

User Manual

FaST-LMM

Factored Spectrally Transformed Linear Mixed Models

Version 2.03

Microsoft Research

Nov 22, 2012

Introduction

FaST-LMM, which stands for *Factored Spectrally Transformed Linear Mixed Models* is a program for performing genome-wide association studies (GWAS) on large data sets. It runs on both Windows and Linux systems, and has been tested on data sets with over 120,000 individuals.

This software is released under the Microsoft Research License Agreement, ("MSR-LA" or the "License"); you may not use the software except in compliance with the License. You can find a copy of the License in the file LICENSE.TXT accompanying this file. Links to updated versions can be found at: <http://mscompbio.codeplex.com>.

Versions of this software may also rely on additional libraries and code distributed under their respective licenses. To re-compile or build this code or derivatives thereof, you may be required to individually download and appropriately license some or all of these additional libraries for your specific use. See the file NOTICE.TXT, accompanying this file for more details.

For help with the software, please contact

Christoph Lippert, christoph.a.lippert@gmail.com
Jennifer Listgarten, jennl@microsoft.com
Carl Kadie, carlk@microsoft.com
Bob Davidson, bobd@microsoft.com
David Heckerman, heckerma@microsoft.com

Citing FaST-LMM

If you use FaST-LMM in any published work, please cite both the software (using the link <http://mscompbio.codeplex.com/>) and the manuscript describing it:

C. Lippert, J. Listgarten, Y. Liu, C.M. Kadie, R.I. Davidson, and D. Heckerman.
FaST Linear Mixed Models for Genome-Wide Association Studies. *Nature Methods*
8: 833-835, Oct 2011 (doi:10.1038/nmeth.1681).

Also, we would appreciate it if you let us know that you are citing it.

Installing FaST-LMM

FaST-LMM is available as a .zip file that extracts to these directories:

<code>fastlmm/Bin</code>	contains the compiled executable files
<code>fastlmm/Cpp</code>	contains C++ source and project files
<code>fastlmm/Data/sampledata</code>	contains sample data and command script
<code>fastlmm/Doc</code>	contains project documentation
<code>fastlmm/Externals</code>	contains other code FaST-LMM depends on

There are executables for Windows (64bit), and for Ubuntu Linux (64bit) under the `fastlmm\Bin` directory and all required .dll files are included in the respective directories. These executables use the MKL math library, which is optimized for Intel

processors but also runs on AMD processors. If one of these options is suitable, please skip ahead to section “Data Preparation” to see how to run FaST-LMM on your data. If not, please see the next section.

Compiling FaST-LMM¹

In addition to the source code, the following external dependencies must be installed and met in order to build FaST-LMM:

Building for Windows

C++ of versions FaST-LMM are built with Visual Studio 2010 (VS)
Any version of VS (Express through Universal) is capable of building FaST-LMM.
If you do not already have a copy of Visual Studio, the Visual Studio 2010 Express edition can be freely downloaded from

<http://www.microsoft.com/express/downloads>

- For the C++ version
FaST-LMMC uses a 3rd party math library for advanced math functions and performance. `fastlmmc` can use either Intel's MKL or AMD's ACML math libraries. Once you have installed the appropriate library, use the Visual Studio IDE to select the appropriate configuration from the solution and build. ACML requires an additional step to tell Visual Studio where it is located. You must set the environment variable `ACML_ROOT` to point to your install location or libraries will not be located—for example,

```
C>set ACML_ROOT=C:\AMD\acml4.4.0
```

You can find more about the math libraries at their respective web sites:

<http://software.intel.com/en-us/articles/intel-mkl>

<http://developer.amd.com/libraries/acml/pages/default.aspx>

With a math library installed, no additional libraries are required to compile the C++ version of FaST-LMM (`fastlmmc`). Double-click the `fastlmmc.sln` file to load Visual Studio and then build the solution associated with your library.

Building the C++ version of FaST-LMM for Linux

FaST-LMM is primarily developed and tested on Windows although we are able to build the C++ version for Linux. We provide a simple script file that uses the GNU toolset with the 3rd party math library to compile the sources in a Linux environment.

- `fastlmmc` uses a 3rd party math library for advanced math functions and performance. The program has been run on Ubuntu Linux and can use either Intel's MKL or AMD's ACML math libraries for Linux. Once you have selected

¹ When building the C++ version of FaST-LMM the build host machine name is captured and used in the program banner to help identify which version of the code is being run. If you build `fastlmmc.exe` and you do not want the machine name captured, you should modify the banner string in `Splash.cpp` to remove it.

and installed the appropriate library, you can then build using the appropriate script file located in the `Cpp` directory. Review of the two files, `DoMKL_linux` and `DoAcml_linux`, will show very simple scripts to compile the program using `g++` and then link the `.o` files with the appropriate math library. The `*.o` files are written to version specific directories, so it is necessary to create the appropriate directory prior to running the script. For more details, see the script.

You can find more about the math libraries for Linux at their respective web sites:
<http://software.intel.com/en-us/articles/intel-mkl>
<http://developer.amd.com/libraries/acml/pages/default.aspx>

Data preparation

FaST-LMM uses four input files containing (1) the SNP data to be tested, (2) the SNP data used to determine the genetic similarities between individuals (which can be different from 1), (3) the phenotype data, and (4, optionally) a set of covariates.

When the realized relationship matrix (RRM) is used for genetic similarity, and when the number of SNPs used to construct the RRM is less than the number of individuals, the runtime and memory footprint of FaST-LMM scales linearly in the number of individuals in the data. When this condition is not met, the runtime and memory footprint of FaST-LMM are cubic and quadratic in the number of individuals, respectively.

All input files should be in ASCII.

Both SNP files (1 and 2 above) should be in PLINK format (`ped/map`, `tped/tfam`, `bed/bim/fam`, or `fam/dat/map`). For the most speed, use the binary format in SNP major order. The phenotype entries in these files must be set to some dummy value and will be ignored (our software uses a separate phenotype file). Sex should be encoded as a single digit. See the PLINK manual <http://pngu.mgh.harvard.edu/~purcell/plink/>¹ for further details. Missing SNP values will be mean imputed. Dosages files are also allowed (see the end of this section).

The required file containing the phenotype (3 above) uses the PLINK alternate phenotype format. It should have at least three columns: `<familyID>`, `<individualID>`, and any number of `<phenotype value>`. The columns are delimited by whitespace (`<tab>` or `<space>`). The default option is to test the first phenotype only. A missing value should be denoted by `-9`, but this can be changed (see options below). The first column, `<familyID>`, is joined with the second column `<individualID>` to create a unique key for the individual that matches an entry for an individual in the PLINK files above.

Example phenotype file for two phenotypes (FaST-LMM/data/sampleddata/pheno.txt) :

1	IND0	2	3.05043
1	IND1	2	1.72797
1	IND2	-9	4.19592
1	IND3	2	3.4492
1	IND4	1	-8.99843

```

1      IND5  1      -0.768613
1      IND6  2       6.73734
...

```

Optionally, the phenotype file may also have a header row, for example, as follows:

```

FID    IID    MyPheno    YourPheno

```

The optional file containing covariates should have at least three columns: `<familyID>`, `<individualID>`, and any number of `<covariate value>`. The columns should be tab delimited. The token for missing values must be the same as that used in the phenotype file. All covariates are processed. Covariate files should not have a header row.

Example covariate file (FaST-LMM/data/sampleddata/covariate.txt) :

```

1      IND0  1
1      IND1  1
1      IND2  1
1      IND3  1
1      IND4  1
1      IND5 -9
1      IND6  1
...

```

Instead of SNP data from which genetic similarities are computed, the user may provide the genetic similarities directly using the `-sim <filename>` option. The file containing the genetic similarities should be tab delimited and have both row and column labels for the IDs (family ID and individual ID separated by a space). The value in the top-left corner of the file should be `var`.

Example similarities file:

```

var    1 IND0      1 IND1      1 IND2      1 IND3      ...
IND0   1.0    0.5    0.5    0.25    ...
IND1   0.5    1.0    0.5    0.5     ...
IND2   0.25   0.5    1.0    0.5     ...
...

```

FaST-LMM supports dosage files in PLINK formats 1 and 2. The files may be uncompressed or compressed (with a .gz extension). Use `-dfile1` or `-dfile2` followed by the file prefix to load test SNPs in format 1 or 2, respectively. Use `-dfile1Sim` or `-dfile2Sim` followed by the file prefix to load similarity SNPs in format 1 or 2, respectively. As described in the PLINK manual, the .map file is optional and will be loaded if present.

Running FaST-LMM

Once you have prepared the files in the proper format, you can run FaST-LMM. Here is a sample call on the synthetic data provided in the .zip file from the directory containing that data and assuming `fastlmmc` is in the path:

```
> fastlmmc -tfile geno_test -tfilesim geno_cov -pheno pheno.txt -covar
covariate.txt -mpheno 1
```

You should see something like the following output on the screen (using the C++ version):

```
FastLmmC v1.08.20120313 - Factored Spectrally Transformed Linear Mixed Models
Copyright Microsoft Corporation
Compiled Mar 13 2012 at 06:20:44 by BOBD01 for Windows
using MKL v10.3.5 - Build: 20110720

++ Start Processing CommandLine:
-- End Processing CommandLine:
++ Start Loading FastLmm Data:
++ Start Loading Covariance Data:
++ Processing PLINK files: [geno_cov]
  Number of Individuals Selected: 270
  Number of Phenotypes/Individual: 1
  Number of SNPs/Individual: 200
  Number of SNPs/Individual Used: 200
-- End Processing PLINK files: [geno_cov]
-- End Loading Covariance Data:
++ Start Loading Test Data:
++ Processing PLINK files: [geno_test]
  Number of Individuals Selected: 270
  Number of Phenotypes/Individual: 1
  Number of SNPs/Individual: 200
  Number of SNPs/Individual Used: 200
-- End Processing PLINK files: [geno_test]
-- End Loading Test Data:
-- End Loading FastLmm Data:
  Compute/Load EigenSym:
  Compute Woodbury low rank: ... done.
  Compute GWAs using LMM:
-- Start lowrank training:
-- Lowrank training done:
  Write output: [geno_test.out.txt]
  Total elapsed time: 311.918 ms
```

When the output file `[geno_test.out.txt]` is loaded in Excel, it should look as follows:

SNP	Chromosome	GeneticDistance	Position	Pvalue	Qvalue	N	NullLogLik	AltLogLik	SNPWeight	SNPWeightSE	WaldStat	NullLogDev	NullGeneDev	NullResid	NullBias
snp2	1	0	3	5.42E-08	1.08E-05	200	-1.83E+02	-1.68E+02	-1.76E-01	1.92E-03	0	1.69E+00	3.69E-02	2.01E-01	1.53E+00
snp110	1	0	111	6.29E-03	6.29E-01	200	-1.83E+02	-1.79E+02	1.27E-01	2.82E-03	0	1.69E+00	3.69E-02	2.01E-01	1.53E+00
snp55	1	0	56	1.60E-02	8.15E-01	200	-1.83E+02	-1.80E+02	8.12E-02	2.04E-03	0	1.69E+00	3.69E-02	2.01E-01	1.53E+00
snp167	1	0	168	1.74E-02	8.15E-01	200	-1.83E+02	-1.80E+02	8.54E-02	2.17E-03	0	1.69E+00	3.69E-02	2.01E-01	1.53E+00
snp140	1	0	141	2.04E-02	8.15E-01	200	-1.83E+02	-1.80E+02	-8.12E-02	2.12E-03	0	1.69E+00	3.69E-02	2.01E-01	1.53E+00
snp171	1	0	172	3.14E-02	8.26E-01	200	-1.83E+02	-1.80E+02	9.36E-02	2.64E-03	0	1.69E+00	3.69E-02	2.01E-01	1.53E+00
snp144	1	0	145	3.27E-02	8.26E-01	200	-1.83E+02	-1.80E+02	7.51E-02	2.13E-03	0	1.69E+00	3.69E-02	2.01E-01	1.53E+00

...

The standard and `-verboseOut` columns are:

`SNP`

The rs# or SNP identifier for the SNP tested. Taken from the PLINK file.

`Chromosome`

The chromosome identifier for the SNP tested or 0 if unplaced. Taken from the PLINK file.

`Genetic Distance`

The location of the SNP on the chromosome. Taken from the PLINK file. Any units are allowed, but typically centimorgans or morgans are used.

`Position`

The base-pair position of the SNP on the chromosome (bp units). Taken from the PLINK file.

`Phenotype [under -verboseOut]`

The name of the phenotype as specified in the header of the phenotype file.

`NoName` means that no header row was specified.

`Pvalue`

The p-value computed for the SNP tested

`Qvalue`

The q -value computed for the SNP tested estimated from the p -values of all test-SNPs in the PLINK file using the procedure of Benjamini and Hochberg

`N`

The sample size or number of individuals that have been used for this analysis

`NumSNPsExcluded [under -excludeByGeneticDistance]`

`IndexExclusionStart [under -excludeByGeneticDistance]`

`DOF [under -verboseOut]`

The degrees of freedom of the statistical test

`NullLogLik`

The log likelihood of the null model

`AltLogLik`

The log likelihood of the alternative model

`SnpWeight`

The fixed-effect weight of the SNP

`SnpWeightSE`

The standard error of the SnpWeight

`NullLogDelta`

The ratio between the residual variance and the genetic variance $\delta = \sigma_e^2 / \sigma_g^2$ on the null model

`NullGeneticVar`

The genetic variance σ_g^2 on the null model

`NullResidualVar`

The residual variance σ_e^2 on the alternative model

`NullBias`

The offset term in the null model

`LogDelta [under -verboseOut]`

The ratio between the residual variance and the genetic variance $\delta = \sigma_e^2 / \sigma_g^2$ on the alternative model

`geneticVar [under -verboseOut]`

The genetic variance σ_g^2 on the alternative model

`ResidualVar [under -verboseOut]`

The residual variance σ_e^2 on the alternative model

`NullBias [under -verboseOut]`

The offset term in the alternative model

`SNPIndex`

The column index of the SNP tested in the PLINK file starting at 1

`SNPCount`

The number of SNPs tested

Speed vs. accuracy considerations

The FaST-LMM inference involves a search over the ratio δ of genetic and environmental variances. As this step represents a non-convex optimization FaST-LMM performs an optimization procedure over several intervals on a logarithmic scale, invoking iterative calls to the likelihood function. The total run-time of this step scales linear in the sample size times a constant that approximately equals the number of intervals considered for the search.

The command line option `-simLearnType Full` is set by default to perform “exact” LMM inference that avoids this potential loss of power by refitting the ratio δ of variances for every SNP tested.

Use the command line option `-simLearnType Once` to gain a constant factor speedup. Using this option, the ratio δ is found on the null-model only and is fixed to that value throughout the testing procedure. Note, though, that on some data sets this could lead to slight loss of power when SNPs with a large effect are tested.

Additionally, the number and coarseness of the search intervals can be adjusted via the command line options `-brentStarts <int>` for the number of intervals,

`-brentMinLogVal <double>` for the minimum of the search scope of $\log\delta$ values,
and
`-brentMaxLogVal <double>`, for the maximum of the search scope of $\log\delta$ values.

By default the search is set conservatively to span 100 intervals over δ values between $\ln(-5)$ and $\ln(5)$.

FaST-LMM-Select: Avoiding proximal contamination and dilution

Proximal contamination and dilution are two phenomena that degrade the performance of LMMs for GWAS². In this section, we describe these two phenomena and how the FaST-LMM software can be used to avoid them.

To understand proximal contamination, first note that a LMM with no fixed effects, using a realized relationship matrix (RRM) for genetic similarities, is mathematically equivalent to linear regression of the SNPs on the phenotype, with weights integrated over independent Normal distributions having the same variance³. That is, a LMM using a given set of SNPs for genetic similarity is equivalent to (Bayesian) linear regression using those SNPs as covariates to correct for confounding. This equivalence implies that, when testing a given SNP, that SNP (and SNPs physically close to it) should be excluded from the computation of genetic similarity. If not, when testing a particular SNP, we would also be using that same SNP as a covariate, making the log likelihood of the null model higher than it should be, thus leading to deflation of the test statistic and loss of power. We call this phenomenon *proximal contamination*.

To combat proximal contamination, we need to exclude the SNP we are testing from the genetic similarity matrix and also those SNPs in close proximity to it. Doing so in a naïve way, however, is extremely computationally expensive. FaST-LMM, however, can do this exclusion efficiently². To use this feature, either genetic distances or positions must be included in the PLINK files specifying both the test SNPs and the SNPs used for genetic similarity. Furthermore, the SNPs must be in non-decreasing order according to chromosome and the distance used. To perform the exclusion, use either the option `-excludeByGeneticDistance` or `-excludeByPosition` followed by a parameter describing the distance (centimorgans or nucleotide position). In practice, we have found the values 2 centimorgans or 2 million positions to work well. The units of the parameters for these flags are the same as the units provided in the SNP files (which is user-determined). That is, if the SNP genetic distances are specified in Morgans, for example, then the option `-excludeByGeneticDistance 0.02` will exclude SNPs within 0.02 Morgans.

The equivalence between LMMs and linear regression also provides an understanding of dilution. Regardless of what form of regression we use for GWAS, the measure of SNP-phenotype association (*e.g.*, a P value) should be determined by conditioning on exactly those SNPs that are associated with the phenotype. These SNPs include causal SNPs or SNPs that tag causal SNPs, and SNPs that are associated by way of confounding (*e.g.*, population structure). By conditioning on causal or tagging SNPs, we reduce the noise in the assessment of the association⁴. By conditioning on SNPs associated by virtue of

confounding, we control for such confounding⁵. Moreover, if a SNP is unrelated to the phenotype, it should not be in the conditioning set. In the particular case where we are using Bayesian linear regression for GWAS, the inclusion in the genetic similarity matrix of SNPs that are unrelated to the phenotype adds variance to the regression on the phenotype, which in turn leads to inflated test statistics and reduced power. We refer to this phenomenon as *dilution*.

To identify those SNPs which should be used for genetic similarity as just specified, we recommend the following approach involving cross validation². This approach is now automated (see next paragraph). First, partition the data across individuals into k blocks (e.g., $k=10$). This partitioning defines k folds for cross validation, where the i^{th} fold has test data equal to the i^{th} block and training data equal to the remaining data. For each fold, use the training data to order SNPs by their linear-regression P values in increasing order. Then, again on the training data, construct genetic similarity matrices with an increasing number of SNPs according to this ordering, and use the resulting model to predict the test data (note, such predictions are done via a Gaussian process corresponding to the mixed model). Next, identify the number m of SNPs that minimizes the mean-squared error (or maximizes the out-of-sample log likelihood) summed over all folds. Finally, use the full data to order SNPs by their linear-regression P values in increasing order, and select the first m SNPs.

This procedure is called by using the `-autoSelect` option. The call outputs the select SNPs to a file, which can then be used in a subsequent call to FaST-LMM to perform GWAS using the `-extractSim` option. Here is an example of calls to perform selection followed by GWAS:

```
> fastlmmc -autoSelect ASout -autoSelectSearchValues ASvalues.txt -randomSeed 1
-autoSelectFolds 10 -bfilesim geno_cov -pheno pheno.txt -covar covariate.txt -
mpheno 1

> fastlmmc -bfile geno_test -bfilesim geno_cov -extractSim ASout.snps.txt
-pheno pheno.txt -covar covariate.txt -mpheno 1
```

In the first call, `ASout` is the file prefix for the output of AutoSelect, consisting of a list of snps (`ASout.snps.txt`) and the statistics of the search (`ASout.xval.txt`). The option `-autoSelectSearchValues` is optional. The filename `ASvalues.txt` contains a white-space delimited list of number-of-SNPs to try in the search. The default search values are 1, 2, ..., 10, 20, ..., 100, 125, 160, 200, 250, 320, 400, 500, 630, 800, and 1000. Note that, if the largest value is selected, the search should be extended to larger values including the total number of SNPs.

In the second call, `-extractSim` is used to select the SNPs listed in `ASout.snps` from the full set of SNPs in `geno_cov`.

If the full data does not fit into memory or AutoSelect runs too slowly to be useful in practice, you can include a pre-processing step in AutoSelect that narrows the set of SNPs over which to search. This pre-processing uses the full data to order SNPs by their linear-regression P values in increasing order. Note that this procedure can lead to

overfitting, because the full data set is used to eliminate poorly predicting features. Consequently, this pre-processing step should only be used when necessary. To direct AutoSelect to use this pre-processing, use the `-topKbyLinReg` option as follows:

```
> fastlmmc -autoSelect ASout -autoSelectSearchValues ASvalues.txt -randomSeed 1
-autoSelectFolds 10 -bfilesim geno_cov -pheno pheno.txt -covar covariate.txt
-mpheo 1 -topKbyLinReg 10000 -memoryFraction 0.2
```

This call narrows the search to the top 10,000 SNPs, limiting memory use during the pre-processing to 20 percent of the available memory.

Command line options

`-file` basefilename
 basename for PLINK's .map and .ped files

`-bfile` basefilename
 basename for PLINK's binary .bed, .fam, and .bin files

`-tfile` basefilename
 basename for PLINK's transposed .tfam and .tped files

`-dfile1` basefilename
 basename for PLINK's .dat, .fam, and (optionally) .map files, format=1

`-dfile2` basefilename
 basename for PLINK's .dat, .fam, and (optionally) .map files, format=2

`-noDosageRangeCheck`
 disables range checking for dosage files. Default: false.

`-pheno` filename
 name of phenotype file

`-mpheo` index
 index for phenotype in -pheno file to process, starting at 1 for the first phenotype column. Cannot be used together with -pheno-name. Default: 1.

`-pheno-name` name
 phenotype name for phenotype in -pheno file to process. If this option is used, the phenotype name must be specified in the header row. Cannot be used together with -mpheo.

`-fileSim` basefilename
 basename for PLINK's .map and .ped files for computing genetic similarity

`-bfileSim` basefilename
 basename for PLINK's binary .bed, .fam, and .bin files for building genetic similarity

`-tfileSim` basefilename
 basename for PLINK's transposed .tfam and .tped files for building genetic similarity

-dfile1Sim basefilename
 basefilename for PLINK's .dat, .fam, and (optionally) .map files for building genetic similarity, format=1

-dfile2Sim basefilename
 basefilename for PLINK's .dat, .fam, and (optionally) .map files for building genetic similarity, format=2

-sim filename
 specifies that genetic similarities are to be read directly from this file

-simOut filename
 specifies that genetic similarities are to be written to this file

-linreg
 specifies that linear regression will be performed. When this option is used, no genetic similarities should be specified.

-covar filename
 optional file containing the covariates

-missingPhenotype <dbl>
 identifier for missing values. If the phenotype for an individual is missing, then the individual is ignored. If a covariate value for an individual is missing, then it is mean imputed. Default: -9.

-out filename
 the name of the output file. Default value is [basefilename].out.txt. If the extension .csv is used, then the output is comma separated. Otherwise, the output is tab separated.

-simLearnType [Full/Once]
 if set to Once, then delta, the ratio of residual to genetic covariance, is optimized only for the null model and used for each alternate model. If set to Full (the default), then the ratio is re-estimated for each alternative model.

-simType [RRM/COVARIANCE]
 if set to RRM (the default), then the RRM is used for genetic similarity. If set to COVARIANCE, then the empirical SNP covariance matrix is used.

-ML
 use maximum likelihood parameter learning (default is REML)

-REML
 use restricted maximum likelihood parameter learning. REML will automatically invoke the F-test.

-Ftest
 use F-test (with ML or REML).

-brentStarts <int>
 number of interval boundary points for optimization of delta (see Section 2.1 of the Supplemental Information). Default: 100.

`-brentMaxIter <int>`
maximum number of iterations per interval for the optimization of delta.
Default: 1e5.

`-brentMinLogVal <double>`
lower interval threshold for (log) delta optimization. Default: -5.

`-brentMaxLogVal <double>`
upper interval threshold for (log) delta optimization. Default: 5.

`-brentTol <double>`
convergence tolerance of Brent's method used to optimize delta. Default: 1e-6.

`-runGwasType [RUN/NORUN]`
run GWAS or exit after computing the spectral decomposition of the genetic similarity matrix. Use `NORUN`, to cache the spectral decomposition. This option, in combination with the next, is useful for parallelizing the tests of many SNPs.
Default: `RUN`.

`-eigen [directoryname]`
load the spectral decomposition object from the directory name. The computations leading to the spectral decomposition of the genetic similarity matrix are skipped (note that that SNP file specifying the genetic similarities must still be given).

`-eigenOut [directoryname]`
save the spectral decomposition object to the directory name. Can be used with `-runGwasType` option.

`-numJobs <int>`
partition the SNPS into `<int>` groups and run FaST-LMM on the partition specified by `-thisjob`.

`-thisJob <int>`
specifies which partition of SNPS created by `-numjobs` to process for this run of FaST-LMM.

`-extract filename`
this is a SNP filter option. FaST-LMM will only analyze the SNPs explicitly listed in the 'filename' (no header, one SNP per line, where the SNP is indicated by the rs# or snp identifier).

`-extractSim filename`
this is a genetic similarity SNP filter option. FaST-LMM will only use SNPs explicitly listed in the 'filename' for computing genetic similarity.

`-extractSimTopK filename <int>`
similar to `-extractSim`, this is a genetic similarity SNP filter option. FaST-LMM will only use the first `<int>` SNPs explicitly listed in the 'filename' for computing genetic similarity.

`-verboseOut`
enable a more detailed and verbose output file with more columns. (See output)

`-setOutputPrecision <int>`

FastLmmC uses doubles for computation and has a default output precision of 16 digits after the decimal point. When working with large numbers of SNPs, writing full 16 digit precision can produce output that is quite large and this output precision may not be necessary. You can reduce the digits written in the output and reduce the file size using the `-SetOutputPrecision <int>` option. The parameter is restricted to the range $3 \leq \text{<int>} \leq 18$.

`-pValuePrintThreshold <dbl>`

this option sets a threshold filter to restrict the report output to include only those SNPs that have a p-value less than the specified value `<dbl>`. When large datasets are used, PvaluePrintThreshold produces smaller and more manageable output files.

The parameter is restricted to the range $0.0 < \text{<dbl>} \leq 1.0$.

`-maxThreads <int>`

the option is passed to the MKL math libraries to ‘suggest’ the level of parallelism to use. Assigning a number larger than the number of cores on your machine may cause the program to run slower. Assigning a number less than the number of cores on your machine may allow your computer to run FastLmmC without consuming all the CPU resources in different phases of the program.

The MaxThreads option is currently ignored when using ACML math libraries.

`-excludeByGeneticDistance <dbl>`

excludes the SNP tested and those within this distance from the genetic similarity matrix. To use this feature genetic distances must be included in the PLINK files specifying both the test SNPs and the SNPs used for genetic similarity. The SNPs must be in non-decreasing order according to chromosome and the distance used.

`-excludeByPosition <int>`

excludes the SNP tested and those within this distance from the genetic similarity matrix. To use this feature positions must be included in the PLINK files specifying both the test SNPs and the SNPs used for genetic similarity. The SNPs must be in non-decreasing order according to chromosome and the distance used.

`-autoSelect filename`

determines the SNPs to include in the similarity matrix (see description in main text). When this option is used, GWAS is not run. SNPs are written to `filename.snps.txt`, and statistics of the SNP search are written to `filename.xval.txt`

`-autoSelectSearchValues <int_list>|<filename>`

when running AutoSelect, this specifies which number-of-SNPs to use for the search. The `<argument>` is either a “list-of-integers” on the command line or the name of a file containing a “list-of-integers.” A “list-of-integers” is a string of integers separated by a whitespace or comma. If you use a list on the command line, it should be enclosed in double quotes. The default values are “0 1 2 3 4 5 6 7 8 9 10 20 30 40 50 60 70 80 90 100 125 160 200 250 320 400 500 630 800 1000”.

`-autoSelectCriterionMSE`
directs AutoSelect to use out-of-sample mean-squared error for the selection criterion (default).

`-autoSelectCriterionLL`
directs AutoSelect to use out-of-sample log likelihood for the selection criterion

`-autoSelectFolds <int>`
specifies the number of folds to be used in AutoSelect. The default is 10.

`-randomSeed <int>`
specifies the random seed used in AutoSelect.

`-topKbyLinReg <int>`
directs AutoSelect to use only the top `<int>` SNPs, as determined by linear regression, while selecting SNPs.

`-memoryFraction <dbl>`
specifies the fraction of memory to use when invoking `-topKByLinReg`. Default is 0.2.

References

1. Purcell S. *et al.* PLINK: a tool set for whole-genome association and population-based linkage analyses. *American Journal of Human Genetics* **81** 559-75 (2007).
2. Listgarten J. *et al.* Improved Linear Mixed Models for Genome-Wide Association Studies. *Nature Methods* 9: 525-526 June 2011 (doi:10.1038/nmeth.2037).
3. Hayes B.J. Visscher P.M. & Goddard M.E. Increased accuracy of artificial selection by using the realized relationship matrix. *Genetics Research* **91** 47-60 (2009).
4. Hoggart C.J. *et al.* Simultaneous analysis of all SNPs in genome-wide and re-sequencing association studies. *PLoS Genetics* **4** e1000130 (2008).
5. Setakis E. Stirnadel H. & Balding D.J. Logistic regression protects against population structure in genetic association studies. *Genome Research* **16** 290-6 (2006).

Revision History

Date	Author(s)	Description of Changes
12/2/2011	Heckerman Davidson	Update for v1.04 add -dosage support
3/13/2012		Update for v1.08 document new output formats option -verboseOut document new similarity option -extractSim
6/1/2012		Update for v2.00 license and numerous other changes Add warning about hostname usage during build process
8/31/2012		Update for V2.01 various bug fixes document new output formats option -SetOutputPrecision
10/1/2012		Update for v2.02 various bug fixes fix -LinReg output document new -PvaluePrintThreshold option
10/25/2012		Update for 2.03 various bug fixes document -AutoSelect options gzip support of dosage files