PACKAGE `crs` FAQ

JEFFREY S. RACINE

Contents

Date: August 11, 2014.

1. Overview and Current Version

This set of frequently asked questions is intended to help users who are encountering unexpected or undesired behavior when trying to use the `crs` package.

If you encounter any issues with the `crs` package, kindly first ensure that you have the most recent version of `crs`, `R`, and `RStudio` (if appropriate) installed. Sometimes issues encountered using outdated versions of software have been resolved in the current versions, so this is the first thing one ought to investigate when the unexpected occurs.

Having ensured that the problem persists with the most recently available versions of all software involved, kindly report any issues you encounter to me, and please include your code, data, version of the `crs` package and version of `R` used so that I can help track down any such issues (racinej@mcmaster.ca). And, of course, if you encounter an issue that you think might be of interest to others, kindly email me the relevant information and I will incorporate it into this FAQ.

This FAQ refers to the most recent version, which as of this writing is 0.15-23. Kindly update your version should you not be using the most current (from within R, `update.packages()` ought to do it, though also see 2.3 below.). See the appendix in this file for cumulative changes between this and previous versions of the `crs` package.

2. Frequently Asked Questions

2.1. **How do I cite the `crs` package?** Once you have installed the `crs` package (`install.packages("crs")`), if you load the `crs` package (`library("crs")`) and type `citation("crs")` you will be presented with the following information.

```
> citation("crs")

To cite package `crs' in publications use:

  Jeffrey S. Racine and Zhenghua Nie (2014). crs: Categorical
  Regression Splines. R package version 0.15-23.
  https://github.com/JeffreyRacine/R-Package-crs/

A BibTeX entry for LaTeX users is

  @Manual{,
    title = {crs: Categorical Regression Splines},
    author = {Jeffrey S. Racine and Zhenghua Nie},
    year = {2014},
    note = {R package version 0.15-23},
```

```
  url = {https://github.com/JeffreyRacine/R-Package-crs/},
}
```

2.2. **I have never used R before. Can you direct me to some introductory material that will guide me through the basics?** There are many excellent introductions to the R environment with more on the way. First, I would recommend going directly to the R website (http://www.r-project.org) and looking under Documentation/Manuals (http://cran.r-project.org/manuals.html) where you will discover a wealth of documentation for R users of all levels. See also the R task views summary page (http://cran.nedmirror.nl/web/views/index.html) for information grouped under field of interest. A few documents that I mention to my students which are tailored to econometricians include http://cran.r-project.org/doc/contrib/Verzani-SimpleR.pdf, Cribari-Neto & Zarkos (1999) [1], Racine & Hyndman (2002) [5] and Farnsworth (2006) [3], to name but a few.

Those looking for exemplar data sets outside of those contained in the crs package are directed to the Ecdat [2] and AER [4] packages.

I maintain a 'Gallery' to provide a forum for users to share code and discover examples and illustrations which can be found at http://socserv.mcmaster.ca/racinej/Gallery/Home.html.

Often the best resource is right down the hall. Ask a colleague whether they use or know anyone who uses R, then offer to buy that person a coffee and along the way drop something like "I keep hearing about the R project. . . I feel like such a Luddite. . . "

2.3. **How do I keep all R packages on my system current?** Run the command update.packages(checkBuilt=TRUE,ask=FALSE), which will not only update all packages that are no longer current, but will also update all packages built under outdated installed versions of R, if appropriate.

2.4. **It seems that there are a lot of packages that must be installed in order to conduct econometric analysis (tseries, lmtest, np, etc.). Is there a way to avoid having to individually install each package individually?** Certainly. The Comprehensive R Archive Network (CRAN) is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. The CRAN 'task view' for computational econometrics might be of particular interest to econometricians. The econometric task view provides an excellent summary of both parametric and nonparametric econometric packages that exist for the R environment and provides one-stop installation for these packages.

See cran.r-project.org/web/views/Econometrics.html for further information.

To automatically install a task view, the `ctv` package first needs to be installed and loaded, i.e.,

```
install.packages("ctv")
library("ctv")
```

The econometric task view can then be installed via `install.views()` and updated via `update.views()` (which first assesses which of the packages are already installed and up-to-date), i.e.,

```
install.views("Econometrics")
```

or

```
update.views("Econometrics")
```

2.5. **Is there a 'gentle guide' to the `crs` package that contains some easy to follow examples?** Perhaps the most gentle introduction is contained in the `crs` package itself in the form of a 'vignette'. To view the vignette run R, install the `crs` package (`install.packages("crs")`), then type `vignette("crs",package="crs")` to view or print the vignette.

See also `vignette("spline_primer",package="crs")` for a vignette that presents a 'gentle' introduction to regression splines.

For a listing of all routines in the `crs` package type: `library(help="crs")`.

2.6. **I noticed you have placed a new version of the `crs` package on CRAN. How can I determine what has been changed, modified, fixed etc?** See the CHANGELOG on the CRAN site (`http://cran.r-project.org/web/packages/crs/ChangeLog`), or go to the end of this document where the CHANGELOG is provided for your convenience.

2.7. **How can I read data stored in various formats such as Stata, SAS, Minitab, SPSS etc. into the R program?** Install the foreign library via `install.packages("foreign")` then do something like

```
mydat <- read.dta("datafile.dta"),
```

where `datafile.dta` is the name of your Stata data file. Note that, as of version 0.8-34, the foreign package function `read.dta` supports reading files directly over the Internet making for more portable code. For instance, one could do something like

```
mydat <- read.dta(file="http://www.principlesofeconometrics.com/stata/mroz.dta")
```

as one could always do with, say, `read.table()`.

2.8. **Where can I get some examples of R code for the crs package in addition to the examples in the help files?** Start `R` then type `demo(package="crs")` and you will be presented with a list of demos for constrained estimation, inference, and so forth. To run one of these demos type, for example, `demo(radial_rgl)` (note that you must first install the `rgl` package to run this particular demo).

To find the location of a demo type `system.file("demo","radial_rgl.R",package="crs")` for example, then you can take the source code for this demo and modify it for your particular application.

2.9. **Can I use crs to perform linear regression?** Certainly! Simply use the options `cv="none"`, `degree=rep(1,q)` and `segments=rep(1,q)` where `q` is the number of continuous predictors and `kernel=FALSE` as per the following example:

```
n <- 1000
x1 <- runif(n)
x2 <- runif(n)
z1 <- rbinom(n,1,.1)
z2 <- rbinom(n,1,.1)
y <- x1+x2^2+z1+z2+rnorm(n)
z1 <- factor(z1)
z2 <- factor(z2)
model.lm <- lm(y~x1+x2+z1+z2)
summary(model.lm)
model.crs <- crs(y~x1+x2+z1+z2,cv="none",kernel=FALSE,degree=rep(1,2),segments=rep(1,2))
summary(model.crs)
```

You will note that the summary statistics are identical for each model. Also, when conducting search the initial values are set so that the first model estimated is linear (albeit with `kernel=TRUE` by default) so if the linear model is optimal it will be the one chosen by cross-validation in probability.

2.10. **I would like more/less information displayed when conducting search using the NOMAD routines...** This is accomplished by feeding the argument `opts=list("DISPLAY_DEGREE"=x)` to `crs` where x is a non-negative integer. Setting x=0 produces no information whatsoever while integers $x \geq 1$ provide successively more information.

2.11. `crs` **consumes a lot of memory when conducting cross-validation via NOMAD. Can I control this?** By default, the minimum degrees of freedom possible is set to

$n - k = 1$ where n is the sample size and k is the trace of the basis matrix. This is done in order to allow for extremely complex relationships to be estimated. But for some situations one might not want to 'go there' for two reasons, namely a) memory consumption and b) improbable models having nearly as many predictors as observations.

During the search process if you have a large number of predictors and a large number of observations, you will end up computing models having a minuscule number of degrees of freedom and, when this occurs, the basis function may well consume an inordinate amount of memory. Plus the final model determined by the search process may well end up being much more parsimonious. By way of illustration if `n=100000`, `k=20`, `degree.max=10`, and `segments.max=10` you will likely compute models with spline bases of dimension `100000x99999` when conducting cross-validated search (i.e. models that hit the minimum number of degrees of freedom).

In such cases you might wish to restrict the minimum degrees of freedom via setting `cv.df.min=n-j` where `n` is your sample size and `j` is, say, `1000` so that the maximum dimension of the B-spline would be less than or equal to `j` (e.g. the maximum dimension of the spline basis would now be `100000x1000`). Of course, you ought to make sure that your final model does not hit this bound as this would indicate that imposing this restriction is binding on the search process which is of course undesirable.

2.12. **When I have a large number of regressors/data the function `crs` just 'sits there' when conducting cross-validation via NOMAD...** First, if you are concerned that the code is indeed just 'sitting there', you can verify that search is progressing by changing the `"DISPLAY_DEGREE"` setting in the opts list along the lines of the following:

```
opts <- list("MAX_BB_EVAL"=10000,
             "EPSILON"=.Machine$double.eps,
             "INITIAL_MESH_SIZE"="r1.0e-01",
             "MIN_MESH_SIZE"=paste("r",sqrt(.Machine$double.eps),sep=""),
             "MIN_POLL_SIZE"=paste("r",sqrt(.Machine$double.eps),sep=""),
             "DISPLAY_DEGREE"=3)

model <- crs(...,opts=opts)
```

will print out in gory detail exactly what the search engine is doing.

However, if this reveals that there something odd going on (i.e. you are seeing a lot of `inf` function values being printed out), then you might wish to begin by restricting the dimension of the combinatoric search process. By default `degree.max=10` for each predictor and `segments.max=10` as well. So this can lead to a basis with 21 columns for one predictor

and when using `basis="tensor"` or `basis="auto"` (which computes both the tensor and additive bases) the dimension of the basis can swamp the number of observations in the sample (e.g. with 4 regressors we can have a tensor product multivariate basis that has up to `21^4=194481` columns when using the default `degree.max=10` for each predictor and `segments.max=10`). For this illustration, the cross-validation function can approach ∞ if the sample size approaches `194491` from above and the search process will be searching for a non-∞ value in order to proceed or terminate.

So, in such cases begin by restricting the dimension of the spline basis matrix by setting, for instance, the minimum degrees of freedom via `cv.df.min=n-j` where `n` is your sample size and `j=500` so that the maximum dimension of the B-spline would be less than or equal to `j`. Or you can restrict the dimension of the spline basis matrix by setting, for instance, `degree.max=2` and `segments.max=2`. Or begin by searching only over the spline degree by setting `complexity="degree"` (the default is `complexity="degree-knots"`). The routine will throw a warning if you have a solution that hits the maximum value of `degree.max` or `segments.max` and offer some practical advice in these cases.

Alternatively, restrict attention to additive (semiparametric) splines by setting `basis="additive"` (e.g. with 4 regressors we can have a tensor product multivariate basis that has up to `21x4=84` columns when using the default `degree.max=10` for each predictor and `segments.max=10`) at the cost of imposing additivity which can be restrictive.

Alternatively, consider kernel regression that does not suffer from this computational limitation (see e.g. the `np` package).

2.13. **My estimated model is not 'smooth' (e.g. cross-validation chooses, say, the spline degree=1 and number of segments=3). How can I modify this?** Unlike, say, smoothing splines [6] that penalize 'roughness' (the second derivative of the estimate) and fix the spline degree at, say, three, regression splines fitted by cross-validation can deliver a model that minimizes the objective function without regard to an ad hoc proxy for smoothness (a strength, not a weakness in my opinion).

(1) `All Predictors Relevant`: If, however, you wish an estimate that is, say, twice continuously differentiable, simply set `degree.min=3` (one degree higher than the desired degree of smoothness) and then determine the appropriate model via cross-validation subject to this constraint (i.e. re-estimate your model with the option `degree.min=3` for example). Otherwise, you can simply override what cross-validation delivered via `cv="none"` and `degree=c(3,3,...)` etc. (i.e. set the degree and segments in an

ad-hoc fashion, which I would not recommend). Of course, this will not be optimal according to the cross-validation criterion but will achieve the desired degree of smoothness.

(2) `Some Predictors Irrelevant`: In some cases the optimal spline degree will be zero or bandwidth one or inclusion indicator zero (the latter occurring when `kernel=FALSE` is specified). This means that cross-validation has automatically removed one or more variables.

In this case first remove these variables from the model to be estimated, then re-run your model on the subset of relevant variables only but following the advice in (1) which will achieve the desired degree of smoothness subject to the irrelevant variables no longer being present in the final model.

2.14. **The crs package implements 'regression splines' and optimizes the spline degree and knot vector by default. But 'smoothing splines'** [6] **typically set the degree to 'cubic' and penalize roughness. Can we use `crs()` (i.e. regression splines) to mirror the cubic smoothing spline approach?** Certainly. Consider the following illustration:

```
library(crs)
data(cps71)
attach(cps71)

model <- crs(logwage~age)
summary(model)
```

You can see that cross-validating both the spline degree and number of knots chooses a degree 2 spline for age and, as a consequence, `plot(model,deriv=1)` produces a derivative that is nonsmooth (i.e. piecewise linear). If one preferred to mirror the cubic spline approach (or for that matter, any fixed order spline), then the following will optimize knots only holding the degree constant at, say, 3 (and will, in general, be faster to solve since we optimize over fewer smoothing parameters).

```
model <- crs(logwage~age,complexity="knots",degree=3)
summary(model)
```

This model uses a higher degree than the previous one that optimized both the degree and number of knots, but fewer knots. However, note that the cross-validation function is minimized (i.e. improved) for the model that optimizes both the degree and number of knots

hence this latter model is optimal according to the cross-validation criterion. See also 2.13 above for further caveats.

2.15. **I estimated a parametric model using the lm() function. How can I compare the cross-validation score from the crs() approach with that for the parametric model?** This can be readily achieved for the parametric model as follows:

```
data(wage1)
model.lm <- lm(lwage ~ married + female + nonwhite + educ +
               exper + tenure, data = wage1)
cv.lm <- mean(residuals(model.lm)^2/(1-hatvalues(model.lm))^2)
cv.lm
```

You can then compare this with that for the `crs` model. If the cross-validation score is lower for one model, that indicates that the model possessing the lowest score is to be preferred.

2.16. **Why do some runs result in a function value of 1.340781e+154 when conducting multistarting?** As of version 0.15-1 we conduct extensive testing for ill-conditioned bases (univariate and multivariate) and adjust search limits accordingly. However, when a multivariate basis is ill-conditioned we apply a large penalty (`sqrt(.Machine$double.xmax)` which equals 1.340781e+154 on most processors). Though the search process will try to detect a minimum it can fail here if the objective function is 'flat' in a neighborhood of the initial values.

When this occurs you can either increase `nmulti` and/or decrease `degree.max` and restart the search.

Alternatively, you might try `singular.ok=TRUE`.

Note also that as of version 0.15-1, the initial search values will be degree one and segment one (i.e. a linear model) unless you provide the vectors `degree=c(...)` and `segments=c(...)` which will then be used instead as the starting values for the first multistart.

2.17. **snomadr appears to be crashing.** If you receive the message

```
Calling NOMAD (Nonsmooth Optimization by Mesh Adaptive Direct Search)


 *** caught segfault ***
address 0x68, cause 'memory not mapped'

Traceback:
 1: .Call(smultinomadRSolve, ret)
```

kindly first ensure that you have write privileges in your current directory (`snomadr` creates temporary files in the current working directory and if this operation fails you may receive this error).

Note this should not occur when using version 0.15-13 or higher since temporary files are no longer generated.

2.18. **How do I estimate the conditional quantile rather than the conditional mean?** Simply specify the option `tau=0.5` to estimate the conditional median regression spline, or any quantile you choose to specify where $\tau \in (0, 1)$, as in

```
model <- crs(y~x1+x2,tau=0.5)
```

See also `demo(cqrs)` for a demonstration.

2.19. **How can I save a PDF of a plot created with the option persp.rgl=TRUE?.** Version 0.15-1 has added support for `RGL` via the `rgl` package which is a 3D real-time rendering device driver system for R using OpenGL. These plots are dynamic so you can spin them and resize them using your keypad/mouse. However, they are not standard graphics objects that can be saved using R commands such as `pdf()`. But they can be saved as a PDF by first calling rgl and then issuing the command `rgl.postscript("foo.pdf","pdf")` where `foo.pdf` is the desired name of your PDF file as the following illustrates:

```
n <- 1000
x1 <- sort(rnorm(n))
x2 <- rnorm(n)
y <- x1^3 + rnorm(n,sd=.1)
model <- crs(y~x1+x2)
plot(model,mean=T,persp.rgl=T)
rgl.postscript("foo.pdf","pdf")
```

However, this pdf driver does not support some features such as transparency etc. A better alternative is to create a `png` file as follows:

```
n <- 1000
x1 <- sort(rnorm(n))
x2 <- rnorm(n)
y <- x1^3 + rnorm(n,sd=.1)
model <- crs(y~x1+x2)
plot(model,mean=T,persp.rgl=T)
rgl.snapshot("foo.png")
```

and then include this in your LaTeX document using `\includegraphics[scale=.5]{foo.png}`.

2.20. **Is it possible to use B-splines instead of indicator bases or kernel weighting for discrete predictors?** Providing the discrete predictors are numeric, certainly. Just make sure they are of type numeric and they will be treated the same as continuous predictors, as in the following example.

```
n <- 1000
x <- runif(n,-2,2)
z <- rbinom(n,1,.5)
y <- x^3 + z + rnorm(n,sd=.1)
model <- crs(y~x+z)
summary(model)
```

Note in summary that you are told 'There are 2 continuous predictors' (i.e. numeric predictors). Note also that the message 'optimal degree equals search maximum (10): rerun with larger degree.max optimal degree equals search maximum (1): rerun with larger degree.max' will appear simply because the default maximum degree is 10 but the program automatically checks for the maximum *well-conditioned* basis which, for the binary numeric predictor, is 1 (you can ignore this message here).

2.21. **I have noticed that as more categorical predictors are added or as the number of outcomes for each categorical predictor increases, computation time increases when `kernel=TRUE`. Why is this so and can anything be done?** When `kernel=TRUE` we need to compute kernel weighted regression for each unique combination of the categorical predictors. So if you have one binary predictor, estimation involves two calls to the weighted least squares solver. Now suppose that you have three categorical predictors each having $c = 10$ outcomes. Now estimation involves $10^3 = 1000$ calls to the weighted least squares solver. This will affect all aspects of estimation (cross-validation etc.) which results in increases in computation time.

As of version 0.15-8 and up, you can potentially reduce computation time *when you have categorical predictors* by discretizing the bandwidth `lambda` (when `lambda.discrete=TRUE` we divide `lambda` $\in [0, 1]$ into `lambda.discrete.num=100 (+1)` values, $(0/100, 1/100, \ldots, 100/100)$) so that integer (discrete) search via NOMAD will be undertaken rather than continuous search which is done by setting `lambda.discrete=TRUE`. This can potentially reduce run-time with little expected loss in accuracy for the cross-validated search procedure. However, when there are many categorical predictors each having > 2 outcomes, search based on `lambda.discrete=TRUE` may be more likely to become ensnared in local minima than the default (i.e. properly treating the bandwidths for the categorical predictors as continuous).

To reduce computation time further, you can also decrease the number of times search is restarted from different (random) starting points by setting `nmulti=`i to $i < 5$ (the default is 5) but I would caution against doing this except when conducting exploratory data analysis (the objective function is typically nonsmooth and may possess multiple local minima).

2.22. **The R function 'lag()' does not work as I expect it to. How can I create the lth lag of a numeric variable in R to be fed to functions in the `crs` package?** You can use the `embed` function to accomplish this task. Here is a simple function that might work more along the lines that you expect. By default we 'pad' the vector with NAs but you can switch this to `FALSE` if you prefer. The function will return a vector of the same length as the original vector with NA's padded for the missing values.

```
lag.numeric <- function(x,l=1,pad.NA=TRUE) {
  if(!is.numeric(x)) stop("x must be numeric")
  if(l < 1) stop("l (lag) must be a positive integer")
  if(pad.NA) x <- c(rep(NA,l),x)
  return(embed(x,l+1)[,l+1])
}
x <- 1:10
x.lag.1 <- lag.numeric(x,1)
```

2.23. **How can I turn off all console I/O?.** To disable all console I/O, set `options(crs.messages=FALSE)` and wrap the function call in `suppressWarnings()` to disable any warnings printed to the console. For instance

```
library(crs)
options(crs.messages=FALSE)
set.seed(42)
n <- 100
x <- sort(rnorm(n))
z <- factor(rbinom(n,1,.5))
y <- x^3 + rnorm(n)
model.crs <- suppressWarnings(crs(y~x+z))
```

ought to produce no console I/O whatsoever in the call to `crs`.

Note that for direct calls to `snomadr` you instead need to set `opts=list("DISPLAY_DEGREE"=0)` (see `?snomadr` for details).

References

[1] Francisco Cribari-Neto and Spyros G Zarkos. R: Yet another econometric programming environment. *Journal of Applied Econometrics*, 14(3):319–29, May-June 1999. Available at http://ideas.repec.org/a/jae/japmet/v14y1999i3p319-29.html.

[2] Yves Croissant. *Ecdat: Data sets for econometrics*, 2011. R package version 0.1-6.1.

[3] Grant V. Farnsworth. Econometrics in R. Technical report, October 2008. Available at http://cran.r-project.org/doc/contrib/Farnsworth-EconometricsInR.pdf.

[4] Christian Kleiber and Achim Zeileis. *Applied Econometrics with R*. Springer-Verlag, New York, 2008. ISBN 978-0-387-77316-2.

[5] J. S. Racine and R. Hyndman. Using R to teach econometrics. *Journal of Applied Econometrics*, 17(2):175–189, 2002.

[6] G. Wahba. *Spline Models for Observational Data*. SIAM, 1990.

Changes from Version 0.15-22 to 0.15-23 [11-Aug-2014]

- Fixed bugs on setting seed in `snomadr` so that successive calls with restarting result in the same solution (thanks to Arne Henningson for detecting and reporting this issue)
- Addressed issue with `cv.aic` and `npglpreg` where degenerate solution could occur when shrinking was invoked
- Enhanced the robust measure of scale to include `mad` which is more robust than `IQR`

Changes from Version 0.15-21 to 0.15-22 [22-Jan-2014]

- `crs.messages=FALSE` was being ignored in `frscv`, `krscv`, `frscvNOMAD`, and `krscvNOMAD`, as was the passing of additional arguments via `opts=list()` to `snomadr` from certain functions
- Default settings for tuning parameters for `snomadr` in `npglpreg` have been optimized based on extensive simulations for simulated and real datasets (we 'optimized the optimizer' so to speak)
- Fixed glitch in `npglpreg` where `cv.aic` was only working properly for the local constant estimator (i.e. `degree=0`)
- `bwscaling` is deprecated in the latest (development) version of np which necessitated changes to `npglpreg`. In the process the way scaling is performed ought to be more robust in general due to changes to default starting values for numeric predictors

Changes from Version 0.15-19 to 0.15-21 [08-Jan-2014]

- Addressed issues with certain non-package files being included via use of `.Rbuildignore`
- NOMAD team corrected dereferencing null pointer in `Mads.cpp` that was uncovered through the use of UB sanitizer checks with clang 3.4

Changes from Version 0.15-18 to 0.15-19 [30-Dec-2013]

- Updated snomadr to NOMAD 3.6.2 (released December 2013) which resolves crash/incompatibility with the clang compiler on Mac OS X Mavericks 10.9.1
- Added new function `clsd()` that does logspline regression jointly choosing the degree and number of segments (knots) - this is to be treated as beta but the univariate continuous only case currently implemented is capable of outperforming existing logspline methods that set the spline degree to an ad hoc value (3)

- `npglpreg()` now optimizes at the level of the bandwidth scaling factors (thanks to Sebastien Le Digabel for the suggestion) to bring parameters to a 'common scale'
- For cross-validation with `npglpreg()` we test for very large bandwidths for the continuous predictors (> bandwidth.switch robust standard deviations) then switch to the global polynomial approximation for computation speed (produces identical results but uses global least squares rather than local least squares). This can result in substantial improvements when large bandwidths are appropriate. Combined with trees (in progress in np) this could lead to marked reductions in computation time with zero loss in accuracy

Changes from Version 0.15-17 to 0.15-18 [29-Dec-2012]

- Using .onUnload rather than .Last.lib in zzz.R
- Stopping rules for crsiv and crsivderiv modified
- Added options to smooth residuals for stopping rule in crsiv (i.e. smooth y-phi(z) w as opposed to separately smoothing y w and phi(z) w)
- Added options to input starting values for crsiv and crsivderiv

Changes from Version 0.15-16 to 0.15-17 [04-Jun-2012]

- Updated NOMAD from 3.5.0 (released Jan 2011) to 3.5.1 (released Mar 2012)

Changes from Version 0.15-15 to 0.15-16 [30-Apr-2012]

- Fixed bug when `deriv=` is used in `crs()` (derivatives for categorical variables were not correct in some cases - derivatives computed by plot and plot-data were correct however)
- Fixed issue when plot called with `"plot-data"` or `"data"` and `par(mfrow())` is not reset
- Startup message points to the faq, faq is now a vignette
- Glitch fixed in `npglpreg()` where kernel types were not being passed to the NOMAD solver...
- Added ckerorder option to `npglpreg()`
- More information output by summary for `npglpreg()` objects (kernel type, order etc.)
- tol for determining ill-conditioned bases now consistent with Octave/Matlab etc. and uses the tolerance `max(dim(x))*max(sqrt(abs(e)))*.Machine$double.eps`

Changes from Version 0.15-14 to 0.15-15 [16-Apr-2012]

- NOTE: ** major change in default setting ** Default basis set to `"auto"` from `"additive"` for more robust results when using default settings with more than one continuous predictor (at the cost of additional computation for all supported bases, more memory required). Warnings changed to reflect this, additional warnings provided, but mainly when `cv="none"` (i.e. user is doing things manually and failing to provide defaults)
- Added switch for discarding singular bases during cross-validation (`singular.ok=FALSE` is default, so default is to discard singular bases)
- More stringent checking for ill-conditioned bases during cross-validation - previous checks were insufficient allowing poorly conditioned bases to creep into the final model (this may result in increased runtime for large datasets and multivariate tensor models)
- Potential runtime improvement for large bases (test for ill-conditioned bases was relying partly on `rcond(t(B)%*%B)`, now using crossprod(B) which is more computationally efficient than `rcond(t(B)%*%B)` and has a substantially smaller memory footprint)
- No longer report basis type for one continuous predictor as all bases are identical in this case

Changes from Version 0.15-13 to 0.15-14 [22-Mar-2012]

- Added support for weights to `crs()`
- Corrected issue with reporting of cv function value by `summary()` with quantile regression splines
- Test for coexistence of pruning and quantile regression splines and stop with a message that these cannot coexist
- Check for valid derivative integer

Changes from Version 0.15-12 to 0.15-13 [05-Mar-2012]

- Added new function `crsivderiv()`
- More options added to `crsiv()`
- Modified vignette (`spline_primer`)
- Corrected error in demo for IV regression when exhaustive search was selected (nmulti needed to be set though it is not used)

- Changes to code to improve compliance with R 'Writing portable packages' guidelines and correct partial argument matches

Changes from Version 0.15-11 to 0.15-12 [7-Dec-2011]

- Added option to limit the minimum degrees of freedom when conducting cross validation (see `cv.df.min` which defaults to 1) which can save memory and computation time for large sample sizes by avoiding computation of potentially very large dimensioned spline bases

Changes from Version 0.15-10 to 0.15-11 [3-Dec-2011]

- Fixed glitch in `bwtype="auto"` when `adaptive_nn` is selected

Changes from Version 0.15-9 to 0.15-10 [3-Dec-2011]

- Fixed regression in code when using `cv.func=cv.aic`
- Added option `bwtype="auto"` for `npglpreg` (automatically determine the bandwidth type via cross-validation)

Changes from Version 0.15-8 to 0.15-9 [25-Nov-2011]

- Fixed glitch when computing/plotting derivatives with `crs` and multiple predictors are of degree zero
- Added support for parallel `npglpreg` (calls `npRmpi` rather than `np` - see demo)
- Improved handling of complex bases via dim.bs (test for negative degrees of freedom prior to attempting to construct the basis function - dramatically reduces memory overhead and can cut down on unnecessary computation)

Changes from Version 0.15-7 to 0.15-8 [16-Nov-2011]

- Added support for `is.fullrank` testing for generalized local polynomial kernel regression for cross-validation (default remains ridging a la Seifert & Gasser), replaced `rcond(t(X)%*%X)` with `is.fullrank(X)` (smaller memory footprint for large datasets)
- Fixed glitch with derivative computation when one or more `degree` is 0 when `kernel=TRUE` and additionally when one or more factors is excluded when using `kernel=FALSE`

- Fixed issue with reported cross-validation score only corresponding to leave-one-out cross-validation by passing back cv function from solver rather than computing post estimation
- Added ability to estimate quantile regression splines
- More rigorous testing for rank deficient fit via `rcond()` in cv function
- Fixed issue where `degree.min` was set > 1 but initial degree was 1 (corresponding to the linear model which is the default for the initial degree otherwise)
- Added the option to treat the continuous bandwidths as discrete with `lambda.discrete.num+1` values in the range [0,1] which can be more computationally efficient when a 'quick and dirty' solution is sufficient rather than conducting mixed integer search treating the lambda as real-valued
- Corrected incorrect warning about using `basis="auto"` when there was only one continuous predictor

Changes from Version 0.15-6 to 0.15-7 [24-Oct-2011]

- Added logical `model.return` to `crs` (default `model.return=FALSE`) which previously returned a list of models corresponding to each unique combination of the categorical predictors when `kernel=TRUE` (the memory footprint could be potentially very large so this allows the user to generate this list if so needed)

Changes from Version 0.15-5 to 0.15-6 [17-Oct-2011]

- Compiler error thrown on some systems due to changes in `Eval_Point.hpp` corrected

Changes from Version 0.15-4 to 0.15-5 [16-Oct-2011]

- Thanks to Professor Brian Ripley, additional Solaris C/C++ compiler warnings/issues have been resolved
- Some internal changes for soon to be deprecated functionality (`sd(<matrix>)`) expected to be deprecated shortly)

Changes from Version 0.15-3 to 0.15-4 [15-Oct-2011]

- Extended the `gsl.bs` functionality to permit out-of-sample prediction of the spline basis and its derivatives
- Added option `knots="auto"` to automatically determine via cross-validation whether to use quantile or uniform knots
- Minor changes to help page examples and descriptions and to the crs vignette

Changes from Version 0.15-2 to 0.15-3 [05-Sept-2011]

- Fixed glitch when all degrees are zero when computing the cross-validation function (also fixes glitch when all degrees are zero when plotting the partial surfaces)
- Added new function `crssigtest` (to be considered in beta status until further notice)
- Added F test for no effect (joint test of significance) in `crs summary`
- Both degree and segments now set to one for first multistart in `crs` (previously only degree was, but intent was always to begin from a linear model (with interactions where appropriate) so this glitch is corrected)
- Test for pathological case in npglpreg when initializing bandwidths where `IQR` is zero but `sd > 0` (for setting robust sd) which occurs when there exist many repeated values for a continuous predictor
- Added 'typical usage' preformatted illustrations for docs

Changes from Version 0.15-1 to 0.15-2 [30-July-2011]

- Renamed COPYING file to COPYRIGHTS

Changes from Version 0.15-0 to 0.15-1 [29-Jul-2011]

- Automated detection of ordered/unordered factors implemented
- Initial degree values set to 1 when conducting NOMAD search (only for initial, when `nmulti > 1` random valid values are generated)
- Multiple tests for well-conditioned B-spline bases, dynamic modification of search boundaries when ill-conditioned bases are detected, and detection of non-positive degrees of freedom and full column rank of the spline basis (otherwise the penalty `sqrt(.Machine$double.xmax)` is returned during search) - this can lead to a significant reduction in the memory footprint
- Added support for generalized B-spline kernel bases (varying order generalized polynomial)
- Corrected issue with plot when variables were cast as `factor` in the model formula
- Fixed glitch with return object and i/o when `cv="bandwidth"` and `degree=c(0,0,...,0)`
- Added tests for pathological cases (e.g. optimize degree and knots but set max degree to min degree or max segments to min hence no search possible).
- Added argument `cv.threshold` that uses exhaustive search for simple cases where the number of objective function evaluations is less than `cv.threshold` (currently set to 1000 but user can set). Naturally exhaustive search is always preferred but often unfeasible, so when it is feasible use it.

- Added additional demos for constrained estimation (Du, Parmeter, and Racine (2011)), inference, and a sine-based function.
- Substantial reductions in run-time realized.
  - Product kernel computation modified for improved run-time of kernel-based cross-validation and estimation.
  - Moved from `lsfit` to `lm.fit` and from `lm` to `lm.wfit`/`lm.fit` in `cv.kernel.spline` and `cv.factor.spline` (compute objective function values). Two effects - R devel indicates `lm.fit`/`lm.wfit` are more robust (confirmed for large number of predictors) and much faster `cv.kernel.spline` function emerges (run-time cut 20-30%).
  - The combined effects of these changes are noticeable. For instance, run-time for `wage1` with 7 predictor cross-validation goes from 510 seconds in 0.15-0 to 304 seconds due to use of `lm.fit`/`lm.wfit` described below to 148 seconds due to the modified kernel function.

Changes from Version 0.14-9 to 0.15-0 [23-Jun-2011]

- Thanks to Professor Brian Ripley, compile on Solaris system issues are resolved, and check/examples are reduced in run time to alleviate the excessive check times by the R development team. Many thanks to them for their patience and guidance.
- Minor changes to `radial_rgl` demo

Changes from Version 0.14-8 to 0.14-9 [20-Jun-2011]

- Cleaned up issues for creating binary for windows
- Setting seed in `snomadr.cpp` via `snomadr.R` for starting points when `nmulti > 0`
- Increased default `MAX_BB_EVAL` from 500 to 10000 (makes a difference for difficult problems) and modified default EPSILON in NOMAD along with other parameters (`MIN_MESH_SIZE`, `MIN_POLL_SIZE`) to reflect actual machine precision (using R's `.Machine$double.eps` where `NOMAD` fixed EPSILON at 1e-13)
- Zhenghua added help functionality for retrieving help via snomadr
- Now default number of restarts in `crs` is 5 (zero is not reliable and I want sensible defaults in this package - higher is better but for many problems this ought to suffice)
- Corrected glitches in interactive demos where options were not being passed, updated docs to reflect demos

Changes from Version 0.14-7 to 0.14-8 [10-Jun-2011]

- `crsiv` now returns a crs model object that supports residuals, fitted, predict and other generic functions. Note that this approach is based on first computing the model via regularization and then feeding a transformed response to a `crs` model object. You can test how close the two approaches are to one another by comparing `model$phihat` with `fitted(model)` via

```
all.equal(as.numeric(fitted(model)),as.numeric(model$phihat))
```

Version 0.14-7 [09-Jun-2011]

- Initial release of the `crs` package on CRAN.