**GENO-DIVER Version 1**

**Running the Program**

**Running the program via command line**

At the current time executable files are available only for linux operating environments. To run the program place the following files in a folder:

- GenoDiver

- macs

- msformatter

Prior to running you will most likely need to check the permissions of the files and if so making them executable (i.e. type “chmod 755 GenoDiver”). Once the permissions have been changed you need to generate a parameter file and place it in the same folder as the previous three files. The parameter file is read by searching for key words that are capitalized and then followed by a colon. Therefore any other phrase will not utilized. ***After running the program it is a strongly recommended to check the top of the log file (i.e. log\_file.txt) within the output folder to ensure the correct parameters were read in based on the ones you specified in the parameter file prior to looking at any of the output!!***

To run the program type in “./GenoDiver” and then the name of your parameter file. Comments on the progress of the program will be printed to the screen in order to provide you with an idea of what part of the program it is currently on. Furthermore you can check the log file in order to provide a description about the progress.

**Trouble Shooting**

If the program is not running correctly the log\_file.txt program within the folder the output was directed should provide knowledge on why and where the simulation crashed or exited. If you are unsure of the problem an e-mail can be sent to [jthoward@ncsu.edu](mailto:jthoward@ncsu.edu) with a copy of the log-file in order for me to replicate the results. We have spent a large amount of time to identify and fix all of the bugs within the program, but if you think something didn’t come out as expected don’t’ hesitate to e-mail me.

**Running in batch mode (i.e. background)**

To submit the job in batch mode the following bash script can placed into a file.

#!/bin/bash

./GenoDiver << BLK

parameterfile.txt

BLK

Assuming the file is called run\_program.sh, the program can be ran without typing in the parameterfile.txt each time the simulation is ran by just typing “./Batch\_Script.sh”. The bash script is located on the git-hub page and is called “Bash\_Script.sh”.

**Running in replicates with the same parameters but different seeds**

To perform multiple replicates of a simulation scenario a simple bash script can be utilized. The script first generates sequence information that can be utilized for all replicates and after the first one the simulations started at the founder generation to create different genetic architectures. The important information can be outputted into a desired folder. It is outlined below and the folder where the files go is “/home/jthoward/PIG\_ROH/Haplofinder/simulation/datafiles3/LD\_very\_very\_high/”. The bash script is located on the git-hub page and is called “Replicate\_Script.sh”.

**Program Parameters**

**Complete Parameter File**

A parameter file that specifies all the parameters in outlined on the git-hub page and is called “complete\_parameter\_file.txt”. Not all of them are required for the program to run, which was done to reduce the complexity of running the simulation. The first example is the minimal parameter file that is required for the program to run. All key words are in capital letters and the parameter(s) specified are separated by spaces. ***Only parameters after the key words impact the simulation.***

A description of all the parameters that can be specified in the program is outlined below. Furthermore, the appendix also contains helpful hints and suggestions on choosing more complex scenarios.

**Parameter Descriptions**

**--------------------------------------------------------------------------------**

**General and Starting Parameters**

**--------------------------------------------------------------------------------**

**START**

Description:

Decides where to start the simulation program either at the sequence or founder step. This option was included to save time and the ability to only generate sequence data once.

Value:

sequence – Starts at sequence generation.

founder – Skips sequence generation and begins at founder generation utilizing sequence information from a previous run.

Usage: “START: sequence”

Type: Mandatory.

Note:

If you are using a different effective population size or MaCS diversity metric you always have to start at the sequence step and any replicate that has the same MaCS parameters can start at the founder step.

**OUTPUTFOLDER**

Description:

Decides where to place the files that are generated from the simulation. A description of the files is outlined in the Appendix I.

Value: Any valid folder name.

Usage: “OUTPUTFOLDER: Scenario1”

Type: Optional. Default is “GenoDiverFiles”.

Note:

If there is already information in the file it will delete files depending on where the start position is. If the start position is “sequence” everything is deleted. If the start position is founder everything but sequence information is deleted. The sequence files could potentially take up a large amount of memory. Lastly, only one sequence file can be generated at a time unless the mac, msformatter and GenoDriver are placed in separate folders then it is allowed.

**SEED**

Description: Declares the seed number.

Value: Can be any integer value.

Usage: “SEED: 1501”

Type: Optional. Default is the system time.

Note:

If not declared by user it will appear in log-file (line 8) in order to generate the same data set again.

**NTHREAD**

Description: Declares the number of threads used for parallel processing.

Value: 1 to max number of cores.

Usage: “NTHREAD: 4”

Type: Optional. Default is 1.

**--------------------------------------------------------------------------------**

**Genome and Marker Information**

**--------------------------------------------------------------------------------**

**CHR**

Description: Sets the number of chromosomes to generate.

Value: Integer value ranging from 1 to 30.

Usage: “CHR: 3”

Type: Mandatory.

**CHR\_LENGTH**

Description: Sets the length of each chromosome, in Megabases.

Value: Integer Value

Usage: “CHR\_LENGTH: 150, 150, 150”

Type: Mandatory.

Note:

There can’t be any spaces after the final chromosome length number or else it will exit the program.

**NUM\_MARK**

Description:

Sets the number of markers per chromosome and are evenly spread across each chromosome.

Value: Integer Value

Usage: “CHR\_LENGTH: 4000, 4000, 4000”

Type: Mandatory.

Note: Quantitative and fitness QTL’s cannot be markers.

Note:

There can’t be any spaces after the final chromosome number of markers or else it will exit the program. The number of markers has to be less than the available number of mutations generated from MaCS and if it less than that it will exit the program.

**MARKER\_MAF**

Description: Minimum allele frequency allowed for markers.

Value: range from 0.0 to 0.50.

Usage: “MARKER\_MAF: 0.05”

Type: Optional. Default is 0.05.

**QTL**

Description:

The number of quantitative QTL for each chromosome. The quantitative QTL location is generated based on a uniform distribution from 0 to the length of the chromosome.

Value: Integer Value

Usage: “QTL: 50”

Type: Mandatory.

Note:

The number of quantitative QTL may be less than specified if multiple QTL, either quantitative or fitness, fall within the same mutation. Lastly,

**QUANTITATIVE\_MAF**

Description: Minimum allele frequency allowed for quantitative QTL.

Value: range from 0.0 to 0.50.

Usage: “QUANTITATIVE\_MAF: 0.05”

Type: Optional. Default is 0.05.

**FIT\_LETHAL**

Description:

The number of lethal fitness QTL for each chromosome. The lethal fitness QTL location is generated based on a uniform distribution from 0 to the length of the chromosome.

Value: Integer Value

Usage: “FIT\_LETHAL: 25”

Type: Optional. Default is 0.

Note:

The number of lethal fitness QTL may be less than specified if multiple QTL, either quantitative or fitness, fall within the same mutation. Lastly, these QTL don’t have any covariance with the quantitative trait.

**FIT\_SUBLETHAL**

Description:

The number of sub-lethal fitness QTL for each chromosome. The sub-lethal fitness QTL location is generated based on a uniform distribution from 0 to the length of the chromosome.

Value: Integer Value

Usage: “FIT\_SUBLETHAL: 25”

Type: Optional. Default is 0.

Note:

The number of sub-lethal fitness QTL may be less than specified if multiple QTL, either quantitative or fitness, fall within the same mutation. Lastly, these QTL don’t have any covariance with the quantitative trait.

**FITNESS\_MAF**

Description: Minimum allele frequency allowed for fitness QTL.

Value: range from 0.0 to 0.50.

Usage: “FITNESS\_MAF: 0.02”

Type: Optional. Default is 0.02.

Note:

Care needs to be taken in choosing a value that is not too large. If it is set to a high value a large number of founders will be killed and the simulation will exit due to a lack of founders required to generate the specified number of males and females.

**FOUNDER\_HAPLOTYPES**

Description:

The number of haplotypes generated by MaCS (Chen et al. 2009). Need to ensure that it is greater than two times the total number of animals needed in the founder population.

Value: Integer Value

Usage: “FOUNDER\_HAPLOTYPES: 4000”

Type: Optional. Default is based on the number of males and females needed within each generation.

**HAPLOTYPE\_SIZE**

Description:

The number of markers contained within a non-overlapping haplotype window. This is used to to generate haplotype based relationship matrices and a summary statistic at the end to measure the number of unique haplotypes contained within a window.

Value: Integer Value

Usage: “HAPLOTYPE\_SIZE: 50”

Type: Optional. Default is 50.

**RECOMBINATION**

Description:

The distribution that generates the location of recombination events. The number of recombination events is generated from a Poisson distribution with a rate parameter fixed at 1.0 across all chromosomes.

Value:

Uniform – Recombination sampled from a Uniform distribution from 0 to 1.0.

Beta – Recombination sampled from a Beta distribution from 0 to 1.0. The parameters that specify the distribution are both set at 0.5 to allow for greater recombination’s to occur at the end of the chromosome than towards the middle.

Usage: “RECOMBINATION: Uniform”

Type: Optional. Default is Uniform.

**--------------------------------------------------------------------------------**

**QTL Distributions**

**--------------------------------------------------------------------------------**

**ADD\_QUAN**

Description:

The parameters that generate the un-scaled quantitative QTL additive effects derived from a gamma distribution. Gamma effects are only positive therefore a binomial distribution with equal frequency of negative or positive effect is sample to generate sign of effect. QTL additive effects are simulated prior to creating the founder population. The effects are scaled such that the sum of the QTL variances in the founder population is equivalent to the proportion to the variance that is due to additive gene action specified.

Value:

shape – shape of gamma distribution.

scale – scale of gamma distribution.

Usage: “ADD\_QUAN: 0.4 1.66”

Type: Optional. Default is 0.4 1.66.

**DOM\_QUAN**

Description:

The parameters that generate the quantitative QTL degree of dominance values derived from a normal distribution. QTL additive effects are simulated prior to creating the founder population. The effects are scaled such that the sum of the QTL variances in the founder population is equivalent to the proportion to the variance that is due to dominant gene action specified. The use has some control over the proportion of dominance effects that display partial or over-dominance by changing the “mean” and “sd” values specified.

Value:

mean – mean of a normal distribution.

sd – standard deviation of a normal distribution.

Usage: “DOM\_QUAN: 0.1 0.2”

Type: Optional. Default is 0.1 0.2.

**LTHA**

Description:

The parameters that generate the lethal fitness QTL additive selection coefficients derived from a gamma distribution. The fitness of the unfavorable homozygous genotype is represented as 1 minus the selection coefficient and across loci fitness acts multiplicatively. No scaling is done on the fitness traits at the current time.

Value:

shape – shape of gamma distribution.

scale – scale of gamma distribution.

Usage: “LTHA: 0.3 0.1”

Type: Optional. Default is 0.3 0.1.

**LTHD**

Description:

The parameters that generate the lethal fitness QTL degree of dominance derived from a gamma distribution. The fitness of the heterozygote is the 1 minus the selection coefficient times the degree of dominance. No scaling is done on the fitness traits at the current time.

Value:

mean – mean of a normal distribution.

sd – standard deviation of a normal distribution.

Usage: “LTHD: 0.05 0.1”

Type: Optional. Default is 0.05 0.1.

**SUBA**

Description:

The parameters that generate the lethal fitness QTL additive selection coefficients derived from a gamma distribution. The fitness of the unfavorable homozygous genotype is represented as 1 minus the selection coefficient and across loci fitness acts multiplicatively. No scaling is done on the fitness traits at the current time.

Value:

shape – shape of gamma distribution.

scale – scale of gamma distribution.

Usage: “SUBA: 0.1 0.2”

Type: Optional. Default is 0.3 0.1.

**SUBD**

Description:

The parameters that generate the lethal fitness QTL degree of dominance derived from a gamma distribution. The fitness of the heterozygote is the 1 minus the selection coefficient times the degree of dominance. No scaling is done on the fitness traits at the current time.

Value:

mean – mean of a normal distribution.

sd – standard deviation of a normal distribution.

Usage: “SUBD: 0.3 0.1”

Type: Optional. Default is 0.05 0.1.

**COVAR**

Description:

Determines the relationship between the additive effects of QTL’s associated with quantitative and sub-lethal traits. The relationship is a function of the number of QTL that are both quantitative and sub-lethal and the rank correlation between the effects.

Value:

Proportion of Pleiotropic QTL – Number of QTL that are both quantitative and sub-lethal

Genetic correlation – The rank correlation between QTL effects.

Usage: “COVAR: 0.5 0.2”

Type: Optional. Default is 0.0 0.0.

Note:

Care needs to be taken in order to generate the type of relationship that is desired between the quantitative and fitness trait. Fitness values range from 0 to 1 and higher values leading to a lower fitness value. If the two traits are antagonistic than under the scenario of high values being favorable, a positive correlation should be given. In order to attain the correct scaling for quantitative trait, a trivariate reduction algorithm is utilized, which only allows the correlation to be positive. Therefore due to this at the current time only positive correlations are allowed. One just needs to change the favorable direction of the quantitative trait to alter the interpretation.

**--------------------------------------------------------------------------------**

**Population Parameters**

**--------------------------------------------------------------------------------**

**FOUNDER\_Effective\_Size**

Description:

Used to generate the population history of the haplotypes generated using the MaCS software (Chen et al. 2009). There are multiple scenario’s that are generated to represent a wide range of LD patterns that are generated by altering the population history parameters (i.e. “eN”) or one can specify their own effective population size and population history. Lastly, a single value can be utilized and the historical population parameter will not be utilized. A thorough description is outlined in the Appendix II.

Value:

- “Ne70”: A scenario that generates a large amount of short range LD.

- “Ne100\_Scen1”: A scenario that generates moderate short range LD.

- “Ne100\_Scen2”: A scenario that generates moderate short range LD.

- ”Ne250”: A scenario that generates the minimal LD.

- ”Ne1000”: A scenario that generates the minimal LD.

- “CustomNe” - Read in own population history parameters to read into MaCS

- Any value greater than 1: Utilizes the value as the effective population size and no population history assumed.

Usage: “FOUNDER\_Effective\_Size: 50”

Type: Mandatory.

**MUTATION**

Description:

Used in the MaCS software as the scaled mutation parameter (Chen et al. 2009) and in the simulation to generate mutation events as generations proceed.

Value:

Mutation Rate– Probability of a new mutation occurring at a given base pair and follows the infinite alleles model. The total number of mutations occurring within a new gamete is generated from a Poisson distribution with a rate parameter equal to the mutation rate times the length of the chromosome in nucleotides.

Proportion of Mutations that can be QTL: This parameter is utilized after the sequences are generated using MaCS and represents the number of non-neutral mutations that occurred out of the total number of new mutations. Each type of QTL (i.e. quantitative, lethal, sub-lethal) has an equal change of being picked as a new QTL.

Usage: “MUTATION: 2.5e-8 0.0”

Type: Optional. Default is 2.5e-8 0.0.

**VARIANCE\_A**

Description: Proportion of variance due to additive gene action.

Value: Range from 0.0 to 1.0

Usage: “VARIANCE\_A: 0.35”

Type: Mandatory.

**VARIANCE\_D**

Description: Proportion of variance due to dominant gene action.

Value: Range from 0.0 to 1.0

Usage: “VARIANCE\_D: 0.05”

Type: Mandatory.

Note:

Care needs to be taken in determining the additive and dominance variance and it implications on the number of quantitative QTL loci that display over-dominance or partial-dominance. Parameters to change that will impact the dominance variance include the QTL MAF frequency, mean and standard deviation of the normal distribution that generates the degree of dominance parameter and the ratio of additive to dominance variance.

**--------------------------------------------------------------------------------**

**Selection and Mating Parameters**

**--------------------------------------------------------------------------------**

**GENERATIONS**

Description: Determines the number of generations you want the simulation to run.

Value: Range from 0 to 40

Usage: “GENERATIONS: 10”

Type: Mandatory.

**INDIVIDUALS**

Description:

Determines the number of males and females in each generation and replacement rate for parents each generation. Care should be taken on picking the number of offspring in order to have enough to chose for the next generation. If number of animals falls below input value will exit program.

Value: Male Number Male Replacement Female Number Female Replacement

Usage: “INDIVIDUALS: 50 0.2 600 0.2”

Type: Mandatory.

**PARITY\_MATES\_DIST**

Description:

Determines the distribution on the number of mating pairs a sire has for each age group. The distribution is generated from a Beta that is parameterized by two parameters. The number of mating pairs for a given age class is determined by splitting the cumulative distribution function (CDF) into quadrants based on the number of age classes that occur within a generation. The proportion of mating pairs out of the total that are appropriated to a given age class is then based on the proportion that fall within the CDF quadrant.

Value: Both parameters have to be positive values.

Usage: “PARITY\_MATES\_DIST: 1 1”

Type: Optional. Default is both parameters being 1, which is very similar to a uniform distribution, such that all age classes have the same proportion of mating pairs.

**PROGENY**

Description: Determines the number of progeny produced by each mating pair. This may not be the actual number of progeny produced if lethal and/or sub-lethal QTL exist in the population. A progeny of a mating pair will be produced if the fitness value of the progeny is greater than a random value derived from a uniform distribution ranging from 0 to 1. The fitness value of the progeny is derived from the multiplicative effect of all lethal and sub-lethal QTL effects.

Value: ranges from 1 to 10.

Usage: “PROGENY: 4”

Type: Mandatory.

**MAXFULLSIB**

Description: Determines the maximum number of full-sibling progeny selected within a family. Once the maximum number is reached the full-sibling with the lowest selection criteria is no longer selected and the next best animal is selected. This process is repeated across all families until all families are below the value.

Value: ranges from 1 to number of progeny produced per each mating pair.

Usage: “MAXFULLSIB: 2”

Type: Optional. Default set at number of progeny produced per each mating pair.

**SELECTION**

Description: The metric used to select individuals and in what direction is favorable.

Value:

select - random, phenotype, true\_bv or ebv

direction – high or low

Usage: “SELECTION: ebv high”

Type: Mandatory.

Note:

For random selection the direction does not impact the results and therefore the direction value doesn’t matter.

**SOLVER\_INVERSE**

Description: Parameters that decide which relationship matrix to use to estimate breeding values, how they will be solved and how the inverse will be calculated.

Value:

relationship matrix –

- pedigree: constructed from the pedigree.

- genomic: constructed from genomic information as outlined in Van Raden (2008) and computing strategies were constructed based on Aguilar et al. (2011).

- h1:

- h2:

- h3: run of homozygosity based.

solver –

- direct: inverts the LHS of mixed model equations.

- pcg: uses the iterative preconditioned conjugate gradient method .

inverse –

- cholesky: update previous inverse based on the algorithm presented by Meyer et al. (2012)

- recursion: utilizing the sequential update algorithm presented by Misztal et al. (2014)

direction – high or low

Usage: “SOLVER\_INVERSE: ebv high”

Type: Mandatory.

Note:

Inverse calculations are only called for genomic-based relationships. Based on the current program design the “cholesky” method is less computationally expensive than the “recursive” method and is therefore preferred. Lastly, at the current time direct solver is faster than the pcg method and is therefore the preferred method.

**MATING**

Description: Parameters that decide how animals are mated.

Value:

random – males and females are randomly mated.

random5 – relationships ≥ 0.5 are not allowed to mate otherwise same as random.

random25 - relationships ≥ 0.25 are not allowed to mate otherwise same as random5.

random125 - relationships ≥ 0.125 are not allowed to mate otherwise same as random25.

Usage: “MATING: random5”.

Type: Mandatory.

Note:

At the current time only avoidance mating’s are used to minimize inbreeding. For each individual a pedigree that is 2 generation deep is recorded and a relationship matrix is constructed from this. Any relationships that are below the cutoff are set to 0 and a simulated annealing algorithm is utilized to find the mating pairs that minimize parental relationships.

**CULLING**

Description: The metric used to cull individuals and at what age an animal is removed due to old age. The direction will be the same one that is used during the selection stage.

Value:

cull- random, phenotype, true\_bv or ebv

max age – any number greater than 1

Usage: “CULLING: ebv 5”

Type: Mandatory.

**--------------------------------------------------------------------------------**

**Output Options**

**--------------------------------------------------------------------------------**

**OUTPUT\_LD**

Description: Used to determine if you need to calculate the linkage dis-equilibrium decay based on the r2 metric for each generation.

Value: “yes” or “no”

Usage: “OUTPUT\_LD: no”

Type: Optional. Default is no.

**GENOTYPES**

Description: Used to determine if genotypes should be exported for a given generation

Value: Either “no” or “yes” and if “yes” provide the generation at which to start exporting genotypes.

Usage: “GENOTYPES: yes 0”

Type: Optional. Default is “yes” and starting at generation 0.

Note:

Generation 0 refers to founder generation.

**Supplemental Information**

**--------------------------------------------------------------------------------**

**Appendix I – Output Information**

**--------------------------------------------------------------------------------**

The folder that contains the information generated from the simulation program contains multiple files and each one is described below:

**Data Summary Files**

**Summary\_Statistics\_QTL:**

*Generation:* Generation Number.

*Quant\_Founder\_Start:* Number of quantitative QTLs derived from founder generation that aren’t fixed.

*Quant\_Founder\_Lost:* Number of quantitative QTLs derived from founder generation that are fixed.

*Mutation\_Quan\_Total:* Number of quantitative QTLs derived from mutation events that aren’t fixed.

*Mutation\_Quan\_Lost:* Number of quantitative QTLs derived from mutation events that are fixed.

*Additive\_Var:* True additive genetic variance based on 2pq[a+d(q-p)]2 summed across quantitative QTLs.

*Dominance\_Var:* True dominance genetic variance based on (2pqd)2 summed across quantitative QTLs.

*Fit\_Founder\_Start:* Number of fitness QTLs derived from founder generation that aren’t fixed.

*Fit\_Founder\_Lost:* Number of fitness QTLs derived from founder generation that are fixed.

*Mutation\_Fit\_Total:* Number of fitness QTLs derived from mutation events that aren’t fixed.

*Mutation\_Fit\_Lost:* Number of fitness QTLs derived from mutation events that are fixed.

*Avg\_Haplotypes\_Window:* Average number of unique haplotypes contained within a haplotype window

*ProgenyDiedFitness:* Number of progeny that died due to fitness.

**Summary\_Statistics\_DataFrame\_Inbreeding:**

*Generation:* Generation Number.

*ped\_f:* Average pedigree based inbreeding parameter.

*gen\_f:* Average genomic relationship diagonal constructed based on Van Raden (2008).

*h1\_f:* Average diagonal of haplotype based relationship matrix.

*h2\_f:* Average diagonal of haplotype based relationship matrix.

*h3\_f:* Average diagonal of ROH based relationship matrix.

*homozy:* Average proportion of the genome homozygous based on marker genotypes.

*fitness:* Average multiplicative fitness value of an individual.

*homozlethal:* Average number of homozygous fitness QTL classified as lethal.

*hetezlethal:* Average number of heterozygous fitness QTL classified as lethal.

*homozysublethal:* Average number of homozygous fitness QTL classified as sub-lethal.

*hetezsublethal:* Average number of heterozygous fitness QTL classified as sub-lethal.

*lethalequiv:* Average lethal equivalents, which is computed as the sum of all fitness selection coeffecients for an individual animal.

**Summary\_Statistics\_DataFrame\_Performance:**

*Generation:* Generation number with the variance in parenthesis.

*phen*: Average phenotypic value with the variance in parenthesis.

*ebv:* Average estimated breeding value with the variance in parenthesis.

*gv:* Average true genotypic breeding value with the variance in parenthesis (Sum of additive + dominance effects within an individual).

*bv:* Average true genotypic breeding value with the variance in parenthesis (Sum of additive effects within an individual).

*dd:* Average true dominance deviation with the variance in parenthesis (Sum of dominance effects within an individual).

*res:* Average residual value with the variance in parenthesis (Sum of additive effects within an individual).

**LD\_Decay**

Is a file that has the average correlation (r2) between two SNP across a range of distances. The distances are in the first row and are in Kilobases. Each row after the first row corresponds to the generation, such that line 2 is generation 0, line 3 is generation 1, etc..

where the subscript refers to either SNP marker 1 or 2 and D is equal to (A1B1 \* A2B2) – (A1B2\*A2B1).

**Data Files**

**log\_file.txt**: This file displays a great deal of information on specifics within each generation and it is advisable that one should look over it after you try a new simulation protocol.

**Low\_Fitness**: A file that describes the animals that died due to the fitness effect.

*Sire*: sire of dead progeny

*Dam*: dam of dead progeny

*Fitness*: Overall fitness value of individual. An animal dies if the random value generated from a uniform distribution is greater than the fitness value.

*QTL\_Fitness*: The genotypes for Fitness QTL for the individual.

**Marker\_Map**: A map file that corresponds to the chromosome and nucleotide position of marker.

*chr:* Chromosome marker is on.

*pos*: Nucleotide position of marker.

**Master\_DataFrame**: A file that contains multiple statistics for individuals that survived.

*ID*: Identification of individual.

*Sire*: Sire Identification of individual.

*Dam*: Dam Identification of individual.

*Sex*: Sex of individual; 0 is male and 1 is female.

*Gen*: Generation the animal was born.

*Age:* Age the animal was removed from the population either at the culling or selection stage.

*Progeny*: Number of progeny generated.

*Dead*: Number of dead progeny generated.

*Ped\_F*: Pedigree based inbreeding metric.

*Gen\_F*: Diagonal of genomic based relationship constructed based on Van Raden (2008).

*Hap1\_F*: Diagonal of haplotype 1 based relationship matrix.

*Hap2\_F*: Diagonal of haplotype 2 based relationship matrix.

*Hap3\_F*: Diagonal of ROH based relationship matrix.

*Homolethal*: Number of homozygous lethal genotypes.

*Heterlethal*: Number of heterozygous lethal genotypes.

*Homosublethal*: Number of homozygous sub-lethal genotypes.

*Hetersublethal*: Number of heterozygous sub-lethal genotypes.

*Letequiv*: Lethal equivalent value, which is the sum of the selection coefficient values.

*Homozy*: Proportion of the genome homozygous.

*Fitness*: Multiplicative Fitness value of the invididual.

*Phen*: Phenotype.

*EBV*: Estimated Breeding Value,

*Acc*: Accuracy of EBV (not included yet).

*GV*: True genotypic value of individual (i.e. sum of BV of DD).

*BV*: True breeding value of individual (i.e. sum of additive effects)*.*

*DD*: True dominance deviation of individual (i.e. sum of dominance effects).

*R*: Residual value of individual.

**Master\_Genotypes**: A file that contains genotypic information for individuals that survived.

*ID*: Identification of individual.

*Marker*: Marker genotypes of individual (0-11; 2-22; 3-12; 4-21).

*QTL*: QTL genotypes of individual (0-11; 2-22; 3-12; 4-21).

**QTL\_new\_old\_Class**: A file that contains information on QTL effects and frequency across generations.

*Chr:* Chromosome QTL is located.

*Pos*: Nucleotide position of QTL.

*Additive\_Selective*: If it is a quantitative trait it refers to the additive effect and if it is a fitness QTL it refers to the selection coefficient.

*Dominance:* The dominance effect for the quantitative or fitness QTL.

*Type*: Refers to the type of QTL:

2 – quantitative trait

4 – fitness lethal

5 – fitness sub-lethal

*Gen*: Generation at which the

*Freq*: Gene frequency across generations with a “\_” as the delimeter.

**Supplementary Files**

CH\*SNP.txt – Haplotype sequence for each chromosome simulated from MaCS.

Map\*.txt – map file corresponding to haplotypes sequence in CH\*SNP.txt

FounderGenotypes – Genotypes across chromosomes for each founder. The line number corresponds to the founder ID and the first column represents the row number of the two haplotypes that created the genotype, followed by the genotype string.

Ginv\_Matrix: The inverse of the Genomic relationship matrix in binary format.

G\_Matrix: The Genomic relationship matrix in binary format.

Linv\_Matrix: The inverse of the cholesky matrix from the previous generation that is utilized to construct inverse relationship matrix using the method outlined by Meyer et al. (2013) to obtain the inverse. This file is in binary format.

Pheno\_GMatrix: Dataframe utilized in constructed genomic relationship matrix.

Pheno\_Pedigree: Dataframe utilized in constructing pedigree relationship matrix.

Previous\_Beta\_PCG: Estimates of solutions for previous generation.

SNPFreq: Frequency of SNP across all chromosomes derived from MaCS.

**--------------------------------------------------------------------------------**

**Appendix II – MaCS Sequence Simulation**

**--------------------------------------------------------------------------------**

The founder genomic information is generated from the MaCS program (Markovian Coalescence Simulator; Chen et al. 2009). Prior to using the program it is advisable to fully understand the coalescent process and a good review paper is Hudson (1991) and Chapter 5 of Charlesworth & Charleworth (2010). The use of MaCS allows for the ancestral population to vary in terms of its population history and size. This allows for different linkage disequilibrium (LD) patterns to be simulated and therefore a variety of species can be simulated. A large number of program utilize MaCS to generate their genomic information such as AlphaSim (Hickey & Gorjanc, 2012) and ms2gs (Pérez-Enciso & Legarra, 2016) and similar MaCS populations history parameters were also utilized in this simulation.

There are 5 default scenarios that represent a range of linkage disequilibrium (LD) patterns that can be called within the simulation, as outlined in the figure below. The five scenarios are called by specifying either “Ne70”, “Ne100\_Scen1”, “Ne100\_Scen2”, “Ne250” or “Ne1000” after the FOUNDER\_Effective\_Size parameter in the parameter file. The user can input a custom effective population size and historical population parameters by using “CustomNe” as the parameter. An easy way to generate your own parameters for MaCS is to utilize a default scenario that resembles the pattern you are wanting and to change the effect population size of the population and determine how the LD pattern changes. If this is specified the program looks for a file called “CustomNe” that is placed in the folder where the program is run. The file should contain two rows, with the first one being the effective population size parameter and the last one being the historical population size parameters. Lastly, if only an integer is given than the value is the effective population size and no population history is simulated.

The generation of sequence information may take some time to compute and the files generated from the program may be large. Due to this, it is advisable to only generate sequence data once for a given scenario and then adjust narrow-sense heritability, broad sense heritability, or selection and mating parameters and start with generating the founder generation.

The default scenarios are outlined below:

Ne70:

- Effective population size = 70;

- Historical population parameters: -eN 0.18 0.71 -eN 0.36 1.43 -eN 0.54 2.14 -eN 0.71 2.86 -eN 0.89 3.57 -eN 1.07 4.29 -eN 1.25 5.00 -eN 1.43 5.71

Ne100\_Scen1:

- Effective population size = 100;

- Historical population parameters: -eN 0.06 2.0 -eN 0.13 3.0 -eN 0.25 5.0 -eN 0.50 7.0 -eN 0.75 9.0 -eN 1.00 11.0 -eN 1.25 12.5 -eN 1.50 13.0 -eN 1.75 13.5 -eN 2.00 14.0 -eN 2.25 14.5 -eN 2.50 15.0 -eN 5.00 20.0 -eN 7.50 25.0 -eN 10.00 30.0 -eN 12.50 35.0 -eN 15.00 40.0 -eN 17.50 45.0 -eN 20.00 50.0 -eN 22.50 55.0 -eN 25.00 60.0 -eN 50.00 70.0 -eN 100.00 80.0 -eN 150.00 90.0 -eN 200.00 100.0 -eN 250.00 120.0 -eN 500.00 200.0 -eN 1000.00 400.0 -eN 1500.00 600.0 -eN 2000.00 800.0 -eN 2500.00 1000.0.

Ne100\_Scen2:

- Effective population size = 100;

- Historical population parameters: -eN 50.00 200.0 -eN 75.00 300.0 -eN 100.00 400.0 -eN 125.00 500.0 -eN 150.00 600.0 -eN 175.00 700.0 -eN 200.00 800.0 -eN 225.00 900.0 -eN 250.00 1000.0 -eN 275.00 2000.0 -eN 300.00 3000.0 -eN 325.00 4000.0 -eN 350.00 5000.0 -eN 375.00 6000.0 -eN 400.00 7000.0 -eN 425.00 8000.0 -eN 450.00 9000.0 -eN 475.00 10000.0.

Ne250:

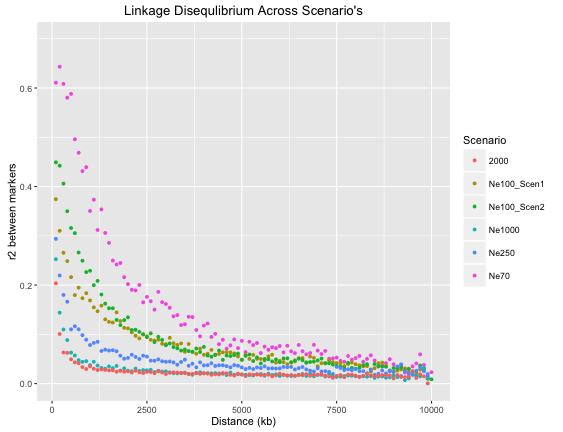
- Effective population size = 100;

- Historical population parameters: -eN 0 1.04 -eN 0 1.08 -eN 0 1.12 -eN 0 1.16 -eN 0.01 1.2 -eN 0.03 1.6 -eN 0.05 2.0 -eN 0.1 2.8 -eN 0.2 4.8 -eN 0.3 5 -eN 0.4 5.2 -eN 0.5 5.4 -eN 0.6 5.6 -eN 0.7 5.7 -eN 0.8 5.8 -eN 0.9 5.9 -eN 1 6 -eN 1 4 -eN 2 8 -eN 3 10 -eN 4 12 -eN 5 14 -eN 6 16 -eN 7 18 -eN 8 20 -eN 9 22 -eN 10 24 -eN 20 28 -eN 40 32 -eN 60 36 -eN 80 40 -eN 100 48 -eN 200 80 -eN 400 160 -eN 600 240 -eN 800 320 -eN 1000 400.

Ne1000:

- Effective population size = 100;

- Historical population parameters: -eN 0.50 2.00 -eN 0.75 2.50 -eN 1.00 3.00 -eN 1.25 3.20 -eN 1.50 3.50 -eN 1.75 3.80 -eN 2.00 4.00 -eN 2.25 4.20 -eN 2.50 4.50 -eN 5.00 5.46 -eN 10.00 7.37 -eN 15.00 9.28 -eN 20.00 11.19 -eN 25.00 13.10 -eN 50.00 22.66 -eN 100.00 41.77 -eN 150.00 60.89 -eN 200.00 80.00"



**Citations**

Aguilar, I., I. Misztal , A. Legarra & S. Tsuruta. 2011. Efficient computation of the genomic relationship matrix and other matrices used in single-step evaluation. J. Anim. Breed. Genet. 128: 422–428.

Charlesworth, B., & D. Charlesworth. 2010. Elements of Evolutionary Genetics. Greenwoord Village, Colorado, USA: Roberts and Company.

Chen, G. K., P. Marjoram & J. D. Wall. 2009. Fast and flexible simulation of DNA sequence data. Genome Res. 19(1):136-42.

Hickey, J. M., G. Gorjanc. 2012. Simulated Data for Genomic Selection and Genome-Wide Association Studies Using a Combination of Coalescent and Gene Drop Methods. G3 2:425–427

Hudson R. R. 1990. Gene genealogies and the coalescent process. Oxford Surveys in Evolutionary Biology. 7:1-45.

Meyer, K., B. Tier, and H.-U. Graser. 2013. Technical note: Updating the inverse of the genomic relationship matrix. J. Anim. Sci. 91:2583–2586.

Misztal, I. A. Legarra, and I. Aguilar. 2014. Using recursion to compute the inverse of the genomic relationship matrix.

Pérez-Enciso M., A. Legarra. 2016. A combined coalescence gene-dropping tool for evaluating genomic selection in complex scenarios (ms2gs). J. Anim. Breed. Genet. 133(2):85-91.

VanRaden P.M. 2008. Efficient Methods to Compute Genomic Predictions. J. Dairy Sci., 91, 4414–4423.

**Example Parameter Files**

**EXAMPLE COMPLETE PARAMETER FILE:**

------ Parameter file for Geno-Driver ---------

-----| Starting Point |-----

START: sequence

OUTPUTFOLDER: GenoDiverFiles

SEED: 1500

NTHREAD: 1

-----| Genome and Marker Information |-----

CHR: 4

CHR\_LENGTH: 150 150 150 150

NUM\_MARK: 4000 4000 4000 4000

MARKER\_MAF: 0.05

QTL: 50

QUANTITATIVE\_MAF: 0.05

FIT\_LETHAL: 0

FIT\_SUBLETHAL: 0

FITNESS\_MAF: 0.02

FOUNDER\_HAPLOTYPES: 4000

HAPLOTYPE\_SIZE: 50

RECOMBINATION: Uniform

-----| QTL Distributions |-----

ADD\_QUAN: 0.4 1.66

DOM\_QUAN: 0.1 0.2

LTHA: 0.3 0.1

LTHD: 0.05 0.1

SUBA: 0.1 0.2

SUBD: 0.3 0.1

COVAR: 0.5 0.2

-----| Population Characteristics |-----

FOUNDER\_Effective\_Size: Ne70

MUTATION: 2.5e-8 0.0

VARIANCE\_A: 0.25

VARIANCE\_D: 0.05

-----| Selection and Mating Parameters |-----

GENERATIONS: 2

INDIVIDUALS: 50 0.2 600 0.2

PARITY\_MATES\_DIST: 1 1

PROGENY: 1

MAXFULLSIB: 2

SELECTION: ebv high

SOLVER\_INVERSE: pedigree pcg cholesky

MATING: random5

CULLING: ebv 5

-----| Output Options |-----

OUTPUT\_LD: no

GENOTYPES: yes 0

----- Parameter file for Geno-Driver ---------

**EXAMPLE MINIMAL PARAMETER FILE: Only using default settings.**

-----| Starting Point |-----

START: sequence

-----| Genome and Marker Information |-----

CHR: 3

CHR\_LENGTH: 150 150 150

NUM\_MARK: 4000 4000 4000

QTL: 50

-----| Population Characteristics |-----

FOUNDER\_Effective\_Size: Ne70

VARIANCE\_A: 0.35

VARIANCE\_D: 0.05

-----| Selection and Mating Parameters |-----

GENERATIONS: 10

INDIVIDUALS: 50 0.2 600 0.2

PROGENY: 1

SELECTION: ebv high

SOLVER\_INVERSE: pedigree pcg cholesky

MATING: random5

CULLING: ebv 5