The crs Package

Jeffrey S. Racine
McMaster University

Abstract

This vignette outlines the implementation of the regression spline method contained in the R **crs** package, and also presents a few illustrative examples.

Keywords: nonparametric, semiparametric, regression spline, categorical data.

Introduction

The **crs** package implements a framework for nonparametric regression splines that admits both continuous and categorical predictors. The categorical predictors can be handled in two ways, (i) using kernel weighting where the kernel functions are tailored to the discrete support of the categorical predictors (**?**), and (ii) using indicator basis functions. The practical difference between these two approaches is that the use of indicator basis functions consume degrees of freedom, while kernel weighting does not.

This package implements the approach described in (**?**) and (**?**) when the option `kernel=TRUE` is selected as described below. As well, this package implements related methods that the author is currently developing.

Data-driven methods (leave-out-out cross-validation, **? ?**, **? ?**) can be used for selecting the spline degree, number of segments/knots, and bandwidths. Details of the implementation are as follows:

(i) the degree of the spline and number of segments (i.e. knots minus one) for each continuous predictor can be set manually as can the bandwidths for each categorical predictor (if appropriate)

(ii) alternatively, leave-one-out cross-validation can be used to select either the degree of the spline (holding the number of segments/knots minus one fixed at any user-set value) and bandwidths for the categorical predictors (if appropriate), or the number of segments (holding the degree of the spline fixed at any user-set value) and bandwidths for the categorical predictors (if appropriate), or the number of segments and the degree of the spline for each continuous predictor and bandwidths for each categorical predictor (if appropriate)

(iii) when indicator basis functions are used instead of kernel smoothing, whether to include each categorical predictor or not can be specified manually or chosen via leave-one-out cross-validation

(iv) we allow the degree of the spline for each continuous predictor to include zero, the inclusion indicator for each categorical predictor to equal zero, and the bandwidth for each categorical predictor to equal one, and when the degree/inclusion indicator is zero or the bandwidth is one, the variable is thereby removed from the regression: in this manner, irrelevant predictors can be automatically removed by cross-validation negating the need for pre-testing (**? ?**, **? ?**)

The design philosophy of the **crs** package aims to closely mimic the behavior of the `lm` function. Indeed, the implementation relies on `lm` for computation of the spline coefficients, obtaining fitted values, prediction and the like. 95% confidence bounds for the fit and derivatives are constructed from asymptotic formulae and automatically generated. Below we describe in more detail the specifics of the implementation for the interested reader.

Implementation

Spline regression methods can be limited in their potential applicability as they are predicated on *continuous* predictors. However, in applied settings we often encounter *categorical* predictors such as strength of preference ("strongly prefer", "weakly prefer", "indifferent" etc.) and so forth. When confronted with categorical predictors, researchers typically break their data into subsets governed by the values of the categorical predictors (i.e. they break their data into 'cells') and then conduct regression using only the response and continuous predictors lying in each cell. Though consistent, this 'frequency' approach can be inefficient. Recent developments in the kernel smoothing of categorical data (**?**) suggest more efficient estimation approaches in such settings. The **crs** package considers two complementary approaches that seamlessly handles the mix of continuous and categorical predictors often encountered in applied settings.

For the continuous predictors the regression spline model employs the B-spline routines in the GNU Scientific Library. The B-spline function is the maximally differentiable interpolative basis function (B-spline stands for 'basis-spline').

Heuristically, we conduct linear (in parameter) regression using the R function `lm`. However, we replace the continuous predictors with B-splines of potentially differing order for every continuous predictor. We adopt the `intercept=FALSE` variants (the B-splines will therefore not sum to one, i.e., an order m B-spline with one segment (two knots/breakpoints) has $m + 1$ columns and we drop the first as is often done) and include instead an intercept in the model. This allows multiple bases to coexist when there is more than one continuous predictor without introducing singularities. In addition to these bases we also optionally allow for inclusion of their tensor product (interaction) defined by

$$B_1 \otimes B_2 \otimes \cdots \otimes B_p,$$

where $\otimes$ is the Kronecker product where the products operate column-wise and B_j is the basis matrix for predictor j as outlined above. When the tensor product of the p B-spline bases is omitted we have a semiparametric 'additive' spline model, otherwise we have a fully nonparametric (with interaction among all variables) model. As well, we include the option to use only the tensor product for the multivariate spline basis should the user so desire. Whether or not to include the tensor product bases only, the additive model bases only, or

both the additive components and their tensor product ('functional ANOVA' bases) can be automatically determined via leave-line-out cross-validation (see the options for `basis=` in `?crs`).

We offer the option to use categorical kernel weighting (`lm(...,weights=L)`) to handle the presence of categorical predictors (see below for a description of `L`). We also offer the option of using indicator basis functions for the categorical predictors (again taking care to remove one column to avoid singularity given the presence of the intercept term in the model). These bases are then treated similar to the bases B_j for continuous predictors described above.

Figure **??** presents an example of a B-spline and its first derivative (the spline derivatives are required in order to compute derivatives from the spline regression model).

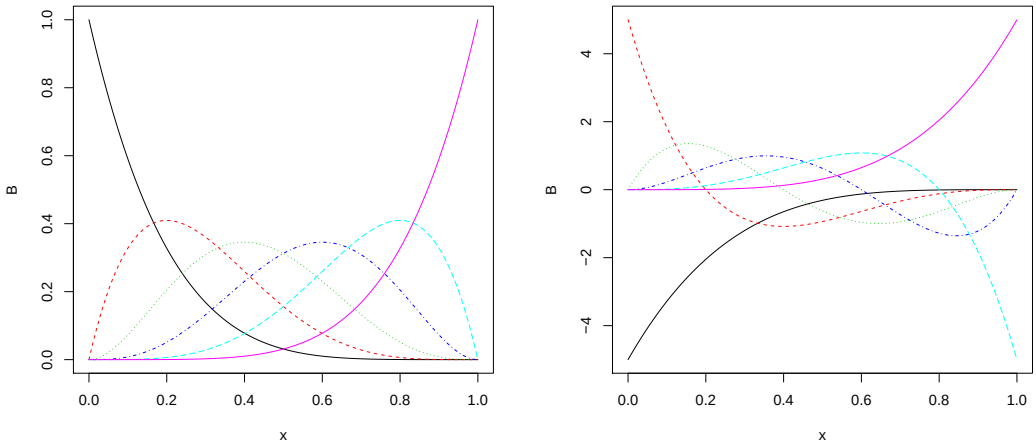

Figure 1: A degree-5 B-spline (left) with one segment (two knots) and its 1st-order derivative (right).

Placement of Knots

The user can determine where knots are to be placed using one of two methods:

(i) knots can be placed at equally spaced quantiles whereby an equal number of observations lie in each segment ('quantile knots')

(ii) knots can be placed at equally spaced intervals ('uniform knots')

Kernel Weighting

Let Z_i be an r-dimensional vector of categorical/discrete predictors. We use z_s to denote the s-th component of z, we assume that z_s takes c_s different values in $D_s \stackrel{def}{=} \{0, 1, \ldots, c_s - 1\}$, $s = 1, \ldots, r$, and let $c_s \geq 2$ be a finite positive constant. For expositional simplicity we will consider the case in which the components of z are unordered.

For an unordered categorical predictor, we suggest using a variant of the kernel function outlined in (**?**) defined as

$$l(Z_{is}, z_s, \lambda_s) = \begin{cases} 1, & \text{when } Z_{is} = z_s, \\ \lambda_s, & \text{otherwise.} \end{cases} \tag{1}$$

Let $\mathbf{1}(A)$ denote the usual indicator function, which assumes the value one if A holds true, zero otherwise. Using (**??**), we can construct a product kernel function given by

$$L(Z_i, z, \lambda) = \prod_{s=1}^{r} l(Z_{is}, z_s, \lambda_s) = \prod_{s=1}^{r} \lambda_s^{\mathbf{1}(Z_{is} \neq z_s)}.$$

Note that when $\lambda_s = 1$ all observations are 'pooled' hence the variable z_s is removed from the resulting estimate, while when $\lambda_s = 0$ only observations lying in a given cell are used to form the estimate.

Estimation

Estimating the model requires construction of the spline bases and their tensor product (if specified) along with the categorical kernel weighting function. Then, for a given degree for each continuous predictor and bandwidth for each categorical predictor (or indicator bases), the model is fit via least-squares. Alternatively, the degree and bandwidth can be jointly determined via leave-one-out cross-validation via the option `cv=TRUE`. This then evaluates the model for each unique combination of degree for each continuous predictor from `degree=0` through `degree=basis.maxdim` (default 5). For each unique combination of degree, the bandwidths ($\in [0, 1]$) are obtained via numerical minimization (see `optim`) and restarting can be conducted via `restarts=`. The model possessing the lowest cross-validation criterion is then selected as the final model.

Cross-Validation

Letting $\hat{\varepsilon}_i$ denote the ith residual from the categorical regression spline model, the leave-one-out cross-validation function is given by

$$CV = \frac{1}{n} \sum_{i=1}^{n} \frac{\hat{\varepsilon}_i^2}{(1 - h_{ii})^2}$$

and this computation can be done with effectively one pass through the data set, where h_{tt} denotes the tth diagonal element of the spline basis projection matrix. Since h_{tt} is computed routinely for robust diagnostics by many statistics programs, this can be computed along with (and hence as cheaply as) the vector of spline coefficients themselves. Thus leave-one-out cross-validation is computationally appealing, particularly for large data sets.

Pruning

Once a model has been selected via cross-validation (i.e. `degree`, `segments`, `include` or `lambda` have been selected), there is the opportunity to (potentially) further refine the model by adding the option `prune=TRUE` to the `crs` function call. Pruning is accomplished by

conducting stepwise cross-validated variable selection using a modified version of the `stepAIC` function in the R **MASS** package where the function `extractAIC` is replaced with the function `extractCV` with additional modifications where necessary. Pruning of potentially superfluous bases is undertaken, however, the pruned model (potentially containing a subset of the bases) is returned *only if its cross-validation score is lower than the model being pruned*. When this is not the case a warning is issued to this effect. A final pruning stage is commonplace in the spline framework and may positively impact on the finite-sample efficiency of the resulting estimator depending on the rank of the model being pruned. Note that this option can only be applied when `kernel=FALSE`.

Illustrative Examples

Next we provide a few illustrative examples that may be of interest to the reader.

Example 1 - One Categorical/One Continuous Predictor

By way of illustration we consider a simple example involving one continuous and one discrete predictor.

```
R> set.seed(42)
R> n <- 1000
R> x <- runif(n)
R> z <- rbinom(n,1,.5)
R> y <- cos(2*pi*x) + z + rnorm(n,sd=0.25)
R> z <- factor(z)
R> model <- crs(y~x+z,basis="auto",basis.maxdim=5,kernel=TRUE)
R> summary(model)


Call:
crs.formula(formula = y ~ x + z, basis.maxdim = 5, kernel = TRUE,
    basis = "auto")

Kernel Weighting/B-spline Bases Regression Spline

There is 1 continuous predictor
There is 1 categorical predictor
Knot type: quantiles
Model complexity proxy: degree-knots
Spline degree/number of segments for x: 3/2
Bandwidth for z: 0.000229
Basis type: additive
Training observations: 1000
Rank of model frame: 5
Residual standard error: 0.2457 on 995 degrees of freedom
Multiple R-squared: 0.9266,   Adjusted R-squared: 0.9263
Cross-validation score: 0.061313731
```

The function `crs` called in this example returns a `crs` object. The generic functions `fitted` and `residuals` extract (or generate) estimated values and residuals. Furthermore, the functions `summary`, `predict`, and `plot` (options `mean=FALSE`, `deriv=FALSE`, `ci=FALSE`, `plot.behavior = c("plot", "plot-data", "data")`) support objects of this type. Figure **??** presents summary output in the form of partial regression surfaces (predictors not appearing on the axes are held constant at their medians/modes).

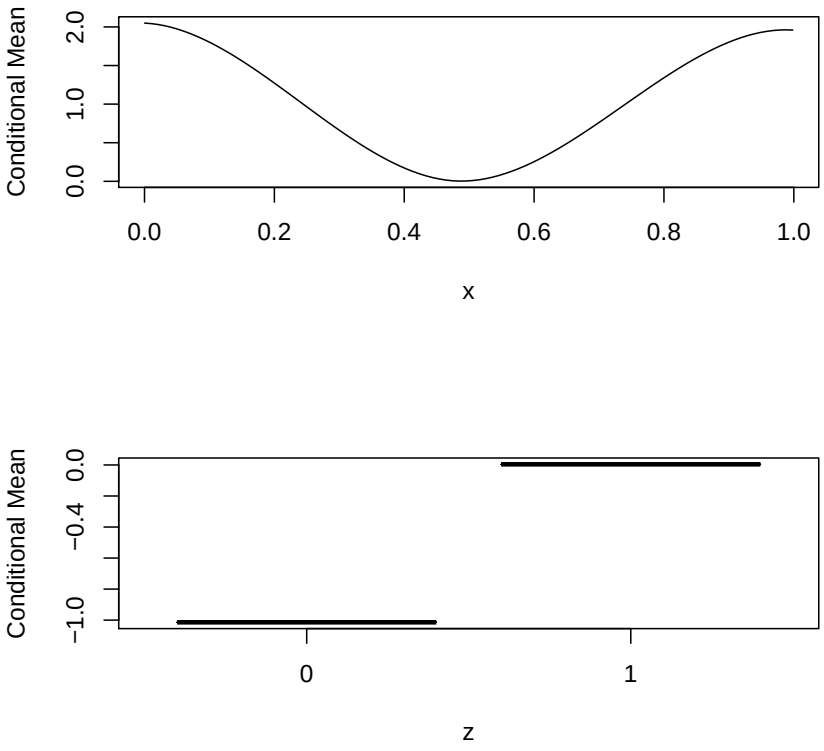

Figure 2: A simple example using `plot(model,mean=TRUE)`.

Example 2 - Regression Discontinuity Design

By way of illustration we consider a simple example involving two continuous predictors and one categorical predictor. In this example there is a 'discontinuity' in the regression surface potentially demarcated by the discrete predictor.

```
R> set.seed(1234)
R> n <- 10000
R> x1 <- runif(n)
R> x2 <- runif(n)
R> dgp <- numeric(n)
```

```
R> z <- ifelse(x1>.5,1,0)
R> dgp <- cos(2*pi*x1)+sin(2*pi*x2)+2*z
R> z <- factor(z)
R> y <- dgp + rnorm(n,sd=1)
R> model <- crs(y~x1+x2+z,complexity="degree",basis="auto",basis.maxdim=8)
R> summary(model)

Call:
crs.formula(formula = y ~ x1 + x2 + z, basis.maxdim = 8, complexity = "degree",
    basis = "auto")

Indicator Bases/B-spline Bases Regression Spline

There are 2 continuous predictors
There is 1 categorical predictor
Knot type: quantiles
Model complexity proxy: degree
Spline degree/number of segments for x1: 6/1
Spline degree/number of segments for x2: 5/1
Inclusion indicator for z: 1
Basis type: additive
Pruning of final model: FALSE
Training observations: 10000
Rank of model frame: 13
Residual standard error: 1.006 on 9987 degrees of freedom
Multiple R-squared: 0.6601,   Adjusted R-squared: 0.6597
Cross-validation score: 1.0133543
```

Figure **??** plots the resulting estimate. The discontinuity occurs when $x_1 > 0.5$ but the nature of the discontinuity is unknown as is the functional form on either side of the potential discontinuity. The categorical regression spline is able to detect this 'break' and testing for a significant break involves nothing more than an (asymptotic) F-test as the following illustrates (note the argument `include=0` says to drop the one categorical predictor or, say, `c(1,1,...,0,1...,1)` for multivariate categorical predictors).

```
R> ## When kernel=FALSE, we could use the anova() function
R> model.res <- crs(y~x1+x2+z,cv=FALSE,degree=model$degree,basis=model$basis,include=0)
R> anova(model.res$model.lm,model$model.lm)

Analysis of Variance Table

Model 1: y ~ P
Model 2: y ~ P
  Res.Df   RSS Df Sum of Sq    F Pr(>F)
1   9988 11175
2   9987 10107  1      1068 1055 <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
R> ## As this only works with kernel=FALSE, we could also do this manually...
R> F <- model$df.residual*(sum(residuals(model.res)^2)
+                          -sum(residuals(model)^2))/sum(residuals(model)^2)
R> F

[1] 1055

R> ## Compute the P-value for the F-statistic
R> P <- 1-pf(F,1,model$df.residual)
R> P

[1] 0
```

Note that `summary` supports the option `sigtest=TRUE` that conducts an F-test for significance for each predictor (though be warned that this can be fragile when `basis="additive-tensor"`).

```
R> summary(model,sigtest=TRUE)

Call:
crs.formula(formula = y ~ x1 + x2 + z, basis.maxdim = 8, complexity = "degree",
    basis = "auto")

Indicator Bases/B-spline Bases Regression Spline

There are 2 continuous predictors
There is 1 categorical predictor
Knot type: quantiles
Model complexity proxy: degree
Spline degree/number of segments for x1: 6/1
Spline degree/number of segments for x2: 5/1
Inclusion indicator for z: 1
Basis type: additive
Pruning of final model: FALSE
Training observations: 10000
Rank of model frame: 13
Residual standard error: 1.006 on 9987 degrees of freedom
Multiple R-squared: 0.6601,   Adjusted R-squared: 0.6597
Cross-validation score: 1.0133543

Predictor significance test:
Predictor x1: Df = 6, F = 808.5, Pr(>F) = 0
Predictor x2: Df = 5, F = 950.4, Pr(>F) = 0
Predictor z: Df = 1, F = 1055, Pr(>F) = 4.658e-220
```

Acknowledgements

I would like to gratefully acknowledge support from the Natural Sciences and Engineering Research Council of Canada (http://www.nserc.ca), the Social Sciences and Humanities

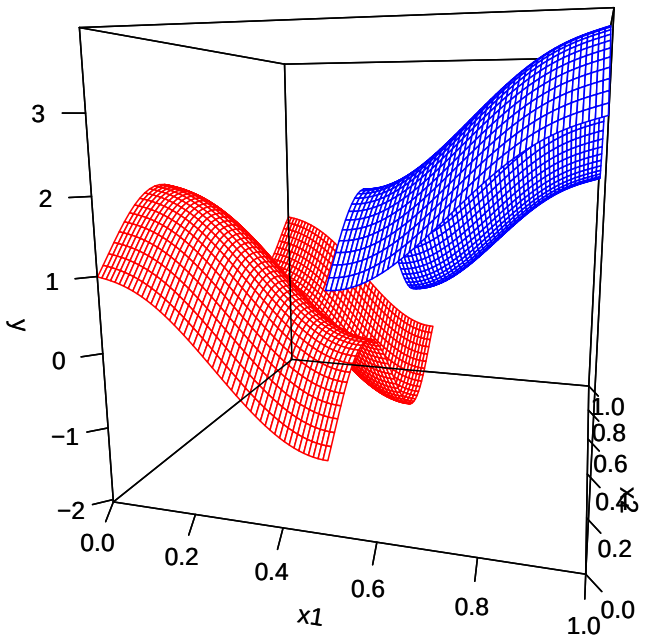

Figure 3: Regression discontinuity design estimated by `crs`.

Research Council of Canada (http://www.sshrc.ca), and the Shared Hierarchical Academic Research Computing Network (http://www.sharcnet.ca).

Affiliation:

Jeffrey S. Racine
Department of Economics
McMaster University
Hamilton, Ontario, Canada, L8S 4L8
E-mail: racinej@mcmaster.ca
URL: http://www.mcmaster.ca/economics/racine/