Version dated: September 14, 2017

RH: HYBRIDIZATION DETECTION WITH HYDE

HyDe: a Python Package for Genome-Scale Hybridization Detection

PAUL D. BLISCHAK[1,*], JULIA CHIFMAN[3], ANDREA D. WOLFE[1],
AND LAURA S. KUBATKO[1,2]

[1]*Department of Evolution, Ecology, and Organismal Biology and*
[2]*Department of Statistics, The Ohio State University, Columbus, OH, 43210, USA;*
[3]*Department of Mathematics and Statistics, American University, Washington, DC, 20016, USA*

Corresponding author: Paul Blischak, Department of Evolution, Ecology, and Organismal Biology, The Ohio State University, 318 W. 12th Avenue, Columbus, OH, 43210, USA; E-mail: blischak.4@osu.edu.

Abstract.—The analysis of hybridization and gene flow among closely related taxa is a common goal for researchers studying speciation and phylogeography in natural populations. Many methods for hybridization detection use simple site pattern frequencies from observed genomic data and compare them to null models that predict an absence of gene flow. The theory underlying the detection of hybridization using these site pattern probabilities exploits the relationship between the coalescent process for gene trees within population trees and the process of mutation along the branches of

the gene trees. For certain models, site patterns are predicted to occur in equal frequency (i.e., their difference is 0), producing a set of functions called *phylogenetic invariants*. In this paper we introduce HyDe, a software package for detecting hybridization using phylogenetic invariants arising under the coalescent model with hybridization. HyDe is written using a combination of C++ and Python, and can be used interactively in Python or through the command line using pre-packaged scripts. We demonstrate the use of HyDe on simulated data, as well as on two empirical data sets from the literature. We focus in particular on identifying individual hybrids within population samples and on distinguishing between hybrid speciation and gene flow. HyDe is freely available as an open source Python package under the GNU GPL v3 on both GitHub (https://github.com/pblischak/HyDe) and the Python Package Index (PyPI: https://pypi.python.org/pypi/phyde).
(Keywords: ABBA-BABA, coalescence, gene flow, hybridization, phylogenetic invariants)

It is increasingly recognized that a strictly bifurcating model of population descent is inadequate to describe the evolutionary history of many species. Hybridization and gene flow are processes that obscure this simple model, and methods to disentangle the signal of admixture from other sources of incongruence (e.g., incomplete lineage sorting [ILS]) are receiving increased attention (Mallet 2005, 2007; Abbott et al. 2013). Early methods for detecting hybridization in the presence of ILS used estimated gene trees to detect deviations from the coalescent model under the

expectation of a bifurcating species tree (Joly et al. 2009; Kubatko 2009; Meng and Kubatko 2009; Gerard et al. 2011). This work was later extended to include searches over network space to infer species networks with reticulate edges (Yu et al. 2011, 2012, 2013, 2014; Solís-Lumis and Ané 2016). Other approaches for detecting hybridization use genome-wide SNP data to test for admixture on rooted, four- or five-taxon trees. ("ABBA-BABA"-like methods; Green et al. 2010; Durand et al. 2011; Patterson et al. 2012; Martin et al. 2014; Eaton and Ree 2013; Pease and Hahn 2015). A common feature of these genome-wide methods for hybridization detection is their use of site patterns to test for deviations from the expected frequency under a neutral coalescent model with no introgression (Green et al. 2010; Durand et al. 2011).

The theoretical underpinnings of these site pattern-based inference methods originate from the study of invariants: functions of phylogenetic model parameters (often in the form of site pattern probabilities) whose expected difference is always 0. The earliest use of invariants to infer phylogenies was a pair of papers from Cavender and Felsenstein (1987) and Lake (1987), who separately derived functions to determine the correct topology for an unrooted quartet using binary and nucleotide data, respectively. Recent applications of phylogenetic invariants include proving identifiability (Chifman and Kubatko 2015), coalescent-based species tree inference (SVDquartets; Chifman and Kubatko 2014), sliding-window analyses of phylogenetic bipartitions (SplitSup; Allman et al. 2016), and the detection of hybridization (Green et al. 2010; Durand et al. 2011; Kubatko and Chifman 2015). The increased interest in methods using invariants is concomitant with the ability to collect genomic sequence data, allowing for accurate estimates of site pattern frequencies. Furthermore, because these methods are based on site pattern frequencies, they offer computationally tractable approaches for analyzing genome-scale data sets.

In this paper we introduce HyDe, a Python package for the detection of

hybridization using phylogenetic invariants. HyDe provides methods to detect hybridization at both the population and individual levels, with additional features to discover individual hybrids within populations. The programming philosophy behind HyDe was to provide a low-level library of data structures and methods for computing site pattern probabilities and conducting hypothesis tests, along with a core set of Python scripts that use this library to automatically parse and analyze genomic data. We describe each of these interfaces in detail below and demonstrate the use of HyDe on both simulated and empirical data sets. We have also provided all of the code for processing, plotting, and interpreting the results of these analyses to facilitate analyses with HyDe for other researchers.

Description

Model Background

In this section we provide a brief overview of the model used by HyDe for detecting hybridization. A more thorough discussion of the theory behind the model and the derivation of the test statistics used by HyDe can be found in Kubatko and Chifman (2015).

Consider a rooted, four-taxon network consisting of an outgroup and a triple of ingroup populations: two parental populations (P1 and P2) and a hybrid population (Hyb) that is a mixture of P1 and P2 [Figure 1]. Under this model, gene trees arise within the parental population trees following the coalescent process (Kingman 1982), where the hybrid population is either (1) sister to P1 with probability $1 - \gamma$ or (2) sister to P2 with probability γ (Meng and Kubatko 2009). Mutations are then placed on these gene trees according to the standard Markov models of nucleotide substitution [e.g., JC69

(Jukes and Cantor 1969), HKY85 (Hasegawa et al. 1985), GTR (Tavaré 1986)]. Given one sampled individual within each population, we can describe a probability distribution on the possible site patterns observed at the tips of the gene trees: $p_{ijkl} = P(\mathrm{O} = i, \mathrm{P1} = j, \mathrm{Hyb} = k, \mathrm{P2} = l)$, with $i, j, k, l \in \{A, G, C, T\}$. Using nucleotide sequence data, we can also estimate these probabilities using the observed site patterns to get $\hat{p}_{ijkl} = \frac{X_{ijkl}}{S}$, where S is the total number of observed sites and X_{ijkl} is the number of times that site pattern $ijkl$ was observed.

These site pattern probabilities form the basis of the test for hybridization that is implemented in HyDe, as well as several other methods. For example, the D-statistic (Patterson et al. 2012) uses the site patterns p_{ijji} and p_{ijij} to test for hybridization assuming a null model of no admixture:

$$\mathbf{D} = \frac{p_{ijji} - p_{ijij}}{p_{ijji} + p_{ijij}} \,. \tag{1}$$

In this case, the expected value of $\mathbf{D}$ is 0, indicating that the p_{ijji} and p_{ijij} are expected to occur in equal frequency (i.e., they are invariant). Because these site patterns contain information about the topology of the underlying population tree, they are called phylogenetic invariants (Cavender and Felsenstein 1987; Lake 1987).

In a similar way, Kubatko and Chifman (2015) presented several different invariants-based test statistics that detect hybridization using linear phylogenetic invariants. Among the invariants that they tested, the ratio of $f_1 = p_{iijj} - p_{ijij}$ and $f_2 = p_{ijji} - p_{ijij}$ provided the most statistical power and is used to form the test statistic that is used by HyDe to detect hybridization. When the model holds, it can be shown that

$$\frac{f_1}{f_2} = \frac{p_{iijj} - p_{ijij}}{p_{ijji} - p_{ijij}} = \frac{\gamma}{1 - \gamma} \,. \tag{2}$$

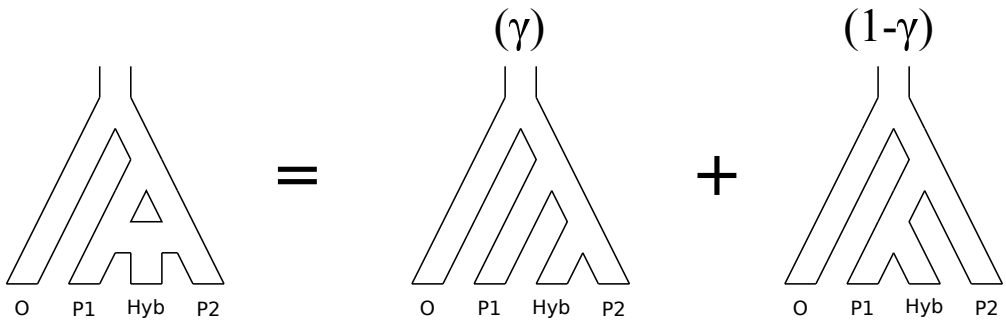

Figure 1: Illustration of the model for detecting hybridization using HyDe. The hybrid population (Hyb) is modeled as a mixture between two parental populations, P1 (1-γ) and P2 (γ).

Derivation of the observed frequencies from this expectation is used to form a test statistic that asymptotically follows a standard Normal distribution, allowing formal hypothesis tests to be conducted ($H_0 : \gamma = 0$ vs. $H_1 : \gamma > 0$). Furthermore, because the ratio of f_1 and f_2 is a function of γ, we can estimate the amount of admixture directly from the observed site pattern frequencies:

$$\hat{\gamma} = \frac{\hat{f}_1}{\hat{f}_1 + \hat{f}_2}\,. \tag{3}$$

New Additions

The software we present here is a complete rewrite of the original program used by Kubatko and Chifman (2015) and contains several added features that we outline below.

Multiple individuals per population.—We have modified the calculation of the site pattern probabilities to accommodate multiple individuals per population, rather than testing each sampled individual separately. We do this by considering all permutations of the individuals in the four populations involved in the hypothesis test. For example, if there are N_{OUT} individuals in the outgroup, N_{P1} individuals in parental population one, N_{HYB} individuals in the putative hybrid population, and N_{P2} individuals in parental

population two, then a total of $N_{OUT} \times N_{P1} \times N_{HYB} \times N_{P2}$ quartets are used to calculate the site patterns for the hypothesis test. Including multiple individuals per population increases the sample size used to calculate the site pattern probabilities and can therefore lead to more accurate detection of hybridization.

Identifying individual hybrids.—An underlying assumption of the coalescent with hybridization is that all the individuals in the hybrid population are admixed (i.e., hybrid speciation; Meng and Kubatko 2009). However, when gene flow, rather than hybrid speciation, is responsible for the introgression of genetic material into the admixed population, it is possible that not all individuals will contain these introgressed alleles. When hybridization detection is conducted on populations with a mix of hybrids and non-hybrids, it is common for invariant-based tests to report significant test statistics, even though it is a violation of the model (see simulations in the **Benchmarks** section below). To help with the detection of non-uniform introgression into the hybrid population, we have included two methods in the new version of HyDe that aim to detect variation in the amount of hybridization in the individuals sampled.

If not all of the individuals in a putative hybrid population are admixed, bootstrap resampling can reveal heterogeneity in the process of introgression when more or fewer hybrid individuals are included in each replicate. For example, if we are testing four diploid individuals in a putative hybrid and only two of them are 50:50 hybrids (γ = 0.5 for each), then the value of gamma for the whole population will be $\gamma = 0.25$. When we resample individuals with replacement and recalculate γ, we will get different values depending on how many times the hybrids are resampled (0 times: $\gamma = 0.0$; 1 time: $\gamma = 0.125$; 2 times: $\gamma = 0.25$; 3 times: $\gamma = 0.375$; 4 times: $\gamma = 0.5$). Because the process of hybridization is not uniform, the value of γ jumps between different values. This can also be visualized by plotting the distribution of γ across the bootstrap

replicates (e.g., Figure 2a).

We have also implemented methods that test all individuals in a putative hybrid separately while treating the parents and outgroup as populations. This approach allows hypothesis tests and estimates of γ to be calculated for every individual to see if it is a hybrid. A caveat with testing each individual is that the number of sites sampled must be enough to have statistical power to detect that hybridization has occurred (Kubatko and Chifman 2015).

Ambiguity codes and missing data.—In the previous version of HyDe, any sites containing ambiguity codes or missing data were ignored when tabulating the site pattern probabilities for a hypothesis test. To allow more of the data to be used, and to account for the potentially large amounts of missing data that are common in high throughput sequencing data sets, we implemented approaches that instead integrate over these nucleotides by considering the possible resolutions of the observed bases into site patterns. As an example, consider the site pattern AGRG. There are two possible resolutions for the ambiguity code R: AGGG and AGAG. To account for this, we add 0.5 to the site pattern counts X_{ijjj} and X_{ijij}. In general, for any site pattern containing ambiguous or missing bases (but not gaps), we find all possible resolutions and add one divided by the total number of resolutions to the corresponding site pattern counts.

Software Design and Implementation

HyDe is implemented using a combination of Python and C++. It also uses Cython, a superset of Python that allows C/C++ capabilities to be incorporated into Python for better efficiency (Behnel et al. 2010). Installing HyDe requires several external Python modules [`numpy`, `scipy`, `pandas`, `cython`, `matplotlib`, and `seaborn`] as well as a C++ compiler that is compliant with the C++11 standard. The goal behind our

implementation of HyDe was to provide both pre-packaged scripts to conduct standard hybridization detection analyses, as well as a library of functions and data structures that researchers could use to customize their analyses. Below we describe the functionality of both of these interfaces, as well as the input files required to run HyDe.

Input files.—The input files for running HyDe are plain text files containing the DNA sequence data and the map of individuals to populations. The DNA sequence data are in sequential Phylip format with the header information removed (i.e., no line for the number of individuals and number of sites). The population map is a two-column table with individual names in the first column and the name of the population that it belongs to in the second column. The third input file that is required for individual-level testing and bootstrapping (optional for running a normal HyDe analysis) is a three column table of triples. A hypothesis test is then run for each triple using the first column as parent one, the second column as the putative hybrid, and the third column as parent two. The outgroup for each test is always the same and is specified separately at the command line. Output files from previous HyDe analyses can also be used to specify which triples are to be tested.

Command line interface.—The command line interface for HyDe consists of a set of Python scripts that can be used to detect hybridization, filter results, conduct individual bootstrapping, and test for hybridization at the individual level. A C++ version of HyDe, called `hyde_cpp`, is also included as part of the software package and can be used either directly for hybridization detection or through the use of the other Python scripts. The three main scripts for detecting hybridization with HyDe provide the majority of the basic functionality that is needed to detect hybridization in an empirical data set and to assess if individuals in putative hybrids are all equally admixed.

- *run_hyde.py*: The *run_hyde.py* script detects hybridization by running a hypothesis

test on all possible triples in a data set in all directions (i.e., a "full" HyDe analysis). It does this by wrapping a call to the C++ version of HyDe (`hyde_cpp`). There is also an option to supply a list of specific triples (three column text file) to test. The script outputs two files: one with all of the hypothesis tests that were conducted and one with only those hypothesis tests that were significant and had estimates of γ between 0 and 1.

- *individual_hyde.py*: The *individual_hyde.py* script runs separate hybridization detection analyses on each individual within a putative hybrid lineage. The only additional input needed for the script is a three column list of triples or a filtered results file from the *run_hyde.py* script.

- *bootstrap_hyde.py*: The *bootstrap_hyde.py* script performs bootstrap resampling of the individuals within a putative hybrid lineage and conducts a hypothesis test for each bootstrap replicate. Similar to the *individual_hyde.py* script, the script uses as input a table of triples or a filtered results file from a previous hybridization detection analysis.

The workflow that we envision for these scripts starts with using the *run_hyde.py* script to test for hybridization on all triples in all possible directions, and to filter out only those triples that have significant evidence for hybridization. Then, using the filtered output file, users can either test all of the individuals in each hybrid lineage or perform bootstrap resampling of the individuals using the *individual_hyde.py* and *bootstrap_hyde.py* scripts, respectively. Users with specific hypotheses that they want to test can simply create a text file listing the triples of interest to be run through any of the three scripts.

Python API.—Each of the command line scripts described above makes use of an

underlying Python library with built-in data structures for processing the hypothesis tests and bootstrapping analyses conducted by HyDe. This library is exposed through the Python module `phyde` (**P**ythonic **Hy**bridization **De**tection). The main data structures that are part of this module are listed below:

- *HydeData*: The *HydeData* class is the primary data structure for reading in and storing DNA sequence data and maps assigning individuals to populations. This class also implements three of the main methods for detecting hybridization at the population level [`test_triple()`] and the individual level [`test_individuals()`], as well as bootstrapping individuals within populations [`bootstrap_triple()`] (see Box 1).

- *HydeResult*: The *HydeResult* class parses the results file output by a hybridization detection analysis and stores the results as a Python dictionary (key-value pair). Values stored by the *HydeResult* class can be accessed by providing the name of the desired value and the names of the taxa in the triple of interest as arguments to the variable used to when reading in the results. For example, if we read the results of a hybridization detection analysis into a variable named `results`, we would access the estimated value of $\hat{\gamma}$ for the triple (`sp1`, `sp2`, `sp3`) using the following code: `results("Gamma", "sp1", "sp2", "sp3")`.

- *Bootstrap*: The *Bootstrap* class is similar to the *HydeResult* class except that it has built-in methods for parsing the format of the bootstrap output file written by the *bootstrap_hyde.py* script. Each bootstrap replicate is printed with a single line containing four pound symbols separating each tested triple (`####\n`). These results are parsed into a Python dictionary that can be used to access the bootstrap replicates for particular triples using the variable name, just like the *HydeResult* class [e.g., `bootstraps("Gamma", "sp1", "sp2", "sp3")`].

- *phyde.viz*: The *viz* submodule uses the `matplotlib` and `seaborn` libraries to implement basic functions for plotting the distribution of bootstrap replicates for any quantity calculated by the *bootstrap_hyde.py* script (Z-score, p-value, γ, etc.) for a specified triple. It does this by taking the name of a *Bootstrap* object variable, the name of the value to be plotted, and the names of the taxa in the triple of interest (Supplemental Materials: HyDe_SysBio.ipynb).

BENCHMARKS

All code used to complete the simulations and example analyses are available as a Jupyter Notebook (HyDe_SysBio.ipynb) on Dryad (dryad.####) along with all output files needed to reproduce the tables and figures.

Non-uniform Hybridization Simulations

To demonstrate the use of the methods in HyDe that are designed to identify individual hybrids, we set up a simulated toy example for four taxa (`out`, `sp1`, `sp2`, `sp3`) that intentionally violated the assumption of hybrid speciation by including a single hybrid individual in the population being tested for admixture (`sp2`). We simulated gene trees using the program `ms` (Hudson 2002) with population branch lengths of 1.0 coalescent unit assuming a model of coalescent independent sites (i.e., one gene tree per site; Kubatko and Chifman 2015) for 100 000, 250 000, and 500 000 sites. The parental populations were simulated with five individuals each and the outgroup had only one individual. The "admixed" population (`sp2`) had four non-admixed individuals that were most closely related to parental population `sp1`, and a single hybrid individual that was a 50:50 ($\gamma = 0.5$) mix between `sp1` and `sp3`. A single DNA base was simulated on

each gene tree for the different number of sites using the program `Seq-Gen` (Rambaut and Grassly 1997). We rescaled gene tree branch lengths from coalescent units to mutation units using a population scaled mutation rate ($\theta = 4N_0\mu$) of 0.1 per site and used a GTR+Gamma model of nucleotide substitution with three rate categories:

```
seq-gen -mGTR -s 0.1 -l 1 -r 1.0 0.2 10.0 0.75 3.2 1.6 \
        -f 0.15 0.35 0.15 0.35 -i 0.2 -a 5.0 -g 3
```

Output from `Seq-Gen` was formatted for input to HyDe using a Python script (*seqgen2hyde.py*; available on Dryad). Hybridization detection was completed for the simulated data sets with HyDe v0.3.1 using the *run_hyde.py*, *bootstrap_hyde.py* (500 bootstrap replicates), and *individual_hyde.py* scripts. Output files were processed and plotted in Python v2.7.13 using the `phyde`, `matplotlib`, and `seaborn` modules.

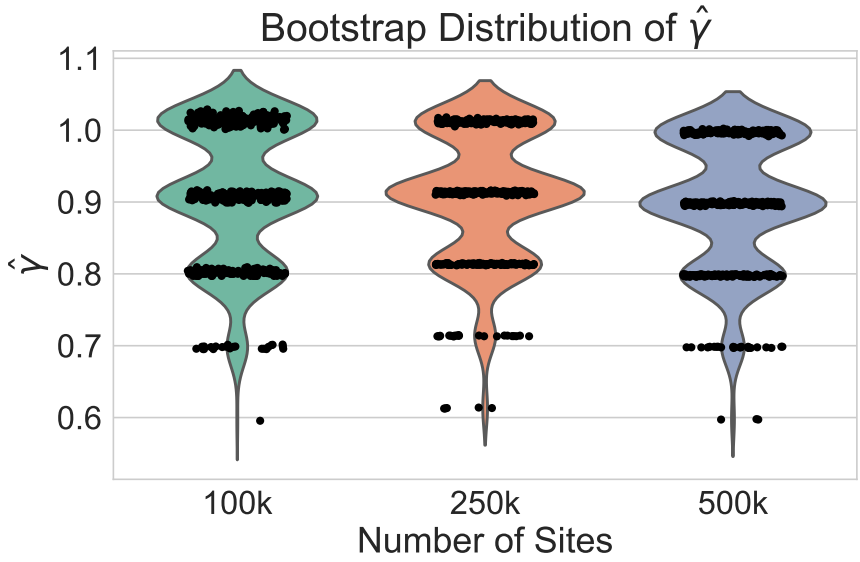

Figure 2: Violin plots of the distribution of $\hat{\gamma}$ across 500 bootstrap replicates for 100 000, 250 000, and 500 000 simulated sites. The black dots in each violin plot represent the actual values of $\hat{\gamma}$ that were estimated by each replicate (black dots are jittered). The pattern of jumping between distinct values of γ with no estimates in between are a strong indication of non-uniform admixture.

Results.—Testing for hybridization at the population level using the *run_hyde.py* script

produced a test statistic indicating that there was significant admixture in the `sp2` population regardless of the number of sites simulated (Table 1). The estimated values of $\hat{\gamma}$ for these tests were close to 0.9, which is inconsistent with the data that we simulated. The value of γ that we would expect is 0.1. However, because this is an explicit violation of the coalescent model with hybridization, the formula for estimation of γ is no longer valid.

If we didn't already know that not all of the individuals were hybrids, we would mistakenly infer that the `sp2` population has $\sim$90% of its genetic material introgressed from population `sp3`. However, when we test each individual using the *individual_hyde.py* script, we correctly infer that only one of the individuals is admixed (individual 'i10', $\hat{\gamma} \approx 0.5$) and that the rest of the individuals are not hybrids. Bootstrap resampling of the individuals in population `sp2` also indicates that hybridization is not uniform. Figure 2 shows the distribution of $\hat{\gamma}$ across all 500 bootstrap replicates and demonstrates that the value of $\hat{\gamma}$ jumps between different values depending on the number of times the hybrid individual is resampled.

Biological Examples

Heliconius *Butterflies*

DNA sequence data from Martin et al. (2013) was downloaded for four populations of *Heliconius* butterflies (248 822 400 sites; Dryad: 10.5061/dryad.dk712). The number of individuals per population was as follows: four individuals of *H. melpomene rosina*, four individuals of *H. melpomene timareta*, four individuals of *H. cydno*, and one individual of *H. hecale*. We tested the hypothesis that *H. cydno* is a hybrid between *H. melpomene rosina* and *H. melpomene timareta* with *H. hecale* as an outgroup

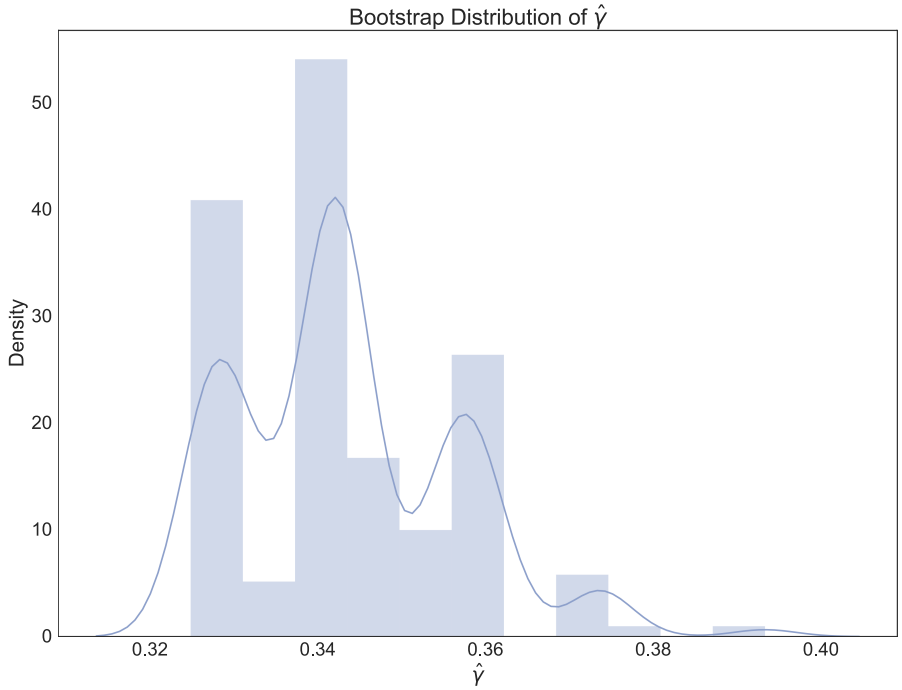

Figure 3: Density plot of the estimated values of $\hat{\gamma}$ across 500 bootstrap replicates testing for hybridization in *Heliconius cydno* between *H. melpomene rosina* and *H. melpomene timareta*.

using the *run_hyde.py* script. We then conducted bootstrapping (500 replicates) of the *H. cydno* individuals, as well as testing each individual separately using the *bootstrap_hyde.py* and *individual_hyde.py* scripts, respectively.

Results.—Significant hybridization was detected in *H. cydno* at the population level (Z-score = 493.757, p-value = $\sim$0.0, $\hat{\gamma} = 0.342$). Each individual also showed significant levels of hybridization, with $\hat{\gamma}$ ranging from 0.324 to 0.393, indicating that hybridization in *H. cydno* is mostly uniform across the individuals sampled. Figure 3 shows a density plot for the estimated values of $\hat{\gamma}$ across the 500 bootstrap replicates. Although the distribution does exhibit some amount of jumping between different values they are still between the lower and upper bounds for the individual level estimates of $\hat{\gamma}$, providing corroborating evidence that all the individuals in the *H. cydno* population are admixed.

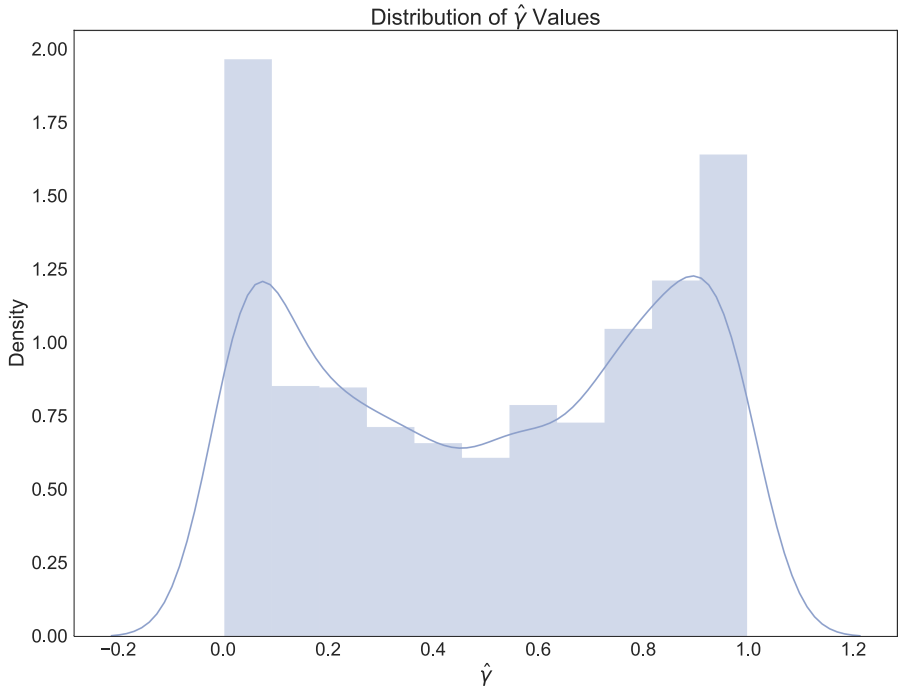

Figure 4: Density plot of the estimated values of $\hat{\gamma}$ across the 2218 hypothesis tests that detected significant levels of hybridization in *Xiphophorous*.

Swordtail Fish (Xiphophorous)

Transcriptome data from Cui et al. (2013) was obtained from the authors for 26 species of swordtail fish (25 species of *Xiphophorous* + one outgroup from the genus *Priapella*). The data were concatenated (3 706 285 sites) and put in sequential format for analysis with HyDe. Each taxon was analyzed at the individual level, resulting in $\binom{26}{3} \times 3 = 7800$ hypothesis tests. Because there were not multiple individuals per taxon, we only analyzed these data using the *run_hyde.py* script.

Results.—Out of 7800 tests, 4600 produced values of $\frac{f_1}{f_2}$ that could be used in a hypothesis test. The other values were automatically filtered out by HyDe because the values of f_1 and f_2 were too close to 0. Of these tests, 2218 reported significant levels of hybridization (Bonferroni corrected p-value of $\frac{0.05}{4600} = 1.087 \times 10^{-5}$). The distribution of the $\hat{\gamma}$ values

for these 2218 significant tests is plotted in Figure 4. This plot shows that most of the admixture occurring in *Xiphophorous* is at low levels ($\hat{\gamma}$ close to either 0.0 or 1.0). However, there are many instances of higher levels of admixture, corroborating the findings of Cui et al. (2013) about the prevalence of hybridization in this group.

Conclusions

Hybridization and gene flow are increasingly recognized as important forces in the evolution of groups across the Tree of Life (Arnold and Kunte 2017). Gaining an understanding of when these processes are occurring and how they may vary among individuals within a population are important steps to understand the spatial distribution of shared genetic variation among diverging lineages. The methods we have implemented in HyDe to detect hybridization at the population and individual level provide a set of computationally efficient methods for researchers to assess patterns of admixture in natural populations. As access to genomic data continues to grow, we anticipate that methods such as HyDe that use phylogenetic invariants will play an important role for phylogenomic inferences in non-model species.

Availability

HyDe is available as an open source Python package distributed under the GNU General Public License v3 on both GitHub (https://github.com/pblischak/HyDe) and the Python Package Index (PyPI: https://pypi.python.org/pypi/phyde). Documentation for HyDe is available on ReadTheDocs (http://hybridization-detection.readthedocs.io).

Supplemental Materials

Simulated data sets, code, and other materials will be made available on the Dryad Digital Repository.

Funding

This work was supported in part by grants from the National Science Foundation under the awards DMS-1106706 (JC and LSK) and DEB-1455399 (ADW and LSK), and National Institutes of Health Cancer Biology Training Grant T32-CA079448 at Wake Forest School of Medicine (JC).

Acknowledgements

The authors would like to thank Cécile Ané and Frank Burbink for the invitation to be a part of this special issue on the impact of gene flow and reticulation in phylogenetics. We would also like to thank R. Cui, C. Solís-Lemus, and C. Ané for help with the *Xiphophorous* data set.

*

References

Abbott, R., D. Albach, S. Ansell, J. W. Arntzen, S. J. E. Baird, N. Bierne, J. Boughman, A. Brelsford, C. A. Buerkle, R. Buggs, R. K. Butlin, U. Dieckmann, F. Eroukhmanoff, A. Grill, S. H. Cahan, J. S. Hermansen, G. Hewitt, A. G. Hudson, C. Jiggins, J. Jones, B. Keller, T. Marczewski, J. Mallet, P. Martinez-Rodriguez, M. Möst, S. Mullen, R. Nichols, A. W. Nolte, C. Parisod, K. Pfennig, A. M. Rice, M. G. Ritchie, B. Seifert, C. M. Smadja, R. Stelkens, J. M. Szymura, R. Väinölä, J. B. W. Wolf, and D. Zinner. 2013. Hybridization and speciation. Journal of Evolutionary Biology 26:229–246.

Allman, E. S., L. S. Kubatko, and J. A. Rhodes. 2016. Split scores: a tool to quantify phylogenetic signal in genome-scale data. Systematic Biology 66:620–636.

Arnold, M. J. and K. Kunte. 2017. Adaptive genetic exchange: a tangled history of admixture and evolutionary innovation. Trends in Ecology and Evolution 32:601–611.

Behnel, S., R. Bradshaw, C. Citro, L. Dalcin, D. Seljebotn, and K. Smith. 2010. Cython: the best of both worlds. Computing in Science Engineering 13:31–39.

Cavender, J. A. and J. Felsenstein. 1987. Invariants of phylogenies in a simple case with discrete states. Journal of Classification 4:57–71.

Chifman, J. and L. S. Kubatko. 2014. Quartet inference from SNP data under the coalescent model. Bioinformatics 30:3317–3324.

Chifman, J. and L. S. Kubatko. 2015. Identifiability of the unrooted species tree topology under the coalescent model with time-reversible substitution processes, site-specific rate variation, and invariable sites. Journal of Theoretical Biology 374:35–47.

Cui, R., M. Schumer, K. Kruesi, R. Walter, P. Andolfatto, and G. G. Rosenthal. 2013. Phylogenomics reveals extensive reticulate evolution in *Xiphophorus* fishes. Evolution 67:2166–2179.

Durand, E. Y., N. Patterson, D. Reich, and M. Slatkin. 2011. Testing for ancient admixture between closely related populations. Molecular Biology and Evolution 28:2239–2252.

Eaton, D. A. R. and R. H. Ree. 2013. Inferring phylogeny and introgression using RADseq data: an example from the flowering plants (*Pedicularis*: Orobanchaceae). Systematic Biology 62:689–706.

Gerard, D., H. L. Gibbs, and L. S. Kubatko. 2011. Estimating hybridization in the presence of coalescence using phylogenetic intraspecific sampling. BMC Evolutionary Biology 11:291.

Green, R. E., J. Krause, A. W. Briggs, T. Maricic, U. Stenzel, M. Kircher, N. Patterson, H. Li, W. Zhai, M. H.-Y. Fritz, N. F. Hansen, E. Y. Durand, A.-S. Malaspinas, J. D. Jensen, T. Marques-Bonet, C. Alkan, K. Prüfer, M. Meyer, H. A. Burbano, J. M. Good, R. Schultz, A. Aximu-Petri, A. Butthof, B. Höber, B. Höffner, M. Siegemund, A. Weihmann, C. Nusbaum, E. S. Lander, C. Russ, N. Novod, J. Affourtit, M. Egholm, C. Verna, P. Rudan, D. Brajkovic, Ž. Kucan, I. Gušic, V. B. Doronichev, L. V. Golovanova, C. Lalueza-Fox, M. de la Rasilla, J. Fortea, A. Rosas, R. W. Schmitz, P. L. F. Johnson, E. E. Eichler, D. Falush, E. Birney, J. C. Mullikin, M. Slatkin, R. Nielsen, J. Kelso, M. Lachmann, D. Reich, and S. Pääbo. 2010. A draft sequence of the Neandertal genome. Science 328:710–722.

Hasegawa, M., H. Kishino, and T. Yano. 1985. Dating of human-ape splitting by a molecular clock of mitochondrial DNA. Journal of Molecular Evolution 22:160–174.

Hudson, R. R. 2002. Generating samples under a Wright-Fisher neutral model of genetic variation. Bioinformatics 18:337–338.

Joly, S., P. A. McLenachan, and P. J. Lockhart. 2009. A statistical approach for distinguishing hybridization and incomplete lineage sorting. American Naturalist 174:E54–E70.

Jukes, T. H. and C. R. Cantor. 1969. Evolution of protein molecules. Pages 21–132, *in* Mammalian protein metabolism (H. Monroe, ed.). New York: Academic Press.

Kingman, J. F. C. 1982. On the genealogy of large populations. Journal of Applied Probability 19:27–43.

Kubatko, L. S. 2009. Identifying hybridization events in the presence of coalescence via model selection. Systematic Biology 58:478–488.

Kubatko, L. S. and J. Chifman. 2015. An invariants-based method for hybridization detection from genome-scale sequence data. bioRxiv doi:10.1101/034348.

Lake, J. A. 1987. A rate-independent technique for analysis of nucleic acid sequences: evolutionary parsimony. Molecular Biology and Evolution 4:167–191.

Mallet, J. 2005. Hybridization as an invasion of the genome. Trends in Ecology and Evolution 20:229–237.

Mallet, J. 2007. Hybrid speciation. Nature 446:279–283.

Martin, S. H., K. K. Dasmahapatra, N. J. Nadeau, C. Salazar, J. R. Walters, F. Simpson, M. Blaxter, A. Manica, J. Mallet, and C. D. Jiggins. 2013. Genome-wide evidence for speciation with gene flow in *Heliconius* butterflies. Genome Research 23:1817–1828.

Martin, S. H., J. W. Davey, and C. D. Jiggins. 2014. Evaluating the use of ABBA-BABA statistics to locate introgressed loci. Molecular Biology and Evolution 32:244–257.

Meng, C. and L. S. Kubatko. 2009. Detecting hybrid speciation in the presence of incomplete lineage sorting using gene tree incongruence: a model. Theoretical Population Biology 75:35–45.

Patterson, N., P. Moorjani, Y. Luo, M. Swapan, N. Rohland, Y. Zhan, T. Genschoreck, T. Webster, and D. Reich. 2012. Ancient admixture in human history. Genetics 192:1065–1093.

Pease, J. B. and M. W. Hahn. 2015. Detection and polarization of introgression in a five-taxon phylogeny. Systematic Biology 64:651–662.

Rambaut, A. and N. C. Grassly. 1997. Seq-Gen: an application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees. Computer Applications in the Biosciences 13:235–238.

Solís-Lumis, C. and C. Ané. 2016. Inferring phylogenetic networks with maximum pseudolikelihood under incomplete lineage sorting. PLoS Genetics 12:e1005896.

Tavaré, S. 1986. Some probabilistic and statistical problems in the analysis of DNA sequences. Lectures on Mathematics in the Life Sciences 17:57–86.

Yu, Y., R. M. Barnett, and L. Nakhleh. 2013. Parsimonious inference of hybridization in the presence of incomplete lineage sorting. Systematic Biology 62:738–751.

Yu, Y., J. H. Degnan, and L. Nakhleh. 2012. The probability of a gene tree topology within a phylogenetic network with applications to hybridization detection. PLoS Genetics 8:e1002660.

Yu, Y., J. Dong, K. J. Liu, and L. Nakhleh. 2014. Maximum likelihood inference of reticulate evolutionary histories. Proceedings of the National Academy of Sciences 111:16448–16453.

Yu, Y., C. Than, J. H. Degnan, and L. Nakhleh. 2011. Coalescent histories on phylogenetic networks and detection of hybridization despite incomplete lineage sorting. Systematic Biology 60:138–149.

Box 1. Example code to run hybridization detection analyses with HyDe using the Python API. The `phyde` module is loaded using the `import` command:

```
# import the phyde module
import phyde
```

Data are read in and stored using the *HydeData* class by passing the names of the input file, map file, and outgroup, as well as the number of individuals, the number of populations, and the number of sites.

```
# read in data using HydeData class
data = phyde.HydeData("data.txt", "map.txt",
                      "outgroup", 16, 4, 50000)
```

With the data read in, we can use methods implemented in the *HydeData* class to test for hybridization at the population level [`test_triple(p1, hyb, p2)`], at the individual level [`test_individuals(p1, hyb, p2)`], and can bootstrap resample individuals [`bootstrap_triple(p1, hyb, p2, nreps)`].

```
# test for hybridization in the "sp2" population
res1 = data. test_triple("sp1", "sp2", "sp3")

# test all individuals in the "sp2" population
res2 = data.test_individuals("sp1", "sp2", "sp3")

# bootstrap individuals in the "sp2" population 200 times
res3 = data.bootstrap_triple("sp1", "sp2", "sp3", 200)
```

Table 1: Results of the population-level hybridization detection analyses for the non-uniform hybridization simulation using the *run_hyde.py* script.

# sites	**Z-score**	**P-value**	$\hat{\gamma}$
100 000	3.798	7.310×10^{-5}	0.908
250 000	5.897	1.854×10^{-9}	0.913
500 000	9.821	$\sim$0.0	0.897

Table 2: Results of the individual-level hybridization detection analyses for the non-uniform hybridization simulation using the *individual_hyde.py* script. Individual 'i10' was the only hybrid in the population and was simulated with $\gamma = 0.5$.

	100 000			250 000			500 000		
Individual	**Z-score**	**P-value**	$\hat{\gamma}$	**Z-score**	**P-value**	$\hat{\gamma}$	**Z-score**	**P-value**	$\hat{\gamma}$
i6	-0.861	0.805	n.s.	-0.641	0.739	n.s.	0.632	0.264	n.s.
i7	-0.175	0.570	n.s.	-0.986	0.838	n.s.	-0.305	0.620	n.s.
i8	0.011	0.496	n.s.	-1.223	0.889	n.s.	-0.040	0.516	n.s.
i9	-1.315	0.906	n.s.	-0.357	0.639	n.s.	0.823	0.205	n.s.
i10	15.743	$\sim$0.0	0.495	24.925	$\sim$0.0	0.513	35.540	$\sim$0.0	0.497

n.s. = not significant.