TASK

The arrival of new customers is modeled in the following way. Let X_t be a continuous time Markov chain, which occupies state i at time 0. Conditional on all the future dynamics of X_t, process N_t is a Poisson process with intensity lambda_t = r(X_t). Here r() is some non-negative function. Each new arrival of a customer is given by a jump of process N_t... Derive a differential equation for the probability of no customers arriving before time t. This equation can be solved by finite difference methods later on.

SOLUTION

First, let us recall that expression E[Z | X_0 = i] denotes the expectation of random variable Z under the condition that Markov chain X_t occupies state i at time 0. We notice that saying "no customers before time t" is equivalent to saying "N_t = 0". Therefore,

P(no customers before time t | X_0 = i) = P(N_t = 0 | X_0 = i) = E[1_{N_t = 0} | X_0 = i]. **(1)**

In the equation (1) above, 1_A is an indicator random variable which equals 1 if A is true and equals 0 otherwise. By the law of iterated expectations, equation (1) can be continued as

P(no customers before time t | X_0 = i) = E[1_{N_t = 0} | X_0 = i] =

= E[E[1_{N_t = 0} | {X_s, s >= 0}, X_0 = i] | X_0 = i] =

= E[P(N_t = 0 | {X_s, s >= 0}, X_0 = i) | X_0 = i] =

= | Recall one of the properties of a Poisson process: N_t has Poisson distribution with parameter int_{s=0}^t lambda(s) ds. Therefore, conditional on {X_s, s >= 0}, N_t has Poisson distribution with parameter int_{s=0}^t r(X_s) ds | =

= E[exp{ -int_{s=0}^t r(X_s) ds } | X_0 = i] = **(2)**

= E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i]. **(2')**

Now let us denote P(no customers before time t | X_0 = i) as g(i,t). Then by (2')

g(i,t) = E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i] =

= | Let us **condition on the state of the Markov chain** X_t at the moment of time h, where h is tiny. We will use the law of iterated expectations again. In the text below the symbol "<>" means "is not equal to". |

= Sum_{j <> i} E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = j] * P(X_h = j | X_0 = i) +

+ E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = i] * P(X_h = i | X_0 = i). **(3)**

Let us recall how a continuous time Markov chain jumps from one state to another. Suppose currently the chain is in state i. We are observing a collection of Poisson processes N_ij, where j <> i. Each process N_ij has intensity A_ij. If process N_ik is the first process to jump, Markov chain X_t moves into state k.

Intensities A_ij are collected in matrix A, called the **generator** of Markov chain X_t. Diagonal elements of the generator are defined as

A_ii = - Sum_{j <> i} A_ij.

We will need the following property of a Poisson process: if the process has intensity *mu* then, for a tiny value of h,

P(exactly one jump in interval [0,h]) = mu * h + e(h),

where e(h) is such function that lim_{h ---> 0} e(h)/h = 0.

Now we are ready to re-write equation (3):

(3) = Sum_{j <> i} E[exp{ -int_{s=0}^t lambda_s ds } **|** X_0 = i, X_h = j] * (A_ij * h + e(h)) +

+ E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = i] * (1 - Sum_{j <> i} A_ij * h + e(h)) =

= | by definition A_ii = - Sum_{j <> i} A_ij | =

= Sum_{j <> i} E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = j] * (A_ij * h + e(h)) +

+ E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = i] * (1 + A_ii * h + e(h)) =

= Sum_{j <> i} E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = j] * A_ij * h +

+ E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = i] * (1 + A_ii * h) + e(h). **(4)**

In the equation (4) above notice the following:

E[exp{ -int_{s=0}^t lambda_s ds } **|** X_0 = i, X_h = j] =

= E[exp{ -int_{s=0}^h lambda_s ds - int_{s=h}^t lambda_s ds } **|** X_0 = i, X_h = j] =

= E[exp{ -int_{s=0}^h lambda_s ds } * exp{ -int_{s=h}^t lambda_s ds } | X_0 = i, X_h = j] =

= E[exp{ -int_{s=0}^h r(X_s) ds } * exp{ -int_{s=h}^t lambda_s ds } | X_0 = i, X_h = j] =

= | We use that fact that, for time s close to time 0, r(X_s) = r(i). | =

= E[(exp{ -int_{s=0}^h r(i) ds } + e(h)) * exp{ -int_{s=h}^t lambda_s ds } | X_0 = i, X_h = j] =

= E[exp{ -r(i) * h } * exp{ -int_{s=h}^t lambda_s ds } | X_0 = i, X_h = j] + e(h) =

= exp{ -r(i) * h } * E[exp{ -int_{s=h}^t lambda_s ds } | X_0 = i, X_h = j] + e(h) =

= exp{ -r(i) * h } * E[exp{ -int_{s=h}^t lambda_s ds } | X_0 = i, X_h = j] + e(h) =

= | We use the stationarity of the Markov chain. It's like **we are starting over** at time h, but now the initial state is state j. | =

= exp{ -r(i) * h } * E[exp{ -int_{s=0}^{t-h} lambda_s ds } **|** X_0 = j] + e(h) =

= | Recall the definition of function g(i,t). | =

= exp{ -r(i) * h } * g(j,t-h) + e(h). **(5)**

All right, now let us substitute formula (5) into equation (4).

(4) = Sum_{j <> i} E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = j] * A_ij * h +

+ E[exp{ -int_{s=0}^t lambda_s ds } | X_0 = i, X_h = i] * (1 + A_ii * h) * (1 + A_ii * h) + e(h)   =

= Sum_{j <> i} exp{ -r(i) * h } * g(i,t-h) * A_ij * h + exp{ -r(i) * h } * g(i,t-h) * (1 + A_ii * h) + e(h)   =

= | We use a well-known property of function exp() implied by its Taylor decomposition:
exp(-r(i) * h) = 1 - r(i) * h + e(h) | =

= Sum_{j <> i} (1 - r(i) * h + e(h)) * g(i,t-h) * A_ij * h +
+ (1 - r(i) * h + e(h)) * g(i,t-h) * (1 + A_ii * h) + e(h)   =

= Sum_{j <> i} (1 - r(i) * h) * g(i,t-h) * A_ij * h + (1 - r(i) * h) * g(i,t-h) * (1 + A_ii * h) + e(h) =
= Sum_{**any j**} (1 - r(i) * h) * g(i,t-h) * A_ij * h   + g(i,t-h) - r(i) * h * g(i,t-h) + e(h)   **(6)**.

Let us re-write equation (6).

g(i,t) = Sum_{**any j**} (1 - r(i) * h) * g(i,t-h) * A_ij * h   + g(i,t-h) - r(i) * h * g(i,t-h) + e(h),

or

g(i,t) - g(i,t-h) = Sum_{**any j**} (1 - r(i) * h) * g(i,t-h) * A_ij * h - r(i) * h * g(i,t-h) + e(h),

or

(g(i,t) - g(i,t-h))/h = Sum_{**any j**} (1 - r(i) * h) * g(i,t-h) * A_ij - r(i) * g(i,t-h) + e(h)/h.   **(7)**

Letting h ---> 0 on both sides of equation (7) leads to:

d g(i,t) / dt = Sum_{**any j**} g(i,t) * A_ij - r(i) * g(i,t),   **(8)**

for any state i. We also state the initial condition, which is

g(i,0) = E[exp{ -int_{s=0}^0 lambda_s ds   } | X_0 = i] = 0.   **(9)**

Equations (8) and (9) form the desired system of equations. This system can be solved using a finite difference method to produce the probability of no customers arriving before time t.
