Insight2 Pseudocode / functional overview

Details of the INSIGHT model are not provided here as they are detailed in "Inference of Natural Selection from Interspersed Genomic Elements Based on Polymorphism and Divergence", Gronau I., Arbiza L, Mohammed J, Siepel A; Molecular Biology and Evolution, Volume 30, Issue 5, 1 May 2013, Pages 1159–1171. https://doi.org/10.1093/molbev/mst019.

The INSIGHT1 code and documentation is available at http://compgen.cshl.edu/INSIGHT/

Likelihood maximization is provided by the Bounded Gradient Descent (LBFGS) algorithm, with analytic derivatives, for the INSIGHT model. The source for this is included in the Insight2 workspace. See last page.

Functional Overview:

Process user arguments (including server mode flag)

Read Database – approximately 3 GB of data
block.bedg (polymorphism and divergence databases)
monoDB (monomorphic site sufficient statistics)
poly.bedg (Low and High frequency Polymorphism sufficient statistics)
polyn.bedg (Low and High frequency Polymorphism sufficient statistics for neutral positions)

If Server mode: obtain input file name, and any argument overrides for this iteration. Input files must have suffix ".bed".

Clear models

Identify Prior Model statistics and parameters from parameters or defaults

- $\rho, \eta, \gamma, \beta_1, \beta_2, \beta_3, \lambda, \lambda_N, \theta, \theta_N$ and weight as number of observations (pseudocounts) from this model

Read sorted (.bed) input file (-fin) that provides genomic positions and ,optionally, weights,

- Aggregate sufficient statistics for positions occurring in each block defined in block.bedg,
- Weighted positions ($W \in \mathbb{I}, 0 \leq W \leq Wmax, w = \frac{W}{Wmax}$) are counted as fraction w of an observation in likelihood calculations.

If refinement of λ, θ values is requested,

- Generate expected (average) values for $\lambda, \lambda_N, \theta, \theta_N$ across input positions for use in prior model

If refinement of β values is requested,

- Directly estimate β_2.
- Use bounded gradient descent on likelihood to identify maximum likelihood values for β_1/β_3 under the constraint that $0 \leq \beta_1, \beta_2, \beta_3 \leq 1$ and $\beta_1 + \beta_2 + \beta_3 = 1$
- The values for β characterize the mean Neutral model of evolution for neutral sites (see INSIGHT) in the blocks containing sites identified by the user via the weighted input file

If refinement of ρ, η, γ values is requested,

- Use bounded gradient descent on log likelihood to identify maximum likelihood values (over for the user specified positions) for ρ, η, γ .
    - If η, γ are at the upper boundary (can happen when ρ is small), relax the upper boundary and try again until hard mas is reached. This is done because when ρ is large, large η, γ upper bounds can destabilize the LBFGS optimizer.
- Use likelihood curvature at maximum likelihood values (Inverse Hessian) to estimate uncertainty in parameters.
- Calculate supplementary statistics and uncertainties (see INSIGHT)

- Dp = rho * eta * lambda *1000
- Pw = rho * gamma * (theta - eta * lambdatheta) * 1000

If expected values or posterior distributions of ρ, η, γ values are requested,

- Generate sampling grid for ρ, η, γ at highest resolution near max likelihood values.
- Spawn multiple threads to
  - sample prior, data likelihood, and centered grid element size at each gridded value for ρ, η, γ.
- Renormalize prior, combine prior likelihood and grid element size into posterior probability distribution for each parameter.
- Sum probabilities to generate marginal distributions for each parameter, well as expectations and median values.
- Calculate credal intervals based on lowest value >= (.95/2) of the probability and highest value <= (.95/2) mass.

If INSIGHT1 input files are requested, insure β values have been estimated, and generate compatible files.

If requested, generate supplementary statistics, generally counts of allele frequencies

If requested, generate detailed allele posterior probabilities.

- This is analogous to the posterior option in INSIGHT1.

If requested, perform counterfactual analysis based on refined model. This analysis is beyond the scope of the present work, p[lease contact author Brad Gulko for more information.

Generate output files, for input file X.bed

- X.insres — model summary
- X.model — model detail
- X.dpos.alp — posterior allele probaiblities
- X.fpos.cfa — counterfactual analysis
- X.lpos — latent class posteriors (INSIGHT1 posteriors)
- X.ins and .beta.ins — INSIGHT1 input files
- X.ppost.marg — parameter marginals
- X.ppost.full — full posterior probability distribution for parameters
- X.done — semaphore indicating the completion of processing, so output is ready for use by another piece of software that might be running in parallel.

If not in server mode, terminate, otherwise read next file name

- <Filename> # <Input parameter overrides for input file>
  - E.g. 0017.bed # -rho 0.06,1,0.05
- Server mode exists to prevent the need to reload the database for each input file, when a collection of input files are to be processed. Database input can require the loading of > 2 GB of data.
- Server mode terminates when the input line “done” is read from stdin

Bounded Gradient Descent (LBFGS)

- Provided by LBFGS V2.1, converted from FORTRAN to C via f2c 2006 version
    - Ciyou Zhu, Richard Byrd, Jorge Nocedal and Jose Luis Morales.
    - See: C. Zhu, R. H. Byrd and J. Nocedal. L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization (1997), ACM Transactions on Mathematical Software, Vol 23, Num. 4, pp. 550 - 560.
- "Cleaned up" by Allin Cottrell, and available at
    - http://www.ece.northwestern.edu/~nocedal/lbfgsb.html .
    - http://users.iems.northwestern.edu/~nocedal/Software/Lbfgsb.2.1.tar.gz
- Wrapped into a C++ object by Brad Gulko 2015

Generally, calls to this are wrapped in a loop attempting to find a solution with a tight tolerance, and loosening the tolerance if the optimization fails. Optimization fails only if all loop iterations fail

Iteration order

- // factr, tolerance relative to machine precision, pgtol minimal gradient.
- case 0: opt.factr = 100.0; opt.pgtol = 0.0; // 100x machine precision (13 sig digs), no slope limit, near perfect
- case 1: opt.factr = 1.0e4; ; opt.pgtol = 0.0;// 1000x machine precision (11 sig digs), no slope limit, great
- case 2: opt.factr = 1.0e6; opt.pgtol = 1e-12; // 1e6 * machine precision, 9 sig digits, very good
- case 3: opt.factr = 1.0e8; opt.pgtol = 1e-10; // 1e8 * MP ~7 significant digits... serviceable
- case 4: opt.factr = 1.0e10; opt.pgtol = 1e-08; // 1e10 * ~5 significant digits barely adequate