

# E-mail and e-mail merge from R

Ben Bolker

April 8, 2011

## 1 Introduction

The `Rmail` package is a tiny package designed for sending e-mail directly from within R, and for mail merges. R is probably not the best platform, but it was the one I knew.

It has the following properties (advantages/disadvantages):

- implemented entirely within R — this may be an advantage if you already have R installed etc., as there is nothing else to install, no need to install any additional software, etc..
- connects directly with SMTP server — efficient, and (as above) no need for additional software
- allows authentication (but not encryption)
- purely command-line — no GUI interface (except for password prompt, which works via Tcl/Tk)
- does not handle attachments

## 2 Example

Here is an extended example of `Rmail`'s use.

*If you haven't done it yet, install the package (R CMD INSTALL Rmail\_1.0.tar.gz from the command line or go to the Packages menu and "Install from local zip file") (to do: add Rmail to my local repository)*

Now load the package:

```
> library(Rmail)
+
Read in data:
> csvfile <- system.file("doc", "ex1.csv", package="Rmail")
+
> gdat <- as.matrix(read.csv(csvfile))
```

`read.csv` reads a comma-separated file; `as.matrix` turns the resulting data frame into a matrix of character strings.

Look at the first few lines:

```
> head(gdat)
+
      Last.name  First.name Co.authors
[1,] "Bloggs"   "Joe"      "Fred Smith"
[2,] "Roe"      "Susan"    "A. N. Other"
[3,] "Gladumovka" "Natalia" "Y. A. N. Other"
      Title
[1,] "The evolution of cooperation by mutual regard"
[2,] "Chaos in a long-term experiment with a plankton community"
[3,] "Foraging-predation risk tradeoff governs evolution of inducible defenses"
      Talk.Poster Session
[1,] "Talk"      "New approaches to the evolution of social behavior"
[2,] "Talk"      "Food Webs"
[3,] "Poster"    "Life history theory"
      Date      Time      Place      e.mail
[1,] "2008/08/05" "10:00:00" "104B MAC" "bloggs@somewhere.com"
[2,] "2008/08/04" "13:30:00" "COS 6-1"  "roe@somewhere.edu"
[3,] "2008/08/05" "5-6:30 PM" "MAC Exhibit hall" "bl@xy.ed.ac.uk"
      judge      jfirst  jemail
[1,] "Karl Gauss" "Karl"  "gauss@wherever"
[2,] "Alfred Lotka" "Al"    "lotka@somewhere"
[3,] "Evelyn Hutchinson" "Evelyn" "hutch@hutch.com"
```

The `judge` (judge's full name), `jfirst` (judge's first name), and `jemail` (judge's e-mail address) are the most important columns here.

Here's our mail message, stored separately as a file in the current working directory (use `getwd()/setwd()` to get or set the working directory, or use the File menu on Windows): in this case it is stored as `ex1-msg.txt`.

```
Subject: competition judging
Reply-to: joe@joeshouse.com
```

Dear #jfirst#,

Thanks for agreeing to judge a presentation ...

Here's your assignment:

```
Presenter: #Last.name#, #First.name#
Co-authors: #Co.authors#
Title: #Title#
Presentation type: #Talk.Poster#
```

Session: #Session#  
Date: #Date#  
Time: #Time#  
Location: #Place#

The judging form is available at <http://...> ; I will also put some copies in an envelope on the bulletin board at the meeting.

Please return completed forms to me at the meeting, or put them back in the envelope, or e-mail me electronic copies.

Please feel free to contact me with any questions -- especially, check your assignment and let me know if I made a mistake.

cheers  
Joe

Run a fake mail-merge:

```
> msgfile <- system.file("doc","ex1-msg.txt",package="Rmail")  
+  
> mailmerge(data=gdat,  
+           msgfile=msgfile,  
+           id="judge",  
+           email="jemail",  
+           mail=FALSE,  
+           auth=FALSE, ## no authorization  
+           user="bolker",  
+           from="bolker@ufl.edu",  
+           host="bolker-laptop@ufl.edu",  
+           server="smtp.somewhere.edu",  
+           port=587,  
+           fake=TRUE,  
+           trace=2)
```

Karl Gauss  
Alfred Lotka  
Evelyn Hutchinson

- `auth` specifies whether you need authorization (i.e., a mail password) to send e-mail via the specified route
- `user` is your username on your mail system
- `port` is 25 by default (typically works if you are sending mail to an “open relay”, e.g. from work to a work e-mail), but 587 is commonly used as a more secure option — if in doubt, check the “Port” option in the account/SMTP server settings in the e-mail client you use

- `hostname`: your machine name
- `from`: the e-mail address from which the mail will appear to originate
- `server`: the name of your SMTP server (check the configuration file for your regular mail client)
- `mail=FALSE` and `fake=FALSE` both say not to send real mail (probably redundant, but I'm paranoid about sending out e-mail by accident!)

Here's what the message to Karl Gauss (our first judge) looks like:

Subject: competition judging  
Reply-to: joe@joeshouse.com

Dear Karl,

Thanks for agreeing to judge a presentation ...

Here's your assignment:

Presenter: Bloggs, Joe  
Co-authors: Fred Smith  
Title: The evolution of cooperation by mutual regard  
Presentation type: Talk  
Session: New approaches to the evolution of social behavior  
Date: 2008/08/05  
Time: 10:00:00  
Location: 104B MAC

The judging form is available at <http://...> ; I will also put some copies in an envelope on the bulletin board at the meeting.

Please return completed forms to me at the meeting, or put them back in the envelope, or e-mail me electronic copies.

Please feel free to contact me with any questions -- especially, check your assignment and let me know if I made a mistake.

cheers  
Joe

If we wanted to get rid of these test files we could run

```
> unlink("mmerge.*")
+
```

Once you are happy that you're sending out the right e-mail, you can do it for real.

```

> mailmerge(data=gdat,
+           msgfile="ex1-msg.txt",
+           id="judge",
+           email="jemail",
+           auth=TRUE,
+           user="bolker",
+           from="bolker@ufl.edu",
+           host="bolker-laptop@ufl.edu",
+           server="smtp.somewhere.edu",
+           port=587,
+           trace=2)

```

A text box will pop up that asks you to enter your password.

## Dealing with attachments: using mutt and ssmtp

Alas, the Rmail package does not (currently) handle attachments.

Possible solutions:

- Write code, from scratch, to encode attachments in proper MIME format. (Advantage: maintains entirely self-contained system. Disadvantage: lots of work, reinventing the wheel.)
- Write some bits of Perl code that use the existing MIME-tools Perl module (see for example <http://search.cpan.org/~doneill/MIME-tools-5.428/lib/MIME/Entity.pm>); then use either `system()` or `RSPerl` to have R call the Perl code appropriately. (Advantage: uses existing code. Disadvantage: users need to have Perl, and the MIME-tools module, installed.)
- Call the `mutt` command-line mail program. (Advantage: easiest. Disadvantage: users need to have `mutt`, and a mail transfer agent such as `sendmail`, `postfix`, or `ssmtp`, installed.)

In this case I am using `mutt`. On my Ubuntu system, all I had to do to set this up was (1) `apt-get install mutt`; (2) `apt-get install ssmtp`; (3) configure `/etc/ssmtp/ssmtp.conf` appropriately. Here's what I used for `ssmtp.conf`:

```

# The person who gets all mail for userids < 1000
# Make this empty to disable rewriting.
root=bolker@mcmaster.ca

# The place where the mail goes. The actual machine name is required no
# MX records are consulted. Commonly mailhosts are named mail.domain.com
mailhub=smtp.googlemail.com:465

# The full hostname
hostname=collywobbles.math.mcmaster.ca

AuthUser=bbolker@gmail.com
AuthPass=XXXXX
UseTLS=YES
Define some helper functions:
> stripwhite <- function(s) {
+   gsub("^[[[:space:]]+", "",
+       gsub("[[:space:]]+$", "", s))
+ }
> capwords <- function(s, strict = TRUE) {
+   cap <- function(s) paste(toupper(substring(s,1,1)),
+                             {s <- substring(s,2); if(strict) tolower(s) else s},
+                             sep = "", collapse = " ")
+   sapply(strsplit(s, split = " "), cap, USE.NAMES = !is.null(names(s)))
+ }

```

First gather information from the class list:

```

> library(gdata) ## for read.xls()
+
> cfile <- system.file("doc", "master_classlist.xls", package="Rmail")
> classlist <- read.xls(cfile, skip=1, as.is=TRUE)
> macid <- stripwhite(classlist$MacID)
> e_mail <- paste(macid, "mcmaster.ca", sep="@")
> subject <- "'HW 1'"

```

In this case I don't have anything interesting to say to the students, but they might be alarmed to get an e-mail with no text in the body.

```

> bodyfile <- tempfile()
+
> writeLines("Please find comments on your homework attached.", con=bodyfile)

```

I have put marked-up versions of the student's homework in files called BMB\_<Surname>\_<First\_initial>.R (I use paste here to assemble the name of the attachment file; sprintf might be useful too).

```

> attachfile <- with(classlist, paste("BMB_", stripwhite(capwords(Surname)),
+                                     "_", substr(Initials, 1, 1), ".R", sep=""))

```

Check for existence of attachments:

```
> f_OK <- file.exists(attachfile)
+
```

Put together a list of information and subset to only students with attachments (*to do/exercise for the reader: for students with missing homework, skip the attachment but insert body text stating that "I haven't received your homework" ...*)

```
> mailframe <- subset(data.frame(attach=attachfile,subject=subject,
+                               e_mail=e_mail),f_OK)
```

Synthesize system appropriate system commands:

```
> mailtxt <- with(mailframe,
+                 paste("mutt", "-a", attach, "-s", subject, "--", e_mail, "<", bodyfile))
```

Execute system commands one at time:

```
> fake <- TRUE
+
+ for (i in seq_along(mailtxt)) {
+   cat(mailframe$Surname[i],mailframe$Initials[i],"\n")
+   if (fake) {
+     cat(mailtxt[i],"\n")
+   } else {
+     system(mailtxt[i])
+   }
+ }
```

Clean up:

```
> unlink(bodyfile)
+
```