

ABC ESTIMATION OF THE SUBSTITUTION RATE.
PRACTICAL EXERCISES 24/03/10

A) *Practical on the ABC rejection algorithm.*

Implement Rejection-ABC for Exp-Poisson example,

$$D_i \sim \text{Poisson}(\lambda), i = 1, 2, \dots, N.$$

1. Take $N = 5$ and simulate N iid Poisson variates with mean $\lambda = 0.5$. These will be your 'data'.

2. Now suppose we wish to simulate the posterior for (the unknown true value of) λ given these data D , using prior $p(\lambda) = \exp(-\lambda/\mu)/\mu$ with $\mu = 1$ and likelihood $p(D|\lambda) = \exp(-N\lambda) \prod_i \lambda^{D_i}/D_i!$.

Implement a function

```
function lambda=myabc(D,sn,mu)
```

taking as input your data D , your prior mean μ for λ and a threshold small number sn for the ABC (corresponding to ϵ below), and returning a sample λ from the posterior $\pi(\lambda|D) \propto p(\lambda)p(D|\lambda)$. Make a histogram of samples from the posterior dbn for λ .

Here is the algorithm for you to implement (with x a trial value for λ).

Aim: draw $X \sim \pi(x|D) \propto p(D|x)p(x)$ (approximately). Fix $\epsilon > 0$ and some measure of distance $d(D', D)$ between 'data sets'.

i. draw $z \sim p(z)$ and $D' \sim p(D'|z)$

ii. while $d(D', D) > \epsilon$ do

$z \sim p(z)$

$D' \sim p(D'|z)$

end

return $X = z$

Define the distance as $|\text{mean}(D) - \text{mean}(D')|$. Accept if that distance is sufficiently small.

B) *Practical on the coalescent and Finite Sites.*

In this question you can choose to implement the coalescent simulation (B.1 simulate a tree) or the finite sites simulation (B.2). Whichever you choose, you can get code for the complementary part from ABC-day webpage.

B.1 Implement a function

```
function s=coal(L)
```

taking as input the number of leaves, and simulating the coalescent in natural time units (ie one unit of time is N_e generations for a population of size N_e). Your code should return a tree, which I recommend you implement as an array $\mathbf{s}(1 : (2L - 1))$ of structures

```
s(i) = struct('name', i, 'seq', [], 'age', 0, 'child', []);
```

with $\mathbf{s}(i).\text{name} = i$, $\mathbf{s}(i).\text{seq}$ a $1 \times N$ base character sequence at this stage left [] empty, $\mathbf{s}(i).\text{age}$ is the age of node i on the tree, which you simulate via the coalescent, and $\mathbf{s}(i).\text{child}$ a 1×2 array giving the labels of the child nodes of this node (and empty if i is a leaf) which again are determined in the coalescent simulation. See the lecture notes for a description of an algorithm you might implement.

Test your code by verifying that the mean root time is equal $2(1 - 1/L)$. The `show()` function can be used to draw the trees you generate. It assumes the root is node $2L - 1$ (this is natural if you simulate the coalescent from the leaves up).

B.2a. Implement a function

```
function nseq=simulate(mu,t,oseq)
```

taking as input a rate `mu`[subs/time], a time interval `t` and a $1 \times N$ character array `oseq` (use `A = 1 C = 2` etc). Your program should simulate the substitution process acting on the sequence, and output a simulated sequence `nseq` evolved from `oseq` over the time `t`.

Implement this for the Jukes Cantor substitution process. At each site there is no change with probability $p = (1 - e^{-4\mu t/3})/4$ and otherwise choose one of the other three bases uniformly at random. Make this function as efficient as you can.

B.2b. Implement a function

```
function s=sequence(s,i,mu)
```

simulating the substitution process on the tree `s` (which you can simulate using the `coal()` code from Question B.1) starting from a given $1 \times N$ sequence `s(i).seq` at node `i` and then generating sequences `s(j).seq` at all nodes `j` below `i` on the tree. Use the `simulate()` code you wrote in B.2a to propagate the sequences down the tree.

Note that the input `mu`, the substitution rate, will have units [subs/ N_e gens] since the unit of time is N_e generations.

To check your code runs, simulate a tree using `coal()` and assign the sequence at the root node by simulating a uniform random sequence of N bases. Then call `s = sequence(s, 2*L-1, mu)` and inspect the matrix `seq = reshape([s(1:L).seq], N, L)'`, using something like `image(seq)`.

Note: this problem has a neat solution using a recursive implementation.

C) *Practical on ABC for the substitution rate from sequence data.*

1. The function `msp()`

```
function c=msp(x)
```

```
%function c=msp(x)
%c=mean number of distinct site patterns
%per site in L=leaves x n=sites sequence
%data x
```

```
[B,i,j]=unique(x','rows');
c=length(i)/length(j);
```

computes the mean number of distinct site patterns per site in the $L \times N$ sequence data matrix `x`. Show how this statistic may be used to define a distance suitable for estimation of the substitution rate μ via ABC.

2. Consider the two data sets `hivseq.mat` and `synth.mat`. If you load `hivseq ndata`; you get $L \times N$ data matrix `ndata`. Similarly if you load `synth sdata`; you get $L \times N$ data matrix `sdata`. Using ABC, estimate μ . For the synthetic data the

true value of μ was $\mu = 0.3$. Take an uniform prior for μ on the interval $0 \leq \mu \leq 1$ (see justification in lecture notes).

In order to answer this question, you need to write a program that simulates a value of μ from the prior (`mu=rand` in this case!) and uses `coal()` and `sequence(...,mu)` to simulate data for that simulated μ . Reject the μ -value if the distance between the `mSP()`-values for the real data and the simulated data are too great. Repeat this lots of times to get an approximate posterior distribution for μ .

Note that a tree-drawing programme `show.m` is available and may be useful if you have the right tree data structure.