Exercises for the lecture „Moderne Methoden der Datenanalyse”

Prof. Dr. S. Hansmann-Menzemer, M. Schiller
Physikalisches Institut der Universität Heidelberg

Apr 27 2010

Some useful *nix commands

We will be using the command line of a Linux system quite frequently, so here is some material to get you started:

Navigating directories

Files are organised in directories (some also call them folders) which can contain files and also nested (sub-)directories. To have the shell print your present working directory, type `pwd`. The system will respond with something like

```
/home/fphys/mschill4
```

The leading `/` tells you that an absolute path is given (i.e. starting at the root of the directory hierarchy). Your working directory is `mschill4` which is a subdirectory inside `fphys4` which is a subdirectory inside `home` which is a directory in the root directory. Observe how the components of path to your present working directory are separated by slashes („`/`”).

Changing the working directory can be done with `cd [path]`, where path represents either an absolute path (like the one `pwd` returned) or a relative path (one without the leading slash, the path is then interpreted relative to your present working directory). By convention, the relative path of `.` represents the present working directory itself, `..` represents its parent directory. If you use `cd` without a path argument, it takes you to your home directory.

Directories can be created with `mkdir`, and, if empty, they can be removed with `rmdir`.

You can get a listing of the contents of a directory by typing `ls`. You can optionally specify the path to a directory which you want listed, if you don't, you get a listing of your current working directory. To get a little more information, you can also type `ls -l`, thus specifying the long listing format. Its output looks like this:

```
total 28
-rw-r--r-- 1 mala users  4226 2009-04-04 18:59 ex01.tex
-rw-r--r-- 1 mala users  1477 2009-04-03 16:51 Makefile
-rw-r--r-- 1 mala users  3191 2009-04-04 18:59 preamb.tex
drwxr-xr-x 2 mala users     6 2009-04-04 19:24 src
```

The first line gives the total size of all files in the directory in units of 1024 bytes, then the different entries are listed.

The first column specifies the access rights of a file: The first character is either `-` for a normal file or `d` for a directory. The following characters in this column represent the access rights for the user, members of his group and others (each gets the three characters `rwx` for read, write and execute permissions, or a `-` if access is not allowed). For example, the first file in the listing is readable and writeable by the user, and readable by members of the same group and others.

We'll ignore the second column. The third column gives the owner of the file, the fourth his or her group. Next, we have the size of the file in bytes, the date and time of the last modification and, finally, its name.

Working with files

Now that you know how to navigate the directory hierarchy and list its contents, we'll briefly have a look how to inspect files, move, copy or delete them.

Text files can easily be viewed with `less somefile`[1], moving around inside `less` can be done with the cursor keys, PageUp, PageDown etc. You leave `less` with `q` (like quit); searching is possible by first pressing `/`, followed by what you want to search for.

If you are not sure if some file is a text file, you can classify its contents with `file somefile` which will then print what kind of file you have, (e.g. a C++ header file, an e-mail message, a jpeg picture, ...) .

To copy a file, use `cp sourcefile destination` where destination can either be another file name or a directory. In the latter case, the file will be copied to the destination directory while retaining its name. Moving or renaming files works essentially the same way, instead of `cp` you use `mv` (for „move”).

To remove a file, use `rm somefile`.

To create a text file (e.g. a macro you're writing), you can use a text editor. Emacs is popular, you can start it for example with `emacs mymacro.C` (if that is the file you want to edit, otherwise, just type emacs and you can start from scratch). Usually, it will open a new window on you X terminal and block your shell window until you exit emacs. You can avoid that by putting emacs into the background, so that you can continue working with the shell:
`emacs mymacro.C &` (Note the `&` at the end of the command!)

Getting help

To get help, you can have a look at the manual pages. For example, try `man ls` to get the manual page of the `ls` command. `man` usually uses `less` to show you the manual page, so you already know how to navigate. Manual pages are good if you already have an idea on how to do a thing, but can't remember the correct syntax. To get an introduction to a new program, you can often have a look at the info pages that come with that program (most GNU programs have one). Type `info` to enter the info hypertext system. Typing `q` should get you back to your shell again.

[1] There is also a text file viewer called `more`; you may want to try it to see why `less` is sometimes `more`, only better...

Summary of the most important *nix commands

Moving around and inspecting directories

print present working directory	`pwd`
change directory	`cd` *path*
make directory	`mkdir` *path*
remove directory	`rmdir` *path*
list directory contents	`ls` *[-l] [path]*

Working with files

copy a file	`cp` *source destination*
move or rename a file	`mv` *source destination*
remove a file	`rm path`
view text file contents	`less` *path*
classify file	`file` *path*
edit file	`emacs` *[path]*

Getting help

view manual page	`man` *[section number] topic*
view info pages	`info`

Wildcards

When specifying file or directory names, one often wants to treat a group of them alike (e.g. copy all files ending in „`.C`” to a different directory). For this purpose, the shell offers so-called wildcards which facilitate these tasks.

`?`	any single character
`*`	arbitrary combination of zero or more characters
`[`*characters*`]`	any of the specified characters, e.g. `[ab42]` matches `a`, `b`, `4` and `2`, `[a-zA-Z0-9]` any letter or number