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 1, Corrected Post-Lecture October 9, 2009

7:30-9:30 pm, 28* September 2009, Kent 120 in Chicago

8:30-10:30 pm, 28 September 2009 at UBS Stamford

8:30-10:30 am, 29 September 2009 at Spring in Singapore

*** {Monday 28 September Yom Kippur is an official U. Chicago holiday, but since we would miss a whole week of classes, we will have the usual evening class in Chicago starting at 7:30pm. The religious holiday ends 42 minutes after sunset, so there is an overlap only in Chicago. Individuals, of course, are free to follow their conscience.**

Sorry, for any inconvenience.}

0.01 Histogram of S&P500 Log-Returns 1988-2008:

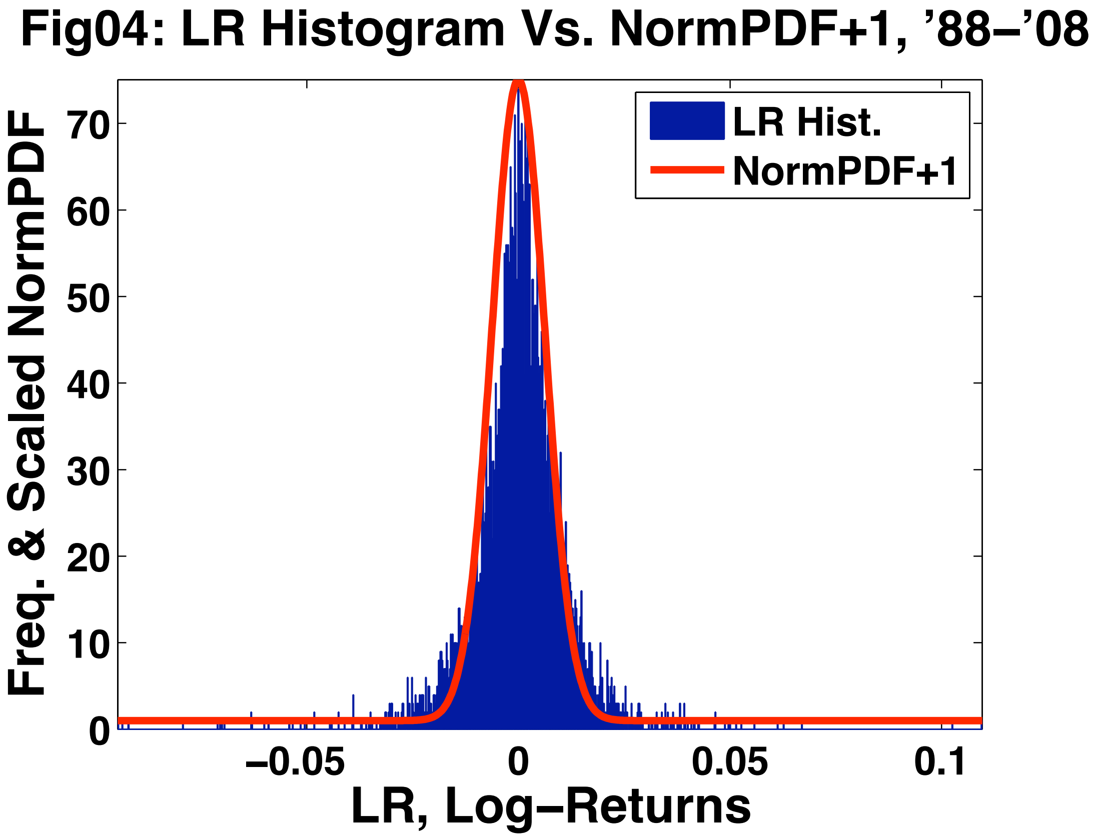
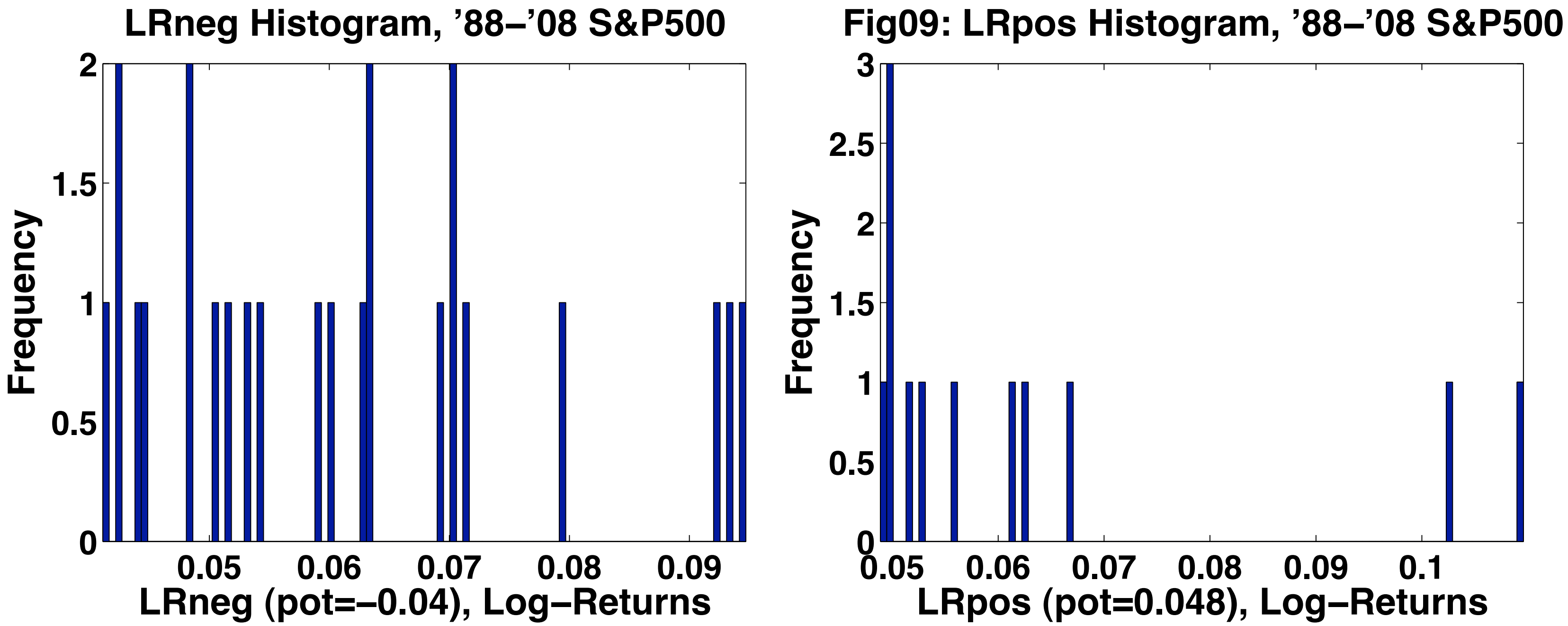


Figure 0.01: S&P500 Daily Log-Return Adjusted Closings from 1988 to 2008 (post-1987) showing long-tails of rare events. Normal kernel-smoothed graph, in **red**, plus one which accounts for non-central and ***normally invisible, but financially important, rare jumps***.

0.02 Extreme Negative Tail Events for Log-Returns ('88-'08):



(a) Extreme Negative Tails.

(b) Extreme Positive Tails.

Figure 0.02: Extreme Negative and Positive Log-Return Tail Events, with Thresholds POT $= -0.04$ and $+0.048$, respectively. POT means Peaks Over (or Under) Threshold. These represent the ***significant crashes or bonanzas*** during the time period. {Note: vertical scale differences.}

Course Outline (tentative)

1. **Introduction to Stochastic Diffusion and Jump Processes**: Basic properties of Poisson and Wiener stochastic processes. Based on the calculus model, differential and incremental models are discussed. The continuous Wiener processes model the background or central part of of financial distributions, while the Poisson jump process models the extreme, long tail behavior of crashes and bubbles of financial distributions.
2. **Stochastic Integration for Stochastic Differential Equations**: While the stochastic differentials and increments are useful in developing stochastic models and numerically simulating solutions, stochastic integration is important for getting explicit solutions or more manageable forms.
3. **Elementary Stochastic Differential Equations (SDEs)**: The stochastic chain rules for jump-diffusions with simple Poisson jump processes, starting from diffusion chain rules to jump chain rules to jump-diffusion chain rules.

4. **Stochastic Differential Equations for General Jump-Diffusions**: Stochastic differential equations with compound Poisson processes, i.e., including randomly distributed jump-amplitudes, state-time dependent coefficients, multi-dimensional SDEs, Martingales and finite rate Levy jump-diffusion formulations.
5. **Applications to Financial Engineering**: Generalized Black-Scholes-Merton option pricing analysis, option pricing for jump-diffusions and stochastic volatility, using risk-neutral measures; also the important event, Greenspan process. Of course, financial models and motivations will be used throughout the course.
6. **Time Series Introduction and the relationship to SDE models**: Time series models such as the discrete AR (autoregressive), MA (moving average), ARMA (combined), and ARCH (conditional "volatility") models, as time allows.

Comments:

- *This will be a more applied course than in the past, starting from stochastic differentials and stochastic integrals, as in the regular calculus, except with basic probabilities, then building up to stochastic differential equations and their solutions, eventually leading to financial applications and some useful abstract notions in stochastic calculus.*
- *Knowledge of basic probability is assumed, but you can review background preliminaries from online sources given below.*
- *For running, current Extended Syllabus for Finm 345, see*

https://chalk.uchicago.edu/ *

OR see

http://www.math.uchicago.edu/ hanson/finm345a09.html *

* *PDF Pine Green fonts mean Click to GO, Active URL Links.*

Course Business

- **Professor**: Floyd B. Hanson
  - Email: fhanson@uchicago.edu;
  - Office Hour: Mondays, 5PM (tentative), Eckhart Lounge/Lab, except in Singapore weeks 3-4.
  - Webpages:
    - *F. B. Hanson Short Homepage*, Financial Mathematics Program, University of Chicago.
    - *F. B. Hanson Long Home Page*, Professor Emeritus, Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago.
- **TAs in Chicago, Singapore & Stamford**: TBA
  - Email: TBA
  - Office Hours: TBA.
  - Review Sessions: TBA.

Texts:

- **Primary Text**: Class explanatory ***FINM 345 Lecture Notes***.
- **Optionally and Highly Recommended**: Floyd B. Hanson, ***Applied Stochastic Processes and Control for Jump-Diffusions: Modeling, Analysis, and Computation***, SIAM Books, October 2007. (Comments: *There is a 30% discount that a registered student can get at* ***Text and Order Page*** *with Coupon Code BKUC09, special for this class. Amazon and other book sellers charge list price.*) Some online material is freely available:
  - ○ ***Sample Chapter (5) Stochastic Calculus for Compound Poisson Jump-Diffusions***;
  - ○ ***Online Appendix B Preliminaries in Probability and Analysis***.
  - ○ ***Online Appendix C: MATLAB Code Listings,*** 46 pages.
  - ○ ***MATLAB Source Codes Table of Contents***.
  - ○ ***MATLAB Source Codes Directory***, 27 files plus directory zipped.
  - ○ ***Post Publication Errata***. (*Please send additional errors.*)

- **Supplemental Text**: Ruey S. Tsay, **Analysis of Time Series**, Wiley, August 2005. (Comments: *Text on time series by U. Chicago business professor; we will have a short introduction of time series in the context of stochastic calculus, but topic is moved from FINM 331 Winter 2009 and will not be in FINM 331 Winter 2010.*)
- **Optional Text**: Steven E. Shreve, **Stochastic Calculus for Finance II: Continuous-Time Models**, Springer Finance, April 2008. (Comment: *This is the Carnegie Mellon Computational Finance course, but is more abstract and much less applied, primarily about diffusions, getting to jumps much later in the book; however, this book is often used in the Financial Mathematics courses here.*)

- **Text for Recommended Computational System — MATLAB**: Desmond J. Higham and Nicolas J. Higham, ***MATLAB Guide***, SIAM Books, 2nd Edition, Order Code OT92, 2005. (Comments: *There is a 30% discount with SIAM student membership, but you can get a* ***complimentary membership*** *if sponsored by a SIAM member. This is probably the best mathematical MATLAB book. Also,* ***R***, ***S***, ***Excel***, ***Maple*** *and* ***Mathematica*** *are acceptable for assignments, but you are on your own.*)

Grading:

- **Homework**:
  - ○ There will be about 8-10 graded homework sets or course projects;
  - ○ You may consult with other student about the ideas involved;
  - ○ Submitted homework must be the individual student's own work;
  - ○ Similar solutions will receive discounted grades with divided credit;
  - ○ Codes and/or worksheets need to be submitted with computational solutions.
- **Exams**: There will be at least one, the final exam, likely take-home.
- **Final Grade**: The grade will be based upon an average of homework and final exam scores, weighted to reflect the number of points involved, i.e., homework will substantially count.

Prerequisite Knowledge:

1. **Introductory Probability**:
   - F. B. Hanson, ***Online Appendix B Preliminaries in Probability and Analysis***,
   - Niels O. Nygaard, ***Introduction to Stochastic Processes***, a concise review of background measure theory, probability theory and stochastic processes, but more abstract than in this course.
2. **Very Basic MATLAB:**
   - The (***MATLAB Student Version***); comes with the Statistics and other toolboxes.
   - MATLAB will be introduced in the course as examples and demonstration codes will be given in the lectures as well as posted online. You should rely heavily on MATLAB Help Windows.
   - See also Hanson's Online MATLAB Programs mentioned above.
   - See also Professor Nygaard's review sessions on various topics for examples, in particular reviews on statistics.

Some Related Resources of the Professor, plus prior FINM 345:

1. **Prior course**: *Math 586 Computational Finance*, Computational Finance, UIC, Spring 2008.
2. **Another Prior course**: *Math 574 Applied Optimal Control: Jump-Diffusion Stochastic Processes*, UIC, Fall 2006. (Comment: *First part of course was on stochastic processes and his book was written for this course and several related courses.*)
   - *Online Appendix C: MATLAB Programs* (listings of sample codes used to make book figures);
   - *MATLAB Source Codes Directory*, source m-files as individual files or zip-file of all m-files.
3. *Quantitative Finance References and Related References*, annotated books and links in finance and related topics.
4. *Autumn 2008 FINM 345 Stochastic Calculus*, Professor Per Mykland.

END of Course Extended Syllabus Review and Begin Course ⟶

FINM 345 Stochastic Calculus:

1. Introduction to Stochastic Diffusion and Jump Processes:

1.1 Stochastic (Random) Nature of Financial Data:

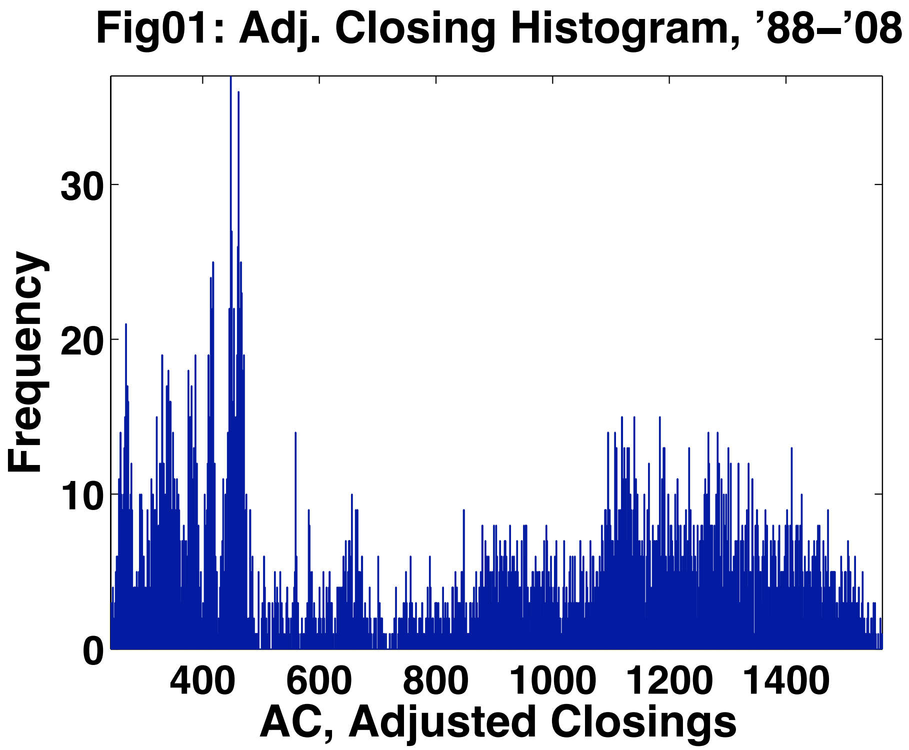


Figure 3: **S&500** Index **Daily Adjusted Closings AC(t)** for **t=1:NAC**, from 1988 to 2008 (post-1987) showing scattered behavior of the price without any recognizable probability distribution seen. (**NAC=AC-count**.)

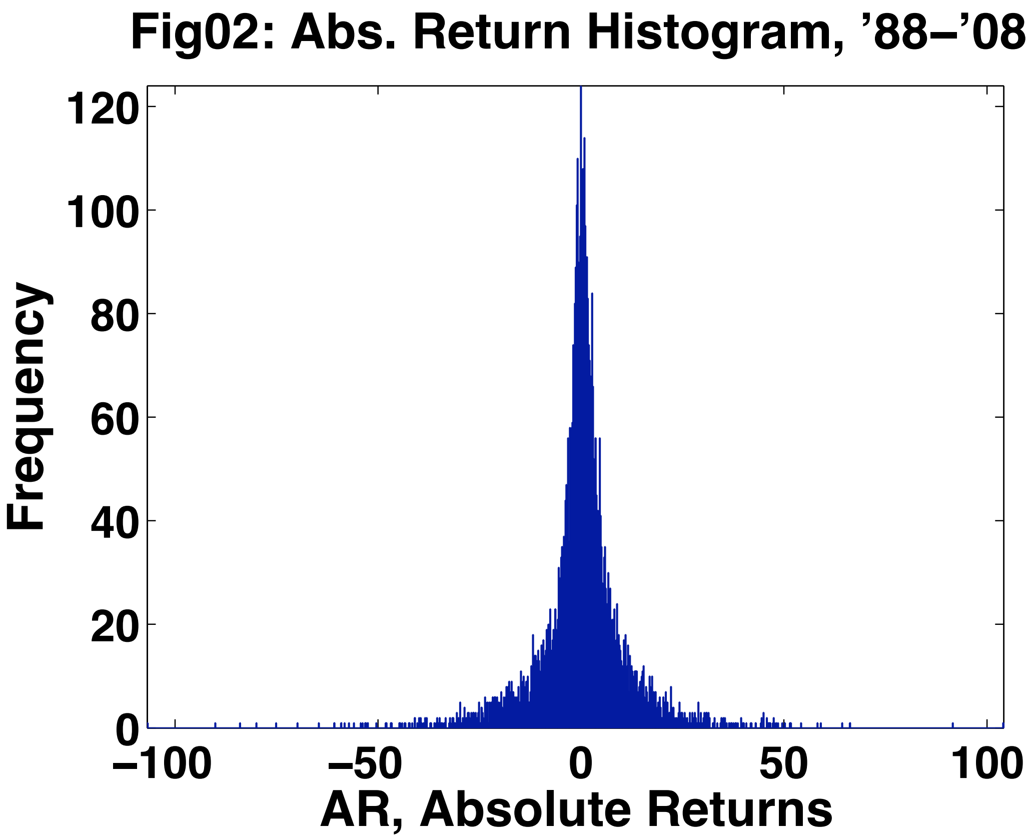


Figure 4: **S&500** Index Daily **Absolute Returns or Differences AR(t) = AC(t+1)-AC(t)** $\equiv$ **ΔAC(t)** for **t=1:NAC-1**, from 1988 to 2008 (post-1987) showing **more organized behavior**, resembling a very narrow normal distributions with many discrete deviations from the normal.

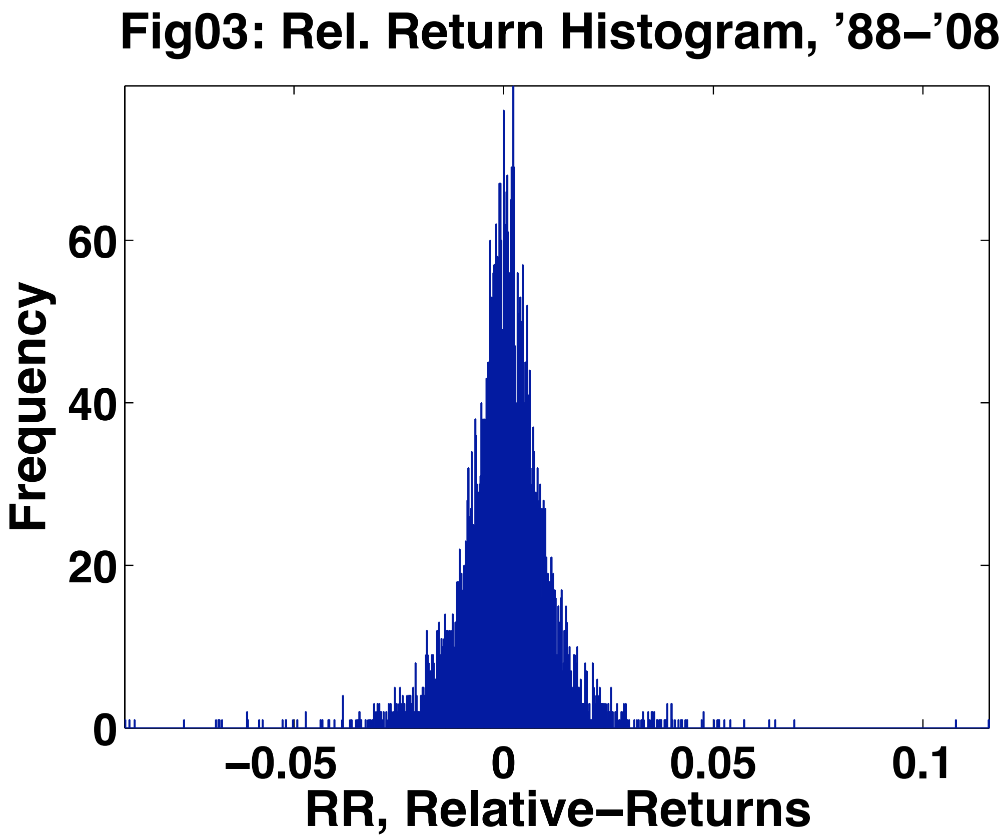


Figure 5: **S&500** Index Daily **Relative Returns** **RR(t) = AC(t+1)/AC(t)-1** $\equiv$ **ΔAC(t)/AC(t)** for **t=1:NAC-1**, from 1988 to 2008 (post-1987) showing a more developed normal distribution with wider spread due to **reduction of the scale** of the returns and many rare tail events.

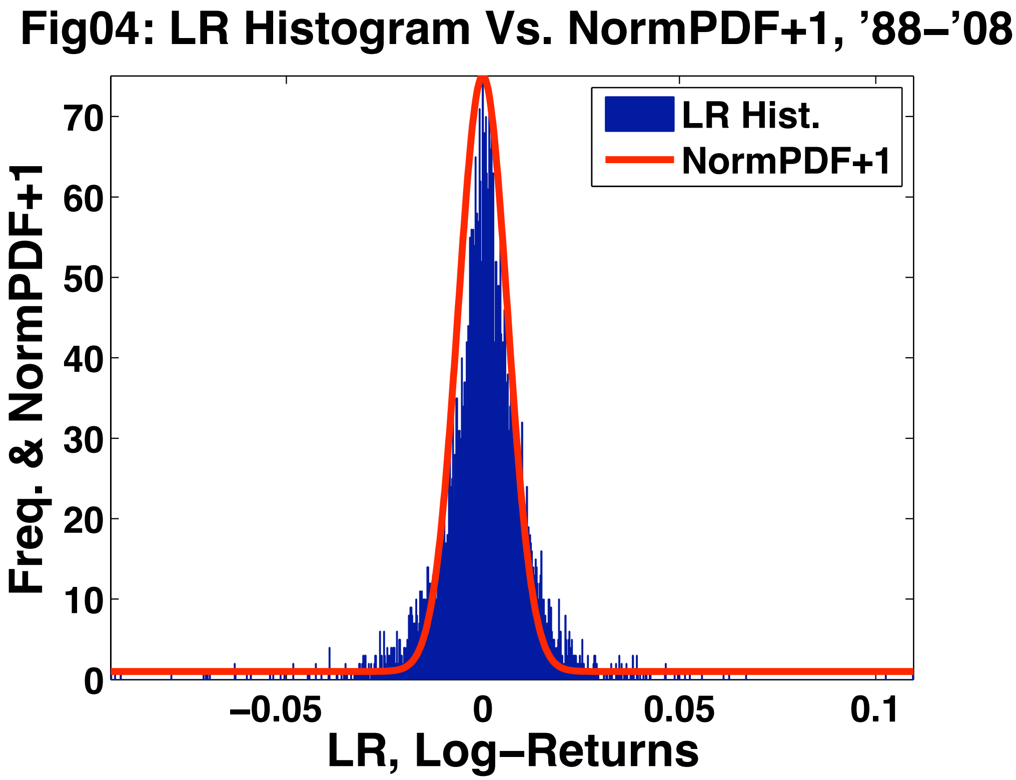


Figure 6: **S&500** Index Daily **Log Returns** **LR(t) = log(AC(t+1))-log(AC(t)) = log(1+RR(t))** $\sim$ **RR(t)** for **t=1:NAC-1** & **RR(t)** $\ll$ **1**, from 1988 to 2008 (post-1987) showing wide spread and tail event behavior similar to **RR(t)**. In **red**, an approximate normal density is overlaid with an added unit to account for fat tails from jump of crashes and bonanzas. **So, the infinite normal tails have little probability compared to jumps.**

1.2. General Markov Processes in Continuous Time: Background for a Toolbox of a Mixture of Central–Normal and Tail–Jump Returns:

- **Definition 1.1:** *A **process** X is simply a function of time t (in this class), $X = X(t)$.*
- **Definition 1.2:** *A **deterministic process** $X(t)$ is a process without any random component, it does not involve chance or it has <u>one reality</u>, so that its average or expectation is the same as the process for all time, i.e., $\mathrm{E}[X(t)] = X(t), \forall t$.*

- **Definition 1.3:** *A* ***stochastic process***, ***X(t)***, *is a process with random components, i.e., a random variable that is a function of time.*[1]
- **Definition 1.4:** *A* ***Markov process***, $\boldsymbol{X(t)}$, *is a stochastic process such that the conditional probability satisfies*

$$\begin{aligned}\mathrm{Prob}[\boldsymbol{X(t+\Delta t) = x | X(s), 0 \le s \le t}]\\ = \mathrm{Prob}[\boldsymbol{X(t+\Delta t) = x | X(t)}],\end{aligned}$$

for any $\boldsymbol{t \geq 0}$ *and* $\boldsymbol{\Delta t \geq 0}$, *and* $\boldsymbol{x}$ *is in the state space* $\boldsymbol{\mathcal{D}_x}$.
{Comment: That is, the change of a Markov process depends on the current time and not on the past.}

[1] Given a probability space $\{\Omega, \mathcal{F}, \mathcal{P}\}$ of sample space Ω, a σ-algebra $\mathcal{F}$ of subsets of Ω and proper probability measure $\mathcal{P}$ on Ω. See Nygaard, ***Introduction to Stochastic Processes***. (This *boilerplate* is obligatory for abstract probability foundations, but it is mentioned here once and will not be mentioned again since it will not be needed in these applied lectures.)

- **Definition 1.5:** *The stochastic process* ***X(t)*** *is a* ***stationary process*** *if the distribution of the increment process* $\Delta X(t) \equiv X(t+\Delta t) - X(t)$ *depends only on the time-step* Δt *and is independent of the current time* t.

  {For example, the distribution for a stationary $X(t)$ can be written

$$\begin{aligned}\mathrm{Prob}[\Delta X(t) \le x] &= f(x;\Delta t)\\ &= \mathrm{Prob}[\Delta X(s) \le x]\\ &= \mathrm{Prob}[\Delta X(0) \le x],\\ &= \mathrm{Prob}[X(\Delta t) \le x],\end{aligned}$$

  for some function f, assuming $t > 0$, $s > 0$ & $X(0) = 0$.}

1.3. Properties of Standard Wiener Process $W(t)$ (Brownian Motion or Diffusion) for Central–Normal Returns:

- Initially, **W(0) = 0** with probability one.
- **W(t)** is a **continuous process**, i.e.,
  $W(t^+) = W(t) = W(t^-)$.
- **W(t)** has **independent increments**, i.e., the increments
  $\Delta W(t_i) \equiv W(t_i + \Delta t_i) - W(t_i) = W(t_{i+1}) - W(t_i)$,
  are mutually independent for all t_i with nonoverlapping time-intervals (excluding pointwise overlap); for example, if the increments $\Delta W(t_i)$ and $\Delta W(t_j)$ are nonoverlapping, then the joint probability
  $\mathrm{Prob}[\Delta W(t_i) \le w_i, \Delta W(t_j) \le w_j] =$
  $\mathrm{Prob}[\Delta W(t_i) \le w_i] \cdot \mathrm{Prob}[\Delta W(t_j) \le w_j]$;

for continuous time processes pointwise is permissible since points of zero measure due not count in probability integrals; however, it is usually assumed the the associated time-intervals are open on the right, $[t_i, t_{i+1})$ and $[t_j, t_{j+1})$, with $t_{i+1} \le t_j$ or $t_{j+1} \le t_i$, corresponding to discrete jump processes.

- The distribution of $\Delta W(t) = W(t + \Delta t) - W(t)$ by definition depends only the increment Δt, but is independent of the current time **t**, so **W(t)** is a **stationary (increment) process**. {*Caution: invalid for variable coefficients.*}
- The process **W(t)** is a **Markov process** by definition, so

$$\begin{aligned}&\mathrm{Prob}[W(t + \Delta t) = w \mid W(s), 0 \le s \le t]\\&\qquad = \mathrm{Prob}[W(t + \Delta t) = w \mid W(t)].\end{aligned}$$

- The $W(t)$ is **normally distributed** with mean $\mu = 0$ and variance $\sigma^2 = t$, i.e., the density of $W(t)$ is
$$\phi_{W(t)}(w) = \phi_n(w; 0, t) = \frac{1}{\sqrt{2\pi t}} \exp\left(-\frac{w^2}{2t}\right),$$
when $-\infty < w < +\infty$ and $t > 0$. The actual distribution function for $W(t)$ is denoted by $\Phi_{W(t)}(w) = \Phi_n(w; 0, t) = \int_{-\infty}^{w} \phi_n(v; 0, t) dv$. Summarizing basic statistics, $\mathrm{E}[W(t)] = 0$ and $\mathrm{Var}[W(t)] = t$. (The general notation $\phi_n(w; \mu, \sigma^2)$ means a normal distribution with mean μ and variance σ^2, see Online Appendix B, Eq. (B.22). Also, when $t = 0^+$ then $\phi_{W(0^+)}(w) = \delta(w)$, where $\delta(w)$ is **Dirac's delta function**, a generalized distribution function with mass concentrated at $w = 0$.)

- Substituting the time-step Δt for t and the corresponding **Wiener increment process** $\Delta W(t) = W(t+\Delta t) - W(t)$ for $W(t)$, the density is

$$\phi_{\Delta W(t)}(w) = \phi_n(w; 0, \Delta t) = \frac{1}{\sqrt{2\pi\Delta t}} \exp\left(-\frac{w^2}{2\Delta t}\right),$$

such that $\mathrm{E}[\Delta W(t)] = 0$ and $\mathrm{Var}[\Delta W(t)] = \Delta t$.

- Since the infinitesimal $dt = (t + dt) - t$ is also an increment, then the Wiener process scales down to the **Wiener differential process** $dW(t) = W(t+dt) - W(t)$ with density

$$\phi_{dW(t)}(w) = \phi_n(w; 0, dt) = \frac{1}{\sqrt{2\pi dt}} \exp\left(-\frac{w^2}{2dt}\right),$$

such that $\mathrm{E}[dW(t)] = 0$ and $\mathrm{Var}[dW(t)] = dt$.

- The **relationship to the standard normal**, i.e., the zero-mean and unit-variance normal, follows from a change of variables,

$$\begin{aligned}\Phi_{W(t)}(w) &= \int_{-\infty}^{w} \exp(-v^2/(2t))dv/\sqrt{2\pi t} \\ &= \int_{-\infty}^{w/\sqrt{t}} \exp(-y^2/2)dy/\sqrt{2\pi} = \Phi_n(x; 0, 1),\end{aligned}$$

  where $x = w/\sqrt{t}$ is the standard normal variate transformation, so $w = \sqrt{t} \cdot x$. (Also see Theorem 1.12 in Hanson's (2007) text for a more detailed statement and proof.)

- The **Wiener increment process and differential process are stationary, Markov processes**, since their distributions depend only on Δt or dt, respectively, but are independent of the current time t.

- **Theorem 1.1. W(t) is Nondifferentiable:** For any fixed $x > 0$ and $t > 0$,

$$\text{Prob}\left[\lim_{\Delta t \to 0^+}\left[\left|\frac{\Delta W(t)}{\Delta t}\right| > x\right]\right] = 1.$$

  For a proof, see Hanson's book, p. 9.

- **Theorem 1.2. Covariance of W(t)**. *If* ***W(t) is a Wiener process****, then* $\text{Cov}[W(t), W(s)] = \min[t, s]$.

  For a proof using overlap, see Hanson's book, p. 4.

- **Corollary 1.1.** *If* $\{t_i, i = 0 : N\}$ *is a time mesh with* N *steps* $\{\Delta t_i = t_i - t_{i-1}, i = 1 : N\}$ *on* ***[0,T]****, then*

$$\text{Cov}[\Delta W(t_i), \Delta W(t_j)] = \text{Var}[\Delta W(t_i)]\delta_{i,j} = \Delta t_i\,\delta_{i,j}$$

  for the increment process, where $\delta_{i,j}$ *is the* ***discrete Kronecker delta***. (Note that there is no overlap in time except at a common endpoint.)

- **Corollary 1.2.** *If t & s are positive, then*

$$\mathrm{Cov}[dW(t), dW(s)] = \mathrm{Var}[dW(t)]\delta(s-t) = dt\,\delta(s-t)$$

for the differential process, where again $\delta(s-t)$ is the continuous Dirac delta function. (Note that only the pointwise overlap counts for infinitesimals.)

- **Wiener Increment Process Moments**:

  1. First, the odd powers: $\mathrm{E}[(\Delta W(t))^{2k+1}] = 0$ when $k = 0, 1, 2, \ldots$ by integrand oddness on a symmetric interval, $(-\infty, +\infty)$.

2. Second, the even powers:

$$\begin{aligned}\mathrm{E}\left[(\Delta W(t))^{2k}\right] &= \int_{-\infty}^{+\infty}\phi_n(w;0,\Delta t)w^{2k}dw \\ &= \frac{2}{\sqrt{2\pi\Delta t}}\int_0^{+\infty}\exp\left(-\frac{w^2}{2\Delta t}\right)w^{2k}dw \\ &= \frac{(2\Delta t)^k}{\sqrt{\pi}}\int_0^{+\infty}\exp\left(-u\right)u^{k-1/2}du \\ &= \frac{(2\Delta t)^k\Gamma(k+1/2)}{\Gamma(1/2)}\end{aligned}$$

for $k = 0, 1, 2, \ldots$, where Γ is the **gamma function** defined by $\Gamma(x) \equiv \int_0^{\infty} e^{-u}u^{x-1}du, x > 0$, with initial condition $\Gamma(1) \equiv 1$ and special value $\Gamma(1/2) = \sqrt{\pi}$ and recursive form $\Gamma(x+1) = x\Gamma(x)$, so that $\Gamma(x+1) = x!$ is the usual factorial function.

3. Lastly, for odd powers of the absolute value,

$\mathrm{E}[|\Delta W(t)|^{2k+1}] = (2\Delta t)^{(2k+1)/2}\Gamma(k+1)/\Gamma(1/2)$,

for $k = 0, 1, 2, \ldots$.

Table 1.1. Expected moments of absolute value of Wiener increments:

m	$\mathrm{E}[\|\Delta W(t)\|^m]$
0	1
1	$\sqrt{2\Delta t/\pi}$
2	Δt
3	$2\Delta t\sqrt{2\Delta t/\pi}$
4	$3(\Delta t)^2$
5	$8(\Delta t)^2\sqrt{2\Delta t/\pi}$
6	$15(\Delta t)^3$
$\vdots$	$\vdots$
2k	$(2k-1)!!(\Delta t)^k$
2k+1	$k!(2\Delta t)^k\sqrt{2\Delta t/\pi}$

Here, the double factorial function is

$$\mathbf{(2\boldsymbol{k}-1)!! = (2\boldsymbol{k}-1)\cdot(2\boldsymbol{k}-3)\ldots 1}.$$

1.4. MATLAB Simulation of Wiener Processes:

- **MATLAB's standard normal distribution (pseudo-)random number generator** is **`randn`**, such each call to **`randn`** produces one "independent" normal variate for each call, **`randn(n,1)`** produces a column-vector of **`n`** rows, **`randn(1,n)`** produces a row-vector of **`n`** columns that is the same size as the construct **`1:n`**, **`randn(m,n)`** produces an $m \times n$ matrix, but **`randn(n)`** produces an $n \times n$ matrix like **`randn(n,n)`** while higher dimensional arrays are available.

- Given **even time-steps** $\Delta t = (T-0)/(n-0)$, with $t_i = 0 + i\Delta t$ for $i = 0:n$, $t_0 = 0$, $t_n = T$ and $\Delta W(t_i) = W(t_{i+1}) - W(t_i)$ for $i = 0:n-1$, then $W(t_j) = W(0) + \sum_{i=0}^{j-1} \Delta W(t_i)$ for $j = 1:n$ with **W(0)=0**.
- Following the previous transformation of the Wiener increment distribution to MATLAB's standard normal implies `DW=sqrt(DT)*randn;` yields one increment, while `DW=sqrt(DT)*randn(n,1) ;` yields all **n** increments for the mesh on $[0, T]$ and `W=zeros(n+1,1)`;`W(2:n+1,1)=cumsum(DW);` yields all **n+1** Wiener process values, including the initial **W(0) = 0**. {***Caution: MATLAB is unit subscript based, so only positive subscripts are legal.***}

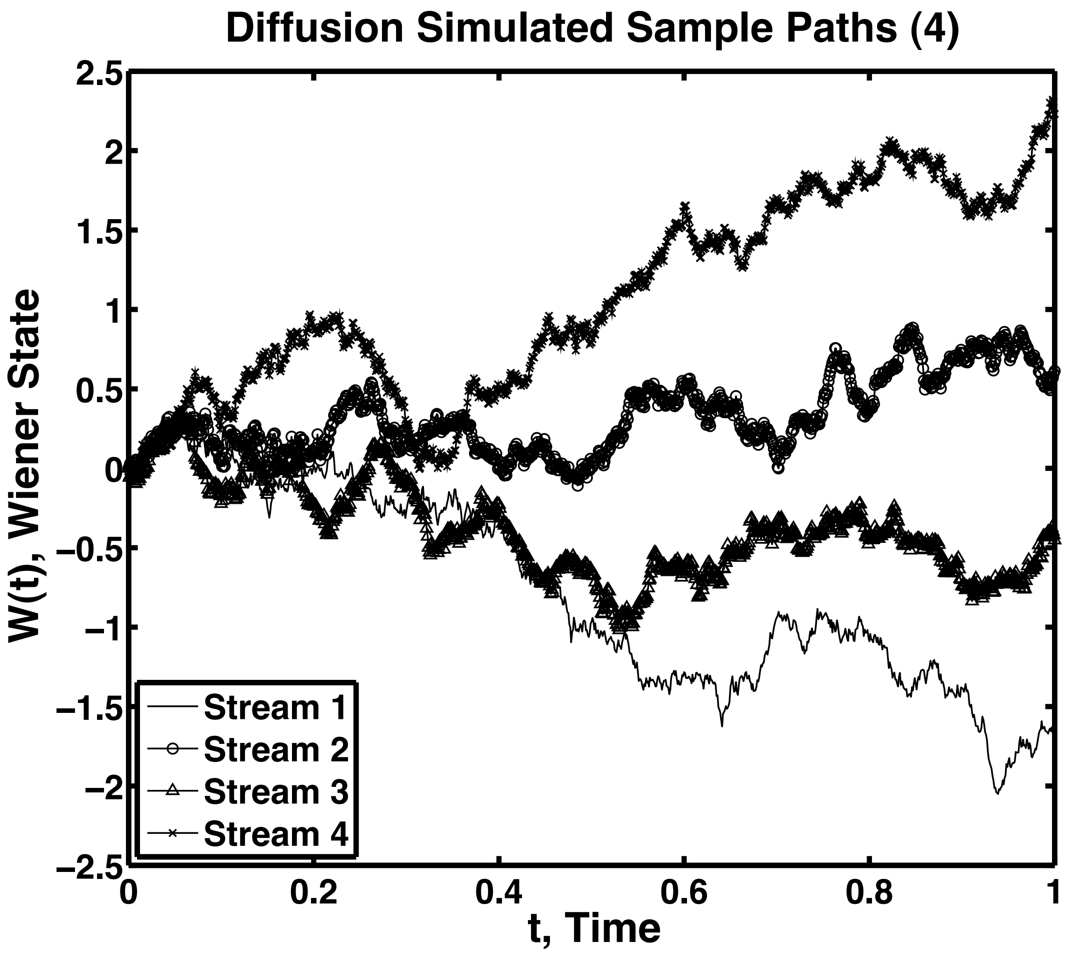


Figure 7: Wiener or Diffusion sample paths for four (4) random states or streams using MATLAB. (See also *sample Wiener trajectory code* in Hanson's (2007) Applied stochastics text, page 6. **Corrected 10/06/09.**)

Wiener Sample Paths MATLAB code example (edited for space):

```
function wiener09fig1
% Fig. 1.1a Book Illustrations for Wiener/Diffusion;
%   RNG Simulation for t in [0,1] with sample variation.
%   Generation is by summing Wiener increments DW;
clc % clear workspace of prior output.
clear % clear variables, but must come before globals;
fprintf(’\nfunction wiener09fig1 OutPut:’);% figure name
nfig = 0;
n = 1000; T = 1.0; dt = T/n; % Set initial time grid.
np = n + 1; % Number of points.
sqrtdt = sqrt(dt);  % Set std. Wiener increment moments;
% for dX(t) = mu*dt + sigma*dW(t); here mu=0, sigma=1,
%     and scaled dW(t) = sqrt(dt)*randn;
t = 0:dt:T; % time row-vector
nstate = 4; % number of states
[s1,s2,s3,s4] = RandStream.create(’mrg32k3a’ ...
,’NumStreams’,nstate);
```

```
DW = zeros(nstate,n); W = zeros(nstate,np); % arrays;
% Also sets initial W(j,1) = 0;
for j = 1:nstate
    if j==1,s=s1;elseif j==2,s=s2;elseif j==3,s=s3;
       else s=s4;end
    DW = sqrtdt*randn(s,[1,n]); % n-sample row-vector;
    W(j,2:np) = cumsum(DW); % Includes W(0)=0 & Vector;
end
fprintf(’\nsize(DW)=[%i,%i]; size(W)=[%i,%i];’ ...
,size(DW),size(W));
nfig = nfig + 1;
scrsize = get(0,’ScreenSize’); % figures screensize;
ss = [5.0,4.0,3.5]; % For ease in finding figures;
fprintf(’\n\nFigure(%i): Diffusion Sample Path(4)\n’ ...
,nfig)
figure(nfig)
marks = {’k-’,’k-o’,’k-^’,’k-x’}; % change marks;
%
```

```
for j = 1:nstate
    plot(t,W(j,1:np),marks{j},’linewidth’,1); hold on;
end
hold off
title(’Diffusion Simulated Sample Paths (4)’...
    ,’FontWeight’,’Bold’,’Fontsize’,24);
ylabel(’W(t), Wiener State’...
    ,’FontWeight’,’Bold’,’Fontsize’,24);
xlabel(’t, Time’...
    ,’FontWeight’,’Bold’,’Fontsize’,24);
hlegend=legend(’Stream 1’,’Stream 2’,’Stream 3’ ...
,’Stream 4’,’Location’,’SouthWest’);
set(hlegend,’Fontsize’,20,’FontWeight’,’Bold’);
set(gca,’Fontsize’,20,’FontWeight’,’Bold’,’linewidth’,3);
set(gcf,’Color’,’White’,’Position’ ...
   ,[scrsize(3)/ss(nfig) 60 scrsize(3)*0.6 scrsize(4)*0.8]);
% End wiener09fig1 Code
```

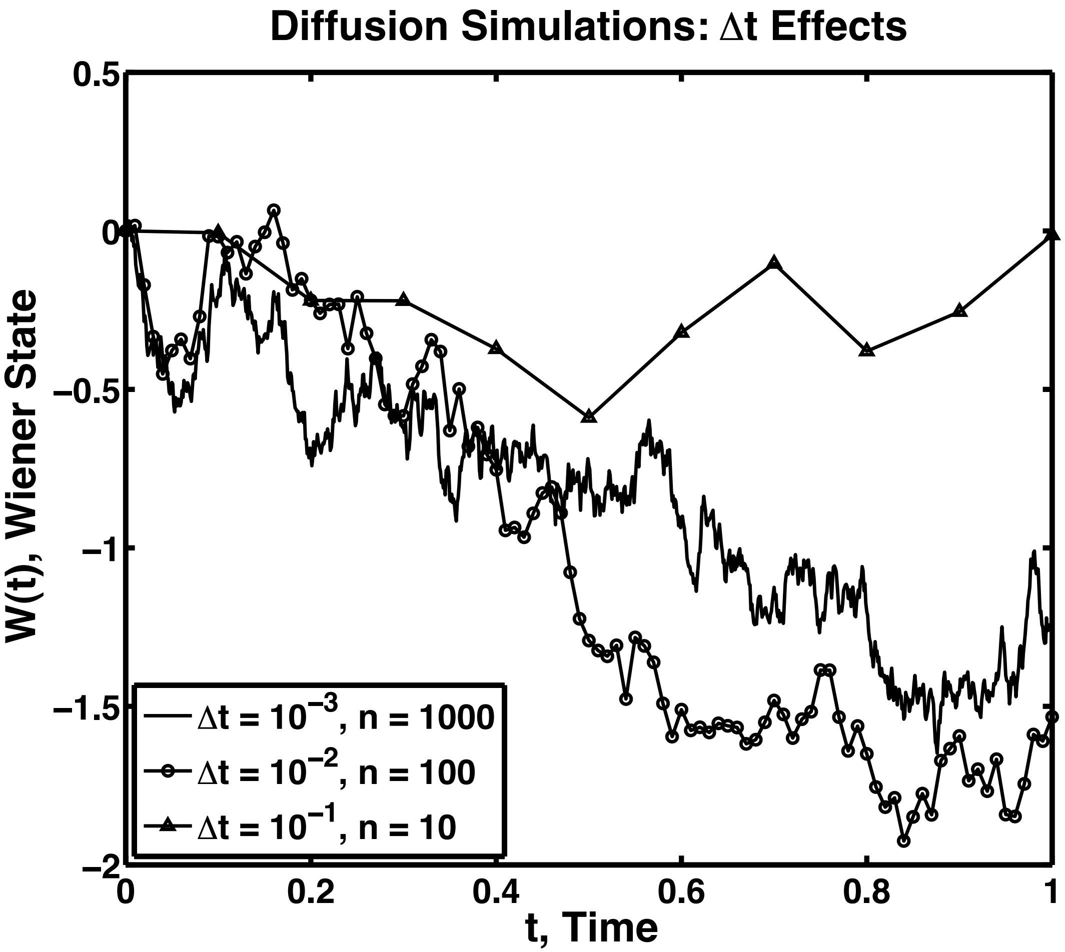


Figure 8: Wiener sample paths for four (4) different time-steps using MATLAB. (See also *sample Wiener trajectory code* in Hanson's (2007) Applied stochastics text, page 6.)

Wiener Sample Paths for Different Time-Steps MATLAB code example:

```
function wiener09fig2
% Fig. 1.1b Book Illustration for Wiener/Diffusion;
%   Generation is by summing Wiener increments;
clc % clear workspace of prior output.
clf % clear figures, else accumulative.
clear % clear variables, but must come before globals;
fprintf(’\nfunction wiener09fig2 OutPut:’);%figure name
nfig = 1;
n = 1000; T = 1.0; dt = T/n; % Several dt’s.
np = n+1; % Total number of Points.
% for dX(t) = mu*dt + sigma*dW(t); here mu=0, sigma=1
%     and scaled dW(t) = sqrt(dt)*randn
ndt = 3; % number of local dt’s.
randn(’state’,1);   % Set state for repeatability;
% Ignore MATLAB mlint "deprecated" warning;
Rn = randn(1,n); % common random sample of n points.
W = zeros(ndt,np); % W array of local vectors;
```

```
% Also sets all W(kdt,1) = 0 for t(1) = 0; unit based.
ts = zeros(ndt,np); % Declare maximal local time vectors;
%%%%% Begin Plot:
nfig = nfig + 1;
scrsize = get(0,’ScreenSize’); % figures screensize;
ss = [5.0,4.0,3.5]; % For ease in finding figures;
fprintf(’\n\nFigure(%i): Diffusion Sample Paths(4)\n’...
    ,nfig)
figure(nfig)
marks = {’k-’,’k-o’,’k-^’,’k-x’}; % change marks;
for kdt = 1:ndt % Test different dt’s:
    sc = 10^(kdt-1); % dt scalar factor;
    ns = n/sc; nps = ns+1; % Local counts;
    dts = sc*dt; % Local time steps;
    sigs = sqrt(dts); % Local diffusion scaling;
    ts(kdt,1:nps) = 0:dts:T; % Local times; W=cumsum;
    W(kdt,2:nps) = sigs*cumsum(Rn(1,sc*(1:ns))); %vector;
    plot(ts(kdt,1:nps),W(kdt,1:nps),marks{kdt} ...
     ,'linewidth',2); hold on;
end
hold off
title('Diffusion Simulations: \Delta{t} Effects'...
    ,'FontWeight','Bold','Fontsize',24);
ylabel('W(t), Wiener State'...
    ,'FontWeight','Bold','Fontsize',24);
xlabel('t, Time'...
    ,'FontWeight','Bold','Fontsize',24);
hlegend=legend('\Delta{t =10^{-3},n=1000'...
    ,'\Delta{t}=10^{-2},n=100' ...
    ,'\Delta{t}=10^{-1},n=10','Location','SouthWest');
set(hlegend,'Fontsize',20,'FontWeight','Bold');
set(gca,'Fontsize',20,'FontWeight','Bold','linewidth',3);
set(gcf,'Color','White','Position' ...
   ,[scrsize(3)/ss(nfig) 60 scrsize(3)*0.6 scrsize(4)*0.8]);
% End wiener09fig2 Code
```

1.3. Properties of Simple Poisson Process P(t) for Rare, Fat Tail Returns (Also called Point Processes or Counting Processes):

- Initially, **P(0) = 0** with probability one.
- **P(t)** is a **piecewise-right-continuous process**, i.e., **P(t$^+$) = P(t) = P(t$^-$)**, except at Poisson Jump Times, $t = T_j$, when $P(T_j^+) = P(T_j) = P(T_j^-) + 1$, so there are instantaneous jumps (discontinuities) of unit magnitude; Poisson jumps are assumed to be sufficiently rare that only one jump can occur at any instant of time.
- The Poisson process **P(t)** has **independent increments**, i.e., the increments $\Delta P(t_i) \equiv P(t_i + \Delta t_i) - P(t_i) = P(t_{i+1}) - P(t_i)$, are mutually independent for all t_i with nonoverlapping time-intervals;

for example, if the increments $\Delta P(t_i)$ and $\Delta P(t_j)$ are nonoverlapping, then the joint probability $\mathrm{Prob}[\Delta P(t_i) \le p_i, \Delta P(t_j) \le p_j] = \mathrm{Prob}[\Delta P(t_i) \le p_i] \cdot \mathrm{Prob}[\Delta P(t_j) \le p_j]$; for right-continuous time processes it is usually assumed the the associated time-intervals are open on the right, but closed on the left for continuity from the right, $[t_i, t_{i+1})$ and $[t_j, t_{j+1})$, with $t_{i+1} \le t_j$ or $t_{j+1} \le t_i$.

- The distribution of $\Delta P(t) = P(t + \Delta t) - P(t)$ by definition depends only the increment Δt, but is independent of the current time **t**, so **P(t)** is a **stationary (increment) process**. $\{$***Caution: This applies to the constant jump rate λ case, so is invalid for variable coefficients.***$\}$

- The process **P(t)** is a **Markov process** by definition, so

$$\mathrm{Prob}[P(t+\Delta t)=p\,|\,P(s), 0\le s\le t]$$
$$=\mathrm{Prob}[P(t+\Delta t)=p\,|\,P(t)].$$

- The $P(t)$ is **Poisson distributed** with the mean $\mu_P = \mathrm{E}[P(t)] = \Lambda = \lambda t = \sigma^2_P = \mathrm{Var}[P(t)]$ equaling the variance, in the constant rate λ case , i.e., the distribution of $P(t)$ **with parameter Λ** is

$$\Phi_{P(t)}(k;\Lambda) = \mathrm{Prob}[P(t)=k] = p_k(\Lambda) = e^{-\Lambda}\frac{\Lambda^k}{k!},$$

for $k=0,1,2,\dots,$ constant $\Lambda > 0$ and $t>0$, with $p_k(0^+) = \delta_{k,0}$. (Note that $\Lambda^k/k!$ is the **k**th Taylor term in the expansion of $\exp(\Lambda)$, a fact useful in calculations.)

- Substituting the time-step Δt for **t** and the corresponding **Poisson increment process** $\Delta P(t) = P(t+\Delta t) - P(t)$ for **P(t)**, the distribution is

$$\Phi_{\Delta P(t)}(k;\lambda\Delta t) = \mathrm{Prob}[\Delta P(t) = k] = p_k(\lambda\Delta t) = e^{-\lambda\Delta t}\frac{(\lambda\Delta t)^k}{k!},$$

such that $\mathrm{E}[\Delta P(t)] = \lambda\Delta t$ and $\mathrm{Var}[\Delta P(t)] = \lambda\Delta t$, with parameter $\Delta\Lambda = \lambda\Delta t$.

- Since the infinitesimal $dt = (t + dt) - t$ is also an increment, then the Poisson process scales down to the **Poisson differential process** $dP(t) = P(t+dt) - P(t)$ with distribution

$$\Phi_{dP(t)}(k;\lambda dt) = \mathrm{Prob}[dP(t) = k] = p_k(\lambda dt) = e^{-\lambda dt}\frac{(\lambda dt)^k}{k!},$$

such that $\mathrm{E}[dP(t)] = \lambda dt$ and $\mathrm{Var}[dP(t)] = \lambda dt$, with parameter $d\Lambda = \lambda dt$.

- The **Poisson increment process and differential process are stationary, Markov processes**, since their distributions depend only on Δt or dt, respectively, but are independent of the current time **t**, in the constant jump rate case.
- **Theorem 1.3. Covariance of P(t)**. *If **P(t) is a Poisson process** with constant jump rate λ, then*
  $$\mathrm{Cov}[P(t), P(s)] = \lambda \min[t, s].$$
  For a proof using overlap, see Hanson's book, p. 16.
- **Corollary 1.3.** *If* $\{t_i, i = 0 : N\}$ *is a time mesh with* N *steps* $\{\Delta t_i = t_i - t_{i-1}, i = 1 : N\}$ *on **[0,T]**, then*
  $$\mathrm{Cov}[\Delta P(t_i), \Delta P(t_j)] = \mathrm{Var}[\Delta P(t_i)]\delta_{i,j} = \lambda \Delta t_i \, \delta_{i,j}$$
  for the increment process with constant λ*, where* $\delta_{i,j}$ *is the Kronecker delta.*

- **Corollary 1.4.** *If* t **&** s *are positive, then*

$$\mathrm{Cov}[dP(t), dP(s)] = \mathrm{Var}[dP(t)]\delta(s-t) = \lambda dt\,\delta(s-t)$$

for the differential process with constant λ, *where* $\delta(s-t)$ *is the Dirac delta function.*

(Comment: Note that only the pointwise overlap counts for infinitesimals. **The Dirac delta function only has meaning under and integral sign, e.g.,**

$$\int_{-\infty}^{\infty} f(s)\delta(s-t)ds = f(t)$$

for some nice function f.)

- **Lemma 1.1. Exponential Distribution of Time Between Jumps:** *Let $P(t)$ be a simple Poisson process, with fixed jump-frequency $\lambda > 0$, and let T_j denote the jth jump-time, then the distribution of the* ***interjump-time*** $\Delta T_j = T_{j+1} - T_j$ *for* $j = 0, 1, 2, \ldots$, *defining* $T_0 = 0$, *conditioned on* T_j, *is*

$$\Phi_{\Delta T_j}(\Delta t) = \mathrm{Prob}[\Delta T_j \le \Delta t] = 1 - e^{-\lambda \Delta t}.$$

(Comment: The basic idea of this proof is that the probability of the time between jumps $\Delta T_j = T_{j+1} - T_j$ less than Δt, conditioned on the prior jump-time T_j, will be the same as the probability that there is at least one jump in the time interval, which is the same as one minus the probability that there are no jumps in the time interval. See Hanson's (2007) book for the proof details.)

- **Two Poisson Probability Complementary Representations**:

  1. First, given a fixed average jump count per time step Δt,

  $$\Delta\Lambda(t) \equiv \int_t^{t+\Delta t} \lambda(s)ds \simeq \lambda(t)\Delta t$$

  (in the simple case, fixed $\lambda(t) = \lambda$ & Δt, **calling `dLambda`=** $\lambda(t)\Delta t$, we can simulate jump counts $N = [N_j]_{1\times n}$ = `poissrnd(dLambda,1,n)` using the Poisson distribution. In finance modeling, an example would be the simulations of jumps at each of **T** daily closings given some $\lambda > 0$ with $\Delta t = 1/252$ years. (**On average there are 252 daily market closing per year in the U.S.; rates per year are standard units.**) In this case, the jump-trajectory is $\{(t_j, P_j) : t_j = (j-1)\Delta t, P_j = \sum_{i=1}^{j-1} N_j; j = 1 : n+1; P_1 = 0\}$, where N_j **is the number of jumps per day**.

2. Second, given a fixed jump-rate $\lambda > 0$, samples of the time intervals for the next jump $DT = [DT_j]_{1\times m}$ can be generated from the exponential distribution in a **Lemma 1.1** and thus general the full Poisson trajectory $\{(T_j, P_j) : (T_1, P_1) = (0, 0); (T_j, P_j) = (\sum_{i=1}^{j/2} DT_i, j/2–1)$ for $j = 2 : 2 : 2m; (T_j, P_j) = (T_{j-1}, (j–1)/2)$ for $j = 3 : 2 : 2m + 1\}$ from the whole sample DT.

   (Note that it takes one more than twice the time-steps to include the dual pre- and post-jump values at each jump-time; it is much easier in MATLAB vector code with `DT=exprnd(1/lambda,1,m)`, **noting that the mean of the exponential distribution is** $\mu_e = 1/\lambda = \mathrm{E}[\Delta T_j]$.)

Comment: The Poisson process distribution and Poisson Inter-Jump distribution equivalent representations are illustrated in the following two qualitatively similar graphs.

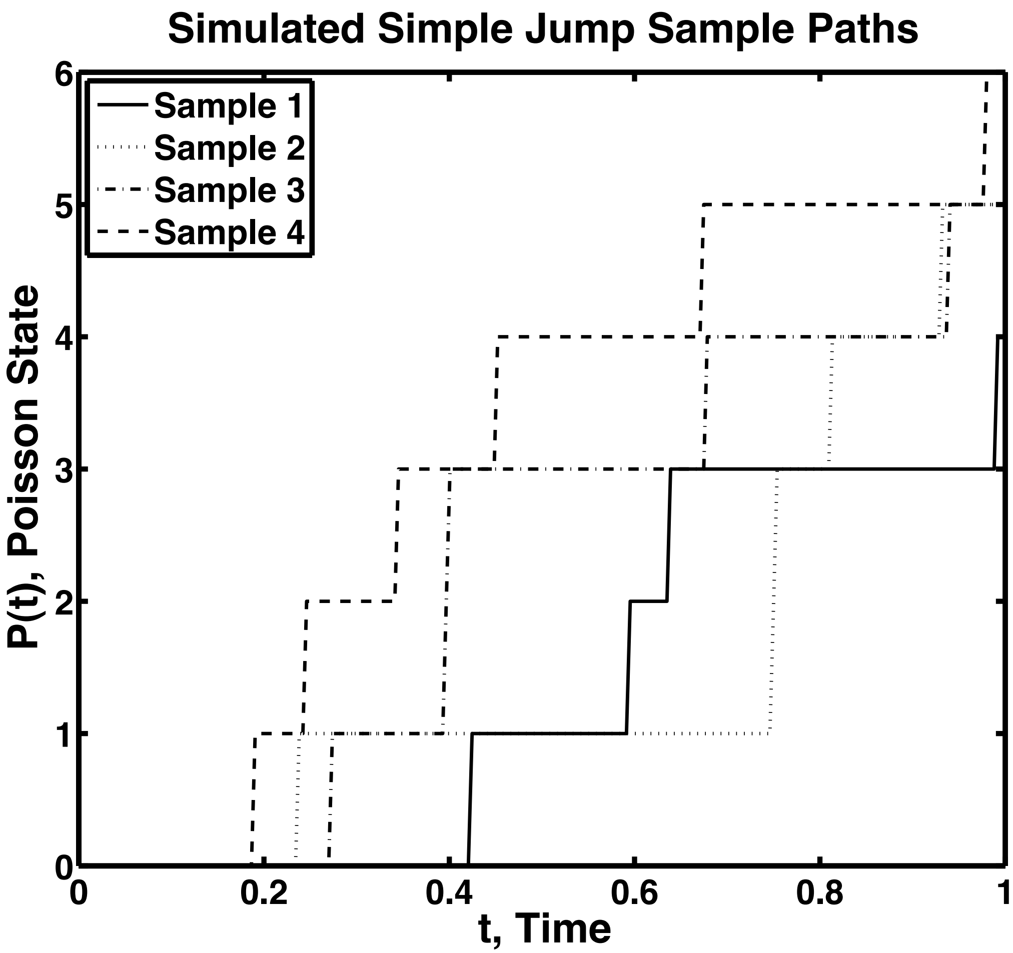


Figure 9: Poisson jump sample paths for four (4) random streams using MATLAB Poisson random generator **poissrnd** with fixed λ and $\Delta t = 1/252$ years. (See also Hanson's (2007) Applied stochastics text, page 15, for older version.)

Poisson Sample Paths from Poisson Jump Times MATLAB code example:

```
function poisson09fig1
% Fig. 1.2b Book Illustration for Incremental (9/2009)
%   Simple Poisson/Jump Process RNG Simulation for
%       Delta{P}(t)=N(t)=P(t+Delta{t})-P(t)=1:K jumps;
%       Time t is unnecessary if Lambda = constant.
%   Generation by MATLAB’s Poisson random generator:
%       DP = poissrnd(Lambda,1,n);
clc % clear variables, but must come before globals.
clf % clear figures, else accumulative.
%
fprintf(’\nfunction poisson09fig1 OutPut:’)
n = 252;dt = 1/n;T = n*dt; % n market days, T  years;
ksamples = 4; marks = {’k-’,’k:’,’k-.’,’k--’};
lambda = 5.241e+00; dLambda = lambda*dt; % histspc88To08
fprintf(’\nn=%i;dt=%7.1e;T=%7.1e;lambda=%7.1e;dLambda=%7.1e;’ .
    ,n,dt,T,lambda,dLambda);
```

```
% Begin Calculation:
P = zeros(ksamples,n+1);
t = 0:dt:T;
nfig = 1;
figure(nfig);
scrsize = get(0,’ScreenSize’); % figure for target screen
ss = [5.0,4.0,3.5]; % figure spacing factors
for ks = 1:ksamples; % Multiple Sample Paths:
    DP = poissrnd(dLambda,1,n);
    P(ks,2:n+1) = cumsum(DP); % P(ks,1) = 0 already;
    plot(t,P(ks,1:n+1),marks{ks},’LineWidth’,2), hold on
end
hold off
fprintf(’\n\nFigure(%i): Jump Sample Paths\n’,nfig);
title(’Simulated Simple Jump Sample Paths’ ...
    ,’FontWeight’,’Bold’,’Fontsize’,24);
ylabel(’P(t), Poisson State’ ...
,’FontWeight’,’Bold’,’Fontsize’,24);
```

```
xlabel('t, Time','FontWeight','Bold','Fontsize',24);
hlegend=legend('Sample 1','Sample 2','Sample 3'...
,'Sample 4','Location','Northwest');
set(hlegend,'Fontsize',20,'FontWeight','Bold');
set(gca,'Fontsize',20,'FontWeight','Bold','linewidth',3);
set(gcf,'Color','White','Position' ...
    ,[scrsize(3)/ss(nfig) 60 scrsize(3)*0.6 scrsize(4)*0.8]);
    % End Code poisson09fig1
```

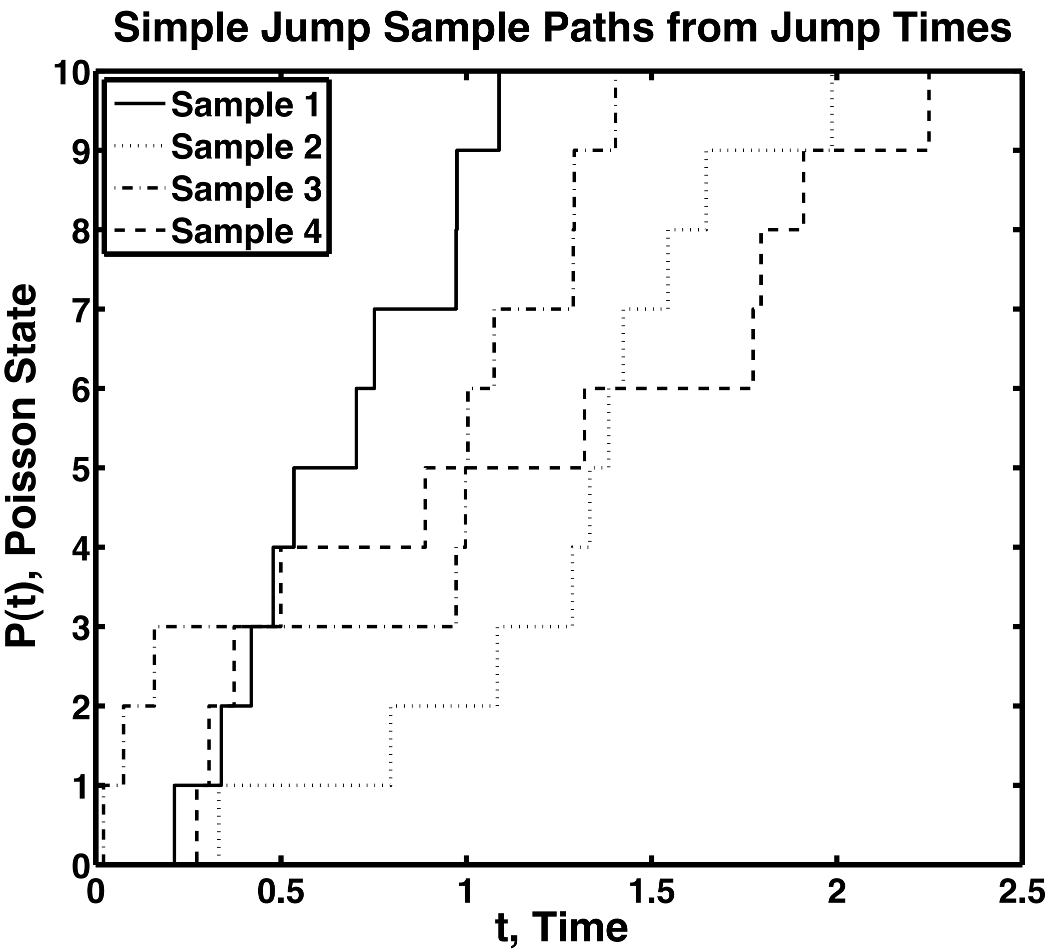


Figure 10: Poisson sample paths for four (4) different time-steps using MATLAB exponential random generator **`exprnd`** for simulated jump times. (See also Hanson's (2007) Applied stochastics text, page 15, for different older version.)

Poisson Sample Paths Using Time-Steps MATLAB code example:

```
function poisson09fig2  % (9/2009)
% Fig. 1.2a Book Illustration for Simple Poisson Process
%   RNG Simulation for the jump times T(j) of P(t).
%   Generation is by Poisson Jump Exponentially dist.
%    increments T(j+1)-T(j), T(j+1) = jth jump time,
%   T(1) == 0.
clc % clear variables, but must come before globals.
clf % clear figures.
%
fprintf(’\nfunction poisson09fig2 OutPut:’)
nfig = 1;
m = 10; me = 2*m; mo =2*m+1; % jump & dual values;
lambda = 5.241e+00; % histspc88To08 est. for lambda;
mue = 1/lambda; % exponential dist. mean is 1/lambda;
ksamples = 4; marks = {’k-’,’k:’,’k-.’,’k--’};
```

```
% Jump-time have dual pre- and post-values of P.
P = zeros(ksamples,mo);
T = zeros(ksamples,mo);
for ks = 1:ksamples; % Multiple exp. dist. Paths:
    DT = exprnd(mue,1,m);
% OR -mue*log(rand(m,1)); if no StatToolbox.
    T(ks,2:2:me) = cumsum(DT); T(ks,3:2:mo) = cumsum(DT);
    P(ks,2:2:me) = 0:m-1; P(ks,3:2:mo) = 1:m;
plot(T(ks,1:mo),P(ks,1:mo),marks{ks},’LineWidth’,2)
hold on
end
hold off
nfig = nfig + 1;
scrsize = get(0,’ScreenSize’); % figure for target screen
ss = [5.0,4.0,3.5]; % figure spacing factors
fprintf(’\n\nFigure(%i):  Simulated Jump Sample Paths\n’...
,nfig)
figure(nfig)
```

```
title('Simple Jump Sample Paths from Jump Times'...
     ,'FontWeight','Bold','Fontsize',24);
xlabel('t, Time','FontWeight','Bold','Fontsize',24);
ylabel('P(t), Poisson State' ...
,'FontWeight','Bold','Fontsize',24);
hlegend=legend('Sample 1','Sample 2','Sample 3' ...
,'Sample 4','Location','Northwest');
set(hlegend,'Fontsize',20,'FontWeight','Bold');
set(gca,'Fontsize',20,'FontWeight','Bold','linewidth',3);
set(gcf,'Color','White','Position' ...
,[scrsize(3)/ss(nfig) 60 scrsize(3)*0.6 scrsize(4)*0.8]);
%%%%%%%%%%%%%%%%%%%%%%
function DT = exprnd(mu,m,n)
% Use if Statistics Toolbox is not available, else %-out;
DT = -mu*log(rand(m,n));
% End Code poisson09fig2
```

- **Poisson Increment Process Moments**:

  1. **Lemma 1.2. Poisson Expectation Sums by Differentiation.** *If* λ *independent of time,*

$$\mathrm{E}[(\Delta P)^m(t)] = e^{-\lambda\Delta t}\sum_{k=0}^{\infty}\frac{(\lambda\Delta t)^k k^m}{k!}$$
$$= \left[e^{-u}\left(u\frac{d}{du}\right)^m e^u\right]\Bigg|_{u=\lambda\Delta t}$$

for $m = 0, 1, 2, \ldots$.

(Comment: The proof is by induction, using properties of the exponential function and its series representation; see Hanson's (2007) text, p. 17, for more information and a Maple code for Poisson moment calculations.)

2. The results for the first few powers are summarized in the following Table:

Table 1.2. Some expected moments (powers) of Poisson increments and their deviations:

m	$\mathrm{E}[(\Delta P)^m(t)]$	$\mathrm{E}[(\Delta P(t)-\lambda\Delta t)^m]$
0	1	—
1	$\lambda\Delta t$	0
2	$\lambda\Delta t(1+\lambda\Delta t)$	$\lambda\Delta t$
3	$\lambda\Delta t(1+3\lambda\Delta t+(\lambda\Delta t)^2)$	$\lambda\Delta t$
4	$\lambda\Delta t(1+7\lambda\Delta t+6(\lambda\Delta t)^2+(\lambda\Delta t)^3)$	$\lambda\Delta t(1+3\lambda\Delta t)$
5	$\lambda\Delta t(1+15\lambda\Delta t+25(\lambda\Delta t)^2+10(\lambda\Delta t)^3+(\lambda\Delta t)^4)$	$\lambda\Delta t(1+10\lambda\Delta t)$

(Comment: In the limit of ***dt*-precision** (as $\boldsymbol{\Delta t \longrightarrow 0}$ then $\boldsymbol{\Delta t \longrightarrow dt}$, keeping only $\mathbf{O}(dt)$) it is easy to guess that the **Poisson process will have an infinite number of moments of $\mathbf{O}(dt)$**, the the Wiener process only has two, not counting the zeroth moment.)

- **Poisson Zero-One Jump Law — Bernoulli Process:**
    1. **Theorem 1.4. $\Delta P(t)$ Zero-One Jump Law Error Magnitude**: As $\Delta t \longrightarrow 0^+$ with constant and bounded λ, then

$$\text{Prob}[\Delta P(t) = 0] = 1 - \lambda\Delta t + \text{O}^2(\lambda\Delta t),$$

$$\text{Prob}[\Delta P(t) = 1] = \lambda\Delta t + \text{O}^2(\lambda\Delta t),$$

$$\text{Prob}[\Delta P(t) > 1] = \text{O}^2(\lambda\Delta t),$$

$$\text{Prob}[(\Delta P)^m(t) = \Delta P(t)] = 1 - \frac{1}{2}(\lambda\Delta t)^2 + \text{O}^3(\lambda\Delta t),$$

$$m \geq 2.$$

(Comment: The proof is by asymptotic expansion by Taylor approximation and, in the last line, relying on the **algebraic zero-one law** that $x^m = x$ only if **x=0** or **x=1**.

This result is the basis for the infinitesimal or short time interval formulation of the Poisson process. However, ***care must be taken NOT to misapply*** the result to situations where the time-step Δt is moderate or more accurately when the product $\lambda\Delta t$ is has a moderate value, invalidating the **1-jump** limit.)

2. **Definition 1.6. Equality to Precision-dt:** *Let f(dt;x) and g(x) be bounded functions for $dt > 0$ and parameter x. The* ***function f is equal to g to precision-dt*** *and write* $f(dt;x) \overset{\text{dt}}{=} g(x)dt$ *if* $f(dt;x) = g(x)dt + \text{o}(dt)$ *as* $dt \longrightarrow 0^+$ *and fixed* x.

   (Comments: A basic condition for much of continuous-time modeling is **precision-dt**. The approximate **precision-Δt** is similarly defined.)

3. **Theorem 1.5. Zero-One Jump Law for dP(t)**: Let $dt > 0$ and $\lambda(t) > 0$ as well also be bounded. Then

$$\begin{aligned}
\mathrm{Prob}[dP(t) = 0] &\overset{\mathrm{dt}}{=} 1 - \lambda(t)dt, \\
\mathrm{Prob}[dP(t) = 1] &\overset{\mathrm{dt}}{=} \lambda(t)dt, \\
\mathrm{Prob}[dP(t) > 1] &\overset{\mathrm{dt}}{=} 0, \\
\mathrm{Prob}[(dP)^2(t) = dP(t)] &\overset{\mathrm{dt}}{=} 1, \\
\mathrm{Prob}[(dP)^m(t) = dP(t)] &\overset{\mathrm{dt}}{=} 1\,, m > 0.
\end{aligned}$$

(Comment: This follows from the previous theorem, except that it is consistent in the infinitesimal case to allow time dependent jump rates $\lambda(t)$.)

4. **Corollary 1.5. Zero-One Jump Law Distribution and Expectation for dP(t)**: *In* ***precision-dt***,

$$\Phi_{dP(t)}(k) = p_k(\lambda(t)dt) \overset{\text{dt}}{=} (1-\lambda(t)dt)\delta_{k,0} + \lambda(t)dt\delta_{k,1}$$

is a generalized representation of the ***differential Poisson distribution*** *and*

$$\mathrm{E}[f(dP(t))] \overset{\text{dt}}{=} (1-\lambda(t)dt)f(0) + \lambda(t)dt f(1)$$

is the expectation, provided ***f(p)*** *is a bounded and continuous function.*

(Comment: The Poisson zero-one jump law is a special case of a **Bernoulli distribution**, concerning Bernoulli trials that have only two outcomes, here with **failure probability** $p = 1 - \lambda(t)\,dt$ for zero jump or **success probability** $1 - p = \lambda(t)\,dt$ for one jump, provided $\lambda(t)dt \ll 1$.)

5. Validity of Zero-One Jump Law?

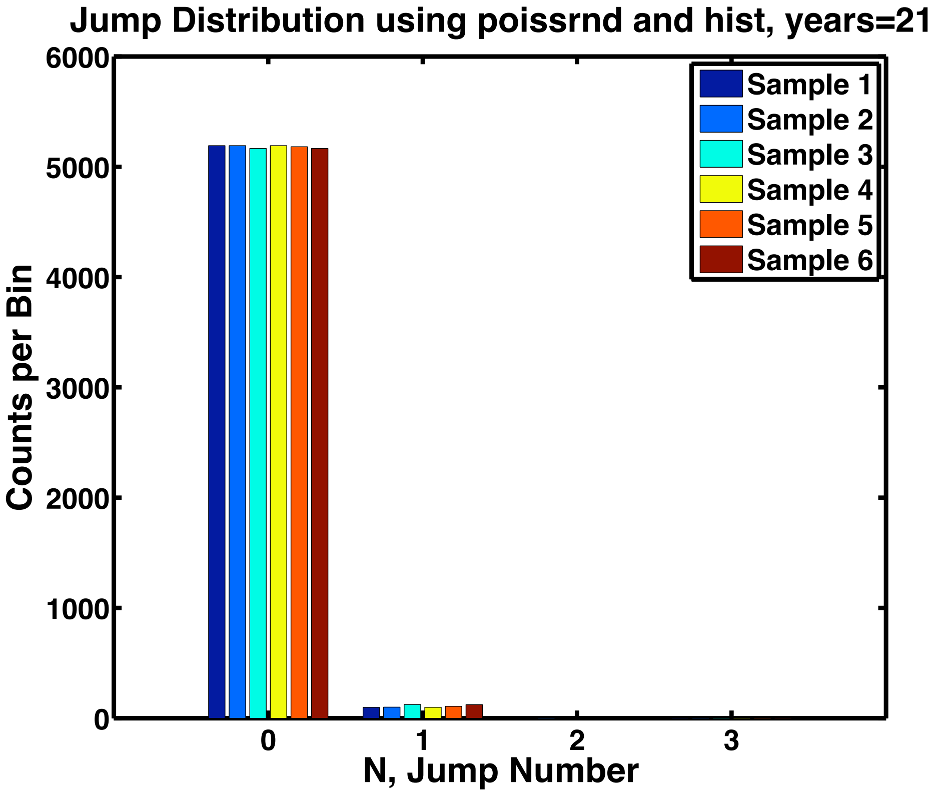


Figure 11: Poisson distribution for **S&P '88-'08** estimate $\lambda = 5.241$ per year, $\Delta t = 1/252$ years, $\Delta\Lambda = 2.151\text{e-}2$ and **n = 5292** `poissrnd` simulations, leading to maximal **2-jump-count = [2,0,1,1,3,2]**, while `poisspdf` predicts $p_k(\Delta\Lambda)$ = **[9.794e-1,2.037e-2,2.118e-4,1.468e-6]** for **k = [0,1,2,3]** jumps.

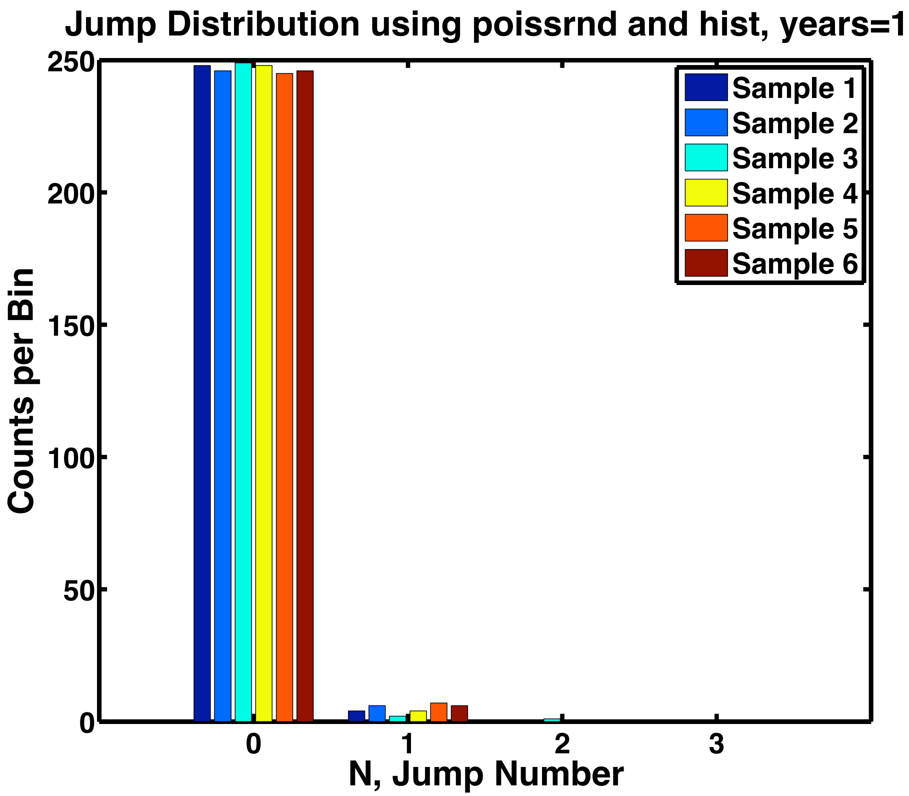


Figure 12: Poisson distribution for **one year of S&P '88-'08** estimate $\lambda = 5.241$ per year, $\Delta t = 1/252$ years, $\Delta\Lambda = 2.151\text{e-}2$ and **n = 252 (for one year only!)** **`poissrnd`** simulations, leading to maximal **2-jump-count =[0,0,1,0,0,0]** so chances of a 2-jump is rare, while **`poisspdf`** predicts $p_k(\Delta\Lambda)$ **= [9.794e-1,2.037e-2,2.118e-4,1.468e-6]** for **k = [0,1,2,3]** jumps.

Summary: So the validity of zero-one jump law for daily observations in financial markets is marginal, since there will be rare two or more jumps that may occur, more so after a long period and less so for a short period such as a year. This is because the fraction of a year in a market day ($\Delta t \simeq 1/252$) is small, but not too small.

However, the zero-one jump law is a reasonable approximation, but not a highly accurate one.

Validity of Zero-One Jump Law MATLAB code example:

```
function poisson09fig3
% Validity of the Zero-One Jump Law for Poisson Processes
clc % clear variables, but must come before globals.
clf % clear figures, else accumulative.
fprintf(’\nfunction poisson09fig3.m OutPut:(%s):\n’...
    ,datestr(now)’);  nfig = 3-1;
scrsize = get(0,’ScreenSize’); % figure spacing: target screen
ss = [5.0,4.0,3.5,3.0]; % figure spacing factors
nsamples = 6; % marks = {’k-’,’k:’,’k-.’,’k--’};
lambda = 5.241e+00; % histspc88To08 for lambda
xc = 0:3; nxc = length(xc); % Begin Calculation:
fprintf(’\nsize(xc)=[%i,%i],length(xc)=%i;’,size(xc),length(xc));
avdays = 252;
for Years = [21,1]
    ndays=Years*avdays;  dt=Years/ndays; %days/year, T years;
    dLambda = lambda*dt;
fprintf(’\nYears=%i;ndays=%i;dt=%7.1e;lambda=%7.1e;dLambda=%7.1e;’...
      ,Years,ndays,dt,lambda,dLambda);
    pk = poisspdf(xc,dLambda);
```

```
fprintf('\nBinCenters=[%9i,%9i,%9i,%9i];',xc);
fprintf('\n        pk=[%9.3e %9.3e %9.3e %9.3e];',pk);
DP = zeros(nsamples,ndays);
nhist = zeros(nsamples,nxc);
for ks = 1:nsamples; % Test Multiple Simulated Sample Paths:
    DP(ks,:) = poissrnd(dLambda,1,ndays);
    fprintf('\nks=%i;max(DP)=%i;sum(DP)=%i;'...
        ,ks,max(DP(ks,:)),sum(DP(ks,:)));
    nhist(ks,:) = hist(DP(ks,:),xc);
    fprintf('\nBinCenters=[%3i,%i,%i,%i];',xc);
    fprintf('\nBinCounts =[%i,%i,%i,%i];',nhist(ks,:));
end
nfig = nfig + 1; figure(nfig);
fprintf('\n2-Jump-Count=[%i,%i,%i,%i,%i,%i];',nhist(:,3));
fprintf('\nFigure(%i):  Simulated Jump Sample Paths\n',nfig);
bar(xc,nhist','grouped'); % rearrange: rows[nhist]=length[xc];
title(['Jump Distribution using poissrnd and hist, years='...
    ,num2str(Years)],'FontWeight','Bold','Fontsize',24);
ylabel('Counts per Bin','FontWeight','Bold','Fontsize',24);
xlabel('N, Jump Number','FontWeight','Bold','Fontsize',24);
hlegend=legend('Sample 1','Sample 2','Sample 3','Sample 4'...
        ,'Sample 5','Sample 6','Location','Northeast');
    set(hlegend,'Fontsize',20,'FontWeight','Bold');
    set(gca,'Fontsize',20,'FontWeight','Bold','linewidth',3);
    set(gcf,'Color','White','Position' ...
        ,[scrsize(3)/ss(nfig) 60 scrsize(3)*0.6 scrsize(4)*0.8]);
end  % End Code poisson09fig3
```

To be Completed in Lecture 2!

1.4. Time-Dependent (NonHomogeneous) Poisson Process:

- Financial markets are very time-dependent, so modelers need to think critically about constant coefficient models, understanding that in some cases time-dependence of coefficients may be difficult to estimate, but perhaps not much more difficult to analyze. Thus, consider $\lambda = \lambda(t)$ so the Poisson process **P(t)** will be **nonstationary**.

- Thus, the Poisson parameter differential is $d\Lambda(t) \equiv \lambda(t)dt$, while the integral parameter, assuming $\Lambda(0) = 0$ as in the constant jump rate case, is

$$\Lambda(t) = \int_0^t \lambda(s)ds.$$

- Then, the Poisson parameter increment is defined by

$$\Delta\Lambda(t) \equiv \Lambda(t+\Delta t) - \Lambda(t) = \int_t^{t+\Delta t} \lambda(s)ds.$$

Thus, $\Delta\Lambda(t) \sim \lambda(t)\Delta t$ only when $\Delta t \ll 1$, i.e., is small, but if not use the integral.

- The temporal Poisson distributions $\mathrm{Prob}[dP(t) = k] = p_k(\Lambda_{[1:3]}(t))$ for the three cases $\Delta P_{[1:3]}(t) = [dP(t), \Delta P(t), P(t)]$ and parameters $\Delta\Lambda_{[1:3]}(t) = [d\Lambda(t), \Delta\Lambda(t), \Lambda(t)]$, are the same

$$\Phi_{\Delta P_i(t)}(k; \Delta\Lambda_i(t)) = e^{-\Delta\Lambda_i(t)} \frac{(\Delta\Lambda_i(t))^k}{k!},$$

for $\mathbf{i = 1{:}3}$ and $\mathbf{k = 0, 1, 2, \ldots}$ jumps, $t \geq 0$ and $\Delta t \geq 0$. (Comment: In MATLAB, $1\!:\!n = [j]_{1\times n}$ is a row-vector.)

- Note that all three Poisson processes are increment processes, even $\Delta P_3(t) = P(t) = P(t) - P(0)$, where $P(0) \equiv 0$. Also, $\Lambda(t)$ is continuous as integrals with $\lambda(t) > 0$ for $t > 0$.
- While the basic statistics for the set of Poisson *increment* processes are similar to the simple constant rate case, i.e., $\mathrm{E}[\Delta P_i(t)] = \Delta\Lambda_i(t) = \mathrm{Var}[\Delta P_i(t)]$. However, the exponential distribution of the interjump times are much more complicated, but see Hanson's (2007), pp. 22-23, and cited background references.

1.5. Martingale Properties of Markov Processes — Expectations Conditioned on the Past:

- **Simple Definition 1.7:** *A* ***martingale M(t)*** *is a stochastic process that principally satisfies*

  $$\mathrm{E}[M(t) \mid M(s), 0 \leq s < t] = M(s),$$

  with some technical side conditions in probability space that ***M(t) is absolutely integrable****, i.e.,* $\mathrm{E}[|M(t)|] < \infty$ *on* ***[0,T]*** *for some finite horizon time* $T < \infty$.

  (Comment: The term Martingale comes from horse racing and abstractly symbolizes a **fair game** since

  $$E[M(t) - M(s) \mid M(s)] = 0, \quad 0 \leq s < t,$$

  i.e., there being no net gain on the average conditioned on past data. Alternately, $E[\Delta M(t) \mid M(t)] = 0, t \geq 0$.)

- **Poisson Examples** (assuming $0 \leq s < t$):

  1. Expanding in increments, $\mathrm{E}[P(t)|P(s)]{=}\mathrm{E}[(P(t)-P(s))+P(s)|P(s)] = \Lambda(t;s)+P(s)$, where $\Lambda(t;s) \equiv \Lambda(t)-\Lambda(s)$ so $P(t)$ is **not a martingale**, but the **zero-mean Poisson**, $\widehat{P}(t) \equiv P(t)-\Lambda(t)$ is a martingale, because $\mathrm{E}[\widehat{P}(t)|\widehat{P}(s)] = \widehat{P}(s)$; hence $\mathrm{E}[\Delta\widehat{P}(t)|\widehat{P}(t)]{=}0$, so implies a fair game.

  2. Again expanding,$\mathrm{E}[P^2(t)|Pt)]{=}\mathrm{E}[((P(t){-}P(s)){+}P(s))^2|P(s)] = \Lambda(t;s)-2P(s)\Lambda(t;s)+P^2(s)$, so $P^2(t)$ **cannot be converted into a martingale** since the cross-term $2P(s)\Lambda(t;s)$ prevents additive separability into **t** and **s** terms.

- **Wiener Examples** (assuming $0 \le s < t$):
  1. Since the Wiener process is a zero mean process **W(t) is a martingale**, i.e., $\mathrm{E}[W(t)|W(s)] = \mathrm{E}[(W(t)-W(s))+W(s)|W(s)] = W(s)$ and $\mathrm{E}[\Delta W(t)|W(t)]=0$ implies a fair game. (*Comment: Zero-meanness helps, but is not sufficient in general. Note also that* $\mathrm{E}[|W(t)|] = \sqrt{2t/\pi} < \sqrt{2T/\pi} < \infty$ *by Table 1.1.*)
  2. Expanding, $\mathrm{E}[W^2(t)|W(s)]{=}\mathrm{E}[((W(t)-W(s))+W(s))^2|W(s)] = (t-s)-2W(s)\cdot 0+W^2(s)$, so rearranging we see that **$(W^2(t)-t)$ is a martingale**, $0 \le t < T < \infty$, since $\mathrm{E}[W^2(t)-t|W(s)] = W^2(s)-s$. (Comment: Note no time-dependent coefficients.)