

# Package ‘earth’

April 12, 2019

**Version** 5.1.1

**Title** Multivariate Adaptive Regression Splines

**Author** Stephen Milborrow. Derived from mda:mars by Trevor Hastie and Rob Tibshirani. Uses Alan Miller's Fortran utilities with Thomas Lumley's leaps wrapper.

**Maintainer** Stephen Milborrow <milbo@sonic.net>

**Depends** R (>= 3.4.0), Formula (>= 1.2-3), plotmo (>= 3.5.4), TeachingDemos (>= 2.10)

**Suggests** gam (>= 1.16), mgcv (>= 1.8-26), mda (>= 0.4-10), MASS (>= 7.3-51)

**Description** Build regression models using the techniques in Friedman's papers "Fast MARS" and "Multivariate Adaptive Regression Splines" <doi:10.1214/aos/1176347963>. (The term "MARS" is trademarked and thus not used in the name of the package.)

**License** GPL-3

**LazyData** yes

**URL** <http://www.milbo.users.sonic.net/earth>

**Repository** CRAN

**Date/Publication** 2019-04-12 10:52:57 UTC

**RoxygenNote** 5.1.1

**NeedsCompilation** yes

## R topics documented:

contr.earth.response . . . . .	2
earth . . . . .	3
earth.object . . . . .	11
etitanic . . . . .	16
evimp . . . . .	18
expand.bpairs . . . . .	19

format.earth . . . . .	21
mars.to.earth . . . . .	24
model.matrix.earth . . . . .	25
ozone1 . . . . .	26
plot.earth . . . . .	27
plot.earth.models . . . . .	30
plot.evimp . . . . .	32
plot.varmod . . . . .	33
plotd . . . . .	34
predict.earth . . . . .	39
predict.varmod . . . . .	41
residuals.earth . . . . .	42
summary.earth . . . . .	44
update.earth . . . . .	46
varmod . . . . .	47

## Index 50

contr.earth.response *Please ignore*

### Description

Contrasts function for factors in the `earth` response. For internal use by `earth`.

### Usage

```
contr.earth.response(x, base, contrasts)
```

### Arguments

x	a factor
base	unused
contrasts	unused

### Value

Returns a diagonal matrix. An example for a 3 level factor with levels A, B, and C:

```

  A B C
A 1 0 0
B 0 1 0
C 0 0 1
```

### Note

`Earth` uses this function internally. You shouldn't need it. It is made publicly available only because it seems that is necessary for `model.matrix`.

**See Also**[contrasts](#)

---

**earth***Multivariate Adaptive Regression Splines*

---

**Description**

Build a regression model using the techniques in Friedman's papers "Multivariate Adaptive Regression Splines" and "Fast MARS".

See the package vignette "[Notes on the earth package](#)".

**Usage**

```
## S3 method for class 'formula'
earth(formula = stop("no 'formula' argument"), data = NULL,
      weights = NULL, wp = NULL, subset = NULL,
      na.action = na.fail,
      pmethod = c("backward", "none", "exhaustive", "forward", "seqrep", "cv"),
      keepxy = FALSE, trace = 0, glm = NULL, degree = 1, nprune = NULL,
      nfold=0, ncross=1, stratify=TRUE,
      varmod.method = "none", varmod.exponent = 1,
      varmod.conv = 1, varmod.clamp = .1, varmod.minspan = -3,
      Scale.y = NULL, ...)

## Default S3 method:
earth(x = stop("no 'x' argument"), y = stop("no 'y' argument"),
      weights = NULL, wp = NULL, subset = NULL,
      na.action = na.fail,
      pmethod = c("backward", "none", "exhaustive", "forward", "seqrep", "cv"),
      keepxy = FALSE, trace = 0, glm = NULL, degree = 1, nprune = NULL,
      nfold=0, ncross=1, stratify=TRUE,
      varmod.method = "none", varmod.exponent = 1,
      varmod.conv = 1, varmod.clamp = .1, varmod.minspan = -3,
      Scale.y = NULL, ...)

## S3 method for class 'fit'
earth(x = stop("no 'x' argument"), y = stop("no 'y' argument"),
      weights = NULL, wp = NULL, subset = NULL,
      na.action = na.fail, offset = NULL,
      pmethod = c("backward", "none", "exhaustive", "forward", "seqrep", "cv"),
      keepxy = FALSE, trace = 0, glm = NULL, degree = 1,
      penalty = if(degree > 1) 3 else 2,
      nk = min(200, max(20, 2 * ncol(x))) + 1,
      thresh = 0.001, minspan = 0, endspan = 0,
      newvar.penalty = 0, fast.k = 20, fast.beta = 1,
```

```
linpreds = FALSE, allowed = NULL,
nprune = NULL, Object = NULL,
Scale.y = NULL, Adjust.endspan = 2, Auto.linpreds = TRUE,
Force.weights = FALSE, Use.beta.cache = TRUE, Force.xtx.prune = FALSE,
Get.leverages = NROW(x) < 1e5, Exhaustive.tol = 1e-10, ...)
```

## Arguments

To start off, look at the arguments `formula`, `data`, `x`, `y`, `nk`, `degree`, and `trace`. If the response is binary or a factor, consider using the `glm` argument. For cross validation, use the `nfold` argument. For prediction intervals, use the `varmod.method` argument.

Most users will find that the above arguments are all they need, plus in some cases `keepxy` and `nprune`. Unless you are a knowledgeable user, it's best not to subvert the standard algorithm by toying with tuning parameters such as `thresh`, `penalty`, and `endspan`.

	Model formula.
<code>formula</code>	Data frame for formula.
<code>x</code>	Matrix or dataframe containing the independent variables.
<code>y</code>	Vector containing the response variable, or, in the case of multiple responses, a matrix or dataframe whose columns are the values for each response.
<code>subset</code>	Index vector specifying which cases to use, i.e., which rows in <code>x</code> to use. Default is <code>NULL</code> , meaning all.
<code>weights</code>	Case weights. Default is <code>NULL</code> , meaning no case weights. If specified, <code>weights</code> must have length equal to <code>nrow(x)</code> before applying <code>subset</code> . Zero weights are converted to a very small nonzero value. In the current implementation, building models with weights can be slow.
<code>wp</code>	Response weights. Default is <code>NULL</code> , meaning no response weights. If specified, <code>wp</code> must have an element for each column of <code>y</code> (after <code>factors</code> in <code>y</code> , if any, have been expanded). Zero weights are converted to a very small nonzero value.
<code>na.action</code>	NA action. Default is <code>na.fail</code> , and only <code>na.fail</code> is supported.
<code>offset</code>	Offset term passed from the formula in <code>earth.formula</code> .
<code>keepxy</code>	Default is <code>FALSE</code> . Set to <code>TRUE</code> to retain the following in the returned value: <code>x</code> and <code>y</code> (or <code>data</code> ), <code>subset</code> , and <code>weights</code> . The function <code>update.earth</code> and friends will use these if present instead of searching for them in the environment at the time <code>update.earth</code> is invoked.

When the `nfold` argument is used with `keepxy=TRUE`, `earth` keeps more data and calls `predict.earth` multiple times to generate `cv.oof.rsq.tab` and `cv.infold.rsq.tab` (see the `cv.` arguments in `earth.object`). It therefore makes cross-validation significantly slower.

<code>trace</code>	Trace <code>earth</code> 's execution. Values: <code>0</code> (default) no tracing
--------------------	---

.3 variance model (the varmod.method arg)  
 .5 cross validation (the nfold arg)  
 1 overview  
 2 forward pass  
 3 pruning  
 4 model mats summary, pruning details  
 5 full model mats, internal details of operation

glm NULL (default) or a list of arguments to pass on to `glm`. See the documentation of `glm` for a description of these arguments See “*Generalized linear models*” in the vignette. Example:  
`earth(survived~., data=etitanic, degree=2, glm=list(family=binomial))`

**The following arguments are for the forward pass.**

degree Maximum degree of interaction (Friedman’s *mi*). Default is 1, meaning build an additive model (i.e., no interaction terms).

penalty Generalized Cross Validation (GCV) penalty per knot. Default is `if(degree>1) 3 else 2`. Simulation studies suggest values in the range of about 2 to 4. The FAQ section in the vignette has some information on GCVs.  
 Special values (for use by knowledgeable users): The value 0 penalizes only terms, not knots. The value -1 means no penalty, so  $GCV = RSS/n$ .

nk Maximum number of model terms before pruning, i.e., the maximum number of terms created by the forward pass. Includes the intercept.  
 The actual number of terms created by the forward pass will often be less than nk because of other stopping conditions. See “*Termination conditions for the forward pass*” in the vignette.  
 The default is semi-automatically calculated from the number of predictors but may need adjusting.

thresh Forward stepping threshold. Default is 0.001. This is one of the arguments used to decide when forward stepping should terminate: the forward pass terminates if adding a term changes  $RSq$  by less than thresh. See “*Termination conditions for the forward pass*” in the vignette.

minspan Minimum number of observations between knots. (This increases resistance to runs of correlated noise in the input data.)  
 The default `minspan=0` is treated specially and means calculate the minspan internally, as per Friedman’s MARS paper section 3.8 with  $alpha = 0.05$ . Set `trace>=2` to see the calculated value.  
 Use `minspan=1` and `endspan=1` to consider all x values.  
 Negative values of minspan specify the maximum number of knots per predictor. These will be equally spaced. For example, `minspan=-3` allows three evenly spaced knots for each predictor. As always, knots that fall in the end zones specified by `endspan` will be ignored.

endspan Minimum number of observations before the first and after the final knot.  
 The default `endspan=0` is treated specially and means calculate the endspan internally, as per the MARS paper equation 45 with  $alpha = 0.05$ . Set `trace>=2` to see the calculated value.  
 Be wary of reducing endspan, especially if you plan to make predictions beyond

or near the limits of the training data. Overfitting near the edges of training data is much more likely with a small endspan. The model's RSq and GRSq won't indicate when this overfitting is occurring. (A `plotmo` plot can help: look for sharp hinges at the edges of the data). See also the `Adjust.endspan` argument.

<code>newvar.penalty</code>	<p>Penalty for adding a new variable in the forward pass (Friedman's <i>gamma</i>, equation 74 in the MARS paper). Default is 0, meaning no penalty for adding a new variable. Useful non-zero values typically range from about 0.01 to 0.2 and sometimes higher — you will need to experiment.</p> <p>A word of explanation. With the default <code>newvar.penalty=0</code>, if two variables have nearly the same effect (e.g. they are collinear), at any step in the forward pass <code>earth</code> will arbitrarily select one or the other (depending on noise in the sample). Both variables can appear in the final model, complicating model interpretation. On the other hand with a non-zero <code>newvar.penalty</code>, the forward pass will be reluctant to add a new variable — it will rather try to use a variable already in the model, if that does not affect RSq too much. The resulting final model may be easier to interpret, if you are lucky. There will often be a small performance hit (a worse GCV).</p>
<code>fast.k</code>	<p>Maximum number of parent terms considered at each step of the forward pass. (This speeds up the forward pass. See the Fast MARS paper section 3.0.) Default is 20. A value of 0 is treated specially (as being equivalent to infinity), meaning no Fast MARS. Typical values, apart from 0, are 20, 10, or 5.</p> <p>In general, with a lower <code>fast.k</code> (say 5), <code>earth</code> is faster; with a higher <code>fast.k</code>, or with <code>fast.k</code> disabled (set to 0), <code>earth</code> builds a better model. However, because of random variation this general rule often doesn't apply.</p>
<code>fast.beta</code>	<p>Fast MARS ageing coefficient, as described in the Fast MARS paper section 3.1. Default is 1. A value of 0 sometimes gives better results.</p>
<code>linpreds</code>	<p>Index vector specifying which predictors should enter linearly, as in <code>lm</code>. The default is FALSE, meaning predictors enter in the standard MARS fashion, i.e., in hinge functions.</p>

The `linpreds` argument does not specify that a predictor *must* enter the model; only that if it enters, it enters linearly. See “*The linpreds argument*” in the [vignette](#).

See also the `Auto.linpreds` argument below (which describes how `earth` will *automatically* treat a predictor as linear under certain conditions).

Details: A predictor's index in `linpreds` is the column number in the input matrix `x` (after factors have been expanded).

`linpreds=TRUE` makes all predictors enter linearly (the TRUE gets recycled).

`linpreds` may be a character vector e.g. `linpreds=c("wind", "vis")`. Note: `grep` is used for matching. Thus "wind" will match all variables that have "wind" in their names. Use "`^wind$`" to match only the variable named "wind".

<code>allowed</code>	<p>Function specifying which predictors can interact and how. Default is NULL, meaning all standard MARS terms are allowed.</p> <p>During the forward pass, <code>earth</code> calls the <code>allowed</code> function before considering a term for inclusion; the term can go into the model only if the <code>allowed</code> function returns TRUE. See “<i>The allowed argument</i>” in the vignette.</p>
----------------------	---

**The following arguments are for the pruning pass.**

pmethod	Pruning method. One of: backward none exhaustive forward seqrep cv. Default is "backward". Specify pmethod="cv" to use cross-validation to select the number of terms. This selects the number of terms that gives the maximum mean out-of-fold RSq on the fold models. Requires the nfold argument. Use "none" to retain all the terms created by the forward pass. If y has multiple columns, then only "backward" or "none" is allowed. Pruning can take a while if "exhaustive" is chosen and the model is big (more than about 30 terms). The current version of the <a href="#">leaps</a> package used during pruning does not allow user interrupts (i.e., you have to kill your R session to interrupt; in Windows use the Task Manager or from the command line use taskkill).
nprune	Maximum number of terms (including intercept) in the pruned model. Default is NULL, meaning all terms created by the forward pass (but typically not all terms will remain after pruning). Use this to enforce an upper bound on the model size (that is less than nk), or to reduce exhaustive search time with pmethod="exhaustive".

**The following arguments are for cross validation.**

nfold	Number of cross-validation folds. Default is 0, no cross validation. If greater than 1, earth first builds a standard model as usual with all the data. It then builds nfold cross-validated models, measuring R-Squared on the out-of-fold (left out) data each time. The final cross validation R-Squared (CVRSq) is the mean of these out-of-fold R-Squareds. The above process of building nfold models is repeated ncross times (by default, once). Use trace=.5 to trace cross-validation. Further statistics are calculated if keepxy=TRUE or if a binomial or poisson model (specified with the glm argument). See " <i>Cross validation</i> " in the vignette.
ncross	Only applies if nfold>1. Number of cross-validations. Each cross-validation has nfold folds. Default 1.
stratify	Only applies if nfold>1. Default is TRUE. Stratify the cross-validation samples so that an approximately equal number of cases with a non-zero response occur in each cross validation subset. So if the response y is logical, the TRUEs will be spread evenly across folds. And if the response is a multilevel factor, there will be an approximately equal number of each factor level in each fold (because a multilevel factor response gets expanded to columns of zeros and ones, see " <i>Factors</i> " in the vignette). We say "approximately equal" because the number of occurrences of a factor level may not be exactly divisible by the number of folds.

**The following arguments are for variance models.**

varmod.method	Construct a variance model. For details, see <a href="#">varmod</a> and the vignette " <a href="#">Variance models in earth</a> ". Use trace=.3 to trace construction of the variance model. This argument requires nfold and ncross. (We suggest at least ncross=30 here to properly calculate the variance of the errors — although you can use a smaller
---------------	--

value, say 3, for debugging.)

The `varmod.method` argument should be one of

"none" Default. Don't build a variance model.

"const" Assume homoscedastic errors.

"lm" Use `lm` to estimate standard deviation as a function of the predicted response.

"rlm" Use `rlm`.

"earth" Use `earth`.

"gam" Use `gam`. This will use either `gam` or the `mgcv` package, whichever is loaded.

"power" Estimate standard deviation as  $\text{intercept} + \text{coef} * \text{predicted.response}^{\text{exponent}}$ , where `intercept`, `coef`, and `exponent` will be estimated by `nls`. This is equivalent to `varmod.method="lm"` except that `exponent` is automatically estimated instead of being held at the value set by the `varmod.exponent` argument.

"power0" Same as "power" but no intercept (offset) term.

"x.lm", "x.rlm", "x.earth", "x.gam" Like the similarly named options above, but estimate standard deviation by regressing on the predictors `x` (instead of the predicted response). A current implementation restriction is that "x.gam" allows only models with one predictor (`x` must have only one column).

`varmod.exponent`

Power transform applied to the rhs before regressing the absolute residuals with the specified `varmod.method`. Default is 1.

For example, with `varmod.method="lm"`, if you expect the standard deviance to increase linearly with the mean response, use `varmod.exponent=1`. If you expect the standard deviance to increase with the square root of the mean response, use `varmod.exponent=.5` (where negative response values will be treated as 0, and you will get an error message if more than 20% of them are negative).

`varmod.conv`

Convergence criterion for the Iteratively Reweighted Least Squares used when creating the variance model.

Iterations stop when the mean value of the coefficients of the residual model change by less than `varmod.conv` percent. Default is 1 percent.

Negative values force the specified number of iterations, e.g. `varmod.conv=-2` means iterate twice.

Positive values are ignored for `varmod="const"` and also currently ignored for `varmod="earth"` (these are iterated just once, the same as using `varmod.conv=-1`).

`varmod.clamp`

The estimated standard deviation of the main model errors is forced to be at least a small positive value, which we call `min.sd`. This prevents negative or absurdly small estimated standard deviations. Clamping takes place in `predict.varmod`, which is called by `predict.earth` when estimating prediction intervals. The value of `min.sd` is determined when building the variance model as `min.sd = varmod.clamp * mean(sd)`. The default `varmod.clamp` is 0.1.

`varmod.minspan`

Only applies when `varmod.method="earth"` or "x.earth". This is the `minspan` used in the internal call to `earth` when creating the variance model (not the main `earth` model).

Default is -3, i.e., three evenly spaced knots per predictor. Residuals tend to be very noisy, and allowing only this small number of knots helps prevent overfitting.



**The following arguments are for internal or advanced use.**

Object	Earth object to be updated, for use by <code>update.earth</code> .
Scale.y	<p><code>Scale</code> the response internally in the forward pass. Scaling here means subtract the mean and divide by the standard deviation.</p> <p>For single-response models, the default is <code>Scale.y = TRUE</code>. Scaling is invisible to the user, up to numerical differences, but does provide better numeric stability. For multiple-response models, the default is <code>FALSE</code>. If <code>Scale.y</code> is set <code>TRUE</code>, each column of the response is independently scaled. This can prevent one response from “overwhelming” the others, and earth typically generates a different set of hinge functions.</p>
Adjust.endspan	<p>In interaction terms, <code>endspan</code> gets multiplied by this value. This reduces the possibility of an overfitted interaction term supported by just a few cases on the boundary of the predictor space (as sometimes seen in our simulation studies). The default is 2. Use <code>Adjust.endspan=1</code> for compatibility with old versions of earth.</p>
Auto.linpreds	<p>Default is <code>TRUE</code>, which works as follows (see <a href="#">example</a>):</p> <p>At any step in the forward pass, if earth discovers that the best knot for the best predictor is at the predictor minimum (in the training data), then earth adds the predictor to the model as a linear “basis function” (with no hinge). Compare the following basis functions (printed in bold) for an example such predictor <code>x</code>:</p> <p><code>Auto.linpreds=TRUE</code> (default): <code>x</code>  <code>Auto.linpreds=FALSE</code>: <math>\max(x-99, 0)</math> where 99 is the minimum <code>x</code> in the training data.</p> <p>Using <code>Auto.linpreds=FALSE</code> always forces a knot, even when the knot is at the minimum value of the variable. This ensures that the basis functions are always expressed as hinge functions (and will always be non-negative).</p> <p>Note that <code>Auto.linpreds</code> affects only how the model behaves <i>outside</i> the training data. Thus <code>predict.earth</code> will make the same predictions from the training data, regardless of whether the earth model was built with <code>Auto.linpreds</code> set <code>TRUE</code> or <code>FALSE</code> (up to possible differences in the size of the model caused by different GCVs because of the different forms of the terms).</p>
Force.weights	<p>Default is <code>FALSE</code>. For testing the <code>weights</code> argument. Force use of the code for handling weights in the earth code, even if <code>weights=NULL</code> or all the weights are the same. This will not necessarily generate an identical model, primarily because the non-weighted code requires some tests for numerical stability that can sometimes affect knot selection.</p>
Use.beta.cache	<p>Default is <code>TRUE</code>. Using the “beta cache” takes a little more memory but is faster (by 20% and often much more for large models). The beta cache uses <math>nk * nk * ncol(x) * sizeof(double)</math> bytes. (The beta cache is an innovation in this implementation of MARS and does not appear in Friedman’s papers. It is not related to the <code>fast.beta</code> argument. Certain regression coefficients in the forward pass can be saved and re-used, thus saving recalculation time.)</p>
Force.xtx.prune	<p>Default is <code>FALSE</code>. This argument pertains to subset evaluation in the pruning pass. By default, if <code>y</code> has a single column then earth calls the <code>leaps</code> routines; if <code>y</code> has multiple columns then earth calls <code>EvalSubsetsUsingXtx</code>. The</p>

- leaps routines are numerically more stable but do not support multiple responses (leaps is based on the QR decomposition and EvalSubsetsUsingXtx is based on the inverse of  $X'X$ ). Setting `Force.xtx.prune=TRUE` forces use of EvalSubsetsUsingXtx, even if y has a single column.
- `Get.leverages` Default is TRUE unless the model has more than 100 thousand cases. The leverages are the diagonal hat values for the linear regression of y on bx. (The leverages are needed only for certain model checks, for example when plotres is called with `versus=4`).  
Details: This argument was introduced to reduce peak memory usage. When  $n \gg p$ , memory use peaks when earth is calculating the leverages.
- `Exhaustive.tol` Default  $1e-10$ . Applies only when `pmethod="exhaustive"`. If the reciprocal of the condition number of bx is less than Exhaustive.tol, earth forces `pmethod="backward"`. See “*XHAUST returned error code -999*” in the vignette.
- ... Dots are passed on to `earth.fit`.

### Value

An S3 model of class “earth”. See [earth.object](#) for a complete description.

### Author(s)

Stephen Milborrow, derived from `mda::mars` by Trevor Hastie and Robert Tibshirani.

The approach used for GLMs was motivated by work done by Jane Elith and John Leathwick (a representative paper is given below).

The `evimp` function uses ideas from Max Kuhn’s `caret` package <https://CRAN.R-project.org/package=caret>.

Parts of Thomas Lumley’s `leaps` package have been incorporated into earth, so earth can directly access Alan Miller’s Fortran functions without going through hidden functions in the leaps package.

### References

The Wikipedia article is recommended for an elementary introduction. The primary references are the Friedman papers, but readers may find the MARS section in Hastie, Tibshirani, and Friedman a more accessible introduction. Faraway takes a hands-on approach, using the `ozone` data to compare `mda::mars` with other techniques. (If you use Faraway’s examples with earth instead of mars, use `$bx` instead of `$x`, and check out the book’s errata.) Friedman and Silverman is recommended background reading for the MARS paper. Earth’s pruning pass uses code from the `leaps` package which is based on techniques in Miller.

Faraway (2005) *Extending the Linear Model with R* <http://www.maths.bath.ac.uk/~jjf23>

Friedman (1991) *Multivariate Adaptive Regression Splines (with discussion)* Annals of Statistics 19/1, 1–141

<http://projecteuclid.org/euclid.aos/1176347963>

doi: [10.1214/aos/1176347963](https://doi.org/10.1214/aos/1176347963)

Friedman (1993) *Fast MARS* Stanford University Department of Statistics, Technical Report 110

<https://statistics.stanford.edu/research/fast-mars>

Friedman and Silverman (1989) *Flexible Parsimonious Smoothing and Additive Modeling* Technometrics, Vol. 31, No. 1. <https://www.tandfonline.com/doi/abs/10.1080/00401706.1989.10488470>

Hastie, Tibshirani, and Friedman (2009) *The Elements of Statistical Learning (2nd ed.)* <http://web.stanford.edu/~hastie/pub.htm>

Leathwick, J.R., Rowe, D., Richardson, J., Elith, J., & Hastie, T. (2005) *Using multivariate adaptive regression splines to predict the distributions of New Zealand's freshwater diadromous fish* Freshwater Biology, 50, 2034-2052 <http://web.stanford.edu/~hastie/pub.htm>, <http://www.botany.unimelb.edu.au/envisci/about/staff/elith.html>

Miller, Alan (1990, 2nd ed. 2002) *Subset Selection in Regression* <http://wp.csiro.au/alanmiller/index.html>

Wikipedia article on MARS [http://en.wikipedia.org/wiki/Multivariate\\_adaptive\\_regression\\_splines](http://en.wikipedia.org/wiki/Multivariate_adaptive_regression_splines)

### See Also

Start with [summary.earth](#), [plot.earth](#), [evimp](#), and [plotmo](#).

Please see the main package vignette “[Notes on the earth package](#)”. The vignette can also be downloaded from <http://www.milbo.org/doc/earth-notes.pdf>.

The vignette “[Variance models in earth](#)” is also included with the package. It describes how to generate prediction intervals for earth models.

### Examples

```
earth.mod <- earth(Volume ~ ., data = trees)
plotmo(earth.mod)
summary(earth.mod, digits = 2, style = "pmax")
```

---

earth.object

*An earth object*

---

### Description

The object returned by the [earth](#) function.

This is an S3 model of [class](#) "earth". It is a list with the components listed below.

*Term* refers to a term created during the forward pass (each line of the output from [format.earth](#) is a term). Term number 1 is always the intercept.

### Value

<code>rss</code>	Residual sum-of-squares (RSS) of the model (summed over all responses, if <code>y</code> has multiple columns).
<code>rsq</code>	$1 - \text{rss}/\text{tss}$ . R-Squared of the model (calculated over all responses, and calculated using the <code>weights</code> argument if it was supplied). A measure of how well the model fits the training data. Note that <code>tss</code> is the total sum-of-squares, $\text{sum}((y - \text{mean}(y))^2)$ .

`gcv` Generalized Cross Validation (GCV) of the model (summed over all responses). The GCV is calculated using the `penalty` argument. For details of the GCV calculation, see equation 30 in Friedman's MARS paper and `earth::get.gcv`.

`grsq`  $1 - \text{gcv} / \text{gcv.null}$ . An estimate of the predictive power of the model (calculated over all responses, and calculated using the `weights` argument if it was supplied). `gcv.null` is the GCV of an intercept-only model. See "Can GRSq be negative?" in the vignette.

`bx` Matrix of basis functions applied to `x`. Each column corresponds to a selected term. Each row corresponds to a row in the input matrix `x`, after taking subset. See `model.matrix.earth` for an example of `bx` handling. Example `bx`:

```

      (Intercept) h(Girth-12.9) h(12.9-Girth) h(Girth-12.9)*h(...
[1,]           1           0.0           4.6           0
[2,]           1           0.0           4.3           0
[3,]           1           0.0           4.1           0
...

```

`dirs` Matrix with one row per MARS term, and with with `ij`-th element equal to

- 0 if predictor `j` is not in term `i`
- 1 if an expression of the form  $h(\text{const} - x_j)$  is in term `i`
- 1 if an expression of the form  $h(x_j - \text{const})$  is in term `i`
- 2 if predictor `j` should enter term `i` linearly (either because specified by the `linpreds` argument or because `earth` discovered that a knot was unnecessary).

This matrix includes all terms generated by the forward pass, including those not in `selected.terms`. Note that here the terms may not all be in pairs, because although the forward pass add terms as hinged pairs (so both sides of the hinge are available as building blocks for further terms), it also deletes linearly dependent terms before handing control to the pruning pass. Example `dirs`:

```

                        Girth Height
(Intercept)             0  0 #intercept
h(12.9-Girth)           -1  0 #2nd term uses Girth
h(Girth-12.9)            1  0 #3rd term uses Girth
h(Girth-12.9)*h(Height-76) 1  1 #4th term uses Girth and Height
...

```

`cuts` Matrix with `ij`-th element equal to the cut point for predictor `j` in term `i`. This matrix includes all terms generated by the forward pass, including those not in `selected.terms`. Note for programmers: the precedent is to use `dirs` for term names etc. and to only use `cuts` where cut information needed. Example `cuts`:

```

                        Girth Height
(Intercept)             0  0 #intercept, no cuts
h(12.9-Girth)           12.9  0 #2nd term has cut at 12.9
h(Girth-12.9)           12.9  0 #3rd term has cut at 12.9
h(Girth-12.9)*h(Height-76) 12.9 76 #4th term has two cuts
...

```

prune.terms	<p>A matrix specifying which terms appear in which pruning pass subsets. The row index of prune.terms is the model size. (The model size is the number of terms in the model. The intercept is counted as a term.) Each row is a vector of term numbers for the best model of that size. An element is 0 if the term is not in the model, thus prune.terms is a lower triangular matrix, with dimensions nprune x nprune. The model selected by the pruning pass is at row number length(selected.terms). Example prune.terms:</p> <pre>[1,] 1 0 0 0 0 0 0 #intercept-only model [2,] 1 2 0 0 0 0 0 #best 2 term model uses terms 1,2 [3,] 1 2 4 0 0 0 0 #best 3 term model uses terms 1,2,4 [4,] 1 2 6 9 0 0 0 #and so on ...</pre>
selected.terms	Vector of term numbers in the selected model. Can be used as a row index vector into cuts and dirs. The first element selected.terms[1] is always 1, the intercept.
fitted.values	Fitted values. A matrix with dimensions nrow(y) x ncol(y) after factors in y have been expanded.
residuals	Residuals. A matrix with dimensions nrow(y) x ncol(y) after factors in y have been expanded.
coefficients	Regression coefficients. A matrix with dimensions length(selected.terms) x ncol(y) after factors in y have been expanded. Each column holds the least squares coefficients from regressing that column of y on bx. The first row holds the intercept coefficient(s).
rss.per.response	A vector of the RSS for each response. Length is the number of responses, i.e., ncol(y) after factors in y have been expanded. The rss component above is equal to sum(rss.per.response).
rsq.per.response	A vector of the R-Squared for each response (where R-Squared is calculated using the weights argument if it was supplied). Length is the number of responses.
gcv.per.response	A vector of the GCV for each response. Length is the number of responses. The gcv component above is equal to sum(gcv.per.response).
grsq.per.response	A vector of the GRSq for each response (calculated using the weights argument if it was supplied). Length is the number of responses.
rss.per.subset	A vector of the RSS for each model subset generated by the pruning pass. Length is nprune. For multiple responses, the RSS is summed over all responses for each subset. The rss above is rss.per.subset[length(selected.terms)]. The RSS of an intercept only-model is rss.per.subset[1].
gcv.per.subset	A vector of the GCV for each model in prune.terms. Length is nprune. For multiple responses, the GCV is summed over all responses for each subset. The gcv above is gcv.per.subset[length(selected.terms)]. The GCV of an intercept-only model is gcv.per.subset[1].

leverages	Diagonal of the hat matrix (from the linear regression of the response on <code>bx</code> ).
penalty, nk, thresh	Copies of the corresponding arguments to <code>earth</code> .
pmethod, nprune	Copies of the corresponding arguments to <code>earth</code> .
weights, wp	Copies of the corresponding arguments to <code>earth</code> .
termcond	Reason the forward pass terminated (an integer).
call	The call used to invoke <code>earth</code> .
terms	Model frame terms. This component exists only if the model was built using <code>earth.formula</code> .
namesx	Column names of <code>x</code> , generated internally by <code>earth</code> when necessary so each column of <code>x</code> has a name. Used, for example, by <code>predict.earth</code> to name columns if necessary.
namesx.org	Original column names of <code>x</code> .
levels	Levels of <code>y</code> if <code>y</code> is a <code>factor</code> , <code>c(FALSE, TRUE)</code> if <code>y</code> is <code>logical</code> , Else <code>NULL</code> .

**The following fields appear only if `earth`'s argument `keepxy` is `TRUE`.**

<code>x, y, data, subset</code>	Copies of the corresponding arguments to <code>earth</code> . Only exist if <code>keepxy=TRUE</code> .
---------------------------------	--

**The following fields appear only if `earth`'s `glm` argument is used.**

<code>glm.list</code>	List of GLM models. Each element is the value returned by <code>earth</code> 's internal call to <code>glm</code> for each response. Thus if there is a single response (or a single binomial pair, see “ <i>Binomial pairs</i> ” in the vignette) this will be a one element list and you access the GLM model with <code>earth.mod\$glm.list[[1]]</code> .
<code>glm.coefficients</code>	GLM regression coefficients. Analogous to the <code>coefficients</code> field described above but for the GLM model(s). A matrix with dimensions <code>length(selected.terms) x ncol(y)</code> after factors in <code>y</code> have been expanded. Each column holds the coefficients from the GLM regression of that column of <code>y</code> on <code>bx</code> . This duplicates, for convenience, information buried in <code>glm.list</code> .
<code>glm.stats</code>	GLM summary statistics such as <code>devratio</code> , <code>AIC</code> , and <code>iters</code> .
<code>glm.bpairs</code>	Is <code>NULL</code> unless there are paired binomial columns. Else a logical vector <code>c(TRUE, FALSE)</code> . See “ <i>Binomial pairs</i> ” in the vignette. Retained for backwards compatibility with old versions of <code>earth</code> .

**The following fields appear only if the `nfold` argument is greater than 1.**

<code>cv.list</code>	List of <code>earth</code> models, one model for each fold ( <code>ncross * nfold</code> models). The fold models have two extra fields, <code>icross</code> (an integer from 1 to <code>ncross</code> ) and <code>ifold</code> (an integer from 1 to <code>nfold</code> ). To save memory, lengthy fields in the fold models are removed unless you use <code>keepxy=TRUE</code> . The “lengthy fields” are <code>\$bx</code> , <code>\$fitted.values</code> , and <code>\$residuals</code> .
----------------------	---

cv.terms	Vector of length $ncross * nfold + 1$ . Number of MARS terms in the model generated at each cross-validation fold, with the final element being the mean of these.
cv.nvars	Vector of length $ncross * nfold + 1$ . Number of predictors in the model generated at each cross-validation fold, with the final element being the mean of these.
cv.groups	Specifies which cases went into which folds. Matrix with two columns and number of rows equal to the the number of cases $nrow(x)$ Elements of the first column specify the cross-validation number, $1:ncross$ . Elements of the second column specify the fold number, $1:nfold$ .
cv.rsq.tab	Matrix with $ncross * nfold + 1$ rows and $nresponse+1$ columns, where $nresponse$ is the number of responses, i.e., $ncol(y)$ after factors in $y$ have been expanded. The first $nresponse$ elements of a row are the <code>cv.rsq</code> 's on the out-of-fold data for each response of the model generated at that row's fold. (A <code>cv.rsq</code> is calculated from predictions on the out-of-fold data using the best model built from the in-fold data; where "best" means the model was selected using the in-fold GCV. The R-Squareds are calculated using the <code>weights</code> argument if it was supplied. The final column holds the row mean (a weighted mean if <code>wp</code> if specified)). The final row holds the column means. The values in this final row is the mean <code>cv.rsq</code> printed by <code>summary.earth</code> .

Example for a single response model (where the mean column is redundant but included for uniformity with multiple response models):

	y	mean
fold1	0.909	0.909
fold2	0.869	0.869
fold3	0.952	0.952
fold4	0.157	0.157
fold5	0.961	0.961
mean	0.769	0.769

Example for a multiple response model:

	y1	y2	y3	mean
fold1	0.915	0.951	0.944	0.937
fold2	0.962	0.970	0.970	0.968
fold3	0.914	0.940	0.942	0.932
fold4	0.907	0.929	0.925	0.920
fold5	0.947	0.987	0.979	0.971
mean	0.929	0.955	0.952	0.946

cv.class.rate.tab

Like `cv.rsq.tab` but is the classification rate at each fold i.e. the fraction of classes correctly predicted. Models with discrete response only. Calculated with `thresh=.5` for binary responses. For responses with more than two levels, the final row is the overall classification rate. The other rows are the classification rates for each level (the level versus not-the-level), which are usually higher than the overall classification rate (predicting the level versus not-the-level is easier

than correctly predicting one of many levels). The `weights` argument is ignored for all cross-validation stats except R-Squareds.

<code>cv.maxerr.tab</code>	Like <code>cv.rsq.tab</code> but is the <code>MaxErr</code> at each fold. This is the signed max absolute value at each fold. Results are aggregated for the final column and final row using the signed max absolute value. The <i>signed max absolute value</i> is defined as the maximum of the absolute difference between the predicted and observed response values, multiplied by <code>-1</code> if the sign of that difference is negative.
<code>cv.auc.tab</code>	Like <code>cv.rsq.tab</code> but is the <code>AUC</code> at each fold. Binomial models only.
<code>cv.cor.tab</code>	Like <code>cv.rsq.tab</code> but is the <code>cor</code> at each fold. Poisson models only.
<code>cv.deviance.tab</code>	Like <code>cv.rsq.tab</code> but is the <code>MeanDev</code> at each fold. Binomial models only.
<code>cv.calib.int.tab</code>	Like <code>cv.rsq.tab</code> but is the <code>CalibInt</code> at each fold. Binomial models only.
<code>cv.calib.slope.tab</code>	Like <code>cv.rsq.tab</code> but is the <code>CalibSlope</code> at each fold. Binomial models only.
<code>cv.oof.rsq.tab</code>	Generated only if <code>keepxy=TRUE</code> or <code>pmethod="cv"</code> . A matrix with <code>ncross * nfold + 1</code> rows and <code>max.terms</code> columns, Each element holds an out-of-fold <code>RSq</code> ( <code>oof.rsq</code> ), calculated from predictions from the out-of-fold observations using the model built with the in-fold data. The final row is the mean over all folds. The R-Squareds are calculated using the <code>weights</code> argument if it was supplied.
<code>cv.infold.rsq.tab</code>	Generated only if <code>keepxy=TRUE</code> . Like <code>cv.oof.rsq.tab</code> but from predictions made on the in-fold observations.
<code>cv.oof.fit.tab</code>	Generated only if the <code>varmod.method</code> argument is used. Predicted values on the out-of-fold data. Dataframe with <code>nrow(data)</code> rows and <code>ncross</code> columns.
<code>varmod</code>	<b>The following field appears only if the <code>varmod.method</code> is specified.</b> An object of class <code>"varmod"</code> . See the <a href="#">varmod</a> help page for a description. Only appears if the <code>varmod.method</code> argument is used.

**See Also**

[earth](#)

---

etitanic

*Titanic data with incomplete cases removed*

---

**Description**

Titanic data with incomplete cases, passenger names, and other details removed.

**Format**

A data frame with 1046 observations on 6 variables.



pclass	passenger class, unordered factor: 1st 2nd 3rd
survived	integer: 0 or 1
sex	unordered factor: male female
age	age in years, min 0.167 max 80.0
sibsp	number of siblings or spouses aboard, integer: 0...8
parch	number of parents or children aboard, integer: 0...6

## Source

This dataset is included in the earth package because it is a convenient vehicle for illustrating earth's GLM and factor handling.

The dataset was compiled by Frank Harrell and Robert Dawson: <http://biostat.mc.vanderbilt.edu/twiki/pub/Main/DataSets/titanic.html>

See also:

<http://biostat.mc.vanderbilt.edu/twiki/pub/Main/DataSets/titanic3info.txt>.

For this version of the Titanic data, passenger details and incomplete cases were deleted and the name changed to etitanic to minimize confusion with other versions ("e" because it is part of the earth package).

Note that survived is an integer (it should arguably be a logical).

In this data the crew are conspicuous by their absence.

Contents of etitanic:

	pclass	survived	sex	age	sibsp	parch
1	1st	1	female	29.000	0	0
2	1st	1	male	0.917	1	2
3	1st	0	female	2.000	1	2
4	1st	0	male	30.000	1	2
5	1st	0	female	25.000	1	2
...						
1309	3rd	0	male	29.000	0	0

How etitanic was built:

```
load("titanic3") # from Harrell's web site
# discard name, ticket, fare, cabin, embarked, body, home.dest
etitanic <- titanic3[,c(1,2,4,5,6,7)]
etitanic <- etitanic[!is.na(etitanic$age),]
save(etitanic, file="etitanic.rda")
```

## References

Further details and analyses of the Titanic data may be found in:

F. Harrell (2001) *Regression Modeling Strategies with Applications to Linear Models, Logistic Regression, and Survival Analysis* <http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/RmS>

**See Also**[earth](#)

---

`evimp`*Estimate variable importances in an earth object*

---

**Description**

Estimate variable importances in an [earth](#) object

**Usage**

```
evimp(object, trim=TRUE, sqrt.=TRUE)
```

**Arguments**

<code>object</code>	An <a href="#">earth</a> object.
<code>trim</code>	If TRUE (default), delete rows in the returned matrix for variables that don't appear in any subsets.
<code>sqrt.</code>	Default is TRUE, meaning take the <a href="#">sqrt</a> of the GCV and RSS importances before normalizing to 0 to 100. Taking the square root gives a better indication of relative importances because the raw importances are calculated using a sum of squares. Use FALSE to not take the square root.

**Value**

This function returns a matrix showing the relative importances of the variables in the model. There is a row for each variable. The row name is the variable name, but with `-unused` appended if the variable does not appear in the final model.

The columns of the matrix are (not all of these are printed by `print.evimp`):

- `col`: Column index of the variable in the `x` argument to `earth`.
- `used`: 1 if the variable is used in the final model, else 0. Equivalently, 0 if the row name has an `-unused` suffix.
- `nsubsets`: Variable importance using the "number of subsets" criterion. Is the number of subsets that include the variable (see "Three Criteria" in the chapter on `evimp` in the [earth](#) vignette "Notes on the earth package").
- `gcv`: Variable importance using the GCV criterion (see "Three Criteria").
- `gcv.match`: 1, except is 0 where the rank using the `gcv` criterion differs from that using the `nsubsets` criterion. In other words, there is a 0 for values that increase as you go down the `gcv` column.
- `rss`: Variable importance using the RSS criterion (see "Three Criteria").
- `rss.match`: Like `gcv.match` but for the `rss`.

The rows are sorted on the `nsubsets` criterion. This means that values in the `nsubsets` column decrease as you go down the column (more accurately, they are non-increasing). The values in the `gcv` and `rss` columns are also non-increasing, except where the `gcv` or `rss` rank differs from the `nsubsets` ranking.

### Note

There is a chapter on `evimp` in the `earth` package vignette “[Notes on the earth package](#)”.

### Acknowledgment

Thanks to Max Kuhn for the original `evimp` code and for helpful discussions.

### See Also

[earth](#), [plot.evimp](#)

### Examples

```
data(ozone1)
earth.mod <- earth(O3 ~ ., data=ozone1, degree=2)
ev <- evimp(earth.mod, trim=FALSE)
plot(ev)
print(ev)
```

---

expand.bpairs

*Expand binomial-pair data from short to long form*

---

### Description

Expand binomial-pair data from “short” to “long” form.

The short form specifies the response with two columns giving the numbers of successes and failures. Example short form:

```
survived died dose  sex
      3    0  10  male
      2    1  10 female
      1    2  20  male
      1    2  20 female
```

The long form specifies the response as single column of TRUEs and FALSEs. For example, the long form of the above data (spaces and comments added):

```
survived dose  sex
  TRUE  10  male  # row 1 of short data: 0 died, 3 survived
  TRUE  10  male
  TRUE  10  male

  FALSE 10 female # row 2 of short data: 1 died, 2 survived
```

```

TRUE  10 female
TRUE  10 female

FALSE  20  male    # row 3 of short data: 2 died, 1 survived
FALSE  20  male
TRUE   20  male

FALSE  20 female   # row 4 of short data: 2 died, 1 survived
FALSE  20 female
TRUE   20 female

```

In this example the total number of survived and died for each row in the short data is the same, but in general that need not be true.

### Usage

```

## S3 method for class 'formula'
expand.bpairs(formula = stop("no 'formula' argument"), data = NULL, sort = FALSE, ...)

## Default S3 method:
expand.bpairs(data = stop("no 'data' argument"), y = NULL, sort = FALSE, ...)

```

### Arguments

formula	Model formula such as <code>survived + died ~ dose + temp</code> .
data	Matrix or dataframe containing the data.
y	Model response. One of: <ul style="list-style-type: none"> <li>o Two column matrix or dataframe of binomial pairs e.g. <code>cbind(survived, died=20-survived)</code></li> <li>o Two-element numeric vector specifying the response columns in data e.g. <code>c(1,2)</code></li> <li>o Two-element character vector specifying the response column names in data e.g. <code>c("survived", "died")</code>. The full names must be used (partial matching isn't supported).</li> </ul>
sort	Default FALSE. Use TRUE to sort the rows of the long data so it is returned in canonical form, independent of the row order of the short data. The long data is sorted on predictor values; predictors on the left take precedence in the sort order.
...	Unused, but provided for generic/method consistency.

### Value

A dataframe of the data in the long form, with expanded binomial pairs. The first column of the data will be the response column (a column of TRUEs and FALSEs).

Additionally, the returned value has two attached attributes:

`bpairs.index` A vector of row indices into the returned data. Can be used to reconstruct the short data from the long data (although this package does not yet provide a function to do so).

`ynames` Column names of the original response (a two-element character vector).

**Examples**

```

survived <- c(3,2,1,1) # short data for demo (too short to build a real model)
died      <- c(0,1,2,2)
dose     <- c(10,10,20,20)
sex      <- factor(c("male", "female", "male", "female"))

short.data <- data.frame(survived, died, dose, sex)

expand.bpairs(survived + died ~ ., short.data) # returns long form of the data

# expand.bpairs(data=short.data, y=cbind(survived, died)) # equivalent
# expand.bpairs(short.data, c(1,2))                    # equivalent
# expand.bpairs(short.data, c("survived", "died"))     # equivalent

# For example models, see the earth vignette
# section "Short versus long binomial data".

```

format.earth

*Format earth objects***Description**

Return a string representing an [earth](#) expression ([summary.earth](#) calls this function internally to display the terms of the earth model).

**Usage**

```

## S3 method for class 'earth'
format(x = stop("no 'x' argument"),
       style = "h", decomp = "anova", digits = getOption("digits"),
       use.names = TRUE, colon.char = ":", ...)

```

**Arguments**

x	An <a href="#">earth</a> object. This is the only required argument.
style	Formatting style. One of " h " (default) more compact " pmax " for those who prefer it " max " is the same as " pmax " but prints max rather than pmax " C " C style expression with zero based indexing " bf " basis function format
decomp	One of " anova " (default) order the terms using the "anova decomposition", i.e., in increasing order of interaction " none " order the terms as created during the earth forward pass.
digits	Number of significant digits. The default is <code>getOption(digits)</code> .

use.names	One of TRUE (default), use variable names if available. FALSE use names of the form x[, 1].
colon.char	Change colons in the returned string to colon.char. Default is ":" (no change). Specifying colon.char="*" can be useful in some contexts to change names of the form x1:x2 to x1*x2.
...	Unused, but provided for generic/method consistency.

### Value

A character representation of the earth model.

If there are multiple responses, format.earth will return multiple strings.

If there are embedded GLM model(s), the strings for the GLM model(s) come after the strings for the standard earth model(s).

### Note

The FAQ section in the package vignette gives precise details of the "anova" ordering.

Using format.earth, perhaps after hand editing the returned string, you can create an alternative to predict.earth. For example:

```
as.func <- function(object, digits = 8, use.names = FALSE, ...)
  eval(parse(text=paste(
    "function(x){\n",
    "if(is.vector(x))\n",
    "  x <- matrix(x, nrow = 1, ncol = length(x))\n",
    "with(as.data.frame(x),\n",
    "format(object, digits = digits, use.names = use.names, style = "pmax", ...),
    ")\n",
    "}\n", sep = "")))

earth.mod <- earth(Volume ~ ., data = trees)
my.func <- as.func(earth.mod, use.names = FALSE)
my.func(c(10,80)) # returns 16.84
predict(earth.mod, c(10,80)) # returns 16.84
```

Note that with pmax the R expression generated by format.earth can handle multiple cases. Thus the expression is consistent with the way predict functions usually work in R — we can give predict multiple cases (i.e., multiple rows in the input matrix) and it will return a vector of predicted values.

The earth package also provides a function format.lm. It has arguments as follows  
format.lm(x, digits=getOption("digits"), use.names=TRUE, colon.char=":")  
(Strictly speaking, format.lm doesn't belong in the earth package.) Example:

```
lm.mod <- lm(Volume ~ Height*Girth, data = trees)
cat(format(lm.mod, colon.char="*"))

# yields:
# 69.4
# - 1.30 * Height
# - 5.86 * Girth
# + 0.135 * Height*Girth
```

**See Also**

[summary.earth](#), [pmax](#),

**Examples**

```
earth.mod <- earth(Volume ~ ., data = trees)
cat(format(earth.mod))

# yields:
# 37.9
# - 3.92 * h(16-Girth)
# + 7.4 * h(Girth-16)
# + 0.484 * h(Height-75)

cat(format(earth.mod, style="pmax"))

# yields:
# 37.9
# - 3.92 * pmax(0, 16 - Girth)
# + 7.4 * pmax(0, Girth - 16)
# + 0.484 * pmax(0, Height - 75)

cat(format(earth.mod, style="C"))

# yields (note zero based indexing):
# 37.927
# - 3.9187 * max(0, 16 - x[0])
# + 7.4011 * max(0, x[0] - 16)
# + 0.48411 * max(0, x[1] - 75)

cat(format(earth.mod, style="bf"))

# yields:
# 37.9
# - 3.92 * bf1
# + 7.4 * bf2
# + 0.484 * bf3
#
# bf1 h(16-Girth)
# bf2 h(Girth-16)
# bf3 h(Height-75)
```

---

 mars.to.earth

 Convert a mars object from the mda package to an earth object
 

---

### Description

Convert a [mars](#) object from the mda package to an [earth](#) object

### Usage

```
mars.to.earth(object, trace=TRUE)
```

### Arguments

object	A mars object, created using <a href="#">mars</a> in the mda package.
trace	If TRUE (default) print a summary of the conversion.

### Value

The value is the same format as that returned by [earth](#) but with skeletal versions of `rss.per.subset`, `gcv.per.subset`, and `prune.terms`.

You can fully initialize these components by calling [update.earth](#) after `mars.to.earth`, but if you do this `selected.terms` may change. However with `pmethod="backward"` a change is unlikely — `selected.terms` would change only if GCVs are so close that numerical errors have an effect.

### Note

#### Differences between mars and earth objects

Perhaps the most notable difference between mars and earth objects is that mars returns the MARS basis matrix in a field called "x" whereas earth returns "bx" with only the selected terms. Also, earth returns "dirs" rather than "factors", and in earth this matrix can have entries of value 2 for linear predictors.

For details of other differences between mars and earth objects, see the comments in the source code of `mars.to.earth`.

#### Weights

The `w` argument is silently ignored by `mars`.

`mars` normalizes `wp` to (euclidean) length 1; `earth` normalizes `wp` to length equal to the number of responses, i.e., the number of columns in `y`. This change was made so an all ones `wp` (or in fact any all constant `wp`) is equivalent to using `no wp`.

If the original call to `mars` used the `wp` argument, `mars.to.earth` will run [update.earth](#) to force consistency. This could modify the model, so a warning is issued.

### See Also

[earth](#), [mars](#)



**Examples**

```

if(require(mda)) {
  mars.mod <- mars(trees[,-3], trees[,3])
  earth.mod <- mars.to.earth(mars.mod)
  # the standard earth functions can now be used
  # note the reconstructed call in the summary
  summary(earth.mod, digits = 2)
}

```

---

model.matrix.earth      *Get the earth basis matrix*

---

**Description**

Get the basis matrix of an [earth](#) model.

**Usage**

```

## S3 method for class 'earth'
model.matrix(object = stop("no 'object' argument"),
  x = NULL, subset = NULL, which.terms = NULL,
  trace = 0,
  ...,
  Env = parent.frame(),
  Callers.name = "model.matrix.earth")

```

**Arguments**

object	An <a href="#">earth</a> model. This is the only required argument.
x	Default is NULL, meaning use the original data used to build the earth model (after taking the original subset, if any). Else x can be a data frame, a matrix, or a vector with length equal to a multiple of the number of columns of the original input matrix x. (There is some leniency here. For example, column names aren't necessary if x has the same number of predictors originally used to build the earth model.)
subset	Which rows to use in x. Default is NULL, meaning use all of x.
which.terms	Which terms to use. Default is NULL, meaning all terms in the earth model (i.e. the terms in object\$selected.terms).
trace	Default 0. Set to non-zero to see which data model.matrix.earth is using.
...	Unused, but provided for generic/method consistency.
Env	For internal use.
Callers.name	For internal use (used by earth in trace messages).

**Value**

A basis matrix `bx` of the same form returned by `earth`. The format of `bx` is described in `earth.object`.

If `x`, `subset`, and `which.terms` are all `NULL` (the default), this function returns the model's `bx`. In this case, it is perhaps easier to simply use `object$bx`.

The matrix `bx` can be used as the input matrix to `lm` or `glm`, as shown below in the example. In fact, that is what `earth` does internally after the pruning pass — it calls `lm.fit`, and additionally `glm` if `earth's glm` argument is used.

**See Also**

[earth](#)

**Examples**

```
data(trees)
earth.mod <- earth(Volume ~ ., data = trees)
summary(earth.mod, decomp = "none") # "none" to print terms in same order as lm.mod below

bx <- model.matrix(earth.mod)      # equivalent to bx <- earth.mod$bx
lm.mod <- lm(trees$Volume ~ bx[, -1]) # -1 to drop intercept
summary(lm.mod)                   # yields same coeffs as above summary
                                   # displayed t values are not meaningful
```

---

ozone1

*Ozone readings in Los Angeles with incomplete cases removed*

---

**Description**

Ozone readings in Los Angeles, with incomplete cases removed.

**Format**

A data frame with 330 observations on 10 variables.

o3	daily maximum of the hourly average ozone concentrations in Upland, CA
vh	500 millibar pressure height, measured at the Vandenberg air force base
wind	wind speed in mph at LAX airport
humidity	humidity in percent at LAX
temp	Sandburg Air Force Base temperature in degrees Fahrenheit
ibh	temperature inversion base height in feet
dpg	pressure gradient from LAX to Daggert in mm Hg
ibt	inversion base temperature at LAX in degrees Fahrenheit
vis	visibility at LAX in miles
doy	day of the year

**Source**

This dataset was copied from `library(faraway)` and the name changed to `ozone1` to prevent a name clash. The data were originally made available by Leo Breiman who was a consultant on a project where the data were generated. Example analyses using these data may be found in Faraway and in Hastie and Tibshirani.

```
> ozone1
      O3   vh wind humidity temp  ibh dpg ibt vis doy
1     3 5710   4     28   40 2693 -25  87 250  33
2     5 5700   3     37   45  590 -24 128 100  34
3     5 5760   3     51   54 1450  25 139  60  35
...
330  1 5550   4     85   39 5000   8  44 100 390
```

**References**

Faraway (2005) *Extending the Linear Model with R* <http://www.maths.bath.ac.uk/~jjf23>

Hastie and Tibshirani (1990) *Generalized Additive Models* <http://web.stanford.edu/~hastie/pub.htm>

**See Also**

[earth](#)

[airquality](#) a different set of ozone data

---

plot.earth

*Plot an earth object*

---

**Description**

Plot an [earth](#) object. By default the plot shows model selection, cumulative distribution of the residuals, residuals versus fitted values, and the residual QQ plot.

This function calls [plotres](#) internally. The first arguments are identical to `plotres`.

**Usage**

```
## S3 method for class 'earth'
plot(x = stop("no 'x' argument"),

     # the following are identical to plotres arguments

     which = 1:4, info = FALSE, versus = 1, standardize = FALSE, delever = FALSE,
     level = 0, id.n = 3, labels.id = NULL, smooth.col = 2, grid.col = 0,
     jitter = 0, do.par = NULL, caption = NULL,
     trace = 0, npoints = 3000, center = TRUE, type = NULL, nresponse = NA,
```

```

# the following are earth specific

col.cv = "lightblue", col.grsq = 1, col.rsq = 2, col.infold.rsq = 0,
col.mean.infold.rsq = 0, col.mean.oof.rsq = "palevioletred",
col.npreds = if(is.null(object$cv.oof.rsq.tab)) 1 else 0, col.oof.labs = 0,
col.oof.rsq = "mistyrose2", col.oof.vline = col.mean.oof.rsq,
col.pch.cv.rsq = 0, col.pch.max.oof.rsq = 0, col.vline = col.grsq,
col.vseg = 0, lty.grsq = 1, lty.npreds = 2, lty.rsq = 5, lty.vline = "12",
legend.pos = NULL, ...)

earth_plotmodsel( # for internal use by plotres
  x, col.rsq = 2, col.grsq = 1, col.infold.rsq = 0,
  col.mean.infold.rsq = 0, col.mean.oof.rsq = "palevioletred",
  col.npreds = NULL, col.oof.labs = 0, col.oof.rsq = "mistyrose2",
  col.oof.vline = col.mean.oof.rsq, col.pch.cv.rsq = 0,
  col.pch.max.oof.rsq = 0, col.vline = col.grsq, col.vseg = 0,
  lty.grsq = 1, lty.npreds = 2, lty.rsq = 5, lty.vline = "12",
  legend.pos=NULL, add = FALSE, jitter = 0,
  max.nterms = length(object$rss.per.subset),
  max.npreds=max(1,get.nused.preds.per.subset(object$dirs,object$prune.terms)),
  ...)

```

## Arguments

**x** An [earth](#) object. This is the only required argument. (It is called "x" for consistency with the generic [plot](#).)

**which, info, versus**

These arguments are identical to [plotres](#). Please see the help page for [plotres](#).

**standardize, delever, level**

**id.n, labels.id, smooth.col**

**grid.col, jitter**

**do.par, caption, trace**

**npoints, center**

**type, nresponse**

**col.cv** Default "lightblue". Color of cross validation line in the residuals plot. This is the residual of the mean out-fold-predicted value.

**The following arguments are for the model selection plot.**

col.grsq	Default 1. Color of GRSq line in the Model Selection plot. Use 0 for no GRSq line.
col.rsq	Default 2. Color of the RSq line in the Model Selection plot. Use 0 for no RSq line.
col.infold.rsq	Color of in-fold RSq lines for each fold in the Model Selection plot. Applies only if nfold and keepxy were used in the original call to earth. Default is 0, lines not plotted.
col.mean.infold.rsq	Color of mean in-fold RSq for each number of terms in the Model Selection plot. Default is 0, line not plotted. Applies only if nfold and keepxy were used in the original call to earth.
col.mean.oof.rsq	Default "palevioletred". Color of mean out-of-fold RSq for each number of terms in the Model Selection plot. Applies only if nfold and keepxy were used in the original call to earth. Use 0 to not plot this line.
col.npreds	Color of the "number of predictors" plot in the Model Selection plot. The default displays the number of predictors unless the oof.rsq's are displayed. Use 0 for no "number of predictors" plot.
col.oof.labs	Color of fold number labels on the oof.rsq lines. Default is 0, no labels.
col.oof.rsq	Color of out-of-fold RSq lines for each fold in the Model Selection plot. Applies only if nfold and keepxy were used in the original call to earth. Default is "mistyrose2", a pale pink. Use 0 to not plot these lines. May be a vector of colors, which will be recycled if necessary.
col.oof.vline	Color of vertical line at the maximum oof.rsq in the Model Selection plot. Default is col.mean.oof.rsq.
col.pch.cv.rsq	Color of point plotted on the oof.rsq line to indicate the cv.rsq. for that fold (i.e., it is plotted at the number of terms selected by the in-fold GCV). Default is 0, point not plotted.
col.pch.max.oof.rsq	Color of point plotted on the oof.rsq line to indicate the maximum oof.rsq for that fold. Default is 0, point not plotted.
col.vline	Color of the vertical line at selected model in the Model Selection plot. Default is col.grsq. This will be at the maximum GRSq unless pmethod="none". Use 0 for no vertical line.
col.vseg	Default is 0. Color of triangular marker at top of vertical line for best GRSq.
lty.grsq	Line type of GRSq line in the Model Selection plot. Default is 1
lty.npreds	Line type of the "number of predictors" plot in the Model Selection plot. Default is 2.
lty.rsq	Line type of RSq line in the Model Selection plot. Default is 5.
lty.vline	Line type of vertical line at selected model in the Model Selection plot. Default is "12".
legend.pos	Position of the legend in the Model Selection plot. Default is NULL meaning automatic. Use legend.pos=NA or 0 for no legend. Can be something like legend.pos="topleft" or legend.pos=c(6, .75).

```
add, max.terms, max.npreds
      earth_plotmodsel arguments for internal use by plotres.
```

```
...      Please see plotres for the details on the dots arguments.
      The ylim argument is treated specially in the model selection plot: ymin equal
      to -1 means use the smallest GRSq or RSq value, excluding the intercept, and
      ymax equal -1 means use the largest GRSq or RSq value.
```

### Note

For details on interpreting the graphs, please see the earth package vignettes “[Notes on the earth package](#)” and “[Variance models in earth](#)”.

Note that cross-validation data will not be displayed unless both `ifold` and `keepxy` were used in the original call to `earth`.

To remove the Number of used predictors from the Model Selection graph (to reduce clutter), use `col.npreds=0`.

`earth_plotmodsel` is provided for use by [plotres](#).

### See Also

[earth](#), [plot.earth.models](#), [plotd](#), [plotmo](#)

### Examples

```
data(ozone1)
earth.mod <- earth(O3 ~ ., data = ozone1, degree = 2)
plot(earth.mod)
```

---

`plot.earth.models`      *Compare earth models by plotting them.*

---

### Description

Compare [earth](#) models by plotting them.

### Usage

```
## S3 method for class 'earth.models'
plot(x = stop("no 'x' argument"), which = c(1:2),
     caption = "", jitter = 0,
     col.grsq = discrete.plot.cols(length(objects)), lty.grsq = 1,
     col.rsq = 0, lty.rsq = 5,
     col.vline = col.grsq, lty.vline = "12",
     col.npreds = 0, lty.npreds = 2,
     legend.text = NULL, do.par = NULL, trace = 0,
     ...)
```

**Arguments**

x	A list of one or more <a href="#">earth</a> objects, or a single <a href="#">earth</a> object. This is the only required argument. (This argument is called 'x' for consistency with the generic <a href="#">plot</a> .)
which	Which plots to plot: 1 model, 2 cumulative distribution of residuals. Default is 1:2, meaning both.
caption	Overall caption. Values: "string" string "" (default) no caption NULL generate a caption from the \$call component of the earth objects.
jitter	Jitter applied to GRSq and RSq values to minimize over-plotting. Default is 0, meaning no jitter. A typical useful value is 0.01.
	<i>For the col arguments below, 0 means do not plot the corresponding graph element. You can use vectors of colors.</i>
col.grsq	Vector of colors for the GRSq plot. The default is <code>discrete.plot.cols(length(x))</code> which is vector of distinguishable colors, the first three of which are also distinguishable on a monochrome printer. You can examine the colors using <code>earth::discrete.plot.cols()</code> .
lty.grsq	Line type for the GRSq plot. Default is 1.
col.rsq	Vector of colors for the RSq plot. Default is 0, meaning no RSq plot.
lty.rsq	Line type for the RSq plot. Default is 5.
col.vline	A vertical line is drawn for each object to show which model size was chosen for that object. The color of the line is <code>col.vline</code> . Default is <code>col.grsq</code> .
lty.vline	Line type of vertical lines (a vertical line is drawn to show the selected model for each object). Can be a vector. Default is 3.
col.npreds	Vector of colors for the "number of predictors" plot within the model selection plot. Default is 0, meaning no "number of predictors" plot. The special value NULL means borrow <code>col.grsq</code> (or <code>col.rsq</code> if <code>col.grsq</code> is NULL).
lty.npreds	Line type of the "number of predictors" plot (in the Model Selection plot). Default is 2.
legend.text, do.par, trace	Please see <a href="#">plotres</a>
...	Please see <a href="#">plotres</a>

**Note**

This function ignores GLM and cross-validation components of the earth model, if any.

**See Also**

[earth](#), [plot.earth](#), [plot.earth.models](#), [plotd](#), [plotmo](#)

**Examples**

```
data(ozone1)
a1 <- earth(O3 ~ ., data = ozone1, degree = 2)
a2 <- earth(O3 ~ .-wind, data = ozone1, degree = 2)
a3 <- earth(O3 ~ .-humidity, data = ozone1, degree = 2)
plot.earth.models(list(a1,a2,a3), ylim=c(.65,.85))
```

---

plot.evimp

*Plot an evimp object (created by the evimp function)*


---

**Description**

Plot an [evimp](#) object.

**Usage**

```
## S3 method for class 'evimp'
plot(x = stop("no 'x' argument"),
     cex.var = 1,
     type.nsubsets = "l", col.nsubsets = "black", lty.nsubsets = 1,
     type.gcv = "l", col.gcv = 2, lty.gcv = 1,
     type.rss = "l", col.rss = "gray60", lty.rss = 1,
     cex.legend = 1, x.legend = nrow(x), y.legend = x[1,"nsubsets"],
     rh.col = 1, do.par = TRUE, ...)
```

**Arguments**

x	An <a href="#">evimp</a> object.
cex.var	cex for variable names. Default is 1. Make smaller (say 0.8) if you have lots of variables.
type.nsubsets	Plot type for nsubsets graph. Default is "l". Use "n" for none, "b" looks good too.
col.nsubsets	Color of nsubsets line. Default is "black".
lty.nsubsets	Line type of nsubsets line. Default is 1.
type.gcv, col.gcv, lty.gcv	As above but for the gcv plot
type.rss, col.rss, lty.rss	As above but for the rss plot
cex.legend	cex for legend strings. Default is 1. Make smaller (say 0.8) if you want a smaller legend.
x.legend	x position of legend. Use 0 for no legend.
y.legend	y position of legend.
rh.col	Color of right hand axis label. Use rh.col=0 for no label, a workaround for when the label is mispositioned.



do.par            Call `par()` for global settings as appropriate. Default is TRUE, which sets `oma=c(bottom.margin,0,0,3)`, `cex=cex.var`. Set to FALSE if you want to append figures to an existing plot.

...                Extra arguments passed to plotting functions.

**See Also**

[earth](#), [evimp](#), [plot.earth.models](#), [plotmo](#)

**Examples**

```
data(ozone1)
earth.mod <- earth(O3 ~ ., data=ozone1, degree=2)
ev <- evimp(earth.mod)
plot(ev)
print(ev)
```

---

plot.varmod	<i>Plot a varmod object (created by calling earth with the varmod argument)</i>
-------------	---

---

**Description**

Plot a variance model (a varmod object).

Typically you call this function for a variance model embedded in an earth model.

**Usage**

```
## S3 method for class 'varmod'
plot(x = stop("no 'x' argument"), which = 1:4,
     do.par = NULL, info=FALSE,
     cex = NULL, caption = NULL,
     line.col = 2, min.sd.col = line.col,
     trace = 0, ...)
```

**Arguments**

`x`                A varmod object. Typically this is embedded in a parent earth object, and so you invoke this function with `plot(earth.mod$varmod)`. The `varmod.method` argument must have been specified when building the earth model.

`which`            Which plots to plot. Default is 1:4 meaning all. The term *parent* below refers to the earth model in which the varmod is embedded.

- 1) fitted vs parent fitted
- 2) fitted vs parent first predictor
- 3) residuals vs fitted
- 4) model selection graph (only when `varmod.method="earth"` or `"x.earth"`).

<code>do.par</code>	Please see <a href="#">plotres</a>
<code>info</code>	Plot some additional information, including lowess fits in the first two plots.
<code>cex</code>	Character expansion.
<code>caption</code>	Default is NULL, meaning automatically generate an overall caption.
<code>line.col</code>	Color of lines in the plots. Default is red.
<code>min.sd.col</code>	Color of the min.sd dotted horizontal line. Default is line.col. Use 0 to not plot this line.
<code>trace,...</code>	Similar to <a href="#">plotres</a>

**Note**

The horizontal red dotted line in the first two plots shows the value of min.sd. See [earth](#)'s `varmod.clamp` argument.

**See Also**

[varmod](#)

**Examples**

```
data(ozone1)

set.seed(1) # optional, for cross validation reproducibility

# note: should really use ncross=30 below but for a quick demo we don't

earth.mod <- earth(O3~temp, data=ozone1, nfold=10, ncross=3, varmod.method="lm")

plot(earth.mod$varmod) # plot the embedded variance model (this calls plot.varmod)
```

---

plotd

*Plot the distribution of predictions for each class*

---

**Description**

Plot the distribution of the predicted values for each class. Can be used for [earth](#) models, but also for models built by [lm](#), [glm](#), [lda](#), etc.

**Usage**

```
plotd(object, hist = FALSE, type = NULL, nresponse = NULL, dichot = FALSE,
      trace = FALSE, xlim = NULL, ylim = NULL, jitter = FALSE, main=NULL,
      xlab = "Predicted Value", ylab = if(hist) "Count" else "Density",
      lty = 1, col = c("gray70", 1, "lightblue", "brown", "pink", 2, 3, 4),
      fill = if(hist) col[1] else 0,
      breaks = "Sturges", labels = FALSE,
```

```
kernel = "gaussian", adjust = 1, zero.line = FALSE,
legend = TRUE, legend.names = NULL, legend.pos = NULL,
cex.legend = .8, legend.bg = "white", legend.extra = FALSE,
vline.col = 0, vline.thresh = .5, vline.lty = 1, vline.lwd = 1,
err.thresh = vline.thresh, err.col = 0, err.border = 0, err.lwd = 1,
xaxt = "s", yaxt = "s", xaxis.cex = 1, sd.thresh = 0.01, ...)
```

## Arguments

	To start off, look at the arguments object, hist, type. For predict methods with multiple column responses, see the nresponse argument. For factor responses with more than two levels, see the dichot argument. Model object. Typically a model which predicts a class or a class discriminant.
object	FALSE (default) to call <code>density</code> internally. TRUE to call <code>hist</code> internally.
type	Type parameter passed to <code>predict</code> . For allowed values see the predict method for your object (such as <code>predict.earth</code> ). By default, plotd tries to automatically select a suitable value for the model in question. (This is "response" for all objects except rpart models, where "vector" is used. The choices will often be inappropriate.) Typically you would set hist=TRUE when type="class".
nresponse	Which column to use when predict returns multiple columns. This can be a column index or column name (which may be abbreviated, partial matching is used). The default is NULL, meaning use all columns of the predicted response.
dichot	Dichotomise the predicted response. This argument is ignored except for models where the observed response is a factor with more than two levels and the predicted response is a numeric vector. The default FALSE separates the response into a group for each factor. With dichot=TRUE the response is separated into just two groups: the first level of the factor versus the remaining levels.
trace	Default FALSE. Use TRUE or 1 to trace plotd — useful to see how plotd partitions the predicted response into classes. Use 2 for more details.
xlim	Limits of the x axis. The default NULL means determine these limits automatically, else specify c(xmin, xmax).
ylim	Limits of the y axis. The default NULL means determine these limits automatically, else specify c(ymin, ymax).
jitter	Jitter the histograms or densities horizontally to minimize overplotting. Default FALSE. Specify TRUE to automatically calculate the jitter, else specify a numeric jitter value.
main	Main title. Values: "string" string "" no title NULL (default) generate a title from the call.
xlab	x axis label. Default is "Predicted Value".
ylab	y axis label. Default is if(hist) "Count" else "Density".

lty	Per class line types for the plotted lines. Default is 1 (which gets recycled for all lines).
col	Per class line colors. The first few colors of the default are intended to be easily distinguishable on both color displays and monochrome printers.
fill	Fill color for the plot for the first class. For hist=FALSE, the default is 0, i.e., no fill. For hist=TRUE, the default is the first element in the col argument.
breaks	Passed to hist. Only used if hist=TRUE. Default is "Sturges". When type="class", setting breaks to a low number can be used to widen the histogram bars
labels	TRUE to draw counts on the hist plot. Only used if hist=TRUE. Default is FALSE.
kernel	Passed to density. Only used if hist=FALSE. Default is "gaussian".
adjust	Passed to density. Only used if hist=FALSE. Default is 1.
zero.line	Passed to plot.density. Only used if hist=FALSE. Default is FALSE.
legend	TRUE (default) to draw a legend, else FALSE.
legend.names	Class names in legend. The default NULL means determine these automatically.
legend.pos	Position of the legend. The default NULL means position the legend automatically, else specify c(x,y).
cex.legend	cex for legend. Default is .8.
legend.bg	bg color for legend. Default is "white".
legend.extra	Show (in the legend) the number of occurrences of each class. Default is FALSE.
vline.thresh	Horizontal position of optional vertical line. Default is 0.5. The vertical line is intended to indicate class separation. If you use this, don't forget to set vline.col.
vline.col	Color of vertical line. Default is 0, meaning no vertical line.
vline.lty	Line type of vertical line. Default is 1.
vline.lwd	Line width of vertical line. Default is 1.
err.thresh	x axis value specifying the error shading threshold. See err.col. Default is vline.thresh.
err.col	Specify up to three colors to shade the "error areas" of the density plot. The default is 0, meaning no error shading. This argument is ignored unless hist=FALSE. If there are more than two classes, err.col uses only the first two. This argument is best explained by running an example:  <pre>data(etitanic) earth.mod &lt;- earth(survived ~ ., data=etitanic) plotd(earth.mod, vline.col=1, err.col=c(2,3,4))</pre>
	The three areas are (i) the error area to the left of the threshold, (ii) the error area to the right of the threshold, and, (iii) the reducible error area. If less than three values are specified, plotd re-uses values in a sensible manner. Use values of 0 to skip areas. Disjoint regions are not handled well by the current implementation.
err.border	Borders around the error shading. Default is 0, meaning no borders, else specify up to three colors.

<code>err.lwd</code>	Line widths of borders of the error shading. Default is 1, else specify up to three line widths.
<code>xaxt</code>	Default is "s". Use <code>xaxt="n"</code> for no x axis.
<code>yaxt</code>	Default is "s". Use <code>yaxt="n"</code> for no y axis.
<code>xaxis.cex</code>	Only used if <code>hist=TRUE</code> and <code>type="class"</code> . Specify size of class labels drawn on the x axis. Default is 1.
<code>sd.thresh</code>	Minimum acceptable standard deviation for a density. Default is 0.01. Densities with a standard deviation less than <code>sd.thresh</code> will not be plotted (a warning will be issued and the legend will say "not plotted").
<code>...</code>	Extra arguments passed to the predict method for the object.

### Note

This function calls `predict` with the data originally used to build the model, and with the type specified above. It then separates the predicted values into classes, where the class for each predicted value is determined by the class of the observed response. Finally, it calls `density` (or `hist` if `hist=TRUE`) for each class-specific set of values, and plots the results.

This function estimates distributions with the `density` and `hist` functions, and also calls `plot.density` and `plot.histogram`. For an overview see Venables and Ripley MASS section 5.6.

### Partitioning the response into classes

Considerable effort is made to partition the predicted response into classes in a sensible way. This is not always possible for multiple column responses and the `nresponse` argument should be used where necessary. The partitioning details depend on the types and numbers of columns in the observed and predicted responses. These in turn depend on the model object and the type argument.

Use the `trace` argument to see how `plotd` partitions the response for your model.

### Degenerate densities

A message such as

```
Warning: standard deviation of "male" density is 0, density is degenerate?
means that the density for that class will not be plotted (the legend will say "not plotted").
```

Set `sd.thresh=0` to get rid of this check, but be aware that histograms (and sometimes x axis labels) for degenerate densities will be misleading.

### Using plotd for various models

This function is included in the `earth` package but can also be used with other models.

Example with `glm`:

```
library(earth); data(etitanic)
glm.model <- glm(sex ~ ., data=etitanic, family=binomial)
plotd(glm.model)
```

Example with `lm`:

```
library(earth); data(etitanic)
lm.model <- lm(as.numeric(sex) ~ ., data=etitanic)
plotd(lm.model)
```

### Using plotd with lda or qda

The plotd function has special handling for `lda` (and `qda`) objects. For such objects, the `type` argument can take one of the following values:

"response" (default) linear discriminant  
 "ld" same as "response"  
 "class" predicted classes  
 "posterior" posterior probabilities

Example:

```
library(MASS); library(earth); data(etitanic)
lda.model <- lda(sex ~ ., data=etitanic)
plotd(lda.model) # linear discriminant by default
plotd(lda.model, type="class", hist=TRUE, labels=TRUE)
```

This handling of `type` is handled internally by plotd and `type` is not passed to `predict.lda` (`type` is used merely to select fields in the list returned by `predict.lda`). The `type` names can be abbreviated down to a single character.

For objects created with `lda.matrix` (as opposed to `lda.formula`), plotd blindly assumes that the `grouping` argument was the second argument.

plotd does not yet support objects created with `lda.data.frame`.

For `lda` responses with more than two factor levels, use the `nresponse` argument to select a column in the predicted response. Thus with the default `type=NULL`, (which gets automatically converted by plotd to `type="response"`), use `nresponse=1` to select just the first linear discriminant. The default `nresponse=NULL` selects all columns, which is typically not what you want for `lda` models.

Example:

```
library(MASS); library(earth);
set.seed(1) # optional, for reproducibility
example(lda) # creates a model called "z"
plot(z, dimen=1) # invokes plot.lda from the MASS package
plotd(z, nresponse=1, hist=1) # equivalent using plotd
# nresponse=1 selects first linear discr.
```

The `dichot=TRUE` argument is also useful for `lda` responses with more than two factor levels.

### TODO

Handle degenerate densities in a more useful way.

Add `freq` argument for `hist`.

### See Also

[density](#), [plot.density](#)  
[hist](#), [plot.histogram](#)  
[earth](#), [plot.earth](#)

**Examples**

```

if (require(earth)) {
  old.par <- par(no.readonly=TRUE);
  par(mfrow=c(2,2), mar=c(4, 3, 1.7, 0.5), mgp=c(1.6, 0.6, 0), par(cex = 0.8))
  data(etitanic)
  mod <- earth(survived ~ ., data=etitanic, degree=2, glm=list(family=binomial))

  plotd(mod)

  plotd(mod, hist=TRUE, legend.pos=c(.25,220))

  plotd(mod, hist=TRUE, type="class", labels=TRUE, xlab="", xaxis.cex=.8)

  par(old.par)
}

```

---

predict.earth

*Predict with an earth model*


---

**Description**

Predict with an [earth](#) model.

**Usage**

```

## S3 method for class 'earth'
predict(object = stop("no 'object' argument"), newdata = NULL,
        type = c("link", "response", "earth", "class", "terms"),
        interval = "none", level = .95,
        thresh = .5, trace = FALSE, ...)

```

**Arguments**

object	An <a href="#">earth</a> object. This is the only required argument.
newdata	Make predictions using newdata, which can be a data frame, a matrix, or a vector with length equal to a multiple of the number of columns of the original input matrix x. Default is NULL, meaning return values predicted from the training set. NAs are allowed in newdata (and the predicted value will be NA unless the NAs are in variables that are unused in the earth model).
type	Type of prediction. One of "link" (default), "response", "earth", "class", or "terms". See the <b>Note</b> below.
interval	Return prediction or confidence levels. Default is "none". Use interval="pint" to get prediction intervals on new data. Requires that the earth model was built with varmod.method. This argument gets passed on as the type argument to <a href="#">predict.varmod</a> . See its help page for details.

level	Confidence level for the interval argument. Default is 0.95, meaning construct 95% confidence bands (estimate the 2.5% and 97.5% levels).
thresh	Threshold, a value between 0 and 1 when predicting a probability. Only applies when type="class". Default is 0.5. See the <b>Note</b> below.
trace	Default FALSE. Set to TRUE to see which data, subset, etc. predict.earth is using.
...	Unused, but provided for generic/method consistency.

### Value

The predicted values (a matrix for multiple response models).

If type="terms", a matrix with each column showing the contribution of a predictor.

If interval="pint" or "cint", a matrix with three columns:

fit: the predicted values

lwr: the lower confidence or prediction limit

upr: the upper confidence or prediction limit

If interval="se", the standard errors.

### Note

#### Predicting with standard earth models

Use the default type="link", or possibly type="class".

Actually, the "link", "response", and "earth" choices all return the same value unless the glm argument was used in the original call to [earth](#).

#### Predicting with earth-GLM models

This section applies to earth models with a GLM component, i.e., when the glm argument was used in the original call to [earth](#).

The "link" and "response" options: see [predict.glm](#) for a description of these. In brief: for logistic models use type="response" to get probabilities, and type="link" to get log-odds.

Use option "earth" to get the linear fit (this gives the prediction you would get if your original call to earth had no glm argument).

#### Predicting with "class"

Use option "class" to get the predicted class. With option "class", this function first makes predictions with type="response" and then assigns the predicted values to classes as follows:

(i) When the response is a *logical*, predict TRUE if the predicted probability is greater than thresh (default 0.5).

(ii) When the response is a *numeric*, predict TRUE if the predicted value is greater than thresh. Actually, this is identical to the above case, although thresh here may legitimately be a value outside the 0...1 range.

(iii) When the response is a *two level factor*, predict the second level if its probability is more than thresh. In other words, with the default thresh=0.5 predict the most probable level.

(iv) When the response is a *three or more level factor*, predict the most probable level (and thresh is ignored).



**Predicting with "terms"**

The "terms" option returns a "link" response suitable for [termpplot](#). Only the additive terms and the first response (for multi-response models) are returned. Also, "terms" always returns the earth terms, and ignores the GLM component of the model, if any.

**See Also**

[earth](#), [predict](#)

**Examples**

```
data(trees)
earth.mod <- earth(Volume ~ ., data = trees)
predict(earth.mod) # same as earth.mod$fitted.values
predict(earth.mod, data.frame(Girth=10, Height=80)) # yields 17.6
predict(earth.mod, c(10,80)) # equivalent
```

---

predict.varmod	<i>Predict with a varmod model</i>
----------------	------------------------------------

---

**Description**

You probably won't need to call this function directly. It is called by [predict.earth](#) when that function's interval argument is used.

**Usage**

```
## S3 method for class 'varmod'
predict(
  object = stop("no 'object' argument"),
  newdata = NULL,
  type = c("pint", "cint", "se", "abs.residual"),
  level = .95,
  trace = FALSE,
  ...)
```

**Arguments**

object	A varmod object.
newdata	Make predictions using newdata. Default is NULL, meaning return values predicted from the training set.
type	Type of prediction. This is the interval argument of <a href="#">predict.earth</a> . One of "pint" Prediction intervals. "cint" Confidence intervals. Cannot be used with newdata.

"se" Standard error of the parent model residuals.

"abs.residual" The absolute residuals of the parent model on which the residual model regresses.

level	Confidence level for the interval argument. Default is .95, meaning construct 95% confidence bands (estimate the 2.5% and 97.5% levels).
trace	Currently unused.
...	Unused, but provided for generic/method consistency.

### Note

predict.varmod is called by predict.earth when its interval argument is used.

### See Also

[predict.earth varmod](#)

### Examples

```
data(ozone1)

set.seed(1) # optional, for cross validation reproducibility

# note: should really use ncross=30 below but for a quick demo we don't
earth.mod <- earth(O3~temp, data=ozone1, nfold=10, ncross=3, varmod.method="lm")

# call predict.earth, which calls predict.varmod

predict(earth.mod, newdata=ozone1[200:203,], interval="pint", level=.95)
```

---

residuals.earth      *Residuals for an earth model*

---

### Description

Residuals of an [earth](#) model.

### Usage

```
## S3 method for class 'earth'
residuals(object = stop("no 'object' argument"),
          type = NULL, warn = TRUE, ...)
```

**Arguments**

object	An <a href="#">earth</a> object. This is the only required argument.
type	One of:  "earth" (default) Residuals from the <a href="#">lm</a> fit on <code>bx</code> . "response" Residuals as above, but for earth-glm models return the <a href="#">glm</a> response residuals. "standardize" Residuals divided by $se * \sqrt{1 - h_{ii}}$ . See the <code>standardize</code> argument of <a href="#">plot.earth</a> . "delever" Residuals divided by $\sqrt{1 - h_{ii}}$ . See the <code>delever</code> argument of <a href="#">plot.earth</a> .  The following options are for earth-glm models. They return the GLM residuals (from the <a href="#">glm</a> fit on <code>bx</code> ). See <a href="#">residuals.glm</a> for details:  "deviance" "pearson" "working" "partial"  The following options for earth-glm models are redundant. They are provided for compatibility with older versions of earth or other functions:  "glm.response" same as "response" "glm.deviance" same as "deviance" "glm.pearson" same as "pearson" "glm.working" same as "working" "glm.partial" same as "partial"
warn	This function gives warnings when the results are not what you may expect. Use <code>warn=FALSE</code> to turn off just these warnings.
...	Unused, but provided for generic/method consistency.

**Value**

The residual values (will be a matrix for multiple response models).

**See Also**

[earth](#)  
[residuals](#)  
[resid](#) identical to [residuals](#)

**Examples**

```
data(etitanic)
earth.mod <- earth(pclass ~ ., data=etitanic, glm=list(family=binomial))
head(resid(earth.mod, warn=FALSE))      # earth residuals, a column for each response
head(resid(earth.mod, type="response")) # GLM response residuals, a column for each response
```

summary.earth

*Summary method for earth objects***Description**

Summary method for [earth](#) objects.

**Usage**

```
## S3 method for class 'earth'
summary(object = stop("no 'object' argument"),
        details = FALSE, style = c("h", "pmax", "max", "C", "bf"),
        decomp = "anova", digits = getOption("digits"), fixed.point=TRUE,
        newdata = NULL, ...)

## S3 method for class 'summary.earth'
print(x = stop("no 'x' argument"),
      details = x$details,
      decomp = x$decomp, digits = x$digits, fixed.point = x$fixed.point,
      newdata = x$newdata, ...)
```

**Arguments**

object	An <a href="#">earth</a> object. This is the only required argument for <code>summary.earth</code> .
x	A <a href="#">summary.earth</a> object. This is the only required argument for <code>print.summary.earth</code> .
details	Default is FALSE. Use TRUE to print more information about <a href="#">earth-glm</a> models. But note that the displayed Standard Errors and statistics for the GLM coefficients are meaningless (because of the amount of preprocessing done by <code>earth</code> to select the regression terms).
style	Formatting style. One of " h " (default) more compact " pmax " for those who prefer it and for compatibility with old versions of <code>earth</code> " max " is the same as " pmax " but prints max rather than pmax " C " C style expression with zero based indexing " bf " basis function format.
decomp	Specify how terms are ordered. Default is "anova". Use "none" to order the terms as created by the <code>forward.pass</code> . See <a href="#">format.earth</a> for a full description.
digits	The number of significant digits. For <code>summary.earth</code> , the default is <code>getOption("digits")</code> . For <code>print.summary.earth</code> , the default is the <code>\$digits</code> component of <code>object</code> .
fixed.point	Method of printing numbers in matrices. Default is TRUE which prints like this (making it easier to compare coefficients):

```
(Intercept) 15.029
h(temp-58)  0.313
h(234-ibt)  -0.046
...
```

whereas `fixed.point=FALSE` prints like this (which is more usual in R):

```
(Intercept)  1.5e+01
h(temp-58)   3.1e-01
h(234-ibt)   -4.6e-02
...
```

Matrices with two or fewer rows are never printed with a fixed point.

<code>newdata</code>	Default NULL. Else print R-Squared for the new data (and the returned object will have <code>newrsq</code> and <code>newdata</code> fields). Additionally, if a variance model is present print the interval coverage table for the new data.
<code>...</code>	Extra arguments are passed to <code>format.earth</code> .

### Value

The value is the same as that returned by `earth` but with the following extra components.

<code>strings</code>	String(s) created by <code>format.earth</code> . For multiple response models, a vector of strings.
<code>newrsq</code>	Only if <code>newdata</code> was passed to <code>summary.earth</code> .
<code>newdata</code>	Only if <code>newdata</code> was passed to <code>summary.earth</code> .
<code>digits</code>	
<code>details</code>	
<code>decomp</code>	
<code>fixed.point</code>	The corresponding arguments, passed on to <code>print.summary.earth</code> .

### Note

The printed Estimated importance uses `evimp` with the `nsubsets` criterion. The most important predictor is printed first, and so on.

### See Also

[earth](#), [evimp](#), [format.earth](#)

### Examples

```
earth.mod <- earth(Volume~ ., data = trees)
summary(earth.mod, digits = 2)
```

---

update.earth	<i>Update an earth model</i>
--------------	------------------------------

---

### Description

Update an [earth](#) model.

### Usage

```
## S3 method for class 'earth'
update(object = stop("no 'object' argument"),
       formula. = NULL, ponly = FALSE, ..., evaluate = TRUE)
```

### Arguments

object	The earth object
formula.	The formula. argument is treated like earth's formula argument.
ponly	Force pruning only, no forward pass. Default is FALSE, meaning update.earth decides automatically if a forward pass is needed. See note below.
...	Arguments passed on to <a href="#">earth</a> .
evaluate	If TRUE (default) evaluate the new call, else return the call. Mostly for compatibility with the generic <a href="#">update</a> .

### Details

If only the following arguments are used, a forward pass is unnecessary, and update.earth will perform only the pruning pass. This is usually much faster for large models.

```
object
glm
trace
nprune
pmethod
Eval.model.subsets
Print.pruning.pass
Force.xtx.prune
Use.beta.cache
Endspan.penalty
Get.leverages
```

This automatic determination to do a forward pass can be overridden with the ponly argument. If ponly=TRUE the forward pass will be skipped and only the pruning pass will be executed. This is useful for doing a pruning pass with new data. (Use earth's data argument to specify the new data.) Typically in this scenario you would also specify penalty=-1. This is because with sufficient new data, independent of the original training data, the RSS not the GCV should be used for evaluating

model subsets (The GCV approximates what the RSS would be on new data — but here we actually have new data, so why bother approximating. This "use new data for pruning" approach is useful in situations where you don't trust the GCV approximation for your data.) By making `penalty=-1`, `earth` will calculate the RSS, not the GCV. See also the description of `penalty` on the [earth](#) help page.

Another (somewhat esoteric) use of `ponly=TRUE` is to do subset selection with a different `penalty` from that used to build the original model.

With `trace=1`, `update.earth` will tell you if `earth`'s forward pass was skipped.

If you used `keepxy=TRUE` in your original call to `earth`, then `update.earth` will use the saved values of `x`, `y`, etc., unless you specify otherwise by arguments to `update.earth`. It can be helpful to set `trace=1` to see which `x` and `y` is used by `update.earth`.

### Value

The value is the same as that returned by [earth](#). If `object` is the only parameter then no changes are made — the returned value will be the same as the original object.

### See Also

[earth](#)

### Examples

```
data(ozone1)

(earth.mod <- earth(O3 ~ ., data = ozone1, degree = 2))

update(earth.mod, formula = O3 ~ . - temp) # requires forward pass and pruning

update(earth.mod, nprune = 8)                # requires only pruning

update(earth.mod, penalty=1, ponly=TRUE)     # pruning pass only with a new penalty
```

---

varmod

*Variance models for estimating prediction intervals*

---

### Description

A *variance model* estimates the variance of predicted values. It can be used to estimate prediction intervals. See the `interval` argument of [predict.earth](#).

A variance model is built by `earth` if `earth`'s `varmod.method` argument is specified. Results are stored in the `$varmod` field of the `earth` model. See the vignette "[Variance models in earth](#)" for details.

You probably won't need to directly call `print.varmod` or `summary.varmod`. They get called internally by [summary.earth](#).

**Usage**

```
## S3 method for class 'varmod'
summary(
  object = stop("no 'object' argument"),
  level = .95,
  style = "standard",
  digits = 2,
  newdata = NULL,
  ...)
```

**Arguments**

object	A varmod object. This is the only required argument.
level	Same as <code>predict.earth</code> 's <code>level</code> argument.
style	Determines how the coefficients of the varmod are printed by <code>summary.varmod</code> : "standard" (default) "unit" for easy comparison normalize the coefficients by dividing by the first coefficient.
digits	Number of digits to print. Default is 2.
newdata	Default NULL. Else print the interval coverage table for the new data.
...	Dots are passed on.

**Note**

A "varmod" object has the following fields:

- `call` The call used internally in the parent model to build the varmod object.
- `parent` The parent earth model.
- `method` Copy of the `varmod.method` argument to the parent model.
- `package` NULL, unless `method="gam"`, in which case either "gam" or "mgcv".
- `exponent` Copy of the `varmod.exponent` argument to the parent model.
- `lambda` Currently always 1, meaning use absolute residuals.
- `rmethod` Currently always "hc2", meaning correct the residuals with  $1/(1-h_{ii})$ .
- `converged` Did the residual submodel IRLS converge?
- `iters` Number of residual model IRLS iterations (1 to 50).
- `residmod` The residual submodel. So for example, if `varmod.method="lm"`, this will be an `lm` object.
- `min.sd` The predicted residual standard deviation is clamped so it will always be at least this value. This prevents prediction of negative or absurdly small variances. See `earth`'s `varmod.clamp` argument. Clamping takes place in `predict.varmod`, which is called by `predict.earth` when estimating prediction intervals.



- `model.var` An  $n \times 1$  matrix. The `model.var` for an observation is the estimated model variance for that observation over all datasets, and is estimated with repeated cross validation. It is the variance of the mean out-of-fold prediction for that observation over `ncross` repetitions.
- `abs.resids` An  $n \times 1$  matrix. The absolute residuals used to build the residual model.
- `parent.x` An  $n \times p$  matrix. Parent earth model  $x$ .
- `parent.y` An  $n \times 1$  matrix. Parent earth model  $y$ .
- `iter.rsq` Weighted R-Squared of residual submodel `residmod`, after IRLS iteration.
- `iter.stderr` Standard errors of the coefficients of the residual submodel `residmod`, after IRLS iteration.

### See Also

[plot.varmod](#), [predict.varmod](#)

### Examples

```
data(ozone1)

set.seed(1) # optional, for cross validation reproducibility

# note: should really use ncross=30 below but for a quick demo we don't

earth.mod <- earth(O3~temp, data=ozone1, nfold=10, ncross=3, varmod.method="lm")

print(summary(earth.mod)) # note additional info on the variance model

old.mfrow <- par(mfrow=c(2,2), mar=c(3, 3, 3, 1), mgp=c(1.5, 0.5, 0))

plotmo(earth.mod, do.par=FALSE, response.col=1, level=.90, main="earth model: O3~temp")

plot(earth.mod, which=3, level=.90) # residual plot: note 90% pred and darker conf intervals

par(par=old.mfrow)
```

# Index

- \*Topic **datasets**
  - etitanic, 16
  - ozone1, 26
- \*Topic **models**
  - contr.earth.response, 2
  - earth, 3
  - evimp, 18
  - format.earth, 21
  - mars.to.earth, 24
  - model.matrix.earth, 25
  - plot.earth, 27
  - plot.earth.models, 30
  - plot.evimp, 32
  - plotd, 34
  - predict.earth, 39
  - residuals.earth, 42
  - summary.earth, 44
  - update.earth, 46
- \*Topic **regression**
  - earth, 3
- \*Topic **smooth**
  - earth, 3
- airquality, 27
- class, 11
- contr.earth.response, 2
- contrasts, 3
- density, 35–38
- earth, 2, 3, 8, 11, 16, 18, 19, 21, 24–28, 30, 31, 33, 34, 37–47
- earth.object, 4, 10, 11, 26
- earth\_plotmodsel (plot.earth), 27
- etitanic, 16
- evimp, 10, 11, 18, 32, 33, 45
- expand.bpairs, 19
- factor, 14
- factors, 4
- format.earth, 11, 21, 44, 45
- gam, 8
- glm, 5, 14, 26, 34, 37, 43, 44
- grep, 6
- hist, 35–38
- lda, 34, 38
- leaps, 7, 9, 10
- legend, 36
- lm, 6, 8, 26, 34, 37, 43
- lm.fit, 26
- logical, 14
- mars, 10, 24
- mars.to.earth, 24
- model.matrix.earth, 12, 25
- nls, 8
- ozone, 10
- ozone1, 26
- plot, 28, 31
- plot.density, 36–38
- plot.earth, 11, 27, 31, 38, 43
- plot.earth.models, 30, 30, 31, 33
- plot.evimp, 19, 32
- plot.histogram, 37, 38
- plot.varmod, 33, 49
- plotd, 30, 31, 34
- plotmo, 6, 11, 30, 31, 33
- plotres, 27, 28, 30, 31, 34
- pmax, 23
- predict, 35, 37, 41
- predict.earth, 14, 35, 39, 41, 42, 47
- predict.glm, 40
- predict.varmod, 39, 41, 49
- print.summary.earth (summary.earth), 44

qda, [38](#)

resid, [43](#)

residuals, [43](#)

residuals.earth, [42](#)

residuals.glm, [43](#)

rlm, [8](#)

Scale, [9](#)

sqrt, [18](#)

summary.earth, [11](#), [15](#), [21](#), [23](#), [44](#), [44](#), [47](#)

summary.varmod (varmod), [47](#)

termplot, [41](#)

update, [46](#)

update.earth, [4](#), [9](#), [24](#), [46](#)

varmod, [7](#), [16](#), [34](#), [42](#), [47](#)