Editor: Dianne P. O'Leary, oleary@cs.umd.edu

BEETLES, CANNIBALISM, AND CHAOS

Analyzing a Dynamical System Model

By Dianne P. O'Leary

A system's evolution over time can be described with a set of equations called a dynamical system. In this installment, we use a dynamical system to model the life cycle of flour beetles to estimate key parameters that describe their behavior.

If you open a container of flour and see small red rods among the powder, you're probably looking at some confused beetles (*Tribolium confusum*) or red beetles (*Tribolium castaneum*), as pictured in Figure 1. These insects progress through several stages of life—egg (two to four days), larva (approximately 14 days), pupa (approximately 14 days), and adult (three years or more). Their life cycle is complicated by one additional fact: they're cannibalistic. Adults and larvae eat eggs, pupae, and immature adults, and adults also eat larvae.

By using a *dynamical system* to model the flour beetle's life cycle, we can estimate the parameters of biological significance that describe their behavior. At the same time, we illustrate some mathematical properties of dynamical systems.

The Model

To try to understand the beetle population's dynamics, entomologists have developed a beetle model with three stages:

- $L(t)$ is the number of (feeding) larvae at time t.
- $P(t)$ is the number of nonfeeding (large) larvae, pupae, and immature adults at time t.
- $A(t)$ is the number of mature adults at time t.

Next, we define five coefficients of interaction, setting all other ones to zero:

- $b > 0$ is the average number of larvae that each adult recruits (that is, tends).
- $0 \le \mu_L \le 1$ is the probability that a larva dies from something other than cannibalism.
- $0 \le \mu_A \le 1$ is the probability that an adult dies from something other than cannibalism.
- $e^{-c_{ea}A(t)}$ is the probability that an egg won't be eaten by an adult between time t and time $t + 1$.
- $e^{-c_{el}L(t)}$ is the probability that an egg won't be eaten by a larva between time t and time $t + 1$.
- $e^{-c_{pa}A(t)}$ is the probability that a pupa won't be eaten by an adult between time t and time $t + 1$.

The rate that adults eat larvae is small; we set it to zero. The model Brian Dennis and his colleagues[1] proposed is

$$L(t+1) = bA(t)\exp(-c_{ea}A(t) - c_{el}L(t)), \tag{1}$$
$$P(t+1) = L(t)(1 - \mu_L), \tag{2}$$
$$A(t+1) = P(t)\exp(-c_{pa}A(t)) + A(t)(1 - \mu_A). \tag{3}$$

We measure time t in 14-day units.

PROBLEM 1.

To get some experience, plot the populations L, P, and A for 100 days for three sets of data: $b = 11.6772$, $\mu_L = 0.5129$, $c_{el} = 0.0093$, $c_{ea} = 0.0110$, $c_{pa} = 0.0178$, $L(0) = 70$, $P(0) = 30$, $A(0) = 70$, and $\mu_A = 0.1$, 0.6, and 0.9. Describe the behavior of the populations in these three cases as if you were speaking to someone who isn't looking at the graphs.

Equilibria and Stability

It's interesting to determine *equilibria populations*, or values for the initial numbers of larvae, pupae, and adults for which the population remains constant. We denote these as A_{fixed}, L_{fixed}, and P_{fixed}. Of course, one such solution is the *extinction solution* of zero larvae, zero pupae, and zero adults. If $c_{el} = 0$, then Dennis and colleagues provide a nonzero solution, valid when $b > \mu_A/(1 - \mu_L)$:

$$A_{fixed} = \log(b(1 - \mu_L)/\mu_A)/(c_{ea} + c_{pa}), \tag{4}$$
$$L_{fixed} = bA_{fixed}\exp(-c_{ea}A_{fixed}), \tag{5}$$
$$P_{fixed} = L_{fixed}(1 - \mu_L). \tag{6}$$

Final Installment

This is the 24th and final installment of Your Homework Assignment. I have enjoyed writing these assignments, and I have especially enjoyed receiving reader comments. The columns will form the basis for 24 chapters of a new textbook, and additional projects will appear on my Web site (www.cs.umd.edu/users/oleary). I'm grateful to Francis Sullivan and Norman Chonacky, editors in chief of *Computing in Science & Engineering* during the past four years, for encouraging me to write this column. It was a pleasure to work with Jennifer Stout, senior editor, whose professionalism, skill, and equanimity were key to the column's success. It has been a great experience.

The equilibrium solution is called *stable* if a colony of beetles with initial population $A(0) \approx A_{fixed}$, $L(0) \approx L_{fixed}$, and $P(0) \approx P_{fixed}$ tends to approach these values as time passes; otherwise, the solution is *unstable*. Let $\mathbf{x}_t$ be a vector with elements $L(t)$, $P(t)$, and $A(t)$. Then our Equations 1 through 3 are

$$\mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t),$$

where

$$\mathbf{F}(\mathbf{x}_t) = \begin{bmatrix} bA(t)\exp(-c_{ea}A(t) - c_{el}L(t)) \\ L(t)(1-\mu_L) \\ P(t)\exp(-c_{pa}A(t)) + A(t)(1-\mu_A) \end{bmatrix}.$$

By Taylor series,

$$\mathbf{F}(\mathbf{x}_t) \approx \mathbf{x}_{fixed} + \mathbf{J}(\mathbf{x}_{fixed})(\mathbf{x}_t - \mathbf{x}_{fixed}),$$

where $\mathbf{J}(\mathbf{x}_{fixed})$ is the Jacobian of $\mathbf{F}$—that is, the 3×3 matrix of partial derivatives. Therefore,

$$\mathbf{x}_{t+1} \approx \mathbf{x}_{fixed} + \mathbf{J}(\mathbf{x}_{fixed})(\mathbf{x}_t - \mathbf{x}_{fixed}), \text{ so}$$

$$\mathbf{x}_{t+1} - \mathbf{x}_{fixed} \approx \mathbf{J}(\mathbf{x}_{fixed})(\mathbf{x}_t - \mathbf{x}_{fixed}), \text{ and}$$

$$\|\mathbf{x}_{t+1} - \mathbf{x}_{fixed}\| \approx \|\mathbf{J}(\mathbf{x}_{fixed})(\mathbf{x}_t - \mathbf{x}_{fixed})\| \le \|\mathbf{J}(\mathbf{x}_{fixed})\|\, \|\mathbf{x}_t - \mathbf{x}_{fixed}\|.$$

We conclude that the new point $\mathbf{x}_{t+1}$ tends to be closer to $\mathbf{x}_{fixed}$ than $\mathbf{x}_t$ is if all eigenvalues of the Jacobian, evaluated at $\mathbf{x}_{fixed}$, are inside the unit circle.

We can therefore label each equilibrium solution for our beetle problem as stable or unstable depending on whether the eigenvalues of the Jacobian matrix of the system

$$\mathbf{F}(L,P,A) = \begin{bmatrix} bA\exp(-c_{ea}A - c_{el}L) \\ L(1-\mu_L) \\ P\exp(-c_{pa}A) + A(1-\mu_A) \end{bmatrix}$$

all lie within the unit circle.

Figure 1. Red beetles. The red beetle's approximate length when fully grown is 3 mm. (Figure copyrighted by Alex Wild, used with permission; all rights reserved.)

Problem 2.

Let $\mu_L = 0.5$, $\mu_A = 0.5$, $c_{el} = 0.01$, $c_{ea} = 0.01$, and $c_{pa} = 0.01$. Plot A_{fixed}, L_{fixed}, and P_{fixed} for b = 1.0, 1.5, 2.0, …, 20.0. To compute these values for each b, use `fsolve`, started from the solution with c_{el} = 0, to solve the equations $\hat{\mathbf{F}}(\mathbf{x}) = \mathbf{F}(\mathbf{x}) - \mathbf{x} = \mathbf{0}$. Provide `fsolve` with the Jacobian matrix for the function $\hat{\mathbf{F}}$, and on your plot, mark the b values for stable equilibria with plus signs.

Stability and Bifurcation

Let's investigate stability a bit more. We know that when $b > \mu_A/(1-\mu_L)$, Equations 4 through 6 give a constant solution to our population model. This means that if we start the model with exactly these numbers of larvae, pupae, and adults, we expect the population at each time to remain constant. Let's see what happens numerically, though. We'll study the solution as a function of μ_A, with the other parameters set to a particular choice of values. We want to know whether the solution for large values of t is constant, periodic, or *chaotic* (with no regular pattern). To decide this, we make a *bifurcation diagram*: we run the LPA iteration for various values of μ_A and plot the last 100 values of the LPA iteration as a function of μ_A.

TOOLS

The genome of the red flour beetle was the first beetle genome sequenced (see www.hgsc.bcm.tmc.edu/projects/tribolium/). For more pictures of it and other insects, see Alex Wild's Web site (www.myrmecos.net).

The main text uses life span and population data from Robert A. Desharnais and Laifu Liu[1] and Stuart M. Bennett (www.the-piedpiper.co.uk/th7a.htm). The dynamical systems model we use comes from Brian Dennis, Robert A. Desharnais, J.M. Cushing, and R.F. Constantino.[2]

A text by Edward R. Scheinerman gives an excellent introduction to dynamical systems, stability, and bifurcation diagrams.[3]

Recently, the LPA model's developers have revisited the problem.[4]

References

1. R.A. Desharnais and L. Liu, "Stable Demographic Limit Cycles in Laboratory Populations of Tribolium Castaneum," *J. Animal Ecology*, vol. 56, no. 3, 1987, pp. 885–906.
2. B. Dennis et al., "Nonlinear Demographic Dynamics: Mathematical Models, Statistical Methods, and Biological Experiments," *Ecological Monographs*, vol. 63, no. 3, 1995, pp. 261–281.
3. E.R. Scheinerman, *Invitation to Dynamical Systems*, Prentice Hall, 1996.
4. R.F. Costantino et al., "Nonlinear Stochastic Population Dynamics: The Flour Beetle Tribolium as an Effective Tool of Discovery," *Advances in Ecological Research*, vol. 37, 2005, pp. 101–141.

PROBLEM 3.

(a) Let $\mu_L = 0.5128$, $c_{el} = 0.0$, $c_{ea} = 0.01$, and $c_{pa} = 0.09$. For $\mu_A = 0.02, 0.04, \ldots, 1.00$, use the LPA relations in Equations 1 through 3 to determine the population for 250 cycles. On a single graph, plot the last 100 values as a function of μ_A to produce the bifurcation diagram.

(b) Determine the largest of the values $\mu_A = 0.02, 0.04, \ldots, 1$ for which the constant solution is stable (that is, well-conditioned).

(c) Explain why the bifurcation diagram isn't just a plot of L_{fixed} versus μ_A when the system is unstable.

(d) Give an example of a value of μ_A for which nearby solutions cycle between two fixed values. Give an example of a value of μ_A for which nearby solutions are chaotic (or at least have a long cycle).

Nurturing vs. Cannibalism: Estimating the Parameters

Now that we understand some properties of our dynamical system model $\mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t)$, let's use some observed data to try to determine the parameter values. Robert Desharnais and Laifu Liu[2] observed four colonies of red beetles for 266 days, making observations every 14 days. With least squares, we can estimate the six parameters in our model using this data. Aside from the initial values $L(0)$, $P(0)$, and $A(0)$, Desharnais and Liu give us three data values $L_{observed}(t)$, $P_{observed}(t)$, and $A_{observed}(t)$ for each time $t = 1, \ldots, 19$. Given values of the six parameters in our model, we can compute predicted values of the populations at each of these times, so we want to determine the parameters that minimize the difference between prediction and observation. Because Desharnais and Liu tell us that errors in the logs of the observed values are approximately equal, we minimize the least-squares function

$$\sum_{t=1}^{19} (\log(L_{observed}(t)) - \log(L_{predicted}(t)))^2 + \sum_{t=1}^{19} (\log(P_{observed}(t)) - \log(P_{predicted}(t)))^2 + \sum_{t=1}^{19} (\log(A_{observed}(t)) - \log(A_{predicted}(t)))^2,$$

where $L_{predicted}(t)$, $P_{predicted}(t)$, and $A_{predicted}(t)$ denote the values obtained from Equations 1 through 3.

PROBLEM 4.

(a) Use `lsqnonlin` to solve the least-squares minimization problem, using each of the four sets of data in `beetledata.m`. In each case, determine the six parameters (μ_L, μ_A, c_{el}, c_{ea}, c_{pa}, and b). Set reasonable upper and lower bounds on the parameters and perhaps start the least-squares iteration with the guess $\mu_L = \mu_A = 0.5$, $c_{el} = c_{ea} = c_{pa} = 0.1$, and $b = 10$. Print the solution parameters and the corresponding residual norm.

(b) Compare your results with those that Desharnais and Liu computed (see `param_dl` in `beetledata.m`). Be sure to include a plot that compares the predicted values with the observed values.

When I asked 25 students to solve Problem 4 with the data from the second colony of beetles, they obtained 13 different answers, all of them different from mine! Unfortunately, none of them gave a good fit to the measured data. When solving a nonlinear least-squares problem, it's important to realize that the function might be *nonconvex*, which means it might have many local minimizers. This makes it quite difficult for an optimization routine such as lsqnonlin to find the globally optimal solution. For difficult optimization problems, it sometimes helps to use a *homotopy* method. In its simplest form, this just means that we repeatedly use the known solution to one problem as a starting guess to solve a difficult nearby problem.

PROBLEM 5.

Consider the data for the second beetle colony. For each value $b = 0.5, 1.0, \ldots, 50.0$, minimize the least-squares function by using lsqnonlin to solve for the five remaining parameters. Plot the square root of the least-squares function versus b, and determine the best set of parameters. How sensitive is the function to small changes in b?

Perform further calculations to estimate the *forward error*—how sensitive the optimal parameters are to small changes in the data—and the *backward error*—how sensitive the model is to small changes in the parameters.

Dennis and his colleagues evaluated the LPA model using a less demanding criterion: they just compared the model's one-step predictions with the true values. (This is akin to a local error evaluation for an ordinary differential equation model; we just ask how much error is produced in a single step, assuming that correct values were given at the previous step.) It would be interesting to repeat the sensitivity analysis under this error criterion.

Acknowledgments

I'm grateful to David E. Gilsinn for helpful comments on this homework assignment.

References

1. B. Dennis et al., "Nonlinear Demographic Dynamics: Mathematical Models, Statistical Methods, and Biological Experiments," *Ecological Monographs*, vol. 63, no. 3, 1995, pp. 261–281.
2. R.A. Desharnais and L. Liu, "Stable Demographic Limit Cycles in Laboratory Populations of Tribolium Castaneum," *J. Animal Ecology*, vol. 56, no. 3, 1987, pp. 885–906.

Partial Solution to Last Issue's Homework Assignment

MONTE CARLO MINIMIZATION AND COUNTING

One, Two, …, Too Many

By Isabel Beichl, Dianne P. O'Leary, and Francis Sullivan

In the last issue's installment of Your Homework Assignment, we looked at Monte Carlo methods for difficult numerical problems. Specifically, we studied three uses of Monte Carlo methods—for function minimization, discrete optimization, and counting.

PROBLEM 1.

Consider the function myf.m (on the Web site; www.computer.org/cise/homework/), with domain $0 \le x \le 7$.

(a) Generate 500 uniformly distributed points on the interval [0, 7] in Algorithm 1, using fmincon for the local minimizer. Make a graph illustrating the minimizer corresponding to each starting point.

(b) $L = 90.3$ is a Lipschitz constant for the function myf.m. Use Algorithm 2 on the interval [0, 7]. Compare the two methods' performance.

(c) (Extra) Try Monte Carlo minimization on your favorite function of n variables for $n > 1$.

Answer:

The programs myfmin.m and myfminL.m on the Web site solve this problem but don't make the graph.

PROBLEM 2.

Use the simulated annealing algorithm to minimize myf.m. Experiment with various choices of T, α, and ε. Describe your experiment and the conclusions you can draw about how to choose parameters to make the method as economical and reliable as possible.

Partial Answer:

The program sim_anneal.m on the Web site is one im-

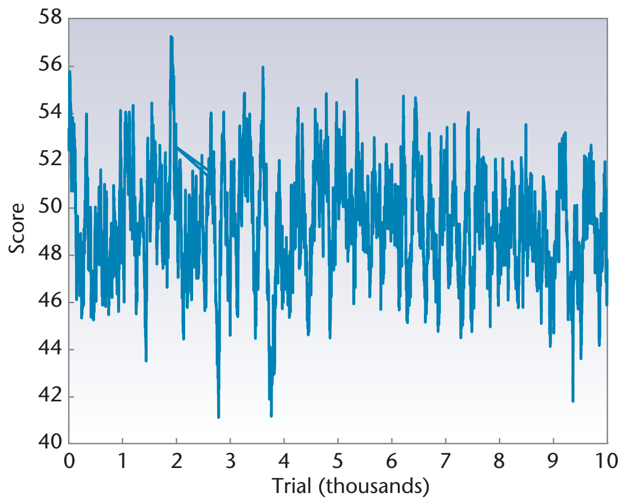

Figure A. Temperature = 1.0 no cooling. Moves are random.

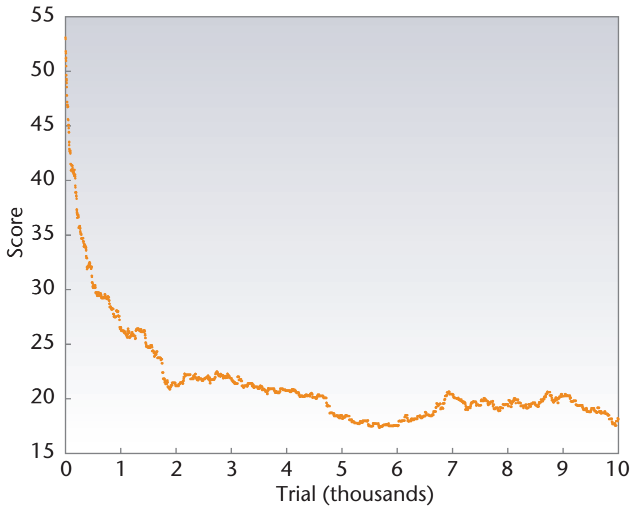

Figure B. Temperature = 0.1 no cooling. Score improves over time.

plementation of simulated annealing, and it can be run by using `problem1_and_2.m`. To finish the problem, experiment with the program. Be sure to measure reliability as well as cost, and run multiple experiments to account for the fact that the method is randomized. Also comment on the number of runs that converge to $x = 1.7922$, which is a local minimizer with a function value not much worse than the global minimizer.

PROBLEM 3.

(a) Matlab provides a naive Monte Carlo solution algorithm for the traveling salesperson problem (TSP). Given an ordering of the cities, it randomly generates a pair of cities and interchanges them if this interchange lowers the total distance. Experiment with the demonstration program `travel.m` and display the program using `type travel` to see how this works.

(b) Write a program to solve a TSP using simulated annealing, and compare your algorithm with that used in part (a).

Answer:

(a) Experiments with the Matlab `travel` code show that it works for up to 50 cities but, as is to be expected, slows down for larger sets. It's interesting and important to note that the solution is always a tour that doesn't cross itself. We'll return to this point shortly.

(b) Figures A through C show the results of simulated annealing for 100 random locations, with temperature $T = 1$, 0.1, and 0.01. Figures D and E show the actual tours for $T = 0.1$ and $T = 0.01$. Note that the result for 0.01 looks pretty good but not that much better than the output for $T = 0.1$, yet the $T = 1$ result looks completely random and gets nowhere near a low-cost tour. This demonstrates that lowering the temperature really does give a better approximation. However, because the $T = 0.01$ tour crosses itself, we know that it's still not the true solution. And we don't know the true minimum score (distance) or an algorithm for setting and changing T. Figuring out how to vary T is called *determining the cooling schedule*. We generally want to use a lower value of T as the solution is approached. The idea is to avoid a move that would bounce away from the solution when we're almost there.

How we design a cooling schedule depends on our analysis of the problem at hand. Some general techniques exist, but cooling schedules are still an active research topic. Typing the phrase "TSP, simulated annealing, cooling schedule" into Google Scholar gives more than 500 pointers to research publications; a query to the main Google site gives more than 12,000 hits.

The most popular general approach for setting a cooling schedule is to change T whenever a proposed move is accepted. Suppose that the initial temperature is T_0 and a proposed move is finally accepted after k trials. Then the temperature is reset to $T_0/\log(k)$. The idea behind the use of $\log(k)$ in the denominator is that the number of trials k required before generating a random number less than $\exp(-1/T)$ is $\exp(1/T)$ on average, so $1/T$ should look something like $\log(k)$. This is the famous logarithmic cooling schedule.[1]

Figures F, G, and H illustrate the application of simulated annealing with a logarithmic cooling schedule to a TSP with 100 random locations. The first two graphs show how the score evolves over 10,000 trials at a low temperature. Note that not many proposed moves that increase the score are accepted and that the score doesn't improve very much. The last is a picture of the best tour obtained. Because it crosses itself, it's not the optimal tour—getting that requires more computation and/or more sophisticated cooling schedules. Solving the TSP for 100 random locations is really quite difficult!

If you think this use of simulated annealing to attack the TSP seems quite informal and heuristic rather than analytic, you're right. In fact, some have argued that simulated annealing isn't really an optimization method but rather a collection of heuristic techniques that help in some cases. However, there's an important recently discovered connection between the central idea of simulated annealing and the use of Monte Carlo to approximate solutions to NP-hard problems, including determining the volume of a bounded convex region in E^n.

Suppose that K is the set in question and we want to determine $Vol(K)$. If n is large, this can be a very hard problem. The most well-developed approach is to define a sequence of convex sets

$$K_0 \subset K_1 \subset K_2 \subset \ldots \subset K_m = K,$$

where $Vol(K_0)$ is easy to evaluate. For each i, perform a random walk in K_i and count how many walkers happen to be in K_{i-1}. This gives an estimate of $Vol(K_{i-1})/Vol(K_i)$ and the product of these estimates for all i is an estimate for $Vol(K)$.

The connection to simulated annealing comes in a couple of ways. For one thing, the random walk can be done by using a Metropolis algorithm with a different rejection rate (that is, a different temperature) for each i. A more recent idea is to recognize that the volume is the integral of the characteristic function of the set K so we can try to approach this integral by integrating a sequence of other, easier functions instead. In particular, we can embed the problem in E^{n+1} by adding an extra coefficient x_0 to the points in E^n and then choose functions $f_0 < f_1 < f_2 < \ldots f_m$, where f_m is the characteristic function of K but the others look like $\exp(-x_0/T)$ in the extra coefficient, x_0.

Another example of simulated annealing is the KRS algorithm (named for its creators, Claire Kenyon, Dana Randall, and Alistair Sinclair). Those who have become fascinated by this subject might want to try to identify the "temperature" in this case to understand why KRS is a form of simulated annealing.

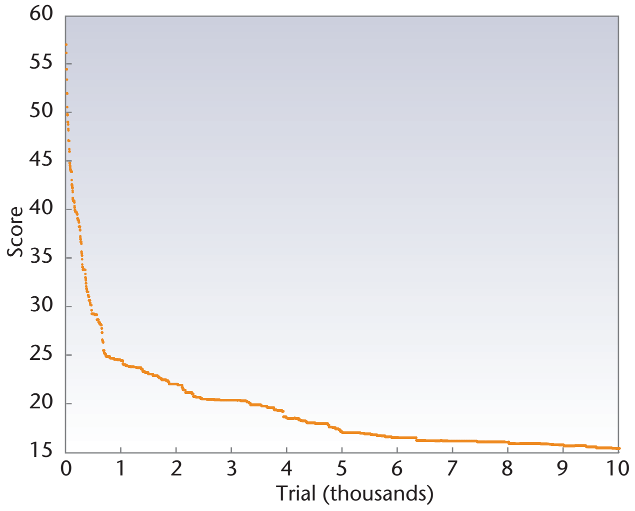

Figure C. Temperature = 0.01 no cooling. Slightly better score.

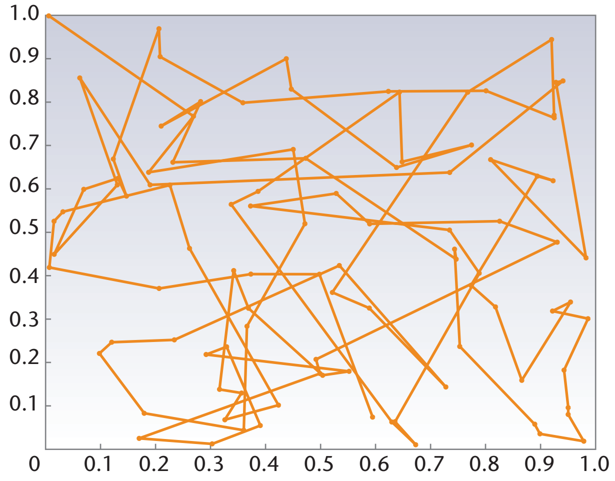

Figure D. Actual tour $T = 0.1$ no cooling.

Problem 4.

(a) Compute the partition function coefficients for a 2×2 lattice by explicit counting. Repeat for a 3×2 lattice. Consider the 4×4 case, and see why it becomes difficult to explicitly count the number of arrangements.

(b) Implement the KRS algorithm, and use it to estimate the partition function for a 4×4 lattice. Try various choices of probabilities and updating intervals. Repeat for a lattice as large as possible (perhaps 10×10).

Partial Answer:

(a) Table 1 gives some explicit counts, some done by hand and some by Thomas DuBois's `latticecount.m`.

Table 1. Partial answer to Problem 4.

	C(0)	C(1)	C(2)	C(3)	C(4)	C(5)	C(6)	C(7)	C(8)
2×2	1	4	2						
2×3	1	7	11	3					
3×3	1	12	44	56	18				
4×4	1	24	224	1,044	2,593	3,388	2,150	552	36
6×6	1	60	1,622	26,172	281,514	2,135,356	11,785,382	48,145,820	146,702,793

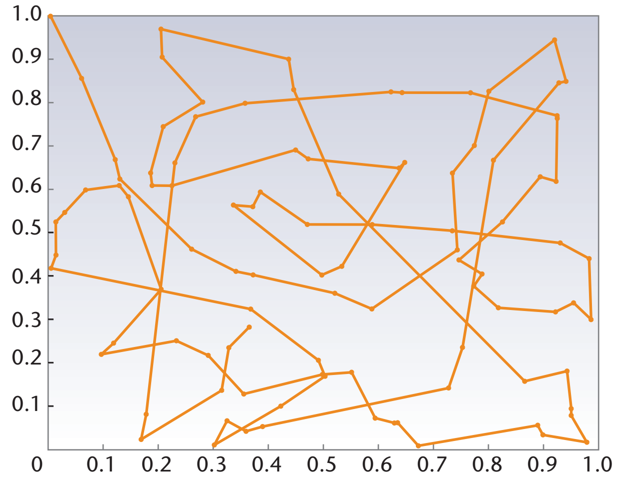

Figure E. Actual tour $T = 0.01$ no cooling.

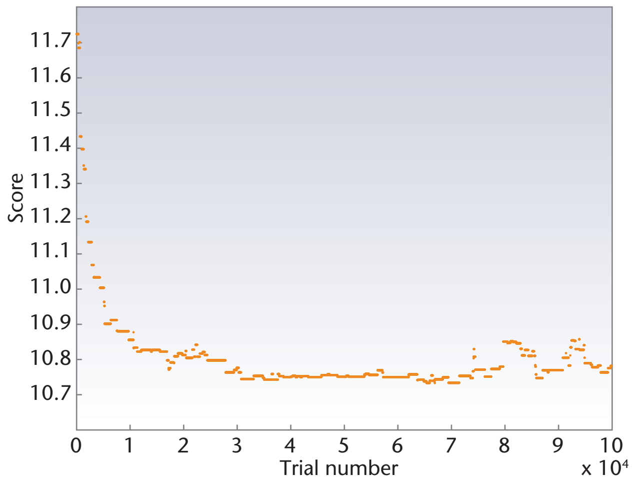

Figure F. Score evolution. Logarithmic schedule for 10,000 steps.

(b) One of the more interesting programming issues in this problem is the *data structure*:

- If we keep track of each *edge* of the lattice, then we need to enumerate rules for deciding whether two edges can be covered at the same time. In our 2×2 lattice, for example, we can't simultaneously have a dimer on the top edge and one on the left edge.
- If we keep track of each *node* of the lattice, then we need to know whether a dimer occupies it, so our first idea might be to represent a monomer by a 0 and a dimer by a 1. But we need more information—whether its dimer partner is above, below, left, or right. Without this additional information, the array

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

tells us that the 2×2 lattice has two dimers on it, but we can't tell whether they're horizontal or vertical.
- A third alternative is to keep track of both edges and nodes. Think of it as a *matching* problem, in which you can match each node with any of its four neighbors in a dimer, or it can be a monomer. We maintain an array of nodes, where the *j*th value is 0 if the node is a monomer, and equal to *k* if (*k*, *j*) is a dimer. We store the edges in an $n^2 \times 4$ array, in which the row index indicates the node at the beginning of the edge, and the entry in the array records the node at the end. Thus, each physical edge has two entries in the array (in rows corresponding to its two nodes), and a few of the entries at the edges are 0 because some nodes have fewer than four edges. We can generate a KRS change by picking an edge from this array, and we update the node array after we decide whether to consider an addition, deletion, or swap.

The program `KRS.m` by Sungwoo Park on the Web site is an efficient implementation of the second alternative. Figure I shows some sample results.

The original paper provides information on how to set the parameters to KRS.[2] Kenyon, Randall, and Sinclair showed that the algorithm samples well if both the number of steps and the interval between records are very large, but, in practice, the algorithm is considerably less sensitive than the analysis predicts.

Disclaimer

Mention of commercial products does not imply endorsement by NIST.

References

1. D. Bertsimas and J. Tsitsikls, "Simulated Annealing," *Statistical Science*, vol. 8, no. 1, 1993, pp. 10–15.
2. C. Kenyon, D. Randall, and A. Sinclair, "Approximating the Number of Monomer-Dimer Coverings of a Lattice," *J. Statistical Physics*, vol. 83, nos. 3 and 4, 1996, pp. 637–659.

Isabel Beichl is a mathematician in the Information Technology Laboratory at the National Institute of Standards and Technology. Contact her at isabel.beichl@nist.gov.

Dianne P. O'Leary is a professor of computer science and a faculty member in the Institute for Advanced Computer Studies and the Applied Mathematics Program at the University of Maryland. She has a BS in mathematics from Purdue University and a PhD in computer science from Stanford. O'Leary is a member of SIAM, the ACM, and AWM. Contact her at oleary@cs.umd.edu; www.cs.umd.edu/users/oleary/.

Francis Sullivan is the director of the IDA Center for Computing Sciences in Bowie, Maryland. From 2000 through 2004, he served as *CiSE* magazine's editor in chief. Contact him at fran@super.org.

Circulation: *Computing in Science & Engineering* (ISSN 1521-9615) is published bimonthly by the AIP and the IEEE Computer Society. IEEE Headquarters, Three Park Ave., 17th Floor, New York, NY 10016-5997; IEEE Computer Society Publications Office, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1314, phone +1 714 821 8380; IEEE Computer Society Headquarters, 1730 Massachusetts Ave. NW, Washington, DC 20036-1903; AIP Circulation and Fulfillment Department, 1NO1, 2 Huntington Quadrangle, Melville, NY 11747-4502. 2007 annual subscription rates: $45 for Computer Society members (print plus online), $76 (sister society), and $100 (individual nonmember). For AIP society members, 2007 annual subscription rates are $45 (print plus online). For more information on other subscription prices, see www.computer.org/subscribe/ or https://www.aip.org/forms/journal_catalog/order_form_fs.html. Computer Society back issues cost $20 for members, $96 for nonmembers; AIP back issues cost $22 for members.

Postmaster: Send undelivered copies and address changes to *Computing in Science & Engineering*, 445 Hoes Ln., Piscataway, NJ 08855. Periodicals postage paid at New York, NY, and at additional mailing offices. Canadian GST #125634188. Canada Post Corporation (Canadian distribution) publications mail agreement number 40013885. Return undeliverable Canadian addresses to PO Box 122, Niagara Falls, ON L2E 6S8 Canada. Printed in the USA.

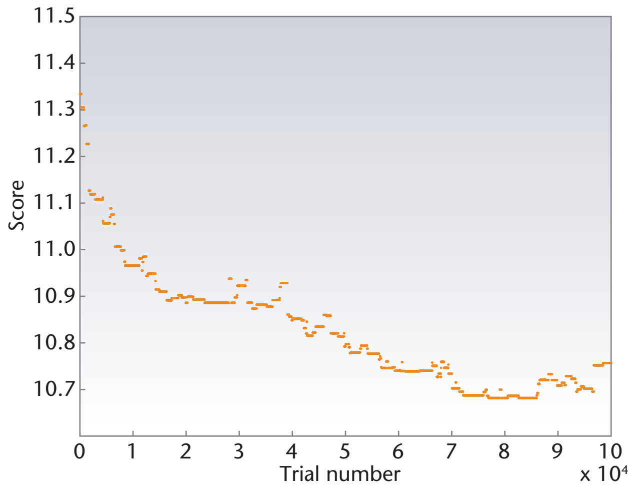

Figure G. Score evolution, repeating the experiment from Figure F.

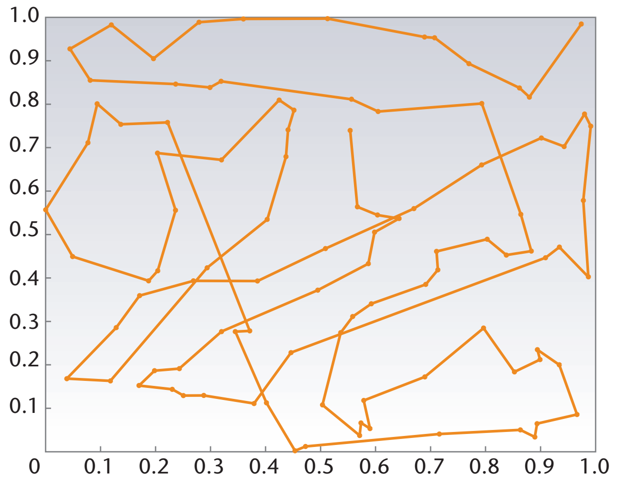

Figure H. Best tour obtained via simulated annealing, logarithmic cooling.

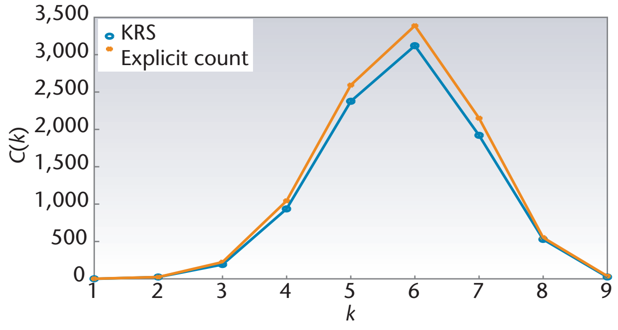

Figure I. Counts obtained by the KRS algorithm and by explicit counting for a 4×4 lattice. For KRS, we set the probabilities to 0.5, the number of steps between records to $\ell = 4$, and the total number of steps to 10^5. Because ℓ was so small, the samples were highly correlated, but the estimates are still quite good.