Documentation `hotspot`, v. 0.3: Software to Support Sperm-Typing for Investigating Recombination Hotspots

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

January 14, 2016

1 Introduction

`hotspot` is a software package for detecting and analyzing recombination hotspots [**?**]. This is done in three steps: Preparation of allele-specific PCR, the actual PCR, and analysis of the PCR products. `hotspot` contains the two programs `asp` and `aso` for designing allele-specific primers and oligos used in the pre-PCR phase. In addition, it contains `xov` for computing the rate of recombination during the post-PCR phase of a project and the program `six` for simulating input to `xov`. Example input data and a script for downloading the mouse genome sequence including the corresponding SNP data are also provided.

2 Dependencies

- `libdivsufsort`: `https://github.com/y-256/libdivsufsort/`
- `tabix`: part of SAMtools [**?**], `http://www.htslib.org/`
- `gsl`: Gnu Scientific Library, `http://www.gnu.org/software/gsl/`

All three programs are also available via `apt-get` and similar package managers.

3 Installation

The four programs that make up `hotspot` are written in C on a computer running Linux and they should work on any UNIX system. However, please contact me at `haubold@evolbio.mpg.de` if you have any problems with the programs.

- Clone the `github` repository

  ```
  git clone https://github.com/evolBioInf/hotspot.git
  ```

- Change into the newly created directory `hotspot`

  ```
  cd hotspot
  ```

- Construct the configuration files

  ```
  autoreconf -i
  ```

- Configure the package

  ```
  ./configure
  ```

- Compile

  ```
  make
  ```

- Install

  ```
  sudo make install
  ```

  If instead of using `make install` you would like to install the binaries by hand, copy them from the `src_*` directories.

4 Getting Started

The following sections give a tutorial introduction to using the components of `hotspot`.

4.1 `aso`

`aso` is a program for designing allele-specific oligonucleotides that are complementary to SNPs contained in recombination hotspots. The program can also find universal oligos that do not intersect any known SNPs or indels.

- List options

  ```
  aso -h
  ```

- Take a look at two the example hotspot coordinates supplied with the program

  ```
  $ cat data/mus/exampleHotSpots19.txt
  # NB: These mouse hotspot data by Smagulova et al. (2011) are provided
  #   as part of the aspro software package for designing
  #   Allele-Specific PRimers and Oligos. The hotspot coordinates by
  #   Smagulova et al. refer to mm9 and will therefore not match the
  #   current mouse assembly.
  # Rerefence: Smagulova et al. (2011). Genome-wide
  #   analysis reveals novel molecular features of mouse recombination
  #   hotspots. Nature, 472:375-378.
  # int   chr     start   end
  int1    chr19   3796569 3799969
  int2    chr19   3804782 3808182
  ```

  Rows starting with a hash are comments and there can be as many comments as you like. This is followed by the hotspot data in four tab-delimited columns: Hotspot name (interval), chromosome, start, and end.

- The two example hotspots come from chromosome 19. To load the genome and SNP data for chromosome 19, execute

  ```
  make get-chr19-data-mus
  ```

- Verify that the genome data was downloaded:

  ```
  ls data/mus/genome/
  ```

- Check that the SNP data was downloaded:

  ```
  ls data/mus/vcf/
  ```

- Now run `aso`

```
aso -g data/mus/genome -s data/mus/vcf data/mus/exampleHotSpots19.txt |
head
int1  rs37745172   19  3796745  GGCATAAGCCTGTGGTT  GGCATAAGTCTGTGGTT
int1  rs214526021  19  3797057  GTGAGTTCGAGGCCAGC  GTGAGTTCAAGGCCAGC
int1  rs232006218  19  3797126  AGCTTTTCCTCTTTTCT  AGCTTTTCTTCTTTTCT
int1  rs254266733  19  3797132  TCCTCTTTTCTGCCTTT  TCCTCTTTCCTGCCTTT
int1  rs212133110  19  3797168  CAGCCTCCTTGTTACTC  CAGCCTCCCTGTTACTC
int1  rs243001379  19  3797209  TAGTAGCAGTTCGGCTC  TAGTAGCAATTCGGCTC
int1  rs37665146   19  3797213  AGCAGTTCGGCTCATAC  AGCAGTTCAGCTCATAC
int1  rs221355178  19  3797237  CATTGTTGCTGTTGCCA  CATTGTTGTTGTTGCCA
int1  rs232704341  19  3797253  ACAGTGGTTTCTTGTGT  ACAGTGGTCTCTTGTGT
int1  rs250384293  19  3797258  GGTTTCTTGTGTTTGGA  GGTTTCTTATGTTTGGA
```

The tab-delimited output consists of six columns:

1. Interval name
2. SNP name
3. Chromosome
4. SNP position
5. First oligo
6. Second oligo

The two oligos only differ in the middle, the SNP position.

- Instead of printing allele-specific oligos, `aso` can also find oligos that do not span any polymorphism. These “universals” are usually longer, say 100 bp (`-l`) and are constructed when using `-u`:

```
aso -u -l 100 -g data/mus/genome                \
-s data/mus/vcf data/mus/exampleHotSpots19.txt |
head
int1  19  3796570  3796669  0.894  GAA...
int1  19  3796869  3796968  1.000  GAG...
int1  19  3797168  3797267  0.981  TTG...
int1  19  3797467  3797566  0.941  GGG...
int1  19  3797766  3797865  1.000  CCG...
int1  19  3798065  3798164  1.000  GTG...
int1  19  3798364  3798463  0.914  TTT...
int1  19  3798663  3798762  0.894  TCT...
int1  19  3798962  3799061  0.382  GCA...
int1  19  3799261  3799360  1.000  TAA...
```

The columns indicate

1. Interval name
2. Start
3. End
4. Complexity. This measure lies between 0 and 1; a sequence consisting of a single nucleotide would have zero complexity; random sequences with equi-probable nucleotides have an expected complexity of 1. Complexities greater than 1 are truncated to 1. We can sort the output according to complexity:

```
aso -n -l 100 -g data/mus/genome                \
-s data/mus/vcf data/mus/exampleHotSpots19.txt |
sort -n -k 5                                   |
head
int1  19  3798962  3799061  0.382  GCA...
```

```
int1  19  3796570  3796669  0.894  GAA...
int1  19  3798663  3798762  0.894  TCT...
int1  19  3798364  3798463  0.914  TTT...
int1  19  3797467  3797566  0.941  GGG...
int1  19  3797168  3797267  0.981  TTG...
int1  19  3796869  3796968  1.000  GAG...
int1  19  3797766  3797865  1.000  CCG...
int1  19  3798065  3798164  1.000  GTG...
int1  19  3799261  3799360  1.000  TAA...
```

Notice the long microsatellelite in the top sequence.

4.2 asp

`asp` is a program for designing PCR primers that have a 3'-end complementary to SNPs in the regions flanking a recombination hotspot. The user can set a maximal and a minimal primer length. Within these bounds the program searches for the primer length that comes closest to an optimal GC-content, which the user can also set.

- Run `asp` to find forward primers in a window of 5kb upstream of the start of the candidate interval:

  ```
  asp -g data/mus/genome -s data/mus/vcf data/mus/exampleHotSpots19.txt | head
  int1  rs36588234   19  3791591  GATTCAGCCAACAA  0.43  34.4  GATTCAGCCAACAG  0.50  37.4
  int1  rs266164282  19  3791605  TTTAGATAGTTGGT  0.29  28.6  TTTAGATAGTTGGG  0.36  31.5
  int1  rs233070165  19  3791753  AGAAGAGCAGTCGG  0.57  40.3  AGAAGAGCAGTCGA  0.50  37.4
  int1  rs586881213  19  3791766  GGTGCTCTTACCCA  0.57  40.3  GGTGCTCTTACCCT  0.57  40.3
  int1  rs258256279  19  3791984  AGTTCCAGGACAGT  0.50  37.4  AGTTCCAGGACAGC  0.57  40.3
  int1  rs215047461  19  3792005  GCACATAAAAACTT  0.29  28.6  GCACATAAAAACTC  0.36  31.5
  int1  rs38604711   19  3792235  TGATTGCAGAAATT  0.29  28.6  TGATTGCAGAAATC  0.36  31.5
  int1  rs212603417  19  3792310  TTAAACCTCCCATT  0.36  31.5  TTAAACCTCCCATC  0.43  34.4
  int1  rs36316803   19  3792353  TATTGTCATGAGCA  0.36  31.5  TATTGTCATGAGCG  0.43  34.4
  int1  rs579758610  19  3792405  CATTACCCATGAGC  0.50  37.4  CATTACCCATGAGG  0.50  37.4
  ```

  The output columns are

  1. Interval name
  2. SNP name
  3. Chromosome
  4. SNP position
  5. Primer for first allele
  6. GC-content for first primer
  7. Melting temperature for first primer
  8. Primer for second allele
  9. CG-content for second primer
  10. Melting temperature for second primer

- To get the reverse primers in the upstream region, run

  ```
  asp -r -g data/mus/genome                        \
  -s data/mus/vcf data/mus/exampleHotSpots19.txt |
  head
  int1  rs249017060  19  3800119  AGAGTGAGTTCCAG       0.50  37.4  AGAGTGAGTTCCAA       0.43  34.4
  int1  rs218978784  19  3800120  CAGAGTGAGTTCCA       0.50  37.4  CAGAGTGAGTTCCC       0.57  40.3
  int1  rs235336981  19  3800166  GCATCCTTTAATCA       0.36  31.5  GCATCCTTTAATCC       0.43  34.4
  int1  rs584116440  19  3800176  CAGTGGTGATGCAT       0.50  37.4  CAGTGGTGATGCAC       0.57  40.3
  int1  rs253442444  19  3800434  CTCTCAAGTGCTGG       0.57  40.3  CTCTCAAGTGCTGA       0.50  37.4
  int1  rs214858261  19  3800484  TGCTCTATGAACCA       0.43  34.4  TGCTCTATGAACCT       0.43  34.4
  int1  rs232338632  19  3800485  TTGCTCTATGAACC       0.43  34.4  TTGCTCTATGAACT       0.36  31.5
  int1  rs261835571  19  3800506  TGGCTGTGGCTGTC       0.64  43.2  TGGCTGTGGCTGTT       0.57  40.3
  int1  rs222895727  19  3800511  TTATGTGGCTGTGG       0.50  37.4  TTATGTGGCTGTGT       0.43  34.4
  int1  rs579285301  19  3800535  TATTTATTTATTTTTGAGG  0.16  36.0  ATTTATTTATTTTTGAGA   0.11  32.1
  ```

  Notice that the two primers need not have the same length. By default, `asp` searches for a primer of length 14–19 with a GC-content as close to 0.5 as possible.

4.3 `xov`

`xov` implements a maximum likelihood procedure for estimating the number of crossover events from the observed number of positive and negative allele-specific PCR reactions. In addition, it uses the likelihood ratio method for estimating a confidence interval. By default this interval is set to 95%, but the user can set it to any level desired.

- Take a look at example input for `xov`

  ```
  cat data/mus/exampleResults.txt
  # This mock data set is taken from
  #   Kauppi et al. (2009). Analysis of human
  #   recombination products from human sperm.
  #   In: Keeney, S. (ed.), Meiosis, Volume 1,
  #   Molecular and Genetic Methods, Volume 557.
  # Int   Chr   Start   End   d|n-k|k    d|n-k|k    d|n-k|k
  Int1   1     1        500   2000|2|6   600|1|7    200|0|8
  Int2   1     1        250   2000|1|5   600|1|7    200|1|7
  Int3   1     1        250   2000|3|3   600|1|7    200|0|8
  Int4   1     1        500   2000|5|3   600|0|8    200|0|8
  Int5   1     1       1500   2000|0|8   600|0|8    200|0|8
  Int6   1     1        500   60|5|7     120|11|1   240|12|0
  ```

  The columns list

  1. Interval
  2. Chromosome
  3. Start
  4. End
  5. Number of molecules, number of positive PCR reactions, number of negative PCR reactions; this triplet of values is repeated for each experiment, three in this example

- Run `xov`

  ```
  xov data/mus/exampleResults.txt
  # Int   Chr   Start   End   Len   [%    %       %] [cM/Mb     cM/Mb    cM/Mb]
  Int1    1     1        500   500   0.004 0.015 0.039    7.448    30.004   78.119
  Int2    1     1        250   250   0.004 0.018 0.046   17.596    70.852  184.259
  Int3    1     1        250   250   0.008 0.027 0.063   33.291   107.923  253.801
  Int4    1     1        500   500   0.011 0.030 0.065   21.009    59.128  129.232
  Int5    1     1       1500  1500   0.000 0.000 0.009    0.000     0.000    5.716
  Int6    1     1        500   500   0.931 1.499 2.362 1862.604 2997.521 4723.249
  ```

  Two sets of confidence intervals are computed: one for the %-recombination frequency, the other for the standard measure of recombination, centi-Morgans per megabase (cM/Mb). These are obtained from the %-frequencies as

  $$\text{cM/Mb} = \frac{\text{\%-Freq}}{\text{Len}} \cdot 10^6.$$

4.4 `six`

`six` simulates typing data given some crossover rate. This can be used for checking the accuracy of `xov`, and for testing assay designs.

- Run `six`

  ```
  six
  # Int   Chr   Start        End          m|n-k|k m|n-k|k  m|n-k|k
  Int1  1     15,000,001  15,002,000  60|8|4  120|10|2 240|12|0
  ```

```
Int2  1      15,000,001  15,002,000  60|7|5  120|9|3  240|11|1
Int3  1      15,000,001  15,002,000  60|8|4  120|7|5  240|12|0
Int4  1      15,000,001  15,002,000  60|7|5  120|10|2 240|11|1
Int5  1      15,000,001  15,002,000  60|5|7  120|8|4  240|9|3
```

where the columns have the same meaning as explained above for `exampleResults.txt`.

- Pipe the results of `six` through `xov`

```
six | xov
# Int  Chr  Start    End      Len  [%    %     %] [cM/Mb  cM/Mb    cM/Mb]
Int1   1    15000001 15002000 2000 0.542 0.869 1.332 270.847 434.393  665.759
Int2   1    15000001 15002000 2000 0.895 1.404 2.142 447.448 701.838 1070.429
Int3   1    15000001 15002000 2000 0.688 1.083 1.645 343.656 541.430  822.117
Int4   1    15000001 15002000 2000 0.534 0.855 1.307 266.739 427.079  652.990
Int5   1    15000001 15002000 2000 0.905 1.447 2.259 452.265 723.175 1129.031
```

- Check accuracy of `xov`

```
six -r 1000 -x 1.5 |
./src_xov/xov      |
awk '!/^#/{s+=$7;c++}END{print s/c}'
1.59407
```

The simulated %-crossover frequency (`-x`) is 1.5, but the average estimate is 1.59. In other words, our estimator has an upward bias [**?**].

References