The code

```
## the purpose of this script is to aggregate large PubMed XML datasets
## An example is an entire year of PubMed Journal Articles in month-sized XML
files
require(XML)
require(ggplot2)

## 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")

## Function to extract a data frame from XML file
## This is modified from christopherBelter's pubmedXML R code
extract_xml <- function(theFile) {
  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)
  theDF <- data.frame(pmid, doi, year, articletitle, journal, volume, issue,
pages, 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)
}

## xml files in Data directory
theFileList <- list.files(path = "Data", full.name = TRUE, pattern = "xml$")

## loop through the files
for (i in 1:length(theFileList)){
  fileName <- theFileList[i]
  theData <- extract_xml(fileName)
  theData$recacc <-as.numeric(theData$accdate - theData$recdate)
  theData$recpub <-as.numeric(theData$pubdate - theData$recdate)
  theData$accpub <-as.numeric(theData$pubdate - theData$accdate)
  theXmlName <- basename(fileName)
  outputName <- paste0("Output/Data/",gsub(".xml",".csv",theXmlName))
  write.csv(theData, file = outputName)
```

```
  rm(theData)
}

## now we load the csvs into one large data frame (assumes all csvs in
Output/Data are for merging)
theMergedData <-
  do.call(rbind,
          lapply(list.files(path = "Output/Data/", full.name = TRUE, pattern =
"csv$"), read.csv))
## remove rows with no recpub information
theMergedData <- theMergedData[complete.cases(theMergedData[ , 15]),]
## remove rows with recpub <1
theMergedData <- theMergedData[which(theMergedData$recpub > 0),]
## summarise by journal
journalInfo <- aggregate(theMergedData[, 15], list(theMergedData$journal),
median)
journalCount <- aggregate(x = theMergedData$journal, by = list(unique.values =
theMergedData$journal), FUN = length)
colnames(journalInfo) <- c("journal","median_recpub")
colnames(journalCount) <- c("journal","articles")
## drop journals that published fewer than 10 articles
journalCount <- journalCount[which(journalCount$articles > 10),]
journalInfo <- merge(x = journalCount, y = journalInfo, by="journal")

## make plots
p1 <- ggplot(theMergedData, aes(x = recpub)) +
  geom_histogram(binwidth=7) +
  geom_vline(aes(xintercept = median(recpub)),
             color="red", linetype="dashed", size=1) +
  xlim(0,730) +
  labs(x = "Received-Published (days)", y = "Papers") +
  theme(axis.text=element_text(size=20), axis.title=element_text(size=
24,face="bold"))
p1
median(theMergedData$recpub)

p2 <- ggplot(journalInfo, aes(x = median_recpub)) +
  geom_histogram(binwidth=7) +
  geom_vline(aes(xintercept = median(median_recpub)),
             color="red", linetype="dashed", size=1) +
  xlim(0,730) +
  labs(x = "Median Received-Published (days)", y = "Journals") +
  theme(axis.text=element_text(size=20), axis.title=element_text(size=
24,face="bold"))
p2
median(journalInfo$median_recpub)

ggsave("Output/Plots/paperSummary.png", p1, dpi = 300)
ggsave("Output/Plots/journalSummary.png", p2, dpi = 300)…
```