Supplemental Methods for LD filtering

Contents

1 LD-based filtering of SNP-pairs **1**
1.1 Scaling of binned r^2 distributions 2
1.2 Bayesian parameter estimation using the binned r^2 distributions 2

2 Code availability **2**

Bibliography **3**

1 LD-based filtering of SNP-pairs

The LD values of SNP-pairs were compared after binning SNP-pairs by base pair separation to control for unknown rates of recombination in *G. f. fuscipes*. Bin length was set at 50 bp with the lower bound inclusive and higher bound exclusive. In other words the bins were defined as $[i, i + 49)$ where $i \in \{1, 1(50), 2(50), 3(50)...n(50)\}$ and n is a positive integer.

To identify SNP-pairs for further investigation in an arbitrary set of binned SNP-pairs, we needed to assign probabilities to each SNP-pair in the bin. The distributions of binned SNP-pairs are bounded by 0 and 1 and do not appear to be Normal in shape. Additionally, in many cases, the data appear to exhibit peaks at both the lower and upper r^2 range. This suggested that the data may be modeled well using the probability density function (PDF) of the Beta distribution:

$$f(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha, \beta)}$$

with shape parameters α and β and where x represents an observed value generated by the distribution and $\mathrm{B}()$ is the Beta function.

The Beta's cumulative distribution function (CDF):

$$F(x; \alpha, \beta) = \frac{\mathrm{B}(x; \alpha, \beta)}{\mathrm{B}(\alpha, \beta)}$$

can be used to describe the probability that a binned SNP-pair will be observed to have an $r^2 \leq x$. It follows that $1 - \mathrm{CDF}$ represents the probability of observing a more extreme value.

For each set of binned r^2 values, the SNP-pairs deemed worthy of further investigation were defined as those where $1 - \mathrm{CDF} \leq 0.01$ after Benjamini-Hochberg (BH) correction for multiple testing [1].

1.1 Scaling of binned r^2 distributions

The Beta distribution is bounded on the non-inclusive interval between 0 and 1. However, there are data in each bin that may have been assigned values of exactly 0 or 1. It is likely that these values are not truly 0 or 1 in the discrete binary sense that a coin-flip is *either* heads or tails. Therefore, the all data for each bin were scaled according to the following scheme

$$((x_i - 0.5) \cdot \theta) + 0.5$$

in order to symmetrically shrink the distribution of values to fit within the open interval $(0, 1)$. In the scheme above, let x_i stand for the r^2 of each SNP-pair in an arbitrary bin set and θ stand for the scaling factor. The relevant results in this paper used $\theta = 0.999$.

1.2 Bayesian parameter estimation using the binned r^2 distributions

In order to use the CDF of the Beta distribution to assign significances to each observed SNP-pair in a bin, it is necessary to learn the distribution parameters (α and β) for each bin given the specific data in each bin. For this we used custom python code that made heavy use of the following third-party data analysis modules: Pandas [2], NumPy [3], SciPy [4], pyMC [5], and StatsModels [6].

We used pyMC to build the model of the Beta distribution and use it to exploit the bin-specific data to estimate the bin-specific values for the α and β parameters of the model [Figure fig_betamod]. The values of α and β were modeled with separate Uniform prior distributions from 0.01 to 10. This model topology was used to create pyMC "model" objects and initialized with the r^2 data from each bin. The parameters of each Beta distribution were then estimated by *maximum a posteriori* (MAP) and used to calculate the $1 - \mathrm{CDF}$ for each SNP-pair in each bin [3,4]. The $1 - \mathrm{CDF}$ values were then BH corrected by bin and filtered with a threshold of $(1 - CDF)_{BH} \leq 0.01$ using statsmodels [1].

2 Code availability

The code used to run this and other parts of the analyses described in this manuscript may be found on github at the following project address.

https://github.com/CacconeLabYale/gloria_soria_ddRAD_2015

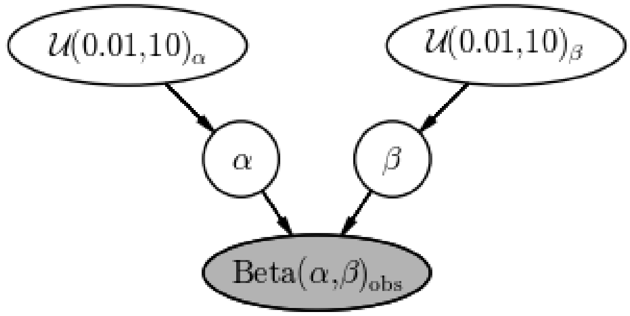

Figure 1: **[fig_betamod]Network representation of the LD Beta model**: Ovals represent modeled probability distributions. Circles represent learned parameters. Grey shading indicates use of observed data.

Bibliography

1. Benjamini Y, Hochberg Y. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society Series B (Methodological). Wiley for the Royal Statistical Society; 1995;57: pp. 289–300. Available: http://www.jstor.org/stable/2346101

2. McKinney W. Data Structures for Statistical Computing in Python. In: Walt S van der, Millman J, editors. Proceedings of the 9th python in science conference. 2010. pp. 51–56.

3. Van Der Walt S, Colbert SC, Varoquaux G. The NumPy array: A structure for efficient numerical computation. Computing in Science and Engineering. 2011;13: 22–30. doi:10.1109/MCSE.2011.37

4. Jones E, Oliphant T, Peterson P, Others. SciPy: Open source scientific tools for Python [Internet]. 2001. Available: http://www.scipy.org/

5. Patil A, Huard D, Fonnesbeck CJ. PyMC: Bayesian Stochastic Modelling in Python. Journal of statistical software. 2010;35: 1–81. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3097064/&tool=pmcentrez/&rendertype=abstract

6. Statsmodels-development-team. StatsModels: Statistics in Python (v0.6.1) [Internet]. Available: http://statsmodels.sourceforge.net/stable/