

Online Appendix

A unifying comparative phylogenetic framework including traits coevolving across interacting lineages

MARC MANCEAU^{1,3,4}, AMAURY LAMBERT^{2,3}, HÉLÈNE MORLON⁴

¹*Muséum National d'Histoire Naturelle, 75005 Paris, France;*

²*Laboratoire Probabilités et Modèles Aléatoires, UPMC Univ Paris 06, 75005 Paris, France;*

³*Center for Interdisciplinary Research in Biology, Collège de France, CNRS UMR 7241, 75005 Paris, France;*

⁴*Institut de Biologie, École Normale Supérieure, CNRS UMR 8197, 75005 Paris, France*

Content.— We provide here the full, uncut, Online Appendix to our paper entitled "A unifying comparative phylogenetic framework including traits coevolving across interacting lineages". The first section is exposed in the attached appendix of the paper, but we let it here for the sake of self-containment. Note that simple equation numbering (e.g. equation (5)) refers to equations printed in the main text, whereas equations exposed in this appendix are designated as, e.g. equation (S5).

CONTENTS

A	Derivation of the distribution in a general setting	2
B	Distribution for some models without interactions between lineages	6
C	Distribution for some models with interactions between lineages	18
D	Simulation and Inference	30
E	Tutorial : using the RPANDA code to study trait coevolution	35

22 A DERIVATION OF THE DISTRIBUTION IN A GENERAL SETTING

23 A.1 *The distribution of trait values is Gaussian*

24 Recall that a vector is Gaussian if all linear combination of its components follows a normal
 25 distribution. We will thus show by induction that all linear combinations of the traits follow a
 26 normal distribution.

27 The process of trait evolution starts either at the stem root with a vector of size d defined
 28 by the initial conditions $X_{\tau_0} = {}^{tr}(X_0^1, \dots, X_0^d)$, or at the crown root with a vector of size $2d$ defined
 29 by the initial conditions : $X_{\tau_0} = {}^{tr}(X_0^1, \dots, X_0^d, X_0^1, \dots, X_0^d)$, or at any other step, provided the initial
 30 conditions are Gaussian by assumption.

31 Now, assume that X_{τ_i} is a Gaussian vector.

32 Then, $\forall t \in (\tau_i, \tau_{i+1})$, after integration we have the following closed expression for the value
 33 of the process X_t .

$$X_t = e^{-tA_i} \left(e^{\tau_i A_i} X_{\tau_i} + \int_{\tau_i}^t e^{sA_i} a_i(s) ds + \int_{\tau_i}^t e^{sA_i} \Gamma_i(s) dW_s \right) \quad (\text{S1})$$

34 Moreover, we have, for any deterministic function Φ ([Gardiner et al. 1985](#)),

$$\int_{t_n}^t \Phi_s dW_s \sim \mathcal{N} \left(0, \int_{t_n}^t \Phi_s {}^{tr} \Phi_s ds \right)$$

35 Hence, X_t is a linear combination of Gaussian vectors, which makes it a Gaussian vector.

36 Last, suppose that at time τ_{i+1} , the j th branch splits, in which case the vector grows. All
 37 linear combinations of the components of X_t at time τ_{i+1}^- have a normal distribution. And the d
 38 additional components added at time τ_{i+1} belong to the components at time τ_{i+1}^- . It follows that
 39 all linear combinations of the new vector still have a normal distribution.

40 A.2 *Integrating the evolution of the distribution through each epoch*

41 Still assuming that we know the (Gaussian) distribution of X_{τ_i} at the beginning of an epoch
 42 (τ_i, τ_{i+1}) , a few more lines allow us to provide a closed formula for the distribution of X_t at all

time $t \in (\tau_i, \tau_{i+1})$. Indeed, using Equation (S1), and the fact that, if X and Y are two independent Gaussian vectors with expectation vectors respectively m_X and m_Y and covariance matrices respectively Σ_X and Σ_Y , then :

$$DX + d \sim \mathcal{N}(Dm_X + d, D\Sigma_X^{tr}D)$$

$$X + Y \sim \mathcal{N}(m_X + m_Y, \Sigma_X + \Sigma_Y)$$

It thus follows that, $\forall t \in [\tau_i, \tau_{i+1}]$,

$$m_t = e^{(\tau_i - t)A_i} m_{\tau_i} + \int_{\tau_i}^t e^{(s-t)A_i} a_i(s) ds \quad (4a)$$

$$\Sigma_t = (e^{(\tau_i - t)A_i}) \Sigma_{\tau_i}^{tr} (e^{(\tau_i - t)A_i}) + \int_{\tau_i}^t (e^{(s-t)A_i} \Gamma_i(s))^{tr} (e^{(s-t)A_i} \Gamma_i(s)) ds \quad (4b)$$

Applying these equations for $t = \tau_{i+1}$ thus gives the distribution of the trait vector at time τ_{i+1} , which is the result stated in Equations (4a, 4b) in the main text.

Remark that, unless one of the very first branches immediately dies at the beginning of the process at a fixed initial condition, the density of the tip distribution has support in \mathbb{R}^{nd} . One can check that Σ_t stays positive definite (implying that $\det \Sigma_t \neq 0$), even when some Γ_i are not positive definite (except the first one).

A.3 Evolution of the distribution through ODE resolution

The expectation and covariance formulae provided in Equations (4a, 4b) require to deal with an integral which is not always straightforward to compute. Alternatively, one can prefer to take the derivative of this expression, get a set of ODEs verified by the expectation and covariance elements through each epoch, and subsequently integrate the ODE system. We show now another way to derive this set of ODEs.

First, we write the stochastic differential equation on any epoch (τ_i, τ_{i+1}) and for each trait k , which is given in the most general setting by :

$$dX_t^{(k)} = \left(a_i^{(k)}(t) - \sum_{m=1}^{n_t d} A_i^{(k,m)} X_t^{(m)} \right) dt + \sum_{m=1}^{n_t d} \Gamma_i^{(k,m)}(t) dW_t^{(m)}$$

Itô's formula (Gardiner et al. 1985) then gives us :

$$\begin{aligned} d \left(X_t^{(k)} X_t^{(l)} \right) &= X_t^{(k)} dX_t^{(l)} + X_t^{(l)} dX_t^{(k)} + d \langle X_t^{(k)}, X_t^{(l)} \rangle \\ &= \left(a_i^{(l)}(t) X_t^{(k)} - \sum_{m=1}^{n_t d} A_i^{(l,m)} X_t^{(m)} X_t^{(k)} \right) dt + \sum_{m=1}^{n_t d} \Gamma_i^{(l,m)}(t) X_t^{(k)} dW_t^{(m)} \\ &\quad + \left(a_i^{(k)}(t) X_t^{(l)} - \sum_{m=1}^{n_t d} A_i^{(k,m)} X_t^{(m)} X_t^{(l)} \right) dt + \sum_{m=1}^{n_t d} \Gamma_i^{(k,m)}(t) X_t^{(l)} dW_t^{(m)} \\ &\quad + \sum_{m=1}^{n_t d} \Gamma_i^{(l,m)}(t) \Gamma_i^{(k,m)}(t) dt \end{aligned}$$

Taking the expectation, it follows that

$$\begin{aligned} \frac{d}{dt} \mathbb{E} \left(X_t^{(k)} X_t^{(l)} \right) &= a_i^{(l)}(t) \mathbb{E} \left(X_t^{(k)} \right) + a_i^{(k)}(t) \mathbb{E} \left(X_t^{(l)} \right) \\ &\quad - \sum_{m=1}^{n_t d} A_i^{(l,m)} \mathbb{E} \left(X_t^{(m)} X_t^{(k)} \right) - \sum_{m=1}^{n_t d} A_i^{(k,m)} \mathbb{E} \left(X_t^{(m)} X_t^{(l)} \right) \\ &\quad + \sum_{m=1}^{n_t d} \Gamma_i^{(l,m)}(t) \Gamma_i^{(k,m)}(t) \end{aligned}$$

In the same fashion, we get

$$\frac{d}{dt} \mathbb{E} \left(X_t^{(k)} \right) = a_i^{(k)}(t) - \sum_{m=1}^{n_t d} A_i^{(k,m)} \mathbb{E} \left(X_t^{(m)} \right) \quad (5a)$$

This leads to

$$\begin{aligned} \frac{d}{dt} \left(\mathbb{E} \left(X_t^{(k)} \right) \mathbb{E} \left(X_t^{(l)} \right) \right) &= \mathbb{E} \left(X_t^{(l)} \right) \frac{d}{dt} \mathbb{E} \left(X_t^{(k)} \right) + \mathbb{E} \left(X_t^{(k)} \right) \frac{d}{dt} \mathbb{E} \left(X_t^{(l)} \right) \\ &= a_i^{(k)}(t) \mathbb{E} \left(X_t^{(l)} \right) - \sum_{m=1}^{n_t d} A_i^{(k,m)} \mathbb{E} \left(X_t^{(m)} \right) \mathbb{E} \left(X_t^{(l)} \right) \\ &\quad + a_i^{(l)}(t) \mathbb{E} \left(X_t^{(k)} \right) - \sum_{m=1}^{n_t d} A_i^{(l,m)} \mathbb{E} \left(X_t^{(m)} \right) \mathbb{E} \left(X_t^{(k)} \right) \end{aligned}$$

Putting together these different parts gives us the ODE satisfied by all covariances :

$$\begin{aligned}
\frac{d}{dt} \text{Cov} \left(X_t^{(k)}, X_t^{(l)} \right) &= \frac{d}{dt} \left(\mathbb{E} \left(X_t^{(k)} X_t^{(l)} \right) - \mathbb{E}(X_t^{(k)}) \mathbb{E}(X_t^{(l)}) \right) \\
&= - \sum_{m=1}^{n_t d} \left[A_i^{(k,m)} \text{Cov} \left(X_t^{(m)}, X_t^{(l)} \right) + A_i^{(l,m)} \text{Cov} \left(X_t^{(m)}, X_t^{(k)} \right) - \Gamma_i^{(l,m)}(t) \Gamma_i^{(k,m)}(t) \right]
\end{aligned} \tag{5b}$$

66 Note that in a vectorial formalism with the expectation vector m and covariance matrix
67 Σ , these sets of ODEs can be written equivalently as follows

$$\frac{dm_t}{dt} = a_i(t) - A_i m_t \tag{S2}$$

$$\frac{d\Sigma_t}{dt} = -A_i \Sigma_t - {}^{tr}\Sigma_t {}^{tr}A_i + \Gamma_i {}^{tr}\Gamma_i \tag{S3}$$

B DISTRIBUTION FOR SOME MODELS WITHOUT INTERACTIONS BETWEEN LINEAGES

B.1 Distribution of classic univariate models

We present in this section how previously known results of analytic tip distribution of univariate models fit in, and can be rediscovered with, our framework. Results are summarized in Table S1.

The scheme is identical for each model :

1. Reduce Equations (4a, 4b) or (5a, 5b) according to the model.
2. Look for an analytical solution at any time τ_i , by calculating manually the expectations and covariances at $\tau_1, \tau_2, \tau_3, \dots$
3. Prove by induction that the analytical solution holds at any time τ_i .

We call $t_{k,l}$ the time of the most recent common ancestor to lineages k and l , and $t_{k,k}$ the death time of lineage k , equal to T if it survives until present (see Fig. S1). We further note $1_{k \text{ alive}}(t)$ the quantity that equals one if lineage k is alive at time t and zero otherwise, and $1_{k=l}$ that equals one if $k = l$ and zero otherwise. Last, $t_1 \wedge t_2$ stands for the minimum of the two values t_1 and t_2 .

The unity vector (vector full of 1) is denoted by V , I refers to the identity matrix (diagonal matrix with diagonal values equal to 1), and U refers to the unity matrix (matrix full of 1). Their size is the same as the size of the vector of traits X_t considered. Considering non-ultrametric trees including fossils amounts to replacing vector V and matrices I and U by their homologs V_{alive} , I_{alive} and U_{alive} , where the subscript specifies that the vector and matrices have 0 on lines and columns corresponding to lineages that are extinct in the given epoch.

B.1.1 Brownian Motion (BM)

We show how to get the well-known expression of the distribution of a trait evolving under BM, on non-necessarily ultrametric trees. We take $a = bV_{\text{alive}}$, $A = 0$ and $\Gamma = \sigma I_{\text{alive}}$, i.e. the process

Code	m_0	Σ_0	$(m_T)^{(k)}$	$(\Sigma_T)^{(k,l)}$
BM	m_0	v_0	$m_0 + bt_{k,k}$	$v_0 + \sigma^2 t_{k,l}$
OU	θ	0	θ	$\frac{\sigma^2}{2\psi} e^{-\psi(t_{k,k}+t_{l,l}-2t_{k,l})} (1 - e^{-2\psi t_{k,l}})$
OU	θ	$\frac{\sigma^2}{2\psi}$	θ	$\frac{\sigma^2}{2\psi} e^{-\psi(t_{k,k}+t_{l,l}-2t_{k,l})}$
ACDC	m_0	v_0	m_0	$v_0 + \frac{\sigma_0^2}{2r} (e^{2rt_{k,l}} - 1)$
DD	m_0	v_0	m_0	$v_0 + \sigma_0^2 \sum_{j=0}^{N-1} e^{2rn\tau_j} (\tau_{j+1} - \tau_j) 1_{t_{k,l} > \tau_j}$

TABLE S1: Analytic tip distribution for models without interactions between traits or lineages. We recall that $t_{k,l}$ is the absolute time of the most recent common ancestor to lineages k and l , and $t_{k,k}$ is the death time of lineage k , equal to T if it survives until present.

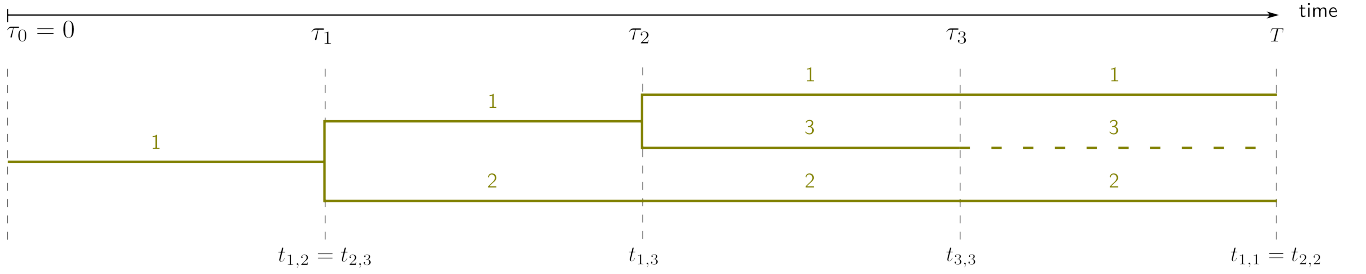


FIGURE S1: Formalism used in analytic formulae presented in Table S1.

92 follows the equation :

$$dX_t = bV_{\text{alive}}dt + \sigma I_{\text{alive}}dW_t$$

93 Equations (4a) and (4b) lead to the following recurrence formulae driving the law of X_t
94 through each epoch $[\tau_i, \tau_{i+1})$:

$$\mathbb{E}(X_t) = \mathbb{E}(X_{\tau_i}) + b(t - \tau_i)V_{\text{alive}}$$

$$\text{Var}(X_t) = \text{Var}(X_{\tau_i}) + \sigma^2(t - \tau_i)I_{\text{alive}}$$

95 Alternatively, Equations (5a) and (5b) lead to the following recurrence formulae driving
96 the law of X_t through each epoch $[\tau_i, \tau_{i+1})$:

$$\begin{aligned}\frac{d}{dt}\mathbb{E}(X_t^{(k)}) &= b1_{\text{k alive}}(t) \\ \frac{d}{dt}\text{Cov}(X_t^{(k)}, X_t^{(l)}) &= \sigma^2 1_{k=l} 1_{\text{k alive}}(t)\end{aligned}$$

97 We can show by induction on i that for any i the expectation and covariance matrix at
98 time τ_i are such that, for any (k, l) :

$$\mathbb{E}(X_{\tau_i}^{(k)}) = \mathbb{E}(X_0) + b(t_{k,k} \wedge \tau_i) \quad (\text{S4})$$

$$\text{Cov}(X_{\tau_i}^{(k)}, X_{\tau_i}^{(l)}) = \text{Var}(X_0) + \sigma^2(t_{k,l} \wedge \tau_i) \quad (\text{S5})$$

99 Indeed, we verify Equations (S4, S5) at step $i = 1$.

100 Now, suppose Equations (S4, S5) hold at step n . Using either Equations (4a, 4b) or (5a,
101 5b), we get :

$$\begin{aligned}\mathbb{E}(X_{\tau_{n+1}^-}^{(k)}) &= \mathbb{E}(X_0) + b(t_{k,k} \wedge \tau_{n+1}) \\ \text{Cov}(X_{\tau_{n+1}^-}^{(k)}, X_{\tau_{n+1}^-}^{(l)}) &= \text{Var}(X_0) + \sigma^2(t_{k,l} \wedge \tau_{n+1})\end{aligned}$$

102 If τ_{n+1} is a death time of a lineage, Equations (S4, S5) are verified at step $n + 1$.

103 If τ_{n+1} is a branching time, we verify that the new lineage inherits the expectation and
104 covariances of its mother, as well as the same coalescence times with other lineages. It also
105 follows that Equations (S4, S5) are verified at step $n + 1$.

106 Finally, by induction, we get the tip distribution :

$$\begin{aligned}\mathbb{E}(X_T^{(k)}) &= \mathbb{E}(X_0) + bt_{k,k} \\ \text{Cov}(X_T^{(k)}, X_T^{(l)}) &= \text{Var}(X_0) + \sigma^2 t_{k,l}\end{aligned}$$

107 B.1.2 Ornstein-Uhlenbeck (OU)

108 We can get another well-known distribution for a trait evolving under an Ornstein-Uhlenbeck
109 process on a tree. We take $a = \psi\theta V_{\text{alive}}$, $A = \psi I_{\text{alive}}$ and $\Gamma = \sigma I_{\text{alive}}$, i.e. the process follows the
110 equation :

$$dX_t = (\psi\theta V_{\text{alive}} - \psi I_{\text{alive}} X_t)dt + \sigma I_{\text{alive}} dW_t$$

Expressions (4a) and (4b) simplify into the following recurrence formulae :

$$\begin{aligned}\mathbb{E}(X_t) &= e^{-\psi(t-\tau_i)I_{\text{alive}}} (\mathbb{E}(X_{\tau_i}) - \theta V_{\text{alive}}) + \theta V_{\text{alive}} \\ \text{Var}(X_t) &= e^{-2\psi(t-\tau_i)I_{\text{alive}}} \left(\text{Var}(X_{\tau_i}) - \frac{\sigma^2}{2\psi} I_{\text{alive}} \right) + \frac{\sigma^2}{2\psi} I_{\text{alive}}\end{aligned}$$

Alternatively, here again, one can prefer to apply Equations (5a) and (5b) :

$$\begin{aligned}\frac{d}{dt}\mathbb{E}(X_t^{(k)}) &= \psi 1_{k\text{ alive}}(t) \left(\theta - \mathbb{E}(X_t^{(k)}) \right) \\ \frac{d}{dt}\text{Cov}(X_t^{(k)}, X_t^{(l)}) &= -\psi(1_{k\text{ alive}}(t) + 1_{l\text{ alive}}(t))\text{Cov}(X_t^{(k)}, X_t^{(l)}) + \sigma^2 1_{k=l}\end{aligned}$$

We can show by induction that for any epoch i , the expectation and covariance matrix at time τ_i are such that, for all (k, l) :

$$\mathbb{E}(X_{\tau_i}^{(k)}) = \theta + e^{-\psi(t_{k,k}\wedge\tau_i)} (\mathbb{E}(X_0) - \theta) \quad (\text{S6})$$

$$\text{Cov}(X_{\tau_i}^{(k)}, X_{\tau_i}^{(l)}) = e^{-\psi(t_{k,k}\wedge\tau_i + t_{l,l}\wedge\tau_i - 2(t_{k,l}\wedge\tau_i))} \left[\frac{\sigma^2}{2\psi} + e^{-2\psi(t_{k,l}\wedge\tau_i)} \left(\text{Var}(X_0) - \frac{\sigma^2}{2\psi} \right) \right] \quad (\text{S7})$$

Indeed, we verify Equations (S6, S7) at step $i = 0$.

Now, suppose Equations (S6, S7) hold at step n . Using either Equations (4a, 4b) or (5a, 5b), we get :

$$\begin{aligned}\mathbb{E}(X_{\tau_{n+1}}^{(k)}) &= \theta + e^{-\psi(t_{k,k}\wedge\tau_{n+1})} (\mathbb{E}(X_0) - \theta) \\ \text{Cov}(X_{\tau_{n+1}}^{(k)}, X_{\tau_{n+1}}^{(l)}) &= e^{-\psi(t_{k,k}\wedge\tau_{n+1} + t_{l,l}\wedge\tau_{n+1} - 2(t_{k,l}\wedge\tau_{n+1}))} \left[\frac{\sigma^2}{2\psi} + e^{-2\psi(t_{k,l}\wedge\tau_{n+1})} \left(\text{Var}(X_0) - \frac{\sigma^2}{2\psi} \right) \right]\end{aligned}$$

If τ_{n+1} is a death time of a lineage, Equations (S6, S7) are verified at step $n + 1$.

If τ_{n+1} is a branching time, we verify that the new lineage inherits the expectation and

covariances of its mother, as well as the same coalescence times with other lineages. It also

follows that Equations (S6, S7) are verified at step $n + 1$.

Finally, by induction, we get the tip distribution :

$$\begin{aligned}\mathbb{E}(X_T^{(k)}) &= \theta + e^{-\psi t_{k,k}} (\mathbb{E}(X_0) - \theta) \\ \text{Cov}(X_T^{(k)}, X_T^{(l)}) &= e^{-\psi(t_{k,k}+t_{l,l}-2t_{k,l})} \left[\frac{\sigma^2}{2\psi} + e^{-2\psi t_{k,l}} \left(\text{Var}(X_0) - \frac{\sigma^2}{2\psi} \right) \right]\end{aligned}$$

Two classes of initial distributions are typically considered in the literature :

1. If we consider a process starting at $X_0 = \theta$ (i.e. with $\mathbb{E}(X_0) = \theta$ and $\text{Var}(X_0) = 0$), we get the following expectation vector m_T and covariance matrix Σ_T at the tips :

$$\begin{aligned}m_T &= {}^{tr}(\theta, \theta, \dots, \theta) \quad \text{and} \quad \Sigma_T = \frac{\sigma^2}{2\psi} \Upsilon_1 \\ \text{where } \Upsilon_1 &= [e^{-\psi(t_{k,k}+t_{l,l}-2t_{k,l})} (1 - e^{-2\psi t_{k,l}})]_{1 \leq k, l \leq K}\end{aligned}$$

2. When $\psi > 0$, if we consider a process starting under its stationary distribution (i.e. $\mathbb{E}(X_0) = \theta$ and $\text{Var}(X_0) = \frac{\sigma^2}{2\psi}$), it simplifies into the following expectation vector and covariance matrix :

$$\begin{aligned}m_T &= {}^{tr}(\theta, \theta, \dots, \theta) \quad \text{and} \quad \Sigma_T = \frac{\sigma^2}{2\psi} \Upsilon_2 \\ \text{where } \Upsilon_2 &= [e^{-\psi(t_{k,k}+t_{l,l}-2t_{k,l})}]_{1 \leq k, l \leq K}\end{aligned}$$

B.1.3 ACDC (accelerating or decelerating rate)

In the ACDC process, the rate of phenotypic evolution varies exponentially through time, with $a = 0$, $A = 0$ and $\Gamma = \sigma_0 e^{rt} I_{\text{alive}}$ (here, $r > 0$). The process follows the equation :

$$dX_t = \sigma_0 e^{rt} I_{\text{alive}} dW_t$$

Here again, we can simplify Equations (4a, 4b) or (5a, 5b). With Equations (4a, 4b), we get the following recurrence formulae driving the law of X_t through each epoch (τ_i, τ_{i+1}) :

$$\mathbb{E}(X_t) = \mathbb{E}(X_{\tau_i})$$

$$\text{Var}(X_t) = \text{Var}(X_{\tau_i}) + \frac{\sigma_0^2}{2r} (e^{2rt} - e^{2r\tau_i}) I_{\text{alive}} dt$$

134 We can show by induction that for any i , the expectation and covariance matrix at time τ_i
 135 are such that, for any (k, l) :

$$\mathbb{E}(X_{\tau_i}^{(k)}) = \mathbb{E}(X_0) \tag{S8}$$

$$\text{Cov}(X_{\tau_i}^{(k)}, X_{\tau_i}^{(l)}) = \text{Var}(X_0) + \frac{\sigma_0^2}{2r} (e^{2r(t_{k,l} \wedge \tau_i)} - 1) \tag{S9}$$

136 Indeed, we verify Equations (S8, S9) at step $i = 0$.

137 Now, suppose Equations (S8, S9) hold at step n . Using either Equations (4a, 4b) or (5a,
 138 5b), we get :

$$\mathbb{E}(X_{\tau_{n+1}}^{(k)}) = \mathbb{E}(X_0)$$

$$\text{Cov}(X_{\tau_{n+1}}^{(k)}, X_{\tau_{n+1}}^{(l)}) = \text{Var}(X_0) + \frac{\sigma_0^2}{2r} (e^{2r(t_{k,l} \wedge \tau_{n+1})} - 1)$$

139 If τ_{n+1} is a death time of a lineage, Equations (S8, S9) are verified at step $n + 1$.

140 If τ_{n+1} is a branching time, we verify that the new lineage inherits the expectation and
 141 covariances of its mother, as well as the same coalescence times with other lineages. It also
 142 follows that Equations (S8, S9) are verified at step $n + 1$.

143 Finally, by induction, we get the tip distribution :

$$\mathbb{E}(X_T^{(k)}) = \mathbb{E}(X_0)$$

$$\text{Cov}(X_T^{(k)}, X_T^{(l)}) = \text{Var}(X_0) + \frac{\sigma_0^2}{2r} (e^{2rt_{k,k}} - 1)$$

144 *B.1.4 ACDC and OU processes lead to the same present-time distributions on ultrametric trees*

145 This has been shown previously in Uyeda et al. 2015. More precisely, OU is equivalent to a
 146 model with accelerating rates at present, and only on ultrametric phylogenies.

Looking at expressions of expectations and covariance matrices under ACDC and OU

with initial conditions $X_0 = \theta$, we see that we can choose parameters such that we get the exact same distribution. First take $\mathbb{E}(X_0) = \theta$: the two expectation vectors are identical. Moreover, we can choose parameters such that the covariance matrices are equal :

$$\begin{aligned} \frac{\sigma^2}{2\psi} e^{-2\psi(T-t_{k,l})} (1 - e^{-2\psi t_{k,l}}) &= \frac{\sigma_0^2}{2r} (e^{2rt_{k,l}} - 1) \\ \iff \frac{\sigma^2}{2\psi} e^{-2\psi T} (e^{2\psi t_{k,l}} - 1) &= \frac{\sigma_0^2}{2r} (e^{2rt_{k,l}} - 1) \\ \iff r = \psi \quad \text{and} \quad \sigma_0^2 = \sigma^2 e^{-2\psi T} \end{aligned}$$

Note that this no longer holds on non-ultrametric trees, neither with different initial conditions on the OU.

B.1.5 Diversity-Dependent (DD)

In the DD process, the rate of phenotypic evolution is fixed at the base of the tree and varies exponentially with the number of lineages in the reconstructed phylogeny, with $a = 0$, $A = 0$ and $B(t) = \sigma_0 e^{rnt} I_{\text{alive}}$. The process follows the equation :

$$dX_t = \sigma_0 e^{rnt} I_{\text{alive}} dW_t$$

Equations (4a, 4b) lead to the following recurrence formulae driving the law of X_t through each epoch (τ_i, τ_{i+1}) :

$$\begin{aligned} \mathbb{E}(X_t) &= \mathbb{E}(X_{\tau_i}) \\ \text{Var}(X_t) &= \text{Var}(X_{\tau_i}) + \sigma_0^2 e^{2rn\tau_i} (t - \tau_i) I_{\text{alive}} \end{aligned}$$

Note that, alternatively, one can again prefer to apply Equations (5a, 5b).

We can then show by induction that for any i , the expectation and covariance matrix at time τ_i are such that, for any (k, l) :

$$\mathbb{E}(X_{\tau_i}^{(k)}) = \mathbb{E}(X_0) \quad (\text{S10})$$

$$\text{Cov}(X_{\tau_i}^{(k)}, X_{\tau_i}^{(l)}) = \text{Var}(X_0) + \sigma_0^2 \sum_{j=0}^{i-1} e^{2rn\tau_j} (\tau_{j+1} - \tau_j) 1_{t_{k,l} > \tau_j} \quad (\text{S11})$$

Indeed, we verify Equations (S10, S11) at step $i = 0$.

Now, suppose Equations (S10, S11) hold at step n . Using either Equations (4a, 4b) or (5a, 5b), we get :

$$\begin{aligned} \mathbb{E}(X_{\tau_{n+1}^-}^{(k)}) &= \mathbb{E}(X_0) \\ \text{Cov}(X_{\tau_{n+1}^-}^{(k)}, X_{\tau_{n+1}^-}^{(l)}) &= \text{Var}(X_0) + \sigma_0^2 \sum_{j=0}^n e^{2rn\tau_j} (\tau_{j+1} - \tau_j) 1_{t_{k,l} > \tau_j} \end{aligned}$$

If τ_{n+1} is a death time of a lineage, Equations (S10, S11) are verified at step $n + 1$.

If τ_{n+1} is a branching time, we verify that the new lineage inherits the expectation and

covariances of its mother, as well as the same coalescence times with other lineages. It also

follows that Equations (S10, S11) are verified at step $n + 1$.

Finally, by induction, we get the tip distribution at present time $\tau_N = T$:

$$\begin{aligned} \mathbb{E}(X_T^{(k)}) &= \mathbb{E}(X_0) \\ \text{Cov}(X_T^{(k)}, X_T^{(l)}) &= \text{Var}(X_0) + \sigma_0^2 \sum_{j=0}^{N-1} e^{2rn\tau_j} (\tau_{j+1} - \tau_j) 1_{t_{k,l} > \tau_j} \end{aligned}$$

B.2 Distribution of classic multivariate models

The same methodology applies to classic multivariate models that incorporate interactions

between traits within lineages but not between lineages. In our formalism, for all i , A_i and Γ_i are

block diagonal, with $d \times d$ blocks on the diagonal corresponding to the traits within each lineage.

We call these blocks respectively A^* and Γ^* . Moreover, the vector a_i is the repetition of identical

sequences a^* of d elements.

Writing the matrix products in Equations (4a, 4b) provides us with $d \times d$ blocks that

behave identically during each epoch. Indeed, we can use :

$$m_{\tau_i}^{*(k)} = \begin{pmatrix} \mathbb{E}(X_{\tau_i}^{(k,1)}) \\ \mathbb{E}(X_{\tau_i}^{(k,2)}) \\ \vdots \\ \mathbb{E}(X_{\tau_i}^{(k,d)}) \end{pmatrix} \text{ and } \Sigma_{\tau_i}^{*(k,l)} = \begin{pmatrix} \text{Cov}(X_{\tau_i}^{(k,1)}, X_{\tau_i}^{(l,1)}) & \text{Cov}(X_{\tau_i}^{(k,1)}, X_{\tau_i}^{(l,2)}) & \dots & \text{Cov}(X_{\tau_i}^{(k,1)}, X_{\tau_i}^{(l,d)}) \\ \text{Cov}(X_{\tau_i}^{(k,2)}, X_{\tau_i}^{(l,1)}) & \text{Cov}(X_{\tau_i}^{(k,2)}, X_{\tau_i}^{(l,2)}) & \dots & \text{Cov}(X_{\tau_i}^{(k,2)}, X_{\tau_i}^{(l,d)}) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(X_{\tau_i}^{(k,d)}, X_{\tau_i}^{(l,1)}) & \text{Cov}(X_{\tau_i}^{(k,d)}, X_{\tau_i}^{(l,2)}) & \dots & \text{Cov}(X_{\tau_i}^{(k,d)}, X_{\tau_i}^{(l,d)}) \end{pmatrix}$$

178 In which case Equations (4a, 4b) lead to the recurrence formulae :

$$\begin{aligned} m_{\tau_{i+1}}^{*(k)} &= e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_{i+1})A^*} m_{\tau_i}^{*(k)} + 1_k \text{ alive}(\tau_{i+1}) \int_{\tau_i}^{\tau_{i+1}} e^{(s - \tau_{i+1})A^*} a^*(s) ds \\ \Sigma_{\tau_{i+1}}^{*(k,l)} &= e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_{i+1})A^*} \Sigma_{\tau_i}^{*(k,l)} \text{tr} \left(e^{(\tau_i - \tau_{i+1})1_l \text{ alive}(\tau_{i+1})A^*} \right) \\ &\quad + 1_{k=l} \int_{\tau_i}^{\tau_{i+1}} \left(e^{(s - \tau_{i+1})A^*} \Gamma^* \right) \text{tr} \left(e^{(s - \tau_{i+1})A^*} \Gamma^* \right) ds \end{aligned}$$

179 We can then prove by induction that for any epoch i and any pair of lineages (k, l)

$$m_{\tau_i}^{*(k)} = e^{-\tau_i \wedge t_{k,k} A^*} m_0^* + \int_0^{\tau_i \wedge t_{k,k}} e^{(s - \tau_i \wedge t_{k,k})A^*} a^*(s) ds \quad (\text{S12})$$

$$\Sigma_{\tau_i}^{*(k,l)} = e^{-\tau_i \wedge t_{k,k} A^*} \Sigma_0^* \text{tr} \left(e^{-\tau_i \wedge t_{l,l} A^*} \right) + \int_0^{\tau_i \wedge t_{k,k}} \left(e^{-\tau_i \wedge t_{k,k} A^*} \Gamma^* \right) \text{tr} \left(e^{-\tau_i \wedge t_{l,l} A^*} \Gamma^* \right) ds \quad (\text{S13})$$

180 Indeed, we verify Equations (S12, S13) at step $i = 0$.

181 Now, suppose Equations (S12, S13) hold at step i . Using Equations (4a, 4b), we get :

$$\begin{aligned} m_{\tau_{i+1}}^{*(k)} &= e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_i)A^*} m_{\tau_i}^{*(k)} + 1_k \text{ alive}(\tau_i) \int_{\tau_i}^{\tau_{i+1}} e^{(s - \tau_{i+1})A^*} a^*(s) ds \\ &= e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_i)A^*} e^{-\tau_i \wedge t_{k,k} A^*} m_0^* + \int_0^{\tau_i \wedge t_{k,k}} e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_i)A^*} e^{(s - \tau_i \wedge t_{k,k})A^*} a^*(s) ds \\ &\quad + 1_k \text{ alive}(\tau_i) \int_{\tau_i}^{\tau_{i+1}} e^{(s - \tau_{i+1})A^*} a^*(s) ds \\ &= e^{-\tau_{i+1} \wedge t_{k,k} A^*} m_0^* + \int_0^{\tau_{i+1} \wedge t_{k,k}} e^{(s - \tau_{i+1} \wedge t_{k,k})A^*} a^*(s) ds \end{aligned}$$

182 as well as :

$$\begin{aligned}
\Sigma_{\tau_{i+1}}^{*(k,l)} &= e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_{i+1})A^*} \Sigma_{\tau_i}^{*(k,l)} \text{tr} \left(e^{(\tau_i - \tau_{i+1})1_l \text{ alive}(\tau_{i+1})A^*} \right) \\
&\quad + 1_{k=l} \int_{\tau_i}^{\tau_{i+1}} \left(e^{(s - \tau_{i+1})A^*} \Gamma^* \right) \text{tr} \left(e^{(s - \tau_{i+1})A^*} \Gamma^* \right) ds \\
&= e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_{i+1})A^*} e^{-\tau_i \wedge t_{k,k} A^*} \Sigma_0^{*tr} \left(e^{-\tau_i \wedge t_{l,l} A^*} \right) \text{tr} \left(e^{(\tau_i - \tau_{i+1})1_l \text{ alive}(\tau_{i+1})A^*} \right) \\
&\quad + \int_0^{t_{k,l} \wedge \tau_i} e^{(\tau_i - \tau_{i+1})1_k \text{ alive}(\tau_{i+1})A^*} \left(e^{-\tau_i \wedge t_{k,k} A^*} \Gamma^* \right) \text{tr} \left(e^{-\tau_i \wedge t_{l,l} A^*} \Gamma^* \right) \text{tr} \left(e^{(\tau_i - \tau_{i+1})1_l \text{ alive}(\tau_{i+1})A^*} \right) ds \\
&\quad + 1_{k=l} \int_{\tau_i}^{\tau_{i+1}} \left(e^{(s - \tau_{i+1})A^*} \Gamma^* \right) \text{tr} \left(e^{(s - \tau_{i+1})A^*} \Gamma^* \right) ds \\
&= e^{-\tau_{i+1} \wedge t_{k,k} A^*} \Sigma_0^{*tr} \left(e^{-\tau_{i+1} \wedge t_{l,l} A^*} \right) + \int_0^{t_{k,l} \wedge \tau_{i+1}} \left(e^{-\tau_{i+1} \wedge t_{k,k} A^*} \Gamma^* \right) \text{tr} \left(e^{-\tau_{i+1} \wedge t_{l,l} A^*} \Gamma^* \right) ds
\end{aligned}$$

If τ_{i+1} is a death time of a lineage, Equations (S12, S13) are verified at step $i + 1$.

If τ_{i+1} is a branching time, we verify that the new lineage inherits the expectation and covariances of its mother, as well as the same coalescence times with other lineages. It also follows that Equations (S12, S13) are verified at step $i + 1$.

Finally, by induction, we get the tip distribution :

$$\begin{aligned}
m_T^{*(k)} &= e^{-t_{k,k} A^*} m_0^* + \int_0^{t_{k,k}} e^{(s - t_{k,k}) A^*} a^*(s) ds \\
\Sigma_T^{*(k,l)} &= e^{-t_{k,k} A^*} \Sigma_0^{*tr} \left(e^{-t_{l,l} A^*} \right) + \int_0^{t_{k,l}} \left(e^{-t_{k,k} A^*} \Gamma^* \right) \text{tr} \left(e^{-t_{l,l} A^*} \Gamma^* \right) ds
\end{aligned}$$

B.2.1 OU-BM model

As a first illustration, consider a model with $d = 3$ traits with equation during each epoch and on each lineage k as follows :

$$\begin{aligned}
dX_t^{(k,1)} &= \psi \left(b_1 + b_2 X_t^{(k,2)} + b_3 X_t^{(k,3)} - X_t^{(k,1)} \right) dt + \sigma_1 dW_t^{(k,1)} \\
dX_t^{(k,2)} &= \sigma_2 dW_t^{(k,2)} \\
dX_t^{(k,3)} &= \sigma_3 dW_t^{(k,3)}
\end{aligned}$$

These equations describe the evolution of two independent traits evolving following a BM (traits 2 and 3), and one trait following an OU with optimal trait value given by a linear

193 combination of traits 2 and 3. Its main interest is to infer the dependence of one trait to two
 194 other independent traits on a phylogeny. Knowing the distribution at the beginning of a given
 195 epoch, we use Equations (4a, 4b) to compute the distribution at the end of the epoch.

196 A is block-diagonal with the following blocks A^* :

$$A^* = \begin{pmatrix} 1 & -b_2 & -b_3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

197 Writing $\Delta = s - \tau_{i+1}$, it follows that $e^{\Delta A_i}$ is block diagonal with 3×3 elements given by :

$$e^{\Delta A^*} = \begin{pmatrix} e^\Delta & -b_2(e^\Delta - 1) & -b_3(e^\Delta - 1) \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

198 Moreover, Γ_i is block-diagonal with diagonal blocks :

$$\Gamma^* = \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{pmatrix}$$

199 The matrix product $(e^{\Delta A_i} \Gamma_i)^{tr} (e^{\Delta A_i} \Gamma_i)$ is thus block-diagonal with 3×3 blocks :

$$\begin{pmatrix} (\sigma_1^2 + b_2^2 \sigma_2^2 + b_3^2 \sigma_3^2) e^{2\Delta} - 2(b_2^2 \sigma_2^2 + b_3^2 \sigma_3^2) e^\Delta + (b_2^2 \sigma_2^2 + b_3^2 \sigma_3^2) & -b_2 \sigma_2^2 (e^\Delta - 1) & -b_3 \sigma_3^2 (e^\Delta - 1) \\ -b_2 \sigma_2^2 (e^\Delta - 1) & \sigma_2^2 & 0 \\ -b_3 \sigma_3^2 (e^\Delta - 1) & 0 & \sigma_3^2 \end{pmatrix}$$

200 These matrices can be used to compute $m_T^{*(k)}$ and $\Sigma_T^{*(k,l)}$, with the help of Equations (S12,
 201 S13).

202 B.2.2 OU-OU model

203 Consider now a model with $d = 2$ traits with equation during each epoch and on each lineage k
 204 given by :

$$\begin{aligned}
dX_t^{(k,1)} &= \psi \left(b_1 + b_2 X_t^{(k,2)} - X_t^{(k,1)} \right) dt + \sigma_1 dW_t^{(k,1)} \\
dX_t^{(k,2)} &= \psi \left(b_3 - X_t^{(k,2)} \right) dt + \sigma_2 dW_t^{(k,2)}
\end{aligned}$$

205 These equations describe the evolution of one trait evolving following an OU (trait 2), and
 206 one trait following an OU with optimal trait value given by an affine transformation of trait 2.
 207 Its main interest is to infer the dependence of one trait to another trait on a phylogeny. Knowing
 208 the distribution at the beginning of a given epoch, we use Equations (4a, 4b) to compute the
 209 distribution at the end of the epoch.

210 A_i is block diagonal, with the following 2×2 blocks A^* :

$$A^* = \begin{pmatrix} 1 & -b_2 \\ 0 & 1 \end{pmatrix}$$

211 Again, writing $\Delta = s - \tau_{i+1}$, it follows that $e^{\Delta A_i}$ is block diagonal with 2×2 elements
 212 given by :

$$e^{\Delta A^*} = \begin{pmatrix} e^\Delta & -b_2 \Delta e^\Delta \\ 0 & e^\Delta \end{pmatrix}$$

213 Moreover, Γ_i is diagonal with repeated values :

$$\Gamma^* = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}$$

214 The matrix product $(e^{\Delta A_i} \Gamma_i)^{tr} (e^{\Delta A_i} \Gamma_i)$ is thus block-diagonal with 2×2 blocks :

$$\begin{pmatrix} \sigma_1^2 e^{2\Delta} + b_2^2 \Delta^2 \sigma_2^2 e^{2\Delta} & -b_2 \sigma_2^2 \Delta e^{2\Delta} \\ -b_2 \sigma_2^2 \Delta e^{2\Delta} & \sigma_2^2 e^{2\Delta} \end{pmatrix}$$

215 These matrices can be used to compute $m_T^{*(k)}$ and $\Sigma_T^{*(k,l)}$, with the help of Equations (S12,
 216 S13).

C DISTRIBUTION FOR SOME MODELS WITH INTERACTIONS BETWEEN LINEAGES

C.1 Distribution with a constant, A symmetric, and $\Gamma = \sigma I$

When $\Gamma = \sigma I$ and A is symmetric, Equations (4a, 4b) become :

$$\begin{aligned}\mathbb{E}(X_t) &= e^{(\tau_i-t)A_i} \mathbb{E}(X_{\tau_i}) + \int_{\tau_i}^t e^{(s-t)A_i} a_i(s) ds \\ \text{Var}(X_t) &= \left(e^{(\tau_i-t)A_i} \right) \text{Var}(X_{\tau_i})^{tr} \left(e^{(\tau_i-t)A_i} \right) + \sigma^2 \int_{\tau_i}^t e^{2(s-t)A_i} ds\end{aligned}$$

If A_i is symmetric with coefficients in \mathbb{R} , it can be diagonalized by orthogonal passage matrices : we can exhibit a matrix Q verifying ${}^{tr}QA_iQ = \Lambda_i$ is diagonal and $Q^{-1} = {}^{tr}Q$.

$$\begin{aligned}\mathbb{E}(X_t) &= Q e^{(\tau_i-t)\Lambda_i} {}^{tr}Q \mathbb{E}(X_{\tau_i}) + Q \left(\int_{\tau_i}^t e^{(s-t)\Lambda_i} ds \right) {}^{tr}Q a_i \\ \text{Var}(X_t) &= Q e^{\Lambda_i(\tau_i-t)} {}^{tr}Q \text{Var}(X_{\tau_i}) Q e^{(\tau_i-t)\Lambda_i} {}^{tr}Q + \sigma^2 Q \left(\int_{\tau_i}^t e^{2(s-t)\Lambda_i} ds \right) {}^{tr}Q\end{aligned}$$

This is the expression that we need for the numerical integration, in particular, of the phenotype matching model.

Note that with A diagonalizable but not symmetric, Equations (4a, 4b) can also be reduced, but the transposition of A is no longer A , and it does not lead exactly to the same expression.

C.2 The phenotype matching (PM) model

We consider here the phenotype matching model introduced in Nuismer and Harmon (2014), with the following equation describing the evolution of any trait k through each epoch :

$$dX_t^{(k)} = \psi \left(\theta - X_t^{(k)} \right) dt + S \left(\left(\frac{1}{n_t} \sum_{l=1}^{n_t} X_t^{(l)} \right) - X_t^{(k)} \right) dt + \sigma dW_t^{(k)}$$

231 We introduce the line vector u , with value u_j that equals 1 if lineage j is alive, and 0
 232 otherwise. In order to use our framework, we further want to express the model in the form given
 233 by Equation (2). This is achieved by taking :

$$\begin{aligned} a_i &= \psi \theta^{tr} u \\ A_i &= (\psi + S) \text{diag}(u) - \frac{S}{u^{tr} u} {}^{tr} u u \\ \Gamma_i &= \sigma \text{diag}(u) \end{aligned}$$

234 where $\text{diag}(u)$ is the diagonal matrix with diagonal elements the elements of the vector u .

235 First, the tip distribution can be computed using the general algorithm that numerically
 236 resolves the set of ODEs given in Equations (5a, 5b). Second, the PM model falls within the
 237 class of models studied in the previous section, that is, with a symmetric A matrix. The tip
 238 distribution can thus be numerically computed faster using this reduction.

239 We describe here a third (and faster) way to derive the tip distribution. It is based on an
 240 analytical reduction of Equations (4a, 4b) that is specific to the PM model.

241 Remark that $\text{diag}(u)$ and ${}^{tr} u u$ commute, leading to the following calculus,

$$\begin{aligned} e^{(\tau_i - \tau_{i+1}) A_i} &= e^{(\tau_i - \tau_{i+1}) ((\psi + S) \text{diag}(u) - \frac{S}{u^{tr} u} {}^{tr} u u)} \\ &= e^{(\tau_i - \tau_{i+1}) (\psi + S) \text{diag}(u)} e^{-(\tau_i - \tau_{i+1}) \frac{S}{u^{tr} u} {}^{tr} u u} \\ &= \text{diag} \left(e^{(\tau_i - \tau_{i+1}) (\psi + S) u} \right) \left(\sum_{k \geq 0} \frac{\left(\frac{-(\tau_i - \tau_{i+1}) S}{u^{tr} u} \right)^k ({}^{tr} u u)^k}{k!} \right) \end{aligned}$$

242 Where e^w is the line vector with elements e^{w_j} . Further, remark that for any $k \geq 1$,

$$\begin{aligned} ({}^{tr} u u)^k &= ({}^{tr} u u) ({}^{tr} u u) ({}^{tr} u u) \dots ({}^{tr} u u) \\ &= {}^{tr} u (u {}^{tr} u) (u {}^{tr} u) \dots (u {}^{tr} u) u \\ &= (u {}^{tr} u)^{k-1} ({}^{tr} u u) \end{aligned}$$

243 For simplicity, we will write in the following $\Delta = \tau_i - \tau_{i+1}$, leading us to

$$\begin{aligned}
e^{\Delta A_i} &= \text{diag} \left(e^{(\psi+S)\Delta u} \right) \left(I + \sum_{k \geq 1} \frac{\left(\frac{-S\Delta}{u^{tr u}} \right)^k (u^{tr u})^{k-1} (tr uu)}{k!} \right) \\
&= \text{diag} \left(e^{(\psi+S)\Delta u} \right) \left(I + \frac{1}{u^{tr u}} \left(\sum_{k \geq 1} \frac{(-(\tau_i - \tau_{i+1})S)^k}{k!} \right) tr uu \right) \\
&= \text{diag} \left(e^{(\psi+S)\Delta u} \right) \left(I + \frac{1}{u^{tr u}} (e^{-S\Delta} - 1) tr uu \right) \\
&= \text{diag} \left(e^{(\psi+S)\Delta u} \right) + \frac{1}{u^{tr u}} \text{diag} \left(e^{-S\Delta} e^{(\psi+S)\Delta u} \right) tr uu - \frac{1}{u^{tr u}} \text{diag} \left(e^{(\psi+S)\Delta u} \right) tr uu \\
&= \text{diag} \left(e^{(\psi+S)\Delta u} \right) + \frac{1}{u^{tr u}} (e^{\psi\Delta} - e^{(\psi+S)\Delta}) tr uu
\end{aligned} \tag{S14}$$

244 Where the last equality is due to the product by $tr u$, allowing to forget the cases where
245 $u_j = 0$ in the exponential.

246 We further need to compute

$$\begin{aligned}
\int_{\tau_i}^{\tau_{i+1}} e^{(s-\tau_{i+1})A_i} a_i ds &= \psi \theta \int_{\tau_i}^{\tau_{i+1}} e^{\psi(s-\tau_{i+1})} ds tr u \\
&= \theta (1 - e^{\psi\Delta}) tr u
\end{aligned} \tag{S15}$$

247 We thus get $m_{\tau_{i+1}^-}$ with the help of Equations (S14) and (S15).

248 Now, in order to simplify Equation (4b), remark that A_i and Γ_i are symmetric, and so are
249 $e^{\Delta A_i}$ and $e^{\Delta A_i} \Gamma_i$. Moreover, Γ_i is diagonal, and commutes with any other matrix, leading to,

$$\Sigma_{\tau_{i+1}^-} = e^{\Delta A_i} \Sigma_{\tau_i} e^{\Delta A_i} + \int_{\tau_i}^{\tau_{i+1}} e^{2(s-\tau_{i+1})A_i} \Gamma_i \Gamma_i ds$$

250 The first term can be computed thanks to Equation (S14). For the second one, remark
251 that $tr uu \text{diag}(u) = tr uu$, thus leading to

$$\begin{aligned}
\int_{\tau_i}^{\tau_{i+1}} e^{2(s-\tau_{i+1})A_i} \Gamma_i \Gamma_i ds &= \sigma^2 \int_{\tau_i}^{\tau_{i+1}} e^{2(\psi+S)(s-\tau_{i+1})} ds \text{diag}(u) \\
&\quad + \frac{\sigma^2}{u^{tr u}} \int_{\tau_i}^{\tau_{i+1}} (e^{2\psi(s-\tau_{i+1})} - e^{2(\psi+S)(s-\tau_{i+1})}) ds tr uu \text{diag}(u) \\
&= \sigma^2 \frac{(1 - e^{2(\psi+S)\Delta})}{2(\psi + S)} \text{diag}(u) + \frac{\sigma^2}{u^{tr u}} \left(\frac{1 - e^{2\psi\Delta}}{2\psi} - \frac{1 - e^{2(\psi+S)\Delta}}{2(\psi + S)} \right) tr uu
\end{aligned} \tag{S16}$$

We thus get $\Sigma_{\tau_{i+1}^-}$ with the help of Equations (S14) and (S16).

C.3 The phenotype matching (PM) model with biogeography

In this section we describe ways to compute the tip distribution under the PM model, taking into account the biogeography (that is, species interact only when they co-occur in the same localities). We consider a fixed number of islands N_I . Matrix U gives us the presence/absence of lineages in the distinct islands, with element u_{ij} that equals 1 if lineage j is present on island i and zero otherwise. Vector S gives the strength of interaction on each island. The model states that the trait of lineage j evolves through phenotype matching with all species that are sympatric :

$$dX_t^{(j)} = \psi \left(\theta - X_t^{(j)} \right) dt + \sum_{i=1}^{N_I} S_i u_{ij} \left(\frac{\sum_{l=1}^n u_{il} X_t^{(l)}}{\sum_{l=1}^n u_{il}} - X_t^{(j)} \right) dt + \sigma dW_t^{(j)}$$

Take for example 5 lineages evolving on 3 distinct islands with the following U matrix on a given epoch :

$$U = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

This means that species number 1 is present on island 2 and 3, species number 2 is only present on island 1, and so on... Said differently, we see that species number 3 interacts on island 1 with species 2, and on island 2 with species 1 and 4. Our species traits are driven by the following equations :

$$\begin{aligned}
dX_t^{(1)} &= \left(\psi \left(\theta - X_t^{(1)} \right) + S_2 \left(\frac{X_t^{(1)} + X_t^{(3)} + X_t^{(4)}}{3} - X_t^{(1)} \right) + S_3 \left(\frac{X_t^{(1)} + X_t^{(5)}}{2} - X_t^{(1)} \right) \right) dt + \sigma dW_t^{(1)} \\
dX_t^{(2)} &= \left(\psi \left(\theta - X_t^{(2)} \right) + S_1 \left(\frac{X_t^{(2)} + X_t^{(3)}}{2} - X_t^{(2)} \right) \right) dt + \sigma dW_t^{(2)} \\
dX_t^{(3)} &= \left(\psi \left(\theta - X_t^{(3)} \right) + S_1 \left(\frac{X_t^{(2)} + X_t^{(3)}}{2} - X_t^{(3)} \right) + S_2 \left(\frac{X_t^{(1)} + X_t^{(3)} + X_t^{(4)}}{3} - X_t^{(3)} \right) \right) dt + \sigma dW_t^{(3)} \\
dX_t^{(4)} &= \left(\psi \left(\theta - X_t^{(4)} \right) + S_2 \left(\frac{X_t^{(1)} + X_t^{(3)} + X_t^{(4)}}{3} - X_t^{(4)} \right) \right) dt + \sigma dW_t^{(4)} \\
dX_t^{(5)} &= \left(\psi \left(\theta - X_t^{(5)} \right) + S_3 \left(\frac{X_t^{(1)} + X_t^{(5)}}{2} - X_t^{(5)} \right) \right) dt + \sigma dW_t^{(5)}
\end{aligned}$$

It thus follows that the vectorial equation can be written :

$$dX_t = \begin{pmatrix} \psi\theta \\ \psi\theta \\ \psi\theta \\ \psi\theta \\ \psi\theta \end{pmatrix} - \begin{pmatrix} \psi + \frac{2}{3}S_2 + \frac{1}{2}S_3 & 0 & -\frac{S_2}{3} & -\frac{S_2}{3} & -\frac{S_3}{2} \\ 0 & \psi + \frac{1}{2}S_1 & -\frac{S_1}{2} & 0 & 0 \\ -\frac{S_2}{3} & -\frac{S_1}{2} & \psi + \frac{1}{2}S_1 + \frac{2}{3}S_2 & -\frac{S_2}{3} & 0 \\ -\frac{S_2}{3} & 0 & -\frac{S_2}{3} & \psi + \frac{2}{3}S_2 & 0 \\ -\frac{S_1}{2} & 0 & 0 & 0 & \psi + \frac{1}{2}S_1 \end{pmatrix} X_t dt + \sigma dW_t$$

Provided no island is empty, the model can be written in our framework with $a = \psi\theta V$,

$\Gamma = \sigma I$, and, finally, A which is the matrix with elements :

$$\begin{aligned}
(A)_{jj} &= \psi + \sum_{i=1}^{N_I} S_i u_{ij} \left(1 - \frac{1}{\sum_{l=1}^n u_{il}} \right) \\
(A)_{jk} &= - \sum_{i=1}^{N_I} S_i u_{ij} u_{ik} \frac{1}{\sum_{l=1}^n u_{il}}
\end{aligned}$$

Matrix A is symmetric, and we can thus use the developments presented in Appendix C.1 to speed up the computation time.

Nonetheless, a better analytical reduction can be derived when islands are exclusive, meaning that species are allowed to occur on one island only. Under this assumption, matrix $U^T U$ is diagonal with element $(U^T U)_{ii}$ being the number of lineages belonging to island i . We

277 now introduce the line vector r , of size N_I , full of ones. For simplicity, we also write in the
 278 following $\Delta = \tau_i - \tau_{i+1}$. With these notations, and provided no island is empty, the model can be
 279 written under our framework with :

$$\begin{aligned} a_i &= \psi \theta^T (rU) \\ A_i &= \text{diag}((\psi r + S)U) - {}^T U \text{diag}(S)(U^T U)^{-1} U \\ \Gamma_i &= \sigma \text{diag}(rU) \end{aligned}$$

280 As for the one island case, we can speed up the computation of the exponential by
 281 remarking that :

$$\begin{aligned} e^{\Delta A_i} &= e^{\Delta \text{diag}((\psi r + S)U)} e^{-\Delta {}^T U \text{diag}(S)(U^T U)^{-1} U} \\ &= e^{\Delta \text{diag}((\psi r + S)U)} \sum_{k \geq 0} \frac{(-\Delta {}^T U \text{diag}(S)(U^T U)^{-1} U)^k}{k!} \end{aligned}$$

282 We then observe that :

$$\begin{aligned} &(-\Delta {}^T U \text{diag}(S)(U^T U)^{-1} U)^k \\ &= (-\Delta {}^T U \text{diag}(S)(U^T U)^{-1} U)(-\Delta {}^T U \text{diag}(S)(U^T U)^{-1} U) \dots (-\Delta {}^T U \text{diag}(S)(U^T U)^{-1} U) \\ &= {}^T U (-\Delta \text{diag}(S))(U^T U)^{-1} (U^T U) (-\Delta \text{diag}(S))(U^T U)^{-1} (U^T U) \dots (U^T U) (-\Delta \text{diag}(S))(U^T U)^{-1} U \\ &= {}^T U (-\Delta \text{diag}(S))^k (U^T U)^{-1} U \end{aligned}$$

283 Thus leading to the following expression :

$$\begin{aligned}
e^{\Delta A_i} &= e^{\Delta \text{diag}((\psi r + S)U)} \left(I + \sum_{k \geq 1} \frac{(-\Delta^T U \text{diag}(S)(U^T U)^{-1} U)^k}{k!} \right) \\
&= \text{diag}(e^{\Delta(\psi r + S)U}) \left(I + {}^T U \left(\sum_{k \geq 1} \frac{(-\Delta \text{diag}(S))^k}{k!} \right) (U^T U)^{-1} U \right) \\
&= \text{diag}(e^{\Delta(\psi r + S)U}) (I + {}^T U (\text{diag}(e^{-\Delta S}) - I) (U^T U)^{-1} U) \\
&= \text{diag}(e^{\Delta(\psi r + S)U}) (I - {}^T U (U^T U)^{-1} U) + \text{diag}(e^{\Delta(\psi r + S)U}) {}^T U \text{diag}(e^{-\Delta S}) (U^T U)^{-1} U \\
&= \text{diag}(e^{\Delta(\psi r + S)U}) (I - {}^T U (U^T U)^{-1} U) + \text{diag}(e^{\Delta(\psi r + S)U}) \text{diag}(e^{-\Delta S U}) {}^T U (U^T U)^{-1} U \\
&= \text{diag}(e^{\Delta(\psi r + S)U}) (I - {}^T U (U^T U)^{-1} U) + \text{diag}(e^{\Delta \psi r U}) {}^T U (U^T U)^{-1} U \tag{S17}
\end{aligned}$$

Where the second to last line holds under the assumption that each species belong to at most one island.

We further need to compute

$$\begin{aligned}
\int_{\tau_i}^{\tau_{i+1}} e^{(s-\tau_{i+1})A_i} a_i ds &= \psi \theta \int_{\tau_i}^{\tau_{i+1}} \text{diag}(e^{(s-\tau_{i+1})\psi r U}) ds {}^T U^T r \\
&= \psi \theta \int_{\tau_i}^{\tau_{i+1}} e^{(s-\tau_{i+1})\psi} ds {}^T U^T r \\
&= \theta (1 - e^{\psi \Delta}) {}^T U^T r \tag{S18}
\end{aligned}$$

We thus get $m_{\tau_{i+1}^-}$ with the help of Equations (S17) and (S18).

We now turn to the reduction of the variance expression. Remark first that A_i and Γ_i are symmetric, and so are $e^{\Delta A_i}$ and $e^{\Delta A_i} \Gamma_i$. Moreover, Γ_i is diagonal, and commutes with $e^{\Delta A_i}$, leading to :

$$\Sigma_{\tau_{i+1}^-} = e^{\Delta A_i} \Sigma_{\tau_i} e^{\Delta A_i} + \int_{\tau_i}^{\tau_{i+1}} e^{2(s-\tau_{i+1})A_i} \Gamma_i \Gamma_i ds$$

The first term can be computed thanks to equation (S17). For the second one we get

$$\begin{aligned}
\int_{\tau_i}^{\tau_{i+1}} e^{2(s-\tau_{i+1})A_i} \Gamma_i \Gamma_i ds &= \sigma^2 \int_{\tau_i}^{\tau_{i+1}} e^{2(s-\tau_{i+1})\text{diag}(r(\psi I+S)U)} ds (I - {}^T U (U^T U)^{-1} U) \text{diag}(rU) \\
&+ \sigma^2 \int_{\tau_i}^{\tau_{i+1}} e^{2(s-\tau_{i+1})\psi \text{diag}(rU)} ds {}^T U (U^T U)^{-1} U \text{diag}(rU) \\
&= \sigma^2 \int_{\tau_i}^{\tau_{i+1}} \text{diag}(e^{2(s-\tau_{i+1})(\psi r+S)U}) ds (\text{diag}(rU) - {}^T U (U^T U)^{-1} U) \\
&+ \sigma^2 \int_{\tau_i}^{\tau_{i+1}} \text{diag}(e^{2(s-\tau_{i+1})\psi rU}) ds {}^T U (U^T U)^{-1} U
\end{aligned} \tag{S19}$$

At the end, we get $\Sigma_{\tau_{i+1}^-}$ with the help of Equations (S17) and (S19).

C.4 The generalist matching mutualism (GMM) model

We recall the model formulation here. Assume that we rank first the n_1 plant traits, before the n_2 butterfly traits in the X vector. Traits evolve following the equation :

$$\begin{aligned}
\forall k \in \{1, \dots, n_1\}, \quad dX_t^{(k)} &= S \left(d_1 + \frac{1}{n_2} \sum_{l=n_1+1}^{n_1+n_2} X_t^{(l)} - X_t^{(k)} \right) dt + \sigma dW_t^{(k)} \\
\forall l \in \{n_1+1, \dots, n_1+n_2\}, \quad dX_t^{(l)} &= S \left(d_2 + \frac{1}{n_1} \sum_{k=1}^{n_1} X_t^{(k)} - X_t^{(l)} \right) dt + \sigma dW_t^{(l)}
\end{aligned}$$

In the general framework formulation, this leads to :

$$\begin{aligned}
a(t) &= {}^{tr}(Sd_1, \dots, Sd_1, Sd_2, \dots, Sd_2) \\
A &= \begin{pmatrix} S & 0 & \dots & 0 & \frac{-S}{n_2} & \dots & \dots & \frac{-S}{n_2} \\ 0 & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & & & \vdots \\ 0 & \dots & 0 & \ddots & \frac{-S}{n_2} & \dots & \dots & \frac{-S}{n_2} \\ \frac{-S}{n_1} & \dots & \dots & \frac{-S}{n_1} & \ddots & 0 & \dots & 0 \\ \vdots & & & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \ddots & \ddots & 0 \\ \frac{-S}{n_1} & \dots & \dots & \frac{-S}{n_1} & 0 & \dots & 0 & S \end{pmatrix} \\
\Gamma &= \sigma I
\end{aligned}$$

297 We would like to be able to compute the expectation and variance easily during each
 298 epoch. We thus want to reduce Equations (4a, 4b). For simplicity, we will write in the following
 299 $\Delta = \tau_i - \tau_{i+1}$. With some work, we can find the generic element of the matrix $e^{\Delta A}$.

300 First, we decompose $A = S(I + Z)$, where I is the identity matrix, and Z is made of two
 301 blocks with elements $\frac{-1}{n_2}$ and $\frac{-1}{n_1}$. I and Z commute, meaning that :

$$e^{\Delta A} = e^{\Delta S(I+Z)} = e^{\Delta SI} e^{\Delta SZ} = e^{\Delta S} e^{\Delta SZ}$$

302 Moreover, we can find by induction the generic element of the matrix Z^k , as presented in
 303 Figure (S2).

$$Z = \begin{pmatrix} & \begin{matrix} \xrightarrow{n_2} \\ \text{yellow block} \\ \xleftarrow{n_2} \end{matrix} \\ \begin{matrix} \xleftarrow{n_1} \\ \text{teal block} \\ \xrightarrow{n_1} \end{matrix} & \end{pmatrix} \quad Z^2 = \begin{pmatrix} \text{teal block} & \\ & \text{yellow block} \end{pmatrix}$$

$$\vdots \quad \vdots$$

$$Z^{2k+1} = \begin{pmatrix} & \text{yellow block} \\ \text{teal block} & \end{pmatrix} \quad Z^{2k} = \begin{pmatrix} \text{teal block} & \\ & \text{yellow block} \end{pmatrix}$$

FIGURE S2: Generic element of the matrix Z^k , $\forall k \in \mathbb{N}^*$.

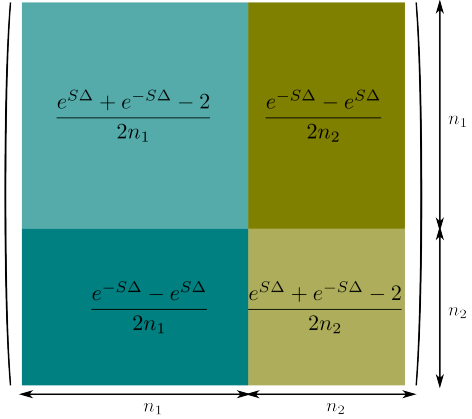
304 We then use this to find the generic element of the matrix

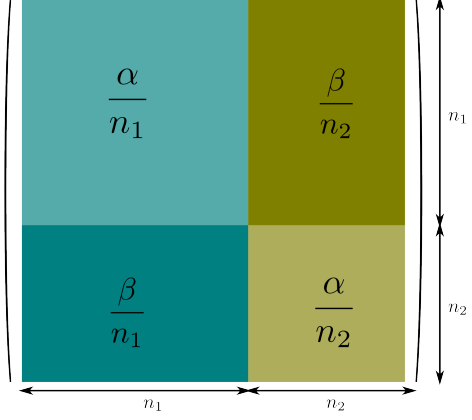
305 $e^{\Delta SZ} = \sum_{k \geq 0} \frac{S^k \Delta^k Z^k}{k!} = I + \sum_{k \geq 1} \frac{S^k \Delta^k Z^k}{k!}$. We recall that the odd and even parts of the
 306 exponential are :

$$e^\lambda - e^{-\lambda} = \sum_{k \geq 0} \frac{\lambda^k}{k!} - \sum_{k \geq 0} \frac{(-1)^k \lambda^k}{k!} = 2 \sum_{k \geq 0} \frac{\lambda^{2k+1}}{(2k+1)!}$$

$$\text{and } e^\lambda + e^{-\lambda} = 2 \sum_{k \geq 0} \frac{\lambda^{2k}}{(2k)!}$$

307 Then, matrices $e^{\Delta SZ}$ and $e^{\Delta A}$ are composed of four distinct blocks, which expressions are
 308 shown in Figure S3.

$$e^{\Delta SZ} = I + \sum_{k \geq 1} \frac{S^k \Delta^k Z^k}{k!} = I +$$


$$e^{\Delta A} = e^{\Delta S} e^{\Delta SZ} = e^{\Delta S} I +$$


$$\text{where } \alpha := \frac{e^{2S\Delta} - 2e^{S\Delta} + 1}{2}$$

$$\beta := \frac{1 - e^{2S\Delta}}{2}$$

FIGURE S3: Generic elements of matrices $e^{\Delta SZ}$ and $e^{\Delta A}$.

We thus got the main element from which we can derive the expectation vector $m_{\tau_{i+1}}^-$:

$$\begin{aligned}
m_{\tau_{i+1}}^- &= e^{\Delta A_i} m_{\tau_i} + \int_{\tau_i}^{\tau_{i+1}} e^{(s-\tau_{i+1})A_i} a_i(s) ds \\
&= e^{\Delta A_i} m_{\tau_i} + \int_{\tau_i}^{\tau_{i+1}} \begin{pmatrix} Sd_1 e^{S(s-\tau_{i+1})} + Sd_1 \frac{e^{2S(s-\tau_{i+1})} - 2e^{S(s-\tau_{i+1})} + 1}{2} + Sd_2 \frac{1 - e^{2S(s-\tau_{i+1})}}{2} \\ \vdots \\ Sd_1 e^{S(s-\tau_{i+1})} + Sd_1 \frac{e^{2S(s-\tau_{i+1})} - 2e^{S(s-\tau_{i+1})} + 1}{2} + Sd_2 \frac{1 - e^{2S(s-\tau_{i+1})}}{2} \\ Sd_2 e^{S(s-\tau_{i+1})} + Sd_1 \frac{1 - e^{2S(s-\tau_{i+1})}}{2} + Sd_2 \frac{e^{2S(s-\tau_{i+1})} - 2e^{S(s-\tau_{i+1})} + 1}{2} \\ \vdots \\ Sd_2 e^{S(s-\tau_{i+1})} + Sd_1 \frac{1 - e^{2S(s-\tau_{i+1})}}{2} + Sd_2 \frac{e^{2S(s-\tau_{i+1})} - 2e^{S(s-\tau_{i+1})} + 1}{2} \end{pmatrix} ds \\
m_{\tau_{i+1}}^- &= e^{\Delta A_i} m_{\tau_i} + \int_{\tau_i}^{\tau_{i+1}} \begin{pmatrix} S \frac{d_1 + d_2}{2} + S \frac{d_1}{2} e^{2S(s-\tau_{i+1})} - S \frac{d_2}{2} e^{2S(s-\tau_{i+1})} \\ \vdots \\ S \frac{d_1 + d_2}{2} + S \frac{d_1}{2} e^{2S(s-\tau_{i+1})} - S \frac{d_2}{2} e^{2S(s-\tau_{i+1})} \\ S \frac{d_1 + d_2}{2} - S \frac{d_1}{2} e^{2S(s-\tau_{i+1})} + S \frac{d_2}{2} e^{2S(s-\tau_{i+1})} \\ \vdots \\ S \frac{d_1 + d_2}{2} - S \frac{d_1}{2} e^{2S(s-\tau_{i+1})} + S \frac{d_2}{2} e^{2S(s-\tau_{i+1})} \end{pmatrix} ds \\
&= e^{\Delta A_i} m_{\tau_i} + \begin{pmatrix} -S \frac{d_1 + d_2}{2} \Delta + \frac{d_1}{4} (1 - e^{2S\Delta}) - \frac{d_2}{4} (1 - e^{2S\Delta}) \\ \vdots \\ -S \frac{d_1 + d_2}{2} \Delta + \frac{d_1}{4} (1 - e^{2S\Delta}) - \frac{d_2}{4} (1 - e^{2S\Delta}) \\ -S \frac{d_1 + d_2}{2} \Delta - \frac{d_1}{4} (1 - e^{2S\Delta}) + \frac{d_2}{4} (1 - e^{2S\Delta}) \\ \vdots \\ -S \frac{d_1 + d_2}{2} \Delta - \frac{d_1}{4} (1 - e^{2S\Delta}) + \frac{d_2}{4} (1 - e^{2S\Delta}) \end{pmatrix}
\end{aligned}$$

We now turn to the derivation of the covariance matrix, which requires simplifying :

$$\int_{\tau_i}^{\tau_{i+1}} (e^{(s-\tau_{i+1})A_i} \Gamma_i(s)) \operatorname{tr} (e^{(s-\tau_{i+1})A_i} \Gamma_i(s)) ds = \sigma^2 \int_{\tau_i}^{\tau_{i+1}} (e^{(s-\tau_{i+1})A_i}) \operatorname{tr} (e^{(s-\tau_{i+1})A_i}) ds$$

The expression of this last matrix is given in Figure [S4](#).

$$\begin{aligned}
& \sigma^2 \int_{\tau_i}^{\tau_{i+1}} \left(e^{(s-\tau_{i+1})A_i} \right) \text{tr} \left(e^{(s-\tau_{i+1})A_i} \right) ds \\
&= \sigma^2 \int_{\tau_i}^{\tau_{i+1}} e^{S(s-\tau_{i+1})} I + \left(\begin{array}{c|c} \frac{\alpha}{n_1} & \frac{\beta}{n_2} \\ \hline \frac{\beta}{n_1} & \frac{\alpha}{n_2} \end{array} \right) e^{S(s-\tau_{i+1})} I + \left(\begin{array}{c|c} \frac{\alpha}{n_1} & \frac{\beta}{n_1} \\ \hline \frac{\beta}{n_2} & \frac{\alpha}{n_2} \end{array} \right) ds \\
&= \sigma^2 \int_{\tau_i}^{\tau_{i+1}} e^{2S(s-\tau_{i+1})} I + \left(\frac{1}{n_1} + \frac{1}{n_2} \right) \left(\begin{array}{c|c} \frac{e^{4S(s-\tau_{i+1})} - 2e^{2S(s-\tau_{i+1})} + 1}{4} & \frac{1 - e^{4S(s-\tau_{i+1})}}{4} \\ \hline \frac{1 - e^{4S(s-\tau_{i+1})}}{4} & \frac{e^{4S(s-\tau_{i+1})} - 2e^{2S(s-\tau_{i+1})} + 1}{4} \end{array} \right) ds \\
&= \sigma^2 \left(\frac{1 - e^{2S\Delta}}{2S} I + \left(\frac{1}{n_1} + \frac{1}{n_2} \right) \left(\begin{array}{c|c} \frac{1 - e^{4S\Delta}}{16S} - \frac{1 - e^{2S\Delta}}{4S} - \frac{\Delta}{4} & \frac{e^{4S\Delta} - 1}{16S} - \frac{\Delta}{4} \\ \hline \frac{e^{4S\Delta} - 1}{16S} - \frac{\Delta}{4} & \frac{1 - e^{4S\Delta}}{16S} - \frac{1 - e^{2S\Delta}}{4S} - \frac{\Delta}{4} \end{array} \right) \right)
\end{aligned}$$

FIGURE S4: Generic elements of matrices that help us compute the covariance matrix of the distribution.

D SIMULATION AND INFERENCE

We do not give any new result in this Appendix section. Instead, we present the ways we implemented numerically simulations and inferences for all models described in the paper. These have been previously described in a number of papers.

D.1 Numerical methods for simulating data

D.1.1 Simulating the whole trajectory of the process

We use the Euler-Maruyama scheme, which works like the Euler scheme for ODEs, but with the addition of a small Gaussian random variable at each time step (Gardiner et al. 1985). We discretize each epoch (τ_i, τ_{i+1}) with a mesh Δ_t . We consider m standard Gaussian vectors of dimension $nd : (U_j)_{j=1}^m$. We approximate our SDE on this interval in the following way :

$$Y_0 = X_0$$

$$Y_{\tau_i+m\Delta_t} = Y_{\tau_i+(m-1)\Delta_t} + (a_i(\tau_i + (m-1)\Delta_t) - A_i Y_{\tau_i+(m-1)\Delta_t})\Delta_t + \Gamma(\tau_i + (m-1)\Delta_t)\sqrt{\Delta_t}U_m$$

When a branching occurs, the values of the process on the splitting branch are duplicated at the end of the vector Y . We then iterate this operation from the root up to present time.

This simulation allows us to get the whole trajectory of the process on the tree, which can mainly be used to produce pictures as in Figure S5, and eventually get a useful intuition on the process. However, we rarely use the whole trajectories, because observed data are only composed of tip trait values.

D.1.2 Simulating values of the process at the tips only

This second simulation protocol allows us to simulate the process values at the tips only. Suppose that we know the vector m of expectations and the covariance matrix Σ at the tips of the tree.

We then simply simulate numerically a Gaussian vector with law :

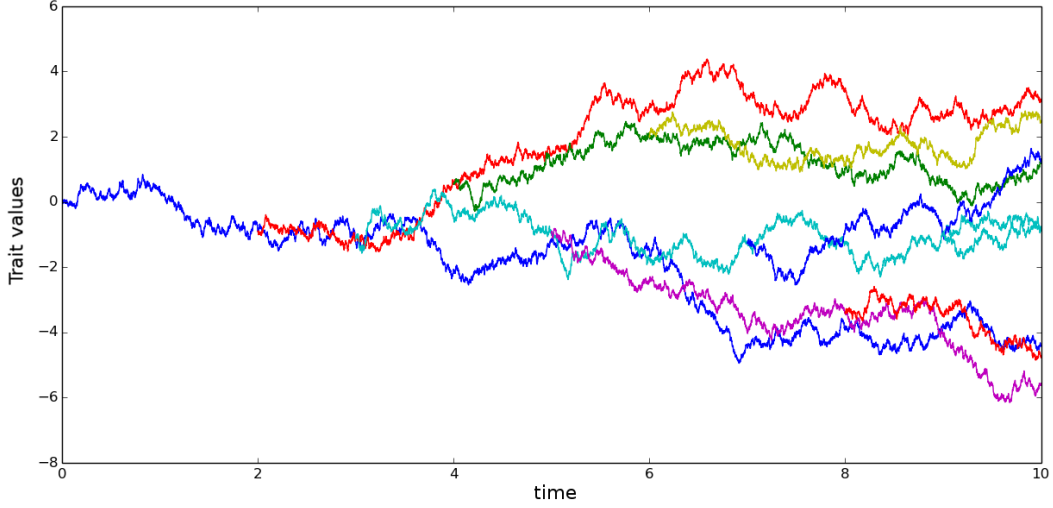


FIGURE S5: Evolution of a Brownian phenotypic trait along a tree, following the SDE : $dX_t = \sigma IdW_t$.

$$X_{t_f} \sim \mathcal{N}(m, \Sigma)$$

332 This is by far the quickest way to get the tip values. However, as the inference protocol
 333 relies on the use of the same vector of expectations and covariance matrix, one may prefer to use
 334 the other simulation protocols to test the consistency between simulation and inference. In case
 335 there is an issue with the derivation of the tip distribution, there would be a discrepancy between
 336 simulations and inferences.

337 D.2 Parameter inference

338 D.2.1 Parameter inference principle

339 We consider here that we know the topology of the true phylogeny with K tips, its branch
 340 lengths, and the state of d phenotypic traits at the tip, denoted by \mathcal{X} .

341 We assume any model of phenotypic evolution relying on linear SDEs, with vector of
 342 parameters p . We can compute the expectation m_p and the covariance Σ_p of the process X at
 343 tree tips, which law is then : $X \sim \mathcal{N}(m_p, \Sigma_p)$. Recall from Appendix A.2 that Σ_p is positive

definite in most cases, and is thus theoretically non-singular. However, one must be cautious with numerical implementations, as numerical approximations might still lead to ‘numerically non-invertible’ matrices. Here, we assume that the variance matrix is invertible, and the density of the vector X is :

$$\forall x \in \mathbb{R}^{Kd}, \quad f(x) = \frac{1}{\sqrt{(2\pi)^{Kd} \det(\Sigma_p)}} e^{-\frac{1}{2} \text{tr}(x-m_p)\Sigma_p^{-1}(x-m_p)}$$

We can thus write the likelihood of the observed phenotypic traits as,

$$\begin{aligned} \mathcal{L}(p) &= f(\mathcal{X}|p) \\ &= \frac{1}{\sqrt{(2\pi)^{Kd} \det(\Sigma_p)}} e^{-\frac{1}{2} \text{tr}(\mathcal{X}-m_p)\Sigma_p^{-1}(\mathcal{X}-m_p)} \end{aligned}$$

The maximum likelihood estimators (MLE) are the parameter values that maximize the likelihood function, that is,

$$\hat{p} = \underset{p}{\operatorname{argmax}} \mathcal{L}(p)$$

Equivalently, we can minimize the following function,

$$-\ln(\mathcal{L}(p)) = \frac{1}{2} Kd \ln(2\pi) + \frac{1}{2} \ln(\det(\Sigma_p)) + \frac{1}{2} \text{tr}(\mathcal{X} - m_p)\Sigma_p^{-1}(\mathcal{X} - m_p)$$

or, removing the constants,

$$U(p) = \ln(\det(\Sigma_p)) + \text{tr}(\mathcal{X} - m_p)\Sigma_p^{-1}(\mathcal{X} - m_p)$$

D.2.2 Analytical derivation of the MLE

Among all models described in the paper, only the BM model allows the analytic derivation of the MLE estimators. Take for illustration a BM model without drift starting with $(m_0, v_0) = (0, 0)$. According to Table S1, the expectation m and covariance matrix Σ at the tips are $m = 0$ and $\Sigma = \sigma^2 T$, where matrix T has element $T^{(k,l)} = t_{k,l}$.

We get the MLE $\hat{\sigma}$ by looking analytically for the minimum of U ,

$$\begin{aligned} U(\sigma) &= \ln(\det(\sigma^2 T)) + \text{tr} \mathcal{X} \frac{T^{-1}}{\sigma^2} \mathcal{X} \\ &= \ln \det T + 2n \ln \sigma + \frac{1}{\sigma^2} \text{tr} \mathcal{X} T^{-1} \mathcal{X} \\ \frac{dU}{d\sigma} &= \frac{2n}{\sigma} - \frac{2}{\sigma^3} \text{tr} \mathcal{X} T^{-1} \mathcal{X} \end{aligned}$$

Thus leading to,

$$\hat{\sigma}^2 = \frac{1}{n} \text{tr} \mathcal{X} T^{-1} \mathcal{X}$$

D.2.3 Speeding up the ML estimation by reducing the dimension of the parameter space

Maximizing the likelihood can take a long time, especially when the dimension of the parameter space is large. It can thus be interesting to make assumptions that lower the number of parameters, when this is biologically tolerable. Examples include,

- starting an OU process with $m_0 = \theta$,
- considering no root variance, $v_0 = 0$,
- starting a PM model with $m_0 = \theta$ (in which case we easily show that the expectation remains θ in all lineages),
- putting $\psi = 0$ in the PM model.

In many models (e.g. BM, OU, ACDC, PM with $m_0 = \theta \dots$), distinct sets of parameters p_1 and p_2 are involved in the computation of m and Σ , and the expectation vector m can be expressed as $m = Cp_1$. In this case, at a given p_2 , we can analytically get the parameters p_1 maximizing $\ln(\mathcal{L}(p_1, p_2))$,

$$\frac{\partial}{\partial p_1} U(p_1, p_2) = 0 \iff \frac{d}{dp_1} \text{tr}(\mathcal{X} - Cp_1) \Sigma_{p_2}^{-1} (\mathcal{X} - Cp_1) = 0$$

Doing so, we get the same formula as in (Hansen 1997; Butler and King 2004), i.e.

$$\hat{p}_1 = (\text{tr} C_1 \Sigma_{p_2}^{-1} C_1)^{-1} \text{tr} C_1 \Sigma_{p_2}^{-1} \mathcal{X}.$$

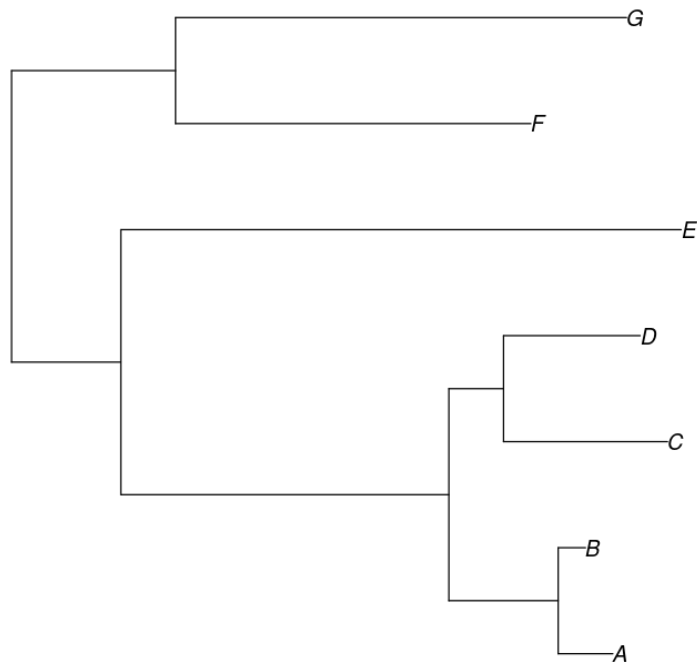
E TUTORIAL : USING THE RPANDA CODE TO STUDY TRAIT COEVOLUTION

The aim of this section is to describe the R code associated to our framework. We describe the class `PhenotypicModel`, we show how to manipulate the different methods included in the class, we illustrate their use around a simple (non-ultrametric) tree, and we finally explain how to use our codes to write new models fitting the framework.

We first need to load useful R packages, along with our codes, and a small, non-ultrametric, tree.

```
In [219]: source("Loading.R")
```

```
newick <- "((((A:1,B:0.5):2,(C:3,D:2.5):1):6,E:10.25):2,(F:6.5,G:8.25):3):1;"  
tree <- read.tree(text=newick)  
plot(tree)
```



E.1 The 'PhenotypicModel' class

Our code is structured around one main R class that we called 'PhenotypicModel', which is intended to mimic the framework that we proposed in the main text. Each object of the 'PhenotypicModel' encompasses informations on the tree, on the parameters of the model, on the starting values, and, finally, on the collection of (a_i, A_i, Γ_i) for all epochs.

E.1.1 Loading a pre-defined model

Because we wanted this code both to be user-friendly and to serve as an illustration of what can be written within this framework, we implemented all models in main Table 1 in a generic constructor `createModel`, in the file 'ModelBank.R', that takes for arguments the tree and the name of the required model.

Available models include :

BM Brownian Motion model with linear drift.

Starts with two lineages having the same value $X_0 \sim \mathcal{N}(m_0, v_0)$.

One trait in each lineage, all lineages evolving independently after branching following the equation.

$$dX_t^{(i)} = ddt + \sigma dW_t^{(i)}$$

BM_from0 Same as above, but starting with two lineages having the same value $X_0 \sim \mathcal{N}(0, 0)$.

BM_from0_driftless Same as above, but with $d = 0$.

OU Ornstein-Uhlenbeck model.

Starts with two lineages having the same value $X_0 \sim \mathcal{N}(m_0, v_0)$.

One trait in each lineage, all lineages evolving independently after branching, following the equation :

$$dX_t^{(i)} = \psi(\theta - X_t)dt + \sigma dW_t^{(i)}$$

405 **OU_from0** Same as above, but starting with two lineages having the same value $X_0 \sim \mathcal{N}(0, 0)$.

406 **ACDC** ACcelerating or DeCelerating model.

407 Starts with two lineages having the same value $X_0 \sim \mathcal{N}(m_0, v_0)$.

408 One trait in each lineage, all lineages evolving independently after branching, following the
409 equation :

$$dX_t^{(i)} = \sigma_0 e^{rt} dW_t^{(i)}$$

410 **DD** Diversity-Dependent model.

411 Starts with two lineages having the same value $X_0 \sim \mathcal{N}(m_0, v_0)$.

412 One trait in each lineage, all lineages evolving independently after branching, following the
413 equation :

$$dX_t^{(i)} = \sigma_0 e^{rn_t} dW_t^{(i)}$$

414 **PM** Phenotype Matching model.

415 Starts with two lineages having the same value $X_0 \sim \mathcal{N}(m_0, v_0)$.

416 One trait in each lineage, all lineages evolving then non-independently following the
417 expression :

$$dX_t^{(i)} = \psi \left(\theta - X_t^{(i)} \right) + S \left(\frac{1}{n} \sum_{k=1}^n X_t^{(k)} - X_t^{(i)} \right) + \sigma dW_t^{(i)}$$

418 **PM_OUless** Simplified Phenotype Matching model.

419 Starts with two lineages having the same value $X_0 \sim \mathcal{N}(m_0, v_0)$.

420 One trait in each lineage, all lineages evolving then non-independently following the
421 expression :

$$dX_t^{(i)} = S \left(\frac{1}{n} \sum_{k=1}^n X_t^{(k)} - X_t^{(i)} \right) + \sigma dW_t^{(i)}$$

422 To get a first glimpse at ‘PhenotypicModel’ objects, we first create two such objects. The
 423 first one is a Brownian Motion (BM), the second one is an Ornstein-Uhlenbeck process (OU).
 424 Note that both models include m_0 and v_0 as parameters.

```
In [220]: modelBM <- createModel(tree, 'BM')
          modelOU <- createModel(tree, 'OU')
```

425 E.1.2 Access to the content of the model

426 The function `show` is intended to give basic information on a specific ‘PhenotypicModel’ object,
 427 whereas the `print` function displays full information.

```
In [221]: show(modelBM)
```

```
*****
*** Object of Class PhenotypicModel ***
*** Name of the model : [1] "BM"
*** Parameters of the model : [1] "m0"      "v0"      "d"      "sigma"
*** Description : Brownian Motion model with linear drift.
Starts with two lineages having the same value  $X_0 \sim \text{Normal}(m_0, v_0)$ .
One trait in each lineage, all lineages evolving independently after branching.
 $dX_t = d \, dt + \text{sigma} \, dW_t$ 
*** Periods : the model is cut into 13 parts.
For more details on the model, call : print(PhenotypicModel)
```

```
*****
```

```
In [222]: print(modelOU)
```

```
*****
*** Object of Class PhenotypicModel ***
*** Name of the model : [1] "OU"
*** Parameters of the model : [1] "m0"      "v0"      "psi"     "theta" "sigma"
*** Description : Ornstein-Uhlenbeck model.
```

```

Starts with two lineages having the same value  $X_0 \sim \text{Normal}(m_0, v_0)$ .

One trait in each lineage, all lineages evolving independently after branching.
 $dX_t = \psi(\theta - X_t) dt + \sigma dW_t$ 

*** Epochs : the model is cut into 13 parts.

[1] 0.00 2.00 3.00 8.00 9.00 9.50 10.00 10.50 11.00 11.25 11.50 12.00
[13] 12.25

*** Lineages branching (to be copied at the end of the corresponding period) :

[1] 1 1 2 1 5 2 1 7 1 4 6 5 3

*** Positions of the new trait at the end of each period :

[1] 2 3 4 5 6 0 7 0 0 0 0 0 0

*** Initial condition :

function (params)
return(list(mean = c(params[1]), var = matrix(c(params[2]))))
<environment: 0x9617460>

*** Vectors a_i, A_i, Gamma_i on each period i :

function (i, params)
{
  vectorU <- getLivingLineages(i, eventEndOfPeriods)
  vectorA <- function(t) return(params[3] * params[4] * vectorU)
  matrixGamma <- function(t) return(params[5] * diag(vectorU))
  matrixA <- params[3] * diag(vectorU)
  return(list(a = vectorA, A = matrixA, Gamma = matrixGamma))
}

<environment: 0x9617460>

*** Constraints on the parameters :

function (params)
return(params[2] >= 0 && params[5] >= 0 && params[3] != 0)
<environment: 0x9617460>

*** Default parameter values : [1] 0 0 1 0 1

*** Tip labels :

```

```
[1] "A" "B" "C" "D" "E" "F" "G"
```

```
*** Tip labels for simulations :
```

```
[1] "A" "F" "E" "G" "C" "D" "B"
```

```
*****
```

428 *E.1.3 List of class attributes*

429 The latter command gave us some insight into how a `PhenotypicModel` is defined. It has the
430 following list of attributes :

431 **name** a name,

432 **paramsNames** the names of all parameters,

433 **comment** a short description,

434 **period** the vector of times at which successive branching and death of lineages occur,

435 **numbersCopy** vector containing the lineage number which branches or dies at the end of each
436 period,

437 **numbersPaste** vector containing the lineage number in which a daughter lineage is placed at
438 the end of each period (zero if the end of the period corresponds to a death),

439 **initialCondition** a function of the parameters giving the initial mean and variance of the
440 gaussian process at the root of the tree,

441 **aAGamma** the functions corresponding to $a_i(t)$, A_i , and $\Gamma_i(t)$ that define the evolution of the
442 process on each period, depending on parameters,

443 **constraints** a function of the parameters giving the definition range,

444 **params0** a vector of default parameter values.

445 Each of these attributes can be accessed and changed through the use of the following
446 syntax.

```

In [223]: modelBM['name']

447 Out[223]: 'BM'

In [224]: modelBM['paramsNames']

448 Out[224]: 'm0' 'v0' 'd' 'sigma'

In [225]: modelOU['paramsNames'] <- c("mean0", "var0", "selectionStrength", "equilibrium",
    "noise")
    show(modelOU)

*****

*** Object of Class PhenotypicModel ***

*** Name of the model : [1] "OU"

*** Parameters of the model : [1] "mean0"          "var0"          "selectionStrength"
[4] "equilibrium"      "noise"

*** Description : Ornstein-Uhlenbeck model.

Starts with two lineages having the same value  $X_0 \sim \text{Normal}(m0, v0)$ .

One trait in each lineage, all lineages evolving independently after branching.

 $dX_t = \psi(\theta - X_t) dt + \sigma dW_t$ 

*** Periods : the model is cut into 13 parts.

For more details on the model, call : print(PhenotypicModel)

*****

449     However, changes must be made cautiously, in order to keep a coherent model. For
450 example, changing 'paramsNames' for a shorter vector would not be authorized, but other
451 deleterious actions could work and lead to issues with methods associated to PhenotypicModel
452 objects.

In [226]: modelOU['paramsNames'] <- c("mean0", "var0")

Error in validityMethod(as(object, superClass)): [PhenotypicModel : validation]
There should be the same number of default parameters and parameter names.

```


E.2 Methods associated to the 'PhenotypicModel' class

All 'PhenotypicModel' objects are associated to methods intended to do the basic operations that we need to do with models of trait evolution, i.e.,

1. simulate tip trait data,
2. compute the likelihood of tip trait data,
3. fit the model to tip trait data.

E.2.1 Simulating tip trait data

The method `simulateTipData` works for any `PhenotypicModel` object. We simply give it the model and the set of parameters and it returns a realisation of the process (tip data).

```
In [227]: dataBM <- simulateTipData(modelBM, c(0,0,0,1))
          dataBM
```

```
*** Simulation of tip trait values ***
```

```
Simulates step-by-step the whole trajectory, but returns only the tip data.
```

```
Computation time : 0.3909395 secs
```

```
Out [227]:
```

A	-2.71863
F	1.043329
E	0.665404
G	-3.440327
C	0.272335
D	-0.7023421
B	-2.010951

A third, optional, argument, changes the behaviour of the method.

- "method=1" : first computes the tip distribution at present, before drawing a realization of this distribution,

- 468 • "method=2" : simulates step-by-step the whole trajectory of the process, plots the
469 trajectories through time, and returns the tip data.
- 470 • "method=3" : (default) simulates step-by-step the whole trajectory of the process, before
471 returning only the tip data.

```
In [228]: dataOU <- simulateTipData(modelOU, c(0,0,1,5,1), method=1)
         dataOU
```

*** Simulation of tip trait values ***

Computes the tip distribution, and returns a simulated dataset drawn in this distribution.

Computation time : 0.0009741783 secs

472 Out [228]:

A	4.179412
B	5.776153
C	4.984526
473 D	4.480901
E	5.693471
F	4.636019
G	5.815942

```
In [229]: simulateTipData(modelBM, c(0,0,0,1), method=2)
```

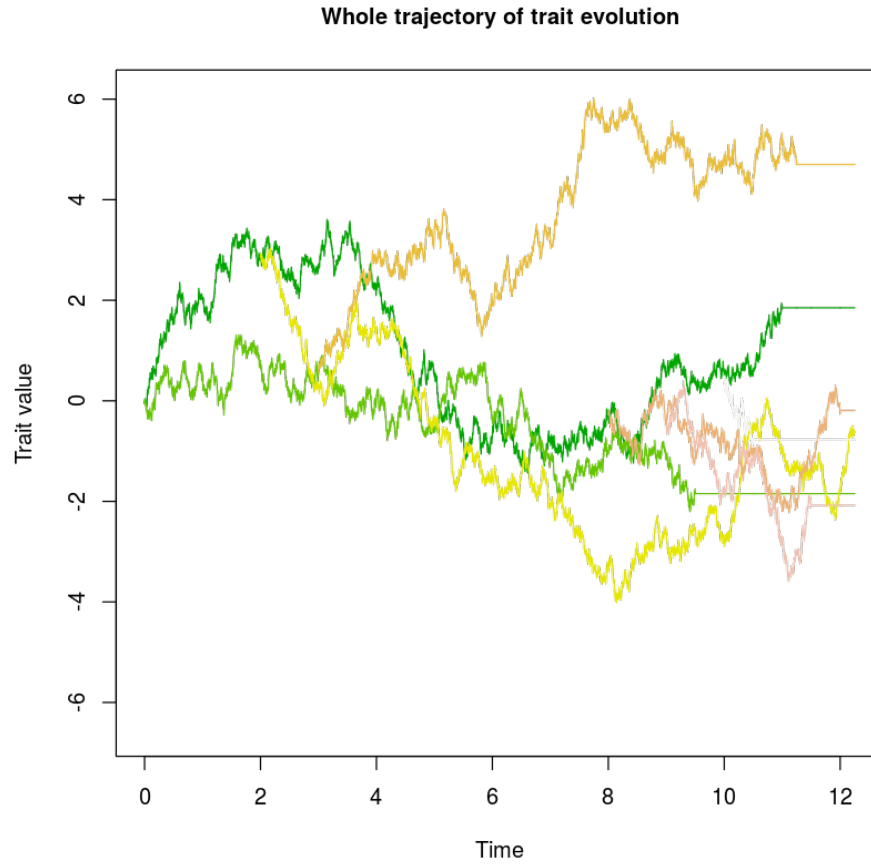
*** Simulation of tip trait values ***

Simulates step-by-step the whole trajectory, plots it, and returns tip data.

Computation time : 0.479032 secs

474 Out [229]:

A	1.850113
F	-1.846854
E	-0.6321431
475 G	4.701758
C	-0.1940776
D	-2.077116
B	-0.7752916



476

477 *E.2.2 Getting the distribution of the model under a given set of parameters*

478 The method `getTipDistribution` computes the mean vector m and variance matrix Σ such
 479 that, under the model, the tip trait data X follows $\mathcal{N}(m, \Sigma)$.

480 The related method `getDataLikelihood` returns the $-\ln(\text{likelihood})$ of a given data set
 481 under the model, with a given set of parameters.

```
In [230]: getTipDistribution(modelBM, c(0,0,1,1))
```

482 **Out [230]:**

	A	11
	B	10.5
	C	12
483	\$mean	D 11.5
	E	12.25
	F	9.5
	G	11.25

		A	B	C	D	E	F	G
	A	11	10	8	8	2	0	0
	B	10.0	10.5	8.0	8.0	2.0	0.0	0.0
484	\$Sigma	C 8	8	12	9	2	0	0
	D	8.0	8.0	9.0	11.5	2.0	0.0	0.0
	E	2.00	2.00	2.00	2.00	12.25	0.00	0.00
	F	0.0	0.0	0.0	0.0	0.0	9.5	3.0
	G	0.00	0.00	0.00	0.00	0.00	3.00	11.25

```
In [231]: getDataLikelihood(modelBM, dataBM, c(0,0,1,1))
```

```
485 Out[231]: 36.0510113479088
```

486 E.2.3 Maximum likelihood estimation of parameters

487 The method `fitTipData` uses the latter two methods to find the set of parameters that
 488 minimizes $-\ln(\text{likelihood})$ for a given model, on a given data set. We can apply this method to
 489 simulated datasets, and compare the maximum likelihood estimators with the parameters used in
 490 the simulation.

491 Note that this function accepts a third, optional, parameter, that is the starting vector
 492 ‘params0’ given to optimize the likelihood. If no value is specified, the function takes the
 493 attribute ‘params0’ in the `PhenotypicModel` object.

```
In [232]: fitTipData(modelBM, dataBM)
```

```
*** Fit of tip trait data ***
```

```
Finds the maximum likelihood estimators of the parameters,
```

```
returns the likelihood and the inferred parameters.
```

```
**WARNING** : This function uses the standard R optimizer "optim".
```

```
It may not always converge well.
```

```
Please double check the convergence by trying
```

```
distinct parameter sets for the initialisation.
```

```
Computation time : 0.02105212 secs
```

```
494 Out[232]:
```

```
495 $value 13.3539168672421
```

```
496 $inferredParams m0 0.112360024529455
```

```
497     v0 4.3703974585017e-08
```

```
498     d -0.0733871266399529
```

```
499     sigma 0.64761762031608
```

```
In [233]: fitTipData(modelOU, dataOU)
```

```
*** Fit of tip trait data ***
```

```
Finds the maximum likelihood estimators of the parameters,
```

```
returns the likelihood and the inferred parameters.
```

```
**WARNING** : This function uses the standard R optimizer "optim".
```

```
It may not always converge well.
```

```
Please double check the convergence by trying
```

```
distinct parameter sets for the initialisation.
```

```
Computation time : 0.2915776 secs
```

```
500 Out[233]:
```

```
501 $value 7.5162883379935
```

```
502 $inferredParams mean0 13.5665180225751
```

```

503      var0 1.6815664554916e-05
504      selectionStrength 0.648513938633288
505      equilibrium 5.05532921748184
506      noise 0.766630199120977

```

507 It doesn't seem quite good, but it also seems like the choice in the starting parameters
 508 m_0, v_0 has a bad influence. As presented in Online Appendix [D.2](#), in many models (e.g. BM, OU,
 509 ACDC, PM with $m_0 = \theta \dots$), distinct sets of parameters p_1 and p_2 are involved in the
 510 computation of m and Σ , and the expectation vector m can be expressed as $m = Cp_1$. In
 511 particular, many models verify $m = {}^{tr}(m_0, m_0, \dots m_0)$. When this is the case, the fit of tip data
 512 can be improved and speeded up by using the third parameter of the function `GLSstyle=TRUE`.

```

In [234]: fitTipData(modelBM, dataBM, GLSstyle=TRUE)
          fitTipData(modelOU, dataOU, GLSstyle=TRUE)

```

```
*** Fit of tip trait data ***
```

```

Finds the maximum likelihood estimators of the parameters,
returns the likelihood and the inferred parameters.

```

```
**WARNING** : This function uses the standard R optimizer "optim".
```

```
It may not always converge well.
```

```

Please double check the convergence by trying
distinct parameter sets for the initialisation.

```

```
Computation time : 0.03260899 secs
```

```
513 Out [234]:
```

```

514 $value 13.5302740469078
515 $inferredParams m0 -0.00550320295296933
516      v0 2.28469756397133e-07
517      d -0.313019528308928
518      sigma 0.663621107698308

```

```
*** Fit of tip trait data ***
```

```
Finds the maximum likelihood estimators of the parameters,  
returns the likelihood and the inferred parameters.
```

```
Computation time : 0.1760004 secs
```

```
519 Out[234]:
```

```
520 $value 7.82305350777471
```

```
521 $inferredParams mean0 5.10957361631891
```

```
522     var0 3.36222531349288e-05
```

```
523     selectionStrength 1.87722870245168
```

```
524     equilibrium -1.98889519193151
```

```
525     noise 1.91905948952067
```

```
526     With so few data in hand, we could also prefer to consider directly models starting with  
527  $(m_0, v_0) = (0, 0)$ . We create two new models ‘BM_from0’ and ‘OU_from0’ with the subtle  
528 difference that  $(m_0, v_0) = (0, 0)$  and the models thus retain respectively only two and three  
529 parameters.
```

```
530     These two models are included in the ‘ModelBank’ file.
```

```
In [235]: modelBMfromZero <- createModel(tree, 'BM_from0')  
          modelBMfromZero['paramsNames']
```

```
531 Out[235]: 'd' 'sigma'
```

```
In [236]: modelOUfromZero <- createModel(tree, 'OU_from0')  
          modelOUfromZero['paramsNames']
```

```
532 Out[236]: 'psi' 'theta' 'sigma'
```

```
In [237]: fitTipData(modelBMfromZero, dataBM)
```

```

*** Fit of tip trait data ***

Finds the maximum likelihood estimators of the parameters,
returns the likelihood and the inferred parameters.

**WARNING** : This function uses the standard R optimizer "optim".

It may not always converge well.

Please double check the convergence by trying
distinct parameter sets for the initialisation.

Computation time : 0.01061678 secs

```

533 **Out[237]:**

```

534 $value 13.3540474589618

535 $inferredParams d -0.0633929373190768

536 sigma 0.647501517840828

```

537 The `fitTipData` function uses the `optim` function available in R to maximize the
538 likelihood. This optimizer is widely used to fit phenotypic models, but is known to sometimes
539 converge on a local optima rather than the maximum likelihood. It is thus important to assess the
540 sensitivity of the solution to the choice of the initial parameter values before drawing conclusions.

541 Finally, the functions `getTipDistribution`, `simulateTipData` and `fitTipData` all have a
542 last optional argument, called `v` for “verbose mode”. With `v=TRUE`, the functions gives
543 informations in the console, whereas with `v=FALSE` the function remains silent.

544 *E.3 Toward an in-depth understanding of the code structure*

545 This section can be skipped if you are not interested in using this framework to build your own
546 model. Otherwise, it is worth understanding how the different models relate to each others.

547 *E.3.1 Relationships between the different classes of models*

548 The superclass, for which all the above-mentionned functions are defined, is the
549 `PhenotypicModel` class. When a model is only known as a `PhenotypicModel`, the method that

550 computes the tip distribution, namely `getTipDistribution` is the most general one. It thus
551 computes the distribution by resolving numerically the ODE system presented in main text
552 Equations (5a, 5b), which can take a lot of time.

553 However, faster algorithms are available to compute the tip distribution under specific
554 models (see e.g. analytical tip distribution formulas in Table S1). This is the rationale to define
555 subclasses :

556 **PhenotypicBM** For the Brownian model.

557 **PhenotypicOU** For the Ornstein-Uhlenbeck model.

558 **PhenotypicACDC** For the Accelerating/Decelerating model.

559 **PhenotypicDD** For the Diversity-Dependent model.

560 **PhenotypicPM** For the Phenotype-Matching model.

561 **PhenotypicGMM** For the Generalist Matching Mutualism model.

562 **PhenotypicADiag** Models for which, $\forall i, A_i$ is symmetric and $\Gamma_i = \sigma I$.

563 For each of these subclasses, an other, more appropriated, function `getTipDistribution`
564 has been written. `PhenotypicModels` which are also `PhenotypicOU`, will preferentially use
565 methods defined for `PhenotypicOU` when they exist.

566 *E.3.2 Application : three different ways to define an OU*

567 In the `createModel` function, the keyword ‘OU’ constructs a model in the class `PhenotypicOU`.
568 In this class, the function `getTipDistribution` uses the analytical formula show in Online
569 Appendix B.1 to speed up the computation of m and Σ .

570 Alternatively, the keyword ‘OUBis’ defines the exact same model, but as an instance of
571 the class `PhenotypicADiag`. Thus, the function `getTipDistribution` uses the reduction show in
572 Online Appendix C.1 to compute m and Σ .

573 Last, the keyword ‘OUter’ still defines the exact same model, but as an instance of the
 574 class `PhenotypicModel`. Thus, the function `getTipDistribution` uses the resolution of the ODE
 575 system to compute m and Σ .

576 The following lines of code show that the function returns the same value with the three
 577 different methods, but do not take the same amount of time.

```
In [240]: modelOU <- createModel(tree, 'OU')
          modelOUbis <- createModel(tree, 'OUbis')
          modelOUter <- createModel(tree, 'OUter')
          params <- c(0,0,0.2,1,2)
```

```
In [241]: getTipDistribution(modelOU, params, v=TRUE)
```

*** Computation of tip traits distribution through the analytical formula for an OU process ***

Computation time : 0.000497818 secs

578 Out[241]:

	A	0.8891968
	B	0.8775436
	C	0.909282
579 \$mean	D	0.8997412
	E	0.9137064
	F	0.8504314
	G	0.8946008

580 \$Sigma

	A	B	C	D	E	F	G
A	9.8772266	7.2724966	2.3654513	2.6142280	0.1171813	0.0000000	0.0000000
B	7.2724966	9.8500442	2.6142280	2.8891687	0.1295054	0.0000000	0.0000000
C	2.36545128	2.61422796	9.91770253	3.23775807	0.09593997	0.00000000	0.00000000
581 D	2.6142280	2.8891687	3.2377581	9.8994816	0.1060301	0.0000000	0.0000000
E	0.11718135	0.12950541	0.09593997	0.10603007	9.92553417	0.00000000	0.00000000
F	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	9.7762923	0.3657529
G	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.3657529	9.8889100

```
In [242]: getTipDistribution(model0Ubis, params, v=TRUE)
```

```
*** Computation of tip traits distribution through integrated formula ***
```

```
(Method working for models with a constant, A diagonalizable, and Gamma constant)
```

```
Computation time : 0.002770185 secs
```

```
582 Out [242]:
```

```
583 $mean
```

A	0.8891968
F	0.8504314
E	0.9137064
G	0.8946008
C	0.909282
D	0.8997412
B	0.8775436

```
584 $Sigma
```

```
585
```

	A	F	E	G	C	D	B
A	9.8772266	0.0000000	0.1171813	0.0000000	2.3654513	2.6142280	7.2724966
F	0.0000000	9.7762923	0.0000000	0.3657529	0.0000000	0.0000000	0.0000000
E	0.11718135	0.00000000	9.92553417	0.00000000	0.09593997	0.10603007	0.12950541
G	0.00000000	0.3657529	0.00000000	9.8889100	0.00000000	0.00000000	0.00000000
C	2.36545128	0.00000000	0.09593997	0.00000000	9.91770253	3.23775807	2.61422796
D	2.6142280	0.00000000	0.1060301	0.00000000	3.2377581	9.8994816	2.8891687
B	7.2724966	0.00000000	0.1295054	0.00000000	2.6142280	2.8891687	9.8500442

```
In [243]: getTipDistribution(model0Uter, params, v=TRUE)
```

```
*** Computation of tip traits distribution through ODE resolution ***
```

```
(Method working for any model)
```

```
Computation time : 0.01829243 secs
```

```
586 Out [243]:
```

	A	0.8891984
	F	0.8504309
	E	0.9137081
587	\$mean	G 0.8946024
	C	0.9092837
	D	0.8997429
	B	0.8775447

588 **\$Sigma**

	A	F	E	G	C	D	B
A	9.8772243	0.0000000	0.1171837	0.0000000	2.3654834	2.6142593	7.2725143
F	0.0000000	9.7762896	0.0000000	0.3657561	0.0000000	0.0000000	0.0000000
E	0.11718371	0.00000000	9.92553239	0.00000000	0.09594306	0.10603262	0.12950776
589	G	0.0000000	0.3657561	0.0000000	9.8889077	0.0000000	0.0000000
C	2.36548343	0.00000000	0.09594306	0.00000000	9.91769978	3.23780799	2.61425810
D	2.6142593	0.0000000	0.1060326	0.0000000	3.2378080	9.8994793	2.8891973
B	7.2725143	0.0000000	0.1295078	0.0000000	2.6142581	2.8891973	9.8500418

```
In [244]: dataOU <- simulateTipData(modelOU, c(0,0,0.2,1,2))
```

```
fitTipData(modelOU, dataOU)
```

```
fitTipData(modelOUbis, dataOU)
```

```
fitTipData(modelOUter, dataOU)
```

*** Simulation of tip trait values ***

Simulates step-by-step the whole trajectory, but returns only the tip data.

Computation time : 0.2363398 secs

*** Fit of tip trait data ***

Finds the maximum likelihood estimators of the parameters,

returns the likelihood and the inferred parameters.

****WARNING**** : This function uses the standard R optimizer "optim".

It may not always converge well.

Please double check the convergence by trying

distinct parameter sets for the initialisation.

Computation time : 0.1814284 secs

590 **Out[244]:**

591 **\$value** 15.0174906724384

592 **\$inferredParams m0** -26.3722559360675

593 **v0** 0.111663973605588

594 **psi** 0.0973609295443122

595 **theta** 14.9673044542728

596 **sigma** 1.12338425846849

*** Fit of tip trait data ***

Finds the maximum likelihood estimators of the parameters,
returns the likelihood and the inferred parameters.

****WARNING**** : This function uses the standard R optimizer "optim".

It may not always converge well.

Please double check the convergence by trying
distinct parameter sets for the initialisation.

Computation time : 0.7557919 secs

597 **Out[244]:**

598 **\$value** 15.0174906724384

599 **\$inferredParams m0** -26.3722559360675

600 **v0** 0.111663973605588

601 **psi** 0.0973609295443122

602 **theta** 14.9673044542728

603 **sigma** 1.12338425846849

*** Fit of tip trait data ***

Finds the maximum likelihood estimators of the parameters,

returns the likelihood and the inferred parameters.

****WARNING**** : This function uses the standard R optimizer "optim".

It may not always converge well.

Please double check the convergence by trying

distinct parameter sets for the initialisation.

Computation time : 6.088683 secs

604 **Out[244]:**

605 **\$value** 15.0174914969285

606 **\$inferredParams m0** -26.3722559360675

607 **v0** 0.111663973605588

608 **psi** 0.0973609295443122

609 **theta** 14.9673044542728

610 **sigma** 1.12338425846849

611 Focusing on the computation time, it is quite easily seen how interesting it can be to do
612 some more analytical work and write more appropriated `getTipDistribution` functions. Still,
613 the default function written for the superclass `PhenotypicModel` should always work.

614 *E.3.3 Using the framework to define a new model*

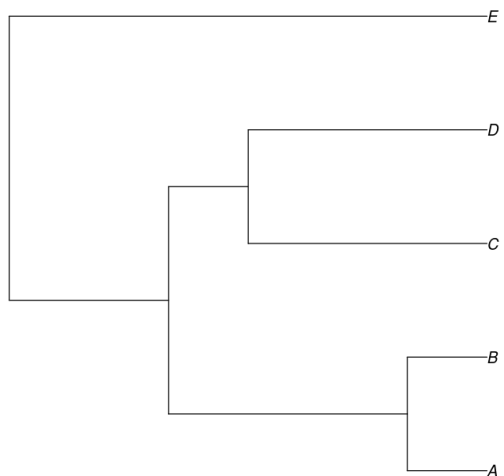
615 We illustrate here how the current code can be used to numerically study a specific model that
616 has not been implemented elsewhere. We focus here on the implementation of the ‘GMM’ model
617 described in the main text, explaining step by step the following procedure, that is generalizable
618 to any model :

- 619 1. we identify what the periods are,
- 620 2. we write the model in a vectorial form on each period,
- 621 3. we implement it naively first,

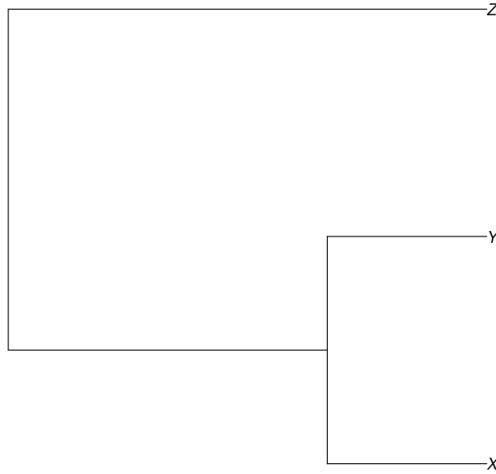
622 4. we make analytical developments to speed up the computation time, and subsequently
623 introduce a new class more appropriated to this model.

624 For simplicity, we implement GMM for two ultrametric trees here. In our example, the
625 two trees will be :

```
In [245]: newick1 <- "((A:1,B:1):3,(C:3,D:3):1):2,E:6);"  
          tree1 <- read.tree(text=newick1)  
          plot(tree1)  
          newick2 <- "(X:1.5,Y:1.5):3,Z:4.5);"  
          tree2 <- read.tree(text=newick2)  
          plot(tree2)
```



626



627

628 The first step consists in implementing a function `endOfPeriodsGMM(tree1, tree2)`,
 629 which takes as input two trees (the trees corresponding to our two interacting clades), and
 630 returns :

- 631 • the list of successive branching times (τ_i) (vector `periods`),
- 632 • information on which branch gives birth at that time (vector `copy`),
- 633 • the number assigned to the newly created branch at that time (vector `paste`),
- 634 • the number of lineages in clade 1 and 2 at each time (vectors `nLineages1` and `nLineages2`),
- 635 • the label of tips at the end (vector `labeling`).

636 For example, our function, called on the two preceding trees, returns :

```
In [246]: endOfPeriodsGMM(tree1, tree2)
```

637 Out[246]:

```
$periods 0 1.5 2 3 4.5 5 6
```

```
$copy 1 3 1 3 5 1 0
```



```
640 $paste 2 4 3 4 7 5 0
```

```
641 $nLineages1 2 2 3 4 4 5 0
```

```
642 $nLineages2 1 2 2 2 3 3 0
```

```
643 $labeling 'A' 'E' 'C' 'D' 'B' 'X' 'Z' 'Y'
```

644 The second step now consists in writing the model in the vectorial form required in the
645 framework, during each epoch i . The form of the a , A and Γ matrices is shown in Online
646 Appendix C.4, and depends on the number of lineages in the two clades during each epoch.

647 We introduce the constructor `createModelCoevolution(tree1, tree2)`, which is a
648 function that takes as input two ultrametric trees corresponding to the two clades, and returns
649 an object of class `PhenotypicModel`. It relies on the central function `aAGamma` that defines the
650 collection of (a_i, A_i, Γ_i) during each epoch.

651 This first version of the GMM implementation allows us to simulate tip data, to get the
652 tip distribution under any parameter set, and to fit tip data.

```
In [248]: modelGMMbis <- createModelCoevolution(tree1, tree2, keyword="GMMbis")
          modelGMMbis
```

```
Out[248]:
```

```
*****
```

```
*** Object of Class PhenotypicModel ***
```

```
*** Name of the model : [1] "GMMbis"
```

```
*** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S" "sigma"
```

```
*** Description : Generalist Matching Mutualism model.
```

```
Starts with 3 or 4 lineages having the same value  $X_0 \sim \text{Normal}(m_0, v_0)$ .
```

```
One trait in each lineage, all lineages evolving then non-independently  
according to the GMM expression.
```

```
*** Periods : the model is cut into 7 parts.
```

```
For more details on the model, call : print(PhenotypicModel)
```

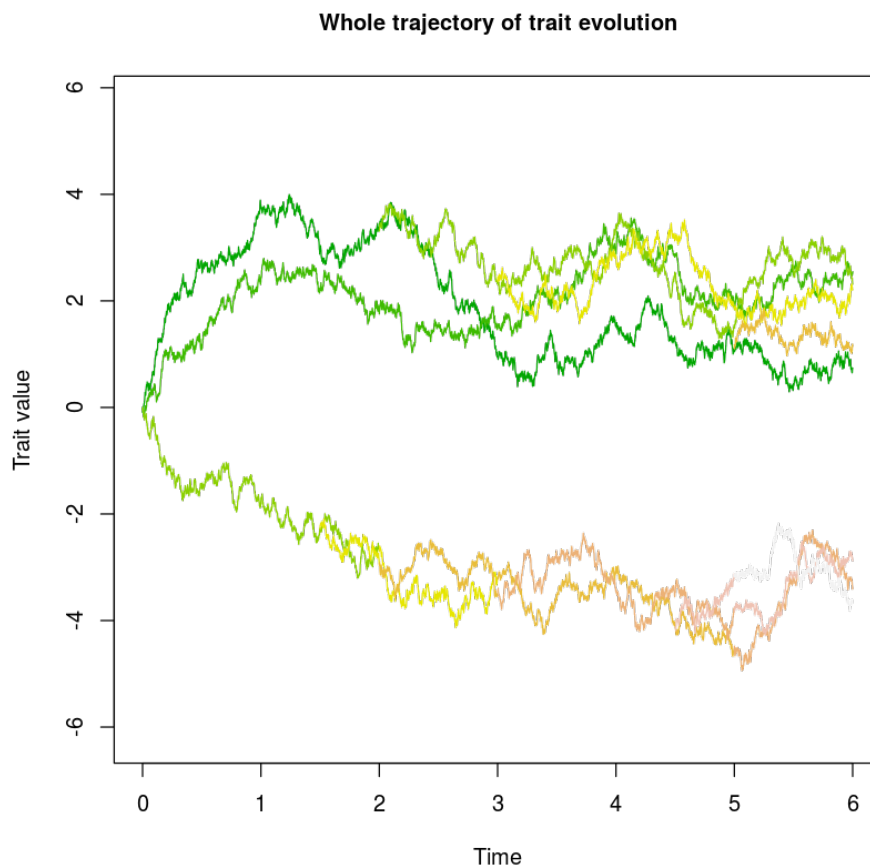
```
*****
```

```
In [249]: dataGMM <- simulateTipData(modelGMMbis, c(0,0,5,-5, 1, 1), method=2)
```

```
*** Simulation of tip trait values ***
```

```
Simulates step-by-step the whole trajectory, plots it, and returns tip data.
```

```
Computation time : 0.319762 secs
```



653

```
In [250]: getTipDistribution(modelGMMbis, c(0,0,5,-5,0.5,1))
```

654 Out [250]:

	A	2.493801
	E	2.493801
	C	2.493801
	D	2.493801
655 \$mean	B	2.493801
	X	-2.493801
	Z	-2.493801
	Y	-2.493801

		A	E	C	D	B	X	Z	Y
	A	2.196011	1.171214	1.215844	1.215844	1.563892	1.399735	1.341619	1.399735
	E	1.171214	2.141458	1.172730	1.172730	1.171214	1.337713	1.279597	1.337713
	C	1.215844	1.172730	2.199045	1.248832	1.215844	1.379237	1.321122	1.379237
656	\$Sigma	D	1.215844	1.172730	1.248832	2.199045	1.215844	1.379237	1.321122
	B	1.563892	1.171214	1.215844	1.215844	2.196011	1.399735	1.341619	1.399735
	X	1.399735	1.337713	1.379237	1.379237	1.399735	2.200083	1.190366	1.423215
	Z	1.341619	1.279597	1.321122	1.321122	1.341619	1.190366	2.158430	1.190366
	Y	1.399735	1.337713	1.379237	1.379237	1.399735	1.423215	1.190366	2.200083

```
In [251]: fitTipData(modelGMMbis, dataGMM, c(0,0,5,-5,1,1))
```

```
*** Fit of tip trait data ***
```

```
Finds the maximum likelihood estimators of the parameters,
```

```
returns the likelihood and the inferred parameters.
```

```
**WARNING** : This function uses the standard R optimizer "optim".
```

```
It may not always converge well.
```

```
Please double check the convergence by trying
```

```
distinct parameter sets for the initialisation.
```

```
Computation time : 3.728739 secs
```

```
657 Out[251]:
```

```
658 $value 6.61385667009296
```

```
659 $inferredParams m0 0.00512480151380221
```

```
660 v0 2.69680996514239e-05
```

```
661 d1 5.03536962882004
```

```
662 d2 -5.83517142115953
```

```
663 S 0.231631941480316
```

```
664 sigma 0.361942471141108
```

665 However, this first implementation relies on the `PhenotypicModel` class, which uses the
666 method `getTipDistribution` that solves the ODE system through each epoch, and thus takes
667 time.

668 The analytical reduction presented in Online Appendix C.4 can also be implemented. To
669 this end, we create a new class named `PhenotypicGMM`, associated with an other function
670 `getTipDistribution`. Using these developments allows us to compute more rapidly the tip
671 distribution under the model.

```
In [252]: modelGMM <- createModelCoevolution(tree1, tree2, keyword="GMM")
         modelGMM
```

Out [252]:

```
*****
*** Object of Class PhenotypicModel ***
*** Name of the model : [1] "GMM"
*** Parameters of the model : [1] "m0"      "v0"      "d1"      "d2"      "S"      "sigma"
*** Description : Generalist Matching Mutualism model.
Starts with 3 or 4 lineages having the same value  $X_0 \sim \text{Normal}(m_0, v_0)$ .
One trait in each lineage, all lineages evolving then non-independently
according to the GMM expression.
*** Periods : the model is cut into 7 parts.
For more details on the model, call : print(PhenotypicModel)
*****
```

```
In [253]: getTipDistribution(modelGMM, c(0,0,5,-5,0.5,1), v=TRUE)
         getTipDistribution(modelGMMbis, c(0,0,5,-5,0.5,1), v=TRUE)
```

```
*** Analytical computation of tip traits distribution ***
(Method working for the GMM model only)
Computation time : 0.0008528233 secs
```

672 Out [253]:

```

673 $mean
      A | 2.493803
      E | 2.493803
      C | 2.493803
      D | 2.493803
      B | 2.493803
      X | -2.493803
      Z | -2.493803
      Y | -2.493803

```

```

674 $Sigma
      | A      E      C      D      B      X      Z      Y
      |-----|
      A | 2.196010 1.171213 1.215843 1.215843 1.563890 1.399736 1.341620 1.399736
      E | 1.171213 2.141459 1.172730 1.172730 1.171213 1.337713 1.279597 1.337713
      C | 1.215843 1.172730 2.199045 1.248832 1.215843 1.379238 1.321122 1.379238
      D | 1.215843 1.172730 1.248832 2.199045 1.215843 1.379238 1.321122 1.379238
      B | 1.563890 1.171213 1.215843 1.215843 2.196010 1.399736 1.341620 1.399736
      X | 1.399736 1.337713 1.379238 1.379238 1.399736 2.200083 1.190366 1.423213
      Z | 1.341620 1.279597 1.321122 1.321122 1.341620 1.190366 2.158430 1.190366
      Y | 1.399736 1.337713 1.379238 1.379238 1.399736 1.423213 1.190366 2.200083

```

*** Computation of tip traits distribution through ODE resolution ***

(Method working for any model)

Computation time : 0.01734638 secs

```

675 Out[253]:

```

```

676 $mean
      A | 2.493801
      E | 2.493801
      C | 2.493801
      D | 2.493801
      B | 2.493801
      X | -2.493801
      Z | -2.493801
      Y | -2.493801

```

		A	E	C	D	B	X	Z	Y	
677	\$Sigma	A	2.196011	1.171214	1.215844	1.215844	1.563892	1.399735	1.341619	1.399735
		E	1.171214	2.141458	1.172730	1.172730	1.171214	1.337713	1.279597	1.337713
		C	1.215844	1.172730	2.199045	1.248832	1.215844	1.379237	1.321122	1.379237
		D	1.215844	1.172730	1.248832	2.199045	1.215844	1.379237	1.321122	1.379237
		B	1.563892	1.171214	1.215844	1.215844	2.196011	1.399735	1.341619	1.399735
		X	1.399735	1.337713	1.379237	1.379237	1.399735	2.200083	1.190366	1.423215
		Z	1.341619	1.279597	1.321122	1.321122	1.341619	1.190366	2.158430	1.190366
		Y	1.399735	1.337713	1.379237	1.379237	1.399735	1.423215	1.190366	2.200083