Ten years of vegan: Looking back

Jari Oksanen

Abstract

The first version of **vegan** was released on September 6, 2001, almost exactly ten years ago. In this paper I tell about some of the early stages of **vegan** history to see where we come from and how **vegan** took its shape. This document discusses mainly events from 2001 to 2005.

Keywords: vegan, history.

Version 1.0-1 (September 6, 2001)

The first CRAN version of vegan was dated on September 5, 2001, but it appeared in CRAN one day later. R version 1.3.1 was released some days earlier, and its `R-FAQ` listed about 120 extension packages (now there are thousands) making **vegan** one of the earliest contributed CRAN packages.

The first release contained only the following functions: `decostand`, `diversity`, `initMDS`, `postMDS`, `procrustes`, `rankindex`, `vectorfit`, `vegdist`, `wascores` and `wisconsin`. Functions `initMDS` and `wisconsin` are living fossils that have survived intact to the latest version (but when I look at the `initMDS` I see that it should be changed). Function `diversity` was unchanged till early 2011 when it was first edited stylistically, and then made more graceful to users supplying their data in other forms than the function expects.

The 1.0-1 release was not a completely random collection of helper functions, but really designed to fill a need in community ecology. Several ecologists (I included) had used Peter Minchin's DECODA software for non-metric multidimensional scaling (NMDS) for years, but that software was somewhat difficult to get when the distributor changed after Peter Minchin moved from Australia to United States. Moreover, it was only available for Windows, and I had switched to Linux a couple of years earlier. Minchin's DECODA was the only piece of software that ran NMDS like it should be run: using adequate dissimilarity measures, and comparing several random starts to find a stable global optimum. The **MASS** package of R had function `isoMDS` to run NMDS, and the purpose of **vegan** was to add the missing pieces to have DECODA-in-R:

- Function `decostand` (= DECODA standardization) provided standardization methods that were found useful with community data. Most useful of these, the Wisconsin double standardization could be run as a single command `wisconsin`.

- R function `dist` (which was still in the extension package **mva**) calculated several dissimilarity indices, and `vegdist` complemented those with several that were known to

be good for community data. The first **vegan** version had Manhattan, Euclidean (duplicating `dist`), Canberra, Bray-Curtis, Kulczyński and Gower dissimilarities. A tiny utility function `rankindex` could be used to compare these indices for their ability of gradient separation.

- NMDS was in **MASS**, but its default was to start NMDS from the solution of principal coordinates analysis (PCoA, function `cmdscale`). Minchin had demonstrated that this was dangerous because it was easily trapped into a local optimum, and therefore **vegan** had function `initMDS` to generate random starting configurations.
- Random solutions had to be compared for stability, and for that purpose we needed Procrustes analysis (function `procrustes`)
- `isoMDS` results were indeterminate as to rotation, scaling and centring, but **vegan** added function `postMDS` that could remove some of these indeterminacies: it could centre the solution, rotate axes to principal components and scale them to half-change units (which was a DECODA innovation).
- Function `wascores` could be used to add species scores to NMDS ordination to silence those people who claimed that you cannot have species results in NMDS.
- Continuous environmental variables could be fitted to the ordination using function `vectorfit`.
- The only function that did not fit suite was `diversity`. I was a bit ashamed of writing something so trivial for a task that anybody could perform without special functions, but I thought that this small piece would not be too disturbing.

When I prepared this DECODA-in-R package, I was in active correspondence with Peter Minchin. We discussed the choice of standardization methods (`decostand`) and dissimilarity indices (`vegdist`). Peter also explained how he fitted the environmental vectors following a Bell Labs memorandum, and how he performs the half-change scaling. Some of this discussion also radiated to `ORDNEWS` email list in 2001 (I am currently blacklisted, and cannot post to `ORDNEWS` any longer).

Originally the idea was not to have every possible functionality in **vegan**, but only the recommended features. I started to slip from this moral principle already in the next release, but the intention was good. Therefore `vegdist` did not contain every popular dissimilarity index, but it only contained those that Peter Minchin and others had found to be good for community separation. The function was also geared to NMDS, and therefore it had no Jaccard index: it is rank-order identical to Bray-Curtis, and therefore does not make a difference in NMDS. It is also possible to calculate Jaccard dissimilarity (J) from the Bray-Curtis (B) as $J = 2B/(1 + B)$, and I thought anybody could do this. Further, I did not have any separate indices for binary and quantitative data, because anybody could use `decostand` to transform data to binary before using `vegdist`. I added argument for binary variants in release 1.6-5 (October 12, 2004) after I got irritated reading on the web that **vegan** does not have a certain binary index. I was also careless with names: it still irritates me that many people think that the same index (like I see it) should have a different name when it is expressed as similarity or dissimilarity or binary or quantitative. Later, function `designdist` (introduced in version 1.8-6, May 9, 2007) expresses the same philosophy by separating index formula

(with a name) and the method of evaluating its terms either as products, parallel minima or presence/absence.

The show-stopper function to the **vegan** release was Procrustes rotation. All other things looked pretty simple to implement, but I had never implemented Procrustes rotation. When preparing a paper with Peter Minchin, I had used his FORTRAN code that was over 200 lines, and I was not sure how to put that in R. I started to implement `procrustes` directly following Mardia's book on multivariate analysis, and then I really saw the power of R: it did not take but some sixteen lines of code! After I succeeded with `procrustes`, I knew that I could have **vegan**, and started to work with other functions. The next task was to implement `vectorfit`. The algorithm that Peter Minchin communicated to me involved inverse of a matrix crossproduct, and I knew from my experience with numerical methods that this is something that should be avoided (*motto*: if you need a matrix inverse, find another algorithm). Doing this via QR decomposition was an easy thing to do in R. Function `vegdist` used C code, but I had implemented dissimilarities so many times that I could simply recycle my old code and copy R sources (thanks to GPL).

Although the first release looked like being a minimal collection of functions, there really were some design principles from the beginning. The most important one was the Unix toolbox idea: make small functions that do one thing, and communicate with other functions. Output of a function can be used as an input in another. It was also important to be a well-behaving R citizen: **vegan** should co-operate with R. Therefore `vegdist` is a plug-in alternative to `dist`. This principle paid off when **vegan** grew in size. I do still get irritated when I see papers, manuscripts, scripts or web pages with huge functions that incorporate, say, data transformation, dissimilarity calculation and analysis in one monolithic body, when all these should be separate so that users could change any of these or use any part in their own analysis. My aversion to long code helped also: I prefer to have function that fits Emacs screen so that you can see the whole function at a glance.

A corollary of the toolbox idea is that everything should be visible to the user. Users should be able to combine small functions in unexpected ways and to see how these functions really work. My idea was that by seeing the functions people would gain a better understanding of the methodology: I could expose the banality of fancy methods and destroy the surrounding mystery. Most people are not interested in computational details, but I think I have seen some hints that **vegan** solutions have been used in other packages which would show that I have reached that design goal.

Version 1.2.0 (November 19, 2001)

I left the moral high ground of **vegan** 1.0-1 two months later, when I ported Mark Hill's DECORANA software to **vegan** as function `decorana`. I had never been a fan of detrended correspondence analysis (DCA), but I felt that it is useful to compare NMDS results against DCA. However, this expanded ordination methods beyond the NMDS turf.

The package also developed: S3 methods like `print`, `summary` and `plot` were introduced both to `decorana` and to `procrustes` of earlier version. This was the first release that started to look like a standard R package.

Version 1.4-0 (May 3, 2002)

Version 1.4.0 was an important major release which introduced function `cca` for constrained correspondence analysis (CCA). This was to become the major feature for many users, and with its associated methods, it makes the greatest bulk of current **vegan**.

Actually, CCA was already available in R as function `CAIV` in Stéphane Dray's **CoCoAn** package (1.0-1 released on Feb 27, 2001 and 1.0-2 on Mar 22, 2002). I wanted to have some features that were missing in his implementation:

- Formula interface: it was important to be able to easily define a model instead of using all variables you happen to have. Model selection was possible with the matrix input, but that is not really encouraged with that interface. Further, with formula interface it is easier to have factor variables as constraints, since users do not need to construct manually their design matrices.
- I wanted to have WA scores in addition to LC scores that Dray's function returned. In addition to the long tradition of having both, I was also impressed by Bruce McCune's analysis that favoured WA scores.
- I wanted to have partial CCA.
- I wanted to have residual unconstrained ordination after constrained (and optional partial) components.
- I wanted to implement this using algorithm building upon Pierre Legendre's formulation of CCA in matrix algebra.

The last point may be the least interesting to a user, but it was one of the driving forces that made me to write this function. When I first saw Pierre Legendre's matrix equations for CCA (or actually, he gave them primarily for redundancy analysis, RDA), I just could not believe my eyes: can this really be as trivial as fitting separate linear regression models for each species, and then subjecting fitted values to correspondence analysis? It really was. About this time I also found Dave Roberts's web pages on ordination where he used S-PLUS, and implemented RDA directly using linear regression with the most standard function `lm`. Pierre Legendre's algebra contained matrix inverses, and my duty was to get rid off those. Replacing linear regression with QR decomposition and weighted PCA with singular value decomposition (with weighting) did the trick. Then I only had to figure out how constrained analysis is really done after partialling out some terms. With functions in modern **vegan**, the constrained part of partial CCA

```
cca(Y ~ X + Condition(Z))
```

really is equivalent to

```
cca(residuals(cca(Y ~ Z)) ~ X + Z)
```

I first implemented only CCA, because Peter Minchin's simulations had demonstrated that in unconstrained ordination CA is uniformly better than PCA: with long gradients it is better, and with short gradients at least as good as PCA. This leaves no niche to RDA which is

based on PCA. However, scientific folklore ignored published results and argued that with short gradients you should use PCA and RDA. The user community did not agree with me, and after several inquiries, I gave up and added function `rda` in minor release 1.4-2 (February 7, 2003). Currently I think I was wrong in opposing RDA so fervently, but I think my opponents were also wrong: The argument of unimodality really concerns unconstrained analysis, but RDA and CCA are ordinations of fitted values from linear regression so that it does not matter which one you use – both are linear. CCA is not more unimodal than RDA, since both analyse linear fits. The difference may only crop out when you use a large number of constraints which allow approximating non-linear models with several linear terms, and releases the analysis from constraints. That is not encouraged in **vegan**: that is why there is a formula interface.

Permutation tests were also introduced in 1.4-0, but permutation by strata appeared in first minor upgrade 1.4-1 (October 4, 2002). This was copied from the **boot** package, and still is the only kind of restricted permutation method in **vegan** until Gavin Simpson releases his **permute** package.

Less conspicuous but hugely important move was to introduce generic `scores` function. I was getting familiar with Dave Roberts's **labdsv** package (which was still unreleased, but on his web page; the first release was on January 18, 2005). Several classical multivariate methods in R did not have a `class` attribute, and therefore no `S3` methods (`plot` etc.) could not be constructed for them (including `isoMDS` of **MASS**). Dave Roberts solved this by having wrapper functions that mainly called the basic function and added a `class` attribute, and some extras. I did not like this, mainly because it hides the real working function, but I felt the need for something similar. As a solution, I wrote `scores` function that accesses the ordination scores and made other functions (such as `plot`) to always use `scores` and never access the results directly (`plot` function should not know where the analysis keeps its results). The `default` method guesses where several classical methods keep their results, and can find them even if there is no `class` attribute. This also means that if you supply a `scores` method, several **vegan** support functions become immediately available and magically work. As a general test case, `ordiplot` was introduced in **vegan** 1.4-2. This was a one case of pluggability, or the toolbox philosophy that makes maintenance of **vegan** much easier.

With `cca` and `rda` (and later with `capscale`), the scaling of results also happens only when you access them with `scores`. Therefore the user does not need to decide how to scale the results during the analysis. The results are like a Schrödinger cat: it only exists (dead or alive) when you look at it.

Dave Roberts's **labdsv** package had much influence on **vegan**. I tried to avoid overlap with **labdsv** and make **vegan** co-operate smoothly with **labdsv**. In addition, I shamelessly stole ideas from Dave. One of these was `ordisurf` (called `surf` at the first release) to fit smooth surfaces for environmental variables instead of vector arrows.

Version 1.6-* (from October 20, 2003)

With NMDS support, CCA and RDA **vegan** catered for many tastes of ordination analysis, and started to gain momentum. Queries and suggestions from users became almost daily. Two important persons at this stage were Roeland Kindt (whom I have never met in person) and Bob O'Hara (whom I met when I held a course at the Helsinki University). They wanted

to expand **vegan** to new territories that were less explored in **vegan**. This work resulted in the next major release 1.6-0 with code name Fisher: the most new features concerned diversity analysis. Roeland Kindt and Bob O'Hara were acknowledged in function documentation and `ChangeLog`, but formally they were added as **vegan** authors only in version 1.6-8 (April 18, 2005). In fact, they had this role from the first 1.6-0 release, and the recognition in the package header was belated.

Roeland Kindt wrote functions on Rényi diversities and Hill numbers, and species accumulation models, and contributed to several other functions, and Bob O'Hara on extrapolated richness and diversity functions. Inspired by their interest, I also started to implement some related methods (another reason was work as an editor of *Journal of Vegetation Science* and editorial task where I had to inspect diversity problems). This yielded in implementing Bastow Wilson's abundance distribution models on Whittaker plots (but generalized to use standard R error distribution families), and fitting Fisher's and Preston's abundance distribution models, and models for extrapolated richness (`specpool`). The 1.6-* series really was a diversity analysis expansion of **vegan**. This in turn attracted later development, in particular after Péter Sólymos joined the **vegan** team.

Another driving force in **vegan** development were user queries. I had never used Primer software, but it was really popular in some fields of ecology. Several people asked me how to implement certain methods in R, and if I was pushed enough I could add a function to **vegan**. Perhaps first such a case was `bioenv` (`anosim` and `mantel` already appeared in 1.4-0, but they were spontaneous instead of being user-induced).

Version 1.6-0 also expanded and improved ordination analyses. Distance-based RDA (dbRDA) was introduced as function `capscale`. The choice of the name was a bit unfortunate, but it has historical reasons: I started to work with Marti Anderson's recently published paper on constrained analysis of proximities (CAP). She had some methodological choices that I did not like, and I changed those points, and actually came back to older method of dbRDA. All this was documented in the help pages, but still caused confusion. Another new feature was adding extended or `stepacross` dissimilarities. I had analysed some difficult and heterogeneous data sets that were not properly ordinated in NMDS, but I got sausage or pretzel shaped configurations. This happened because `isoMDS` did not handle tied maximum dissimilarities (of value 1) adequately. I could not change `isoMDS` so I decided to implement extended dissimilarities to "break" the ties before the analysis. **Vegan** 2 will introduce Minchin's `monoMDS` which can break ties, and usually does not need `stepacross`, but at that time this was a crucial feature of successful NMDS. As a collateral damage, I also implemented perhaps the fastest minimum spanning tree in R: compare it with other alternatives with, say, 100 000 points. It just appeared when I was trying to change Sedgwick's *Algorithms in C* to a working C code.

A cycle in **vegan** development was closed in 1.6-5 (October 13, 2004) when `metaMDS` was introduced to wrap most of the functions of the first **vegan** release. Now `initMDS` and `postMDS` are almost unknown to most users, and many other founding functions nearly invisible. Version 2.0-0 (to be released soon) will even replace `isoMDS` with a better **vegan** alternative `monoMDS` written by Peter Minchin.

Vegan gained its adolescence with 1.6-* series, and started to develop as an ever-expanding collection of functions in 1.8-1 (released October 12, 2006), or nearly five years ago. Its user base and authorship expanded. Pierre Legendre was the next new author. Four years ago, on September 6, 2007 **vegan** was registered to R-Forge, and from that on most of the current

authors have been with the development, and history can be also tracked from `vegan-devel` mailing list and other documentation in R-Forge. Actually, **vegan** development is faster now than it was in early steps. However, I feel that these times are so close to us that it is too early to start to remember them. Perhaps later (when I'm older), perhaps someone else.

Affiliation:

Department of Biology, University of Oulu, Finland