

Multivariate adaptive regression spline

In [statistics](#), **multivariate adaptive regression splines (MARS)** is a form of [regression analysis](#) introduced by [Jerome H. Friedman](#) in 1991.^[1] It is a [non-parametric regression](#) technique and can be seen as an extension of [linear models](#) that automatically models nonlinearities and interactions between variables.

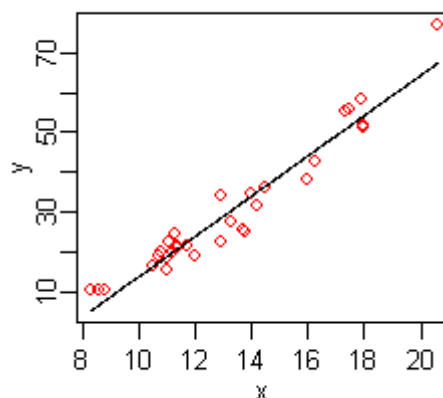
The term "MARS" is trademarked and licensed to Salford Systems. In order to avoid trademark infringements, many open-source implementations of MARS are called "Earth".^{[2][3]}

The basics

This section introduces MARS using a few examples. We start with a set of data: a matrix of input variables x , and a vector of the observed responses y , with a response for each row in x . For example, the data could be:

x	y
10.5	16.4
10.7	18.8
10.8	19.7
...	...
20.6	77.0

Here there is only one [independent variable](#), so the x matrix is just a single column. Given these measurements, we would like to build a model which predicts the expected y for a given x .



A linear model

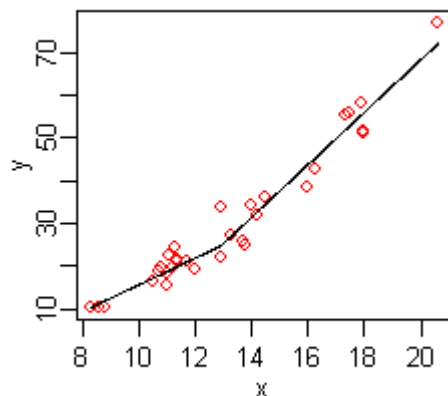
A [linear model](#) for the above data is

$$\hat{y} = -37 + 5.1x$$

The hat on the \hat{y} indicates that \hat{y} is estimated from the data. The figure on the right shows a plot of this function: a line giving the predicted \hat{y} versus x , with the original values of y shown as red dots.

The data at the extremes of x indicates that the relationship between y and x may be non-linear (look at the red dots relative to the regression line at low and high values of x). We thus turn to MARS to automatically build a model taking into account non-linearities. MARS software constructs a model from the given x and y as follows

$$\begin{aligned}\hat{y} = & 25 \\ & + 6.1 \max(0, x - 13) \\ & - 3.1 \max(0, 13 - x)\end{aligned}$$



A simple MARS model of the same data

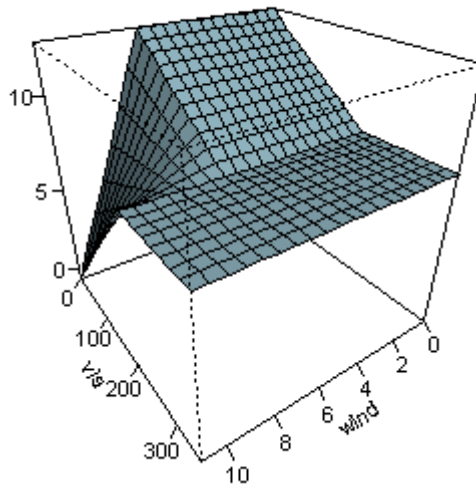
The figure on the right shows a plot of this function: the predicted \hat{y} versus x , with the original values of y once again shown as red dots. The predicted response is now a better fit to the original y values.

MARS has automatically produced a kink in the predicted y to take into account non-linearity. The kink is produced by *hinge functions*. The hinge functions are the expressions starting with **max** (where **max**(a, b) is a if $a > b$, else b). Hinge functions are described in more detail below.

In this simple example, we can easily see from the plot that y has a non-linear relationship with x (and might perhaps guess that y varies with the square of x). However, in general there will be multiple [independent variables](#), and the relationship between y and these variables will be unclear and not easily visible by plotting. We can use MARS to discover that non-linear relationship.

An example MARS expression with multiple variables is

$$\begin{aligned}
 \text{ozone} = & 5.2 \\
 & + 0.93 \max(0, \text{temp} - 58) \\
 & - 0.64 \max(0, \text{temp} - 68) \\
 & - 0.046 \max(0, 234 - \text{ibt}) \\
 & - 0.016 \max(0, \text{wind} - 7) \max(0, 200 - \text{vis})
 \end{aligned}$$



Variable interaction in a MARS model

This expression models air pollution (the ozone level) as a function of the temperature and a few other variables. Note that the last term in the formula (on the last line) incorporates an interaction between **wind** and **vis**.

The figure on the right plots the predicted **ozone** as **wind** and **vis** vary, with the other variables fixed at their median values. The figure shows that wind does not affect the ozone level unless visibility is low. We see that MARS can build quite flexible regression surfaces by combining hinge functions.

To obtain the above expression, the MARS model building procedure automatically selects which variables to use (some variables are important, others not), the positions of the kinks in the hinge functions, and how the hinge functions are combined.

The MARS model

MARS builds models of the form

$$\hat{f}(x) = \sum_{i=1}^k c_i B_i(x).$$

The model is a weighted sum of basis functions $B_i(x)$. Each c_i is a constant coefficient. For example, each line in the formula for ozone above is one basis function multiplied by its coefficient.

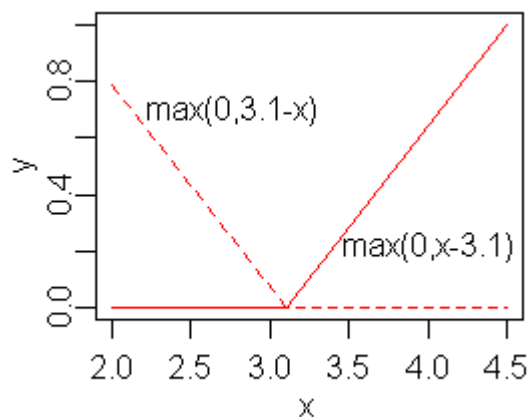
Each **basis function** $B_i(x)$ takes one of the following three forms:

1) a constant 1. There is just one such term, the intercept. In the ozone formula above, the intercept term is 5.2.

2) a *hinge* function. A hinge function has the form **$\max(0, x - \text{constant})$** or **$\max(0, \text{constant} - x)$** . MARS automatically selects variables and values of those variables for knots of the hinge functions. Examples of such basis functions can be seen in the middle three lines of the ozone formula.

3) a product of two or more hinge functions. These basis functions can model interaction between two or more variables. An example is the last line of the ozone formula.

Hinge functions



A mirrored pair of hinge functions with a knot at $x=3.1$

A key part of MARS models are *hinge functions* taking the form

$$\max(0, x - c)$$

or

$$\max(0, c - x)$$

where c is a constant, called the *knot*. The figure on the right shows a mirrored pair of hinge functions with a knot at 3.1.

A hinge function is zero for part of its range, so can be used to partition the data into disjoint regions, each of which can be treated independently. Thus for example a mirrored pair of hinge functions in the expression

$$6.1 \max(0, x - 13) - 3.1 \max(0, 13 - x)$$

creates the [piecewise](#) linear graph shown for the simple MARS model in the previous section.

One might assume that only piecewise linear functions can be formed from hinge functions, but hinge functions can be multiplied together to form non-linear functions.

Hinge functions are also called [ramp](#), [hockey stick](#), or [rectifier](#) functions. Instead of the **max** notation used in this article, hinge functions are often represented by $[\pm(x_i - c)]_+$ where $[\cdot]_+$ means take the positive part.

The model building process

MARS builds a model in two phases: the forward and the backward pass. This two-stage approach is the same as that used by [recursive partitioning](#) trees.

The forward pass

MARS starts with a model which consists of just the intercept term (which is the mean of the response values).

MARS then repeatedly adds basis function in pairs to the model. At each step it finds the pair of basis functions that gives the maximum reduction in sum-of-squares [residual](#) error (it is a [greedy algorithm](#)). The two basis functions in the pair are identical except that a different side of a mirrored hinge function is used for each function. Each new basis function consists of a term already in the model (which could perhaps be the intercept term) multiplied by a new hinge function. A hinge function is defined by a variable and a knot, so to add a new basis function, MARS must search over all combinations of the following:

- 1) existing terms (called *parent terms* in this context)
- 2) all variables (to select one for the new basis function)
- 3) all values of each variable (for the knot of the new hinge function).

To calculate the coefficient of each term MARS applies a linear regression over the terms.

This process of adding terms continues until the change in residual error is too small to continue or until the maximum number of terms is reached. The maximum number of terms is specified by the user before model building starts.

The search at each step is done in a [brute-force](#) fashion, but a key aspect of MARS is that because of the nature of hinge functions the search can be done relatively quickly using a fast least-squares update technique. Actually, the search is not quite brute force. The search can be sped up with a [heuristic](#) that reduces the number of parent terms to consider at each step ("Fast MARS"^[4]).

The backward pass

The forward pass usually builds an [overfit](#) model. (An overfit model has a good fit to the data used to build the model but will not generalize well to new data.) To build a model with better generalization ability, the backward pass prunes the model. It removes terms one by one, deleting the least effective term at each step until it finds the best submodel. Model subsets are compared using the Generalized cross validation (GCV) criterion described below.

The backward pass has an advantage over the forward pass: at any step it can choose any term to delete, whereas the forward pass at each step can only see the next pair of terms.

The forward pass adds terms in pairs, but the backward pass typically discards one side of the pair and so terms are often not seen in pairs in the final model. A paired hinge can be seen in the equation for \hat{y} in the first MARS example above; there are no complete pairs retained in the ozone example.

Generalized cross validation

The backward pass uses generalized cross validation (GCV) to compare the performance of model subsets in order to choose the best subset: lower values of GCV are better. The GCV is a form of [regularization](#): it trades off goodness-of-fit against model complexity.

(We want to estimate how well a model performs on *new* data, not on the training data. Such new data is usually not available at the time of model building, so instead we use GCV to estimate what performance would be on new data. The raw [residual sum-of-squares](#) (RSS) on the training data is inadequate for comparing models, because the RSS always increases as MARS terms are dropped. In other words, if the RSS were used to compare models, the backward pass would always choose the largest model—but the largest model typically does not have the best generalization performance.)

The formula for the GCV is

$$\text{GCV} = \text{RSS} / (N \cdot (1 - (\text{effective number of parameters}) / N)^2)$$

where RSS is the residual sum-of-squares measured on the training data and N is the number of observations (the number of rows in the \mathbf{x} matrix).

The *EffectiveNumberOfParameters* is defined in the MARS context as

$$(\text{effective number of parameters}) = (\text{number of mars terms}) + (\text{penalty}) \cdot ((\text{number of Mars terms}) - 1) / 2$$

where **penalty** is about 2 or 3 (the MARS software allows the user to preset penalty).

Note that

$$(\text{number of Mars terms} - 1) / 2$$

is the number of hinge-function knots, so the formula penalizes the addition of knots. Thus the GCV formula adjusts (i.e. increases) the training RSS to take into account the flexibility of the model. We penalize flexibility because models that are too flexible will model the specific realization of noise in the data instead of just the systematic structure of the data.

Generalized cross-validation is so named because it uses a formula to approximate the error that would be determined by leave-one-out validation. It is just an approximation but works well in practice. GCVs were introduced by Craven and [Wahba](#) and extended by Friedman for MARS.

Constraints

One constraint has already been mentioned: the user can specify the maximum number of terms in the forward pass.

A further constraint can be placed on the forward pass by specifying a maximum allowable degree of interaction. Typically only one or two degrees of interaction are allowed, but higher degrees can be used when the data warrants it. The maximum degree of interaction in the first MARS example above is one (i.e. no interactions or an *additive model*); in the ozone example it is two.

Other constraints on the forward pass are possible. For example, the user can specify that interactions are allowed only for certain input variables. Such constraints could make sense because of knowledge of the process that generated the data.

Pros and cons

No regression modeling technique is best for all situations. The guidelines below are intended to give an idea of the pros and cons of MARS, but there will be exceptions to the guidelines. It is useful to compare MARS to [recursive partitioning](#) and this is done below. (Recursive partitioning is also commonly called *regression trees*, *decision trees*, or [CART](#); see the [recursive partitioning](#) article for details).

- MARS models are more flexible than [linear regression](#) models.
- MARS models are simple to understand and interpret.^[5] Compare the equation for ozone concentration above to, say, the innards of a trained [neural network](#) or a [random forest](#).
- MARS can handle both continuous and [categorical data](#).^{[6][7]} MARS tends to be better than recursive partitioning for numeric data because hinges are more appropriate for numeric variables than the piecewise constant segmentation used by recursive partitioning.
- Building MARS models often requires little or no data preparation.^[5] The hinge functions automatically partition the input data, so the effect of outliers is contained. In this respect MARS is similar to [recursive partitioning](#) which also partitions the data into disjoint regions, although using a different method. (Nevertheless, as with most statistical modeling techniques, known outliers should be considered for removal before training a MARS model.)
- MARS (like recursive partitioning) does automatic [variable selection](#) (meaning it includes important variables in the model and excludes unimportant ones). However, there can be some arbitrariness in the selection, especially when there are correlated predictors, and this can affect interpretability^[5]
- MARS models tend to have a good bias-variance trade-off. The models are flexible enough to model non-linearity and variable interactions (thus MARS models have fairly low bias), yet the constrained form of MARS basis functions prevents too much flexibility (thus MARS models have fairly low variance).
- MARS is suitable for handling fairly large datasets. It is a routine matter to build a MARS model from an input matrix with, say, 100 predictors and 10^5 observations. Such a model can be built in about a minute on a 1 GHz machine, assuming the maximum degree of interaction of MARS terms is limited to one (i.e. additive terms only). A degree two model with the same data on the same 1 GHz machine takes longer—about 12 minutes. Be aware that these times are highly data dependent. Recursive partitioning is much faster than MARS.

- With MARS models, as with any non-parametric regression, parameter confidence intervals and other checks on the model cannot be calculated directly (unlike [linear regression](#) models). [Cross-validation](#) and related techniques must be used for validating the model instead.
- MARS models do not give as good fits as [boosted](#) trees, but can be built much more quickly and are more interpretable. (An 'interpretable' model is in a form that makes it clear what the effect of each predictor is.)
- The `earth`, `mda`, and `polyspline` implementations do not allow missing values in predictors, but free implementations of regression trees (such as `rpart` and `party`) do allow missing values using a technique called surrogate splits.
- MARS models can make predictions quickly. The prediction function simply has to evaluate the MARS model formula. Compare that to making a prediction with say a [Support Vector Machine](#), where every variable has to be multiplied by the corresponding element of every support vector. That can be a slow process if there are many variables and many support vectors.
- The resulting fitted function is not smooth (not differentiable along hinges).

Extensions and related concepts

- [Generalized linear models](#) (GLMs) can be incorporated into MARS models by applying a link function after the MARS model is built. Thus, for example, MARS models can incorporate [logistic regression](#) to predict probabilities.
- [Non-linear regression](#) is used when the underlying form of the function is known and regression is used only to estimate the parameters of that function. MARS, on the other hand, estimates the functions themselves, albeit with severe constraints on the nature of the functions. (These constraints are necessary because discovering a model from the data is an [inverse problem](#) that is not [well-posed](#) without constraints on the model.)
- [Recursive partitioning](#) (commonly called CART). MARS can be seen as a generalization of recursive partitioning that allows the model to better handle numerical (i.e. non-categorical) data.
- [Generalized additive models](#). From the user's perspective GAMs are similar to MARS but (a) fit smooth [loess](#) or polynomial [splines](#) instead of MARS basis functions, and (b) do not automatically model variable interactions. The fitting method used internally by GAMs is very different from that of MARS. For models that do not require automatic discovery of variable interactions GAMs often compete favorably with MARS.

- **TSMARS**. Time Series Mars is the term used when MARS models are applied in a time series context. Typically in this set up the predictors are the lagged time series values resulting in autoregressive spline models. These models and extensions to include moving average spline models are described in "Univariate Time Series Modelling and Forecasting using TSMARS: A study of threshold time series autoregressive, seasonal and moving average models using TSMARS".
- **Bayesian MARS** (BMARS) uses the same model form, but builds the model using a Bayesian approach. It may arrive at different optimal MARS models because the model building approach is different. The result of BMARS is typically an ensemble of posterior samples of MARS models, which allows for probabilistic prediction.^[8]

See also

- [Linear regression](#)
- [Local regression](#)
- [Rational function modeling](#)
- [Segmented regression](#)
- [Spline interpolation](#)
- [Spline regression](#)

References

1. Friedman, J. H. (1991). "Multivariate Adaptive Regression Splines". *The Annals of Statistics*. **19** (1): 1–67. CiteSeerX 10.1.1.382.970 (<https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.382.970>) . doi:10.1214/aos/1176347963 (<https://doi.org/10.1214/aos/1176347963>) . JSTOR 2241837 (<https://www.jstor.org/stable/2241837>) . MR 1091842 (<https://mathscinet.ams.org/mathscinet-getitem?mr=1091842>) . Zbl 0765.62064 (<https://zbmath.org/?format=complete&q=an:0765.62064>) .
2. CRAN Package earth (<https://cran.r-project.org/web/packages/earth/index.html>)
3. Earth – Multivariate adaptive regression splines in Orange (Python machine learning library) (<http://orange.biolab.si/blog/2011/12/20/earth-multivariate-adaptive-regression-splines/>)
4. Friedman, J. H. (1993) *Fast MARS*, Stanford University Department of Statistics, Technical Report 110
5. Kuhn, Max; Johnson, Kjell (2013). *Applied Predictive Modeling*. New York, NY: Springer New York. doi:10.1007/978-1-4614-6849-3 (<https://doi.org/10.1007/978-1-4614-6849-3>) . ISBN 9781461468486.

6. Friedman, Jerome H. (1993). "Estimating Functions of Mixed Ordinal and Categorical Variables Using Adaptive Splines". In Stephan Morgenthaler; Elvezio Ronchetti; Werner Stahel (eds.). *New Directions in Statistical Data Analysis and Robustness*. Birkhauser.
7. Friedman, Jerome H. (1991-06-01). "Estimating Functions of Mixed Ordinal and Categorical Variables Using Adaptive Splines" (<https://apps.dtic.mil/sti/citations/ADA590939>) . *DTIC*. Archived (<https://web.archive.org/web/20220411085148/https://apps.dtic.mil/sti/citations/ADA590939>) from the original on April 11, 2022. Retrieved 2022-04-11.
8. Denison, D. G. T.; Mallick, B. K.; Smith, A. F. M. (1 December 1998). "Bayesian MARS" (<https://link.springer.com/content/pdf/10.1023/A:1008824606259.pdf>) (PDF). *Statistics and Computing*. **8** (4): 337–346. doi:10.1023/A:1008824606259 (<https://doi.org/10.1023%2FA%3A1008824606259>) . ISSN 1573-1375 (<https://www.worldcat.org/issn/1573-1375>) . S2CID 12570055 (<https://api.semanticscholar.org/CorpusID:12570055>) .

Further reading

- Hastie T., Tibshirani R., and Friedman J.H. (2009) *The Elements of Statistical Learning* (<http://www-stat.stanford.edu/~tibs/ElemStatLearn>) , 2nd edition. Springer, ISBN 978-0-387-84857-0 (has a section on MARS)
- Faraway J. (2005) *Extending the Linear Model with R* (<http://www.maths.bath.ac.uk/~jjf23>) , CRC, ISBN 978-1-58488-424-8 (has an example using MARS with R)
- Heping Zhang and Burton H. Singer (2010) *Recursive Partitioning and Applications* (<https://www.amazon.com/Recursive-Partitioning-Applications-Springer-Statistics/dp/1441968237>) , 2nd edition. Springer, ISBN 978-1-4419-6823-4 (has a chapter on MARS and discusses some tweaks to the algorithm)
- Denison D.G.T., Holmes C.C., Mallick B.K., and Smith A.F.M. (2004) *Bayesian Methods for Nonlinear Classification and Regression* (http://www.stat.tamu.edu/~bmallick/wileybook/book_code.html) , Wiley, ISBN 978-0-471-49036-4
- Berk R.A. (2008) *Statistical learning from a regression perspective*, Springer, ISBN 978-0-387-77500-5

External links

Several free and commercial software packages are available for fitting MARS-type models.

Free software

- [R](#) packages:

- `earth` function in the `earth` (<https://cran.r-project.org/web/packages/earth/index.html>) package
- `mars` function in the `mda` (<https://cran.r-project.org/web/packages/mda/index.html>) package
- `polymars` function in the `polspline` (<https://cran.r-project.org/web/packages/polspline/index.html>) package. Not Friedman's MARS.
- `bass` function in the `BASS` (<https://cran.r-project.org/web/packages/BASS/index.html>) package for Bayesian MARS.
- Matlab code:
 - [ARESLab: Adaptive Regression Splines toolbox for Matlab](http://www.cs.rtu.lv/jekabsons/regression.html) (<http://www.cs.rtu.lv/jekabsons/regression.html>)
 - [Code](https://web.stat.tamu.edu/~bmallick/wileybook/book_code.html) (https://web.stat.tamu.edu/~bmallick/wileybook/book_code.html) from the book *Bayesian Methods for Nonlinear Classification and Regression*^[1] for Bayesian MARS.
- Python
 - [Earth – Multivariate adaptive regression splines](http://orange.biolab.si/blog/2011/12/20/earth-multivariate-adaptive-regression-splines/) (<http://orange.biolab.si/blog/2011/12/20/earth-multivariate-adaptive-regression-splines/>)
 - [py-earth](https://github.com/jcrudy/py-earth/) (<https://github.com/jcrudy/py-earth/>)
 - [pyBASS](https://github.com/lanl/pyBASS) (<https://github.com/lanl/pyBASS>) for Bayesian MARS.

Commercial software

- [MARS](http://www.salford-systems.com/mars.php) (<http://www.salford-systems.com/mars.php>) from Salford Systems. Based on Friedman's implementation.
- [STATISTICA Data Miner](https://web.archive.org/web/20101203023609/http://www.statsoft.com/products/data-mining-solutions/) (<https://web.archive.org/web/20101203023609/http://www.statsoft.com/products/data-mining-solutions/>) from StatSoft
- [ADAPTIVEREG](http://support.sas.com/documentation/cdl/en/statug/65328/HTML/default/viewer.htm#statug_adaptivereg_overview.htm) (http://support.sas.com/documentation/cdl/en/statug/65328/HTML/default/viewer.htm#statug_adaptivereg_overview.htm) from SAS.
 1. Denison, D. G. T.; Holmes, C. C.; Mallick, B. K.; Smith, A. F. M. (2002). *Bayesian methods for nonlinear classification and regression*. Chichester, England: Wiley. ISBN 978-0-471-49036-4.

Retrieved from "https://en.wikipedia.org/w/index.php?title=Multivariate_adaptive_regression_spline&oldid=1130941885"