
--------------	----------

---

predict.RAMP                      *Model prediction based on a fitted RAMP object.*

---

### Description

Similar to the usual predict methods, this function returns predictions from a fitted 'RAMP' object.

### Usage

```
## S3 method for class 'RAMP'
predict(object, newdata = NULL, type = c("link",
    "response", "class"), allpath = FALSE, ...)
```

### Arguments

object	Fitted 'RAMP' model object.
newdata	Matrix of new values for x at which predictions are to be made, without the intercept term.
type	Type of prediction required. Type 'response' gives the fitted values for 'gaussian', fitted probabilities for 'binomial', fitted mean for 'poisson', and the fitted relative risk for 'cox'. Type 'link' returns the linear predictors for 'binomial', 'poisson' and 'cox' models; for 'gaussian' models it is equivalent to type 'response'. Type 'class' applies only to 'binomial' models, and produces the class label corresponding to the maximum probability (0-1 labels).
allpath	allpath = T will output all the predictions on the solution path. allpath = FALSE will only output the one the criterion selected in the 'RAMP' object.
...	Not used. Other arguments to predict.

### Value

The object returned depends on type.

### See Also

[RAMP](#), [print.RAMP](#)

---

print.RAMP	<i>Result summary of a fitted RAMP object.</i>
------------	--

---

### Description

Similar to the usual print methods, this function summarize results from a fitted 'RAMP' object.

### Usage

```
## S3 method for class 'RAMP'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

### Arguments

x	Fitted 'RAMP' model object.
digits	The number of significant digits for the coefficient estimates.
...	Not used. Other arguments to predict.

### Value

No value is returned.

### See Also

[RAMP](#)

---

RAMP	<i>Regularization Algorithm under Marginality Principle (RAMP) for high dimensional generalized quadratic regression.</i>
------	---

---

### Description

Regularization Algorithm under Marginality Principle (RAMP) for high dimensional generalized quadratic regression.

### Usage

```
RAMP(X, y, family = "gaussian", penalty = "LASSO", gamma = NULL,
      inter = TRUE, hier = "Strong", eps = 1e-15, tune = "EBIC",
      penalty.factor = rep(1, ncol(X)), inter.penalty.factor = 1, lam.list,
      lambda.min.ratio, max.iter = 100, max.num, n.lambda = 100,
      ebic.gamma = 1, refit = TRUE, trace = FALSE)
```

**Arguments**

<code>X</code>	input matrix, of dimension <code>nobs x nvars</code> ; each row is an observation vector.
<code>y</code>	response variable, of dimension <code>nobs x 1</code> . continuous for <code>family='gaussian'</code> , binary for <code>family='binomial'</code> .
<code>family</code>	response type. Default is <code>'gaussian'</code> . The other choice is <code>'binomial'</code> for logistic regression.
<code>penalty</code>	Choose from <code>LASSO</code> , <code>SCAD</code> and <code>MCP</code> . Default is <code>'LASSO'</code> .
<code>gamma</code>	concavity parameter. If <code>NULL</code> , the code will use 3.7 for <code>'SCAD'</code> and 2.7 for <code>'MCP'</code> .
<code>inter</code>	whether to select interaction effects. Default is <code>TRUE</code> .
<code>hier</code>	whether to enforce strong or weak heredity. Default is <code>'Strong'</code> .
<code>eps</code>	the precision used to test the convergence. Default is <code>1e-15</code> .
<code>tune</code>	tuning parameter selection method. <code>'AIC'</code> , <code>'BIC'</code> , <code>'EBIC'</code> and <code>'GIC'</code> are available options. Default is <code>EBIC</code> .
<code>penalty.factor</code>	A multiplicative factor for the penalty applied to each coefficient. If supplied, <code>penalty.factor</code> must be a numeric vector of length equal to the number of columns of <code>X</code> . The purpose of <code>penalty.factor</code> is to apply differential penalization if some coefficients are thought to be more likely than others to be in the model. In particular, <code>penalty.factor</code> can be 0, in which case the coefficient is always in the model without shrinkage.
<code>inter.penalty.factor</code>	the penalty factor for interactions effects. Default is 1. larger value discourage interaction effects.
<code>lam.list</code>	a user supplied $\lambda$ sequence. typical usage is to have the program compute its own <code>lambda</code> sequence based on <code>lambda.min.ratio</code> and <code>n.lambda</code> . supplying a value of $\lambda$ overrides this.
<code>lambda.min.ratio</code>	optional input. smallest value for <code>lambda</code> , as a fraction of <code>max.lam</code> , the (data derived) entry value. the default depends on the sample size <code>n</code> relative to the number of variables <code>p</code> . if <code>n &gt; p</code> , the default is 0.0001. otherwise, the default is 0.01.
<code>max.iter</code>	maximum number of iteration in the computation. Default is 100.
<code>max.num</code>	optional input. maximum number of nonzero coefficients.
<code>n.lambda</code>	the number of <code>lambda</code> values. Default is 100.
<code>ebic.gamma</code>	the gamma parameter value in the EBIC criteria. Default is 1.
<code>refit</code>	whether to perform a MLE refit on the selected model. Default is <code>TRUE</code> .
<code>trace</code>	whether to trace the fitting process. Default is <code>FALSE</code> .

**Value**

An object with S3 class `RAMP`.

`a0` intercept vector of `length(lambda)`.

mainInd	index for the selected main effects.
interInd	index for the selected interaction effects
beta.m	coefficients for the selected main effects.
beta.i	coefficients for the selected interaction effects.

**See Also**

[predict.RAMP](#), [print.RAMP](#)

**Examples**

```

set.seed(0)
n = 500
p = 10 #Can be changed to a much larger number say 50000
x = matrix(rnorm(n*p),n,p)
eta = 1 * x[,1] + 2 * x[,3] + 3*x[,6] + 4*x[,1]*x[,3] + 5*x[,1]*x[,6]
y = eta + rnorm(n)
xtest = matrix(rnorm(n*p),n,p)
eta.test = 1 * xtest[,1] + 2 * xtest[,3] + 3*xtest[,6] +
4*xtest[,1]*xtest[,3] + 5*xtest[,1]*xtest[,6]
ytest = eta.test + rnorm(n)
fit1 = RAMP(x, y)
fit1 ###examine the results
ypred = predict(fit1, xtest)
mean((ypred-ytest)^2)

#fit1.scad = RAMP(x, y, penalty = 'SCAD')
#fit1.scad ###examine the results

#fit1.mcp = RAMP(x, y, penalty = 'MCP')
#fit1.mcp ###examine the results

##Now, try a binary response
#y = rbinom(n, 1, 1/(1+exp(-eta)))
#fit2 = RAMP(x, y, family='binomial') ###for binary response

## Weak heredity
eta = 1 * x[,1] + 3*x[,6] + 4*x[,1]*x[,3] + 5*x[,1]*x[,6]
y = eta + rnorm(n)
eta.test = 1 * xtest[,1] + 3*xtest[,6] + 4*xtest[,1]*xtest[,3] +
5*xtest[,1]*xtest[,6]
ytest = eta.test + rnorm(n)

fit3 = RAMP(x, y, hier = 'Strong')
fit3 ###examine the results
ypred3 = predict(fit3, xtest)
mean((ypred3-ytest)^2)
fit4 = RAMP(x, y, hier = 'Weak')
fit4
ypred4 = predict(fit4, xtest)
mean((ypred4-ytest)^2)

```

# Index

`predict.RAMP`, 2, 5

`print.RAMP`, 2, 3, 5

`RAMP`, 2, 3, 3