cchar, v. 1.13: Counting Characters in Sequence Data

Bernhard Haubold

March 14, 2019

1 Introduction

cchar is a computer program for counting characters in FASTA formatted sequence data. In contrast to the UNIX utility wc, which counts every character in a file, cchar only considers alphanumerical characters and ignores FASTA headers.

2 Getting Started

cchar was written in C on a computer running Linux; it is intended to run on any UNIX system with a C compiler. However, please contact me at haubold@evolbio.mpg.de if you have problems with the program.

- Unpack the program

  ```
  tar -xvzf cchar_XXX.tgz
  ```

  where XXX indicates the version.

- Change into the newly created directory

  ```
  cd Cchar_XXX
  ```

  and list its contents

  ```
  ls
  ```

- Generate cchar

  ```
  make
  ```

- List its options

  ```
  ./cchar -h
  ```

- Test the program

  ```
  cchar test.fasta
  ```

3 Change Log

1. v. 1.6 (November 7, 2010)
   - First version distributed.
2. v. 1.7 (April 14, 2012)
   - Fixed segmentation fault when processing of multiple files.
   - Improved handling of separate sequences (`-s` switch).
3. v. 1.8 (September 4, 2012)
   - Set the default buffer size to `BUFSIZ`, down from `32*BUFSIZ`, which caused problems when reading from `stdin`.
4. v. 1.9 (April 15, 2017)
   - Fixed handling of headers when using `-s` with multiple files.
   - Simplified code by using library function `getline`.
5. v. 1.10 (May 24, 2017)
   - Fixed bug in separate handling of sequences `-s`.
   - Use colon as field separator in conjunction with `-s`, which makes sorting for sequence length easier.
6. v. 1.11 (June 21, 2017)
   - Fixed printing of nucleotide counts for individual sequences when using `-s`.
7. v. 1.12 (November 6, 2018)
   - Fixed bug in `interface.c`
8. v. 1.13 (March 14, 2019)
   - Included `-E` for printing the exact number of characters, rather than formatting the count with `%g`, which can lead to truncated numbers.

4 Listings

4.1 The Driver Program: `cchar.c`

```
/***** cchar.c ************************************
 * Description: Count characters in FASTA files
 * Author: Bernhard Haubold, haubold@evolbio.mpg.de
 * File created on Mon Jul 19 22:06:24 2004.
 ************************************************/
#include <unistd.h>
#include <fcntl.h>
#include <string.h>
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
#include "cchar.h"
#include "interface.h"
#include "eprintf.h"
```

```
16 double totalChar(double *ch){
     int i;
     double s;

     s = 0;
21   for(i=0;i<NUMCHAR;i++)
       s += ch[i];
     return s;
   }

26 void printCchar(double *ch){
     int i;
     long s;

     printf("# Char\tCount\tFraction\n");
31   s = (long)totalChar(ch);
     for(i=0;i<NUMCHAR;i++)
       if(ch[i]>0)
         printf("%c\t%.0f\t%f\n",i,ch[i],ch[i]/(double)s);
   }
36
   int scanFile(Args *args, FILE *fp, short *dic, char *headerBuf, double *chs
       ){
     int i, j;
     int numSeq;
     ssize_t c;
41   size_t len = 0;
     double ch[NUMCHAR]; /* character count per sequence */
     char *line;

     /* initialize character array */
46   for(i=0;i<NUMCHAR;i++){
       ch[i] = 0.0;
       chs[i] = 0.0;  /* total sum of characters */
     }
     numSeq = 0;
51   line = 0;
     while((c = getline(&line, &len, fp)) != -1){
         if(line[0] == '>'){
           numSeq++;
           if(args->s && numSeq > 1){
56           if(args->E)
               printf("%s: %ld\n", headerBuf, (long)totalChar(ch));
             else
               printf("%s: %g\n", headerBuf, totalChar(ch));
             printCchar(ch);
61           }
           headerBuf = strncpy(headerBuf, line, args->b);
           headerBuf[c-1] = '\0';    /* remove newline character */
           for(j=0; j<NUMCHAR; j++){
             chs[j] += ch[j];
66           ch[j] = 0;
           }
         }else{
```

```
        for(i=0; i<c; i++){
          if(dic[(int)line[i]])
            ch[(int)line[i]]++;
        }
      }
    }
    if(args->s){
      if(args->E)
        printf("%s: %ld\n",headerBuf, (long)totalChar(ch));
      else
        printf("%s: %g\n",headerBuf,totalChar(ch));
      printCchar(ch);
    }
    for(j=0;j<NUMCHAR;j++)
      chs[j] += ch[j];
    return numSeq;
  }

  /* Count the occurrences of characters in an input sequence. */
  int main(int argc, char *argv[]){
    FILE *fp;
    int i, j, bufferSize;
    int numSeq;
    short *dic;
    char *version, *headerBuf;
    Args *args;
    double ch[NUMCHAR], chs[NUMCHAR];

    version = "1.13";
    args = getArgs(argc, argv);
    /* Create dictionary of alphanumerical characters. */
    dic = (short *)emalloc(sizeof(short)*NUMCHAR);
    for(i=0;i<NUMCHAR;i++){
      chs[i] = 0;
      if(isalnum((char)i))
        dic[i] = 1;
      else
        dic[i] = 0;
    }
    bufferSize = args->b;
    headerBuf = (char *)emalloc((bufferSize+1)*sizeof(char));
    numSeq = 0;
    if(args->h || args->e)
      printUsage(version);
    if(args->v)
      printSplash(version);
    if(args->numInputFiles == 0){
      fp = stdin;
      numSeq = scanFile(args, fp, dic, headerBuf, ch);
      printf("# Total number of input characters: %.0f\n", totalChar(ch));
      printCchar(ch);
    }else{
      for(i=0;i<args->numInputFiles;i++){
        fp = efopen(args->inputFiles[i],"r");
```

```
        if(args->numInputFiles > 1)
          printf("# %s\n",args->inputFiles[i]);
        numSeq += scanFile(args, fp, dic, headerBuf, ch);
126     for(j=0;j<NUMCHAR;j++){
          chs[j] += ch[j];
        }
        fclose(fp);
      }
131   if(!(args->s && numSeq < 2)){
        printf("# Total number of input characters: %.0f\n", totalChar(chs));
        printCchar(chs);
      }
    }
136 free(args);
    free(progname());

    return 0;
  }
```