Tools

The Matlab function `ode23s` provides a good solver for Problem 1's ordinary differential equations (ODEs). Most ODE software provides a mechanism for stopping the integration when some quantity goes to zero; in `ode23s`, using the Events property in an option vector accomplishes this. Charles van Loan's book[1] provides a good introduction to the numerical solution of ODEs; more specialized texts cover the reasons for preferring a *stiff* solver like `ode23s` for certain types of ODEs.[2]

For Problem 2, we can use ODE software, including `ode23s`, to solve certain differential algebraic equations (DAEs); in Matlab, using the `Mass` property in the option vector accomplishes this. Model 2 is a very simple DAE; Kathryn Brenan, Steven Campbell, and Linda Petzold's book provides more information on the theory and solution of such problems.[3]

Delay differential equations (DDEs) such as those in Problem 3 arise in many applications, including circuit analysis. To learn more, consult a text such as Richard Bellman and Kenneth Cooke's book[4] or Jack Hale and Sjoerd Lunel's book.[5] In Matlab (Release 13), we can solve certain DDEs by using `dde23`.

Stochastic differential equations are an active research area. Desmond Higham[6] gives a good introduction to computational aspects and supplies references for further investigation.

Model 1 is Kermack and McKendrick's SIR model, first introduced in 1927. Nicholas Britton discusses it in more detail.[7]

James Callahan presents the differential equations leading to Model 4,[8] by following a model with one space dimension given in an older text.[9]

References

1. C.F. van Loan, *Introduction to Scientific Computing*, Prentice Hall, 2000.
2. U.M. Ascher and L.R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM Press, 1998.
3. K.E. Brenan, S.L. Campbell, and L.R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations, Classics in Applied Mathematics 14*, SIAM Press, 1995.
4. R. Bellman and K.L. Cooke, *Differential-Difference Equations*, Academic Press, 1963.
5. J.K. Hale and S.M. Verduyn Lunel, *Introduction to Functional Differential Equations*, Springer-Verlag, 1993.
6. D.J. Higham, "An Algorithmic Introduction to Numerical Simulation of Stochastic Differential Equations," *SIAM Rev.*, vol. 43, 2001, pp. 525–546.
7. N.F. Britton, *Essential Mathematical Biology*, Springer, 2003.
8. J. Callahan, *The Spread of a Contagious Illness*, Smith College; www.math.smith.edu/~callahan/ili/pde.html.
9. D.S. Jones and B.D. Sleeman, *Differential Equations and Mathematical Biology*, George Allen and Unwin, 1983.

Partial Solution to Last Issue's Homework Assignment

Models of Infection: Person to Person

By Dianne P. O'Leary

We have mn patients in a hospital ward, and one of them becomes infected. We track $I(t)$, the proportion of the infected population; $S(t)$, the proportion of the population that never has been infected, and $R(t)$, the remaining proportion. We let τ be the probability of being infected by a sick neighbor.

Problem 1. Run the model for $m = n = 10$, $k = 4$, and $\tau = 0.2$ until there are no infected patients. Plot $I(t)$, $S(t)$, and $R(t)$ in a single graph. If possible, display the epidemic as a movie, where each pixel's color represents a patient's state.

Problem 2. Next, we add a probability δ of patients being moved to a different bed. Modify your model to include mobility and run it for $\delta = 0.01$ until no infected patients remain. Display the results as in Problem 1.

Problem 3. Suppose that each day, each susceptible individual has a probability ν of being vaccinated. Rerun your model with $\nu = 0.1$ until no infected patients remain. Display the results as in Problem 1, and then compare the three models' results.

Answer: Figure 1 shows the simulation results for each of these three models. (The Matlab program that generated the results is at www.computer.org/cise/homework.) Generally, mobility increases the infection rate and vaccination dramatically decreases it. In our sample runs, the infection peaks

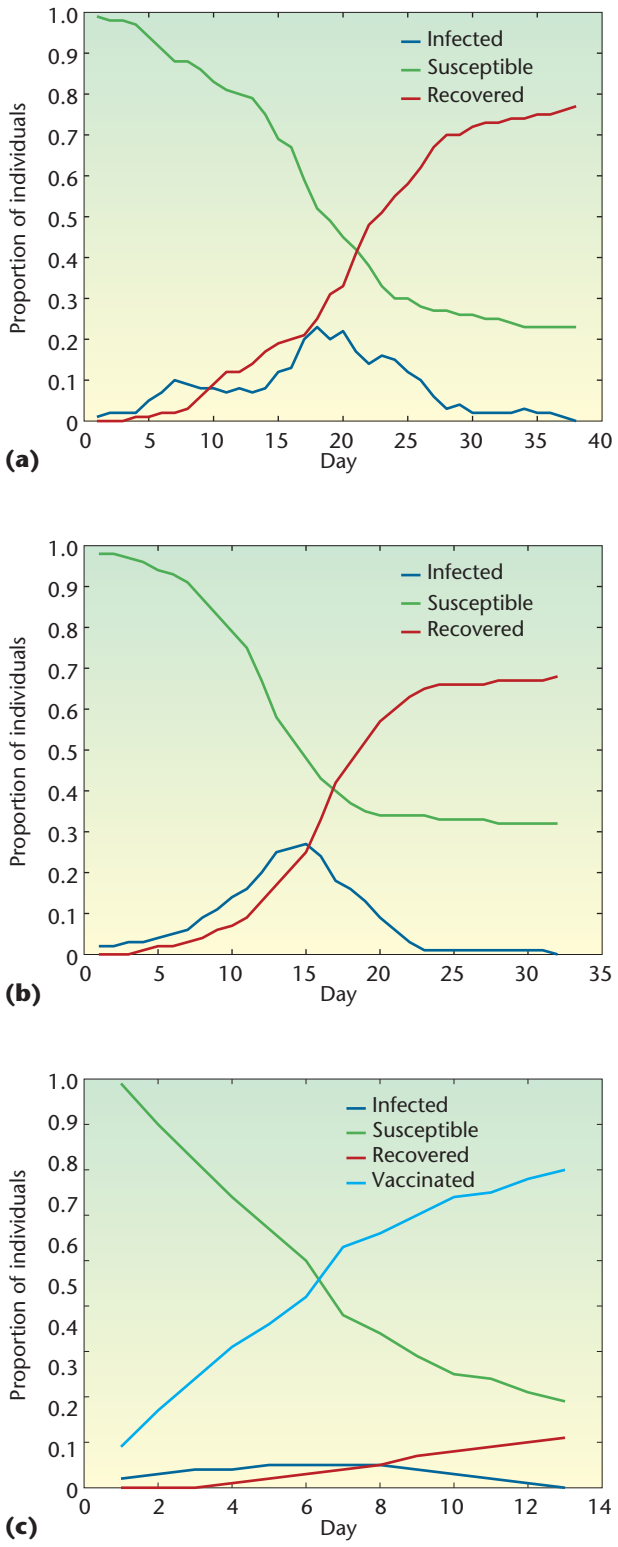

Figure 1. Proportion of individuals infected by day in a 10 × 10 grid of hospital beds. (a) The infection rate $\tau = 0.2$, (b) the infection rate $\tau = 0.2$ and mobility rate $\delta = 0.01$, and (c) the infection rate $\tau = 0.2$, mobility rate $\delta = 0.01$, and vaccination rate $\nu = 0.1$.

around day 18 with no mobility, and around day 15 when patients are moved. Individual runs might vary, though.

Problem 4. Run Problem 3's model 1,000 times, recording the number of individuals who become infected in each run. (This is equal to the number of recovered individuals when the run is terminated.) Plot this data as a histogram, and then compute the mean number of recovered individuals and the variance in this number. Try several different values of ν to see whether the variance changes.

Answer: Figure 2 shows the histograms for $\nu = 0$, 0.1, 0.2, and 0.3. The mean percent of the population infected drops from 73.6 percent for $\nu = 0$ (with a variance of 4.5 percent) to 4.1 percent for $\nu = 0.3$ (with a variance of only 0.06 percent).

Problem 5. Develop a vaccination strategy that will, on average, limit the epidemic to 20 percent of the population. Do this by using a nonlinear equation solver to solve the problem $R(\nu) - 0.2 = 0$, where $R(\nu)$ is the mean number of recovered individuals when we use a vaccination rate of ν. For each value of ν the solver presents, you will need to get a reliable estimate of R by running the model multiple times. Use Problem 4's variance estimates to determine how many runs to use, and then justify your choice.

Answer: From Problem 4, we know that a very low vaccination rate (somewhat less than nu = 0.1) is sufficient to dramatically reduce the infection rate. But using a nonlinear equation solver on a noisy function is quite dangerous; it is easily fooled by outliers, and changing the starting guess, you can make it produce almost any value.

Problem 6.

a. Construct the *transition matrix* $\boldsymbol{A}$ corresponding to this Markov chain: element $a_{i,j}$ is the probability of transitioning to state i from state j.

b. Let $\mathbf{e}_1$ be the column vector with 1 in position 1 and zeroes elsewhere. If we begin in day one in the first state, then vector $\boldsymbol{A}\mathbf{e}_1$ tells us the probabilities of being in each of the states on day two. Prove this.

c. Similarly, $\boldsymbol{A}^2\mathbf{e}_1$ gives the probabilities for day

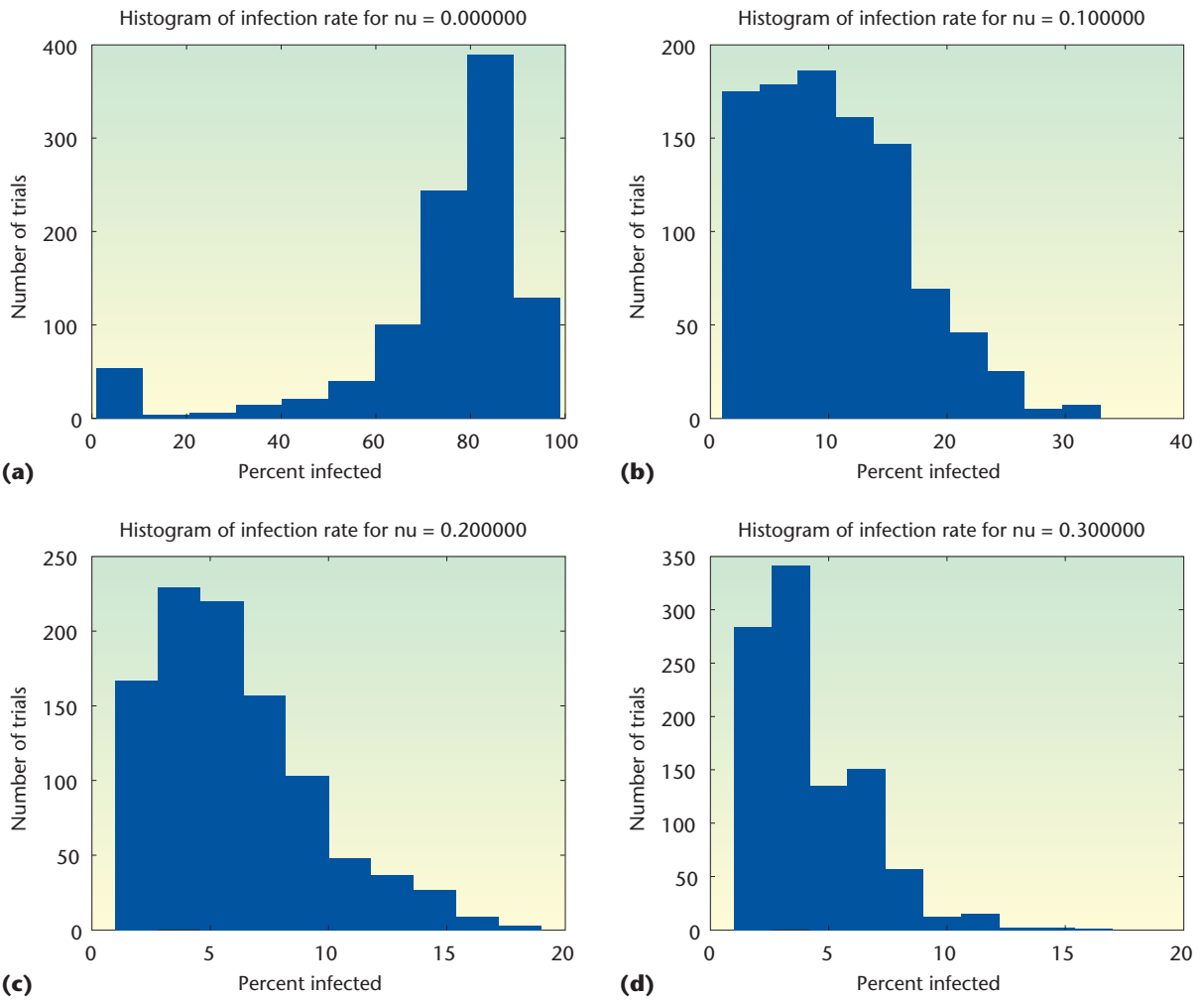

Figure 2. Results of 1,000 trials for a 10 × 10 grid of hospital beds. The infection rate is $\tau = 0.2$, and the vaccination rate varies.

three. For efficiency, this should be computed as $A(A\mathbf{e}_1)$ rather than as $(A^2)\mathbf{e}_1$. Explain why, by doing the operations counts.

d. If we compute $\mathbf{z} = A^j\mathbf{e}_1$ for a large enough j, we will have the (exact) probabilities of being in each state after the epidemic passes. Use this fact to compute the probabilities of having one, two, or three infected individuals, and compare these probabilities with the results of a Monte Carlo experiment as performed in Problem 4 but using three individuals. How many Monte Carlo simulations does it take to get two digits of accuracy in the probabilities?

e. In this simple problem, you can determine the three probabilities directly from Figure 3, by determining the probability of a transition from state A to states P, Q, R, and S. Show how to derive these probabilities, giving the same answer as you obtained via the Markov chain computation earlier.

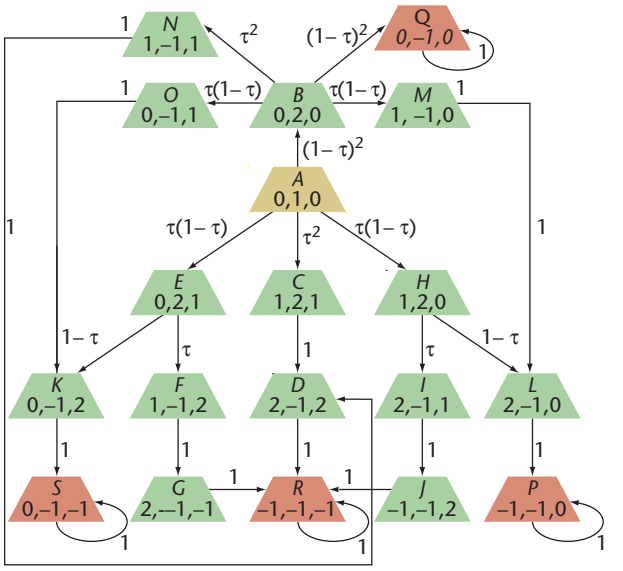

Figure 3. A Markov chain that models three patients, with the middle patient infected. The yellow state is our starting state; the red states are the possible outcomes when the infection runs its course, corresponding to one, two, or three patients eventually infected.

Answer:

a. Figure 3 gives the transition probabilities; the matrix is given in the Matlab code on the Web site.

b. $\boldsymbol{A}\mathbf{e}_1$ is equal to column 1 of $\boldsymbol{A}$, which contains the probabilities of transitioning from state 1 to any other state. Generally, if $\mathbf{p}$ is a vector of the probabilities of initially being in each of the states, then $\boldsymbol{A}\mathbf{p}$ is the vector of probabilities of being in them at time 1.

c. Computing $\boldsymbol{A}(\boldsymbol{A}\mathbf{e}_1)$ costs $2s^2$ multiplications, where s is the number of states. Computing $(\boldsymbol{A}^2)\mathbf{e}_1$ costs $s^3 + s^2$ multiplications, which grows quite a bit larger when s is large. We should also take advantage of the zeros in $\boldsymbol{A}$ and avoid multiplying by them. If we do this for our matrix, then $\boldsymbol{A}$ has 21 nonzero elements whereas $\boldsymbol{A}^2$ has 23, so again it takes more multiplications to form $(\boldsymbol{A}^2)\mathbf{e}_1$ than to form $\boldsymbol{A}(\boldsymbol{A}\mathbf{e}_1)$. We also should note that the product $\boldsymbol{A}\mathbf{e}_1$ is just the first column of $\boldsymbol{A}$, so we could compute it without multiplications.

d. In this experiment, it took 280 Monte Carlo simulations to get two digits of accuracy. Asking for three digits raises the number of trials into the 10,000s because the variance is high relative to threshold.

e. There is only one path to state Q (corresponding to a single infection), and the product of the probabilities of transitions along this path are $(1 - \tau)^4$. There are two paths to state S; summing the product of the probabilities along the paths gives $(\tau(1 - \tau)^2 + \tau(1 - \tau)^3)$. The probability of reaching state P is the same, so the probability of two infections is twice this number. Similarly, the probability of reaching state R, corresponding to three infections, is $\tau^2 + 2\tau^2(1 - \tau) + (1 - \tau)^2\tau^2$.

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. Her interests include numerical linear algebra, optimization, and scientific computing. She received a BS in mathematics from Purdue University and a PhD in computer science from Stanford. She is a member of SIAM, ACM, and AWM. Contact her at the Computer Science Dept., Univ. of Maryland, College Park, MD 20742; oleary@cs.umd.edu; www.cs.umd.edu/users/oleary/.