Continuous time Markov chains (week 10)
Solutions

1 Stochastic simulation of Lotka and Volterra's predator-prey model. To implement Gillespie's algorithm in the case of Lotka and Volterra's predator-prey model, the following steps were taken.

1) Time and CTMC state were initialized as $T_0 = 0$ and $\mathbf{X}(0) = \begin{bmatrix} 50 \\ 100 \end{bmatrix}$
2) All hazards $h_i(\mathbf{X})$ were calculated for each state as:
   - $h_1(X, Y) = c_1 X = X$
   - $h_2(X, Y) = c_2 XY = 0.005XY$
   - $h_1(X, Y) = c_3 * Y = 0.6Y$
3) The transition rate $\nu(\mathbf{X})$ was calculated for each state as
$$\nu(X, Y) = h_1(X, Y) + h_2(X, Y) + h_3(X, Y) = X + 0.005XY + 0.6Y$$
4) The random time of the next reaction was drawn for each state by choosing a value from the exponential distribution with parameter $\nu(X, Y)$.
5) Time was incremented by the above value.
6) An specific event, i, was drawn (at the above time) with the probabilities $P\{R_i\}$:
   - $P\{R_1\} = \frac{X}{\nu X,Y}$
   - $P\{R_2\} = \frac{0.005XY}{\nu X,Y}$
   - $P\{R_3\} = \frac{0.6Y}{\nu X,Y}$
7) The state vector was updated using the stoichiometry left and right matrices:
$$S^{(l)} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix} S^{(r)} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \\ 0 & 0 \end{bmatrix}$$
   When were derived from the reactions specified by the model, namely:
$$\begin{aligned} R_1 : X &\rightarrow 2X \\ R_2 : X + Y &\rightarrow 2Y \\ R_3 : Y &\rightarrow \emptyset \end{aligned}$$
   The state was updated with the equation $\mathbf{X}_{t+1} = \mathbf{X}_t - s_i^{(l)} + s_i^{(r)}$, where s_i is the i^{th} row of either stoichiometry matrix and i is determined by which event is drawn.
8) This process (from step 2) was repeated for every state until $t_{max} = 30$ seconds had passed.

The code for the simulation of the model follows.

```
%predprey.m
```

```
function [X, T] = predprey

X(1, 1) = 50;
X(2, 1) = 100;
T(1) = 0;

S_l = [1, 0; 1, 1; 0, 1];
S_r = [2, 0; 0, 2; 0, 0];

h = zeros(3, 1);


t_max = 30;
c = [1; 0.005; 0.6];

i = 1;
while T(i) < t_max
    x = X(1, i);
    y = X(2, i);
    if x == 0
        break
    else
    dep = [x; x*y; y];
    for j = 1:3;
        h(j) = c(j)*dep(j);
    end
    nu = sum(h);
    t = exprnd(1/nu);
    T(i+1) = T(i) + t;
    prob = rand(1);
    if prob <= h(1)/nu;
        reaction = 1;
    elseif prob <= (h(1)+h(2))/nu;
        reaction = 2;
    else
        reaction = 3;
    end
    X(1, (i+1)) = x - S_l(reaction, 1) + S_r(reaction, 1);
    X(2, (i+1)) = y - S_l(reaction, 2) + S_r(reaction, 2);
    i = i + 1;
    end
end

figure;
plot(T, X);
title('Predator and prey populations over time when X_0=50 and Y_0=100');
```

```
legend('Prey', 'Predators');
xlabel('time');
ylabel('population size');
axis([0, 30, 0, 800]);

figure;
plot(X(1, :), X(2, :));
title('State space representation X vs. Y');
xlabel('X=Prey');
ylabel('Y=Predators');
```

Representative plots follow on the next page.

Fig. 1. The two populations over time, showing the boom and bust cycles.

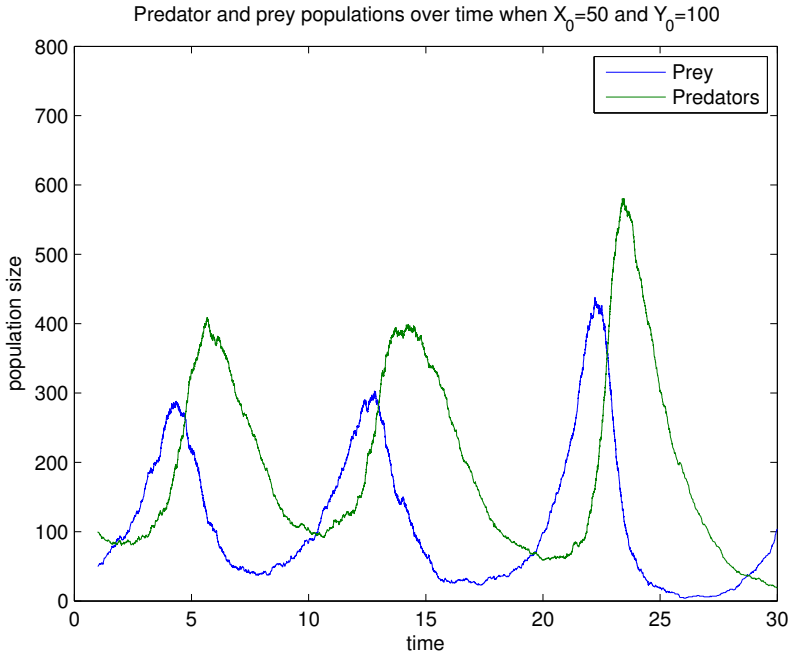
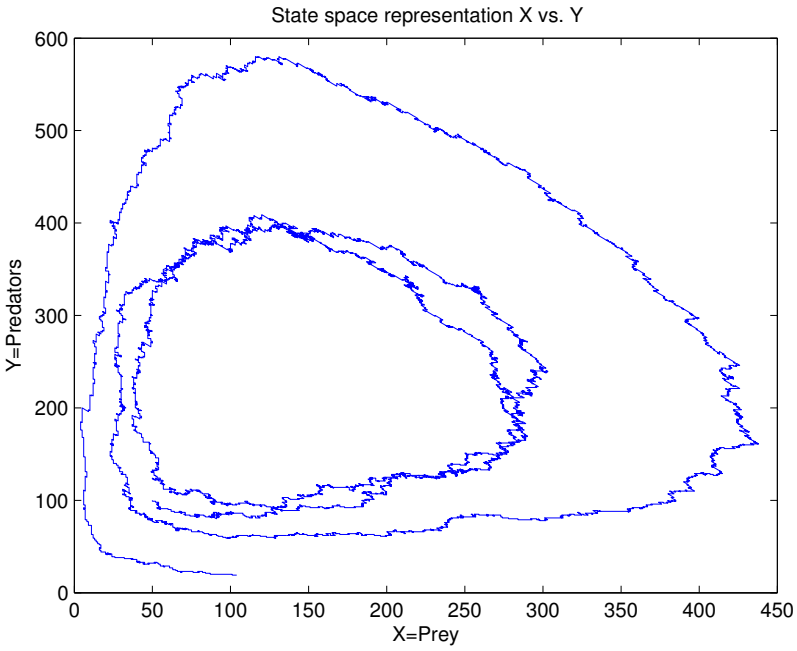


Fig. 2. The model in state space.



2 Stochastic simulation of the lac operon.

The annotated code for the simulation of the model follows.

- **Code to generate the left and right stoichiometry matrices.**

```
%lacoperon_matrices.m

function [sl, sr] = lacoperon_matrices

%Each type of molecule is assigned a row in the matrices for easy reference
G = 1; L = 2; betaG = 3; Op = 4; mRNA = 5; AOp = 6;
ROp = 7; LRP = 8; CAP = 9; LRPL = 10; CAPG = 11;

%initialize the stoichiometry matrices
sl = zeros(16, 11);
sr = zeros(16, 11);

%lactose digestion
sl(1, L) = 1;
sl(1, betaG) = 1;
sr(1, G) = 1;
sr(1, betaG) = 1;

%glucose consumption
sl(2, G) = 1;

%regular transcription
sl(3, Op) = 1;
sr(3, Op) = 1;
sr(3, mRNA) = 1;

%activated transcription
sl(4, AOp) = 1;
sr(4, AOp) = 1;
sr(4, mRNA) = 1;

%repressed transcription
sl(5, ROp) = 1;
sr(5, ROp) = 1;
sr(5, mRNA) = 1;

%operon repression
sl(6, LRP) = 1;
sl(6, Op) = 1;
sr(6, ROp) = 1;

%operon liberalization
```

```
sl(7, ROp) = 1;
sr(7, LRP) = 1;
sr(7, Op) = 1;

%repressor neutralization
sl(8, LRP) = 1;
sl(8, L) = 1;
sr(8, LRPL) = 1;

%repressor dissociation
sl(9, LRPL) = 1;
sr(9, LRP) = 1;
sr(9, L) = 1;

%operon activation
sl(10, CAP) = 1;
sl(10, Op) = 1;
sr(10, AOp) = 1;

%operon deactivation
sl(11, AOp) = 1;
sr(11, CAP) = 1;
sr(11, Op) = 1;

%CAP neutralization
sl(12, CAP) = 1;
sl(12, G) = 1;
sr(12, CAPG) = 1;

%CAP dissociation
sl(13, CAPG) = 1;
sr(13, CAP) = 1;
sr(13, G) = 1;

%protein synthesis
sl(14, mRNA) = 1;
sr(14, mRNA) = 1;
sr(14, betaG) = 1;

%mRNA decay
sl(15, mRNA) = 1;

%betaG decay
sl(16, betaG) = 1;
```

- **Code to generate the hazards, $h_i(\mathbf{X})$, the transition rate$\nu(\mathbf{X})$, and the probability distribution given by $h_i(\mathbf{X})/\nu(\mathbf{X})$.**

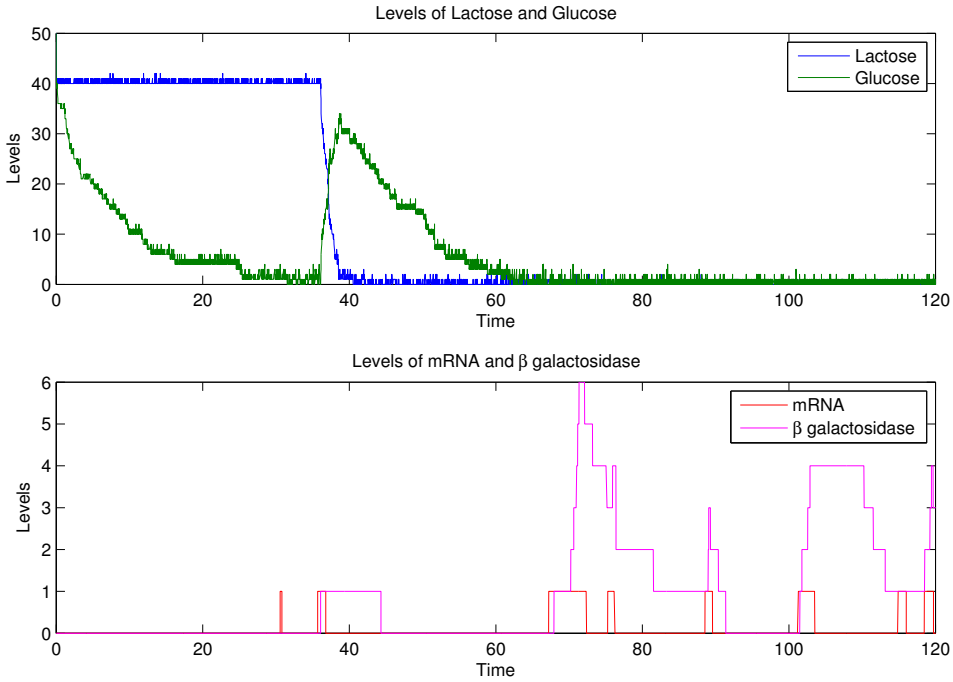
```
   %lacoperon_h

function [nu, prob] = lacoperon_h(X)

%The vector of constants c_i
c = [1, 0.1, 0.01, 0.1, 0.001, 1, 1, 10, 1, 1, 1, 10, 1, 1, 1, 0.1];

%The row nubmers of each of the species in the state matrix
G = 1; L = 2; betaG = 3; Op = 4; mRNA = 5; AOp = 6;
ROp = 7; LRP = 8; CAP = 9; LRPL = 10; CAPG = 11;

h = zeros(1, 16);

%The hazards calculated for each reaction
 h(1) = c(1)*X(L)*X(betaG);
 h(2) = c(2)*X(G);
 h(3) = c(3)*X(Op);
 h(4) = c(4)*X(AOp);
 h(5) = c(5)*X(ROp);
 h(6) = c(6)*X(LRP)*X(Op);
 h(7) = c(7)*X(ROp);
 h(8) = c(8)*X(LRP)*X(L);
 h(9) = c(9)*X(LRPL);
 h(10) = c(10)*X(CAP)*X(Op);
 h(11) = c(11)*X(AOp);
 h(12) = c(12)*X(CAP)*X(G);
 h(13) = c(13)*X(CAPG);
 h(14) = c(14)*X(mRNA);
 h(15) = c(15)*X(mRNA);
 h(16) = c(16)*X(betaG);

 %The transition rate, nu
 nu = sum(h);

 %The probability distributionto determine which event happens
 prob = h./nu;
```

- **Code to execute the simulation and plot the results.**

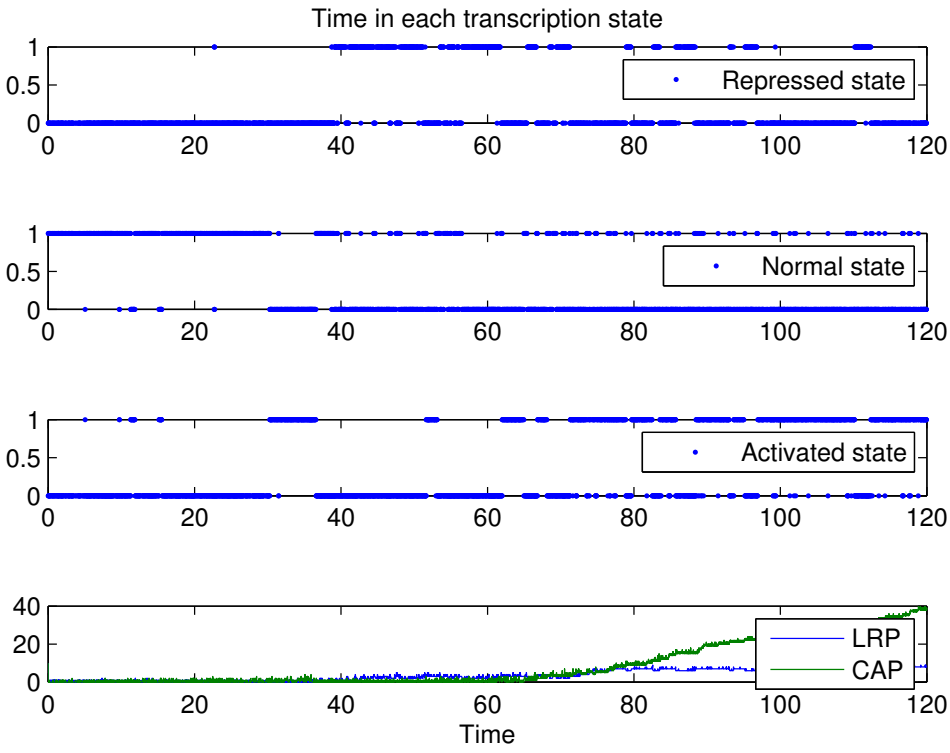
```
    % lacoperon.m

function X = lacoperon(sl, sr)

X = zeros(11, 1);
T(1) = 0;

G = 1; L = 2; betaG = 3; Op = 4; mRNA = 5; AOp = 6;
```

```
ROp = 7; LRP = 8; CAP = 9; LRPL = 10; CAPG = 11;

X(G, 1) = 50;
X(L, 1) = 50;
X(Op, 1) = 1;
X(LRP, 1) = 10;
X(CAP, 1) = 10;


t_max = 120;


i = 1;
while T(i) < t_max
current_state = X(:, i);
[nu, prob] = lacoperon_h(current_state);
t = exprnd(1/nu);
T(i+1) = T(i) + t;
cum_p=cumsum(prob);
event=sum(cum_p<rand(1))+1;
X(:, i+1) = X(:, i) - sl(event, :)' + sr(event, :)';
i = i+1;
end

%Graph monitoring levels of lactose, glucose, mRNA and betaG
figure;
subplot(211)
plot(T,X(L, :), T, X(G, :));
legend('Lactose', 'Glucose');
axis([0, 120, 0, 50]);
title('Levels of Lactose and Glucose');
xlabel('Time'); ylabel('Levels');
subplot(212)
title('Levels of mRNA and \beta galactosidase');
plot(T, X(mRNA, :), 'r', T, X(betaG, :), 'm');
legend('mRNA', '\beta galactosidase');
xlabel('Time'); ylabel('Levels');
%Graph monitoring time in transcription state and number of CAP and LRP
%molecules
figure;
title('Time in each transcription state');
subplot(411)
plot(T, X(ROp, :),'.');
legend('Repressed state');
axis([0, 120, 0, 1]);
subplot(412)
```

```
plot(T, X(Op, :), '.');
legend('Normal state');
axis([0, 120, 0, 1]);
subplot(413)
plot(T, X(AOp, :), '.');
legend('Activated state');
axis([0, 120, 0, 1]);
subplot(414)
plot(T, X(LRP, :), T, X(CAP, :));
legend('LRP', 'CAP');
axis([0, 120, 0, 40]);
xlabel('Time');
```



As shown above, the levels of βgalactosidase rise when glucose supply falls because it is needed to transform the lactose into glucose. The mRNA levels rise at the same time because mRNA is used in the production of βgalactosidase. Once supplies of both glucose and lactose are short (after around time 70), there is an initial surplus of βgalactosidase because there production has not slowed, but there is not enough lactose to combine to glucose, so the βgalactosidase levels remain high.



This graphic shows the time spent in each transcription state as well as the levels of LRP and CAP. We can see that as levels of CAP rise, the operon is in activated state more often, which aligns with the fact that CAP is used to push the operon from normal state to activated state.