`mutator`, v. 0.3: Mutate a DNA Sequence

Bernhard Haubold
Max-Planck-Institute for Evolutionary Biology, Plön, Germany

December 6, 2018

1 Introduction

`mutator` takes as input a FASTA formatted DNA sequence and prints out a mutated version. Only the residue designations `A`, `C`, `G`, and `T` may get mutated. All other characters are printed out unchanged.

2 Getting Started

`mutator` was written in C on a computer running Linux and should work on any standard UNIX system. However, please contact me at `haubold@evolbio.mpg.de` if you have any problems with the program.

- Unpack the program

  ```
  tar -xvzf mutator_XXX.tgz
  ```

  where `XXX` indicates the version.
- Change into the newly created directory

  ```
  cd Mutator_XXX
  ```

  and list its contents

  ```
  ls
  ```

- Generate `mutator`

  ```
  make
  ```

- List its options

  ```
  ./mutator -h
  ```

- To run the program, first inspect the test sequence

  ```
  cat test.fasta
  ```

  then generate a maximally mutated version

  ```
  ./mutator -m 1 test.fasta
  ```

3 Listing

The following listing documents the driver program for `mutator`.

```c
/***** mutator.c ***********************************
 * Description:
 * Author: Bernhard Haubold, haubold@evolbio.mpg.de
 * Date: Wed Aug  3 10:14:58 2016
 ***************************************************/
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include "interface.h"
#include "eprintf.h"
#include "gsl_rng.h"

gsl_rng *ran;
char dict[256];
int bufferSize = 1024;
char buffer[1024];

void initDict(){
  int i;
  char *dna = "ACGT";

  for(i=0;i<256;i++)
    dict[i] = 0;
  for(i=0;i<4;i++)
    dict[(int)dna[i]] = 1;
}

int printHeader(char *buf, int s){
  while(buf[s] != ’\n’ && buf[s] != ’\0’ && buf[s] != EOF)
    printf("%c", buf[s++]);
  printf("\n");
  return s;
}

char mutate(char c){
  char *dna = "ACGT";
  char m;
  int r;

  r = gsl_rng_uniform(ran) * 4;
  m = dna[r];
  while(m == c){
    r = gsl_rng_uniform(ran) * 4;
    m = dna[r];
  }
  return m;
}

void scanFile(int fd, Args *args){
  int c, i, p;
  p = 0;
  while((c = read(fd,buffer,bufferSize)) > 0){
    for(i=0;i<c;i++){
      if(buffer[i] == ’>’){
        p = 0;
        i = printHeader(buffer, i);
      }else if(dict[(int)buffer[i]]){
        if(gsl_rng_uniform(ran) < args->m || ++p == args->p)
          printf("%c",mutate(buffer[i]));
        else
          printf("%c",buffer[i]);
      }else{
        printf("%c",buffer[i]);
      }
    }
  }
}

int main(int argc, char ⋆argv[]){
  int i;
  char ⋆version;
  Args ⋆args;
  int fd;

  version = "0.3";
  setprogname2("mutator");
  args = getArgs(argc, argv);
  if(args->v)
    printSplash(version);
  if(args->h || args->e)
    printUsage(version);
  ran = ini_gsl_rng(args);
  initDict();
  if(args->numInputFiles == 0){
    fd = 0;
    scanFile(fd, args);
  }else{
    for(i=0;i<args->numInputFiles;i++){
      fd = open(args->inputFiles[i],0);
      scanFile(fd, args);
      close(fd);
    }
  }
  free_gsl_rng(ran, args);
  free(args);
  free(progname());
  return 0;
}
```

4 Change Log

- Version 0.1 (August 4, 2016)

  - First running version
- Version 0.2 (April 17, 2017)
  - Added `-p` option to mutate a specific position.
- Version 0.3 (April 17, 2017)
  - Reset position counter to make position-specific mutation universal for all input sequences in multi-fasta files.