pamPower, v. 0.7: Take PAM-1 Substitution Matrix to some Power

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

January 7, 2015

1 Introduction

pamPower is one of three programs for teaching how PAM matrices are computed. The other two programs are pamLog and pamNormalize. For details on how to apply these programs, see the chapter on PAM computation in my *Problems Book*.

2 Getting Started

pamPower 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 pamPower_XXX.tgz
  ```

  where XXX indicates the version.

- Change into the newly created directory

  ```
  cd PamPower_XXX
  ```

  and list its contents

  ```
  ls
  ```

- Generate pamPower

  ```
  make
  ```

- List its options

  ```
  ./pamPower -h
  ```

- Test program

  ```
  ./pamPower -n 70 pam1.txt
  ```

3 Listing

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

```
/***** pamPower.c ********************************
 * Description:
 * Author: Bernhard Haubold, haubold@evolbio.mpg.de
 * Date: Tue Jan  6 11:53:01 2015
 ***********************************************/
#include <stdio.h>
#include <stdlib.h>
#include "interface.h"
#include "eprintf.h"
#include "pam.h"

void scanFile(FILE *fp, Args *args);

int main(int argc, char *argv[]){
  int i;
  char *version;
  Args *args;
  FILE *fp;

  version = "0.7";
  setprogname2("pamPower");
  args = getArgs(argc, argv);
  if(args->v)
    printSplash(version);
  if(args->h || args->e)
    printUsage(version);
  if(args->numInputFiles == 0){
    fp = stdin;
    scanFile(fp, args);
  }else{
    for(i=0;i<args->numInputFiles;i++){
      fp = efopen(args->inputFiles[i],"r");
      scanFile(fp, args);
      fclose(fp);
    }
  }
  free(args);
  free(progname());
  return 0;
}

void scanFile(FILE *fp, Args *args){
  int i;
  SubstitutionMatrix *matA, *matB;

  matA = readSubstitutionMatrix(fp);
  matB = duplicateSubstitutionMatrix(matA);
  /* matrix multiplication */
  for(i=1;i<args->n;i++)
    matA = matrixMult(matA, matB);
  outputMatrix(stdout,matA,args->f);
```

```
}
```

4 Change Log

- Version 0.7 (January 6, 2015)
  - First version with reorganized code.
  - Reformatted the output for better legibility.