`td`, v. 0.3: Compute Tajima's D

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

June 8, 2011

1 Introduction

Tajima's D is a statistic for testing the null hypothesis of neutral evolution from samples of homologous DNA sequences (Tajima, 1989). The program `td` implements the computation of this statistic as a function of the average number of pairwise mismatches, π, the number of segregating sites, S, and the sample size, n.

2 Getting Started

`td` was written in C on a computer running Mac OS X 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 td_XXX.tgz
  ```

  where `XXX` indicates the version.

- Change into the newly created directory

  ```
  cd Td_XXX
  ```

  and list its contents

  ```
  ls
  ```

- Generate `td`

  ```
  make
  ```

- List its options

  ```
  ./td -h
  ```

3 Listing

The following listings document the driver program for `td` and the function for computing Tajima's D.

3.1 The Driver Program: `td.c`

```c
/***** td.c ****************************************
 * Description: Compute Tajima's D.
 * Reference: Tajima, F. (1989). Statistical method
 *    for testing the neutral mutation hypothesis
 *    by DNA polymorphism. Genetics, 123:585-595.
 * Author: Bernhard Haubold, haubold@evolbio.mpg.de
 * Date: Wed Jun  8 12:49:56 2011
 **************************************************/
#include <stdio.h>
#include "tajd.h"
#include "eprintf.h"
#include "interface.h"

int main(int argc, char *argv[]){
  Args *args;
  char *version;
  double td, a1, a2, b1, b2, c1, c2, e1, e2;

  version = "0.3";
  setprogname2("td");
  args = getArgs(argc, argv);

  if(args->v)
    printSplash(version);
  if(args->h || args->e)
    printUsage(version);
  td = tajd(args->n, args->s, args->p);
  printf("Tajima's␣D:␣%f\n",td);
  if(args->d){
    a1 = a1f(args->n);
    a2 = a2f(args->n);
    b1 = b1f(args->n);
    b2 = b2f(args->n);
    c1 = c1f(a1,b1);
    c2 = c2f(args->n,a1,a2,b2);
    e1 = e1f(a1,c1);
    e2 = e2f(a1,a2,c2);
    printf("a1:␣%.4f\n",a1);
    printf("a2:␣%.4f\n",a2);
    printf("b1:␣%.4f\n",b1);
    printf("b2:␣%.4f\n",b2);
    printf("c1:␣%.4f\n",c1);
    printf("c2:␣%.4f\n",c2);
    printf("e1:␣%.4f\n",e1);
    printf("e2:␣%.4f\n",e2);
  }
  return 0;
}
```

3.2 Calculating Tajima's D: `tajd.c`

```c
/***** tajd.c **************************************
 * Description: Calculate Tajima's D
 * when the number of sequences, the
 * number of segregating sites,
 * and the average pairwise differences
 * (pi) are known.  It also reports all
 * the coefficients for Tajima's
 * D (a1, a2, b1, b2, c1, c2, e1, e2).
 * Author: Bernhard Haubold, haubold@evolbio.mpg.de
 * Date: Wed Jun  8 13:14:24 2011
 ***********************************************/
#include <stdio.h>
#include <math.h>
#include "tajd.h"

double
tajd(int nsam, int segsites, double sumk)
{

  double  a1, a2, b1, b2, c1, c2, e1, e2;
  double td;

  if( segsites == 0 ) return( 0.0) ;
  a1 = a1f(nsam);
  a2 = a2f(nsam);
  b1 = b1f(nsam);
  b2 = b2f(nsam);
  c1 = c1f(a1, b1);
  c2 = c2f(nsam, a1, a2, b2);
  e1 = e1f(a1, c1);
  e2 = e2f(a1, a2, c2);

  td = (sumk - (segsites/a1))/sqrt((e1*segsites) + ((e2*segsites)*(segsites-1)));
  return td;
}

double a1f(int nsam)
{
  double a1;
  int i;
  a1 = 0.0;
  for (i=1; i<=nsam-1; i++) a1 += 1.0/i;
  return (a1);
}


double a2f(int nsam)
{
  double a2;
  int i;
  a2 = 0.0;
  for (i=1; i<=nsam-1; i++) a2 += 1.0/(i*i);
  return (a2);
}
```

```
double b1f(int nsam)
{
  double b1;
  b1 = (nsam + 1.0)/(3.0*(nsam-1.0));
  return (b1);
}


double b2f(int nsam)
{
  double b2;
  b2 = (2*(nsam*nsam + nsam + 3.0))/(9*nsam*(nsam - 1));
  return (b2);
}


double e1f(double a1, double c1)
{
  double e1;
  e1 = c1/a1;
  return (e1);
}

double e2f(double a1, double a2, double c2)
{
  double e2;
  e2 = c2/((a1*a1)+a2);
  return (e2);
}


double c1f(double a1, double b1)
{
  double c1;
  c1 = b1 - (1/a1);
  return (c1);
}


double c2f(int nsam, double a1, double a2, double b2)
{
  double c2;
  c2 = b2 - ((nsam+2)/(a1*nsam)) + (a2/(a1 * a1));
  return (c2);
}
```

4 Change Log

- Version 0.3 (June 8, 2011)
  - First version posted as part of `bioBox`.

References

F. Tajima. Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. *Genetics*, 123: 585–595, 1989.