AN INTRODUCTION TO GENETIC ALGORITHMS FOR NUMERICAL OPTIMIZATION

Solutions for Exercises of Section 2

In what follows $A = 30$ is the alphabet size, $S = 27$ is the sentence length, and p is the probability of a given letter to undergo mutation (which we loosely call *mutation rate*). And remember that we produce 10 copies of the current best sentence before throwing in mutations. A very useful trick when computing probabilities is the following: the probability of something happening is often easier to calculate as one minus the probability of that thing *not* happening.

Exercise 1:

(a) The probability of getting a given letter wrong is clearly $(A-1)/A$; therefore the probability P of getting *all* 27 letters wrong is

$$P(\text{all wrong}) = \left(\frac{A-1}{A}\right)^{27} = 0.4004 \qquad \text{for } A = 30, S = 27$$

(b) This one is almost trivial once you gave the result in (a); if P is the probability of getting all letters wrong, then $1-P$ is the (complementary) probability of *not* doing so, which is equal to the probability of getting *at least* one letter right:

$$P(\text{at least one right}) = 1 - \left(\frac{A-1}{A}\right)^{27} = 0.5996 \qquad \text{for } A = 30, S = 27$$

(c) This one is a bit trickier. The probability of getting one specific letter right is $1/A$; from (a), that of getting all 26 other letters wrong is $[(A-1)/A]^{26}$. There are S ways to get exactly one letter right (either the first is right, or the second is right, etc.); so the probabibility we're after is

$$P(\text{only one right}) = \frac{S}{A}\left(\frac{A-1}{A}\right)^{26} = 0.3728 \qquad \text{for } A = 30, S = 27$$

How's your head?

Exercise 2:

(a) The probability P we are after is equal to the probability P_1 of mutating the last incorrect letter to the correct letter in one of the ten sentences times probability P_2 of *not* mutating any of the other, currently correct letters in that sentence. Let's first work out individually those two probabilities.

In any one of the 10 trial sentences, the probability of a mutation taking place *and* in doing so producing the correct letter is p/A. The probability of this *not* happening in any of the ten new trial sentences is $(1-p/A)^{10}$, so that the probability of getting the needed mutation in *any one* of the ten trial sentences is

$$P_1 = 1 - (1 - p/A)^{10} \tag{2.1}$$

Now, the probability that *none* of the remaining 26 letters suffer a mutation is $(1-p)^{26}$, if we neglect the possibility that a mutation occurs but replaces an existing correct letter with itself (probability $1/A$). Therefore P_2 is just

$$P_2 = (1-p)^{26} \tag{2.2}$$

Therefore the probability P of getting that last letter right in one of the ten sentences is

$$P = P_1 P_2 = (1 - (1-p/A)^{10})(1-p)^{26} = 0.00256 \qquad \text{for}\, p = 10^{-2} \tag{2.3}$$

(b) Calculating the probability P of all sentences being degraded from 26 to 25 correct letters by mutation proceeds similarly. For a given sentence, this probability is the product of the probability P_1 of destroying one of the 26 currently correct letters times the probability P_2 of *not* improving the remaining incorrect letter. First let's do P_1:

(*i*) probability of hitting given correct letter is p

(*ii*) probability of *not* hitting given correct letter is $1-p$

(*iii*) probability of not hitting *any* correct letter is $(1-p)^{26}$

(*iv*) probability of hitting *any* correct letter is then: $P_1 = 1-(1-p)^{26}$

Now P_2; from Exercise 2(a), the probability of getting the last letter right in one sentence is p/A, so that the probability of this *not* happening is just $P_2 = (1-p/A)$. Therefore, the probability P of degrading *all ten* trial sentences is

$$P = (P_1 P_2)^{10} = 4.12 \times 10^{-7}$$

You should be happy to note that this probability is very much smaller than the probability worked out in part (a). Otherwise there would be no chance of ever getting a sentence with 27 correct letters. How's that headache?

Exercise 3:

Go back to Figures 6 and 7 in the Tutorial; getting those last few correct letters is really where most of the time is spent. How many generations does that take, on average? Well, think about it a bit and you'll find that the number of generations needed to go from 26 to 27 correct letters is basically the inverse of the probability we computed in Exercise 2(a), i.e., $(0.0256)^{-1} = 390$. Now go through the exact same logic as in Problem 2(a), but for the case of going from 25 to 26 correct letters; you get $P_1 = 1 - (1 - p/A)^{2\times 10} = 0.0066$ (the "2" in the exponent because there are now two ways to improve a sentence). The probability of none of the 25 correct letters being hit by mutation is now $(1 - p)^{25} = 0.778$, so that eq. (2.3) yields $P = 0.0051$. Thus this step can be expected to require 195 generations.

By this sort of inductive logic you can figure out that the probability of going from a sentence with currently k correct letters to at least one of ten mutated copies having $k + 1$ correct letters is

$$P_k = (1 - p)^k (1 - (1 - p/A)^{10(S-k)}), \tag{2.5}$$

from which it follows that the number n of generations required is

$$n = \sum_{k=1}^{S-1} \left[(1 - p)^k (1 - (1 - p/A)^{10(S-k)}) \right]^{-1} . \tag{2.6}$$

Computing this sum yields $n = 1435$; Figure 6 is a little faster than average, but not all that much.

Exercise 4:

This Exercise is in fact pretty much a straightforward generalization of Exercise 2. Given a mutation rate p, we are after the number of currently correct letters N such that the probability $P^{(+)}$ of improving any one of ten sentences is equal to the probability $P^{(-)}$ of damaging all ten sentences.

The first of these two probabilities is readily calculated following the same logic as for computing P_1 in Exercise (2); the result is

$$P^{(+)} = 1 - (1 - p/A)^{10(S-N)} \tag{2.7}$$

The second probability is also computed in exactly the same way as in part (b) of Exercise 2, with the exception that we now dealing with the probability of destroying any one of N currently correct letters in all 10 trial sentences.

Repeating steps (i) through (iv) of Exercise 2(b), the probability of hitting any one of the N currently correct letter is

$$P_1 = 1 - (1 - p)^N$$

while the probability of *not* mutating a currently incorrect to a correct letter is

$$P_2 = (1 - p/A)^{S-N}$$

so that the probability of degrading all 10 sentences is, as before,

$$P^{(-)} = (P_1 P_2)^{10} \tag{2.8}$$

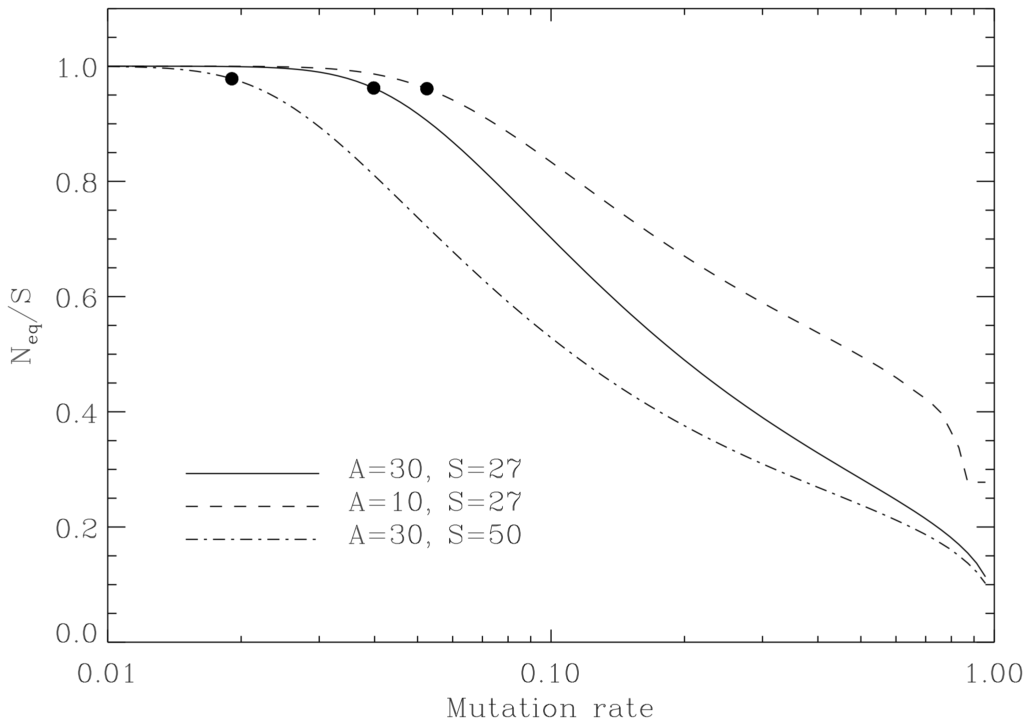

Figure 2.1: Equilibrium correct letter count $N_{\rm eq}$, normalized to sentence length S, as a function of mutation rate for a few different alphabet sizes (A) an sentence lengths. The solid dots correspond to the mutation rate leading to equilibrium one letter away from S. This represents an "optimal" mutation rate.

Equating eqs. (2.7) to (2.9):

$$\left[\left(1-(1-p)^N\right)\left(1-\frac{p}{A}\right)^{S-N}\right]^{10} = 1-\left(1-\frac{p}{A}\right)^{10(S-N)}. \tag{2.9}$$

For a given p, eq. (2.9) allows to compute an "equilibrium" error rate $N_{\rm eq}$. Don't waste time trying to solve analytically for N in eq. (2.9); you need to do it numerically. The bisection method (Press *et al.* 1992, §9.1) works just fine. The following Figure shows plots of $N_{\rm eq}/S$ vs p across a wide range of p's for a few distinct combinations of alphabet sizes A and sentence lengths S.

(b) The idea now is to have p that as large as possible to favor rapid convergence but still small enough to have $N_{\rm eq} \gtrsim S-1$, and so avoid leveling to a nonzero error plateau (as the dotted line on Fig. 7 of the Tutorial). This means setting $N=S-1$ in eq. (2.9), and solving for p. For $A=30$ and $S=27$, the corresponding mutation rate is $p \simeq 0.04$, and is indicated as solid dots on the solid curve of Fig. 2.1. The optimal mutation rate is also indicated in this way on Figure 2.1 for the other two cases considered. Note how the optimal p value is sensitive to the sentence length and alphabet size.

Exercise 5:

There are a number of unrealistic things with the example, in terms of an analog of biological evolution:

(1) We have available a target against which to measure the quality of our trial sentences. There is no such thing in evolution. However, it should be clear that the process will function as long as we are able to *rank* target sentences of a given generation with respect to one another, rather than against an external target of perfection. That's what natural selection does.

(2) Think of a sentence as a "chromosome" and each letters as a "gene". The contribution of a given gene (letter) to a sentence's "fitness" (score in correct letter) is completely independent from the contribution of other genes; this is certainly not the case in biology, where genes interact with one another in very complex ways (this "nonlinearity" is called epistasis in genetic parlance).

(3) Our sentences are "reproducing" asexually; some life forms do just that, but others (including ourselves) don't.

Paul Charbonneau, HAO/NCAR, August 1998.

E-mail questions or queries to: `paulchar@ncar.ucar.edu`