Base R code given is below. I tried to annotate things that I think need addressing with a number in hard brackets colored red. E.g., [1].

prepareInput <- function(

data\_path, [1]

meta\_path, [1]

metaColName

){

# Open Datasets

data [2] <- data.frame(read.csv(data\_path, header=T, check.names=F [3], sep=','))

metadata <- data.frame(read.csv(meta\_path, header=T, check.names=F, sep=','))

print(data) [4]

# Remove rows if the metadata id is NA

metadata <- data.frame(metadata[!is.na[5](metadata[, metaColName]), ])

# Standardize NA representation

data[data== "NaN"] <- NA [6]

data[data== "Filterered"] <- NA [6]

col2select <- intersect(meta\_data[7][, metaColName], colnames(data))

rownames(data) <- 1:nrow(data)

# Convert elements to numeric in the quantitative part of the data

quant.data [8] <- sapply(data[, col2select], as.numeric)

# Add rownames to quant.data

rownames(quant.data) <- rownames(data)

# Remove columns where quantitative data is completely missing

quant.data <- data.frame(quant.data[rowSums(is.na(quant.data)) != ncol(quant.data), ]) [9]

[10]

# Make sure the rownames are consistent with annotation and quantitative data

data <- data[rownames(quant.data), ] [11]

return(data)

}

prepareInput("./Data/Placenta-Example.csv",[12]

"./Data/Metadata-Example.csv",

"Columns")

Above, I have highlighted sections in the base code that are either bugs, or where I think there is room for improvement. I’ll address these first, and then I will explain what I did to further improve. I’ll end it with some other things that could be added to further improve it.

**[1]** data\_path and meta\_path: This isn’t really a bug, but personally, I interpret 'path' as the path leading up to a file, but explicitly not including the file. Because of this, I find data\_path and meta\_path a bit misleading. I thought about adding separate variables for the path, the data file, and the meta file. However, then there should really be two paths (one for each file separately since the meta file might be stored elsewhere from the main data), but that's getting unwieldly. Instead, I decided to just rename the variable to something more descriptive: data\_file\_w\_path. I realize it might seem a bit unnecessary, but the first thing I thought of when I saw this function was “this just sets paths, where are the actual files?”. I think it also improves the readability for others using the script.

**[2]**: I don’t like having a variable named data, and then having a function called data.frame. This could be misinterpreted by someone unfamiliar with the code. I will rename the variables for data and metadata to data\_df and metadata\_df to indicate that they have been made into dataframes.

**[3]** check.names=F: I’m 50/50 about this. For the data file, I think this should be T instead. By setting it to TRUE, R can correct for column names starting with numbers, or containing ‘-’, spaces, or other disallowed characters. However, if the columns in the data file are made syntactically correct, the column names provided in the meta file won’t match. So either both need to be ‘fixed’ in the same way (with a secondary step with make.names() on just the contents of the ‘Columns’ column in the meta file, or they both just need to be left as-is. I will leave them both as check.names=F for simplicity.

**[4]** print(data): I don’t think there is a lot of utility in printing the entire data set (or as much of it as R will allow. Removing that and adding in print(head(data)) and print(dim(data)) may be more informative at a glance.

**[5]** metadata <- data.frame(metadata[!is.na(metadata[, metaColName]), ])

This is.na only catches an ‘NA’ if it is actually written as ‘NA’ in the cell. If it is just blank, it leaves the line in. Looking at this closer, this is actually probably more of an issue with read.csv rather than this line itself. To rectify this, I added an extra command to the read.csv line (na.strings=c('',' ','NA','NaN','na','nan','NAN','Nan')).

**[6]** data[data== "NaN"] <- NA; data[data== "Filterered"] <- NA

There isn’t technically anything wrong with these lines. However, in light of the above addition to read.csv, a similar expanded line can be added to the read.csv for the data file (na.strings=c('',' ','NA','NaN','na','nan','NAN','Nan', 'Filtered')).

**[7]** meta\_data

Spelling mistake in original code

**[8]** quant.data

Personally I don’t like having periods in variable names. I don’t know if it is a common practice in R, but it makes it seem like it is a call to a function or something. I renamed these to quant\_data

**[9]** The command as a whole does not appear to be correct

quant.data <- data.frame(quant.data[rowSums(is.na(quant.data)) != ncol(quant.data), ])

I’m interpreting this as saying that if the sum of a row does not equal the number of columns, then the row should be kept (but the description says columns). But really, if empty columns are the concern, then it should sum the number of NA per column, and it should not be equal to the number of rows.

**[10]** Correct for empty rows as well?

I’m 50/50 on this. It seems that imputation is quite common for proteins that are missing, but I don’t think you can impute if there are no data points at all(?). So there should be some code to filter out rows with 0 entries across columns as well.

[11] Corrected to account for the fact that rows AND columns were removed

[12] Incorrect file names

**Improvements on my end**

I tried to imagine what could go wrong with processing these files. I added a bunch of checks to the script to verify that the files were good to go, and also added more print statements so that the user knows what’s going on.

Checks added:

* Missing files (separate for each file)
* More/better handling of NA-type fillers in the files
* Files of 0 size
* Files containing only headers
* Check to see if the column name passed in (‘Columns’) is actually in the metadata file
* Check to see if there are NAs in the ‘Columns’ column
  + Remove them, inform the user, and proceed
* Checked to see if the columns in ‘Columns’ are actually in the data file
  + If one is missing, inform the user and continue
  + If none match, inform the user and halt import
* Check for duplicate row entries in the protein names column
  + Remove duplicates, but inform the user which proteins were duplicated
* Added in simultaneous removal of rows and columns containing only NA in the data file
  + Inform the user that empty rows and/or columns have been found (and specify which ones)
  + Print out before and after data frame sizes
* Added print statement that import had finished

Once I finished, there were a couple of other things that could probably have been added as well, but I think there were enough changes made already.

**Other potential changes:**

* Check for duplicate entries in the ‘Columns’ column (metadata)
* Check for duplicate column names in the data file
* Check for a missing header (data file)
  + I suppose this could be checked by seeing if the first row had non-numeric/NA data in it
* Check for a missing header (metadata file)
  + This is already sort of checked in my redone code with metaColName %in% colnames(metadata\_df) == FALSE
  + The lack of presence of this item could also be taken to mean the header is missing
* Missing protein/gene names
  + Also could be check by looking for non-numeric data in the first 1-2 columns
* Empty gene/protein names
  + Should be removed because the intensities can’t correspond to anything?
* Warning when sending in .txt files instead of .csv (which may already be in csv format
  + Could check if it actually is a csv and proceed, but print a warning that a txt file was input so the user should double check
* Files with comment rows (# or ;) at the top or bottom
  + Parse file first for rows starting with those characters and remove