FinM 345/Stat 390 Stochastic Calculus, Autumn 2009

Floyd B. Hanson, Visiting Professor

Email: fhanson@uchicago.edu

Master of Science in Financial Mathematics Program
University of Chicago

Lecture 5 (from Chicago)

More Jump-Diffusion Calculus, Simple and Compound

6:30-9:30 pm, 26 October 2009, Kent 120 in Chicago

7:30-10:30 pm, 26 October 2009 at UBS Stamford

7:30-10:30 am, 27 October 2009 at Spring in Singapore

FinM 345 Stochastic Calculus:

5. More Jump-Diffusion Calculus — Simple & Compound:

5.1. Jump-Diffusion Rules and SDEs Continued:

(finishing Chapt. 4 of the textbook)

- ***5.1.1. Solution Simulations for Linear Jump-Diffusion SDEs with Constant Coefficients:*** Upon merging and modifying the simulation algorithms for small time increments in Figure 7 on L1-p32 and using the cumulative sum of normal RNG Wiener increment approximations, together with the cumulative sum of uniform RNG Poisson increment approximations with the binomial RNG with `n=1` to model the "Bernoulli" zero-one jump law, we show simulations of the financially relevant linear jump-diffusion process with constant parameters solution (4.39) on L4-51 in Figure 5.1.

The basic simulation is performed on the approximate exponent increment

$$\Delta Y_i \simeq (\mu_0 - \sigma_0^2/2)\Delta t + \sigma_0 \Delta W_i + \ln(1+\nu_0)\Delta P_i, \quad (5.1)$$

corresponding to SDE (5.1), where $\Delta t = 0.001$ for this MATLAB-generated figure, $\Delta W_i \simeq \mathrm{DW}(i)$, $\Delta t = \mathrm{Dt}$, where $\mathrm{DW} = \texttt{sqrt}(\mathrm{Dt})*\texttt{randn}(1, N)$; and $\Delta P = \texttt{binornd}(1, \lambda_0 Dt, 1, N)$, approximating the zero-one jump law through Bernoulli form of the binomial RNG when the first parameter is $n = 1$ and the second is the Poisson parameter $\Delta\Lambda = \lambda_0 \Delta t$. {**In place of calls by random states to** $\texttt{randn}(1, N)$**, use serial calls to** $\texttt{normrnd}(0,\texttt{sqrt}(\mathrm{Dt}), 1, N)$**.**}

The sample path of the state exponent, $\mathbf{YS}$, starting from a zero initial condition $\mathbf{YS(1)=0}$ rather than $\mathbf{ln}(x_0)$, for $i=1:N$, is approximated by

$$\mathbf{YS}(i+1)=\mathbf{YS}(i)+(\mu_0-\sigma_0^2/2)*\mathbf{Dt}+\sigma_0*\mathbf{DW}(i)$$
$$+\mathtt{log}(1+\nu_0)*\mathbf{DP}(i),$$

with $t(i+1)=i*\mathbf{Dt}$. The sample path of the desired state, $\mathbf{XS}$, is approximated by

$$X(t(i+1))\simeq \mathbf{XS}(i+1)=x_0*\exp(\mathbf{YS}(i+1)).$$

The mean trajectory, $\mathbf{XM}$, is given by

$$\mathbf{E}[X(t(i+1))]\simeq \mathbf{XM}(i+1)$$
$$=x_0*\exp((\mu_0+\lambda_0*\nu_0)t(i+1)).$$

The mean trajectory is also displayed in the figure along with the upper $\mathbf{XT}$ exponential standard deviation estimate

$$\mathbf{E}[X(t(i+1))]*V(i+1) \simeq \mathbf{XT}(i+1)$$
$$= \mathbf{XM}(i+1)*V(i+1)$$

and lower $\mathbf{XB}$ exponential standard deviation estimate

$$\mathbf{E}[X(t(i+1))]/V(i+1) \simeq \mathbf{XB}(i+1)$$
$$= \mathbf{XM}(i+1)/V(i+1),$$

where the factor

$$V(i+1) = \exp\left(\sqrt{\mathbf{Var}[Y(t(i+1))]}\right)$$
$$= \exp\left(\sqrt{(\sigma_0^2 + \lambda_0 * \log^2(1+\nu_0))t(i+1)}\right)$$

is the exponential of the standard deviation of the exponent process $Y(t)$ in discrete form.

Alternatively, one plus or minus the **coefficient of variation** formula (4.42) on page L4-56 could be used to form a deviation factor, but the factor $V(i+1)$ above is more appropriate since it corresponds better to the finite difference simulation approximation. Although the jump-amplitude is only a 10% decrement, the jumps are very noticeable in the figure, while both the jump and diffusion component processes result in excesses beyond the indicated upper and lower standard deviation estimates. The estimates correspond to rough confidence intervals and not bounds. See the updated, vectorized code called ***linjumpdiff09fig1.m*** and is also found in the Chalk Course Documents, for the MATLAB code used to produce the new Figures 5.1–5.3.

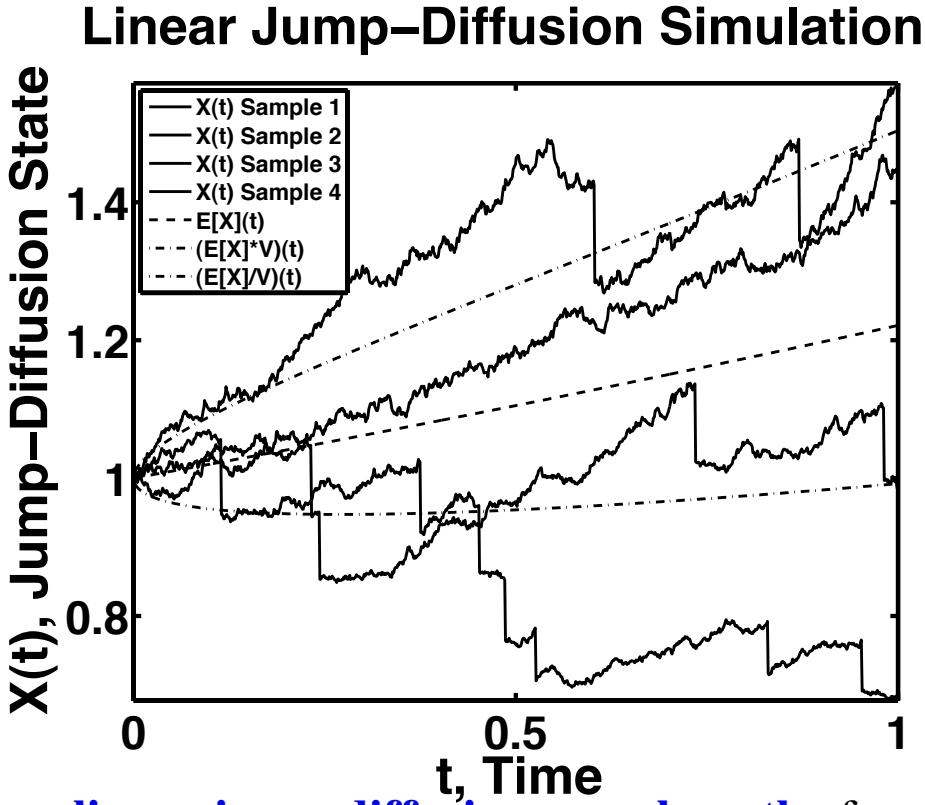

Figure 5.1: **Four linear jump-diffusion sample paths** for constant coefficients are simulated using MATLAB with $N = 1000$ sample points, maximum time $T = 1.0$ and four `randn` and four sequential Bernoulli {`binornd; n=1`} **zero-one** states. Parameter values are $\mu_0 = 0.5$, $\sigma_0 = 0.10$, $\nu_0 = -0.10$, $\lambda_0 = 3.0$ and $x_0 = 1.0$. In addition to the four simulated states, the expected state $\mathbf{E}[X(t)]$ and two deviation measures $\mathbf{E}[X(t)] * V(t)$ and $\mathbf{E}[X(t)]/V(t)$ are displayed, where the factor $V(t)$ is based on the standard deviation of the state exponent $Y(t)$.

The same code, in the case of constant parameters, can be used for the pure-diffusion model in the Equation (3.2) on L3-p53 by setting $\nu_0 = 0$ for the diffusion as shown in Figure 5.2 or for the pure-jump model in the Equation (4.15) on L4-p21 by setting $\sigma_0 = 0$ for the jump process as shown in Figure 5.3.

See also the old Program C.14, called **`linjumpdiff03fig1.m`** in the Online Appendix C, for the MATLAB code used to produce the older Figures 4.3–4.5 in the textbook, using just the uniform generator **`U=rand(1,N)`** and acceptance-rejection for the zero-one law, with a $P0 = (0, 1 - \lambda_0 \Delta t)$ and $P1 = [1 - \lambda_0 \Delta t, 1)$ partition of $(0, 1)$, such that if $U(i) \in P1$ there is a jump at the ith time-step, else not.

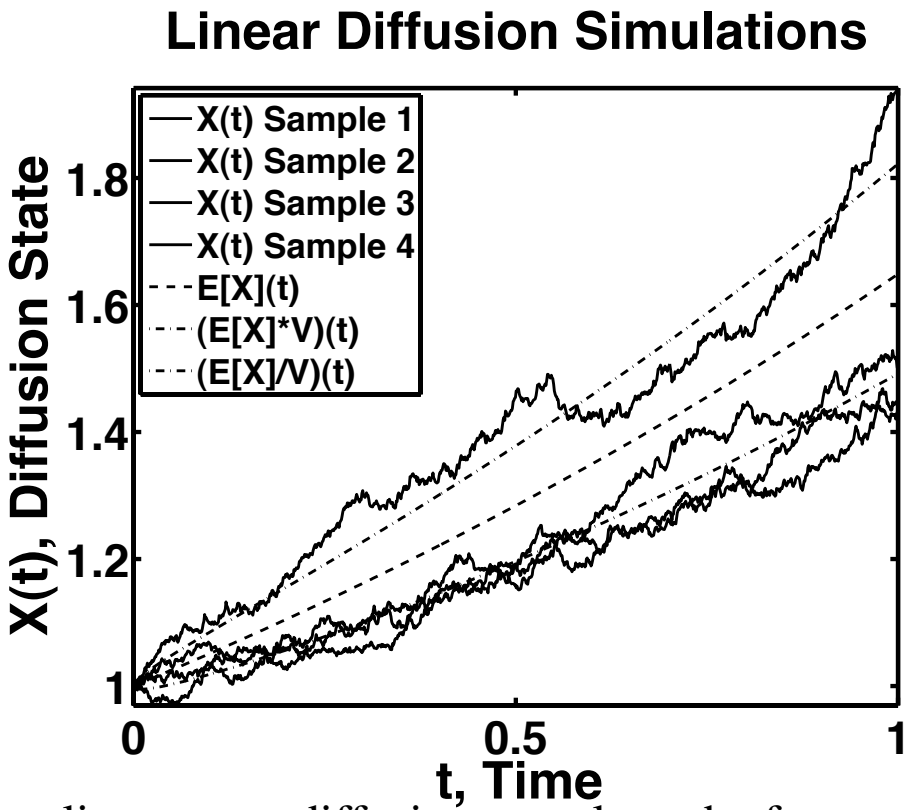

Figure 5.2: Four linear pure-diffusion sample paths for constant coefficients are simulated using MATLAB with $N = 1000$ sample points, maximum time $T = 1.0$ and four `randn` states. Parameter values are $\mu_0 = 0.5$, $\sigma_0 = 0.10$, $\nu_0 = 0.0$, and $x_0 = 1.0$. In addition to the four simulated states, the expected state $\mathbf{E}[X(t)]$ and two deviation measures $\mathbf{E}[X(t)] * V(t)$ and $\mathbf{E}[X(t)]/V(t)$ are displayed, where the factor $V(t)$ is based on the standard deviation of the state exponent $Y(t)$.

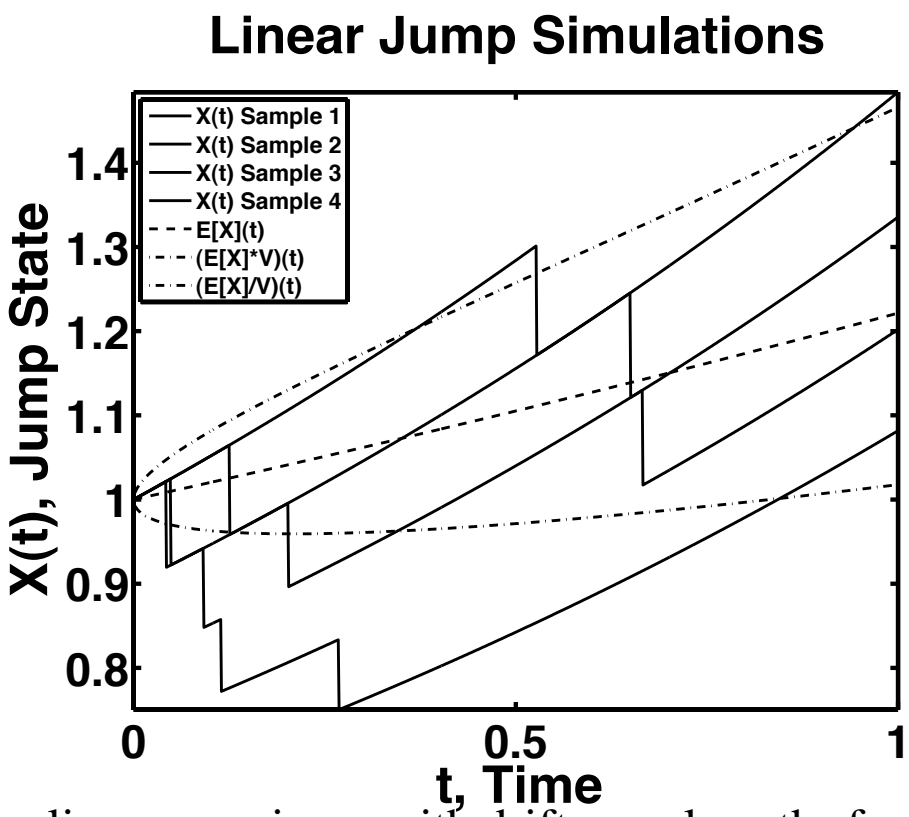

Figure 5.3: Four linear pure jump with drift sample paths for constant coefficients are simulated using MATLAB with $N = 1000$ sample points, maximum time $T = 1.0$ and four sequential Bernoulli{`binornd; n=1`} **zero-one** states. Parameter values are $\mu_0 = 0.5$, $\sigma_0 = 0.0$, $\nu_0 = -0.10$, and $x_0 = 1.0$. In addition to the four simulated states, the expected state $\mathbf{E}[X(t)]$ and two deviation measures $\mathbf{E}[X(t)] * V(t)$ and $\mathbf{E}[X(t)]/V(t)$ are displayed, where the factor $V(t)$ is based on the standard deviation of the state exponent $Y(t)$.

- ***Linear Jump-Diffusion Simulations, revised book MATLAB code example (edited)***:

```
function linjumpdiff09fig1
% Revised Book Illustration for Linear Jump Diffusion
%   Simulation with constant coefficients for t
%   with sample variation:
%      DX(t)=X(t)*(mu*Dt+sig*DW(t)+nu*DP(t),
%      X(0) = x0.
%  Or log-state:
%    DY(t)=(mu-sig^2/2)*Dt+sig*DW(t)+log(1+nu)*DP(t),
%     Y(0) = log(x0).
%   Generation summing increments DW, fixed Dt
%     with Poisson jump increment added.
%   SMALL increments assumed, for zero-one jump law
%   Allows Separate Driver Input and Special Jump
%       or Diffusion Handling.
clc % clear variables,
clf % clear figures
```

```
fprintf('\nfunction linjumpdiff09fig1 OutPut:');
%%% Initialize input to jdsimulator
N = 1000; T = 1.0; % Set time grid: Fixed Delta{t}.
mu = 0.5; sig = 0.10; nu = -0.10; lambda = 3;
% set constant parameters and call:
jdsimulator(mu,sig,nu,lambda,N,T);
% END INPUT FOR JUMP-DIFFUSION SIMULATOR.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function jdsimulator(mu,sig,nu,lambda,N,T)
idiff = 1; ijump = 1;
if sig == 0, idiff = 0; end
if nu == 0, ijump = 0; end
nfig = 0; % figure counter.
NI = N+1;  Dt = T/NI;
sqrtdt = sqrt(Dt); % standard Wiener time scale.
sigsqrtdt = sig*sqrtdt;
muddt = (mu - sig^2/2)*Dt; % corrected drift term.
Lambda = lambda*Dt;  % Poisson parameter;
```

```
lognu = log(1 + nu); % log relative jump amplitude.
% Begin Sample Path Calculation:
t = 0:Dt:T; kstates = 4; x0 = 1.0;
XS = zeros(kstates,NI+1); % declare state vector.
DW = zeros(1,NI);
DP = zeros(1,NI);
MSO = muddt*ones(1,NI);
for kstate = 1:kstates % Multiple Sample Paths:
    if idiff == 1
      randn('state',kstate-1); % initial normal state
      DW = sigsqrtdt*randn(1,NI); % sig*sqrt(dt)*dW
    end
    if ijump == 1
      DP = lognu*binornd(1,Lambda,1,NI);% lognu*DP(t)
    end % for  0-1 law samples, assume Lambda<<1;
    MS = zeros(1,NI+1); % Set Initials
    WS = zeros(1,NI+1);
    PS = zeros(1,NI+1);
```

```
    MS(1,2:NI+1) = cumsum(MSO); % Set Sums
    WS(1,2:NI+1) = cumsum(DW);
    PS(1,2:NI+1) = cumsum(DP);
    YS = MS + WS + PS;
    XS(kstate,:) = x0*exp(YS);%  Vector exponential.
end
% Compute Mean State Path and +/- One Std. Deviation:
muxexp = mu + lambda*nu;
XM = x0*exp(muxexp*t); % vector-exp
sigyexp = sig^2 + lambda*lognu^2;
V = exp(sqrt(sigyexp*t));
XT = XM.*V;
XB = XM./V;
% Begin Plot:
nfig = nfig + 1;
kjd = 4 - 2*idiff - ijump;
NP = N + 2;
stitle = {'Linear Jump-Diffusion Simulations' ...
```

```
        ,'Linear Diffusion Simulations' ...
        ,'Linear Jump Simulations'};
sylabel = {'X(t), Jump-Diffusion State','X(t), ...
    Diffusion State','X(t), Jump State'};
fprintf('\n\nFigure(%i):  Linear J-D Sims\n',nfig)
figure(nfig)
scrsize = get(0,'ScreenSize'); % target screen
ss = [5.0,4.0,3.5]; % figure spacing
plot(t,XS(1,1:NP),'k-' ...
    ,t,XS(2,1:NP),'k-' ...
    ,t,XS(3,1:NP),'k-' ...
    ,t,XS(4,1:NP),'k-' ...
    ,t,XM,'k--' ...
    ,t,XT,'k-.' ...
    ,t,XB,'k-.','LineWidth',2);
axis tight;
title(stitle(kjd)...
    ,'FontWeight','Bold','Fontsize',36);
```

```
ylabel(sylabel(kjd)...
    ,'FontWeight','Bold','Fontsize',36);
xlabel('t, Time'...
    ,'FontWeight','Bold','Fontsize',36);
hlegend=legend('X(t) Sample 1','X(t) Sample 2'...
   ,'X(t) Sample 3','X(t) Sample 4','E[X](t)'...
   ,'(E[X]*V)(t)','(E[X]/V)(t)','Location','NorthWest');
set(hlegend,'Fontsize',16,'FontWeight','Bold');
set(gca,'Fontsize',32,'FontWeight','Bold'...
    ,'linewidth',3);
set(gcf,'Color','White','Position' ...
,[scrsize(3)/ss(nfig) 60 scrsize(3)*.6 scrsize(4)*.8]);
% End JDSimulator Code
```

- **Remarks**:
  - *Simulation caution:* Note that the constant coefficient closed-form solution (4.39) on L4-51 is not used directly, i.e.,

$$X_i = X_0(1+\nu_0)^{P_i} \exp\left((\mu_0 - \sigma_0^2/2)t_i + \sigma_0 W_i\right),$$

for $i = 0:(n+1)$, where $t_{n+1} = T$ is the final time, by directly simulating the random variables P_i and W_i, since they are not independent of either earlier or later values P_j and W_j for $j \neq i$. So such a simulation would be incorrectly approximated.

{**Note that MATLAB, the MATrix LABoratory, is unit based, note zero based, since zero subscripts for arrays are illegal, so $i = 1:(n+2)$ and NOT $i = 0:(n+1)$.**}

- However, simulating the increment set $\{\Delta P_i, \Delta W_i\}$ for $i = 0:n$ would be an appropriate use of the approximate independence property of the pseudo-random number generators of ΔP_i and ΔW_i, i.e.,

$$X_{i+1} = X_i(1+\nu_0)^{\Delta P_i} \exp\left((\mu_0 - \sigma_0^2/2)\Delta t_i + \sigma_0 \Delta W_i\right),$$

for $i = 0:n$, and we note that

$$\Delta Y_i = (\mu_0 - \sigma_0^2/2)\Delta t_i + \sigma_0 \Delta W_i + \ln(1+\nu_0)\Delta P_i$$

and that $\exp(\ln(1+\nu_0)\Delta P_i) = (1+\nu_0)^{\Delta P_i}$ using the exponential-logarithm inverse relationship.

- Considering finite precision arithmetic, this would be similar to using

$$W_{i+1} = \sum_{j=0}^{i} \Delta W_j \quad \& \quad P_{i+1} = \sum_{j=0}^{i} \Delta P_j$$

for $(i+1) = 1 : (n+1)$.

It is **important to build simulations in independent increments**.

See the textbook for further SDE analysis and computational references.

• Linear Jump-Diffusion SDEs with Time-Dependent Coefficients (The Market is Time-Dependent):

While linear constant coefficient SDEs often occur in applications such as elementary finance, time-dependence of market parameters can also play an important role. For this reason, our attention returns to the time-dependent coefficients of the financially relevant linear SDE solution (4.47) on page L4-p50 and the expected state trajectory. However, the procedure is more complex than in the simple constant coefficient case, since the expectations of exponentials of integrals are required. In the following two lemmas and related corollaries, we first consider the pure-diffusion case and then the pure-jump case.

Lemma 5.1.1. Expectation of $\exp(\int \sigma dW(s))$:
Let $\sigma(t)$ be square integrable on $[t_0, t]$. Then

$$\mathrm{E}\left[\exp\left(\int_{t_0}^{t}\sigma(s)dW(s)\right)\right] \stackrel{\text{ifa}}{=} \exp\left(\frac{1}{2}\int_{t_0}^{t}\sigma^2(s)ds\right). \quad (5.2)$$

Sketch of Proof:

To keep the justification reasonably brief and maintain the usefulness as an integration technique, the stochastic diffusion integral will first be formally decomposed into a forward Itô sum, averaged and then recomposed back into a deterministic integral. The justification of each step will be indicated in shorthand on the sign of the relation, but the more rigorous Itô limits will be omitted.

It is remarkable that we can reduce this expectation the exponential of a diffusion integral to such an explicit form.

Let $t_i = t_0 + i * \Delta t$ for $i = 0 : n+1$ be a proper partition of $[t_0, t]$ with $\Delta t = (t - t_0)/(n+1)$,
$\Delta W_i = W(t_{i+1}) - W(t_i)$ and $\sigma_i = \sigma(t_i)$ for $i = 0 : n$, so

$$\mathbf{E}\left[\exp\left(\int_{t_0}^{t} \sigma(s) dW(s)\right)\right] \overset{\text{ifa}}{\simeq} \mathbf{E}\left[\exp\left(\sum_{i=0}^{n} \sigma_i \Delta W_i\right)\right]$$

$$\overset{\text{loe}}{=} \mathbf{E}\left[\prod_{i=0}^{n} \exp\left(\sigma_i \Delta W_i\right)\right]$$

$$\overset{\text{ind}}{\underset{\text{inc}}{=}} \prod_{i=0}^{n} \mathbf{E}_{\Delta W_i}\left[\exp\left(\sigma_i \Delta W_i\right)\right]$$

$$\overset{norm}{\underset{dist}{=}} \prod_{i=0}^{n} \int_{-\infty}^{+\infty} \frac{\exp\left(-\frac{w^2}{2\Delta t} + \sigma_i w\right)}{\sqrt{2\pi \Delta t}} dw$$

$$\overset{comp}{\underset{sq}{=}} \prod_{i=0}^{n} \exp\left(\sigma_i^2 \Delta t/2\right) \overset{\text{loe}}{=} \exp\left(\sum_{i=0}^{n} \sigma_i^2 \Delta t/2\right)$$

$$\overset{\text{ifa}}{\simeq} \exp\left(\tfrac{1}{2} \int_{t_0}^{t} \sigma^2(s) ds\right). \quad \square$$

Lemma 5.1.2. Expectation of $\exp(\int \ln(1+\nu)dP(s))$:
Let $\lambda(t)\nu(t)$ be integrable on $[t_0, t]$. Then

$$\mathrm{E}\left[\exp\left(\int_{t_0}^{t}\ln(1+\nu(s))dP(s)\right)\right] \stackrel{\text{ifa}}{=} \exp\left(\int_{t_0}^{t}\lambda(s)\nu(s)ds\right). \quad (5.3)$$

Sketch of Proof:

Again, to keep the justification reasonably brief and maintain the usefulness as an integration technique, the stochastic jump integral will first be formally decomposed into a forward Itô sum, averaged and then recomposed back into a deterministic integral. The justification of each step will be indicated in shorthand on the sign of the relation, but the more rigorous Itô limits will be omitted.

Again it is remarkable that we can reduce this expectation of the exponential of a jump integral to such an explicit form.

Again, let $t_i = t_0 + i * \Delta t$ for $i = 0 : n+1$ be a proper partition of $[t_0, t]$ with $\Delta t = (t - t_0)/(n+1)$, $\Delta P_i = P(t_{i+1}) - P(t_i)$, $\lambda_i = \lambda(t_i)$ and $\nu_i = \nu(t_i)$ for $i = 0 : n$, so

$$\mathbf{E}\left[\exp\left(\int_{t_0}^{t} \ln(1+\nu(s))dP(s)\right)\right] \overset{\text{ifa}}{\simeq} \mathbf{E}\left[\exp\left(\sum_{i=0}^{n} \ln(1+\nu_i)\Delta P_i\right)\right]$$

$$\overset{law}{\underset{exp}{=}} \mathbf{E}\left[\prod_{i=0}^{n} \exp\left(\ln(1+\nu_i)\Delta P_i\right)\right]$$

$$\overset{\text{ind}}{\underset{\text{inc}}{=}} \prod_{i=0}^{n} \mathbf{E}_{\Delta P_i}\left[\exp\left(\ln(1+\nu_i)\Delta P_i\right)\right]$$

$$\overset{pois}{\underset{dist}{=}} \prod_{i=0}^{n} \sum_{k=0}^{\infty} \exp(-\lambda_i \Delta t) \frac{(\lambda_i \Delta t)^k}{k!} (1+\nu_i)^k$$

$$\overset{exp}{\underset{sum}{=}} \prod_{i=0}^{n} \exp(-\lambda_i \Delta t + \lambda_i \Delta t(1+\nu_i))$$

$$\overset{\text{ifa}}{\simeq} \exp\left(\int_{t_0}^{t} \lambda(s)\nu(s)ds\right). \quad \square$$

Using the **diffusion and jump Lemmas 5.1.1 and 5.1.2, remarkable for turning the expectation of the exponential into the exponential of a regular integral when the coefficients are variable,** the expectation of the state trajectory $X(t)$ in (4.37) on p. L4-p50 for the financially relevant linear SDE with time-dependent coefficients in (4.34) on p. L4-p48 can be readily calculated, as follows.

Theorem 5.1.3 Expectation of $X(t)$ in the Linear Jump-Diffusion SDE with Time-Dependent Coefficients Case:
Let $\mu(t)$, $\sigma^2(t)$ and $\lambda(t)\nu(t)$ be integrable on $[t_0, t]$. Then

$$\begin{aligned}\mathrm{E}[X(t)] &= \mathrm{E}\left[x_0 \exp\left(\int_{t_0}^{t}\left((\mu(s)-\sigma^2(s)/2)ds+\sigma(s)dW(s)\right.\right.\right.\\ &\qquad \left.\left.\left.+\ln(1+\nu(s))dP(s)\right)\right)\right] \qquad (5.4)\\ &\overset{\text{ifa}}{=} x_0 \exp\left(\int_{t_0}^{t}\left(\mu(s)+\lambda(s)\nu(s)\right)ds\right).\end{aligned}$$

Proof: The proof is left as an algebraic homework exercise for the reader, using Lemmas 5.1.1—5.1.2 on pages L5-p21 to L5-23.

For the corresponding variance $\mathbf{Var}[\boldsymbol{X}(t)]$ is left as a combined homework exercise. Note that the expectation and the variance results for the time-dependent case easily reduce to the financially relevant linear SDE, constant coefficients results given in (4.40) on L4-p52 for the expectation and (4.41) on L4-p54 for the variance when the coefficients $\boldsymbol{\mu}(t)$, $\boldsymbol{\nu}(t)$ and $\boldsymbol{\sigma}(t)$ become constant.

5.2.4. SDE Models Exactly Transformable to Purely Time-Varying Coefficients: In this section, a catalogue of exactly transformable jump-diffusion SDE models is given. First the notational correlations are listed for ease of interpreting the list of models and their transformations, where conditions are applicable:

- **SDE Models and Their Transformations:**
  - **Original SDE (4.23 on L4-p31):**

$$dX(t) = f(X(t),t)dt + g(X(t),t)dW(t) + h(X(t),t)dP(t).$$

  - **Transformed Process:** $Y(t) = F(X(t),t)$.
  - **Transformed SDE:**

$$dY(t) = (F_t + F_x f + \tfrac{1}{2}F_{xx}g^2)dt + F_x g dW(t) + [F]dP(t).$$

  - **Target, Stateless and Explicit SDE:**

$$dY(t) = C_1(t)dt + C_2(t)dW(t) + C_3(t)dP(t).$$

- **Target, Stateless Coefficients in terms of Original:**

$$C_1(t) = F_t + F_x f + \tfrac{1}{2} F_{xx} g^2;$$

$$C_2(t) = F_x g;$$

$$C_3(t) = [F] \equiv F(x + h(x,t), t) - F(x,t).$$

- **Original Coefficients as Transformed:**

$$f(x,t) = (C_1(t) - F_t(x,t) - \tfrac{1}{2} F_{xx}(x,t) C_2^2(t) / F_x^2(x,t)) / F_x(x,t);$$

$$g(x,t) = C_2(t) / F_x(x,t);$$

$$h(x,t) = -x + F^{-1}(F(x,t) + C_3(t)).$$

Table 1: Example transforms original coefficients in terms of target and transform coefficients.

Transform $Y \rightarrow F(x,t)$	**Drift or Plant Coefficient** $f(x,t)$	**Gaussian Coefficient** $g(x,t)$	**Poisson Coefficient** $h(x,t)$
x	$C_1(t)$	$C_2(t)$	$C_3(t)$
$a(t)x+b(t)$	$\frac{C_1(t)-a'(t)x-b'(t)}{a(t)}$	$\frac{C_2(t)}{a(t)}$	$\frac{C_3(t)}{a(t)}$
$a(t)x^2$	$\frac{C_1(t)-a'(t)x^2-\frac{C_2^2(t)}{4a(t)x^2}}{2a(t)x}$	$\frac{C_2(t)}{2a(t)x}$	$-x\pm\sqrt{x^2+\frac{C_3(t)}{a(t)}}$
$\frac{a(t)}{x+b(t)}$	$\frac{C_2^2(x+b)^3}{a^2}-\frac{C_1(x+b)^2}{a}+\frac{a'(x+b)}{a}-b'$	$-\frac{C_2(x+b)^2}{a}$	$-\frac{C_3(x+b)^2}{C_3(x+b)+a}$
$a(t)e^{b(t)x}$	$-(\frac{a'}{ab}+\frac{b'x}{b})+\frac{C_1e^{-bx}}{ab}-\frac{1}{2}\frac{C_2^2e^{-2bx}}{a^2b}$	$\frac{C_2e^{-bx}}{ab}$	$\frac{1}{b}\ln\left(\frac{C_3e^{-bx}}{a}+1\right)$
$a(t)\ln(x)+b(t)$	$\left(\frac{C_1}{a}+\frac{C_2^2}{2a^2}-\frac{a'}{a}\ln(x)-\frac{b'}{a}\right)x$	$\frac{C_2}{a}x$	$\left(e^{C_3/a}-1\right)x$

See Kloeden and Platen's 1992 book for more diffusion examples. Also, see the textbook for an additional, linear-fractional transform.

5.2. Compound (Space-Time) Poisson Jump-Diffusion Stochastic Calculus:

(beginning Chapt. 5 of the textbook)

5.2.0. Compound (Space-Time) Poisson Processes:

Space-time Poisson processes are also called general compound Poisson processes, marked Poisson point processes and Poisson noise with randomly distributed jump-amplitudes conditioned on a Poisson jump in time. The marked adjective refers to marks which are the underlying stochastic process for the Poisson jump-amplitude or the space component of the space-time Poisson process, whereas the jump-amplitudes of the simple Poisson process are deterministic or fixed with unit magnitude. The compound or space-time Poisson process is a generalization of the Poisson process.

The compound or space-time Poisson process formulation is needed because jump-amplitude come in many random sizes in financial market and other real situations.

- **5.2.1. Properties of Compound Processes**:
  - **Space-time Poisson differential process:** The basic space-time or mark-time Poisson differential process denoted as

$$d\Pi(t) = \int_{\mathcal{Q}} h(t,q)\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}}) \tag{5.5}$$

on the **Poisson mark space** $\mathcal{Q}$ can be defined using the **Poisson random measure** $\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}})$, which is shorthand measure notation for the measure-set equivalence $\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}}) = \mathcal{P}([t,t+dt),[q,q+dq))$. The jump-amplitude $h(t,q)$ is assumed to be continuous and bounded in its arguments.

- **Poisson mark Q:** The space Poisson mark Q is the underlying **independent, identically distributed (IID) random variables** for the mark-dependent jump-amplitude coefficient denoted by $h(t,Q)$ **(not $h(t,Q)=1$ typo)**, i.e., the space part of the space-time Poisson process. The realized variable $Q=q$ is used in expectations or conditional expectations, as well as in definition of the type (5.5).

- **Time-integrated, space-time Poisson process:**

$$\Pi(t)=\int_0^t\int_{\mathcal{Q}} h(t,q)\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}}). \tag{5.6}$$

- **Unit jumps:** However, if the jumps have unit amplitudes, $h(t,Q) \equiv 1$, then the space time process in (5.5) must be the same as the simple differential Poisson process $dP(t;Q)$ modified with a mark parameter argument to allow for generating mark realizations, and we must have the equivalence

$$\int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) \equiv dP(t;Q), \tag{5.7}$$

giving the jump number count on the right-continuous time interval $[t, t+dt)$ and the parameter Q is redundant here. Integrating both sides of (5.7) on $[0,t]$ gives the jump-count up to time t,

$$\int_0^t \int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) = \int_0^t dP(s;Q) = P(t;Q). \tag{5.8}$$

Further, in terms of Poisson random measure $\mathcal{P}(\overline{\mathrm{dt}}, \{1\})$ on the fixed set $\mathcal{Q} = \{1\}$, purely the number of jumps, in the right-continuous time interval $[t, t + dt)$, is obtained,

$$\int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) = \mathcal{P}(\overline{\mathrm{dt}}, \{1\}) = P(\overline{\mathrm{dt}}) = dP(t; 1) \equiv dP(t)$$

and the marks are irrelevant. Note, the set measure notation with $P\left(\overline{\mathrm{dt}}\right) = dP(t)$.

- **Purely time-dependent jumps:** If $h(t, Q) = h_1(t)$, then

$$\int_{\mathcal{Q}} h_1(t)\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) \equiv h_1(t)dP(t; Q). \qquad (5.9)$$

- **Compound Poisson process form:** An alternate form of the space-time Poisson process (5.6) that many may find more comprehensible is the marked generalization of the **simple Poisson process $P(t;Q)$, with IID random mark generation**, that is, the counting sum called the **compound Poisson process** or **marked point process**,

$$\Pi(t) = \sum_{k=1}^{P(t;Q)} h(T_k^-, Q_k), \tag{5.10}$$

where $h(T_k^-, Q_k)$ is the kth jump-amplitude, T_k^- is the prejump value of the kth random jump-time, Q_k is the corresponding random jump-amplitude mark realization.

{**Poisson random measure and compound Poisson are two forms for representing the same thing and while we do not emphasize measure theory much in this course, Poisson random measure passes the utility test by making some calculations easier.**}

For the special case that $P(t;Q)$ is zero the following reverse-sum convention is used,

$$\sum_{k=1}^{0} h(T_k^-, Q_k) \equiv 0 \text{ for any } h. \tag{5.11}$$

The corresponding differential process has the expectation,

$$\mathrm{E}[dP(t;Q)] = \lambda(t)dt,$$

although it is possible that the jump-rate is mark-dependent (see Øksendal and Sulem (2005), for example) so that

$$\mathrm{E}[dP(t;Q)] = \mathrm{E}_Q[\lambda(t;Q)]dt.$$

However, it will be assumed here that the jump-rate is mark-independent to avoid complexities with iterated expectations later.

- **Zero-one law compound Poisson differential process form:** Given the Poisson compound process form in (5.10), the corresponding **zero-one jump law** for the compound Poisson differential process is

$$d\Pi(t) = h(t, Q)dP(t; Q), \tag{5.12}$$

such that the jump in $\Pi(t)$ at $t = T_k$ is given by

$$[\Pi](T_k) \equiv \Pi(T_k^+) - \Pi(T_k^-) = h(T_k^-, Q_k). \tag{5.13}$$

For consistency with the Poisson random measure and compound Poisson process forms, it is necessary that

$$\int_0^t h(s, Q)dP(s; Q) = \int_0^t \int_{\mathcal{Q}} h(s, q)\mathcal{P}(\overline{\mathrm{ds}}, \overline{\mathrm{dq}})$$

$$= \sum_{k=1}^{P(t;Q)} h(T_k^-, Q_k).$$

So

$$\int_0^t dP(s;Q) = \int_0^t \int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{ds}}, \overline{\mathrm{dq}}) = P(t;Q)$$

and

$$dP(t;Q) = \int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}).$$

{Note that the selection of the random marks depends on the existence of the Poisson jumps and that the mechanism is embedded in $dP(t;Q)$ in the formulation of this book.}

- In the **Poisson random measure notation** $\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})$, the arguments $\overline{\mathrm{dt}}$ and $\overline{\mathrm{dq}}$ are semiclosed subintervals when these arguments are expanded,

$$\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) = \mathcal{P}([t, t+dt), [q, q+dq)).$$

These subintervals are closed on the left and open on the right due to the definition of the increment, leaving no overlap between differential increments and correspondings to the simple Poisson right continuity property that

$$\Delta P(t;Q) \to P(t^+;Q) - P(t^-;Q) \ \ as \ \ \Delta t \to 0^+,$$

so we can write $\Delta P(t;Q) = P([t, t+\Delta t];Q)$ and $dP(t;Q) = P([t, t+dt);Q)$. When $t_{n+1} = t$ and $t_{i+1} = t_i + \Delta t_i$, the covering set of intervals is $\{[t_i, t_i + \Delta t_i) \ \ for \ \ i = 0:n\}$ plus t.

If the marks Q are continuously distributed, then closed subintervals can also be used in the q argument. For the one-dimensional mark space $\mathcal{Q}$, $\mathcal{Q}$ can be a finite interval such as $\mathcal{Q} = [a, b]$ or an infinite interval such as $\mathcal{Q} = (-\infty, +\infty)$. Also, these subintervals are convenient in partitioning continuous intervals since they avoid overlap at the nodes.

- **$\mathcal{P}$ has independent increments** on nonoverlapping intervals in time t and marks q, i.e., $\mathcal{P}_{i,k} = \mathcal{P}([t_i, t_i + \Delta t_i), [q_k, q_k + \Delta q_k))$ is independent of $\mathcal{P}_{j,\ell} = \mathcal{P}([t_j, t_j + \Delta t_j), [q_\ell, q_\ell + \Delta q_\ell))$, provided that the time interval $[t_j, t_j + \Delta t_j)$ has no overlap with $[t_i, t_i + \Delta t_i)$ and the mark interval $[q_k, q_k + \Delta q_k)$ has no overlap with $[q_\ell, q_\ell + \Delta q_\ell)$. Recall that $\Delta P(t_i; Q) \equiv P(t_i + \Delta t_i; Q) - P(t_i; Q)$ is associated with the time interval $[t_i, t_i + \Delta t_j]$, open on the left since the process $P(t_i; Q)$ has been subtracted to form the increment.

- The **expectation of $\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})$** is

$$\mathbf{E}[\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})] = \Phi_Q(\overline{\mathrm{dq}})\lambda(t)dt \overset{\mathrm{gen}}{=} \phi_Q(q)dq\lambda(t)dt, \quad (5.14)$$

where, in detail,

$$\begin{aligned}\Phi_Q(\overline{\mathrm{dq}}) &= \Phi_Q([q, q+dq)) = \Phi_Q(q+dq) - \Phi_Q(q) \\ &= \mathbf{Prob}[Q \leq q+dq] - \mathbf{Prob}[Q \leq q] \\ &= \mathbf{Prob}[q < Q \leq q+dq] \overset{\mathrm{gen}}{=} \phi_Q(q)dq\end{aligned}$$

is the probability distribution measure of the Poisson amplitude mark in measure-theoretic notation corresponding to the mark distribution function $\Phi_Q(q)$. The corresponding mark density will be equal to $\phi_Q(q)$ if Q is continuously distributed. Generalized $\left(\mathrm{symbol} \overset{\mathrm{gen}}{=}\right)$ densities will be assumed for almost all distributions encountered in applications, **i.e., the densities exist even if the distribution is not differentiable**.

- **Conservation of Mark Probability:** It is also assumed that Φ_Q is a proper distribution **on** $\mathcal{Q}$,

$$\int_{\mathcal{Q}} \Phi_Q(\overline{\mathrm{dq}}) = \int_{\mathcal{Q}} \phi_Q(q)dq = 1.$$

- **Poisson random measure $\mathcal{P}(\Delta \mathrm{t}_i, \Delta \mathrm{q}_j)$ is Poisson distributed**, i.e.,

$$\mathrm{Prob}[\mathcal{P}(\Delta \mathrm{t}_i, \Delta \mathrm{q}_j) = k] = e^{-\overline{\mathcal{P}}_{i,j}} \left(\overline{\mathcal{P}}_{i,j}\right)^k / k!, \quad (5.15)$$

where

$$\begin{aligned}\overline{\mathcal{P}}_{i,j} &= \mathrm{E}[\mathcal{P}(\Delta \mathrm{t}_i, \Delta \mathrm{q}_j)] = \Phi_Q(\Delta \mathrm{q}_j) \textstyle\int_{\Delta \mathrm{t}_i} \lambda(t)dt \\ &= \Phi_Q(\Delta \mathrm{q}_j)\Lambda(\Delta \mathrm{t}_i)\end{aligned}$$

for sets $\Delta \mathrm{t}_i \equiv [t_i, t_i + \Delta t_i)$ in time and $\Delta \mathrm{q}_j \equiv [q_j, q_j + \Delta q_j)$ in marks.

Thus, as Δt_i and Δq_j approach 0^+, they can be replaced by dt and dq, respectively, so

$$\mathbf{Prob}[\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) = k] = e^{-\overline{\mathcal{P}}} \left(\overline{\mathcal{P}}\right)^k / k!, \quad (5.16)$$

where

$$\overline{\mathcal{P}} = \mathrm{E}[\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})] = \phi_Q(q) dq \lambda(t) dt,$$

so by the zero-one jump law,

$$\mathbf{Prob}[\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) = k] \underset{\mathbf{zol}}{\overset{\mathbf{dt}}{=}} (1 - \overline{\mathcal{P}})\delta_{k,0} + \overline{\mathcal{P}}\delta_{k,1}.$$

- **Expectation of $dP(t;Q) = \int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})$:**

$$\begin{aligned}\mathrm{E}\left[\int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})\right] &= \lambda(t)dt \int_{\mathcal{Q}} \phi_Q(q)dq = \lambda(t)dt \cdot 1 \\ &= \mathrm{E}[dP(t;Q)],\end{aligned} \tag{5.17}$$

corresponding to the earlier Poisson equivalence (5.7) and using the above proper distribution property. Similarly,

$$\mathrm{E}\left[\int_0^t \int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{ds}}, \overline{\mathrm{dq}})\right] = \mathrm{E}[P(t;Q)] = \int_0^t \lambda(s)ds = \Lambda(t).$$

- **Variance of $\int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}}) \equiv dP(t;Q)$:** By definition,

$$\mathrm{Var}\left[\int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})\right] = \mathrm{Var}[dP(t;Q)] = \lambda(t)dt. \tag{5.18}$$

- **Covariance of $\int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})$:** Since

$$\mathrm{Var}\left[\int_{\mathcal{Q}} \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}})\right] = \int_{\mathcal{Q}}\int_{\mathcal{Q}} \mathrm{Cov}[\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}_1}), \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}_2})],$$

then

$$\mathrm{Cov}[\mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}_1}), \mathcal{P}(\overline{\mathrm{dt}}, \overline{\mathrm{dq}_2})] \stackrel{\mathrm{gen}}{=} \lambda(t)dt\phi_{\mathcal{Q}}(q_1) \cdot\delta(q_1 - q_2)dq_1 dq_2, \tag{5.19}$$

analogous to (1.48) on p. 22 in textbook, for $\mathrm{Cov}[dP(s_1), dP(s_2)]$, but omitted from Lecture 1.

- **Finite Second Order Moments for Jump-Amplitude Function h:** Let

$$\int_{\mathcal{Q}} |h(t,q)|^2 \phi_Q(q) dq < \infty \tag{5.20}$$

for all $t \geq 0$ and, in particular,

$$\int_0^t \int_{\mathcal{Q}} |h(s,q)|^2 \phi_Q(q) dq \lambda(s) ds < \infty. \tag{5.21}$$

{In this class, we treat these as technical conditions, unlikely to be violated by realistic $\{h(t,q)$ & $\phi_Q(q)\}$ functions and infinite time is only a purely mathematical limit.}

- **Poisson Ransom Measure and Compound Poisson Process Equivalence:** From Theorem {2.2.4, L2-p47} or {3.12; (3.12), p.70, textbook},

$$\int_0^t \int_{\mathcal{Q}} h(s,q)\mathcal{P}(\overline{\mathrm{ds}},\overline{\mathrm{dq}}) = \sum_{k=1}^{P(t;Q)} h(T_k^-, Q_k), \quad (5.22)$$

i.e., the jump-amplitude counting version of the space-time integral, where T_k is the kth jump-time of a Poisson process $P(t;Q)$ and provided comparable assumptions are satisfied. This is also consistent for the infinitesimal counting sum form in (5.10) and the convention (5.11) applies for (5.22). The form (5.22) is somewhat awkward due to the presence of three random variables, $P(t;Q)$, T_k and Q_k, requiring multiple iterated expectations. {See the comments on L5-p35.}

- **Time-Independent Jump-Amplitude, $h(t,q)=h_2(q)$, for Compound Poisson Processes:** Then

$$\begin{aligned}\Pi_2(t) &= \int_0^t \int_{\mathcal{Q}} h_2(q)\mathcal{P}(\overline{\mathrm{ds}}, \overline{\mathrm{dq}}) \\ &= \int_{\mathcal{Q}} h_2(q)\mathcal{P}([0,t), \overline{\mathrm{dq}}) = \sum_{k=1}^{P(t;Q)} h_2(Q_k),\end{aligned} \tag{5.23}$$

  where the sum is zero when $P(t;Q)=0$, the jump-amplitudes $h_2(Q_k)$ form a set of IID random variables independent of the jump-times of the Poisson process $P(t;Q)$. The simplest case is when $h(t,q)=q$ and that is usually the desirable form after reducing the SDE to state-independent coefficients.

• **Mean by Double-Iterated Expectations for Time-Independent Jump-Amplitude and Mark-Independent Jump-Rate (dropping the picked parameter for simplicity, $P(t;Q)=P(t)$):** Using $\Lambda(t)\equiv\int_0^t\lambda(s)ds$,

$$\begin{aligned}\mathrm{E}[\Pi_2(t)]&=\mathrm{E}_{P(t)}\left[\mathrm{E}_Q\left[\sum_{k=1}^{P(t)}h_2(Q_k)|P(t)\right]\right]\\&\overset{\text{iid}}{=}\mathrm{E}_{P(t)}\left[\sum_{k=1}^{P(t)}\mathrm{E}_Q[h_2(Q)]\right]\\&=\mathrm{E}_{P(t)}\left[P(t)\mathrm{E}_Q[h_2(Q)]\right]=\overline{h}_2\Lambda(t),\end{aligned}$$

where the IID property has been used, i.e., $E_Q[h_2(Q_k)]\overset{\text{iid}}{=}\mathrm{E}_Q[h_2(Q)]\equiv\overline{h}_2$; **also** $\sum_{k=1}^{P(t)}C=CP(t)$, **any constant C, and** $\mathrm{E}_{P(t)}[P(t)]=\Lambda(t)$.

Variance by Double-Iterated Expectations for Time-Independent Jump-Amplitude and Mark-Independent Jump-Rate ($P(t;Q)=P(t)$):

Then,

$$
\begin{aligned}
\mathbf{Var}[\Pi_2(t)] &= \mathbf{E}\left[\left(\sum_{k=1}^{P(t)} h_2(Q_k) - \overline{h}_2 \Lambda(t)\right)^2\right] \\
&= \mathbf{E}\left[\left(\sum_{k=1}^{P(t)} \left(h_2(Q_k) - \overline{h}_2\right) + \overline{h}_2 (P(t) - \Lambda(t))\right)^2\right] \\
&= \mathbf{E}_{P(t)}\Big[\sum_{k_1=1}^{P(t)} \sum_{k_2=1}^{P(t)} \mathbf{E}_Q\left[\left(h_2(Q_{k_1}) - \overline{h}_2\right)\left(h_2(Q_{k_2}) - \overline{h}_2\right)\right] \\
&\quad + 2\overline{h}_2 (P(t) - \Lambda(t)) \sum_{k=1}^{P(t)} \mathbf{E}_Q\left[h_2(Q_k) - \overline{h}_2\right] \\
&\quad + \overline{h}_2^2 (P(t) - \Lambda(t))^2\Big] \\
&= \mathbf{E}_{P(t)}\Big[P(t)\mathbf{Var}_Q[h_2(Q)] + 2\overline{h}_2 (P(t) \\
&\quad - \Lambda(t)) P(t) \cdot 0 + \overline{h}_2^2 (P(t) - \Lambda(t))^2\Big] \\
&= \left(\mathbf{Var}_Q[h_2(Q)] + \overline{h}_2^2\right) \Lambda(t) = \mathbf{E}_Q\left[h_2^2(Q)\right] \Lambda(t),
\end{aligned}
$$

using the IID property, **separation into mean-zero forms (!)** and the variance-expectation identity (B.186).

- **Compound Poisson Processes with Both Time- and Mark-Dependence, $h(t,q)$ and $\lambda(t;q)$:**

  Then

$$\Pi(t) = \int_0^t \int_{\mathcal{Q}} h(s,q)\mathcal{P}(\overline{\mathrm{ds}}, \overline{\mathrm{dq}}) = \sum_{k=1}^{P(t;Q)} h(T_k^-, Q_k); \quad (5.24)$$

  however, the **iterated expectations technique is not very useful for the compound Poisson form**, due to the additional dependence introduced by the jump-time T_k and the jump-rate $\lambda(t;q)$.

- **Compare Flexibility of Poisson Random Measure Form for Both Time- and Mark-Dependence:**

$$\begin{aligned}\mathrm{E}[\Pi(t)] &= \mathrm{E}\left[\int_0^t \int_{\mathcal{Q}} h(s,q)\mathcal{P}(\overline{\mathrm{ds}}, \overline{\mathrm{dq}})\right] \\ &= \int_0^t \int_{\mathcal{Q}} \lambda(s,q)h(s,q)\phi_Q(q)dq\ ds \\ &= \int_0^t \mathrm{E}_Q[\lambda(s,Q)h(s,Q)]ds.\end{aligned}$$

- **Theorem 5.2.1. Basic Infinitesimal Moments of the Space-Time Poisson Process:**

$$\begin{aligned}\mathrm{E}[d\Pi(t)] &= \lambda(t)dt\int_{\mathcal{Q}} h(t,q)\phi_Q(q)dq \\ &\equiv \lambda(t)dt\mathrm{E}_Q[h(t,Q)] \equiv \lambda(t)dt\overline{h}(t)\end{aligned} \tag{5.25}$$

and

$$\begin{aligned}\mathrm{Var}[d\Pi(t)] &= \lambda(t)dt\int_{\mathcal{Q}} h^2(t,q)\phi_Q(q)dq \\ &= \lambda(t)dt\mathrm{E}_Q[h^2(t;Q)] \equiv \lambda(t)dt\overline{h^2}(t).\end{aligned} \tag{5.26}$$

Proof: The jump-amplitude function $h(t,Q)$ **is independently distributed**, through the mark process Q, from the underlying Poisson counting process here, except that this **jump in space is conditional on the occurrence of the jump-time or -count of the underlying Poisson process**.

However, the function $h(t,q)$ **is deterministic** since it depends on the realization q in the space-time Poisson definition, rather than the random variable Q.

The **infinitesimal mean** (5.25) is straightforward:

$$\begin{aligned}\mathrm{E}[d\Pi(t)] &= \mathrm{E}\left[\int_{\mathcal{Q}} h(t,q)\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}})\right] = \int_{\mathcal{Q}} h(t,q)\mathrm{E}[\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}})] \\ &= \lambda(t)dt\int_{\mathcal{Q}} h(t,q)\phi_Q(q)dq = \lambda(t)dt\,\mathrm{E}_Q[h(t,Q)] \\ &\equiv \lambda(t)dt\,\overline{h}(t);\end{aligned}$$

note that the expectation operator applied to the mark integral can be moved to apply just to the Poisson random measure $\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}})$, since $h(t,q)$ is deterministic.

However, the result for the **infinitesimal variance** in (5.26) is not so obvious, but the covariance formula for two Poisson random measures with differing mark variables $\mathbf{Cov}[\mathcal{P}(\overline{\mathbf{dt}}, \overline{\mathbf{dq_1}}), \mathcal{P}(\overline{\mathbf{dt}}, \overline{\mathbf{dq_2}})]$ in (5.19) will be made useful by converting it to the mean-zero Poisson random measure

$$\begin{aligned}\widetilde{\mathcal{P}}(\overline{\mathbf{dt}}, \overline{\mathbf{dq}}) \equiv & \mathcal{P}(\overline{\mathbf{dt}}, \overline{\mathbf{dq}}) - \mathbf{E}[\mathcal{P}(\overline{\mathbf{dt}}, \overline{\mathbf{dq}})] \\ = & \mathcal{P}(\overline{\mathbf{dt}}, \overline{\mathbf{dq}}) - \phi_{\boldsymbol{Q}}(q) dq \boldsymbol{\lambda}(t) dt\end{aligned} \tag{5.27}$$

$$
\begin{aligned}
\mathbf{Var}[d\Pi(t)] &= \mathbf{E}\left[\left(\int_{\mathcal{Q}} h(t,q)\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}}) - \overline{h}(t)\lambda(t)dt\right)^2\right] \\
&= \mathbf{E}\left[\left(\int_{\mathcal{Q}} \left(h(t,q)\mathcal{P}(\overline{\mathrm{dt}},\overline{\mathrm{dq}}) - h(t,q)\phi_Q(q)\lambda(t)dt\right)\right)^2\right] \\
&= \mathbf{E}\left[\left(\int_{\mathcal{Q}} h(t,q)\widetilde{\mathcal{P}}(\overline{\mathrm{dt}},\overline{\mathrm{dq}})\right)^2\right] \\
&= \mathbf{E}\left[\int_{\mathcal{Q}} h(t,q_1)\int_{\mathcal{Q}} h(t,q_2)\widetilde{\mathcal{P}}(\overline{\mathrm{dt}},\overline{\mathrm{dq_1}})\widetilde{\mathcal{P}}(\overline{\mathrm{dt}},\overline{\mathrm{dq_2}})\right] \\
&= \int_{\mathcal{Q}} h(t,q_1)\int_{\mathcal{Q}} h(t,q_2)\mathbf{Cov}\left[\widetilde{\mathcal{P}}(\overline{\mathrm{dt}},\overline{\mathrm{dq_1}}),\widetilde{\mathcal{P}}(\overline{\mathrm{dt}},\overline{\mathrm{dq_2}})\right] \\
&= \int_{\mathcal{Q}} h(t,q_1)\int_{\mathcal{Q}} h(t,q_2)\delta(q_2-q_1)\phi_Q(q_1)dq_1\lambda(t)dt \\
&= \lambda(t)dt\int_{\mathcal{Q}} h^2(t,q_1)\phi_Q(q_1)dq_1 \\
&= \lambda(t)dt\mathbf{E}_Q\left[h^2(t,Q)\right] \equiv \lambda(t)dt\overline{h^2}(t). \qquad \square
\end{aligned}
$$

- **5.2.2. Examples of Compound Process Jump-Amplitude Mark Distributions**: For finance models these arise from linear models, where $h = h(x,q) = \nu(q)x$ such the log-transformed jump-amplitude is chosen such that $\ln(1+\nu(q)) \equiv q$, suppressing the time-dependence of the IID RVs for simplicity, except for the domain $[a,b]$, either finite or infinite. The inverse is $\nu(q) = e^q - 1$.
  - **Log-Uniformly Distributed Jump-Amplitudes:** The uniform distribution is and example of a finite range, continuous distribution for the jump-amplitude mark Q,

$$\phi_Q^{(\mathrm{uq})}(q) = \frac{1}{b-a} U(q;(a,b)),\ a < 0 < b, \quad (5.28)$$

where $U(q;(a,b)) = 1_{q\in(a,b)}$ is the step or indicator function for a finite **crash to rally** interval $[a,b]$, i.e., $U(q;(a,b))$ is one when $a < q < b$ and zero otherwise.

The first few uniform moments are

$$\mathbf{E}_Q^{(\mathbf{uq})}[1] = \frac{1}{b-a}\int_a^b dq = 1,$$

$$\mathbf{E}_Q^{(\mathbf{uq})}[Q] = \frac{1}{b-a}\int_a^b q dq = \frac{b+a}{2}$$

and

$$\mathbf{Var}_Q^{(\mathbf{uq})}[Q] = \frac{1}{b-a}\int_a^b \left(q - \frac{b+a}{2}\right)^2 dq = \frac{(b-a)^2}{12}.$$

The expected jump-amplitude is

$$\mathbf{E}_Q^{(\mathbf{uq})}[\nu(Q)] = \frac{1}{b-a}\int_a^b (e^q - 1) dq = \frac{e^b - e^a}{b-a} - 1.$$

{**Note that time-dependence can be introduced with out destroying the simplicity of the IID RV formulation by using $[a(t), b(t)]$ as the domain. The market jump distribution does vary over time periods, some periods more noisy than others.**}

- **Log-Double-Uniformly Distributed Jump-Amplitudes:** Since the psychology of the bull and bear markets, dominating the crashes and the rallies, respectively, it is desirable to separate the loss side of the log-return marks from the gain side, modifying the single uniform distribution. The double-uniform distribution is also a finite range distribution, as is the market distribution, for the jump-amplitude mark Q given by

$$\phi_Q^{(\mathrm{duq})}(q) = \frac{p_1}{|a|}U(q;(a,0)) + \frac{p_2}{b}U(q;(0,b)), \quad (5.29)$$

where $a<0<b$, the crash-rally probabilities satisfy $p_1+p_2=1$, $(a,0)$ is the finite crash or loss interval and $(0,b)$ is the finite rally or gain interval, whereas the double-uniform density is zero otherwise.

Obviously, $\mathbf{E}_{Q}^{(\mathbf{duq})}[Q] = 1$, while basic statistics are

$$\mathbf{E}_{Q}^{(\mathbf{duq})}[Q] = \frac{1}{2}(p_2 b + p_1 a)$$

and

$$\mathbf{Var}_{Q}^{(\mathbf{duq})}[Q] = \frac{1}{3}(p_2 b^2 + p_1 a^2) - \frac{1}{4}(p_2 b + p_1 a)^2.$$

The log-uniform and log-double-uniform mark distributions have been used in a variety of financial applications since their finiteness is consistent with the finiteness of market distributions. They have the fattest of tails, in fact are all tail, appropriate for market data that exhibit non-normal data through fat tails. They are the simplest of distributions whereas it is extremely difficult to extract any jump-amplitude distribution from the market data.

- **Log-Double-Exponentially Distributed Jump-Amplitudes:**
  The double-exponential mark distribution has been used by S. G. Kou of Columbia University along with co-workers. This distribution in on the infinite domain and leads to some closed forms like the normal distribution, both are called Laplace distributions. The double-exponential has the form

$$\phi_Q^{(\mathrm{deq})}(q) = \frac{p_1}{|\mu_1|} e^{q/|\mu_1|} U(q;(-\infty,0)) + \frac{p_2}{\mu_2} e^{-q/\mu_2} U(q;(0,\infty)), \tag{5.30}$$

  where $\mu_1 < 0 < \mu_2$, the crash-rally probabilities satisfy $p_1 + p_2 = 1$, $(-\infty, 0)$ is the infinite crash or loss interval and $(0, \infty)$ is the infinite rally or gain interval, whereas the double-exponential density is zero otherwise.

By construction, $\mathbf{E}_Q^{(\mathbf{deq})}[Q] = 1$, while basic statistics are

$$\mathbf{E}_Q^{(\mathbf{deq})}[Q] = p_2\mu_2 + p_1\mu_1$$

and

$$\mathbf{Var}_Q^{(\mathbf{deq})}[Q] = 2(p_2\mu_2^2 + p_1\mu_1^2) - (p_2\mu_2 + p_1\mu_1)^2.$$

However, despite the appealing analytical properties of the double exponential, the genuine exponentials tails are inconsistent with the fat tail properties of real market log-return distribution. **Thus, tail-wise,**

$$\exp(-|q/\mu|) \ll 1, \ \ for \ \ |q| \gg 1,$$

i.e., the double exponential tails are much thinner than the uniform tails.

- **Normally Distributed Jump-Amplitudes:**
  In his pioneering jump-diffusion finance paper, Merton (1976) used lognormally distributed jump-amplitudes for the compound Poisson component. This was another early fix of the Black-Scholes (1973) model, beyond his removal of constant coefficients and one-dimensionality assumptions in his companion justification paper (1973). The normal distribution has the usual form,

$$\phi_Q^{(\mathrm{nq})}(q) = \frac{e^{-0.5(q-\mu^{(\mathrm{nq})})^2/(\sigma^{(\mathrm{nq})})^2}}{\sqrt{2\pi(\sigma^{(\mathrm{nq})})^2}}, \qquad (5.31)$$

  so $\mathbf{E}_Q^{(\mathrm{nq})}[Q] = \mu^{(\mathrm{nq})}$ and $\mathbf{Var}_Q^{(\mathrm{nq})}[Q] = (\sigma^{(\mathrm{nq})})^2$. **The thinnest of the far tails are even much smaller than the other two examples since**

$$e^{-0.5(q-\mu)^2/\sigma^2} \ll e^{-|q/\mu|} \ll 1, \ \ for \ \ |q| \gg 1.$$

END PRESENTED LECTURE 5 HERE!!!

- **Normal-Uniform Hybrid Marks:**
  The very, very thin tails of the normal is consequence of the insistence on infinite domain for exact integrals and for using the large number of statistical tests and methods available. Just truncating the normal to finite range does not fatten the tails noticeably. However, an alternate idea is the combine the truncated normal and the uniform distribution, i.e.,

$$\phi_Q^{(\mathrm{nuq})}(q) = \left(\frac{p_u}{b-a} + \frac{p_n \phi_n(q;\mu_n,\sigma_n^2)}{\Phi_n(a,b;\mu_n,\sigma_n^2)} \right) U(q;(a,b)), \quad (5.32)$$

  where $a<0<b$, $\Phi_n(a,b;\mu_n,\sigma_n^2)$ is the distribution on (a,b), while p_u and p_n are the respective uniform and normal probabilities such that $p_u+p_n=1$.

- **MATLAB Mark Simulations:**
  - **Uniform on (a,b):**
    $Q^{(\mathrm{uq})}$ `=a+(b-a)*rand=unifrnd(a,b)`.
  - **Normal for (μ,σ):**
    $Q^{(\mathrm{nq})}$ `=mu+sigma*randn=normrnd(mu,sigma)`.
  - **Double-Uniform for $(a<0<b)$:**
    $Q^{(\mathrm{duq})}$ `=bin0(1,p1)*unifrnd(a,0)`
    `+(1-bin0(1,p1))*unifrnd(0,b)`.
  - **Double-Exponential for $(\mu_1<0<\mu_2)$:**
    $Q^{(\mathrm{deq})}$ `=-bin0(1,p1)*exprnd(-mu1)`
    `+(1-bin0(1,p1))*exprnd(mu2)`.
  - **Normal-Uniform for $(a<0<b,\mu,\sigma)$:**

$Q^{(\mathrm{nuq})}$ `=bin0(1,pu)*unifrnd(a,b)+(1-bin0(1,pu))`
`*Accepted[normrnd(mu,sigma);(a,b)]`.

- **Example — Linear Mark-JD Simulatorfor Log-Uniformly Distributed Jump-Amplitudes:**
  The linear SDE jump-diffusion simulator MATLAB code `linjumpdiff03fig1.m` in Online Appendix C can be converted from the simple discrete jump process to the distributed jump process here. The primary change is the generation of another set of random numbers for the mark process Q, e.g.,

$$Q = a + (b - a) * \texttt{rand}(1, n + 1)$$

  for a set of $n + 1$ uniformly distributed marks on (a, b) so that the jump-amplitudes of $X(t)$ are log-uniformly distributed.

An example is demonstrated in Figure **??** for uniformly distributed marks Q on $(a,b) = (-2,+1)$ and time-dependent coefficients $\{\mu_d(t), \sigma_d(t), \lambda(t)\}$. The MATLAB linear mark-jump-diffusion code ***linmarkjumpdiff09fig1.m*** , vectorization revision of **`linmarkjumpdiff06fig1.m`** found in Online Appendix C, is a modification of the pure linear jump-diffusion SDE simulator code **`linjumpdiff03fig1.m`** in Figure 5.1 for constant coefficients and discrete mark-independent jumps.

The state exponent $Y(t)$ is simulated, and it is best to simulate $Y(t)$ not $X(t)$, with constant time-steps Δt as

$$YS(i+1) = YS(i) + (\mu(i) - \sigma^2(i)/2) * \Delta t + \sigma(i) * DW(i) + Q(i) * DP(i)$$

with $t(i+1) = t0 + i * \Delta t$ for $i = 1 : n+1$ with $n = 1000$, $t0 = 0$, $0 \le t(i) \le T = 2$ where $Q(i) = \ln(1+\nu(i))$ and $X(i) = x0 * \exp(Y(i))$.

The mean state is calculated by exponent

$$YM(i+1) = YM(i) + (\mu(i) + \lambda(i) * \overline{\nu}) * \Delta t$$

where $\overline{\nu} \equiv \mathrm{E}[Q]$ and $XM(i) = x0 * \exp(YM(i))$.

The **`cumsum`** can be used to handle the above $i \longrightarrow i+1$ recursions so primarily vector code can be obtained.

The incremental Poisson jump term $\Delta P(t_i) = P(t_i + \Delta t) - P(t_i)$ is simulated by a binomial RNG **`binornd`** for count **`n=1`** and time-dependent vector parameter Λ, i.e., the Bernoulli 0-1 process. In the older code used a uniform random number generator on $(0, 1)$ using the **acceptance-rejection technique** (see p. 270, text) to implement the zero-one jump law to obtain the probability of $\lambda(i)\Delta t$ that a jump is accepted there. The same random state is used to obtain the simulations of uniformly distributed Q on (a, b) conditional on a jump event. This technique is useful in problems for which the Statistics Toolbox does not have an appropriate RNG.

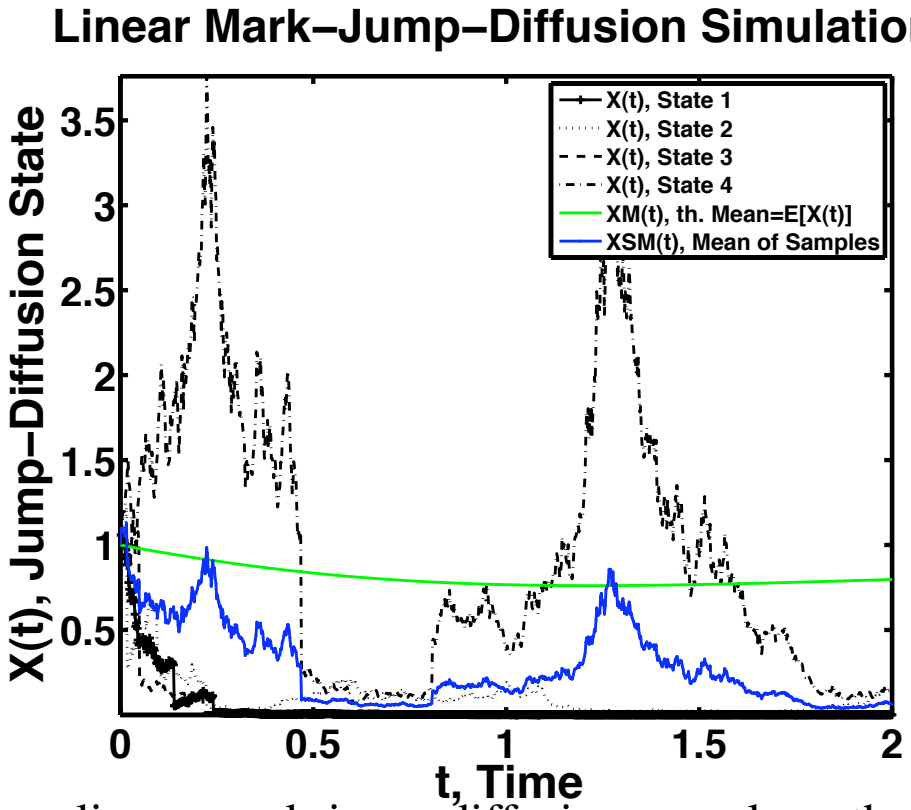

Figure 5.4: Four linear mark-jump-diffusion sample paths for time-dependent coefficients are simulated using MATLAB with $N = 1,000$ time-steps, maximum time $T = 2.0$ and four `randn` and vector `binornd` states. Initially, $x_0 = 1.0$. Parameter values are given in vectorized functions using vector functions and dot-element operations, $\mu_d(t) = 0.1 * \sin(t)$, $\sigma_d(t) = 1.5 * \exp(-0.01 * t)$ and $\lambda = 3.0 * \exp(-t. * t)$. The marks are uniformly distributed on $[-2.0, +1.0]$.

- ***Linear Mark-Jump-Diffusion Simulations for Variable Coefficients, revised book MATLAB code example (edited)***:

```
function linmarkjumpdiff09fig1
% Revised Linear for Linear Marked-Jump-Diff. 10/09
%   SDE RNG Simulation with variable coefficients
%     for t in [0,T]with sample variation:
%    DX(t) = X(t)*(mu(t)*Dt + sig(t)*DW(t) + nu(Q)*DP(t),
%     X(0) = x0.
%   Or log-state:
%      DY(t)=(mu(t)-sig^2(t)/2)*Dt+sig(t)*DW(t)+Q*DP(t),
%      Y(0) = log(x0) and Q = ln(1+nu(Q)).
%   Generation is by summing Wiener increments DW
%     with Poisson jump increment added .
%   Sufficiently SMALL increments assumed,
%         so zero-one jump law;
%   For demonstration purposes, Q will be assumed to be
%     (qdist =1) UNIFORM on (qparm1,qparm2)=(a,b)
%     OR
```

```
%     (qdist=2) NORMAL  with (qparm1,qparm2)=(muj,sj2).
%   Allows Separate Driver Input and Special Jump
%       or Diffusion Handling.
clc % clear variables,
clf % clear figures
fprintf('\nfunction linmarkjumpdiff09fig1 OutPut:');
%%% Initialize input to jdsimulator with parameters:
N = 1000; t0 = 0; T = 2.0; % Set initial time grid:
idiff = 1; ijump = 1; x0 = 1.0;
qdist = 1;a = -2;b = +1;qparm1 = a;qparm2 = b; %Uniform
%OR E.G., Normal distribution:
%qdist = 2;muj = 0.28;sj2=+0.15;qparm1=muj;qparm2=sj2;
% set constant parameters.
fprintf('\nN=%i; x0=%6.3f;t0=%6.3f;T=%6.3f;',N,x0,t0,T);
fprintf('\nqdist=%i*; qparm1=%6.3f; qparm2=%6.3f;'...
    ,qdist,qparm1,qparm2);
fprintf('\n  * qdist=1 for uniform Q-distribution.');
fprintf('\n  * qdist=2 for normal Q-distribution.');
```

```
%
jdsimulator(idiff,ijump,qdist,qparm1,qparm2 ...
     ,N,x0,t0,T);
%
% END INPUT FOR JUMP-DIFFUSION SIMULATOR.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function jdsimulator(idiff,ijump,qdist,qparm1,qparm2 ...
    ,N,x0,t0,T)
nfig = 0; % initial figure counter.
NI = N + 1;  Dt = (T-t0)/NI;
t = t0:Dt:T; tv = t(1,1:NI); % Compute time vector;
fprintf('\nN=%i; NI=%i; length(t)=%i; ',N,NI,length(t));
kjd = 4 - 2*idiff - ijump;
NP = N + 2; % #plot_points = #time_steps + 1.
muv = mu(tv); % Get time-dependent coefficient vectors
if idiff == 1, sigv = sigma(tv); end
if ijump == 1, lamv = lambda(tv); end
if qdist==1 % Average nu(Q)=exp(Q)-1, UNIFORM Q-Dist.
```

```
    numean=(exp(qparm2)-exp(qparm1))/(qparm2-qparm1)-1;
elseif qdist==2 % Average nu(Q)=exp(Q)-1, NORMAL Q-Dist.
    numean=exp(qparm1-qparm2/2)-1;
end
sqrtDt = sqrt(Dt); % Standard Wiener increment moments.
sigsqrtDt = sqrtDt*sigv;
Lamv = Dt*lamv;
MuDt = Dt*muv;
MuldDt = MuDt - 0.5*sigsqrtDt.^2; % Get Ito correction.
% Begin Sample Path Calculation:
kstates = 4;
XS = zeros(kstates,NI+1); % declare global state vector.
DW = zeros(1,NI);
QDP = zeros(1,NI);
MS = zeros(1,NI+1);
MS(1,2:NI+1) = cumsum(MuldDt);
% Compute Mean State Path:
MSMDt = MuDt+numean*Lamv; % Mean exponent;
```

```
YM = zeros(1,NI+1);
YM(1,2:NI+1) = cumsum(MSMDt);
XM = x0*exp(YM);
for kstate = 1:kstates % Test Simulated Sample Paths:
    if idiff == 1
      randn('state',kstate); % Set initial normal state
      DW = sigsqrtDt.*randn(1,NI);%sig*sqrt(Dt)*dP, NRNG
    end
    if ijump == 1
      if qdist == 1 %Generate Uniform mark vector Q.
        Q = qparm1+(qparm2-qparm1)*rand(1,NI);
      elseif qdist == 2 %Generate Normal mark vector Q.
        sj = sqrt(qparm2); Q = qparm1+sj*randn(1,NI);
      end
      QDP = Q.*binornd(1,Lamv,1,NI); % Q.*DP(t), 0-1.
    end
    WS = zeros(1,NI+1);
    PS = zeros(1,NI+1);
```

```
    WS(1,2:NI+1) = cumsum(DW);% Set Sums
    PS(1,2:NI+1) = cumsum(QDP);
    YS = MS + WS + PS;
    XS(kstate,:) = x0*exp(YS);% Invert exponent to state.
end
fprintf('\nNP=%i;size(t)=[%i,%i];size(XS)=[%i,%i];size(XM)=[%i,
    ,NP,size(t),size(XS),size(XM));
XSM = mean(XS,1);
fprintf('\nNP=%i; size(XSM)=[%i,%i];',NP,size(XM));
% Begin Plot:
scrsize = get(0,'ScreenSize');
ss = 5.2; dss = 0.2; ssmin = 3.0;
nfig = nfig + 1;
stitle = {'Linear Mark-Jump-Diffusion Simulations' ...
        ,'Linear Diffusion Simulations' ...
        ,'Linear Mark-Jump Simulations'};
sylabel = {'X(t), Jump-Diffusion State','X(t) ...
        ,Diffusion State','X(t), Jump State'};
```

```
slegend = {'X(t), State 1  ','X(t), State 2  ' ...
        ,'X(t), State 3  ','X(t), State 4  ' ...
        ,'XM(t), th. Mean=E[X(t)]  '...
        ,'XSM(t), Mean of Samples '};
fprintf('\n\nFigure(%i):  Linear Jump-Diffusion Sims\n' ...
        ,nfig)
figure(nfig)
plot(t,XS(1,1:NP),'k+-' ...
    ,t,XS(2,1:NP),'k:' ...
    ,t,XS(3,1:NP),'k--' ...
    ,t,XS(4,1:NP),'k-.' ...
    ,t,XM(1:NP),'g-' ...
    ,t,XSM(1:NP),'b.-' ...
    ,'LineWidth',2); % Add for more States?
axis tight;
title(stitle(kjd),'FontWeight','Bold','Fontsize',32);
ylabel(sylabel(kjd),'FontWeight','Bold','Fontsize',32);
xlabel('t, Time','FontWeight','Bold','Fontsize',32);
```

```
hlegend=legend(slegend,'Location','NorthEast');
set(hlegend,'Fontsize',16,'FontWeight','Bold');
set(gca,'Fontsize',28,'FontWeight','Bold','linewidth',3);
ss = max(ss - dss,ssmin);
set(gcf,'Color','White','Position' ...
   ,[scrsize(3)/ss 60 scrsize(3)*0.60 scrsize(4)*0.80]);
%
% End JDSimulator Code
%
% linear Time-Dependent SDE Coefficient Functions:
% (Change with application; fns. must be vectorizable,
%  using vector element dot operations or vector fns.)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function M = mu(t)
% drift coefficient example, change with applications:
M = 0.1*sin(t);
% end mu(t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function S = sigma(t)
% drift coefficient example, change with applications:
S = 1.5*exp(-0.01*t);
% end sigma(t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function L = lambda(t)
% drift coefficient example, change with applications:
L = 3.0*exp(-t.*t);
% end lambda(t)
% End Variable Coefficients Subfunction Code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% End function linmarkjumpdiff09fig1.m
```