Assessment 4 Part 2

Dulki Nihinsa Danthanarayana

2025-09-28

1. Downloading Coding DNA Sequences for E. coli and Barnesiella intestinihominis

```
library("R.utils")
```

```
## Loading required package: R.oo
## Loading required package: R.methodsS3
## R.methodsS3 v1.8.2 (2022-06-13 22:00:14 UTC) successfully loaded. See ?R.methodsS3 for help.
## R.oo v1.27.1 (2025-05-02 21:00:05 UTC) successfully loaded. See ?R.oo for help.
##
## Attaching package: 'R.oo'
## The following object is masked from 'package:R.methodsS3':
##
##     throw
## The following objects are masked from 'package:methods':
##
##     getClasses, getMethods
## The following objects are masked from 'package:base':
##
##     attach, detach, load, save
## R.utils v2.13.0 (2025-02-24 21:20:02 UTC) successfully loaded. See ?R.utils for help.
##
## Attaching package: 'R.utils'
## The following object is masked from 'package:utils':
##
##     timestamp
## The following objects are masked from 'package:base':
##
##     cat, commandArgs, getOption, isOpen, nullfile, parse, warnings
```

```
URL="https://ftp.ensemblgenomes.ebi.ac.uk/pub/bacteria/release-62/fasta/bacteria_0_collection/escherichi
download.file(URL,destfile="ecoli_cds.fa.gz")
gunzip("ecoli_cds.fa.gz")
```

The script begins by loading the R.utils library to enable file decompression. It then specifies the URL for a compressed FASTA file containing Escherichia coli coding sequences, downloads it as ecoli_cds.fa.gz, uncompresses the file using gunzip.

```
library("R.utils")

URL="https://ftp.ensemblgenomes.ebi.ac.uk/pub/bacteria/release-62/fasta/bacteria_114_collection/barnesi
download.file(URL,destfile="barnesiella intestinihominis_cds.fa.gz")
gunzip("barnesiella intestinihominis_cds.fa.gz")
```

The script begins by loading the R.utils library to enable file decompression. It then specifies the URL for a compressed FASTA file containing Barnesiella intestinihominis coding sequences, downloads it as barnesiella intestinihominis_cds.fa.gz, uncompresses the file using gunzip.

Number of coding sequences in E.coli

```
library("seqinr")
```

```
##
## Attaching package: 'seqinr'
```

```
## The following object is masked from 'package:R.oo':
##
##     getName
```

Load the seqinr library: This library provides tools for biological sequence retrieval and analysis.

```
cdse <- seqinr::read.fasta("ecoli_cds.fa")
```

The code reads the uncompressed FASTA file ecoli_cds.fa containing Escherichia coli coding sequences into R using the read.fasta function from the seqinr package.

```
length(cdse)
```

```
## [1] 4239
```

The above code figure out the number of coding sequences in Escherichia coli.

Number of coding sequences in Barnesiella intestinihominis

```
cdsb <- seqinr::read.fasta("barnesiella intestinihominis_cds.fa")
```

The code reads the uncompressed FASTA file barnesiella intestinihominis_cds.fa containing Barnesiella intestinihominis coding sequences into R using the read.fasta function from the seqinr package.

```
length(cdsb)
```

```
## [1] 2803
```

The above code figure out the number of coding sequences in Barnesiella intestinihominis.

3. Calculating the length of all coding sequences in E. coli and Barnesiella intestinihominis

E. coli

```
lene <- as.numeric(summary(cdse)[,1])

sum(lene)
```

```
## [1] 3978528
```

The code above extracts the lengths of genes from the cdse object by converting the first column of the summary to numeric values. It then calculates the total length of all genes by summing these values.

Barnesiella intestinihominis

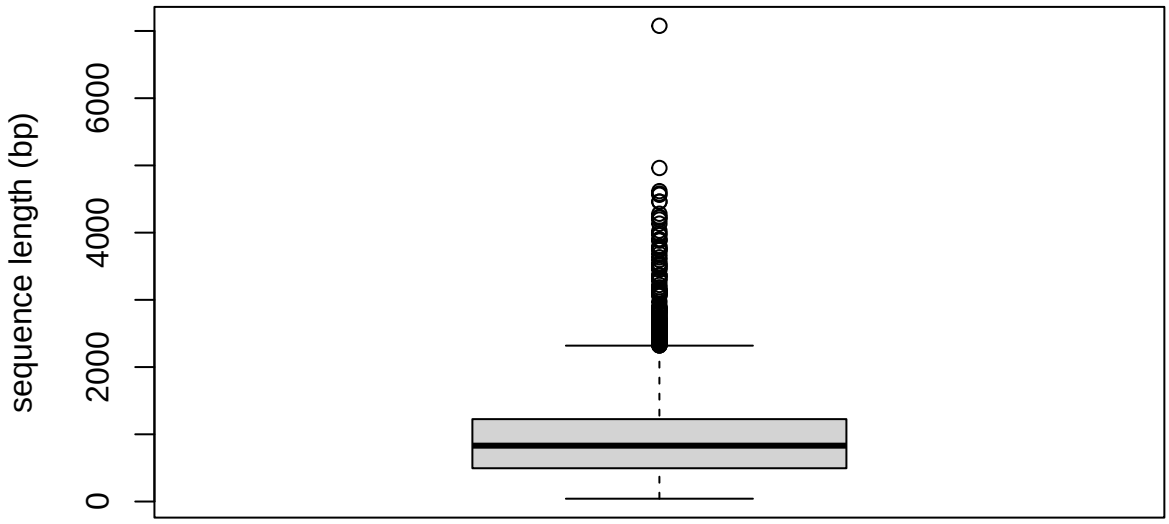
```
lenb <- as.numeric(summary(cdsb)[,1])

sum(lenb)
```

```
## [1] 3074151
```

The code above extracts the lengths of genes from the cdsb object by converting the first column of the summary to numeric values. It then calculates the total length of all genes by summing these value

Boxplot of coding sequence length in E. coli

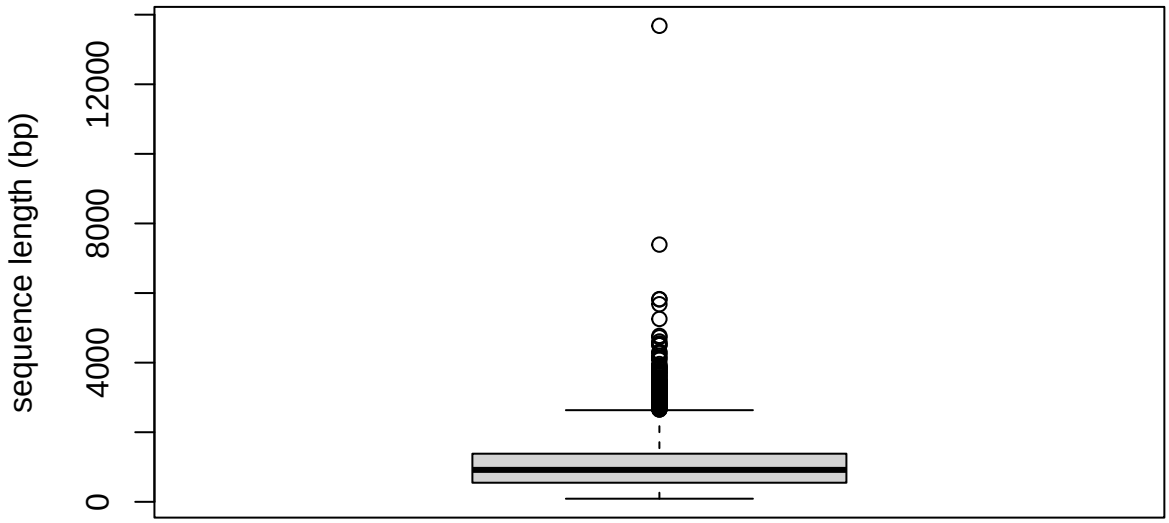
```
boxplot(lene,ylab="sequence length (bp)")
```



This code creates a boxplot of gene lengths, labeling the y-axis as “sequence length (bp).”

Boxplot of coding sequence length in Barnesiella intestinihominis

```
boxplot(lenb,ylab="sequence length (bp)")
```



This code creates a boxplot of gene lengths, labeling the y-axis as "sequence length (bp)."

Mean and median coding sequence length of E. coli

```
mean(lene)
```

```
## [1] 938.5534
```

```
median(lene)
```

```
## [1] 831
```

Mean and median coding sequence length of Barnesiella intestinihominis

```
mean(lenb)
```

```
## [1] 1096.736
```

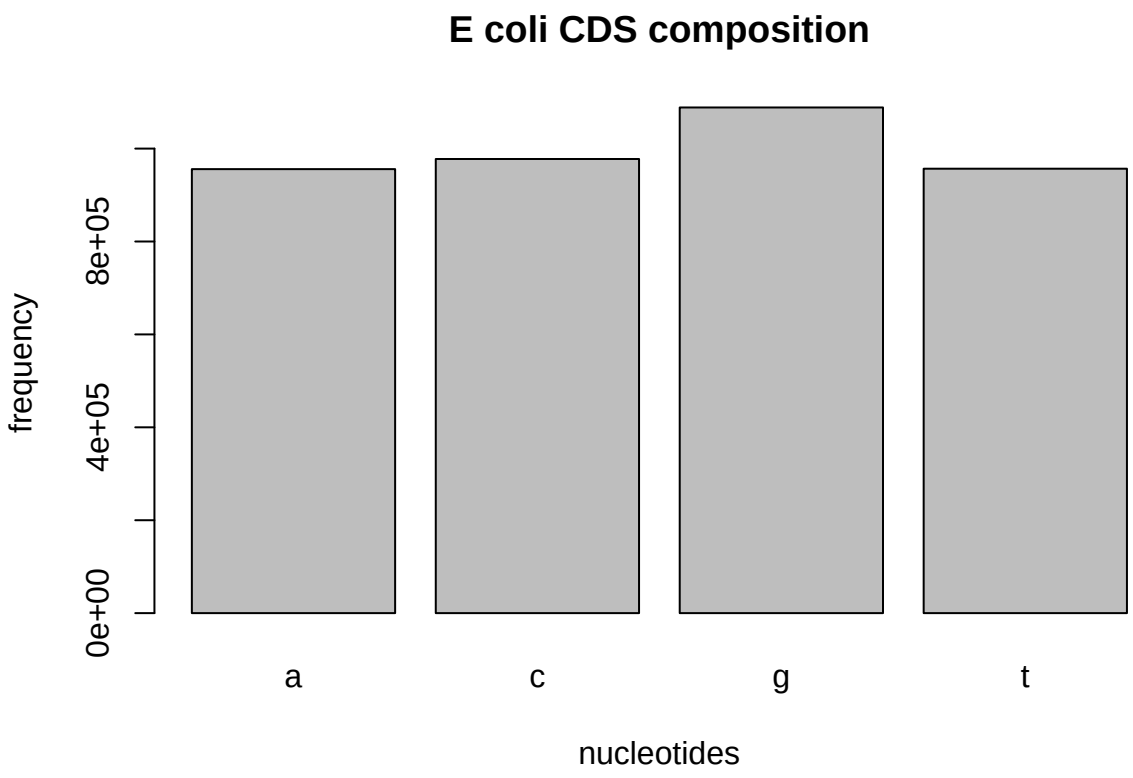
```
median(lenb)
```

```
## [1] 918
```

4. Calculating the frequency of DNA bases in the total coding sequences for E. coli and Barnesiella intestinihominis

E. coli

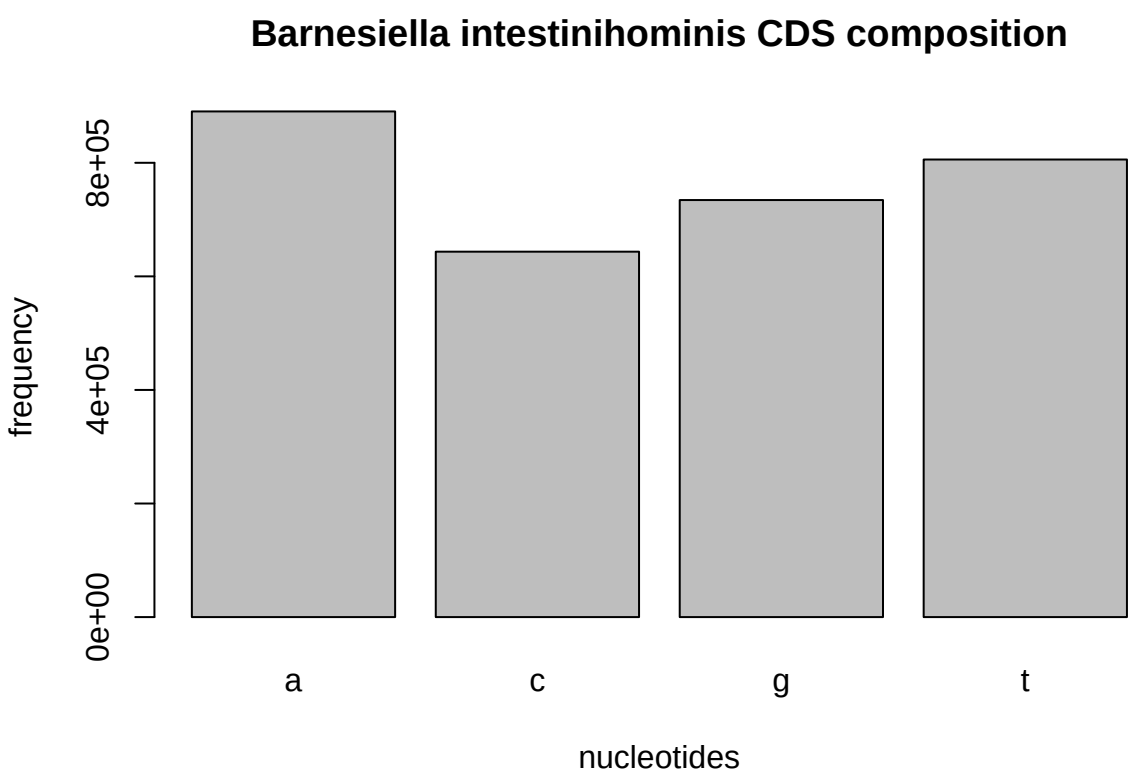
```r
dna_E <- unlist(cdse)

dna_composition_E.coli <- count(dna_E,1)

barplot(dna_composition_E.coli,xlab="nucleotides",ylab="frequency", main="E coli CDS composition")
```



The code dna_E <- unlist(cdse) unlist the cdse and stores it in the dna variable and then calculates the frequency of each nucleotide using dna_composition_E.coli <- count(dna_E,1), which stores these frequencies in dna_composition_E.coli. Then this data is visualized with a barplot(dna_composition_E.coli, xlab="nucleotides", ylab="frequency", main="E coli CDS composition"). This creates a bar plot where the x-axis represents the nucleotides (A, C, G, T), the y-axis shows their frequencies, and the plot title indicates that it displays the nucleotide composition of E. coli coding sequences.

Barnesiella intestinihominis

```r
dna_B <- unlist(cdsb)

dna_composition_B.intestinihominis <- count(dna_B,1)

barplot(dna_composition_B.intestinihominis,xlab="nucleotides",ylab="frequency", main="Barnesiella intes
```



The code dna_B <- unlist(cdsb) unlist the cdsb and stores it in the dna variable and then calculates the frequency of each nucleotide using dna_composition_B.intestinihominis <- count(dna_B,1), which stores these frequencies in dna_composition_B.intestinihominis. Then this data is visualized with a barplot(dna_composition_B.intestinihominis, xlab="nucleotides", ylab="frequency", main="Barnesiella intestinihominis CDS composition"). This creates a bar plot where the x-axis represents the nucleotides (A, C, G, T), the y-axis shows their frequencies, and the plot title indicates that it displays the nucleotide composition of Barnesiella intestinihominis coding sequences.

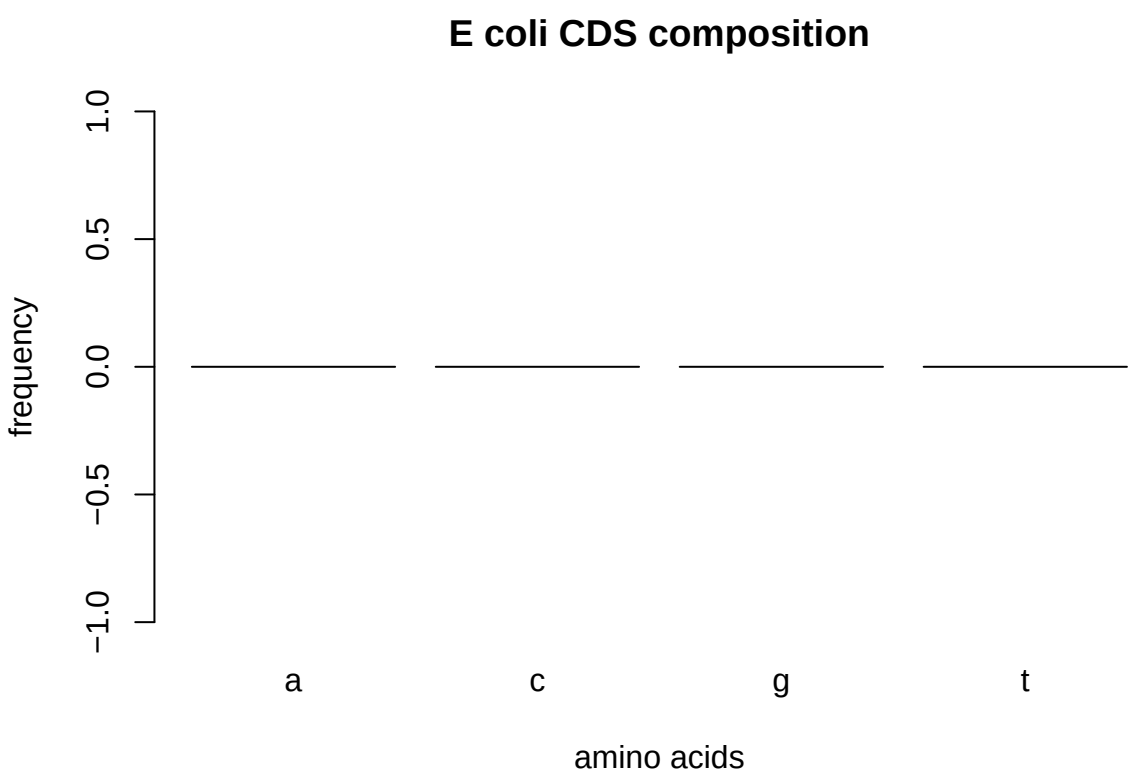
The frequency of amino acid in the total protein sequence for E. coli and Barnesiella intestinihominis

E. coli

```
prot_E.coli <- lapply(cdse, translate)
```

In the above code, lapply applies the translate function to each element of the cdse list, resulting in prot_E.coli, a list of translated protein sequences.

```
amino_E <- unlist(prot_E.coli)

aminoacid_composition_E.coli <- count(amino_E,wordsize=1)

barplot(aminoacid_composition_E.coli,xlab="amino acids",ylab="frequency", main="E coli CDS composition")
```

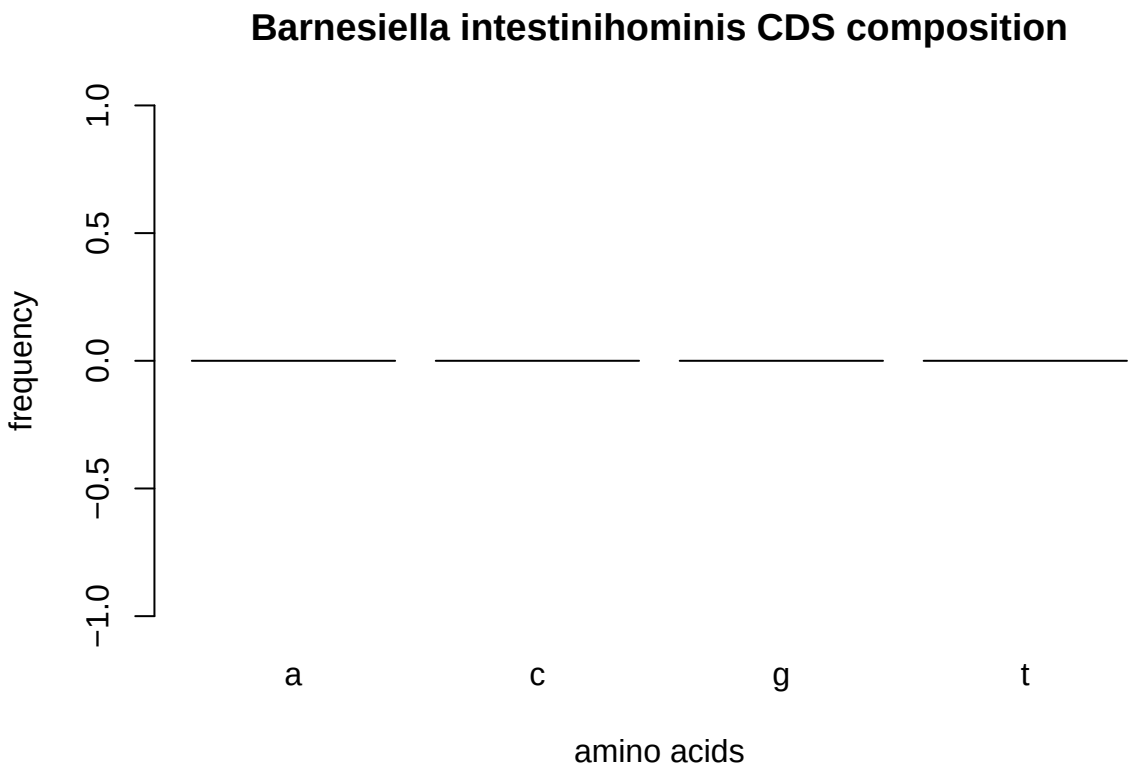


Barnesiella intestinihominis

```
prot_B.intestinihominis <- lapply(cdsb, translate)
```

In the above code, lapply applies the translate function to each element of the cdsb list, resulting in prot_B.intestinihominis, a list of translated protein sequences.

```
amino_B <- unlist(prot_B.intestinihominis)

aminoacid_composition_B.intestinihominis <- count(amino_B,wordsize=1)

barplot(aminoacid_composition_B.intestinihominis,xlab="amino acids",ylab="frequency", main="Barnesiella
```



5. Creating a codon usage table and quantifying the codon usage bias among all coding sequences

E.coli

```
codon_E <- unlist(cdse)

uco(codon_E,as.data.frame=TRUE)
```

```
##      AA codon   eff         freq      RSCU
## aaa Lys   aaa 44592 0.0336244963 1.5346652
## aac Asn   aac 28454 0.0214556741 1.1049453
## aag Lys   aag 13521 0.0101954793 0.4653348
## aat Asn   aat 23049 0.0173800461 0.8950547
## aca Thr   aca  9116 0.0068738991 0.5133967
## acc Thr   acc 31139 0.0234802922 1.7536924
## acg Thr   acg 19081 0.0143879847 1.0746075
## act Thr   act 11689 0.0088140639 0.6583034
## aga Arg   aga  2573 0.0019401648 0.2111584
## agc Ser   agc 21291 0.0160544302 1.6718055
## agg Arg   agg  1420 0.0010707478 0.1165351
## agt Ser   agt 11487 0.0086617463 0.9019787
## ata Ile   ata  5486 0.0041367058 0.2069902
## atc Ile   atc 33524 0.0252786960 1.2648816
## atg Met   atg 37007 0.0279050443 1.0000000
## att Ile   att 40501 0.0305396870 1.5281282
```

```
## caa Gln   caa 20402 0.0153840818 0.6939574
## cac His   cac 12890 0.0097196752 0.8594766
## cag Gln   cag 38397 0.0289531706 1.3060426
## cat His   cat 17105 0.0128979864 1.1405234
## cca Pro   cca 11163 0.0084174348 0.7606814
## ccc Pro   ccc  7238 0.0054577975 0.4932198
## ccg Pro   ccg 31074 0.0234312791 2.1174787
## cct Pro   cct  9225 0.0069560903 0.6286201
## cga Arg   cga  4619 0.0034829465 0.3790674
## cgc Arg   cgc 29441 0.0221999192 2.4161344
## cgg Arg   cgg  7079 0.0053379039 0.5809523
## cgt Arg   cgt 27979 0.0210975014 2.2961524
## cta Leu   cta  5149 0.0038825918 0.2179763
## ctc Leu   ctc 14811 0.0111682009 0.6270047
## ctg Leu   ctg 70714 0.0533217311 2.9935864
## ctt Leu   ctt 14586 0.0109985402 0.6174796
## gaa Glu   gaa 52679 0.0397224803 1.3801514
## gac Asp   gac 25347 0.0191128478 0.7477432
## gag Glu   gag 23659 0.0178400152 0.6198486
## gat Asp   gat 42449 0.0320085720 1.2522568
## gca Ala   gca 26743 0.0201654984 0.8481293
## gcc Ala   gcc 34117 0.0257258463 1.0819888
## gcg Ala   gcg 45082 0.0339939797 1.4297335
## gct Ala   gct 20185 0.0152204534 0.6401484
## gga Gly   gga 10350 0.0078043940 0.4257245
## ggc Gly   ggc 39536 0.0298120310 1.6262263
## ggg Gly   ggg 14581 0.0109947699 0.5997573
## ggt Gly   ggt 32779 0.0247169305 1.3482920
## gta Val   gta 14430 0.0108809087 0.6141144
## gtc Val   gtc 20350 0.0153448713 0.8660588
## gtg Val   gtg 34996 0.0263886543 1.4893658
## gtt Val   gtt 24213 0.0182577576 1.0304610
## taa Stp   taa  2726 0.0020555341 1.9292286
## tac Tyr   tac 16160 0.0121854113 0.8641480
## tag Stp   tag   294 0.0002216900 0.2080679
## tat Tyr   tat 21241 0.0160167278 1.1358520
## tca Ser   tca  9303 0.0070149060 0.7304874
## tcc Ser   tcc 11390 0.0085886036 0.8943621
## tcg Ser   tcg 11830 0.0089203846 0.9289117
## tct Ser   tct 11111 0.0083782243 0.8724546
## tga Stp   tga  1219 0.0009191842 0.8627035
## tgc Cys   tgc  8574 0.0064652052 1.1152445
## tgg Trp   tgg 20196 0.0152287479 1.0000000
## tgt Cys   tgt  6802 0.0051290326 0.8847555
## tta Leu   tta 18323 0.0138164165 0.7756807
## ttc Phe   ttc 21974 0.0165694448 0.8523496
## ttg Leu   ttg 18148 0.0136844582 0.7682723
## ttt Phe   ttt 29587 0.0223100101 1.1476504
```

The code codon_E <- unlist(cdse) unlist the cdse and stores it in the codon variable and then calculates the codon usage using uco(codon_E,as.data.frame=TRUE) giving the values in a data frame.

```
uco(codon_E,index="rscu",as.data.frame=TRUE)
```

```
##      AA codon   eff         freq      RSCU
```

```
## aaa Lys   aaa 44592 0.0336244963 1.5346652
## aac Asn   aac 28454 0.0214556741 1.1049453
## aag Lys   aag 13521 0.0101954793 0.4653348
## aat Asn   aat 23049 0.0173800461 0.8950547
## aca Thr   aca  9116 0.0068738991 0.5133967
## acc Thr   acc 31139 0.0234802922 1.7536924
## acg Thr   acg 19081 0.0143879847 1.0746075
## act Thr   act 11689 0.0088140639 0.6583034
## aga Arg   aga  2573 0.0019401648 0.2111584
## agc Ser   agc 21291 0.0160544302 1.6718055
## agg Arg   agg  1420 0.0010707478 0.1165351
## agt Ser   agt 11487 0.0086617463 0.9019787
## ata Ile   ata  5486 0.0041367058 0.2069902
## atc Ile   atc 33524 0.0252786960 1.2648816
## atg Met   atg 37007 0.0279050443 1.0000000
## att Ile   att 40501 0.0305396870 1.5281282
## caa Gln   caa 20402 0.0153840818 0.6939574
## cac His   cac 12890 0.0097196752 0.8594766
## cag Gln   cag 38397 0.0289531706 1.3060426
## cat His   cat 17105 0.0128979864 1.1405234
## cca Pro   cca 11163 0.0084174348 0.7606814
## ccc Pro   ccc  7238 0.0054577975 0.4932198
## ccg Pro   ccg 31074 0.0234312791 2.1174787
## cct Pro   cct  9225 0.0069560903 0.6286201
## cga Arg   cga  4619 0.0034829465 0.3790674
## cgc Arg   cgc 29441 0.0221999192 2.4161344
## cgg Arg   cgg  7079 0.0053379039 0.5809523
## cgt Arg   cgt 27979 0.0210975014 2.2961524
## cta Leu   cta  5149 0.0038825918 0.2179763
## ctc Leu   ctc 14811 0.0111682009 0.6270047
## ctg Leu   ctg 70714 0.0533217311 2.9935864
## ctt Leu   ctt 14586 0.0109985402 0.6174796
## gaa Glu   gaa 52679 0.0397224803 1.3801514
## gac Asp   gac 25347 0.0191128478 0.7477432
## gag Glu   gag 23659 0.0178400152 0.6198486
## gat Asp   gat 42449 0.0320085720 1.2522568
## gca Ala   gca 26743 0.0201654984 0.8481293
## gcc Ala   gcc 34117 0.0257258463 1.0819888
## gcg Ala   gcg 45082 0.0339939797 1.4297335
## gct Ala   gct 20185 0.0152204534 0.6401484
## gga Gly   gga 10350 0.0078043940 0.4257245
## ggc Gly   ggc 39536 0.0298120310 1.6262263
## ggg Gly   ggg 14581 0.0109947699 0.5997573
## ggt Gly   ggt 32779 0.0247169305 1.3482920
## gta Val   gta 14430 0.0108809087 0.6141144
## gtc Val   gtc 20350 0.0153448713 0.8660588
## gtg Val   gtg 34996 0.0263886543 1.4893658
## gtt Val   gtt 24213 0.0182577576 1.0304610
## taa Stp   taa  2726 0.0020555341 1.9292286
## tac Tyr   tac 16160 0.0121854113 0.8641480
## tag Stp   tag   294 0.0002216900 0.2080679
## tat Tyr   tat 21241 0.0160167278 1.1358520
## tca Ser   tca  9303 0.0070149060 0.7304874
## tcc Ser   tcc 11390 0.0085886036 0.8943621
```

```
## tcg Ser   tcg 11830 0.0089203846 0.9289117
## tct Ser   tct 11111 0.0083782243 0.8724546
## tga Stp   tga  1219 0.0009191842 0.8627035
## tgc Cys   tgc  8574 0.0064652052 1.1152445
## tgg Trp   tgg 20196 0.0152287479 1.0000000
## tgt Cys   tgt  6802 0.0051290326 0.8847555
## tta Leu   tta 18323 0.0138164165 0.7756807
## ttc Phe   ttc 21974 0.0165694448 0.8523496
## ttg Leu   ttg 18148 0.0136844582 0.7682723
## ttt Phe   ttt 29587 0.0223100101 1.1476504
```

The code uco(codon_E,index="rscu",as.data.frame=TRUE) quantify the codon usage bias among all coding sequences and give the values in a data frame.

Barnesiella intestinihominis

```
codon_B <- unlist(cdsb)

uco(codon_B,as.data.frame=TRUE)
```

```
##      AA codon   eff         freq      RSCU
## aaa Lys   aaa 44748 0.0436686422 1.4328071
## aac Asn   aac 21531 0.0210116549 0.8424203
## aag Lys   aag 17714 0.0172867240 0.5671929
## aat Asn   aat 29586 0.0288723618 1.1575797
## aca Thr   aca 13535 0.0132085249 0.9309764
## acc Thr   acc 18350 0.0179073832 1.2621660
## acg Thr   acg 14725 0.0143698211 1.0128280
## act Thr   act 11544 0.0112655494 0.7940296
## aga Arg   aga  6923 0.0067560117 0.8361953
## agc Ser   agc  9496 0.0092669488 0.8531896
## agg Arg   agg  3629 0.0035414656 0.4383291
## agt Ser   agt  9723 0.0094884734 0.8735849
## ata Ile   ata 23182 0.0226228315 0.9423322
## atc Ile   atc 25957 0.0253308962 1.0551340
## atg Met   atg 25323 0.0247121888 1.0000000
## att Ile   att 24663 0.0240681086 1.0025338
## caa Gln   caa 20924 0.0204192963 1.2739505
## cac His   cac  7564 0.0073815502 0.8111963
## cag Gln   cag 11925 0.0116373594 0.7260495
## cat His   cat 11085 0.0108176209 1.1888037
## cca Pro   cca  3282 0.0032028355 0.3325902
## ccc Pro   ccc 12546 0.0122433804 1.2713822
## ccg Pro   ccg 13109 0.0127928004 1.3284353
## cct Pro   cct 10535 0.0102808873 1.0675922
## cga Arg   cga  8020 0.0078265511 0.9686965
## cgc Arg   cgc  8472 0.0082676485 1.0232914
## cgg Arg   cgg  8908 0.0086931319 1.0759537
## cgt Arg   cgt 13723 0.0133919902 1.6575340
## cta Leu   cta  5261 0.0051341004 0.3449761
## ctc Leu   ctc 14128 0.0137872213 0.9264060
## ctg Leu   ctg 13534 0.0132075490 0.8874560
## ctt Leu   ctt 12563 0.0122599703 0.8237853
## gaa Glu   gaa 44135 0.0430704282 1.3202214
## gac Asp   gac 24298 0.0237119127 0.8346530
## gag Glu   gag 22725 0.0221768547 0.6797786
```

```
## gat Asp   gat 33925 0.0331067017 1.1653470
## gca Ala   gca 14811 0.0144537467 0.8369215
## gcc Ala   gcc 25256 0.0246468049 1.4271345
## gcg Ala   gcg 11077 0.0108098138 0.6259253
## gct Ala   gct 19644 0.0191701709 1.1100186
## gga Gly   gga 25553 0.0249366410 1.4456121
## ggc Gly   ggc 13664 0.0133344133 0.7730146
## ggg Gly   ggg 11723 0.0114402318 0.6632063
## ggt Gly   ggt 19765 0.0192882523 1.1181670
## gta Val   gta 19112 0.0186510032 1.1419182
## gtc Val   gtc 15436 0.0150636712 0.9222818
## gtg Val   gtg 16589 0.0161888599 0.9911721
## gtt Val   gtt 15810 0.0154286501 0.9446278
## taa Stp   taa  1721 0.0016794881 1.8419550
## tac Tyr   tac 14316 0.0139706865 0.6218940
## tag Stp   tag   411 0.0004010863 0.4398858
## tat Tyr   tat 31724 0.0309587915 1.3781060
## tca Ser   tca  5110 0.0049867427 0.4591195
## tcc Ser   tcc 10429 0.0101774441 0.9370171
## tcg Ser   tcg 19158 0.0186958936 1.7212938
## tct Ser   tct 12864 0.0125537100 1.1557951
## tga Stp   tga   671 0.0006548149 0.7181591
## tgc Cys   tgc  5606 0.0054707788 0.8672649
## tgg Trp   tgg 12820 0.0125107713 1.0000000
## tgt Cys   tgt  7322 0.0071453875 1.1327351
## tta Leu   tta 14512 0.0141619589 0.9515858
## ttc Phe   ttc 24893 0.0242925608 1.0633944
## ttg Leu   ttg 31504 0.0307440981 2.0657909
## ttt Phe   ttt 21925 0.0213961513 0.9366056
```

The code codon_B <- unlist(cdsb) unlist the cdsb and stores it in the codon variable and then calculates the codon usage using uco(codon_B,as.data.frame=TRUE) giving the values in a data frame.

```
uco(codon_B,index="rscu",as.data.frame=TRUE)
```

```
##      AA codon   eff         freq      RSCU
## aaa Lys   aaa 44748 0.0436686422 1.4328071
## aac Asn   aac 21531 0.0210116549 0.8424203
## aag Lys   aag 17714 0.0172867240 0.5671929
## aat Asn   aat 29586 0.0288723618 1.1575797
## aca Thr   aca 13535 0.0132085249 0.9309764
## acc Thr   acc 18350 0.0179073832 1.2621660
## acg Thr   acg 14725 0.0143698211 1.0128280
## act Thr   act 11544 0.0112655494 0.7940296
## aga Arg   aga  6923 0.0067560117 0.8361953
## agc Ser   agc  9496 0.0092669488 0.8531896
## agg Arg   agg  3629 0.0035414656 0.4383291
## agt Ser   agt  9723 0.0094884734 0.8735849
## ata Ile   ata 23182 0.0226228315 0.9423322
## atc Ile   atc 25957 0.0253308962 1.0551340
## atg Met   atg 25323 0.0247121888 1.0000000
## att Ile   att 24663 0.0240681086 1.0025338
## caa Gln   caa 20924 0.0204192963 1.2739505
## cac His   cac  7564 0.0073815502 0.8111963
## cag Gln   cag 11925 0.0116373594 0.7260495
```

```
## cat His   cat 11085 0.0108176209 1.1888037
## cca Pro   cca  3282 0.0032028355 0.3325902
## ccc Pro   ccc 12546 0.0122433804 1.2713822
## ccg Pro   ccg 13109 0.0127928004 1.3284353
## cct Pro   cct 10535 0.0102808873 1.0675922
## cga Arg   cga  8020 0.0078265511 0.9686965
## cgc Arg   cgc  8472 0.0082676485 1.0232914
## cgg Arg   cgg  8908 0.0086931319 1.0759537
## cgt Arg   cgt 13723 0.0133919902 1.6575340
## cta Leu   cta  5261 0.0051341004 0.3449761
## ctc Leu   ctc 14128 0.0137872213 0.9264060
## ctg Leu   ctg 13534 0.0132075490 0.8874560
## ctt Leu   ctt 12563 0.0122599703 0.8237853
## gaa Glu   gaa 44135 0.0430704282 1.3202214
## gac Asp   gac 24298 0.0237119127 0.8346530
## gag Glu   gag 22725 0.0221768547 0.6797786
## gat Asp   gat 33925 0.0331067017 1.1653470
## gca Ala   gca 14811 0.0144537467 0.8369215
## gcc Ala   gcc 25256 0.0246468049 1.4271345
## gcg Ala   gcg 11077 0.0108098138 0.6259253
## gct Ala   gct 19644 0.0191701709 1.1100186
## gga Gly   gga 25553 0.0249366410 1.4456121
## ggc Gly   ggc 13664 0.0133344133 0.7730146
## ggg Gly   ggg 11723 0.0114402318 0.6632063
## ggt Gly   ggt 19765 0.0192882523 1.1181670
## gta Val   gta 19112 0.0186510032 1.1419182
## gtc Val   gtc 15436 0.0150636712 0.9222818
## gtg Val   gtg 16589 0.0161888599 0.9911721
## gtt Val   gtt 15810 0.0154286501 0.9446278
## taa Stp   taa  1721 0.0016794881 1.8419550
## tac Tyr   tac 14316 0.0139706865 0.6218940
## tag Stp   tag   411 0.0004010863 0.4398858
## tat Tyr   tat 31724 0.0309587915 1.3781060
## tca Ser   tca  5110 0.0049867427 0.4591195
## tcc Ser   tcc 10429 0.0101774441 0.9370171
## tcg Ser   tcg 19158 0.0186958936 1.7212938
## tct Ser   tct 12864 0.0125537100 1.1557951
## tga Stp   tga   671 0.0006548149 0.7181591
## tgc Cys   tgc  5606 0.0054707788 0.8672649
## tgg Trp   tgg 12820 0.0125107713 1.0000000
## tgt Cys   tgt  7322 0.0071453875 1.1327351
## tta Leu   tta 14512 0.0141619589 0.9515858
## ttc Phe   ttc 24893 0.0242925608 1.0633944
## ttg Leu   ttg 31504 0.0307440981 2.0657909
## ttt Phe   ttt 21925 0.0213961513 0.9366056
```

The code uco(codon_B,index="rscu",as.data.frame=TRUE) quantify the codon usage bias among all coding sequences and give the values in a data frame.

6. Identifying 10 protein sequence k-mers of length 3-5 which are the most over- and under-represented k-mers in Barnesiella intestinihominis and E.coli

```
#
prots_B.intestinihominis <- unlist(prot_B.intestinihominis)
```

```
prots_E.coli <- unlist(prot_E.coli)
```

The translated protein sequences are converted to simple sequence vectors by unlist.

```
# Define amino acid alphabet
aa <- s2c("ACDEFGHIKLMNPQRSTVWY")

kmers_B.intestinihominis <- list()
kmers_E.coli <- list()

for (k in 3:5) {
  kmers_B.intestinihominis[[as.character(k)]] <- count(prots_B.intestinihominis, wordsize = k, alphabet

  kmers_E.coli[[as.character(k)]] <- count(prots_E.coli, wordsize = k, alphabet = aa, freq = TRUE)
}
```

k-mers (k = 3 to 5) are counted by count function.

```
top10_over_B.intestinihominis <- head(sort(kmers_B.intestinihominis[[as.character(k)]], decreasing = TRU
top10_over_B.intestinihominis
```

```
##
##        GSGKS        GKSTL        GKTTL        ILLLS        LKDAS        LLSLL
## 1.978823e-05 1.780940e-05 1.681999e-05 1.681999e-05 1.681999e-05 1.681999e-05
##        GVILI        LFLLL        LLLSL        LSGGE
## 1.583058e-05 1.583058e-05 1.583058e-05 1.583058e-05
```

```
top10_under_B.intestinihominis <- head(sort(kmers_B.intestinihominis[[as.character(k)]], decreasing = FA
top10_under_B.intestinihominis
```

```
##
## AAAAH AAAAR AAAAT AAACC AAACD AAACH AAACK AAACL AAACM AAACP
##     0     0     0     0     0     0     0     0     0     0
```

```
top10_over_E.coli <- head(sort(kmers_E.coli[[as.character(k)]], decreasing = TRUE), 10)
top10_over_E.coli
```

```
##
##        GKSTL        GSGKS        AALAA        LLAAL        SGSGK        LLLDE
## 4.444509e-05 3.218438e-05 2.835290e-05 2.835290e-05 2.835290e-05 2.682031e-05
##        LAAAL        LLLAL        LLLLL        LDEPT
## 2.605402e-05 2.605402e-05 2.605402e-05 2.528772e-05
```

```
top10_under_E.coli <- head(sort(kmers_E.coli[[as.character(k)]], decreasing = FALSE), 10)
top10_under_E.coli
```

```
##
## AAAAC AAAAW AAACC AAACH AAACI AAACK AAACM AAACN AAACP AAACS
##     0     0     0     0     0     0     0     0     0     0
```

10 protein sequence k-mers of length 3-5 which are the most over- and under-represented k-mers are identified by head command by sorting tge sequences.

```
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 22.04.5 LTS
##
```

```
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.10.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0
##
## locale:
##  [1] LC_CTYPE=C.UTF-8       LC_NUMERIC=C           LC_TIME=C.UTF-8
##  [4] LC_COLLATE=C.UTF-8     LC_MONETARY=C.UTF-8    LC_MESSAGES=C.UTF-8
##  [7] LC_PAPER=C.UTF-8       LC_NAME=C              LC_ADDRESS=C
## [10] LC_TELEPHONE=C         LC_MEASUREMENT=C.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] seqinr_4.2-36     R.utils_2.13.0    R.oo_1.27.1       R.methodsS3_1.8.2
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.1.0        digest_0.6.37     MASS_7.3-55       evaluate_1.0.5
##  [5] rlang_1.1.6       cli_3.6.5         rmarkdown_2.29    tools_4.1.2
##  [9] ade4_1.7-23       xfun_0.53         yaml_2.3.10       fastmap_1.2.0
## [13] compiler_4.1.2    htmltools_0.5.8.1 knitr_1.50
```