MapPop 1.0: Software for selective mapping and bin mapping

Dan Brown, Todd Vision

January 7, 2000

Contents

1 Introduction

1.1 Welcome to MapPop

MapPop is publicly available software that implements *selective mapping* (Vision *et al.* 1999, Brown *et al.* 2000) for Maximum Bin Length (MBL) and Expected Bin Length (EBL) and also implements *bin mapping* (Brown and Vision 1999). These are methodologies that aid in the construction of high density whole-genome maps and may also be used in other genetic mapping contexts. In selective mapping, one chooses a globally high-resolution mapping sample from a large, sparsely genotyped mapping population. All subsequent markers to be mapped are genotyped only in the selected individuals. This allows many markers to be placed on a high resolution map with a minimum of genotyping. Bin mapping is used to order newly genotyped markers relative to a high-confidence framework map and to each other. Bin mapping may also be used, under certain assumptions, to estimate genetic distances between newly genotyped markers. The logic behind these methods and the algorithms employed in this software are more fully described in the literature cited above. This document is meant to serve as a companion guide for users of the software who are already familiar with the underlying principles.

1.2 License Agreement

MapPop, including its documentation, source code, executables (referred hereafter as MapPop), is distributed under the following license terms at no charge. Installation of the program on any computer or any use of the program implies that the user and the user's organization (herein referred to as "you") agree to the following terms:

This software and documentation may be freely distributed. This software is provided on an "as is" basis, with no warranty of any type, including warranty of suitability for any particular purpose or ability to function correctly on any type of computer. No technical support can be guaranteed. In particular, no warranty is made that the algorithms contained within this program are of utility for the research purposes for which they were developed.

You may redistribute MapPop. However, the entire package, including documentation, software, this license, and source code, must be preserved.

You may modify MapPop, and distribute your results, but you must (a) preserve all copyright notices, license agreements and credits in software and documentation, (b) add your own notice which makes it clear immediately that it is a modified version, (c) also distribute the unmodified version along with your modified version, (d) distribute the modified version under this licensing agreement, and (e) notify the copyright holders of MapPop that you are distributing a modified version, and supply us a full copy of source code. Note that this precludes selling a modified version of MapPop.

You may also not disassemble the MapPop executables, since this would involve violating the license for the Matlab libraries with which they are built.

2 Version History

This documentation covers MapPop version 1.0, released in December of 1999.

2.1 Availability

Binaries, source code and documentation may be obtained from:
http://ars-genome.cornell.edu/software.html.

At present, the MapPop executable is only available for Windows 32-bit operating systems (including Microsoft Windows 95, 98, 2000 and NT). The archive containing this executable is 'mappop.zip'.

Those with access to Matlab (on any platform) may download the *.m and *.c source files that allow MapPop to be run as a Matlab script. Up-to-date details are available the 'readme' file at the MapPop website. We encourage users who feel confident with Matlab to experiment with these files. They are largely self-documenting.

Those with access to the Matlab compiler may download the (different) set of source files and use these to generate a free-standing executable. Please see the documentation in the source files and that from Mathworks, Inc. for further guidance. We'd be delighted to hear from users who have access to Matlab on platforms for which MapPop executables are not currently available. If you enter this territory, please familiarize yourself with the license restrictions regarding redistribution.

2.2 Contact Information

Further information can be obtained from the authors:

Daniel G. Brown
Dept. of Computer Science
Cornell University
Ithaca, NY 14853
fax: (607) 255-4428
email: snowman@cs.cornell.edu

Todd J. Vision
USDA-ARS Center for Bioinformatics and Comparative Genomics
604 Rhodes Hall
Cornell University
Ithaca, NY 14853
email: tv23@cornell.edu

3 Installation

3.1 Windows95/98/NT

1. Save the compressed archive (mappop.zip) to your local hard drive.

2. Extract the files to a single directory (best named MapPop) using your favorite unzipping utility (*e.g.* WinZip). The files in this archive include:
   a. mappop.exe - the executable itself
   b. mappop.pdf - this documentation in Adobe Portable Document Format (pdf)
   c. barleyframe.txt - a sample file for selective mapping
   d. barleynew.txt - a sample file for bin mapping
   e. libmat.dll - a Matlab library
   f. libmatlb.dll - ditto
   g. libmmfile.dll - ditto
   h. libmax.dll - ditto
   i. libut.dll - ditto

3. If a full installation of Matlab is present on your system, you may wish to discard the *.dll files and set your path to include the directory in which they are found. Otherwise, we recommend keeping the *.dll files in the same directory as the executable or in another directory that is in your path. The *.dll files are only installed in your Matlab installation if you have purchased the Matlab C/C++ library.

4. Set your path to include the MapPop directory so that the executable may be launched from the command shell in whatever directory you happen to be working. For guidance, see your Windows documentation or ask your local systems administrator.

5. Invoke the executable by typing “mappop” in the command shell. Alternatively, double-clicking on the icon will open up a console in which MapPop may be run. The disadvantage to the latter option is that the console must be configured by right-clicking on the title bar, selecting “properties”, and adjusting the settings. Otherwise, the interface will be unusable.

4 Is My Data Appropriate for Selective Mapping?

One should consider the following (non-exhaustive) list of factors which can affect the success of selective mapping:

4.1 Types of Mapping Populations

The source of those breakpoints in the mapping population (*i.e* recombinant *vs.* radiation-induced) and the particular crossing scheme or pedigree are not, in themselves, relevant (or even visible) to MapPop. Similarly, the distances between framework markers may be derived using any mapping function. However, the calculations made by MapPop itself assume a uniform distribution of breakpoint placement within each framework interval. It may be that thi sassumption is not warranted in certain datasets. MapPop 1.0 further assumes that each breakpoint occurs at a unique site. For this reason, MapPop 1.0 should not be applied to pedigrees in which breakpoints may be identical-by-descent. Users who would like to select from a multigenerational or inbred pedigree are encouraged to contact the authors directly.

4.2 Dominant Markers

There is no loss of information when dominant markers are genotyped in a recombinant inbred population or with the use of markers that are dominant for the non-recurrent parent in a backcross. But, in many cases, dominant markers will introduce ambiguities concerning breakpoint placement. MapPop handles dominant markers by positing the minimal number of breakpoints needed to account for the data. The presence of many dominant markers in a segregating cross will limit the success of sample selection. For this reason, their use in a framework map should be as markers of last resort.

4.3 Missing Data and Errors

MapPop is robust to a certain level of error and missing data in the framework map and will, in fact, flag many suspected errors that are detected in the bin mapping mode. But a great many missing or erroneous genotypes for new markers on small selected samples (of size less than 30 or 40) can lead to catastrophic mapping errors. If re-scoring missing or suspect genotypes is not practical given the genotyping technology you are using, then it is best to simply avoid the use of small selected samples. The lower the polymorphic information content per locus in the mapping population being used, the more of a concern this should be; backcross populations are particularly problematic in this regard.

4.4 Gaps and How to Treat Them

Even the most carefully produced genetic maps contain some intermarker intervals that are longer than strictly desirable. This has two negative consequences. First, the increased probability of multiple breakpoints in long intervals undermines MapPop's ability to account for all the breakpoints in each individual. Second, when breakpoints are detected in long intervals, their location is imprecisely known. Thus, breakpoints in long intervals will contribute to the uncertainty in estimating the resolution of the map. However, the presence of a small number of long gaps does not appear to have a major effect on sample selection quality in those populations that the authors have anlayzed. The best course of action appears to be (1) close gaps where possible but (2) if not possible, do not artificially break a linkage group when the markers flanking the gap provide any information content at all (*i.e.* are less than 50 cM or 100 cR apart).

4.5 Breakpoint Density and Genome Length

The presence of mapping samples with good resolution relative to the full population is inversely related to the breakpoint density within each individual and to the length of the genome. The former can be controlled by the crossing scheme or pedigree. (However, we note that choosing a low-resolution crossing scheme for the sole purpose of applying selective mapping would be simply tossing the baby out with bathwater). We strongly recommend that the user take advantage of the simulation features of MapPop to determine the appropriate crossing scheme, base population size and sample size, since the breakpoint density and genome length are quantities that can be estimated prior to any commitment to a particular population.

4.6 Framework Marker Density

The expected Bin Length is far more robust to sparse framework density than is maximum (although at very sparse densities, the calculated EBL of a sample is likely to be a slight overestimate of the actual EBL). For EBL, therefore, framework densities of 10-20 cM are sufficient. For MBL, one can select a higher quality sample by using more densely packed framework markers. This is a variable which, at least for MBL, is best explored by simulation.

4.7 Base Population vs. Sample Size

The issue of how large a base population to start with and how large a sample to select is largely a matter of what resources are available for the project, the resolution desired and the the number of markers to be placed. This issue can be easily explored by simulation, as shown in a tutorial example below.

5 Input

If MapPop is not being used in simulation mode, it will be necessary to supply either one of two file types as input. To implement selective mapping, MapPop requires a *framework map* file. To implement bin mapping, MapPop requires a *new marker* file. Both files are easily generated and edited using standard spreadsheet software.

5.1 Framework Map File

The framework map file is an ASCII text file with character strings delimited by whitespace (either tabs or spaces). It should contain the following columns:

1. the name of each *framework marker*

2. the linkage group of each marker

3. the position of each marker (in map units) from one end of the linkage group

4 through end. the genotype at each marker in each individual of the *base population*.

The first line of the file contains column headings, including the names or identifiers of all the individuals in the base population. Names of individuals must be distinct from one another as should the names of markers. All headings and entries must consist of strings uninterrupted by whitespace. Genotypes must be a single character. Genotypes need not be delimited by whitespace, although they may be.

Here is a portion of the sample file “barleyframe.txt”. This data comes from chromosome VII of the IGRI x Franka cross described in Graner *et. al.* 1994. The “1”, “2”, “3”, *etc.* in the first lin, are the names (admittedly fairly boring ones) of each of the backcross individuals in this dataset. “ABG312”, “MWG530”, *etc.*, the entries in the first column, are the names of the markers. Since these data come from a backcross, the genotypes in this population are either “1” (homozygous for the recurrent parent), “2” (heterozygous) or “-” (missing). We could just as well have used any other three characters in place of “”, “2” and “-”, since the conversion string for these characters is specified on the command line when the file is fed to MapPop.

marker	chrom	site	1	2	3	4	5	6	7	8
ABG312	7	0	1	1	1	1	1	2	1	1
Ris45a	7	10.3	1	1	1	1	1	2	1	1
MWG530	7	21.8	1	1	1	1	1	2	1	2
cMWG703	7	33.4	1	2	1	1	1	2	1	2
MWG564	7	40.4	1	2	1	1	1	2	1	2
MWG2291	7	53.1	1	2	1	2	2	2	1	2
MWG836	7	72.7	1	2	2	2	2	2	1	2
Brz	7	84.4	1	1	2	2	2	2	1	2
MWG2301	7	103.1	1	1	2	2	2	2	2	-
MWG626	7	113.3	1	1	2	2	2	2	2	2
PBI19	7	123.7	1	1	2	2	2	2	2	2
cMWG696	7	148.9	1	2	2	2	-	2	2	2
MWG2269	7	164.3	1	2	2	2	2	2	2	2
MWG2262a	7	175.7	1	1	2	2	2	2	2	2
MWG2062	7	200.1	1	1	2	1	2	2	2	1

One is unlikely to come into contact with the hard upper limit to the size of this dataset, which is determined by Matlab's matrix-handling library functions. However, a large dataset will cause a significant slowdown in the running time of MapPop. The test dataset here is quite small compared to the framework map datasets that MapPop has been designed to handle (on the order of 500 individuals and 500 markers).

5.2 New Marker File

The new marker file is similar to the framework map file but does not columns for the linkage group and map position of the new markers (since, of course, that is what one wishes to determine). A portion of the "barleynew.txt" file, again from chromosome VII of the IGRI x Franka cross, is shown below. Note that only those individuals in the selected sample need be present in this file.

marker	1	2	3	4	5	6	7	8	9	10
Ris17a	1	1	1	1	1	2	1	1	1	1
MWG851a	1	1	1	1	1	2	1	1	1	2
MWG555a	1	1	1	1	1	2	1	1	2	2
MWG2074	1	1	1	1	1	2	1	2	2	2
MWG807	1	1	1	1	1	2	1	2	2	2
PBI35	1	1	1	1	1	2	1	2	2	2
MWG2080	1	1	1	1	1	2	1	2	2	2
ABC151a	1	2	1	1	1	2	1	2	2	2
cMWG773	1	2	1	1	1	2	1	2	2	2

The names of the individuals must correspond with those in the framework map file and marker names should be distinct from those in the framework map file (unless you choose to include a framework marker in the new marker file for testing purpoposes).

6 The MapPop Command-line Interface

Valid MapPop commands include LOADFRAME, LOADNEW, SAMPLEMAX, ADDNEW, SET, SIMFRAME, SIMNEW, HELP, QUIT, DISPLAYFRAME, and SAMPLEEXP. Note that you may type any prefix of a command name that uniquely identifies it, for example “loadf” for LOADFRAME. Case does not matter.

6.1 The ADDNEW Command

The ADDNEW command places new markers onto an existing framework map. For details on how this works, we suggest reading the paper by Brown and Vision (1999).

MapPop’s ADDNEW command works somewhat differently for simulated data than for data that was loaded into memory from a file. First, if a sample was determined from the framework map, the population is limited to just that sample. Second, it determines detailed information about the quality of the mapping, given that the actual sites of all markers are available.

Syntax:

```
ADDNEW <filename>
```

If the optional filename parameter is included, all mapping output is sent to that file as well as the screen. This is easily parsed by, for example, perl scripts.

6.2 The DISPLAYFRAME Command

This function is disabled in the executable form of MapPop and is only available from within Matlab. The DISPLAYFRAME command displays a graphical quick-and-dirty representation of a framework map, to guarantee that it has been properly loaded.

To close the image, simply select ‘Close’ from the image’s File menu; you may also print it from that menu.

Syntax for DISPLAYFRAME

```
DISPLAYFRAME
```

6.3 The DISPLAYOPTS command

The DISPLAYOPTS command shows the current value of all MapPop global options.

Syntax for DISPLAYOPTS

```
DISPLAYOPTS
```

6.4 The HELP Command

The HELP command will provide a list of the available commands and help sebmenus. Typing

```
HELP <COMMAND>
```

will bring up specific information on the function and use of a particular command. In addition, HELP is available for the following topics:

HELP LICENSE: information about the license and lack of warranty.

HELP AUTHORS: contact information for MapPop's creators.

HELP DISTRIB: to obtain the latest release.

HELP CREDITS: for the credits.

6.5 The LOADFRAME Command

The LOADFRAME command directs MapPop to read a file containing information for a framework map.

A framework map file has two parts, a header line, which identifies the names of the organisms described in the file, and marker lines, each of which identifies the location of a framework marker, and its genotype for each of the population members. More detail about framework map files is available in the MapPop manual.

Syntax for LOADFRAME:

```
LOADFRAME <filename> <GenotypeString>
```

The filename must be the complete path to the framework map file.

The optional GenotypeString parameter identifies the characters used in the framework map to indicate the genotype of each marker/population member pair.

Example: loadframe foo AB-abH

This loads the framework map in file foo, where the genotype codes are as follows:
A = homozygous parent 1
B = homozygous parent 2
- = missing data
a = dominant marker, parent 1
b = dominant marker, parent 2
H = segregating site.

6.6 The LOADNEW Command

The LOADNEW command directs MapPop to read a file containing information about new markers to be imposed on an already-loaded framework map. The population members used in typing the new markers must all be members of the framework mapping population, though this is tested when the ADDNEW command is executed.

Syntax for LOADNEW:

```
LOADNEW <filename> <GenotypeString>
```

The filename must be the complete path to the framework map file.

The optional GenotypeString parameter identifies the characters used in the framework map to indicate the genotype of each marker/population member pair.

Example: loadnew foo AB-abH

This loads the new markers in file foo, where the genotype codes are as follows:
A = homozygous parent 1

B = homozygous parent 2
- = missing data
a = dominant marker, parent 1
b = dominant marker, parent 2
H = segregating site.

6.7 The QUIT Command

The QUIT command ends a MapPop session. No files are saved; there is no continuity between consecutive MapPop sessions.

Syntax for QUIT

```
QUIT
```

6.8 The SAMPLEMAX Command

The SAMPLEMAX command chooses a sample from an already-loaded framework map, attempting to choose a sample with minimum expected maximum bin size; this is approximately equivalent to attempting to minimize the maximum error made in mapping, though the two are not the same. For detailed explanation of the goals of this optimization procedure, see the MapPop manual, and the references it cites.

Syntax for SAMPLEMAX

```
SAMPLEMAX <size> <time> <choices>
```

The size parameter is the size sample desired. If not supplied, the sample chosen will be a SampleRatio fraction of the whole population. SampleRatio is a global MapPop parameter; type HELP SET for information about setting this parameter. By default, it is 0.5.

The time parameter is the number of seconds spent finding the sample. If not supplied, the program will spend Time seconds searching, where Time is also a MapPop global parameter, with default value 300 seconds.

The choices parameter is used directly in sampling. In choosing samples, sometimes the algorithm chooses a new sample member randomly from among the top choices; if choices = 5, then one of the top 5 choices is chosen. If not supplied, there is a MapPop global parameter Choices, which defaults to 5.

Example: samplemax 30 5 2

This takes five seconds to choose a sample of size 30; when choices are made, they are made from the best 2 possibilities.

6.9 The SAMPLEEXP Command

The SAMPLEEXP command chooses a sample from an already-loaded framework map, attempting to choose a sample with minimum expected bin size; this is approximately equivalent to attempting to minimize the average error made in mapping, though the two are not the same. For detailed explanation of the goals of this optimization procedure, see the MapPop manual, and the references it cites.

Syntax for SAMPLEEXP

```
SAMPLEEXP <size> <time> <choices>
```

The size parameter is the size sample desired. If not supplied, the sample chosen will be a SampleRatio fraction of the whole population. SampleRatio is a global MapPop parameter; type HELP SET for information about setting this parameter. By default, it is 0.5.

The time parameter is the number of seconds spent finding the sample. If not supplied, the program will spend Time seconds searching, where Time is also a MapPop global parameter, with default value 300 seconds.

The choices parameter is used directly in sampling. In choosing samples, sometimes the algorithm chooses a new sample member randomly from among the top choices; if choices = 5, then one of the top 5 choices is chosen. If not supplied, there is a MapPop global parameter Choices, which defaults to 5.

Example: SAMPLEEXP 30 5 2

This takes five seconds to choose a sample of size 30; when choices are made, they are made from the best 2 possibilities.

6.10 The SET Command

The SET command allows the user to change the values of MapPop's pre-set default parameters.

Syntax for SET

```
SET <variable> <new-value>
```

The variable parameter is the name of the variable which needs to be changed.

The new-value parameter is the value to be assigned to variable. The format for this depends on the parameter being set. Do HELP SET [variable] for information about the format (and function) of a MapPop global parameter

MapPop's variables are:

Choices - The Choices variable is the default number of different options the greedy algorithm chooses at each step in the first half of computing a new sample. To change this variable, do SET Choices [new-value], where new-value is an integer. The default is 5.

ErrorRate - The ErrorRate variable is the expected frequency of errors in the data. This is used when placing markers; double recombinants are indistinguishable from errors, but often errors are actually more common. To change this variable, do SET ErrorRate [new-value], where new-value is a real number. The default is 0.01.

FrameDist- The FrameDist variable is the default distance between framework markers in a simulated map. To change this variable, do SET FrameDist [new-value], where new-value is a real number. The default is 0.15.

GenomeLength - The GenomeLength variable is the default length of a simulated genome; there is expected to be one breakpoint per unit. To change this variable, do SET GenomeLength [new-value], where new-value is a real number. The default is 10.0.

NewDist - The NewDist parameter is the default distance between consecutive new markers in a simulated population. To change this variable, do SET NewDist [new-value], where new-value is a real number. The default is 0.02.

PopSize - The PopSize variable is the default number of members in a simulated population. To change this variable, do SET PopSize [new-value], where new-value is an integer. The default is 100. Command completed successfully.

QuickInsts - The QuickInsts variable is the number of instantiations of a loaded framework map which are used in choosing new samples. During the process of the greedy algorithm, the effect of adding a new population member is computed on all of the instantiations, and the added member will be one of the best found. To change this variable, do SET QuickInsts [new-value], where new-value is an integer. The default is 10. Command completed successfully.

SampleRatio - The SampleRatio variable is the default fraction of a population which is assigned to a sample by the SAMPLEEXP and SAMPLEMAX commands. To change this variable, do SET SampleRatio [new-value], where new-value is a real number. The default is 0.5. Command completed successfully.

TestInsts - The TestInsts variable is the number of instantiations used in selecting a population sample; the chosen population sample will be the sample found with the best mean performance against all TestInsts instantiations. To change this variable, do SET TestInsts [new-value], where new-value is an integer. The default is 50. Command completed successfully.

Time - The Time variable is the default number of seconds the sample selection algorithms take to find a mapping sample. To change this variable, do SET Time [new-value], where new-value is an integer. The default is 300. Command completed successfully.

TypeString - The TypeString variable is the default coding string for new genotypes; each marker/population member pair is coded as one of the six characters in the string. The default is 012345, which means:
0 = homozygous parent 1
1 = homozygous parent 2
2 = missing data
3 = dominant marker, parent 1
4 = dominant marker, parent 2
5 = segregating site.
To change this variable, do SET TypeString [new-value], where new-value is a six-character string, as above. Command completed successfully.

6.11 The SIMFRAME Command

The SIMFRAME command generates a simulated framework map for analysis of the quality of methods found in MapPop. All markers are placed in a single linkage group, and the framework includes the two ends of the genome.

Breakpoints are assumed to be generated by a Poisson process, with one breakpoint expected per unit (i.e., 1 unit = 100 cM).

Syntax for SIMFRAME

```
SIMFRAME <popsize> <genomelen> <nummarkers> <even> <seed>
```

The popsize parameter is the size of the simulated population. If not supplied, the population will have size PopSize. PopSize is a global MapPop parameter; type HELP SET for information on how to chang its value. By default, it is 100.

The genomelen parameter is the length of the simulated genome, i.e., the expected number of breakpoints per population member. If not supplied, the MapPop parameter GenomeLen is used; defaulting to 10.0 M, this can be changed with the SET command.

The nummarkers parameter is the number of markers in the framework, including markers for either end of the genome. If not supplied, there will be one marker every FrameDist units. FrameDist is a MapPop parameter and can be changed with the SET command. Its default is 0.15 units (i.e., 15 cM).

The even parameter indicates whether markers are regularly placed or not. If set to 1, markers are evenly placed along the genome, while if it is 0, the markers are uniformly placed at random. The default is 1.

The seed parameter is the seed for the random number generator. If not supplied, the seed is just the next element in Matlab's random number generator's sequence.

6.12 The SIMNEW Command

The SIMNEW command adds simulated new markers to a simulated framework map, to validate MapPop's use in mapping new markers.

Syntax for SIMNEW

```
SIMNEW <markers> <even> <seed>
```

The markers parameter is the number of new markers to be added to the map. If not supplied, the markers are placed evenly, separated by the distance found in MapPop parameter NewDist. This defaults to .02 units (2 cM), and can be changed; type HELP SET for help on changing parameters.

The even parameter indicates whether markers are evenly spaced, the default and the value when it is set to 1, or uniformly distributed at random along the whole genome.

The optional seed parameter is the seed for the random number generator. If not supplied, the next seed in Matlab's random number generator's sequence is used.

7 Tutorial Example

This tutorial will cover the basics of selective mapping and bin mapping, both on simulated and real data, using MapPop.

Provided either that you are in the directory containing the executable or that the same directory is included in your path, you may start the program by typing “MapPop” on the command line of your shell (case insensitive in Windows).

7.1 Simulation Mode

We begin by investigating the help system.

```
MapPop [1]: help
Valid MapPop commands (type HELP (commandname) for more details):
LOADFRAME, LOADNEW, SAMPLEMAX, ADDNEW, SET, SIMFRAME, SIMNEW,
HELP, QUIT, DISPLAYFRAME, and SAMPLEEXP.
Note that you may type any prefix of a command name that
uniquely identifies it, for example ‘‘loadf’’ for LOADFRAME.
Case does not matter.
HELP LICENSE: information about the license and lack of warranty.
HELP AUTHORS: contact information for  MapPop’s creators.
HELP DISTRIB: to obtain the latest release.
HELP CREDITS: for the credits.
Command completed successfully
```

To get detailed instructions on the syntax of the SIMFRAME command, we type:

```
MapPop[2]: help simframe
The SIMFRAME command generates a simulated framework map for
analysis of the quality of methods found in MapPop.  All markers
are placed in a single linkage group, and the framework includes
the two ends of the genome.

Breakpoints are assumed to be generated by a Poisson process,
with one breakpoint expected per unit (i.e., 1 unit = 100 cM).

Syntax for SIMFRAME
   SIMFRAME <popsize> <genomelen> <nummarkers> <even> <seed>
```

etc.

We are going to generate a population of size 200, with a 1,000 cM genome and framework markers spaced on average every 10 cM. Note that MapPop simulates one long linkage group (a simplification which makes little difference unless one desires to simulate an organism with extremely small chromosomes). Note also that the [genomelen] parameter entered in Morgans (not centiMorgans) and that the [nummarkers] parameter is also entered in Morgans (and not the actual number of markers on the map). We tell MapPop to space the markers according to a uniform distribution - not strictly evenly. We supply a random number seed in this case, although this is generally only done for testing purposes.

```
MapPop [3]: simframe 200 10 .1 0 123456
Command completed successfully
```

Now that MapPop knows what framework population to use, it can select a sample that minimizes either the maximum bin length or expected bin length criteria. We select a sample of size 50 using the latter criteria. We instruct MapPop to spend 20 seconds searching. During this time, MapPop iterates a randomized greedy algorithm, starting from an empty sample and adding one member to it at each step by choosing from among the best [choices] lines. (If this were SAMPLEMAX rather than SAMPLEEXP, an additional optimization step would follow). The algorithm will iterate the greedy algorithm as many times as the [time] parameter allows and retain the best sample found.

```
MapPop [4]: SAMPLEEXP 50 20 3
Choosing sample to minimize expected bin length.
Size desired: 50
Time allowed: 20 sec
Number of iterations: 70
Expected bin length : 0.0312533 map units
Sample lines:
SIM-LINE-5
SIM-LINE-6
SIM-LINE-7
SIM-LINE-9
SIM-LINE-12
SIM-LINE-16
SIM-LINE-24
SIM-LINE-25
SIM-LINE-26
SIM-LINE-27
```

etc.

In this case, MapPop does 70 iterations. The time required for a single iteration is very dependent on the size of the dataset. We strongly recommend initial testing to determine how long a single iteration lasts and how many iterations are necessary for convergence given the dataset and the parameter settings being used.

The selected sample found has an expected bin length of 3.1 cM. The fifty lines in the sample are listed in the output.

Now we add new markers to the selected sample as if we had gone away for some time and done further genotyping. Here, we tell MapPop to add 100 new markers, to space them uniformly (not evenly) along the linkage group and to use the random number seed 654321.

```
MapPop [5]: simnew 100 0 654321
Command completed successfully.
```

At this point, we are ready to do bin mapping with the ADDNEW command. This command takes an outfile name as an optional parameter. MapPop will send output to this file, creating it if it does not exist and overwriting it if it does. This file will be in the same directory and can be opened with any text viewer or editor. Whether or not a file is specified, the output will also be sent to STDOUT. STDOUT is your terminal by default - but it may be redirected (see below).

```
MapPop [6]: addnew map.out
Preliminary placement of markers into intervals...0.882 sec.
Correction of uncertain framework genotypes...0.050 sec.
Check for spurious double crossovers...1.091 sec.
Revised placement of changed markers...0.050 sec.
Assignment of genotypes to bins...2.764 sec.
Final placement of new markers...4.136 sec.
Marker placement report:
Marker        Link   Mean Pos   Pos Std.   Left Flanking Markers and p-vals
SIM-NEW-1        1       0.13      0.016   SIM-FRAME-1    0.550   SIM-FRAME-2    0.450
SIM-NEW-2        1       0.15      0.013   SIM-FRAME-1    0.524   SIM-FRAME-2    0.476
SIM-NEW-3        1       0.25      0.020   SIM-FRAME-2    1.000
SIM-NEW-4        1       0.30      0.014   SIM-FRAME-3    0.550   SIM-FRAME-2    0.450
SIM-NEW-5        1       0.32      0.017   SIM-FRAME-3    0.604   SIM-FRAME-2
```

etc.

```
Detailed bin view:
SIM-FRAME-1
   Bin #1 (midpt: 0.007):
   Bin #2 (midpt: 0.022):
   Bin #3 (midpt: 0.037):
   Bin #4 (midpt: 0.053):
   Bin #5 (midpt: 0.067):
   Bin #6 (midpt: 0.082):
   Bin #7 (midpt: 0.098):
   Bin #8 (midpt: 0.113):
   Bin #9 (midpt: 0.128):   SIM-NEW-1   (0.999)
SIM-FRAME-2
   Bin #1 (midpt: 0.149):   SIM-NEW-2   (1.000)
   Bin #2 (midpt: 0.170):
   Bin #3 (midpt: 0.184):
   Bin #4 (midpt: 0.198):
   Bin #5 (midpt: 0.211):
   Bin #6 (midpt: 0.225):
   Bin #7 (midpt: 0.239):
   Bin #8 (midpt: 0.252):   SIM-NEW-3   (1.000)
   Bin #9 (midpt: 0.266):
   Bin #10 (midpt: 0.280):
```

etc.

```
Possible errors or fixes to omissions in framework genotypes:
None.
Possible errors or fixes to omissions in new genotypes:
None.
RMS error: 0.021 map units.
Number of markers more than twice as far: 6
Badly placed markers:
SIM-NEW-16
SIM-NEW-18
SIM-NEW-19
SIM-NEW-65
SIM-NEW-73
SIM-NEW-96
Number of markers badly placed relative to estimated error: 8
Badly placed markers:
SIM-NEW-16
SIM-NEW-18
SIM-NEW-19
SIM-NEW-64
SIM-NEW-65
SIM-NEW-73
SIM-NEW-89
SIM-NEW-96
Command completed successfully.
```

The output is extensive but essentially simple. The first few lines report on the time taken to perform each of the steps in the bin mapping procedure. Following this is the marker placement report. The columns from left to right are

1. Marker - one entry for each marker in the new marker dataset.

2. Link - the linkage group assignment, always 1 for simulated data.

3. Mean Pos - the expected position of the marker in cM from the zero-point on the linkage group. This is a weighted average of the midpoints of bins in which the marker has a placement probability of at least 0.01.

4. Pos Std. - the standard deviation of the expected position, again weighted by each bin in which the marker has a placment probability of at leats 0.01.

5. Left Flanking Markers and p-vals - For each framework interval in which the marker has been assigned with placement probabiility at least 0.01, the framework marker with the lower position (closer to the 0 cM terminus) is listed along with the corresponding placement probability.

Following this is a "detailed bin view". For each framework interval, the bins are enumerated along with their expected midpoints, the markers that place in those bins are listed along with the probability that they belong in that bin.

Following this is a list of possible errors or fixes to omissions (filled in missing data) in the framework and new genotype matrices. These two lists are empty in simulations, but we will see what they look like below when analyzing real data.

Following this is output that is specific to simulation mode. "RMS" is the root mean squared error of marker placement. The error for each marker is the exact placement subtracted from the estimated placement (or "Mean Pos."). (The exact placement of all markers is saved by an internal variable generated when the SIMNEW command is invoked). Markers placed more than twice as far as the RMS error are listed, as are markers for which the actual error is more than twice as great as the estimated error ("Pos. Std.")

To return to the shell, type

```
MapPop [4]: quit
```

7.2 Selective Mapping Mode

To perform selective mapping with an actual dataset, we use the LOADFRAME command. This instructs MapPop to load a frmaework map file. Here, we will be using "barleyframe.txt", data for a set of markers from chromosome 7 on the barley IGRI x Franka doubled haploid population (Graner *et al.* 1994). The genotype codes in this dataset are
1 IGRI homozygote
2 Franka homozygote
- unknown
Since these do not correspond to the defaults (see SET [TypeString]), we specify this code on the command line.

```
MapPop [1]: loadframe barleyframe.txt 12-xxx
Markers typed: 15
Lines typed: 70
Generating test instantiations
Command completed successfully.
```

MapPop estimates the precise positions of visible breakpoints in the framework map by generating many different "instantiations", or randomly resolving breakpoints to exact sites located between markers that have incompatible genotypes. The number of instantiations can be controlled with the SET command.

Another variable that can be controlled is the time limit of the greedy search (used for both the SAMPLEMAX and SAMPLESUM procedures). We will set this to 100 seconds, so that it is shorter than the default (300 seconds) but we need not specify it on the command line when we invoke either of those two commands.

```
MapPop [2]: set time 100
Command completed successfully.
```

Now we select a sample that minimizes the maximum bin length by invoking the SAMPLEMAX command. We also include a value for the [size] parameter, which specifies the size of the sample to be chosen. By leaving the other two parameters blank, we have implicitly instructed MapPop to use the defaults, in this case 100 for [time] and [5] for choices. Most parameters in MapPop are optional. Neglecting one of the few mandatory parameters will result in an error message.

```
MapPop [3]: samplemax 40
Choosing sample to minimize expected max bin length.
Size desired: 40
Time allowed: 100 sec
Number of iterations: 588
Expected max bin length : 10.0508 map units
Sample Lines:
2
3
4
8
9
10
```

etc.

Return to the command shell by typing

```
MapPop [4]: quit
```

We could have automated this entire process, and sent the output to a file to viewed and edited at leisure, by using redirect commands from the shell. To do this, we would first make a text file (call it “commands.txt”) with the commands that were fed to MapPop just now:

```
loadframe barleyframe.txt 12-xxx
set time 100
samplemax 40
quit
```

Then, we would invoke MapPop, tell it to substitute the lines of “commands.txt” for the standard input (rather than the keyboard) and to substitute “output.txt” for the standard output (rather than the terminal screen). This would result in a “output.txt” file that looked something like this:

```
MapPop version 0.9alpha.
Copyright 1999, Daniel G. Brown and Todd J. Vision
This software comes with NO WARRANTY.
Type ’help license’ for more details.
Type ’help’ for general help.

Markers typed: 15
```

```
Lines typed: 70
Generating test instantiations
Command completed successfully.
Command completed successfully.
Choosing sample to minimize expected max bin length.
Size desired: 40
Time allowed: 100 sec
Number of iterations: 588
Expected max bin length : 9.87282 map units
Sample Lines:
2
4
8
9
10
```

etc.

7.3 Bin Mapping Mode

Bin mapping requires that both a framework marker file and a new marker file be loaded. A sample new marker file for the barley IGRI x Franka chromosome 7 dataset is provided: "barleynew.txt". The command for loading the new marker file is LOADNEW, and the syntax of the command is identical to that of LOADFRAME. So first

```
MapPop [1]: loadframe barleyframe.txt 12-xxx
Markers typed: 15
Lines typed: 70
Generating test instantiations
Command completed successfully.
```

and then

```
MapPop [2]: loadnew barleynew.txt 12-xxx
Markers types: 28
Lines typed: 70
Command completed successfully.
```

Bin mapping is done by invoking the ADDNEW command. The only relevant variable that can be set to affect the outcome of bin mapping is [ErrorRate], which allows MapPop to detect and account for individual genotype errors. The default is 0.01, or 1%.

MapPop will assume that all individuals in the new marker file are part of the selected sample on which you wish to map. In this way, you can use MapPop to perform bin mapping using either selected or unselected samples. In fact, the new marker data for barley chromosome 7 includes all the same individuals that are in the framework map file.

We use the optional argument for the ADDNEW command to specify an outfile called “map.out”. This most of the screen output to the file specified, with the exception of the timing profiles for each step in the algorithm.

```
MapPop[3]: addnew map.out
Preliminary placement of markers into intervals...0.151 sec.
Correction of uncertain framework genotypes...0.040 sec.
Revised placement of new markers...0.070 sec.
Check for spurious double crossovers...0.370 sec.
Revised placement of changed markers...0.010 sec.
Assignment of genotypes to bins...0.561 sec.
Final placement of new markers...0.671 sec.
Marker placement report:
Marker    Link   Mean Pos   Pos Std.   Left Flanking Markers and p-vals
Ris17a       1       1.93      1.774   ABG312     1.000
MWG851a      1       4.51      1.667   ABG312     1.000
MWG555a      1      10.48      1.349   ABG312     0.994
MWG2074      1      17.69      1.897   Ris45a     1.000
MWG807       1      21.62      1.354   Ris45a     0.531   MWG530     0.469
PBI35        1      25.02      1.605   MWG530     1.000
MWG2080      1      27.60      1.800   MWG530     1.000
ABC151a      1      40.45      1.122   MWG564     0.521   cMWG703    0.479
cMWG773      1      47.39      1.873   MWG564     1.000
```

etc.

```
Detailed bin view:
ABG312
   Bin #1 (midpt: 0.644):   Ris17a (0.500)
   Bin #2 (midpt: 1.931):
   Bin #3 (midpt: 3.219):   Ris17a (0.500)
   Bin #4 (midpt: 4.506):   MWG851a (1.000)
   Bin #5 (midpt: 5.794):
   Bin #6 (midpt: 7.081):
   Bin #7 (midpt: 8.369):
Ris45a
   Bin #1 (midpt: 10.478):   MWG555a (1.000)
   Bin #2 (midpt: 12.764):
   Bin #3 (midpt: 14.407):
   Bin #4 (midpt: 16.050):
   Bin #5 (midpt: 17.693):   MWG2074 (1.000)
   Bin #6 (midpt: 19.336):
MWG530
   Bin #1 (midpt: 21.623):   MWG807 (1.000)
   Bin #2 (midpt: 23.733):
```

```
   Bin #3 (midpt: 25.022):   PBI35 (1.000)
```

etc.

```
   Possible errors or fixes to omissions in framework genotypes:
14    , MWG530  : 2
50    , MWG836  : 2
22    , Brz     : 1
8     , MWG2301 : 2
21    , MWG626  : 1
5     , cMWG696 : 2
43    , MWG2262a: 1
Possible errors or fixes to omissions in new genotypes:
None.
Command completed successfully.
```

Note that in the last few lines of the output, MapPop has flagged several individual genotypes. These are values that were either unknowns or appear to be in error. For each one, MapPop lists the name of the individual, the name of the marker, and the likely true genotype. Here there were 6 in the framework genotype dataset but not among the new markers.

Again, to return to the shell, type

```
MapPop [4]: quit
```

8 Bugs and wishes

8.1 Known bugs

- Command prompts appear at irregular places in redirected output. This appears to be inherent to the Matlab interface library routines.
- MapPop may crash if parameters are chosen that cause illogical or out-of-bounds operations (*e.g.* Specifying a simulated linkage group length in centiMorgans rather than Morgans).

Please send your bug reports to us, via the contact information listed above. We will fix them in future versions of the software.

8.2 Wish List

Many features will be added to upcoming versions of MapPop:

- A feature that helps suggest when further time spent in finding samples is not needed.
- A graphical user interface.
- A graphical representation of the mapping results.
- And many others.

Please contact us with your wishes, as well, and we will consider them for future versions of the software.

9 Credits and Acknowledgments

MapPop was written by Daniel G. Brown and Todd J. Vision, in Matlab and C, in Spring to Fall, 1999.

MapPop was written largely in Matlab, which is a product of the Mathworks, Inc. While Matlab is at times extremely frustrating as a development environment, and its proprietary nature makes it very difficult to develop public domain software in it, we would still like to thank the Mathworks for developing a very useful prototyping and programming tool.

MapPop's development was supported by NSF grant DBI-98-72617 (PI: Steve Tanksley), NSF grant CCR-970029 (PI: David Shmoys) and ONR grant N0014-96-1-00500 (PI: Michael J. Todd) and by support from the USDA-ARS Center for Bioinformatics and Comparative Genomics to Todd Vision.

10 Literature Cited

Brown, D.G., T.J. Vision, S.D. Tanksley, (2000) Selective mapping: a discrete optimization approach to selecting a population sample for use in a high-density linkage mapping project. Proceedings of the 11th SIAM-ACM Symposium on Discrete Algorithms.

Brown, D.G. and T.J. Vision (1999) A computationally novel way to place new markers onto genetic maps. Cornell University Tech Report. TR-CCOP-99-9

Graner, A. E. Bauer, A. Kellerman, S. Kirchner, J. K. Muraya *et al.* (1994) Progress of RFLP map-construction in winter barley. Barley Genetics Newsletter 23:53-59.

Vision, T.J., D.G. Brown, D.B. Shmoys, R.T. Durrett, S.D. Tanksley, (1999) Selective mapping: A strategy for optimizing the construction of high-density linkage maps. Manuscript.