Last Updated: 12/11/2015

Table of Contents

Introduction

ATHENA applies grammatical evolution to optimize neural networks for detection and modeling of gene-gene interactions. It replicates the features of GENN and incorporates new features. It will be extended in the future to allow for additional search algorithms and different model representations.

Example

ATHENA takes one command line argument, a configuration file specifying all the parameters for the run.

```
athena example.config
```

Configuration file

ATHENA takes the name of a configuration file as its single command-line argument. The configuration file should list all the parameters for controlling the analysis. It should be in the format of keyword <whitespace> value. Each keyword should be on its own line. Comments begin with a '#' and will be ignored by the program.

Note on commonly modified parameters

Many of the parameters in ATHENA have default values and so do not have to be included in the configuration file. Sample configuration files are included with the

software package that can be downloaded from the laboratory website. These can be modified to suit your needs.

Some of the most commonly modified parameters include DATASET (which must designate the input file containing the phenotype and any SNP data), CONTINFILE (which contains any data not matching SNP genotype data) and GRAMMARFILE (which provides the grammar for the run). Sample grammar files are provided in the package download. The values in the DATASET can be adjusted by setting the DUMMYENCODE parameter and values in the CONTINFILE can be adjusted with the CONTINADJUST parameter. If a continuous status (outcome) variable is used, STATUSADJUST can normalize the values (NORMMAX is typically the option passed to the parameter in this case). Setting the OUT parameter specifies the base portion of all the output files for an analysis. NUMSTEPS specifies how many time times the number of generations specified in the algorithm are run. After each step, the best networks are exchanged if the parallelized version is run.

For algorithm-specific parameters, POPSIZE sets the population size of the evolutionary algorithm and larger sizes increase the search space (and run time) of an analysis. GENSPERSTEP sets the number of generations performed for each step of the analysis (as specified by NUMSTEPS). INCLUDEALLSNPS determines whether all the variables in the input files are included or only those specified in the grammar file. MAXDEPTH sets the maximum grammar depth for sensible initialization in GE so that a higher MAXDEPTH will result in larger initial networks in the analysis.

General parameters

General parameters affect the program as a whole and specify parameters such as input files and the start of algorithm specific parameters.

Data files

DATASET *filename*

Required parameter specifies path to dataset file for analysis.

MAPFILE *filename*

Optional parameter specifies names of input variables in DATASET file. If not specified, the variables will be identified as G1, G2, etc.

CONTINFILE *filename*

Optional parameter specifies filename containing continuous variables. If present, the individuals must be in the same order as the DATASET file.

CONTINMAP *filename*

Optional parameter specifies filename identifying variables in CONTINFILE. If not specified, the variables will be identified as C1, C2, etc. File can optionally identify variables as belonging to groups for purposes of normalization of variables within groups.

Data file parameters

IDINCLUDED *TRUE|FALSE*

If set to TRUE, then the first column of DATAFILE should contain an ID. The default is FALSE.

MISSINGVALUE *number*

Indicates the value for missing data in DATAFILE. The default is -1.

CONTINMISS *number*

Indicates the value for missing data in CONTINFILE. The default is -9999.

STATUSMISSINGVALUE *number*

Specifies the value for an individual in the DATAFILE with no status. These individuals will be skipped in the analysis. The default is -1.

DUMMYENCODE *ADD_QUAD|ADDITIVE|CUSTOM|NONE*

Specifies the encoding for values in DATASET. The default is none. ADD_QUAD converts the data into two dummy variables as described by Jurg Ott. ADDITIVE converts the data into a neural network compatible representation (0=>-1, 1=>0, 2=>1). CUSTOM allows for a user specified encoding to be used. It should follow the keyword CUSTOM and the values should be separated by a ':' as below:

DUMMYENCODE CUSTOM -1:1:2

CONTINADJUST *MINMAX|MINMAXGROUP|MAKECATEGORICAL|NONE*

Specifies data transformation to perform on the values in CONTINFILE. MINMAX divides the difference between a value and the maximum value by the difference between minimum and maximum in the variable. MINMAXGROUP performs the same operation except it does so within the groups that can be specified in the CONTINMAP file. MAKECATEGORICAL converts the values into categorical values (0,1,2,3,4,etc.). The default is none.

STATUSADJUST *MINMAX|NONE*

MINMAX adjusts the status value in the DATAFILE in the same manner as it does for the CONTINADJUST parameter. The default is none.

Other input files

SPLITFILE *filename*

Optional parameter specifies file to use for splitting data. ATHENA will not conduct its own cross-validation procedure for splitting data.

TRAINFILE *filename*

Training genotype data file that can be used instead of using DATASET and CV if user already has split data.

TESTFILE *filename*

Testing genotype data file that can be used instead of using DATASET and CV if user already has split data.

CONTRAINFILE *filename*

Training continuous variable input file that can be used instead of using CONTINFILE and CV if user already has split data.

CONTESTFILE *filename*

Testing continuous variable input file that can be used instead of using CONTINFILE and CV if user already has split data.

VALIDATIONSUMFILE *filename*

Optional parameter specifies a .sum file output by ATHENA in a previous run. ATHENA will run all the models in the .sum file against the DATAFILE and CONTINFILE (if present) for validation purposes.

BIOFILTERFILE *filename*

Optional parameter specifies a file listing two variable models that can be utilized by the algorithms in ATHENA. For example, in GENN the models are used to seed the initial population of networks.

General Algorithm

ALGORITHM *name*

Required parameter indicates start algorithm-specific parameters. Current algorithms are GENN (grammatical evolution neural networks) and GESYMBREG (grammatical evolution symbolic regression).

END

Keyword marks the end of the algorithm parameters.

NUMSTEPS *integer*

In parallel implementations, it specifies the number of times the best solutions are migrated between populations. The best solutions are exchanged after each interval specified in the GENSPERSTEP parameter of the algorithm. The default is 1.

RANDSEED *integer*

Seeds the random number generator. The default is 1.

CV *integer*

Required parameter specifies the number of cross-validations to perform.

CVRESTART *integer*

This parameter can be used to restart at a cross-validation interval later than the first. For example, if a system issue caused the interruption of the run after 2 cross-validations. The software could be run starting from "CVRESTART 3" and finish the run. The output will be appended to the existing files.

BESTSELECT *TRUE|FALSE*

When set to true, a single best model will be constructed from the best models across the cross-validations. The selection of the best model varies by algorithm. The default is false.

Output

OUT *path*

Specifies the output path for all output. Different extensions will be applied to the base name. The default is "athena."

INDOUTPUT *TRUE|FALSE*

When set to true, outputs the scores for every individual evaluated by the best model in each cross-validation. The default is false.

WRITECV *TRUE|FALSE*

If true, the individuals and genotypes used for training in each cross-validation are output to files named cv.1.txt, cv.2.txt, etc. Default is false.

SUMMARYONLY *TRUE|FALSE|BEST|ALL*

If set to true, only the .sum file is produced. When set to false, the .dot and .best files are also produced. Setting it to Best will produce the .sum and .best files, while selecting all will produce all the files listed plus the .all files that list all the models in the final population.

LOG *None|Overview|Summary|Variables*

Controls logging output. The default is none. Overview generates only the basic log file. Summary generates a .models.log file in addition to the basic log file. It contains a list of all models and scores during the run. Variables only records the variables in

the models in the .models.log file. Variables and Summary should only be used for short runs as they produce extensive output.

ALLNODESBEST *TRUE|FALSE*

When set to true, outputs best model for each node of a parallelized run at end of each cross-validation run. The default is false.

GENN/GESR parameters

GENN/GESR algorithm parameters only affect the parameters of the algorithm specified when running the program. The last four parameter in the table are recommended for using with GESR algorithm. They are optional for GENN.

Initialization

SENSIBLEINIT *TRUE|FALSE*

When set to true, the initial population will be constructed using the grammar provided by the user in such a way to insure that all members of the initial population will be valid. If false, the population will be constructed randomly and may not all form valid networks. This can be the case when the grammar involves recursion (as it usually does with GENN). The default is true.

GROWRATE *TRUE|FALSE*

When sensible initialization is on, the GROWRATE determines the fraction of the initial population that will be set to the Grow method rather than full. A full network is one that reaches the maximum depth of the grammar tree. The default is 0.5.

MAXDEPTH *integer*

This parameter establishes the maximum depth of network in grammar tree form. Sensible initialization uses it to insure that the resulting networks that are translated from the grammar do not exceed this limit. The default is 10.

TAILSIZE *integer*

Adds the specified number of codons to the end of the genome in the initial population. Default is 0.

TAILRATIO *number*

Adds the specified fraction of codons to the end of the genome in the initial population. Default is 0.0.

MINSIZE *number*

When not using sensible initialization, it specifies the minimum size of the genomes in the initial population. Default is 50.

MAXSIZE *number*

When not using sensible initialization, it specifies the maximum size of the genomes in the initial population. Default is 1000.

BIOFILTERFRACT *number*

Specifies the fraction of initial population that will be initialized using models provided by a bio filter file. If there aren't enough models in the file, the extra models will be initialized with either sensible initialization or random initialization based on that parameter. Default is 0.0.

BIOMODELSELECTION *ORDERED|ROULETTE*

Determines the method for selecting the models from the bio filter file. Options are ORDERED (where the models are taken in order of implication index) and ROULETTE (where the models are weighted based on implication index and selected randomly. Default is ORDERED.

GA

POPSIZE *integer*

Specifies the number of networks in each population. Default is 100.

PROBCROSS *number*

Specifies the probability of a crossover during evolution. Default is 0.9.

PROBMUT *number*

Specifies the mutation rate per codon in the genomes. Default is 0.01.

GENSPERSTEP *integer*

Specifies the number of generations performed for each step specified in the general parameters. Default is 100.

EFFECTIVEXO *TRUE|FALSE*

When set to true, crossovers occur only within the effective coding region of the genome. Default is false.

GE

GRAMMARFILE *filename*

Specifies required grammar file for use in grammatical evolution.

INCLUDEALLSNPS *TRUE|FALSE*

When true, all variables in the DATASET and CONTINFILE will be used regardless of what is specified in the grammar file. Default is false.

BLOCKCROSSGENS *integer*

Specifies the number of generations that crossover will use the block crossover which matches compatible regions of the genomes and insures the crossover will not be destructive. If the number exceeds the total number of generations in the run, block crossover will be used for the entire run. Default is 0.

Fitness

CALCTYPE *BALANCEDACC|AUC|RSQUARED*

Sets the fitness metric for evaluating individuals in the evolutionary population. BALANCEDACC calculates balanced accuracy in a case/control set while AUC uses

area under the curve in such a set. RSQUARED calculates r-squared for input data with a continuous outcome. Default is BALANCEDACC.

REQUIREALLVARS *TRUE|FALSE*

If true, only solutions that include all variables in the dataset are evaluated for fitness. Default is false.

REQUIREALLONCE *TRUE|FALSE*

Only solutions that include each variable once are evaluated for fitness. Default is false.

BACKPROPSTART *integer*
Specifies the first generation to run backpropagation on. If set to 0, will run backpropagation after initialization of population. If set to < 0, no backpropagation will occur (default). It should only be used with a grammar that contains only additive nodes.

BACKPROPFREQ *integer*
Specifies frequency of backpropagation during run. If set to zero, backpropagation will not repeat during the run after the generation specified by BACKPROPSTART.

FITNESSGOAL *number*
The run will terminate when the fitness reaches the value set. Otherwise, the run continues until all generations are completed.

Selection

GASELECTION *ROULETTE|DOUBLE*
Specifies the selection method for the evolutionary algorithm. Default is ROULETTE.

DOUBLETOURNF *integer*
Sets the number of individuals in the first round of the double tournament selection method. Default is 7.

DOUBTOURND *number*
The pressure that is put on parsimony during double tournament selection (D/2 = probability that the smaller solution wins in a size tournament, so for D=1.4 there is a 70% probability that the smaller individual will win)

DOUBTOURNFITFIRST *TRUE|FALSE*

If TRUE, fitness is tested first during the double tournament. If FALSE, the size tournament is first. Default is TRUE.

PRUNEPLANT *number*
Specifies the frequency of the prune and plant operation. After selection, crossover and mutation each individual is checked against the probability of prune and plant. One branch of the genome is replaced with a minimal branch and that pruned branch is then planted back into the population. Default is 0 and this options is currently only compatible with GESYMBREG.

Variables

NUMGENSRESTRICTVARS *integer*

Specifies the mumber of generations that the grammar used by the algorithm is restricted to only variables that are part of the initialized networks. Default is 0.

RESETVARSMIGRATION *TRUE|FALSE*

This parameter is used in conjunction with NUMGENSRESTRICTVARS. After a migration, the population will use a new grammar. If this parameter is true, the new grammar will only include variables that are in the population after the migration. When set to false, any new variables that migrated in will be added but all older variables will be maintained whether or not they are in the current population. Default is false.

Sample configuration file

```
# this is a comment
# sample configuration for use with ATHENA
ALGORITHM GENN
MINSIZE 20
MAXSIZE 300
MAXDEPTH 10
SENSIBLEINIT TRUE
POPSIZE 100
PROBCROSS 0.9
PROBMUT 0.01
GRAMMARFILE additive.gram
CALCTYPE BALANCEDACC
EFFECTIVEXO TRUE
GENSPERSTEP 10
```

```
INCLUDEALLSNPS TRUE
END GENN
# specify general parameters for run
DATASET sample.dat
IDINCLUDED FALSE
MISSINGVALUE -1
DUMMYENCODE TRUE
RANDSEED 771
OUT sample-output
CV 5
NUMSTEPS 2
WRITECV FALSE
```

Algorithms

GENN (Grammatical Evolution Neural Network)

Grammatical evolution (GE) is an evolutionary algorithm that uses linear genomes and grammars to define the populations. In GE, each individual consists of a binary genome divided into codons. Mutation takes place on individual bits but crossover only takes place between the codons. Translating codons using the grammar produces an individual or phenotype. The resulting individual can then be tested for fitness in the population and the usual evolutionary operators can be carried out. By using a grammar to define the phenotype, GE separates the genotype from the phenotype and allows greater genetic diversity within the population than other evolutionary algorithms. In GENN, the grammar creates a neural network that accepts variables from the dataset.

The type of status in the dataset determines the fitness used in the algorithm. When the status is binary (affected or unaffected), the fitness of a network is determined by the balanced accuracy, [(sensitivity+specificity)/2]. When the status is a continuous variable, fitness for the network is the R-squared (coefficient of determination).

ATHENA can be run using a cross-validation framework. In that case the data are divided into a training set and a testing set for each cross-validation interval. For example, in 10-fold cross-validation the training set will be 9/10 of the data and the testing set will be 1/10. The training set is utilized to set the fitness of each solution in the population during the running of the algorithm. After the best neural network is produced, its predictive ability is evaluated by determining the score of the testing set.

Symbolic Regression

ATHENA can use symbolic regression as another algorithm using grammatical evolution and the same cross-validation framework. The goal of symbolic regression (SR) is to find a mathematical function that accurately maps independent variables to a dependent variable. This is different from linear or logistic regression in that you do not have to specify the coefficients, the variables, or how they are structured together in advance. A popular way of adjusting the symbolic function is by using computational evolution.

Files

Input files

ATHENA utilizes a variety of input files. ATHENA accepts data in a simple format. Each line is a separate individual. The first column is the ID (if that option is on) or it is the status. After that information, each additional column contains the value at a locus for the genotype data. The continuous data file is similar except there is no status column.

Genotype data file (no ID)

ATHENA accepts data in a simple format. Each line is a separate individual. The first column is the ID (if that option is on) or it is the status. After that information, each additional column contains the value at a locus for the genotype data. The continuous data file is similar except there is no status column.

```
0 2 1 2 2 1 1 2 2 2 2 1 1
0 1 2 0 1 0 2 1 1 2 1 2 1
0 0 1 2 1 0 1 2 0 1 2 2 1
0 1 1 2 0 1 1 2 1 2 0 1 1
0 2 2 2 1 1 0 2 1 1 2 0 1
1 2 1 2 1 2 1 1 2 1 2 0 1
1 2 1 0 1 1 1 1 2 2 1 0 0
1 1 2 1 2 1 2 1 2 0 0 2 1
1 1 2 1 2 0 2 1 1 1 2 1 2
1 2 2 0 0 1 1 1 0 0 2 1 1
```

Genotype data file (with ID)

```
ind1  0 1 1 1 1 1 1 2 0 2 0 1
ind2  0 2 2 2 1 2 1 1 0 1 1 1 1
ind3  0 2 1 0 2 1 1 1 2 2 1 1 1
ind4  0 1 1 1 2 2 1 1 2 1 1 1 2
ind5  0 2 1 1 2 1 1 1 1 1 2 0 2
ind6  1 2 2 1 1 1 0 2 1 1 2 1 1
ind7  1 1 1 2 2 1 2 2 1 1 2 1 2
ind8  1 1 1 1 1 1 1 1 1 2 2 1 2
ind9  1 1 2 1 0 2 1 1 2 0 1 1 0
ind10 1 2 1 2 1 2 1 1 1 2 2 1 1
```

Map file

The map file must be ordered to match the data file. It consists of three columns: chromosome, ID and position. Currently, ATHENA only uses the ID column and so any values can be placed in the first and third columns

```
1        rs1      10502
1        rs2      220020
1        rs3      303034
2        rs4      10201
3        rs5      3303049
```

Continuous data file

The continuous data input file is very similar to the genotype data input file except the numbers are restricted to 0,1,2. If the genotype data file has an ID, then the continuous input file must have matching IDs in it.

```
ind1  22.5 114.8 0.5
ind2  11.8 122   0.7
ind3  17.3 119.5 0.56
ind4  15.8 120.3 0.72
ind5  19.2 118.2 0.88
ind6  9.5  98.8  0.77
ind7  14.8 112.4 0.35
ind8  11.8 119.9 0.78
ind9  14.8 125   0.25
ind10 15.5 104   0.33
```

Continuous map file

The continuous map file has one required column (ID) and an optional column which identifies the type of value.

```
BMI measurement
BP measurement
WBC measurement
EXPR1 expression
EXPR2 expression
```

Grammar File

The grammar file can be altered by the user to modify the behavior of GENN during a run. For example, the number of input connections into a node can be edited or the types of neurons can be changed (for example, to only use the PA (additive) nodes).

```
<p>       ::= <pn>(<pinput>)
<pn>      ::= PA
            | PS
            | PM
            | PD
<pinput>  ::=  <W>(<winput>)<,><W>(<winput>)<,>2
            |  <W>(<winput>)<,><W>(<winput>)<,><W>(<winput>)<,>3
            |  <W>(<winput>)<,><W>(<winput>)<,><W>(<winput>)<,><W>(<winput>)<,>4
<winput>  ::= <cop><,><v>
            | <cop><,><p>
<cop>     ::= (<cop><op><cop>)
            | <Concat>(<num>)
<Concat>  ::= Concat
<,>       ::= ,
<W>       ::= W
<op>      ::= +
            | -
            | *
            | /
<num>     ::=  .<dig><dig>3
            |  <dig>.<dig><dig>4
<dig>     ::= 0
            | 1
            | 2
            | 3
            | 4
            | 5
            | 6
            | 7
            | 8
            | 9
<v>       ::= G1-4
```

Bio Filter Model File

ATHENA accepts a list of models that can be incorporated into the initialization of networks. The first two columns list the marker IDs. The last column is the implication index which is a score designating how many sources specify the model.

```
rs50 rs40 3
rs96 rs24 3
```

```
rs82 rs51 3
rs44 rs14 2
rs89 rs5 2
rs85 rs76 2
rs71 rs10 2
```

Output files

Summary file

ATHENA produces a tab-delimited summary file listing the variables from the best model and its scores for each cross validation interval in the analysis. It also identifies the fitness used and reports how much missing data is in the models (if any). In addition it writes the networks as equations that can be used in other software. The file has the extension .athena.sum

CV	Variables	BALANCEDACC Training		Testing	Training-missing		Training-AUC	Testing-missing	Testing-AUC
1	G10 G14 G1	0.623125	0.58	0%	0.672894	0%	0.624838		
2	G10 G20 G1 G22	0.615	0.67	0%	0.645274	0%	0.703488		
3	G10 G10 G1	0.615625	0.6075	0%	0.642148	0%	0.6428		
4	G10 G1	0.616875	0.58	0%	0.650742	0%	0.627225		
5	G23 G7 G13 G1 G10 G9	0.6225	0.6	0%	0.660894	0%	0.637287		

Best model file

The best model files display the actual network produced by ATHENA. It has the extension cv<#>.<# rank in CV>.best. For example, the best model from cross validation one has the extension .cv1.1.best.

```
CV: 1
Model Rank: 1
Training result: 0.623125
Testing result: 0.58
Training-missing:       0%
Training-AUC:   0.672894
Model:
PA( W(0.52, PS( W(29.91,G10), W(2,G14),2)), W(0.84,G1),2)
```

Dot file

ATHENA produces dot-compatible files that can be converted into image files using the dot program from the Graphviz visualization project (http://www.graphviz.org/). The files have the extension .cv<#>.<# rank in

CV>.dot. For example, the best model from cross validation one has the extension .cv1.1.dot.

```
digraph G{
        graph [ dpi = 300 ];
        size="7.5,11.0";
        dir="none";
        rankdir="LR";
        orientation="landscape";
        PADD1 [shape="doublecircle" style="bold" label="PADD"];
        W1->PADD1;
        W1 [shape="circle" style="bold" label="W"];
        const1->W1;
        const1 [shape="box" style="bold" label="17.06"];
        G101->W1;
        G101 [shape="box" style="filled" label="G10"];
        W2->PADD1;
        W2 [shape="circle" style="bold" label="W"];
        const2->W2;
        const2 [shape="box" style="bold" label="268.18"];
        G11->W2;
        G11 [shape="box" style="filled" label="G1"];
}
```

Individual score files

ATHENA can produce an optional file displaying the score that each individual receives when being processed by the best evolved networks. The file has the extension .ind_scores.txt. All individuals for all cross validations appear in a single file for a run. When ID numbers are present in the data files they are identified by those numbers. Otherwise, the file identifies each individual by the line number in the original data file. The lists the predicted and observed scores for every individual for every cross-validation.

```
Training                                Testing                         Training
Testing                         Training                                Testing
Training                                Testing                         Training
Testing
CV#1-ID Pred    Pred-Status     Obs     CV#1-ID Pred    Pred-Status     Obs     CV#2-ID
Pred    Pred-Status     Obs     CV#2-ID Pred    Pred-Status     Obs     CV#3-ID Pred
Pred-Status     Obs     CV#3-ID Pred    Pred-Status     Obs     CV#4-ID Pred    Pred-
Status     Obs     CV#4-ID Pred    Pred-Status     Obs     CV#5-ID Pred    Pred-Status
Obs     CV#5-ID Pred    Pred-Status     Obs
1808    0.5     0       1       1371    0.420676        0       1       1371    1.64342e-
13      0       1       1808    0.0054863       0       1       1214    1.30881e-19     0
1       1391    0.5     0       1       1371    0       0       1       50      3.89885e-
08      0       1       1371    1.22473e-32     0       1       1545    0.000926387     0
1
```

```
1677    0.420676        0       1       1214    0.301535        0       1       1214
2.09689e-17     0       1       1677    5.09129e-12     0       1       1784    1.30881e-
19      0       1       1975    1.30881e-19     0       1       1934    0.5     0       1
575     0       0       1       1214    2.8026e-44      0       1       1821    1.69207e-
41      0       1
```

Cross-validation file

ATHENA can produce optional files listing the individuals in each cross validation's training set. The files are named .train.cv.1.txt, .train.cv.2.txt, etc. They contain the values of the genotypes that are used, so they are shown with dummy encoding if that option was set in the configuration file.

```
403 1 0 0 0 -1 1 -1 -1 -1 1 1 0 0 1 -1 -1 -1 0 -1 0 1 1 0 0 -1 0
1371 1 -1 0 1 -1 -1 0 0 -1 0 1 -1 -1 -1 0 -1 -1 -1 -1 -1 1 1 1 -1 0 -1
1934 1 0 0 0 -1 -1 -1 -1 -1 1 0 0 0 -1 -1 0 -1 -1 0 -1 0 0 -1 0 1 -1
1214 1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 0 -1 -1 0 1 -1 -1 0 -1 0 -1 -1 -1
1784 1 0 0 0 -1 -1 0 -1 -1 -1 -1 -1 0 -1 0 0 -1 0 -1 -1 -1 1 -1 -1 1 0
1693 1 0 0 -1 1 -1 0 -1 0 1 0 -1 0 -1 -1 -1 -1 -1 0 -1 0 0 1 -1 0 0
```

Split file

ATHENA can produces a file listing the individuals in each split of the data that is used for cross-validation. The file has the extension .cvsplit and writes all the splits to a single file. Each split is identified by the word split followed by an individual ID per line. This file can be used to load a specific data split into ATHENA using the SPLITFILE parameter.

```
split
403
1371
1934
1214
1784
1693
1443
1741
115
168
1922
1514
1430
1639
1703
```

Log files

ATHENA can produce a variety of log output that can be helpful in viewing the progress of the algorithms used and deciding how to adjust the parameters in the configuration file.

Overview log file

Using OVERVIEW for the LOG parameter generates an overview log file for each cross-validation of the run. The files contain information on each generation of the run. It records the number of valid neural networks, average size(in codons), average depth of the grammar used, average fitness, maximum fitness, minimum fitness, average number of genotype variables, average number of continuous variables, maximum number of epochs (if back propagation is used) and the variables in the best model for that generation.

```
Gen  ValidNN    AvgSize   AvgDepth AvgFit  MaxFit   MinFit   AvgGeno  AvgCov
AvgEpochs  MaxEpoch  Bestmod
0    150        172       1.92     0.502   0.586    0.454    8.62     0      0
0          G1_G9_G9_G10_G12_G13_G13_G13_G16_G16_G22
1    102        137       1.88     0.506   0.586    0.455    7.77     0      0
0          G1_G9_G9_G10_G12_G13_G13_G13_G16_G16_G22
2    104        97.5      1.78     0.506   0.586    0.476    6.23     0      0
0          G1_G9_G9_G10_G12_G13_G13_G13_G16_G16_G22
3    111        77.8      1.75     0.505   0.597    0.463    5.54      0   0
0          G3_G1_G8_G8_G6_G9_G11_G14_G20
```

Detailed Models log file

If the LOG parameter is set to DETAILED, then a .models.log file will be written for each cross-validation in the run. It will list the scores and write out the networks for each genome in the population for each generation. WARNING: This can result in very large log files and slow the analysis significantly if a large POSIZE and/or large number of generations are set.

```
GEN     RANK    BALANCEDACC FITNESS     GRAM_DEPTH      NN_DEPTH        NUM_G
NUM_C
0       1       0.58625 10      2       11      0       PA ( W ( ( Concat ( . 7
7 3 ) / Concat ( 8 . 3 3 ) ) , PM ( W ( Concat ( . 6 4 3 ) , G14 ) , W ( Concat
( 0 1 ) , G17 ) , W ( Concat ( 8 . 5 3 ) , G17 ) , W ( Concat ( 0 6 . 7 6 5 ) ,
G11 ) , W ( Concat ( . 2 3 3 ) , G14 ) , 5 ) ) , W ( Concat ( . 3 1 3 ) , PM
```

```
( W ( Concat ( 2 1 ) , G2 ) , W ( Concat ( 3 . 9 3 ) , G14 ) , W ( Concat ( 6 .
4 2 4 ) , G13 ) , W ( Concat ( 0 . 2 2 4 ) , G23 ) , 4 ) ) , W ( ( Concat ( . 7
0 3 ) / ( ( Concat ( 3 . 6 3 ) + Concat ( 8 1 ) ) + ( Concat ( 9 1 . 5 2 5 ) +
Concat ( 6 1 ) ) ) ) , G10 ) , W ( Concat ( 6 2 . 2 2 5 ) , G10 ) , 4 )
```

```
0       2       0.57    10      1       3       0       PA ( W ( ( ( ( Concat (
4 0 . 5 9 5 ) * Concat ( 3 1 ) ) / ( Concat ( 8 . 5 4 4 ) + Concat ( 8 4 . 4 8
5 ) ) ) / ( ( Concat ( . 5 1 3 ) - Concat ( . 7 3 3 ) ) / ( Concat ( 9 . 5 7
4 ) * Concat ( 1 1 ) ) ) ) , G6 ) , W ( ( ( ( Concat ( 9 1 ) - Concat ( 1 1 ) )
* ( Concat ( 3 1 ) / Concat ( 4 1 ) ) ) - ( ( Concat ( 8 6 . 5 6 5 ) / Concat (
. 2 3 3 ) ) * ( Concat ( 3 . 4 2 4 ) - Concat ( 5 . 1 3 ) ) ) ) , G1 ) , W
( ( ( ( Concat ( 5 . 1 0 4 ) + Concat ( . 0 4 3 ) ) - ( Concat ( 4 9 . 7 3 5 )
+ Concat ( 0 . 1 3 ) ) ) * ( ( Concat ( . 4 4 3 ) + Concat ( . 5 6 3 ) ) /
( Concat ( 4 . 2 2 4 ) / Concat ( 1 . 6 3 ) ) ) ) , G5 ) , 3 )
```

Variables Models log file

When the LOG parameter is set to VARIABLES, then a .models.log file will be written for each cross-validation in the run. In this case, the file will only report the variables that appear in each model in the run.

```
Gen: 0
G14 G17 G17 G11 G14 G2 G14 G13 G23 G10 G10
G6 G1 G5
G20 G11 G3 G10 G20 G18 G23
G14 G12 G1
G10 G4 G24
G3 G9 G13 G10 G5 G24 G17 G10 G25
G4 G19 G6 G13 G24 G19 G3 G16 G15 G5 G20 G22 G15
G6 G21 G17 G20 G14 G1 G17 G2 G1 G25 G10
G21 G7 G24 G13 G10 G17 G17 G3 G7 G8 G21 G3
G22 G10 G3 G12
```