```r
setwd("~/04_Champion_Lab/02_N-terminal_Acetylation/OnePot")
library(readxl)
library(stringr)
library(tidyr)
library(dplyr)
library(seqinr)
library(Peptides)
library(cleaver)

#One pot import and cleaning
opDF <- read.csv("Data/db.protein-peptides.csv")

opDF <- opDF[!grepl("CONTAM", opDF$Accession),]
opDF$Peptide <- str_remove(opDF$Peptide, "^.\\.")
opDF$Peptide <- str_remove(opDF$Peptide, "\\..$")


opDF$NtermAcet_L <- grepl("^[A-Z]\\(\\+42\\.01\\)", opDF$Peptide)
opDF$NtermAcet_H <- grepl("^[A-Z]\\(\\+45\\.03\\)", opDF$Peptide)
opDF$UnLabelled <- !opDF$NtermAcet_H & !opDF$NtermAcet_L

names(opDF) <- str_replace_all(names(opDF), "Intensity.New", "Intensity_Sample")
names(opDF) <- str_replace_all(names(opDF), "Area.New", "Area_Sample")
names(opDF) <- str_replace_all(names(opDF), "X.Spec.New", "nSpec_Sample")


opDF <- opDF  %>% pivot_longer(which((grepl("^Area", names(opDF)) |
                                        grepl("^Intensity", names(opDF)) |
                                        grepl("^nSpec", names(opDF)))),
                                           names_to = c(".value", "measure"),
                                           names_sep = "_")
drops <- c("Unique","Top","scan", "Source File", "AScore")
opDF <- opDF[,!names(opDF) %in% drops]


opDF$Accession2 <- str_remove(str_extract(opDF$Accession, "^[^\\|]*\\|"), "\\|")


opDFsummary <- opDF %>% group_by(Peptide, Mass, m.z, z,
                                 RT, Start, End, Length) %>%
  summarise(Accessions = paste(unique(Accession2), collapse = ";"),
            rowsN = n(),
            valuesArea = sum(!is.na(Area)),
            meanArea = mean(Area, na.rm = T),
            sdArea = sd(Area, na.rm = T),
            maxArea = max(Area, na.rm = T),
            valuesIntensity = sum(!is.na(Intensity)),
            meanIntensity = mean(Intensity, na.rm = T),
            sdIntensity = sd(Intensity, na.rm = T),
            maxIntensity = max(Intensity, na.rm = T),
            valuesSpec = sum(nSpec > 0))


opDFsummary$str_seq <- str_remove_all(opDFsummary$Peptide,
"\\([\\+\\-][0-9]+\\.[0-9]+\\)")

opDFsummary$IEP <- NA
for (i in 1:nrow(opDFsummary)) {
  opDFsummary$IEP[i] <- computePI(s2c(opDFsummary$str_seq[i]))
  if (i %% 500 == 0) {
    print(i/ nrow(opDFsummary) *100)
  }
}
#save as dataframe
write.csv(opDFsummary, "Data/OnePotSummarisedData.csv",
          row.names = F, quote = F)


#enriched data (Thompson et al.) import and cleaning (trypsin only)
```

```r
enrDF <- read_excel("Data/Cristal Marinum.xlsx", sheet = "Raw NTA Peptides")

enrDF <- enrDF[!grepl("GluC", enrDF$File),]

enrDF$Peptide <- paste0(enrDF$Sequence, "|", enrDF$Modifications)

enrDFsummary <- enrDF %>% group_by(Peptide, Sequence, `Theor m/z`, `Theor z`,
                            `Theor MW`, Length) %>%
  summarise(Genes = paste(unique(Gene), collapse = ";"),
            rowsN = n(),
            meanIntensity = mean(`Intensity (Peptide)`, na.rm = T),
            maxIntensity = max(`Intensity (Peptide)`, na.rm = T),
            valuesIntensity = sum(!is.na(`Intensity (Peptide)`)))

#function to return start position of peptide from proteome db
return_start_position <- function(peptide, accession, database){
  protein_names <- names(database)
  protein_index <- which(grepl(accession, protein_names))
  if(length(protein_index) > 1) {
    return(NA)
  } else if (length(protein_index) < 1) {
    return(NA)
  } else {
    protein <- toupper(c2s(database[[protein_index[1]]]))
  }
  matches_df <- str_locate(protein, peptide)
  if (length(matches_df) < 1) {
    return(NA)
  } else {
    start_position <- matches_df[[1,1]]
  }
  return(start_position)
}

#read in database
db <- read.fasta("Data/SDW_codon_MARINUM_MetStart.fasta", seqtype = "AA")


enrDFsummary$IEP <- NA
enrDFsummary$Start <- NA
for (i in 1:nrow(enrDFsummary)) {
  enrDFsummary$IEP[i] <- computePI(s2c(enrDFsummary$Sequence[i]))
  enrDFsummary$Start[i] <- return_start_position(enrDFsummary$Sequence[i],
                                                 enrDFsummary$Genes[i],
                                                 database = db)
  if (i %% 500 == 0) {
    print(i/ nrow(enrDFsummary) *100)
  }
}

#saving
write.csv(enrDFsummary, "Data/EnrichedSummarisedData.csv",
          row.names = F, quote = F)


#same but with GluC
enrDF <- read_excel("Data/Cristal Marinum.xlsx", sheet = "Raw NTA Peptides")

enrDF <- enrDF[grepl("GluC", enrDF$File),]

enrDF$Peptide <- paste0(enrDF$Sequence, "|", enrDF$Modifications)

enrDFsummary <- enrDF %>% group_by(Peptide, Sequence, `Theor m/z`, `Theor z`,
                                   `Theor MW`, Length) %>%
  summarise(Genes = paste(unique(Gene), collapse = ";"),
            rowsN = n(),
            meanIntensity = mean(`Intensity (Peptide)`, na.rm = T),
            maxIntensity = max(`Intensity (Peptide)`, na.rm = T),
            valuesIntensity = sum(!is.na(`Intensity (Peptide)`)))
```

```r


return_start_position <- function(peptide, accession, database){
  protein_names <- names(database)
  protein_index <- which(grepl(accession, protein_names))
  if(length(protein_index) > 1) {
    return(NA)
  } else if (length(protein_index) < 1) {
    return(NA)
  } else {
    protein <- toupper(c2s(database[[protein_index[1]]]))
  }
  matches_df <- str_locate(protein, peptide)
  if (length(matches_df) < 1) {
    return(NA)
  } else {
    start_position <- matches_df[[1,1]]
  }
  return(start_position)
}
db <- read.fasta("Data/SDW_codon_MARINUM_MetStart.fasta", seqtype = "AA")


enrDFsummary$IEP <- NA
enrDFsummary$Start <- NA
for (i in 1:nrow(enrDFsummary)) {
  enrDFsummary$IEP[i] <- computePI(s2c(enrDFsummary$Sequence[i]))
  enrDFsummary$Start[i] <- return_start_position(enrDFsummary$Sequence[i],
                                                 enrDFsummary$Genes[i],
                                                 database = db)
  if (i %% 500 == 0) {
    print(i/ nrow(enrDFsummary) *100)
  }
}


write.csv(enrDFsummary, "Data/EnrichedSummarisedDataGLUC.csv",
          row.names = F, quote = F)





#list of Nterminal peptides from genome
NtermPeptides <- list()

for (i in 1:length(db)) {
  protein <- db[[i]]
  seq <- toupper(c2s(protein))
  MMAR <- str_remove(str_extract(attr(protein, 'name'), "^[^\\|]+|"), "\\|")
  peptide <- cleave(seq, enzym = "trypsin")[[1]][1]
  NtermPeptides[[MMAR]] <- peptide
}

NtermGenome <- data.frame(Accession = names(NtermPeptides),
                          Peptide = unlist(NtermPeptides))

NtermGenome$Length <- nchar(NtermGenome$Peptide)
NtermGenome$secondAA <- substr(NtermGenome$Peptide, 2,2)
NtermGenome$metCleavedPeptide <- str_remove(NtermGenome$Peptide, "^M")


NtermGenome$IEP <- NA
for (i in 1:nrow(NtermGenome)) {
  if (NtermGenome$secondAA[i] %in% c("G", "A", "S", "T", "V", "C", "P")) {
    NtermGenome$IEP[i] <- computePI(s2c(NtermGenome$metCleavedPeptide[i]))
  } else {
    NtermGenome$IEP[i] <- computePI(s2c(NtermGenome$Peptide[i]))
  }
  if (i %% 500 == 0) {
    print(i/ nrow(NtermGenome) *100)
  }
}


write.csv(NtermGenome, "Data/GenomeSummarisedData_GASTVCP.csv", row.names = F, quote = F)







#all genome peptides (not just n terminal)


proteinDFList <- list()

#filters
minLength <- 5
maxLength <- 50



for (i in 1:length(db)) {
  if (i %% 100 == 0) {
    print(i / length(db))
  }
  protein <- db[[i]]
  seq <- toupper(c2s(protein))
  seq <- str_remove(seq, "\\*")
  MMAR <- str_remove(str_extract(attr(protein, 'name'), "^[^\\|]+|"), "\\|")
  peptides <- cleave(seq, enzym = "trypsin")[[1]]
  peptideDF <- data.frame(Accession = MMAR,
                          Peptide = peptides)
  peptideDF$NtermPeptide <- c(T, rep(F, nrow(peptideDF) - 1))
  NtermPeptide <- peptideDF$Peptide[1]
  secondAA <- substr(NtermPeptide, 2,2)
  if (secondAA %in% c("G", "A", "S", "T", "V", "C", "P")) {
    peptideDF$Peptide[1] <- str_remove(peptideDF$Peptide[1], "^M")
  }
  peptideDF$Length <- nchar(peptideDF$Peptide)

  peptideDF <- peptideDF[peptideDF$Length >= minLength &
                           peptideDF$Length <= maxLength,]


  if (nrow(peptideDF) > 0) {
    peptideDF$IEP <- NA
    peptideDF$mass <- NA
    peptideDF$IEP <- sapply(peptideDF$Peptide, function(peptide) computePI(s2c(peptide)))
    peptideDF$mass <- sapply(peptideDF$Peptide, mw)

    proteinDFList[[MMAR]] <- peptideDF
  } else {
    print(i)
    print("noPeps")
  }

}

genomePeps <- bind_rows(proteinDFList)

write.csv(genomePeps, "Data/GenomeSummarisedData_allPeptides_GASTVCP.csv", row.names = F, quote = F)

```