Let's formalise the discussion above.

The data we have about a tree can be considered as the matrix D, where D_{ij} is the value of the jth residue in the ith individual. For a given data set, D, we want to calculate the likelihood of a given tree, T, which is binary and has a leaf associated with each individual (each row of the matrix).

So we want $\mathcal{L}(T) = P(D|T)$. We saw that model each site as being independent of all other sites. That is, we consider

$$P(D|T) = \prod_{i=1}^{L} P(D_{:,i}|T).$$

But we saw that to calculate $P(D_{:,i}|T)$, we need to sum over all possible unknown values for the sites at the internal nodes. For a given tree, let $\mathbf{A}$ be the matrix of unknown ancestral sequences at each node of the tree, where each row corresponds to an ancestral node and each column to a site.

Then for each column i,

$$P(D_{:,i}|T) = \sum_{A_{:,i}} P(D_{:,i}, A_{:,i}|T).$$

That is, for each site we sum of all possible assignments of ancestral values to get the likelihood for that site. It turns out that this sum over all ancestral values is easy enough to calculate using a dynamic programming approach, so we just need to figure out how to calculate each element of the sum: $P(D_{:,i}, A_{:,i}|T)$.

We make one further assumption: that each lineage is independent. In that case, the probability of the tree with states known at every node, $P(D_{:,i}, A_{:,i}|T)$, becomes a simple product of the likelihoods of each lineage of the tree.

That means all we need is a model that gives us the probability of mutating from base a to base b over an arbitrary length of time (which is given by the branch length).

We have omitted one detail here, and that is the probability of observing the sequence of the most recent common ancestor of the full sample.

We will develop the method to make this calculation in the next Section and then come back to the problem of summing over all such possibilities.

21.2 Markov processes

The model we use is a continuous time Markov model (often called *Markov processes*). We haven't got time to go into this in much detail but the main idea behind it is relatively intuitive.

We've already seen a simple Markov process, in the form of the Poisson process. Recall that in the Poisson process, 'events' can occur at any time, times between events are exponentially distributed and, in a very period of time, we expect either to see no events or one event (but not two or more). In the Poisson process, we only considered a single type of event and simply counted them when they occurred so that the state of the process was just a count of the number of things that had occurred so was strictly increasing.

The model we use for mutations is similar to the Poisson process in that waiting times between events are exponential and, in a short slice of time, nothing or just one event will occur. The events are substitutions so we want to keep track of the state of the process at each time point. Let $X(t)$ the state of the process at time t (so $X(t) \in \{A, C, G, T\}$). The difference from a Poisson process is that we allow that the rate of different events occurring may be different. For example, the rate of mutations from state A to state G may be different from the rate at which A mutates to T.

Let q_{ab} be the instantaneous rate of transitions from state a to state b, for any states $a \neq b$. It may help to think of this in terms of flow: q_{ab} is the rate of flow form state a to state b.

q_{aa} is the total rate of flow out of a, so it is the sum of rates from a to b for $b \neq a$ and is defined to be negative, $q_{aa} = -\sum_{a \neq b} q_{ab}$. This means that the length of time spent in state a is exponentially distributed with rate $-q_{aa}$.

Now define the rate matrix Q to have off diagonal elements q_{ab} and diagonal elements $q_{aa} = -\sum_{a \neq b} q_{ab}$.

In the case of modelling DNA mutations, Q has the form

$$Q = \begin{pmatrix} -\sum_{j \neq A} q_{Aj} & q_{AC} & q_{AG} & q_{AT} \\ q_{CA} & -\sum_{j \neq C} q_{Cj} & q_{CG} & q_{CT} \\ a_{GA} & q_{GC} & -\sum_{j \neq G} q_{Gj} & q_{GT} \\ q_{TA} & q_{TC} & q_{TG} & -\sum_{j \neq T} q_{Tj} \end{pmatrix}.$$

where q_{ij}, $i \neq j$ can be interpreted as the instantaneous rate that i mutates to j.

Now define $P_{ab}(t) = \Pr(X(t) = b | X(0) = a)$ the probability of starting in state a at time 0 and being in state b at time t and form the matrix $P(t) = [P_{ab}(t)]$. It turns out that we can derive $P(t)$ from Q by taking the matrix exponential:

$$P(t) = \exp(Qt) = \sum_{k=0}^{\infty} \frac{(Qt)^k}{k!}.$$

The matrix exponential can sometimes be calculated analytically but, for general Q, use numerical methods which are available for many standard numerical libraries.

The matrix exponential follows the rules we would hope for the exponential function. For example:

- $\exp(\mathbf{0}) = \mathbf{I}_n$
- when $\mathbf{AB} = \mathbf{BA}$, $\exp(\mathbf{A} + \mathbf{B}) = \exp(\mathbf{A}) \exp(\mathbf{B})$

- When $\mathbf{B}$ is invertible, $\exp(\mathbf{BAB}^{-1}) = \mathbf{B}\exp(\mathbf{A})\mathbf{B}^{-1}$
- If $\mathbf{A} = \mathrm{diag}(a_1, \ldots, a_n)$, then $\exp(\mathbf{A}) = \mathrm{diag}(\exp(a_1), \ldots, \exp(a_n))$.

21.3 Models of sequence mutation

There are a lot of parameters to specify or estimate in a full model of sequence mutation: on the face of it, we need to specify each of the q_{ab}s (giving us 12 parameters when there are 4 bases). In reality, we make various simplifying assumptions to reduce the number of parameters and make estimation simpler.

We insist that models of mutation are *reversible* and *stationary.*

A process is stationary when , if run for long enough, it settles down to some equilibrium, $\pi = [\pi_A, \pi_C, \pi_G, \pi_T]$. Formally, π is defined by the equation $\pi P(t) = \pi$ for any t. That is, if the process starts in equilibrium, running the process further will leave it in equilibrium. π is called the equilibrium distribution or the stationary distribution.

A process is reversible when it looks the same running backwards as it does running forwards. Strictly, a process is reversible when it satisfies the detailed balance conditions $\pi_i p_{ij}(t) = \pi_j p_{ji}(t)$ for all values of i, j and t.

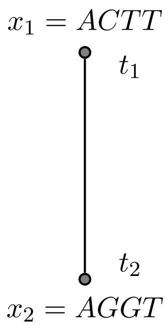
It helps when specifying substitution matrices to normalise them so that the average number of mutations per unit time is one. Given an unnormalised matrix $\hat{Q}$, it can be normalised by multiplying each entry by some constant β, so that the normalised matrix is $Q = \beta\hat{Q}$. The value of β depends on $\hat{Q}$.

Scaling the substitution matrix in this manner means that time is measured in substitutions. That is, one time unit corresponds to the time in which we would expect to see one mutation at any given site.

Summary of assumptions we make

- Each site is identical to all others in the evolutionary processes operating on it.
- Each site is free to change independently of all other sites.
- These two assumptions are usually stated as sites are independent and identically distributed, or i.i.d.
- Substitution probabilities do not change with time or over the tree. This is known mathematically as a homogeneous Markov process.
- The mutation process is time-reversible meaning that the process looks the same whether it is run forward or backward in time.
- Mutation process are independent across across different branches.

So given a mutation model (that is, given Q and μ), we can easily determine the probability of observing a descendant sequence given an ancestral sequence. For example, given the sequence $x_1 = ACTT$ at time t_1 and the $x_2 = AGGT$ at time t_2 and assuming



But given a tree with samples at the tips, the ancestral sequences are unknown to us. But how does this work when we don't know the ancestral sequences? We need to sum over all possible ancestral sequences. It turns out that this is relatively easy to achieve using a dynamic programming approach known as Felsenstein's peeling (or pruning) algorithm which performs this calculation in polynomial time. We omit the details.