Profiling of Indel Phases in Coding Regions

by

Ziqi Zhu

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved October 2022 by the
Graduate Supervisory Committee:

Reed Cartwright, Chair
Jay Taylor
Jeremy Wideman
Marco Mangone

ARIZONA STATE UNIVERSITY

December 2022

ABSTRACT

Advances in sequencing technology have generated an enormous amount of data over the past decade. Equally advanced computational methods are needed to conduct comparative and functional genomic studies on these datasets, in particular tools that appropriately interpret indels within an evolutionary framework. The evolutionary history of indels is complex and often involves repetitive genomic regions, which makes identification, alignment, and annotation difficult. While previous studies have found that indel lengths in both deoxyribonucleic acid and proteins obey a power law, probabilistic models for indel evolution have rarely been explored due to their computational complexity.

In my research, I first explore an application of an expectation-maximization algorithm for maximum-likelihood training of a codon substitution model. I demonstrate the training accuracy of the expectation-maximization on my substitution model. Then I apply this algorithm on a published 90 pairwise species dataset and find a negative correlation between the branch length and non-synonymous selection coefficient.

Second, I develop a post-alignment fixation method to profile each indel event into three different phases according to its codon position. Because current codon-aware models can only identify the indels by placing the gaps between codons and lead to the misalignment of the sequences. I find that the mouse-rat species pair is under purifying selection by looking at the proportion difference of the indel phases. I also demonstrate the power of my sliding-window method by comparing the post-aligned and original gap positions.

Third, I create an indel-phase moore machine including the indel rates of three phases, length distributions, and codon substitution models. Then I design a gillespie simulation that is capable of generating true sequence alignments. Next I develop an importance sampling method within the expectation-maximization algorithm that can successfully train the indel-phase model and infer accurate parameter estimates from alignments.

Finally, I extend the indel phase analysis to the 90 pairwise species dataset across three alignment methods, including Mafft+sw method developed in chapter 3, coati-sampling methods applied in chapter 4, and coati-max method. Also I explore a non-linear relationship between the dN/dS and Zn/(Zn+Zs) ratio across 90 species pairs.

DEDICATION

For my family and friends, who taught me to be curious, positive, and persistent.

Remember fellows, work hard and be kind, then amazing things will happen.

ACKNOWLEDGMENTS

Thanks to my advisor, whose input provided me the space and support I needed to work and grow. Thanks to my committee, whose input improved this dissertation and the work described herein. Thanks to the members of my lab past and present, who created a genuine and intellectually-stimulating environment to work in and provided helpful feedback. Thanks to the Arizona State University School of Life Sciences and College of Liberal Arts and Sciences for the TA support within last five years.

TABLE OF CONTENTS

CHAPTER

CHAPTER Page

APPENDIX

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

BACKGROUND

Advances in sequencing technology have generated an enormous amount of data over the past decade (Van Dijk et al. 2014). Equally advanced computational methods are needed to conduct comparative and functional genomic studies on these datasets. These methods include the application of a substitution (chapter 2) model and the development of an insertion-deletion (chapter 4) model of evolution for phylogenetic inference, multiple sequence alignment, and ancestral state reconstruction, as well as post-alignment fixation of coding sequences prevalent in genomic datasets (Ranwez et al. 2011), through indel-phase profiling method (chapter 3).

Substitution models of evolution are used to make predictions about the substitution process in molecular sequences along the branches of a tree or phylogeny. These models can be classified into two main groups: empirical models, which are based on observed frequencies of substitutions in sets of alignments, and mechanistic models, which use estimated evolutionary parameters to calculate substitution matrices. These parameterized models are more advanced in that they can take into account how mutational biases and natural selection may skew these observed transition rates (Yang and Nielsen 2008). Parameter optimization for these models are commonly done through a Maximum Likelihood (ML) method. In the second chapter, I present an application of an expectation-maximization algorithm for maximum-likelihood training of a codon substitution model (MG94+GTR).

Insertion-deletion (Indels) mutations are one of the most common classes of mutations in the genome, second only to single base substitution (Taylor, Ponting, and Copley 2004). Indels are normally generated by DNA polymerase errors or incorrect DNA repair during the replication process. The evolutionary history of indels is complex and often involves repetitive genomic regions, which makes identification, alignment, and annotation difficult (Kunkel 2004). Therefore, accurate and reproducible methods of identifying indels are crucial for understanding their evolutionary patterns and predicting unknown gene functions. In the third chapter, I present an application of a post-alignment fixation method (sliding-window method) to profile each indel event into three different phases (Figure 3.1) according to its position within a codon. This method helps to find all three indel phase events that improves upon current codon-aware aligners that can only support phase 0 indels.

Compared to substitution models, indel models are far less developed due to their complexity. For example, indel rates are normally inferred based on gap counts, while a single gap can reflect more than one event (Miklós, Lunter, and Holmes 2004). The excess of deletion over insertion events (Gu and Li 1995; Ophir and Graur 1997; Zhang and Gerstein 2003), also requires models to evaluate separate indel length distributions. In the fourth chapter, I present an innovative indel-phase model that makes several improvements to current models: 1) while previous models for indel evolution assumes one indel rate, our model proposes three rates for each insertion and deletion event based on the corresponding indel phases; 2) this model is richer because of the inclusion of an MG94 + GTR substitution model within; 3) I apply the importance sampling method within the EM algorithm to remove the alignment bias when inferring parameters from the indel-phase model. The simulation results demonstrate the efficiency and accuracy of my indel-phase model.

In the last chapter, I extend our indel phase analysis to a published 90 pairwise species across the tree of life, including 15 eukaryotic, 6 archaeal, and 69 bacterial clades. Also, I apply three different alignment methods on this dataset and compare the phase proportions and distributions between them. I find the effective purifying selection existing withing coding regions by looking at the phase proportions across 90 species pairs.

In summary, I develop and evaluate methods for profiling the indel phases and estimating the evolutionary parameters derived from substitution and indel models in pairwise alignments. The post-alignment fixation method (sliding-window) is able to generate better alignments, where better alignments help improve the phylogenetic inference, ancestral sequence reconstruction, and gene annotation area (Rosenberg 2009). Besides, the EM algorithm and importance sampling method is capable of estimating the indel rates of three different phases and length distributions from genomic data by applying my innovative indel-phase model. There are more ways to do so, and many more that are not discussed here. Each has strengths and weaknesses, but understanding what these are is critical to success.

Chapter 2

APPLICATION OF AN EM ALGORITHM IN SIMULATING MG94+GTR SUBSTITUTION MODEL

2.1 Introduction

Various substitution models have been commonly used for DNA, RNA, and protein sequence analysis, and are particularly effective in detecting the signals of natural selection acting on proteins. An early application of such models include dating divergence time between species in either genes or proteins, and inferring their phylogenetic trees in the molecular biology area (Felsenstein 1981). The most commonly used substitution models are continuous-time, finite-state Markov models (Karlin 2014). The Markov models means that at any given time, the next state is only dependent on the current state and is independent of anything in the past. So their instantaneous rate matrix R_{ij} can be parameterized in various mathematical forms based on different biological assumptions, regardless of the unseen substitution history. Two kinds of Markov models have been used to describe the codon-level sequence evolution process (Kosiol, Holmes, and Goldman 2007). Empirical models emphasize on outlining the substitution patterns observed dependent on huge quantities of data, rather than specifying the biological parameters including transition-transversion ratio, codon frequency bias, and selective pressures (Liò and Goldman 1998). However, mechanistic models include those parameters that can regulate sequence evolution, allowing the testing of hypotheses related to those parameters for a diversity of data sets.

Mechanistic models, including Muse and Gaut (MG94) and Goldman and Yang (GY94), can describe a Markovian substitution process with a state space consisting of the 61 sense codons (Suchard, Weiss, and Sinsheimer 2001). The MG94 model is the first model that accounts for the coding structure of nucleotides. The main advantage is estimating the non-synonymous substitution rates that it corrects for multiple hits at a codon (Muse and Gaut 1994). In contrast to the GY94 model formulation, MG94 has entries of the Markov generator that is proportional to the stationary probability of the target nucleotide instead of the codons, which is desirable (Huelsenbeck and Dyer 2004). Parameter estimates such as nucleotide frequencies, nucleotide exchangeabilities, and selection pressure on non-synonymous substitutions can be derived from the preconceived MG94 models (Muse and Gaut, 1994)

There exists software that can optimize the parameters from the codon substitution models. For example, codeml is commonly used for phylogenetic analyses of DNA and protein sequences using maximum likelihood (ML) method through GY94 models (Yang 2007). Hyphy provides a platform for carrying out likelihood-based analyses on multiple sequence data by using MG94 model. In this research, I introduce a EM method for maximum likelihood training of the MG94 model. Expectation maximization (EM) algorithm is an iterative method to find the local maximum likelihood estimates (Dempster, Laird, and Rubin 1977). This algorithm can be used to train substitution models where the structural context of residues are treated as hidden variables that evolve over time (Holmes and Rubin 2002). Typical studies of EM applications include using Baum-Welch (a special algorithm of EM) to train profile HMMs with Dirichlet mixture priors (Brown et al. 1993), and using Inside-Outside algorithm to train SCFG profiles for RNA (Durbin et al. 1998). Besides, alternative methods such as Newton Raphson and Nelder-Mead algorithm are commonly used for multidimensional optimization. However, Newton's method might fail if there are points of inflection, local maxima, or minima around the root of the first derivative or the initial value (Moré and Sorensen 1982). In comparison, the Nelder-Mead method is capable of searching for the local optimum very fast in a multidimensional space due to its derivative-free superiority, as long as the substitution model is less heavily parameterized, such as mechanistic codon models (wenci 1979).

A special EM algorithm (Phylo-EM) introduced by (Holmes and Rubin, 2002) is implemented in this paper. This method reduces the computational burden of the estimation process largely by repeated utilization of an expectation (E) step followed by the maximization (M) step. In E-step, a likelihood function is calculated and the sufficient statistics are obtained for solving the EM optimization. The sufficient statistics are expectations taken over the posterior distribution of Markov chain trajectories conditional on observed residues. In M-step, through maximizing the sum over all possible histories with respect to the current parameter estimates, a new parameter set is generated. The algorithm stops until the log likelihood converges to a local maximum.

This paper presents an application of the Phylo-EM algorithm via simulating a codon substitution model (MG94). Expanding on the theoretical proof of the feasibility of the phylo-EM, I develop it in a computational language (R), and subsequently conduct 100 simulations of pseudo datasets and measure the accuracy of the parameter estimates. Furthermore, I implement the Phylo-EM on 90 real species pairs sampled from the tree of life (Zou and Zhang 2021), including 15 eukaryotic, 6 archaeal, and 69 bacterial clades. Moreover, our results indicate that there is a negative correlation between branch length (τ) and selective pressure (ω). However, genome size (Simpson 2014), sequencing error, (Schneider et al. 2009), and inaccurate orthologs can all generate bad alignments, leading to the bias of parameter estimates. Hence, this correlation is likely due to the technical artifact, including an unproved sequence quality unity across 90 species and potential artificial selection by researchers.

2.2 Materials and Methods

2.2.1 Sequence Data

The dataset I selected for our research was a 90 pairwise sequence alignments deriving from Zou and Zhang 2021, including 15 eukaryotic, 6 archaeal, and 69 bacterial species pairs. Some of them also have a phylogenetically-related species as an outside reference, which means their sequence data actually draws from the multiple alignments. Also, four mammalian clades, fruit flies, and yeasts are extracted directly from distinctive databases, and other eukaryotic clades are derived from Ensembl database (https://useast.ensembl.org/index.html). Most prokaryotic clades are sampled from available strains in the ATGC database (Kristensen, Wolf, and Koonin 2016).

2.2.2 Codon Substitution Model

MG94+GTR Model. Codon evolution is typically modeled as a continuous-time Markov process (Rodrigue, Lartillot, and Philippe 2008), and two different types of models are commonly used: GY94 and MG94 (Goldman and Yang 1994; Muse and Gaut 1994). MG94 models codon evolution using a instantaneous transition rate matrix, $R_{i,j}$ where i and j refer to different codons and

$$R_{i,j} = \begin{cases} 0, & \text{if i and j differ at more than one nucleotide change} \\ r_{i_c j_c}\pi_{j_c}, & \text{if i and j differ by a synonymous change at codon position c} \\ \omega r_{i_c j_c}\pi_{j_c}, & \text{if i and j differ by a nonsynonymous change at codon position c} \end{cases}$$

Additionally, $R_{i,i}$ is set such that rows of R sum to zero. i_c and j_c refer to the nucleotides at position c in codons i and j respectively. $r_{i_c j_c}$ is one of the six nucleotide exchangeability parameters from the GTR model (Tavaré et al. 1986). π_{j_c} is the frequency of nucleotide with the constraint of sum of π_{j_c} equals to 1. ω is the coefficient of selective strength on non-synonymous codon substitutions. The stationary frequency of codon i in this model is $\pi_i = \pi_{i_1}\pi_{i_2}\pi_{i_3}$, since codon stationary probabilities are proportional to the product of nucleotide frequencies at each of the three codon positions (Rodrigue, Lartillot, and Philippe 2008). This model is time reversible, satisfying the equality $R_{i,j}\pi_i = R_{j,i}\pi_j$, where π_i and π_j are the frequencies of codons i and j.

When ω equals to 1, this model represents the well-known generalised time-reversible (GTR) model (Tavaré et al. 1986) describing the nucleotide level substitutions.

$$r_{i_c j_c} = \begin{bmatrix} *, & \sigma_{AC}, & \sigma_{AG}, & \sigma_{AT} \\ \sigma_{CA}, & *, & \sigma_{CG}, & \sigma_{CT} \\ \sigma_{GA}, & \sigma_{GC}, & *, & \sigma_{GT} \\ \sigma_{TA}, & \sigma_{TC}, & \sigma_{TG}, & * \end{bmatrix} \tag{2.1}$$

GTR is the most neutral, independent, and time-reversible model possible. It requires six nucleotide exchange rate parameters (Equ. 2.1), as well as four equilibrium base frequency parameters. I chose the GTR model for my research due to its parameter richness and symmetry.

Following the MG94 model, Goldman and Yang 1994 also develop a model with parameters on codon fitness GY94 model is similar to MG94 except for their π_j is the frequency of codons instead of nucleotides, which are calculated by the product of three nucleotides in each codon position ($\pi_j = \pi_{j_1}\pi_{j_2}\pi_{j_3}$). In my research, I selected MG94 over GY94 because the latter tends to push the states away from the Markovian equilibrium (Rodrigue, Lartillot, and Philippe 2008). For example, If there exists a DNA sequence whose nucleotide frequencies of A/T are higher than G/C, then the instantaneous transition rate of CGC → CTC is higher than the transitional rate of ATA → AGA under the MG style models. This is consistent with the mutational bias ($T > G$). While under the GY style models, the result switches because the codon stationary probabilities are proportional to the product of three nucleotides.

Codon Substitution Probability. Using matrix R, the probability that codon I evolves into codon J over evolutionary distance t is

$$P(I=i,J=j;t) = \left(e^{Rt}\right)_{ij} \times \pi_i \tag{2.2}$$

where π_i is the stationary frequency of codon i. Furthermore, consider two related sequences, and let A and B be the observed sets of aligned codons in these sequences. The probability that A evolves into B over evolutionary distance t is

$$P(A=\{a_1,\ldots,a_n\},B=\{b_1,\ldots,b_n\}) = \prod_{k=1}^{n} P(a_k,b_k;t) \tag{2.3}$$

2.2.3 Expectation-Maximization and Phylo-EM

The Expectation-Maximization (EM) algorithm is an iterative method of finding local maximum likelihood estimates of parameters for statistical models, when the model depends on unobserved latent variables. Estimated model parameters are updated through a two-step procedure. The E-step calculates a function, Q, for the expected log-likelihood of observed data given current parameter estimates. The M-step the calculates new parameters by finding parameters that maximize the Q function defined in the previous step.

E-Step. Let X represent the set of observed data, Z represent the set of unobserved latent data, and θ represent a vector of model parameters. Let $L(\theta;X,Z) \propto P(X,Z;\theta)$ be the full-likelihood function. The E-step defines the Q function as the expected value of the full-likelihood function of θ given the current estimate of the parameters θ_t:

$$Q(\theta|\theta_t) = \sum_Z P(Z|X;\theta_t)\log L(\theta;X,Z) \tag{2.4}$$

M-Step. The M-step finds parameters that maximize Q:

$$\theta_{t+1} = \arg\max_\theta Q(\theta|\theta_t) \tag{2.5}$$

The EM algorithm alternates E-steps and M-steps until a stopping criterion is satisfied. The final estimate θ_t is a (local) maximum likelihood estimate of the model parameters given the observed data, X.

Phylo-EM. Phylo-EM (Holmes and Rubin 2002) is an EM algorithm for estimating substitution rate matrices from pairwise alignments. Following Holmes and Rubin 2002, let a and b be the residues observed at a single column of a pairwise alignment, where a is considered to the the ancestor of b. The path from a to b can be treated as a continuous-time Markov chain, where h_t is the state of the chain at time t. The end points of this chain are $h_0 = a$ and $h_T = b$. Let R represent the instantaneous transition rate matrix of our Markov chain, and $M(T) = e^{R \times T}$ represent the transition matrix for discrete time T. Phylo-EM splits this history into three sections (Figure 2.1): (1) a evolves into i after time t with probability $M(t)_{ai}$, (2) i evolves into j after time dt with probability R_{ij}dt, and (3) j evolves into b after time $T-t$ with probability $M(T-t)_{jb}$.

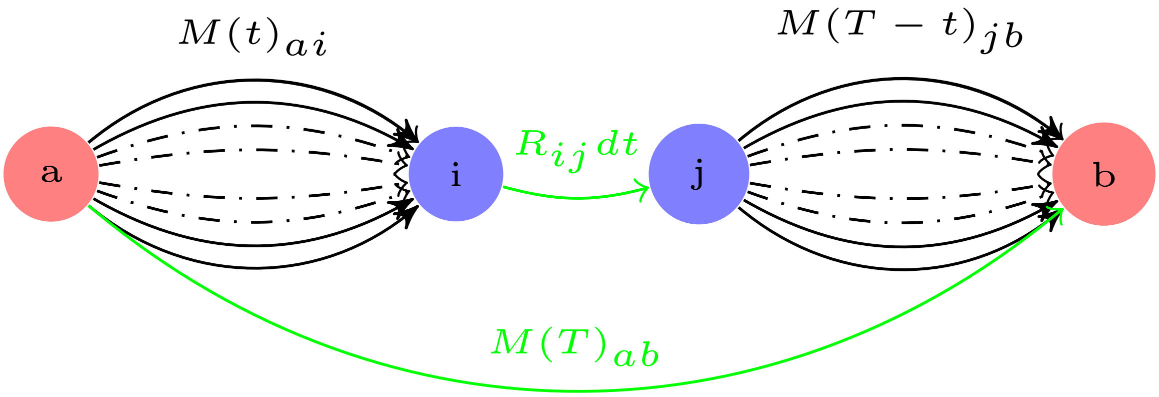


Figure 2.1: **Phylo-EM Model |** Phylo-EM, a special expectation maximization algorithm for maximum likelihood training of substitution rate matrices in sequence alignment. This algorithm can be applied in training hidden substitution models, where the structure of a residue is treated as a hidden variable which evolved during time T (Holmes and Rubin 2002). R_{ij} is the instantaneous rate of substitution from residue i to j. $M(t)_{ai}$ is the transition matrix from residue a to i, $M(T-t)_{jb}$ is the transition matrix from residue j to b. $M(T)_{ab}$ is the transition matrix from observed residue a to b containing all possible substitution paths.

In Phylo-EM's E-step, the Q function for residues a and b is calculated as

$$\begin{aligned} Q(\theta|\theta_t,a,b) &= \sum_i \sum_{j,i\neq j} C_{ij}\log R_{ij} + \sum_i W_i R_{ii} \\ C_{ij} &= \frac{1}{M(T)_{ab}}\left(\int_0^T M(t)_{ai}(R_{ij}dt)M(T-t)_{jb}\right) \\ W_{ij} &= \frac{1}{M(T)_{ab}}\left(\int_0^T M(t)_{ai}(dt)M(T-t)_{jb}\right) \end{aligned} \tag{2.6}$$

C_{ij} represents the expected number of times that the transition $i \to j$ occurs, and W_i represents the expected amount of time that the system spent in state i. Those two sufficient statistics are expectations taken over the posterior distribution of Markov chain trajectories conditional on observed residues a and b.

Estimating MG94+GTR Parameters. The M-step of the Phylo-EM updates parameter estimates by maximizing the Q function. The parameters of MG95+GTR are ω and 6 σ's, which are updated as follows:

$$\begin{aligned}\omega : A_w = \sum\sum_{i,j,nsyn} C_{ij},\ \ B_w = \sum\sum_{i,j,nsyn} W_i R_{i,i}\\ \sigma : A_\sigma = \sum\sum_{i,j,\sigma} C_{ij},\ \ B_\sigma = \sum\sum_{i,j,\sigma} W_i R_{i,i}\\ \hat{\omega} = A_w / B_w\\ \hat{\sigma} = A_\sigma / B_\sigma\end{aligned} \tag{2.7}$$

where A_w is the expected number of non-synonymous transitions from codon i to j, and B_w is the expected amount of time spent on the non-synonymous transitions from codon i to j. A_σ is the expected number of one of the six nucleotide transitions, and B_σ is the expected amount of time spent on one of the six nucleotide transitions. The estimate of omega was calculated by A_w divided by B_w, and the estimate of six σs were calculated by A_σ divided by B_σ.

Additionally, the E-step computation takes a space complexity O (61^6) and time complexity O (61^4). The whole EM process would not stop until the root mean square error (RMSE) of parameters between iteration was less than the default threshold (10^{-4}).

2.2.4 Parameter Space

The ideal range of parameters $\{\pi, \omega, \sigma\}$ should be comparatively wide and biologically reliable. For example, (i) the frequency of nucleotide A (π_A) is uniformly sampled between 0 and 0.5, and the remaining three nucleotides can be obtained via the Chargaff's rule ($\pi_A = \pi_T$; $\pi_C = \pi_G$). (ii) ω, as a measure of the selective strength bearing on non-synonymous substitutions, is uniformly sampled within the range of (0,0.5] (Zou and Zhang 2021). (iii) The six nucleotide exchangeabilities (σ) follow a gamma distribution with a shape parameter α and a scale parameter β, where α is the reverse of square of coefficient of variation (CV), and β is the mean substitution rate divided by α. In order to obtain a relatively wide distribution of σ with a lower mean of τ, I choose the mean substitution rate of 0.4 and the CV of 1. Since a lower τ approximates to the estimates across real species data, and a higher CV tends to reduce the bias from random simulations. All 100 true parameter values are shown in the supplementary table A.1.

2.2.5 Initial Parameter Estimate

Good initial parameters are the key to reducing the running time of EM convergence. (i) Because of the missing stop codons in the dataset, our initial values of π are estimated via an EM method on a multinomial distribution of codon frequencies. The E-step calculates a likelihood function of 61 sense codons, and the M-step updates the nucleotide frequencies by including the expected frequencies of stop codons. (ii) σs are estimated through a distance method applied on the GTR model (Felsenstein and Felenstein 2004). Constrained from the neutral assumption of the method, I only seek for the third positional change of each 4-fold degenerate site of each codon. (iii) ω is approximated by the ratio of observed non-synonymous substitutions to the expected non-synonymous substitutions. The observed substitutions are extracted directly from the sequence pairs. The expected substitutions are calculated by multiplying the number of codons by the expected substitution rates, which can be obtained by setting up the value of ω equals to 1 in the MG94 model. This removes the purifying selection constraints against the non-synonymous codons.

2.2.6 Simulations

I conducted 100 simulations for training the GTR+MG94 model within the Phylo-EM method. Each simulation utilized a random parameter set to generate pseudo sequence pairs, ran the EM algorithm, and measured the accuracy of the estimates. Besides, three gradients of codon length $\{10^5, 10^6, 10^7\}$ were selected for each hundred simulations, due to their similar range compared with the 90 species data from the paper.

2.2.7 Nelder-Mead Algorithm For Derivative-free Optimization

I compare the Phylo-EM to a numerically optimizing algorithm called Nelder-Mead algorithm, which is a direct search method without necessarily knowing the derivatives with respect to parameters. I find this method in a R package (dfoptim) and conduct 100 identical simulations. Notably, I modify the convergence criterion of this package so it stays consistent with the Phylo-EM method, which should be the RMSE of parameters instead of log likelihood between iterations.

2.3 Results

2.3.1 Validation

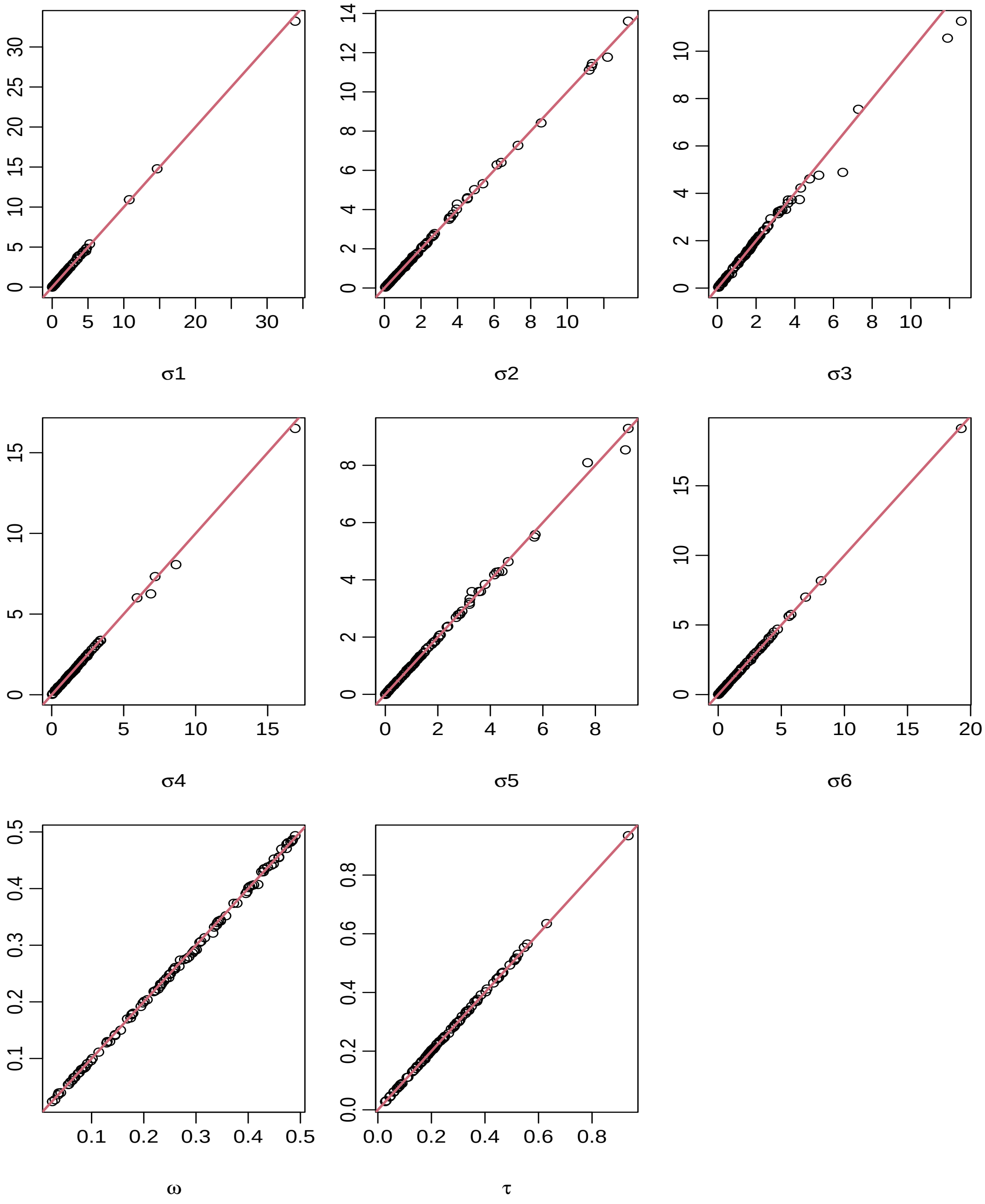


Figure 2.2: **QQplot of Parameter Estimates in the Codon Length of** 10^5 **of 100 Simulations |** Estimated parameters include six σs, ω, and τ. $\{\sigma1,\sigma2,\sigma3,\sigma4,\sigma5,\sigma6\}$ corresponds to the $\{\sigma_{AC},\sigma_{AG},\sigma_{AT},\sigma_{CG},\sigma_{CT},\sigma_{GT}\}$.

To examine the power of the Phylo-EM algorithm for estimating the parameters extracted from the GTR + MG94 substitution models, we generate a qqplot reflecting the correlation between the true and estimated parameters, including six σs, ω, and τ. The diagonal line of Figure 2.2 measures the accuracy between the true and estimated parameters in the codon length of 10^5. All eight parameter estimates across 100 simulations approximate the true value, except for a few simulated runs. The 100 parameter estimates table is in the APPENDIX A.2. Figure 2.3 displays the root mean squared errors (RMSE) between the true and estimated parameters across 100 simulations in the codon length of $\{10^5, 10^6, 10^7\}$. All three sample sizes are capable of generating accurate parameter estimates with an extremely low RMSE. The mean RMSE values for three sample sizes are 0.061, 0.0168, and 0.007 respectively. The 95% confidence interval for three sample sizes are [0.042,0.080], [0.011,0.022], and [0.005,0.010]. The RMSE value decreases with the increase of sample size across 100 simulations. The two figures above demonstrated an excellent training accuracy of the Phylo-EM algorithm on the MG94+GTR model.

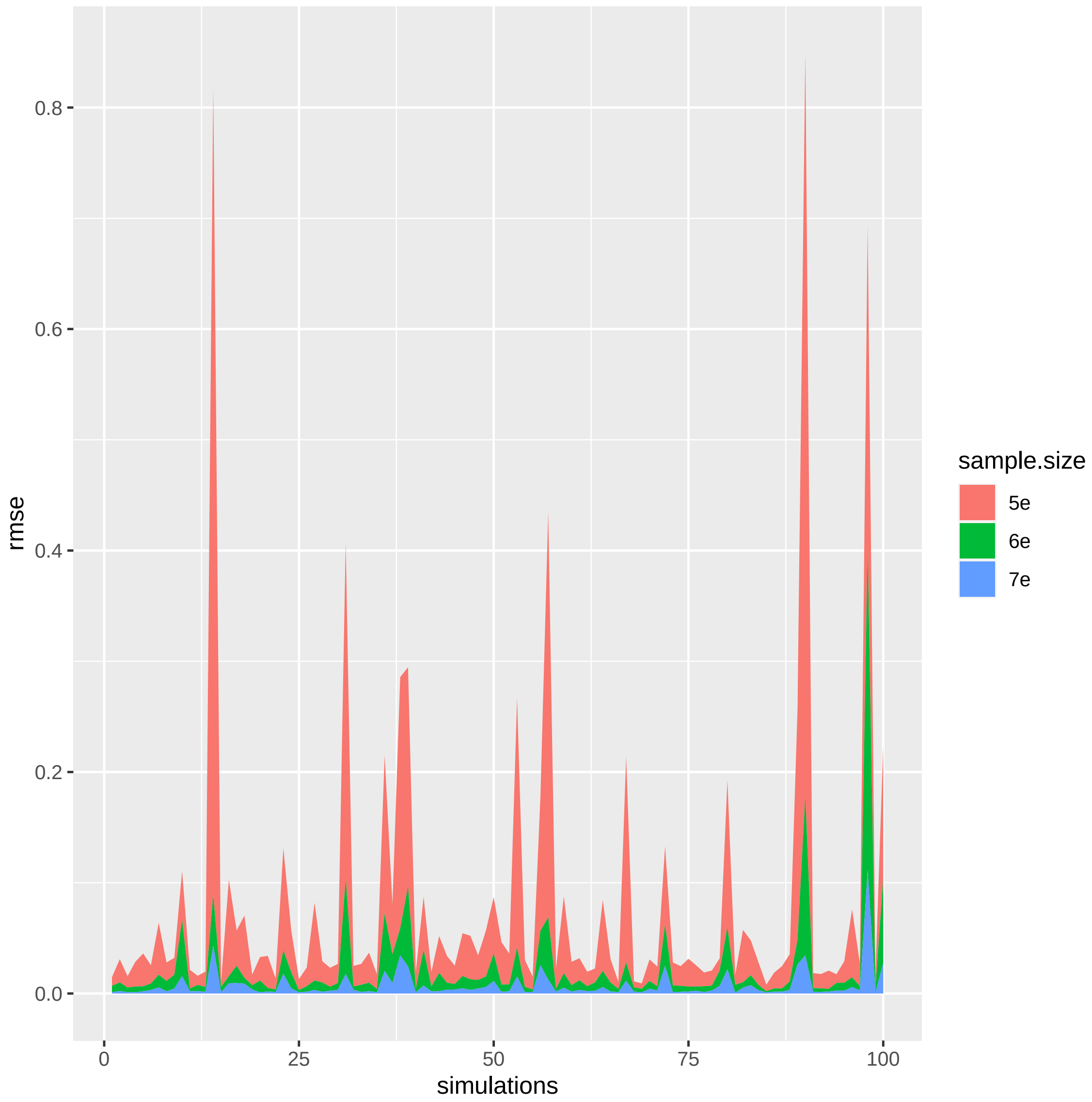


Figure 2.3: **Root Mean Squared Errors of Parameter Estimates across 100 Simulations in Three Gradient of Sample Size.** | The label of {5e,6e,7e} means the codon length of $\{10^5, 10^6, 10^7\}$respectively.

2.3.2 Data Analysis of 90 Species

After validating the accuracy of the Phylo-EM, I apply it to a published dataset – a series of concatenated homologous protein coding sequences from 90 species pairs, including 15 eukaryotic, 6 archaeal, and 69 bacterial clades (Zou and Zhang 2021). I generate a data distribution plot of eight parameters from 90 species (Fig 2.4). The mean value of σ_{AC}, σ_{AG}, σ_{AT}, σ_{CG}, σ_{CT}, σ_{GT} are 1.23, 2.22, 0.84, 1.16, 2.12, and 0.80 respectively. And the transition rates of nucleotides σ_{AG}, σ_{CT} are higher than the four transversion rates σ_{AC}, σ_{AT}, σ_{CG}, σ_{GT}. The mean value of ω, τ are 0.12 and 0.82. The parameter estimates of 90 species pairs table is in the APPENDIX A.4.

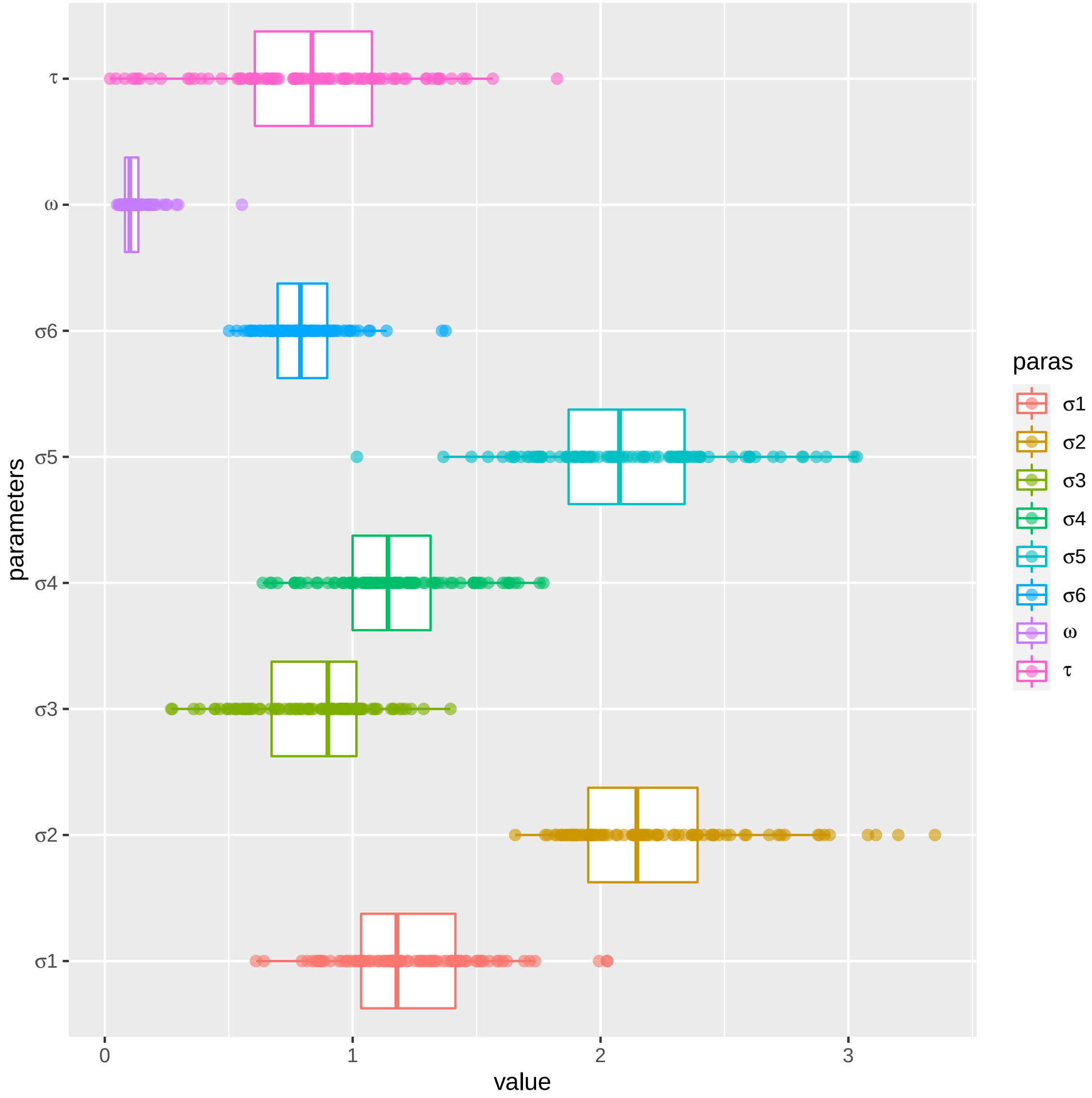


Figure 2.4: **Distribution of 8 Parameter Estimates from 90 Species Pairs.** | The box boundaries are the 25% and 75% quartile. The middle line marks the mean value.

To further investigate the relationship between the non-synonymous selective coefficient (ω) and the branch length (τ), I bootstrap 10 replications in each species with the same sample size. The mean of the standard error of the ω and τ of 90 species is 0.0002 and 0.0011, which are extremely small. Figure 2.5 displays a negative correlation between the parameter estimates of ω and τ across 90 species, which can be explained by several mechanisms discussed later. I also found that most eukaryotic branches (number 1 to 16) have a larger ω and smaller τ, while most prokaryotic branches (number 21 to 90) display the opposite pattern.

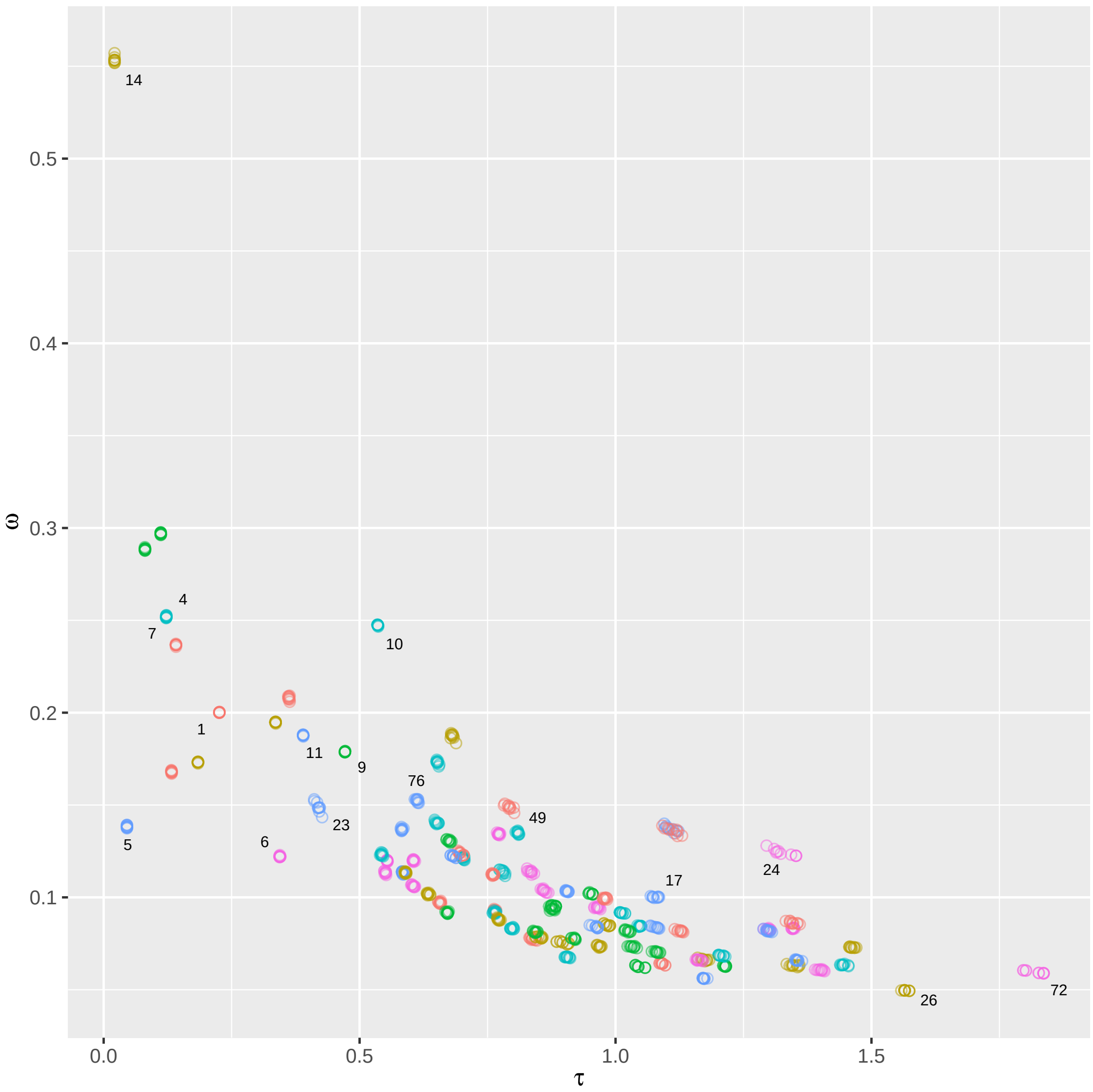


Figure 2.5: **The Correlation Plot between ω and τ.** | All species names are represented from number 1 to 90, with 10 bootstrapping replicates in every species. Eukaryotic species taxon names can be found in APPENDIX D.

2.3.3 Performance

Figure 2.6 displays the difference of running EM between using the random and estimated values as initial parameters. The average running time with random values is around 15 iterations, which decreases to 11 iterations by using the estimated ones. And the average time spent on 100 simulations is 20% less. Moreover, I need at least 5 iterations to pass the true log likelihood threshold by using random initial parameters, reducing to 3 iterations via estimates.

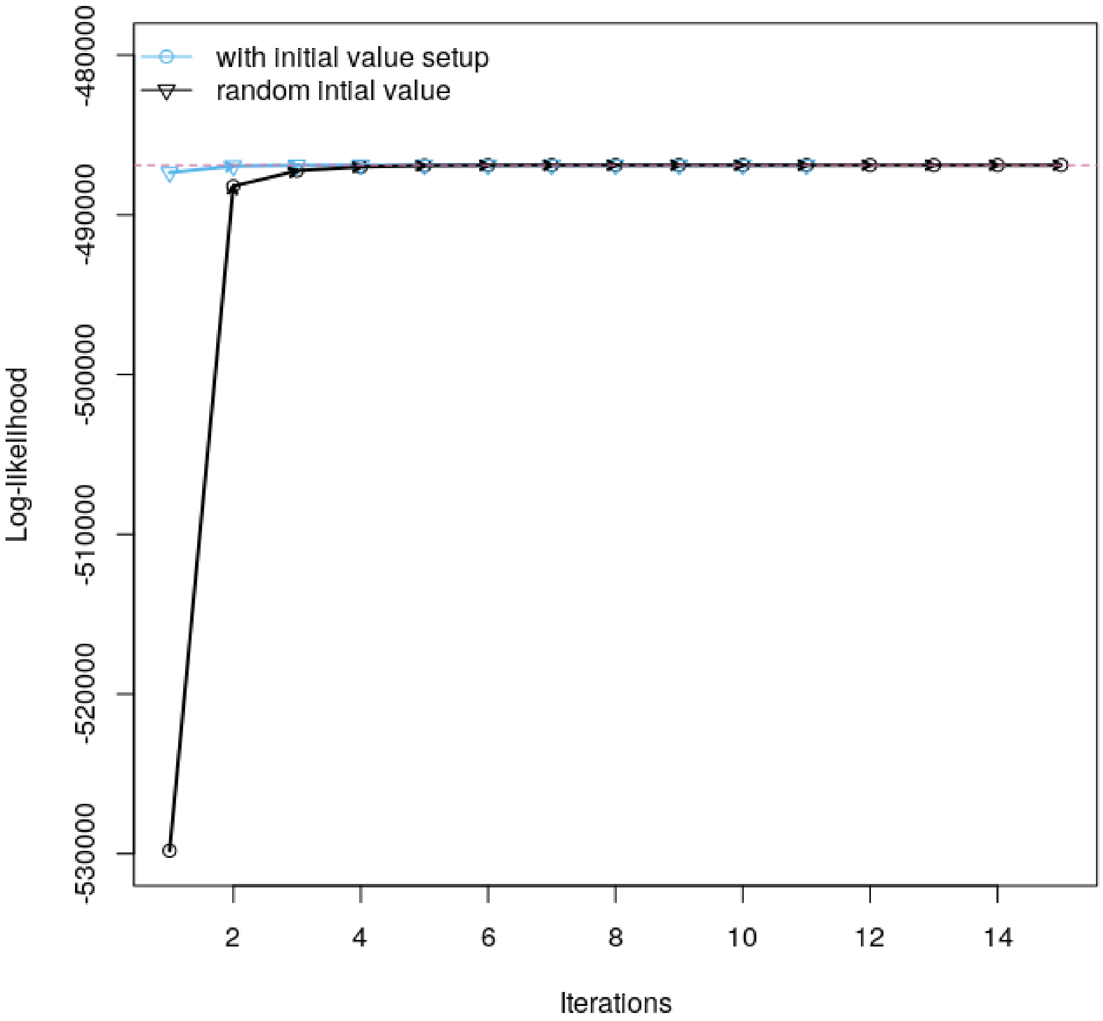


Figure 2.6: **Comparison of EM Running Time Difference between Using Random and Estimated Parameters as Initial Values.** | The pink dash line marks the value of the true LL.

I also assess the performance of the EM and nmkb algorithms for training the MG94 + GTR model. The mean running time of EM for each simulation is around 100 minutes, while the running time of the nmkb derivative-free optimization is about 15 seconds under the same stopping criterion. Figure 2.7 displays a highly overlapped RMSE distribution between the two methods. The mean RMSE value for the EM and nmkb method is 0.061 and 0.062 in the sample size of 10^5. The parameter estimates from nmkb method can be found in APPENDIX A.3.

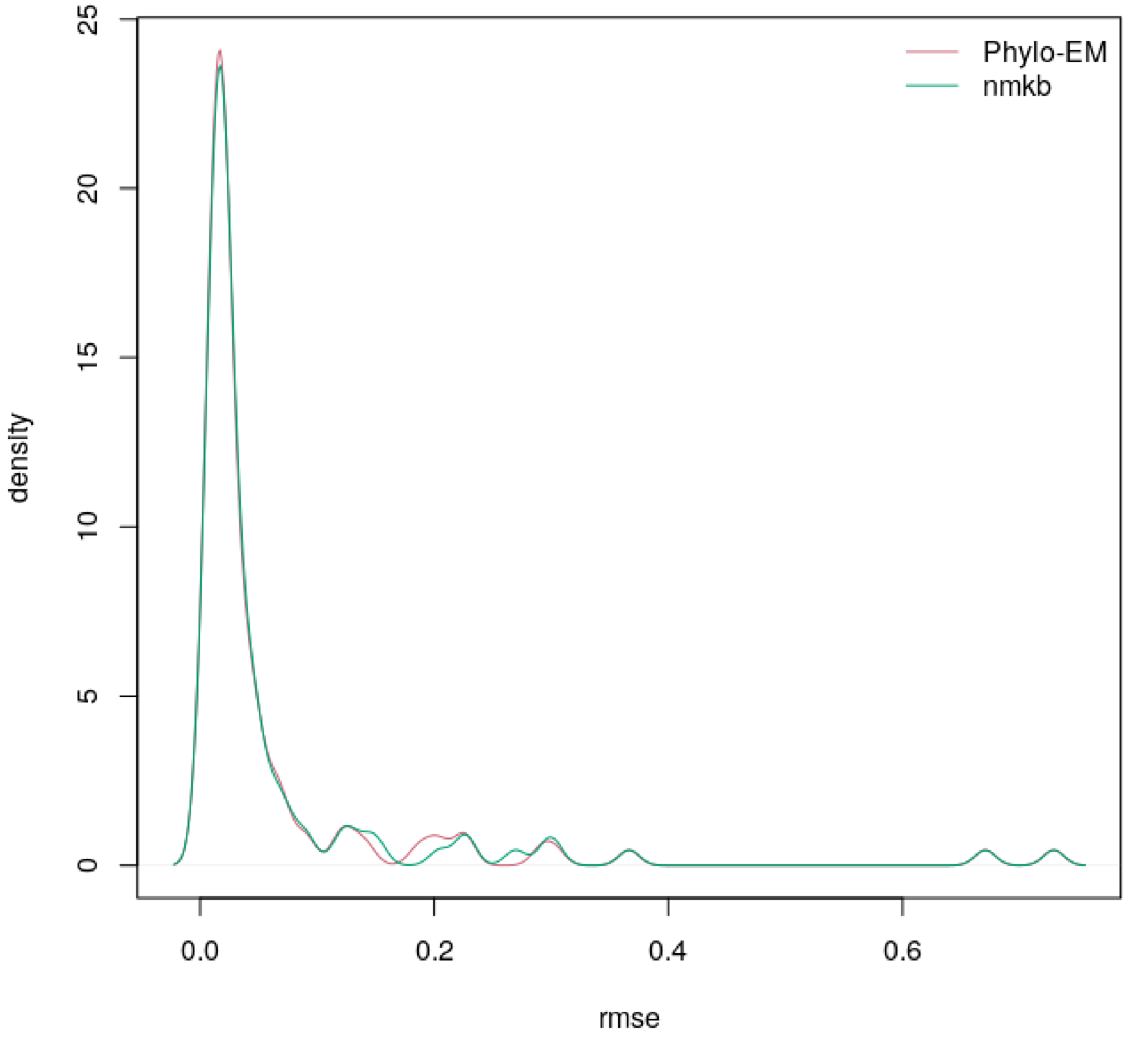


Figure 2.7: **The Distribution Plot of RMSE from Phylo-EM and Nmkb Method.** | The green line is nmkb method and the pink line is Phylo-EM method.

2.4 Discussion

The EM algorithm proposed by Holmes and Rubin 2002 is an efficient tool of training hidden substitution matrices. I apply this algorithm for simulating the MG94 + GTR models and obtaining the parameter estimates including six nucleotide exchangeabilities σ, selective pressure on non-synonymous changes ω, and branch length τ. While numerical optimization performs faster than EM for estimating the parameters of the MG94 + GTR model, EM method is still academically useful for its guaranteed increased likelihood, closed form solutions, and compatibility with incomplete datasets (Couvreur 1997). Especially in the case of complex models (ex. empirical codon model) with a large amount of parameters (Kosiol, Holmes, and Goldman 2007).

Additionally, I apply the Phylo-EM method on 90 species pairs. Figure 2.5 shows that the species carrying a smaller non-synonymous selective coefficient (ω) are mostly prokaryotes, where the corresponding branch length τ are larger. It is still unclear to me that the negative correlation between τ and ω is due to biological or technical reasons. Therefore, I propose several explanations here: (i) The effective population size (Ne) of eukaryotes is substantially (by orders of magnitude) smaller than prokaryotes, and consequently, purifying selection is not strong enough to eliminate most of the non-synonymous substitutions compared to the prokaryotes (Sela, Wolf, and Koonin 2016). (ii) Species pairs with a smaller τ are more likely to be recently diverged, where the existing mutations are still unfixed (Charlesworth 2020). Hence, many non-synonymous differences that are not fixed yet get recorded into the dataset, leading to an overestimate of ω. (iii) The sequence quality can also be an issue. For instance, The difference of genome sizes across 90 pairwise species tends to cause the discrepancy of the sequence quality. The larger the genome size, the lower the sequence quality normally is (Baker 2012). Next, even a list of one-to-one orthologous genes are claimed to be collected for each species, some paralogs are likely to mix into the dataset due to the limitation of the genome annotations and phylogenies (Smith and Hahn 2021).

In addition to the relationship analysis above, I also found that $Thermococcus_{72}$ (ATGC279) species pairs and $Lactobacillus_{26}$ (ATGC056) species pairs both had a very large τ (1.8249, 1.5656), while the corresponding value of ω were close to zero (0.0593, 0.0495). A lower ω and higher τ suggested that those species were under strong purifying selection and with a high substitution rate in each nucleotide site, which seems implausible. Since it required most substitution events were synonymous and the mutational burden fell onto the third position of almost every codon. Therefore, I suspected that the researchers had to recover good conserved coding regions by over-filtering some disqualified sequence dataset, causing the bias of my parameter estimates.

2.4.1 Limitations

It is challenging to verify any of the above explanations because: (i) requires us to obtain the effective population size (Ne) of all 90 species pairs. However, the estimation methods of Ne are various and sometimes unreliable (Wang, Santiago, and Caballero 2016), and most species lack estimated Ne as well. (ii) requires single nucleotide polymorphism (SNPs) data to recognize whether the non-synonymous codon changes are fixed yet (Ochman 2003). Other historical events that can shape the dynamics of substitutions, including Hill-Robertson effect, and background selection are not discussed either (Comeron, Williford, and Kliman 2008). (iii) demands retrieving the sequencing errors and pulling out the sequencing quality of each alignment. But our 90 pairwise alignments are extracted from various sources, including Ensembl, ATGC, flybase, and OrthoMaM database. Most aligned prokaryotes are extracted from the multiple sequence alignment indirectly, which can also be inaccurate if the phylogenetic trees are misplaced.

Chapter 3

INDEL PHASE ANALYSIS FROM THE SLIDING-WINDOW METHOD

3.1 Introduction

Indels refer to insertion and deletion events that can be caused by DNA polymerase errors or incorrect DNA repair during the replication process. The evolutionary history of indels are complicated as their identification, alignments, and annotations (Kunkel 2004). Accurate and repetitive methods of identifying indels are necessary for understanding their evolutionary patterns and essential roles in gene functions. Especially for indels occurring within the coding domain sequences that have been commonly used as drug targets, protein structure predictors, species diagnostics, and evolutionary markers (Ajawatanawong and Baldauf 2013). Proteins are macromolecules which perform a variety of functions that are essential to organisms, such as catalyzing metabolic reactions, providing structures and support, and transporting molecules within cells and throughout the body (Cozzone 2002). Exons and introns are two features of a gene, where the coding regions are considered as exon parts that are interrupted by non-coding regions known as introns. The coding frames normally begin at the 5' end with a start codon and stop at the 3' end with a stop codon (Furuno et al. 2003). The function of a coding region can be changed by mutations, including substitutions and indels. A synonymous mutation is a change in the DNA sequences that codes for the same amino acid, while a non-synonymous mutation is a change that codes for different amino acids. Notably, the coding frames must be a continuous stretch of codons in which length is multiple of three. I focus on the indels from the coding regions under selection, while the indels within neutral areas, including intergenic regions and introns, are excluded from our research scope.

According to Taylor, Ponting, and Copley 2004, there exists three indel phases in each codon, including phase 0 indels occurring between codons, phase 1 and phase 2 indels occurring from the second and third positions of the codon respectively (Fig 3.1a). Phase 0 indels never change the surrounding amino acids, while phase 1 and phase 2 indels can cause an amino acid substitution that changes the protein sequences. Current codon-aware models can only identify the indels by placing the gap locations between codons without breaking the reading frame. However, they only support the indel events between two adjacent codons and ignore the phase 1 and phase 2 events (Kosiol, Holmes, and Goldman 2007), which can lead to a misalignment of sequences (Fig 3.1b).

Indel-phase analysis are important but understudied components of molecular evolution, which are discussed by a few papers only. Taylor, Ponting, and Copley 2004 found an excess of deletions over insertion in rat lineage by including a nucleotide window around all indels symmetrically. Chaux, Messer, and Arndt 2007 proved that the frequencies of inserted and deleted amino acids differ from background amino acid frequencies in the human proteome by simulating the indel phases and comparing with the observed mammalian data. Hu and Ng 2013 proposed a decision tree algorithm within SIFT to predict the functional effects of amino acids around all indel phases. Zhao et al. 2013 developed a support vector machine-based method (DDIG-in) to discriminate between disease-causing and neutral indel phases from the 1,000 Genomes Project.

Additionally, insertions and deletions are two different mutational processes. Kvikstad et al. 2007 proposed that insertions are more likely caused by recombination, while deletions are mostly associated with replication-related features, such as DNA polymerase errors and incorrect DNA repair. Taylor, Ponting, and Copley 2004 found an excess of deletions over insertions particularly for the rat lineage compared with other mammals. Lynch and Walsh 2007 suggested that there was an overall deletion bias in prokaryotes and insertion bias in most eukaryotes due to the predominance of mobile-element activities. Sjödin, Bataillon, and Schierup 2010 found a lower frequency of deletion segregation in humans, providing evidence that deletions were under stronger purifying selection than insertions.

In this research, I converted the idea of sliding-window method from previous papers into a computational language (R), and identify all three indel phases. Our method can generate a local optimal alignment by sliding the gaps from each independent window. Unlike current codon-aware aligners that only work with phase 0 indels, this supports all three indel phases. Besides, our results demonstrate that the proportion of three indel phases is strongly correlated with the purifying selection strength in coding regions between focus species and within insertions and deletions separately. Hence, I strongly suggest that including indel phases to generate better alignments, which helps reduce the inflation of estimates of sequence divergence and lead to a more accurate inference of the tree between phylogenetic-related species (Kapli, Yang, and Telford 2020).

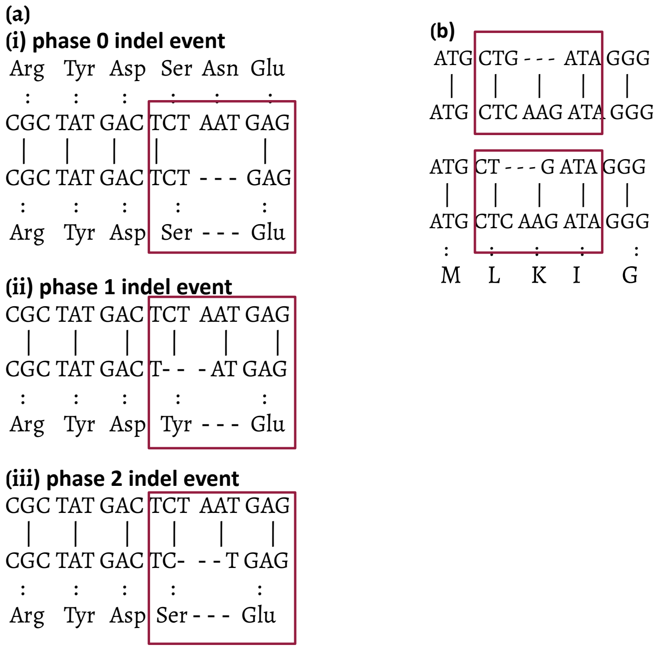


Figure 3.1: **The Description of Three Different Indel Phases |** a. Phase 0 and phase 2 indel events do not change the adjacent amino acid (Ser, Glu), but phase 1 indel event changes the Ser to Tyr at the left edge of the gap. b. Misalignment due to the limitation of the current codon-aware aligners. (top) phase 0 alignment from the aligner (MAFFT), (bottom) phase 2 alignment from our sw method.

3.2 Materials & Methods

3.2.1 Species Choice

I selected *Mus musculus*, *Rattus norvegicus*, and *Homo sapiens* as my species dataset. The main reasons were twofold: 1) they had a well-annotated genomic region and high-quality sequence data. 2) They had an appropriate evolutionary branch length (Mouse and rat last shared a common ancestor $\sim$ 16 million years ago (Springer et al. 2003)), because a shorter branch length means the number of indels might be insufficient, and a longer branch length could lead to erroneous alignments. Also, I included an outgroup species to help distinguish from insertion to deletion events, through a maximum parsimony method (Saitou and Imanishi 1989). The outgroup species is a more distantly related species that serves as a reference group which guarantees that the least number of changes occurred within the phylogenetic tree through the maximum parsimony method.

3.2.2 Homology Call

In order to study the indels between species, I found a series of homologous IDs of genes and transcripts from the Ensembl database. By using the ensembl API (https://rest.ensembl.org/), a set of 14547 homologous genes were extracted. Every gene was identified in each of the human, mouse, and rat (1:1:1 orthologs). Next, I used two main tags (ortholog_one2one, with_ccds) to filter out the gene IDs with obscure orthologs and lack of coding domain sequences (CDS). Then I specified the tag of is_canonical==1 to extract the canonical transcript from each gene because one gene could have multiple transcripts (Canonical means the longest transcript in Ensembl). And I generated the fasta files of CDS from canonical transcripts.

3.2.3 Sequence Alignment

I translated all CDS into amino acid using Biostrings::translate in R and specified the parameter of if.fuzzy.codon=solve. Then all fuzzy codons (i.e codon with IUPAC ambiguities) that were translated non ambiguously to an amino acid or stop codon (*) would be translated, and ambiguous fuzzy codons would be translated to X. And I conducted a global alignment of the protein sequences via MAFFT (multiple sequence aligner), since MAFFT was comparatively fast and required no phylogenetic tree beforehand (Katoh et al. 2005). Next, I reversely mapped the alignments back into CDS based on the amino acid to codon ratio (1:3).

3.2.4 Quality Control

Quality control is indispensable in every step of the data extraction process. After downloading the CDS, I filtered out any gene pairs in which the length of sequence were not multiple of three, containing early stop codons, and extra 3' 5' UTR. The last one required the download of a human GTF file (GENCODE) and removed any transcripts with cds_start_NF and cds_end_NF tags. Then I measured the alignment similarity of every gene pair in nucleotide and amino acid level between species. The result of the similarity distribution plot (Figure 3.2) demonstrates the high quality of our aligning data, because the similarity score between mouse and rat were generally higher than the rest of species pairs, corresponding to the phylogenetic branch length. The file size change of each quality control step can be found in APPENDIX B.1.

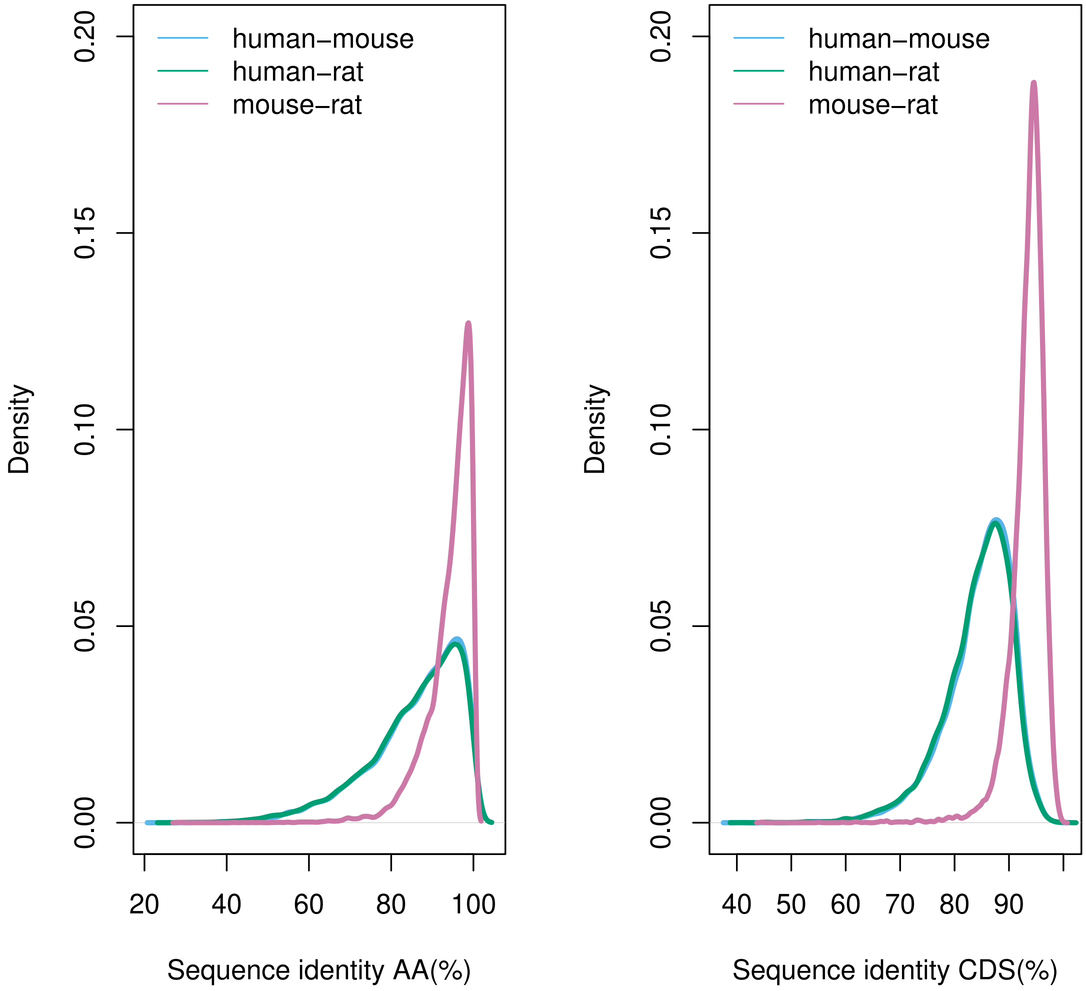


Figure 3.2: **Similarity Distribution of AA and CDS Sequences in Pairwise Species.** | Left figure represents AA sequence and right figure represents CDS.

3.2.5 Sliding Window Method

Sliding-window (sw) method was capable of modifying the current alignments by updating the indel phases. The indel length followed a zeta power-law or geometric distribution, and the existence of longer indels was much less frequent in coding sequences (Cartwright 2009a). Hence, I only select shorter indels of {3, 6, 9, 12 nts} because longer gaps were considered as a combination of short indels together which was unlikely to be influenced by a single evolutionary force (Ex. purifying selection). The sw method treats each gap as a dynamic unit, allowing it to slide from left to right progressively, one nucleotide at a time until reaching the edge of the window. The window size was the same as the gap length. After the process of window sliding, 2n alternative alignments would be generated when the window size was n (Fig 3.3a).

In order to find the optimal alignment, the similarity score of each alternative alignment was calculated. Figure 3.3a displayed the original (MAFFT) plus alternative alignments of an indel event, among which the highest similarity score was 83.33 %. And if multiple optimal scores appeared, that gap had no single best indel phase and would be marked out as + sign.

Moreover, sw method required all of the gap units to be mutually independent and two rules were created for that purpose: 1) any adjacent gaps of the same sequence cannot be too close to merge into a single gap during the sliding process (Fig 3.3b upper). 2) Any adjacent gaps of different sequences cannot be too close to overlap the same aligning space. (Fig 3.3b lower). All failed gaps were marked out as = sign. Additionally, I found that small-sized windows (3 or 6 nt) were supposed to include more gaps due to the extra space generated between each gap unit. Large-sized windows (9 or 12nt) led to a less amount of gaps, while having a higher chance of generating a single optimal alignment due to the extra nucleotides. Hence, making a good choice of the window size was important for the accuracy of the sw method.

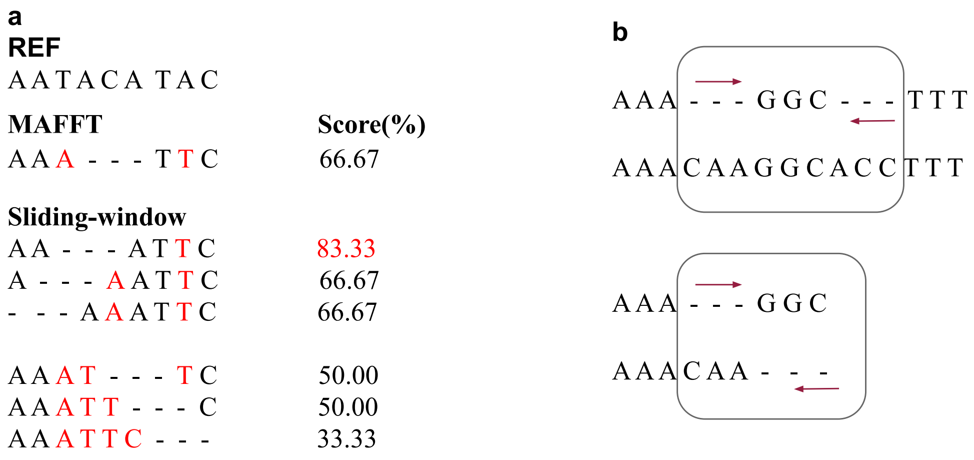


Figure 3.3: **The Application Rules of Sliding-window Method.** | a. Find an optimal alignment via the sw method. b. Upper alignment is an example of 1st rule violation. Two gaps are very close on the same sequence that can merge into a single gap during the dynamic sliding process. Lower alignment is an example of 2nd rule violation. Two adjacent gaps are on separate sequences that might overlap the same aligning space during the sliding process. (a. arrow – the moving direction of gaps. b. REF – reference sequence; MAFFT – Mafft alignment; sliding-window – alternative alignments generated by this method, Score – similarity score based on the percentage of number of matches).

3.2.6 Identify Three Indel Phases

All qualified indel events were recorded and distinguished after the sliding window fixation. By taking account of the outgroup species sequence, I was capable of separating from insertions to deletions through the maximum parsimony method. Besides, I calculated the proportion of non-synonymous and synonymous substitutions from phase 1 and phase 2 indels and made a bar plot.

3.2.7 Measure of Power for SW Method

To test the performance of the sliding window method, I calculated the positional difference of gaps between before and after sw method fixation. The original gap locations were generated by the MAFFT aligner, which was assumed to grab the most true gap locations for its speed and accuracy (Katoh et al. 2005). And I generated a positional difference distribution plot for all gaps, in which the range of distribution is from zero to a defined window size.

I assumed that the distribution bias of the positional difference plot was caused by the systematic bias within Mafft. To investigate further, 1) I extracted all surrounding codons of each window and summarized them into 20 distinctive empirical codon model (ECM) subsets based on the codons of similar physico-chemical properties (Kosiol, Holmes, and Goldman 2007); 2) I reversed the coding sequences and realigned them using the same method.

3.3 Results

3.3.1 Indel Phase Proportions

Figure 3.4 displays the genome-wide proportion of the indel phases. The upper left histogram shows the indel phase proportion across four different gap lengths in the window size of three nucleotides. Regardless of the window sizes, I find a consistent trend of the proportion of indel phases – the proportion of phase 0 indels is greater than the proportion of phase 2 indels, greater than the proportion of phase 1 indels.

Figure 3.5 displays the proportion of non-synonymous vs synonymous indels from phase 1 and phase 2 events. The proportion of non-synonymous substitutions are much higher in phase 1 indels (about 95%) than phase 2 indels (about 30%) across four different window sizes. And the proportion of synonymous substitutions are much lower in phase 1 indels (about 5%) than in phase 2 indels (about 70%).

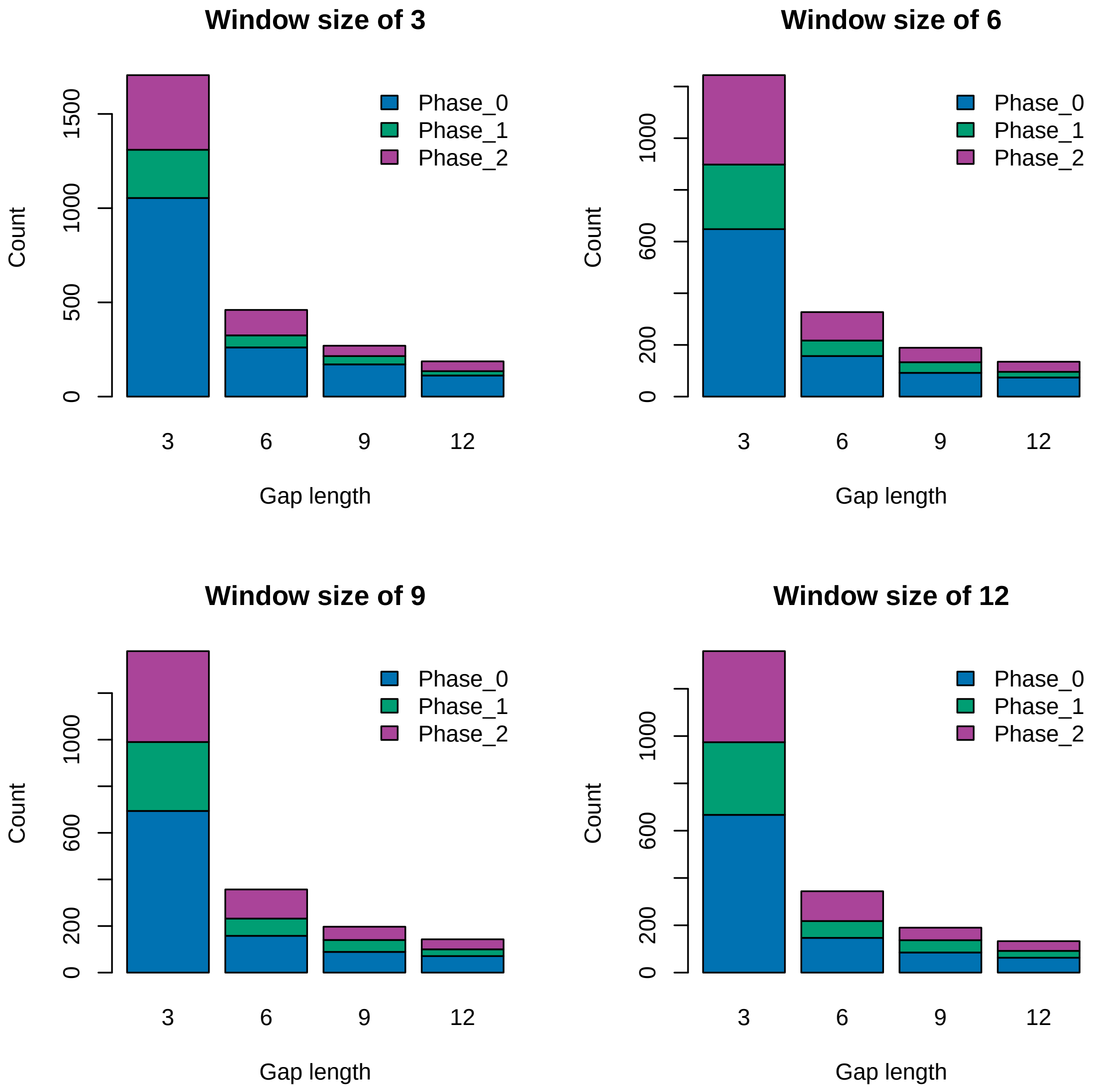


Figure 3.4: **The Proportion of Indel Phases of Different Gap Lengths across Four Window Sizes.** | (X-axis is the gap length, Y-axis is the count of each indel phase).

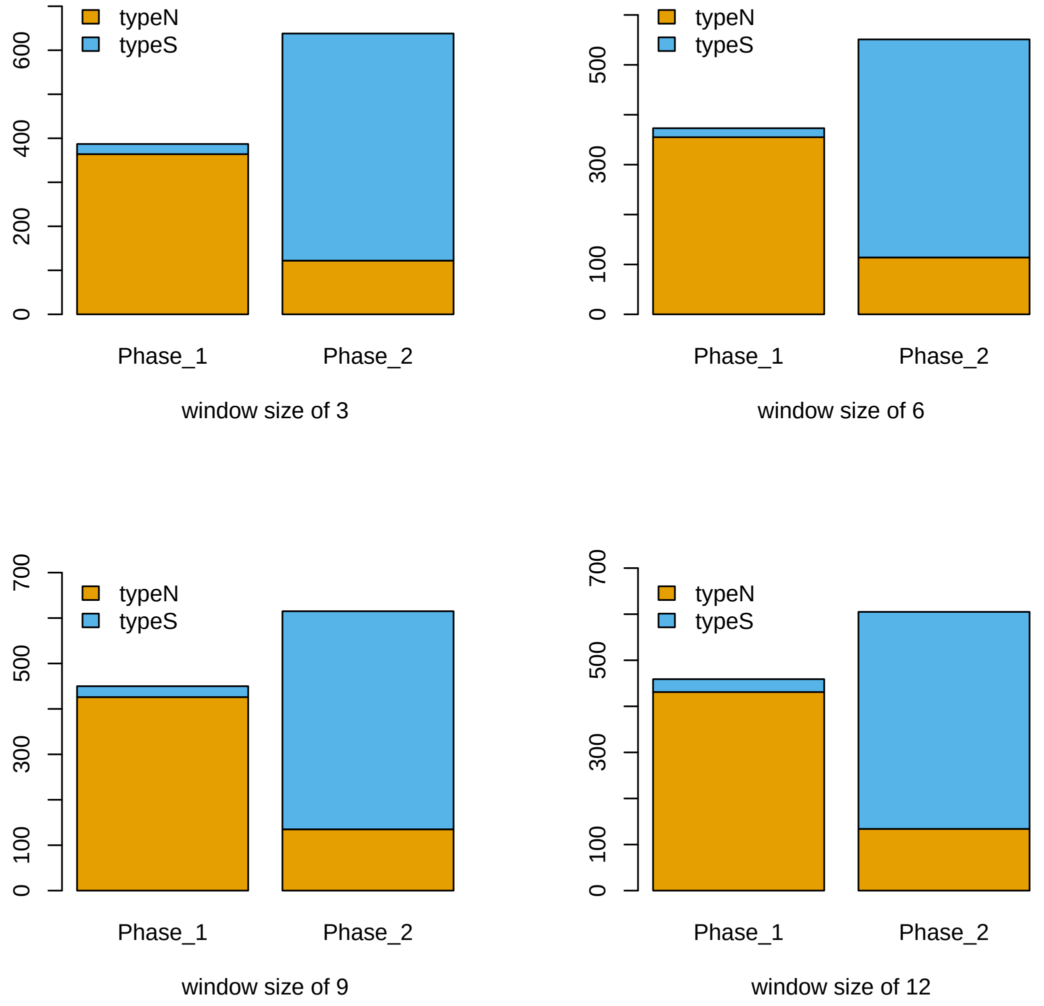
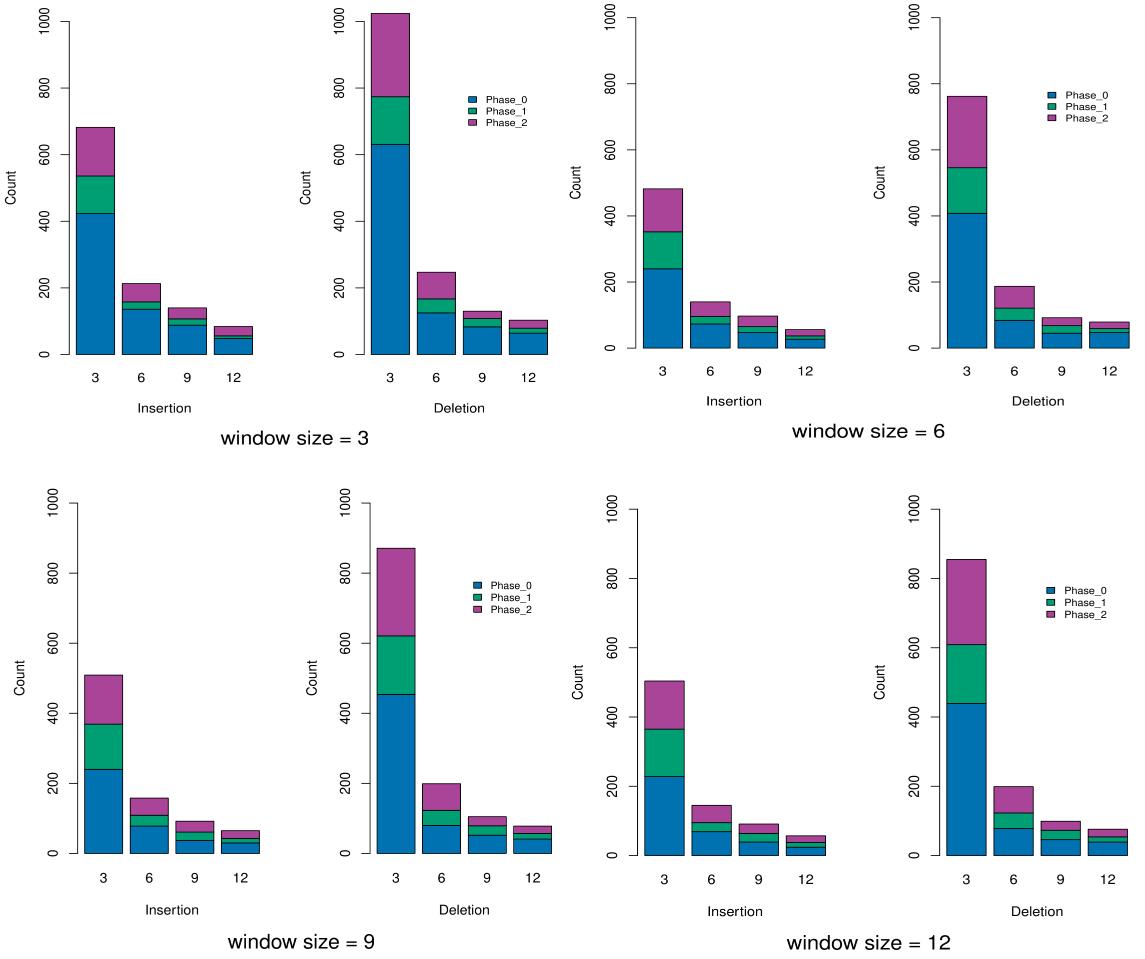


Figure 3.5: **The Proportion of Synonymous vs Non-synonymous Indels within Phase-1 and Phase-2 Events across Four Window Sizes.** | (X-axis is the indel phases, Y-axis is the count of typeN or typeS indels).

Figure 3.6 displays the genome-wide proportion of the indel phases of insertion and deletion events separately. Regardless of the window size, there are more deletion events than insertion events within coding regions by looking at the count (y-axis) of the histogram — referred as deletion bias. Although I find no significant difference of phase proportions between insertion and deletion events, except for phase 1 proportions of window size of 9 and 12, in which their p value of non-parametric chi-squared proportion test are less than 0.05 (APPENDIX B.2).



window size = 3

window size = 6

window size = 9

window size = 12

Figure 3.6: **The Proportion of Phases of Different Gap Lengths between Insertion and Deletion Events across Four Window Sizes.** | (X-axis is the gap length, Y-axis is the count of each insertion or deletion phase).

3.3.2 Positional Difference of gaps between SW and MAFFT

Figure 3.7 displays a bell-shaped similar distribution of positional difference of gaps across four different window sizes. The frequency density decreases when a gap moves away from its original position zero, which suggests that the sliding window method is trustworthy. However, the distribution of positional difference is a bit biased due to the existence of multiple peaks, especially at the right edge of the windows, such as position +3 and +6. However, such bias is reduced in a larger window size (9,12 nts), suggesting that including more nucleotides information into the windows helps the SW method to find an optimal alignment.

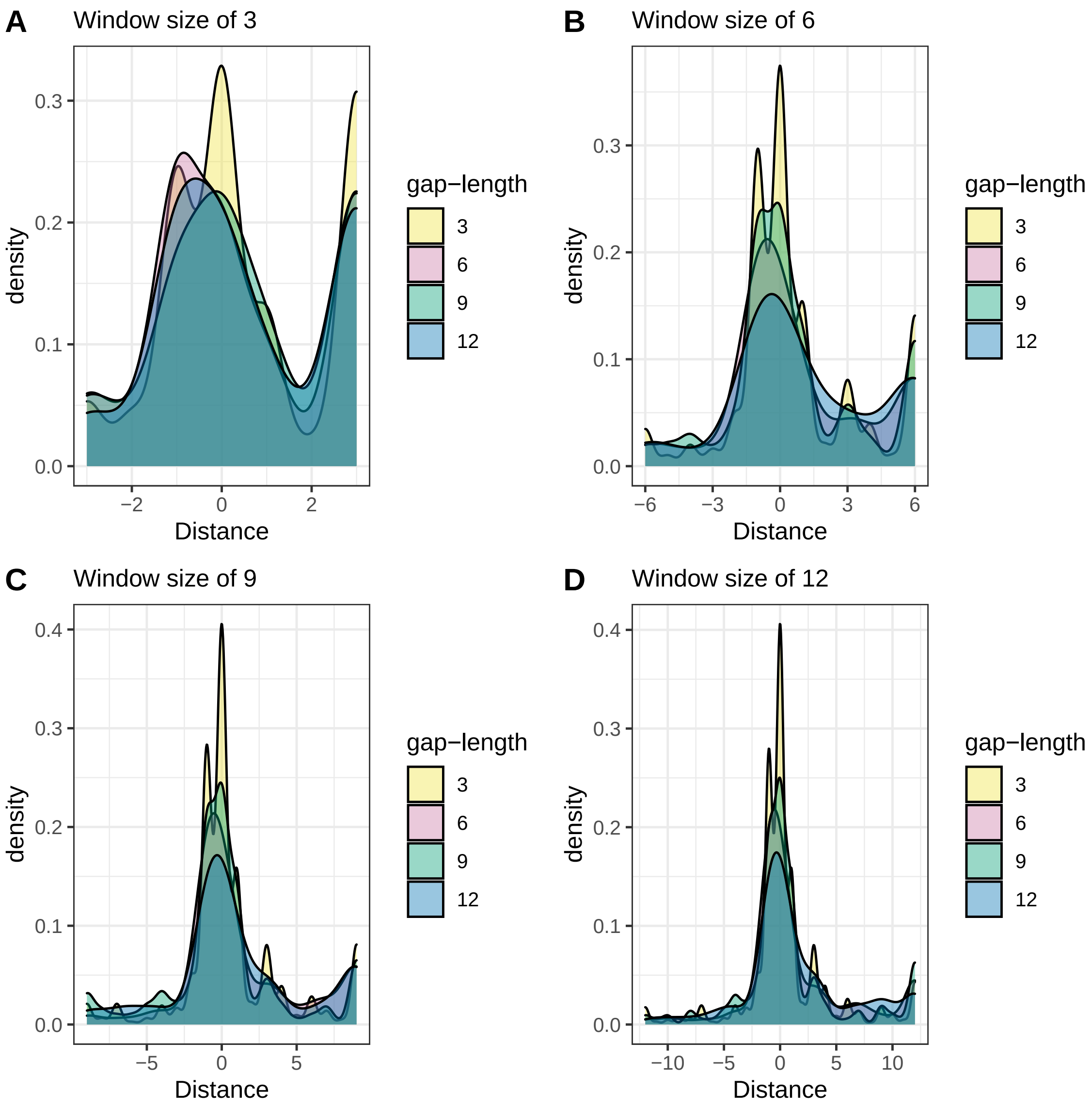


Figure 3.7: **Positional Differences of Gap Lengths across Four Window Sizes between the Mafft and SW Method.** | Each position on the x-axis represents the relative nucleotide distance between the original gap position generated by MAFFT and the new position generated by the sw method. All MAFFT gaps are at the position 0 and the sw gaps can be any positions on the x-axis. The y-axis counts the frequencies of each position.

3.4 Discussion

3.4.1 Indel Phase Analysis

The proportion of indel phases suggests that the protein coding regions undergo effective purifying selection in focal species. Since phase 0 indels only change the number of codons, phase 1 and phase 2 indels can change a single surrounding amino acid that encodes. I find that the phase 1 indels are more likely to cause an amino acid substitution due to the less codon degeneracy, indicating that they are the most deleterious events among three. The results from Figure 3.5 also suggest that the phase 1 indels have a higher chance of resulting in non-synonymous codon substitutions compared with the phase 2 indels.

Although insertions and deletions are two different processes that are caused by different mechanisms, I find a similar trend of phase proportions (phase0>phase2>phase1) in both. Although our results provide no evidence that either insertions or deletions are under different selective strength at a significant level, there is an overall deletion bias across four different window sizes within coding regions, which are also discussed in many previous papers (Gregory 2004; Loewenthal et al. 2021; Montoya-Burgos, Boursot, and Galtier 2003).

3.4.2 Positional Difference Distribution Bias

The probability of finding gaps decreases when the distance between original Mafft alignment and sliding-window fixed gaps gets further. However, the edge bias from positional difference distribution is asymmetric. At first, I assumed that this distribution asymmetry was because Mafft preferred to place ambiguous gaps in a specific position such that the surrounding codons of the similar physico-chemical properties aligned to each other. However, this assumption was rejected because no obvious empirical codon model (ECM) patterns existed within the aligning windows. Therefore, I preferred to consider this distribution bias was due to the systematic bias within Mafft. For example, sequence context might be the main reason that affects the gap position choice in Mafft, especially when there are many repetitive residues.

Chapter 4

EM TRAINING OF AN INDEL-PHASE PROBABILISTIC MODEL

4.1 Introduction

Enormous amounts of data has been generated in the last decades due to the advancement of sequencing technology and bioinformatic tools. The corresponding models and algorithms are also developed to provide solutions to biological and bioinformatic problems. However, probabilistic-based models for indels are far less developed compared with the substitution models. In chapter 3, while I applied a sliding-window method to do the post-aligning indel phase fixation, the proportion of indel phases cannot represent the actual indel rates over the evolutionary time. To solve this, I developed an indel-phase probabilistic model that describes both substitution and indel processes. And I developed a novel EM training method that was capable of estimating the instantaneous indel rates of different phases, indel length distribution, nucleotide exchangeabilities, non-synonymous selective coefficient, and branch length. The simulation results demonstrated the feasibility and accuracy of our EM algorithm for training the indel-phase model.

Many previous papers study the estimation of indel rates and indel length distribution. For instance, Metzler et al. 2001 reduced the alignment variability by sampling sequence alignments and estimating the indels from their joint posterior distribution. Holmes 2005 proposed an EM method of estimating the indel parameters derived from the TKF91 model in multiple sequence alignments. Cartwright 2009b presented an EM algorithm built on a pair hidden Markov model handling indels in neutrally evolving DNA sequences. In those studies, while the indel rates were derived from the observed gap counts between alignments, a single gap actually reflected the possibility of multiple indel events and misled the computation of sequence likelihood.

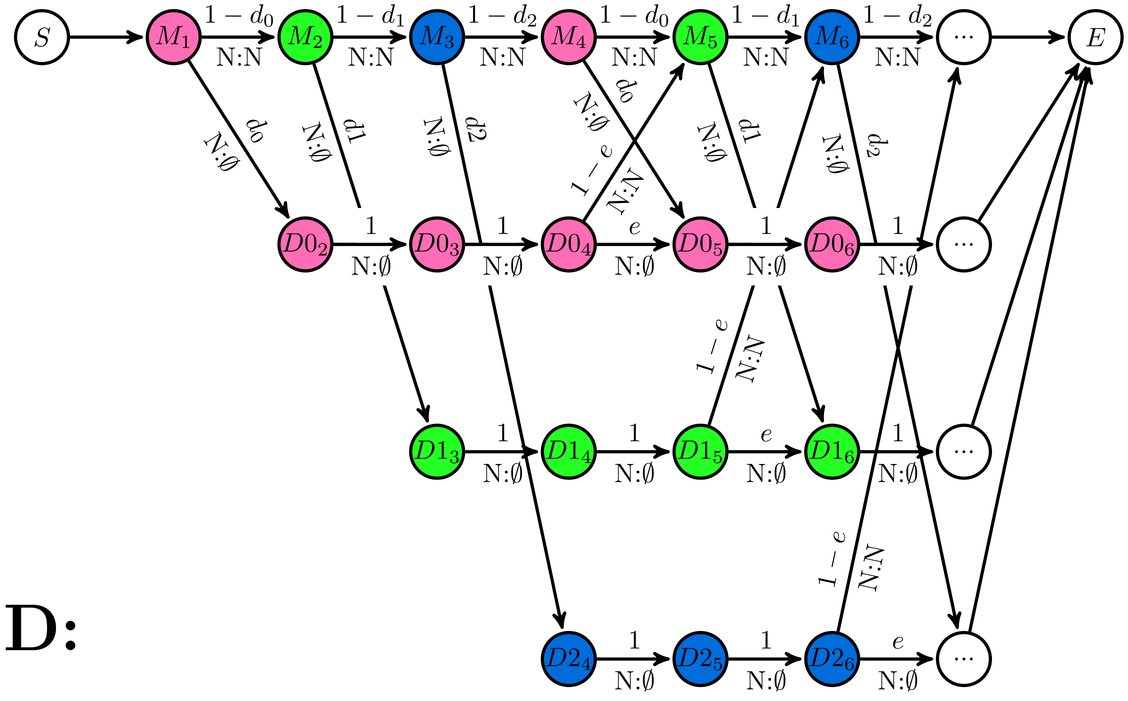
Two main distributions were utilized for describing the indel length: geometric and Zipfian (one of a family of related discrete power law probability distributions). Although there was evidence that Zipfian distribution fitted better with the biological datasets in both coding (Benner, Cohen, and Gonnet 1993) and non-coding regions (Saitou and Ueda 1994), geometric distribution was computationally tractable and easily calculated. Many researches accentuated the difference between insertions and deletions(Metzgar et al. 2002; Tao et al. 2007). They utilized pseudogenes or outgroups to distinguish insertions from deletions, which might increase the memory cost and restrict the data volume. However, due to the lack of outgroup sequence in my research, I randomly selected one of the two sequences as the ancestor, and the other one was automatically assigned as the descendant when calculating the sequence likelihood,.

The existing methods of training the probabilistic models for indel evolution normally assume a single indel rate for all gaps between sequences, equal length distributions between insertions and deletions; and introduction of alignment bias (Cartwright 2009b; Holmes 2005; Lunter 2007). Here, I introduced several improvements in our indel-phase model study: 1) our indel-phase model proposed three rates of each indel phase instead of one rate for all indel events; 2) this model is richer because of the inclusion of the GTR+MG94 model for describing the substitution process; 3) During the EM training of the indel-phase model, the alignment bias was greatly removed through applying an importance sampling method.

4.2 Methods and Materials

4.2.1 Indel-phase Model

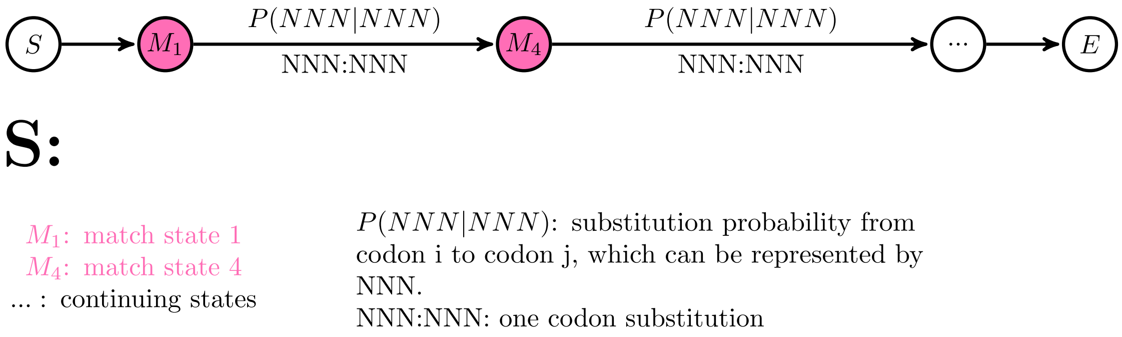
Indel-phase model The indel-phase model was designed as a Moore machine with a finite number of states, whose current output was determined by its current state through a transition function (Giantamidis, Tripakis, and Basagiannis 2021). In my research, I splitted our alignment into three independently ordered processes — deletion (Fig 4.1 a), substitution (Fig 4.1 b), and insertion (Fig 4.1 c), where each has their own Moore machinery pathway and transitional weight between states. The evolutionary order of deletion → substitution → insertion makes our model symmetric, which means the calculation of the aligning likelihood stays invariant, unnecessary to know which one of the sequences is ancestor/descendent. Because the synonymous/non-synonymous mutation type of a codon stays invariant no matter which sequence to start for calculating the likelihood.



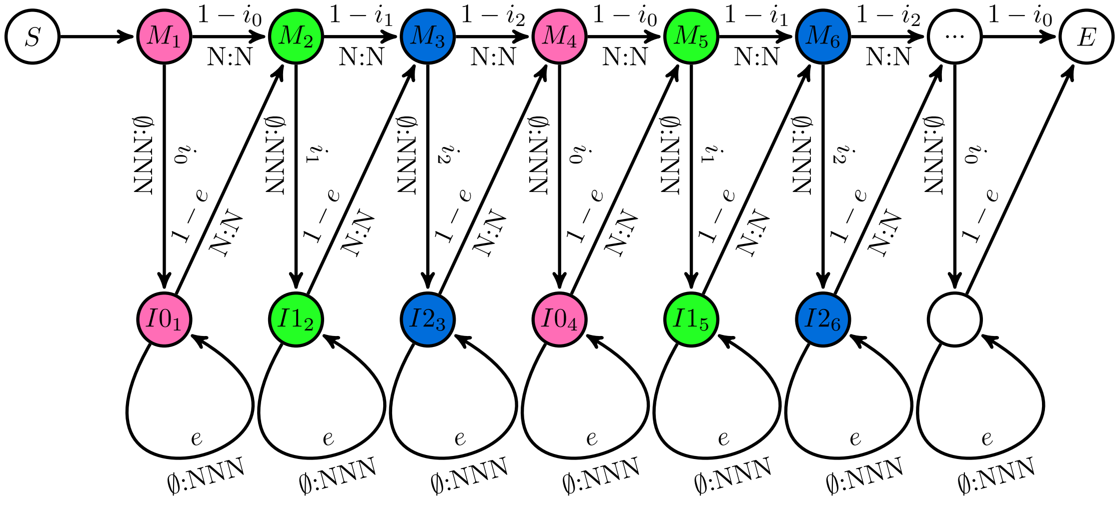
S: Start
M_1: match state 1
M_2: match state 2
M_3: match state 3
$D0_i$: phase-0 deletion of state i
$D1_i$: phase-1 deletion of state i
$D2_i$: phase-2 deletion of state i
... : continuing states
E: End

d_0:	phase-0 deletion opening weight
d_1:	phase-1 deletion opening weight
d_2:	phase-2 deletion opening weight
e:	deletion extension weight in the unit of codon
N:∅:	one nucleotide deletion
N:N:	nucleotide to nucleotide match

(a) For deletions, there are three phases of deletion states ($D0_i, D1_i, D2_i$) corresponding to each position of the codon, where i denotes the position of the matching state respectively. If the sequence is in the deleting state, it can either keep deleting the gaps in the unit of codon (the extending probability is e) or exit the gap state and go to the next matching state (the transition probability is $1-e$)


S
M_1
$P(NNN|NNN)$
NNN:NNN
M_4
$P(NNN|NNN)$
NNN:NNN
$\cdots$
E
S:
M_1: match state 1
M_4: match state 4
... : continuing states
$P(NNN|NNN)$: substitution probability from codon i to codon j, which can be represented by NNN.
NNN:NNN: one codon substitution

(b) For substitutions, the probability of codon i change to codon j equals the codon substitution matrix $P = e^{Rt}$, where R is the instantaneous rate matrix derived from the GTR+MG94 model and t is the evolutionary branch length between species. The figure above displays that the substitution jumps from matching state 1 to 4 directly, suggesting that our substitution matrix is in the unit of codon instead of nucleotide.



I:

$I0_i$: phase-0 insertion of state i
$I1_i$: phase-1 insertion of state i
$I2_i$: phase-2 insertion of state i
... : continuing states

i_0: phase-0 insertion opening weight
i_1: phase-1 insertion opening weight
i_2: phase-2 insertion opening weight
e: insertion extension weight in the unit of codon
$\emptyset$:NNN: one codon insertion
N:N: nucleotide to nucleotide match

(c) For insertions, there are three phases of insertion states ($I0_i$,$I1_i$,$I2_i$), corresponding to each position of the codon. If the sequence is in the inserting state, it can either keep inserting the gaps in the unit of codon (the extending probability is e) or exit the gap state and go to the next matching state (the transition probability is $1-e$).

Figure 4.1: Moore Machine Pathway of the Indel-phase Model

Of note, I must assign an end edge transitional weight for deletion, substitution, and insertion machines respectively. If a codon chooses to end in a deletion state, suggesting no further action is needed and the transitional weight is 1. If a codon chooses to end in a matching state, suggesting that no phase-0 insertion action is allowed to occur, which means the transitional weight is $1-i_0$. If a codon chooses to end in an insertion state, suggesting that the machine must exit the current inserting bubble with a transitional weight of $1-e$.

6nt alignment example Suppose that there is a 6 nt alignment:

$$\begin{array}{ccccccccc} A & - & - & - & A & A & A & A & A \\ A & A & A & A & A & - & - & - & T \\ * & & & & * & & & & * \end{array}$$

where $*$ represents the substitution codon. In order to calculate the likelihood of this alignment, I need to split the calculation into three processes. For deletions, the machinery pathway is $1 \rightarrow 2 \rightarrow 3 \rightarrow D2_4 \rightarrow D2_5 \rightarrow D2_6 \rightarrow 7$, with a transitional probability of $(1-d_0)(1-d_1)(d_2)(1-e)$. For substitution, the machinery pathway is $1 \rightarrow 4$, with a substitution probability of $P(AAT|AAA)$ that is calculated from the matrix exponential of the MG94+GTR model. For insertion, the machinery pathway is $1 \rightarrow 2 \rightarrow I1_2 \rightarrow 3 \rightarrow 4$, with a transitional probability of $(1-i_0)(i_1)(1-e)(1-i_2)$. The total likelihood of this sequence is the sum of the above three likelihoods together.

4.2.2 Indel Rates

Abstractly, we treat gaps as one or more overlapping or neighboring indels assembling together, whereas an indel is a specific evolutionary event that adds or deletes residues from a sequence. Without ancestor lineage reference, a gap in the alignment of one sequence may be caused by either one or more deletions in that sequence or one or more insertions in the other sequence. Popular methods like maximum parsimony tend to reassemble adjoining single indel events so that the alignment scores can be maximized, which underestimates the true indel rate along an evolutionary branch Knudsen and Miyamoto 2003. To simplify the model, I treat a single gap from the alignments as a single indel event. Next, the model assumes both substitutions and indels follow a continuous-time Poisson process. If the chain does not terminate, the probability of the next state of I or D is $g_i = 1 - e^{r_i t}$, where r_i is the instantaneous indel rate of phase i relative to substitution rate, t is the branch length between pairwise sequences measured in expected number of substitutions per base pair. SO the $e^{r_i t}$ represents the probability that no indel events have occurred.

4.2.3 Indel Length Distribution

I assumed the length of insertions and deletions followed different geometric distributions in our model. The maximum likelihood estimate of gap extension weight e_i, e_d is calculated by $1 - 1/\hat{g_s}$, where $\hat{g_s}$ represented the average gap size of insertions and deletions separately (dividing the sum of gap length by the total number of gaps).

4.2.4 Codon Substitution Model

The substitution matrix R_{ij} was represented by the MG94 +GTR codon substitution model (Muse and Gaut 1994; Tavaré et al. 1986), and the details about the matrix construction can be found in the methods of chapter 2.

4.2.5 Coati-sampling Method

To fulfill the purpose of aligning coding sequences as codons but also allows indels within codons, I select the Coati-sampling aligner that is built on the Forward algorithm within the context of Hidden Markov model developed by our lab (https://github.com/jgarciamesa/coati). This aligner has no restrictions for gap locations, options for forcing the gap length to be multiple of three nucleotides, and more importantly, allows to generate many sample alignments for a single pairwise sequence with a sequence likelihood.

4.2.6 EM Method

I used the EM method to find the local maximum likelihood estimates of parameters. The E-step generated a function for the expectation of the log-likelihood:

$$
\begin{aligned}
Q(\theta|\theta_t) = & \sum_i \sum_j D_{ij} log(P_{ij}) \\
& + N_0 log(g_0) + N_1 log(g_1) + N_2 log(g_2) \\
& + M_0 log(1-g_0) + M_1 log(1-g_1) + M_2 log(1-g_2) \\
& + (L_i - N_i) log(e_i) + N_i log(1-e_i) + (L_d - N_d) log(e_d) + N_d log(1-e_d) \\
& - \sum_1^n log(\Pi_k) - \sum_1^m log(\Pi_k)
\end{aligned}
\tag{4.1}
$$

where D_{ij} is the codon substitutions $i \rightarrow j$ observed from data, $P_{i,j}$ is the codon substitution matrix from i to j ($1 <= j <= 64$), which is calculated as the exponential of the rate matrix R_{ij}. N_0, N_1, N_2 is the total number of gaps phase0, phase1, and phase2, g_0, g_1, g_2 is the gap openings of three phases. M_0, M_1, M_2 is the total number of non-gap phases, which equals the difference between codon lengths and number of gap phases. L_i, L_d are the total length of inserting gaps and deleting gaps, N_i, N_d are the total number of inserting gaps and deleting gaps, e_i, e_d is the gap extensions of insertion and deletion events. π_k represents the nucleotide frequency of ancestor and descendent sequences in which the lengths are n and m. I removed their likelihood because they stayed invariant in all of the aligning samples. In addition, the E-step has a time complexity of number of sequences times sequence length O(500*100). Hence, I developed a grouping method within the E step to speed up the performance: 1) Dividing the groups to ensure that each group has identical alignments. 2) recording the likelihood of each group alignment; 3) extracting the number of groups and group size.

The M-step maximizes that function with respect to the parameters θ via utilizing the summary statistics obtained from E-step.

$$\theta_{t+1} = \underset{\theta}{argmax} Q(\theta|\theta_t)$$

Where θ represents our parameter set $\{6\sigma s, \omega, \tau, g_i, e_i\}$. Gap opening weight g_i and gap extension weight e was solved by taking the first derivative of likelihood function Q and the value was $N_i/(N_i+M_i)$ and $1-1/(L/N)$ respectively. For practical purposes, I utilized the Nmkb derivative-free optimization method to obtain the estimates of substitution models within M-step. Finally, I used the root mean square error ($< 10^{-4}$) between the joint estimates and true parameters as the stop criterion for the EM process.

4.2.7 Importance Sampling

Doing the sum over all possible alignments for the indel-phase model is expensive due to the indels happening inside codons. Instead I utilized COATi to sample alignments and applied an importance sampling method to estimate the required statistics for the EM. Importance sampling is a statistical technique that is used to estimate the properties of a particular distribution without sampling from that distribution directly. (Kloek and Van Dijk 1978). In our research, two distinctive distributions are from the indel-phase model and coatiM model respectively. The indel-phase model is capable of calculating the likelihood of the alignment without actually generating the alignment samples. For example, I was able to obtain 100 samples from each unaligned pairwise sequence after the coati alignments, and calculated their sequence likelihood g(x). Then I generated a second likelihood value f(x) of the same alignments from our indel-phase model. The likelihood ratio of f(x)/g(x) is called the importance weight w, which is the key for correcting the distribution difference between two models. Because it always encourages a higher weight to frequently occurring aligning samples. For example, for a 100 alignment samples A_i, the average weighted number of gaps is $\hat{N}=\sum_i^n w_i N(A_i)/100$. Additionally, for the convenience of calculation, we used the normalized importance weight instead of the raw weight, because sometimes I can only compute w up to a proportionality constant.

4.2.8 Parameter Space

I need to assign an appropriate weight space for our true parameters $\{\pi, \sigma, \tau, \omega, g_i, e\}$.1) The four nucleotide frequencies π, six nucleotide exchangeabilities σ , and selective strength on non-synonymous substitutions ω, are generated via the same methods of chapter 2. This time I selected a smaller τ value to avoid the possibility of overlapping indels. 2) The gap opening weight are represented by the formula $1-e^{rt}$ with indel rate r, in which the indel rate is about 12-16 indels per 100 substitutions according to Cartwright 2009b. And the indel rates of three phases are mutually independent and within the range {0.05,0.15} when the substitution rate is normalized as 1. 3) The gap extensions are represented by the average gap size of $1-1/g_s$, where g_s is determined in the unit of codons instead of nucleotides. I selected a boundary of gap size {1,4}, ensuring that the gap extension weights were always positive, and the largest gap was the same as the size used in the sliding window method of chapter 3.

4.2.9 Gillespie Simulation

I need to simulate a series of true alignment based on my generated true parameters. Gillespie 1977 proposed an algorithm that was able to generate a statistically correct trajectory of a stochastic equation system, which has been applied on simulating the evolutionary process of a single DNA sequence (Sipos et al. 2011). The probability of a random event occurring equals the $r_i/(\sum_{i=1}^n r_i)$. Our gillespie algorithm separates all three processes (deletion, substitution, insertion) to make it easier to estimate parameters. Without separation there would be ambiguity when multiple events happen at one codon.

There are several key points about our gillespie algorithm: 1) since both mutations and indels follow a continuous-time Poisson distribution, I must use the same branch length τ for all three independent processes (deletion, substitution, and insertion). 2) Different indel phases can occur at different rates r0, r1, r2. 3) The indel lengths are a multiple of three nucleotides and follow a geometric distribution. Inserted nucleotides are drawn from the stationary distribution of nucleotide frequencies. 4) No substitutions will occur inside the inserted fragments. 5) The total number of inserting positions is n+1, if the sequence length were n. The first inserting position is on the left side of the first nucleotide, which is normally called the immortal link. The total number of deleting positions equals the sequence length n (Thorne, Kishino, and Felsenstein 1991). 6) Alignments are normalized such that insertions occur before deletions of the same indel phase. 7) The evolutionary order of our Gillespie simulation is deletion $\rightarrow$ substitution $\rightarrow$ insertion, because this is a symmetric model for calculating likelihood. It guarantees that the evolutionary track of a substitution process never gets wiped out because the mutational type (synonymous or nonsynonymous) of a codon surrounding an indel stays unchanged after the alignment. 8) The waiting time between each evolutionary event follows an exponential distribution, and the simulation process stops until the accumulated time passes our branch length τ.

4.3 Results

4.3.1 Gillespie Simulation Results

To establish a baseline for the performance of our method, we generated a qqplot for each parameter. Figure 4.2 displays the correlation between the true parameters and parameter estimates from gillespie simulated alignments. The diagonal line measures the accuracy between the true and estimated parameters. Since the Gillespie simulation only generates the true alignment, their differences are due to the randomness of the simulation process instead of any systematic error.

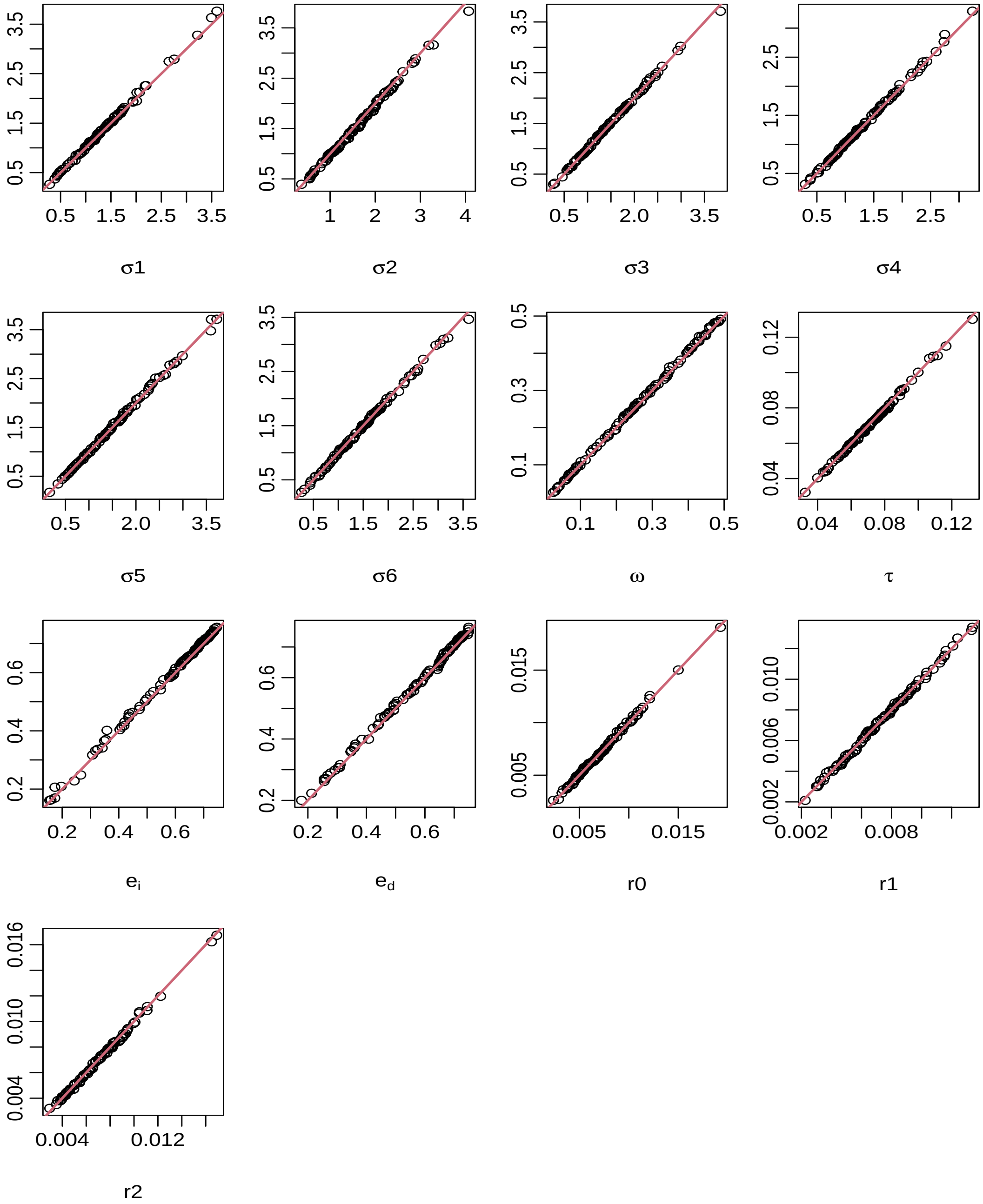


Figure 4.2: **QQplot of 13 Parameters via Gillespie Simulation.** | True parameter values are on the x-axis and parameter estimates are on the y-axis, including Six σs, τ, ω, gap extensions e_i, e_d, and three indel phase rates $r0$, $r1$, $r2$.

Figure 4.3 displays the distribution of error percentage between the true and estimated parameters across 100 simulations. I measured the error percentage as an absolute value of (true-est)/true and was always positive. The mean value of error percentage is extremely low as well as the variance. For example, the lowest and highest mean value of error percentage is 0.013 (τ), and 0.027 (r_0) (table 4.1).

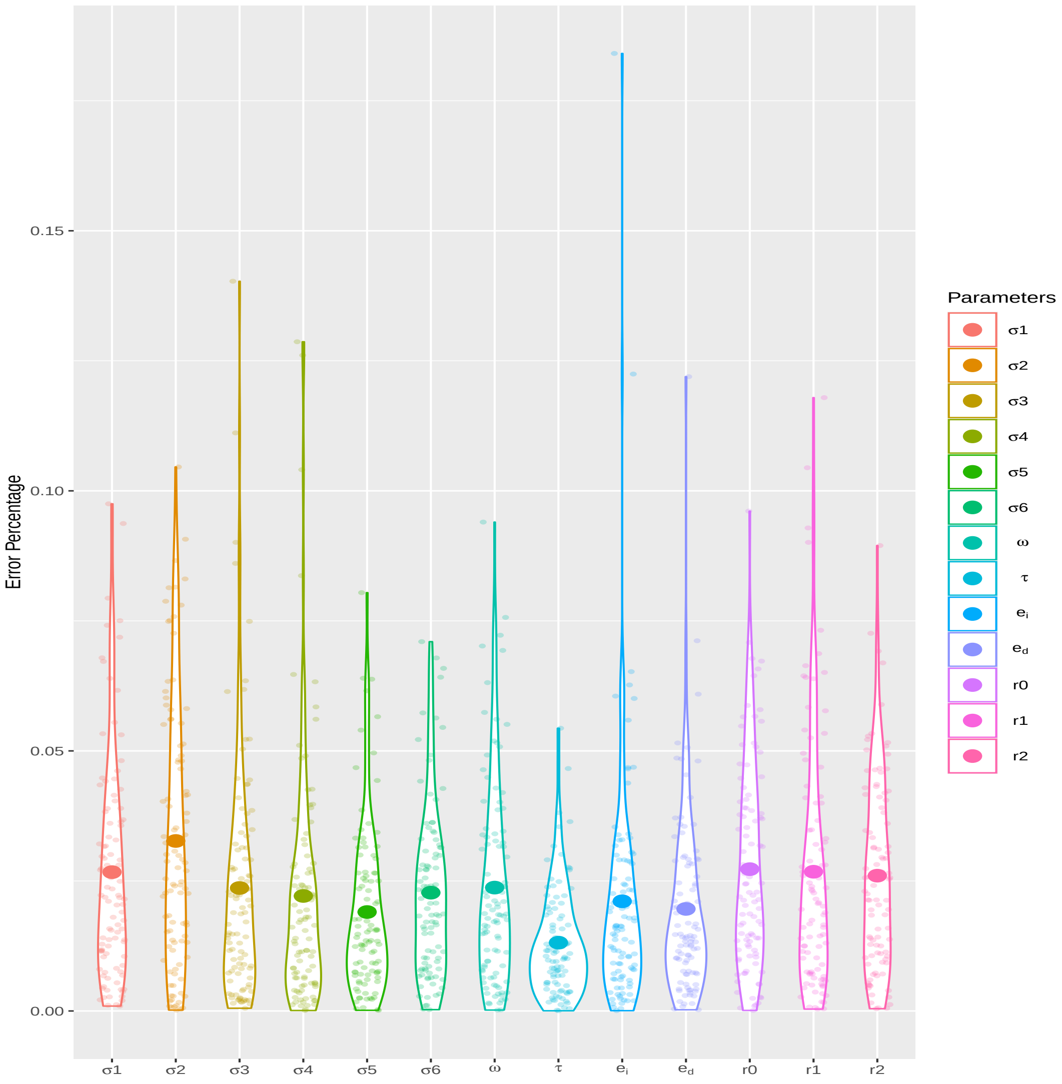


Figure 4.3: **Error Percentage Distribution Plot of 13 Parameters via Gillespie Simulation.** | Each color represents a parameter estimate.

	σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	e_i	e_d	r0	r1	r2
u	0.027	0.033	0.024	0.022	0.019	0.023	0.024	0.013	0.021	0.020	0.027	0.027	0.026
s	0.000	0.001	0.001	0.001	0.000	0.000	0.000	0.000	0.001	0.000	0.000	0.001	0.000

Table 4.1: **The Mean and Variance Table of Error Percentage of Parameters from Gillespie Simulation** | μ and s represent the mean and variance value.

4.3.2 EM + Importance Sampling Results

To examine the performance of our importance sampling method and the EM algorithm, I generated a qqplot for each parameter. Figure 4.4 displays the correlation between the true parameters and our parameters estimates . Almost every simulated point remains on the diagonal line. Since the importance sampling and EM method works on a series of unaligned sequences, the error percentage is considered as the combination of the randomness from Gillespie simulation and the sampling noise from our method. Theoretically, the more samples generated, the more accurate the parameter estimates are.

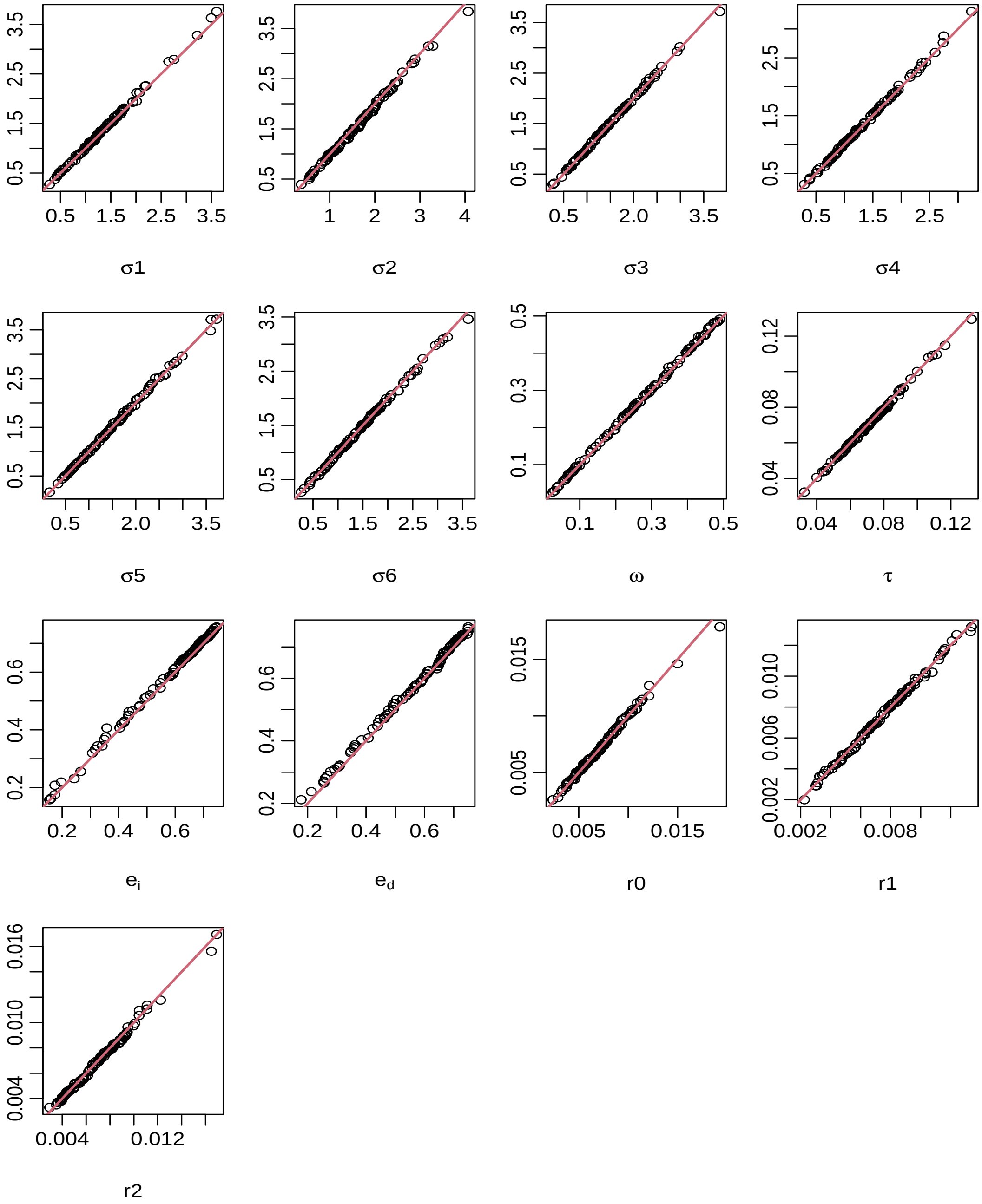


Figure 4.4: **QQplot of 13 Parameter Estimates via EM + Importance Sampling Method.** | True parameter values are on the x-axis and parameter estimates are on the y-axis, including Six σs, τ, ω, gap extensions e_i, e_d, and three indel phase rates $r0$, $r1$, $r2$.

Figure 4.5 displays the distribution of error percentage between the true and estimated parameters across 100 simulations. Similarly, the mean value and variance of error percentage for each parameter are extremely low. For instance, the range of the average error percentage value of 13 parameters is [0.013, 0.036] from τ and r_0 respectively (table 4.2.

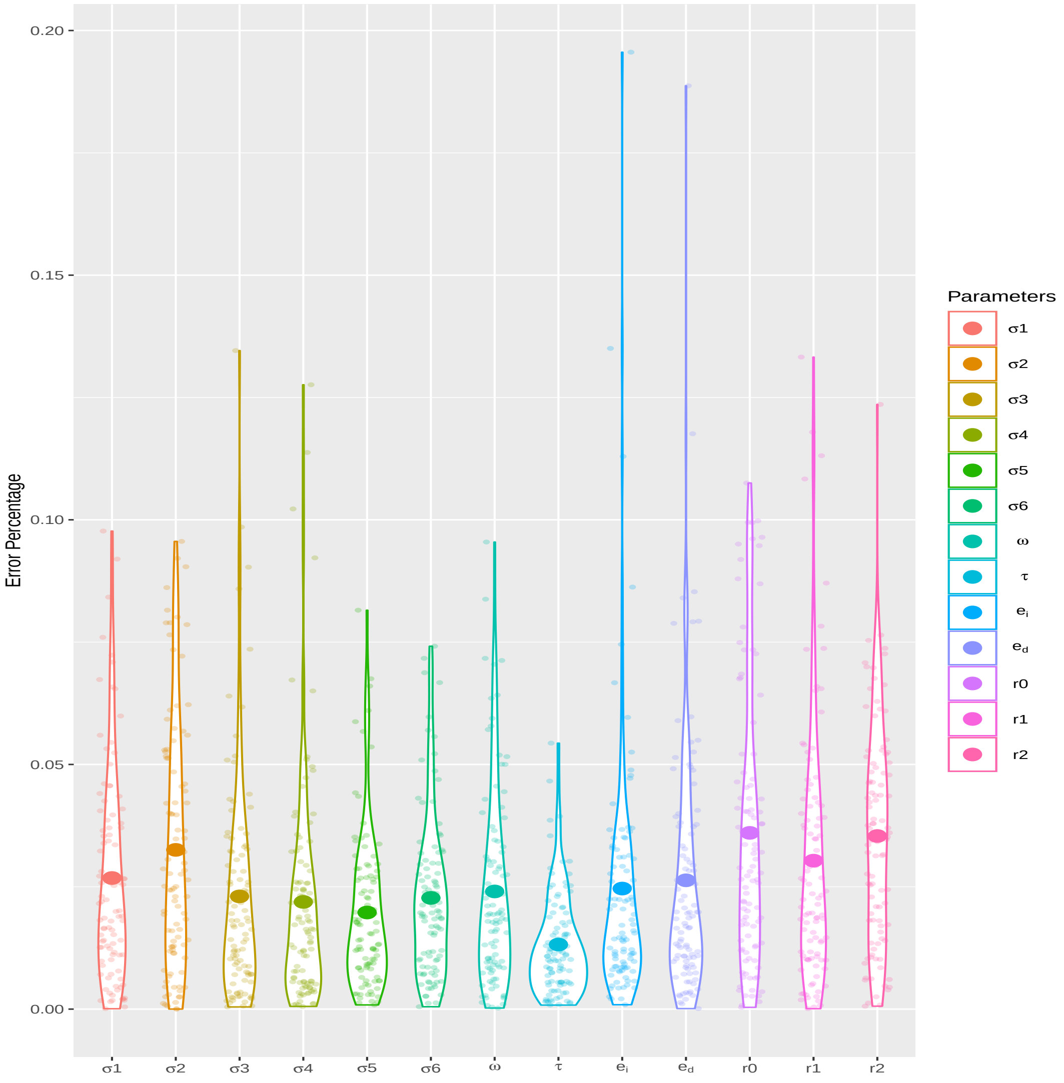


Figure 4.5: **Error Percentage Distribution Plot of 13 Parameters via EM + Importance Sampling Method.** | Each color represents a parameter estimate.

	σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	e_i	e_d	r0	r1	r2
u	0.027	0.033	0.023	0.022	0.020	0.023	0.024	0.013	0.025	0.026	0.036	0.030	0.035
s	0.000	0.001	0.001	0.001	0.000	0.000	0.000	0.000	0.001	0.001	0.001	0.001	0.001

Table 4.2: **The Mean and Variance Table of Error Percentage of Parameters from EM + Importance Sampling Method |** μ and s represent the mean and variance value.

4.4 Discussion

In order to understand the evolutionary process of substitution and indels together, I first conducted the Gillespie simulation to generate a series of simulated true alignments. Then I unaligned the true alignments and realigned them with the coati-sampler. Due to the complexity of the indel-phase model, I developed an EM + importance sampling method to train this model observed on the coati-sampling alignments, and measured the accuracy of the parameter estimates. The parameter estimates results demonstrate an excellent performance of the EM + importance sampling method of training my indel-phase model.

It is worthy of pointing out that the parameter has the lowest average value of error percentage is the branch length τ, and the highest one is phase0-indel rate r_0 in both Gillespie simulation and EM + importance sampling method. Because the substitution events occur much more frequently than the indels during the same evolutionary time. And the branch length τ actually measures the number of substitutions per site that is expected to be much higher than the number of indels per site. Similarly, the mean of error percentage of substitution parameters $\{6\sigma s, \omega, \tau\}$ are lower than the indel parameters $\{r_0, r_1, r_2, e_i, e_d\}$ during the same evolutionary time. Those results suggest that the more data collected, the less the noise would be.

In the future, I can vary the indel-phase model to obtain better performance. For example, I can utilize the zeta distribution instead of the traditional geometric distribution for indel length since Cartwright 2009b mentioned that the zeta power-law model might be a better fit based on many empirical studies. Additionally, I can convert our R scripts to C++ scripts to improve the computational speed for the EM training process, allowing us to increase the sample sizes to obtain more accurate parameter estimates.

Chapter 5

INDEL PHASE ANALYSIS VIA THREE PAIRWISE ALIGNMENT METHODS

5.1 Introduction

The identification and understanding of mechanism behind indel phases is still an unexplored area in molecular and computational biology. Utilizing the correct indel and substitution models more accurately reflects the evolutionary process and leads to fewer biases in phylogenetic inference (Arenas 2015). In this chapter, I apply three alignment methods to do our indel phase analysis, including the mafft+sw, coati-max, and coat-sampling alignment, where the latter two utilize the ML estimates from our indel-phase model in chapter 4, to a dataset of 90 species (Zou and Zhang 2021). While the mafft+sw approach is a post-alignment correction that searches for the most likely phase of an indel, the coati-max and coati-sampling approaches actively consider gaps within the three indel phases during alignment. The coati-max method searches for the best alignment given the maximum log likelihood, while the coati-sampling method generates all possible alignments with their likelihood in a specified sample size.

The positive correlation between dN/dS and the ML estimates of ω, and the similar distribution between phase proportions and indel rates across 90 species demonstrate the effectiveness of our indel-phases model. In this chapter, I proposed a new summary statistic – Zn/(Zn+Zs) ratio, which was defined as the number of observed non-synonymous indel events (Zn), in a given evolutionary time, divided by the total number of non-synonymous indel events and synonymous indel events (Zs). Zheng, Graur, and Azevedo 2018 found a strong positive correlation between the deletion events and amino acid replacement in mammalian protein sequences. Lack of independence between these processes is biologically plausible because natural selection will affect persistence and fixation of both point mutations and insertions or deletions. Besides, our results suggest that the proportion of Zn indels is positively correlated to the ratio of divergence at nonsynonymous and synonymous sites, which can be represented as dN/dS ratio.

5.2 Methods & Materials

5.2.1 Coati-max & Coati-sampling Method

The coati-max is an aligning method built on the Viterbi algorithm, while the coati-sampling is an aligning method built on the Forward algorithm (https://github.com/jgarciamesa/coati). Both can accommodate different indel phase events. The coati-max method finds an optimal alignment given a pairwise HMM, and the coati-sampling method averages all possible alignments between the sequence pairs, eliminating bias introduced by the best alignment and allowing the alignment uncertainty to improve the parameter estimates.

5.2.2 Alignment

Mafft alignment utilized the default parameters and the global alignment algorithm. Coati-max and coati-sampling alignment took advantage of the ML estimates of the indel-phase model from 90 species datasets. Parameters including six $\sigma s, \omega, \tau, g, e$ were utilized as the default parameters for the alignment method. Of note, I took the average of phase 0, phase 1, and phase 2 gap opening weight as g, and the average of insertion and deletion extension weight as e. as the coati methods only accepted one gap opening and one gap extension value. Also, the gap extension weight calculated from the indel-phase model was based on the codon unit, while the calculation from coati aligners was based on a nucleotide unit. Hence, I need to triple the gap length and recalculate the extension weight before using aligners.

5.2.3 Quality Control

Even though the data was extracted from a published paper (Zou and Zhang 2021), I still conducted further data filtering to ensure that the filtered data was suitable for our research goal. Our dataset contained all of the gene pairs instead of a concatenated sequence of each species from chapter 2. I contacted the authors and obtained the aligned coding sequence pairs, removing all aligned gaps to restore the unaligned cds. The main quality control was executed in two-steps: 1) any unaligned sequence with ambiguous nucleotide (N), early stop codons, and in which length were not divisible by three were filtered out; 2) any alignment would be filtered out if they failed our gap-length statistics standard. After I obtained the filtered gene pairs from coati-max, the same gene ID was used to filter out the alignments generated by mafft+sw and coati-sampling methods.

Figure 5.1 left displayed a hex plot of one of the species pairs (01_FcaCaf). Only the gene pairs that satisfy the conditions ($f_{diff} < 0.1, f_{adj} < 0.5$) remained in the database. The f_{diff} was the percentage of the sequence length difference per gene pair, and the f_{adj} was the penalty of gap per gene pair, which was calculated as $f_{diff} =\mid len_A - len_B \mid / (len_A + len_B)$, $f_{adj} = (gap.len_A + gap.len_B - \mid len_A - len_B \mid)/(len_A + len_B)$ separately. The threshold value $\{0.1,0.5\}$ was selected based on this empirical data. The complete figure of all 90 species is in APPENDIX D.4.

Figure 5.1 right was a set of three graphs representing the post-filtering quality of the data. A-plot showed that the evolutionary distance (Erickson 2010) per gene pair followed a mixture gamma distribution (components=2) approximately. B-plot displayed that the empirical CDF of the distance followed an uniform distribution roughly, and C-plot was a Q-Q plot between the theoretical CDF and the empirical CDF of the data.

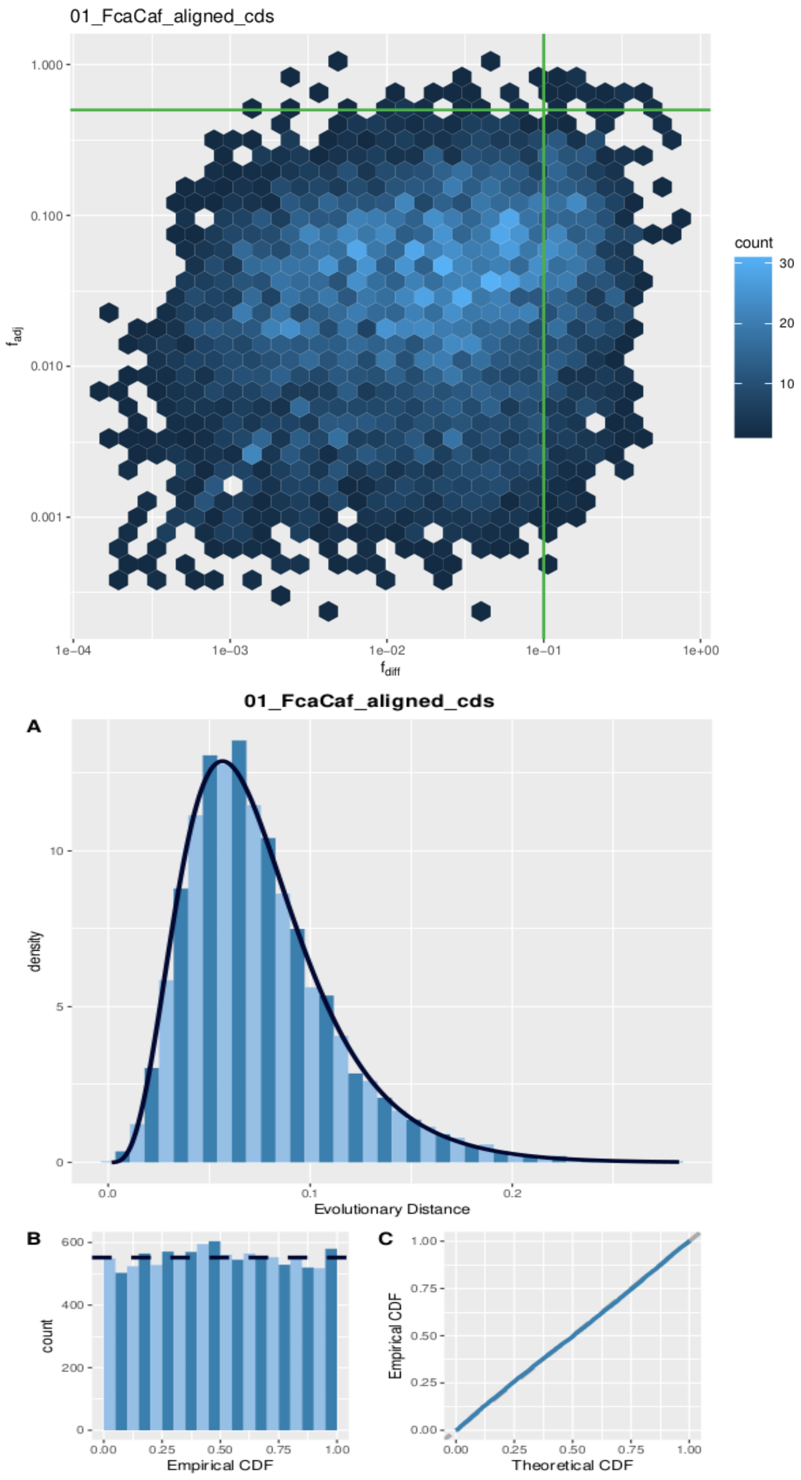


Figure 5.1: **Quality Control** | Upper figure is the hex plot of one species (01_$FcaCaf$). f_{diff} is on the x-axis and f_{adj} is on the y-axis. The green line is the threshold. Lower figure is a set of three plots representing the distribution of evolutionary distance.

5.2.4 dN/dS vs ω, Phase Proportions vs Normalized Indel Rates

To test the relationship between our direct summary statistic (dN/dS, phase proportions) and maximum-likelihood estimates (ω, indel rates), I generated a correlation plot between dN/dS and ω, and a distribution plot between phase proportions and normalized indel rates of 90 species across three alignment methods. I normalized the indel rates so that the sum of them equal to 1 so they could compare with the phase proportions.

5.2.5 Indel Phase Proportions across 90 Species

I extended the phase proportion analysis from mouse-rat species pair to 90 pairwise species spanning over eukaryotes, prokaryotes, and archaea realm. I only applied a single window size (6) on the mafft+sw method. And the coati-max/sampling method automatically generated different indel phases during the alignment. Then I created a table of genome-wide indel phase proportions of each species (APPENDIX D.2), a phase proportion plot across 90 species, and a distribution plot of three indel phases.

5.2.6 dN/dS vs Zn/(Zn+Zs) Ratio

The dN/dS ratio is commonly used as an indicator of selective pressure acting on a set of homologous protein coding genes. And the interpretation of dN/dS $<$ 1 is implied as negative selection, dN/dS $\sim$ 1 as neutrality, and dN/dS $>$ 1 as positive selection. I measured the dN/dS ratio using the (Nei and Gojobori 1986) method,

$$
\begin{aligned}
P_n &= N_d/N \\
P_s &= S_d/S \\
d_n d_s &= log(1-4P_n/3)/log(1-4P_s/3)
\end{aligned}
$$

where N_d, S_d is the number of observed nonsynonymous, synonymous substitutions, N, S is the number of expected nonsynonymous, synonymous substitutions. Inspired by the formulas above, I proposed a new summary statistic – Zn/(Zn+Zs) ratio,which was defined as the number of observed non-synonymous indel events (Z_n), in a given evolutionary time, divided by the sum of non-synonymous indel events and synonymous indel events (Z_s). I generated a distribution plot of dN/dS vs Zn/(Zn+Zs)ratio. The genome-wide ratio across 90 species can be found in APPENDIX D.3.

I investigated the relationship between Zn/(Zn+Zs) vs dN/dS ratio by applying a nonlinear least square fitted function with a logistic regression model, because Zn and Zs followed a binomial distribution with their sum as a weight argument. Next, I generated test data by using the rolling average of their dN/dS and Zn/(Zn+Zs) ratio per 201 gene pairs of each species. The rolling average helps avoid the possibility of zero indel event of any gene pair so the logistic function never collapses. And the total number of rolling average statistics equals the number of gene pairs - 201 + 1 for every species. Next, I was capable of predicting the test data via our nls-logistic model. In the meanwhile, I generated several relevant statistics including genome-wide GC percentage, sample size, neutral Zn/(Zn+Zs) value, asymptote and initial slope of our nls-logistic regression function of each species. Notably, neutral Zn/(Zn+Zs) was measured by calculating the number of Zn and Zs type of indels from the simulated deletion of the 3nt gap in any two codons generated by an empirical codon frequency from the data. Asymptote line draws from the max value of our nls model when the ω approximates to 1.

5.3 Results

5.3.1 Compare dN/dS with ω

Figure 5.2 presents a positive linear correlation between omega (ω) (ML estimates extracted from GTR+MG94 substitution model) and dN/dS (1989 method) of three alignment methods. The coefficient of determination (R^2) of each regression line is around 0.66 which is high, with a small p-value ($\ll 0.05$). The slope of the linear equation of sw, max, and sampling methods is about 1.15, 0.60, 0.60 respectively. The linear regression from the coati-max and coati-sampling method are more similar compared with the mafft+sw method.

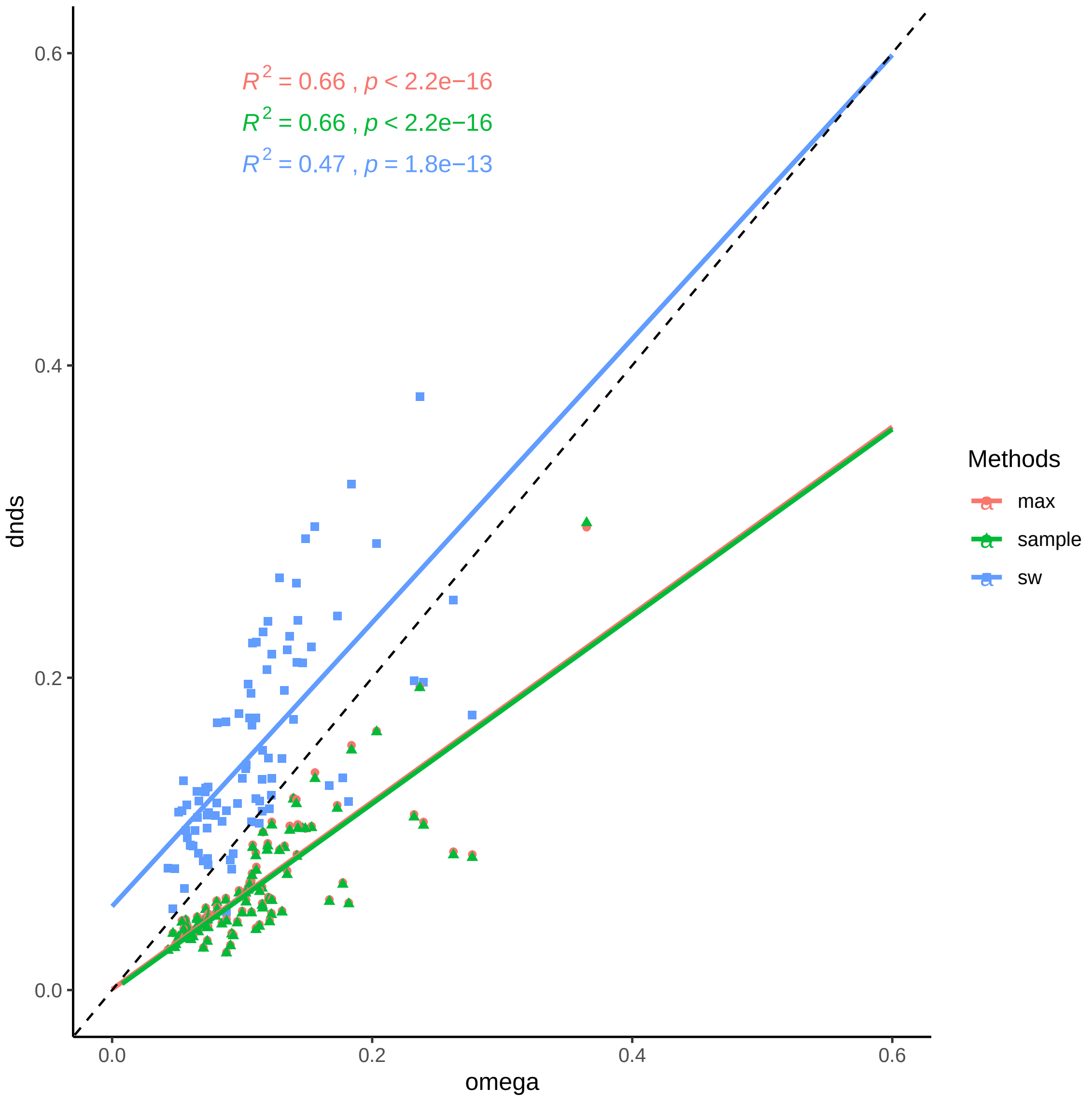
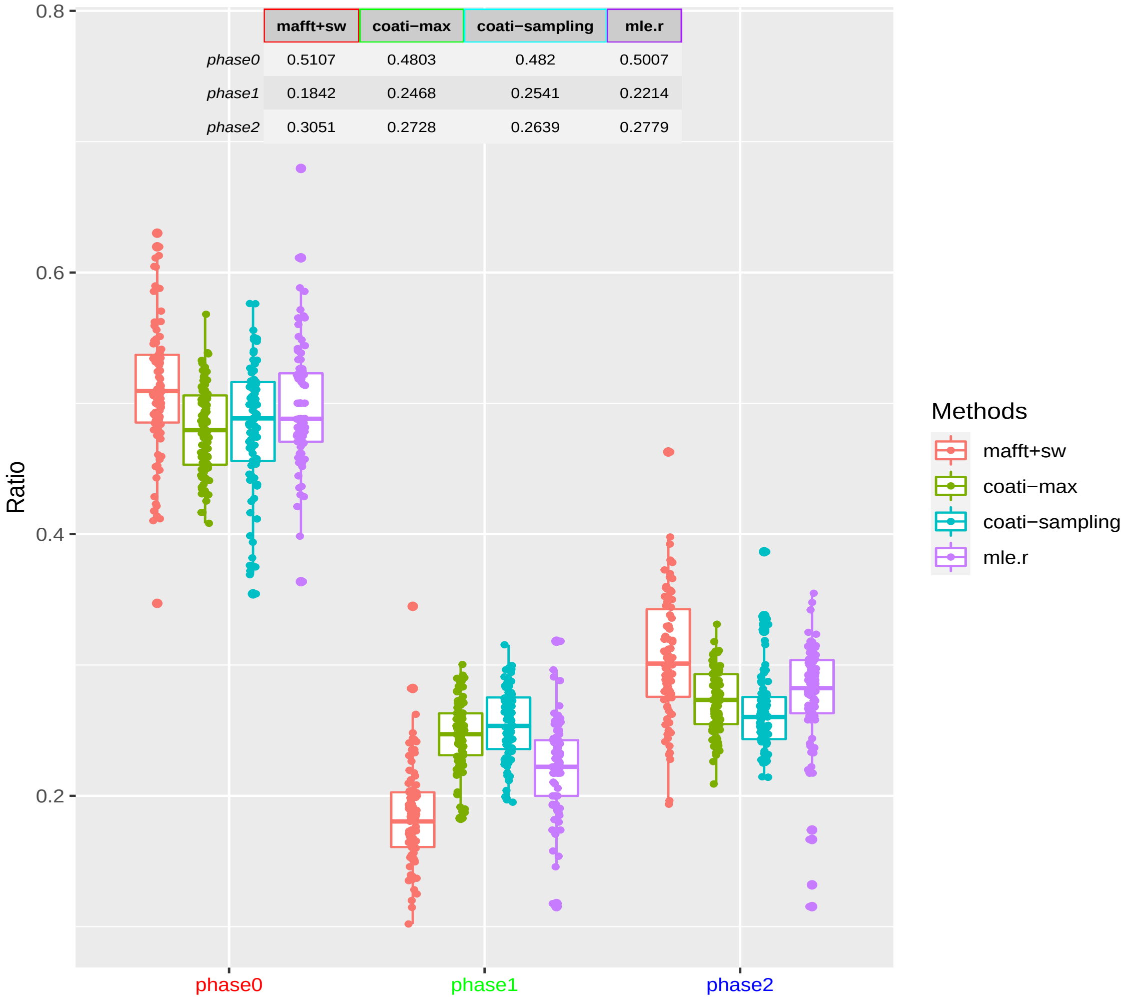


Figure 5.2: **Correlation Plot between dN/dS and ω across 90 Species Pairs |** Three linear regression lines describe the relationship between MLE of ω and dN/dS ratio. The dashed line is a diagonal line with a slope of 1.

5.3.2 Compare Phase Proportions with Indel Rates

Figure 5.3 compares the distributions between the normalized ML estimate of indel rates and the observed phase proportions across three alignment methods. I found that the distribution of the normalized indel rates displayed a similar trend as the observed proportion of indel phases, which was $r_0 > r_2 > r_1$. Also, the mean value of the phase proportions and the indel rates are very close between the coati-max and coati-sampling method.



	mafft+sw	coati−max	coati−sampling	mle.r
phase0	0.5107	0.4803	0.482	0.5007
phase1	0.1842	0.2468	0.2541	0.2214
phase2	0.3051	0.2728	0.2639	0.2779

Figure 5.3: **Distribution Comparison between ML Estimates of Indel Rates and Phase Proportions from Three Methods across 90 Species Pairs** | The table represents the mean value of each phase proportions across three alignment methods and the ML estimates of indel rates.

5.3.3 Phase Proportions across 90 Species

Figure 5.4 displays the phase proportion histogram of 90 species across three different alignment methods. The phase proportion trend of $phase0 > phase2 > phase1$ are universal across all of the species regardless of the method.

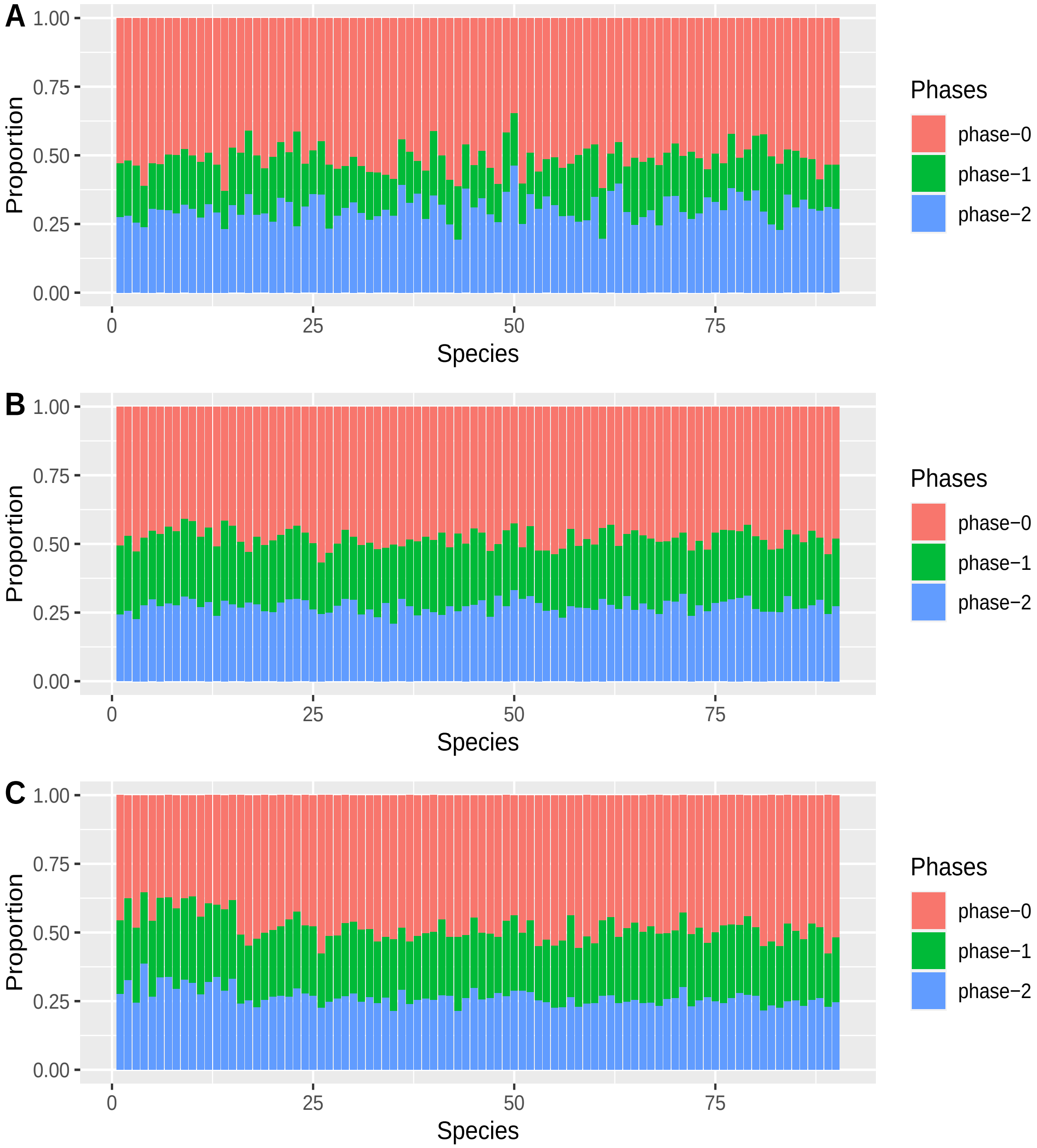
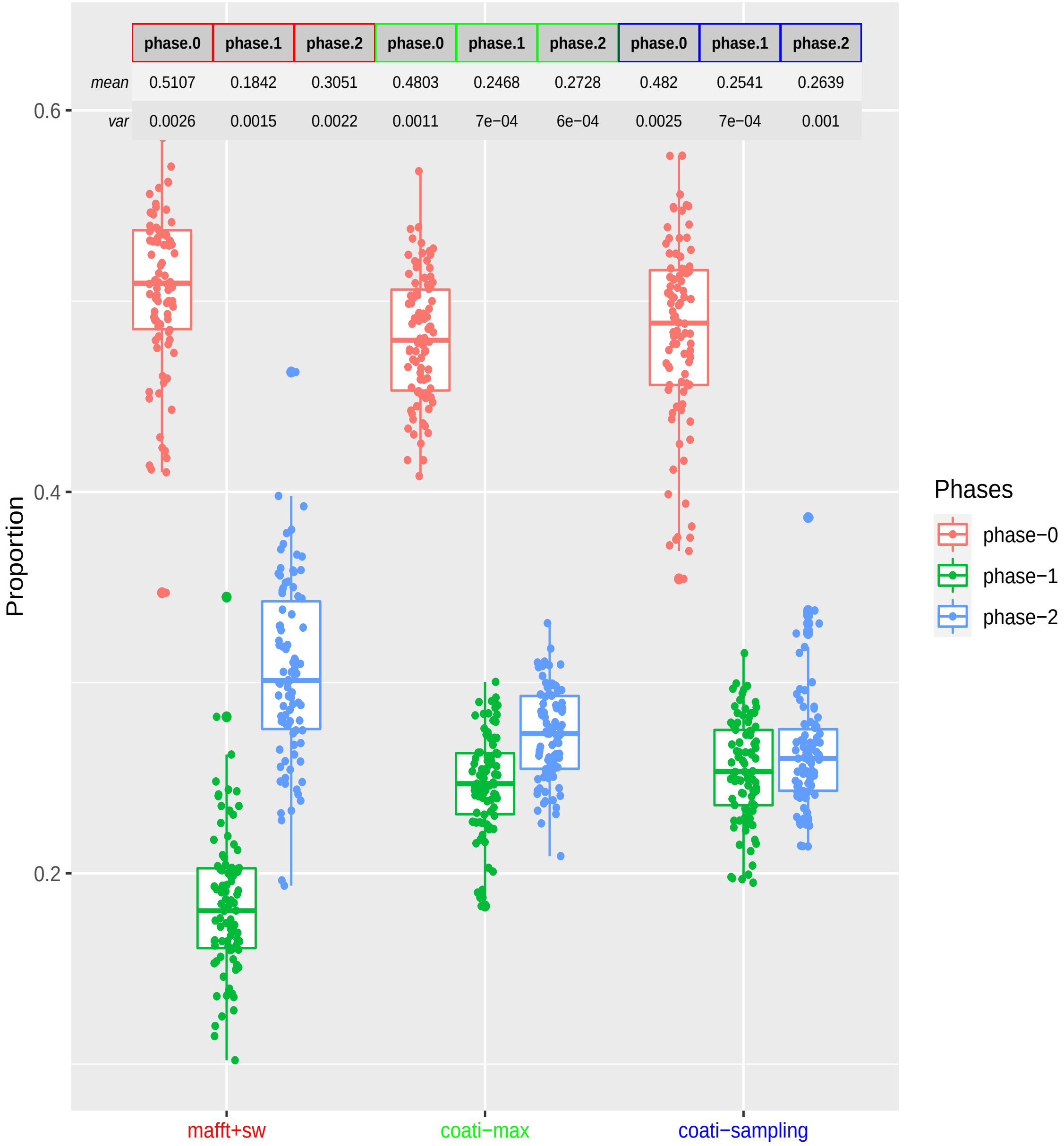


Figure 5.4: **Phase Proportion Histogram across 90 Species Pairs in Three Alignment Methods |** A. mafft+sw. B. coati-max. C. coati-sampling.

Figure 5.5 displays the distribution of phase proportions of 90 species across three alignment methods. The genome-wide phase distribution of coati-max and coati-sampling are more similar due to their model resemblance compared with the mafft+sw distribution. For example, the mean value of phase 0 proportion of mafft+sw is above 0.5, and the other two are slightly below 0.5. The mean value of phase 1 proportion of mafft+sw is below 0.2, and the other two are slightly above 0.2. The variance of phase distribution across 90 species is very small (< 0.001). Besides, I found that the proportion difference between phase 1 and phase 2 are more distinctive in mafft+sw compared to the other two methods, which suggested more effective purifying selection presented with the mafft+sw aligning data. Additionally, the qqplot and shapiro-wilk's test demonstrated that most of the phase proportion features followed a normal distribution except for the phase 0 and phase 2 indels of coati-sampling method, and phase 1 indels of mafft+sw method which is likely due to their smallest sample size(see APPENDIX D.4).



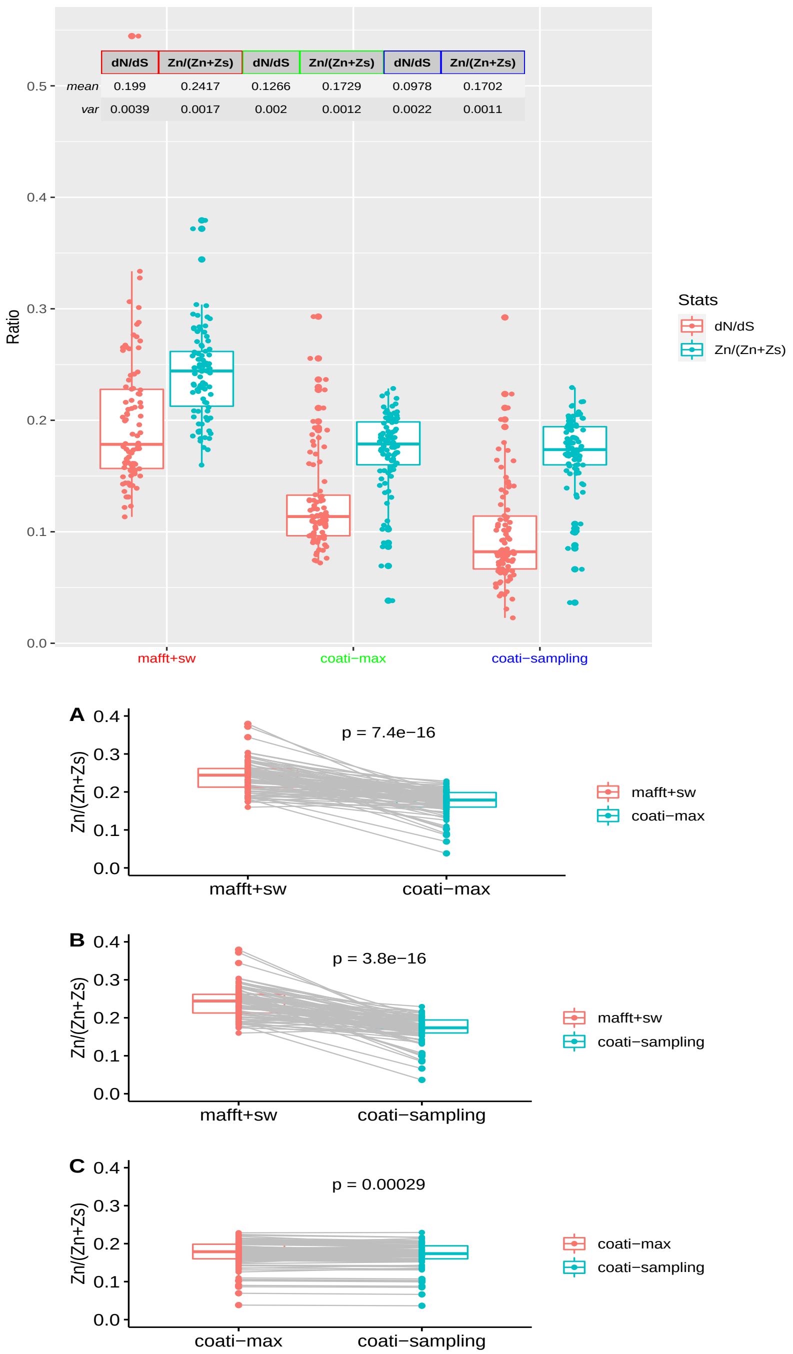
	phase.0	phase.1	phase.2	phase.0	phase.1	phase.2	phase.0	phase.1	phase.2
mean	0.5107	0.1842	0.3051	0.4803	0.2468	0.2728	0.482	0.2541	0.2639
var	0.0026	0.0015	0.0022	0.0011	7e−04	6e−04	0.0025	7e−04	0.001

Figure 5.5: **Phase Proportion Distribution across 90 Species Pairs in Three Alignment Methods** | Top: mean/variance table of each indel phase, red: mafft+sw. green: coati-max. blue: coati-sampling.

5.3.4 dN/dS and Zn/(Zn+Zs) Distribution

Figure 5.6 displays the genome-wide distribution of dN/dS and Zn/(Zn+Zs) ratio of 90 species across three alignment methods. I found that the total number of indels from mafft+sw method are far less than the one from the other two methods. A general trend of larger proportions of Zs indels displays in every species. The upper figure shows a general pattern of dN/dS ratio that is smaller than the Zn/(Zn+Zs) ratio across three alignment methods. The dN/dS distribution is positively skewed which suggests their mean is greater than the mode, while the Zn/(Zn+Zs) distribution is negatively skewed, which suggests their mean is less than the mode.

Additionally, the mean and variance of dN/dS and Zn/(Zn+Zs) ratio are closer in coati-max and coati-sampling methods. The lower figure displays the difference of Zn/(Zn+Zs) value between any two methods. Although the paired-sample wilcoxon signed rank test rejected the null hypothesis (< 0.05) that the median of the population of differences between the paired data is zero, the comparatively large p-value from coat-max/sampling group indicated their data similarity.



	dN/dS	Zn/(Zn+Zs)	dN/dS	Zn/(Zn+Zs)	dN/dS	Zn/(Zn+Zs)
mean	0.199	0.2417	0.1266	0.1729	0.0978	0.1702
var	0.0039	0.0017	0.002	0.0012	0.0022	0.0011

Figure 5.6: **Genome-wide Distribution of Zn/(Zn+Zs) Value.** | Upper: genome-wide distribution of dN/dS and Zn/(Zn+Zs) value of 90 species across three alignment methods (top: mean/variance table of dN/dS and Zn/(Zn+Zs) value). Lower: Zn/(Zn+Zs) ratio comparison between methods, p value is calculated from the Wilcoxon signed rank test. The Wilcoxon signed rank test between the mafft+sw and coati-max method (A), mafft+sw and coati-sampling method (B), and coati-max and coati-sampling method (C).

5.3.5 The Relationship between dN/dS and Zn/(Zn+Zs)

Figure 5.7 displays a relationship between dN/dS and Zn/(Zn+Zs) ratio of two species pairs (05_Droso, 10_ants). The upper figure displays a positive change of rolling average of Zn and Zs per 201 gene pairs with the increment of the dN/dS value. For every rolling average value, the Zn is always lower than Zs suggesting that the non-synonymous substitutions suffer more effective purifying selection. The lower figure presents a positive correlation between dN/dS and Zn/(Zn+Zs) rolling ratio, suggesting that the proportion of non-synonymous indels elevates as the effective purifying selection decreases. Besides, while the Zn/(Zn+Zs) approximates the asymptote when the dN/dS ratio increases, the asymptote value is still lower than the neutral Zn/(Zn+Zs) value in most of our species. The complete correlation figure of 90 species pairs can be found in Appendix D.4.

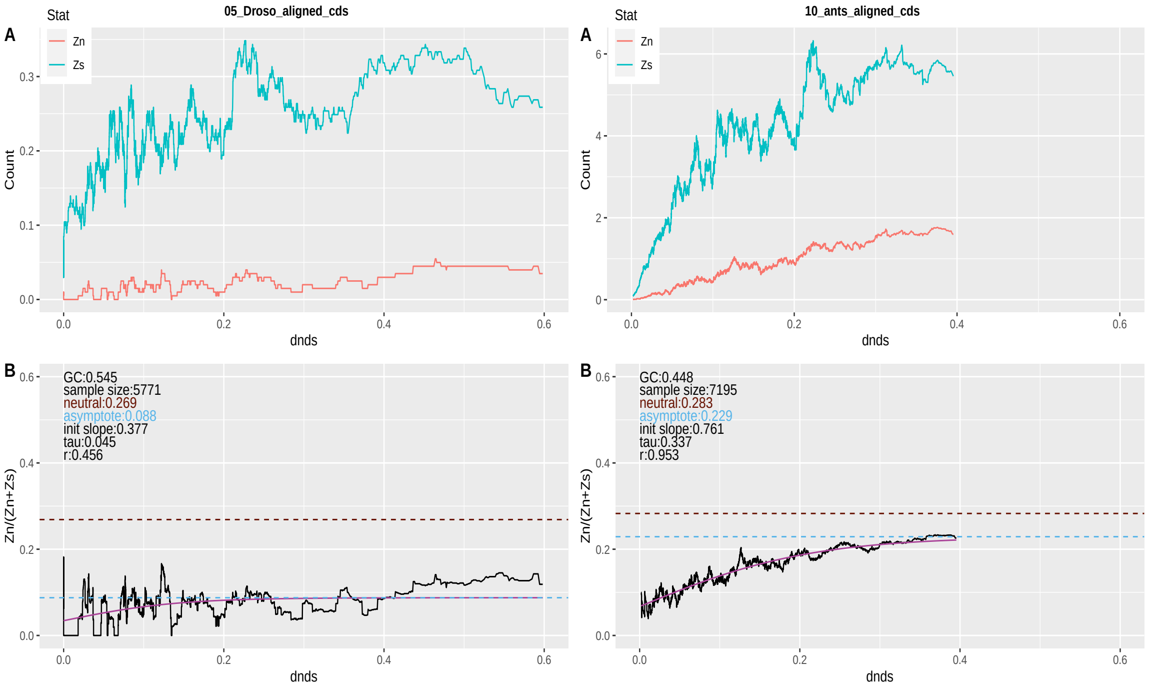


Figure 5.7: **Correlation Plot between dN/dS and Zn/(Zn+Zs) Ratio of Drosophila and Ant Species Pairs.** | Upper: Each line represents the rolling mean of Zn and Zs value. Lower: a nonlinear red line represents the correlation between Zn/(Zn+Zs) and dN/dS values. Upper left of the figure: genome-wide GC percentage, sample size – the number of gene pairs, neutral Zn/(Zn+Zs) is measured using the expected number of Zn and Zs value; asymptote line represents the maximum value of Zn/(Zn+Zs) when dN/dS increases to 1, τ is the branch length, init-slope represents the initial slope of the nonlinear model, and r is the Pearson correlation coefficient.

Positive initial slope According to figure 5.8, almost every species pair has a positive initial slope from the nonlinear least square model (nls) of logistic function. This consistency suggests that the Zn/(Zn+Zs) value is in accordance with the change of dN/dS ratio. Besides, the species pairs with a negative initial slope (ATGC260, ATGC279, ATGC371) have a large branch length (τ) but a small sample size which is less than 1500 gene pairs.

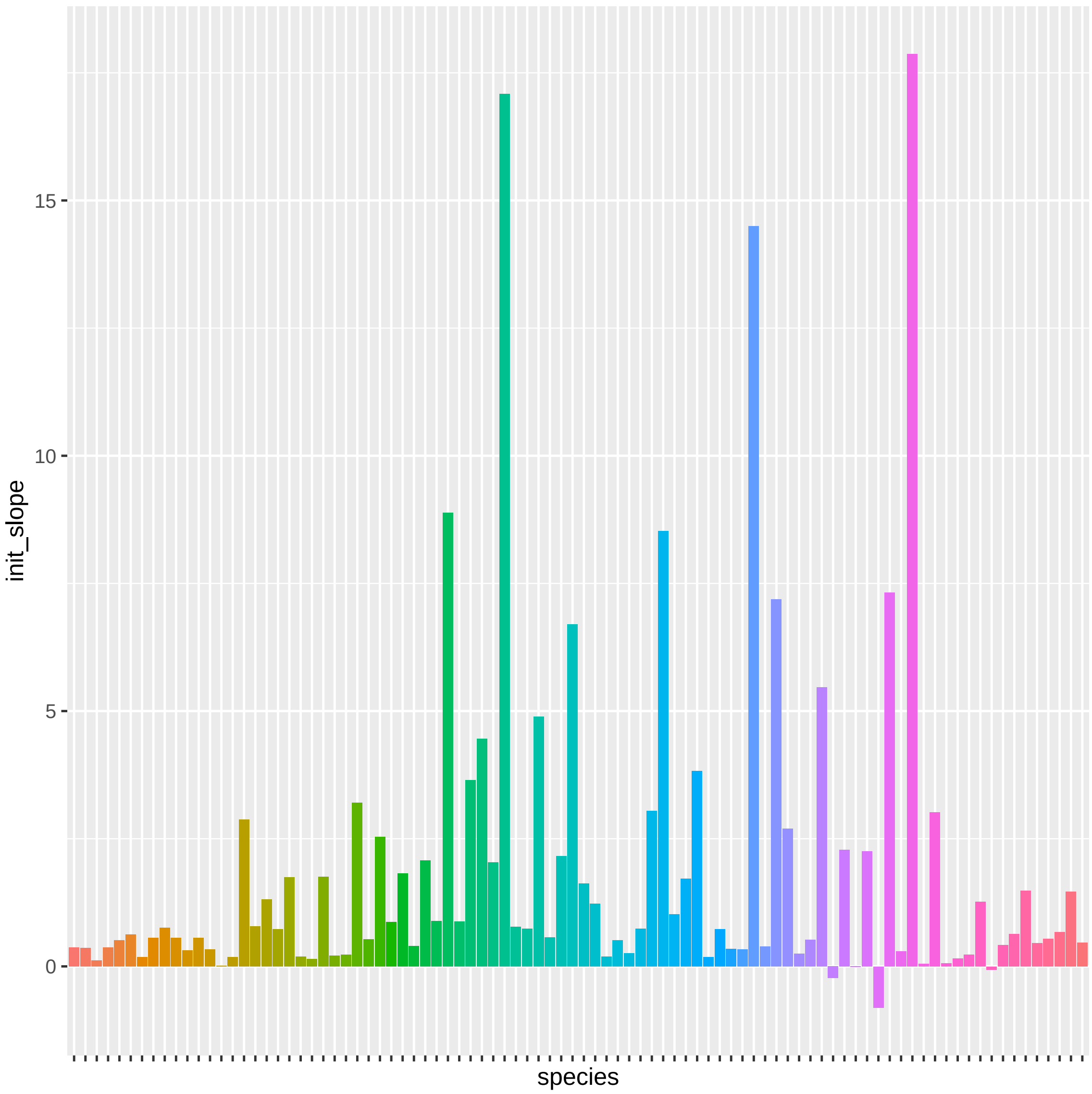


Figure 5.8: **Initial Slope Value of The Nls Model across 90 Species Pairs |** x-axis represents the 1-90 species pairs from left to right. y-axis represents the initial slope of our Nls model.

Asymptote vs neutral Zn/(Zn+Zs) value According to figure 5.9 A, the general distribution value of neutral Zn/(Zn+Zs) value is higher than the distribution value of asymptote across 90 species pair. The mean of neutral Zn/(Zn+Zs) and asymptote value is 0.271, 0.197 respectively. Besides, figure 5.9 B displays that the asymptote value is lower in almost every species pair, except for 11 outliers. This suggests gene pairs with a higher dN/dS ratio cannot guarantee that the neutral evolution of indels is happening within that species.

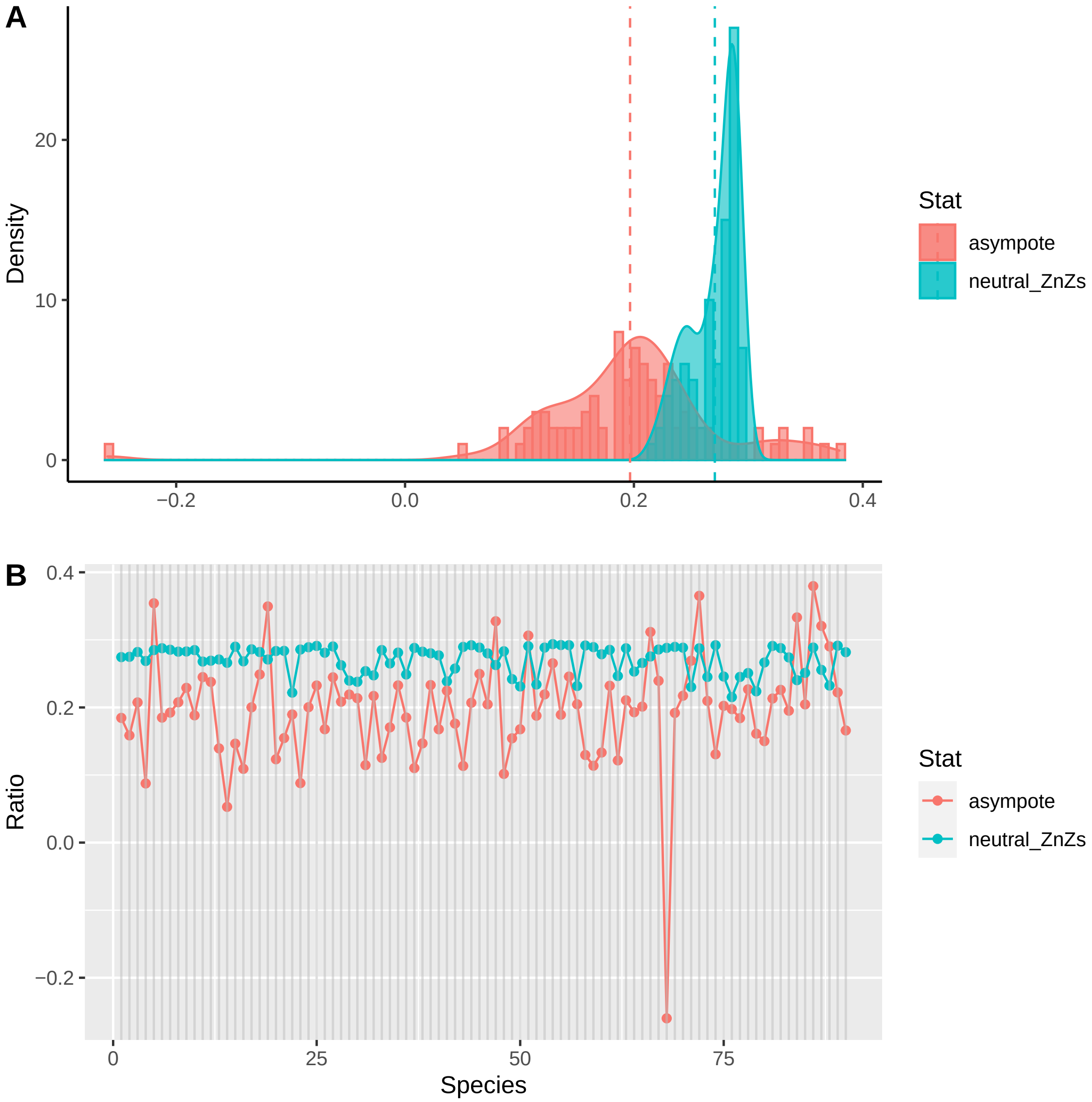


Figure 5.9: **Pairwise Comparison of Asymptote vs Neutral Zn/(Zn+Zs) Value |** A: The histogram of the distribution of asymptote and Neutral Zn/(Zn+Zs) value. The dashed vertical lines represent the mean value of each group. B: Pairwise comparison between the asymptote and neutral Zn/(Zn+Zs) value across 90 species pairs.

5.4 Discussion

The correlation between the MLE of ω and dN/dS ratio, MLE of indel rates and phase proportions suggested the effectiveness of our indel-phase model on the real sequence data. The indel phase proportion results demonstrated a universal purifying selection effect on the genomic coding regions of 90 species pairs regardless of the alignment methods. The similarity of phase proportions, dN/dS ratio and Zn/(Zn+Zs) ratio distributions between coati-max and coati-sampling was attributed to their model resemblance, which was different from the mafft+sw method. The positive correlation between Zn/(Zn+Zs) and dN/dS ratio showed a great potential of using Zn/(Zn+Zs) as a secondary indicator of natural selection.

5.4.1 dN/dS vs ω, Phase Proportions vs Indel Rates

There is a linear correlation between the dN/dS ratio and the ML estimates of ω of 90 species across three alignment methods. This consistency demonstrated the accuracy of our model on estimating parameters. Since the measure of dN/dS ratio required the observed non-synonymous over synonymous ratio normalized by the expected non-synonymous over synonymous ratio, which was different from the way of estimating the ω from the EM+importance sampling method. Also, we found that the lowest R-squared value from the mafft+sw method was likely due to the less amount of genomic data from the gap curation filtering process (chapter 3), compared with the coati-max and coati-sampling method.

While there is no obvious linear correlation between the phase proportions and ML estimates of indel rates, which is mostly likely due to the insufficient number of gaps compared to the substitutions. Still, their similar distribution results suggest that three normalized indel rates display the same trend as the phase proportions across three methods — phase 0 > phase 2 > phase 1.

5.4.2 Zn/(Zn+Zs) vs dN/dS ratio

I modeled the positive correlation between Zn/(Zn+Zs) and dN/dS ratio with a nonlinear least square of logistic model. Most of the initial slopes of the functions are positive with a high Pearson correlation coefficient (r), reflecting a good prediction of the data trend from this model. Through observing the data, I found that: 1) the closer two species are, the harder it is to get good estimates of the correlation function due to the lack of indel events. For example, 05_droso and 10_ants species pairs both belong to the insect branch, while the former displays a flat curve with a lower r value that could be attributed to its small branch length between species. 2) A large sample size increases the effectiveness of the model. For example, species pairs ATGC260, ATGC279, and ATGC371 with a negative initial slope of their model that fails to accurately predict the data that might be due to their small sample size. Besides, I also found that the 06_nematode displays a reduction in Zn and Zs count after reaching the peak, which is likely to be an artifact because their species divergence is large($\tau > 1$).

Additionally, the positive initial slope of the nls logistic regression of almost every species pairs indicates that the proportion of non-synonymous indel events (Zn) increases as the proportion of non-synonymous substitution (dN) events increases, reflecting the reduction of the effectiveness of purifying selection. Besides, all three pairwise species with a negative initial slope have a very small sample size that can lead to inaccurate initial slope estimates of our nls model.

Moreover, neutral Zn/(Zn+Zs) represents an expected ratio under the neutral evolutionary process, and an asymptote value represents an upper limit value of Zn/(Zn+Zs) from the nls logistic function when the dN/dS increases up to 1. The general pattern of neutral ratio on top of the asymptote (Fig 5.9 b) across 90 species indicates that the evolution of indels of almost every species are still influenced by effective purifying selection and have not entered into the neutral evolutionary process yet. Besides, there are 11 species pairs whose neutral Zn/(Zn+Zs) is lower than the asymptote value. I found that all of those 11 species pairs have a comparatively small sample size ($\sim$ 1000 gene pairs), presenting a linear shape of our nonlinear model, potentially leading to the biased estimates of the asymptote line.

In summary, while the results of positive correlation between Zn/(Zn+Zs) and dN/dS ratio across 90 species pairs are limited to the shorter indel data (3, 6, 9, 12 nts), our Zn/(Zn+Zs) ratio can still be very informative and indicative of the strength of the natural selection on current pairwise species.

Chapter 6

CONCLUSION

While insertions and deletions (indels) are common molecular evolutionary events, the study of indel phases and probabilistic models for indel evolution have received much less attention due to their model complexity. In this work, I developed and evaluated methods for profiling the indel phases and estimating the evolutionary parameters derived from substitution and indel models in pairwise alignments. In chapter 2 I provided an example of the expectation maximization algorithm for maximum-likelihood training for a substitution model, where the structural context of a residue was treated as a hidden variable that can evolve over time. In chapter 3 I developed a post-alignment fixation method that was capable of profiling each indel event into three different phases where the current codon-aware aligner was unable to. Expanding on the model established in chapters 2 and 3, I developed an indel-phase model that could describe the substitution and indel process together in pairwise sequences. In chapter 5 I extended our indel phase analysis to a more complex dataset via three different alignment methods, which utilized some important results from all previous chapters.

Current genomic datasets are largely unpolished and require advanced models and algorithms to handle uncertainties in alignments, erroneous estimates of evolutionary parameters, and other issues that could negatively impact comparative and functional genomic studies. Our substitution and indel models of evolution can be incorporated into future alignment software to robustly model coding sequence evolution and improve upon current aligners that do not support indel phases.

REFERENCES

Ajawatanawong, Pravech, and Sandra L Baldauf. 2013. "Evolution of protein indels in plants, animals and fungi." *BMC evolutionary biology* 13 (1): 1–15.

Arenas, Miguel. 2015. "Trends in substitution models of molecular evolution." *Frontiers in genetics* 6:319.

Baker, Monya. 2012. "De novo genome assembly: what every biologist should know." *Nature methods* 9 (4): 333–337.

Benner, Steven A, Mark A Cohen, and Gaston H Gonnet. 1993. "Empirical and structural models for insertions and deletions in the divergent evolution of proteins." *Journal of molecular biology* 229 (4): 1065–1082.

Brown, Michael, Richard Hughey, Anders Krogh, I Saira Mian, Kimmen Sjölander, and David Haussler. 1993. "Using Dirichlet mixture priors to derive hidden Markov models for protein families." In *Ismb,* 1:47–55.

Cartwright, Reed A. 2009a. "Problems and solutions for estimating indel rates and length distributions." *Molecular biology and evolution* 26 (2): 473–480.

Cartwright, Reed A. 2009b. "Problems and solutions for estimating indel rates and length distributions." *Molecular biology and evolution* 26 (2): 473–480.

Charlesworth, Brian. 2020. "How long does it take to fix a favorable mutation, and why should we care?" *The American Naturalist* 195 (5): 753–771.

Chaux, Nicole de la, Philipp W Messer, and Peter F Arndt. 2007. "DNA indels in coding regions reveal selective constraints on protein evolution in the human lineage." *BMC evolutionary biology* 7 (1): 1–12.

Comeron, Josep M, A Williford, and RM Kliman. 2008. "The Hill–Robertson effect: evolutionary consequences of weak selection and linkage in finite populations." *Heredity* 100 (1): 19–31.

Couvreur, Christophe. 1997. "The EM algorithm: A guided tour." In *Computer intensive methods in control and signal processing,* 209–222. Springer.

Cozzone, Alain J. 2002. "Proteins: Fundamental chemical properties." *Encyclopedia of Life Sciences. No month listed—2002. John Wiley & Sons Ltd,* 1–10.

Dempster, Arthur P, Nan M Laird, and Donald B Rubin. 1977. "Maximum likelihood from incomplete data via the EM algorithm." *Journal of the Royal Statistical Society: Series B (Methodological)* 39 (1): 1–22.

Durbin, Richard, Sean R Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids.* Cambridge university press.

Erickson, Keith. 2010. “The jukes-cantor model of molecular evolution.” *Primus* 20 (5): 438–445.

Felsenstein, Joseph. 1981. “Evolutionary trees from DNA sequences: a maximum likelihood approach.” *Journal of molecular evolution* 17 (6): 368–376.

Felsenstein, Joseph, and Joseph Felenstein. 2004. *Inferring phylogenies.* Vol. 2. Sinauer associates Sunderland, MA.

Furuno, Masaaki, Takeya Kasukawa, Rintaro Saito, Jun Adachi, Harukazu Suzuki, Richard Baldarelli, Yoshihide Hayashizaki, and Yasushi Okazaki. 2003. “CDS annotation in full-length cDNA sequence.” *Genome research* 13 (6b): 1478–1487.

Giantamidis, Georgios, Stavros Tripakis, and Stylianos Basagiannis. 2021. “Learning Moore machines from input–output traces.” *International Journal on Software Tools for Technology Transfer* 23 (1): 1–29.

Gillespie, Daniel T. 1977. “Exact stochastic simulation of coupled chemical reactions.” *The journal of physical chemistry* 81 (25): 2340–2361.

Goldman, Nick, and Ziheng Yang. 1994. “A codon-based model of nucleotide substitution for protein-coding DNA sequences.” *Molecular biology and evolution* 11 (5): 725–736.

Gregory, T Ryan. 2004. “Insertion–deletion biases and the evolution of genome size.” *Gene* 324:15–34.

Gu, Xun, and Wen-Hsiung Li. 1995. “The size distribution of insertions and deletions in human and rodent pseudogenes suggests the logarithmic gap penalty for sequence alignment.” *Journal of molecular evolution* 40 (4): 464–473.

Holmes, I, and GM Rubin. 2002. “An expectation maximization algorithm for training hidden substitution models.” *Journal of molecular biology* 317 (5): 753–764.

Holmes, Ian. 2005. “Using evolutionary expectation maximization to estimate indel rates.” *Bioinformatics* 21 (10): 2294–2300.

Hu, Jing, and Pauline C Ng. 2013. “SIFT Indel: predictions for the functional effects of amino acid insertions/deletions in proteins.” *PloS one* 8 (10): e77940.

Huelsenbeck, John P, and Kelly A Dyer. 2004. “Bayesian estimation of positively selected sites.” *Journal of Molecular Evolution* 58 (6): 661–672.

Kapli, Paschalia, Ziheng Yang, and Maximilian J Telford. 2020. “Phylogenetic tree building in the genomic age.” *Nature Reviews Genetics* 21 (7): 428–444.

Karlin, Samuel. 2014. *A first course in stochastic processes.* Academic press.

Katoh, Kazutaka, Kei-ichi Kuma, Hiroyuki Toh, and Takashi Miyata. 2005. “MAFFT version 5: improvement in accuracy of multiple sequence alignment.” *Nucleic acids research* 33 (2): 511–518.

Kloek, Tuen, and Herman K Van Dijk. 1978. “Bayesian estimates of equation system parameters: an application of integration by Monte Carlo.” *Econometrica: Journal of the Econometric Society,* 1–19.

Knudsen, Bjarne, and Michael M Miyamoto. 2003. “Sequence alignments and pair hidden Markov models using evolutionary history.” *Journal of molecular biology* 333 (2): 453–460.

Kosiol, Carolin, Ian Holmes, and Nick Goldman. 2007. “An empirical codon model for protein sequence evolution.” *Molecular biology and evolution* 24 (7): 1464–1479.

Kristensen, David M, Yuri I Wolf, and Eugene V Koonin. 2016. “ATGC database and ATGC-COGs: an updated resource for micro-and macro-evolutionary studies of prokaryotic genomes and protein family annotation.” *Nucleic acids research,* gkw934.

Kunkel, Thomas A. 2004. “DNA replication fidelity.” *Journal of Biological Chemistry* 279 (17): 16895–16898.

Kvikstad, Erika M, Svitlana Tyekucheva, Francesca Chiaromonte, and Kateryna D Makova. 2007. “A macaque’s-eye view of human insertions and deletions: differences in mechanisms.” *PLoS computational biology* 3 (9): e176.

Liò, Pietro, and Nick Goldman. 1998. “Models of molecular evolution and phylogeny.” *Genome research* 8 (12): 1233–1244.

Loewenthal, Gil, Dana Rapoport, Oren Avram, Asher Moshe, Elya Wygoda, Alon Itzkovitch, Omer Israeli, Dana Azouri, Reed A Cartwright, Itay Mayrose, et al. 2021. “A probabilistic model for indel evolution: differentiating insertions from deletions.” *Molecular biology and evolution* 38 (12): 5769–5781.

Lunter, Gerton. 2007. “Probabilistic whole-genome alignments reveal high indel rates in the human and mouse genomes.” *Bioinformatics* 23 (13): i289–i296.

Lynch, Michael, and Bruce Walsh. 2007. *The origins of genome architecture.* Vol. 98. Sinauer Associates Sunderland, MA.

Metzgar, David, Li Liu, Christian Hansen, Kevin Dybvig, and Christopher Wills. 2002. "Domain-level differences in microsatellite distribution and content result from different relative rates of insertion and deletion mutations." *Genome research* 12 (3): 408–413.

Metzler, Dirk, Roland Fleißner, Anton Wakolbinger, and Arndt von Haeseler. 2001. "Assessing variability by joint sampling of alignments and mutation rates." *Journal of molecular evolution* 53 (6): 660–669.

Miklós, István, Gerton A Lunter, and Ian Holmes. 2004. "A "long indel" model for evolutionary sequence alignment." *Molecular Biology and Evolution* 21 (3): 529–540.

Montoya-Burgos, Juan Ignacio, Pierre Boursot, and Nicolas Galtier. 2003. "Recombination explains isochores in mammalian genomes." *Trends in genetics* 19 (3): 128–130.

Moré, Jorge J, and Danny C Sorensen. 1982. *Newton's method.* Technical report. Argonne National Lab., IL (USA).

Muse, Spencer V, and Brandon S Gaut. 1994. "A likelihood approach for comparing synonymous and nonsynonymous nucleotide substitution rates, with application to the chloroplast genome." *Molecular biology and evolution* 11 (5): 715–724.

Nei, Masatoshi, and Takashi Gojobori. 1986. "Simple methods for estimating the numbers of synonymous and nonsynonymous nucleotide substitutions." *Molecular biology and evolution* 3 (5): 418–426.

Ochman, Howard. 2003. "Neutral mutations and neutral substitutions in bacterial genomes." *Molecular biology and evolution* 20 (12): 2091–2096.

Ophir, Ron, and Dan Graur. 1997. "Patterns and rates of indel evolution in processed pseudogenes from humans and murids." *Gene* 205 (1-2): 191–202.

Ranwez, Vincent, Sébastien Harispe, Frédéric Delsuc, and Emmanuel JP Douzery. 2011. "MACSE: Multiple Alignment of Coding SEquences accounting for frameshifts and stop codons." *PloS one* 6 (9): e22594.

Rodrigue, Nicolas, Nicolas Lartillot, and Hervé Philippe. 2008. "Bayesian comparisons of codon substitution models." *Genetics* 180 (3): 1579–1591.

Rosenberg, Michael S. 2009. *Sequence alignment: methods, models, concepts, and strategies.* Univ of California Press.

Saitou, Naruya, and Tadashi Imanishi. 1989. "Relative efficiencies of the Fitch-Margoliash, maximum-parsimony, maximum-likelihood, minimum-evolution, and neighbor-joining methods of phylogenetic tree construction in obtaining the correct tree."

Saitou, Naruya, and Shintaroh Ueda. 1994. “Evolutionary rates of insertion and deletion in noncoding nucleotide sequences of primates.” *Molecular Biology and Evolution* 11 (3): 504–512.

Schneider, Adrian, Alexander Souvorov, Niv Sabath, Giddy Landan, Gaston H Gonnet, and Dan Graur. 2009. “Estimates of positive Darwinian selection are inflated by errors in sequencing, annotation, and alignment.” *Genome biology and evolution* 1:114–118.

Sela, Itamar, Yuri I Wolf, and Eugene V Koonin. 2016. “Theory of prokaryotic genome evolution.” *Proceedings of the National Academy of Sciences* 113 (41): 11399–11407.

Simpson, Jared T. 2014. “Exploring genome characteristics and sequence quality without a reference.” *Bioinformatics* 30 (9): 1228–1235.

Sipos, Botond, Tim Massingham, Gregory E Jordan, and Nick Goldman. 2011. “PhyloSim-Monte Carlo simulation of sequence evolution in the R statistical computing environment.” *BMC bioinformatics* 12 (1): 1–6.

Sjödin, Per, Thomas Bataillon, and Mikkel H Schierup. 2010. “Insertion and deletion processes in recent human history.” *PLoS One* 5 (1): e8650.

Smith, Megan L, and Matthew W Hahn. 2021. “New approaches for inferring phylogenies in the presence of paralogs.” *Trends in Genetics* 37 (2): 174–187.

Springer, Mark S, William J Murphy, Eduardo Eizirik, and Stephen J O’Brien. 2003. “Placental mammal diversification and the Cretaceous–Tertiary boundary.” *Proceedings of the National Academy of Sciences* 100 (3): 1056–1061.

Suchard, Marc A, Robert E Weiss, and Janet S Sinsheimer. 2001. “Bayesian selection of continuous-time Markov chain evolutionary models.” *Molecular biology and evolution* 18 (6): 1001–1013.

Tao, Shiheng, Yanhui Fan, Wenjuan Wang, Guoji Ma, Lijing Liang, and Qi Shi. 2007. “Patterns of insertion and deletion in mammalian genomes.” *Current Genomics* 8 (6): 370–378.

Tavaré, Simon, et al. 1986. “Some probabilistic and statistical problems in the analysis of DNA sequences.” *Lectures on mathematics in the life sciences* 17 (2): 57–86.

Taylor, Martin S, Chris P Ponting, and Richard R Copley. 2004. “Occurrence and consequences of coding sequence insertions and deletions in Mammalian genomes.” *Genome research* 14 (4): 555–566.

Thorne, Jeffrey L, Hirohisa Kishino, and Joseph Felsenstein. 1991. “An evolutionary model for maximum likelihood alignment of DNA sequences.” *Journal of Molecular Evolution* 33 (2): 114–124.

Van Dijk, Erwin L, Hélène Auger, Yan Jaszczyszyn, and Claude Thermes. 2014. “Ten years of next-generation sequencing technology.” *Trends in genetics* 30 (9): 418–426.

Wang, J, E Santiago, and Armando Caballero. 2016. “Prediction and estimation of effective population size.” *Heredity* 117 (4): 193–206.

wenci, yu. 1979. “THE CONVERGENTE PROPERTY OF THE SIMPLEX EVOLUTIONARY TECHNIQUE.” *Scientia Sinica (in Chinese)* 9 (S1): 69–77.

Yang, Ziheng. 2007. “PAML 4: phylogenetic analysis by maximum likelihood.” *Molecular biology and evolution* 24 (8): 1586–1591.

Yang, Ziheng, and Rasmus Nielsen. 2008. “Mutation-selection models of codon substitution and their use to estimate selective strengths on codon usage.” *Molecular biology and evolution* 25 (3): 568–579.

Zhang, Zhaolei, and Mark Gerstein. 2003. “Patterns of nucleotide substitution, insertion and deletion in the human genome inferred from pseudogenes.” *Nucleic acids research* 31 (18): 5338–5348.

Zhao, Huiying, Yuedong Yang, Hai Lin, Xinjun Zhang, Matthew Mort, David N Cooper, Yunlong Liu, and Yaoqi Zhou. 2013. “DDIG-in: discriminating between disease-associated and neutral non-frameshifting micro-indels.” *Genome biology* 14 (3): 1–13.

Zheng, Yichen, Dan Graur, and Ricardo BR Azevedo. 2018. “Correlated selection on amino acid deletion and replacement in mammalian protein sequences.” *Journal of Molecular Evolution* 86 (6): 365–378.

Zou, Zhengting, and Jianzhi Zhang. 2021. “Are nonsynonymous transversions generally more deleterious than nonsynonymous transitions?” *Molecular biology and evolution* 38 (1): 181–191.

APPENDIX A

SUPPLEMENTARY MATERIAL FOR CHAPTER 2

True Parameters for 100 Simulations

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
0.601	2.585	0.170	0.522	2.349	1.879	0.076	0.245
0.162	0.969	0.594	1.510	1.773	2.856	0.207	0.210
1.840	1.278	1.020	1.079	2.776	0.525	0.074	0.451
3.924	1.815	0.818	1.177	0.037	0.369	0.177	0.303
0.074	0.497	3.153	1.914	1.491	1.811	0.398	0.274
3.526	1.281	0.273	1.118	0.112	1.745	0.055	0.149
4.229	1.541	1.181	1.400	2.697	0.366	0.310	0.180
1.174	0.259	1.182	1.790	2.101	2.591	0.220	0.314
0.633	1.259	2.207	0.945	0.115	0.310	0.239	0.493
0.569	1.524	1.760	1.206	2.060	2.294	0.041	0.146
1.648	1.389	0.469	1.274	2.849	0.223	0.249	0.242
2.205	0.859	1.003	1.049	1.146	1.758	0.101	0.385
1.129	0.191	0.448	2.385	1.557	1.703	0.129	0.370
14.651	2.668	12.609	1.343	1.445	0.093	0.347	0.081
0.046	0.474	0.906	3.377	0.625	3.473	0.407	0.213
1.509	11.201	0.173	1.665	1.302	0.683	0.228	0.134
2.445	3.752	1.386	5.930	0.167	0.032	0.473	0.229
0.083	0.051	2.157	2.532	0.178	1.856	0.251	0.466
2.950	0.324	0.567	2.557	0.776	0.059	0.357	0.199
0.067	2.057	0.277	2.020	1.329	0.424	0.283	0.333
0.073	0.176	0.085	2.065	0.086	8.156	0.067	0.178
1.091	0.308	0.083	0.313	3.220	3.539	0.222	0.445
2.820	2.320	1.723	1.259	0.646	4.283	0.030	0.072
3.433	0.276	1.551	1.241	3.206	0.043	0.437	0.341
1.221	0.520	1.029	2.780	0.031	1.434	0.025	0.403
0.382	0.248	2.378	2.943	0.087	1.793	0.219	0.467
0.044	1.523	0.737	2.063	0.435	0.822	0.198	0.558
0.171	1.414	0.082	2.005	0.736	1.816	0.379	0.330
1.506	1.472	1.918	0.374	0.439	1.492	0.335	0.177
0.793	1.080	1.707	3.411	0.704	0.476	0.080	0.373
3.490	11.366	4.307	1.715	0.036	2.592	0.268	0.079
1.281	0.340	0.581	2.066	0.456	1.252	0.484	0.630
2.529	2.232	0.471	0.043	1.902	0.798	0.401	0.517
0.604	0.207	0.284	2.446	3.296	0.765	0.092	0.208
1.802	1.223	0.594	2.181	0.706	0.248	0.083	0.464
0.201	2.023	1.736	16.912	1.117	1.334	0.419	0.109
1.646	1.150	3.346	1.891	1.880	0.398	0.037	0.287
1.121	4.923	11.900	0.573	0.981	5.610	0.114	0.060
0.427	3.950	3.535	1.519	9.144	5.612	0.301	0.059
1.083	1.707	2.457	0.376	0.879	1.249	0.431	0.188
4.392	1.298	3.647	0.443	1.197	3.468	0.088	0.075
1.885	0.180	0.549	0.235	2.924	3.558	0.232	0.186
0.091	4.546	3.230	0.765	1.043	1.334	0.232	0.084
0.019	2.744	1.799	0.978	1.832	2.111	0.296	0.191
2.531	3.634	0.265	2.332	0.614	0.445	0.372	0.113
2.533	1.333	1.243	0.396	4.156	0.382	0.066	0.147
3.910	3.535	0.202	1.324	0.142	3.276	0.343	0.368
2.432	2.352	4.768	0.221	0.989	1.765	0.259	0.234
0.503	4.523	1.075	1.663	0.966	0.812	0.257	0.445
0.115	7.304	1.649	2.418	1.067	3.717	0.429	0.159
0.465	0.284	0.450	2.476	0.875	4.417	0.333	0.219
0.935	0.671	0.434	2.096	0.002	0.519	0.179	0.408
3.543	0.298	1.851	6.891	0.167	1.018	0.425	0.164
2.234	0.503	2.047	0.239	1.208	0.214	0.490	0.370
0.319	1.068	1.977	0.404	0.021	3.387	0.403	0.451
10.759	1.139	7.288	0.497	4.451	4.704	0.476	0.091
1.116	12.203	1.786	0.308	3.206	5.774	0.036	0.028
0.443	0.094	1.484	2.494	0.398	0.519	0.341	0.512
0.856	2.289	0.073	1.320	5.674	1.825	0.298	0.141
1.703	0.429	0.079	1.183	4.324	0.384	0.396	0.199
1.026	0.642	1.814	1.048	2.384	1.842	0.068	0.173
0.594	1.314	2.570	1.725	1.168	0.269	0.444	0.250
0.802	0.246	1.945	0.688	1.227	2.699	0.248	0.163
1.924	6.388	2.752	0.064	0.340	2.381	0.243	0.086
1.299	3.974	1.124	0.305	1.874	0.247	0.086	0.295
0.924	0.470	0.052	2.594	0.746	0.195	0.099	0.432

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
2.055	0.085	2.624	1.527	5.711	3.998	0.287	0.191
1.918	0.741	2.058	1.550	0.342	1.036	0.347	0.547
1.688	1.061	0.380	2.379	1.260	0.345	0.201	0.935
1.565	1.783	1.503	0.273	0.484	2.091	0.063	0.195
1.262	2.683	0.316	1.179	2.018	1.163	0.340	0.132
5.238	5.388	1.446	0.220	4.233	0.703	0.278	0.203
0.045	1.140	3.263	0.624	2.838	0.214	0.459	0.328
0.551	1.358	4.248	0.438	0.412	0.344	0.056	0.307
1.034	1.729	3.662	1.114	0.295	0.381	0.130	0.287
3.736	0.414	1.273	0.600	1.231	0.658	0.260	0.288
0.618	2.677	1.883	0.517	1.642	1.627	0.194	0.222
2.055	2.228	0.793	0.584	1.310	1.476	0.485	0.242
0.908	0.236	2.125	3.253	0.228	1.683	0.293	0.463
4.812	1.158	5.247	1.244	3.800	4.015	0.156	0.129
0.213	1.613	1.688	3.047	0.115	0.279	0.168	0.510
1.689	0.782	2.088	1.565	0.441	0.471	0.430	0.340
1.957	0.225	3.147	1.878	0.734	1.489	0.482	0.286
0.988	0.616	2.595	1.793	0.490	0.581	0.449	0.194
0.591	0.425	3.149	3.190	0.019	0.139	0.145	0.523
0.183	0.843	0.578	1.769	1.379	2.790	0.448	0.302
0.289	2.339	0.267	2.365	0.545	0.711	0.175	0.350
1.562	1.828	1.190	1.169	0.211	3.027	0.135	0.193
0.175	0.577	1.767	8.635	3.561	2.406	0.317	0.207
1.730	13.334	6.481	1.036	9.252	19.260	0.307	0.044
0.690	0.079	1.946	2.070	1.311	1.701	0.474	0.264
1.806	2.119	1.408	1.641	0.909	0.375	0.464	0.340
4.723	0.432	1.292	0.736	0.778	0.016	0.459	0.201
3.230	3.562	0.470	0.082	0.230	0.790	0.411	0.248
1.369	0.710	1.134	2.111	0.134	2.294	0.059	0.249
4.366	6.153	1.455	0.350	4.682	0.301	0.145	0.197
0.342	1.686	1.551	1.469	3.632	0.210	0.251	0.282
33.921	8.573	0.120	7.184	7.704	4.175	0.270	0.031
1.858	0.160	0.428	2.183	0.321	2.351	0.234	0.361
4.701	11.321	3.832	0.954	0.816	6.923	0.177	0.047

Table A.1: True Parameters for 100 Simulations.

Parameter Estimates of 100 Simulations in The Codon Length of 10^5 via Phylo-Em Method

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
0.592	2.602	0.173	0.512	2.351	1.875	0.075	0.243
0.154	0.970	0.605	1.486	1.750	2.902	0.204	0.211
1.846	1.253	1.027	1.089	2.781	0.530	0.073	0.452
3.972	1.790	0.849	1.173	0.035	0.362	0.178	0.302
0.069	0.506	3.231	1.909	1.472	1.791	0.395	0.276
3.495	1.268	0.277	1.129	0.111	1.775	0.054	0.148
4.255	1.519	1.194	1.273	2.687	0.365	0.305	0.181
1.176	0.238	1.197	1.761	2.078	2.605	0.219	0.318
0.626	1.275	2.206	0.908	0.119	0.295	0.240	0.492
0.624	1.472	1.751	1.284	2.056	2.354	0.040	0.148
1.667	1.394	0.488	1.261	2.814	0.234	0.249	0.241
2.199	0.845	1.018	1.056	1.145	1.756	0.100	0.392
1.138	0.198	0.436	2.379	1.577	1.674	0.128	0.370
14.785	2.660	10.551	1.352	1.436	0.067	0.344	0.081
0.040	0.468	0.908	3.373	0.631	3.472	0.406	0.213
1.537	11.115	0.179	1.444	1.336	0.634	0.232	0.134
2.480	3.769	1.372	6.009	0.170	0.038	0.470	0.232
0.084	0.045	2.155	2.375	0.195	1.874	0.250	0.469
2.972	0.314	0.568	2.557	0.769	0.052	0.352	0.202
0.069	2.064	0.232	2.024	1.293	0.409	0.280	0.336
0.074	0.176	0.090	1.986	0.087	8.176	0.067	0.178
1.114	0.302	0.085	0.300	3.214	3.545	0.220	0.446
2.845	2.313	1.497	1.281	0.609	4.165	0.027	0.073
3.515	0.266	1.588	1.236	3.147	0.049	0.438	0.340
1.228	0.526	1.027	2.789	0.027	1.409	0.024	0.402
0.381	0.249	2.420	2.935	0.087	1.771	0.218	0.466

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
0.049	1.542	0.541	2.064	0.414	0.811	0.198	0.564
0.177	1.387	0.057	2.004	0.720	1.851	0.372	0.335
1.536	1.446	1.922	0.367	0.447	1.469	0.340	0.173
0.813	1.074	1.713	3.369	0.711	0.487	0.080	0.372
3.932	11.301	3.585	1.718	0.000	2.479	0.263	0.080
1.295	0.326	0.540	2.067	0.433	1.263	0.484	0.634
2.548	2.248	0.473	0.026	1.862	0.815	0.402	0.517
0.595	0.210	0.286	2.384	3.335	0.786	0.091	0.206
1.821	1.200	0.578	2.192	0.710	0.234	0.082	0.466
0.192	2.034	1.746	16.510	1.087	1.322	0.406	0.110
1.692	1.068	3.272	1.898	1.838	0.411	0.036	0.289
1.070	5.015	11.267	0.572	0.998	5.625	0.111	0.060
0.339	3.974	3.064	1.529	8.874	5.506	0.286	0.060
1.113	1.706	2.458	0.356	0.863	1.263	0.435	0.188
4.498	1.288	3.720	0.425	1.201	3.433	0.085	0.076
1.905	0.173	0.570	0.226	2.906	3.550	0.228	0.188
0.077	4.551	3.321	0.764	1.035	1.348	0.222	0.086
0.020	2.775	1.788	0.991	1.817	2.055	0.289	0.194
2.571	3.613	0.266	2.327	0.619	0.439	0.374	0.112
2.482	1.317	1.254	0.488	4.178	0.379	0.067	0.148
3.990	3.504	0.200	1.266	0.141	3.237	0.336	0.377
2.486	2.339	4.765	0.211	0.984	1.738	0.258	0.235
0.524	4.604	0.999	1.657	0.937	0.821	0.257	0.446
0.139	7.273	1.650	2.546	1.027	3.687	0.430	0.158
0.449	0.268	0.458	2.406	0.846	4.490	0.322	0.225
0.936	0.684	0.359	2.096	0.000	0.497	0.177	0.412
3.518	0.290	1.859	6.254	0.174	1.042	0.429	0.162
2.298	0.502	2.036	0.217	1.212	0.216	0.494	0.369
0.327	1.049	1.986	0.389	0.020	3.369	0.404	0.450
10.910	1.103	7.541	0.512	4.279	4.694	0.481	0.090
1.157	11.772	2.644	0.299	3.591	5.754	0.039	0.028
0.456	0.094	1.533	2.489	0.390	0.520	0.343	0.508
0.882	2.222	0.038	1.345	5.497	1.821	0.292	0.141
1.717	0.424	0.074	1.196	4.268	0.390	0.391	0.200
1.045	0.639	1.799	1.020	2.378	1.884	0.066	0.172
0.598	1.295	2.583	1.698	1.176	0.276	0.444	0.250
0.801	0.230	1.938	0.671	1.243	2.717	0.243	0.165
1.960	6.402	2.927	0.058	0.322	2.366	0.241	0.088
1.303	4.022	1.134	0.303	1.842	0.237	0.083	0.297
0.937	0.467	0.040	2.590	0.745	0.195	0.096	0.432
2.044	0.077	2.101	1.532	5.651	3.988	0.285	0.191
1.917	0.733	2.057	1.552	0.352	1.036	0.343	0.553
1.691	1.063	0.391	2.377	1.268	0.344	0.201	0.934
1.584	1.751	1.487	0.275	0.479	2.128	0.061	0.196
1.286	2.698	0.302	1.180	1.980	1.164	0.332	0.132
5.395	5.318	1.438	0.299	4.296	0.713	0.277	0.204
0.051	1.151	3.295	0.612	2.793	0.222	0.456	0.328
0.544	1.344	4.223	0.450	0.424	0.382	0.054	0.306
1.019	1.699	3.723	1.116	0.299	0.373	0.130	0.289
3.770	0.402	1.289	0.567	1.217	0.649	0.261	0.288
0.604	2.671	1.911	0.510	1.648	1.638	0.192	0.222
2.058	2.215	0.793	0.561	1.308	1.504	0.482	0.244
0.901	0.233	2.130	3.239	0.236	1.658	0.292	0.467
4.853	1.215	4.885	1.246	3.841	3.958	0.150	0.130
0.218	1.617	1.664	3.052	0.111	0.280	0.170	0.513
1.698	0.775	2.088	1.434	0.462	0.468	0.434	0.339
1.994	0.215	3.221	1.879	0.710	1.468	0.486	0.285
0.989	0.589	2.609	1.745	0.493	0.586	0.454	0.194
0.602	0.418	3.142	3.191	0.017	0.137	0.142	0.530
0.180	0.839	0.589	1.777	1.345	2.775	0.452	0.301
0.278	2.298	0.260	2.381	0.530	0.739	0.172	0.352
1.543	1.771	1.213	1.152	0.216	3.045	0.130	0.198
0.153	0.583	1.775	8.057	3.602	2.311	0.314	0.210
1.804	13.606	4.610	1.014	9.289	19.121	0.307	0.044
0.696	0.074	1.972	2.076	1.295	1.681	0.478	0.261
1.831	2.125	1.399	1.624	0.896	0.382	0.471	0.337
4.750	0.426	1.292	0.705	0.769	0.015	0.441	0.206
3.230	3.581	0.478	0.078	0.228	0.787	0.407	0.251
1.361	0.706	1.175	2.087	0.129	2.319	0.059	0.246
4.388	6.281	1.350	0.350	4.637	0.290	0.141	0.197
0.336	1.689	1.594	1.477	3.595	0.204	0.248	0.281
33.234	8.422	0.117	7.328	8.091	4.286	0.275	0.031

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
1.858	0.148	0.426	2.181	0.311	2.380	0.231	0.368
4.434	11.447	3.737	0.949	0.926	7.002	0.180	0.047

Table A.2: Parameter Estimates of 100 Simulations in The Codon Length of 10^5 via Phylo-Em Method.

Parameter Estimates of 100 Simulations in The Codon Length of 10^5 via Nmkb Method

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
0.592	2.602	0.173	0.512	2.351	1.875	0.075	0.243
0.154	0.970	0.605	1.486	1.750	2.902	0.204	0.211
1.846	1.253	1.027	1.089	2.781	0.530	0.073	0.452
3.972	1.790	0.849	1.173	0.035	0.362	0.178	0.302
0.069	0.506	3.231	1.909	1.472	1.791	0.395	0.276
3.495	1.268	0.277	1.129	0.111	1.775	0.054	0.148
4.255	1.519	1.194	1.273	2.687	0.365	0.305	0.181
1.176	0.238	1.197	1.761	2.078	2.605	0.219	0.318
0.626	1.275	2.206	0.908	0.119	0.295	0.240	0.492
0.624	1.472	1.751	1.284	2.056	2.354	0.040	0.148
1.667	1.394	0.488	1.261	2.814	0.234	0.249	0.241
2.199	0.845	1.018	1.056	1.145	1.756	0.100	0.392
1.138	0.198	0.436	2.379	1.577	1.674	0.128	0.370
14.785	2.660	10.551	1.352	1.436	0.067	0.344	0.081
0.040	0.468	0.908	3.373	0.631	3.472	0.406	0.213
1.537	11.115	0.179	1.444	1.336	0.634	0.232	0.134
2.480	3.769	1.372	6.009	0.170	0.038	0.470	0.232
0.084	0.045	2.155	2.375	0.195	1.874	0.250	0.469
2.972	0.314	0.568	2.557	0.769	0.052	0.352	0.202
0.069	2.064	0.232	2.024	1.293	0.409	0.280	0.336
0.074	0.176	0.090	1.986	0.087	8.176	0.067	0.178
1.114	0.302	0.085	0.300	3.214	3.545	0.220	0.446
2.845	2.313	1.497	1.281	0.609	4.165	0.027	0.073
3.515	0.266	1.588	1.236	3.147	0.049	0.438	0.340
1.228	0.526	1.027	2.789	0.027	1.409	0.024	0.402
0.381	0.249	2.420	2.935	0.087	1.771	0.218	0.466
0.049	1.542	0.541	2.064	0.414	0.811	0.198	0.564
0.177	1.387	0.057	2.004	0.720	1.851	0.372	0.335
1.536	1.446	1.922	0.367	0.447	1.469	0.340	0.173
0.813	1.074	1.713	3.369	0.711	0.487	0.080	0.372
3.932	11.301	3.585	1.718	0.000	2.479	0.263	0.080
1.295	0.326	0.540	2.067	0.433	1.263	0.484	0.634
2.548	2.248	0.473	0.026	1.862	0.815	0.402	0.517
0.595	0.210	0.286	2.384	3.335	0.786	0.091	0.206
1.821	1.200	0.578	2.192	0.710	0.234	0.082	0.466
0.192	2.034	1.746	16.510	1.087	1.322	0.406	0.110
1.692	1.068	3.272	1.898	1.838	0.411	0.036	0.289
1.070	5.015	11.267	0.572	0.998	5.625	0.111	0.060
0.339	3.974	3.064	1.529	8.874	5.506	0.286	0.060
1.113	1.706	2.458	0.356	0.863	1.263	0.435	0.188
4.498	1.288	3.720	0.425	1.201	3.433	0.085	0.076
1.905	0.173	0.570	0.226	2.906	3.550	0.228	0.188
0.077	4.551	3.321	0.764	1.035	1.348	0.222	0.086
0.020	2.775	1.788	0.991	1.817	2.055	0.289	0.194
2.571	3.613	0.266	2.327	0.619	0.439	0.374	0.112
2.482	1.317	1.254	0.488	4.178	0.379	0.067	0.148
3.990	3.504	0.200	1.266	0.141	3.237	0.336	0.377
2.486	2.339	4.765	0.211	0.984	1.738	0.258	0.235
0.524	4.604	0.999	1.657	0.937	0.821	0.257	0.446
0.139	7.273	1.650	2.546	1.027	3.687	0.430	0.158
0.449	0.268	0.458	2.406	0.846	4.490	0.322	0.225
0.936	0.684	0.359	2.096	0.000	0.497	0.177	0.412
3.518	0.290	1.859	6.254	0.174	1.042	0.429	0.162
2.298	0.502	2.036	0.217	1.212	0.216	0.494	0.369
0.327	1.049	1.986	0.389	0.020	3.369	0.404	0.450
10.910	1.103	7.541	0.512	4.279	4.694	0.481	0.090

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
1.157	11.772	2.644	0.299	3.591	5.754	0.039	0.028
0.456	0.094	1.533	2.489	0.390	0.520	0.343	0.508
0.882	2.222	0.038	1.345	5.497	1.821	0.292	0.141
1.717	0.424	0.074	1.196	4.268	0.390	0.391	0.200
1.045	0.639	1.799	1.020	2.378	1.884	0.066	0.172
0.598	1.295	2.583	1.698	1.176	0.276	0.444	0.250
0.801	0.230	1.938	0.671	1.243	2.717	0.243	0.165
1.960	6.402	2.927	0.058	0.322	2.366	0.241	0.088
1.303	4.022	1.134	0.303	1.842	0.237	0.083	0.297
0.937	0.467	0.040	2.590	0.745	0.195	0.096	0.432
2.044	0.077	2.101	1.532	5.651	3.988	0.285	0.191
1.917	0.733	2.057	1.552	0.352	1.036	0.343	0.553
1.691	1.063	0.391	2.377	1.268	0.344	0.201	0.934
1.584	1.751	1.487	0.275	0.479	2.128	0.061	0.196
1.286	2.698	0.302	1.180	1.980	1.164	0.332	0.132
5.395	5.318	1.438	0.299	4.296	0.713	0.277	0.204
0.051	1.151	3.295	0.612	2.793	0.222	0.456	0.328
0.544	1.344	4.223	0.450	0.424	0.382	0.054	0.306
1.019	1.699	3.723	1.116	0.299	0.373	0.130	0.289
3.770	0.402	1.289	0.567	1.217	0.649	0.261	0.288
0.604	2.671	1.911	0.510	1.648	1.638	0.192	0.222
2.058	2.215	0.793	0.561	1.308	1.504	0.482	0.244
0.901	0.233	2.130	3.239	0.236	1.658	0.292	0.467
4.853	1.215	4.885	1.246	3.841	3.958	0.150	0.130
0.218	1.617	1.664	3.052	0.111	0.280	0.170	0.513
1.698	0.775	2.088	1.434	0.462	0.468	0.434	0.339
1.994	0.215	3.221	1.879	0.710	1.468	0.486	0.285
0.989	0.589	2.609	1.745	0.493	0.586	0.454	0.194
0.602	0.418	3.142	3.191	0.017	0.137	0.142	0.530
0.180	0.839	0.589	1.777	1.345	2.775	0.452	0.301
0.278	2.298	0.260	2.381	0.530	0.739	0.172	0.352
1.543	1.771	1.213	1.152	0.216	3.045	0.130	0.198
0.153	0.583	1.775	8.057	3.602	2.311	0.314	0.210
1.804	13.606	4.610	1.014	9.289	19.121	0.307	0.044
0.696	0.074	1.972	2.076	1.295	1.681	0.478	0.261
1.831	2.125	1.399	1.624	0.896	0.382	0.471	0.337
4.750	0.426	1.292	0.705	0.769	0.015	0.441	0.206
3.230	3.581	0.478	0.078	0.228	0.787	0.407	0.251
1.361	0.706	1.175	2.087	0.129	2.319	0.059	0.246
4.388	6.281	1.350	0.350	4.637	0.290	0.141	0.197
0.336	1.689	1.594	1.477	3.595	0.204	0.248	0.281
33.234	8.422	0.117	7.328	8.091	4.286	0.275	0.031
1.858	0.148	0.426	2.181	0.311	2.380	0.231	0.368
4.434	11.447	3.737	0.949	0.926	7.002	0.180	0.047

Table A.3: Parameter Estimates of 100 Simulations in The Codon Length of 10^5 via Nmkb Method.

Parameter Estimates of 90 Species Pairs

Species	σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
01FcaCaf	0.866	2.256	0.589	1.100	2.382	0.793	0.200	0.226
02MusRat	0.872	2.381	0.668	0.767	2.531	0.789	0.173	0.184
03HsaMmu	0.838	2.230	0.700	1.002	2.403	0.840	0.288	0.081
04Gallus	0.852	2.197	0.797	1.015	2.373	0.823	0.252	0.122
05Droso	1.024	1.888	1.200	0.818	2.108	1.070	0.138	0.046
07yeast	0.642	2.878	0.464	1.043	3.022	0.533	0.122	0.344
08Arab	1.133	1.909	0.941	1.169	2.060	0.964	0.236	0.141
09SarMon	0.975	2.155	0.765	1.050	2.282	0.834	0.195	0.336
10ants	0.610	2.727	0.623	0.637	3.034	0.502	0.179	0.471
11malaria	1.035	2.095	0.494	1.633	2.398	0.908	0.247	0.535
12fugu	1.009	1.964	0.905	1.121	2.091	0.911	0.188	0.390
13Phytophthora	0.910	2.184	1.029	0.785	2.311	0.829	0.120	0.554
14Fusarium	0.817	2.508	0.597	0.667	2.697	0.718	0.168	0.133
15Oryza	1.156	1.656	1.156	1.193	1.750	1.137	0.553	0.021

Species	σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
16Solanum	1.105	1.979	0.915	1.292	2.168	0.871	0.297	0.112
ATGC001	1.269	1.880	0.956	0.990	2.175	0.799	0.068	0.905
ATGC013	1.401	1.787	0.951	1.491	1.748	0.898	0.100	1.076
ATGC015	1.528	1.866	0.822	1.253	1.705	0.932	0.083	1.347
ATGC016	0.885	2.390	0.383	0.860	2.813	0.710	0.097	0.655
ATGC021	1.194	2.681	0.494	0.765	2.219	0.987	0.085	0.981
ATGC022	1.254	2.523	0.529	0.987	2.284	0.926	0.096	0.879
ATGC028	1.153	1.895	1.287	1.621	1.366	0.715	0.121	0.702
ATGC044	2.026	2.453	0.272	1.040	2.625	1.360	0.150	0.418
ATGC045	1.605	2.924	0.526	1.547	2.436	0.704	0.125	1.322
ATGC050	1.220	2.589	0.443	1.074	2.726	0.716	0.124	0.695
ATGC056	1.428	2.177	0.582	1.251	1.898	0.805	0.050	1.566
ATGC058	1.551	2.370	0.727	1.059	2.321	0.624	0.062	1.046
ATGC069	1.418	1.893	1.166	0.959	1.651	0.989	0.084	1.049
ATGC071	1.033	2.030	1.010	1.333	2.026	0.632	0.113	0.584
ATGC073	1.193	1.965	1.078	1.434	1.679	0.678	0.134	0.771
ATGC075	1.164	1.971	0.948	1.090	2.047	0.861	0.093	0.764
ATGC078	1.026	1.950	0.976	1.283	2.149	0.649	0.102	0.634
ATGC095	0.796	2.884	0.558	0.697	2.601	0.577	0.102	0.952
ATGC099	1.456	1.816	0.959	1.229	1.925	0.665	0.092	1.012
ATGC110	1.313	1.955	0.916	1.138	1.925	0.848	0.056	1.174
ATGC111	1.223	1.951	1.030	1.239	1.882	0.744	0.120	0.606
ATGC112	1.501	1.870	1.037	1.218	1.959	0.774	0.064	1.088
ATGC117	1.290	2.137	0.567	1.171	2.277	0.839	0.076	0.895
ATGC120	1.382	1.962	0.875	1.223	1.901	0.762	0.063	1.215
ATGC127	1.367	2.069	0.832	1.077	2.034	0.716	0.068	1.207
ATGC130	1.162	2.582	0.974	1.061	1.606	0.885	0.122	0.684
ATGC134	1.041	2.062	0.842	1.189	2.356	0.674	0.113	0.550
ATGC136	1.067	2.128	0.895	1.050	1.935	0.973	0.078	0.837
ATGC138	1.438	2.369	0.907	1.654	2.081	0.588	0.063	1.341
ATGC143	1.623	2.221	0.913	1.606	2.318	0.683	0.094	0.876
ATGC145	1.714	3.349	0.268	0.927	2.871	1.008	0.140	0.653
ATGC146	0.993	2.458	0.818	1.007	1.861	0.766	0.084	1.078
ATGC152	1.129	2.392	0.686	1.066	1.739	0.808	0.094	0.966
ATGC155	2.027	2.373	0.772	1.669	2.585	0.855	0.148	0.793
ATGC159	0.876	2.744	0.792	0.791	2.294	0.789	0.073	0.969
ATGC169	1.341	2.294	1.093	1.184	1.762	0.724	0.130	0.674
ATGC171	1.170	2.446	0.703	1.365	2.601	0.780	0.092	0.765
ATGC173	1.161	2.002	1.163	1.322	1.763	0.771	0.152	0.612
ATGC178	1.591	1.944	1.017	1.219	2.345	0.886	0.106	0.605
ATGC181	1.402	2.173	0.924	1.632	1.897	0.772	0.086	1.347
ATGC182	1.051	2.316	0.995	0.928	2.339	0.810	0.066	1.173
ATGC186	1.692	1.777	0.687	1.769	1.898	1.025	0.082	1.023
ATGC188	0.957	1.995	1.009	1.502	1.934	0.587	0.123	0.543
ATGC201	1.299	3.111	0.446	0.673	2.338	0.664	0.137	1.108
ATGC202	1.517	1.839	0.537	1.148	1.546	1.375	0.104	0.860
ATGC223	1.994	2.478	0.881	0.994	2.192	0.920	0.136	1.111
ATGC232	0.979	2.299	0.975	0.963	2.043	0.919	0.088	0.771
ATGC235	1.421	1.901	1.030	1.334	1.953	0.696	0.073	1.032
ATGC239	1.038	2.232	0.779	1.092	2.329	0.594	0.083	0.798
ATGC242	1.314	1.961	0.890	1.102	2.072	0.671	0.103	0.904
ATGC243	1.327	1.902	0.972	1.229	1.867	0.745	0.083	1.297
ATGC244	1.398	2.230	0.628	1.127	2.128	0.736	0.082	1.129
ATGC246	1.499	2.904	0.359	0.768	2.329	0.942	0.188	0.680
ATGC258	0.947	3.079	0.554	0.854	2.818	0.607	0.070	1.078
ATGC260	1.168	3.201	0.509	1.520	2.910	0.897	0.114	0.779
ATGC277	1.069	2.337	0.991	1.165	1.796	0.805	0.136	0.583
ATGC279	1.012	2.717	0.571	1.070	2.177	0.627	0.059	1.825
ATGC301	1.124	2.168	1.043	1.115	1.994	0.732	0.113	0.760
ATGC309	1.516	2.189	1.011	1.489	2.600	0.603	0.078	0.852
ATGC330	1.148	2.138	1.099	0.965	1.920	0.991	0.077	0.921
ATGC332	1.197	1.847	1.395	1.755	1.017	0.778	0.173	0.652
ATGC339	1.084	1.982	1.032	1.247	1.960	0.807	0.084	0.963
ATGC341	1.187	2.161	1.087	0.999	1.865	0.856	0.114	0.834
ATGC352	1.139	2.131	0.919	1.395	1.973	0.562	0.208	0.362
ATGC361	1.160	2.016	1.033	0.900	2.054	0.899	0.113	0.590
ATGC369	1.459	2.009	0.822	1.231	2.403	0.901	0.081	0.844
ATGC371	1.104	2.416	0.744	1.177	2.233	0.706	0.063	1.445
ATGC372	0.857	2.455	0.750	1.347	1.837	0.649	0.082	1.298
ATGC374	1.281	1.923	1.189	1.243	1.479	1.065	0.066	1.161
ATGC376	1.333	1.881	1.236	1.147	1.649	0.904	0.099	0.979
ATGC389	1.736	1.824	1.027	1.487	1.756	0.671	0.073	1.459

Species	σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ
ATGC393	1.034	2.133	0.960	0.997	2.306	0.680	0.092	0.672
ATGC420	1.272	1.842	1.028	1.511	1.638	0.699	0.135	0.809
ATGC428	1.431	1.927	1.214	1.068	1.714	0.792	0.066	1.355
ATGC431	1.584	1.861	0.873	1.405	1.732	0.749	0.061	1.399

Table A.4: Parameter Estimates of 90 Species Pairs.

APPENDIX B

SUPPLEMENTARY MATERIAL FOR CHAPTER 3

File Size Change in the Pipeline

Pipeline step	File size
Homologous genes	14547
Sequence alignments	13827
Gaps curation	2034, 1712, 1652, 1605

Table B.1: **File Size Change in the Pipeline.** | The bottom step has four different file sizes because we use four window sizes.

Phase Proportions of Insertion and Deletion Events across Four Window Sizes

	window size of 3			**window size of 6**			**window size of 9**			**window size of 12**		
	P0	P1	P2	P0	P1	P2	P0	P1	P2	P0	P1	P2
Insertion	0.621	0.145	0.234	0.499	0.210	0.290	0.467	0.239	0.294	0.452	0.253	0.295
Deletion	0.600	0.150	0.250	0.521	0.188	0.291	0.500	0.202	0.298	0.490	0.209	0.301
P.value	0.283	0.730	0.349	0.345	0.219	0.972	0.139	0.044	0.845	0.093	0.020	0.766

Table B.2: **Phase Proportions of Insertion and Deletion Events across Four Window Sizes.** | Insertion and Deletion rows represent the proportion of indel phases in each window size. The P.value row represent the p value of testing the null that the proportion of a specific phase is equal between insertion and deletion groups.

APPENDIX C

SUPPLEMENTARY MATERIAL FOR CHAPTER 4

True Parameters for 100 Simulations

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	ext.I	ext.D	r0	r1	r2
1.016	1.902	0.716	0.969	1.811	1.622	0.076	0.077	0.594	0.734	0.005	0.012	0.010
0.731	1.216	1.024	1.457	1.564	1.969	0.207	0.072	0.722	0.385	0.010	0.007	0.004
0.435	2.188	1.564	0.381	2.275	0.834	0.074	0.080	0.266	0.563	0.009	0.006	0.004
1.189	1.106	1.205	1.912	0.738	1.652	0.177	0.078	0.689	0.178	0.008	0.006	0.005
0.844	2.863	1.576	1.004	1.693	0.675	0.398	0.066	0.746	0.616	0.005	0.009	0.008
0.890	0.599	0.948	2.153	1.986	1.468	0.055	0.080	0.681	0.688	0.005	0.007	0.006
0.487	1.172	1.886	1.510	1.714	1.105	0.310	0.078	0.642	0.213	0.006	0.010	0.006
0.880	1.311	0.749	1.554	2.591	1.596	0.220	0.062	0.685	0.543	0.005	0.007	0.005
1.241	1.327	1.777	0.854	1.563	0.904	0.239	0.072	0.498	0.712	0.008	0.010	0.004
1.376	1.678	1.823	0.383	1.125	0.892	0.041	0.082	0.696	0.749	0.006	0.008	0.008
1.272	2.370	0.826	1.614	0.761	1.019	0.249	0.078	0.684	0.703	0.005	0.007	0.009
1.237	1.721	1.823	0.572	0.969	1.812	0.101	0.053	0.692	0.462	0.007	0.004	0.007
1.125	1.397	1.319	0.958	1.717	1.589	0.129	0.068	0.649	0.494	0.006	0.007	0.008
1.176	1.703	2.507	0.958	2.810	1.758	0.347	0.061	0.161	0.462	0.009	0.009	0.008
1.391	1.419	1.479	1.833	0.490	1.481	0.407	0.083	0.736	0.448	0.011	0.012	0.008
1.086	1.413	0.644	1.772	2.417	2.531	0.228	0.058	0.578	0.574	0.008	0.005	0.007
0.679	2.300	1.552	0.529	2.264	0.437	0.473	0.058	0.726	0.525	0.005	0.003	0.008
2.011	1.947	0.836	2.179	2.720	0.781	0.251	0.056	0.358	0.552	0.006	0.004	0.005
0.875	1.089	2.325	0.955	2.009	1.108	0.357	0.079	0.522	0.587	0.011	0.011	0.009
3.602	0.660	1.221	1.113	0.902	1.842	0.283	0.075	0.307	0.750	0.010	0.005	0.010
1.484	0.906	2.225	0.874	0.689	1.238	0.067	0.117	0.686	0.648	0.011	0.011	0.017
1.764	2.008	1.855	1.105	0.637	0.978	0.222	0.069	0.656	0.358	0.009	0.006	0.005
1.209	0.971	2.199	1.267	2.631	1.143	0.030	0.054	0.474	0.732	0.004	0.008	0.004
1.219	2.229	1.353	0.757	0.833	2.596	0.437	0.059	0.625	0.476	0.005	0.006	0.005
1.552	1.844	1.327	0.713	1.261	1.611	0.025	0.100	0.719	0.709	0.010	0.008	0.007
0.387	1.330	1.099	1.947	1.821	1.306	0.219	0.083	0.744	0.310	0.006	0.009	0.009
1.936	1.131	1.306	1.157	2.821	1.157	0.198	0.040	0.731	0.293	0.004	0.005	0.006
0.737	1.509	2.108	1.750	0.890	1.568	0.379	0.043	0.647	0.684	0.003	0.005	0.004
0.453	1.427	0.576	0.969	3.721	1.320	0.335	0.067	0.624	0.664	0.007	0.009	0.006
1.057	0.956	0.868	0.731	1.345	3.039	0.080	0.077	0.325	0.571	0.010	0.008	0.007
1.469	2.891	2.596	0.787	1.024	2.954	0.268	0.046	0.725	0.649	0.005	0.007	0.004
3.495	0.982	3.844	0.734	0.579	2.212	0.484	0.061	0.619	0.723	0.005	0.006	0.008
1.118	1.531	1.410	2.353	1.219	0.457	0.401	0.058	0.627	0.655	0.007	0.006	0.008
1.528	0.543	1.696	1.017	0.878	2.427	0.092	0.083	0.435	0.664	0.009	0.006	0.005
3.219	0.776	0.269	1.117	1.814	0.746	0.083	0.071	0.635	0.278	0.010	0.005	0.005
1.229	3.187	1.281	2.737	0.487	0.877	0.419	0.079	0.690	0.569	0.008	0.005	0.009
0.791	2.341	1.473	0.770	3.592	1.136	0.037	0.064	0.197	0.674	0.008	0.009	0.009
1.627	1.820	1.034	1.079	1.636	1.716	0.114	0.060	0.680	0.494	0.005	0.006	0.007
1.585	1.984	2.930	1.314	0.689	1.935	0.301	0.074	0.717	0.256	0.008	0.005	0.004
1.940	1.224	1.002	1.897	1.329	0.679	0.431	0.057	0.558	0.746	0.003	0.007	0.008
1.295	2.447	1.317	1.648	0.594	0.748	0.088	0.068	0.404	0.713	0.008	0.008	0.006
1.078	0.615	2.450	2.272	1.048	0.544	0.232	0.090	0.727	0.423	0.009	0.008	0.007
0.283	2.204	2.341	1.466	1.098	1.182	0.232	0.051	0.174	0.727	0.005	0.003	0.007
1.746	2.034	1.385	0.503	1.307	1.843	0.296	0.075	0.472	0.268	0.007	0.011	0.008
1.300	2.479	1.399	0.944	0.638	1.186	0.372	0.077	0.418	0.723	0.007	0.009	0.009
1.029	3.289	0.884	0.937	1.522	1.290	0.066	0.083	0.703	0.687	0.007	0.004	0.007
0.475	1.817	1.827	0.698	1.152	1.545	0.343	0.089	0.582	0.493	0.007	0.013	0.009
2.070	1.430	2.229	0.519	1.735	1.185	0.259	0.061	0.493	0.698	0.005	0.008	0.004
1.182	1.956	2.989	1.013	1.143	2.070	0.257	0.052	0.436	0.303	0.007	0.007	0.004
0.807	1.498	1.434	1.081	2.181	1.734	0.429	0.062	0.174	0.256	0.007	0.008	0.004
1.452	1.748	0.594	1.563	1.909	1.034	0.333	0.061	0.680	0.439	0.007	0.003	0.008
1.566	1.788	2.066	1.321	1.908	1.014	0.179	0.052	0.682	0.658	0.005	0.003	0.004
1.443	2.091	0.762	0.778	1.411	3.109	0.425	0.047	0.354	0.363	0.004	0.006	0.004
0.540	1.957	1.768	1.956	0.552	1.495	0.490	0.077	0.723	0.649	0.011	0.005	0.005
1.720	1.066	1.159	1.248	0.690	2.317	0.403	0.057	0.670	0.261	0.007	0.008	0.006
1.453	1.135	1.465	1.562	1.343	1.576	0.476	0.075	0.156	0.610	0.006	0.008	0.005
1.063	2.008	1.285	1.186	0.896	2.703	0.036	0.049	0.614	0.649	0.005	0.007	0.003
2.653	1.036	1.684	0.902	2.178	0.895	0.341	0.066	0.511	0.308	0.006	0.003	0.008
2.196	2.065	0.596	1.206	0.546	1.972	0.298	0.107	0.635	0.666	0.012	0.009	0.008
1.696	1.668	2.453	0.709	1.131	1.451	0.396	0.082	0.672	0.349	0.012	0.006	0.008
1.282	4.071	1.788	0.296	0.639	0.627	0.068	0.071	0.547	0.360	0.007	0.005	0.004
1.421	1.311	0.557	2.748	1.143	1.349	0.444	0.085	0.317	0.714	0.010	0.009	0.006
1.487	0.628	1.072	0.859	2.989	0.970	0.248	0.055	0.706	0.728	0.004	0.008	0.006
1.236	1.431	1.735	1.058	2.081	1.322	0.243	0.065	0.738	0.442	0.005	0.004	0.008
2.179	1.350	1.168	0.986	2.032	0.433	0.086	0.085	0.745	0.551	0.004	0.005	0.011
2.014	0.366	2.271	1.290	2.330	0.613	0.099	0.044	0.625	0.711	0.002	0.002	0.004

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	ext.I	ext.D	r0	r1	r2
1.410	0.565	0.910	1.883	0.948	1.795	0.287	0.090	0.591	0.650	0.005	0.009	0.006
0.880	1.625	0.853	1.024	1.684	2.325	0.347	0.076	0.712	0.592	0.007	0.005	0.008
0.602	1.524	1.342	2.366	0.341	1.528	0.201	0.055	0.681	0.698	0.008	0.004	0.006
1.651	0.561	2.160	1.512	0.501	1.487	0.063	0.092	0.601	0.256	0.007	0.012	0.012
1.628	1.023	2.196	0.823	1.721	1.766	0.340	0.068	0.595	0.606	0.010	0.008	0.009
1.146	2.398	1.266	0.945	1.475	1.781	0.278	0.045	0.639	0.655	0.005	0.004	0.007
1.027	0.637	1.806	2.433	0.985	1.094	0.459	0.089	0.449	0.749	0.007	0.007	0.006
1.036	1.501	0.618	1.203	2.347	1.364	0.056	0.071	0.595	0.728	0.005	0.009	0.004
1.569	0.955	0.670	1.347	2.500	0.926	0.130	0.069	0.736	0.600	0.008	0.006	0.008
1.173	1.001	1.471	1.182	1.663	1.481	0.260	0.071	0.436	0.505	0.007	0.010	0.010
1.033	1.408	1.943	1.594	1.339	0.814	0.194	0.076	0.677	0.501	0.004	0.010	0.010
1.149	0.836	1.610	1.094	1.336	1.868	0.485	0.066	0.342	0.642	0.007	0.007	0.008
1.224	1.684	0.298	2.597	1.452	3.613	0.293	0.033	0.668	0.690	0.005	0.003	0.004
1.389	1.415	1.113	2.317	0.164	0.323	0.156	0.111	0.349	0.470	0.006	0.008	0.016
1.645	1.341	0.836	1.705	0.789	1.458	0.168	0.080	0.633	0.703	0.007	0.005	0.005
0.989	0.551	2.263	1.194	0.740	1.304	0.430	0.089	0.628	0.677	0.008	0.008	0.009
0.428	1.195	0.933	1.817	1.485	1.776	0.482	0.109	0.673	0.494	0.015	0.013	0.007
1.226	2.100	1.785	0.888	1.441	0.264	0.449	0.096	0.686	0.477	0.011	0.012	0.007
1.133	1.640	0.932	1.466	1.253	1.484	0.145	0.061	0.620	0.551	0.008	0.009	0.004
1.697	0.815	0.829	0.393	3.603	1.623	0.448	0.065	0.686	0.611	0.003	0.004	0.005
1.471	1.558	1.802	0.828	0.965	2.043	0.175	0.070	0.661	0.679	0.004	0.007	0.006
1.272	1.721	0.860	1.608	0.719	2.581	0.135	0.064	0.706	0.562	0.008	0.007	0.009
0.642	1.202	1.180	3.236	2.872	1.478	0.317	0.069	0.683	0.678	0.006	0.008	0.009
0.915	2.820	1.265	0.750	1.852	3.198	0.307	0.061	0.583	0.347	0.007	0.004	0.005
1.276	2.612	0.905	1.189	1.724	0.538	0.474	0.069	0.410	0.723	0.004	0.010	0.009
2.754	2.000	0.566	1.022	1.254	0.662	0.464	0.067	0.733	0.709	0.004	0.009	0.008
0.966	0.946	0.894	1.304	1.649	2.465	0.459	0.079	0.548	0.408	0.008	0.010	0.007
1.531	0.953	1.028	0.658	2.293	1.669	0.411	0.069	0.629	0.714	0.005	0.010	0.007
1.073	1.651	1.217	1.626	1.259	1.122	0.059	0.068	0.732	0.728	0.009	0.009	0.005
1.722	1.450	0.876	1.362	1.356	1.578	0.145	0.082	0.421	0.538	0.008	0.011	0.005
1.728	1.102	0.988	1.349	1.422	1.531	0.251	0.078	0.621	0.719	0.006	0.009	0.009
0.537	1.411	2.039	0.532	1.238	1.096	0.270	0.132	0.663	0.697	0.019	0.008	0.011
1.303	2.511	0.459	1.188	1.251	1.174	0.234	0.060	0.723	0.588	0.007	0.008	0.008
0.510	2.292	1.251	1.842	0.431	1.520	0.177	0.082	0.243	0.644	0.009	0.005	0.004

Table C.1: **True Parameters for 100 Simulations.** | ext.I and ext.D are gap extension weight of insertions and deletions, with codon length as the unit; r0, r1, r2 are the indel phase rates over the branch length.

100 Parameter Estimates via Gillespie Simulation Method

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	ext.I	ext.D	r0	r1	r2
1.046	1.814	0.747	0.987	1.868	1.581	0.079	0.076	0.597	0.735	0.004	0.012	0.011
0.742	1.148	1.046	1.425	1.606	2.023	0.213	0.072	0.730	0.399	0.010	0.007	0.004
0.468	2.138	1.578	0.421	2.259	0.810	0.077	0.079	0.248	0.551	0.009	0.006	0.004
1.265	1.052	1.231	1.892	0.740	1.623	0.184	0.075	0.687	0.200	0.007	0.007	0.005
0.880	2.798	1.585	1.024	1.674	0.660	0.401	0.066	0.755	0.614	0.005	0.009	0.008
0.889	0.590	0.911	2.226	1.953	1.476	0.058	0.080	0.685	0.697	0.005	0.007	0.006
0.520	1.075	1.891	1.523	1.754	1.096	0.315	0.076	0.663	0.223	0.006	0.009	0.006
0.888	1.295	0.739	1.630	2.567	1.553	0.227	0.062	0.697	0.557	0.005	0.007	0.005
1.260	1.271	1.781	0.961	1.600	0.892	0.250	0.070	0.509	0.720	0.008	0.010	0.004
1.475	1.526	1.827	0.399	1.124	0.951	0.043	0.080	0.717	0.758	0.006	0.009	0.008
1.269	2.287	0.834	1.607	0.770	1.038	0.254	0.077	0.690	0.721	0.005	0.006	0.008
1.253	1.655	1.813	0.600	0.972	1.842	0.109	0.052	0.692	0.479	0.007	0.004	0.007
1.122	1.308	1.331	0.942	1.810	1.632	0.134	0.066	0.648	0.495	0.006	0.007	0.008
1.151	1.772	2.470	0.945	2.775	1.779	0.353	0.060	0.163	0.476	0.009	0.009	0.008
1.396	1.419	1.479	1.892	0.486	1.481	0.415	0.082	0.743	0.470	0.011	0.012	0.008
1.109	1.301	0.631	1.773	2.524	2.505	0.232	0.058	0.576	0.581	0.007	0.005	0.006
0.690	2.225	1.564	0.562	2.299	0.443	0.482	0.058	0.720	0.523	0.005	0.003	0.008
2.118	1.847	0.840	2.168	2.818	0.753	0.267	0.053	0.402	0.552	0.006	0.004	0.005
0.856	1.026	2.398	0.959	2.069	1.134	0.366	0.078	0.523	0.592	0.011	0.011	0.009
3.763	0.678	1.284	1.114	0.844	1.792	0.284	0.075	0.317	0.764	0.011	0.004	0.011
1.517	0.838	2.242	0.870	0.698	1.249	0.068	0.115	0.700	0.654	0.011	0.011	0.017
1.817	1.911	1.849	1.133	0.654	0.950	0.235	0.067	0.658	0.383	0.010	0.007	0.004

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	ext.I	ext.D	r0	r1	r2
1.283	0.982	2.190	1.276	2.586	1.119	0.029	0.054	0.474	0.738	0.004	0.008	0.004
1.213	2.226	1.326	0.759	0.832	2.552	0.445	0.059	0.632	0.487	0.006	0.006	0.006
1.612	1.700	1.297	0.731	1.294	1.663	0.026	0.100	0.727	0.717	0.009	0.008	0.007
0.373	1.285	1.134	1.916	1.924	1.287	0.223	0.082	0.756	0.316	0.006	0.009	0.009
1.945	1.100	1.265	1.181	2.805	1.083	0.195	0.040	0.737	0.298	0.004	0.005	0.006
0.739	1.513	2.117	1.746	0.909	1.532	0.373	0.044	0.656	0.677	0.003	0.005	0.004
0.448	1.403	0.581	0.978	3.712	1.288	0.336	0.066	0.642	0.653	0.007	0.009	0.006
1.039	0.897	0.890	0.714	1.328	3.095	0.082	0.077	0.332	0.575	0.010	0.008	0.007
1.502	2.886	2.626	0.789	0.992	2.984	0.266	0.046	0.714	0.645	0.005	0.007	0.005
3.631	1.011	3.712	0.742	0.572	2.135	0.484	0.061	0.638	0.729	0.005	0.006	0.008
1.125	1.505	1.421	2.359	1.204	0.481	0.403	0.058	0.637	0.665	0.007	0.006	0.008
1.549	0.502	1.736	1.031	0.892	2.424	0.096	0.082	0.459	0.684	0.009	0.006	0.005
3.277	0.729	0.299	1.135	1.801	0.738	0.084	0.071	0.646	0.289	0.010	0.005	0.005
1.344	3.161	1.257	2.890	0.502	0.873	0.426	0.077	0.706	0.579	0.008	0.005	0.009
0.749	2.330	1.470	0.766	3.715	1.113	0.040	0.063	0.209	0.688	0.009	0.009	0.009
1.699	1.834	0.968	1.093	1.581	1.721	0.113	0.060	0.680	0.494	0.005	0.005	0.008
1.645	1.954	2.936	1.322	0.645	1.927	0.304	0.073	0.726	0.269	0.007	0.005	0.004
1.925	1.173	0.998	1.949	1.373	0.658	0.432	0.057	0.584	0.749	0.003	0.007	0.008
1.291	2.425	1.398	1.680	0.596	0.706	0.094	0.068	0.413	0.727	0.008	0.008	0.006
1.066	0.607	2.512	2.246	1.065	0.564	0.236	0.090	0.736	0.435	0.009	0.008	0.007
0.264	2.258	2.245	1.501	1.075	1.131	0.235	0.051	0.169	0.731	0.005	0.003	0.007
1.787	2.044	1.299	0.514	1.281	1.805	0.306	0.074	0.484	0.282	0.006	0.011	0.008
1.336	2.454	1.430	0.945	0.629	1.197	0.382	0.074	0.418	0.735	0.007	0.009	0.009
1.130	3.155	0.871	1.016	1.615	1.245	0.070	0.079	0.717	0.701	0.007	0.004	0.007
0.494	1.730	1.851	0.685	1.141	1.549	0.336	0.089	0.588	0.508	0.007	0.013	0.009
2.126	1.430	2.181	0.515	1.722	1.145	0.268	0.060	0.502	0.707	0.005	0.008	0.004
1.181	1.930	3.021	1.035	1.136	2.056	0.257	0.052	0.445	0.307	0.007	0.007	0.004
0.871	1.402	1.451	1.093	2.180	1.706	0.433	0.062	0.206	0.262	0.008	0.009	0.004
1.425	1.752	0.615	1.571	1.876	1.068	0.330	0.060	0.690	0.445	0.007	0.003	0.007
1.558	1.736	2.356	1.311	1.923	1.020	0.177	0.053	0.693	0.664	0.005	0.003	0.004
1.463	2.052	0.785	0.749	1.432	3.022	0.432	0.046	0.370	0.375	0.004	0.007	0.004
0.564	1.907	1.773	2.028	0.552	1.499	0.491	0.076	0.729	0.652	0.011	0.005	0.005
1.755	1.041	1.176	1.254	0.700	2.275	0.410	0.056	0.681	0.270	0.007	0.008	0.006
1.523	1.124	1.514	1.559	1.336	1.519	0.482	0.075	0.161	0.614	0.006	0.008	0.005
1.030	1.844	1.331	1.213	0.938	2.725	0.038	0.049	0.633	0.627	0.005	0.006	0.003
2.748	1.005	1.681	0.938	2.178	0.870	0.343	0.066	0.535	0.307	0.006	0.003	0.007
2.255	2.092	0.647	1.214	0.541	1.925	0.291	0.108	0.653	0.682	0.012	0.009	0.008
1.693	1.665	2.425	0.720	1.121	1.427	0.405	0.080	0.676	0.358	0.013	0.007	0.008
1.353	3.832	1.818	0.313	0.690	0.592	0.075	0.069	0.541	0.371	0.008	0.005	0.004
1.466	1.308	0.573	2.765	1.114	1.358	0.445	0.084	0.336	0.724	0.010	0.009	0.006
1.499	0.617	1.044	0.836	2.967	0.978	0.261	0.054	0.708	0.728	0.004	0.008	0.006
1.238	1.379	1.760	1.069	2.109	1.291	0.242	0.066	0.741	0.449	0.005	0.004	0.008
2.259	1.294	1.149	0.994	2.071	0.403	0.089	0.084	0.750	0.546	0.005	0.005	0.011
1.950	0.393	2.330	1.285	2.400	0.571	0.098	0.044	0.638	0.710	0.003	0.002	0.005
1.428	0.566	0.927	1.879	0.924	1.781	0.289	0.089	0.593	0.648	0.006	0.009	0.006
0.851	1.585	0.851	1.024	1.726	2.308	0.363	0.074	0.734	0.584	0.007	0.005	0.008
0.597	1.438	1.394	2.418	0.342	1.511	0.205	0.055	0.679	0.709	0.008	0.004	0.006
1.638	0.532	2.142	1.547	0.500	1.514	0.064	0.091	0.615	0.272	0.007	0.013	0.012
1.686	1.024	2.081	0.807	1.671	1.754	0.343	0.069	0.590	0.624	0.010	0.008	0.009
1.133	2.203	1.314	0.823	1.514	1.783	0.293	0.044	0.643	0.646	0.005	0.004	0.007
1.023	0.623	1.849	2.429	0.983	1.083	0.465	0.090	0.464	0.741	0.007	0.007	0.005
1.048	1.418	0.617	1.237	2.382	1.365	0.058	0.069	0.605	0.723	0.005	0.009	0.004
1.638	0.933	0.640	1.363	2.509	0.876	0.134	0.069	0.754	0.619	0.008	0.005	0.008
1.141	0.988	1.506	1.147	1.637	1.525	0.260	0.072	0.450	0.529	0.007	0.010	0.010
1.058	1.367	1.924	1.657	1.313	0.789	0.193	0.077	0.695	0.516	0.004	0.010	0.010
1.145	0.856	1.622	1.101	1.292	1.856	0.486	0.065	0.341	0.640	0.007	0.007	0.007
1.280	1.586	0.320	2.594	1.444	3.466	0.303	0.032	0.666	0.681	0.005	0.004	0.004
1.410	1.416	1.143	2.306	0.170	0.324	0.160	0.110	0.366	0.471	0.006	0.008	0.016
1.628	1.277	0.839	1.750	0.791	1.453	0.172	0.078	0.643	0.701	0.007	0.005	0.005
1.017	0.531	2.259	1.200	0.748	1.261	0.448	0.087	0.636	0.687	0.009	0.008	0.009
0.431	1.187	0.936	1.810	1.475	1.747	0.485	0.109	0.695	0.512	0.015	0.013	0.007
1.195	2.098	1.795	0.917	1.450	0.270	0.451	0.096	0.699	0.483	0.011	0.012	0.007
1.146	1.550	0.885	1.499	1.270	1.517	0.148	0.060	0.624	0.553	0.007	0.009	0.004
1.731	0.801	0.866	0.386	3.478	1.572	0.448	0.066	0.708	0.604	0.004	0.003	0.005
1.514	1.510	1.760	0.849	0.985	1.999	0.177	0.069	0.662	0.696	0.004	0.007	0.006
1.287	1.663	0.878	1.672	0.730	2.501	0.142	0.062	0.701	0.570	0.008	0.007	0.009
0.661	1.098	1.182	3.291	2.856	1.539	0.311	0.069	0.687	0.681	0.006	0.008	0.009
0.903	2.822	1.380	0.745	1.839	3.121	0.317	0.060	0.583	0.362	0.007	0.004	0.005
1.297	2.628	0.864	1.192	1.717	0.551	0.472	0.070	0.404	0.728	0.004	0.010	0.009
2.792	1.943	0.577	1.050	1.285	0.645	0.482	0.065	0.738	0.730	0.004	0.009	0.007
0.945	0.970	0.895	1.286	1.663	2.419	0.470	0.077	0.557	0.400	0.008	0.010	0.006
1.525	0.939	1.026	0.624	2.346	1.750	0.414	0.069	0.637	0.739	0.006	0.010	0.008

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	ext.I	ext.D	r0	r1	r2
1.153	1.526	1.255	1.581	1.292	1.138	0.059	0.067	0.738	0.739	0.010	0.009	0.005
1.780	1.482	0.888	1.377	1.344	1.508	0.149	0.080	0.431	0.545	0.008	0.011	0.005
1.739	1.083	1.003	1.381	1.412	1.456	0.246	0.079	0.623	0.720	0.006	0.009	0.009
0.560	1.263	2.059	0.509	1.299	1.083	0.271	0.130	0.682	0.706	0.019	0.008	0.011
1.285	2.413	0.445	1.264	1.268	1.165	0.241	0.060	0.730	0.585	0.007	0.008	0.008
0.523	2.284	1.285	1.840	0.438	1.511	0.178	0.082	0.228	0.635	0.009	0.005	0.005

Table C.2: 100 Parameter Estimates via Gillespie Simulation Method.

100 Parameter Estimates via Importance Sampling Method within EM

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	ext.I	ext.D	r0	r1	r2
1.045	1.815	0.747	0.993	1.864	1.581	0.079	0.076	0.599	0.735	0.004	0.012	0.011
0.740	1.154	1.042	1.430	1.607	2.016	0.213	0.072	0.730	0.403	0.010	0.007	0.004
0.469	2.134	1.575	0.420	2.257	0.819	0.077	0.079	0.256	0.555	0.009	0.006	0.004
1.267	1.049	1.238	1.892	0.740	1.619	0.184	0.074	0.692	0.212	0.007	0.006	0.005
0.883	2.800	1.592	1.022	1.673	0.657	0.401	0.066	0.755	0.613	0.005	0.009	0.008
0.894	0.591	0.914	2.224	1.949	1.475	0.058	0.080	0.689	0.701	0.005	0.007	0.006
0.519	1.080	1.889	1.532	1.755	1.094	0.315	0.076	0.666	0.238	0.006	0.010	0.006
0.891	1.295	0.740	1.628	2.558	1.556	0.226	0.062	0.699	0.557	0.005	0.007	0.005
1.261	1.271	1.784	0.951	1.600	0.888	0.250	0.070	0.514	0.721	0.008	0.010	0.004
1.474	1.524	1.827	0.400	1.121	0.959	0.043	0.080	0.717	0.759	0.006	0.008	0.009
1.269	2.284	0.834	1.606	0.770	1.042	0.253	0.078	0.694	0.723	0.005	0.006	0.008
1.248	1.661	1.820	0.600	0.975	1.832	0.109	0.052	0.695	0.487	0.007	0.004	0.007
1.127	1.310	1.329	0.945	1.809	1.624	0.133	0.066	0.648	0.499	0.006	0.007	0.008
1.146	1.775	2.471	0.948	2.767	1.778	0.350	0.061	0.163	0.476	0.008	0.009	0.008
1.391	1.423	1.476	1.896	0.484	1.483	0.415	0.082	0.743	0.470	0.011	0.012	0.008
1.107	1.300	0.633	1.784	2.522	2.501	0.231	0.058	0.577	0.580	0.007	0.005	0.006
0.692	2.226	1.562	0.564	2.300	0.444	0.483	0.058	0.722	0.533	0.004	0.004	0.008
2.120	1.845	0.840	2.167	2.813	0.756	0.267	0.053	0.406	0.554	0.006	0.004	0.005
0.852	1.028	2.396	0.960	2.066	1.138	0.365	0.078	0.521	0.594	0.011	0.011	0.009
3.759	0.676	1.283	1.114	0.841	1.801	0.284	0.075	0.320	0.764	0.011	0.005	0.011
1.517	0.841	2.241	0.870	0.697	1.249	0.069	0.115	0.701	0.653	0.011	0.010	0.017
1.810	1.911	1.850	1.132	0.655	0.959	0.235	0.067	0.660	0.388	0.010	0.006	0.004
1.290	0.984	2.195	1.274	2.585	1.116	0.029	0.054	0.480	0.739	0.004	0.008	0.005
1.217	2.223	1.328	0.757	0.829	2.556	0.445	0.059	0.638	0.500	0.006	0.006	0.006
1.618	1.698	1.301	0.725	1.295	1.663	0.026	0.100	0.727	0.718	0.009	0.008	0.007
0.371	1.290	1.134	1.915	1.925	1.284	0.223	0.082	0.757	0.320	0.006	0.009	0.009
1.944	1.100	1.279	1.182	2.804	1.078	0.195	0.040	0.738	0.316	0.004	0.005	0.006
0.739	1.513	2.127	1.744	0.910	1.534	0.372	0.044	0.657	0.678	0.003	0.005	0.004
0.446	1.404	0.581	0.972	3.712	1.293	0.338	0.066	0.645	0.654	0.007	0.009	0.006
1.041	0.899	0.887	0.708	1.333	3.094	0.082	0.076	0.332	0.578	0.010	0.008	0.007
1.501	2.891	2.631	0.791	0.990	2.975	0.267	0.046	0.714	0.647	0.005	0.007	0.005
3.630	1.014	3.720	0.740	0.572	2.137	0.484	0.061	0.639	0.730	0.005	0.006	0.008
1.123	1.505	1.422	2.360	1.205	0.481	0.403	0.058	0.641	0.667	0.007	0.006	0.008
1.549	0.496	1.734	1.032	0.901	2.421	0.096	0.082	0.467	0.686	0.009	0.006	0.005
3.277	0.735	0.296	1.131	1.807	0.737	0.085	0.070	0.649	0.302	0.010	0.005	0.005
1.349	3.154	1.259	2.878	0.505	0.873	0.426	0.078	0.705	0.580	0.007	0.005	0.010
0.755	2.314	1.463	0.768	3.720	1.115	0.040	0.063	0.219	0.689	0.008	0.008	0.009
1.692	1.835	0.970	1.092	1.585	1.724	0.113	0.060	0.680	0.500	0.005	0.005	0.008
1.645	1.955	2.929	1.321	0.647	1.927	0.304	0.073	0.729	0.270	0.007	0.005	0.004
1.928	1.175	0.998	1.946	1.370	0.659	0.432	0.057	0.584	0.750	0.003	0.007	0.008
1.292	2.430	1.383	1.679	0.601	0.704	0.095	0.068	0.425	0.727	0.008	0.008	0.006
1.070	0.607	2.512	2.245	1.061	0.565	0.236	0.090	0.737	0.439	0.009	0.008	0.007
0.270	2.255	2.246	1.501	1.076	1.126	0.235	0.051	0.175	0.735	0.005	0.003	0.007
1.790	2.045	1.296	0.512	1.280	1.808	0.305	0.074	0.484	0.289	0.007	0.011	0.008
1.335	2.449	1.434	0.943	0.628	1.199	0.383	0.074	0.420	0.735	0.007	0.009	0.009
1.124	3.151	0.875	1.024	1.611	1.247	0.070	0.079	0.718	0.704	0.007	0.004	0.007
0.500	1.723	1.849	0.682	1.142	1.555	0.336	0.089	0.588	0.509	0.008	0.013	0.008
2.125	1.432	2.180	0.516	1.720	1.143	0.268	0.060	0.510	0.710	0.005	0.008	0.004
1.182	1.934	3.021	1.032	1.131	2.064	0.257	0.052	0.450	0.310	0.007	0.008	0.004
0.875	1.406	1.450	1.098	2.178	1.702	0.433	0.062	0.208	0.265	0.007	0.009	0.004
1.423	1.755	0.619	1.572	1.872	1.065	0.329	0.060	0.691	0.447	0.007	0.003	0.007
1.564	1.738	2.344	1.311	1.919	1.019	0.178	0.053	0.701	0.672	0.005	0.003	0.004
1.467	2.049	0.785	0.756	1.429	3.022	0.430	0.046	0.375	0.381	0.004	0.007	0.004
0.564	1.908	1.773	2.024	0.551	1.501	0.491	0.076	0.729	0.656	0.011	0.005	0.005

σ_{AC}	σ_{AG}	σ_{AT}	σ_{CG}	σ_{CT}	σ_{GT}	ω	τ	ext.I	ext.D	r0	r1	r2
1.758	1.041	1.178	1.256	0.697	2.271	0.411	0.056	0.682	0.281	0.007	0.009	0.006
1.526	1.130	1.500	1.557	1.335	1.520	0.481	0.076	0.158	0.615	0.007	0.008	0.005
1.026	1.844	1.326	1.214	0.936	2.730	0.038	0.049	0.637	0.630	0.005	0.007	0.003
2.749	1.003	1.681	0.939	2.175	0.871	0.343	0.066	0.545	0.323	0.006	0.004	0.007
2.253	2.093	0.647	1.213	0.540	1.930	0.291	0.108	0.653	0.683	0.012	0.009	0.008
1.685	1.668	2.421	0.725	1.115	1.430	0.405	0.080	0.677	0.363	0.013	0.007	0.008
1.350	3.840	1.813	0.311	0.691	0.592	0.075	0.069	0.542	0.375	0.008	0.005	0.004
1.471	1.308	0.572	2.763	1.111	1.359	0.445	0.084	0.344	0.725	0.010	0.009	0.006
1.507	0.616	1.045	0.834	2.964	0.976	0.261	0.054	0.712	0.730	0.004	0.008	0.006
1.239	1.374	1.766	1.072	2.108	1.288	0.242	0.066	0.744	0.460	0.005	0.004	0.008
2.260	1.296	1.144	0.993	2.072	0.402	0.089	0.084	0.751	0.548	0.005	0.005	0.011
1.949	0.393	2.335	1.284	2.399	0.572	0.098	0.044	0.644	0.718	0.003	0.002	0.005
1.428	0.566	0.924	1.880	0.922	1.781	0.290	0.089	0.598	0.650	0.006	0.009	0.007
0.850	1.585	0.851	1.027	1.729	2.305	0.362	0.074	0.737	0.590	0.007	0.005	0.008
0.600	1.437	1.390	2.418	0.342	1.513	0.205	0.055	0.684	0.713	0.008	0.004	0.005
1.633	0.536	2.139	1.549	0.503	1.514	0.064	0.091	0.614	0.271	0.007	0.013	0.012
1.688	1.028	2.074	0.808	1.670	1.751	0.343	0.069	0.592	0.625	0.010	0.008	0.009
1.130	2.210	1.310	0.824	1.511	1.795	0.294	0.044	0.646	0.646	0.005	0.004	0.007
1.022	0.622	1.854	2.426	0.984	1.083	0.465	0.090	0.463	0.741	0.007	0.007	0.005
1.049	1.419	0.617	1.234	2.381	1.368	0.058	0.069	0.610	0.727	0.005	0.009	0.004
1.633	0.935	0.646	1.363	2.507	0.874	0.134	0.069	0.755	0.623	0.009	0.005	0.008
1.146	0.988	1.507	1.144	1.635	1.523	0.260	0.072	0.451	0.532	0.006	0.010	0.010
1.059	1.368	1.923	1.655	1.316	0.788	0.193	0.076	0.695	0.520	0.004	0.010	0.010
1.146	0.858	1.622	1.099	1.289	1.858	0.486	0.065	0.344	0.640	0.007	0.007	0.007
1.292	1.579	0.320	2.594	1.447	3.459	0.302	0.032	0.673	0.685	0.005	0.004	0.004
1.406	1.410	1.149	2.307	0.170	0.331	0.160	0.110	0.366	0.471	0.006	0.008	0.016
1.627	1.280	0.834	1.749	0.791	1.458	0.172	0.078	0.647	0.708	0.007	0.005	0.005
1.022	0.531	2.256	1.198	0.749	1.264	0.448	0.087	0.638	0.691	0.009	0.008	0.009
0.431	1.187	0.940	1.810	1.473	1.747	0.486	0.109	0.698	0.518	0.015	0.013	0.008
1.194	2.101	1.795	0.919	1.449	0.268	0.451	0.096	0.699	0.482	0.011	0.012	0.007
1.144	1.553	0.884	1.492	1.276	1.520	0.149	0.060	0.629	0.562	0.007	0.009	0.004
1.729	0.800	0.865	0.387	3.482	1.572	0.447	0.066	0.712	0.607	0.004	0.004	0.005
1.511	1.510	1.769	0.850	0.985	1.995	0.177	0.069	0.666	0.699	0.004	0.007	0.006
1.285	1.658	0.879	1.672	0.732	2.504	0.142	0.062	0.703	0.572	0.008	0.007	0.009
0.653	1.094	1.184	3.301	2.860	1.540	0.312	0.069	0.688	0.682	0.006	0.008	0.009
0.905	2.817	1.380	0.744	1.835	3.130	0.317	0.060	0.586	0.367	0.006	0.004	0.005
1.299	2.632	0.865	1.190	1.716	0.549	0.472	0.070	0.407	0.728	0.004	0.010	0.009
2.791	1.943	0.576	1.052	1.287	0.645	0.482	0.065	0.740	0.732	0.004	0.009	0.007
0.948	0.962	0.894	1.286	1.663	2.427	0.469	0.078	0.564	0.409	0.008	0.010	0.006
1.520	0.941	1.028	0.626	2.345	1.750	0.413	0.069	0.637	0.741	0.006	0.009	0.008
1.150	1.525	1.258	1.586	1.292	1.135	0.059	0.068	0.740	0.740	0.010	0.009	0.005
1.781	1.478	0.887	1.377	1.343	1.510	0.148	0.080	0.430	0.543	0.008	0.012	0.005
1.736	1.084	1.000	1.384	1.413	1.453	0.246	0.079	0.628	0.721	0.006	0.009	0.008
0.570	1.276	2.043	0.511	1.320	1.097	0.270	0.130	0.684	0.709	0.018	0.008	0.011
1.281	2.412	0.442	1.265	1.274	1.166	0.241	0.060	0.731	0.587	0.007	0.008	0.008
0.519	2.282	1.292	1.840	0.439	1.512	0.178	0.082	0.231	0.637	0.009	0.005	0.005

Table C.3: 100 Parameter Estimates via Importance Sampling Method within EM.

APPENDIX D

SUPPLEMENTARY MATERIAL FOR CHAPTER 5

Pairwise Species Number 1-16 Name Extension Table

Species	Taxon name	Common name
01FcaCaf	Canis lupus familiaris, Felis catus	Dog, Cat
02MusRat	Mus musculus, Rattus norvegicus	Mouse, Rat
03HsaMmu	Homo sapiens, Macaca mulatta	Human, Macaca
04Gallus	Gallus gallus, Meleagris gallopavo	Chicken, Wild turkey
05Droso	Drosophila sechellia, Drosophila simulans	Fruit fly
06Nematode	NA	Roundworm genus
07yeast	Saccharomyces cerevisiae, Saccharomyces paradoxus	Yeast, Wild yeast
08Arab	Arabidopsis thaliana, Arabidopsis lyrata	Thale cress,Sand Cress
09SarMon	Monodelphis domestica, Sarcophilus harrisii	Short-tailed opossum, Tasmanian devil
10ants	Atta cephalotes, Solenopsis invicta	Leaf cutting ant, Red imported fire ant
11malaria	Plasmodium vivax, Plasmodium knowlesi	Malaria parasite genus
12fugu	Takifugu rubripes, Tetraodon nigroviridis	Japanese puffer, Spotted green pufferfish
13Phytophthora	Phytophthora infestans, Phytophthora parasitica	Potato blight, Black shank
14Fusarium	Fusarium graminearum, Fusarium pseudograminearum	Gibberella zeae, Crown rot
15Oryza	Oryza sativa Japonica, Oryza glaberrima	Asian rice, African rice
16Solanum	Solanum tuberosum, Solanum lycopersicum	Potato, Tomato

Table D.1: Pairwise Species Number 1-16 Name Extension Table

Phase Proportions of 90 Species across Three Alignment Methods

	mafft+sw			coati-max			coati-sampling		
Species	P0	P1	P2	P0	P1	P2	P0	P1	P2
01FcaCaf	0.529	0.196	0.275	0.506	0.252	0.242	0.456	0.268	0.276
02MusRat	0.519	0.202	0.280	0.470	0.274	0.256	0.376	0.298	0.326
04Gallus	0.537	0.208	0.254	0.528	0.246	0.226	0.483	0.273	0.244
05Droso	0.611	0.151	0.238	0.479	0.245	0.277	0.354	0.259	0.387
06Nematode	0.529	0.165	0.306	0.452	0.250	0.298	0.457	0.278	0.265
07yeast	0.532	0.167	0.301	0.464	0.263	0.273	0.375	0.290	0.335
08Arab	0.497	0.203	0.300	0.438	0.280	0.282	0.372	0.291	0.337
09SarMon	0.499	0.212	0.289	0.455	0.269	0.276	0.412	0.294	0.294
10ants	0.477	0.203	0.320	0.408	0.284	0.308	0.376	0.297	0.327
11malaria	0.501	0.193	0.306	0.417	0.284	0.300	0.369	0.315	0.316
12fugu	0.524	0.202	0.273	0.474	0.257	0.269	0.443	0.284	0.273
13Phytophthora	0.492	0.186	0.322	0.441	0.271	0.288	0.394	0.288	0.319
14Fusarium	0.535	0.173	0.292	0.510	0.253	0.237	0.399	0.263	0.338
15Oryza	0.630	0.138	0.232	0.417	0.290	0.293	0.416	0.296	0.287
16Solanum	0.473	0.210	0.318	0.434	0.287	0.279	0.382	0.287	0.331
ATGC001	0.491	0.226	0.283	0.492	0.241	0.267	0.508	0.253	0.240
ATGC013	0.410	0.231	0.359	0.530	0.183	0.287	0.549	0.199	0.252
ATGC015	0.500	0.218	0.282	0.474	0.247	0.279	0.523	0.249	0.227
ATGC016	0.548	0.164	0.288	0.505	0.240	0.255	0.501	0.246	0.253
ATGC021	0.506	0.235	0.259	0.488	0.261	0.251	0.491	0.242	0.266
ATGC022	0.452	0.202	0.345	0.468	0.245	0.287	0.478	0.253	0.269
ATGC028	0.490	0.180	0.330	0.447	0.255	0.298	0.453	0.283	0.265
ATGC044	0.414	0.345	0.241	0.433	0.267	0.299	0.425	0.279	0.296
ATGC045	0.531	0.156	0.312	0.459	0.247	0.294	0.474	0.249	0.277
ATGC050	0.481	0.160	0.358	0.499	0.240	0.262	0.478	0.254	0.269
ATGC056	0.449	0.194	0.357	0.568	0.187	0.245	0.576	0.198	0.226
ATGC058	0.534	0.233	0.233	0.533	0.218	0.249	0.513	0.240	0.247
ATGC069	0.549	0.171	0.280	0.499	0.227	0.274	0.511	0.229	0.260
ATGC071	0.538	0.154	0.308	0.450	0.251	0.299	0.466	0.268	0.267
ATGC073	0.507	0.164	0.329	0.474	0.231	0.296	0.462	0.260	0.278
ATGC075	0.539	0.171	0.289	0.504	0.254	0.243	0.489	0.264	0.247
ATGC078	0.562	0.173	0.265	0.496	0.243	0.261	0.488	0.248	0.263
ATGC095	0.562	0.160	0.278	0.520	0.247	0.233	0.533	0.224	0.243
ATGC099	0.571	0.128	0.301	0.514	0.201	0.285	0.515	0.222	0.262
ATGC110	0.586	0.135	0.279	0.503	0.288	0.209	0.525	0.261	0.214
ATGC111	0.443	0.165	0.392	0.509	0.191	0.299	0.484	0.225	0.291
ATGC112	0.487	0.186	0.327	0.485	0.241	0.273	0.533	0.228	0.239
ATGC117	0.520	0.120	0.360	0.491	0.271	0.238	0.513	0.233	0.254
ATGC120	0.556	0.176	0.267	0.475	0.263	0.263	0.504	0.237	0.259
ATGC127	0.412	0.235	0.353	0.487	0.263	0.250	0.498	0.249	0.254
ATGC134	0.500	0.180	0.320	0.459	0.300	0.241	0.453	0.277	0.270
ATGC136	0.590	0.162	0.248	0.512	0.216	0.272	0.517	0.215	0.269
ATGC138	0.613	0.194	0.194	0.462	0.283	0.254	0.517	0.269	0.214
ATGC143	0.459	0.162	0.378	0.500	0.227	0.273	0.510	0.229	0.261
ATGC145	0.535	0.155	0.310	0.443	0.280	0.277	0.446	0.258	0.297
ATGC146	0.484	0.172	0.344	0.460	0.247	0.294	0.502	0.242	0.256
ATGC152	0.545	0.169	0.286	0.526	0.239	0.234	0.505	0.234	0.261
ATGC155	0.605	0.140	0.256	0.502	0.187	0.311	0.517	0.204	0.279
ATGC159	0.418	0.215	0.367	0.451	0.276	0.273	0.458	0.275	0.268
ATGC169	0.347	0.190	0.463	0.425	0.244	0.331	0.438	0.275	0.287
ATGC171	0.604	0.146	0.250	0.513	0.188	0.299	0.502	0.212	0.286
ATGC173	0.492	0.150	0.359	0.436	0.255	0.309	0.456	0.262	0.282
ATGC178	0.559	0.136	0.305	0.525	0.190	0.285	0.550	0.197	0.252
ATGC181	0.515	0.136	0.350	0.524	0.220	0.256	0.527	0.228	0.245
ATGC182	0.507	0.174	0.319	0.539	0.203	0.258	0.547	0.228	0.225
ATGC186	0.546	0.175	0.278	0.517	0.252	0.231	0.530	0.243	0.227
ATGC188	0.531	0.189	0.280	0.445	0.283	0.272	0.437	0.300	0.264
ATGC192	0.500	0.241	0.259	0.508	0.223	0.268	0.556	0.216	0.229
ATGC201	0.475	0.262	0.262	0.484	0.250	0.266	0.515	0.245	0.241
ATGC202	0.461	0.191	0.348	0.503	0.239	0.258	0.540	0.218	0.242
ATGC223	0.620	0.184	0.196	0.443	0.258	0.300	0.456	0.275	0.269
ATGC232	0.493	0.137	0.370	0.430	0.292	0.278	0.445	0.284	0.271
ATGC235	0.452	0.151	0.398	0.507	0.230	0.263	0.517	0.242	0.241
ATGC239	0.541	0.165	0.293	0.465	0.226	0.309	0.484	0.269	0.247

Species	P0	P1	P2	P0	P1	P2	P0	P1	P2
ATGC242	0.510	0.243	0.247	0.451	0.290	0.259	0.465	0.282	0.253
ATGC243	0.525	0.200	0.275	0.469	0.248	0.283	0.499	0.258	0.243
ATGC244	0.509	0.192	0.299	0.480	0.259	0.261	0.478	0.279	0.243
ATGC246	0.537	0.220	0.244	0.493	0.263	0.243	0.504	0.263	0.232
ATGC258	0.490	0.160	0.350	0.491	0.216	0.293	0.503	0.239	0.258
ATGC260	0.457	0.190	0.352	0.478	0.233	0.289	0.492	0.248	0.260
ATGC277	0.503	0.204	0.293	0.459	0.223	0.318	0.427	0.272	0.300
ATGC279	0.488	0.244	0.268	0.524	0.238	0.238	0.507	0.263	0.230
ATGC301	0.511	0.201	0.288	0.490	0.235	0.276	0.483	0.266	0.251
ATGC309	0.551	0.102	0.347	0.521	0.224	0.255	0.539	0.197	0.264
ATGC330	0.495	0.176	0.330	0.460	0.256	0.284	0.499	0.252	0.249
ATGC332	0.530	0.170	0.300	0.450	0.262	0.289	0.474	0.284	0.242
ATGC339	0.421	0.198	0.380	0.451	0.251	0.298	0.471	0.269	0.260
ATGC341	0.509	0.125	0.366	0.454	0.242	0.303	0.472	0.249	0.278
ATGC352	0.480	0.185	0.336	0.431	0.259	0.311	0.441	0.286	0.272
ATGC361	0.429	0.199	0.373	0.473	0.263	0.263	0.482	0.250	0.269
ATGC369	0.423	0.282	0.295	0.487	0.260	0.253	0.550	0.236	0.215
ATGC371	0.504	0.248	0.248	0.521	0.226	0.253	0.533	0.233	0.234
ATGC372	0.532	0.241	0.228	0.518	0.232	0.250	0.549	0.225	0.226
ATGC374	0.479	0.164	0.356	0.449	0.242	0.309	0.467	0.284	0.249
ATGC376	0.485	0.205	0.311	0.465	0.273	0.262	0.495	0.253	0.252
ATGC389	0.510	0.152	0.338	0.494	0.242	0.265	0.525	0.243	0.232
ATGC393	0.513	0.183	0.304	0.453	0.271	0.276	0.468	0.279	0.253
ATGC420	0.588	0.115	0.297	0.477	0.227	0.296	0.481	0.258	0.261
ATGC428	0.535	0.153	0.312	0.538	0.218	0.245	0.576	0.195	0.229
ATGC431	0.533	0.162	0.305	0.481	0.246	0.273	0.518	0.236	0.246

Table D.2: **Phase Proportions of 90 Species across Three Alignment Methods** | P0–Phase0, P1–Phase1, P2-Phase2.

dN/dS and Zn/(Zn+Zs) Ratio of 90 Species across Three Alignment Methods

	mafft+sw		**coati-max**		**coati-sampling**	
Species	dN/dS	Zn/Zn+Zs	dN/dS	Zn/Zn+Zs	dN/dS	Zn/Zn+Zs
01FcaCaf	0.236	0.247	0.129	0.110	0.120	0.107
02MusRat	0.216	0.251	0.121	0.106	0.113	0.103
04Gallus	0.288	0.266	0.110	0.104	0.106	0.101
05Droso	0.173	0.190	0.103	0.069	0.101	0.066
06Nematode	0.155	0.212	0.095	0.177	0.061	0.174
07yeast	0.131	0.248	0.092	0.131	0.079	0.131
08Arab	0.275	0.267	0.181	0.102	0.174	0.099
09SarMon	0.223	0.260	0.123	0.136	0.111	0.133
10ants	0.155	0.255	0.109	0.158	0.090	0.155
11malaria	0.240	0.250	0.191	0.180	0.158	0.168
12fugu	0.212	0.254	0.128	0.142	0.114	0.139
13Phytophthora	0.131	0.249	0.096	0.156	0.079	0.152
14Fusarium	0.229	0.244	0.097	0.087	0.093	0.085
15Oryza	0.545	0.181	0.293	0.038	0.292	0.036
16Solanum	0.301	0.275	0.199	0.090	0.194	0.088
ATGC001	0.123	0.283	0.074	0.168	0.050	0.156
ATGC013	0.218	0.282	0.119	0.169	0.082	0.177
ATGC015	0.188	0.255	0.114	0.204	0.064	0.205
ATGC016	0.161	0.219	0.092	0.143	0.069	0.142
ATGC021	0.174	0.294	0.098	0.190	0.065	0.182
ATGC022	0.212	0.262	0.105	0.153	0.075	0.171
ATGC028	0.243	0.232	0.170	0.189	0.142	0.179
ATGC044	0.204	0.379	0.163	0.150	0.145	0.169
ATGC045	0.263	0.250	0.187	0.222	0.141	0.217
ATGC050	0.171	0.247	0.110	0.204	0.080	0.213
ATGC056	0.113	0.184	0.074	0.126	0.031	0.135
ATGC058	0.131	0.260	0.084	0.173	0.056	0.168
ATGC069	0.174	0.228	0.111	0.177	0.078	0.168
ATGC071	0.205	0.197	0.132	0.187	0.111	0.180
ATGC073	0.271	0.208	0.171	0.198	0.140	0.202

Species	dN/dS	Zn/Zn+Zs	dN/dS	Zn/Zn+Zs	dN/dS	Zn/Zn+Zs
ATGC075	0.176	0.230	0.107	0.170	0.082	0.160
ATGC078	0.180	0.216	0.115	0.180	0.093	0.169
ATGC095	0.175	0.174	0.111	0.160	0.075	0.160
ATGC099	0.161	0.186	0.114	0.147	0.078	0.152
ATGC110	0.122	0.189	0.072	0.188	0.042	0.185
ATGC111	0.224	0.203	0.128	0.155	0.107	0.160
ATGC112	0.161	0.239	0.088	0.176	0.060	0.159
ATGC117	0.144	0.200	0.084	0.201	0.055	0.196
ATGC120	0.150	0.251	0.076	0.215	0.045	0.208
ATGC127	0.136	0.271	0.081	0.181	0.046	0.177
ATGC134	0.210	0.197	0.120	0.155	0.101	0.153
ATGC136	0.139	0.188	0.090	0.135	0.067	0.143
ATGC138	0.162	0.226	0.108	0.220	0.071	0.199
ATGC143	0.199	0.284	0.133	0.187	0.103	0.194
ATGC145	0.241	0.282	0.178	0.224	0.149	0.217
ATGC146	0.155	0.280	0.105	0.185	0.065	0.174
ATGC152	0.179	0.234	0.111	0.185	0.079	0.176
ATGC155	0.306	0.279	0.237	0.164	0.201	0.165
ATGC159	0.141	0.304	0.097	0.208	0.070	0.202
ATGC169	0.265	0.231	0.193	0.213	0.164	0.199
ATGC171	0.161	0.208	0.109	0.171	0.084	0.168
ATGC173	0.265	0.229	0.192	0.162	0.164	0.161
ATGC178	0.201	0.271	0.127	0.184	0.105	0.169
ATGC181	0.203	0.184	0.127	0.200	0.081	0.195
ATGC182	0.143	0.246	0.092	0.155	0.057	0.168
ATGC186	0.186	0.258	0.116	0.207	0.081	0.206
ATGC188	0.206	0.208	0.145	0.174	0.124	0.165
ATGC192	0.167	0.259	0.121	0.192	0.073	0.184
ATGC201	0.227	0.344	0.184	0.201	0.135	0.204
ATGC202	0.196	0.247	0.128	0.181	0.095	0.172
ATGC223	0.267	0.264	0.227	0.229	0.180	0.229
ATGC232	0.154	0.160	0.107	0.170	0.083	0.171
ATGC235	0.158	0.258	0.096	0.200	0.065	0.191
ATGC239	0.157	0.203	0.090	0.183	0.067	0.181
ATGC242	0.167	0.303	0.114	0.207	0.081	0.195
ATGC243	0.189	0.225	0.106	0.209	0.063	0.205
ATGC244	0.152	0.228	0.087	0.212	0.054	0.204
ATGC246	0.286	0.244	0.211	0.202	0.173	0.196
ATGC258	0.149	0.230	0.095	0.172	0.064	0.185
ATGC260	0.264	0.295	0.160	0.183	0.131	0.189
ATGC277	0.266	0.261	0.161	0.206	0.138	0.196
ATGC279	0.172	0.293	0.094	0.201	0.023	0.213
ATGC301	0.230	0.245	0.137	0.187	0.108	0.183
ATGC309	0.175	0.184	0.110	0.164	0.085	0.157
ATGC330	0.165	0.242	0.100	0.194	0.074	0.175
ATGC332	0.328	0.244	0.255	0.190	0.224	0.184
ATGC339	0.160	0.231	0.113	0.171	0.082	0.165
ATGC341	0.216	0.223	0.129	0.199	0.101	0.202
ATGC352	0.334	0.232	0.230	0.173	0.211	0.170
ATGC361	0.178	0.248	0.114	0.166	0.092	0.166
ATGC369	0.178	0.372	0.099	0.178	0.073	0.169
ATGC371	0.150	0.285	0.083	0.186	0.040	0.191
ATGC372	0.158	0.291	0.115	0.176	0.063	0.181
ATGC374	0.162	0.202	0.104	0.165	0.075	0.174
ATGC376	0.228	0.242	0.131	0.180	0.098	0.180
ATGC389	0.211	0.216	0.117	0.202	0.074	0.195
ATGC393	0.142	0.190	0.094	0.169	0.074	0.168
ATGC420	0.277	0.176	0.176	0.197	0.144	0.192
ATGC428	0.144	0.229	0.096	0.174	0.053	0.171
ATGC431	0.156	0.190	0.079	0.206	0.044	0.202

Table D.3: dN/dS and Zn/(Zn+Zs) Ratio of 90 Species across Three Alignment Methods.

ML Estimates of omega and Normalized Indel Rates r

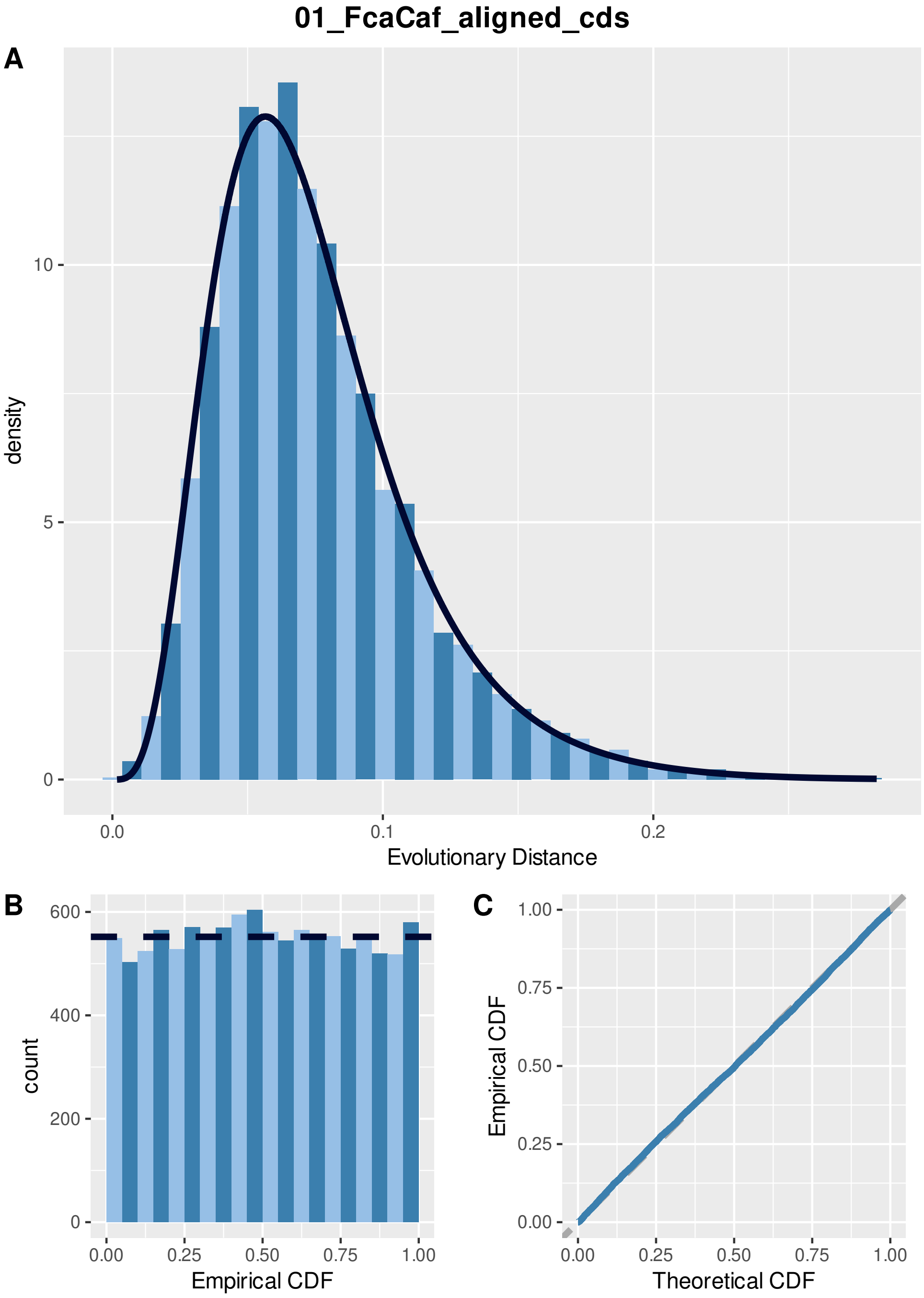
Species	ω	r0.norm	r1.norm	r2.norm

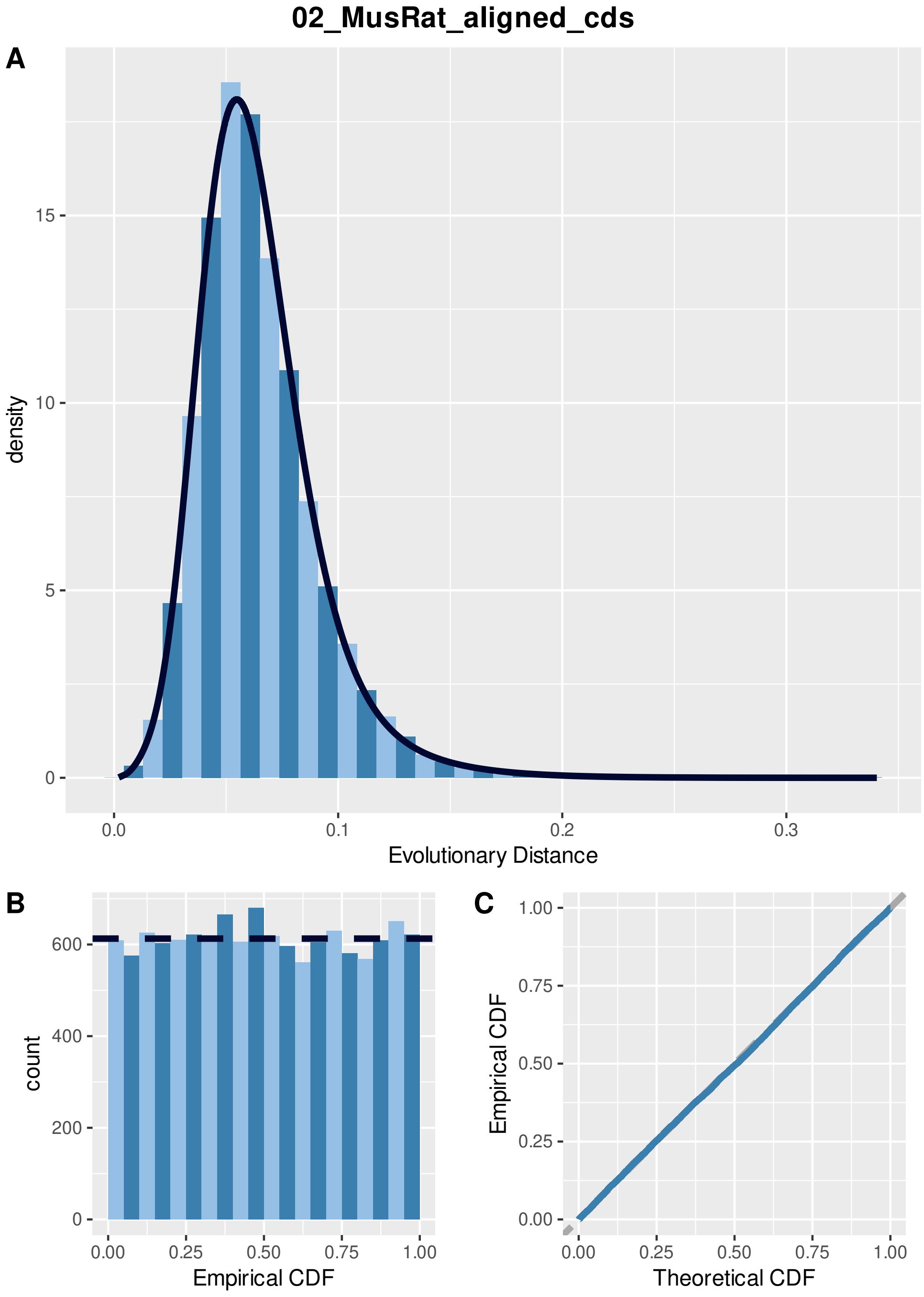
Species	ω	r0.norm	r1.norm	r2.norm
01FcaCaf	0.153	0.560	0.220	0.220
02MusRat	0.146	0.520	0.240	0.240
04Gallus	0.149	0.680	0.189	0.132
05Droso	0.116	0.539	0.154	0.308
06Nematode	0.058	0.454	0.269	0.277
07yeast	0.115	0.471	0.235	0.294
08Arab	0.203	0.471	0.255	0.274
09SarMon	0.142	0.551	0.232	0.217
10ants	0.177	0.430	0.262	0.308
11malaria	0.240	0.364	0.318	0.318
12fugu	0.133	0.525	0.243	0.233
13Phytophthora	0.114	0.480	0.240	0.280
14Fusarium	0.129	0.565	0.217	0.217
15Oryza	0.365	0.769	0.115	0.115
16Solanum	0.237	0.544	0.193	0.263
ATGC001	0.073	0.539	0.192	0.269
ATGC013	0.111	0.500	0.187	0.312
ATGC015	0.066	0.484	0.258	0.258
ATGC016	0.122	0.522	0.174	0.304
ATGC021	0.121	0.471	0.206	0.324
ATGC022	0.131	0.467	0.233	0.300
ATGC028	0.119	0.477	0.231	0.292
ATGC044	0.139	0.533	0.200	0.267
ATGC045	0.262	0.435	0.247	0.318
ATGC050	0.167	0.522	0.217	0.261
ATGC056	0.088	0.526	0.158	0.316
ATGC058	0.092	0.567	0.200	0.233
ATGC069	0.096	0.523	0.200	0.277
ATGC071	0.107	0.500	0.210	0.289
ATGC073	0.135	0.457	0.228	0.314
ATGC075	0.100	0.514	0.189	0.297
ATGC078	0.103	0.514	0.200	0.286
ATGC095	0.182	0.586	0.171	0.244
ATGC099	0.113	0.540	0.180	0.280
ATGC110	0.070	0.516	0.226	0.258
ATGC111	0.105	0.519	0.185	0.296
ATGC112	0.051	0.519	0.222	0.259
ATGC117	0.093	0.549	0.193	0.258
ATGC120	0.048	0.429	0.257	0.314
ATGC127	0.091	0.500	0.217	0.283
ATGC134	0.106	0.488	0.232	0.279
ATGC136	0.084	0.488	0.209	0.302
ATGC138	0.057	0.565	0.261	0.174
ATGC143	0.120	0.500	0.190	0.309
ATGC145	0.232	0.478	0.174	0.348
ATGC146	0.074	0.500	0.222	0.278
ATGC152	0.116	0.500	0.232	0.268
ATGC155	0.142	0.452	0.193	0.355
ATGC159	0.107	0.500	0.200	0.300
ATGC169	0.143	0.475	0.200	0.325
ATGC171	0.123	0.500	0.182	0.318
ATGC173	0.137	0.445	0.241	0.315
ATGC178	0.108	0.588	0.118	0.294
ATGC181	0.067	0.469	0.250	0.281
ATGC182	0.060	0.474	0.263	0.263
ATGC186	0.072	0.482	0.259	0.259
ATGC188	0.110	0.455	0.254	0.291
ATGC192	0.058	0.455	0.227	0.318
ATGC201	0.277	0.480	0.240	0.280
ATGC202	0.115	0.542	0.146	0.312
ATGC223	0.123	0.398	0.288	0.314
ATGC232	0.080	0.485	0.242	0.273
ATGC235	0.064	0.458	0.250	0.292
ATGC239	0.073	0.485	0.242	0.273
ATGC242	0.088	0.462	0.250	0.288
ATGC243	0.060	0.475	0.225	0.300
ATGC244	0.062	0.436	0.291	0.273
ATGC246	0.173	0.488	0.244	0.268
ATGC258	0.066	0.533	0.200	0.267
ATGC260	0.108	0.478	0.217	0.304
ATGC277	0.120	0.421	0.237	0.342
ATGC279	0.047	0.572	0.190	0.238

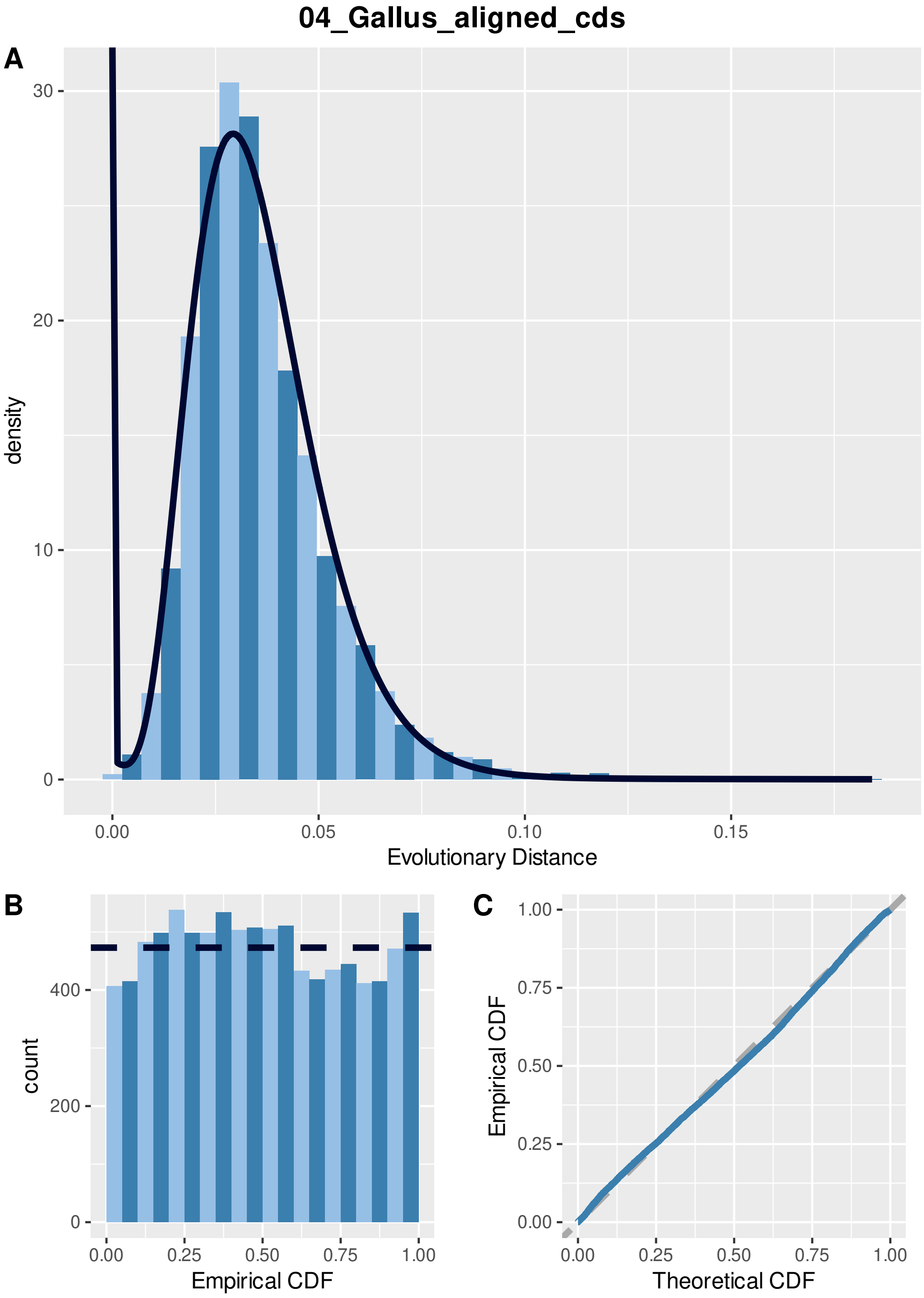
Species	ω	r0.norm	r1.norm	r2.norm
ATGC301	0.098	0.488	0.219	0.293
ATGC309	0.072	0.478	0.217	0.304
ATGC330	0.065	0.462	0.256	0.282
ATGC332	0.156	0.514	0.222	0.264
ATGC339	0.074	0.488	0.219	0.293
ATGC341	0.088	0.476	0.222	0.302
ATGC352	0.184	0.462	0.250	0.288
ATGC361	0.103	0.482	0.222	0.296
ATGC369	0.074	0.522	0.217	0.261
ATGC371	0.049	0.482	0.296	0.222
ATGC372	0.073	0.611	0.222	0.167
ATGC374	0.054	0.458	0.250	0.292
ATGC376	0.081	0.526	0.237	0.237
ATGC389	0.055	0.485	0.242	0.273
ATGC393	0.080	0.470	0.245	0.286
ATGC420	0.111	0.478	0.217	0.304
ATGC428	0.056	0.500	0.222	0.278
ATGC431	0.043	0.522	0.217	0.261

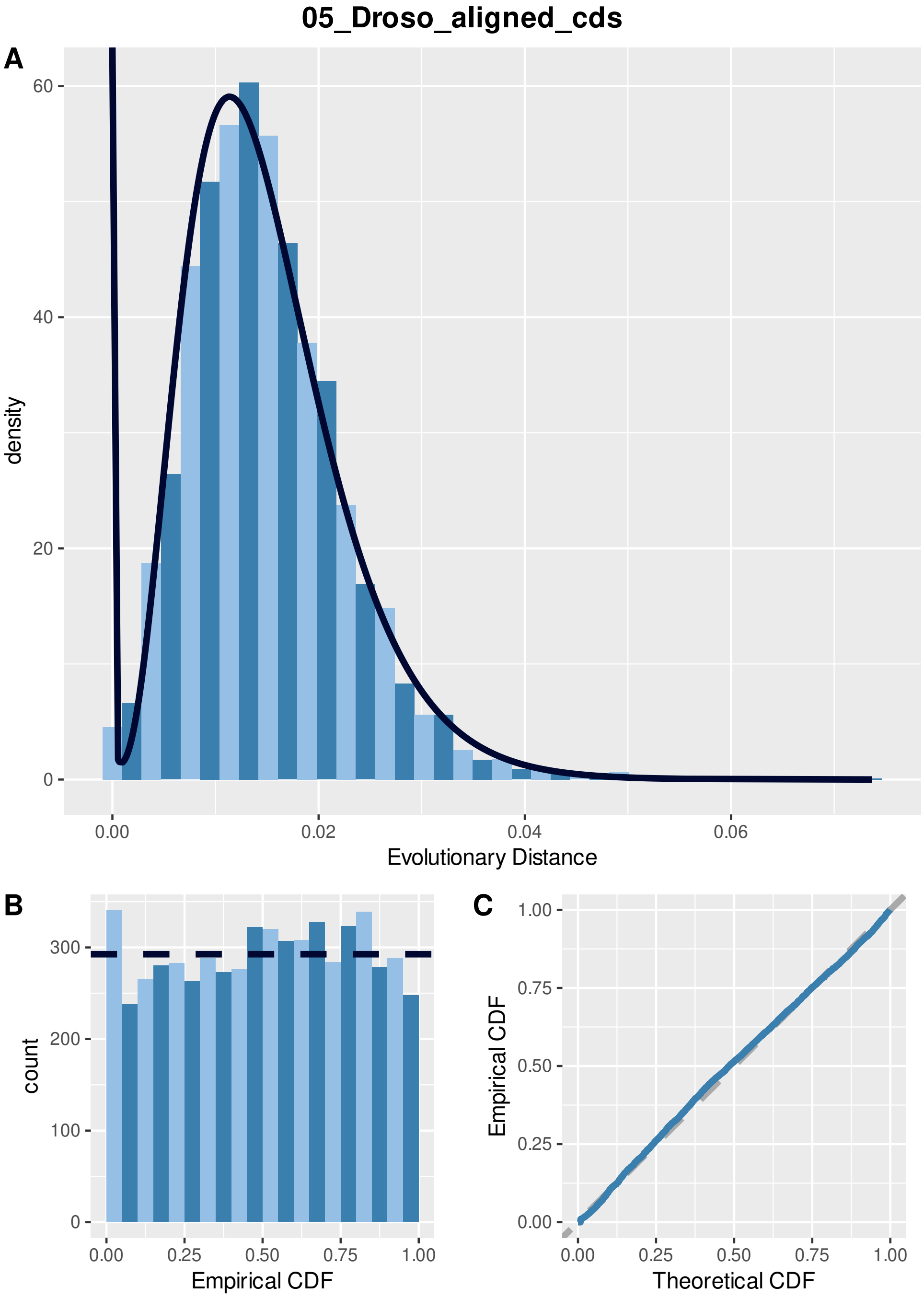
Table D.4: **MLE Estimates of ω and Normalized Indel Rates r.** | r0.norm, r1.norm, and r2.norm are normalized rates of phase0, phase1, and phase2 indel events.

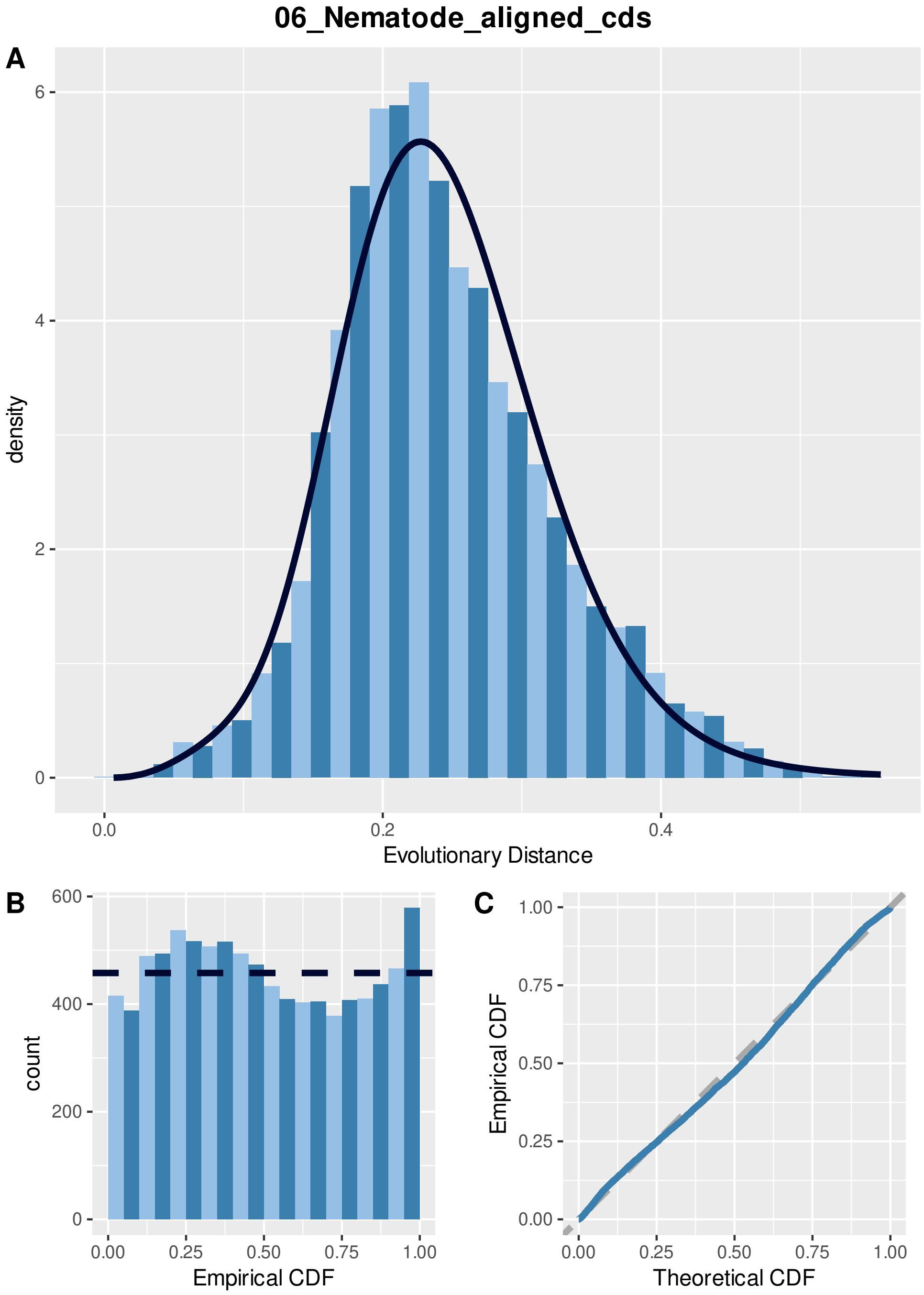
Post-filtering Distance Distribution Plot of 90 Species

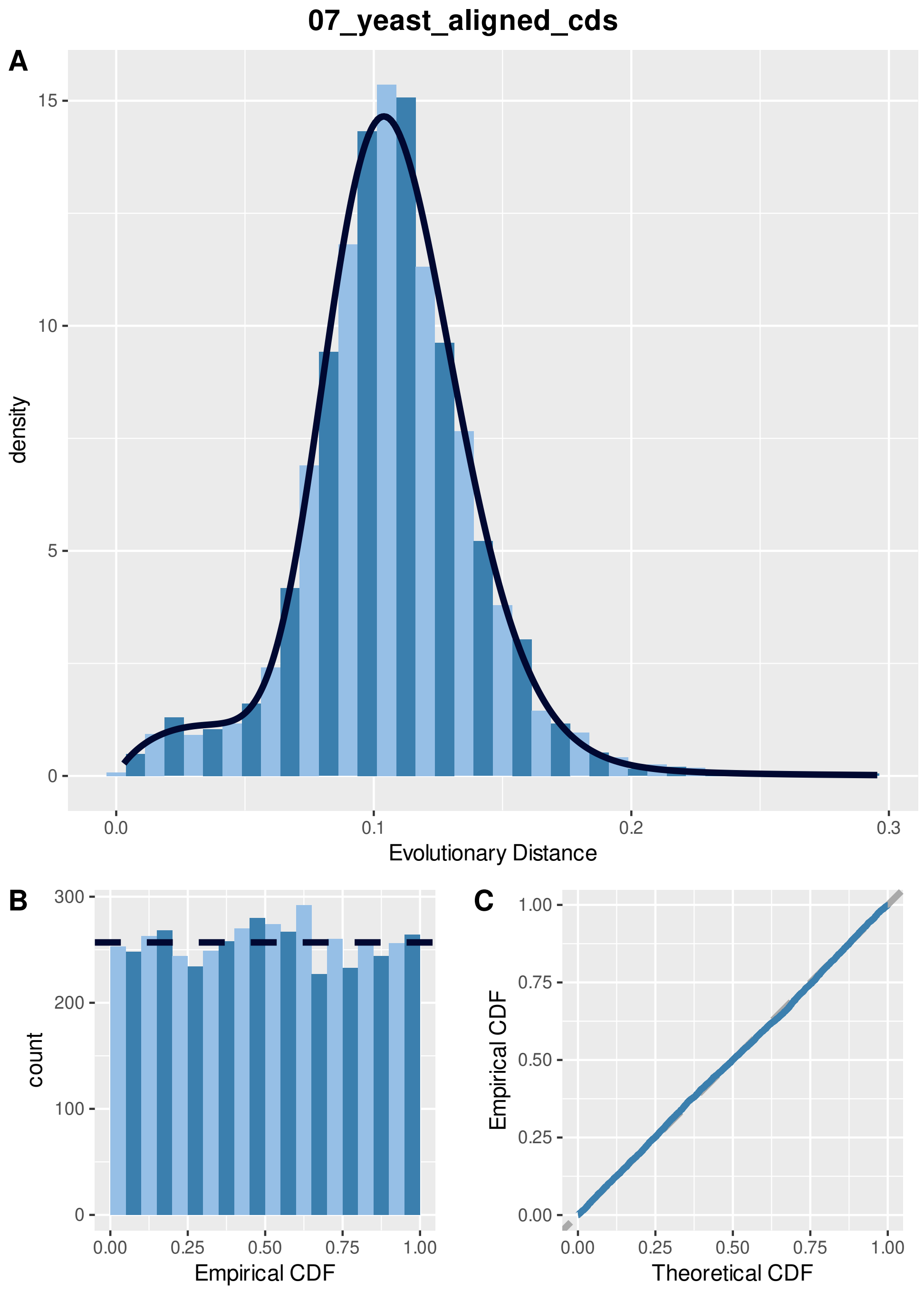

01_FcaCaf_aligned_cds
A
density
10
5
0
0.0
0.1
0.2
Evolutionary Distance
B
count
600
400
200
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

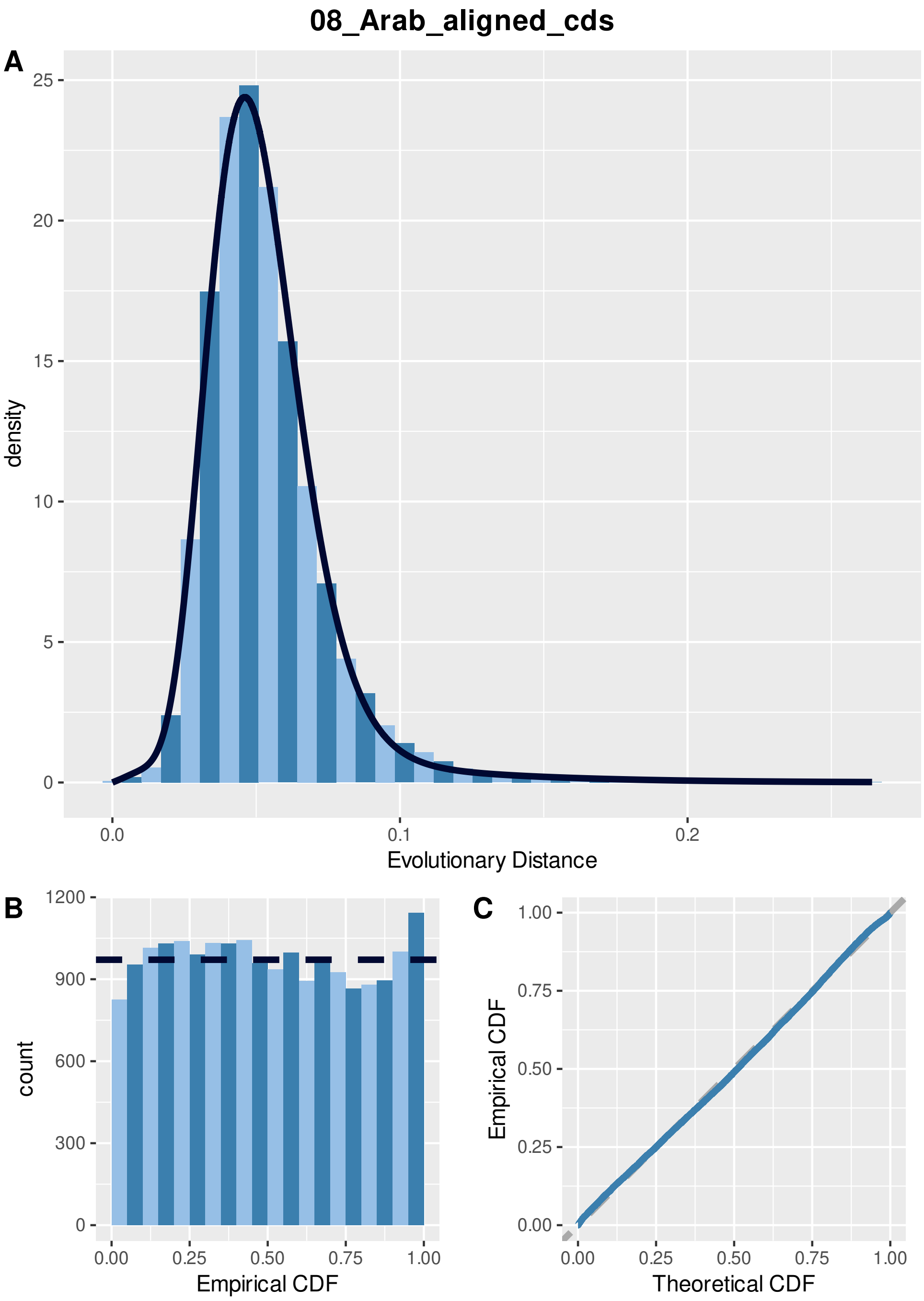

02_MusRat_aligned_cds
A
density
15
10
5
0
0.0
0.1
0.2
0.3
Evolutionary Distance
B
count
600
400
200
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

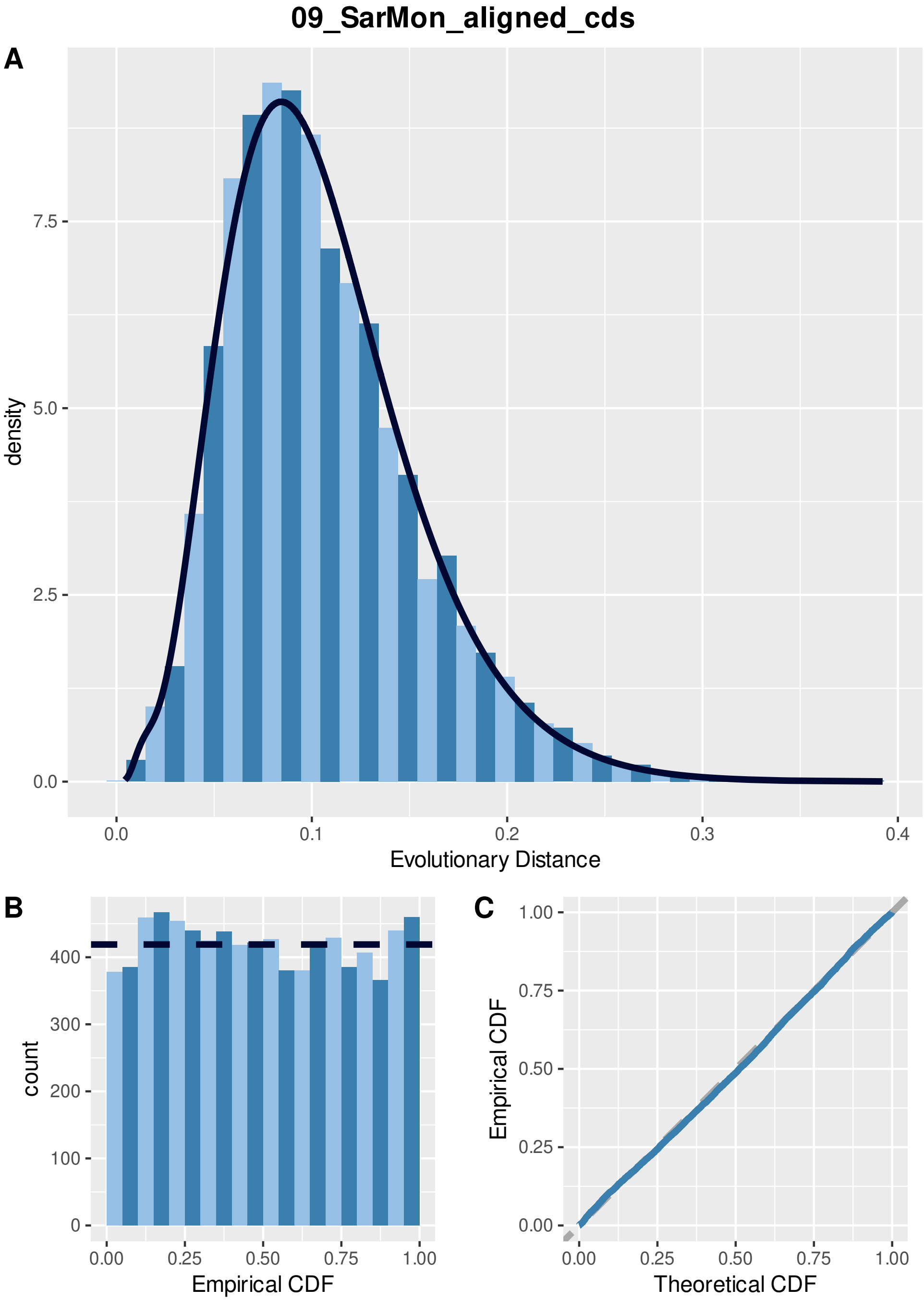

04_Gallus_aligned_cds
A
density
30
20
10
0
0.00
0.05
0.10
0.15
Evolutionary Distance
B
count
400
200
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

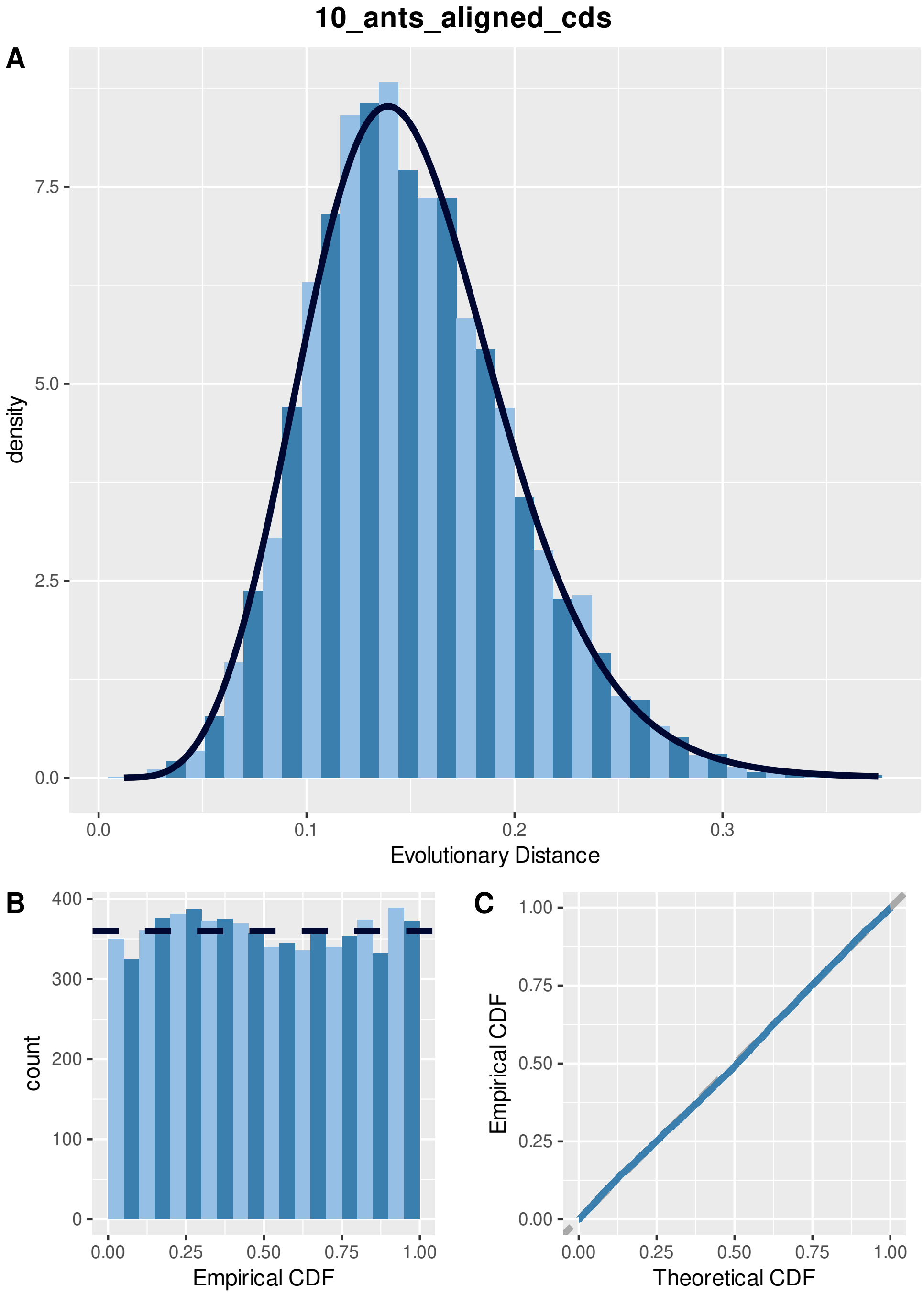

05_Droso_aligned_cds
A
density
60
40
20
0
0.00
0.02
0.04
0.06
Evolutionary Distance
B
count
300
200
100
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

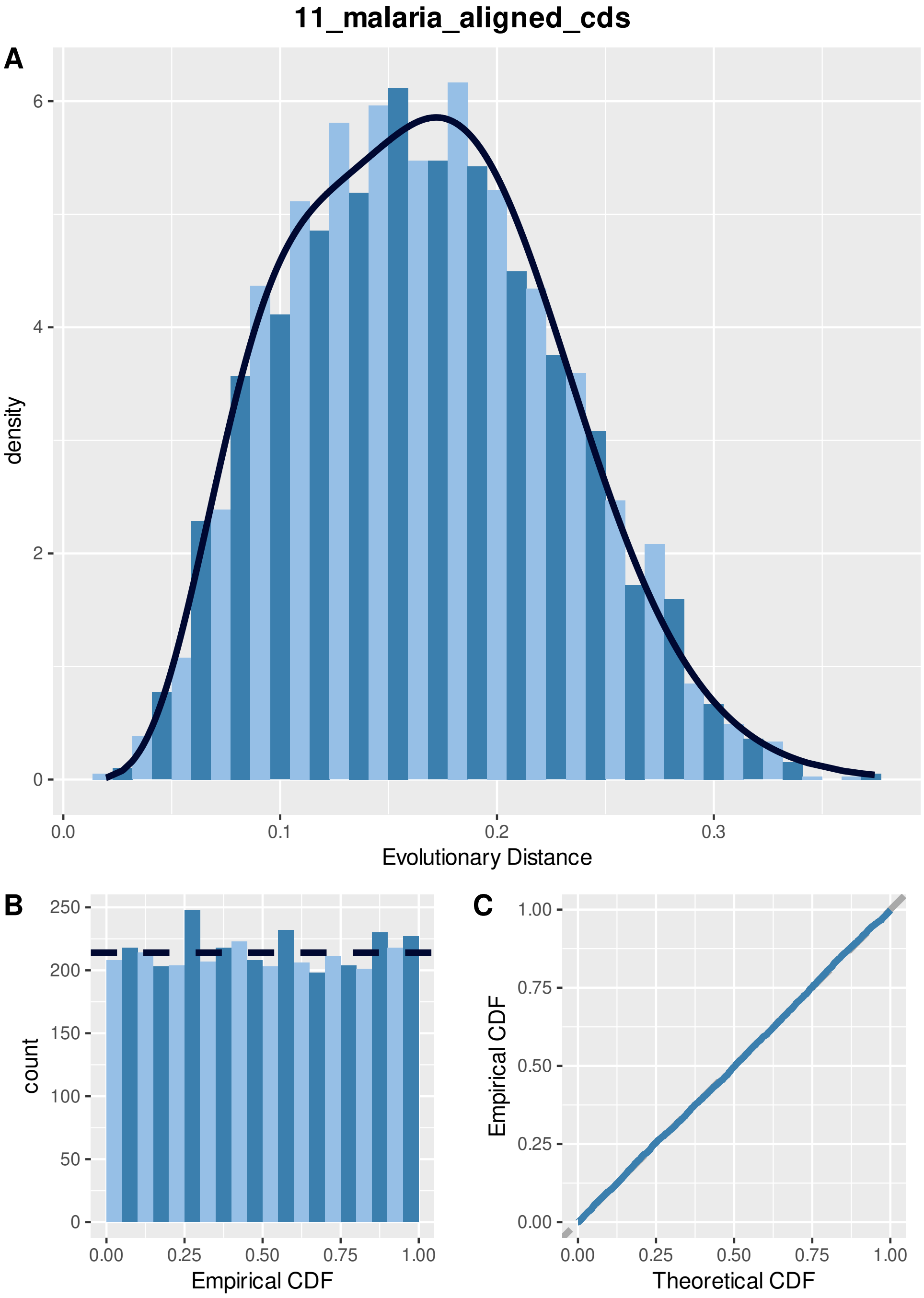

06_Nematode_aligned_cds
A
density
6
4
2
0
0.0
0.2
0.4
Evolutionary Distance
B
count
600
400
200
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

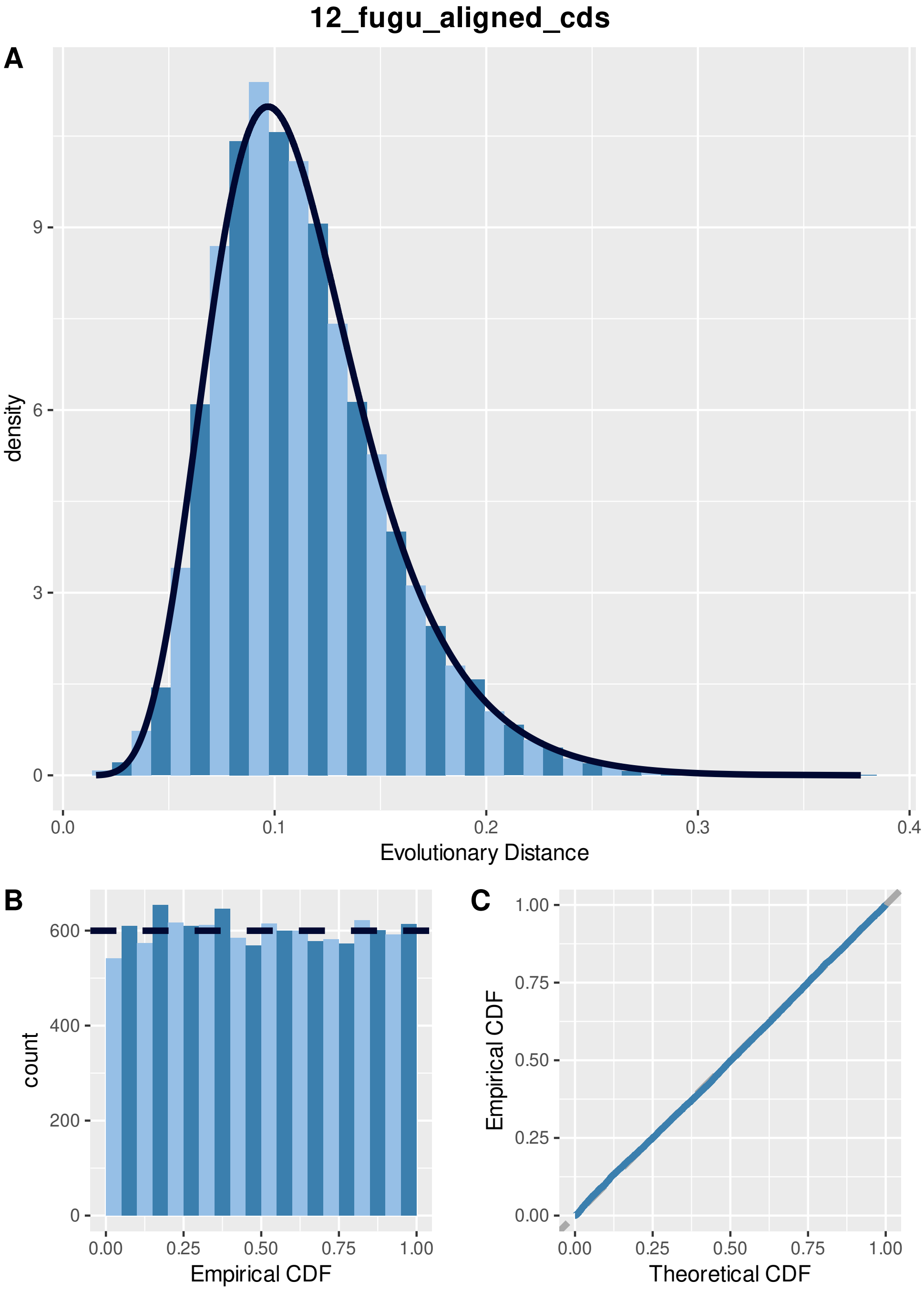

07_yeast_aligned_cds
A
density
15
10
5
0
0.0
0.1
0.2
0.3
Evolutionary Distance
B
count
300
200
100
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

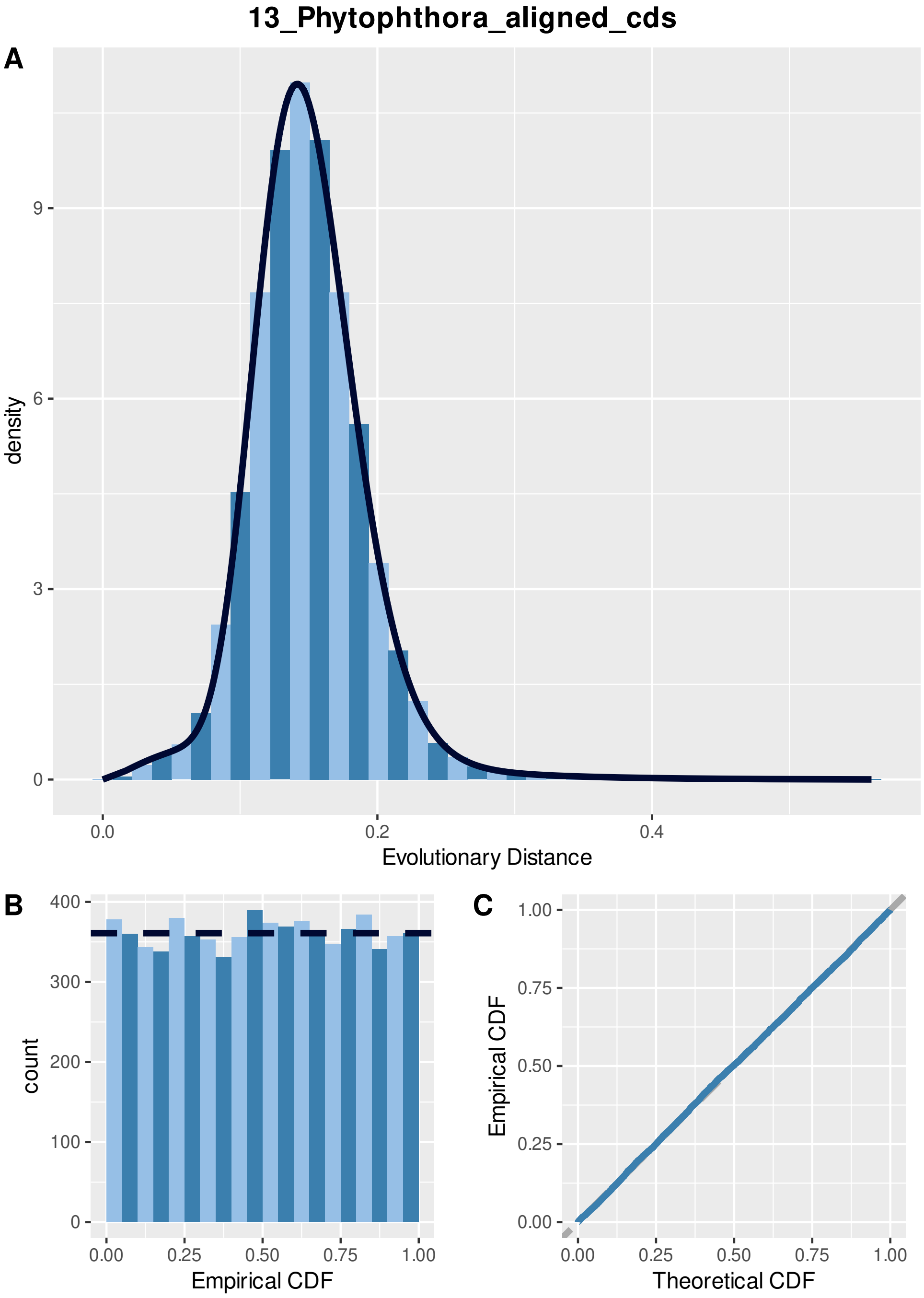

08_Arab_aligned_cds
A
25
20
15
10
5
0
density
0.0
0.1
0.2
Evolutionary Distance
B
1200
900
600
300
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

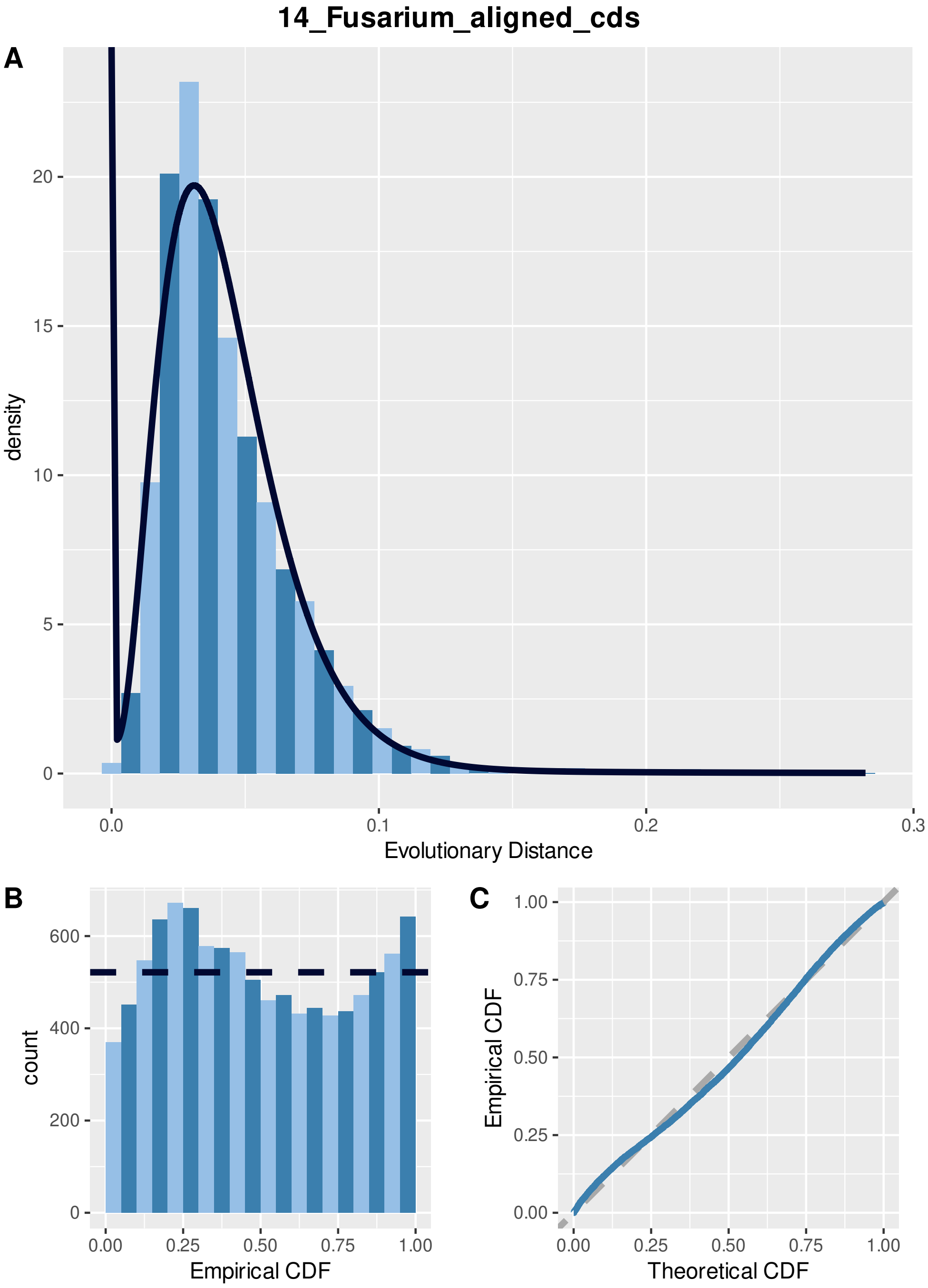

09_SarMon_aligned_cds
A
density
7.5
5.0
2.5
0.0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
400
300
200
100
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

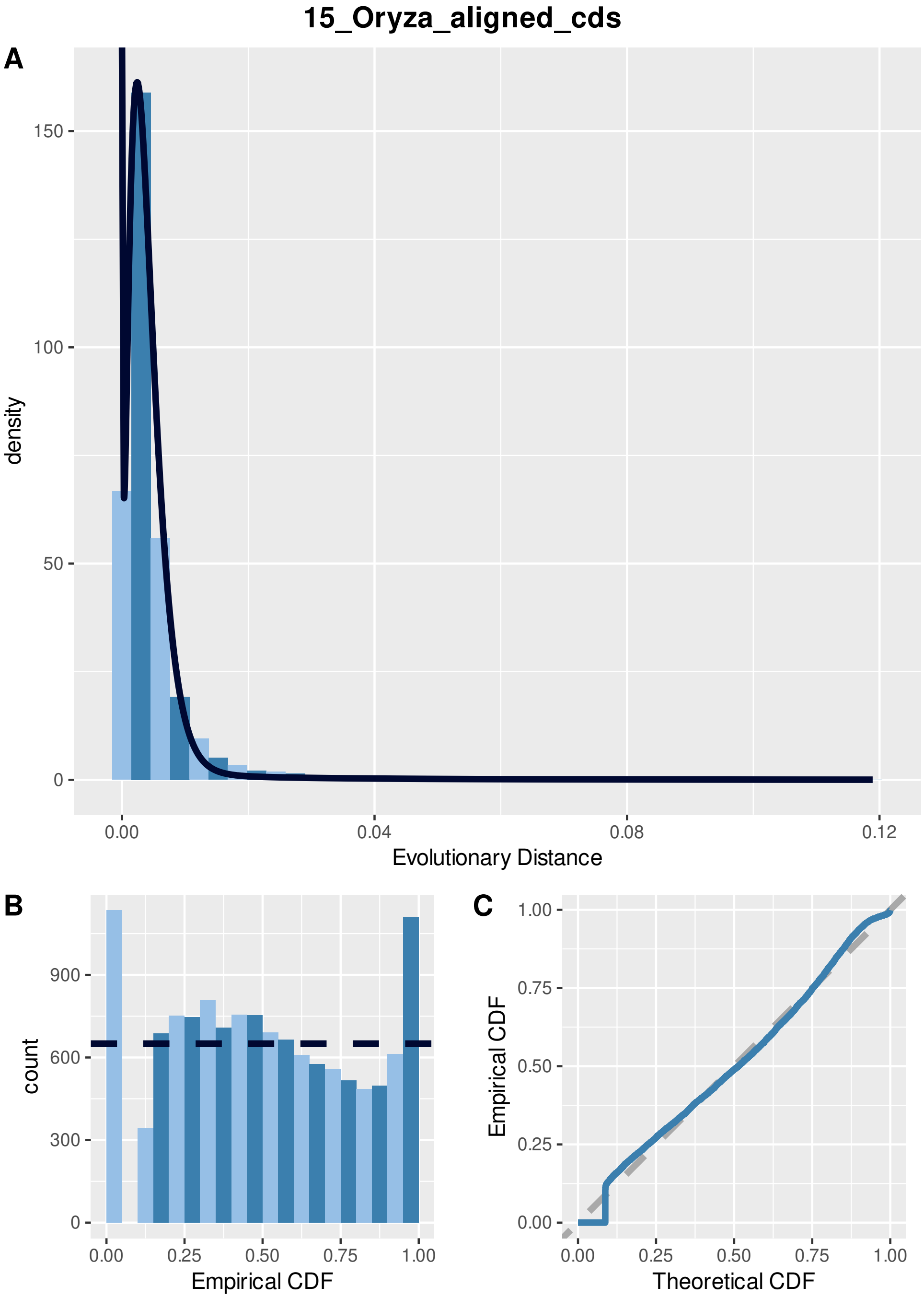

10_ants_aligned_cds
A
density
7.5
5.0
2.5
0.0
0.0
0.1
0.2
0.3
Evolutionary Distance
B
count
400
300
200
100
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

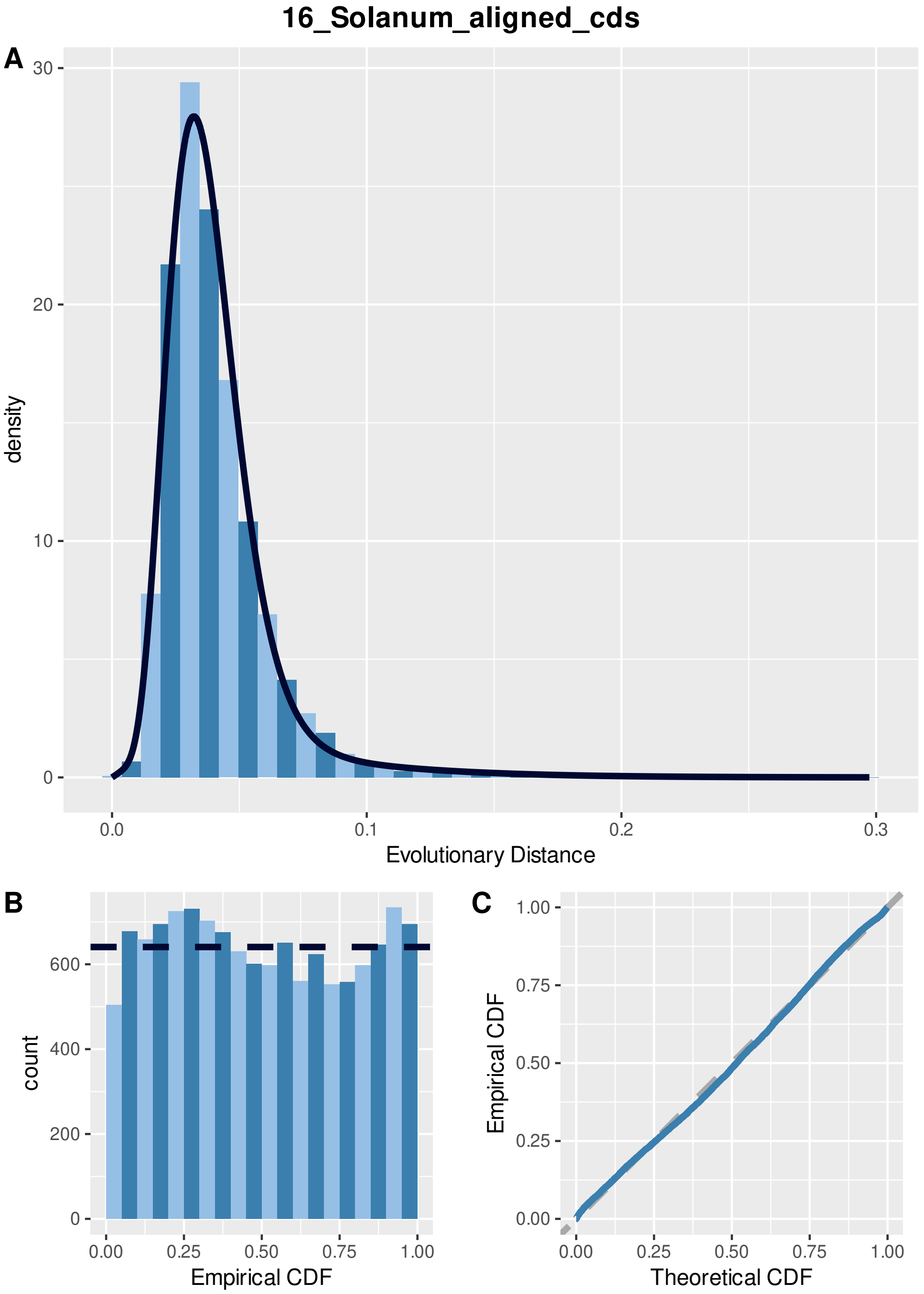

11_malaria_aligned_cds
A
density
6
4
2
0
0.0
0.1
0.2
0.3
Evolutionary Distance
B
count
250
200
150
100
50
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

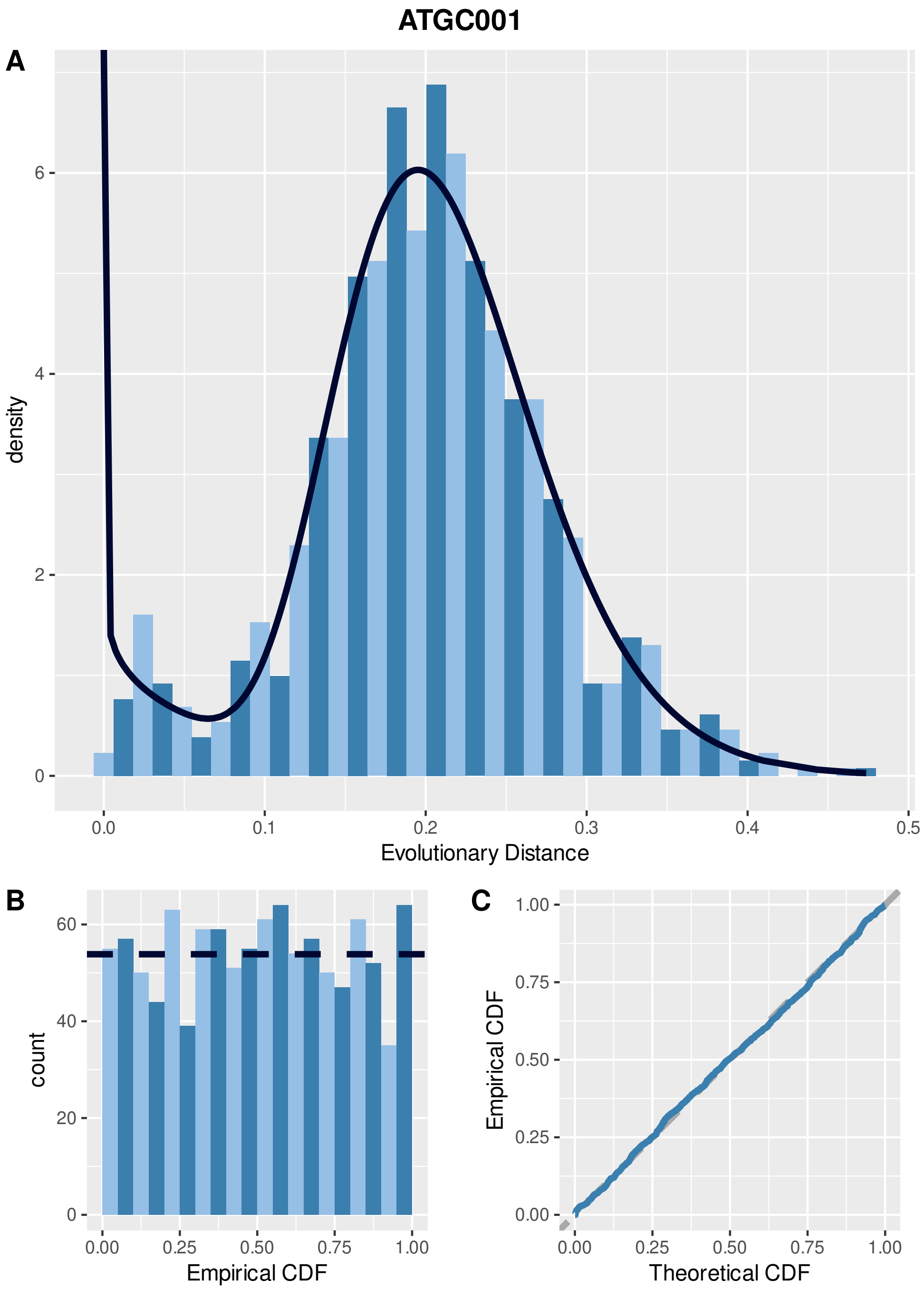

12_fugu_aligned_cds
A
density
0
3
6
9
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
0
200
400
600
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

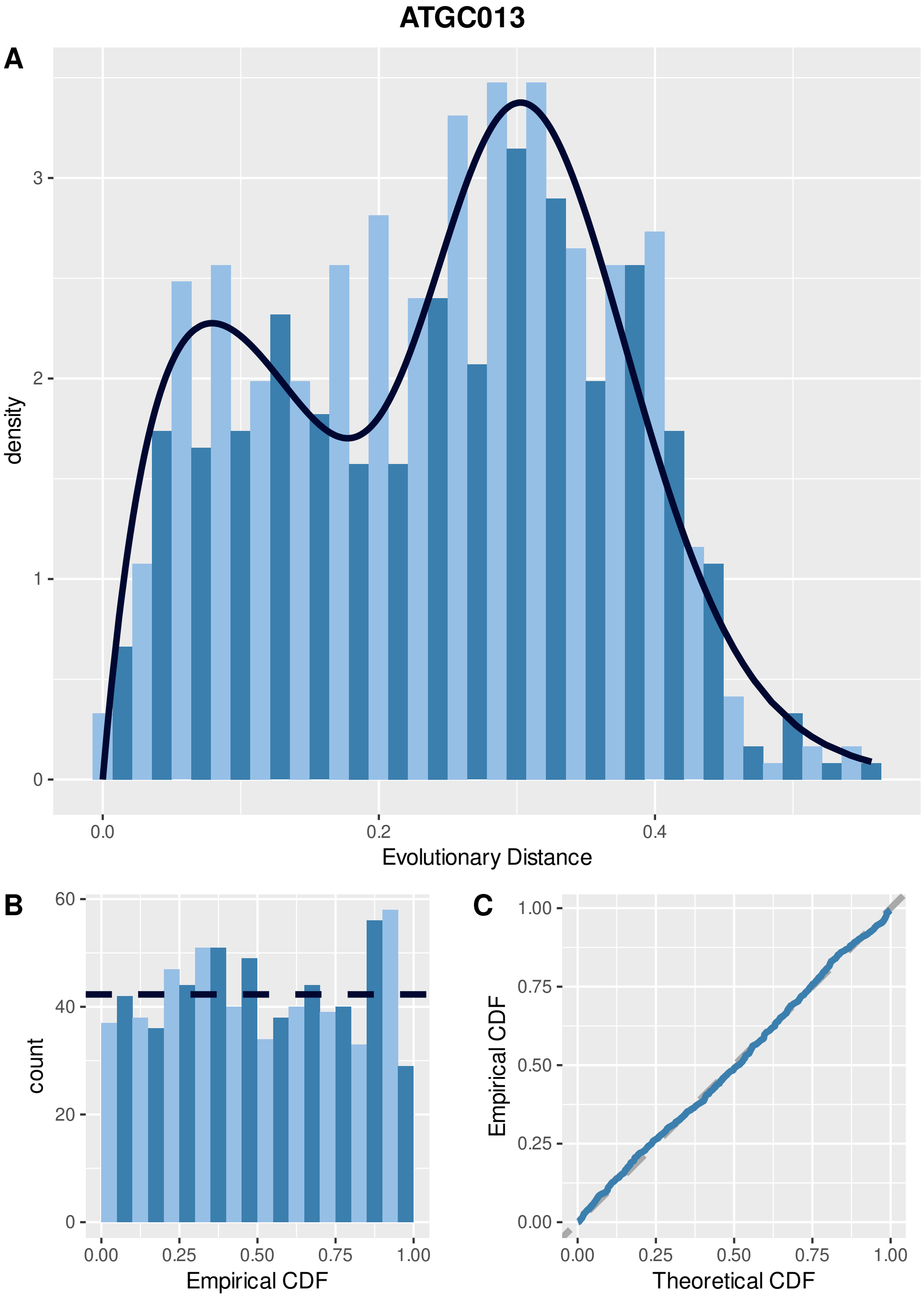

13_Phytophthora_aligned_cds
A
density
0
3
6
9
0.0
0.2
0.4
Evolutionary Distance
B
count
0
100
200
300
400
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

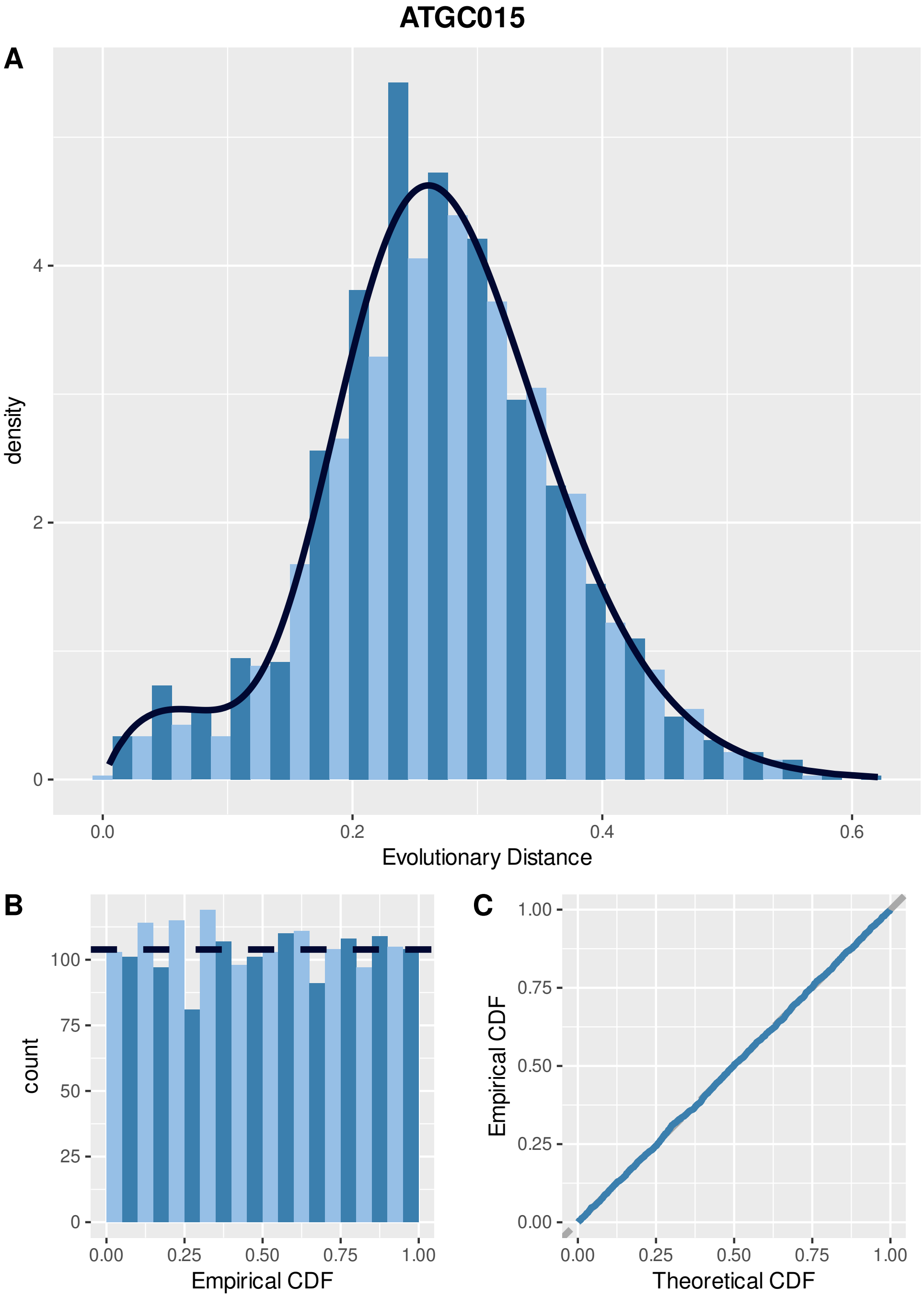

14_Fusarium_aligned_cds
A
density
0
5
10
15
20
0.0
0.1
0.2
0.3
Evolutionary Distance
B
count
0
200
400
600
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

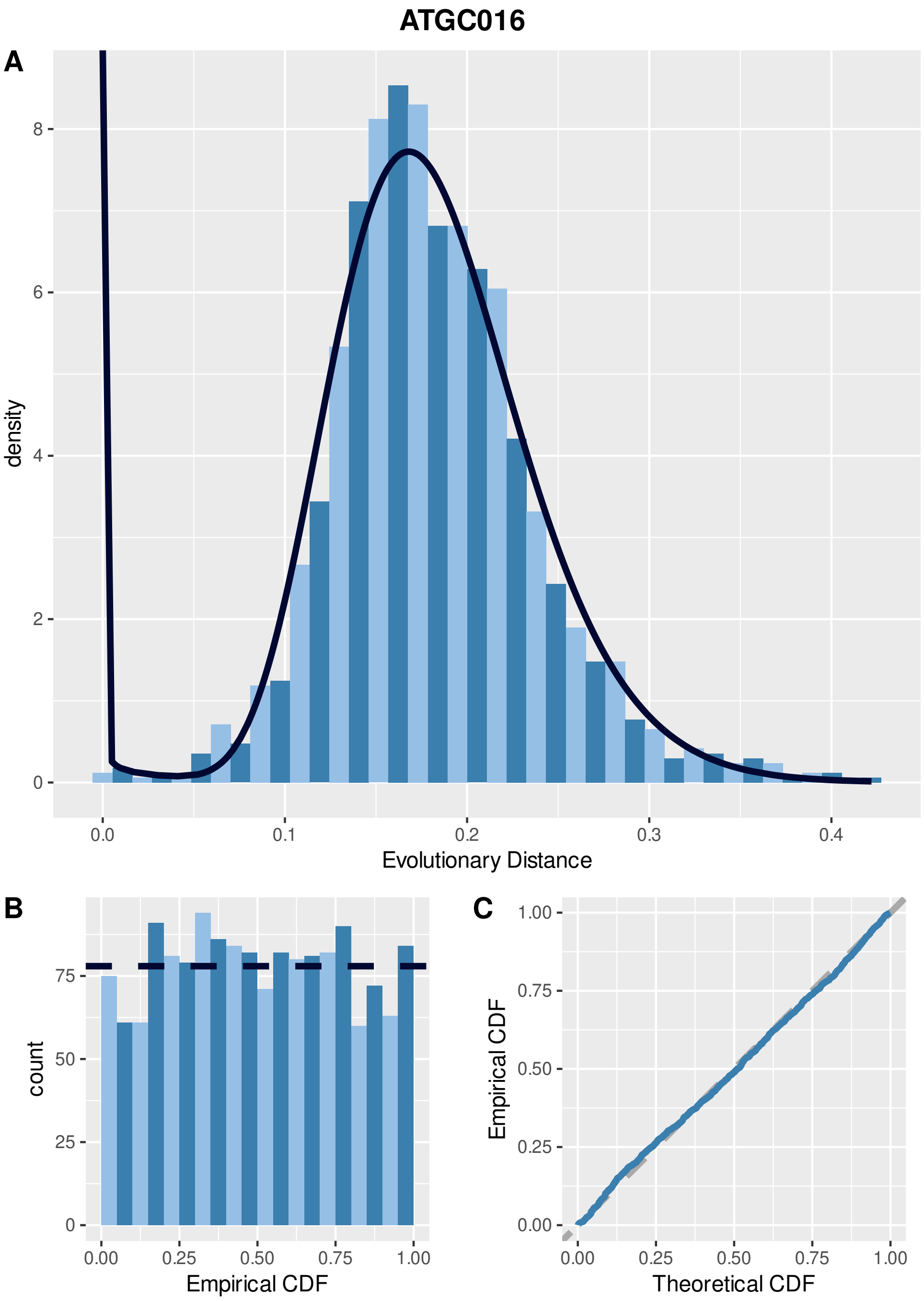

15_Oryza_aligned_cds
A
density
150
100
50
0
0.00
0.04
0.08
0.12
Evolutionary Distance
B
count
900
600
300
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

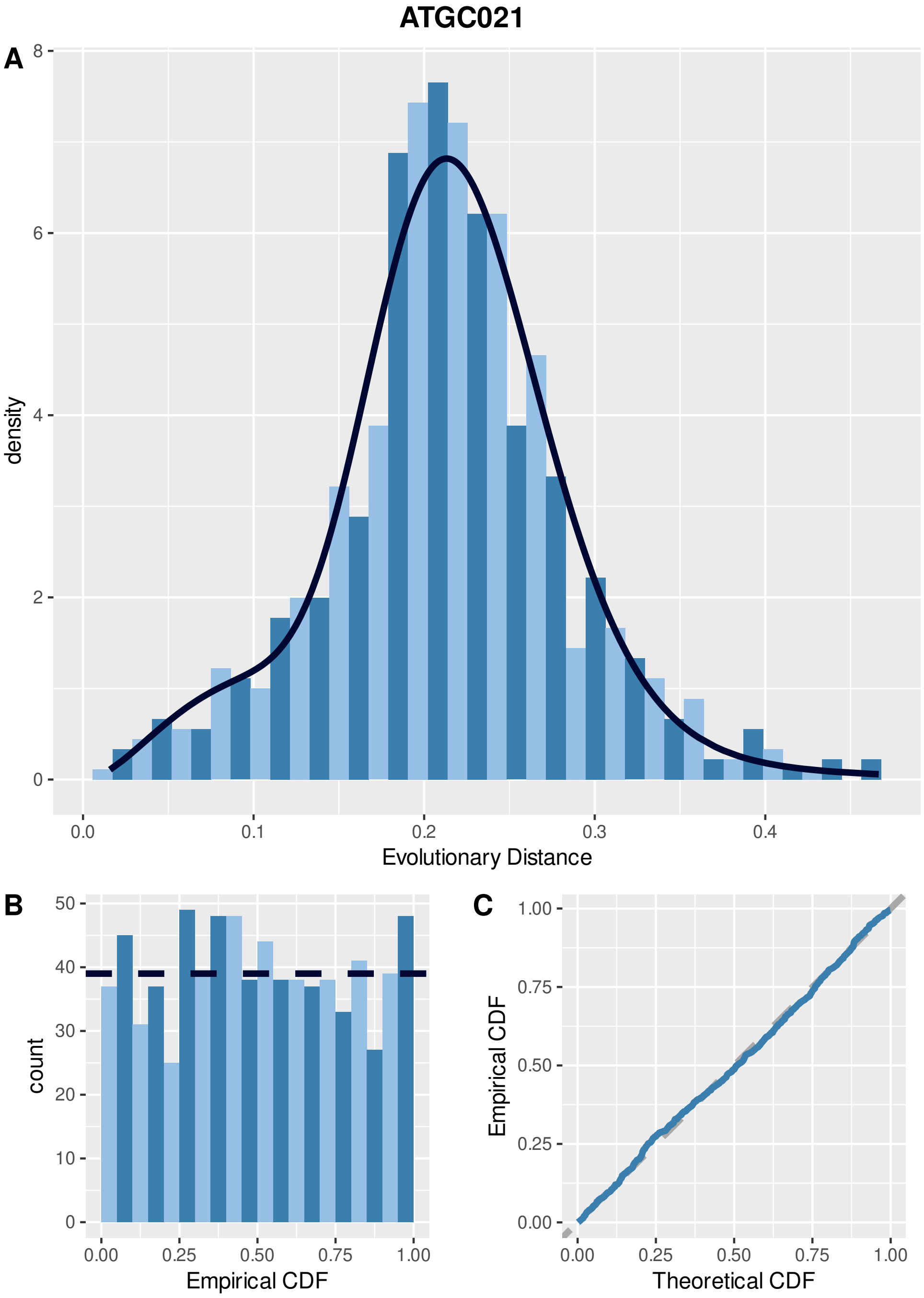

16_Solanum_aligned_cds
A
30
20
10
0
density
0.0
0.1
0.2
0.3
Evolutionary Distance
B
600
400
200
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC001
A
density
0
2
4
6
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
0
20
40
60
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

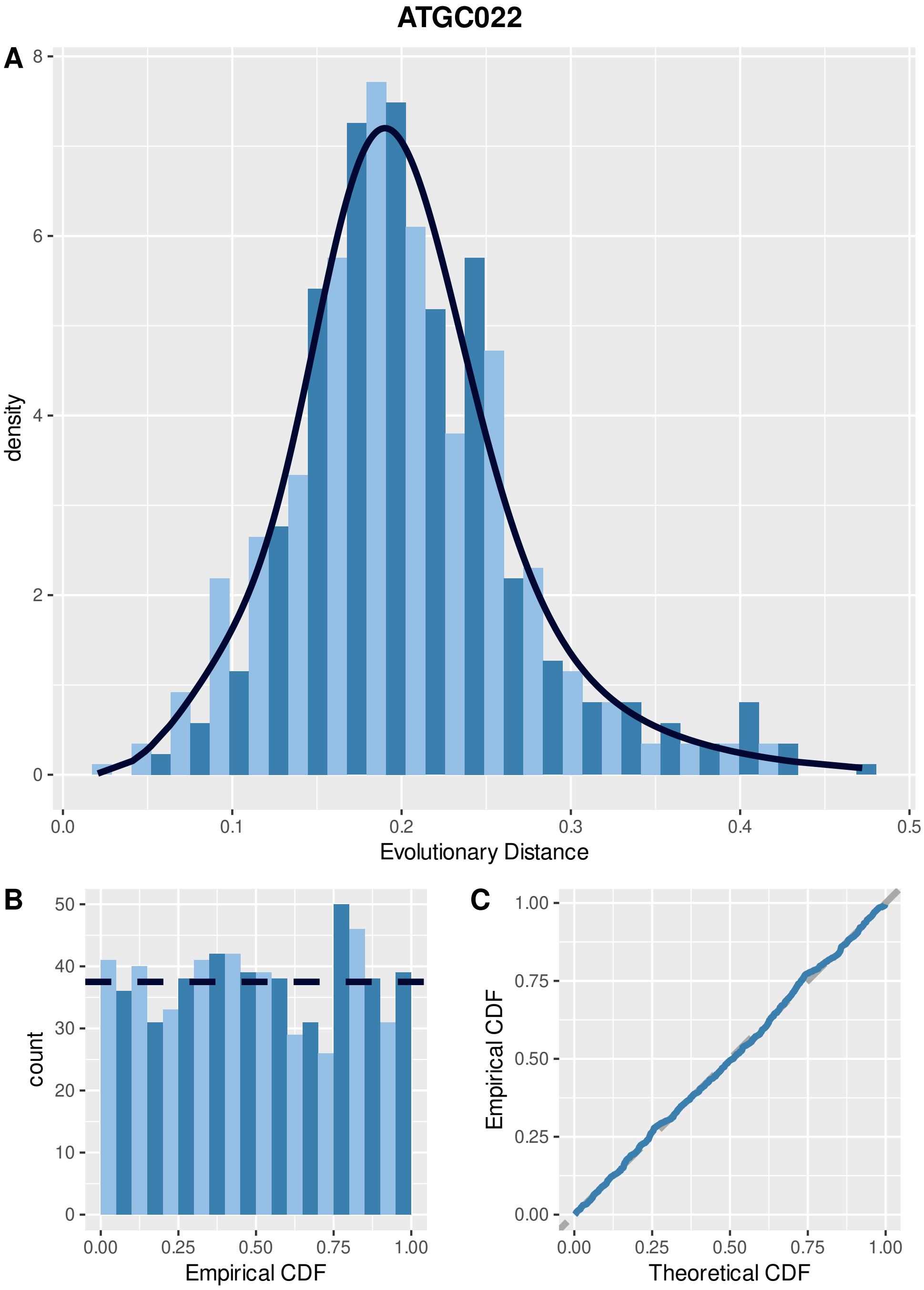

ATGC013
A
density
3
2
1
0
0.0
0.2
0.4
Evolutionary Distance
B
count
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

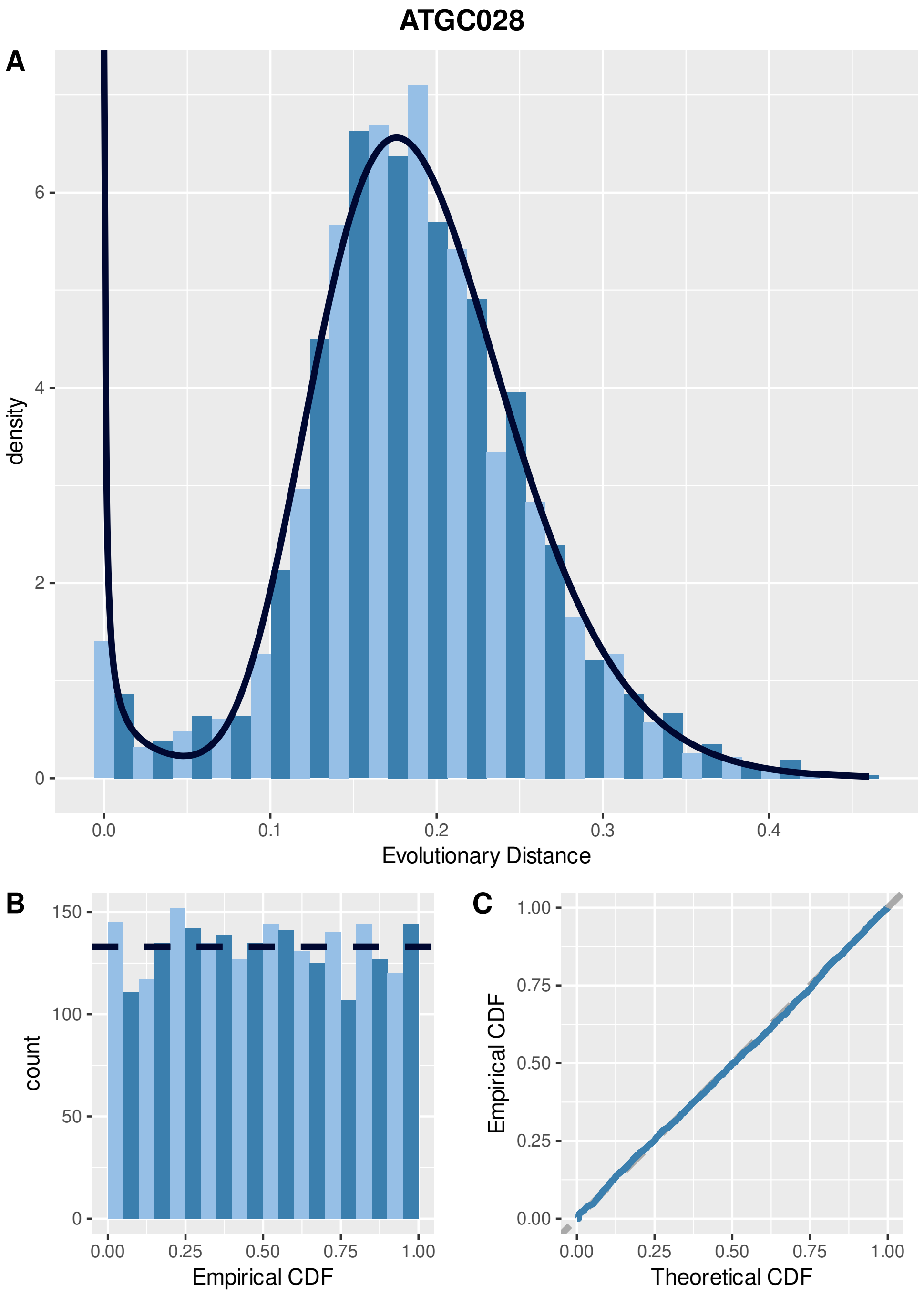

ATGC015
A
density
4
2
0
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
100
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

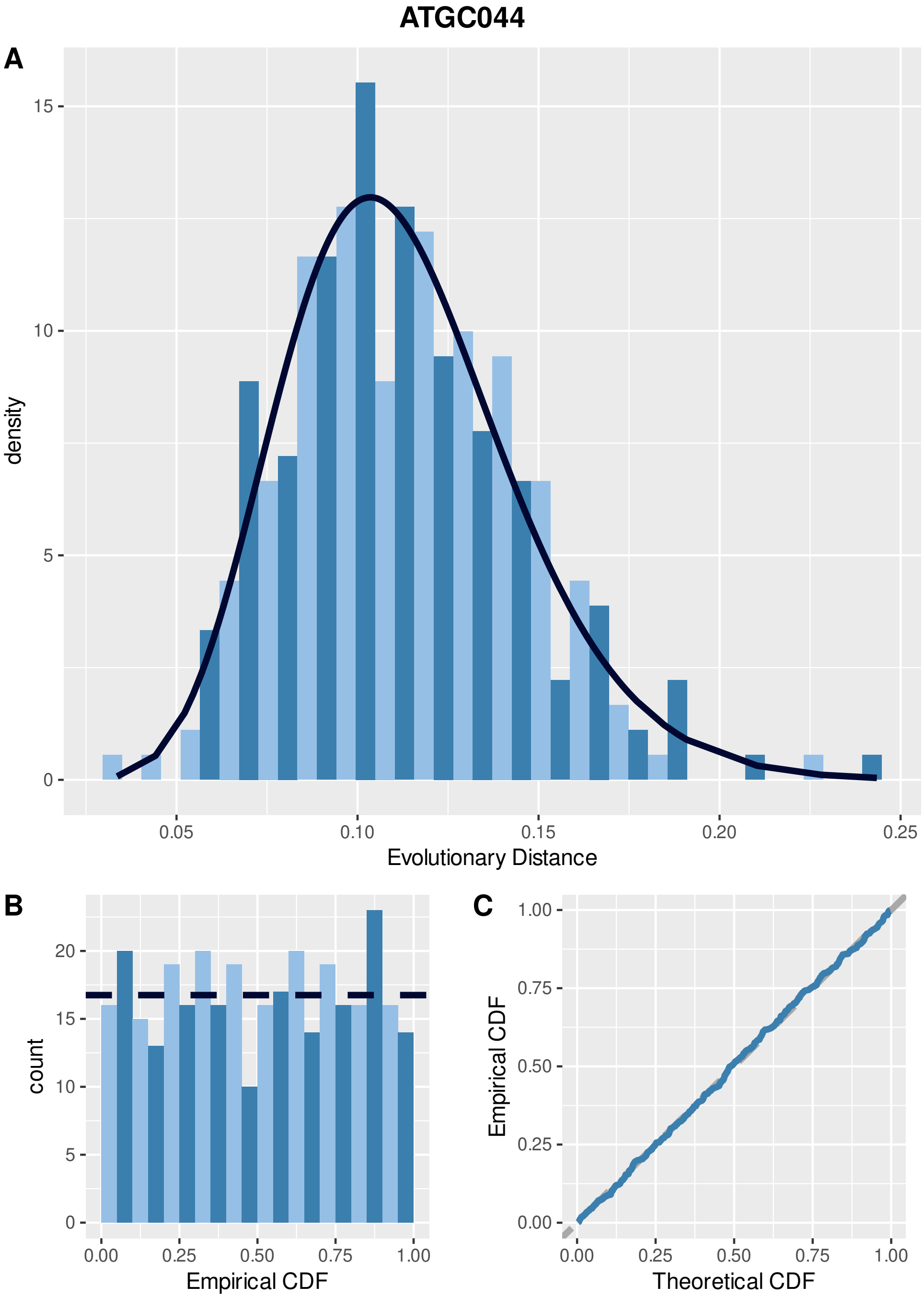

ATGC016
A
density
8
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF



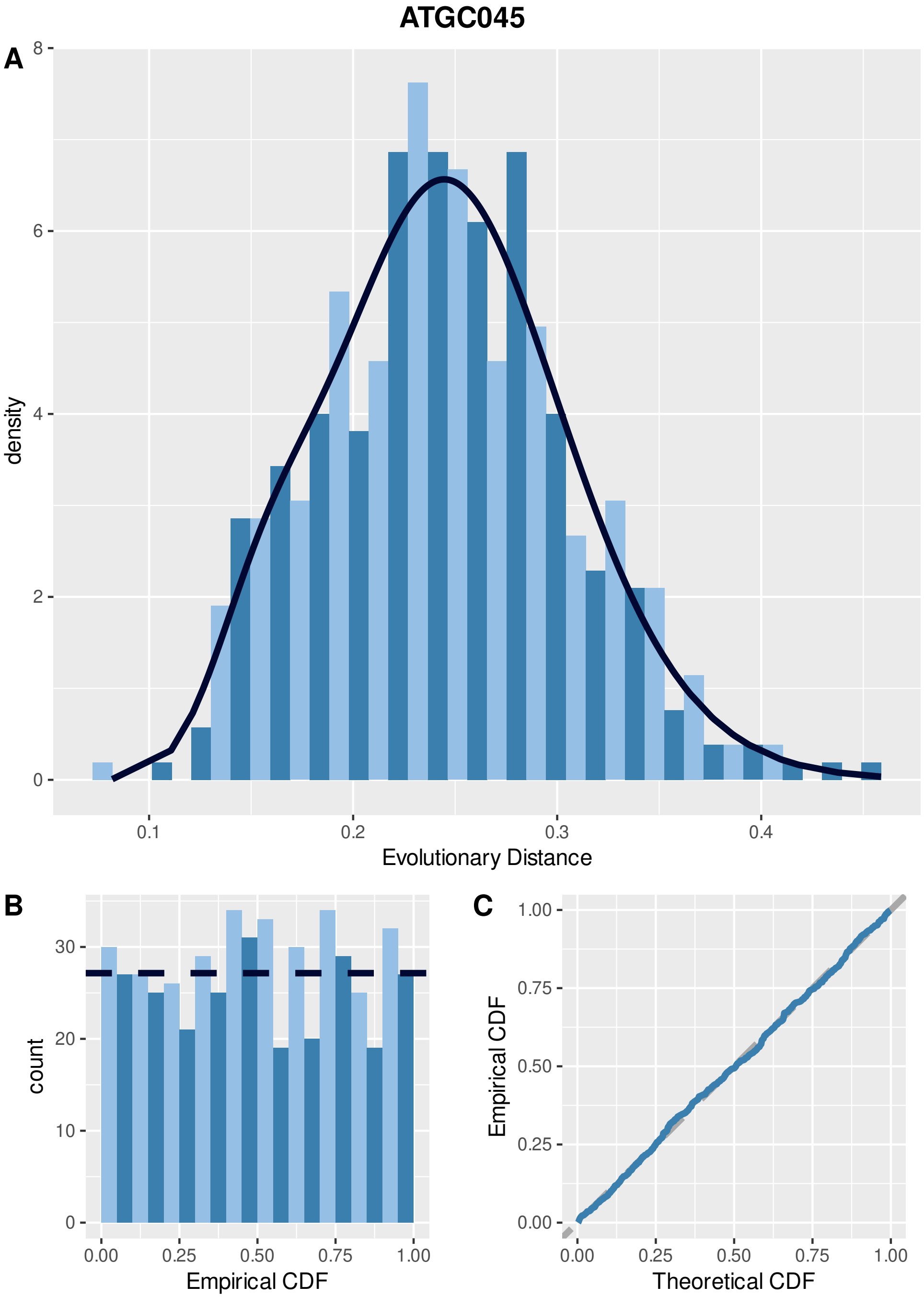
ATGC021
A
B
C
density
Evolutionary Distance
0
2
4
6
8
0.0
0.1
0.2
0.3
0.4
count
Empirical CDF
0
10
20
30
40
50
0.00
0.25
0.50
0.75
1.00
Empirical CDF
Theoretical CDF
0.00
0.25
0.50
0.75
1.00

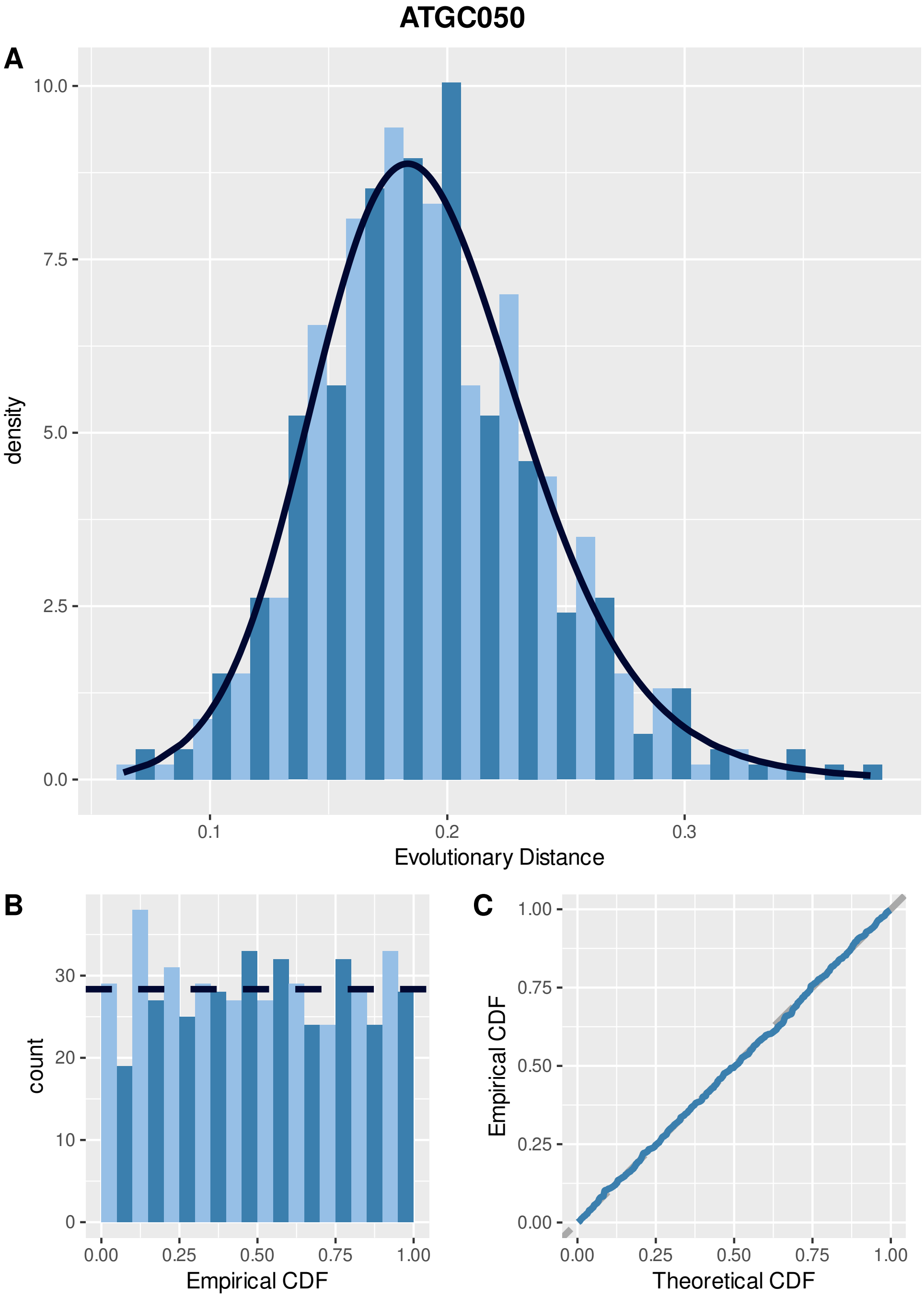

ATGC022
A
8
6
4
2
0
density
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
50
40
30
20
10
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

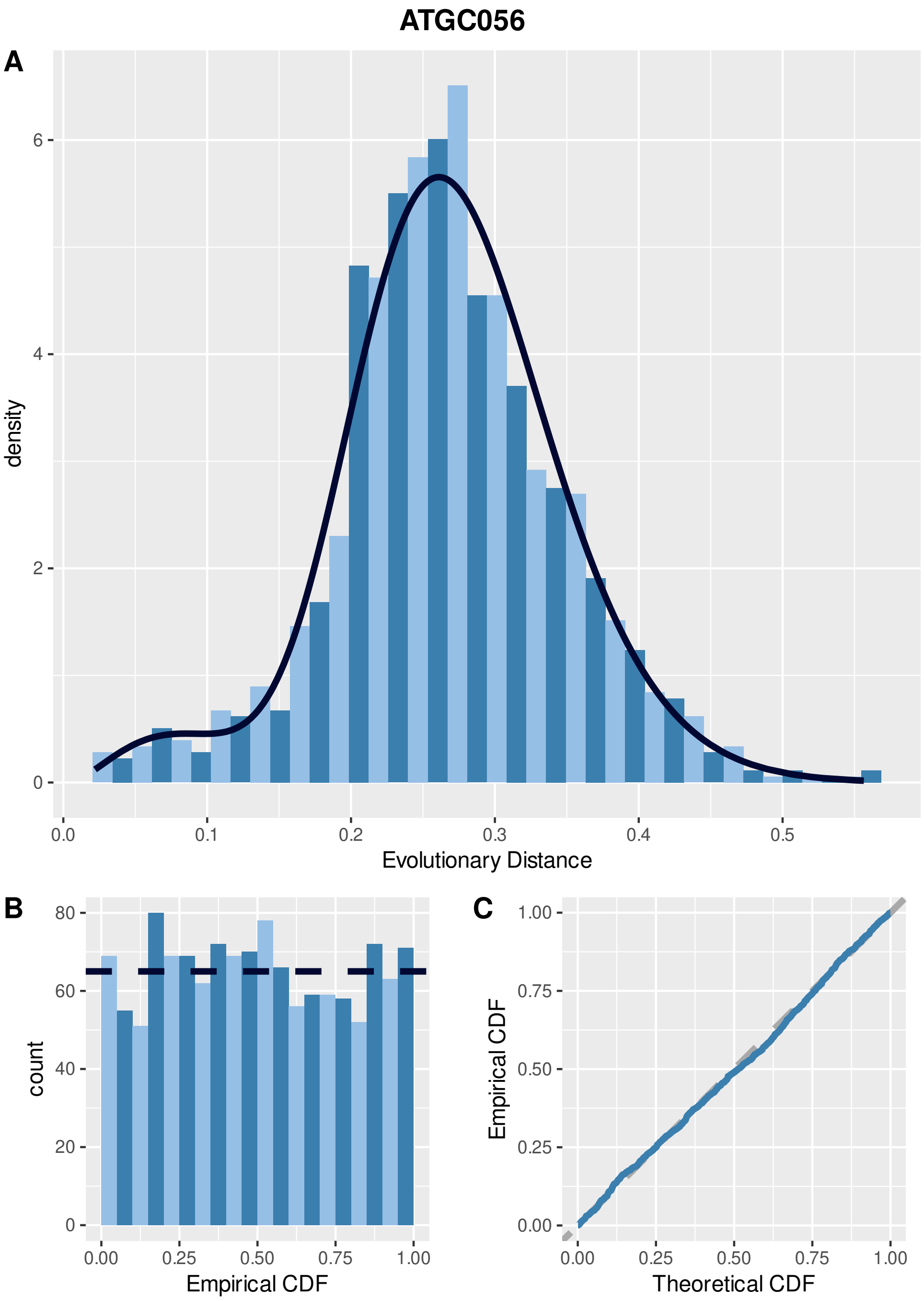

ATGC028
A
density
0
2
4
6
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
0
50
100
150
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

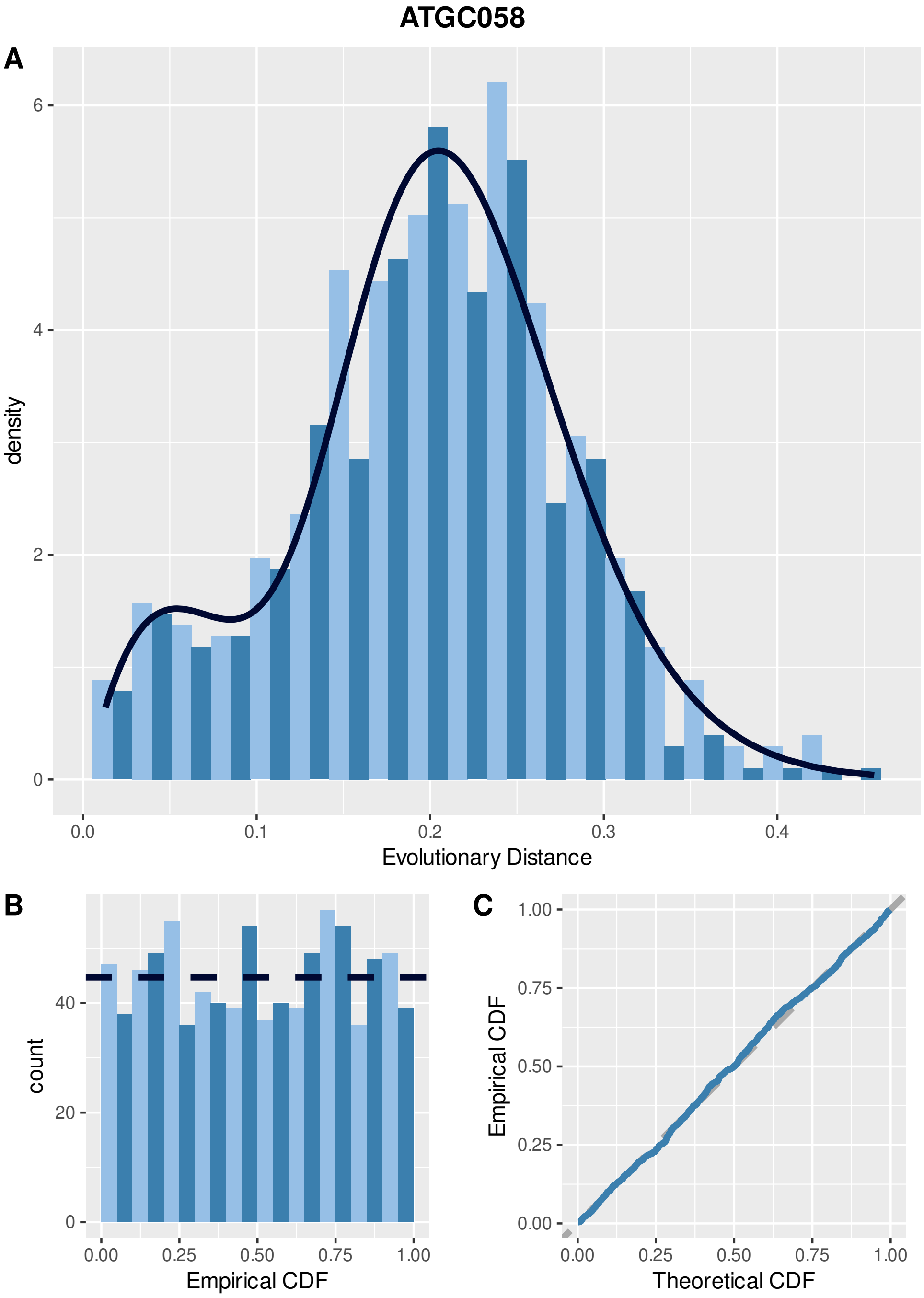


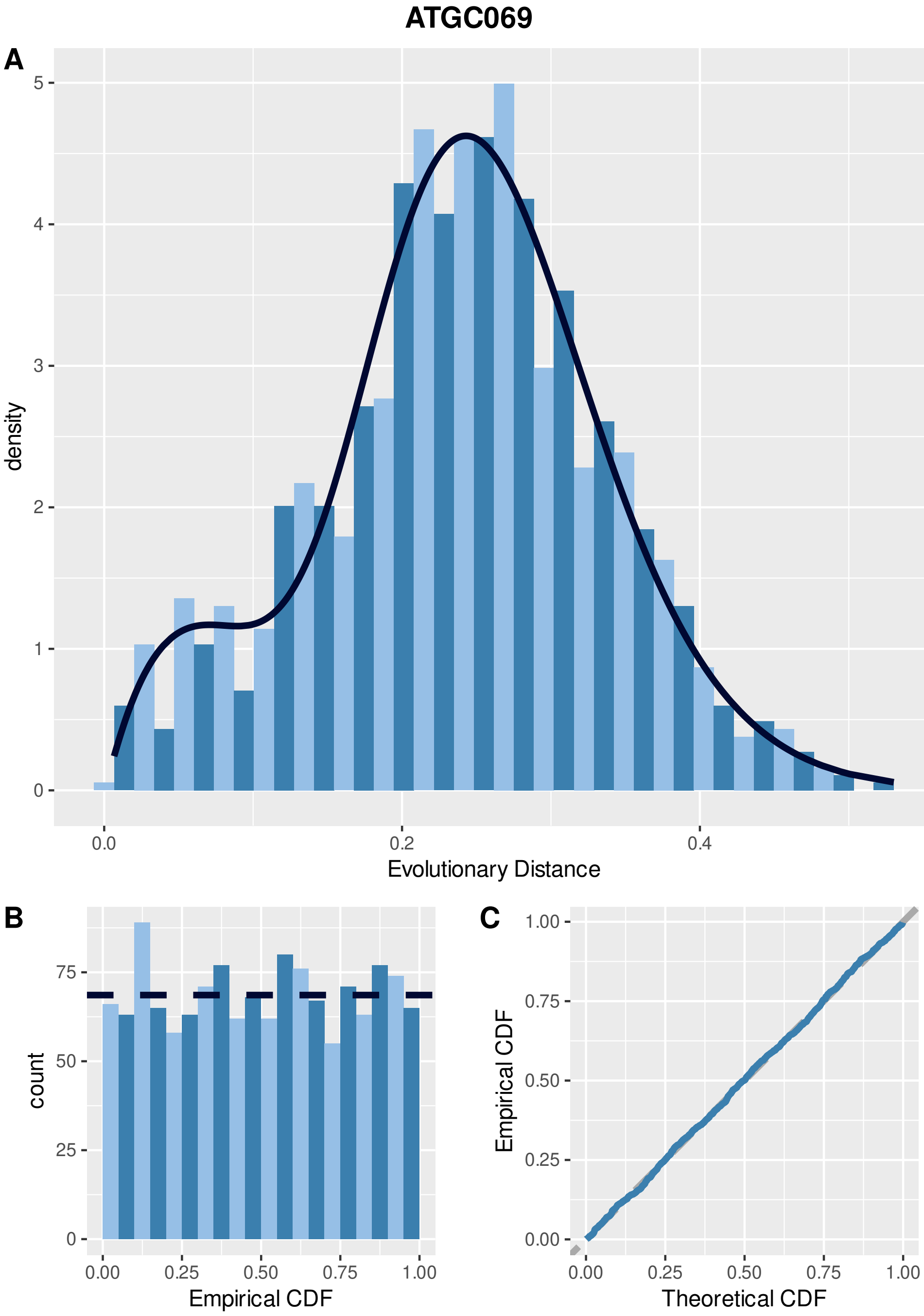
ATGC044
A
density
15
10
5
0
0.05
0.10
0.15
0.20
0.25
Evolutionary Distance
B
count
20
15
10
5
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

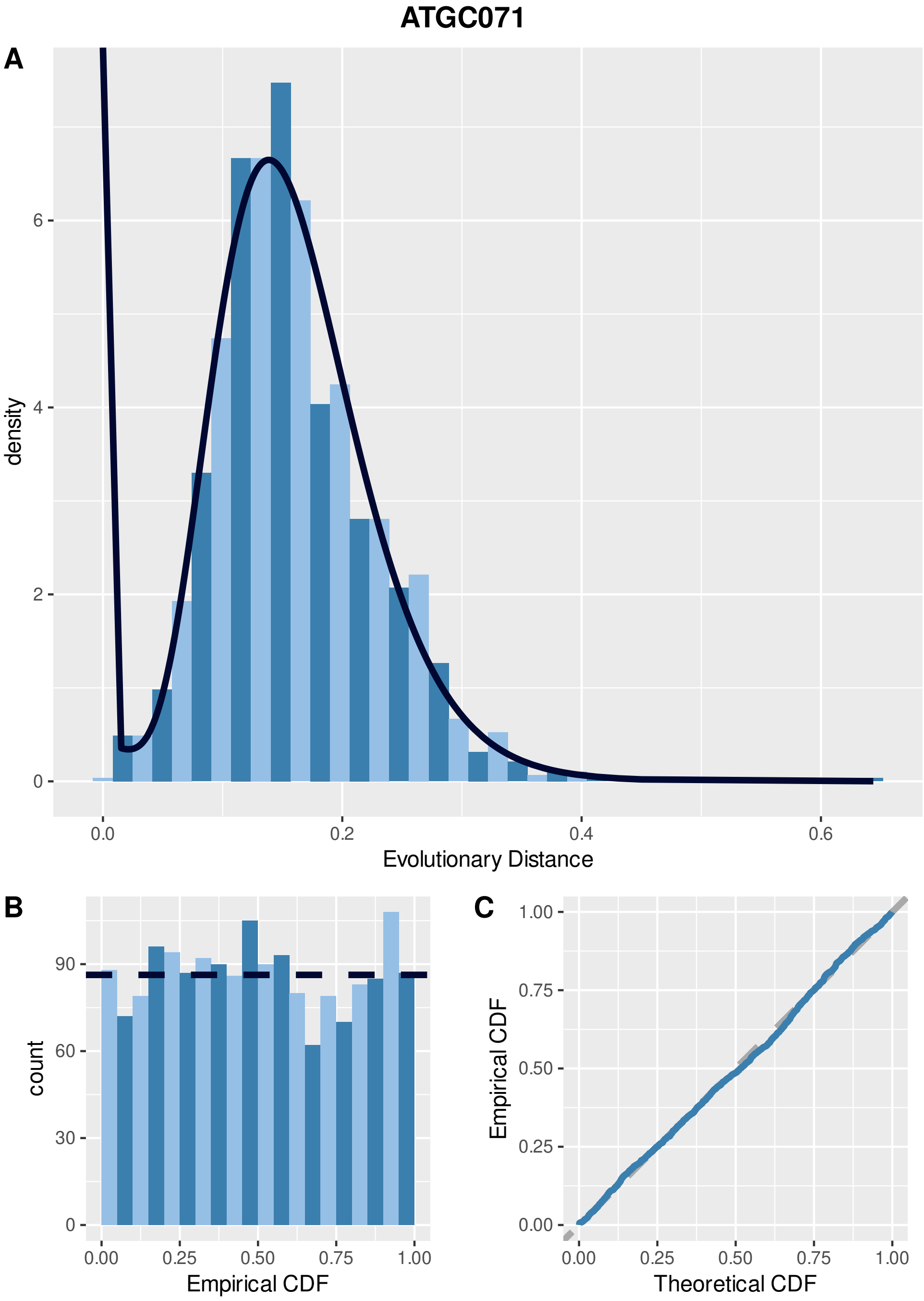

ATGC045
A
8
6
4
2
0
density
0.1
0.2
0.3
0.4
Evolutionary Distance
B
30
20
10
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

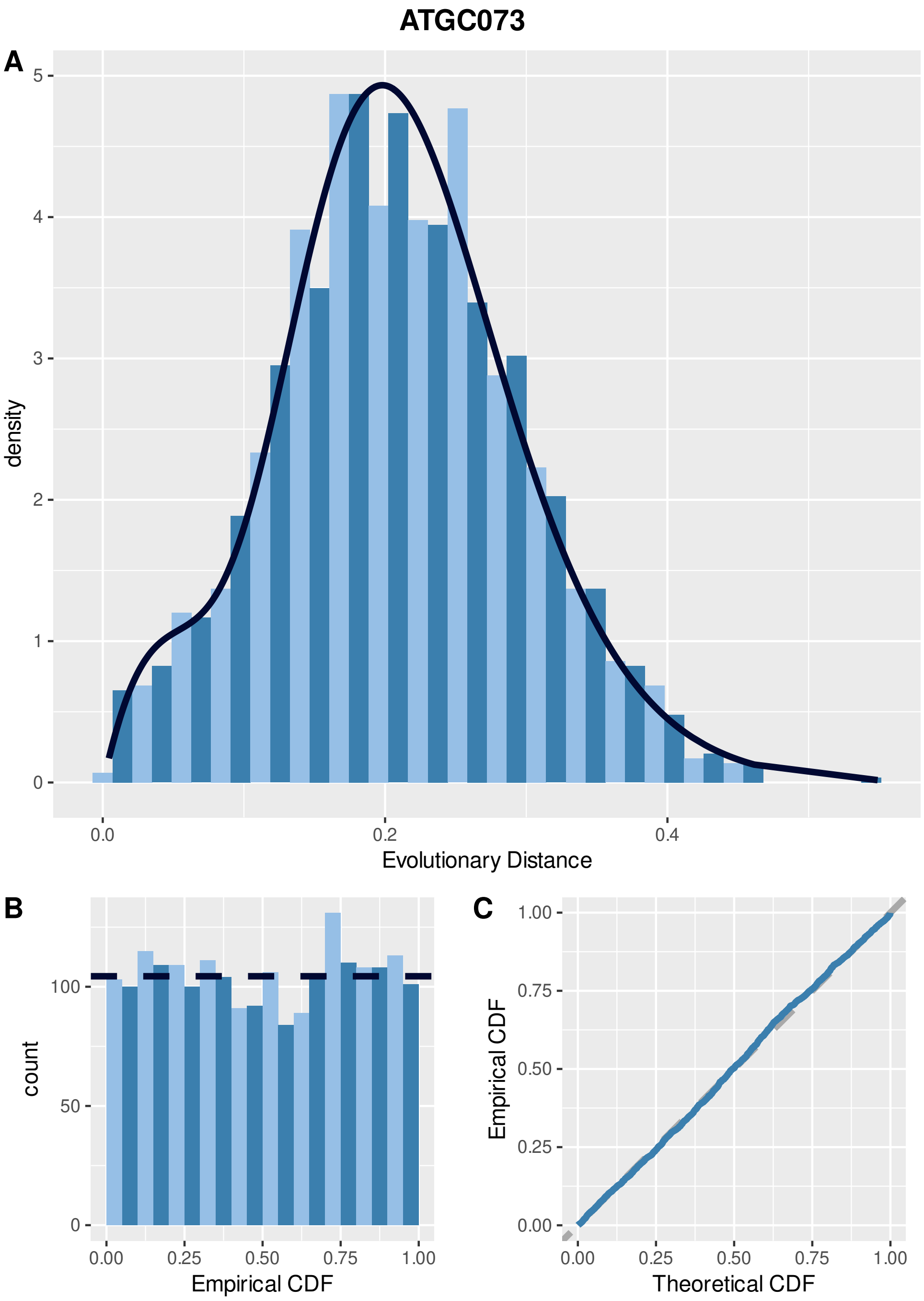

ATGC050
A
density
10.0
7.5
5.0
2.5
0.0
0.1
0.2
0.3
Evolutionary Distance
B
count
30
20
10
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

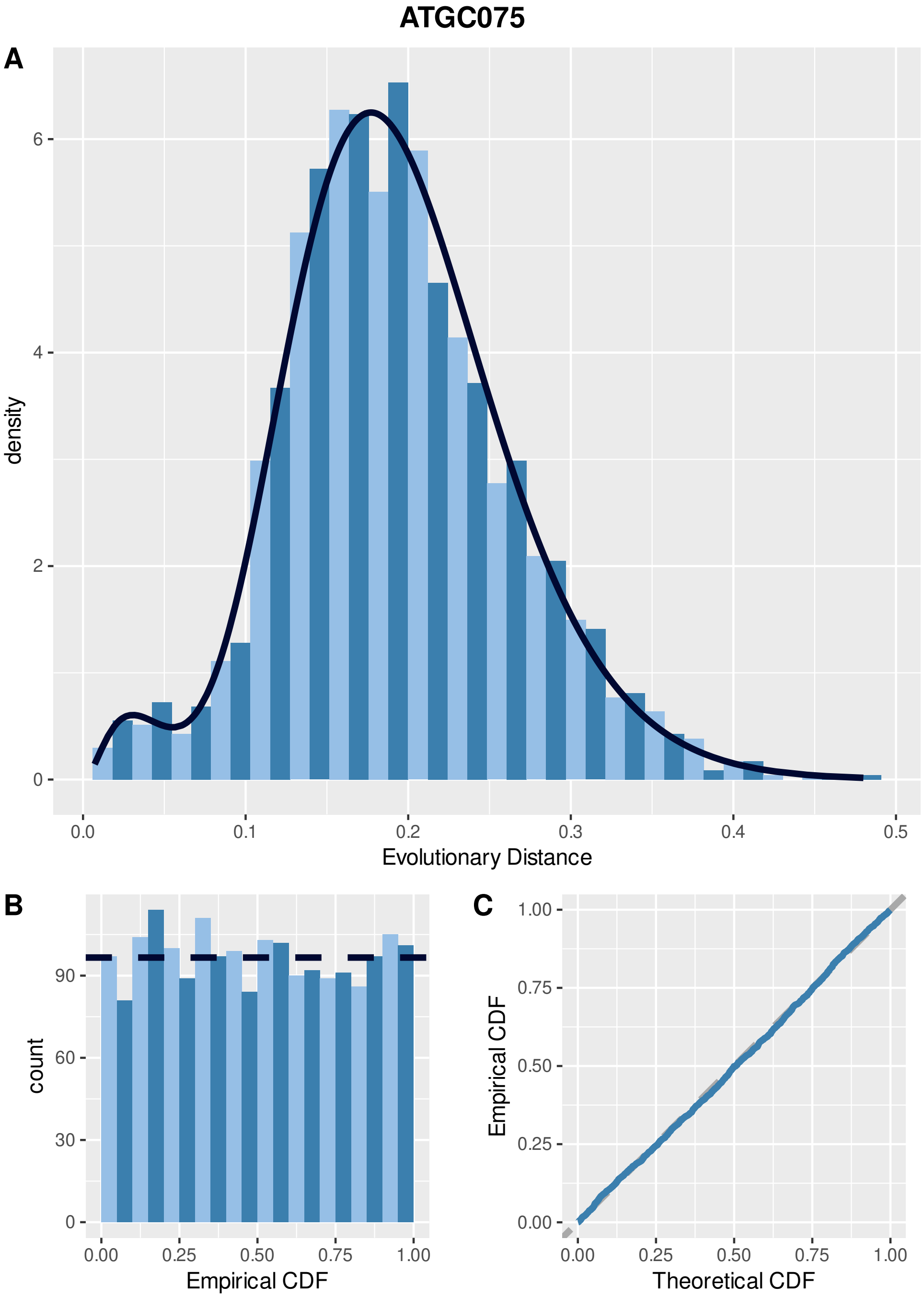

ATGC056
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
80
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

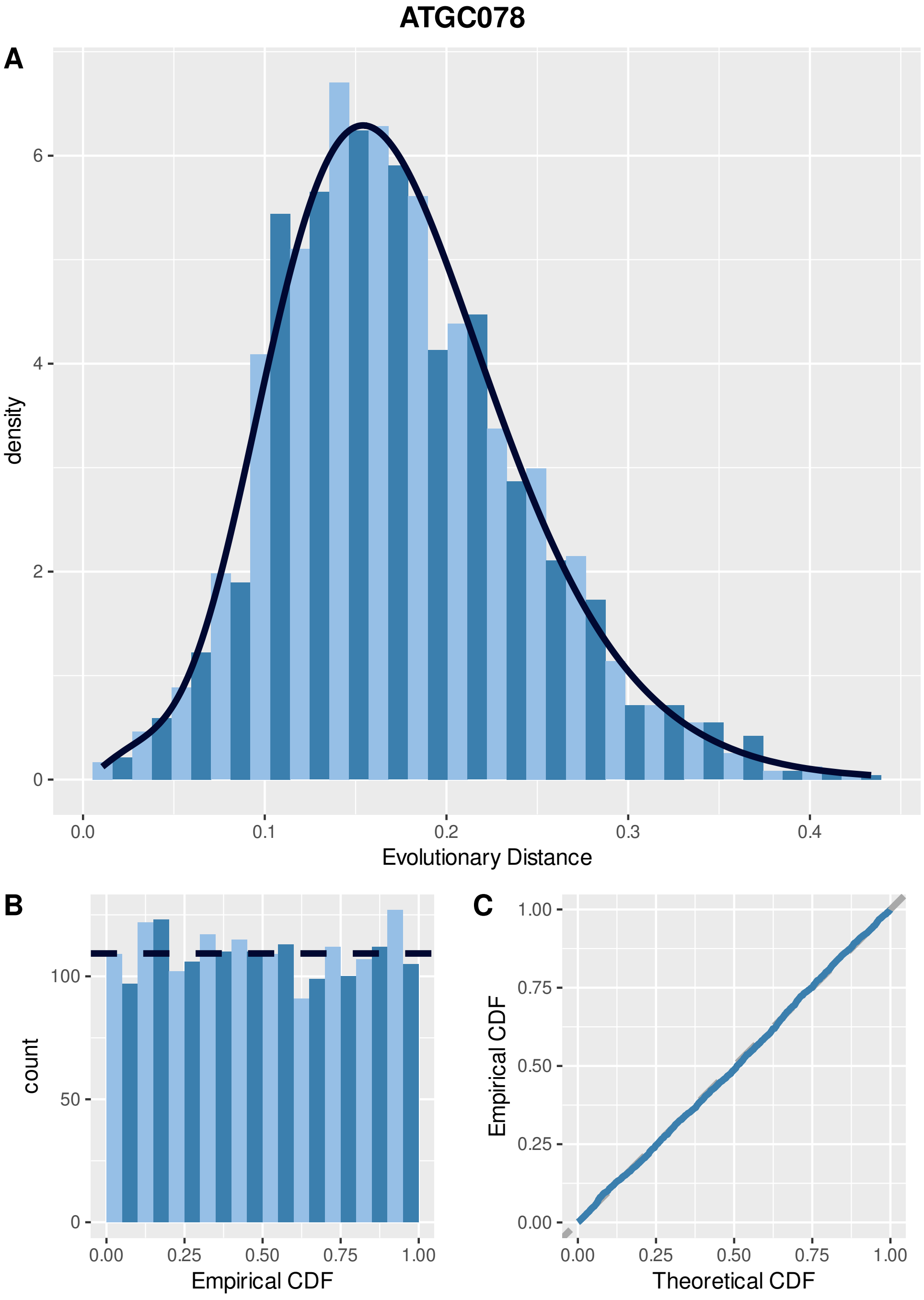

ATGC058
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

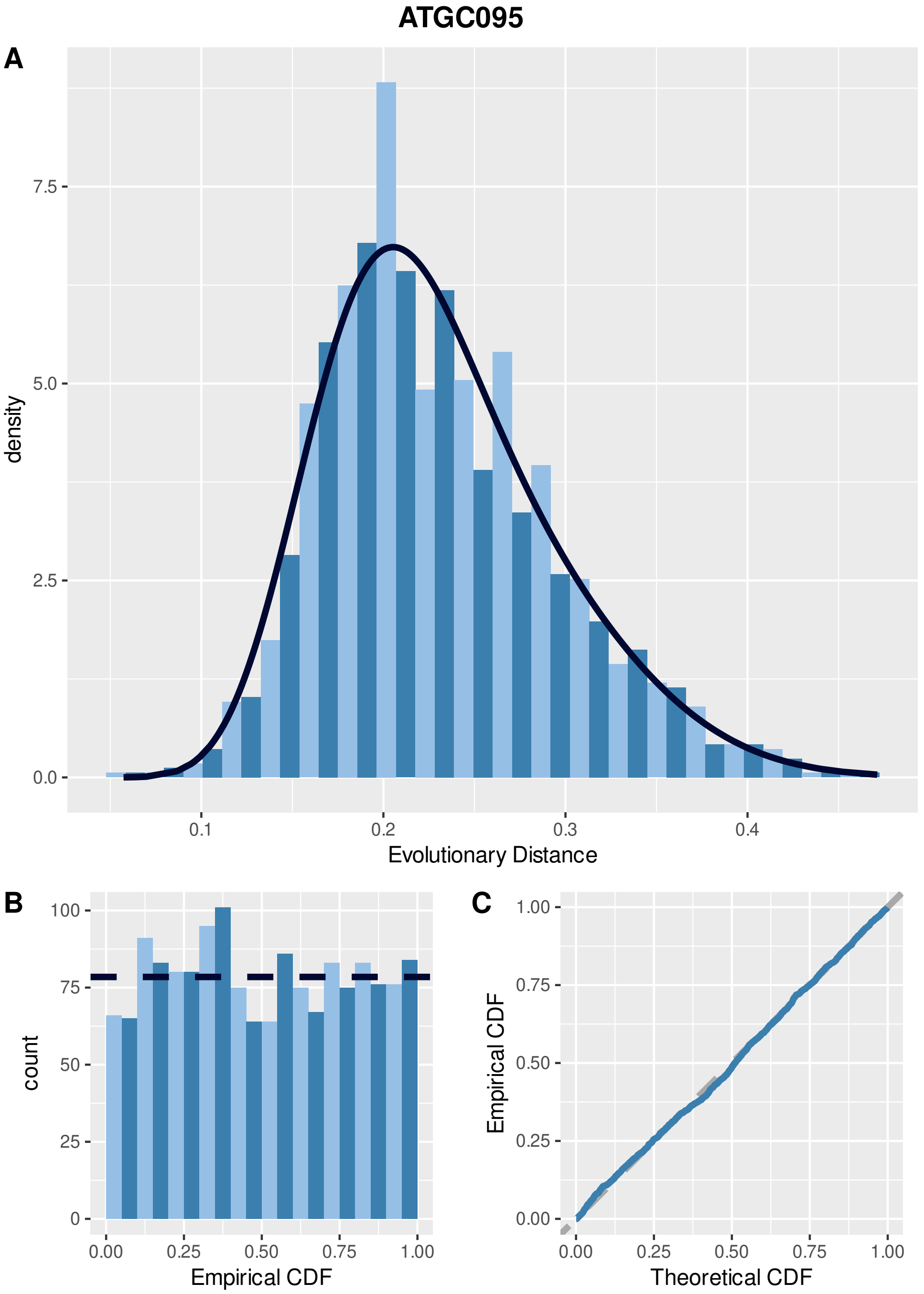

ATGC069
A
density
5
4
3
2
1
0
0.0
0.2
0.4
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

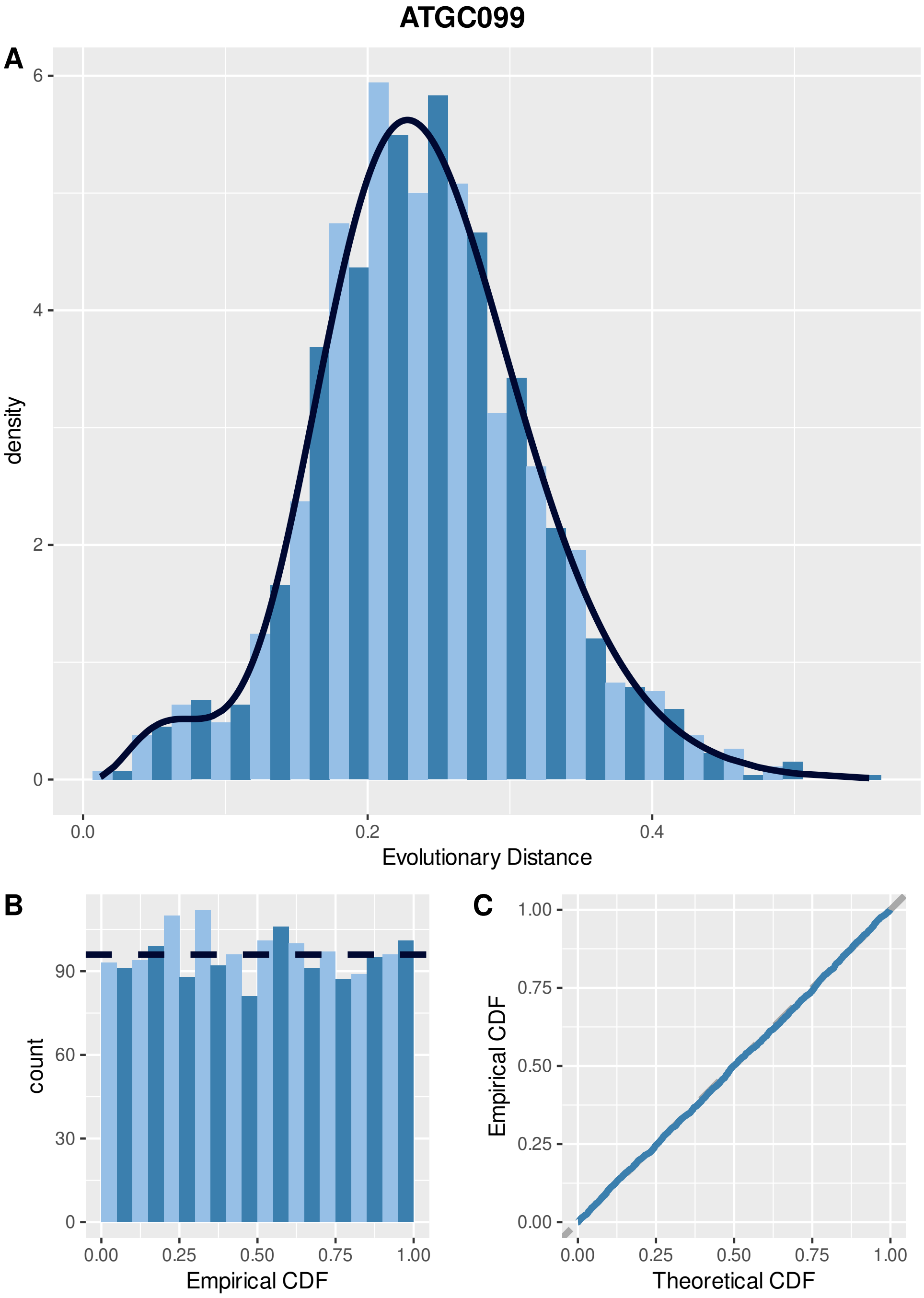

ATGC071
A
density
0
2
4
6
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
0
30
60
90
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

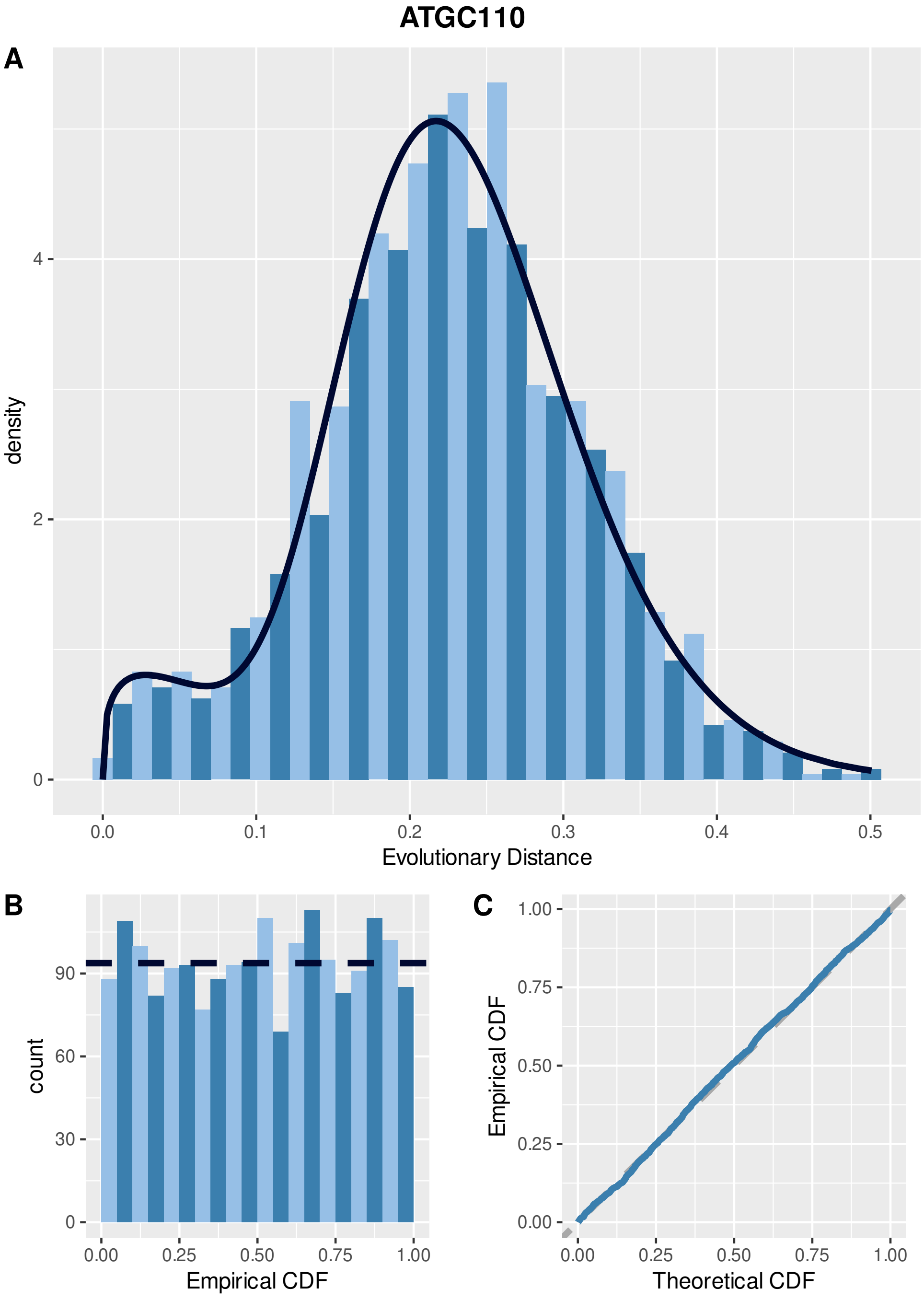

ATGC073
A
density
0
1
2
3
4
5
0.0
0.2
0.4
Evolutionary Distance
B
count
0
50
100
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

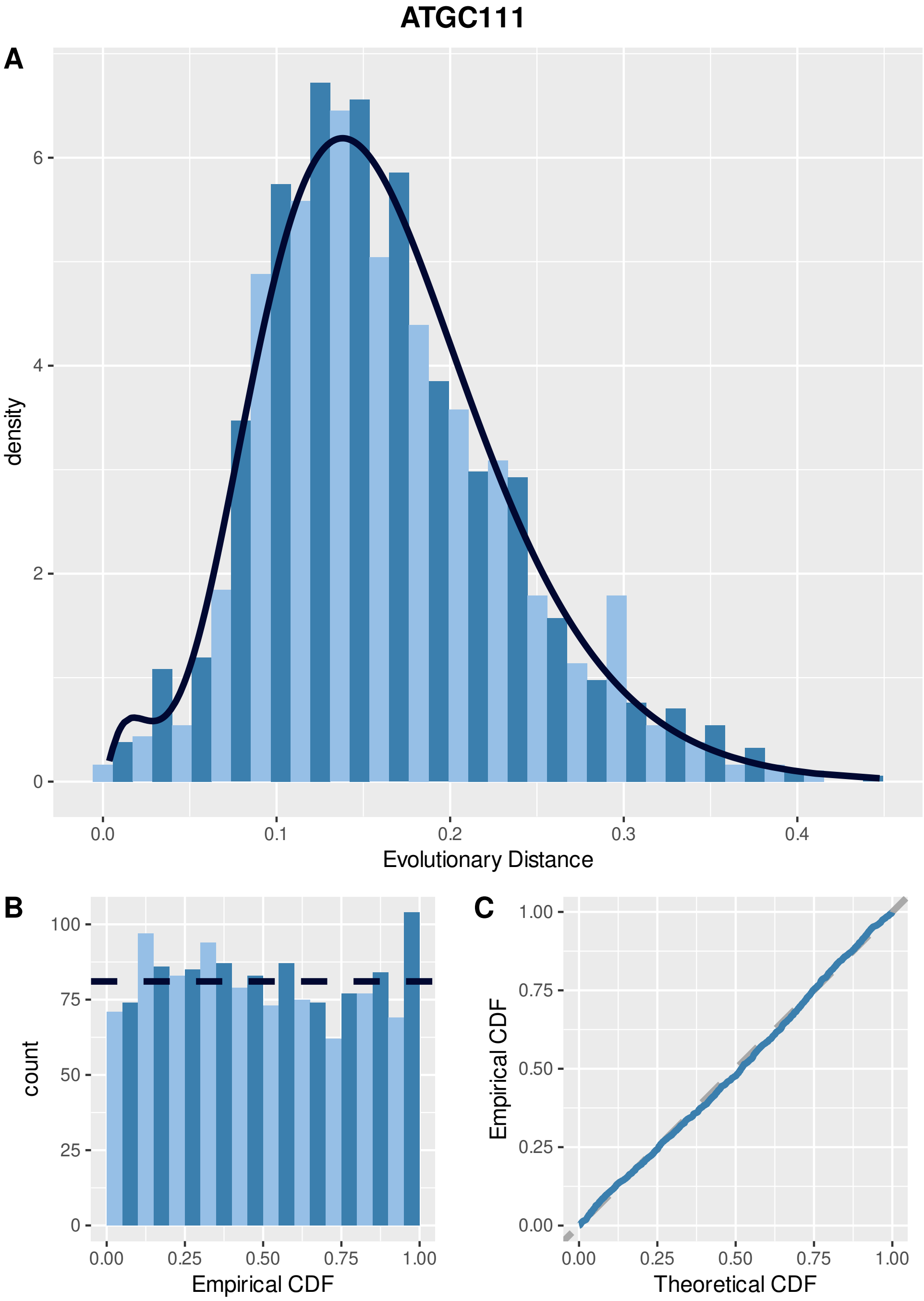

ATGC075
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
90
60
30
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

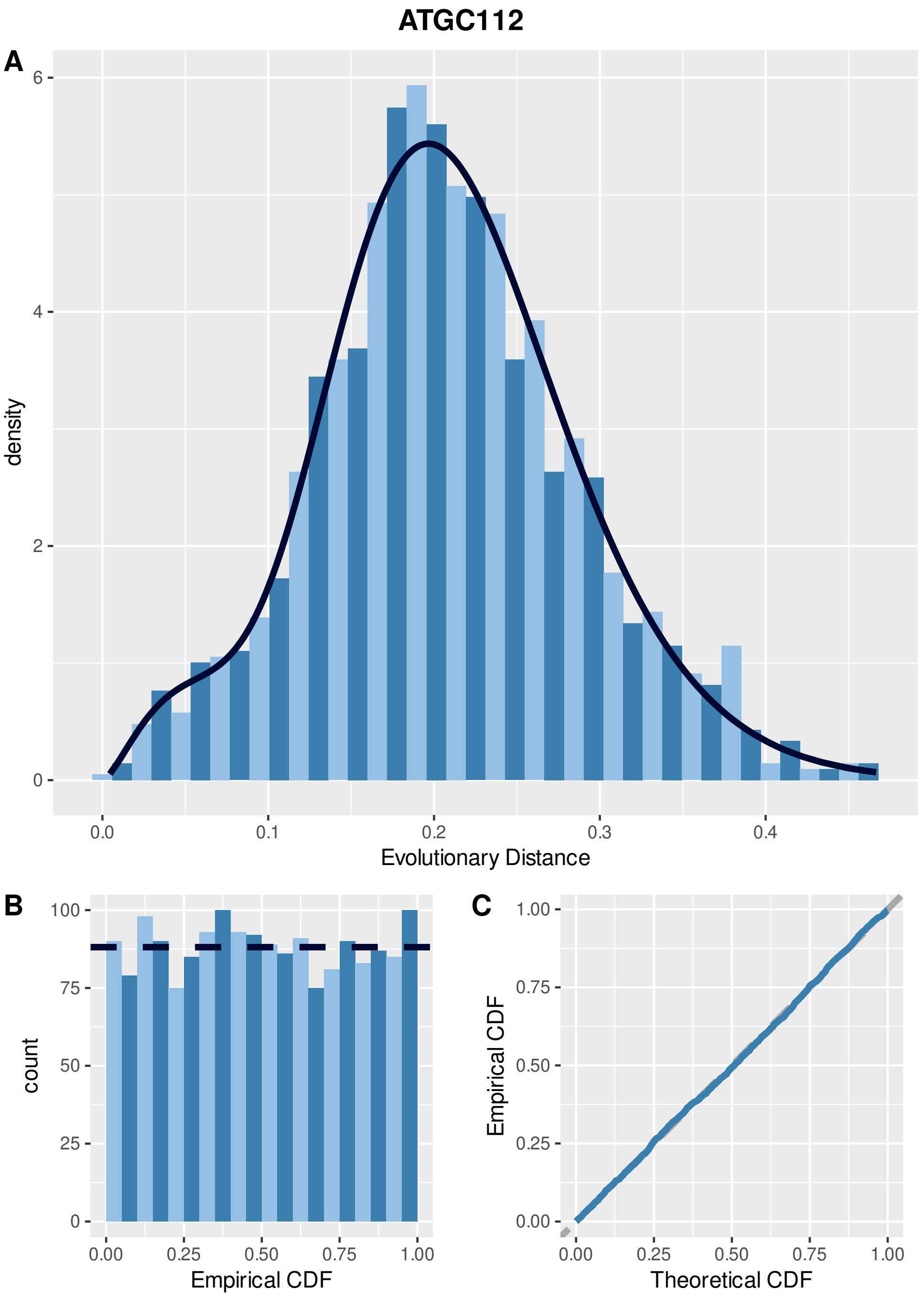

ATGC078
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
100
50
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

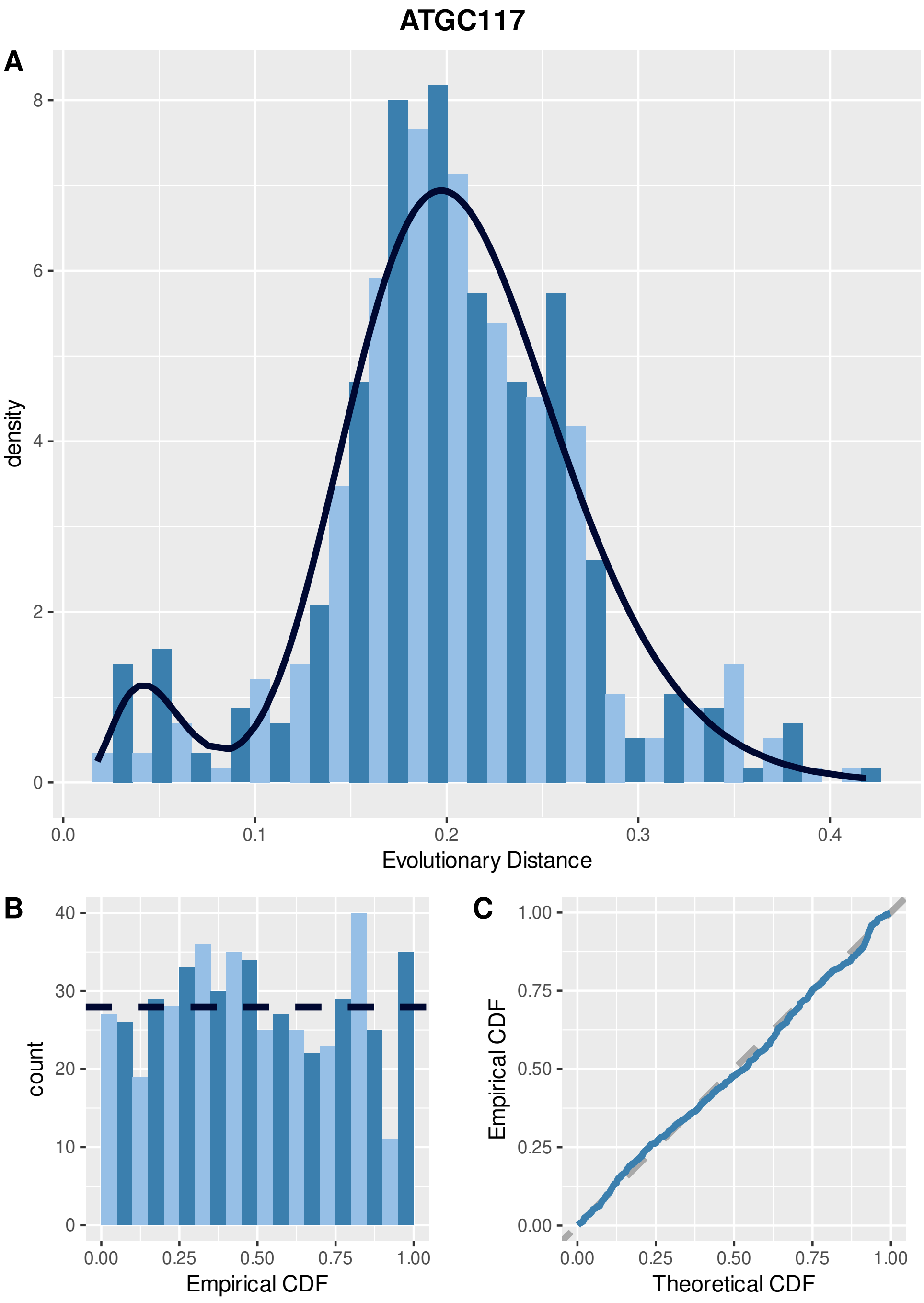

ATGC095
A
density
7.5
5.0
2.5
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
100
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

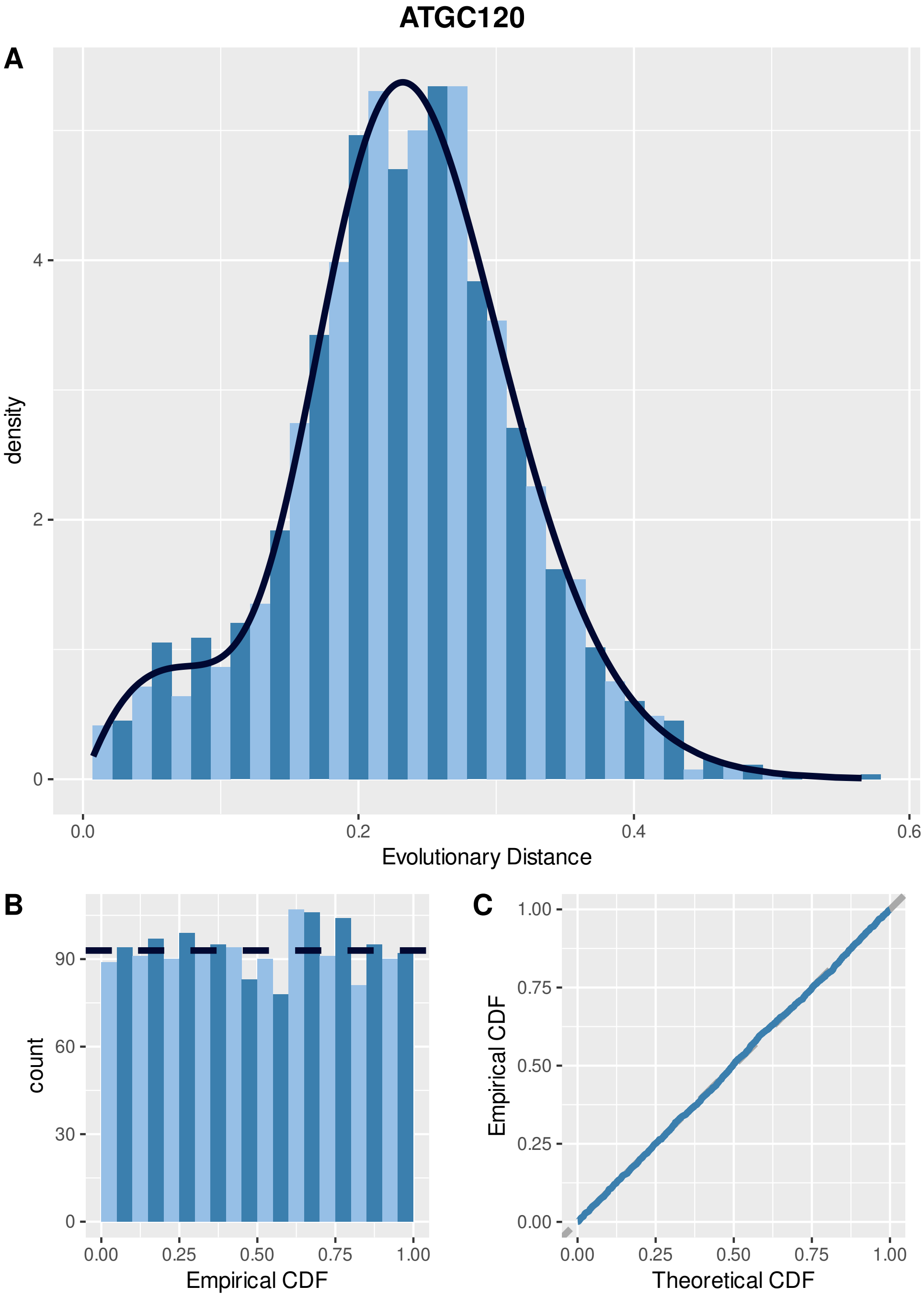

ATGC099
A
density
6
4
2
0
0.0
0.2
0.4
Evolutionary Distance
B
count
90
60
30
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

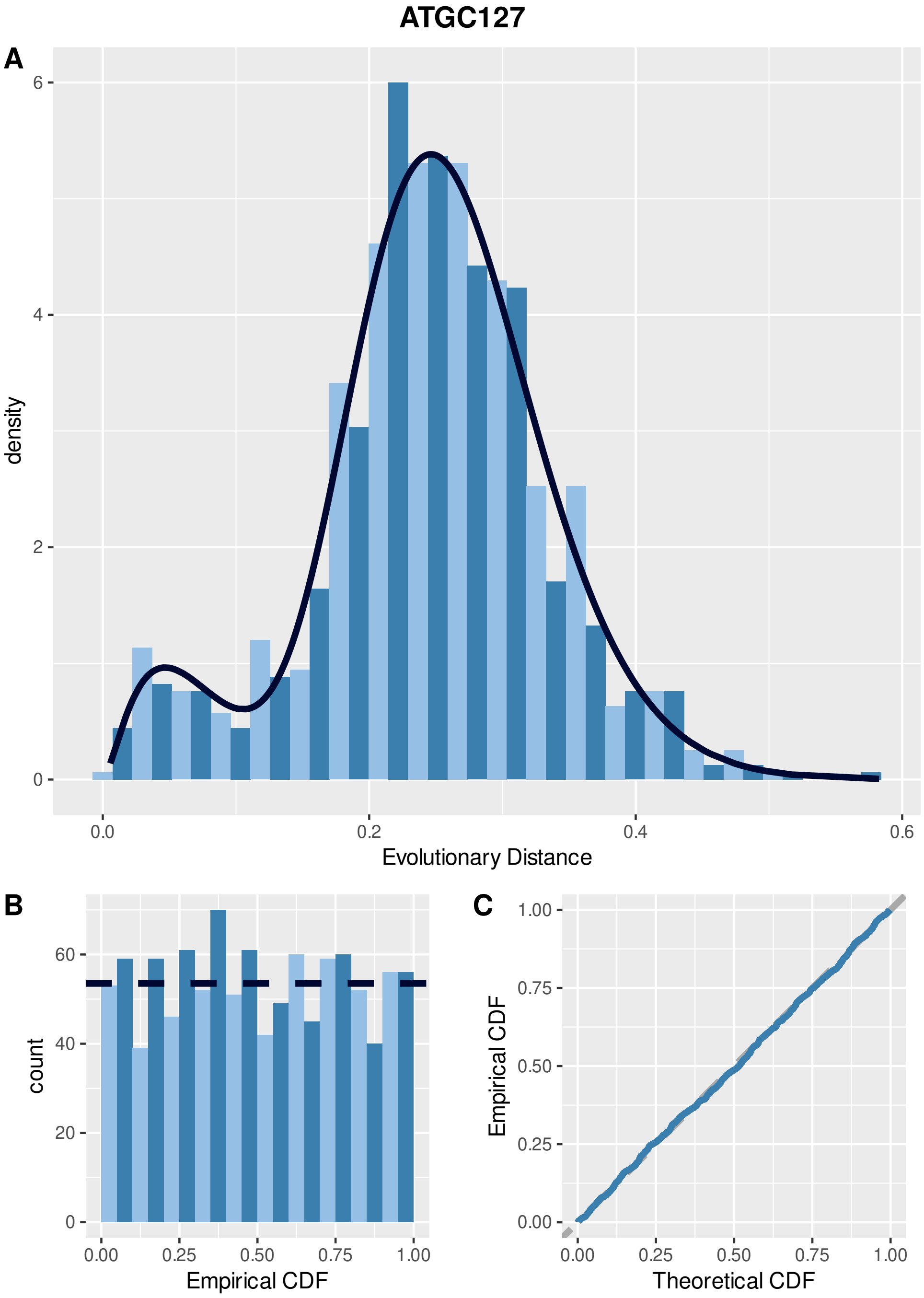

ATGC110
A
density
0
2
4
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
0
30
60
90
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

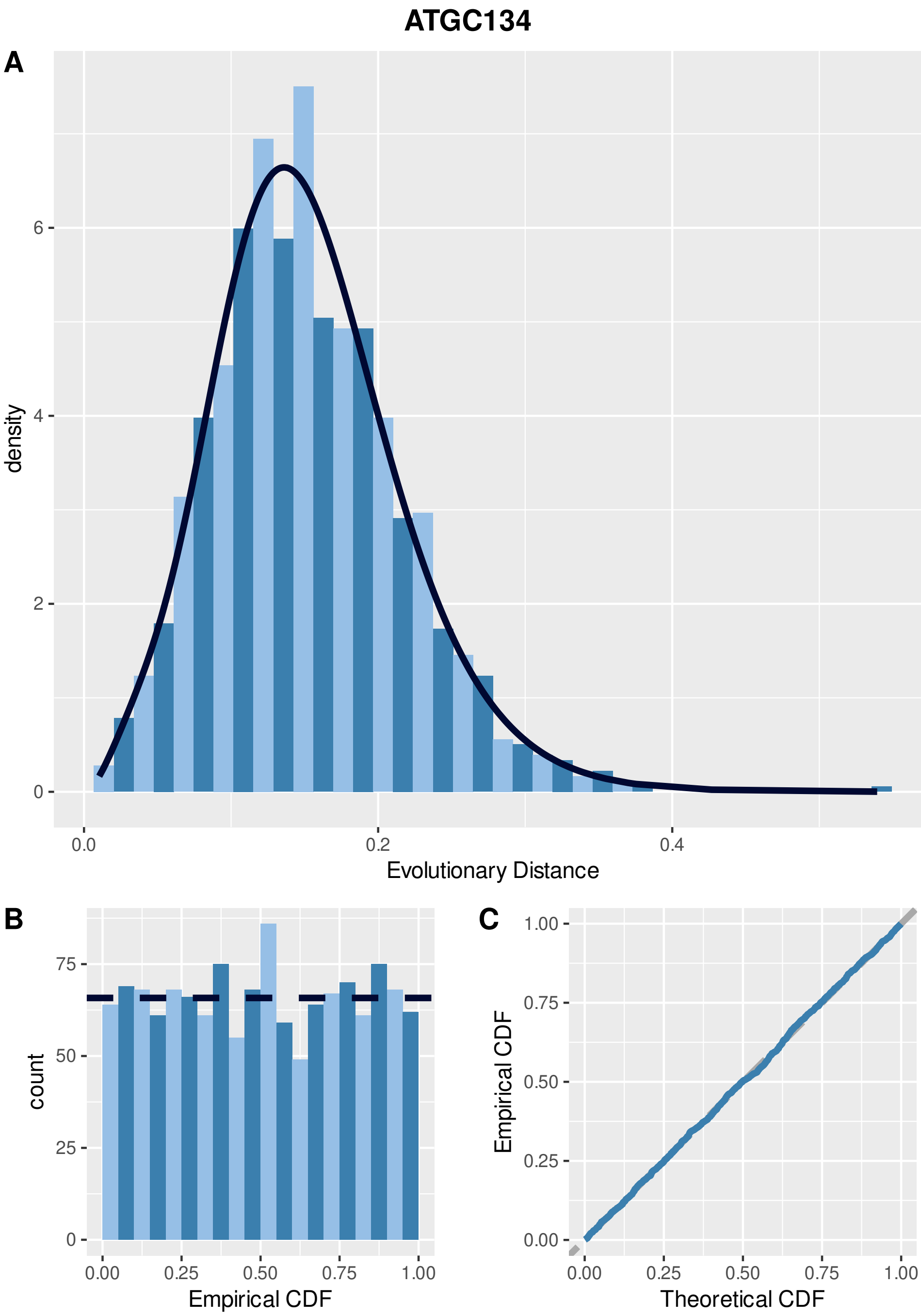

ATGC111
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
100
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

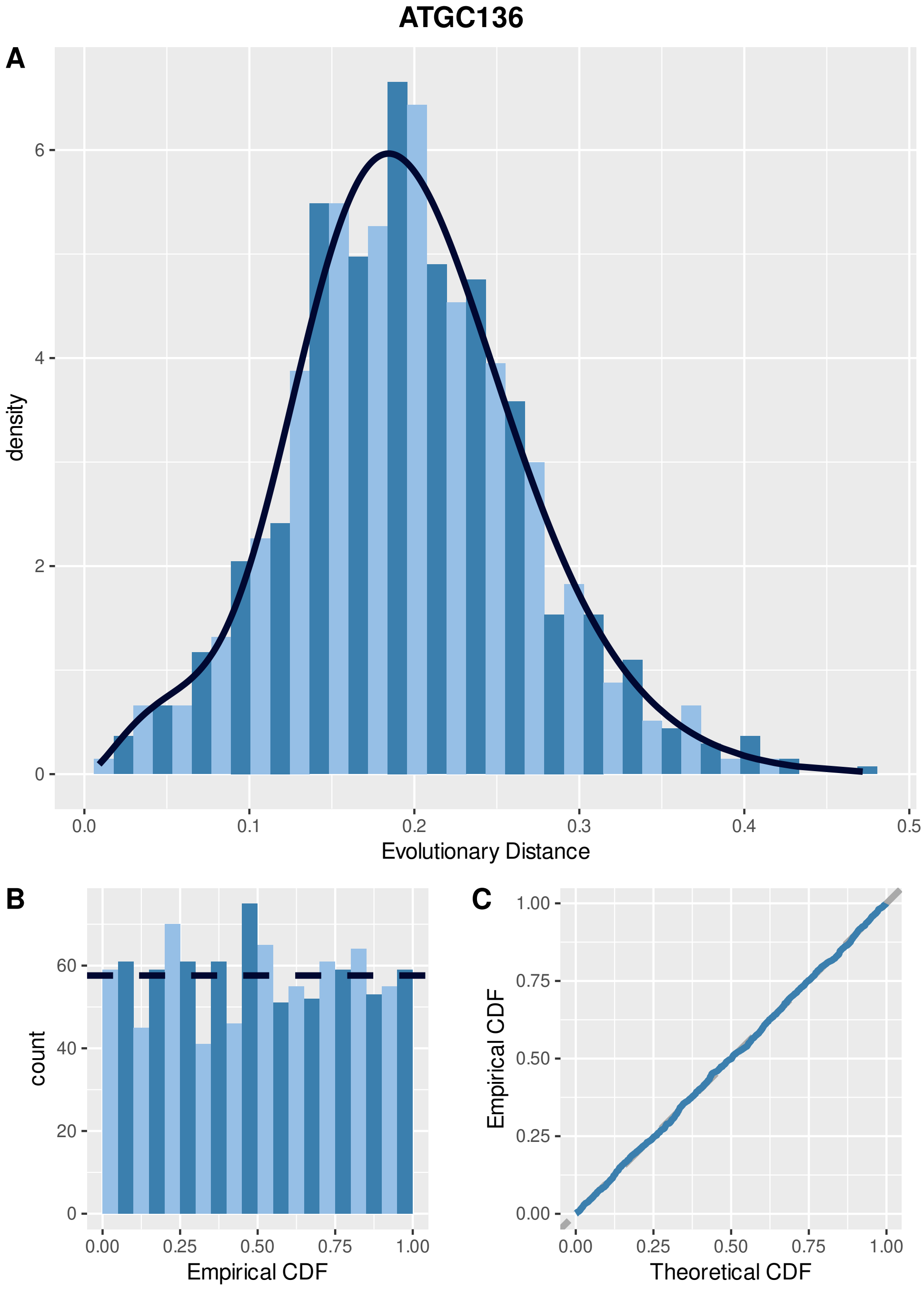

ATGC112
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
100
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

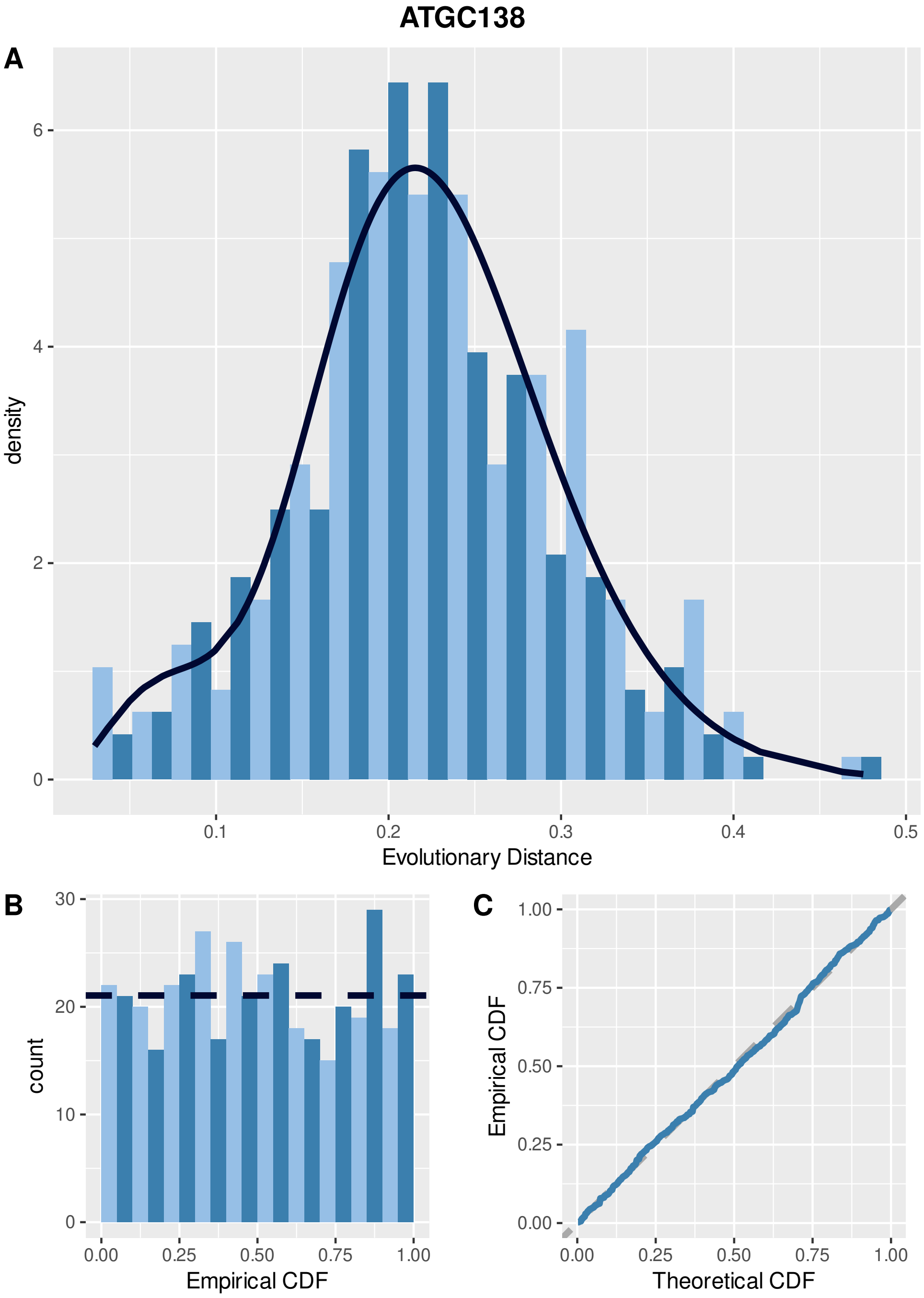

ATGC117
A
density
0
2
4
6
8
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
0
10
20
30
40
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

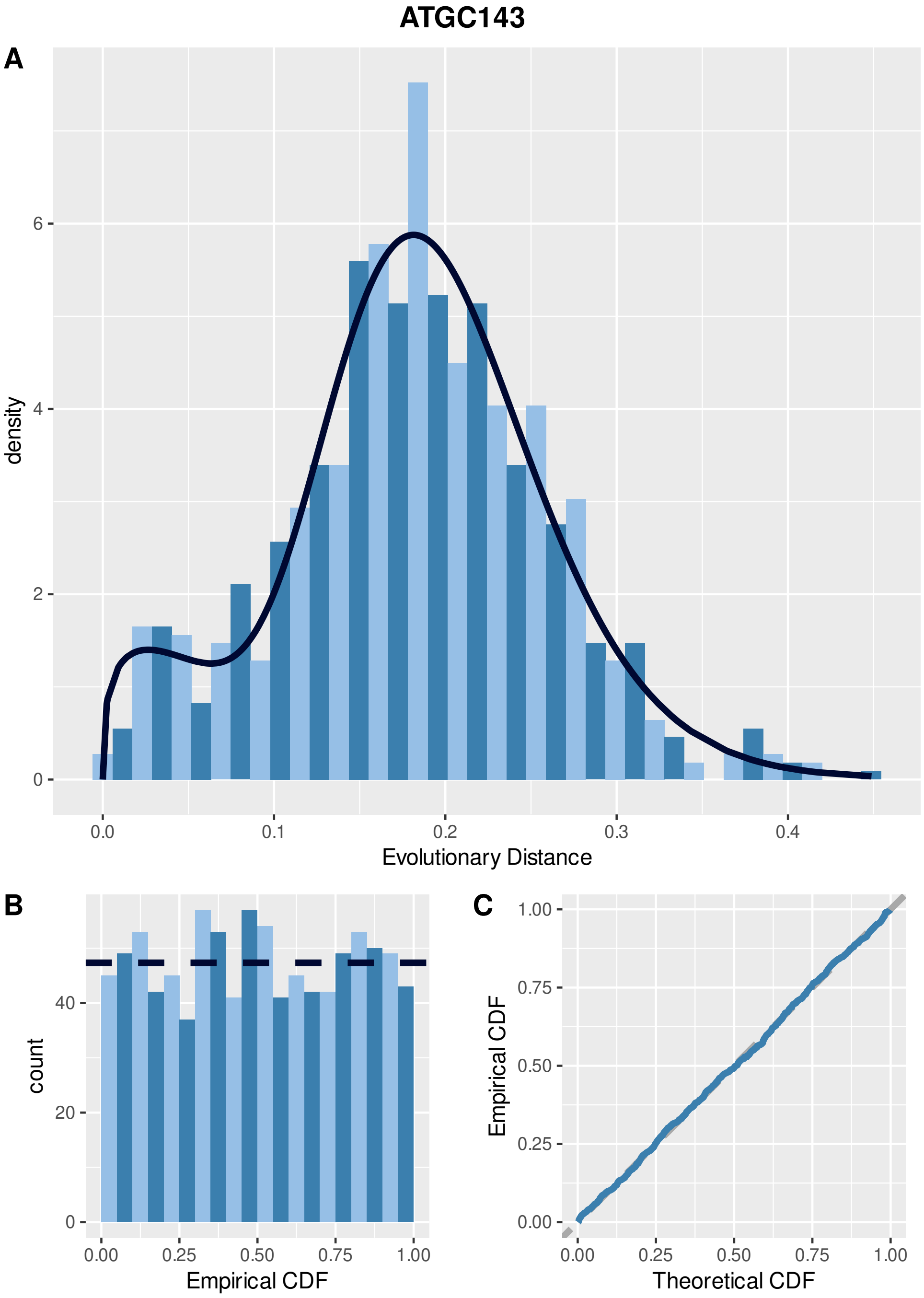

ATGC120
A
density
0
2
4
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
0
30
60
90
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

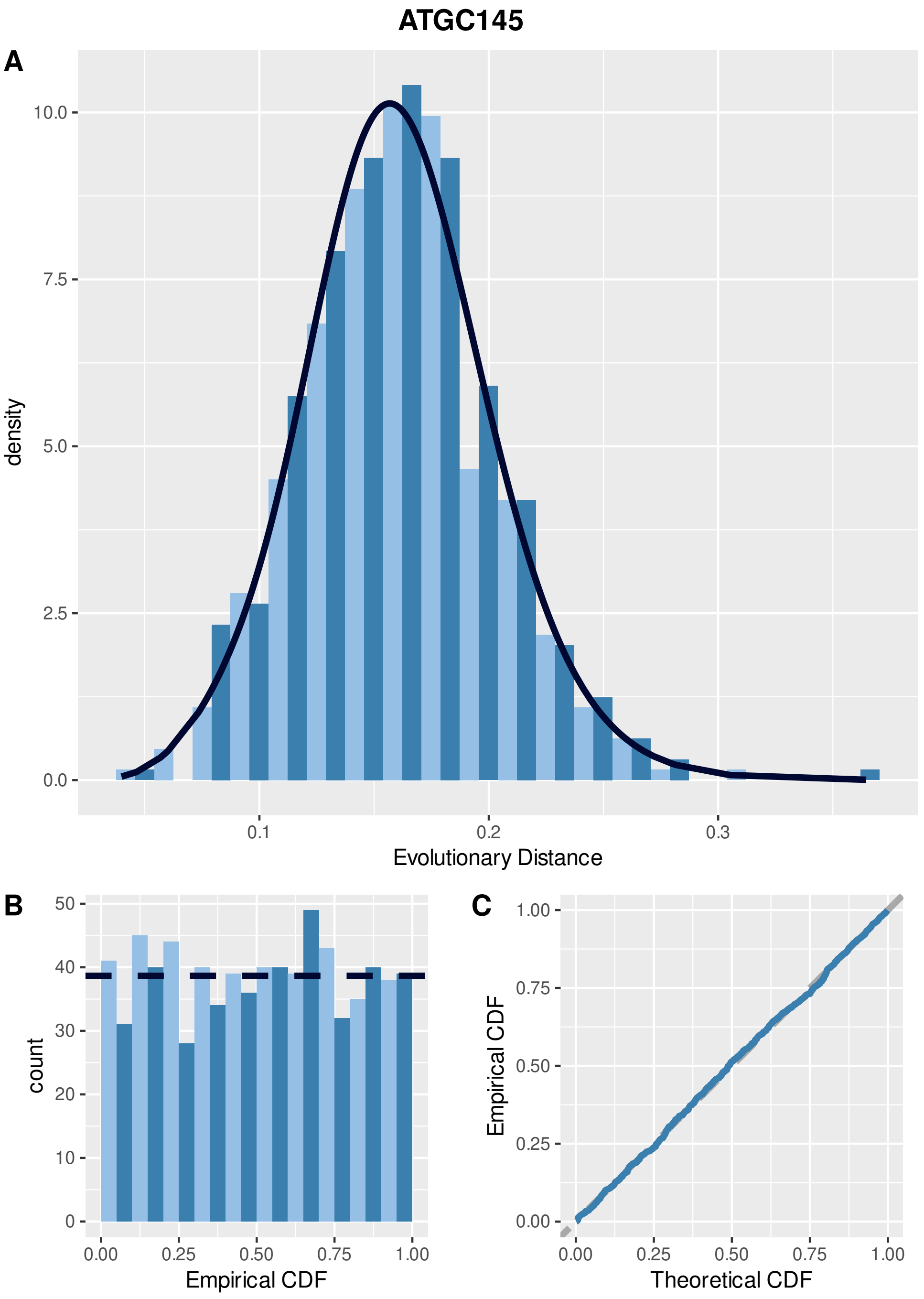

ATGC127
A
6
4
2
0
density
0.0
0.2
0.4
0.6
Evolutionary Distance
B
60
40
20
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

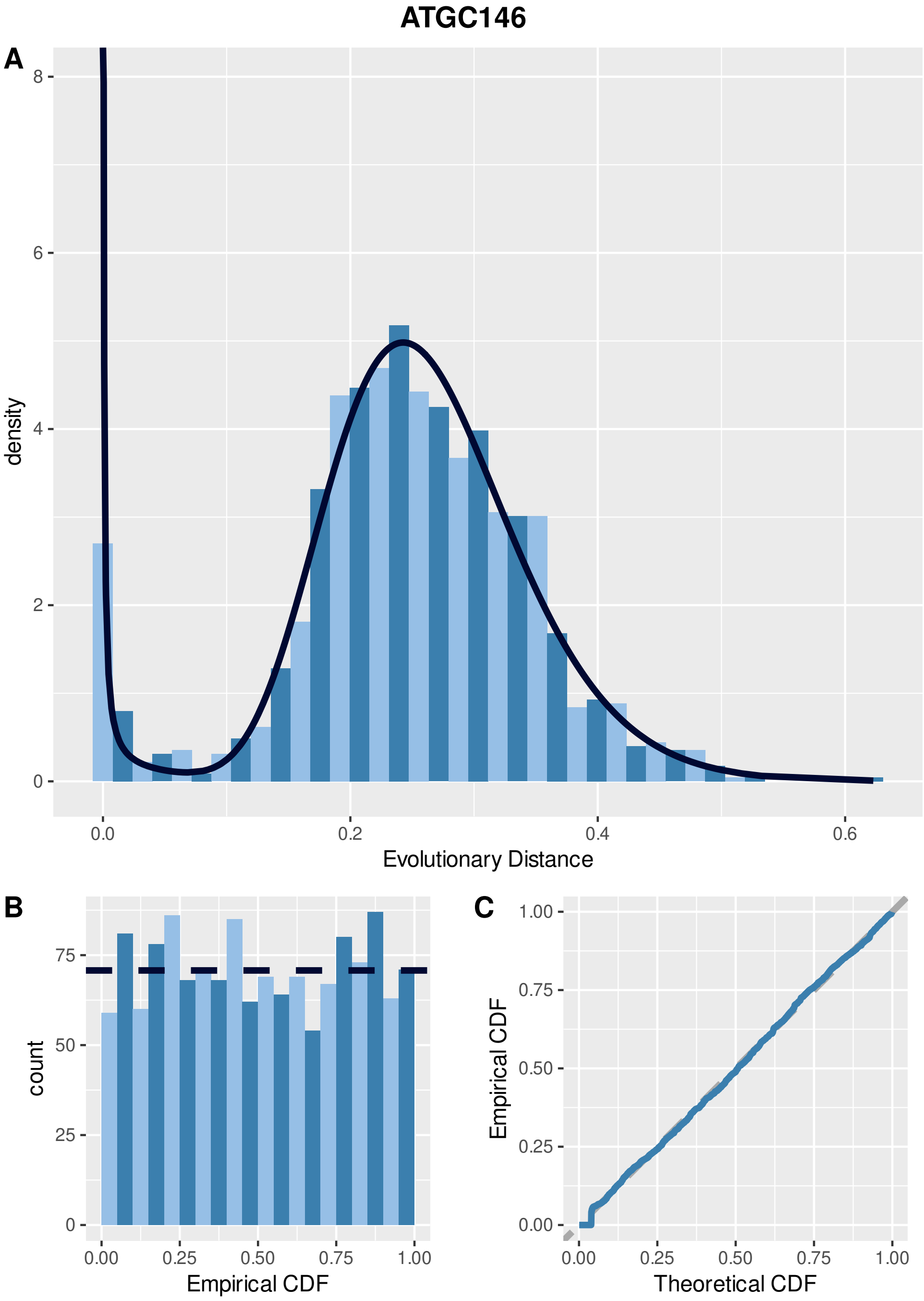

ATGC134
A
density
6
4
2
0
0.0
0.2
0.4
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

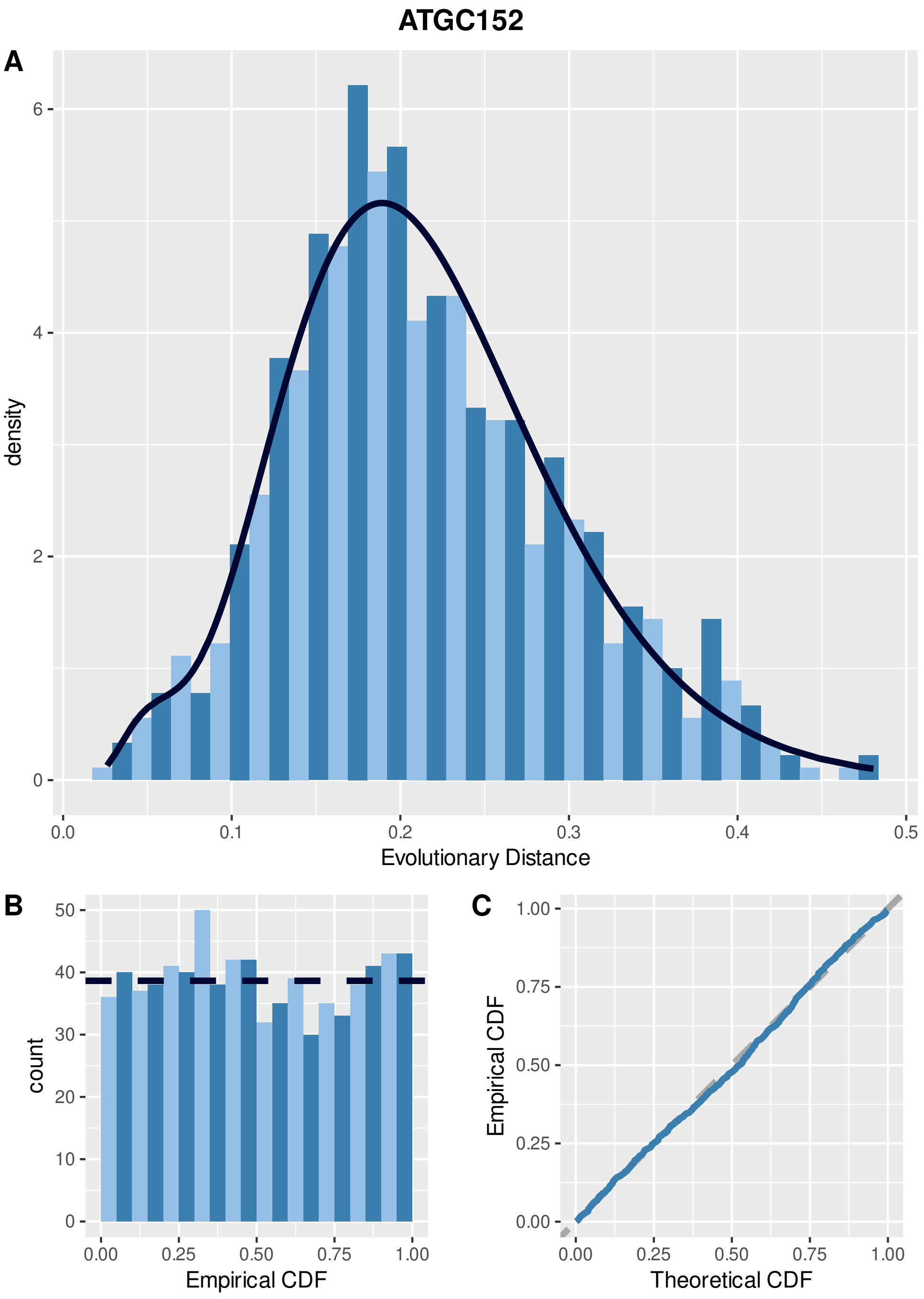

ATGC136
A
6
4
2
0
density
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
60
40
20
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

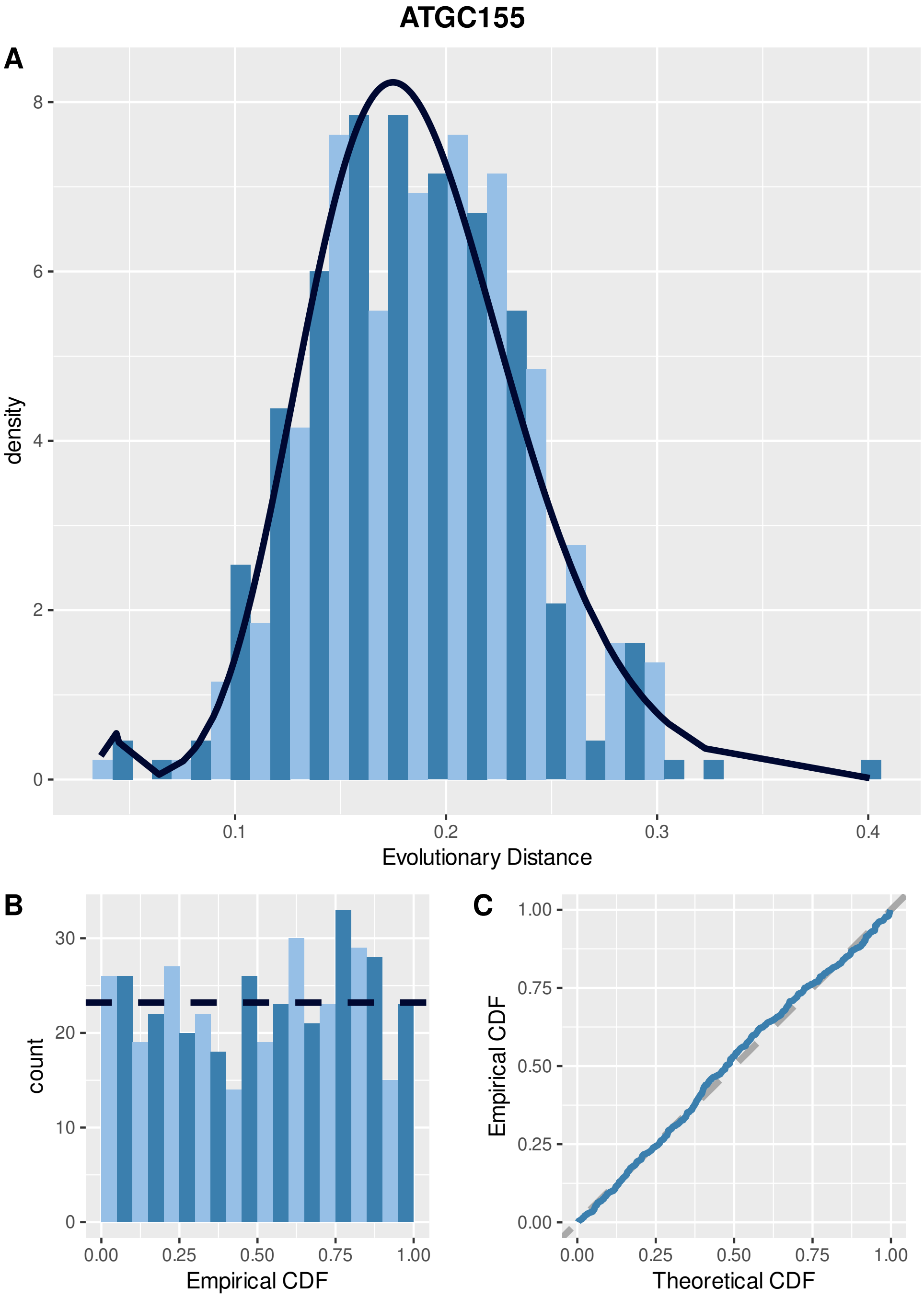

ATGC138
A
6
4
2
0
density
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
30
20
10
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

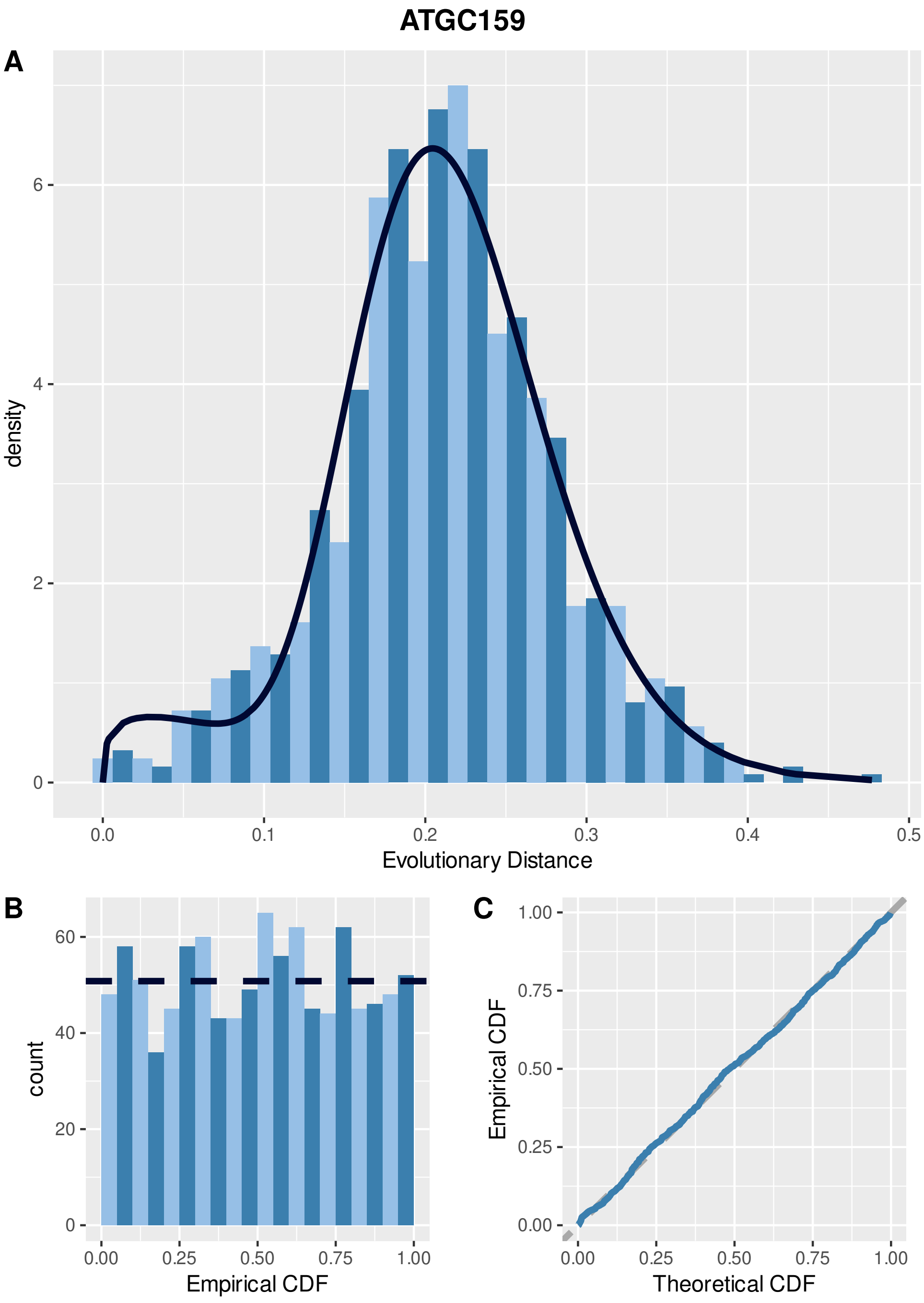

ATGC143
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

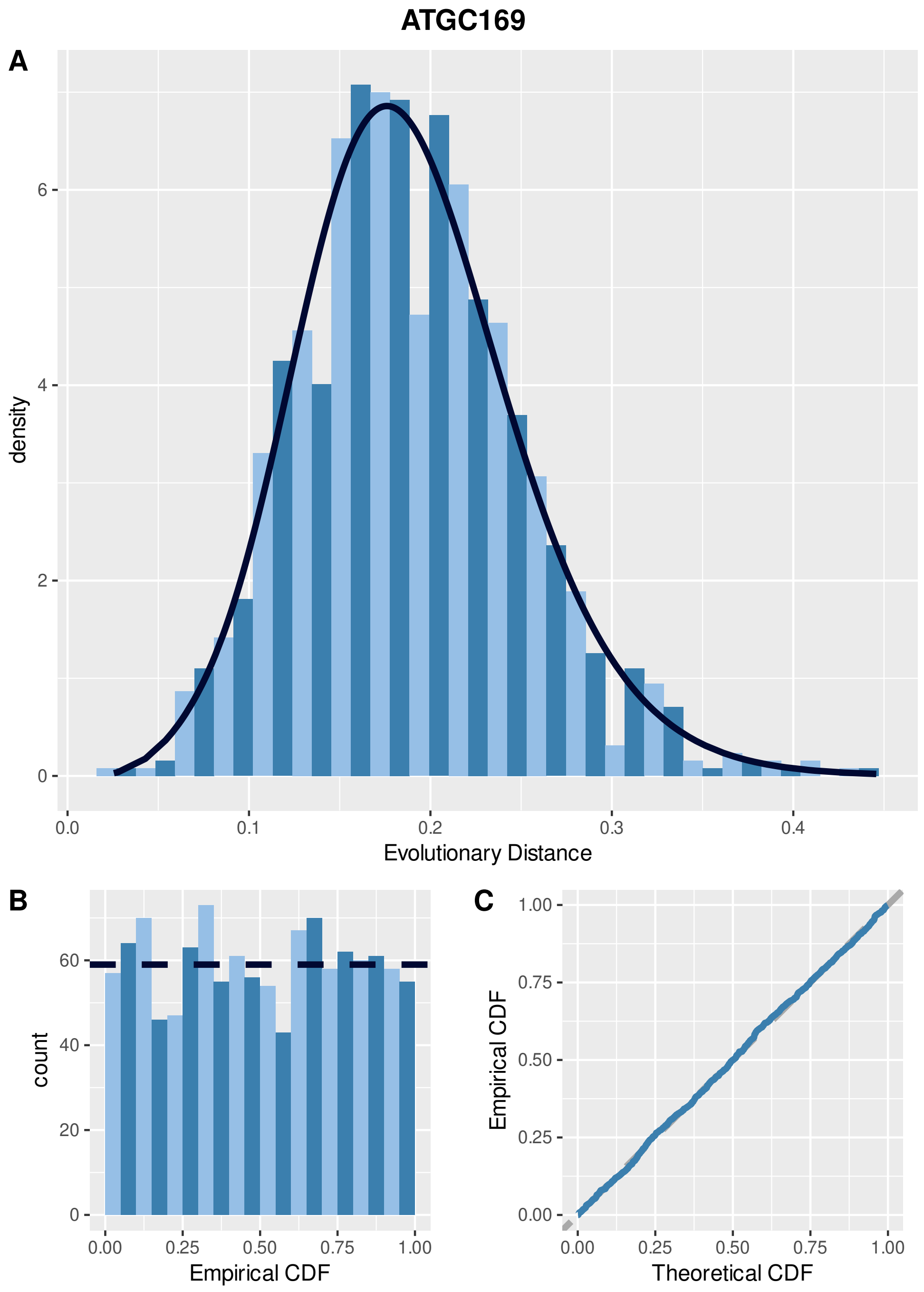

ATGC145
A
density
10.0
7.5
5.0
2.5
0.0
0.1
0.2
0.3
Evolutionary Distance
B
count
50
40
30
20
10
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

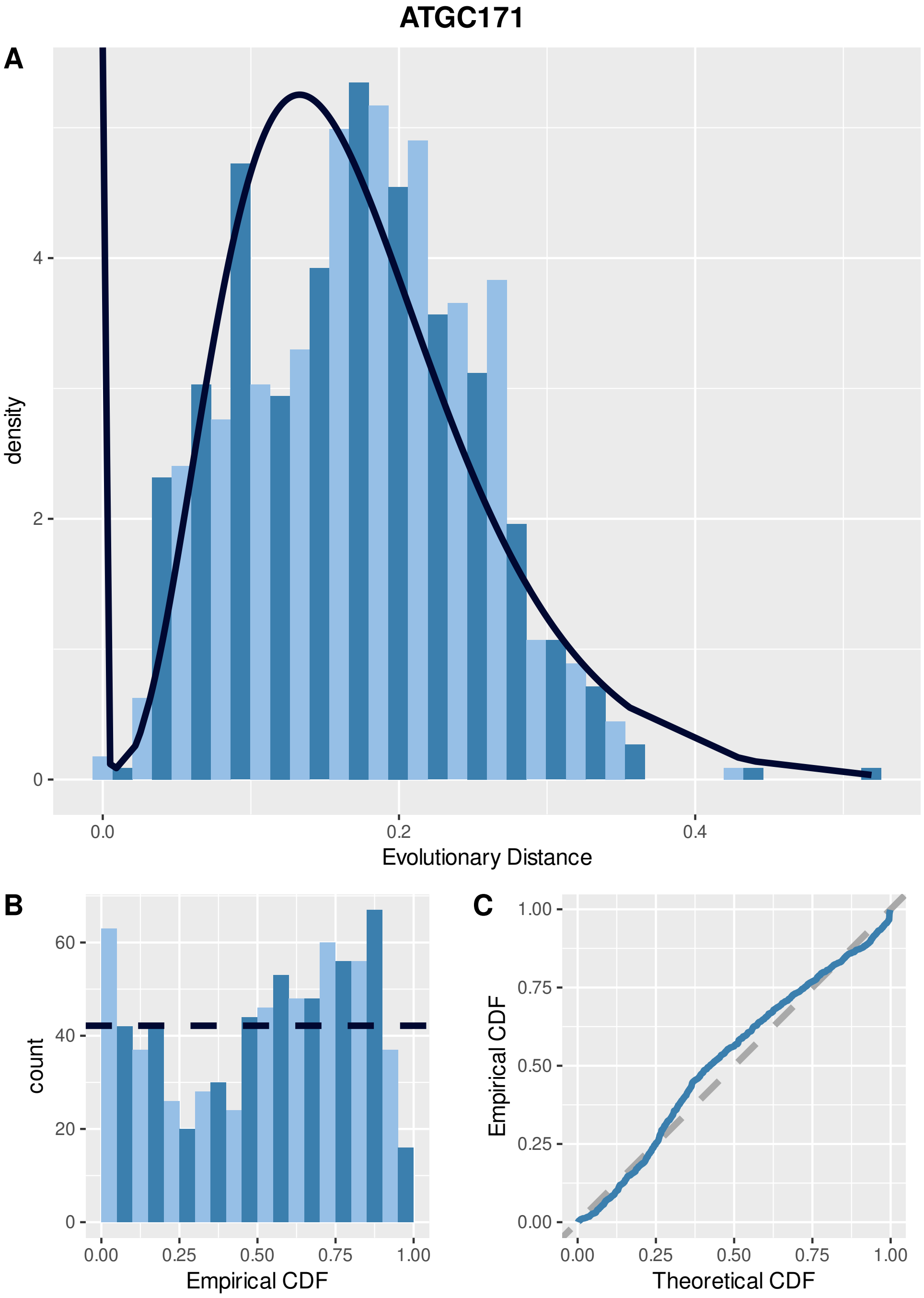

ATGC146
A
8
6
4
2
0
density
0.0
0.2
0.4
0.6
Evolutionary Distance
B
75
50
25
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

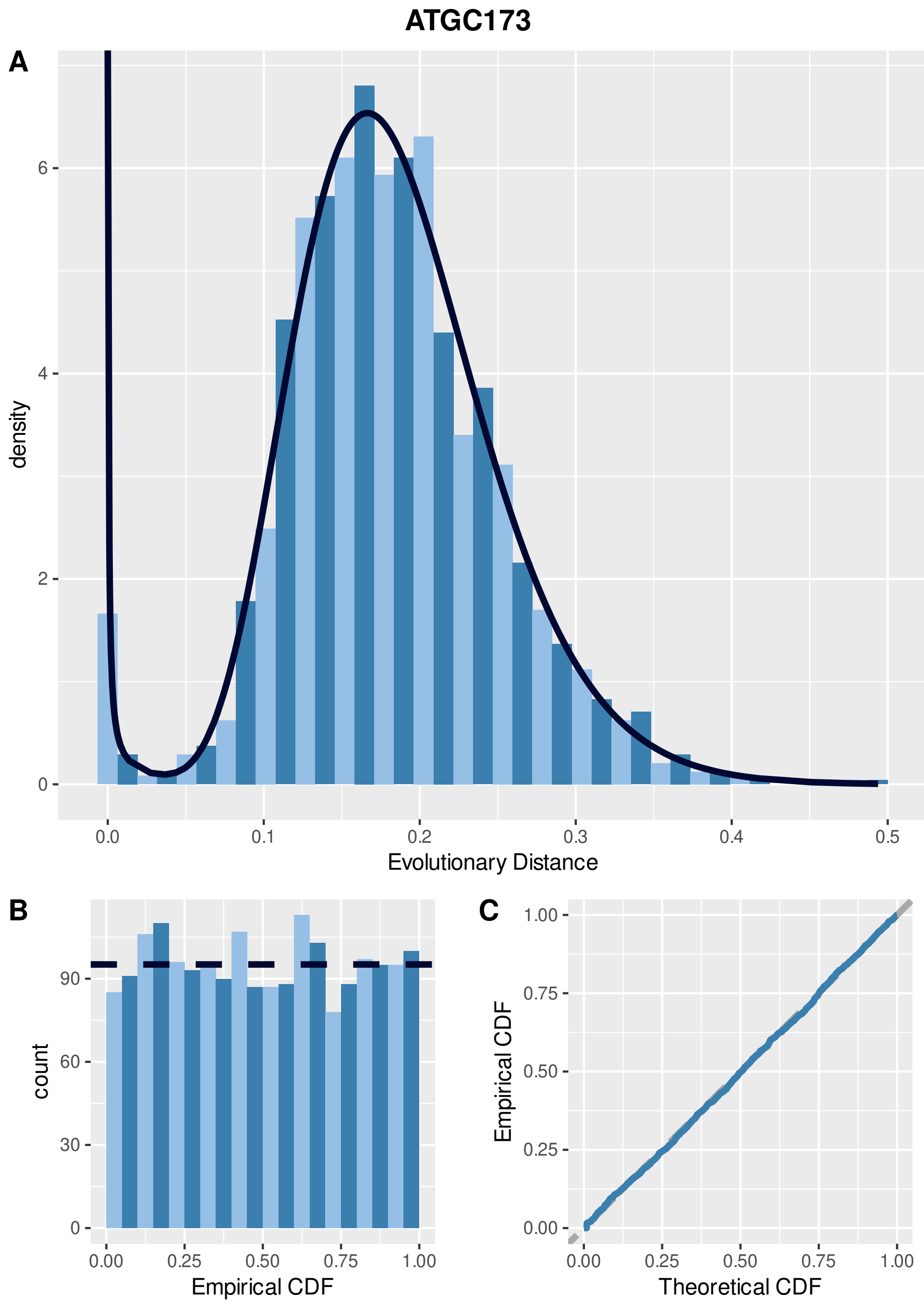

ATGC152
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
50
40
30
20
10
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

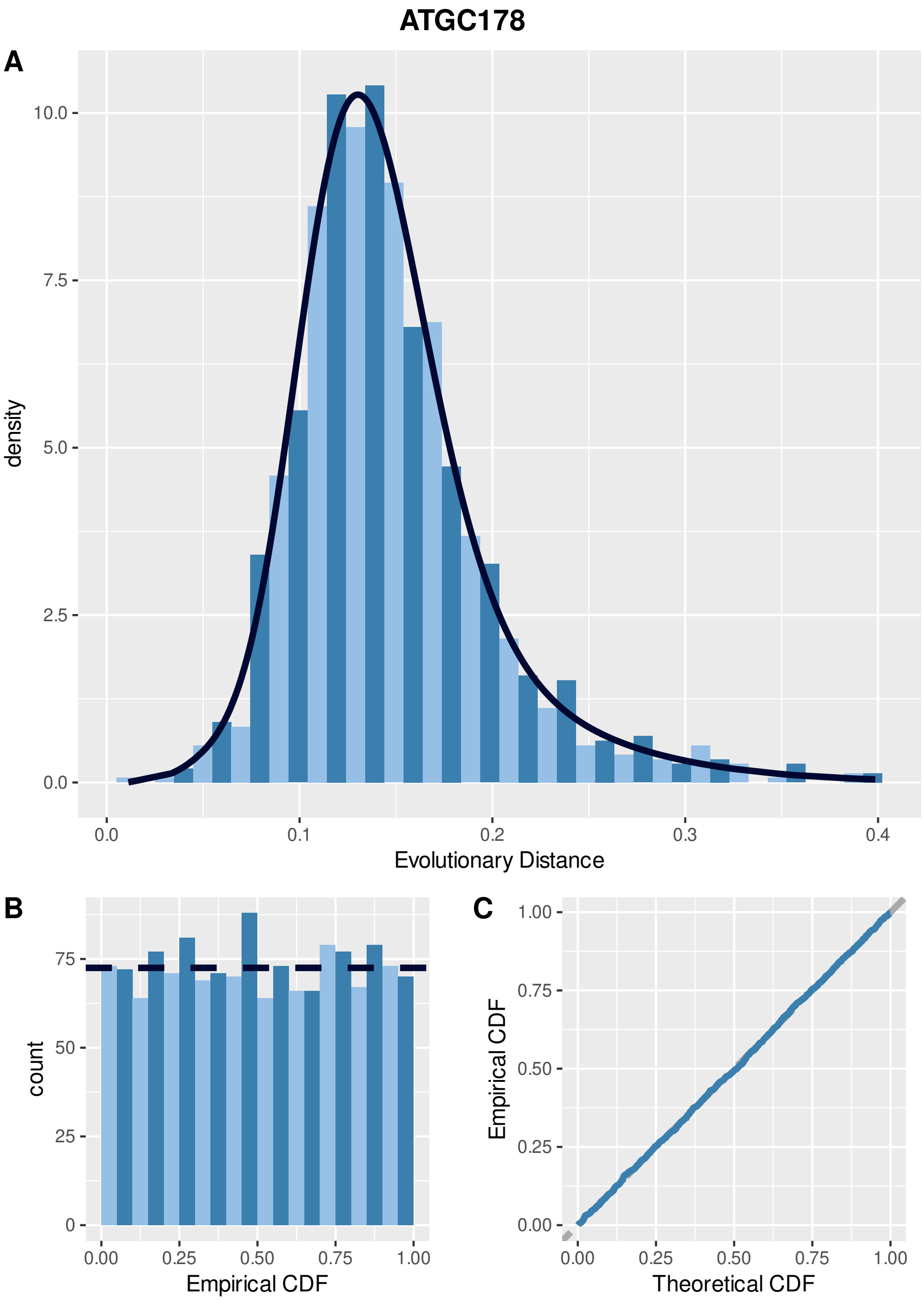

ATGC155
A
8
6
4
2
0
density
0.1
0.2
0.3
0.4
Evolutionary Distance
B
30
20
10
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

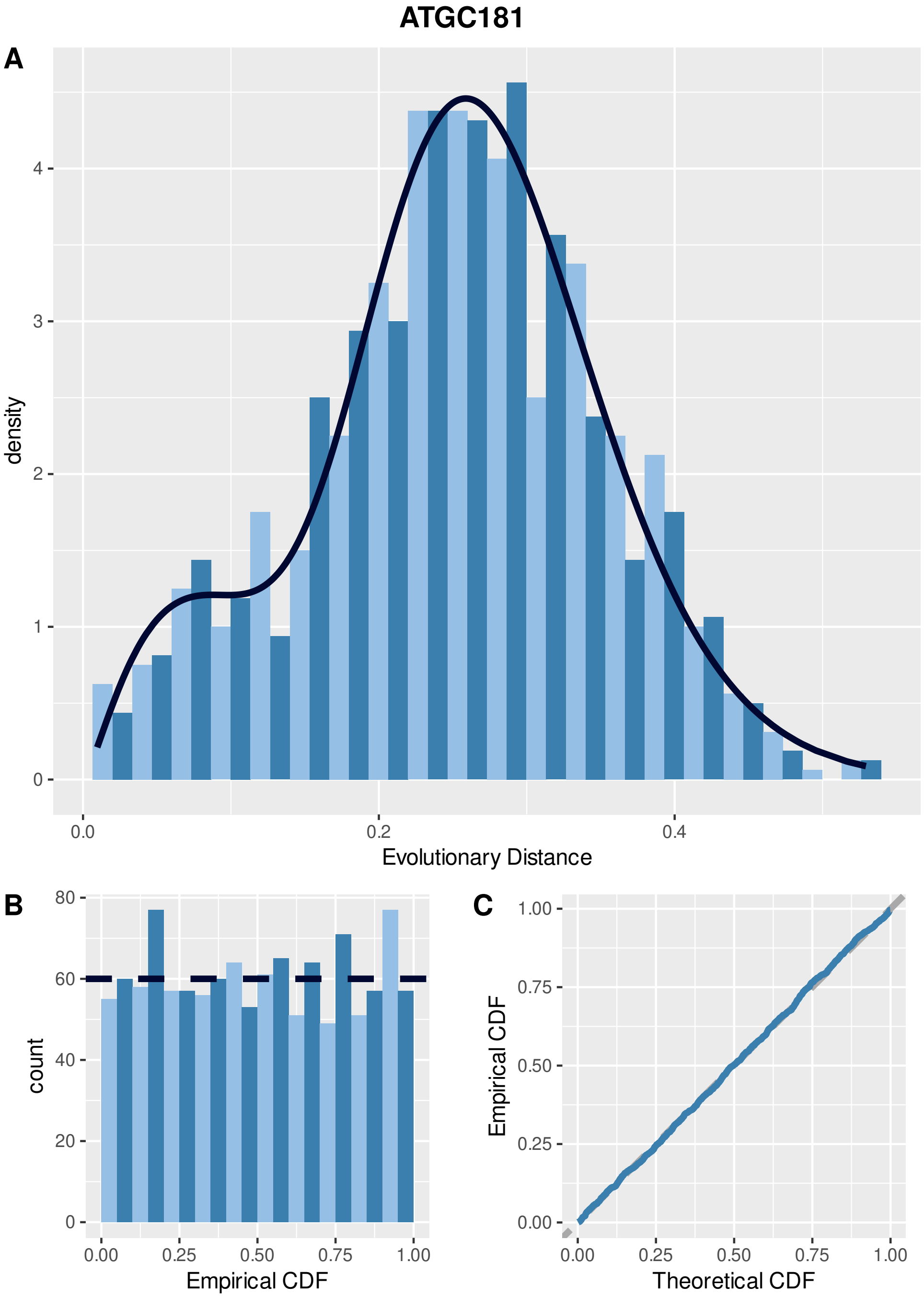

ATGC159
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

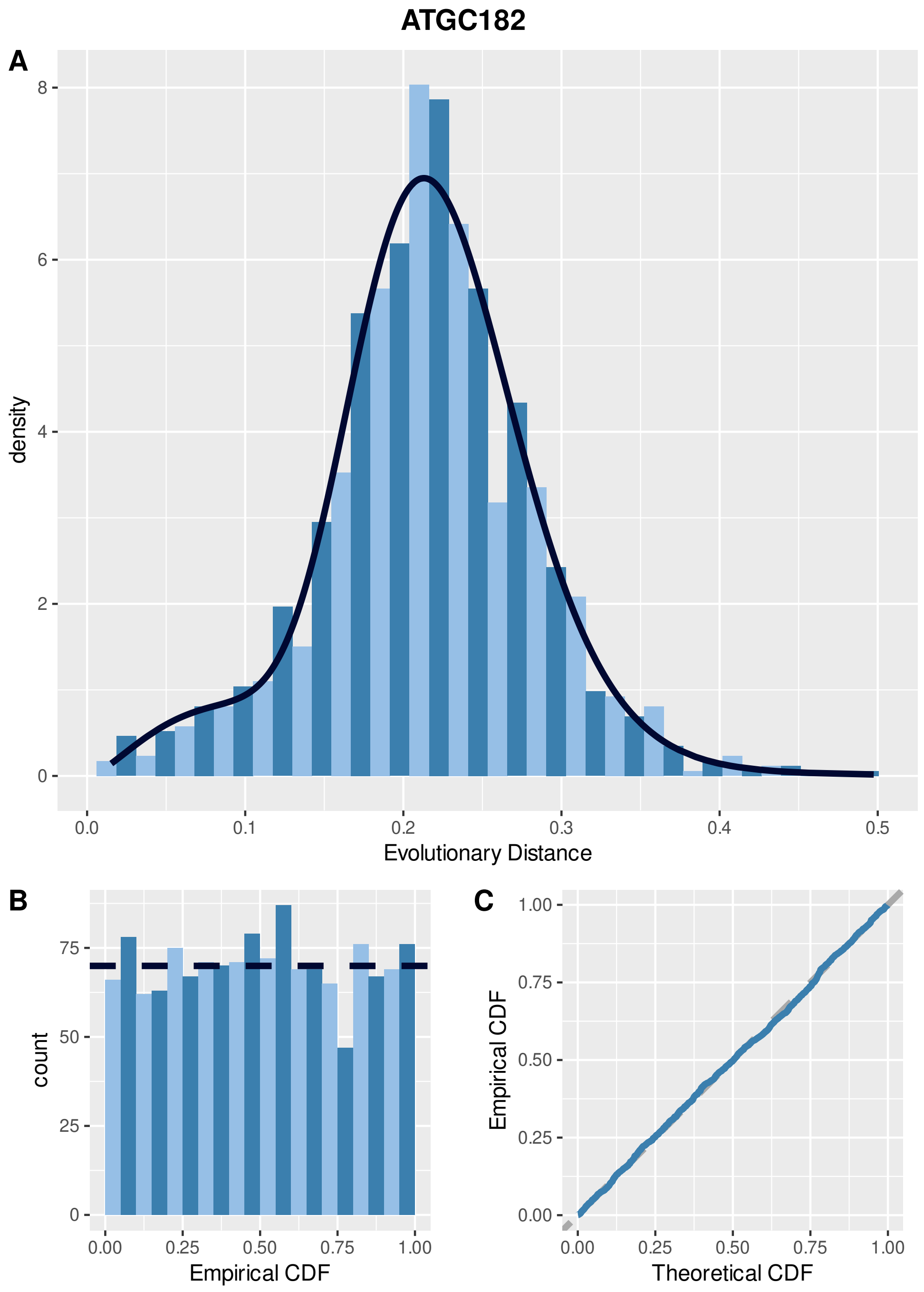

ATGC169
A
density
0
2
4
6
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
0
20
40
60
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

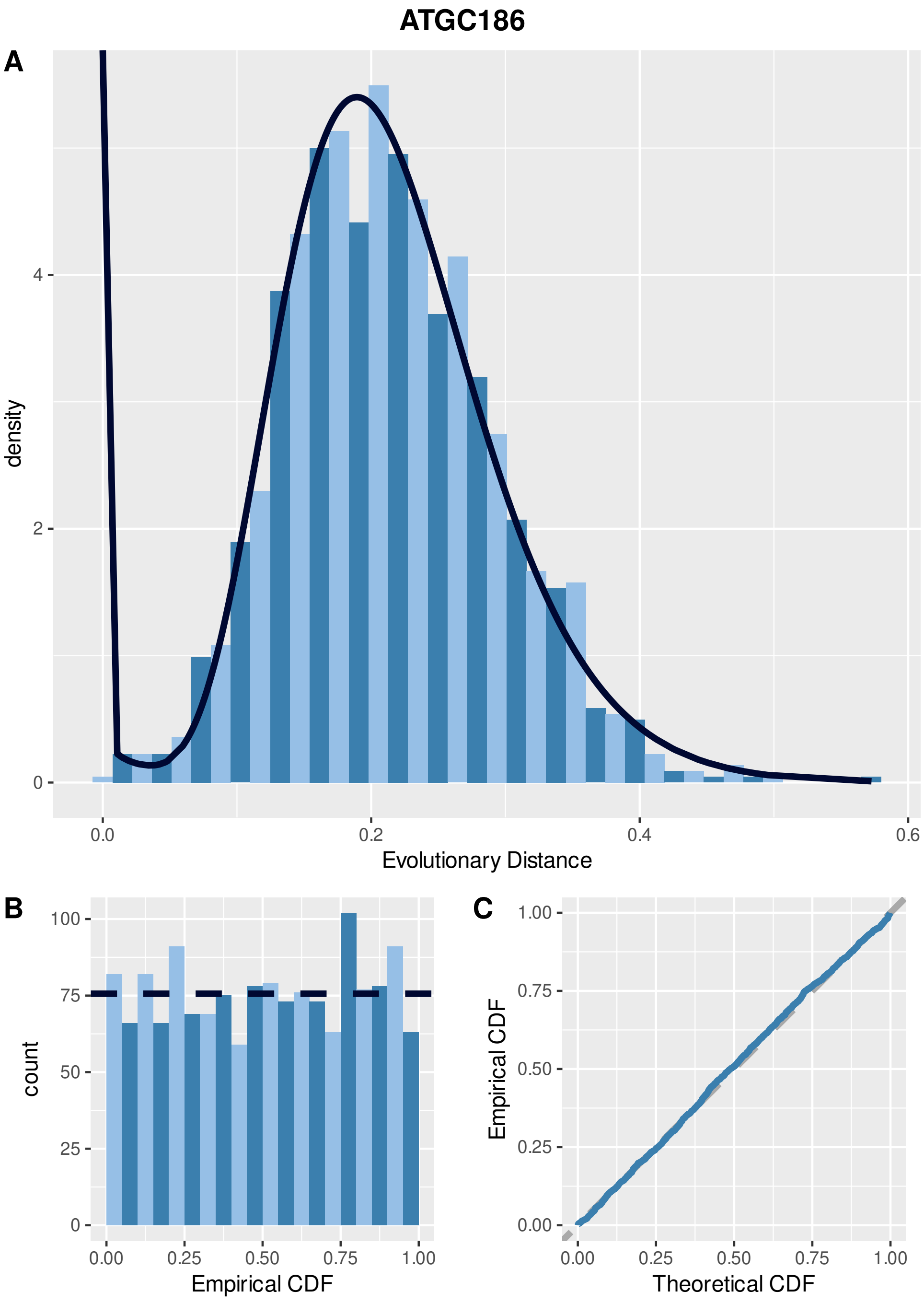

ATGC171
A
density
0
2
4
0.0
0.2
0.4
Evolutionary Distance
B
count
0
20
40
60
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

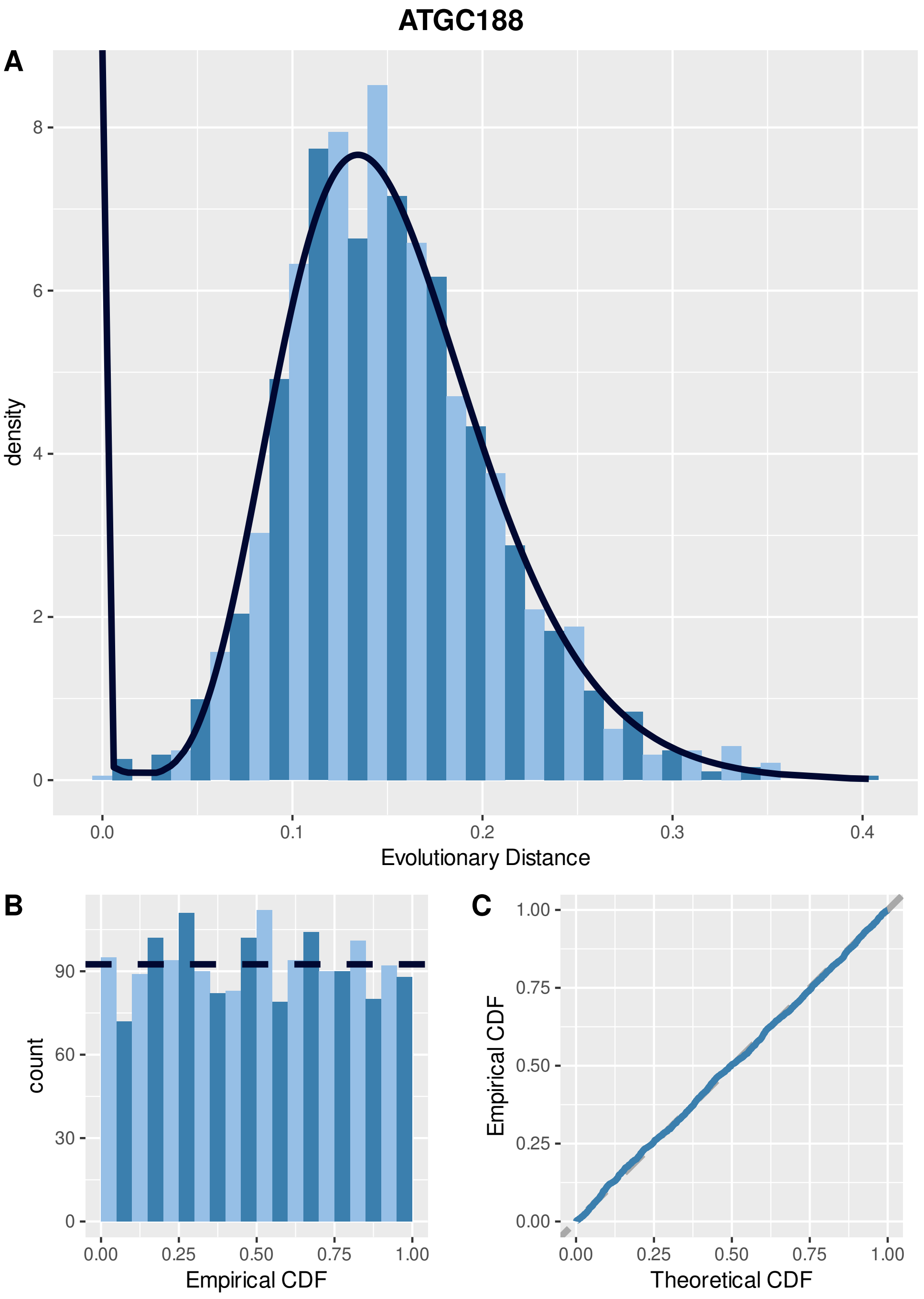

ATGC173
A
density
0
2
4
6
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
0
30
60
90
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

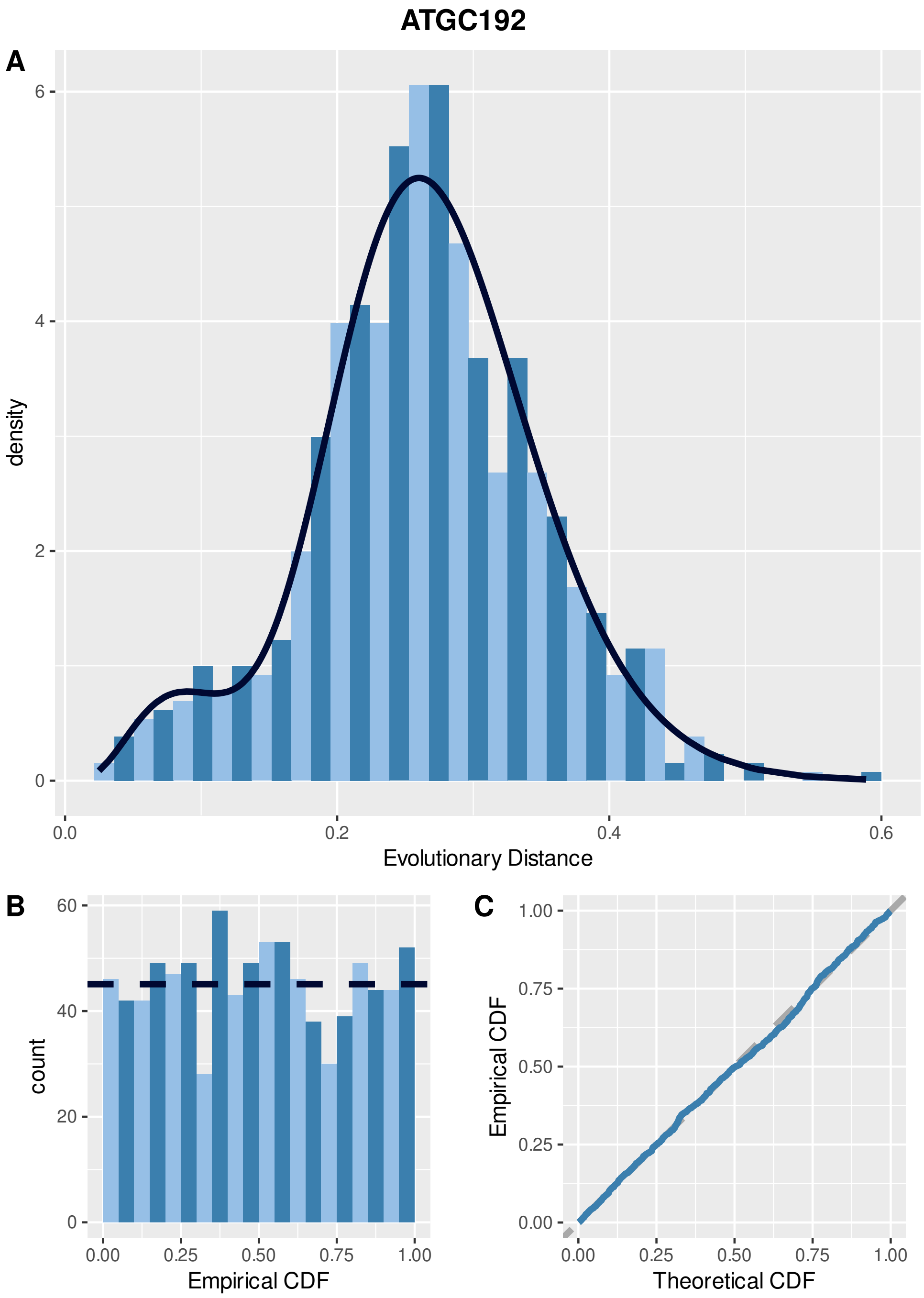

ATGC178
A
density
10.0
7.5
5.0
2.5
0.0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

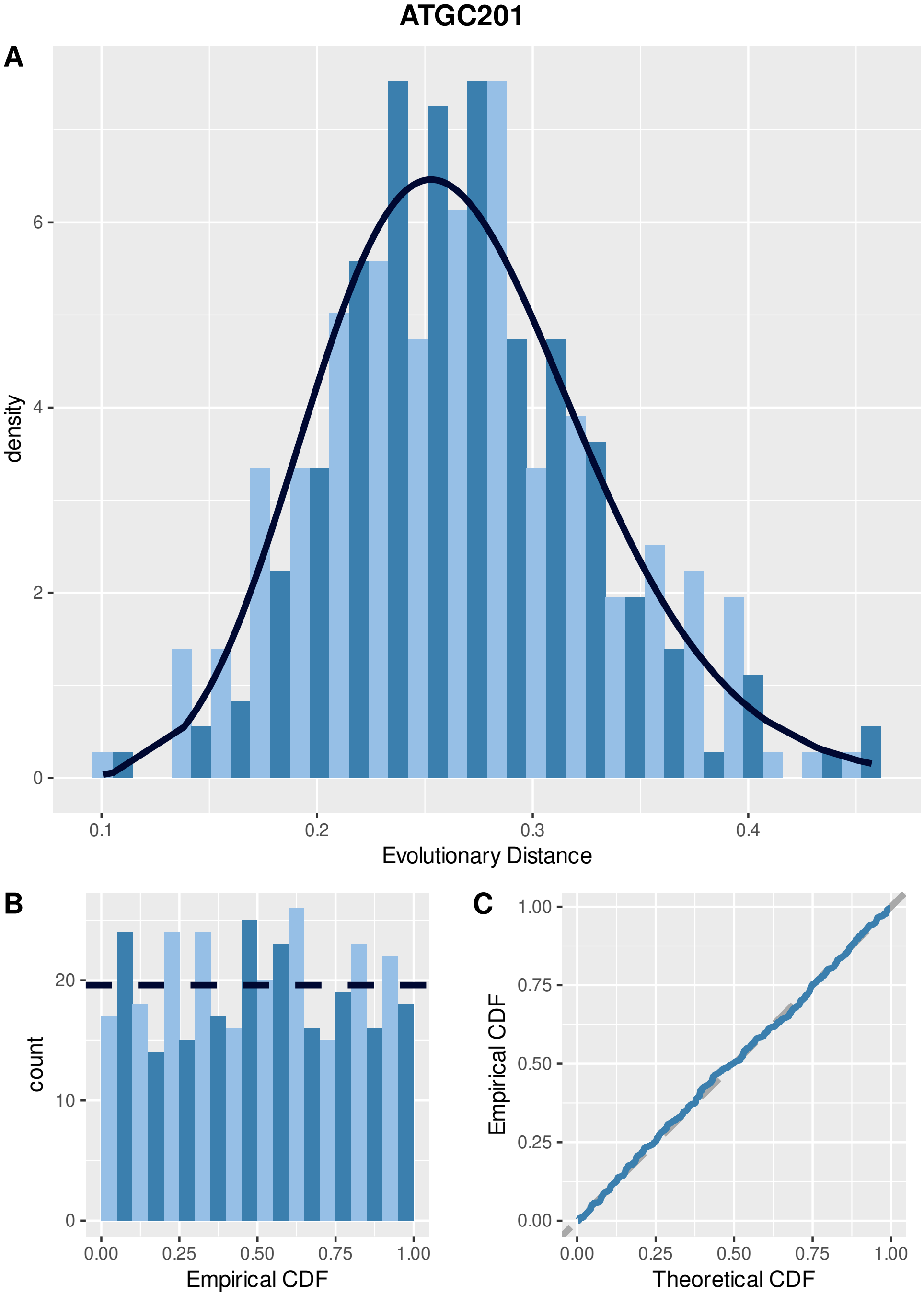

ATGC181
A
density
4
3
2
1
0
0.0
0.2
0.4
Evolutionary Distance
B
count
80
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

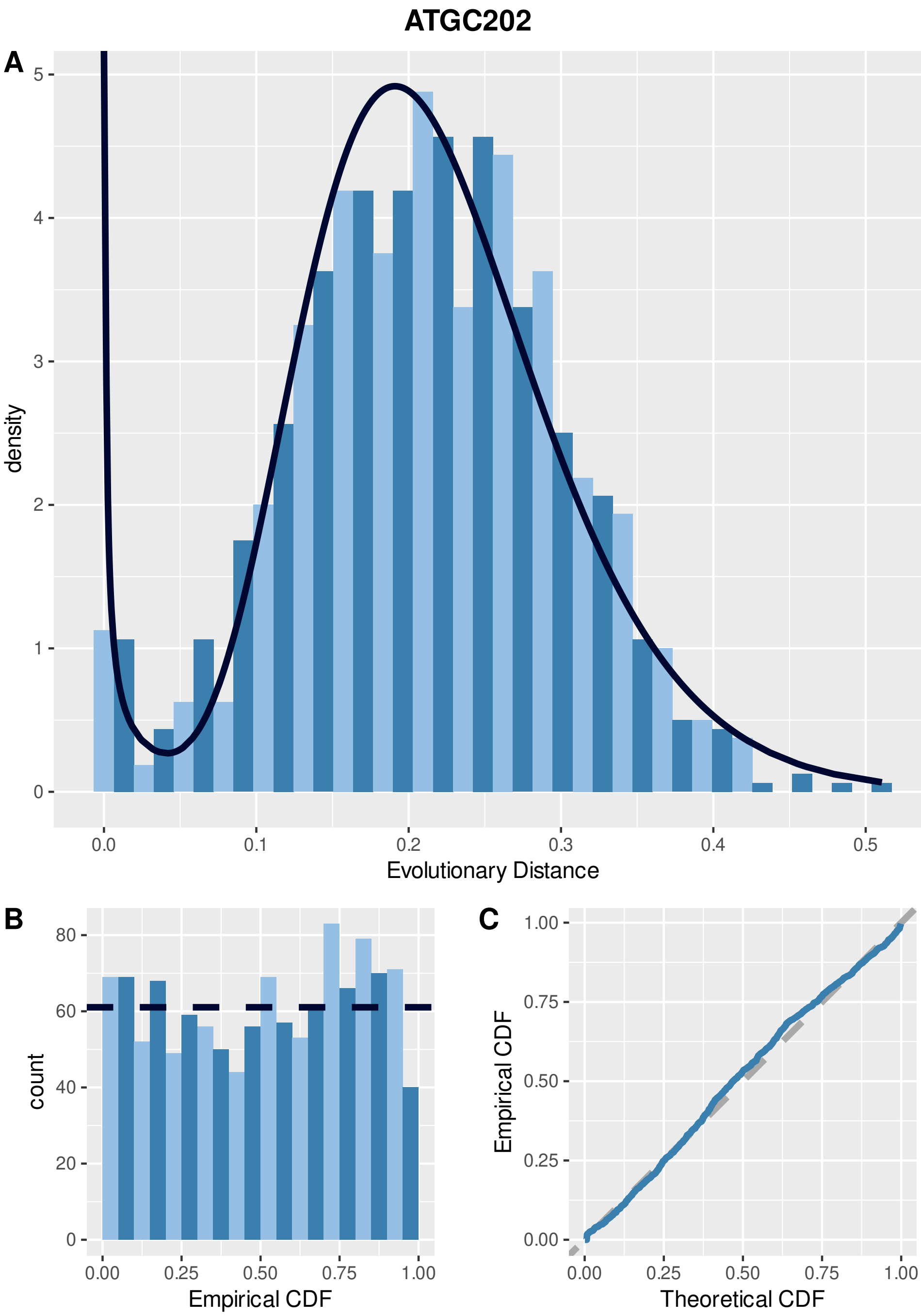

ATGC182
A
density
8
6
4
2
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

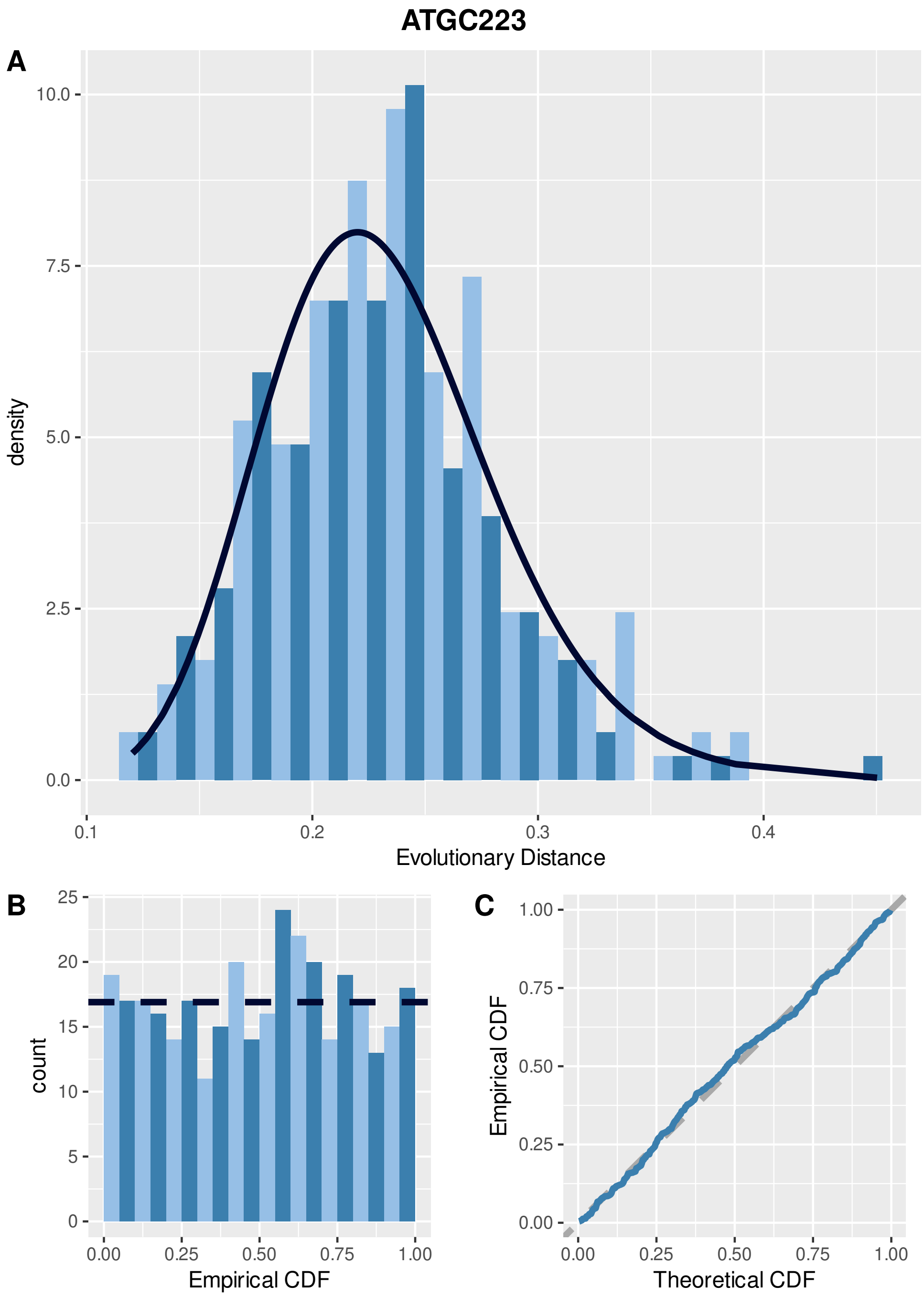

ATGC186
A
density
0
2
4
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
0
25
50
75
100
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

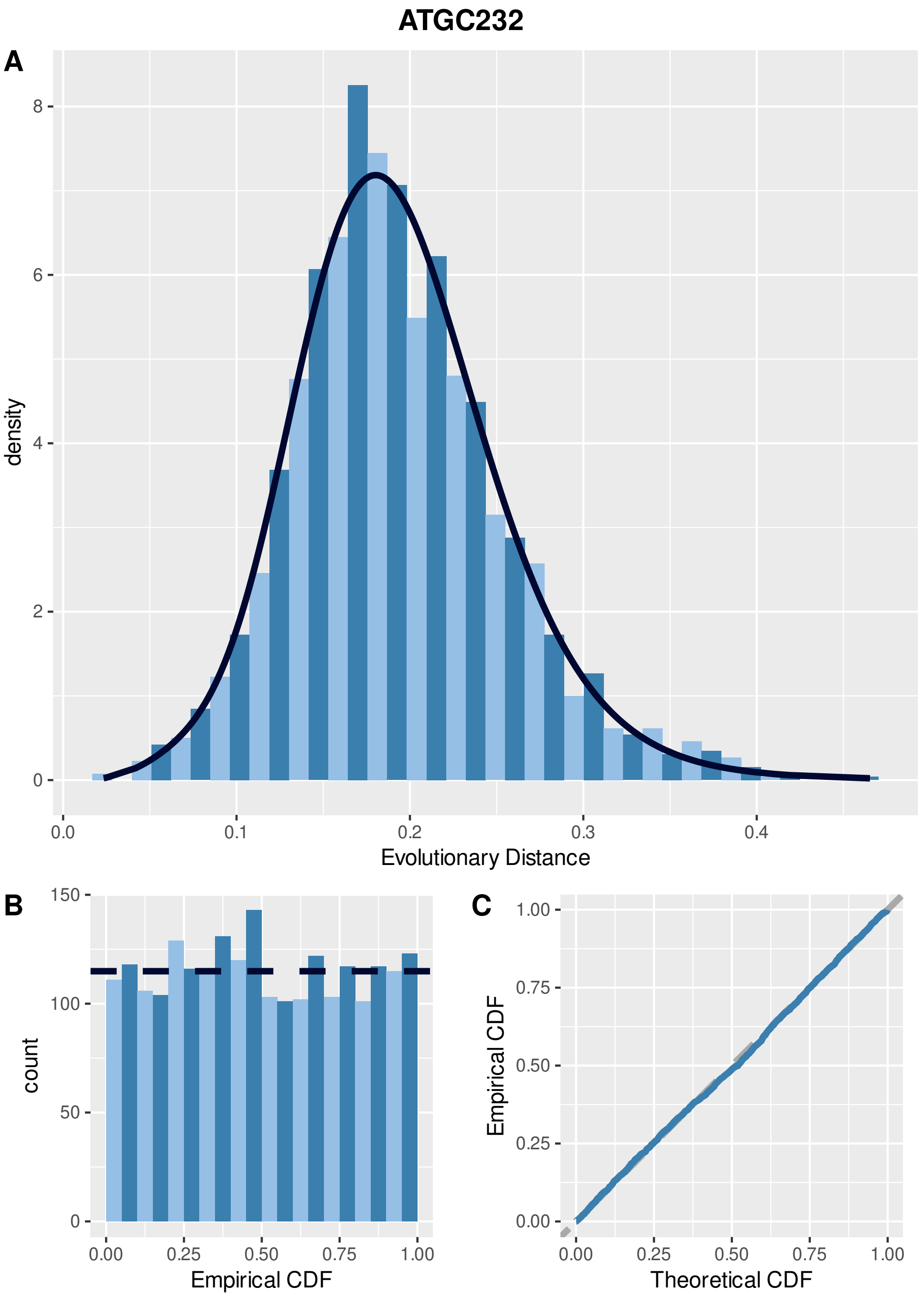

ATGC188
A
density
0
2
4
6
8
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
0
30
60
90
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

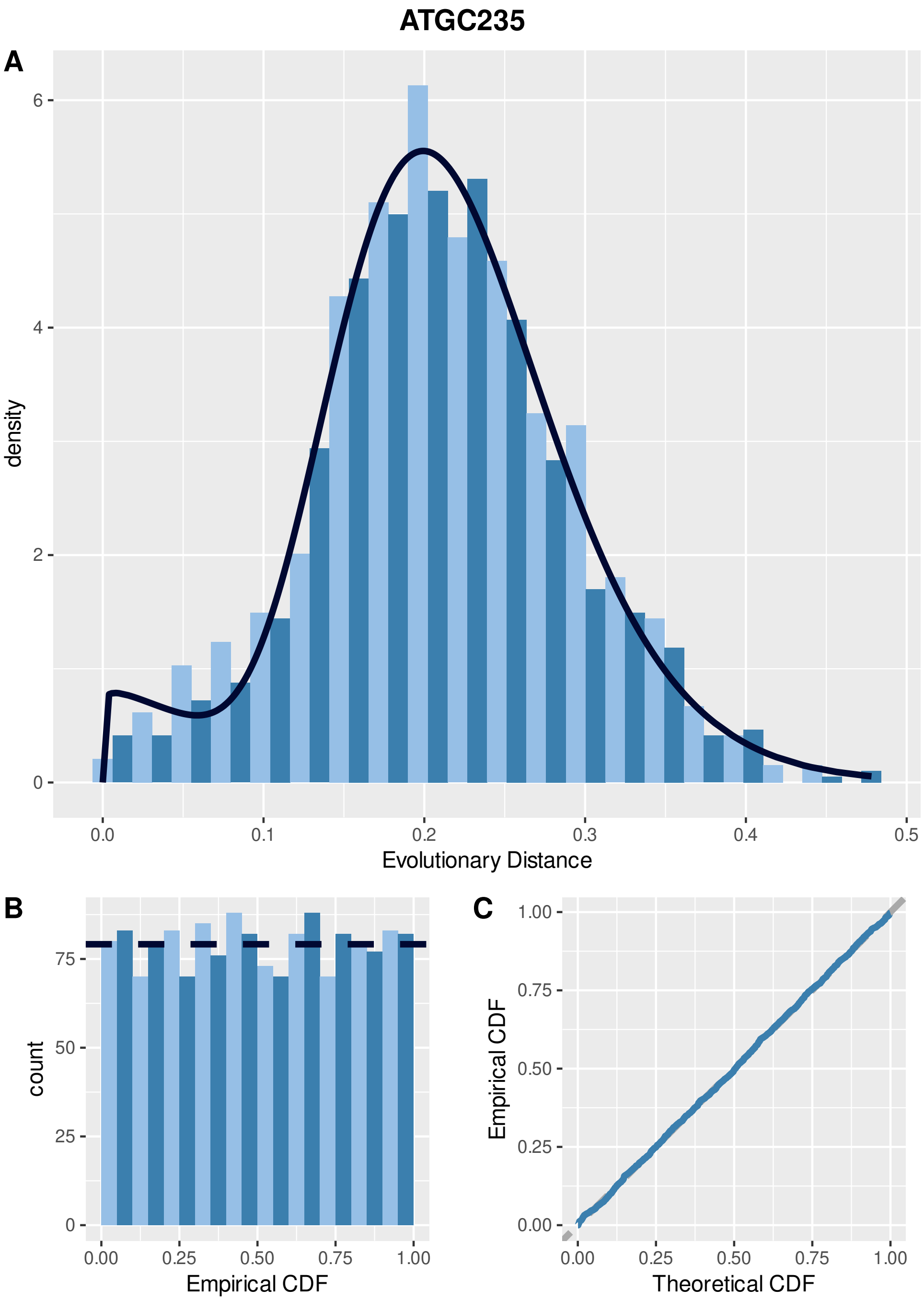

ATGC192
A
density
6
4
2
0
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

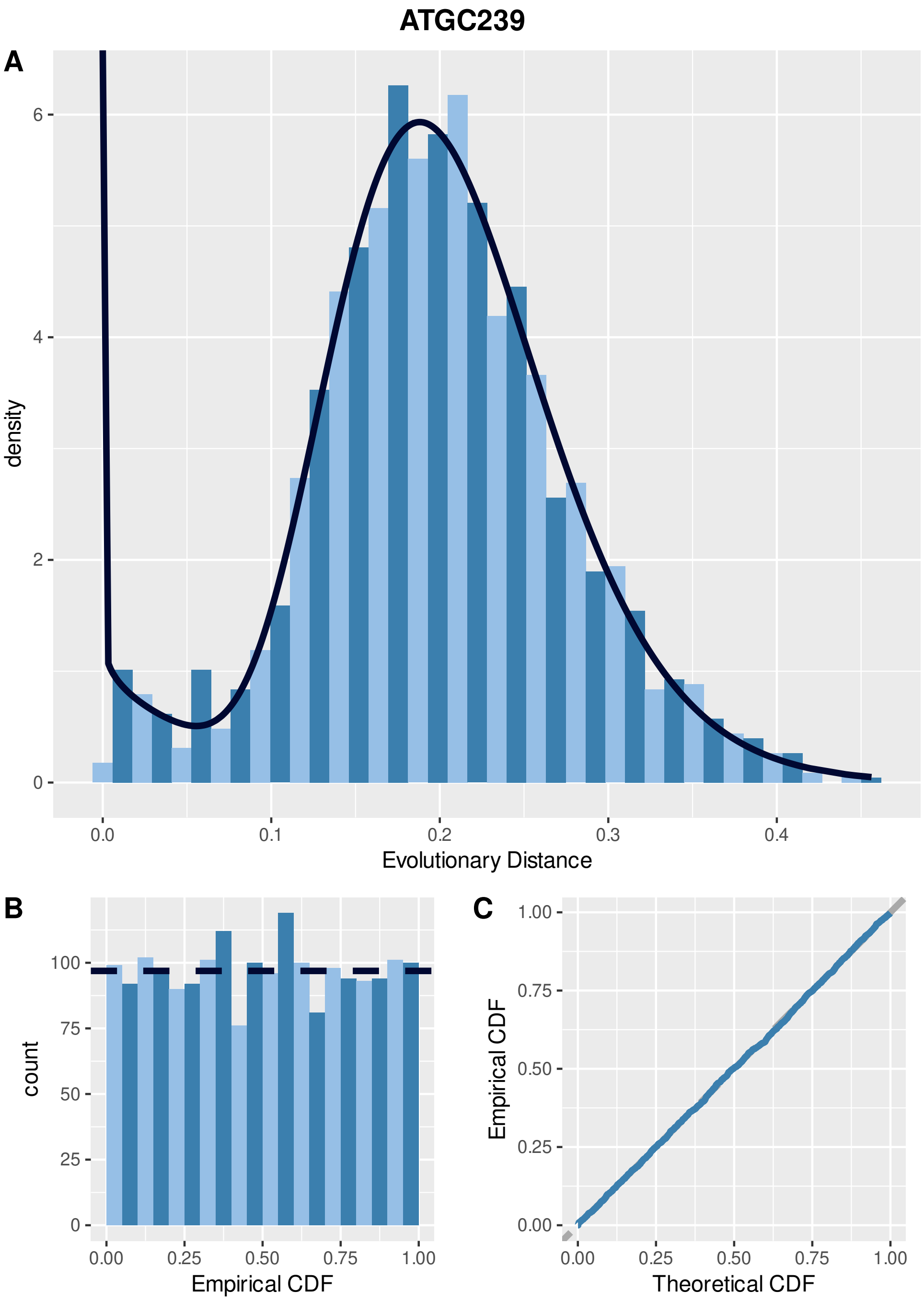

ATGC201
A
density
0
2
4
6
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
0
10
20
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

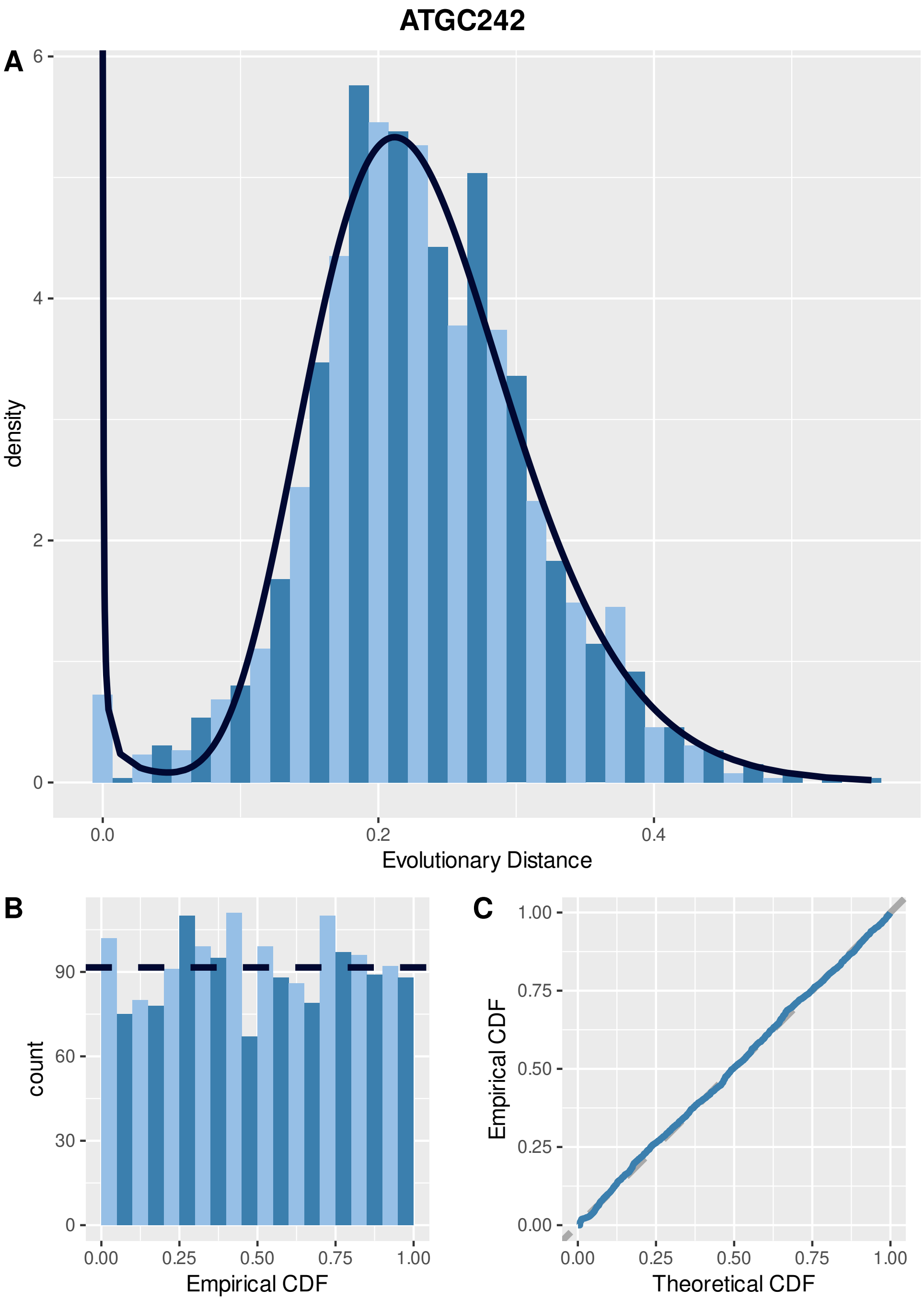

ATGC202
A
density
5
4
3
2
1
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
80
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC223
A
density
10.0
7.5
5.0
2.5
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
25
20
15
10
5
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

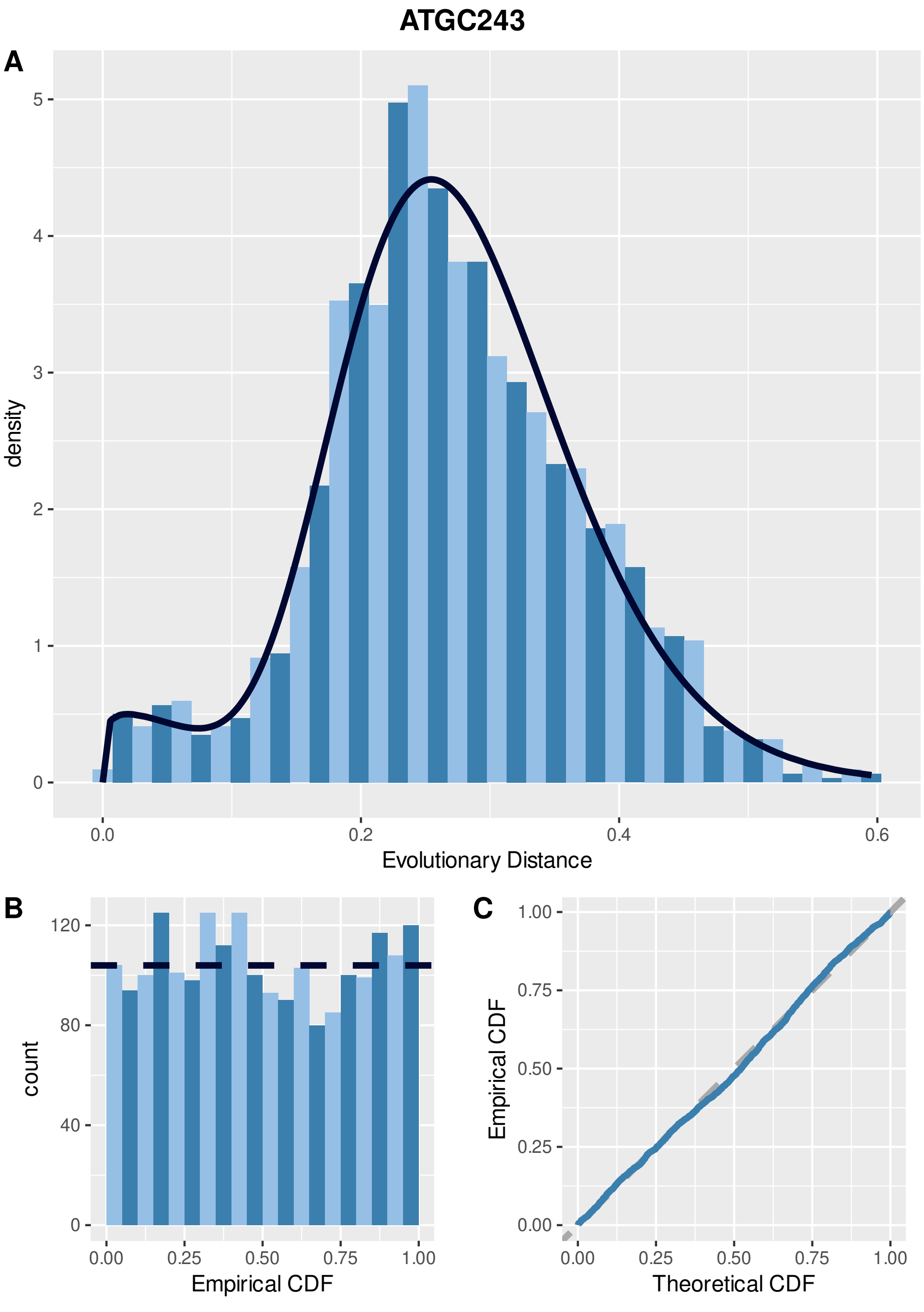

ATGC232
A
density
8
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
150
100
50
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

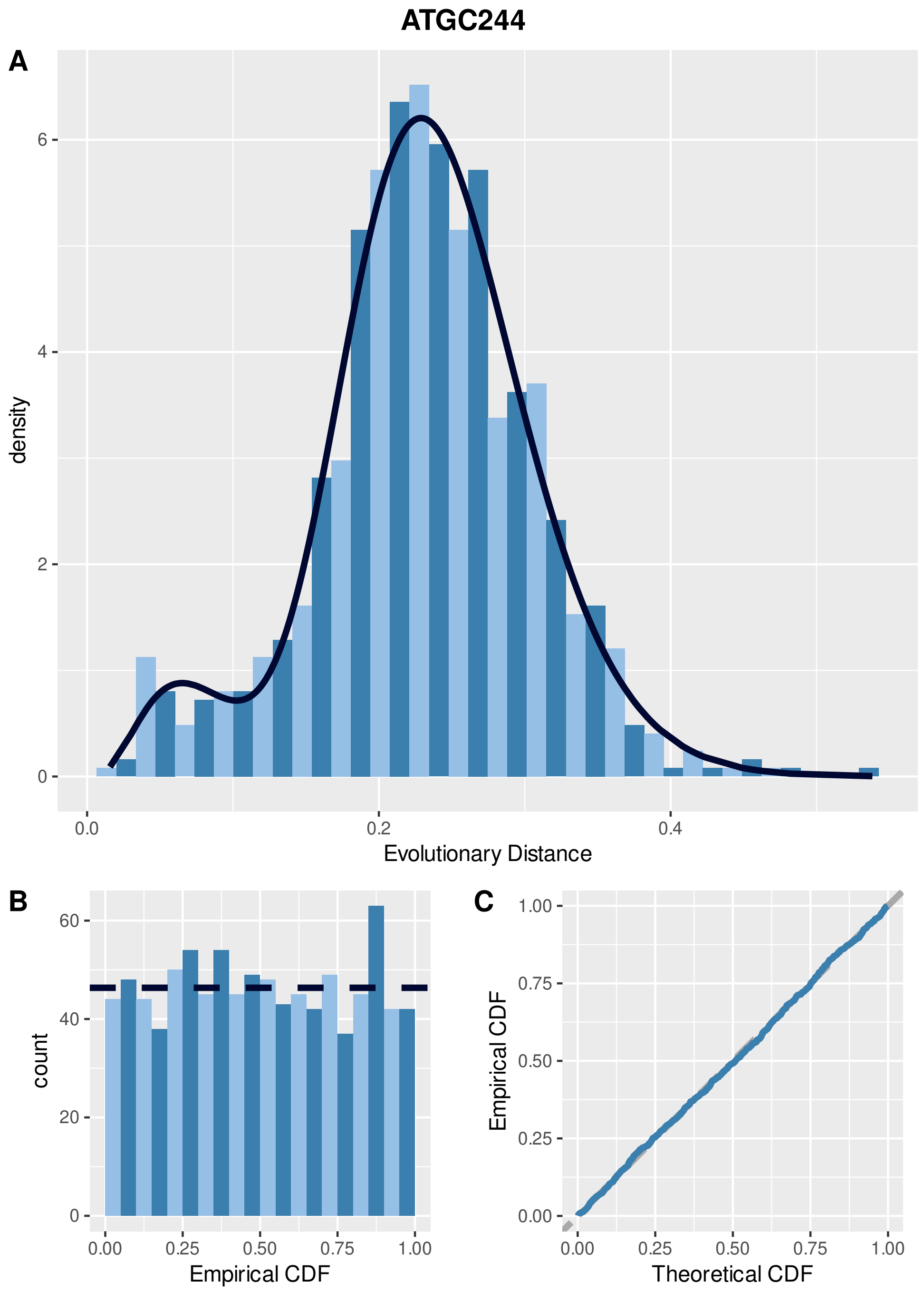

ATGC235
A
6
4
2
0
density
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
75
50
25
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

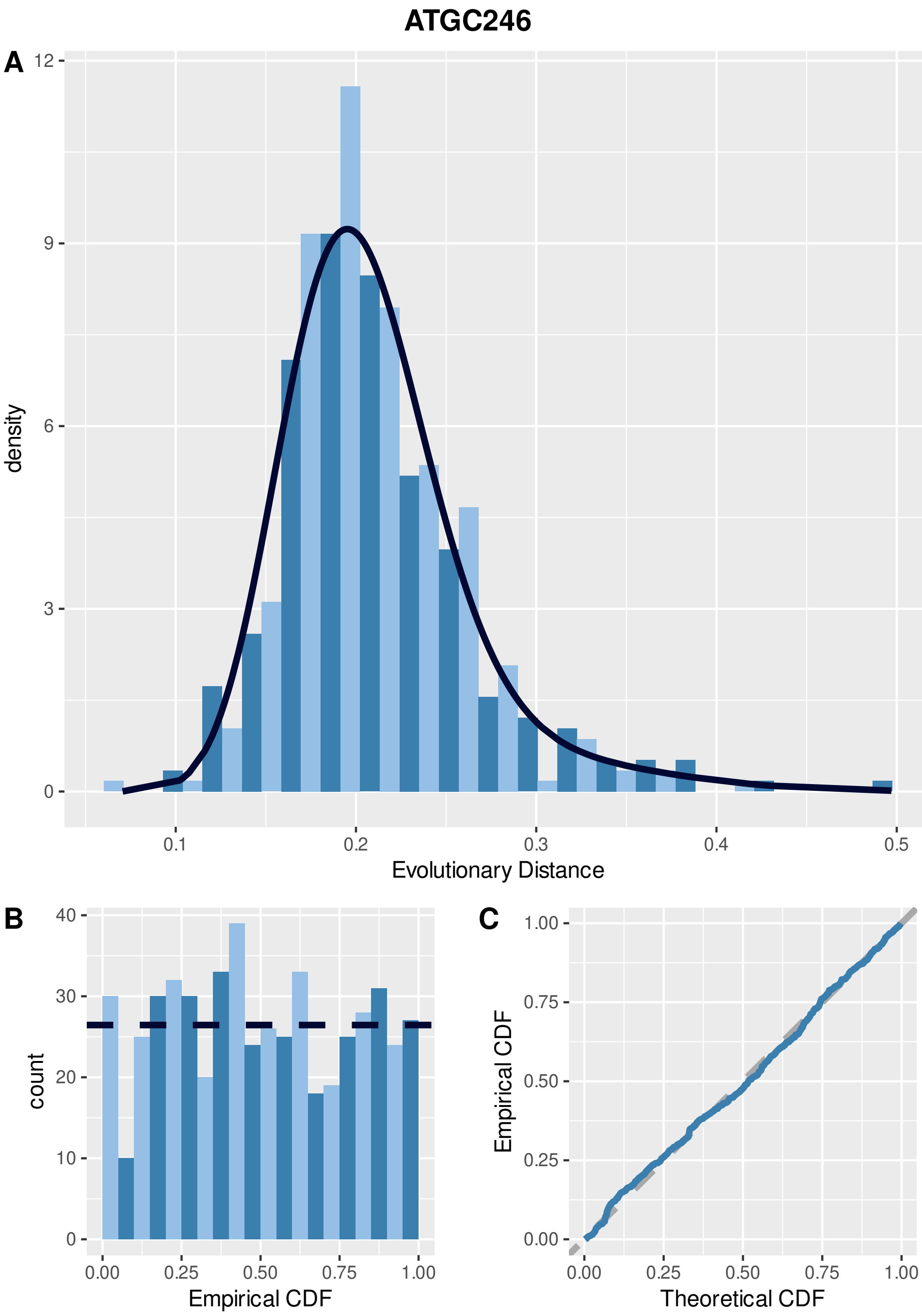

ATGC239
A
6
4
2
0
density
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
100
75
50
25
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

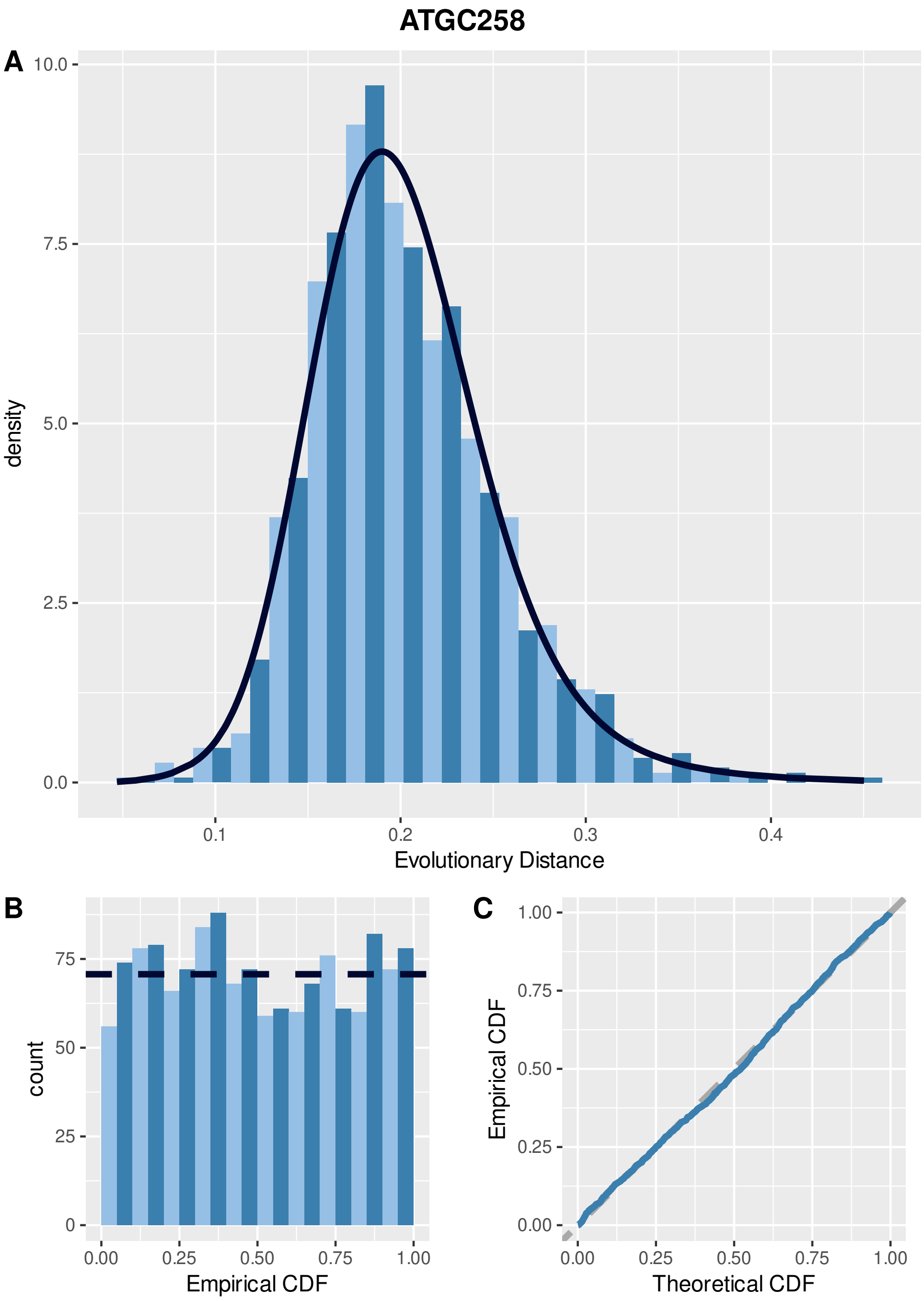


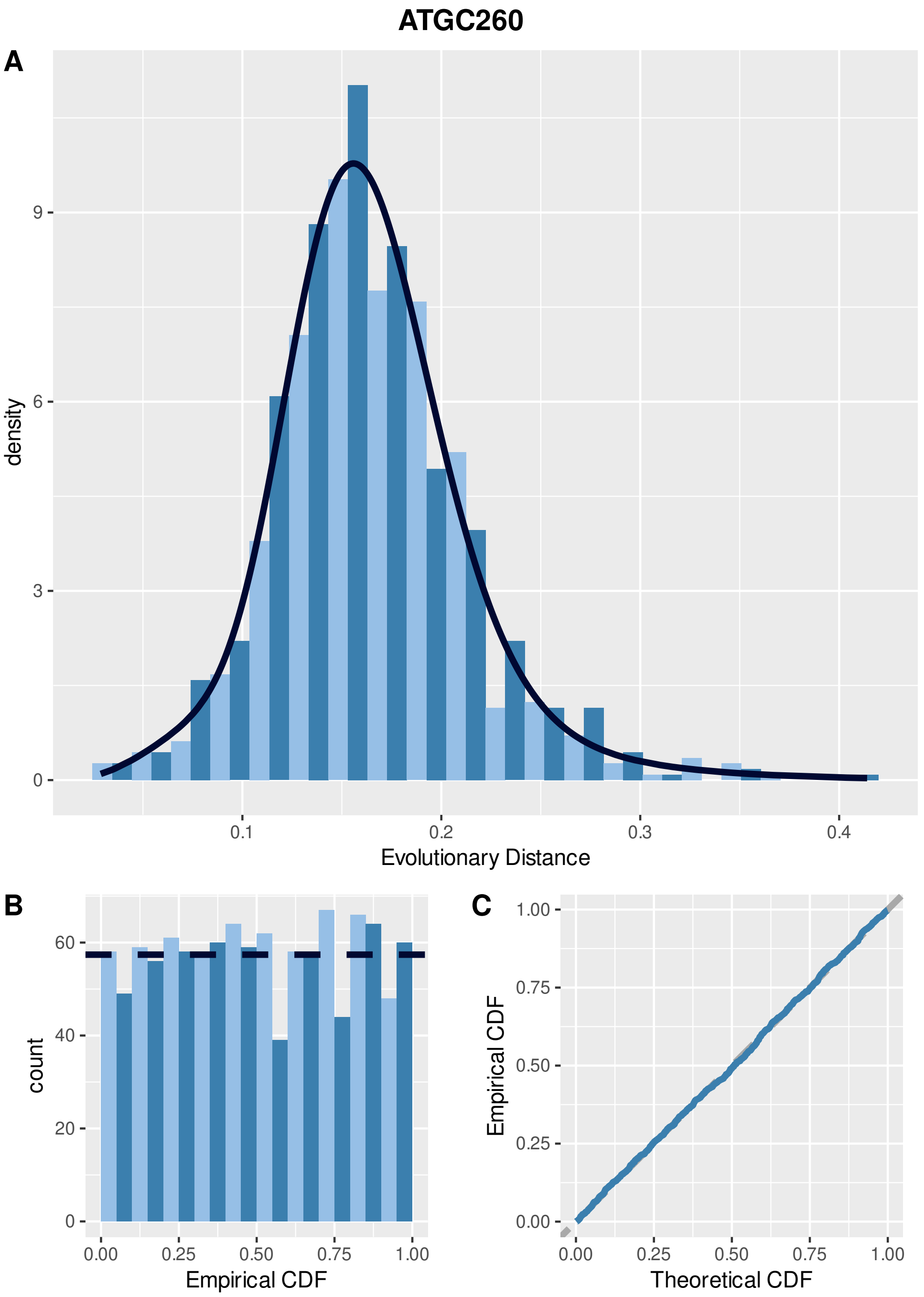
ATGC242
A
6
4
2
0
density
0.0
0.2
0.4
Evolutionary Distance
B
90
60
30
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

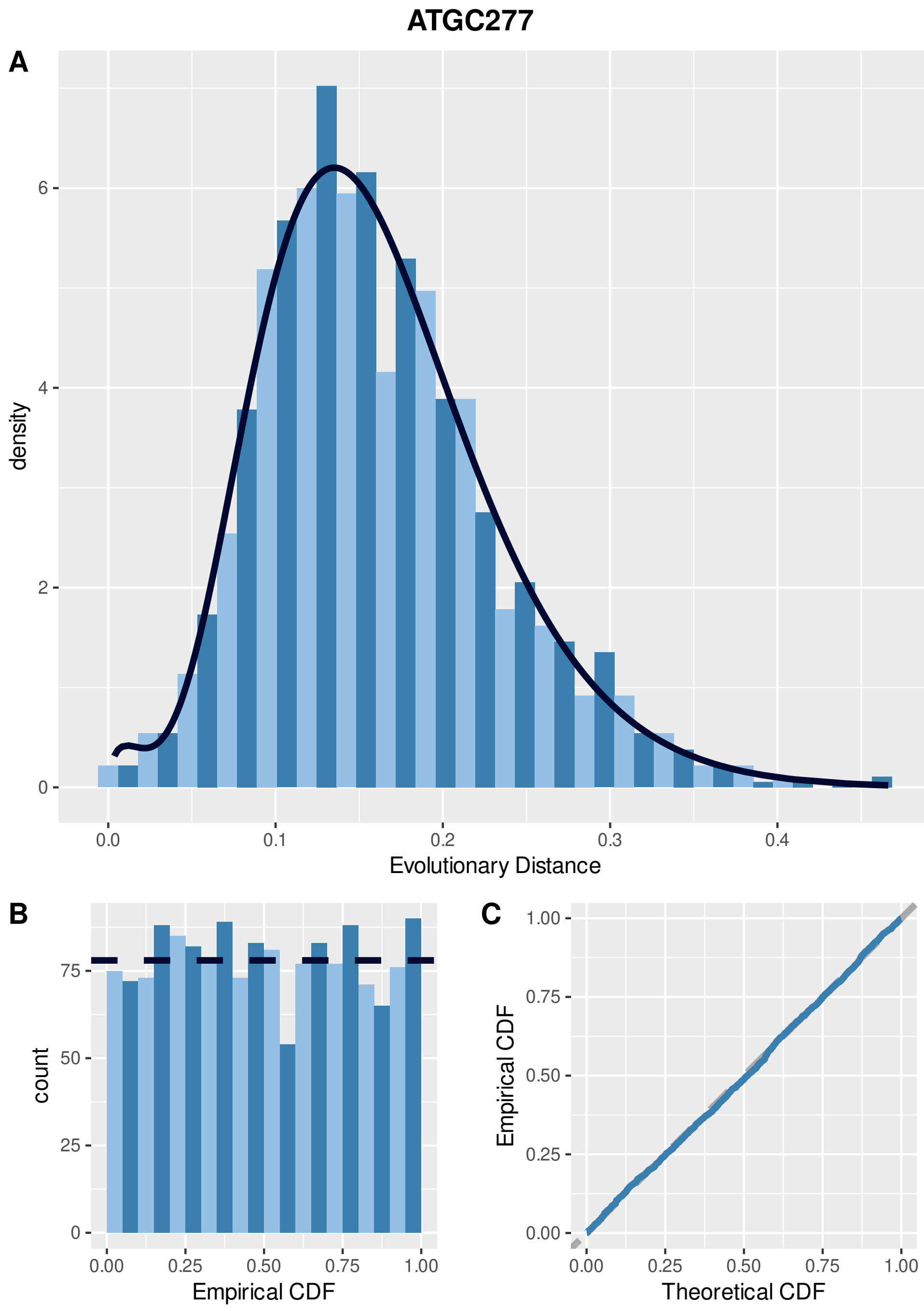

ATGC243
A
density
5
4
3
2
1
0
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
120
80
40
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

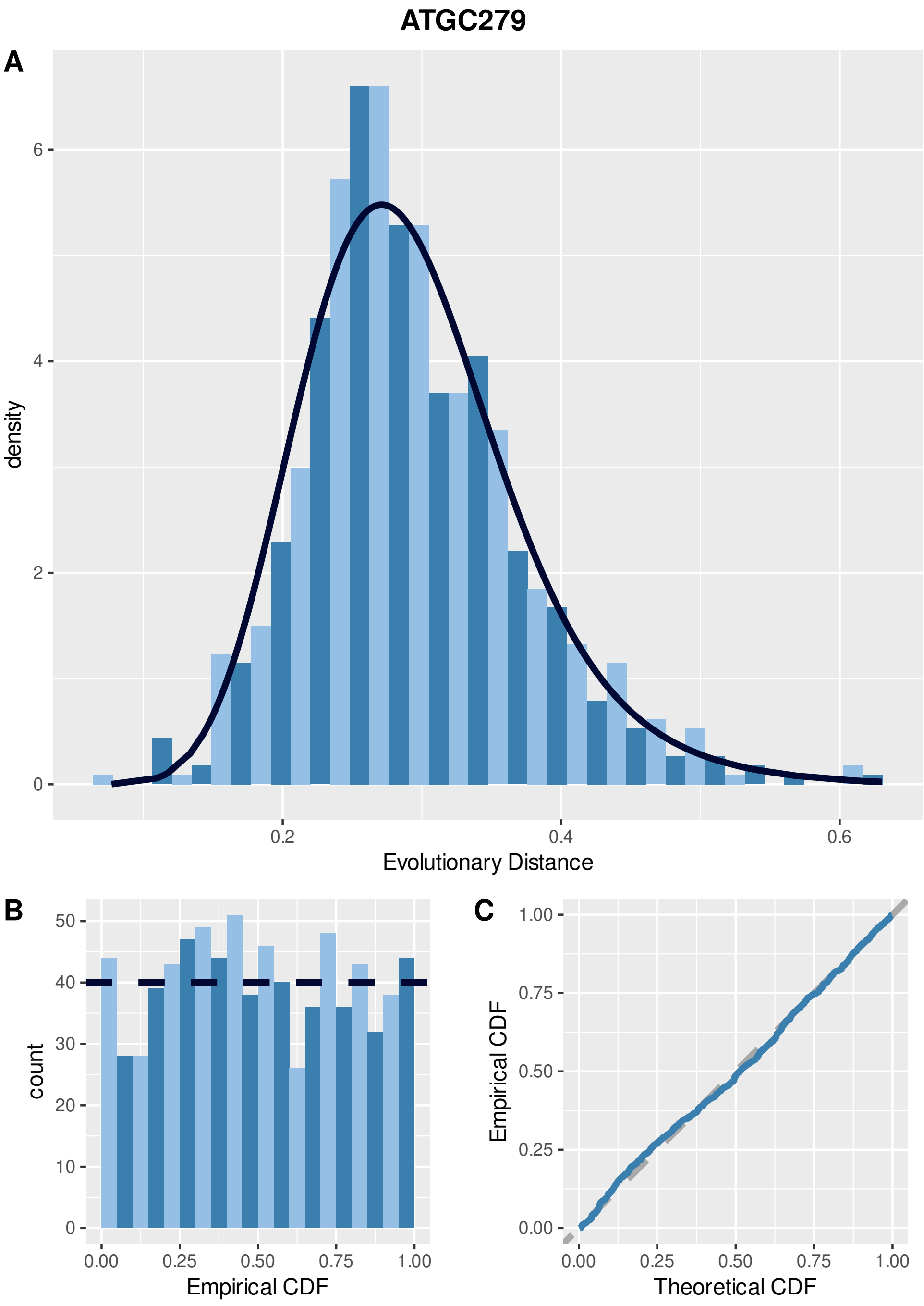

ATGC244
A
density
6
4
2
0
0.0
0.2
0.4
Evolutionary Distance
B
count
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

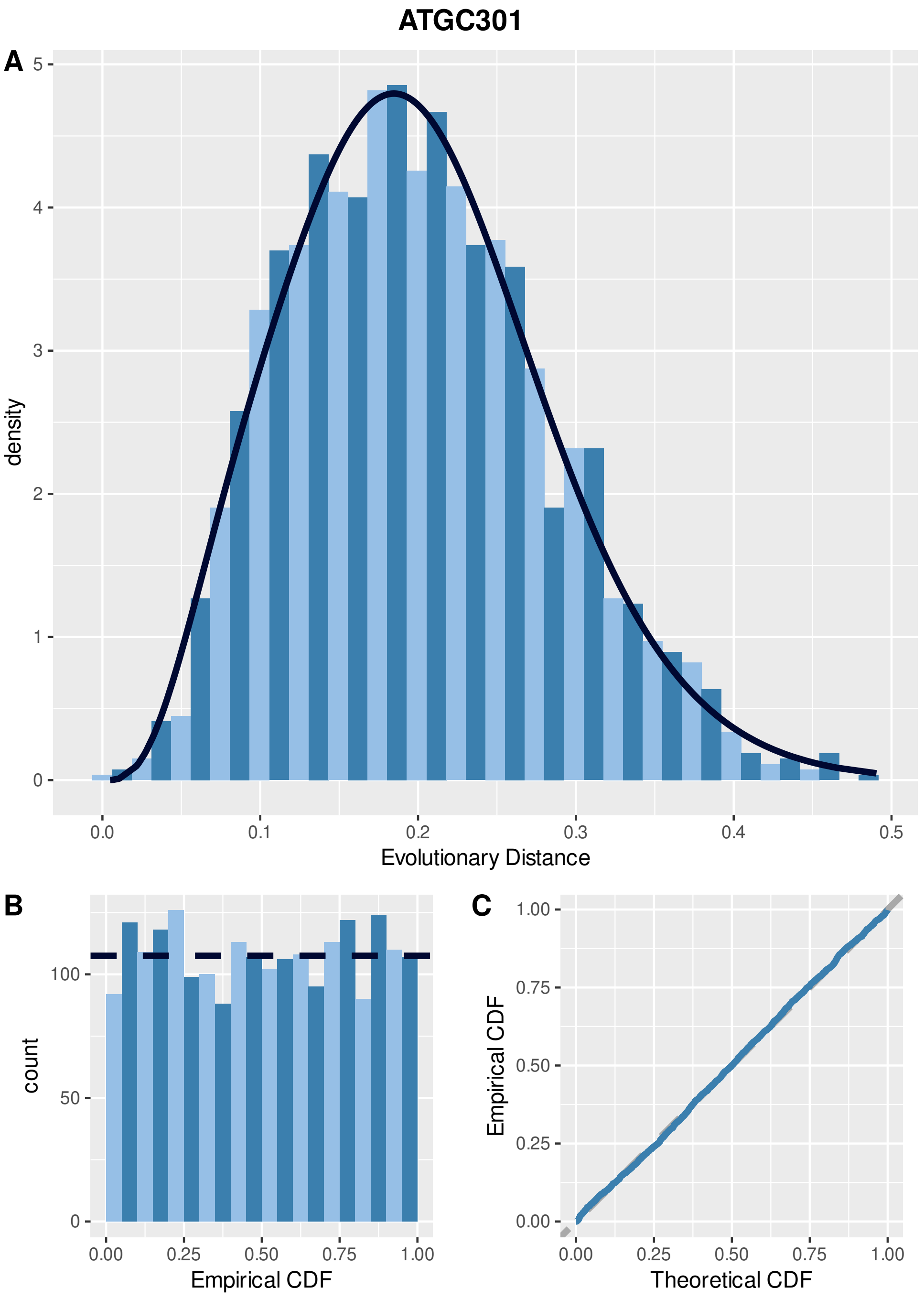

ATGC246
A
12
9
6
3
0
density
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
40
30
20
10
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

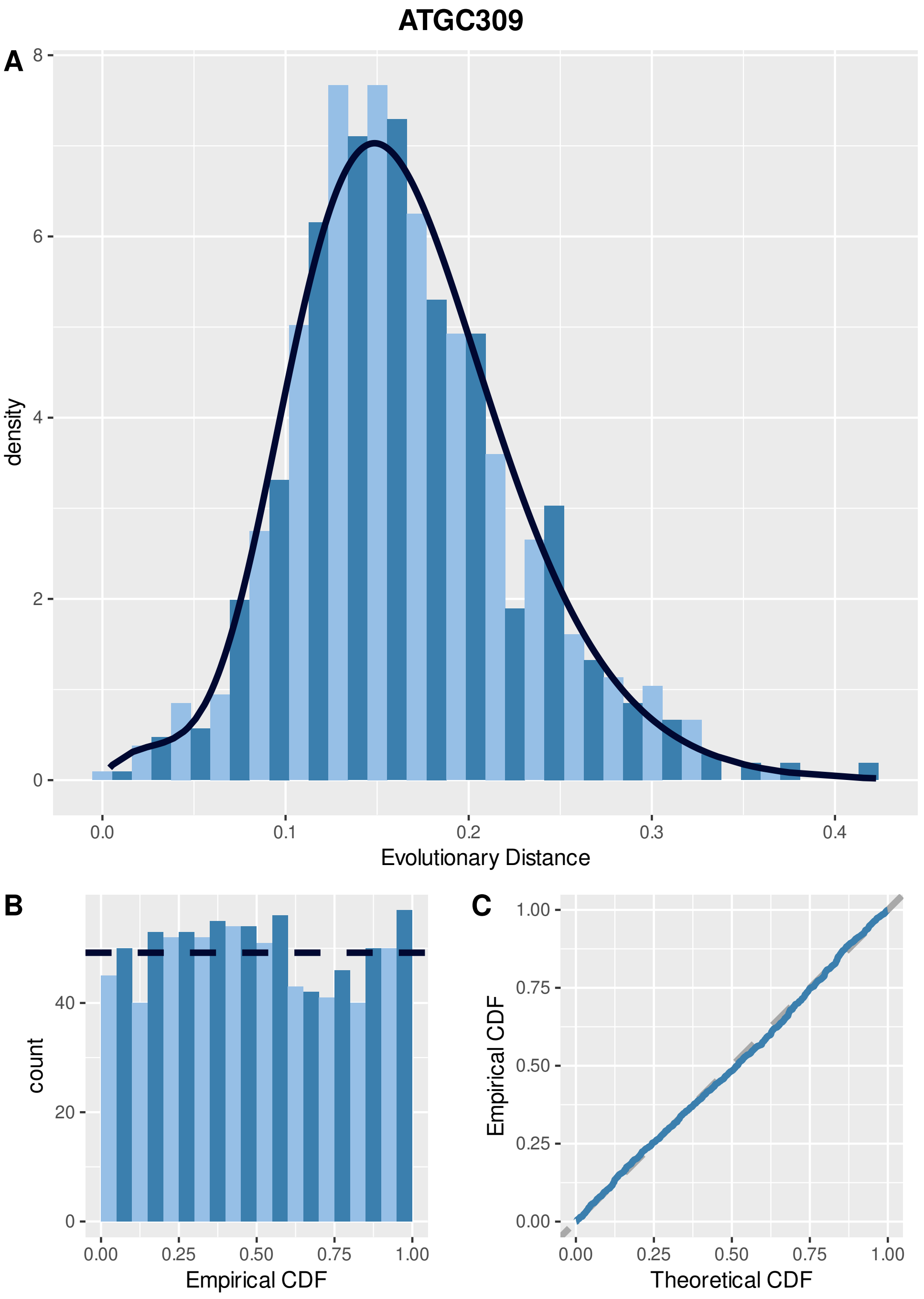

ATGC258
A
10.0
7.5
5.0
2.5
0.0
density
0.1
0.2
0.3
0.4
Evolutionary Distance
B
75
50
25
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

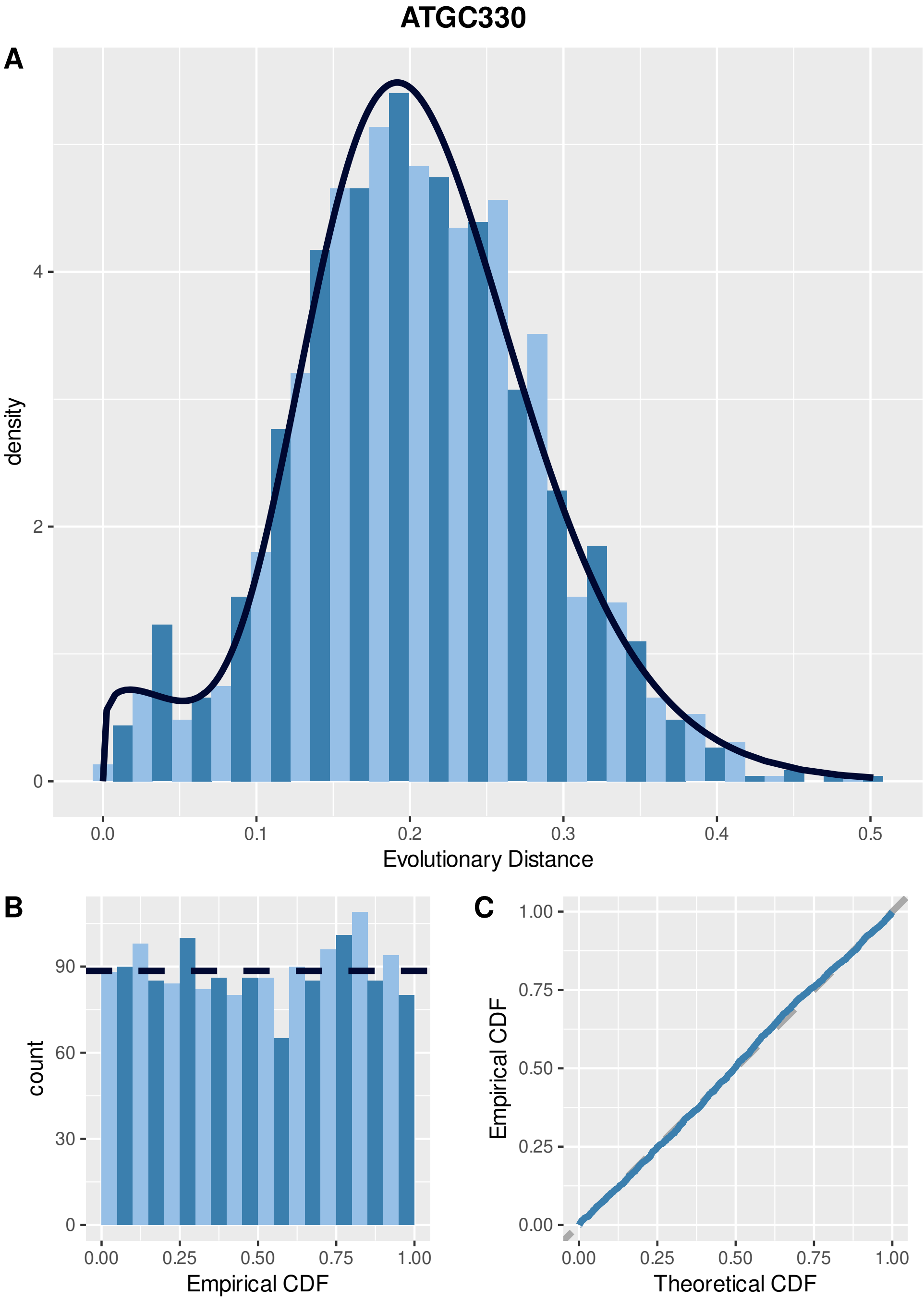

ATGC260
A
density
9
6
3
0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC277
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

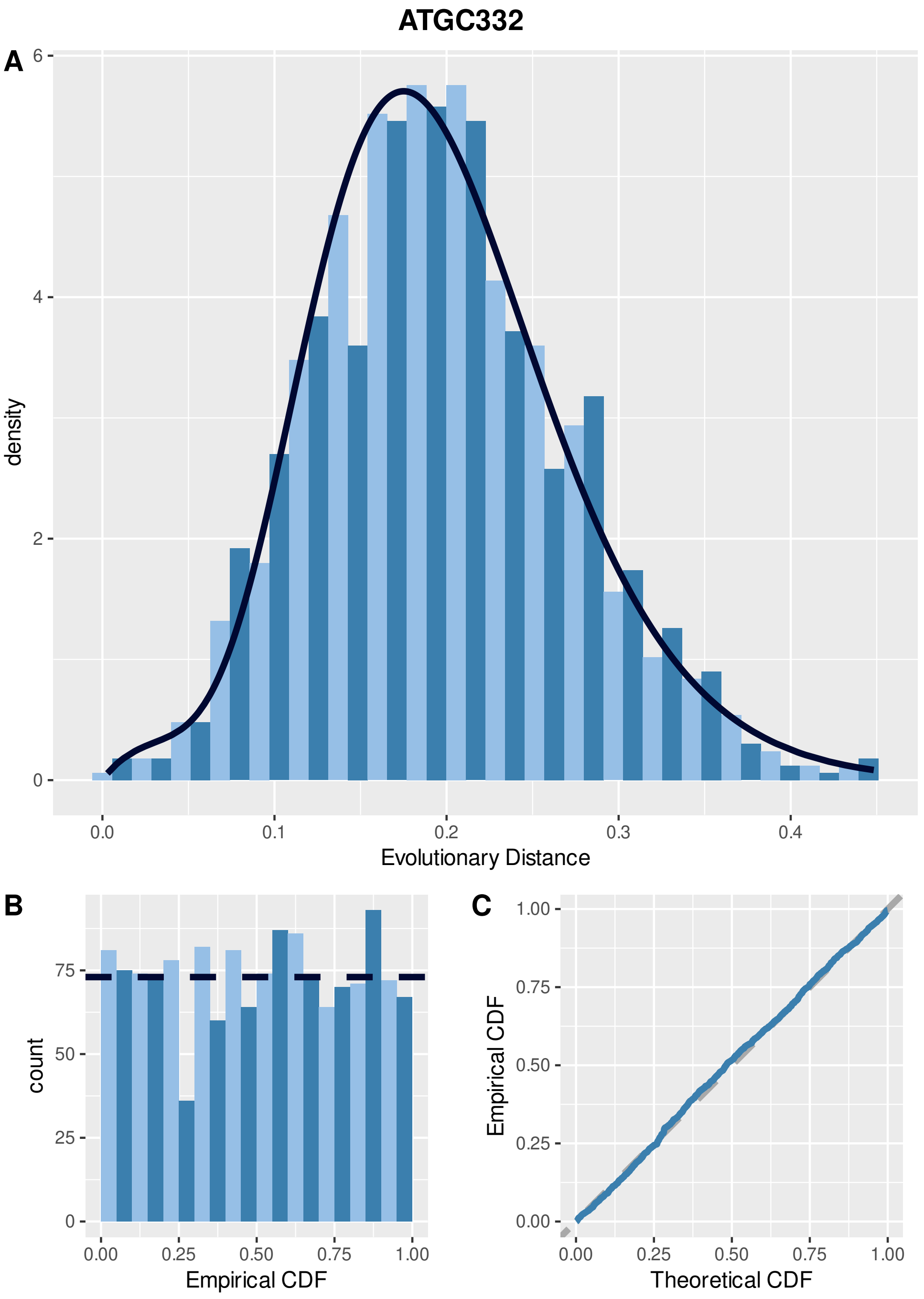

ATGC279
A
density
6
4
2
0
0.2
0.4
0.6
Evolutionary Distance
B
count
50
40
30
20
10
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

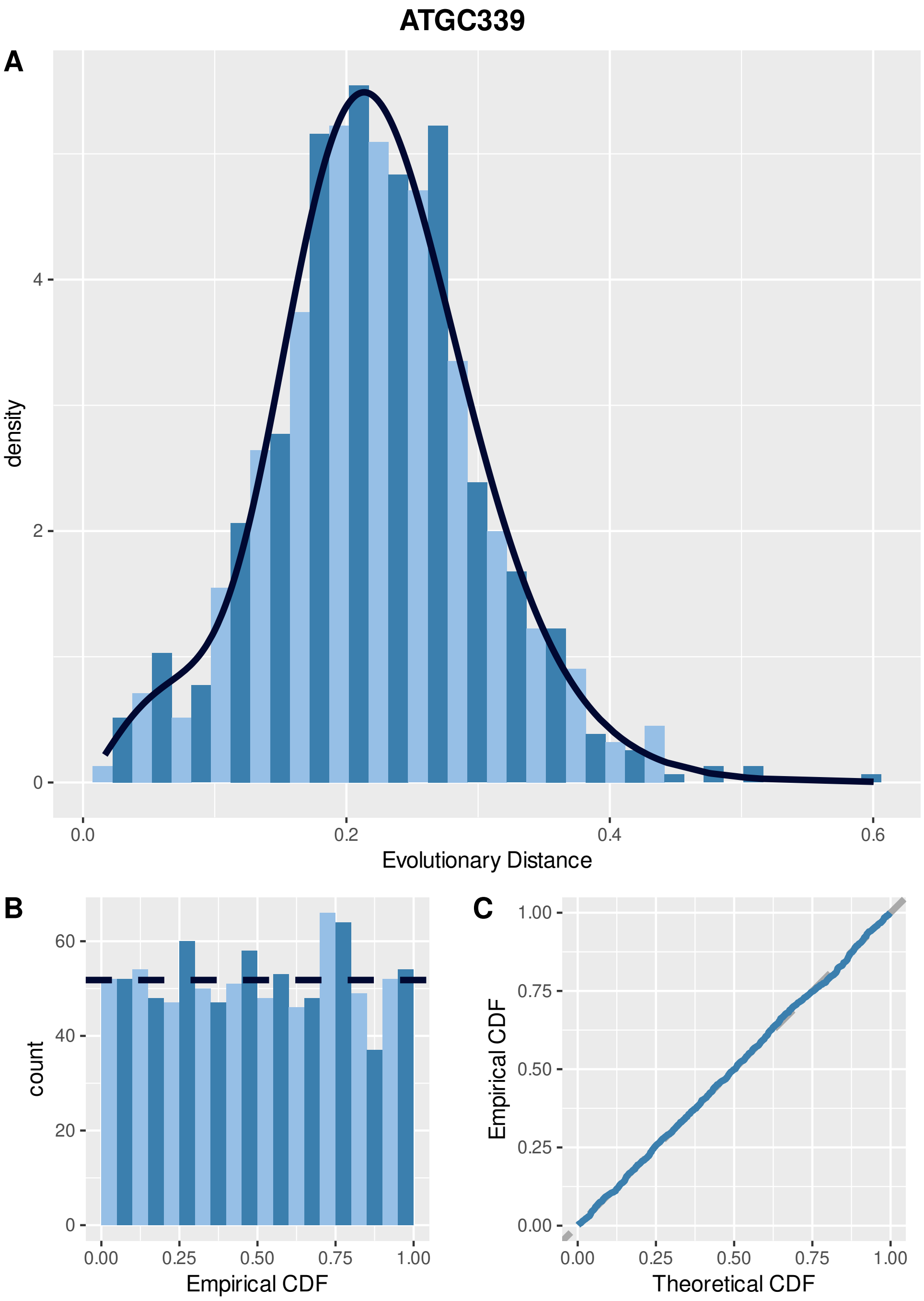

ATGC301
A
density
5
4
3
2
1
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
100
50
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

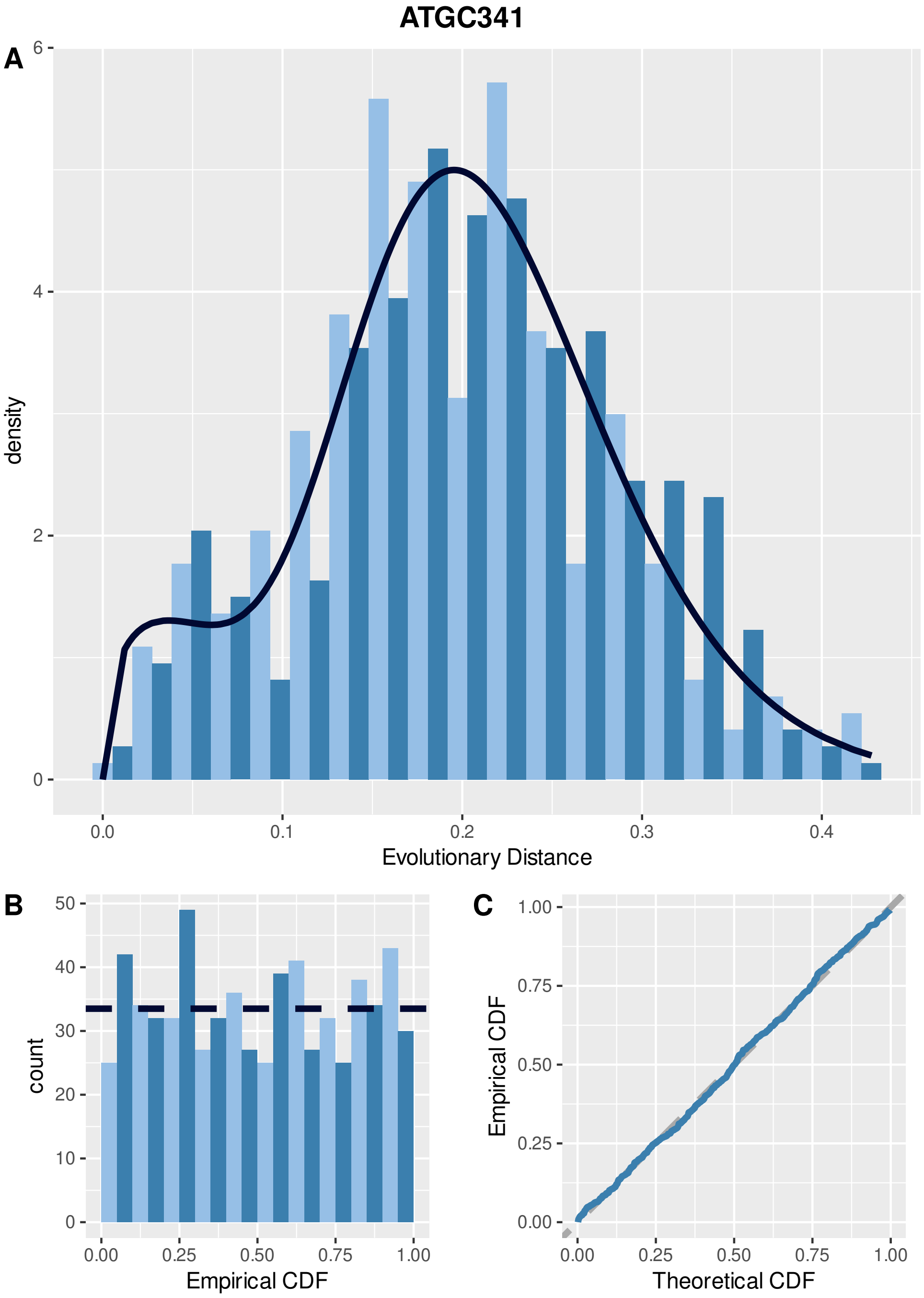

ATGC309
A
density
8
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

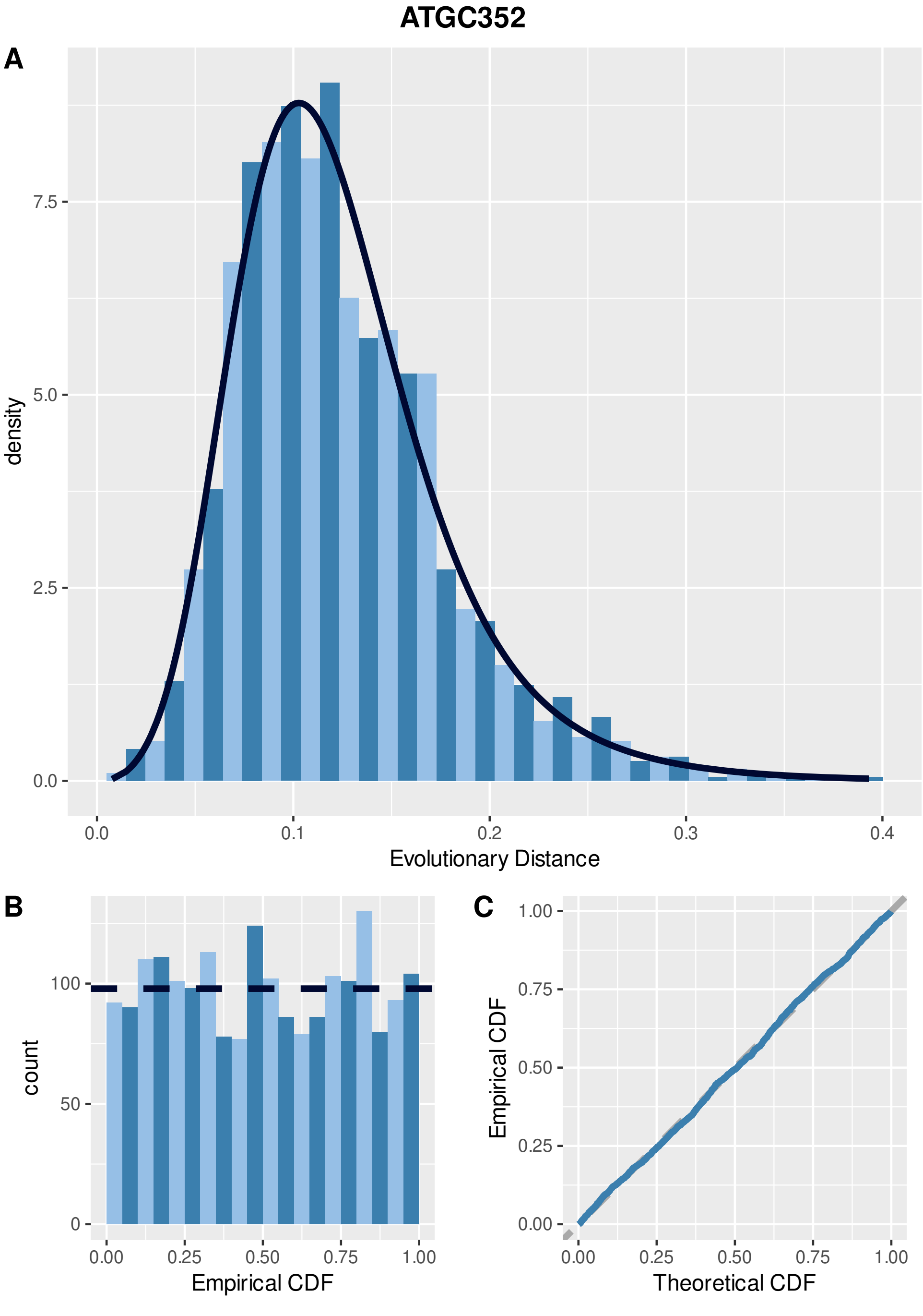


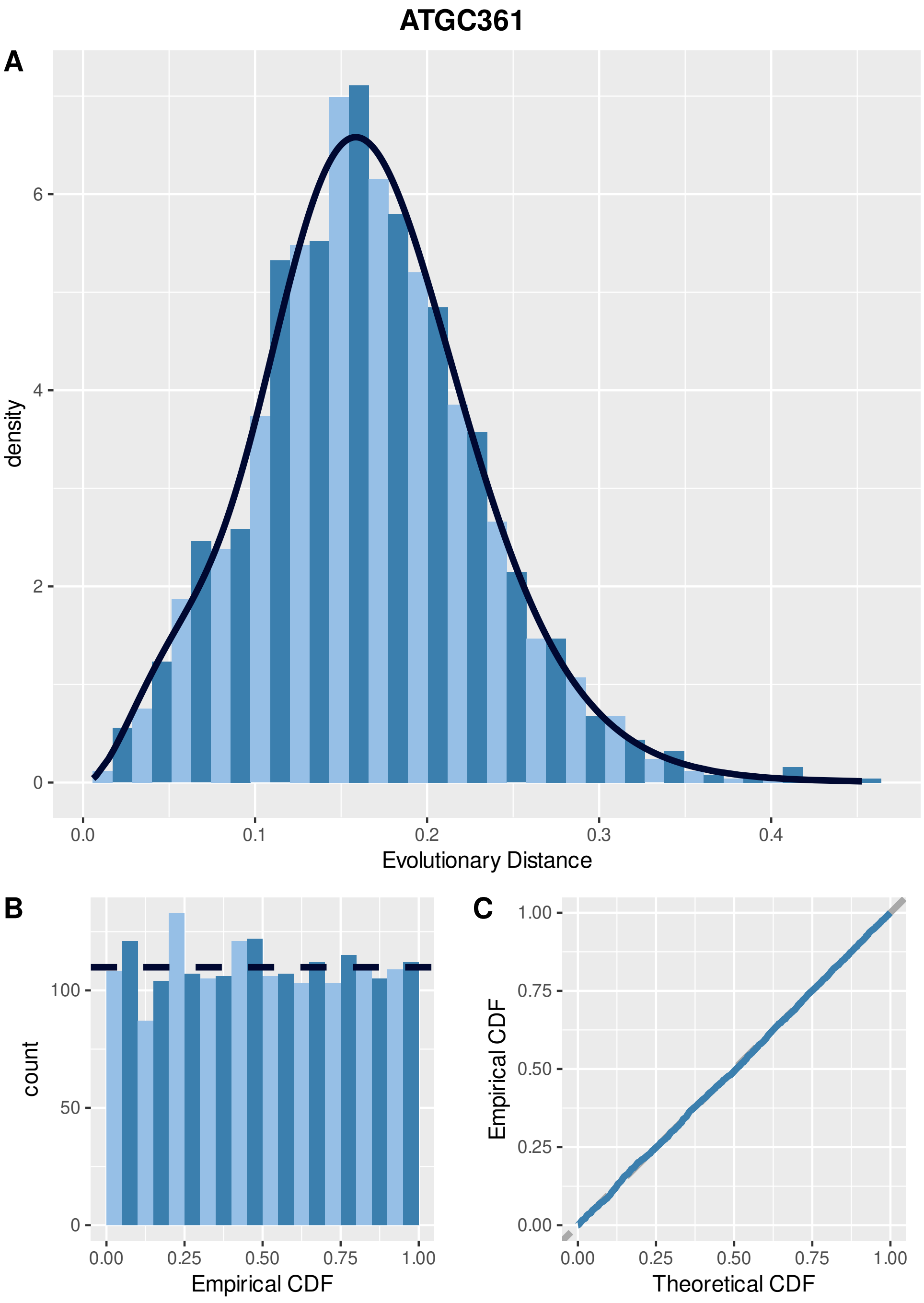
ATGC330
A
density
4
2
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
90
60
30
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

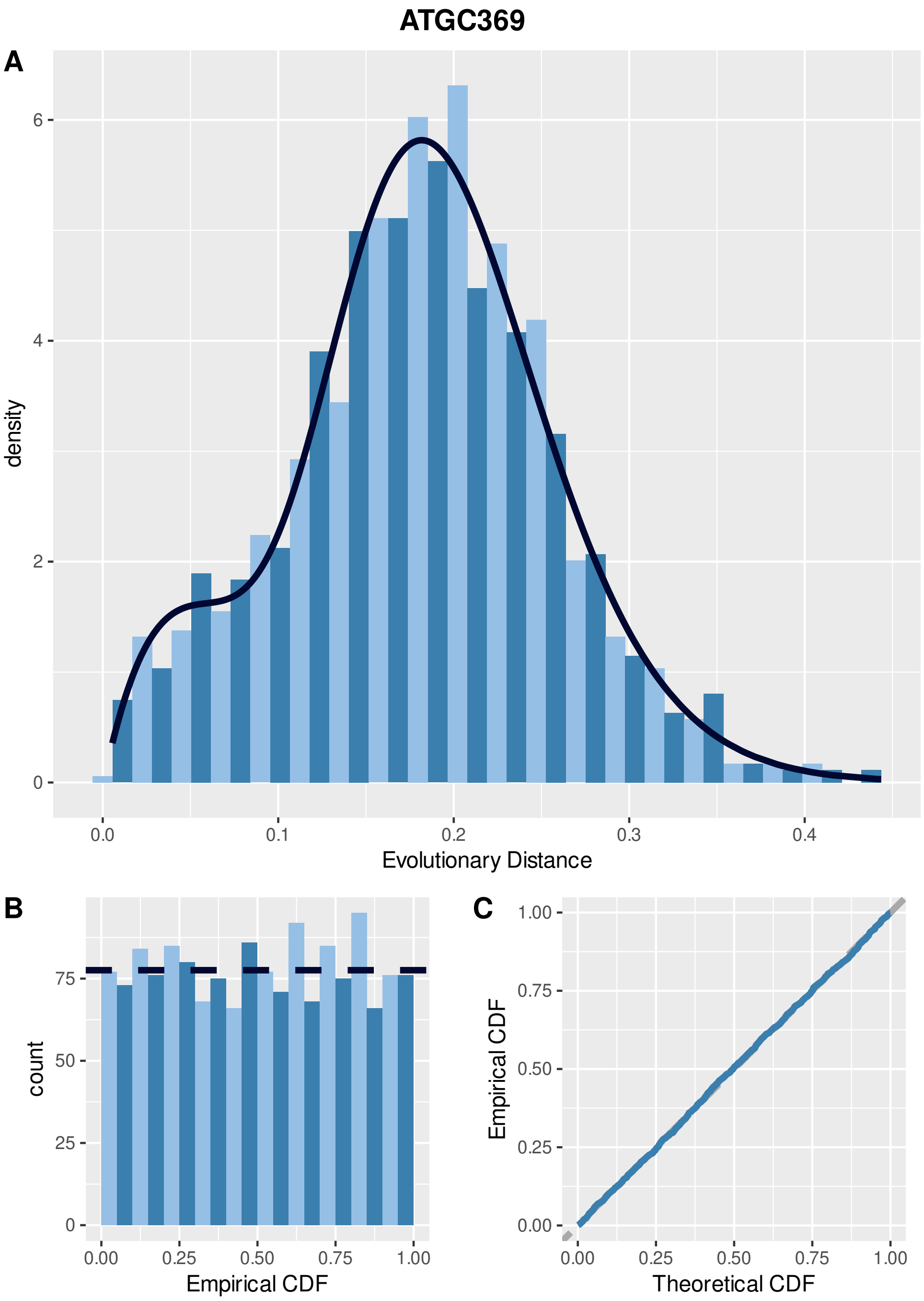

ATGC332
A
6
4
2
0
density
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
75
50
25
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

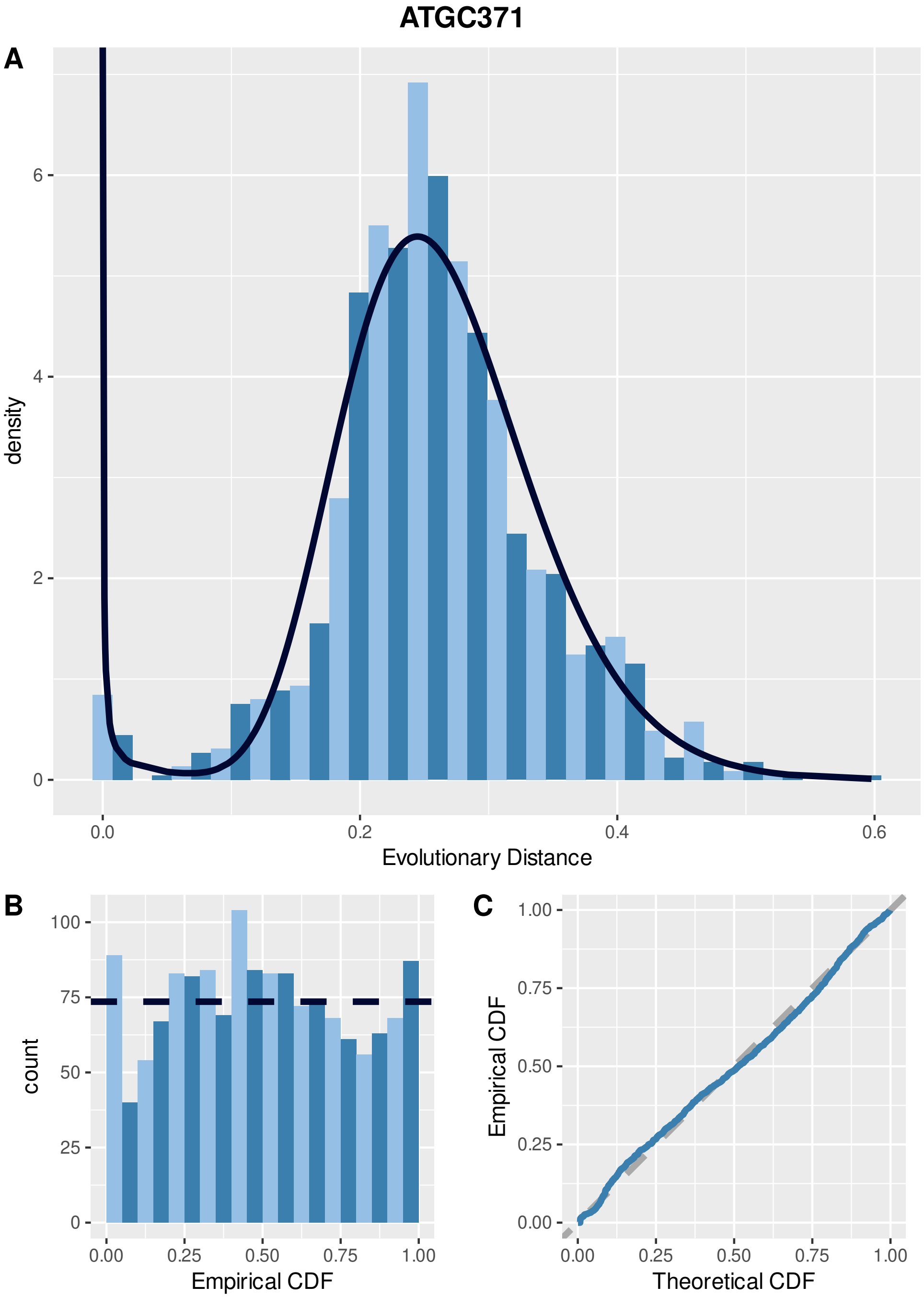

ATGC339
A
density
4
2
0
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC341
A
6
4
2
0
density
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
50
40
30
20
10
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

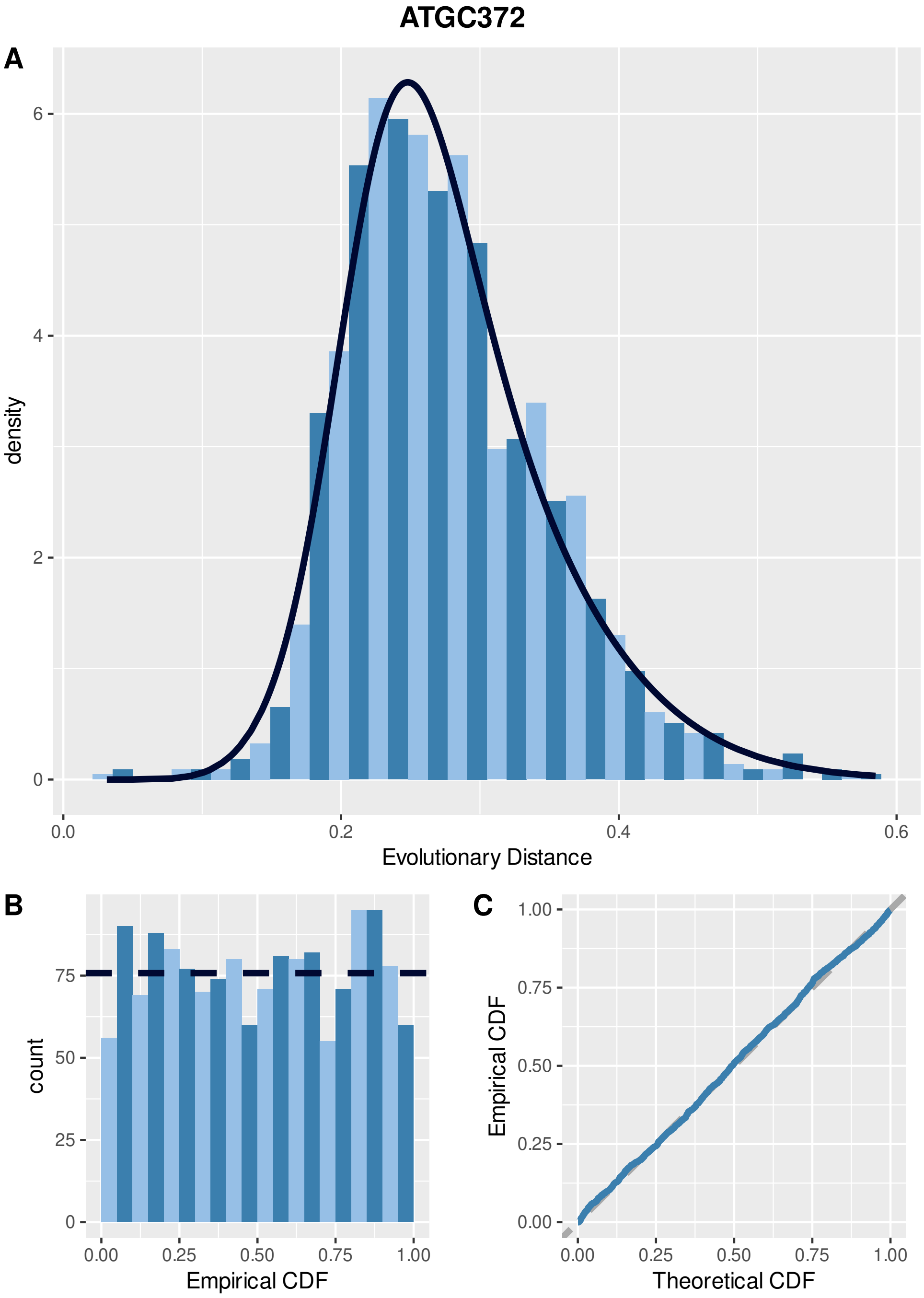

ATGC352
A
density
7.5
5.0
2.5
0.0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
100
50
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

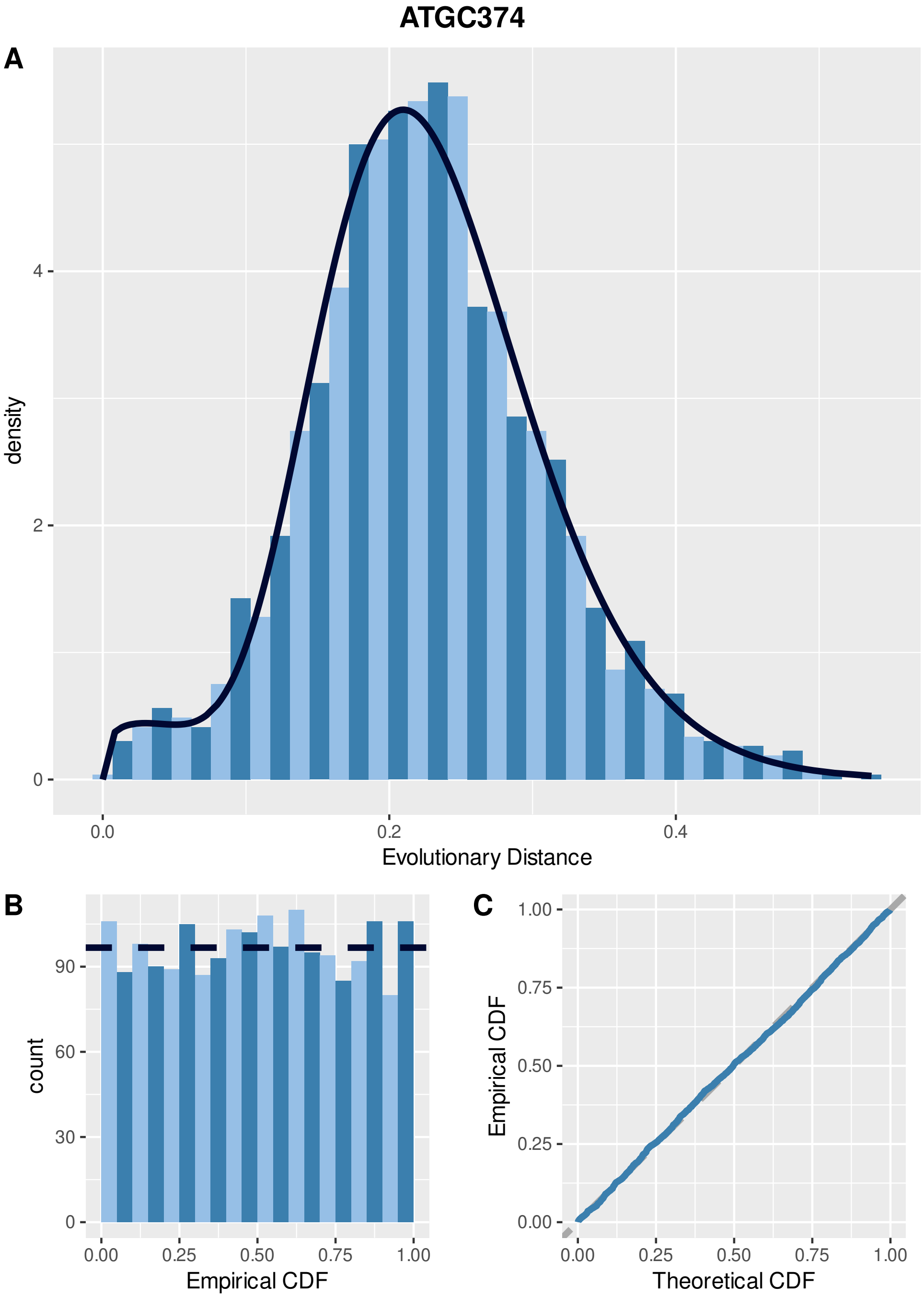

ATGC361
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
100
50
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

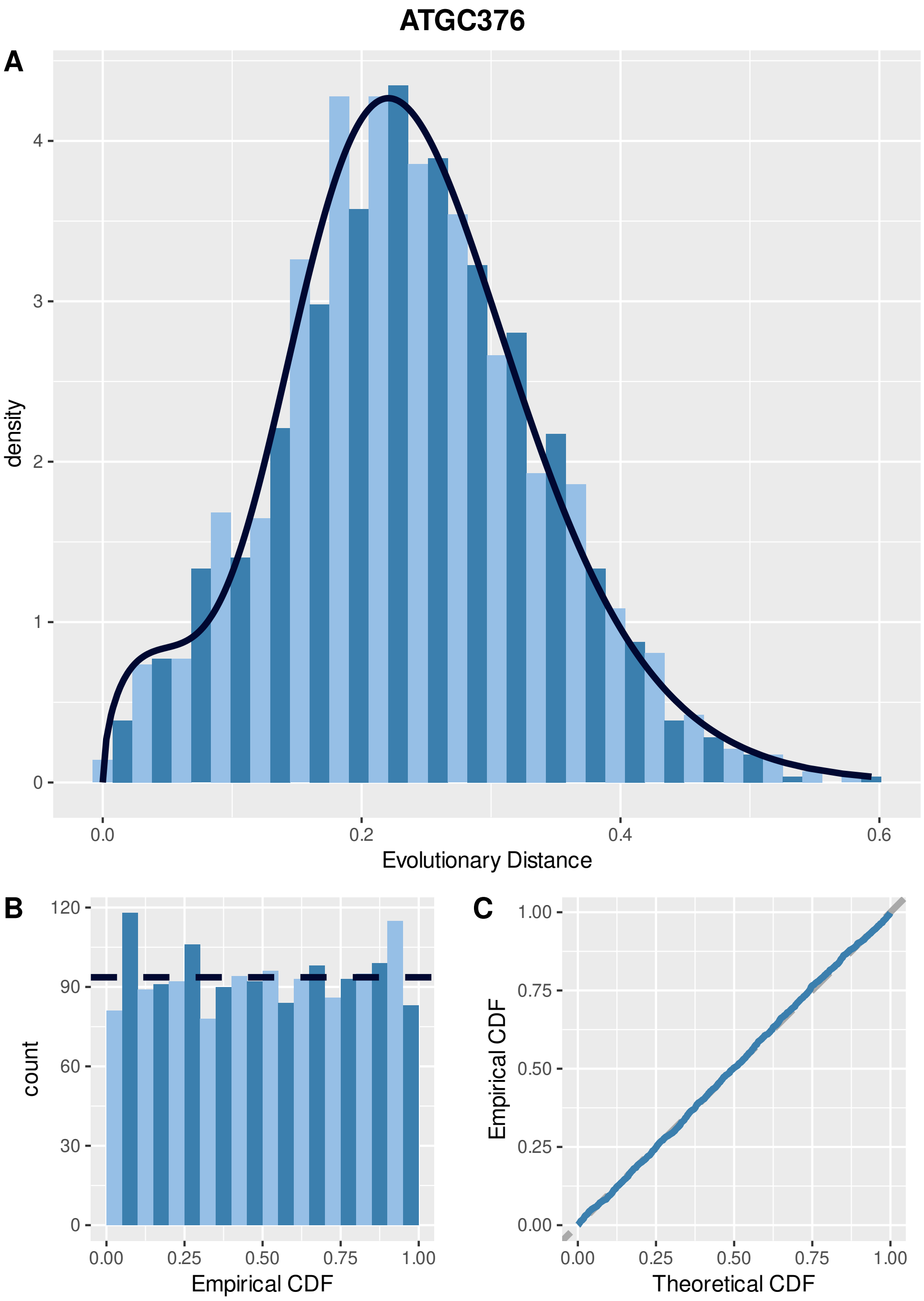

ATGC369
A
density
6
4
2
0
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

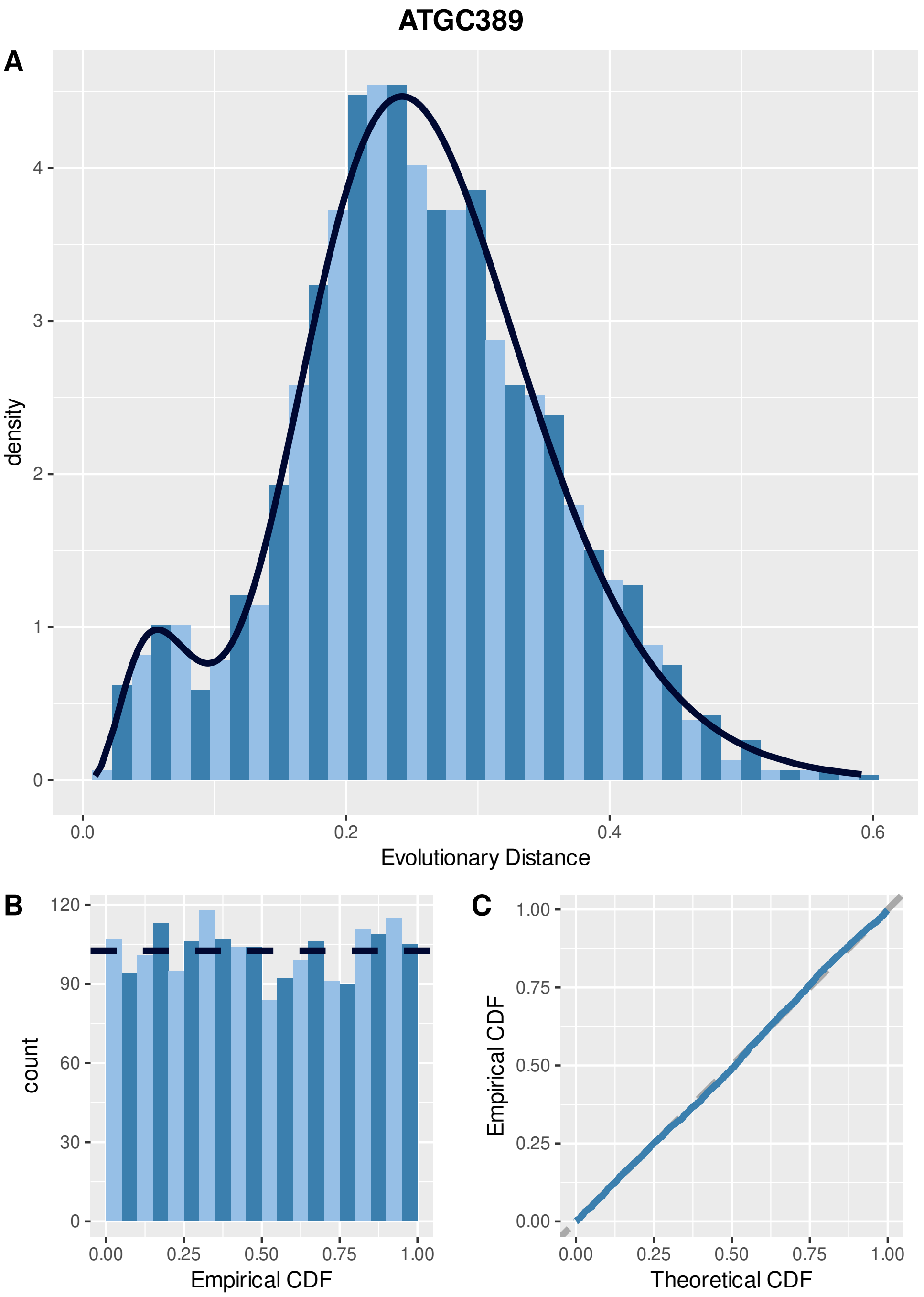


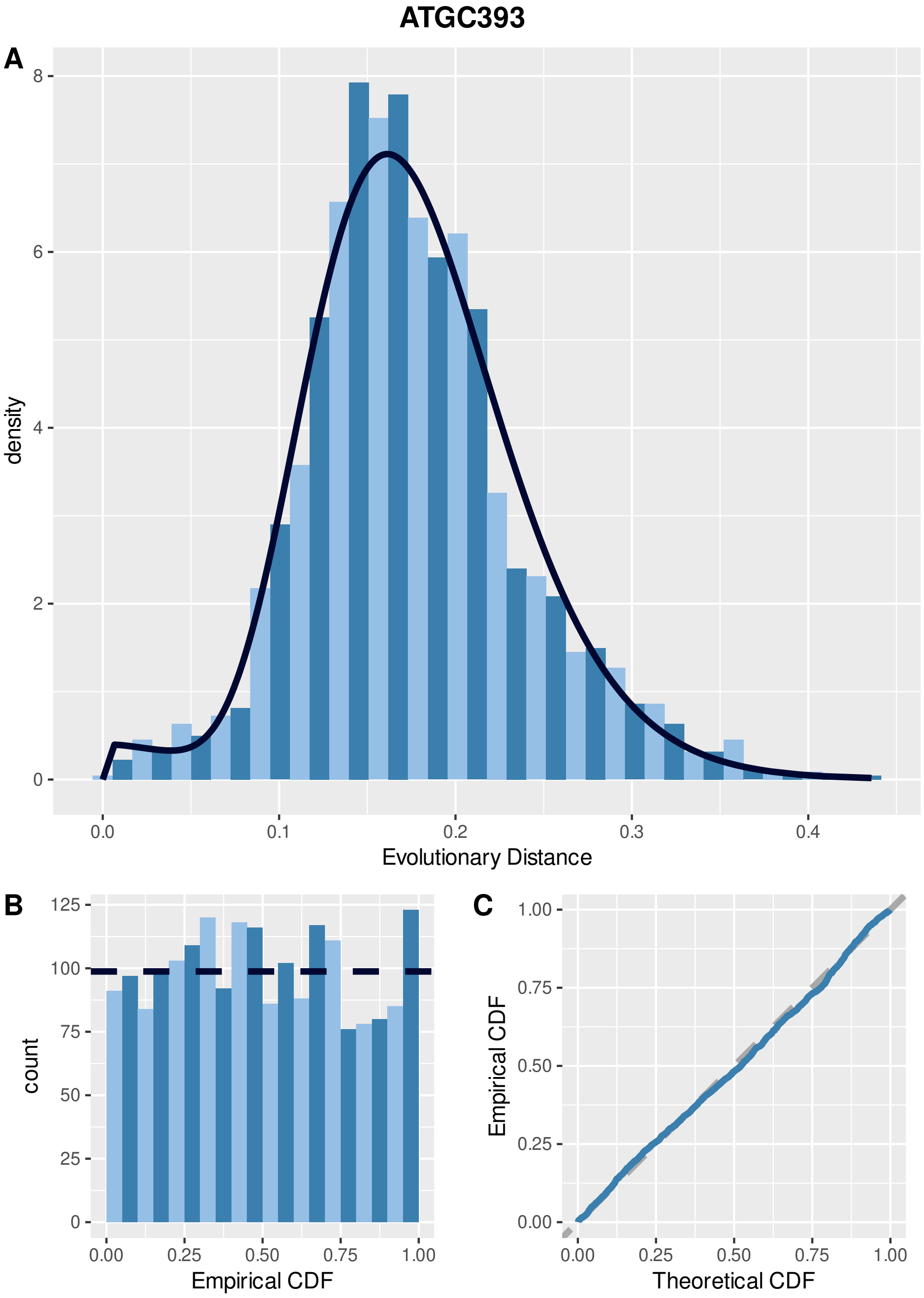
ATGC371
A
6
4
2
0
density
0.0
0.2
0.4
0.6
Evolutionary Distance
B
100
75
50
25
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

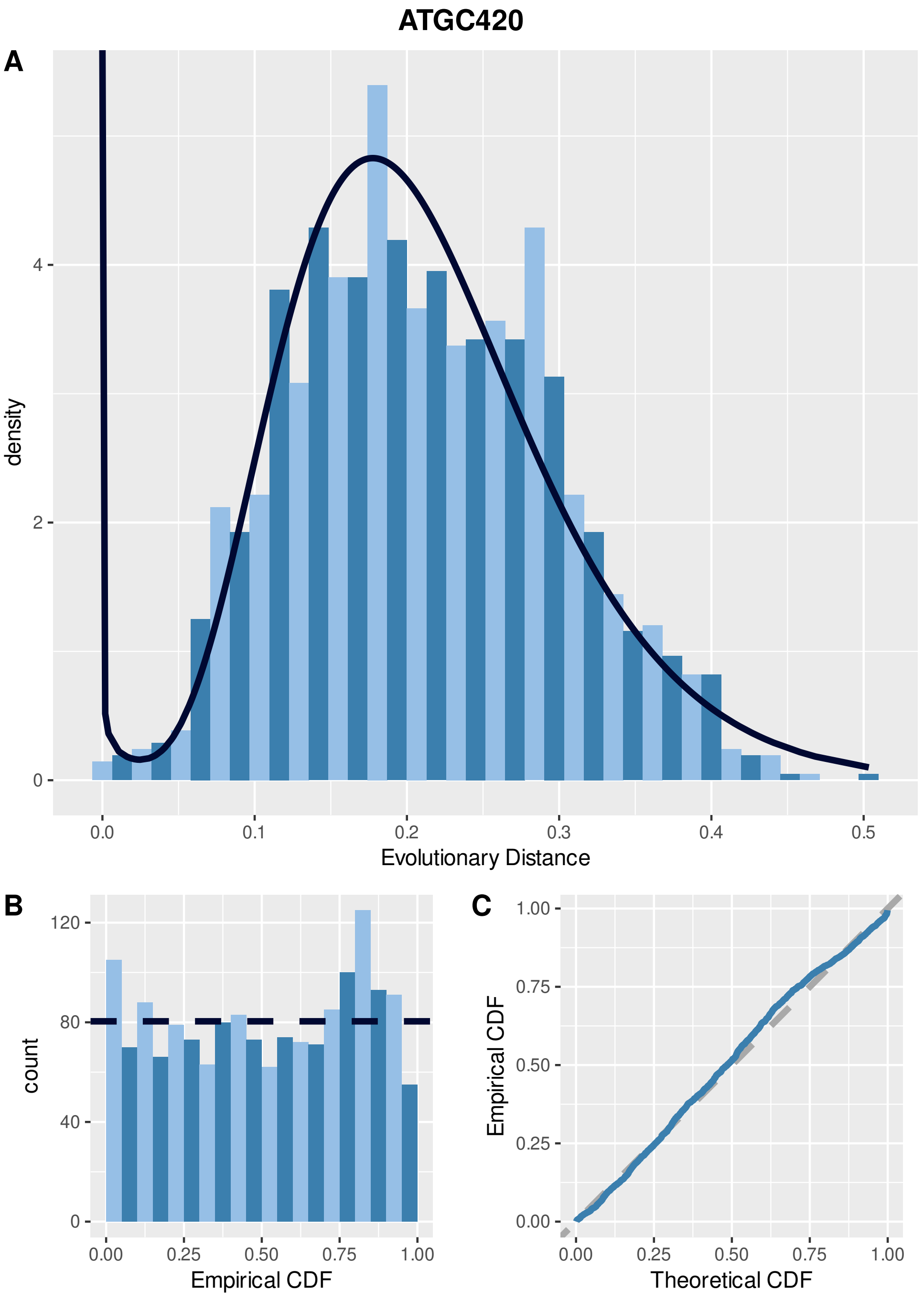

ATGC372
A
density
6
4
2
0
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
75
50
25
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

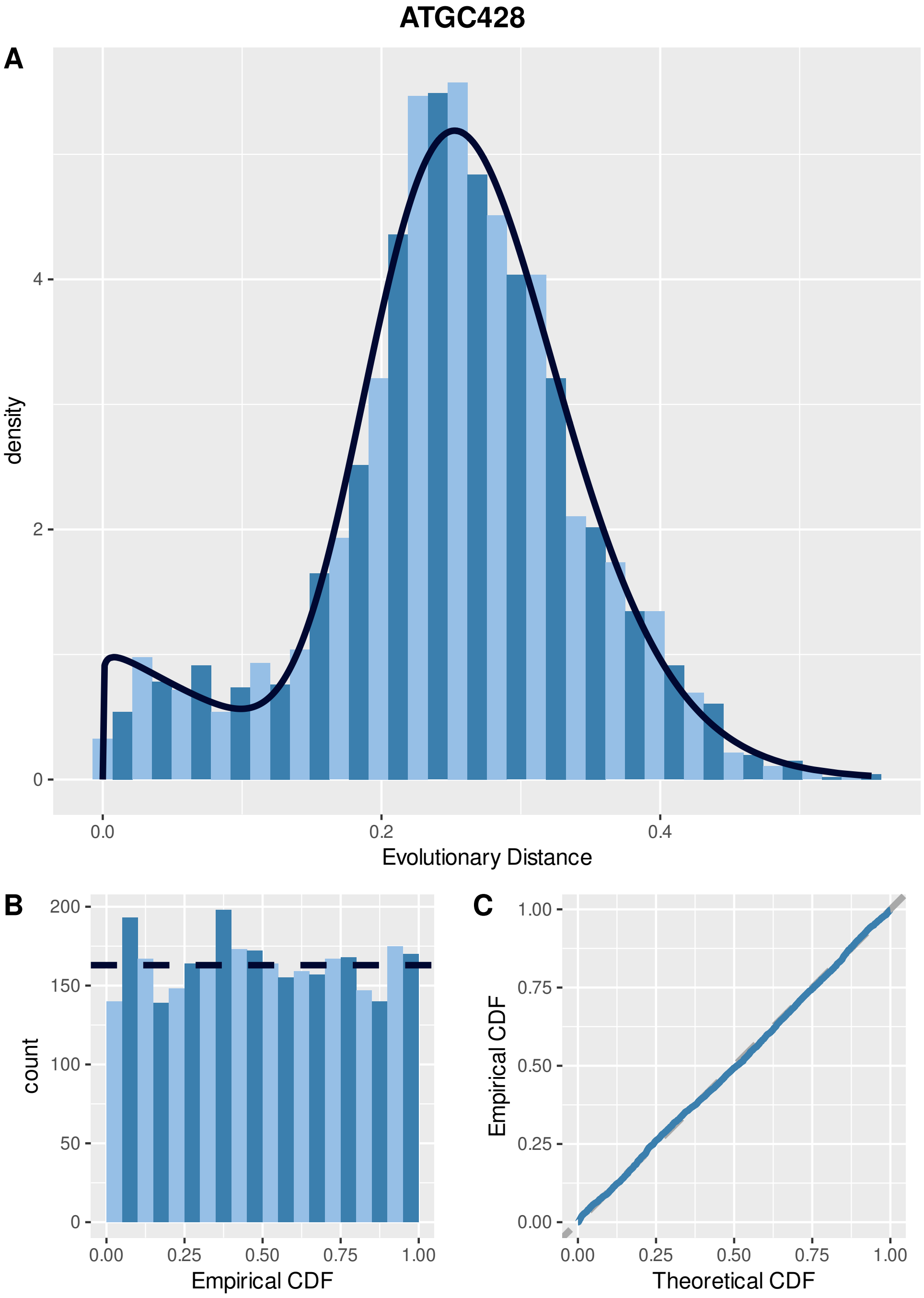

ATGC374
A
density
4
2
0
0.0
0.2
0.4
Evolutionary Distance
B
count
90
60
30
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC376
A
4
3
2
1
0
density
0.0
0.2
0.4
0.6
Evolutionary Distance
B
120
90
60
30
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC389
A
density
0
1
2
3
4
0.0
0.2
0.4
0.6
Evolutionary Distance
B
count
0
30
60
90
120
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC393
A
8
6
4
2
0
density
0.0
0.1
0.2
0.3
0.4
Evolutionary Distance
B
125
100
75
50
25
0
count
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC420
A
density
4
2
0
0.0
0.1
0.2
0.3
0.4
0.5
Evolutionary Distance
B
count
120
80
40
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF


ATGC428
A
density
0
2
4
0.0
0.2
0.4
Evolutionary Distance
B
count
0
50
100
150
200
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

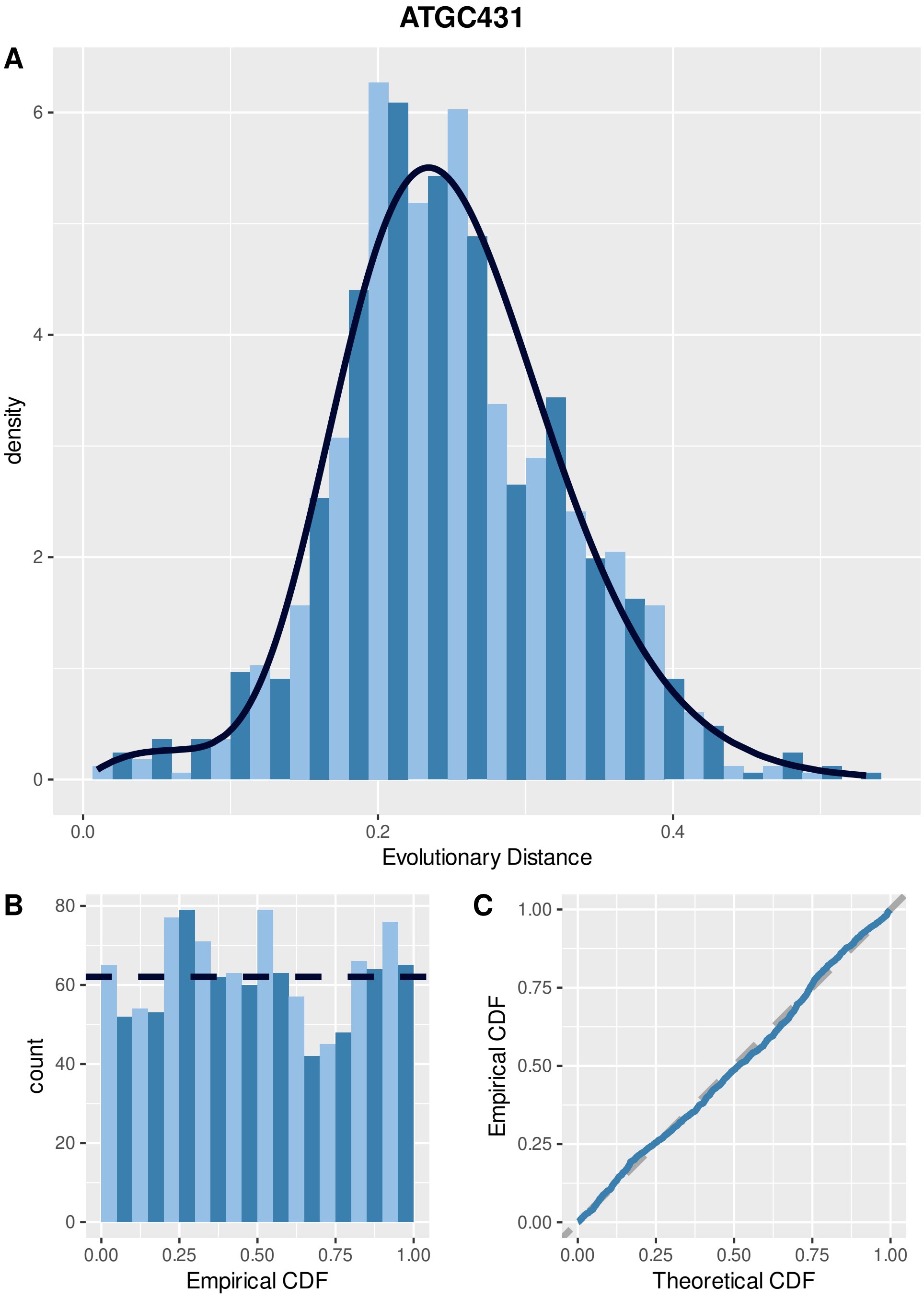
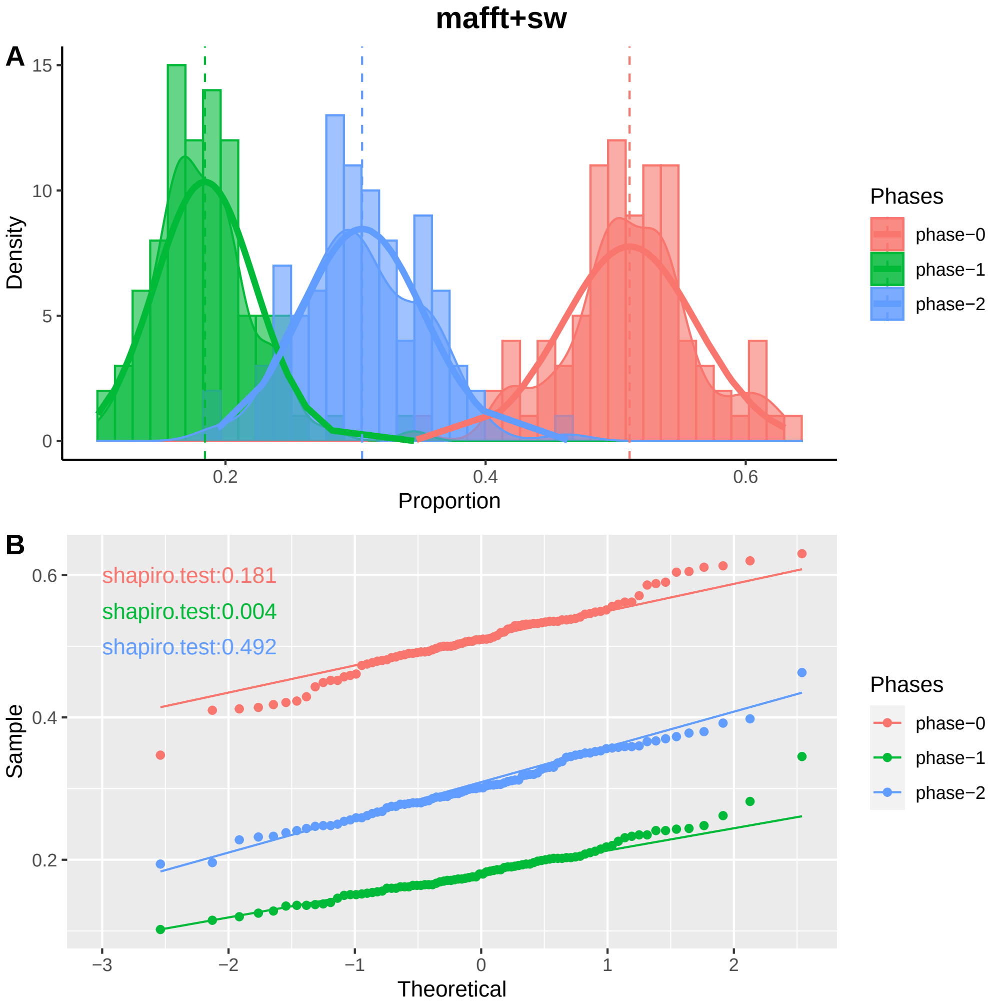
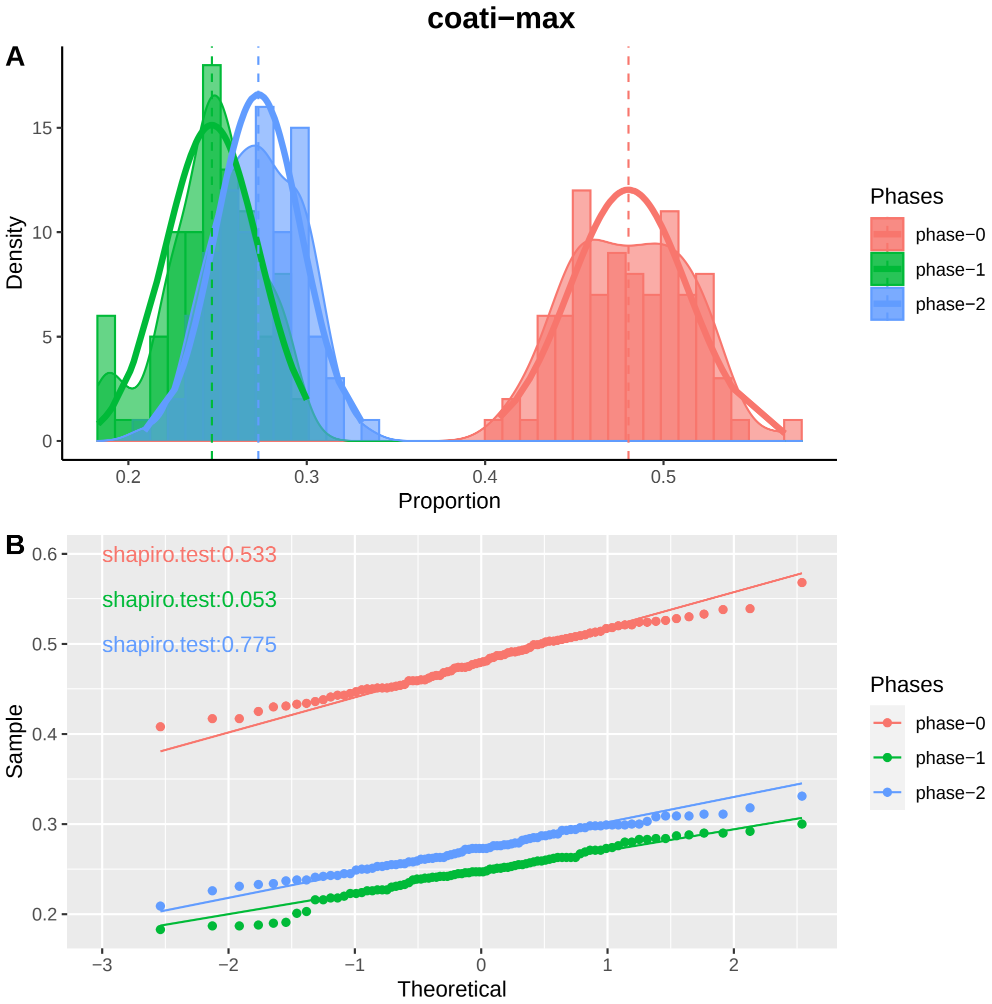

ATGC431
A
density
6
4
2
0
0.0
0.2
0.4
Evolutionary Distance
B
count
80
60
40
20
0
0.00
0.25
0.50
0.75
1.00
Empirical CDF
C
Empirical CDF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
Theoretical CDF

Figure D.1: **Post-filtering Distance Distribution Plot of 90 Species |** A-plot is the evolutionary distance per gene pair that approximates a mixture gamma distribution (components=2). B-plot is the empirical CDF of the distance that follows an uniform distribution roughly, and C-plot is a Q-Q plot between the theoretical CDF and the empirical CDF of the distance data.

Histogram and QQplot of Phase Proportions of 90 Species


mafft+sw
A
Density
15
10
5
0
0.2
0.4
0.6
Proportion
Phases
phase−0
phase−1
phase−2
B
Sample
0.6
0.4
0.2
shapiro.test:0.181
shapiro.test:0.004
shapiro.test:0.492
−3
−2
−1
0
1
2
Theoretical
Phases
phase−0
phase−1
phase−2


coati−max
A
Density
15
10
5
0
0.2
0.3
0.4
0.5
Proportion
Phases
phase−0
phase−1
phase−2
B
shapiro.test:0.533
shapiro.test:0.053
shapiro.test:0.775
Sample
0.6
0.5
0.4
0.3
0.2
−3
−2
−1
0
1
2
Theoretical
Phases
phase−0
phase−1
phase−2

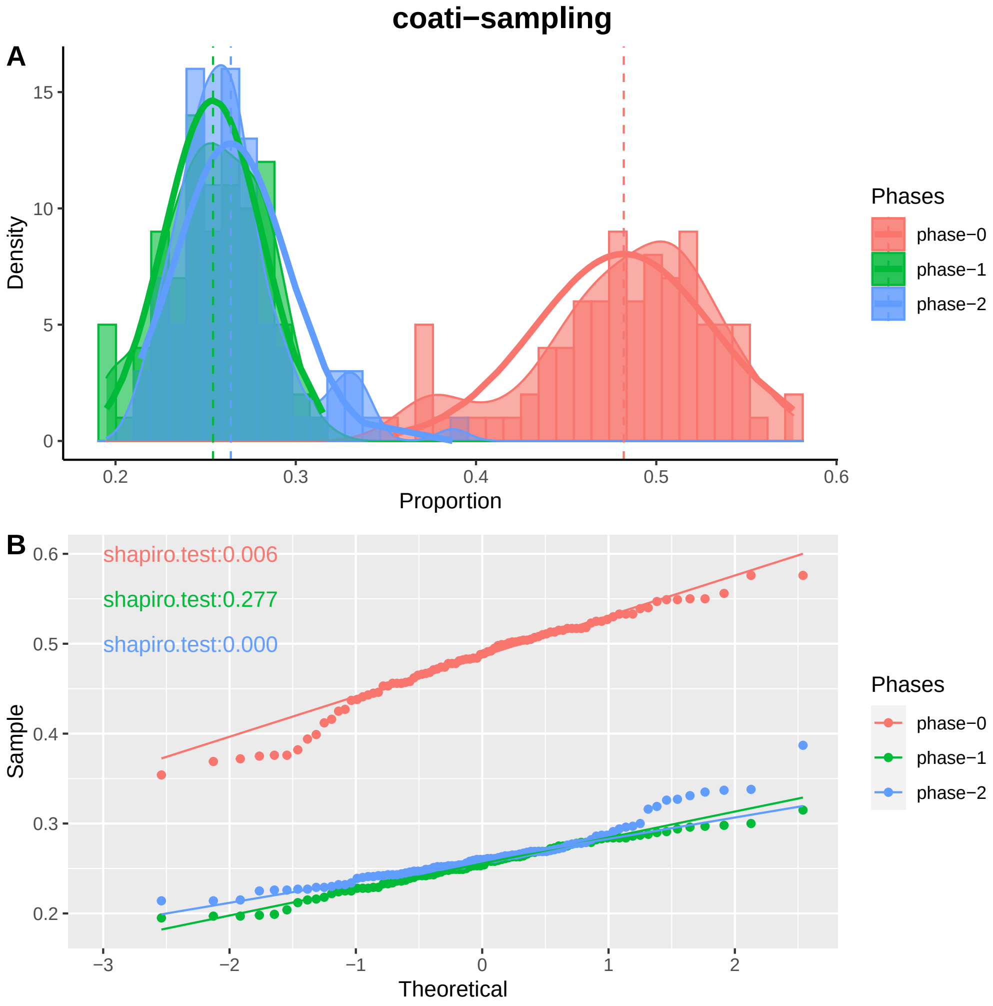
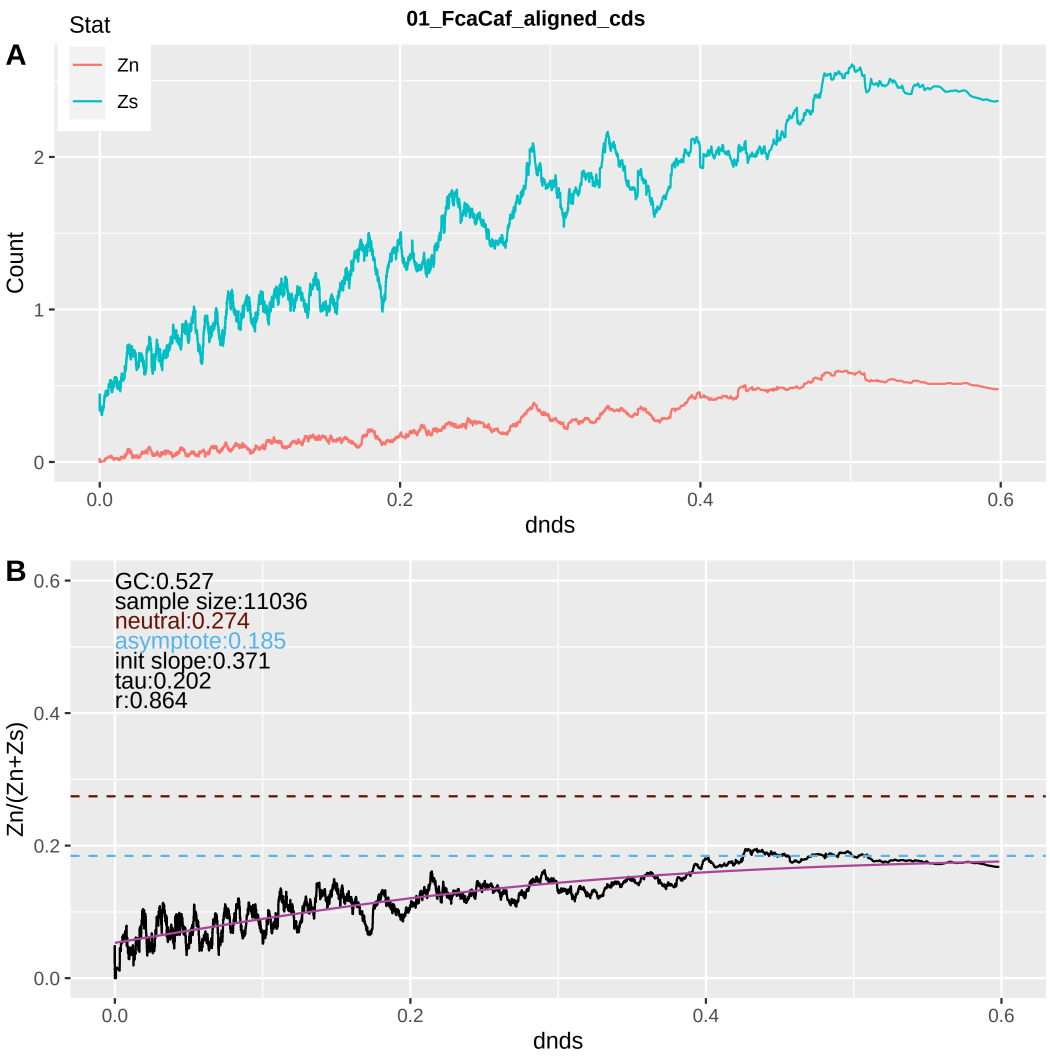
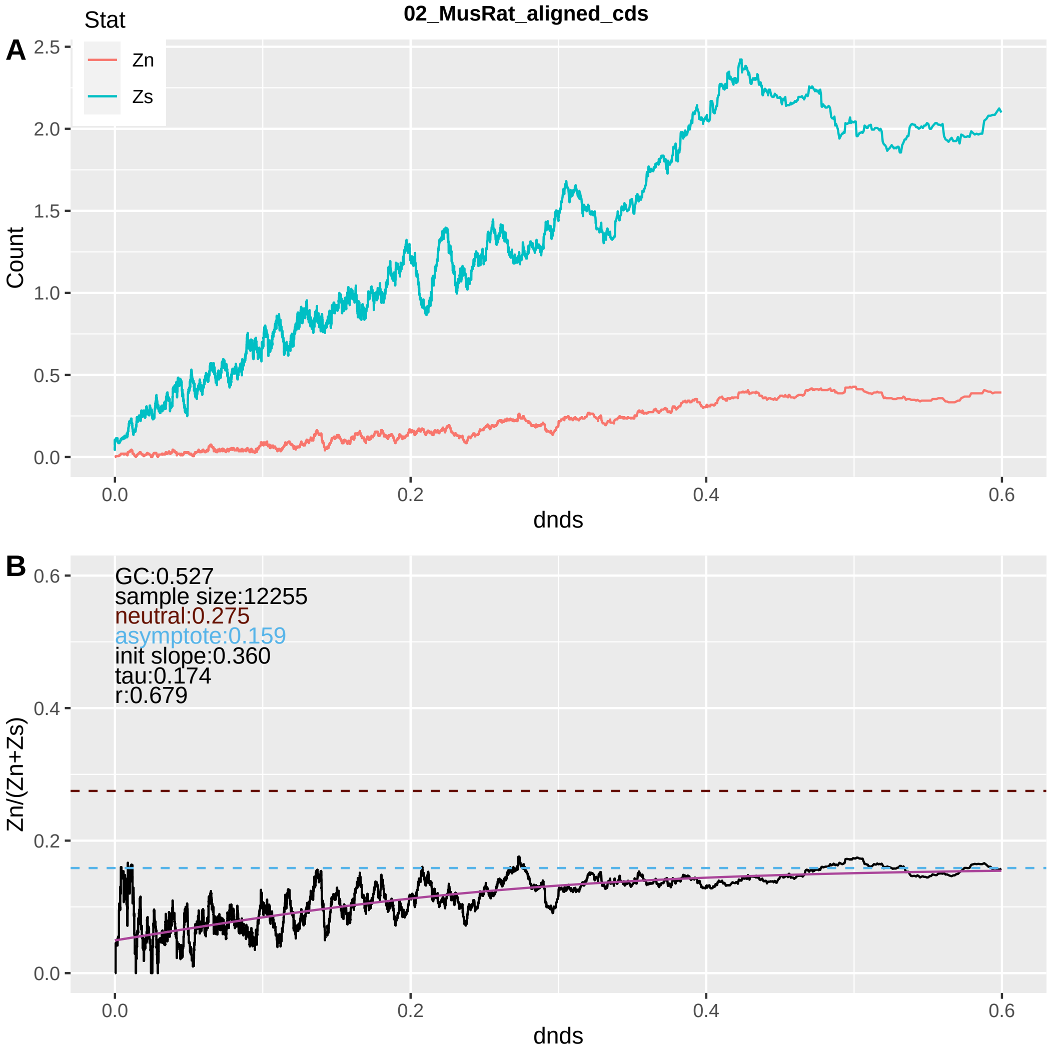
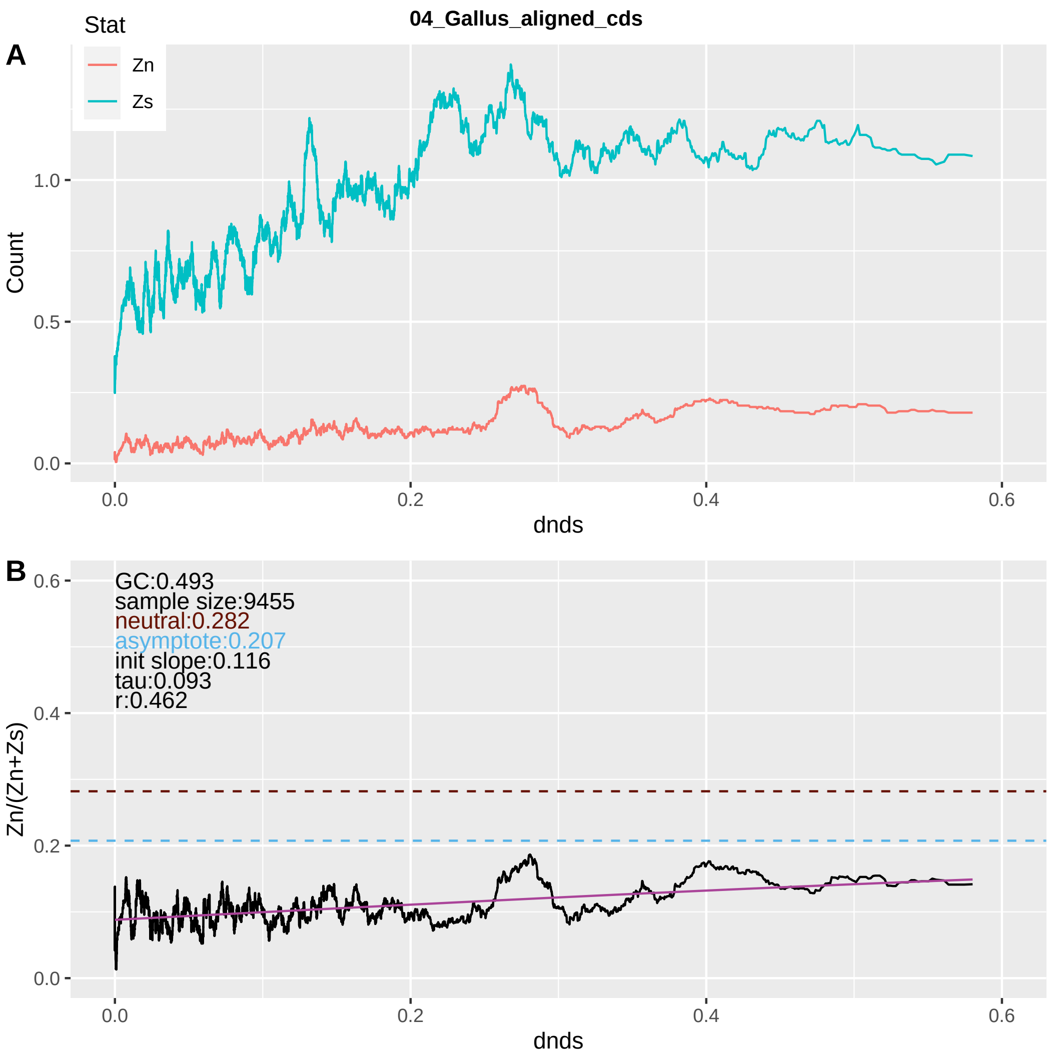

coati−sampling
A
Density
15
10
5
0
0.2
0.3
0.4
0.5
0.6
Proportion
Phases
phase−0
phase−1
phase−2
B
shapiro.test:0.006
shapiro.test:0.277
shapiro.test:0.000
Sample
0.6
0.5
0.4
0.3
0.2
−3
−2
−1
0
1
2
Theoretical
Phases
phase−0
phase−1
phase−2

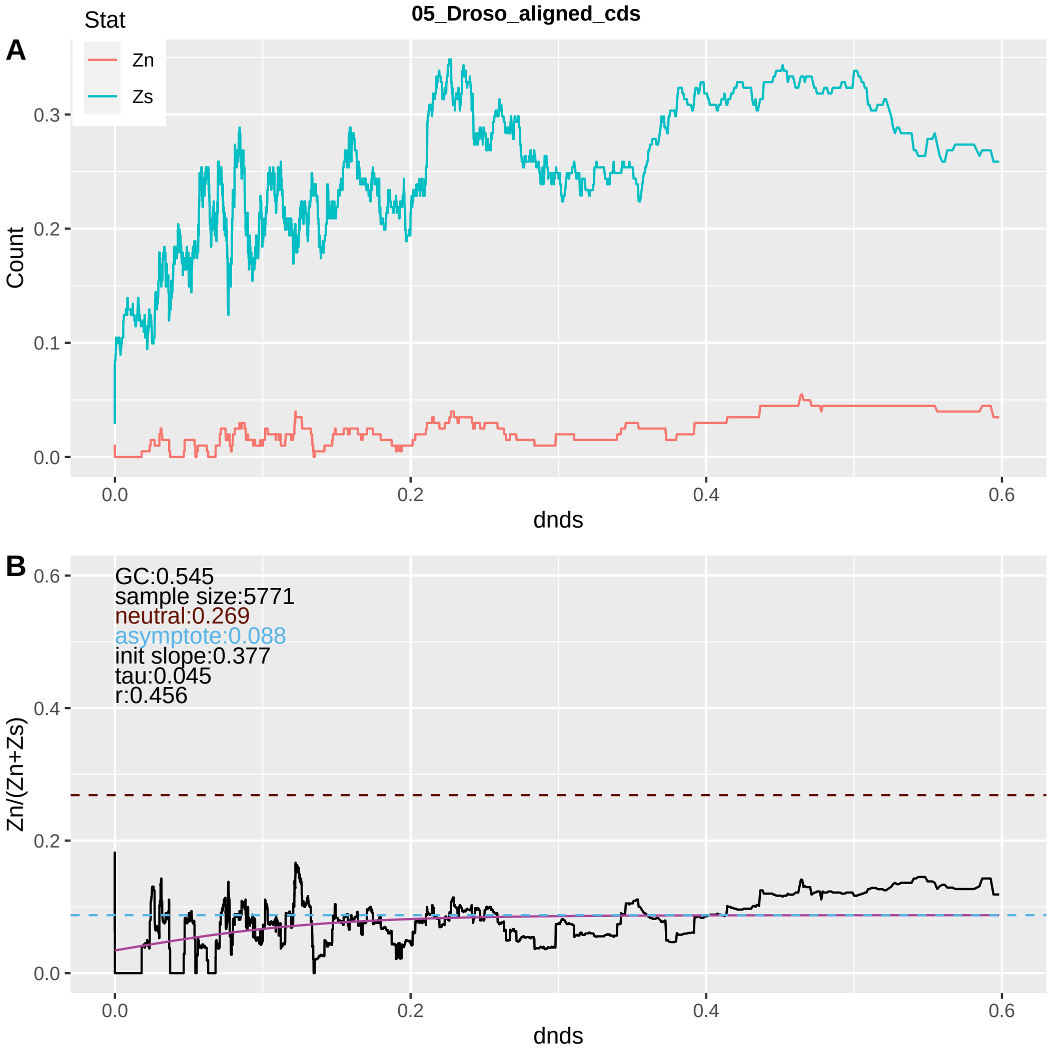
Figure D.2: **Histogram and QQplot of Phase Proportions of 90 Species |** Upper figure: The dashed line indicates the mean of each distribution, the thick curve is the shape of normal distribution. Lower figure: QQplot, the solid line is the straight diagonal line of each sample distribution.

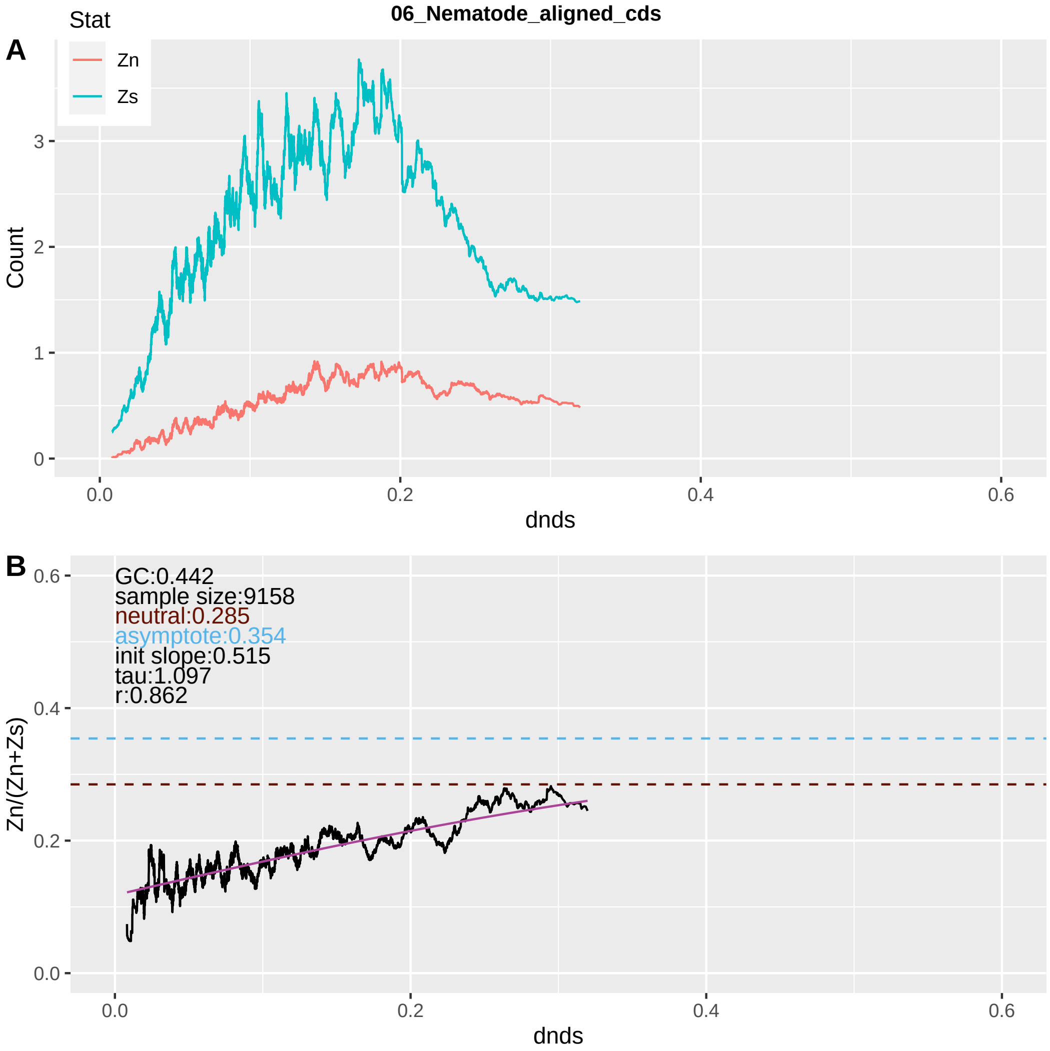
Correlation Plot between Dnds and Zn/(Zn+Zs) of 90 Species

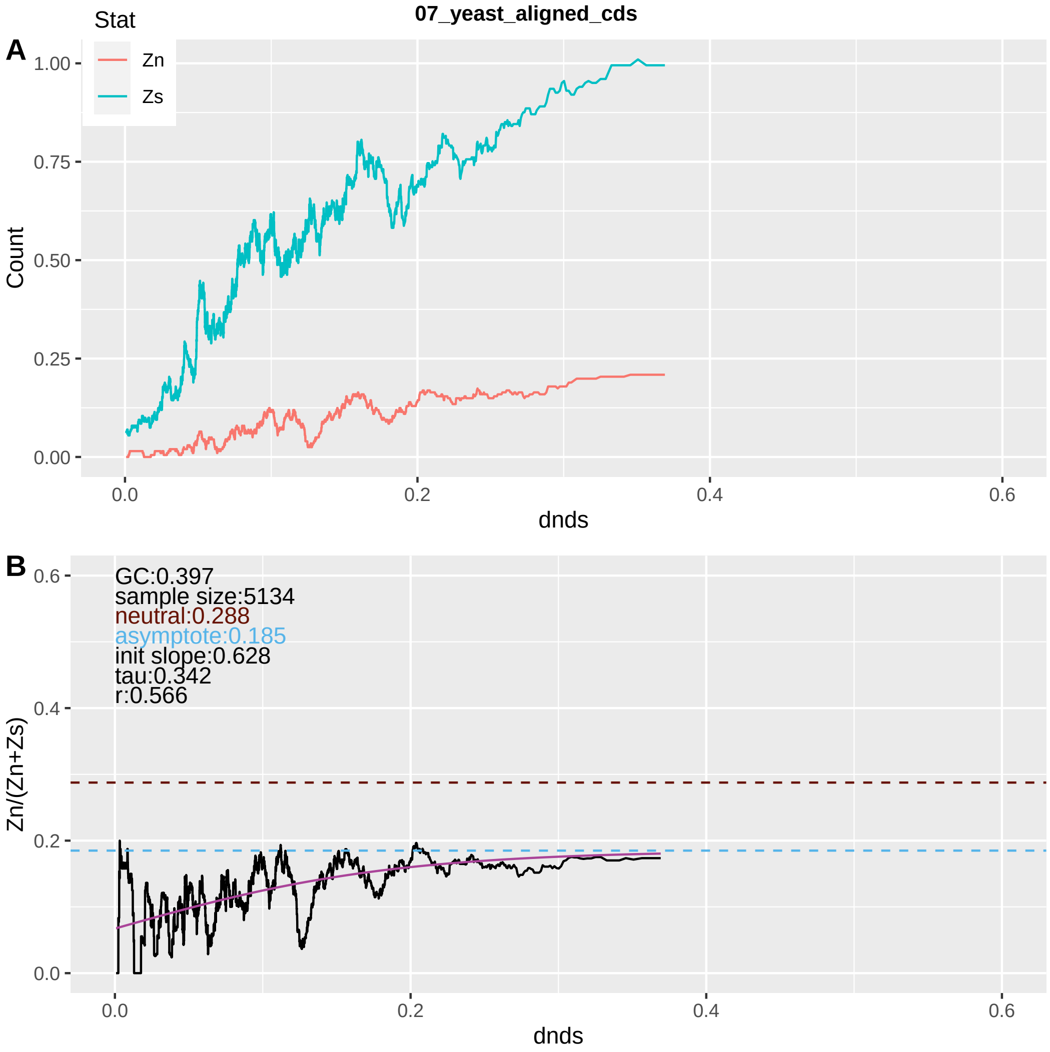

01_FcaCaf_aligned_cds
A
Stat
Zn
Zs
Count
2
1
0
0.0
0.2
0.4
0.6
dnds
B
GC:0.527
sample size:11036
neutral:0.274
asymptote:0.185
init slope:0.371
tau:0.202
r:0.864
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

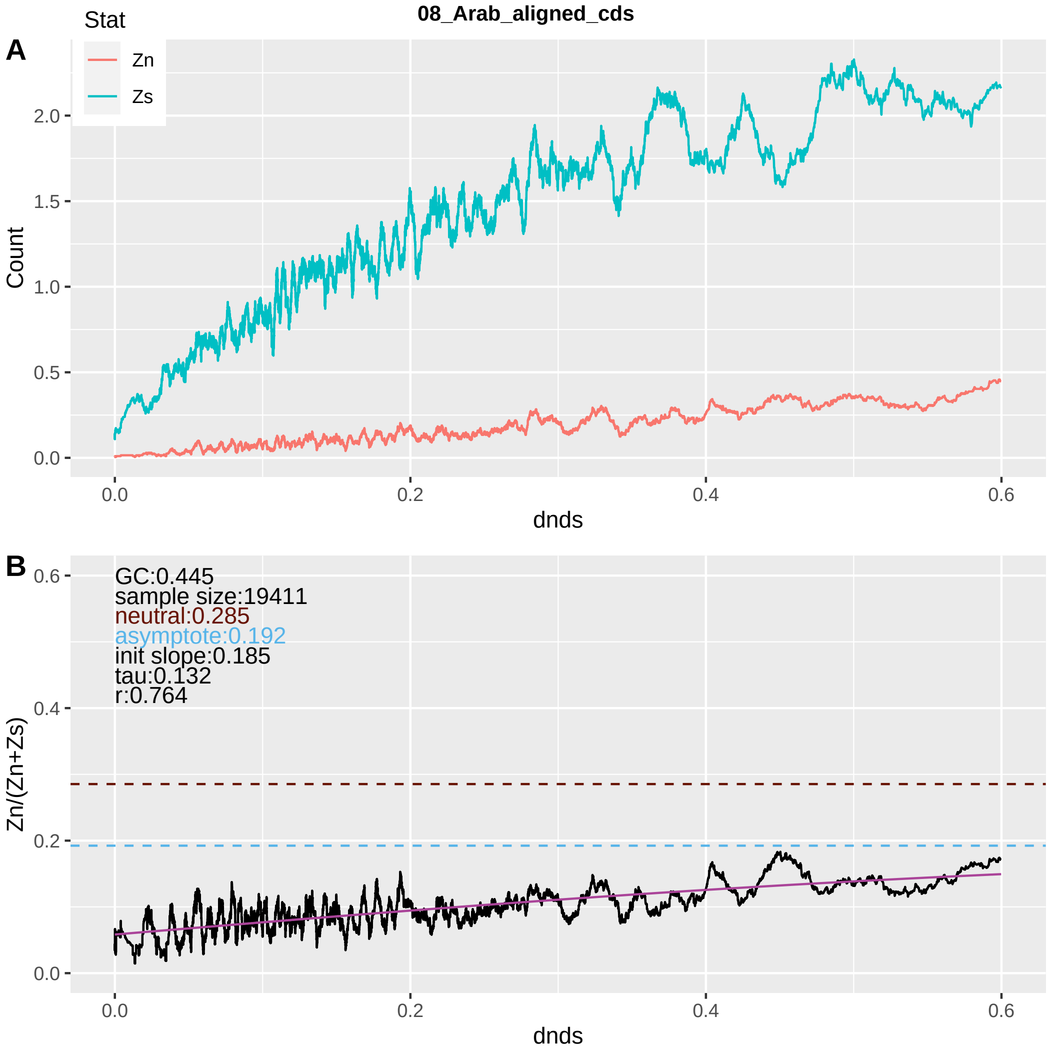

02_MusRat_aligned_cds
A
Stat
Zn
Zs
Count
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
dnds
B
GC:0.527
sample size:12255
neutral:0.275
asymptote:0.159
init slope:0.360
tau:0.174
r:0.679
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

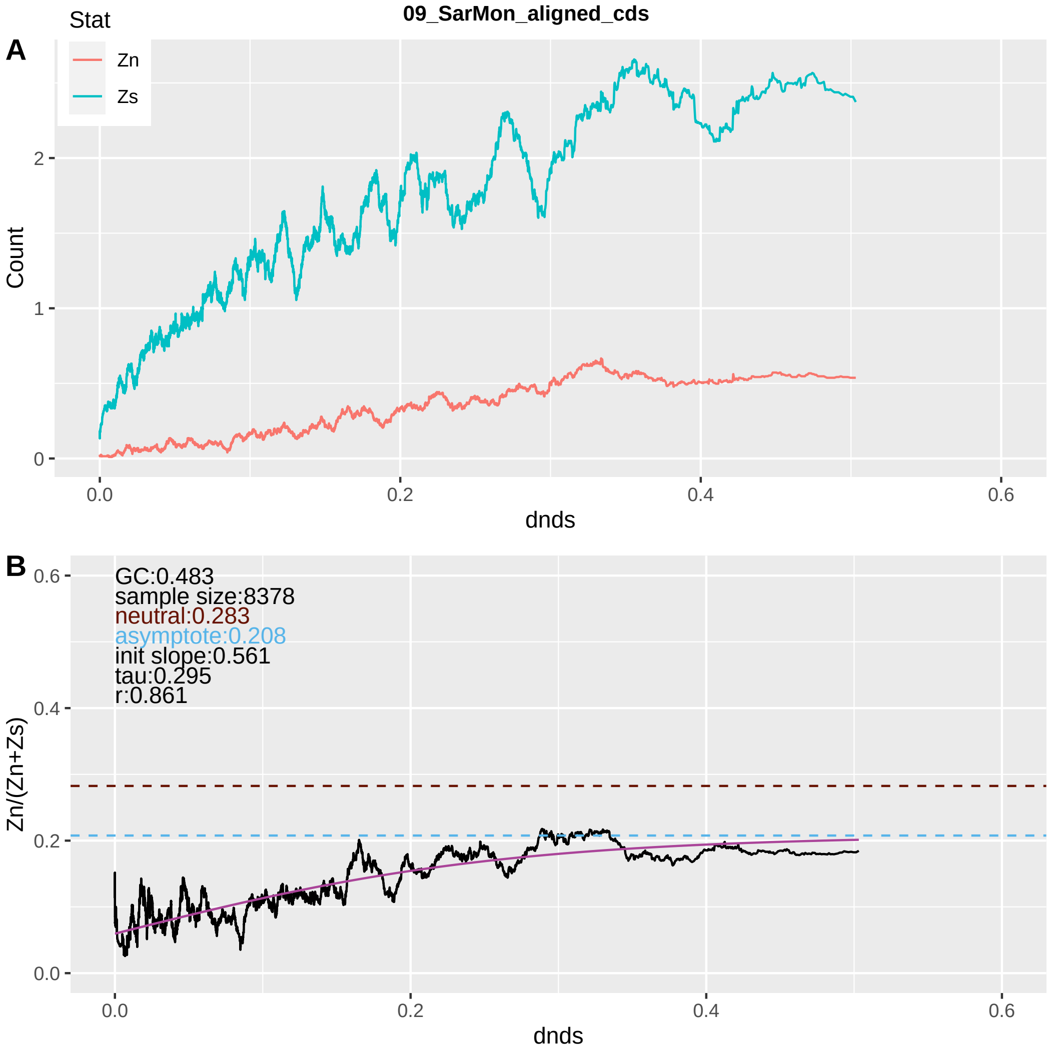

04_Gallus_aligned_cds
A
Stat
Zn
Zs
Count
1.0
0.5
0.0
0.0
0.2
0.4
0.6
dnds
B
GC:0.493
sample size:9455
neutral:0.282
asymptote:0.207
init slope:0.116
tau:0.093
r:0.462
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

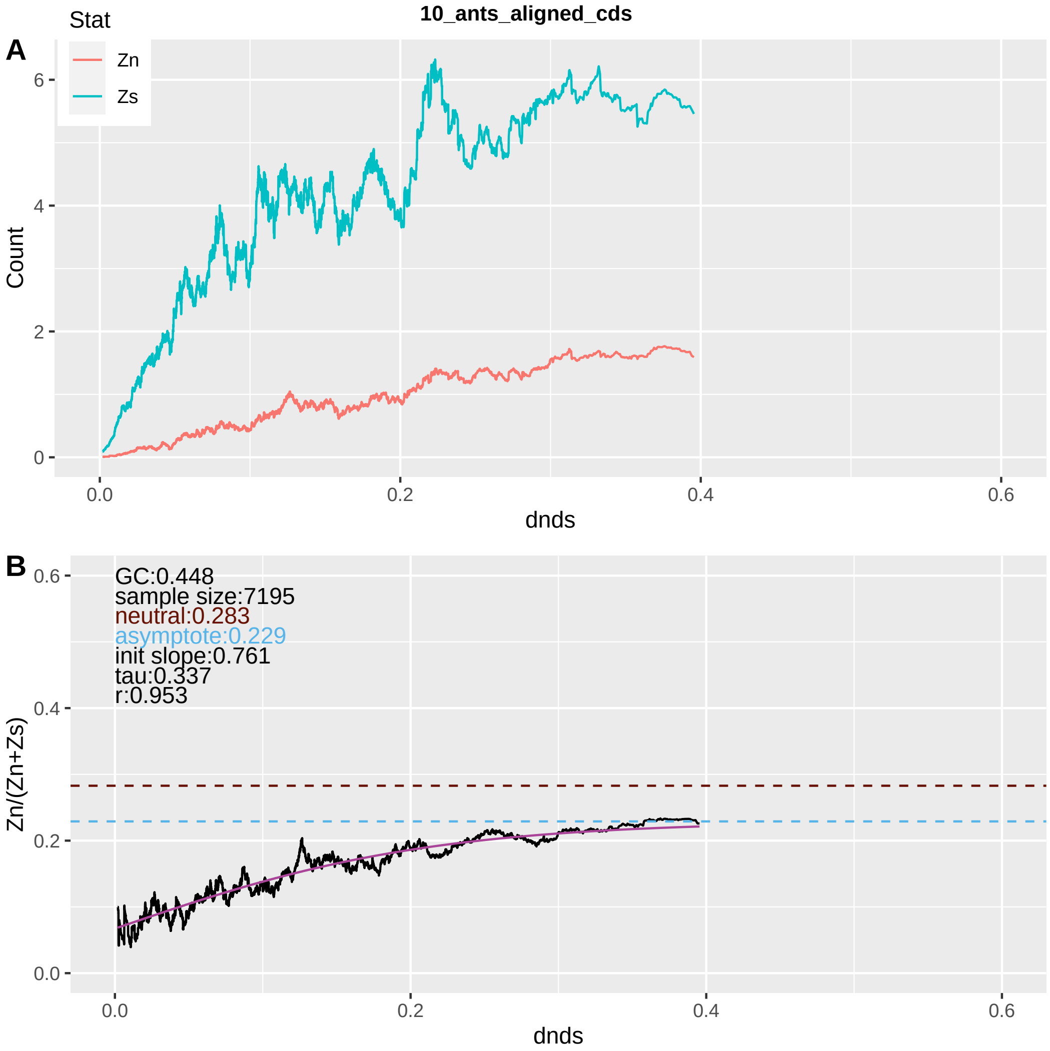

05_Droso_aligned_cds
A
Stat
Zn
Zs
Count
0.0
0.1
0.2
0.3
0.0
0.2
0.4
0.6
dnds
B
GC:0.545
sample size:5771
neutral:0.269
asymptote:0.088
init slope:0.377
tau:0.045
r:0.456
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

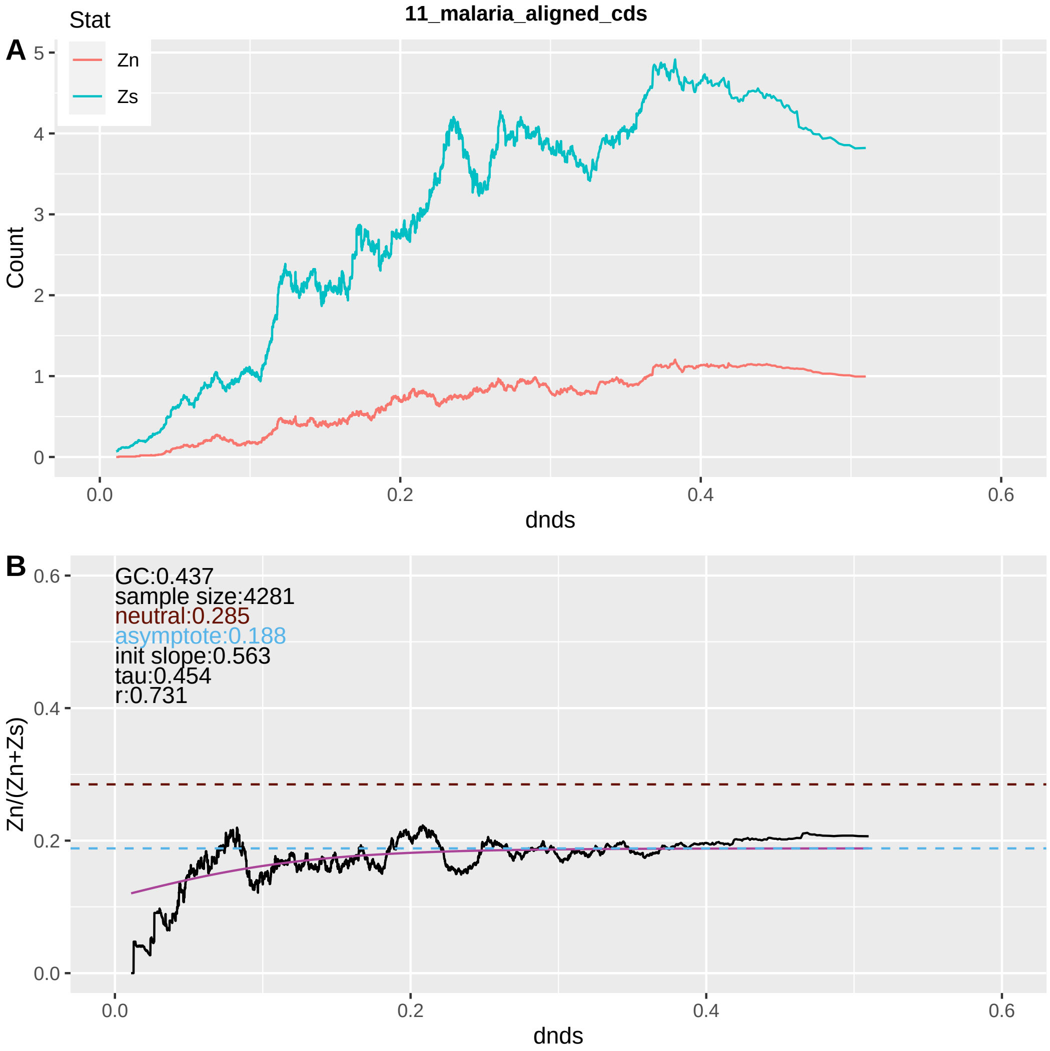

06_Nematode_aligned_cds
A
Stat
Zn
Zs
Count
0
1
2
3
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
GC:0.442
sample size:9158
neutral:0.285
asymptote:0.354
init slope:0.515
tau:1.097
r:0.862
0.0
0.2
0.4
0.6
dnds

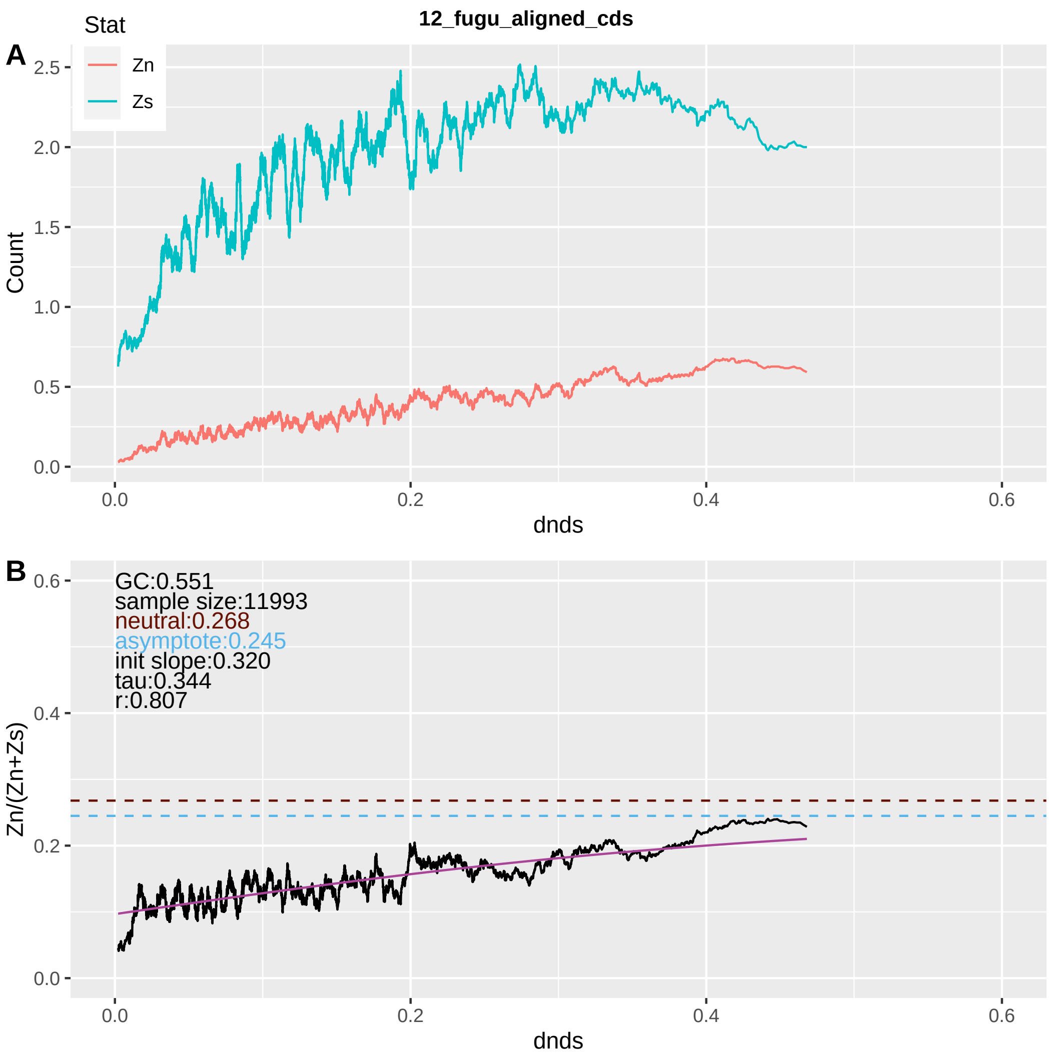

07_yeast_aligned_cds
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.397
sample size:5134
neutral:0.288
asymptote:0.185
init slope:0.628
tau:0.342
r:0.566
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

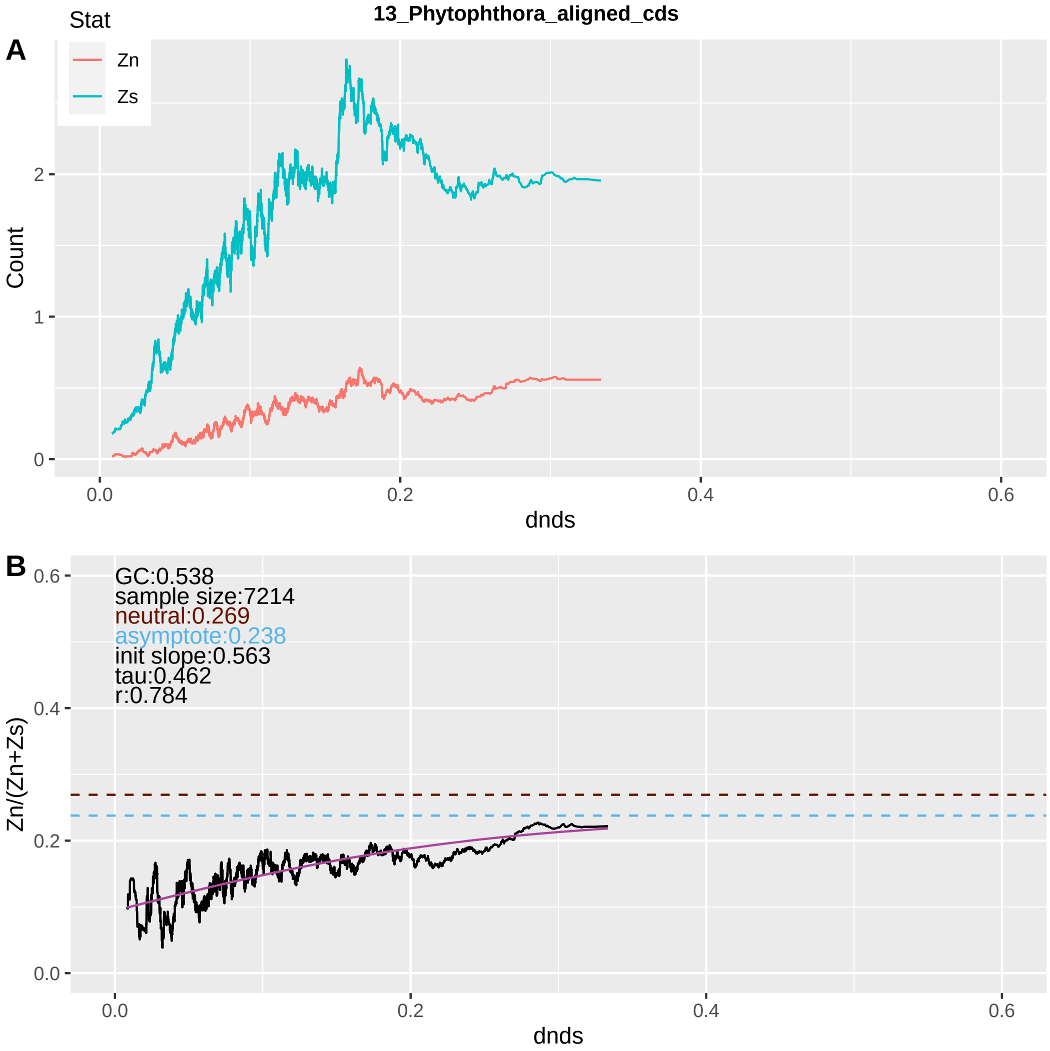

08_Arab_aligned_cds
A
Stat
Zn
Zs
2.0
1.5
1.0
0.5
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
GC:0.445
sample size:19411
neutral:0.285
asymptote:0.192
init slope:0.185
tau:0.132
r:0.764
0.0
0.2
0.4
0.6
dnds

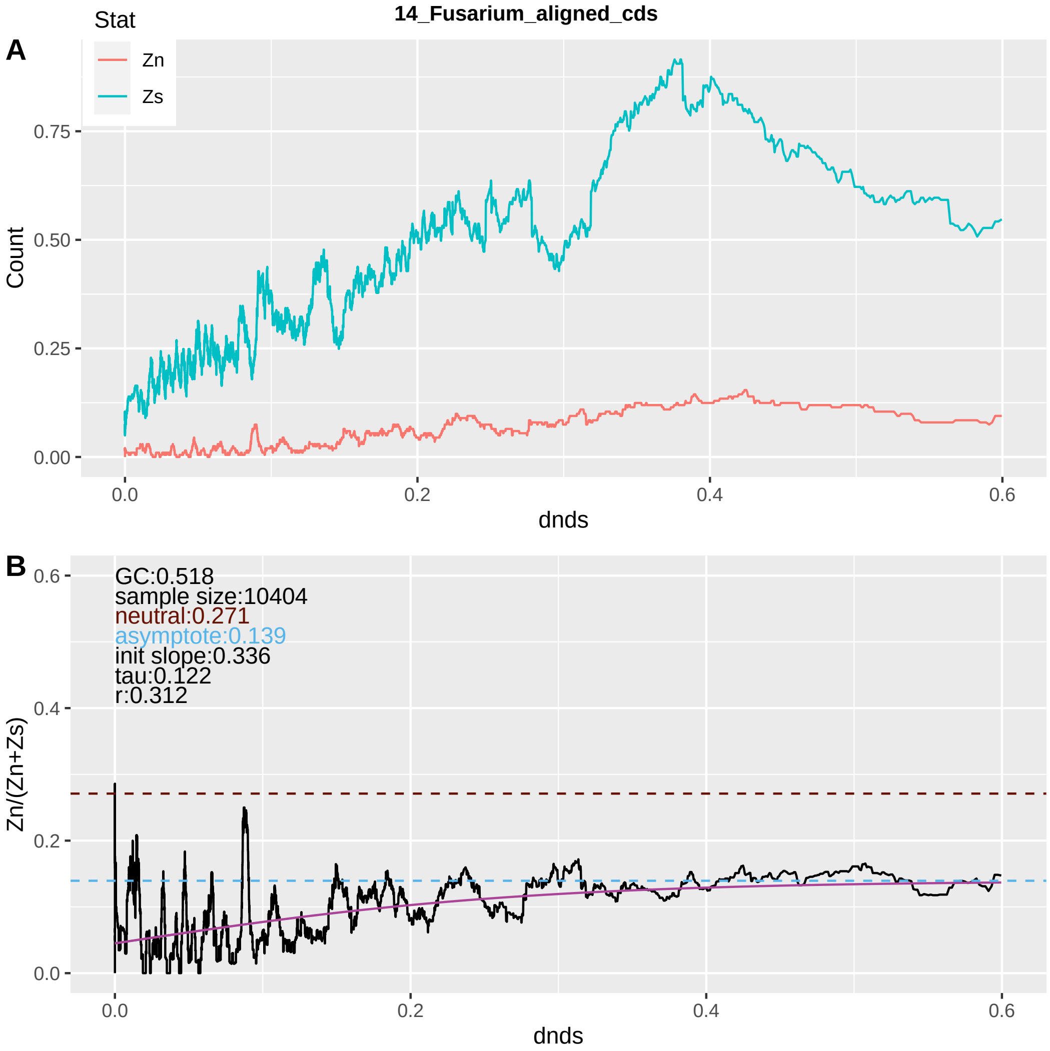

09_SarMon_aligned_cds
A
Stat
Zn
Zs
Count
0
1
2
0.0
0.2
0.4
0.6
dnds
B
GC:0.483
sample size:8378
neutral:0.283
asymptote:0.208
init slope:0.561
tau:0.295
r:0.861
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

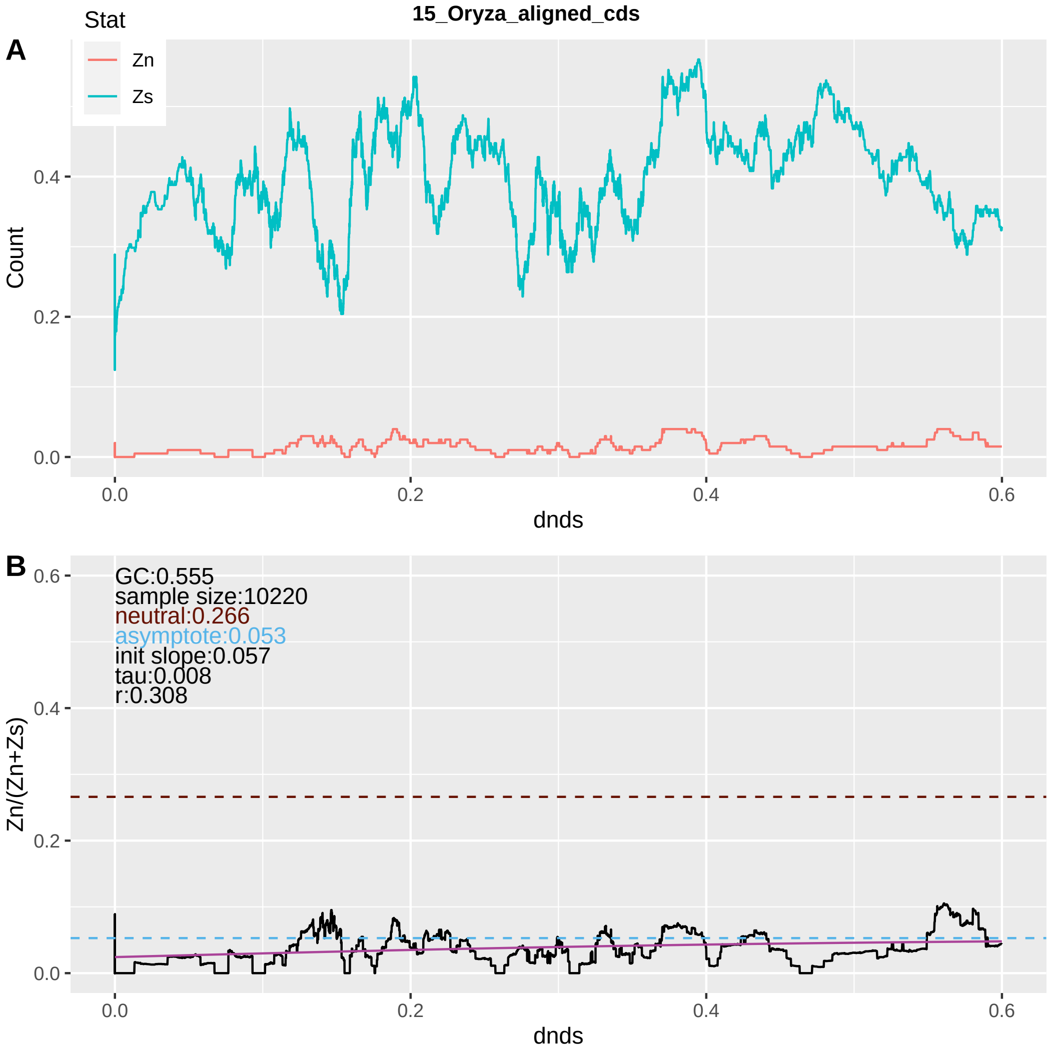

10_ants_aligned_cds
A
Stat
Zn
Zs
Count
0
2
4
6
0.0
0.2
0.4
0.6
dnds
B
GC:0.448
sample size:7195
neutral:0.283
asymptote:0.229
init slope:0.761
tau:0.337
r:0.953
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

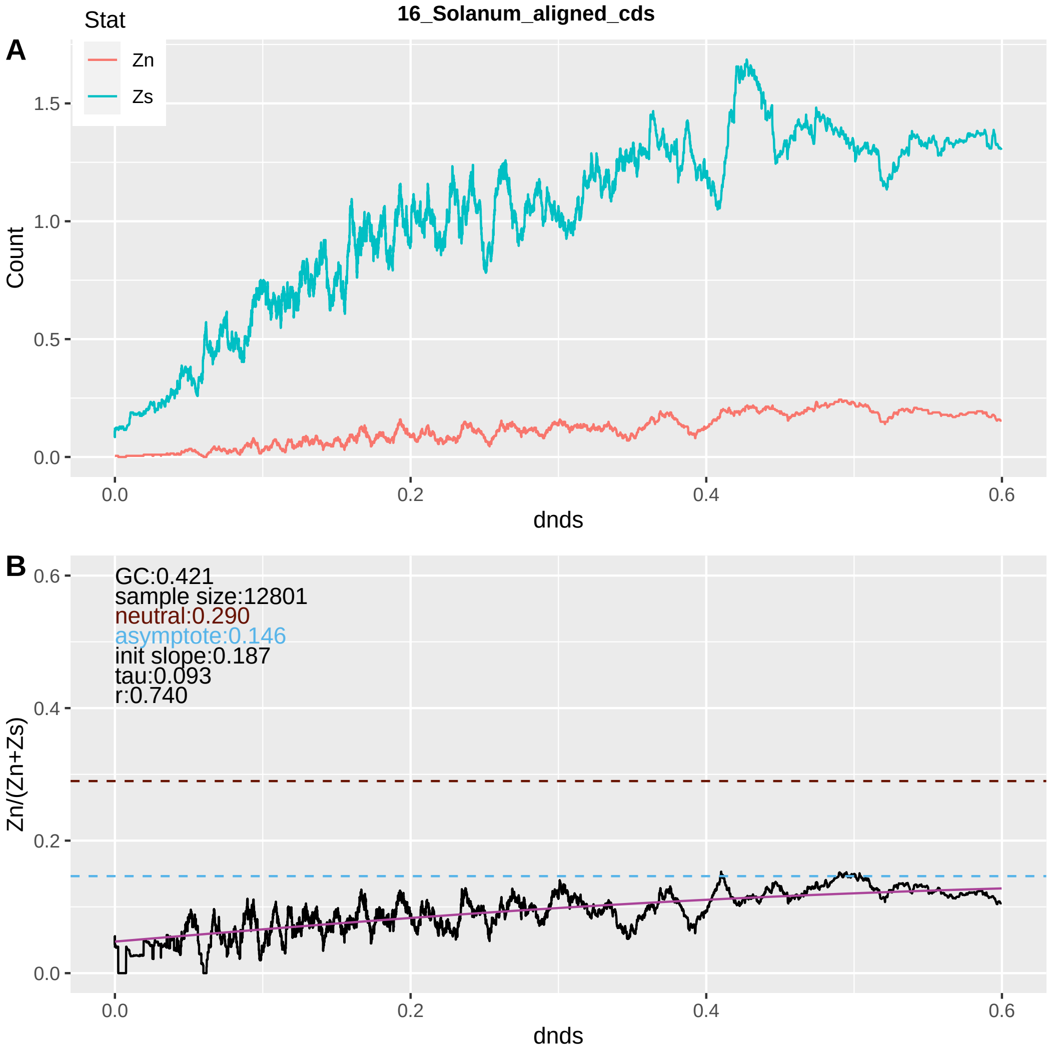

11_malaria_aligned_cds
Stat
Zn
Zs
A
Count
5
4
3
2
1
0
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
GC:0.437
sample size:4281
neutral:0.285
asymptote:0.188
init slope:0.563
tau:0.454
r:0.731
0.0
0.2
0.4
0.6
dnds

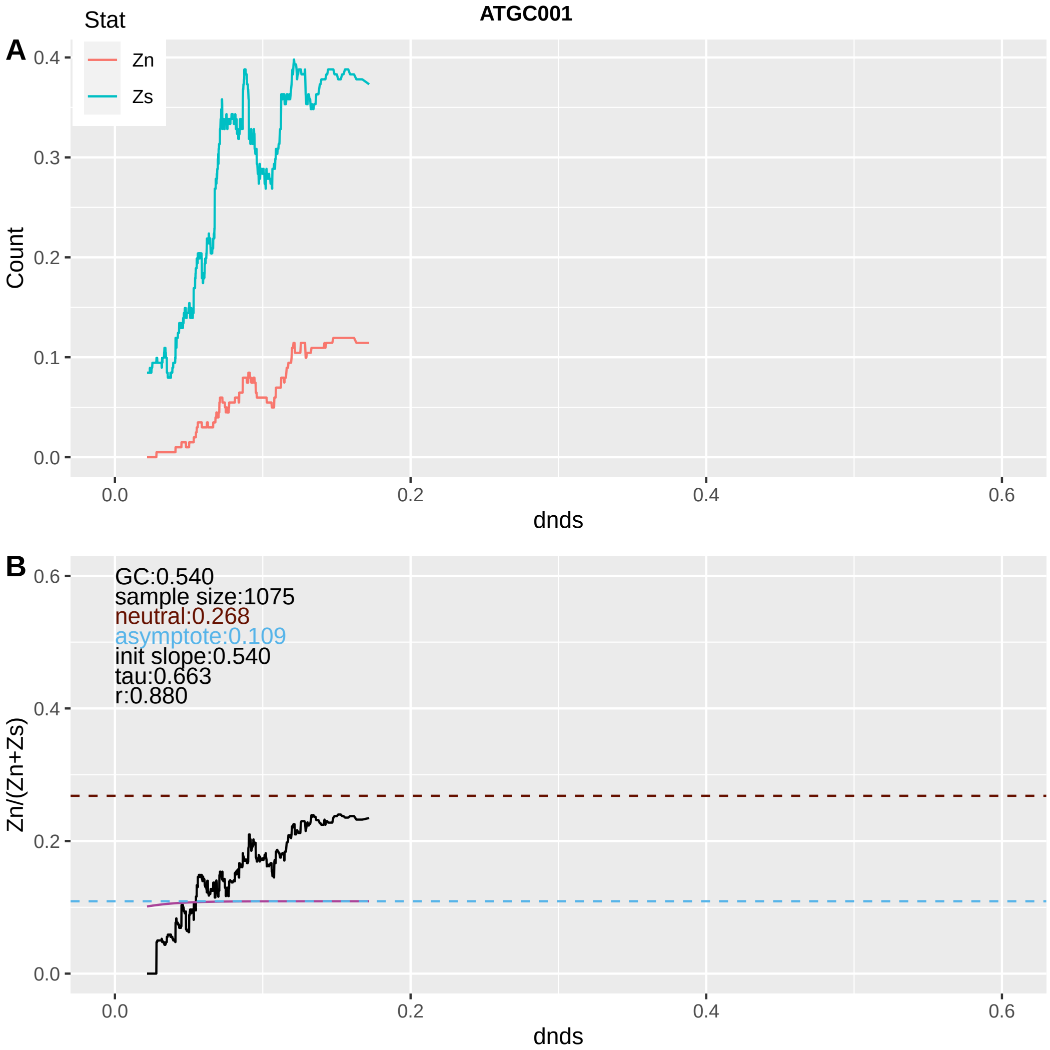

12_fugu_aligned_cds
A
Stat
Zn
Zs
Count
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
dnds
B
GC:0.551
sample size:11993
neutral:0.268
asymptote:0.245
init slope:0.320
tau:0.344
r:0.807
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

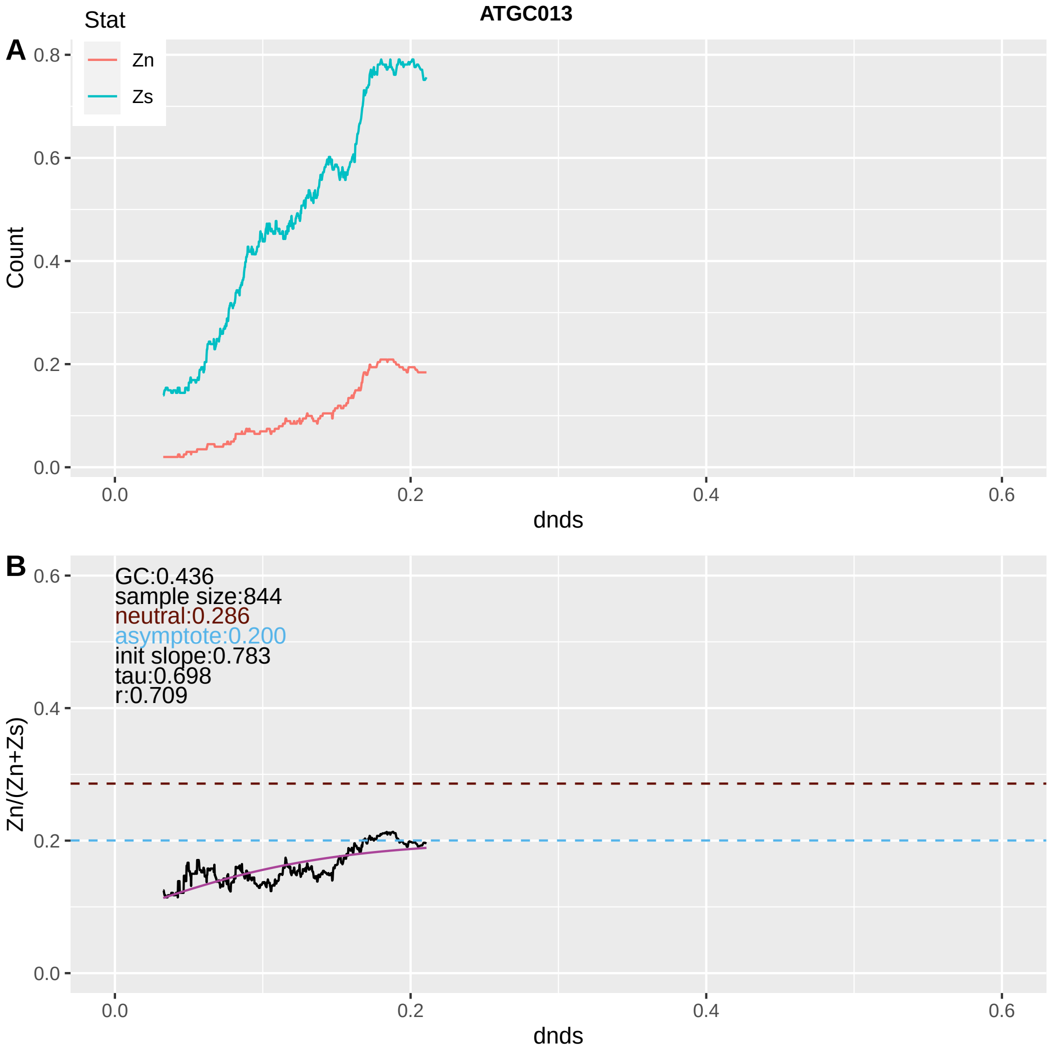

13_Phytophthora_aligned_cds
A
Stat
Zn
Zs
Count
0
1
2
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
GC:0.538
sample size:7214
neutral:0.269
asymptote:0.238
init slope:0.563
tau:0.462
r:0.784
0.0
0.2
0.4
0.6
dnds

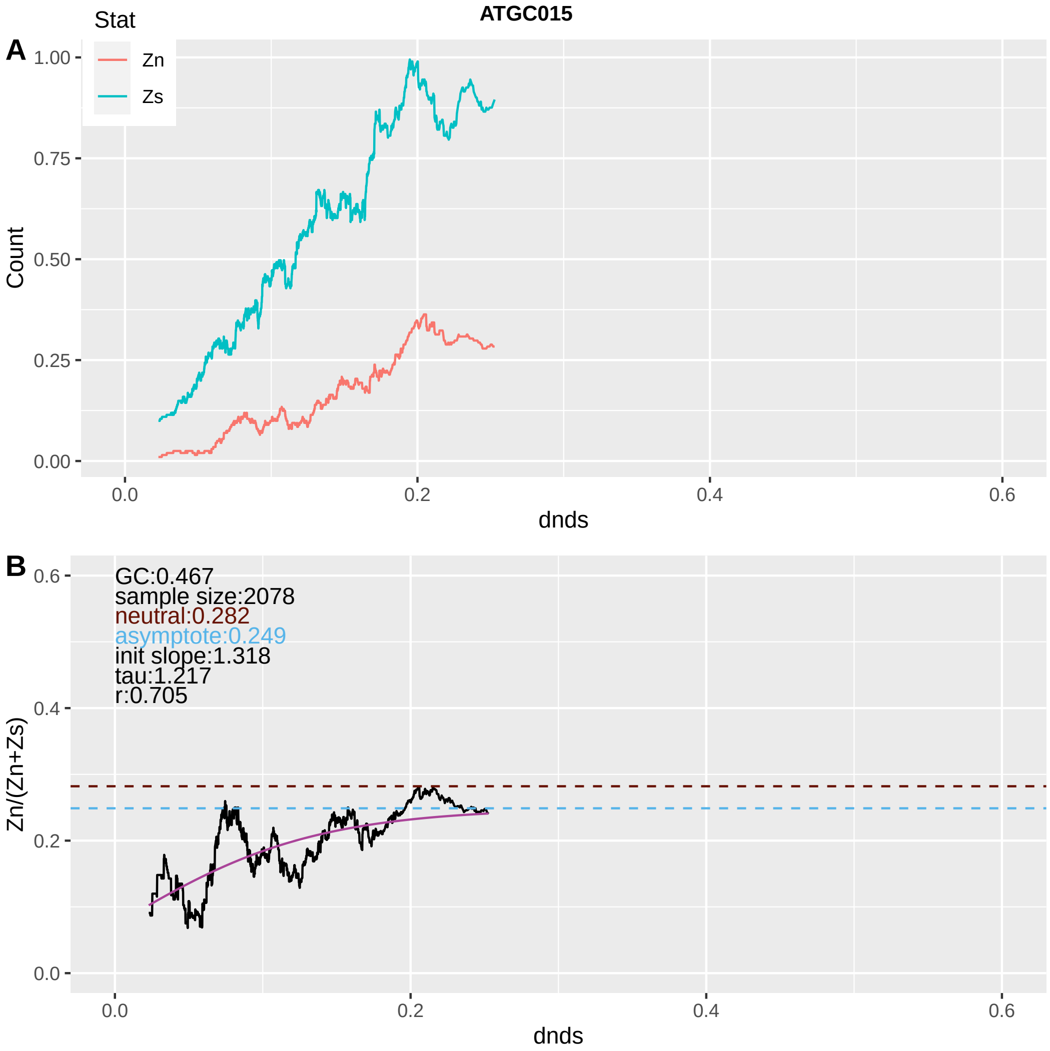

14_Fusarium_aligned_cds
A
Stat
Zn
Zs
Count
0.00
0.25
0.50
0.75
0.0
0.2
0.4
0.6
dnds
B
GC:0.518
sample size:10404
neutral:0.271
asymptote:0.139
init slope:0.336
tau:0.122
r:0.312
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

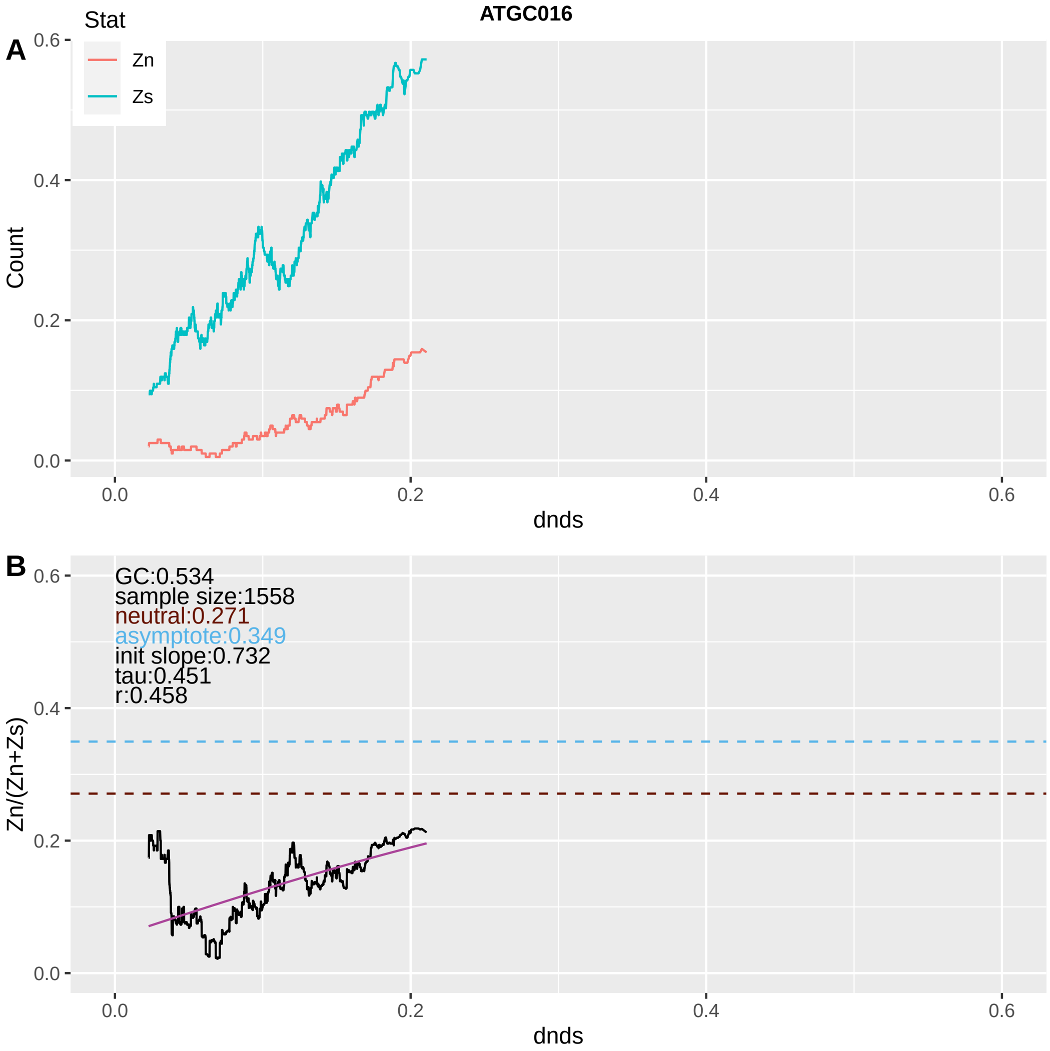

15_Oryza_aligned_cds
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.0
0.2
0.4
0.6
dnds
B
GC:0.555
sample size:10220
neutral:0.266
asymptote:0.053
init slope:0.057
tau:0.008
r:0.308
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

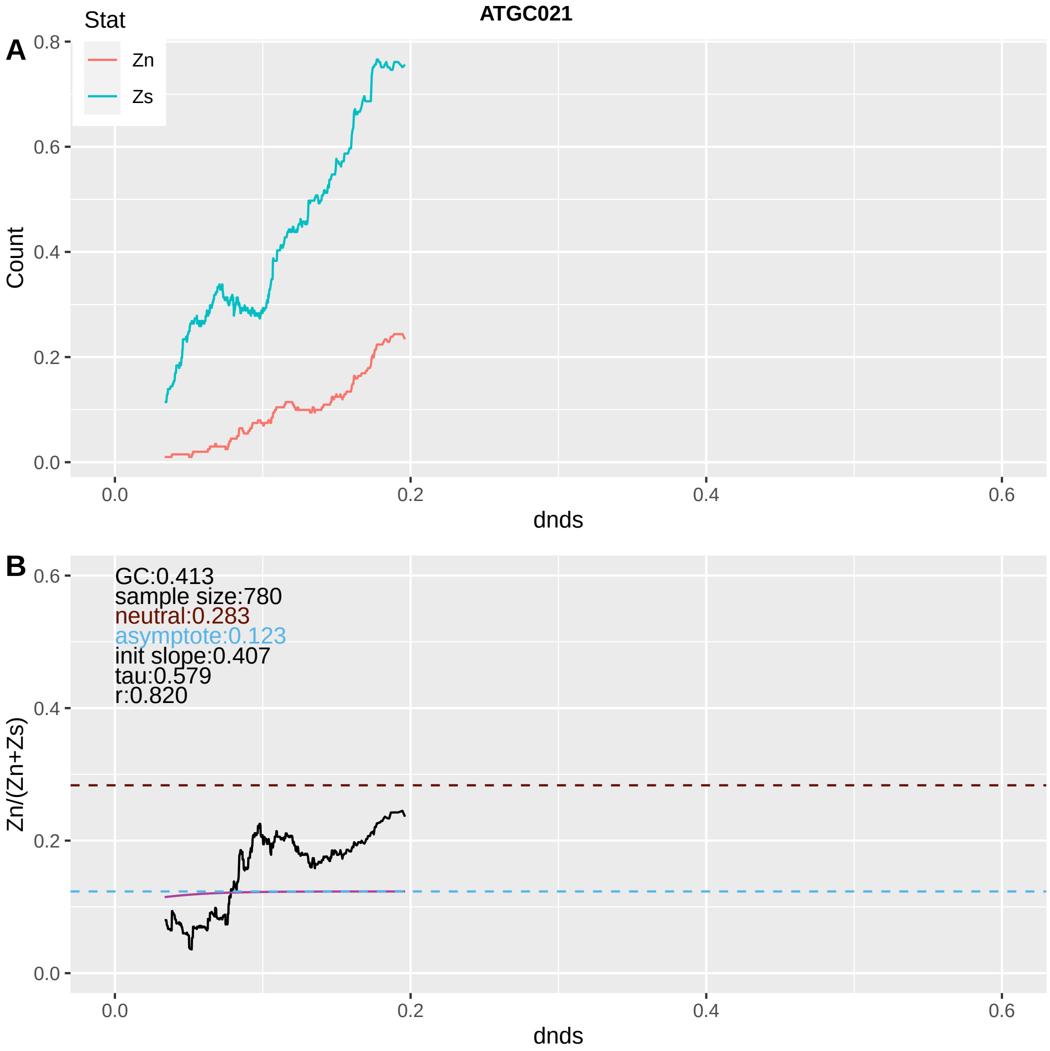

16_Solanum_aligned_cds
Stat
Zn
Zs
A
Count
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
GC:0.421
sample size:12801
neutral:0.290
asymptote:0.146
init slope:0.187
tau:0.093
r:0.740
0.0
0.2
0.4
0.6
dnds

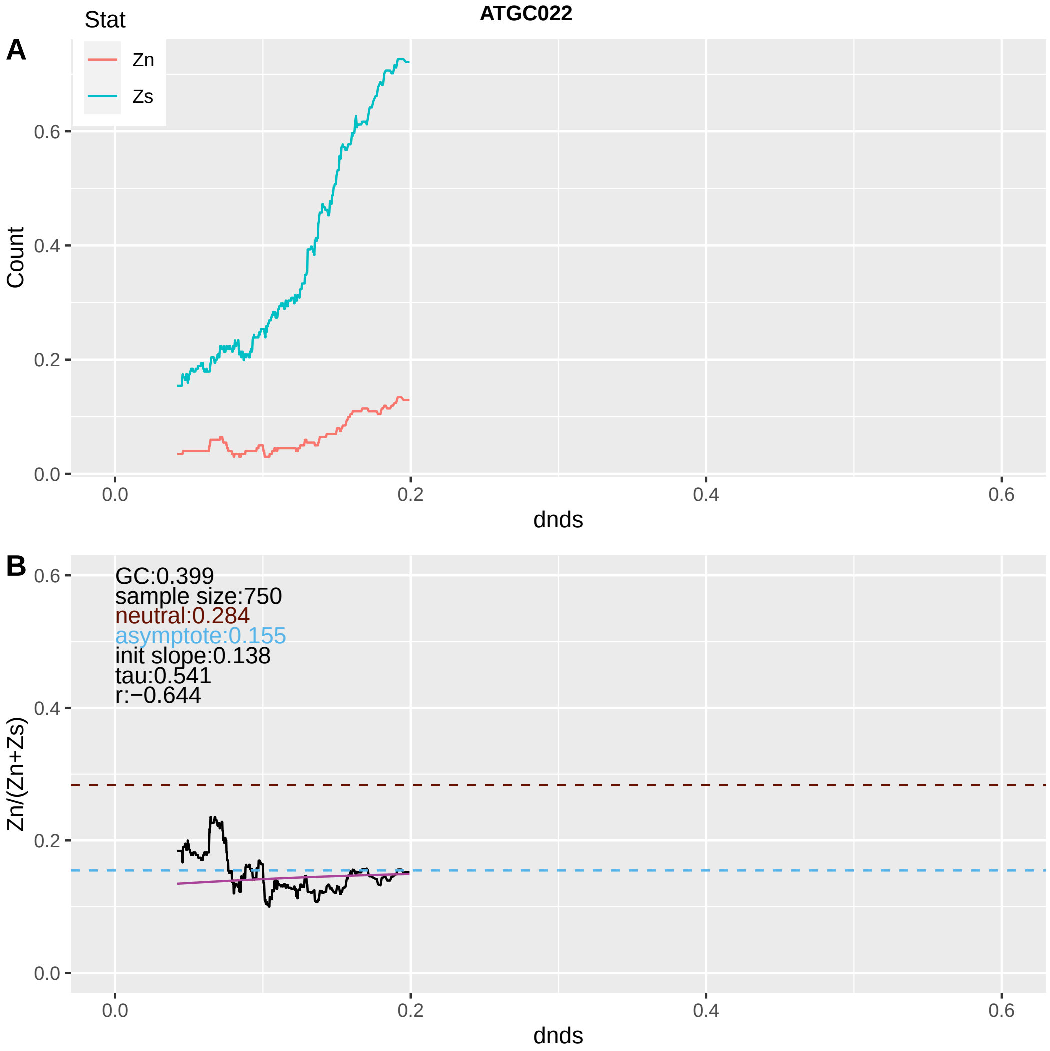

ATGC001
A
Stat
Zn
Zs
Count
0.0
0.1
0.2
0.3
0.4
0.0
0.2
0.4
0.6
dnds
B
GC:0.540
sample size:1075
neutral:0.268
asymptote:0.109
init slope:0.540
tau:0.663
r:0.880
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

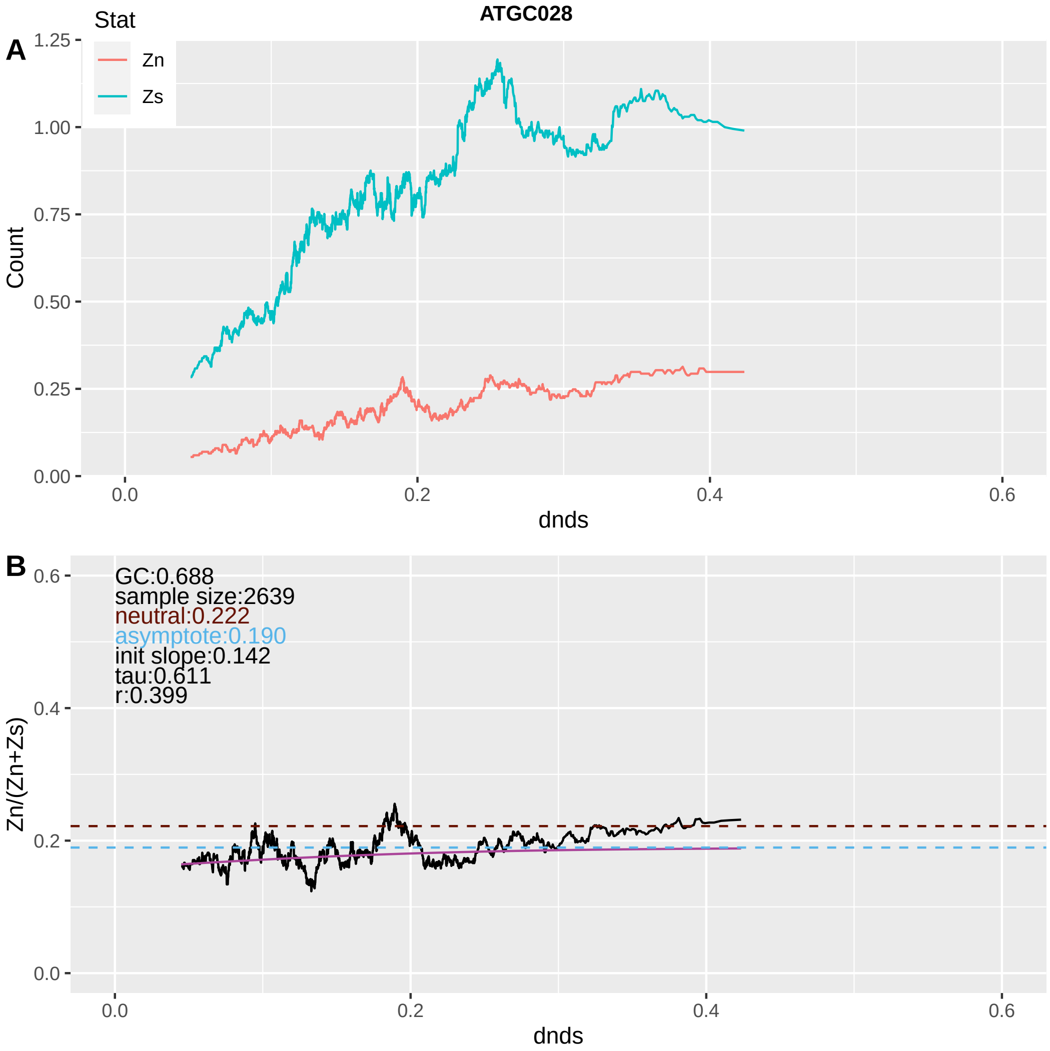

ATGC013
A
Stat
Zn
Zs
0.8
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.436
sample size:844
neutral:0.286
asymptote:0.200
init slope:0.783
tau:0.698
r:0.709
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

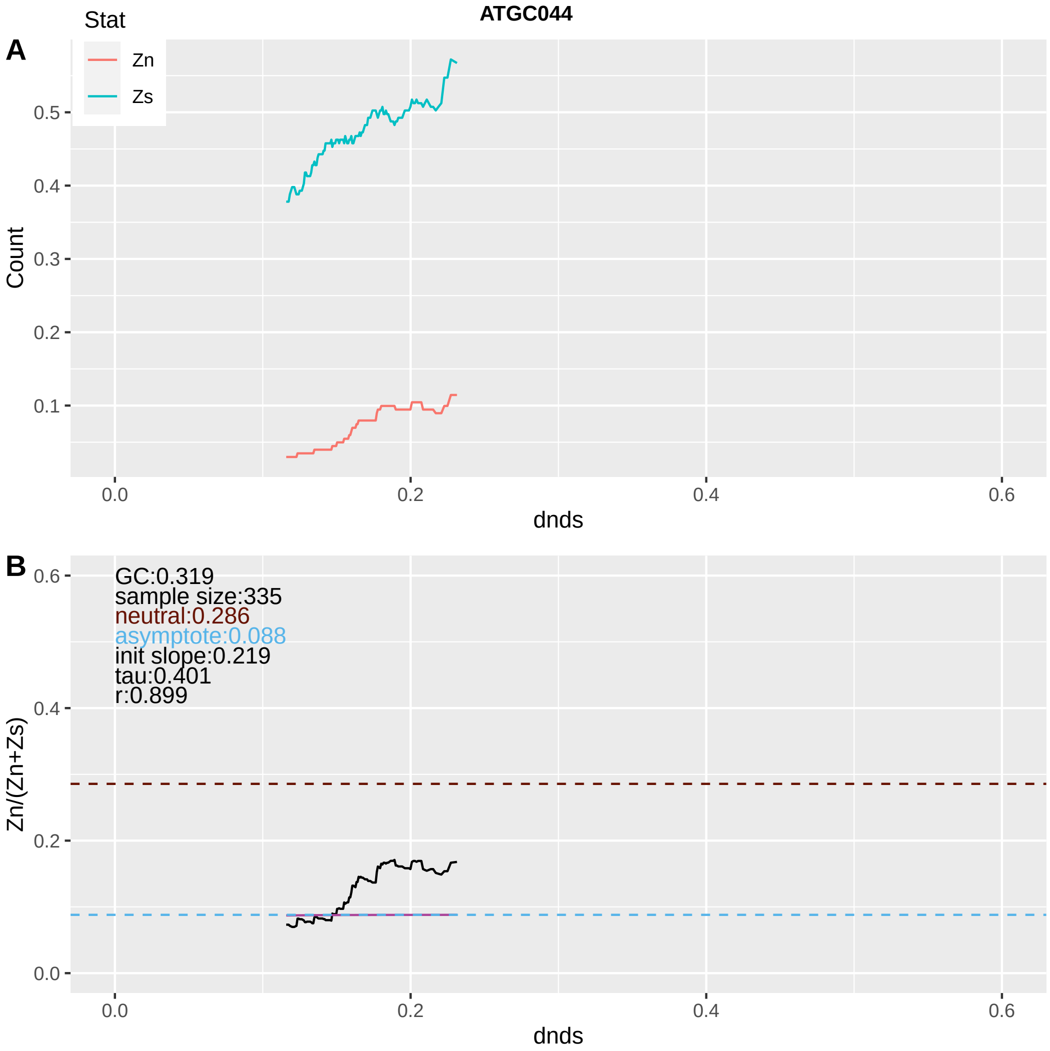

ATGC015
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.467
sample size:2078
neutral:0.282
asymptote:0.249
init slope:1.318
tau:1.217
r:0.705
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

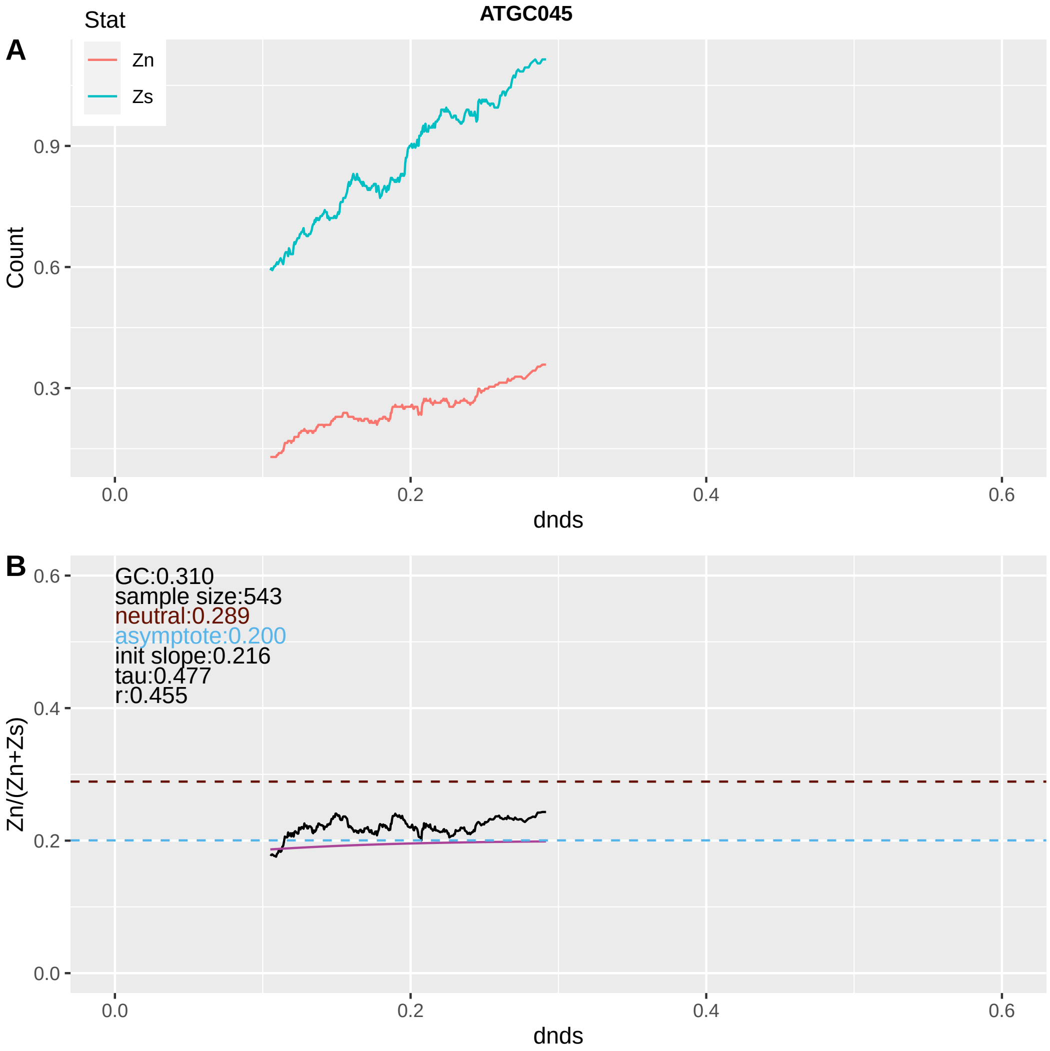

ATGC016
A
B
Stat
Zn
Zs
Count
dnds
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
GC:0.534
sample size:1558
neutral:0.271
asymptote:0.349
init slope:0.732
tau:0.451
r:0.458

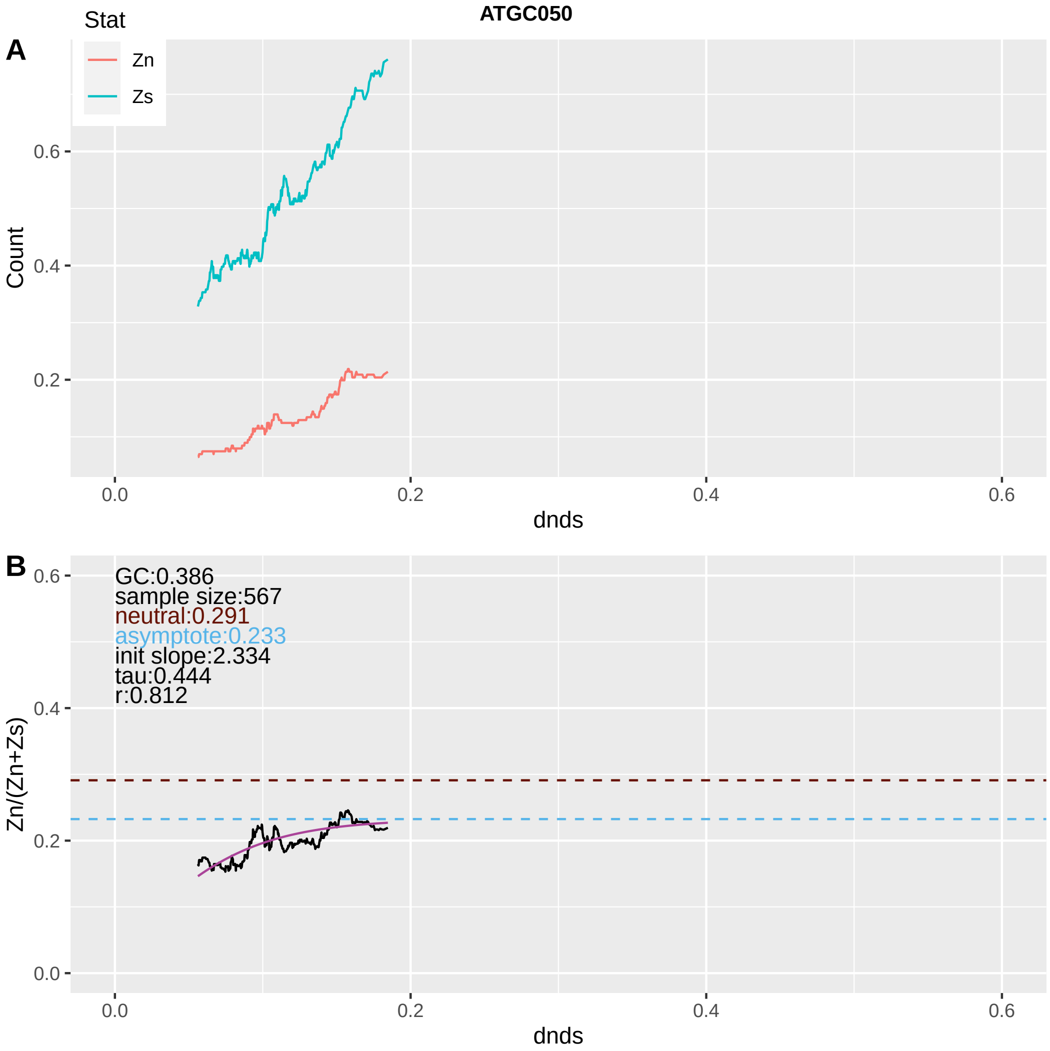

ATGC021
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
0.8
dnds
0.0
0.2
0.4
0.6
B
GC:0.413
sample size:780
neutral:0.283
asymptote:0.123
init slope:0.407
tau:0.579
r:0.820
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6

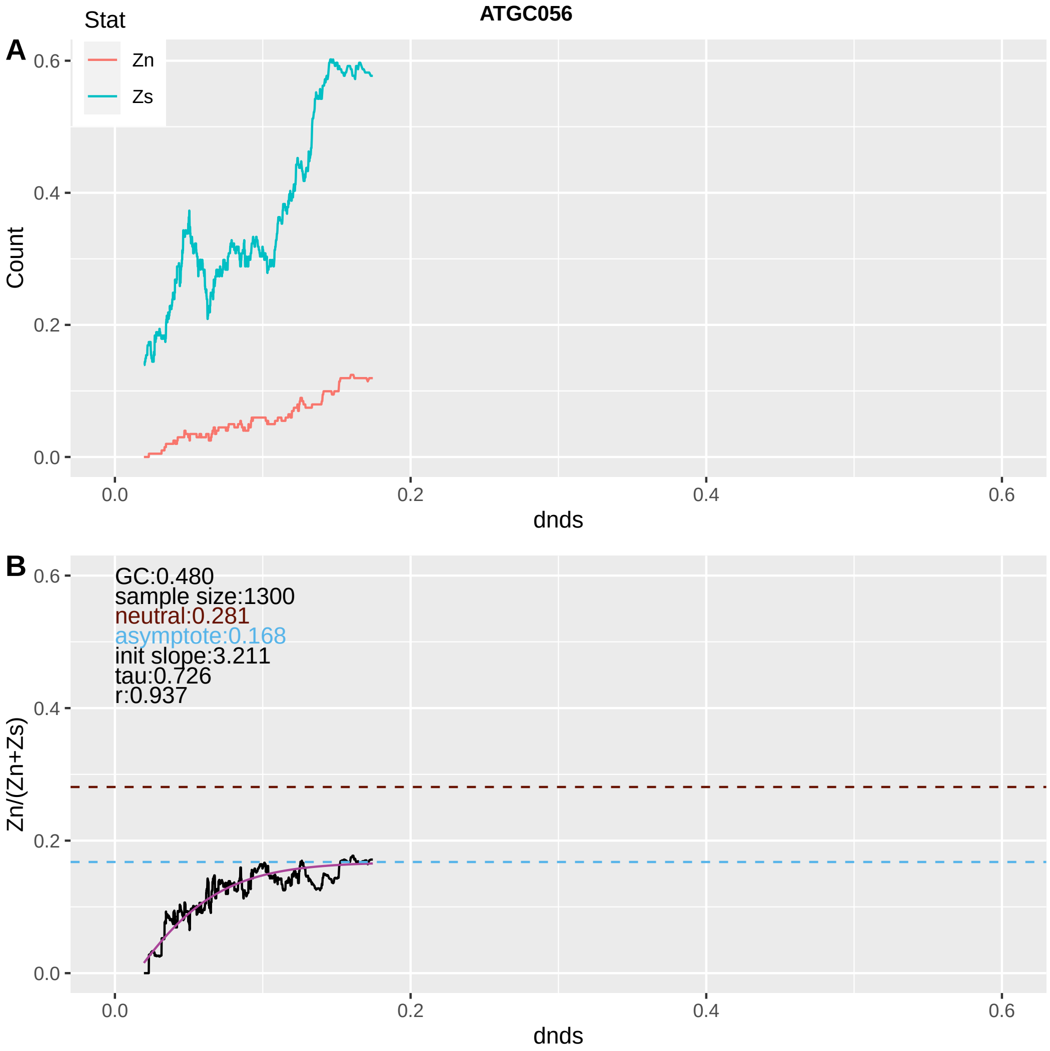

ATGC022
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds
B
GC:0.399
sample size:750
neutral:0.284
asymptote:0.155
init slope:0.138
tau:0.541
r:−0.644
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

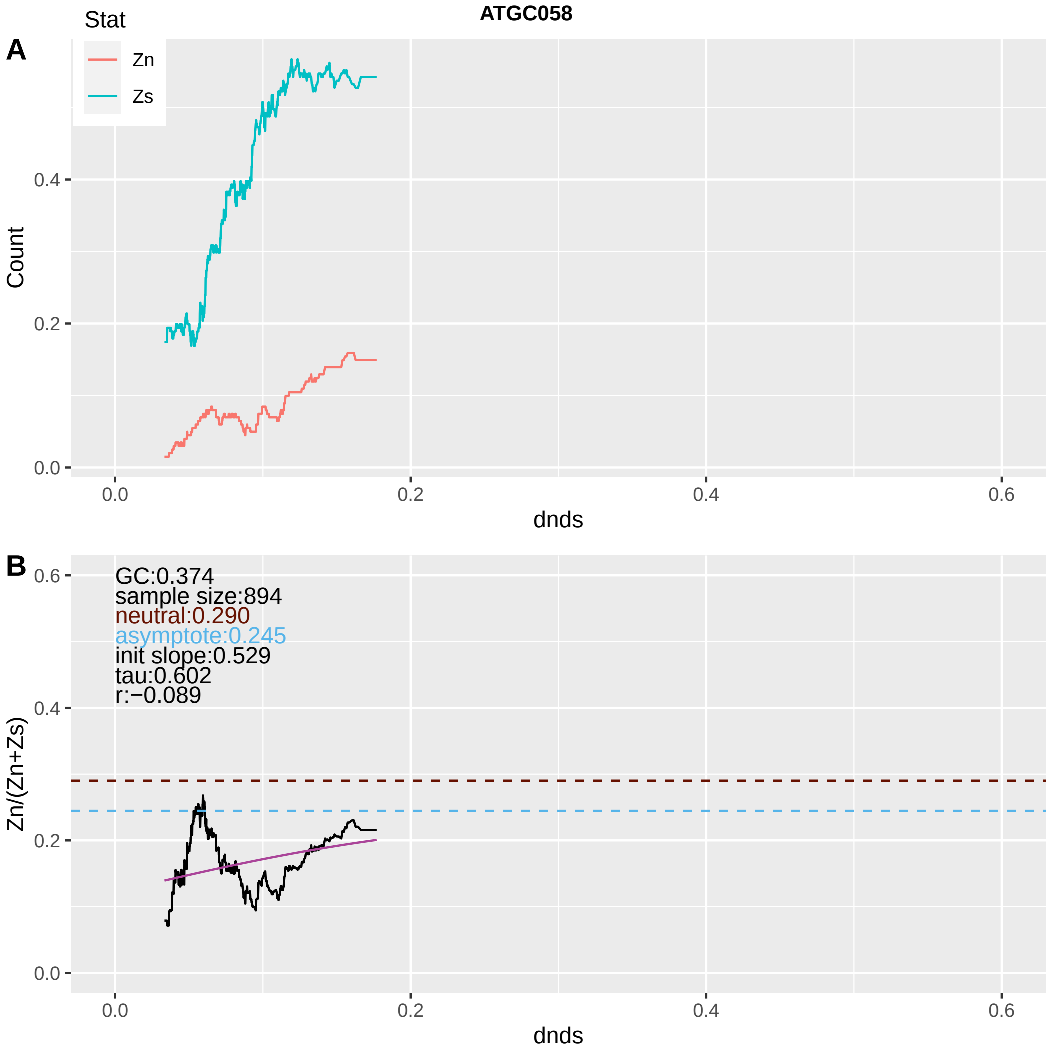

ATGC028
A
Stat
Zn
Zs
Count
1.25
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.688
sample size:2639
neutral:0.222
asymptote:0.190
init slope:0.142
tau:0.611
r:0.399
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

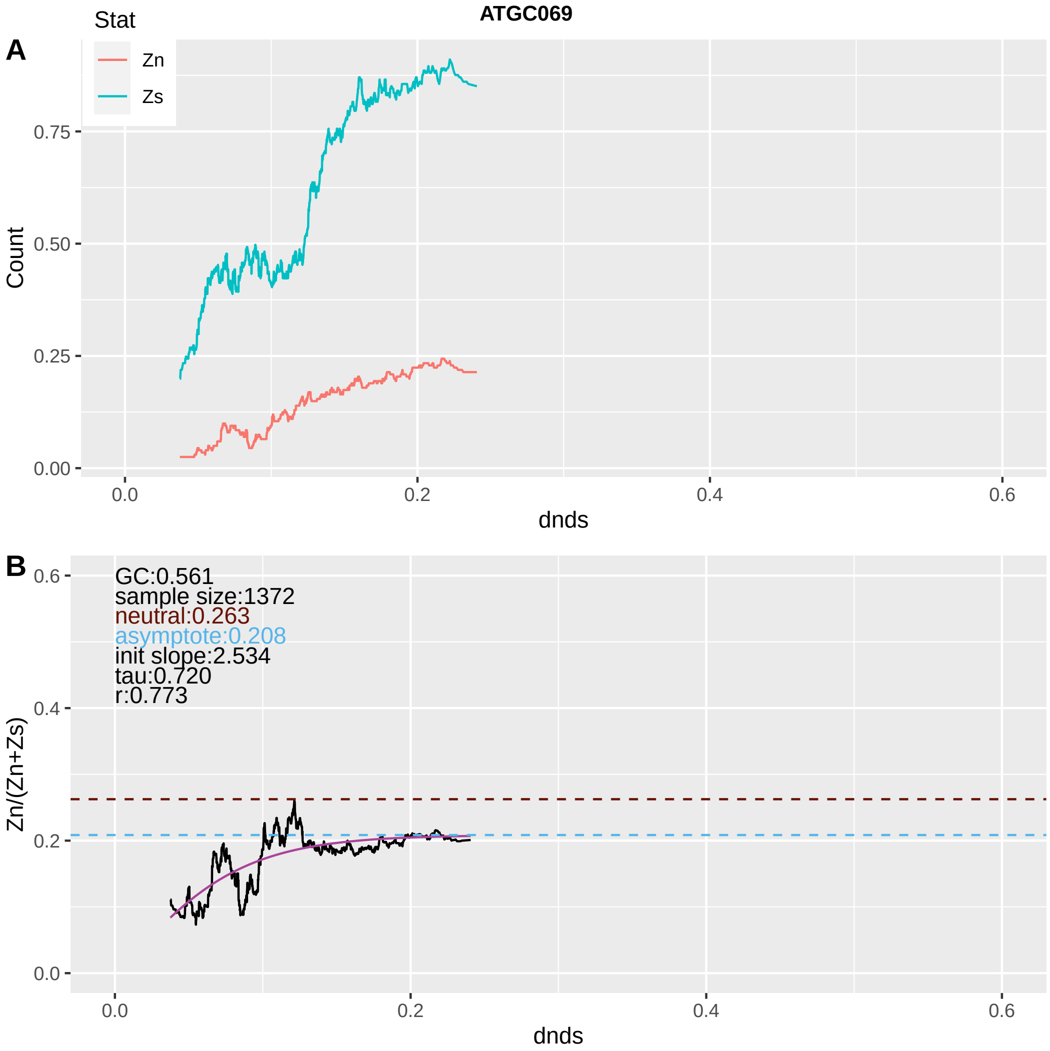

ATGC044
A
Stat
Zn
Zs
0.5
0.4
0.3
0.2
0.1
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
GC:0.319
sample size:335
neutral:0.286
asymptote:0.088
init slope:0.219
tau:0.401
r:0.899
0.0
0.2
0.4
0.6
dnds

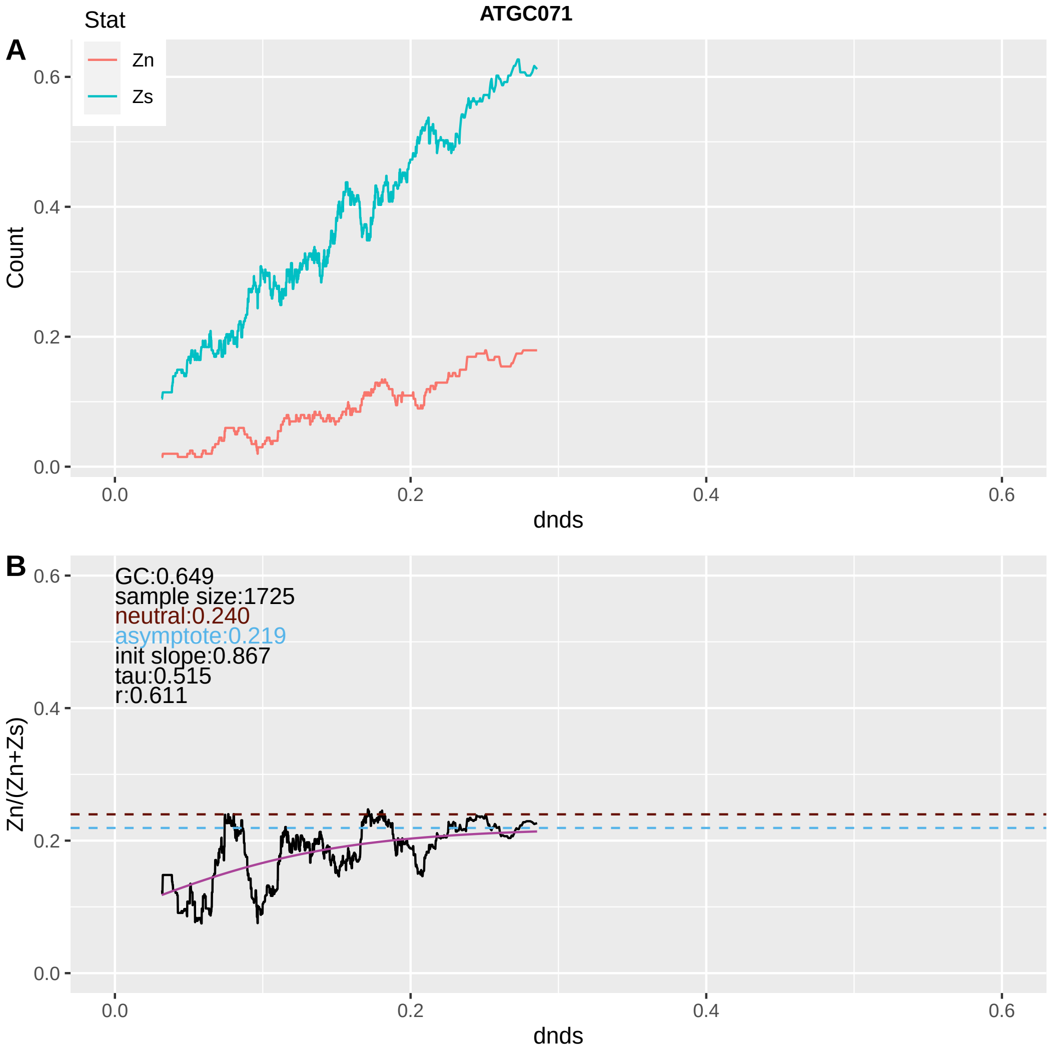

ATGC045
A
Stat
Zn
Zs
Count
0.9
0.6
0.3
0.0
0.2
0.4
0.6
dnds
B
GC:0.310
sample size:543
neutral:0.289
asymptote:0.200
init slope:0.216
tau:0.477
r:0.455
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

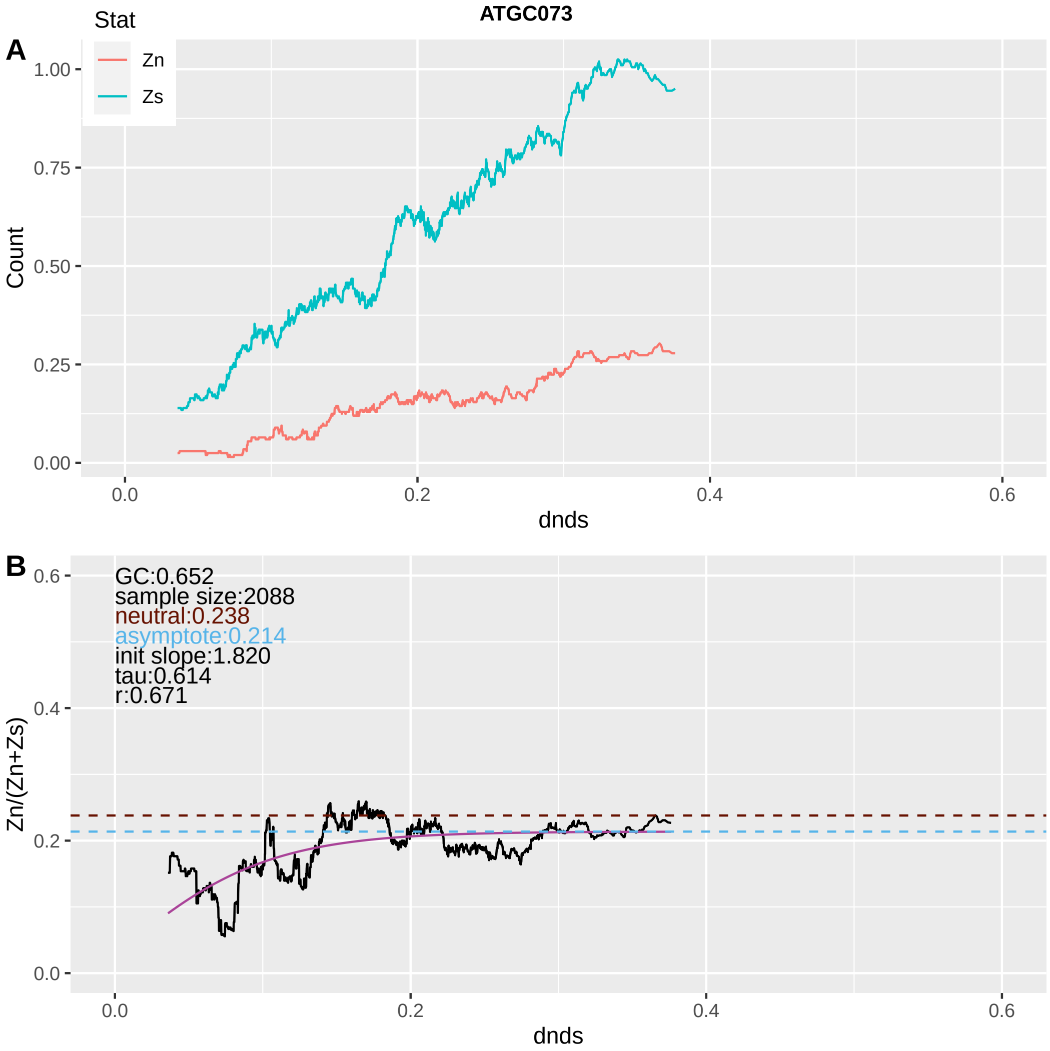

ATGC050
A
Stat
Zn
Zs
Count
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds
B
GC:0.386
sample size:567
neutral:0.291
asymptote:0.233
init slope:2.334
tau:0.444
r:0.812
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

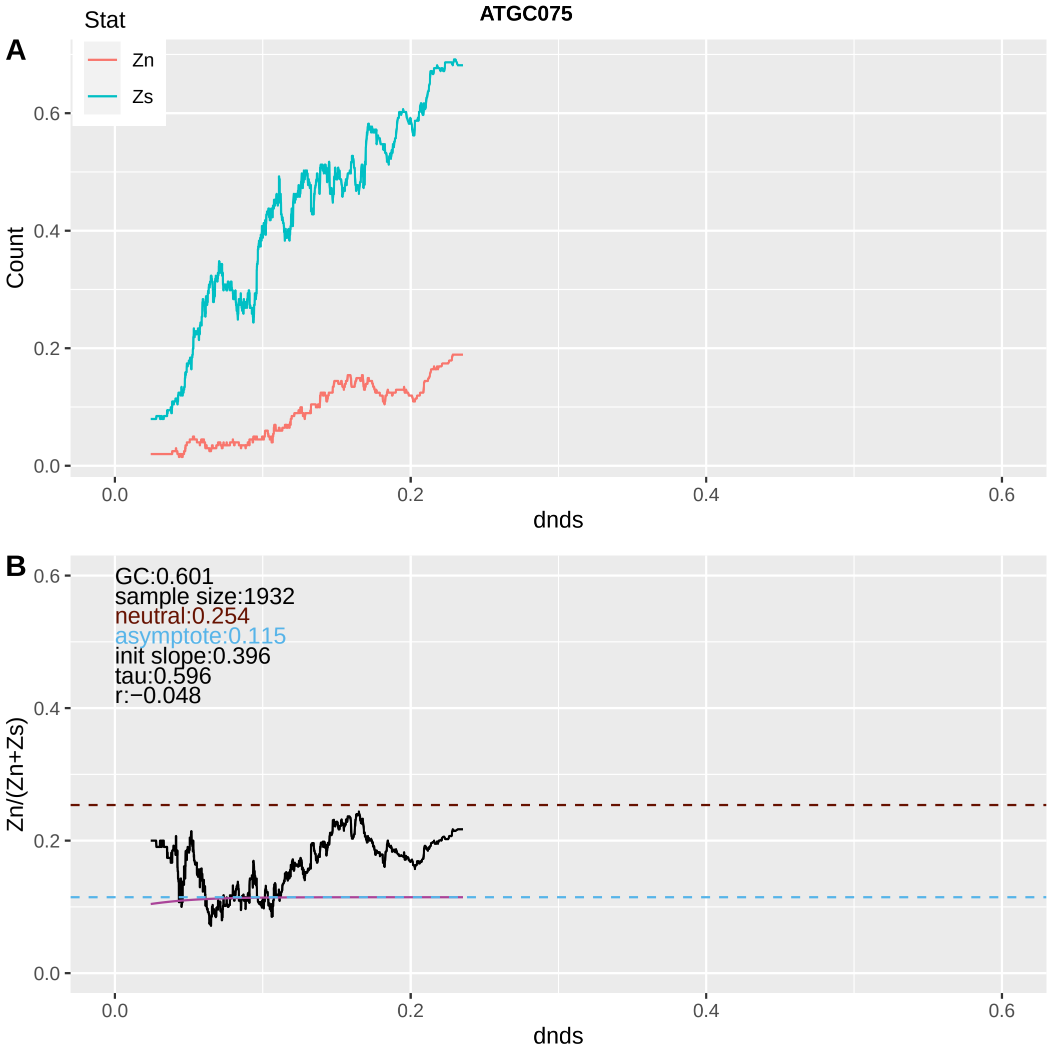

ATGC056
A
Stat
Zn
Zs
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.480
sample size:1300
neutral:0.281
asymptote:0.168
init slope:3.211
tau:0.726
r:0.937
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

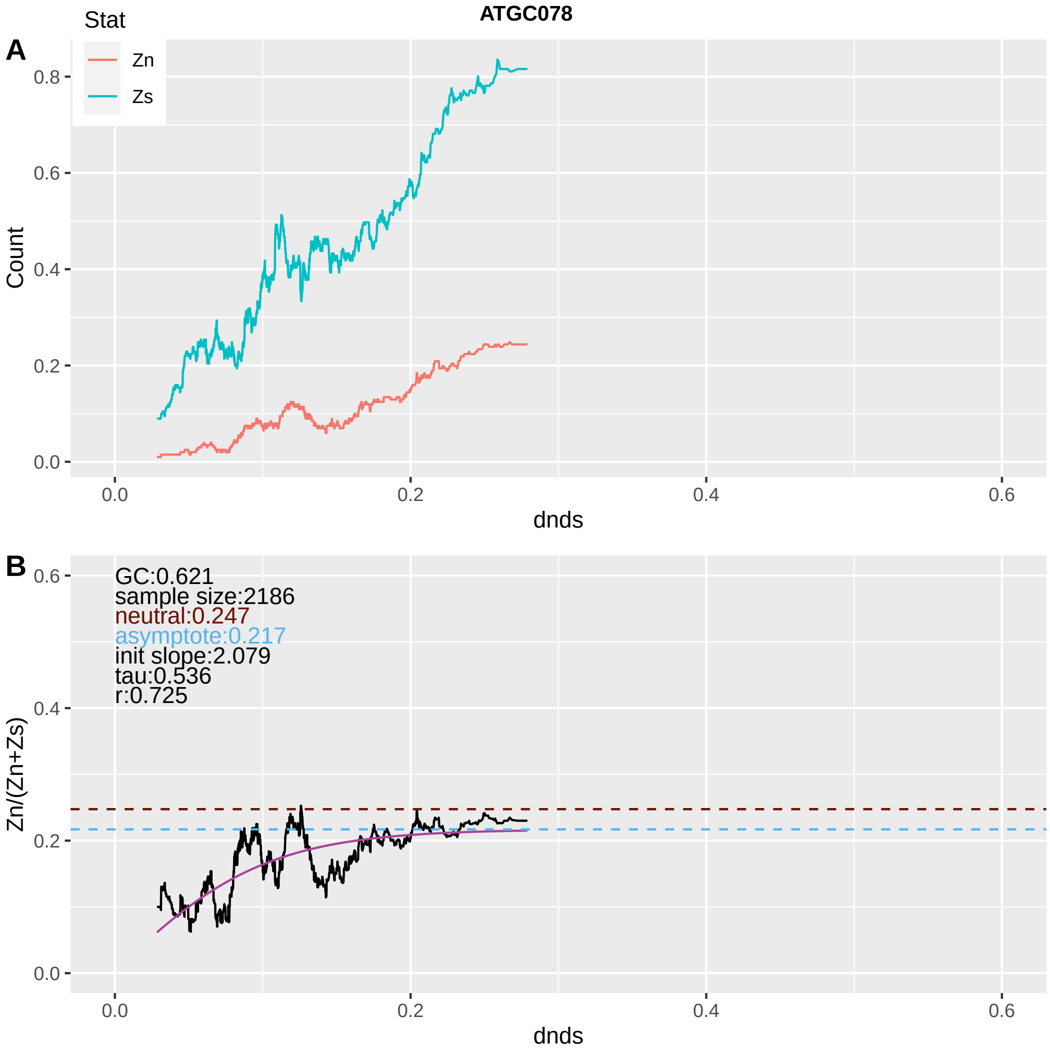

ATGC058
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.0
0.2
0.4
0.6
dnds
B
GC:0.374
sample size:894
neutral:0.290
asymptote:0.245
init slope:0.529
tau:0.602
r:−0.089
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

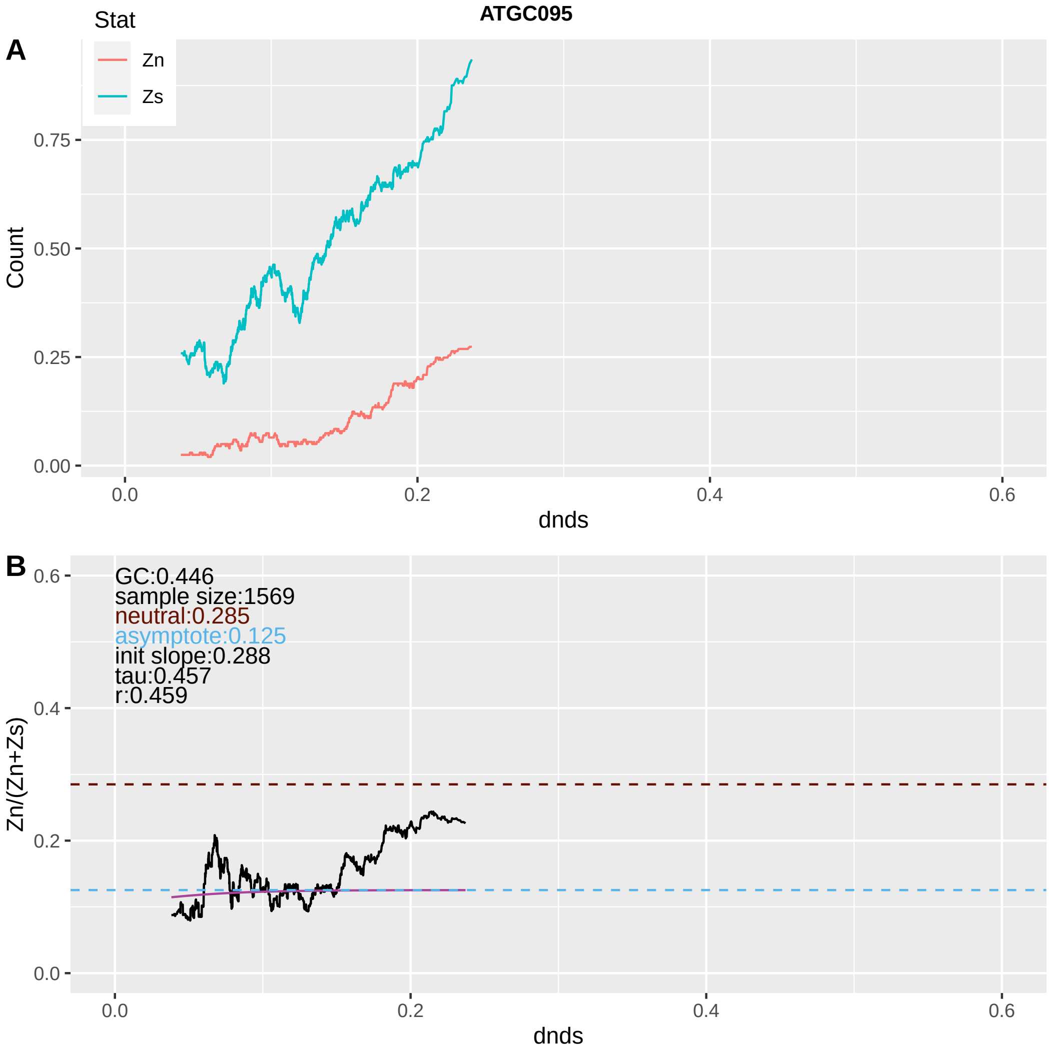

ATGC069
A
Stat
Zn
Zs
Count
0.00
0.25
0.50
0.75
0.0
0.2
0.4
0.6
dnds
B
GC:0.561
sample size:1372
neutral:0.263
asymptote:0.208
init slope:2.534
tau:0.720
r:0.773
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

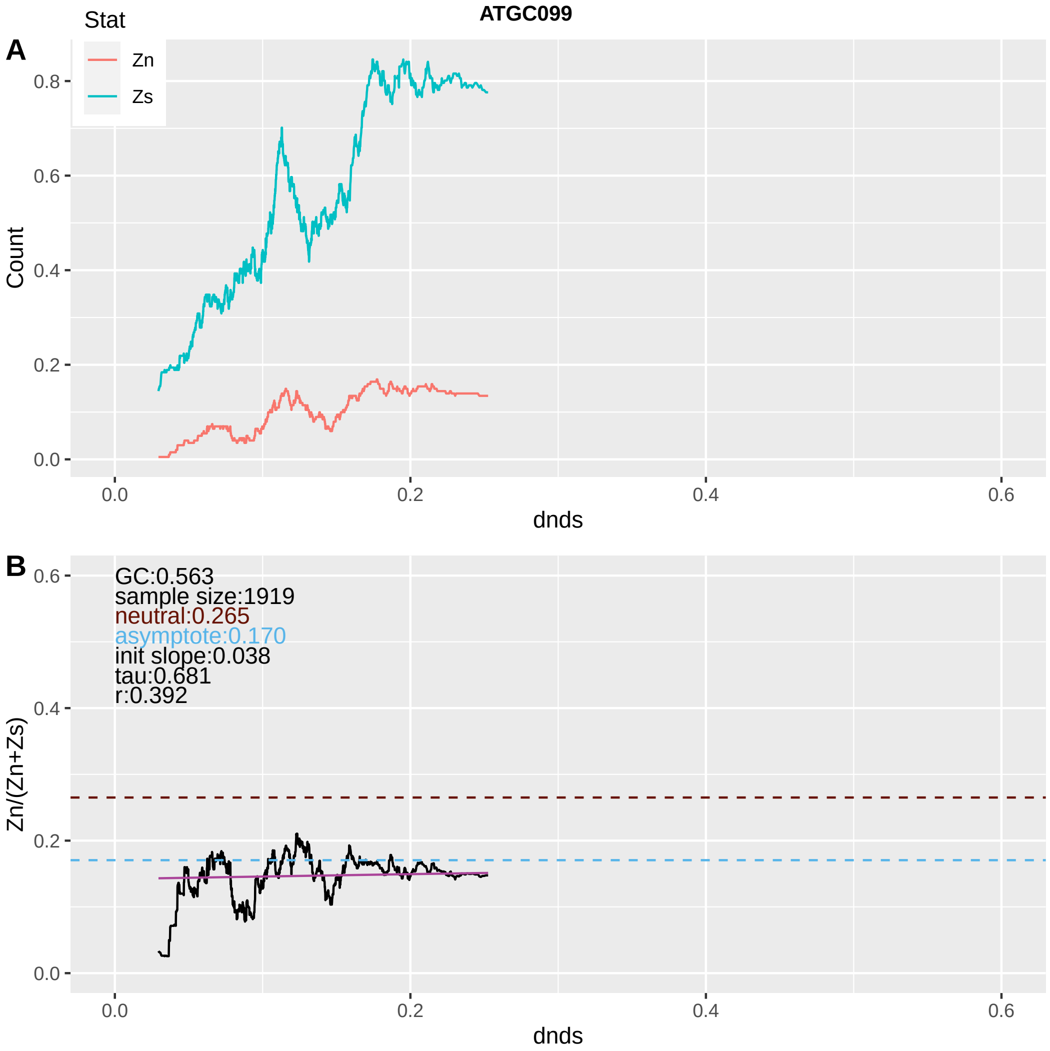

ATGC071
Stat
Zn
Zs
A
0.0
0.2
0.4
0.6
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.649
sample size:1725
neutral:0.240
asymptote:0.219
init slope:0.867
tau:0.515
r:0.611
0.0
0.2
0.4
0.6
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

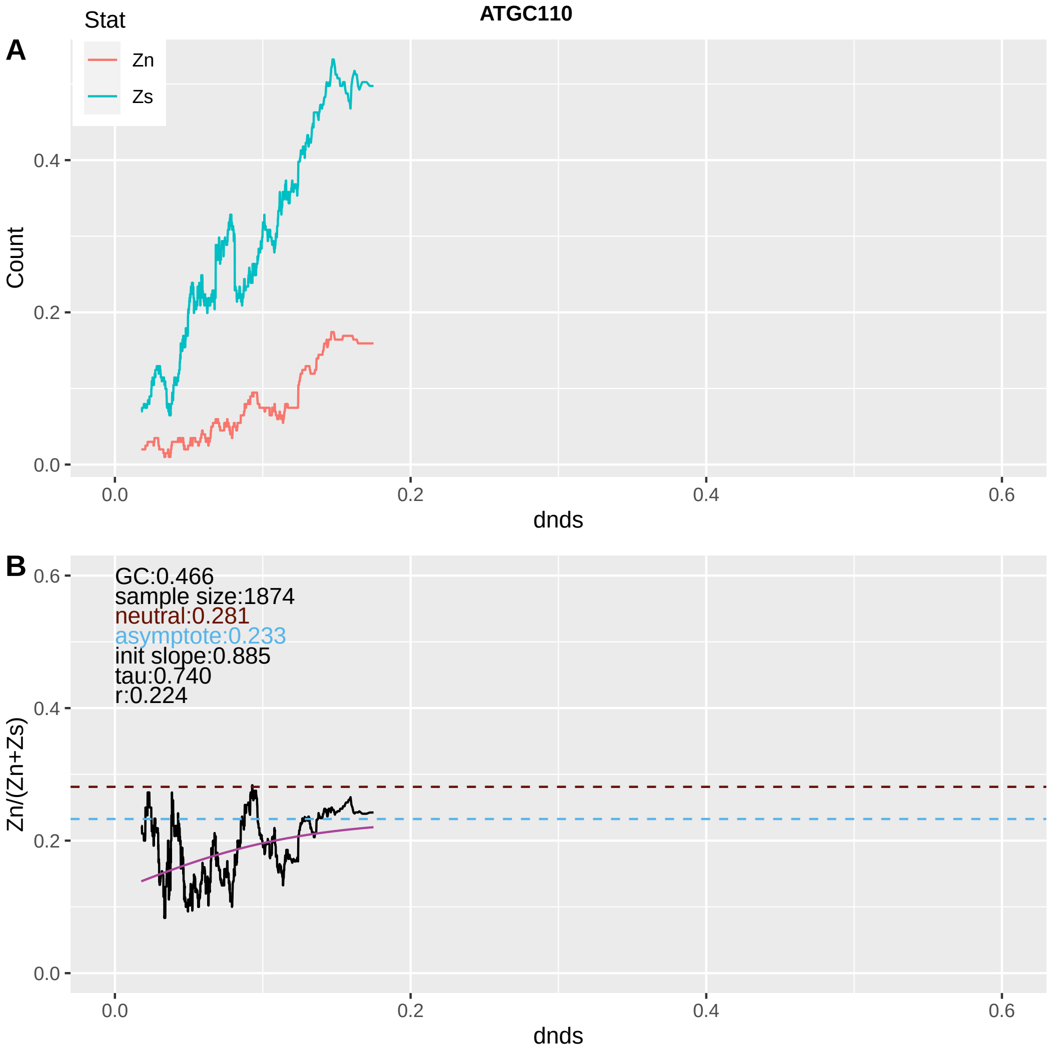

ATGC073
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.652
sample size:2088
neutral:0.238
asymptote:0.214
init slope:1.820
tau:0.614
r:0.671
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

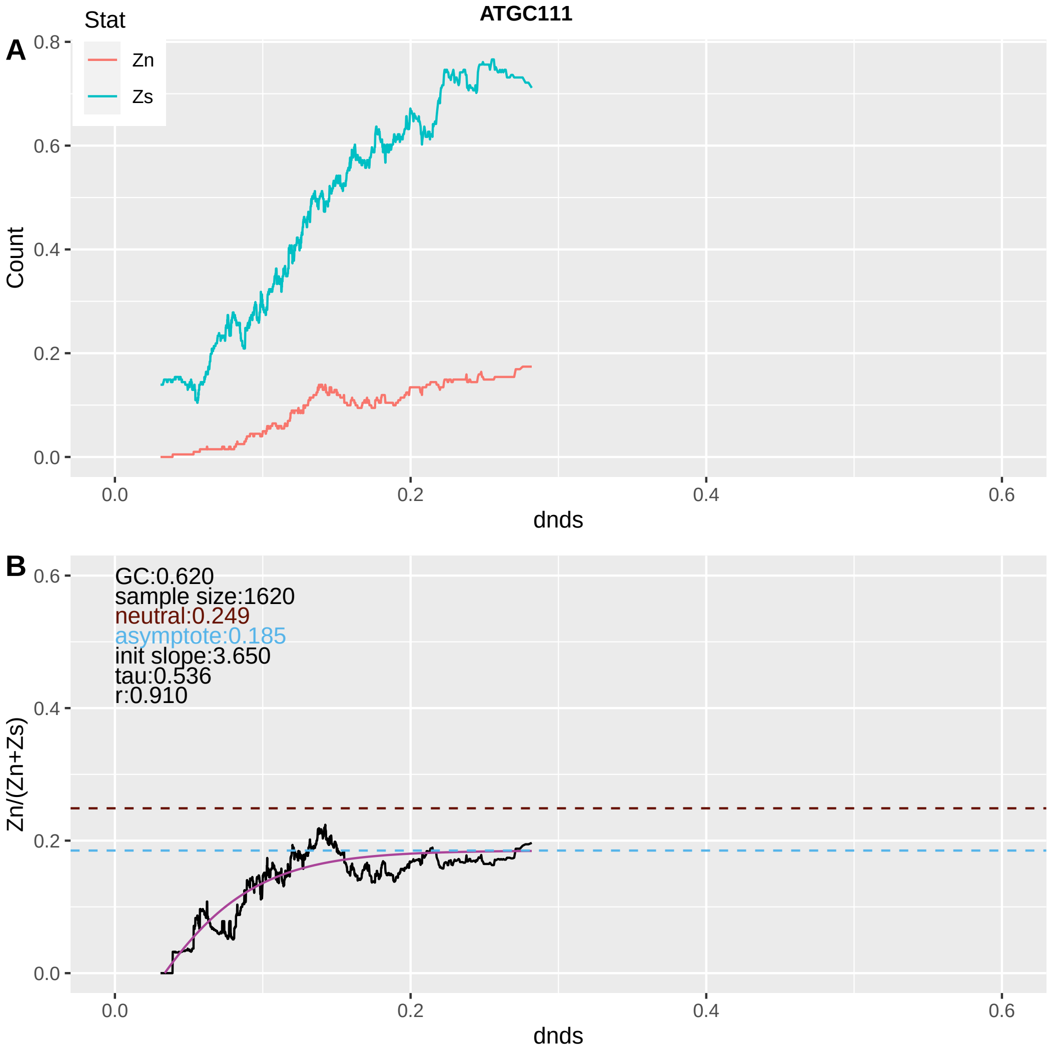

ATGC075
Stat
A
Zn
Zs
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.601
sample size:1932
neutral:0.254
asymptote:0.115
init slope:0.396
tau:0.596
r:−0.048
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

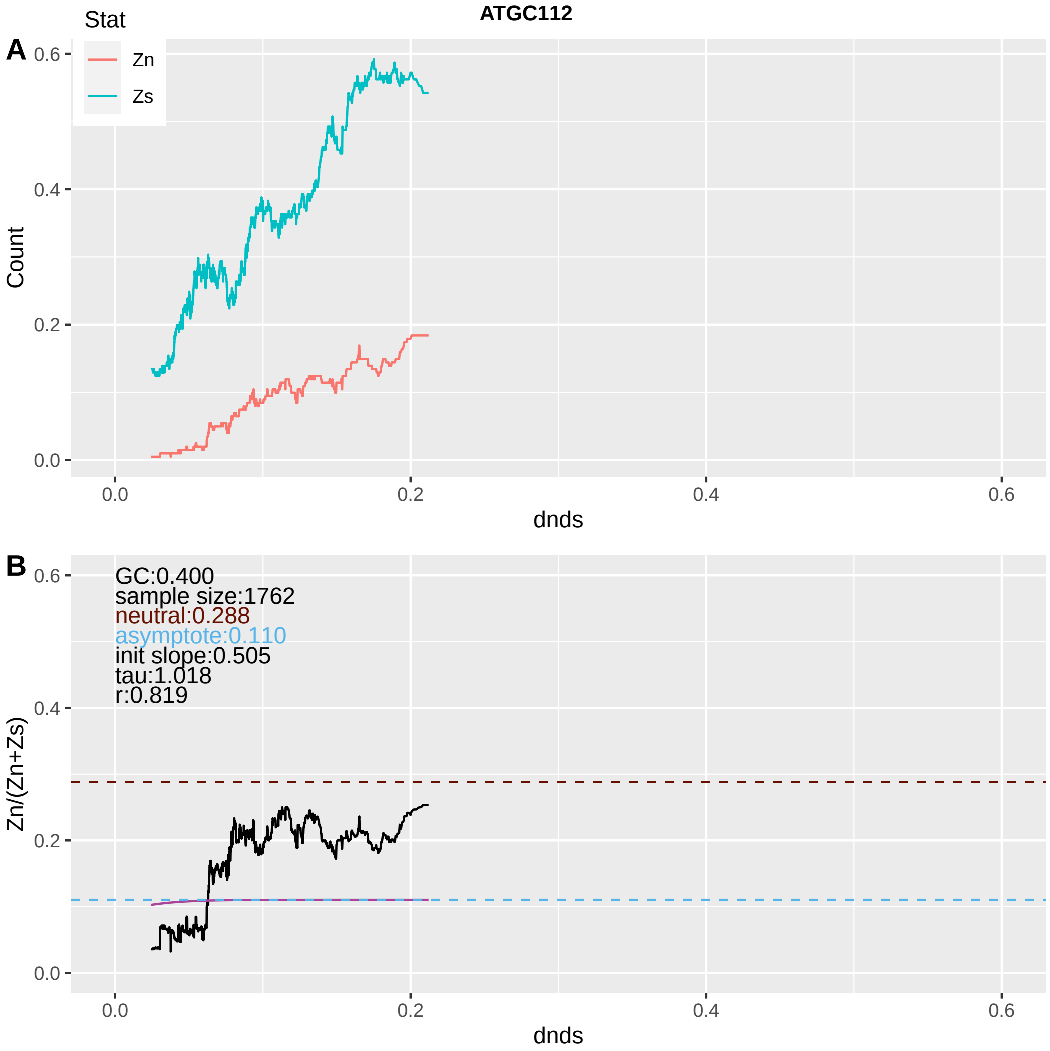

ATGC078
Stat
Zn
Zs
A
0.8
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.621
sample size:2186
neutral:0.247
asymptote:0.217
init slope:2.079
tau:0.536
r:0.725
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

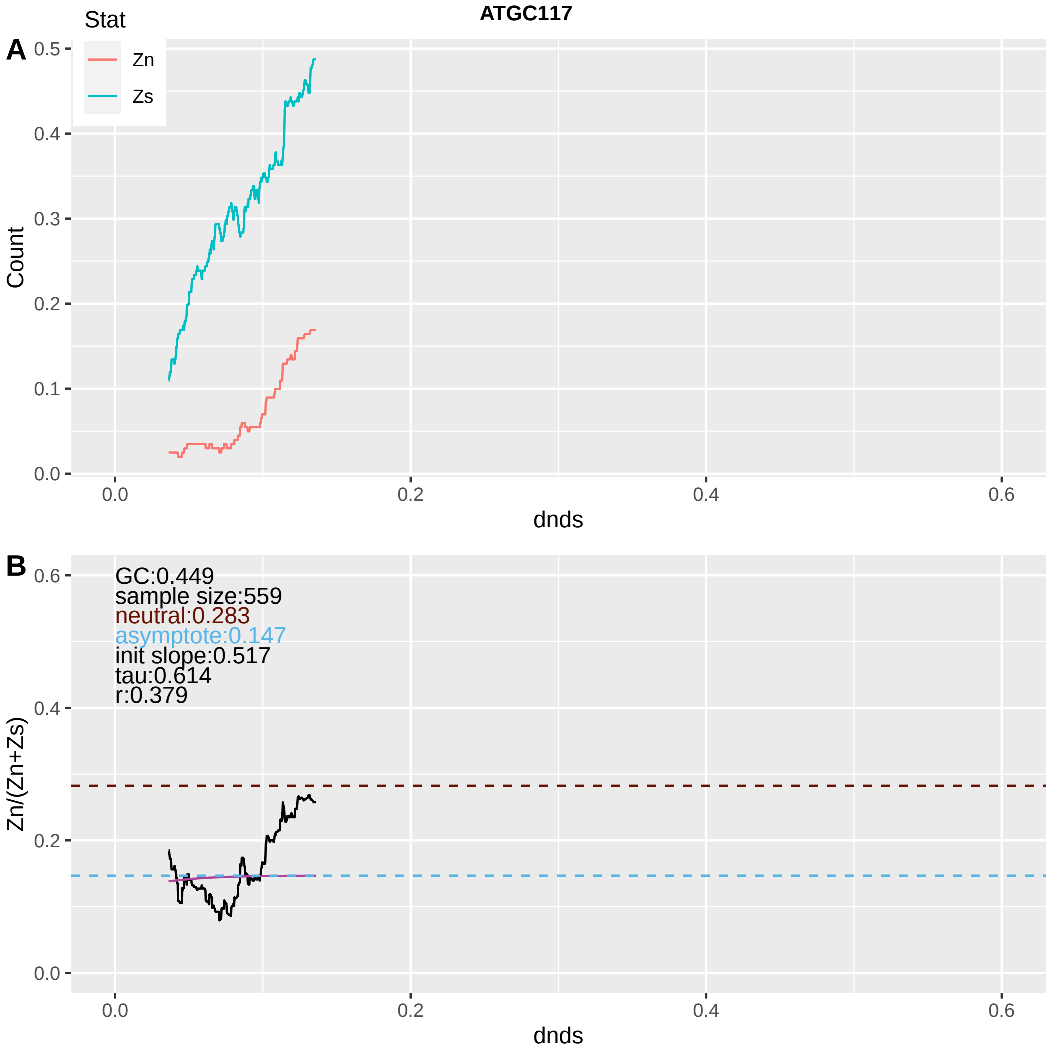

ATGC095
A
Stat
Zn
Zs
Count
0.00
0.25
0.50
0.75
0.0
0.2
0.4
0.6
dnds
B
GC:0.446
sample size:1569
neutral:0.285
asymptote:0.125
init slope:0.288
tau:0.457
r:0.459
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

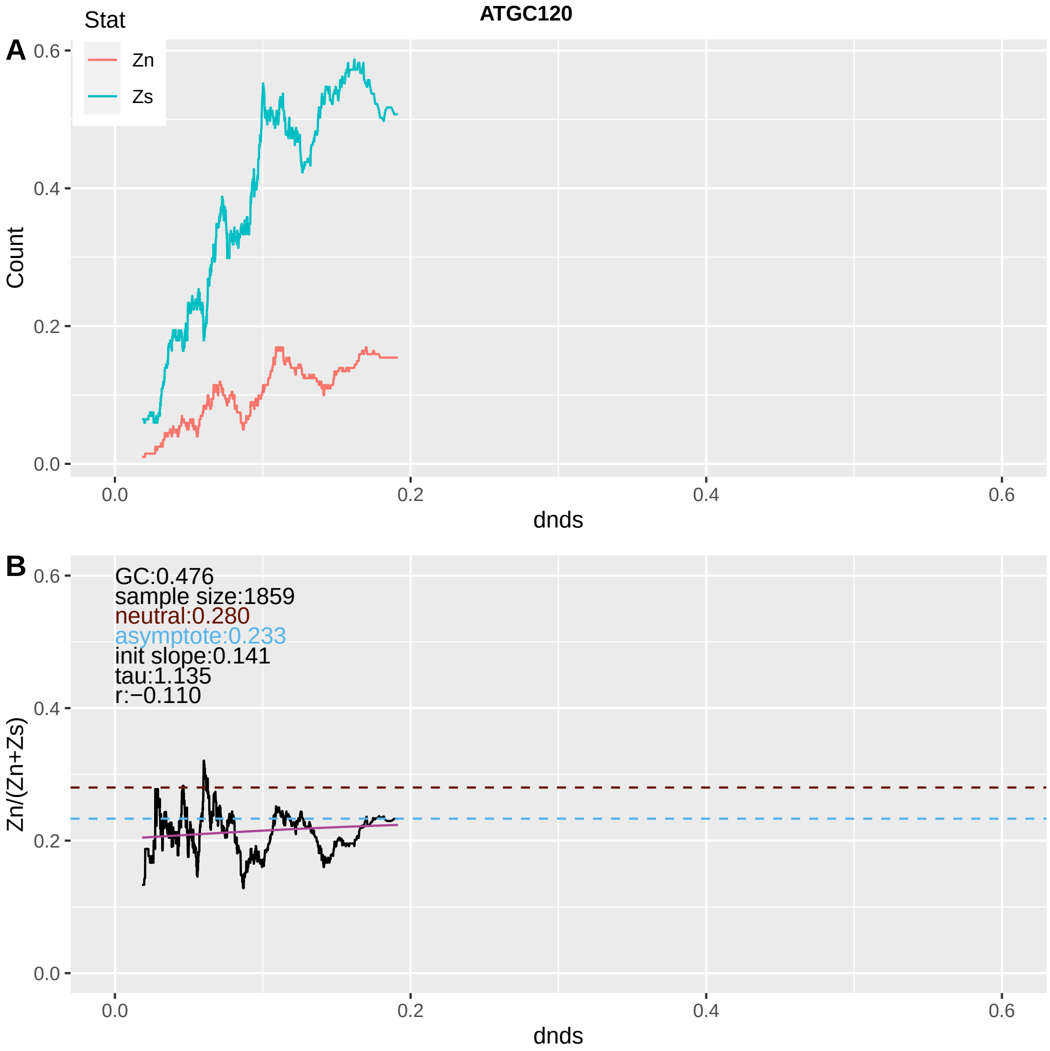

ATGC099
Stat
A
Zn
Zs
0.8
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
GC:0.563
sample size:1919
neutral:0.265
asymptote:0.170
init slope:0.038
tau:0.681
r:0.392
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

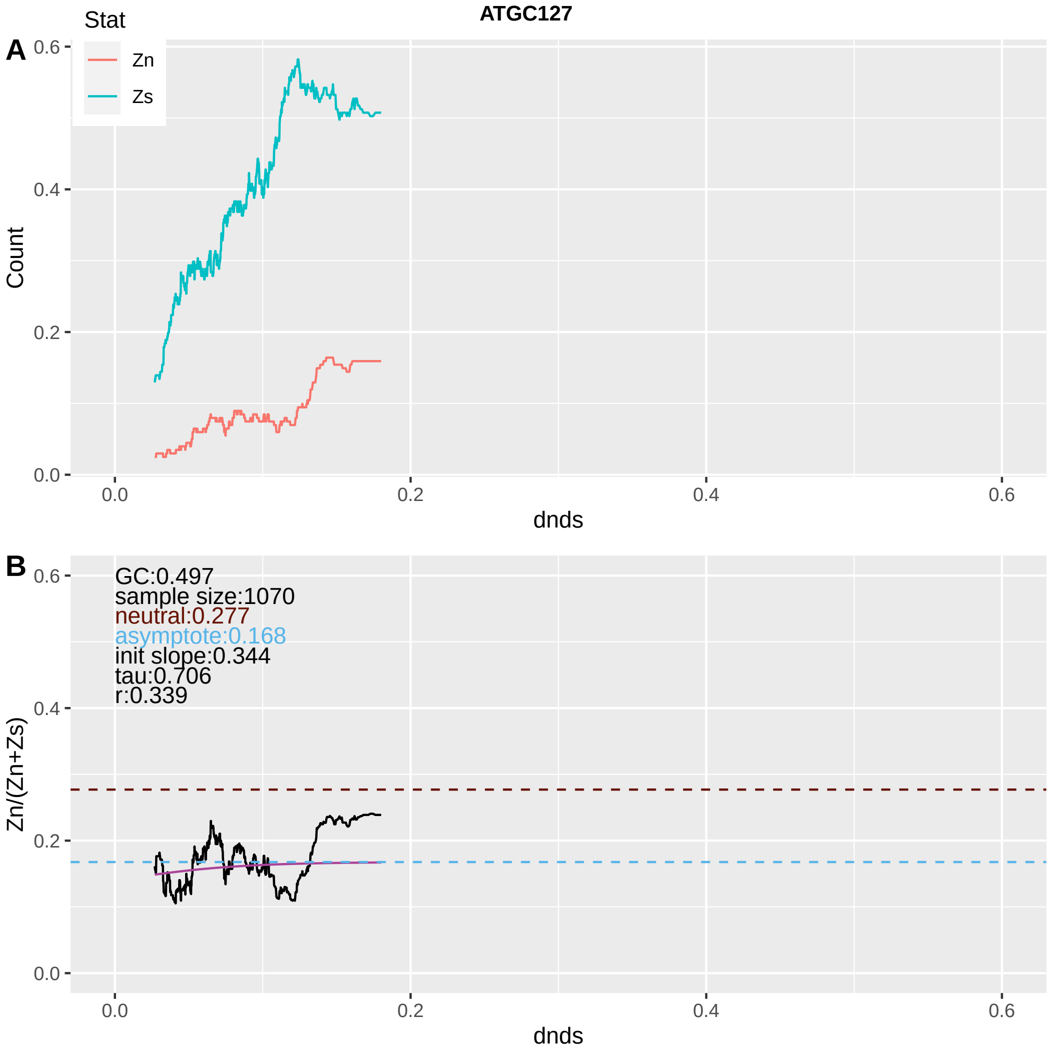

ATGC110
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.0
0.2
0.4
0.6
dnds
B
GC:0.466
sample size:1874
neutral:0.281
asymptote:0.233
init slope:0.885
tau:0.740
r:0.224
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

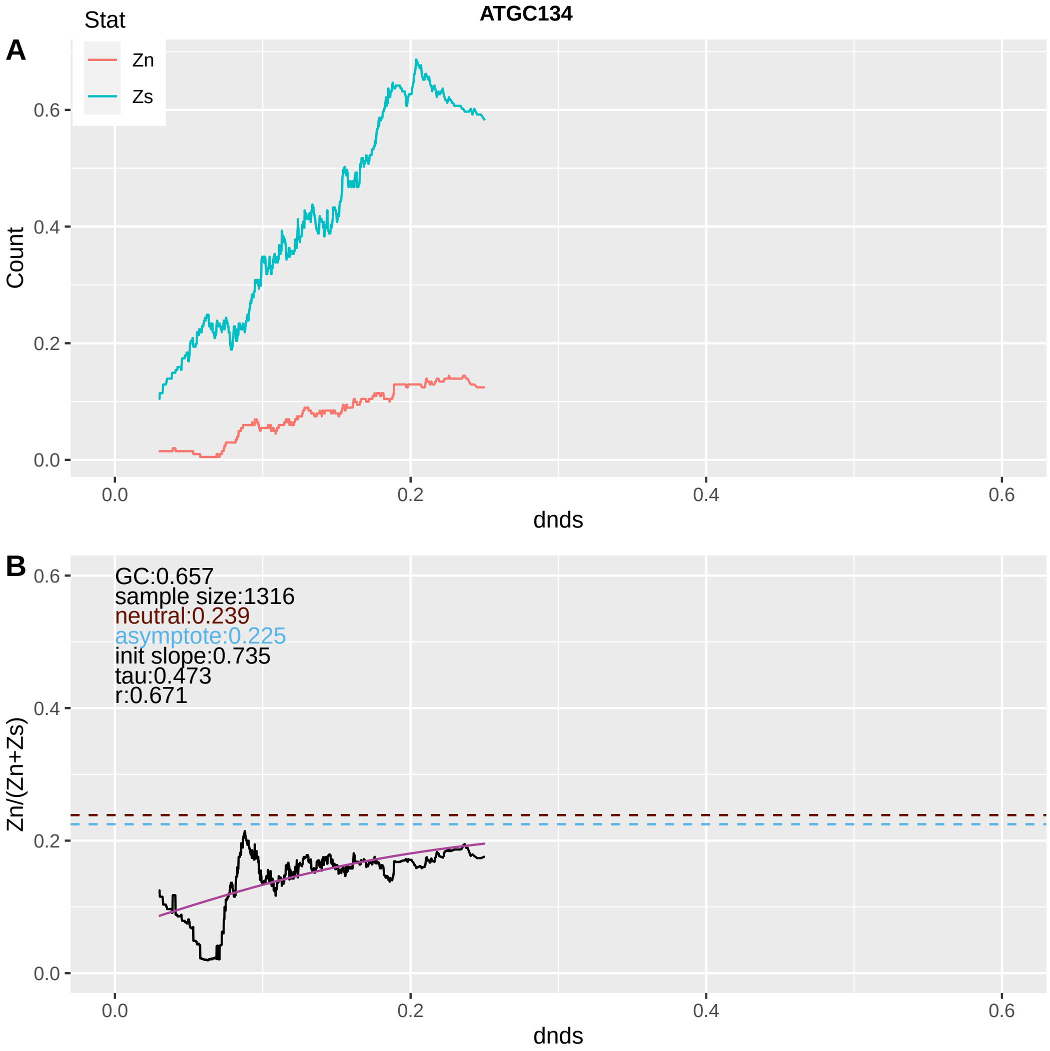

ATGC111
A
Stat
Zn
Zs
Count
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
GC:0.620
sample size:1620
neutral:0.249
asymptote:0.185
init slope:3.650
tau:0.536
r:0.910
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

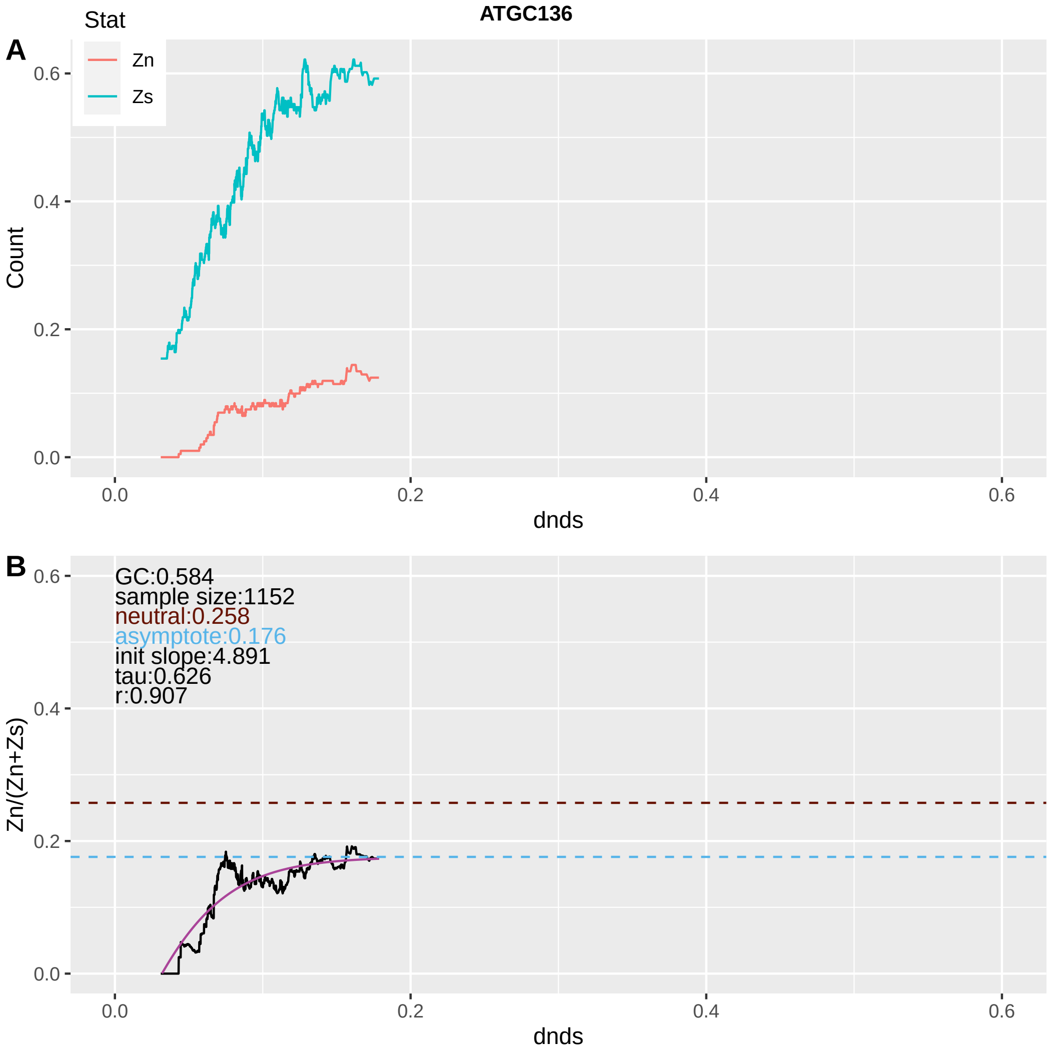

ATGC112
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6
B
GC:0.400
sample size:1762
neutral:0.288
asymptote:0.110
init slope:0.505
tau:1.018
r:0.819
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6

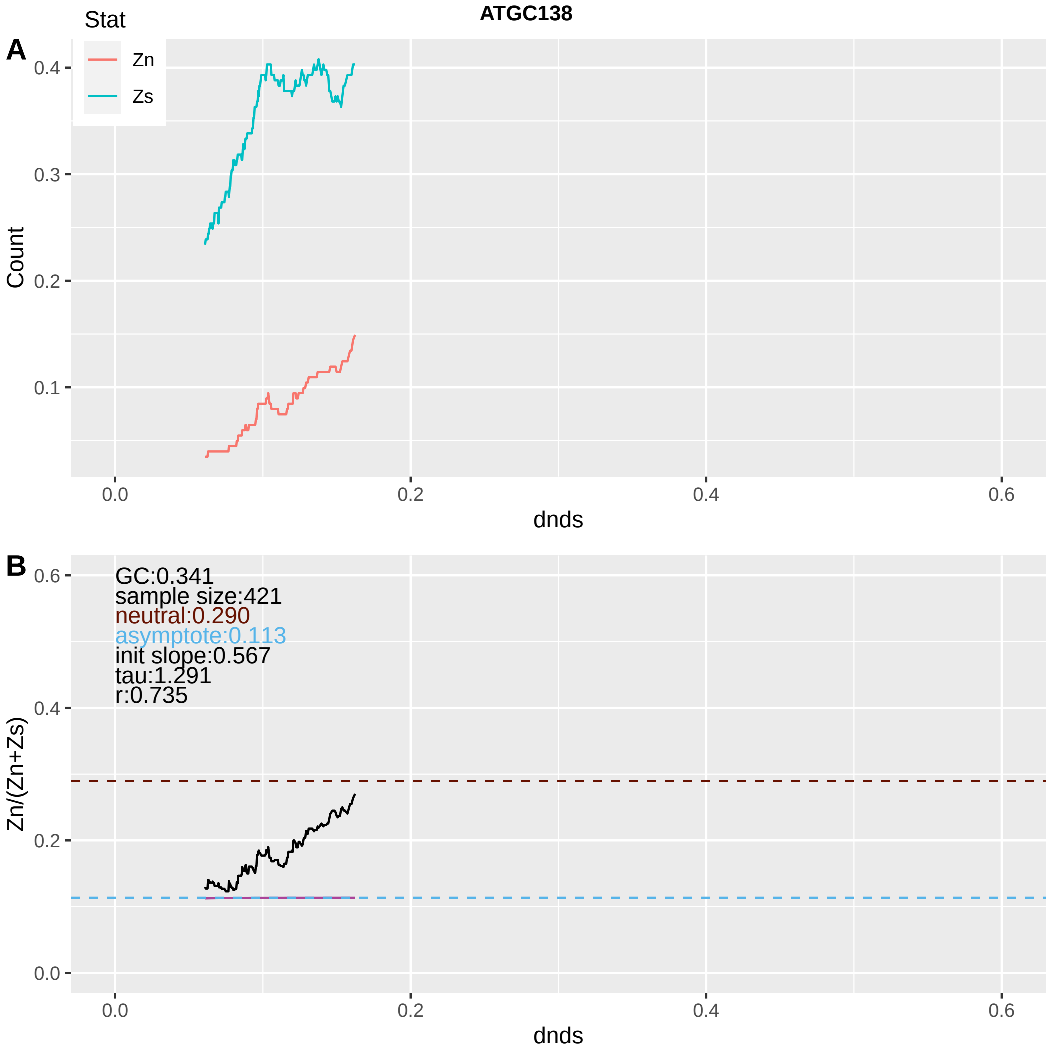

ATGC117
Stat
A
Zn
Zs
0.5
0.4
0.3
0.2
0.1
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.449
sample size:559
neutral:0.283
asymptote:0.147
init slope:0.517
tau:0.614
r:0.379
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

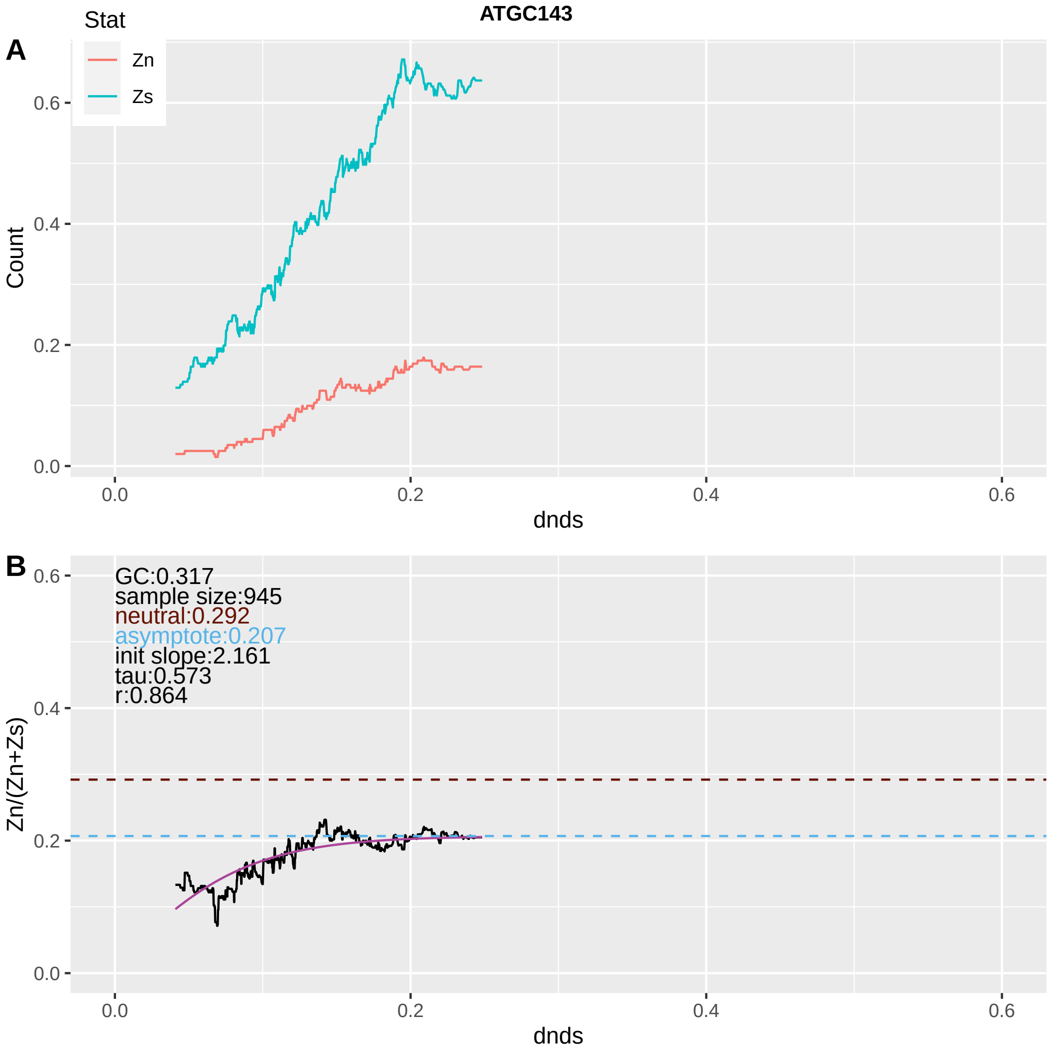

ATGC120
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.476
sample size:1859
neutral:0.280
asymptote:0.233
init slope:0.141
tau:1.135
r:−0.110
Zn/(Zn+Zs)
dnds

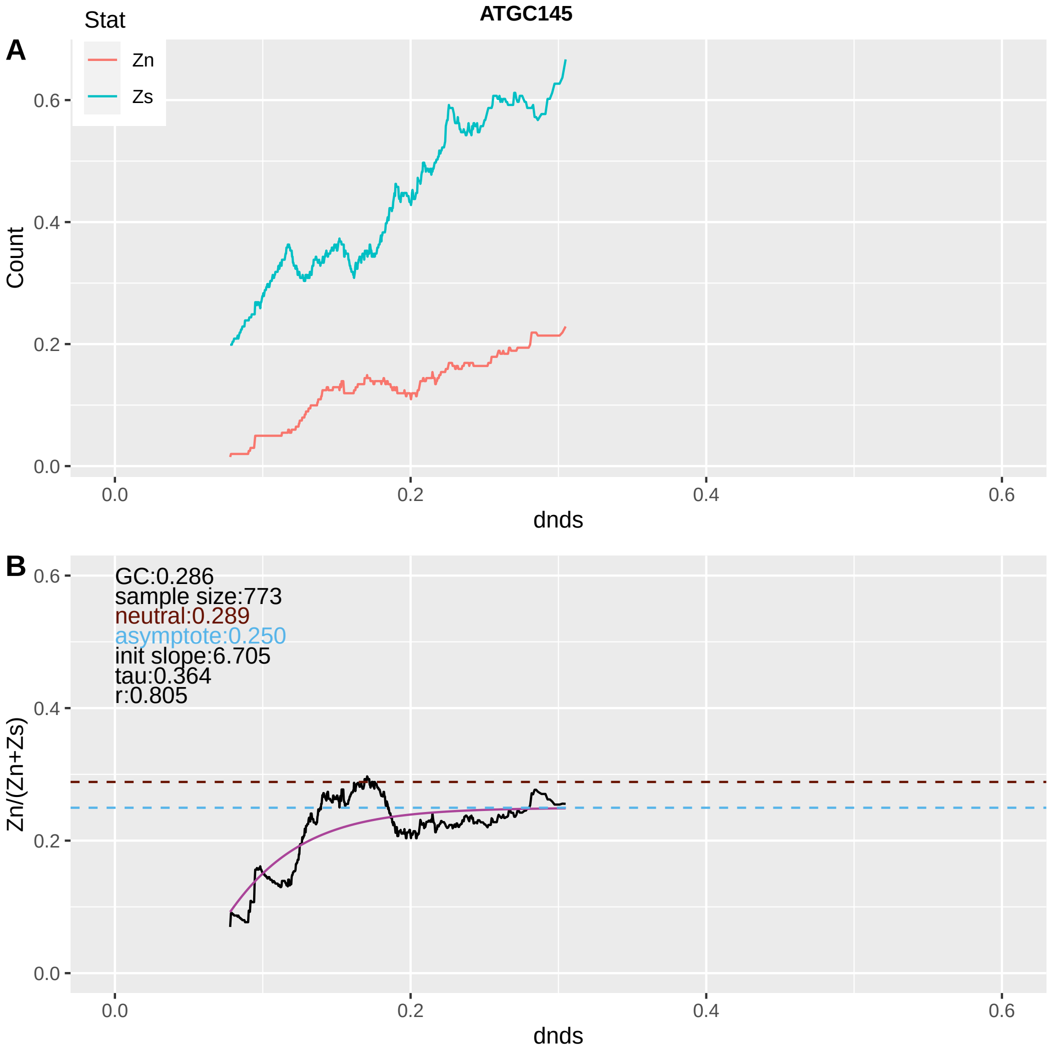

ATGC127
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.497
sample size:1070
neutral:0.277
asymptote:0.168
init slope:0.344
tau:0.706
r:0.339
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

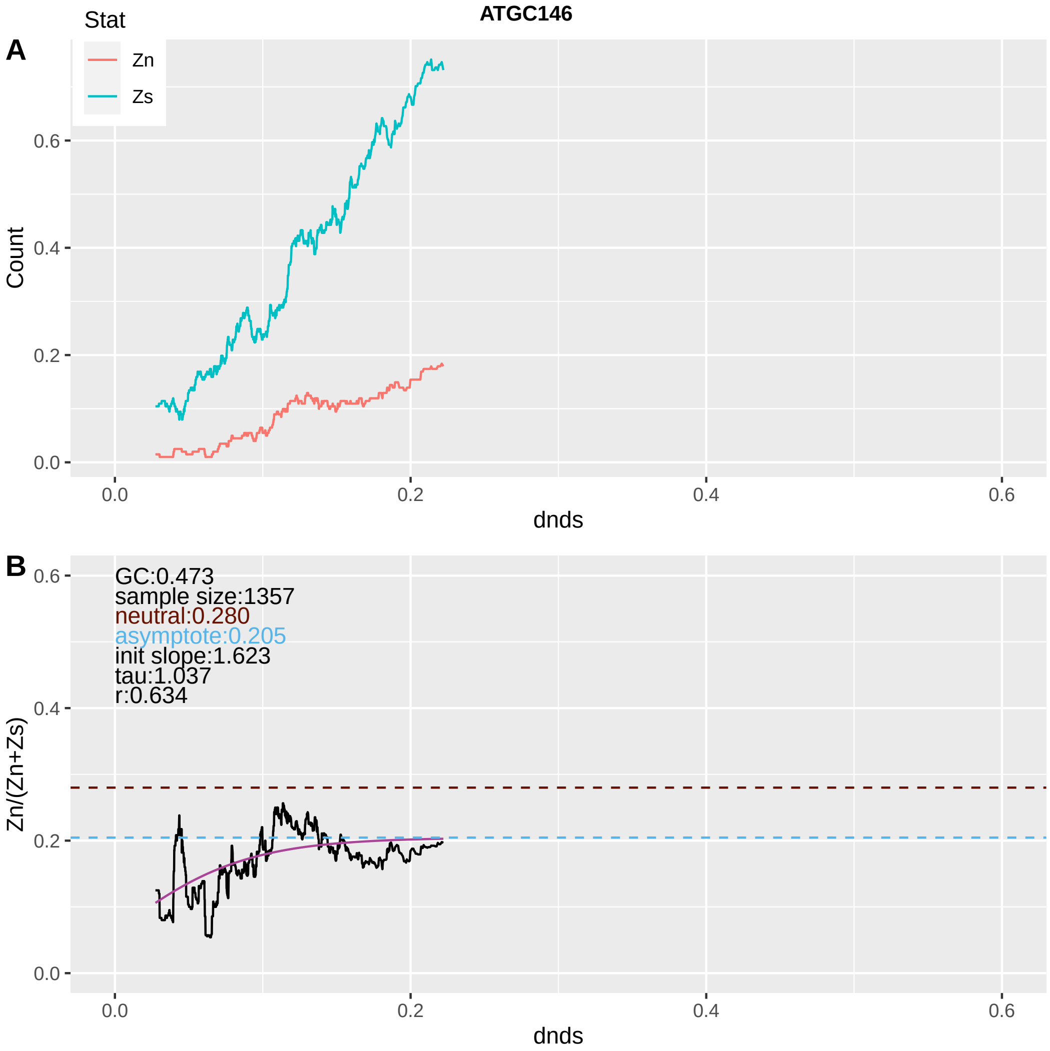

ATGC134
Stat
A
Zn
Zs
Count
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds
B
GC:0.657
sample size:1316
neutral:0.239
asymptote:0.225
init slope:0.735
tau:0.473
r:0.671
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

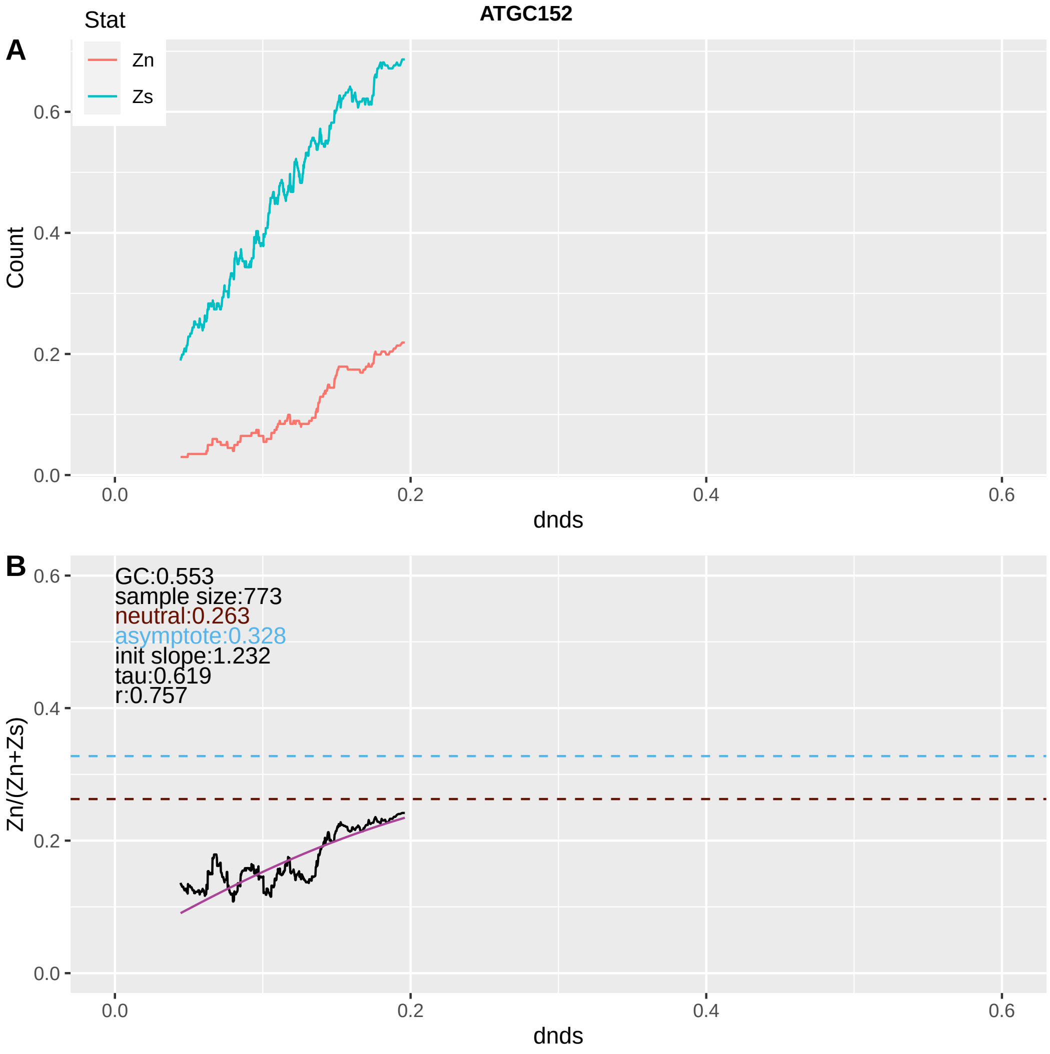

ATGC136
A
Stat
Zn
Zs
0.0
0.2
0.4
0.6
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.584
sample size:1152
neutral:0.258
asymptote:0.176
init slope:4.891
tau:0.626
r:0.907
0.0
0.2
0.4
0.6
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

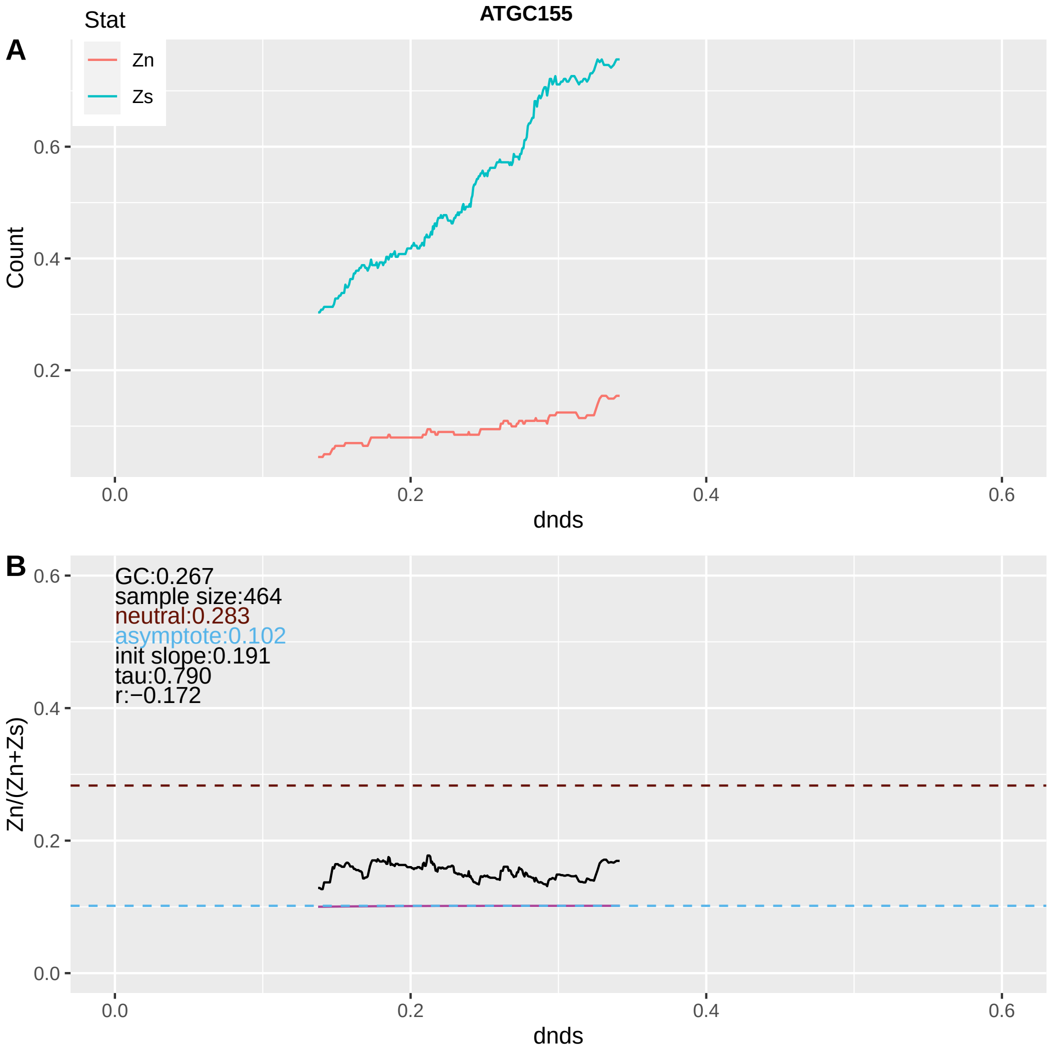

ATGC138
A
Stat
Zn
Zs
Count
0.4
0.3
0.2
0.1
0.0
0.2
0.4
0.6
dnds
B
GC:0.341
sample size:421
neutral:0.290
asymptote:0.113
init slope:0.567
tau:1.291
r:0.735
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

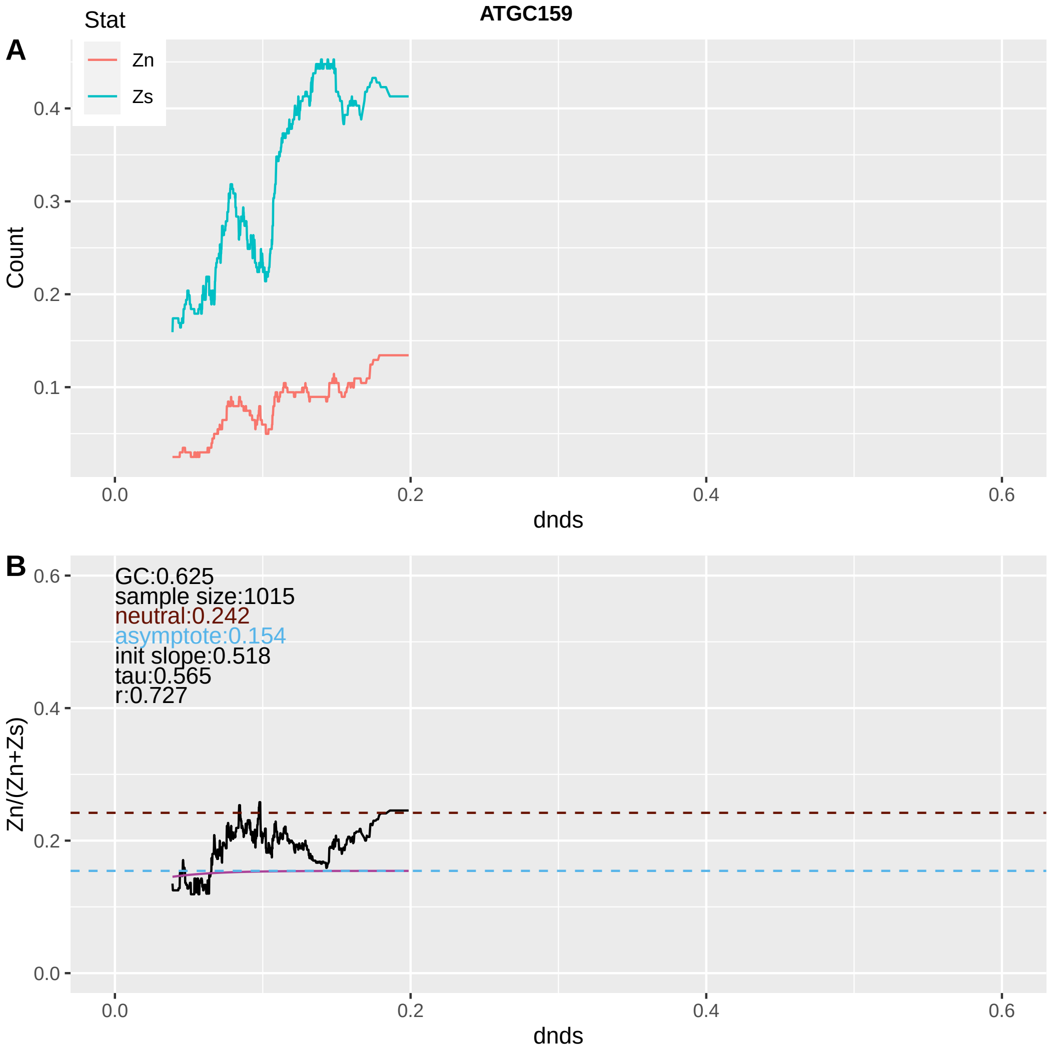

ATGC143
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6
B
GC:0.317
sample size:945
neutral:0.292
asymptote:0.207
init slope:2.161
tau:0.573
r:0.864
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6

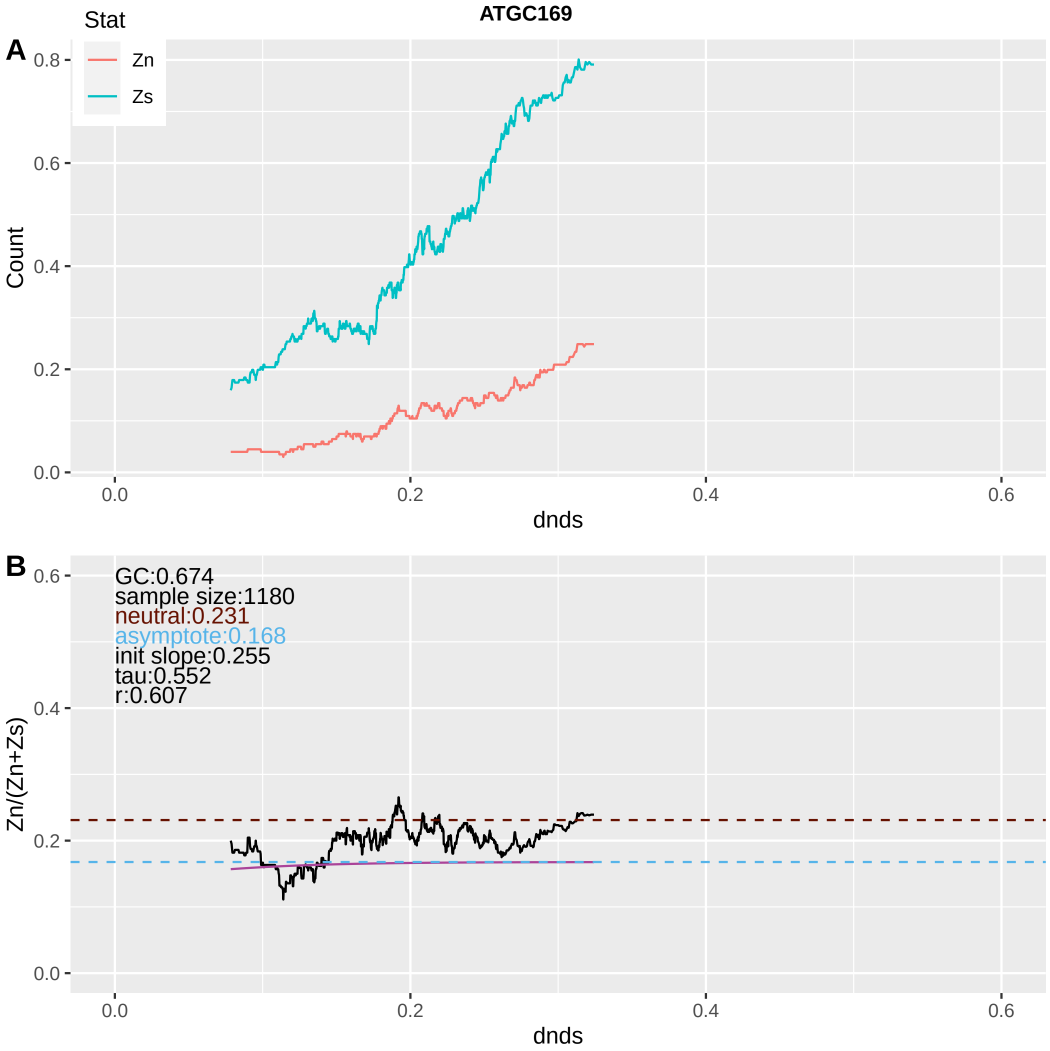

ATGC145
Stat
Zn
Zs
A
Count
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6
B
GC:0.286
sample size:773
neutral:0.289
asymptote:0.250
init slope:6.705
tau:0.364
r:0.805
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6


ATGC146
Stat
Zn
Zs
A
Count
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6
B
GC:0.473
sample size:1357
neutral:0.280
asymptote:0.205
init slope:1.623
tau:1.037
r:0.634
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6

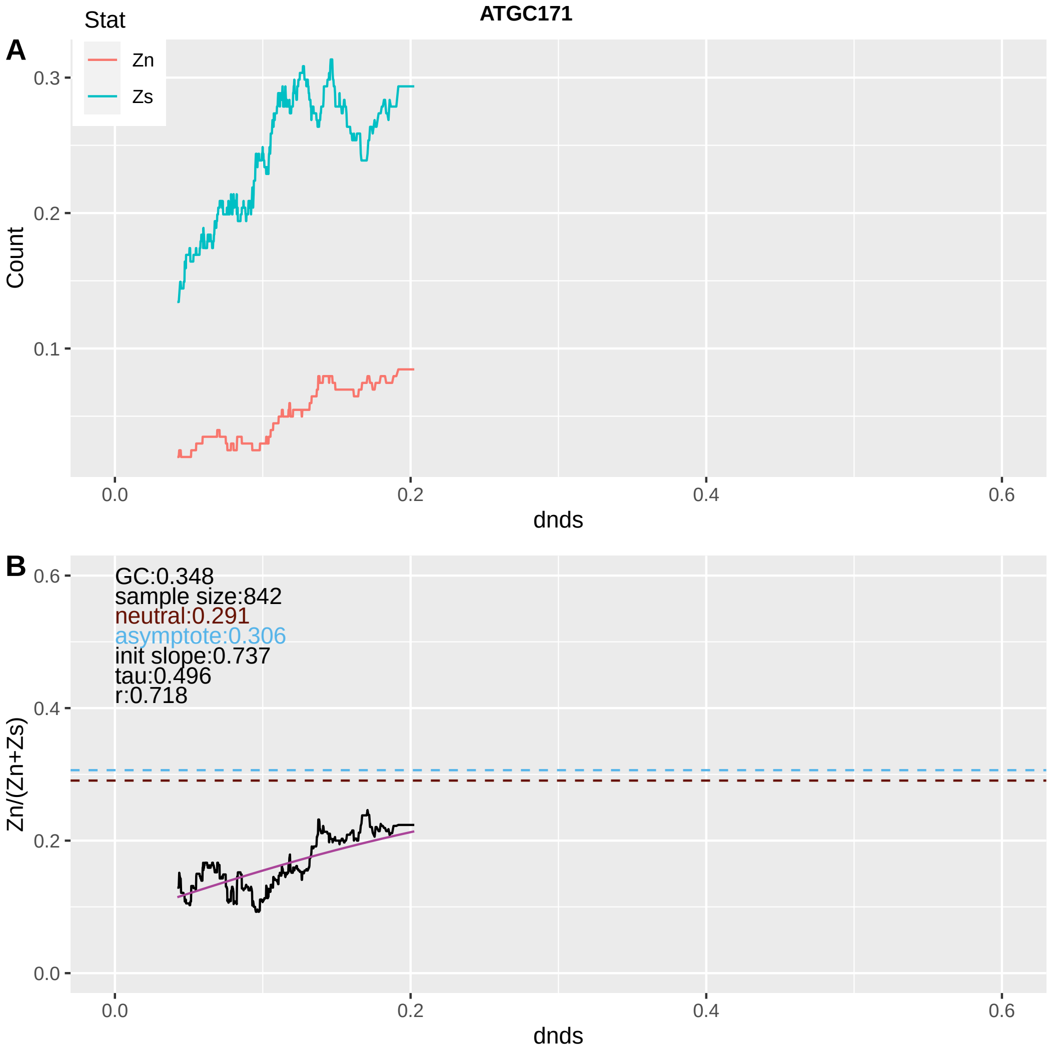

ATGC152
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6
B
GC:0.553
sample size:773
neutral:0.263
asymptote:0.328
init slope:1.232
tau:0.619
r:0.757
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6

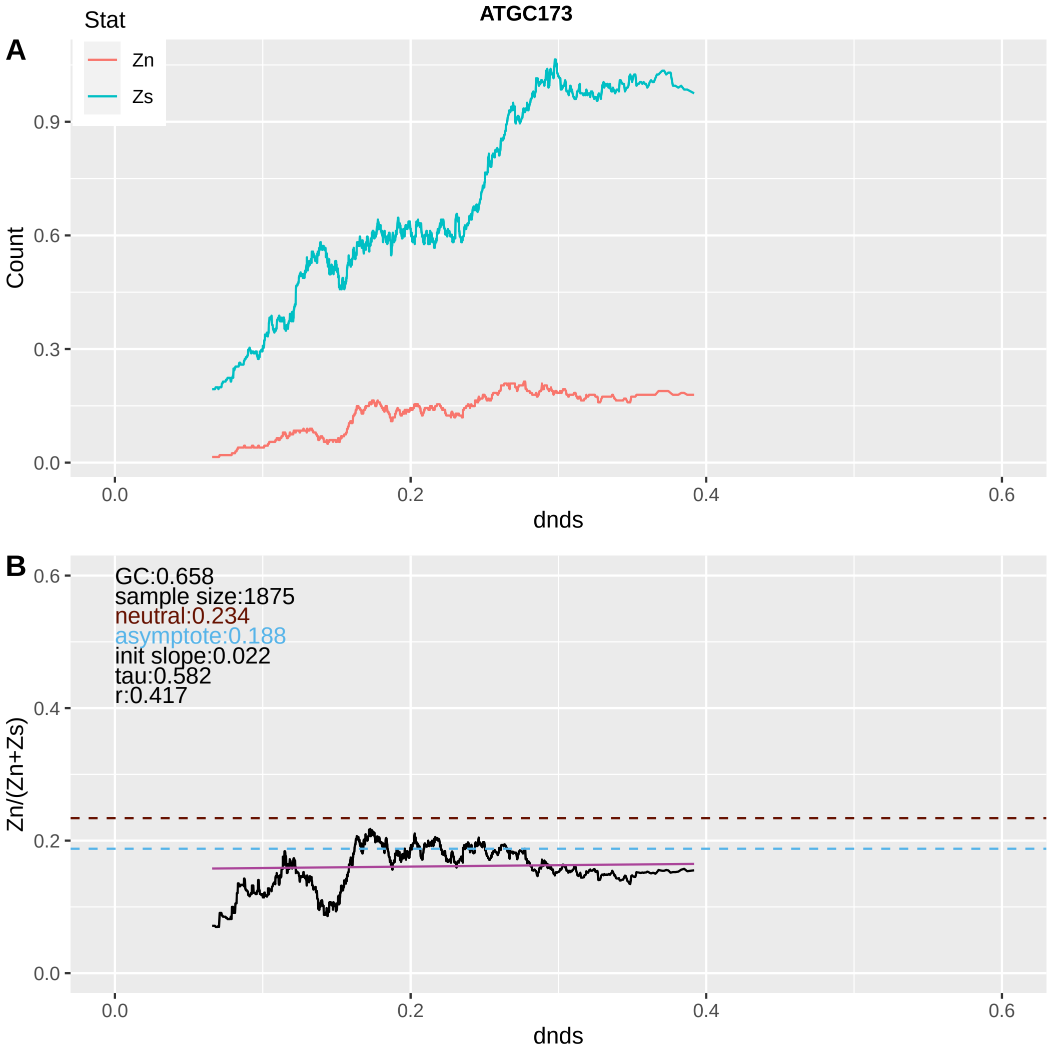

ATGC155
Stat
A
Zn
Zs
Count
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds
B
GC:0.267
sample size:464
neutral:0.283
asymptote:0.102
init slope:0.191
tau:0.790
r:−0.172
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

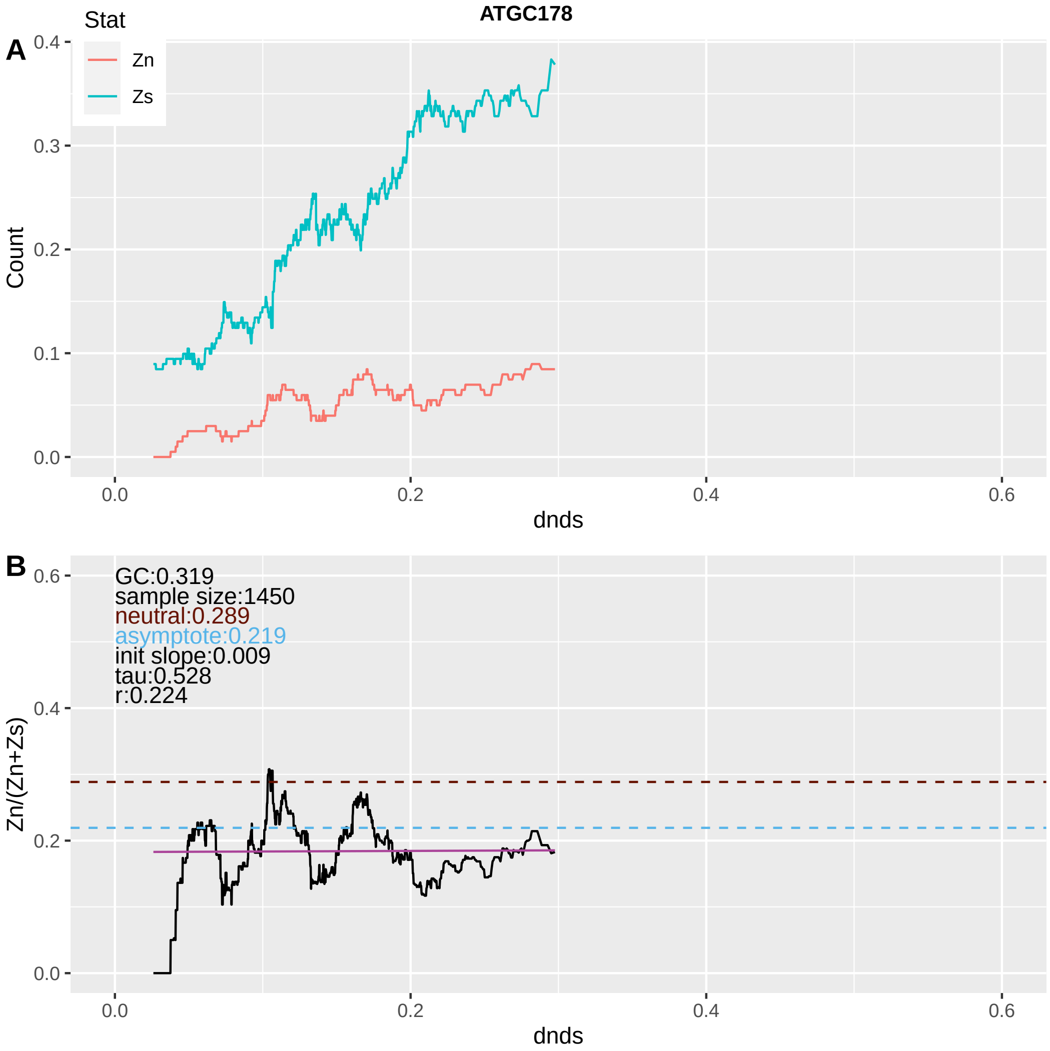

ATGC159
A
Stat
Zn
Zs
0.4
0.3
0.2
0.1
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
GC:0.625
sample size:1015
neutral:0.242
asymptote:0.154
init slope:0.518
tau:0.565
r:0.727
0.0
0.2
0.4
0.6
dnds

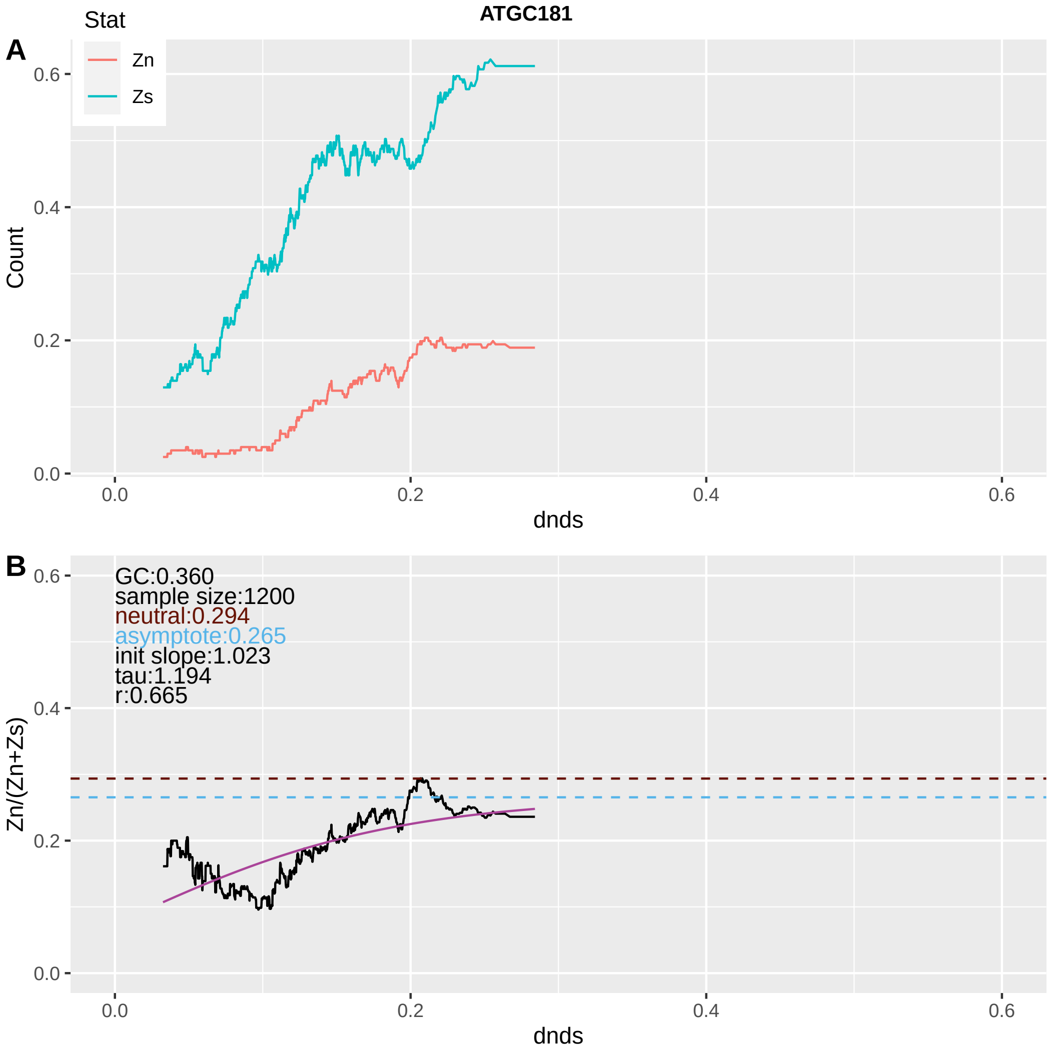


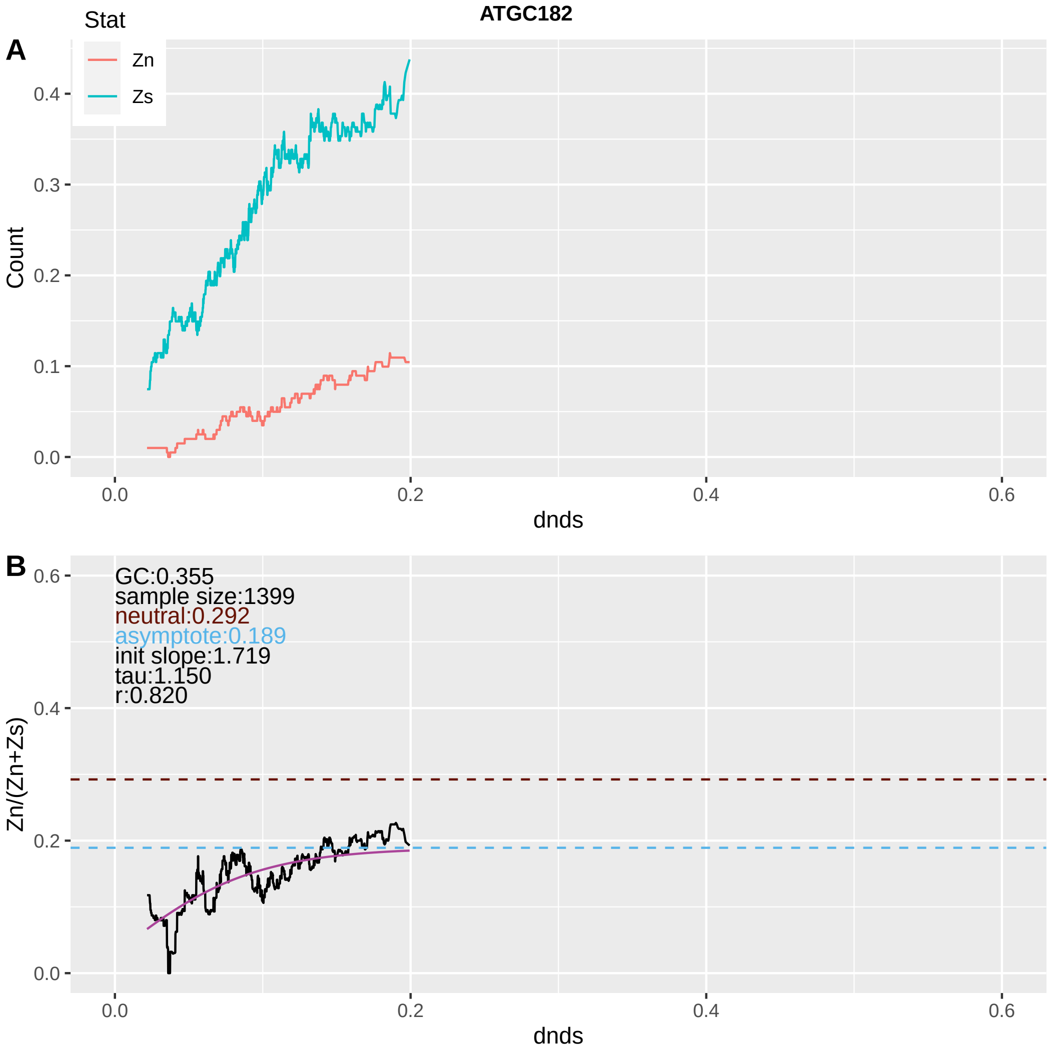
ATGC169
A
Stat
Zn
Zs
0.8
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
GC:0.674
sample size:1180
neutral:0.231
asymptote:0.168
init slope:0.255
tau:0.552
r:0.607
0.0
0.2
0.4
0.6
dnds

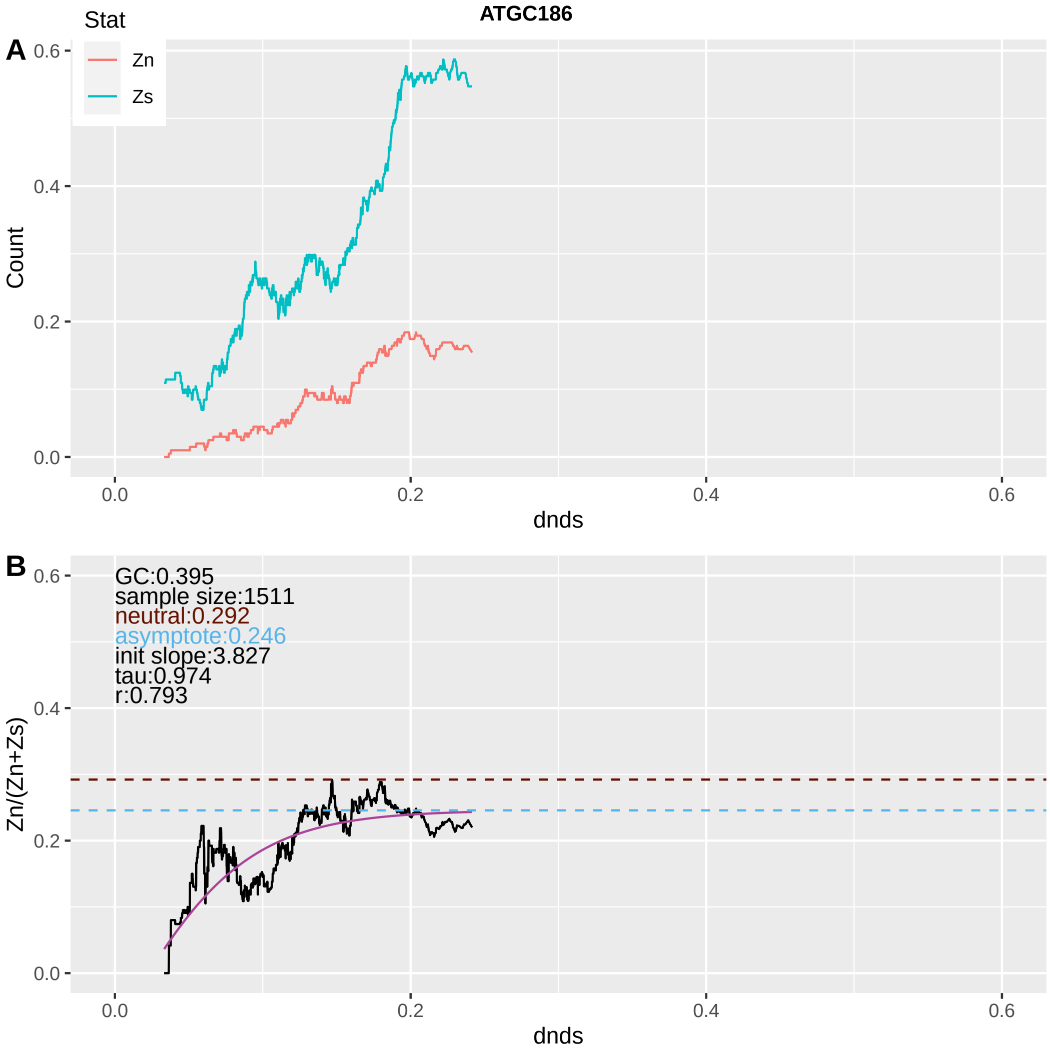

ATGC171
Stat
A
Zn
Zs
Count
0.3
0.2
0.1
0.0
0.2
0.4
0.6
dnds
B
GC:0.348
sample size:842
neutral:0.291
asymptote:0.306
init slope:0.737
tau:0.496
r:0.718
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

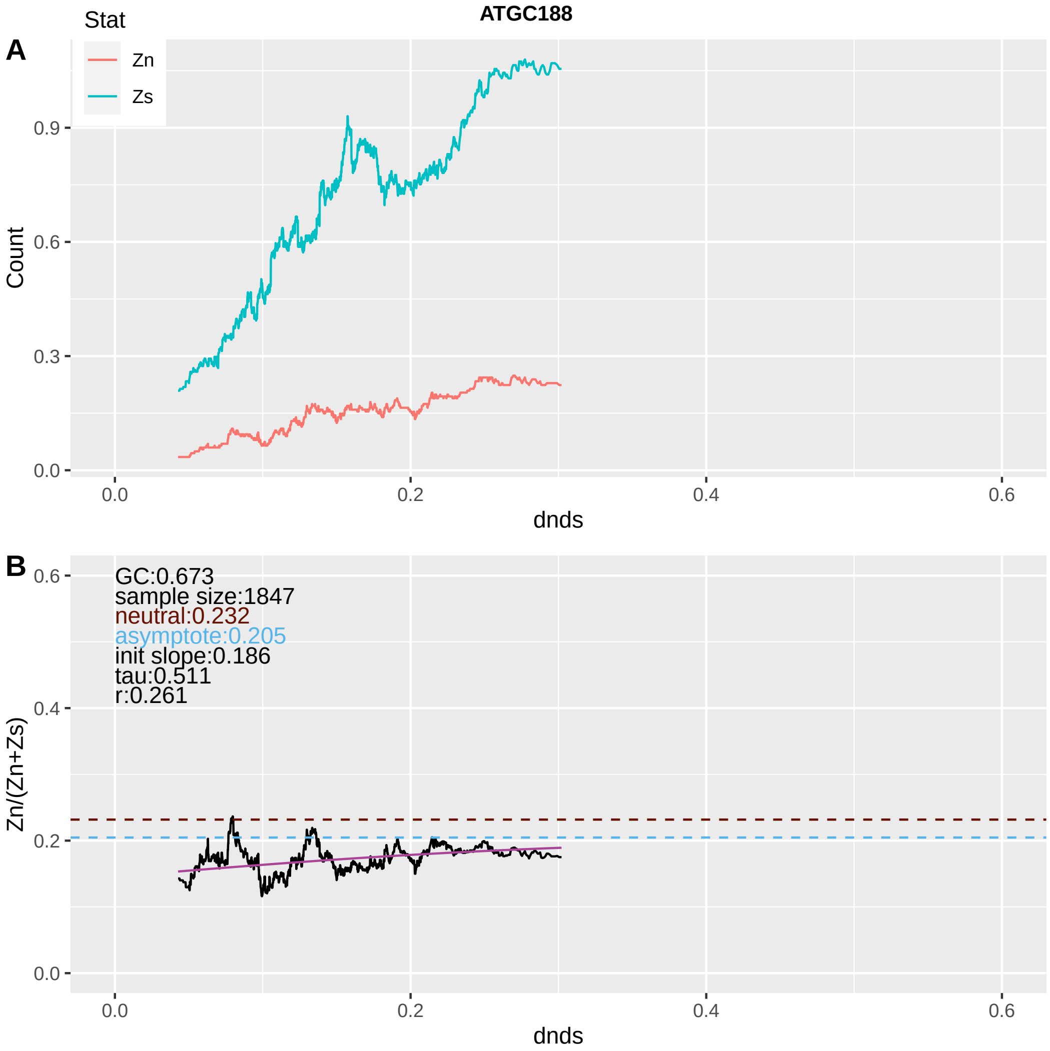

ATGC173
A
Stat
Zn
Zs
Count
0.0
0.3
0.6
0.9
0.0
0.2
0.4
0.6
dnds
B
GC:0.658
sample size:1875
neutral:0.234
asymptote:0.188
init slope:0.022
tau:0.582
r:0.417
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

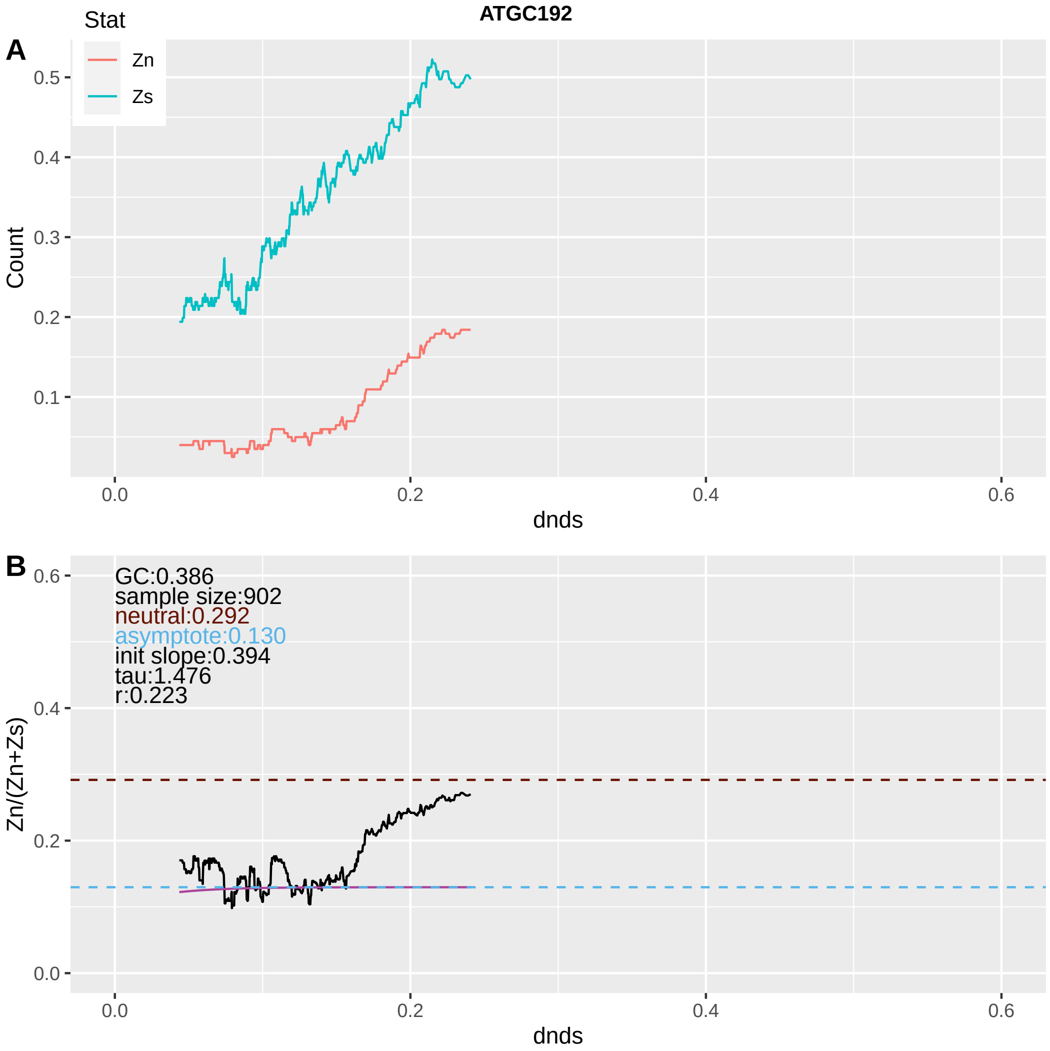

ATGC178
Stat
Zn
Zs
A
0.4
0.3
0.2
0.1
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
GC:0.319
sample size:1450
neutral:0.289
asymptote:0.219
init slope:0.009
tau:0.528
r:0.224
0.0
0.2
0.4
0.6
dnds

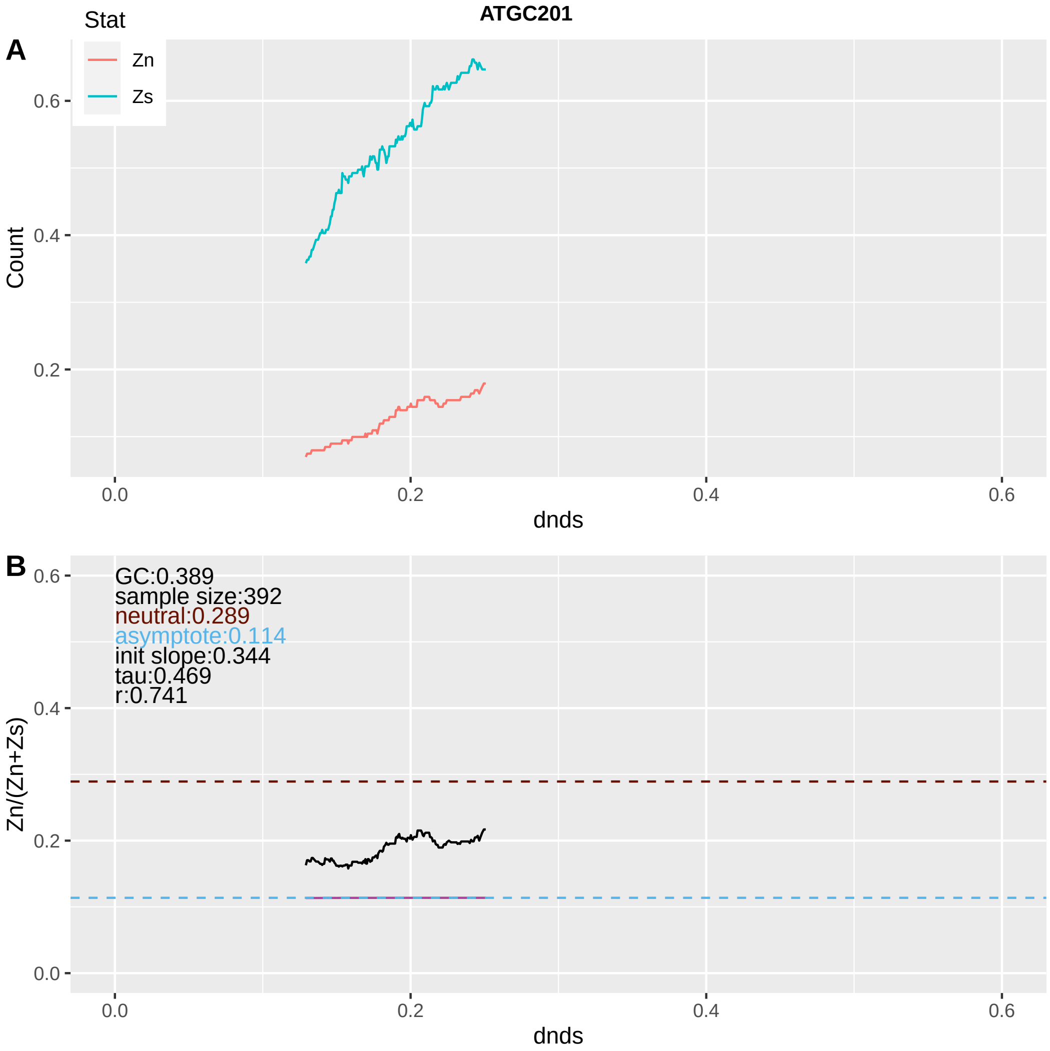

ATGC181
Stat
A
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.360
sample size:1200
neutral:0.294
asymptote:0.265
init slope:1.023
tau:1.194
r:0.665
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

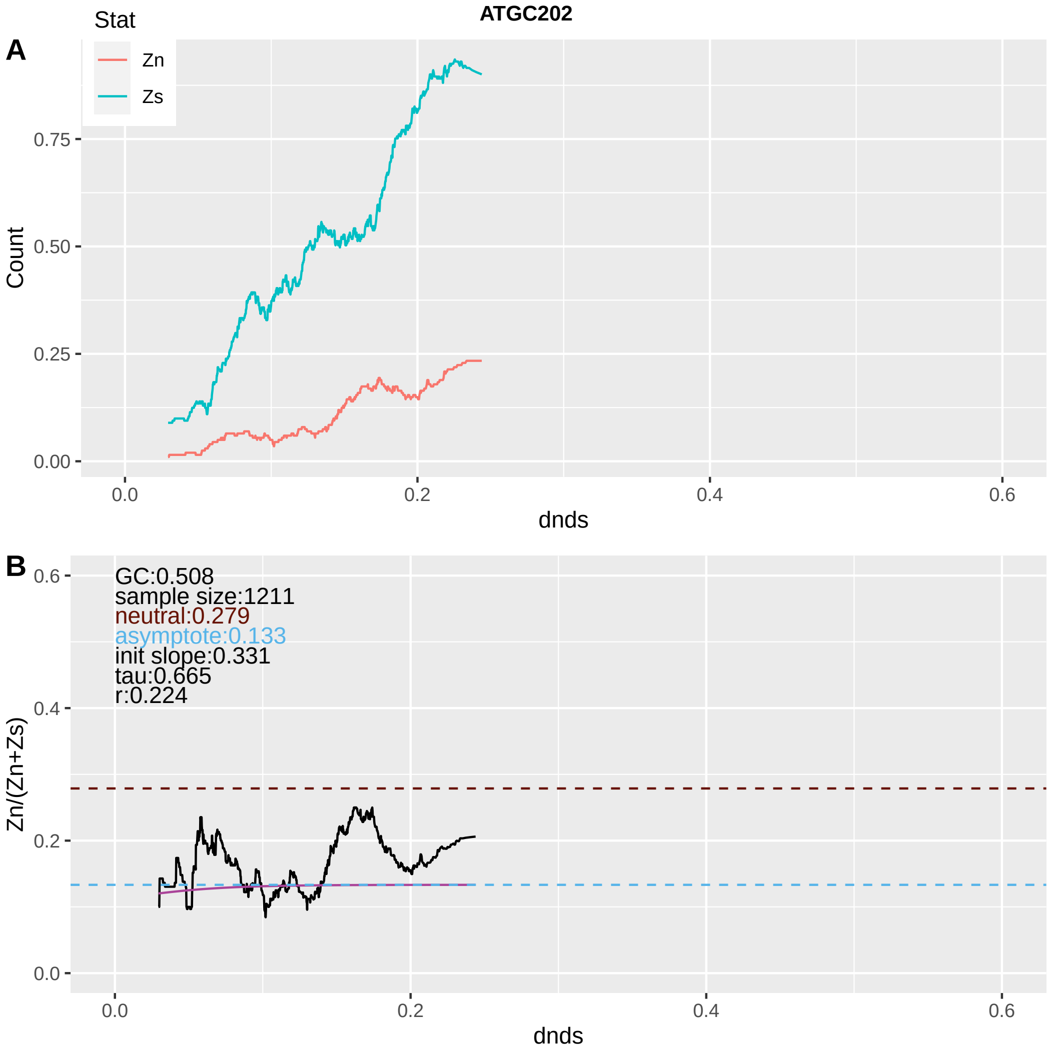

ATGC182
A
Stat
Zn
Zs
Count
0.0
0.1
0.2
0.3
0.4
0.0
0.2
0.4
0.6
dnds
B
GC:0.355
sample size:1399
neutral:0.292
asymptote:0.189
init slope:1.719
tau:1.150
r:0.820
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds


ATGC186
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6
B
GC:0.395
sample size:1511
neutral:0.292
asymptote:0.246
init slope:3.827
tau:0.974
r:0.793
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6

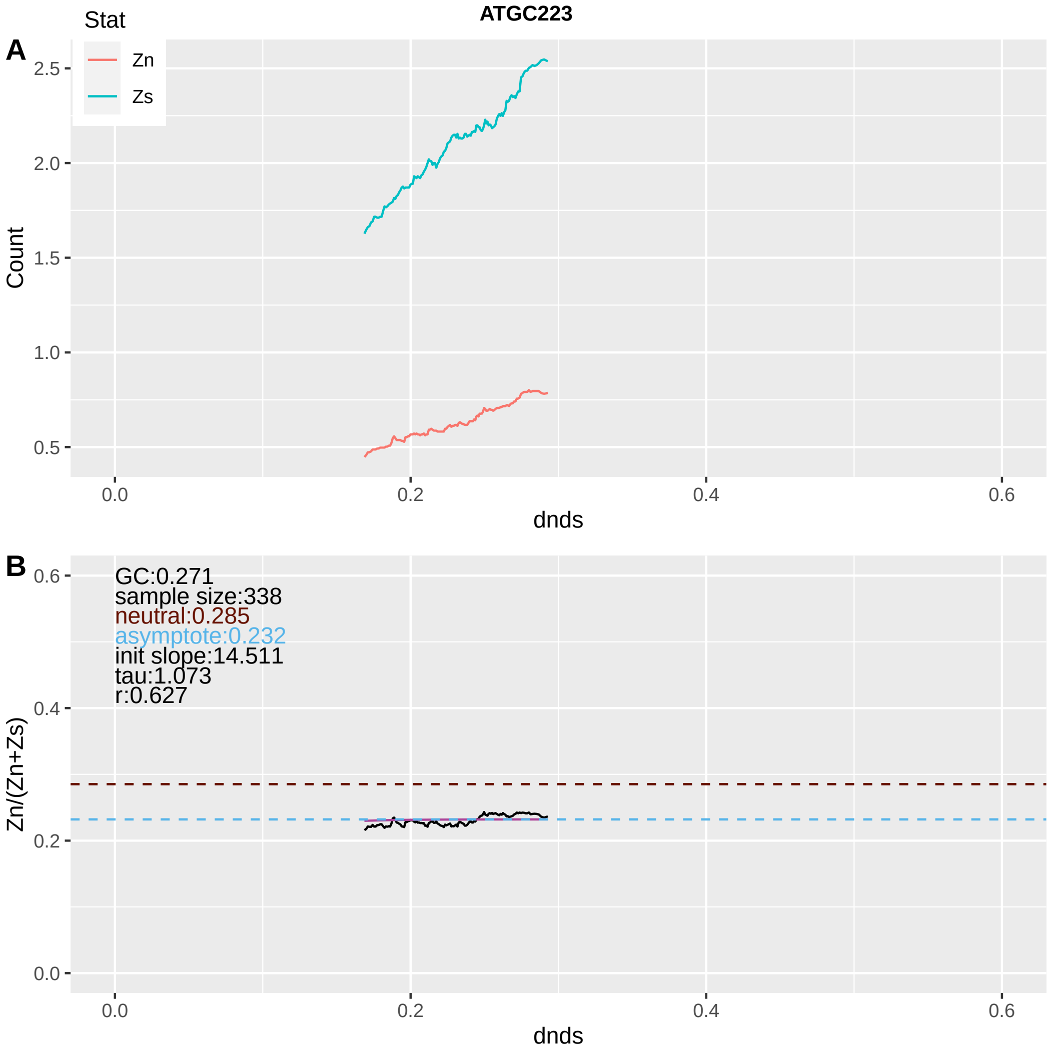

ATGC188
A
Stat
Zn
Zs
Count
0.0
0.3
0.6
0.9
0.0
0.2
0.4
0.6
dnds
B
GC:0.673
sample size:1847
neutral:0.232
asymptote:0.205
init slope:0.186
tau:0.511
r:0.261
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

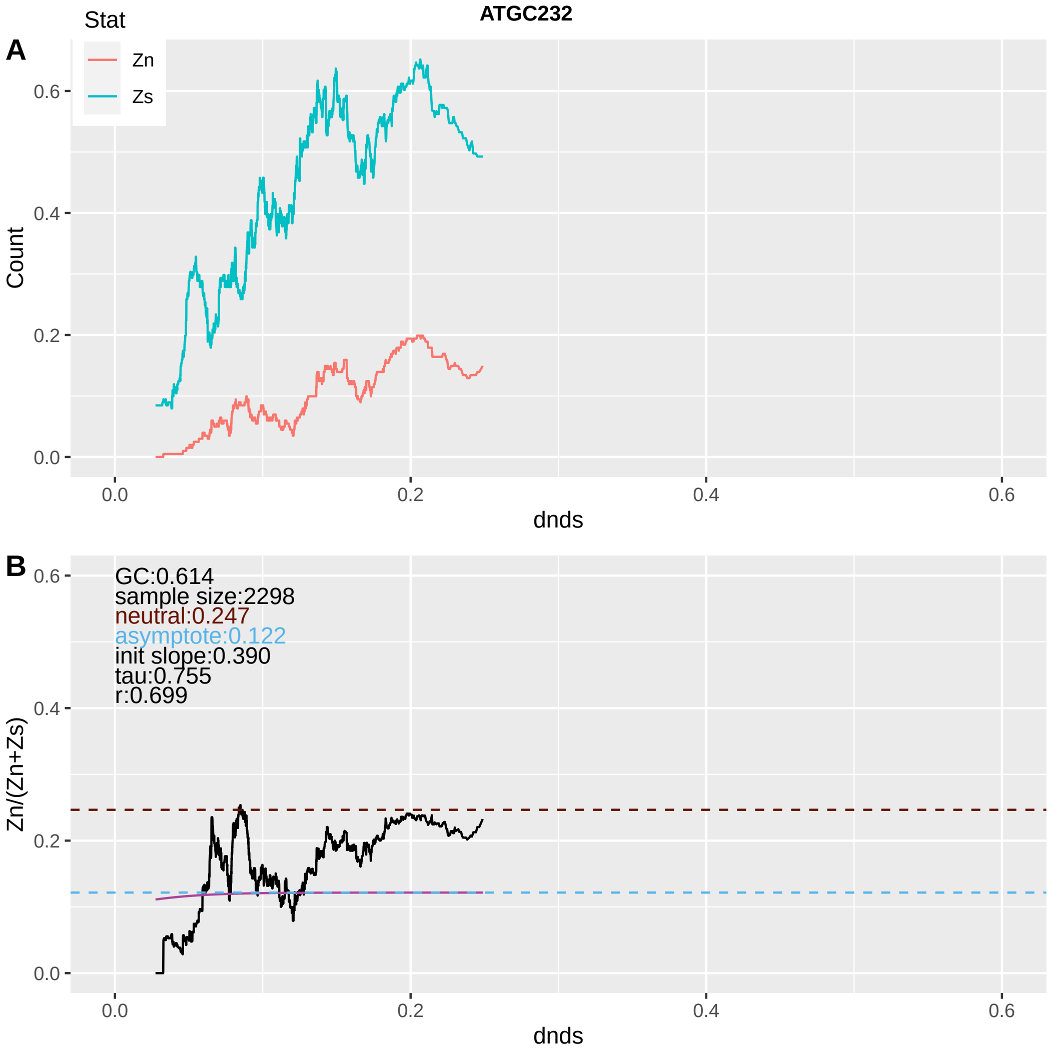

ATGC192
A
Stat
Zn
Zs
Count
0.5
0.4
0.3
0.2
0.1
0.0
0.2
0.4
0.6
dnds
B
GC:0.386
sample size:902
neutral:0.292
asymptote:0.130
init slope:0.394
tau:1.476
r:0.223
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

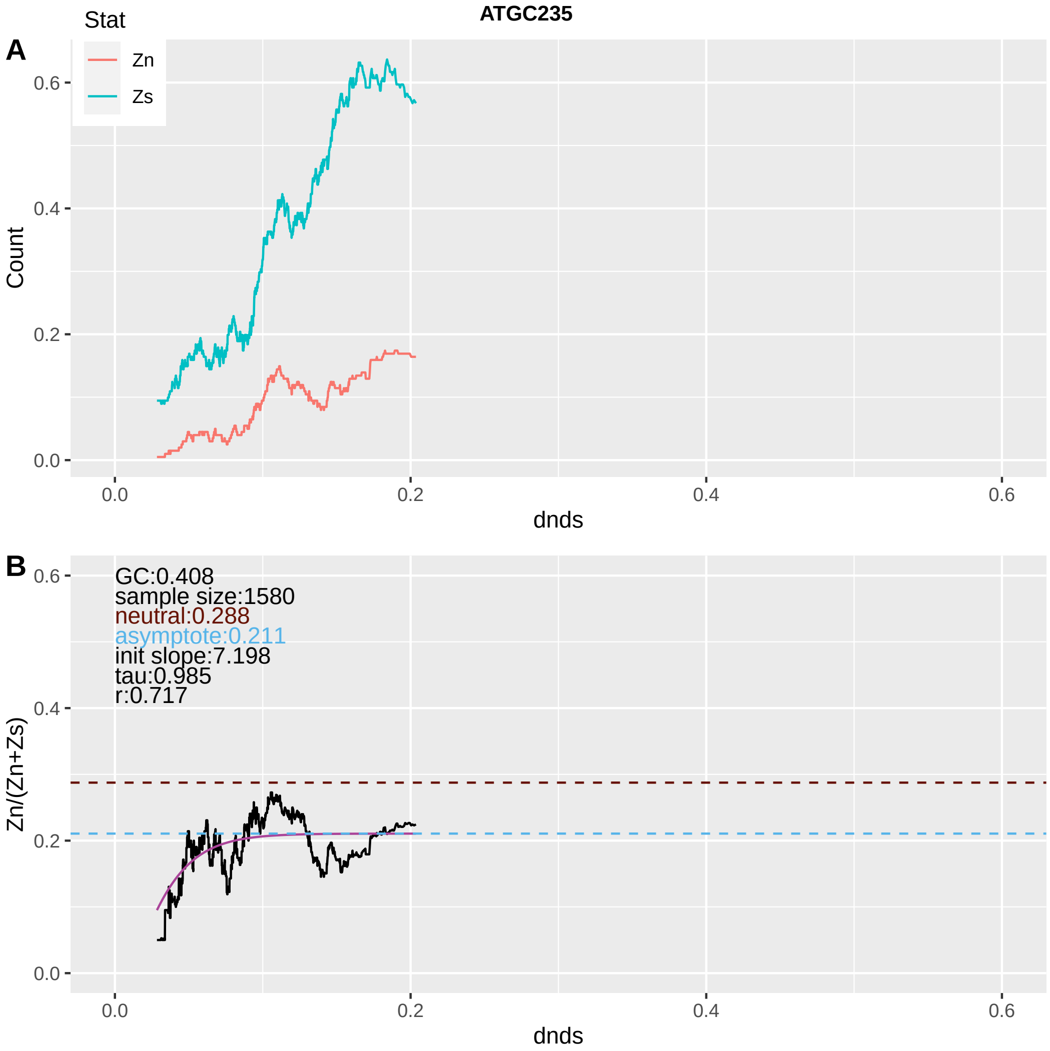

ATGC201
Stat
A
Zn
Zs
Count
0.6
0.4
0.2
0.0
0.2
0.4
0.6
dnds
B
GC:0.389
sample size:392
neutral:0.289
asymptote:0.114
init slope:0.344
tau:0.469
r:0.741
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

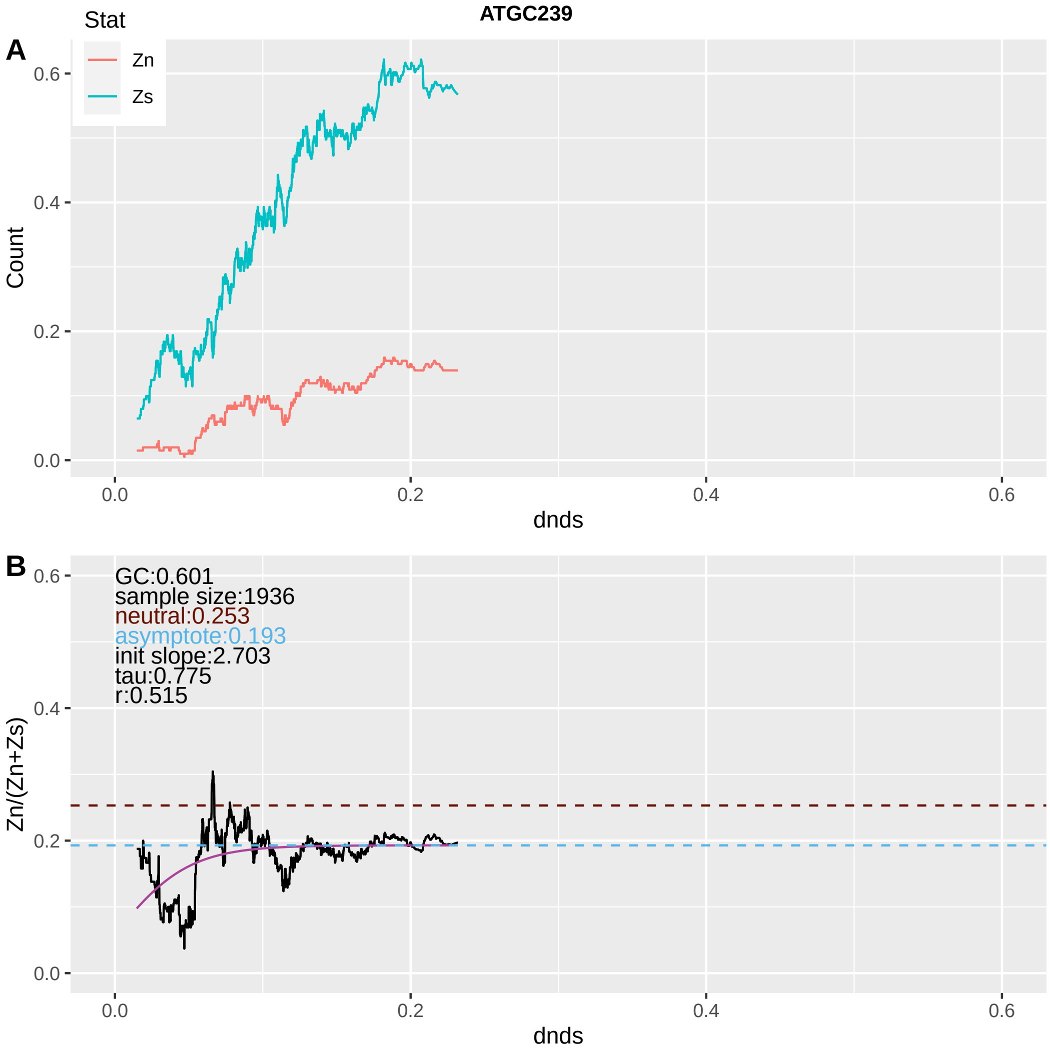


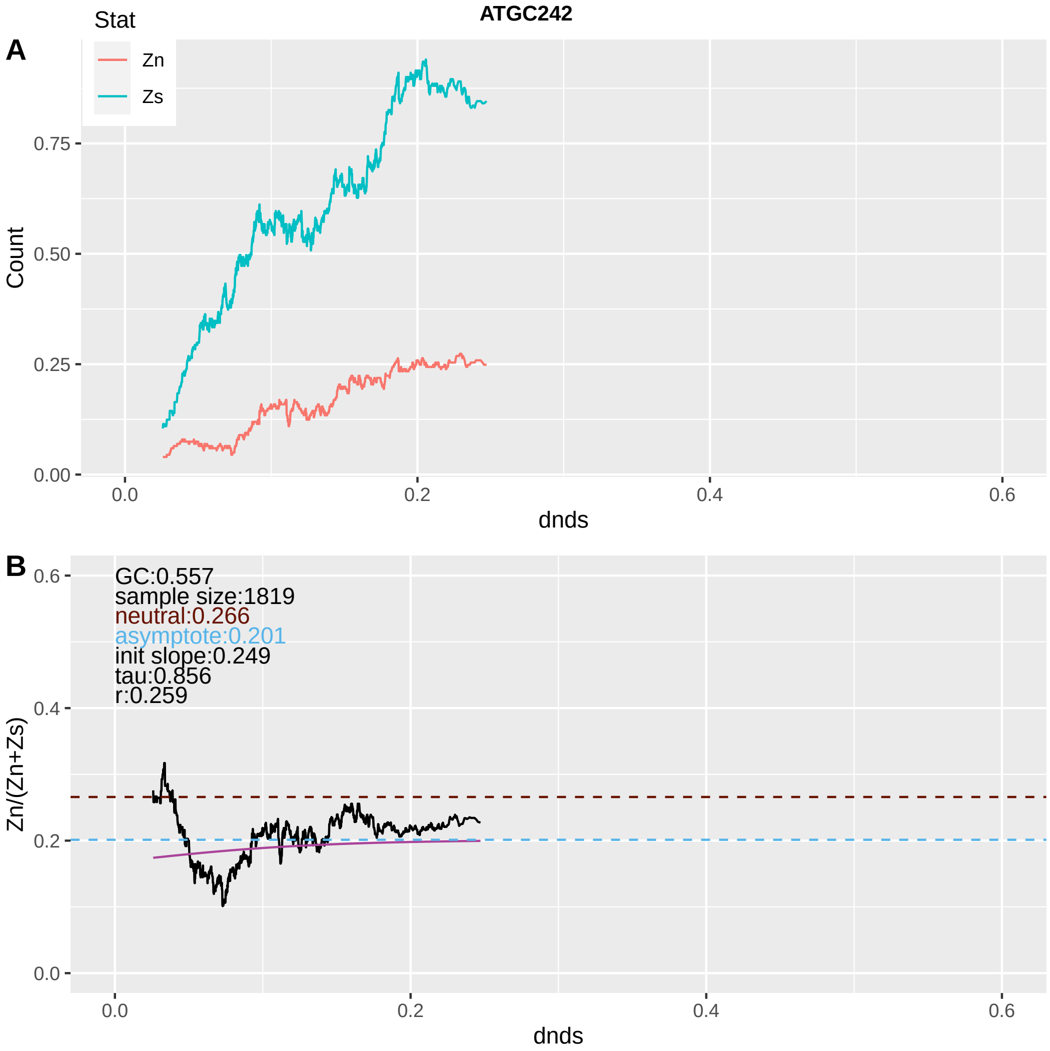
ATGC202
A
Stat
Zn
Zs
Count
0.00
0.25
0.50
0.75
0.0
0.2
0.4
0.6
dnds
B
GC:0.508
sample size:1211
neutral:0.279
asymptote:0.133
init slope:0.331
tau:0.665
r:0.224
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

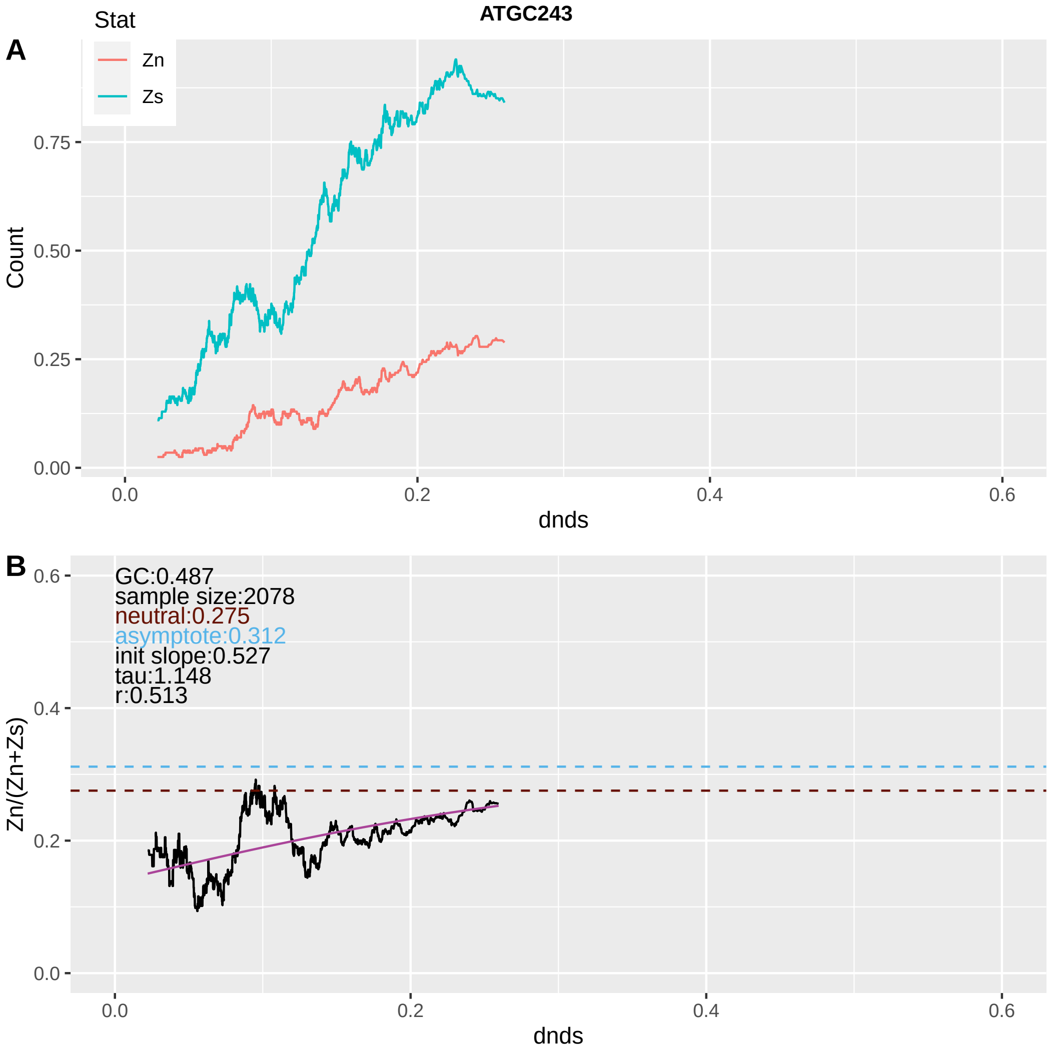

ATGC223
Stat
Zn
Zs
A
Count
2.5
2.0
1.5
1.0
0.5
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
GC:0.271
sample size:338
neutral:0.285
asymptote:0.232
init slope:14.511
tau:1.073
r:0.627
0.0
0.2
0.4
0.6
dnds

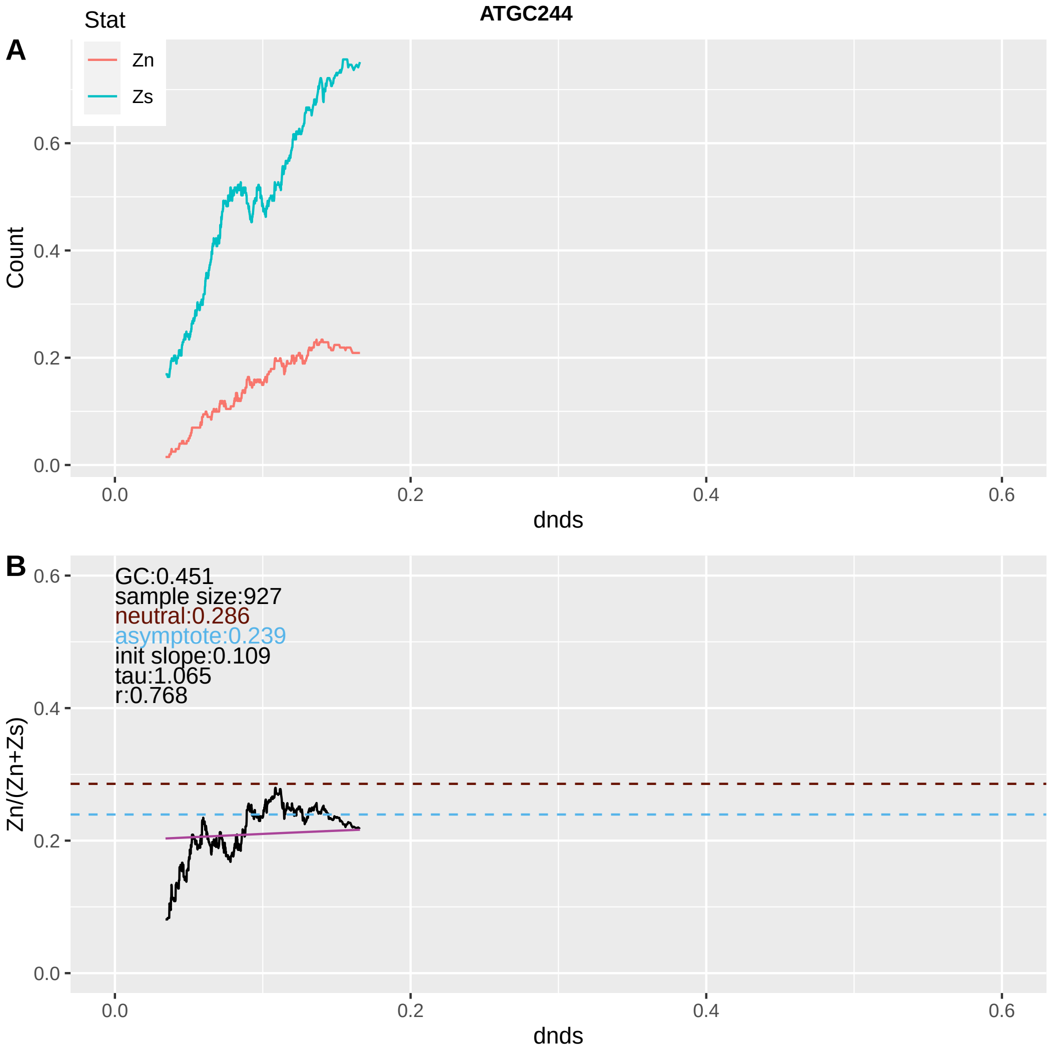

ATGC232
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds
B
GC:0.614
sample size:2298
neutral:0.247
asymptote:0.122
init slope:0.390
tau:0.755
r:0.699
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

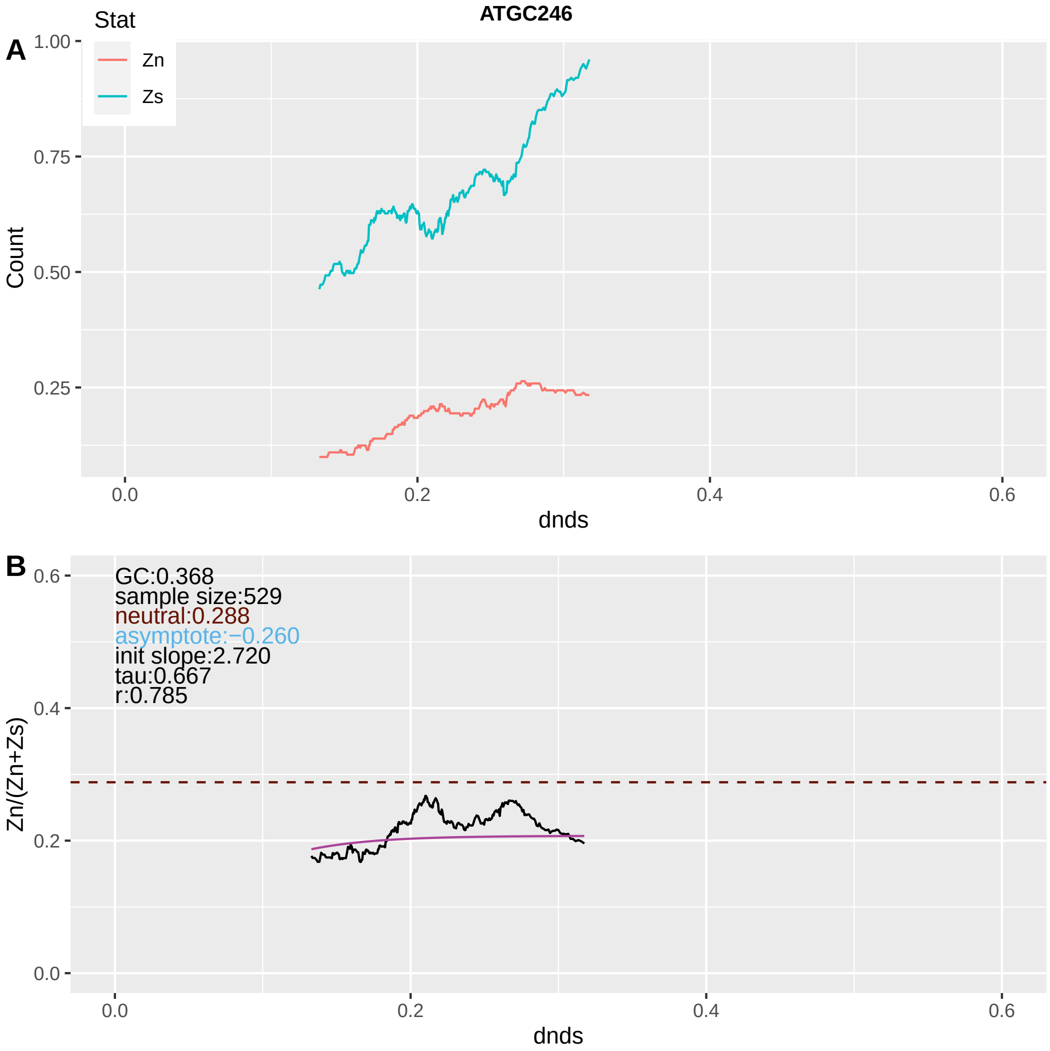

ATGC235
Stat
A
Zn
Zs
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.408
sample size:1580
neutral:0.288
asymptote:0.211
init slope:7.198
tau:0.985
r:0.717
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

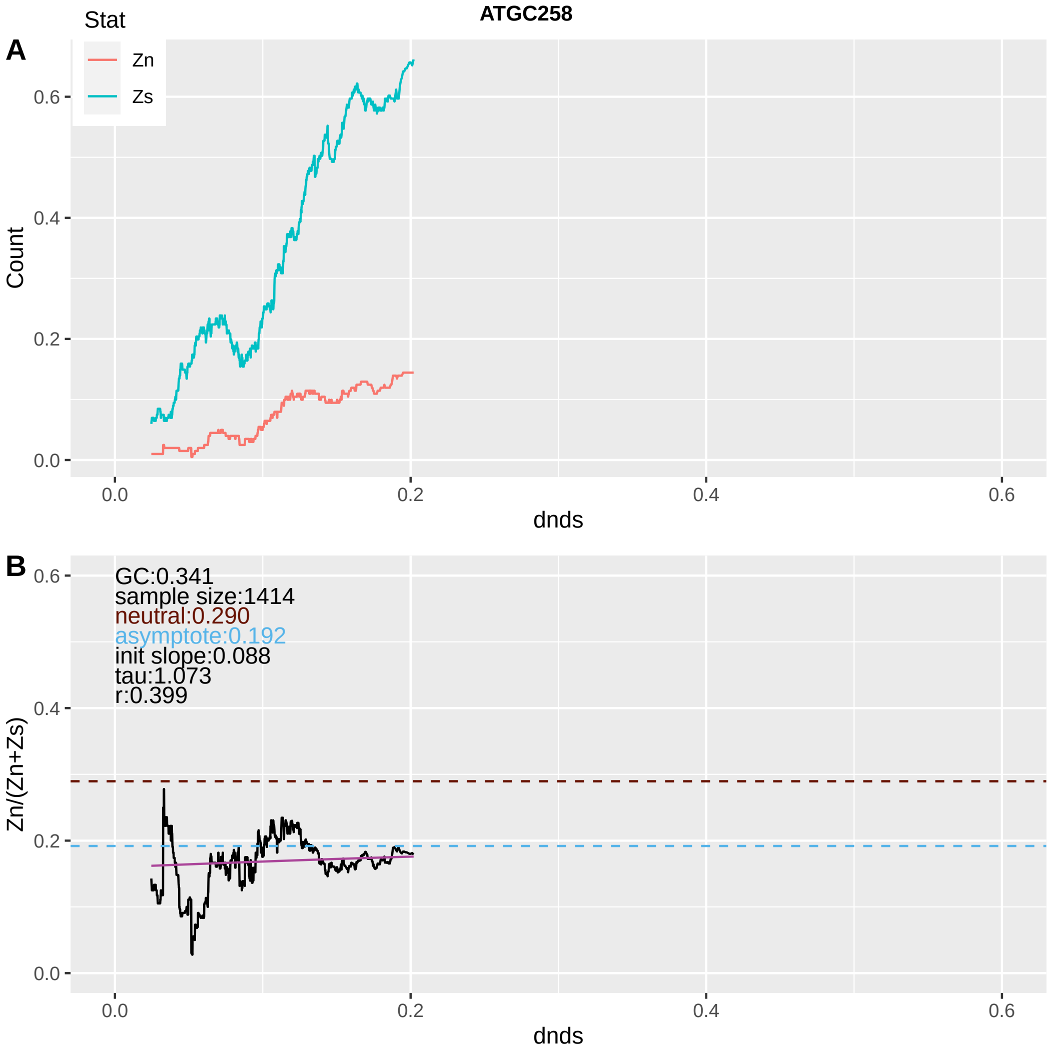

ATGC239
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6
B
GC:0.601
sample size:1936
neutral:0.253
asymptote:0.193
init slope:2.703
tau:0.775
r:0.515
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6

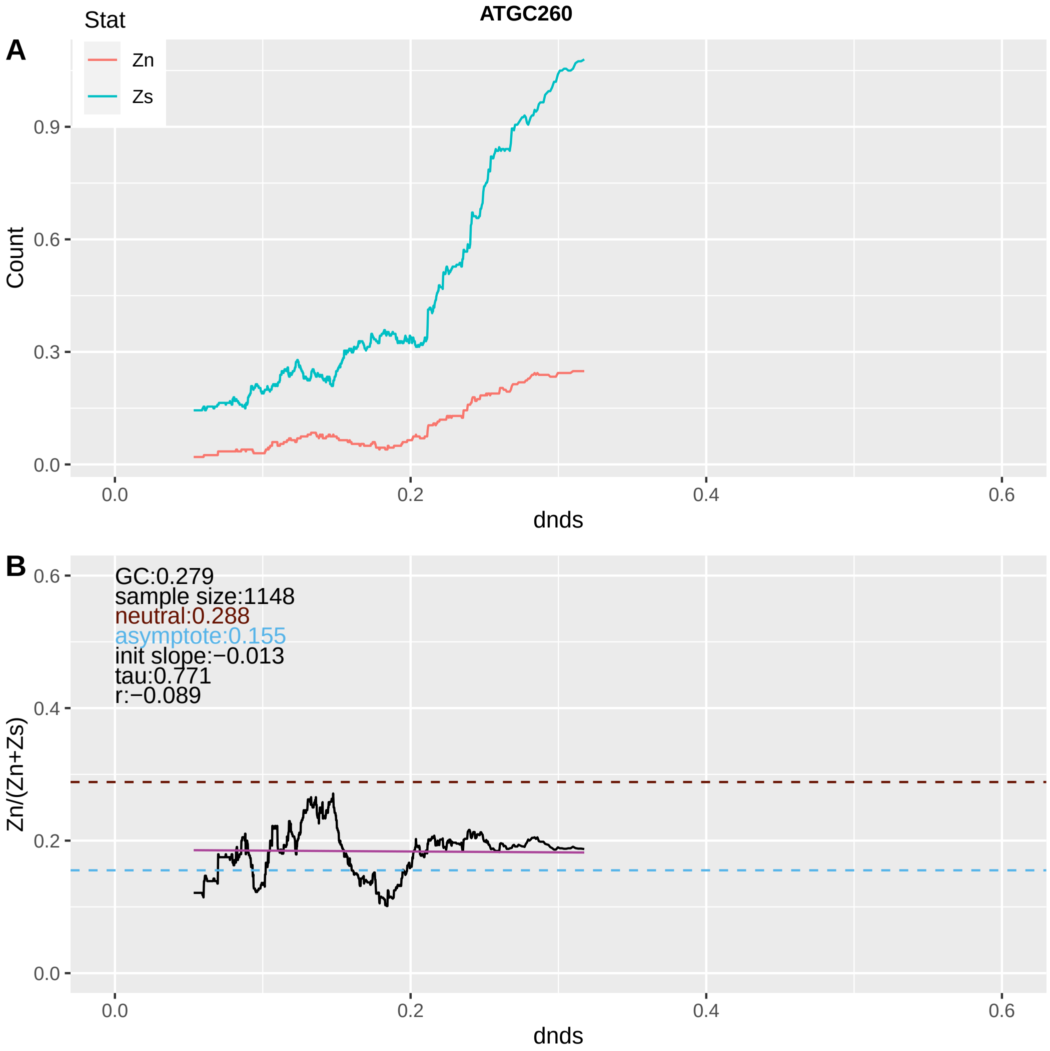

ATGC242
A
Stat
Zn
Zs
Count
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.557
sample size:1819
neutral:0.266
asymptote:0.201
init slope:0.249
tau:0.856
r:0.259
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

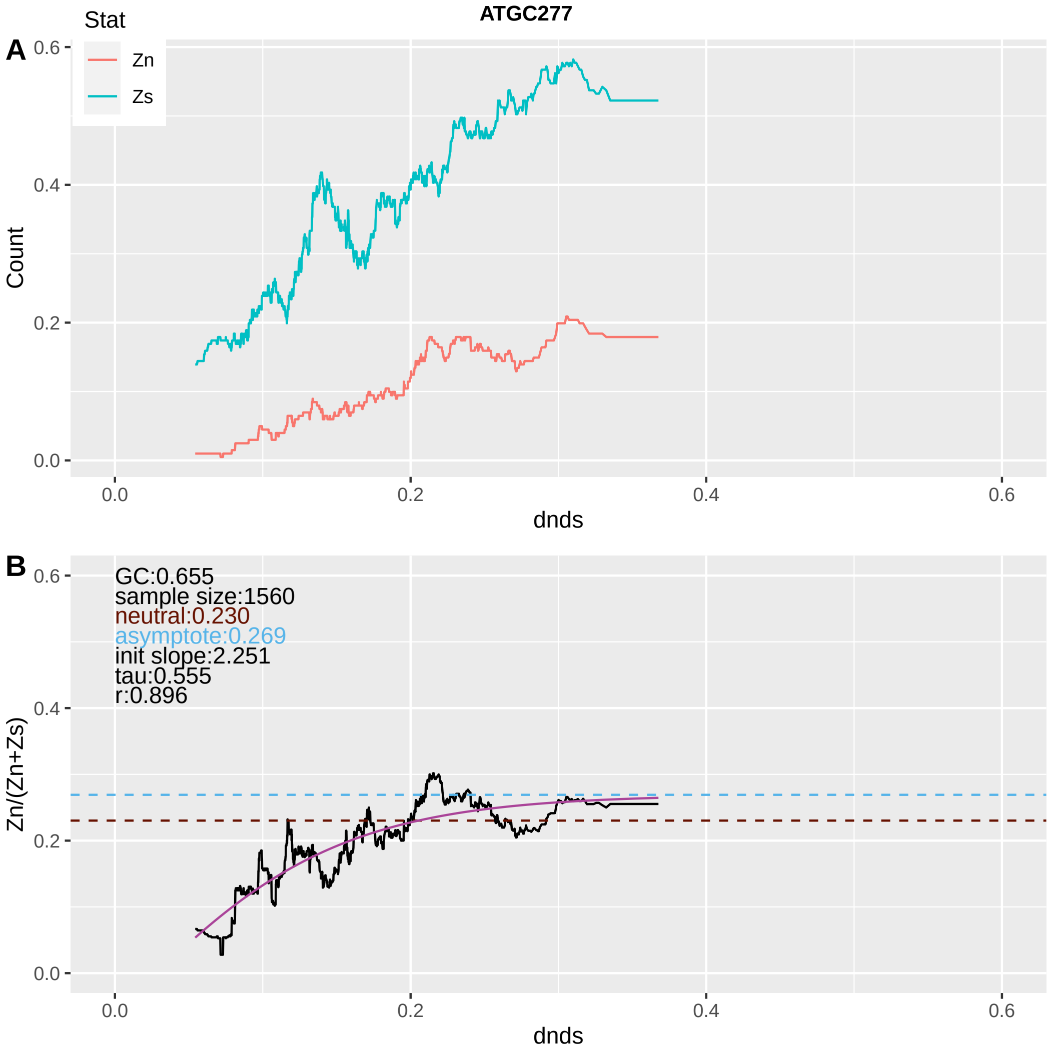

ATGC243
A
Stat
Zn
Zs
Count
0.00
0.25
0.50
0.75
0.0
0.2
0.4
0.6
dnds
B
GC:0.487
sample size:2078
neutral:0.275
asymptote:0.312
init slope:0.527
tau:1.148
r:0.513
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

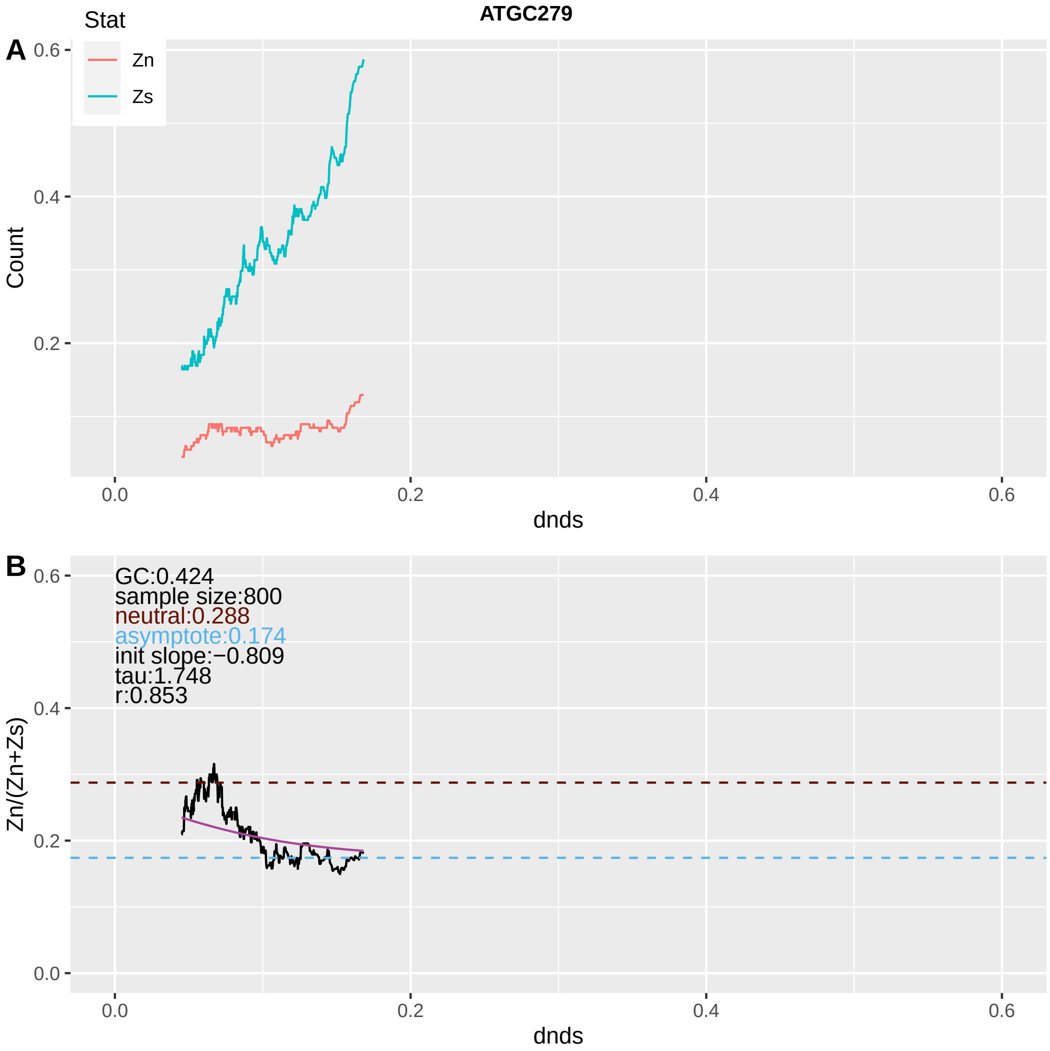

ATGC244
Stat
A
Zn
Zs
Count
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds
B
GC:0.451
sample size:927
neutral:0.286
asymptote:0.239
init slope:0.109
tau:1.065
r:0.768
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

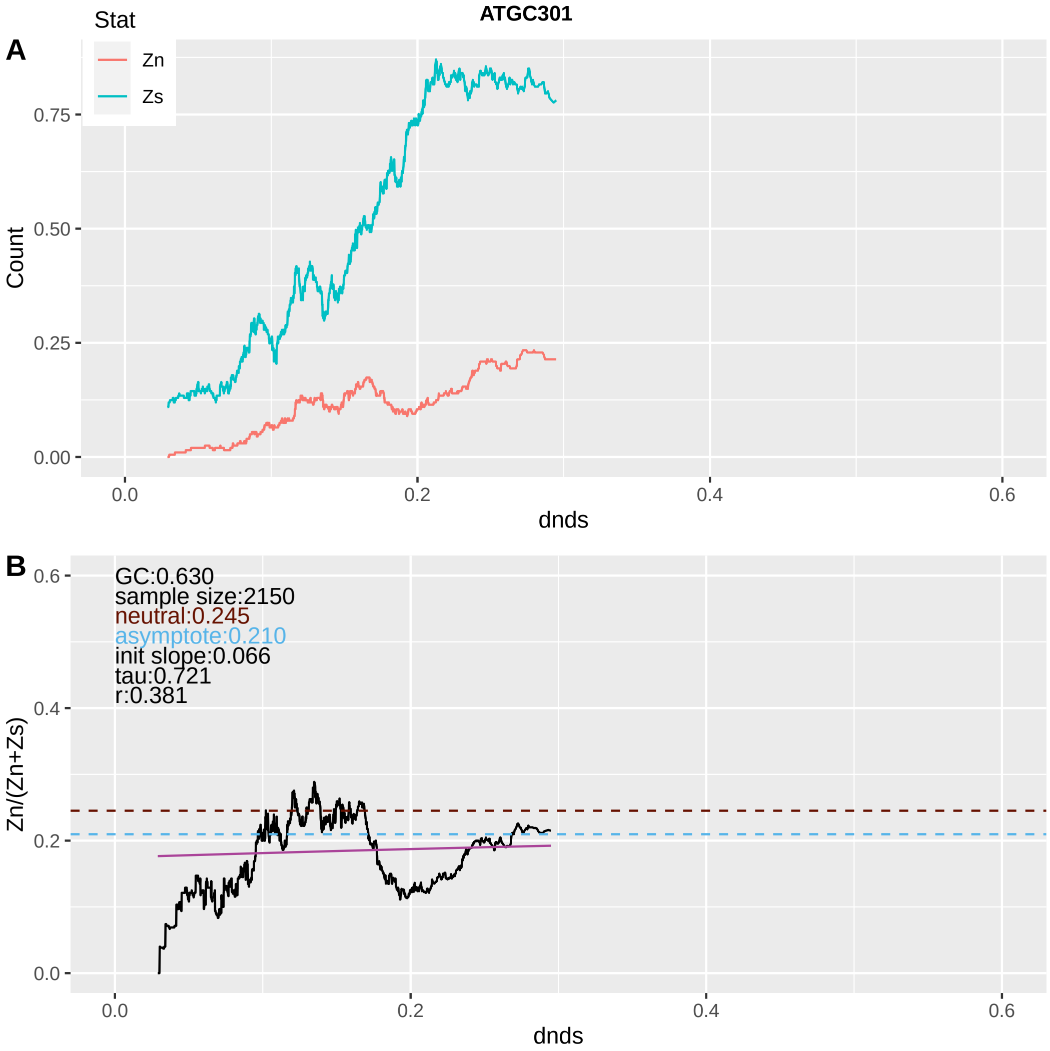

ATGC246
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.0
0.2
0.4
0.6
dnds
B
GC:0.368
sample size:529
neutral:0.288
asymptote:−0.260
init slope:2.720
tau:0.667
r:0.785
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

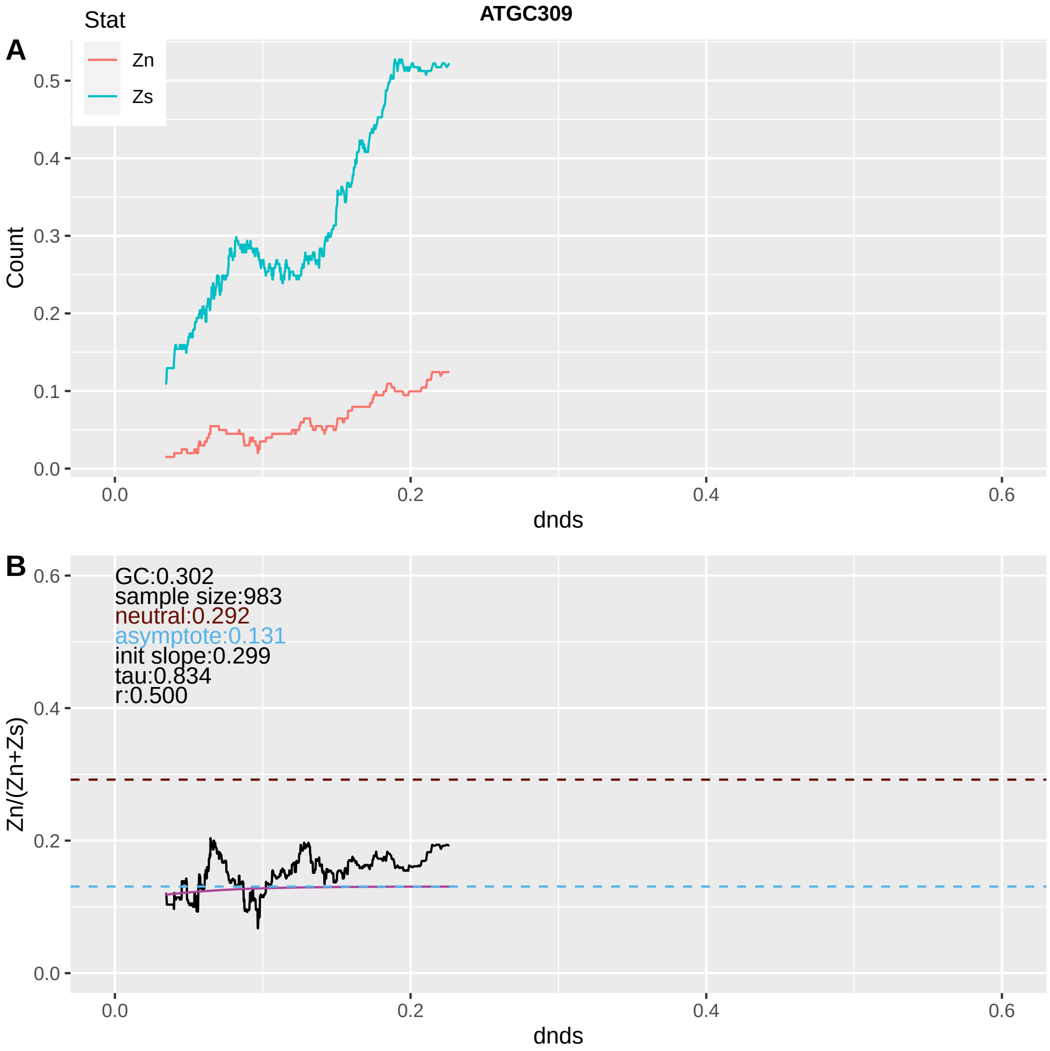

ATGC258
Stat
A
Zn
Zs
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.341
sample size:1414
neutral:0.290
asymptote:0.192
init slope:0.088
tau:1.073
r:0.399
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

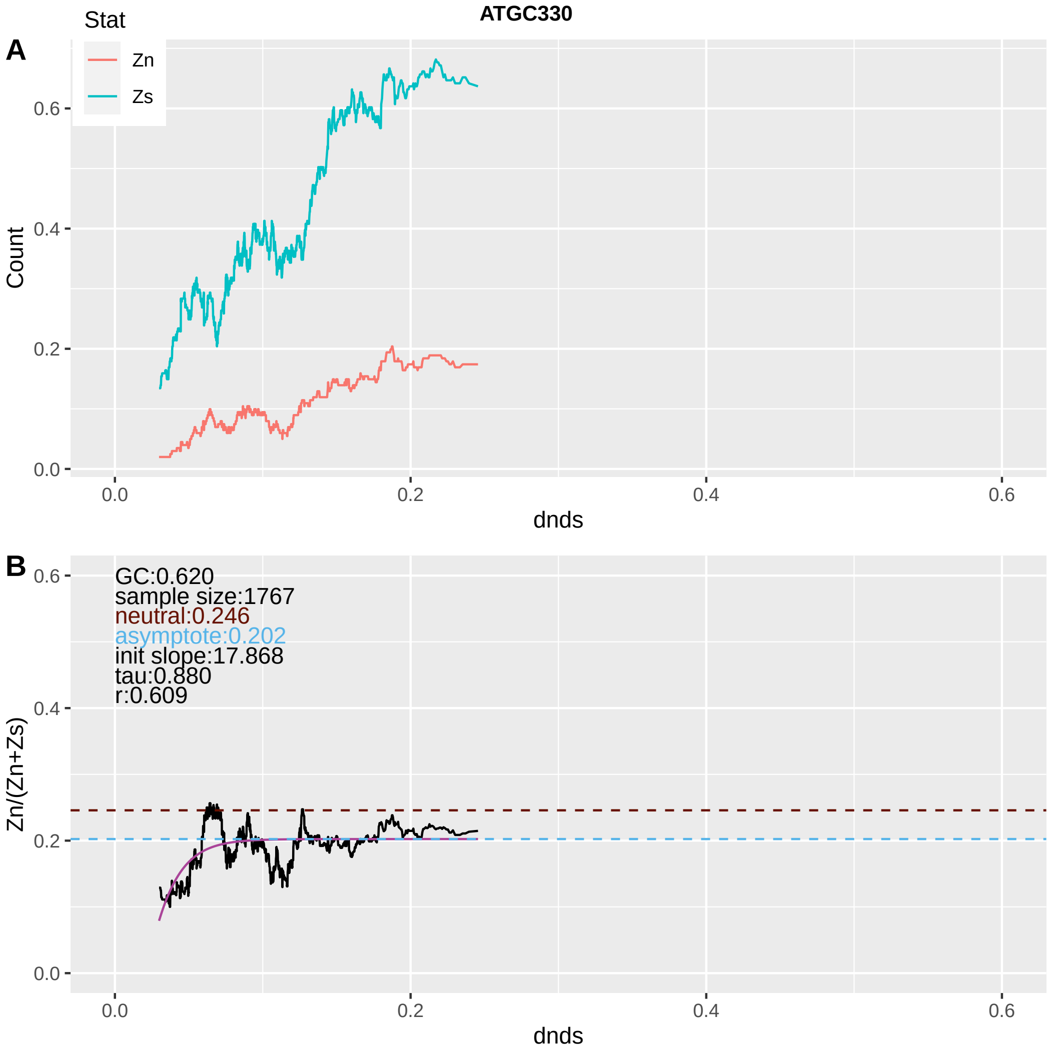

ATGC260
A
Stat
Zn
Zs
0.9
0.6
0.3
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.279
sample size:1148
neutral:0.288
asymptote:0.155
init slope:−0.013
tau:0.771
r:−0.089
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

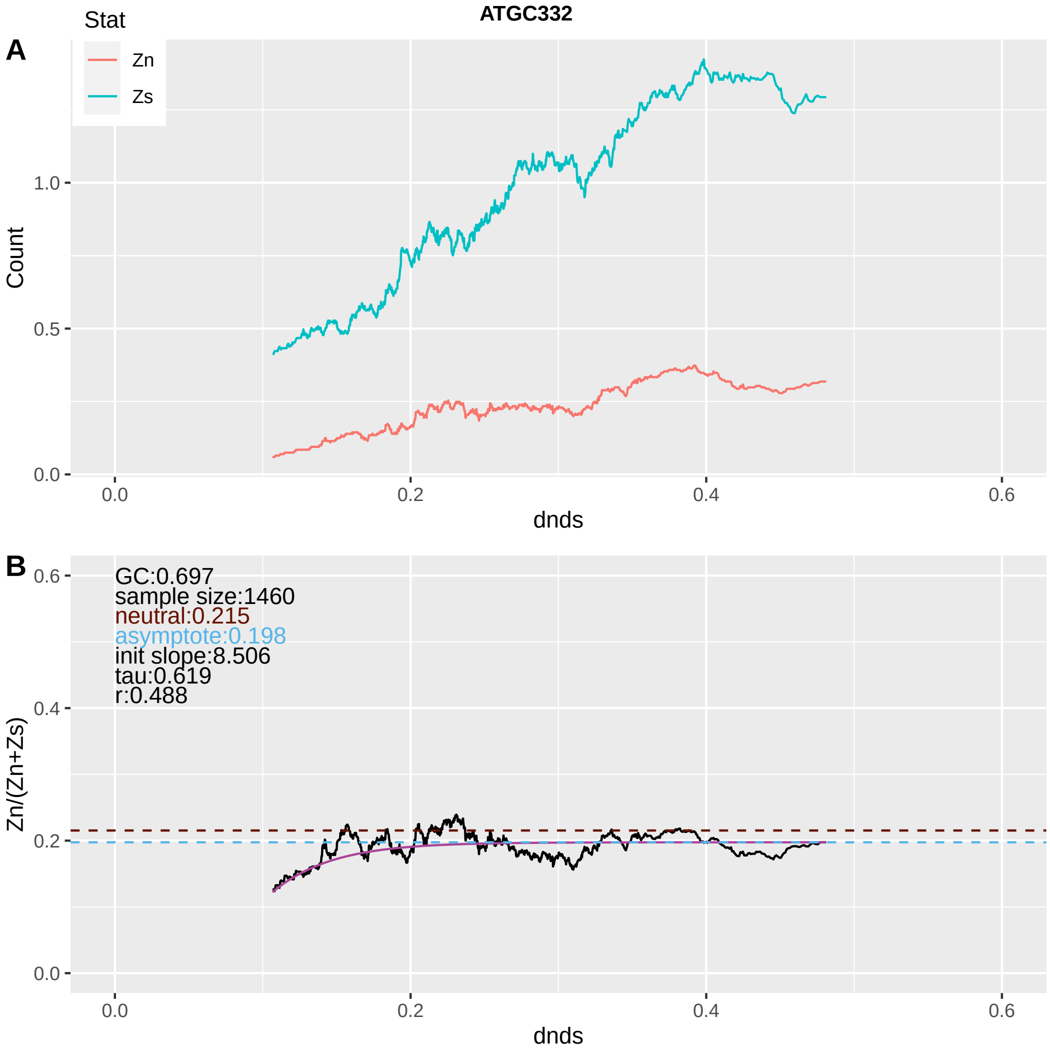

ATGC277
Stat
A
Zn
Zs
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.655
sample size:1560
neutral:0.230
asymptote:0.269
init slope:2.251
tau:0.555
r:0.896
Zn/(Zn+Zs)
dnds


ATGC279
A
Stat
Zn
Zs
Count
0.6
0.4
0.2
0.0
0.2
0.4
0.6
dnds
B
GC:0.424
sample size:800
neutral:0.288
asymptote:0.174
init slope:−0.809
tau:1.748
r:0.853
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

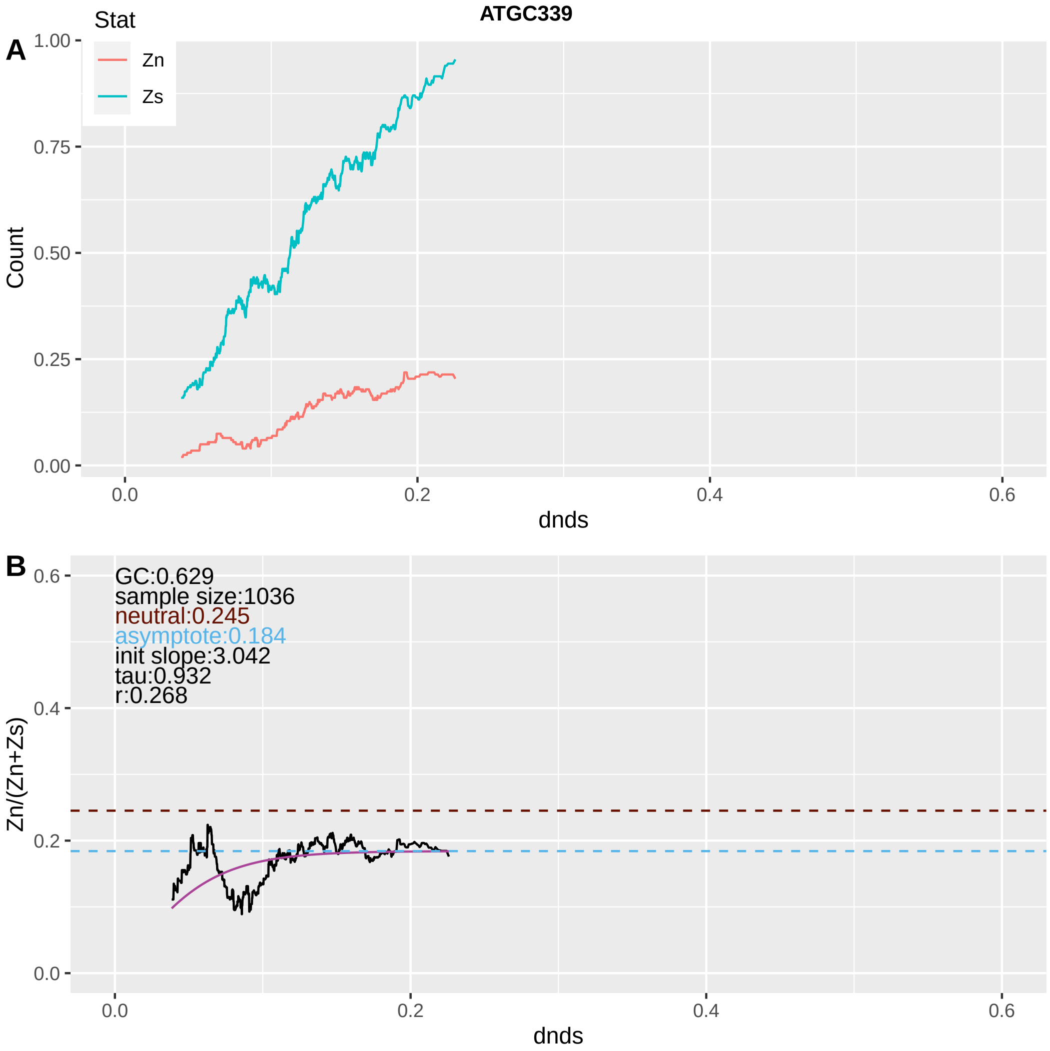

ATGC301
A
Stat
Zn
Zs
0.75
0.50
0.25
0.00
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.630
sample size:2150
neutral:0.245
asymptote:0.210
init slope:0.066
tau:0.721
r:0.381
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

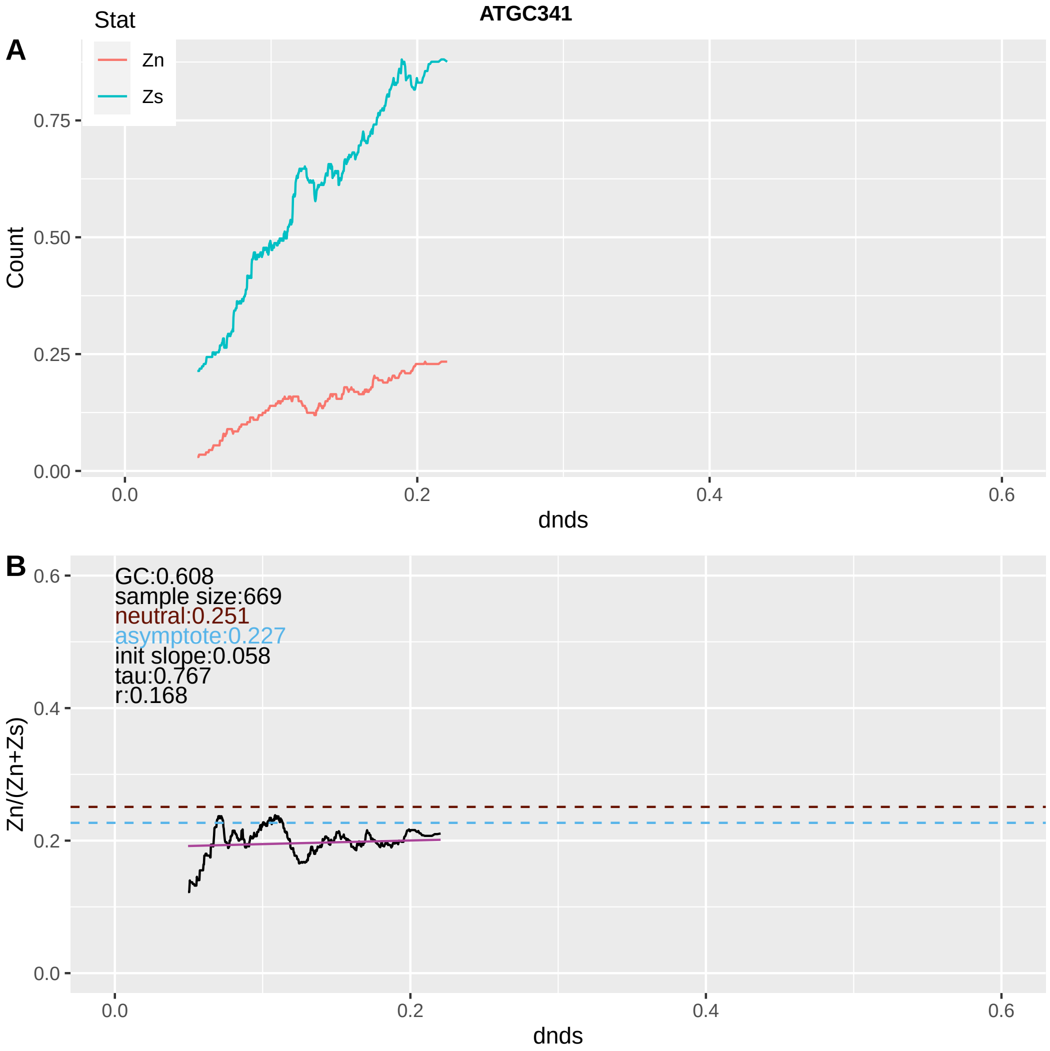

ATGC309
Stat
Zn
Zs
A
Count
0.0
0.1
0.2
0.3
0.4
0.5
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
GC:0.302
sample size:983
neutral:0.292
asymptote:0.131
init slope:0.299
tau:0.834
r:0.500
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

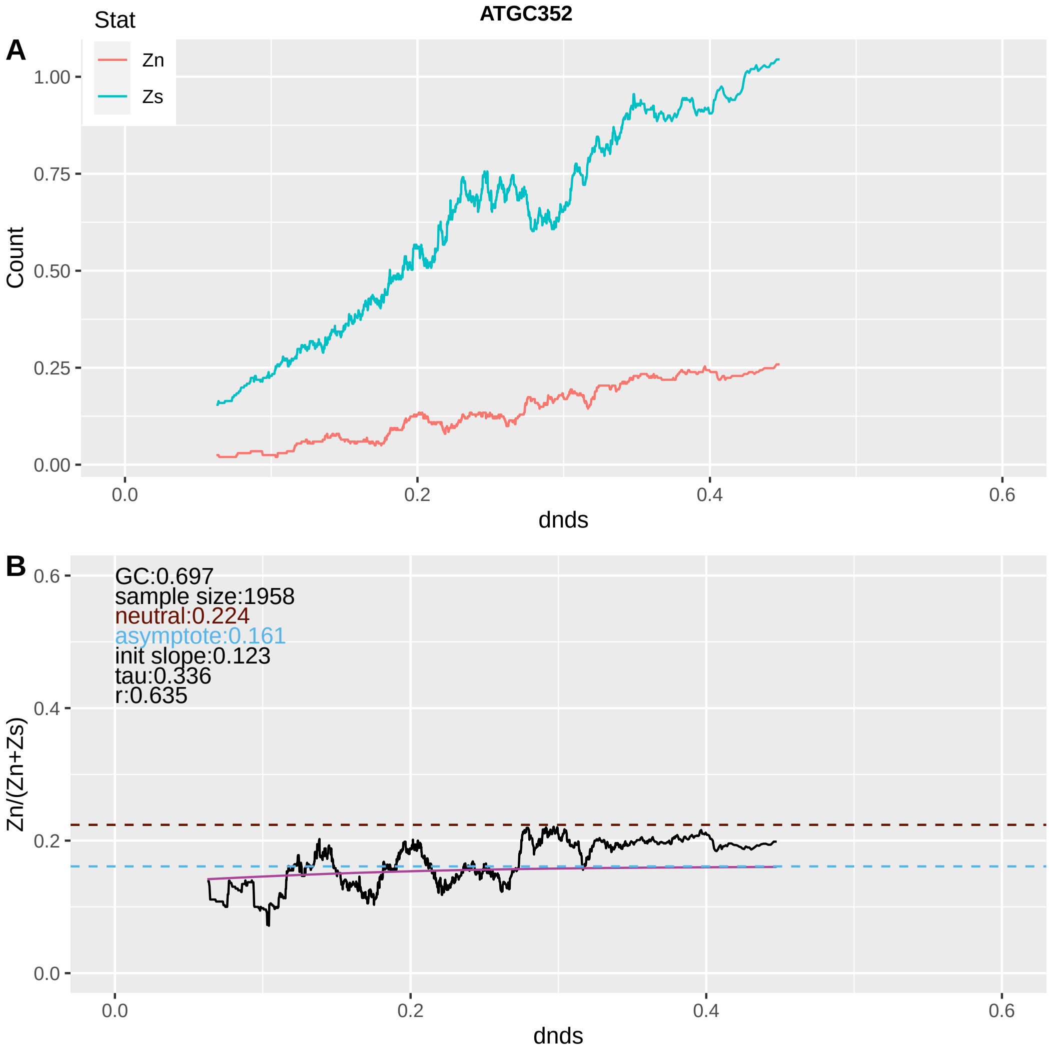

ATGC330
Stat
A
Zn
Zs
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.620
sample size:1767
neutral:0.246
asymptote:0.202
init slope:17.868
tau:0.880
r:0.609
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

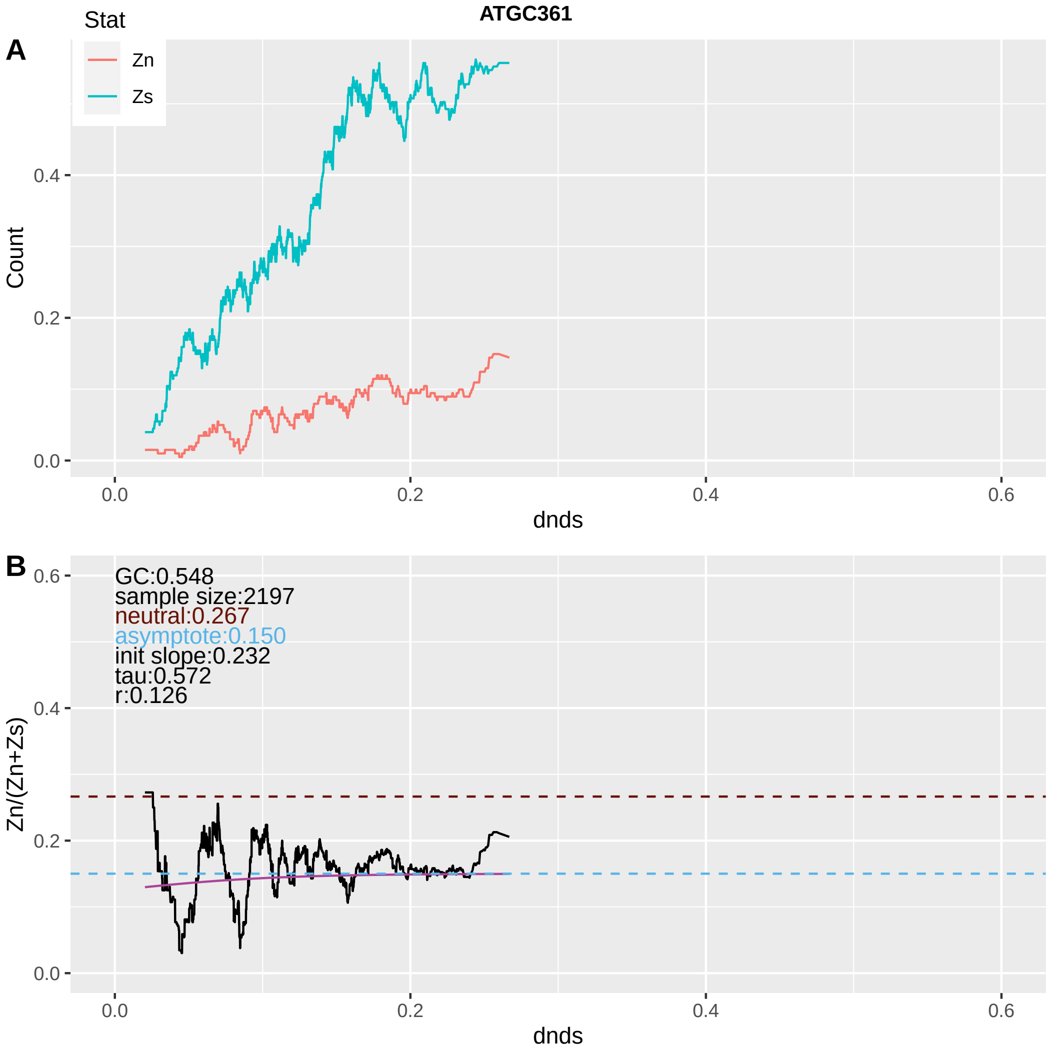


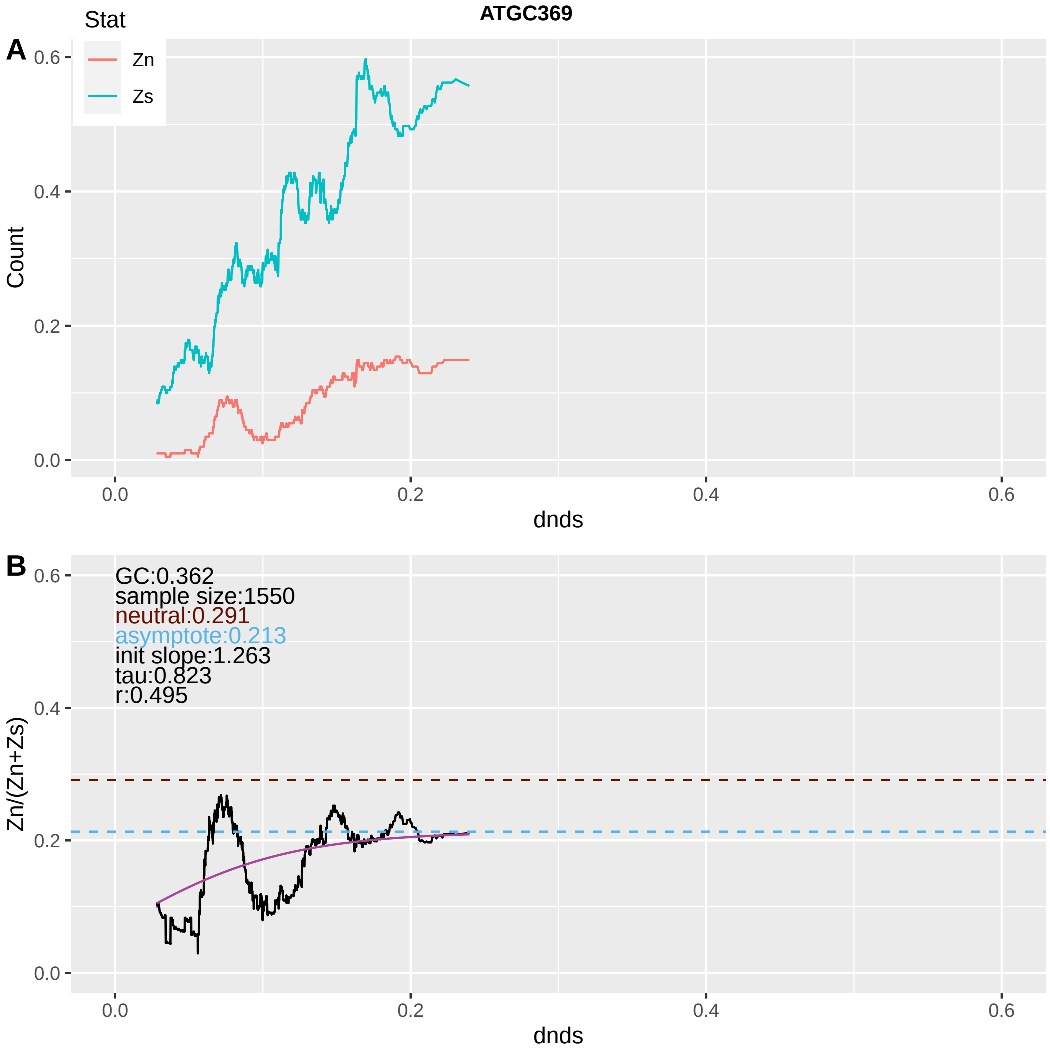
ATGC332
Stat
Zn
Zs
A
Count
1.0
0.5
0.0
0.0
0.2
0.4
0.6
dnds
B
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
GC:0.697
sample size:1460
neutral:0.215
asymptote:0.198
init slope:8.506
tau:0.619
r:0.488
0.0
0.2
0.4
0.6
dnds

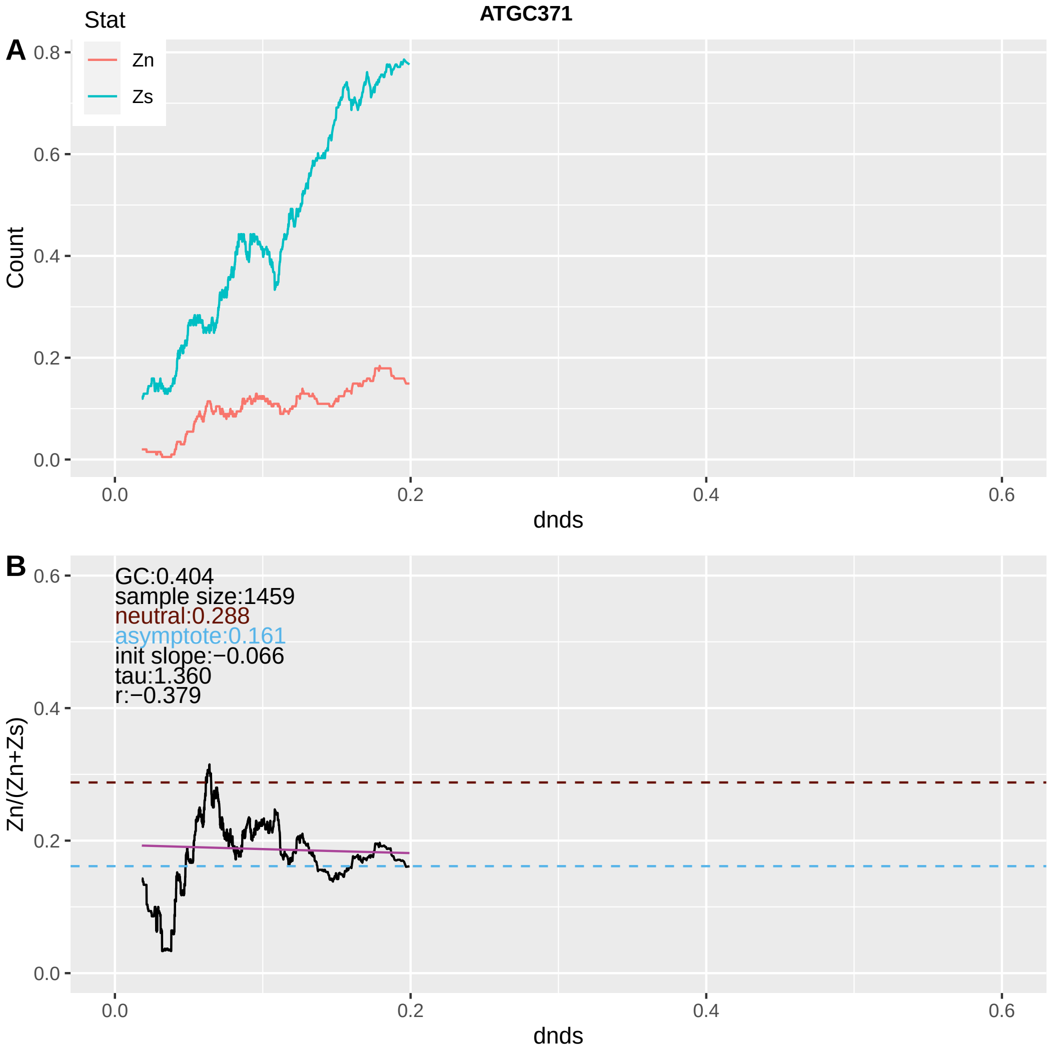

ATGC339
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.629
sample size:1036
neutral:0.245
asymptote:0.184
init slope:3.042
tau:0.932
r:0.268
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

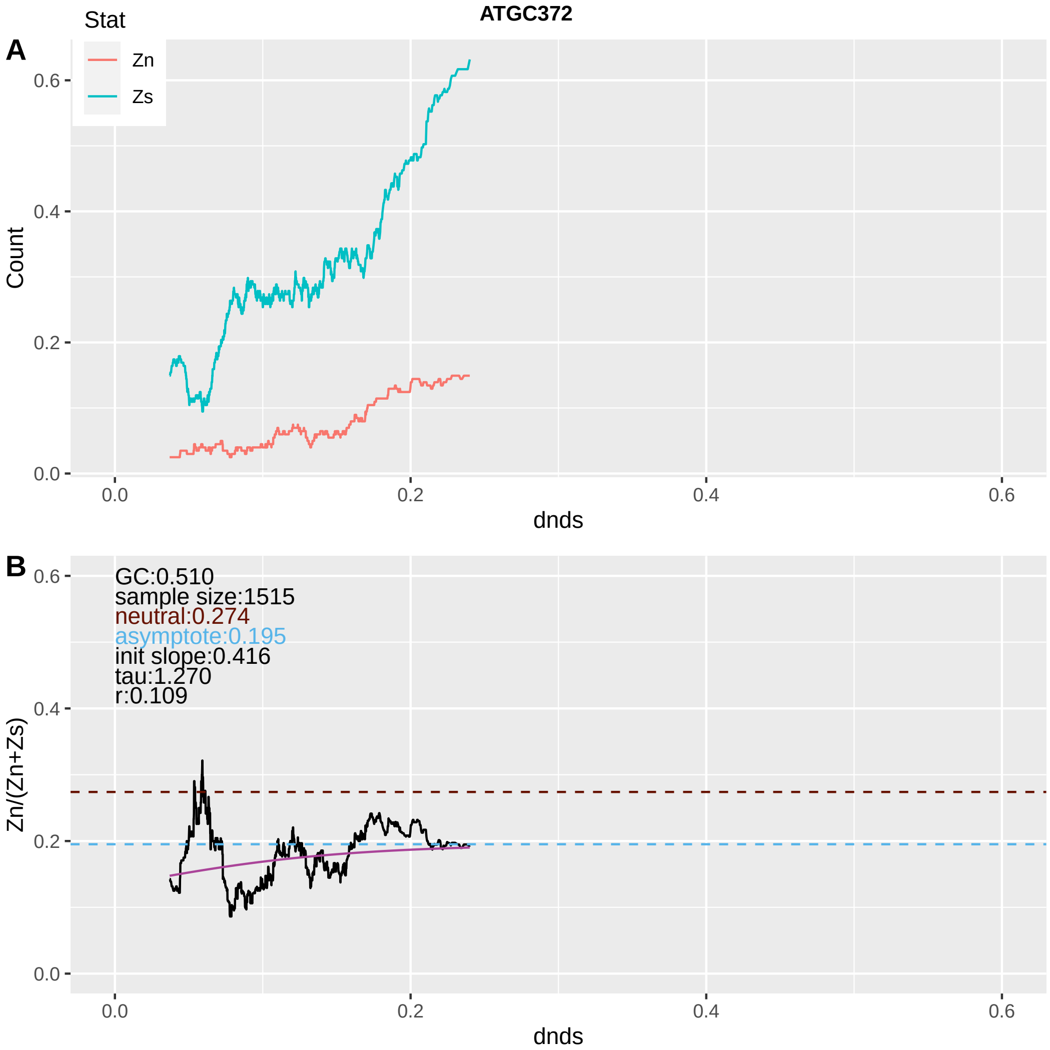

ATGC341
A
Stat
Zn
Zs
Count
0.00
0.25
0.50
0.75
0.0
0.2
0.4
0.6
dnds
B
GC:0.608
sample size:669
neutral:0.251
asymptote:0.227
init slope:0.058
tau:0.767
r:0.168
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

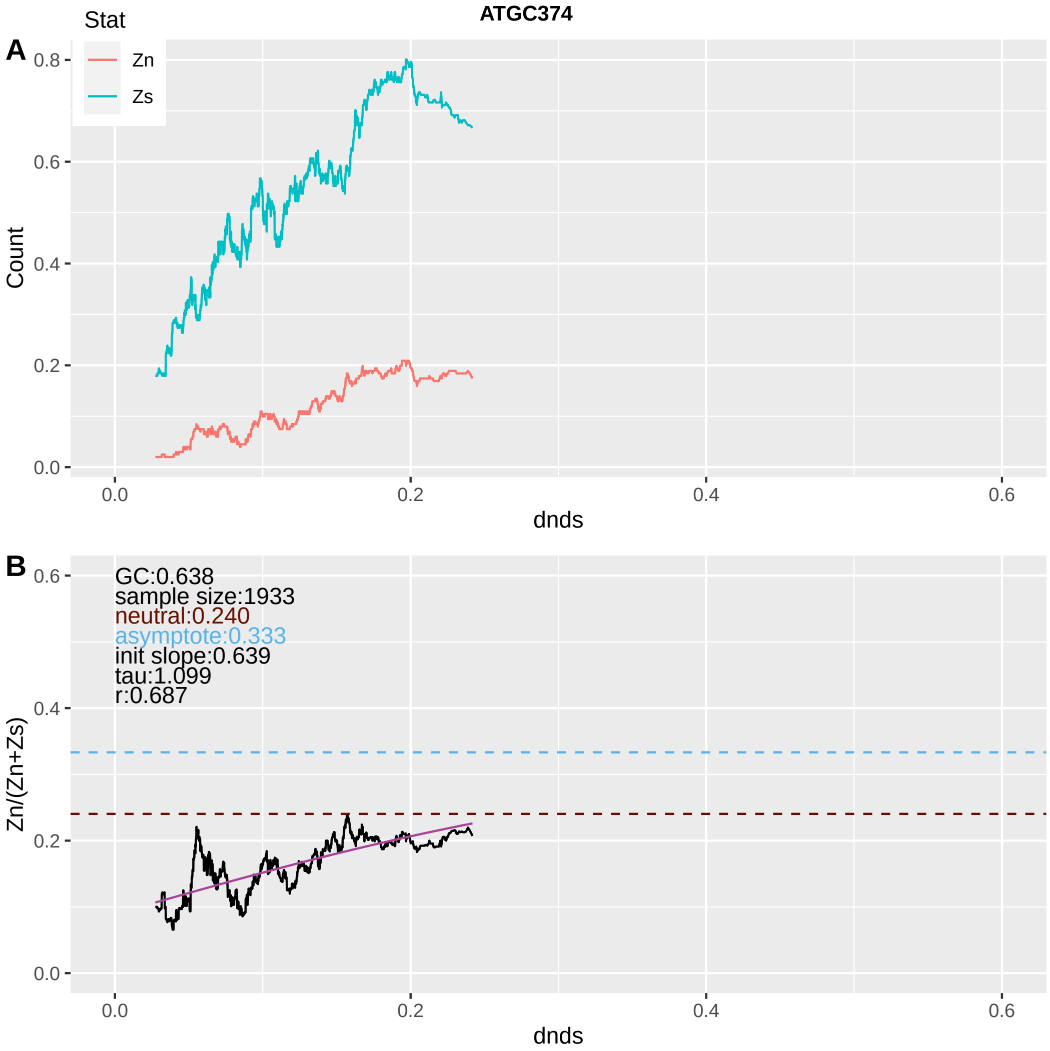

ATGC352
Stat
A
Zn
Zs
1.00
0.75
0.50
0.25
0.00
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
GC:0.697
sample size:1958
neutral:0.224
asymptote:0.161
init slope:0.123
tau:0.336
r:0.635
0.0
0.2
0.4
0.6
dnds

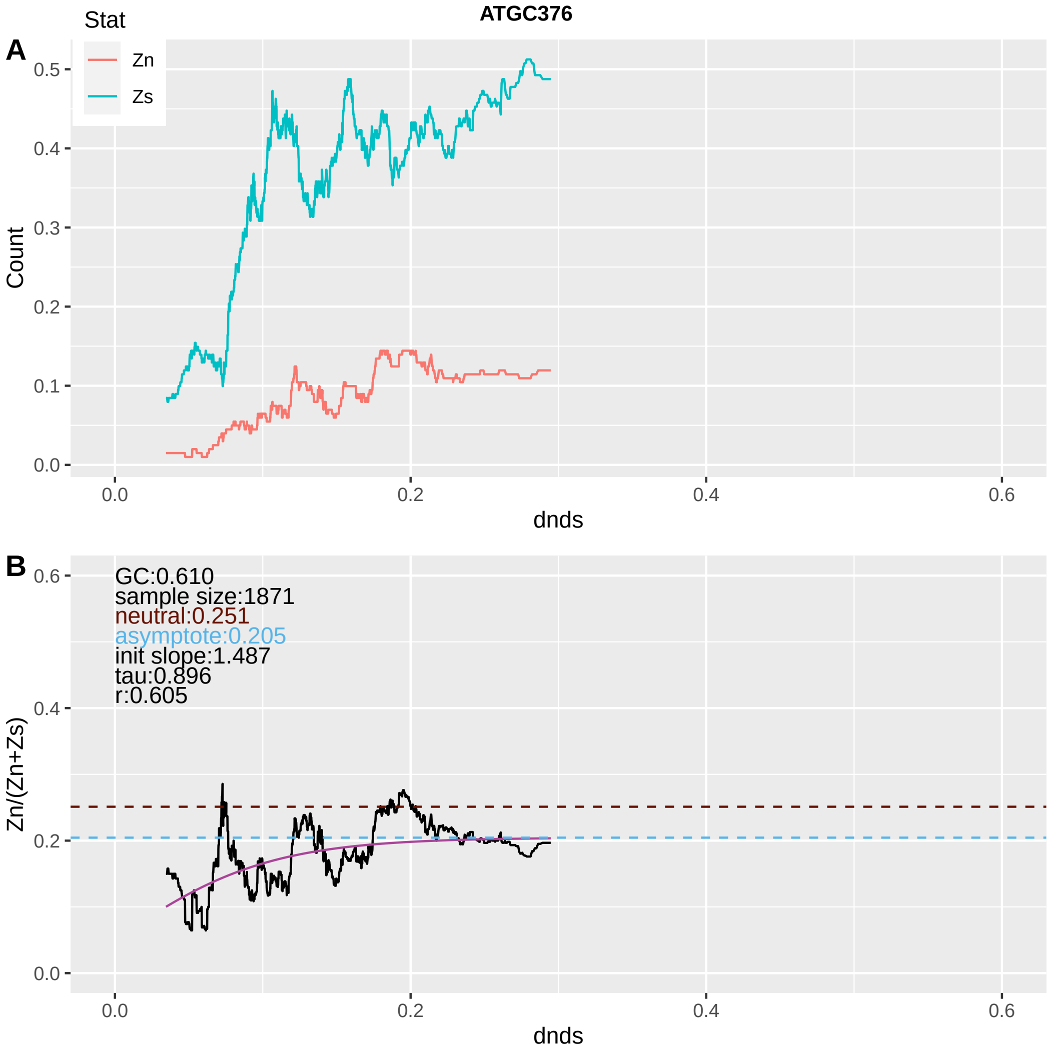

ATGC361
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.0
0.2
0.4
0.6
dnds
B
GC:0.548
sample size:2197
neutral:0.267
asymptote:0.150
init slope:0.232
tau:0.572
r:0.126
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
dnds

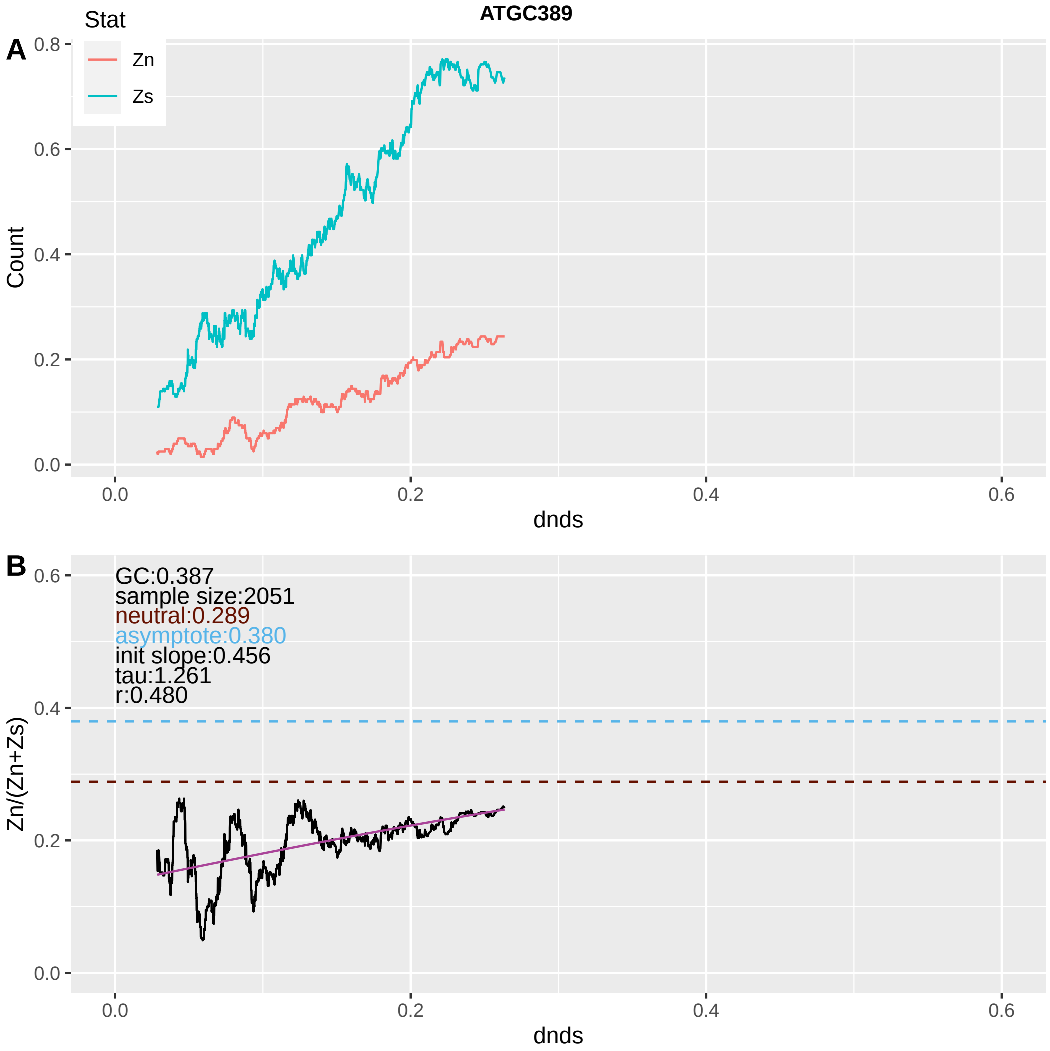

ATGC369
Stat
A
Zn
Zs
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.362
sample size:1550
neutral:0.291
asymptote:0.213
init slope:1.263
tau:0.823
r:0.495
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

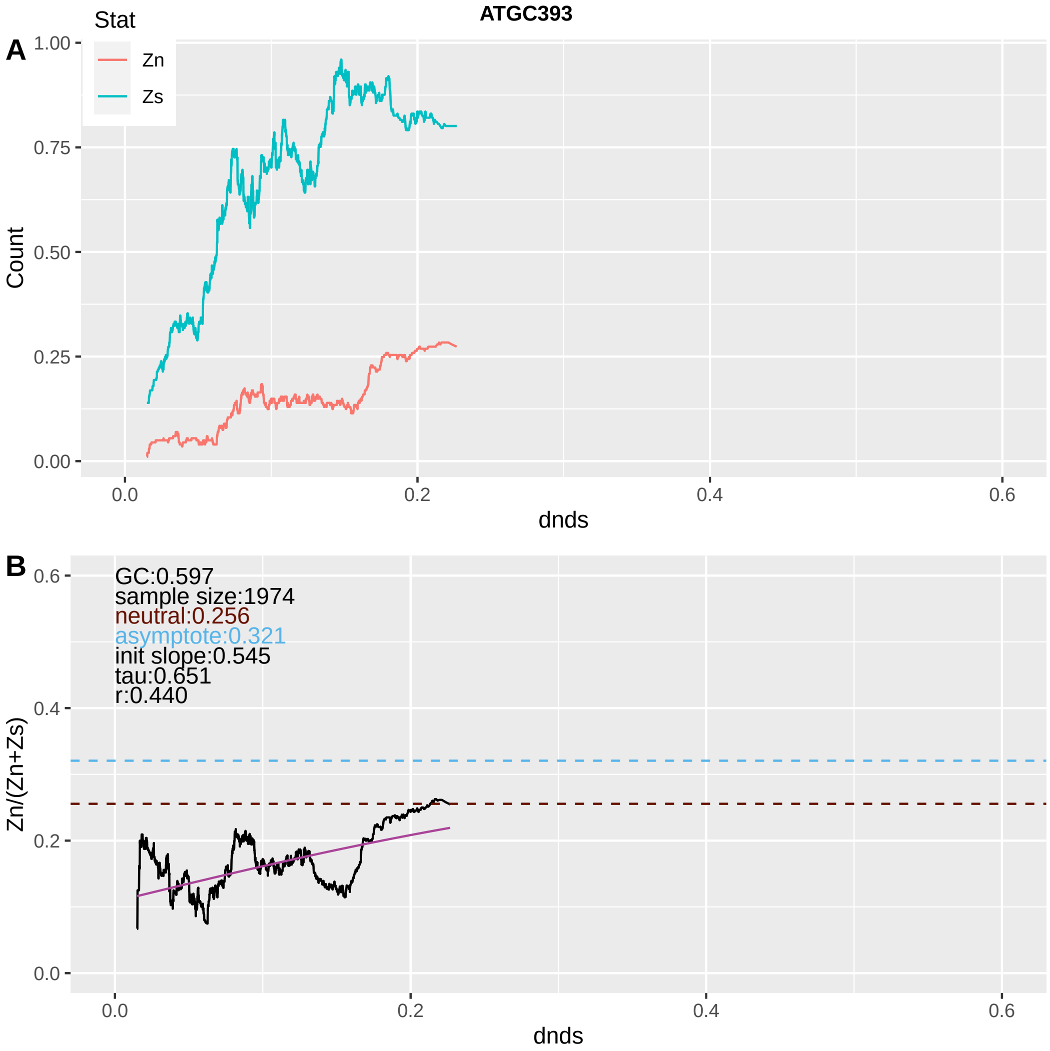

ATGC371
Stat
Zn
Zs
A
0.8
0.6
0.4
0.2
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
GC:0.404
sample size:1459
neutral:0.288
asymptote:0.161
init slope:−0.066
tau:1.360
r:−0.379
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds

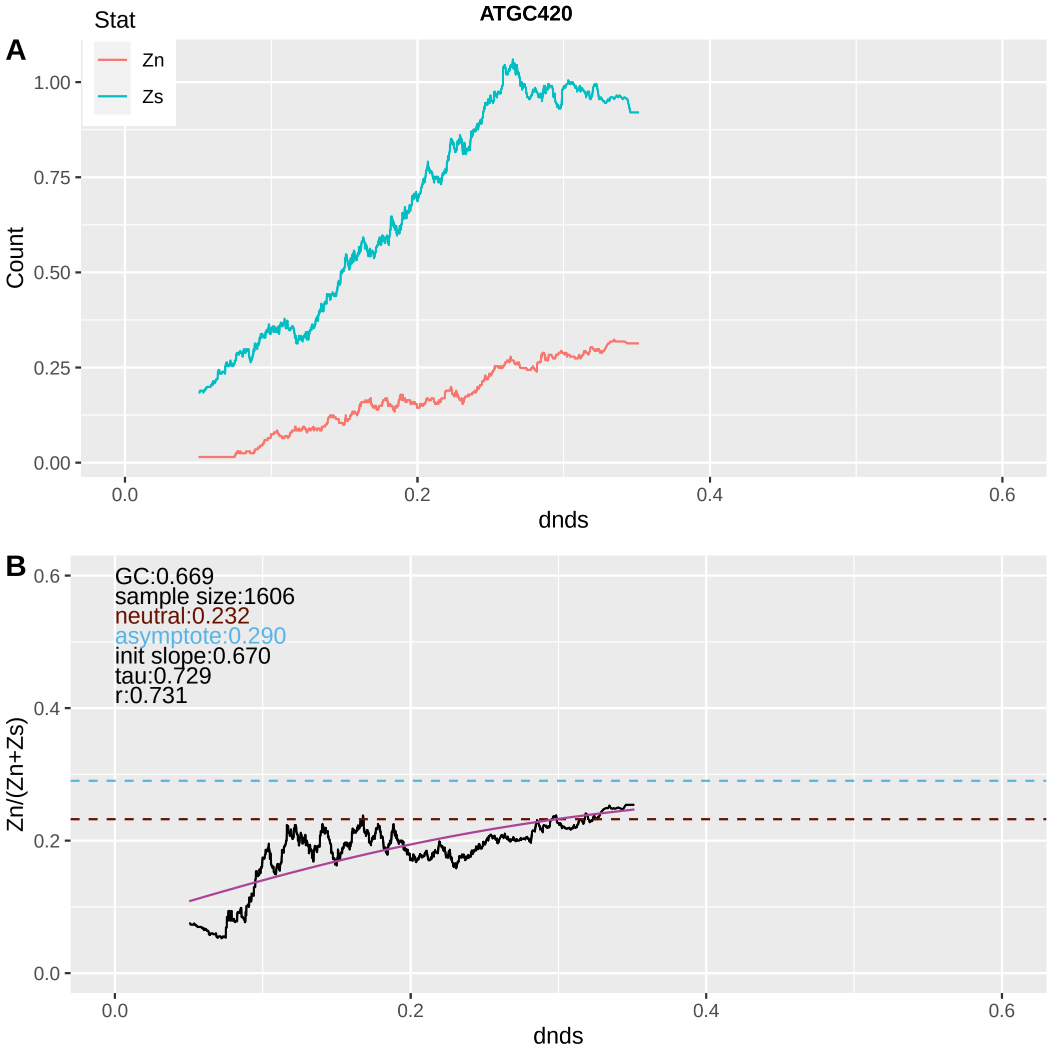

ATGC372
A
Stat
Zn
Zs
Count
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds
B
GC:0.510
sample size:1515
neutral:0.274
asymptote:0.195
init slope:0.416
tau:1.270
r:0.109
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

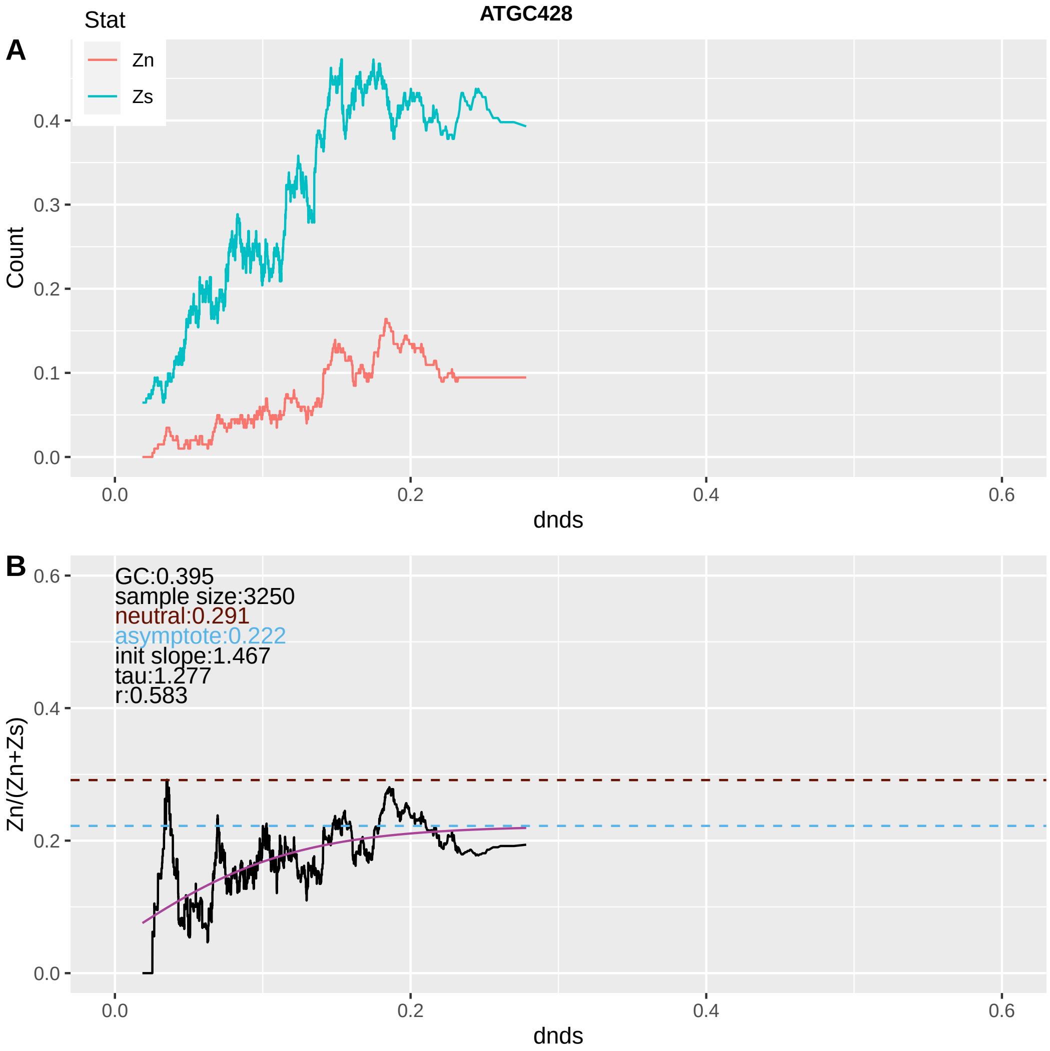

ATGC374
A
Stat
Zn
Zs
Count
0.0
0.2
0.4
0.6
0.8
dnds
0.0
0.2
0.4
0.6
B
GC:0.638
sample size:1933
neutral:0.240
asymptote:0.333
init slope:0.639
tau:1.099
r:0.687
Zn/(Zn+Zs)
0.0
0.2
0.4
0.6
dnds
0.0
0.2
0.4
0.6


ATGC376
A
Stat
Zn
Zs
Count
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.2
0.4
0.6
dnds
B
GC:0.610
sample size:1871
neutral:0.251
asymptote:0.205
init slope:1.487
tau:0.896
r:0.605
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds


ATGC389
A
Stat
Zn
Zs
Count
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds
B
GC:0.387
sample size:2051
neutral:0.289
asymptote:0.380
init slope:0.456
tau:1.261
r:0.480
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds


ATGC393
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.597
sample size:1974
neutral:0.256
asymptote:0.321
init slope:0.545
tau:0.651
r:0.440
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds


ATGC420
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.669
sample size:1606
neutral:0.232
asymptote:0.290
init slope:0.670
tau:0.729
r:0.731
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds


ATGC428
A
Stat
Zn
Zs
0.4
0.3
0.2
0.1
0.0
Count
0.0
0.2
0.4
0.6
dnds
B
0.6
0.4
0.2
0.0
Zn/(Zn+Zs)
GC:0.395
sample size:3250
neutral:0.291
asymptote:0.222
init slope:1.467
tau:1.277
r:0.583
0.0
0.2
0.4
0.6
dnds

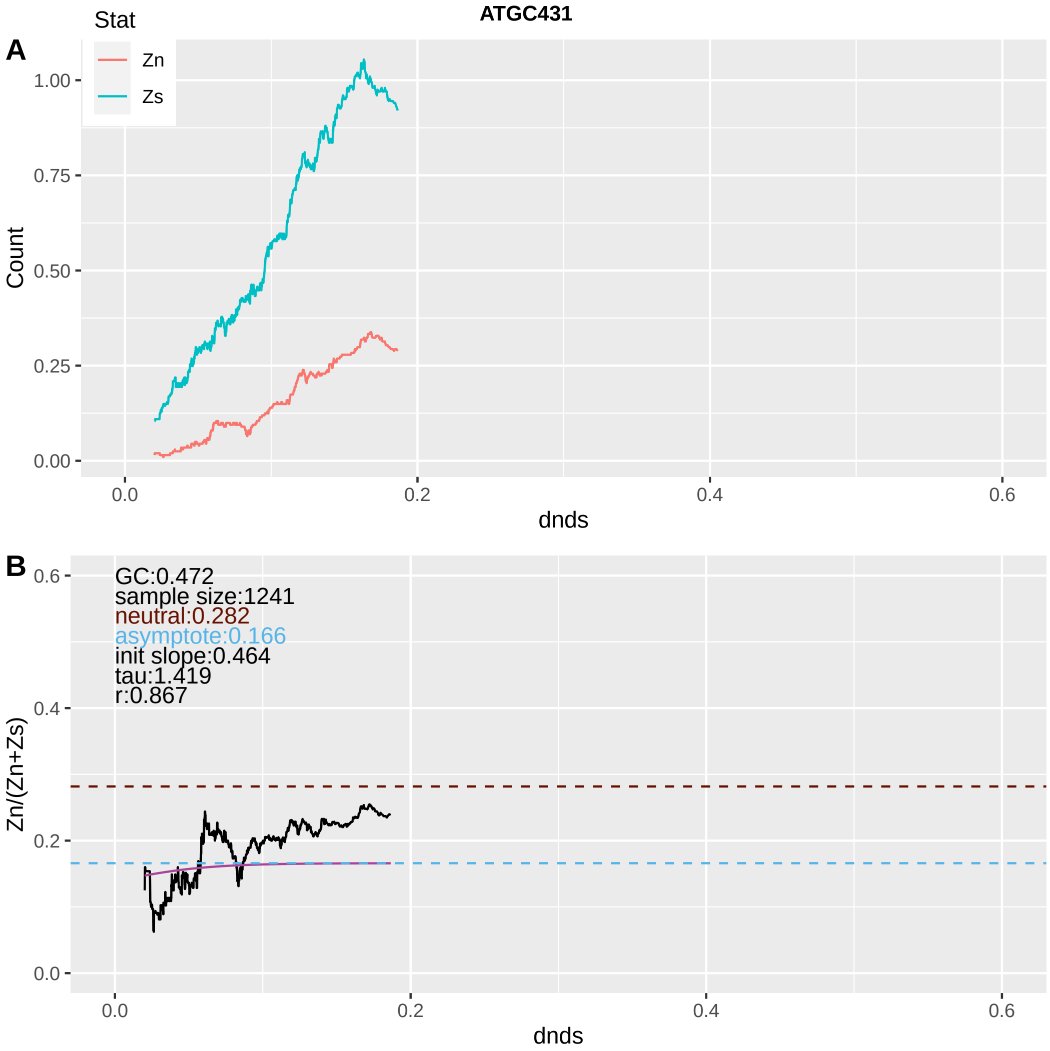

ATGC431
A
Stat
Zn
Zs
Count
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
dnds
B
GC:0.472
sample size:1241
neutral:0.282
asymptote:0.166
init slope:0.464
tau:1.419
r:0.867
Zn/(Zn+Zs)
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
dnds

Figure D.3: Correlation Plot between Dnds and Zn/(Zn+Zs) of 90 Species

APPENDIX E

CORE SCRIPTS FOR ALL CHAPTERS

cha2: Phylo-EM Training of the GTR+MG94 Model

```r
#Generate 100 simulation data from the Phylo-EM method
suppressWarnings(library(matlib))
suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(stringr))
suppressPackageStartupMessages(library(seqinr))
suppressPackageStartupMessages(library(R.utils))
suppressPackageStartupMessages(library(plyr))
suppressPackageStartupMessages(library(expm))
suppressPackageStartupMessages(library(purrr))
suppressPackageStartupMessages(library(matlib))
suppressPackageStartupMessages(library(jsonlite))
suppressPackageStartupMessages(library(SQUAREM))
#setwd("~/Dropbox (ASU)/Indel_project/Script")
#getwd()

#4*4 GTR matrix
GTR = function(Si, pai){
  r1 = matrix(0, 4, 4)
  r1[lower.tri(r1)] = Si
  r   = r1 + t(r1)
  r   = t(r*pai)
  diag(r) = -rowSums(r)
  return(r)
}

#Construct a 61*61 MG94 matrix
MG94 = function(g, oga, cod){
  R  = matrix(0, 61, 61)
  for (i in 1:61) {
    for (j in 1:61) {
      if(i == j){
        R[i, j] = 0
      }else if(sum(cod[i, ] != cod[j, ]) > 1){#more than 1 nucleotide changes
        R[i, j] = 0
      }else{
        if(codonstrs[j] %in% syn[[codonstrs[i]]]){#syn_subs
          w = 1
        }else{#nonsyn_subs
          w = oga
        }
        pos = which(cod[i, ] != cod[j, ])
        x   = which(DNA_BASES == cod[i, pos])
        y   = which(DNA_BASES == cod[j, pos])
        R[i, j] = w * g[x, y]
      }
    }
  }
  diag(R) = -rowSums(R)

  return(R)
}

#Log-likelihood
LL = function(theta){
  rmat  = GTR(theta[1:6], f0)
  Rmat  = MG94(rmat, theta[7], cod)
  llt = sum(log(expm(Rmat)*cf0)*dat1)
  return(llt)
}

#EM
em = function(p){
  s   = p[1:6]
  w   = p[7]
  r   = GTR(s, f0)
  R   = MG94(r, w, cod)
  #print(sum(R) %>% round(13))

  #Simulate branch length
  T1   = -sum(diag(r)*f0)
  #print(T1)

  ##make r, R symmetric
  fmat = outer(sqrt(cf0), 1/sqrt(cf0))
  S    = R * fmat

  ##calculate eigenvectors and values.
  eig = eigen(S)
  D   = eig$values
  V   = eig$vectors

  ##calculate Prob(b|a)
  pab = V %*% diag(exp(D)) %*% t(V)
  Pab = pab * t(fmat)
  #print(rowSums(Pab))


  ##Log likelihood "the smaller, the better"
  ll = sum(log(Pab*cf0)*dat1)
  cat(sprintf("%.6f\n", ll))



  ##construct the Jkl matrix
  J = outer(D/T1, D/T1, function(x,y) {
    ifelse(x-y == 0,
           T1*exp(x*T1),
           exp(y*T1)*(expm1((x-y)*T1))/(x-y))
  })

  ##calculate the expected values
  # W[a,b,i,i] is the expected time spent state i on a branch going from a -> b
  # U[a,b,i,j] is the expected number of events going from i -> j on a branch
   ↪  going from a->b
  W = array(0, c(61,61,61,61))
  U = array(0, c(61,61,61,61))

  tm = system.time(
    for(a in 1:61) {
      for(b in 1:61) {
        for(i in 1:61) {
          for(j in 1:61) {
            ff = sqrt(cf0[i]*cf0[b]/cf0[a]/cf0[j])
            o  = outer(V[a,]*V[i,], V[j,]*V[b,])   ##cheat: V[i,] = t(V)[,i]
            W[a,b,i,j] = ff * sum(o*J)
          }
        }
        W[a,b,,] = W[a,b,,] / Pab[a,b]
        U[a,b,,] = R * W[a,b,,]
      }
    }
  )

  ##calculate expected values by summing over observations --a,b is sumable.
  Wh = array(0, c(61,61))
  Uh = array(0, c(61,61))
  for(i in 1:61) {
    for(j in 1:61) {
      Wh[i,j] = sum(W[,,i,j] * dat1)
      Uh[i,j] = sum(U[,,i,j] * dat1)
    }
  }
  Wh = diag(Wh)

  ##M-Step maximize sigmas.
  sigma.Cij = sapply(seq(6), function(x){sum(Uh[sigma.id[[x]]])})
  sigma.Wij = c()
  for (k in 1:6) {
    ichunks   = ceiling(sigma.id[[k]]/61)
    sigma.Wij = c(sigma.Wij, sum(Wh[ichunks]* t(R)[sigma.id[[k]]])/s[k])
  }
  s = sigma.Cij/sigma.Wij


  ##M-Step maximize omega
  w.Cij = sum(sapply(omega.id, function(x){Uh[x[1], x[2]]}))
  Rii   = c()
  for (i in 1:61) {
    ith.non = omega.id[which(sapply(omega.id, "[[", 1) == i)]
    ith.sum = sum(sapply(ith.non, function(x){R[x[1], x[2]]})) / w
    Rii     = c(Rii, ith.sum)
  }
  w.Wij = sum(Wh * Rii)
  w     = w.Cij/w.Wij

  ##reconstruct gtr and mg94.
  pNew = c(s,w)
  #print(pNew)
  return(pNew)
}


##############################################################Part II Model
↪  building
main = function(Name, inFile){
  #Construct codons and its degeneracy
  stp = c(49,51,57)
  cod64 = cbind(rep(DNA_BASES, each = 16),
                rep(DNA_BASES, times = 4, each = 4),
                rep(DNA_BASES, 16))
  cod          = cod64[-stp,]
  codonstrs    = apply(cod, 1, stringr::str_c, collapse = "")
  syn          = syncodons(codonstrs)
  names(syn)   = toupper(names(syn))
  syn          = lapply(syn, toupper)

  #setup global
  cod <<- cod
  codonstrs <<- codonstrs
  syn <<- syn


  #indicators
  #Name    = "Results/est100/1.5e.json"
  #inFile = "../test_90_species/Results/truePar_100.txt"
  n     = as.numeric(str_extract(basename(Name), "[^.]+"))
  trueP = read.table(inFile, header=T, sep="")
  tP    = unlist(trueP[n,])
  pow   = as.numeric(gsub(".*[.]([^.]+)[e].*", "\\1", Name))

  #True parameters, unnormalized
  Pi    = tP[1:4]
  Sigma = tP[5:10]
  Tau   = tP[11]
  omega = tP[12]

  #Set up GTR matrix
  gtr = GTR(Sigma, Pi)
  #print(-sum(diag(gtr)*Pi))
  mg94 = MG94(gtr, omega, cod)

  #Set up mg94 matrix and normalize it.
  Pi2 = sapply(seq(61), function(x){prod(Pi[match(cod[x, ], DNA_BASES)])})
  Pi2 = Pi2/sum(Pi2)
  #print(sum(Pi2))

  #Create Symmetric matrix
  o   = outer(sqrt(Pi2), 1/sqrt(Pi2))
  s94 = mg94 * o
  #print(s94[1, ])

  p94 = expm(s94)* t(o)
  P94 = p94* Pi2
  #print(rowSums(p94))
  #print(sum(P94))

  #Normlize sigma as well
  #Sigma = Sigma/T

  #Build omega list for M-step.
  ##Locating all non-syn locations in 64*64 R matrix.
  omega.id = c()
  for (i in 1:61) {
    for (j in 1:61) {
      if((i != j) &&
         (sum(cod[i, ] != cod[j, ]) == 1) &&
         (!(codonstrs[j] %in% syn[[codonstrs[i]]])) ){
        omega.id = c(omega.id, i, j)
      }
    }
  }
  omega.id = split(omega.id, ceiling(seq_along(omega.id) / 2))

  #Build sigma list for M-step
  ##Locating all 6-sigma locations in 64*64 R matrix.
  ii = GTR(1:6, rep(1,4))
  diag(ii) = 0
  I = matrix(0, 61, 61)
  for (i in 1:61) {
    for (j in 1:61) {
      if(i == j){
        I[i, j] = 0
      }else if(sum(cod[i, ] != cod[j, ]) > 1){
        I[i, j] = 0
      }else{
        pos = which(cod[i, ] != cod[j, ])
        x   = which(DNA_BASES == cod[i, pos])
        y   = which(DNA_BASES == cod[j, pos])
        I[i, j] = ii[x, y]
      }
    }
  }
  sigma.id = sapply(1:6, function(x){which(I == x)})


  ############################################################Part III simulate
   ↪  data and run EM.
  ssize = 10^pow
  #Create sample data
  set.seed(8088)

  dat  = sample(61*61, ssize, replace=TRUE, prob = P94)
  dat  = table(dat)
  dat  = as.data.frame(dat)
  id1  = as.numeric(as.vector(dat[[1]]))
  id2  = as.numeric(as.vector(dat[[2]]))
  dat1 = matrix(0, 61, 61)
  for (i in 1:length(id1)) {
    dat1[id1[i]] = id2[i]
  }


  ##empirical f
  f1     = rowSums(dat1) + colSums(dat1)
  f1.id  = sapply(seq(61), function(x){match(cod[x, ], DNA_BASES)})
  base.f = c()
  for (i in seq(4)) {
    base.id = sapply(seq(61), function(x){length(which(f1.id[, x] == i))})
    base.f  = c(base.f, sum(base.id*f1))
  }
  f  = base.f/sum(base.f)
  #print(sum(f))
```

```r
##em to obtain the est of f0
f0=f
sum_61 = sum(f1)
for (i in 1:20) {
  Pi_stp   = c(f0[1]^2*f0[4],f0[1]*f0[3]*f0[4],f0[3]*f0[1]*f0[4])
  Pi_61    = 1 - sum(Pi_stp)
  sum_stp  = sum_61/Pi_61 - sum_61
  stp_norm = Pi_stp/sum(Pi_stp) *sum_stp
  cf0 = sapply(seq(61), function(x){prod(f0[match(cod[x, ], DNA_BASES)])})
  ll  = sum(f1*log(cf0))- sum_61*log(Pi_61)
  #cat(sprintf("%i: %.6f\n", i, ll))

  denom = 3*(sum_61+sum_stp)
  f0    = c(base.f[1]+2*stp_norm[1]+stp_norm[2]+stp_norm[3],
          base.f[2],
          base.f[3]+stp_norm[2]+stp_norm[3],
          base.f[4]+sum_stp)/denom
  #print(sum(f0))
}

#simu LL
r2      = GTR(Sigma, Pi)
R2      = MG94(r2, omega, cod)
ll.sim  = sum(log(expm(R2)*Pi2)*dat1)
#empirical LL
r1      = GTR(Sigma, f0)
cf0     = cf0/sum(cf0)
R1      = MG94(r1, omega, cod)
ll.emp  = sum(log(expm(R1)*cf0)*dat1)

if(ll.sim<ll.emp){
  print("Yes!")
}else{
  print("come on man!")
}

##init sigma
#create nuc transition matrix [4-fold-degeneracy-codons only]
fourD_id = which(sapply(syn, function(x){length(x) == 4}))
k=1
ndat = matrix(0,4,4)
while (k<length(fourD_id)) {
  i = j = fourD_id[k:(k+3)]
  ndat = ndat+dat1[i,j]
  k=k+4
}
fn = rowSums(ndat) + colSums(ndat)  #neutral freq
fn = fn/sum(fn)
#print(sum(fn))

#symmetric average of dat
sdat = matrix(0,4,4)
diag(sdat) = diag(ndat)
sdat = (ndat + t(ndat))/2
sf   = colSums(sdat)/sum(sdat)
#print(sum(sf))
Dhat  = diag(sf)
```

```
phat  = t(sdat/colSums(sdat))
#logm(phat)

eigP  = eigen(phat)
Athat = eigP$vectors %*% diag(log(eigP$values)) %*%  inv(eigP$vectors)
that  = -sum(diag(Athat*Dhat))
#print(that)
Ahat = t(Athat)/that
#print(Ahat)
s   = Ahat[lower.tri(t(Ahat))]

if(any(s<0)){
  cat(sprintf("neg-s0:  %i\n",n))
  s = runif(6)
}

#init omega
#obs non-syn change.
obs.non = sum(sapply(omega.id, function(x){dat1[x[1],x[2]]}))

#exp non-syn change
gtr2  = GTR(s, f0)
mg942 = MG94(gtr2, 1, cod)
P942  = expm(mg942)
#print(sum(P942))

exp.non = 0
for (i in 1:61) {
  ith.non = omega.id[which(sapply(omega.id,'[[',1)==i)]
  exp.non = exp.non + f1[i]*sum(sapply(ith.non, function(x){P942[x[1],x[2]]}))
}
w = obs.non/(exp.non/2)


#init para >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
#setup global
f0       <<- f0
cf0      <<- cf0
dat1     <<- dat1
omega.id <<- omega.id
sigma.id <<- sigma.id

p0   = c(s,w)
p0   = runif(7)
tm4b = system.time({
  ps4b = fpiter(par=p0, fixptfn=em, control=list(tol=1e-4, trace=TRUE,
   ↪  intermed=TRUE))
})
pd   = ps4b$p.intermed
llv = c()
tv   = c()
for (i in 1:nrow(pd)) {
  rmat    = GTR(pd[i,1:6],f0)
  tv[i]   = -sum(diag(rmat)*f0)
  llv[i]  = LL(pd[i,1:7])
}
```

```
  nuc.mat         = matrix(f0,nrow(pd),4,byrow=T)
  ps4b$p.intermed = cbind(pd,llv,tv,nuc.mat)
  ps4B            = toJSON(ps4b)

  #output
  write(ps4B,Name)
}

###########
args = commandArgs(trailingOnly=TRUE)
main(args[1],args[2])
```

cha2: Parameter Space Setup

```
library(expm)
library(stats)
library(Biostrings)

##################function setup
# Set up 4*4 GTR matrix
GTR = function(si, pai){
  r1 = matrix(0, 4, 4)
  r1[lower.tri(r1)] = si
  r  = r1 + t(r1)
  r  = t(r*pai)
  diag(r) = -rowSums(r)
  return(r)
}

# Construct a 64*64 MG94 matrix
MG94 = function(g, oga, cd, m){
  R  = matrix(0, m, m)
  for (i in 1:m) {
    for (j in 1:m) {
      if(i == j){
        R[i, j] = 0
      }else if(sum(cd[i, ] != cd[j, ]) > 1){#more than 1 nucleotide changes
        R[i, j] = 0
      }else{
        if(codonstrs[j] %in% syn[[codonstrs[i]]]){#syn_subs
          w = 1
        }else{#nonsyn_subs
          w = oga
        }
        pos = which(cd[i, ] != cd[j, ])
        x   = which(DNA_BASES == cd[i, pos])
        y   = which(DNA_BASES == cd[j, pos])
        R[i, j] = w * g[x, y]
      }
    }
  }
  diag(R) = -rowSums(R)

  return(R)
}
```

```r
set.seed(8088)
######################PART I: parameter space set up
##pi
##2(A+C)=1; highC-G or highA-T case.
nd  = 100
PiA = runif(nd,0.0,0.50)
PiT = PiA
PiC = 0.50 - PiA
PiG = PiC
Pi.all = list(PiA, PiC, PiG, PiT)
Pi.all = sapply(1:nd, function(x){sapply(Pi.all, "[[", x)})
print(dim(Pi.all))
print(colSums(Pi.all)) #1

##omega
##from paper
wv = runif(nd,0.02,0.5) #sim

##six sigmas
##set up the mean rate first; then b =1/a ; 1-sigma rule
##cv = 1; a = 1/(cv^2)

##distance Tau (total mutations load--only determined by neutral gtr matrix)
#Tau = runif(nd,0.05,5.5)  #(put 80% of the simulations in the area of saturated
↪  third-positions,seems too unreal!!)
#we setup the a*b=0.4
cv = 1
a  = 1/(cv^2)
b  = 0.4/a
Sigmas = matrix(0,6,100)
tv = c()
for (i in 1:100) {
  pai = Pi.all[,i]
  si  = rgamma(6, shape=a, scale=b)
  gtr = GTR(si, pai)
  Sigmas[,i] = si
  tv[i]  = -sum(diag(gtr)*pai)
}

mean(tv)
hist(tv, prob = TRUE, xlim = c(0,2))
stv = sort(tv)
lines(stv,dgamma(stv,shape=a,scale=b),col='magenta',lwd=3)
plot(density(tv), xlab='tau', ylab='density')


##normalize sigma.
nSigmas = sapply(1:100, function(x){Sigmas[,x]/tv[x]})
print(nSigmas)

##output the 100 parameters
tPar = matrix(0,100,12)
for (i in 1:nd) {
  tPar[i,1:4]  = Pi.all[,i]
  tPar[i,5:10] = Sigmas[,i]
  tPar[i,11]    = tv[i]
  tPar[i,12]    = wv[i]
}
colnames(tPar)=c('A','C','G','T','s1','s2','s3','s4','s5','s6','tau','omega')
write.table(tPar,"Results/truePar_100.txt",quote=F,sep="\t",
            row.names=F)
```

cha2: Nelder-Mead Algorithm for Derivative-free Optimization

```
suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(stringr))
suppressPackageStartupMessages(library(seqinr))
suppressPackageStartupMessages(library(R.utils))
suppressPackageStartupMessages(library(plyr))
suppressPackageStartupMessages(library(expm))
suppressPackageStartupMessages(library(purrr))
suppressPackageStartupMessages(library(matlib))
suppressPackageStartupMessages(library(jsonlite))

#rewrite the tolrence criterion from the nmkb package as as rms then apply the
↪  nmkb method

# library(devtools)
# library(roxygen2)
# setwd("~/Dropbox (ASU)/Indel_project/Script/90")
# getwd()
# devtools::create("nmkb")

#setwd("~/Dropbox (ASU)/Indel_project/Script")
#document
#devtools::document(pkg="./nmkb")
#load_package
#devtools::load_all(path="~/Dropbox (ASU)/Indel_project/Script/90/nmkb")
getwd()
devtools::load_all(path="../Script/90/nmkb")

#sources all functions needed.
sub = "~/Dropbox (ASU)/Indel_project/Script/sources/"
source(paste0(sub,"codon_call.R"))
source(paste0(sub,"gtr.R"))
source(paste0(sub,"mg94.R"))
source(paste0(sub,"omega_coor.R"))
source(paste0(sub,"sigma_coor.R"))
source(paste0(sub,"pseudo_data.R"))
source(paste0(sub,"init_f.R"))
source(paste0(sub,"LL.R"))
source(paste0(sub,"init_sigma.R"))
source(paste0(sub,"init_omega.R"))
source(paste0(sub,"phylo_em.R"))


# -log_likelihood
LL_min  = function(theta){
  val = LL(cod,codonstrs,syn,theta,f0,cf0,dat,num)
  return(-val)
}
```

```
##########################################
main = function(Name, inFile){
  #example
  #Name   = "Results/nmkb100.1/17.5e.json"
  #inFile = "../../test_90_species/Results/truePar_100.txt"
  n     = as.numeric(str_extract(basename(Name), "[^.]+"))
  trueP = read.table(inFile, header=T, sep="")
  tP    = unlist(trueP[n,])

  #True parameters, unnormalized
  Pi    = tP[1:4]
  Sigma = tP[5:10]
  Tau   = tP[11]
  omega = tP[12]

  #61 or 64?
  num_dim = '61'
  num = as.numeric(num_dim)
  print(num)

  #>codon strs
  cdc = codon_call(num)
  cod = cdc[[1]]
  codonstrs = cdc[[2]]
  syn = cdc[[3]]

  #>
  gtr = GTR(Sigma,Pi)

  #>
  mg94 = MG94(gtr,omega,cod,codonstrs,syn,num)

  #>
  omega.id = omega_coor(cod, codonstrs, syn, num)
  sigma.id = sigma_coor(cod, num)

  ##generate pmat
  Pi2 = sapply(seq(num), function(x){prod(Pi[match(cod[x, ], DNA_BASES)])})
  Pi2 = Pi2/sum(Pi2)

  o   = outer(sqrt(Pi2),1/sqrt(Pi2))
  s94 = mg94*o    #symmetric matrix
  p94 = expm(s94)*t(o)
  P94 = p94*Pi2
  print(sum(P94))


  #>>>>>>>>>>>>>>>>>>>simualte data
  set.seed(8088)
  dat = pseudo_data(1e+5,P94,num)

  #>
  nuc_codon_freq = init_f(cod,dat,num)
  f0  = nuc_codon_freq[[1]]
  cf0 = nuc_codon_freq[[2]]
  print(sum(f0))
  print(sum(cf0))

  #test if sim.LL<emp.LL
  ll.sim = sum(log(expm(mg94)*Pi2)*dat)
  ll.emp = LL(cod,codonstrs,syn,c(Sigma,omega),f0,cf0,dat,num)
  if(ll.sim<ll.emp){
    print("Yes!")
  }else{
    print("come on man!")
  }

  #>
  s = init_sigma(syn,dat)
  w = init_omega(cod,codonstrs,syn,dat,f0,s,omega.id,num)

  #running SQUAREM
  num      <<- num
  cod      <<- cod
  codonstrs<<- codonstrs
  syn      <<- syn
  f0       <<- f0
  cf0      <<- cf0
  dat      <<- dat
  omega.id <<- omega.id
  sigma.id <<- sigma.id


  p0   = c(s,w)
  if(any(p0>1)){
    len = length(which(p0>1))
    p0[which(p0>1)] = runif(len)
  }
  tm = system.time({
    pb = nmkb::nmkb(fn=LL_min, par=p0, lower=0.0, upper=2.5,
     ↪  control=list(tol=1e-4,trace=TRUE))
  })

  rmat   = GTR(pb$par[1:6],f0)
  tv     = -sum(diag(rmat)*f0)
  pb$par = c(pb$par,tv,f0)
  pj     = toJSON(pb)

  write(pj,Name)
}

#########################
args = commandArgs(trailingOnly=T)
main(args[1],args[2])
```

cha3: Homology Call

```
#extract the gene ID from ensembl API

# rm(list=ls())
# invisible(dev.off())
######################
# .libPaths()
# [1] "/home/zzhu49/R/x86_64-pc-linux-gnu-library/3.4"
# [2] "/usr/local/lib/R/site-library"
# [3] "/usr/lib/R/site-library"
# [4] "/usr/lib/R/library"

library(BiocGenerics)
library(S4Vectors)
library(Biostrings)
library(biomaRt)
library(readr)
library(dplyr)
# setwd("Dropbox (ASU)/Indel_project/Script")

# human = useDataset("hsapiens_gene_ensembl", mart = ensembl)
# attributes = c("ensembl_gene_id",
#                 "mmusculus_homolog_ensembl_gene",
#                 "mmusculus_homolog_orthology_type" ,
#                 "rnorvegicus_homolog_ensembl_gene",
#                 "rnorvegicus_homolog_orthology_type"
# )
# orth_filtered = orth %>% filter(`mmusculus_homolog_orthology_type` ==
 ↪  "ortholog_one2one" &
#                                   `rnorvegicus_homolog_orthology_type` ==
 ↪  "ortholog_one2one" )


# input  = "../Input/homo3.1.txt"
# output = "../test_human_mouse_rat/Raw_data/geneId.txt"

main =function(input, output){
  #Read input
  name = read_delim(input, "\t", col_names = FALSE)

  #Extract
  ensembl = useMart("ensembl")
  data    = useDataset(paste0(toString(name[1, 1]), "_gene_ensembl"), mart =
   ↪  ensembl)l

  # b = listAttributes(human)
  # b£name
  # a = listFilters(human)
  # a£name

  attributes = c("ensembl_gene_id")
  for (i in 2:nrow(name)) {
    att1 = paste0(toString(name[i, 1]), "_homolog_ensembl_gene")
    att2 = paste0(toString(name[i, 1]), "_homolog_orthology_type")
    attributes = c(attributes ,att1, att2)
  }

  filters = "with_ccds"
  orth    = getBM(attributes, filters, values = TRUE,
                  mart = data, uniqueRows = TRUE)
  # Element-wise comparison with &.
  orth_filtered = orth[(orth[3] == "ortholog_one2one" & orth[5] ==
   ↪  "ortholog_one2one"), ]
  orth_com      = orth_filtered[, c(1, 2, 4)]

  # Write data
  write.table(orth_com, output, sep = '\t', row.names = F, col.names = F, quote =
   ↪  FALSE)

}

args = commandArgs(trailingOnly = TRUE)
main(args[1], args[2])
```

```
#Extract the CDS from ensembl API.

#setwd("~/Dropbox (ASU)/Indel_project/Script")
library(methods)
library(httr)
library(jsonlite)
library(xml2)
library(readr)
library(Biostrings)
library(stringr)


# file  = "../test_human_mouse_rat/Raw_data_2/geneId.txt"
# n     = "ENSG00000128815"
# output= "../test_human_mouse_rat/Raw_data/cds_seq/"

########################################3
main = function(n, file, output){

  server = "http://rest.ensembl.org"
  geneId = read_delim(file, "\t", col_names = FALSE)

  n          = str_extract(basename(n), "[^.]+")
  gene.stem  = geneId[which(geneId[[1]] == n), ]

  cds.list = list()
  for (i in 1:length(gene.stem)) {
      name = gene.stem[[i]]
      ext  = paste("/lookup/id/", name, "?expand=1",sep = "")
      r    = GET(paste(server, ext, sep = ""), content_type("application/json"),
       ↪  timeout(30))
      stop_for_status(r)
      id   = fromJSON(toJSON(content(r)))
      id_  = id$Transcript[["id"]][id$Transcript$is_canonical==1]

      #quality control
      if(is.null(id_)){
        quit()
      }

      ext_1 = paste("/sequence/id/",id_,"?type=cds",sep = "")
      r_1   = GET(paste(server, ext_1, sep = ""), content_type("text/x-fasta"))
      stop_for_status(r_1)
      cds_  = (content(r_1))

      #quality control
      if(is.null(cds_)){
        quit()
      }

      cds.list = c(cds.list, cds_)
  }

  write.table(cds.list, paste0(output, n, ".fa"), quote = FALSE,
              row.names = FALSE, col.names = FALSE, append = FALSE, sep = "\n")

}
```

```r
args = commandArgs(trailingOnly = TRUE)
main(args[1], args[2], args[3])
```

cha3: Sliding Window Method

```r
# Update the gaps
suppressWarnings(suppressMessages(library(tidyverse)))
suppressWarnings(suppressMessages(library(dplyr)))
suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(BiocGenerics))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(S4Vectors))
suppressPackageStartupMessages(library(seqinr))
suppressPackageStartupMessages(library(stringi))

#setwd("~/Dropbox (ASU)/Indel_project/test_human_mouse_rat/Data_6")

# Initial word size (Window) as 27(6+3+6).
# convert ---AAA/AAA--- to ===AAA/AAA===
# convert ---/---       to ===/===
# convert ---AAA---     to ===AAA===

#Update any large gaps
filter_Long = function(x, X){
  if(length(x) == 0){
    return(X)
  }else{
    st  = start(x)
    en  = end(x)
    wid = width(x)
    for(i in 1:length(x)){
      subseq(X, start = st[i], end = en[i]) = stri_rand_strings(1, wid[i], '[=]')
    }
  }
  return(X)
}

#Clean up of start/end position of the seq
start_stop_test = function(X){
  Start = substr(X, 1, Window)
  IsGap.1 = grepl('-', Start)

  Stop = substr(X, (nchar(X) - (Window - 1)), nchar(X))
  IsGap.2 = grepl('-', Stop)

  if(IsGap.1 == TRUE){
    Start = gsub('-', '=', Start)
    substr(X, 1, Window) = Start
  }

  if(IsGap.2 == TRUE){
    Stop = gsub('-', '=', Stop)
    substr(X, (nchar(X) - (Window - 1)), nchar(X)) = Stop
  }
  return(X)
}

#Adjust gap distance
ad_gap_dis = function(y, seq, ref){

  if(length(y) == 0){return(c(seq, ref))}

  pos.all = start(y)
  wid.all = width(y)
  pos.en  = end(y)

  for(i in 1:length(pos.all)){
    pos = pos.all[i]
    wid = wid.all[i]
    en  = pos.en[i]
    left   = substr(seq, start = pos - Wall - 3, stop = pos - 1)
     ↪    ## extra three is to make sure each gap is independent.
    right  = substr(seq, start = pos + wid, stop = pos + wid + (Wall + 2))
    window = substr(ref, start = pos - Wall - 3, stop = pos + wid + (Wall + 2))

    l1  = grep("[-=]", left)
    r1  = grep("[-=]", right)
    Wid = grep("[-=]", window)

    if(length(l1) != 0 | length(r1) != 0 | length(Wid) != 0){
      subseq(seq, start = pos, end = en) = stri_rand_strings(1, wid, '[=]')
    }
  }
  res = c(seq, ref)
  return(res)
}


# seq1 = "AAT---AAACAAAGAATGCTTACTGT---ATAAGGCTTACTGTTCTAGCG---ATCACCGCG---TCATGT
 ↪   CTAGTTATGAACGGC------GGTTTAACATTGAATAGCAAGGCACTTCCATAATAGGGCCGTC---GTAATTGTCT
 ↪   AATATAG------ATAGTA---"
# seq2 = "TAA------AA---AATTTGATGCTACATTGGATGAGTCTACTTCGAGCGCGCCGCATCGATTGCAAGAGC
 ↪   AGTGTTGCCT---AAGAGCCGTTAGATGCGTCGTTG---ATCGCGTCCGATAATTCGGGAGTTG---CCCAATATTT
 ↪   AATATGATGA---TAGCTATAA"

# inFile = "../Raw_data/pair_mafft/ENSG00000000460.fa"
# oudir  = "Mafft/updated_cds/"
# num1   = '6'; num2 = '12'

main = function(inFile,ouDir,num1,num2){

  dna = readDNAStringSet(inFile)
  name = names(dna)

  #Set up vars
  num1 = as.numeric(num1)
  num2 = as.numeric(num2)
  Window  <<- num1
  Wall    <<- num2
  #String mode
  spec.1   = toString(dna[[1]])
  spec.2   = toString(dna[[2]])

  #Find gap range
  dna.1 = str_split(as.character(dna), '')
  g     = lapply(dna.1, function(x) { IRanges(x == '-')})
  g     = IRangesList(g)
  m     = g[[1]]
  r     = g[[2]]

  wid.m = width(m)
  wid.r = width(r)

  #Test gap range
  gap = c(3,6,9,12)
  m.null = m[!(wid.m %in% gap)]
  r.null = r[!(wid.r %in% gap)]

  ########################################PART II
  M = filter_Long(m.null,spec.1)
  R = filter_Long(r.null,spec.2)

  #Clean up of start-end region of the seq
  M.1 = start_stop_test(M)
  R.1 = start_stop_test(R)


  #Update old gap length
  M.2 = str_split(as.character(M.1),'')
  R.2 = str_split(as.character(R.1),'')

  g.m = lapply(M.2, function(x) { IRanges(x == '-')})
  g.r = lapply(R.2, function(x) { IRanges(x == '-')})

  g.m = IRangesList(g.m)[[1]]
  g.r = IRangesList(g.r)[[1]]

  #Update the gap via swapping reference
  m.all = g.m[width(g.m)==3 | width(g.m)==6 | width(g.m)==9 | width(g.m)==12 ]
  r.all = g.r[width(g.r)==3 | width(g.r)==6 | width(g.r)==9 | width(g.r)==12 ]

  M_R = ad_gap_dis(m.all, M.1, R.1)
  R_M = ad_gap_dis(r.all, M_R[2], M_R[1])

  #Update the focal-seq
  new.M    = R_M[2]
  new.R    = R_M[1]
  new_seq = list(new.M,new.R)

  write.fasta(sequences=new_seq, names=name, nbchar=80,
              open="w", as.string=TRUE, file.out=paste0(ouDir,basename(inFile)))

}


###############################################
```

```
#compatible with run_test.R
args = commandArgs(trailingOnly=T)
if(!interactive()){
 main(args[1], args[2], args[3], args[4])
}
```

```r
# Slide the gaps and find the best hit.
suppressWarnings(suppressMessages(library(tidyverse)))
suppressWarnings(suppressMessages(library(dplyr)))
suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(BiocGenerics))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(S4Vectors))
suppressPackageStartupMessages(library(seqinr))
suppressPackageStartupMessages(library(stringi))
suppressPackageStartupMessages(library(stringr))

#setwd("~/Dropbox (ASU)/Indel_project/test_human_mouse_rat/Data_6")

#Hamming distance
Align = function(u, v){
  u = str_split(u, "")
  v = str_split(v, "")
  u = u[[1]]
  v = v[[1]]

  id_pos    = length(which(u[u == v] != '-'))        # Cal the identical number
   ↪  of identical bases excluding gaps.
  align_pos = Wall * 2

  sim = 100 * (id_pos) / (align_pos)
  return (sim)
}

#left_sliding
left_slide = function(seq_v, index, wid){
  swap(seq_v[index - 1], seq_v[index + wid - 1])
  seq    = paste(seq_v, collapse = "")
  return (seq)
}

#right_sliding
right_slide = function(seq_v, index, wid){
  swap(seq_v[index], seq_v[index + wid])
  seq    = paste(seq_v, collapse = "")
  return (seq)
}

#Convert string to vector
str_convert = function(s){
  s1 = str_split(s, "")
  s1 = s1[[1]]
  return(s1)
}

#Generate the pseudo seq with an optimal alignment
Merge = function(idx, wid, x, y){
  best_aligned = x

  # Original window seq
  start = idx - Wall
  stop  = idx + Wall + wid - 1
  wid.1   = substr(x, start, stop)
  wid_ref = substr(y, start, stop)

  #Original alignment score
  sim_ref_public = Align(wid_ref, wid.1)

  #left_aligning
  sim_total.l   = c()
  all_aligned.l = c()
  new_aligned.l = x
  idx.l         = idx
  for(k in 1:Window){
    new_aligned.l = left_slide(str_convert(new_aligned.l), idx.l, wid)
    wid.L         = substr(new_aligned.l, start, stop)

    sim_total.l[k]   = Align(wid_ref, wid.L)
    all_aligned.l[k] = new_aligned.l
    idx.l            = idx.l - 1
  }

  #right_aligning
  sim_total.r   = c()
  all_aligned.r = c()
  new_aligned.r = x
  idx.r         = idx
  for(k in 1:Window){
    new_aligned.r = right_slide(str_convert(new_aligned.r), idx.r, wid)
    wid.R         = substr(new_aligned.r, start, stop)

    sim_total.r[k]   = Align(wid_ref, wid.R)
    all_aligned.r[k] = new_aligned.r
    idx.r            = idx.r + 1
  }

  #Filtering multiple optimal allignment ---/+++
  sim_all   = c(sim_ref_public, sim_total.l, sim_total.r)
  alg_all   = c(x, all_aligned.l, all_aligned.r)
  index.sim = which(sim_all == max(sim_all))
  if(length(index.sim) > 1){
    subseq(best_aligned,start = idx, end = idx + wid - 1) = stri_rand_strings(1,
     ↪  wid, '[+]')
  }else{
    best_aligned = alg_all[index.sim]
  }

  return (best_aligned)
}


# inFile = "../test_human_mouse_rat/Data_6/Mafft/updated_cds/ENSG00000000460.fa"
# oudir  = "../test_human_mouse_rat/Data_6/Data/Mafft/mapped_cds/"

# seq1 = "AAT===AAACAAAGAATGCTTACTGT---ATAAGGCTTACTGTTCTAGCG===ATCACCGCG===TCATGT⌋
 ↪  CTAGTTATGAACGGC------GGTTTAACATTGAATAGCAAGGCACTTCCA---TAATAGGGCCGTC===GTAATTG⌋
 ↪  TCTAATATAG------ATAGTA==="
# seq2 = "TAA------AA===AATTTGATGCTACATTGGATGAGTCTACTTCGAGCGCGCCGCATCGATTGCAAGAGC⌋
 ↪  AGTGTTGCCT===AAGAGCCGTTAGATGCGTCGTTG---ATCGCGTCCGATAATTCGGGAGTTGTGC===CCCAATA⌋
 ↪  TTTAATATGATGA===TAGCTATAA"
```

```
#################################################
main = function(inFile,ouDir,num1,num2){
  dna = readBStringSet(inFile)
  name = names(dna)

  #Set up var (global var)
  num1 = as.numeric(num1)
  num2 = as.numeric(num2)
  Window  <<- num1
  Wall    <<- num2

  #String mode
  spec.1   = toString(dna[[1]])
  spec.2   = toString(dna[[2]])

  #Capture all gaps of different lengths(3,6,9,12)
  dna.1 = str_split(as.character(dna), '')
  g     = lapply(dna.1, function(x) { IRanges(x == '-')})
  g     = IRangesList(g)
  m     = g[[1]]
  r     = g[[2]]

  wid.m = width(m)
  wid.r = width(r)
  l.m   = length(wid.m)
  l.r   = length(wid.r)
  pos.m = start(m)
  pos.r = start(r)


  if((l.m > 0) & (l.r > 0)){
    for(i in 1:l.m){
      spec.1 = Merge(pos.m[i], wid.m[i], spec.1, spec.2)
    }
    for(i in 1:l.r){
      spec.2 = Merge(pos.r[i], wid.r[i], spec.2, spec.1)
    }
  }else if((l.m > 0) & (l.r == 0)){
    for(i in 1:l.m){
      spec.1 = Merge(pos.m[i], wid.m[i], spec.1, spec.2)
    }
  }else if((l.m == 0) & (l.r > 0)){
    for(i in 1:l.r){
      spec.2 = Merge(pos.r[i], wid.r[i], spec.2, spec.1)
    }
  }else{
    quit()
  }

  #Filter out the seq with '+++' only.
  flag = unlist(lapply(c(spec.1, spec.2), function(x){grepl('-', x)}))
  if(any(flag)==TRUE){
    new_seq = list(spec.1, spec.2)
    write.fasta(sequences=new_seq, names=name, nbchar=80,
                open="w", as.string = TRUE, file.out=paste0(ouDir,
                 ↪  basename(inFile)) )
  }else{
    quit()
  }
}

########################################
#compatible with run_test.R
args = commandArgs(trailingOnly=TRUE)
 if(interactive()){
  main(args[1], args[2], args[3], args[4])
 }
```

cha3: Phase Proportion Analysis

```
# Calculate the proportion of phase 0 / phase 1 / phase 2 indels
suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(stringr))

#setwd("~/Dropbox (ASU)/Indel_project/test_human_mouse_rat/Data_6")

#Generate dataframe: pos | width
phasing = function(file){
  dna = readBStringSet(file, format = "fasta")
  l   = length(dna)

  dna.1 = str_split(as.character(dna), '')
  g = lapply(dna.1, function(x) { IRanges(x == '-')})
  g = IRangesList(g)

  m = g[[l-1]]
  r = g[[l]]

  wid.m = width(m)
  wid.r = width(r)

  l.m = length(wid.m)
  l.r = length(wid.r)

  pos.m = start(m)
  pos.r = start(r)


  df.m = data.frame("pos" = pos.m, "wid" = wid.m)
  df.r = data.frame("pos" = pos.r, "wid" = wid.r)

  if(l.m > 0 & l.r == 0){
    return(df.m)
  }else if(l.m == 0 & l.r > 0){
    return(df.r)
  }else if(l.m > 0 & l.r > 0){
    df = merge(df.m, df.r, all = TRUE)
    return(df)
  }else{
    return(NULL)
  }

}

#Distinguish phases
Record = function(PST){
  if(is.null(PST)){
    return(PST)
  }
  rem    = PST %% 3
  phase.0 = length(which(rem == 1))
  phase.1 = length(which(rem == 2))
  phase.2 = length(which(rem == 0))

  df.phase = data.frame(Phase_0 = phase.0, Phase_1 = phase.1, Phase_2 = phase.2)
  return(df.phase)
}

#Generate the phase dataframe
phase_gen = function(inD){
  File.total = list.files(inD, full.names=TRUE)
  PST = c()
  for(i in 1:length(File.total)){
    phase_score = phasing(File.total[i])
    PST         = rbind(PST, phase_score)
    print(i)
  }

  pos.3  = PST[PST$wid == 3, ]$pos
  pos.6  = PST[PST$wid == 6, ]$pos
  pos.9  = PST[PST$wid == 9, ]$pos
  pos.12 = PST[PST$wid == 12,]$pos

  phase.3  = Record(pos.3)
  phase.6  = Record(pos.6)
  phase.9  = Record(pos.9)
  phase.12 = Record(pos.12)

  Phase.df  = rbind(phase.3, phase.6, phase.9, phase.12)
  Phase.df
}

#ggplot of phase prop
gg_phase = function(dat,win){
  df      = as.table(noquote(t(dat)))
  colnames(df) = c('3','6','9','12')
  safe_colorblind_palette = c("#0072B2","#009E73","#AA4499")

  barplot(df, main=paste0("Window size of ",win), xlab="Gap length", ylab="Count",
          col=safe_colorblind_palette)
  legend("topright", bg="transparent", bty="n", rownames(df),
   ↪  fill=safe_colorblind_palette, cex=1)
}

################################################
#ouFig  = "Figure/Phase.mafft.pdf"
```

```
main = function(ouFig){

  dir1  = "../Data_3/Mafft/mapped_cds"
  dir2  = "Mafft/mapped_cds"
  dir3  = "../Data_9/Mafft/mapped_cds"
  dir4  = "../Data_12/Mafft/mapped_cds"

  pha1 = phase_gen(dir1)
  pha2 = phase_gen(dir2)
  pha3 = phase_gen(dir3)
  pha4 = phase_gen(dir4)


  #ggplot
  pdf(ouFig,onefile=T)
  par(mfrow=c(2,2))
  gg_phase(pha1,3)
  gg_phase(pha2,6)
  gg_phase(pha3,9)
  gg_phase(pha4,12)

  dev.off()

}

#######################################
args = commandArgs(trailingOnly=TRUE)
main(args[1])
```

```
# Calculate the Positional difference distribution of indels.

suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(BiocGenerics))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(S4Vectors))
suppressPackageStartupMessages(library(stringr))
suppressPackageStartupMessages(library(ggplot2))
suppressPackageStartupMessages(library(ggpubr))

#setwd("~/Dropbox (ASU)/Indel_project/test_human_mouse_rat/Data_6")

#return pos | wid | flag
phasing = function(file){
  dna = readBStringSet(file, format = "fasta")
  l   = length(dna)

  dna.1 = str_split(as.character(dna), '')
  g1 = lapply(dna.1, function(x) { IRanges(x %in% c("-", "+"))})
  g2 = lapply(dna.1, function(x) { IRanges(x == "+")})
  g1 = IRangesList(g1)
  g2 = IRangesList(g2)

  mr     = g1[(l-1):l]
  wid.mr = unlist(width(mr))
  pos.mr = unlist(start(mr))

  # set up flag as "+++"
  flag  = rep(0, length(wid.mr))
  flag1 = unlist(start(g2[(l-1):l]))
  flag[which(pos.mr %in% flag1)] = rep(1, length(flag1))

  df.mr = data.frame("pos" = pos.mr, "wid" = wid.mr, "flag" = flag)
  return(df.mr)
}

#generate the distance dataframe
dis_gen = function(dir1,dir2){
  Files  = list.files(dir2, full.names=F)
  PST.1 = c()
  PST.2 = c()
  for(i in 1:length(Files)){
    file1 = paste0(dir1, "/", Files[i])
    file2 = paste0(dir2, "/", Files[i])

    pScore.1 = phasing(file1)
    pScore.2 = phasing(file2)
    PST.1    = rbind(PST.1, pScore.1)
    PST.2    = rbind(PST.2, pScore.2)
    print(i)
  }

  ##Filter out the "+++"
  PST.2f = PST.2[PST.2$flag == 0,]
  PST.1f = PST.1[which(PST.2$flag == 0),]

  ##updated_cds
  pos1.3  = PST.1f[PST.1f$wid == 3, ]$pos
  pos1.6  = PST.1f[PST.1f$wid == 6, ]$pos
  pos1.9  = PST.1f[PST.1f$wid == 9, ]$pos
  pos1.12 = PST.1f[PST.1f$wid == 12,]$pos

  ##mapped_cds
  pos2.3  = PST.2f[PST.2f$wid == 3, ]$pos
  pos2.6  = PST.2f[PST.2f$wid == 6, ]$pos
  pos2.9  = PST.2f[PST.2f$wid == 9, ]$pos
  pos2.12 = PST.2f[PST.2f$wid == 12,]$pos

  diff.3  = pos2.3  - pos1.3
  diff.6  = pos2.6  - pos1.6
  diff.9  = pos2.9  - pos1.9
  diff.12 = pos2.12 - pos1.12

  dat = data.frame(
    type     = c(rep(3,length(diff.3)),rep(6,length(diff.6)),rep(9,length(diff.9)
     ↪  ),rep(12,length(diff.12))),
    distance = c(diff.3,diff.6,diff.9,diff.12)
  )
  dat$type = as.factor(dat$type)

  dat
}

#ggplot of each window size
gg_win = function(dat,win){
  g = dat %>%
    ggplot(aes(x=distance,fill=type)) +
    geom_density(alpha=0.4) +
     ↪  scale_fill_manual(values=c("#F0E442","#CC79A7","#009E73","#0072B2")) +
    theme_bw() + labs(fill="gap-length",x="Distance",subtitle=paste0("Window size
     ↪  of ",win))
  g
}


###################################################################
#ouFig = "Figure/dis.mafft.pdf"
main = function(ouFig){
  dir1a   = "../Data_3/Mafft/updated_cds"
  dir1b   = "../Data_3/Mafft/mapped_cds"
  dir2a   = "Mafft/updated_cds"
  dir2b   = "Mafft/mapped_cds"
  dir3a   = "../Data_9/Mafft/updated_cds"
  dir3b   = "../Data_9/Mafft/mapped_cds"
  dir4a   = "../Data_12/Mafft/updated_cds"
  dir4b   = "../Data_12/Mafft/mapped_cds"

  dat1 = dis_gen(dir1a,dir1b)
  dat2 = dis_gen(dir2a,dir2b)
  dat3 = dis_gen(dir3a,dir3b)
  dat4 = dis_gen(dir4a,dir4b)

  gg1 = gg_win(dat1,3)
  gg2 = gg_win(dat2,6)
  gg3 = gg_win(dat3,9)
  gg4 = gg_win(dat4,12)

  #ggplot
  pdf(ouFig,onefile=T)
  gg    = ggarrange(gg1, gg2, gg3, gg4, labels=c("A","B","C","D"), ncol=2, nrow=2)
  print(gg)
  dev.off()
}

#######################################
args = commandArgs(trailingOnly=TRUE)
main(args[1])
```

```
#Nei Gojobori (1986) method
#pairwise dnds ratio

suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(stringr))
suppressPackageStartupMessages(library(seqinr))

#setwd("~/Dropbox (ASU)/Indel_project")

#Determine number of non-syn/syn sites
cal_sites = function(x,tag){
  unit.x  = substr(x, tag, tag + 2)
  toMatch = c('-', '\\+', '=', 'N')
  s.site  = 0

  if(grepl(paste0(toMatch, collapse = "|"), unit.x)){
    return(c(0, 0))
  }else{
    section = codon[[which(sapply(codon, function(X){unit.x %in% X}))]]
    for (k in 1:3) {
      Base     = DNA_BASES[substr(unit.x, k, k) != DNA_BASES]
      unit.sub = unit.x
      for (w in 1:length(Base)) {
        substr(unit.sub, k, k) = Base[w]
        if(unit.sub %in% section){
          s.site = s.site + 1/3
        }
      }
    }
  }
  n.site = 3 - s.site
  res = c(n.site, s.site)
  return(res)
}

#Determine number of non-syn/syn substitutions
cal_subs = function(x,y,tag){
  unit.x  = substr(x, tag, tag + 2)
  unit.y  = substr(y, tag, tag + 2)
  sub.s   = 0
  sub.n   = 0
  toMatch = c('-', '\\+', '=', 'N')

  if(grepl(paste0(toMatch, collapse = "|"), unit.x) || grepl(paste0(toMatch,
   ↪  collapse = "|"), unit.y)){
    return(c(sub.s, sub.n))
  }else{
    section = codon[[which(sapply(codon, function(X){unit.x %in% X}))]]
    mut.num = mapply(function(X, Y) sum(X!=Y), strsplit(unit.x, ""),
     ↪  strsplit(unit.y, ""))
    mut.pos = mapply(function(X, Y) which(X!=Y), strsplit(unit.x, ""),
     ↪  strsplit(unit.y, ""))
    if(mut.num == 1){#ATG/TTG
      if(unit.y %in% section){
        sub.s = sub.s + 1
      }else{
        sub.n = sub.n + 1
      }
    }else if(mut.num == 2){#ATG/TCG
      unit.list = collect_allSubs(unit.x, unit.y, mut.pos)
      sub.rate  = cal_subsub(unit.list, sub.s, sub.n, section)
      sub.s = sub.rate[1] / 2
      sub.n = sub.rate[2] / 2
    }else if(mut.num == 3){#ATG/CCT
      unit.list = collect_allSubs(unit.x, unit.y, mut.pos)
      sub.rate  = cal_subsub(unit.list, sub.s, sub.n, section)
      sub.s = sub.rate[1] / 6
      sub.n = sub.rate[2] / 6
    }else{#no substitution ATG/ATG
      return(c(sub.s, sub.n))
    }
  }
  return(c(sub.s, sub.n))
}

# Find all possible substitution pathways.
# unit.x = "TTT"
# unit.y = "TAC"
# unit.y = "GAC"
# mut.pos = 2:3
# mut.pos = 1:3
collect_allSubs = function(unit.x,unit.y,mut.pos){
  unit.set  = c()
  for (i in 1:length(mut.pos)) {
    unit.sub = unit.x
    substr(unit.sub, mut.pos[i], mut.pos[i]) = substr(unit.y, mut.pos[i],
     ↪  mut.pos[i])
    unit.set = c(unit.set, unit.sub)
    if (length(mut.pos) == 3) {
      for (j in mut.pos[mut.pos != mut.pos[i]]) {
        unit.sb               = unit.sub
        substr(unit.sb, j, j) = substr(unit.y, j, j)
        unit.set              = c(unit.set, unit.sb)
      }
    }
  }
  unit.list = list()
  if(length(unit.set) == 2){
    for (k in 1:length(unit.set)) {
      unit.list = c(unit.list, list(c(unit.x, unit.set[k], unit.y)))
    }
  }else{
    for (k in 1:length(unit.set)) {
      if (k %% 3 == 1) {
        for (w in (k + 1):(k + 2)) {
          unit.list = c(unit.list, list(c(unit.x, unit.set[k], unit.set[w],
           ↪  unit.y)))
        }
      }
    }
  }
  return(unit.list)
}
```

```
# Sum the number of substitutions of all possible pathways
# section = c("TTT","TTC")
# 12 , 6
cal_subsub = function(unit.list,sub.s,sub.n,sec){
  for (m in 1:length(unit.list)) {
    sec.var = sec
    for (n in 2:(length(unit.list[[m]])) ) {
      if(unit.list[[m]][n] %in% sec.var){
        sub.s = sub.s + 1
      }else{
        sub.n = sub.n + 1
      }
      sec.var = codon[[which(sapply(codon, function(X){unit.list[[m]][n] %in%
       ↪  X}))]]
    }
  }
  return(c(sub.s, sub.n))
}


############################################################
#inD = "test_human_mouse_rat/Data_6/Mafft/mapped_cds"
#ouF = "test_human_mouse_rat/Data_6/Results/dnds.txt"

main = function(inD,ouF){

  #Create a codon table
  codon <<- list (c("TTT","TTC"),
                  c("TTA","TTG","CTT","CTC","CTA","CTG"),
                  c("ATT","ATC","ATA"),
                  c("ATG"),
                  c("GTT","GTC","GTA","GTG"),
                  c("TCT","TCC","TCA","TCG","AGT","AGC"),
                  c("CCT","CCC","CCA","CCG"),
                  c("ACT","ACC","ACA","ACG"),
                  c("GCT","GCC","GCA","GCG"),
                  c("TAT","TAC"),
                  c("CAT","CAC"),
                  c("CAA","CAG"),
                  c("AAT","AAC"),
                  c("AAA","AAG"),
                  c("GAT","GAC"),
                  c("GAA","GAG"),
                  c("TGT","TGC"),
                  c("TGG"),
                  c("CGT","CGC","CGA","CGG","AGA","AGG"),
                  c("GGT","GGC","GGA","GGG"),
                  c("TAA","TGA","TAG")
  )

  Files = list.files(inD, full.names=TRUE)

  n.m  = 0 #number of non-synonymous sites (mouse)
  s.m  = 0 #number of synonymous sites (mouse)
  n.r  = 0 #number of non-synonymous sites (rat)
  s.r  = 0 #number of synonymous sites (rat)
  Nd   = 0 #number of non-synonymous mutations
  Sd   = 0 #number of synonymouse mutations

  for (i in 1:length(Files)) {
    dna    = readBStringSet(Files[i])
    len    = width(dna)[1]

    M      = toupper(toString(dna[1]))
    R      = toupper(toString(dna[2]))

    #Cal. the dN/dS of each species
    j = 1
    while (j<len) {
      nSite_M  = cal_sites(M,j)
      n.m      = n.m + nSite_M[1]
      s.m      = s.m + nSite_M[2]

      nSite_R  = cal_sites(R,j)
      n.r      = n.r + nSite_R[1]
      s.r      = s.r + nSite_R[2]

      Subs     = cal_subs(M,R,j)
      Sd       = Sd + Subs[1]
      Nd       = Nd + Subs[2]

      j = j+3
    }
    print(i)
  }

  #cal. the W (dN/dS)
  non.num = 0.5*(n.m+n.r)
  syn.num = 0.5*(s.m+s.r)

  PN = Nd/non.num
  PS = Sd/syn.num

  #observed vs expected ratio
  obs.r = Nd/Sd
  exp.r = non.num/syn.num
  cat(sprintf("obs.ratio:%.3f exp.ratio:%.3f\n", obs.r,exp.r))

  #jukes cantor formula (1969)
  dNdS = log(1-4*PN/3)/log(1-4*PS/3)

  #output dnds
  write(dNdS,file=ouF)
}

########################################
args = commandArgs(trailingOnly=TRUE)
main(args[1], args[2])
```

cha4: Gillespie Simulation

```
suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(stringr))
suppressPackageStartupMessages(library(seqinr))
suppressPackageStartupMessages(library(plyr))
suppressPackageStartupMessages(library(R.utils))

#setwd("~/Dropbox (ASU)/Indel_project/chapter3")

#deletion starts to the right(pos included)
sim_del = function(s, dnaB, pos, posB, k){
  po = pos %% 3
  flag = 0
  if(po == 2){#phase 1
    ref    = paste0(s[pos-1],  s[pos+k], s[pos+k+1],collapse="")
    sub1   = paste0(s[pos-1],  s[pos],   s[pos+1],  collapse="")
    sub2   = paste0(s[pos+k-1],s[pos+k], s[pos+k+1],collapse="")
    sec    = syn[[ref]]
    if(sub1 %in% sec || sub2 %in% sec){
      flag = 1
    }
  }else if(po == 0){#phase2
    ref    = paste0(s[pos-2],  s[pos-1],  s[pos+k],collapse="")
    sub1   = paste0(s[pos-2],  s[pos-1],  s[pos],  collapse="")
    sub2   = paste0(s[pos+k-2],s[pos+k-1],s[pos+k],collapse="")
    sec    = syn[[ref]]
    if(sub1 %in% sec || sub2 %in% sec){
      flag = 1
    }
  }else{#phase0
    flag = 1
  }

  #AAA A-- -AA
  #AAA ANN NAA
  #--- --- NAA
  if((flag==1) || (runif(1L)<=Wz)){#selection check
    i=0
    while(i<k){
      #AAA --- A-- -AA
      #AA- --- --- --A
      if(dnaB[posB+i]=='-'){
        k=k+1
      }else{
        dnaB[posB+i]='-'
      }
      i=i+1
    }
  }

  return(dnaB)
}

#substition starts on the choosing position.
sim_sub = function(s, dnaB, pos, posB, gtr){
  mu = sample(DNA_BASES, 1L, prob = gtr[which(DNA_BASES == s[pos]),])  #If nuc A,
  ↪   it must change to C,G,T.

  po = pos %% 3
  flag = 0
  if(po == 2){#phase 1
```

```
    flag  = 0
  }else if(po == 0){#phase2
    ref = paste0(s[(pos-2):pos],collapse="")
    sub = paste0(s[pos-2],s[pos-1],mu,collapse="")
    sec = syn[[ref]]
    if(sub %in% sec){
      flag = 1
    }
  }else{#phase0
    ref = paste0(s[pos:(pos+2)],collapse="")
    sub = paste0(mu, s[pos+1],s[pos+2],collapse="")
    sec = syn[[ref]]
    if(sub %in% sec){
      flag = 1
    }
  }

  if((flag==1) || (runif(1L)<W)){
    dnaB[posB] = mu
  }
  return(dnaB)
}

#insertion starts to the left(including immortal link)
sim_ins = function(s, dnaB, pos, posB, k){
  ins = sample(DNA_BASES,k,prob=Pi,replace=TRUE)  #throw a k-sided dice for
   ↪  inserted nucs.

  #type-N:0, type-S:1
  po   = pos %% 3
  flag = 0

  if(po == 2){#phase2
    ref     = paste0(s[pos-1], s[pos], s[pos+1], collapse="")
    sub1    = paste0(s[pos-1], s[pos], ins[1],   collapse="")
    sub2    = paste0(ins[k-1], ins[k], s[pos+1], collapse="")
    sec     = syn[[ref]]
    if(sub1 %in% sec || sub2 %in% sec){
      flag = 1
    }
  }else if(po == 1){#phase1
    ref     = paste0(s[pos],  s[pos+1], s[pos+2], collapse = "")
    sub1    = paste0(s[pos],  ins[1],   ins[2],   collapse = "")
    sub2    = paste0(ins[k],  s[pos+1], s[pos+2], collapse = "")
    sec     = syn[[ref]]
    if(sub1 %in% sec || sub2 %in% sec){
      flag = 1
    }
  }else{#phase0
    flag = 1
  }

  #selection check
  if( (flag==1) || (runif(1L)<=Wz) ){
    newB = append(dnaB,ins,posB)
    stamp = 1
  }else{
    newB  = dnaB
    stamp = 0
  }
  res = list(newB,stamp)
  return(res)
}

#sequential simulation with same brlen#######
D_onestep = function(r1,ext,dna0){
  dnaB = dna0
  tau  = 0
  #total rates
  phase   = (seq_along(dnaB)+2) %% 3 + 1
  phase.r = cbind(r1[phase],rep(0,length(phase)))

  for (iter in 1:500) {
    #no events should occur again on the deletion box
    phase.r[which(dnaB=='-')] = 0

    #sample position and mutation types
    Pos  = sample(nrow(phase.r), 1L, prob=phase.r[,1])

    #Poisson waiting time
    total_rate = sum(phase.r[,1])
    tau = tau + rexp(1,total_rate)
    if((total_rate==0) || (tau>=brlen)){
      break
    }

    #count number of gaps before current position
    ngap = length(which(phase.r[(1:Pos),1]==0))

    #update dna
    dna  = dnaB[which(dnaB!='-')]
    pos  = Pos-ngap

    #draw geometric dist of indel length
    k = rgeom(1,1-ext)
    k = 3*(k+1)
    if(length(dna)-pos<k){#delete the end
      next
    }

    dnaB = sim_del(dna, dnaB, pos, Pos, k)
  }
  cat(sprintf("iter:%d  tau:%.6f\n", iter,tau))

  if((length(dnaB)!=length(dna0)) || (length(dnaB) %%3 !=0)){
    print("Warning:after deletion, alignment length should be the same!")
  }
  return(dnaB)
}

S_onestep = function(r2,gtr,dnaB){
  tau = 0
  #total rates
  phase = (seq_along(dnaB)+2) %% 3 + 1
  for (iter in 1:1000) {
    #update the mu
    phase.r = cbind(r2[dnaB],rep(0,length(phase)))

    #sample position and mutation types
    Pos  = sample(nrow(phase.r), 1L, prob=phase.r[,1])

    #Poisson waiting time
    total_rate = sum(phase.r[,1])
    tau = tau + rexp(1,total_rate)
    if((total_rate==0) || (tau>=brlen)){
      break
    }

    #count number of gaps before current position
    ngap = length(which(phase.r[(1:Pos),1]==0))

    #update dna
    dna  = dnaB[which(dnaB!='-')]
    pos  = Pos-ngap
    dnaB = sim_sub(dna, dnaB, pos, Pos, gtr)
  }

  cat(sprintf("iter:%d  tau:%.6f\n", iter,tau))
  return(dnaB)
}

I_onestep= function(r1,ext,dnaB1){
  tau = 0

  #total rates
  phase.r = matrix(0,length(dnaB1)+1,2)
  phase   = (seq_along(dnaB1)) %% 3 + 1
  phase.r[1,1] = r1[1]
  phase.r[2:nrow(phase.r),] = cbind(r1[phase],rep(0,length(phase)))
  #no events should occur again on the deletion box
  phase.r[which(dnaB1=='-')+1] = 0


  for (iter in 1:500) {
    #sample position and mutation types
    Pos  = sample(nrow(phase.r), 1L, prob=phase.r[,1])

    #Poisson waiting time
    total_rate = sum(phase.r[,1])
    tau = tau + rexp(1,total_rate)
    if((total_rate==0) || (tau>=brlen)){
      break
    }

    #draw geometric dist of indel length
    k = rgeom(1,1-ext)
    k = 3*(k+1)

    #count number of gaps before current position
    posB = Pos - 1
    if(posB == 0){#immortal link
      ins      = sample(DNA_BASES,k,prob=Pi,replace=TRUE)
      dnaB1    = append(dnaB1,ins,posB)
      Isinsert = 1
    }else{
      ngap     = length(which(phase.r[(1:posB),1]==0))
      #update dna
      dna      = dnaB1[which(dnaB1!='-')]
      pos      = posB-ngap
      dnaBs    = sim_ins(dna, dnaB1, pos, posB, k)
      dnaB1    = dnaBs[[1]]
      Isinsert = dnaBs[[2]]
    }

    if(Isinsert==1){
      Bi = append(phase.r[,2],rep(1,k),Pos)
      if(Pos %% 3 == 1){#phase0
        Rate = append(phase.r[,1],rep(r1[c(2,3,1)],k/3),Pos)
      }else if(Pos %% 3 == 2){#phase1
        Rate = append(phase.r[,1],rep(r1[c(3,1,2)],k/3),Pos)
      }else{#phase2
        Rate = append(phase.r[,1],rep(r1,k/3),Pos)
      }
      phase.r = cbind(Rate,Bi)
    }
  }

  cat(sprintf("iter:%d  tau:%.6f\n", iter,tau))
  if(nrow(phase.r)!=length(dnaB1)+1){
    print("Warning: immortal link does not exist!")
  }
  res = list(dnaB1,phase.r)
  return(res)
}

#Align it back
align_back = function(dna0,dnaB2){
  A    = dna0
  A    = insert(A,1,'#')
  B    = dnaB2[[1]]
  bi   = dnaB2[[2]][,2]

  bipos = which(bi==1)
  if(length(bipos)!=0){
    for(i in 1:length(bipos)){
      A = append(A,'-',bipos[i]-1)
    }
  }
  A = A[-1]
  if(length(A)!=length(B)){
    print("Warning: someting wrong with the back alignment!")
  }

  DnaA  = paste0(A,collapse="")
  DnaB  = paste0(B,collapse="")
  Align = BStringSet(c(DnaA,DnaB))
  return(Align)
```

```
}



#setwd("~/Dropbox (ASU)/Indel_project/chapter3")
#######################################################
main = function(ouD,inF,l,omega_z,ss){

  #read input
  #ouD = "Gs/98"
  #inF = "trueP.100.txt"
  cmd = paste0("mkdir -p ",ouD, sep=' ')
  system(cmd)

  tag = as.numeric(str_extract(basename(ouD), "[^.]+"))
  tP  = read.table(inF,header=T,sep='')
  tp  = unlist(tP[tag,])

  Pi    <<- tp[1:4]
  Sigma <<- tp[5:10]
  W     <<- tp[11]
  brlen <<- tp[12]
  ext    = tp[13:14]
  r1     = tp[15:17]/(2*brlen)

  omegaz = as.numeric(omega_z)  #znzs: omega_z=1 (default)
  Wz     <<- 1


  # construct codons and its degeneracy
  codons = cbind(rep(DNA_BASES, each = 16),
                 rep(DNA_BASES, times = 4, each = 4),
                 rep(DNA_BASES, 16))
  codonstrs  = apply(codons, 1, stringr::str_c, collapse = "")
  syn        = syncodons(codonstrs)
  names(syn) = toupper(names(syn))
  syn        <<- lapply(syn, toupper)

  #construct GTR model
  #Sigma = c(0.7135135,2.9447236,0.3246753,1.1608040,2.9123377,0.6428571)
  #Sigma  = 1:6
  r      = matrix(0,4,4)
  r[lower.tri(r)] = Sigma
  r       = r + t(r)
  r       = t(r*Pi)
  diag(r) = -rowSums(r)
  T   = -sum(diag(r)*Pi)
  gtr = r / T
  print(-sum(diag(gtr)*Pi))

  r2        = -diag(gtr)
  r2[5]     = 0
  names(r2) = c(DNA_BASES,'-')
  diag(gtr) = 0     #avoid negative prob.


  #########################################PART II Run simulation
  set.seed(8088)
  len    = as.numeric(l)
  dna0   = sample(DNA_BASES, len, prob=Pi, replace=TRUE)
  dna00 = paste0(dna0,collapse='')

  sim.IDS = list()
  ssize    = as.numeric(ss)
  for(i in 1:ssize){
    dnaB   = D_onestep(r1,ext[2],dna0)
    dnaB1 = S_onestep(r2,gtr,dnaB)
    dnaB2 = I_onestep(r1,ext[1],dnaB1)
    Align = align_back(dna0,dnaB2)

    names(Align) = c('Seq1','Seq2')
    writeXStringSet(Align, paste0(ouD,'/',i,".fa"))
    sim.IDS[[i]] = Align
  }
}


########################################
args = commandArgs(trailingOnly=T)
main(args[1],args[2],args[3],args[4],args[5])
```

cha4: EM + Importance Sampling

```
# EM + importance sampling method

# Apply the importance sampling Wi = f/g
# f(x)--ziqi's phase_coati model and (x)--juan's coatiM
# A -- ancestor, B -- descendent

suppressWarnings(suppressMessages(library(tidyverse)))
suppressPackageStartupMessages(library(Matrix))
suppressPackageStartupMessages(library(stringr))
suppressPackageStartupMessages(library(seqinr))
suppressPackageStartupMessages(library(stringi))
suppressPackageStartupMessages(library(Biostrings))
suppressPackageStartupMessages(library(jsonlite))
suppressPackageStartupMessages(library(dfoptim))
suppressPackageStartupMessages(library(doParallel))
suppressPackageStartupMessages(library(profvis))


######################################functions
#Count the nucleotide freq
count_freq = function(input){
  nuc.count = 0
  for (i in input){
    dna        = readDNAStringSet(i)
    nuc.count = nuc.count + oligonucleotideFrequency(dna,width=1)
  }
  nuc.freq   = colSums(nuc.count)/sum(nuc.count)
  if(sum(nuc.freq)==1){
    return(nuc.freq)
  }else{
    print("The nucleotide frequency sums up not 1!")
  }
}

#Read sample
read_sample = function(D,K){
  Alist = list()
  for (i in 1:K) {
    Alist[[i]] = DNAStringSet(c(D$Seq1[i],D$Seq2[i]))
  }
  return(Alist)
}

#-Log-likelihood of
LL_min  = function(theta){
  nuc_codon_freq = init_f(cod,DD,ncd)
  fw  = nuc_codon_freq[[1]]
  cfw = nuc_codon_freq[[2]]

  rmat = GTR(theta[1:6], fw)
  Rmat = MG94(rmat,theta[7],cod,codonstrs,syn,ncd)
  -sum(log(expm(Rmat)*cfw)*DD)
}

#remove effect of scaling factor [ancestor/descendant]
test_pab = function(ab,f0){
  dnaAB    = DNAStringSet(gsub('-','',ab))
  ab.count = oligonucleotideFrequency(dnaAB,width=1)
  sumPab   = sum(ab.count[1,]*log(f0)) + sum(ab.count[2,]*log(f0))
  return(sumPab)
}




#####################################################################
# ouD   = "JsonD/58/"
# inD   = "Gs_trim/58"

# inD   ="../test_90_species/Raw_data/cds/06_Nematode_aligned_cds"
# ouD   ="90/JsonD/06_Nematode_aligned_cds/"
# ouF   ="90/Results/PISE/06_Nematode_aligned_cds.est.json"
# ssize ="100"
# ncdon ='61'
# spList='90/species/06_Nematode_aligned_cds.txt'
main = function(inD,ouD,spList,ouF,ssize,ncdon){

  sub1 = "../Script/sources/"
  sub2 = "../Script/chapter3/"
  source(paste0(sub1,"codon_call.R"))
  source(paste0(sub1,"gtr.R"))
  source(paste0(sub1,"mg94.R"))
  source(paste0(sub1,"init_f.R"))
  source(paste0(sub1,"LL.R"))
  source(paste0(sub2,"phase_indel_prob3.R"))

  #construct codons and its degeneracy
  ncd      <<- as.numeric(ncdon)
```

```
co.res     = codon_call(ncd)
codons     = co.res[[1]]
codonstrs = co.res[[2]]
syn        = co.res[[3]]

#nmkb method
cod       <<- codons
codonstrs<<- codonstrs
syn       <<- syn

######################################
Files = list.files(inD, full.names=T)
namev = str_extract(basename(Files),'[^.]+')

if(namev[1]=='1'){#simulation
  index = as.numeric(namev)
  Files = Files[order(index)]
  tag   = as.numeric(basename(inD))
}else{#90 species
  Fbname = list.files(inD, full.names=F)
  kept   = read.table(spList,header=F)[,1]
  index  = which(Fbname %in% kept)
  Files  = Files[index]
}
n = length(Files)


#Initial parameters
set.seed(8088)
f0  = count_freq(Files)
p0  = rep(0.1,7)

g0  = rep(0.01,6)
e0  = rep(0.8,2)  #codon as unit

rmat= GTR(p0[1:6],f0)
t0  = -sum(diag(rmat)*f0)

#Iterate through the black box
##default:ssize=100
max.it = 20
K      = as.numeric(ssize)
lljv   = matrix(NA,n,max.it)
llzv   = matrix(NA,n,max.it)

N      = matrix(NA,n,6)
M      = matrix(NA,n,6)
E      = matrix(NA,n,2)
colnames(N) = c('i0','i1','i2','d0','d1','d2')
colnames(M) = c('no_i0','no_i1','no_i2','no_d0','no_d1','no_d2')
colnames(E) = c('avg.gap.len.I','avg.gap.len.D')

Wv = list()


#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>RUNNING ALGO
iter=1
```

```
tm = system.time({
  repeat{
    D = matrix(0,ncd,ncd)
    if(iter>max.it){
      cat("Pass the", max.it, "iterations limits!")
      break}

    #pre-cal avg gap size
    avg.gap = 1/(1-e0)
    if(any(avg.gap<3)){
      print("Average gap length less than 3!")
      break
      }
    e3    = 1-1/(avg.gap/3)

    #prellocate gtr/mg94 mat, codon freq.
    cf0   = sapply(seq(ncd), function(x){prod(f0[match(cod[x,],DNA_BASES)])})
    cf0   = cf0/sum(cf0)
    Rmat  = MG94(rmat,p0[7],cod,codonstrs,syn,ncd)
    Pmat  = log(expm(Rmat))

    for (j in 1:n) {#E-step
      ##coati-sampler
      input = Files[j]
      ouJ   = paste0(ouD,j,'.json')
      cmd   = paste("bash ../Script/chapter3/coati_sampler.sh", input, ouJ, K,
                    mean(g0),mean(e0),f0[1],f0[2],f0[3],f0[4],p0[1],p0[2],p0[3] ⌋
                     ↪   ,p0[4],p0[5],p0[6],p0[7],t0,sep='
                     ↪   ')
      system(cmd)
      Input.tmp    = fromJSON(ouJ)
      Dat.tmp      = Input.tmp %>% tidyr::unpack(aln)

      colnames(Dat.tmp)[1:2] = c('Seq1','Seq2')
      aln      = Dat.tmp %>% dplyr::group_by(Seq1,Seq2)
      aln2     = aln %>% dplyr::group_keys(Seq1,Seq2,weight,log_weight)
      gpsize   = group_size(aln)
      ngroups  = n_groups(aln)

      print(ngroups)

      llj          = aln2$log_weight
      lljv[j,iter] = sum(llj*gpsize)/K
      Alist        = read_sample(aln2,ngroups)
      #llj2 = exp(llj)/sum(exp(llj)*gpsize)
      #print(sum(llj2*gpsize))

      ##preset summary stats
      Gm        = matrix(0,ngroups,6)
      Mm        = matrix(0,ngroups,6)
      codon_arr = array(0,c(ncd,ncd,ngroups))
      llz       = rep(0,ngroups)
      gl        = matrix(0,ngroups,2)

      ##adding scalling factor
      scal.pab = test_pab(Alist[[1]],f0)
      #profvis({
        for(i in 1:ngroups){
          A      = Alist[[i]]
          res1   = ziqi_prob(A,g0,e3,Pmat,codonstrs,f0,ncd)
          Gm[i,] = res1[[1]]
          Mm[i,] = res1[[2]]
          codon_arr[,,i] = res1[[3]]
          llz[i] = res1[[4]]-scal.pab
          gl[i,] = res1[[5]]
        }
      #})


      llzv[j,iter] = sum(llz*gpsize)/K
      if(any(llz==-Inf)){#low-quality alignment
        next
      }

      ############
      #cal. weight
      uwi           = llz-llj
      uwi.max       = max(llz-llj)
      uwi.dif       = uwi - uwi.max
      Wi            = exp(uwi.dif)/sum(exp(uwi.dif)*gpsize)
      print(sum(Wi*gpsize))
      Wv[[j]]       = Wi*gpsize


      ##weighted gap phases
      N[j,] = colSums(Wi*gpsize*Gm)
      M[j,] = colSums(Wi*gpsize*Mm)

      ##weighted average gap length
      #E[j,] = colSums(Wi*gl*gpsize,na.rm=T) #>
      E[j,] = colSums(Wi*gl*gpsize)

      datw = matrix(0,ncd,ncd)
      for (j in 1:ngroups) {
        dat.wei = Wi[j]*codon_arr[,,j]
        datw    = datw + dat.wei*gpsize[j]
      }
      D = D+datw/n
    }


    ############################
    #M step: para estimates
    ##gap opening
    nnew = colMeans(N,na.rm=T)
    mnew = colMeans(M,na.rm=T)
    gnew = nnew/(nnew+mnew)


    ##gap extension
    w.avg.gap = c(mean(E[which(E[,1]>=1),1]),mean(E[which(E[,2]>=1),2]))
    enew1     = 1 - 1/(w.avg.gap*3)
    enew      = 1 - 1/w.avg.gap
```

```r
    if(any(enew<0)){
      print("Warning: Updated gap extension prob (unit of 3) < 0!")
      break
      }

    DD <<- D
    pb = nmkb(fn=LL_min, par=p0, lower=0,
     ↪  control=list(tol=1e-5,trace=F,maxfeval=5000))   #change the tolerance
    if(pb$convergence != 0){
      cat("Warning: failed convergence!")
    }else{
      pnew = pb$par
    }

    ##cal. the tau
    nuc_f = init_f(cod,DD,ncd)
    fnew  = nuc_f[[1]]
    rmat  = GTR(pnew[1:6],fnew)
    tnew  = -sum(diag(rmat)*fnew)

    #print the results
    print(gnew[1]+gnew[4])
    print(gnew[2]+gnew[5])
    print(gnew[3]+gnew[6])
    print(mean(enew))
    print(pnew[1:6]/tnew)
    print(pnew[7])
    print(tnew)

    #rmse tolerance
    p    = c(g0,e0,p0)
    q    = c(gnew,enew1,pnew)
    delta= (q-p)^2
    rmse = sqrt(mean(delta))
    cat(sprintf("iter:%i, rmse:%.6f\n",iter, rmse))
    if(rmse<=1e-4){
      break
    }else{
      if(iter>10){
        K=1e+4
      }
      iter= iter+1
      f0  = fnew
      p0  = pnew
      g0  = gnew
      e0  = enew1
      t0  = tnew
    }
  }
})

cat(sprintf("Running loops: %i\n  Running time:%.3f mins", iter, tm[3]/60))
################################################################################

#output the parameter estimates summary stats
par_lst = list('nuc.freq'=fnew, 'sigmas'=pnew[1:6]/tnew, 'omega'=pnew[7],
 ↪  'branch.length'=tnew,
                 'gap.openning'=gnew,'gap.extension'=enew)
  sum_lst = list('gap.phases'=nnew,'avg.gap.size'=w.avg.gap)

  par_est = toJSON(par_lst)
  sum_stat= toJSON(sum_lst)

  dname = dirname(ouF)
  fname = str_extract(basename(ouF),'[^.]+')
  ouF2  = paste0(dname,'/',fname,'.sum.json')
  write(par_est,ouF)
  write(sum_stat,ouF2)

  #output the weight matrix (last weight)
  dname2 = gsub('(.*)/\\w+', '\\1', dname)
  ouF3   = paste0(dname2,'/weightM/',fname,'.txt')
  lapply(Wv,write,ouF3,append=T,ncolumns=1e+4)

  #output rmse
  ouF3   = paste0(dname2,'/RMSE/',fname,'.txt')
  write(rmse,ouF3,ncolumns=1)

  #output llzv matrix
  ouF4   = paste0(dname2,'/LLZ/',fname,'.tsv')
  write.table(llzv,ouF4,col.names=F,row.names=F,sep='\t')

}


################################
args = commandArgs(trailingOnly=T)
main(args[1],args[2],args[3],args[4],args[5],args[6])
```

```r
#1: output the prob of ziqi's model
#2: output the summuary stats

#Cal. the inserted freq prob
prob_insert = function(As2,g1,f0){
  if(length(g1)==0){
    return (0)
  }else{
      pos1 = start(g1)
      pos2 = end(g1)
      insN = c()
      for(j in 1:length(g1)){
        insN = c(insN,As2[pos1[j]:pos2[j]])
      }
  }
  f.rank = match(insN,DNA_BASES)
  f.prob = sum(log(f0[f.rank]))
  return(f.prob)
}


# 0 1 1; 0 1 0; 0 1 1; 0 2 0; 0 0 1; 0 0 2
#Count number of gap edges.
#Format: ins: del:
count_gap = function(As,g){
  phase = matrix(0,3,2)
  lG = c(0,0)
  lT = c(0,0)
  if(length(unlist(g)) == 0){
    return(list(phase,lG,lT))
  }else{#separate the ins from del
    for (j in seq(2)) {
      pos  = start(g[[j]])
      wid  = width(g[[j]])

      rem = pos %% 3
      phase[1,j] = length(which(rem == 1))
      phase[2,j] = length(which(rem == 2))
      phase[3,j] = length(which(rem == 0))
    }
  }
  #gap number; gap length
  lG = c(length(g[[1]]),length(g[[2]]))
  lT = c(sum(width(g[[1]])), sum(width(g[[2]])))/3

  res = list(phase,lG,lT)
  return(res)
}

# 4 3 3; 6 5 6; 4 3 3; 4 2 4; 6 6 5; 4 4 2
#1--- AAA AAA    2AAA --- AAA      3---AAA
#1AAA --- AAA    2AAA AAA ---      3AAAAAA

#Count number of no-gap edges
count_nogap = function(As,g,gCount){
  lenA  = length(As[[1]])
  ug    = unlist(g)
  lenT  = sum(width(ug))

  last0 = 0 #last site
  if(lenA %in% end(ug)){
    lastg0 = ug[which(end(ug) %in% lenA)]
    last0  = last0 + 1
    if((start(lastg0)-1) %in% end(ug)){#2
      seclastg0 = ug[which((start(lastg0)-1) %in% end(ug))]
      last0     = last0 + 1
    }
  }

  endflag = 0 #ins-1, del-2, match-0
  if(last0==1){
    end1 = end(g)[[1]]
    if(end(lastg0) %in% end1){#ins>end
      gCount[1,1] = gCount[1,1]-1
      endflag = 1
    }else{#del>end
      gCount[1,2] = gCount[1,2]-1
      endflag = 2
    }
  }else if(last0==2){#ins>del>end
    gCount[1,] = gCount[1,]-1
    endflag    = 2
  }

  exp.edge = (lenA-lenT)/3
  M        = matrix(0,3,2)
  for (j in 1:3) {
    M[j,]   = exp.edge - gCount[j,]
  }
  res = list(M,endflag)
  return(res)
}


#Remove all gap-positioned string
rmgap = function(A,As,g){
  g    = unlist(g)
  rm.A =
  ↪  stri_sub_replace_all(A,from=sort(start(g)),to=sort(end(g)),replacement='')
  rm.A = DNAStringSet(rm.A)
  if(all(width(rm.A)%%3 == 0)){
    return(rm.A)
  }else{
    print("Warning:removed-gap sequences are not multiple of three")
    break
  }
}

#Generate obs-codon matrix
countN = function(rA, codonstrs, ncd){
  seqs = str_split(rA,'')
  len  = length(seqs[[1]])
  nmat = matrix(0,ncd,ncd)
  i=1
  while(i<len) {
    c1 = paste0(seqs[[1]][i:(i+2)], collapse = '')
    c2 = paste0(seqs[[2]][i:(i+2)], collapse = '')
    coor1 = which(codonstrs %in% c1)
    coor2 = which(codonstrs %in% c2)
    nmat[coor1,coor2] = nmat[coor1,coor2] + 1
    i=i+3
  }

  return(nmat)
}




ziqi_prob = function(A,g0,e3,Pmat,codonstrs,f0,ncd){

  As = str_split(A,'')
  g  = IRangesList(lapply(As, function(x){IRanges(x=='-')}))

  #cal the inserted freq prob
  insP = prob_insert(As[[2]],g[[1]],f0)


  #count number of gap edges
  Cg    = count_gap(As,g)
  N.012 = Cg[[1]]
  num.g = Cg[[2]]
  len.g = Cg[[3]]

  Cng   = count_nogap(As,g,N.012)
  M.012 = Cng[[1]]
  Eflag = Cng[[2]]


  ##end state prob
  if(Eflag==0){#match
    endP = log(1-g0[1])
    endP = unname(endP)
  }else if(Eflag==1){#ins
    endP = log(1-e3[1])
  }else{#del
    endP = log(1)
  }

  #extension prob of ins/del
  avg.g = len.g/num.g
  if(all(len.g==0)){
    avg.g = c(0,0)
  }else if(len.g[1]==0){
    avg.g[1] = 0
  }else if(len.g[2]==0){
    avg.g[2] = 0
  }
  scoreE = (len.g[1]-num.g[1])*log(e3[1])+num.g[1]*log(1-e3[1]) +
   ↪  (len.g[2]-num.g[2])*log(e3[2])+num.g[2]*log(1-e3[2])
  ##transition prob
  scoreT = log(prod(g0^N.012)) + sum(log((1-g0)^M.012))
  #print(scoreT+scoreE+endP)

  rA     = rmgap(A,As,g)
  dat    = countN(rA,codonstrs,ncd)
  scoreP = sum(Pmat*dat)

  ##sum LL
  score_ziqi = scoreE + scoreT + endP + scoreP + insP
  if(score_ziqi==-Inf){
    print("minus infinity z score!")
  }
  #summary stat
  res = list(c(N.012),c(M.012),dat,score_ziqi,avg.g)

  #print(score_ziqi)
  return(res)
}
```

cha4: Parameter Space Setup

```
library(expm)
library(stats)
library(Biostrings)

#GTR matrix
GTR = function(si, pai){
  r1 = matrix(0, 4, 4)
  r1[lower.tri(r1)] = si
  r  = r1 + t(r1)
  r  = t(r*pai)
  diag(r) = -rowSums(r)
  return(r)
}


#######################################
set.seed(8088)
#set up the nuc freq.
#narrow the freq range so the repeats would be less.
nd  = 100
PiA = runif(nd,0.1,0.4)
PiT = PiA
PiC = 0.50 - PiA
PiG = PiC
Pi.all = list(PiA, PiC, PiG, PiT)
Pi.all = sapply(1:nd, function(x){sapply(Pi.all, "[[", x)})
#print(colSums(Pi.all))


#set up omega from zhou's paper
wv = runif(nd,0.02,0.5)

#keep the brlen between [0,0.1]
#set up 6 sigmas, choose the mean rate as 0.1, lower the cv
cv = 0.5
a  = 1/(cv^2)
b  = 0.1/a                    #>>adjustable
Sigmas = matrix(0,6,nd)
tv = c()
for (i in 1:nd) {
  pai        = Pi.all[,i]
  si         = rgamma(6, shape=a, scale=b)
  gtr        = GTR(si, pai)
  Sigmas[,i] = si
  tv[i]      = -sum(diag(gtr)*pai)
}
#summary(tv)
#hist(tv, prob = TRUE, xlim = c(0,2))
#stv = sort(tv)
#lines(stv,dgamma(stv,shape=a,scale=b),col='magenta',lwd=2,lty='dotted')
#plot(density(tv))
#dev.off()

#normalize sigma.
norm.Sig = sapply(1:nd, function(x){Sigmas[,x]/tv[x]})
#print(norm.Sig)

#set up the indel rate ([12,16]/100)
#assume P_ins = P_del
r0  = runif(nd,0.05,0.15)
r1  = runif(nd,0.05,0.15)
r2  = runif(nd,0.05,0.15)

r       = matrix(c(r0,r1,r2),nd,3)
rt      = r*tv
g.open  = 1-exp(-r*tv)


#setup the gap extension
#pick a range for avg gap size and convert to extention prob
#T'=T/3, e=(T'-G)/T' -> T'/G=1/(1-e))
lower =3.5/3
higher=4
avg.gap1 = runif(nd,lower,higher)
avg.gap2 = runif(nd,lower,higher)
avg.gap = matrix(c(avg.gap1,avg.gap2),nd)
ext     = 1-1/avg.gap

##output the 100 parameters
tPar = matrix(0,100,17)
for (i in 1:nd) {
  tPar[i,1:4]   = Pi.all[,i]
  tPar[i,5:10]  = norm.Sig[,i]
  tPar[i,11]    = wv[i]
  tPar[i,12]    = tv[i]
  tPar[i,13:14]= ext[i,]
  tPar[i,15:17]= rt[i,]

}
colnames(tPar)=c('A','C','G','T','s1','s2','s3','s4','s5','s6','omega','tau','ext.I','ext.D','r0*t','r1*t','r2*t')
```

```r
write.table(tPar,"trueP.100.txt",quote=F,sep="\t",
            row.names=F)
```

cha5: Quality Control

```r
library(tidyverse)
library(mixtools)

setwd("~/Dropbox (ASU)/Indel_project/Script/90/supp")
set.seed(403497)
num_gamma_components = 2

files <- fs::dir_ls(path="../../../test_90_species/Raw_data/JCdis_sum",
↪   glob="*.max.tsv")

output <- tibble(
    name=character(),
    f1=numeric(), shape1=numeric(), scale1=numeric(),
    f2=numeric(), shape2=numeric(), scale2=numeric()
)

for(f in files) {
    dat      <-  read_tsv(f,show_col_types=FALSE)
    dat_name <-  basename(str_replace(f, ".max.tsv$", ""))

    cat(str_glue("Fitting {dat_name}...\n\n"))

    dat <- dat %>% mutate(
        total_len = lenA+lenB,
        diff = abs(lenA-lenB),
        f = (gapA_len+gapB_len)/total_len,
        fdiff = diff/total_len,
        fadj = f-fdiff
        )

    # Calculate Distances
    d <- dat %>% filter(fadj <= 0.5 & fdiff <= 0.1) %>%
        transmute(
            p = mismatch_count/(match_count+mismatch_count),
            t = -0.75*log(1-4/3*p)
        )

    #fit to gamma mixture
    #add tiny offset to manage t=0 data points
    #quiet mixtools's `cat()` calls.
    quiet_em <- quietly(gammamixEM)
    model    <- quiet_em(d$t+1e-8, k=num_gamma_components)
    model <- model$result

    shape <- model$gamma.pars[1,]
    scale <- model$gamma.pars[2,]
    f <- model$lambda
    output <- bind_rows(output, tibble(
        name = dat_name,
        f1 = f[1], shape1 = shape[1], scale1 = scale[1],
        f2 = f[2], shape2 = shape[2], scale2 = scale[2]
    ))
    write_csv(output, "fitem.csv")
}
```

~~~
---
title: "Distances"
output:
  pdf_document:
    latex_engine: pdflatex
    fig_width: 6.375
    fig_height: 8.875
mainfont: Source Sans Pro
fontsize: 11pt
geometry: margin=1in
header-includes:
  \AtBeginDocument{\let\maketitle\relax}
---

```{r, echo = FALSE, message = FALSE}
library(tidyverse)
library(colorspace)
library(ggpubr)

knitr::opts_chunk$set(echo = FALSE, dev = "cairo_pdf")  #carip_pdf is just a
 ↪  gradphic device
files <- fs::dir_ls(path="../../../test_90_species/Raw_data/JCdis_sum",
 ↪  glob="*.max.tsv")

num_gamma_components = 2

models <- read_csv("fitem.csv", show_col_types=FALSE) %>%
    nest(data=!name) %>%
    deframe() %>%
    map(as.list)
~~~

```{r, results="asis", fig.align="center"}
for(f in files) {
    dat      <- read_tsv(f, show_col_types=FALSE)
    dat_name <- basename(str_replace(f, ".max.tsv$", ""))

    dat <- dat %>% mutate(
        total_len = lenA+lenB,
        diff = abs(lenA-lenB),
        f = (gapA_len+gapB_len)/total_len,
        fdiff = diff/total_len,
        fadj = f-fdiff
        )

    # Calculate Distances
    d  <- dat %>% filter(fadj <= 0.5 & fdiff <= 0.1) %>%
        transmute(
            p = mismatch_count/(match_count+mismatch_count),
            t = -0.75*log(1-4/3*p)
        )

    model <- models[[dat_name]]

    # Calculate density function
    x <- d$t
    y <- model$f1*dgamma(x, shape=model$shape1, scale=model$scale1)
```

~~~
    y <- y + model$f2*dgamma(x, shape=model$shape2, scale=model$scale2)

    # Calculate CDF
    u <- model$f1*pgamma(x, shape=model$shape1, scale=model$scale1)
    u <- u + model$f2*pgamma(x, shape=model$shape2, scale=model$scale2)

    par(mfrow=c(3,1))

    tbl <- tibble(t=x,y=y,u=u)

    col_indigo      <- "#000831"
    col_salvia_blue <- "#96bfe6"
    #Visualize the density of t
    cz <- col_salvia_blue
    cz <- c(cz,darken(cz, amount=0.33))
    gg1 <- ggplot(tbl, aes(t)) +
      geom_histogram(aes(y=..density..),bins=40,fill=rep(cz,20)) +
      geom_line(aes(x=t,y=y), color=col_indigo, lwd=1.5) +
      xlab("Evolutionary Distance")

    #Visualize model fit
    cz <- col_salvia_blue
    cz <- c(cz,darken(cz, amount=0.33))
    gg2 <- ggplot(tbl, aes(u)) +
      geom_histogram(breaks=seq(0,1,0.05), fill=rep(cz,10)) +
      geom_hline(aes(yintercept=nrow(tbl)/20), lwd=1.5, lty=2, col=col_indigo) +
      xlab("Empirical CDF")

    #P-P plot
    gg3 <- ggplot(tibble(x=ppoints(length(u),a=0), y=sort(u)), aes(x=x,y=y)) +
      geom_abline(slope=1,intercept=0, lwd=1.5, lty=2, col="darkgray") +
      geom_line(lwd=1.5,col=cz[2]) +
      xlab("Theoretical CDF") + ylab("Empirical CDF")

    gg <- ggarrange(gg2 + theme(aspect.ratio=1),
          gg3 + theme(aspect.ratio=1), labels=c("B","C"), ncol=2, nrow=1)
    gg <- ggarrange(gg1, gg, labels=c("A", ""),
          ncol=1,nrow=2, heights=c(2,1))

    gg <- annotate_figure(gg, top=text_grob(dat_name,
               face = "bold", size=14))



    print(gg)

    cat("\\newpage")
}
```
~~~

```
rmarkdown::render("suppl.Rmd",output_file="../../../test_90_species/Figure/QC2/su⌋
↪  ppl.pdf")
```

cha5: NLS Correlation between dN/dS and Zn/(Zn+Zs)

```
#rolling mean of Zn/(Zn+Zs) vs dnds
#build model based on raw data and predict on rolling stats
suppressWarnings(suppressMessages(library(tidyverse)))
suppressPackageStartupMessages(library(ggpubr))
suppressPackageStartupMessages(library(zoo))
suppressPackageStartupMessages(library(minpack.lm))

#setwd("~/Dropbox (ASU)/Indel_project/test_90_species")

#cal. rolling stats(moving average) default:k=201
ma = function(x) rollmean(x, k=201)

#extract the species name
naming = function(x) sub("\\..*", "", basename(x))

#cal the GC %
gc_perc = function(x){
  dat    = read_tsv(x,show_col_types=F)
  datsum = unname(colSums(dat))
  gc.perc= (datsum[1]+datsum[2])/(datsum[3]+datsum[4])
  return(gc.perc)
}

#helper functions for logistic curve
logis = function(x,location,scale) {
  xx = (x-location)/scale
  1/(1+exp(-xx))
}

#helper functions for logistic  derivative
d_logis = function(x, location, scale) {
  xx = (x-location)/scale
  (1/scale)*exp(-xx)/((1+exp(-xx))^2)
}


#################################################################
# ouFig = "../test_human_mouse_rat/Figure/znzs/dnds_ZnZs.pdf"
#
# inD1   = "Results/ZD_sum"
# inD2   = "Results/GC"
# inF    = "Results/ZnZs/neutral_ZnZs.txt"
# inF1   = "MLE90.tab.csv"
# ouFig  = "Figure/ZnZs/roll_ZnZs_max.pdf"
# ouF    = "Results/ZnZs/coef_max.tsv"
# pat    = "max"

main = function(inD1,inD2,inF,inF1,ouFig,ouF,pat){
  Files1 = list.files(inD1,full.names=T,pattern=pat)
  Files2 = list.files(inD2,full.names=T)
```

```
N     = length(Files1)
namev = unlist(lapply(Files1, function(x){naming(x)}))
spnew  = c()
for (i in 1:length(namev)) {
  spnew[i] = str_remove(namev[i],"_aligned_cds")
  spnew[i] = str_remove(spnew[i],"_")
}

#neutral Zn/(Zn+Zs)
neuZ = read.table(inF,header=T)
neuZ = neuZ$neutral_cod_ratio

#cal. the GC%
gc.perc = unlist(lapply(Files2, function(x){gc_perc(x)}))

#cal. the tau
dat = read.table(inF1,header=T,sep=",",quote=NULL)
tau = dat$tau


coefv = matrix(0,N,6)
pdf(ouFig,onefile=T)
for(i in 1:N){
  print(i)
  dat = read_tsv(Files1[i],show_col_types=F)
  dat = dat %>% mutate(Pn=Nd/N, Ps=Sd/S)
  dat = dat %>% mutate(dn=-0.75*log(1-4*Pn/3),
                       ds=-0.75*log(1-4*Ps/3),
                       w =dn/ds)

  ##drop 1 row and sort by w
  dat = dat %>% filter(is.finite(w)) %>% arrange(w)
  dat = dat %>% mutate(p=Zn/(Zn+Zs), N=Zn+Zs)
  dat2= dat %>% filter(N>0)


  #############################################################
  ##calculate a logistic regression model using glm
  m      = glm(p ~ w, family=binomial(), weight=N, data=dat2)

  coe    = setNames(coefficients(m),NULL)
  scale  = 1/coe[2]
  mu     = -coe[1]/coe[2]
  y0     = logis(0,mu,scale)   #y-intercept
  slope0 = d_logis(0,mu,scale) #slope at the y-intercept

  ##choose an asymptote that is 20% larger than our y0
  ##this seems help prevent bad fits
  ybig = if(slope0>0) y0*1.2 else y0/1.2

  ##estimate starting parameters for our NLS model
  f = 2*abs(ybig-y0)
  s = f/(4*slope0)
  v = ybig - (slope0>0)*f

  ##run nls estimation
  nls_fit = nls(p ~ f*logis(w,0,s)+v, data=dat2, weights=N,
                start = list(s=s, f=f, v=v), control=list(warnOnly=T))


  #Check convergence
  if(nls_fit$convInfo$isConv) {
    #use estimated parameter estimates
    coe2 = coefficients(nls_fit)
    f    = coe2["f"]
    v    = coe2["v"]
    s    = coe2["s"]
  }

  ##Calculate final parameters
  y0     = f*0.5+v
  slope0 = f/(4*s)
  ymax   =  f+v
  ymin   =  v


  #################################
  tab    = tibble(w = ma(dat$w),
                 Zs= ma(dat$Zs),
                 Zn= ma(dat$Zn),
                 p = Zn/(Zn+Zs))

  #ypred = predict(m,list(w=tab$w),type='response')
  ypred = f*logis(tab$w,0,s)+v

  ##rmse
  rmse = sqrt(mean((tab$p-ypred)^2))

  ##correlation
  corr = cor(ypred,tab$p)

  ##gg-plot
  tablong = tab %>% pivot_longer(cols=Zs:Zn, names_to="Stat", values_to="Count")
  gg_a    = ggplot(tablong, aes(x=w, y=Count, color=Stat)) + geom_line() +
   ↪  theme(legend.position=c(.05,.95)) + labs(x='dnds') + xlim(0,0.6)

  gg_b    = ggplot(tab, aes(x=w, y=p)) + geom_line(col="#000000") +
   ↪  labs(x='dnds',y='Zn/(Zn+Zs)') + xlim(0,0.6) + ylim(0,0.6)
  gg_b    = gg_b + geom_hline(yintercept=neuZ[i], linetype="dashed",
   ↪  color="#661100") +
            annotate("text",x=0,y=0.6,label=sprintf("GC:%.3f",gc.perc[i]),hjust⌋
             ↪  =0)
             ↪  +
            annotate("text",x=0,y=0.57,label=sprintf("sample
             ↪  size:%i",nrow(dat)),hjust=0) +
            annotate("text",x=0,y=0.54,label=sprintf("neutral:%.3f",neuZ[i]),hj⌋
             ↪  ust=0,color="#661100")
  gg_B    = gg_b + geom_line(aes(x=w,y=!!ypred),color="#AA4499",inherit=F)

  gg_B1   = gg_B + {if(slope0>0) geom_hline(yintercept=ymax, linetype="dashed",
   ↪  color="#56B4E9")  else
            geom_hline(yintercept=ymin, linetype="dashed", color="#56B4E9")} +
              {if(slope0>0) annotate("text",x=0,y=0.51,label=sprintf("asymptote:%⌋
               ↪   .3f",ymax),hjust=0,color="#56B4E9")
               ↪   else
              annotate("text",x=0,y=0.51,label=sprintf("asymptote:%.3f",ymin),hju⌋
               ↪   st=0,color="#56B4E9")}

    gg_B1   = gg_B1 + annotate("text",x=0,y=0.48,label=sprintf("init
     ↪   slope:%.3f",slope0),hjust=0) +
                      annotate("text",x=0,y=0.45,label=sprintf("tau:%.3f",tau[i])⌋
                       ↪   ,hjust=0)
                       ↪   +
                      annotate("text",x=0,y=0.42,label=sprintf("r:%.3f",corr),hju⌋
                       ↪   st=0)

    gg      = ggarrange(gg_a, gg_B1, labels=c("A","B"), ncol=1,nrow=2)
    gg      = annotate_figure(gg,top=text_grob(namev[i],face="bold",size=10))
    print(gg)

    coefv[i,] = c(nrow(dat),rmse,corr,y0,slope0,ymax)
  }
  dev.off()

  ##output a table
  coe.tab = data.frame(species=spnew,neutral=neuZ,GC=gc.perc,
            yintercept=coefv[,4],init_slope=coefv[,5],ymax=coefv[,6],sample_size=⌋
             ↪   coefv[,1],rmse=coefv[,2],corr=coefv[,3])
  write.table(coe.tab, ouF, sep="\t",append=F, quote=F, row.names=F, col.names=T)
}


####################################################
args = commandArgs(trailingOnly=T)
main(args[1],args[2],args[3],args[4],args[5],args[6],args[7])
```