As the year is closing down, why not spend some of the free time to explore your email data using R and the tidyverse? When I learned that Mac OS Mail stores its internal data in a SQLite database file I was hooked. A quick dive in your email archive might uncover some of your old acquaintances. Let’s take a peak.

Obviously, the below is only applicable when you are a regular user of the Mail app for Mac OS. As a first step, you need to locate the file Envelope Index that tends to be located in ~/Library/Mail/V6/MailData/. Copy this file somewhere and adjust the path provided in mail\_db to point to this copy. Do not work with the original file.

library(DBI)

library(tidyverse)

library(lubridate)

library(ExPanDaR)

library(ggridges)

mail\_db <- "data/EI"

con <- dbConnect(RSQLite::SQLite(), mail\_db)

Now you have established a database connection to your copy of Mac OS Mail’s internal data. I you receive an error message, check whether you have the required packages (including RSQLite) installed. With this established connection, we can now see what the database has in store for us.

kable(dbListTables(con), col.names = "List of Tables")

| **List of Tables** |
| --- |
| action\_ews\_messages |
| action\_imap\_messages |
| action\_labels |
| action\_messages |
| addresses |
| attachments |
| duplicates\_unread\_count |
| events |
| ews\_copy\_action\_messages |
| ews\_folders |
| imap\_copy\_action\_messages |
| imap\_labels |
| imap\_messages |
| labels |
| last\_spotlight\_check\_date |
| local\_message\_actions |
| mailbox\_actions |
| mailboxes |
| messages |
| properties |
| recipients |
| sqlite\_sequence |
| subjects |
| threads |

Hey, this is an impressive list of tables. For this bog post, I am mostly interested in exploring the development of my email activity in terms of senders and receivers over time. Thus, I focus on the relations messages, addresses and recipients.

messages <- dbListFields(con, "messages")

recipients <- dbListFields(con, "recipients")

addresses <- dbListFields(con, "addresses")

max\_members <- max(length(messages),

length(recipients),

length(addresses))

length(messages) <- max\_members

length(recipients) <- max\_members

length(addresses) <- max\_members

df <- data.frame(messages, recipients, addresses, stringsAsFactors = FALSE)

df[is.na(df)] <- ""

kable(df)

| **messages** | **recipients** | **addresses** |
| --- | --- | --- |
| ROWID | ROWID | ROWID |
| message\_id | message\_id | address |
| document\_id | type | comment |
| in\_reply\_to | address\_id |  |
| remote\_id | position |  |
| sender |  |  |
| subject\_prefix |  |  |
| subject |  |  |
| date\_sent |  |  |
| date\_received |  |  |
| date\_created |  |  |
| date\_last\_viewed |  |  |
| mailbox |  |  |
| remote\_mailbox |  |  |
| flags |  |  |
| read |  |  |
| flagged |  |  |
| size |  |  |
| color |  |  |
| type |  |  |
| conversation\_id |  |  |
| snippet |  |  |
| fuzzy\_ancestor |  |  |
| automated\_conversation |  |  |
| root\_status |  |  |
| conversation\_position |  |  |
| deleted |  |  |

OK. These relations provide enough data to play. Let’s create a table that is organized by message and contains sender and receiver info as well as sending/receiving time.

sql <- paste("SELECT messages.ROWID as message\_id, date\_sent,",

"date\_received, a1.address as sender\_address,",

"a1.comment as sender\_comment, a2.address as recipient\_address,",

"a2.comment as recipient\_comment, snippet",

"FROM messages left join addresses AS a1 on messages.sender = a1.ROWID",

"LEFT JOIN recipients on recipients.message\_id = messages.ROWID",

"LEFT JOIN addresses AS a2 on recipients.address\_id = a2.ROWID")

res <- dbSendQuery(con, sql)

df <- dbFetch(res)

dbClearResult(res)

dbDisconnect(con)

df[,c("date\_sent", "date\_received")] <- lapply(df[,c("date\_sent", "date\_received")],

function(x) as.POSIXct(x, origin = "1970-01-01"))

The resulting data frame contains all messages, including messages sent to multiple recipients (with me on the sending or receiving end). To limit the messages to the ones where I am involved, I match sender\_address and receiver\_address to a vector my\_adresses containing my email addresses (not disclosed here for privacy reasons).

df %>%

filter(tolower(sender\_address) %in% my\_addresses |

tolower(recipient\_address) %in% my\_addresses) %>%

distinct(.keep\_all = TRUE) %>%

arrange(date\_received) -> emails

In addition, I prepare a panel dataset that contains data at email address year level.

emails %>%

filter(!tolower(sender\_address) %in% my\_addresses) %>%

mutate(address = tolower(sender\_address),

year = year(date\_received)) %>%

group\_by(year, address) %>%

summarise(emails\_received\_from = n()) -> emails\_received

emails %>%

filter(!tolower(recipient\_address) %in% my\_addresses) %>%

mutate(address = tolower(recipient\_address),

year = year(date\_received)) %>%

group\_by(year, address) %>%

summarise(emails\_sent\_to = n()) -> emails\_sent

panel <- full\_join(emails\_received, emails\_sent) %>%

replace\_na(list(emails\_sent\_to = 0, emails\_received\_from = 0)) %>%

arrange(year, -emails\_sent\_to, -emails\_received\_from)

Time for a first analysis. How does my email in- and outflow develop over the years?

panel %>%

group\_by(year) %>%

summarise(sent\_mails = sum(emails\_sent\_to),

received\_mails = sum(emails\_received\_from)) %>%

gather(key = "var", value = "count", -year) %>%

ggplot(aes(year, count)) +

geom\_bar(aes(fill = var), position = "dodge", stat="identity") +

scale\_fill\_discrete(name="Direction",

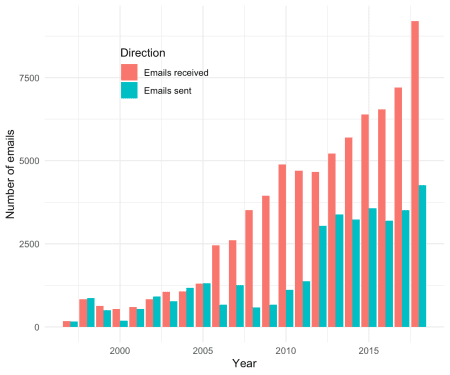
labels=c("Emails received", "Emails sent")) +

xlab("Year") +

ylab("Number of emails") +

theme\_minimal() +

theme(legend.position=c(.3, .8))



Hmm. I have no interest in extrapolating this time trend… Just for fun and giggles: When do I send emails (by time of day)?

emails %>%

filter(sender\_address %in% my\_addresses) %>%

mutate(sent\_tod = hour(date\_sent)) %>%

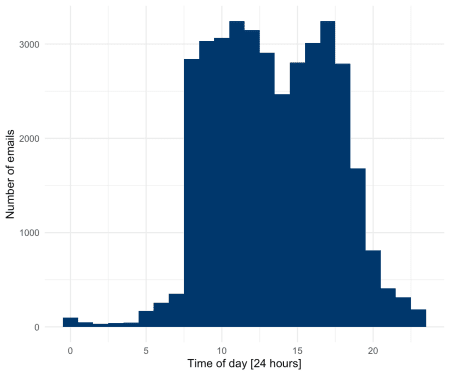
ggplot() +

geom\_histogram(aes(sent\_tod), binwidth=1, fill = "#00376C") +

xlab("Time of day [24 hours]") +

ylab("Number of emails") +

theme\_minimal()



No real surprises here (at least for me), besides that I seem to take a dip in the early afternoon. Does this sending behavior exhibit any interesting time trends?

emails %>%

filter(sender\_address %in% my\_addresses) %>%

mutate(sent\_tod = hour(date\_sent) + minute(date\_sent)/60 + second(date\_sent)/3600,

year = as.factor(year(date\_sent))) %>%

ggplot(aes(x = sent\_tod, y = year, fill = ..x..)) +

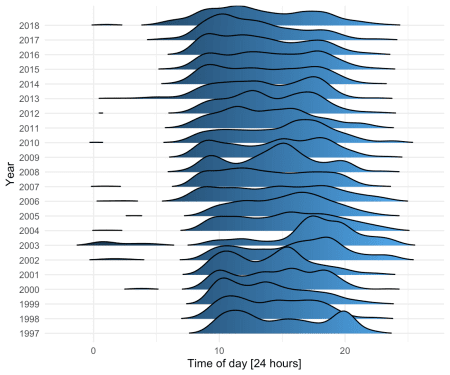
geom\_density\_ridges\_gradient(scale =2, rel\_min\_height = 0.01) +

ylab("Year") +

xlab("Time of day [24 hours]") +

theme\_minimal() +

theme(legend.position = "none")



2002 and 2003 stand out (I was based in the U.S. for most of this period) and it seems as if I am gradually becoming more of an early starter.

But enough on this. This is supposed to be about old acquaintances to fit this special end-of-the-year mood. Let’s dive more into the personal sphere. For this, I define an “email contact” as an email address with an email exchange in a given year, meaning that I was both, at the receiving and sending end, at least once.

panel %>%

group\_by(year) %>%

filter(emails\_sent\_to > 0,

emails\_received\_from > 0) %>%

summarise(nr\_email\_contacts = n()) %>%

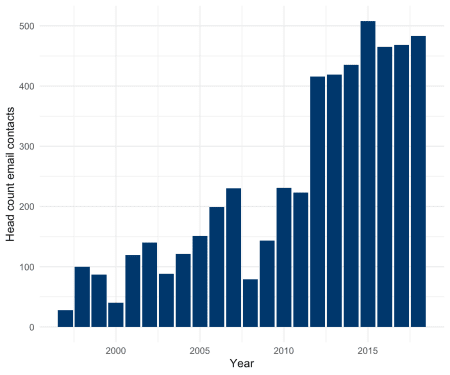
ggplot(aes(year, nr\_email\_contacts)) +

geom\_bar(stat="identity", fill = "#00376C") +

xlab("Year") +

ylab("Head count email contacts") +

theme\_minimal()



I would never have never guessed that I am exchanging emails with that many people (No, I am not a spammer …, I hope). Who are my “top contacts”?

panel %>%

filter(emails\_sent\_to > 0,

emails\_received\_from > 0) %>%

group\_by(address) %>%

summarise(email\_contacts = sum(emails\_sent\_to) + sum(emails\_received\_from),

sent\_rec\_ratio = sum(emails\_sent\_to) / sum(emails\_received\_from),

emails\_sent\_to = sum(emails\_sent\_to),

emails\_received\_from = sum(emails\_received\_from)) %>%

arrange(-email\_contacts) -> email\_contacts

# head(email\_contacts, 50)

I am not including this output here for obvious privacy reasons. If you are interested to follow your email contacts over time, you can use my ExPanD shiny app for quick exploration.

panel %>%

filter(emails\_sent\_to > 0,

emails\_received\_from > 0) %>%

ExPanD(cs\_id = "address", ts\_id = "year", components = c(missing\_values = FALSE,

by\_group\_violin\_graph = FALSE,

by\_group\_bar\_graph = FALSE))

Using ExPanD, I encountered a reasonable amount of my “old acquaintances” when focusing on earlier years of the panel. To do this a little bit more systematically, I prepare a linking table, linking the most prominent email contact addresses in the panel file to real people names in an n(email addresses) to 1(person) look-up table. This table has the name email\_contacts\_names and the fields address and name. Based on this data I produce a new panel sample containing only those email contacts linked to actual persons that I care about.

panel %>%

left\_join(email\_contacts\_names) %>%

filter(!is.na(name)) %>%

group\_by(year, name) %>%

summarise(emails\_sent\_to = sum(emails\_sent\_to),

emails\_received\_from = sum(emails\_received\_from)) %>%

filter(emails\_sent\_to > 0,

emails\_received\_from > 0) %>%

mutate(email\_contacts = emails\_sent\_to + emails\_received\_from,

sent\_rec\_ratio = emails\_sent\_to / emails\_received\_from) %>%

arrange(year, -email\_contacts) -> panel\_email\_contacts

Using this data, you can now easily explore how your contacts changed over time. As one example I prepared a graph highlighting the (relative) number of email contacts with a selected group of people over time. The vector people contains the names of these people.

single\_out\_people <- function(df, names, relative = FALSE) {

df$relative <- relative

palette <- function(n) {

hues = seq(15, 375, length = n)

c("#B0B0B0", hcl(h = hues, l = 65, c = 100)[1:n])

}

df %>%

mutate(name = ifelse(name %in% names, name, "Other")) %>%

group\_by(year) %>%

mutate(val = ifelse(relative,

email\_contacts/sum(email\_contacts),

email\_contacts)) %>%

select(year, name, val) %>%

group\_by(year, name) %>%

summarise(val = sum(val)) %>%

spread(key = name, value = val, fill = 0) %>%

gather(key = name, value = val, -year) %>%

mutate(name = factor(name, levels = c("Other", rev(names)))) %>%

arrange(year, name) %>%

ggplot(aes(x = year, y = val, fill = name)) +

geom\_area() +

scale\_fill\_manual(values = palette(length(names))) +

xlab("Year") +

ylab(ifelse(relative,

"Share of annual email contacts",

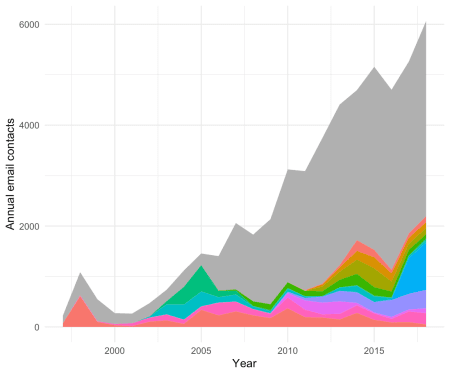
"Annual email contacts")) +

theme\_minimal() -> p

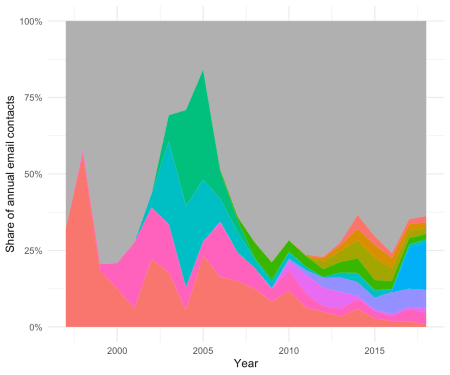
if (relative) p + scale\_y\_continuous(labels = scales::percent) else p

}

single\_out\_people(panel\_email\_contacts, people, FALSE) + theme(legend.position="none")



single\_out\_people(panel\_email\_contacts, people, TRUE) + theme(legend.position="none")



The grayish areas marks “Other”. With legends excluded for privacy reasons, you can follow old and new friends through time with this little display.

While the Envelope Index database seems to offer much more data that is worth exploring, I will it leave it here. Happy New Year everybody and if you are in the mood: [Auld Lang Syne](https://upload.wikimedia.org/wikipedia/commons/e/e3/Auld_Lang_Syne.ogg).

Enjoy!