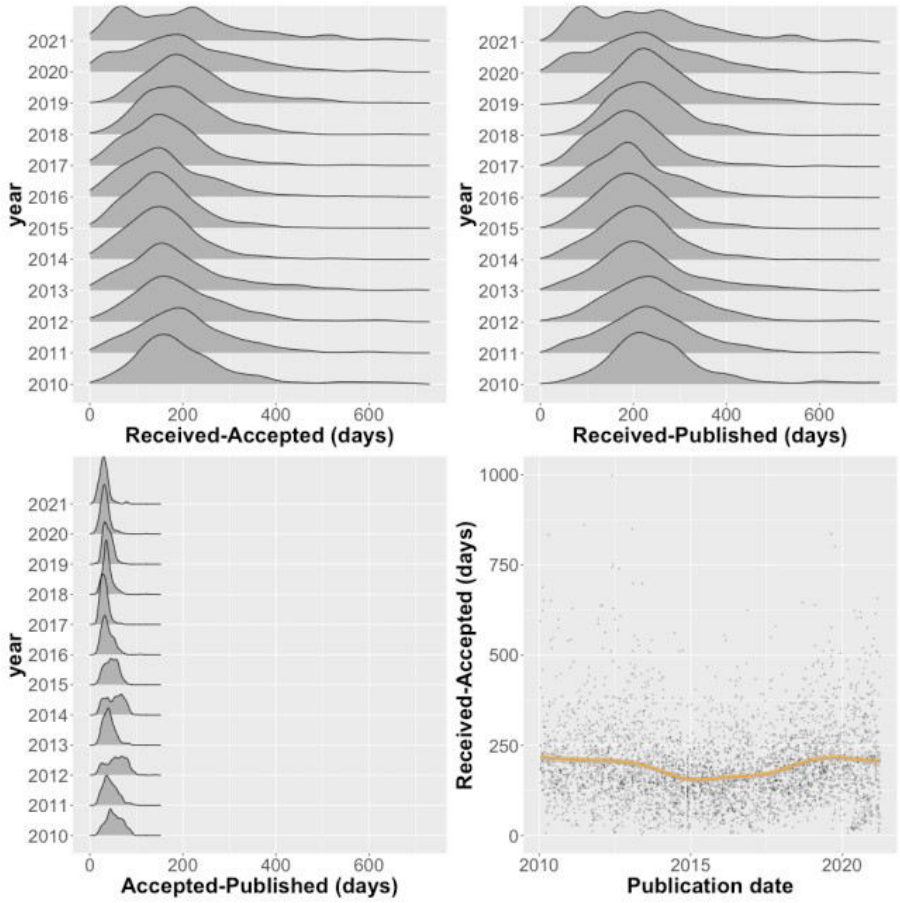
Publication lag times at Cell

There was some discussion on Twitter about whether publication lag times had increased over time and whether preprinting had affected the delays to work appearing in print. The lag times at Cell were being discussed and I thought I would run the analysis.

The median lag time for a paper to go from received to accepted at Cell is pretty stable, following a temporary dip 2015-2017. What was interesting to me is the almost bimodal distribution of lag times in 2020 and 2021. There is a population of short lag time papers appearing while the other population has got longer. I suspect this is the COVID-19 effect: Tthe quick papers are the ones related to SARS-CoV-2 and the others have taken longer because of disruption to labwork caused by the pandemic. A quick look at a few short lag time papers confirms this.



Publication lag times at Cell

The code

The key part of code is here:

```
# search pubmed using a search term (use_history allows retrieval of
all records)
pp <- entrez_search(db="pubmed", term="cell[ta] AND 2010 : 2021[pdat]
AND (journal article[pt] NOT review[pt])", use_history = TRUE)
pp_rec <- entrez_fetch(db="pubmed", web_history=pp$web_history,
rettype="xml", parsed=TRUE)
# save records as XML file
saveXML(pp_rec, file = "Data/records.xml")
```

We use two rentrez functions that identify the records we want to retrieve and then fetch them using the PubMed API e-utils. There is a default limit of 20 records, set by the retmax argument, but we need to retrieve ~6000 records. To do this we take advantage of the use_history argument. From here I could save the records in XML format. Doing so meant I could plug this file back into my previous code (it did not need any further cleaning up). Note that an API key is not required for light use.

The full code is:

```
library(devtools)
install_github("ropensci/rentrez")
library(rentrez)
require(XML)
require(ggplot2)
require(ggridges)
require(gridExtra)

## Setup preferred directory structure in wd
ifelse(!dir.exists("Data"), dir.create("Data"), "Folder exists
already")
ifelse(!dir.exists("Output"), dir.create("Output"), "Folder exists
already")
ifelse(!dir.exists("Output/Data"), dir.create("Output/Data"), "Folder
exists already")
ifelse(!dir.exists("Output/Plots"), dir.create("Output/Plots"), "Folder
exists already")
ifelse(!dir.exists("Script"), dir.create("Script"), "Folder exists
already")

# search pubmed using a search term (use_history allows retrieval of
all records)
pp <- entrez_search(db="pubmed", term="cell[ta] AND 2010 : 2021[pdat]
AND (journal article[pt] NOT review[pt])", use_history = TRUE)
pp_rec <- entrez_fetch(db="pubmed", web_history=pp$web_history,
rettype="xml", parsed=TRUE)
# save records as XML file
saveXML(pp_rec, file = "Data/records.xml")
filename <- "Data/records.xml"
# # or select XML file
# filename <- file.choose()

## extract a data frame from XML file
## This is modified from christopherBelter's pubmedXML R code
extract_xml <- function(theFile) {
  library(XML)
  newData <- xmlParse(theFile)
  records <- getNodeSet(newData, "//PubmedArticle")
  pmid <- xpathSApply(newData,"//MedlineCitation/PMID", xmlValue)
  doi <- lapply(records, xpathSApply, ".//ELocationID[@EIdType =
\"doi\"]", xmlValue)
  doi[sapply(doi, is.list)] <- NA
  doi <- unlist(doi)
```

```
  authLast <- lapply(records, xpathSApply, ".//Author/LastName",
xmlValue)
  authLast[sapply(authLast, is.list)] <- NA
  authInit <- lapply(records, xpathSApply, ".//Author/Initials",
xmlValue)
  authInit[sapply(authInit, is.list)] <- NA
  authors <- mapply(paste, authLast, authInit, collapse = "|")
  year <- lapply(records, xpathSApply, ".//PubDate/Year", xmlValue)
  year[sapply(year, is.list)] <- NA
  year <- unlist(year)
  articletitle <- lapply(records, xpathSApply, ".//ArticleTitle",
xmlValue)
  articletitle[sapply(articletitle, is.list)] <- NA
  articletitle <- unlist(articletitle)
  journal <- lapply(records, xpathSApply, ".//ISOAbbreviation",
xmlValue)
  journal[sapply(journal, is.list)] <- NA
  journal <- unlist(journal)
  volume <- lapply(records, xpathSApply, ".//JournalIssue/Volume",
xmlValue)
  volume[sapply(volume, is.list)] <- NA
  volume <- unlist(volume)
  issue <- lapply(records, xpathSApply, ".//JournalIssue/Issue",
xmlValue)
  issue[sapply(issue, is.list)] <- NA
  issue <- unlist(issue)
  pages <- lapply(records, xpathSApply, ".//MedlinePgn", xmlValue)
  pages[sapply(pages, is.list)] <- NA
  pages <- unlist(pages)
  abstract <- lapply(records, xpathSApply, ".//Abstract/AbstractText",
xmlValue)
  abstract[sapply(abstract, is.list)] <- NA
  abstract <- sapply(abstract, paste, collapse = "|")
  recdatey <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'received']/Year", xmlValue)
  recdatey[sapply(recdatey, is.list)] <- NA
  recdatem <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'received']/Month", xmlValue)
  recdatem[sapply(recdatem, is.list)] <- NA
  recdated <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'received']/Day", xmlValue)
  recdated[sapply(recdated, is.list)] <- NA
  recdate <- mapply(paste, recdatey, recdatem, recdated, collapse =
"|")
  accdatey <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'accepted']/Year", xmlValue)
  accdatey[sapply(accdatey, is.list)] <- NA
  accdatem <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'accepted']/Month", xmlValue)
  accdatem[sapply(accdatem, is.list)] <- NA
  accdated <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'accepted']/Day", xmlValue)
```

```
  accdated[sapply(accdated, is.list)] <- NA
  accdate <- mapply(paste, accdatey, accdatem, accdated, collapse =
"|")
  # use pubmed date as the published date. This seems safe for older
records.
  pubdatey <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'pubmed']/Year", xmlValue)
  pubdatey[sapply(pubdatey, is.list)] <- NA
  pubdatem <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'pubmed']/Month", xmlValue)
  pubdatem[sapply(pubdatem, is.list)] <- NA
  pubdated <- lapply(records, xpathSApply, ".//PubMedPubDate[@PubStatus
= 'pubmed']/Day", xmlValue)
  pubdated[sapply(pubdated, is.list)] <- NA
  pubdate <- mapply(paste, pubdatey, pubdatem, pubdated, collapse =
"|")
  ptype <- lapply(records, xpathSApply, ".//PublicationType", xmlValue)
  ptype[sapply(ptype, is.list)] <- NA
  ptype <- sapply(ptype, paste, collapse = "|")
  theDF <- data.frame(pmid, doi, authors, year, articletitle, journal,
volume, issue, pages, abstract, recdate, accdate, pubdate, ptype,
stringsAsFactors = FALSE)
  ## convert the dates
  theDF$recdate <- as.Date(theDF$recdate, format="%Y %m %d")
  theDF$accdate <- as.Date(theDF$accdate, format="%Y %m %d")
  theDF$pubdate <- as.Date(theDF$pubdate, format="%Y %m %d")
  return(theDF)
}

## Script to make the plots
theData <- extract_xml(filename)
write.csv(theData, file = "Output/Data/pubmed_data.csv")
theData$recacc <-as.numeric(theData$accdate - theData$recdate)
theData$recpub <-as.numeric(theData$pubdate - theData$recdate)
theData$accpub <-as.numeric(theData$pubdate - theData$accdate)
# multi-year comparison
p1 <- ggplot(theData, aes(x = recacc, y = year, group = year)) +
  geom_density_ridges() + xlim(0,730) +
  labs(x = "Received-Accepted (days)") +
  theme(axis.text=element_text(size=20), axis.title=element_text(size=
24,face="bold"))
p2 <- ggplot(theData, aes(x = recpub, y = year, group = year)) +
  geom_density_ridges() + xlim(0,730) +
  labs(x = "Received-Published (days)") +
  theme(axis.text=element_text(size=20), axis.title=element_text(size=
24,face="bold"))
p3 <- ggplot(theData, aes(x = accpub, y = year, group = year)) +
  geom_density_ridges() + xlim(0,730) +
  labs(x = "Accepted-Published (days)") +
  theme(axis.text=element_text(size=20), axis.title=element_text(size=
24,face="bold"))
p4 <- ggplot(theData, aes(x = pubdate, y = recacc)) +
```

```
  geom_point(alpha = 0.2,size = 0.3) +
  geom_smooth(color = "orange") +
  ylim(0,min(max(theData$recacc),1000, na.rm = TRUE)) +
  labs(x = "Publication date", y = "Received-Accepted (days)") +
  theme(axis.text=element_text(size=20), axis.title=element_text(size=
24,face="bold"))

png("Output/Plots/yearPlot.png", width = 1000, height = 1000)
grid.arrange(p1, p2, p3, p4, nrow = 2)
dev.off()
```

—