Cleaning the `mattpkg` package

Matt Nunes

19th April 2010

There were 5 errors in the files for `mattpkg` which needed to be solved before the package would pass all the `R CMD check` tests. I chose to break the package in those 5 ways because they are the ones which come up most frequently in practice.

1. S3 method inconsistency

If you define and use any S3 (or for that matter, S4) methods/generics, they have to conform, in arguments, to any other existing method from that generic stem, i.e. all plot methods have to have the same arguments as each other, all print methods have to have the same argument list etc. The `print.cubic` method (for class cubic) fails since the standard `print` functions have the argument list of `x,...`, not `cubic,...`. The corresponding *.Rd* file has the same problem, since they are often created using a `package.skeleton` call. In other words, to solve this, the *R* code for `print.cubic` has to be changed to take `x` as its first argument, and the arguments/usage fields in the documentation file changed accordingly as well.

2. *DESCRIPTION* **file**

In the task description, I mentioned that the package depended on both `roots` and `secondpkg`. This is omitted in the `Depends` field of the `mattpkg` *DESCRIPTION* file. This causes the `R CMD check` to say *no visible global function definition for ...*, basically because the function `solve.cubic` tries to use a function which doesn't exists in *R*'s eyes. Simply put the dependency back in the *DESCRIPTION* file.

3. *global 'T' or 'F'*

This is one of the most annoying ones. `R CMD check` cannot cope with 'T' or 'F' (instead of TRUE or FALSE) in function code, since it treats them as unknown variables. All boolean variables, especially in argument lists (but in general function code as well) must have the explicit values of TRUE or FALSE.

4. *doc/codoc mismatch*

This one is a simple mistake: the argument list of the *R* code of `plotngon` doesn't match what appears in its *.Rd* file. In this case, the ... argument is missing from the documentation. It needs to be inserted as an `item` after `r` in the argument list.

5. Example/C code dll error

`R CMD check` has a problem here because (for some reason) it can't find the C shared object which is called by `checkngon` (*mattpkg.so*). In other words, the library hasn't been loaded when the example was run. Since the package has compiled code, this needs to be done at the start of the *R* session. Since this type of issue is dealt with in *zzz.R*, there should be a *library.dynam* call to load the shared object (I have left the call in but it has been commented out). Uncomment the call to solve the problem.

After all these errors have been corrected, the package should (build and) check OK.